CHAPTER 5

Cyclic Codes

BINARY FIELD ARITHMETIC

In general, we can construct codes with symbols from any Galois field $GF(q)$, where q is either a prime p or a power of p; however, codes with symbols from the binary field $GF(2)$ or its extension $GF(2^m)$ are most widely used in digital data transmission and storage systems because information in these systems is universally coded in binary form for practical reasons. In this text we are concerned only with binary codes and codes with symbols from the field $GF(2^m)$. Most of the results presented in this text can be generalized to codes with symbols from any finite field $GF(q)$ with $q \neq 2$ or 2^m. In this section we discuss arithmetic over the binary field $GF(2)$, which will be used in the remainder text of this book.

In binary arithmetic we use modulo-2 addition and multiplication, which are defined by Tables 2.3 and 2.4, respectively. This arithmetic is actually equivalent to ordinary arithmetic, except that we consider 2 to be equal to 0 (i.e., $1+1=2=0$). Note that since $1+1=0$, $1=-1$. Hence, in binary arithmetic, subtraction is the same as addition. To illustrate how the ideas of ordinary algebra can be used with binary arithmetic, we consider the following set of equations:

$$X+Y=1,$$
$$X+Z=0,$$
$$X+Y+Z=1.$$

These can be solved by adding the first equation to the third, giving $Z=0$. Then, from the second equation, since $Z=0$ and $X+Z=0$, we obtain $X=0$. From the first equation, since $X=0$ and $X+Y=1$, we have $Y=1$. We can substitute these solutions into the original set of equations and verify that they are correct.

Because we were able to solve the preceding equations, they must be linearly independent, and the determinant of the coefficients on the left side must be nonzero. If the determinant is nonzero, it must be 1. This result can be verified as follows.

$$\begin{vmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{vmatrix} = 1 \cdot \begin{vmatrix} 0 & 1 \\ 1 & 1 \end{vmatrix} - 1 \cdot \begin{vmatrix} 1 & 1 \\ 1 & 1 \end{vmatrix} + 0 \cdot \begin{vmatrix} 1 & 0 \\ 1 & 1 \end{vmatrix}$$
$$= 1 \cdot 1 - 1 \cdot 0 + 0 \cdot 1 = 1.$$

We could have solved the equations by Cramer's rule:

$$X = \frac{\begin{vmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{vmatrix}}{\begin{vmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{vmatrix}} = \frac{0}{1} = 0, \quad Y = \frac{\begin{vmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{vmatrix}}{\begin{vmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{vmatrix}} = \frac{1}{1} = 1, \quad Z = \frac{\begin{vmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \end{vmatrix}}{\begin{vmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{vmatrix}} = \frac{0}{1} = 0.$$

Polynomials:

THEOREM 2.10 Any irreducible polynomial over $GF(2)$ of degree m divides $X^{2^m-1}+1$.

As an example of Theorem 2.10, we can check that X^3+X+1 divides $X^{2^3-1}+1 = X^7+1$:

$$
\begin{array}{r|l}
X^3+X+1 & X^4+X^2+X+1 \\
\hline
 & X^7 \qquad\qquad\qquad\qquad\qquad\qquad +1 \\
 & X^7 + X^5 + X^4 \\
\hline
 & X^5 + X^4 \qquad\qquad\qquad +1 \\
 & X^5 \quad + X^3 + X^2 \\
\hline
 & X^4 + X^3 + X^2 \quad +1 \\
 & X^4 \quad\quad + X^2 + X \\
\hline
 & X^3 \quad\quad + X + 1 \\
 & X^3 \quad\quad + X + 1 \\
\hline
 & 0.
\end{array}
$$

An irreducible polynomial $p(X)$ of degree m is said to be *primitive* if the smallest positive integer n for which $p(X)$ divides $X^n + 1$ is $n = 2^m - 1$. We may check that $p(X) = X^4 + X + 1$ divides $X^{15} + 1$ but does not divide any $X^n + 1$ for $1 \leq n < 15$. Hence, $X^4 + X + 1$ is a primitive polynomial. The polynomial $X^4 + X^3 + X^2 + X + 1$ is irreducible but it is not primitive, since it divides $X^5 + 1$. It is not easy to recognize a primitive polynomial; however, there are tables of irreducible polynomials in which primitive polynomials are indicated [6, 8]. For a given m, there may be more than one primitive polynomial of degree m. A list of primitive polynomials is given in Table 2.7. For each degree m, we list only a primitive polynomial with the smallest number of terms.

TABLE 2.7: List of primitive polynomials.

m		m	
3	$1 + X + X^3$	14	$1 + X + X^6 + X^{10} + X^{14}$
4	$1 + X + X^4$	15	$1 + X + X^{15}$
5	$1 + X^2 + X^5$	16	$1 + X + X^3 + X^{12} + X^{16}$
6	$1 + X + X^6$	17	$1 + X^3 + X^{17}$
7	$1 + X^3 + X^7$	18	$1 + X^7 + X^{18}$
8	$1 + X^2 + X^3 + X^4 + X^8$	19	$1 + X + X^2 + X^5 + X^{19}$
9	$1 + X^4 + X^9$	20	$1 + X^3 + X^{20}$
10	$1 + X^3 + X^{10}$	21	$1 + X^2 + X^{21}$
11	$1 + X^2 + X^{11}$	22	$1 + X + X^{22}$
12	$1 + X + X^4 + X^6 + X^{12}$	23	$1 + X^5 + X^{23}$
13	$1 + X + X^3 + X^4 + X^{13}$	24	$1 + X + X^2 + X^7 + X^{24}$

Comments:

Before leaving this section, we derive another useful property of polynomials over $GF(2)$. Consider

$$\begin{aligned} f^2(X) &= (f_0 + f_1X + \cdots + f_nX^n)^2 \\ &= [f_0 + (f_1X + f_2X^2 + \cdots + f_nX^n)]^2 \\ &= f_0^2 + f_0 \cdot (f_1X + f_2X^2 + \cdots + f_nX^n) \\ &\quad + f_0 \cdot (f_1X + f_2X^2 + \cdots + f_nX^n) + (f_1X + f_2X^2 + \cdots + f_nX^n)^2 \\ &= f_0^2 + (f_1X + f_2X^2 + \cdots + f_nX^n)^2. \end{aligned}$$

Expanding the preceding equation repeatedly, we eventually obtain

$$f^2(X) = f_0^2 + (f_1X)^2 + (f_2X^2)^2 + \cdots + (f_nX^n)^2.$$

Since $f_i = 0$ or 1, $f_i^2 = f_i$. Hence, we have

$$\begin{aligned} f^2(X) &= f_0 + f_1X^2 + f_2(X^2)^2 + \cdots + f_n(X^2)^n \\ &= f(X^2). \end{aligned} \tag{2.9}$$

It follows from (2.9) that, for any $i \geq 0$,

$$[f(X)]^{2^i} = f(X^{2^i}). \tag{2.10}$$

CONSTRUCTION OF GALOIS FIELD $GF(2^m)$

In this section we present a method for constructing the Galois field of 2^m elements ($m > 1$) from the binary field $GF(2)$. We begin with the two elements 0 and 1 from

$GF(2)$ and a new symbol α. Then, we define a multiplication "$\cdot$" to introduce a sequence of powers of α as follows:

$$
\begin{aligned}
&0 \cdot 0 = 0, \\
&0 \cdot 1 = 1 \cdot 0 = 0, \\
&1 \cdot 1 = 1, \\
&0 \cdot \alpha = \alpha \cdot 0 = 0, \\
&1 \cdot \alpha = \alpha \cdot 1 = \alpha, \\
&\alpha^2 = \alpha \cdot \alpha, \\
&\alpha^3 = \alpha \cdot \alpha \cdot \alpha, \\
&\quad \vdots \\
&\alpha^j = \alpha \cdot \alpha \cdot \cdots \cdot \alpha \ (j \text{ times}), \\
&\quad \vdots
\end{aligned}
\tag{2.11}
$$

It follows from the preceding definition of multiplication that

$$\begin{aligned} 0 \cdot \alpha^j &= \alpha^j \cdot 0 = 0, \\ 1 \cdot \alpha^j &= \alpha^j \cdot 1 = \alpha^j, \\ \alpha^i \cdot \alpha^j &= \alpha^j \cdot \alpha^i = \alpha^{i+j}. \end{aligned} \tag{2.12}$$

Now, we have the following set of elements on which a multiplication operation "$\cdot$" is defined:

$$F = \{0, 1, \alpha, \alpha^2, \cdots, \alpha^j, \cdots\}.$$

The element 1 is sometimes denoted by α^0.

Next, we put a condition on the element α so that the set F contains only 2^m elements and is closed under the multiplication "$\cdot$" defined by (2.11). Let $p(X)$ be a primitive polynomial of degree m over $GF(2)$. We assume that $p(\alpha) = 0$ (i.e., α is a root of $p(X)$). Since $p(X)$ divides $X^{2^m-1} + 1$ (Theorem 2.10) we have

$$X^{2^m-1} + 1 = q(X)p(X). \tag{2.13}$$

If we replace X with α in (2.13), we obtain

$$\alpha^{2^m-1} + 1 = q(\alpha)p(\alpha).$$

Because $p(\alpha) = 0$, we have

$$\alpha^{2^m-1} + 1 = q(\alpha) \cdot 0.$$

If we regard $q(\alpha)$ as a polynomial of α over $GF(2)$, it follows from (2.7) that $q(\alpha) \cdot 0 = 0$. As a result, we obtain the following equality:

$$\alpha^{2^m-1} + 1 = 0.$$

Adding 1 to both sides of $\alpha^{2^m-1} + 1 = 0$ (using modulo-2 addition), we obtain the following equality:

$$\alpha^{2^m-1} = 1. \tag{2.14}$$

Therefore, under the condition that $p(\alpha) = 0$, the set F becomes finite and contains the following elements:

$$F^* = \{0, 1, \alpha, \alpha^2, \cdots, \alpha^{2^m-2}\}.$$

The nonzero elements of F^* are closed under the multiplication operation "$\cdot$" defined by (2.11). To see this, let i and j be two integers such that $0 \leq i, \ j < 2^m - 1$. If $i + j < 2^m - 1$, then $\alpha^i \cdot \alpha^j = \alpha^{i+j}$, which is obviously a nonzero element in F^*. If $i + j \geq 2^m - 1$, we can express $i + j$ as follows: $i + j = (2^m - 1) + r$, where $0 \leq r < 2^m - 1$. Then,

$$\alpha^i \cdot \alpha^j = \alpha^{i+j} = \alpha^{(2^m-1)+r} = \alpha^{2^m-1} \cdot \alpha^r = \alpha^r,$$

which is also a nonzero element in F^*. Hence, we conclude that the nonzero elements of F^* are closed under the multiplication "$\cdot$" defined by (2.11). In fact, these nonzero elements form a commutative group under "$\cdot$". First, we see that the element 1 is the unit element. From (2.11) and (2.12) we see readily that the multiplication operation "$\cdot$" is commutative and associative. For $0 < i < 2^m - 1$, α^{2^m-i-1} is the multiplicative inverse of α^i, since

$$\alpha^{2^m-i-1} \cdot \alpha^i = \alpha^{2^m-1} = 1.$$

(Note that $\alpha^0 = \alpha^{2^m-1} = 1$.) It will be clear in the discussion that follows that $1, \alpha, \alpha^2, \cdots, \alpha^{2^m-2}$ represent $2^m - 1$ distinct elements. Therefore, the nonzero elements of F^* form a commutative group of order $2^m - 1$ under the multiplication operation "$\cdot$" defined by (2.11).

Comments:

EXAMPLE 2.7

Let $m = 4$. The polynomial $p(X) = 1 + X + X^4$ is a primitive polynomial over $GF(2)$. Set $p(\alpha) = 1 + \alpha + \alpha^4 = 0$. Then, $\alpha^4 = 1 + \alpha$. Using this relation, we can construct $GF(2^4)$. The elements of $GF(2^4)$ are given in Table 2.8. The identity $\alpha^4 = 1 + \alpha$ is used repeatedly to form the polynomial representations for the elements of $GF(2^4)$. For example,

$$\alpha^5 = \alpha \cdot \alpha^4 = \alpha(1 + \alpha) = \alpha + \alpha^2,$$
$$\alpha^6 = \alpha \cdot \alpha^5 = \alpha(\alpha + \alpha^2) = \alpha^2 + \alpha^3,$$
$$\alpha^7 = \alpha \cdot \alpha^6 = \alpha(\alpha^2 + \alpha^3) = \alpha^3 + \alpha^4 = \alpha^3 + 1 + \alpha = 1 + \alpha + \alpha^3.$$

To multiply two elements α^i and α^j, we simply add their exponents and use the fact that $\alpha^{15} = 1$. For example, $\alpha^5 \cdot \alpha^7 = \alpha^{12}$, and $\alpha^{12} \cdot \alpha^7 = \alpha^{19} = \alpha^4$. Dividing α^j by α^i, we simply multiply α^j by the multiplicative inverse α^{15-i} of α^i. For example, $\alpha^4/\alpha^{12} = \alpha^4 \cdot \alpha^3 = \alpha^7$, and $\alpha^{12}/\alpha^5 = \alpha^{12} \cdot \alpha^{10} = \alpha^{22} = \alpha^7$. To add α^i and α^j, we use their polynomial representations given in Table 2.8. Thus,

$$\alpha^5 + \alpha^7 = (\alpha + \alpha^2) + (1 + \alpha + \alpha^3) = 1 + \alpha^2 + \alpha^3 = \alpha^{13},$$
$$1 + \alpha^5 + \alpha^{10} = 1 + (\alpha + \alpha^2) + (1 + \alpha + \alpha^2) = 0.$$

There is another useful representation for the field elements in $GF(2^m)$. Let $a_0 + a_1\alpha + a_2\alpha^2 + \cdots + a_{m-1}\alpha^{m-1}$ be the polynomial representation of a field element β. Then, we can represent β by an ordered sequence of m components called an *m-tuple*, as follows:

$$(a_0, a_1, a_2, \cdots, a_{m-1}),$$

where the m components are simply the m coefficients of the polynomial representation of β. Clearly, we see that there is one-to-one correspondence between this m-tuple and the polynomial representation of β. The zero element 0 of $GF(2^m)$ is represented by the zero m-tuple $(0, 0, \cdots, 0)$. Let $(b_0, b_1, \cdots, b_{m-1})$ be the m-tuple

representation of γ in $GF(2^m)$. Adding β and γ, we simply add the corresponding components of their m-tuple representations as follows:

$$(a_0 + b_0, a_1 + b_1, \cdots, a_{m-1} + b_{m-1}),$$

where $a_i + b_i$ is carried out in modulo-2 addition. Obviously, the components of the resultant m-tuple are the coefficients of the polynomial representation for $\beta + \gamma$. All three representations for the elements of $GF(2^4)$ are given in Table 2.8.

Galois fields of 2^m elements with $m = 3$ to 10 are given in Appendix A.

TABLE 2.8: Three representations for the elements of $GF(2^4)$ generated by $p(X) = 1 + X + X^4$.

Power representation	Polynomial representation	4-Tuple representation
0	0	(0 0 0 0)
1	1	(1 0 0 0)
α	α	(0 1 0 0)
α^2	α^2	(0 0 1 0)
α^3	α^3	(0 0 0 1)
α^4	$1 + \alpha$	(1 1 0 0)
α^5	$\alpha + \alpha^2$	(0 1 1 0)
α^6	$\alpha^2 + \alpha^3$	(0 0 1 1)
α^7	$1 + \alpha + \alpha^3$	(1 1 0 1)
α^8	$1 + \alpha^2$	(1 0 1 0)
α^9	$\alpha + \alpha^3$	(0 1 0 1)
α^{10}	$1 + \alpha + \alpha^2$	(1 1 1 0)
α^{11}	$\alpha + \alpha^2 + \alpha^3$	(0 1 1 1)
α^{12}	$1 + \alpha + \alpha^2 + \alpha^3$	(1 1 1 1)
α^{13}	$1 + \alpha^2 + \alpha^3$	(1 0 1 1)
α^{14}	$1 + \alpha^3$	(1 0 0 1)

BASIC PROPERTIES OF A GALOIS FIELD $GF(2^m)$

In ordinary algebra we often see that a polynomial with real coefficients has roots not from the field of real numbers but from the field of complex numbers that contains the field of real numbers as a subfield. For example, the polynomial $X^2+6X+25$ does not have roots from the field of real numbers but has two complex-conjugate roots, $-3 + 4i$ and $-3 - 4i$, where $i = \sqrt{-1}$. This situation is also true for polynomials with coefficients from $GF(2)$. In this case, a polynomial with coefficients from $GF(2)$ may not have roots from $GF(2)$ but has roots from an extension field of $GF(2)$. For example, $X^4 + X^3 + 1$ is irreducible over $GF(2)$ and therefore it does not have roots from $GF(2)$; however, it has four roots from the field $GF(2^4)$. If we substitute the elements of $GF(2^4)$ given by Table 2.8 into $X^4 + X^3 + 1$, we find that α^7, α^{11}, α^{13}, and α^{14} are the roots of $X^4 + X^3 + 1$. We may verify this result as follows:

$$
\begin{aligned}
&(X + \alpha^7)(X + \alpha^{11})(X + \alpha^{13})(X + \alpha^{14}) \\
&\quad = [X^2 + (\alpha^7 + \alpha^{11})X + \alpha^{18}][X^2 + (\alpha^{13} + \alpha^{14})X + \alpha^{27}] \\
&\quad = (X^2 + \alpha^8 X + \alpha^3)(X^2 + \alpha^2 X + \alpha^{12}) \\
&\quad = X^4 + (\alpha^8 + \alpha^2)X^3 + (\alpha^{12} + \alpha^{10} + \alpha^3)X^2 + (\alpha^{20} + \alpha^5)X + \alpha^{15} \\
&\quad = X^4 + X^3 + 1.
\end{aligned}
$$

THEOREM 2.11 Let $f(X)$ be a polynomial with coefficients from $GF(2)$. Let β be an element in an extension field of $GF(2)$. If β is a root of $f(X)$, then for any $l \geq 0$, β^{2^l} is also a root of $f(X)$.

Proof. From (2.10), we have

$$[f(X)]^{2^l} = f(X^{2^l}).$$

Substituting β into the preceding equation, we obtain

$$[f(\beta)]^{2^l} = f(\beta^{2^l}).$$

Since $f(\beta) = 0$, $f(\beta^{2^l}) = 0$. Therefore, β^{2^l} is also a root of $f(X)$. **Q.E.D.**

The element β^{2^l} is called a *conjugate* of β.

$$f(X) = 1 + X^3 + X^4 + X^5 + X^6$$

$$\begin{aligned} f(\alpha^4) &= 1 + \alpha^{12} + \alpha^{16} + \alpha^{20} + \alpha^{24} = 1 + \alpha^{12} + \alpha + \alpha^5 + \alpha^9 \\ &= 1 + (1 + \alpha + \alpha^2 + \alpha^3) + \alpha + (\alpha + \alpha^2) + (\alpha + \alpha^3) = 0. \end{aligned}$$

The conjugates of α^4 are

$$(\alpha^4)^2 = \alpha^8, \quad (\alpha^4)^{2^2} = \alpha^{16} = \alpha, \quad (\alpha^4)^{2^3} = \alpha^{32} = \alpha^2.$$

[Note that $(\alpha^4)^{2^4} = \alpha^{64} = \alpha^4$.] It follows from Theorem 2.11 that α^8, α, and α^2 must also be roots of $f(X) = 1 + X^3 + X^4 + X^5 + X^6$. We can check that α^5 and its conjugate, α^{10}, are roots of $f(X) = 1 + X^3 + X^4 + X^5 + X^6$. Therefore, $f(X) = 1 + X^3 + X^4 + X^5 + X^6$ has six distinct roots in $GF(2^4)$.

Let β be a nonzero element in the field $GF(2^m)$. It follows from Theorem 2.8 that

$$\beta^{2^m-1} = 1.$$

Adding 1 to both sides of $\beta^{2^m-1} = 1$, we obtain

$$\beta^{2^m-1} + 1 = 0.$$

This says that β is a root of the polynomial $X^{2^m-1} + 1$. Hence, every nonzero element of $GF(2^m)$ is a root of $X^{2^m-1} + 1$. Because the degree of $X^{2^m-1} + 1$ is $2^m - 1$, the $2^m - 1$ nonzero elements of $GF(2^m)$ form all the roots of $X^{2^m-1} + 1$. Summarizing the preceding result, we obtain Theorem 2.12.

THEOREM 2.12 The $2^m - 1$ nonzero elements of $GF(2^m)$ form all the roots of $X^{2^m-1} + 1$.

Since the zero element 0 of $GF(2^m)$ is the root of X, Theorem 2.12 has the following corollary:

COROLLARY 2.12.1 The elements of $GF(2^m)$ form all the roots of $X^{2^m} + X$.

Because any element β in $GF(2^m)$ is a root of the polynomial $X^{2^m} + X$, β may be a root of a polynomial over $GF(2)$ with a degree less than 2^m. Let $\phi(X)$ be the polynomial of *smallest degree* over $GF(2)$ such that $\phi(\beta) = 0$. [We can easily prove that $\phi(X)$ is unique.] This polynomial $\phi(X)$ is called the *minimal polynomial of β*. For example, the minimal polynomial of the zero element 0 of $GF(2^m)$ is X, and the minimal polynomial of the unit element 1 is $X + 1$. Next, we derive a number of properties of minimal polynomials.

THEOREM 2.13 The minimal polynomial $\phi(X)$ of a field element β irreducible.

THEOREM 2.14 Let $f(X)$ be a polynomial over $GF(2)$. Let $\phi(X)$ be the minimal polynomial of a field element β. If β is a root of $f(X)$, then $f(X)$ is divisible by $\phi(X)$.

THEOREM 2.15 The minimal polynomial $\phi(X)$ of an element β in $GF(2^m)$ divides $X^{2^m} + X$.

Theorem 2.15 says that all the roots of $\phi(X)$ are from $GF(2^m)$. Then, what are the roots of $\phi(X)$? This question is answered by the next two theorems.

THEOREM 2.16 Let $f(X)$ be an irreducible polynomial over $GF(2)$. Let β be an element in $GF(2^m)$. Let $\phi(X)$ be the minimal polynomial of β. If $f(\beta) = 0$, then $\phi(X) = f(X)$.

Proof. It follows from Theorem 2.14 that $\phi(X)$ divides $f(X)$. Since $\phi(X) \neq 1$ and $f(X)$ is irreducible, we must have $\phi(X) = f(X)$. **Q.E.D.**

Theorem 2.16 says that if an irreducible polynomial has β as a root, it is the minimal polynomial $\phi(X)$ of β. It follows from Theorem 2.11 that β and its conjugates $\beta^2, \beta^{2^2}, \cdots, \beta^{2^l}, \cdots$ are roots of $\phi(X)$. Let e be the smallest integer such that $\beta^{2^e} = \beta$. Then, $\beta^2, \beta^{2^2}, \cdots, \beta^{2^{e-1}}$ are all the distinct conjugates of β (see Problem 2.15). Since $\beta^{2^m} = \beta$, $e \leq m$ (in fact e divides m).

THEOREM 2.17 Let β be an element in $GF(2^m)$, and let e be the smallest nonnegative integer such that $\beta^{2^e} = \beta$. Then,

$$f(X) = \prod_{i=0}^{e-1} (X + \beta^{2^i})$$

is an irreducible polynomial over $GF(2)$.

THEOREM 2.18 Let $\phi(X)$ be the minimal polynomial of an element β in $GF(2^m)$. Let e be the smallest integer such that $\beta^{2^e} = \beta$. Then

$$\phi(X) = \prod_{i=0}^{e-1} (X + \beta^{2^i}). \tag{2.23}$$

EXAMPLE 2.8

Consider the Galois field $GF(2^4)$ given by Table 2.8. Let $\beta = \alpha^3$. The conjugates of β are

$$\beta^2 = \alpha^6, \quad \beta^{2^2} = \alpha^{12}, \quad \beta^{2^3} = \alpha^{24} = \alpha^9.$$

The minimal polynomial of $\beta = \alpha^3$ is then

$$\phi(X) = (X + \alpha^3)(X + \alpha^6)(X + \alpha^{12})(X + \alpha^9).$$

Multiplying out the right-hand side of the preceding equation with the aid of Table 2.8, we obtain

$$\begin{aligned}
\phi(X) &= [X^2 + (\alpha^3 + \alpha^6)X + \alpha^9][X^2 + (\alpha^{12} + \alpha^9)X + \alpha^{21}] \\
&= (X^2 + \alpha^2 X + \alpha^9)(X^2 + \alpha^8 X + \alpha^6) \\
&= X^4 + (\alpha^2 + \alpha^8)X^3 + (\alpha^6 + \alpha^{10} + \alpha^9)X^2 + (\alpha^{17} + \alpha^8)X + \alpha^{15} \\
&= X^4 + X^3 + X^2 + X + 1.
\end{aligned}$$

TABLE 2.8: Three representations for the elements of $GF(2^4)$ generated by $p(X) = 1 + X + X^4$.

Power representation	Polynomial representation	4-Tuple representation
0	0	(0 0 0 0)
1	1	(1 0 0 0)
α	α	(0 1 0 0)
α^2	α^2	(0 0 1 0)
α^3	α^3	(0 0 0 1)
α^4	$1 + \alpha$	(1 1 0 0)
α^5	$\alpha + \alpha^2$	(0 1 1 0)
α^6	$\alpha^2 + \alpha^3$	(0 0 1 1)
α^7	$1 + \alpha + \alpha^3$	(1 1 0 1)
α^8	$1 + \alpha^2$	(1 0 1 0)
α^9	$\alpha + \alpha^3$	(0 1 0 1)
α^{10}	$1 + \alpha + \alpha^2$	(1 1 1 0)
α^{11}	$\alpha + \alpha^2 + \alpha^3$	(0 1 1 1)
α^{12}	$1 + \alpha + \alpha^2 + \alpha^3$	(1 1 1 1)
α^{13}	$1 + \alpha^2 + \alpha^3$	(1 0 1 1)
α^{14}	$1 + \alpha^3$	(1 0 0 1)

EXAMPLE 2.9

Suppose that we want to determine the minimal polynomial $\phi(X)$ of $\gamma = \alpha^7$ in $GF(2^4)$. The distinct conjugates of γ are

$$\gamma^2 = \alpha^{14}, \quad \gamma^{2^2} = \alpha^{28} = \alpha^{13}, \quad \gamma^{2^3} = \alpha^{56} = \alpha^{11}.$$

Hence, $\phi(X)$ has degree 4 and must be of the following form:

$$\phi(X) = a_0 + a_1 X + a_2 X^2 + a_3 X^3 + X^4.$$

Substituting γ into $\phi(X)$, we have

$$\phi(\gamma) = a_0 + a_1\gamma + a_2\gamma^2 + a_3\gamma^3 + \gamma^4 = 0.$$

Using the polynomial representations for γ, γ^2, γ^3, and γ^4 in the preceding equation, we obtain the following:

$$a_0 + a_1(1 + \alpha + \alpha^3) + a_2(1 + \alpha^3) + a_3(\alpha^2 + \alpha^3) + (1 + \alpha^2 + \alpha^3) = 0$$

$$(a_0 + a_1 + a_2 + 1) + a_1\alpha + (a_3 + 1)\alpha^2 + (a_1 + a_2 + a_3 + 1)\alpha^3 = 0.$$

For the preceding equality to be true, the coefficients must equal zero:

$$\begin{aligned} a_0 + a_1 + a_2 \qquad + 1 &= 0, \\ a_1 \qquad\qquad\quad &= 0, \\ a_3 + 1 &= 0, \\ a_1 + a_2 + a_3 + 1 &= 0. \end{aligned}$$

Solving the preceding linear equations, we obtain $a_0 = 1$, $a_1 = a_2 = 0$, and $a_3 = 1$. Therefore, the minimal polynomial of $\gamma = \alpha^7$ is $\phi(X) = 1 + X^3 + X^4$. All the minimal polynomials of the elements in $GF(2^4)$ are given by Table 2.9.

TABLE 2.9: Minimal polynomials of the elements in $GF(2^4)$ generated by $p(X) = X^4 + X + 1$.

Conjugate roots	Minimal polynomials
0	X
1	$X + 1$
$\alpha, \alpha^2, \alpha^4, \alpha^8$	$X^4 + X + 1$
$\alpha^3, \alpha^6, \alpha^9, \alpha^{12}$	$X^4 + X^3 + X^2 + X + 1$
α^5, α^{10}	$X^2 + X + 1$
$\alpha^7, \alpha^{11}, \alpha^{13}, \alpha^{14}$	$X^4 + X^3 + 1$

THEOREM 2.19 Let $\phi(X)$ be the minimal polynomial of an element β in $GF(2^m)$. Let e be the degree of $\phi(X)$. Then e is the smallest integer such that $\beta^{2^e} = \beta$. Moreover, $e \leq m$.

THEOREM 2.20 If β is a primitive element of $GF(2^m)$, all its conjugates $\beta^2, \beta^{2^2}, \cdots$ are also primitive elements of $GF(2^m)$.

In the construction of the Galois field $GF(2^m)$ we use a primitive polynomial $p(X)$ of degree m and require that the element α be a root of $p(X)$. Because the powers of α generate all the nonzero elements of $GF(2^m)$, α is a primitive element. In fact, all the conjugates of α are primitive elements of $GF(2^m)$. To see this, let n be the order of α^{2^l} for $l > 0$. Then

$$(\alpha^{2^l})^n = \alpha^{n2^l} = 1.$$

Also, it follows from Theorem 2.9 that n divides $2^m - 1$:

$$2^m - 1 = k \cdot n. \tag{2.24}$$

Because α is a primitive element of $GF(2^m)$, its order is $2^m - 1$. For $\alpha^{n2^l} = 1$, $n2^l$ must be a multiple of $2^m - 1$. Since 2^l and $2^m - 1$ are relatively prime, n must be divisible by $2^m - 1$, say

$$n = q \cdot (2^m - 1). \tag{2.25}$$

From (2.24) and (2.25) we conclude that $n = 2^m - 1$. Consequently, α^{2^l} is also a primitive element of $GF(2^m)$. In general, we have the following theorem.

EXAMPLE 2.10

Consider the field $GF(2^4)$ given by Table 2.8. The powers of $\beta = \alpha^7$ are

$$\beta^0 = 1,\ \beta^1 = \alpha^7,\ \beta^2 = \alpha^{14},\ \beta^3 = \alpha^{21} = \alpha^6,\ \beta^4 = \alpha^{28} = \alpha^{13},$$
$$\beta^5 = \alpha^{35} = \alpha^5,\ \beta^6 = \alpha^{42} = \alpha^{12},\ \beta^7 = \alpha^{49} = \alpha^4,\ \beta^8 = \alpha^{56} = \alpha^{11},$$
$$\beta^9 = \alpha^{63} = \alpha^3,\ \beta^{10} = \alpha^{70} = \alpha^{10},\ \beta^{11} = \alpha^{77} = \alpha^2,\ \beta^{12} = \alpha^{84} = \alpha^9,$$
$$\beta^{13} = \alpha^{91} = \alpha,\ \beta^{14} = \alpha^{98} = \alpha^8,\ \beta^{15} = \alpha^{105} = 1.$$

Clearly, the powers of $\beta = \alpha^7$ generate all the nonzero elements of $GF(2^4)$, so $\beta = \alpha^7$ is a primitive element of $GF(2^7)$. The conjugates of $\beta = \alpha^7$ are

$$\beta^2 = \alpha^{14},\quad \beta^{2^2} = \alpha^{13},\quad \beta^{2^3} = \alpha^{11}.$$

We may readily check that they are all primitive elements of $GF(2^m)$.

THEOREM 2.21 If β is an element of order n in $GF(2^m)$, all its conjugates have the same order n. (The proof is left as an exercise.)

EXAMPLE 2.11

Consider the element α^5 in $GF(2^4)$ given by Table 2.8. Since $(\alpha^5)^{2^2} = \alpha^{20} = \alpha^5$, the only conjugate of α^5 is α^{10}. Both α^5 and α^{10} have order $n = 3$. The minimal polynomial of α^5 and α^{10} is $X^2 + X + 1$, whose degree is a factor of $m = 4$. The conjugates of α^3 are α^6, α^9, and α^{12}. They all have order $n = 5$.

COMPUTATIONS USING GALOIS FIELD $GF(2^m)$ ARITHMETIC

Here we perform some example computations using arithmetic over $GF(2^m)$. Consider the following linear equations over $GF(2^4)$ (see Table 2.8):

$$\begin{aligned} X + \alpha^7 Y &= \alpha^2, \\ \alpha^{12} X + \alpha^8 Y &= \alpha^4. \end{aligned} \tag{2.26}$$

$$X = \frac{\begin{vmatrix} \alpha^2 & \alpha^7 \\ \alpha^4 & \alpha^8 \end{vmatrix}}{\begin{vmatrix} 1 & \alpha^7 \\ \alpha^{12} & \alpha^8 \end{vmatrix}} = \frac{\alpha^{10} + \alpha^{11}}{\alpha^8 + \alpha^{19}} = \frac{1 + \alpha^3}{\alpha + \alpha^2} = \frac{\alpha^{14}}{\alpha^5} = \alpha^9,$$

$$Y = \frac{\begin{vmatrix} 1 & \alpha^2 \\ \alpha^{12} & \alpha^4 \end{vmatrix}}{\begin{vmatrix} 1 & \alpha^7 \\ \alpha^{12} & \alpha^8 \end{vmatrix}} = \frac{\alpha^4 + \alpha^{14}}{\alpha^8 + \alpha^{19}} = \frac{\alpha + \alpha^3}{\alpha + \alpha^2} = \frac{\alpha^9}{\alpha^5} = \alpha^4.$$

As one more example, suppose that we want to solve the equation

$$f(X) = X^2 + \alpha^7 X + \alpha = 0$$

over $GF(2^4)$. The quadratic formula will not work because it requires dividing by 2, and in this field, $2 = 0$. If $f(X) = 0$ has any solutions in $GF(2^4)$, the solutions can be found simply by substituting all the elements of Table 2.8 for X. By doing so, we would find that $f(\alpha^6) = 0$ and $f(\alpha^{10}) = 0$, since

$$f(\alpha^6) = (\alpha^6)^2 + \alpha^7 \cdot \alpha^6 + \alpha = \alpha^{12} + \alpha^{13} + \alpha = 0,$$
$$f(\alpha^{10}) = (\alpha^{10})^2 + \alpha^7 \cdot \alpha^{10} + \alpha = \alpha^5 + \alpha^2 + \alpha = 0.$$

Thus, α^6 and α^{10} are the roots of $f(X)$, and $f(X) = (X + \alpha^6)(X + \alpha^{10})$.

The preceding computations are typical of those required for decoding codes such as BCH and Reed–Solomon codes, and they can be programmed quite easily on a general-purpose computer. It is also a simple matter to build a computer that can do this kind of arithmetic.

DESCRIPTION OF CYCLIC CODES

$$\mathbf{v} = (v_0, v_1, \cdots, v_{n-1}),$$

$$\mathbf{v}^{(1)} = (v_{n-1}, v_0, \cdots, v_{n-2})$$

$$\mathbf{v}^{(i)} = (v_{n-i}, v_{n-i+1}, \cdots, v_{n-1}, v_0, v_1, \cdots, v_{n-i-1}).$$

DEFINITION 5.1 An (n, k) linear code C is called a *cyclic code* if every cyclic shift of a codeword in C is also a codeword in C.

To develop the algebraic properties of a cyclic code, we treat the components of a codeword $\mathbf{v} = (v_0, v_1, \cdots, v_{n-1})$ as the coefficients of a polynomial as follows:

$$\mathbf{v}(X) = v_0 + v_1 X + v_2 X^2 + \cdots + v_{n-1} X^{n-1}.$$

TABLE 5.1: A $(7, 4)$ cyclic code generated by $g(X) = 1 + X + X^3$.

Messages	Code vectors	Code polynomials
(0000)	0000000	$0 = 0 \cdot g(X)$
(1000)	1101000	$1 + X + X^3 = 1 \cdot g(X)$
(0100)	0110100	$X + X^2 + X^4 = X \cdot g(X)$
(1100)	1011100	$1 + X^2 + X^3 + X^4 = (1 + X) \cdot g(X)$
(0010)	0011010	$X^2 + X^3 + X^5 = X^2 \cdot g(X)$
(1010)	1110010	$1 + X + X^2 + X^5 = (1 + X^2) \cdot g(X)$
(0110)	0101110	$X + X^3 + X^4 + X^5 = (X + X^2) \cdot g(X)$
(1110)	1000110	$1 + X^4 + X^5 = (1 + X + X^2) \cdot g(X)$
(0001)	0001101	$X^3 + X^4 + X^6 = X^3 \cdot g(X)$
(1001)	1100101	$1 + X + X^4 + X^6 = (1 + X^3) \cdot g(X)$
(0101)	0111001	$X + X^2 + X^3 + X^6 = (X + X^3) \cdot g(X)$
(1101)	1010001	$1 + X^2 + X^6 = (1 + X + X^3) \cdot g(X)$
(0011)	0010111	$X^2 + X^4 + X^5 + X^6 = (X^2 + X^3) \cdot g(X)$
(1011)	1111111	$1 + X + X^2 + X^3 + X^4 + X^5 + X^6$ $= (1 + X^2 + X^3) \cdot g(X)$
(0111)	0100011	$X + X^5 + X^6 = (X + X^2 + X^3) \cdot g(X)$
(1111)	1001011	$1 + X^3 + X^5 + X^6$ $= (1 + X + X^2 + X^3) \cdot g(X)$

$$\mathbb{v}^{(i)} = (v_{n-i}, v_{n-i+1}, \cdots, v_{n-1}, v_0, v_1, \cdots, v_{n-i-1}).$$

$$\mathbb{v}^{(i)}(X) = v_{n-i} + v_{n-i+1}X + \cdots + v_{n-1}X^{i-1} + v_0X^i + v_1X^{i+1} + \cdots + v_{n-i-1}X^{n-1}.$$

There exists an interesting algebraic relationship between $\mathbb{v}(X)$ and $\mathbb{v}^{(i)}(X)$. Multiplying $\mathbb{v}(X)$ by X^i, we obtain

$$X^i\mathbb{v}(X) = v_0X^i + v_1X^{i+1} + \cdots + v_{n-i-1}X^{n-1} + \cdots + v_{n-1}X^{n+i-1}.$$

The preceding equation can be manipulated into the following form:

$$\begin{aligned} X^i\mathbb{v}(X) &= v_{n-i} + v_{n-i+1}X + \cdots + v_{n-1}X^{i-1} + v_0X^i + \cdots + v_{n-i-1}X^{n-1} \\ &\quad + v_{n-i}(X^n + 1) + v_{n-i+1}X(X^n + 1) + \cdots + v_{n-1}X^{i-1}(X^n + 1) \qquad (5.1) \\ &= \mathbb{q}(X)(X^n + 1) + \mathbb{v}^{(i)}(X), \end{aligned}$$

where $\mathbb{q}(X) = v_{n-i} + v_{n-i+1}X + \cdots + v_{n-1}X^{i-1}$

THEOREM 5.1 The nonzero code polynomial of minimum degree in a cyclic code C is unique.

THEOREM 5.2 Let $\mathbf{g}(X) = g_0 + g_1 X + \cdots + g_{r-1} X^{r-1} + X^r$ be the nonzero code polynomial of minimum degree in an (n, k) cyclic code C. Then, the constant term g_0 must be equal to 1.

THEOREM 5.3 Let $\mathbf{g}(X) = 1 + \mathbf{g}_1 X + \cdots + \mathbf{g}_{r-1} X^{r-1} + X^r$ be the nonzero code polynomial of minimum degree in an (n, k) cyclic code C. A binary polynomial of degree $n-1$ or less is a code polynomial if and only if it is a multiple of $\mathbf{g}(X)$.

THEOREM 5.4 In an (n, k) cyclic code, there exists one and only one code polynomial of degree $n - k$,

$$g(X) = 1 + g_1 X + g_2 X^2 + \cdots + g_{n-k-1} X^{n-k-1} + X^{n-k}. \tag{5.4}$$

Every code polynomial is a multiple of $g(X)$, and every binary polynomial of degree $n - 1$ or less that is a multiple of $g(X)$ is a code polynomial.

The number of binary polynomials of degree $n - 1$ or less that are multiples of $g(X)$ is 2^{n-r}. It follows from Theorem 5.3 that these polynomials form all the code polynomials of the (n, k) cyclic code C. Because there are 2^k code polynomials in C, then 2^{n-r} must be equal to 2^k. As a result, we have $r = n - k$ [i.e., the degree of $g(X)$ is $n - k$]. Hence, the nonzero code polynomial of minimum degree in an (n, k) cyclic code is of the following form:

$$g(X) = 1 + g_1 X + g_2 X^2 + \cdots + g_{n-k-1} X^{n-k-1} + X^{n-k}.$$

It follows from Theorem 5.4 that every code polynomial $\mathbb{v}(X)$ in an (n, k) cyclic code can be expressed in the following form:

$$\begin{aligned}\mathbb{v}(X) &= \mathbb{u}(X)\mathbb{g}(X) \\ &= (u_0 + u_1 X + \cdots + u_{k-1}X^{k-1})\mathbb{g}(X).\end{aligned}$$

If the coefficients of $\mathbb{u}(X)$, $u_0, u_1, \cdots, u_{k-1}$, are the k information digits to be encoded, $\mathbb{v}(X)$ is the corresponding code polynomial. Hence, the encoding can be achieved by multiplying the message $\mathbb{u}(X)$ by $\mathbb{g}(X)$. Therefore, an (n, k) cyclic code is completely specified by its nonzero code polynomial of minimum degree, $\mathbb{g}(X)$,

given by (5.4). The polynomial $\mathbf{g}(X)$ is called the *generator polynomial* of the code. The degree of $\mathbb{g}(X)$ is equal to the number of parity-check digits of the code. The generator polynomial of the $(7, 4)$ cyclic code given in Table 4.1 is $\mathbf{g}(X) = 1 + X + X^3$. We see that each code polynomial is a multiple of $\mathbf{g}(X)$.

THEOREM 5.5 The generator polynomial $\mathbf{g}(X)$ of an (n, k) cyclic code is a factor of $X^n + 1$.

At this point, a natural question is whether, for any n and k, there exists an (n, k) cyclic code. This question is answered by the following theorem.

THEOREM 5.6 If $\mathbf{g}(X)$ is a polynomial of degree $n-k$ and is a factor of X^n+1, then $\mathbf{g}(X)$ generates an (n, k) cyclic code.

Theorem 5.6 says that any factor of $X^n + 1$ with degree $n - k$ generates an (n, k) cyclic code. For large n, $X^n + 1$ may have many factors of degree $n - k$. Some of these polynomials generate good codes, and some generate bad codes. How to select generator polynomials to produce good cyclic codes is a very difficult problem, and coding theorists have expended much effort in searching for good cyclic codes. Several classes of good cyclic codes have been discovered, and they can be practically implemented.

EXAMPLE 5.1

The polynomial $X^7 + 1$ can be factored as follows:

$$X^7 + 1 = (1 + X)(1 + X + X^3)(1 + X^2 + X^3).$$

There are two factors of degree 3, and each generates a (7, 4) cyclic code. The (7, 4) cyclic code given by Table 5.1 is generated by $\mathbb{g}(X) = 1 + X + X^3$. This code has a minimum distance of 3 and it is a single-error-correcting code. Notice that the code is not in systematic form. Each code polynomial is the product of a message polynomial of degree 3 or less and the generator polynomial $\mathbb{g}(X) = 1 + X + X^3$. For example, let $\mathbb{u} = (1\,0\,1\,0)$ be the message to be encoded. The corresponding message polynomial is $\mathbb{u}(X) = 1 + X^2$. Multiplying $\mathbb{u}(X)$ by $\mathbb{g}(X)$ gives us the following code polynomial:

$$\begin{aligned} \mathbb{v}(X) &= (1 + X^2)(1 + X + X^3) \\ &= 1 + X + X^2 + X^5, \end{aligned}$$

or the codeword $(1\,1\,1\,0\,0\,1\,0)$.

$$\mathbb{u} = (u_0, u_1, \cdots, u_{k-1}).$$

$$\mathbb{u}(X) = u_0 + u_1 X + \cdots + u_{k-1} X^{k-1}.$$

Multiplying $\mathbb{u}(X)$ by X^{n-k}, we obtain a polynomial of degree $n-1$ or less:

$$X^{n-k}\mathbb{u}(X) = u_0 X^{n-k} + u_1 X^{n-k+1} + \cdots + u_{k-1} X^{n-1}.$$

Dividing $X^{n-k}\mathbb{u}(X)$ by the generator polynomial $\mathbb{g}(X)$, we have

$$X^{n-k}\mathbb{u}(X) = \mathbb{a}(X)\mathbb{g}(X) + \mathbb{b}(X) \tag{5.6}$$

where $\mathbb{a}(X)$ and $\mathbb{b}(X)$ are the quotient and the remainder, respectively. Because the degree of $\mathbb{g}(X)$ is $n-k$, the degree of $\mathbb{b}(X)$ must be $n-k-1$ or less; that is,

$$\mathbb{b}(X) = b_0 + b_1 X + \cdots + b_{n-k-1} X^{n-k-1}.$$

Rearranging (5.6), we obtain the following polynomial of degree $n-1$ or less:

$$\mathbb{b}(X) + X^{n-k}\mathbb{u}(X) = \mathbb{a}(X)\mathbb{g}(X). \tag{5.7}$$

This polynomial is a multiple of the generator polynomial $\mathbb{g}(X)$ and therefore it is a code polynomial of the cyclic code generated by $\mathbb{g}(X)$. Writing out $\mathbb{b}(X) + X^{n-k}\mathbb{u}(X)$, we have

$$\begin{aligned}\mathbb{b}(X) + X^{n-k}\mathbb{u}(X) = {} & b_0 + b_1 X + \cdots + b_{n-k-1}X^{n-k-1} \\ & + u_0 X^{n-k} + u_1 X^{n-k+1} + \cdots + u_{k-1}X^{n-1},\end{aligned} \tag{5.8}$$

which corresponds to the codeword

$$(b_0, b_1, \cdots, b_{n-k-1}, u_0, u_1, \cdots, u_{k-1}).$$

In summary, encoding in systematic form consists of three steps:

Step 1. Premultiply the message $\mathbf{u}(X)$ by X^{n-k}.

Step 2. Obtain the remainder $\mathbf{b}(X)$ (the parity-check digits) from dividing $X^{n-k}\mathbf{u}(X)$ by the generator polynomial $\mathbf{g}(X)$.

Step 3. Combine $\mathbf{b}(X)$ and $X^{n-k}\mathbf{u}(X)$ to obtain the code polynomial $\mathbf{b}(X) + X^{n-k}\mathbf{u}(X)$.

EXAMPLE 5.2

Consider the (7, 4) cyclic code generated by $\mathbf{g}(X) = 1 + X + X^3$. Let $\mathbf{u}(X) = 1 + X^3$ be the message to be encoded. Dividing $X^3\mathbf{u}(X) = X^3 + X^6$ by $\mathbf{g}(X)$,

$$
\begin{array}{r|l}
 & X^3 + X \quad \text{(quotient)} \\
\hline
X^3 + X + 1 & X^6 \qquad\qquad\quad X^3 \\
 & \underline{X^6 \quad + X^4 + \; X^3} \\
 & \qquad X^4 \\
 & \qquad \underline{X^4 \qquad\quad + X^2 + X} \\
 & \qquad\qquad\qquad X^2 + X \quad \text{(remainder)}
\end{array}
$$

we obtain the remainder $\mathbf{b}(X) = X + X^2$. Thus, the code polynomial is $\mathbf{v}(X) = \mathbf{b}(X) + X^3\mathbf{u}(X) = X + X^2 + X^3 + X^6$, and the corresponding codeword is $\mathbf{v} = (0\,1\,1\,1\,0\,0\,1)$, where the four rightmost digits are the information digits. The 16 codewords in systematic form are listed in Table 5.2.

TABLE 5.2: A (7, 4) cyclic code in systematic form generated by $g(X) = 1 + X + X^3$.

Message	Codeword	
(0 0 0 0)	(0 0 0 0 0 0 0)	$0 = 0 \cdot g(X)$
(1 0 0 0)	(1 1 0 1 0 0 0)	$1 + X + X^3 = g(X)$
(0 1 0 0)	(0 1 1 0 1 0 0)	$X + X^2 + X^4 = Xg(X)$
(1 1 0 0)	(1 0 1 1 1 0 0)	$1 + X^2 + X^3 + X^4 = (1 + X)g(X)$
(0 0 1 0)	(1 1 1 0 0 1 0)	$1 + X + X^2 + X^5 = (1 + X^2)g(X)$
(1 0 1 0)	(0 0 1 1 0 1 0)	$X^2 + X^3 + X^5 = X^2g(X)$
(0 1 1 0)	(1 0 0 0 1 1 0)	$1 + X^4 + X^5 = (1 + X + X^2)g(X)$
(1 1 1 0)	(0 1 0 1 1 1 0)	$X + X^3 + X^4 + X^5 = (X + X^2)g(X)$
(0 0 0 1)	(1 0 1 0 0 0 1)	$1 + X^2 + X^6 = (1 + X + X^3)g(X)$
(1 0 0 1)	(0 1 1 1 0 0 1)	$X + X^2 + X^3 + X^6 = (X + X^3)g(X)$
(0 1 0 1)	(1 1 0 0 1 0 1)	$1 + X + X^4 + X^6 = (1 + X^3)g(X)$
(1 1 0 1)	(0 0 0 1 1 0 1)	$X^3 + X^4 + X^6 = X^3g(X)$
(0 0 1 1)	(0 1 0 0 0 1 1)	$X + X^5 + X^6 = (X + X^2 + X^3)g(X)$
(1 0 1 1)	(1 0 0 1 0 1 1)	$1 + X^3 + X^5 + X^6 = (1 + X + X^2 + X^3)g(X)$
(0 1 1 1)	(0 0 1 0 1 1 1)	$X^2 + X^4 + X^5 + X^6 = (X^2 + X^3)g(X)$
(1 1 1 1)	(1 1 1 1 1 1 1)	$1 + X + X^2 + X^3 + X^4 + X^5 + X^6$ $= (1 + X^2 + X^5)g(X)$

GENERATOR AND PARITY-CHECK MATRICES OF CYCLIC CODES

Consider an (n, k) cyclic code C with generator polynomial $g(X) = g_0 + g_1 X + \cdots + g_{n-k} X^{n-k}$. In Section 5.1 we showed that the k code polynomials $g(X)$, $Xg(X)$, $\cdots$, $X^{k-1}g(X)$ span C. If the n-tuples corresponding to these k code polynomials are used as the rows of a $k \times n$ matrix, we obtain the following generator matrix for C:

$$\mathbb{G} = \begin{bmatrix} g_0 & g_1 & g_2 & \cdot & \cdot & \cdot & \cdot & \cdot & g_{n-k} & 0 & 0 & 0 & \cdot & \cdot & 0 \\ 0 & g_0 & g_1 & g_2 & \cdot & \cdot & \cdot & \cdot & \cdot & g_{n-k} & 0 & 0 & \cdot & \cdot & 0 \\ 0 & 0 & g_0 & g_1 & g_2 & \cdot & \cdot & \cdot & \cdot & \cdot & g_{n-k} & 0 & \cdot & \cdot & 0 \\ \cdot & & & & & & & & & & & & & & \cdot \\ \cdot & & & & & & & & & & & & & & \cdot \\ \cdot & & & & & & & & & & & & & & \cdot \\ 0 & 0 & \cdot & \cdot & \cdot & 0 & g_0 & g_1 & g_2 & \cdot & \cdot & \cdot & \cdot & \cdot & g_{n-k} \end{bmatrix}. \tag{5.9}$$

$$\mathbb{g}(X) = 1 + \bar{X} + X^3 \quad (7, 4) \text{ cyclic code}$$

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

Clearly, $\mathbb{G}$ is not in systematic form. If we add the first row to the third row, and if we add the sum of the first two rows to the fourth row, we obtain the following matrix:

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

which is in systematic form. This matrix generates the same code as $\mathbb{G}$.

Recall that the generator polynomial $\mathbf{g}(X)$ is a factor of $X^n + 1$, say

$$X^n + 1 = \mathbf{g}(X)\mathbf{h}(X), \tag{5.10}$$

where the polynomial $\mathbf{h}(X)$ has degree k and is of the following form:

$$\mathbf{h}(X) = h_0 + h_1 X + \cdots + h_k X^k$$

with $h_0 = h_k = 1$. Next, we want to show that a parity-check matrix of C may be obtained from $\mathbf{h}(X)$. Let $\mathbf{v} = (v_0, v_1, \cdots, v_{n-1})$ be a codeword in C. Then, $\mathbf{v}(X) = \mathbf{a}(X)\mathbf{g}(X)$. Multiplying $\mathbf{v}(X)$ by $\mathbf{h}(X)$, we obtain

$$\begin{aligned} \mathbf{v}(X)\mathbf{h}(X) &= \mathbf{a}(X)\mathbf{g}(X)\mathbf{h}(X) \\ &= \mathbf{a}(X)(X^n + 1) \\ &= \mathbf{a}(X) + X^n\mathbf{a}(X). \end{aligned} \tag{5.11}$$

Because the degree of $\mathbf{a}(X)$ is $k-1$ or less, the powers $X^k, X^{k+1}, \cdots, X^{n-1}$ do not appear in $\mathbf{a}(X) + X^n\mathbf{a}(X)$. If we expand the product $\mathbf{v}(X)\mathbf{h}(X)$ on the left-hand side of (5.11), the coefficients of $X^k, X^{k+1}, \cdots, X^{n-1}$ must be equal to zero. Therefore, we obtain the following $n-k$ equalities:

$$\sum_{i=0}^{k} h_i v_{n-i-j} = 0 \quad \text{for } 1 \le j \le n-k. \tag{5.12}$$

Now, we take the *reciprocal* of $\mathbf{h}(X)$, which is defined as follows:

$$X^k\mathbf{h}(X^{-1}) \triangleq h_k + h_{k-1}X + h_{k-2}X^2 + \cdots + h_0X^k. \tag{5.13}$$

We can easily see that $X^k \mathbb{h}(X^{-1})$ is also a factor of $X^n + 1$. The polynomial $X^k \mathbb{h}(X^{-1})$ generates an $(n, n-k)$ cyclic code with the following $(n-k) \times n$ matrix as a generator matrix:

$$\mathbb{H} = \begin{bmatrix} h_k & h_{k-1} & h_{k-2} & \cdot & \cdot & \cdot & \cdot & \cdot & h_0 & 0 & \cdot & \cdot & \cdot & \cdot & 0 \\ 0 & h_k & h_{k-1} & h_{k-2} & \cdot & \cdot & \cdot & \cdot & \cdot & h_0 & 0 & \cdot & \cdot & \cdot & 0 \\ 0 & 0 & h_k & h_{k-1} & h_{k-2} & \cdot & \cdot & \cdot & \cdot & \cdot & h_0 & \cdot & \cdot & \cdot & 0 \\ \vdots & & & & & & & & & & & & & & \vdots \\ 0 & 0 & \cdot & \cdot & \cdot & 0 & h_k & h_{k-1} & h_{k-2} & \cdot & \cdot & \cdot & \cdot & \cdot & h_0 \end{bmatrix}. \tag{5.14}$$

It follows from the $n - k$ equalities of (5.12) that any codeword $\mathbb{v}$ in C is orthogonal to every row of $\mathbb{H}$. Therefore, $\mathbb{H}$ is a parity-check matrix of the cyclic code C, and the row space of $\mathbb{H}$ is the dual code of C. Since the parity-check matrix $\mathbb{H}$ is obtained from the polynomial $\mathbb{h}(X)$, we call $\mathbb{h}(X)$ the *parity polynomial* of C. Hence, a cyclic code is also uniquely specified by its parity polynomial.

Besides deriving a parity-check matrix for a cyclic code, we have also proved another important property, which is stated in the following theorem.

THEOREM 5.7 Let C be an (n, k) cyclic code with generator polynomial $\mathbb{g}(X)$. The dual code of C is also cyclic and is generated by the polynomial $X^k \mathbb{h}(X^{-1})$, where $\mathbb{h}(X) = (X^n + 1)/\mathbb{g}(X)$.

EXAMPLE 5.3

Consider the (7, 4) cyclic code given in Table 5.1 with generator polynomial $g(X) = 1 + X + X^3$. The parity polynomial is

$$h(X) = \frac{X^7 + 1}{g(X)}$$
$$= 1 + X + X^2 + X^4.$$

The reciprocal of $h(X)$ is

$$X^4 h(X^{-1}) = X^4(1 + X^{-1} + X^{-2} + X^{-4})$$
$$= 1 + X^2 + X^3 + X^4.$$

This polynomial $X^4 h(X^{-1})$ divides $X^7 + 1 : (X^7 + 1)/X^4 h(X^{-1}) = 1 + X^2 + X^3$. If we construct all the codewords of the (7, 3) code generated by $X^4 h(X^{-1}) = 1 + X^2 + X^3 + X^4$, we will find that it has a minimum distance of 4. Hence, it is capable of correcting any single error and simultaneously detecting any combination of double errors.

We also can easily form the generator matrix in systematic form. Dividing X^{n-k+i} by the generator polynomial $\mathbb{g}(X)$ for $i = 0, 1, \cdots, k-1$, we obtain

$$X^{n-k+i} = \mathbb{a}_i(X)\mathbb{g}(X) + \mathbb{b}_i(X), \tag{5.15}$$

where $\mathbb{b}_i(X)$ is the remainder with the following form:

$$\mathbb{b}_i(X) = b_{i0} + b_{i1}X + \cdots + b_{i,n-k-1}X^{n-k-1}.$$

Because $\mathbb{b}_i(X) + X^{n-k+i}$ for $i = 0, 1, \cdots, k-1$ are multiples of $\mathbb{g}(X)$, they are code polynomials. Arranging these k code polynomials as rows of a $k \times n$ matrix, we obtain

$$\mathbb{G} = \begin{bmatrix} b_{00} & b_{01} & b_{02} & \cdots & b_{0,n-k-1} & 1 & 0 & 0 & \cdots & 0 \\ b_{10} & b_{11} & b_{12} & \cdots & b_{1,n-k-1} & 0 & 1 & 0 & \cdots & 0 \\ b_{20} & b_{21} & b_{22} & \cdots & b_{2,n-k-1} & 0 & 0 & 1 & \cdots & 0 \\ & \vdots & & & \vdots & & & & & \vdots \\ b_{k-1,0} & b_{k-1,1} & b_{k-1,2} & \cdots & b_{k-1,n-k-1} & 0 & 0 & 0 & \cdots & 1 \end{bmatrix}, \tag{5.16}$$

which is the generator matrix of C in systematic form. The corresponding parity-check matrix for C is

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & b_{00} & b_{10} & b_{20} & \cdots & b_{k-1,0} \\ 0 & 1 & 0 & \cdots & 0 & b_{01} & b_{11} & b_{21} & \cdots & b_{k-1,1} \\ 0 & 0 & 1 & \cdots & 0 & b_{02} & b_{12} & b_{22} & \cdots & b_{k-1,2} \\ & \vdots & & & & & \vdots & & & \vdots \\ 0 & 0 & 0 & \cdots & 1 & b_{0,n-k-1} & b_{1,n-k-1} & b_{2,n-k-1} & \cdots & b_{k-1,n-k-1} \end{bmatrix}. \tag{5.17}$$

EXAMPLE 5.4

Again, consider the (7, 4) cyclic code generated by $\mathbf{g}(X) = 1 + X + X^3$. Dividing X^3, X^4, X^5, and X^6 by $\mathbf{g}(X)$, we have

$$\begin{aligned}
X^3 &= \mathbf{g}(X) + (1 + X), \\
X^4 &= X\mathbf{g}(X) + (X + X^2), \\
X^5 &= (X^2 + 1)\mathbf{g}(X) + (1 + X + X^2), \\
X^6 &= (X^3 + X + 1)\mathbf{g}(X) + (1 + X^2).
\end{aligned}$$

Rearranging the preceding equations, we obtain the following four code polynomials:

$$\begin{aligned}
\mathbf{v}_0(X) &= 1 + X + X^3, \\
\mathbf{v}_1(X) &= X + X^2 + X^4, \\
\mathbf{v}_2(X) &= 1 + X + X^2 + X^5, \\
\mathbf{v}_3(X) &= 1 + X^2 + X^6.
\end{aligned}$$

Taking these four code polynomials as rows of a 4×7 matrix, we obtain the following generator matrix in systematic form for the (7, 4) cyclic code:

$$\mathbf{G} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix},$$

which is identical to the matrix G' obtained earlier in this section.

Comments: Division and Multiplication Circuits

ENCODING OF CYCLIC CODES

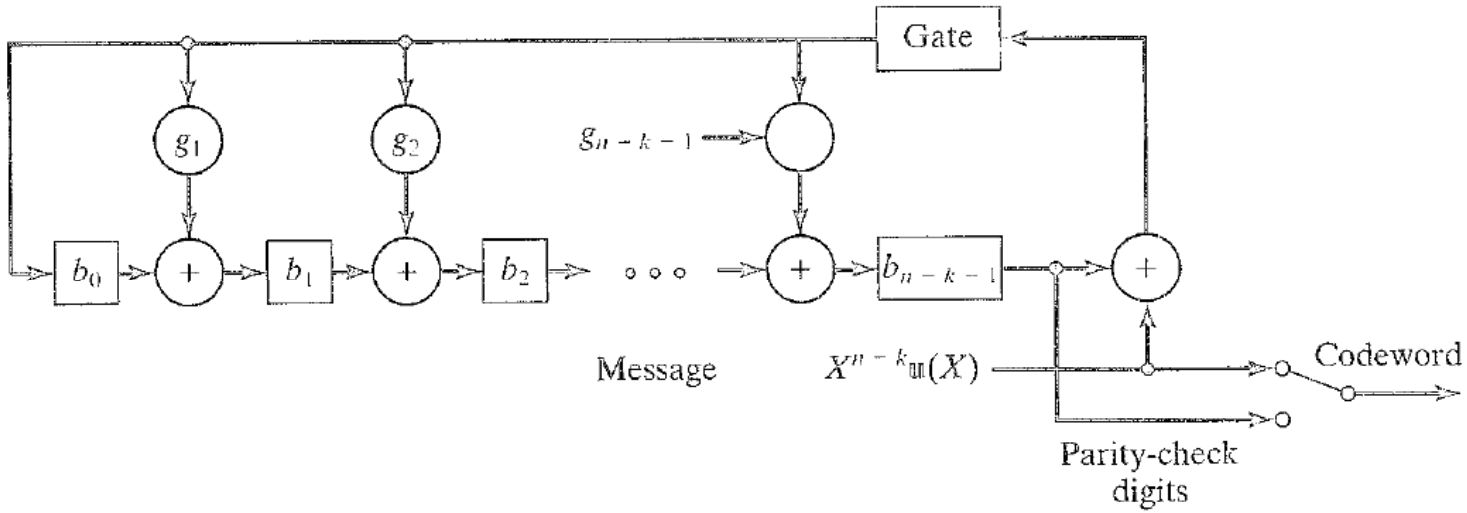
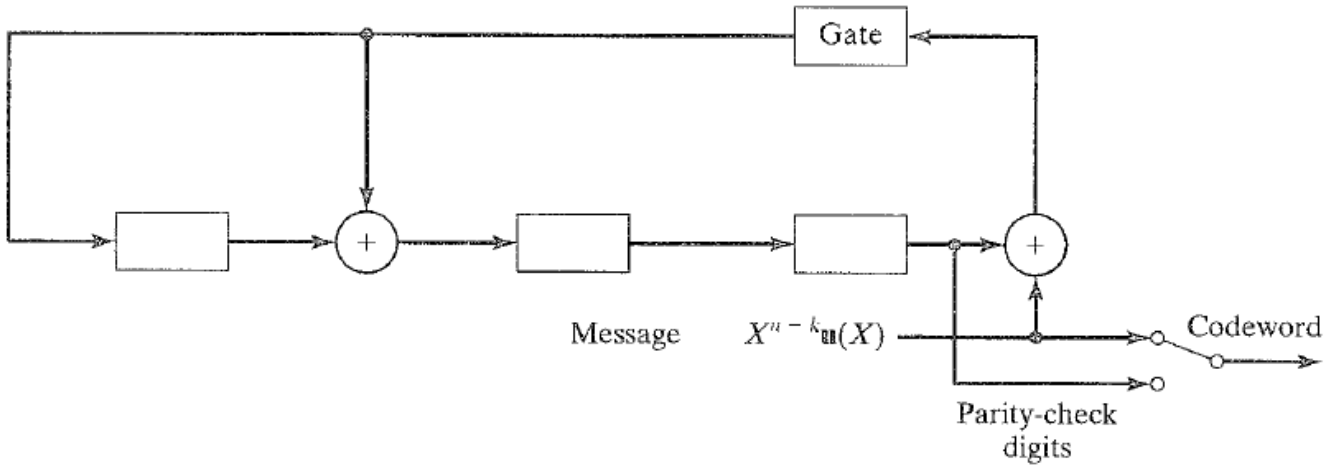


FIGURE 5.1: Encoding circuit for an (n, k) cyclic code with generator polynomial $\mathbb{g}(X) = 1 + g_1 X^2 + \cdots + g_{n-k-1} X^{n-k-1} + X^{n-k}$.

Step 1. Turn on the gate. The k information digits $u_0, u_1, \cdots, u_{k-1}$ [or $\mathbb{u}(X) = u_0 + u_1 X + \cdots + u_{k-1} X^{k-1}$ in polynomial form] are shifted into the circuit and simultaneously into the communication channel. Shifting the message $\mathbb{u}(X)$ into the circuit from the front end is equivalent to premultiplying $\mathbb{u}(X)$ by X^{n-k}. As soon as the complete message has entered the circuit, the $n-k$ digits in the register form the remainder, and thus they are the parity-check digits.

Step 2. Break the feedback connection by turning off the gate.

Step 3. Shift the parity-check digits out and send them into the channel. These $n-k$ parity-check digits $b_0, b_1, \cdots, b_{n-k-1}$, together with the k information digits, form a complete codeword.



EXAMPLE 5.5

Consider the $(7, 4)$ cyclic code generated by $g(X) = 1 + X + X^3$. The encoding circuit based on $g(X)$ is shown in Figure 5.2. Suppose that the message $\mathbf{u} = (1\,0\,1\,1)$ is to be encoded. As the message digits are shifted into the register the contents of the register change as follows:

Input	Register contents
	0 0 0 (initial state)
1	1 1 0 (first shift)
1	1 0 1 (second shift)
0	1 0 0 (third shift)
1	1 0 0 (fourth shift)

After four shifts, the contents of the register are $(1\,0\,0)$. Thus, the complete codeword is $(1\,0\,0\,1\,0\,1\,1)$, and the code polynomial is $1 + X^3 + X^5 + X^6$.

Encoding of a cyclic code can also be accomplished by using its parity polynomial $\mathbb{h}(X) = h_0 + h_1 X + \cdots + h_k X^k$. Let $\mathbb{v} = (v_0, v_1, \cdots, v_{n-1})$ be a codeword. We have shown in Section 5.2 that the components of $\mathbb{v}$ satisfy the $n - k$ equalities

of (5.12). Since $h_k = 1$, the equalities of (5.12) can be put into the following form:

$$v_{n-k-j} = \sum_{i=0}^{k-1} h_i v_{n-i-j} \quad \text{for } 1 \leq j \leq n-k \tag{5.18}$$

which is known as a *difference equation*. For a cyclic code in systematic form, the components $v_{n-k}, v_{n-k+1}, \cdots, v_{n-1}$ of each codeword are the information digits. Given these k information digits, (5.18) is a rule for determining the $n - k$ parity-check digits, $v_0, v_1, \cdots, v_{n-k-1}$. An encoding circuit based on (5.18) is shown in Figure 5.3. The feedback connections are based on the coefficients of the parity polynomial $\mathbb{h}(X)$. (Note that $h_0 = h_k = 1$.) The encoding operation can be described in the following steps:

Step 1. Initially, gate 1 is turned on and gate 2 is turned off. The k information digits $\mathfrak{u}(X) = u_0 + u_1 X + \cdots + u_{k-1} X^{k-1}$ are shifted into the register and the communication channel simultaneously.

Step 2. As soon as the k information digits have entered the shift register, gate 1 is turned off and gate 2 is turned on. The first parity-check digit,

$$\begin{aligned} v_{n-k-1} &= h_0 v_{n-1} + h_1 v_{n-2} + \cdots + h_{k-1} v_{n-k} \\ &= u_{k-1} + h_1 u_{k-2} + \cdots + h_{k-1} u_0, \end{aligned}$$

is formed and appears at point P.

Step 3. The register is shifted once. The first parity-check digit is shifted into the channel and is also shifted into the register. Now, the second parity-check digit,

$$\begin{aligned} v_{n-k-2} &= h_0 v_{n-2} + h_1 v_{n-3} + \cdots + h_{k-1} v_{n-k-1} \\ &= u_{k-2} + h_1 u_{k-3} + \cdots + h_{k-2} u_0 + h_{k-1} v_{n-k-1}, \end{aligned}$$

is formed at P.

Step 4. Step 3 is repeated until $n-k$ parity-check digits have been formed and shifted into the channel. Then, gate 1 is turned on and gate 2 is turned off. The next message is now ready to be shifted into the register.

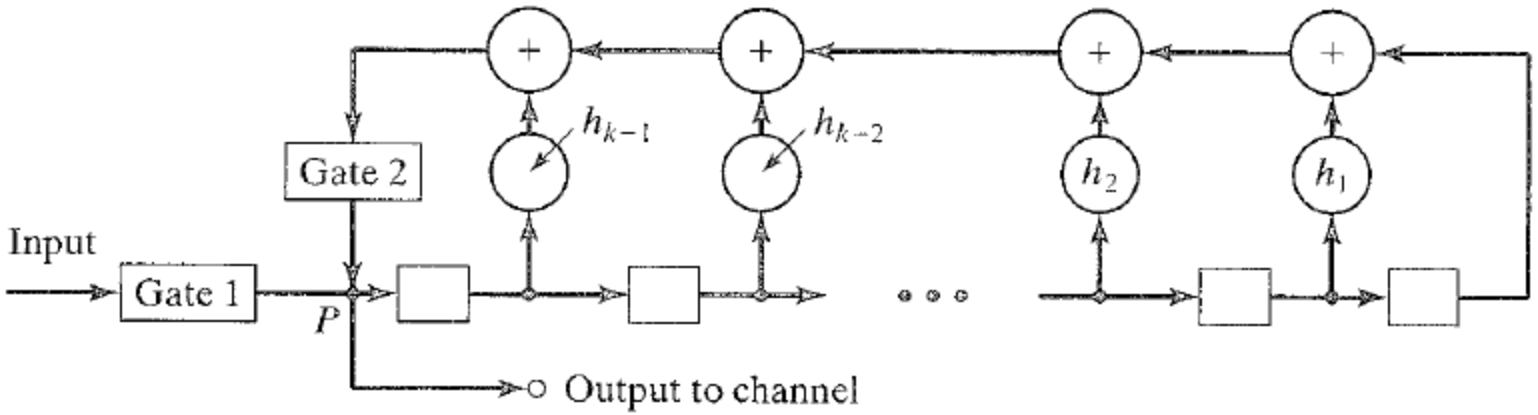


FIGURE 5.3: Encoding circuit for an (n, k) cyclic code based on the parity polynomial $h(X) = 1 + h_1 X + \cdots + X^k$.

EXAMPLE 5.6

The parity polynomial of the (7, 4) cyclic code generated by $g(X) = 1 + X + X^3$ is

$$h(X) = \frac{X^7 + 1}{1 + X + X^3} = 1 + X + X^2 + X^4.$$

The encoding circuit based on $h(X)$ is shown in Figure 5.4. Each codeword is of the form $v = (v_0, v_1, v_2, v_3, v_4, v_5, v_6)$, where v_3, v_4, v_5, and v_6 are message digits, and v_0, v_1, and v_2 are parity-check digits. The difference equation that determines the parity-check digits is

$$\begin{aligned} v_{3-j} &= 1 \cdot v_{7-j} + 1 \cdot v_{6-j} + 1 \cdot v_{5-j} + 0 \cdot v_{4-j} \\ &= v_{7-j} + v_{6-j} + v_{5-j} \qquad \text{for } 1 \le j \le 3. \end{aligned}$$

Suppose that the message to be encoded is (1 0 1 1). Then, $v_3 = 1$, $v_4 = 0$, $v_5 = 1$, $v_6 = 1$. The first parity-check digit is

$$v_2 = v_6 + v_5 + v_4 = 1 + 1 + 0 = 0.$$

The second parity-check digit is

$$v_1 = v_5 + v_4 + v_3 = 1 + 0 + 1 = 0.$$

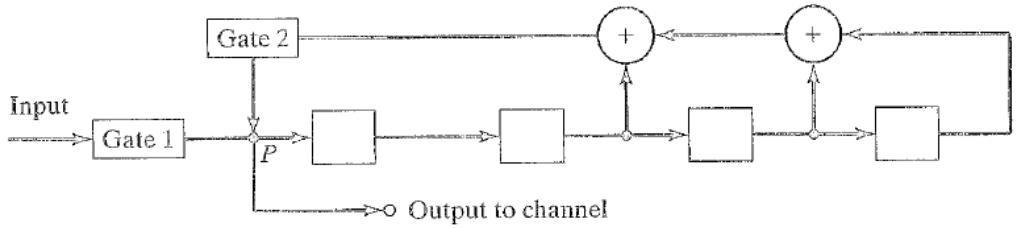


FIGURE 5.4: Encoding circuit for the (7, 4) cyclic code based on its parity polynomial $h(X) = 1 + X + X^2 + X^4$.

The third parity-check digit is

$$v_0 = v_4 + v_3 + v_2 = 0 + 1 + 0 = 1.$$

Thus, the codeword that corresponds to the message (1 0 1 1) is (1 0 0 1 0 1 1).

SYNDROME COMPUTATION AND ERROR DETECTION

Let $\mathbf{r} = (r_0, r_1, \cdots, r_{n-1})$ be the received vector.

$$\mathbf{r}(X) = r_0 + r_1 X + r_2 X^2 + \cdots + r_{n-1} X^{n-1}.$$

Dividing $\mathbf{r}(X)$ by the generator polynomial $\mathbf{g}(X)$, we obtain

$$\mathbf{r}(X) = \mathbf{a}(X)\mathbf{g}(X) + \mathbf{s}(X).$$

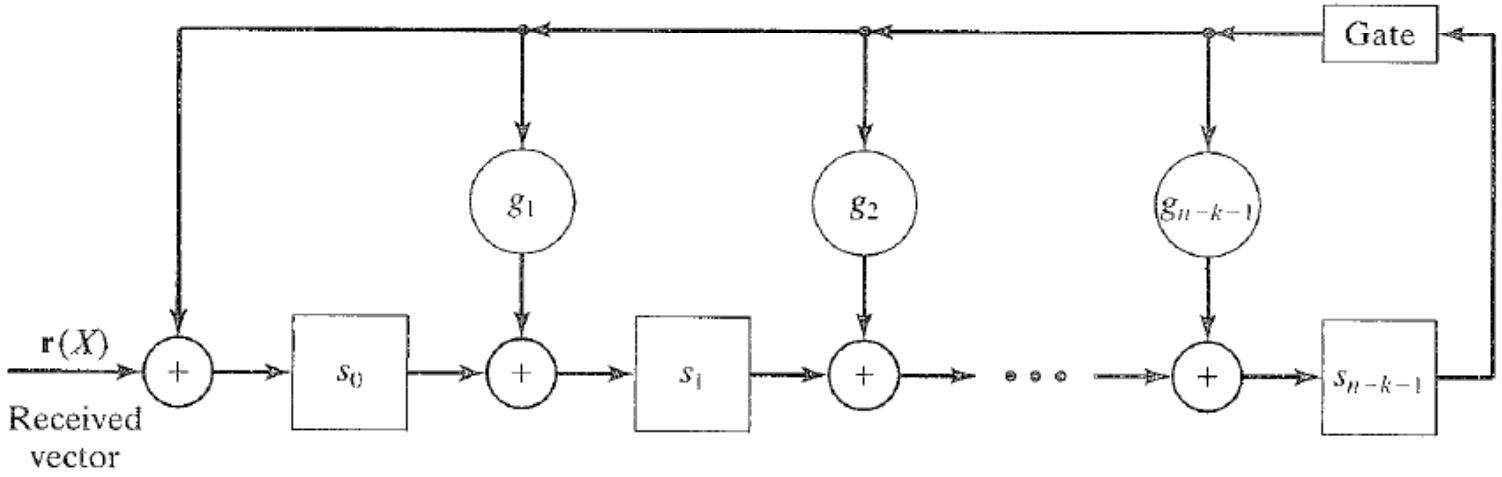


FIGURE 5.5: An $(n-k)$-stage syndrome circuit with input from the left end.

THEOREM 5.8 Let $\mathrm{s}(X)$ be the syndrome of a received polynomial $\mathrm{r}(X) = r_0 + r_1 X + \cdots + r_{n-1} X^{n-1}$. Then, the remainder $\mathrm{s}^{(1)}(X)$ resulting from dividing $X\mathrm{s}(X)$ by the generator polynomial $\mathrm{g}(X)$ is the syndrome of $\mathrm{r}^{(1)}(X)$, which is a cyclic shift of $\mathrm{r}(X)$.

It follows from Theorem 5.8 that the remainder $\mathbb{s}^{(i)}(X)$ resulting from dividing $X^i\mathbb{s}(X)$ by the generator polynomial $\mathbb{g}(X)$ is the syndrome of $\mathbb{r}^{(i)}(X)$, which is the ith cyclic shift of $\mathbb{r}(X)$. This property is useful in decoding cyclic codes. We can obtain the syndrome $\mathbb{s}^{(1)}(X)$ of $\mathbb{r}^{(1)}(X)$ by shifting (or clocking) the syndrome register once with $\mathbb{s}(X)$ as the initial contents and with the input gate disabled. This is because shifting the syndrome register once with $\mathbb{s}(X)$ as the initial contents is equivalent to dividing $X\mathbb{s}(X)$ by $\mathbb{g}(X)$. Thus, after the shift, the register contains $\mathbb{s}^{(1)}(X)$. To obtain the syndrome $\mathbb{s}^{(i)}(X)$ of $\mathbb{r}^{(i)}(X)$, we simply shift the syndrome register i times with $\mathbb{s}(X)$ as the initial contents.

EXAMPLE 5.7

A syndrome circuit for the (7, 4) cyclic code generated by $g(X) = 1 + X + X^3$ is shown in Figure 5.6. Suppose that the received vector is $\mathbf{r} = (0010110)$. The syndrome of $\mathbf{r}$ is $\mathbf{s} = (101)$. Table 5.3 shows the contents in the register as the received vector is shifted into the circuit. At the end of the seventh shift, the register contains the syndrome $\mathbf{s} = (101)$. If the register is shifted once more with the input gate disabled, the new contents will be $\mathbf{s}^{(1)} = (1\ 0\ 0)$, which is the syndrome of $\mathbf{r}^{(1)}(X) = (0001011)$, a cyclic shift of $\mathbf{r}$.

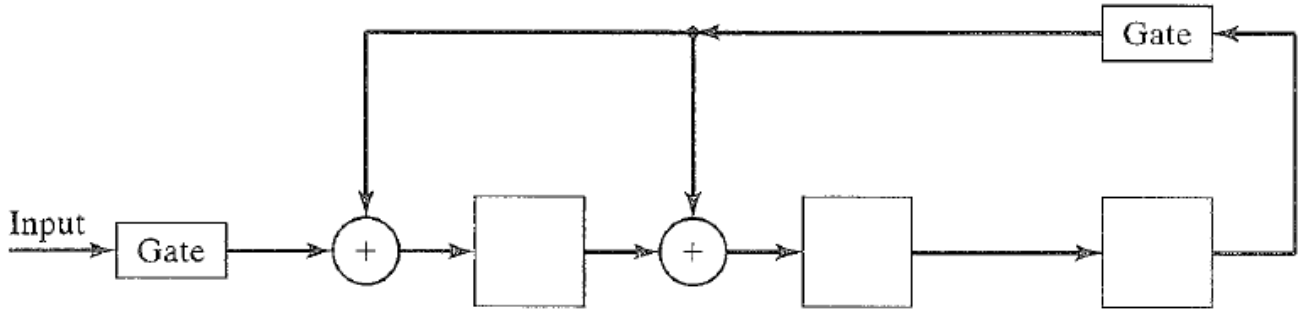


FIGURE 5.6: Syndrome circuit for the (7, 4) cyclic code generated by $g(X) = 1 + X + X^3$.

TABLE 5.3: Contents of the syndrome register shown in Figure 5.6 with $\mathbf{r} = (0\,0\,1\,0\,1\,1\,0)$ as input.

Shift	Input	Register contents
		000 (initial state)
1	0	000
2	1	100
3	1	110
4	0	011
5	1	011
6	0	111
7	0	101 (syndrome $\mathbf{s}$)
8	—	100 (syndrome $\mathbf{s}^{(1)}$)
9	—	010 (syndrome $\mathbf{s}^{(2)}$)

Let $\mathbb{v}(X)$ be the transmitted codeword, and let $\mathbb{e}(X) = e_0 + e_1 X + \cdots + e_{n-1} X^{n-1}$ be the error pattern. Then, the received polynomial is

$$\mathbb{r}(X) = \mathbb{v}(X) + \mathbb{e}(X). \tag{5.26}$$

Because $\mathbb{v}(X)$ is a multiple of the generator polynomial $\mathbb{g}(X)$, combining (5.19) and (5.26), we have the following relationship between the error pattern and the syndrome:

$$\mathbb{e}(X) = [\mathbb{a}(X) + \mathbb{b}(X)]\mathbb{g}(X) + \mathbb{s}(X), \tag{5.27}$$

where $\mathbb{b}(X)\mathbb{g}(X) = \mathbb{v}(X)$. This shows that the syndrome is equal to the remainder resulting from dividing the error pattern by the generator polynomial. The syndrome can be computed from the received vector; however, the error pattern $\mathbb{e}(X)$ is unknown to the decoder. Therefore, the decoder has to estimate $\mathbb{e}(X)$ based on the syndrome $\mathbb{s}(X)$. If $\mathbb{e}(X)$ is a coset leader in the standard array and if table-lookup decoding is used, $\mathbb{e}(X)$ can correctly be determined from the syndrome.

From (5.27) we see that $\mathbb{s}(X)$ is identical to zero if and only if either the error pattern $\mathbb{e}(X) = 0$ or $\mathbb{e}(X)$ is identical to a codeword. If $\mathbb{e}(X)$ is identical to a code polynomial, $\mathbb{e}(X)$ is an undetectable error pattern. Cyclic codes are very effective for detecting errors, random or burst. The error-detection circuit is simply a syndrome circuit with an OR gate whose inputs are the syndrome digits. If the syndrome is not zero, the output of the OR gate is 1, and the presence of errors has been detected.

Now, we investigate the error-detecting capability of an (n, k) cyclic code. Suppose that the error pattern $\mathbb{e}(X)$ is a burst of length $n - k$ or less (i.e., errors are confined to $n - k$ or fewer consecutive positions). Then, we can express $\mathbb{e}(X)$ in the following form:

$$\mathbb{e}(X) = X^j \mathbb{B}(X),$$

where $0 \leq j \leq n - 1$, and $\mathbb{B}(X)$ is a polynomial of degree $n - k - 1$ or less. Because the degree of $\mathbb{B}(X)$ is less than the degree of the generator polynomial $\mathbb{g}(X)$, $\mathbb{B}(X)$ is not divisible by $\mathbb{g}(X)$. Since $\mathbb{g}(X)$ is a factor of $X^n + 1$, and X is not a factor of $\mathbb{g}(X)$, $\mathbb{g}(X)$ and X^j must be relatively prime. Therefore, $\mathbb{e}(X) = X^j \mathbb{B}(X)$ is not

divisible by $\mathbb{g}(X)$. As a result, the syndrome caused by $\mathbb{e}(X)$ is not equal to zero. This implies that an (n, k) cyclic code is capable of detecting any error burst of length $n - k$ or less. For a cyclic code, an error pattern with errors confined to i high-order positions and $l - i$ low-order positions is also regarded as a burst of length l or less. Such a burst is called an *end-around* burst. For example,

$$\mathbb{e} = (\underset{\hookleftarrow}{1}\ \underset{\leftarrow}{0}\ \underset{\leftarrow}{1}\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ \underset{\hookrightarrow}{1}\ \underset{\rightarrow}{1}\ \underset{\rightarrow}{0}\ \underset{\rightarrow}{1})$$

is an end-around burst of length 7. An (n, k) cyclic code is also capable of detecting all the end-around error bursts of length $n - k$ or less (the proof of this is left as a problem). Summarizing the preceding results, we have the following property:

THEOREM 5.9 An (n, k) cyclic code is capable of detecting any error burst of length $n - k$ or less, including the end-around bursts.

In fact, a large percentage of error bursts of length $n - k + 1$ or longer can be detected. Consider the bursts of length $n - k + 1$ starting from the ith digit position and ending at the $(i + n - k)$th digit position (i.e., errors are confined to digits $e_i, e_{i+1}, \cdots, e_{i+n-k}$, with $e_i = e_{i+n-k} = 1$). There are 2^{n-k-1} such bursts. Among these bursts, the only one that cannot be detected is

$$\mathbf{e}(X) = X^i \mathbf{g}(X).$$

Therefore, the fraction of undetectable bursts of length $n - k + 1$ starting from the ith digit position is $2^{-(n-k-1)}$. This fraction applies to bursts of length $n - k + 1$ starting from any digit position (including the end-around case). Therefore, we have the following result:

THEOREM 5.10 The fraction of undetectable bursts of length $n-k+1$ is $2^{-(n-k-1)}$.

For $l > n-k+1$, there are 2^{l-2} bursts of length l starting from the ith digit position and ending at the $(i+l-1)$th digit position. Among these bursts, the undetectable ones must be of the following form:

$$\mathbf{e}(X) = X^i \mathbf{a}(X)\mathbf{g}(X),$$

where $\mathbf{a}(X) = a_0 + a_1 X + \cdots + a_{l-(n-k)-1} X^{l-(n-k)-1}$, with $a_0 = a_{l-(n-k)-1} = 1$. The number of such bursts is $2^{l-(n-k)-2}$. Therefore, the fraction of undetectable bursts of length l starting from the ith digit position is $2^{-(n-k)}$. Again, this fraction applies to bursts of length l starting from any digit position (including the end-around case), which leads to the following conclusion:

THEOREM 5.11 For $l > n - k + 1$, the fraction of undetectable error bursts of length l is $2^{-(n-k)}$.

The preceding analysis shows that cyclic codes are very effective for burst-error detection.

EXAMPLE 5.8

The (7, 4) cyclic code generated by $g(X) = 1 + X + X^3$ has a minimum distance of 3. It is capable of detecting any combination of two or fewer random errors or any burst of length 3 or less. It also detects many bursts of length greater than 3.

DECODING OF CYCLIC CODES

Decoding of cyclic codes consists of the same three steps as for decoding linear codes: syndrome computation, association of the syndrome with an error pattern, and error correction. It was shown in Section 5.4 that syndromes for cyclic codes can be computed with a division circuit whose complexity is linearly proportional to the number of parity-check digits (i.e., $n - k$). The error-correction step is simply adding (modulo-2) the error pattern to the received vector. This addition can be performed with a single EXCLUSIVE-OR gate if correction is carried out serially (i.e., one digit at a time); n EXCLUSIVE-OR gates are required if correction is carried out in parallel, as shown in Figure 3.8. The association of the syndrome with an error pattern can be completely specified by a decoding table. A straightforward approach to the design of a decoding circuit is via a combinational logic circuit that implements the table-lookup procedure; however, the limit to this approach is that the complexity of the decoding circuit tends to grow exponentially with the code length and with the number of errors that are going to be corrected. Cyclic codes have considerable algebraic and geometric properties. If these properties are properly used, decoding circuits can be simplified.

Meggitt decoder

Comments:

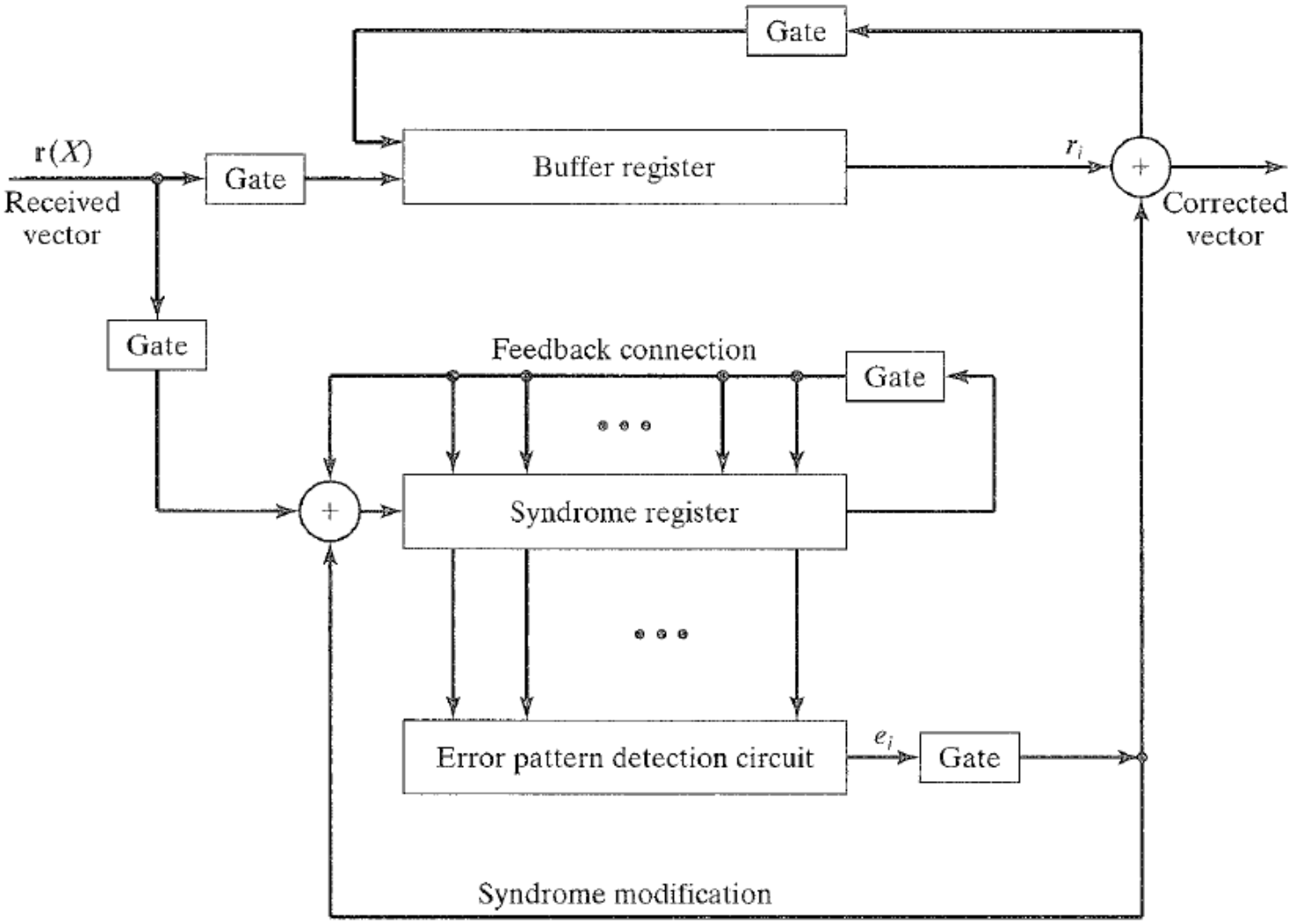


FIGURE 5.8: General cyclic code decoder with received polynomial $\mathbb{r}(X)$ shifted into the syndrome register from the left end.

Step 1. The syndrome is formed by shifting the entire received vector into the syndrome register. The received vector is simultaneously stored in the buffer register.

Step 2. The syndrome is read into the detector and is tested for the corresponding error pattern. The detector is a combinational logic circuit that is designed in such a way that its output is 1 if and only if the syndrome in the syndrome register corresponds to a correctable error pattern with an error at the highest-order position X^{n-1}. That is, if a 1 appears at the output of the detector, the received symbol in the rightmost stage of the buffer register is assumed to be erroneous and must be corrected; if a 0 appears at the output of the detector, the received symbol at the rightmost stage of the buffer register is assumed to be error-free, and no correction is necessary. Thus, the output of the detector is the estimated error value for the symbol to come out of the buffer.

Step 3. The first received symbol is read out of the buffer. At the same time, the syndrome register is shifted once. If the first received symbol is detected to be an erroneous symbol, it is then corrected by the output of the detector. The output of the detector is also fed back to the syndrome register to modify the syndrome (i.e., to remove the error effect from the syndrome). This operation results in a new syndrome, which corresponds to the altered received vector shifted one place to the right.

Step 4. The new syndrome formed in step 3 is used to detect whether the second received symbol (now at the rightmost stage of the buffer register) is an erroneous symbol. The decoder repeats steps 2 and 3. The second received symbol is corrected in exactly the same manner as the first received symbol was corrected.

Step 5. The decoder decodes the received vector symbol by symbol in the manner outlined until the entire received vector is read out of the buffer register.

Flowchart:

EXAMPLE 5.9

Consider the decoding of the (7, 4) cyclic code generated by $g(X) = 1 + X + X^3$. This code has a minimum distance of 3 and is capable of correcting any single error over a block of seven digits. There are seven single-error patterns. These seven error patterns and the all-zero vector form all the coset leaders of the decoding table. Thus, they form all the correctable error patterns. Suppose that the received polynomial $r(X) = r_0 + r_1X + r_2X^2 + r_3X^3 + r_4X^4 + r_5X^5 + r_6X^6$ is shifted into the syndrome register from the left end. The seven single-error patterns and their corresponding syndrome are listed in Table 5.4.

TABLE 5.4: Error patterns and their syndromes with the received polynomial $r(X)$ shifted into the syndrome register from the left end.

Error pattern $e(X)$	Syndrome $s(X)$	Syndrome vector (s_0, s_1, s_2)
$e_6(X) = X^6$	$s(X) = 1 + X^2$	(1 0 1)
$e_5(X) = X^5$	$s(X) = 1 + X + X^2$	(1 1 1)
$e_4(X) = X^4$	$s(X) = X + X^2$	(0 1 1)
$e_3(X) = X^3$	$s(X) = 1 + X$	(1 1 0)
$e_2(X) = X^2$	$s(X) = X^2$	(0 0 1)
$e_1(X) = X^1$	$s(X) = X$	(0 1 0)
$e_0(X) = X^0$	$s(X) = 1$	(1 0 0)

We see that $\mathbf{e}_6(X) = X^6$ is the only error pattern with an error at location X^6. When this error pattern occurs, the syndrome in the syndrome register will be (1 0 1) after the entire received polynomial $\mathbf{r}(X)$ has entered the syndrome register. The detection of this syndrome indicates that r_6 is an erroneous digit and must be corrected. Suppose that the single error occurs at location X^i [i.e., $\mathbf{e}_i(X) = X^i$] for $0 \leq i < 6$. After the entire received polynomial has been shifted into the syndrome register, the syndrome in the register will not be (1 0 1); however, after another $6 - i$ shifts, the contents in the syndrome register will be (1 0 1), and the next received digit to come out of the buffer register will be the erroneous digit. Therefore, only the syndrome (1 0 1) needs to be detected, and this can be accomplished with a single three-input AND gate. The complete decoding circuit is shown in Figure 5.9. Figure 5.10 illustrates the decoding process. Suppose that the codeword $\mathbf{v} = (1\,0\,0\,1\,0\,1\,1)$ [or $\mathbf{v}(X) = 1 + X^3 + X^5 + X^6$] is transmitted and $\mathbf{r} = (1\,0\,1\,1\,0\,1\,1)$ [or $\mathbf{r}(X) = 1 + X^2 + X^3 + X^5 + X^6$] is received. A single error occurs at location X^2. When the entire received polynomial has been shifted into the syndrome and buffer registers, the syndrome register contains (0 0 1). In Figure 5.10, the contents in the syndrome register and the contents in the buffer register are recorded after each shift. Also, there is a pointer to indicate the error location after each shift. We see that after four more shifts the contents in the syndrome register are (1 0 1), and the erroneous digit r_2 is the next digit to come out from the buffer register.

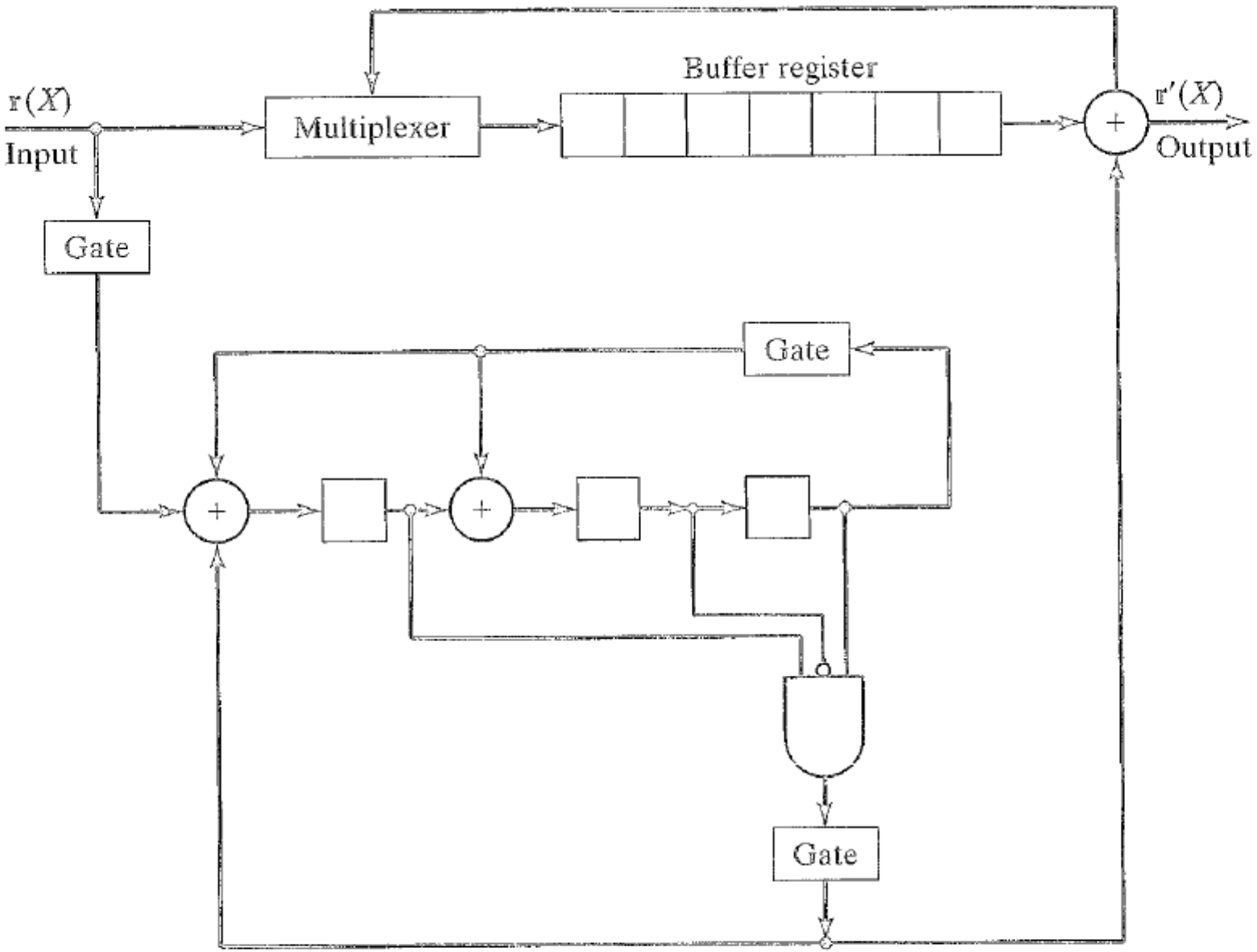


FIGURE 5.9: Decoding circuit for the $(7,4)$ cyclic code generated by $g(X) = 1+X+X^3$.

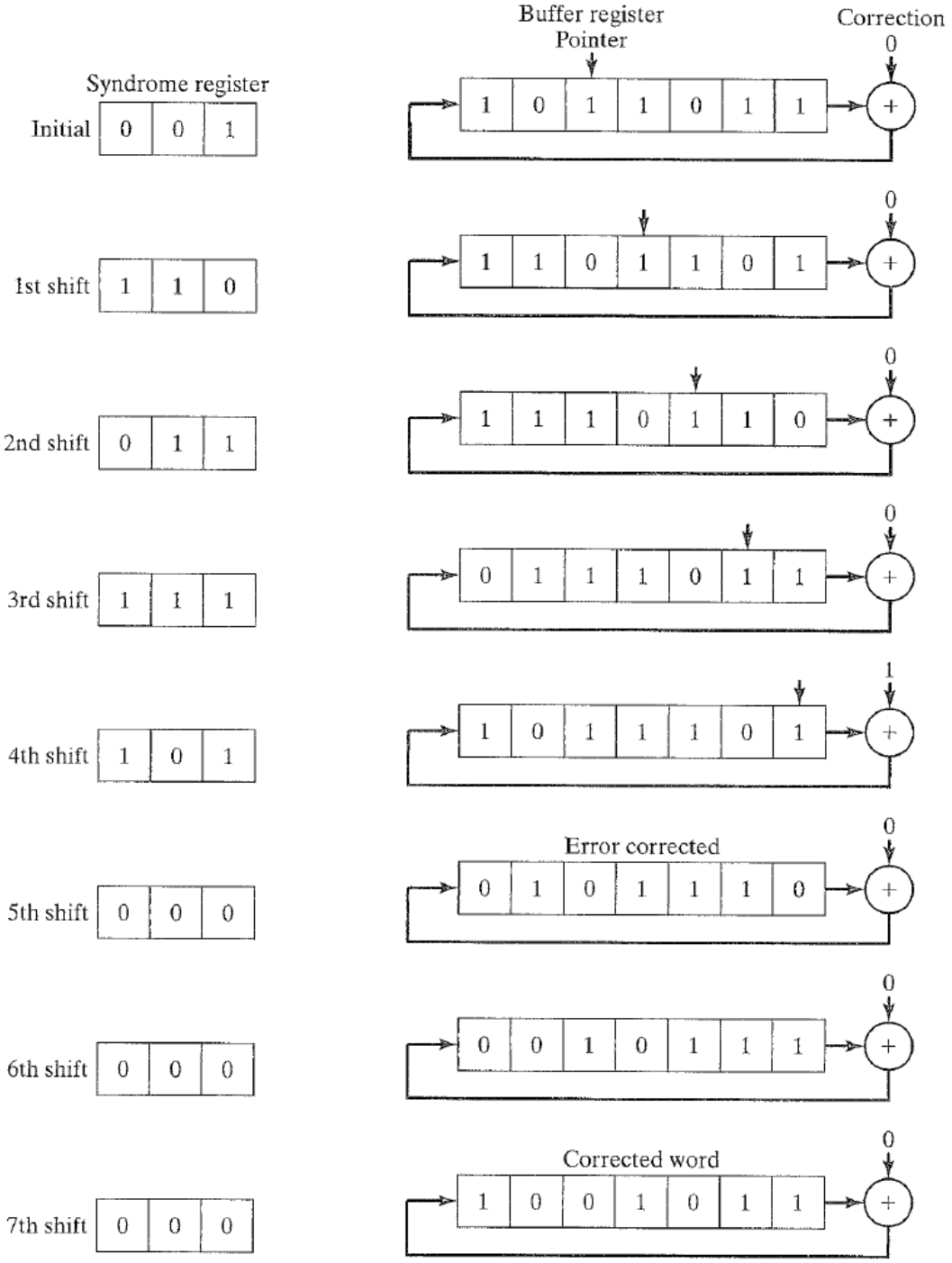


FIGURE 5.10: Error-correction process of the circuit shown in Figure 5.9.

The (7, 4) cyclic code considered in Example 5.9 is the same code considered in Example 3.9. Comparing the decoding circuit shown in Figure 3.9 with the decoding circuit shown in Figure 5.9, we see that the circuit shown in Figure 5.9 is simpler than the circuit shown in Figure 3.9. Thus, the cyclic structure does simplify the decoding circuit; however, the circuit shown in Figure 5.9 takes a longer time to decode a received vector because the decoding is carried out serially. In general, there is a trade-off between speed and simplicity, as they cannot be achieved at the same time.

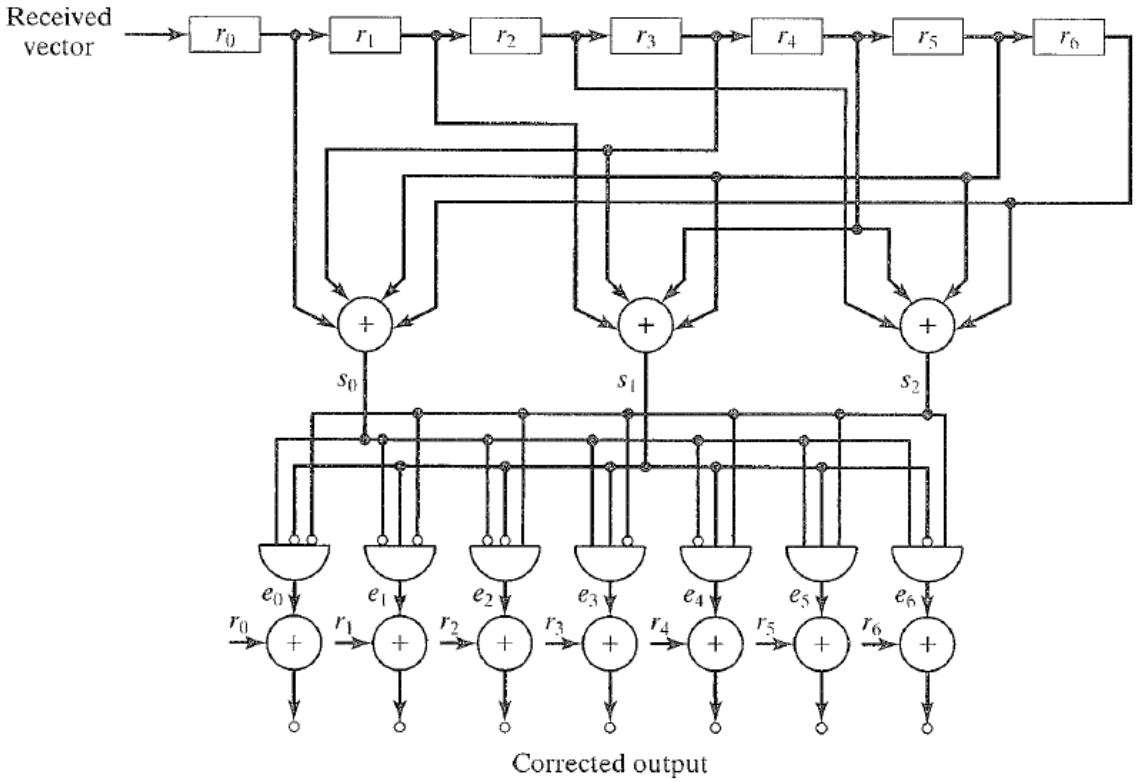


FIGURE 3.9: Decoding circuit for the (7, 4) code given in Table 3.1.

To decode a cyclic code, the received polynomial $\mathbb{r}(X)$ may be shifted into the syndrome register from the right end for computing the syndrome. When $\mathbb{r}(X)$ has been shifted into the syndrome register, the register contains $\mathbb{s}^{(n-k)}(X)$, which is the syndrome of $\mathbb{r}^{(n-k)}(X)$, the $(n-k)$th cyclic shift of $\mathbb{r}(X)$. If $\mathbb{s}^{(n-k)}(X)$ corresponds to an error pattern $\mathbb{e}(X)$ with $e_{n-1} = 1$, the highest-order digit r_{n-1} of $\mathbb{r}(X)$ is erroneous and must be corrected. In $\mathbb{r}^{(n-k)}(X)$, the digit r_{n-1} is at the location X^{n-k-1}. When r_{n-1} is corrected, the error effect must be removed from $\mathbb{s}^{(n-k)}(X)$. The new syndrome, denoted by $\mathbb{s}_1^{(n-k)}(X)$, is the sum of $\mathbb{s}^{(n-k)}(X)$ and the remainder $\wp(X)$ resulting from dividing X^{n-k-1} by the generator polynomial $\mathbb{g}(X)$. Because the degree of X^{n-k-1} is less than the degree of $\mathbb{g}(X)$,

$$\wp(X) = X^{n-k-1}.$$

Therefore,

$$\mathbb{s}_1^{(n-k)}(X) = \mathbb{s}^{(n-k)}(X) + X^{n-k-1},$$

which indicates that the effect of an error at the location X^{n-1} on the syndrome can be removed by feeding the error digit into the syndrome register from the right end through an EXCLUSIVE-OR gate, as shown in Figure 5.11. The decoding process of the decoder shown in Figure 5.11 is identical to the decoding process of the decoder shown in Figure 5.8.

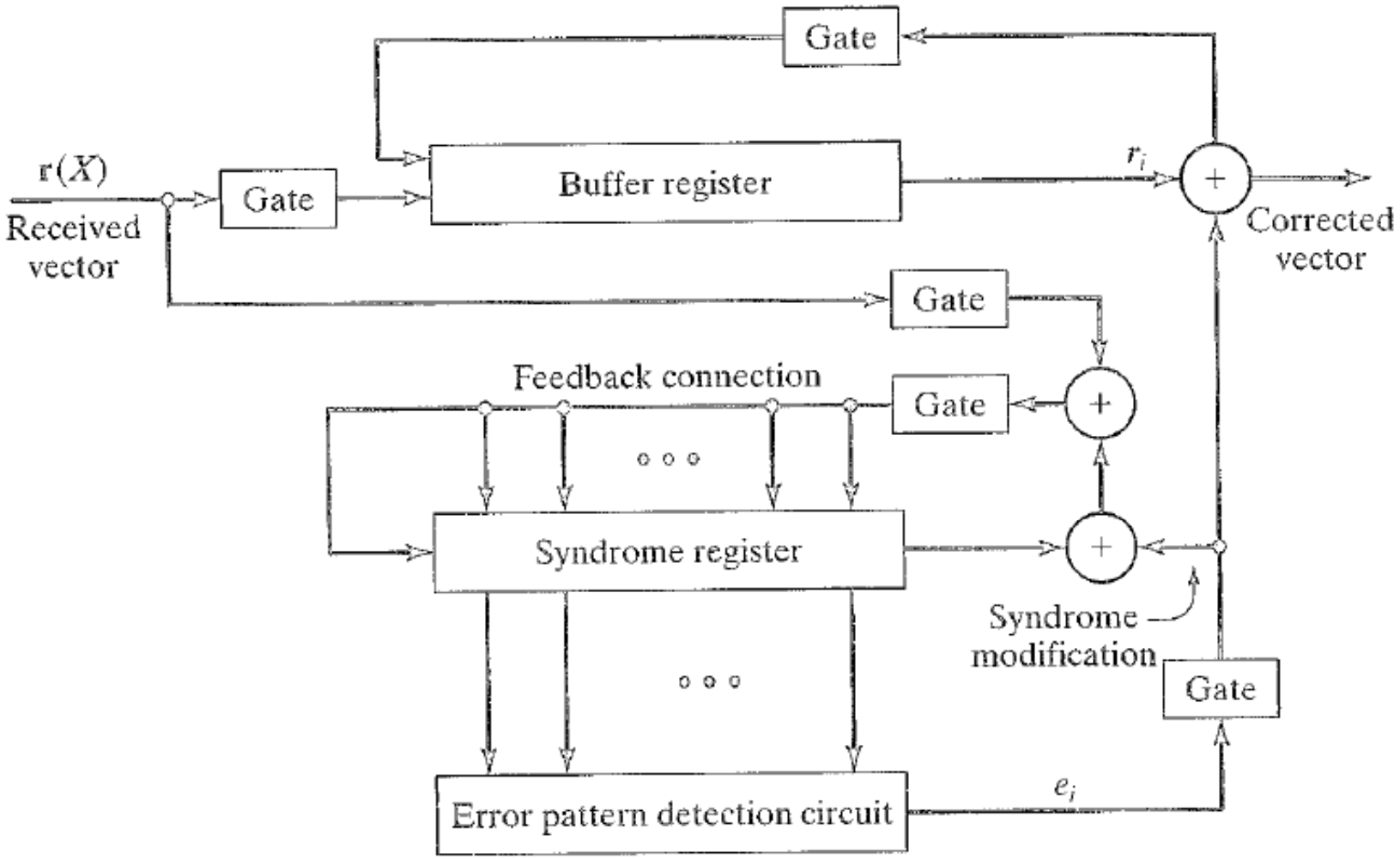


FIGURE 5.11: General cyclic code decoder with received polynomial $\mathbb{r}(X)$ shifted into the syndrome register from the right end.

EXAMPLE 5.10

Again, we consider the decoding of the (7, 4) cyclic code generated by $g(X) = 1 + X + X^3$. Suppose that the received polynomial $r(X)$ is shifted into the syndrome register from the right end. The seven single-error patterns and their corresponding syndromes are listed in Table 5.5.

We see that only when $e(X) = X^6$ occurs, the syndrome is (0 0 1) after the entire polynomial $r(X)$ has been shifted into the syndrome register. If the single error occurs at the location X^i with $i \neq 6$, the syndrome in the register will not be (0 0 1) after the entire received polynomial $r(X)$ has been shifted into the syndrome register; however, after another $6 - i$ shifts, the syndrome register will contain (0 0 1). Based on this result, we obtain another decoding circuit for the (7, 4) cyclic code generated by $g(X) = 1 + X + X^3$, as shown in Figure 5.12. We see that the circuit shown in Figure 5.9 and the circuit shown in Figure 5.12 have the same complexity.

TABLE 5.5: Error patterns and their syndromes with the received polynomial $r(X)$ shifted into the syndrome register from the right end.

Error pattern $e(X)$	Syndrome $s^{(3)}(X)$	Syndrome vector (s_0, s_1, s_2)
$e(X) = X^6$	$s^{(3)}(X) = X^2$	(0 0 1)
$e(X) = X^5$	$s^{(3)}(X) = X$	(0 1 0)
$e(X) = X^4$	$s^{(3)}(X) = 1$	(1 0 0)
$e(X) = X^3$	$s^{(3)}(X) = 1 + X^2$	(1 0 1)
$e(X) = X^2$	$s^{(3)}(X) = 1 + X + X^2$	(1 1 1)
$e(X) = X$	$s^{(3)}(X) = X + X^2$	(0 1 1)
$e(X) = X^0$	$s^{(3)}(X) = 1 + X$	(1 1 0)

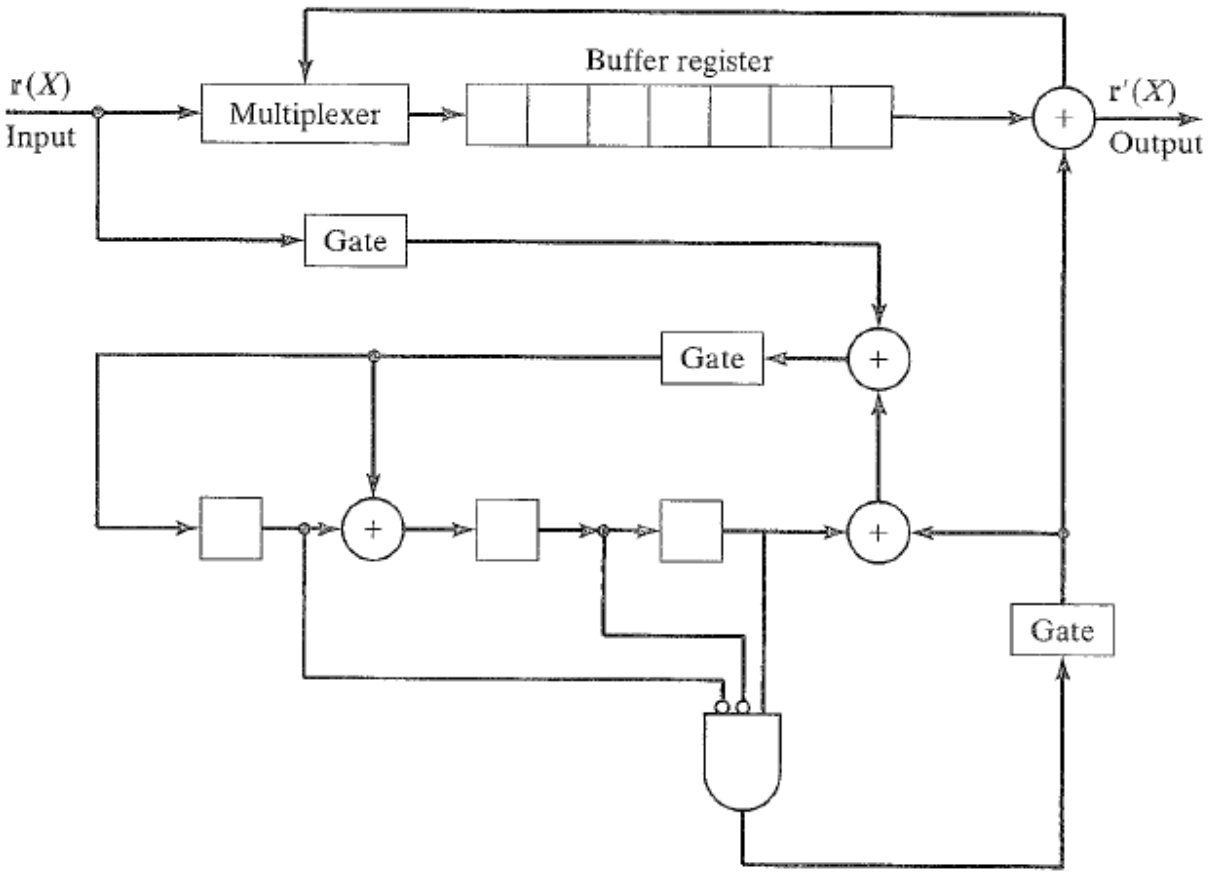


FIGURE 5.12: Decoding circuit for the (7, 4) cyclic code generated by $g(X) = 1 + X + X^3$.

CYCLIC HAMMING CODES

The Hamming codes presented in Section 4.1 can be put in cyclic form. A cyclic Hamming code of length $2^m - 1$ with $m \geq 3$ is generated by a primitive polynomial $\mathfrak{p}(X)$ of degree m.

$$\mathbb{G} = \begin{bmatrix} b_{00} & b_{01} & b_{02} & \cdots & b_{0,n-k-1} & 1 & 0 & 0 & \cdots & 0 \\ b_{10} & b_{11} & b_{12} & \cdots & b_{1,n-k-1} & 0 & 1 & 0 & \cdots & 0 \\ b_{20} & b_{21} & b_{22} & \cdots & b_{2,n-k-1} & 0 & 0 & 1 & \cdots & 0 \\ & \vdots & & & \vdots & & & & & \vdots \\ b_{k-1,0} & b_{k-1,1} & b_{k-1,2} & \cdots & b_{k-1,n-k-1} & 0 & 0 & 0 & \cdots & 1 \end{bmatrix}$$

$$\mathbb{H} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & b_{00} & b_{10} & b_{20} & \cdots & b_{k-1,0} \\ 0 & 1 & 0 & \cdots & 0 & b_{01} & b_{11} & b_{21} & \cdots & b_{k-1,1} \\ 0 & 0 & 1 & \cdots & 0 & b_{02} & b_{12} & b_{22} & \cdots & b_{k-1,2} \\ & \vdots & & & & & \vdots & & & \vdots \\ 0 & 0 & 0 & \cdots & 1 & b_{0,n-k-1} & b_{1,n-k-1} & b_{2,n-k-1} & \cdots & b_{k-1,n-k-1} \end{bmatrix}.$$

In the following we show that the cyclic code defined previously is indeed a Hamming code, by examining its parity-check matrix in systematic form. The method presented in Section 5.2 is used to form the parity-check matrix. Dividing X^{m+i} by the generator polynomial $\mathbb{p}(X)$ for $0 \leq i < 2^m - m - 1$, we obtain

$$X^{m+i} = \mathbb{a}_i(X)\mathbb{p}(X) + \mathbb{b}_i(X), \tag{5.28}$$

where the remainder $\mathbb{b}_i(X)$ is of the form

$$\mathbb{b}_i(X) = b_{i0} + b_{i1}X + \cdots + b_{i,m-1}X^{m-1}.$$

Because X is not a factor of the primitive polynomial $\mathbb{p}(X)$, X^{m+i} and $\mathbb{p}(X)$ must be relatively prime. As a result, $\mathbb{b}_i(X) \neq 0$. Moreover, $\mathbb{b}_i(X)$ consists of at least two terms. Suppose that $\mathbb{b}_i(X)$ has only one term, say X^j with $0 \leq j < m$. It follows from (5.28) that

$$X^{m+i} = \mathbb{a}_i(X)\mathbb{p}(X) + X^j.$$

Rearranging the preceding equation, we have

$$X^j(X^{m+i-j} + 1) = \mathbb{a}_i(X)\mathbb{p}(X).$$

Because X^j and $\mathbb{p}(X)$ are relatively prime, the foregoing equation implies that $\mathbb{p}(X)$ divides $X^{m+i-j}+1$; however, this is impossible since $m+i-j<2^m-1$, and $\mathbb{p}(X)$ is a primitive polynomial of degree m. [Recall that the smallest positive integer n such that $\mathbb{p}(X)$ divides X^n+1 is 2^m-1.] Therefore, for $0 \le i < 2^m-m-1$, the remainder $\mathbb{b}_i(X)$ contains at least two terms. Next we show that for $i \ne j$, $\mathbb{b}_i(X) \ne \mathbb{b}_j(X)$. From (5.28) we obtain

$$\mathbb{b}_i(X) + X^{m+i} = \mathbb{a}_i(X)\mathbb{p}(X),$$

$$\mathbb{b}_j(X) + X^{m+j} = \mathbb{a}_j(X)\mathbb{p}(X).$$

Suppose that $\mathbb{b}_i(X) = \mathbb{b}_j(X)$. Assuming that $i < j$ and combining the preceding two equations above we obtain the following relation:

$$X^{m+i}(X^{j-i}+1) = [\mathbb{a}_i(X) + \mathbb{a}_j(X)]\mathbb{p}(X).$$

This equation implies that $\mathbb{p}(X)$ divides $X^{j-i}+1$, but this is impossible, since $i \ne j$ and $j-i<2^m-1$. Therefore, $\mathbb{b}_i(X) \ne \mathbb{b}_j(X)$.

Let $\mathbb{H} = [\mathbb{I}_m \ \mathbb{Q}]$ be the parity-check matrix (in systematic form) of the cyclic code generated by $\mathbb{p}(X)$, where $\mathbb{I}_m$ is an $m \times m$ identity matrix and $\mathbb{Q}$ is an $m \times (2^m - m - 1)$ matrix. Let $\mathbb{b}_i = (b_{i0}, b_{i1}, \cdots, b_{i,m-1})$ be the m-tuple corresponding to $\mathbb{b}_i(X)$. It follows from (5.17) that the matrix $\mathbb{Q}$ has the $2^m - m - 1$ $\mathbb{b}_i$'s with $0 \leq i < 2^m - m - 1$ as all its columns. It follows from the preceding analysis that no two columns of $\mathbb{Q}$ are alike, and each column has at least two 1's. Therefore, the matrix $\mathbb{H}$ is indeed a parity-check matrix of a Hamming code, and $\mathbb{p}(X)$ generates this code.

The polynomial $\mathbb{p}(X) = 1 + X + X^3$ is a primitive polynomial. Therefore, the (7, 4) cyclic code generated by $\mathbb{p}(X) = 1 + X + X^3$ is a Hamming code. A list of primitive polynomials with degree ≥ 3 is given in Table 2.7.

Cyclic Hamming codes can easily be decoded. To devise the decoding circuit, all we need to know is how to decode the first received digit. All the other received digits will be decoded in the same manner and with the same circuitry. Suppose that a single error has occurred at the highest-order position, X^{2^m-2}, of the received vector $\mathbb{r}(X)$ [i.e., the error polynomial is $\mathbb{e}(X) = X^{2^m-2}$]. Suppose that $\mathbb{r}(X)$ is shifted into the syndrome register from the right end. After the entire $\mathbb{r}(X)$ has entered the register, the syndrome in the register is equal to the remainder resulting from dividing $X^m \cdot X^{2^m-2}$ (the error polynomial preshifted m times) by the generator polynomial $\mathbb{p}(X)$. Because $\mathbb{p}(X)$ divides $X^{2^m-1}+1$, the syndrome is of the following form:

$$\mathbb{s}(X) = X^{m-1}.$$

Therefore, if a single error occurs at the highest-order location of $\mathbb{r}(X)$, the resultant syndrome is $(0, 0, \cdots, 0, 1)$. If a single error occurs at any other location of

$\mathbb{r}(X)$, the resultant syndrome will be different from $(0, 0, \cdots, 0, 1)$. Based on this result only a single m-input AND gate is needed to detect the syndrome pattern $(0, 0, \cdots, 0, 1)$. The inputs to this AND gate are $s'_0, s'_1, \cdots, s'_{m-2}$ and s_{m-1}, where s_i is a syndrome digit and s'_i denotes its complement. A complete decoding circuit for a cyclic Hamming code is shown in Figure 5.13. The decoding operation is described in the following steps:

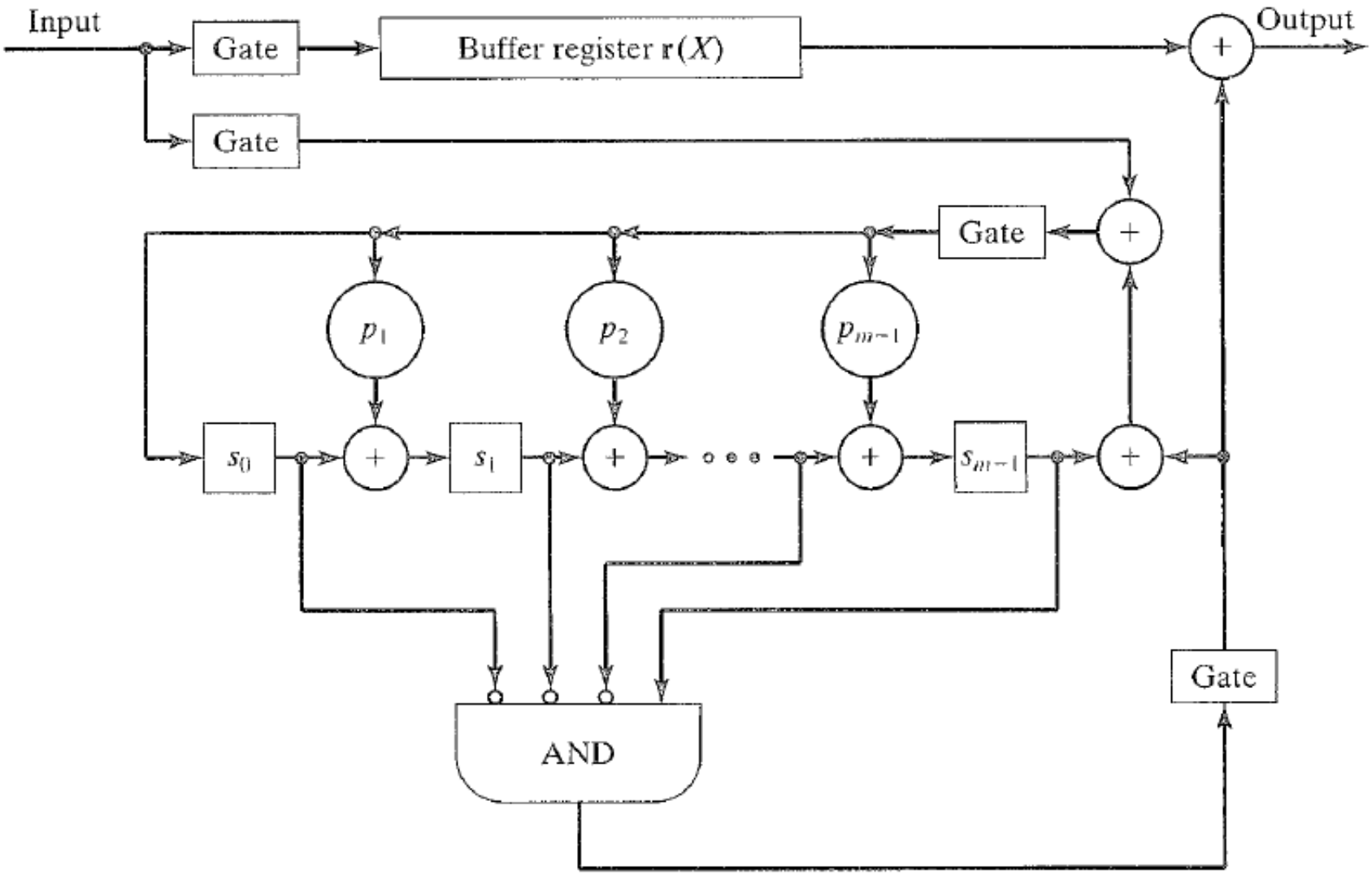


FIGURE 5.13: Decoder for a cyclic Hamming code.

Step 1. The syndrome is obtained by shifting the entire received vector into the syndrome register. At the same time, the received vector is stored in the buffer register. If the syndrome is zero, the decoder assumes that no error has occurred, and no correction is necessary. If the syndrome is not zero, the decoder assumes that a single error has occurred.

Step 2. The received word is read out of the buffer register digit by digit. As each digit is read out of the buffer register the syndrome register is shifted cyclically once. As soon as the syndrome in the register is $(0, 0, 0, \cdots, 0, 1)$ the next digit to come out of the buffer is the erroneous digit, and the output of the m-input AND gate is 1.

Step 3. The erroneous digit is read out of the buffer register and is corrected by the output of the m-input AND gate. The correction is accomplished by an EXCLUSIVE-OR gate.

Step 4. The syndrome register is reset to zero after the entire received vector is read out of the buffer.

The cyclic Hamming code presented here can be modified to correct any single error and simultaneously to detect any combination of double errors. Let

Comments:

ERROR-TRAPPING DECODING

$$\mathbf{r}(X) = \mathbf{v}(X) + \mathbf{e}(X)$$

$$\mathbf{e}(X) = \mathbf{a}(X)\mathbf{g}(X) + \mathbf{s}(X)$$

$$\mathbf{e}(X) = e_k X^k + e_{k+1} X^{k+1} + \cdots + e_{n-1} X^{n-1}$$

If $\mathbf{r}(X)$ is cyclically shifted $n-k$ times, the errors will be confined to $n-k$ low-order parity positions, $X^0, X^1, \cdots, X^{n-k-1}$ of $\mathbf{r}^{(n-k)}(X)$. The corresponding error pattern is then

$$\mathbf{e}^{(n-k)}(X) = e_k + e_{k+1} X + \cdots + e_{n-1} X^{n-k-1}.$$

Because the syndrome $\mathbf{s}^{(n-k)}(X)$ of $\mathbf{r}^{(n-k)}(X)$ is equal to the remainder resulting from dividing $\mathbf{e}^{(n-k)}(X)$ by $\mathbf{g}(X)$ and since the degree of $\mathbf{e}^{(n-k)}(X)$ is less than $n-k$, we obtain the following equality:

$$\mathbf{s}^{(n-k)}(X) = \mathbf{e}^{(n-k)}(X) = e_k + e_{k+1} X + \cdots + e_{n-1} X^{n-k-1}.$$

Multiplying $\mathbf{s}^{(n-k)}(X)$ by X^k, we have

$$\begin{aligned} X^k \mathbf{s}^{(n-k)}(X) &= \mathbf{e}(X) \\ &= e_k X^k + e_{k+1} X^{k+1} + \cdots + e_{n-1} X^{n-1}. \end{aligned}$$

This result says that if errors are confined to the $n - k$ high-order positions of the received polynomial $\mathbb{r}(X)$, the error pattern $\mathbb{e}(X)$ is identical to $X^k \mathbb{s}^{(n-k)}(X)$, where $\mathbb{s}^{(n-k)}(X)$ is the syndrome of $\mathbb{r}^{(n-k)}(X)$, the $(n - k)$th cyclic shift of $\mathbb{r}(X)$. When this event occurs, we simply compute $\mathbb{s}^{(n-k)}(X)$ and add $X^k \mathbb{s}^{(n-k)}(X)$ to $\mathbb{r}(X)$. The resultant polynomial is the transmitted code polynomial.

Comments:

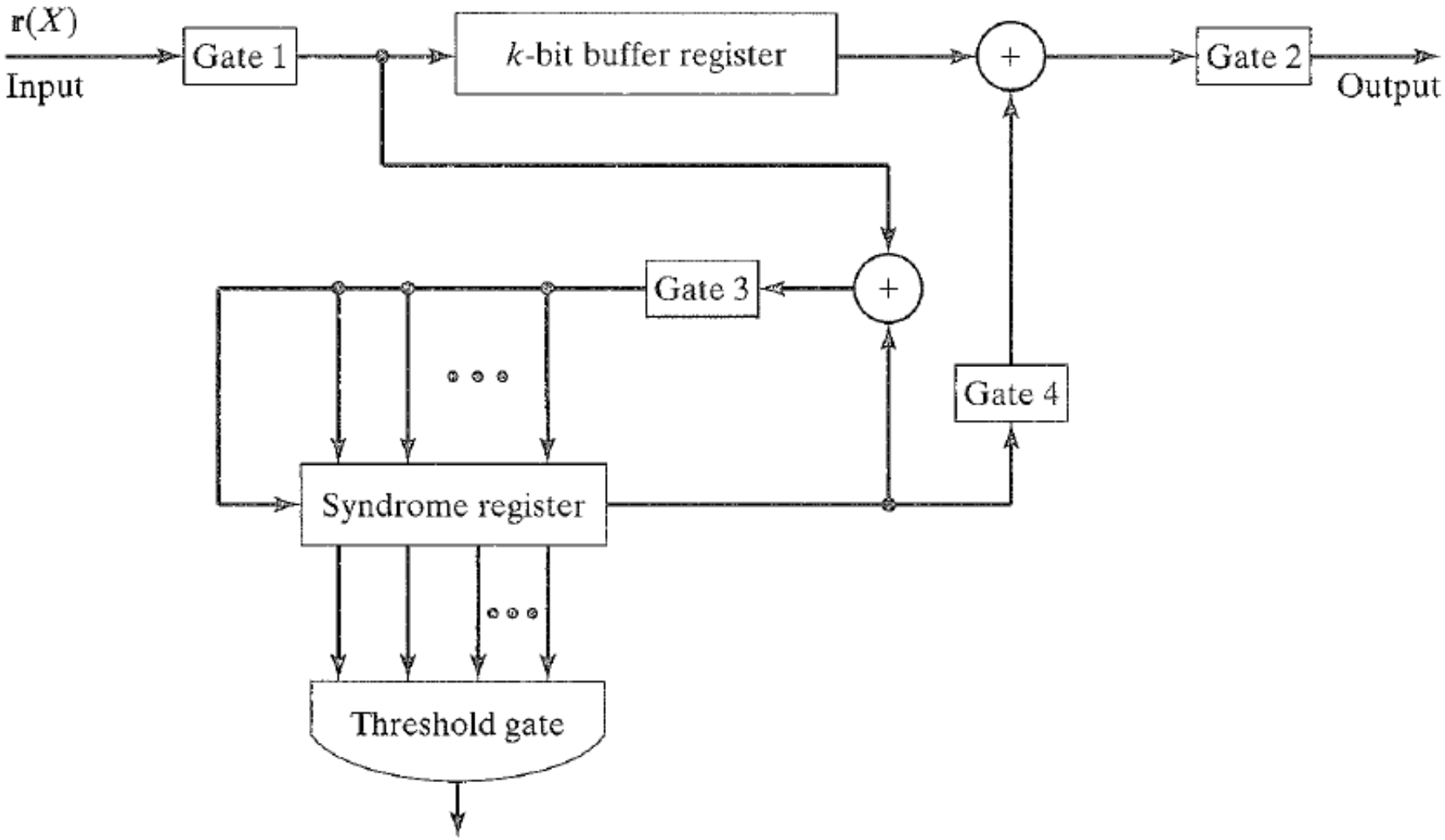


FIGURE 5.14: Error-trapping decoder.

Step 1. The received polynomial $\mathfrak{r}(X)$ is shifted into the buffer and syndrome registers simultaneously with gates 1 and 3 turned on and all the other gates turned off. Because we are interested only in the recovery of the k information digits, the buffer register has to store only the k received information digits.

Step 2. As soon as the entire $\mathfrak{r}(X)$ has been shifted into the syndrome register, the weight of the syndrome in the register is tested by an $(n-k)$-input *threshold* gate whose output is 1 when t or fewer of its inputs are 1; otherwise, it is zero.

a. If the weight of the syndrome is t or less, the syndrome digits in the syndrome register are identical to the error digits at the $n-k$ high-order positions $X^k, X^{k+1}, \cdots, X^{n-1}$ of $\mathfrak{r}(X)$. Now, gates 2 and 4 are turned on and the other gates are turned off. The received vector is read out of the buffer register one digit at a time and is corrected by the error digits shifted out from the syndrome register.

b. If the weight of the syndrome is greater than t, the errors are *not* confined to the $n-k$ high-order positions of $\mathfrak{r}(X)$, and they have not been trapped in the syndrome register. Go to step 3.

Step 3. The syndrome register is cyclically shifted once with gate 3 turned on and other gates turned off. The weight of the new syndrome is tested. (a) If it is t or less, the errors are confined to the locations $X^{k-1}, X^k, \cdots, X^{n-2}$ of $r(X)$, and the contents in the syndrome register are identical to the errors at these locations. Because the first received digit r_{n-1} is *error-free*, it is read out of the buffer register with gate 2 turned on. As soon as r_{n-1} has been read out, gate 4 is turned on and gate 3 is turned off. The contents in the syndrome register are shifted out and are used to correct the next $n-k$ received digits to come out from the buffer register. (b) If the weight of the syndrome is greater than t, the syndrome register is shifted once more with gate 3 turned on.

Step 4. The syndrome register is continuously shifted until the weight of its contents drops to t or less. If the weight goes down to t or less at the end of the ith shift, for $1 \leq i \leq k$, the first i received digits, $r_{n-i}, r_{n-i+1}, \cdots, r_{n-1}$, in the buffer are *error-free*, and the contents in the syndrome register are identical to the errors at the locations $X^{k-i}, X^{k-i+1}, \cdots, X^{n-i-1}$. As soon as the i error-free received digits have been read out of the buffer register, the contents in the syndrome register are shifted out and are used to correct the next $n-k$ received digit to come out from the buffer register. When the k received information digits have been read out of the buffer register and have been corrected, gate 2 is turned off. Any nonzero digits left in the syndrome register are errors in the parity part of $r(X)$, and they will be ignored.

Step 5. If the weight of the syndrome never goes down to t or less by the time the syndrome register has been shifted k times, either an error pattern with errors confined to $n-k$ consecutive end-around locations has occurred or an uncorrectable error pattern has occurred. The syndrome register keeps being shifted. Suppose that the weight of its contents becomes t or less at the end of $k+l$ shifts with $1 \leq l \leq n-k$. Then, errors are confined to the $n-k$ consecutive end-around locations, $X^{n-l}, X^{n-l+1}, \cdots, X^{n-1}, X^0, X^1, \cdots, X^{n-k-l-1}$ of $\mathbb{r}(X)$. The l digits in the l leftmost stages of the syndrome register match the errors at the l high-order locations $X^{n-l}, X^{n-l+1}, \cdots, X^{n-1}$ of $\mathbb{r}(X)$. Because the errors at the $n-k-l$ parity locations are not needed, the syndrome register is shifted $n-k-l$ times with all the gates turned off. Now, the l errors at the locations $X^{n-l}, X^{n-l+1}, \cdots, X^{n-1}$ of $\mathbb{r}(X)$ are contained in the l rightmost stages of the syndrome register. With gates 2 and 4 turned on and other gates turned off, the received digits in the buffer register are read out and are corrected by the corresponding error digits shifted out from the syndrome register. This completes the decoding operation.

It is possible to implement the error-trapping decoding in a different manner so that the error patterns with errors confined to $n-k$ consecutive end-around locations can be corrected as fast as possible. This can be done by shifting the received vector $\mathbf{r}(X)$ into the syndrome register from the *left* end, as shown in Figure 5.15. This variation is based on the following facts. If the errors are confined to $n-k$ low-order parity positions $X^0, X^1, \cdots, X^{n-k-1}$ of $\mathbf{r}(X)$, then after the entire $\mathbf{r}(X)$ has entered the syndrome register, the contents in the register are identical to the error digits at the locations $X^0, X^1, \cdots, X^{n-k-1}$ of $\mathbf{r}(X)$. Suppose that the errors are not confined to the $n-k$ low-order positions of $\mathbf{r}(X)$ but are confined to $n-k$ consecutive locations (including the end-around case), say $X^i, X^{i+1}, \cdots, X^{(n-k)+i-1}$. After $n-i$ cyclic shifts of $\mathbf{r}(X)$, the errors will be shifted to the $n-k$ low-order positions of $\mathbf{r}^{(n-i)}(X)$, and the syndrome of $\mathbf{r}^{(n-i)}(X)$ will be identical to the errors confined to positions $X^i, X^{i+1}, \cdots, X^{(n-k)+i-1}$ of $\mathbf{r}(X)$. The operation of the decoder shown in Figure 5.15 is as follows:

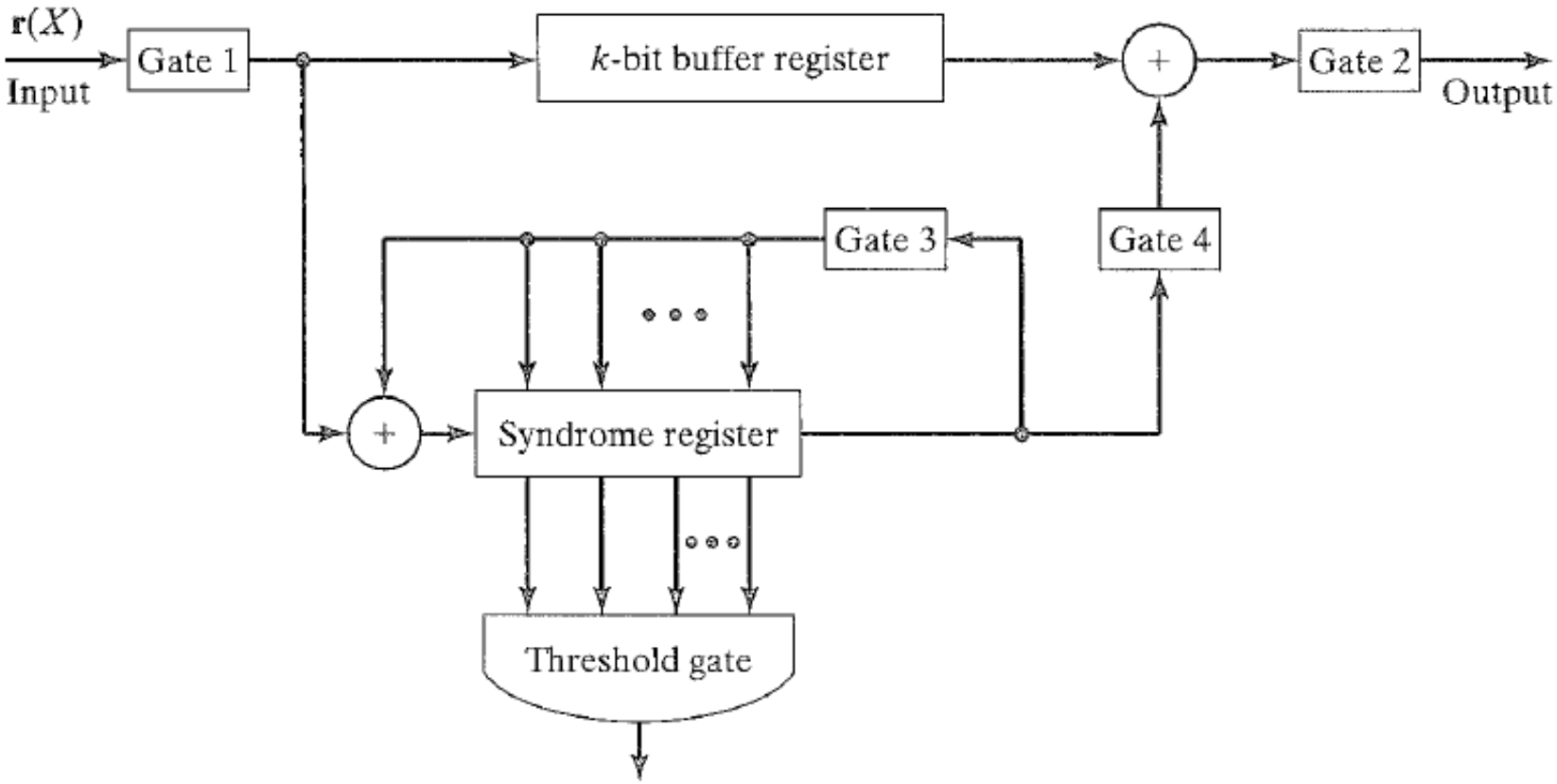


FIGURE 5.15: Another error-trapping decoder.

Step 1. Gates 1 and 3 are turned on and the other gates are turned off. The received vector $\mathbb{r}(X)$ is shifted into the syndrome register and simultaneously into the buffer register (only the k received information digits

are stored in the buffer register). As soon as the entire $\mathbb{r}(X)$ has been shifted into the syndrome register, the contents of the register form the syndrome $\mathbb{s}(X)$ of $\mathbb{r}(X)$.

Step 2. The weight of the syndrome is tested. (a) *If the weight is t or less*, the errors are confined to the $(n-k)$ low-order parity positions $X^0, X^1, \cdots, X^{n-k-1}$ of $\mathbb{r}(X)$. Thus, the k received information digits in the buffer register are *error-free*. Gate 2 is then turned on, and the error-free information digits are read out of the buffer with gate 4 turned off. (b) If the weight of the syndrome is greater than t, the syndrome register is then shifted once with gate 3 turned on and the other gates turned off. Go to step 3.

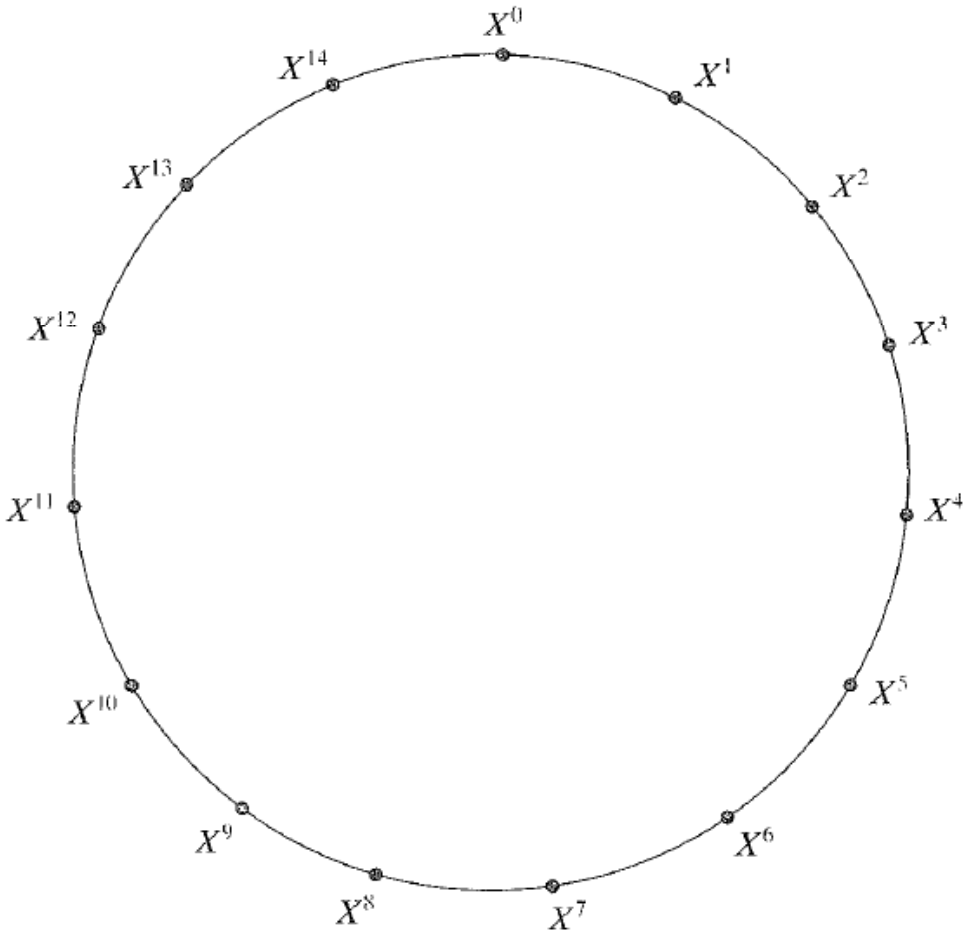
Step 3. The weight of the new contents in the syndrome register is tested. (a) If the weight is t or less, the errors are confined to the position $X^{n-1}, X^0, X^1, \cdots, X^{n-k-2}$ of $r(X)$ (end-around case). The leftmost digit in the syndrome register is identical to the error at the position X^{n-1} of $r(X)$; the other $n-k-1$ digits in the syndrome register match the errors at parity positions $X^0, X^1, \cdots, X^{n-k-2}$ of $r(X)$. The output of the threshold gate turns gate 3 off and sets a clock to count from 2. The syndrome register is then shifted (in step with the clock) with gate 3 turned off. As soon as the clock has counted to $n-k$, the contents of the syndrome register will be $(0\,0\cdots 0\,1)$. The rightmost digit matches the error at position X^{n-1} of $r(X)$. The k received information digits are then read out of the buffer, and the first received information digit is corrected by the 1 coming out from the syndrome register. The decoding is thus completed. (b) If the weight of the contents in the syndrome register is greater than t, the syndrome register is shifted once again with gate 3 turned on and other gates turned off. Go to step 4.

Step 4. Step 3(b) repeats until the weight of the contents of the syndrome register drop to t or less. If the weight drops to t or less after the ith shift, for $1 \leq i \leq n-k$, the clock starts to count from $i+1$. At the same time the syndrome register is shifted with gate 3 turned off. As soon as the clock has counted to $n-k$, the rightmost i digits in the syndrome register match the errors in the first i received information digits in the buffer register. The other information digits are error-free. Gates 2 and 4 are then turned on. The received information digits are read out of the buffer for correction.

Step 5. If the weight of the contents of the syndrome register never drops to t or less by the time that the syndrome register has been shifted $n - k$ times (with gate 3 turned on), gate 2 is then turned on, and the received information digits are read out of the buffer one at a time. At the same time the syndrome register is shifted with gate 3 turned on. As soon as the weight of the contents of the syndrome register drops to t or less, the contents match the errors in the next $n - k$ digits to come out of the buffer. Gate 4 is then turned on, and the erroneous information digits are corrected by the digits coming out from the syndrome register with gate 3 turned off. Gate 2 is turned off as soon as k information digits have been read out of the buffer.

EXAMPLE 5.11

Consider the (15, 7) cyclic code generated by $\mathbf{g}(X) = 1 + X^4 + X^6 + X^7 + X^8$. This code has a minimum distance of $d_{min} = 5$, which will be proved in Chapter 6. Hence, the code is capable of correcting any combination of two or fewer errors over a block of 15 digits. Suppose that we decode this code with an error-trapping decoder.



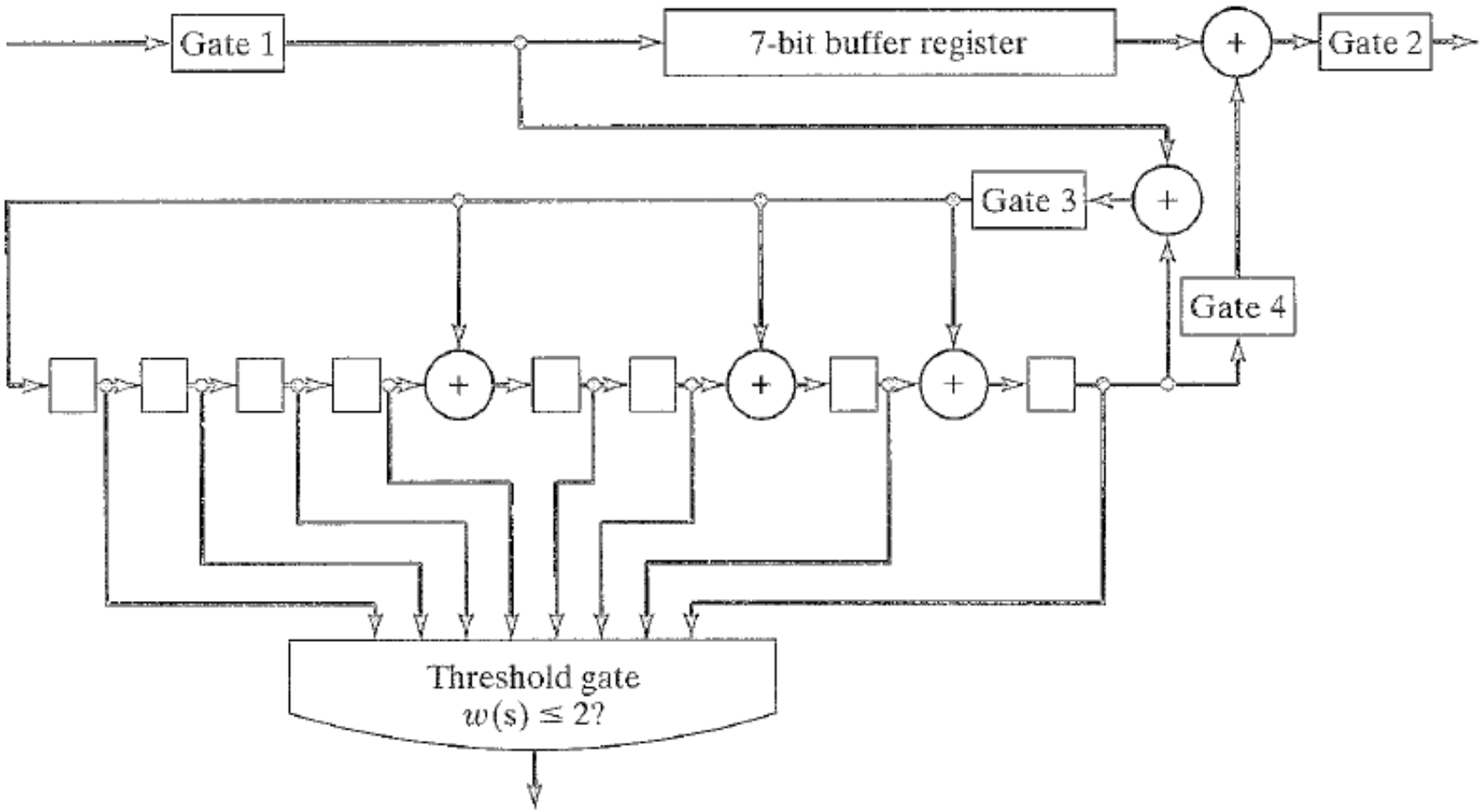


FIGURE 5.17: Error-trapping decoder for the (15, 7) cyclic code generated by $g(X) = 1 + X^4 + X^6 + X^7 + X^8$.

THE (23, 12) GOLAY CODE

As pointed out in Section 4.6, the (23, 12) Golay code [22] with a minimum distance of 7 is the only known multiple-error-correcting binary perfect code. This code can be put in cyclic form, and hence it can be encoded and decoded based on its cyclic structure. It is generated either by

$$g_1(X) = 1 + X^2 + X^4 + X^5 + X^6 + X^{10} + X^{11}$$

or by

$$g_2(X) = 1 + X + X^5 + X^6 + X^7 + X^9 + X^{11}.$$

Both $g_1(X)$ and $g_2(X)$ are factors of $X^{23} + 1$ and $X^{23} + 1 = (1 + X)g_1(X)g_2(X)$.

The encoding can be accomplished by an 11-stage shift register with feedback connections according to either $g_1(X)$ or $g_2(X)$. If the simple error-trapping scheme described in Section 5.7 is used for decoding this code, some of the double-error patterns and many of the triple-error patterns cannot be trapped. For example, consider the double-error patterns $e(X) = X^{11} + X^{22}$. The two errors are never confined to $n - k = 11$ consecutive positions, no matter how many times $e(X)$ is cyclically shifted. Therefore, they can never be trapped in the syndrome register and cannot be corrected. We can also readily see that the triple-error pattern $e(X) = X^5 + X^{11} + X^{22}$ cannot be trapped. Therefore, if the simple error-trapping scheme for decoding the Golay code is used, some of its error-correcting capability will be lost; however, the decoding circuitry is simple.

There are several practical ways to decode the (23, 12) Golay code up to its error-correcting capability $t = 3$. Two of the best are discussed in this section. Both are refined error-trapping schemes.

SHORTENED CYCLIC CODES

Given an (n, k) cyclic code C consider the set of codewords for which the l leading high-order information digits are identical to zero. There are 2^{k-l} such codewords, and they form a linear subcode of C. If the l zero information digits are deleted from each of these codewords, we obtain a set of 2^{k-l} vectors of length $n - l$. These 2^{k-l} shortened vectors form an $(n - l, k - l)$ linear code. This code is called a *shortened cyclic code* (or *polynomial code*), and it is not cyclic. A shortened cyclic code has at least the same error-correcting capability as the code from which it is derived.

Let $\mathbf{r}(X) = r_0 + r_1 X + \cdots + r_{n-l-1} X^{n-l-1}$ be the received polynomial. Suppose that $\mathbf{r}(X)$ is shifted into the syndrome register from the right end. If the decoding circuit for the original cyclic code is used for decoding the shortened code, the proper syndrome for decoding the received digit r_{n-l-1} is equal to the remainder resulting from dividing $X^{n-k+l}\mathbf{r}(X)$ by the generator polynomial $\mathbf{g}(X)$. Because shifting $\mathbf{r}(X)$ into the syndrome register from the right end is equivalent to premultiplying $\mathbf{r}(X)$ by X^{n-k}, the syndrome register must be cyclically shifted another l times after the entire $\mathbf{r}(X)$ has been shifted into the register. Now, we want to show how these extra l shifts can be eliminated by modifying the connections of the syndrome register. Dividing $X^{n-k+l}\mathbf{r}(X)$ by $\mathbf{g}(X)$, we obtain

$$X^{n-k+l}\mathbf{r}(X) = \mathbf{a}_1(X)\mathbf{g}(X) + \mathbf{s}^{(n-k+l)}(X), \tag{5.39}$$

where $\mathbf{s}^{(n-k+l)}(X)$ is the remainder and the syndrome for decoding the received digit r_{n-l-1}. Next, we divide X^{n-k+l} by $\mathbf{g}(X)$. Let $\boldsymbol{\rho}(X) = \rho_0 + \rho_1 X + \cdots + \rho_{n-k-1} X^{n-k-1}$ be the remainder resulting from this division. Then, we have the following relation:

$$\boldsymbol{\rho}(X) = X^{n-k+l} + \mathbf{a}_2(X)\mathbf{g}(X). \tag{5.40}$$

Multiplying both sides of (5.40) by $\mathbb{r}(X)$ and using the equality of (5.39), we obtain the following relation between $\rho(X)\mathbb{r}(X)$ and $\mathbb{s}^{(n-k+l)}(X)$:

$$\rho(X)\mathbb{r}(X) = [\mathbb{a}_1(X) + \mathbb{a}_2(X)\mathbb{r}(X)]\mathbb{g}(X) + \mathbb{s}^{(s-k+l)}(X). \tag{5.41}$$

The foregoing equality suggests that we can obtain the syndrome $\mathbb{s}^{(n-k+l)}(X)$ by multiplying $\mathbb{r}(X)$ by $\rho(X)$ and dividing the product $\rho(X)\mathbb{r}(X)$ by $\mathbb{g}(X)$. Computing $\mathbb{s}^{(n-k+l)}(X)$ in this way, we can avoid the extra l shifts of the syndrome register. Simultaneously multiplying $\mathbb{r}(X)$ by $\rho(X)$ and dividing $\rho(X)\mathbb{r}(X)$ by $\mathbb{g}(X)$ can be accomplished by the circuit shown in Figure 5.19. As soon as the received polynomial $\mathbb{r}(X)$ has been shifted into the register, the contents in the register form the syndrome $\mathbb{s}^{(n-k+l)}(X)$, and the first received digit is ready to be decoded.

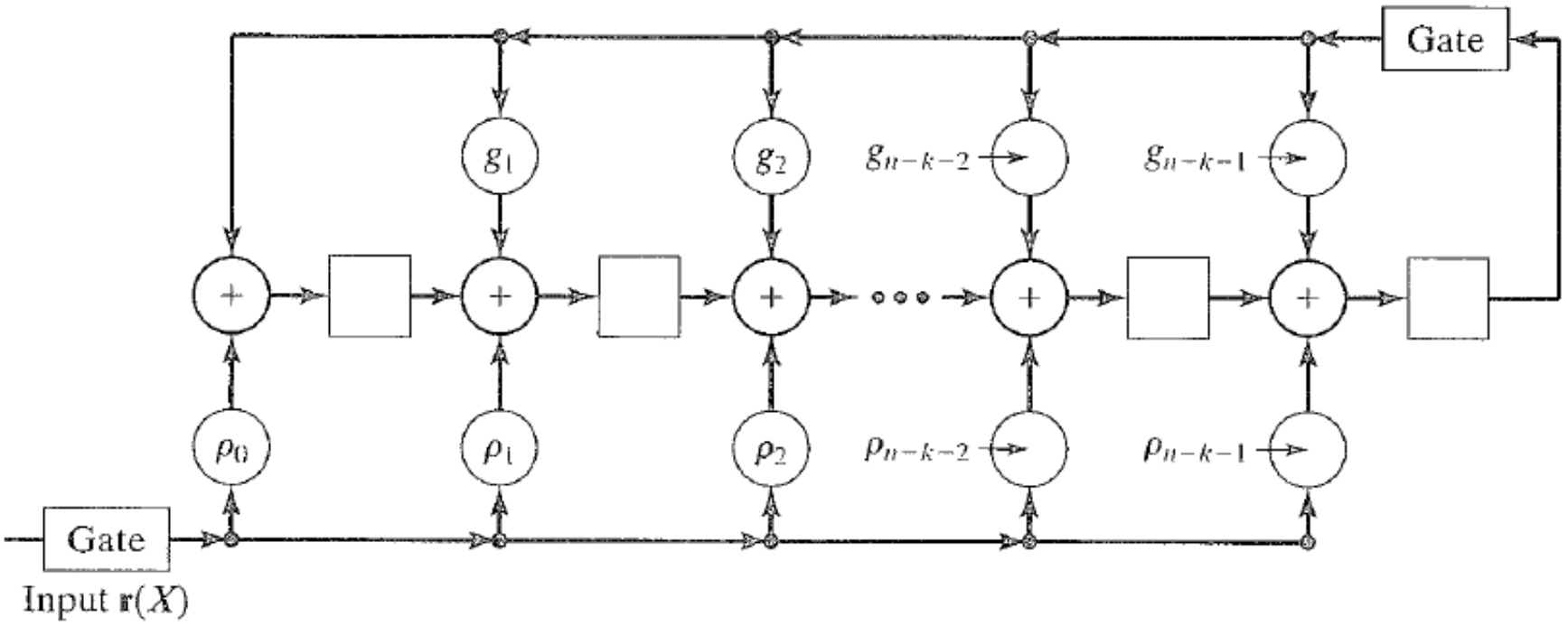


FIGURE 5.19: Circuit for multiplying $\mathbb{r}(X)$ by $\rho(X) = \rho_0 + \rho_1 X + \cdots + \rho_{n-k-1} X^{n-k-1}$ and dividing $\rho(X)\mathbb{r}(X)$ by $\mathbb{g}(X) = 1 + g_1 X + \cdots + X^{n-k}$.

EXAMPLE 5.12

For $m = 5$, there exists a (31, 26) cyclic Hamming code generated by $g(X) = 1 + X^2 + X^5$. Suppose that it is shortened by three digits. The resultant shortened code is a (28, 23) linear code. The decoding circuit for the (31, 26) cyclic code is shown in Figure 5.20.

This circuit can be used to decode the (28, 23) shortened code. To eliminate the extra shifts of the syndrome register, we need to modify the connections of the syndrome register. First, we need to determine the polynomial $\rho(X)$. Dividing X^{n-k+3} by $g(X) = 1 + X^2 + X^5$, we have

$$
\begin{array}{r|l}
X^5 + X^2 + 1 & X^3 + 1 \\
\hline
 & X^8 \\
 & X^8 + X^5 + X^3 \\
\hline
 & X^5 + X^3 \\
 & X^5 + X^2 + 1 \\
\hline
 & X^3 + X^2 + 1
\end{array}
$$

and $\rho(X) = 1 + X^2 + X^3$. The modified decoding circuit for the (28, 23) shortened code is shown in Figure 5.21.

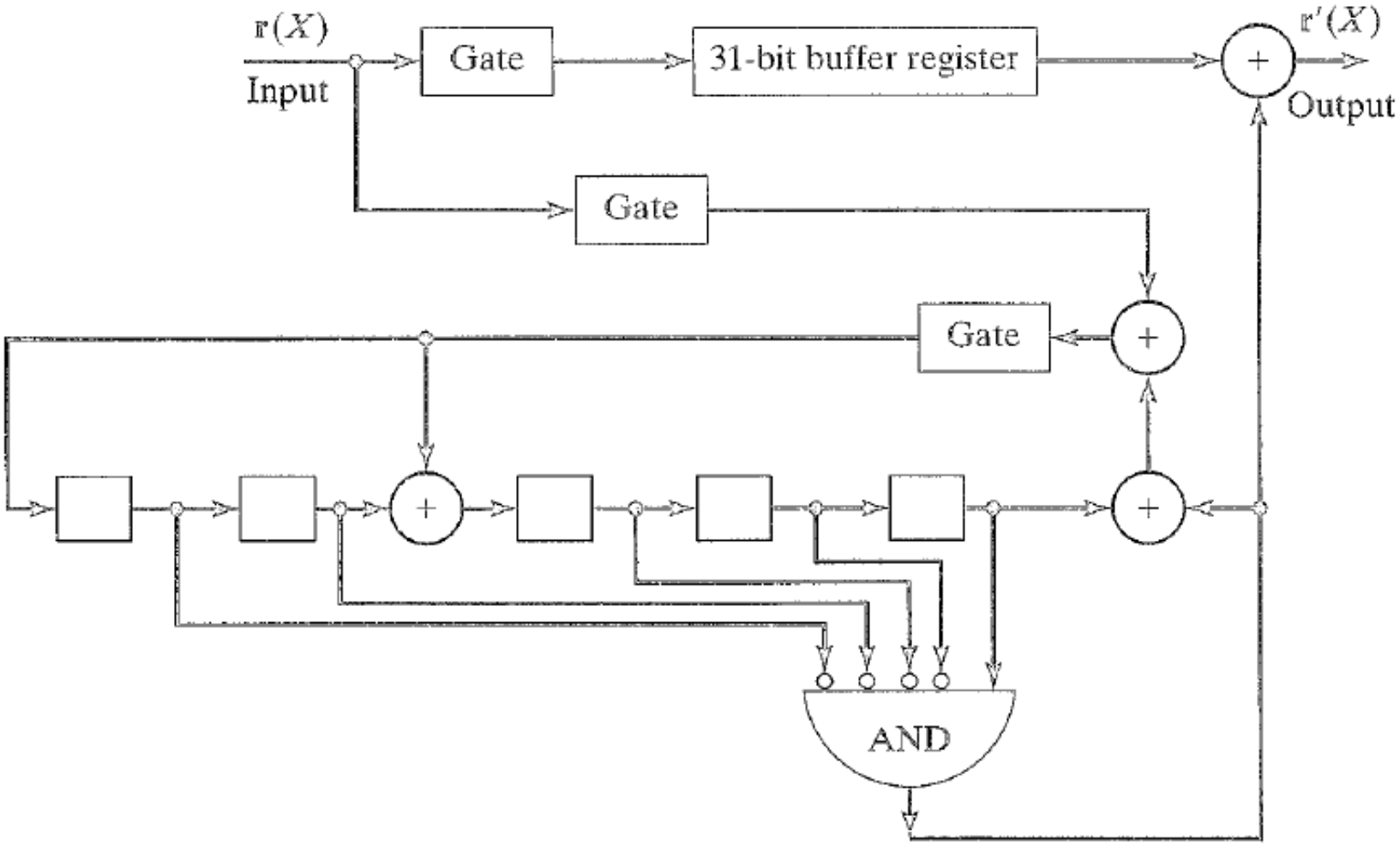


FIGURE 5.20: Decoding circuit for the (31, 26) cyclic Hamming code generated by $g(X) = 1 + X^2 + X^5$.

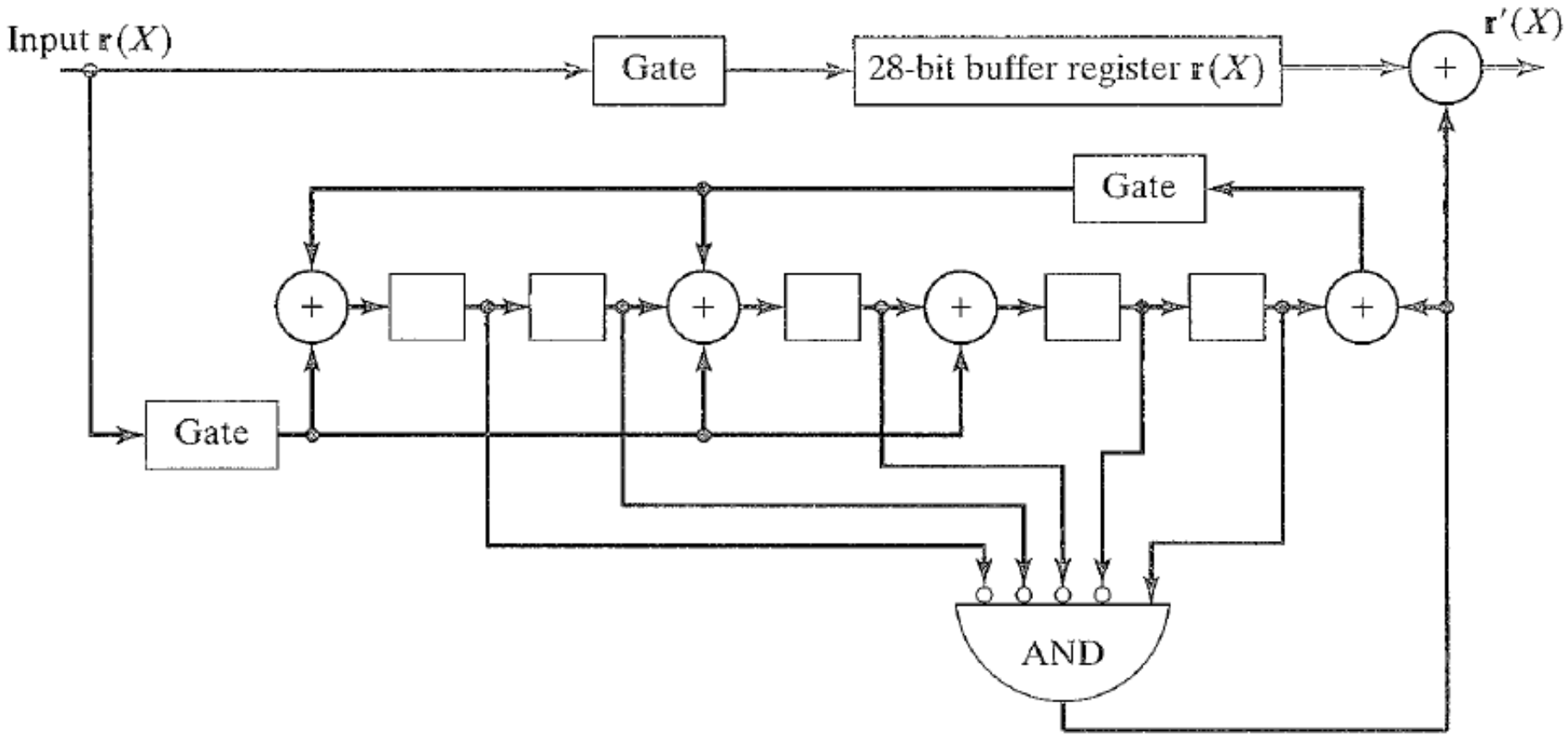


FIGURE 5.21: Decoding circuit for the (28, 23) shortened cyclic code generated by $\mathbf{g}(X) = 1 + X^2 + X^5$.

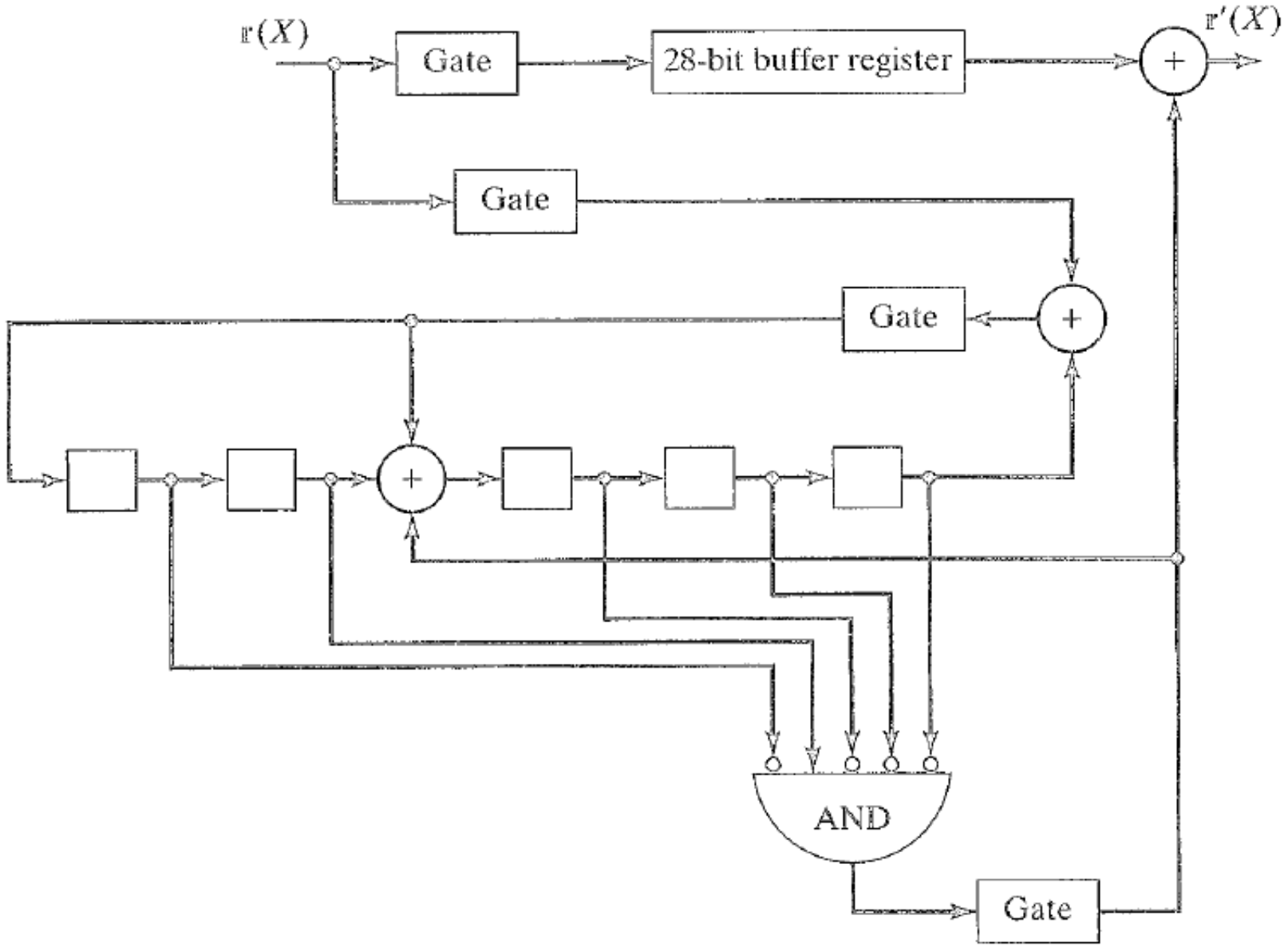


FIGURE 5.22: Another decoding circuit for the (28, 23) shortened Hamming code generated by $g(X) = 1 + X^2 + X^5$.

Shortened cyclic codes for error detection in conjunction with ARQ protocols are widely used for error control, particularly in computer communications. In these applications they are often called *cyclic redundancy check* (CRC) codes. A CRC code is, in general, generated by either a primitive polynomial $\mathfrak{p}(X)$ or a polynomial $\mathfrak{g}(X) = (X+1)\mathfrak{p}(X)$. A number of CRC codes have become international standards for error detection in various contexts. A few standard CRC codes follow:

CCITT X-25 (Consultative Committee for International Telegraphy and Telephony, Recommendation X-25)

$$\begin{aligned}\mathfrak{g}(X) &= (X+1)(X^{15}+X^{14}+X^{13}+X^{12}+X^4+X^3+X^2+X+1)\\ &= X^{16}+X^{12}+X^5+1,\end{aligned}$$

ANSI (American National Standards Institute)

$$\mathfrak{g}(X) = (X+1)(X^{15}+X+1) = X^{16}+X^{15}+X^2+1,$$

IBM-SDLC (IBM Synchronous Data Link Control)

$$\begin{aligned}\mathfrak{g}(X) &= (X+1)^2(X^{14}+X^{13}+X^{12}+X^{10}+X^8+X^6+X^5+X^4+X^3+X+1)\\ &= X^{16}+X^{15}+X^{13}+X^7+X^4+X^2+X+1,\end{aligned}$$

IEC TC57

$$\begin{aligned}\mathfrak{g}(X) &= (X+1)^2(X^{14}+X^{10}+X^9+X^8+X^5+X^3+X^2+X+1)\\ &= X^{16}+X^{14}+X^{11}+X^8+X^6+X^5+X^4+1,\end{aligned}$$

IEEE Standard 802.3

$$\begin{aligned}\mathfrak{g}(X) = {} & X^{32}+X^{26}+X^{23}+X^{22}+X^{16}+X^{12}+X^{11}\\ & +X^{10}+X^8+X^7+X^5+X^4+X^2+X+1.\end{aligned}$$

CYCLIC PRODUCT CODES

If the component codes C_1 and C_2 are cyclic, and if their lengths, n_1 and n_2, are relatively prime, the product code $C_1 \times C_2$ is cyclic if the code digits of a code array are transmitted in a proper order [3, 25, 26]. We start with the upper right corner and move down and to *the left* on a 45° diagonal, as shown in Figure 5.23. When we reach the end of a column, we move to the top of the next column. When we reach the end of a row, we move to the rightmost digit of the next row.

Because n_1 and n_2 are relatively prime, there exists a pair of integers a and b such that

$$an_1 + bn_2 = 1.$$

Let $\mathbf{g}_1(X)$ and $\mathbf{h}_1(X)$ be the generator and parity polynomials of the (n_1, k_1) cyclic code C_1, and let $\mathbf{g}_2(X)$ and $\mathbf{h}_2(X)$ be the generator and parity polynomials of the (n_2, k_2) cyclic code C_2. Then, it is possible to show [25, 26] that the generator polynomial $\mathbf{g}(X)$ of the cyclic product code of C_1 and C_2 is the greatest common divisor (GCD) of $X^{n_1 n_2} - 1$ and $\mathbf{g}_1(X^{bn_2})\mathbf{g}_2(X^{an_1})$; that is,

$$\mathbf{g}(X) = \text{GCD}[X^{n_1 n_2} - 1, \mathbf{g}_1(X^{bn_2})\mathbf{g}_2(X^{an_1})], \tag{5.42}$$

and the parity polynomial $\mathbf{h}(X)$ of the cyclic product code is the greatest common divisor of $\mathbf{h}_1(X^{bn_2})$ and $\mathbf{h}_2(X^{an_1})$; that is,

$$\mathbf{h}(X) = \text{GCD}[\mathbf{h}_1(X^{bn_2}), \mathbf{h}_2(X^{an_1})]. \tag{5.43}$$

The complexity of the decoder for cyclic product codes is comparable to the complexity of the decoders for both the (n_1, k_1) code and the (n_2, k_2) code. At the receiving end of the channel, the received vector may again be rearranged as a rectangular array. Thus, the decoder can decode each of the row (or column) codewords separately and then decode each of the column (or row) codewords.

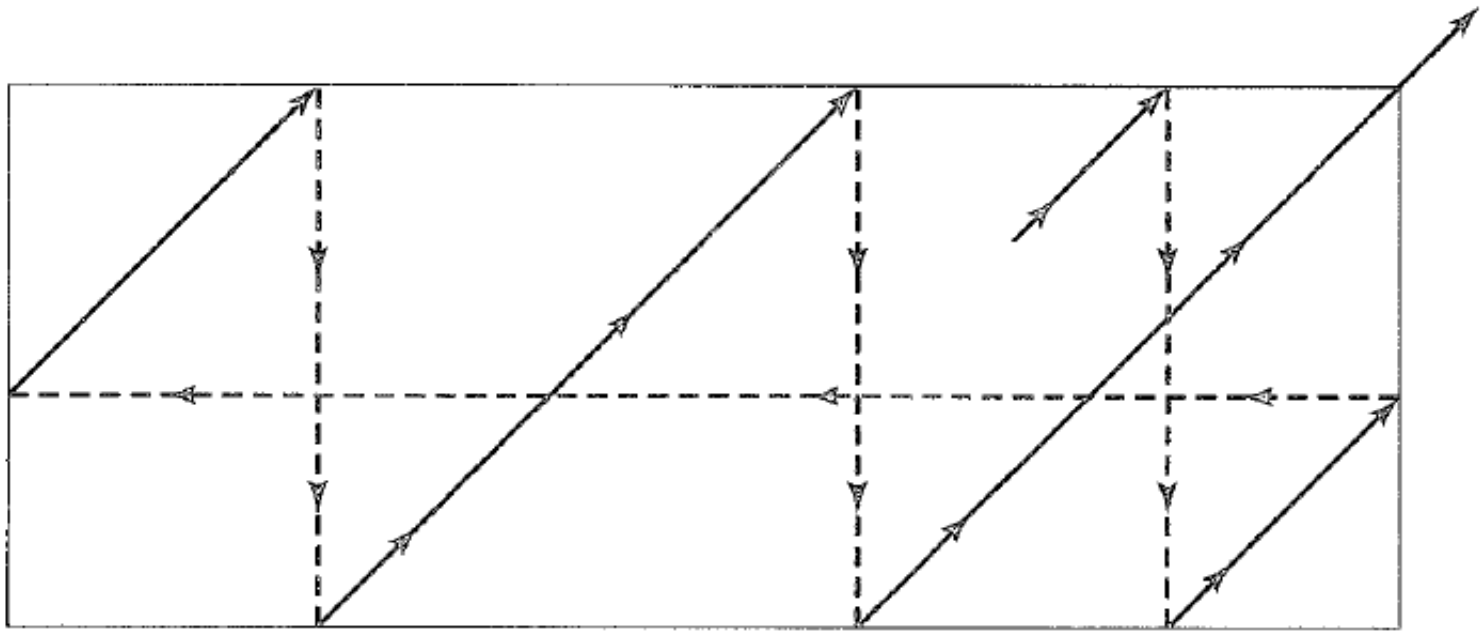


FIGURE 5.23: Transmission of a cyclic product code.

QUASI-CYCLIC CODES

A *quasi-cyclic code* is a linear code for which cyclically shifting a codeword a fixed number $n_0 \neq 1$ (or a multiple of n_0) of symbol positions either to the right or to the left results in another codeword. It is clear that for $n_0 = 1$, a quasi-cyclic code is a cyclic code. The integer n_0 is called the *shifting constraint.* It is clear that the dual of a quasi-cyclic code is also quasi-cyclic.

As an example, consider the (9, 3) code generated by the following generator matrix:

$$\mathbb{G} = \begin{bmatrix} 111 & 100 & 110 \\ 110 & 111 & 100 \\ 100 & 110 & 111 \end{bmatrix}.$$

TABLE 5.6: The codewords of the (9, 3) quasi-cyclic code.

000	000	000
111	100	110
110	111	100
100	110	111
001	011	010
011	010	001
010	001	011
101	101	101

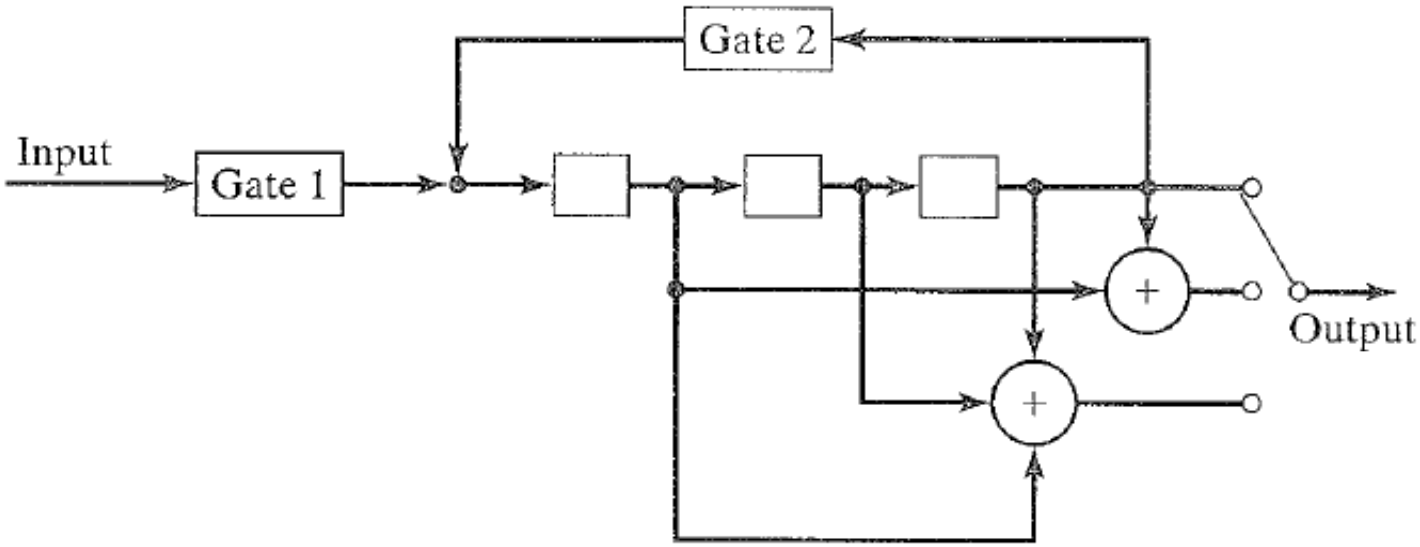


FIGURE 5.24: An encoding circuit for a (9, 3) quasi-cyclic code.

three information symbols have been shifted into the register, gate 1 is deactivated and gate 2 is activated. The information symbol c_2 and two parity-check symbols $p_2^{(1)}$ and $p_2^{(2)}$ appear at the output terminals and then are shifted into the channel. The two parity-check symbols are given by

$$p_2^{(1)} = c_0 + c_2,$$

$$p_2^{(2)} = c_0 + c_1 + c_2.$$

Next, the register is shifted once. The content of the register is now (c_2, c_0, c_1). The information symbol c_1 and two parity-check symbols $p_1^{(1)}$ and $p_1^{(2)}$ appear at the output terminals, and they are shifted into the channel. The parity-check symbols $p_1^{(1)}$ and $p_1^{(2)}$ are given by

$$p_1^{(1)} = c_1 + c_2,$$

$$p_1^{(2)} = c_0 + c_1 + c_2.$$

At this point, the register is shifted once again. The content of the register is now (c_1, c_2, c_0), and the information symbol c_0 and two parity-check symbols $p_0^{(1)}$ and $p_0^{(2)}$ appear at the output terminals.

These three symbols are then shifted into the channel. The two parity-check symbols are given by

$$p_0^{(1)} = c_0 + c_1,$$

$$p_0^{(2)} = c_0 + c_1 + c_2.$$

This completes the encoding. The codeword has the form

$$\mathbf{v} = (p_0^{(2)}, p_0^{(1)}, c_0, p_1^{(2)}, p_1^{(1)}, c_1, p_2^{(2)}, p_2^{(1)}, c_2),$$

which consists of three blocks, each of which consists of one unaltered information symbol and two parity-check symbols. This form may also be regarded as a systematic form.

For an (mn_0, mk_0) quasi-cyclic code with shifting constraint n_0, the generator matrix in systematic form is

$$\mathbb{G} = \begin{bmatrix} \mathbb{P}_0\mathbb{I} & \mathbb{P}_1\mathbb{0} & \cdots & \mathbb{P}_{m-1}\mathbb{0} \\ \mathbb{P}_{m-1}\mathbb{0} & \mathbb{P}_0\mathbb{I} & \cdots & \mathbb{P}_{m-2}\mathbb{0} \\ \vdots & \vdots & & \vdots \\ \mathbb{P}_1\mathbb{0} & \mathbb{P}_2\mathbb{0} & \cdots & \mathbb{P}_0\mathbb{I} \end{bmatrix}, \tag{5.44}$$

where $\mathbb{I}$ and $\mathbb{0}$ represent the $k_0 \times k_0$ identity and zero matrices, respectively, and $\mathbb{P}_i$ is an arbitrary $k_0 \times (n_0 - k_0)$ matrix. Each row (as a sequence of m $k_0 \times n_0$ matrices) is the cyclic shift (to the right) of the row immediately above it, and the row at the top is the cyclic shift of the bottom row. Each column of $\mathbb{G}$ is the downward cyclic shift of the column on its left (or the upward cyclic shift of the column on its right). A message for the code consists of m k_0-bit blocks, and a codeword consists of m n_0-bit blocks. Each of these m n_0-bit blocks consists of k_0 unaltered information symbols and $n_0 - k_0$ parity-check symbols. The parity-check matrix corresponding to the generator matrix given by (5.44) is

$$\mathbb{H} = \begin{bmatrix} \mathbb{I}\mathbb{P}_0^T & \mathbb{O}\mathbb{P}_{m-1}^T & \cdots & \mathbb{O}\mathbb{P}_1^T \\ \mathbb{O}\mathbb{P}_1^T & \mathbb{I}\mathbb{P}_0^T & \cdots & \mathbb{O}\mathbb{P}_2^T \\ \vdots & \vdots & & \vdots \\ \mathbb{O}\mathbb{P}_{m-1}^T & \mathbb{O}\mathbb{P}_{m-2}^T & \cdots & \mathbb{I}\mathbb{P}_0^T \end{bmatrix},$$

where $\mathbb{I}$ and $\mathbb{O}$ represent the $(n_0 - k_0) \times (n_0 - k_0)$ identity and zero matrices, respectively, and $\mathbb{P}_i^T$ is the transpose of $\mathbb{P}_i$. Consider the (9, 3) quasi-cyclic code given previously for which $k_0 = 1$ and $n_0 = 3$. The parity-check matrix in systematic form is

$$\mathbb{H} = \begin{bmatrix} 101 & 001 & 001 \\ 011 & 001 & 000 \\ 001 & 101 & 001 \\ 000 & 011 & 001 \\ 001 & 001 & 101 \\ 001 & 000 & 011 \end{bmatrix}.$$

A more general form for the generator matrix of an (mn_0, mk_0) quasi-cyclic code is

$$\mathbb{G} = \begin{bmatrix} \mathbb{G}_0 & \mathbb{G}_1 & \cdots & \mathbb{G}_{m-1} \\ \mathbb{G}_{m-1} & \mathbb{G}_0 & \cdots & \mathbb{G}_{m-2} \\ \vdots & \vdots & & \vdots \\ \mathbb{G}_2 & \mathbb{G}_3 & \cdots & \mathbb{G}_1 \\ \mathbb{G}_1 & \mathbb{G}_2 & \cdots & \mathbb{G}_0 \end{bmatrix}, \tag{5.46}$$

where each $\mathbb{G}_i$ is a $k_0 \times n_0$ submatrix. We see that $\mathbb{G}$ given by (5.46) displays the cyclic structure among the rows and columns in terms of the submatrices $\mathbb{G}_i$'s. For $0 \leq j < m$, let $\mathbb{M}_j \stackrel{\Delta}{=} [\mathbb{G}_j, \mathbb{G}_{j-1}, \cdots, \mathbb{G}_{j+1}]^T$ denote the jth column of $\mathbb{G}$ (with $\mathbb{G}_m = \mathbb{G}_0$). $\mathbb{M}_j$ is an $mk_0 \times n_0$ submatrix. Then, we can put $\mathbb{G}$ in the following form:

$$\mathbb{G} = [\mathbb{M}_0, \mathbb{M}_1, \cdots, \mathbb{M}_{m-1}].$$

For $0 \leq l < n_0$, let $\mathbb{Q}_l$ be the $mk_0 \times m$ submatrix that is formed by taking the lth columns from $\mathbb{M}_0, \mathbb{M}_1, \cdots, \mathbb{M}_{m-1}$. Then, we can put $\mathbb{G}$ in the following form:

$$\mathbb{G}_c = [\mathbb{Q}_0, \mathbb{Q}_1, \cdots, \mathbb{Q}_{n_0-1}].$$

Each column of $\mathbb{Q}_l$ consists of mk_0 bits that are regarded as m k_0-bit bytes (a *byte* is a group of k_0 binary digits). In terms of bytes, $\mathbb{Q}_l$ is regarded as an $m \times m$ matrix that has the following cyclic structure: (1) each row is the cyclic shift (to the right) of the row immediately above it, and the top row is the cyclic shift of the bottom row; (2) each column is the downward cyclic shift of the column on its left, and the leftmost column is the downward cyclic shift of the rightmost column. The matrix $\mathbb{Q}_l$ is called a *circulant*. Therefore, $\mathbb{G}_c$ consists of n_0 circulants. Most often, quasi-cyclic codes are studied in circulant form.

EXAMPLE 5.14

Consider the (15, 5) quasi-cyclic code with parameters $m = 5$, $n_0 = 3$, and $k_0 = 1$ that is generated by the following generator matrix:

$$\mathbf{G} = \begin{bmatrix} 001 & 100 & 010 & 110 & 110 \\ 110 & 001 & 100 & 010 & 110 \\ 110 & 110 & 001 & 100 & 010 \\ 010 & 110 & 110 & 001 & 100 \\ 100 & 010 & 110 & 110 & 001 \end{bmatrix}.$$
$$\mathbf{M}_0 \quad \mathbf{M}_1 \quad \mathbf{M}_2 \quad \mathbf{M}_3 \quad \mathbf{M}_4$$

This quasi-cyclic code has a minimum distance of 7. In circulant form, the generator matrix takes the following form:

$$\mathbf{G} = \begin{bmatrix} 01011 & 00111 & 10000 \\ 10101 & 10011 & 01000 \\ 11010 & 11001 & 00100 \\ 01101 & 11100 & 00010 \\ 10110 & 01110 & 00001 \end{bmatrix}.$$
$$\mathbf{Q}_0 \quad \mathbf{Q}_1 \quad \mathbf{Q}_2$$