≃ 60min V39

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:

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

$GF(p)$, $GF(2)=\{0,1\}$

عدد اول

Polynomials:

$(GF(p),+,\times)$

در پیمانه‌ی p

$f(x)=f_n x^n+f_{n-1}x^{n-1}+\dots+f_0 \quad f_n\neq 0$

$\deg(f(x))=n$

$f_i\in GF(p)$ or $GF(q)$ ($q=2^m$)

$$\begin{cases} f(x)=f_n x^n+\dots+f_1x+f_0 \\ g(x)=g_m x^m+\dots+g_1x+g_0 \quad n>m \end{cases} \Rightarrow f(x)+g(x)=f_n x^n+f_{n-1}x^{n-1}+\dots+(f_m+g_m)x^m+\dots+(f_1+g_1)x+(f_0+g_0)$$

$g_i\in GF(p)$

جمع در $GF(p)$

مثال
$$\begin{cases} f(x)=x^2+x+1 \\ g(x)=x^2+x \end{cases} \longrightarrow f(x)+g(x)=(\cancel{1+1})x^2+(\cancel{1+1})x+1=1$$

$$f(x)\cdot g(x)=\left(\sum_{i=0}^{n} f_i x^i\right)\cdot\left(\sum_{j=0}^{m} g_j x^j\right)=f_0\cdot g_0+(f_0g_1+f_1g_0)x+\dots+f_ng_mx^{n+m}$$

$$\longrightarrow f(x)\cdot g(x)=(x^2+x+1)\cdot(x^2+x)=x^4+x$$

V42

$f(x)-f(x)=0$ ~~$f(x)=f(x)$~~

- کلیه‌ی چندجمله‌ای‌ها روی میدان $GF(p)$ با عمل جمع چندجمله‌ای‌ها تشکیل یک گروه آبلی می‌دهند.
- در عمل ضرب چندجمله‌ای‌ها:
  - عضو خنثی عمل ضرب چندجمله‌ای $f(x)=1$ است
  - لذا برای چندجمله‌ای‌ها عضو معکوس ضربی وجود ندارد
  - شرط پذیری و جابه‌جایی پذیری عمل ضرب نیز به سادگی قابل بررسی است.

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

تحویل ناپذیر

$Z=\{0,\pm1,\pm2,\dots\} \Rightarrow Z_p=\{0,1,\dots,p-1\}$ عدد اول $\quad (Z_p,+,\times)$ پیمانه‌ی

$\{0,1,\dots,p-1\}$

$a=pq+r \Rightarrow a\equiv r$

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

$a(x)=f(x)q(x)+r(x) \Rightarrow \deg(r(x))<\deg(f(x))$

$f(x)=f_0 \xrightarrow{f_0\neq 0}$ درجه صفر

$f(x)=0 \to -\infty$ درجه

$n=p_1^{e_1}\times p_2^{e_2}\times p_3^{e_3}\times\dots\times p_n^{e_n}$

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

B2

اگر $f(x)$ بر هیچ چندجمله‌ای درجه‌ی کوچکتر از آن بخش‌پذیر نباشد، آن را یک چندجمله‌ای تحویل ناپذیر (تجزیه نشدنی) گوییم.

(irreducible)

در حالت کلی هر چندجمله‌ای دلخواه را می‌توان به صورت یکتا به چندجمله‌ای‌های تحویل‌ناپذیر تجزیه کرد

(بشرطی که چندجمله‌ای‌های تحویل‌ناپذیر monic باشند)

$f(x)=af_1^{e_1}(x)f_2^{e_2}(x)\dots f_n^{e_n}(x)$

$a\in GF(q)$

تعریف: چندجمله‌ای monic $f(x)=f_0+f_1x+\dots+f_nx^n$ را گوییم اگر $f_n=1$ باشد.

theorem 2.10

$f(x)=f_0+f_1x+\dots+f_nx^m$

$f_0\neq 0 \quad f_m\neq 0$

$f(x)\mid x^{2^m-1}+1$

$\Rightarrow x^{2^m-1}+1=f(x)y(x)$

$f(x)\mid x^e+1$ (*)

(*) کوچکترین مقدار ممکن در رابطه‌ی

B1 ادامه

$e<2^m-1$

e را نمای چندجمله‌ای $f(x)$ می‌نامیم، اگر $e=2^m-1$

Polynomial ادامه

فصل 2

$$f(x) = f_0 + f_1 x + \dots + f_n x^n$$

$$f_i \in GF(p) \text{ or } GF(q)$$

p^m

B1

اگر فرض کنید کلیه جمله‌های چندجمله‌ای دلخواه روی میدان $GF(p)$ را به صورت زیر در نظر بگیریم.

$$a(x) = a_0 + a_1 x + \dots + a_n x^n \qquad a_i \in GF(p)$$

اگر $f(x)$ یک چندجمله‌ای تحویل‌ناپذیر از درجه‌ی m باشد

$$\Rightarrow a(x) = q(x) f(x) + r(x) \quad , \deg(r(x)) \leq m-1$$

$$r(x) = r_0 + r_1 x + \dots + r_{m-1} x^{m-1} \quad , r_i \in GF(p)$$

مجموع چندجمله‌ای‌های $r(x)$ حداکثر به تعداد p^m می‌باشد.

* درصورت چندجمله‌ای‌ها اگر دو چندجمله‌ای غیرصفر در هم ضرب شوند، حاصل یک چندجمله‌ای خواهد شد.

اگر ضرب دو چندجمله‌ای صفر شد، حداقل یکی از آنها برابر صفر بوده است.

مجموعه تمام چندجمله‌ای‌ها در میدان $GF(p)$

$$\frac{R[x]}{(f(x))} = \{r_0 + r_1 x + \dots + r_{m-1} x^{m-1} / r_i \in GF(p)\}$$

Z_p

چندجمله‌ای تحویل‌ناپذیر باشد

$$Z_p = \{0, 1, \dots, p-1\}$$

$$(GF(p^m), +, \times)$$

$f(x)$ درپیمانه

اگر $\gcd(a, n) = 1$ باشد، آنگاه معادله‌ی فوق جواب یکتا دارد که همان a^{-1} می‌باشد $\Rightarrow$

$$a \cdot x = 1 \mod n$$

$$a \in Z_n = \{0, 1, \dots, n-1\}$$

اگر $n = p$ باشد (عدد اول) تمام اعداد $a \in \{1, 2, \dots, p-1\}$ معکوس وجود دارد

درمجموع بیان شد، تمام $r(x)$ نسبت به $f(x)$ اول می‌باشد.

$$\gcd(f(x), r(x)) = 1$$

$\boxed{B2}$

صفر a را ریشه‌ی چند جمله‌ای $f(x)$ گوییم اگر $(x-a) \mid f(x) \Rightarrow f(x)=(x-a)g(x)$.

واضح است که $f(a)=0$.

مثال) $f(x)=x^4+x^3+x^2+1$

$f(1)=1+1+1+1=0$, $f(x)=(x+1)\underbrace{(q(x))}_{x^3+x+1}$

$$\begin{array}{l|l} x^4+x^3+x^2+1 & x+1 \\ \hline x^4+x^3 & x^3+x+1 \\ \hline \quad 1 & \\ \hline \quad 0 & \end{array}$$

درجه چندجمله‌ای $m \to$

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 $1 \le n < 15$. 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.

$(x^5+1) = (x+1)(x^4+x^3+x^2+x+1)$

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	$1+X+X^2+X^7+X^{24}$

آخر برنده

$f(x) \to x^m f\left(\frac{1}{x^m}\right)$

$1+x+x^3 \to x^3\left(1+\frac{1}{x}+\frac{1}{x^3}\right) = x^3+x^2+1$

$f(x) \mid x^e+1 \Rightarrow x^e+1 = f(x)q(x) \Rightarrow 1 = f(x)q(x)+x^e \Rightarrow 1 \mid f(x)$, x^e

مثال: $1 \mid 1+x+x^2+x^3+x^4$, $1+x$

$2^4-1=15$

$1 \mid 1+x+x^4$, $1+x+x^2$

$x+x^4$

$x+x^2+x^5$

$x^2+x^4+x^5$

$x^2+x^3+x^6$

$\vdots$

Comments:

$x+x^2+x^3+x^4$

$x+x^2+x^3+x^4+x^5$

$\boxed{x^5} \to x^e$

x^{15}

اگر n به صورت $n = p_1^{e_1} \times p_2^{e_2} \times \cdots p_k^{e_k}$ تجزیه شود

$$\phi(n) = n \times \left(1-\frac{1}{p_1}\right)\left(1-\frac{1}{p_2}\right)\cdots\left(1-\frac{1}{p_k}\right) = n\prod_{i=1}^{k}\left(1-\frac{1}{p_i}\right)$$

چندجمله‌ای‌های Primitive را از درجه‌ی m به تعداد $\lambda(m)$ وجود دارد، که $\lambda(m)$ برابر است با:

$$\lambda(m) = \frac{\phi(2^m-1)}{m} \qquad \lambda(m) = \frac{\phi(p^m-1)}{m}$$

در عبارت فوق تابع $\phi(\cdot)$، همان تابع اویلر می‌باشد. (1:17min)

($\phi(n)$: برابر است با تعداد اعداد مجموعه $\{1,2,\dots,n\}$ که نسبت به n اول باشند.)

مثال) $\phi(10)=4$

$\{1,2,3,4,5,6,7,8,9,10\}$

$$\lambda(3) = \frac{\phi(2^3-1)}{3} = \frac{\phi(7)}{3} = \frac{6}{3} = 2$$

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 "·" 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

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

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:

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

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 $0 \le i,\ j < 2^m - 1$. If $i + j < 2^m - 1$, then $\alpha^i \cdot \alpha^j = \alpha^{i+j}$, which is obviously a nonzero element in F^*. If $i + j \ge 2^m - 1$, we can express $i + j$ as follows: $i + j = (2^m - 1) + r$, where $0 \le r < 2^m - 1$. 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).

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 $\alpha^7, \alpha^{11}, \alpha^{13}$, and α^{14} are the roots of X^4+X^3+1. We may verify this result as follows:

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

v49

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

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

Substituting β into the preceding equation, we obtain

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

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

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

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

add +1

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

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 $X^{2^m} + X$.

اگر ۰ را هم شامل شد می‌توانیم در یک ضرب کنیم.

اگر β یک عضو میدان باشد ، $\varphi(X)$ را چندجمله‌ای مینیمال گوییم به شرطی که ریشه‌ی این چندجمله‌ای β باشد با حداقل درجه ممکن

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.

چندجمله‌ای مینیمال هر عنصری قطعاً تحویل ناپذیر است.

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 $\phi(X)$.

$\varphi(x) \mid f(x)$

فرض می‌کنیم $f(x) = \varphi(x)a(x) + r(x)$

$\xrightarrow{\beta}$ $0 = 0 \cdot a(\beta) + r(\beta) \longrightarrow r(\beta) = 0$

$r(X) \neq 0 \qquad \deg(r) < \deg(\varphi(x))$

$\Rightarrow \varphi(x) \mid f(x)$

اثبات → $\beta \in GF(2^m)$ $\quad X^{2^m}+X$ $\quad \phi(x) \mid f(x)$ اگر $f(x)$ داشته باشیم که β ریشه‌ی آن باشد $\phi(x)$ را $f(x)$ عادی کند.

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.**

$f(\beta)=0 \Rightarrow \varphi(x) \mid f(x)$ → طبق قضیه‌ی قبل

چند جمله‌ای مینیمال چگونه بدست می‌آید

$\varphi(\beta)=0,\ \varphi(\beta^{2^i})=0 \rightarrow (\beta,\beta^2,\beta^{2^2},\dots,\beta^{2^{e-1}}),\ \beta^{2^e}=1 \Rightarrow \beta^{2^m}=\beta \Rightarrow e\le m$

مزدوج‌های β هم ریشه φ هستند

$e \mid m \Rightarrow m=qe$

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 $\beta^2,\beta^{2^2},\cdots,\beta^{2^i},\cdots$ 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 $\beta^{2^m}=\beta$, $e\le m$ (in fact e divides m).

دلیل

میدان 2^m-1 (بدون صفر عضو دارد)

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,

①

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

$f(\beta)=0$ می‌شود

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

درجه $f(x)$ برابر e

$\Rightarrow f(x)$

می‌خواهیم $GF(2)$ را ثابت کنیم.

$f(x)=f_0+f_1x+\dots+f_ex^e \qquad f_i\in\{0,1\}$

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

$i'=i+1$ $\qquad \beta$

$$\boxed{(x+\beta^{2^i})^2=x^2+(\beta^{2^i})^2=x^2+\beta^{2^{i+1}}}$$

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

$\Rightarrow (f(x))^2=f(x^2) \qquad f_i^2=f_i \ \text{if}\ f_i\in\{0,1\}$

$\Rightarrow f(x)=f_0+f_1x+\dots+f_ex^e$

② اگر $f(x)$ تحویل‌پذیر باشد

22:18 min

$f(x)=a(x)\cdot b(x)$

می‌دانیم $f(\beta)=0$ است، پس:

$a(\beta)\cdot b(\beta)=0$

اگر فرض کنیم $a(\beta)$ برابر صفر باشد، بنابراین β و مزدوج‌های خود ریشه‌های $a(x)$ خواهد بود.

$\beta,\beta^2,\beta^{2^2},\dots,\beta^{2^{e-1}} \quad (\beta^{2^e}=\beta)$

بنابراین درجه‌ی $a(x)$ برابر e خواهد شد چون درجه $f(x)$ نیز برابر e است

پس $f(x)$ تحویل‌ناپذیر است.

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:

(تعدیل نلنید $a_0 = 1$)

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

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:

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

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, $e \le m$.

ثابت

$e \le m \Rightarrow e | m \Rightarrow \begin{cases} \beta^{2^m} = \beta \\ \beta^{2^e} = \beta \end{cases} \Rightarrow \beta^j = \beta^i \Rightarrow j = i \bmod 2^e - 1$

$\beta^{j-i} = 1 \Rightarrow \beta^{j-i+1} = \beta$

THEOREM 2.20 If $\cancel{\beta}$ α 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

$\alpha \quad \alpha^{2^l}$

$\{\alpha^0=1, \alpha, \alpha^2, \dots, \alpha^{2^m-2}\}$

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

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

$a|b, b|a \Rightarrow a=b$

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

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

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

$\boxed{n | 2^m - 1}$ $\quad 2^m-1 | n \times 2^l \Rightarrow 2^m-1 | n$

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.

اگر Z_n^* گروه‌ی است و اگر α مولد، این گروه Z_n^* ، $n=2,4,p^n,2p^n$ برابر n ، $\{Z_n^* = \{1,2,\dots,n-1\}\} | Z_n^*$

عدد اول باشد

فرض باشد آنگاه $\beta=\alpha^i$ نیز مولد گروه است، اگر و تنها اگر $\gcd(i,\varphi(n))=1$

باشد.

$Z_n^* = \{i \in Z_n \mid \gcd(i,n)=1\}$

$(Z_n^*, \times)$

$Z^* = \{\alpha^0=1, \alpha, \alpha^2, \dots, \alpha^{\varphi(n)-1}\}$

8

38min

$\varphi(2^3-1)=8$

$\boxed{\alpha^{\varphi(n)}=1}$

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 $n = 5$.

$(X+\alpha^5)(X+\alpha^{10})$

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):

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

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

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

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

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

$f(\beta)=0 \Rightarrow f(\beta^{2^i})=0$ وقتی است که f در GF(2) باشد. ←

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

$f_i \in GF(p)$
$\Downarrow$
$f(\beta)=0 \Rightarrow f(\beta^{p^i})=0$

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

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	$1 + X^3 + X^5 + X^6$ $= (1 + X + X^2 + X^3) \cdot g(X)$

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

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

V52

$V(x) = v_0 + v_1 x + \cdots + v_{n-1} x^{n-1}$

$(v_0, v_1, \ldots, v_{n-1}) \xrightarrow{\text{shift}} (v_{n-1}, v_0, v_1, \ldots, v_{n-2}) \xrightarrow{i\text{-shift}}$

$(v_{n-i}, v_{n-i+1}, \ldots, v_{n-i-1})$

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

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

The preceding equation can be manipulated into the following form:

$$X^i\mathbf{v}(X) = \boxed{v_{n-i} + v_{n-i+1}X + \cdots + v_{n-1}X^{i-1} + v_0X^i + \cdots + v_{n-i-1}X^{n-1}} \quad V^{(i)}(x) \quad V^{(i)}(x) + (x^n+1)q(x)$$

$$+ v_{n-i}(X^n + 1) + v_{n-i+1}X(X^n + 1) + \cdots + v_{n-1}X^{i-1}(X^n + 1) \qquad (5.1)$$

$$= \mathbf{q}(X)(X^n + 1) + \mathbf{v}^{(i)}(X),$$

$\boxed{V^{(i)}(x) = x^i V(x) \bmod x^n + 1}$

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

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

فرض کنید $g(x) = g_0 + g_1 x + \cdots + g_{r-1} x^{r-1} + x^r$ ، حداقل درجه را داشته باشد.

$g(x) \in C(n,k)$

فرض کنید این کلمه‌ی کد یکتا نباشد

$$g'(x) = g'_0 + g'_1 x + \cdots + g'_{r-1} x^{r-1} + x^r$$

$$g(x) + g'(x) \in C(n,k) \Rightarrow g'(x) + g(x) = (g'_0 + g_0) + \cdots + (g'_{r-1} + g_{r-1}) x^{r-1} + 0$$

خلاف فرض است بنابراین $g(x)$ یکتاست.

THEOREM 5.2 Let $g(X) = \overset{1}{g_0} + g_1 X + \cdots + g_{r-1} X^{r-1} + X^r$ 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.

اگر $g_0 = 0$ باشد.

$$\underline{v} = (0, g_1, g_2, \ldots, g_{r-1}, 1, 0\ 0 \ldots 0)$$

$$v^{(n-1)} = (g_1, g_2, \ldots, g_{r-1}, 1, 0 \ldots 0) \Rightarrow v^{(n-1)}(x) = g_1 + g_2 x + \cdots + x^{r-1}$$

خلاف فرض است

بنابراین $g_0 = 1$ است.

یعنی $g(x)$ generator است.

ص ۷۵۲

THEOREM 5.3 Let $g(X) = 1 + g_1 X + \cdots + g_{r-1} X^{r-1} + X^r$ be the nonzero code polynomial of minimum degree in an (n, k) cyclic code C. A binary polynomial of degree $n-1$ or less is a code polynomial if and only if it is a multiple of $g(X)$.

$$v(x) = v_0 + v_1 x + \cdots + v_{n-1} x^{n-1} = a(x) g(x) = (a_0 + a_1 x + \cdots + a_{n-r-1} x^{n-r-1}) g(x)$$

$$= a_0 g(x) + a_1 x g(x) + \cdots + \underbrace{a_{n-r-1} x^{n-r-1} g(x)}_{} \in C(n,k) \qquad g(x) \in C(n,k)$$

$$(1, g_1, \ldots, g_{r-1}, 1, 0\ 0\ 0 \ldots 0)$$

$$(0\ 0 \ldots 1, g_1, g_2, \ldots, g_{r-1}\ 1)$$

چون shift یافته‌های $g(x)$ شد.

درجه r

ص ۷۵۳

اگر $v(x)$ یک کلمه کد معتبر باشد اما $g(x)$ این چندجمله‌ای را عاد نکند

$$v(x) = q(x) g(x) + b(x) \Rightarrow b(x) = \underbrace{v(x)}_{\in C(n,k)} + \underbrace{q(x) g(x)}_{\in C(n,k)} \Rightarrow b(x) \in C(n,k)$$

درجه‌ی $b(x)$ حداکثر $r-1$ می‌باشد.

اما درجه‌ی $b(x)$ از r کمتر است پس خلاف فرض در نظر گرفته شده مبنی $g(x)$ است. پس $b(x) = 0$ است.

$V(x) = V_0 + V_1 x + \cdots + V_{n-1} x^{n-1} = (u_0 + u_1 x + \cdots + u_{k-1} x^{k-1}) g(x)$

$\Rightarrow \boxed{r = n-k}$

پس درجه $g(x)$ برابر تعداد بیت‌های پرچک است.

$\boxed{g(x) | v(x)}$

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 $v(X)$ in an (n, k) cyclic code can be expressed in the following form:

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

If the coefficients of $u(X)$, $u_0, u_1, \cdots, u_{k-1}$, are the k information digits to be encoded, $v(X)$ is the corresponding code polynomial. Hence, the encoding can be achieved by multiplying the message $u(X)$ by $g(X)$. Therefore, an (n, k) cyclic code is completely specified by its nonzero code polynomial of minimum degree, $g(X)$, given by (5.4). The polynomial $g(X)$ is called the *generator polynomial* of the code. The degree of $g(X)$ 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 $g(X) = 1 + X + X^3$. We see that each code polynomial is a multiple of $g(X)$.

$v(x) = u(x)g(x)$

$x^i v(x) = q(x)(x^n+1) + v^{(i)}(x)$

$\Rightarrow v^{(i)}(x) = x^i v(x) + q(x)(x^n+1)$

$= g(x)(x^i u(x) + q(x)h(x)) \longrightarrow g(x) \mid v^{(i)}(x) \Rightarrow v^{(i)}(x) \in C(n,k)$

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

$g(x) = (1 \; g_1 \; g_2 \cdots g_{n-k-2} \; 1 \; 0 \cdots 0)$, $g(x) = 1 + g_1 x + \cdots + g_{n-k-1} x^{n-k-1} + x^{n-k}$

$\rightarrow x^i g(x) = q(x)(x^n+1) + g^{(i)}(x)$ $\qquad v^{(i)}(x) = x^i v(x) \bmod x^n+1$

$\Rightarrow x^i g(x) + g^{(i)}(x) = q(x)(x^n+1) \Rightarrow g(x)(x^i + g'(x)) = q(x)(x^n+1)$

$g'(x) g(x)$

اگر $i=1$ قرار دهیم $g(x)=1$ فراهم شد

$\Rightarrow x^n+1 = g(x) \cdot b(x) \Rightarrow g(x) \mid x^n+1$

$xv(x) = v_0 x + v_1 x^2 + \cdots + v_{n-2} x^{n-1} + v_{n-1} x^n$

(A) اگر $v(x)$ یک کد به صورت فوق باشد،

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

$g(x) \mid v(x)$ و $g(x) \mid x^n+1$ بنابراین $g(x) \mid v^{(1)}(x)$ بنابراین $v^{(1)}(x)$ یکی از کلمات کد ساخته شده به صورت (A) می باشد.

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 $g(X)$ is a polynomial of degree $n-k$ and is a factor of X^n+1, then $g(X)$ generates an (n, k) cyclic code.

روش 2
با روش صفحه

$g^{(i)}(x) = x^i g(x) \bmod x^n+1$

$i=k \Rightarrow g^{(k)}(x) = x^k \underbrace{g(x)}_{n-k} \bmod x^n+1 \Rightarrow x^k g(x) = (1+x^n) + g^{(k)}(x)$

$g^{(k)}(x) \in (n,k)$ $\qquad g^{(k)}(x) = a(x)g(x)$ $\qquad x^i v(x) = q(x)(x^n+1) + v^{(i)}(x)$

$x^n+1 = x^k g(x) + a(x)g(x) = (x^k + a(x))g(x)$

$\Rightarrow g(x) \mid x^n+1$

5.6

$g(x) \mid x^n+1$ $\qquad \deg(g(x)) = n-k$ $\qquad , v(x) = u(x)g(x) = (u_0 + u_1 x + \cdots + u_{k-1} x^{k-1})g(x)$ $\quad 2^k$ تعداد

$n-1$ درجه

$v(x)$ به این ترتیب تولید می شود

$= u_0 + u_1 x g(x) + \cdots + u_{k-1} x^{k-1} g(x)$

(1)

ادامه A

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.

V56

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 $\mathbf{g}(X) = 1 + X + X^3$. 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 $\mathbf{g}(X) = 1 + X + X^3$. For example, let $\mathbf{u} = (1\,0\,1\,0)$ be the message to be encoded. The corresponding message polynomial is $\mathbf{u}(X) = 1 + X^2$. Multiplying $\mathbf{u}(X)$ by $\mathbf{g}(X)$ gives us the following code polynomial:

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

or the codeword (1 1 1 0 0 1 0).

$g_1(x) = x+1$ $c(7,6)$

$g_2(x) = 1+x+x^3$ $c(7,4)$

$g_3(x) = 1+x^2+x^3$ $c(7,4)$

$g_4(x) = (1+x)(1+x+x^3)$ $c(7,3)$

$g_5(x) = (1+x)(1+x^2+x^3)$ $c(7,3)$

$g_6(x) = (1+x^2+x^3)(1+x+x^3)$ $c(7,1)$

سیستماتیک نیست

$v(x) = u(x)\,g(x)$ (n-1, k-1, n-k)

$\Rightarrow (v_0, v_1, \dots, v_{n-1}) \rightarrow (b_0, b_1, \dots, b_{n-k-1}, u_0, u_1, \dots, u_{k-1})$

$\Rightarrow u(x) = u_0 + u_1 x + \dots + u_{k-1} x^{k-1} \xrightarrow{\times x^{n-k}} x^{n-k} u(x) = u_0 x^{n-k} + \dots + u_{k-1} x^{n-1}$

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

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

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

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

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

$$X^{n-k}\mathbf{u}(X) = \mathbf{a}(X)\overset{n-k}{\mathbf{g}(X)} + \overset{n-k-1}{\mathbf{b}(X)} \tag{5.6}$$

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

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

$v(x) = x^{n-k} a(x) + b(x) = a(x)\,b(x) \in C(n,k)$

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

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

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

$$\begin{aligned}\mathbf{b}(X) + X^{n-k}\mathbf{u}(X) = {} & b_0 + b_1X + \cdots + b_{n-k-1}X^{n-k-1} \\ & + u_0X^{n-k} + u_1X^{n-k+1} + \cdots + u_{k-1}X^{n-1},\end{aligned} \tag{5.8}$$

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: تبدیل کردن به کد سیستماتیک

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 $\mathbf{b}(X) + X^{n-k}\mathbf{u}(X)$.

ویژگی های کد تغییر نمی کند علت (min 13 ص56)

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 $\mathbf{g}(X)$,

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

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

GENERATOR AND PARITY-CHECK MATRICES OF CYCLIC CODES

Consider an (n, k) cyclic code C with generator polynomial $g(X) = g_0 + g_1X + \cdots + g_{n-k}X^{n-k}$. 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:

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

$\Rightarrow \underset{1\times n}{V} = \underset{1\times k}{U} \cdot \underset{k\times n}{G}$

$$V(x) = (u_0 + u_1x + \cdots + u_{k-1}x^{k-1})(1 + g_1x + \cdots + g_{n-k-1}x^{n-k-1} + x^{n-k})$$

$$= u_0g(x) + u_1xg(x) + \cdots + u_{k-1}x^{k-1}g(x)$$

$g(x) \to \underline{g}_0 = (1 \;\; g_1 \;\; g_2 \cdots g_{n-k-1} \;\; 1 \;\; 0 \;\; 0 \cdots 0)$

$xg(x) \to \underline{g}_1 = (0 \;\; 1 \;\; g_1 \;\; g_2 \cdots g_{n-k-1} \;\; 1 \;\; 0 \cdots 0)$

$x^2g(x) \to \underline{g}_2 = (0 \;\; 0 \;\; 1 \;\; g_1 \;\; g_2 \cdots g_{n-k-1} \;\; 1 \;\; 0 \cdots 0)$

k بردار مستقل خطی اند

بدلیل وجود قطری که تمام ۱ است باعث می شود که وقتی که داشته باشیم فقط موقعی صفر شود که تمامی ضرایب برابر صفر شود.

برای اینکه سیستماتیک شود جای I کنیم

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

وجود ندارد

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

shift

$G = [P \; I]$

برای که سیستماتیک

پس systematic نیست

بس مستقل خطی هستند

برای صفر کردن و تبدیل به I جلو سطر اول را در سطر سوم جمع کنیم. (روش مشابه برای سطر دوم)

Clearly, $\mathbf{G}$ 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:

$$\mathbf{G}' = \begin{bmatrix} \overbrace{1 \;\; 1 \;\; 0}^{P} & \overbrace{1 \;\; 0 \;\; 0 \;\; 0}^{I} \\ 0 \;\; 1 \;\; 1 & 0 \;\; 1 \;\; 0 \;\; 0 \\ 1 \;\; 1 \;\; 1 & 0 \;\; 0 \;\; 1 \;\; 0 \\ 1 \;\; 0 \;\; 1 & 0 \;\; 0 \;\; 0 \;\; 1 \end{bmatrix},$$

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

توضیح تکمیلی

* نباید ترتیب ستونی ترکیب کنیم.

با جابجا کردن ستون های G زیر فضا عوض می شود ولی cyclic بودن تغییر نمی کند.

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

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

$(1 + g_1 x + \cdots + x^{n-k})$

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

$$\mathbf{h}(X) = \underbrace{h_0}_{1} + h_1 X + \cdots + \underbrace{h_k}_{1} X^k$$

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

درجه $n-1$ $k-1$ $n-k$

$v(x) = x^{n-k} u(x) + bx = a(x) g(x)$

$v(x) = u(x) \cdot g(x)$

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

$(v_0 + v_1 x + \cdots + v_{n-2} x^{n-2} + v_{n-1} x^{n-1})(h_0 + h_1 x + \cdots + h_k x^k)$

$x^k \overbrace{(v_0 h_k + v_1 h_{k-1} + \cdots + v_k h_0)}^{0} \Rightarrow v_0 h_k + \cdots + v_k h_0 = 0$

$\vdots$

$x^{n-1} \overbrace{(v_{n-k-1} h_k + \cdots + v_{n-2} h_1 + v_{n-1} h_0)}^{0} \Rightarrow v_{n-k-1} h_k + \cdots + v_{n-1} h_0 = 0$

حداقل درجه n

حداکثر درجه $n+k-1$

$x^k, x^{k+1}, \ldots, x^{n-1}$

$0 \times x^k$

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:

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

(A) $x^k h(x^{-1}) \,/\, x^n+1$; $x^n+1 = g(x).h(x) \Rightarrow$

reciprocal $x^n(1+\frac{1}{x^n}) = x^n g(\frac{1}{x}) h(\frac{1}{x}) \Rightarrow x^n+1 = x^{n-k} g(\frac{1}{x}) x^k h(x^{-1})$

$= g_1(x).\, x^k h(x^{-1})$

$\Rightarrow x^k h(x^{-1}) \mid x^n+1$ ✓

$h(x) = h_0 + h_1 x + \cdots + h_k x^k$ $\qquad v.H^T = 0$

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

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

parity polynomial

It follows from the $n-k$ equalities of (5.12) that any codeword $\mathbf{v}$ 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 $\mathbf{g}(X)$. The dual code of C is also cyclic and is generated by the polynomial $X^k\mathbb{h}(X^{-1})$, where $\mathbb{h}(X) = (X^n+1)/\mathbf{g}(X)$.

برای cyclic جامع فاکتوری x^n+1 باشد.

(A)

EXAMPLE 5.3

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

$n = 2^m - 1$

$$\mathbf{h}(X) = \frac{X^7 + 1}{\mathbf{g}(X)}$$

متناظر به H نیست

$$= 1 + X + X^2 + X^4.$$

The reciprocal of $\mathbf{h}(X)$ is

$$X^4\mathbf{h}(X^{-1}) = X^4(1 + X^{-1} + X^{-2} + X^{-4})$$

متناظر به H است.

$$= 1 + X^2 + X^3 + X^4.$$

$C_d(7,3)$ کد دوگان Generator

This polynomial $X^4\mathbf{h}(X^{-1})$ divides $X^7 + 1$: $(X^7 + 1)/X^4\mathbf{h}(X^{-1}) = 1 + X^2 + X^3$. If we construct all the codewords of the (7, 3) code generated by $X^4\mathbf{h}(X^{-1}) = 1 + X^2 + X^3 + X^4$, 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.

$1 + l \cdot 1 = 4 \Rightarrow l = 2$

$2t + 2 \cdot 4 \geq t + 1$

$G \to H$

$v = u \cdot G$

$G \cdot H^T = 0$

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

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

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

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

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

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

x^0, x^1, x^{k-1}

$x^{n-k} u(x) \bmod g(x) = b(x)$

با عملیات سطری I را می‌سازیم.

$x^{n-k-i} = a_i(x) g(x) + b(x)$

$i = 0, 1, \ldots, k-1$

$H = [I_{n-k} \; P^T]$

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

$X^{n-k+i} \bmod g(x) = b_i(x)$

$i = 0, 1, 2, 3$

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

$n = 7$

$k = 4$

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:

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

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

---

* توضیح مهم ~

1:15 min

V58

Comments: Division and Multiplication Circuits

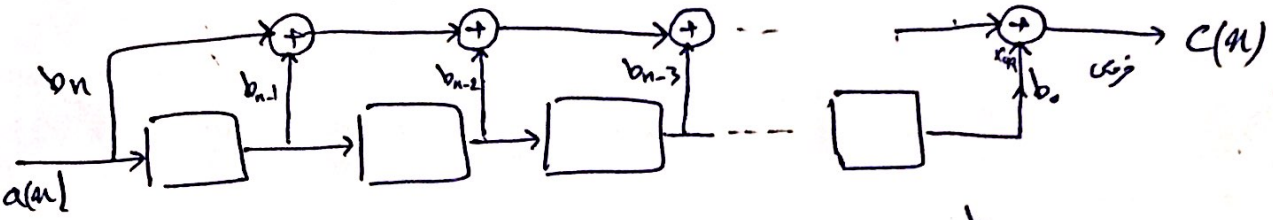
فرض کنید بخواهیم $a(x)$ و $b(x)$ را در هم ضرب کنیم

$$a(x) = a_0 + a_1 x + \cdots + a_m x^m \qquad a_i \in \{0,1\}$$

$$b(x) = b_0 + b_1 x + \cdots + b_n x^n$$

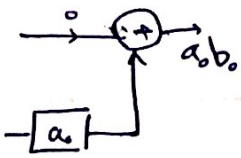
$$c(x) = a(x)\cdot b(x) = (a_0 + a_1 x + \cdots + a_{m-1}x^{m-1} + a_m x^m)(b_0 + b_1 x + \cdots b_{n-1}x^{n-1} + b_n x^n)$$

$$= a_0 b_0 + (a_0 b_1 + a_1 b_0)x + \cdots + (a_{m-1}b_n + a_m b_{n-1})x^{n+m-1} + a_m b_n x^{n+m}$$



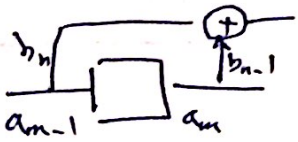
$a_0, a_1, \ldots, a_m$

$0, 0, 0$

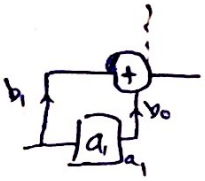


در لحظه‌ی اول:

① $a_m \xrightarrow{b_n} a_m \times b_n$



$\Rightarrow a_{m-1}b_n + a_m b_{n-1}$



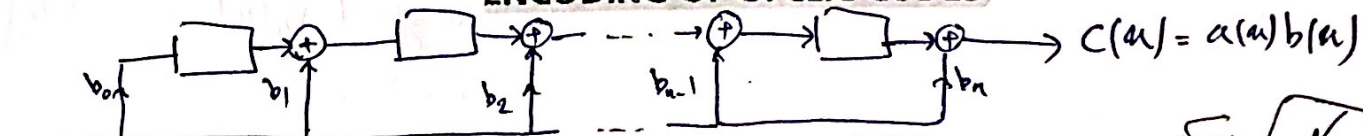
$\Rightarrow a_0 b_1 + a_1 b_0$

$$\Rightarrow c(x) = c_0 + c_1 x + \cdots + c_{n+m}x^{n+m}$$

$(c_0, c_1, \ldots, c_{n+m})$

$\Leftarrow$ تعداد کلاک‌ها برابر است با $m+n$

ENCODING OF CYCLIC CODES



$a_0, a_1, \ldots, a_m$

ادامه برگه

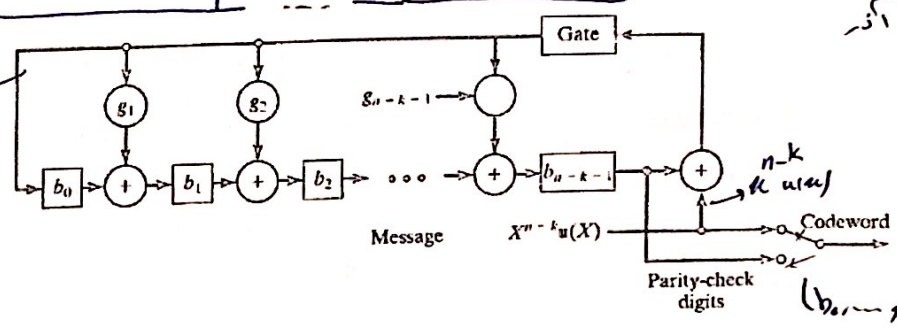


FIGURE 5.1: Encoding circuit for an (n, k) cyclic code with generator polynomial $g(X) = 1 + g_1 X^2 + \cdots + g_{n-k-1}X^{n-k-1} + X^{n-k}$.

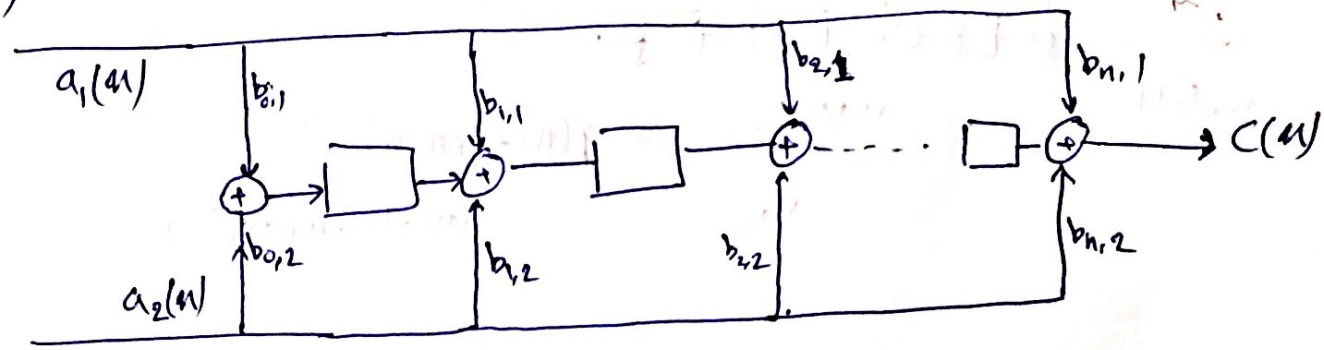
~58min

V59

۱۱/۵

فرض کنید بخواهیم $c(x) = a_1(x)b_1(x) + a_2(x)b_2(x)$ (ضرب)

• $\deg(a_1(x)) = m_1$



$\deg(a_2(x)) = m_2$

$\deg(b_1(x)) = n_1$

$\deg(b_2(x)) = n_2$

$\Rightarrow$ $n = \max(n_1, n_2)$

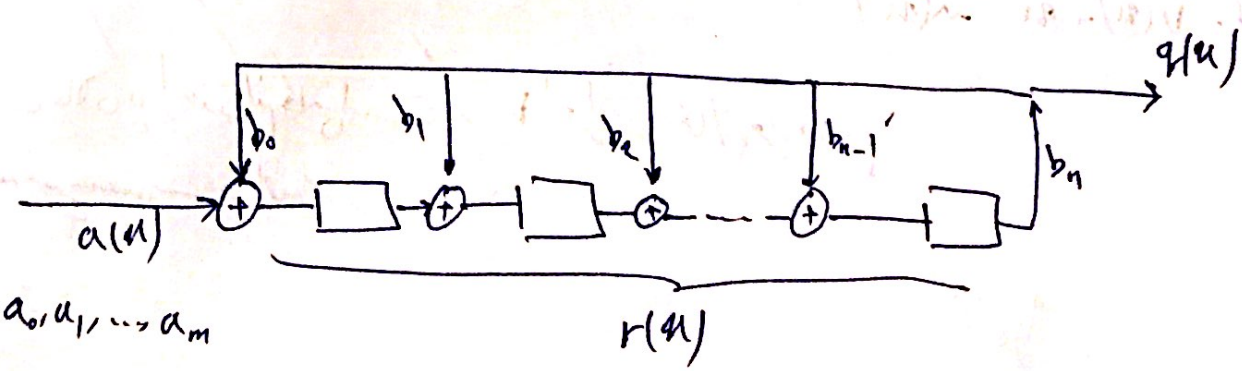
$m = \max(m_1, m_2)$ $\longrightarrow$ تعداد کلاک ها $= m+n$

اگر بخواهیم عمل تقسیم دو چند جمله‌ای را انجام دهیم

$a(x) = b(x)q(x) + r(x)$

$a(x) = a_0 + a_1 x + \dots + a_m x^m$

$b(x) = b_0 + b_1 x + \dots + b_n x^n$



۲۰۲

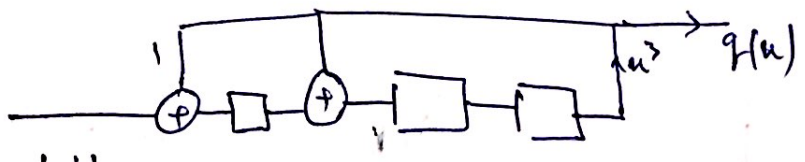
40min

تمرین 7

مثال)

$a(x) = x^3 + x^5 + x^6$

$b(x) = 1 + x + x^3$



0001011

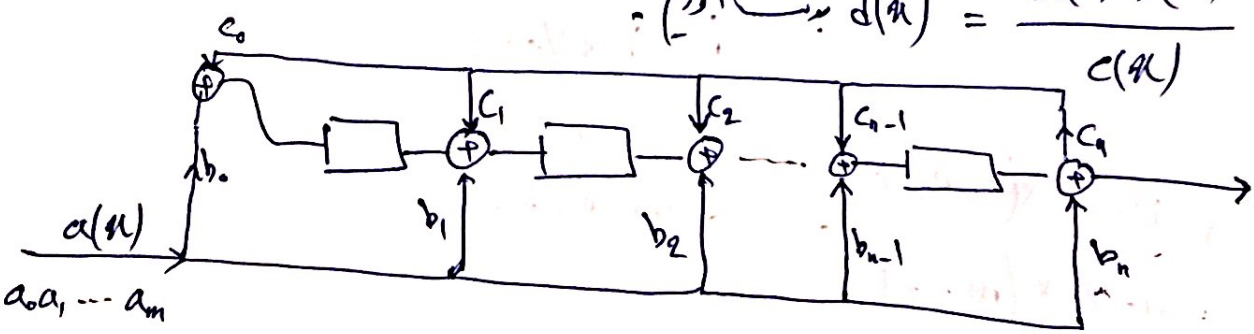
1111000

$\Rightarrow q(x) = 1 + x + x^2 + x^3$

100

$r(x) = 1 + 0x + 0x^2 = 1$

اگر بخواهیم $d(x) = \dfrac{a(x) \cdot b(x)}{c(x)}$ بدست آوریم.



$q(x) b(x) = q(x) c(x) + r(x)$

قسمت پایینی ضرب کننده است.

$v(x) = u(x) g(x)$

برای اینکه سیستماتیک باشد

$x^{n-k} u(x) \bmod g(x) = b(x)$

$v(x) = b(x) + x^{n-k} u(x)$

* اگر قسمت بالای کد را حذف کنیم کافی است که b را ضرایب g(x) قرار دهیم
⇐ یک کد غیر سیستماتیک خواهد شد.

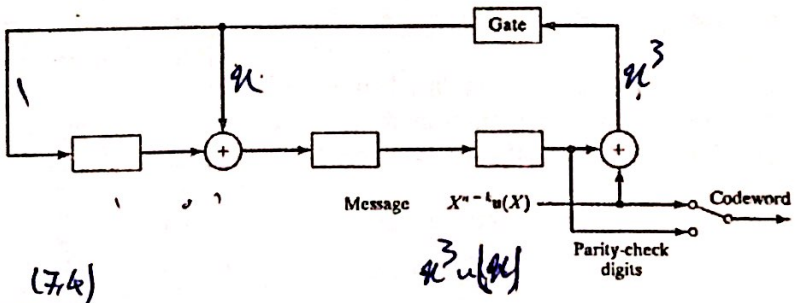
مریم

Step 1. Turn on the gate. The k information digits $u_0, u_1, \cdots, u_{k-1}$ [or $\mathbf{u}(X) = u_0 + u_1 X + \cdots + u_{k-1} X^{k-1}$ in polynomial form] are shifted into the circuit and simultaneously into the communication channel. Shifting the message $\mathbf{u}(X)$ into the circuit from the front end is equivalent to premultiplying $\mathbf{u}(X)$ 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.

*



(7,4)

EXAMPLE 5.5

Consider the (7, 4) cyclic code generated by $\mathbf{g}(X) = 1 + X + X^3$. The encoding circuit based on $\mathbf{g}(X)$ is shown in Figure 5.2. Suppose that the message $\mathbf{u} = (1011)$ is to be encoded. As the message digits are shifted into the register the contents of the register change as follows:

Input	Register contents
	000 (initial state)
1	110 (first shift)
1	101 (second shift)
0	100 (third shift)
1	100 (fourth shift)

$\Rightarrow V = 1001011$

1011

1 → 110

1 → 101

After four shifts, the contents of the register are (100). Thus, the complete codeword is (1001011), 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 $\mathbf{h}(X) = h_0 + h_1 X + \cdots + h_k X^k$. Let $\mathbf{v} = (v_0, v_1, \cdots, v_{n-1})$ be a codeword. We have shown in Section 5.2 that the components of $\mathbf{v}$ satisfy the $n-k$ equalities

$V = (V_0, V_1, \ldots, V_{n-k-1}, u_0, \ldots, u_{k-1})$

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

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

$V_{n-k-1} = \sum_{i=0}^{k-1} h_i V_{n-i-1}$

$V_{n-k-2} = \ldots$

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 $\mathbf{h}(X)$. (Note that $h_0 = h_k = 1$.) The encoding operation can be described in the following steps:

$V_0 = \sum_{i=0}^{k-1} h_i V_{n-i-(n-k)}$

$k-i$

Step 1. Initially, gate 1 is turned on and gate 2 is turned off. The k information digits $\mathbf{u}(X) = u_0 + u_1 X + \cdots + u_{k-1} X^{k-1}$ 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.

وقتی u_0 در خروجی به وجود آمد

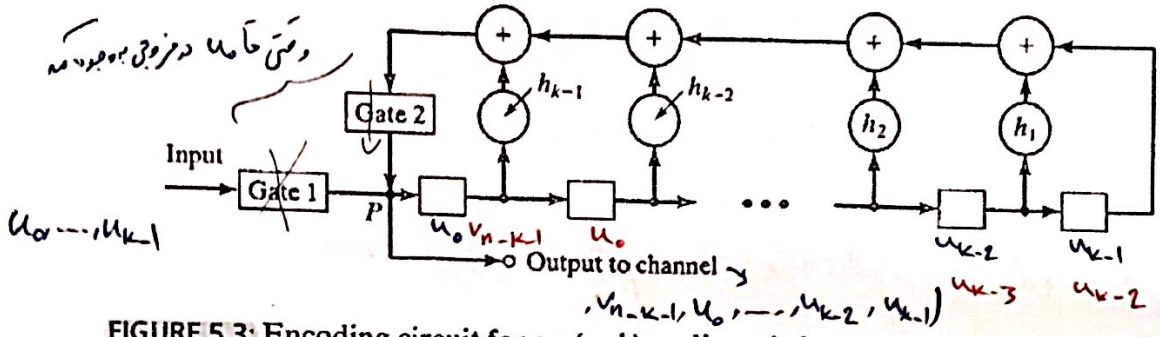


FIGURE 5.3: Encoding circuit for an (n, k) cyclic code based on the parity polynomial $\mathbf{h}(X) = 1 + h_1 X + \cdots + X^k$.

$\Rightarrow h_0 u_{k-1} + h_1 u_{k-2} + \dots + h_{k-1} u_0$

$v_{n-k-1} = h_0 u_{k-1} + h_1 u_{k-2} + \dots + h_{k-1} u_0$

صرحیه طول شیفت k تاست

*در مدار فوق از یک shift register به طول k استفاده می‌شود درحالیکه در مدار قبلی طول شیفت رجیستر برابر n-k شیفت بعد، بنابراین اگر $n-k > k$ باشد، استفاده از مدار فوق مقرون به صرفه است.

EXAMPLE 5.6

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

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

The encoding circuit based on $\mathbf{h}(X)$ is shown in Figure 5.4. Each codeword is of the form $\mathbf{v} = (v_0, v_1, v_2, v_3, v_4, v_5, v_6)$, 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

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

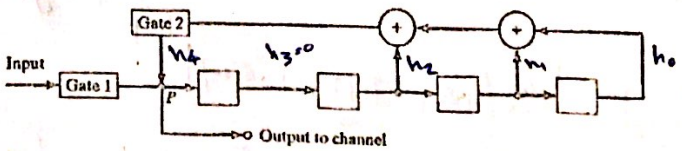


FIGURE 5.4: Encoding circuit for the (7, 4) cyclic code based on its parity polynomial $\mathbf{h}(X) = 1 + X + X^2 + X^4$.

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.

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

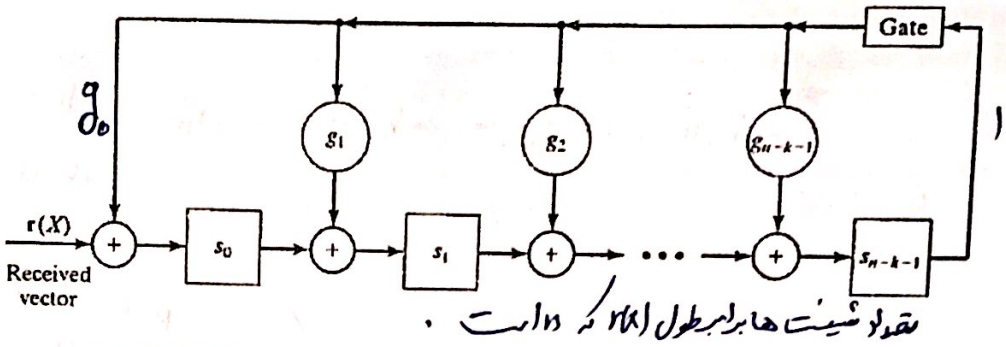
Dividing $\mathbf{r}(X)$ by the generator polynomial $\mathbf{g}(X)$, we obtain

حداکثر $n-k-1$

$$\mathbf{r}(X) = \mathbf{a}(X)\overset{n-k \text{ درجه}}{\mathbf{g}(X)} + \mathbf{s}(X)$$

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

اگر جمله‌ی باقی‌مانده باشیم s صفر نمی‌شود ولی اگر جمله‌ی باقی‌مانده باشیم صفر می‌شود و عادی کند.



تعداد شیفت‌ها برابر طول r(x) که داریم.

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

یک شیفت به صورت cyclic به دست آمده است

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

$xs(x)$

$$\boxed{\begin{aligned} r(x) \bmod g(x) &= s(x) \\ r^{(1)}(x) \bmod g(x) &= s^{(1)}(x) \end{aligned}}$$

حسابی داشتیم:

$$x^i v(x) = q(x)(x^n+1) + v^{(i)}(x)$$

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

$$\Rightarrow xr(x) = r_{n-1}(x^n+1) + r^{(1)}(x) \Rightarrow r^{(1)}(x) = r_{n-1}(x^n+1) + xr(x) \quad (1)$$

اگر 2 طرف عبارت فوق را بر $g(X)$ تقسیم کنیم

$$\Rightarrow c(x)g(x) + p(x) = r_{n-1} g(x) h(x) + x(a(x)g(x) + s(x)) \quad (2)$$

$$\Rightarrow xs(x) = g(x)\left(c(x) + r_{n-1} h(x) + xa(x)\right) + p(x) \Rightarrow \boxed{xs(x) \bmod g(x) = p(x)} = s^{(1)}(x)$$

$p(x)$، در واقع طبق رابطه‌ی (1) و (2)، همان $r^{(1)}(x)$ می‌باشد.

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

EXAMPLE 5.7

صفحه اول 62

A syndrome circuit for the (7, 4) cyclic code generated by $\mathbf{g}(X) = 1 + X + X^3$ 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 $\mathbf{r}$.

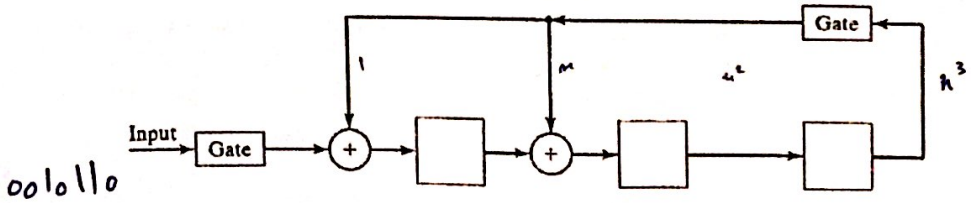


FIGURE 5.6: Syndrome circuit for the (7, 4) cyclic code generated by $\mathbf{g}(X) = 1 + X + X^3$.

TABLE 5.3: Contents of the syndrome register shown in Figure 5.6 with $\mathbf{r} = (0010110)$ 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 s)
8	—	100 (syndrome $s^{(1)}$)
9	—	010 (syndrome $s^{(2)}$)

$s(x) = 1 + x^2$

$xs(x) = x(1+x^2) \bmod g(x)$

$s^{(1)}(x)$

$r(x)$

0010110

$r^{(1)}(x)$

0001011

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

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

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

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

where $\mathbf{b}(X)\mathbf{g}(X) = \mathbf{v}(X)$. 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 $\mathbf{e}(X)$ is unknown to the decoder. Therefore, the decoder has to estimate $\mathbf{e}(X)$ based on the syndrome $\mathbf{s}(X)$. If $\mathbf{e}(X)$ is a coset leader in the standard array and if table-lookup decoding is used, $\mathbf{e}(X)$ can correctly be determined from the syndrome.

From (5.27) we see that $\mathbf{s}(X)$ is identical to zero if and only if either the error pattern $\mathbf{e}(X) = 0$ or $\mathbf{e}(X)$ is identical to a codeword. If $\mathbf{e}(X)$ is identical to a code polynomial, $\mathbf{e}(X)$ 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.

$\Rightarrow e(x) \bmod g(x) = e(x) = b(x)$ $\qquad e = (1\,x\,x\,x\,1\,0\,0\,\cdots\,0)$ ④

$r(x) \bmod g(x) = s(x) \neq 0 \qquad b(x) = s(x)$

Now, we investigate the error-detecting capability of an (n, k) cyclic code. Suppose that the error pattern $\mathbf{e}(X)$ 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 $\mathbf{e}(X)$ in the following form:

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

* خطاهای burst بطول n-k را می‌توانیم تشخیص دهیم.

where $0 \le j \le n-1$, 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 $\mathbf{g}(X)$, $\mathbb{B}(X)$ is not divisible by $\mathbf{g}(X)$. Since $\mathbf{g}(X)$ is a factor of $X^n + 1$, and X is not a factor of $\mathbf{g}(X)$, $\mathbf{g}(X)$ and X^j must be relatively prime. Therefore, $\mathbf{e}(X) = X^j \mathbb{B}(X)$ is not divisible by $\mathbf{g}(X)$. As a result, the syndrome caused by $\mathbf{e}(X)$ 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, ④

اگر یک خطای burst بطول حداکثر n-k رخ دهد.

$$\mathbf{e} = (1\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 0\ 1)$$

1: burst

$B(x) = B_0 + B_1 x + \cdots + B_{n-k-1} x^{n-k-1}$

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: ④

③ $\boxed{x\,x\,1 \quad 1\,x\,x\,1}$

$r(x) = v(x) + e(x)$

$r(x) \bmod g(x) = s(x)$

$e(x) = \cancel{x^i} B(x) = s(x)$; $g(x) = 1 + g_1 x + \cdots + g_{n-k-1} x^{n-k-1} + x^{n-k}$

$x^i \nmid g(x)$

$1\,x\,x\,x\,x\,x\,1$

$e = (B_0, B_1, \cdots, B_{n-k-1}, \cdots)$

دوبیت می

$\Rightarrow e(x) = x^j B(x)$

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. شرح V62

$e(x) = x^i B(x)$ & $B(x) = 1 + b_1 x + \cdots + b_{n-k-1} x^{n-k-1} + x^{n-k} \rightarrow$ (n-k+1) دوبیتی بطول 2^{n-k-1} burst تعداد این برابر

$\rightarrow e = (1\ b_1 \cdots b_{n-k-1}, 1\,0\,0\,0\,0\,0)$ & $r(x) = v(x) + e(x) \rightarrow r(x) \bmod g(x) = e(x) \bmod g(x)$

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 $\xrightarrow{\text{برابر}} e(x) = B(x)$

$g(x) = 1 + g_1 x + \cdots + g_{n-k-1} x^{n-k-1} + x^{n-k}$

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

* تنها یک خطای burst قابل تشخیص نیست، این خطای burst چه بود؟

$x^i g(x)$ می‌باشد، پس می‌توان گفت که $\frac{1}{2^{n-k-1}}$ burst از burst های ممکن قابل تشخیص نخواهد بود

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:

$\boxed{e(x) = x^i B(x)}$

$B(x) = 1 + b_1 x + \cdots + b_{l-2} x^{l-2} + x^{l-1}$ (تئوری 5.10) burst افزایش طول

اگر $l > n-k+1$ باشد، تعداد این burst ها برابر 2^{l-2}

$B(x) = e(x) = a(x) \cdot g(x)$ (دوبیتی l، $l-n-k-1$ دوبیتی، $n-k$ دوبیتی)

* اگر ارور بیش برابر فاکتورهای $g(x)$ باشد قابل تشخیص نیست. یعنی

تعداد چندجمله‌ای‌های $a(x)$ برابر $2^{l-(n-k-1)-2}$ می‌باشد بنابراین به نسبت $\frac{2^{l-(n-k)-2}}{2^{l-2}} = 2^{-(n-k)}$ قابل تشخیص نیست

احتمال غیرقابل تشخیص بودن

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

(صفحه ی قبل)

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:

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

به دلیل اینکه طول Burst بالاست.

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.

$n-k=3$ تمام Burst های به طول 3 قابل تشخیص است.

تنها 1/4 Burst ها به طول 4 قابل تشخیص نیستند

تنها 1/8 Burst ها به طول بزرگتر از 4 قابل تشخیص نیستند.

} یعنی با افزایش خطا شرایط بهتر می‌شود !!

DECODING OF CYCLIC CODES

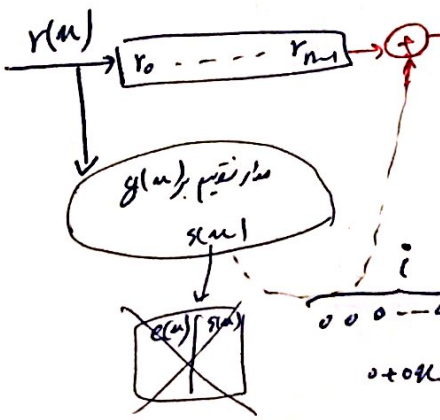
Decoding of cyclic codes consists of the same three steps as for decoding linear codes: syndrome computation, association of the syndrome with an error pattern, and error correction. It was shown in Section 5.4 that syndromes for cyclic codes can be computed with a division circuit whose complexity is 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

$r(x) \bmod g(x) = s(x)$

اگر فرض کنیم $r(x) = v(x) + e(x)$ باشد

Comments:



$r'_0, r'_1, r'_2, \dots, r'_{n-1} = \underline{v}'$ تخمین کد (حتی ممکن است نامعتبر باشد)

$r(x) \bmod g(x) = s(x)$

اگر ورودی $r(x)$ قطع شود و مدار تقسیم ۱ کلاک بخورد محتوای رجیستر میشود برابر $s^{(i)}(x)$ خواهد شد

$r^{(i)}(x) \bmod g(x) = s^{(i)}(x)$

$\overbrace{0\ 0\ 0 \cdots 0}^{i}\ r_0\ r_1 \cdots r_{n-1}$

$0 + 0x + \dots + 0x^{i-1} + r_0 x^i + \dots + r_{n-1} x^{n-1+i}$

$x^i r(x)$

$x^i r(x) = a(x) g(x) + p(x)$ (*)

$x^i r(x) = a'(x)(x^n+1) + r^{(i)}(x)$ $\Rightarrow$

$r^{(i)}(x) = x^i r(x) + a'(x)(x^n+1)$

$s^{(i)}(x) = r^{(i)}(x) \bmod g(x)$

$= x^i r(x) + a'(x)(x^n+1) \bmod g(x) = x^i r(x) \bmod g(x)$

$\frac{g(x)}{x^n+1}$

$= p(x) = s^{(i)}(x)$

$\Rightarrow$ (*) $\Rightarrow x^i r(x) \bmod g(x) = p(x)$

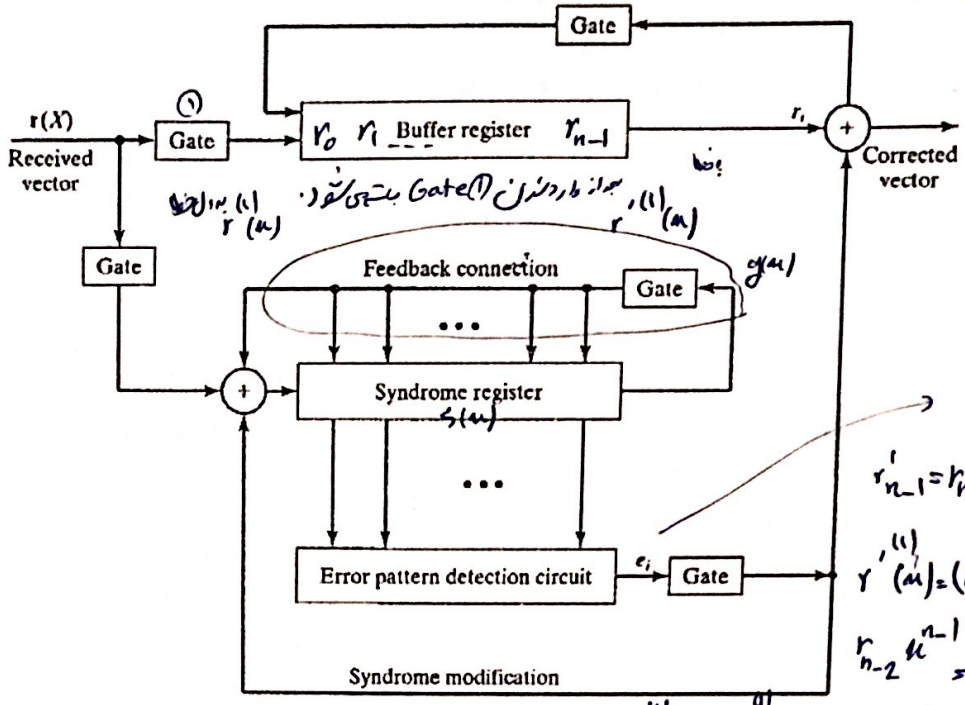


FIGURE 5.8: General cyclic code decoder with received polynomial $\mathbf{r}(X)$ shifted into the syndrome register from the left end.

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.

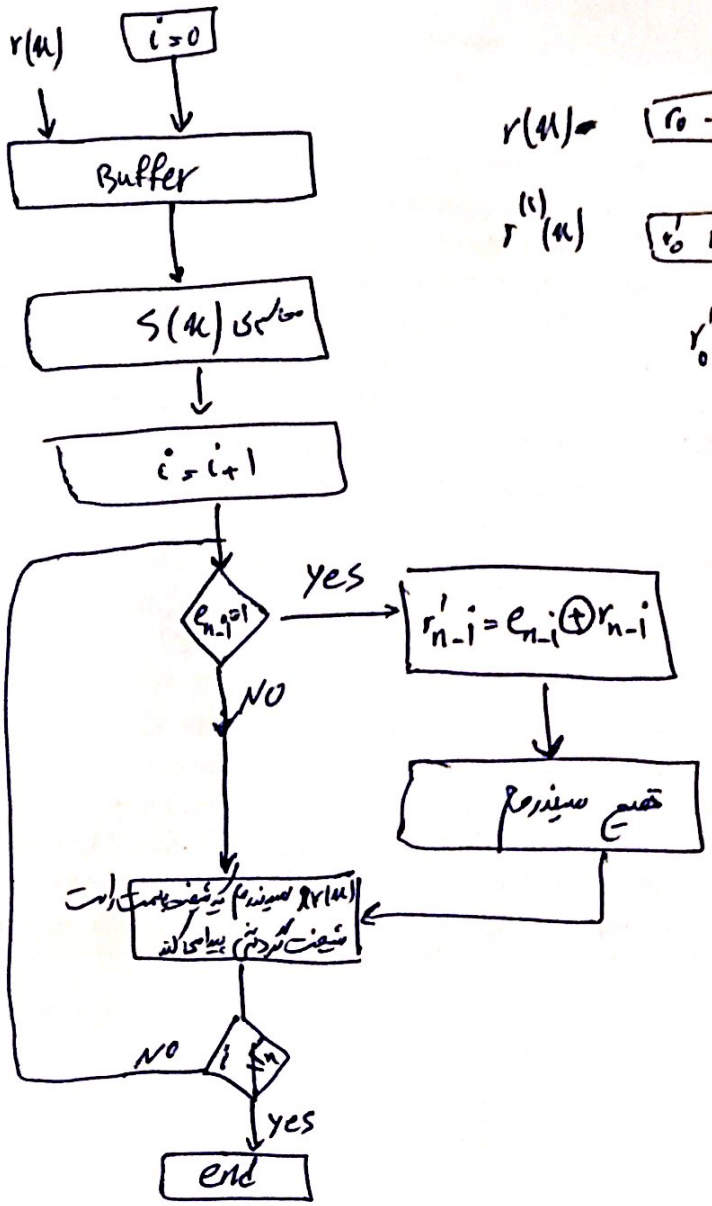
Step 2. The syndrome is read into the detector and is tested for the corresponding error pattern. The detector is a combinational logic circuit that is designed in such a way that its output is 1 if and only if the syndrome in the syndrome register corresponds to a correctable error pattern with an error 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:



$r(x) = r_0 \; - \; - \; - \; r_{n-1}$

$r^{(1)}(x) \quad r'_0 \; r'_1 \; - \; - \; - \; r'_{n-1}$

$r'_0 =$

EXAMPLE 5.9 V65

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.

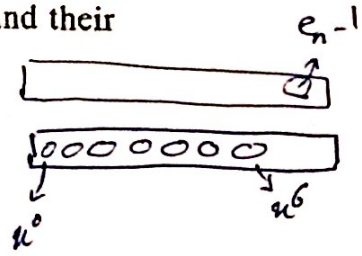


TABLE 5.4: Error patterns and their syndromes with the received polynomial $r(X)$ shifted into the syndrome register from the left end.

Error pattern $e(X)$	Syndrome $s(X)$	Syndrome vector (s_0, s_1, s_2)
$e_6(X) = X^6$	$s(X) = 1 + X^2$	(101)
$e_5(X) = X^5$	$s(X) = 1 + X + X^2$	(111)
$e_4(X) = X^4$	$s(X) = X + X^2$	(011)
$e_3(X) = X^3$	$s(X) = 1 + X$	(110)
$e_2(X) = X^2$	$s(X) = X^2$	(001)
$e_1(X) = X^1$	$s(X) = X$	(010)
$e_0(X) = X^0$	$s(X) = 1$	(100)

$r(x) = v(x) + x^6 \bmod g(x)$

$e_6(x) = x^6 \quad s(x) = 1 + x^2 \quad 101$

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 $0 \le i < 6$. 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.

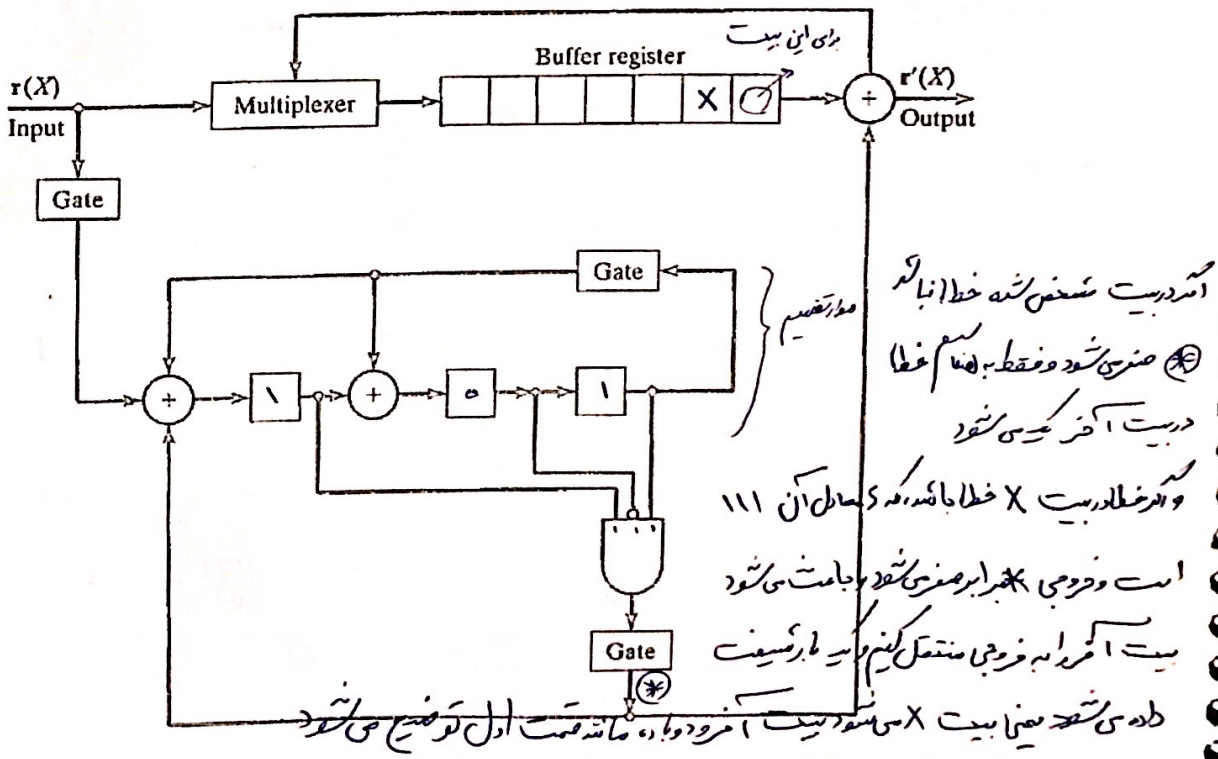


FIGURE 5.9: Decoding circuit for the (7, 4) cyclic code generated by $\mathbf{g}(X) = 1 + X + X^3$.

محل خطا

مثال

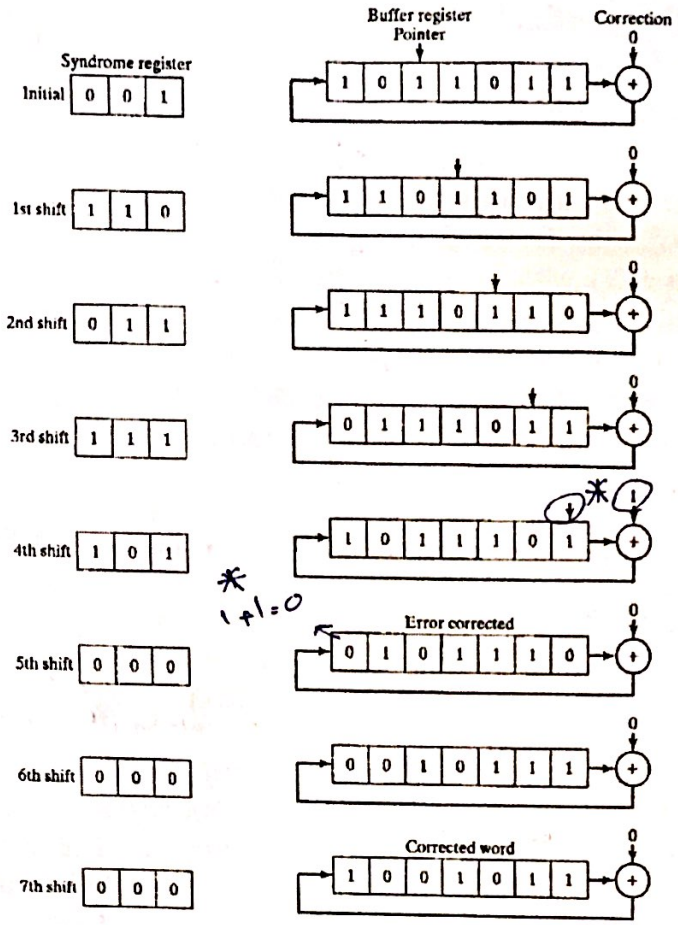
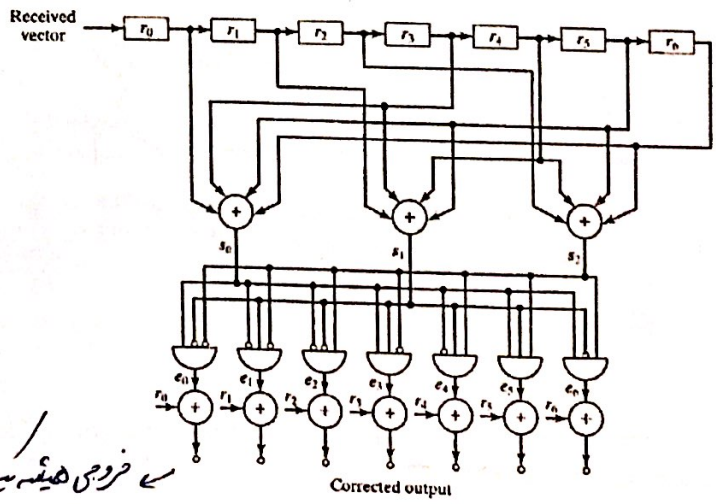


FIGURE 5.10: Error-correction process of the circuit shown in Figure 5.9.

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.



خروجی همیشه یک codeword معتبر است ولی در مدار قبلی (سریالی) اینطور نیست

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

$r(x) = r_0 + r_1 x + \cdots + r_{n-1} x^{n-1}$

$r(x) = v(x) + e(x)$; $e(x) = e_0 + e_1 x + \cdots + e_{n-1} x^{n-1}$

$e^{(1)}(x) = e_{n-1} + e_0 x + \cdots + e_{n-2} x^{n-2}$

$r(x) \longrightarrow s(x)$

$r^{(n-k)}(x) \longrightarrow s^{(n-k)}(x)$

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

$$e^{(n-k)}(x) = \underbrace{e_{n-(n-k)}}_{e_k} + \underbrace{e_{n-(n-k)-1}}_{e} x + \cdots \underbrace{e_{n-1}}_{\text{اگر خطا در محل آخر باشد}} x^{n-k-1} + \rho(X) = X^{n-k-1}. \quad e_0 x^{n-k} + \cdots + e_{k-1} x^{n-1}$$

Therefore,

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

حال اگر فرض کنیم خطا در محل $n-1$ باشد $e^{(n-k)}(x) = e_{n-1} x^{n-k-1}$

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.

دوم

$\Rightarrow$ جمله $s^{(n-k)}(x) = (r^{n-1}(x) + e_{n-1} x^{n-k-1}) \bmod g(x) = s^{(n-k)} + e_{n-1} x^{n-k-1}$

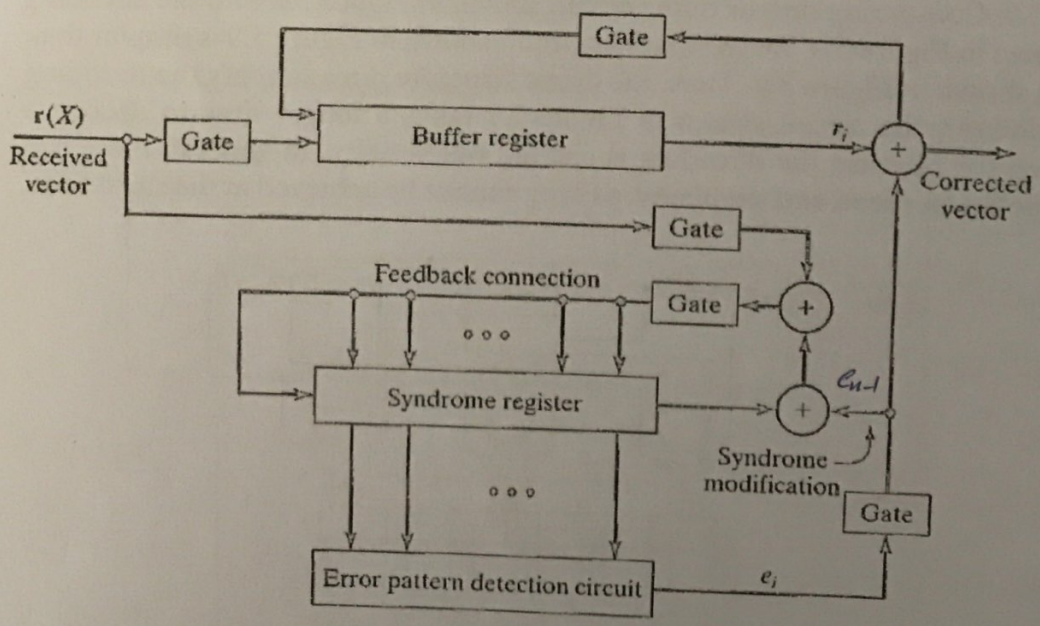


FIGURE 5.11: General cyclic code decoder with received polynomial $\mathbf{r}(X)$ shifted into the syndrome register from the right end.

EXAMPLE 5.10

Again, we consider the decoding of the (7, 4) cyclic code generated by $\mathbf{g}(X) = 1 + X + X^3$. Suppose that the received polynomial $\mathbf{r}(X)$ 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 $\mathbf{e}(X) = X^6$ occurs, the syndrome is (001) after the entire polynomial $\mathbf{r}(X)$ 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 (001) after the entire received polynomial $\mathbf{r}(X)$ has been shifted into the syndrome register; however, after another $6 - i$ shifts, the syndrome register will contain (001). Based on this result, we obtain another decoding circuit for the (7, 4) cyclic code generated by $\mathbf{g}(X) = 1 + X + X^3$, 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.

$e(x) = x^6$

$s^{(n-k)}(x) = s^{(3)}(x)$

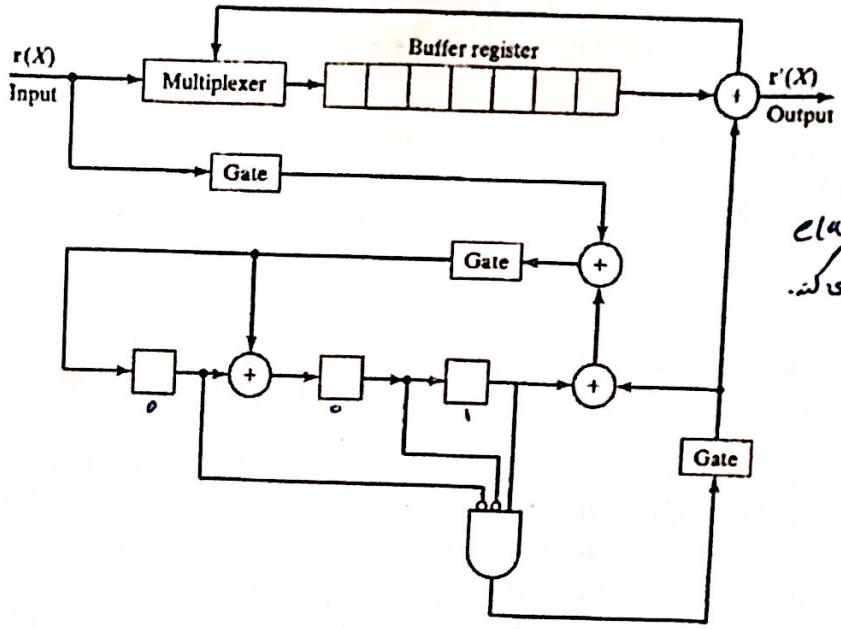
سیندرم این خطا شده

$\boxed{e_0 \cdots\cdots e_6}$

$\boxed{e_4 e_5 e_6 e_0 e_1 e_2 e_3}$

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

Error pattern $\mathbf{e}(X)$	Syndrome $\mathbf{s}^{(3)}(X)$	Syndrome vector (s_0, s_1, s_2)
$\mathbf{e}(X) = X^6$	$\mathbf{s}^{(3)}(X) = X^2$	(001)
$\mathbf{e}(X) = X^5$	$\mathbf{s}^{(3)}(X) = X$	(010)
$\mathbf{e}(X) = X^4$	$\mathbf{s}^{(3)}(X) = 1$	(100)
$\mathbf{e}(X) = X^3$	$\mathbf{s}^{(3)}(X) = 1 + X^2$	(101)
$\mathbf{e}(X) = X^2$	$\mathbf{s}^{(3)}(X) = 1 + X + X^2$	(111)
$\mathbf{e}(X) = X$	$\mathbf{s}^{(3)}(X) = X + X^2$	(011)
$\mathbf{e}(X) = X^0$	$\mathbf{s}^{(3)}(X) = 1 + X$	(110)



$e(x) = x^6$
را تصحیح می کند.

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 $p(X)$ of degree m.

$n = 2^m - 1$

$n - k = m$

$k = 2^m - m - 1$

$H = [\ -\ ]_{m \times 2^m - 1}$

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

اگر $p(x)$، چندجمله‌ای مولد کدما باشد، کدی چندجمله‌ای primitive از درجه m باشد.

$u(x) \times x^{n-k} \bmod g(x) = b(x)$

$\Rightarrow v(x) = b(x) + x^{n-k} \cdot u(x)$

$p(x) \mid x^{2^m - 1} + 1$

$n = 2^m - 1$

$g(x) = p(x)$

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

60min

اگر غلبت کنیم به ازای $i \neq j$ داریم $b_i(x) \neq b_j(x)$

اگر فرض کنیم $i \neq j$ ولی $b_i(x) = b_j(x)$ ($j > i$)

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 $\mathbf{p}(X)$ for $0 \le i < 2^m - m - 1$, we obtain

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

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

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

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

ادامه

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

$x^{i+m} \bmod g(x) = b_i(x) \Rightarrow x^{i+m} = a_1(x)g(x) + b_i(x)$

$x^{j+m} \bmod g(x) = b_j(x) \Rightarrow x^{j+m} = a_2(x)g(x) + b_j(x)$

Rearranging the preceding equation, we have

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

$\Rightarrow x^{i+m} + x^{j+m} = (a_1(x) + a_2(x))g(x) + 0$

$\hookrightarrow x^{i+m}(1 + x^{j-i}) = \underbrace{(a_1(x) + a_2(x))}_{a(x)} g(x)$

$\Rightarrow p(x) \mid 1 + x^{j-i}$ بنابراین باشد

ادامه

خلاف فرض است زیرا $j - i < 2^m - 1$ می‌باشد (توجه کنید که $p(x)$، یک چند جمله‌ای primitive است پس $b_i(x) \neq b_j(x)$ می‌باشد.)

اگر $b_i(x) = 0$ باشد، این فرض غیر ممکن است زیرا $\gcd(p(x), x^{i+m}) = 1$ می‌باشد.

جایگزین کنیم که $x^i \neq b_i(x)$ می‌باشد، $i = 0, 1, \ldots, m-1$

اگر $x^i = b_j$ باشد، در این صورت

$x^{j+m} = a(x)p(x) + b_j(x) = a(x)p(x) + x^i$

$\Rightarrow x^{i+m}(1 + x^{j-i}) = a(x)p(x)$

$\Rightarrow p(x) = g(x);\ a_1(x) + a_2(x) = a(x)$

$\gcd(p(x), x^{i+m}) = 1$

$\boxed{p(x) = g(x)}$

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

ادامه

$a(x)p(x)$

$x^{j+m} + x^i = a(x)p(x) \Rightarrow x^i(1 + x^{j+m-i})$

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

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

مبین استدلال قبلی

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

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

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

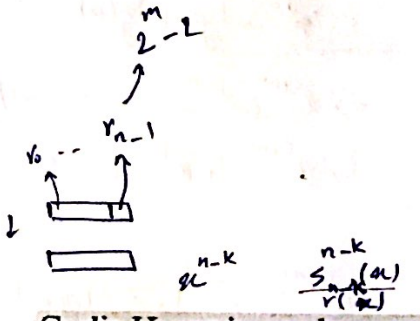
Let $\mathbb{H} = [\mathbb{I}_m \ \mathbb{Q}]$ be the parity-check matrix (in systematic form) of the cyclic code generated by $\mathbb{p}(X)$, where $\mathbb{I}_m$ is an $m \times m$ identity matrix and $\mathbb{Q}$ is an $m \times (2^m - m - 1)$ matrix. Let $\mathbb{b}_i = (b_{i0}, b_{i1}, \cdots, b_{i,m-1})$ 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 $0 \le i < 2^m - m - 1$ 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.

$n = 2^m - 1$

$n - k = m$

$k = 2^m - 1 - m$



V68

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:

$e(x) = x^{2^m-2}$

$p(x)$

$r(x) = r(x) + e(x) \bmod$

$= 0 + x^{2^m-2} \bmod p(x)$

$x^{n-k} r(x) \bmod p(x) = 1 \Rightarrow s^{(n-k)}(x) = s^{(m)}(x) = x^{m-1}$ ✓

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

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:

$x^{m-1} x^{2^m-1} \bmod p(x)$

$p(x) \mid x^{2^m-1} + 1 \rightarrow x^{2^m-1} + 1 = q(x)p(x)$

$\Rightarrow x^{2^m-1} = q(x)p(x) + 1$

$\Rightarrow x^{2^m-1} = 1$

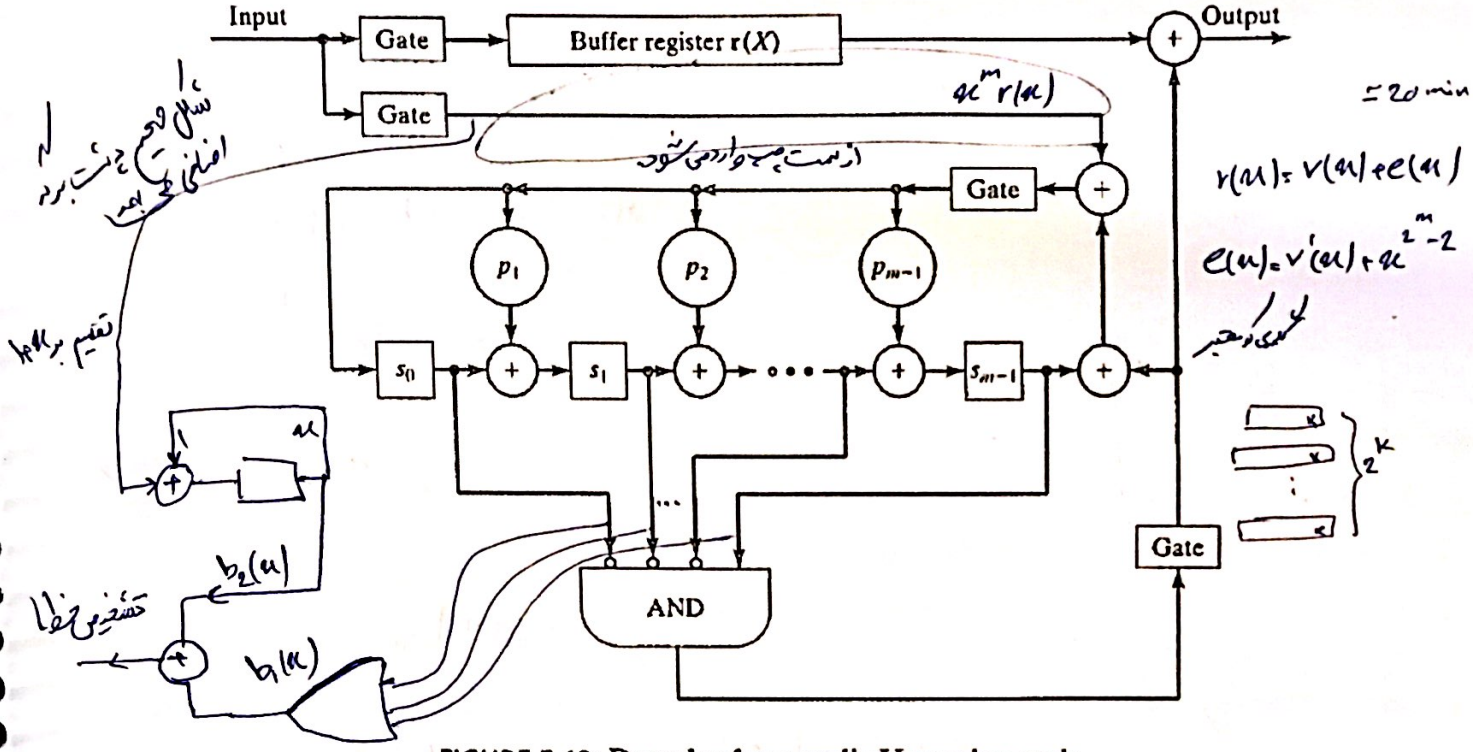


FIGURE 5.13: Decoder for a cyclic Hamming code.

Ⓐ Ⓑ ادامه مبحث (این مبحث تغییرات در مدار فوق) و تغییرات مربوط به مبحث است.

$r(x) = q_1(x)p(x) + b_1(x)$

$r(x) = q_2(x)(1+x) + b_2(x) \quad b_2(x) = 0, 1$

$b_1(x) = 0, b_2 = 0$: خطایی نداریم یا بیشتر از خطا یک لکه‌ای که معتبر در C_2 است.

$b_1(x) = 0, b_2 = 1$ تعداد خطاها فرد است و $r(x)$ در ریاضی یک کد که معتبر در C_1 است.

$b_1(x) \neq 0, b_2 = 0$ تعداد خطاها زوج است.

$b_1(x) \neq 0, b_2 = 1$ تعداد خطاها فرد است و $r(x)$ در ریاضی متعلق به C_1 نیست. ← تصحیح خطای تک بیتی

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: $(n,k) = (2^m-1,\ 2^m-1-m) \rightarrow (2^m,\ 2^m-m-1)$ → یک بیت پریتی اضافه کرده‌ایم.

$(n-k) = m+1$

$\rightarrow (2^m-1,\ 2^m-m-2)$ → یک بیت دیگر هم کرده‌ایم.

$(n-k) = m+1$

یک بیت به پریتی اضافه کردیم.

$$H_{new} = \begin{bmatrix} & H & & 0 \\ & & & 0 \\ & & & 0 \\ & & & \vdots \\ 1 & 1 & 1 & \cdots & 1 \end{bmatrix}$$

در فصل 3 مساوی 4

در این حالت d_{min} از 3 به 4 تبدیل می‌شود.

Extended Hamming code:

$$g(x) = (1+x)\,p(x)$$

فرض کنید مولد که برابر $g(x)$ باشد.

یک چندجمله‌ای از درجه‌ی m و primitive است؛ درجه‌ی $g(x)$ برابر $m+1$ می‌شود.

C_1: Hm $C_1(2^m-1,\ 2^m-m-1)$ if $v_1 \in C_1 \Rightarrow p(x) \mid v_1(x)$

C_2: EHm $C_2(2^m-1,\ 2^m-m-2)$ if $v_2 \in C_2 \Rightarrow g(x) \mid v_2(x)$

می‌دانیم $p(x) \mid g(x)$. بنابراین می‌توان گفت تمام کدهای متعلق C_2 به C_1 نیز تعلق دارند، به عبارت دیگر $C_2 \subset C_1$ می‌باشد.

$ord(C_2) = 2^{(2^m-m-2)} = \frac{1}{2} ord(C_1)$

ERROR-TRAPPING DECODING

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

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

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

$v(x) = q(x)(1+x)p(x)$

$v(1) = 0$

کدهای C_2 همگی دارای فرد زوج می‌باشند.

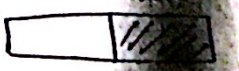
بنابراین می‌توان نتیجه گرفت که حداقل وزن کلمات کد C_2، بزرگتر یا مساوی 4 است. $d_{min} \geq 4$

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

هدف

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

$e(x)$



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:

$e^{(n-k)}(x)$

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

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

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

خطا داریم — با فرض خطا در اینجا

$$X^k \mathbf{s}^{(n-k)}(X) = \mathbf{e}(X)$$

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

در نهایت: e بدست آمده را XOR می‌کنیم

Subject : ( ) Date :

$$x^i = q_i(x)p(x) + b_i(x) \quad ; i = 0, \dots, 2^m - 2$$

تمامی $b_i(x)$ متفاوت می‌باشند.

$$x^i + x^j + x^k = q(x)p(x) + b(x)$$

می‌توان i، j و k ای یافت که $b(x)$ مخالف با صفر باشد.

$$x^i(1 + x^{j-i} + x^{k-i}) = q(x)p(x) \qquad b(x) \neq 0$$

$$x^l = q_l(x)p(x) + b(x)$$

اگر x^l به گونه‌ای انتخاب شود که باقیمانده‌ی آن بر $p(x)$ دقیقاً برابر $b(x)$ گردد.

$$x^i + x^j + x^k + x^l \leq (q(x) + q_l(x))p(x) \quad ; \quad \text{if } x^l = x^k \Rightarrow x^i(1 + x^{j-i}) = 0 \mathbb{F}(x)$$

از طرف دیگر $x^i + x^j + x^k + x^l$ بر $1+x$ قابل تقسیم است.

$$\Rightarrow x^i + x^j + x^k + x^l = Q(x)(1+x)p(x) = Q(x)g(x)$$

بنابراین $g(x) \mid x^i + x^j + x^k + x^l$

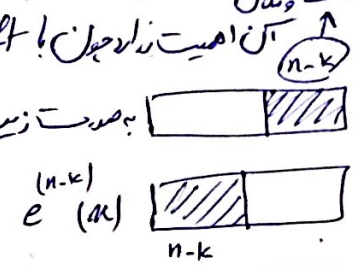
$x^i + x^j + x^k + x^l$ معادل یک کلمه‌ی کد معتبر در C_2 باشد، بنابراین d_{min} دقیقاً برابر 4 است.

Arman

مکمل می‌شیل

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

مهم وجود خطا در فاصله‌ی n-k است و مکان آن اهمیت ندارد چون با shift می‌توانیم به صورت زیر تبدیل کنیم.



Comments:

$e(x) = x^j B(x)$

حداکثر دارای درجه‌ی n-k-1 می‌باشد.

* اگر خطا محدود به n-k بیت بالای بردار کد باشد اما محدود به n-k بیت متوالی باشد، یعنی محدود به $x^i, x^{i+1}, \dots, x^{i+n-k-1}$ در انتهای نباشد. enhance

شامل خطاهای

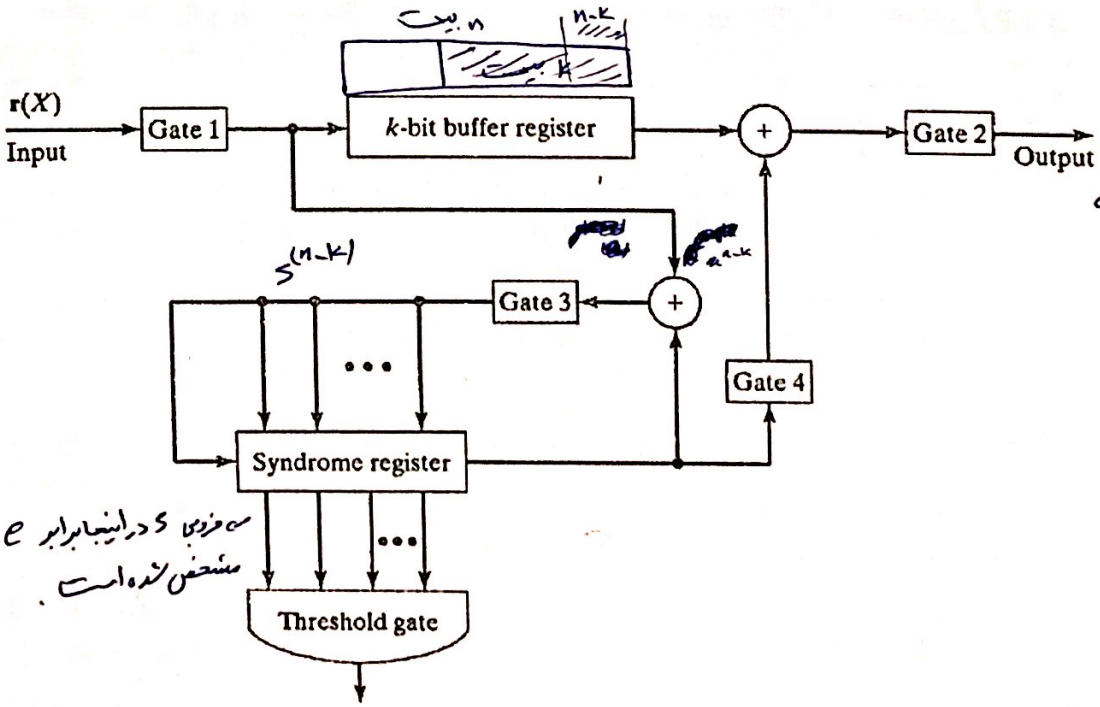
مهم + نکته $\approx 25\,min$

* فرض کنید $e(x) = x^j B(x)$، دارای حداکثر وزن t باشد. (t قدرت تصحیح خطای کد است.)

$$e(x) = x^j B(x) = a(x)g(x) + s(x) \Rightarrow x^j B(x) + s(x) = a(x)g(x) \Rightarrow g(x) \mid x^j B(x) + s(x)$$

کلمه کد معتبر خواهد شد.

اگر وزن $s(x)$ حداکثر برابر t، $s(x) \neq x^j B(x)$ باشد، بنابراین وزن $x^j B(x) + s(x)$ حداکثر دارای وزن 2t می‌باشد، که خلاف فرض کد است $2t+1 \le d_{min} \le 2t+2$ است



$\approx 30\,min$ عملیات decoding

مقدار s در اینجا برابر e در n-k بیت مشخص شده است.

FIGURE 5.14: Error-trapping decoder.

Step 1. The received polynomial $\mathbf{r}(X)$ 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 $\mathbf{r}(X)$ 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.

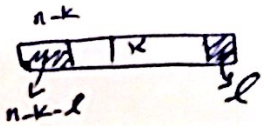
a. If the weight of the syndrome is t or less, the syndrome digits in the syndrome register are identical to the error digits at the $n-k$ high-order positions $X^k, X^{k+1}, \cdots, X^{n-1}$ of $\mathbf{r}(X)$. Now, gates 2 and 4 are turned on and the other gates are turned off. The received vector 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 $\mathbf{r}(X)$, 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 $\mathbf{r}(X)$, 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 $1 \le i \le k$, 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 $\mathbf{r}(X)$, 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 $1 \le l \le n-k$. 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 $\mathbf{r}(X)$. 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 $\mathbf{r}(X)$. 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 $\mathbf{r}(X)$ 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:

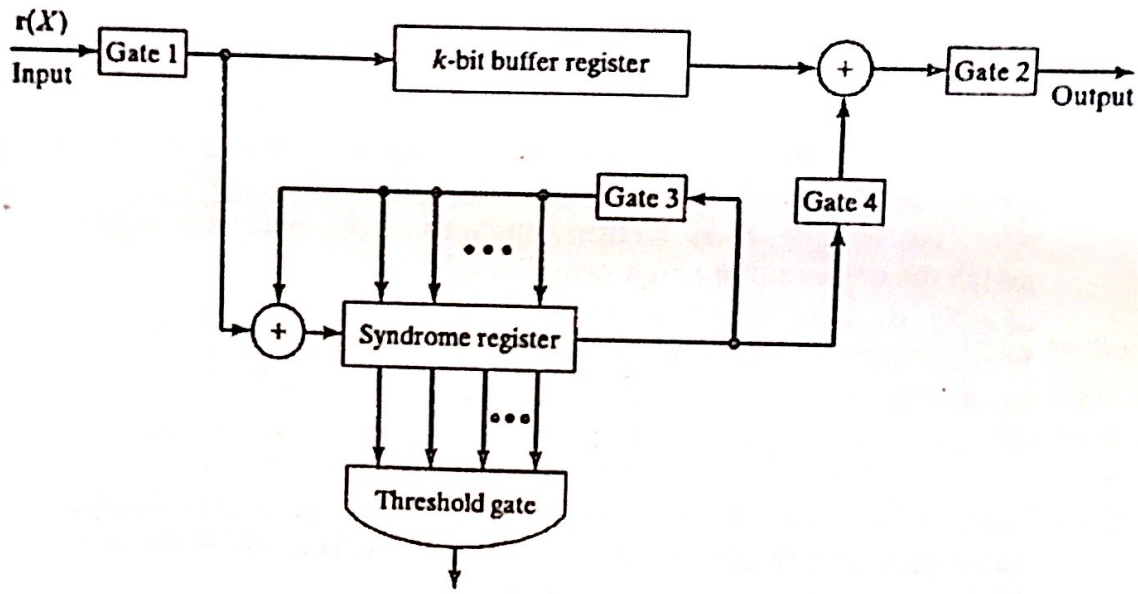


FIGURE 5.15: Another error-trapping decoder.

Step 1. Gates 1 and 3 are turned on and the other gates are turned off. The received vector $\mathbf{r}(X)$ 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 $\mathbf{r}(X)$ has been shifted into the syndrome register, the contents of the register form the syndrome $\mathbf{s}(X)$ of $\mathbf{r}(X)$.

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 $\mathbf{r}(X)$. 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.

other gates turned off. Go to step 3.

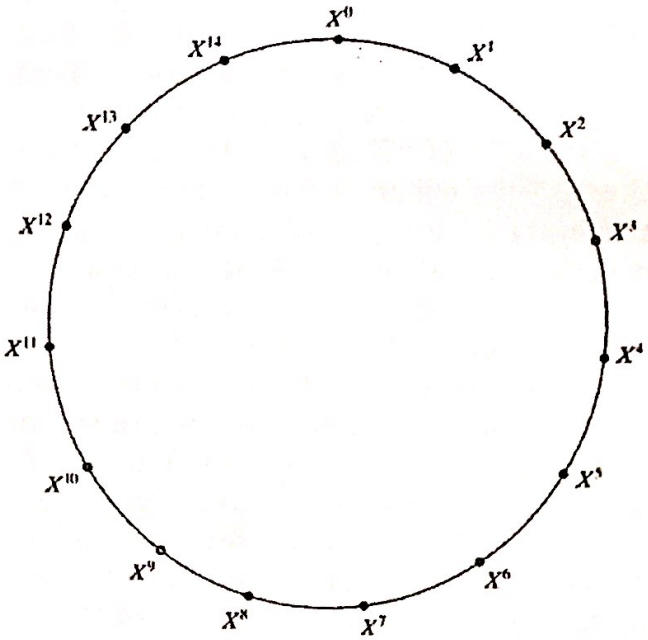
Step 3. The weight of the new contents in the syndrome register is tested. (a) If the weight is t or less, the errors are confined to the position $X^{n-1}, X^0, X^1, \cdots, X^{n-k-2}$ of $\mathbb{r}(X)$ (end-around case). The leftmost digit in the syndrome register is identical to the error at the position X^{n-1} of $\mathbb{r}(X)$; 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 $\mathbb{r}(X)$. 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 $\mathbb{r}(X)$. 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 $1 \le i \le n-k$, 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 $g(X) = 1 + X^4 + X^6 + X^7 + X^8$. 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.



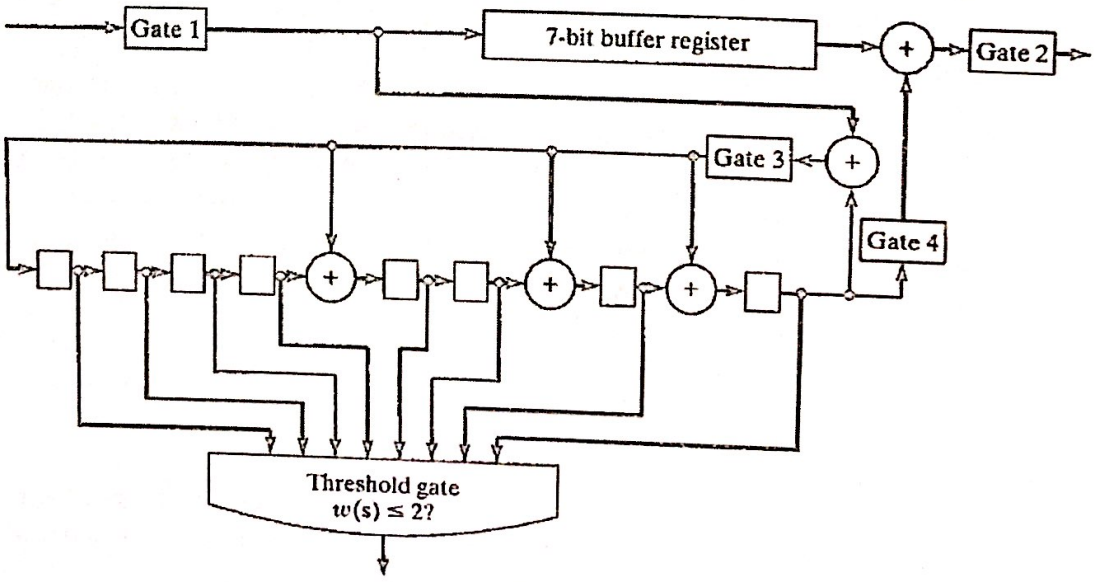


FIGURE 5.17: Error-trapping decoder for the (15, 7) cyclic code generated by $g(X) = 1 + X^4 + X^6 + X^7 + X^8$.

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 $X^{23} + 1 = (1 + X)g_1(X)g_2(X)$.

$n-k=11$

$d_{min}=7 \rightarrow t=3$

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.

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.

$V(x) = V_0 + V_1 x + \cdots + V_{n-1} x^{n-1}$

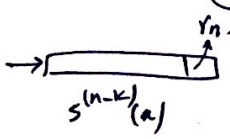
$\Rightarrow r(x) = V_{new} + K(x)$

$\xrightarrow{\text{shifted}} V_{new} = V_0 + V_1 x + \cdots + V_{n-l-1} x^{n-l-1} + 0 + 0 + \cdots + 0 x^{n-1}$

V_{n-1-l}

$V_{new}(x) = u(x) \cdot g(x)$

$k-1-l$ بیت



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 $\rho(X) = \rho_0 + \rho_1 X + \cdots + \rho_{n-k-1} X^{n-k-1}$ be the remainder resulting from this division. Then, we have the following relation:

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

$x^{n-k+l} = a_2(x) g(x) + \rho(x)$

$\Rightarrow \rho(x) r(x) = x^{n-k+l} r(x) + a_2(x) g(x) r(x)$

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

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

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

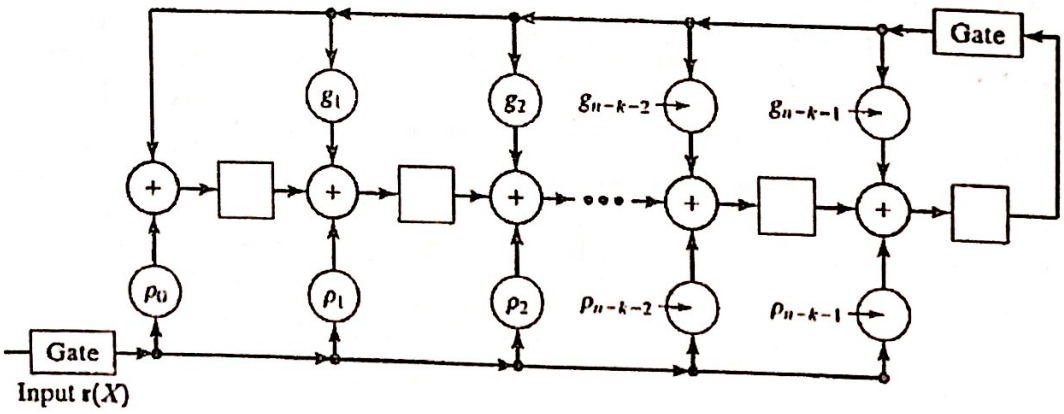


FIGURE 5.19: Circuit for multiplying $\mathbf{r}(X)$ by $\rho(X) = \rho_0 + \rho_1 X + \cdots + \rho_{n-k-1}X^{n-k-1}$ and dividing $\rho(X)\mathbf{r}(X)$ by $\mathbf{g}(X) = 1 + g_1 X + \cdots + X^{n-k}$.

$\frac{r(x)\,\rho(x)}{g(x)}$

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

$n-k=5$

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

$x^8 \,|\, g(x)$

$\rho(x) = x^3 + x^2 + 1$

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

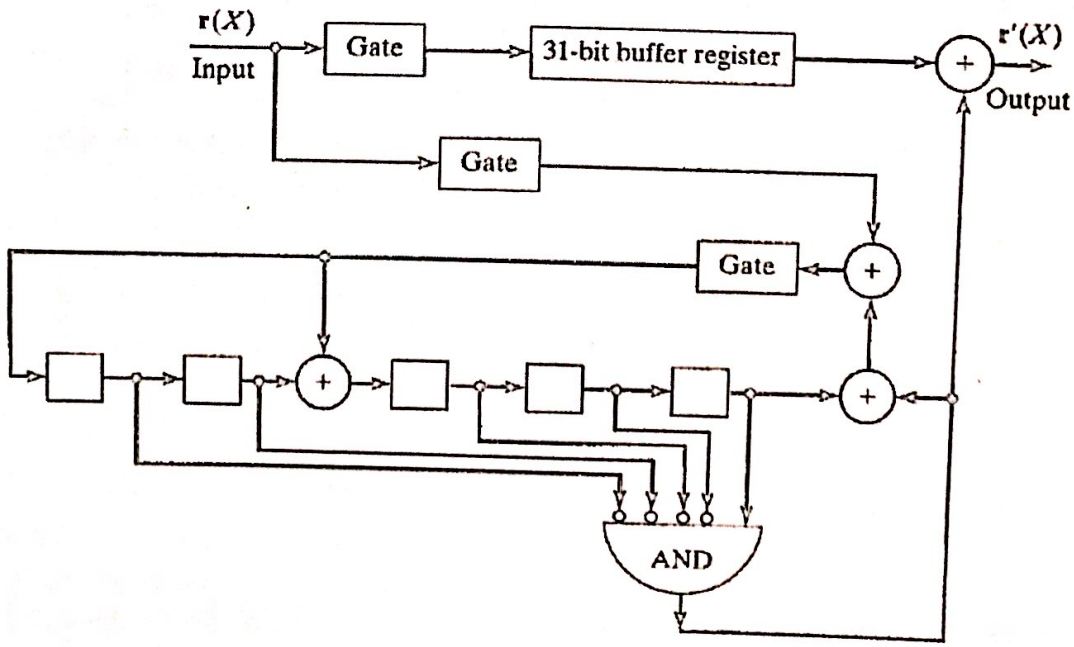


FIGURE 5.20: Decoding circuit for the (31, 26) cyclic Hamming code generated by $g(X) = 1 + X^2 + X^5$.

V75

$(n,k) \to (n-l, k-l)$

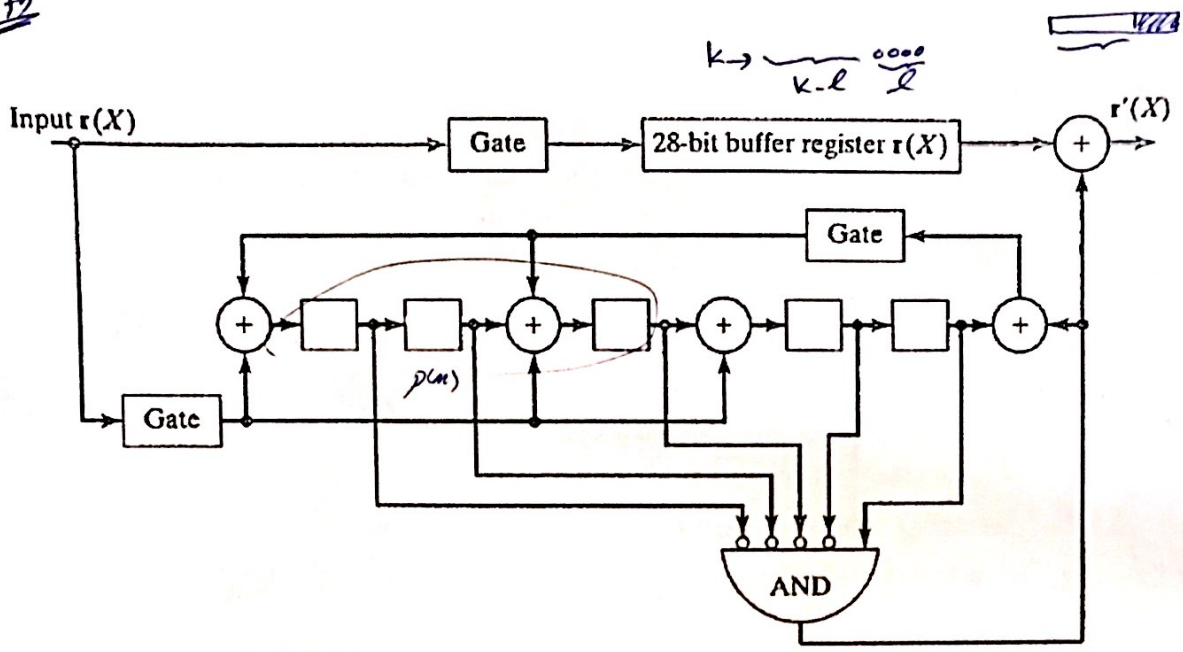
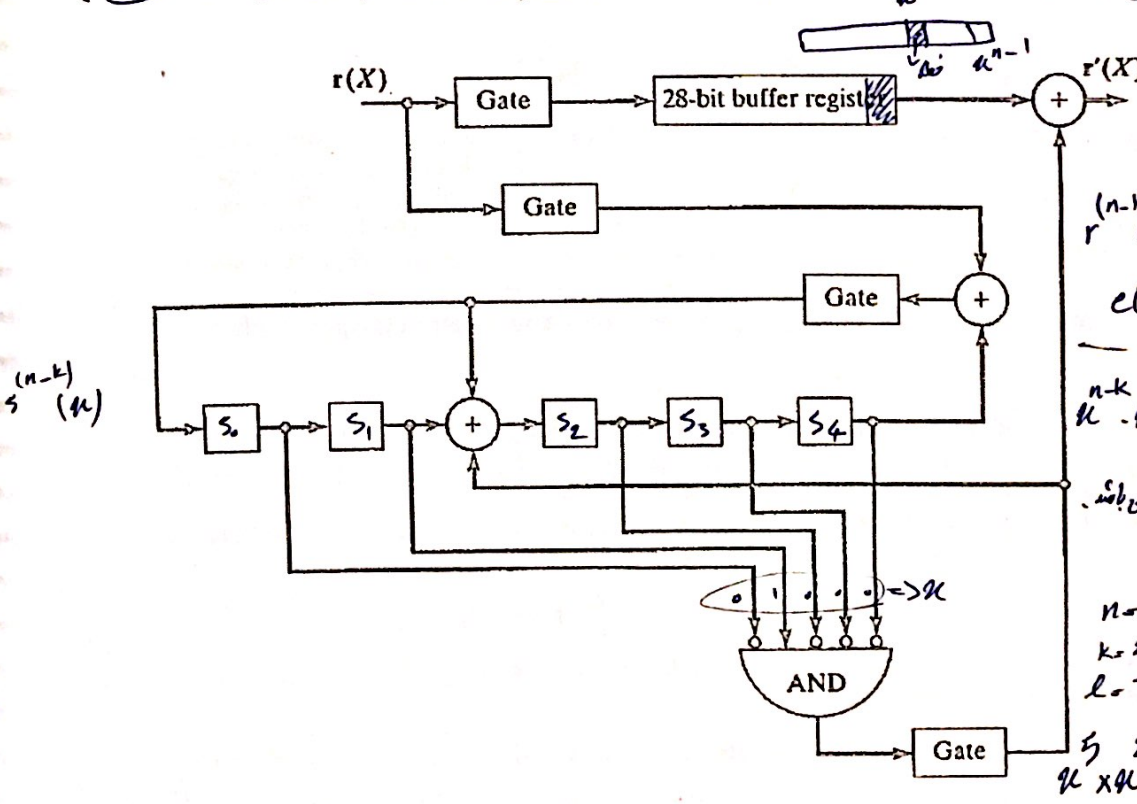


FIGURE 5.21: Decoding circuit for the (28, 23) shortened cyclic code generated by $g(X) = 1 + X^2 + X^5$.

اگر خطا در بیت آخر رجیستر قرار گرفته باشد $e(x) = x^{n-l-1}$ می‌باشد، چون $r(x)$ از سمت راست وارد شده است.



$r(x) = v(x) + e(x)$

$r(x) \bmod g(x) = e(x)$

$r^{(n-k)}(x) \bmod g(x) = e^{(n-k)}(x)$

$e(x) = x^{n-l-1}$

$x^{n-k} \cdot x^{n-l-1} \bmod g(x) = p(x)$

درجه‌ی $p(x)$ حداکثر $n-k-1$ می‌باشد.

$n = 31 \Rightarrow n-k = 5$

$k = 26$

$l = 3$

$x^5 \times x^{27} = x^{32} \bmod 1+x^2+x^5 = x$

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

توضیح 7^{min} اسم

$x^{31} = 1$

$(x^{31}+1) = g(x)(1+x^2+x^5)$

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 $\mathbf{p}(X)$ or a polynomial $\mathbf{g}(X) = (X+1)\mathbf{p}(X)$. A number of CRC codes have become international standards for error detection in various contexts. A few standard CRC codes follow:

(A) cyclic برای ساخت

$\deg(p(x)) = m$

$p(x) \mid x^{2^m-1}+1 = q(x)\,p(x)$

$n = 2^m - 1$ طول کلمه

$x^n+1 = q(x)\,g(x)$

$(2^m-1,\ 2^m-m-1)$

یا

اگر ترم زوج تا کنیم

$(2^m-1-l,\ 2^m-m-1-l)$

(B)

طول پیامی که می‌خواهیم CRC را روی آن اعمال کنیم حداکثر برابر 2^m-1-m خواهد بود.

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

$$\mathbf{g}(X) = (X+1)(X^{15}+X^{14}+X^{13}+X^{12}+X^4+X^3+X^2+X+1)$$
$$= X^{16}+X^{12}+X^5+1.$$

ANSI (American National Standards Institute)

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

IBM-SDLC (IBM Synchronous Data Link Control)

$$\mathbf{g}(X) = (X+1)^2(X^{14}+X^{13}+X^{12}+X^{10}+X^8+X^6+X^5+X^4+X^3+X+1)$$
$$= X^{16}+X^{15}+X^{13}+X^7+X^4+X^2+X+1.$$

IEC TC57

$$\mathbf{g}(X) = (X+1)^2(X^{14}+X^{10}+X^9+X^8+X^5+X^3+X^2+X+1)$$
$$= X^{16}+X^{14}+X^{11}+X^8+X^6+X^5+X^4+1.$$

IEEE Standard 802.3

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

$c(n_1, k_1)$

$c(n_2, k_2)$ $\rightarrow c_{new}(n_1 n_2, k_1 k_2)$

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.

شرط برای cyclic

$\gcd(n_1, n_2) = 1$

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

$$an_1 + bn_2 = 1.$$

$\gcd(\alpha, \beta) = d$

$a\alpha + b\beta = d$

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,

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

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,

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

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.

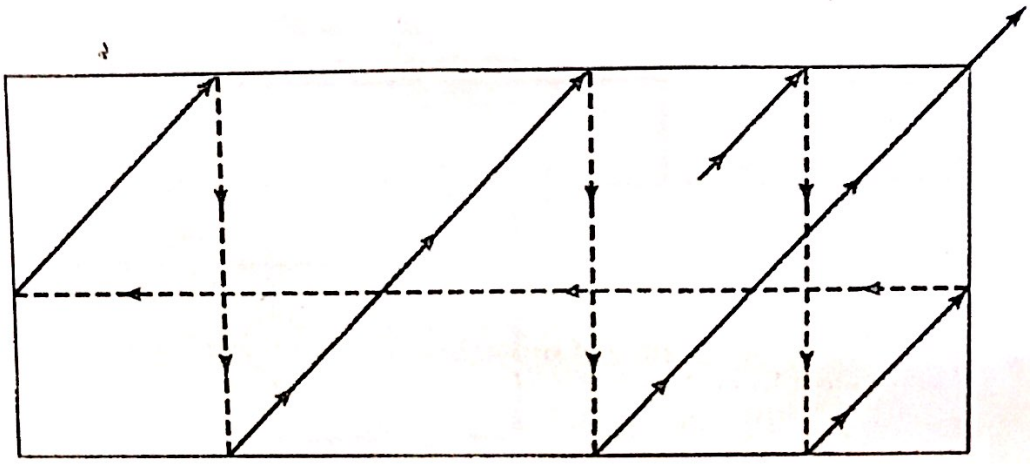


FIGURE 5.23: Transmission of a cyclic product code.

شبه سیکلیک

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:

$$\mathbb{G} = \begin{bmatrix} 111 & 100 & 110 \\ 110 & 111 & 100 \\ 100 & 110 & 111 \end{bmatrix}_{3\times 9}.$$

در کدهای کروتنی: هر سینت کروتنی دلخواه روی یک کد خود یک کدی که معتبر است ($n_0=1$)

$n_0 = 3$

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

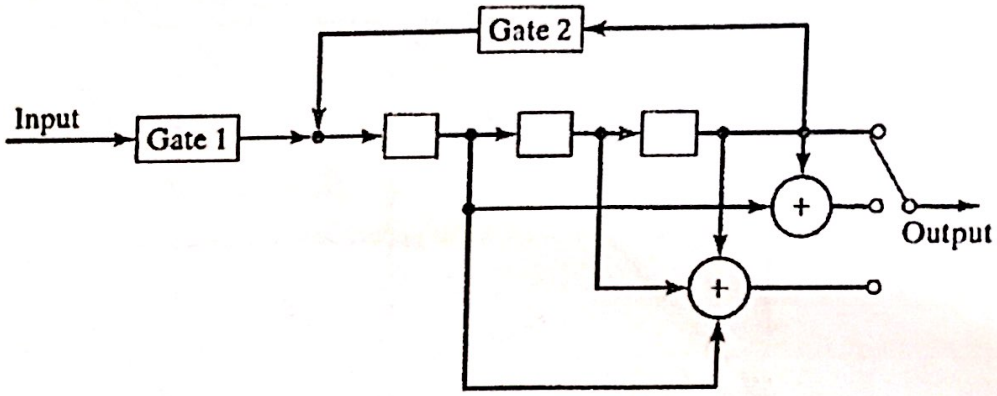


FIGURE 5.24: An encoding circuit for a (9, 3) quasi-cyclic code.

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 $0 \le j < m$, let $\mathbb{M}_j \triangleq [\mathbb{G}_j, \mathbb{G}_{j-1}, \cdots, \mathbb{G}_{j+1}]^T$ 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 $\mathbf{Q}_l$ be the $mk_0 \times m$ submatrix that is formed by taking the lth columns from $\mathbf{M}_0, \mathbf{M}_1, \cdots, \mathbf{M}_{m-1}$. Then, we can put $\mathbf{G}$ in the following form:

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

Each column of $\mathbf{Q}_l$ 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, $\mathbf{Q}_l$ 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 $\mathbf{Q}_l$ is called a *circulant*. Therefore, $\mathbf{G}_c$ 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:

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

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

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

MATLAB

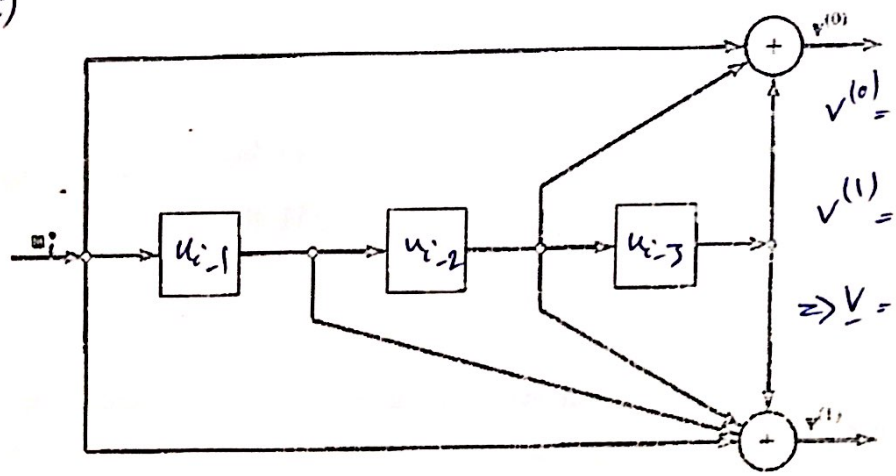
1. دو چندجمله‌ای دلخواه روی $GF(2)$ را باهم جمع کنید.
2. 2 چندجمله‌ای دلخواه روی $GF(2)$ را در هم ضرب کنید.
3. چندجمله‌ای $a(x)$ با درجه دلخواه را بر چندجمله‌ای $b(x)$ با درجه‌ی دلخواه تقسیم کنید.

~36min
V75

Convolutional Codes

$u_i \rightarrow$ coder $\rightarrow v(u_i)$

$v(u_i, u_{i-1}, \ldots, u_{i-m})$



$v^{(0)} = u_i + u_{i-2} + u_{i-3}$

$v^{(1)} = u_i + u_{i-1} + u_{i-2} + u_{i-3}$

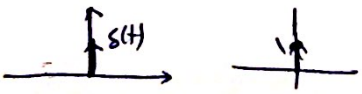
$\Rightarrow \underline{v} = (v^{(0)}, v^{(1)})$

FIGURE 11.1: A rate $R = 1/2$ binary nonsystematic feedforward convolutional encoder with memory order $m = 3$.

$\delta(t) \rightarrow S \rightarrow h(t)$

$v^{(0)} \rightarrow$ 1011

$v^{(1)} \rightarrow$ 1111



The *information sequence* $\mathbf{u} = (u_0, u_1, u_2, \cdots)$ enters the encoder one bit at a time. Because the encoder is a linear system, the two encoder *output sequences* $\mathbf{v}^{(0)} = (v_0^{(0)}, v_1^{(0)}, v_2^{(0)}, \cdots)$ and $\mathbf{v}^{(1)} = (v_0^{(1)}, v_1^{(1)}, v_2^{(1)}, \cdots)$ can be obtained as the convolution of the input sequence $\mathbf{u}$ with the two encoder *impulse responses*. The impulse responses are obtained by letting $\mathbf{u} = (1\ 0\ 0 \cdots)$ and observing the two output sequences. For an encoder with memory order m, the impulse responses can last at most $m + 1$ time units and are written as $\mathbf{g}^{(0)} = (g_0^{(0)}, g_1^{(0)}, \cdots, g_m^{(0)})$ and $\mathbf{g}^{(1)} = (g_0^{(1)}, g_1^{(1)}, \cdots, g_m^{(1)})$. For the encoder of Figure 11.1,

$$\mathbf{g}^{(0)} = (1\ 0\ 1\ 1) \tag{11.1a}$$

$$\mathbf{g}^{(1)} = (1\ 1\ 1\ 1). \tag{11.1b}$$

The impulse responses $\mathbf{g}^{(0)}$ and $\mathbf{g}^{(1)}$ are called the *generator sequences* of the encoder.

We can now write the *encoding equations* as

$$\mathbf{v}^{(0)} = \mathbf{u} \circledast \mathbf{g}^{(0)}, \tag{11.2a}$$

$$\mathbf{v}^{(1)} = \mathbf{u} \circledast \mathbf{g}^{(1)}, \tag{11.2b}$$

$$v_l^{(j)} = \sum_{i=0}^{m} u_{l-i} g_i^{(j)} = u_l g_0^{(j)} + u_{l-1} g_1^{(j)} + \cdots + u_{l-m} g_m^{(j)}, \quad j = 0, 1, \tag{11.3}$$

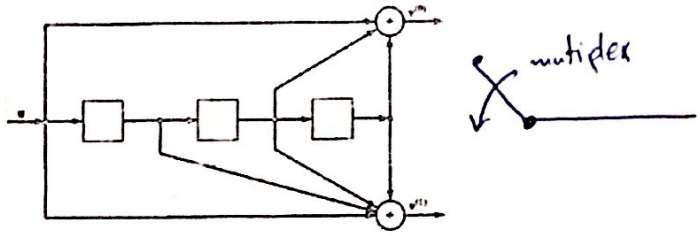


FIGURE 11.1: A rate $R = 1/2$ binary nonsystematic feedforward convolutional encoder with memory order $m = 3$.

$$v_l^{(0)} = u_l \qquad + u_{l-2} + u_{l-3} \tag{11.4a}$$

$$v_l^{(1)} = u_l + u_{l-1} + u_{l-2} + u_{l-3}, \tag{11.4b}$$

$$\mathbf{v} = (v_0^{(0)} v_0^{(1)}, v_1^{(0)} v_1^{(1)}, v_2^{(0)} v_2^{(1)}, \cdots).$$

For example, let the information sequence $\mathbf{u} = (1\ 0\ 1\ 1\ 1)$. Then, the output sequences are $R = \frac{1}{2}$

$$\mathbf{v}^{(0)} = (1\,0\,1\,1\,1) \circledast (1\,0\,1\,1) = (1\,0\,0\,0\,0\,0\,0\,1), \tag{11.6a}$$

$$\mathbf{v}^{(1)} = (1\,0\,1\,1\,1) \circledast (1\,1\,1\,1) = (1\,1\,0\,1\,1\,1\,0\,1), \tag{11.6b}$$

and the codeword is

$$\mathbf{v} = (1\,1, 0\,1, 0\,0, 0\,1, 0\,1, 0\,1, 0\,0, 1\,1). \tag{11.7}$$

If we interlace the generator sequences $\mathbf{g}^{(0)}$ and $\mathbf{g}^{(1)}$ and then arrange them in the matrix

$$\mathbf{G} = \begin{bmatrix} g_0^{(0)} g_0^{(1)} & g_1^{(0)} g_1^{(1)} & g_2^{(0)} g_2^{(1)} & \cdots & g_m^{(0)} & g_m^{(1)} & & & \\ & g_0^{(0)} g_0^{(1)} & g_1^{(0)} g_1^{(1)} & \cdots & g_{m-1}^{(0)} g_{m-1}^{(1)} & & g_m^{(0)} & g_m^{(1)} & \\ & & g_0^{(0)} g_0^{(1)} & \cdots & g_{m-2}^{(0)} g_{m-2}^{(1)} & & g_{m-1}^{(0)} g_{m-1}^{(1)} & & g_m^{(0)} g_m^{(1)} \\ & & & \ddots & & & & & \ddots \end{bmatrix}, \tag{11.8}$$

$R = \frac{1}{2}$

where the blank areas are all zeros, we can rewrite the encoding equations in matrix form as

$$\mathbf{v} = \mathbf{u}\mathbf{G}. \tag{11.9}$$

ورودی 100 بیت ⇒ خروجی $= 200 + 2 \times \underset{m}{3} = 206$

For the encoder of Figure 11.1, if $\mathbf{u} = (1\ 0\ 1\ 1\ 1)$, then

$$\mathbf{v} = \mathbf{u}\mathbf{G}$$

$$= \overbrace{(1\ 0\ 1\ 1\ 1)}^{5\text{ بیت}} \underbrace{\begin{bmatrix} 11 & 01 & 11 & 11 & & & & \\ & 11 & 01 & 11 & 11 & & & \\ & & 11 & 01 & 11 & 11 & & \\ & & & 11 & 01 & 11 & 11 & \\ & & & & 11 & 01 & 11 & 11 \end{bmatrix}}_{\text{code sequence}} \tag{11.10}$$

$$= (1\,1, 0\,1, 0\,0, 0\,1, 0\,1, 0\,1, 0\,0, 1\,1). \qquad (5\times 2) + (3\times 2)$$

which agrees with our previous calculation using discrete convolution.

EXAMPLE 11.2 A Rate $R = 2/3$ Nonsystematic Feedforward Convolutional Encoder

As a second example of a nonsystematic feedforward convolutional encoder, consider the binary rate $R = 2/3$ encoder with memory order $m = 1$ shown in Figure 11.2. The encoder consists of $k = 2$ shift registers, each with $m = 1$

$u_i + u_{i-1} + u_{i-2}$

$R = \frac{2}{3}$ (k, n)

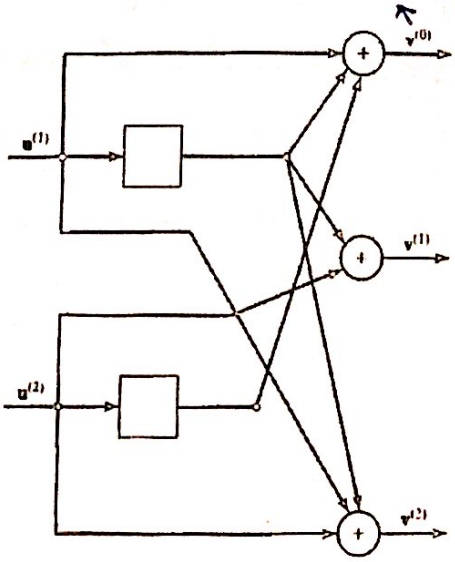


FIGURE 11.2: A rate $R = 2/3$ binary nonsystematic feedforward convolutional encoder with memory order $m = 1$.

$$\mathbf{g}_1^{(0)} = (1\ 1) \qquad \mathbf{g}_1^{(1)} = (0\ 1) \qquad \mathbf{g}_1^{(2)} = (1\ 1), \tag{11.11a}$$

$$\mathbf{g}_2^{(0)} = (0\ 1) \qquad \mathbf{g}_2^{(1)} = (1\ 0) \qquad \mathbf{g}_2^{(2)} = (1\ 0), \tag{11.11b}$$

$$\mathbf{G} = \begin{bmatrix} g_{1,0}^{(0)} g_{1,0}^{(1)} g_{1,0}^{(2)} & g_{1,1}^{(0)} g_{1,1}^{(1)} g_{1,1}^{(2)} & \cdots & g_{1,m}^{(0)} & g_{1,m}^{(1)} & g_{1,m}^{(2)} & \\ g_{2,0}^{(0)} g_{2,0}^{(1)} g_{2,0}^{(2)} & g_{2,1}^{(0)} g_{2,1}^{(1)} g_{2,1}^{(2)} & \cdots & g_{2,m}^{(0)} & g_{2,m}^{(1)} & g_{2,m}^{(2)} & \\ & g_{1,0}^{(0)} g_{1,0}^{(1)} g_{1,0}^{(2)} & \cdots & g_{1,m-1}^{(0)} & g_{1,m-1}^{(1)} & g_{1,m-1}^{(2)} & g_{1,m}^{(0)} g_{1,m}^{(1)} g_{1,m}^{(2)} \\ & g_{2,0}^{(0)} g_{2,0}^{(1)} g_{2,0}^{(2)} & \cdots & g_{2,m-1}^{(0)} & g_{2,m-1}^{(1)} & g_{2,m-1}^{(2)} & g_{2,m}^{(0)} g_{2,m}^{(1)} g_{2,m}^{(2)} \\ & \ddots & & & & & \ddots \end{bmatrix}.$$

$$\mathbf{v} = \mathbf{u}\mathbf{G} = (\underbrace{1\,1, 0\,1, 1\,0}_{\text{3 قالب}}) \begin{bmatrix} 101 & 111 & & \\ 011 & 100 & & \\ & 101 & 111 & \\ & 011 & 100 & \\ & & 101 & 111 \\ & & 011 & 100 \end{bmatrix} = (\underbrace{1\,1\,0, 0\,0\,0, 0\,0\,1, 1\,1\,1}_{\text{4 قالب، یکی بیشتر بودنش به دلیل حافظه جا طول کدایست کامل محاسبه}}). \tag{11.18}$$

*در حالت کلی اگر طول پیام برابر $k \times l$ باشد و مزدوج n بیتی باشد، چه صورتیکه طول بند کنترل شیفت رجیستر مورد استفاده برابر M باشد، در این صورت دنباله‌ی کد برابر $l \times n + M \times n$ بیت خواهد بود.

به عبارت دیگر نرخ کد، کدهای کانولوشن در حالت کلی برابر

$$R = \frac{k \times l}{l \times n + M \times n} = \frac{k}{n + \frac{M \times n}{l}} = \frac{k}{n}$$

که l به اندازه‌ی کافی بزرگ باشد.

Definition 11.1 Let ν_i be the length of the ith shift register in a convolutional encoder with k input sequences, $i = 1, 2, \cdots, k$.

For example, for the rate $R = 1/2$ encoder of Figure 11.1, $\nu = 3$, and for the rate $R = 2/3$ encoder of Figure 11.2, $\nu_1 = \nu_2 = 1$.

Definition 11.2 The encoder *memory order* m is defined as

$$m = \max_{1 \le i \le k} \nu_i; \tag{11.19}$$

that is, m is the maximum length of all k shift registers.

For the rate $R = 1/2$ encoder of Figure 11.1, $m = \nu_1 = 3$, and for the rate $R = 2/3$ encoder of Figure 11.2, $m = \max(\nu_1, \nu_2) = \max(1, 1) = 1$.

Definition 11.3 The *overall constraint length* ν of the encoder is defined as

$$\nu = \sum_{1 \le i \le k} \nu_i; \tag{11.20}$$

that is, ν is the sum of the lengths of all k shift registers.

For the rate $R = 1/2$ encoder of Figure 11.1, $\nu = \nu_1 = 3$, and for the rate $R = 2/3$ encoder of Figure 11.2, $\nu = \nu_1 + \nu_2 = 1 + 1 = 2$.

Definition 11.4 A rate $R = k/n$ convolutional encoder with overall constraint length ν is referred to as an (n, k, ν) encoder.

The rate $R = 1/2$ encoder of Figure 11.1 is a $(2, 1, 3)$, encoder and the rate $R = 2/3$ encoder of Figure 11.2 is a $(3, 2, 2)$ encoder.

Here we note that for an $(n, 1, \nu)$ encoder, the overall constraint length ν equals the memory order m; that is, $\nu = m$ when $k = 1$; however, for $k > 1$, in general, $m \leq \nu$, and in the case when each shift register has the same length, $\nu = km$.

EXAMPLE 11.3 A (4, 3, 3) Nonsystematic Feedforward Convolutional Encoder

A $(4, 3, 3)$ binary nonsystematic feedforward convolutional encoder in which the shift register lengths are $\nu_1 = 0$, $\nu_2 = 1$, and $\nu_3 = 2$ is shown in Figure 11.3. The memory order is $m = 2$, and the overall constraint length is $\nu = 3$. Following the notation of Example 11.2, the generator sequences are given by

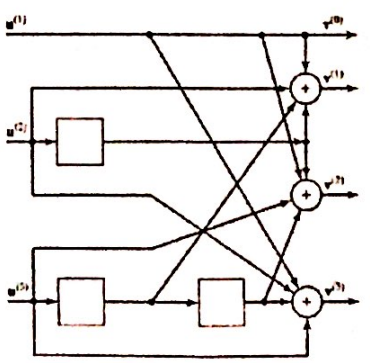
$$\mathbf{g}_1^{(0)} = (100) \quad \mathbf{g}_1^{(1)} = (100) \quad \mathbf{g}_1^{(2)} = (100) \quad \mathbf{g}_1^{(3)} = (100), \tag{11.21a}$$

$$\mathbf{g}_2^{(0)} = (000) \quad \mathbf{g}_2^{(1)} = (110) \quad \mathbf{g}_2^{(2)} = (010) \quad \mathbf{g}_2^{(3)} = (100), \tag{11.21b}$$

$$\mathbf{g}_3^{(0)} = (000) \quad \mathbf{g}_3^{(1)} = (010) \quad \mathbf{g}_3^{(2)} = (101) \quad \mathbf{g}_3^{(3)} = (101). \tag{11.21c}$$

≈1:05min

غیرسیستماتیک



(4,1,3)

n=4 ,V=3

FIGURE 11.3: A (4, 3, 3) binary nonsystematic feedforward convolutional encoder.

In the general case of an (n, k, ν) feedforward encoder with memory order m, the generator matrix is given by

$$\mathbf{G} = \begin{bmatrix} \mathbf{G}_0 & \mathbf{G}_1 & \mathbf{G}_2 & \cdots & \mathbf{G}_m & & \\ & \mathbf{G}_0 & \mathbf{G}_1 & \cdots & \mathbf{G}_{m-1} & \mathbf{G}_m & \\ & & \mathbf{G}_0 & \cdots & \mathbf{G}_{m-2} & \mathbf{G}_{m-1} & \mathbf{G}_m \\ & & & \ddots & & & \end{bmatrix}. \tag{11.22}$$

where each $\mathbf{G}_l$ is a $k \times n$ submatrix whose entries are

$$\mathbf{G}_l = \begin{bmatrix} g_{1,l}^{(0)} & g_{1,l}^{(1)} & \cdots & g_{1,l}^{(n-1)} \\ g_{2,l}^{(0)} & g_{2,l}^{(1)} & \cdots & g_{2,l}^{(n-1)} \\ \vdots & \vdots & & \vdots \\ g_{k,l}^{(0)} & g_{k,l}^{(1)} & \cdots & g_{k,l}^{(n-1)} \end{bmatrix}. \tag{11.23}$$

DEFINITION 11.5 An (n, k, ν) convolutional code is the set of all output sequences (codewords) produced by an (n, k, ν) convolutional encoder; that is, it is the row space of the encoder generator matrix $\mathbf{G}$.

$v^{(0)} = u_i + u_{i-2} + u_{i-3}$

$v^{(1)} = u_i + u_{i-1} + u_{i-2} + u_{i-3}$

$\longrightarrow$ $v^{(0)}(D) = u(D) + D^2 u(D) + D^3 u(D)$

$v^{(1)}(D) = u(D) + D u(D) + D^2 u(D) + D^3 u(D)$

$\Rightarrow \begin{cases} v^{(0)}(D) = u(D)\cdot(1 + D^2 + D^3) \\ v^{(1)}(D) = u(D)(1 + D + D^2 + D^3) \end{cases}$

In any linear system, time-domain operations involving convolution can be replaced by more convenient transform-domain operations involving polynomial multiplication. Because a convolutional encoder is a linear system, each sequence in the encoding equations can be replaced by a corresponding polynomial, and the convolution operation replaced by polynomial multiplication. In the polynomial representation of a binary sequence, the sequence itself is represented by the coefficients of the polynomial. For example, for a $(2, 1, \nu)$ encoder, the encoding equations become

$$\mathbf{v}^{(0)}(D) = \mathbf{u}(D)\mathbf{g}^{(0)}(D), \tag{11.24a}$$

$$\mathbf{v}^{(1)}(D) = \mathbf{u}(D)\mathbf{g}^{(1)}(D), \tag{11.24b}$$

V^{77}

where

$$\mathbf{u}(D) = u_0 + u_1 D + u_2 D^2 + \cdots \tag{11.25a}$$

$v(D) = u(D)G(D)$

$\underset{2}{uu}$ $\underset{1}{uu}$ $\underset{1\ 1\ 2}{uu}$

is the information sequence,

$$\mathbf{v}^{(0)}(D) = v_0^{(0)} + v_1^{(0)} D + v_2^{(0)} D^2 + \cdots \tag{11.25b}$$

and

$$\mathbf{v}^{(1)}(D) = v_0^{(1)} + v_1^{(1)} D + v_2^{(1)} D^2 + \cdots \tag{11.25c}$$

are the encoded sequences,

$$\mathbf{g}^{(0)}(D) = g_0^{(0)} + g_1^{(0)} D + \cdots + g_m^{(0)} D^m \tag{11.25d}$$

and

$$\mathbf{g}^{(1)}(D) = g_0^{(1)} + g_1^{(1)} D + \cdots + g_m^{(1)} D^m \tag{11.25e}$$

are the *generator polynomials* of the code, and all operations are modulo-2. We can then write the codeword as

$$\mathbf{V}(D) = [\mathbf{v}^0(D), \mathbf{v}^1(D)] \tag{11.26a}$$

or, after multiplexing, as

برای نوشتن دنباله خروجی $\Rightarrow$

$$\mathbf{v}(D) = \mathbf{v}^{(0)}(D^2) + D\mathbf{v}^{(1)}(D^2). \tag{11.26b}$$

The indeterminate D can be interpreted as a delay operator, where the power of D denotes the number of time units a bit is delayed with respect to the initial bit in the sequence.

10111

EXAMPLE 11.1 (Continued) 1101000101001 1

For the (2, 1, 3) encoder of Figure 11.1, the generator polynomials are $\mathbf{g}^{(0)}(D) = 1 + D^2 + D^3$ and $\mathbf{g}^{(1)}(D) = 1 + D + D^2 + D^3$. For the information sequence $\mathbf{u}(D) = 1 + D^2 + D^3 + D^4$, the encoding equations are

$$\mathbf{v}^{(0)}(D) = (1 + D^2 + D^3 + D^4)(1 + D^2 + D^3) = 1 + D^7, \tag{11.27a}$$

10000001

$$\mathbf{v}^{(1)}(D) = (1 + D^2 + D^3 + D^4)(1 + D + D^2 + D^3) = 1 + D + D^3 + D^4 + D^5 + D^7, \tag{11.27b}$$

and we can write the codeword as

$$\mathbf{V}(D) = [1 + D^7, 1 + D + D^3 + D^4 + D^5 + D^7], \tag{11.28a}$$

or

11 01000101001 1

$$\mathbf{v}(D) = 1 + D + D^3 + D^7 + D^9 + D^{11} + D^{14} + D^{15}. \tag{11.28b}$$

Note that in each case the result is the same as previously computed using convolution and matrix multiplication.

We can rewrite Eqs. (11.31), (11.32), and (11.33) to provide a means of representing the multiplexed codeword $\mathbf{v}(D)$ directly in terms of the input sequences. A little algebraic manipulation yields

$$\mathbf{v}(D) = \sum_{i=1}^{k} \mathbf{u}^{(i)}(D^n)\mathbf{g}_i(D), \qquad (11.36)$$

where

V77

$$\mathbf{g}_i(D) \triangleq \mathbf{g}_i^{(0)}(D^n) + D\mathbf{g}_i^{(1)}(D^n) + \cdots + D^{n-1}\mathbf{g}_i^{(n-1)}(D^n), \quad 1 \leq i \leq k, \qquad (11.37)$$

is a *composite generator polynomial* relating the ith input sequence to $\mathbf{v}(D)$.

EXAMPLE 11.1 (Continued)

$k \longrightarrow n$

For the (2, 1, 3) encoder of Figure 11.1, the composite generator polynomial is

$$\mathbf{g}(D) = \mathbf{g}^{(0)}(D^2) + D\mathbf{g}^{(1)}(D^2) = 1 + D + D^3 + D^4 + D^5 + D^6 + D^7, \qquad (11.38)$$

and for $\mathbf{u}(D) = 1 + D^2 + D^3 + D^4$, the codeword is

$$\begin{aligned}\mathbf{v}(D) &= \mathbf{u}(D^2)\mathbf{g}(D) = (1 + D^4 + D^6 + D^8)(1 + D + D^3 + D^4 + D^5 + D^6 + D^7)\\ &= 1 + D + D^3 + D^7 + D^9 + D^{11} + D^{14} + D^{15},\end{aligned}$$

11001010100101 (11.39)

which again agrees with previous calculations.

(n, k, ν)

k 1 k

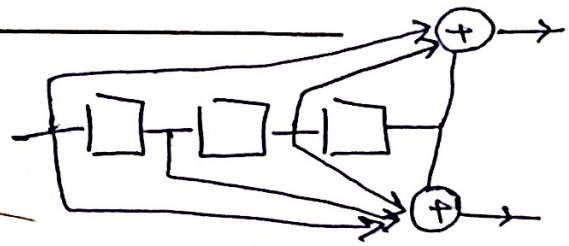
n-k

مدل سیستماتیک

شیفت با شماره بالا

$1+D+D^2+D^3$

$1+D^2+D^3$



An important subclass of convolutional encoders is the class of *systematic encoders*. In a systematic encoder, k output sequences, called *systematic output sequences*, are exact replicas of the k input sequences. Although, in general, the k systematic output sequences can be any set of k output sequences, we will follow the convention that the *first* k output sequences are the systematic sequences; that is,

$$\mathbf{v}^{(i-1)} = \mathbf{u}^{(i)}, \quad i = 1, 2, \cdots, k, \qquad (11.40)$$

and the generator sequences satisfy

$$\mathbf{g}_i^{(j)} = \begin{cases} 1 \text{ if } j = i - 1 \\ 0 \text{ if } j \neq i - 1 \end{cases} \qquad i = 1, 2, \cdots, k. \qquad (11.41)$$

In the case of *systematic feedforward encoders,* the (time-domain) generator matrix is given by

$$\mathbf{G} = \begin{bmatrix} \mathbf{IP}_0 & \mathbf{0P}_1 & \mathbf{0P}_2 & \cdots & \mathbf{0P}_m & & & \\ & \mathbf{IP}_0 & \mathbf{0P}_1 & \cdots & \mathbf{0P}_{m-1} & \mathbf{0P}_m & & \\ & & \mathbf{IP}_0 & \cdots & \mathbf{0P}_{m-2} & \mathbf{0P}_{m-1} & \mathbf{0P}_m & \\ & & & \ddots & & & \ddots & \end{bmatrix}, \tag{11.42}$$

where $\mathbb{I}$ is the $k \times k$ identity matrix, $\mathbb{0}$ is the $k \times k$ all-zero matrix, and $\mathbb{P}_l$ is the $k \times (n-k)$ matrix whose entries are

$$\mathbb{P}_l = \begin{bmatrix} g_{1,l}^{(k)} & g_{1,l}^{(k+1)} & \cdots & g_{1,l}^{(n-1)} \\ g_{2,l}^{k} & g_{2,l}^{(k+1)} & \cdots & g_{2,l}^{(n-1)} \\ \vdots & \vdots & & \vdots \\ g_{k,l}^{(k)} & g_{k,l}^{(k+1)} & \cdots & g_{k,l}^{(n-1)} \end{bmatrix}. \tag{11.43}$$

$$\mathbb{H} = \begin{bmatrix} \mathbb{P}_0^T & \mathbb{I} & & & & & & & & & & \\ \mathbb{P}_1^T & \mathbb{0} & \mathbb{P}_0^T & \mathbb{I} & & & & & & & & \\ \mathbb{P}_2^T & \mathbb{0} & \mathbb{P}_1^T & \mathbb{0} & \mathbb{P}_0^T & \mathbb{I} & & & & & & \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & & & & & \\ \mathbb{P}_m^T & \mathbb{0} & \mathbb{P}_{m-1}^T & \mathbb{0} & \mathbb{P}_{m-2}^T & \mathbb{0} & \cdots & \mathbb{P}_0^T & \mathbb{I} & & & \\ & & \mathbb{P}_m^T & \mathbb{0} & \mathbb{P}_{m-1}^T & \mathbb{0} & \cdots & \mathbb{P}_1^T & \mathbb{0} & \mathbb{P}_0^T & \mathbb{I} & \\ & & & & \mathbb{P}_m^T & \mathbb{0} & \cdots & \mathbb{P}_2^T & \mathbb{0} & \mathbb{P}_1^T & \mathbb{0} & \mathbb{P}_0^T & \mathbb{I} \\ & & & & & & & & & & & & \ddots \end{bmatrix},$$

where $\mathbb{I}$ is the $k \times k$ identity matrix, $\mathbb{0}$ is the $k \times k$ all-zero matrix, and $\mathbb{P}_l$ is the $k \times (n-k)$ matrix whose entries are

$$\mathbb{P}_l = \begin{bmatrix} g_{1,l}^{(k)} & g_{1,l}^{(k+1)} & \cdots & g_{1,l}^{(n-1)} \\ g_{2,l}^{k} & g_{2,l}^{(k+1)} & \cdots & g_{2,l}^{(n-1)} \\ \vdots & \vdots & & \vdots \\ g_{k,l}^{(k)} & g_{k,l}^{(k+1)} & \cdots & g_{k,l}^{(n-1)} \end{bmatrix}. \tag{11.43}$$

Similarly, the $k \times n$ (transform-domain) generator matrix becomes

$$\mathbf{G}(D) = \begin{bmatrix} 1 & 0 & \cdots & 0 & g_1^{(k)}(D) & \cdots & g_1^{(n-1)}(D) \\ 0 & 1 & \cdots & 0 & g_2^{(k)}(D) & \cdots & g_2^{(n-1)}(D) \\ \vdots & \vdots & & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 & g_k^{(k)}(D) & \cdots & g_k^{(n-1)}(D) \end{bmatrix}_{k \times n}. \tag{11.44}$$

$k \times k$

$$\mathbf{H}(D) = \begin{bmatrix} g_1^{(k)}(D) & g_2^{(k)}(D) & \cdots & g_k^{(k)}(D) & 1 & 0 & \cdots & 0 \\ g_1^{(k+1)}(D) & g_2^{(k+1)}(D) & \cdots & g_k^{(k+1)}(D) & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots \\ g_1^{(n-1)}(D) & g_2^{(n-1)}(D) & \cdots & g_k^{(n-1)}(D) & 0 & 0 & \cdots & 1 \end{bmatrix}_{n-k \times n}, \tag{11.46a}$$

$(n-k) \times (n-k)$

Any codeword $\mathbf{v}$ (or $\mathbf{V}(D)$) must then satisfy the parity-check equations

$$\mathbf{v}\mathbf{H}^T = \mathbf{0}, \tag{11.47}$$

where the matrix $\mathbf{0}$ has semi-infinite dimensions in this case, or

$$\underset{1 \times n}{\mathbf{V}(D)}\underset{n \times (n-k)}{\mathbf{H}^T(D)} = \underset{1 \times (n-k)}{\mathbf{0}(D)}, \quad * \tag{11.48}$$

where $\mathbf{0}(D)$ represents the $1 \times (n-k)$ matrix of all-zero sequences. In other words, the set of all codewords $\mathbf{v}$ (or $\mathbf{V}(D)$) is the null space of the parity-check matrix $\mathbf{H}$ (or $\mathbf{H}(D)$).

EXAMPLE 11.4 A Rate $R = 1/2$ Systematic Feedforward Convolutional Encoder

Consider the (2, 1, 3) systematic feedforward encoder shown in Figure 11.4. The generator sequences are $\mathbf{g}^{(0)} = (1\ 0\ 0\ 0)$ and $\mathbf{g}^{(1)} = (1\ 1\ 0\ 1)$, and the generator matrices are given by

$$\mathbf{G} = \begin{bmatrix} 11 & 01 & 00 & 01 & & & \\ & 11 & 01 & 00 & 01 & & \\ & & 11 & 01 & 00 & 01 & \\ & & & \ddots & & \ddots & \end{bmatrix}, \tag{11.49}$$

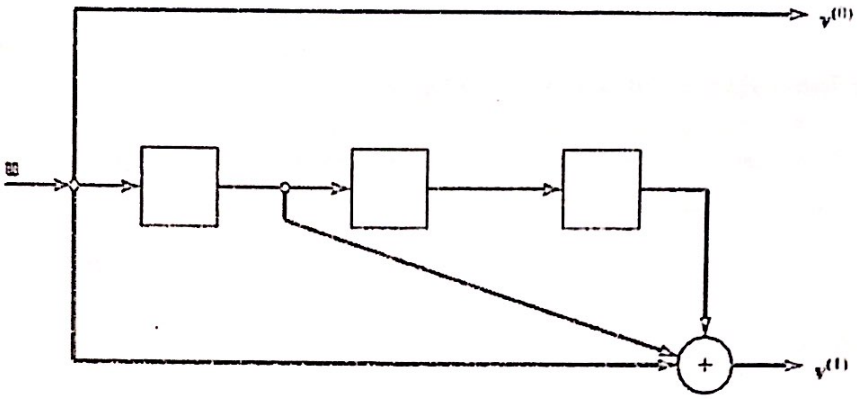


FIGURE 11.4: A (2, 1, 3) binary systematic feedforward convolutional encoder.

$$\mathbb{G}(D) = [1 \quad 1 + D + D^3]. \tag{11.50}$$

For an input sequence $\mathbf{u}(D) = 1 + D^2 + D^3$, the output information sequence is

$$\mathbf{v}^{(0)}(D) = \mathbf{u}(D)\mathbf{g}^{(0)}(D) = (1 + D^2 + D^3)(1) = 1 + D^2 + D^3, \tag{11.51}$$

the output parity sequence is

$$\begin{aligned} \mathbf{v}^{(1)}(D) &= \mathbf{u}(D)\mathbf{g}^{(1)}(D) = (1 + D^2 + D^3)(1 + D + D^3) \\ &= 1 + D + D^2 + D^3 + D^4 + D^5 + D^6, \end{aligned} \tag{11.52}$$

and the codeword is given by

$$\mathbb{V}(D) = [1 + D^2 + D^3 \quad 1 + D + D^2 + D^3 + D^4 + D^5 + D^6] \tag{11.53a}$$

or

$$\begin{aligned} \mathbf{v}(D) &= \mathbf{v}^{(0)}(D^2) + D\mathbf{v}^{(1)}(D^2) \\ &= 1 + D + D^3 + D^4 + D^5 + D^6 + D^7 + D^9 + D^{11} + D^{13}, \end{aligned} \tag{11.53b}$$

which we can also write as

$$\mathbf{v} = (11, 01, 11, 11, 01, 01, 01) \tag{11.53c}$$

$$\mathbb{G}(D) = [1 \quad 1 + D + D^3]. \tag{11.50}$$

For an input sequence $\mathbb{u}(D) = 1 + D^2 + D^3$, the output information sequence is

$$\mathbb{v}^{(0)}(D) = \mathbb{u}(D)\mathbb{g}^{(0)}(D) = (1 + D^2 + D^3)(1) = 1 + D^2 + D^3, \tag{11.51}$$

the output parity sequence is

$$\begin{aligned}\mathbb{v}^{(1)}(D) = \mathbb{u}(D)\mathbb{g}^{(1)}(D) &= (1 + D^2 + D^3)(1 + D + D^3) \\ &= 1 + D + D^2 + D^3 + D^4 + D^5 + D^6,\end{aligned} \tag{11.52}$$

and the codeword is given by

$$\mathbb{V}(D) = [1 + D^2 + D^3 \quad 1 + D + D^2 + D^3 + D^4 + D^5 + D^6] \tag{11.53a}$$

or

$$\begin{aligned}\mathbb{v}(D) &= \mathbb{v}^{(0)}(D^2) + D\mathbb{v}^{(1)}(D^2) \\ &= 1 + D + D^3 + D^4 + D^5 + D^6 + D^7 + D^9 + D^{11} + D^{13}.\end{aligned} \tag{11.53b}$$

which we can also write as

$$\mathbb{v} = (11, 01, 11, 11, 01, 01, 01). \tag{11.53c}$$

Using (11.45) and (11.46b), we see that the parity-check matrices are given by

$$\mathbb{H} = \begin{bmatrix} 11 & & & & & \\ 10 & 11 & & & & \\ 00 & 10 & 11 & & & \\ 10 & 00 & 10 & 11 & & \\ & 10 & 00 & 10 & 11 & \\ & & 10 & 00 & 10 & 11 \\ & & & \ddots & & \ddots \end{bmatrix} \tag{11.54}$$

and

$$\mathbb{H}(D) = [\mathbb{h}^{(0)}(D) \quad 1] = [1 + D + D^3 \quad 1]. \tag{11.55}$$

For the codeword of (11.53) we then have

$$\mathbb{v}\mathbb{H}^T = [11, 01, 11, 11, 01, 01, 01] \begin{bmatrix} 1 & 1 & 0 & 1 & & & \\ 1 & 0 & 0 & 0 & & & \\ & 1 & 1 & 0 & 1 & & \\ & 1 & 0 & 0 & 0 & & \\ & & 1 & 1 & 0 & 1 & \\ & & 1 & 0 & 0 & 0 & \\ & & & 1 & 1 & 0 & 1 \\ & & & 1 & 0 & 0 & 0 \\ & & & & 1 & 1 & 0 \\ & & & & 1 & 0 & 0 \\ & & & & & 1 & 1 \\ & & & & & 1 & 0 \\ & & & & & & 1 \end{bmatrix} = \mathbb{0}, \tag{11.56}$$

or

$$\begin{aligned}\mathbb{V}(D)\mathbb{H}^T(D) &= [1 + D^2 + D^3 \quad 1 + D + D^2 + D^3 + D^4 + D^5 + D^6]\begin{bmatrix} 1 + D + D^3 \\ 1 \end{bmatrix} \\ &= \mathbb{0}(D).\end{aligned} \tag{11.57}$$

EXAMPLE 11.5 A Rate $R = 2/3$ Systematic Feedforward Convolutional Encoder

Now, consider the rate $R = 2/3$ systematic feedforward encoder with generator matrix

$$\mathbf{G}(D) = \begin{bmatrix} 1 & 0 & 1+D+D^2 \\ 0 & 1 & 1+D \end{bmatrix}_{2\times 3}. \tag{11.58}$$

Using (11.46b), we can write the parity-check matrix as

$$\mathbf{H}(D) = \begin{bmatrix} \mathbf{h}^{(0)}(D) & \mathbf{h}^{(1)}(D) & 1 \end{bmatrix} = \begin{bmatrix} 1+D+D^2 & 1+D & 1 \end{bmatrix}_{1\times 3}. \tag{11.59}$$

The controller canonical form realization of the encoder requires a total of $\nu = \nu_1 + \nu_2 = 3$ delay elements and is shown in Figure 11.5(a). This realization results in a (3, 2, 3) encoder. Moreover, since the output information sequences are given by $\mathbf{v}^{(0)}(D) = \mathbf{u}^{(1)}(D)$ and $\mathbf{v}^{(1)}(D) = \mathbf{u}^{(2)}(D)$, and the output parity sequence is given by

$$\begin{aligned} \mathbf{v}^{(2)}(D) &= \mathbf{u}^{(1)}(D)\mathbf{g}_1^{(2)}(D) + \mathbf{u}^{(2)}(D)\mathbf{g}_2^{(2)}(D), \\ &= (1+D+D^2)\mathbf{u}^{(1)}(D) + (1+D)\mathbf{u}^{(2)}(D), \end{aligned} \tag{11.60}$$

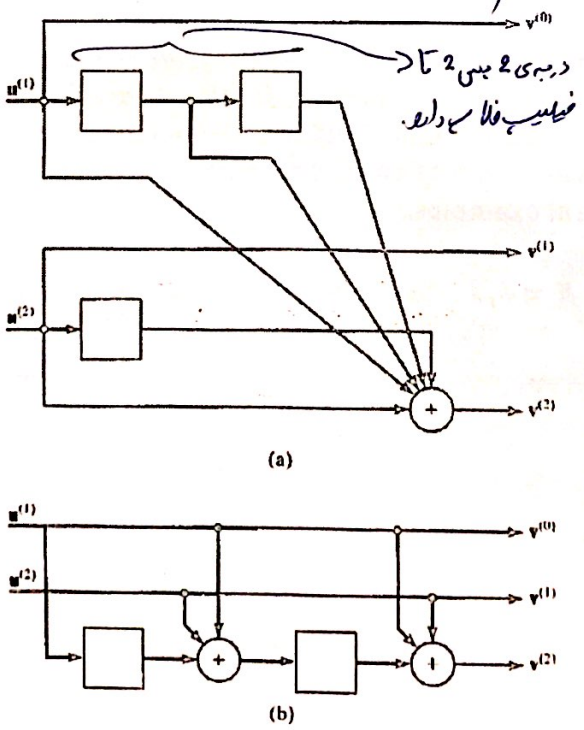
the encoder can also be realized as shown in Figure 11.5(b). Note that this realization requires only two delay elements rather than three.

$n=3,\ k=2$

$$v(D) = u(D)\,G(D) = \begin{pmatrix} u^{(1)}(D) & u^{(2)}(D) \end{pmatrix} \begin{pmatrix} 1 & 0 & 1+D+D^2 \\ 0 & 1 & 1+D \end{pmatrix}$$

$$= \left(u^{(1)}(D),\ u^{(2)}(D),\ u^{(1)}(D)(1+D+D^2) + u^{(2)}(D)(1+D) \right)$$

$\Leftarrow$ $u^{(1)} \longrightarrow v^{(0)}(D)$, $u^{(2)} \longrightarrow v^{(1)}(D)$, $\longrightarrow v^{(2)}(D)$



درجه‌ی 2 یعنی 2 تا ضلیب فلاپ دارد.

(3,2,3)

(3, 2,2)

از روی این شکل طراحی کنیم

$$G(D) = \begin{bmatrix} 1 & 0 & D^2+D+1 \\ 0 & 1 & D+1 \end{bmatrix}$$

FIGURE 11.5: (a) Controller canonical form and (b) observer canonical form realizations of a rate $R = 2/3$ binary systematic feedforward convolutional encoder.

The encoder realization of Figure 11.5(b) is called *observer canonical form*.

EXAMPLE 11.5 (Continued)

For an observer canonical form realization, we rewrite the generator matrix of (11.58) as

$$\mathbf{G}(D) = \begin{bmatrix} 1 & 0 & D^2+D+1 \\ 0 & 1 & D+1 \end{bmatrix} \tag{11.61}$$

and the parity-check matrix of (11.59) as

$$\mathbf{H}(D) = \begin{bmatrix} D^2+D+1 & D+1 & 1 \end{bmatrix}. \tag{11.62}$$

In this way we can clearly identify the coefficients of the polynomials in (11.61) and (11.62) with the connections in Figure 11.5(b). For this realization, we see that $\nu_0 = 0$, $\nu_1 = 0$, $\nu_2 = 2$, $m = \max_{(0 \le j \le n-1)} \nu_j = 2$, and $\nu = \Sigma_{(0 \le j \le n-1)} \nu_j = 2$. Thus, the observer canonical form realization results in a (3, 2, 2) encoder.

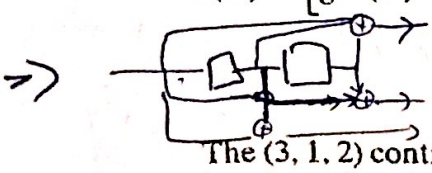
We will continue our discussion of *minimal encoder realizations* after introducing another class of systematic encoders. *Systematic feedback encoders* generate the same codes as corresponding nonsystematic feedforward encoders but exhibit a different mapping between information sequences and codewords. We again illustrate the basic concepts using several examples.

EXAMPLE 11.6 A Rate $R = 1/3$ Systematic Feedback Convolutional Encoder

$R = 1/3$

Consider the rate $R = 1/3$ nonsystematic feedforward convolutional encoder with generator matrix

$$\mathbf{G}(D) = \begin{bmatrix} \mathbf{g}^{(0)}(D) & \mathbf{g}^{(1)}(D) & \mathbf{g}^{(2)}(D) \end{bmatrix} = \begin{bmatrix} 1+D+D^2 & 1+D^2 & 1+D \end{bmatrix}. \tag{11.63a}$$



The (3, 1, 2) controller canonical form realization is shown in Figure 11.6(a). Reversing the order of the polynomials in (11.63a), we have

$$\mathbf{G}(D) = \begin{bmatrix} D^2+D+1 & D^2+1 & D+1 \end{bmatrix}. \tag{11.63b}$$

from which we obtain the (3, 1, 5) observer canonical form encoder realization shown in Figure 11.6(b). Note that in this case the controller canonical form realization has overall constraint length $\nu = 2$, that is, 4 states, whereas the observer canonical form realization requires 32 states! Now, we divide each entry in $\mathbb{G}(D)$ by the polynomial $\mathbf{g}^{(0)}(D) = 1 + D + D^2$ to obtain the modified generator matrix

$$\begin{aligned}\mathbb{G}'(D) &= [1 \quad \mathbf{g}^{(1)}(D)/\mathbf{g}^{(0)}(D) \quad \mathbf{g}^{(2)}(D)/\mathbf{g}^{(0)}(D)] \\ &= [1 \quad (1+D^2)/(1+D+D^2) \quad (1+D)/(1+D+D^2)]\end{aligned} \tag{11.64}$$

in systematic feedback form. We note in this case that the impulse response of the encoder represented by $\mathbb{G}'(D)$ has infinite duration; that is, the feedback shift register realization of $\mathbb{G}'(D)$ is an *infinite impulse response* (IIR) linear system. For this reason, the (time-domain) generator matrix $\mathbb{G}'$ corresponding to $\mathbb{G}'(D)$ contains sequences of infinite length. For example, the ratio of generator polynomials $\mathbf{g}^{(1)}(D)/\mathbf{g}^{(0)}(D)$ in (11.64) produces the power series $(1 + D^2)/(1 + D + D^2) = 1 + D + D^2 + D^4 + D^5 + D^7 + D^8 + D^{10} + \cdots$, whose time-domain equivalent is the infinite-length sequence $(11101101101\cdots)$.

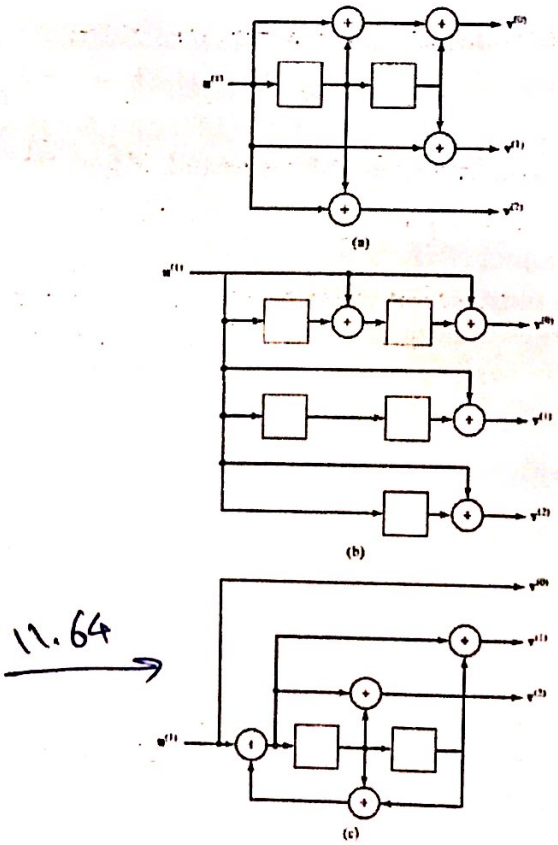


FIGURE 11.6: (a) The controller canonical form and (b) the observer canonical form realizations of a nonsystematic feedforward encoder, and (c) the controller canonical form realization of an equivalent systematic feedback encoder.

$u = (10\ 11)$

$u(D) = 1 + D^2 + D^3$

توضيح + مثال فيديو ~ Gomin

That the generator and parity-check sequences in Example 11.6 exhibit IIR behavior is characteristic of feedback encoders. For this reason, feedback encoders are more easily described using the (transform-domain) matrices $\mathbb{G}'(D)$ and $\mathbb{H}'(D)$, whose entries contain rational functions, rather than the corresponding (time-domain) matrices $\mathbb{G}'$ and $\mathbb{H}'$, whose entries contain sequences of infinite length. Because the response to a single nonzero input in a feedback encoder has infinite duration, the codes produced by feedback encoders are called ***recursive convolutional codes.*** Thus, we say that a systematic feedback encoder produces a *systematic recursive convolutional code* (SRCC). It is important to note that the SRCC produced by a systematic feedback encoder contains exactly the same set of codewords as the nonrecursive convolutional code produced by an equivalent nonsystematic feedforward encoder; however, as noted in Example 11.6, the mapping between information sequences and codewords is different in the two cases. In particular, for nonrecursive codes, information sequences of weight 1 produce finite-length codewords, whereas for recursive codes, information sequences of weight 1 produce codewords of infinite length. This property of recursive codes turns out to be a key factor in the excellent performance achieved by the class of parallel concatenated convolutional codes, or turbo codes, to be discussed in Chapter 16.

---

delay

$\mathbb{I}D^{l}$ $k \times k$

$\mathbb{G}(D) \cdot \mathbb{G}^{-1}(D)$ $k \times n$ $n \times k$

In some cases it is desirable to recover a noisy version of the information sequence directly from the codeword, without decoding. For example, this may be the case when the channel noise conditions are being tested, when the decoder is malfunctioning, or when the channel SNR is high enough that decoding is not necessary. In general, this process requires an inverter to recover the information sequence from the codeword. For systematic encoders the inverter is trivial, since the information sequence is directly available. For nonsystematic encoders, on the other hand, an $n \times k$ inverse matrix $\mathbb{G}^{-1}(D)$ must exist such that

$$\mathbb{G}(D)\mathbb{G}^{-1}(D) = \mathbb{I}D^{l} \tag{11.83}$$

for some $l \geq 0$, where $\mathbb{I}$ is the $k \times k$ identity matrix. Equations (11.32) and (11.83) then imply that

$$\mathbb{V}(D)\mathbb{G}^{-1}(D) = \mathbb{U}(D)\mathbb{G}(D)\mathbb{G}^{-1}(D) = \mathbb{U}(D)D^{l}, \tag{11.84}$$

and the information sequence can be recovered with an l-time unit delay from the codeword by letting $\mathbb{V}(D)$ be the input to the n-input, k-output linear sequential circuit whose transfer function matrix is $\mathbb{G}^{-1}(D)$.

For an $(n, 1, \nu)$ convolutional encoder, a generator matrix $\mathbb{G}(D)$ has a feedforward inverse $\mathbb{G}^{-1}(D)$ of delay l if and only if

$$\mathrm{GCD}[\mathbf{g}^{(0)}(D), \mathbf{g}^{(1)}(D), \cdots, \mathbf{g}^{(n-1)}(D)] = D^l \tag{11.85}$$

for some $l \geqq 0$, where GCD denotes greatest common divisor. For an (n, k, ν) encoder with $k > 1$, let $\Delta_i(D)$, $i = 1, 2, \cdots, \binom{n}{k}$, be the determinants of the $\binom{n}{k}$ distinct $k \times k$ submatrices of the generator matrix $\mathbb{G}(D)$. Then, a feedforward inverse of delay l exists if and only if

$$\mathrm{GCD}\left[\Delta_i(D) : i = 1, 2, \cdots, \binom{n}{k}\right] = D^l \quad D^{\ell} \tag{11.86}$$

for some $l \geqq 0$. Procedures for constructing feedforward inverses $\mathbb{G}^{-1}(D)$ are given in Massey and Sain [14].

EXAMPLE 11.9 Feedforward Encoder Inverses

a. For the (2, 1, 3) encoder of Figure 11.1, the generator matrix is given by

$$\mathbb{G}(D) = \left[\mathbf{g}^{(0)}(D) \quad \mathbf{g}^{(1)}(D)\right] = \left[1 + D^2 + D^3 \quad 1 + D + D^2 + D^3\right]_{1\times 2} \tag{11.87a}$$

$$\mathrm{GCD}[\mathbf{g}^{(0)}(D),\ \mathbf{g}^{(1)}(D)] = \mathrm{GCD}[1+D^2+D^3,\ 1+D+D^2+D^3] = 1, \tag{11.87b}$$

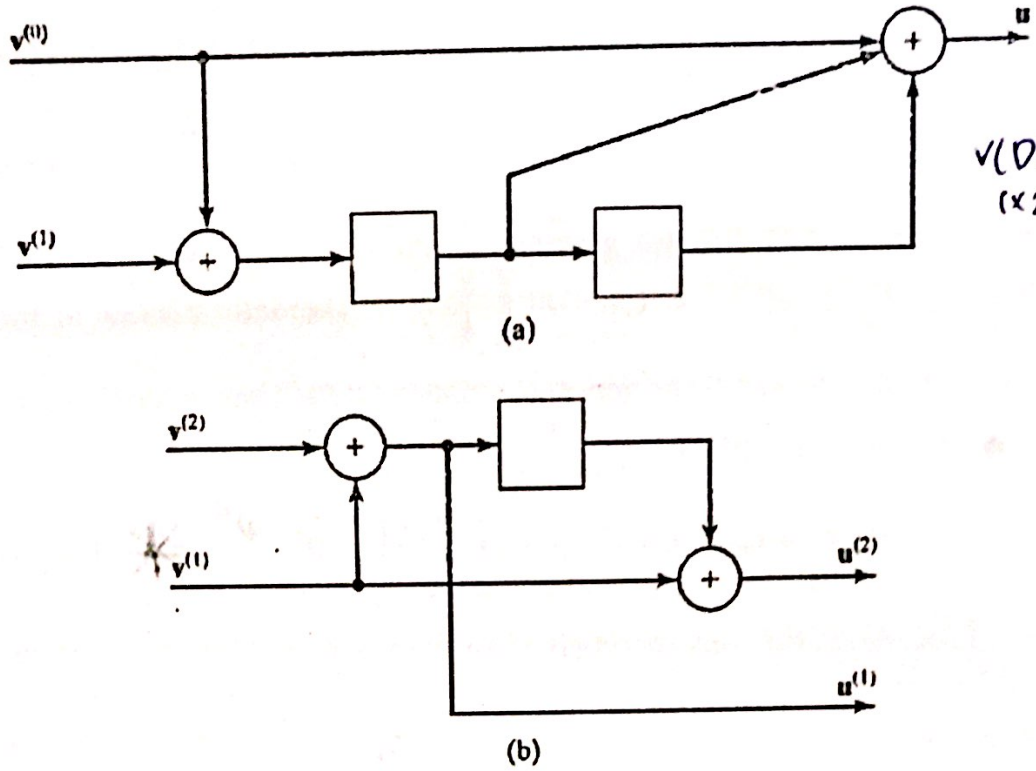
and the transfer function matrix

$G(D)\cdot G^{-1}(D) = I\cdot D^{\ell}$

$$\mathbb{G}^{-1}(D) = \begin{bmatrix} \overset{g_1(D)}{1 + D + D^2} \\ \underset{g_2(D)}{D + D^2} \end{bmatrix}_{2\times 1} \tag{11.88}$$

$k \times k \rightarrow k = 1$

provides the required feedforward inverse of delay 0; that is, $\mathbb{G}(D)\mathbb{G}^{-1}(D) = 1$. A realization of the encoder inverse is shown in Figure 11.12(a).



$v(D) \cdot G(D) = \underset{1\times 1}{0}$
$1\times 2 \quad 2\times 1$

FIGURE 11.12: Feedforward encoder inverses for (a) a (3, 1, 2) encoder and (b) a (3, 2, 2) encoder.

$G^{-1}(D) = \begin{bmatrix} 1+D+D^2 \\ D+D^2 \end{bmatrix}$

b. For the (3, 2, 2) encoder of Figure 11.2 with generator matrix $\mathbf{G}(D)$ given in (11.34), the 2×2 submatrices of $\mathbf{G}(D)$ yield determinants $1 + D + D^2$, $1 + D^2$, and 1. Because

$$\mathrm{GCD}[1 + D + D^2,\ 1 + D^2,\ 1] = 1, \tag{11.89}$$

there exists a feedforward inverse of delay 0. The required transfer function matrix is given by

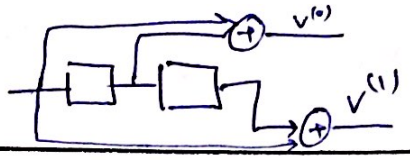
$$\mathbf{G}^{-1}(D) = \begin{bmatrix} 0 & 0 \\ 1 & 1+D \\ 1 & D \end{bmatrix}, \tag{11.90}$$

and its realization is shown in Figure 11.12(b).

قدرت تصحیح خطا و قدرت کد به ≈30min

اگر ورودی $u=1000$ باشد

$V=2000 \approx 2\times 2$ ← Code sequence ربط دارد



(10)

V80

EXAMPLE 11.10 A Catastrophic Encoder

Consider the (2, 1, 2) encoder whose generator matrix is given by

$k\times l$ ← $n\times l + m\times n$ ← code sequence

shift register

$$\mathbf{G}(D) = \left[\mathbf{g}^{(0)}(D)\ \ \mathbf{g}^{(1)}(D)\right] = \left[1+D \quad 1+D^2\right]. \tag{11.91a}$$

In this case we see that

$$\mathrm{GCD}[\mathbf{g}^{(0)}(D)\ \ \mathbf{g}^{(1)}(D)] = \mathrm{GCD}[1+D,\ \ 1+D^2] = 1+D, \neq D^l \tag{11.91b}$$

چون ماتریس معکوس ندارد

and a feedforward inverse does not exist. If the information sequence is $\mathbf{u}(D) = \frac{1}{1+D} = 1 + D + D^2 + \cdots$, then the output sequences are $\mathbf{v}^{(0)}(D) = 1$ and $\mathbf{v}^{(1)}(D) = 1 + D$; that is, the codeword has only weight 3 even though the information sequence has infinite weight. If this codeword is transmitted over a BSC, and the three nonzero bits are changed to zeros by the channel noise, then the received sequence will be all zero. A maximum likelihood decoder will then produce the all-zero codeword as its estimate, since this is a valid codeword and it agrees exactly with the received sequence. Thus, the estimated information sequence will be $\mathbf{u}(D) = \mathbf{0}(D)$, implying an infinite number of decoding errors caused by a finite number (only three in this case) of channel errors. Clearly, this is very undesirable, and the encoder is said to be subject to catastrophic error propagation; that is, it is a *catastrophic encoder*.

≈10min مهم

$u(D) = 1 + D + \cdots + D^n$ ⇐ $u = \underbrace{111\cdots1}_{1000}$ $n=1000$

اگر فقط پیام شامل 1000 بیت 1 باشد.

$\Rightarrow v(D) = u(D)G(D) = [u(D)(1+D) \quad u(D)(1+D^2)] = [(1+D)^{n+1} \quad (1+D^{n+1})(1+D)]$, $D^n+1 = (D+1)(1+D+D^2+\cdots+D^{n-1})$

$\Rightarrow v(D) = [(1+D)^{n+1}, 1+D+D^{n+1}+D^{n+2}]$ $\quad v = 11\,00\cdots0\,11\,01$ $\quad \boxed{w(v)=5}$

اگر یک خطا داشته باشیم باعث می‌شود 1000 تا شود. (catastrophic) افتضاح

STRUCTURAL PROPERTIES OF CONVOLUTIONAL CODES

$\underline{u_i} \rightarrow \boxed{\text{coder}} \xrightarrow{\underline{v_i}}$ $\quad \underline{v_i} = f(u_i, u_{i-1}, \ldots, u_{i-l})$

حالت کدر در لحظه i ام

$\underline{v_i} = (u_i, \sigma_i)$

$\sigma_{i+1} = g(u_i, \sigma_i)$

Because a convolutional encoder is a linear sequential circuit, its operation can be described by a state diagram. The state of an encoder is defined as its shift register contents. For an (n, k, ν) encoder in controller canonical form, the ith shift register at time unit l (when $u_l^{(1)}, u_l^{(2)}, \cdots, u_l^{(k)}$ are the encoder inputs) contains ν_i bits, denoted as $s_{l-1}^{(i)}, s_{l-2}^{(i)}, \cdots, s_{l-\nu_i}^{(i)}$, where $s_{l-1}^{(i)}$ represent the contents of the leftmost delay element, and $s_{l-\nu_i}^{(i)}$ represents the contents of the rightmost delay element, $1 \le i \le k$.

DEFINITION 11.6 The *encoder state* σ_l at time unit l is the binary ν-tuple

$$\sigma_l = (s_{l-1}^{(1)} s_{l-2}^{(1)} \cdots s_{l-\nu_1}^{(1)} s_{l-1}^{(2)} s_{l-2}^{(2)} \cdots s_{l-\nu_2}^{(2)} \cdots s_{l-1}^{(k)} s_{l-2}^{(k)} \cdots s_{l-\nu_k}^{(k)}). \tag{11.93}$$

Thus, we see from (11.20) that there are a total of 2^ν different possible states. For an $(n, 1, \nu)$ encoder, the encoder state at time unit l is simply

k register shift

$$\sigma_l = (s_{l-1}\ s_{l-2}\ \cdots\ s_{l-\nu}). \tag{11.94}$$

$\boxed{s_{l-1} \mid - - - \mid s_{l-\nu}}$

ν

$i = b_0 + 2b_1 + \cdots + 2^{v-1} b_{v-1}$

$\boxed{i \to \text{state}}$

$\leftarrow S_i$ یا σ_i

اندیس کمیت بیشتر داشتیم

$$\sigma_k = (u_{l-1}^{(1)} u_{l-2}^{(1)} \cdots u_{l-\nu_1}^{(1)} u_{l-1}^{(2)} u_{l-2}^{(2)} \cdots u_{l-\nu_2}^{(2)} \cdots u_{l-1}^{(k)} u_{l-2}^{(k)} \cdots u_{l-\nu_k}^{(k)}). \tag{11.95}$$

$$\sigma = (u_{l-1} \ u_{l-2} \ \cdots \ u_{l-\nu}). \tag{11.96}$$

Each new input block of k bits causes the encoder to shift; that is, there is a transition to a new state. Hence, there are 2^k branches leaving each state in the state diagram, one corresponding to each different input block. For an $(n, 1, \nu)$ encoder, therefore, there are only two branches leaving each state. Each branch in the state diagram is labeled with the k inputs $(u_l^{(1)}, u_l^{(2)}, \cdots, u_l^{(k)})$ causing the transition and the n corresponding outputs $(v_l^{(0)}, v_l^{(1)}, \cdots, v_l^{(n-1)})$. The state diagrams for the feedforward controller canonical form realizations of the $(2, 1, 3)$ encoder of Figure 11.1 and the $(3, 2, 2)$ encoder of Figure 11.2 are shown in Figures 11.13(a) and 11.13(b), respectively. The states are labeled $S_0, S_1, \cdots, S_{2^\nu-1}$, where by convention S_i represents the state whose binary ν-tuple representation $b_0, b_1, \cdots, b_{\nu-1}$ is equivalent to the integer $i = b_0 2^0 + b_1 2^1 + \cdots + b_{\nu-1} 2^{\nu-1}$. For example, for the $(2, 1, 3)$ encoder of Figure 11.1, if the shift register contents at time l are given by $\sigma_l = (s_{l-1} \ s_{l-2} \ s_{l-3}) = (u_{l-1} \ u_{l-2} \ u_{l-3})$, the state at time l is denoted as S_i, where $i = u_{l-1} + 2u_{l-2} + 4u_{l-3}$.

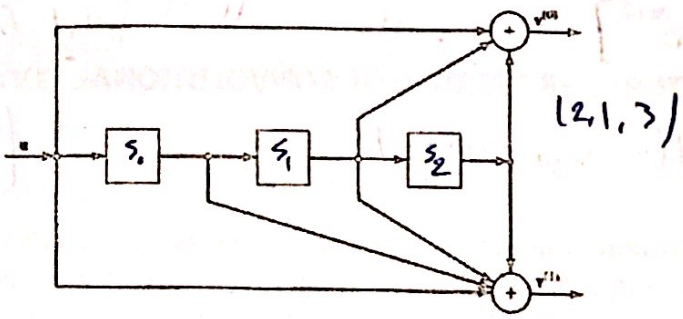


FIGURE 11.1: A rate $R = 1/2$ binary nonsystematic feedforward convolutional encoder with memory order $m = 3$.

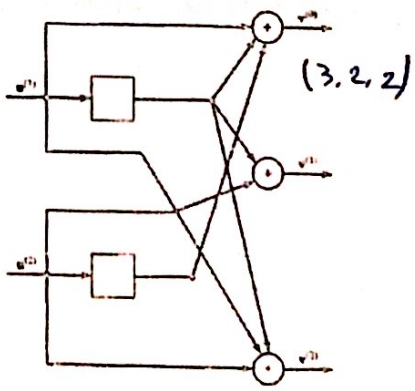


FIGURE 11.2: A rate $R = 2/3$ binary nonsystematic feedforward convolutional encoder with memory order $m = 1$.

(A)

(11) محتوای shift ، 3 تا که نام گذاری کردیم ⇐ $2^3=8$ حالت داریم .

از شکل صفحه قبل

مشخص کننده v

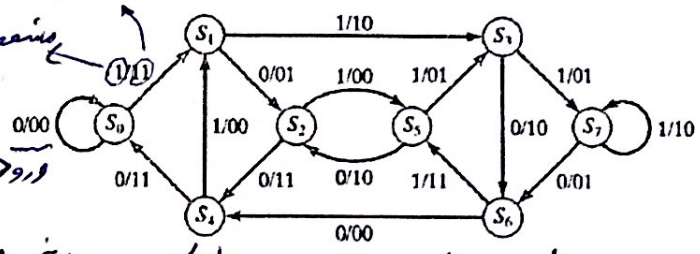
مشخص کننده u

(13) $u=1101$

حالت اولیه همیشه S_0

1 1 0 1

11 10 10 11 10 11 11



(a)

ورودی صفر، خروجی صفر می‌شود.

اگر ورودی 1 شود (در شکل بالای صفحه قبل) خروجی 11 می‌شود.

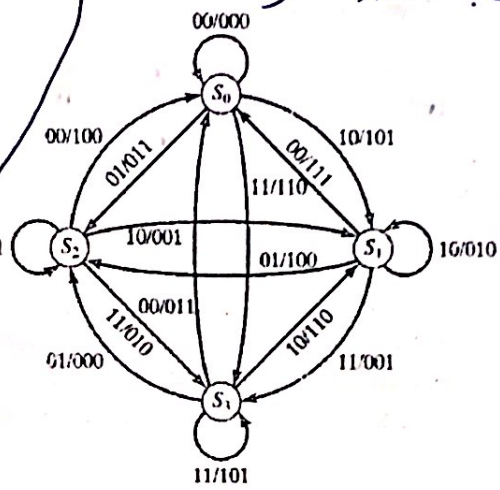
در حالتی به وجود می‌آید که shift می‌دهیم تا به state ، صفر بدهیم و خروجی در مسیر بازگشت هستند.

S_1 نام گذاری می‌کنیم

اگر ورودی صفر بدهیم ⇐ خروجی 01 می‌شود

S_2 نام گذاری می‌شود.

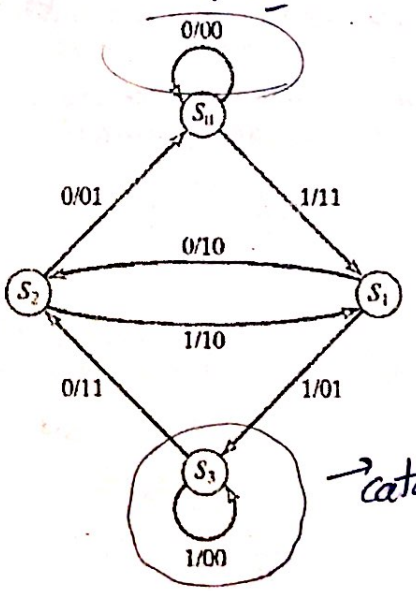
و به همین ترتیب ...



(b)

FIGURE 11.13: Encoder state diagrams for (a) the (2, 1, 3) encoder of Figure 11.1 and (b) the (3, 2, 2) encoder of Figure 11.2.

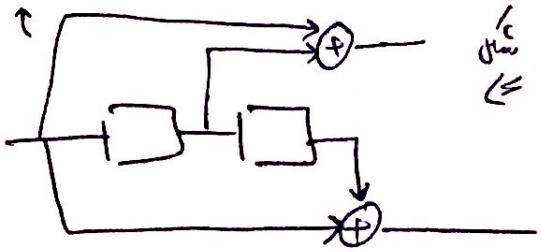
این loop استثنا است و در همه‌ی state diagram وجود دارد



یک loop داریم که weight صفر شود پس catastrophic است.

2 تا shift داریم ⇐ $2^2=4$ حالت

FIGURE 11.14: State diagram of a (2, 1, 2) catastrophic encoder.

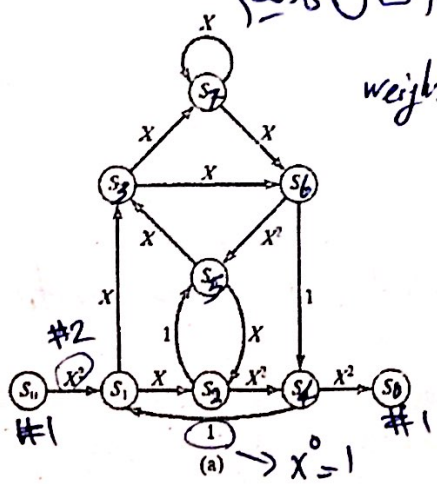


مثال ⇐ $G(D)=[1+D \;,\; 1+D^2]$

#1 دو طرف ۲ می‌گذاریم • weight

#2 چون ۱۱ است weight آن را با X^2 نشان می‌دهیم

#3 اگر از ورودی به خروجی، X^0 را در هم ضرب کنیم، weight codesequence را بدست می‌آوریم.



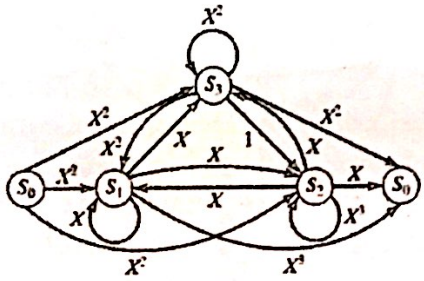


FIGURE 11.15: Modified encoder state diagrams for (a) the (2, 1, 3) encoder of Figure 11.1 and (b) the (3, 2, 2) encoder of Figure 11.2.

EXAMPLE 11.11 Computing Path Gains in a State Diagram

a. The path representing the state sequence $S_0S_1S_3S_7S_6S_5S_2S_4S_0$ in Figure 11.15(a) has path gain $X^2 \cdot X^1 \cdot X^1 \cdot X^1 \cdot X^2 \cdot X^1 \cdot X^2 \cdot X^2 = X^{12}$, and the corresponding codeword has weight 12.

b. The path representing the state sequence $S_0S_1S_3S_2S_0$ in Figure 11.15(b) has path gain $X^2 \cdot X^1 \cdot X^0 \cdot X^1 = X^4$, and the corresponding codeword has weight 4.

We can determine the codeword WEF of a code by considering the modified state diagram of the encoder as a *signal flow graph* and applying Mason's gain formula [15] to compute its *transfer function*,

تعداد دنباله‌های که با وزن d

$$A(X) = \sum_d A_d X^d, \tag{11.97}$$

where A_d is the number of codewords of weight d.

In a signal flow graph, a path connecting the initial state to the final state that does not go through any state twice is called a *forward path*. Let F_i be the gain of the ith forward path. A closed path starting at any state and returning to that state without going through any other state twice is called a *cycle*. Let C_i be the gain of the ith cycle. A set of cycles is *nontouching* if no state belongs to more than one cycle in the set. Let $\{i\}$ be the set of all cycles, $\{i', j'\}$ be the set of all pairs of nontouching cycles, $\{i'', j'', l''\}$ be the set of all triples of nontouching cycles, and so on. Then, we define

$$\Delta = 1 - \sum_i C_i + \sum_{i',j'} C_{i'}C_{j'} - \sum_{i'',j'',l''} C_{i''}C_{j''}C_{l''} + \cdots, \tag{11.98}$$

where $\sum_i C_i$ is the sum of the cycle gains, $\sum_{i',j'} C_{i'}C_{j'}$ is the product of the cycle gains of two nontouching cycles summed over all pairs of nontouching cycles, $\sum_{i'',j'',l''} C_{i''}C_{j''}C_{l''}$ is the product of the cycle gains of three nontouching cycles summed over all triples of nontouching cycles, and so on. Finally, Δ_i is defined exactly like Δ but only for that portion of the graph not touching the ith forward path; that is, all states along the ith forward path, along with all branches connected to those states, are removed from the graph when computing Δ_i. We can now state Mason's formula for computing the transfer function $A(X)$ of a graph as

$$A(X) = \frac{\sum_i F_i \Delta_i}{\Delta}, \tag{11.99}$$

where the sum in the numerator is over all forward paths. We give several examples to clarify this procedure.

EXAMPLE 11.12 Computing the WEF of a (2, 1, 3) Code

Consider the code produced by the (2, 1, 3) nonsystematic feedforward encoder of Figure 11.1, whose modified state diagram is shown in Figure 11.15(a). There are 11 cycles in the graph of Figure 11.15(a):

Cycle 1:	$S_1S_3S_7S_6S_5S_2S_4S_1$	$(C_1 = X^8)$
Cycle 2:	$S_1S_3S_7S_6S_4S_1$	$(C_2 = X^3)$
Cycle 3:	$S_1S_3S_6S_5S_2S_4S_1$	$(C_3 = X^7)$
Cycle 4:	$S_1S_3S_6S_4S_1$	$(C_4 = X^2)$
Cycle 5:	$S_1S_2S_5S_3S_7S_6S_4S_1$	$(C_5 = X^4)$
Cycle 6:	$S_1S_2S_5S_3S_6S_4S_1$	$(C_6 = X^3)$
Cycle 7:	$S_1S_2S_4S_1$	$(C_7 = X^3)$
Cycle 8:	$S_2S_5S_2$	$(C_8 = X)$
Cycle 9:	$S_3S_7S_6S_5S_3$	$(C_9 = X^5)$
Cycle 10:	$S_3S_6S_5S_3$	$(C_{10} = X^4)$
Cycle 11:	S_7S_7	$(C_{11} = X)$

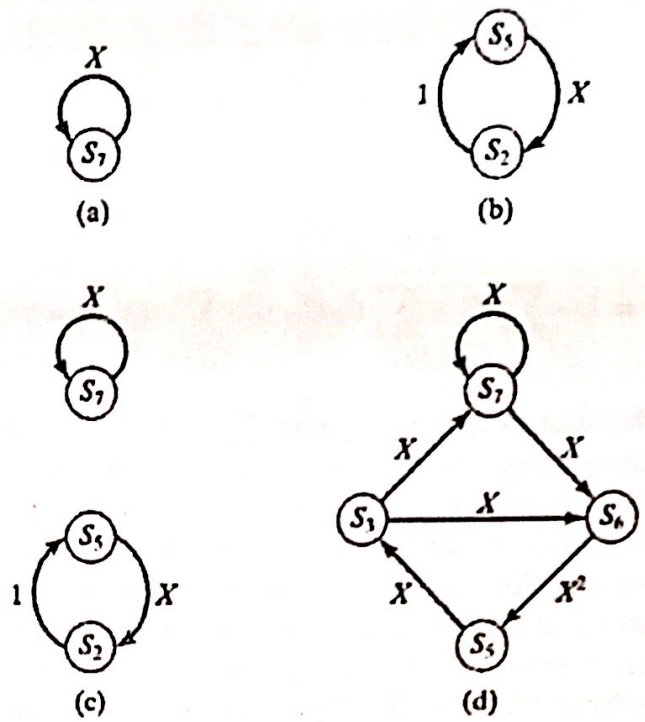


FIGURE 11.16: Subgraphs for computing Δ_i in Example 11.12.

There are 10 pairs of nontouching cycles:

Cycle Pair 1:	(Cycle 2, Cycle 8)	$(C_2C_8 = X^4)$
Cycle Pair 2:	(Cycle 3, Cycle 11)	$(C_3C_{11} = X^8)$
Cycle Pair 3:	(Cycle 4, Cycle 8)	$(C_4C_8 = X^3)$
Cycle Pair 4:	(Cycle 4, Cycle 11)	$(C_4C_{11} = X^3)$
Cycle Pair 5:	(Cycle 6, Cycle 11)	$(C_6C_{11} = X^4)$
Cycle Pair 6:	(Cycle 7, Cycle 9)	$(C_7C_9 = X^8)$
Cycle Pair 7:	(Cycle 7, Cycle 10)	$(C_7C_{10} = X^7)$
Cycle Pair 8:	(Cycle 7, Cycle 11)	$(C_7C_{11} = X^4)$
Cycle Pair 9:	(Cycle 8, Cycle 11)	$(C_8C_{11} = X^2)$
Cycle Pair 10:	(Cycle 10, Cycle 11)	$(C_{10}C_{11} = X^5)$

There are 2 triples of nontouching cycles:

Cycle Triple 1: (Cycle 4, Cycle 8, Cycle 11)	$(C_4C_8C_{11} = X^4)$
Cycle Triple 2: (Cycle 7, Cycle 10, Cycle 11)	$(C_7C_{10}C_{11} = X^8)$

There are no other sets of nontouching cycles. Therefore,

$$\begin{aligned}\Delta &= 1-(X^8+X^3+X^7+X^2+X^4+X^3+X^3+X+X^5+X^4+X)\\ &\quad +(X^4+X^8+X^3+X^3+X^4+X^8+X^7+X^4+X^2+X^5)-(X^4+X^8)\\ &= 1-2X-X^3. \end{aligned} \tag{11.100}$$

There are 7 forward paths in the graph of Figure 11.15(a):

Forward Path 1:	$S_0S_1S_3S_7S_6S_5S_2S_4S_0$	$(F_1 = X^{12})$
Forward Path 2:	$S_0S_1S_3S_7S_6S_4S_0$	$(F_2 = X^7)$
Forward Path 3:	$S_0S_1S_3S_6S_5S_2S_4S_0$	$(F_3 = X^{11})$
Forward Path 4:	$S_0S_1S_3S_6S_4S_0$	$(F_4 = X^6)$
Forward Path 5:	$S_0S_1S_2S_5S_3S_7S_6S_4S_0$	$(F_5 = X^8)$
Forward Path 6:	$S_0S_1S_2S_5S_3S_6S_4S_0$	$(F_6 = X^7)$
Forward Path 7:	$S_0S_1S_2S_4S_0$	$(F_7 = X^7)$

Forward paths 1 and 5 touch all states in the graph, and hence the subgraph not touching these paths contains no states. Therefore,

$$\Delta_1 = \Delta_5 = 1. \tag{11.101}$$

The subgraph not touching forward paths 3 and 6 is shown in Figure 11.16(a), and hence,

$$\Delta_3 = \Delta_6 = 1 - X. \tag{11.102}$$

The subgraph not touching forward path 2 is shown in Figure 11.16(b), and hence,

$$\Delta_2 = 1 - X. \tag{11.103}$$

The subgraph not touching forward path 4 is shown in Figure 11.16(c), and hence,

$$\Delta_4 = 1-(X+X)+(X^2) = 1-2X+X^2. \tag{11.104}$$

The subgraph not touching forward path 7 is shown in Figure 11.16(d), and hence,

$$\Delta_7 = 1-(X+X^4+X^5)+(X^5) = 1-X-X^4. \tag{11.105}$$

The transfer function for this graph, that is, the WEF of the associated (2, 1, 3) code, is then given by

$$\begin{aligned} A(X) &= \frac{X^{12}\cdot 1 + X^7(1-X)+X^{11}(1-X)+X^6(1-2X+X^2)+ X^8\cdot 1+X^7(1-X)+X^7(1-X-X^4)}{1-2X-X^3}\\ &= \frac{X^6+X^7-X^8}{1-2X-X^3} \\ &= X^6+3X^7+5X^8+11X^9+25X^{10}+\cdots. \end{aligned} \tag{11.106}$$

$$\begin{array}{l|l} X^6+X^7-X^8 & 1-2X-X^3 \\ \hline & X^6+3X^7 \\ & 3X^7-X^8+X^9 \end{array}$$

وزن 6 code sequence

code sequence 3 تا

وزن 7

5 تا وزن

8

EXAMPLE 11.13 Computing the WEF of a (3, 2, 2) Code

Consider the code produced by the (3, 2, 2) nonsystematic feedforward encoder of Figure 11.2, whose modified state diagram is shown in Figure 11.15(b). It can be shown that (see Problem 11.17) there are 8 cycles, 6 pairs of nontouching cycles, and 1 triple of nontouching cycles in the graph of Figure 11.15(b), and

$$\begin{aligned}\Delta &= 1 - (X^2 + X^4 + X^3 + X + X^2 + X + X^2 + X^3) \\ &\quad + (X^6 + X^2 + X^4 + X^3 + X^4 + X^5) - (X^6) \\ &= 1 - 2X - 2X^2 - X^3 + X^4 + X^5 .\end{aligned} \tag{11.107}$$

There are 15 forward paths in the graph of Figure 11.15(b) (see Problem 11.8), and

$$\begin{aligned}\sum_i F_i \Delta_i &= X^5(1 - X - X^2 - X^3 + X^5) + X^4(1 - X^2) + X^6 \cdot 1 + X^5(1 - X^3) \\ &\quad + X^4 \cdot 1 + X^3(1 - X - X^2) + X^6(1 - X^2) + X^6 \cdot 1 + X^5(1 - X) \\ &\quad + X^8 \cdot 1 + X^4(1 - X - X^2 - X^3 + X^4) + X^7(1 - X^3) + X^6 \cdot 1 \\ &\quad + X^3(1 - X) + X^6 \cdot 1 \\ &= 2X^3 + X^4 + X^5 + X^6 - X^7 - X^8 .\end{aligned} \tag{11.108}$$

Hence, the codeword WEF is given by

$$\begin{aligned}A(X) &= \frac{2X^3 + X^4 + X^5 + X^6 - X^7 - X^8}{1 - 2X - 2X^2 - X^3 + X^4 + X^5} \\ &= 2X^3 + 5X^4 + 15X^5 + \cdots ,\end{aligned} \tag{11.109}$$

and this code contains two nonzero codewords of weight 3, five of weight 4, fifteen of weight 5, and so on.

Additional information about the weight properties of a code can be obtained using essentially the same procedure. If the modified state diagram is augmented by labeling each branch corresponding to a nonzero information block with W^w, where w is the weight of the k information bits on that branch, and labeling each branch with L, then the *codeword input–output weight enumerating function* (IOWEF) of the code is given by

$$A(W, X, L) = \sum_{w,d,l} A_{w,d,l} W^w X^d L^l . \tag{11.110}$$

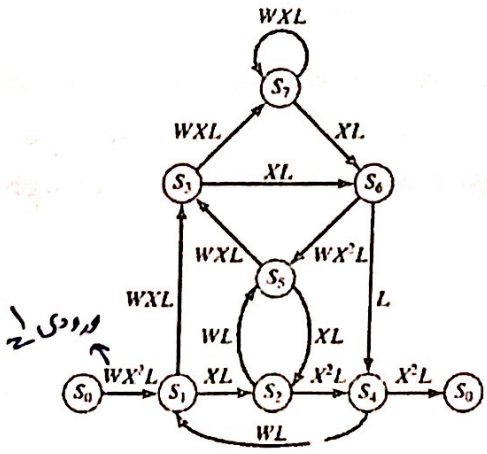


FIGURE 11.17: Augmented modified state diagram for the (2, 1, 3) encoder of Figure 11.1.

EXAMPLE 11.12 (Continued)

For the graph of Figure 11.17, it can be shown that (see Problem 11.18)

$$\begin{aligned}\Delta = 1 &- (X^8W^4L^7 + X^3W^3L^5 + X^7W^3L^6 + X^2W^2L^4 + X^4W^4L^7 + X^3W^3L^6 \\ &+ X^3WL^3 + XWL^2 + X^5W^3L^4 + X^4W^2L^3 + XWL) + (X^4W^4L^7 \\ &+ X^8W^4L^7 + X^3W^3L^6 + X^3W^3L^5 + X^4W^4L^7 + X^8W^4L^7 + X^7W^3L^6 \\ &+ X^4W^2L^4 + X^2W^2L^3 + X^5W^3L^4) - (X^4W^4L^7 + X^8W^4L^7) \\ = 1 &- XW(L + L^2) - X^2W^2(L^4 - L^3) - X^3WL^3 - X^4W^2(L^3 - L^4)\end{aligned} \tag{11.111}$$

and

$$\begin{aligned}\sum_i F_i\Delta_i &= X^{12}W^4L^8 \cdot 1 + X^7W^3L^6(1 - XWL^2) + X^{11}W^3L^7(1 - XWL) \\ &\quad + X^6W^2L^5[1 - XW(L + L^2) + X^2W^2L^3] + X^8W^4L^8 \cdot 1 \\ &\quad + X^7W^3L^7(1 - XWL) + X^7WL^4(1 - XWL - X^4W^2L^3) \\ &= X^6W^2L^5 + X^7WL^4 - X^8W^2L^5.\end{aligned} \tag{11.112}$$

Hence, the codeword IOWEF is given by

$$\begin{aligned}A(W, X, L) &= \frac{X^6W^2L^5 + X^7WL^4 - X^8W^2L^5}{1 - XW(L + L^2) - X^2W^2(L^4 - L^3) - X^3WL^3 - X^4W^2(L^3 - L^4)} \\ &= X^6W^2L^5 + X^7(WL^4 + W^3L^6 + W^3L^7) \\ &\quad + X^8(W^2L^6 + W^4L^7 + W^4L^8 + 2W^4L^9) + \cdots.\end{aligned} \tag{11.113}$$

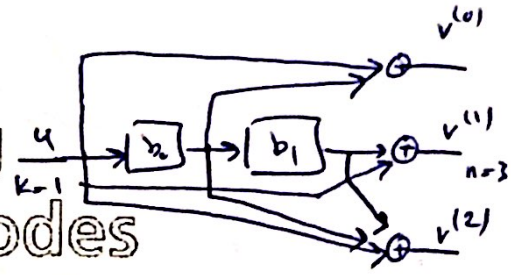
یک پیام با وزن 2

طول codeseq $2 \times 5 = 10$
وزن آن 6 است.

$\boxed{1:15 \text{ min}}$ ← تجدید شود

V80

Optimum Decoding of Convolutional Codes



THE VITERBI ALGORITHM

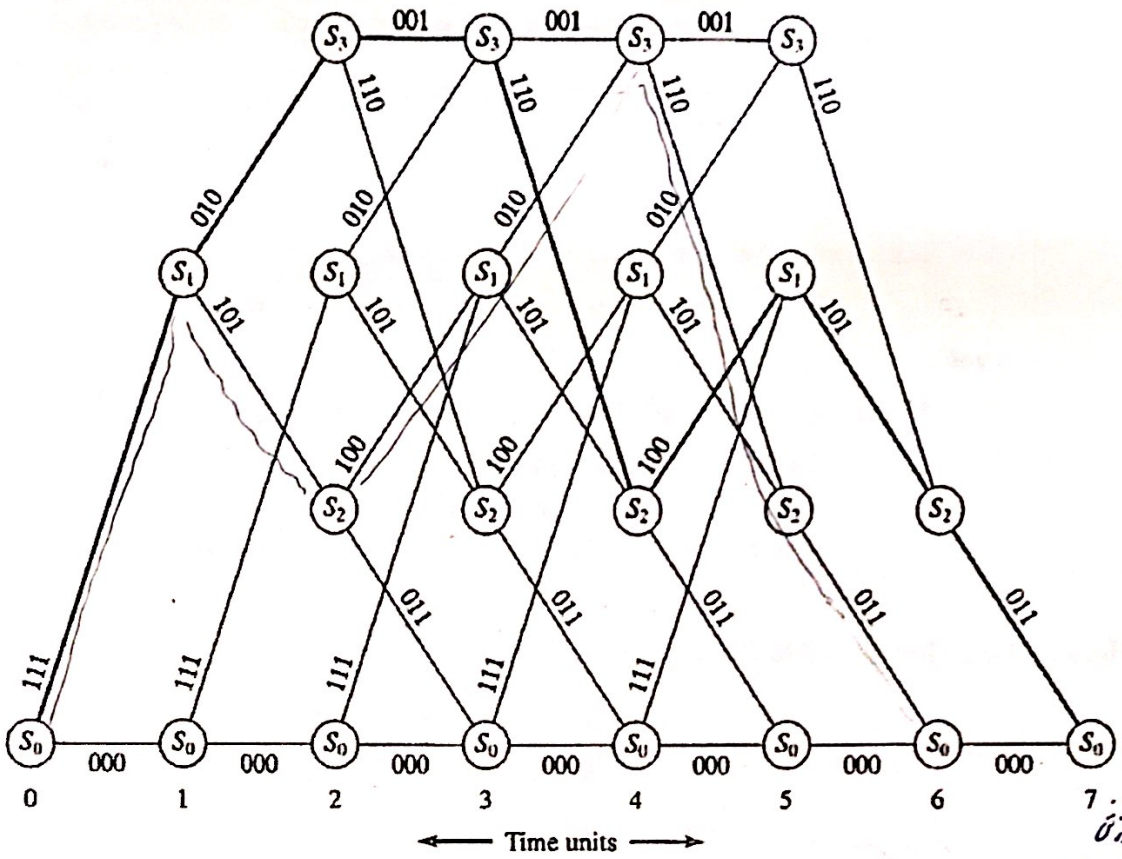
V83

b_0, b_1 : حالت دلخواه

$$\mathbf{G}(D) = [1 + D \quad 1 + D^2 \quad 1 + D + D^2]$$

00 S_0
01 S_2
10 S_1
11 S_3

and an information sequence of length $h = 5$. The trellis diagram contains $h + m + 1 = 8$ time units or levels, and these are labeled from 0 to $h + m = 7$ in Figure 12.1. For a terminated code, assuming that the encoder always starts in state S_0 and returns to state S_0, the first $m = 2$ time units correspond to the encoder's departure from state S_0, and the last $m = 2$ time units correspond to the encoder's return to state S_0. It follows that not all states can be reached in the first m or the last m time units; however, in the center portion of the trellis, all states are possible, and each time unit contains a replica of the state diagram. There are $2^k = 2$ branches leaving and entering each state. The upper branch leaving each state at time unit i represents the input bit $u_i = 1$, and the lower branch represents $u_i = 0$. Each branch is labeled with the $n = 3$ corresponding outputs $\mathbf{v}_i$, and each of the $2^h = 32$ codewords of length $N = n(h + m) = 21$ is represented by a unique path through the trellis. For example, the codeword corresponding to the information sequence $\mathbf{u} = (1\ 1\ 1\ 0\ 1)$



کم از تن

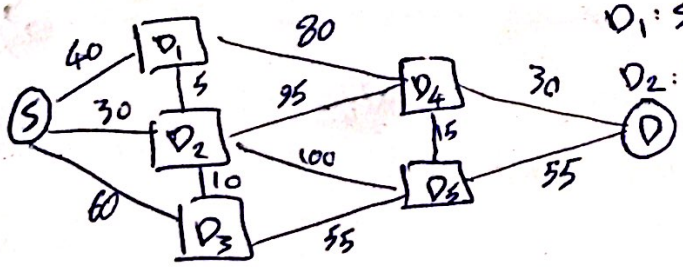
$u = D_0$ 11

FIGURE 12.1: Trellis diagram for a (3, 1, 2) encoder with $h = 5$.

الگوریتم Viterbi:

مسئله فروشنده دوره‌گرد

Comment: فرض کنید شخصی از مبدأ S به مقصد D بخواهد برود. اگر در بین مسیرهای ممکن نزدیکترین راه را انتخاب کند



D_1: S, D_2, D_1 35

D_2: S, D_2 30

D_3: S, D_2, D_3 40

D_4: D_3, D_5, D_4 110

D_5: D_4, D_5 95

برای رسیدن به D_4، D_5

D: $D_4 \to D_5$

140

Now, assume that an information sequence $\mathbf{u} = (\mathbf{u}_0, \cdots, \mathbf{u}_{h-1})$ of length $K^* = kh$ is encoded into a codeword $\mathbf{v} = (\mathbf{v}_0, \mathbf{v}_1, \cdots, \mathbf{v}_{h+m-1})$ of length $N = n(h+m)$ and that a Q-ary sequence $\mathbf{r} = (\mathbf{r}_0, \mathbf{r}_1, \cdots, \mathbf{r}_{h+m-1})$ is received over a binary-input, Q-ary output *discrete memoryless channel* (DMC). Alternatively, we can write these sequences as $\mathbf{u} = (u_0, u_1, \cdots, u_{K^*-1})$, $\mathbf{v} = (v_0, v_1, \cdots, v_{N-1})$, and $\mathbf{r} = (r_0, r_1, \cdots, r_{N-1})$, where the subscripts now simply represent the ordering of the symbols in each sequence. As discussed in Section 1.4, the decoder must produce an estimate $\hat{\mathbf{v}}$ of the codeword $\mathbf{v}$ based on the received sequence $\mathbf{r}$. A *maximum likelihood* (ML) *decoder* for a DMC chooses $\hat{\mathbf{v}}$ as the codeword $\mathbf{v}$ that maximizes the log-likelihood function $\log P(\mathbf{r}|\mathbf{v})$. Because for a DMC

k n

کل دنباله برابر N بیت باشد.

$$P(\mathbf{r}|\mathbf{v}) = \prod_{l=0}^{h+m-1} P(\mathbf{r}_l|\mathbf{v}_l) = \prod_{l=0}^{N-1} P(r_l|v_l), \tag{12.2}$$

طول پیام برابر hxk و طول بزرگترین قسمت رجیستر m

it follows that

$$\log P(\mathbf{r}|\mathbf{v}) = \sum_{l=0}^{h+m-1} \log P(\mathbf{r}_l|\mathbf{v}_l) = \sum_{l=0}^{N-1} \log P(r_l|v_l), \tag{12.3}$$

$N = n \times h + m \times n$

$h+m$

where $P(r_l|v_l)$ is a channel transition probability. This is a minimum error probability decoding rule when all codewords are equally likely.

$\log P(r|v) = d(r,v) \log \frac{P}{1-P} + N \log(1-P)$

اگر $P < \frac{1}{2}$ باشد.

The log-likelihood function $\log(\mathbf{r}|\mathbf{v})$, denoted by $M(\mathbf{r}|\mathbf{v})$, is called the *metric* associated with the path (codeword) $\mathbf{v}$. The terms $\log P(\mathbf{r}_l|\mathbf{v}_l)$ in the sum of (12.3) are called *branch metrics* and are denoted by $M(\mathbf{r}_l|\mathbf{v}_l)$, whereas the terms $\log P(r_l|v_l)$ are called *bit metrics* and are denoted by $M(r_l|v_l)$. Hence, we can write the path metric $M(\mathbf{r}|\mathbf{v})$ as

$$M(\mathbf{r}|\mathbf{v}) = \sum_{l=0}^{h+m-1} M(\mathbf{r}_l|\mathbf{v}_l) = \sum_{l=0}^{h+m-1} \log P(\mathbf{r}_l|\mathbf{v}_l) = \sum_{l=0}^{N-1} M(r_l|v_l) = \sum_{l=0}^{N-1} \log P(r_l|v_l). \tag{12.4}$$

We can now express a partial path metric for the first t branches of a path as

$$M([\mathbf{r}|\mathbf{v}]_t) = \sum_{l=0}^{t-1} M(\mathbf{r}_l|\mathbf{v}_l) = \sum_{l=0}^{t-1} \log P(\mathbf{r}_l|\mathbf{v}_l) = \sum_{l=0}^{nt-1} M(r_l|v_l) = \sum_{l=0}^{nt-1} \log P(r_l|v_l). \tag{12.5}$$

Step 1. Beginning at time unit $t = m$, compute the partial metric for the single path entering each state. Store the path (the survivor) and its metric for each state.

Step 2. Increase t by 1. Compute the partial metric for all 2^k paths entering a state by adding the branch metric entering that state to the metric of the connecting survivor at the previous time unit. For each state, compare the metrics of all 2^k paths entering that state, select the path with the largest metric (the survivor), store it along with its metric, and eliminate all other paths.

Step 3. If $t < h + m$, repeat step 2; otherwise, stop.

THEOREM 12.1 The final survivor $\hat{\mathbf{v}}$ in the Viterbi algorithm is the maximum likelihood path; that is,

$$M(\mathbf{r}|\hat{\mathbf{v}}) \geq M(\mathbf{r}|\mathbf{v}), \text{ all } \mathbf{v} \neq \hat{\mathbf{v}}. \tag{12.6}$$

Proof. Assume that the maximum likelihood path is eliminated by the algorithm at time unit t, as illustrated in Figure 12.2. This implies that the partial path metric of the survivor exceeds that of the maximum likelihood path at this point. Now, if the remaining portion of the maximum likelihood path is appended onto the survivor at time unit t, the total metric of this path will exceed the total metric of the maximum likelihood path; but this contradicts the definition of the maximum likelihood path as the path with the largest metric. Hence, the maximum likelihood path cannot be eliminated by the algorithm; that is, it must be the final survivor. **Q.E.D.**

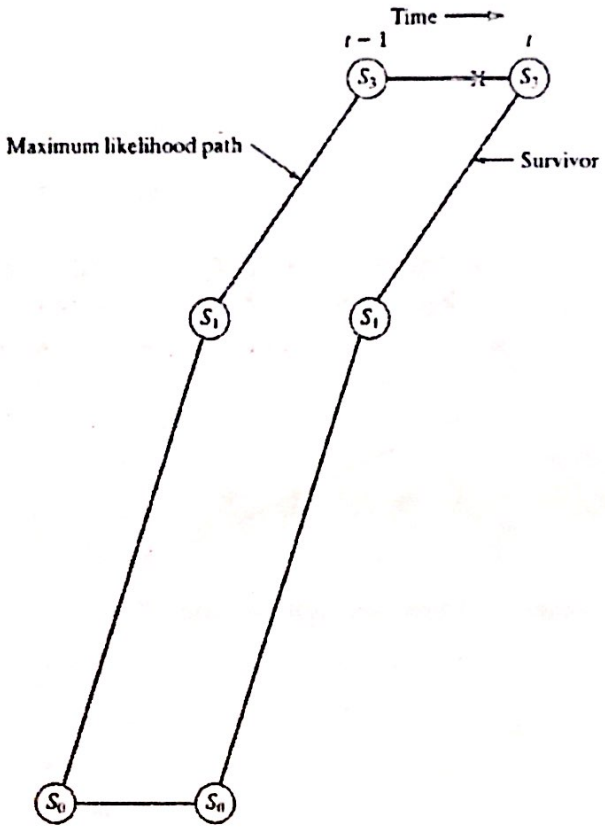


FIGURE 12.2: Elimination of the maximum likelihood path.

From an implementation point of view, it is more convenient to use positive integers as metrics rather than the actual bit metrics. The bit metric $M(r_l|v_l) = \log P(r_l|v_l)$ can be replaced by $c_2[\log P(r_l|v_l) + c_1]$, where c_1 is any real number and c_2 is any positive real number.

EXAMPLE 12.1 The Viterbi Algorithm for a Binary-Input, Quaternary-Output DMC

Consider the binary-input, quaternary-output ($Q = 4$) DMC shown in Figure 12.3. Using logarithms to the base 10, we display the bit metrics for this channel in a *metric table* in Figure 12.4(a). Choosing $c_1 = 1$ and $c_2 = 17.3$, we obtain the *integer metric table* shown in Figure 12.4(b). Now, assume that a codeword from the trellis diagram of Figure 12.1 is transmitted over the DMC of Figure 12.3 and that the quaternary received sequence is given by

$$\mathbf{r} = (1_1 1_2 0_1,\ 1_1 1_1 0_2,\ 1_1 1_1 0_1,\ 1_1 1_1 1_1,\ 0_1 1_2 0_1,\ 1_2 0_2 1_1,\ 1_2 0_1 1_1). \tag{12.7}$$

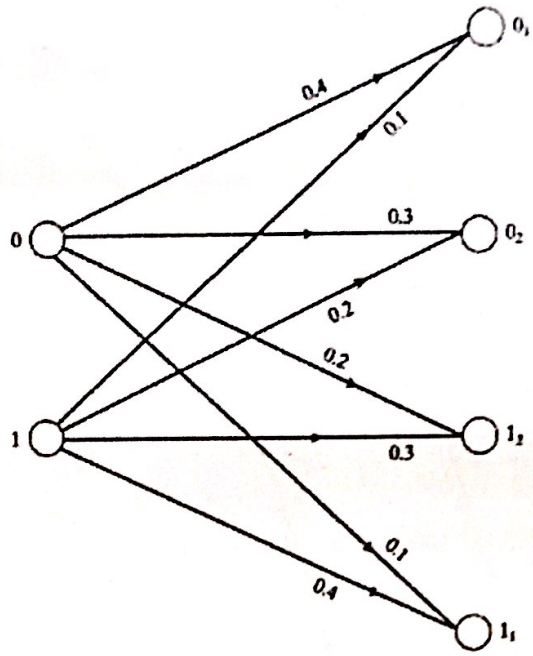


FIGURE 12.3: A binary-input, quaternary-output DMC.

v_l \ r_l	0_1	0_2	1_2	1_1
0	−0.4	−0.52	−0.7	−1.0
1	−1.0	−0.7	−0.52	−0.4

(a)

v_l \ r_l	0_1	0_2	1_2	1_1
0	10	8	5	0
1	0	5	8	10

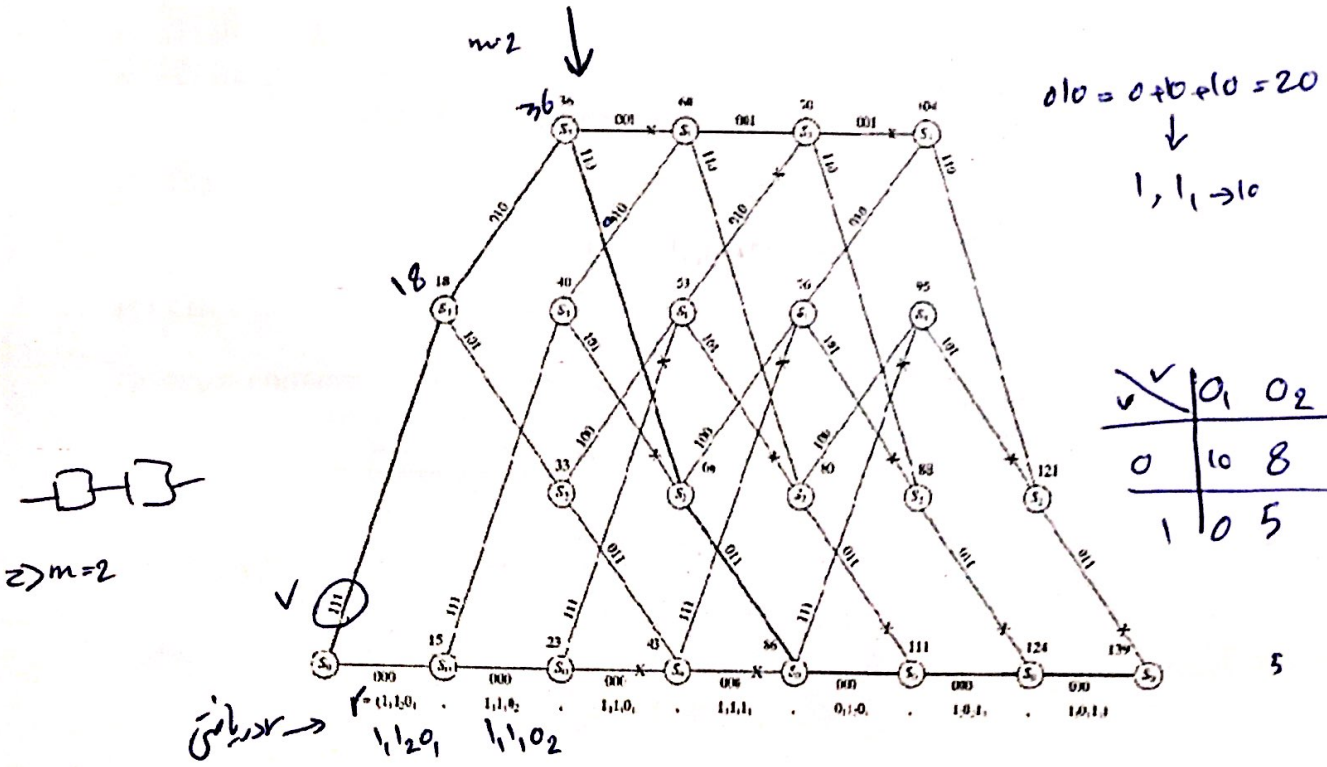
(b)

FIGURE 12.4: Metric tables for the channel of Figure 12.3.

The application of the Viterbi algorithm to this received sequence is shown in Figure 12.5. The numbers above each state represent the metric of the survivor for that state, and the paths eliminated at each state are shown crossed out on the trellis diagram. The final survivor,

$$\hat{\mathbf{v}} = (111, 010, 110, 011, 000, 000, 000), \tag{12.8}$$

is shown as the highlighted path. This surviving path corresponds to the decoded information sequence $\hat{\mathbf{u}} = (1\,1\,0\,0\,0)$. Note that the final $m = 2$ branches in any trellis path correspond to 0 inputs and hence are not considered part of the information sequence.



v \ r	0_1	0_2	1_2	1_1
0	10	8	5	0
1	0	5	8	10

In the special case of a *binary symmetric channel* (BSC) with transition probability $p < 1/2$,[1] the received sequence $\mathbf{r}$ is binary ($Q = 2$) and the log-likelihood function becomes (see (1.11))

$$\underbrace{\log P(\mathbf{r}|\mathbf{v})}_{\text{metric}} = d(\mathbf{r}, \mathbf{v}) \log \frac{p}{1-p} + N \log(1-p), \tag{12.9}$$

where $d(\mathbf{r}, \mathbf{v})$ is the Hamming distance between $\mathbf{r}$ and $\mathbf{v}$. Because $\log \frac{p}{1-p} < 0$ and $N \log(1-p)$ is a constant for all $\mathbf{v}$, an MLD for a BSC chooses $\mathbf{v}$ as the codeword $\hat{\mathbf{v}}$ that minimizes the Hamming distance

$$d(\mathbf{r}, \mathbf{v}) = \sum_{l=0}^{h+m-1} d(\mathbf{r}_l, \mathbf{v}_l) = \sum_{l=0}^{N-1} d(r_l, v_l). \tag{12.10}$$

Hence, when we apply the Viterbi algorithm to the BSC, $d(\mathbf{r}_l, \mathbf{v}_l)$ becomes the branch metric, $d(r_l, v_l)$ becomes the bit metric, and the algorithm must find the path through the trellis with the smallest metric, that is, the path closest to $\mathbf{r}$ in Hamming distance. The operation of the algorithm is exactly the same, except that the Hamming distance replaces the log-likelihood function as the metric, and the survivor at each state is the path with the *smallest* metric.

EXAMPLE 12.2 The Viterbi Algorithm for a BSC

An example of the application of the Viterbi algorithm to a BSC is shown in Figure 12.6. Assume that a codeword from the trellis diagram of Figure 12.1 is transmitted over a BSC and that the received sequence is given by

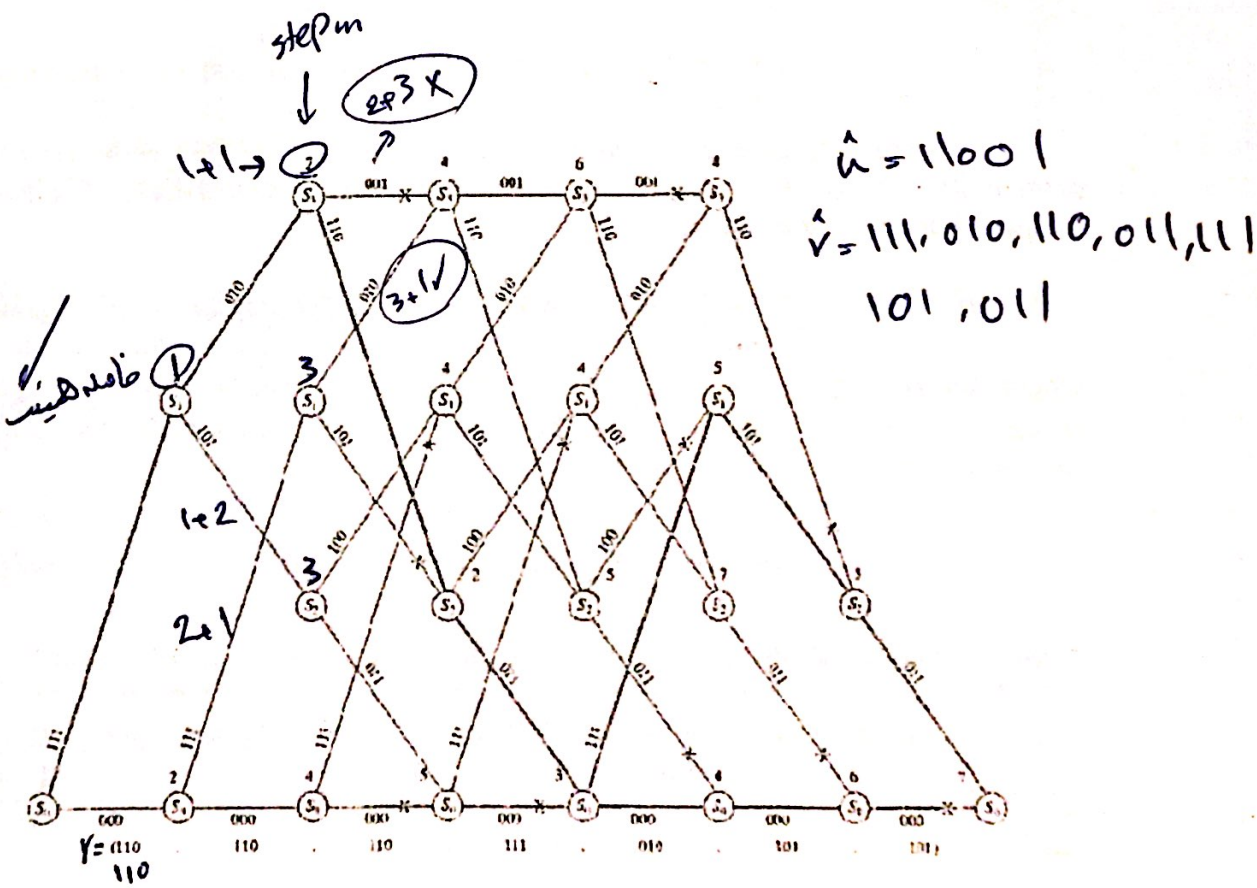
$$\mathbf{r} = (110, 110, 110, 111, 010, 101, 101). \tag{12.11}$$

The final survivor,

در کد قالبی امکان نداشت

$$\hat{\mathbf{v}} = (111, 010, 110, 011, \overline{111}, 101, 011). \tag{12.12}$$

is shown as the highlighted path in the figure, and the decoded information sequence is $\hat{\mathbf{u}} = (11001)$.



Now, consider a binary-input, additive white Gaussian noise (AWGN) channel with no demodulator output quantization, that is, a binary-input, continuous-output channel. Assume the channel inputs 0 and 1 are represented by the BPSK signals (see (1.1)) $\pm\sqrt{\frac{2E_s}{T}}\cos(2\pi f_0 t)$, where we use the mapping $1 \to +\sqrt{E_s}$ and $0 \to -\sqrt{E_s}$.

Normalizing by $\sqrt{E_s}$, we consider the codeword $\mathbf{v} = (v_0, v_1, \cdots, v_{N-1})$ to take on values ± 1 according to the mapping $1 \to +1$ and $0 \to -1$, and the (normalized) received sequence $\mathbf{r} = (r_0, r_1, \cdots, r_{N-1})$ to be real-valued (unquantized). The conditional probability density function (pdf) of the (normalized) received symbol r_l given the transmitted bit v_l is

$$p(r_l|v_l) = \sqrt{\frac{E_s}{\pi N_0}} e^{\left[-\frac{(r_l\sqrt{E_s} - v_l\sqrt{E_s})^2}{N_0}\right]} = \sqrt{\frac{E_s}{\pi N_0}} e^{-(E_s/N_0)(r_l - v_l)^2}, \tag{12.13}$$

where N_0/E_s is the (normalized) one-sided noise power spectral density. If the channel is *memoryless*, the log-likelihood function of the received sequence $\mathbf{r}$ given the transmitted codeword $\mathbf{v}$ is

$$\begin{aligned}
M(\mathbf{r}|\mathbf{v}) &= \ln p(\mathbf{r}|\mathbf{v}) = \ln \prod_{l=0}^{N-1} p(r_l|v_l) = \sum_{l=0}^{N-1} \ln p(r_l|v_l) \\
&= -\frac{E_s}{N_0}\sum_{l=0}^{N-1}(r_l - v_l)^2 + \frac{N}{2}\ln\frac{E_s}{\pi N_0} \quad \circledast \\
&= -\frac{E_s}{N_0}\sum_{l=0}^{N-1}(r_l^2 - 2r_l v_l + 1) + \frac{N}{2}\ln\frac{E_s}{\pi N_0} \\
&= (2\frac{E_s}{N_0})\sum_{l=0}^{N-1}(r_l v_l) - \frac{E_s}{N_0}(|\mathbf{r}|^2 + N) + \frac{N}{2}\ln\frac{E_s}{\pi N_0} \\
&= C_1(\mathbf{r}\cdot\mathbf{v}) + C_2.
\end{aligned} \tag{12.14}$$

برای max کردن، ⊛ چون ثابت است
⇐ تاثیری در max کردن ندارد

where $C_1 = (2E_s/N_0)$ and $C_2 = -[(E_s/N_0)(|\mathbf{r}|^2 + N) - (N/2)\ln(E_s/\pi N_0)]$ are constants independent of the codeword $\mathbf{v}$ and $\mathbf{r}\cdot\mathbf{v}$ represents the inner product (correlation) of the received vector $\mathbf{r}$ and the codeword $\mathbf{v}$. Because C_1 is positive, the trellis path (codeword) that maximizes the correlation $\mathbf{r}\cdot\mathbf{v}$ also maximizes the log-likelihood function $\ln p(\mathbf{r}|\mathbf{v})$. It follows that the path metric corresponding to the codeword $\mathbf{v}$ is given by $M(\mathbf{r}|\mathbf{v}) = \mathbf{r}\cdot\mathbf{v}$, the branch metrics are $M(\mathbf{r}_l|\mathbf{v}_l) = \mathbf{r}_l\cdot\mathbf{v}_l$, $l = 0, 1, \cdots, h+m-1$, the bit metrics are $M(r_l|v_l) = r_l v_l$, $l = 0, 1, \cdots, N-1$, and the Viterbi algorithm finds the path through the trellis that maximizes the correlation with the received sequence. (It is important to remember in this case that the (real-valued) received sequence $\mathbf{r}$ is correlated with trellis paths (codewords) $\mathbf{v}$ labeled according to the mapping $1 \to +1$ and $0 \to -1$.)

مهم*

بعد از این مدوسل بالا

$$P(E) < \sum_{d=d_{free}}^{\infty} A_d P_d < \sum_{d=d_{free}}^{\infty} A_d \left[2\sqrt{p(1-p)}\right]^d = A(X)|_{X=2\sqrt{p(1-p)}}, \quad (12.25)$$

just as in (12.23). For small p, the bound is dominated by its first term, that is, the free distance term, and the event-error probability can be approximated as

$$P(E) \approx A_{d_{free}} \left[2\sqrt{p(1-p)}\right]^{d_{free}} \approx A_{d_{free}} 2^{d_{free}} p^{d_{free}/2}. \quad (12.26)$$

Example:

$$P_b(E) < \sum_{d=d_{free}}^{\infty} B_d P_d < \sum_{d=d_{free}}^{\infty} B_d \left[2\sqrt{p(1-p)}\right]^d = B(X)|_{X=2\sqrt{p(1-p)}} \quad (12.29)$$

for any convolutional encoder with bit WEF $B(X)$. (We note here that the bound of (12.25) on the event-error probability $P(E)$ depends only on the code, whereas the bound of (12.29) on the bit-error probability $P_b(E)$ depends on the encoder.) For small p, the bound of (12.29) is dominated by its first term, so that

$$P_b(E) \approx B_{d_{free}} \left[2\sqrt{p(1-p)}\right]^{d_{free}} \approx B_{d_{free}} 2^{d_{free}} p^{d_{free}/2}. \quad (12.30)$$

EXAMPLE 12.4 Evaluating the Bit-Error Probability

For the (3, 1, 2) encoder of (12.1), $B_{d_{free}} = 1$, and $d_{free} = 7$. Thus, for $p = 10^{-2}$, we obtain

$$P_b(E) \approx 2^7 p^{7/2} = 1.28 \times 10^{-5}, \quad (12.31)$$

the same as for the event-error probability. In other words, when p is small, the most likely error event is that the weight-7 path is decoded instead of the all-zero path, thereby causing one information bit error. Typically, then, each event error causes one bit error, which is reflected in the approximate expressions for $P(E)$ and $P_b(E)$.

Slightly tighter versions of the bounds on $P(E)$ and $P_b(E)$ given in (12.25) and (12.29), respectively, have been derived in [11] and [13].

If the BSC is derived from an AWGN channel with BPSK modulation, optimum coherent detection, and binary-output quantization (hard decisions), then from (1.4) we have

$$p = Q\left(\sqrt{\frac{2E_s}{N_0}}\right). \tag{12.32}$$

Using the bound of (1.5) as an approximation yields

$$p \approx \tfrac{1}{2}e^{-E_s/N_0}, \tag{12.33}$$

and for a convolutional code with free distance d_{free},

$$P_b(E) \approx B_{d_{free}} 2^{d_{free}/2} e^{-(d_{free}/2)(E_s/N_0)} \tag{12.34}$$

when p is small, that is, when the channel SNR E_s/N_0 is large. Defining the *energy per information bit*, E_b, as

$$E_b \triangleq \frac{E_s}{R}, \tag{12.35}$$

and noting that $\frac{1}{R}$ is the number of transmitted symbols per information bit, we can write

$$P_b(E) \approx B_{d_{free}} 2^{d_{free}/2} e^{-(Rd_{free}/2)(E_b/N_0)} \quad \text{(with coding)} \tag{12.36}$$

for large bit SNR E_b/N_0.

On the other hand, if no coding is used, that is, $R = 1$ and $E_s = E_b$, the BSC transition probability p is the bit-error probability $P_b(E)$ and

$$P_b(E) = Q\left(\sqrt{\frac{2E_b}{N_0}}\right) \approx \frac{1}{2}e^{-E_b/N_0} \quad \text{(without coding).} \tag{12.37}$$

Comparing (12.36) and (12.37), we see that for a fixed E_b/N_0 ratio, the (negative) exponent with coding is larger by a factor of $Rd_{free}/2$ than the exponent without coding. Because the exponential term dominates the error probability expressions for large E_b/N_0, the factor $Rd_{free}/2$, in decibels, is called the *asymptotic coding gain* γ:

$$\gamma \triangleq 10 \log_{10} \left(\frac{Rd_{free}}{2} \right) \text{ dB} \tag{12.38}$$

in the hard-decision case.

It is instructive to compare the approximate expression for $P_b(E)$ given in (12.36) for a BSC with a similar expression obtained for a binary-input, continuous-output AWGN channel from (12.45b). For large E_b/N_0, the first term in the bit WEF dominates the bound of (12.45b), and we can approximate $P_b(E)$ as

$$P_b(E) \approx B_{d_{free}} (e^{-RE_b/N_0})^{d_{free}} = B_{d_{free}} e^{-Rd_{free}E_b/N_0}. \tag{12.46}$$

Comparing the exponent of (12.46) with that of (12.36), we see that the exponent of (12.46) is larger by a factor of 2. This difference is equivalent to a 3-dB energy (or power) advantage for the continuous-output AWGN channel over the BSC, since to achieve the same error probability on the BSC, the transmitter must generate an additional 3 dB of signal energy (or power). This energy advantage illustrates the benefits of allowing soft decisions, that is, an unquantized demodulator output, instead of making hard decisions. The asymptotic coding gain in the soft-decision case is given by

$$\gamma \triangleq 10 \log_{10} (Rd_{free}) \text{ dB}, \tag{12.47}$$

an increase of 3 dB over the hard-decision case. The decoder complexity increases, however, owing to the need to accept real-valued inputs.

.....

Now, defining the function

$$f(x) = Q(\sqrt{2x})e^{x}, \tag{12.50}$$

we can write

$$P_d \le f\left(\frac{d_{free}RE_b}{N_0}\right)e^{-\frac{dRE_b}{N_0}}. \tag{12.51}$$

Finally, substituting (12.51) into the bounds of (12.25) and (12.29), we obtain the following expressions for the event- and bit-error probabilities of a binary-input, continuous-output AWGN channel:

$$\begin{aligned} P(E) &< \sum_{d=d_{free}}^{\infty} A_d f\left(\frac{d_{free}RE_b}{N_0}\right)e^{-\frac{dRE_b}{N_0}} \\ &= f\left(\frac{d_{free}RE_b}{N_0}\right)A(X)|_{X=\exp(-RE_b/N_0)} \end{aligned} \tag{12.52a}$$

$$\begin{aligned} P_b(E) &< \sum_{d=d_{free}}^{\infty} B_d f\left(\frac{d_{free}RE_b}{N_0}\right)e^{-\frac{dRE_b}{N_0}} \\ &= f\left(\frac{d_{free}RE_b}{N_0}\right)B(X)|_{X=\exp(-RE_b/N_0)}. \end{aligned} \tag{12.52b}$$

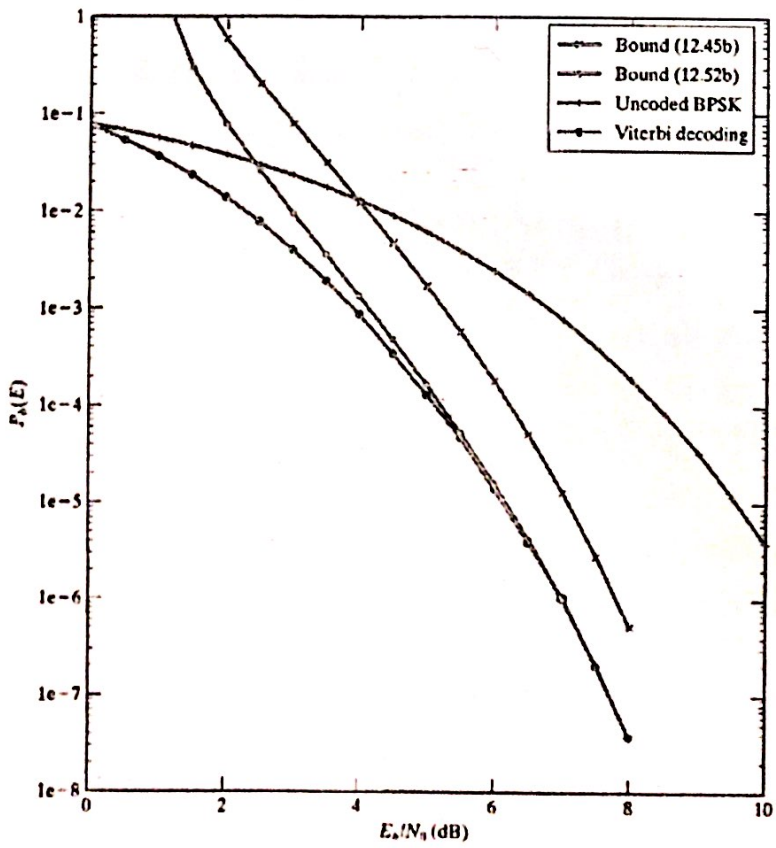


FIGURE 12.11: Performance bounds for convolutional codes.