ECE 405/511
Error Control Coding

Cyclic Codes

Definition

- A code C is cyclic if

  1) C is a linear block code

  2) a cyclic shift of any codeword

$$\mathbf{c}_i = (c_0, c_1, \cdots, c_{n-1})$$

  is another codeword

$$\mathbf{c}_j = (c_{n-1}, c_0, c_1, \cdots, c_{n-2})$$

- Examples:

  $C_1 = \{000, 111\}$

  $C_2 = \{000, 101, 011, 110\}$

Another Example

- C_3 = {0000,1001,0110,1111} is not cyclic

- Interchange positions 3 and 4
  (equivalent code)

- $C_{3'}$ = {0000,1010,0101,1111} is cyclic

- Code polynomials

$$c(x) = c_0 + c_1 x + \cdots + c_{n-1} x^{n-1}, \quad c_i \in \mathrm{GF}(q)$$

- GF(q)[x] is the set of polynomials with coefficients from GF(q)
- GF(q)[x] is a commutative ring with identity (not a field)

- Define the ring of polynomials modulo $f(x)$ of degree n as $GF(q)[x]/f(x)$
- This is a finite ring
- Example: choose $f(x)=x^2-1$ which in GF(2) is x^2+1
  - then the ring is $GF(2)[x]/(x^2+1)$
  - x^2+1 is not irreducible
  - elements are $\{0, 1, x, x+1\}$

- Over any field GF(q)

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

  so x^n-1 is never irreducible for n>1
- Let R_n denote GF(2)[x]/(x^n-1)
- Any polynomial of degree ≥ n can be reduced modulo x^n-1 to a polynomial of degree less than n

$$x^n \rightarrow 1$$
$$x^{n+1} \rightarrow x$$
$$x^{n+2} \rightarrow x^2$$

ECE 405/511 Test

- Friday, February 17, 2023 10:30 AM
  - constitutes 20% of the final grade
- Test will cover material up to bounds on codes
- Shortening and extending are included but not the Hamming, Gilbert, and Gilbert-Varshamov bounds.
  - Moreira and Farrell Chapter 2 (not Section 2.11)
  - Assignments 1 and 2 (Problems 1-4)
- Aids allowed: 1 sheet of paper 8.5×11 in^2
  calculator

Ideals

- Let R be a ring. A nonempty subset $I \subseteq R$ is called an Ideal if it satisfies the following
  - I forms a group under addition
  - $a \bullet r \in I$ for all $a \in I$ and $r \in R$
    - superclosed under multiplication
- Examples
  - {0} and R are trivial Ideals in R
  - $\{0, x^4+x^3+x^2+x+1\}$ is an Ideal in $R_5 = GF(2)[x]/(x^5-1)$
  - even numbers in Z (even integers)

Ideal Example

- R_3 = GF(2)[x]/(x^3-1)

0	$\to 000$	1	$\to 100$
x	$\to 010$	$1+x$	$\to 110$
x^2	$\to 001$	$1+x^2$	$\to 101$
$x+x^2$	$\to 011$	$1+x+x^2$	$\to 111$

$I = \{0, 1+x, 1+x^2, x+x^2\}$ is an Ideal in R_3

{000, 110, 101, 011} is a cyclic code

Theorem

A code which is a vector subspace over a field GF(q) is a cyclic code iff it corresponds to an ideal in GF(q)[x]/(x^n-1) (the ring of polynomials modulo x^n-1)

Cyclic Code Generation

- Let $f(x)$ be any polynomial in R_n and let $< f(x) >$ denote the subset of R_n consisting of all multiples of $f(x)$ modulo x^n-1

$$< f(x) > = \{r(x)f(x) \mid r(x) \in R_n\}$$

- $< f(x) >$ is the cyclic code generated by $f(x)$
- Example: $C = < 1+x^2 >$ in R_3 = GF(2)[x]/(x^3-1)
  - Multiplying by all 8 elements in R_3 produces only 4 distinct codewords

$$C=\{0,1+x,1+x^2,x+x^2\}$$

Generator Polynomial

- Any cyclic code can be generated by a polynomial from R_n
- Let C be a cyclic code in R_n. Then we have the following facts:
  1. There exists a unique monic polynomial $g(x)$ of smallest degree in C
  2. $C = < g(x) >$
  3. $g(x) | x^n - 1$

  $g(x)$ is called the generator polynomial of the cyclic code

Cyclic Codes

- Any polynomial $c(x)$ of degree less than n is in C iff $g(x)|c(x)$
- If $g(x)$ has degree n-k, $|C|=q^k$
- Every codeword has the form

$$c(x) = m(x)g(x)$$

codeword polynomial of degree n-1 or less

message polynomial of degree k-1 or less

generator polynomial of degree n-k

- To determine the possible $g(x)$, factor x^n-1
- Example:

$$x^3-1 = (x+1)(x^2+x+1) \text{ over GF}(2)$$

Generator polynomial	Code in R_3	Code in 3-tuples
1	R_3	V_3
$x+1$	$\{0,1+x,1+x^2,x+x^2\}$	{000,110,101,011}
x^2+x+1	$\{0,1+x+x^2\}$	{000,111}
x^3-1	$\{0\}$	{000}

Generator Matrix

- Since $$c(x) = m(x)g(x) = (m_0 + m_1 x + \cdots + m_{k-1}x^{k-1})g(x)$$
$$= m_0 g(x) + m_1 x g(x) + \cdots + m_{k-1}x^{k-1}g(x)$$
$$= [m_0 \quad m_1 \quad \cdots \quad m_{k-1}] \begin{bmatrix} g(x) \\ xg(x) \\ \vdots \\ x^{k-1}g(x) \end{bmatrix} = \mathbf{mG}$$

$$\mathbf{G} = \begin{bmatrix} g_0 & g_1 & \cdots & g_{n-k} & & & \mathbf{0} \\ & g_0 & g_1 & \cdots & g_{n-k} & & \\ & \ddots & \ddots & & \ddots & & \\ & & g_0 & g_1 & \cdots & g_{n-k} & \\ \mathbf{0} & & & g_0 & g_1 & \cdots & g_{n-k} \end{bmatrix}$$

is a generator matrix for the cyclic code

Generator Matrix Example

- R_7 = GF(2)[x]/(x^7-1)
- $x^7-1 = (1+x+x^3)(1+x^2+x^3)(1+x)$
- $g(x) = 1+x+x^3$

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

- *C* is a (7,4,3) code – a binary cyclic code
- All binary cyclic codes with $g(x)$ a primitive polynomial are equivalent to Hamming codes

Example

- $g(x) = (1+x+x^3)(1+x) = 1+x^2+x^3+x^4$

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

- C is a (7,3,4) binary cyclic code

Parity Check Matrix

- The generator matrix is not in systematic form.
  How to find the parity check matrix?
- $g(x)$ is a factor of x^n-1, i.e. $g(x)h(x) = x^n$-1
- $h(x)$ is a monic polynomial with degree k, and is the generator polynomial of a cyclic code C', but not necessarily of the dual code of C.
- For the (7,4,3) code example

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

- $x^7-1 = (1+x+x^3)(1+x^2+x^3)(1+x)$
- $g(x) = 1+x+x^3$

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

- $h(x) = (1+x^2+x^3)(1+x) = 1+x+x^2+x^4$

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

- $g(x)h(x)=0 \bmod x^n-1$ (in R_n) is not the same as vectors in V_n being orthogonal.
- Let **H** be the matrix generated from

  $h^*(x)=x^k h(x^{-1})=h_k+xh_{k-1}+\ldots+x^k h_0$ reciprocal poly. of $h(x)$

$$
\mathbf{H}=\begin{bmatrix}
h_k & h_{k-1} & \cdots & h_1 & h_0 & & & 0 \\
 & h_k & h_{k-1} & \cdots & h_1 & h_0 & & \\
 & & \ddots & \ddots & & \ddots & \ddots & \\
 & & & h_k & h_{k-1} & \cdots & h_1 & h_0 \\
0 & & & & h_k & h_{k-1} & \cdots & h_1 \; h_0
\end{bmatrix}
$$

Parity Check Matrix $\mathbf{H}$

- $c(x)h(x) = m(x)g(x)h(x) = m(x)(x^n\text{-}1) = \text{-}m(x)+ x^n m(x)$
- $m(x)$ has degree < k, thus the coefficients of x^k to $x^{n\text{-}1}$ in $c(x)h(x)$ must be zero

$$
\left.
\begin{aligned}
& c_0 h_k + c_1 h_{k-1} + \cdots + c_k h_0 = 0 \\
& c_1 h_k + c_2 h_{k-1} + \cdots + c_{k+1} h_0 = 0 \\
& \vdots \\
& c_{n-k-1} h_k + c_{n-k} h_{k-1} + \cdots + c_{n-1} h_0 = 0
\end{aligned}
\right.
\quad \Rightarrow \quad \mathbf{cH}^{\mathrm{T}} = \mathbf{0}
$$

Hamming Code Example (Cont.)

- $h^*(x) = 1+x^2+x^3+x^4$ generates the parity check matrix and the dual cyclic code of the code generated by $g(x)$

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

- **H** is the parity check matrix for the (7,4,3) Hamming code
- $h^*(x)=1+x^2+x^3+x^4$ is the generator polynomial for a (7,3,4) cyclic code since $h^*(x) | x^n-1$

Hamming Code Example (Cont.)

- $h^*(x)=1+x^2+x^3+x^4$ is the generator polynomial for a (7,3,4) cyclic code since $h^*(x) | x^n-1$
- $g(x)=1+x^2+x^3+x^4$

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

Hamming Code Example (Cont.)

- To construct the parity check matrix for the (7,3,4) code, use $h(\mathrm{x}) = 1+x^2+x^3$
- $h^*(x) = 1+x+x^3$ is the generator polynomial for a (7,4,3) cyclic code since $h^*(x)|x^n\text{-}1$
- $h^*(x)$ generates the parity check matrix **H** as well as the dual cyclic code

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

Binary Cyclic Codes of Length 7

- x^7-1=$(1+x+x^3)(1+x^2+x^3)(1+x)$

- $g(x) = 1+x$ (7,6,2)

  dual code $h^*(x) = 1+x+x^2+x^3+x^4+x^5+x^6$ (7,1,7)
- $g(x) = 1+x+x^3$ (7,4,3)

  dual code $h^*(x) = 1+x^2+x^3+x^4$ (7,3,4)
- $g(x) = 1+x^2+x^3$ (7,4,3)

  dual code $h^*(x) = 1+x+x^2+x^4$ (7,3,4)

Systematic Cyclic Codes

- $R_7 = GF(2)[x]/(x^7-1)$
- $x^7-1 = (1+x+x^3)(1+x^2+x^3)(1+x)$
- $g(x) = 1+x+x^3$

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

- C is a (7,4,3) code – not in systematic form
- To transform into systematic form:
  - permute columns 1 and 4, then add rows 2 and 4 to get a new row 4

Systematic Generator Matrix

- Permute columns 1 and 4, then add rows 2 and 4 to get a new row 4.
- The resulting generator matrix has a systematic form [**P** $\mathbf{I}_k$], but is not cyclic

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

- Check: divide the last row of **G**' by $g(x)$
  - $1+x+x^2+x^6$ is not divisible by $g(x) = 1+x+x^3$

Systematic Generator Matrix

- We require an algebraic means of generating a systematic code while preserving divisibility by $g(x)$.
- Approach: divide x^i by $g(x)$, $i = n\text{-}k$ to $n\text{-}1$

  $x^i = g(x)q_i(x)+d_i(x)$ $d_i(x)$ has degree less than $n\text{-}k$

  rearranging $x^i - d_i(x) = g(x)q_i(x)$ divisible by $g(x)$
- $x^i - d_i(x)$ has only one non-zero coefficient for degrees $n\text{-}k$ to $n\text{-}1$
- Use $x^i - d_i(x)$ to form **G**

$$\mathbf{G} = [\mathbf{P}\ \ \mathbf{I}_k] \qquad \mathbf{H} = [\mathbf{I}_{n-k}\ \ -\mathbf{P}^{\mathrm{T}}]$$

Example

- $g(x) = 1+x+x^3$

x^i	$g(x)q_i(x)$	$d_i(x)$	$x^i + d_i(x)$
x^3	$(1+x+x^3)\cdot 1$	$1+x$	$1+x+x^3$
x^4	$(1+x+x^3)\cdot x$	$x+x^2$	$x+x^2+x^4$
x^5	$(1+x+x^3)\cdot(1+x^2)$	$1+x+x^2$	$1+x+x^2+x^5$
x^6	$(1+x+x^3)\cdot(1+x+x^3)$	$1+x^2$	$1+x^2+x^6$

$$\mathbf{G} = \left[\begin{array}{ccc|cccc} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{array}\right]$$

Systematic Encoding

- Systematic encoding is achieved by multiplying $m(x)$ by x^{n-k} and dividing this product by $g(x)$ to obtain $d(x)$

- $c(x) = m(x)x^{n-k} + m(x)x^{n-k}/g(x)$

  use the remainder $d(x)$

- Example (7,4,3) code

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

  $m(x)x^{n-k} = x^5+x^4+x^3$ divide by $g(x) = x^3+x+1 \longrightarrow d(x) = x$

  $c(x) = x^5+x^4+x^3+x$

  c = 0101110

Another Example

- R_{23} = GF(2)[x]/(x^{23}-1)
- x^{23}-1 =
  $(x+1)(x^{11}+x^{10}+x^6+x^5+x^4+x^2+1)(x^{11}+x^9+x^7+x^6+x^5+x+1)$
  $= (x+1)g_1(x)g_2(x)$
- $g_1(x) = g_2{}^*(x)$
- $g_1(x)$ and $g_2(x)$ both generate (23,12,7) codes

Ternary Example

- $x^{11}-1 = (x-1)(x^5+x^4-x^3+x^2-1)(x^5-x^3+x^2-x-1)$ over GF(3)
  $= (x-1)g_1(x)g_2(x)$
- $g_1(x)$ and $g_2(x)$ both generate (11,6,5) codes

Implementation of Cyclic Codes

- Encoding
  - in non-systematic form: $c(x) = m(x)g(x)$
  - in systematic form: $c(x) = m(x)x^{n-k}+d(x)$

    $d(x)$ is the remainder of $m(x)x^{n-k}/g(x)$

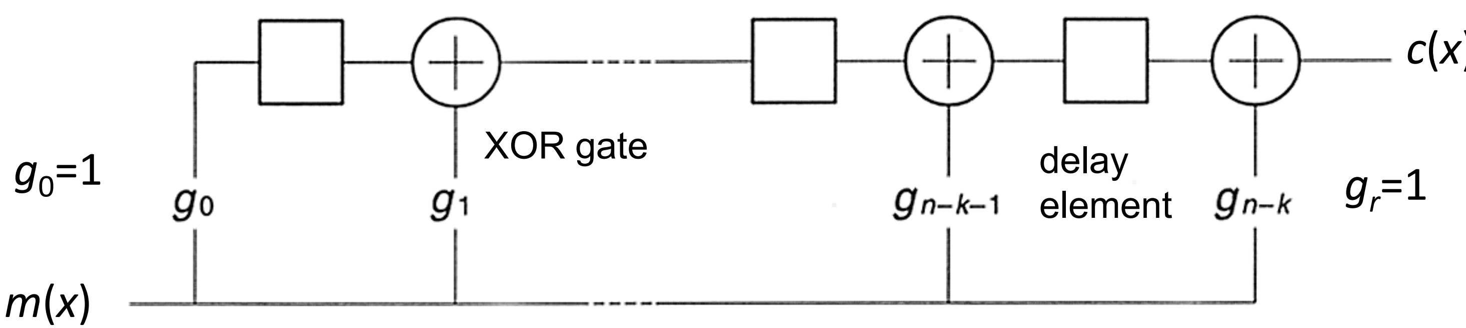
- Thus we require circuits for multiplying and dividing polynomials
- Solution: use shift registers

Nonsystematic Binary Cyclic Code Encoder

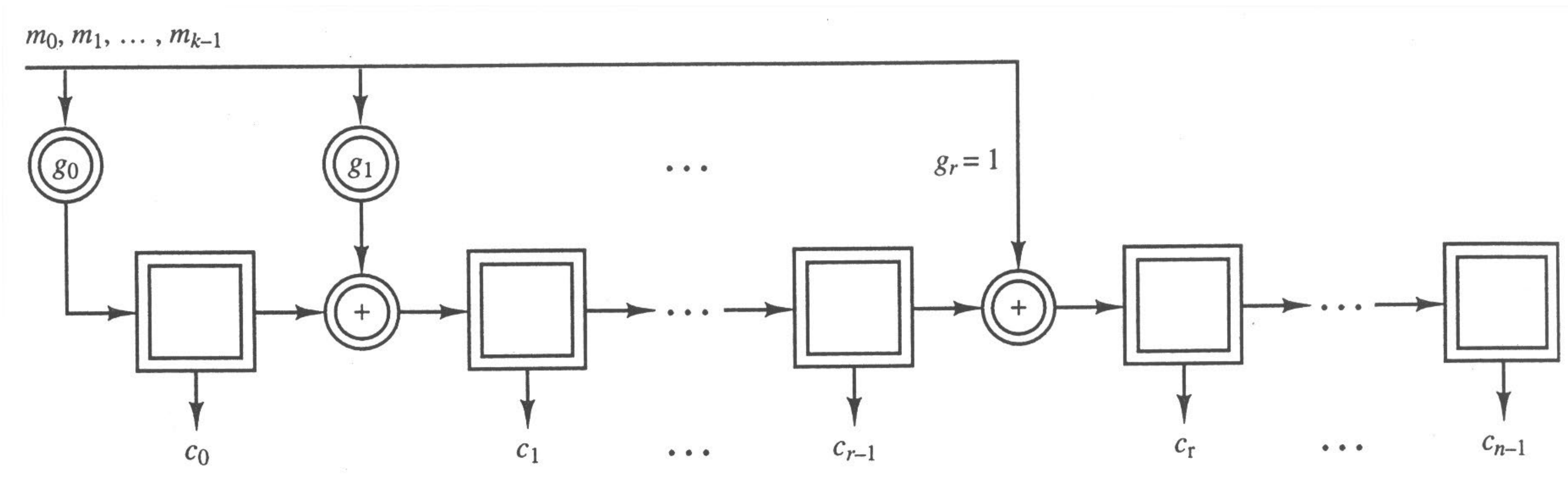
- Encoding can be done by multiplying two polynomials
  - a message polynomial $m(x)$ and the generator polynomial $g(x)$
- The generator polynomial is

$$g(x) = g_0 + g_1 x + \dots + g_r x^r \qquad \text{of degree } r = n\text{-}k$$

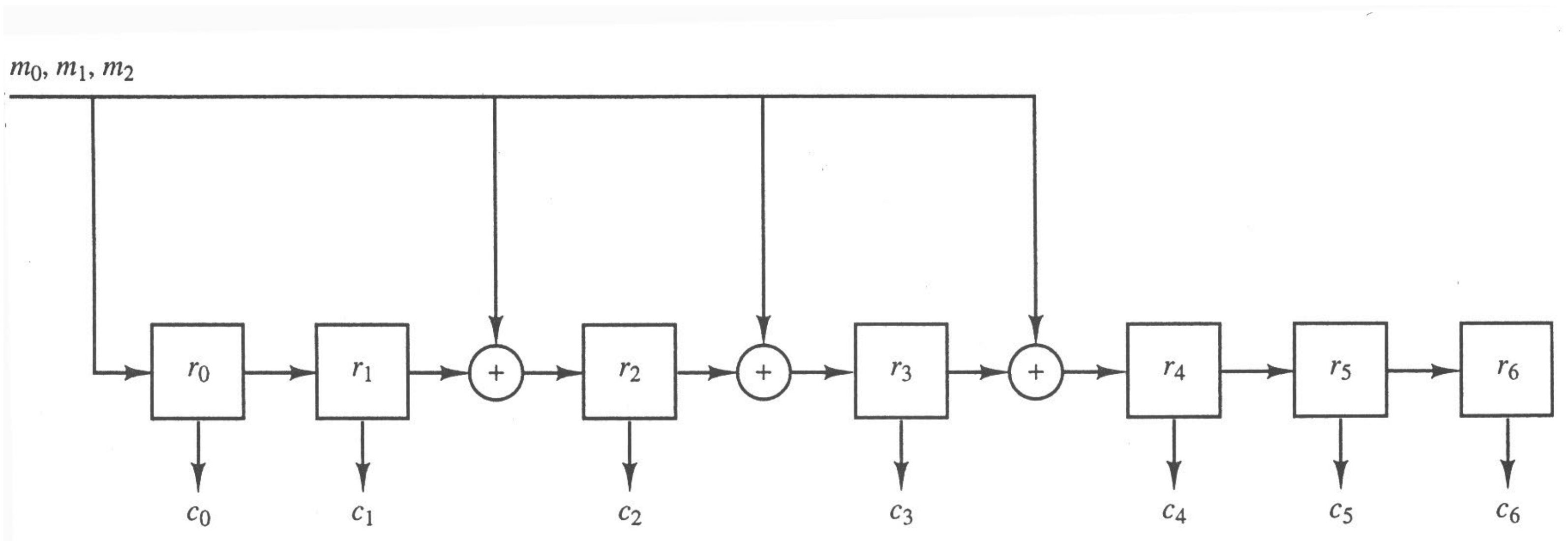
- If a message vector m is represented by a polynomial $m(x)$ of degree k-1, $m(x)$ is encoded as $c(x) = m(x)g(x)$ using the following shift register circuit



Nonsystematic Shift Register Encoder



Encoder for the (7,3) Binary Cyclic Code with $g(x) = 1+x^2+x^3+x^4$



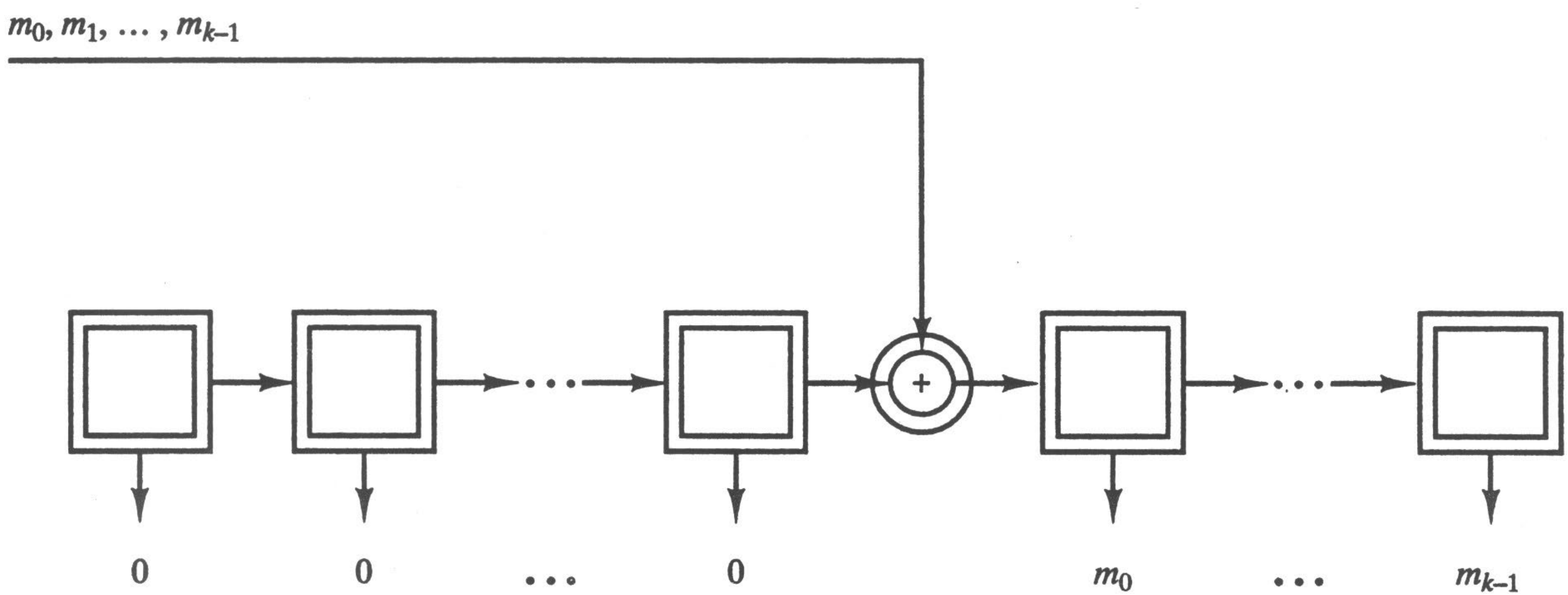
Shift Register Cell Contents

- Encoding $m(x) = x^2 + 1$

SR cells	r_0	r_1	r_2	r_3	r_4	r_5	r_6
Initial state	0	0	0	0	0	0	0
Input $m_2 = 1$	1	0	1	1	1	0	0
Input $m_1 = 0$	0	1	0	1	1	1	0
Input $m_0 = 1$	1	0	0	1	0	1	1
Final state = $\mathbf{c}_4$	1	0	0	1	0	1	1

Shift Register Multiplication

- Multiplication of $m(x)$ by x^{n-k}



Polynomial Division

- Polynomial division is performed using a
  Linear Feedback Shift Register (LFSR)
- This circuit divides a polynomial *a*(*x*) by the
  polynomial *g*(*x*)
- The result in the register is the remainder *d*(*x*)
- Consider the long division

$$g_r x^r + g_{r-1}x^{r-1} + \cdots + g_1 x + g_0 \overline{\big) a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \cdots + a_1 x + a_0}$$

- The first term in the quotient is $\frac{a_{n-1}}{g_r}x^{k-1}$

- The remainder after subtracting $\frac{a_{n-1}}{g_r}x^{k-1}g(x)$ from $a(x)$ is

$$\left(a_{n-2}-\frac{a_{n-1}}{g_r}g_{r-1}\right)x^{n-2}+\cdots+\left(a_{k-1}-\frac{a_{n-1}}{g_r}g_0\right)x^{k-1}+a_{k-2}x^{k-2}+\cdots+a_1x+a_0$$

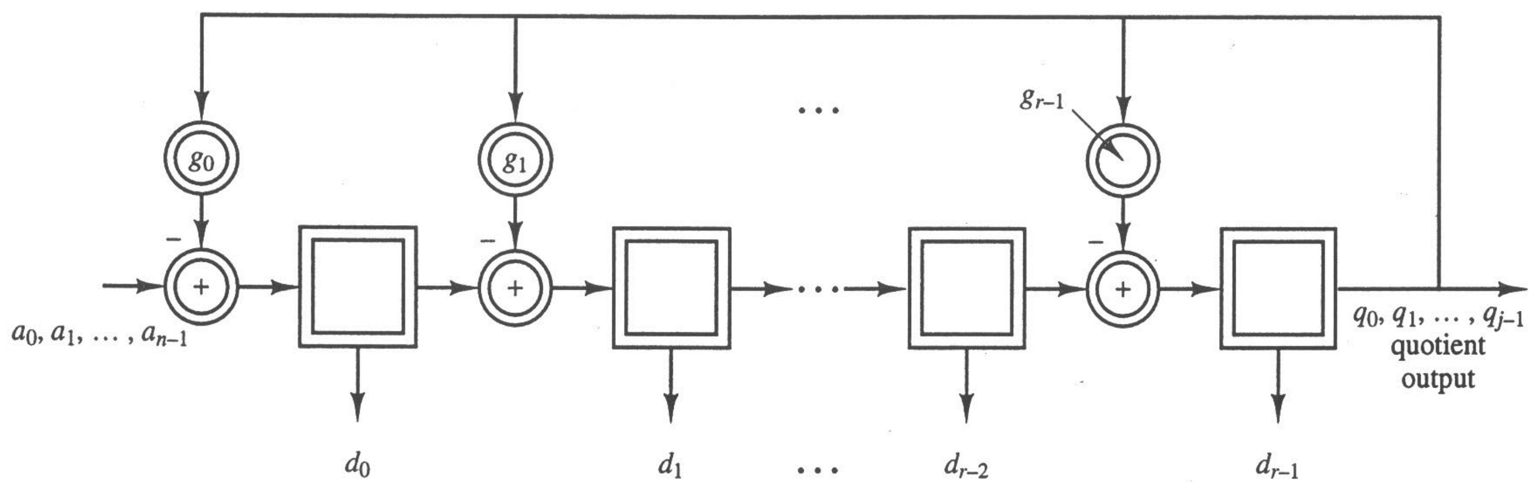
- Since g_r=1 this is

$$\left(a_{n-2}-a_{n-1}g_{r-1}\right)x^{n-2}+\cdots+\left(a_{k-1}-a_{n-1}g_0\right)x^{k-1}+a_{k-2}x^{k-2}+\cdots+a_1x+a_0$$

- After *n* shifts, *a*(*x*) has been input and the remainder *d*(*x*) is located in the shift register
- For a binary generator polynomial
  - g_0=1

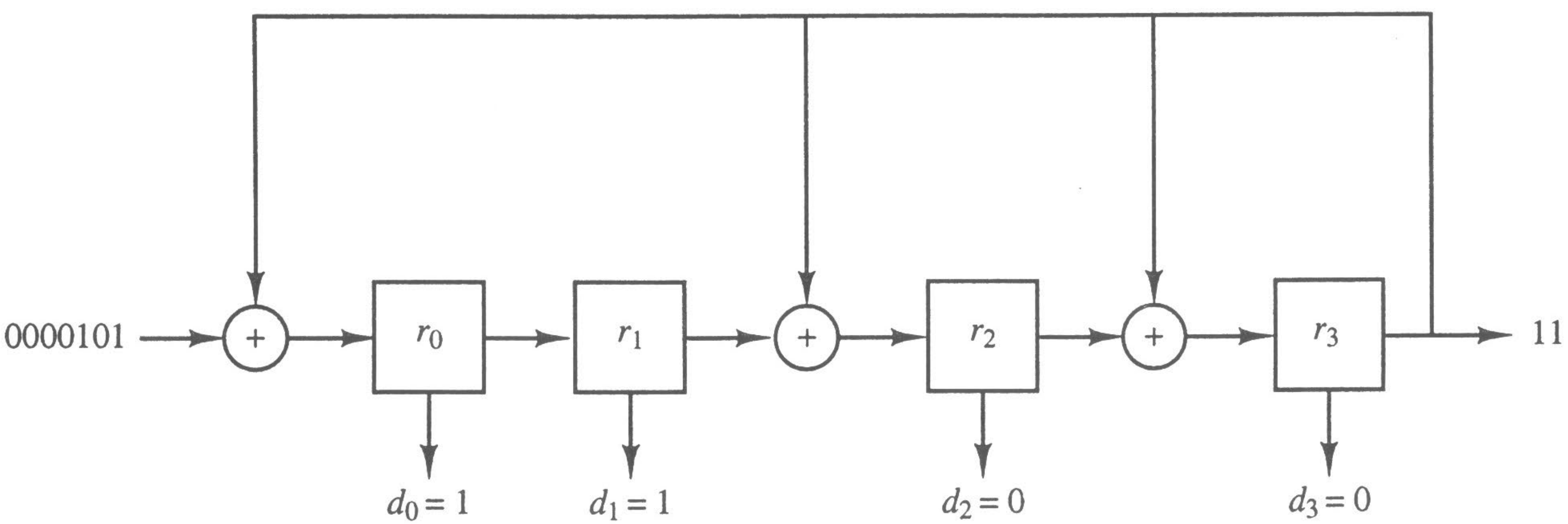
Polynomial Division

- Division of $a(x)$ by $g(x)$



Shift Register Division

- Division of $x^6 + x^4$ by $x^4 + x^3 + x^2 + 1$

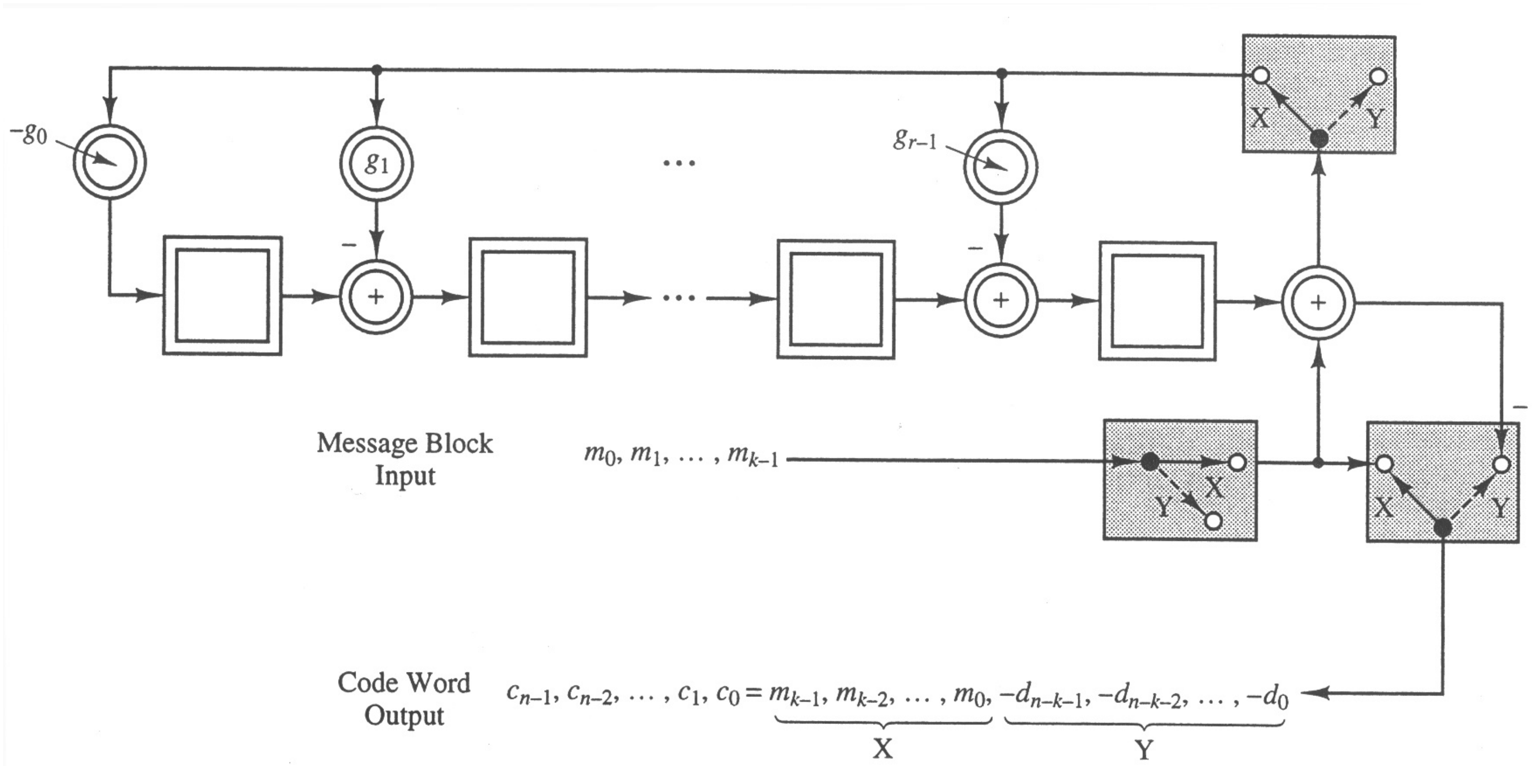


Shift Register Cell Contents

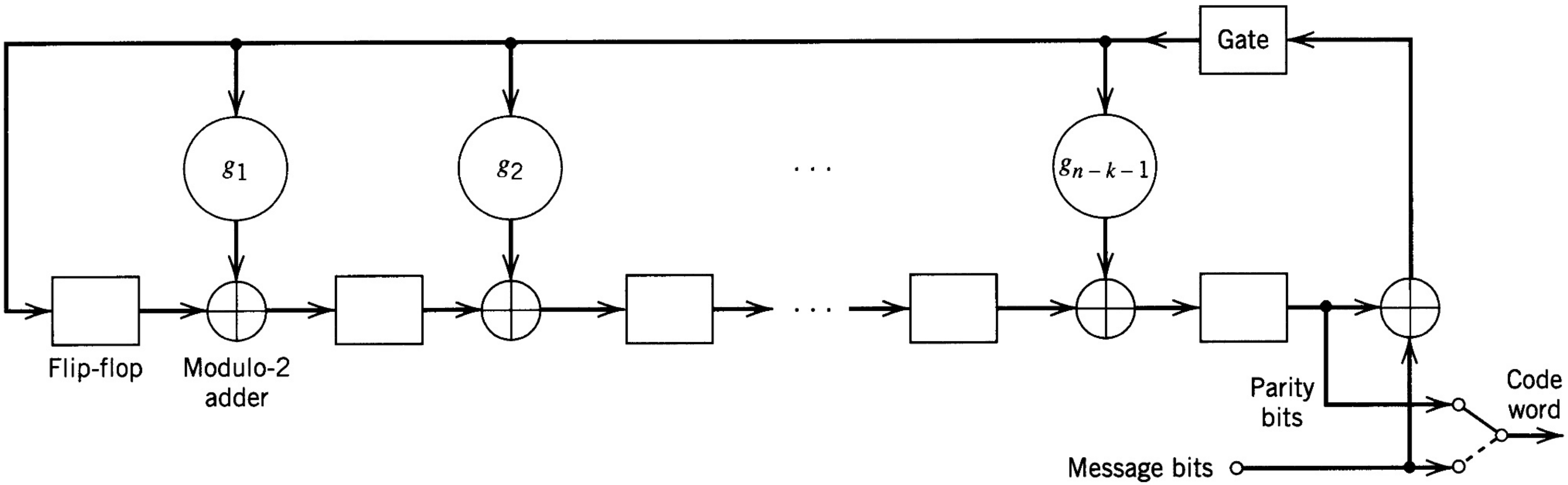
- Division of $x^6 + x^4$ by $x^4 + x^3 + x^2 + 1$

SR cells	r_0	r_1	r_2	r_3	
Initial state	0	0	0	0	
Input $a_6 = 1$	1	0	0	0	
Input $a_5 = 0$	0	1	0	0	
Input $a_4 = 1$	1	0	1	0	
Input $a_3 = 0$	0	1	0	1	
Input $a_2 = 0$	1	0	0	1	
Input $a_1 = 0$	1	1	1	1	
Input $a_0 = 0$	1	1	0	0	
Final state $= r$	1	1	0	0	$\Leftrightarrow d(x) = x + 1$

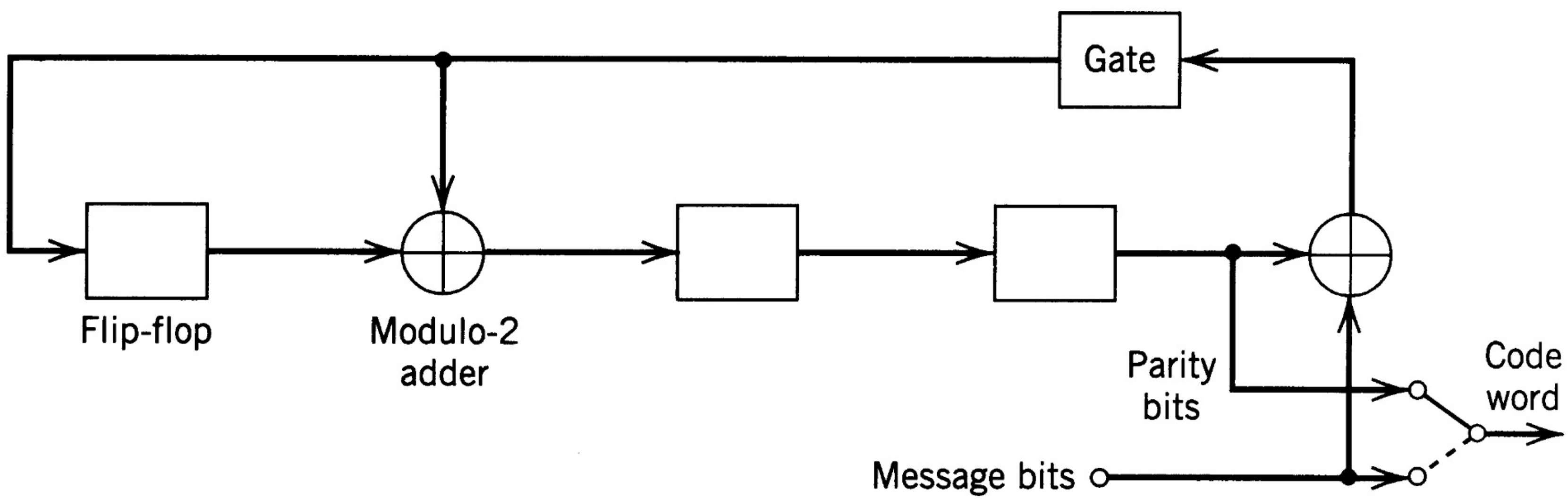
Encoder for an (n,k) Cyclic Code



Encoder for a Binary (*n*,*k*) Cyclic Code



Encoder for the (7,4) Cyclic Code Generated by $g(x) = 1+x+x^3$



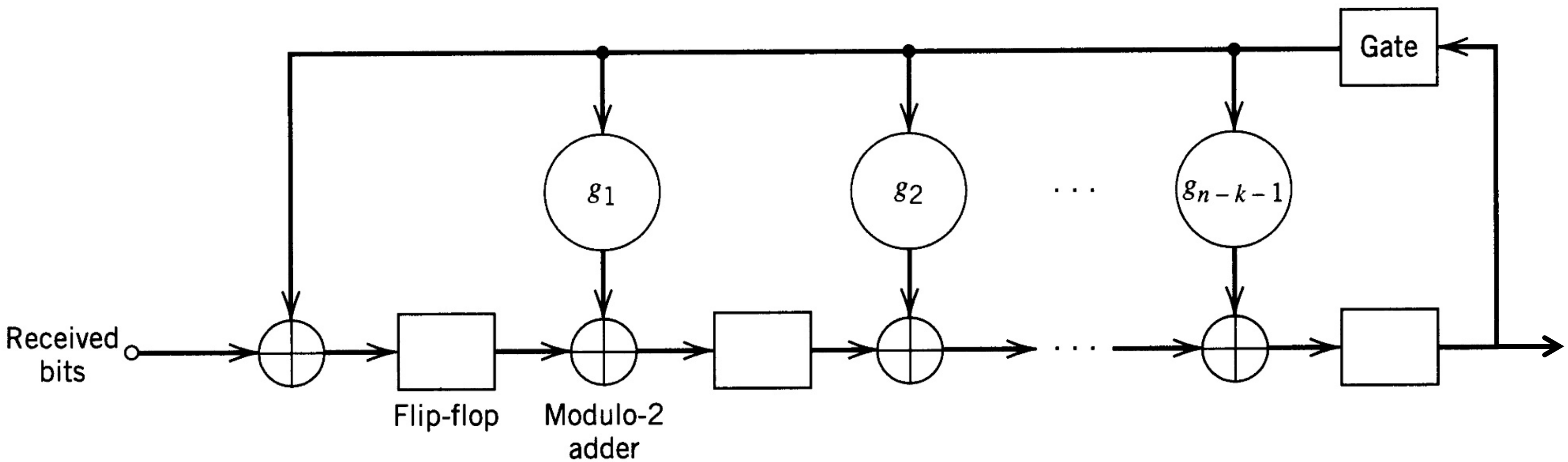
Encoding $m(x) = 1+x^2+x^3$

input	r_0	r_1	$\mathbf{r_2}$	output
1	1	1	0	1
1	1	0	1	1
0	1	0	0	0
1	1	0	0	1
-		1	0	0
-			1	0
-				1

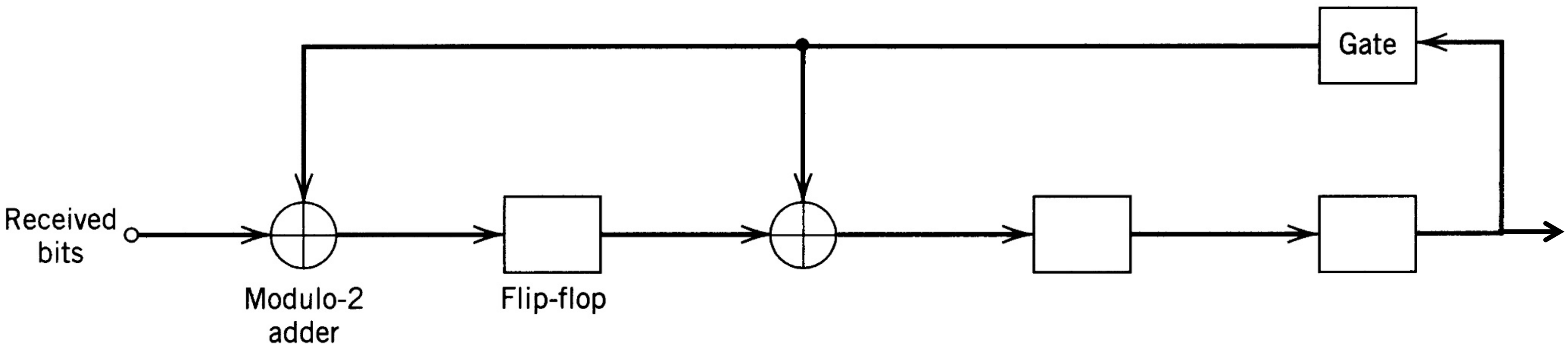
Encoding $1+x^2$ with $g(x) = 1+x^2+x^3+x^4$

input	r_0	r_1	r_2	r_3	output
1	1	0	1	1	1
0	1	1	1	0	0
1	1	1	0	0	1
-		1	1	0	0
-			1	1	0
-				1	1
-					1

Binary Syndrome Computation Circuit



Syndrome Circuit for the (7,4) Cyclic Code Generated by $g(x) = 1+x+x^3$



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

Syndrome for $x^2+x^4+x^5$

input	s_0	s_1	s_2
0	0	0	0
1	1	0	0
1	1	1	0
0	0	1	1
1	0	1	1
0	1	1	1
0	1	0	1

Shortened Cyclic Codes

- Systematic cyclic codes can be shortened by setting the *j* most significant bits of the codeword (message bits) to zero
- The resulting length is only limited by the length of the original cyclic code n and the redundancy $r=n-k$
- An (n,k) code is shortened to an $(n-j, k-j)$ code
- Since we are using a subset of the original codewords, the error correction and detection capability is at least as good as the original cyclic code

- Shortened cyclic codes are usually not cyclic, but we can still use the same shift registers for encoding and decoding as the original cyclic codes.
- Shortened cyclic codes are often called polynomial codes
- Widely used shortened cyclic codes:
  - Cyclic Redundancy Check (CRC) codes
- CRC codes are used for error detection and as hash functions

Cyclic Redundancy Check Codes

- A common choice for the generator polynomial is

  $g(x) = (x+1)b(x)$ (to detect all odd error patterns)

  where $b(x)$ is a primitive polynomial
- Example: CRC-12

  $g(x) = (x^{11}+x^2+1)(x+1)$

  This is a cyclic code of length $n = 2^{11}-1 = 2047$ and dimension $k = 2047-12= 2035$
- Only 12 bits of redundancy (parity bits)

CRC CODE	GENERATION POLYNOMIAL
CRC-4	$g_4(x) = x^4 + x^3 + x^2 + x + 1$
CRC-7	$g_7(x) = x^7 + x^6 + x^4 + 1 = (x^4 + x^3 + 1)(x^2 + x + 1)(x + 1)$
CRC-8	$g_8(x) = (x^5 + x^4 + x^3 + x^2 + 1)(x^2 + x + 1)(x + 1)$
CRC-12	$g_{12}(x) = x^{12} + x^{11} + x^3 + x^2 + x + 1 = (x^{11} + x^2 + 1)(x + 1)$
CRC-ANSI	$g_{ANSI}(x) = x^{16} + x^{15} + x^2 + 1 = (x^{15} + x + 1)(x + 1)$
CRC-CCITT	$g_{CCITT}(x) = x^{16} + x^{12} + x^5 + 1$ $= (x^{15} + x^{14} + x^{13} + x^{12} + x^4 + x^3 + x^2 + x + 1)(x + 1)$
CRC-SDLC	$g_{SDLC}(x) = x^{16} + x^{15} + x^{13} + x^7 + x^4 + x^2 + x + 1$ $= (x^{14} + x^{13} + x^{12} + x^{10} + x^8 + x^6 + x^5 + x^4 + x^3 + x + 1)$ $\cdot (x + 1)^2$
CRC24	$g_{24}(x) = x^{24} + x^{23} + x^{14} + x^{12} + x^8 + 1$ $= (x^{10} + x^8 + x^7 + x^6 + x^5 + x^4 + x^3 + x + 1)$ $\cdot (x^{10} + x^9 + x^6 + x^4 + 1)(x^3 + x^2 + 1)(x + 1)$
CRC32$_A$[Mer]	$x^{32} + x^{30} + x^{22} + x^{15} + x^{12} + x^{11} + x^7 + x^6 + x^5 + x$ $(x^{10} + x^9 + x^8 + x^6 + x^2 + x + 1)(x^{10} + x^7 + x^6 + x^3 + 1)$ $\cdot (x^{10} + x^8 + x^5 + x^4 + 1)(x + 1)(x)$
CRC-32$_B$[Ga12]	$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5$ $+ x^4 + x^2 + x + 1$

Long CRC Polynomials

CRC	$g(x)$
CRC-32$_B$ (IEEE 802)	$x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^{8}+x^{7}+x^{5}+x^{4}+x^{2}+x+1$
CRC-32	$x^{32}+x^{30}+x^{29}+x^{28}+x^{26}+x^{20}+x^{19}+x^{17}+x^{16}+x^{15}+x^{11}+x^{10}+x^{7}+x^{6}+x^{4}+x^{2}+x+1 = (x^{28}+x^{22}+x^{20}+x^{19}+x^{16}+x^{14}+x^{12}+x^{9}+x^{8}+x^{6}+1)(x+1)(x^{3}+x^{2}+1)$
CRC-40 (GSM)	$x^{40}+x^{26}+x^{23}+x^{17}+x^{3}+1$
CRC-64 (SWISS-PROT)	$x^{64}+x^{4}+x^{3}+x+1$
CRC-64 (improved)	$x^{64}+x^{63}+x^{61}+x^{59}+x^{58}+x^{56}+x^{55}+x^{52}+x^{49}+x^{48}+x^{47}+x^{46}+x^{44}+x^{41}+x^{37}+x^{36}+x^{34}+x^{32}+x^{31}+x^{28}+x^{26}+x^{23}+x^{22}+x^{19}+x^{16}+x^{13}+x^{12}+x^{10}+x^{8}+x^{7}+x^{5}+x^{3}+1$

- Coverage is the fraction of words that will be detected in error should the input be completely corrupted (worst case: a random sequence of symbols)

$$\lambda = \frac{q^n - q^k}{q^n} = 1 - q^{-(n-k)} = 1 - q^{-r}$$

- For example, CRC-12

$$\lambda = 1 - 2^{-12} = 0.999756$$

- The larger *r*=*n*-*k*, the greater the coverage

Burst Errors

- Hardware faults and multipath fading environments cause burst errors
  - Error patterns of the form

    e = …0000NXXX…XXXN0000…

    N≠0, X any symbol
  - A binary burst error of length 6 is

    e = …0001XXXX100…
- CRC codes are particularly well suited for detecting burst errors

- It can be shown that a q-ary CRC code constructed from a cyclic code can detect
  - All burst error patterns of length n-k = r or less where r is the degree of $g(x)$
  - A fraction $1\text{-}q^{1-r}/(q\text{-}1)$ of all burst error patterns of length r+1
  - A fraction $1\text{-}q^{-r}$ of all burst error patterns of length $b > r+1$
- Example: CRC-12 (q=2, r=12)
  - detects 99.95% of all length 13 burst errors
  - detects 99.976% of all length > 13 burst errors