UNIT –V

Spread Spectrum Techniques and Information theory Concepts

Part-II

Measures of Information

- Information is a measure of uncertainty. The less is the probability of occurrence of a certain message, the higher is the information.

- Since the information is closely associated with the uncertainty of the occurrence of a particular symbol, When the symbol occurs the information associated with its occurrence is defined as:

$$I_k = \log\left(\frac{1}{P_k}\right) = -\log(P_k)$$

where P_k is the probabilit y of occurrence of symbol 'k'

and I_k is the informatio n carried by symbol 'k'.

Entropy

- *Entropy* is defined in terms of probabilistic behaviour of a source of information
- In information theory the source output are discrete random variables that have a certain fixed finite alphabet with certain probabilities
  - *Entropy is an average information content for the given source symbol. (bits/message)*

$$H(\mathrm{k}) = \sum_{k=0}^{K-1} p_k \log_2\left(\frac{1}{p_k}\right)$$

Example

Find the entropy of the source alphabets $\{S_0, S_1, S_2\}$ with respective probabilities

$$p_0 = \tfrac{1}{4}$$
$$p_1 = \tfrac{1}{4}$$
$$p_2 = \tfrac{1}{2}$$

Solution:

W.K.T ,

$$H(\mathrm{k}) = \sum_{k=0}^{K-1} p_k \log_2\left(\frac{1}{p_k}\right)$$

$$= p_0 \log_2\left(\frac{1}{p_0}\right) + p_1 \log_2\left(\frac{1}{p_1}\right) + p_2 \log_2\left(\frac{1}{p_2}\right)$$

$$= \frac{1}{4}\log_2(4) + \frac{1}{4}\log_2(4) + \frac{1}{2}\log_2(2)$$

$$= \frac{3}{2}\text{ bits}$$

Rate of information

- If a source generates at a rate of ‘r’ messages per second, the rate of information ‘R’ is defined as the average number of bits of information per **second.**

- ‘H’ is the average number of bits of information per message.

Hence, $R = rH$ bits/sec

Source Coding

- *Source coding* (lossless data compression) means that we will remove redundant information from the signal prior the transmission.
- Basically this is achieved by assigning short descriptions to the most frequent outcomes of the source output and vice versa.
- The common source-coding schemes are

  huffman coding

  Shannon-Fano Coding

  lempel-ziv coding

  prefix coding

What is need for source coding ?

- For analog sources , the sources coding is related to amplitude distribution and the function of the source waveform.

- For discrete sources, the sources coding is related to information content and correlation among the symbols of the source.

Source encoding must satisfy two functional requirements,

1.Efficient representation of data in binary form

2. It must be uniquely decodable.

Source Coding Theorem

- *Source coding theorem* states that the output of any information source having entropy H units per symbol can be encoded into an alphabet having N symbols in such a way that the source symbols are represented by code words having a weighted average length not less than **H/log N**.

- Hence source coding theorem says that encoding of messages from a source with entropy H can be done, bounded by the fundamental information theoretic limitation that the **Minimum average number of symbols/message is H/logN.**

The average code word length $\bar{L}$, of source encoder is given as,

$$\bar{L} = \sum_{k=0}^{K-1} p_k l_k$$

$\bar{L}$ - represents average number of bits per source symbol

The code Efficiency is given by,

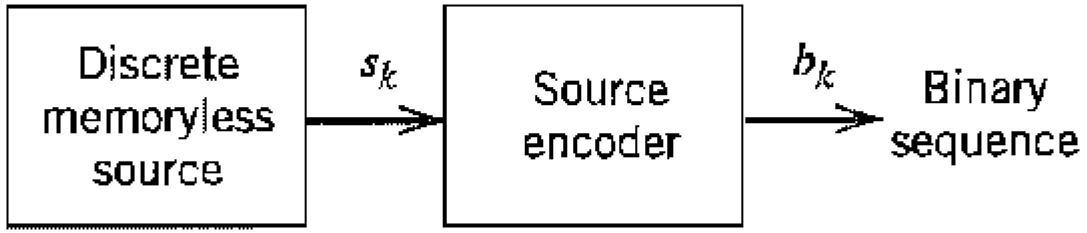
$$\eta = \frac{L_{\min}}{\bar{L}}$$

$L_{\min}$ - represents the minimum possible value of $\bar{L}$

With $\bar{L} \geq L_{\min}$, we clearly have $\eta \leq 1$. The source encoder is said to be *efficient* when η approaches unity.

The average code word length for any distortionless source encoding scheme is bounded as

$$\bar{L} \geq H(\mathcal{S})$$



According to source coding theorem, the entropy represents a fundamental limit on the average number of bits per source symbol , but no smaller than the entropy ,Hence we can rewrite the efficiency of a source coder in terms of the entropy as

$$\eta = \frac{H(\mathcal{S})}{\bar{L}}$$

Huffman Coding

With the Huffman code in the binary case the two least probable source output symbols are joined together, resulting in a new message alphabet with one less symbol

Application : JPEG, MPEG, MP3

ADVANTAGES:

- ***uniquely*** decodable code
- ***smallest*** average codeword length

DISADVANTAGES:

- LARGE tables give complexity
- sensitive to channel errors

Procedure

- Create a list for the symbols, in decreasing order of probability. The symbols with the lowest probability are assigned a ‘0’ and a ‘1’.

- These two symbols are combined into a new symbol with the probability equal to the sum of their individual probabilities. The new symbol is placed in the list as per its probability value.

- The procedure is repeated until we are left with 2 symbols only for which 0 and 1 are assigned.

- Huffman code is the bit sequence obtained by working backwards and tracking sequence of 0’s and 1’s assigned to that symbol and its successors.

Huffman Coding: Example

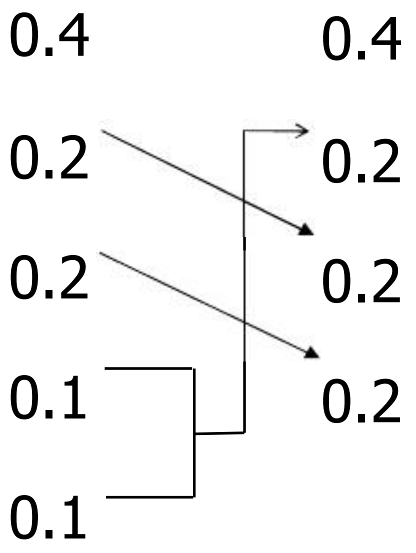
Source Symbol s_k	Symbol Probability p_k
s_0	0.1
s_1	0.2
s_2	0.4
s_3	0.2
s_4	0.1

Solution

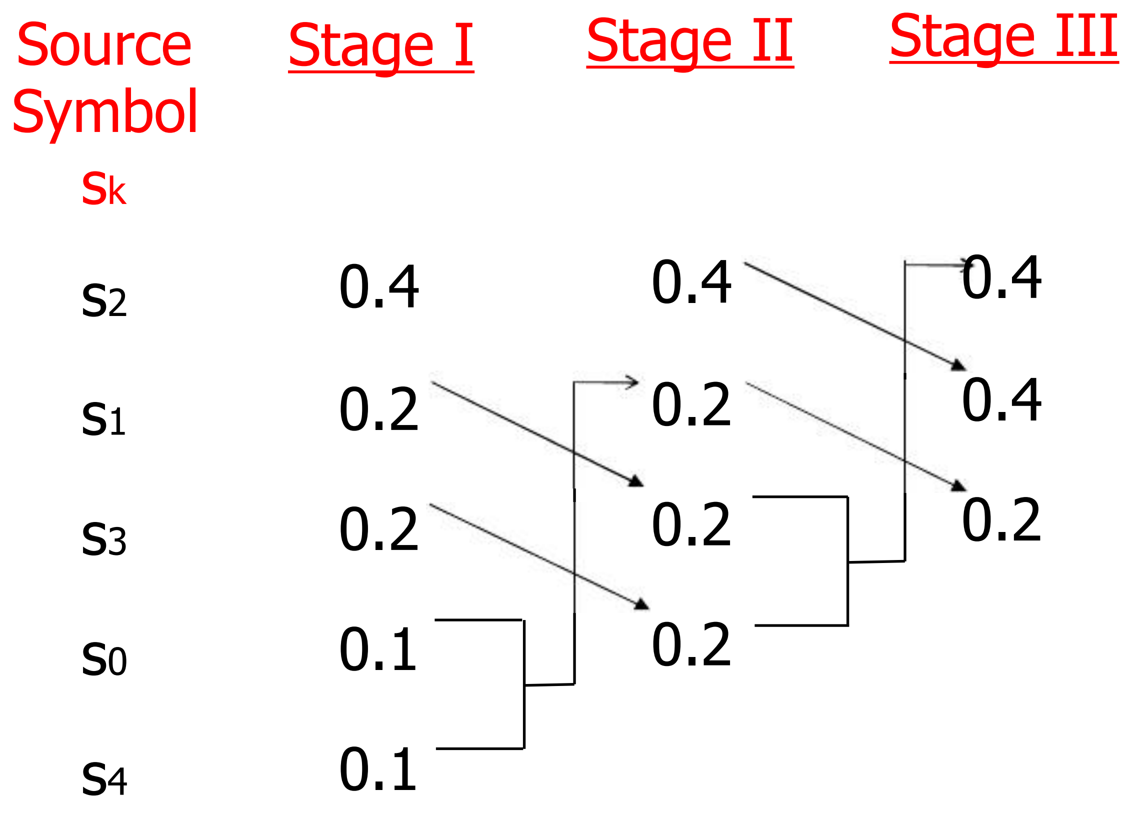
Source Symbol s_k	Stage I
s_2	0.4
s_1	0.2
s_3	0.2
s_0	0.1
s_4	0.1

Solution

Source Symbol s_k	Stage I	Stage II
s_2	0.4	0.4
s_1	0.2	0.2
s_3	0.2	0.2
s_0	0.1	0.2
s_4	0.1	

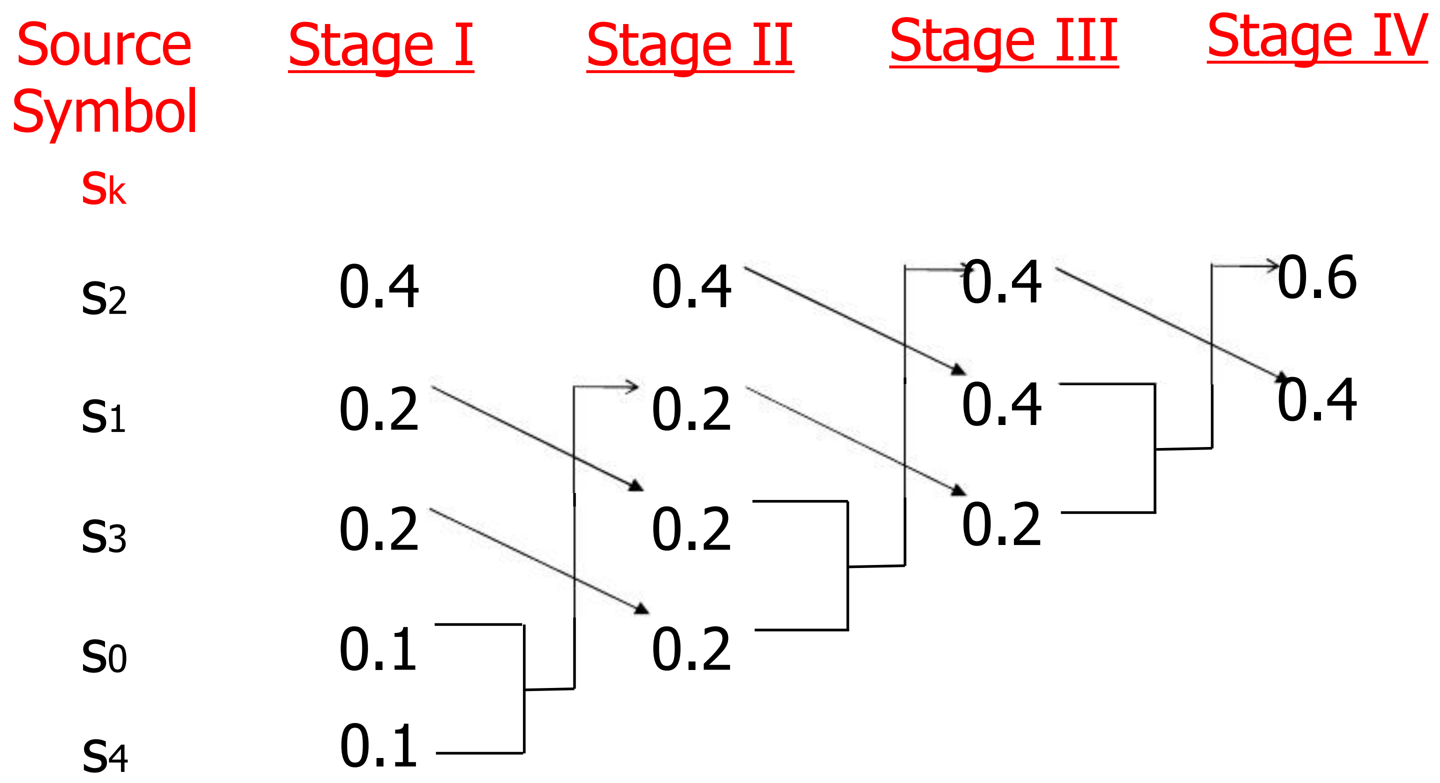


Solution



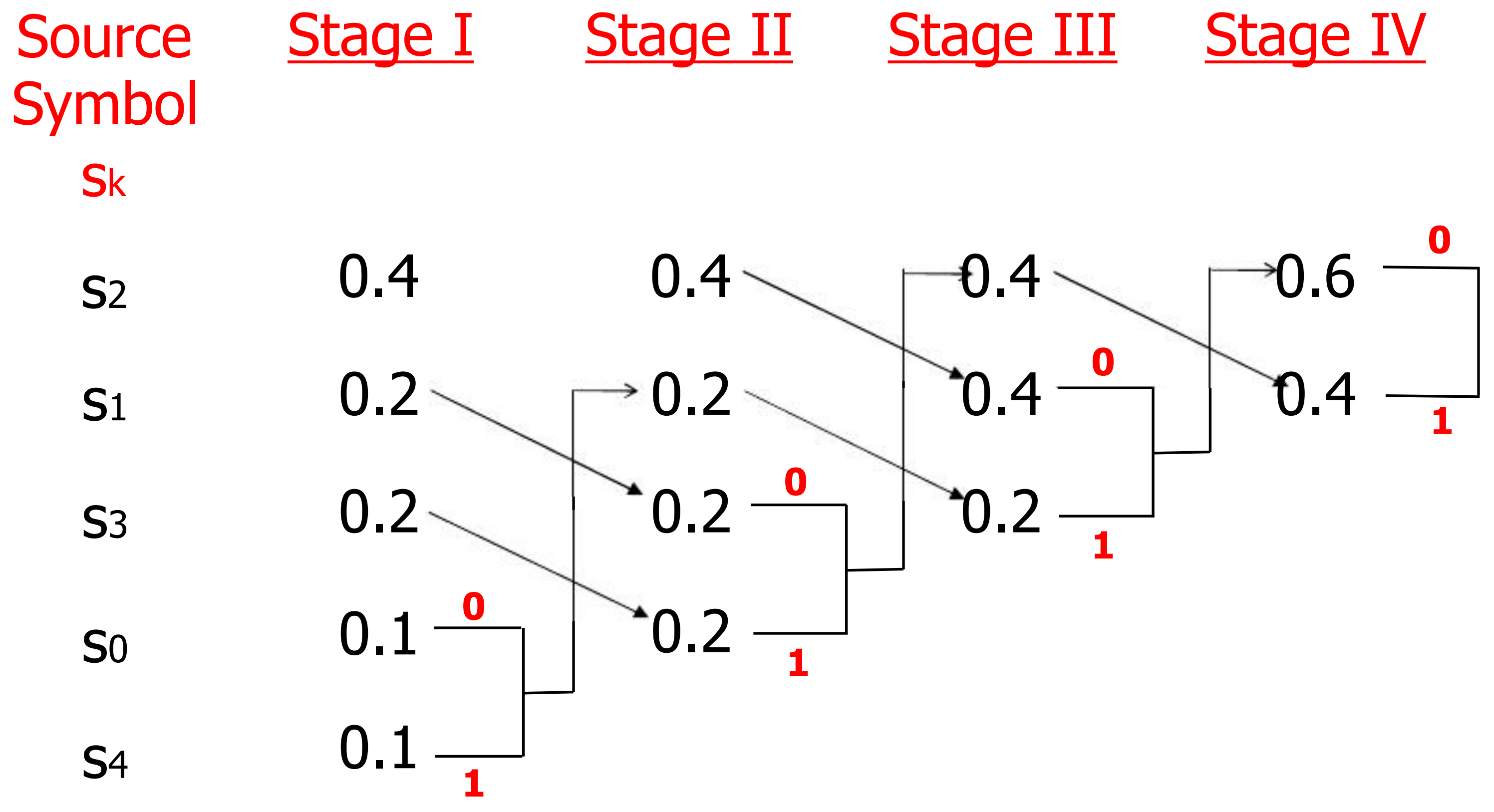
Source Symbol
Sk
Stage I
Stage II
Stage III
S2
S1
S3
S0
S4
0.4
0.2
0.2
0.1
0.1
0.4
0.2
0.2
0.2
0.4
0.4
0.2

Solution



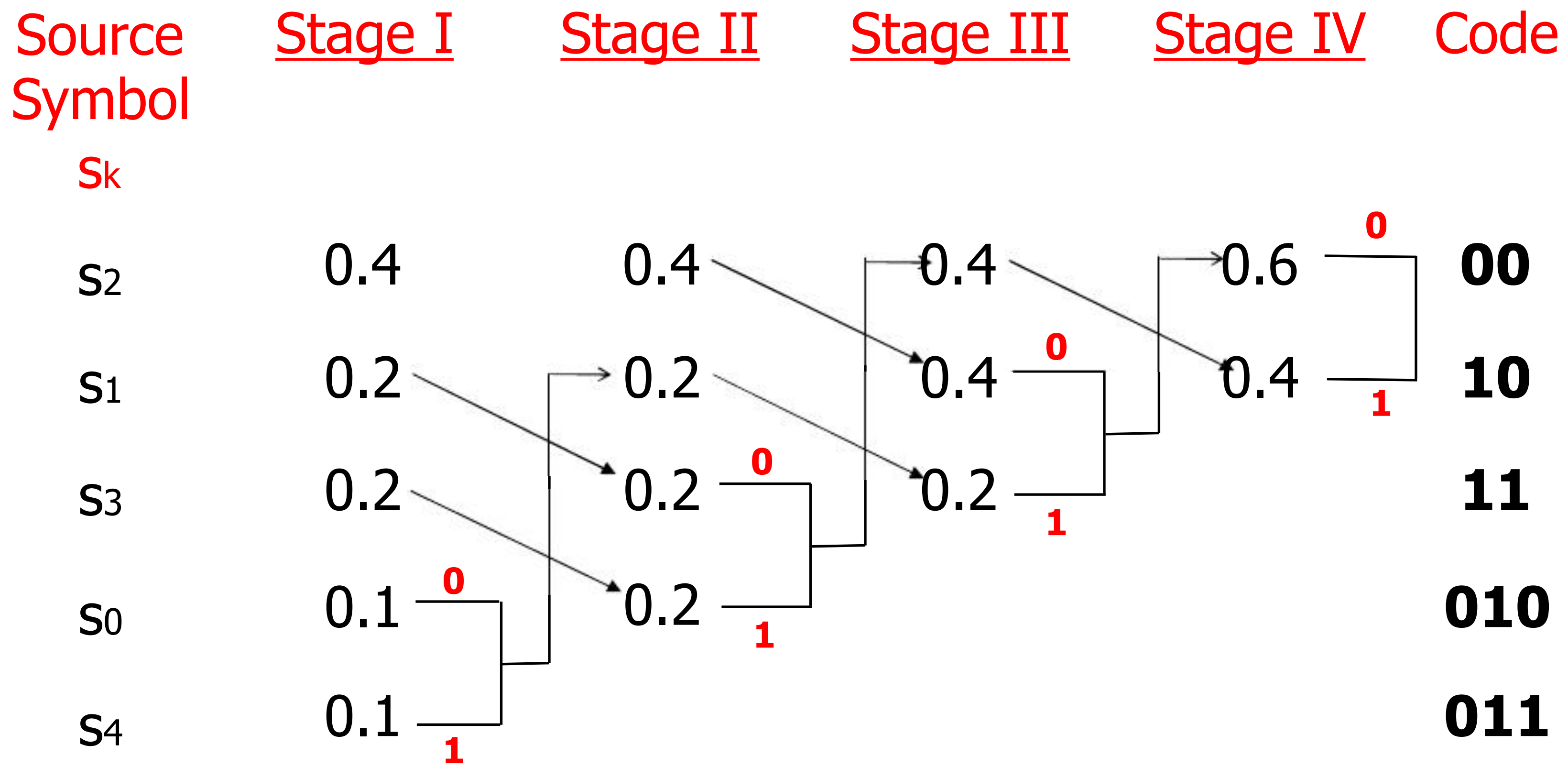
Source Symbol
s_k
Stage I
Stage II
Stage III
Stage IV
s_2 0.4 0.4 0.4 0.6
s_1 0.2 0.2 0.4 0.4
s_3 0.2 0.2 0.2
s_0 0.1 0.2
s_4 0.1

Solution



Source Symbol
s_k
Stage I
Stage II
Stage III
Stage IV
s_2 0.4
s_1 0.2
s_3 0.2
s_0 0.1
s_4 0.1
0.4
0.2
0.2
0.2
0.4
0.4
0.2
0.6
0.4
0
1
0
1
0
1
0
1

Solution



Source Symbol s_k	Stage I	Stage II	Stage III	Stage IV	Code
s_2	0.4	0.4	0.4	0.6	**00**
s_1	0.2	0.2	0.4	0.4	**10**
s_3	0.2	0.2	0.2		**11**
s_0	0.1	0.2			**010**
s_4	0.1				**011**

The average code word length is,

$$\begin{aligned}\bar{L} &= 0.4(2) + 0.2(2) + 0.2(2) + 0.1(3) + 0.1(3) \\ &= 2.2\end{aligned}$$

The entropy is,

$$\begin{aligned}H(\mathcal{S}) = {} & 0.4 \log_2\left(\frac{1}{0.4}\right) + 0.2 \log_2\left(\frac{1}{0.2}\right) + 0.2 \log_2\left(\frac{1}{0.2}\right) \\ & + 0.1 \log_2\left(\frac{1}{0.1}\right) + 0.1 \log_2\left(\frac{1}{0.1}\right) \\ = {} & 0.52877 + 0.46439 + 0.46439 + 0.33219 + 0.33219 \\ = {} & 2.12193 \text{ bits}\end{aligned}$$

Exercise

Find Huffman code for the following source

Symbol	Probability
h	0.1
e	0.1
l	0.4
o	0.25
w	0.05
r	0.05
d	0.05

Shannon-Fano Coding

- In Shannon–Fano coding, the symbols are arranged in order from most probable to least probable, and then divided into two sets whose total probabilities are as close as possible to being equal. All symbols then have the first digits of their codes assigned; symbols in the first set receive "0" and symbols in the second set receive "1".

- As long as any sets with more than one member remain, the same process is repeated on those sets, to determine successive digits of their codes. When a set has been reduced to one symbol, of course, this means the symbol's code is complete and will not form the prefix of any other symbol's code.

Shannon’s Channel Coding theorem

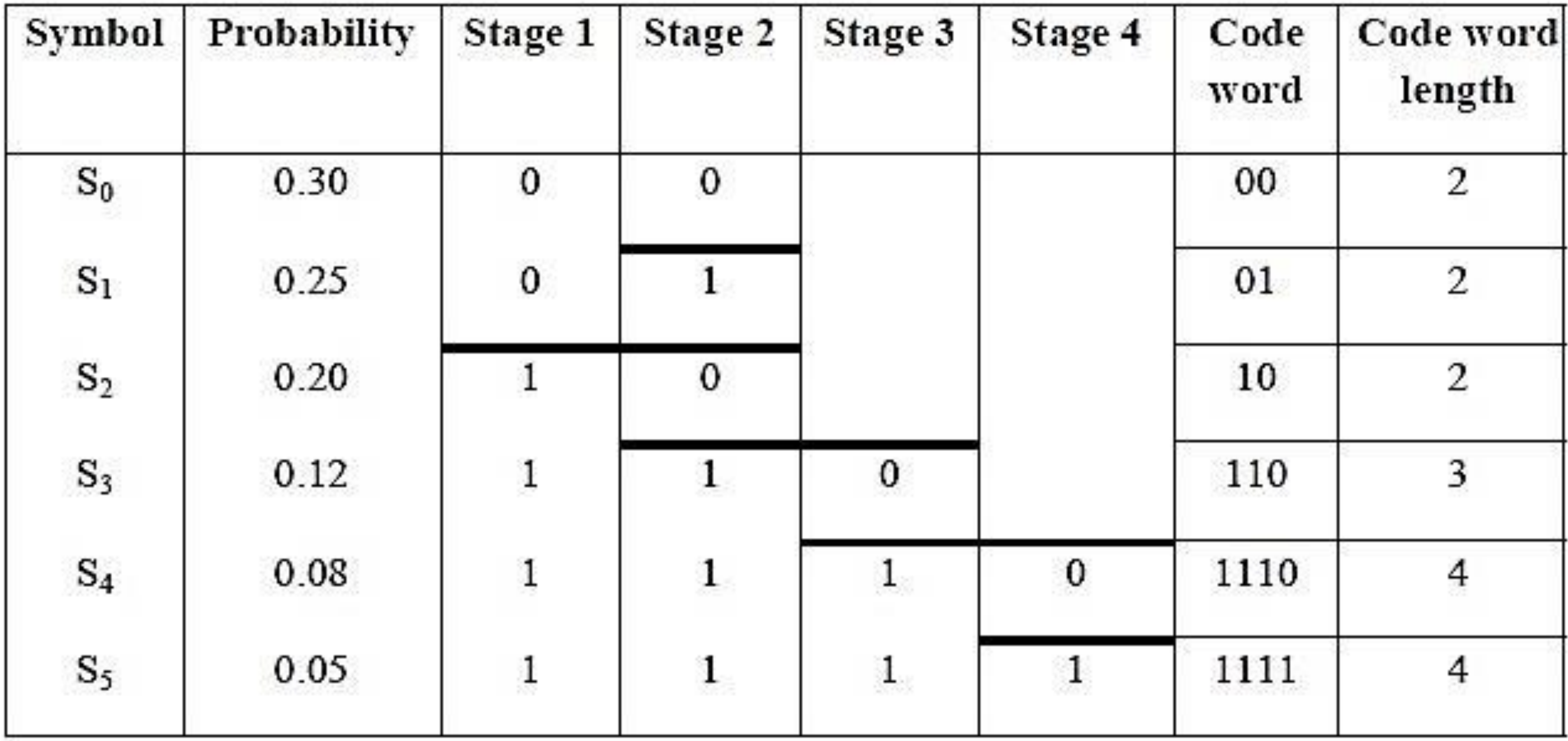
- The Shannon theorem states that given a noisy channel with channel capacity C and information transmitted at a rate R, then if $R < C$ there exist codes that allow the probability of error at the receiver to be made arbitrarily small. This means that theoretically, it is possible to transmit information nearly without error at any rate below a limiting rate, C.

- The converse is also important. If $R > C$, an arbitrarily small probability of error is not achievable. All codes will have a probability of error greater than a certain positive minimal level, and this level increases as the rate increases. So, information cannot be guaranteed to be transmitted reliably across a channel at rates beyond the channel capacity.

Shannon Coding: Example

A discrete memoryless source has six source symbols S0, S1, S2, S3, S4 and S5 with their probability assignments P(S0) = 0.30, P(S1) = 0.25, P(S2) = 0.20, P(S3) = 0.12, P(S4) = 0.08 and P(S5) = 0.05. Encode the source with Shannon-Fano codes. Calculate its efficiency.

Solution



Symbol	Probability	Stage 1	Stage 2	Stage 3	Stage 4	Code word	Code word length
S_0	0.30	0	0			00	2
S_1	0.25	0	1			01	2
S_2	0.20	1	0			10	2
S_3	0.12	1	1	0		110	3
S_4	0.08	1	1	1	0	1110	4
S_5	0.05	1	1	1	1	1111	4

The average code word length is,

$\bar{L}$ = 0.30 (2) + 0.25 (2) + 0.20 (2) + 0.12(3) + 0.08(4) + 0.05(4)
= **2.38 bits/symbol**

The entropy is,

$$H(\mathcal{S}) = 0.3\log_2\left(\frac{1}{0.3}\right)+0.25\log_2\left(\frac{1}{0.25}\right)+0.2\log_2\left(\frac{1}{0.2}\right)$$
$$+0.12\log_2\left(\frac{1}{0.12}\right)+0.8\log_2\left(\frac{1}{0.08}\right)+0.05\log_2\left(\frac{1}{0.05}\right)$$

$$H(\mathcal{S}) = \mathbf{2.36 bits/symbol}$$

$$\eta = \frac{H(\mathcal{S})}{\bar{L}} = \frac{2.36}{2.38} = 99.15\%$$

Linear Block Codes

Channel Coding

Why?

To increase the resistance of digital communication systems to channel noise via error control coding

How?

By mapping the incoming data sequence into a channel input sequence and inverse mapping the channel output sequence into an output data sequence in such a way that the overall effect of channel noise on the system is minimized

Redundancy is introduced in the channel encoder so as to reconstruct the original source sequence as accurately as possible.

The implication of Error Control Coding

- Addition of redundancy implies the need for increased transmission bandwidth

- It also adds complexity in the decoding operation

- Therefore, there is a design trade-off in the use of error-control coding to achieve acceptable error performance considering bandwidth and system complexity.

Types of Error Control Coding

- **Linear Block codes**
- **Convolutional codes**

Linear Block Codes

An (n,k) block code indicates that the codeword has n number of bits and k is the number of bits for the original binary message

A code is said to be linear if any two code words in the code can be added in modulo-2 arithmetic to produce a third code word in the code

Linear Block Codes

Codes in which the message bits are transmitted in an unaltered form.

Example : Consider an (n,k) linear block code

There are 2^k number of distinct message blocks and 2^n number of distinct code words

Let $m_0,m_1,\ldots.m_{k-1}$ constitute a block of k-bits binary message

By applying this sequence of message bits to a linear block encoder, it adds n-k bits to the binary message

Let $b_0,b_1,\ldots.b_{n-k-1}$ constitute a block of n-k-bits redundancy

This will produce an n-bits code word

Let $c_0,c_1,\ldots.c_{n-1}$ constitute a block of n-bits code word

Using vector representation they can be written in a row vector notation respectively as

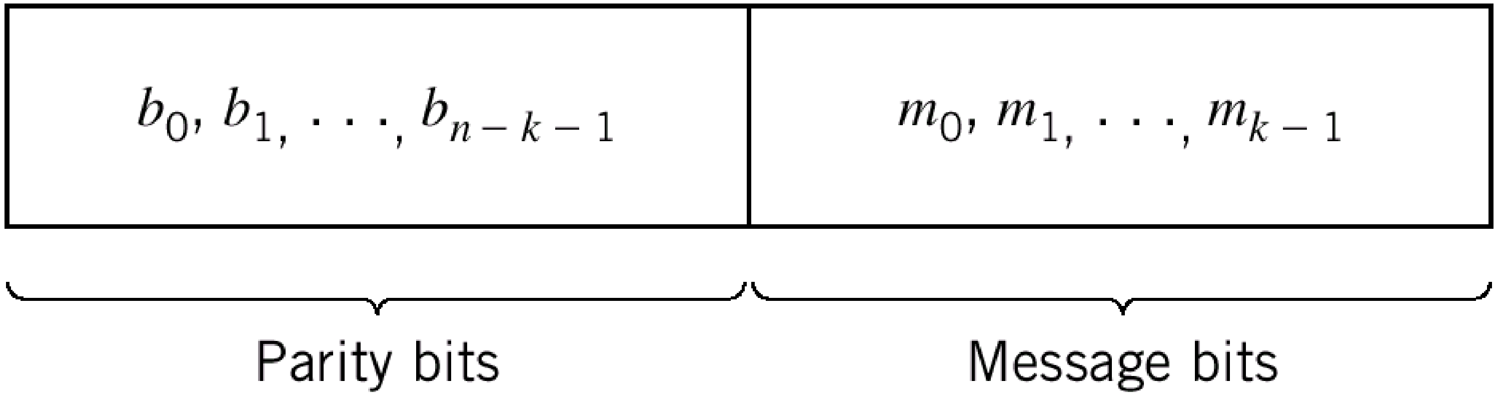
$(c_0\ c_1 \ldots.\ c_n)$, $(m_0\ m_1 \ldots.\ m_{k-1})$ **and** $(b_0\ b_1 \ldots.\ b_{n-k-1})$

Linear Block Codes

Using matrix representation, we can define

c, the 1-by-n code vector = [c_0 c_1 …. c_n]
m, the 1-by-k message vector =[m_0 m_1 …. m_{k-1}]
b, the 1-by-(n-k) parity vector = [b_0 b_1 …. b_{n-k-1}]

With a systematic structure, a code word is divided into 2 parts. 1 part occupied by the binary message only and the other part by the redundant (parity) bits.



The (n-k) left-most bits of a code word are identical to the corresponding parity bits
The k right-most bits of a code word are identical to the corresponding message bits

Linear Block Codes

In matrix form, we can write the code vector,**c** as a partitioned row vector in terms of vectors **m** and **b**

$$\mathbf{c}=[\mathbf{b}\ \ \mathbf{m}]$$

Given a message vector m, the corresponding code vector, c for a systematic linear (n,k) block code can be obtained by a matrix multiplication

$$\mathbf{c}=\mathbf{m}.\mathbf{G}$$

Where **G** is the k-by-n generator matrix.

Linear Block Codes

The generator matrix, G

G, the k-by-n generator matrix has the general structure

$$\mathbf{G} = [\mathbf{I_k} \,\vdots\, \mathbf{P}]$$

Where I_k is the k-by-k identity matrix and

P is the k-by-(n-k) coefficient matrix

$$I_k = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix} \qquad P = \begin{bmatrix} P_{00} & P_{01} & \dots & P_{0,n-k-1} \\ P_{10} & P_{11} & \dots & P_{1,n-k-1} \\ \vdots & \vdots & & \vdots \\ P_{k-1,0} & P_{k-1,1} & \dots & P_{k-1,n-k-1} \end{bmatrix}$$

The identity matrix simply reproduces the message vector for the first k elements of **c**
The coefficient matrix generates the parity vector,**b** via **b=m**.**P**
The elements of P are found via research on coding.

Cyclic Codes

A subclass of linear codes having a cyclic structure.

The code vector can be expressed in the form

$$c = (c_{n-1}\ \ c_{n-2}\ \ \ldots\ldots c_1\ c_0)$$

A new code vector in the code can be produced by cyclic shifting of another code vector.
For example, a cyclic shift of all n bits one position to the left gives

$$c' = (c_{n-2}\ \ c_{n-3}\ \ \ldots\ldots c_1\ c_0\ c_{n-1})$$

A second shift produces another code vector, c''

$$c'' = (c_{n-3}\ \ c_{n-4}\ \ \ldots\ldots c_1\ c_0\ c_{n-1}\ c_{n-2})$$

Cyclic Codes

The cyclic property can be treated mathematically by associating a code vector, c with the code polynomial, c(X)

$$c(X) = c_0 + c_1X + c_2X^2 + \ldots\ldots c_{n-1}X^{n-1}$$

The power of X denotes the positions of the codeword bits.

The coefficients are either 1s and 0s.

An (n,k) cyclic code is defined by a generator polynomial, g(X)

$$g(X) = X^{n-k} + g_{n-k-1}X^{n-k-1} + \ldots\ldots + g_1X + 1$$

The coefficient g are such that g(X) is a factor of $X^n + 1$

Cyclic Codes – Encoding Procedure

To encode an (n,k) cyclic code

1. Multiply the message polynomial , m(X) by X^{n-k}
2. Divide X^{n-k}.m(X) by the generator polynomial, g(X) to obtain the remainder polynomial, b(X)
3. Add b(X) to $X^{n-k} \cdot$m(X) to obtain the code polynomial

Cyclic Codes - Example

The (7,4) Hamming Code

For message sequence 1001

The message polynomial, $m(X) = 1 + X^3$ (1001)

1. Multiply by X^{n-k} (X^3) gives $X^3 + X^6$

2. Divide by the generator polynomial, $g(X)$ that is a factor of $X^n + 1$

For the (7,4) Hamming code is defined by its generator polynomials, $g(X)$ that are factors of $X^7 + 1$

With $n = 7$, we can factorize $X^7 + 1$ into three irreducible polynomials

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

Cyclic Codes - Example

For example we choose the generator polynomial, $1 + X + X^3$ and perform the division we get the remainder, $b(X)$ as $X^2 + X$ (011)

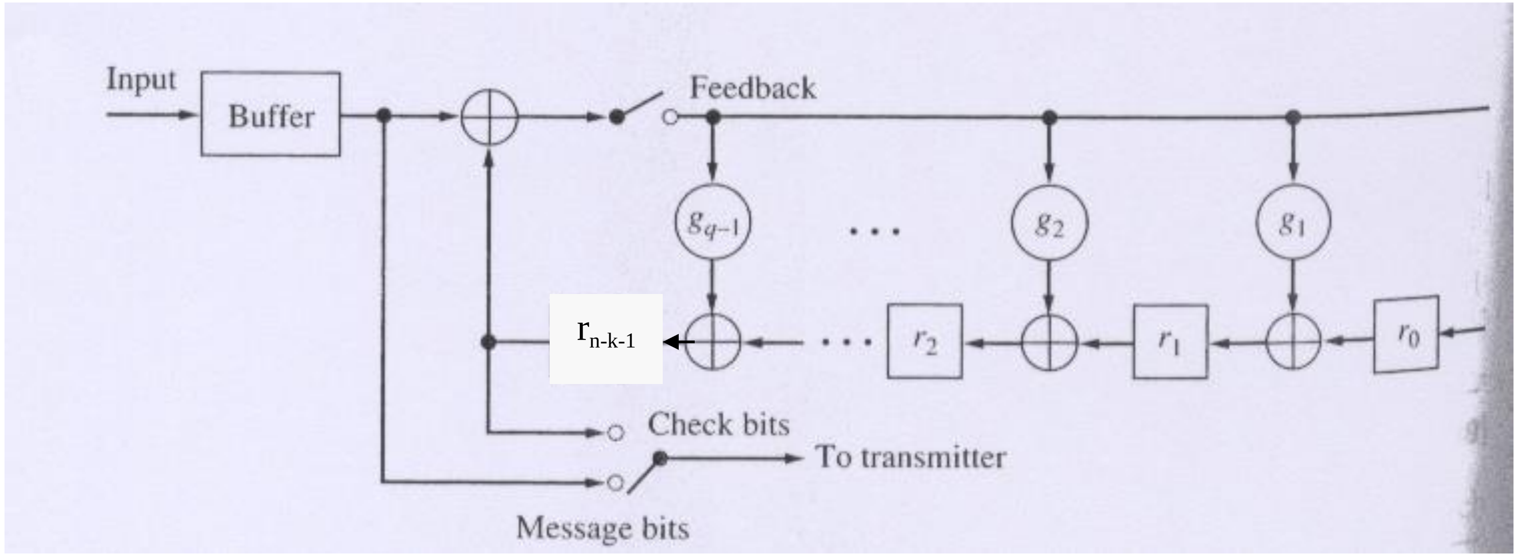
3. Add $b(X)$ with $X^3 + X^6$ to obtain the code polynomial, $c(X)$

$$c(X) = X + X^2 + X^3 + X^6$$

So the codeword for message sequence 1001 is 0111001

Cyclic Codes – Implementation

The Cyclic code is implemented by the shift-register encoder with (n-k) stages



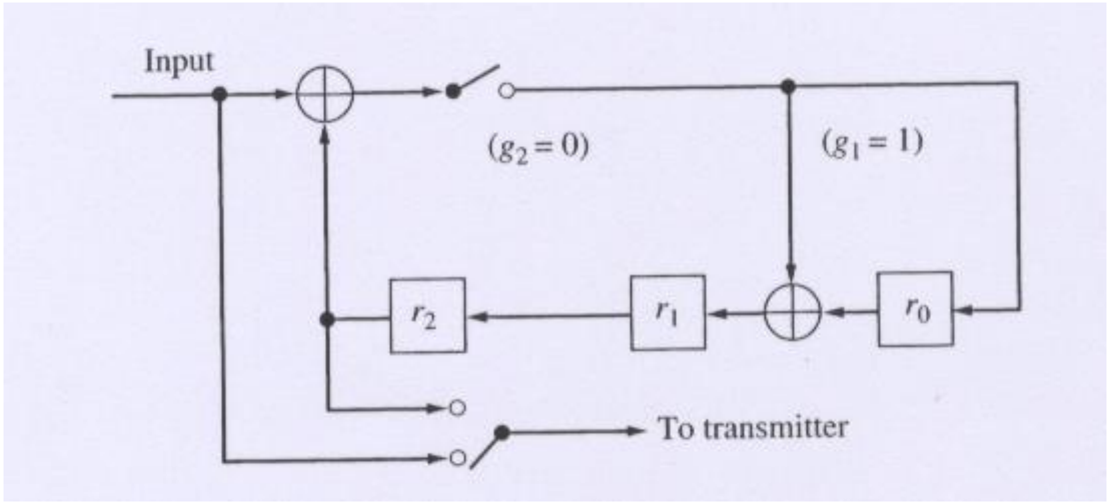
Encoding starts with the feedback switch closed, the output switch in the message bit position, and the register initialized to the all-zero state.

The k message bits are shifted into the register and delivered to the transmitter. After k shift cycles, the register contains the b check bits.

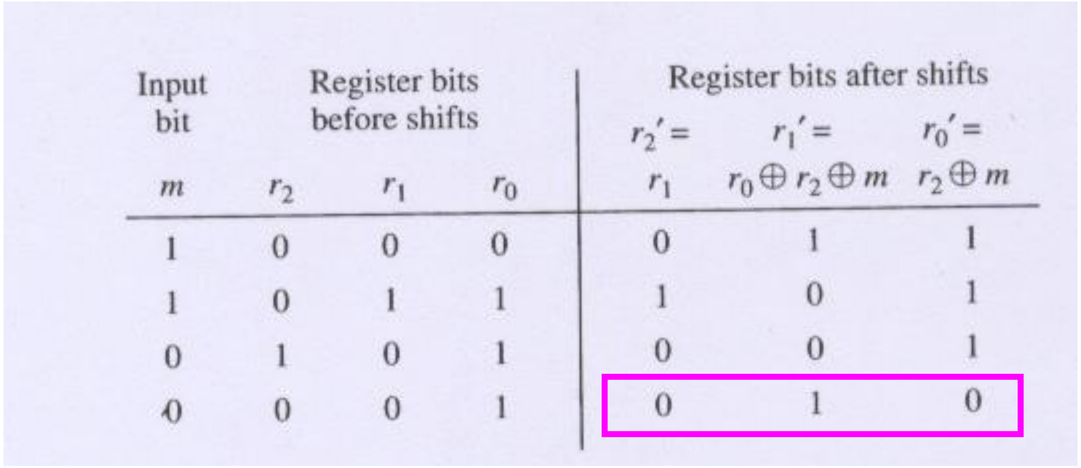
The feedback switch is now opened and the output switch is moved to the check bits to deliver them to the transmitter.

Cyclic Codes – Implementation example

The shift-register encoder for the (7,4) Hamming Code has (7-4=3) stages



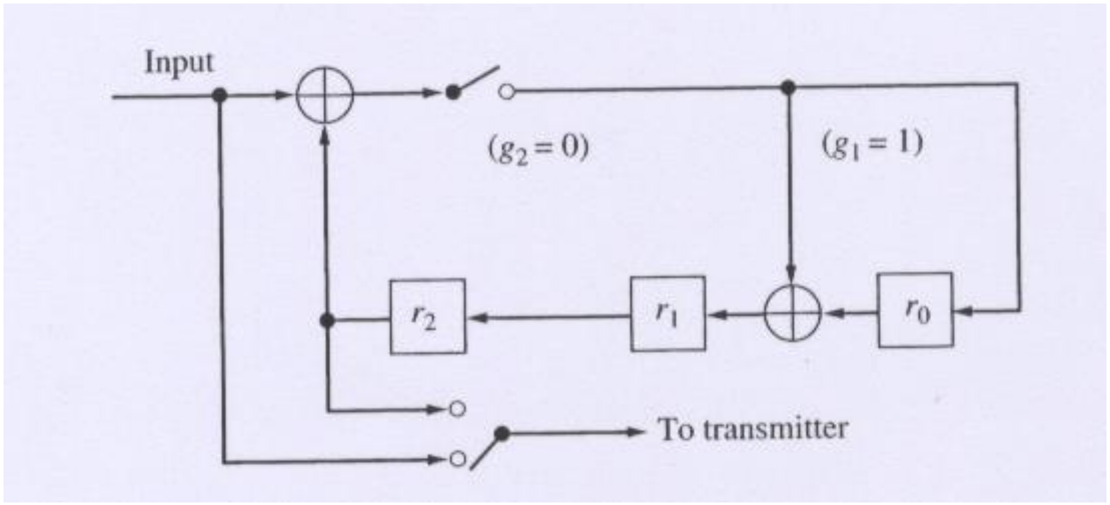
When the input message is 0011, after 4 shift cycles the redundancy bits are delivered



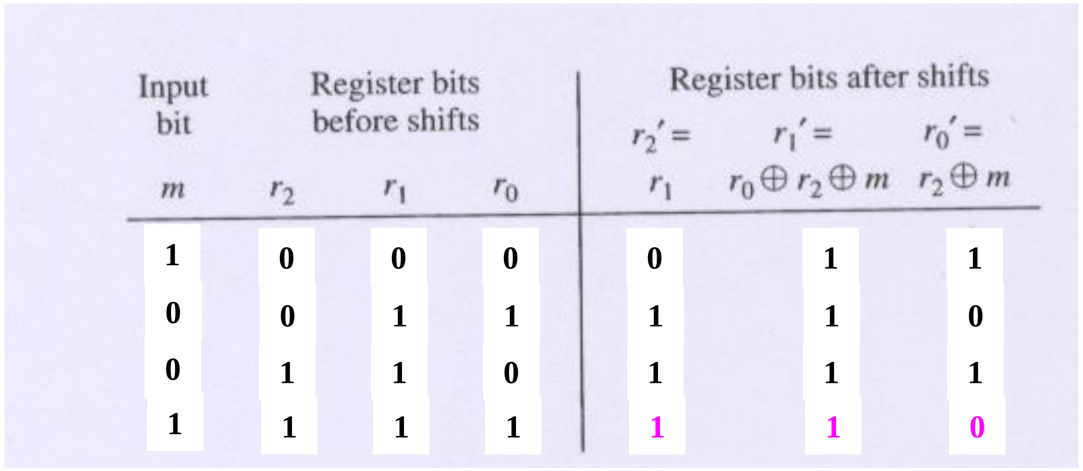
Input bit	Register bits before shifts			Register bits after shifts		
				$r_2' =$	$r_1' =$	$r_0' =$
m	r_2	r_1	r_0	r_1	$r_0 \oplus r_2 \oplus m$	$r_2 \oplus m$
1	0	0	0	0	1	1
1	0	1	1	1	0	1
0	1	0	1	0	0	1
0	0	0	1	0	1	0

Cyclic Codes – Implementation Exercise

The shift-register encoder for the (7,4) Hamming Code has (7-4=3) stages



When the input message is 1001, after 4 shift cycles the redundancy bits are delivered



Input bit	Register bits before shifts			Register bits after shifts		
				$r_2' =$	$r_1' =$	$r_0' =$
m	r_2	r_1	r_0	r_1	$r_0 \oplus r_2 \oplus m$	$r_2 \oplus m$
1	0	0	0	0	1	1
0	0	1	1	1	1	0
0	1	1	0	1	1	1
1	1	1	1	1	1	0

The check bits is 011