Channel Coding

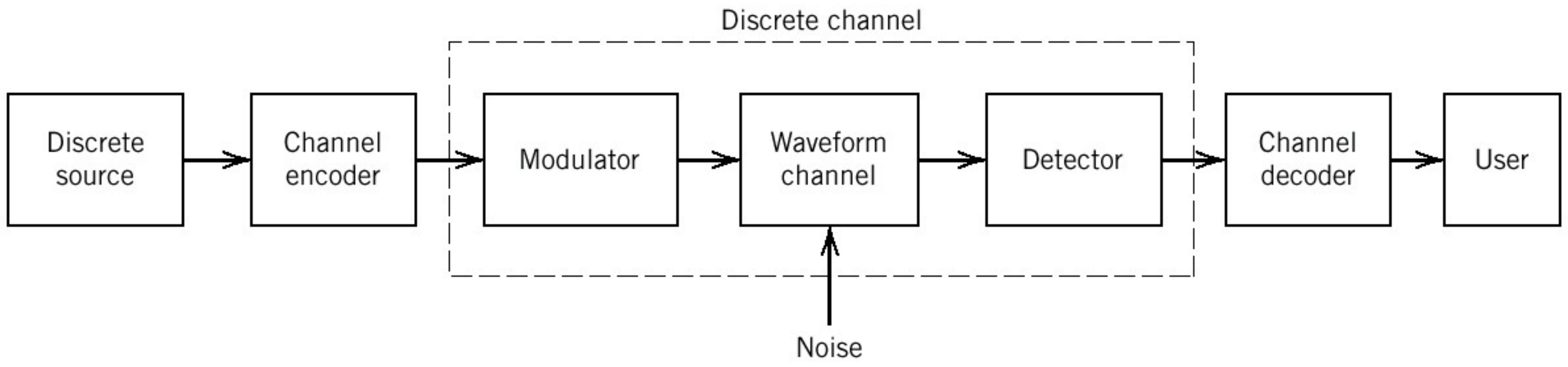
Why?
To increase the resistance of digital communication systems to channel noise via error control coding

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 minimised

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

Error Control Coding

Error control for data integrity may be exercised by means of forward error correction (FEC).



The discrete source generates information in the form of binary symbols.
The channel encoder accepts message bits and adds redundancy to produce encoded data at higher bit rate.
The channel decoder uses the redundancy to decide which message bits were actually transmitted.

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

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

Block Codes

Usually in the form of (n,k) block code where n is the number of bits of the codeword and k is the number of bits for the binary message

To generate an (n,k) block code, the channel encoder accepts information in successive k-bit blocks

For each block add (n-k) redundant bits to produce an encoded block of n-bits called a code-word

The (n-k) redundant bits are algebraically related to the k message bits

The channel encoder produces bits at a rate called the <u>channel data rate, R_0</u>

$$R_0 = \left(\frac{n}{k}\right) R_s$$

Where R_s is the bit rate of the information source

and n/k is the code rate

Forward Error-Correction (FEC)

The channel encoder accepts information in successive k-bit blocks and for each block it adds (n-k) redundant bits to produce an encoded block of n-bits called a code-word.
The channel decoder uses the redundancy to decide which message bits were actually transmitted.
In this case, whether the decoding of the received code word is successful or not, the receiver does not perform further processing.
In other words, if an error is detected in a transmitted code word, the receiver does not request for retransmission of the corrupted code word.

Automatic-Repeat Request (ARQ) scheme

Upon detection of error, the receiver requests a repeat transmission of the corrupted code word
There are 3 types of ARQ scheme

- Stop-and-Wait
- Continuous ARQ with pullback
- Continuous ARQ with selective repeat

Types of ARQ scheme

Stop-and-wait

• A block of message is encoded into a code word and transmitted
• The transmitter <u>stops and waits</u> for feedback from the receiver either an acknowledgement of a correct receipt of the codeword or a retransmission request due to error in decoding.
• The transmitter resends the code word before moving onto the next block of message

Types of ARQ scheme

Continuous ARQ with pullback (or go-back-N)

•Allows the receiver to send a feedback signal while the transmitter is sending another code word

•The transmitter continues to send a succession of code words until it receives a retransmission request.

•It then stops and pulls back to the particular code word that was not correctly decoded and retransmits the complete sequence of code words starting with the corrupted one.

Continuous ARQ with selective repeat

•Retransmits the code word that was incorrectly decoded only.

•Thus, eliminates the need for retransmitting the successfully decoded code words.

Figure 13.1-7

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

Code Vectors

Any n-bit code word can be visualised in an n-dimensional space as a vector whose elements having coordinates equal the bits in the code word

For example a code word 101 can be written in a row vector notation as (1 0 1)

Matrix representation of block codes

The code vector can be written in matrix form:

A block of k message bits can be written in the form of 1-by-k matrix

Modulo-2 operations

The encoding and decoding functions involve the binary arithmetic operation of modulo-2

Rules for modulo-2 operations are…..

Modulo-2 operations

The encoding and decoding functions involve the binary arithmetic operation of modulo-2

Rules for modulo-2 operations are:

Modulo-2 addition

0 + 0 = 0

1 + 0 = 1

0 + 1 = 1

1 + 1 = 0

Modulo-2 multiplication

0 x 0 = 0

1 x 0 = 0

0 x 1 = 0

1 x 1 = 1

Linear Block Code – Example : The Repetition Code

The additional (redundancy) bits (n-k) are identical to k

Example : A (5,1) repetition code.

The original binary message has 1 bit. (5-1=4) bits are added to the binary message to form a code word and the 4 additional bits are identical to the 1 bit binary message.
So, you have 2 code words either 11111 or 00000.
In the case of error, 1 will changed to 0 and/or vice versa and the decoder will know that it has wrongly received a code word.

Parity-check Codes

Codes are based on the notion of parity.

The parity of a binary word is said to be even when the word contains and even number of 1s and odd parity when it has odd number of 1s.

A group of n-bits codewords are constructed from a group of n-1 message bits. One check bit is added to the n-1 message bits such that all the codewords have the same parity

When the received codeword has different parity, we know that an error has occurred

Example : n=3 and even parity

The binary message are 00,01,10,11

The check bit is added such that all the code words have even parity

So, the resulting code words are $00\underline{0}$,$01\underline{1}$,$10\underline{1}$ and $11\underline{0}$

Systematic 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

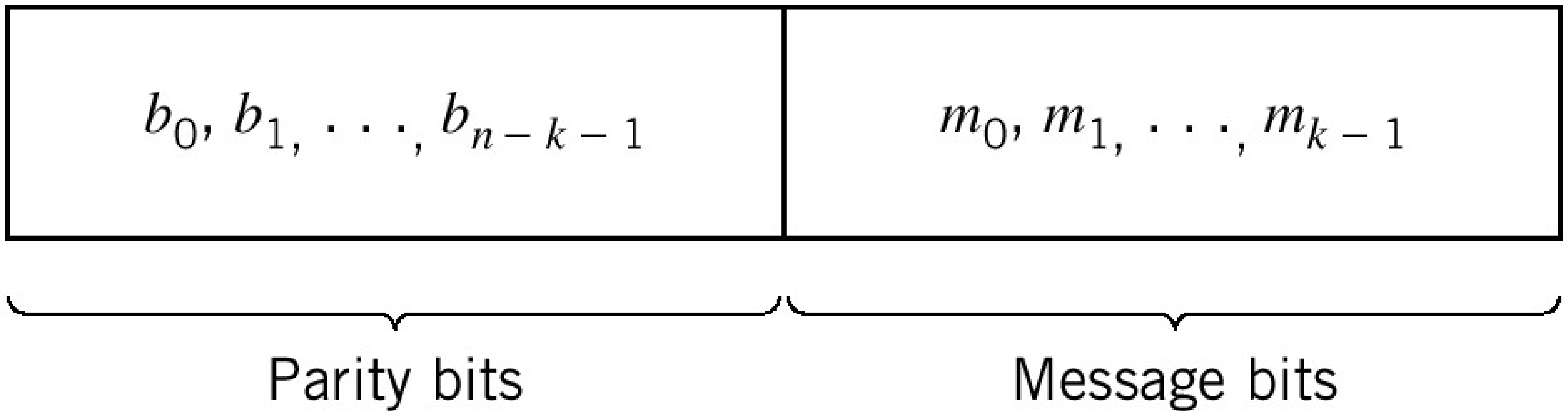
$$(c_0 \ c_1 \ \ldots \ c_n) \ , \quad (m_0 \ m_1 \ \ldots \ m_{k-1}) \ \textbf{and} \ (b_0 \ b_1 \ \ldots \ b_{n-k-1})$$

Systematic 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 <u>systematic</u> 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

Systematic Block Codes

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

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

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.

Systematic 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

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

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.

Hamming Code

A type of (n, k) linear block codes with the following parameters

- Block length, $n = 2^m - 1$
- Number of message bits, $k = 2^m – m - 1$
- Number of parity bits : $n-k=m$
- $m >= 3$

Hamming Code – Example

A (7,4) Hamming code with the following parameters

n=7; k=4, m=7-4=3

Hamming Code – Example

The parity vector,b is generated by **b=m.P**

For a given block of message bits m = (m1 m2 m3 m4), we can work out the parity vector, b and hence the code word, **c = mG** for the (7,4) Hamming Code.

Exercise: Try to work out the codewords for the (7,4) Hamming Code.

Codewords for (7,4) Hamming Code

Message Word	Parity bits	Code words
0000	000	0000000
0001	101	1010001
0010	111	1110010
0011	010	0100011
0100	011	0110100
0101	110	1100101
0110	100	1000110
0111	001	0010111
1000	110	1101000
1001	011	0111001
1010	001	0011010
1011	100	1001011
1100	101	1011100
1101	000	0001101
1110	010	0101110
1111	111	1111111