ADVANCED COMMUNICATION LAB *MANUAL*

VI Semester (19ET6DLADC)

Name of the Student :

Semester /Section :

USN :

Batch :

Vision of the Institute

To impart quality technical education with a focus on Research and Innovation emphasizing on Development of Sustainable and Inclusive Technology for the benefit of society.

Mission of the Institute

- To provide an environment that enhances creativity and Innovation in pursuit of Excellence.
- To nurture teamwork in order to transform individuals as responsible leaders and entrepreneurs.
- To train the students to the changing technical scenario and make them to understand the importance of Sustainable and Inclusive technologies.

Name of the Student :

Semester /Section :

USN :

Batch :

DAYANANDA SAGAR COLLEGE OF ENGINEERING

(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)

DEPARTMENT OF TELECOMMUNICATION ENGINEERING, BENGALURU-560078

VISION OF THE DEPARTMENT

- ***To disseminate quality technical education for achieving Academic Excellence through focused research encompassing Mobile, Sensor, and Telecommunication networks with a thrust on Space Communication and Telecommunication Standards.***

MISSION OF THE DEPARTMENT

- ***By disseminating the knowledge of devices, systems, and technologies that are impacting the Telecommunication field.***
- ***By educating students towards multidisciplinary practices of Telecommunication industry and standards for a successful career.***
- ***By inculcating research attitude among graduates through continuous life long learning related to Telecommunication and its allied domain***

PROGRAMME EDUCATIONAL OBJECTIVES [PEOs]

Program Educational Objectve-1 (PEO1): Graduate contributes service to Telecommunication Engineering industry, government organizations, and society by applying skills acquired through formal technical education.

Program Educational Objectve-2 (PEO2): The graduate demonstrates ability to apply Scientific, Mathematical and Telecommunication Engineering knowledge for solving real life problems.

Program Educational Objectve-3 (PEO3): The graduate possess temperament of professional attitude, effective communication skills, and multidisciplinary approach to resolve Telecommunication domain specific challenges.

Program Educational Objectve-4 (PEO4): The graduate demonstrates continuous learning to update knowledge of emerging Telecommunication technologies through higher education, for sustainable career development to strengthen human values, with a focus on safety and environmental concern.

PROGRAMME SPECIFIC OUTCOMES [PSOs]

PSO – 1: To analyze, design and implement Engineering solutions for Signal Processing application and Communication Systems.

PSO – 2 : To design, synthesize and provide solutions for Communication Network Applications

PSO – 3: To analyze design and synthesize Embedded based System for real time scenarios.

ADVANCED COMMUNICATION LAB

Course code:	19ET6DLADC	**Credits:**	02
L: P: T: S:	0:2:1:0	**CIE Marks:**	50
Exam Hours:	03	**SEE Marks:**	50

Course Objectives

1. Study and analyze analog and digital communication using optical fiber
2. Understand the various concepts related to wireless communication like modulation techniques, free space propagation, indoor and outdoor propagation Model .
3. Implement the various Forward Error Correction Codes and evaluate the performance of the various encoding and decoding techniques

Course Outcomes :After completion of the course, the graduates will be able to

CO1 To analyze analog and digital communication link using optical fiber and examine different losses.
CO2 To implement different wireless modulation techniques and analyze its BER
CO3 To analyze and determine median path loss using different free space propagation models.
CO4 To implement encoding techniques for Linear block codes, cyclic codes, BCH codes and its different decoding approaches
CO5 To implement Convolution codes and its important Viterbi decoding technique.
CO6 To implement the fundamental concepts of Galois Field.

Mapping of Course outcomes to Program outcomes

	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO 1	PSO 2	PSO 3
CO1	3	3	3	3	3	1	1	1	1	3	1	2	3	1	3
CO2	3	3	3	3	3	1	1	1	1	3	1	2	3	1	3
CO3	3	3	3	3	3	1	1	1	1	3	1	2	3	1	3
CO4	3	3	3	3	3	1	1	1	1	3	1	1	3	1	3
CO5	3	3	3	2	3	1	1	1	1	3	1	1	3	1	3
CO6	3	3	3	3	3	1	1	1	1	3	1	1	3	1	3

List of Experiments

Part A

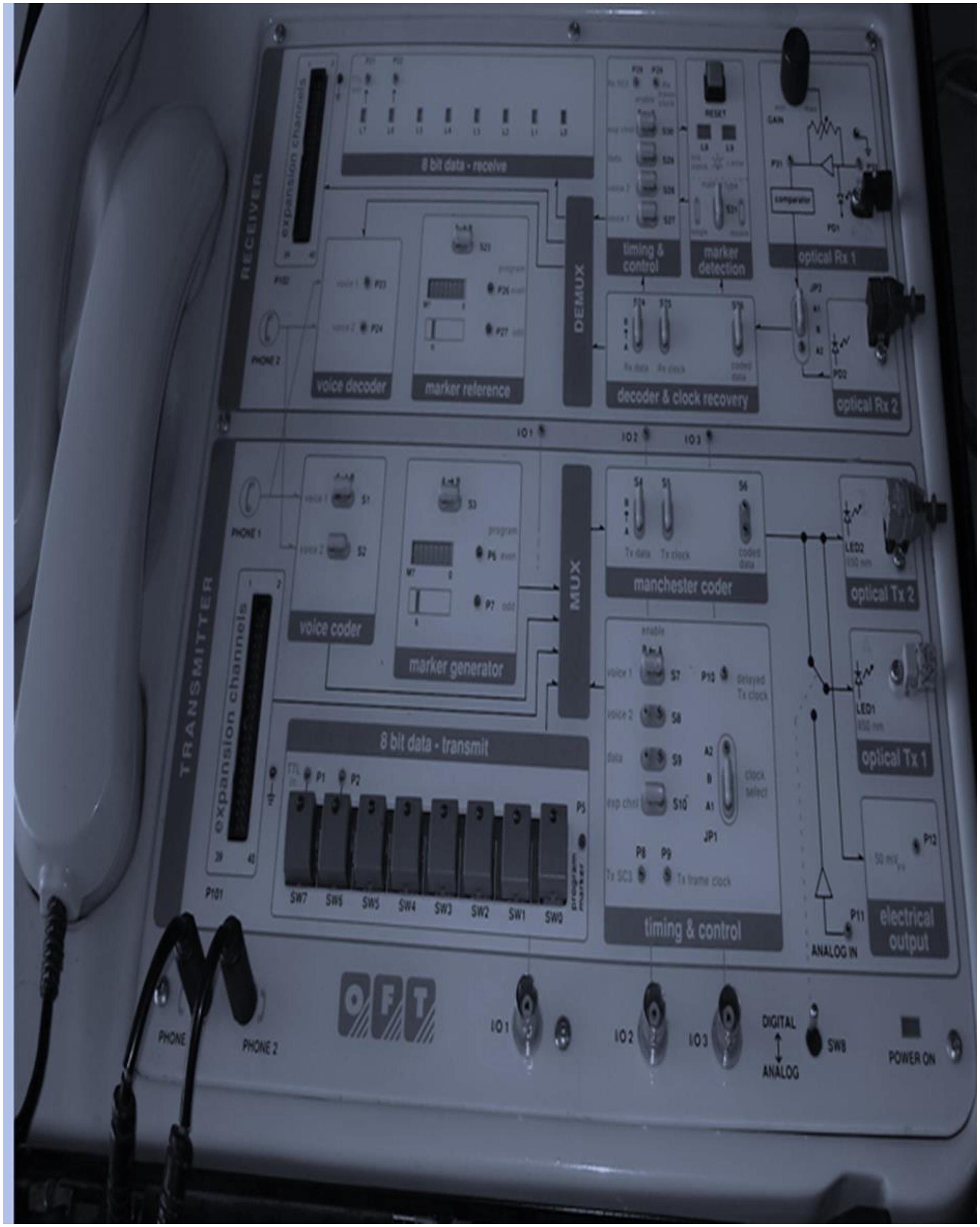
1. To set up a communication using 850nm wavelength optical fiber analog link
2. To set up a communication using 650nm wavelength optical fiber Digital link
3. To determine the free space loss and the power received using Matlab program
4. To write a Matlab program to calculate the median path loss for different outdoor propagation models.
5. To Simulate a 16 QAM Modulator/Demodulator using Simulink and analyse BER

6. Simulate Modulator/Demodulator of BPSK and QPSK and compare BER with 16-QAM

Part B

7. Encoding and Decoding (n,k) Hamming codes
8. Encoding and Decoding (n,k) cyclic codes
9. Realize Galois field GF(2^m) and construct the addition and multiplication table
10. Encoding and Decoding of Binary BCH Codes
11. Encoding and Decoding of Reed Solomon Codes
12. Encoding and Decoding of Convolution Codes.

OPTICAL FIBER EXPERIMENTS



Optical Fiber Trainer Kit

The Benchmark Optical Fiber Trainer - OFT - is the base-line for every fiber optic laboratory. It demonstrates state-of-the-art concepts in fiber optic communications. It provides for easy experimentation and effectively bridges the gap between a leading-edge technology and currently available technical training tools. As the concepts demonstrated are no different from those encountered in real-life systems, it can provide significant insights, of direct relevance, to the student and practicing engineers alike. In addition, it can serve as a ready-made communications platform for photo typing and in applications calling for fibre optic capabilities.

Features

- Eleven usable 64 kbps channels
- User definable frame marker (two alternating 8-bit markers - can be set to CCITT compatible)
- On-board two digitized voice channels, one 8-bit data channel and several user-expansion channels
- Demonstrates fully operational integrated voice/data fibre-optic communication link
- RS - 232C communications module optional - demonstrates computer communications over fibre
- Time Division Multiplexing of voice, data & user-defined data streams
- Modular design enables configuration with user-defined modules
- Wide scope for experimentation through use of external circuitry interfaced to kit
- Comprehensive manual describes wide range of experiments - can form basis of courses
- Ready -to -use kit comes complete with accessories

Optical	
Wavelengths	650nm and 850nm
FWHM spectral width	100nm
Fibre	1000 micron plastic fibre (1m, 3m and 20m lengths included)
Max link length	30m for 650nm optical digital link
	5m for 850nm link
Max data rate	2 Mbits/sec (NRZ)

Experiment No: 1 Date: __________

Analog communication link using optical fiber

Aim: To set up an 850nm fiber optic analog link.

Components required: OPTICAL FIBER TRAINER KIT
Oscilloscope, function generator- 1Hz to 10MHZ.

Introduction: The experiment is designed to familiarize the user with OFT. An analog fiber optic link is to be set up in this experiment. The LED-850nm LED. The fiber is a multimode fiber with a core diameter of 1000µm. The detector is a simple PIN detector.

The LED Optical power output is directly proportional to the current driving the LED. Similarly for the PIN diode the current is proportional to the amount of light falling on the detector. Thus even though the LED and the PIN DIODE are non-linear devices the current in the PIN diode are non-linear devices the current in the PIN diode is directly proportional to the driving current of the LED. This makes the optical communication system a linear system.

The fiber used in optical fiber trainer kit is Multimode Plastic fiber with 1000µm core diameter. unlike its glass-glass and plastic coated silica fiber counterparts this fiber has very high attenuation. It is useful mainly for short links such as in local area networks especially where there could be serious EMI problems. While the loss in plastic fiber is high for all wavelength regions, the loss at 850nm is much higher than at 650nm.

Procedure:

1. Set the switch SW8 to the analog position. Switch the power on.
2. Feed a 1Vp_p (peak to peak) sinusoidal signal at 1KHz from a function generator to the analog in post P11 using the following procedure.

i) Connect a BNC-BNC Cable from the function generator to the BNC socket I/01

ii) Connect the signal post I/02 to the analog IN Post P11 using a patch chord.

3. Connect one end of the 1m fiber to the LED Source LED1 in the optical Tx1 block.
4. Connect I/03 to P31 of optical receiver1 using a connecting wire or a patch chord.

5. Connect I/01 and I/02 using a connecting wire or a patch chord.

6. Connect a BNC-BNC cable from the I/02 to the CRO and assign that as the input signal and set the input frequency 1kHZ and 2Vp-p in the function generator.

7. Connect a BNC-BNC Cable from I/ 03 socket to CRO TO view the output signal.

8. Adjust gain knob in trainer kit such that no clipping of output wave forms takes place and does not disturb the gain throughout the experiment. Note down the output voltages and calculate the gain as per the tabular column.

   Tabular column:

Freq(Hz)	Output voltage (Vo)	Gain =20log Vo/Vi

Experiment 2: Set up an fiber optic analog link: calculation of attenuation loss and bending loss in analog link 850nm LED.

Procedure:

1. Set up an fiber analog link as explained in the first experiment and note down the input voltage and output voltage values with a 1m fiber connected between optical reveicer1. Set the values 1Vp-p and 10KHz sinusoidal signal with zero dc.

2. Repeat the same for 3m fiber and note as per the tabular column.

3. Calculate attenuation loss as per the formula

   $(P3/ P1) = (V3/ V1) = \exp (-\alpha (L3 - L1))$

   Where α is attenuation in neper/m and P1 and P3 are received optical powers with 1m and 3m fiber and calculate where α^1 is the fiber loss or the attenuation loss

α^1 (db/m) = 4.343 α (m^{-1})

Tabular column:

Fiber length in m	Input		Output	
	Voltage (v)	Freq (Hz)	Voltage (v)	Freq (Hz)
1m				
3m				
5m				

Bending loss:

	750 cm Fiber	1m Fiber
Diameter in cm	Voltage(v) loss	Voltage(v) loss
2cm		
3cm		
7cm		

To find critical radius of curvature: **$R_c = (3n_1^2\lambda) / (4\pi (n_1^2 - n_2^2)^{3/2})$**

n_1= 1.46 for plastic and n_2 =1.458 for silica λ= 850nm

Experiment No: 2 Date: ________

Digital communication link using optical fiber

AIM: To set up a 650nm digital link and to measure the maximum bit rate can be transmitted.

Introduction: The OFT trainer kit can be used to set up two fiber optic digital links one at wavelengths of 650nm and other at 850nm. LED1--850nm LED2---650nm.

PD1 in the optical rx1 block is a PIN detector which gives a current proportional to the optical power falling on the detector. The received signal is amplified and converted to a TTL signal using a comparator. The gain control plays a crucial role in this conversion.PD2 in the optical receiver block is another receiver which directly gives out TTL Signals.

Equipments required: OPTICAL FIBER TRAINER KIT
Oscilloscope, function generator- 1Hz to 10MHZ.

Procedure:

I. Set the switch SW8 to the digital position. Switch the power on.

II. Connect a 1m optical fiber between LED2 650nm optical tx2 and the PIN detector1. Feed a 5Vp-p square wave signal at 10kHz from the function generator using the following procedure.

- Connect a BNC-BNC Cable from the function generator to the BNC socket I/01

III. Connect I/03 to P31 of optical receiver1 using a connecting wire or a patch chord.

IV. Connect I/01 and I/02 using a connecting wire or a patch chord.

V. Connect a BNC-BNC cable from the I/02 to the CRO and assign that as the input signal and set the input frequency 10kHZ and 5Vp-p in the function generator.

VI. Connect a BNC-BNC Cable from I / 03 socket to CRO TO view the output signal.

VII. Adjust gain knob in trainer kit such that no clipping of output wave forms takes place and does not disturb the gain throughout the experiment. Note down the output voltages and calculate the gain as per the tabular column.

Tabular column:

Freq(Hz)	Output voltage (Vo)	Gain =20log Vo/Vi

Coupling loss:

1. Connect one end of the 1m fiber to LED 2 and the other end to the detector PD1.
2. Drive the LED with a 10kHz TTL Signal and note down asV1 and disconnect the fiber from the detector.
3. Take the 3m fiber and connect one end to the detector PD1 The optical signal can be seen emerging from the other end of the 1m fiber.
4. Bring the free ends of the 2 fibers using the possible and align them as shown in fiber alignment unit. Note down as V4.
5. Calculate the coupling loss using the formula

$$\eta = -10 \log (V_4/V_1) - \alpha^{1} (L_3 + L_1)$$

Numerical aperture:

Numerical aperture of a fiber is a measure of the acceptance angle of light in the fiber. Light which is launched at angles does not get coupled to propagating modes in the fiber and therefore does not reach the receiver at the other end of the fiber.

Procedure:

1. To set up an 650nm digital with 1m fiber. Drive a 5Vp-p square wave signal of 10KHz

Insert the other end of fiber into numerical aperture measurement unit and adjust the fiber such that its tip is 0.5cm, 1cm from the screen and calculate the numerical aperture using the formula.

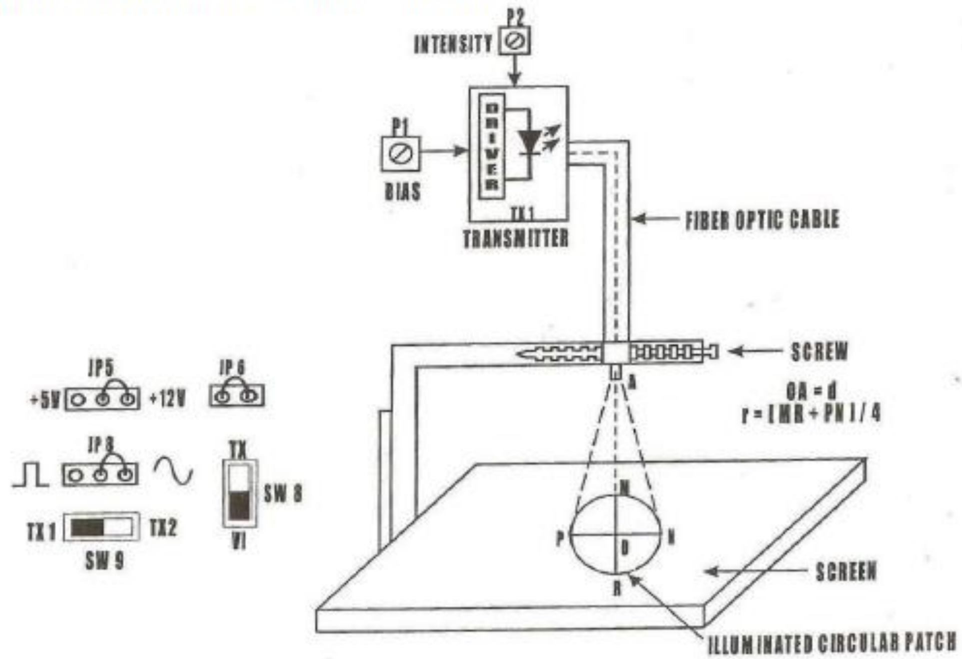
$$X = \frac{DE+BC}{4}$$

d= the distance from tip of the fiber till the white screen

Numerical aperture = Sinθ= X / (d^2 +x^2) $^{1/2}$

$$\text{Sin}\theta = \frac{x}{(d^2+x^2)^{1/2}}$$

Circuit Diagram-6: Set up to measure NA



Experiment No: 3a Date: __________

FREESPACE PROPAGATION – PATH LOSS MODEL

Aim: To determine the free space loss and the power received using Matlab program.

Theory:

The free space path loss, also known as FSPL is the loss in signal strength that occurs when an electromagnetic wave travels over a line of sight path in free space. In these circumstances there are no obstacles that might cause the signal to be reflected refracted, or that might cause additional attenuation.

The free space path loss calculations only look at the loss of the path itself and do not contain any factors relating to the transmitter power, antenna gains or the receiver sensitivity levels.

To understand the reasons for the free space path loss, it is possible to imagine a signal spreading out from a transmitter. It will move away from the source spreading out in the form of a sphere. As it does so, the surface area of the sphere increases. As this will follow the law of the conservation of energy, as the surface area of the sphere increases, so the intensity of the signal must decrease.

As a result of this it is found that the signal decreases in a way that is inversely proportional to the square of the distance from the source of the radio signal

Free space path loss formula

The free space path loss formula or free space path loss equation is quite simple to use. Not only is the path loss proportional to the square of the distance between the transmitter and receiver, but the signal level is also proportional to the square of the frequency in use.

$$FSPL = (4\pi d/ \lambda)2 = (4\pi df/ c)2$$

FSPL is the Free space path loss, **d** is the distance of the receiver from the transmitter (metres), **λ** is the signal wavelength (metres), **f** is the signal frequency (Hertz), **c** is the speed of light in a vacuum (metres per second)

The free space path loss formula is applicable to situations where only the electromagnetic wave is present, i.e. for far field situations. It does not hold true for near field situations.

Decibel version of free space path loss equation

Most RF comparisons and measurements are performed in decibels. This gives an easy and consistent method to compare the signal levels present at various points. Accordingly it is very convenient to express the free space path loss formula, FSPL, in terms of decibels..

FSPL (dB) = 20 log10 (d) + 20 log10 (f) + 32.44

Where: **d** is the distance of the receiver from the transmitter (km) **f** is the signal frequency (MHz)

Affect of antenna gain on path loss equation

The equation above does not include any component for antenna gains. It is assumed that the antenna gain is unity for both the transmitter. In reality, though, all antennas will have a certain amount of gain and this will affect the overall affect. Any antenna gain will reduce the "loss" when compared to a unity gain system. The figures for antenna gain are relative to an isotropic source, i.e. an antenna that radiates equally in all directions.

FSPL (dB) = 20 log10 (d) + 20 log10 (f) + 32.44 -Gtx - Grx

Where: **Gtx** is the gain of the transmitter antenna relative to an isotropic source (dBi) **Grx** is the gain of the receiver antenna relative to an isotropic source (dBi)
The free space path loss equation or formula given above, is an essential tool that is required when making calculations for radio and wireless systems either manually or within applications such as wireless survey tools, etc. By using the free space path loss equation, it is possible to determine the signal strengths that may be expected in many scenarios. While the free space path loss formula is not fully applicable where there are other interactions, e.g. reflection, refraction, etc as are present in most real life applications, the equation can nevertheless be used to give an indication of what may be expected. It is obviously fully applicable to satellite systems where the paths conform closely to the totally free space scenarios

Power Received :

[Pr] = [Pt] + [Gt] + [Gr] – [FSPL]

Pr – Received power Pt – Transmitted power
Gt – Gain of the transmitting antenna Gr – Gain of the receiving antenna
Program:

```
clc;
close all;
clear all;
f=input('enter the frequency in Mhz: ');
L=300/f; %calculating wavelength
disp('thus the wavelength is: ');
L %displaying wavelength
d=input('enter the distance in km: ');
Gt=input('enter the transmitting antenna gain in db: ');
Gr=input('enter the receiving antenna gain in db: ');
Pt=input('enter the transmitted power in db: ');
ls=32.44+20*log10(d)+20*log10(f); %calculating path loss
disp(sprintf('%s %d %s','the path loss is:',ls,'db'));%displaying path loss
Pr=Pt+Gt+Gr-ls; %calculating recieved power in db
disp(sprintf('%s %d %s','the recieved power is:',Pr,'db'));
Pr=10^(Pr/10); %calculating recieved power in watts
disp(sprintf('%s %d %s','the recieved power is:',Pr,'watts'));
%displaying recieved power in watts
```

Result:

The program for power received by an antenna and path loss in Free space propagation was simulated successfully.

Experiment No: 3b   Date: ___________

OUTDOOR PROPAGATION MODEL - OKUMURA MODEL and HATA MODEL

Aim:

write a Matlab program to calculate the median path loss for Okumura model and Hata model for outdoor propagation.

OKUMURA MODEL

The Okumura model for Urban Areas is a Radio propagation model that was built using the data collected in the city of Tokyo, Japan. The model is ideal for using in cities with many urban structures but not many tall blocking structures. The model served as a base for the Hata Model.

Okumura model was built into three modes: the ones for urban, suburban and open areas. The model for urban areas was built first and used as the base for others.

Coverage

Frequency = 150 MHz to 1920 MHz
Mobile Station Antenna Height: between 1 m and 10 m
Base station Antenna Height: between 30 m and 1000 m
Link distance: between 1 km and 100 km

Mathematical formulation

The Okumura model is formally expressed as:

$L = L_{FSL} + A_{MU} - H_{MG} - H_{BG} - \Sigma K_{CORRECTION}$

where,

L = The median path loss. Unit: Decibel (dB)
L_{FSL} = The Free Space Loss. Unit: Decibel(dB)
A_{MU} = Median attenuation.Unit: Decibel(dB)
H_{MG} = Mobile station antenna height gain factor.
H_{BG} = Base station antenna height gain factor.
$K_{correction}$ = Correction factor gain (such as type of environment, water surfaces, isolated obstacle etc.)

Okumura model does not provide a mean to measure the Free space loss. However, any standard method for calculating the free space loss can be used.

Program:

```
clc;
clear all;
close all;
Lfsl=input('enter the free space loss:');
Amu=input('enter the median attenuation value:');
Hmg=input('enter the Mobile station antenna height gain factor:');
Hbg=input('enter the Base station antenna height gain factor:');
Kc=input('enter the Correction factor gain:');
L=Lfsl+Amu-Hmg-Hbg-Kc; %calculating median path loss
disp(sprintf('%s %f %s','the median path loss:',L,'dB'));
```

HATA MODEL

In wireless communication, the Hata Model for Urban Areas, also known as the *Okumura-Hata model* for being a developed version of the Okumura Model, is the most widely used radio frequency propagation model for predicting the behavior of cellular transmissions in built up areas. This model incorporates the graphical information from Okumura model and develops it further to realize the effects of diffraction, reflection and scattering caused by city structures. This model also has two more varieties for transmission in Suburban Areas and Open Areas.

Hata Model predicts the total path loss along a link of terrestrial microwave or other type of cellular communications.

This particular version of the Hata model is applicable to the radio propagation within urban areas. This model is suited for both point-to-point and broadcast transmissions and it is based on extensive empirical measurements taken. PCS is another extension of the Hata model. The Walfisch and Bertoni Model is further advanced.

Coverage

Frequency: 150 MHz to 1500 MHz
Mobile Station Antenna Height: between 1 m and 10 m
Base station Antenna Height: between 30 m and 200 m
Link distance: between 1 km and 20 km.

Mathematical formulation

Hata Model for Urban Areas is formulated as:

$$L_U = 69.55 + 26.16 \log f - 13.82 \log h_B - C_H + [44.9 - 6.55 \log h_B] \log d.$$

For small or medium sized city,

$$C_H = 0.8 + (1.1 \log f - 0.7) h_M - 1.56 \log f.$$

and for large cities,

$$C_H = 8.29 (\log (1.54 h_M))^2 - 1.1, \text{ if } 150 \le f \le 200$$

$$C_H = 3.2 (\log (11.75 h_M))^2 - 4.97, \text{ if } 200 \le f \le 1500$$

Where,
L_U = Path loss in Urban Areas (dB)
h_B= Height of base station Antenna. (m)
h_M = Height of mobile station Antenna. (m)
f= Frequency of Transmission (MHz).

C_H = Antenna height correction factor
d= Distance between the base and mobile stations (km).

The term "small city" means a city where the mobile antenna height not more than 10 meters. i.e. $1 \leq h_M \leq 10m$

Program:

```
clc;
clear all;
close all;
f=input('enter the frequency of transmisson in mhz:');
Hb=input('enter the height of base station Antenna in meter:');
Hm=input('enter the height of mobile station Antenna in meter:');
d=input('enter the distance between the base and mobile stations:');
n=input('enter 0 for small city and 1 for large city:');
if n==0
ch=0.8+(1.1*log10(f)-0.7)*Hm-1.56*log10(f);
else
if f>=150 && f<=200
ch=8.29*(log10(1.54*Hm))^.2-1.1;
else
if f>=200 && f<=1500
ch=3.2*(log10(11.75*Hm))^.2-4.97;
end;
end;
end;
Lu=69.55+26.26*log10(f)-13.82*log10(Hb)-ch+(44.9-6.55*log10(Hb))*log10(d);
disp(sprintf('%s %f %s','Path loss in Urban Areas=',Lu,'db'));
```

Result:

The program for Okumura Model and Hata Model – Outdoor Propagation was simulated successfully

Experiment No: 4

Date: ___________

Aim: To build a Simulink model of 16 QAM Modulator and Demodulator

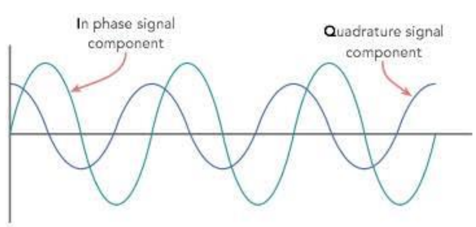
Software required: Matlab-Simulink-2015a

Theory:

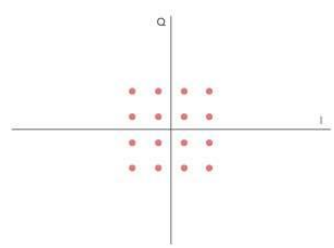
Quadrature Amplitude Modulation (QAM) combines amplitude & phase changes to give additional capacity & is widely used for data communications.

QAM is a signal in which two carriers shifted in phase by 90 degrees (i.e. sine and cosine) are modulated and combined. As a result of their 90° phase difference they are in quadrature and this gives rise to the name. Often one signal is called the In-phase or "I" signal, and the other is the quadrature or "Q" signal.

The resultant overall signal consisting of the combination of both I and Q carriers contains of both amplitude and phase variations. In view of the fact that both amplitude and phase variations are present it may also be considered as a mixture of amplitude and phase modulation.

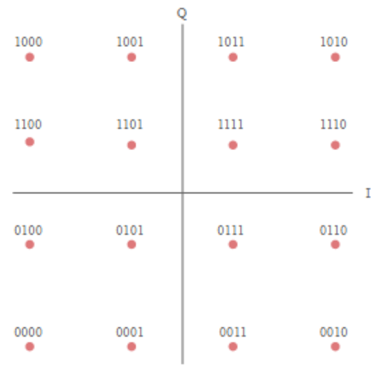


Quadrature amplitude modulation concept



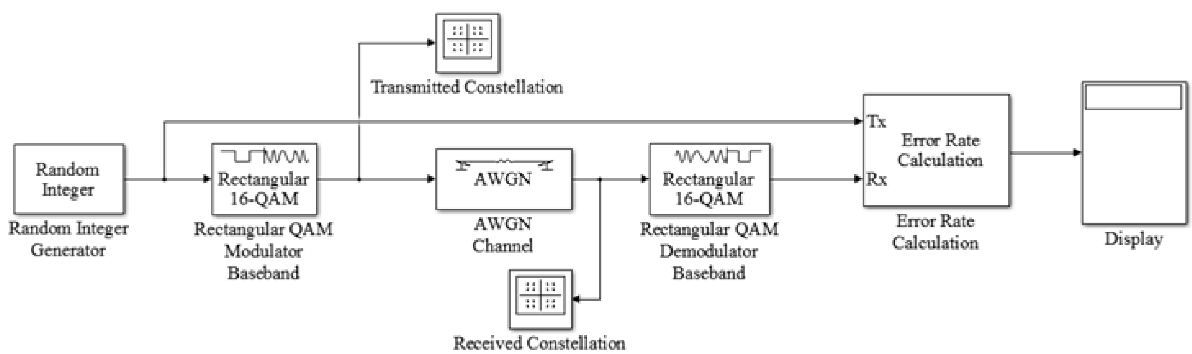
Constellation Diagram

As shown above, the constellation points are typically arranged in a square grid with equal horizontal and vertical spacing. Although data is binary the most common forms of QAM, although not all, are where there constellation can form a square with the number of points equal to a power of 2 i.e. 4 16, 64 . . . . , i.e. 16QAM, 64QAM, etc



Bit sequence mapping for a 16QAM signal

The Simulink model of 16 QAM modulator and demodulator is shown below



16 QAM Simulation Model

- Build the Simulink model shown in Fig
- Double-click on the Random Integer Generator and adjust the set size to a proper value (Remember that the input to the 16 QAM modulator should be from the set {0, 1, 2, …, 15}).
- In the Random Integer Generator block, set the Sample Time to 0.25e-6 (i.e. 1 μs)
- In the AWGN block, set the Symbol period parameter to 1e-6 (i.e. 1 μs) and the Number of bits per symbol parameter to 4 (since 16 QAM uses 4 bits per symbol).
- For the Error Rate Calculation block, set the Output data field to “port” so you can connect the Display block.
- The Display Block will show you three values. The first value is the BER, the second value is the number of incorrect bits, and the third value is the total number of bits received.
- Set the simulation time to 10 seconds.

- In both 16 QAM Modulator and Demodulator blocks, set the Constellation ordering to Gray, set the Normalization method to Peak Power, and set the value of the Peak power to 1 Watt.
- In this experiment, you will adjust the value of the Eb/No in the AWGN block, starting from 3, incrementing by 1 every step, and ending at 15, and observe the error rate displayed in the Display block. Make a table recording the value of Eb/No and the corresponding BER.
- Plot BER vs. Eb/No and compare with the theoretical values. Comment on the results.

Results:

16- QAM BER VS Eb/No

Eb/No	Bit error rate (BER)

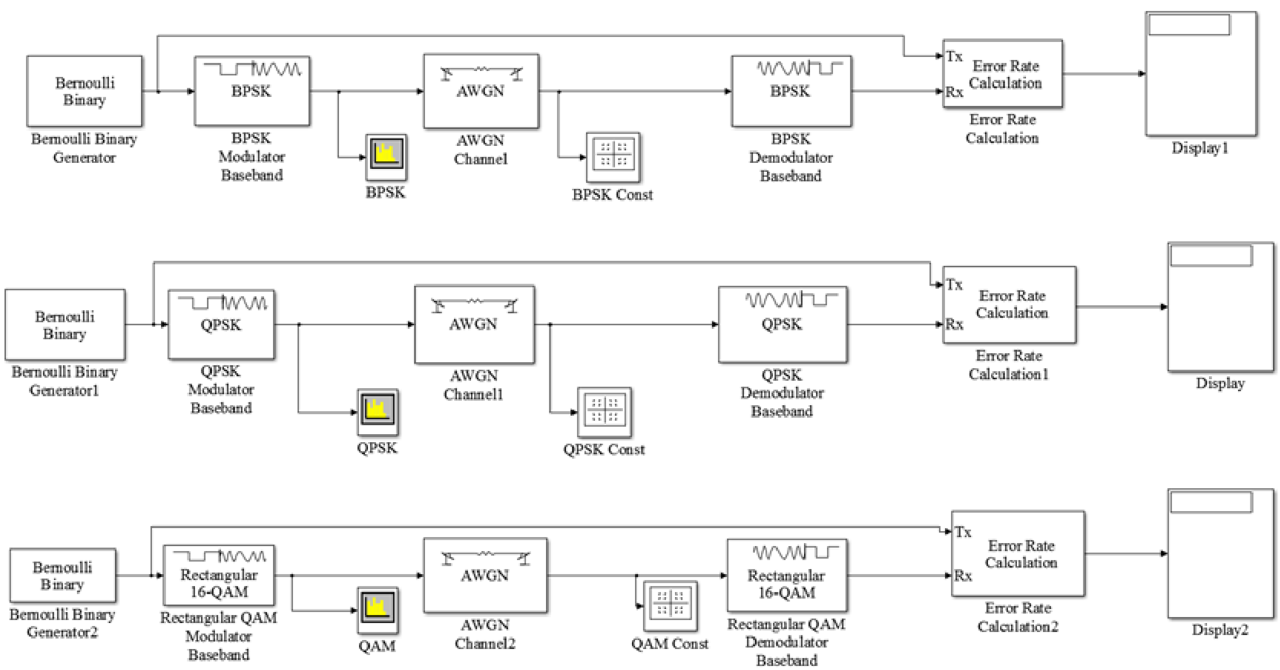
Experiment No: 5 Date: ________

Aim: To build a Simulink model of BPSK and QPSK and compare with 16 QAM Modulato/Demodulator

Software required: Matlab-Simulink-2015a

Comparing 16 QAM, QPSK, and BPSK

In this experiment, you will learn about the trade-off of using 16 QAM, QPSK, and BPSK.



16 QAM vs. QPSK vs. BPSK

- Build the Simulink model shown in Figure.
- Double click on the Bernoulli generator for the BPSK part. Set the sample time to 1e-6 Double click on the Bernoulli generator for the QPSK part. Set the sample time to 0.5e-6
- Double click on the Bernoulli generator for the 16 QAM part. Set the sample time to 0.25e-6
- For the 16 QAM Modulator and Demodulator blocks, use Gray Constellation ordering, set the Normalization method to Peak Power, and set the Peak Power value to 1 Watt.
- Choose the same value of SNR for both AWGN blocks

- Set the simulation time to 10 seconds.
- Run the simulation and observe the bit error rate and the number of transmitted samples from the Display block for both schemes, and observe the used bandwidth for both schemes from the spectrum analyzer block. **Explain** your observations.

PART B

Information Theory and Error Control Coding

Noise or Error is the main problem in the signal, which disturbs the reliability of the communication system. **Error control coding** is the coding procedure done to control the occurrences of errors. These techniques help in Error Detection and Error Correction.

There are many different error correcting codes depending upon the mathematical principles applied to them. But, historically, these codes have been classified into **Linear block codes** and **Convolution codes**.

Linear Block Codes:

In the linear block codes, the parity bits and message bits have a linear combination, which means that the resultant code word is the linear combination of any two code words.

Let us consider some blocks of data, which contains **k** bits in each block. These bits are mapped with the blocks which has **n** bits in each block. Here **n**is greater than **k**. The transmitter adds redundant bits which are **(n-k)** bits.

The **(n-k)** bits added here are **parity bits**. Parity bits help in error detection and error correction, and also in locating the data. In the data being transmitted, the left most bits of the code word correspond to the message bits and the right most bits of the code word correspond to the parity bits.

Convolution Codes:

Convolutional codes comes under a category of error correction codes in which an n bit symbol consists of each m bit information symbol to be encoded where code rate is m/n (n>=m) and the transformation acts as a function of the last K information symbols, here the constraint length of the code is denoted by K.

Experiment No: 7 Date: __________

Hamming Codes - Encoder and Decoder

Aim: To Encode and Decode using Hamming Codes

Tool Required : Matlab R2015a

Theory:

Hamming code is a set of error-correction codes that can be used to detect and correct bit errors that can occur when computer data is moved or stored. Hamming code is named for R. W. Hamming of Bell Labs.

Like other error-correction code, Hamming code makes use of the concept of parity and parity bits, which are bits that are added to data so that the validity of the data can be checked when it is read or after it has been received in a data transmission. Using more than one parity bit, an error-correction code can not only identify a single bit error in the data unit, but also its location in the data unit.

Matlab Code:

```
clear all;
clc;
close all;
n = 7; %# of code word bits per block
k = 4; %# of message bits per block
A = [ 1 1 1;1 1 0;1 0 1;0 1 1 ];%Parity sub matrix-Need binary(decimal combination of 7,6,5,3)
G = [ eye(k) A ];%Generator matrix
H = [ A' eye(n-k) ];%Parity-check matrix
% ENCODER%
msg = [ 1 1 1 1 ]; %Message block vector-change to any 4 bit sequence
code = mod(msg*G,2);%Encode message
% CHANNEL ERROR (add one error to code)%
%code (1) = ~code (1);
 code_e (2) = ~code (2);
%code (3) = ~code (3);
%code (4) = ~code (4); %Pick one, comment out others
%code (5) = ~code (5);
```

```
%code (6) = ~code (6);
%code (7) = ~code (7);
recd = cat(2,code(1),code_e(2),code(3:7)); %Received code word with error
% DECODER%
syndrome = mod(recd * H',2)
%Find position of the error in code word (index)
find = 0;
for ii = 1:n
   if ~find
      errvect = zeros(1,n);
      errvect(ii) = 1;
      search = mod(errvect * H',2);
      if search == syndrome
         find = 1;
         index = ii;
      end
   end
end
disp(['Position of error in codeword=',num2str(index)]);
correctedcode = recd;
correctedcode(index) = mod(recd(index)+1,2)%Corrected codeword
%Strip off parity bits
msg_decoded=correctedcode;
msg_decoded=msg_decoded(1:4)
```

Results:

```
Message
     1     1     1     1

Code Vector
     1     1     1     1     1     1     1

The Received Vector is
     1     0     1     1     1     1     1


syndrome =

     1     1     0

Position of error in codeword=2

correctedcode =

     1     1     1     1     1     1     1


msg_decoded =

     1     1     1     1
```

Experiment No: 8 Date: __________

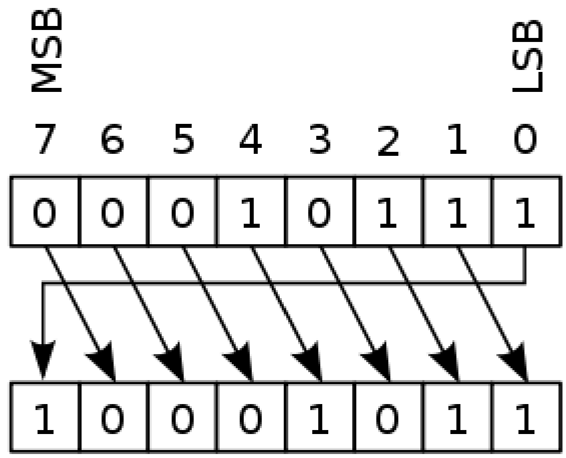
Cyclic Codes - Encoder and Decoder

Aim: To Encode and Decode using Cyclic Codes

Tool Required: Mat lab R2015a

Theory: In coding theory, a cyclic code is a block code, where the circular shifts of each code word gives another word that belongs to the code. They are error-correcting codes that have algebraic properties that are convenient for efficient error detection and correction.

If 00010111 is a valid code word, applying a right circular shift gives the string 10001011. If the code is cyclic, then 10001011 is again a valid code word. In general, applying a right circular shift moves the least significant bit (LSB) to the leftmost position, so that it becomes the most significant bit (MSB); the other positions are shifted by 1 to the right.



Matlab Code:

```
clc;
clear all;
close all;
N=7;
K=4;
polynomial=cyclpoly(N,K);% generator polynomial
[H,G,K]=cyclgen(N,polynomial); % Generator Matrix G and Parity check matrix H
% Encoder
msg = [ 1 0 1 1 ]; %Message block vector-change to any 4 bit sequence
code = mod(msg*G,2);% Code Vector
% CHANNEL ERROR(add one error to code)%
%code(1)= ~code(1);
 code_e(2)= ~code(2);
%code(3)= ~code(3);
```

```
%code(4)= ~code(4);%Pick one,comment out others
%code(5)= ~code(5);
%code(6)= ~code(6);
%code(7)= ~code(7);
recd = cat(2,code(1),code_e(2),code(3:7)); %Received codeword with error
% Decoder
t=syndtable(H);
msg_decoded = decode(recd,N,K,'cyclic/binary',polynomial,t);
disp('Message');
disp(msg);
disp('codeword');
disp(code);
disp('Received code');
disp(recd);
disp('Corrected codeword');
disp(msg_decoded);
```

```
Message
     1     0     1     1

codeword
     0     0     0     1     0     1     1

Received code
     0     1     0     1     0     1     1

Corrected codeword
     1     0     1     1

>> |
```

Experiment No: 9 Date: ___________

Galois Field elements and Galois Field arithmetic

Aim: To Generate Galois Field elements and Galois Field arithmetic

Tool Required: Mat lab R2015a

Theory:

Galois Field, named after Evariste Galois, also known as finite field, refers to a field in which there exist finitely many elements. It is particularly useful in translating computer data as they are represented in binary forms. That is, computer data consist of combination of two numbers, 0 and 1, which are the components in Galois field whose number of elements is two. Representing data as a vector in a Galois Field allows mathematical operations such as multiplication, addition, subtraction and division to scramble data easily and effectively.

Matlab Code:

```
clc;
clear all;
close all;
% Generation of Galois Field of GF(2^4)in binary using the primitive
% polynomial x^4+x+1
p = 2; m =4;
field = gftuple([-1:p^m-2]',m,p);

% Generation of Galois Field multiplication table
disp('Multiplication Table');
% Create Galois Field array in GF(2^4)
elements = gf([1:2^m-1]',m);
mtable = elements * elements'

%Generation of galois Field addition table
disp('Addition table');
e = repmat([0:2^m-1],2^m,1);% Replicates the values from 0 to 15, 16 times.
f = gf(e,m); % Create a Galois array.
atable = f + f' % Add f to its own matrix transpose
```

Results:

1. Array Elements

Element	Polynomial representation	Binary representation
0	0	(0000)
α^0	1	(1000)
α^1	X	(0100)
α^2	X^2	(0010)
α^3	X^3	(0001)
α^4	X+1	(1100)
α^5	X^2+X	(0110)
α^6	X^2+ X^3	(0011)

α^7	$1+X+X^3$	(1101)
α^8	$1+X^2$	(1010)
α^9	$X+X^3$	(0101)
α^{10}	$1+X+X^2$	(1110)
α^{11}	$X+X^2+X^3$	(0111)
α^{12}	$1+X+X^2+X^3$	(1111)
α^{13}	$1+X^2+X^3$	(1011)
α^{14}	$1+X^3$	(1001)

2. Multiplication Table

X	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
2	2	4	6	8	10	12	14	3	1	7	5	11	9	15	13
3	3	6	5	12	15	10	9	11	8	13	14	7	4	1	2
4	4	8	12	3	7	11	15	6	2	14	10	5	1	13	9
5	5	10	15	7	2	13	8	14	11	4	1	9	12	3	6
6	6	12	10	11	13	7	1	5	3	9	15	14	8	2	4
7	7	14	9	15	8	1	6	13	10	3	4	2	5	12	11
8	8	3	11	6	14	5	13	12	4	15	7	10	2	9	1
9	9	1	8	2	11	3	10	4	13	5	12	6	15	7	14
10	10	7	13	14	4	9	3	15	5	8	2	1	11	6	12
11	11	5	14	10	1	15	4	7	12	2	9	13	6	8	3
12	12	11	7	5	9	14	2	10	6	1	13	15	3	4	8
13	13	9	4	1	12	8	5	2	15	11	6	3	14	10	7
14	14	15	1	13	3	2	12	9	7	6	8	4	10	11	5
15	15	13	2	9	6	4	11	1	14	12	3	8	7	5	10

3. Addition Table

+	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	1	0	3	2	5	4	7	6	9	8	11	10	13	12	15	14
2	2	3	0	1	6	7	4	5	10	11	8	9	14	15	12	13
3	3	2	1	0	7	6	5	4	11	10	9	8	15	14	13	12
4	4	5	6	7	0	1	2	3	12	13	14	15	8	9	10	11
5	5	4	7	6	1	0	3	2	13	12	15	14	9	8	11	10
6	6	7	4	5	2	3	0	1	14	15	12	13	10	110	8	9
7	7	6	5	4	3	2	1	0	15	14	13	12	11	10	9	8
8	8	9	10	11	12	13	14	15	0	1	2	3	4	5	6	7
9	9	8	11	10	13	12	15	14	1	0	3	2	5	4	7	6
10	10	11	8	9	14	15	12	13	2	3	0	1	6	7	4	5
11	11	10	9	8	15	14	13	12	3	2	1	0	7	6	5	4
12	12	13	14	15	8	9	10	11	4	5	6	7	0	1	2	3
13	13	12	15	14	9	8	11	10	5	4	7	6	1	0	3	2
14	14	15	12	13	10	11	8	9	6	7	4	5	2	3	0	1
15	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Experiment No: 10 Date: __________

Binary BCH Codes -Encoder and Decoder

Aim: To Generate Binary BCH encoder and Decoder

Tool Required: Mat lab R2015a

Theory: In coding theory, the BCH codes or Bose–Chaudhuri–Hocquenghem codes form a class of cyclic error-correcting codes that are constructed using polynomials over a finite field(also called Galois field). One of the key features of BCH codes is that during code design, there is a precise control over the number of symbol errors correctable by the code. In particular, it is possible to design binary BCH codes that can correct multiple bit errors. Another advantage of BCH codes is the ease with which they can be decoded, namely, via an algebraic method known as syndrome decoding. This simplifies the design of the decoder for these codes, using small low-power electronic hardware.

Mat Lab Code:

```
clc;
clear all;
close all;
% Generator polynomial for Single error correcting BCH code over GF (2^4);
 % g1(x) =x4+x+1
 % Generator polynomial for double error correcting BCH code over GF (2^4);
 % g2(x) =x8+x7+x6+x4+1
 % Generator polynomial for triple error correcting BCH code over GF (2^4);
 % g3(x) =x10+x8+x5+x4+x2+x+1
m = 4;
n = 2^m-1;   % Codeword length
%If error correcting capability t=1, n=15, k=11;
%If error Correcting capability t=2, n=15, k=7;
%If error Correcting capability t=3, n=15, k=5;

k = 5;       % Message length
nwords=1; % number of words to be encoded
% Message that needs to be encoded
disp('Message');
msgTx =gf(randi([0 1],nwords,k))
[g,t] = bchgenpoly(n,k);
```

```
disp('Code Vector');
enc = bchenc(msgTx,n,k)
disp('Received data');
noisycode = enc + randerr(nwords,n,1:t)
disp('Decoded vector');
msgRx = bchdec(noisycode,n,k)
```

Results:

```
Message
msgTx = GF(2) array.
Array elements =
           1           0           1           1           1
Code Vector
enc = GF(2) array.
Array elements =
  Columns 1 through 10
           1           0           1           1           1           0           0           0           0           1
  Columns 11 through 15
           0           1           0           0           1
Received data
noisycode = GF(2) array.
Array elements =
  Columns 1 through 10
           1           0           1           1           0           0           0           0           0           1
  Columns 11 through 15
           0           1           0           0           1
Decoded vector
msgRx = GF(2) array.
Array elements =
           1           0           1           1           1
```

Experiment No: 11 Date: __________

Encoding and Decoding using Non-Binary BCH codes (Reed Solomon Codes) Over GF (2^3)

Aim: To Generate Encoder and Decoder using Non-Binary BCH codes (Reed Solomon Codes) Over GF (2^3)

Tool Required: Mat lab R2015a

Theory: Reed-Solomon codes are block-based error correcting codes with a wide range of applications in digital communications and storage. Reed-Solomon codes are used to correct errors in many systems including:

1. Storage devices (including tape, Compact Disk, DVD, barcodes, etc)
2. Wireless or mobile communications (including cellular telephones, microwave links, etc)
3. Satellite communications
4. Digital television / DVB
5. High-speed modems such as ADSL, xDSL, etc.

The Reed-Solomon encoder takes a block of digital data and adds extra "redundant" bits. Errors occur during transmission or storage for a number of reasons (for example noise or interference, scratches on a CD, etc). The Reed-Solomon decoder processes each block and attempts to correct errors and recover the original data. The number and type of errors that can be corrected depends on the characteristics of the Reed-Solomon code.

Matlab code:

```
clc;
clear all;
close all;
m = 3;          % Number of bits per symbol
n = 2^m - 1;    % Codeword length
k = 3;          % Message length
disp('The generator polynomial is');
genpoly = rsgenpoly(n,k)
disp('The message is');
msg = gf([ 7 3 6],m)
disp('Code Vector');
code = rsenc(msg,n,k,genpoly)
disp('Received vector');
errors = gf([0 7 0 0 0 0 0],m);
noisycode = code + errors
disp('Decoded vector');
decoded = rsdec(noisycode,n,k,genpoly)
```

Results:

```
The generator polynomial is
genpoly = GF(2^3) array. Primitive polynomial = D^3+D+1 (11 decimal)
Array elements =
           1           3           1           2           3
The message is
msg = GF(2^3) array. Primitive polynomial = D^3+D+1 (11 decimal)
Array elements =
           7           3           6
Code Vector
code = GF(2^3) array. Primitive polynomial = D^3+D+1 (11 decimal)
Array elements =
           7           3           6           2           2           7           6
Received vector
noisycode = GF(2^3) array. Primitive polynomial = D^3+D+1 (11 decimal)
Array elements =
           7           4           6           2           2           7           6
Decoded vector
decoded = GF(2^3) array. Primitive polynomial = D^3+D+1 (11 decimal)
Array elements =
           7           3           6
>>
```

Ex: genpoly=[1 3 1 2 3]= $x^4 + \alpha^3 x^3 + x^2 + \alpha x + \alpha^3$

Experiment No: 12 Date: __________

Encoding and Decoding of Convolution Codes.

Aim: Encoding and decoding of convolution codes using Trellis

Tool Required: Mat lab R2015a

Theory: A convolutional encoder processes the information sequence continuously. The n-bit encoder output at a particular time depends not only on the k-bit information sequence, but also on m previous input blocks i.e., a convolutional encoder has a memory order of m. The set of sequences produced by a k-input, n-output encoder of memory order m is called an (n, k, m) convolutional code. The values of n and k are much smaller for convolutional codes compared to the block codes.

Matlab code:

```
clc;
clear all;
close all;
disp('Enter the input data');
data = [1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0];
%data = randi([0 1],10,1);
disp(data);
trellis= poly2trellis(3,[7 5]);
%The poly2trellis function accepts a polynomial description of a
%convolutional encoder and returns the corresponding trellis structure description.
% It is a (2, 1, 2) Convolution encoder. ‘3’ indicates the number bits in the connection vector. ‘7’
% and ‘5’ are connection vectors.
disp('The convolutional encoder output is');
code = convenc(data,trellis);
disp(code);
tblen = 14;
decoded_data=vitdec(code,trellis,tblen,'trunc','hard');
disp('The Decoded data is');
disp(decoded_data);
biterr=biterr(data,decoded_data);
```

Results

```
Enter the input data
  Columns 1 through 15

    1     1     0     0     1     1     0     0     1     1     0     0     1     1     0

  Columns 16 through 30

    0     1     1     0     0     1     1     0     0     1     1     0     0     1     1

  Columns 31 through 32

    0     0



The convolutional encoder output is
  Columns 1 through 14

    1     1     0     1     0     1     1     1     1     1     0     1     0     1

  Columns 15 through 28

    1     1     1     1     0     1     0     1     1     1     1     1     0     1

  Columns 29 through 42

    0     1     1     1     1     1     0     1     0     1     1     1     1     1

  Columns 43 through 56

    0     1     0     1     1     1     1     1     0     1     0     1     1     1

  Columns 57 through 64

    1     1     0     1     0     1     1     1
```

```
The Decoded data is
  Columns 1 through 14

     1     1     0     0     1     1     0     0     1     1     0     0     1     1

  Columns 15 through 28

     0     0     1     1     0     0     1     1     0     0     1     1     0     0

  Columns 29 through 32

     1     1     0     0

>> biterr

biterr =

     0
```

PROGRAM OUTCOMES (POs)

1. **Engineering knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2. **Problem analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

3. **Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

4. **Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5. **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

6. **The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

7. **Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

8. **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. **Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.