American International University- Bangladesh

Department of Computer Engineering

COE 3201: Data Communication Laboratory

Title: Study of Digital to Analog Conversion using MATLAB

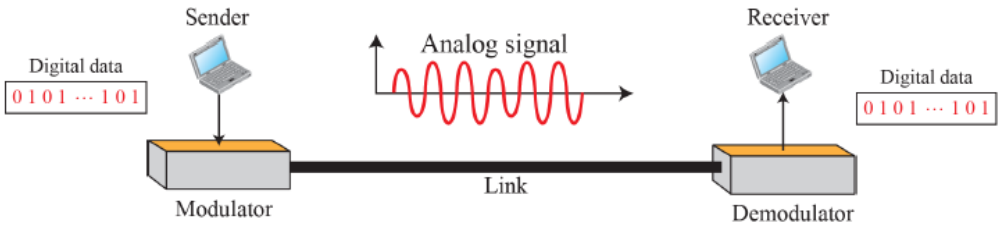
Abstract:

This experiment is designed to-

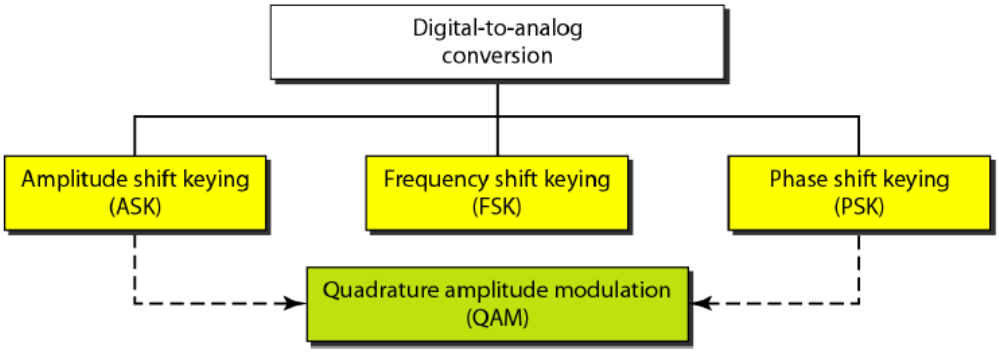
1.To understand the use of MATLAB for solving communication engineering problems.
2.To develop understanding of Digital to Analog conversion using MATLAB.

Introduction:

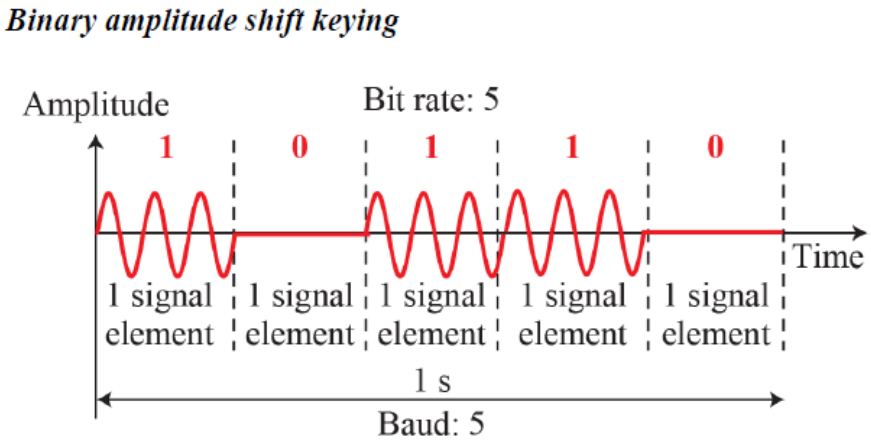
I. DIGITAL TO ANALOG CONVERSION: Digital to analog conversion is the process of changing one of the characteristics of an analog signal based on the information in digital data Figure bellow shows the relationship between the digital information, the digital to analog modulating process, and the resultant analog signal.



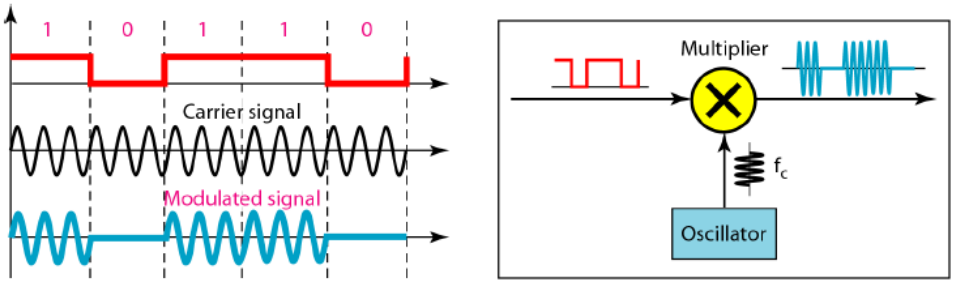
II. TYPES OF DIGITAL TO ANALOG CONVERSION:



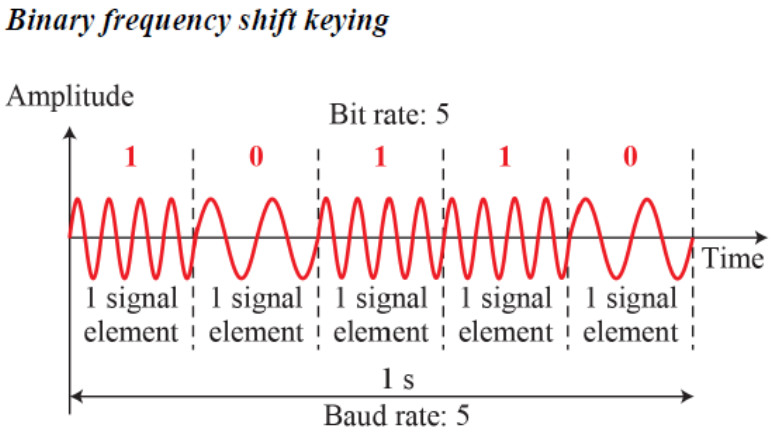
ASK: In amplitude shift keying, the amplitude of the carrier signal is varied to create signal elements Both frequency and phase remain constant while the amplitude changes.



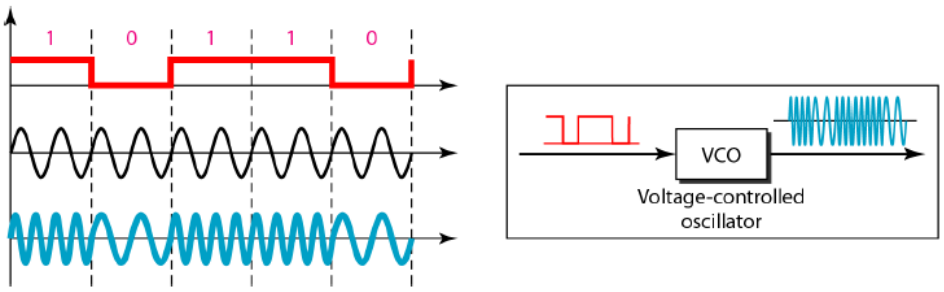
Implementation of binary ASK:



FSK: In frequency shift keying, the frequency of the carrier signal is varied to represent data The frequency of the modulated signal is constant for the duration of one signal element, but changes for the next signal element if the data element changes Both peak amplitude and phase remain constant for all signal elements

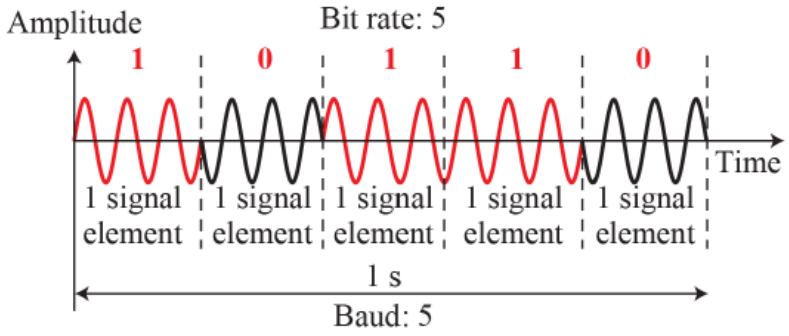


Implementation of BFSK:

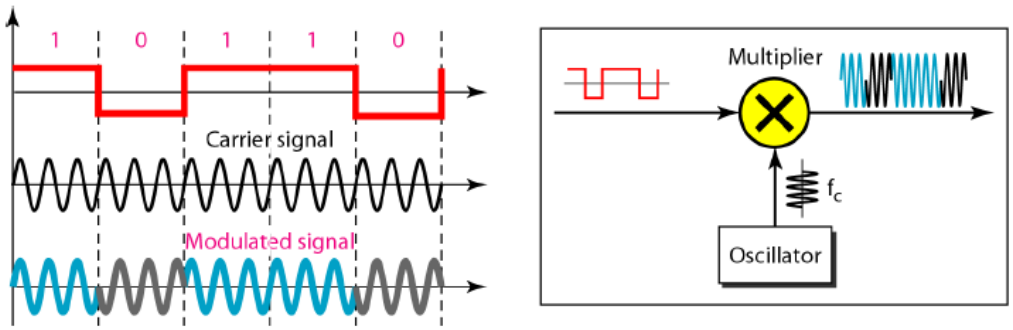


PSK: In phase shift keying, the phase of the carrier is varied to represent two or more different signal elements Both peak amplitude and frequency remain constant as the phase changes Today, PSK is more common than ASK or FSK. However, we will see shortly that QAM, which combines ASK and PSK, is the dominant method of digital to analog modulation

Binary phase shift keying



Implementation of BPSK



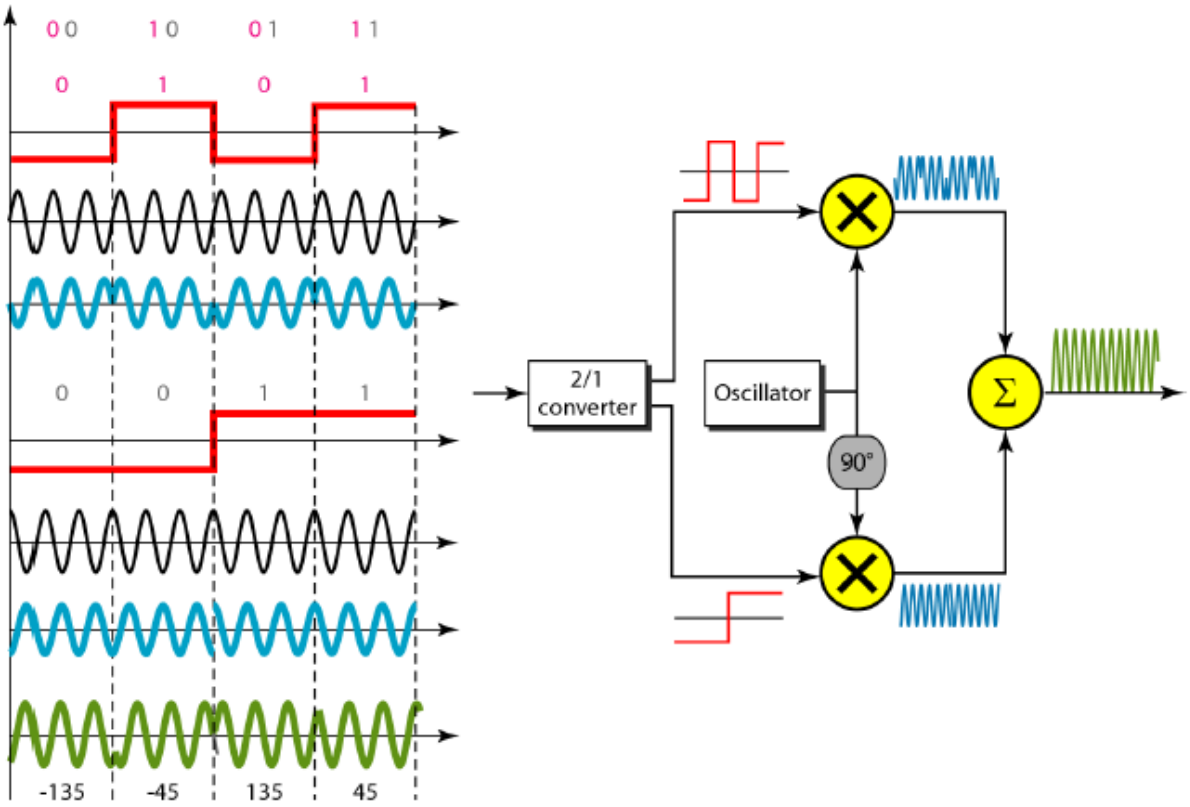
MATLAB code for digital to analog modulation (ask, fsk and psk):

```
close all;
clc;
f=5;
f2=10;
x=[1 1 0 0 1 0 1 0] % input signal ;
nx=size(x,2);

i=1;
while i<nx+1
     t = i:0.001:i+1;
    if x(i)==1
       ask=sin(2*pi*f*t);
       fsk=sin(2*pi*f*t);
       psk=sin(2*pi*f*t);
    else
        ask=0;
        fsk=sin(2*pi*f2*t);
        psk=sin(2*pi*f*t+pi);
    end
    subplot(3,1,1);
    plot(t,ask);
    hold on;
    grid on;
    axis([1 10 -1 1]);
 title('Amplitude Shift Key')
    subplot(3,1,2);
    plot(t,fsk);
    hold on;
    grid on;
    axis([1 10 -1 1]);
title('Frequency Shift Key')

    subplot(3,1,3);
    plot(t,psk);
    hold on;
    grid on;
    axis([1 10 -1 1]);
 title('Phase Shift Key')

    i=i+1;
end
```

QPSK: The simplicity of BPSK enticed designers to use 2 bits at a time in each signal element, thereby decreasing the baud rate and eventually the required bandwidth. The scheme is called quadrature PSK or QPSK because it uses two separate BPSK modulations; one is in-phase, the other quadrature (out-of-phase).

The incoming bits are first passed through a serial-to-parallel conversion that sends one bit to one modulator and the next bit to the other modulator. If the duration of each bit in the incoming signal is T, the duration of each bit sent to the corresponding BPSK signal is 2T. This means that the bit to each BPSK signal has one-half the frequency of the original signal. Figure bellow shows the idea.

The two composite signals created by each multiplier are sine waves with the same frequency, but different phases. When they are added, the result is another sine wave, with one of four possible phases: 45°, −45°, 135°, and −135°. There are four kinds of signal elements in the output signal (L = 4), so we can send 2 bits per signal element (r = 2).



MATLAB code for QPSK

```
close all;
clc;
f=10;
x=[00 10 01 11] % input signal ;
x1=[0 1 0 1];
x2=[0 0 1 1];
nx=size(x1,2);

i=1;
while i<nx+1
    t = i:0.001:i+1;
    if x1(i)==1
       psk1=sin(2*pi*f*t);
    else
       psk1=sin(2*pi*f*t+pi);
    end

    if x2(i)==1
       psk2=sin(2*pi*f*t+pi/2);
    else
       psk2=sin(2*pi*f*t+pi+pi/2);
    end

    QPSK = psk1+psk2;

    subplot(3,1,1);
    plot(t,psk1);
    hold on;
    grid on;
    axis([1 4 -1 1]);
    title('PSK1')

    subplot(3,1,2);
    plot(t,psk2);
    hold on;
    grid on;
```

```
    axis([1 4 -1 1]);
    title('PSK2')

    subplot(3,1,3);
    plot(t,QPSK);
    hold on;
    grid on;
    axis([1 4 -2 2]);
    title('QPSK')

    i=i+1;
end
```