Communication Lab II

(ELC3940)

Experiment No.: 04

Object:

Generate a 16-QAM signal (assume carrier frequency of 500 kHz) using a rectangular constellation. Plot the QAM signal corresponding to the first 20 bits obtained from Exp. 2.

G. No: GL3136

S. No: A3EL-02

F. No: 19ELB056

Name: Maha Zakir Khan

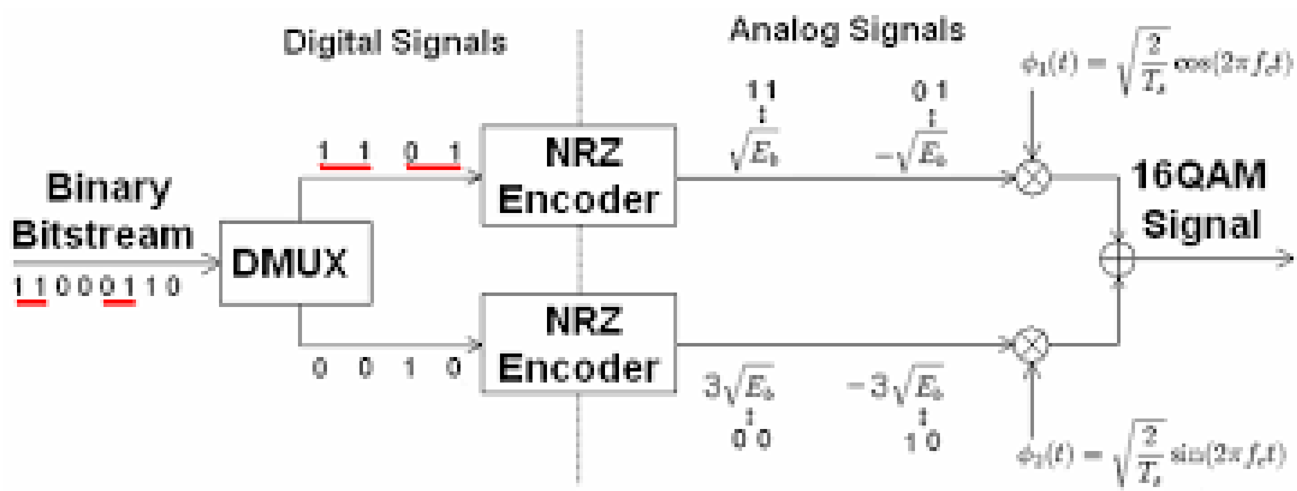
Date of performing experiment: 15|02|2022

Date of report submission: 22|02|2022

Software Used:

MATLAB®, Release 2021a (R2021a), a programming platform designed specifically to analyze and design systems and products. The heart of MATLAB is the MATLAB language, a matrix-based language, it provides vast library of mathematical functions for linear algebra, statistics, Fourier analysis, filtering, optimization, numerical integration and solving ordinary differential equations.

Procedure:



(a). Line coding

Take 20 sample value of speech signal encoded binary bitstream with bit-rate Rb, and perform polar-NRZ signaling to generate bit sequence X[n], where n = 1,2,3…20

(b). DMUX

Demultiplex the bit sequence X[n] to two channels with each 4-bit sequence:

$$(X[k] \ to \ X[k+3], \ k = 1,5,10,15 \times Tb \ )$$

$$Tb \ = \frac{1}{Rb} = Time \, period \, of \, a \, pulse$$

distributed to in-phase channel as: $X_i\,(X[k]\;X[k+1])$

and to quadrature-phase channel as: $X_q\,(X[k+2]\;X[k+3])$

Where $i = q = 1,2,3,4,5 \times Ts$

Ts is the symbol time period

(c). Symbol generation

Symbols representing each 2-bit combination by a voltage level is generated. Using gray-scale mapping 4 levels $[3 \ 1 \ -1 \ -3] \times \sqrt{Eb}$ are generated according to the bit combination. Using this scheme, in-phase and quadrature-phase symbols S_i and S_q are produced. Eb is average energy per bit

(d). Carrier multiplication

The generated in-phase and quadrature-phase symbols S_i and S_q are multiplied with orthogonal basis vectors ϕ_i and ϕ_q where:

$\phi_i = Ac\cos(2\pi fct)$ fc is carrier frequency, Ts is the symbol time period

$\phi_q = Ac\sin(2\pi fct)$ $Ac = \sqrt{\left(\frac{1}{Ts}\right)}$

This leads to 4-ASK modulated signal:

$$S\phi_i = Ac\cos(2\pi fct) \times S_i$$

$$S\phi_q = Ac\sin(2\pi fct) \times Sq$$

(e). Addition

The two channel ASK modulated signals are finally added to produce the 16-Level QAM signal

$$QAM\,[i] = S\phi_i + S\phi_q$$

Where $i = q = \{1,2,3,4,5\} \times Ts$

Program:

Using .wav file 'Hello5' to read the speech signal in MATLAB and performed PCM encoding using the function PcmEncoder from previous *Experiment-02*

Note: Used a random generated binary bit sequence as an alternative to the speech signal values

```
info = audioinfo('C:\Users\Maha Khan\Downloads\Hello5.wav');
[signal , f] = audioread('C:\Users\Maha Khan\Downloads\Hello5.wav'); % Voice signal with
                                                                    % dual channel
                                       % and fs as the sampling frequency
[X ,Rb] = PcmEncoder(signal,info);     % Calling Function to psm encode
                                       % the signal
Tb = 1/Rb;  % Tb - Time period of pulse , Rb - Bit-rate

% X = randi([0 1],1,20); % Using a random generated binary sequence for better plot results

A = 1;      % Voltage level
Ns = 20;    % Number of samples
nseq = 0:0.1:Ns-0.1;
```

PcmEncoder.m : Function that samples, quantizes and encodes input signal and returns encoded signal with its bit-rate

<u>Input arguments:</u> Signal = input signal, info = audio info of the signal .wav file as a struct;

<u>Output arguments:</u> Encoded = encoded signal, Rb = bit-rate;

```
function [Encoded,Rb] = PcmEncoder(Signal,info)

dur = info.Duration;          % Signal duration in seconds
Ts1 = info.TotalSamples;      % Total no. of samples in time-domain
n1 = linspace(0,dur,Ts1);
y1 = Signal(:,1);                % Channel 1 of the signal sig

% Sampling Signal Waveform
fs = 16000;                   % Sampling frequency given in the Experiment
k = 4;                        % no.bit for 16 bit quantizer
l = 2^k;                       % 16 level
n = 0:1/fs:dur;
y = interp1(n1,y1,n);         % Signal Waveform with sampling frequency fs
Ns = length(y);

 ymax = max(y);
 ymin = min(y);


%Quantizing Signal Waveform
partition = linspace(ymin,ymax,l-1);
codebook = linspace(ymin,ymax,l);

[index,quants] = quantiz(y,partition,codebook); %quantizer

coder = [0 0 0 0;        % 4-bit Coder
         0 0 0 1;
         0 0 1 1;
         0 0 1 0;
         0 1 1 0;
         0 1 1 1;
         0 1 0 1;
         0 1 0 0;
```

```
         1 1 0 0;
         1 1 0 1;
         1 1 1 1;
         1 1 1 0;
         1 0 1 0;
         1 0 1 1;
         1 0 0 1;
         1 0 0 0];

pcm = linspace(0,0,Ns*k); % Array to store pcm signal
i = 1;
c = 1;

% Encoding Quantized Signal
     while i <= length(pcm)
         pcm(i:i+3) = coder(index(c)+1,:);
         i = i+4;
         c = c+1;
     end

   Encoded = pcm;
   Rb = k*fs;
end
```

(a). Line coding

```
%Using Polar NRZ signalling to generate our bit sequence
polar = [];
c = 1;
for i = 1:Ns % Encoding signal to polar NRZ format and upsampling the signal by 10
    if X(i) == 0
    polar(c:c+9)= ones(1,10)*(-A);
    else
        polar(c:c+9) =ones(1,10)*(A);
    end
    c = c+10;
end
```

(b). DMUX

```
% De-Multiplexing sequence
for j = 1:2*fseq:length(nseq)/2
    i(j:j+fseq*2-1) = sequence(c:c+fseq*2-1);        % To store b0 and b1 in In-phase
    q(j:j+fseq*2-1) = sequence(c+fseq*2:c+fseq*4-1);% To store b2 and b3 in Quadrature-phase
    c = c+fseq+4;
end
```

(c). Symbol generation

Symgen.m: Function to generate 4-level symbols based on gray-scale mapping of input bit sequence.

Input arguments: i = input upsampled by 10 sequence, fseq = frequency of upsampled input sequence, A = unit $\sqrt{Eb}$ level;

Output arguments: sym = generated 4 level symbol, ns = no. of symbols generated

```matlab
function [sym, ns] = symGen(i,fs,A)
%UNTITLED2 Summary of this function goes here
%   Detailed explanation goes here
c = 1;
s = [];
%samples = ones(fs);
for j = 1:2*fs:length(i)
    if i(j) == -A && i(j+fs+1) == -A
            s(c)= 3;
    elseif i(j) == -A && i(j+fs+1) == A
            s(c)= 1;
    elseif i(j) == A && i(j+fs+1) == A
            s(c)= -1;
    elseif i(j) == A && i(j+fs+1) == -A
            s(c)= -3;
    end
    c = c+1;
end
sym = s;
ns = length(s);
end
```

```matlab
% Symbol generation
[si, ns] = symGen(i,fseq,A);    % si contains In-Phase symbol generated by function symGen
[sq, ns] = symGen(q,fseq,A);    % whereas sq contains Quadrature-Phase symbol

sk = complex(si,sq);  % Complex symbol containg in-phase and quadrature-phase symbol of a bit sequence

% Up-sampling sk symbol sequence to 10 times for better plot tracing
resk = [];
c = 1;
for i = 1:ns
    resk(c:c+9)= ones(1,10)*sk(i);
    c = c+10;
end
```

(d). Carrier multiplication

```
% Carrier Multiplication / 4-ASK modulation
Ts = Tb*4;        % Time period of the generated symbol is 4 times the time period of the original bit sequence X
fs = Rb/4;        % Sampling frequency of the symbol is 1/4 times the bit-rate
fc = fs*5;        % Taking carrier frequency as 5 times the sampling frequency of original bit sequence X
fcint = 1/(16*fc);
t = 0:fcint:ns*10/fc-fcint;

phi1 = cos(2*pi*fc*t); % In-phase carrier
phi2 = sin(2*pi*fc*t); % Quadrature-phase carrier


c = 1;
ps1 = [];  % To store In-phase ASK-modulated waveform
ps2 = [];  % To store Quadrature-phase ASK-modulated waveform
nt = 2*length(t)/10;
for j = 1:nt:length(t)
    ps1(j:j+nt-1)=phi1(j:j+nt-1)*si(c);
    ps2(j:j+nt-1)=phi2(j:j+nt-1)*sq(c);
    c = c+1;
end
```

(e). Addition

```
% Addition
qam = ps1+ps2;    % QAM-Modulated waveform genrated by adding in-phase and
                  % quadrature-phase ASK-modulated waveforms
```

Signal figures and plots:

```
% Figures and plots showing various steps of the modulation scheme
figure('NumberTitle', 'off', 'Name', 'ASK-Modulated In-phase Component');
subplot(3,1,1);
plot(t*20/t(end),phi1,'LineWidth',1);
xlabel('Time t (in Ts) \rightarrow');
ylabel('Carrier Amplitude Ac');
title('Carrier Signal \phi_{i}(t)= A_{c}cos(2\pifct)')
xticks([4 8 12 16 20]);
xticklabels({'\bf1Ts','\bf2Ts','\bf3Ts','\bf4Ts','\bf5Ts'});
axis([0 20 -1.5 1.5]);

subplot(3,1,2);
stairs(0:ns*10-1,real(resk),'LineWidth',2);
axis([0 ns*10-1 -4 4]);
xlabel('Discrete Time n (in Ts) \rightarrow');
ylabel('In-phase Amplitude I_{i} \rightarrow');
title('In-phase component I[n]');
yline(0,'--');
yticks([-3 -1 1 3]);
xticks([15 25 35 45 54]);
xticklabels({'\bf1Ts','\bf2Ts','\bf3Ts','\bf4Ts','\bf5Ts'});

subplot(3,1,3);
plot(t*20/t(end),ps1,'LineWidth',1);
xlabel('Time t (in Ts) \rightarrow');
ylabel('Amplitude Ac \times I_{i} \rightarrow')
title('Modulated signal s_{i}(t) = I[n]\times\phi_{i}(t)')
yticks([-3 -1 1 3]);
xticks([4 8 12 16 20]);
xticklabels({'\bf1Ts','\bf2Ts','\bf3Ts','\bf4Ts','\bf5Ts'});
```

```
txt = ['Time Period Ts = Tb\times4: ' num2str(Ts) ' seconds '];
text(17,-6,txt,'FontSize',10,'FontWeight','bold','Color','r')
txt1 = ['Carrier Frequency fc: ' num2str(fc) ' hertz '];
text(17,-7,txt1,'FontSize',10,'FontWeight','bold','Color','r')
sgtitle('4-level ASK-Modulation (for In-phase)')


figure('NumberTitle', 'off', 'Name', 'ASK-Modulated Quadrature-phase Component');
subplot(3,1,1);
plot(t*20/t(end),phi2,'LineWidth',1,'Color','r');
xlabel('Time t (in Ts) \rightarrow');
ylabel('Carrier Amplitude Ac');
title('Carrier Signal \phi_{q}(t)= A_{c}sin(2\pifct)')
xticks([4 8 12 16 20]);
xticklabels({'\bf1Ts','\bf2Ts','\bf3Ts','\bf4Ts','\bf5Ts'});
axis([0 20 -1.5 1.5]);

subplot(3,1,2);
stairs(0:ns*10-1,imag(resk),'LineWidth',2,'Color','r');
axis([0 ns*10-1 -4 4]);
xlabel('Discrete Time n (in Ts) \rightarrow');
ylabel('Quadrature-phase Amplitude Q_{i} \rightarrow');
title('Quadrature-phase component Q[n]');
yline(0,'--');
yticks([-3 -1 1 3]);
xticks([15 25 35 45 54]);
xticklabels({'\bf1Ts','\bf2Ts','\bf3Ts','\bf4Ts','\bf5Ts'});
```

```
subplot(3,1,3);
plot(t*20/t(end),ps2,'LineWidth',1,'Color','r');
xlabel('Time t (in Ts) \rightarrow');
ylabel('Amplitude Ac \times I_{i} \rightarrow')
title('Modulated signal s_{q}(t) = Q[n]\times\phi_{q}(t)')
yticks([-3 -1 1 3]);
xticks([4 8 12 16 20]);
xticklabels({'\bf1Ts','\bf2Ts','\bf3Ts','\bf4Ts','\bf5Ts'});

txt = ['Time Period Ts = Tb\times4: ' num2str(Ts) ' seconds '];
text(17,-6,txt,'FontSize',10,'FontWeight','bold','Color','r')
txt1 = ['Carrier Frequency fc: ' num2str(fc) ' hertz '];
text(17,-7,txt1,'FontSize',10,'FontWeight','bold','Color','r')
sgtitle('4-level ASK Modulation (for Quadrature-phase)')
```

```
figure('NumberTitle', 'off', 'Name', '16-Level QAM Modulated Bit-sequence ');

subplot(4,1,1);
stem(nseq,sequence,'Marker','.','Color','r');
hold on
stairs(nseq,sequence,'LineWidth',2,'Color','#0072BD');
yline(0,':');
axis([0 20 -1.5 1.5]);
title('NRZ-Polar Data Bit-Sequence');
xlabel('Discrete time n (in Tb) \rightarrow');
ylabel('Amplitude in volts');
xticks([1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20]);
xticklabels({'\bf1Tb','\bf2Tb','\bf3Tb','\bf4Tb','\bf5Tb',...
    '\bf6Tb','\bf7Tb','\bf8Tb','\bf9Tb','\bf10Tb',...
    '\bf11Tb','\bf12Tb','\bf13Tb','\bf14Tb','\bf15Tb',...
    '\bf16Tb','\bf17Tb','\bf18Tb','\bf19Tb','\bf20Tb'});
txt = ['Time Period Tb: ' num2str(Tb) ' seconds '];
text(17,-2.5,txt,'FontSize',10,'FontWeight','bold','Color','r')
txt = ['Bit-rate Rb: ' num2str(Rb) ' bits/second'];
text(17,-3.0,txt,'FontSize',10,'FontWeight','bold','Color','r')
```

```
subplot(3,1,2);
stairs(0:ns*10-1,real(resk),'LineWidth',2);
hold on
stairs(0:ns*10-1,imag(resk),'LineWidth',1,'Color','r');
hold off
axis([0 ns*10-1 -5 5]);
xlabel('Discrete Time n (in Ts) \rightarrow');
ylabel('Symbol Amplitude S_{i} \rightarrow');
legend('In-phase component I','Quadrature-phase component Q');
title('16-QAM Symbol S[n] = I[n] + Q[n]\iota')
h3 = yline(0,'--');
set( get( get( h3, 'Annotation'), 'LegendInformation' ), 'IconDisplayStyle', 'off' );
xticks([15 25 35 45 54]);
xticklabels({'\bf1Ts','\bf2Ts','\bf3Ts','\bf4Ts','\bf5Ts'});


subplot(3,1,3);
plot(t*20/t(end),qam,'LineWidth',1);

axis([0 20 -5 5]);
xlabel('Time t  (in Ts) \rightarrow');
ylabel('Signal Amplitude s(t) \rightarrow');
title('Modulated Signal s(t) = s_{i}(t) + s_{q}(t)')
h3 = yline(0,'--');
set( get( get( h3, 'Annotation'), 'LegendInformation' ), 'IconDisplayStyle', 'off' );
xticks([4 8 12 16 20]);
xticklabels({'\bf1Ts','\bf2Ts','\bf3Ts','\bf4Ts','\bf5Ts'});
```

```matlab
txt = ['Symbol Duration Ts = Tb\times4: ' num2str(Ts) ' seconds '];
text(17,-7,txt,'FontSize',10,'FontWeight','bold','Color','r')
txt1 = ['Carrier Frequency fc: ' num2str(fc) ' hertz '];
text(17,-8,txt1,'FontSize',10,'FontWeight','bold','Color','r')
sgtitle('16-QAM (Quadrature Amplitude Modulated) Signal')



% Scatter plot for plottting Constellation diagram of 16 level QAM
M = 16;
x = 0:M-1;
symgray = qammod(x,M,'gray'); % 16-QAM output (Gray-coded)
scatterplot(symgray,1,0,'b*');
for k = 1:M
    text(real(symgray(k)) - 0.0,imag(symgray(k)) + 0.3,...
        dec2base(x(k),2,4),'Color',[1 0 0]);
     text(real(symgray(k)) - 0.5,imag(symgray(k)) + 0.3,...
         num2str(x(k)),'Color',[1 0 0]);
end
title('16-QAM Symbol Mapping')
axis([-4 4 -4 4])
```

Result:

(1) Random Binary Bit-sequence:

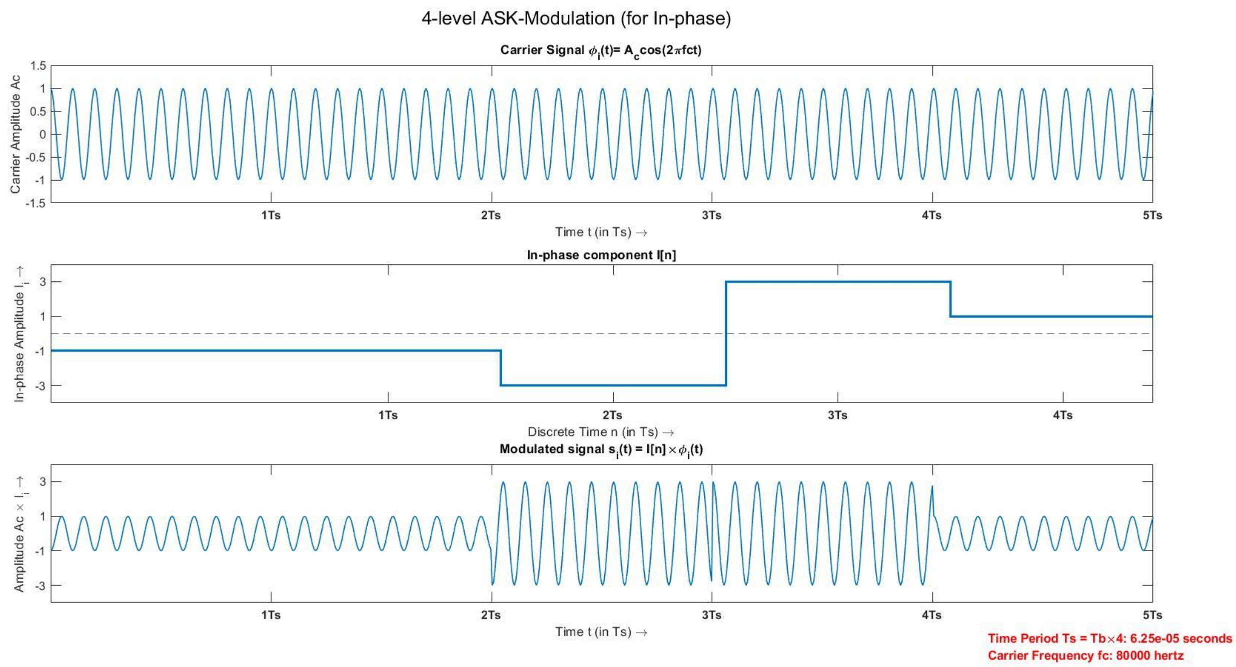


Fig 1.a

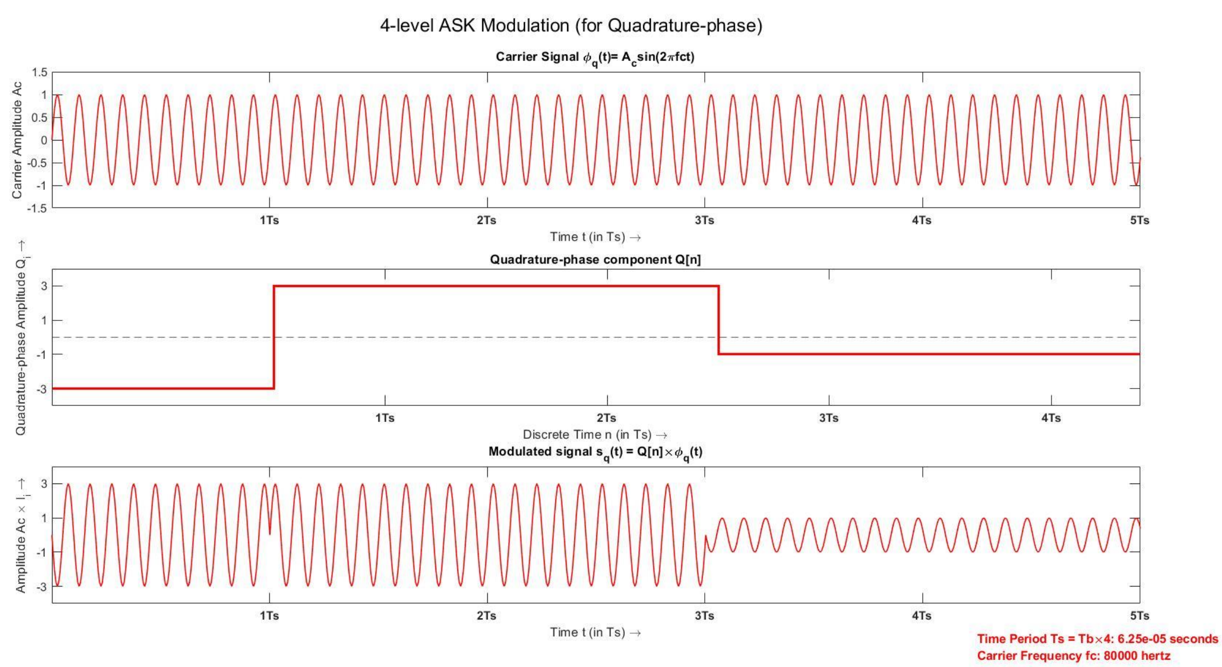
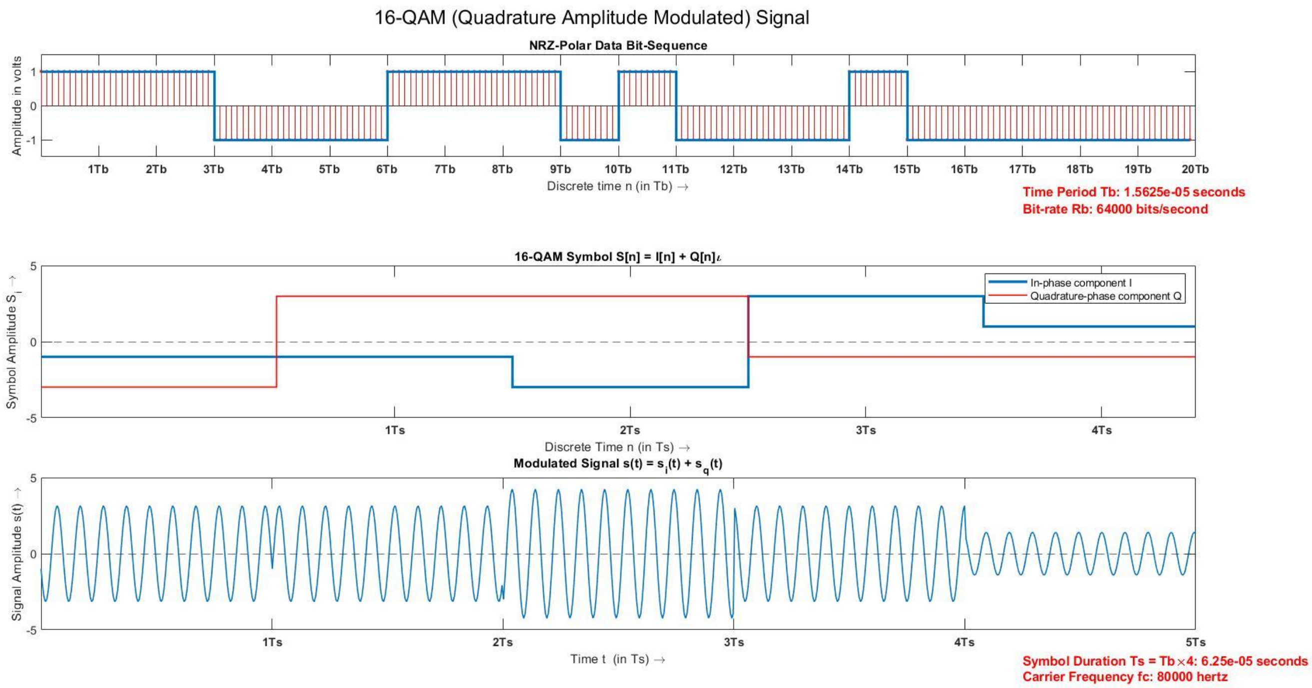


Fig 1.b



16-QAM (Quadrature Amplitude Modulated) Signal
NRZ-Polar Data Bit-Sequence
Amplitude in volts
Discrete time n (in Tb) →
Time Period Tb: 1.5625e-05 seconds
Bit-rate Rb: 64000 bits/second
16-QAM Symbol S[n] = I[n] + Q[n]ι
In-phase component I
Quadrature-phase component Q
Symbol Amplitude S_i →
Discrete Time n (in Ts) →
Modulated Signal s(t) = s_i(t) + s_q(t)
Signal Amplitude s(t) →
Time t (in Ts) →
Symbol Duration Ts = Tb×4: 6.25e-05 seconds
Carrier Frequency fc: 80000 hertz

Fig 1.c

(2) Voice signal Bit-sequence:

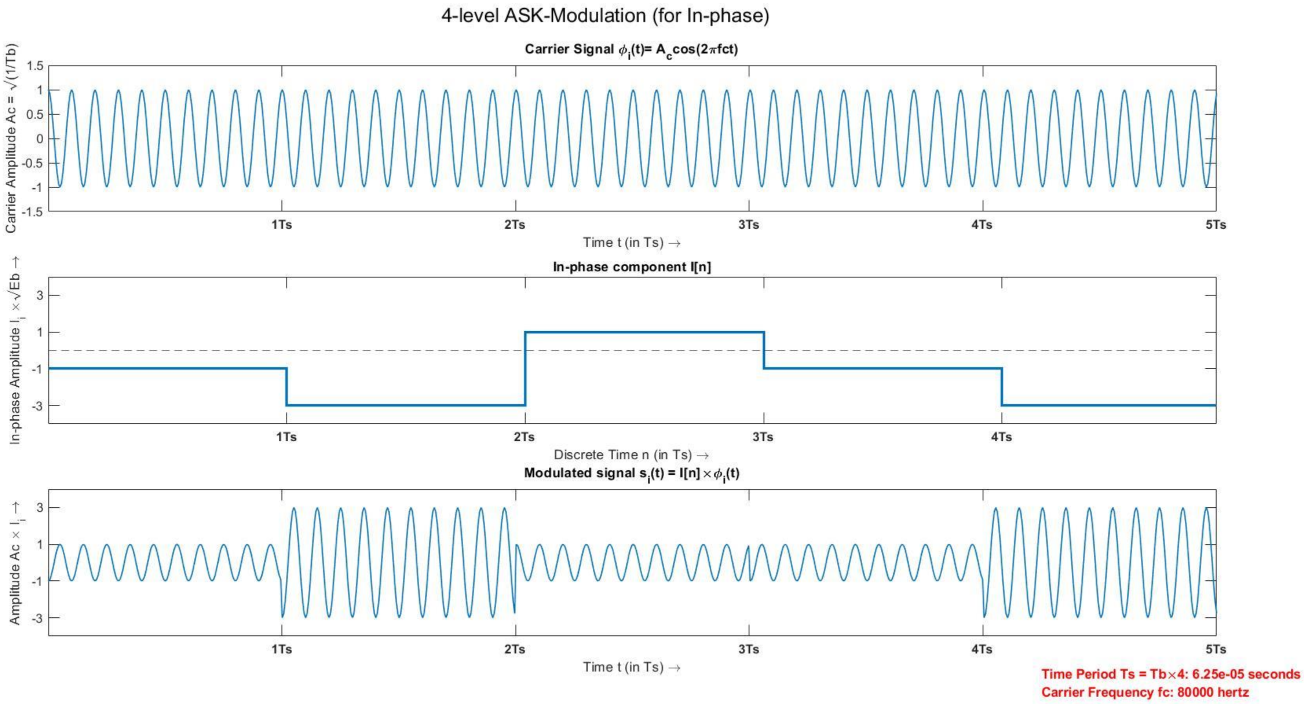


Fig 2.a

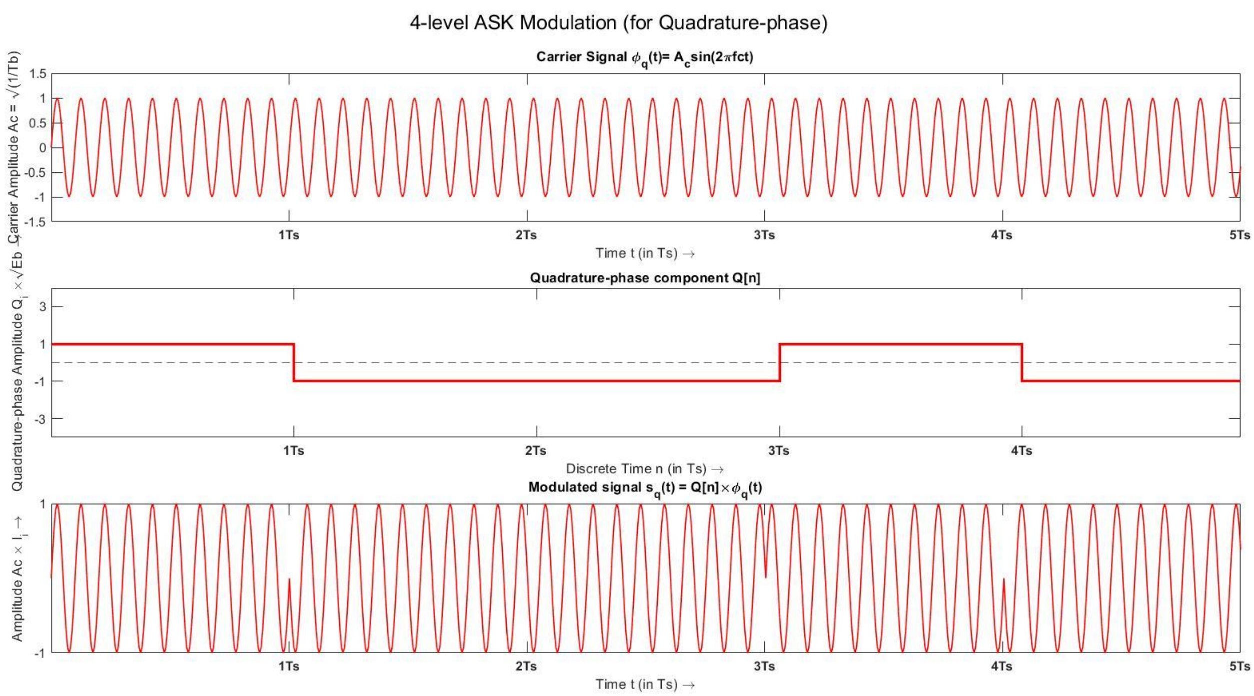


Fig 2.b

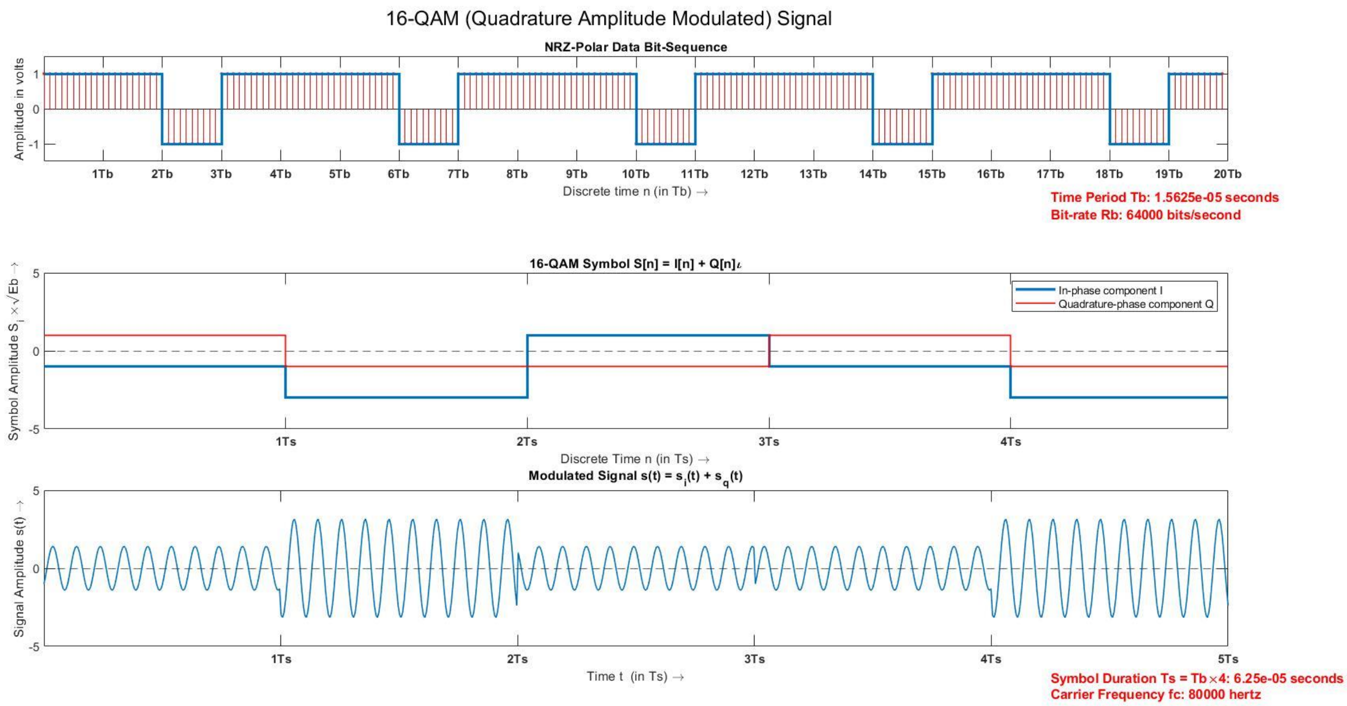
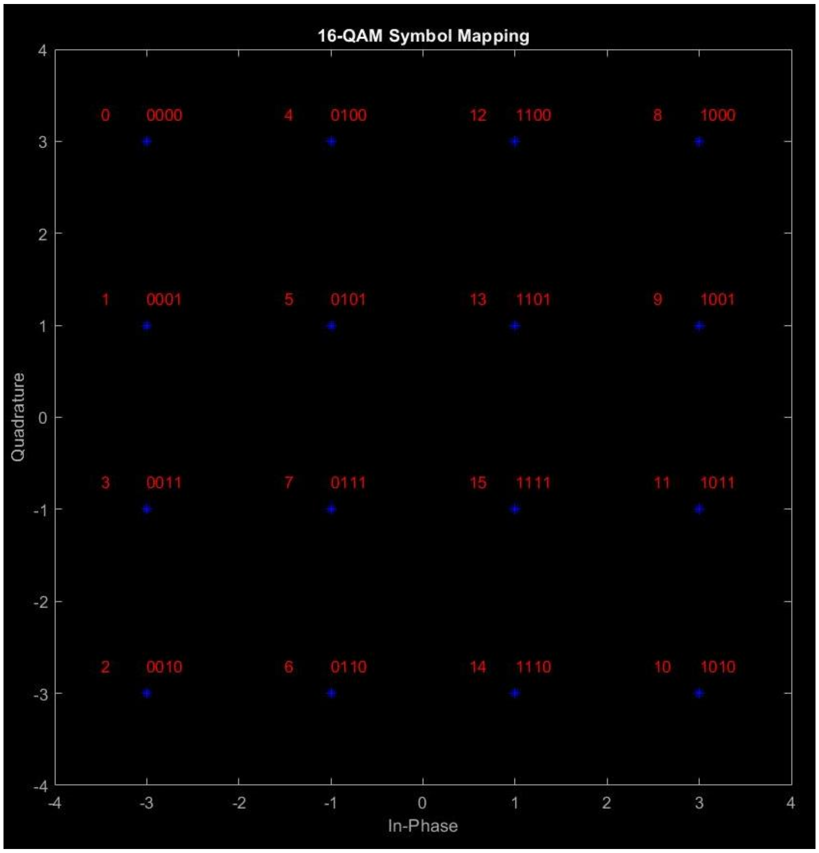

16-QAM (Quadrature Amplitude Modulated) Signal
NRZ-Polar Data Bit-Sequence
Amplitude in volts
Discrete time n (in Tb) →
Time Period Tb: 1.5625e-05 seconds
Bit-rate Rb: 64000 bits/second
16-QAM Symbol S[n] = I[n] + Q[n]ι
In-phase component I
Quadrature-phase component Q
Discrete Time n (in Ts) →
Modulated Signal s(t) = s_i(t) + s_q(t)
Signal Amplitude s(t) →
Time t (in Ts) →
Symbol Duration Ts = Tb×4: 6.25e-05 seconds
Carrier Frequency fc: 80000 hertz

Fig 2.c

(3) Constellation diagram for 16-QAM:



Discussion:

While performing this Experiment, I used a random bit sequence along with the encoded voice channel to show phase and amplitude change more clearly for different symbol values. The bit sequence has a bit-rate of 48k samples/s and I've arbitrarily chosen a frequency of 5*43khz i.e., 80khz as the frequency of my carrier signal. It is to be noted that the in-phase and quadrature-phase bit sequence component are 4-ASK modulated individually.

From Fig 2.awe can observe that when there is a change in symbol level, the amplitude of the carrier changes according, and when there is a transition from $+1\sqrt{Eb}$ to -$1\sqrt{Eb}$ at 3Ts, the phase of the carrier changes by 180 degrees, it was also observed when the transition does not cross the zero-line, there is no change in phase, but only in amplitude of the carrier as seen at 1Ts when signal transitions from -$1\sqrt{Eb}$ to -$3\sqrt{Eb}$. Also, when there is a sign as well as amplitude change in symbol, both phase and amplitude of the carrier as seen at 2Ts when signal transition from -$3\sqrt{Eb}$ to $+1\sqrt{Eb}$

The constellation diagram shows the 16-QAM Symbol mapping of the gray-coded bit sequence on the complex plane. The diagram was not plotted for the given bit sequence but for all possible 16 4-bit binary combination of values