Faculty of Engineering
Credit hours system

Communications-1
EECS306

PCM Project

Rawan Yasser Hashem	**1210008**
Youssef Ahmed Abdelbar	**1210090**

Defining the Used Parameters

```
%parameter definition
Fs = 10000;
dt = 1/Fs;
Tmin=0;                %starting time
Tmax= 2/30;      %ending time
t = Tmin:dt:Tmax; %drawing two periods of the Message
frequency = 30;        %Frequency of the message in Hertz
W = 2*pi*frequency;    %Frequency of the message in rad/sec
Amplitude = 2;         %Amplitude of the message
Number_of_Bits=input('Enter Number of bits per sample :  ');
Quantization_Levels = 2^Number_of_Bits; %number of quantization error

sampling_rate_to_nyquist=input('Sampling rate to the Nyquist rate : ');
Nyquist_Sampling_Rate=2*frequency; %nyquist rate of the message
Sampling_Frequency= sampling_rate_to_nyquist*Nyquist_Sampling_Rate;% sampling rate
```

Fig.1

Start by defining the time interval the signal will be drawn in , and defining the frequency of the signal , the amplitude , and get from the user the number of bits per sample and calculate the number of quantization levels , calculating the Nyquist sampling frequency and get from the user the sampling rate to the Nyquist rate to obtain the sampling frequency.

Generating the Message Signal

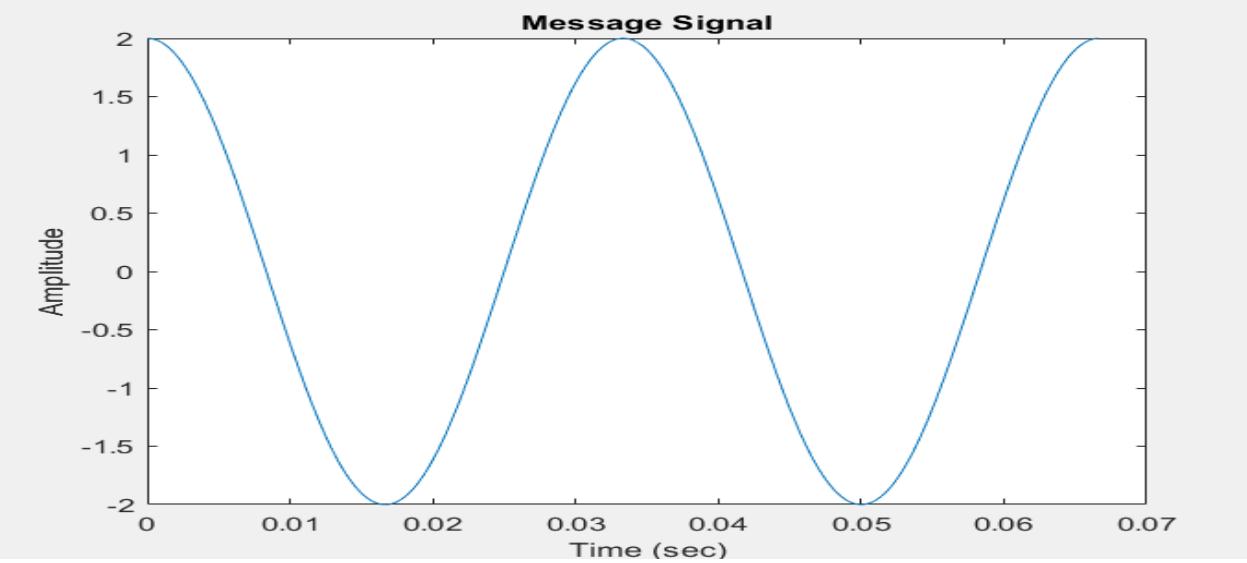


Fig.2

This analog signal, has a frequency **30 Hz** and a peak amplitude of **2 volt**, is generated by using the following code(Fig.3):

```
%Generate the analog sinusoidal signal with frequency 30 HZ and Amplitude 2
%volts and plot it
Analog_Message= Amplitude .* cos(W*t); %sinusoidal signal
% Plot the analog signal
figure();
plot(t,Analog_Message);
xlabel('Time (sec)');
ylabel('Amplitude');
title('Message Signal');
```

Fig.3

Requirement 1

(If the Sampling rate equals the Nyquist rate)

(8 level quantizer)

1- The Sampled Signal

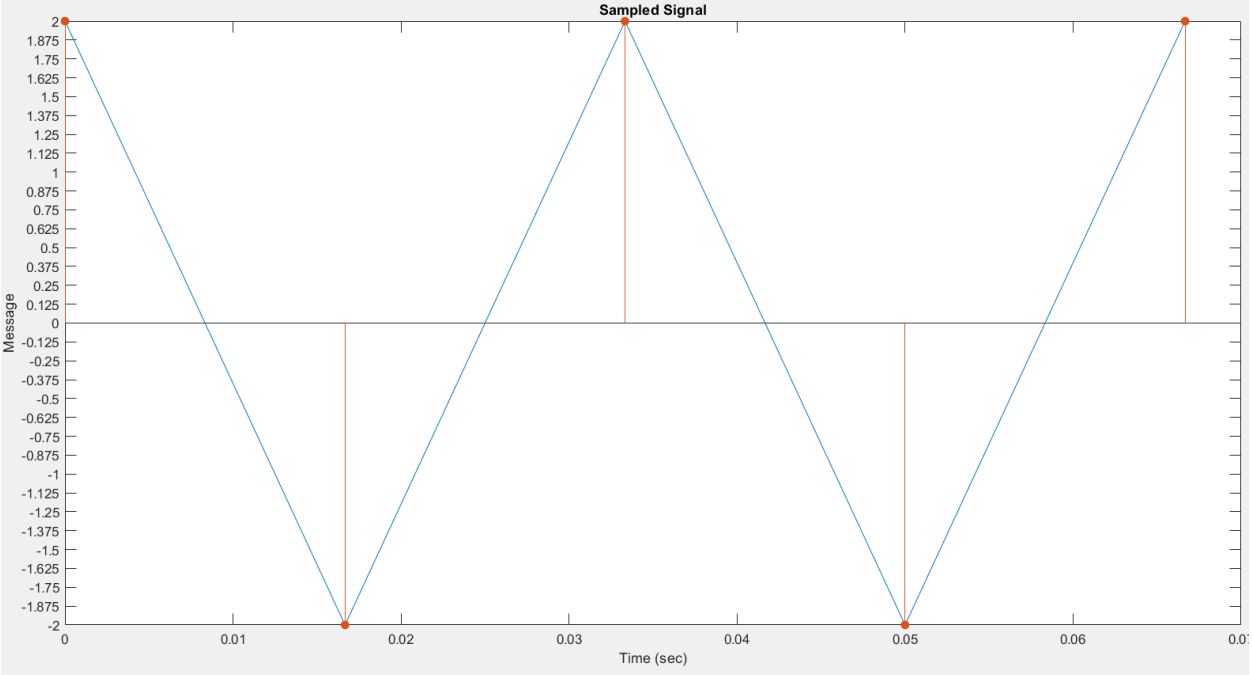


Fig.4

Here, Sampling frequency = NR = 2*Fm = 60 Hz and Ts = 1/Fs = 1/60 sec.

So, we can see (fig.4) that the samples are taken every half period

And it's generated by the following code:

```
%Sampling process
Ts=1/Sampling_Frequency; %sampling time
nsMax=floor(Tmax/Ts);     %get the range of time samples
n=0:nsMax;
t_sampling=n*Ts;
Sampled_Message= Amplitude .* cos(W*t_sampling); %the sampled signal

figure();
plot(t_sampling,Sampled_Message);
ylim([-2 2]);
xlabel('Time (sec)');
ylabel('Message');
title('Sampled Signal');
hold on
stem(t_sampling,Sampled_Message,"filled");% function to draw delta at the sampling time
set(gca,'ytick',-2:2/16:2);
hold off
```

Fig.5

Where, t_sampling = n*Ts and stem is a function that draws deltas at the sampling time

2- Quantized Signal

Here in this case, we used number of bits per sample “n” = 3 bits. So, we have number of levels “L” = 2^3 = 8 levels

Calculating the step size: Step_size = (Vmax - Vmin) / Quantization_Levels; Since, Vmax = 2 and Vmin = -2, we got step size = 0.5

Taking the levels as midrise and doing the quantization process by a function “quantiz”:

[index , Quantized_value] = quantiz (Sampled_Message , partition , codebook);

Where, “Partition” makes partitions between Vmax and Vmin with a difference of a step size:
partition = Vmin : Step_size : Vmax;
And “Codebook” expresses the quantization value for each partition :

codebook = Vmin-(Step_size/2) : Step_size : Vmax+(Step_size/2);

Then, the first output is array named “index” which determines to which level each sample belongs and second output is array named “quantized_value” which is the output of the quantizer that contains the quantization values of the input signal to each sample.

So, at the end we have levels at -1.75:0.5:1.75 , and the samples are quantized as shown in fig.6:

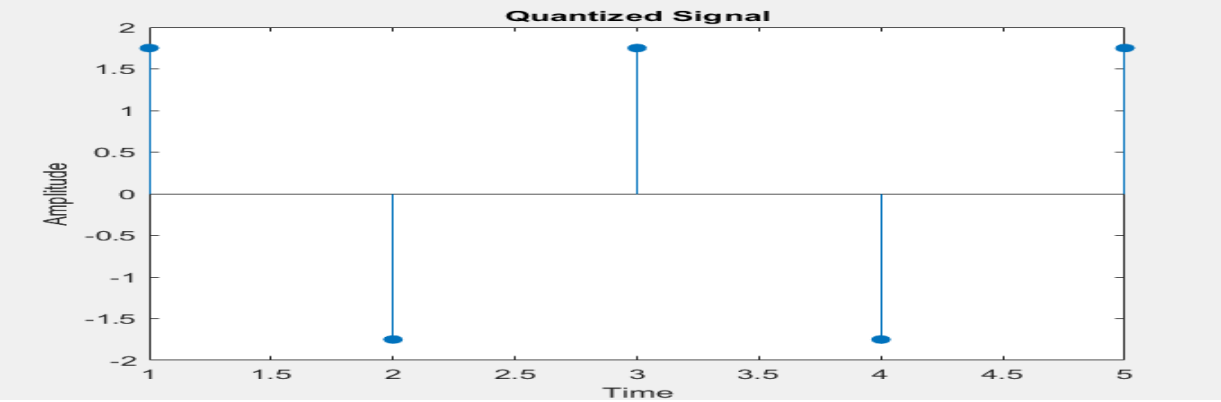


Fig.6

3- Encoded Signal

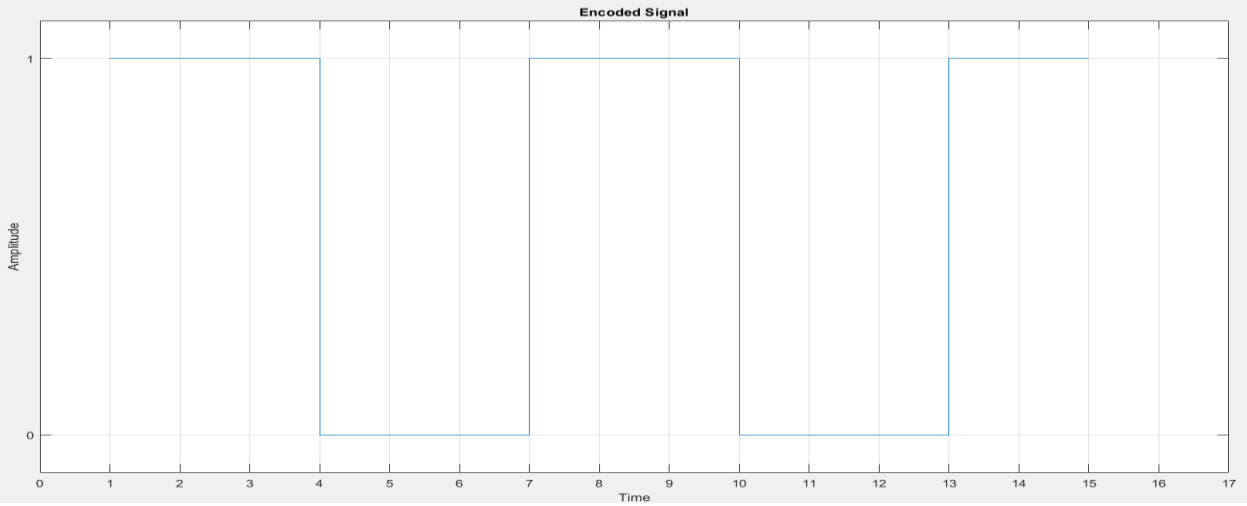


Fig.7

As it is clarified it is a stream of bits and used code is unipolar NRZ to represent this stream of bits, to get this stream of bits we used the following code , there is matrix called code and obtained using function de2bi which convert each leveled sample to binary, in this matrix the row has binary code for a sample . so, to get the stream of bits in 1-D array we did this loop and got the stream of bits in coded.

```
%encoding process
code=de2bi(index,'left-msb');      % Convert the decimal to binary of the indices
k=1;
for i=1:Length1
    for j=1:Number_of_Bits
        coded(k)=code(i,j);        % convert code matrix to a coded row vector
                                   % each row is a code for a sample
        j=j+1;
        k=k+1;
    end
    i=i+1;
end
```

Fig.8

4- Recovered Signal

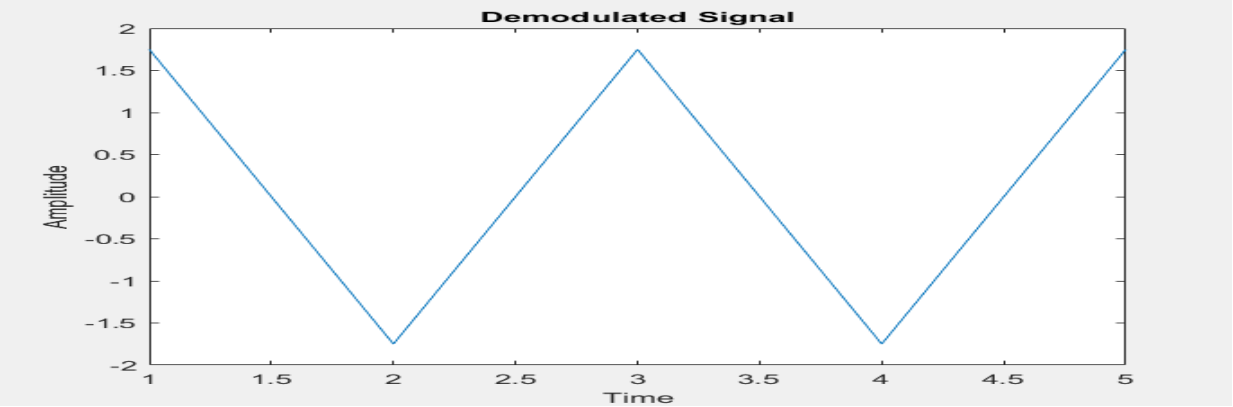


Fig.9

As we can see, the recovered signal is not the same as the sampled signal, since there is a distortion introduced by the quantization process, and the small number of samples which lead to recover a signal not like the message signal.

```
%Demodulation process
qunt=reshape(coded,Number_of_Bits,length(coded)/Number_of_Bits);
index=bi2de(qunt','left-msb');          % Getback the index in decimal form
Quantized_value=Step_size*index+Vmin+(Step_size/2); % getback Quantized values

figure();

plot(Quantized_value);                              % Plot Demodulated signal
title('Demodulated Signal');
ylabel('Amplitude');
xlabel('Time');
%the bit rate function takes the sampling frquency and number of bits
y= Bit_Rate_function(Sampling_Frequency,Number_of_Bits);
disp('The bit rate :')
disp(y);
```

Fig.10

As illustrated in fig.10 function reshape returns matrix qunt this matrix each column contains the code of a sample ,column 1 has binary code of sample 1 and column 2 has the binary code of sample 2 … ,

To get the decimal number of the quantized level to re-draw the demodulated signal is by using function bi2de and take the transpose of matrix qunt to read the first sample binary code and convert it into decimal , and get back the quantized value using Quantized_value = Step_size*index + Vmin+step_size/2.

5- Bit Rate (n*Fs)

Building a function that calculates the bit rate:

```
function result = Bit_Rate_function(Sampling_rate,Number_of_Bits)
                result = Sampling_rate*Number_of_Bits;
end
```

We got Rb = 3*60 = 180 bps

Requirement 2

(If the Sampling rate is twice the Nyquist rate)

(8 level quantizer)

1- The Sampled Signal

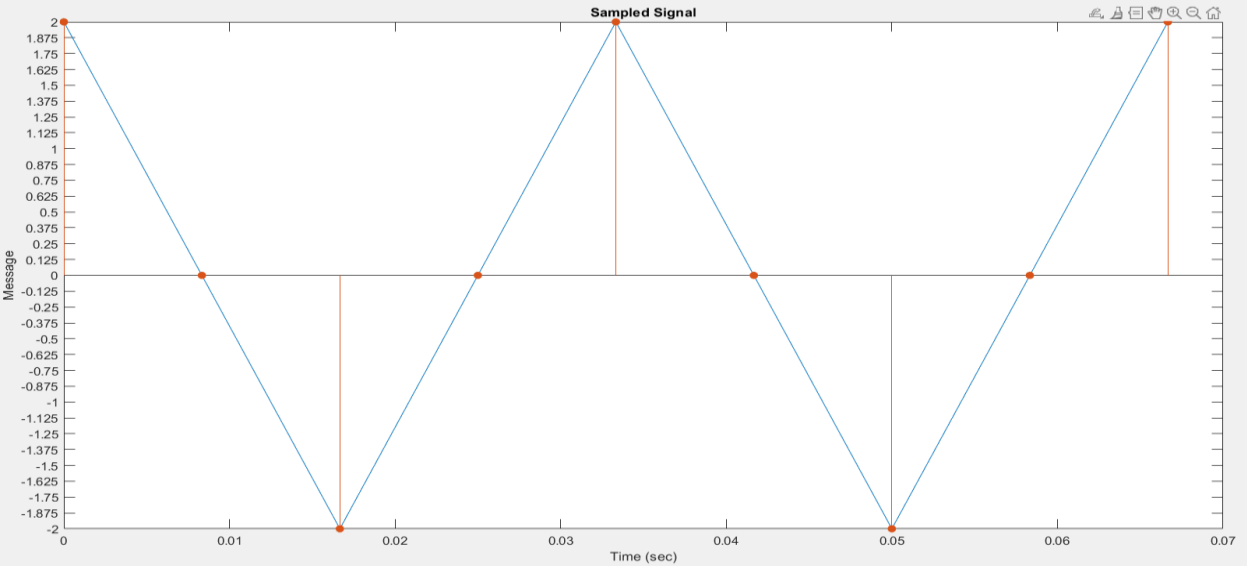


Fig.11

Here, Fs = 2*NR = 2*2*Fm = 120 Hz and Ts = 1/Fs = 1/120 sec.

So, we can see that the samples are taken every quarter period

Compared to the sampled signal in requirement 1:

They actually look the same just with more zeros interspersed between the previous samples, because all the added samples are at zero amplitude so they won't contribute any additional information about the signal's shape or characteristics

2- Quantized Signal

Same as requirement 1:

Number of bits "n" = 3 and Number of levels "L" = 8

Taking the levels as midrise and step size = 0.5

We have Levels at -1.75:0.5:1.75

And the samples are quantized as shown in the figure:

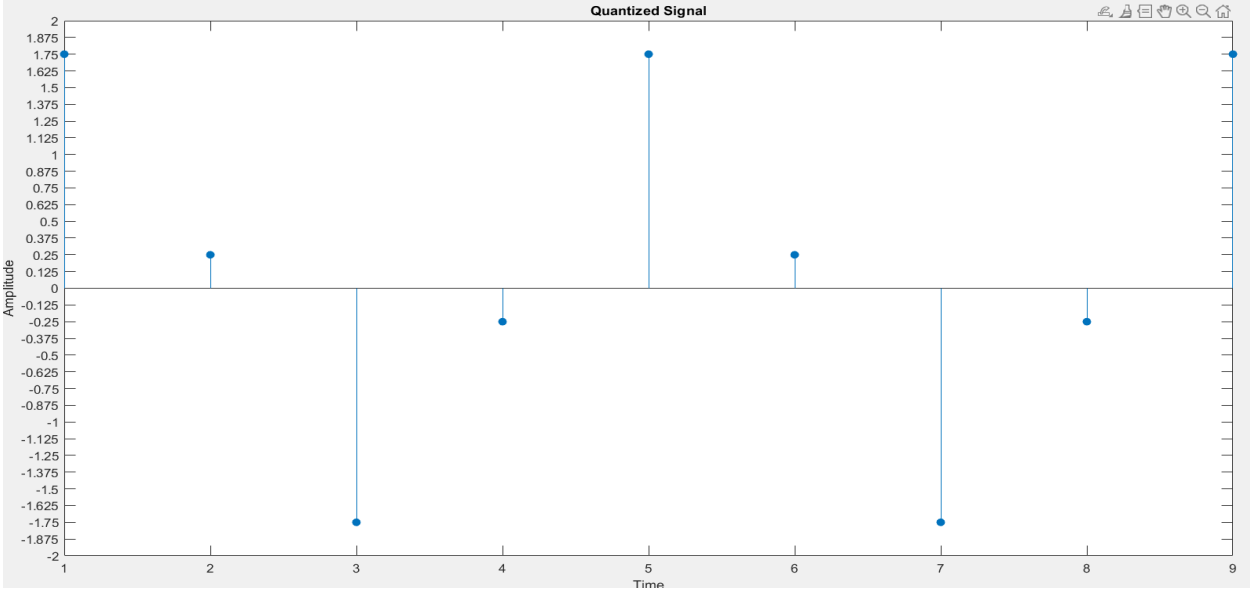


Fig.12

Since, we have more samples than requirement 1 so, there are more quantization values of the input signal despite that the sampled signal isn't affected, and this is because we're taking the levels as midrise so the samples which have zero amplitude will have an absolute quantization value of 0.25

<u>But why some of them have a quantization value of +0.25 and others of -0.25 ?</u>

Since, the values of the zero amplitude samples are represented as numerical values very close to zero but not exactly zero, one has a positive value that is approximately zero and one has a negative value that is approximately zero and so on….

This is due to the plotting accuracy and numerical precision limitations in matlab and it means that even if you want a number to be zero, due to these limits, the computer might represent it as a very tiny number close to zero instead of exactly zero.

3- Encoded Signal

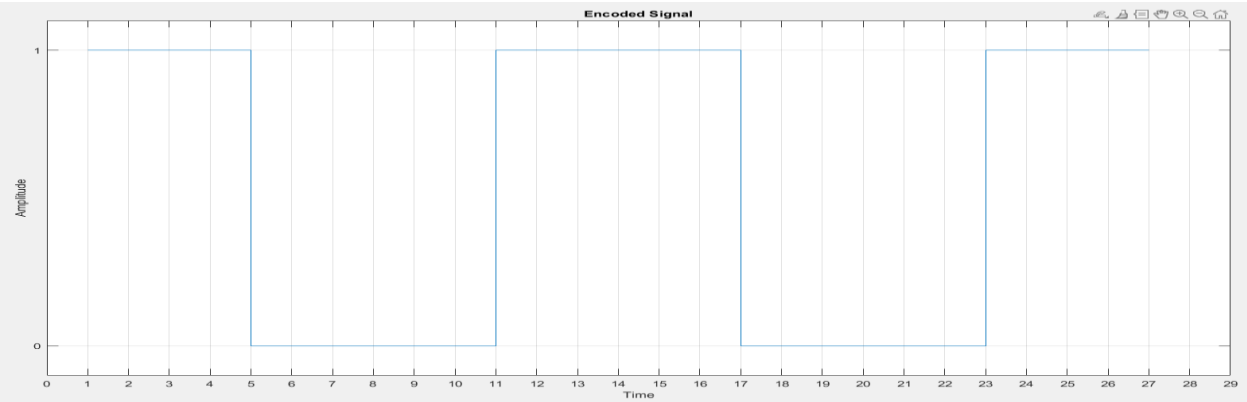
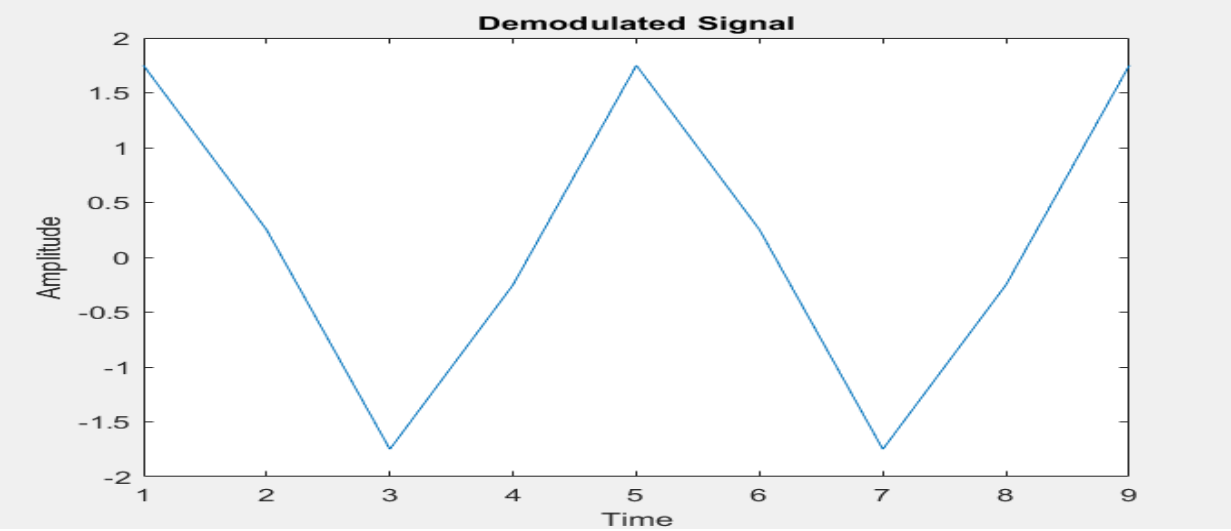


Fig.13

4- Recovered Signal



Fig,14

The recovered signal is not the same as the sampled signal due to the distortion introduced by the quantization process and can't

But Compared to the recovered signal in requirement 1:

This signal is a little bit different, since there are more samples so the quantization error is reduced

5- Bit Rate

We got Rb = 3*120 = 360 bps

Requirement 3

(If the Sampling rate equals the Nyquist rate)

(16 level quantizer)

1- The Sampled Signal

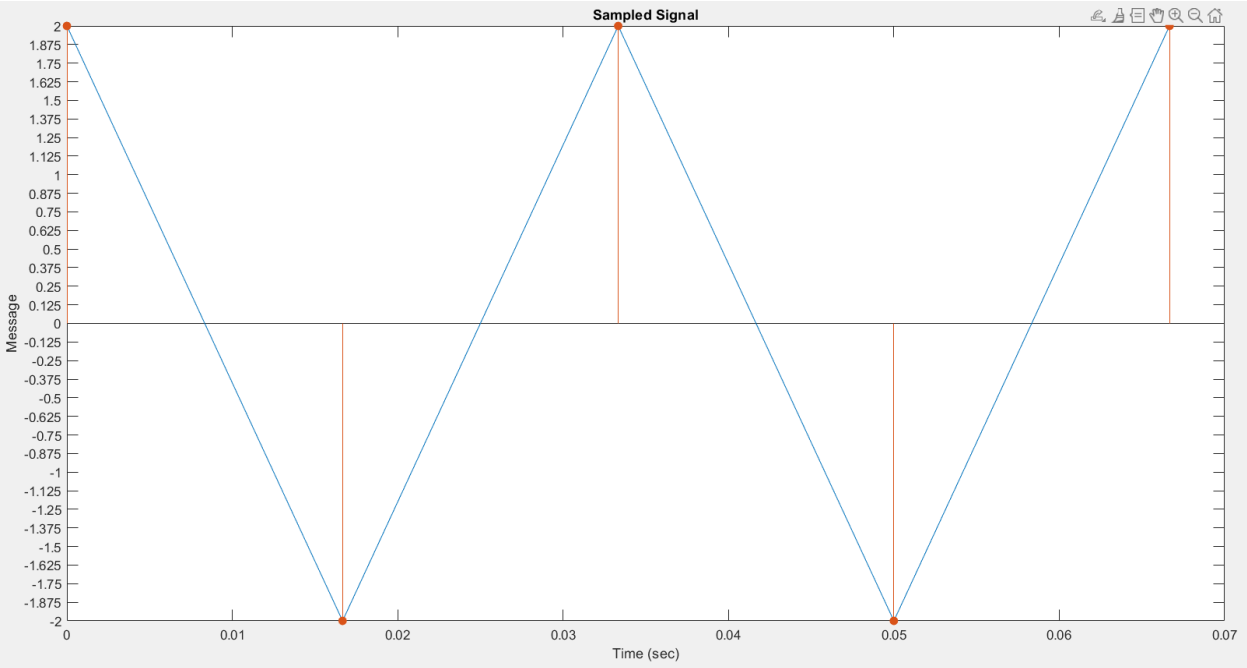


Fig.15

Here, Fs = NR = 2*Fm = 60 Hz and Ts = 1/Fs = 1/60 sec.

So, we can see that the samples are taken every half period as requirement 1

2- Quantized Signal

Number of bits “n” = 4 and Number of levels “L” = 16

Taking the levels as midrise and step size = (2-(-2))/16 = 0.25

We have Levels at -1.875:0.25:1.875

And the samples are quantized as shown in the figure:

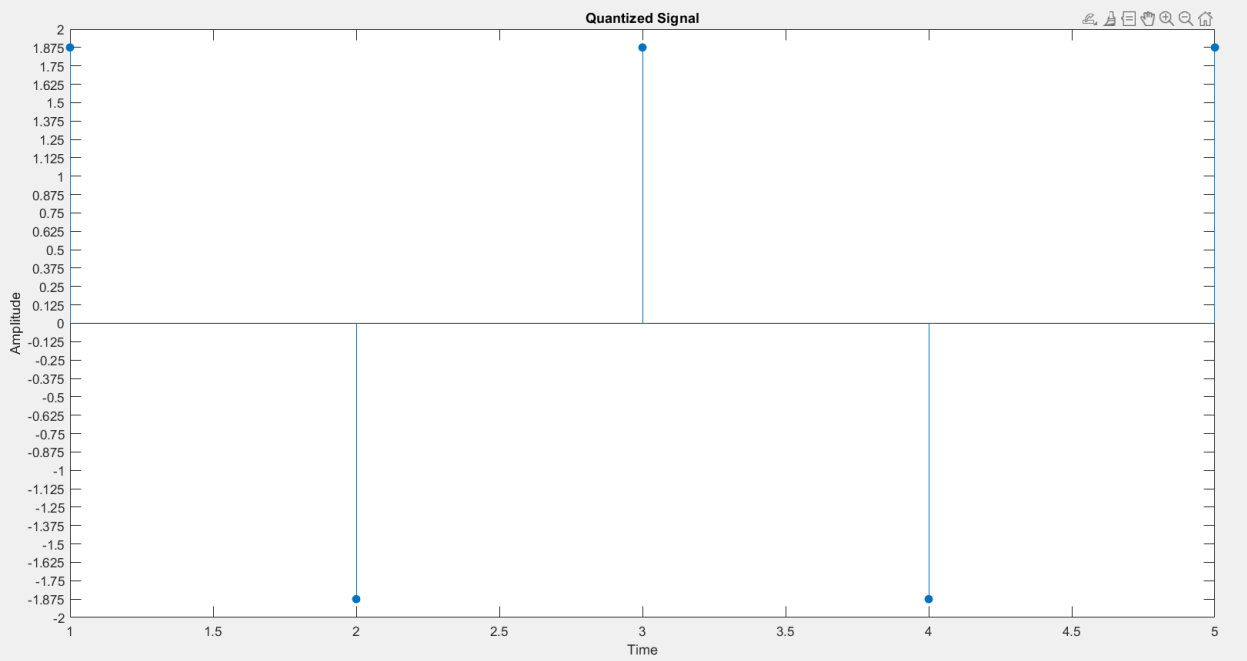


Fig.16

3- Encoded Signal

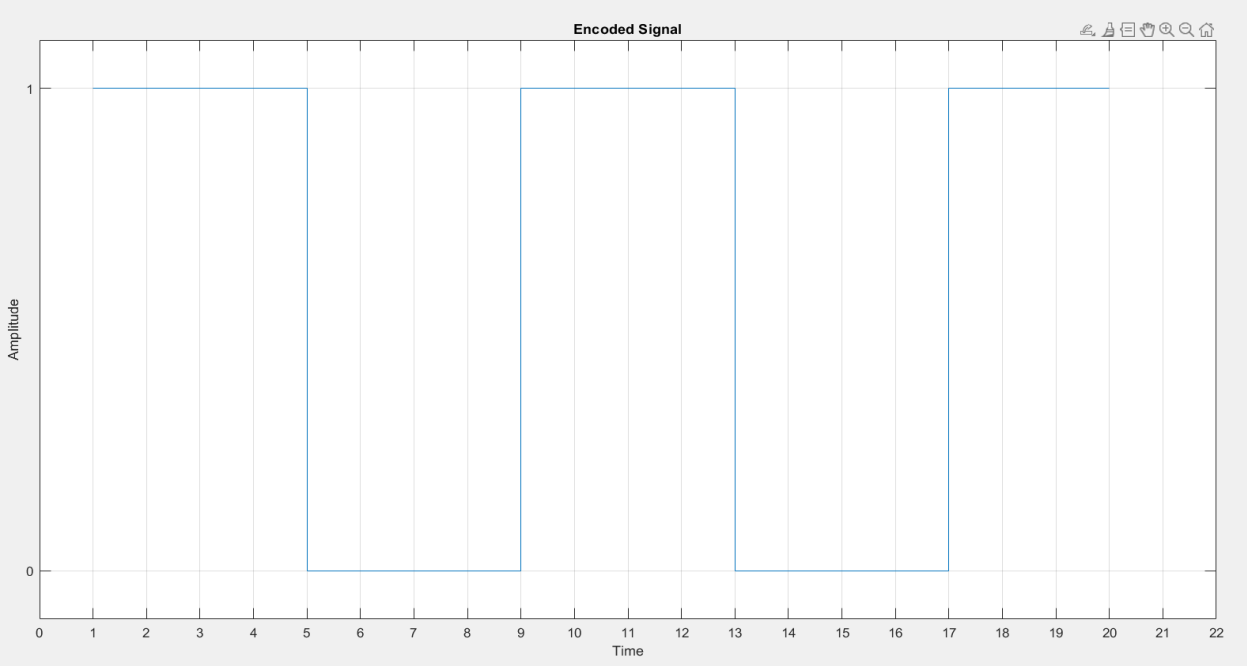


Fig.17

4- Recovered Signal

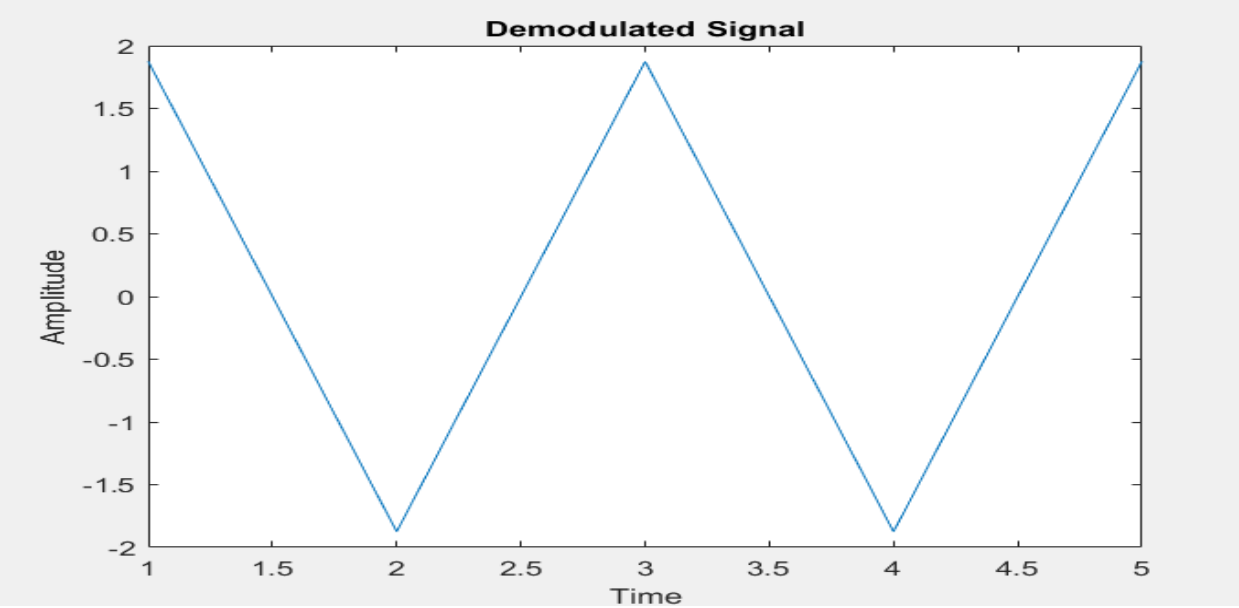


Fig.18

The recovered signal is near to the sampled messaged but there is a less quantization error than requirement 1 because number of level has increased ,and the recovered message is not like the original message because the number is very small to follow the original message.

5- Bit Rate

We got Rb = 4*60 = 240 bps