CERTIFICATE

This is to certify that

Krunal Rajeshbhai Rank

Of

B. Technology (2nd Year) Computer Engineering Division B

Roll No. U18CO081

has successfully completed the course

Communication Systems Laboratory

during the Year **2019 – 20**

at

Sardar Vallabhbhai National Institute of Technology

Teacher In-Charge

INDEX

EXPERIMENT 1: Basic Functions/Signals in MATLAB

Date: 20/01/2020

Aim: To plot basic functions sine, cosine, tangent and exponential in MATLAB. Plot basic signals such as unit impulse, unit step and unit ramp. Plot the periodic signals impulse train, square wave, saw tooth wave and triangular wave.

Theory/Equations:

Sine Wave:

A **sine wave** or **sinusoid** is a mathematical curve that describes a smooth periodic oscillation. A sine wave is a continuous wave. It is named after the function sine, of which it is the graph. It occurs often in pure and applied mathematics, as well as physics, engineering, signal processing and many other fields. Its most basic form as a function of time (*t*) is:

$$y(t) = A\sin(2\pi ft + \varphi) = A\sin(\omega t + \varphi)$$

where:

- ***A***, *amplitude*, the peak deviation of the function from zero.
- ***f***, *ordinary frequency*, the *number* of oscillations (cycles) that occur each second of time.
- $\omega = 2\pi f$, *angular frequency*, the rate of change of the function argument in units of radians per second
- φ, *phase*, specifies (in radians) where in its cycle the oscillation is at $t = 0$.

  When φ is non-zero, the entire waveform appears to be shifted in time by the amount φ/ω seconds. A negative value represents a delay, and a positive value represents an advance.

Cosine Wave:

A cosine wave is a signal waveform with a shape identical to that of a sine wave , except each point on the cosine wave occurs exactly 1/4 cycle earlier than the corresponding point on the sine wave. A cosine wave and its corresponding sine wave have the same frequency, but the cosine wave leads the sine wave by 90 degrees of phase .

$$f(t) = A\cos(2\pi ft + \varphi) = Asin(\omega t + \varphi)$$

Tangent Wave:

$$f(t) = \tan(t)$$

The tan function operates element-wise on arrays. The function accepts both real and complex inputs.
For real values of X, tan(X) returns real values in the interval [-∞, ∞].
For complex values of X, tan(X) returns complex values.

Exponential Function:

An exponential function can be defined as:

$$f(t) = e^t$$

It is expected to rise at a very fast rate within a short span of time. Usually, in electronics, a degrading exponential function is found common with the coefficient of 't' usually negative.

Unit Step Function:

$$f(t) = \begin{cases} 0, & x < 0 \\ 1, & x > 0 \end{cases}$$

Unit **Impulse Function:**

$$f(t) = \begin{cases} 0, & x = 0 \\ 1, & x \neq 0 \end{cases}$$

Ramp Function:

$$f(t) = \begin{cases} 0, & x < 0 \\ t, & x \geq 0 \end{cases}$$

Impulse Train function with a period T:

$$f(t) = \begin{cases} 0, & x \neq nT \\ 1, & x = nT \end{cases} n \epsilon Z$$

Square Wave function with a period T:

$$f(t) = 1(-1)^{\lfloor 2t/T \rfloor}$$

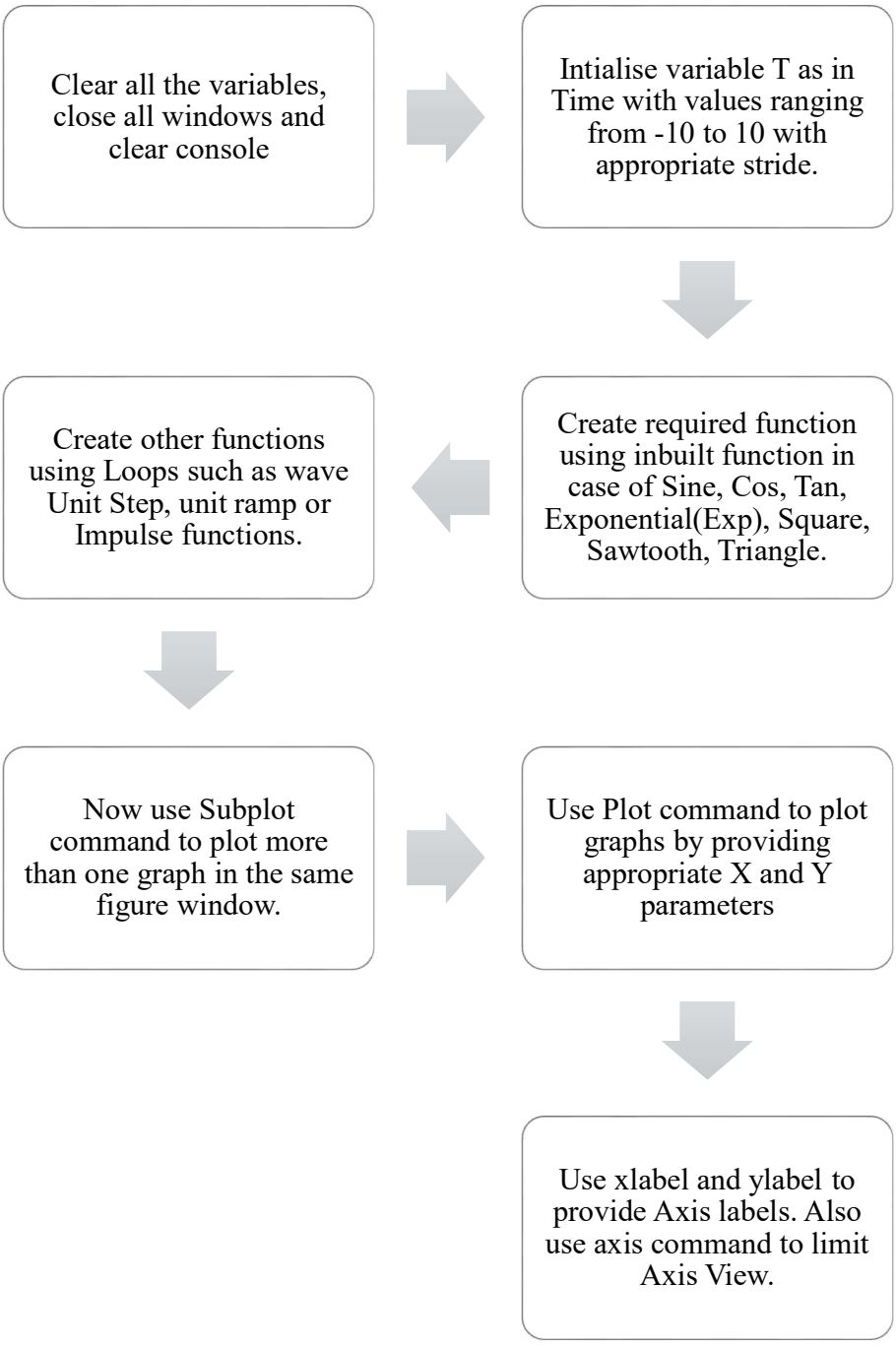
Saw-tooth Wave function with a period T:

$$f(t) = t - \lfloor t \rfloor$$

Triangle Wave function with a period T:

$$f(t) = \int_0^t sgn(\sin(t))dt$$

Flowchart/Algorithm:



1. Clear console, screen and close windows using the commands clc, clear all, close all.
2. Initialise t with values from -10 to 10 with step as necessary.
3. Use in-built sin(), cos(), tan(), exp() while using plot function to plot respective graphs.
4. For Unit step function, create unit_step array with value 1 when t>0 and 0 otherwise, and plot it against t.
5. For Unit Impulse function, create unit_impulse array with value 1 when t=0 and 0 otherwise, and plot it against t.
6. For Ramp function, create ramp array with value t when t>0 and 0 otherwise and plot it against t.
7. For Impulse train function, create impulse_train array with value 1 when t = nT and 0 otherwise using for loop.
8. For square function, use in-built function square().
9. For sawtooth function, use in-built function sawtooth().
10. For triangle function, use in built function sawtooth() with width=0.5.
11. Use subplot command to create more than one graphs in one figure window. Use subplot command before plotting any graph such as subplot(<graph count x>,<graph count y>,<graph pos>) where graph count x = No. of graphs to be displayed horizontally, graph count y = No. of graphs to be displayed vertically, graph pos = Position of graph on the window which is usually row majored (1 – top left , 2 –top right , 3 – bottom left, 4 – bottom right in case of 2 by 2 subplot).
12. To plot graph, use plot() command with 1st parameter as the quantity for X axis, that is 't' and 2nd parameter as the quantity for Y axis, that is, any one of the above derived quantities.
13. To provide X axis label, use xlabel(<string>) and to provide Y axis label, use ylabel(<string>).
14. Use title(<string>) to provide title to the graph.
15. Use axis([]) to provide axis limits to the graph.
16. Do the above process for all required graphs.
17. It is recommended to use Sections as shown in the Code section so as to plot the graphs in a legible form. Sections can be used in the following fashion:- %% <title of section>
18. Run the written code using F5 or section using CTRL + F5.
19. Save the graphs from File > Save As.
20. The required experiment has been completed successfully.

Code:

```
%% Part A


clc; %Clear console

clear all; %Clear variables

close all; %Close all windows except Editor window


t = -10:0.001:10; %Initialise t

subplot(2,2,1); %Using subplot to plot next graphs

plot(t,sin(t)); %Plotted Sine graphs

xlabel('Time(in s)'); %X Label for Sine graph

ylabel('Voltage(in V)'); %Y Label for Sine graph

title('Sine');%Title for Sine graph

axis([-10 10 -2 2]); %Axis limitations for Sine graph


subplot(2,2,2);

plot(t,cos(t)); %Plotted Cosine graph

axis([-10 10 -2 2]);

xlabel('Time(in s)');

ylabel('Voltage(in V)');
```

```
title('Cosine');


subplot(2,2,3);
plot(t,tan(t)); %Plotted Tangent graph
axis([-10 10 -2 2]);
xlabel('Time(in s)');
ylabel('Voltage(in V)');
title('Tangent');


subplot(2,2,4);
plot(t,exp(t)); %Plotted Exponential graph
axis([-10 10 -2 20000]);
xlabel('Time(in s)');
ylabel('Voltage(in V)');
title('Exponential');
%% Part B
clc;
clear all;
close all;
```

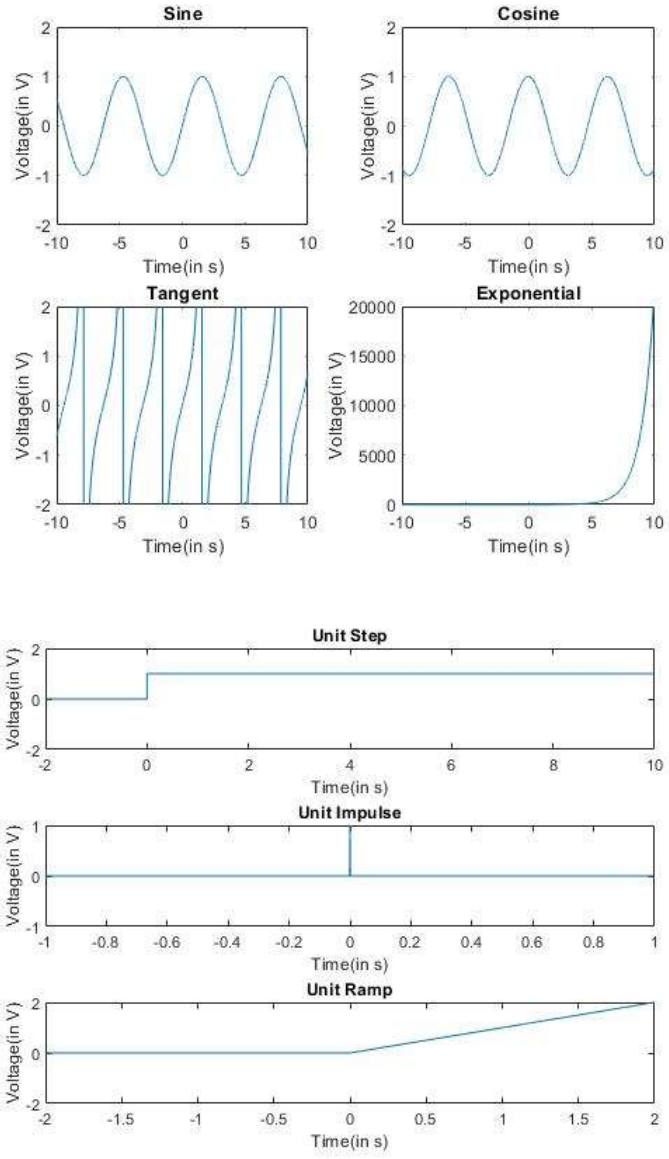
```
t = -10:0.001:10;

unit_step = double(t>0); %Set values for unit_step for different values
of t

unit_impulse = double(t==0); %Set value for unit_impulse for different
values of t (1 when t=0 else 0)

unit_ramp = zeros(size(t)); %Initialise unit_ramp with 0

[m n] =  size(t); %Retrieve size of t for the below loop

for i=1:n

   if (t(1,i)>0)

       unit_ramp(1,i)=t(1,i); %Set unit_ramp = t when t>0

   end

end


subplot(3,1,1);

plot(t,unit_step) %Plotted Unit Step graph

axis([-2 10 -2 2]);

xlabel('Time(in s)');

ylabel('Voltage(in V)');

title('Unit Step');


subplot(3,1,2);
```

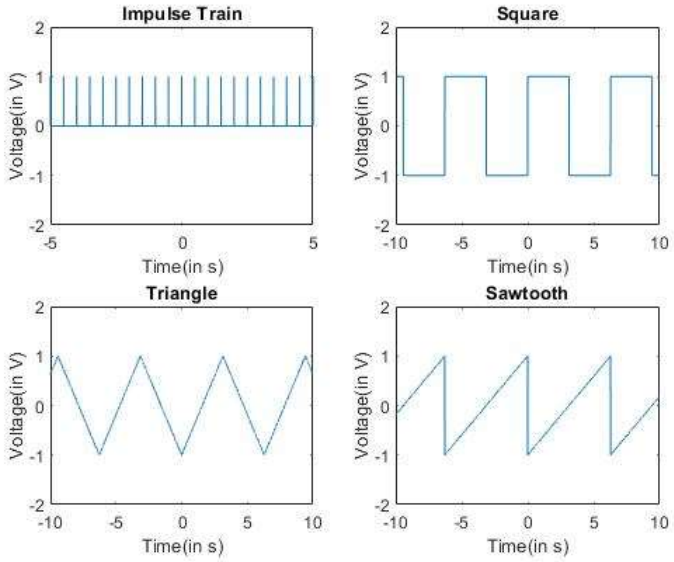
```
plot(t,unit_impulse) %Plotted Unit Impulse graph

axis([-1 1 -1 1]);

xlabel('Time(in s)');

ylabel('Voltage(in V)');

title('Unit Impulse');



subplot(3,1,3);

plot(t,unit_ramp) %Plotted Unit Ramp graph

axis([-2 2 -2 2]);

xlabel('Time(in s)');

ylabel('Voltage(in V)');

title('Unit Ramp');



%% Part C



clc;

clear all;

close all;



t = -10:0.001:10;
```

```
time_period =0.5; %Set period for Impulse strain and triangle function

[m n] = size(t);



impulse_train=double(mod(t,time_period)==0);   %Iterating Impulse over
certain period

sqr = square(t); %Generating Square Wave

saw = sawtooth(t); %Generating Sawtooth Wave

trng1 = sawtooth(t,time_period); %Generating triangle wave by setting
width = time_period


subplot(2,2,1);

plot(t,impulse_train); %Plotted Impulse Train graph

axis([-5,5 -2 2]);

xlabel('Time(in s)');

ylabel('Voltage(in V)');

title('Impulse Train');


subplot(2,2,2);

plot(t,sqr); %Plotted Square Wave graph

axis([-10 10 -2 2]);
```

```
xlabel('Time(in s)');

ylabel('Voltage(in V)');

title('Square');


subplot(2,2,3);

plot(t,trngl); %Plotted Triangle Wave graph

axis([-10 10 -2 2]);

xlabel('Time(in s)');

ylabel('Voltage(in V)');

title('Triangle');


subplot(2,2,4);

plot(t,saw); %Plotted Sawtooth Wave graph

axis([-10 10 -2 2]);

xlabel('Time(in s)');

ylabel('Voltage(in V)');

title('Sawtooth');
```

Result/Output Waveforms:





Conclusion:

The basic functions such as sine, cosine, tangent and exponential as well as various signals such as unit impulse, unit step and unit ramp and periodic signals such as impulse train, square wave, saw tooth and triangular wave have been plotted using MATLAB successfully and appropriate code along with diagram have been observed and mentioned.

Remarks: **Signature**

EXPERIMENT 2: SAMPLING AND RECONSTRUCTION OF SIGNAL

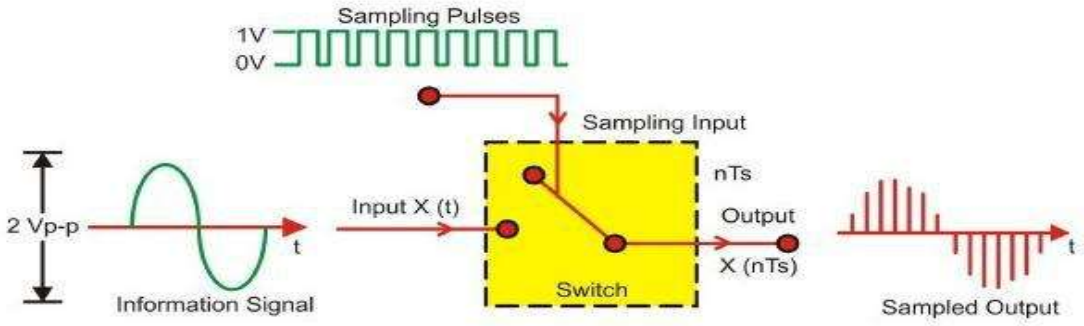
Date: 27/01/2020

Aim: Study of Sampling and Reconstruction of signal. Verify Nyquist criteria. Model ST21O1 W kit, connecting wires, CRO/DSO

Apparatus: Model ST 2151 W kit, connection wires, CRO/DSO

Sampling Theory:

The signals we use in the real world, such as our voice, are called "analog" signals. To process these signals for digital communication, we need to convert analog signals to "digital" form. While an analog signal is continuous in both time and amplitude, a digital signal is discrete in both time and amplitude. To convert continuous time signal to discrete time signal, a process is used called as sampling. The value of the signal is measured at certain intervals in time. Each measurement is referred to as a sample.



In electronics, a sample and hold circuit is used to interface real world, changing analogue signals to a subsequent system such as an analog-to-digital converter. The purpose of this circuit is to hold the analogue value steady for a short time while the converter or other following system performs some operation that takes a little time. In most circuits, a capacitor is used to store the analogue voltage and an electronic switch or gate is used to alternately connect and disconnect the capacitor from the analogue input. The rate at which this switch is operated is the sampling rate of the system.

NYQUIST CRITERION (SAMPLING THEOREM):

The Nyquist Criterion states that a continuous signal band limited to f_m Hz can be completely represented by and reconstructed from the samples taken at a rate greater than or equal to 2fm samples/second. The minimum sampling frequency is calls as NYQUIST RATE i.e. for faithful reproduction.

SAMPLE AND HOLD:

One way to maintain reasonable pulse energy is to hold the sample value until the next sample is taken. This technique is formed as Sample and Hold technique. A buffered Sample and Hold circuit consists of unity gain buffers preceding and succeeding the charging capacitor. The high input impedance of the proceeding buffer prevents the loading of the message source and also ensures that the capacitor charges by a constant rate irrespective of the source impedance.

Procedure:

A. Set up for Sampling and reconstruction of signal.

Initial set up of trainer:
Duty cycle selector switch position : Position 5
Sampling selector switch : Internal position

1. Connect the power cord to the trainer. Keep the power switch in 'Off' position.
2. Connect 1 KHz Sine wave to signal Input as shown in Fig.1.1.
3. Switch 'On' the trainer's power supply & Oscilloscope.
4. Connect BNC connector to the CRO and to the trainer's output port.

You can observe the process of step-by-step generating sine wave signal from Square wave of 1 KHz at TP3, TP4, TP5 and TP6 respectively.

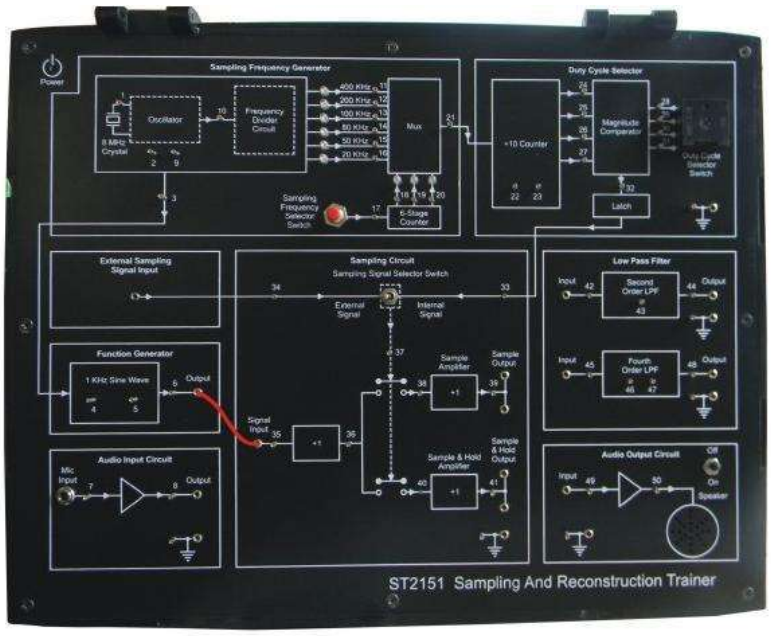


Fig. 1.1. Connection diagram for sampling a signal

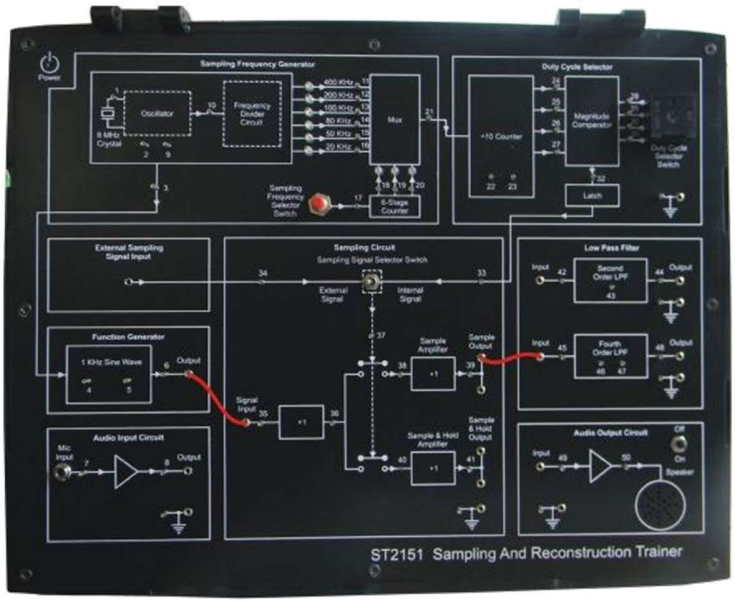


Fig. 1.2. Connection diagram for reconstruction of a sampled signal

B. Set up for effect of Sample Amplifier and Sample and Hold Amplifier on reconstructed signal.

Set up for effect of II order and IV order Low Pass Filter on reconstructed signal.

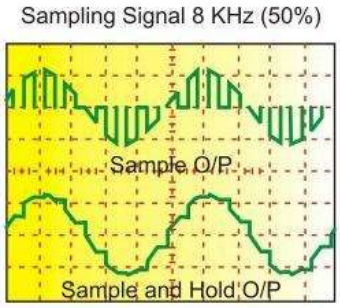
Initial set up of trainer:
Duty cycle selector switch position : Position 5
Sampling selector switch: Internal position

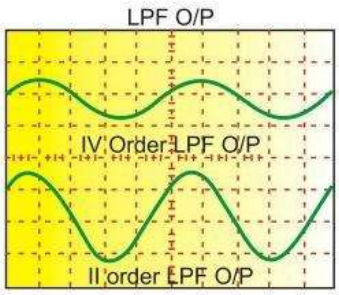
1. Connect the power cord to the trainer. Keep the power switch in ‘Off’ position.
2. Connect 1 KHz Sine wave to signal Input.
3. Switch ‘On’ the trainer's power supply & Oscilloscope.
4. Connect BNC connector to the CRO and to the trainer’s output port.
5. Select sampling frequency of 8 KHz by Sampling Frequency Selector Switch pressed till
6. KHz signal LED glows.
7. Observe 1 KHz sine wave and Sample Output (TP39) on oscilloscope. The display shows
8.KHz sine wave being sampled at 8 KHz, so there are 8 samples for every cycle of the sine wave.
9. Connect Sample Output to Fourth Order low pass filter Input as shown in figure 1.2. Observe the filtered output (TP48) on the oscilloscope. The display shows the reconstructed 1 KHz sine wave.

10. Similarly observe the sampled 1 KHz sine wave at and Sample and Hold Output (TP41) on oscilloscope. The display shows 1 KHz sine wave being sampled and hold signal at 8 KHz. Connect Sample and Hold Output to Second Order low pass filter Input and observe the filtered output (TP44) on oscilloscope. The display shows the reconstructed1 KHz sine wave.

11. By pressing Sampling Frequency Selector Switch, change the sampling frequency from 2 KHz, 5 KHz, 10 KHz, 20 KHz up to 40 KHz (Sampling frequency is 1/10th of the frequency indicated by the illuminated LED). Observe how Sample output (TP39) and Sample and Hold Output (TP41) changes in each case.

Sample Observations:





Observation Table:

Input Frequency (In KHz)	Sampling Frequency (In KHz)	Duty Cycle (In %)	Order of Low Pass Filter	Output Frequency
1.000	2	50	2	1.000
1.000	2	50	4	1.000
1.000	5	50	2	0.998
1.000	5	50	4	1.000
1.000	20	50	2	0.996
1.000	20	50	4	1.003
1.000	10	10	4	0.999
1.000	10	30	4	1.000
1.000	10	60	4	1.000
1.000	10	90	4	1.001

Output Waveforms:

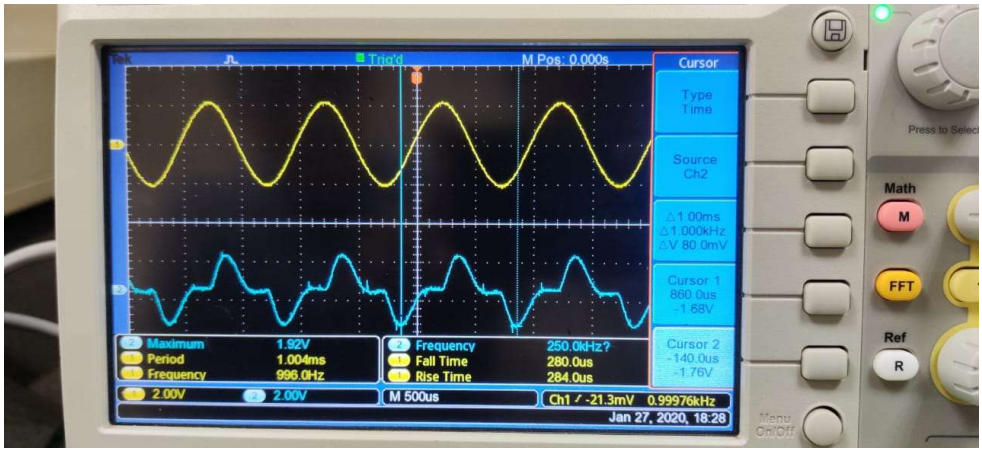


Figure 1: Sampling Frequency : 2KHz, 50% Duty Cycle, 2nd Order Low Pass Filter Output

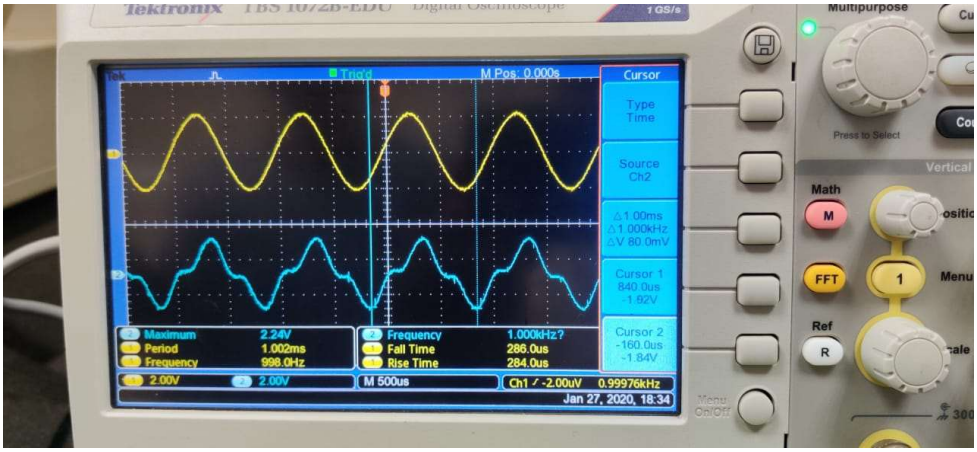


Figure 2: Sampling Frequency : 2KHz, 50% Duty Cycle, 4th Order Low Pass Filter Output

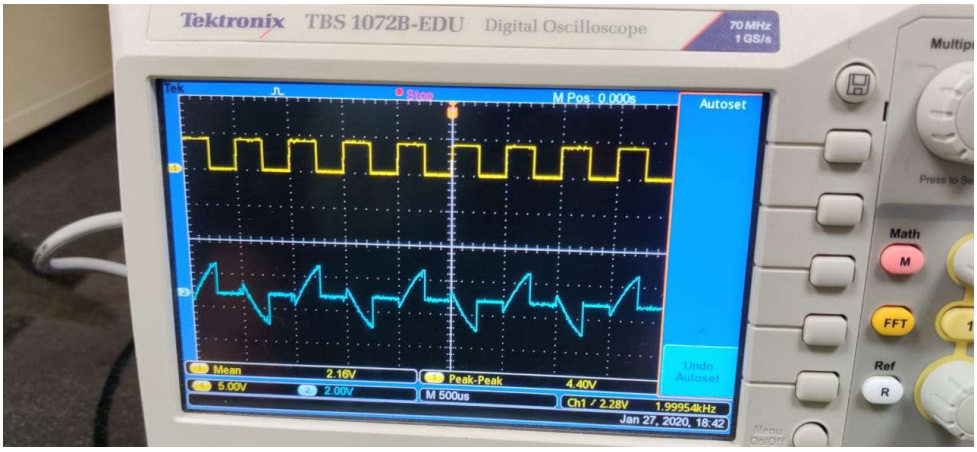


Figure 3: Sampling Frequency : 2KHz, 50% Duty Cycle, Sample Amplifier Output

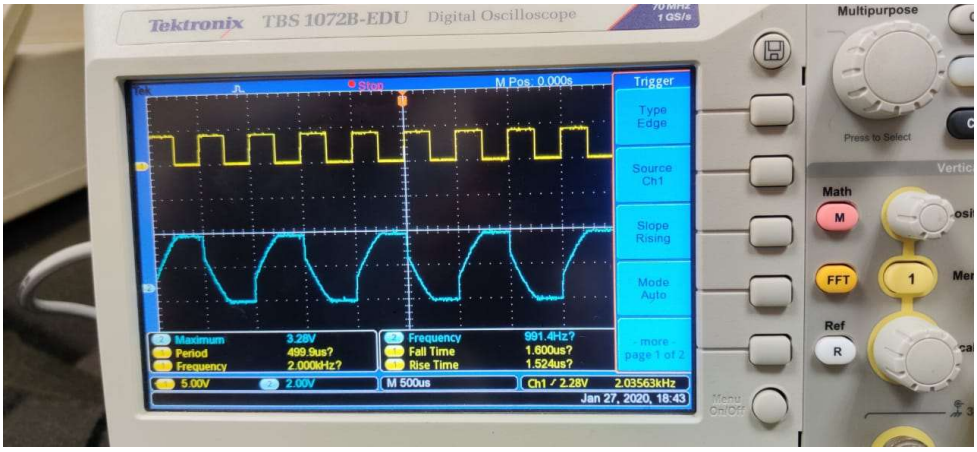


Figure 4: Sampling Frequency : 2KHz, 50% Duty Cycle, Sample and Hold Circuit Output

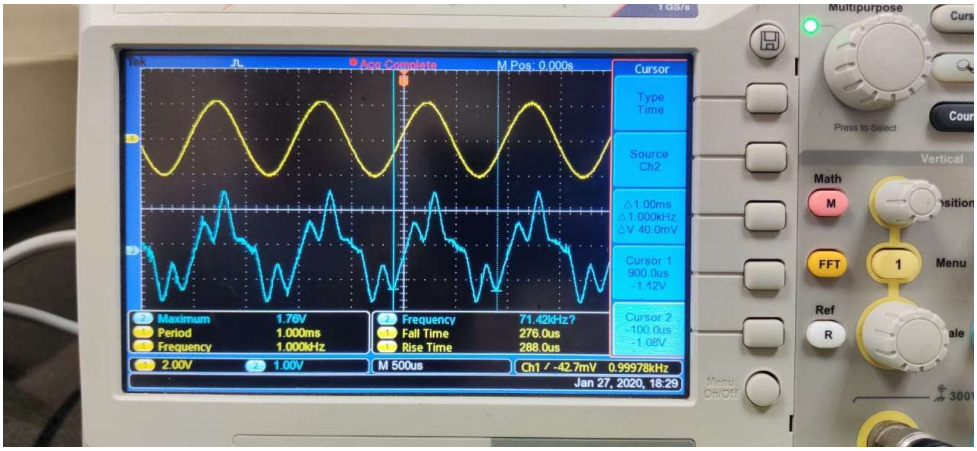


Figure 5: Sampling Frequency : 5KHz, 50% Duty Cycle, 2nd Order Low Pass Filter Output

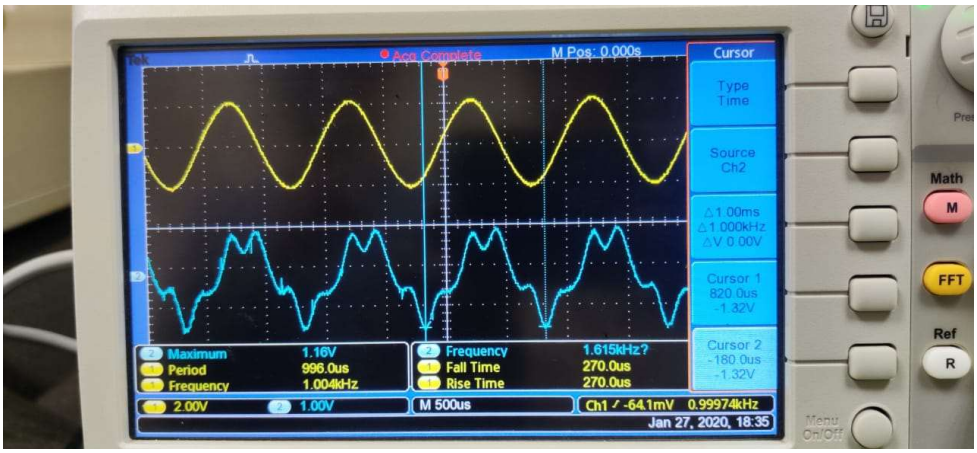


Figure 6: Sampling Frequency : 5KHz, 50% Duty Cycle, 4th Order Low Pass Filter Output

Figure 7: Sampling Frequency : 5KHz, 50% Duty Cycle, Sample Amplifier Output

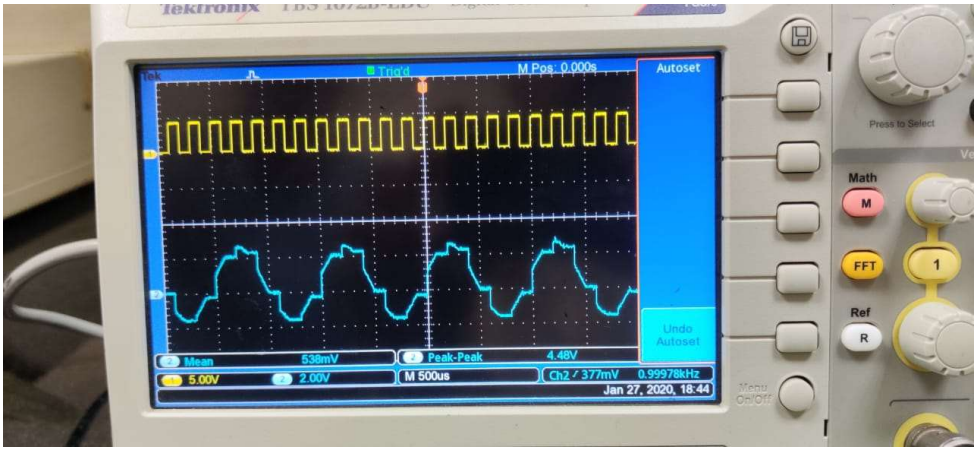


Figure 8: Sampling Frequency : 5KHz, 50% Duty Cycle, Sample and Hold Circuit Output

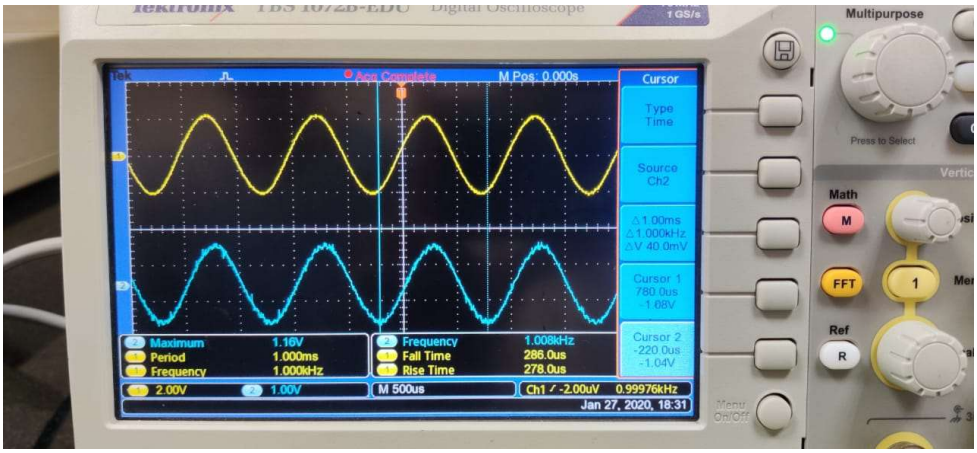


Figure 9: Sampling Frequency : 20KHz, 50% Duty Cycle, 2nd Order Low Pass Filter Output

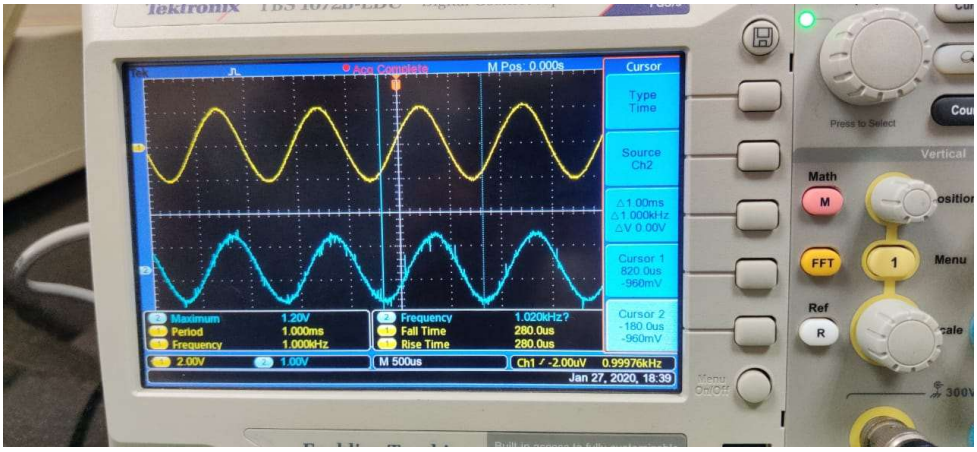


Figure 10: Sampling Frequency : 20KHz, 50% Duty Cycle, 4th Order Low Pass Filter Output

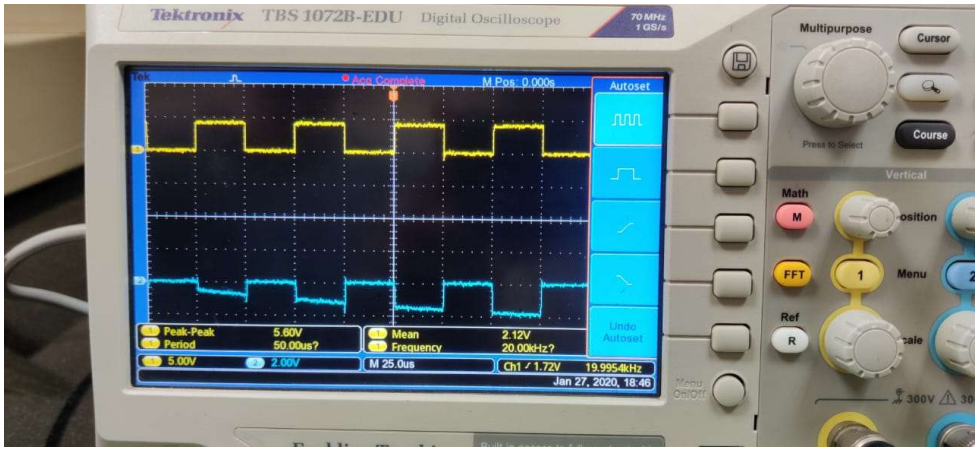


Figure 11: Sampling Frequency : 20KHz, 50% Duty Cycle, Sample Amplifier Output

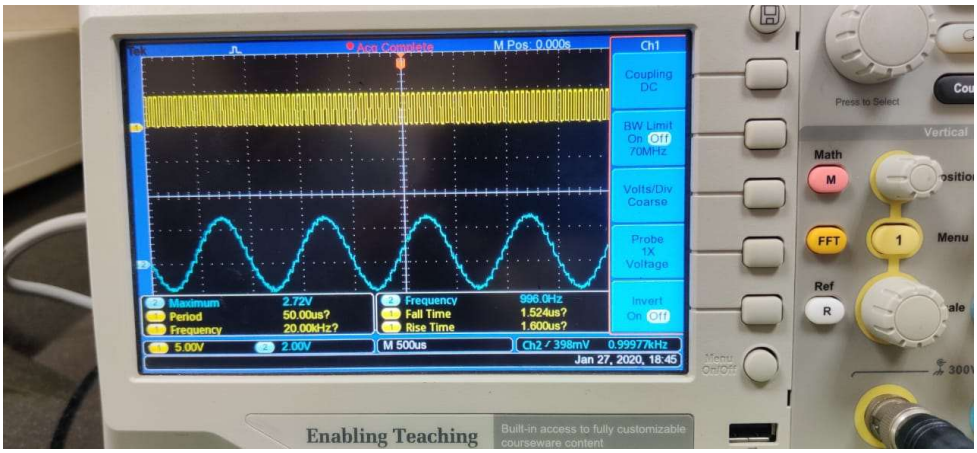


Figure 12: Sampling Frequency : 20KHz, 50% Duty Cycle, Sample and Hold Circuit Output

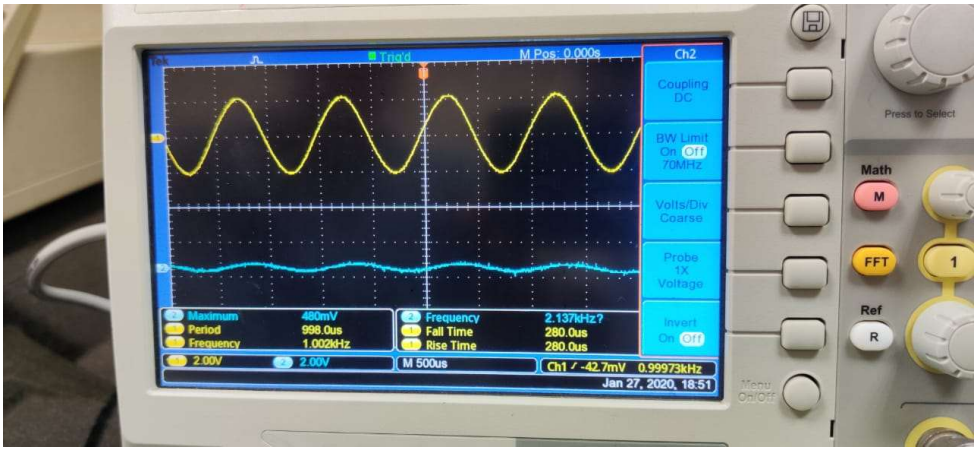


Figure 13: Sampling Frequency : 10KHz, 10% Duty Cycle, 4th Order Low Pass Filter Output

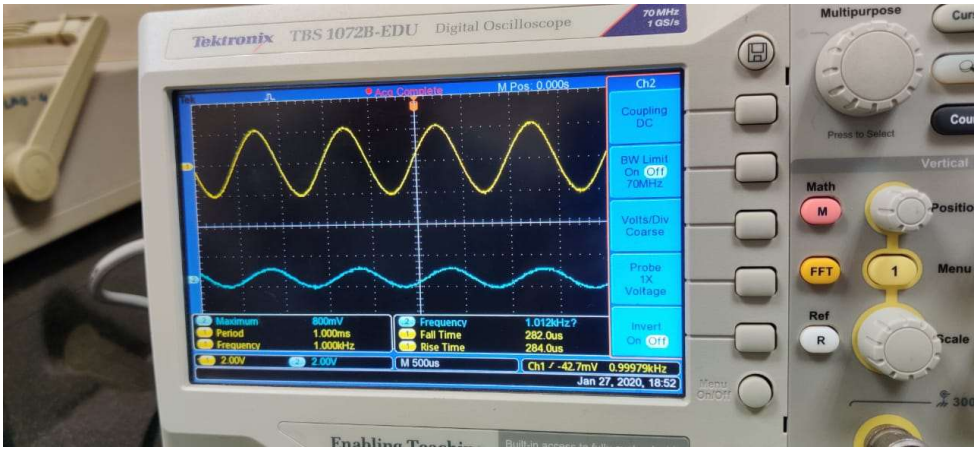


Figure 14: Sampling Frequency : 10KHz, 30% Duty Cycle, 4th Order Low Pass Filter Output

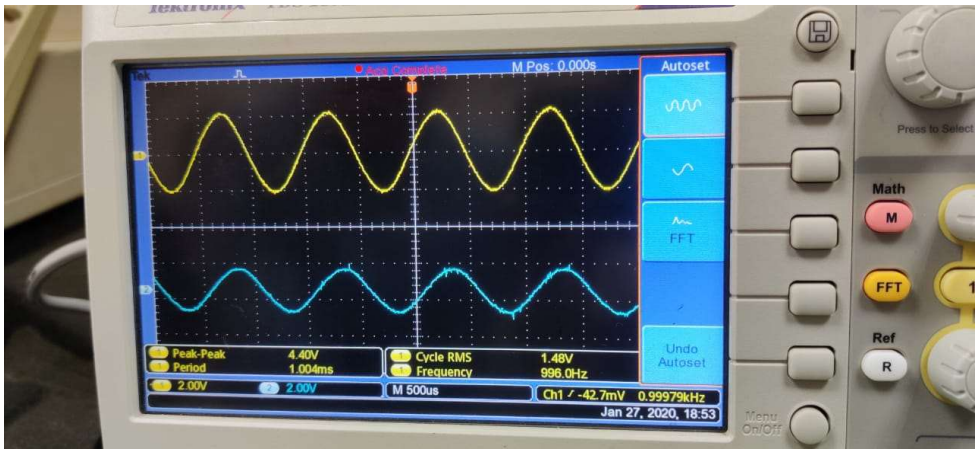


Figure 15: Sampling Frequency : 10KHz, 60% Duty Cycle, 4th Order Low Pass Filter Output

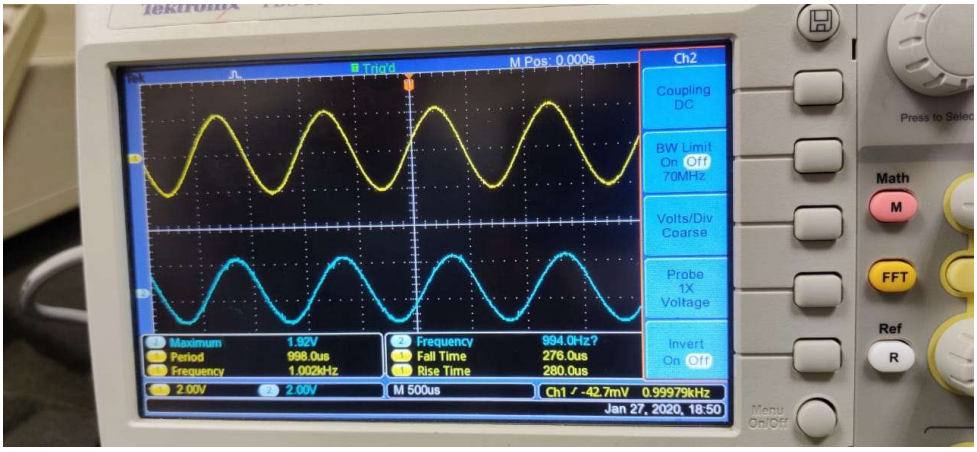


Figure 16: Sampling Frequency : 10KHz, 90% Duty Cycle, 4th Order Low Pass Filter Output

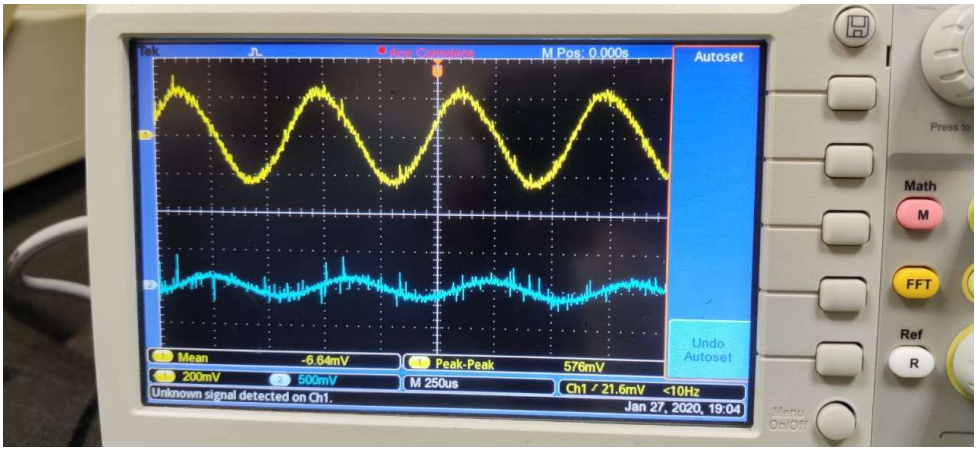


Figure 17: Microphone Voice Output

Conclusion:

Sampling and reconstruction of the signal with frequency 1KHz is thoroughly studied by varying the frequency of Sampling, duty cycle, switching between Sample Amplifier and Sample and Hold Circuit and switching between 2nd Order and 4th Order Low Pass filter.

Ultimately, Nyquist criteria is successfully verified by recording output waveforms as shown above.

Remarks: **Signature**

EXPERIMENT 3: PULSE AMPLITUDE MODULATION/DEMODULATION

Date: 03/02/2020

Aim: To study Pulse Amplitude Modulation:

1. **To modulate signal by Pulse Amplitude Modulation Scheme using Natural & Flat top sampling.**
2. **To demodulate signal by Pulse Amplitude Modulation Scheme using Sample & Hold, Flat Top.**
3. **Verify the sampling theorem by changing modulating & carrier frequency.**

Apparatus: ST2110 PAM trainer kit, DSO, Connecting Leads, Probes

Theory:

Pulse-amplitude modulation (PAM), is a form of signal modulation where the message information is encoded in the amplitude of a series of signal pulses. It is an analog pulse modulation scheme in which the amplitudes of a train of carrier pulses are varied according to the sample value of the message signal. Demodulation is performed by detecting the amplitude level of the carrier at every single period.

There are two types of pulse amplitude modulation:

Single polarity PAM: In this a suitable fixed DC bias is added to the signal to ensure that all the pulses are positive.

Double polarity PAM: In this the pulses are both positive and negative.

Pulse-amplitude modulation is widely used in modulating signal transmission of digital data, with non-baseband applications having been largely replaced by pulse-code modulation, and, more recently, by pulse-position modulation.

In particular, all telephone modems faster than 300 bit/s use quadrature amplitude modulation (QAM). (QAM uses a two-dimensional constellation).

Procedure:

1. Connect the circuit as shown in Fig. 1.1.
2. Output of Sine wave to Modulation Signal in PAM block keeping the switch in 1 kHz position
3. 8 kHz pulse output to Pulse IN
4. Switch On power supply.
5. Monitor the outputs at tp. 3, 4 and 5, these are natural, Sample and Hold flat top outputs respectively.
6. Observe the difference between the two outputs and try giving reasons behind them.
7. Try Varying the amplitude and frequency of sine wave by amplitude pot and frequency change over switch. Observe the effect on all the two outputs.
8. Also, try varying the frequency of pulse, by connecting the pulse input to the 4 frequencies available i.e. 8.16.32. 64 kHz in pulse output look.
9. For demodulation part Connect the sample output low pass filter input and Output of low pass filter to input of AC amplifier. Keep the gain pot in AC amplifier block in max position.
10. Follow the steps as of modulation part.
11. Monitor the output of AC amplifier. It should be a pure sine wave similar to input.
12. Similarly connect the sample and hold and flat top outputs to Law Pass Filter and see the demodulated waveform at the output of AC amplifier.

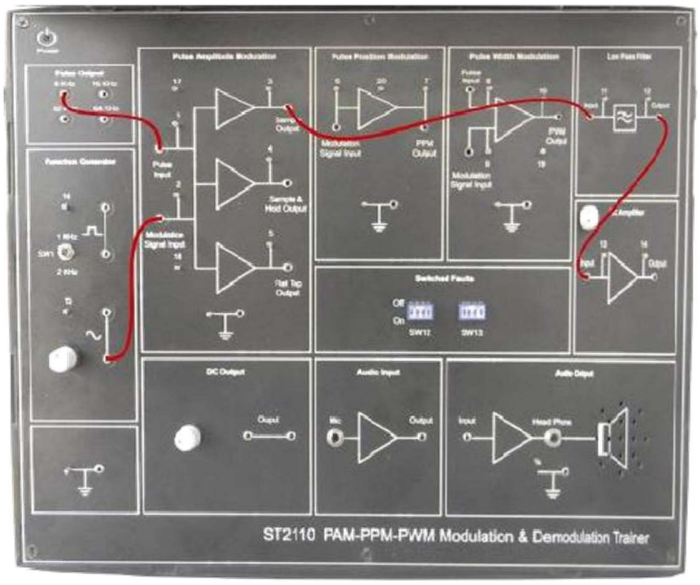


Fig. Connection diagram for Pulse Amplitude Modulation and Demodulation

Observation: Output Voltage: 20 V peak to peak

Output Waveforms:

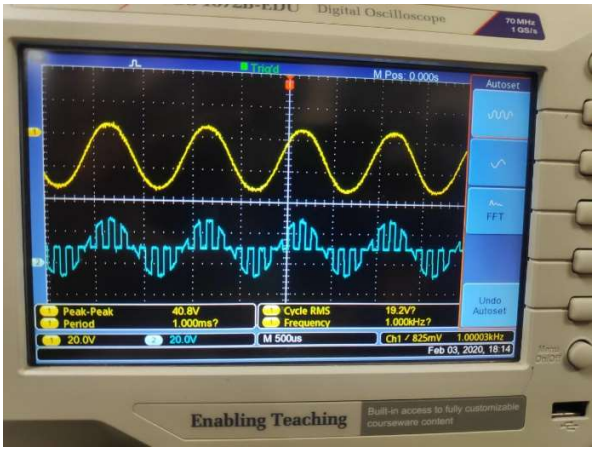


Figure 1: Sampling Frequency: 8KHz, Pulse Amplitude Modulation Sampled Output

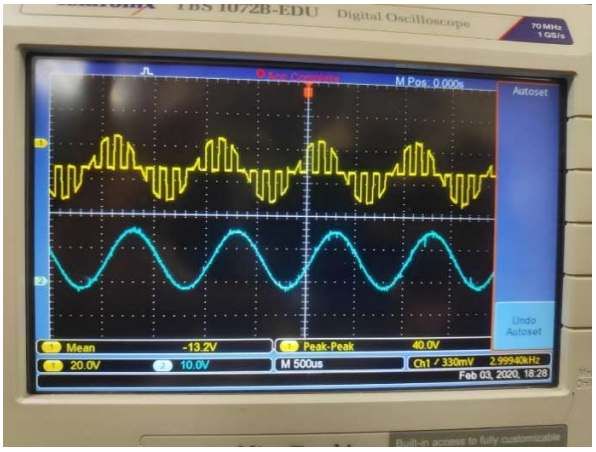


Figure 2: Sampling Frequency: 8KHz, Pulse Amplitude Modulation Sampled Output Reconstructed

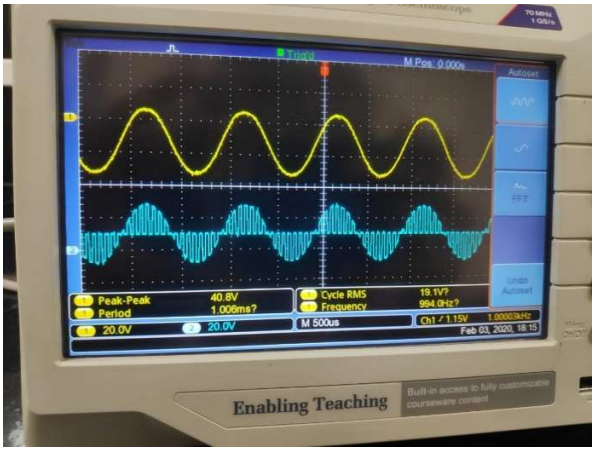


Figure 3: Sampling Frequency: 16KHz, Pulse Amplitude Modulation Sampled Output

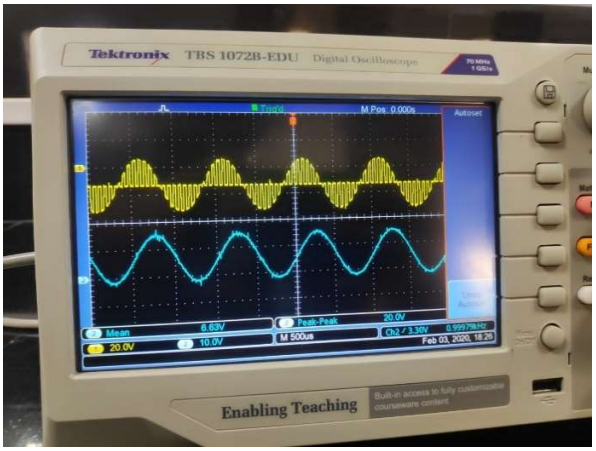


Figure 4: Sampling Frequency: 16KHz, Pulse Amplitude Modulation Sampled Output Reconstructed

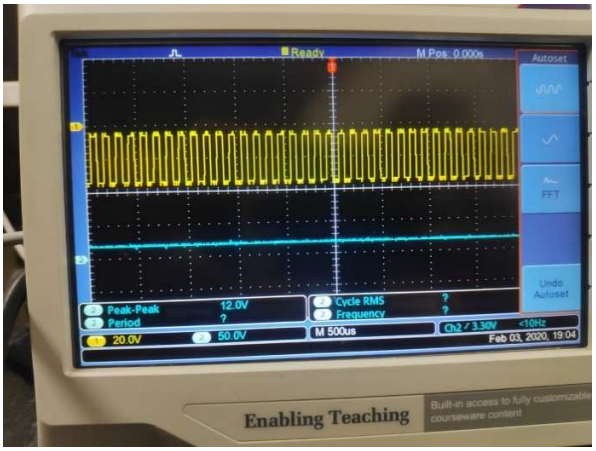


Figure 5: Sampling Frequency: 8KHz, Pulse Amplitude Modulation DC Supply

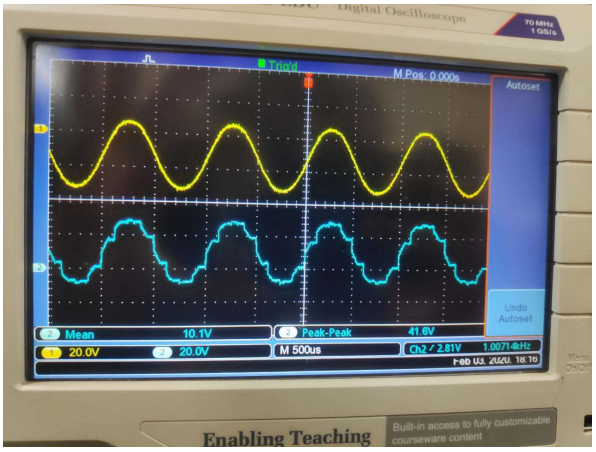


Figure 6: Sampling Frequency: 8KHz, Pulse Amplitude Modulation Sample and Hold Output

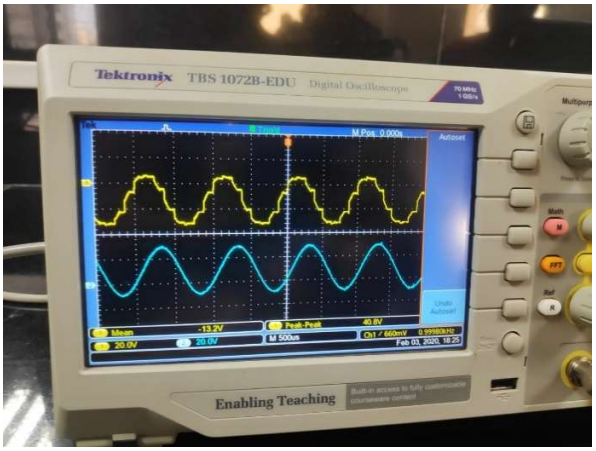


Figure 7: Sampling Frequency: 8KHz, Pulse Amplitude Modulation Sample and Hold Output Reconstructed

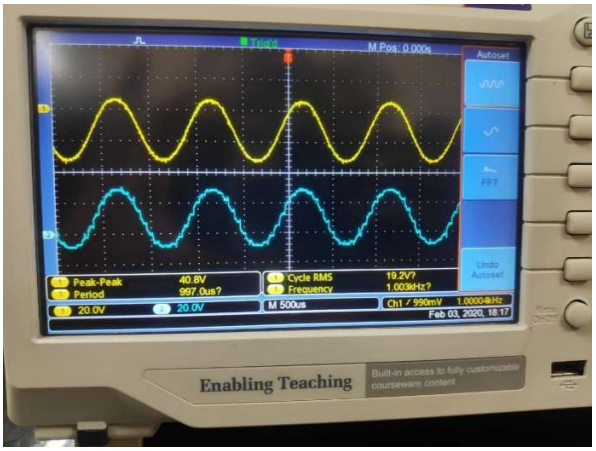


Figure 8: Sampling Frequency: 16KHz, Pulse Amplitude Modulation Sample and Hold Output

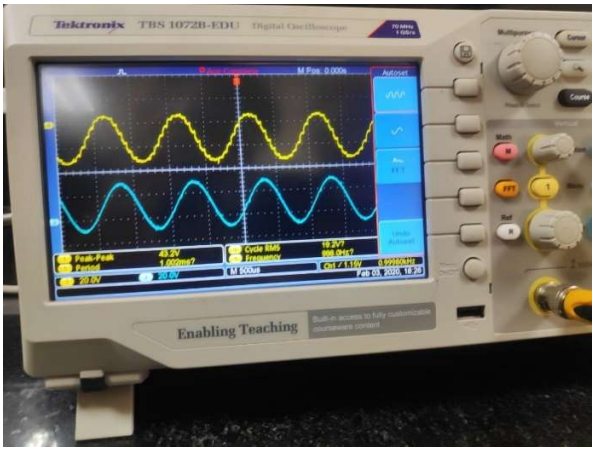


Figure 9: Sampling Frequency: 16KHz, Pulse Amplitude Modulation Sample and Hold Output Reconstructed

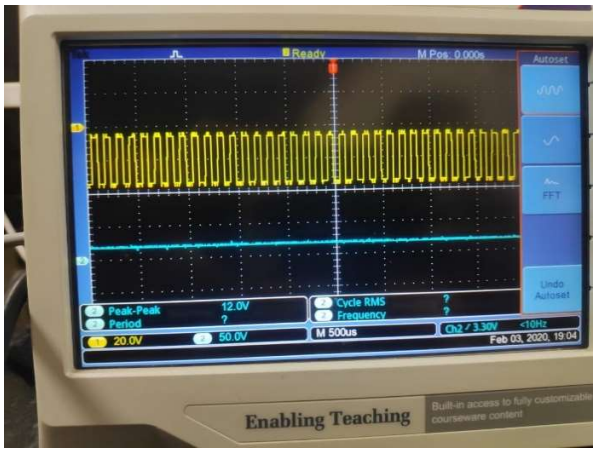


Figure 10: Sampling Frequency: 8KHz, Pulse Amplitude Modulation Sample and Hold Output DC Supply

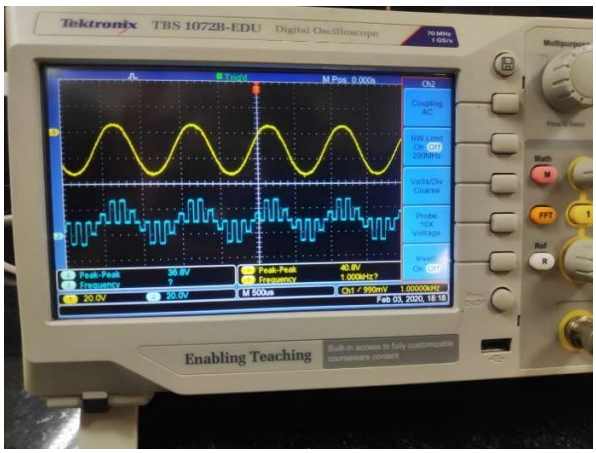


Figure 11: Sampling Frequency: 8KHz, Pulse Amplitude Flat Top Output

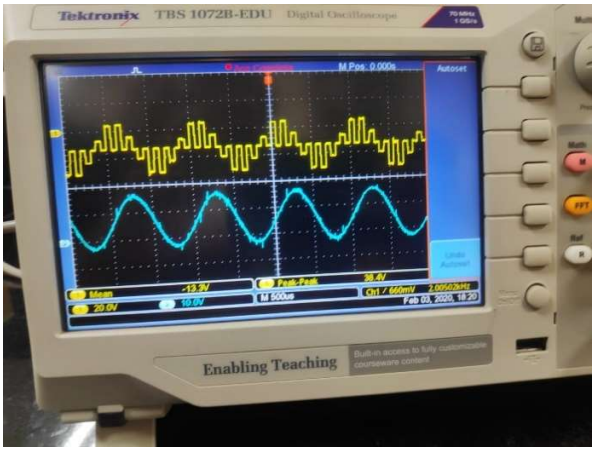


Figure 12: Sampling Frequency: 8KHz, Pulse Amplitude Flat Top Output Reconstructed

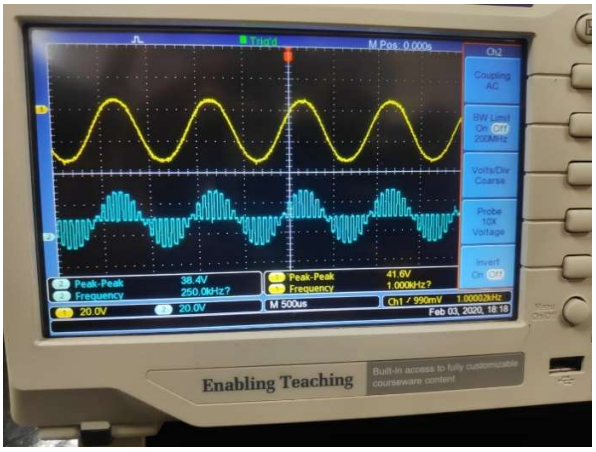


Figure 13: Sampling Frequency: 16KHz, Pulse Amplitude Flat Top Output

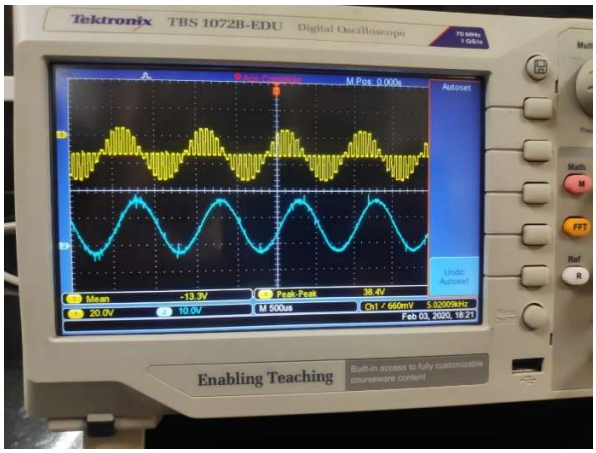


Figure 14: Sampling Frequency: 16KHz, Pulse Amplitude Flat Top Output Reconstructed

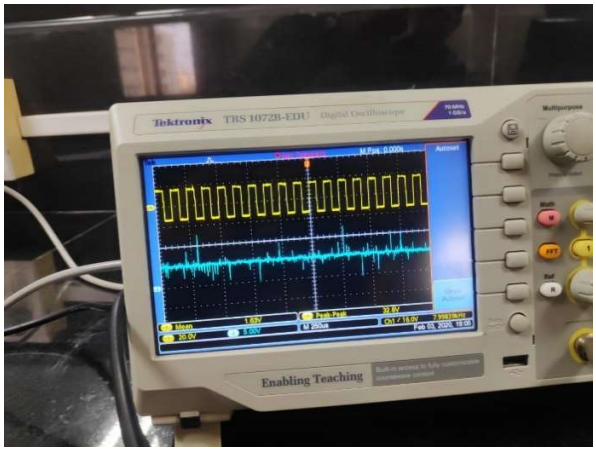


Figure 15: Sampling Frequency: 8KHz, Pulse Amplitude Flat Top Output DC Supply

(The picture shows signal with more noise)

Conclusion:

Pulse Amplitude Modulation has been successfully observed and all the waveforms have been recorded as shown in the above pictures.

The amplitude of the pulse was significantly changed by the Input Signal in case of AC signal.

As we increase the Sampling frequency, the reconstructed output waveform becomes more refined and accurate.

No output waveform was observed when DC Supply was used as Input Signal.

Remarks: **Signature**

EXPERIMENT 4: PULSE POSITION MODULATION/DEMODULATION

Date: 03/02/2020

Aim: To study Pulse Position Modulation/Demodulation

Apparatus: ST2110 PAM trainer kit, DSO, Connecting Leads, Probes

Theory:

Pulse-position modulation (PPM) is a form of signal modulation in which M message bits are encoded by transmitting a single pulse in one of 2^M possible required time shifts.[1][2] This is repeated every **T** seconds, such that the transmitted bit rate is **M/T** bits per second. It is primarily useful for optical communications systems, which tend to have little or no multipath interference.

Procedure:

PART A: PPM using DC input

1. Connect the power supplies of ST2110. Make the connections as shown in the Fig. 1.1 Connect the DC output to input of PPM block .Switch 'ON' the power.
2. Observe the output of PPM block at TP7.
3. Vary the DC output while observing the output of PPM block.
4. Switch ‘On’ the switched faults No. 1, 2, & 6 one by one & observe their effects PPM input and try to locate them.

PART B: PPM using sine wave input

1. Connect the power supplies of ST2110. Make the connections as shown in the Fig. 1.2.
2. Connect the sine wave output of FG block to input of PPM block .Switch 'ON' the power.
3. Keep the oscilloscope at 0.5mS / div, time base speed and in X-5 mode, and observe the pulse position modulated waveform at the pulse position modulation block output at TP 7.
4. Vary the amplitude of sine wave and observe the pulse position modulation, keep the amplitude preset in center. Here you can best observe the pulse modulation.
5. Switch ‘On’ the switched faults No. 1, 2, & 6 one by one & observe their effects PPM input and try to locate them.

Part C: Pulse Position Demodulation:

1. Make the connections as shown in Fig. 1.3. Switch 'On' the power supply & oscilloscope.
2. Observe the waveform at the TP12 output of low pass filter block.
3. Then observe the demodulated output at TP14 output of AC amplifier.

4. Switch 'On' the switched faults No. 1, 2, & 6 one by one & observe their effects PPM input and try to locate them.

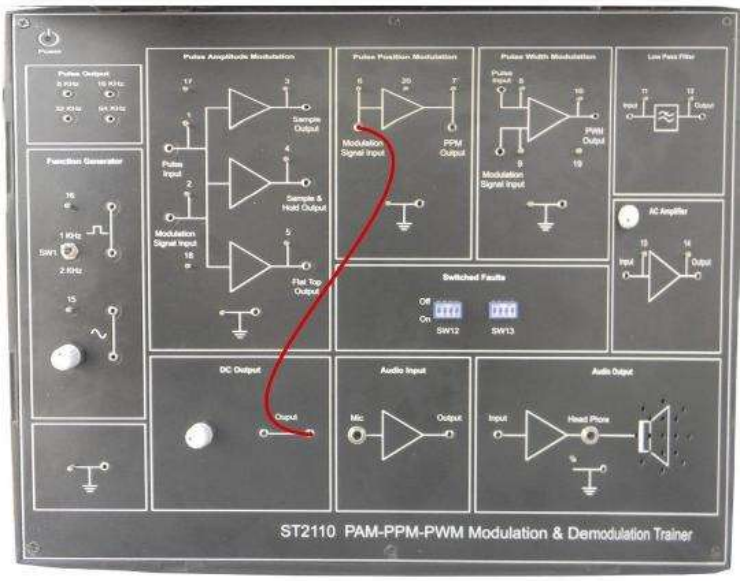


Fig. 1.1 Connection diagram for Pulse Amplitude Modulation and Demodulation

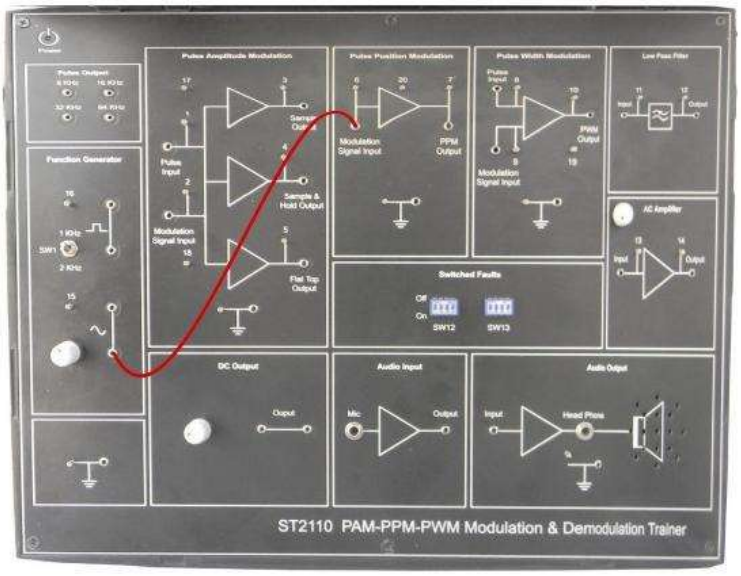


Fig. 1.2 Connection diagram for PPM with Sinusoidal input

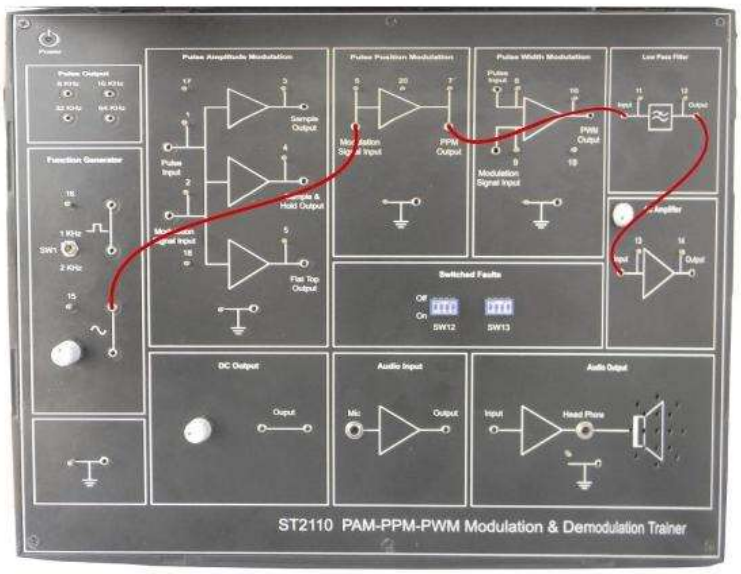
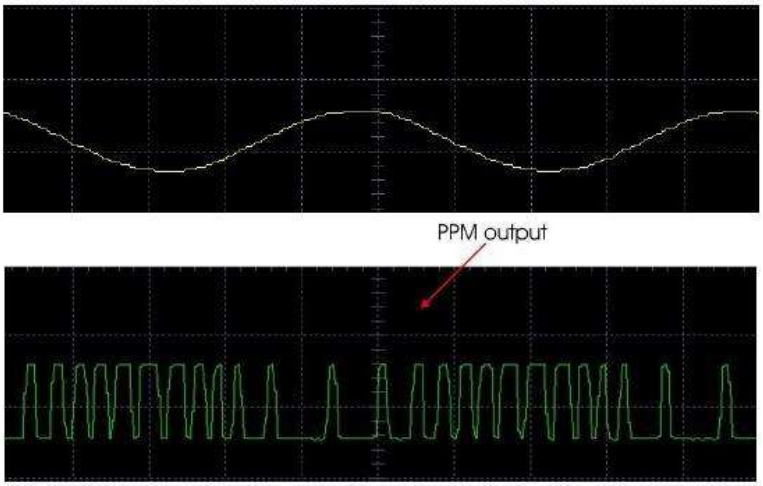


Fig. 1.3 Connection diagram for PPM demodulation

Observation:



Pulse Position Modulation

Observation Table:

Input Signal Frequency	Pulse Frequency	Modulated Signal Frequncy	Demodulated Signal	Remarks
1KHz	8 KHz	8 KHz	0.998 KHz	Output Waveform obtained
1 KHz	16 KHz	16 KHz	1.001 KHz	More Accurate Output Waveform Obtained
DC Supply (0 KHz)	8 KHz	No Significant Output	No Significant Output	No Output Observed

Output Waveforms:

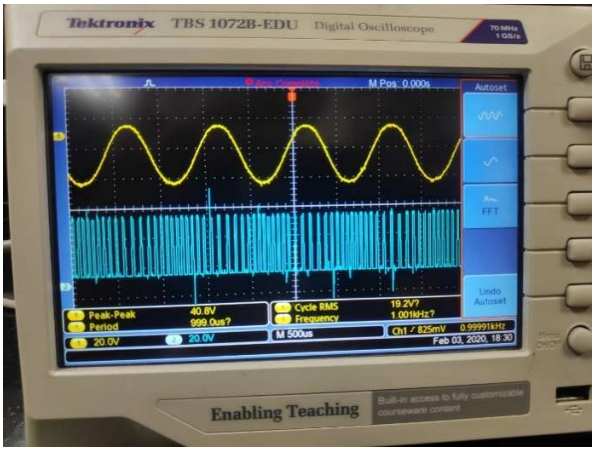


Figure 1: Sampling Frequency: 8KHz, Pulse Position Modulation

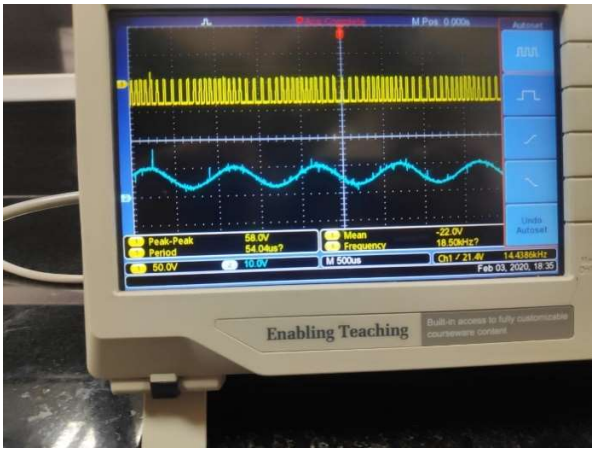


Figure 2: Sampling Frequency: 8KHz, Pulse Position Modulation Reconstructed

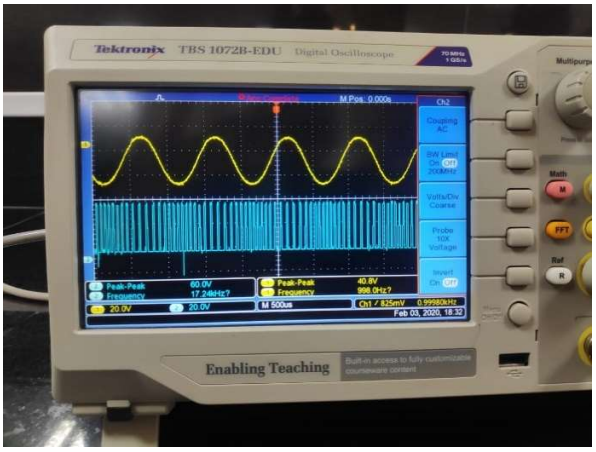


Figure 3: Sampling Frequency: 16KHz, Pulse Position Modulation

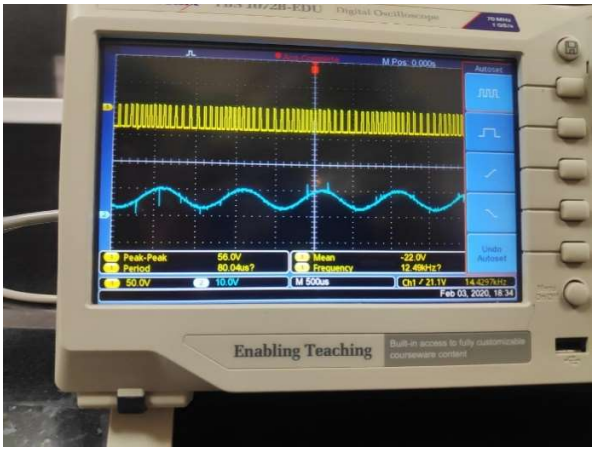


Figure 4: Sampling Frequency: 16KHz, Pulse Amplitude Modulation Sampled Output Reconstructed

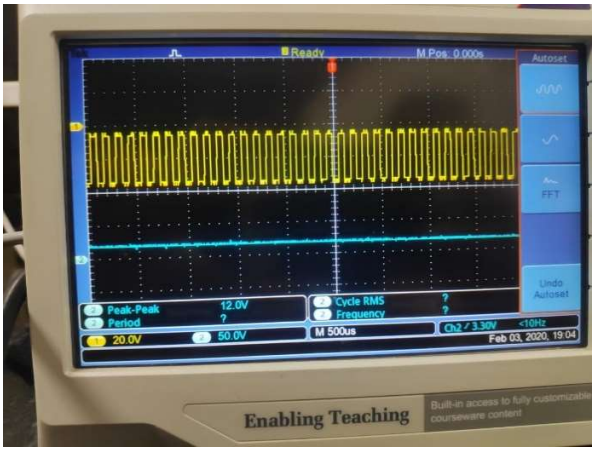


Figure 5 Sampling Frequency: 8KHz, Pulse Position Modulation DC Supply

Conclusion:

Pulse Position Modulation has been carefully observed and output waveforms have been recorded.

The position of the pulse was significantly changed by the Input Signal in case of AC signal.

As we increase the Sampling frequency, the reconstructed output waveform becomes more refined and accurate.

There was no effect on DC Supply by Pulse Position Modulation.

Remarks: **Signature**

EXPERIMENT 5: PULSE WIDTH MODULATION/DEMODULATION

Date: 03/02/2020

Aim: To study Pulse Width Modulation/Demodulation

Apparatus: ST2110 PAM trainer kit, CRO, DSO, Connecting Leads, Probes

Theory:

Pulse width modulation (PWM), or pulse-duration modulation (PDM), is a method of reducing the average power delivered by an electrical signal, by effectively chopping it up into discrete parts.

The average value of voltage (and current) fed to the load is controlled by turning the switch between supply and load on and off at a fast rate. The longer the switch is on compared to the off periods, the higher the total power supplied to the load.

Along with MPPT maximum power point tracking, it is one of the primary methods of reducing the output of solar panels to that which can be utilized by a battery.

PWM is particularly suited for running inertial loads such as motors, which are not as easily affected by this discrete switching, because they have inertia to react slowly.

The PWM switching frequency has to be high enough not to affect the load, which is to say that the resultant waveform perceived by the load must be as smooth as possible.

Procedure:

1. Connect the circuit as shown in Figure 1.1. Switch on the power supply.
2. Observe the output on PWM output block.
3. Vary the amplitude of sine wave and see its effect on pulse output.
4. Vary the sine wave frequency by switching the frequency selector switch to 2 KHz.
5. Also, change the frequency of the pulse by connecting the pulse input to different pulse frequencies viz. 8 KHz, 16 KHz, 32 KHz and see the variations in the PWM output.
6. Switch ‘On’ fault No. 1, 2, & 5 one by one & observes their effect on PWM output and try to locate them.
7. Connect the output of PWM to the low pass filter and the output of the low pass filter to the AC amplifier. Observe the demodulated output.

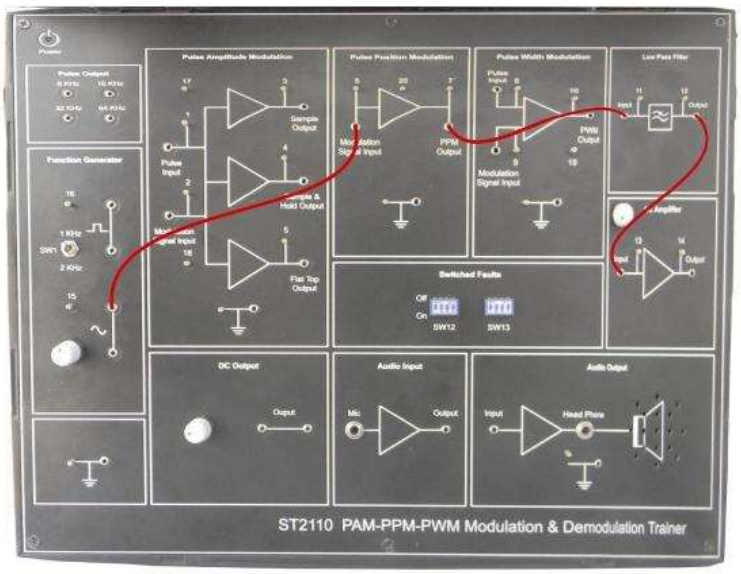
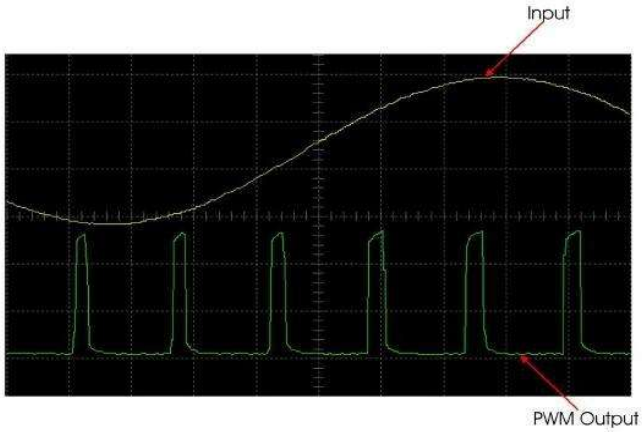


Fig. 1.1 Connection diagram for PWM modulation and demodulation

Observation:



Pulse Width Modulation

Observation Table:

Input Signal Frequency	Pulse Frequency	Modulated Signal Frequncy	Demodulated Signal	Remarks
1KHz	8 KHz	8 KHz	0.98 KHz	Output Waveform obtained
1 KHz	16 KHz	16 KHz	1.0 KHz	More Accurate Output Waveform Obtained
DC Supply (0 KHz)	8 KHz	No Significant Output	No Significant Output	No Output Observed

Output Waveforms:

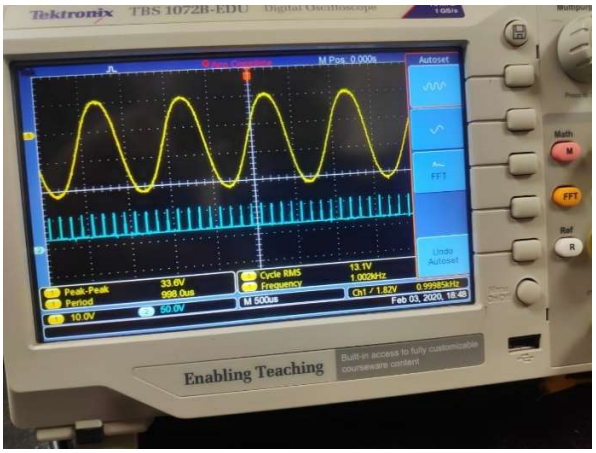


Figure 1: Sampling Frequency: 8KHz, Pulse Position Modulation

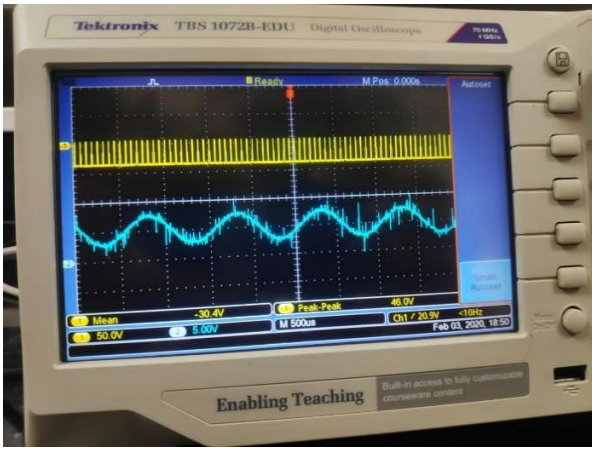


Figure 2: Sampling Frequency: 8KHz, Pulse Width Modulation Reconstructed

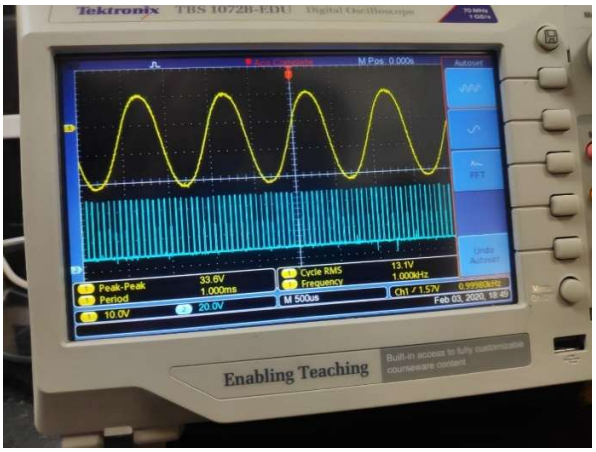


Figure 3: Sampling Frequency: 16KHz, Pulse Width Modulation

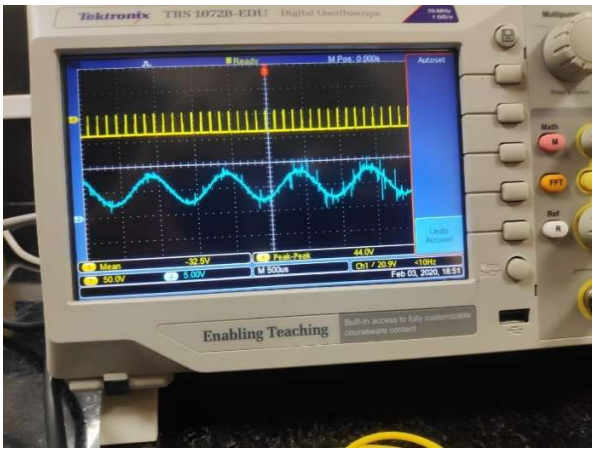


Figure 4: Sampling Frequency: 16KHz, Pulse Width Modulation Sampled Output Reconstructed

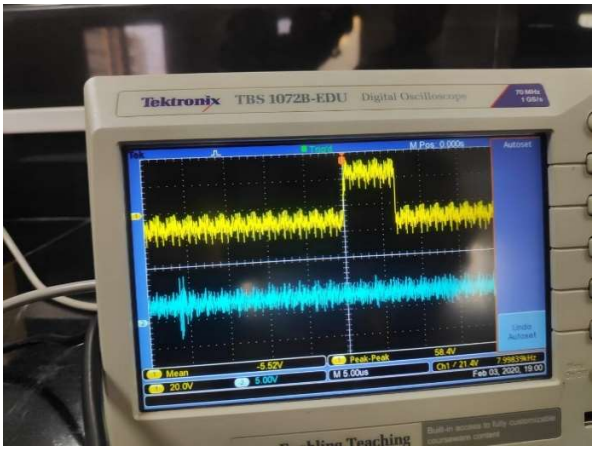


Figure 5 Sampling Frequency: 8KHz, Pulse Width Modulation DC Supply

(The picture shows too much noise)

Conclusion:

Pulse Width Modulation has been employed as Modulation and Demodulation technique on AC as well as DC Signal. Output waveforms are recorded as shown in the picture.

The width of the pulse was modified according to the amplitude of the Input Signal.

As we increase the frequency of the Pulse signal, the output waveform became more accurate.

DC Signal had no effect from Pulse Width Modulation.

Remarks: **Signature**

EXPERIMENT 6: FOURIER SERIES

Date: 10/02/2020

Aim: To compute the Fourier Coefficients for Exponential and Square Wave, and plot their magnitude and phase spectra

Theory/Equations:

In mathematics, a Fourier series is a periodic function composed of harmonically related sinusoids, combined by a weighted summation. With appropriate weights, one cycle (or period) of the summation can be made to approximate an arbitrary function in that interval (or the entire function if it too is periodic). As such, the summation is a synthesis of another function. The discrete-time Fourier transform is an example of Fourier series.

The process of deriving the weights that describe a given function is a form of Fourier analysis.

For functions on unbounded intervals, the analysis and synthesis analogies are Fourier transform and inverse transform.

If **s(x)** is a complex-valued function of a real variable **x**, both components (real and imaginary part) are real-valued functions that can be represented by a Fourier series. The two sets of coefficients and the partial sum are given by:

$$c_{Rn} = \frac{1}{P}\int_P \mathrm{Re}\{s(x)\}\cdot e^{-i\frac{2\pi nx}{P}}\,dx \quad \text{and} \quad c_{In} = \frac{1}{P}\int_P \mathrm{Im}\{s(x)\}\cdot e^{-i\frac{2\pi nx}{P}}\,dx$$

$$s_N(x) = \sum_{n=-N}^{N} c_{Rn}\cdot e^{i\frac{2\pi nx}{P}} + i\cdot\sum_{n=-N}^{N} c_{In}\cdot e^{i\frac{2\pi nx}{P}} = \sum_{n=-N}^{N}\left(c_{Rn}+i\cdot c_{In}\right)\cdot e^{i\frac{2\pi nx}{P}}.$$

Defining $c_n \triangleq c_{Rn} + i\cdot c_{In}$ yields:

$$s_N(x) = \sum_{n=-N}^{N} c_n\cdot e^{i\frac{2\pi nx}{P}}. \quad \text{(Eq.5)}$$

This is identical to **Eq.4** except c_n and c_{-n} are no longer complex conjugates. The formula for c_n is also unchanged:

$$c_n = \frac{1}{P}\int_P \mathrm{Re}\{s(x)\}\cdot e^{-i\frac{2\pi nx}{P}}\,dx + i\cdot\frac{1}{P}\int_P \mathrm{Im}\{s(x)\}\cdot e^{-i\frac{2\pi nx}{P}}\,dx$$

$$= \frac{1}{P}\int_P \left(\mathrm{Re}\{s(x)\}+i\cdot\mathrm{Im}\{s(x)\}\right)\cdot e^{-i\frac{2\pi nx}{P}}\,dx = \frac{1}{P}\int_P s(x)\cdot e^{-i\frac{2\pi nx}{P}}\,dx.$$

Flowchart/Algorithm:

1. Clear console, screen and close windows using the commands clc, clear all, close all.
2. Initialise **t** with values from **-4 π** to **4 π** with step as necessary.
3. Initialise **N** with number of harmonics used as necessary (usually **11** or **100**).
4. Initialise time period **T** with **π** for Exponential Wave or **2 π** for Square Wave case.
5. Initialise Angular Frequency **w** using **T** and **t**.
6. Initialise Exponential Function using exp(t).
7. Calculate the Fourier coefficients **D** (or **C** as you may note it) using Integral function in MATLAB using the above formula.
8. Now plot a graph of absolute of **D** (Magnitude of **D**) against Angular Frequency **w** and its phase(Angle(**D**)) against Angular Frequency **w**.
9. Now find the actual Fourier Series by using Fourier Coefficients and the equation in the Blue box above.
10. Plot the Fourier Series against variable **t**.
11. You have obtained Plots for Magnitude, Phase of Fourier Coefficients and Fourier Series against time.
12. Now repeat the above process for the square wave. But use the time period of Square wave as **2 π**.
13. Record the graphs accordingly.

Code:

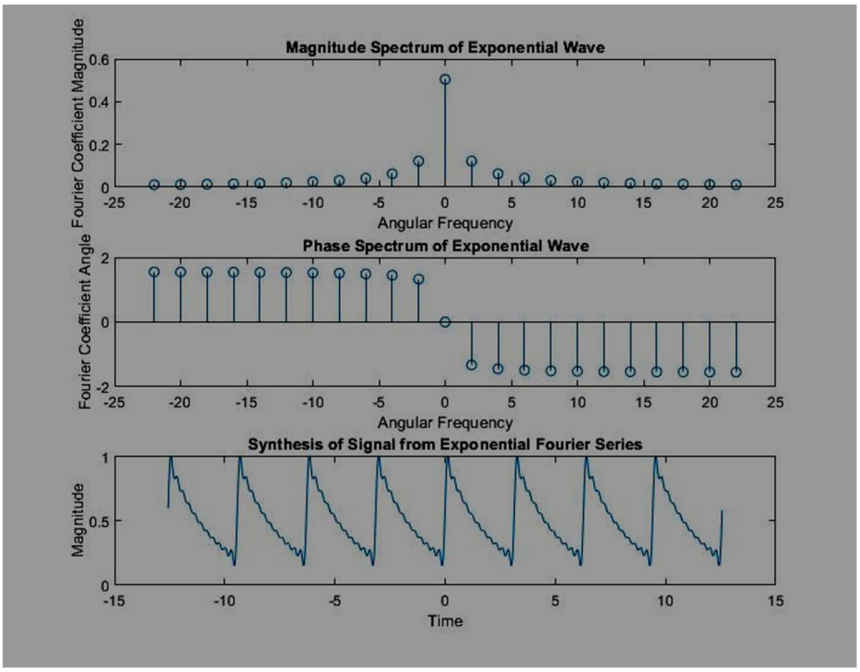
```
%% Exponential Wave

clc;

clear all;

close all;

T = pi;

N = 100;

t = -4*pi:0.01:4*pi;

w = (-N:N)*2*pi/T;

D = zeros(1,2*N+1);
```

```
for i=1:length(w)
    D(i) =(1/T)*integral(@(t)exp(-t/2).*exp(-1i*w(i)*t),0,T);
end
subplot(3,1,1);
stem(w,abs(D));
xlabel('Angular Frequency');
ylabel('Fourier Coefficient Magnitude');
title('Magnitude Spectrum of Exponential Wave');
subplot(3,1,2);
stem(w,angle(D));
xlabel('Angular Frequency');
ylabel('Fourier Coefficient Angle');
title('Phase Spectrum of Exponential Wave');
subplot(3,1,3);
[m,n] = size(t);
g = zeros(m,n);
for i=1:length(t)
    g(i) = sum(D.*(exp(1i*w*t(i))));
end
plot(t,g);
```

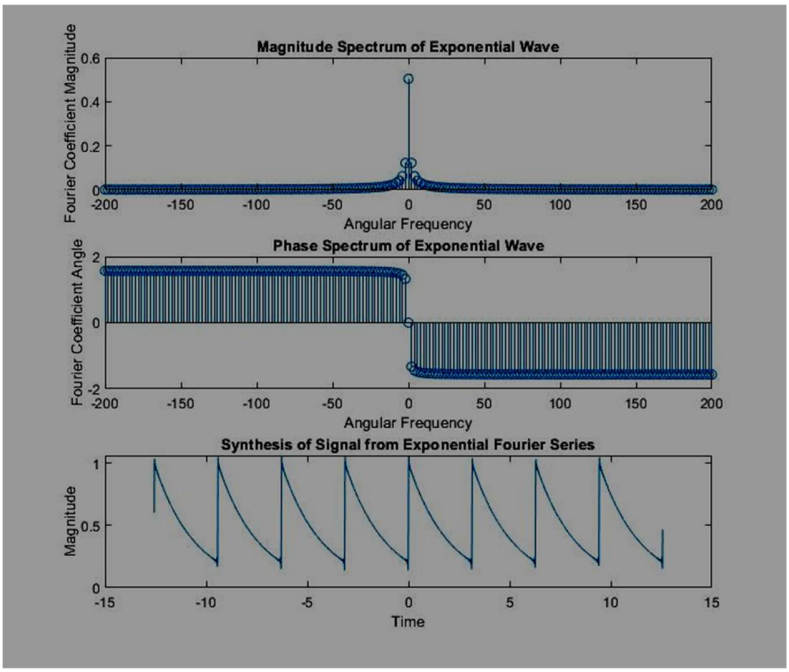
```
xlabel('Time');

ylabel('Magnitude');

title('Synthesis of Signal from Exponential Fourier Series');


%% Square Wave

clc;

clear all;

close all;

T = 2*pi;

N = 11;

t = -4*pi:0.01:4*pi;

w = (-N:N)*2*pi/T;

D = zeros(1,2*N+1);

for i=1:length(w)

    D(i) =(1/T)*integral(@(t)square(t).*exp(-1i*w(i)*t),0,T);

end

subplot(3,1,1);

stem(w,abs(D));

xlabel('Angular Frequency');

ylabel('Fourier Coefficient Magnitude');
```

```
title('Magnitude Spectrum of Square Wave');

subplot(3,1,2);

stem(w,angle(D));

xlabel('Angular Frequency');

ylabel('Fourier Coefficient Angle');

title('Phase Spectrum of Square Wave');

subplot(3,1,3);

[m,n] = size(t);

g = zeros(m,n);

for i=1:length(t)

    g(i) = sum(D.*(exp(1i*w*t(i))));

end

plot(t,g);

xlabel('Time');

ylabel('Magnitude');

title('Synthesis of Signal from Square Fourier Series');
```

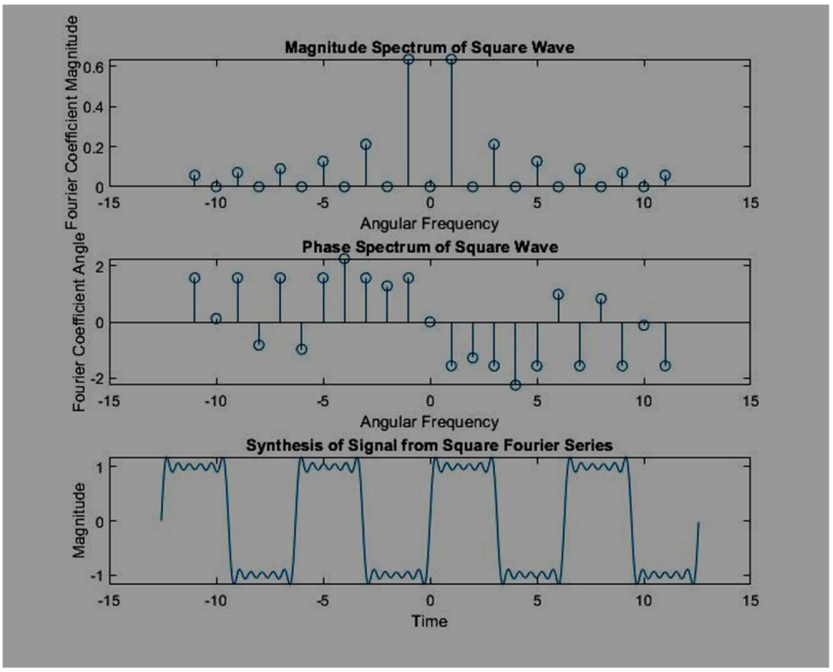
Result/Output Waveforms:



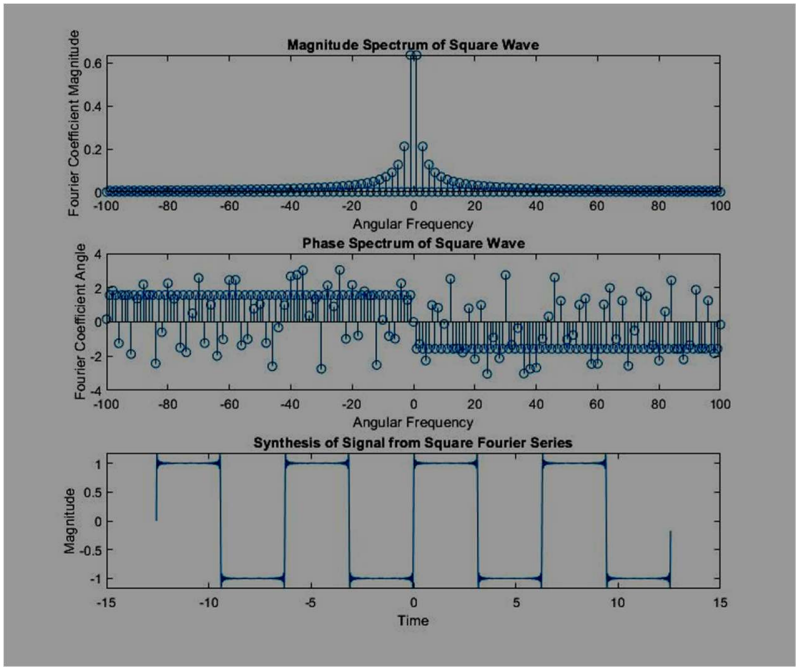
N = 11



N = 100


Magnitude Spectrum of Square Wave
Fourier Coefficient Magnitude
Angular Frequency
Phase Spectrum of Square Wave
Fourier Coefficient Angle
Angular Frequency
Synthesis of Signal from Square Fourier Series
Magnitude
Time

N = 11


Magnitude Spectrum of Square Wave
Fourier Coefficient Magnitude
Angular Frequency
Phase Spectrum of Square Wave
Fourier Coefficient Angle
Angular Frequency
Synthesis of Signal from Square Fourier Series
Magnitude
Time

N = 100

Conclusion:

The Fourier Coefficients for Exponential Wave and Square Wave were derived for N=11 harmonics and N=100 harmonics in MATLAB. Also, their magnitudes and phases were plotted against angular frequency.

Fourier transforms of Sine, Cosine and Tangent waves were also derived which were left as an exercise.

Fourier Transform graph was constructed using above derived Fourier Coefficients.

It was observed that the Fourier Transform curve becomes smoother as we include higher harmonics of sine and cosine waves.

Remarks: **Signature**

EXPERIMENT 7: FREQUENCY MODULATION (FM)

Date: 28/06/2020

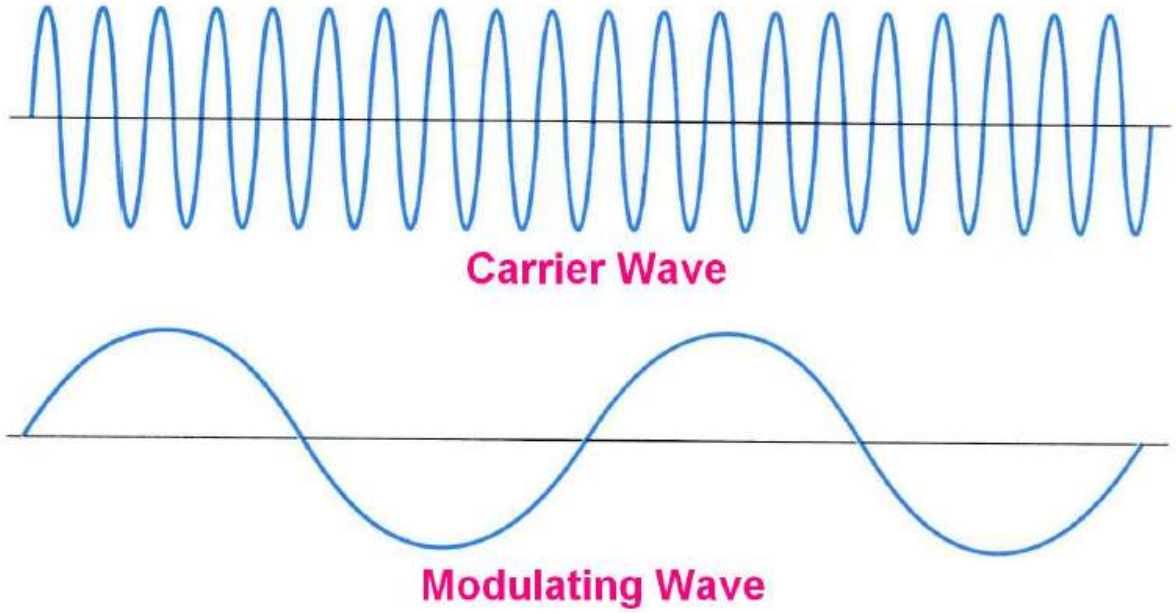
Aim: To study frequency modulation and demodulation and observe the waveforms.

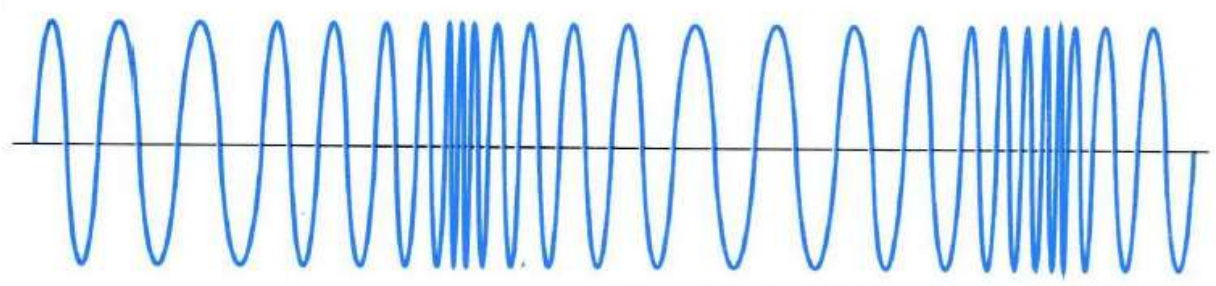
a) **Observe the spectra of FM signal in labAlive virtual communication lab and Calculate the modulation index for FM**
b) **To perform FM transmission via virtual lab labAlive for the audio signal**
c) **To perform FM reception via virtual lab labAlive for the obtained recorded signal**

Apparatus: LabAlive Software

Theory:

Frequency Modulation (FM):
The frequency of the carrier waveform varies with the information signal





Frequency Modulated Wave

Frequency modulation is a system in which the amplitude of the modulated carrier is kept constant, while its frequency is varied by the modulating signal, the modulating signal is sinusoidal. This signal has two important parameters which must be represented by the modulation process without distortion: namely its amplitude and frequency.

If carrier signal, $e_c = E_c \sin \omega_c t$ and modulating signal, $e_m = E_m \sin \omega_m t$ then, the peak or maximum frequency deviation:

$\Delta f \; \alpha \;\; e_m$

$\Delta f = k_f e_m$

Where, k_f is proportionality constant[V/Hz], and e_m is the instantaneous value of the modulating signal amplitude. Thus the frequency of the FM signal is:

$$e_s(t) = e_c + \Delta f = e_c + k_f e_m(t)$$

Then,

$$e_s(t) = e_c + k_f E_m \sin \omega_m t$$

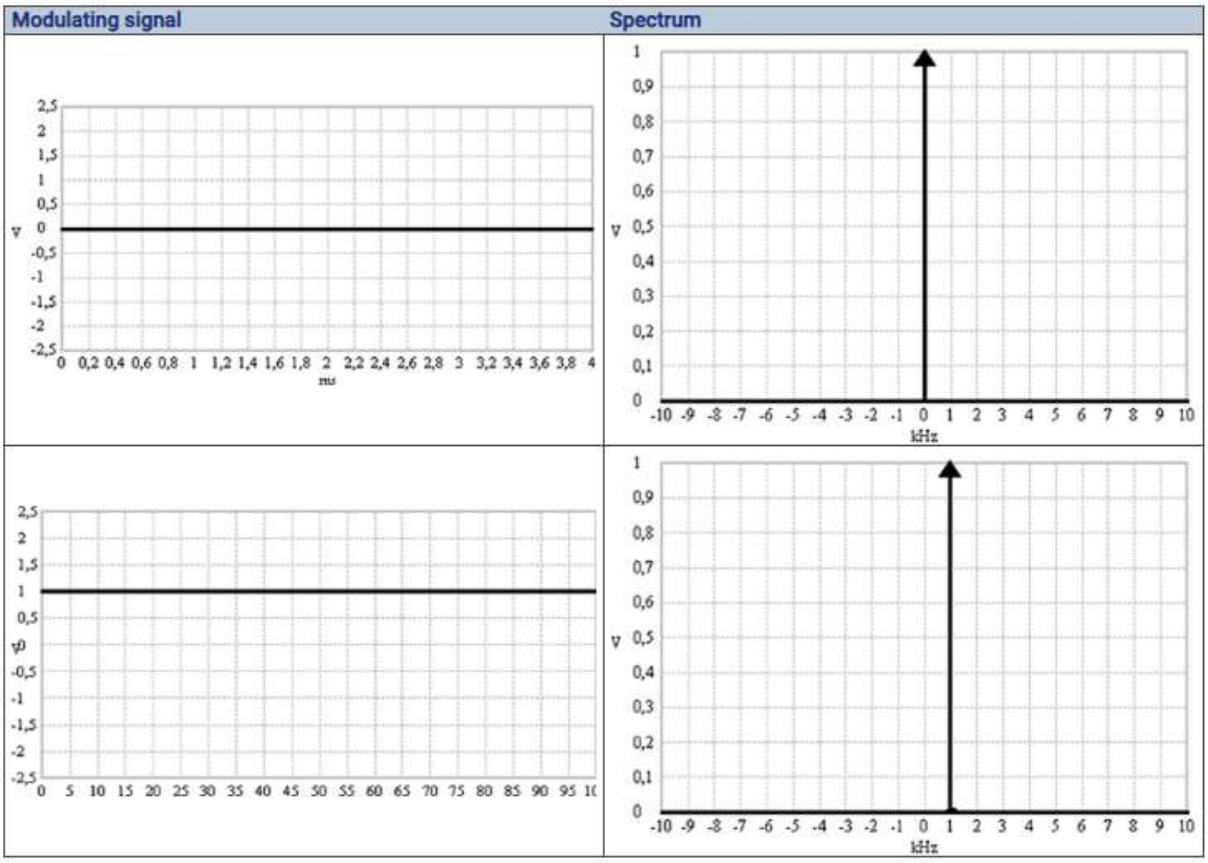
Then the equation for the FM signal is:

$$e_s(t) = E_c \sin(\omega_c t + \beta \sin \omega_m t)$$

Where, β = modulation index, which can be greater than 1. It is measured in radians

β = Freq. Deviation / Modulating Freq.

$$\beta = \frac{\Delta f}{f_m}$$

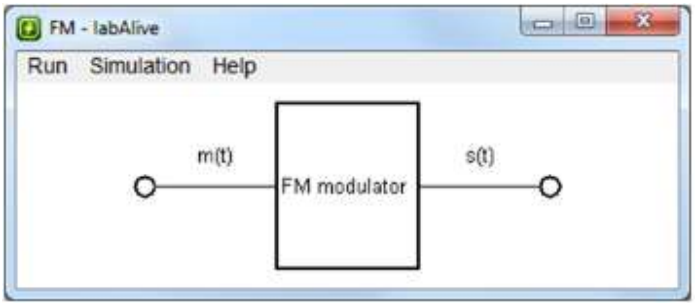


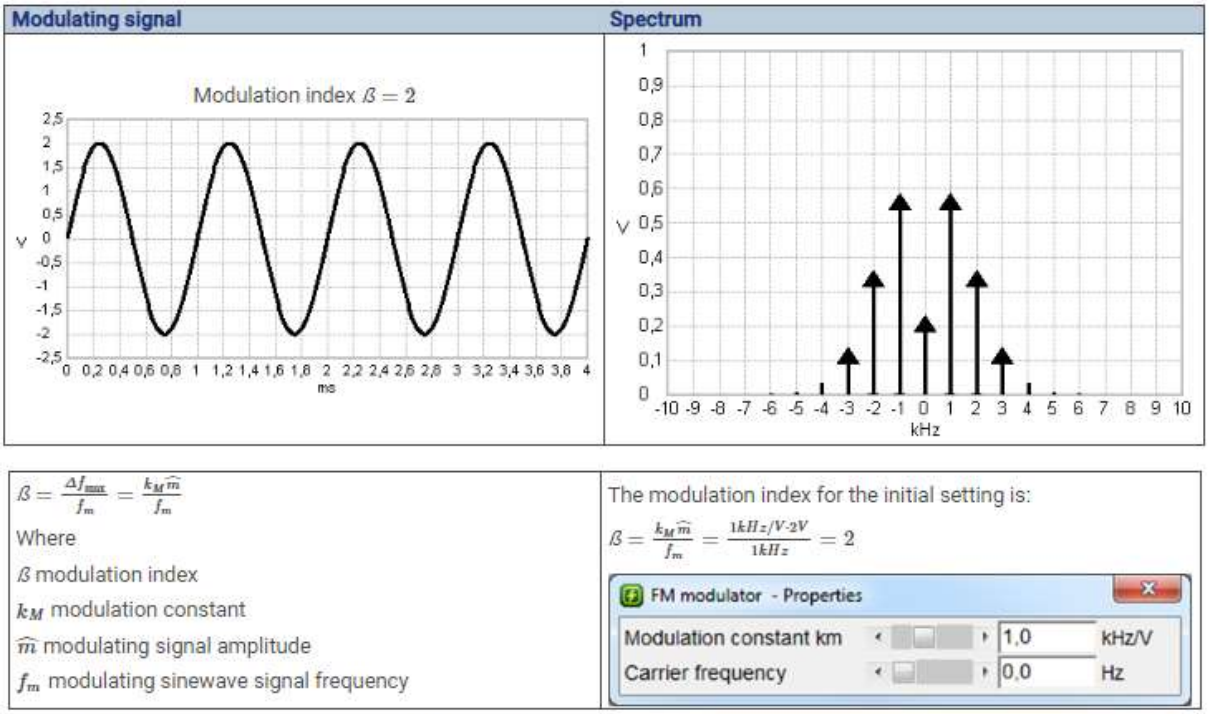
Frequency modulation example - frequency deviation is 1 kHz for a 1V-DC modulating signal

Procedure:

Part a): In this experiment a sinewave signal is frequency modulated. Modulating signal and modulator parameters determine the spectrum of the resulting FM transmission signal.

- On launching the experiment, you will see the following windows:





The modulation index ß is the ratio of the maximum frequency deviation of the carrier to the frequency of the sinewave modulating signal.

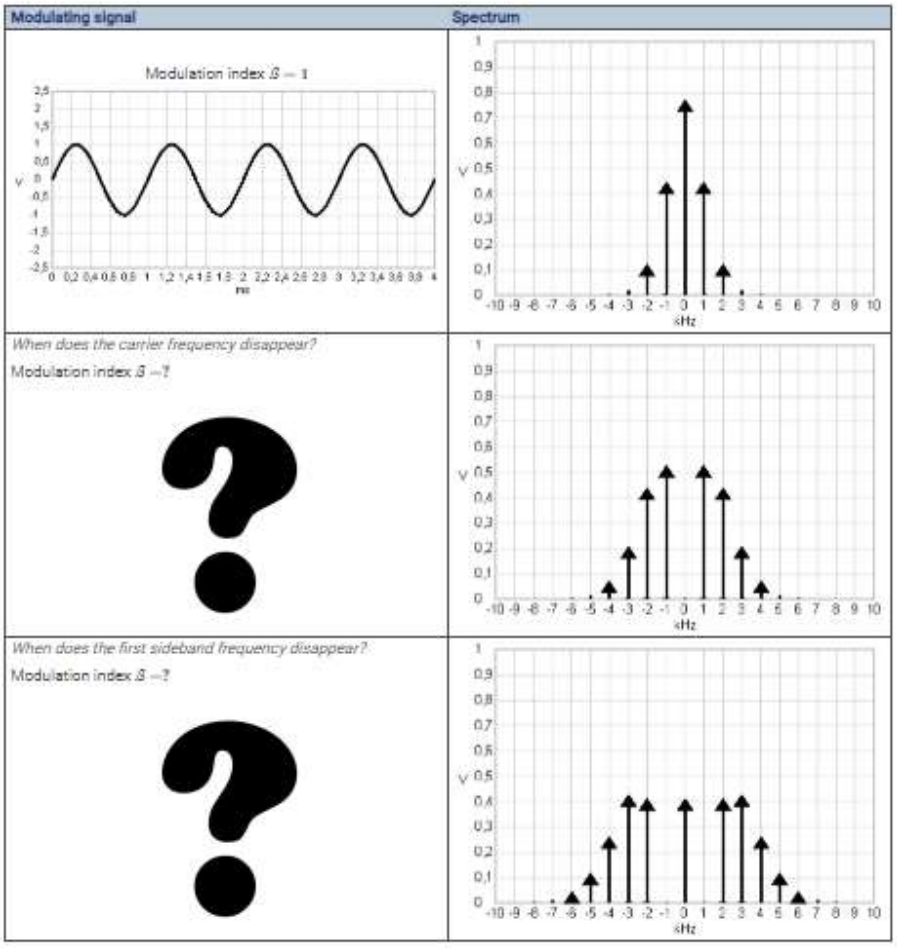
The Bessel function values at the resulting modulation index determine the spectrum of the FM signal.

- Vary the modulating signal amplitude $\hat{m}$.

- The modulation index is proportional to the modulating signal amplitude. In this setting the amplitude in Volts is the modulation index:

$$\beta = \frac{\Delta f}{f_m} = \frac{k_f \hat{m}}{f_m} = \frac{\frac{1kHz}{V} \times \hat{m}}{1kHz} = \frac{\hat{m}}{V}$$

- The adjusted modulating signal amplitude determines the spectral amplitudes of the carrier and sideband frequencies. For some values the carrier or specific sideband frequencies disappear. This relates to zero crossings of the respective Bessel function at the corresponding modulation index.



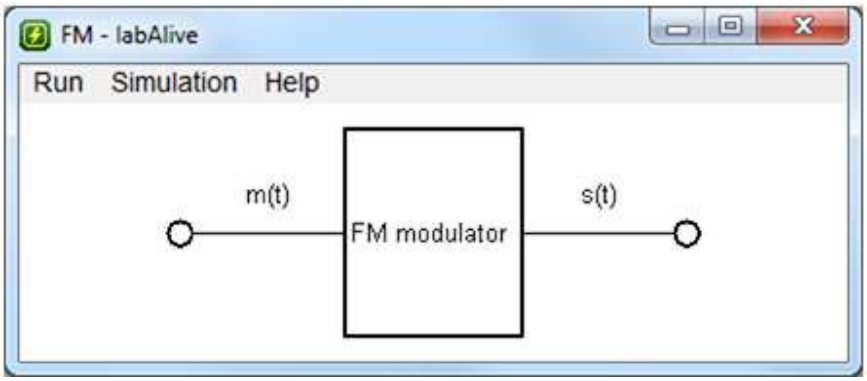
FM signal spectra for sinewave modulation with different modulation indices.

The carrier frequency is 0 Hz in this setting. It might be changed via the modulator properties.

NEXT STEPS

- When do the 2nd and 3rd sideband frequencies disappear?
- Vary the modulating sinewave signal frequency.
- Select different waveforms (signal generator properties) and regard the FM spectrum.
- Use the Bessel functions to determine the spectrum of an FM signal with $\beta = 3$

This simulation implements frequency modulation. The FM signal is generated for the chosen modulating signal. Its spectrum is shown in a spectrum analyzer. All parameters of the modulating signal and modulator can be adjusted.



To change the different settings click on the corresponding wiring:

Adjust parameters of input signal m(t)	Signal Generator - Properties Amplitude 1,0 V Frequency 1,0 kHz Output On Waveform Sine
Adjust parameters of FM modulator Left click on FM modulator: FM modulator	FM modulator - Properties Modulation constant km 1,0 kHz/V Carrier frequency 0,0 Hz
Open measure for transmission signal Right click on s(t): s(t)	Spectrum Analyzer Complex Oscilloscope Power Meter Multimeter Signal Viewer Constellation diagram

Adjust parameters of FM

- Click on the launch tab on the link https://www.etti.unibw.de/labalive/experiment/fm/
- Change the amplitude and frequency of the modulating signal, observe the spectra (attach the output waveforms you observe) and complete the following observation table.
- Note down the frequency deviation from the spectra of FM signal as suggested in the figures below

Reference: Communication Systems by Simon Haykin, 4th edition. Refer Example 2.2 on page 116-117.

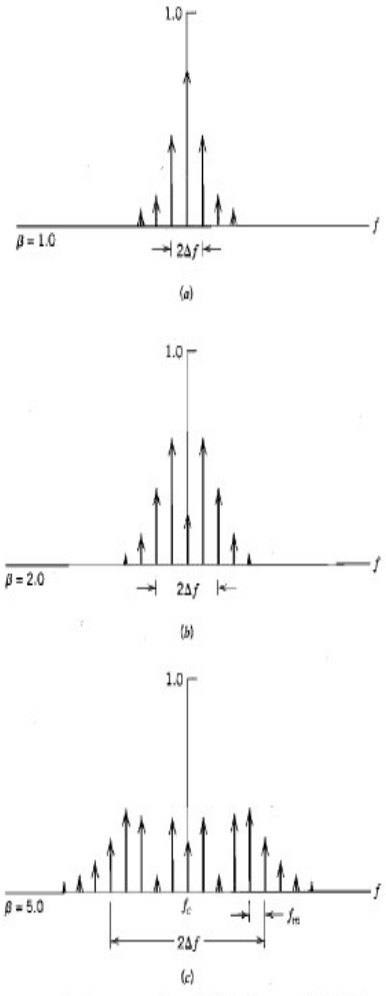


FIGURE 2.24 Discrete amplitude spectra of an FM signal, normalized with respect to the carrier amplitude, for the case of sinusoidal modulation of fixed frequency and varying amplitude. Only the spectra for positive frequencies are shown.

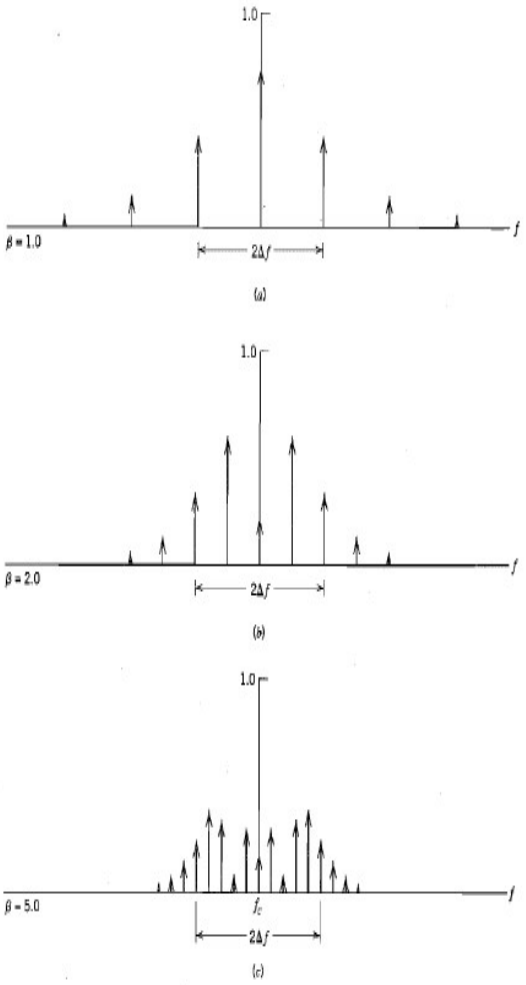


FIGURE 2.25 Discrete amplitude spectra of an FM signal, normalized with respect to the carrie amplitude, for the case of sinusoidal modulation of varying frequency and fixed amplitude. Only the spectra for positive frequencies are shown.

Observation:

Sr. No.	Modulating Signal Frequency, fm	Modulating Signal Amplitude	Frequency Deviation, del(F)	Modulation Index
1	1 KHz	2.4 V	2.4 KHz	2.4
2	1 KHz	3 V	3 KHz	3
3	2 KHz	4.8 V	4.8 KHz	2.4
4	2 KHz	3 V	6 KHz	3
5	3 KHz	7.2 V	7.2 KHz	2.4
6	3 KHz	3 V	9 KHz	3

Output Waveforms:

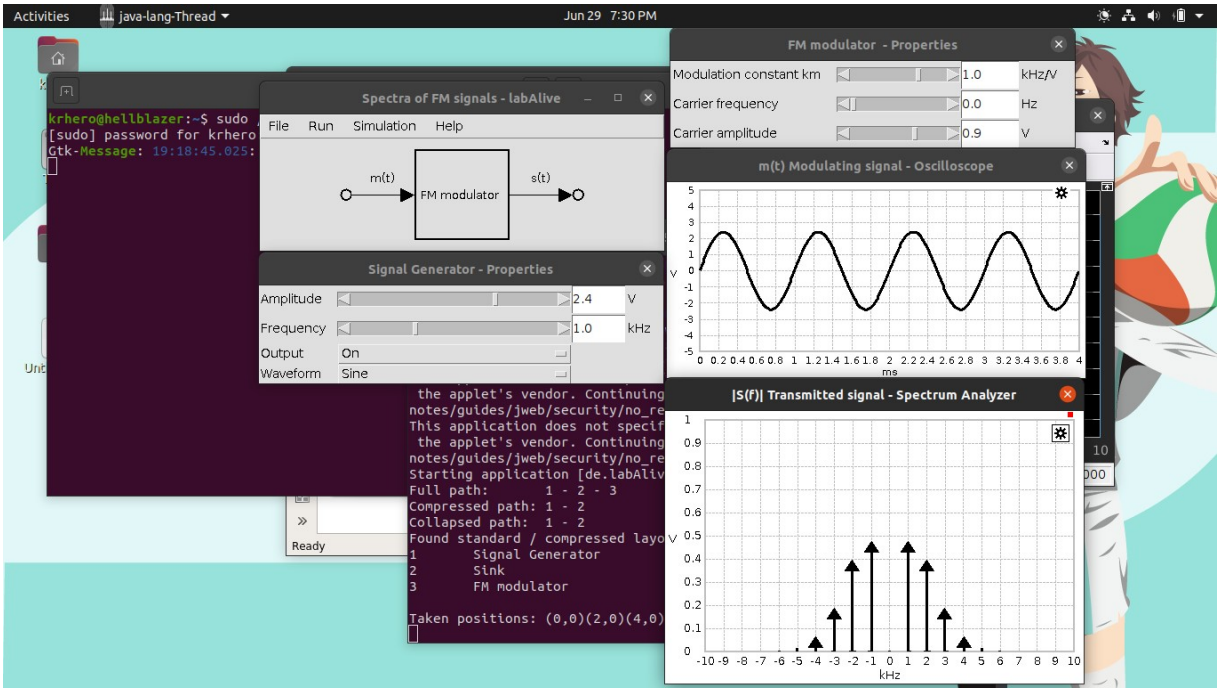


Figure: Amplitude: 2.4 V, Fm = 1 KHz

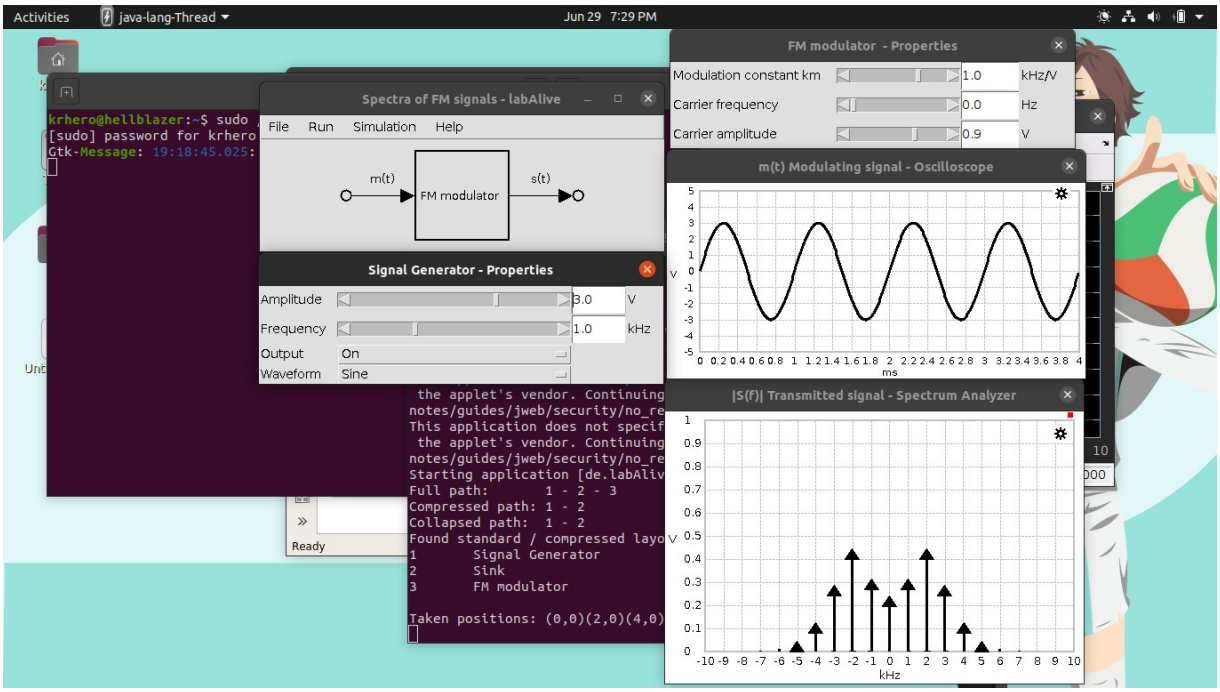


Figure: Beta : 3, Fm = 1 KHz

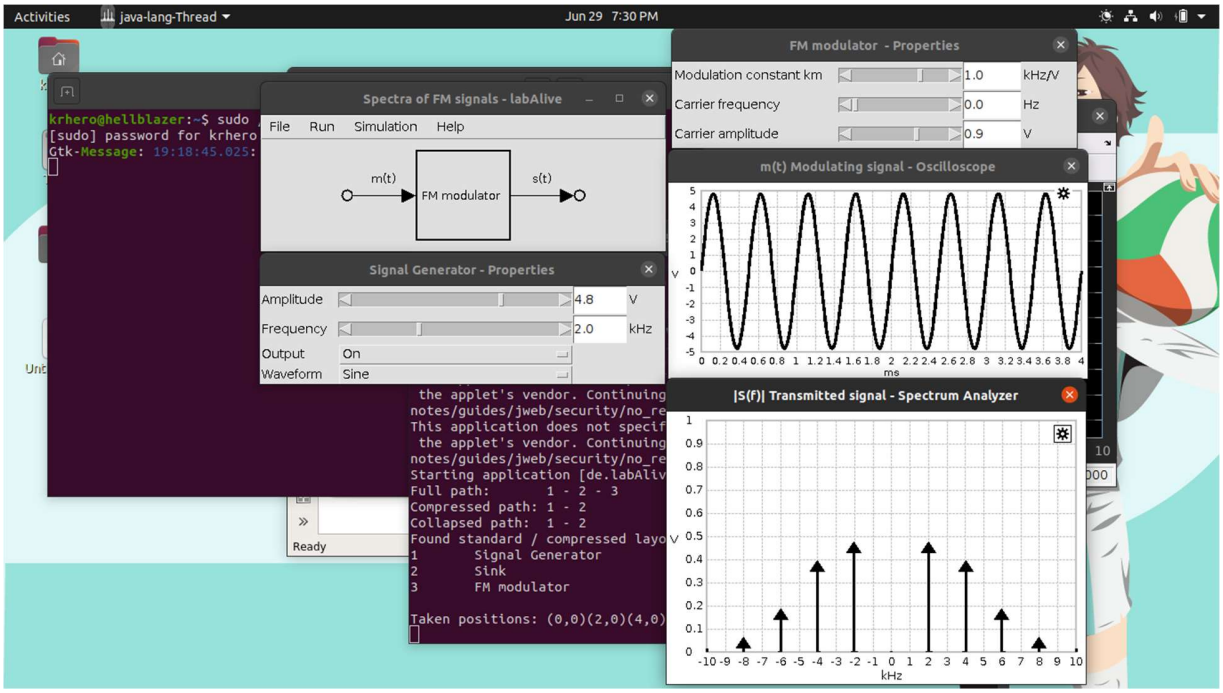


Figure: Amplitude: 4.8 V, Fm: 2 KHz

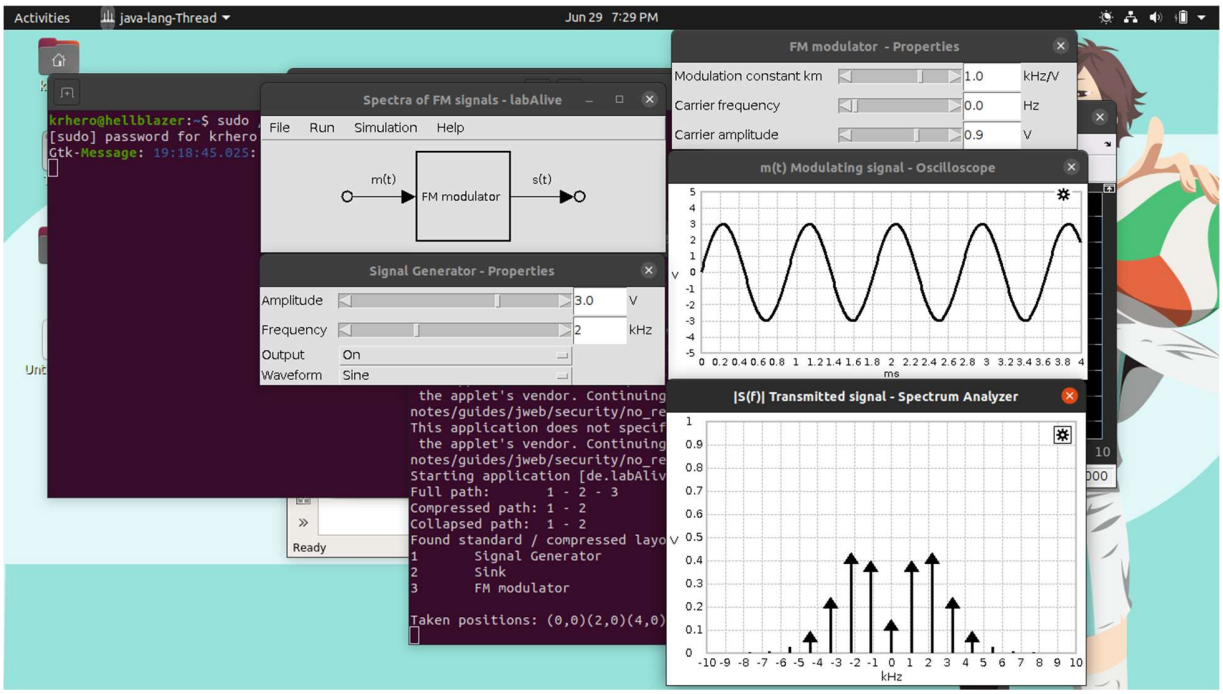


Figure: Beta: 3, Fm: 2 KHz

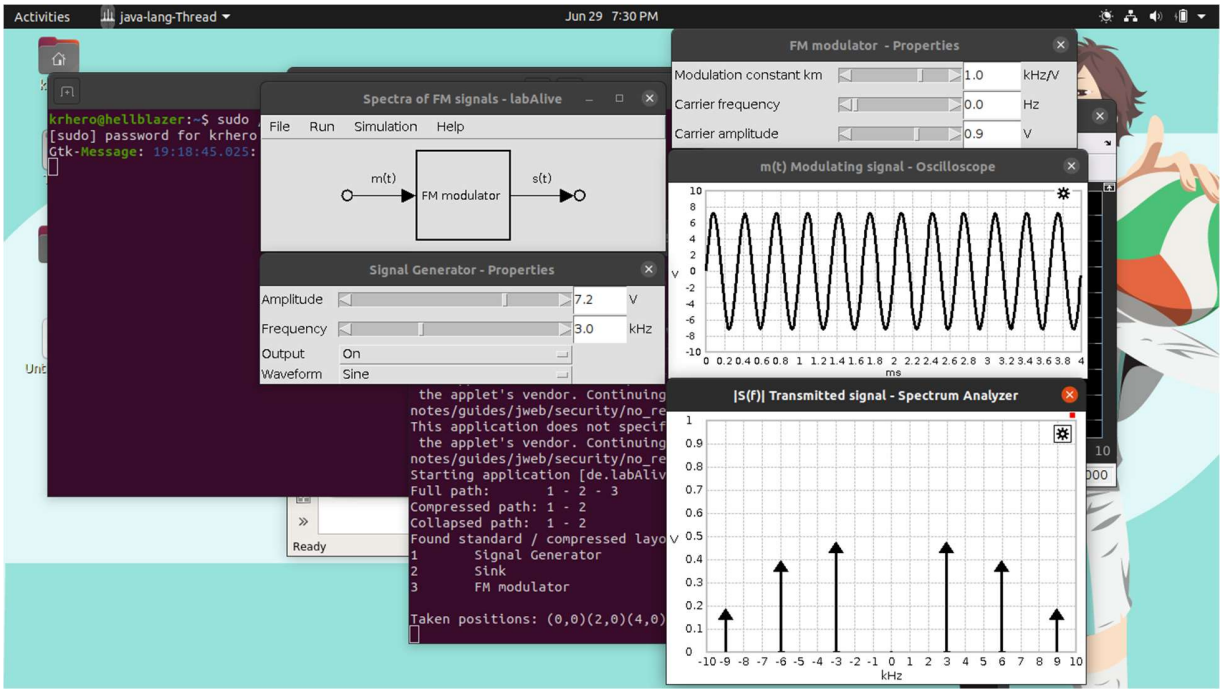


Figure: Amplitude: 7.2 V, Fm: 3 KHz

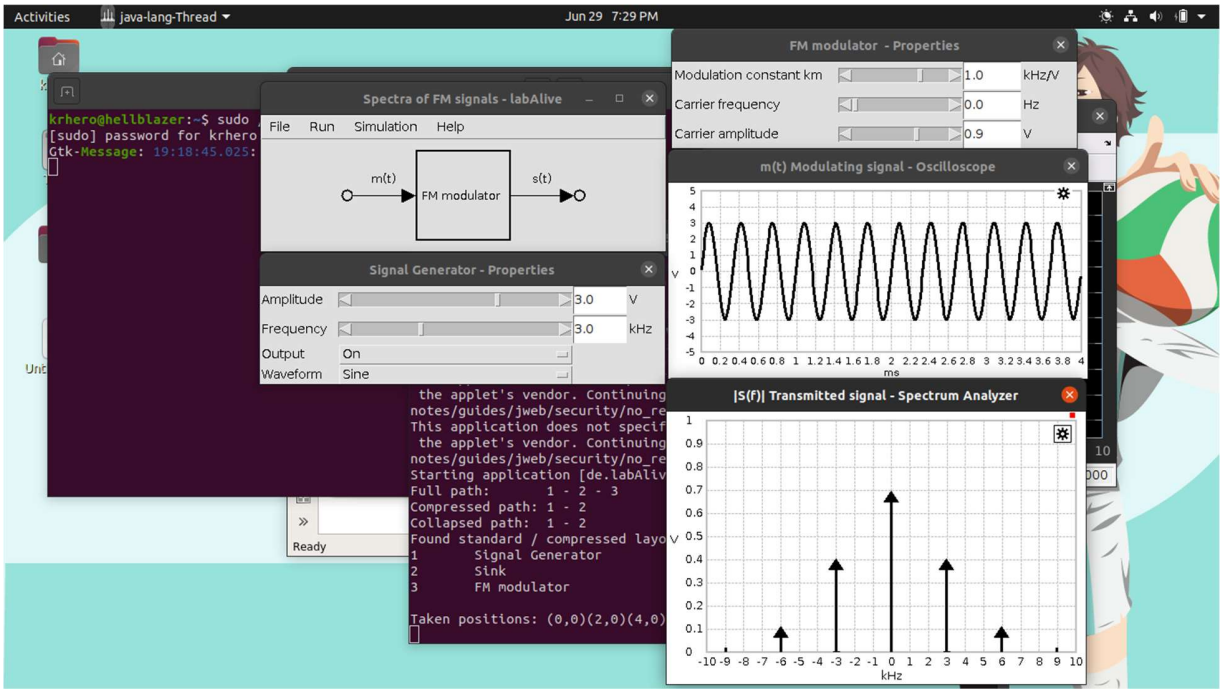
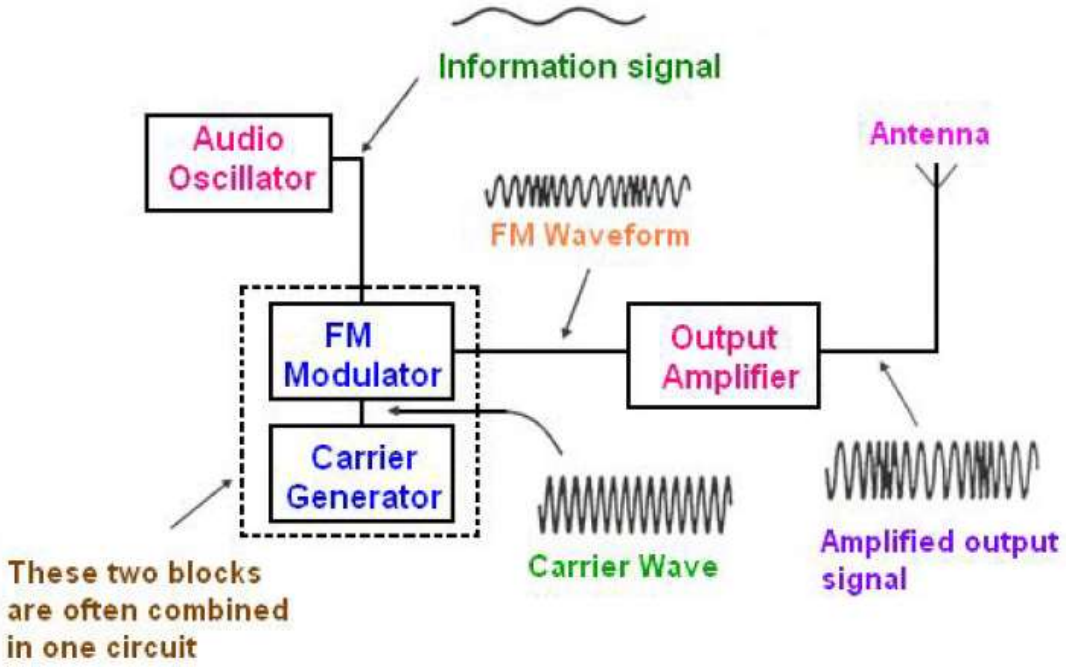


Figure: Beta: 3, Fm: 3 KHz

Part b) To send an audio file via FM transmission link

FM Transmitter:

The block diagram is shown in figure.



Procedure:

START

Initially a music signal provided by the server is frequency modulated. You might select your own audio file in the format 44.1kHz, 16 bit, stereo or the microphone *Line in*.

SET FREQUENCY DEVIATION

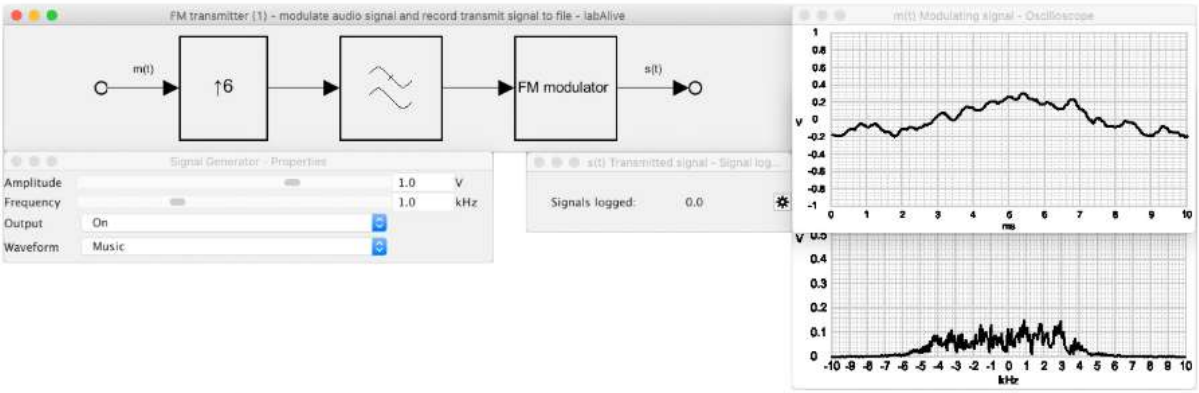
Increase the signal generator's amplitude so that the frequency deviation, i.e. the maximum shift away from the carrier frequency, is 75 kHz. In this base band simulation the carrier frequency is set to 0 Hz.

RECORD THE FM MODULATED SIGNAL TO A FILE

Open the settings of the signal logger - click the gear wheel settings icon. Click *Start save samples to file* to record the transmitted signal to a file.

DEMODULATE THE FM MODULATED SIGNAL

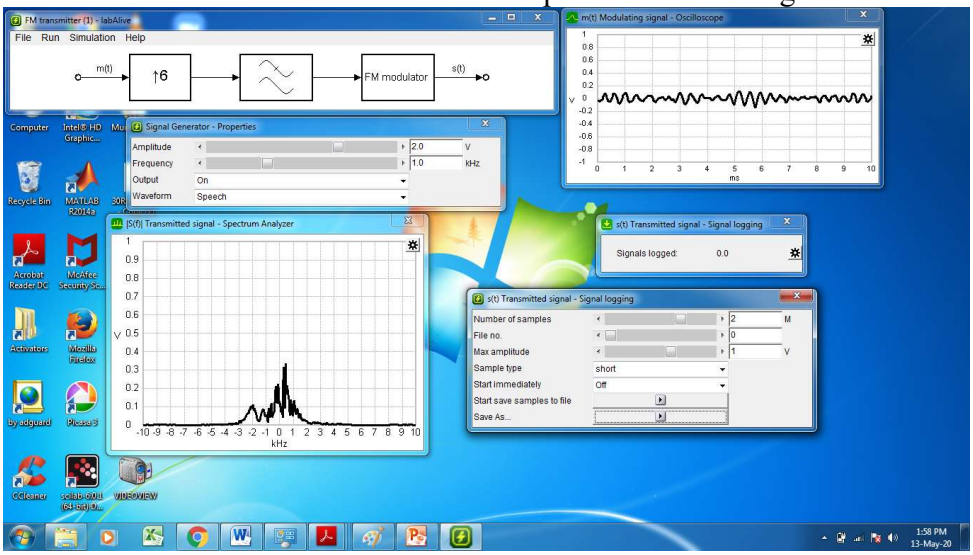
Go the the FM receiver and demodulate the FM transmitted signal.



FM transmitter - modulate an audio signal and record the transmit signal to file.

Notes:

- You may modify the input signal from the signal generator properties
- Varying the frequency and amplitude of input signal will change frequency deviation.
- To save the samples, click on the settings icon in signal logging window and save it. You will use this same file for reception.
- You can scale the graph by clicking on setting icon in modulating signal window and frequency spectrum analyser window.
- Attach all the waveforms obtained for speech and music signal.



Output Waveforms:

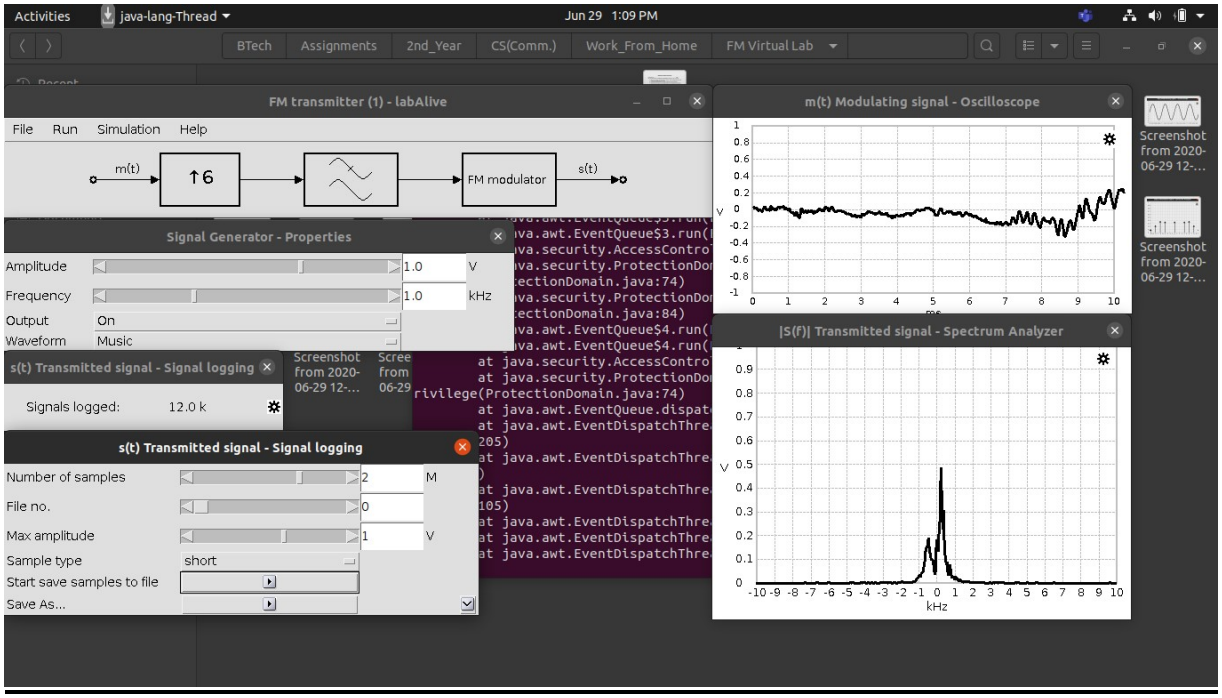
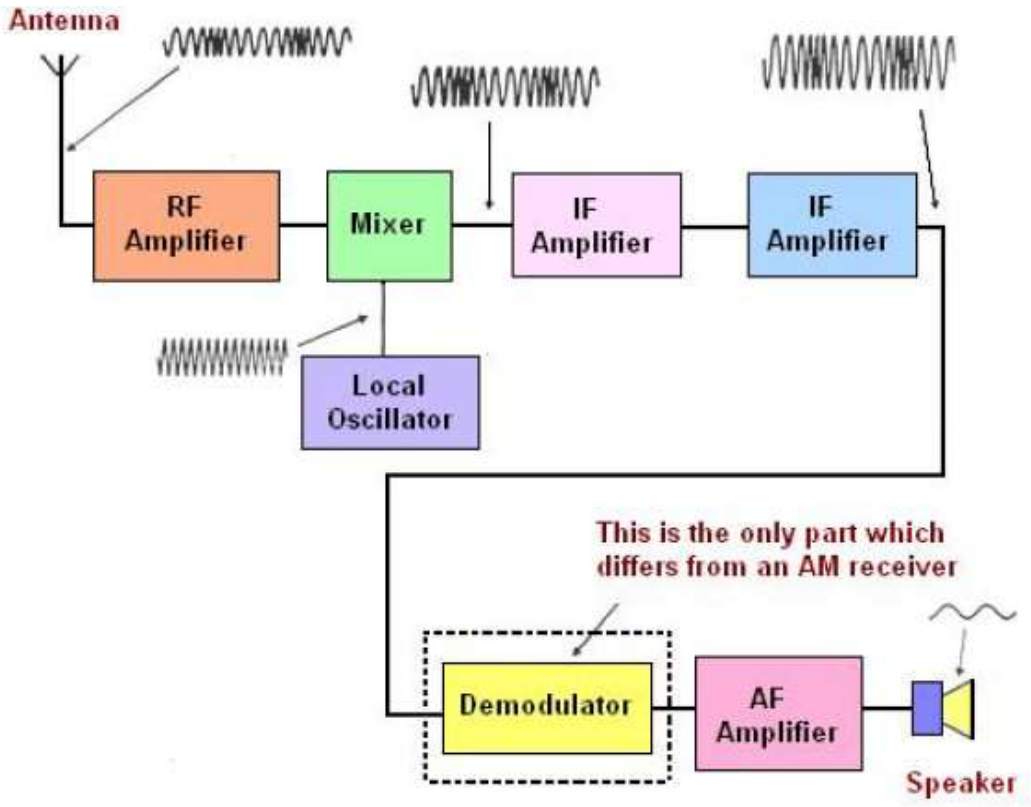


Figure: FM Modulation

Part C): To receive an audio file via FM receiver.

FM receiver:



FM Receiver

Procedure:

START

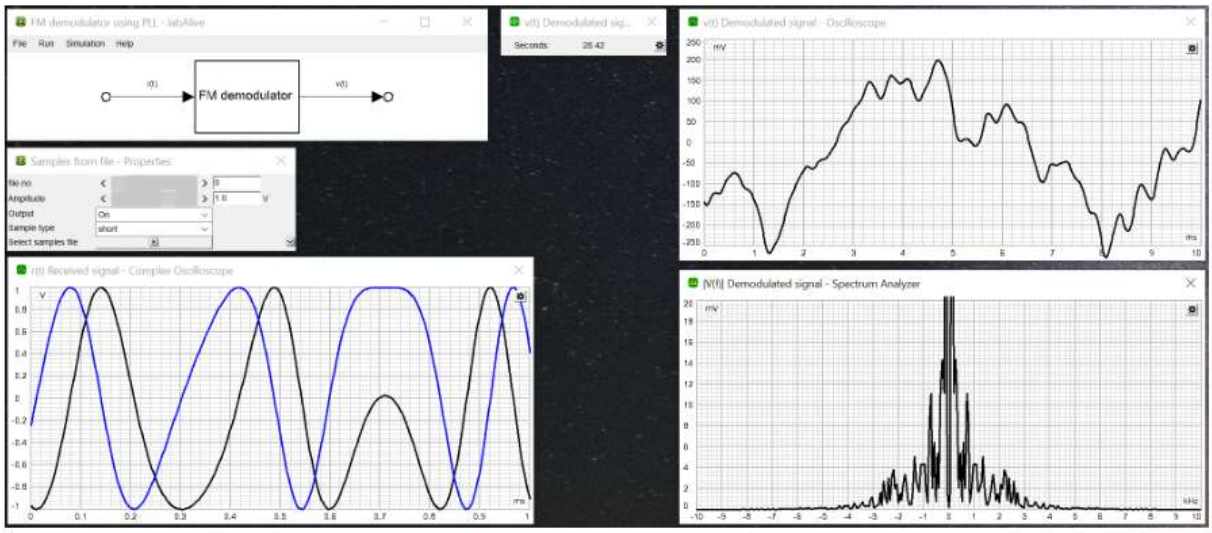
Select the file of the FM modulated signal you created with the FM transmitter.

LISTEN TO THE DEMODULATED SIGNAL

Enjoy the demodulated audio signal. It should be fine if you modulated the audio signal properly. If it's too quiet or distorted analyze if the frequency deviation is too large or too small.

VARY FREQUENCY DEVIATION AND CREATE DIFFERENT FM MODULATED SIGNALS

Vary the modulating signal's amplitude and thus also the frequency deviation using the FM transmitter.



FM receiver using a PLL demodulator.

Notes:

- In the signal logging window, change the sample type to double and number of samples to 1G if you're not able to save the file.
- Listen to received audio signal and change the parameter values in modulator and demodulator block to listen the beat if hissing sound is coming.
- Take the screenshots of the graphs observed and paste it in the output waveforms

Output Waveforms:

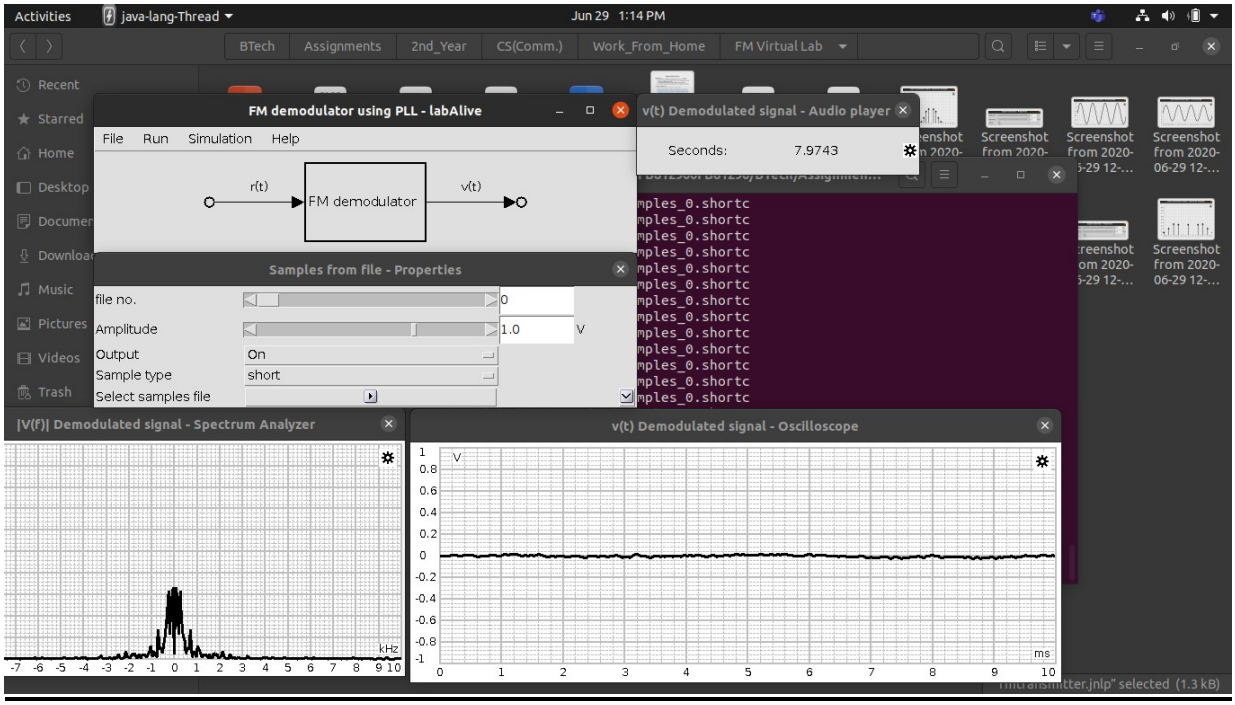


Figure: FM Demodulation

Conclusion:

- FM Modulation and Demodulation has been successfully done using LabAlive software.
- Amplitude and Frequency of input message signal has been varied to generate required frequency spectra.
- Input music audio has been FM modulated and demodulated using given parameters for frequency deviation and base band frequency.

Remarks: Signature

EXPERIMENT 8: ASK, FSK, BPSK Modulation

Date: 28/06/2020

Aim:

a) **To Generate and demodulate an amplitude shift keying(ASK) signal in MATLAB.**
b) **To study Frequency Shift Keying (FSK) Modulation in MATLAB Simulink.**
c) **To study Binary Phase Shift Keying (BPSK) Modulation in MATLAB Simulink.**

Apparatus: MATLAB Software, MATLAB Simulink

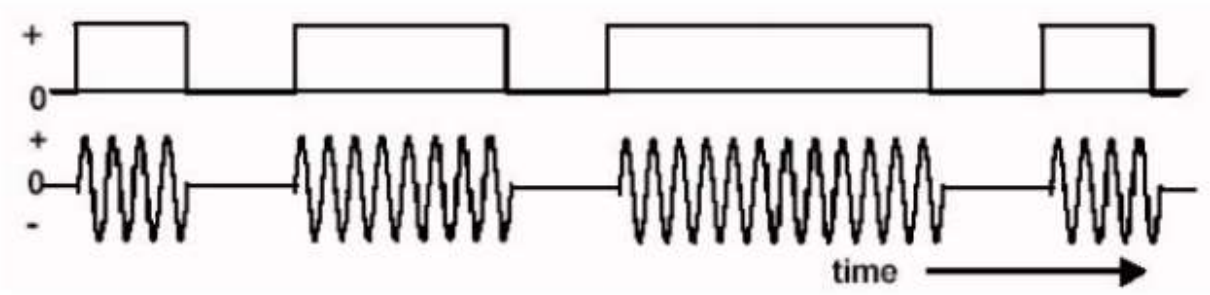
Amplitude Shift Keying(ASK) Modulation

Theory:

Amplitude Shift Keying (ASK) is the digital modulation technique. 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. In ASK, the amplitude of the carrier assumes one of the two amplitudes dependent on the logic states of the input bit stream. This modulated signal can be expressed as:

$$x_c(t) = \begin{cases} 0 & \text{symbol "0"} \\ A\cos \omega_c t & \text{symbol "1"} \end{cases}$$

Amplitude shift keying (ASK) in the context of digital signal communications is a modulation process, which imparts to a sinusoid two or more discrete amplitude levels. These are related to the number of levels adopted by the digital message. For a binary message sequence there are two levels, one of which is typically zero. Thus the modulated waveform consists of bursts of a sinusoid. Figure 1 illustrates a binary ASK signal (lower), together with the binary sequence which initiated it (upper). Neither signal has been band limited.

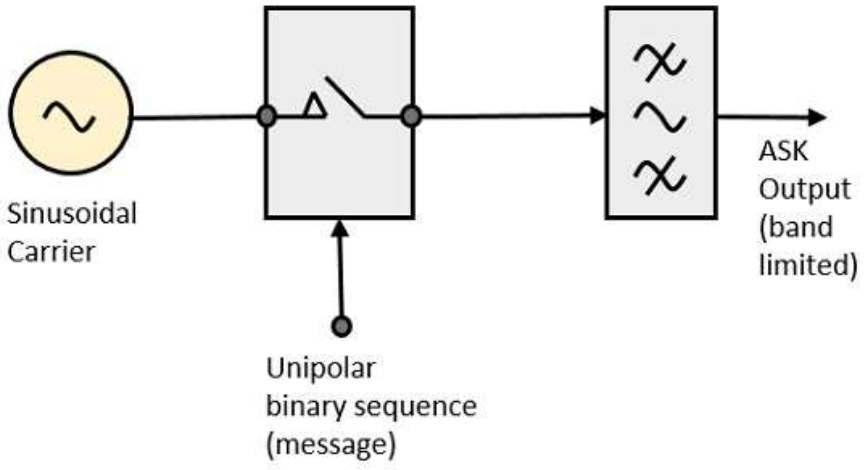


Message Signal & ASK Signal

There are sharp discontinuities shown at the transition points. These result in the signal having an unnecessarily wide bandwidth. Band limiting is generally introduced before transmission, in which case these discontinuities would be 'rounded off'. The band limiting may be applied to the digital message, or the modulated signal itself. The data rate is often made a sub-multiple of the carrier frequency.

ASK Modulator:

The ASK modulator block diagram comprises of the carrier signal generator, the binary sequence from the message signal and the band-limited filter. Following is the block diagram of the ASK Modulator.



The carrier generator, sends a continuous high-frequency carrier. The binary sequence from the message signal makes the unipolar input to be either High or Low. The high signal closes the switch, allowing a carrier wave. Hence, the output will be the carrier signal at high input. When there is low input, the switch opens, allowing no voltage to appear. Hence, the output will be low.

The band-limiting filter shapes the pulse depending upon the amplitude and phase characteristics of the band-limiting filter or the pulse-shaping filter.

ASK Demodulator:

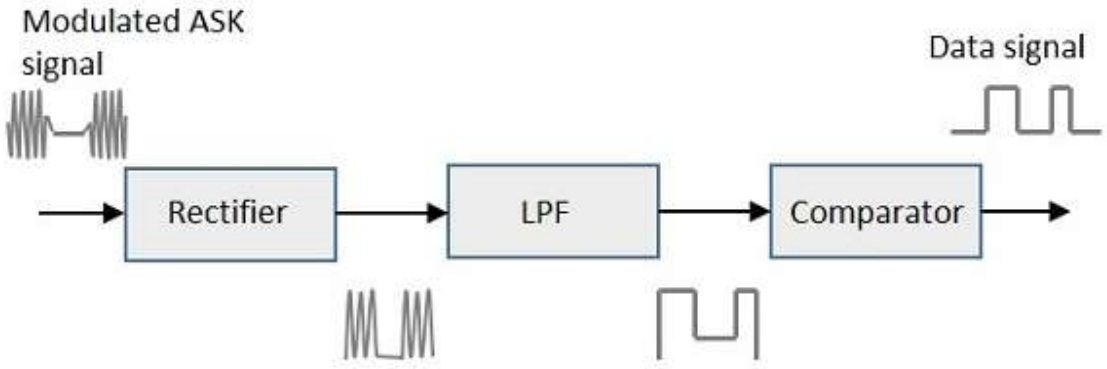
There are two types of ASK Demodulation techniques.

- Asynchronous ASK Demodulation/detection
- Synchronous ASK Demodulation/detection

The clock frequency at the transmitter, when matches with the clock frequency at the receiver, it is known as a Synchronous method, as the frequency gets synchronized. Otherwise, it is known as Asynchronous.

Asynchronous ASK Demodulator:

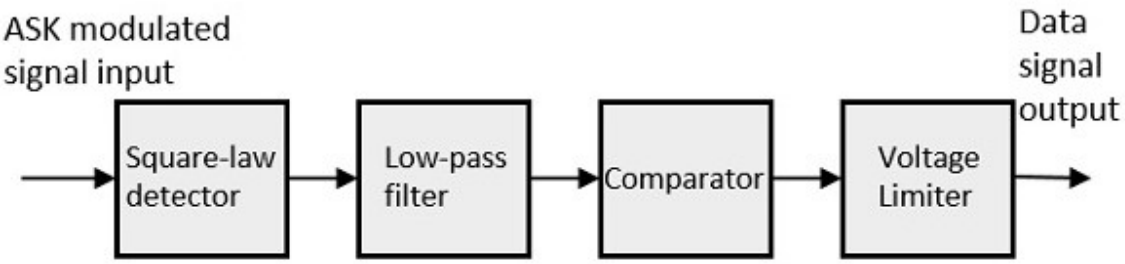
The Asynchronous ASK detector consists of a half-wave rectifier, a low pass filter, and a comparator. Following is the block diagram for the same.



The modulated ASK signal is given to the half-wave rectifier, which delivers a positive half output. The low pass filter suppresses the higher frequencies and gives an envelope detected output from which the comparator delivers a digital output.

Synchronous ASK Demodulator:

Synchronous ASK detector consists of a Square law detector, low pass filter, a comparator, and a voltage limiter. Following is the block diagram for the same.



The ASK modulated input signal is given to the Square law detector. A square law detector is one whose output voltage is proportional to the square of the amplitude modulated input voltage. The low pass filter minimizes the higher frequencies. The comparator and the voltage limiter help to get a clean digital output.

Algorithm to implement ASK modulation & demodulation on MATLAB:

ASK Modulation:

1. Generate a carrier signal of frequency fc.
2. Start a FOR Loop.
3. Generate a binary sequence, a message signal.
4. Generate ASK modulated signal, which will transmit carrier signal for logic 1 and zero signal for logic 0.
5. Plot message signal and ASK modulated signal.
6. End FOR Loop.
7. Plot binary data and carrier.

ASK Demodulation:

1. Start FOR Loop.
2. Perform correlation of ASK signal with carrier to get decision variable.
3. Make decision to get demodulated binary data. If x > 0, choose '1' else choose '0'.
4. Plot the demodulated binary data.

Code:

```
% Experiment: ASK Modulation and Demodulation

% Author: Krunal Rank

% RollNo. : U18CO081


%% Part 1


clc;

clear all;

close all;


t = 0:0.01:8;

bit_stream =  [1 0 1 1 0 0 0 1];

A_c = 1;

f_c = 1;

w_c = 2 * pi * f_c;

m = zeros(1,length(t));

for i = 1: length(t)

    temp = idivide(t(i),int32(1));

    if temp==8

        temp = temp - 1;
```

```
    end
    m(i) = bit_stream(temp+1);
end

c = A_c*sin(w_c*t);

ask_signal = c.*m;

subplot(3,1,1);
plot(t,m);
ylim([-2 2]);
title('Message Signal');

subplot(3,1,2);
plot(t,c);
ylim([-2 2]);
title('Carrier Signal');

subplot(3,1,3);
plot(t,ask_signal);
```

```
ylim([-2 2]);

title('ASK Signal');


%% Part 2


ask_signal_rectified = abs(ask_signal);


subplot(2,1,1);

plot(t,ask_signal_rectified);

ylim([-2 2]);

title('Rectified ASK Signal');


output_signal = ask_signal_rectified>0;


subplot(2,1,2);

plot(t,output_signal);

ylim([-2 2]);

title('Output Signal');
```

Output Waveforms:

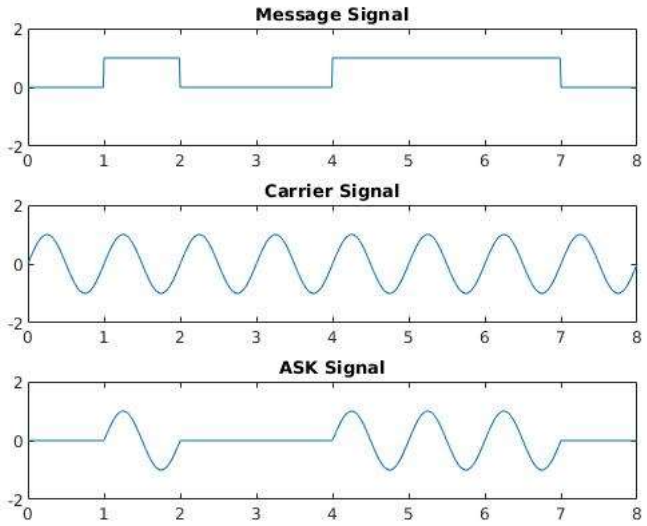


Figure: ASK Modulation using BIT Stream [0 1 0 0 1 1 1 0]

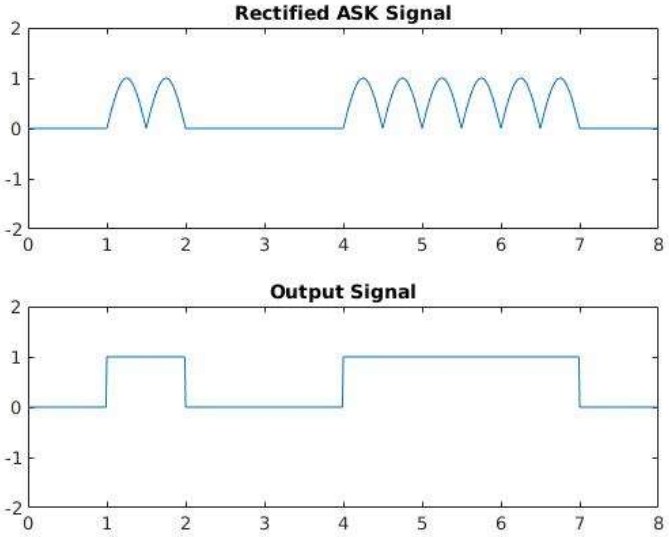


Figure: ASK Demodulation

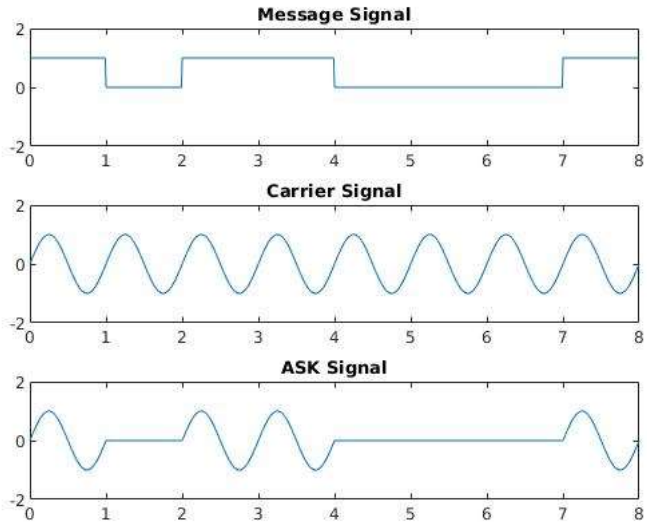


Figure: ASK Modulation using BIT Stream [1 0 1 1 0 0 0 1]

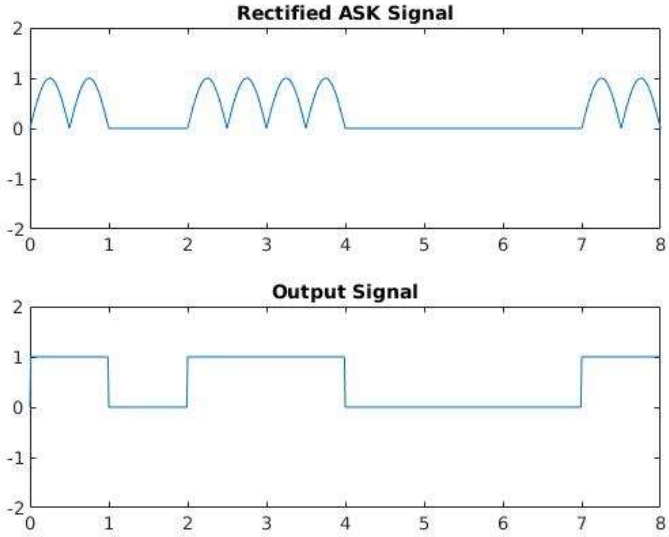


Figure: ASK Demodulation

Frequency Shift Keying (FSK) Modulation

Block Diagram:

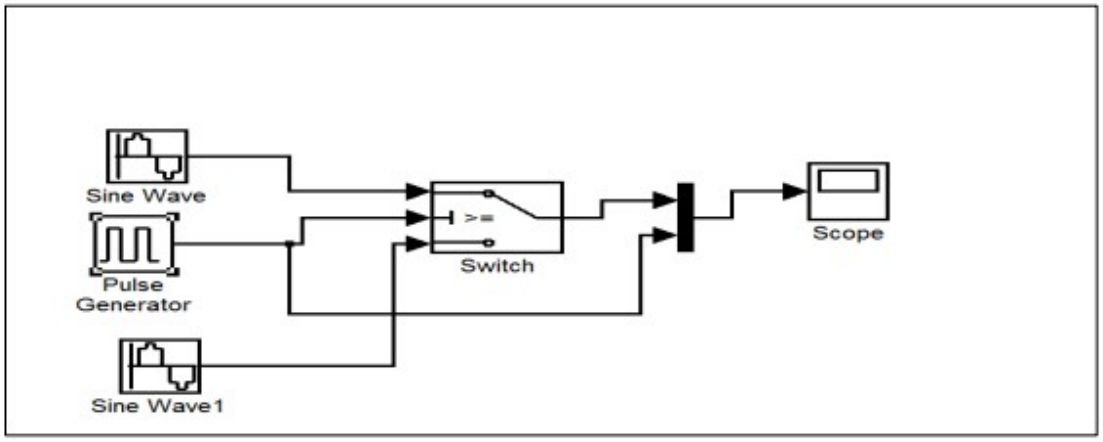


Fig.1: Block Diagram of FSK Modulator in Simulink MATLAB

Theory:

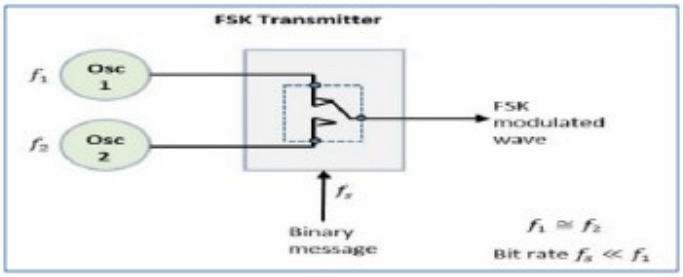


Fig.2: Basic principle of FSK modulator block

Frequency-shift keying (FSK) is a frequency modulation scheme in which digital information is transmitted through discrete frequency changes of a carrier wave. The simplest FSK is binary FSK (BFSK). BFSK uses a pair of discrete frequencies to transmit binary (0s and 1s) information. With this scheme, the "1" is called the mark frequency and the "0" is called the space frequency. If the incoming bit is 1, a signal with frequency f1 is sent for the duration of the bit. If the bit is 0, a signal with frequency f2 is sent for the duration of this bit. This is the basic principle behind FSK modulation.

Waveform:

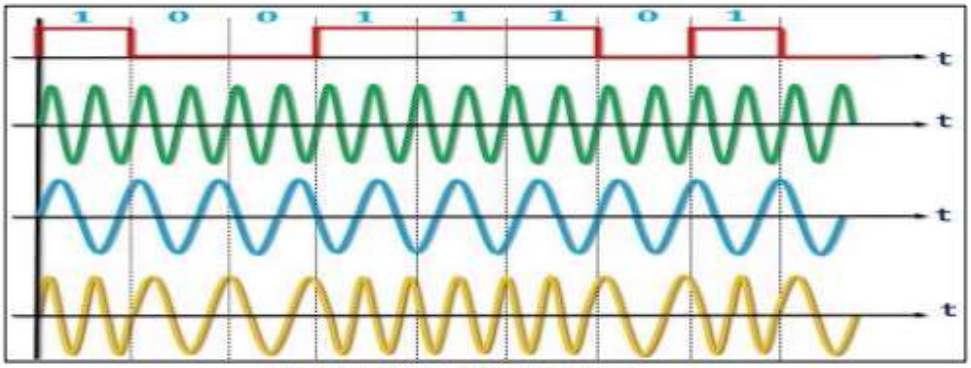


Fig.3: Waveform of FSK Modulator

First waveform is Digital bit stream according to it switching process will be proceed. Second And third waveforms are HIGH frequency carrier wave & LOW frequency carrier waveform Respectively. Fourth is FSK modulation wave, here when input bit stream is 1 then get HIGH Frequency and 0 then LOW frequency.

Procedure:

Modulation:

1. Connect all the blocks in Simulink according to given steps.(Which is given in FSK_designingStep document).

2. After designing entire diagram click on RUN.

3. Observe the waveforms at output of modulator using virtual scope.

Observation Table:

Higher Frequency	Lower Frequency
100 Hz	400 Hz
1 KHz	2 KHz
2 KHz	5 KHz

Result:

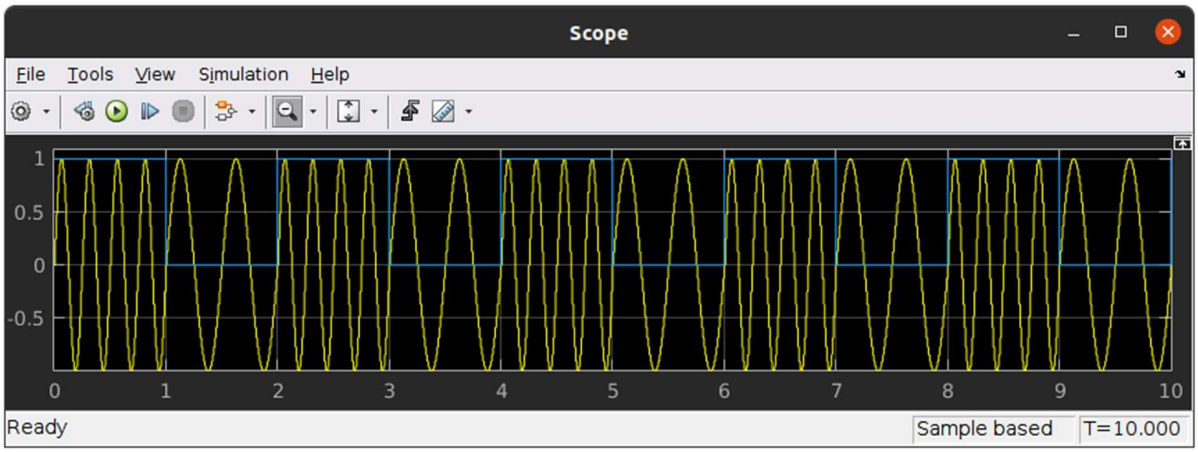


Figure: FSK Modulation

Binary Phase Shift Keying (BPSK) Modulation

Block Diagram:

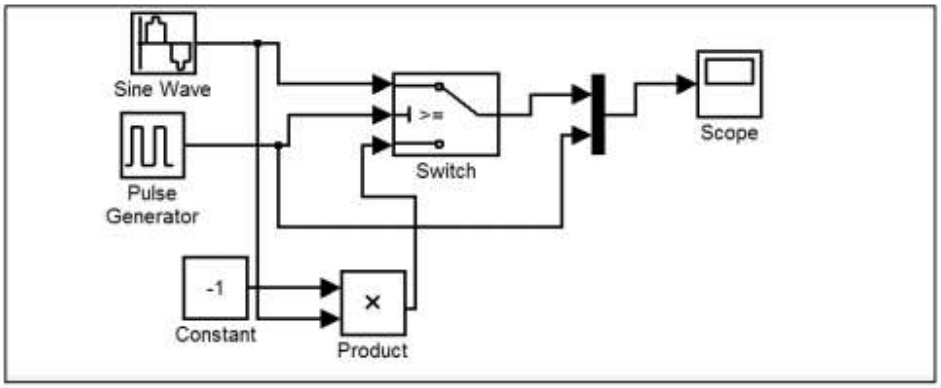


Fig.1: Block Diagram of BPSK Modulator in Simulink MATLAB

Theory:

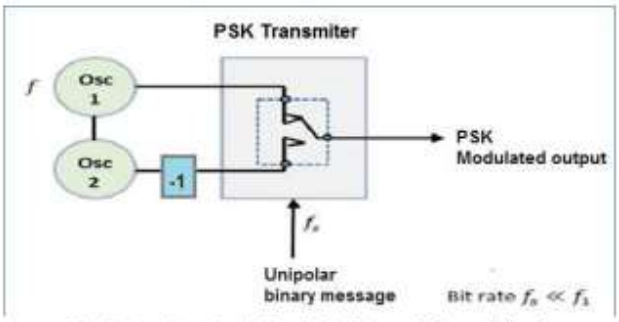


Fig.2: Basic principle of BPSK modulator block

BPSK is a digital modulation scheme which is analogues to phase modulation. Binary Phase Shift Keying (BPSK) is the simplest form of PSK. In binary phase shift keying two output phases are possible for a single carrier frequency one out of phase represent logic 1 and logic 0. As the input digital binary signal change state the phase of output carrier shift two angles that are 180 degrees out of phase.

Waveform:

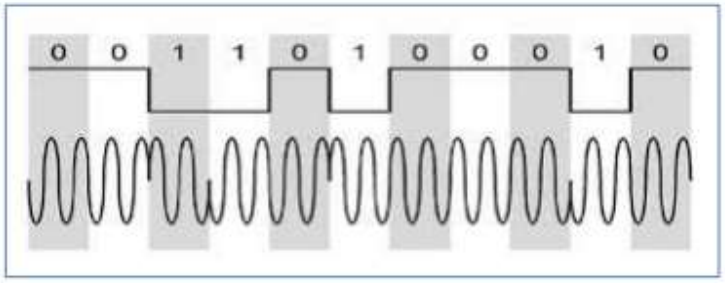


Fig.3: Waveform of BPSK Modulator

First waveform is Digital bit stream according to it switching process will be proceed. Second is BPSK modulation wave, here when input bit stream is 1 then get direct signal wave at output without shifting. But when 0 is come then carrier signal multiply with -1 so that whatever amplitude of carrier signal has that reverse its value; and output of such signal shows 180 degrees phase shifted. So when input 1 then output is 0 degrees shifted and 0 then 180 degrees shifted.

Procedure:

Modulation:

1. Connect all the blocks in Simulink according to given steps.(Which is given in PSK_designingStep document).

2. After designing entire diagram click on RUN.

3. Observe the waveforms at output of modulator using virtual scope.

Observation Table:

Frequency
100 Hz
1 KHz
2 KHz

Result:

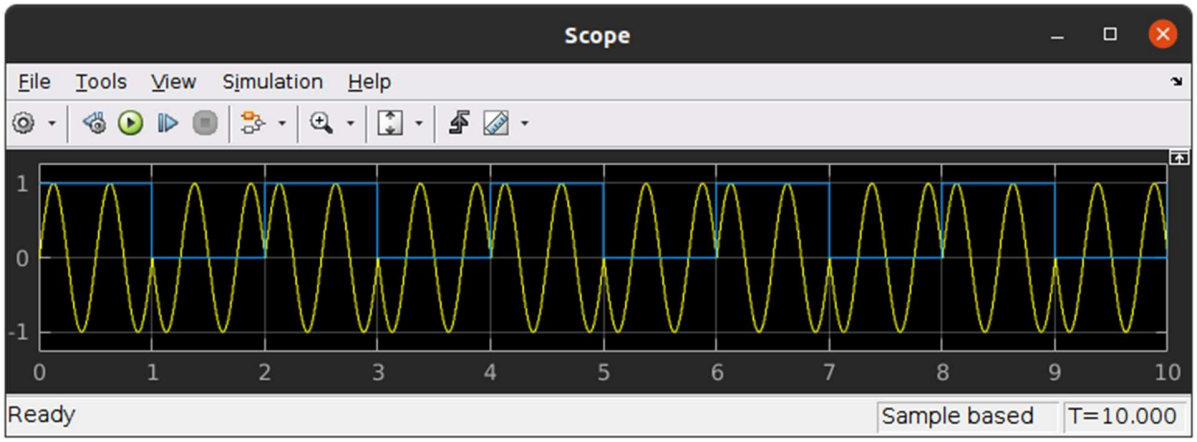


Figure: BSK Modulation

Conclusion:

- ASK Modulation has been done correctly in MATLAB. The required code and graph has been attached for two sequences of BIT streams.
- FSK Modulation has been done using MATLAB Simulink whose block diagram has been mentioned above.
- PSK Modulation has been done using MATLAB Simulink with required block diagram and observations in the scope (graphs).

Remarks: **Signature**

EXPERIMENT 9: AMPLITUDE MODULATION IN MATLAB

Date: 28/06/2020

Aim:

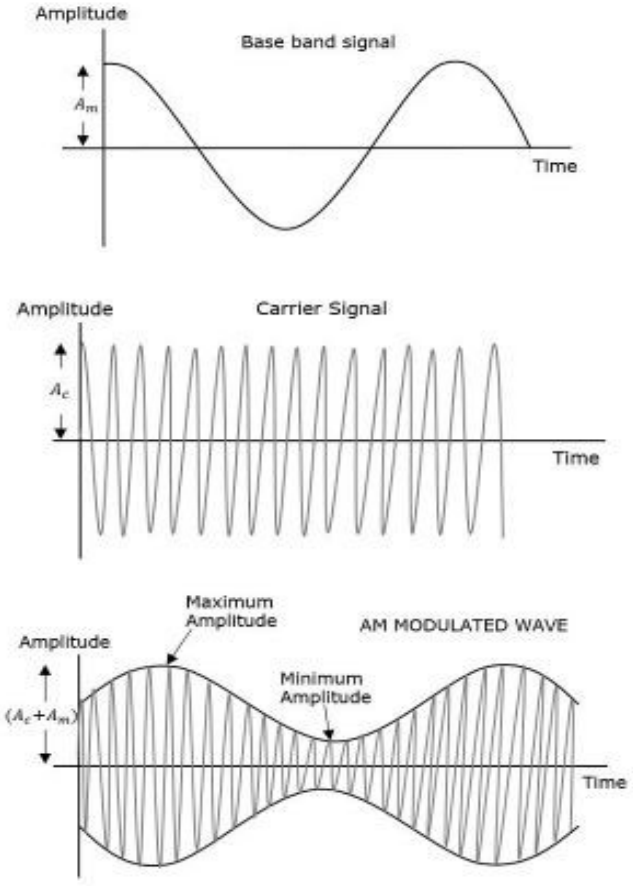
To implement amplitude modulation and demodulation using MATLAB.

Apparatus: MATLAB Software

Theory:

A continuous-wave goes on continuously without any intervals and it is the baseband message signal, which contains the information. This wave has to be modulated.

According to the standard definition, "The amplitude of the carrier signal varies in accordance with the instantaneous amplitude of the modulating signal." Which means, the amplitude of the carrier signal containing no information varies as per the amplitude of the signal containing information, at each instant. This can be well explained by the following figures.



The first figure shows the modulating wave, which is the message signal. The next one is the carrier wave, which is a high frequency signal and contains no information. While, the last one is the resultant modulated wave.

It can be observed that the positive and negative peaks of the carrier wave, are interconnected with an imaginary line. This line helps recreating the exact shape of the modulating signal. This imaginary line on the carrier wave is called as Envelope. It is the same as that of the message signal.

Time-domain Representation of the Waves

Let the modulating signal be,

$$m(t) = A_m cos(2\pi f_m t)$$

and the carrier signal be,

$$c(t) = A_c cos(2\pi f_c t)$$

Where,

Am and *Ac* are the amplitude of the modulating signal and the carrier signal respectively. *fm* and *fc* are the frequency of the modulating signal and the carrier signal respectively. Then, the equation of Amplitude Modulated wave will be

$$s(t) = [A_c + A_m \, cos(2\pi f_m t)]cos(2\pi f_c t)$$

Modulation index:

A carrier wave, after being modulated, if the modulated level is calculated, then such an attempt is called as Modulation Index or Modulation Depth. It states the level of modulation that a carrier wave undergoes.

$$s(t) = A_c \left[1 + \left(\frac{A_m}{A_c}\right) cos(2\pi f_m t)\right]cos(2\pi f_c t)$$

$$s(t) = A_c \left[1 + \mu \, cos(2\pi f_m t)\right]cos(2\pi f_c t)$$

Where, μ is Modulation index and it is equal to the ratio of Am and Ac. Hence, we can calculate the value of modulation index by using the above formula, when the amplitudes of the message and carrier signals are known.

Now, let us derive one more formula for Modulation index by considering Equation. We can use this formula for calculating modulation index value, when the maximum and minimum amplitudes of the modulated wave are known.

Let Amax and Amin be the maximum and minimum amplitudes of the modulated wave. We will get the maximum amplitude of the modulated wave, when *cos(2πfmt)* is 1. *Amax=Ac+Am.*

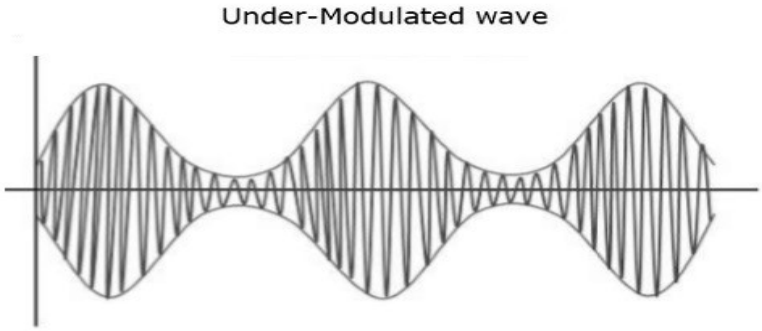
We will get the minimum amplitude of the modulated wave, when *cos(2πfmt)*is -1. *Amin=Ac−Am*

$$Amax+Amin = Ac+Am+Ac-Am=2\,Ac \;\Rightarrow\; Ac = \frac{Amax+Amin}{2}$$

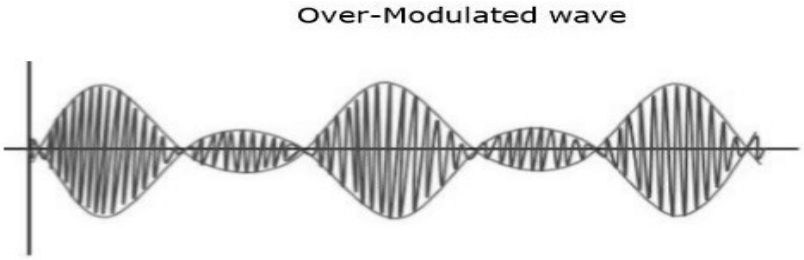
$$Amax-Amin = Ac+Am-(Ac-Am)=2\,Am \;\Rightarrow Am=\frac{Amax-Amin}{2}$$

$$\frac{\text{Am}}{\text{Ac}} = \frac{(Amax-Amin)/2}{(Amax+Amin)/2} \Rightarrow \mu = \frac{Amax-Amin}{Amax+Amin}$$

Therefore, there are the two formulas for Modulation index. The modulation index or modulation depth is often denoted in percentage called as Percentage of Modulation. We will get the **percentage of modulation**, just by multiplying the modulation index value with 100. For a perfect modulation, the value of modulation index should be 1, which implies the percentage of modulation should be 100%. For instance, if this value is less than 1, i.e., the modulation index is 0.5, then the modulated output would look like the following figure. It is called as **Under-modulation**. Such a wave is called as an **under-modulated wave**.



If the value of the modulation index is greater than 1, i.e., 1.5 or so, then the wave will be an **over-modulated wave**. It would look like the following figure. As the value of the modulation index increases, the carrier experiences a 180^{o} phase reversal, which causes additional sidebands and hence, the wave gets distorted. Such an over-modulated wave causes interference, which cannot be eliminated.



Reference: Modern Digital and Analog Communication by B.P.Lathi

Bandwidth of AM:

Bandwidth (BW) is the difference between the highest and lowest frequencies of the signal. Mathematically, we can write it as $BW = f_{max} - f_{min}$ Consider the following equation of amplitude modulated wave.

$$s(t) = A_c[1 + \mu\, cos(2\pi f_m t)]cos(2\pi f_c\, t)$$

$$s(t) = A_c\, cos(2\pi f_c t) + A_c\, \mu\, cos(2\pi f_c t)cos(2\pi f_m t)$$

$$s(t) = A_c\, cos(2\pi f_c t) + \frac{A_c\, \mu}{2} cos[2\pi(f_c + f_m)t] + \frac{A_c\, \mu}{2} cos[2\pi(f_c - f_m)t]$$

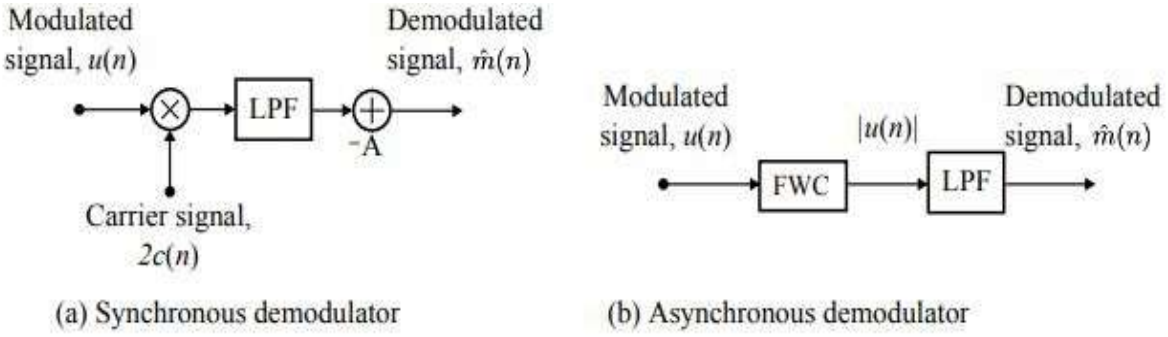
Hence, the amplitude modulated wave has three frequencies. Those are carrier frequency *fc,* upper sideband frequency *fc+fm* and lower sideband frequency *fc−fm* Here, *fmax=fc+fm* and *fmin=fc−fm.* Substitute, *fmax* and *fmin* values in bandwidth formula.

$$BW=(fc+fm) - (fc-fm) \Rightarrow BW=2fm$$

Thus, it can be said that the bandwidth required for amplitude modulated wave is twice the frequency of the modulating signal.

Amplitude Demodulation:

There are two types of demodulation: synchronous demodulation and asynchronous demodulation. They differ in the use of a carrier signal in demodulation processes. The receiver must generate a carrier signal synchronized in phase and frequency for synchronous demodulation, but the carrier signal is not needed in asynchronous demodulation.



Synchronous Demodulation. This kind of demodulation can be referred to as a coherent or product detector. Figure (a) shows a flow chart of a synchronous demodulator. At the receiver, we multiply the incoming modulated signal by a local carrier of frequency and phase in synchronism with the carrier used at the transmitter.

$$e(t) = u(t)c(t) = [A + m(t)]\, cos^2(w_c t)$$

$$= \frac{1}{2}\,[A + m(t)] + \frac{1}{2}\,[A + m(t)]\, cos(2w_c t)$$

Let us denote $m_a(t) = A + m(t)$

$$e(t) = \frac{1}{2}\, m_a(t) + \frac{1}{2}\, m_a(t)\, cos(2w_c t)$$

The fourier transform of the signal e(t) is

$$E(w) = \frac{1}{2} M_a(w) + \frac{1}{2}[M_a(w + 2w_c) + M_a(w - 2w_c)$$

The spectrum $E(w)$ consists of three components as shown in figure. The first component is the message spectrum. The two other components, which are the modulated signal of m(t) with carrier frequency $2w_c$., are centered at $\pm 2w_c$. The signal *e(t)* is then filtered by a lowpass filter (LPF) with a cut-off frequency of fc to yield $\frac{m_a(t)}{2}$. We can fully get $m_a(t)$ by multiplying the output by two. We can also get rid of the inconvenient fraction 1 2 from the output by using the carrier 2 $cos(w_c)$. instead of $cos(w_c)$. Finally, the message signal m(t) can be recovered by ˆm(t) = $m_a(t)$ − A.

Asynchronous Demodulation. An asynchronous demodulator can be referred to as an envelope detector. For an envelope detector, the modulated signal *u(t)* must satisfy the requirement that *A + m(t) ≥ 0, ∀t.* A block diagram of the asynchronous demodulator is shown in Figure (b). The incoming modulated signal, u(t), is passed through a full wave rectifier (FWR) which acts as an absolute function. The FWR output which is the absolute value of *u(t), |u(t)|,* is then filtered by a low-pass filter resulting in the demodulated signal, *ˆm(t).*

Algorithm:

- Define the sampling frequency $f_s \geq 2(f_m)$ say, $f_s = 100Hz$;
- Define the time range using the sampling frequency t= -10 : $1/f_s$: 10
- Consider, message signal, $m = A_m \cos \omega_m t$, where $f_m = 1Hz$ and carrier signal $c = A_c \cos \omega_c t$where $f_c = 10Hz$. Keep the amplitude of message and carrier signal same.
- Assume $w_m = 2\pi f_m$ and $w_c = 2\pi f_c$
- For AM, $\boldsymbol{s(t) = [A_c + A_m cos(2\pi f_m t)]cos(2\pi f_c t)}$
- Figure1: Plot input signal, carrier signal, AM signal using subplot;
- Figure 2: plot the frequency spectrum of AM signal using the commands fft and fftshift.
  - Define n=length(t) %gives no. of columns in t
  - Define the step size for frequency axis fp which should be of same size as that of t. (matrix dimensions must match for plotting).
    - df=fs/n; where fs is the sampling frequency
  - Define frequency axis fp = -fs/2:df:fs/2-df (±df for getting same size matrices)
  - Take the fourier transform of AM using fft command
  - Y = fft(x) returns the discrete Fourier transform (DFT) of vector x, computed with a fast Fourier transform (FFT) algorithm.
  - Y = fftshift(X) rearranges the outputs of fft by moving the zero-frequency component to the center of the array. It is useful for visualizing a Fourier transform with the zero-frequency component in the middle of the spectrum.
  - Can write in a single syntax as y=fftshift(fft(x));
  - Plot the frequency spectrum of AM using the command plot(fp,y).
- **Demodulate the AM signal**
  - Multiply the above signal with carrier signal.

- Do .* element wise multiplication

- Do the low pass filtering of the above signal using butter and filter command.
  - [b,a] = butter(n,Wn,'ftype'). **This creates a filter**
  - [b,a] = butter(n,Wn) designs an order n lowpass digital Butterworth filter with normalized cutoff frequency Wn. It returns the filter coefficients in length n+1 row vectors b and a, with coefficients in descending powers of z.

$$H(z) = \frac{b(1) + b(2)z^{-1} + \ldots + b(n+1)z^{-n}}{1 + a(2)z^{-1} + \ldots + a(n+1)z^{-n}}$$

  - where the string 'ftype' is one of the following:
    - 'high' for a highpass digital filter with normalized cutoff frequency Wn
    - 'low' for a lowpass digital filter with normalized cutoff frequency Wn
    - 'stop' for an order 2*n bandstop digital filter if Wn is a two-element vector, Wn = [w1 w2]. The stopband is w1 < ω < w2.
    - 'bandpass' for an order 2*n bandpass filter if Wn is a two-element vector, Wn = [w1 w2]. The passband is w1 < ω < w2. Specifying a two-element vector, Wn, without an explicit 'ftype' defaults to a bandpass filter.
    - Cutoff frequency, Wn is that frequency where the magnitude response of the filter is . For butter, the normalized cutoff frequency Wn must be a number between 0 and 1, where 1 corresponds to the Nyquist frequency, π radians per sample.
    - Choose order, n=3 ; Wn=fm/fs; ftype='low'
- Use the filter command to apply the filter
  - y = filter(b,a,X), where b and a are the coefficients obtained in the previous step and X will the multiplied signal output. y is the filtered output. Adjust order so that you get approximate input signal.
- Figure 3: Plots for AM demodulation
  - Include three figures using subplot(211) to subplot(212)
    - 1st - multiplied signal output (modulated AM signal*Carrier Signal)
    - 2nd – filtered output y. Include message signal as well in this
      - Example: plot(t,m,t,y)

Expected Output waveforms:

Figure 1:

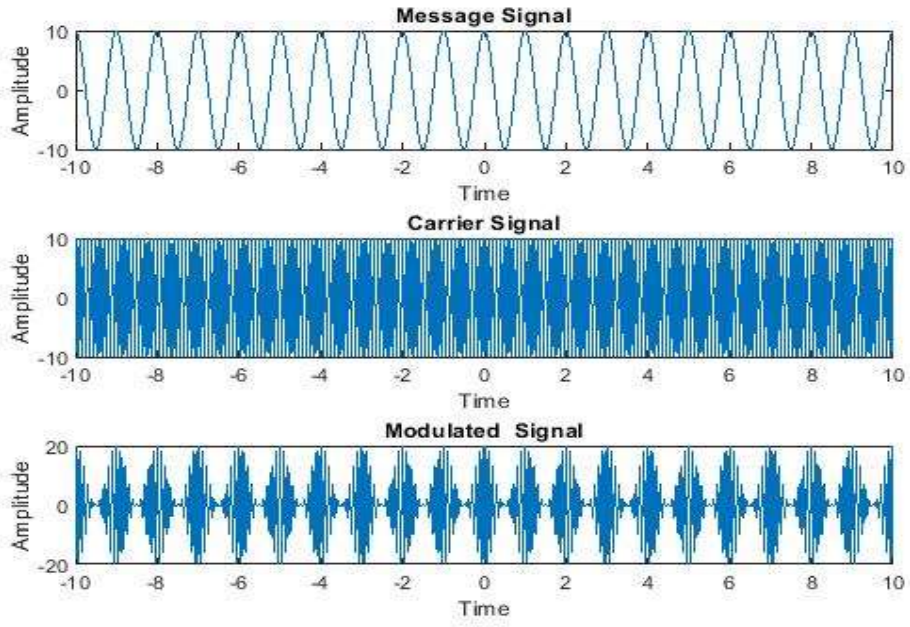


Figure 2:

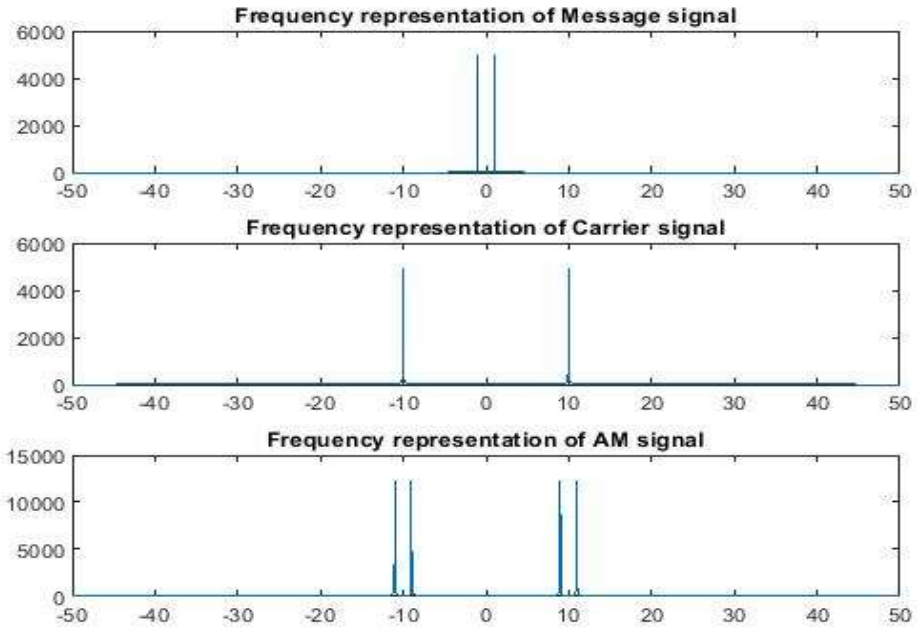
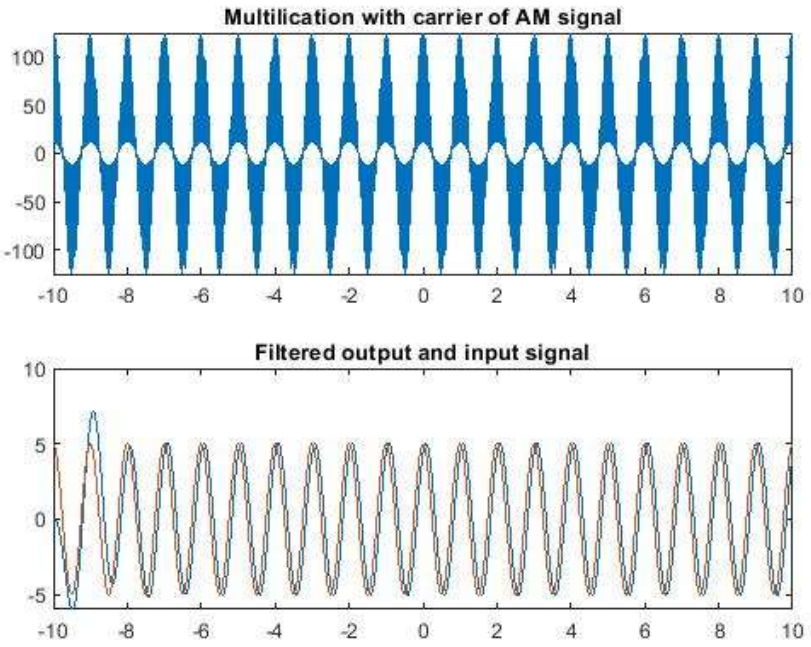


Figure 3:



Code:

```
% Experiment 8: Amplitude Modulation and Demodulation

% Author: Krunal Rank

% RollNo: U18CO081



%% Part 1: Amplitude Modulation

clc;

clear all;

close all;



f_s = 100;
```

```
t = -10:(1/f_s):10;


f_m = 1;

f_c = 10;


w_m = 2 * pi * f_m;

w_c = 2 * pi * f_c;


A_m = 10;

A_c = A_m;


m = A_m * cos(w_m*t);

c = A_c * cos(w_c*t);


s = m.*c;


subplot(3,1,1);

plot(t,m);

xlabel('t (in S)');

ylabel('m(t) (in V)');
```

```
title('Message Signal');


subplot(3,1,2);

plot(t,c);

xlabel('t (in S)');

ylabel('c(t) (in V)');

title('Carrier Signal');


subplot(3,1,3);

plot(t,s);

xlabel('t (in S)');

ylabel('s(t) (in V)');

title('Modulated Signal');


%% Part 2: Frequency Spectrum of Modulated Signal


n = length(t);

df = f_s/n;


fp = -f_s/2:df:f_s/2 - df;
```

```
fourier_m = fftshift(fft(m));

fourier_c = fftshift(fft(c));

fourier_s = fftshift(fft(s));


subplot(3,1,1);

plot(fp,fourier_m);

xlabel('f (in Hz)');

ylabel('m(f) (in V)');

title('Message Signal');


subplot(3,1,2);

plot(fp,fourier_c);

xlabel('f (in Hz)');

ylabel('c(f) (in V)');

title('Carrier Signal');


subplot(3,1,3);

plot(fp,fourier_s);

xlabel('f (in Hz)');
```

```
ylabel('s(f) (in V)');

title('Modulated Signal');


%% Part 3: Demodulation of AM Signal


s_1 = s.*c;


[b, a] = butter(3,f_m/f_s,'low');


[y,z] = filter(b,a,s_1);

mu = mean(y);

y = filter(b,a,y,z)-mu;

subplot(2,1,1);

plot(t,s_1);

xlabel('t (in S)');

ylabel('s(t)*c(t) (in V^2)');

title('Multilication with carrier of AM Signal');


subplot(2,1,2);

plot(t,m,t,y);
```

```
xlabel('t (in S)');

ylabel('m(t),y(t) (in V)');

title('Filtered Output and Input Signal');
```

Output Waveforms:

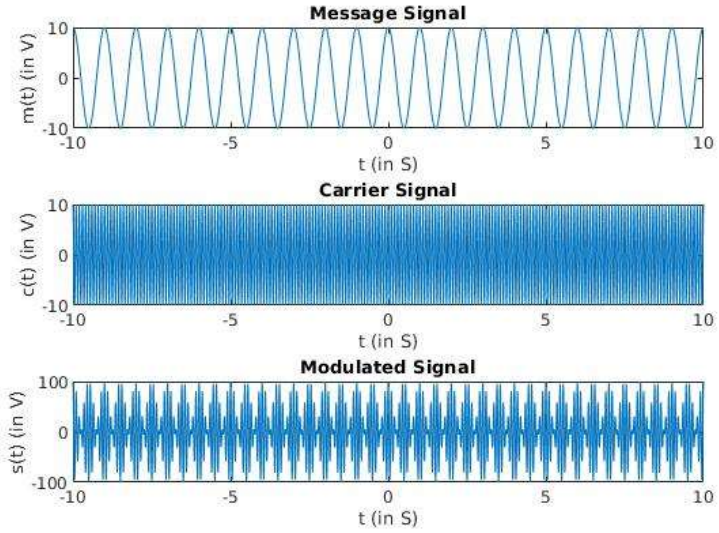


Figure: AM Modulation

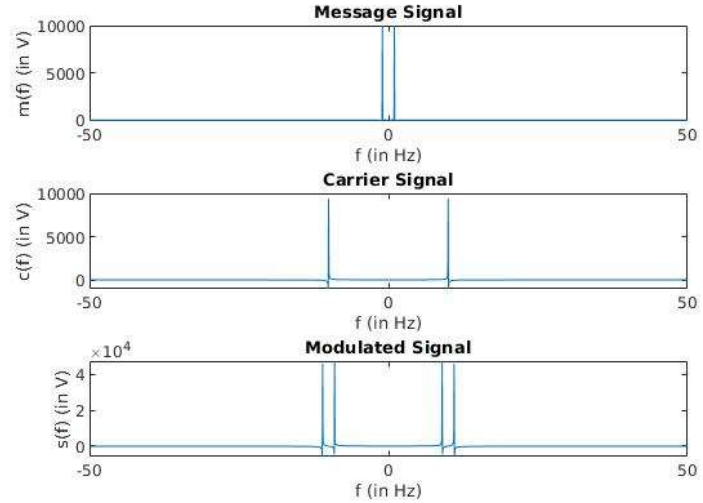


Figure: AM Modulation Frequency Spectrum

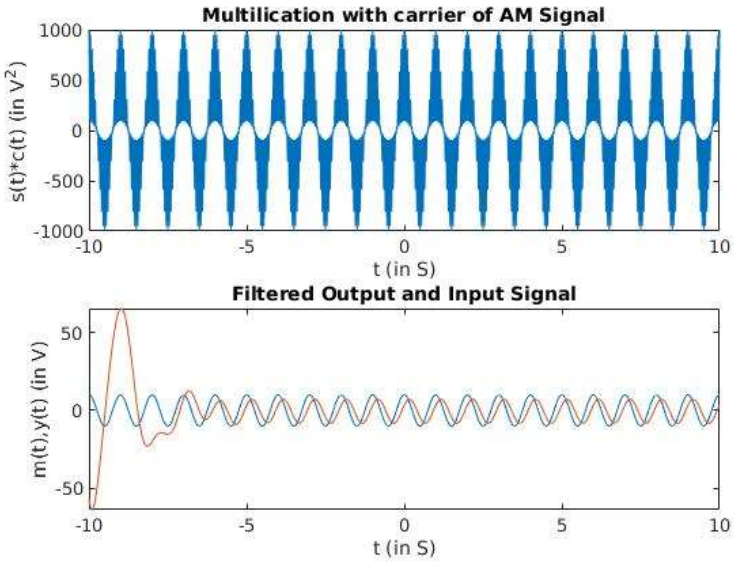


Figure: AM Demodulation

Conclusion:

AM Modulation and Demodulation has been done using MATLAB.

AM Modulation required usage of vectors and the frequency spectrum has been plot using numerical fast fourier shifted transforms.

AM Demodulation has been done using a low pass band filter available in MATLAB.

Remarks: **Signature:**

EXPERIMENT 10: DELTA MODULATION-DEMODULATION

Date: 28/06/2020

Aim:

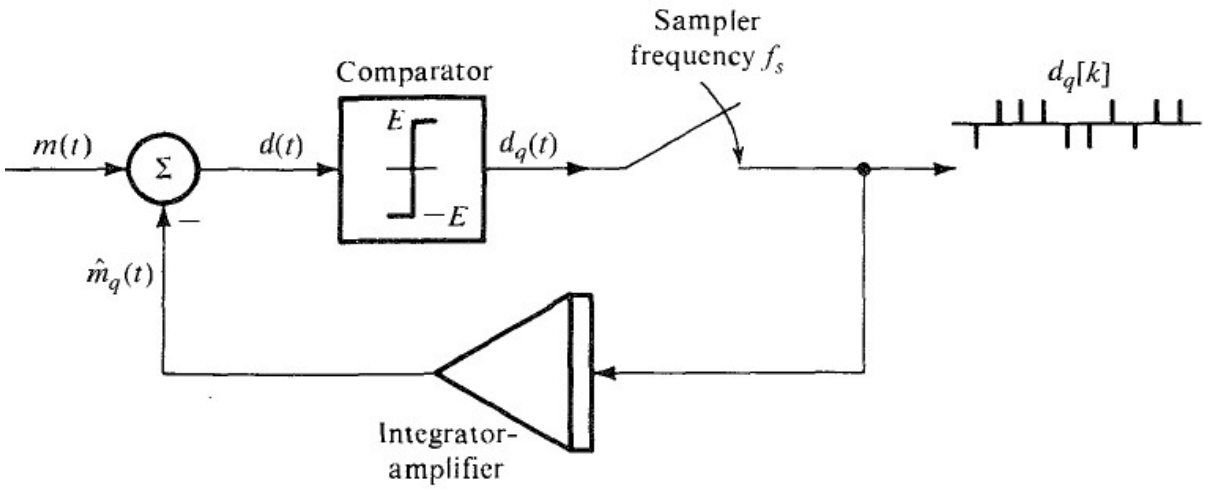
Write a MATLAB code to modulate and demodulate the given signal by Delta Modulation Technique.

Apparatus: MATLAB Software

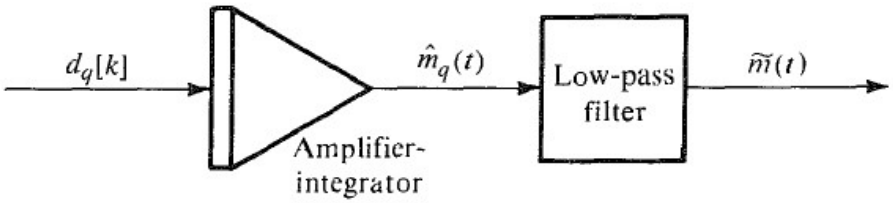
Theory:

A delta modulation (DM or Δ-modulation) is an analog-to-digital and digital-to-analog signal conversion technique used for transmission of voice information where quality is not of primary importance. DM is the simplest form of differential pulse-code modulation (DPCM) where the difference between successive samples is encoded into n-bit data streams. In delta modulation, the transmitted data are reduced to a 1-bit data stream. Its main features are:

- The analog signal is approximated with a series of segments.
- Each segment of the approximated signal is compared to the preceding bits and the successive bits are determined by this comparison.
- Only the change of information is sent, that is, only an increase or decrease of the signal amplitude from the previous sample is sent whereas a no-change condition causes the modulated signal to remain at the same 0 or 1 state of the previous sample.
- To achieve high signal-to-noise ratio, delta modulation must use oversampling techniques, that is, the analog signal is sampled at a rate several times higher than the Nyquist rate.



(a) Delta modulator

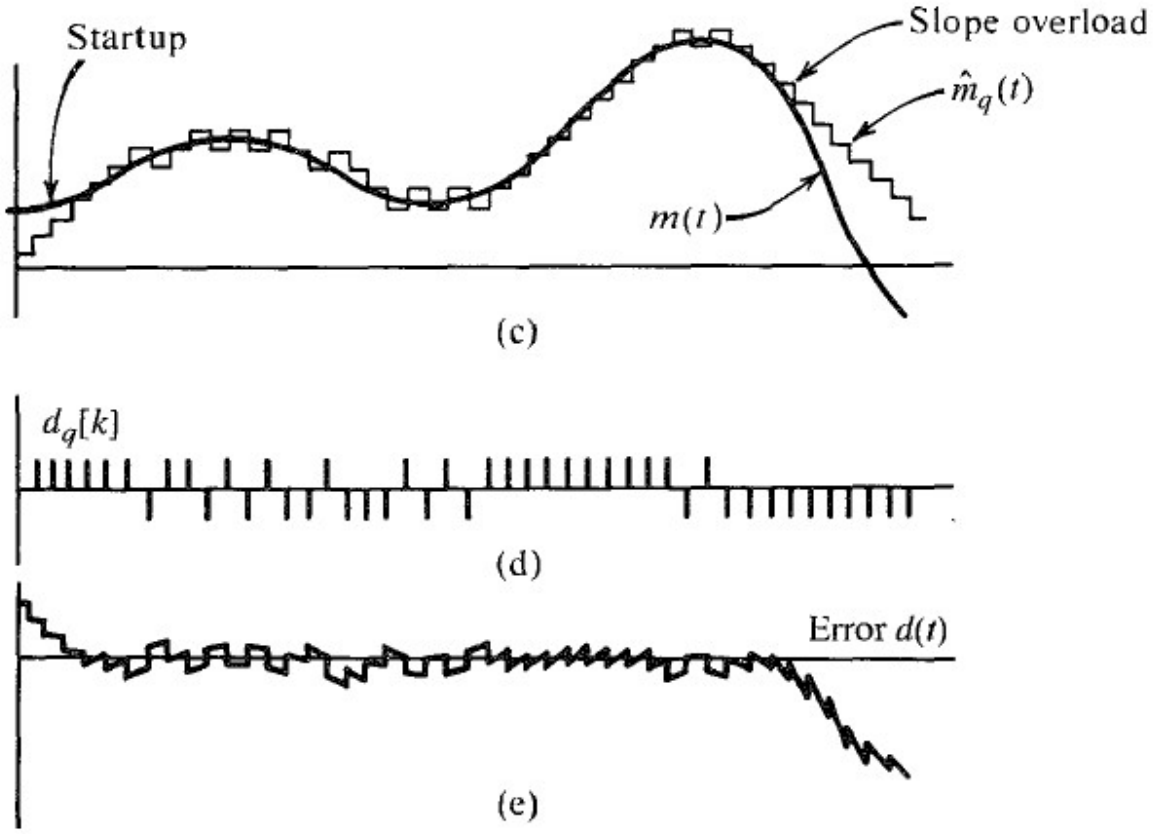


(b) Delta demodulator

The modulator consists of comparator and sampler in the direct path and an integrator amplifier in the feedback path. The analog signal $m(t)$ is compared with the feedback signal $\hat{m}_q(t)$. The error signal $d(t) = m(t) - \hat{m}_q(t)$ is applied to the comparator. If $d(t)$ is positive, the comparator output is of constant amplitude E and if $d(t)$ is negative, the comparator output is of constant value –E. Thus, the difference is the binary signal that is needed to generate 1 bit DPCM (Differential Pulse Code Modulation). The comparator output is sampled by a sampler at sampling frequency f_s, samples per second where f_s is much higher than the nyquist sampling rate. The sampler thus produces train of narrow pulses $d_q(k)$ (to simulate impulses) with a positive pulse when $m(t) > \hat{m}_q(t)$ and negative pulse when $m(t) < \hat{m}_q(t)$. Each sample is coded by a single binary pulse. The pulse train $d_q(k)$ is delta modulated pulse train. The modulated signal $d_q(k)$ is amplified and integrated in the feedback path to generate $\hat{m}_q(t)$, which tries to follow $m(t)$.

Each pulse in $d_q(k)$ at the input of the integrator gives rise to a step function (positive or negative depending on the polarity of pulse) in $\hat{m}_q(t)$. If for example, $m(t) > \hat{m}_q(t)$, a positive pulse is generated in $d_q(k)$, which gives rise to a positive step in $\hat{m}_q(t)$ trying to equalize $\hat{m}_q(t)$ to $m(t)$ in small steps at every sampling instant as shown in figure below. It can be seen that $\hat{m}_q(t)$ is kind of staircase approximation of $m(t)$. When $\hat{m}_q(t)$ is passed through a low pass filter, the coarseness of staircase in $\hat{m}_q(t)$ is eliminated and we get smoother and better approximation of $m(t)$. The

demodulator at the receiver consists of an amplifier-integrator (identical to that of feedback path of modulator) followed by a low pass filter.



In DM, the modulated signal carries information not about the signal samples but about the difference between successive samples. If the difference is positive or negative, a positive or negative pulse is generated in the modulated signal $d_q(k)$. Basically, therefore, DM carries the information about the derivative of $m(t)$, hence the name delta modulation. This can also be seen from the fact that integration of delta modulated signal yields $\hat{m}_q(t)$, which is an approximation of $m(t)$. The information of difference between successive samples is transmitted by a 1 bit code word.

Threshold of Coding and Overloading

Threshold and overloading effects can be seen in the figure c. Variations in $m(t)$, smaller than the step value (threshold of coding) are lost in DM. Moreover, If $m(t)$ is too fast, derivative of it $\dot{m}(t)$ is too high, $\hat{m}_q(t)$ cannot follow $m(t)$ and overloading occurs. This is known as slope overloading, which gives rise to the slope overload noise. This noise is one of the basic limiting factors in the performance of DM. We should expect slope overload rather than amplitude overload in DM, because DM basically carries the information about $m(t)$. The granular nature of the output signal gives rise to the granular noise similar to the quantization noise. The slope overload noise can be reduced by increasing E (the step size). This unfortunately increases the granular noise. There is

an optimum value of E, which yields the best compromise giving the minimum overall noise. This optimum value of E depends on the sampling frequency f_s and the nature of the signal.

The slope overload occurs when $\hat{m}_q(t)$ cannot follow $m(t)$. During the sampling interval T_s, $\hat{m}_q(t)$ is capable of changing by σ, where σ is the height of the step (amplitude). Hence, the maximum slope that $m(t)$ can follow is $\frac{\sigma}{T_s}$, or σf_s, where f_s, is the sampling frequency. Hence, no overload occurs if

$$|\dot{m}(t)| \geq \sigma f_s$$

Consider the case of tone modulation (meaning a sinusoidal message):

$$m(t) = A\cos\omega t$$

The condition for no overload is

$$|\dot{m}(t)|_{max} = \omega A < \sigma f_s$$

Hence, the maximum amplitude A_{max} of this signal that can be tolerated without overload is given by

$$A_{max} = \frac{\sigma f_s}{\omega}$$

The overload amplitude of the modulating signal is inversely proportional to the frequency ω. For higher modulating frequencies, the overload occurs for smaller amplitudes. For voice signals, which contain all frequency components up to (say) 4 kHz, calculating A_{max} by using$\omega = 2\pi \times 4000$ in above equation will give an overly conservative value. It has been shown that A_{max} for voice signals can be calculated by using $\omega_r \approx 2\pi \times 800$

$$[A_{max}]_{voice} \sim \frac{\sigma f_s}{\omega_r}$$

Thus, the maximum voice signal amplitude A_{max} that can be used without causing slope overload in DM is the same as the maximum amplitude of a sinusoidal signal of reference frequency f_r that can be used without causing slope overload in the same system

Fortunately, the voice spectrum (as well as the television video signal) also decays with frequency and closely follows the overload characteristics. For this reason, DM is well suited for voice (and television) signals. Actually, the voice signal spectrum (curve b) decreases as $\frac{1}{\omega}$ up to 2000 Hz, and beyond this frequency, it decreases as $\frac{1}{\omega^2}$. If we had used a double integration in the feedback circuit instead of a single integration, A_{max} would be proportional to $\frac{1}{\omega^2}$. Hence, a better match between the voice spectrum and the overload characteristics is achieved by using a single integration up to 2000 Hz and a double integration beyond 2000 Hz. Such a circuit (the double integration) responds fast but has a tendency to instability, which can be reduced by using some low-order prediction along with double integration. A double integrator can be built by placing in

cascade two low-pass RC integrators with time constants R1 C1 = 1 /200pi and R2C2 = 1/4000pi, respectively. This results in single integration from 100 to 2000 Hz and double integration beyond 2000 Hz.

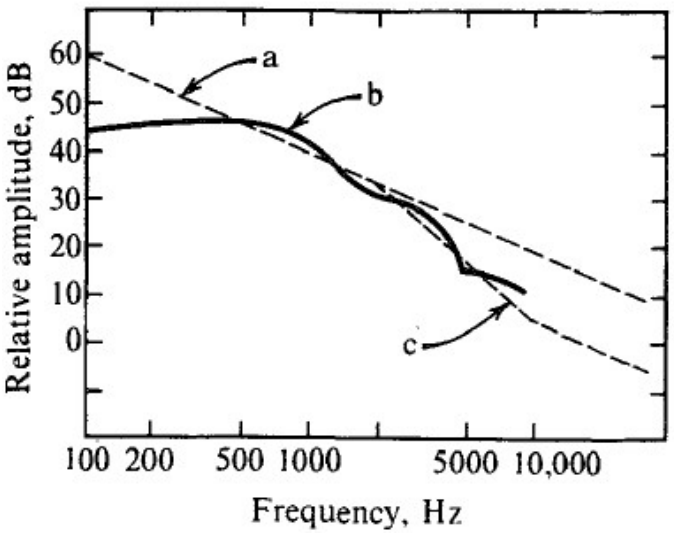


Figure 6.21 Voice signal spectrum.

Delta Modulation : A special case of DPCM

Sample correlation used in DPCM is further exploited in ***delta modulation (DM)*** by oversampling (typically four times the Nyquist rate) the baseband signal. This increases the correlation between adjacent samples, which results in a small prediction error that can be encoded using only one bit (L = 2). Thus, DM is basically a 1-bit DPCM, that is, a DPCM that uses only two levels (L = 2) for quantization of $m[k] - m_q[k]$. In comparison to PCM (and DPCM), it is a very simple and inexpensive method of *A/D* conversion. A 1-bit codeword in DM makes word framing unnecessary at the transmitter and the receiver. This strategy allows us to use fewer bits per sample for encoding a baseband signal.

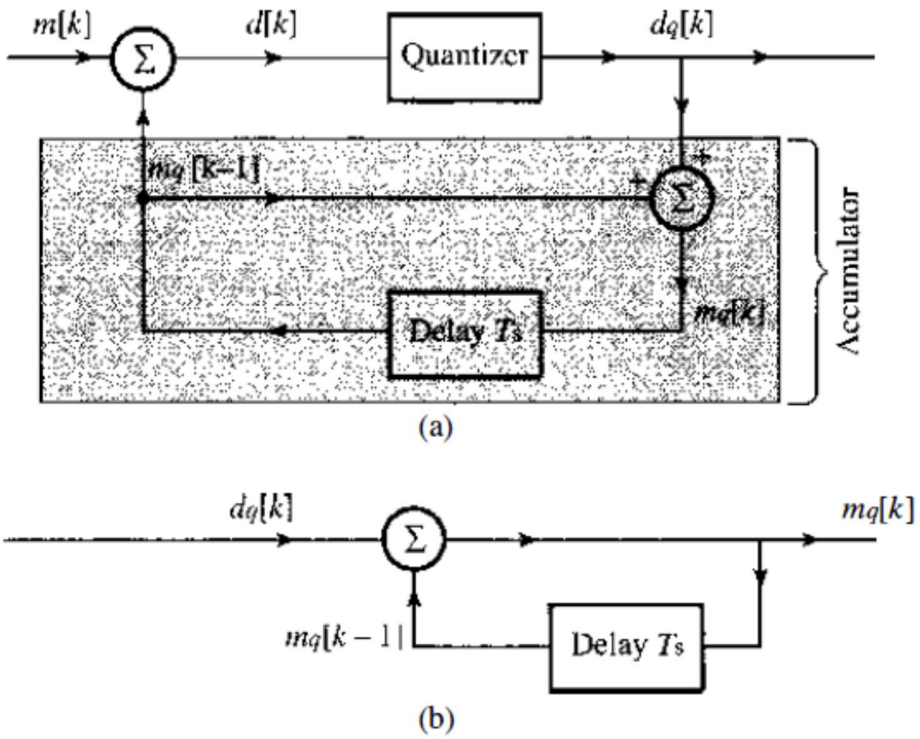
In DM, we use a first-order predictor, which, as seen earlier, is just a time delay of T_s, (the sampling interval). Thus, the DM transmitter (modulator) and receiver (demodulator) are identical to those of the DPCM in Fig. below, with a time delay for the predictor, as shown in Fig, from which we can write

$$m_q[k] = m_q[k-1] + d_q[k]$$

Hence,

$$m_q[k-1] = m_q[k-2] + d_q[k-1]$$

Figure 6.30
Delta modulation is a special case of DPCM.



Or we can write,

$$m_q[k] = m_q[k-2] + d_q[k-1] + d_q[k]$$

Proceeding in this manner, assuming zero initial condition, i.e. $m_q[0]$=0, we write

$$m_q[k] = \sum_{m=0}^{k} d_q[m]$$

This shows that the receiver (demodulator) is just an accumulator (adder). If the output $d_q[k]$ is represented by impulses, then the accumulator (receiver) may be realized by an integrator because its output is the sum of the strengths of the input impulses (sum of the areas underthe impulses). We may also replace with an integrator the feedback portion of the modulator (which is identical to the demodulator). The demodulator output is $m_q[k]$, which when passed through a low-pass filter yields the desired signal reconstructed from the quantized samples.

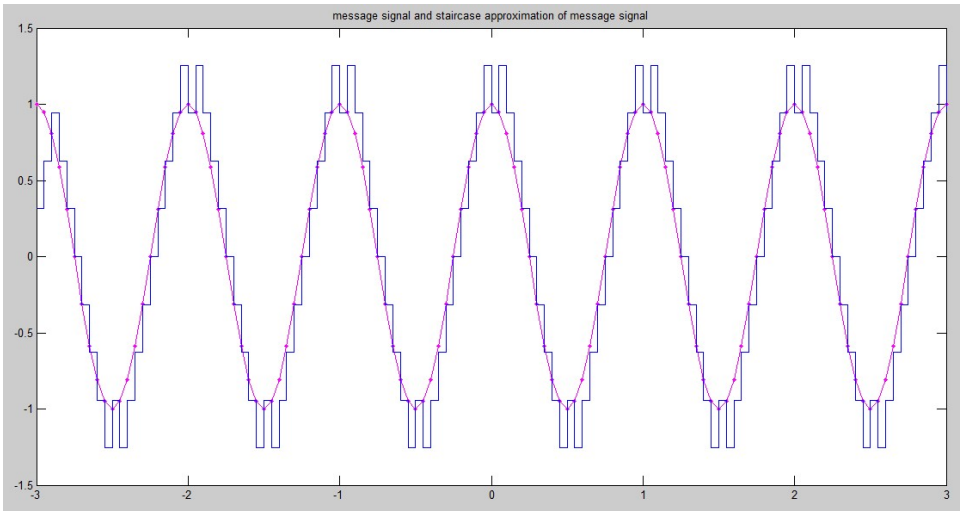
Algorithm:

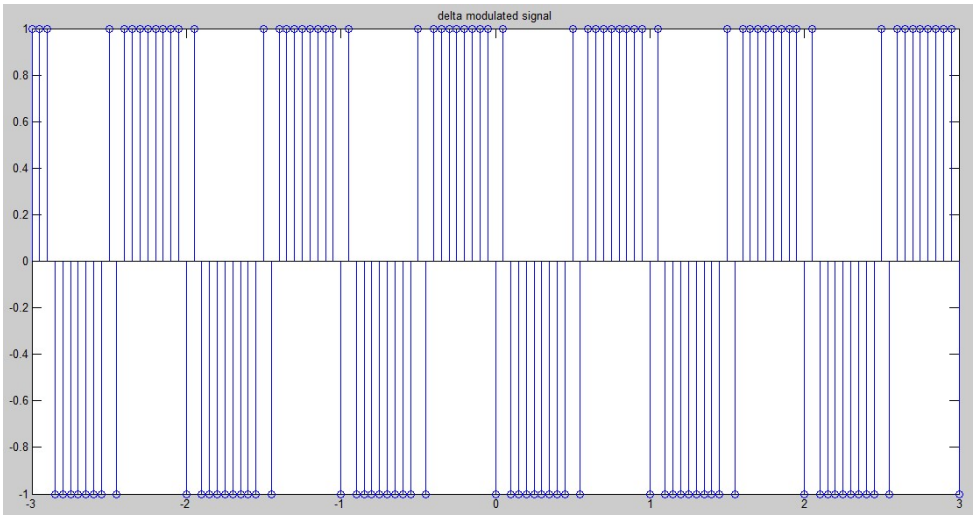
- Impement the block diagram of DM as a special case of DPCM (Figure 6.30, page 295, "Modern Analog and Digital Communication" by B.P. Lathi 4th edition)
- Consider the input/message signal as sinusoidal `m=Am*cos(2*pi*fm*t)`, with parameters, Am=1V, fm=1Hz.
- Define the time range with sampling frequency fs=20*fm (oversampling), hence, t can be defined as `t=-3:1/fs:3;`
- Define the step size del for the delta modulator which should satisfy the condition

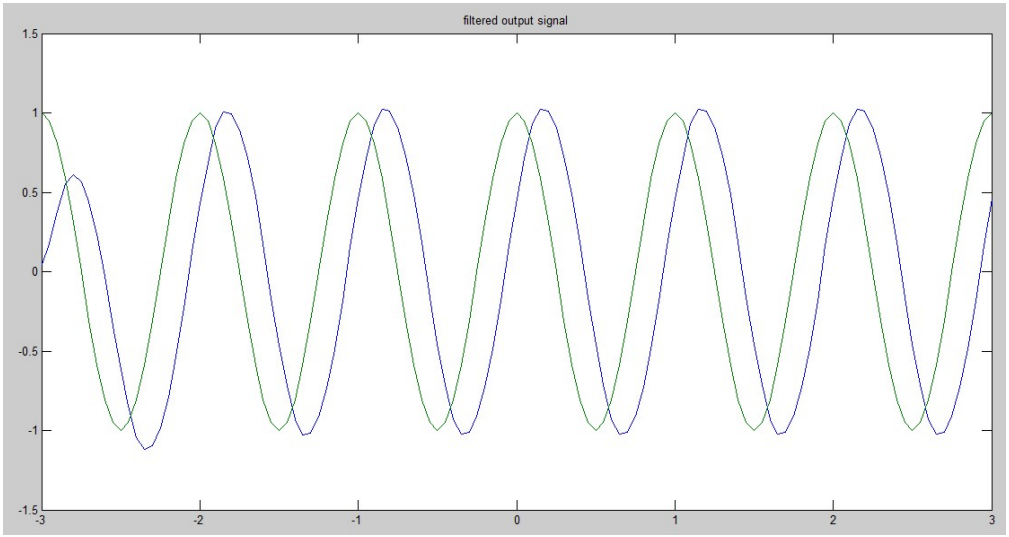
$$\Delta \leq \frac{2\pi A_m f_m}{f_s}$$

  Hence, `del=(2*pi*fm*Am)/fs;`
- Choose the index i=1:length(t) ; length(t)=max no. of columns in t.
  - If i=1, then mq=0.
    - So, the difference signal `d(i)=m(i);`
    - Use the sign function of MATLAB to determine whether d is +ve or –ve
    - Determine the approximate difference value dq by applying hard limiting operation i.e. by multiplying sign(d) with del.
    - Approximated message signal mq=dq, for i=1.
  - Else
    - The difference signal, `d(i)=m(i)-mq(i-1);`
    - The approximated difference operation will be same as in case for i=1.
    - Approximated message signal (staircase approximation), `mq(i)=dq(i)+mq(i-1);`
- Figure1: plot the message signal and staircase approximation signal the same window. Use the command hold on. And use command stairs(t,mq) for approximated message signal.
- Figure2: plot the delta modulated signal, consider the modulated output x to be +1 if dq>0 else x will be -1.
  - Use stem command for plotting as it is a discrete time signal
- For demodulation, pass the approximated message signal via low pass filter. (See FM demodulation algorithm for filtering logic).
- Figure 3: plot filtered output signal and original input signal in the same window.
  - `plot(t,2*y,t,m);`

Expected Ouput Waveforms:


message signal and staircase approximation of message signal
1.5
1
0.5
0
-0.5
-1
-1.5
-3
-2
-1
0
1
2
3


delta modulated signal
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-3
-2
-1
0
1
2
3


filtered output signal
1.5
1
0.5
0
-0.5
-1
-1.5
-3
-2
-1
0
1
2
3

Code:

```
% Experiment : Delta Modulation / Demodulation

% Author : Krunal Rank

% RollNo. : U18CO081



A_m = 1;

f_m = 1;

f_s = 20*f_m;



t = -3:1/f_s:3;



m = A_m*cos(2*pi*f_m*t);




del = (2*pi*f_m*A_m)/f_s;



dq = zeros(1,length(t));

d = zeros(1,length(t));

mq = zeros(1,length(t));
```

```
for i = 1: length(t)
    if i == 1
        mq(i) = 0;
        d(i) = m(i);
        dq(i) = sign(d(i))*del;
        mq(i) = dq(i);
    else
        d(i) = m(i) - mq(i-1);
        dq(i) = sign(d(i))*del;
        mq(i) = dq(i) + mq(i-1);
    end
end

%% Part 1

plot(t,m);
hold on;
stairs(t,mq);
hold off;
```

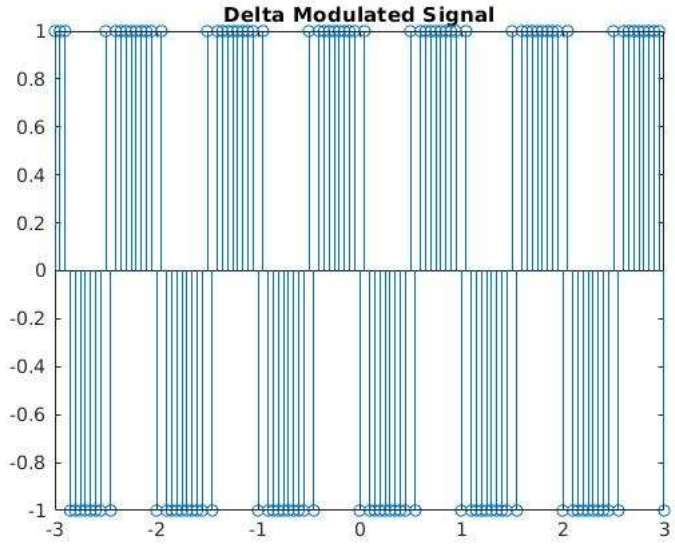
```
title('Message Signal and Staircase Approximation of Message Signal');


%% Part 2


x = zeros(1,length(t));
for i = 1: length(t)
    if dq(i) > 0
        x(i) = 1;
    else
        x(i) = -1;
    end
end


stem(t,x);
title('Delta Modulated Signal');


%% Part 3


[b,a] = butter(2,f_m/(0.7*f_s),'low');
```

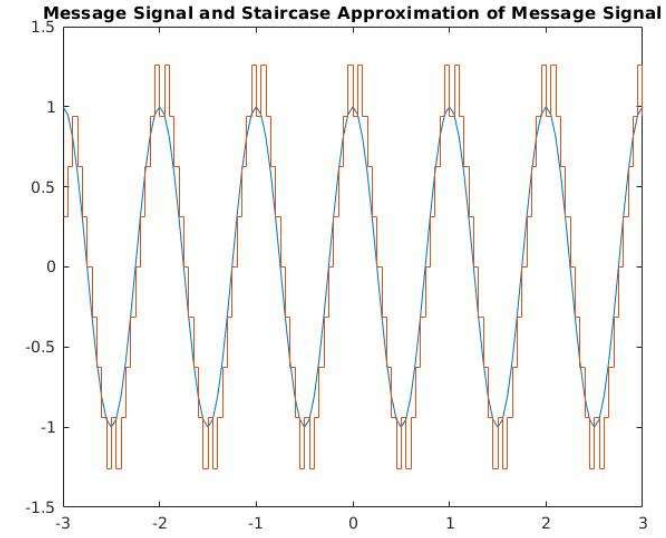
```
y = filter(b,a,x);
```

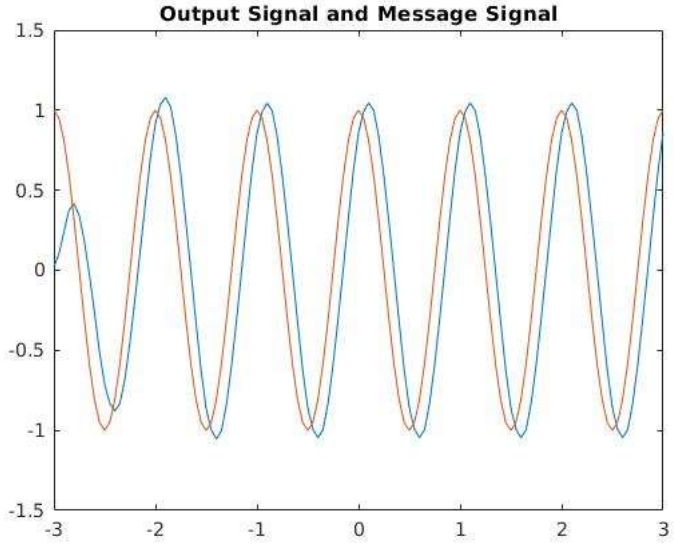
```
plot(t,2*y,t,m);
```

```
title('Output Signal and Message Signal');
```

Output Waveforms:




Message Signal and Staircase Approximation of Message Signal
1.5
1
0.5
0
-0.5
-1
-1.5
-3
-2
-1
0
1
2
3


Output Signal and Message Signal
1.5
1
0.5
0
-0.5
-1
-1.5
-3
-2
-1
0
1
2
3

Conclusion:

Delta Modulation technique has been applied on given message signal. The simulation was done using MATLAB.

Delta Demodulation was done using low pass band filter available in MATLAB.

The graphs formed correctly matches the expected results with some changes done in the filter.

Remarks: **Signature:**

EXPERIMENT 11: FREQUENCY MODULATION IN MATLAB

Date: 28/06/2020

Aim:

To implement frequency modulation and demodulation using MATLAB.

Apparatus: MATLAB Software

Theory:

When the angle of the carrier wave is varied in some manner with respect to modulating signal $m(t)$, the technique of modulation is known as angle modulation or exponential modulation.

General form, $s(t) = A\cos(\omega_c t + \varphi(t))$, where $s(t)$ is the modulated signal, ω_c is the carrier frequency in radians and $\varphi(t)$ is the phase function which is time varying and captures the information, which you want to convey.

So, modulating signal $m(t)$ somehow modifies this $\varphi(t)$

Let $\theta_i(t) = \omega_c t + \varphi(t)$ be the instantaneous phase of the carrier signal, then instantaneous frequency can be obtained by taking the derivative of instantaneous phase w.r.t. time, t.

$\omega_i(t) = \frac{d\theta_i(t)}{dt} = \omega_c + \frac{d\varphi(t)}{dt}$ is the instantaneous frequency of the carrier wave or modulated signal

Where, $\varphi(t)$ is the instantaneous phase deviation and $\frac{d\varphi(t)}{dt}$ is the instantaneous frequency deviation.

Phase Modulation: (PM)

The instantaneous phase deviation carries the information or $\varphi(t)$ is varied linearly with $m(t)$

$$\varphi(t) \quad \alpha \quad m(t)$$

$$\varphi(t) \quad = \quad k_p\, m(t)$$

Where, k_p is the phase modulation constant/ phase sensitivity of the modulator measured in radians/volt.

For PM, instantaneous phase $\theta_i(t) = \omega_c t + k_p\, m(t)$

And instantaneous frequency $\omega_i(t) = \omega_c + k_p \frac{dm(t)}{dt}$

Modulated signal for PM, $s(t) = A\cos(\omega_c t + k_p\, m(t))$

Frequency Modulation: (FM)

The instantaneous frequency carries the information or message signal.

$$\frac{d\varphi(t)}{dt} \ \alpha \ \ m(t)$$

$$\frac{d\varphi(t)}{dt} = k_f \ m(t)$$

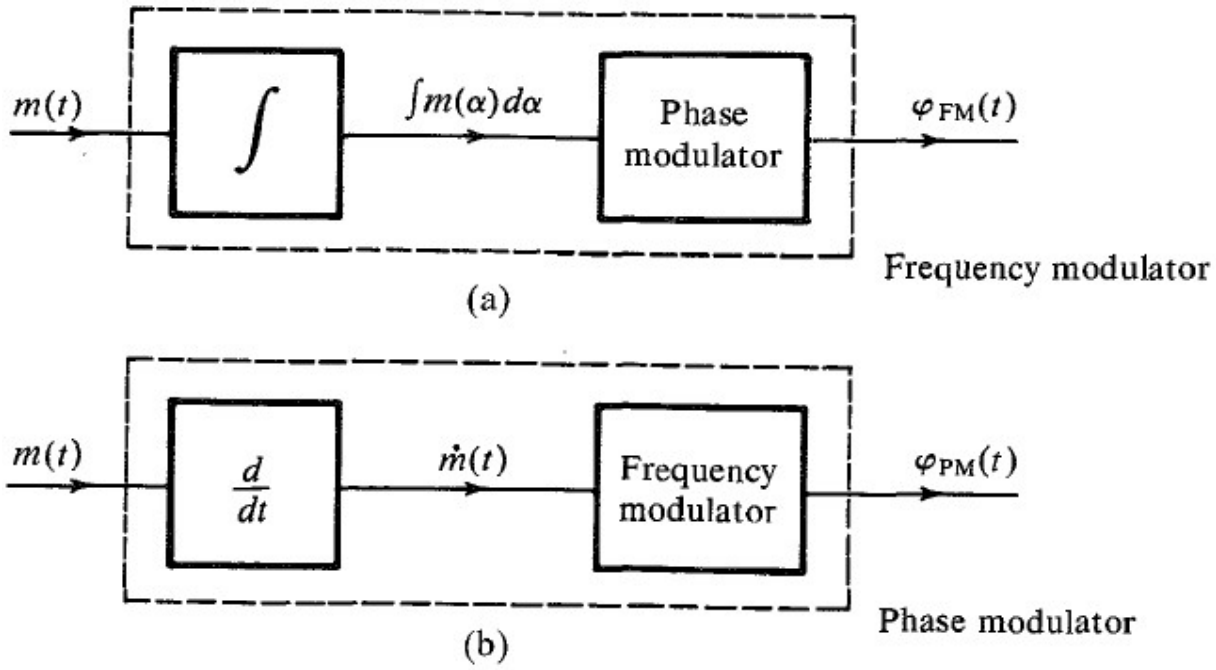
Where, k_f is the frequency modulation constant/ frequency sensitivity of the modulator measured in radians/(seconds*volt) or Hz/volt.

For FM, instantaneous phase deviation, $\varphi(t) = k_f \int_{-\infty}^{t} m(\alpha)d\alpha$

Hence, instantaneous phase, $\theta_i(t) = \omega_c t + k_f \int_{-\infty}^{t} m(\alpha)d\alpha$

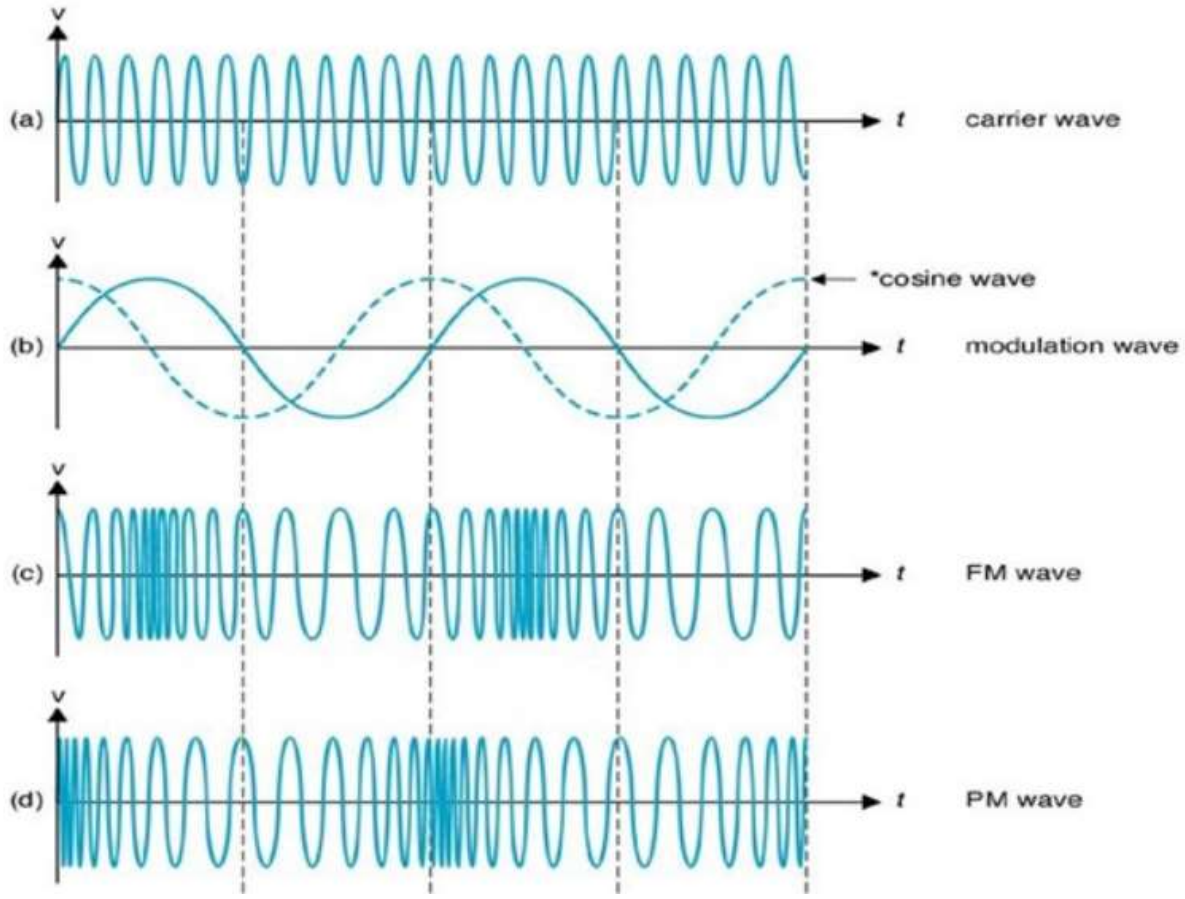
And instantaneous frequency, $\omega_i(t) = \omega_c + k_f \ m(t)$

Modulated Signal for FM, $s(t) = A\cos(\omega_c t + k_f \int_{-\infty}^{t} m(\alpha)d\alpha)$



Phase and frequency modulation are inseparable.

Considering, message signal as sine wave, the derivative of it will be cosine and respective FM and PM wave are shown in the next figure.



or else if we consider $m(t) = A_m \cos \omega_m t$, the instantaneous phase deviation for PM and FM will be;

PM,

$$\varphi(t) \quad = \quad k_p \, m(t) = k_p A_m \cos \omega_m t$$

FM,

$$\varphi(t) = k_f \int_{-\infty}^{t} m(\alpha) d\alpha = k_f \int_{-\infty}^{t} A_m \cos \omega_m \alpha \, d\alpha = \frac{k_f A_m \ \sin \omega_m t}{\omega_m}$$

Hence, the modulated signal for PM and FM become

$$\boldsymbol{s(t) = A \cos\left(\omega_c t + k_p \, m(t)\right) = A \cos\left(\omega_c t + k_p \, A_m \cos \omega_m t\right) = A \cos\left(\omega_c t + \beta_{pm} \cos \omega_m t\right)}$$

Where, $\beta_{pm} = \ k_p \ A_m$ is the modulation index for PM

$$s(t) = A\cos(\omega_c t + k_f \int_{-\infty}^{t} m(\alpha)d\alpha)$$

$$= A\cos\left(\omega_c t + \frac{k_f A_m \sin\omega_m t}{\omega_m}\right) = A\cos\left(\omega_c t + \beta_{fm}\sin\omega_m t\right)$$

Where, $\beta_{fm} = \frac{k_f A_m}{\omega_m} = \frac{\Delta f}{f_m}$ is the modulation index for FM

$\Delta f = k_f A_m$, is the peak frequency deviation

Frequency Deviation:

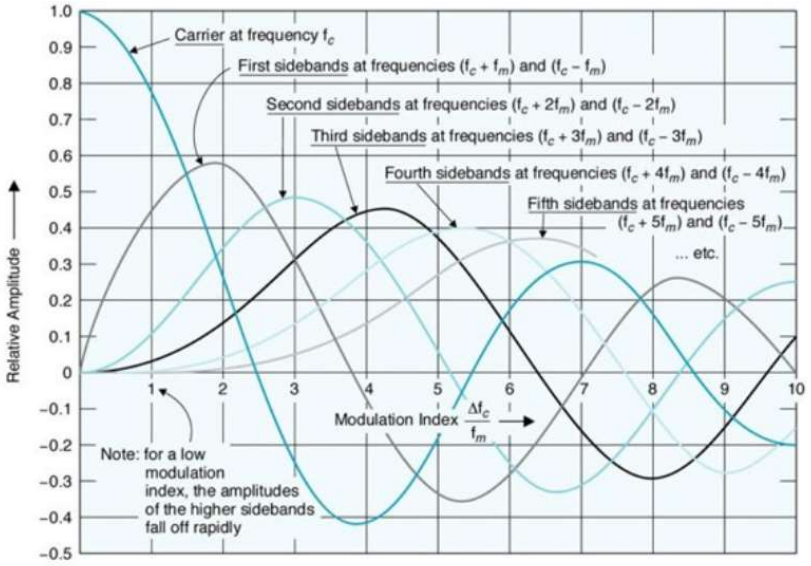
- The amount of change in the carrier frequency produced, by the amplitude of the input modulating signal, is called **frequency deviation**.
- The Carrier frequency swings between fmax and fmin as the input varies in its amplitude.
- The difference between fmax and fc is known as frequency deviation. Δf = fmax – fc
- Similarly, the difference between fc and fmin also is known as frequency deviation. Δf = fc –fmin

The frequency modulated signal can also be expressed in terms of Bessel function by taking the fourier series expansion.

Reference: Modern Digital and Analog Communication by B.P.Lathi

$$s(t)_{FM} = A\sum_{n=-\infty}^{\infty} J_n(\beta)\cos\big((\omega_c + n\omega_m)\big)t$$

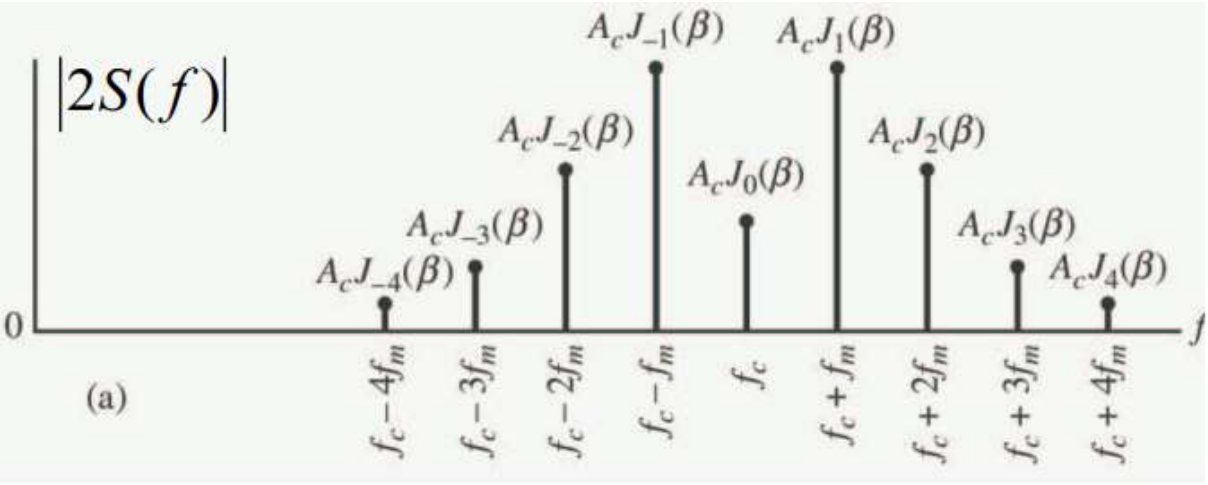
Where, $J_n(\beta)$ is Bessel function of n order with argument β.



By taking the fourier transform of the above equation, we get

$$S(f) = \frac{A}{2}\sum_{n=-\infty}^{\infty} J_n(\beta)\left[\delta(f-(f_c+nf_m)) + \delta(f+(f_c+nf_m)\right]$$

Theoretically, single tone FM has carrier component with infinite number of side-bands at $(f_c \pm nf_m)$ and its bandwidth is infinite.



Bandwidth of FM:

As the value of n increase, the significant power level in sideband component decreases

i.e. $lim_{n\to\infty} J_n(\beta) = 0$.

So we consider only k sidebands on either sides of f_c and look at these 2k+1 components i.e. from $(f_c + kf_m)$ to $(f_c - kf_m)$ and if we consider the power ratio,

Power ratio = $\frac{\frac{1}{2}A_c^2\sum_{k=-\infty}^{\infty} J_n(\beta)^2}{\frac{1}{2}A_c^2} = \frac{sideband\ power}{carrier\ power} = J_0(\beta)^2 + 2\sum_{n=1}^{k} J_n(\beta)^2$

$(since, J_0(\beta)$ is the carrier related component and $J_n(\beta)$ is an even function)

The bandwidth is defined as band of frequencies over which signal contains 98% of its power that means the power ratio is 0.98.

In general, $BW = 2kf_m$

In empirical sense, if we choose $k = \beta + 1$, then power ratio turns to 0.98. (depends on the Bessel function values)

Hence, $BW = 2(\beta+1)f_m = 2(\beta f_m + f_m) = 2(\Delta f + f_m)$ $[since\ \beta = \frac{\Delta f}{f_m}]$

Frequency Demodulation:

We know that the equation of FM wave is

$$s(t) = A_c \cos\left(2\pi f_c t + 2\pi k_f \int m(t)\, dt\right)$$

Differentiate the above equation with respect to '**t**'.

$$\frac{ds(t)}{dt} = -A_c\left(2\pi f_c + 2\pi k_f m(t)\right) \sin\left(2\pi f_c t + 2\pi k_f \int m(t)\, dt\right)$$

We can write,

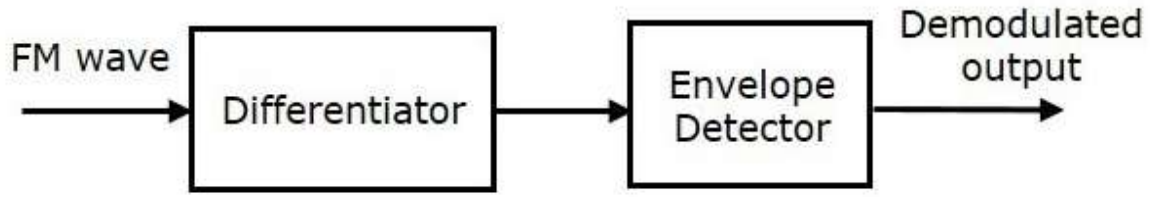
$-\sin\theta$ as $\sin\left(\theta - 180^0\right)$.

$$\Rightarrow \frac{ds(t)}{dt} = A_c\left(2\pi f_c + 2\pi k_f m(t)\right) \sin\left(2\pi f_c t + 2\pi k_f \int m(t)\, dt - 180^0\right)$$

$$\Rightarrow \frac{ds(t)}{dt} = A_c\left(2\pi f_c\right)\left[1 + \left(\frac{k_f}{k_c}\right) m(t)\right] \sin\left(2\pi f_c t + 2\pi k_f \int m(t)\, dt - 180^0\right)$$

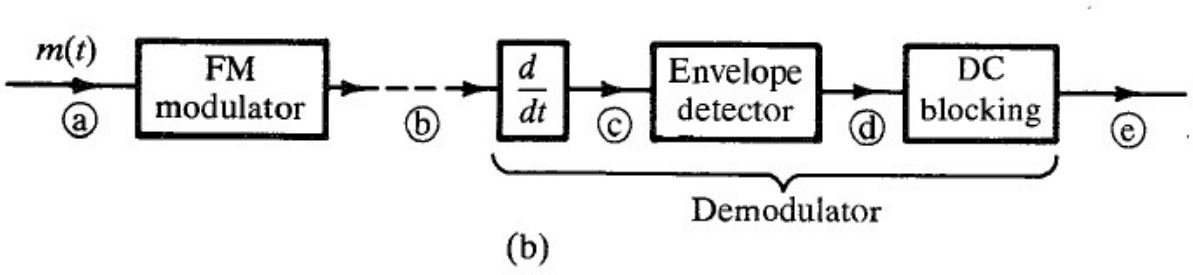
In the above equation, the amplitude term resembles the envelope of AM wave and the angle term resembles the angle of FM wave. Here, our requirement is the modulating signal m(t)m(t). Hence, we can recover it from the envelope of AM wave.

The following figure shows the block diagram of FM demodulator using frequency discrimination method.



This block diagram consists of the differentiator and the envelope detector. Differentiator is used to convert the FM wave into a combination of AM wave and FM wave. This means, it converts the frequency variations of FM wave into the corresponding voltage (amplitude) variations of AM wave. We know the operation of the envelope detector. It produces the demodulated output of AM wave, which is nothing but the modulating signal.

The FM communication link is shown below:



(b)

Algorithm:

- Define the sampling frequency $f_s > 2(f_c + k_{max}f_m)$ say, $f_s = 100Hz$;
- Define the time range using the sampling frequency t= -10 : $1/f_s$: 10
- Consider, message signal, $m = A_m \cos \omega_m t$, where $f_m = 1Hz$ and carrier signal $c = A_c \cos \omega_c t$ where $f_c = 10Hz$. Keep the amplitude of message and carrier signal same.
- Assume $k_p = 1$ and $k_f = 2\pi f_m$
- For PM, $\boldsymbol{s = A\cos\left(\omega_c t + k_p\, m(t)\right) = A\cos\left(\omega_c t + k_p\, A_m \cos \omega_m t\right)}$
- For FM, $\boldsymbol{s(t) = A\cos(\omega_c t + k_f \int_{-\infty}^{t} m(\alpha)d\alpha) = A\cos\left(\omega_c t + \frac{k_f A_m \sin \omega_m t}{\omega_m}\right)}$
  Use integral function in MATLAB in general so it can be done for any input signal.
- Figure1: Plot input signal, carrier signal, FM signal, derivative of input signal and PM signal using subplot(511) to subplot(515);
  Use diff function in matlab for the derivative of the signal
  For example, y=diff(x), pad the last value of the signal so as to ensure the same size because difference values will be one less than the given values.
  y=[diff(x) diff(end)]
- Figure 2: plot the frequency spectrum of FM and PM signals using the commands fft and fftshift.
  - Define n=length(t) %gives no. of columns in t
  - Define the step size for frequency axis fp which should be of same size as that of t. (matrix dimensions must match for plotting).
    - df=fs/n; where fs is the sampling frequency
  - Define frequency axis fp = -fs/2:df:fs/2-df ($\pm$df for getting same size matrices)
  - Take the fourier transform of FM and PM using fft command
  - Y = fft(x) returns the discrete Fourier transform (DFT) of vector x, computed with a fast Fourier transform (FFT) algorithm.
  - Y = fftshift(X) rearranges the outputs of fft by moving the zero-frequency component to the center of the array. It is useful for visualizing a Fourier transform with the zero-frequency component in the middle of the spectrum.
  - Can write in a single syntax as y=fftshift(fft(x));

- Plot the frequency spectrum of FM and PM using the command plot(fp,y).Use subplot(211) to (212).

- **Demodulate the FM signal**
  - Take the derivative of FM signal using diff function of MATLAB as mentioned in one of the previous step.
  - Multiply the above signal with carrier signal, this will look like AM wave.
    - Do .* element wise multiplication
- Do the low pass filtering of the above signal using butter and filter command.
  - [b,a] = butter(n,Wn,'ftype'). **This creates a filter**
  - [b,a] = butter(n,Wn) designs an order n lowpass digital Butterworth filter with normalized cutoff frequency Wn. It returns the filter coefficients in length n+1 row vectors b and a, with coefficients in descending powers of z.

$$H(z) = \frac{b(1) + b(2)z^{-1} + \ldots + b(n+1)z^{-n}}{1 + a(2)z^{-1} + \ldots + a(n+1)z^{-n}}$$

  - where the string 'ftype' is one of the following:
    - 'high' for a highpass digital filter with normalized cutoff frequency Wn
    - 'low' for a lowpass digital filter with normalized cutoff frequency Wn
    - 'stop' for an order 2*n bandstop digital filter if Wn is a two-element vector, Wn = [w1 w2]. The stopband is w1 < ω < w2.
    - 'bandpass' for an order 2*n bandpass filter if Wn is a two-element vector, Wn = [w1 w2]. The passband is w1 < ω < w2. Specifying a two-element vector, Wn, without an explicit 'ftype' defaults to a bandpass filter.
    - Cutoff frequency, Wn is that frequency where the magnitude response of the filter is . For butter, the normalized cutoff frequency Wn must be a number between 0 and 1, where 1 corresponds to the Nyquist frequency, π radians per sample.
    - Choose order, n=4 ; Wn=fm/fs; ftype='low'
- Use the filter command to apply the filter
  - y = filter(b,a,X), where b and a are the coefficients obtained in the previous step and X will the multiplied signal output (derivative of FM signal*Carrier Signal). y is the filtered output. Adjust order so that you get approximate input signal.
- Figure 3: Plots for FM demodulation
  - Include three figures using subplot(311) to subplot(313)
    - 1st – derivative of FM signal
    - 2nd - multiplied signal output (derivative of FM signal*Carrier Signal)
    - 3rd – filtered output y. Include message signal as well in this
      - Example: plot(t,m,t,y)

Expected Output waveforms:

Figure 1:

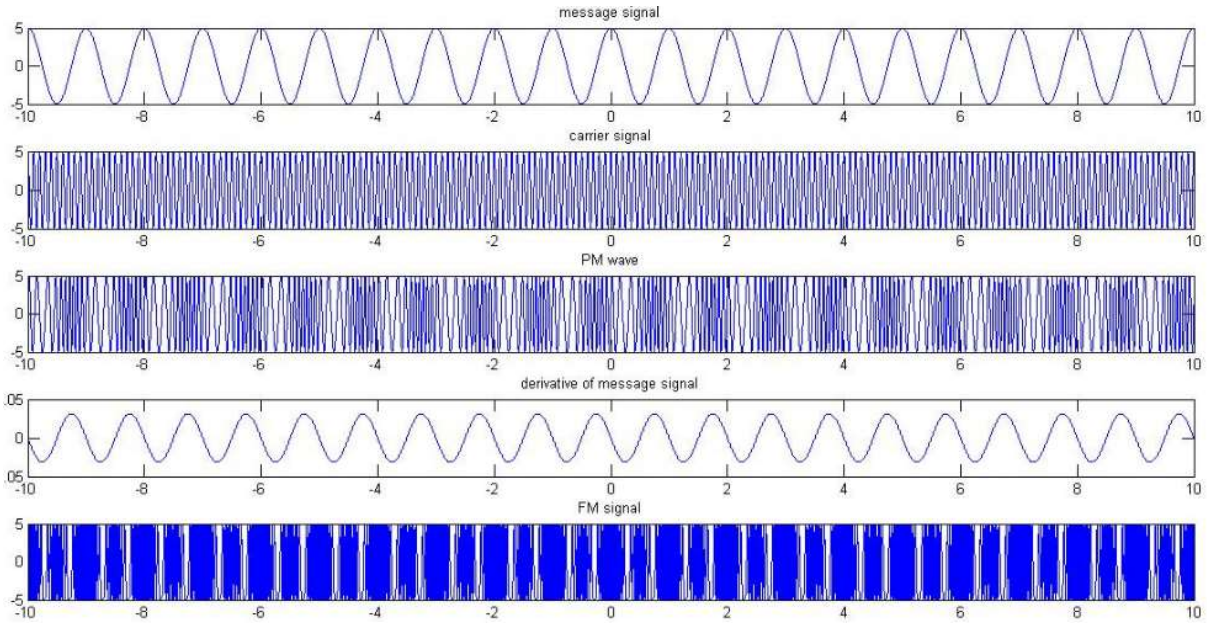


Figure 2:

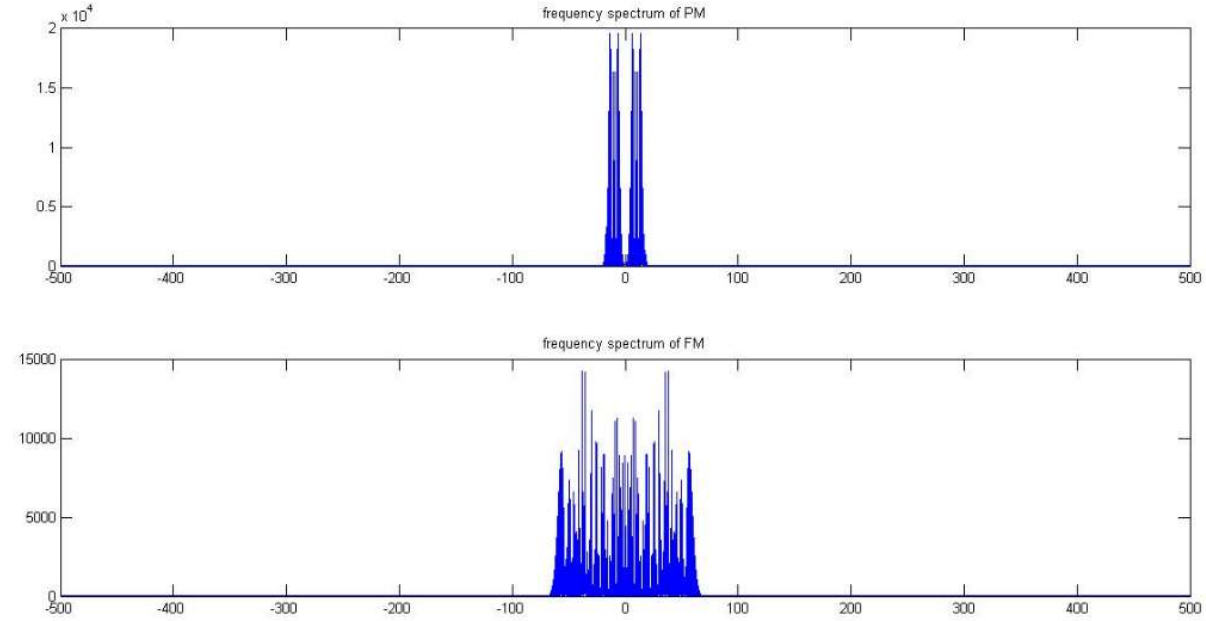
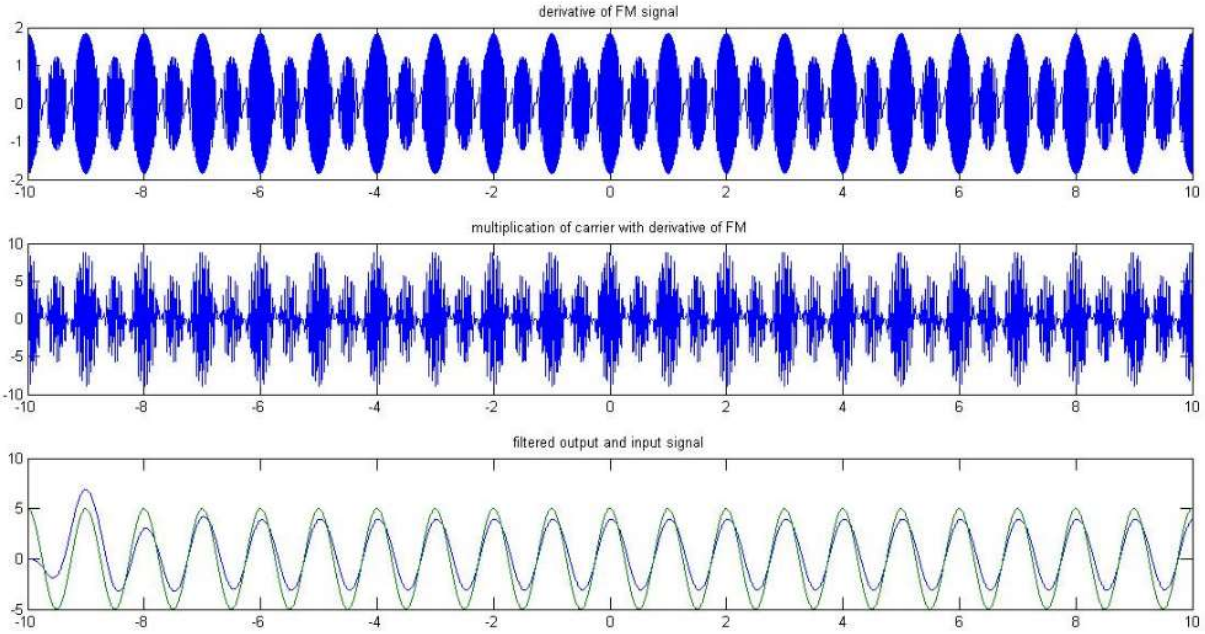


Figure 3:



Code:

```
% Experiment : Frequency Modulation and Demodulation

% Author: Krunal Rank

% RollNo. : U18CO081


%% Part 1

clc;

close all;

clear all;


f_s = 100;
```

```
t = -10:1/f_s:10;


A_m = 5;

A_c = A_m;

f_m = 1;

f_c = 10;


w_m = 2 * pi * f_m;

w_c = 2 * pi * f_c;


m = A_m*cos(w_m * t);

c = A_c*cos(w_c * t);

d_m = diff(m);

d_m = [d_m d_m(end)];


syms x;

m_x = @(x) A_m*cos(w_m*x);


k_p = 1;
```

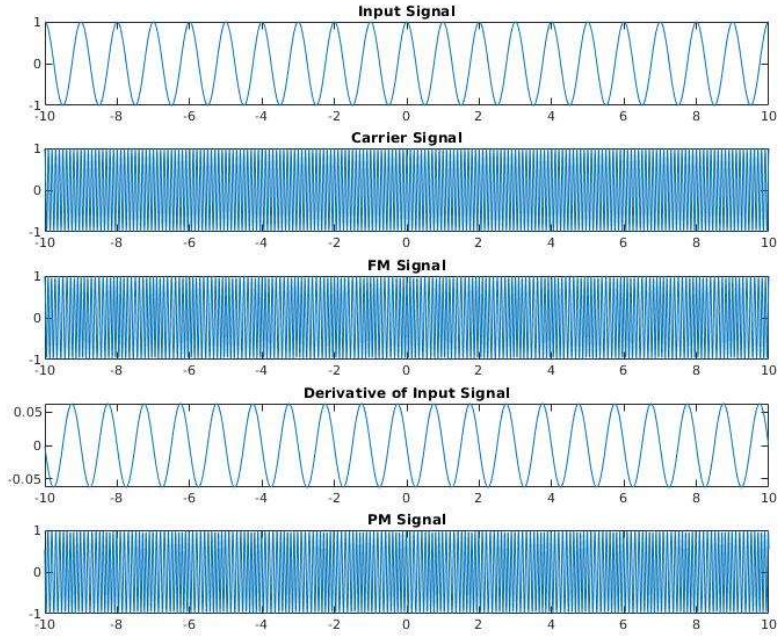
```
k_f = 2 * pi * f_m;


s_p = A_c * cos(w_c*t + k_p*m);

s_f = zeros(1,length(t));

for i = 1: length(t)

    s_f(i) = A_c * cos(w_c*t(i) + integral(m_x,-10,t(i)));

end


subplot(5,1,1);

plot(t,m);

title('Input Signal');


subplot(5,1,2);

plot(t,c);

title('Carrier Signal');


subplot(5,1,3);

plot(t,s_f);

title('FM Signal');
```

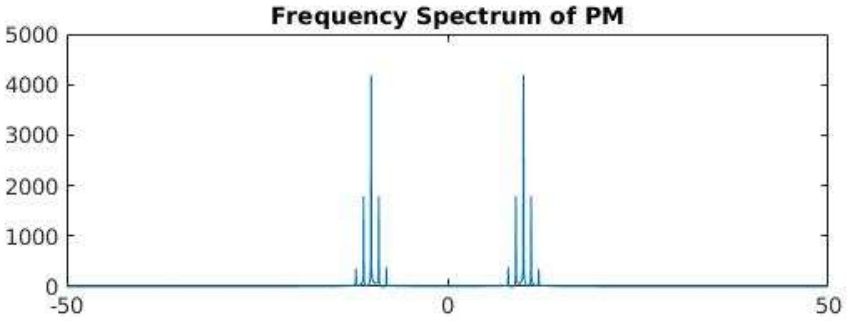
```
subplot(5,1,4);
plot(t,d_m);
title('Derivative of Input Signal');

subplot(5,1,5);
plot(t,s_p);
title('PM Signal');

%% Part 2

n = length(t);
df = f_s/n;
f_p = -f_s/2:df:(f_s/2 - df);

y_p = fftshift(fft(s_p));
y_f = fftshift(fft(s_f));

subplot(2,1,1);
plot(f_p,abs(y_p));
title('Frequency Spectrum of PM');
```

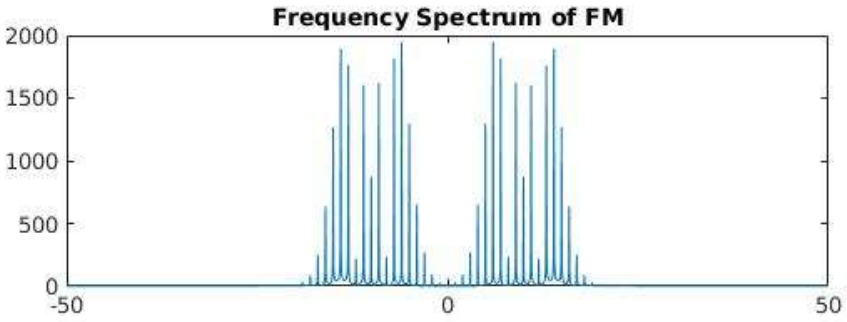
```
subplot(2,1,2);

plot(f_p,abs(y_f));

title('Frequency Spectrum of FM');


%% Part 3


d_s_f = diff(s_f);

d_s_f = [d_s_f d_s_f(end)];


y_1 = d_s_f.*c;


[b,a] = butter(2,f_c/f_s,'low');


y_2 = filter(b,a,y_1);




subplot(3,1,1);

plot(t,d_s_f);

title('Derivative of FM Signal');
```

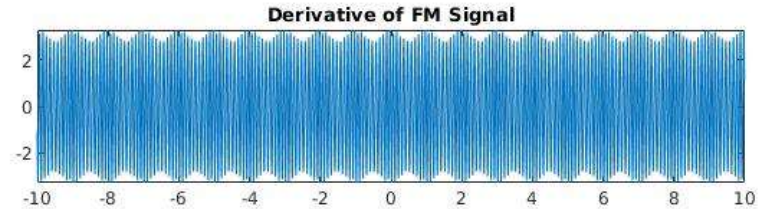
```
subplot(3,1,2);

plot(t,y_1);

title('Multiplied FM Signal with Carrier Signal');


subplot(3,1,3);

plot(t,y_2,t,m);

title('Filtered Ouput and Input Signal');
```

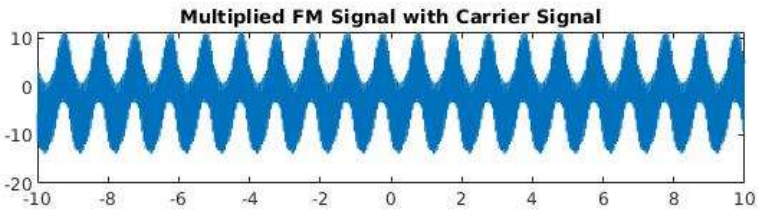
Output Waveforms:

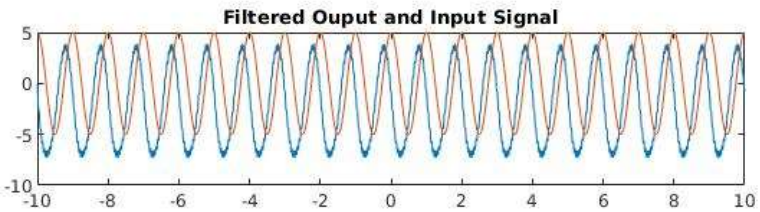



Frequency Spectrum of PM
5000
4000
3000
2000
1000
0
-50
0
50


Frequency Spectrum of FM
2000
1500
1000
500
0
-50
0
50


Derivative of FM Signal
2
0
-2
-10
-8
-6
-4
-2
0
2
4
6
8
10


Multiplied FM Signal with Carrier Signal
10
0
-10
-20
-10
-8
-6
-4
-2
0
2
4
6
8
10


Filtered Ouput and Input Signal
5
0
-5
-10
-10
-8
-6
-4
-2
0
2
4
6
8
10

Conclusion:

Frequency and Phase modulation along with their demodulation has been accomplished using MATLAB.

Their graphs has been studied and verified with the expected results, as attached in the above section.

Frequency and Phase Modulation are difficult to identify from their graphs.

Remarks: **Signature:**

EXPERIMENT 12: NUMERICAL APERTURE OF OPTICAL FIBER

Date: 28/06/2020

Aim:

To find the Numerical Aperture of given optical fiber.

Apparatus: Emitter, Fiber cable, Fiber stand, Detector

Theory:

What is optic fiber?

Optical fibers are fine transparent glass or plastic fibers which can propagate light. They work under the principle of total internal reflection from diametrically opposite walls. In this way light can be taken anywhere because fibers have enough flexibility. This property makes them suitable for data communication, design of fine endoscopes, micro sized microscopes etc. An optic fiber consists of a core that is surrounded by a cladding which is normally made of silica glass or plastic. The core transmits an optical signal while the cladding guides the light within the core. Since light is guided through the fiber it is sometimes called an optical wave guide. The basic construction of an optic fiber is shown in figure (1).

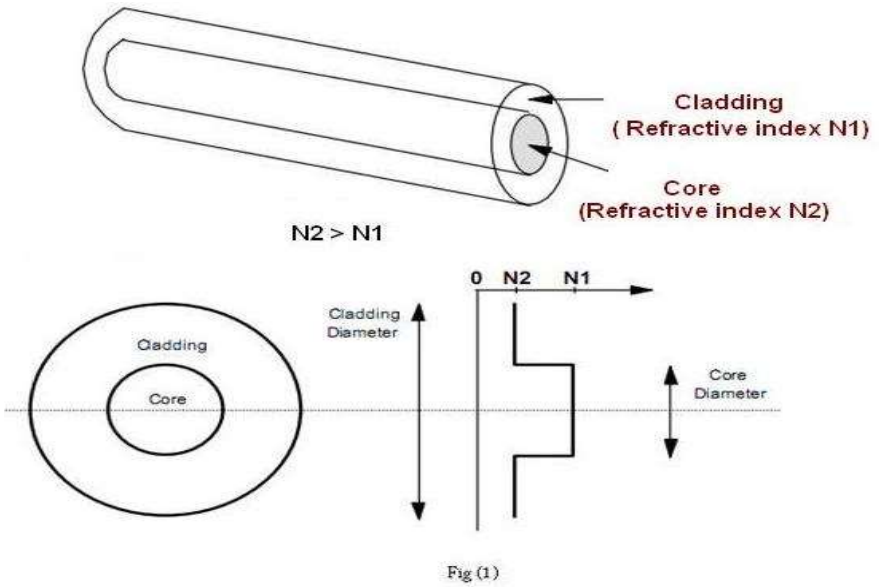


Fig (1)

In order to understand the propagation of light through an optical fiber, consider the figure (2). Consider a light ray (i) entering the core at a point A, travelling through the core until it reaches the core cladding boundary at point B. As long as the light ray intersects the core-cladding boundary at small angles, the ray will be reflected back in to the core to travel on to point C where the process of reflection is repeated .i.e., total internal reflection takes place. Total internal reflection occurs only when the angle of incidence is greater than the critical angle. If a ray enters

an optic fiber at a steep angle (ii), when this ray intersects the core-cladding boundary, the angle of intersection is too large. So, reflection back in to the core does not take place and the light ray is lost in the cladding. This means that to be guided through an optic fiber, a light ray must enter the core with an angle less than a particular angle called the acceptance angle of the fiber. A ray which enters the fiber with an angle greater than the acceptance angle will be lost in the cladding.

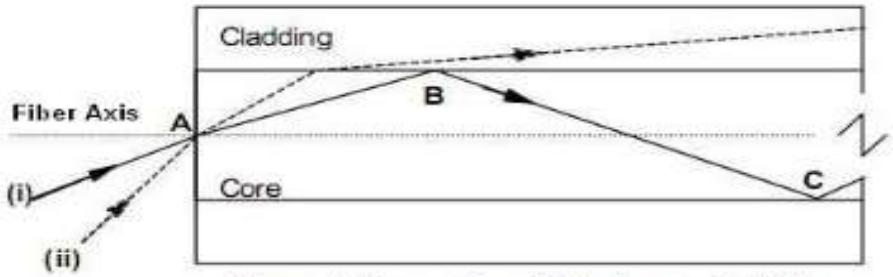


Figure 2 Propagation of light in an optical fibre

Consider an optical fibre having a core of refractive index n_1 and cladding of refractive index n_2. let the incident light makes an angle i with the core axis as shown in figure (3).

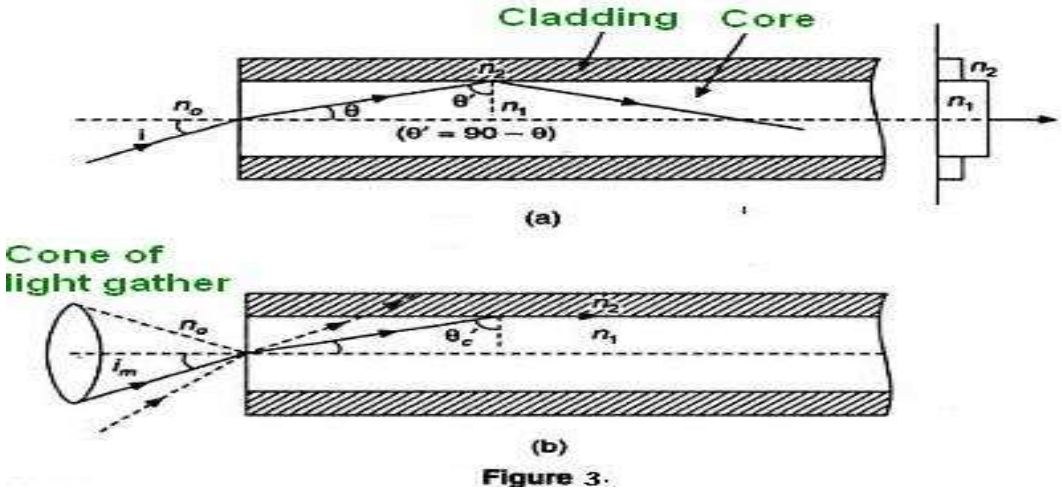


Figure 3.

Then the light gets refracted at an angle θ and fall on the core-cladding interface at an angle where,

$$\theta' = (90 - \theta) \quad \text{---------- (1)}$$

By Snell's law at the point of entrance of light in to the optical fiber we get,

$$n_0 \sin i = n_1 \sin \theta \quad \text{---------- (2)}$$

Where n_0 is refractive index of medium outside the fiber. For air n_0 =1.

When light travels from core to cladding it moves from denser to rarer medium and so it may be totally reflected back to the core medium if θ' exceeds the critical angle θ'_c. The critical angle is

that angle of incidence in denser medium (n1) for which angle of refraction become 90°. Using Snell's laws at core cladding interface,

$$n_1 \sin \theta'_c = n_2 \sin 90$$

or

$$Sin\, \theta'_c = \frac{n_2}{n_1} \quad \text{---------- (3)}$$

Therefore, for light to be propagated within the core of optical fiber as guided wave, the angle of incidence at core-cladding interface should be greater than θ'_c. As i increases, θ increases and so θ' decreases. Therefore, there is maximum value of angle of incidence beyond which, it does not propagate rather it is refracted in to cladding medium (fig: 3(b)). This maximum value of i say i_m is called maximum angle of acceptance and $n_0 \sin i_m$ is termed as the numerical aperture (NA). From equation(2),

$$NA = n_0 \sin i_m = n_1 \sin \theta$$

$$= n_1 \sin(90 - \theta_c)$$

$$Or\, NA = n_1\, Cos\, \theta'_c$$

$$= n_1 \sqrt{1 - Sin^2 \theta'_c}$$

$$\sin \theta'_c = \frac{n_2}{n_1}$$

From equation (2)

$$NA = n1 \sqrt{1 - \frac{n_2^2}{n_1^2}}$$

Therefore,

$$NA = \sqrt{n_1^2 - n_2^2}$$

The significance of NA is that light entering in the cone of semi vertical angle i_m only propagate through the fibre. The higher the value of i_m or NA more is the light collected for propagation in the fibre. Numerical aperture is thus considered as a light gathering capacity of an optical fibre. Numerical Aperture is defined as the Sine of half of the angle of fibre's light acceptance cone. i.e. NA= Sin θ_a where θ_a, is called acceptance cone angle.

Let the spot size of the beam at a distance d (distance between the fiber end and detector) as the radius of the spot(r). Then,

$$\sin\theta = \frac{r}{\sqrt{r^2 + d^2}} \quad \text{---------- (4)}$$

Procedure:

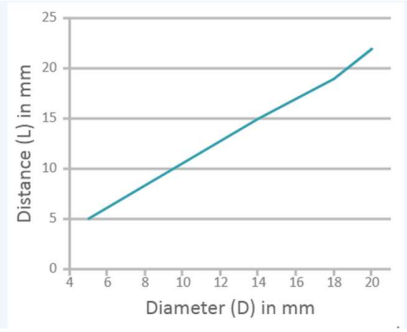
- Select all tools i.e. emitter, fiber, fiber stand, output screen by clicking on that.
- Now press start button.
- Vary the distance of screen (L) by scrolling the button. Diameter (D) will also vary. Note down that value.
- Repeat that process and get the value of L and D.

Fig.4 Tools arrangement

Observations:

DATA TABLE		
SR. NO	**Distance of screen (L) in mm**	**Diameter(D) in mm**
1	5	5.554
2	10	11.11
3	15	16.664
4	20	22.22

DRAW GRAPH:



Conclusion:

In this experiment, we found the numerical aperture of the given optical fibre by using virtual lab.

The observed value for the Numerical Aperture was 0.426 which was well within the experimental error of 10%.

Remarks: **Signature:**