Demodulation of BFSK Signal

Prepared in partial fulfillment of the course
“Digital Signal Processing”

Moosa Mahsoom
2016A8PS0310G
Pragnesh Patel
2016A3PS0183G
Devam Awasare
2016A3PS0657G

Under the guidance of
Dr. Sarang Dhongdi

April 2019

INDEX

Problem Statement:

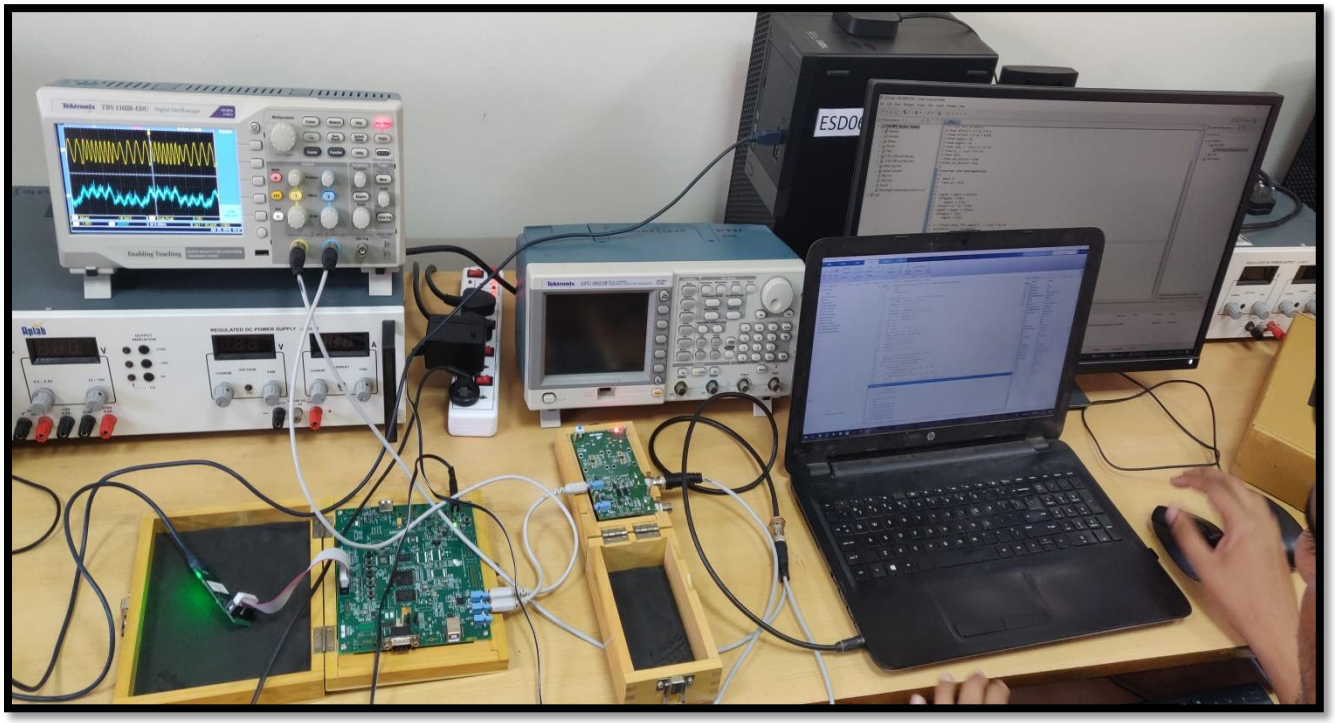
The goal of the project is to demodulate a modulated BFSK signal, first on MATLAB and then on 6748 DSK.

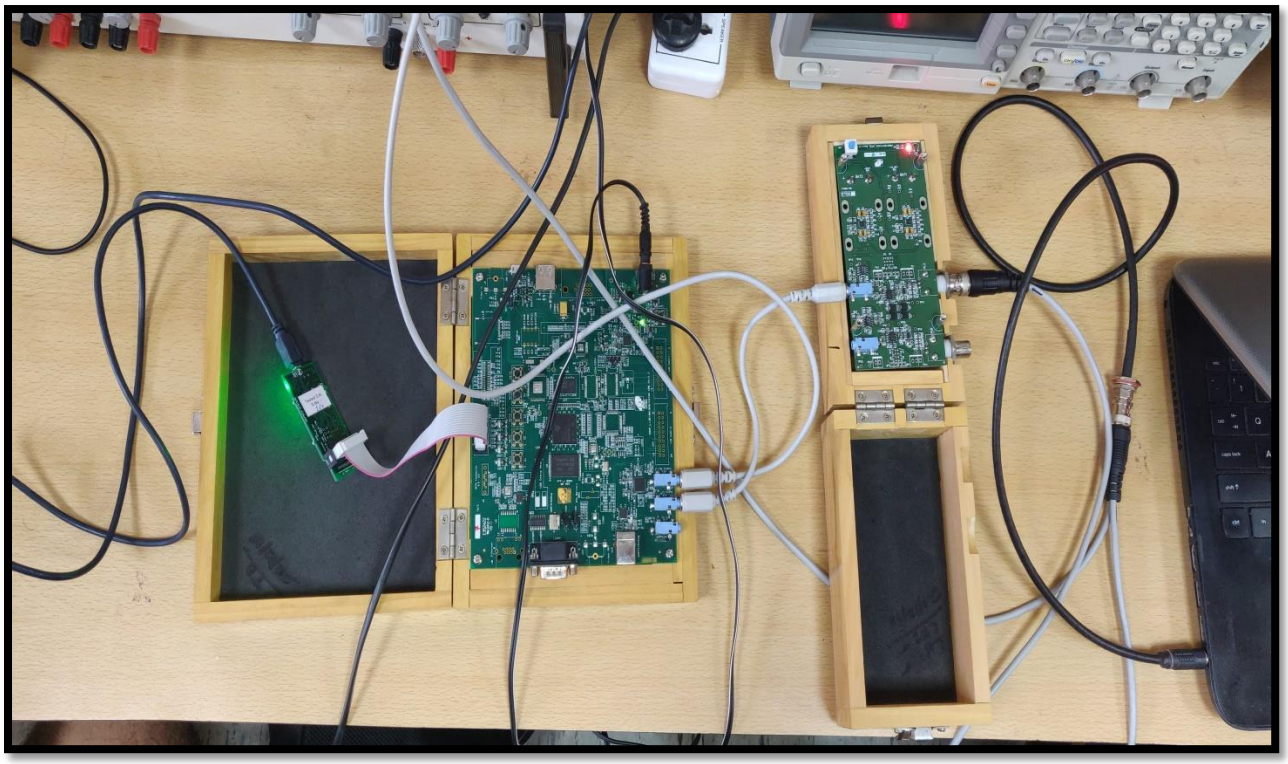
Assumptions:

We have assumed that -

- The modulated signal will be generated on MATLAB using soundsc() function
- The modulated signal from MATLAB will be fed as input to the DSK.
- In the demodulated signal, f1 = 400Hz => 0 and f2 = 800Hz => 1.
- If the output signal is oscillating with positive amplitude then the data transmitted is assumed to be 1 and if it is oscillating with negative amplitude, the data transmitted is assumed to be 0.

Apparatus Setup:





Modulate Signal generated from Matlab is sent to the Audio IN pin on the DSK via the Protection PCB through 3.5mm audio port of laptop.

The JTAG and power cable is connected to the DSK and the Protection PCB is switched on.

The Audio OUT pin on the DSK and audio port of laptop are connected to the Digital Oscilloscope to view the output and input respectively.

Method used:

In order to demodulate the BFSK modulated signal we multiplied our input wave with both the FSK frequencies (400Hz and 800Hz in our case) in parallel to get out_detect1 and out_detect2.

out_detect1= [A*sin(W0 + Ω)t] *sin(W0 + Ω)t = A[cos(0) – cos2(W0 + Ω)t]/2
= A[1– cos2(W0 + Ω)]/2
out_detect2 = [A*sin(W0 + Ω)t] *sin(W0 – Ω)t
= A[cos(2Ω)t – cos(2W0)t]/2

Then we subtract the resulting waves.

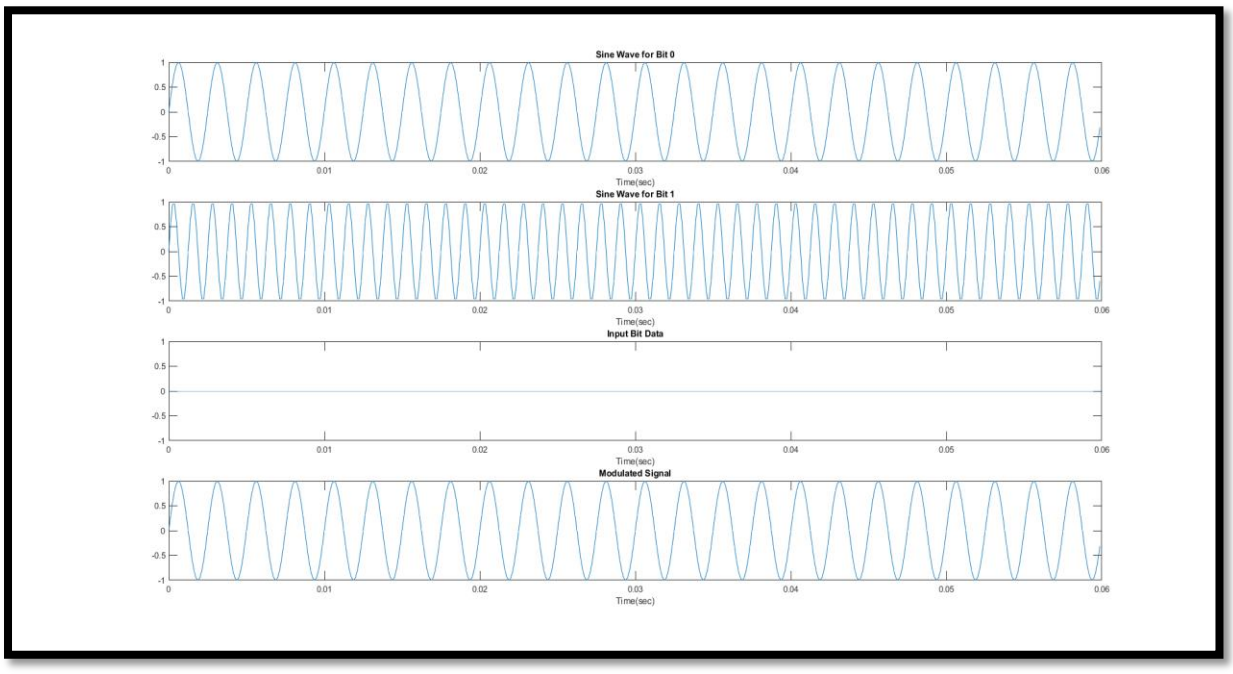
V(diff) = out_detect1 – out_detect2
= {A [1– cos2(W0 + Ω)t]/2} – { A[cos(2Ω)t –cos(2W0)t]/2}
= (A/2) – (A/2)[cos(2W0)t – cos(2Ω)t –cos2(W0 + Ω)t]

Then we use a Kaiser Low Pass Filter to filter out the high frequency components. This will only leave a DC component which can be equated to either 0 or 1.

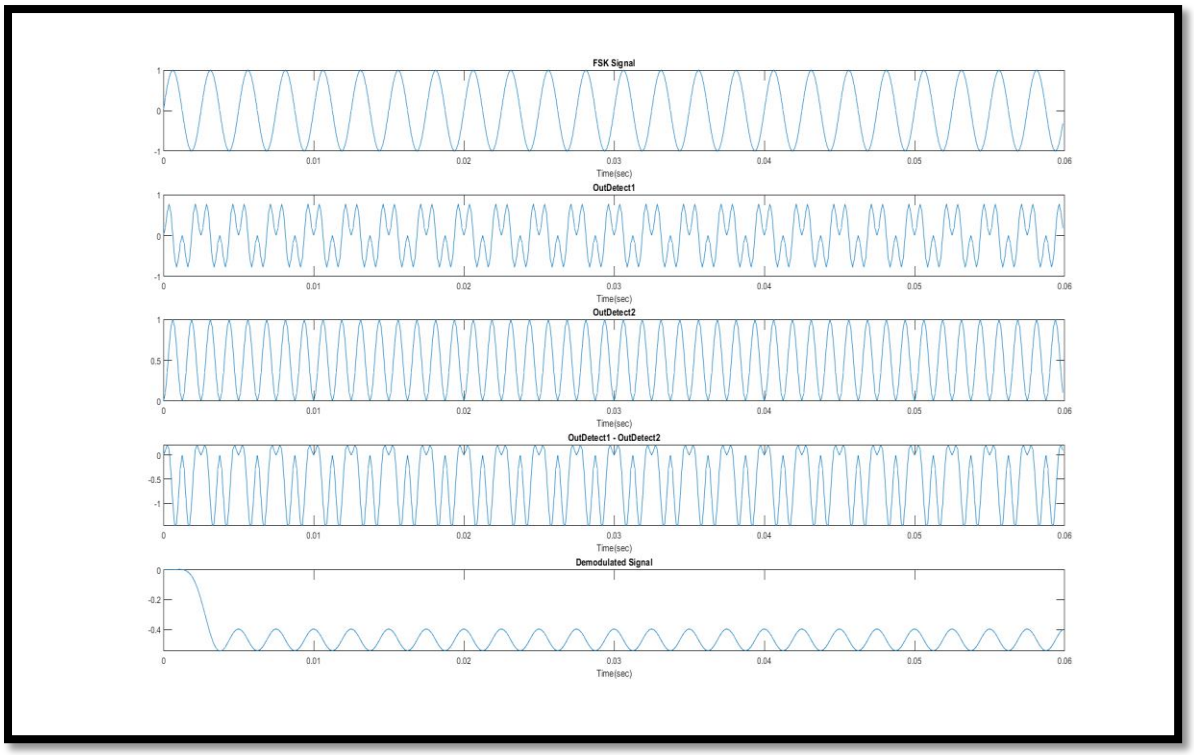
The order of the filter used is 30 and Beta is 5.
Coefficient file:

```
// kaiser.cof
// this file was generated using function L138_fir_coeffs.m

#define N 31

float h[N] = {
3.6711E-02,7.2800E-02,1.2026E-01,1.7918E-01,2.4894E-01,3.2820E-01,
4.1490E-01,5.0634E-01,5.9930E-01,6.9021E-01,7.7532E-01,8.5098E-01,
9.1381E-01,9.6092E-01,9.9011E-01,1.0000E+00,9.9011E-01,9.6092E-01,
9.1381E-01,8.5098E-01,7.7532E-01,6.9021E-01,5.9930E-01,5.0634E-01,
4.1490E-01,3.2820E-01,2.4894E-01,1.7918E-01,1.2026E-01,7.2800E-02,
3.6711E-02
};
```

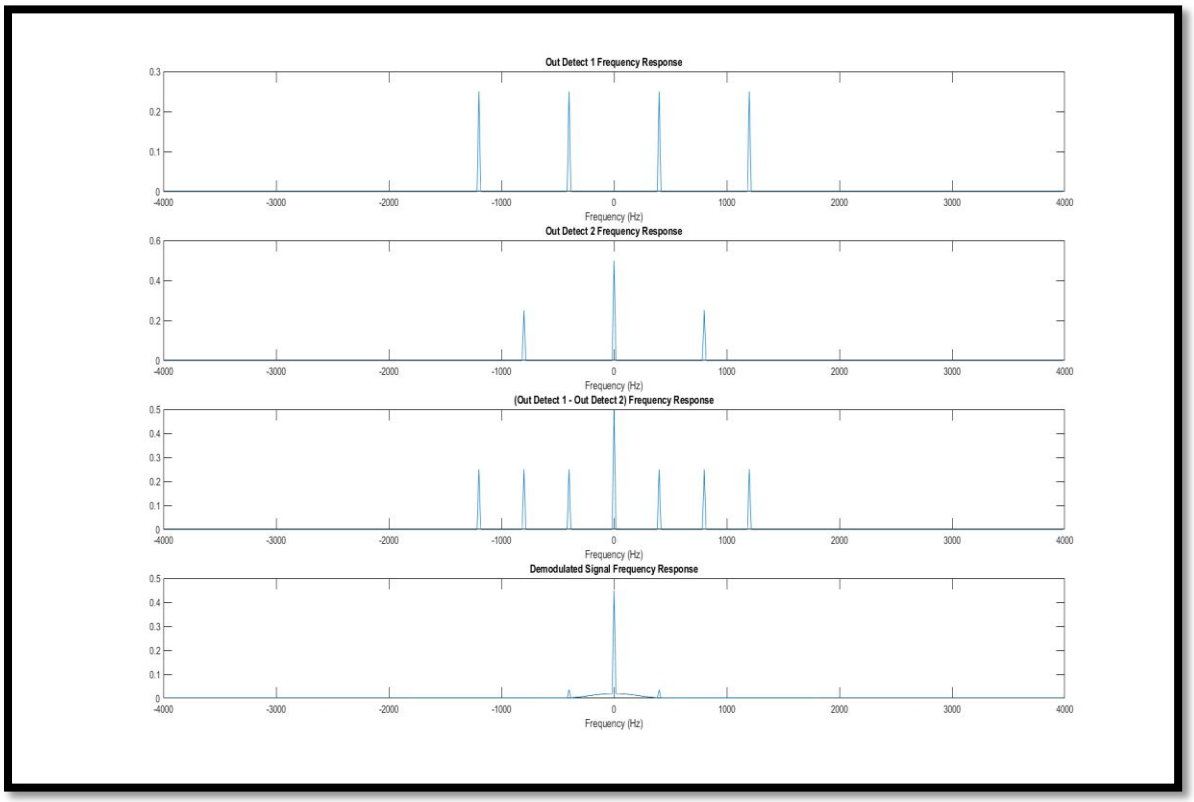
Matlab results:



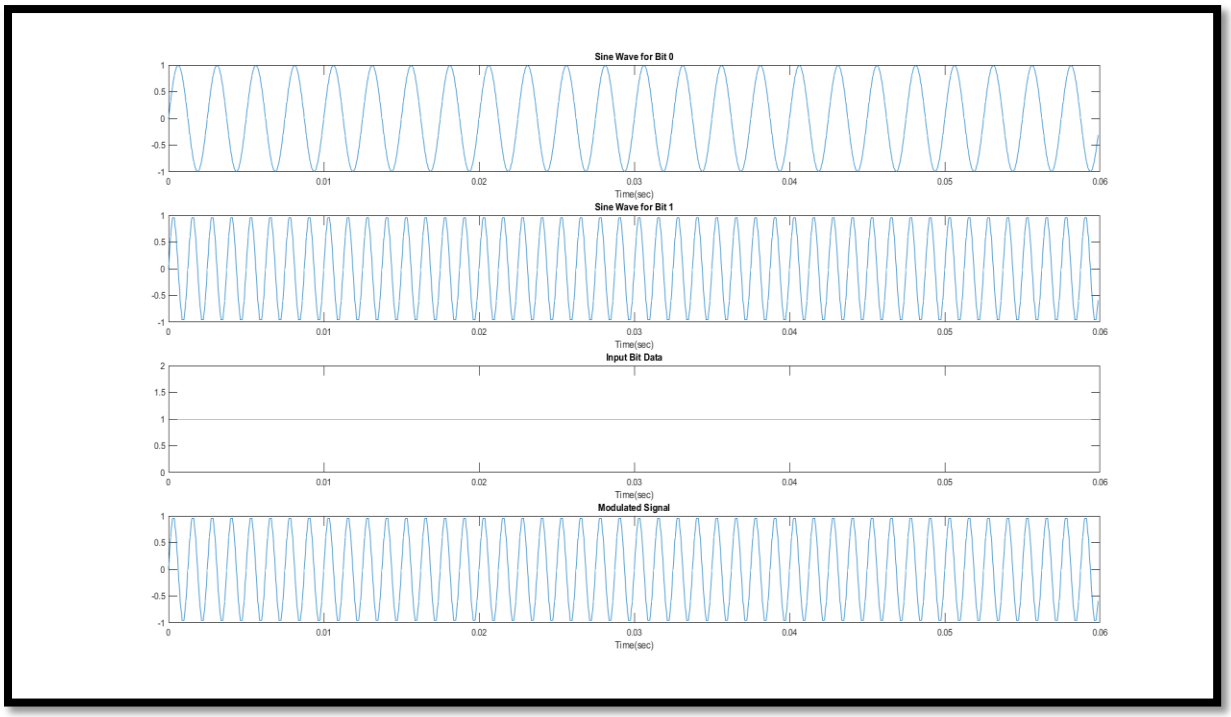
Bit 0 Modulation



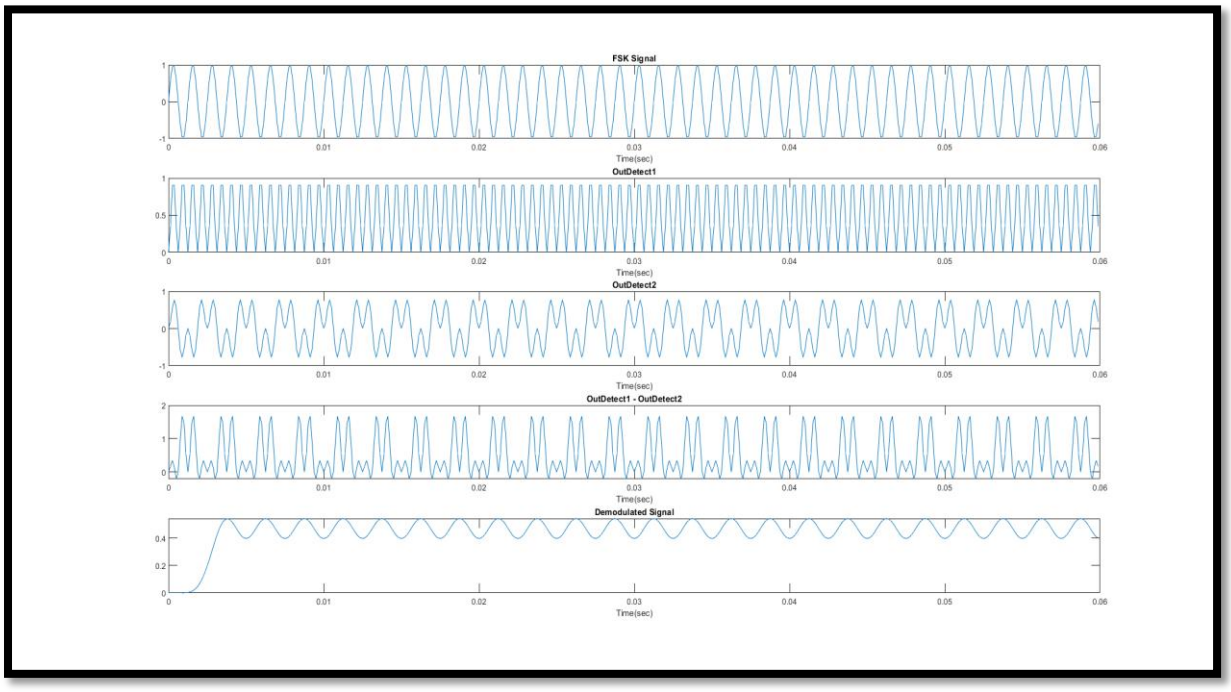
Bit 0 Demodulation



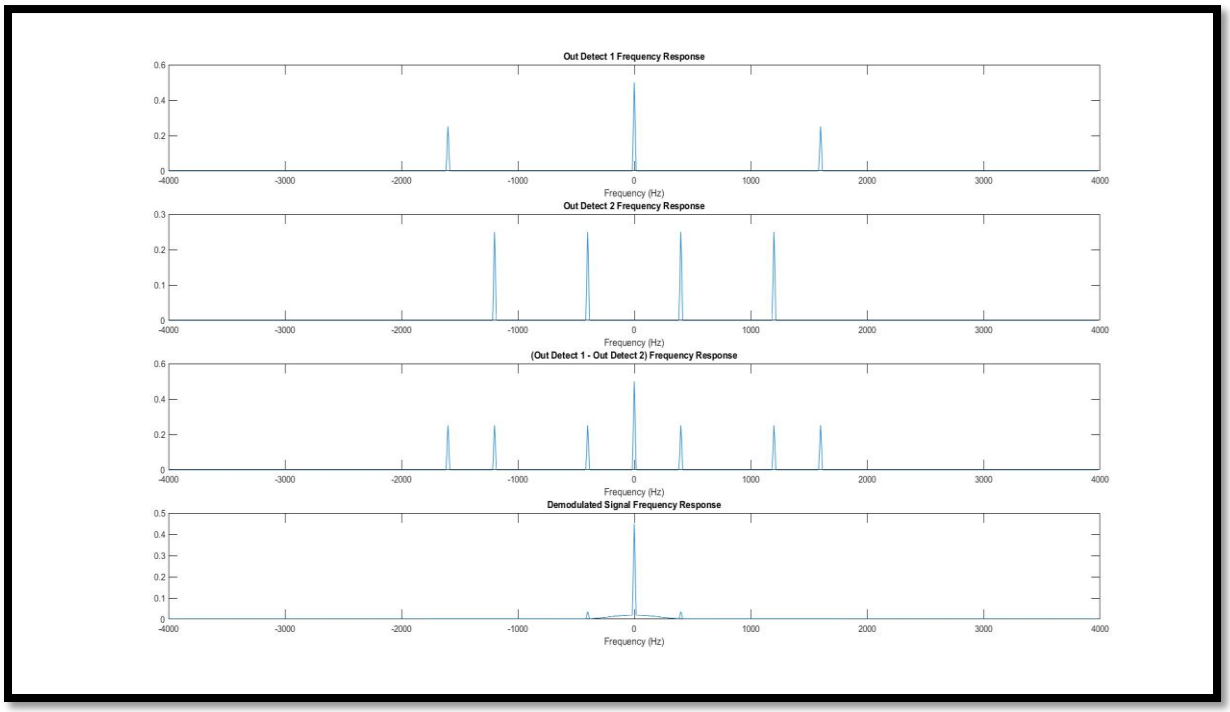
Bit 0 FFT



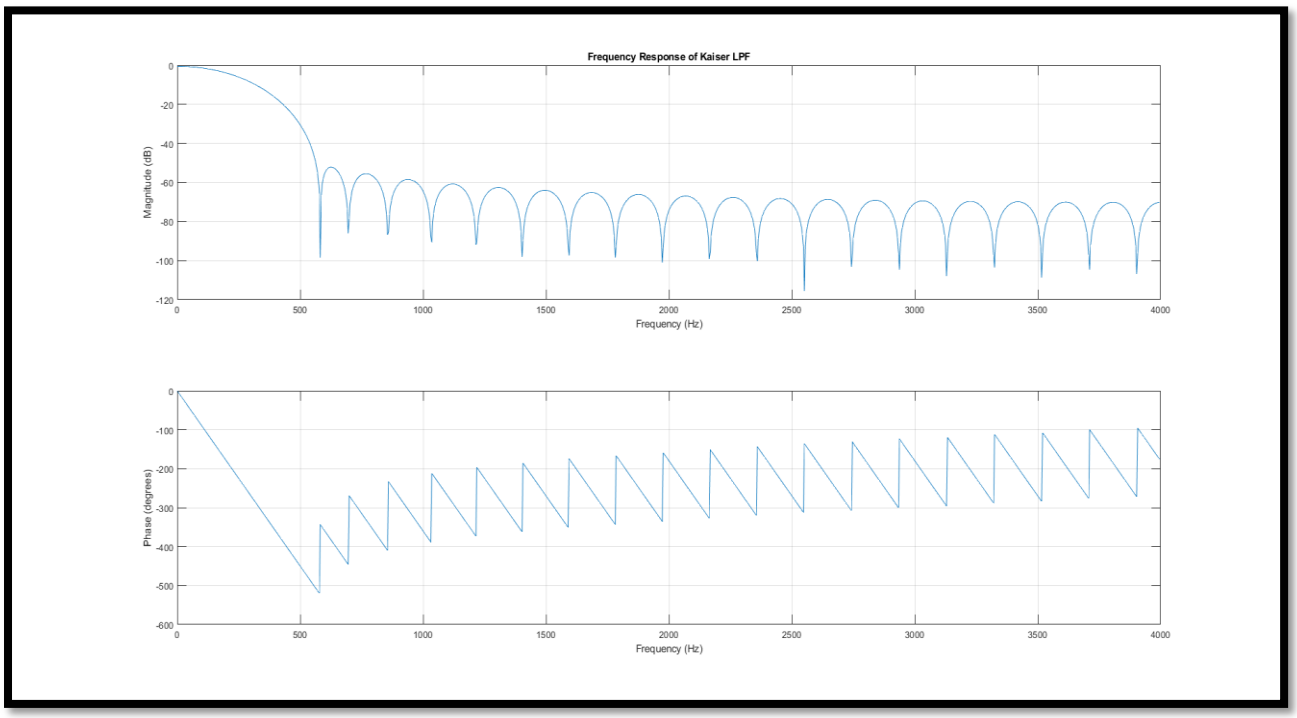
Bit 1 Modulation



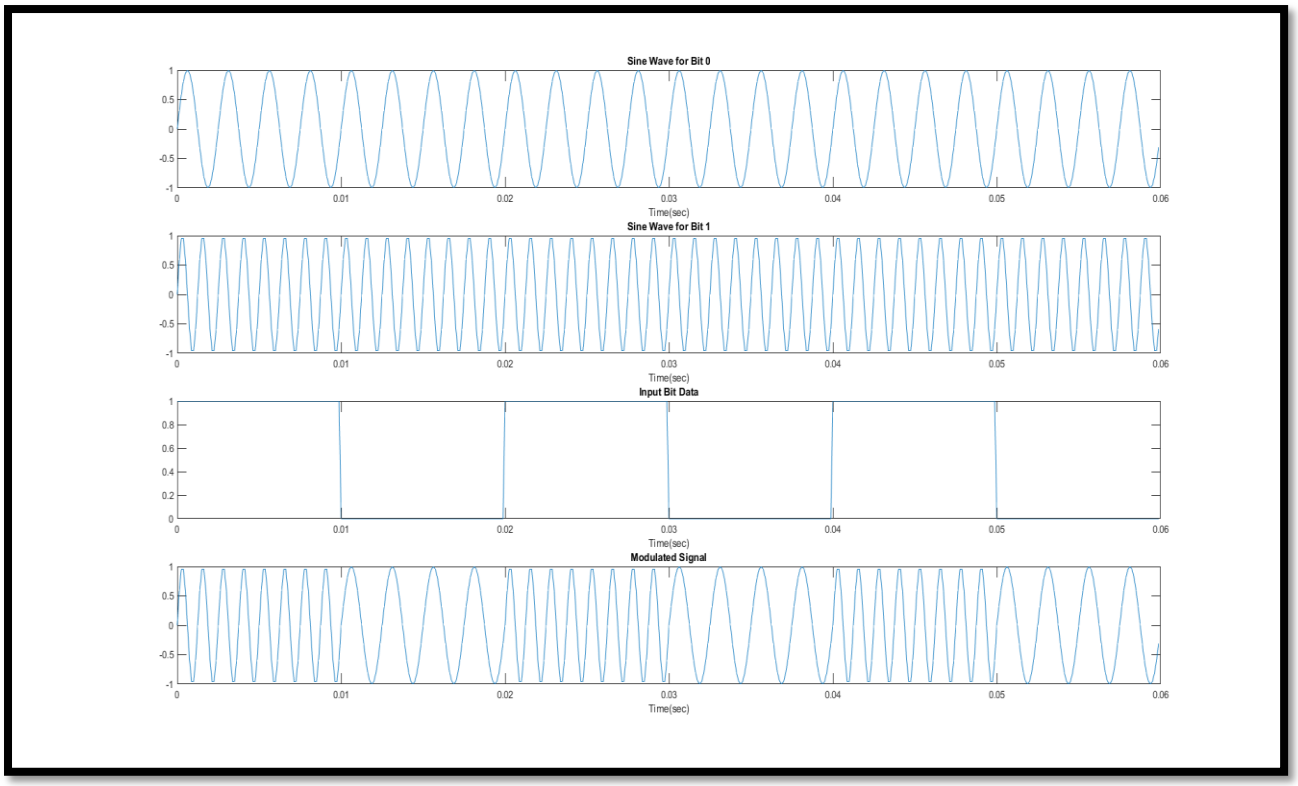
Bit 1 Demodulation



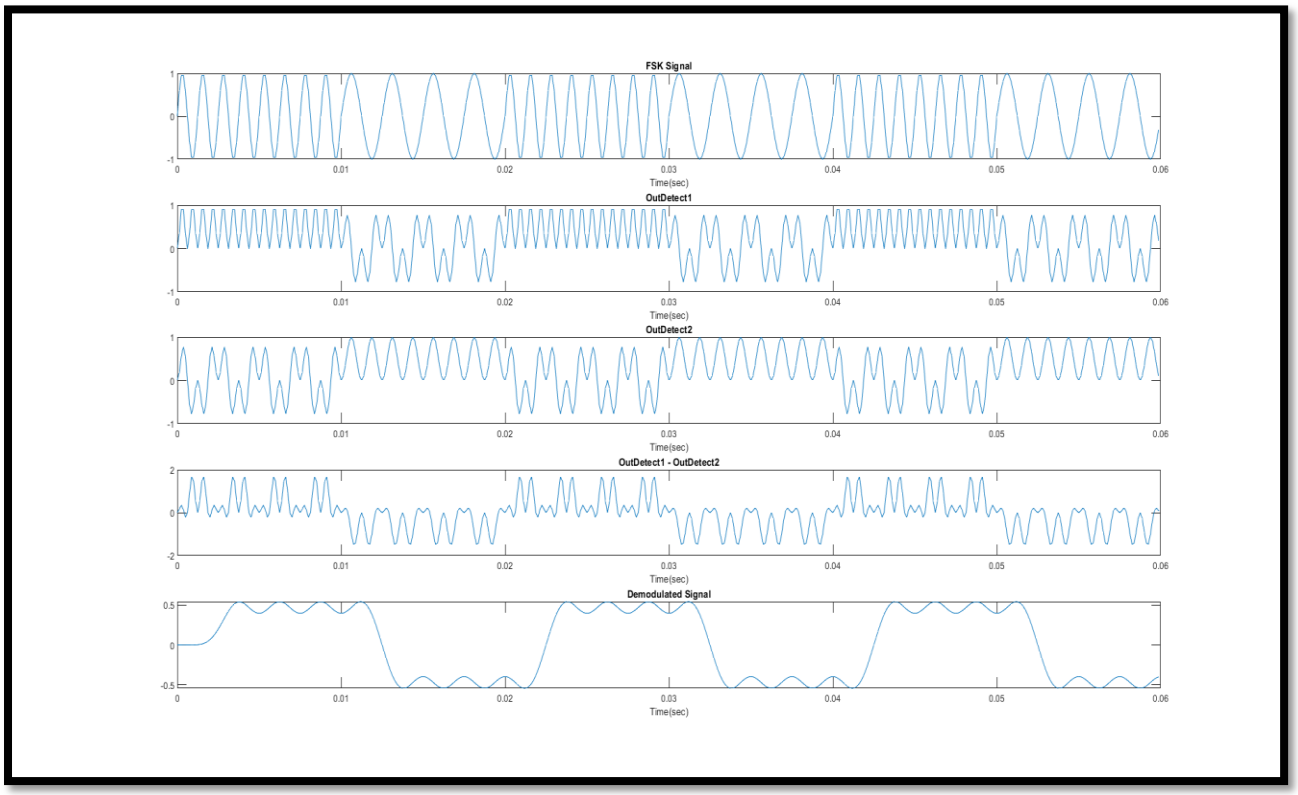
Bit 1 FFT



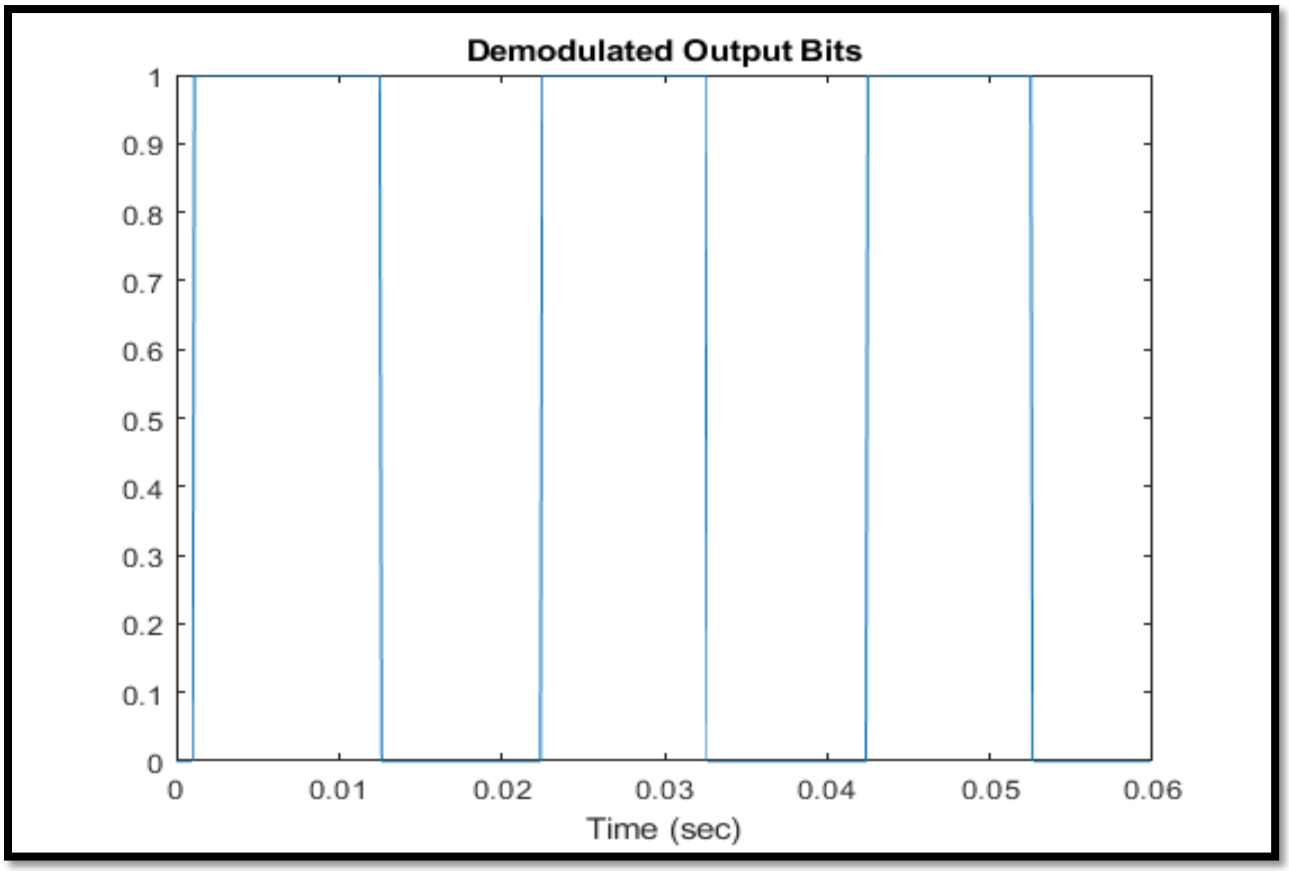
Kaiser LPF



Alternating sequence of 0s and 1s Modulation

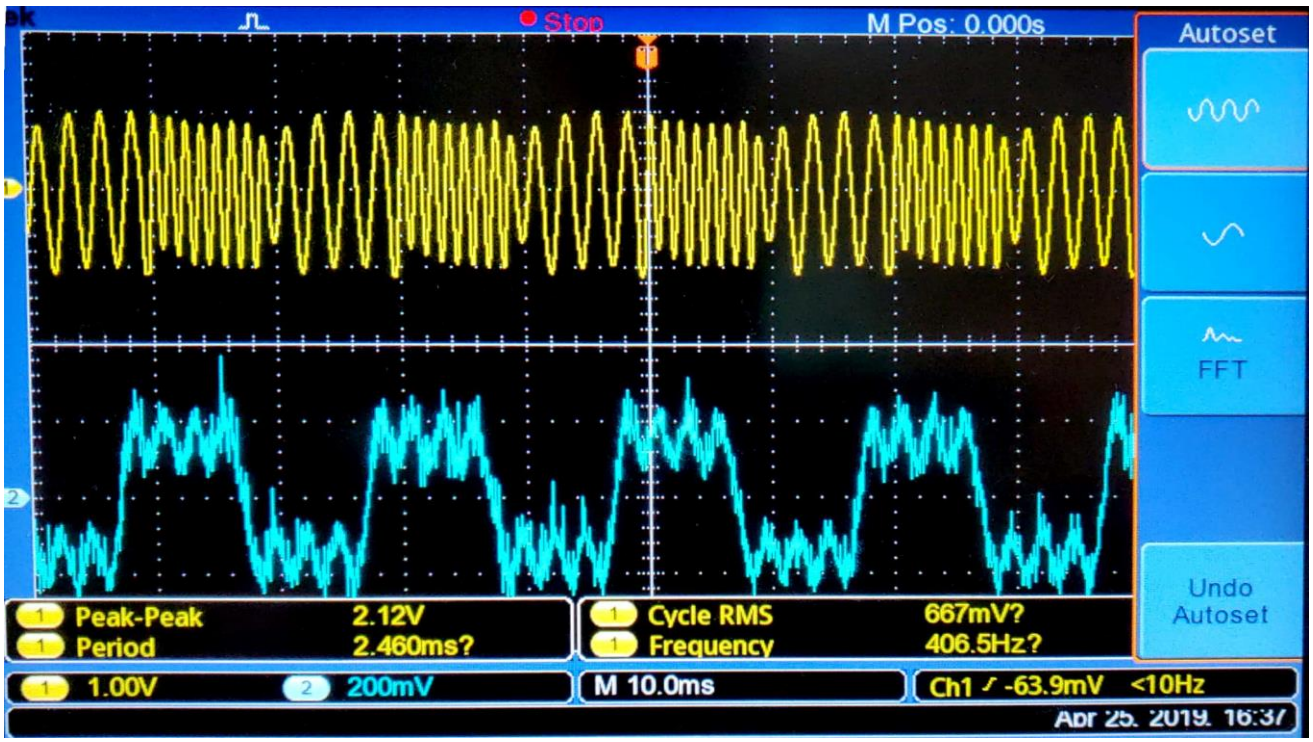


Alternating sequence of 0s and 1s Demodulation



Demodulated Output Bits to show sequence of [1 0 1 0 1 0]

DSK Output:



Appendix

Matlab code:

```
%% Modulation

input_data = [1 0 1 0 1 0]; %% Input Bit Sample
Fs = 8000; %% Sampling Frequency
BitRate = 100; %% Bitrate
T = 1/Fs; %% Sample Period
BitPeriod = 1/BitRate; %% period of a bit
StopTime = 0.06; %% Period of Observation
t = 0 : T : StopTime - T;
tb = 0 : T : BitPeriod - T;
f1 = 400;
f2 = 800;
y1 = sin ( 2 * pi * f1 * t);
y2 = sin ( 2 * pi * f2 * t);
figure,subplot(4,1,1),plot(t,y1);
xlabel('Time(sec)');
title('Sine Wave for Bit 0');
subplot(4,1,2),plot(t,y2);
xlabel('Time(sec)');
title('Sine Wave for Bit 1');

%Generate Modulated Wave from Input Sequence
FSK_Signal = zeros(1,length(t));
FSK_Signal_slice = zeros(1,length(tb));
for i = 1 : 1 : length(input_data)
   if(input_data(i) == 1)
      FSK_Signal_slice = sin (2 * pi * f2 * tb);
   elseif(input_data(i) == 0)
      FSK_Signal_slice =  sin (2 * pi * f1 * tb);
   end
```

```
   FSK_Signal(length(tb)*(i-1)+1:length(tb)*i) = FSK_Signal_slice;
   tb = tb + 0.01;
end

%Normalise tb after Modulation Operation
tb = tb - length(input_data) * 0.01;
%Generate Input Digital Signal for Display
Digital_Signal = zeros(1,length(t));
Digital_Signal_slice = zeros(1,length(tb));

for i = 1 : 1 : length(input_data)
   if(input_data(i) == 1)
      Digital_Signal_slice(1:length(tb)) = 1;
   elseif (input_data(i) == 0)
      Digital_Signal_slice(1:length(tb)) = 0;
   end
   Digital_Signal(length(tb)*(i-1)+1:length(tb)*i) = Digital_Signal_slice;
   tb = tb + 0.01;
end

subplot(4,1,3),plot(t,Digital_Signal);
xlabel('Time(sec)');
title('Input Bit Data');
subplot(4,1,4),plot(t,FSK_Signal);
xlabel('Time(sec)');
title('Modulated Signal');

%% Demodulation
out_detect1 = FSK_Signal .* sin (2 * pi * f2 * t);
out_detect2 = FSK_Signal .* sin (2 * pi * f1 * t);
diff = out_detect1 - out_detect2;
figure,subplot(5,1,1),plot(t,FSK_Signal);
xlabel('Time(sec)');
title('FSK Signal');
subplot(5,1,2),plot(t,out_detect1);
```

```
xlabel('Time(sec)');
title('OutDetect1');
subplot(5,1,3),plot(t,out_detect2);
xlabel('Time(sec)');
title('OutDetect2');
subplot(5,1,4),plot(t,diff);
xlabel('Time(sec)');
title('OutDetect1 - OutDetect2');

% Kaiser LPF Filters out the higher frequency components
% leaving behind the DC Components
%%[n,Wn,beta,ftype] = kaiserord([200,300],[1 0],[0.002 0.001],8000);
n =30;
Wn = 2*250/Fs;
beta = 5;
hh = fir1(n,Wn,'low',kaiser(n+1,beta),'noscale');
Demodulated_Signal = filter(hh,1,diff);
subplot(5,1,5),plot(t,Demodulated_Signal);
xlabel('Time(sec)');
title('Demodulated Signal');
figure,freqz(hh,1,1024,8000); %% Frequency Response of Filter
title('Frequency Response of Kaiser LPF');

Output_Bit = ones (1, length(Demodulated_Signal));
for i = 1 : length(Demodulated_Signal)
   if(Demodulated_Signal(i) > 0)
      Output_Bit(i) = 1;
   else
      Output_Bit(i) = 0;
   end
end
figure
plot(t, Output_Bit)
%% Spectrum of Signals
% Generate Spectrum of Resulting Signals
```

```
N=size(t);
N = N(2);
%%Fourier Transform
outd1_sp = fftshift(fft(out_detect1));
outd2_sp = fftshift(fft(out_detect2));
diff_sp = fftshift(fft(diff));
demod_sp = fftshift(fft(Demodulated_Signal));
%%Frequency Specs
dF = Fs/N; %Hertz
f = -Fs/2:dF:Fs/2-dF;   %Hertz

%%Spectrum Plot
figure;
subplot(4,1,1),plot(f,abs(outd1_sp)/N);
xlabel('Frequency (Hz)');
title('Out Detect 1 Frequency Response');

subplot(4,1,2),plot(f,abs(outd2_sp)/N);
xlabel('Frequency (Hz)');
title('Out Detect 2 Frequency Response');

subplot(4,1,3),plot(f,abs(diff_sp)/N);
xlabel('Frequency (Hz)');
title('(Out Detect 1 - Out Detect 2) Frequency Response');

subplot(4,1,4),plot(f,abs(demod_sp)/N);
xlabel('Frequency (Hz)');
title('Demodulated Signal Frequency Response');

%% Input to DSK
% Generate Wave to supply to DSK for Real-Time Operation
DSK_Input = zeros(1, 1000 * length(FSK_Signal));
for i = 1:1000
   DSK_Input (length(FSK_Signal)*(i-1)+1:length(FSK_Signal)*i) = FSK_Signal;
end
```