Department of Electronics and Telecommunication Engineering

University of Moratuwa

EN2570 – Digital Signal Processing

Project Report

Design of a Bandpass FIR filter

Index no: 180427N

Abstract

This report discusses the procedure of implementing a finite duration impulse response (FIR) bandpass filter using direct method closed form along with Kaiser windowing technique. Fourier series analysis also employed for designing purposes while considering given specifications. This is a software-based implementation using MATLAB 2017. The operation of the designed filter is evaluated by comparing the results of the ideal bandpass filter. Ultimately, this project results that efficient FIR filter can be implemented using the Kaiser window technique.

Contents

1. Introduction

Filter designing takes a great place in Digital Signal Processing. In this report, I have discussed the implantation procedure of direct closed-form FIR filter using Kaiser window technique. MATLAB has been used for software-based implementation. Fourier series has been used to obtain window function and there are several parameters have been computed. The operation of the designed filter has been verified by filtering a given input signal with an ideal filter. The frequency response of the signal (DFT) has been obtained using the fast Fourier transform (FFT) algorithm (fft command in matlab). The input signal contains three different frequencies that lie in the lower stopband, passband, and upper stopband. The ideal filter results in only the passband frequency, so our designed filter also should give the same result. Throughout this project, I have obtained the results that we expected.

2. Basic theory and designing procedure.

Implementation of this FIR bandpass filter has been done using Fourier series analysis along with Kaiser window technique. Usually, this is done by truncating the impulse response of the ideal frequency response while preserving required information using Kaiser widow. The following steps clearly describes the procedure of implementing this filter while satisfying given specifications.

The parameters that are used to implement this bandpass filter according to the given specification in the task are listed below.

Parameters	Values	Units
Maximum passband ripple, $\tilde{A}_{\mathrm{p}}$	0.07	dB
Maximum stopband attenuation, $\tilde{A}_{\mathrm{a}}$	47	dB
Lower passband edge, Ω_{p1}	1000	rad/s
Upper passband edge, Ω_{p2}	1400	rad/s
Lower stopband edge, Ω_{a1}	850	rad/s
Upper stopband edge, Ω_{a2}	1500	rad/s
Sampling frequency, Ω_s	3800	rad/s

Table 1

Step 1

Determine impulse response $h(nT)$ of idealized frequency response of bandpass filter with cutoff frequencies Ω_{c1} and Ω_{c2} using Fourier series.

$$\mathrm{H}\left(e^{j\Omega \mathrm{T}}\right) = \begin{cases} 1 & for -\Omega_{c2} \leq \Omega \leq -\Omega_{c1} \\ 1 & for \quad \Omega_{c1} \leq \Omega \leq \Omega_{c2} \\ 0 & otherwise \end{cases}$$

$$h(nT) = \begin{cases} \dfrac{2}{\Omega_s}(\Omega_{c2} - \Omega_{c1}) & for\, n = 0 \\ \dfrac{1}{n\pi}(sin(\Omega_{c2}\mathrm{nT}) - sin(\Omega_{c1}\mathrm{nT})) & otherwise \end{cases}$$

Step 2

Using the given specifications (Table 1) the next necessary few parameters are calculated.

Derived parameters	Formulas	Values	Units
Lower transition width, $Bt1$	$\Omega_{p1} - \Omega_{a1}$	150	rad/s
Upper transition width, $Bt2$	$\Omega_{a2} - \Omega_{p2}$	100	rad/s
Critical transition width, Bt	Min ($Bt1$, $Bt2$)	100	rad/s
Lower cutoff frequency, Ω_{c1}	$\Omega_{p1} - \frac{Bt}{2}$	950	rad/s
Upper cutoff frequency, Ω_{c2}	$\Omega_{p2} + \frac{Bt}{2}$	1450	rad/s
Sampling period, T	$\frac{2\pi}{\Omega s}$	0.0017	s

Step 3

δ is estimated such that $Ap \leq \widetilde{Ap}$ and $Aa \geq \widetilde{Aa}$; where Ap, Aa are actual passband ripple and actual stopband attenuation, respectively.

Thus δ is defined as:

$$\delta = \min(\delta p, \delta a)$$

Where δp *and* δa are given by,

$$\delta = \frac{10^{0.05\widetilde{Ap}} - 1}{10^{0.05\widetilde{Ap}} + 1} \quad \text{and} \quad \delta a = 10^{-0.05\widetilde{Aa}}$$

Step 4

Actual stopband attenuation is then calculated using the δ that we obtained in step 3 as,

$$Aa = 20\log\left(\frac{1}{\delta}\right) = -20\log(\delta)$$

Step 5

In step 5 another Kaiser window parameter α is chosen as,

$$\alpha = \begin{cases} 0 & for\ Aa \leq 21\ dB \\ 0.5842(Aa - 21)^{0.4} + 0.07886(Aa - 21) & for\ 21 < Aa \leq 50\ dB \\ 0.1102(Aa - 8.7) & for\ Aa > 50dB \end{cases}$$

Step 6

The parameter D is chosen as,

$$D = \begin{cases} 0.9222\ , & for\ Aa \leq 21\ dB \\ \dfrac{Aa - 7.95}{14.36}\ , & for\ Aa > 21\ dB \end{cases}$$

Then the length of the filter N (lowest odd value) is chosen that would satisfy the inequality,

$$N \geq \frac{\Omega s\ D}{Bt} + 1$$

Step 7

Using derived parameters, the Kaiser window function is obtained as follows,

$$w_k(nT) = \begin{cases} \dfrac{I_o(\beta)}{I_o(\alpha)}\ , & |n| \leq \dfrac{N-1}{2} \\ 0\ , & otherwise \end{cases}$$

Where,

$$\beta = \alpha\sqrt{1 - \left(\frac{2n}{N-1}\right)^2}\ , \quad I_o(x) = 1 + \sum_{k=1}^{\infty}\left[\frac{1}{k!}\left(\frac{x}{2}\right)^k\right]^2$$

The below table shows the predefined parameters values that I obtained for my filter implementation.

δp	0.004
δa	0.0045
δ	0.004
Aa	47.895 dB
α	4.3008
D	2.7817
N	107

Step 8

This is last step of the procedure of Kaiser window technique where the required filter is obtained by performing multiplication between ideal impulse response $h(nT)$ and window function $w_k(nT)$. Then its Z transform is calculated of shifted filter to obtain causal filter.

Designed filter: $h'(nT) = h(nT)w_k(nT)$

Causal filter: $h'(nT)causal = h'\left((n - \frac{N-1}{2})T\right)$

Z domain representation: $H'_{\omega}(z) = Z^{-(N-1)/2} H_{\omega}(z)$ (Fourier transform can be found by substituting $z = e^{j\Omega T}$).

3. Evaluation of designed filter

In order to evaluate the designed filter, an excitation $x(nT)$ is generated with three different frequencies, each one of falls in different band. Then this generated signal is filtered by the ideal bandpass filter and designed bandpass filter separately. Hence, the operation of the designed filter is evaluated by comparing the results of those filtering.

$$x(nT) = \sum_{i=1}^{3} sin(\Omega_i nT)$$

where,

$\Omega_1 = 425 \ \ rad/s$, middle frequency of the lower stopband.

$\Omega_2 = 1200\, rad/s$, middle frequency of the passband.

$\Omega_3 = 1700\, rad/s$, middle frequency of the upper stopband.

4. Results and Discussions

The following plots are depicting the results that were obtained during the implementation of the FIR bandpass filter using MATLAB. Figure1 shows the causal impulse response of the designed filter, it is symmetric as we expected. Figure2 shows the magnitude response of the designed filter over the range of 0 to $\Omega_s/2$. You can notice in the figure 2 that the stop band attenuation is greater than 47 dB (specified attenuation) as we expected.

Note: In MATLAB plot I have mentioned Ω_s as ws, both are same.

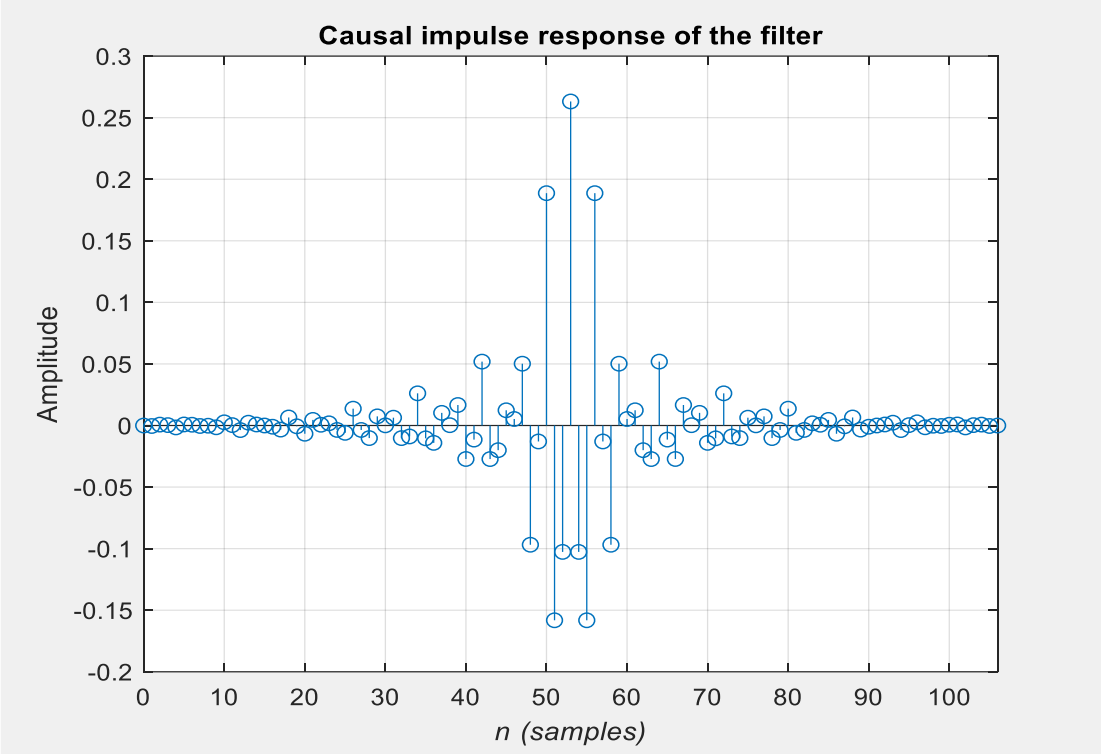

Figure 1

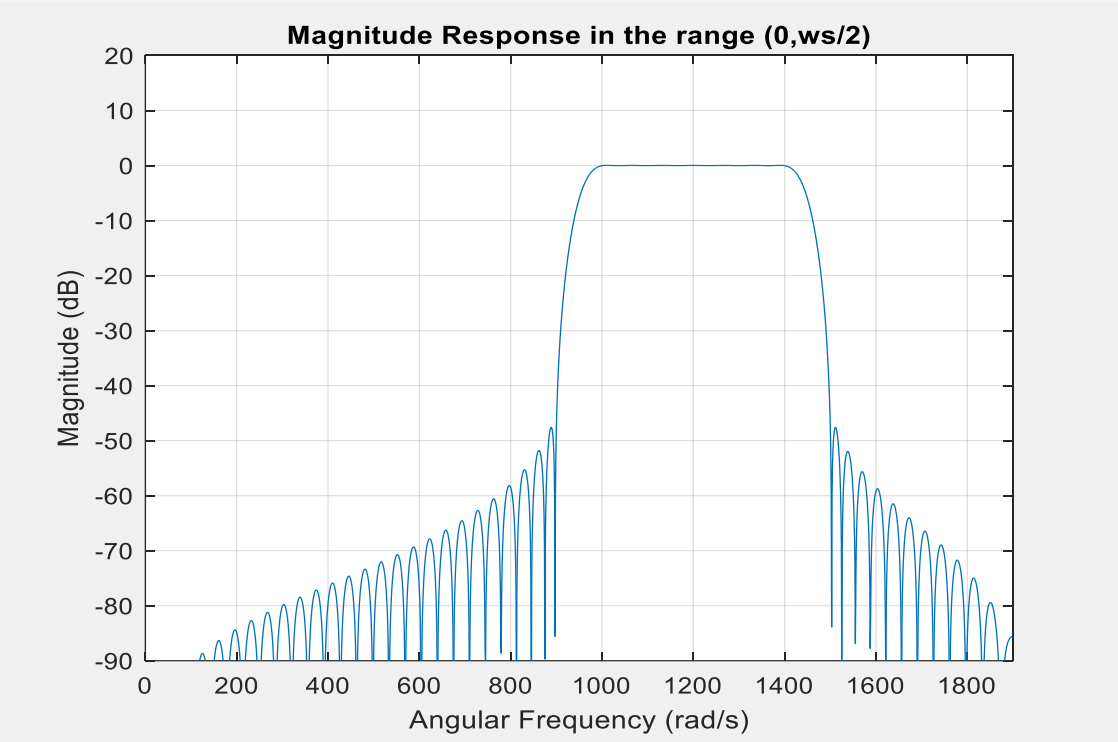

Figure 2

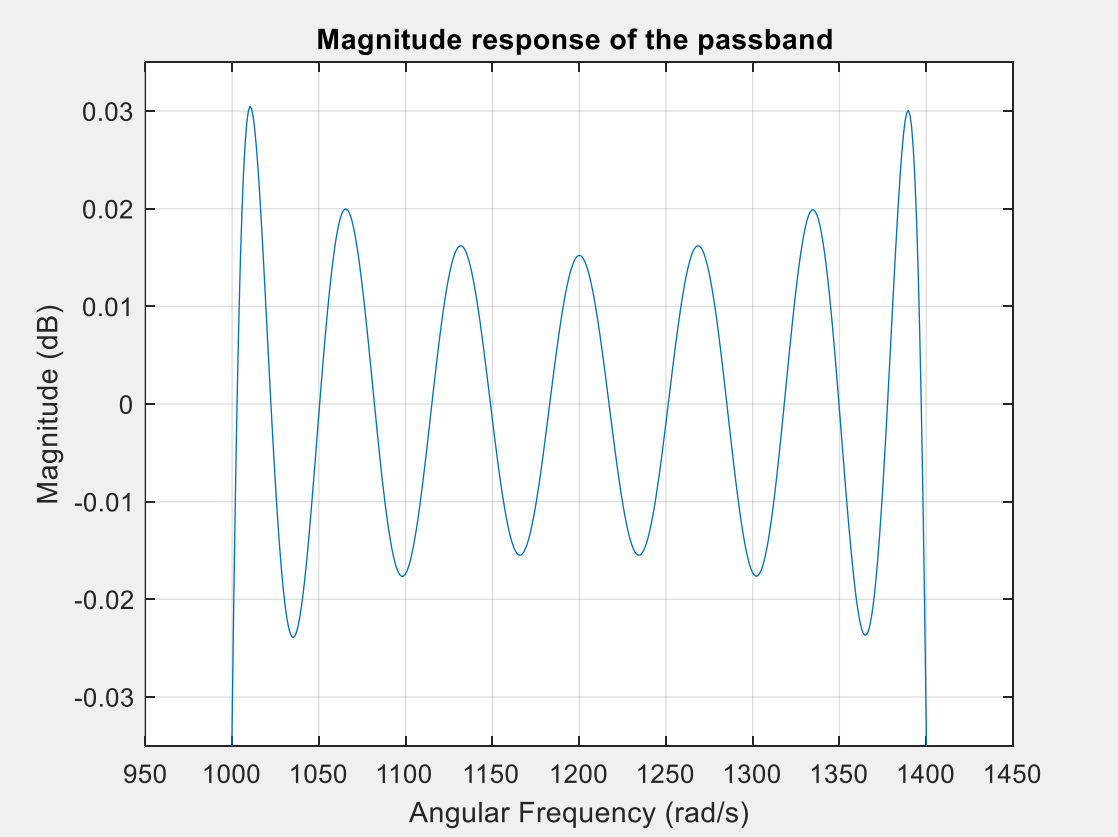

Figure 3

The figure 3 shows the passband ripple of the designed bandpass filter. According to given specification lower and upper passband frequencies are 1000 rad/s and 1400 rad/s respectively. Further maximum passband ripple is 0.07 dB. We could notice in figure three the designed filter achieved given specification precisely.

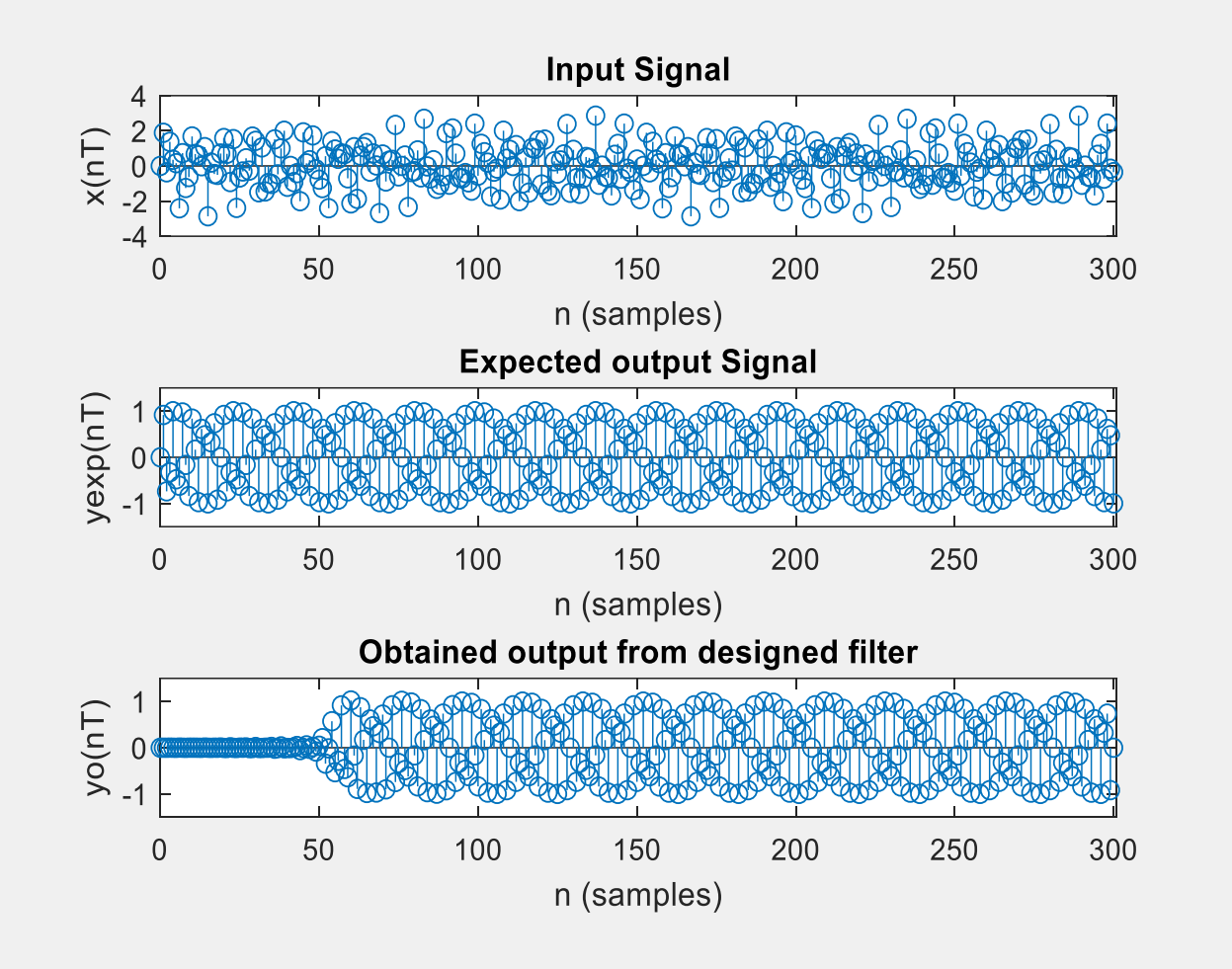

Figure 4

Figure 4 shows the time domain analysis during the evaluation process of the designed filter. In order to achieve steady state response, I considered 300 samples of the input signal that I described above. We can observe in figure 4 that the input signal which has three different frequencies and expected output signal which has only passband frequency and output signal obtained from designed bandpass filter. Further, obtained output is pretty much same as expected output except first 50 samples. Due to the transient effect, we cannot get expected output at the beginning of the signal. However, in steady state we could achieve what we want.

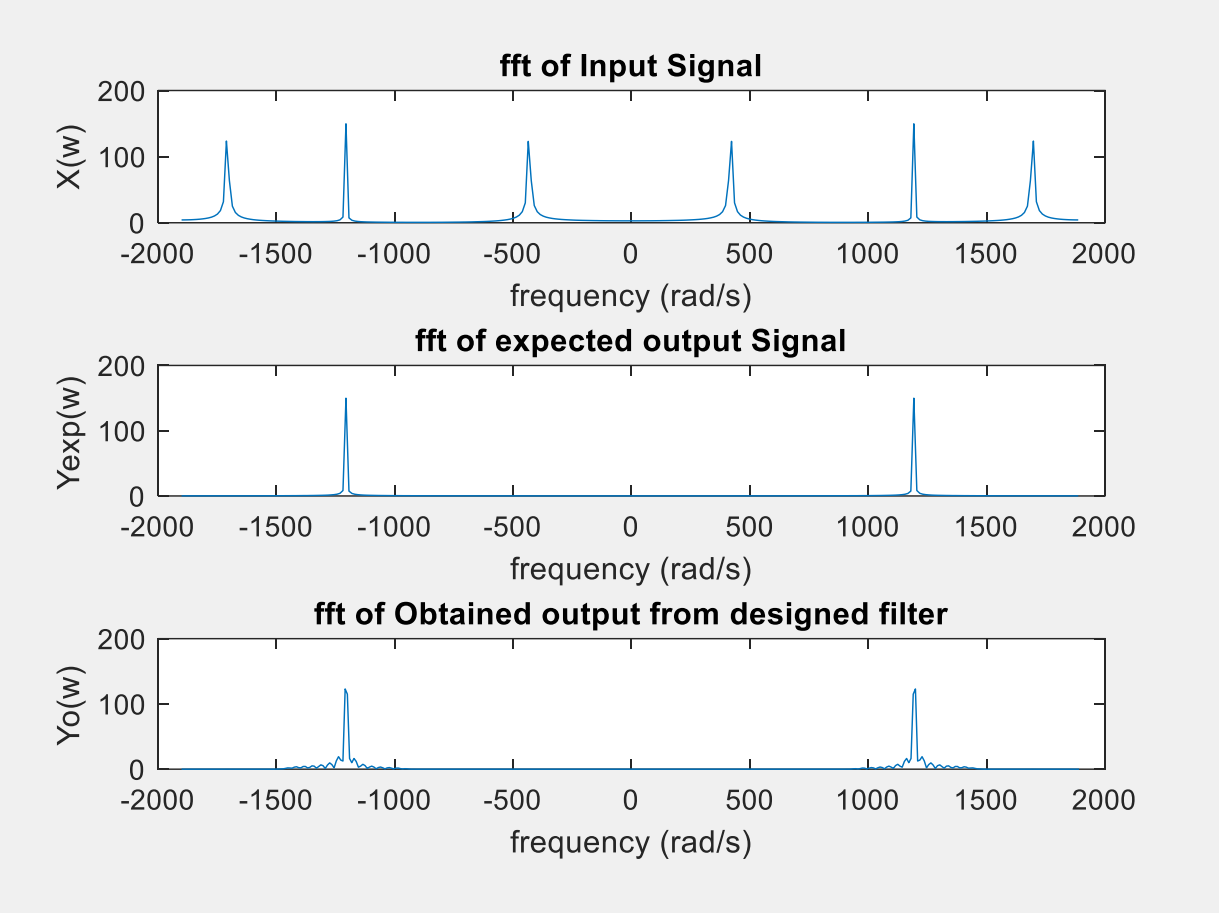

Figure 5

Figure 5 shows the frequency domain analysis corresponded to time domain analysis in figure 4. I considered both positive and negative frequency to plot frequency domain representation. As per the specification the input signal has three different frequencies. Further expected and obtained outputs are pretty much similar to each other.

5. Conclusion

As we discussed the comparison between the ideal and designed bandpass filter above, both results pretty much the same thing. That means designing of FIR filter using Kaiser window technique is efficient in terms of performance. Although the designed filter has passband ripple and stopband attenuation, we could achieve better performance of the filter. However, this method requires a higher order of the filter which leads to inefficiency in terms of hardware requirement. Thus, we cannot conclude this method is very efficient in all circumstances. How ever it is often used in many applications. Further, to overcome from this higher order issue several optimization techniques is used.

6. Acknowledgement

I would like to thank our lecturer Dr. Chamira Edussooriya for providing such an opportunity to learn more about the FIR filter. I gained much knowledge about FIR filter designing using Kaiser window technique and curiosity of this field has increased.

7. References

[1] A. Antoniou. (2005). *Digital Signal Processing: Signals, Systems, and Filters* [Online]. Available: http://www.d-filter.ece.uvic.ca/SupportMaterials.html

8. Appendix (MATLAB code)

MATLAB codes to implement the filter.

1. Filter implementation

```
% Filter implementation
% Filter specifiations
% index no : 180427n
clc;
clear all;
A = 4;
B = 2;
C = 7;
Ap  = 0.03 + (0.01*A);  % Maximum passband ripple, dB
Aa  = 45 + B;           % Maximum stopband ripple, dB
wp1 = C*100 + 300;      % Lower passband edge, rad/s
wp2 = C*100 + 700;      % Upper passband edge, rad/s
wa1 = C*100 + 150;      % Lower stopband edge, rad/s
wa2 = C*100 + 800;      % Upper stopband edge, rad/s
ws  = 2*(C*100 + 1200); %sampling frequency rad/s
T   = 2*pi/ws;          %sampling period

bt1 = wp1 - wa1;        %Lower transition width
bt2 = wa2 - wp2;        %Upper transition width
bt  = min(bt1, bt2);    %Critical transition width
wc1 = wp1 - bt/2;       %Lower cutoff frequency
wc2 = wp2 + bt/2;       %Upper cutoff frequency
```

```
%deriving delta
delta_p = (10^(0.05*Ap) - 1)/(10^(0.05*Ap) + 1);
delta_a = 10^(-0.05*Aa);
delta   = min(delta_p, delta_a);

%Actual stopband attenuation
act_Aa = 20*log10(1/delta);

%calculating alpha
if act_Aa <= 21
    alpha = 0;
elseif act_Aa > 50
    alpha = 0.1102*(act_Aa - 8.7);
else
    alpha = 0.5842*(act_Aa-21)^0.4 + 0.07886*(act_Aa-21);
end

%calculating D
if act_Aa <= 21
    D = 0.9222;
else
    D = (act_Aa - 7.95)/14.36;
end

% calculating lowest odd value for N(lenth of the filter)
N = ceil(ws*D/bt + 1);
if mod(N,2) == 0
    N = N + 1;
end
disp(N);
% determining kaiser window parameters
nmax = (N - 1)/2; %maximum time steps (because of symmetry only..
                ... considered positive part)

kwn = zeros(1, nmax+ 1);   % row vector to store kaiser window value
```

```
% obtaining Io(alpha)
Io_alpha = 1;
k = 1;
while true
   bassel = (1/factorial(k) * (alpha/2)^k)^2; % current bassel value
   Io_alpha = Io_alpha + bassel;
   k = k + 1;
   if bassel < 10e-6      % bassel limit taught in lecture
       break
   end
end

% obtaining Io(beta)
for k = 1:nmax+1
    beta = alpha * sqrt(1 - (2*(k-1)/(N-1))^2); % calculating beta
    Io_beta = 1;    % initializing
    j = 1;
    while true
        bassel = (1/factorial(j) * (beta/2)^j)^2;
        Io_beta = Io_beta + bassel;
        j = j + 1;
        if bassel < 10e-6      % bassel limit taught in lecture
           break
        end
    end
    kwn(k) = Io_beta/Io_alpha; % uptataing window rowvector
end
```

```
% idealized impulse response of filter
hi_nT = zeros(1,nmax+ 1);
for n = 1:nmax+1
    if n == 1
        hi_nT(n) = (2/ws)*(wc2 - wc1);
    else
        hi_nT(n) = ((sin(wc2*(n-1)*T) - sin(wc1*(n-1)*T))/((n-1)*pi));
    end
end
```

```
% determining causal impulse response
htemp = hi_nT .* kwn;
hn = [fliplr(htemp(2:end)), htemp]; %fliping positive value to negative

figure;
stem(0:2*nmax, hn);
title("Causal impulse response of the filter");
xlim([0 N-1]); grid on;
ylabel("Amplitude"), xlabel("\it{n} (samples)");

% frequency response of the filter
[freq_res, f] = freqz(hn, 1, 2048);  % f - rad/sample H(w) = freq_res
w = (f*ws)/(2*pi);                   % w - rad/sec

% magnitute response in the range of (0, ws/2)
mag_freq_res = 20*log10(abs(freq_res));
figure;
plot(w, mag_freq_res);
title('Magnitude Response in the range (0,ws/2)');
xlabel('Angular Frequency (rad/s)');
ylabel('Magnitude (dB)');
axis([0, ws/2, -90, 20]);
grid on;
```

```
% magnitude response of the passband
figure;
plot(w, mag_freq_res);
title('Magnitude response of the passband ');
xlabel('Angular Frequency (rad/s)');
ylabel('Magnitude (dB)');
axis([wc1, wc2, -Ap/2, Ap/2]);
grid on;
```

2. Input signal generation and filtering

```
% Input signal generation
w1 = wa1/2;
w2 = wp1 + (wp2 - wp1)/2;
w3 = wa2 + (ws/2 - wa2)/2;

samples = 300;
n = 0:samples;
xnT = sin(w1*n*T) + sin(w2*n*T) + sin(w3*n*T);

% Expected output signal
yexp = sin(w2*n*T);

% Filtered signal
N_point = length(xnT) + length(hn) -1;
Xw = fft(xnT,N_point);
Hw = fft(hn, N_point);
filtered_out = ifft(Xw.*Hw);
```

3. plotting time domain signal

```
figure;
subplot(3,1,1); stem(n, xnT); title("Input Signal");
xlabel('n (samples)'); ylabel('x(nT)');
axis([0, (samples+1), -4, 4]);

subplot(3,1,2); stem(n, yexp); title("Expected output Signal");
xlabel('n (samples)'); ylabel('yexp(nT)');
axis([0, (samples+1), -1.5, 1.5]);

subplot(3,1,3); stem(n, filtered_out(1:samples+1));
title("Obtained output from designed filter");
xlabel('n (samples)'); ylabel('yo(nT)');
axis([0, (samples+1), -1.5, 1.5]);
```

4. frequency domain analysis

```
% freqency domain analisis
Xw = fft(xnT);
n1 = length(Xw);
Yexpw = fft(yexp);
f_n = (0 : n1-1)/(ws/n1);%freqency vector for input and expected output

Xshift = fftshift(Xw);
Yshift = fftshift(Yexpw);
fshift = (-n1/2 : n1/2 -1)*(ws/n1);

YoW = fft(filtered_out, N_point);
Ye_shift = fftshift(YoW);
fe_shift = (-N_point/2 : N_point/2 -1)*(ws/N_point); % freq vector

figure;
subplot(3,1,1); plot(fshift, abs(Xshift));
title("fft of Input Signal");
xlabel('frequency (rad/s)'); ylabel('X(w)');

subplot(3,1,2); plot(fshift, abs(Yshift));
title("fft of expected output Signal");
xlabel('frequency (rad/s)'); ylabel('Yexp(w)');

subplot(3,1,3); plot(fe_shift, abs(Ye_shift));
title("fft of Obtained output from designed filter");
xlabel('frequency (rad/s)'); ylabel('Yo(w)');
```