EN2570 - DIGITAL SIGNAL PROCESSING

FIR FILTER DESIGN

PROJECT REPORT

DEPARTMENT OF ELECTRONICS AND TELECOMMUNICATION ENGINEERING

UNIVERSITY OF MORATUWA

NAME : J.S.M JAYAWARDHANA

INDEX NO : 160247T

ABSTRACT

The report contains the procedure of designing a Finite Impulse Response (FIR) Digital Filter to the given specifications and its results. The designed filter is a bandpass filter which attenuates all the frequencies except the frequencies in a desired range.

The designing was done using Matlab R2016a version using windowing method. The window used is the Kaiser Window. The required specifications have been achieved and the they are included in the report graphically.

INTRODUCTION

A processing of continuous time signal in the discrete time domain is important and useful in many ways. This requires the proper sampling of the continuous time signal. Processing, storing and transmitting digital signals are easier and feasible comparing to continuous time signals. For the processing, we require filters for getting the desired frequency components from a signal which contains many undesired frequencies. Therefore, the digital filters have a higher importance in many fields. Image processing and machine vision applications, telecommunication, sound engineering, bio medical applications are few of those fields.

An ideal filter will attenuate all the undesired frequencies completely and pass the desired frequency components without any phase distortion or magnitude distortion. Implementation of an ideal filter is not practical however. A practical filter will attenuate the undesired frequencies into some extent and there can be a phase distortion or/and magnitude distortion in the pass band frequencies. Also, we cannot achieve a sharp transition between passband and stopband using a practical filter. Therefore, our design will have some specified limitations for these practical issues.

A digital filter can be designed using many methods. An FIR filter can be designed using windowing method using various windows. The selection of the proper window depends on the specifications of the filter.

FILTER DESIGN

Specifications of The Filter

According to the index number 160247T, the calculated parameters of the filter are as follows.

- Maximum Passband ripple Ap' = 0.07 dB
- Maximum Stopband attenuation Aa' = 44 dB
- Lower Passband edge wp1 = 1100 rad/s
- Upper Passband edge wp2 = 1400 rad/s
- Lower Stopband edge wa1 = 950 rad/s
- Upper stopband edge wa2 = 1500 rad/s
- Sampling frequency ws = 3800 rad/s

Method

Step 01 – Calculation of the following parameters

- Sampling Period T = 2π/ws = 0.0017 s
- Critical Transition Width Bt = min[(wp1-wa1),(wa2-wp2)] = 100 rad/s
- Lower cut-off frequency wc1 = wp1 - Bt/2 = 1050 rad/s
- Upper cut-off frequency wc2 = wp2 + Bt/2 = 1450 rad/s

Step 02 – Choosing δ

δ is chosen such that the actual passband ripple, Ap is equal to or less than specified passband ripple Ãp, and the actual minimum stopband attenuation, Aa is equal or greater than the specified minimum stopband attenuation, Ãa .

Let δ = min ($\delta p'$, $\delta a'$)

Where $\delta p' = \frac{10^{0.05Ap'}-1}{10^{0.05Ap'}+1}$ and $\delta a' = 10^{-0.05Aa'}$

Step 03 – Calculation of the stopband attenuation

The relevant stopband attenuation Aa is calculated as $Aa = -20 \log(\delta)$

Step 04 – Choosing α

$$\alpha = \begin{cases} 0 & Aa \leq 21 \\ 0.5842(Aa-21)^{24} + 0.07886(Aa-21) & for\ 21 < Aa \leq 50 \\ 0.007886(Aa-21) & for\ Aa > 50 \end{cases}$$

Step 05 – Choosing D

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

Step 06 – Choosing N

The minimum value is selected for N that would satisfy the inequality $N \geq \frac{WsD}{Bt} + 1$

Step 07 – Generating wk(nT)

$wk(nT)$ is derived using the following equations:

$$\beta = \alpha \sqrt{1 - \left(\frac{2n}{N-1}\right)^2}$$

$$I_0(x) = 1 + \sum_{k=1}^{\infty} \left[\frac{1}{k!}\left(\frac{x}{2}\right)^k\right]^2$$

$$w_k(nT) = \begin{cases} \dfrac{I_0(\beta)}{I_0(\alpha)} & for\ |n| \leq (N-1)/2 \\ 0 & otherwise \end{cases}$$

The following parameters were calculated by following the above steps.

δ = 0.04

Aa = 47.895

Ap = 0.07

α = 4.3001

D = 2.7817

N = 107

Step 08 – Generating h(nT)

The ideal frequency response of the filter is,

$$H\left(e^{j\omega}\right) = \begin{cases} 1 & for -\omega_{c2} \leq \omega \leq -\omega_{c1} \\ 1 & for\ \omega_{c1} \leq \omega \leq \omega_{c2} \\ 0 & otherwise \end{cases}$$

$$h(nT) = \begin{cases} \frac{2}{\omega_s}(\omega_{c2} - \omega_{c1}) & when\ \ n = 0 \\ \frac{2}{n\omega_s}(\sin(\omega_{c2}nT) - \sin(\omega_{c1}nT)) & otherwise \end{cases}$$

Step 09 – Truncation of the Ideal Impulse Response

The ideal impulse response is symmetrically truncated using the Kaiser window. The resulting impulse response has a length of N. The causal impulse response is got by shifting the truncated response by (N-1)/2.

$hk(nT) = h(nT) \times wk(nT)$

TESTING

The output of the designed filter for an input x(nT) is compared with the ideal output.

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

Where,

ω_1 = Mid frequency of the lower stopband (475 rad/s)

ω_2 = Mid frequency of the passband (1250 rad/s)

ω_3 = Mid frequency of the upper stopband (1700 rad/s)

RESULTS

For the frequency domain representation of the signals, the Fast Fourier Transform (FFT) has been used which is an approximation for Discrete Fourier Transform(DFT). The required specifications have been well achieved.

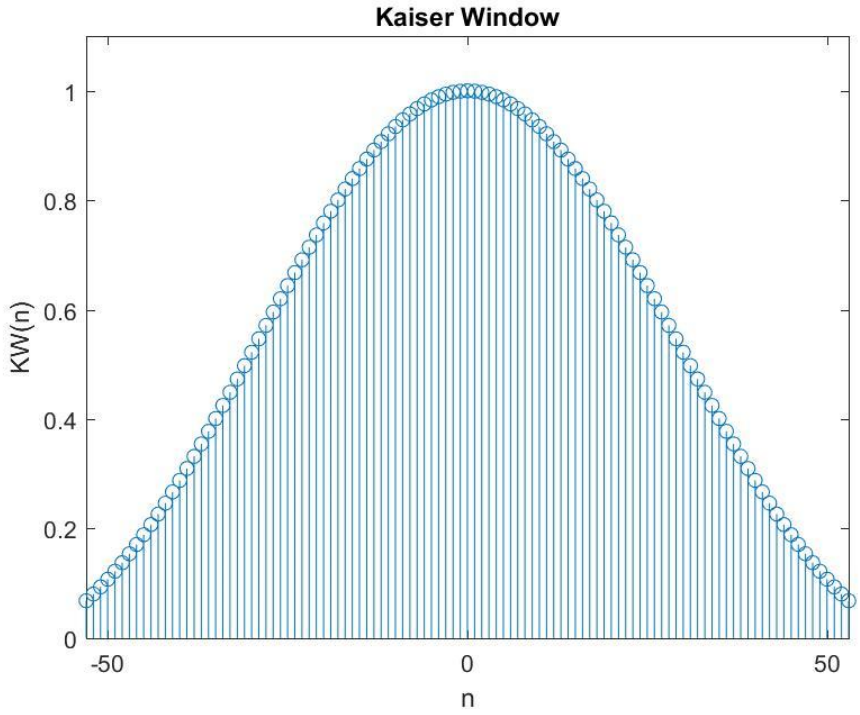

Figure 1

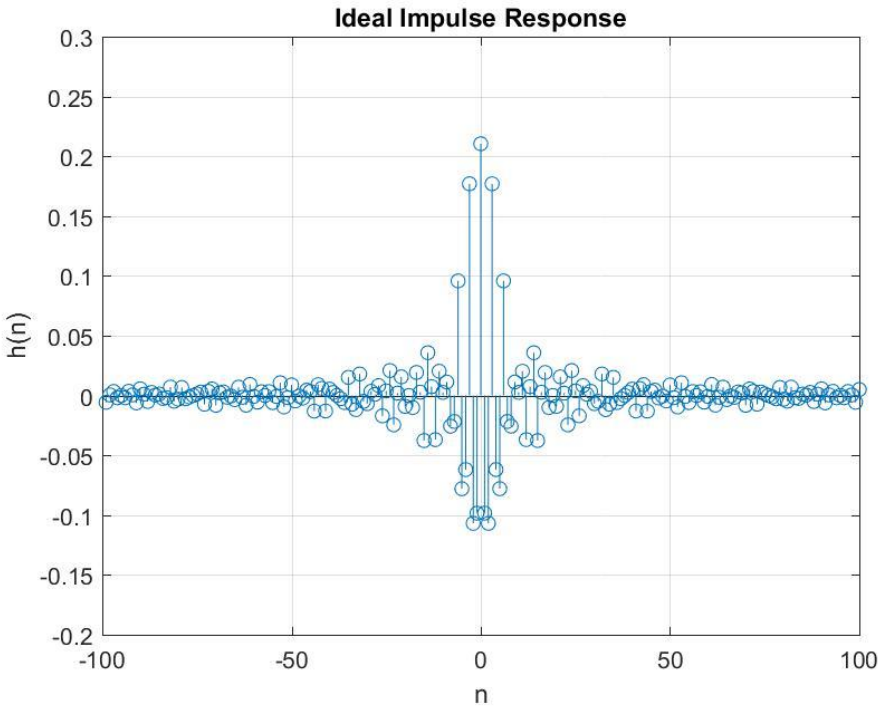

Figure 2

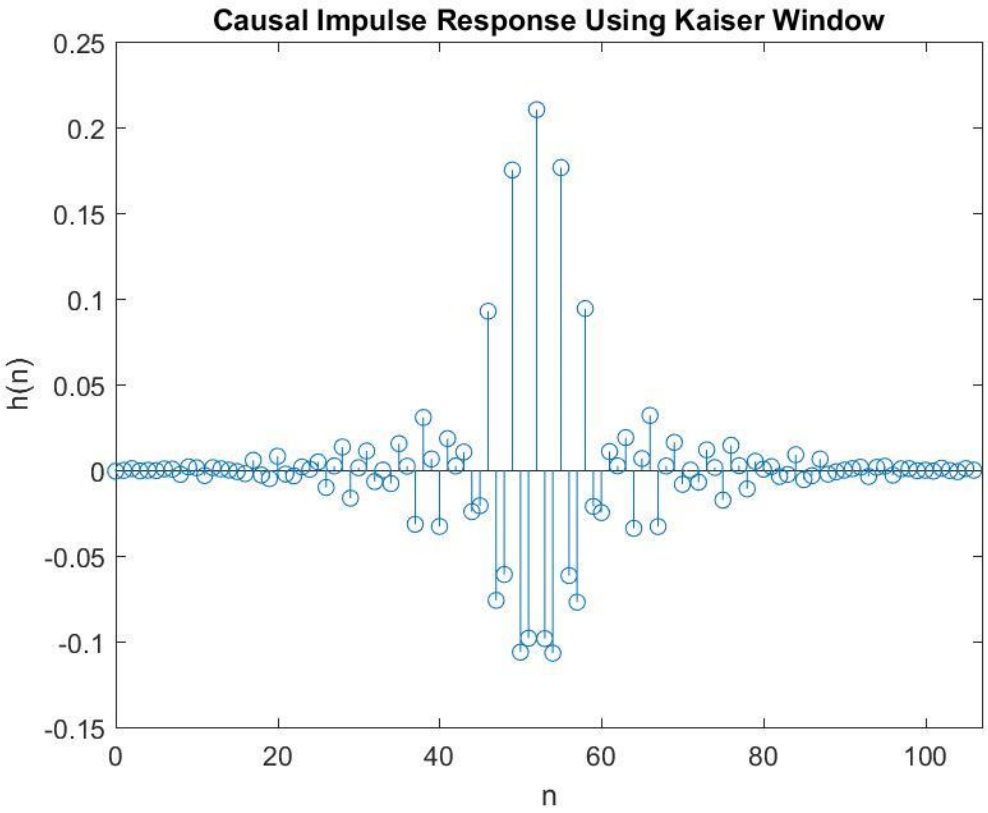

Figure 3

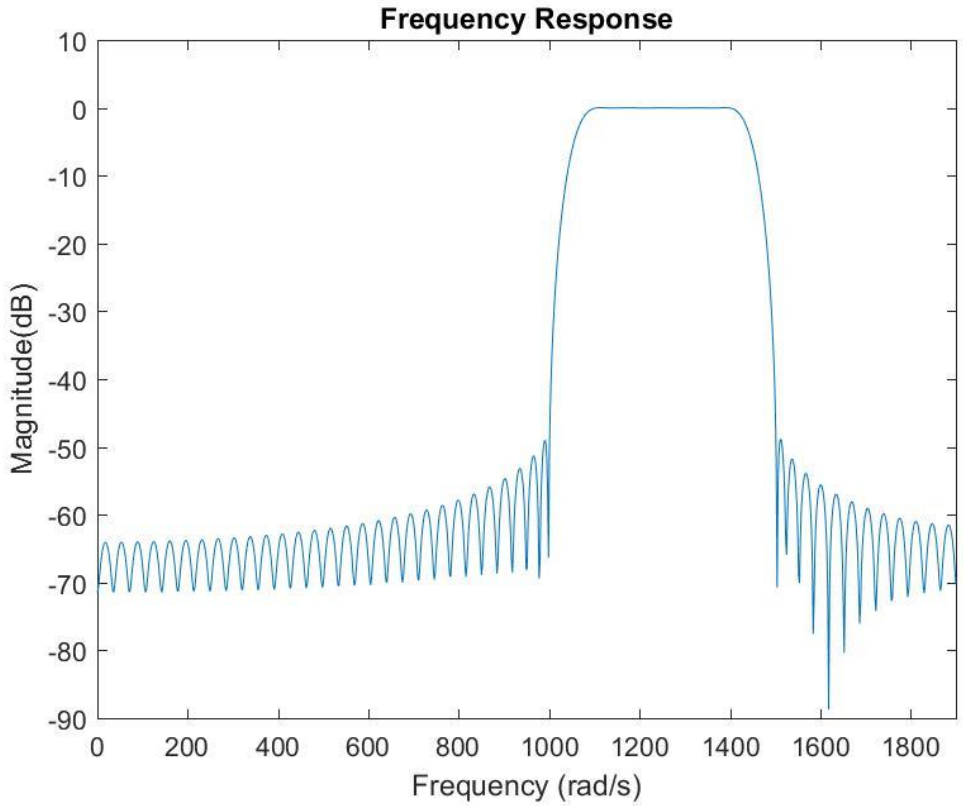

Figure 4

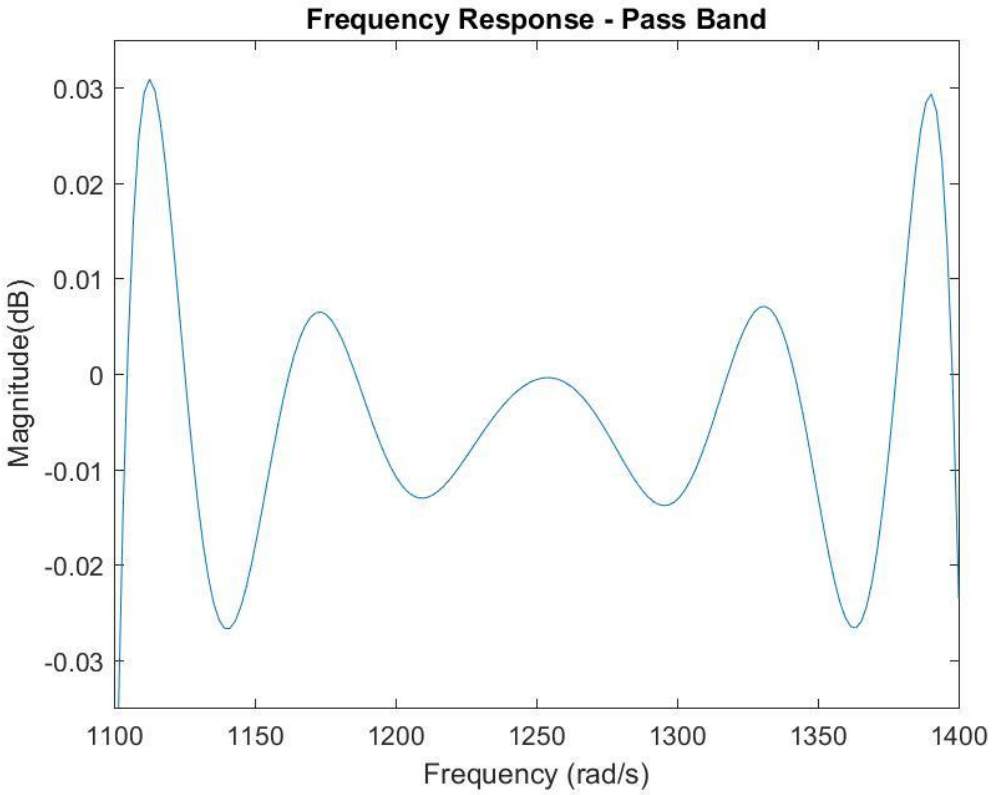

Figure 5

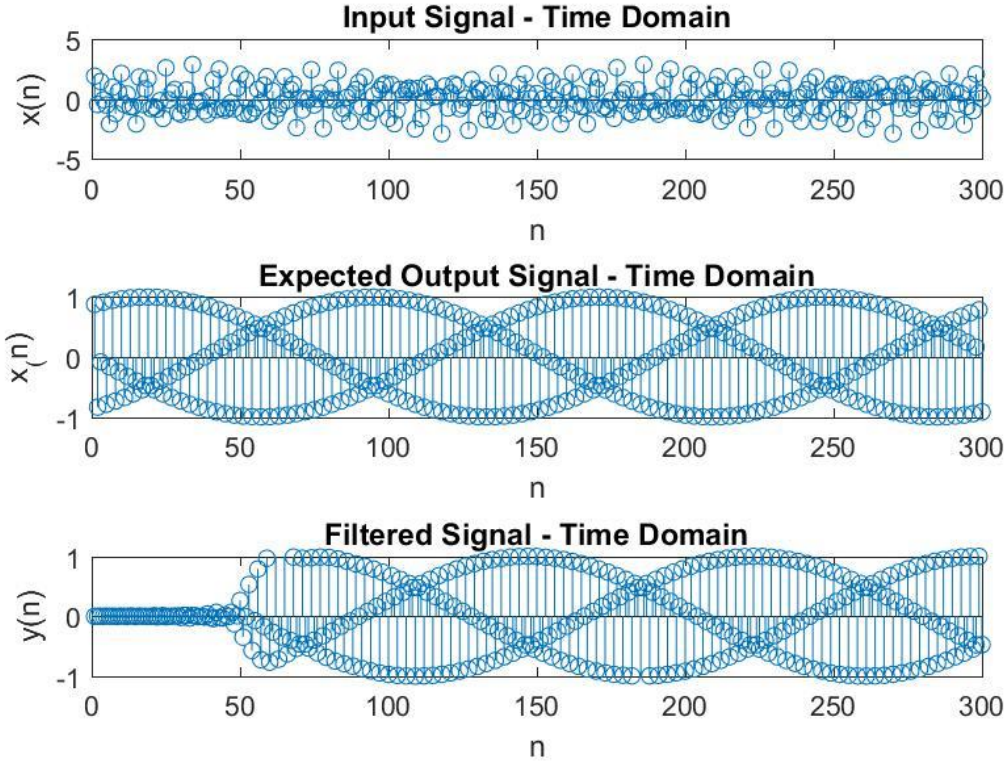

Figure 6 - Comparison of Results in Time Domain

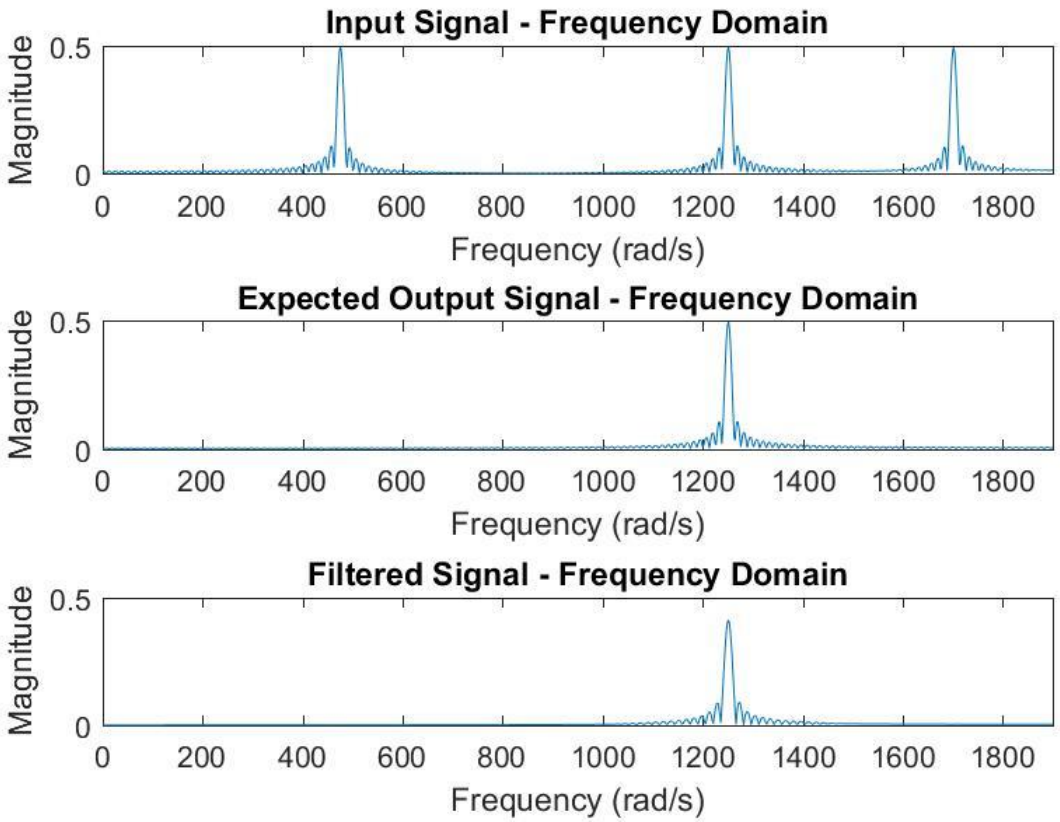

Figure 7 - Comparison of Results in Frequency Domain

DISCUSSION

The plots show that the designed filter accepts the specifications given. Figure 3 shows that the impulse response of the designed filter is causal and symmetric. Figure 4 shows that the stop band gain is less than -44dB which is the requirement. Therefore, the undesired frequencies will be attenuated well. The figure 5 shows the pass band ripple which is less than 0.07dB. Therefore, the distortion to the pass band frequencies is reduced enough. Figure 6 and figure 7 shows the time domain and frequency domain representation of the expected and received outputs for the given input. They show that the frequencies in the stop band have been attenuated almost completely while passing the pass band frequencies with a negligible distortion.

The windowing method can be concluded as a proper method for the design of practical digital filters. Kaiser window has shown very positive results for the windowing method. However, the computational complexities will increase the cost and the complexity of the implementation of the filter. The efficiency of the filter can be low because of these complexities.

REFERENCES

[1]A. Antoniou. (2005). Digital Signal Processing: Signals, Systems, and Filters [Online].

Available at : http://fmipa.umri.ac.id/wp-content/uploads/2016/03/Andreas-Intoniou-Digital-signal-processing.9780071454247.31527.pdf

[2]Matlab Documentation

Available at: https://www.mathworks.com/help/matlab/

APPENDICES

Appendix I (MATLAB code)

```
close all; clc;
%IndexNo:160247T
A = 2; B = 4; C = 7;

tildAp = 0.05+(0.01*A) ;      %0.07dB
tildAa = 40+B ;               %44dB
omegaP1 = (C*100)+400 ;       %1100 rad/s
omegaP2 = (C*100)+700;        %1400 rad/s
omegaA1 = (C*100)+250;        %950 rad/s
omegaA2 = (C*100)+800;        %1500 rad/s
omegaS = 2*((C*100)+1200);    %3800 rad/s

Ts = (2*pi)/omegaS;
Bt1 = omegaP1 - omegaA1;
Bt2 = omegaA2 - omegaP2;
Bt = min(Bt1,Bt2);
omegac1 = omegaP1-Bt/2;
omegac2 = omegaP2+Bt/2;
omegaC1 = (omegac1)*Ts;
omegaC2 = (omegac2)*Ts;


 %Ideal Frequency Response
 omega = linspace(0,pi,1000);
 idfreqRes = linspace(0,0,1000);
for i =  1:1:1000
    if (omega(i)>=omegaC1 && omega(i)<=omegaC2)
        idfreqRes(i) = 1;
    else
        idfreqRes(i) = 0;
    end
end
 figure;
 plot(omega.*(1/Ts),idfreqRes);
 xlim([0,omegaS/2]);
 ylim([-0.5,1.5]);
 title('Ideal Frequency Response')
 xlabel('Frequency (rad/s)')
 ylabel('H(w)')
 grid on
 print('Ideal Frequency Response.jpg','-djpeg');
```

```
%Impulse Response
lnt = 100;
m = linspace(-lnt+1,lnt,2*lnt);
impRes = linspace(0,0,2*lnt);
for k = m
    if k==0
        impRes(k+lnt)=(1/pi)*(omegaC2-omegaC1);
    else
        impRes(k+lnt)=(1/(k*pi))*(sin(omegaC2*k)-sin(omegaC1*k));
    end
end
figure;
stem(m,impRes);
xlim([-lnt,lnt]);
ylim([-0.2,0.3]);
title('Ideal Impulse Response')
xlabel('n')
ylabel('h(n)')
grid on
print('Ideal Impulse Response.jpg','-djpeg');


%KaiserWindowGeneration
tilddeltP = (10^(0.05*tildAp)-1)/(10^(0.05*tildAp)+1);
tilddeltA = 10^(-0.05*tildAa);
delta = min(tilddeltP,tilddeltA);
Ap = 20*log10((1+delta)/(1-delta));
Aa = -20*log10(delta);
alpha = 0.584*(Aa-21)^(0.4)+0.07886*(Aa-21);
D = (Aa-7.95)/14.36;
N = round(1+((omegaS*D)/Bt)) ;
if mod(N,2)==0
    N = N+1;
end
k = 20;
Kwind = linspace(0,0,N);
n = -(N-1)/2:1:(N-1)/2;
for m = n
    beta = alpha*sqrt(1-((2*m)/(N-1))^2);
    Kwind(m+(N-1)/2+1) = iox(beta,k)/iox(alpha,k);
end
figure;
stem(n,Kwind);
xlim([-(N-1)/2,(N-1)/2]);
ylim([0,1.1])
title('Kaiser Window')
xlabel('n')
ylabel('KW(n)')
print('Kaiser Window.jpg','-djpeg');
```

```
%Getting filter impulse responese using Kasier window
kwnt = linspace(0,0,N);
for a = 1:1:N
    kwnt(a)=Kwind(a)*impRes(lnt-(N-1)/2+a);
end
figure;
stem(n+(N-1)/2,kwnt);
xlim([0,N]);
title('Causal Impulse Response Using Kaiser Window')
xlabel('n')
ylabel('h(n)')
print('Causal Impulse Response Using Kaiser Window.jpg','-djpeg');

%Frequency Responce
freqRes = fft(kwnt,2000);
freqResn = freqRes(1,1:1000);
figure;
plot(omega.*(1/Ts),20*log10(abs(freqResn)));
title('Frequency Response')
xlabel('Frequency (rad/s)')
ylabel('Magnitude(dB)')
xlim([0,omegaS/2]);
print('Frequency Response.jpg','-djpeg');


%Pass Band Frequency Response
figure;
plot(omega.*(1/Ts),20*log10(abs(freqResn)));
title('Frequency Response - Pass Band')
xlabel('Frequency (rad/s)')
ylabel('Magnitude(dB)')
xlim([omegaP1,omegaP2]);
ylim([-tildAp/2,tildAp/2]);
print('Frequency Response Pass Band.jpg','-djpeg');

%Input Signal
n = 1:1:300;
omega1 = omegaA1/2;
omega2 = (omegaP1+omegaP2)/2;
omega3 = (omegaA2+(omegaS/2))/2;
inpSig = sin(omega1*n*Ts)+sin(omega2*n*Ts)+sin(omega3*n*Ts);
Hw = abs(fft(inpSig,2000));

%Convolution
outSig = linspace(0,0,length(inpSig));
for a = 1:1:length(inpSig)
    for b = 1:1:a
        if b<length(kwnt)
            outSig(a) = outSig(a) + kwnt(b)*inpSig(a+1-b);
        end
    end
end
Hw_filt = fft(outSig,2000);
```

```
figure;
subplot(3,1,1);
stem(n,inpSig);
title('Input Signal - Time Domain')
xlabel('n')
ylabel('x(n)')
xlim([0,length(inpSig)]);

subplot(3,1,2);
stem(n,x_nT);
title('Expected Output Signal - Time Domain')
xlabel('n');
ylabel('x_(n)');
xlim([0,length(inpSig)]);

subplot(3,1,3);
stem(n,outSig);
title('Filtered Signal - Time Domain')
xlabel('n')
ylabel('y(n)')
xlim([0,length(inpSig)]);
ylim([-1,1]);
print('Time Domain Comparison.jpg','-djpeg');


figure;
subplot(3,1,1);
plot(omega.*(1/Ts),Hw(1,1:1000).*(2*Ts));
title('Input Signal - Frequency Domain')
xlabel('Frequency (rad/s)')
ylabel('Magnitude')
xlim([0,omegaS/2]);

subplot(3,1,2);
plot(omega.*(1/Ts),abs(X_jw(1,1:1000)).*(2*Ts));
title('Expected Output Signal - Frequency Domain')
xlabel('Frequency (rad/s)')
ylabel('Magnitude')
xlim([0,omegaS/2]);

subplot(3,1,3);
plot(omega.*(1/Ts),abs(Hw_filt(1,1:1000)).*(2*Ts));
title('Filtered Signal - Frequency Domain')
xlabel('Frequency (rad/s)')
ylabel('Magnitude')
xlim([0,omegaS/2]);
print('Frequncy Domain Comparison.jpg','-djpeg');


function Io = iox(x,k)
Io=1;
for a = 1:1:k
    Io = Io + ((1/factorial(a))*(x/2)^(a))^2;
end
```