EE338 : Digital Signal Processing

Fitler Desgin Assignment

NAVJOT SINGH | 130110071 | Filter No. **73**

Specifications

Derived parameters for design:
M=73
q=7
r=3
Signal Bandwidth = 45KHz
Sampling Frequency (f_s) = 100KHz
Transition Bandwidth = 2KHz
Tolerance(δ)= 0.15 (Not in magnitude squared)

Following sections would discuss design of Bandpass and Bandstop filters.

Bandpass Filter Design

Specifications

Passband = Monotonic
Stopband = Monotonic
B$_l$ = 14.9 KHz
B$_h$ = 24.9 KHz
B$_l$s = 12.9 KHz
B$_h$s = 26.9 KHz

Normalized Discrete Filter Specifications

$$\omega = \frac{2B_f\pi}{f_s}$$

ω_{s1} = 0.8105
ω_{p1} = 0.9362
ω_{p2} = 1.5645
ω_{s2} = 1.6902

Analog Filter Specifications (Using Bilinear Transformation)

$$\Omega = tan(\omega/2)$$

Ω_{s1} = 0.4290
Ω_{p1} = 0.5056

Ω_{p2} = 0.9937
Ω_{s2} = 1.1271

IIR Filter Design (Butterworth Lowpass)

Bandpass Analog filter parameters

$$\Omega_0 = \sqrt{\Omega_{p1}\Omega_{p2}}$$
$$B = \Omega_{p2} - \Omega_{p1}$$

Ω_0=0.7088
B=0.4882

Bandpass frequencies to lowpass

$$\Omega_L = \frac{\Omega^2 - \Omega_0^2}{B\Omega}$$
$$\Omega_s = min(\frac{\Omega_{S1}^2 - \Omega_0^2}{B * \Omega_{S1}}, \frac{\Omega_{S2}^2 - \Omega_0^2}{B * \Omega_{S2}})$$

Equivalent Lowpass filter specifications
Ω_p=1
Ω_s=1.3958

Butterworh lowpass filter parameters
Equations Used:

$$D_1 = \frac{1}{1 - {\delta_1}^2} - 1 \qquad D_2 = \frac{1}{{\delta_2}^2} - 1$$
$$N \geq \frac{log(\sqrt{\frac{D2}{D1}})}{2log(\frac{\Omega_{Ls}}{\Omega_{Lp}})} \qquad \frac{\Omega_p}{D_1^{\frac{1}{2N}}} \leq \Omega_c \leq \frac{\Omega_s}{D_2^{\frac{1}{2N}}}$$

D_1=0.3841
D_2=43.4444
N=8
Ω_c=1.0822

Analog Lowpass Transfer function

$$H_{analog,LPF}(s_l) = \frac{1.8807}{s^8 + 5.5470s^7 + 15.3844s^6 + 27.6851s^5 + 35.2288s^4 + 32.4210s^3 + 21.0980s^2 + 8.9083s^1 + 1.8807}$$

Poles in LHP
-0.2111 + 1.0614i
-0.6012 + 0.8998i
-0.8998 + 0.6012i
-1.0614 + 0.2111i
-1.0614 - 0.2111i
-0.8998 - 0.6012i
-0.6012 - 0.8998i
-0.2111 - 1.0614i

Conversion to Bandpass Analog Transfer Function

$$s_l = \frac{s_{bp}^2 + \Omega_0^2}{B s_{bp}}$$

$$H_{analog,BP}(s_b p) = \frac{10^{-4}. \quad 3.6427 s^8}{Denom(s)}$$

$$Denom(s) = 0.0601s^{16} + 0.1626s^{15} + 0.4616s^{14} + 0.7654s^{13} + 1.2084s^{12} + 1.4019s^{11} + 1.5188s^{10} + 1.2950s^9 + 1.0259s^8$$
$$+0.6506s^7 + 0.3834s^6 + 0.1778s^5 + 0.0770s^4 + 0.0245s^3 + 0.0074s^2 + 0.0013s0.0002$$

Conversion to Digital Domain

$$s = \frac{1 - z^{-1}}{1 + z^{-1}}$$

$$H_{digital,BP}(z) = \frac{Num(z)}{10^4.Denom(z)}$$

$$Num(z) = 0.0364z^{-16} - 0.2914z^{-14} + 1.0199z^{-12} - 2.0399z^{-10} + 2.5499z^{-8} - 2.0399z^{-6} + 1.0199z^{-4} - 0.2914z^{-2} + 0.0364$$

$$Denom(z) = 0.0026z^{-16} - 0.0170z^{-15} + 0.0784z^{-14} - 0.2508z^{-13} + 0.6583z^{-12} - 1.3981z^{-11} + 2.5475z^{-10} - 3.9303z^{-9}$$
$$+5.2882z^{-8} - 6.0971z^{-7} + 6.1343z^{-6} - 5.2359z^{-5} + 3.8393z^{-4} - 2.2884z^{-3} + 1.1195z^{-2} - 0.3830z^{-1} + 0.0922$$

Pole Zero Plot

Using command zplane() in MATLAB:

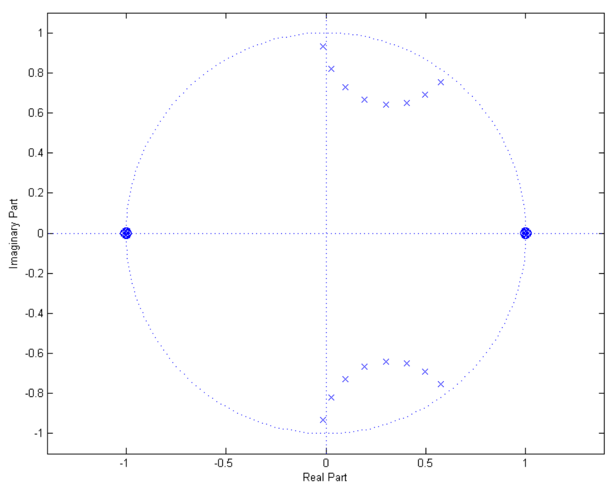


Figure 1: Pole Zero plot of Discrete time Transfer Function

Frequency/Phase Response

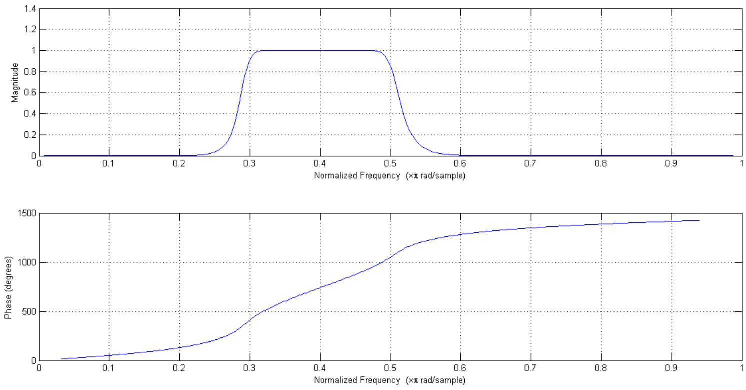
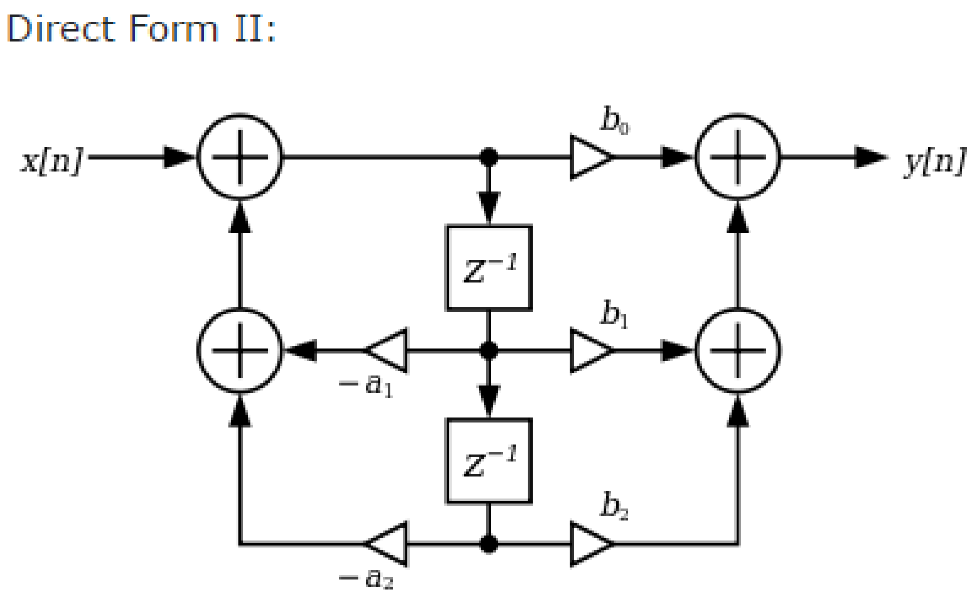


Figure 2: Frequency and Phase Responses

Using command freqz() in MATLAB:

Direct form II Realization



Tap Coefficients:

Numerator: 1 0 -0.2914 0 1.0199 0 -2.0399 0 2.5499 0 -2.0399 0 1.0199 0 -0.2914 0 1
Denominator :0.0026 -0.0170 0.0784 -0.2508 0.6583 -1.3981 2.5475 -3.9303 5.2882 -6.0971 6.1343 -5.2359 3.8393 -2.2884 1.1195 -0.3830 0.0922
Gain = 0.0364

FIR Filter Design (Kaiser Window)

Deriving Parameters for Kaiser window

$$(2N+1) > 1 + \frac{(A-8)}{2.285\Delta\omega_T}$$
$$\Delta\omega_T = \|\omega_s - \omega_p\|$$
$$A = -20 * log_1 0(\delta)$$

For our bandpass filter, we will choose $min\{\omega_{s2} - \omega_{p2}, \omega_{p1} - \omega_{s1}\}$
Hence, we obtain the following
$\Delta\omega_T$=0.1257
A=16.4782
N $\geq$ 15 = 20 (To avoid tolerance band overshoot)
as A<21,we have α=0, and thus β =0

Calculation of Ideal Impulse Response

As we have N=20, we would only calculate the ideal impulse response coefficients from -20 to +20, i.e 41 points in total.
Thus, we have the following

$$h_{ideal}[n] = \frac{sin(n\omega_{c2}) - sin(n\omega_{c1})}{n\pi}, n \neq 0$$
$$h_{ideal}[n] = \frac{\omega_{c2} - \omega_{c1}}{\pi}, n = 0$$

Obtained using MATLAB: (Starting from $h_{ideal}[-N]$)

0.0300	0.0050	-0.0148	-0.0035	-0.0040	-0.0248	-0.0086	0.0411	0.0397	-0.0183
-0.0373	-0.0045	-0.0084	-0.0343	0.0282	0.1210	0.0453	-0.1574	-0.1747	0.0738
0.2400	0.0738	-0.1747	-0.1574	0.0453	0.1210	0.0282	-0.0343	-0.0084	-0.0045
-0.0373	-0.0183	0.0397	0.0411	-0.0086	-0.0248	-0.0040	-0.0035	-0.0148	0.0050
0.0300									

Windowing by a Kaiser Window

As $\beta = 0$, we are essentially using a rectangular window.
hence, h_{filter}=$h_{ideal}.Kaiser_{N,\beta}$
where $Kaiser_{N,\beta}$=1 for -20 to +20;

Transfer function in Discrete Domain

$$h_{filter}[z] = 0.0300 + 0.0050z^{-1} - 0.0148z^{-2} - 0.0035z^{-3} - 0.0040z^{-4} - 0.0248z^{-5} - 0.0086z^{-6} + 0.0411z^{-7} + 0.0397z^{-8} + \\ -0.0183z^{-9} - 0.0373z^{-10} - 0.0045z^{-11} - 0.0084z^{-12} - 0.0343z^{-13} + 0.0282z^{-14} + 0.1210z^{-15} + 0.0453z^{-16} - 0.1574z^{-17} \\ -0.1747z^{-18} + 0.0738z^{-19} + 0.2400z^{-20} + 0.0738z^{-21} - 0.1747z^{-22} - 0.1574z^{-23} + 0.0453z^{-24} + 0.1210z^{-25} + 0.0282z^{-26} \\ -0.0343z^{-27} - 0.0084z^{-28} - 0.0045z^{-29} - 0.0373z^{-30} - 0.0183z^{-31} + 0.0397z^{-32} + 0.0411z^{-33} - 0.0086z^{-34} - 0.0248z^{-35} \\ -0.0040z^{-36} - 0.0035z^{-37} - 0.0148z^{-38} + 0.0050z^{-39} + 0.0300z^{-40}$$

Frequency/Phase Response

Using command freqz() in MATLAB:

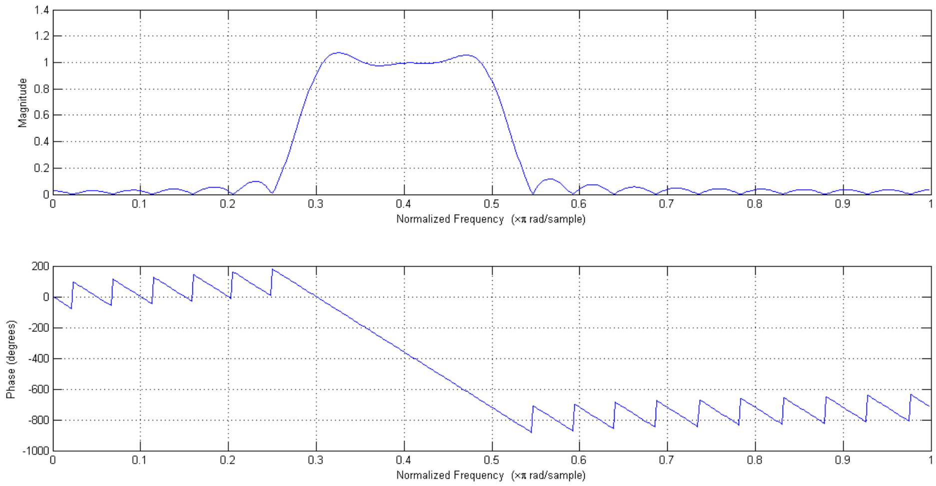


Figure 3: Frequency and Phase Responses of FIR Bandpass filter

Bandstop Filter Design

Passband = Equiripple
Stopband = Monotonic
B_l = 16.3 KHz
B_h = 26.3 KHz
B_lp = 14.3 KHz
B_hp = 28.3 KHz

Normalized Discrete Filter Specifications

$$\omega = \frac{2B_f\pi}{f_s}$$

ω_{p1} = 0.8985
ω_{s1} = 1.0242
ω_{s2} = 1.6525
ω_{p2} = 1.7781

Analog Filter Specifications (Using Bilinear Transformation)

$$\Omega = tan(\omega/2)$$

Ω_{p1} = 0.4821
Ω_{s1} = 0.5621
Ω_{s2} = 1.0852
Ω_{p2} = 1.2323

IIR Filter Design (Chebyshev Type I)

Bandstop Analog filter parameters

$$\Omega_0 = \sqrt{\Omega_{p1}\Omega_{p2}}$$

$$B = \Omega_{p2} - \Omega_{p1}$$

Ω_0=0.7708
B=0.7501

Bandstop frequencies to lowpass

$$\Omega_L = \frac{B\Omega}{\Omega_0^2 - \Omega^2}$$

$$\Omega_s = min(\frac{B * \Omega_{S1}}{\Omega_0^2 - \Omega_{S1}^2}, \frac{B * \Omega_{S2}}{\Omega_0^2 - \Omega_{S2}^2})$$

Equivalent Lowpass filter specifications
Ω_p=1
Ω_s=1.3949

Chebyshev lowpass filter parameters

Equations Used:

$$D_1 = \frac{1}{1 - {\delta_1}^2} - 1 \qquad D_2 = \frac{1}{{\delta_2}^2} - 1$$

$$N \geq \frac{acosh(\sqrt{\frac{D2}{D1}})}{acosh(\frac{\Omega_{Ls}}{\Omega_{Lp}})} \qquad \epsilon \leq \sqrt{D_1}$$

D_1=0.3841
D_2=43.4444
ϵ=0.6197
N=4
Ω_p=1

Analog Lowpass Transfer function

$$H_{analog,LPF}(s_l) = \frac{-0.2017}{s^4 + 0.8342s^3 + 1.3479s^2 + 0.6243s + 0.2373}$$

Poles on LHP
-0.1222 - 0.9698i
-0.2949 - 0.4017i
-0.2949 + 0.4017i
-0.1222 + 0.9698i

Conversion to Bandstop Analog Transfer Function

$$s_l = \frac{Bs}{\Omega_0^2 + s^2}$$

$$H_{analog,BS}(s_{bs}) = \frac{10^{-1}(1.2124s^8 + 2.8812s^6 + 2.5676s^4 + 1.0169s^2 + 0.1510)}{0.1426s^8 + 0.2815s^7 + 0.7949s^6 + 0.7133s^5 + 1.0341s^4 + 0.4238s^3 + 0.2806s^2 + 0.0590s + 0.0178}$$

Conversion to Digital Domain

$$s = \frac{1 - z^{-1}}{1 + z^{-1}}$$

$$H_{digital,BP}(z) = \frac{Num(z)}{Denom(z)}$$

$Num(z) = -0.7829z^{-8}+1.5948z^{-7}-4.3499z^{-6}+5.1979z^{-5}-7.1866z^{-4}+5.1979z^{-3}-4.3499z^{-2}+1.5948z^{-1}-0.7829$

$Denom(z) = 0.7923z^{-8}-1.1424z^{-7}+2.1638z^{-6}-3.5581z^{-5}+6.6787z^{-4}-6.3124z^{-3}+7.1495z^{-2}-4.9700z^{-1}+3.7476$

Pole Zero Plot

Using command zplane() in MATLAB:

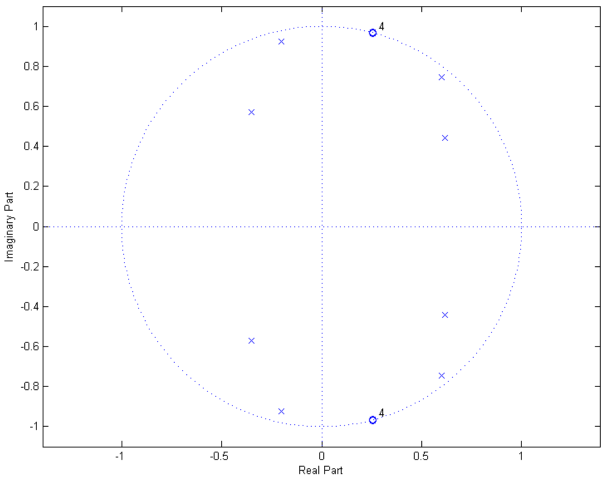


Figure 4: Pole Zero plot of Discrete time Transfer Function

Frequency/Phase Response

Using command freqz() in MATLAB:

Direct form II Realization

Tap Coefficients:
Numerator: 1 1.5948 -4.3499 5.1979 -7.1866 5.1979 -4.3499 1.5948 1
Denominator: 0.7923 -1.1424 2.1638 -3.5581 6.6787 -6.3124 7.1495 -4.9700 3.7476
Gain = 0.7829

FIR Filter Design (Kaiser Window)

Deriving Parameters for Kaiser window

$$(2N + 1) > 1 + \frac{(A - 8)}{2.285\Delta\omega_T}$$

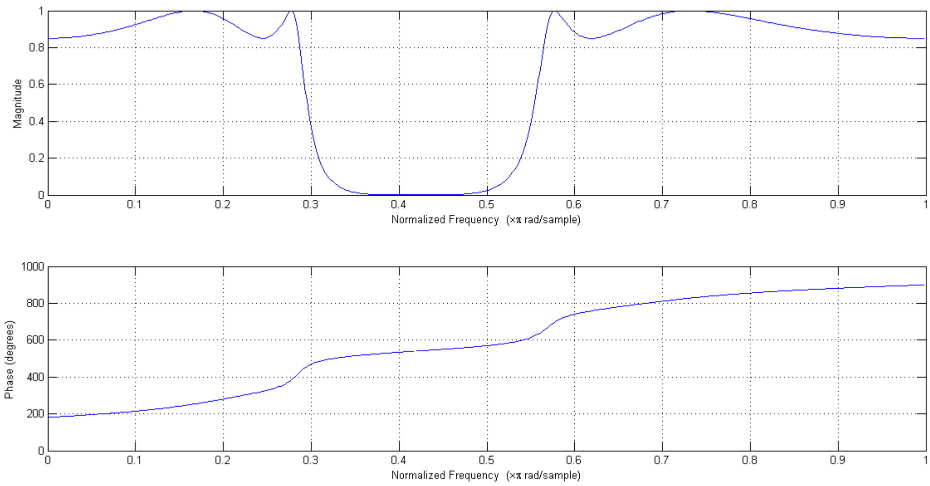
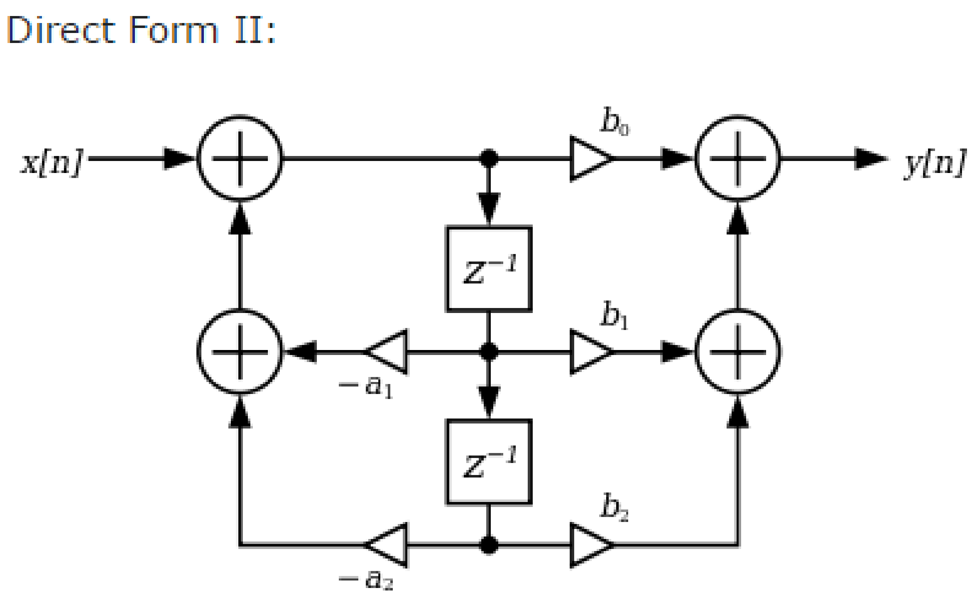


Figure 5: Frequency and Phase Responses



$$\Delta\omega_T = \|\omega_s - \omega_p\|$$

$$A = -20 * log_1 0(\delta)$$

For our bandstop filter, we will choose $min\{\omega_{p2} - \omega_{s2}, \omega_{s1} - \omega_{p1}\}$
Hence, we obtain the following
$\Delta\omega_T$=0.1257
A=16.4782
N $\geq$ 15 = 20 (To avoid tolerance band overshoot)
as A<21,we have α=0, and thus β =0

Calculation of Ideal Impulse Response

As we have N=20, we would only calculate the ideal impulse response coefficients from -20 to +20, i.e 41 points in total.

Thus, we have the following

$$h_{ideal}[n] = \frac{-(sin(n\omega_{c2}) - sin(n\omega_{c1}))}{n\pi}, n \neq 0$$

$$h_{ideal}[n] = 1 - \frac{\omega_{c2} - \omega_{c1}}{\pi}, n = 0$$

Obtained using MATLAB: (Starting from $h_{ideal}[-N]$)

0.0019	-0.0247	-0.0086	0.0034	-0.0083	0.0085	0.0381	0.0057	-0.0489	-0.0270
0.0256	0.0153	0.0028	0.0437	0.0143	-0.1111	-0.0950	0.1233	0.1948	-0.0540
0.7600	-0.0540	0.1948	0.1233	-0.0950	-0.1111	0.0143	0.0437	0.0028	0.0153
0.0256	-0.0270	-0.0489	0.0057	0.0381	0.0085	-0.0083	0.0034	-0.0086	-0.0247
0.0019									

Windowing by a Kaiser Window

As $\beta = 0$, we are essentially using a rectangular window.
hence, h_{filter}=$h_{ideal}.Kaiser_{N,\beta}$
where $Kaiser_{N,\beta}$=1 for -20 to +20;

Transfer function in Discrete Domain

$$h_{filter}[z] = 0.0019 - 0.0247z^{-1} - 0.0086z^{-2} + 0.0034z^{-3} - 0.0083z^{-4} + 0.0085z^{-5} + 0.0381z^{-6} + 0.0057z^{-7} - 0.0489z^{-8} - 0.0270z^{-9} + 0.0256z^{-10} + 0.0153z^{-11} + 0.0028z^{-12} + 0.0437z^{-13} + 0.0143z^{-14} - 0.1111z^{-15} - 0.0950z^{-16} + 0.1233z^{-17} + 0.1948z^{-18} - 0.0540z^{-19} + 0.7600z^{-20} - 0.0540z^{-21} + 0.1948z^{-22} + 0.1233z^{-23} - 0.0950z^{-24} - 0.1111z^{-25} + 0.0143z^{-26} + 0.0437z^{-27} + 0.0028z^{-28} + 0.0153z^{-29} + 0.0256z^{-30} - 0.0270z^{-31} - 0.0489z^{-32} + 0.0057z^{-33} + 0.0381z^{-34} + 0.0085z^{-35} - 0.0083z^{-36} + 0.0034z^{-37} - 0.0086z^{-38} - 0.0247z^{-39} + 0.0019z^{-40}$$

Frequency/Phase Response

Using command freqz() in MATLAB:

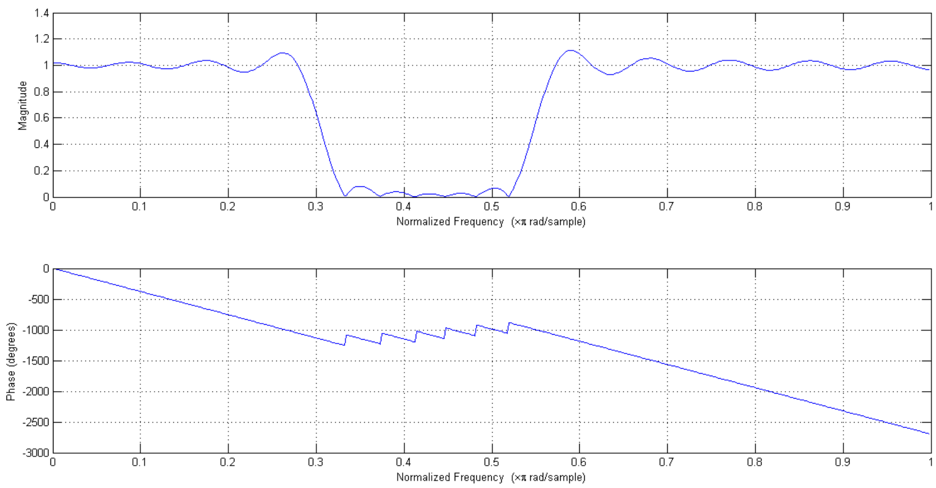


Figure 6: Frequency and Phase Responses of FIR Bandpass filter

MATLAB Codes

Bandpass Filter

IIR Butterworth Design

```
% Filter number 73
% Navjot Singh
% 130110071
% Butterworth

% Global Parameters
delta=0.15;
f_sample = 100000;
f_message = 45000;
t_width = 2000;

% Given analog frequencies corresponding to filter number
B_l= 14900;
B_h= 24900;
B_l_s= B_l - t_width;
B_h_s= B_h + t_width;
analog_specs= [B_l_s, B_l, B_h, B_h_s];

% Conversion to digital specification
digital_specs= (analog_specs*2*pi)/f_sample;

% Bilinear transformation to analog domain
% In the form (Omega_s1,Omega_p1,Omega_p2,Omega_s2)
analog_freq= tan(digital_specs/2);

% Calculation of paramteres
% For Bandpass to Lowpass transfromation
Omega_0= sqrt(analog_freq(2)*analog_freq(3));
B= analog_freq(3)-analog_freq(2);

% Bandpass to Lowpass transformation
% Using Omega_lp = (Omega_bp/B) - (Omega_0^2)/(B*Omega_bp)
% Gives Passband at +-1
analog_lp_freq= (analog_freq/B) - ((Omega_0^2)./(B*analog_freq));
% Choosing the stringent of the two stopbands
Omega_lp_s= min(abs(analog_lp_freq(1)),abs(analog_lp_freq(4)));
Omega_lp_p= 1;

% Design on Butterworth Filter
D1= 1/(1-delta)^2 -1;
D2= 1/delta^2 -1;
N_temp = 0.5*log(D2/D1)/log(Omega_lp_s/Omega_lp_p);
N = ceil(N_temp);           %Order of Butterworth filter
omega_t1=Omega_lp_p/(D1^(1/(2*N)));
omega_t2=Omega_lp_s/(D2^(1/(2*N)));
Omega_c=(omega_t1+omega_t2)/2;

% Poles of Butterworth filter (all poles)
% Are of the form s=j*Omega_c*exp(j(2k+1)pi/2N)
```

```
% Hence real part = -Omega_c*sin((2k+1)pi/2N)
% Hence imaginary part = Omega_c*cos((2k+1)pi/2N)
poles_total_real=zeros(1,2*N);
poles_total_imag=zeros(1,2*N);
for k= 0:(2*N)-1
    poles_total_real(k+1) = -1*Omega_c*sin((((2*k)+1)*pi)/(2*N));
    poles_total_imag(k+1) = Omega_c*cos((((2*k)+1)*pi)/(2*N));
end

% Taking poles on the LHP Omega_c^N;
poles_lp=[];
for k= 1:(2*N)
     if(poles_total_real(k)<0)
         poles_lp=[poles_lp,poles_total_real(k)+j*poles_total_imag(k)];
     end
end
poles_lp;

% Preparing Transfer Function
syms s
num_lp(s)=Omega_c^N * (s^0);
denom_lp(s)=s^0;
real(poly(poles_lp));     %return polynomial coeffs whose roots are poles_lp
for k=1:N                 %for checking
   denom_lp(s)= (s-poles_lp(k))*denom_lp(s);
end
T_analog_lp(s)=num_lp(s)/denom_lp(s);


% % Lowpass to Bandpass Transformation
syms s_bp;
s= (s_bp^2 + Omega_0^2)/(B*s_bp);
T_analog_bp(s_bp) = T_analog_lp(s);


% Converting to Digital domain
% Let the variable be z^-1
% Hence s_bp -> (1-z^-1/1+z^-1)
% P.S : coeffs start from z^-n
syms invz
s_bp = (1-invz)/(1+invz);
T_digital_bp(invz)=T_analog_bp(s_bp);
[num,denom]=numden(T_digital_bp);
coeff_num=real(sym2poly(num));
coeff_den=real(sym2poly(denom));
freqz(coeff_num,coeff_den);
```

FIR Kaiser Window Design

```
% Filter number 73
% Navjot Singh
% 130110071
% FIR Filter using Kaiser Window
```

```
% Global Parameters
delta=0.15;
f_sample = 100000;
f_message = 45000;
t_width = 2000;

% Given analog frequencies corresponding to filter number
B_l= 14900;
B_h= 24900;
B_l_s= B_l - t_width;
B_h_s= B_h + t_width;
analog_specs= [B_l_s, B_l, B_h, B_h_s];

% Conversion to digital specification
digital_specs= (analog_specs*2*pi)/f_sample;

% Parameters for Kaiser Window
A = -1*20*log10(delta);
omega_t= min(digital_specs(2)-digital_specs(1),digital_specs(4)-digital_specs(3));
N_1 = (A-8)/(2*2.285*omega_t);
N = ceil(N_1)+5;  % Adjusted to get overshoot within Tolerance
if A>50
    alpha=0.1102*(A-8.7);
else
    if(A>=21)
        alpha=0.5842*((A-21)^0.4)+(0.07886*(A-21));
    else
        alpha=0;
    end
end
beta= alpha/N;

% Preparing Ideal Impulse Response
omega_c1=0.5*(digital_specs(1)+digital_specs(2));
omega_c2=0.5*(digital_specs(3)+digital_specs(4));
h_ideal= zeros((2*N)+1,1);
for k=-N:1:N
    h_ideal(k+N+1)=(sin(omega_c2*k)-sin(omega_c1*k))/(pi*k);
end
h_ideal(N+1)=(omega_c2-omega_c1)/pi;
h_ideal;
% Generate Kaiser window coefficients
k_window=kaiser((2*N)+1,alpha);

% Windowing the Ideal Impulse Response
h_filter=h_ideal.*k_window;
freqz(h_filter);
```

Bandstop Filter

IIR Chebyshev Design

```
% Filter number 73
% Navjot Singh
```

```
% 130110071
% Chebyshev

% Global Parameters
delta=0.15;
f_sample = 100000;
f_message = 45000;
t_width = 2000;

% Given analog frequencies corresponding to filter number
B_l= 16300;
B_h= 26300;
B_l_p= B_l - t_width;
B_h_p= B_h + t_width;
analog_specs= [B_l_p, B_l, B_h, B_h_p];

% Conversion to digital specification
digital_specs= (analog_specs*2*pi)/f_sample;

% Bilinear transformation to analog domain
% In the form (Omega_p1,Omega_s1,Omega_s2,Omega_p2)
analog_freq= tan(digital_specs/2);

% Calculation of paramteres
% For Bandstop to Lowpass transfromation
Omega_0= sqrt(analog_freq(1)*analog_freq(4));
B= analog_freq(4)-analog_freq(1);

% Bandpass to Lowpass transformation
% Using Omega_lp = B/((Omega_0^2/Omega_bp)- Omega_bp)
% Gives Passband at +-1
analog_lp_freq= B./(((Omega_0^2)./analog_freq)-analog_freq);
% Choosing the stringent of the two stopbands
Omega_lp_s= min(abs(analog_lp_freq(2)),abs(analog_lp_freq(3)));
Omega_lp_p= 1;

% Design of Chebyshev Filter
D1= 1/(1-delta)^2 -1;
D2= 1/delta^2 -1;
epsilon=sqrt(D1);
N_temp = (acosh(sqrt(D2)/epsilon))/(acosh(Omega_lp_s/Omega_lp_p));
N = ceil(N_temp);

% Poles of Chebyshev filter (all poles)
% Are of the form s= j*Omega_p*cos(A_k+jB_k)
% Hence real part = Omega_p*sin(A_k)*sinh(B_k)
% Hence imaginary part = Omega_p*cos(A_k)*cosh(B_k)
% Where A_k=(2k+1/2N)*pi
% and B_k= asinh(1/epsilon)/N
poles_total_real=zeros(1,2*N);
poles_total_imag=zeros(1,2*N);
for k= 0:1:(2*N)-1
    A_k = (((2*k)+1)*pi)/(2*N);
    B_k = asinh(1/epsilon)/N;
    poles_total_real(k+1) = Omega_lp_p*sin(A_k)*sinh(B_k);
    poles_total_imag(k+1) = Omega_lp_p*cos(A_k)*cosh(B_k);
```

```
end
poles_total=poles_total_real+j*poles_total_imag;

% Taking poles on the LHP ;
poles_lp=[];
for k= 1:(2*N)
     if(poles_total_real(k)<0)
         poles_lp=[poles_lp,poles_total_real(k)+j*poles_total_imag(k)];
     end
end

% Preparing Transfer Function
% Numerator depends upon parity on N
% To get the gain at Omega_p equal to 1
syms s
if(mod(N,2)==0)
    num_gain=1/sqrt(1+epsilon^2);
else
    num_gain=1;
end
num_lp(s)= (-1^N)*num_gain*prod(poles_lp)*(s^0);
denom_lp(s)=s^0;
real(poly(poles_lp));     %return polynomial coeffs whose roots are poles_lp
for k=1:N                 %for checking
   denom_lp(s)= (s-poles_lp(k))*denom_lp(s);
end
T_analog_lp(s)=num_lp(s)/denom_lp(s);


% % Lowpass to Bandstop Transformation
syms s_bs;
s= (B*s_bs)/(s_bs^2 + Omega_0^2);
T_analog_bs(s_bs) = T_analog_lp(s);



% Converting to Digital domain
% Let the variable be z^-1
% Hence s_bs -> (1-z^-1/1+z^-1)
% P.S : coeffs start from z^-n
syms invz
s_bs = (1-invz)/(1+invz);
T_digital_bs(invz)=T_analog_bs(s_bs);
[num,denom]=numden(T_digital_bs);
coeff_num=real(sym2poly(num)) ;
coeff_den=real(sym2poly(denom));
freqz(coeff_num,coeff_den);
```

FIR Kaiser Window Design

```
% Filter number 73
% Navjot Singh
% 130110071
% FIR Filter using Kaiser Window
```

```
% Global Parameters
delta=0.15;
f_sample = 100000;
f_message = 45000;
t_width = 2000;

% Given analog frequencies corresponding to filter number
B_l= 16300;
B_h= 26300;
B_l_p= B_l - t_width;
B_h_p= B_h + t_width;
analog_specs= [B_l_p, B_l, B_h, B_h_p];

% Conversion to digital specification
digital_specs= (analog_specs*2*pi)/f_sample;

% Parameters for Kaiser Window
A = -1*20*log10(delta);
omega_t= min(digital_specs(2)-digital_specs(1),digital_specs(4)-digital_specs(3));
N_1 = (A-8)/(2*2.285*omega_t);
N = ceil(N_1)+6;   % Adjusted to get overshoot within Tolerance
if A>50
    alpha=0.1102*(A-8.7);
else
    if(A>=21)
        alpha=0.5842*((A-21)^0.4)+(0.07886*(A-21));
    else
        alpha=0;
    end
end
beta= alpha/N;

% Preparing Ideal Impulse Response
omega_c1=0.5*(digital_specs(1)+digital_specs(2));
omega_c2=0.5*(digital_specs(3)+digital_specs(4));
h_ideal= zeros((2*N)+1,1);
for k=-N:1:N
    h_ideal(k+N+1)=-1*(sin(omega_c2*k)-sin(omega_c1*k))/(pi*k);
end
h_ideal(N+1)=1-(omega_c2-omega_c1)/pi;
h_ideal;
% Generate Kaiser window coefficients
k_window=kaiser((2*N)+1,alpha);

% Windowing the Ideal Impulse Response
h_filter=h_ideal.*k_window;
freqz(h_filter);
```