Item 3

```
close all; clear all; clc;
```

a. Windowing method

```
% According to the table, Hann window should be used since it provides at
% least 44 dB attenuation which is greater than A = 40 dB

ws1 = 0.1*pi;
wp1 = 0.2*pi;
wp2 = 0.6*pi;
ws2 = 0.8*pi;
As = 40; % minimum stopband attenuation in dB
Rp = 1; % maximum passband ripple in dB

% the minimum transition width-
% between the stopband and passband frequencies
tr_width = min((wp1-ws1), (ws2-wp2));

% The filter order M is computed by taking 11 times the-
% ratio of pi to the transition width (tr_width), and then-
% rounding up to the nearest integer. The +1 is added to-
% ensure an odd filter length.
M = ceil(11*pi/tr_width) + 1; % filter order
disp('The lowest order FIR filter that will meet the desired specifications
is a Hanning window of order:');
```

```
The lowest order FIR filter that will meet the desired specifications is a Hanning window of order:
```

```
disp(M);
```

```
   111
```

```


% Calculate the Hann windowed impulse response
n = 0:M-1;
wc1 = (ws1 + wp1)/2;
wc2 = (wp2 + ws2)/2;
hd = ideal_lp(wc2, M) - ideal_lp(wc1, M);
whann = (hann(M))';
h = hd .* whann;

% Compute numerator coefficients
b = impz(h, 1);
disp('The numerator coefficients, B:');
```

```
The numerator coefficients, B:
```

```
disp(b);
```

```
      0
-0.0000
-0.0000
 0.0001
 0.0000
 0.0001
 0.0003
-0.0001
 0.0002
 0.0001
-0.0010
-0.0003
-0.0006
-0.0018
 0.0004
-0.0000
-0.0006
 0.0032
 0.0014
 0.0009
 0.0045
-0.0008
-0.0016
 0.0013
-0.0070
-0.0046
-0.0004
-0.0085
 0.0009
 0.0060
-0.0021
 0.0122
 0.0113
-0.0013
 0.0130
-0.0000
-0.0155
 0.0019
-0.0190
-0.0247
 0.0051
-0.0173
-0.0032
 0.0362
 0.0021
 0.0293
 0.0562
-0.0137
 0.0203
 0.0144
-0.1065
-0.0285
-0.0715
-0.2792
 0.1129
 0.5500
 0.1129
-0.2792
-0.0715
-0.0285
-0.1065
 0.0144
 0.0203
-0.0137
```

```
    0.0562
    0.0293
    0.0021
    0.0362
   -0.0032
   -0.0173
    0.0051
   -0.0247
   -0.0190
    0.0019
   -0.0155
   -0.0000
    0.0130
   -0.0013
    0.0113
    0.0122
   -0.0021
    0.0060
    0.0009
   -0.0085
   -0.0004
   -0.0046
   -0.0070
    0.0013
   -0.0016
   -0.0008
    0.0045
    0.0009
    0.0014
    0.0032
   -0.0006
   -0.0000
    0.0004
   -0.0018
   -0.0006
   -0.0003
   -0.0010
    0.0001
    0.0002
   -0.0001
    0.0003
    0.0001
    0.0000
    0.0001
   -0.0000
   -0.0000
         0
```

```
% Calculate frequency response
[dbwin, mag, pha, grd, wwin] = freqz_m(b, 1);

% Plot the magnitude response
plot(wwin/pi, dbwin);
title('Magnitude Response in dB');
grid on;
xlabel('Frequency (pi units)');
ylabel('Magnitude (dB)');
ylim([-100, 50]);
```

b. Frequency sampling method

```
M = 40;
alphafs = (M-1)/2;
l = 0:M-1;
wl = (2*pi/M)*l; %frequency grid

%According to Ingles and Proakis, the optimum values of T1-
% and T2 for M = 40 and seven samples in the passband are:
T1 = 0.109021;
T2 = 0.59417456;

Hrs=[zeros(1,5),T1,T2,ones(1,7),T2,T1,zeros(1,9),T1,T2,ones(1,7),T2,T1,zeros(
1,4)];
Hdr = [0,0,1,1,0,0];
% Hdr represents the desired frequency response in the-
% passband region for a type-II linear phase FIR filter.-
% Specifically, Hdr is a vector that describes the desired-
% response in the passband region using binary values.
wdl = [0,0.1,0.2,0.6,0.8,1]; % define the frequency edges

% determine the indices k1 and k2 for the angle calculation
k1 = 0:floor((M-1)/2);
k2 = floor((M-1)/2)+1:M-1;
angH = [-alphafs*(2*pi)/M*k1, alphafs*(2*pi)/M*(M-k2)];

H = Hrs.*exp(j*angH); % frequency response
h = real(ifft(H,M)); % length of the impulse response
[dbfreqsam,mag,pha,grd,wfreqsam] = freqz_m(h,1);
[Hr,ww,a,L] = Hr_Type2(h);

disp('The numerator coefficients, B:');
```

```
The numerator coefficients, B:
```

```
disp(h);
```

```
   -0.0012    0.0024    0.0041   -0.0049   -0.0032   -0.0017   -0.0092    0.0165    0.0193   -0.0135    0.
```

```

% Plot the magnitude response
plot(wfreqsam/pi, dbfreqsam);
title('Magnitude Response in dB');
grid on;
xlabel('Frequency (pi units)');
ylabel('Magnitude (dB)');
```

```
ylim([-100, 50]);
```

```
% To verify that the filter meets the specs, the passband ripple is
% computed
delta_w = 2*pi/1000; % Frequency resolution
% Extract magnitude response within passband range
passband_mag = mag(round(wp1/delta_w)+1:round(wp2/delta_w));
% Calculate actual passband ripple
Rpfreqsamp = round(max(passband_mag) - min(passband_mag), 1); % Actual
passband ripple
disp('The actual passband ripple is:');
```

```
The actual passband ripple is:
```

```
disp(Rpfreqsamp);
```

```
     1
```

```
% The designed filter does meet the specs.
```

Compared to the design using the windowing method, frequency sampling method has lower filter order. Windowing methods such as the Hann require designing the desired frequency response by multiplying an ideal impulse response with a window function. On the other hand, the frequency sampling method requires directly specifying the desired frequency response at specific frequency points and then performing an inverse Fourier transform to get the impulse response.

The windowing method is more flexible in shaping the frequency response as it permits choice between different window functions and and adjustments of parameters.

The frequency sampling method allows for pinpoint control over the desired frequency response by specifying it at specific frequency points. However, it may be less flexible in terms of shaping complex frequency responses.

In the windowing method, the magnitude response gradually transitions from the passband to the stopband. Meanwhile, the magnitude response of the frequency-sampled FIR filter is exact at the specified frequency points and demonstrates sharp transitions between the passband and the stopband.

c. Chebyshev-I

```
 wpche1 = [0.2,0.6];
 wsche1 = [0.1 ,0.8];
 Rp = 1;
 As = 40;
 [Nche1,wcche1] = cheb1ord(wpche1,wsche1,Rp,As);
 [Bche1 ,Ache1] = cheby1(Nche1,Rp,wcche1, "bandpass");
 disp('The lowest-order Chebyshev-I filter has an order of:');
```

```
The lowest-order Chebyshev-I filter has an order of:
```

```
disp(Nche1);
```

```
     4
```

```
disp('The numerator coefficients, B:');
```

```
The numerator coefficients, B:
```

```
disp(Bche1);
```

```
    0.0243         0    -0.0970         0    0.1456         0    -0.0970         0    0.0243
```

```
disp('The denominator coefficients, A:');
```

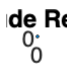
```
The denominator coefficients, A:
```

```
disp(Ache1);
```

```
    1.0000   -2.1381    3.4271   -3.9061    4.0313   -3.0253    1.9903   -0.8593    0.3074
```

```
 % Calculate the frequency response
[hche1, wche1] = freqz(Bche1, Ache1);

% Plot the magnitude response
figure;
plot(wche1/pi, abs(hche1));
title('Magnitude Response');
xlabel('Frequency (pi units)');
ylabel('Magnitude');
grid on;
```



d. Chebyshev-I lowpass filter prototype

```
 wpche1 = [0.2*pi,0.6*pi];
 wsche1 = [0.1*pi ,0.8*pi];
 Rp = 1;
 As = 40;

centerfreq = (wpche1(2) + wpche1(1)) / 2;
[Bche1LP, Ache1LP] = cheby1(Nche1, Rp, centerfreq, 's');

 disp('The numerator coefficients, B:');
```

```
The numerator coefficients, B:
```

```
disp(Bche1LP);
```

```
        0         0         0         0    0.6126
```

```
 disp('The denominator coefficients, A:');
```

```
The denominator coefficients, A:
```

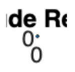
```
 disp(Ache1LP);
```

```
    1.0000    1.1973    2.2959    1.4737    0.6873
```

```

[hche1LP, wche1LP] = freqs(Bche1LP, Ache1LP);

% Plot the magnitude response
figure;
plot(wche1LP/pi, abs(hche1LP));
title('Magnitude Response');
xlabel('Frequency (pi units)');
ylabel('Magnitude');
grid on;
```



e. derive the desired digital bandpass filter

```
bwBP = wpche1(2) - wpche1(1);


[BBP, ABP] = lp2bp(Bche1LP, Ache1LP, centerfreq, bwBP);
[hBP, wBP] = freqs(BBP, ABP);
figure(); plot(wBP/pi, abs(hBP));
title('LP to BP Magnitude Response')
xlabel('Frequency (pi units)');
ylabel('Magnitude');
grid on;
```

```

Fs=0.5; %2 Hz
[BZb, AZb] = bilinear(BBP, ABP, Fs);
 disp('The numerator coefficients, B:');
```

```
The numerator coefficients, B:
```

```
 disp(BZb);
```

```
    0.0148    0.0000   -0.0591   -0.0000    0.0886   -0.0000   -0.0591   -0.0000    0.0148
```

```
 disp('The denominator coefficients, A:');
```

```
The denominator coefficients, A:
```

```
disp(AZb);
```

```
    1.0000    1.3469    2.7138    2.5116    3.1585    1.9904    1.7053    0.6089    0.3550
```

```

[Hzb, Wzb] = freqz_m(BZb, AZb);
plot(Wzb, Hzb)
title('Bilinear Transformation technique');
xlabel('Normalized Frequency (*pi rad/sample)');
ylabel('Magnitude (dB)');
grid on;
```

Judging from the plot after the Analog-to-digital transformation, there is an error in my procedure that caused the filter to not meet the specs. Troubleshooting is necessary to meet the specs as well as adjusting the center frequency and the bandwidth.

Stacking the curves:

```
figure();

subplot(311);
plot(wwin/pi, dbwin);
title('Windowing Method Magnitude Response');
grid on;
xlabel('Frequency (pi units)');
ylabel('Magnitude (dB)');
ylim([-100, 50]);

subplot(312);
plot(wfreqsam/pi, dbfreqsam);
title('Frequency Sampling Magnitude Response');
grid on;
xlabel('Frequency (pi units)');
ylabel('Magnitude (dB)');
ylim([-100, 50]);

subplot(313);
plot(wche1/pi, abs(hche1));
title('Digital Chebyshev-I Magnitude Response');
xlabel('Frequency (pi units)');
ylabel('Magnitude');
grid on;
```

```

figure();
subplot(311);
```

```
plot(wBP/pi, abs(hBP));
title('LP to BP Magnitude Response')
xlabel('Frequency (pi units)');
ylabel('Magnitude');
xlim([0,1]);
grid on;
```

Functions

```
function hd = ideal_lp(wc,M)
% Ideal LowPass filter computation
% --------------------------------
% [hd] = ideal_lp(wc,M)
% hd = ideal impulse response between 0 to M-1
% wc = cutoff frequency in radians
% M = length of the ideal filter
%
alpha = (M-1)/2; n = [0:1:(M-1)];
m=n- alpha; fc = wc/pi; hd = fc*sinc(fc*m);
end

function [db,mag,pha,grd,w] = freqz_m(b,a);
% Modified version of freqz subroutine
% ------------------------------------
% [db,mag,pha,grd,w] = freqz_m(b,a);
% db = Relative magnitude in dB computed over 0 to pi radians
% mag = absolute magnitude computed over 0 to pi radians
% pha = Phase response in radians over 0 to pi radians
% grd = Group delay over 0 to pi radians
% w = 501 frequency samples between 0 to pi radians
% b = numerator polynomial of H(z) (for FIR: b=h)
% a = denominator polynomial of H(z) (for FIR: a=[1])
%
[H,w] = freqz(b,a,1000,'whole');
H = (H(1:1:501))'; w = (w(1:1:501))';
mag = abs(H); db = 20*log10((mag+eps)/max(mag));
pha = angle(H); grd = grpdelay(b,a,w);
end

function [Hr,w,b,L] = Hr_Type2(h);
% Computes Amplitude response of a Type-2 LP FIR filter
% -----------------------------------------------------
% [Hr,w,b,L] = Hr_Type2(h)
% Hr = Amplitude Response
% w = frequencies between [0 pi] over which Hr is computed
% b = Type-2 LP filter coefficients
% L = Order of Hr
% h = Type-2 LP impulse response
%
M = length(h); L = M/2;
b = 2*[h(L:-1:1)]; n = [1:1:L]; n = n-0.5;
w = [0:1:500]'*pi/500; Hr = cos(w*n)*b';
end
```

References:

Ingle, V. K., & Proakis, J. G. (2011). *Digital signal processing using MATLAB*. Cengage Learning.