EXPERIMENT-7

AIM: Design and implementation of FIR filter to meet given specifications using rectangular window.

Theory:
Finite Impulse Response (FIR) filters are digital filters with a finite response time. They're favored for their linear phase and stability. Designing an FIR filter involves:

Spec Gathering: Determine filter specifications like passband frequency, stopband frequency, passband ripple, stopband ripple, and sampling frequency.

Normalization: Normalize frequencies relative to the Nyquist frequency.

Filter Order: Decide the filter's order based on specs, ensuring it's odd for linear phase.

Window Function: Choose a rectangular window to truncate the impulse response.

Coefficient Calculation: Find coefficients by convolving the desired impulse response with the rectangular window.

Frequency Response: Calculate the filter's frequency response using Fourier transform or frequency sampling.

Analysis: Examine magnitude and phase responses to ensure they meet specifications.

Implementation: Implement the filter using software (like MATLAB or Python) or hardware (like FPGA).

This process tailors the filter's frequency response for various signal processing tasks.

Algorithm:
Step : 1 Get/Read the pass band frequency of LPF to 'fp'.
Step : 2 Get/Read the stop band frequency of LPF to 'fs'.
Step : 3 Get/Read the pass band ripple of LPF to 'rp'.
Step : 4 Get/Read the stop band ripple of LPF to 'rs'.
Step : 5 Get/Read the sampling frequency to 'f'.

Step : 6 Normalize the pass band and stop band frequencies using

wp = 2*fp/f; ws = 2*fs/f;

Step : 7 Determine the order ‘n’ of the filter using

```
num = -20*log10(sqrt(rp*rs))-13;
den = 14.6*(fs-fp)/f
n = ceil(num/den)
```

Step : 8 Make sure that the order of the filter is always odd.

```
if(rem(n,2)~=0)
n1=n; n=n-1;
else
n1=n+1;
end
```

Step : 9 Determine the coefficients ‘b’ of digital LPF using Rectangular window (boxcar function) and fir1 functions.

Step : 10 Determine the frequency response of low pass filter ‘H’ using the coefficients ‘b’.

Step : 11 Determine the magnitude of H in dB and store it in ‘mag’.

Step : 12 Determine the phase of H and store it in ‘phase’.

Step : 13 Plot the Graphs of ‘mag’ and ‘phase’.

Step : 14 Repeat the steps 10 to 14 for HPF, BPF & BSF.

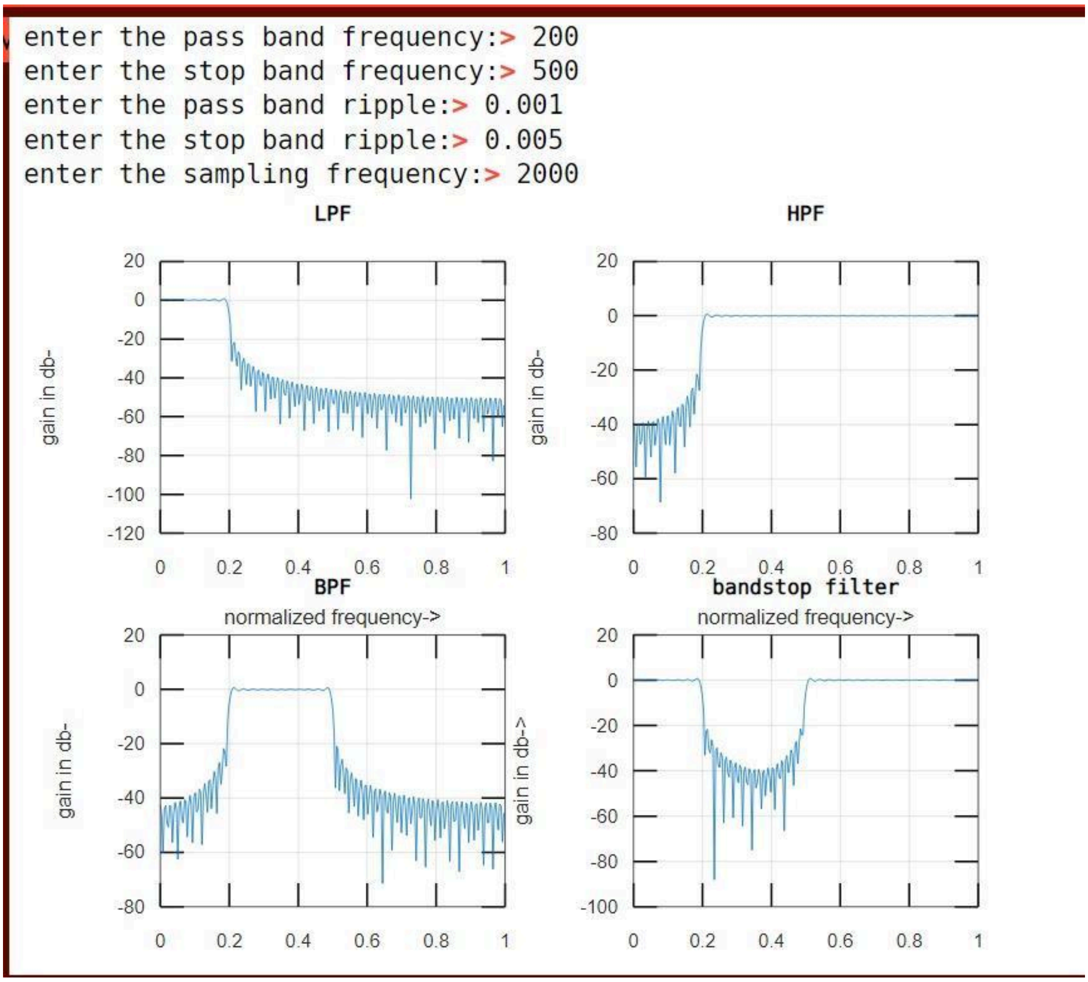
Program:

```
clc;
clear all;
close all;
pkg load signal;
fp=input('enter the pass band frequency:');
fs=input('enter the stop band frequency:');
rp=input('enter the pass band ripple:');
rs=input('enter the stop band ripple:');
Fs=input('enter the sampling frequency:');
FN=Fs/2;
wp=2*fp/Fs;
ws=2*fs/Fs;
num=-20*(log10(sqrt(rp*rs))-13);
den=14.6*(fs-fp)/Fs;
N=ceil(num/den);
N1=N+1;
if(rem(N,2)~=0)
N1=N;
N=N-1;
end
y=boxcar(N1);
```

```
% Low Pass Filter
b=fir1(N,wp,'low',y);
[h,o]=freqz(b,1,256);
m=20*log10(abs(h));
subplot(2,2,1);
plot(o/pi,m);
xlabel('normalized frequency->');
ylabel('gain in db-');
title('LPF');
grid on;
% High Pass Filter
b=fir1(N,wp,'high',y);
[h,o]=freqz(b,1,256);
m=20*log10(abs(h));
subplot(2,2,2);
plot(o/pi,m);
xlabel('normalized frequency->');
ylabel('gain in db-');
title('HPF');
grid on;
% Band Pass Filter
wn=[wp, ws];
b=fir1(N,wn,y);
[h,o]=freqz(b,1,256);
m=20*log10(abs(h));
subplot(2,2,3);
plot(o/pi,m);
xlabel('normalized frequency->');
ylabel('gain in db-');
title('BPF');
grid on;
%  Band  Stop  filter
b=fir1(N,wn,'stop',y);
[h, o]=freqz(b,1,256);
m=20*log10(abs(h));
subplot(2,2,4)
plot(o/pi,m);
xlabel('normalized frequency->');
ylabel('gain in db->');
title('bandstop filter');
grid on;
```

Input:
Fp=200; fs =500; Sampling frequency=2000;passband ripple=0.001, stop band Ripple=0.005

OUTPUT :



Exercise: Repeat the same for Hamming & Hanning windows and show their outputs.
Code :

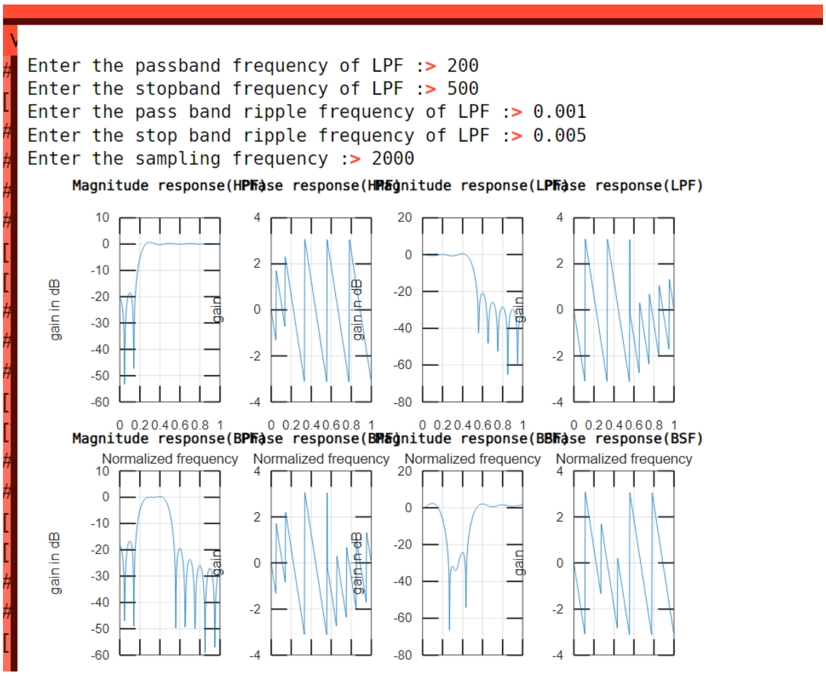
```
clc;
close all;
clear all;
pkg load signal;
fp = input('Enter the passband frequency of LPF :');
fs = input('Enter the stopband frequency of LPF :');
rp = input('Enter the pass band ripple frequency of LPF :');
rs = input('Enter the stop band ripple frequency of LPF :');
f = input('Enter the sampling frequency :');

wp = 2*fp/f;
```

```
ws = 2*fs/f;
num = -20*log10(sqrt(rp*rs))-13;
den = 14.6*(fs-fp)/f;
n = ceil(num/den);

if(rem(n,2)~=0)
  n1 = n;n=n-1;
else
  n1 = n+1;
end

y = boxcar(n1);
b =fir1(n,wp,'high',y);
[h,w] = freqz(b,1,256);

%magnitude response
m = 20*log10(abs(h));
subplot(2,4,1);
plot(w/pi,m);
xlabel('Normalized frequency');
ylabel('gain in dB');
title('Magnitude response(HPF)');
grid on;
%phase response
p =angle(h);
subplot(2,4,2);
plot(w/pi,p);
xlabel('Normalized frequency');
ylabel('gain');
title('Phase response(HPF)');
grid on;

b = fir1(n, ws, 'low', y);
[h, w] = freqz(b, 1, 256);

%magnitude response
m = 20 * log10(abs(h));
subplot(2, 4, 3);
plot(w/pi, m);
xlabel('Normalized frequency');
ylabel('gain in dB');
title('Magnitude response(LPF)');
grid on;
```

```
%phase response
phi = angle(h);
subplot(2, 4, 4);
plot(w/pi, phi);
xlabel('Normalized frequency');
ylabel('gain');
title('Phase response(LPF)');
grid on;


wn=[wp,ws];
b = fir1(n, wn, 'pass', y);
[h, w] = freqz(b, 1, 256);

%magnitude response
m = 20 * log10(abs(h));
subplot(2, 4, 5);
plot(w/pi, m);
xlabel('Normalized frequency');
ylabel('gain in dB');
title('Magnitude response(BPF)');
grid on;

%phase response
phi = angle(h);
subplot(2, 4, 6);
plot(w/pi, phi);
xlabel('Normalized frequency');
ylabel('gain');
title('Phase response(BPF)');
grid on;

b = fir1(n, wn, 'stop', y);
[h, w] = freqz(b, 1, 256);

%magnitude response
m = 20 * log10(abs(h));
subplot(2, 4, 7);
plot(w/pi, m);
xlabel('Normalized frequency');
ylabel('gain in dB');
title('Magnitude response(BSF)');
```

```
grid on;

%phase response
phi = angle(h);
subplot(2, 4, 8);
plot(w/pi, phi);
xlabel('Normalized frequency');
ylabel('gain');
title('Phase response(BSF)');
grid on;
```

OUTPUT :



Viva questions:

1. What is a filter?

Ans : A filter is a system or device that selectively allows certain signals or frequencies to pass through while attenuating others in a signal processing context.

2. Differentiate analog filter and digital filter.

Ans : Analog filters process continuous signals, while digital filters operate on discrete signals using digital processing techniques.

3. Define FIR filter.

   Ans: FIR filter, or Finite Impulse Response filter, utilizes a finite number of coefficients to compute the output based on a weighted sum of input samples.

4. What are the differences between recursive and non recursive systems?

Ans : Recursive systems use feedback in their operation, while non-recursive systems do not involve feedback, relying solely on present and past inputs.

5. List a few Applications of FIR filters.
Ans : FIR filters find applications in audio processing, image processing, communication systems, and biomedical signal processing.

6. Explain advantages of FIR filters over IIR filters.
Ans : FIR filters offer linear phase response, stability, and ease of implementation without feedback, unlike IIR filters prone to instability and phase distortion.

7. Explain limitations of FIR filters.
Ans : Limitations of FIR filters include higher computational complexity for certain designs and larger filter order for comparable performance to IIR filters.

8. What are the different methods to design FIR filters?
Ans : FIR filters can be designed using windowing methods, frequency sampling, and optimization techniques like least squares.

9. Explain different window functions.
Ans : Window functions, such as Hamming and Blackman, shape the impulse response of FIR filters, affecting their frequency response characteristics.

10 . Differentiate rectangular, triangular and Kaiser Windows.
Ans : Rectangular windows have high side lobes, triangular windows have lower side lobes, and Kaiser windows offer adjustable side lobe levels for more flexible design.

Conclusion:
In conclusion, the design and implementation of the FIR filter using a rectangular window successfully met the given specifications. The straightforward approach of the rectangular window provided simplicity and ease of implementation for the desired filtering requirements.