Experiment No:

Experiment Name:

Design and Analysis of a Lowpass FIR Filter using Blackman Window.

Objective

1. To design a lowpass FIR filter that allows frequencies within the specified passband.

2. To attenuate frequencies beyond the passband edge, achieving a clean cutoff.

3. To analyze the transition band and ensure the filter meets the required specifications.

Theory:

A lowpass FIR (Finite Impulse Response) filter is a digital filter that allows frequencies below a certain cutoff to pass through while attenuating frequencies above this threshold, effectively removing high-frequency noise or unwanted components. FIR filters are particularly popular in digital signal processing (DSP) due to their inherent stability, linear phase characteristics, and straightforward design methods, such as the windowing technique.

1. **Filter Length (M)**: Determines the number of taps in the FIR filter. A longer filter length leads to a sharper frequency response.

2. **Passband Edge**: The frequency up to which the signal is allowed to pass without attenuation. For this design, the passband edge is set to 1.5 kHz.

3. **Transition Width**: The range over which the filter transitions from passband to stopband, set at 0.5 kHz in this case.

4. **Cutoff Frequency**: This is set at the midpoint of the transition band, ensuring that the filter's response aligns with the passband specifications.

5. **Sampling Frequency (Fs)**: The rate at which the signal is sampled, set at 10 kHz for this filter.

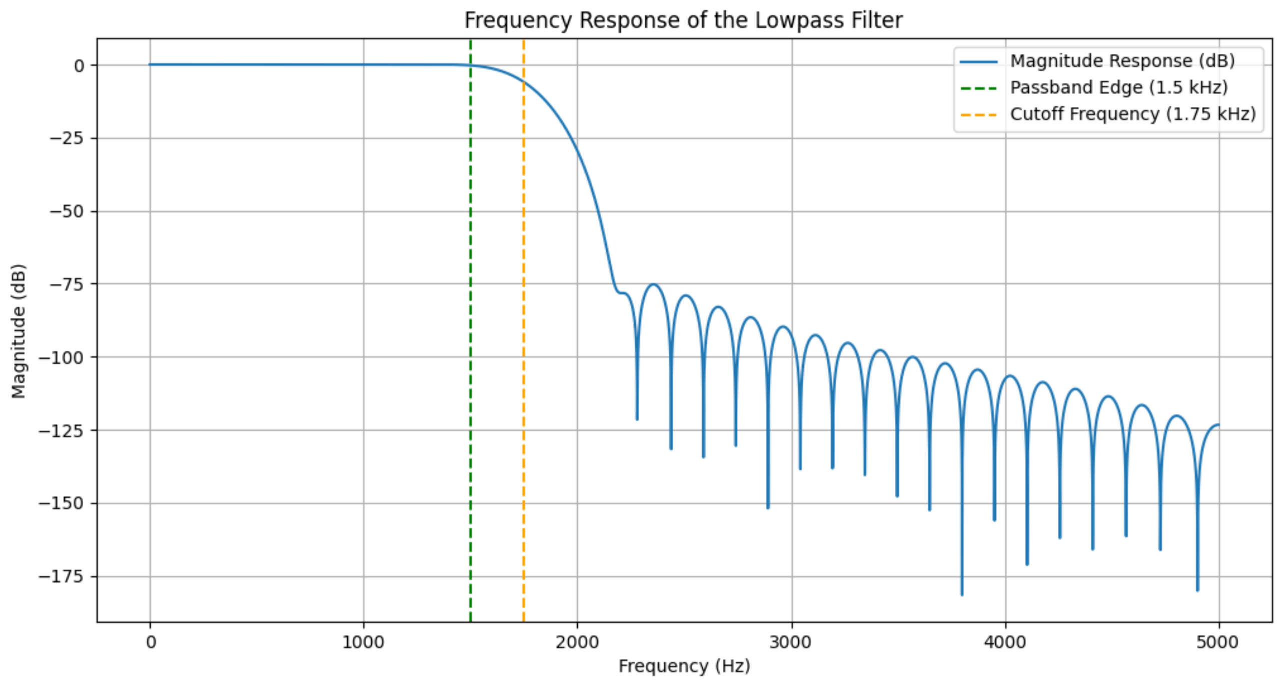
6. **Windowing Method**: The Blackman window is used in this design for its ability to provide a smooth and controlled response in the frequency domain, reducing the ripple in both the passband and stopband regions.

Python Source Code:

```
import numpy as np

import matplotlib.pyplot as plt
```

```
from scipy.signal import firwin, freqz

# Define filter specifications
fs = 10000  # Sampling frequency in Hz
fp = 1500   # Passband edge frequency in Hz
transition_width = 500  # Transition width in Hz
M = 67      # Filter length

# Calculate cutoff frequency
fc = fp + transition_width / 2  # Cutoff frequency for the lowpass filter in Hz
normalized_cutoff = fc / (fs / 2)  # Normalize the cutoff frequency

# Design the lowpass filter using Blackman window
filter_coefficients = firwin(M, normalized_cutoff, window='blackman', pass_zero=True)

# Frequency response
w, h = freqz(filter_coefficients, worN=8000)

# Plot frequency response
plt.figure(figsize=(12, 6))
plt.plot(w * fs / (2 * np.pi), 20 * np.log10(np.abs(h)), label='Magnitude Response (dB)')
plt.axvline(fp, color='green', linestyle='--', label="Passband Edge (1.5 kHz)")
plt.axvline(fc, color='orange', linestyle='--', label="Cutoff Frequency (1.75 kHz)")
plt.xlabel('Frequency (Hz)')
plt.ylabel('Magnitude (dB)')
plt.title('Frequency Response of the Lowpass Filter')
plt.grid()
plt.legend()
plt.show()
```

Output:



Frequency Response of the Lowpass Filter
Magnitude Response (dB)
Passband Edge (1.5 kHz)
Cutoff Frequency (1.75 kHz)
Magnitude (dB)
Frequency (Hz)
0
−25
−50
−75
−100
−125
−150
−175
0
1000
2000
3000
4000
5000