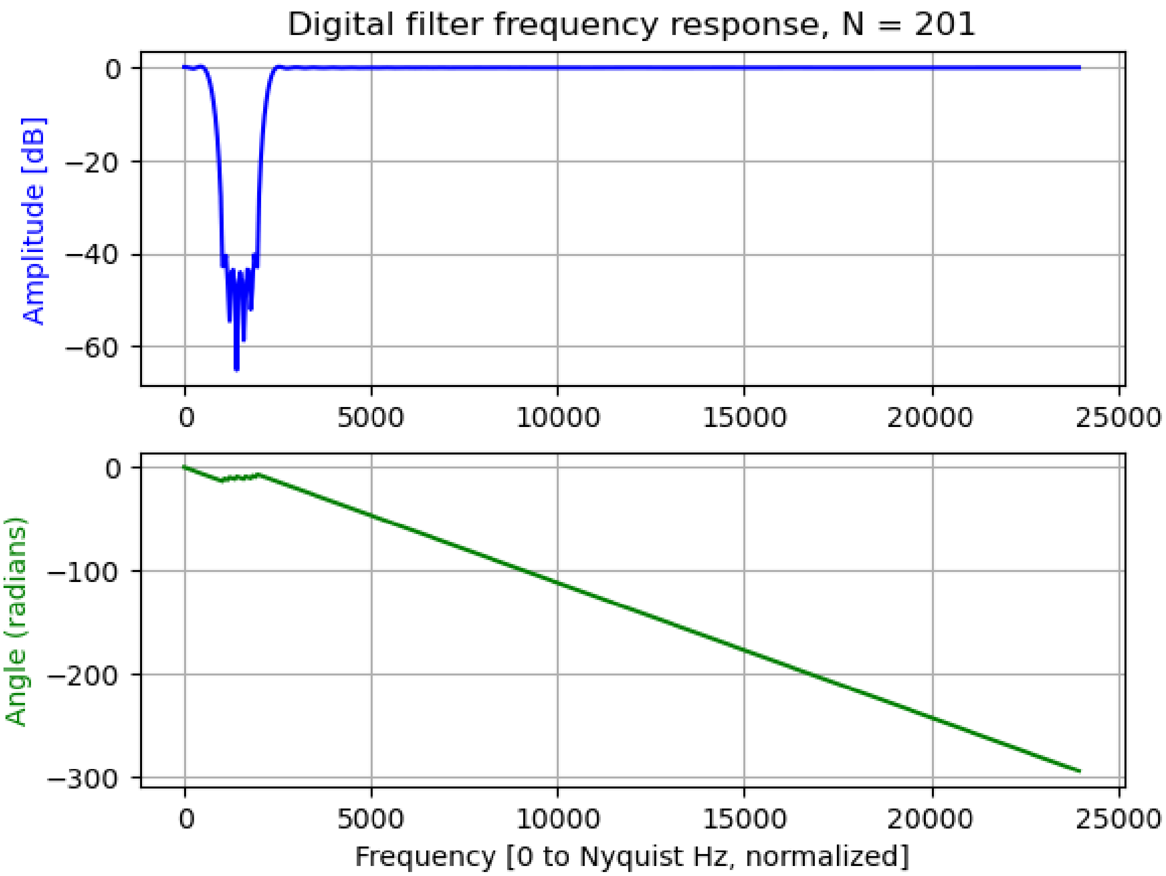
In [7]:

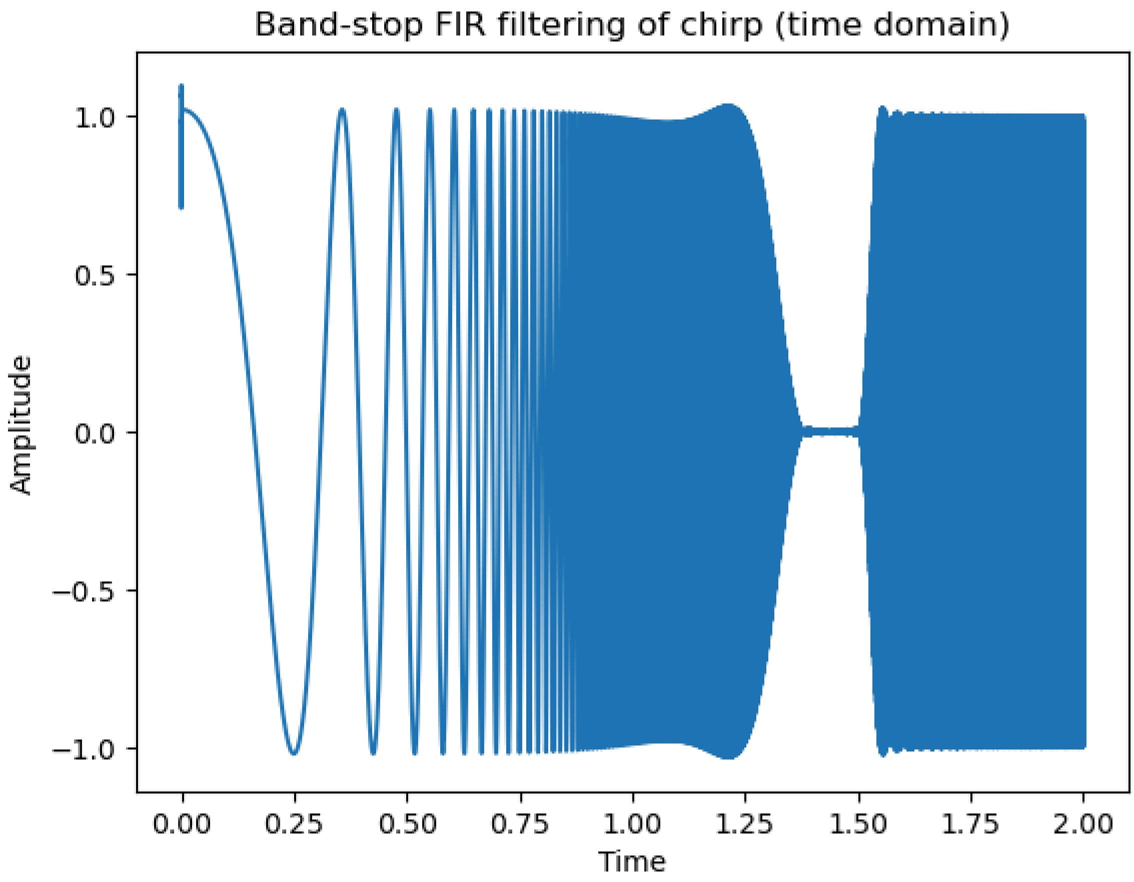
```
import numpy as np
import matplotlib.pyplot as plt
from scipy import signal

# Your filter design here
# firls() can be called via signal.firls()

#larger numtaps=longer FIR filter

#Question 1:
# Using a longer filter improves performance by providing a sharper transition
#However, it also increases computational complexity and latency, relating to t

#Question 2:
#With an order of 201, I observed that I could make transition bands up to 600H
#meet the specifications - frequencies between 1k and 2khz attenuated below -20
#all other frequencies passing through with approximately unity gain.
#If I make the left (<1000Hz-1000Hz) transition band wider than 600Hz, I notice
#1000Hz start to be attenuated below 0dB. If I make the right transition band (
#I notice frequencies above 2000Hz still attenuated below 0dB, and unity gain f
#If I make the right transition band wider than 1000Hz, I notice jumps in ampli

Fs = 48000
nyq = Fs / 2
numtaps = 201
desired = [1,1,0,0,1,1]
weights = [1,10,1]

bands = [0, 600, 1000, 2000, 2400, nyq]
b = signal.firls(numtaps, bands, desired, weights, fs=Fs)

# Signal analysis
w, h = signal.freqz(b, fs=Fs)

plt.figure()
plt.subplot(2,1,1)
plt.title('Digital filter frequency response, N = ' + str(len(b)))
plt.plot(w, 20 * np.log10(abs(h)), 'b') #get rid of pi if setting fs in freqz
plt.ylabel('Amplitude [dB]', color='b')
plt.grid()
plt.axis('tight')

plt.subplot(2,1,2)
angles = np.unwrap(np.angle(h))
plt.plot(w, angles, 'g')
plt.ylabel('Angle (radians)', color='g')
plt.grid()
plt.axis('tight')
plt.xlabel('Frequency [0 to Nyquist Hz, normalized]')
plt.show()
```

```
/var/folders/nl/k9vvzpz57vj2dcsjwk1pfq780000gp/T/ipykernel_93827/303683386.p
y:30: DeprecationWarning: You are passing weight=[1, 10, 1] as a positional a
rgument. Please change your invocation to use keyword arguments. From SciPy
1.14, passing these as positional arguments will result in an error.
  b = signal.firls(numtaps, bands, desired, weights, fs=Fs)
```



In [9]:

```
import numpy as np
import matplotlib.pyplot as plt
from scipy import signal


F_s = 48000
t = [i / F_s for i in range(2 * F_s)]
test_data = signal.chirp(t, 1, t[-1], 24000, method='logarithmic')


x = np.asarray(test_data, dtype=float)
y = np.convolve(x, b, mode='same')





plt.plot(t, y)
plt.xlabel('Time')
plt.ylabel('Amplitude')
plt.title('Band-stop FIR filtering of chirp (time domain)')
plt.show()
```


Band-stop FIR filtering of chirp (time domain)
Amplitude
Time