German University in Cairo
الجامعة الألمانية بالقاهرة

Signals and Systems Theory

MATLAB Project

Presented by:
Dr. Haitham Omran
Electronics Department

Email: haitham.omran@guc.edu.eg

رجب 1437 هـ
Spring 2016

- First Part:
  - Oktay text book: Chapter 6: pages 595-596
  - Objectives
    - Learn how to handle a sampled audio file on MATLAB
    - Plot the signal in time domain with correct scaling
    - Learn how to find the frequency content of the audio file using MATLAB Fast Fourier Transform function FFT.
    - Explore the concept of Aliasing and it effect on sampled signal

Part A

MATLAB Projects

6.30. In this project we will explore the concept of aliasing especially in the way it exhibits itself in an audio waveform. MATLAB has a built-in sound file named "`handel`" which contains a recording of Handel's Hallelujah Chorus. It was recorded with a sampling rate of $f_s = 8192$ Hz. Use the statement

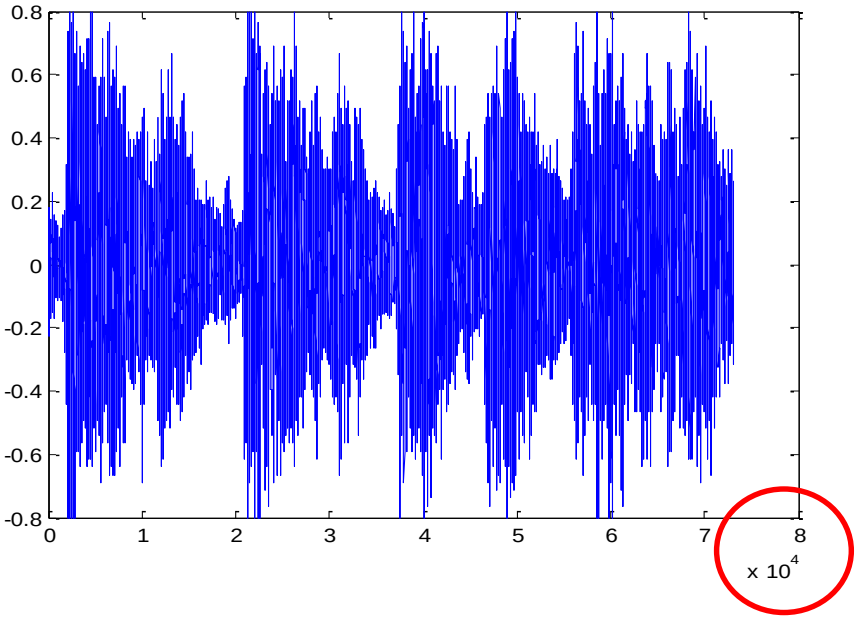
```
>>  load handel
```

which loads two new variables named "`Fs`" and "`y`" into the workspace. The scalar variable "`Fs`" holds the sampling rate, and the vector "`y`" holds 73113 samples of the recording that corresponds to about 9 seconds of audio. Once loaded, the audio waveform may be played back with the statement

```
>>  sound(y,Fs)
```

Part A

a. Graph the audio signal as a function of time. Create a continuous time variable that is correctly scaled for this purpose.



Note: The x axis should be scaled to reflect time in sec

Part A

b. Compute and graph the frequency spectrum of the signal using the function `fft(..)`. Create an appropriate frequency variable to display frequencies in Hertz, and use it in graphing the spectrum of the signal. Refer to the discussion in Section 5.8.5 of Chapter 5 for using the DFT to approximate the continuous Fourier transform.

Notes:

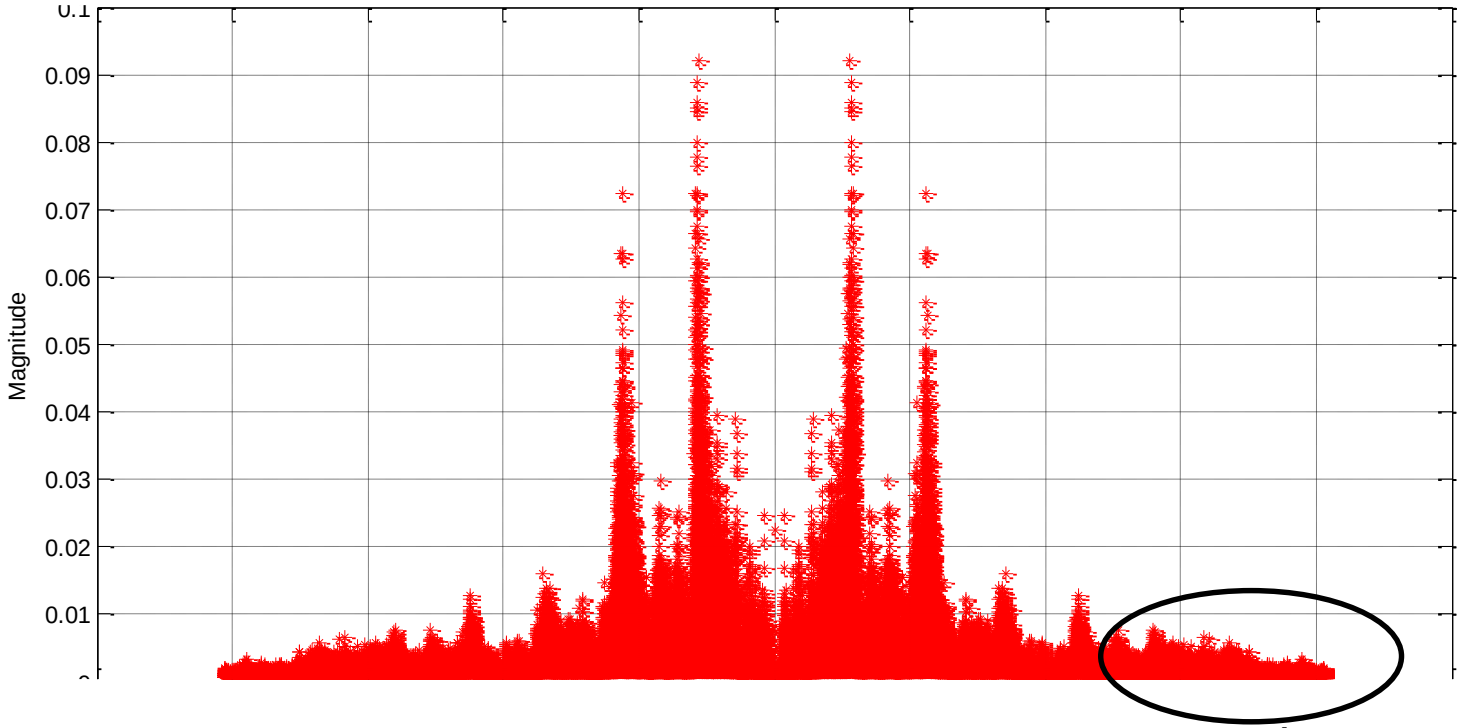
- The signal is sampled at N equally spaced time instants in the interval $0 \leq t \leq t_1$, and thus NT = $t_1$where T is the sampling period.

- Based on the Nyquist sampling theorem, only the first half of the FFT samples represent an approximation to the transform in the positive frequency range$0 \leq \omega < \omega s/2$.

- The second half of the DFT samples represent an approximation in the negative frequency range $-\omega s/2 \leq \omega < 0$.

Revise Matlab Exersice 5.14 for importance

Part A



Note: The x axis should be scaled to reflect the correct frequency in Hz

Part A

c. Downsample the audio signal in vector "y" using a downsampling rate of $D = 2$. Do not use an anti-aliasing filter for this part. Play back the resulting signal using the function `sound(..)`. Be careful to adjust the sampling rate to reflect the act of downsampling or it will play too fast.

Note: Find the MATLAB function to perform down sampling.

d. Repeat part (c) this time using an anti-aliasing filter. A simple Chebyshev type-I lowpass filter may be designed with the following statement (see Chapter 10 for details):

```
>> [num,den] = cheby1(5,1,0.45)
```

Process the audio signal through the anti-aliasing filter using the statement

```
>> yfilt = filter(num,den,y);
```

Part A

The vector "`yfilt`" represents the signal at the output of the anti-aliasing filter. Downsample this output signal using $D = 2$ and listen to it. How does it compare to the sound obtained in part (c)? How would you explain the difference between the two sounds?

e. Repeat parts (c) and (d) using a downsampling rate of $D = 4$. The anti-aliasing filter for this case should be obtained by

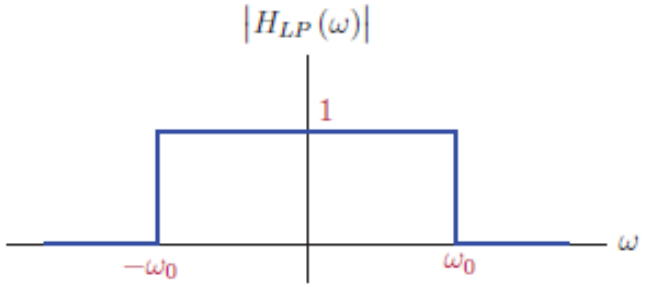
```
>>  [num,den] = cheby1(5,1,0.23)
```

- Second Part
  - Elaboration on Oktay project where the previous audio file Handel has been mixed with some noise tones
- Objective
  - Find out the tones that has been mixed with the voice file using the FFT script developed in part one
  - Filtering our the original signal from the noisy one by using a filter generated using the MATLAB interactive tool fdatool

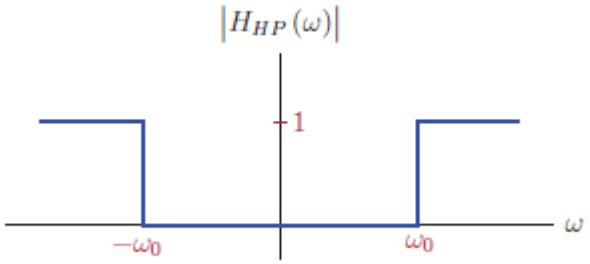
Part B

- The file handel_mix on eee site is has been mixed with three artificial tunes.
  - Convert the file to frequency domain using FFT and determine the frequencies of the tunes mixed with the original audio file.
- Use the fdatool on matlab to generate a digital filter
  - Determine the type of the filter you need to extract the original audio
  - Plot the response of the filter using fdatool
  - Export the filter to the work space.
- Use the exported filter to extract the original audio from the noisy file.
  - Plot the FFT of the noisy file after filtering
  - Play the audio file to insure that noise has been removed

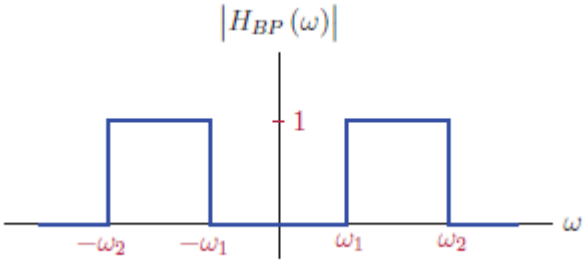
Types of Filters



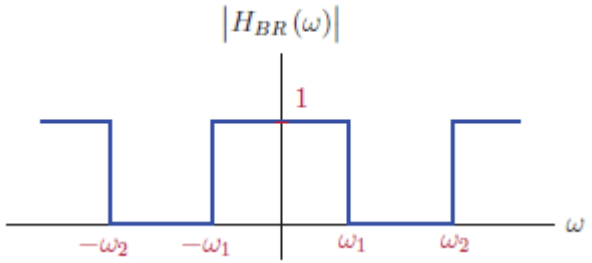
Low pass filter



High pass filter

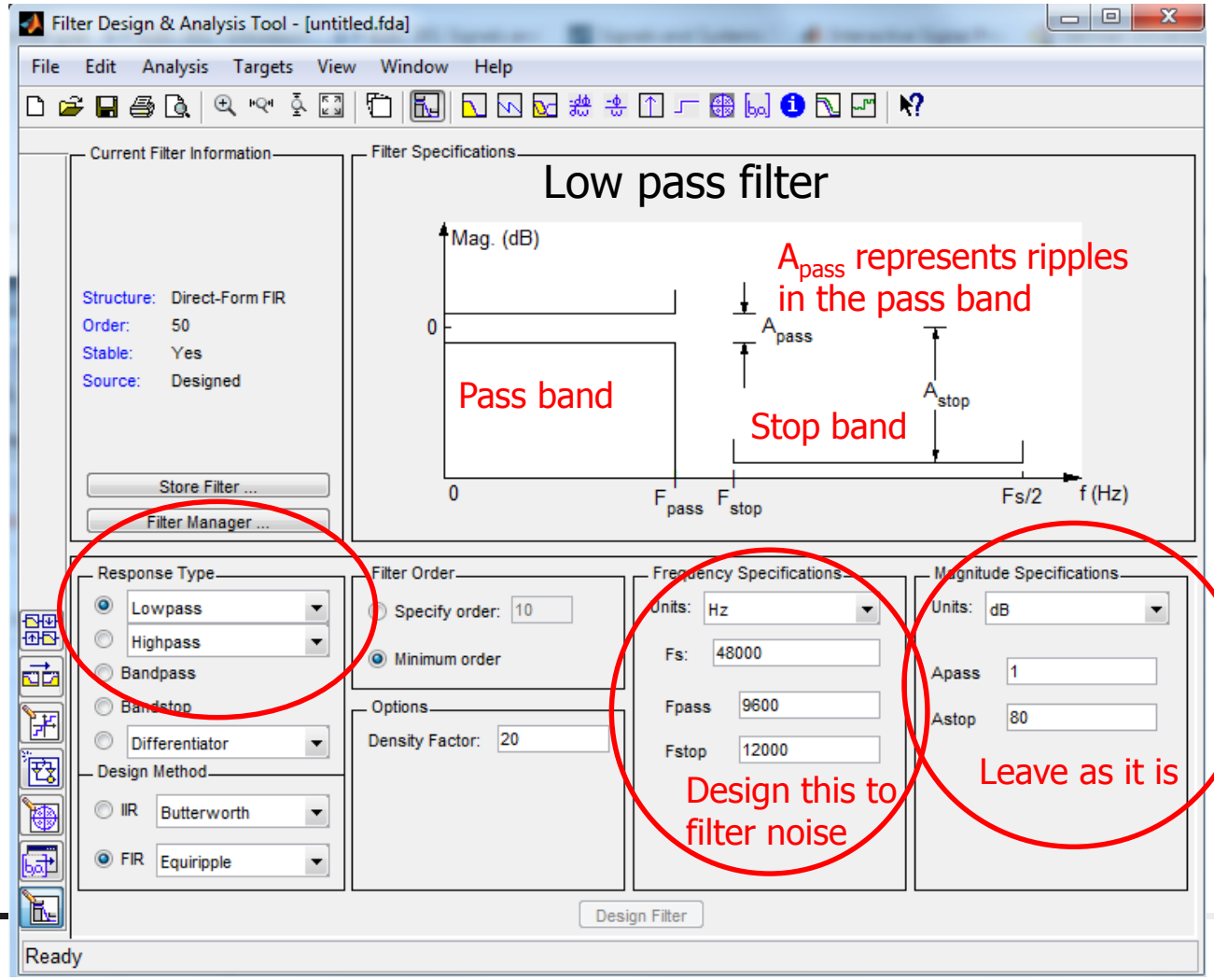


Band pass filter



Band stop filter

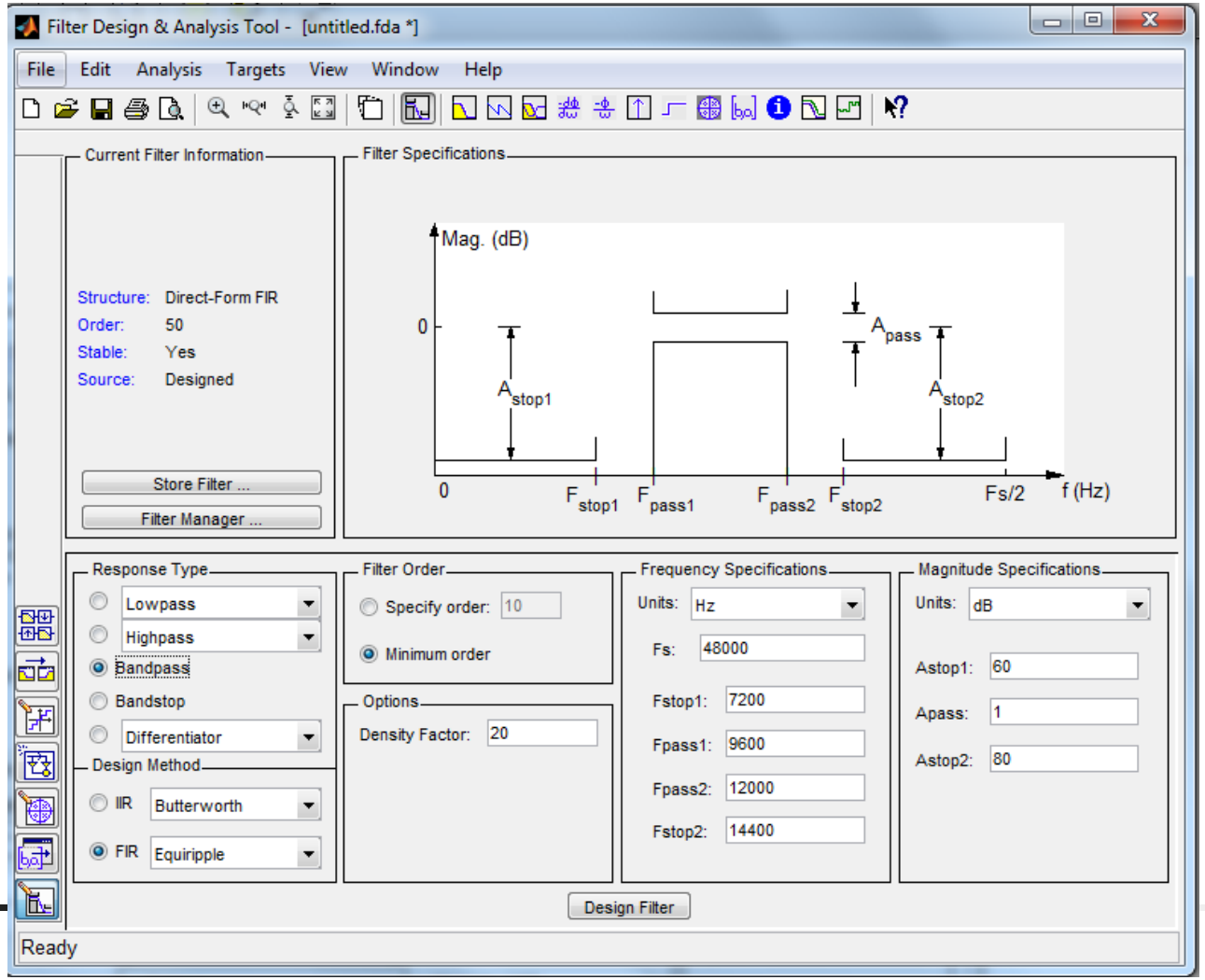
Filter Design and Analysis Tool (FDATOOL)



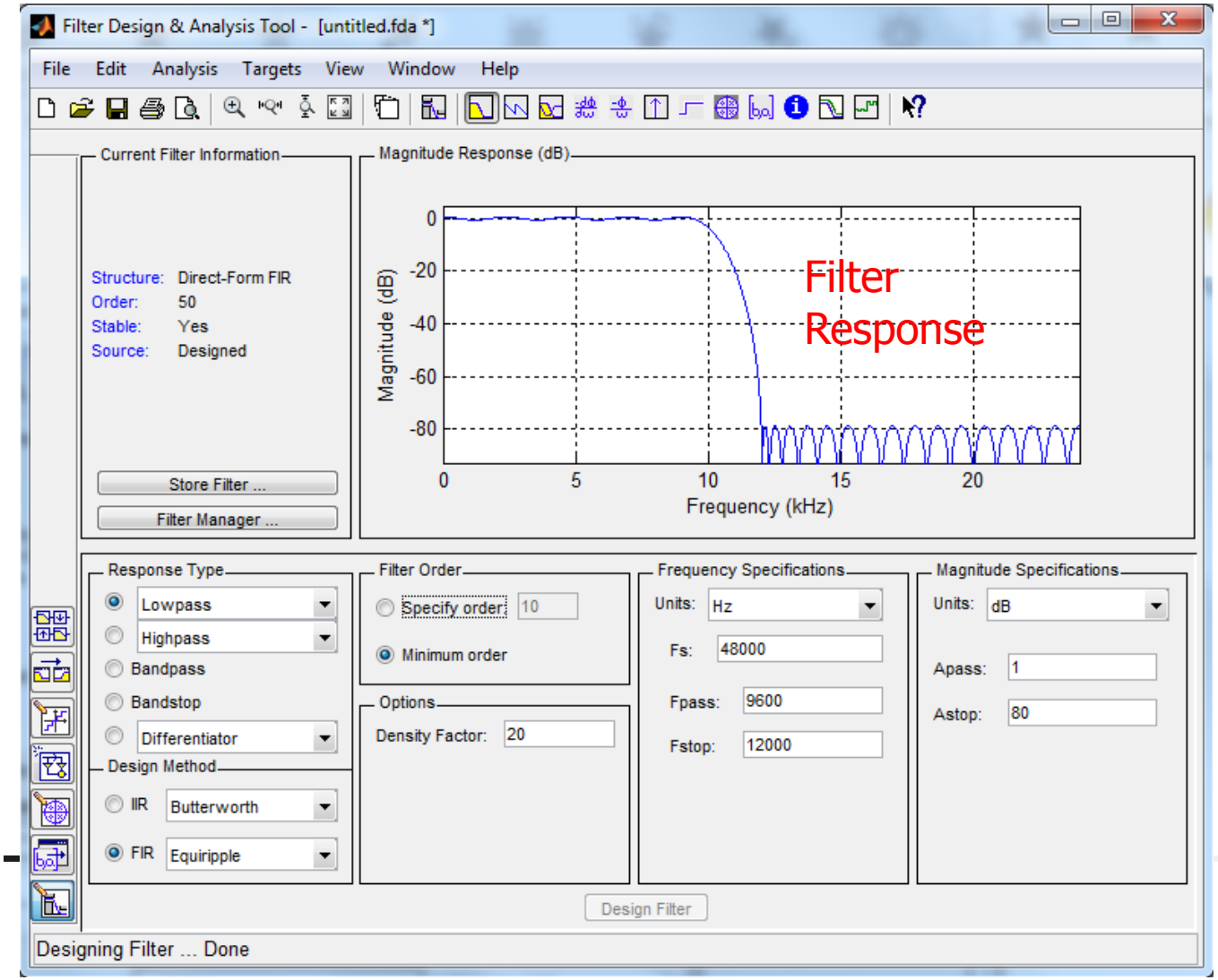
Type of filter

Note: the low pass filter may not be suitable

Band Pass Filter



Low Pass Filter Response



Project Grades and delivery procedure

- Part 1: 5%
- Part 2: 3%
- Single student project
- Due date: Thursday 26-27/4/2016 in lab
- A printed report should be delivered personally to your lab TA containing the MATLAB code with short explanation for each line and plotted outputs of each part
- A CD containing running MATLAB m files for each part, together with the processed audio files should be delivered.

- Copied reports or code means zero for both versions !

Thank You

- Questions ?