Mechtron 3TB4: Embedded Systems Design II Tutorial Lab 3

Building a Digital Filter using MATLAB

Note: Connect to the VPN prior to starting Quartus Prime, otherwise compilation will be unavailable.

Accessing MATLAB

If you do not have a local copy of MATLAB on your personal computer, you can use the department's licenses through the Virtual Machine.

Virtual Machine Installation Instructions: https://www.cas.mcmaster.ca/support/index.php/Virtual_Desktop

Note: You need to login using your CAS password, which MAY be different from your MacID password!

Note: If this is your first time using the VM, you will need to reset your password first by accessing the link in the third bullet point on this site: https://www.cas.mcmaster.ca/account/

Building a Digital Filter using MATLAB

In this tutorial we are going to decrypt secret information using a software filter. Each group will be given a sound file that contains a secret code for that group only. A deliberate noise at some frequency is added to the file in order to disguise the information. Your task is to build a software filter using Matlab to filter out the noise so that the secret information can be revealed (audible by headphone).

The sound file for your group can be found on Avenue. The naming of the file is according to your lab section and your group number. The type of filter we are going to build is the FIR filter that was discussed in class. Matlab has a set of DSP functions that enables us to do this easily.

Please follow the instructions and fill in the blanks:

Start Matlab. The following commands perform the filtering; before you execute them, read the help files of these functions by using "help FunctionName" to understand their use.

```
% Read the .wav file (replace file_name by your group's file
% Variable x stores the wave and fs stores the sampling rate

[x, fs]=audioread('ABSOLUTE_PATH_TO\file_name.wav');

% Perform FFT on the original signal to determine the frequency of the \noise"

L=length(x);
NFFT=2^nextpow2(L);
X=fft(x,NFFT)/fs;
```

```
% Show the sampling rate

fs

% We know the sampling rate is ____________________

% We need now to plot our FFT to find the source of the noise.
% Plot single-sided amplitude spectrum

f=fs/2*linspace(0,1,NFFT/2+1);
plot(f,2*abs(X(1:NFFT/2+1)));
% Reading the FFT we realize that the frequency we want to remove is _________
% (Hint: our noise is a pure sine wave)
% Now specify the frequency you want to eliminate by setting:

fkill=______________ ;

% Hint: fkill is always in the range of 0 { 1, and is normalized to frequency of fs/2.

% Determine the coefficients of the FIR filter that will remove that frequency.
% Start off the following blank with the value 4,  to  numbers larger than 160.
% Note: the following filter only works with EVEN numbers.

coeff=firgr(_____,[0,fkill-0.1, fkill, fkill+0.1, 1], [1,1,0,1,1],{'n','n','s','n','n'});

% Plot the filter
% Plot the frequency response of the designed filter to verify that it satisfies the
% Requirements

freqz(coeff,1);

% You should try different filter lengths in the firgr command and find out which one is
% the best. Filter length of 4 is terrible. Ideally, your filter should only filter out the
% noise while passing all other signals. Try increasing your filter length until you can
% achieve an adequate result.
% Be sure to plot (with freqz()) each time you create a new filter.  If you pick a filter
% length too large, the filter will \blow up". If you are unsure whether your filter has
% blown up or not, seek help from a TA.

coeff*32768

% Save these coefficients in a text file,  You will need them when coding the FIR filter.

fid=fopen('ABSOLUTE_PATH_TO\Your_Text_File_Name','w');

% If you make a typing error with the following for-end block, you need to start from the
\for" line again.
```

```
for i=1:length(coeff)
fprintf(fid,'coeff[%3.0f]=%10.0f;\n',i-1,32768*coeff(i));
end

fclose(fid);

% Filter the input signal x(t) using the designed FIR filter to get y(t).
y=filtfilt(coeff,1,x);

% Perform FFT on the filtered signal to observe the absence of frequency of the \noise".

Y=fft(y,NFFT)/L;

% Play the unfiltered sound (your system must have a
% working speaker or headphone)
% Multiply by 3 to make the volume 3 times louder.

sound(3*x,fs);

% Pause 5 seconds. (this is only necessary if you run these commands as a script)
pause(5);

% Play the filtered sound

sound(3*y,fs);

% The secret code for your group is _____________________

% Create two plots to compare

subplot(2,1,1);

% The first plot shows the FFT of the original signal.

plot(f,2*abs(X(1:NFFT/2+1)));
xlabel('frequency (Hz)');
ylabel('|X(f)|');

% The second plot shows the FFT of the filtered signal.

subplot(2,1,2);
plot(f, 2*abs(Y(1:NFFT/2+1)));
xlabel('frequency(Hz)');

% Write the filtered audio file to disk.
```

```
audiowrite('ABSOLUTE_PATH_TO\Your_Filtered.wav',y,fs);
```