**Lab 7**

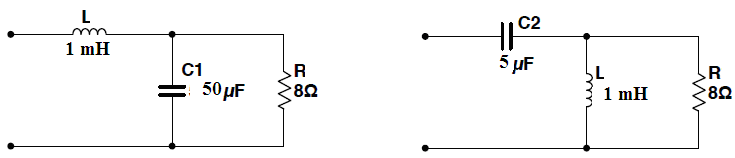
**Audio Crossover Networks**

**Purpose:**

In this lab, we will learn about the physical application of transfer functions and how they can be used to mathematically represent electrical systems such as the sub-woofer and tweeter networks.

**Background:**

In audio applications, engineers are usually concerned about the efficient transformation of energy from an electrical signal (***V***, ***I***) to a sonic signal (differing air pressures). Speakers are commonly designed for a limited dynamic range of sound. Woofers usually cover 35 Hz – 3.2 kHz, Tweeters commonly cover 2 kHz –30 kHz, and the Mid speakers overlap both and commonly address frequencies between 200 Hz – 4 kHz. Rather than sending the entire signal to different speakers, cross over networks are used to increase and/or decrease the effective impedance of each speaker, thereby increasing and/or decreasing the power sent to that speaker. Note that the cut-off frequency for each of the networks shown below is, . The LC configuration creates a low pass filter and CL configuration creates a high pass filter.



**Fig. 1: Sub-woofer Network Fig. 2: Tweeter Network**

***First make a note of the actual values for the inductor and the capacitor available in the lab before proceeding. Note that* R=8 Ω *is the speaker.***

**Procedure:**

**Step 1 (Transfer Functions)**: Assuming that each speaker can be represented as an 8 Ω resistor, derive the Transfer Functions for Figs. 1 and 2 by the ***voltage divider*** method, referring them as *FL* (filter low) and *FH* (filter high), respectively.

**A:** **Sub-woofer Network**





Where,,, and .

***We will use the following version for the Matlab simulation*,**



**B:** **Tweeter Network**





Where,,,, and.

***We will use the following version for the Matlab simulation*,**

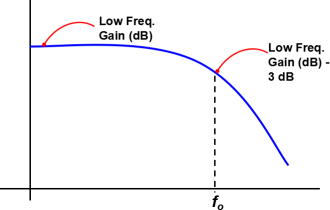
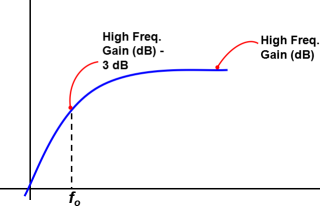


**Step 2 (Time-domain):** Convert the two transfer functions to the time domain, i.e. calculate the ***Impulse Response*** for the circuits in Figs. 1 and 2.

** and **

**Step 3 (Circuit Simulations):** It is helpful to understand how this circuit will change with respect to frequency variations. Rather than conducting repeated transient analysis and recording the variations in amplitude and phase (Phasor Domain calculations) for various frequencies, circuit simulators can perform an AC Sweep (sweep the drive frequency). Using ***Multisim***, layout the circuits shown in Figures 1 and 2, attach an AC source, and perform an AC decade sweep from 35 Hz to 35 kHz selecting **Simulate>>Analysis>>AC Sweep** or by using the **BODE Plotter**. Choose the amplitude of input voltage, *Vin* = 1 V (AC amp=1 V) so that when you plot output voltage *Vout*, you will have the transfer function itself (*Vout*/*Vin*). In your lab report, please include a single plot showing the transfer function (*Vout*/*Vin*) on a vertical axis. The horizontal axis is the frequency axis. This type of plot is commonly referred to as a ***BODE Plot***.

**Step 4 (Cut-off Frequency):** Power delivered to the load is proportional the *V*2 and/or *I*2. The point where the magnitude of the load voltage reaches  is commonly referred to as the “half power point” or “cutoff frequency.” From your circuit simulations, identify the half-power frequency of each network. If you have a plot with dB units, locate the -3dB point (e.g. -3dB down from the low frequency gain for low pass filter and -3dB down from high frequency gain for high pass filter) since 20\*log10(0.707) = -3dB. Note the frequency at this point. This is the cut-off frequency of your network. Please refer to Fig. 2.

**(a) (b)**

**Fig. 2: Location of cut-off frequency for (a) low pass filter and (b) high pass filter**

**Step 5** (**Numeric Simulations**): Using the **tf** function feature in MATLAB, load the Transfer Functions found earlier into MATLAB and save them as FL (Filter Low) and FH (Filter High). ***Please see the appendix.***

a) Using MATLAB’s bode function, plot the frequency response of the two circuit networks and include the graphs in your lab book. Compare the graphs to those obtained by the circuit simulator and explain any discrepancies.

b) Using MATLAB, load an audio file, filter it, and play the results.

**Step 6** (**Hardware**): Construct the circuits shown in Figures 1 and 2. Use a generic (MID range) 8Ω speaker as the 8Ω resister load in each circuit. Apply the 1V AC function generator as the input and vary the input frequency over the range from 35 Hz to 35 kHz. Document at what frequency each speaker is no longer producing an audible sound.

**Appendix**

% Lab 6 EE2260

%=====================================

% Transfer functions: FL and FH

% Impulse responses: fl and fh

%=====================================

clc;%reset the workspace command line

clear all; %clear all the variables

close all; %close all the plots

display('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*');

display('Lab7: Audio Crossover Networks');

display('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*');

L=1\*10^(-3); % Inductance; use the available value

C1=47\*10^(-6); % Capacitance, use the available value

C2=5\*10^(-6); % Capacitance; use the available value

R=8; % Resistance; speaker

FL=tf( [1/C1/L], [1 1/R/C1 1/C1/L]); %Define transfer function FL

FH=tf( [1 0 0], [1 1/R/C2 1/L/C2]); %Define transfer function FH

figure(1)

subplot(2,2,1),bode(FL),title('FL')

subplot(2,2,2),bode(FH),title('FH')

subplot(2,2,3),impulse(FL),title('fl')

subplot(2,2,4),impulse(FH),title('fh')

a1=1/2/C1/R;

w1=sqrt(1/C1/L-(1/2/C1/R)^2);

K=1/(C1\*L)/w1;

a2=1/2/C2/R;

w2=sqrt(1/C2/L-(1/2/C2/R)^2);

K1=1/(R\*C2);

K2=(1/(C2\*L)-0.5/(R\*C2)^2)/w2;

t=0:1e-6:0.004;

fl = K\*exp(-a1\*t).\*sin(w1\*t);

fh = -K1\*exp(-a2\*t).\*cos(w2\*t)-K2\*exp(-a2\*t).\*sin(w2\*t);

fh(1)=fh(1)+1;

figure(2)

subplot(2,2,1),plot(t,fl),xlabel('Time [s]'), ylabel('Amplitude'),title('fl')

subplot(2,2,2),plot(t,fh),axis([0 5e-4 min(fh) 1e4]), xlabel('Time [s]'),ylabel('Amplitude'),title('fh')

display('Press any key to play "Hallelujah" by Handel: ')

pause

%==================================================

% Processing the "Hallelujah" song

%==================================================

load handel % Load the "Hallelujah Chorus"

display('Playing the song...')

soundsc(y,Fs) % Play the sound, Fs=sampling frequency, y=audio data

pause(10)

% t=[]; % Reset t

t = [0:250]/Fs; %create a time vector

fl = K \* exp(-a1\*t).\*sin(w1\*t); %Impulse Response of Lowpass filter

fh = -K1\*exp(-a2\*t).\*cos(w2\*t)-K2\*exp(-a2\*t).\*sin(w2\*t); %Impulse Response of Highpass filter

fh(1)=fh(1)+Fs; % This approximates the delta function %This may not be necessary

yfl = conv(y,fl); %Lowpass filter output

display('Playing the Lowpass filter (subwoofer only) output...')

soundsc(yfl,Fs) % Play the lowpass version

pause(10)

yfh = conv(y,fh); %Highpass filter output

display('Playing the Highpass filter (Tweeter only) output...')

soundsc (yfh,Fs) % Play the highpass version

figure(2)

subplot(2,2,3),plot(t(1:100),yfl(1:100)),xlabel('Time[s]'), ylabel('Amplitude'),title('Lowpass filtered song')

subplot(2,2,4),plot(t(1:100),yfh(1:100)),xlabel('Time [s]'), ylabel('Amplitude'),title('Highpass filtered song')