

PRACTICAL FILTER DESIGN & IMPLEMENTATION LAB

BEFORE YOU BEGIN

PREREQUISITE LABS

- ▶ Introduction to Oscilloscope
- ▶ Introduction to Arbitrary/Function Generator

EXPECTED KNOWLEDGE

- ▶ Understanding of LTI systems.
- ▶ Laplace transform for circuit analysis
- ▶ Transfer Functions analysis and synthesis
- ▶ Practical filtering design with MATLAB

EQUIPMENT

- ▶ Workstation with MATLAB 6.
- ▶ Digital Multimeter
- ▶ Programmable Power Supply
- ▶ AFG3000 Series Arbitrary/Function Generator
- ▶ TDS3034B Digital Phosphor Oscilloscope

MATERIALS

- ▶ Complete set of resistors in ECE Toolkit
- ▶ Complete set of capacitors in ECE Toolkit
- ▶ Set of Operational Amplifiers
- ▶ Solderless Breadboard
- ▶ Oscilloscope Probes
- ▶ 3 ½ floppy
- ▶ Speaker

OBJECTIVES

After completing this lab, you should be able to design and build a circuit that implements a practical filter given a set of filter specifications.

INTRODUCTION

Filters can be defined by a set of five parameters: passband gain (A) measured in dB, passband ripple (A_p) measured in dB, pass-band edge frequencies (ω_p) measured in rad/sec, stopband edge frequencies (ω_s) measured in rad/sec, and stop-band attenuation (A_s) measured in dB. You should know how to use MATLAB to find a transfer function of several types of filters that meet these specifications. Specifically, you should be familiar with Butterworth, Chebyshev, and Elliptical filters. In this lab, you will design and build a Chebyshev type 2 audio filter.

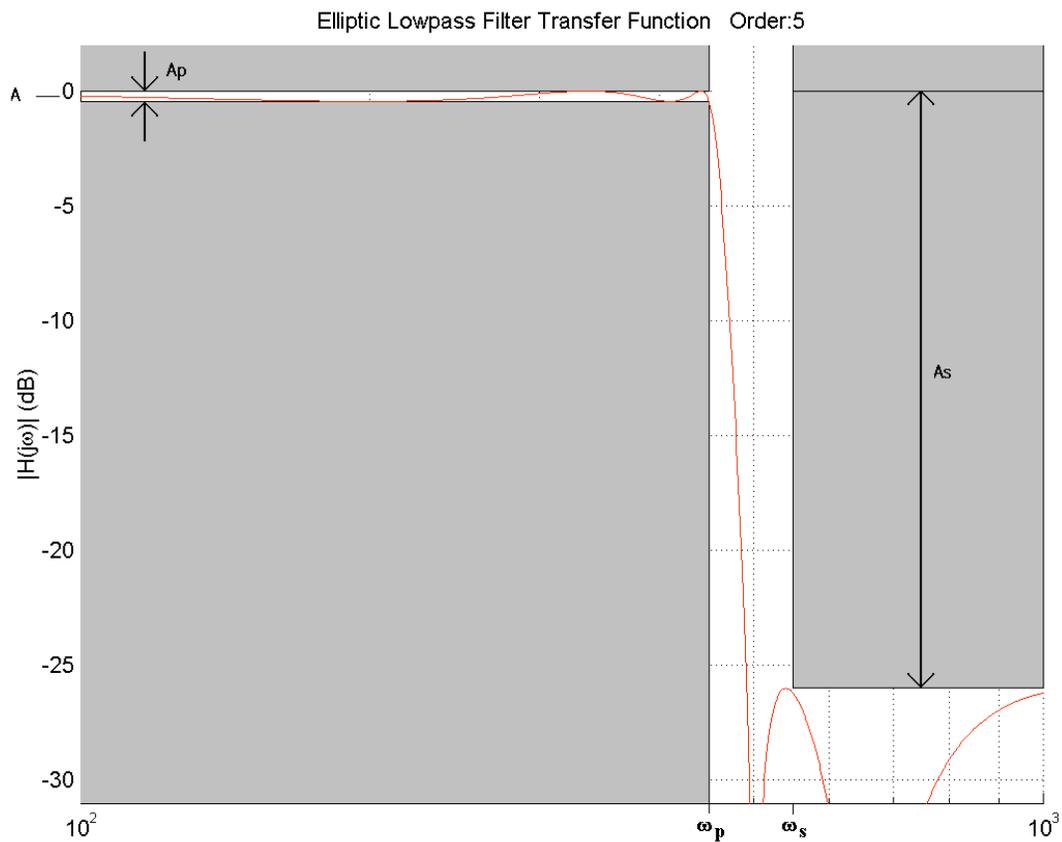


Figure 1. Graphical example of lowpass filter specifications.

PRELAB

THE CHEB2ORD COMMAND

The `cheb2ord` function can be used to determine the order and natural frequency for a Chebyshev type 2 filter. The syntax for `cheb2ord` is:

$$[N, Wn] = \text{cheb2ord}(Wp, Ws, Rp, Rs, 's')$$

where

Wp is the passband frequency in rad/sec.

Ws is the stopband frequency in rad/sec.

Rp is the passband ripple in dB.

Rs is the stopband attenuation in dB.

's' indicates the filter is continuous time, not discrete-time

THE CHEBY2 COMMAND

The **cheby2** function returns the polynomial coefficients for the numerator and denominator of the transfer function for a Chebyshev type 2 filter, based on the order and natural frequency supplied from the **cheb2ord** function. The syntax for **cheby2** is:

$$[B, A] = \text{cheby2}(N, Rs, Wn, \text{'type'}, \text{'s'})$$

where

N is the order supplied from cheb2ord.

Rs is the stopband ripple in dB.

Wn is the natural frequency in rad/sec.

'type' is the type of filter. e.g. 'high', 'low'

's' indicates radian mode.

Type "help cheby2" at a MATLAB prompt for more specific information on how to use this function.

Given the following filter parameters, you will use MATLAB to determine the transfer function, and pole and zero locations for a Chebyshev type 2 filter.

Passband Gain: 1 dB in passband

Fp: 1 kHz

Fs: 700 Hz

Passband ripple: ≤ 1 dB

Stopband attenuation: ≥ 40 dB

Answer question 1.

To find the order and cutoff frequency of our filter, we will use the **cheb2ord** function in MATLAB. Use this information and the **cheby2** function to determine the polynomial coefficients for the transfer function of the filter. With these polynomial coefficients, the transfer function can be created using the **tf** function.

Answer questions 2 – 5.

Before you can synthesize this transfer function, you must determine the zeros and poles. Remember that zeros are defined as the roots of the numerator of the transfer function and poles are defined as the roots of the denominator of the transfer function. These can be determined using the "roots" command in MATLAB. Solve for the poles and zeros of your transfer function. Store them to variables 'z' and 'p' respectively, so that they will be easier to use later.

Answer question 6.

Notice that the zeros and poles that are returned are complex values. You may also notice that they are in complex conjugate pairs. For synthesis, you will need to combine the pairs of complex zeros and poles into second-order polynomials with real coefficients. As a result, each complex-conjugate pair of zeros and poles will be implemented using one second-order stage of

our filter. The real zero and pole will be implemented using a first-order stage. These stages are described in detail below.

LAB

Cascading Transfer Functions

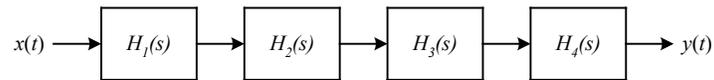


Figure 2. Example of cascaded transfer functions with active outputs.

Practical analog filters are usually implemented as a cascade of first or second-order filters. Each of these simpler filters is called a *stage*. In order to cascade each stage, the output of the first stage becomes the input of the next stage, and so on. This is equivalent to multiplying the individual transfer functions for each stage with that of the next. **Figure 2** gives a flow diagram of cascading each stage.

Each of the stages below has a negative leading coefficient due to the active outputs connected to an inverting amplifier. If there are an even number of stages, these negative coefficients will cancel and should not affect the transfer function of the overall cascaded circuit.

First-Order Stage

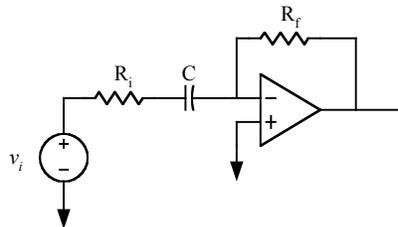


Figure 3. First order highpass active filter.

The first-order highpass filter shown in Figure 3 should be used to implement the stage that contains the real pole and the real zero. Note that the zero is at 0 rad/s. The transfer function of this circuit is given below.

$$H(s) = \frac{-k \cdot s}{(s - p)}$$

You should be able to show that the transfer function parameters k and p have the following relationship to the circuit parameters.

$$p = -\frac{1}{R_i C} \quad k = -\frac{R_f}{R_i}$$

Second-Order Stage

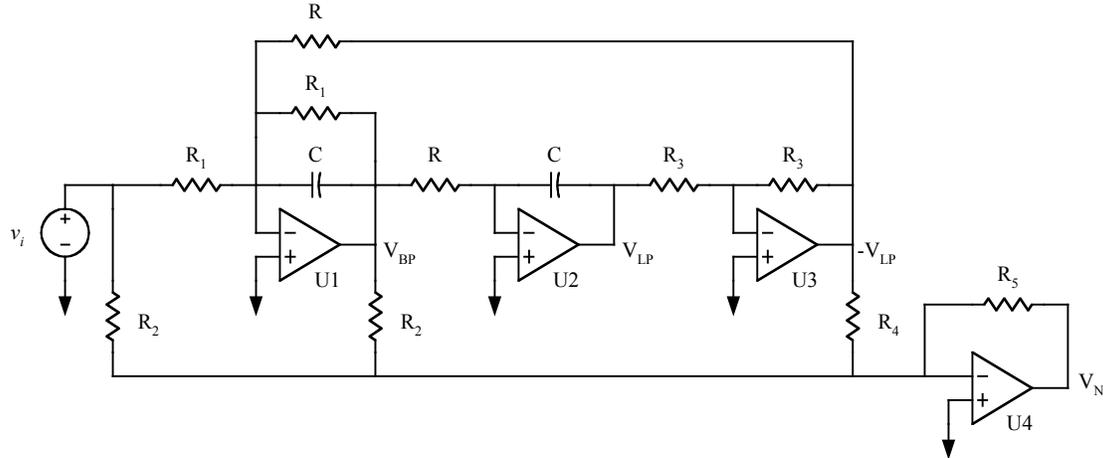


Figure 4. Second-order notch filter with two imaginary zeros and two complex poles.

The circuit in Figure 4 represents the basic circuit that will be used in the other stages of your filter (one stage for each pair of complex poles and pair of imaginary zeros). It consists of four parts. A bandpass filter (U1), an integrator (U2), an inverter (U3), and a summing amplifier (U4). The bandpass filter, integrator, and inverter each have their own output. The output of the circuit is labeled V_N . The transfer function is given below.

$$H(s) = \frac{-k(s-z)(s-z^*)}{(s-p)(s-p^*)} = -k \frac{s^2 - (z+z^*)s + zz^*}{s^2 - (p+p^*)s + pp^*} = -k \frac{s^2 + b_1s + b_0}{s^2 + a_1s + a_0}$$

Since you want the passband gain to be 1 (0 dB) for all of the filter stages, the gain denoted by k should be 1. Note that since the zeros are purely imaginary, the coefficient $b_1=0$. The relationship of the transfer function coefficients to the circuit parameters is described by the following equations.

$$H(s) = -k \frac{s^2 + \omega_z^2}{s^2 + \frac{\omega_n}{Q}s + \omega_n^2} \quad (1.1)$$

$$\omega_n = \frac{1}{RC} \quad Q = \frac{R_1}{R} \quad \omega_z = \omega_n \sqrt{1 - \frac{R_2}{R_4Q}} \quad k = \frac{R_5}{R_2} \quad (1.2)$$

To choose the circuit values, you will need to choose the complex conjugate pairs of zeros and poles for each second-order stage and determine the value of Q for each stage. You should be able to easily show that these relationships are given by the equations below.

$$\omega_n = |p| \quad \omega_z = |z| \quad Q = -\frac{\omega_n}{(p+p^*)}$$

These equations can be implemented using the following is the MATLAB code.

Zeros

```
>> abs(z)
```

Poles

```
>> abs(p)
```

Q's

```
>> -abs(p)./(2*real(p))
```

Answer question 7.

You may choose which pairs of imaginary complex-conjugate pairs of zeros are grouped with each pair of complex-conjugate pair of poles. Note, however, that the real pole and zero should be implemented by the first-order stage and the zeros and poles have units of rad/sec.

Individual Stage Analysis

Once you have decided which complex-conjugate pairs of zeros will be grouped with each complex-conjugate pair of poles, determine the transfer function for each stage. Simulate each transfer function in MATLAB and generate the bode plot, impulse response, and step response for each transfer function.

Answer questions 8 – 11.

With values for the zero, pole, and Q for each stage, you should be able to pick the circuit component values for each stage of the filter. When building the circuit, try to get as close as possible to your calculated component values, but you do not have to be exact. Although this will introduce some error in the expected response and the filter may not satisfy the design specifications, the actual analog filter response should be close to the ideal response modeled in MATLAB.

Answer question 12 – 15.

Circuit Implementation and Validation

Build the circuit with the component values that you calculated. Measure the output of the circuit with a sinusoidal input that has a 1 V amplitude. Record the amplitude of the output of each stage for many frequencies between 100 Hz and 10 kHz. Record many frequencies near the transition zone of the filter. Once you have verified that each stage is working correctly, repeat the process for the entire cascaded filter.

Answer questions 16 – 19.

Drive the circuit with an impulse and a then a step from the Arbitrary/Function Generator (AFG). How does this compare with the output of MATLAB?

Answer questions 20 – 21.

Hook up a speaker to the output of the circuit and apply a sinusoidal signal from the AFG that sweeps from 100 Hz and 10 kHz repeatedly. This signal is called a chirp.

Answer questions 22 – 23.