EE 451 – LAB 6

IIR Filter Design

In this laboratory you will design IIR filters with different. The FIR discussed in the previous laboratory does not have an analog counterpart. Design of IIR filters, on the other hand, usually makes use of the vast knowledge already available on analog filters. The design procedure involves converting an analog filter to a discrete filter using a transformation.

Consider the general input-output equation

$$y(n) = \sum_{k=0}^{2} b_k x(n-k) - \sum_{l=1}^{2} a_l y(n-l)$$

or equivalently,

$$y(n) = b_0 x(n) + b_1 x(n-1) + b_2 x(n-2) - a_1 y(n-1) - a_2 y(n-2)$$

This recursive equation represents an IIR. The z-domain representation of the input-output equation described above can be implemented using different structures: namely Direct Forms I & II. The Direct Form I structure requires 2N delay elements for an Nth order filter. Direct Form II is one of the most commonly used structures as it requires half as many delays. A Direct Form II implementation would require the use of an intermediate variable w(n),

$$w(n) = x(n) - a_1 w(n-1) - a_2 w(n-2) \text{ and } y(n) = b_0 w(n) + b_1 w(n-1) + b_2 w(n-2)$$

Taking the z-transform we find,

$$X(z) = (1 + a_1 z^{-1} + a_2 z^{-2}) W(z) \text{ and } Y(z) = (b_0 + b_1 z^{-1} + b_2 z^{-2}) W(z)$$

Thus

$$H(z) = \frac{Y(z)}{X(z)} = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

It is possible to implement higher order systems, i.e. fourth order IIR structures, as a cascade of Direct Form II second sections. A fourth order transfer function can be expressed as,

$$H(z) = \frac{Y(z)}{X(z)} = \frac{(b_{01} + b_{11} z^{-1} + b_{21} z^{-2})(b_{02} + b_{12} z^{-1} + b_{22} z^{-2})}{(1 + a_{11} z^{-1} + a_{21} z^{-2})(1 + a_{12} z^{-1} + a_{22} z^{-2})}$$

Simulation of the IIR in MATLAB

1. Use MATLAB to design an elliptic filter to meet the following specifications:
   f_{pass}: 4 kHz
   A_{pass}: 0.1 dB
   f_{stop}: 4.5 kHz
   A_{stop}: 50 dB

2. What order of filter is this? Plot the magnitude of the filter you have designed.

3. Implement the filter as a cascade of second-order filter sections. Store the filter coefficients in a text file. This file will be a header file which you will include in your program and compile in the main program.

```
//=========== elliptic.cof =========
// This file is used in the IIR lab
// Created by Hector Erives 8/2008

 #define Sections 4

float b[Sections][3]={
{ 1.0000000000, 0.1172762163, 1.0000000000},
            . . .
{ 1.0000000000,-1.6703650090, 1.0000000000}};
float a[Sections][3]={
{ 1.0000000000,-1.5341039500, 0.6090265482},
            . . .
{ 1.0000000000,-1.7035048681, 0.9772932709}};
```

4. Use MATLAB to find out the order of a Butterworth filter to meet the same specifications of Part 1.

5. What order of filter is this? Plot the magnitude of the filter you have designed.

Implementation of the IIR filter on the C6713

1. Write a program that implements the elliptic filter on the C6713 that meets the specifications given above Part 1. Output the input signal to one channel and the filtered signal through the other channel.

2. Start CCS and begin a new project. Create and add a configuration file to the project. Select File → New → DSP/BIOS Configuration. We will use a HWI to (i) read a new input sample from the codec, (ii) calculate the filter output, and (iv) output to the codec.

3. Connect a function generator to the board and vary the frequency. Record the magnitude response of the filter.

4. Implement a Butterworth filter on the C6713 that meets the specifications given above in Part 1. Connect a function generator to the board, vary the frequency, and record the magnitude response of the filter.