EE 311
Notes on Cascade Second Order Sections

Design a filter using MATLAB and use the tf2sos function to translate it to second order sections. If you print sos you will get an array that looks like the following. This one is for an 8th order elliptic filter.

```
//Second order sections
//0.053443  0.082450   0.053443  1.000000  -0.663623  0.203018
//1.000000  0.271035   1.000000  1.000000  -0.508603  0.598833
//1.000000  -0.169783  1.000000  1.000000  -0.407663  0.862756
//1.000000  -0.285008  1.000000  1.000000  -0.376070  0.969274
```

Each second order section is on one line with the numerator being the three coefficients on the left and the denominator being the three coefficients on the right. For example

```
//0.053443  0.082450   0.053443  1.000000  -0.663623  0.203018
```

is a second order section written in equation form as:

$$\frac{y}{x} = \frac{0.053443z^2 + 0.082540z + 0.053443}{z^2 - 0.663623z + 0.203018}$$

The corresponding difference equation is:

$$y_n = 0.053443x_n + 0.082540x_{n-1} + 0.05344x_{n-2} + 0.663623y_{n-1} - 0.203028y_{n-1}$$

To write this in C-code we use b_0, b_1, and b_2 for the numerator coefficients and a_1, a_2 for the denominator coefficients. Since there are multiple sections we create two subscripts where the first subscript is the section number. The definition of coefficients for the first two sections are:

```
//Section 1
//0.053443  0.082450  0.053443  1.000000  -0.663623  0.203018
const float b10 = 0.053443;
const float b11 = 0.082450;
const float b12 = 0.053443;
const float a11 = -0.663623;
const float a12 = 0.203018;

//Section 2
//1.000000  0.271035  1.000000  1.000000  -0.508603  0.598833
const float b20 = 1.000000;
const float b21 = 0.271035;
const float b22 = 1.000000;
const float a21 = -0.508603;
const float a22 = 0.598833;
```

Sections 3 and 4 are similar. Each section can then be implemented as usual where I have created a temporary variable, *w* to do the numerator terms and the usual variable y to do the output terms. We need to define all of the following variables:

```
int main(void)
   {unsigned int xInt;
    float x, y, y10, y20, y30, y40;
    float w10, w11, w12;
    float w20, w21, w22;
    float w30, w31, w32;
    float w40, w41, w42;
```

In this notation the first subscript is the section number and the second is the delay amount. So that w12 is w1(n-2) etc.

For the sections the implementation equation looks like this: (x is the normalize input variable from the A/D converter.)

```
        //first section
        w10 = x - a11*w11 - a12*w12;
        y10 = b10*w10 + b11*w11 + b12*w12;
        //second section with 'y10' as the input, and 'y20' as the output
        w20 = y10 - a21*w21 - a22*w22;
        y20 = b20*w20 + b21*w21 + b22*w22;
        //third section with 'y20' as the input, and 'y30' as the output
        w30 = y20 - a31*w31 - a32*w32;
        y30 = b30*w30 + b31*w31 + b32*w32;
        //fourth section with 'y30' as the input, and 'y40' as the output
        w40 = y30 - a41*w41 - a42*w42;
        y40 = b40*w40 + b41*w41 + b42*w42;
```

The variable y40 is the final output and can be sent to the D/A converter. After doing the output to the D/A all of the variable need to be shifted:

```
     //Shift values for each section
        w12 = w11;
        w11 = w10;
        w22 = w21;
        w21 = w20;
        w32 = w31;
        w31 = w30;
        w42 = w41;
        w41 = w40;
```