**LAB 8 (OMAP-L138)**

**Infinite Impulse Response System (*IIR*)**

**objectives**

* Design an IIR notch (stopband) filter
* Program the OMAP-L138 LCDK processor to implement IIR notch filter.

**Introduction**

An infinite impulse response system is a system where the output depends on past outputs in addition to present and past inputs. The difference equation of an IIR filter is of this form

*y*[*n*] + b1*y*[*n*-1] +…+ bN*y*[*n*-*N*] = a0*x*[*n*] + a1*x*[*n*-1] +…+ aM*x*[*n*-*M*]. 🡪 1

This system is of *N*th order. The impulse response *h*[*n*] of an IIR filter usually consists of a large number (infinite) of samples but with time the samples amplitude decay to zero. The impulse response *h*[*n*] of *IIR* system can be found from the difference equation by taking the z-transform as follow

*Y*(z) + b1*z*-1*Y*(z) + b2*z*-2*Y*(z) +…+ bN *z-NY*(z) = a0 *X*(z) + a1 *z*-1*X*(z) + a2 *z*-2 *X*(z) +…+ aM *z*-M*X*(z).

The impulse response *h*[*n*] can be found by taking the inverse z-transform of the transfer function *H*(z) in the above equation. Depending on the system the impulse response might be of this form

*h*[*n*] = A*u*[*n*] + *n*γ*n**u*[*n*] + γ1*n**u*[*n*]+ ….

The output *y*[*n*] is calculated by the linear convolution of the input *x*[*n*] with the impulse response *h*[*n*], as follow

In this lab we will program the DSP LCDK board to be a second order notch IIR filter and the output will be calculated as the samples of the input signal arrive to the DSP board. The output for the second order filter will be of this form.

*y*[*n*] = -*b*1*y*[*n*-1] – *b*2 *y*[*n*-2] + *a*0 *x*[*n*] + *a*1*x*[*n*-1] + *a*2 *x*[*n*-2]

The constants *b*1, *b*2, *a*0, *a*1, and *a*2 will depend on the type of filter and system. To calculate the output we need to create a memory to hold the values of two past inputs *x*[*n*-1] and *x*[*n*-2] and the two past outputs *y*[*n*-1] and *x*[*n*-2].

**Prelab Exercises**

For the second order IIR system described by the difference equation

1. Find the transfer function *H*(z)
2. Plot the poles and zeros of the system in the z-plane for *r* = 0.81.
3. What type of filter is this system?
4. Plot the magnitude of the frequency response *H*(Ω) based on the locations of the poles and zeros.
5. Find the impulse response *h*[*n*]

**lab Exercises**

Download the audio file “SentenceToneFinal” from the course website to Matlab workspace. Listen to the audio file and write down what you heard. Let us call the downloaded audio *x*[*n*].

Write down the sentence: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. What is the sampling rate of the audio file?
2. Use the spectrogram function in Matlab to find the frequency of the tone noise. What is this frequency?
3. Program the LCDK DSP board to filter out the tone noise from the signal *x*[*n*] using the notch filter described by the transfer function *H*(*z*) and the difference equation to generate *y*[*n*].

🡪 2

Where *r* can have a value between 0.49 and 0.81. Try different values to get the best filtering without deteriorating the original signal.

1. Play the downloaded audio signal *x*[*n*] in Matlab to the input of the LCDK DSP board and listen to the signal from the output of the LCDK DSP board through the computer speaker. What is coming from the LCDK board should be *y*[*n*]. Can you hear a difference in audio quality? Experiment with different values of *r*.

Write down the sentence: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Call your instructor to verify your code: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Hint:** To calculate the output based on Eq. 2, you will need to declare two buffers (arrays) of size three to store current and two previous inputs and current and two previous outputs. You need to initialize the array to have zero values in the cells of the array. Define the input array as xRight [ ] and the output array as yRight [ ]. To relate the variables in the C code to the mathematical symbols of equation 2 remember the following:

The first cell xRight[0] stores the current input x[n], the second cell xRight [1] stores the input one sample earlier x[n-1], the third cell xRight [2] stores the input two samples earlier x[n-2]. Similarly, yRight[0] is the current output y[n], yRight[1] is the output one sampling interval earlier and *y*Right[2] is the output two sampling interval earlier.

Now, every time an input comes to the DSP board, the Interrupt Service Routine (ISRs.) will be called and the ISRs function should carry out the following two steps before the next input arrives.

**Step 1**: Calculate the current output using the difference equation as follow

yRight[0] = – *r\*yRight*[2] + *xRight*[0] + *xRight*[2];

**Step 2**: Shift *x*[0] to be stored in *x*[1]. The old *x*[1] will be lost but that is okay because we already used it in step 1 and we will not need it to calculate future outputs. Assign the current input to *x*[0].

**Step 3:** Shift *y*[0] to be stored in *y*[1]. Assign the current output to *y*[0].

**Lab Report**

Your lab report will consist of the prelab and the CCS code for the FIR filter and your explanation of the code and questions raised in the lab manual.