FIR Filter Implementations

Purpose

The purpose of this experiment is to implement FIR filters in real time.

Background

In this experiment, an FIR filter will be implemented in real time. There are various things that make this experiment challenging. First, the program for filtering must be implemented in C. The C code for implementing an FIR filter is simple, but may cause problems because this program is meant to work in real time. In previous classes, convolutions have been done by hand or in MATLAB. This is the first time a convolution will be done in real time.

In the real time filter, the convolution is implemented by using the difference equation directly. Using $x[n]$ as the input and $y[n]$ as the output, the difference equation for an FIR filter is,

$$y[n] = h[0]x[n] + h[1]x[n-1] + \cdots + h[N-1]x[n-N+1].$$

This experiment is also challenging because this may be the first design implemented in hardware. The design may meet the specifications in MATLAB but fail when implemented on the DSK. The AIC used for input and output is not perfect, and this must be taken into account when designing the filter. For example, in one of the designs the output must be between 0 and -1 dB from 1000 to 3000Hz. If the AIC amplifies the signal, then it must be attenuated by the digital filter in order to stay below 0 dB.

Having to show that the design meets the specifications is the final challenge. This means using the function generator along with the oscilloscope and multimeter to show that the filter is linear phase and that it meets the magnitude specifications.

Design and implementation are generally done in steps. If the problem is very simple it may be possible to design and implement all at once, but this is generally not the case. In most cases, the design and implementation are broken into parts, and each part is verified to work before moving on to the next part. In this experiment, the problem is not difficult, but it is difficult to verify what is going on in the DSK if the output is not what you expect.

I have seen the same thing happen many times: a student writes the complete code for the DSK in C and runs it. It does not work, and the student has no idea why. Instead of implementing the solution all at once on the DSK, it is better to implement a very simple filter on the PC. This way every step of the problem will be easily verifiable, and easy to debug. A program written for this experiment will also help with other experiments and probably with your final project.

I would recommend the following procedure. Start with a program on the PC that can read and write numbers from a simple ASCII file. This would basically be the **loopc.c** program from the DSK, but for the PC. Once this program is running, use it to implement a very simple, say third order, FIR filter. Make shure that the numbers are read and written from the files simulating what is done on the DSK. In other words, instead of reading all the values at the beginning, read one value from the file, compute the output, write it to another file, read the next value, etc. This C program on the PC is very simple to debug, because you can write the values at each step to the screen. Select the filter coefficients and the input so that the output is easy to calculate by hand and see if the program is doing what you want. Once this program is running, then go ahead and implement the filter on the DSK.

Experiments

Experiment 1.1.

Design a FIR linear phase filter in Matlab for the magnitude specifications given in ***Figure 1.1***. Implement the filter using firnc.asm. The filter must meet the specifications in MATLAB, but does not need to meet the specifications when implemented on the DSK.

a. **Set** the sampling frequency to 8kHz. What is the cutoff (-3dB) frequency in Hz?
b. **Change** the sampling frequency to 10kHz. What is the new cutoff frequency? What should it be theoretically?
c. **Reset** the sampling frequency to 8kHz and remove the anti-aliasing filter. Connect the oscilloscope so you can see both the input and output. Input a frequency of 300Hz and determine the delay from input to output. Next, verify that the filter is linear phase.
d. **Answer** the following questions at the end of this statement. With a sampling frequency of 8kHz and no anti-aliasing filter, increase the input frequency from 500Hz to 7.5kHz. What happens to the signal? Explain why this happens. Is this really a lowpass filter?

Experiment 1.2

Design a minimum order FIR linear phase filter for the magnitude specifications given in ***Figure 1.2*** in Matlab. Modify **loopc.c** to implement the filter.
Show that the filter you implement on the DSK meets the specifications.

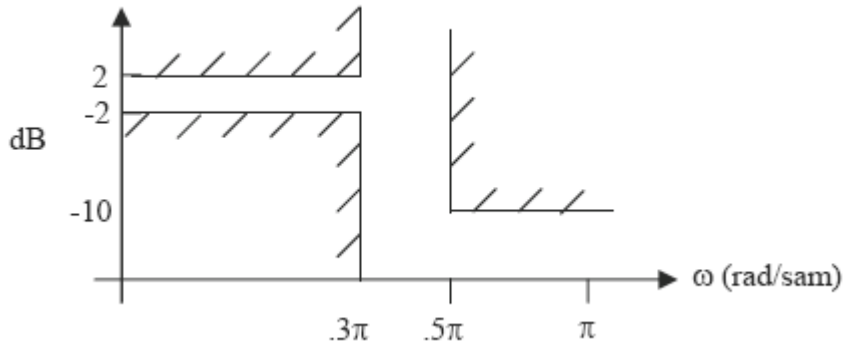

Figure 3.1. *Specifications for the Lowpass filter of experiment 1.1.*

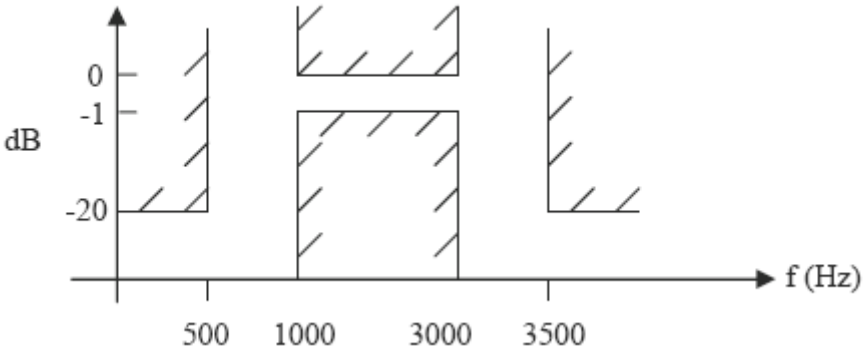

Figure 3.2. *Specifications for the Bandpass filter of experiment 1.2.*