

525.446

## DSP Hardware Laboratory

### Assignment #3

Due : 9/30/2009 (electronic project archive before 7:20 p.m.)

### FIR, Lab 1

#### Introduction

This laboratory assignment will introduce the implementation of a FIR filter in a DSP. In addition, the DDS developed for Lab 2 will be used as an “agile frequency source” to create a chirp signal to test the frequency response of the FIR filter implementation. This assignment will focus on designing a filter in MATLAB, implementing the filter entirely in C, and measuring its frequency response. This is the first of a two-part assignment – next week’s lab, **FIR, Lab 2** will focus on DSP-specific features that can accelerate the FIR implementation as well as DSP libraries optimized to perform fast filtering.

#### Reading

Read Chapter 4 (Finite Impulse Response Filters) of Chassaing. Some of the chapter should help you with this assignment, and in preparation for the **FIR, Lab 2** assignment.

#### Simulation

Design a 64-tap bandpass FIR filter for a sampling rate of 48 kHz. Use the MATLAB application FDATool. The passband should extend from 3000 Hz to 5000 Hz, and signals 1000Hz out of the passband should be attenuated by at least 40dB. Plot the theoretical amplitude response in dB (note that this can be done directly from FDATool or with the freqz command). Export the coefficients for use in the DSP implementation.

#### DSP Implementation

The block diagram below illustrates the processing architecture.

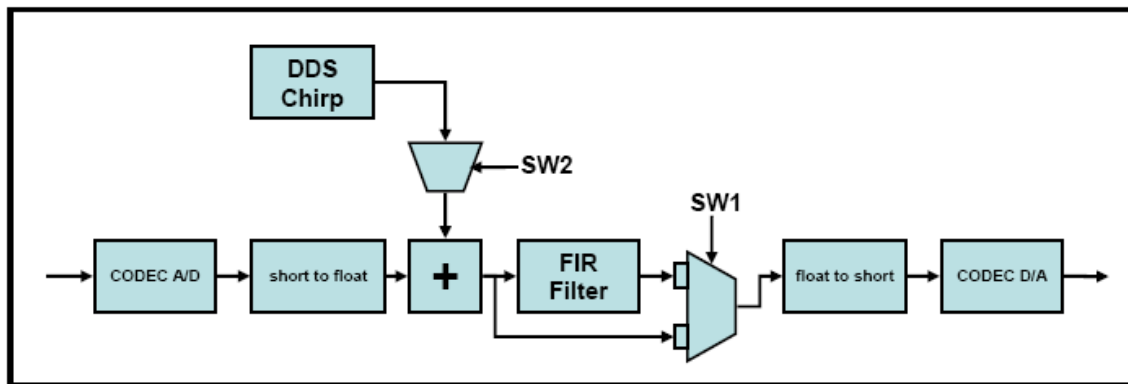


Figure 1: Processing architecture.

Implement the filter designed above using either an interrupt or polling to process audio data received from the codec. Perform all filter processing using single precision (i.e., use the datatype “float”). The FIR filter implementation has the following additional requirements:

- IF SW1 is depressed, light LED1 and include the FIR filter loopback path. If the switch is not depressed, do not include the FIR filter. This will allow the processing system to run in “loopback mode” and in “loopback filtered mode.”

- If SW2 is depressed, light LED2 and add a linearly chirped DDS signal into the codec input, prior to your filter. The DDS signal should repetitively chirp from 100Hz to 12kHz over a approximately a 12 second period.
- Light LED3 for the entire time period you are processing a new input sample.
- Rather than shifting samples, your program should implement a circular buffer manually (i.e., don't use any assembly, including the built-in support for circular addressing – perform the address calculations in C).
- Unless instructed otherwise, compile the implementation with the symbolic debug support and no optimization.

### Analysis

Perform the following analysis on the FIR filter implementation:

- Measure the amplitude response of the filter. You can do this by including the DDS chirp and the filter in the processing chain. Compare with the theoretical response.
- Measure the amount of time it takes to execute your filter processing (you may include the DDS chirp generation). You can do this by measuring the duty cycle of the LED3 signal using an oscilloscope available in the computer lab. You may also measure using the timer method described in class.
- Longer filters will provide sharper transitions and more stopband attenuation. How long of a filter do you estimate that you can implement without code changes? Explain.
- Recompile with the highest level of optimization (and support for symbolic debugging off), and re-measure the amount of time it takes to execute your filter processing

### Grading

To allow us to grade your implementation, please do the following:

- Turn in a very simple document with:
  - The following plots:
    - Theoretical Frequency response of the filter generated in MATLAB
    - Measured frequency response of the filter as implemented in the DSP
    - If the above plots don't match, concrete justification is needed
  - A summary of the measurements from the Analysis section (execution time with/without optimization, and your estimated longest filter achievable)
- Upload an electronic copy of the self-contained code composer project (zipped please).
- Demonstrate the operation of the filter to an instructor.

### References

The general structure for this lab, as well as some of the particular details, was taken from *Communication System Design Using DSP Algorithms, With Laboratory Experiments for the TMS3206711*.

## Grading Sheet

### In-Class Demo

\_\_\_ / 10 : Hook DSK input to PC or iPod. Observe pass-through mode where output = input

\_\_\_ / 10 : Depress SW1. LED1 lights

\_\_\_ / 10 : signal from PC or iPod is perceptibly filtered

\_\_\_ / 10 : Release SW1, Depress SW2, observe smoothly swept tone over approx 12 seconds

\_\_\_ / 10 : With no input source other than DDS, qualitatively demo the FIR filter

### “Report” :

\_\_\_ / 10 : Plots requested

- Theoretical freq response of short and long filters
- Recorded frequency response of short and long filters
- Explanation of deviations is absolutely necessary

\_\_\_ / 10 : Summary of measurements from the analysis section of the lab

- Filter processing time
- Maximum filter length estimation and justification?
- Improvement via optimization?

### Code from zipped electronic submission :

\_\_\_ / 10 : Circular Buffers implemented in C (i.e. no data shifting)

\_\_\_ / 10 : Code clarity, quality / commenting

\_\_\_ / 10 : Appropriate Submission – i.e. only two files received by instructors.

- 1) zipped or project directory with all required files, and immediately compilable project. Zipped file Name = lab3\_yourlastname
- 2) Images / results of measurements are in 1 self contained document