



Figure 1: Front end of a receiver.

ECE-486 Laboratory 4, Spring 2009 Due March 30

Objectives

- Learn about complex representations of bandpass signals.
- Implement a receiver “front-end” consisting of a complex mixer and low-pass filter in real-time.

Assignment

Background

Figure 1 shows the front end of a receiver. The input to the receiver is a bandpass signal containing information that occupies the band near discrete-time frequency f_0 . The complex mixer translates the incoming bandpass signal to “baseband”, and the lowpass-filter removes any undesired signals which are outside of the signal band. The output signal $y(n)$ is a complex signal which preserves all information required to reconstruct the original bandpass waveform.

You will implement this complex receiver front-end to examine an incoming bandpass signal at $12.5 \text{ kHz} \pm 1 \text{ kHz}$.

Tasks

TASK 1: Using a pre-defined function interface, develop and test a subroutine to digitally implement the function of a (real) mixer in a real-time DSP system under Linux.

One common way of implementing a mixer is by using a lookup table. A lookup table contains a minimum number of terms required to store a periodic waveform. By circularly accessing the table of values, an incoming signal may be multiplied by any periodic waveform (sine, cosine, square-wave, etc.).

The required function interface is defined in the header comments of the `mixer.c` and `mixer.h` files provided on the lab web site. You are **not allowed** to change the format of any of the function calls. Complete the `mixer.c` and `mixer.h` files, and develop a separate routine to test your implementations.

TASK 2: Design an IIR filter to use as the lowpass filter of Figure 1. Use the interactive filter design toolbox in Matlab (`fdatool`). Your filter must have a passband of 1 kHz and reject frequencies above 1.5 kHz by at least 70 dB. Using the filter design toolbox, design a 'Chebyshev Type II' IIR filter of a minimum order. Use a sampling frequency of 48 ksps. Under 'Magnitude Specifications', set $A_{pass} = 1$ dB and $A_{stop} = 70$ dB. Record the order of the filter, number of coefficients and turn in the plot of the magnitude response.

TASK 3: Implement the receiver front-end (complex mixer followed by the IIR lowpass filter) in real time under Linux.

To implement a complex mixer you will need two lookup tables: one for the real part of the signal, and a second for the imaginary part.

Note that the output of your system is a complex waveform $y(n)$. You can view your system output using the $x - y$ mode of a scope by writing the real part of the output to the left channel and imaginary part of the output to the right channel.

Please address the following questions:

1. Run your real-time implementation of the complex receiver front end. Use a sampling frequency of 48 ksps, and record the average input buffer size (in samples), the average execution time for your code (in μsec per input buffer), and the total time associated with each input buffer (in μsec).
2. How many multiplications per second are required for each of your IIR filters?
3. Using the interactive filter design toolbox in Matlab (`fdatool`), design an FIR filter that meets same filter specifications as the IIR filter designed above. Use the FIR equiripple design to obtain a minimum order filter. You must specify the filter order (use $W_{pass} = 17.88$ and $W_{stop} = 3162$). Adjust the filter order until the specifications are met. Record the number of coefficients for your design and turn in the plot of the magnitude response.
4. How many multiplications per second would be required for each filter if the above FIR filter was used?
5. Since the filter output bandwidth is 1 kHz, the sampling frequency of the filter output may be reduced. For example, the sampling frequency could be reduced by a factor of 16 resulting

in 3 ksps. This implies discarding 15 out of every 16 output samples. For the FIR filter the discarded samples don't need to be calculated. For this 3 ksps output rate (assuming the same FIR filter is used) how many multiplications per second would be required for each filter?

What should be handed in:

1. Hand in
 - (a) Source code listings (the completed `mixer.c` and `mixer.h` functions, your test program, listings for your real-time implementation, and any other Matlab/C code),
 - (b) Plots of the FIR and IIR filter magnitude responses,
 - (c) Answers to the above questions,
 - (d) Documentation showing how you tested your routine and results showing that it works.

This documentation will be collected at the beginning of class on the due date.

2. Create a TAR archive containing all C source code files (.c and .h files) required to implement your receiver front-end in real-time (including your modified `mixer.c` and `mixer.h` files). Include the 'Makefile' required to compile your real-time program. The TAR file should be labeled `lab4.tar` and uploaded on the ECE 486 website. The TAR file must be uploaded before class (11:00 AM) on the due date.
3. Meet with the TA during your assigned group meeting time in the week of the due date.