**ECE 4522/5514: DIgital Signal Processing**

# Computer Assignment (CA) No. 1: Real-Time Audio

The goal of this assignment is to familiarize you with how to process real-time audio in a C/C++ program. We will assume you have access to a Linux or Mac laptop (dual-boot your machine if necessary). These things can be done under Windows as well using a combination of tools such as Powershell.

The tasks to be accomplished are:

1. Install sox (which should be part of most Linux distributions).
2. Practice recording and playing audio from the command line with these commands:

(a) To record audio from your microphone into a file:

rec -e signed -b 16 -r 8000 -c 1 -t raw x.raw

This records a raw file, x.raw, containing one channel of data represented as 16-bit signed integers. We can modify this command to send the recorded data to stdout:

rec -e signed -b 16 -r 8000 -c 1 -t raw -

(b) To play an audio file from the command line:

play -e signed -c 1 -b 16 -r 8000 -t raw x.raw

This is the reverse of the command in (a).

(c) To pipe these together:

rec -e signed -b 16 -r 8000 -c 1 -t raw - | play -e signed -c 1 -b 16 -r 8000 -t raw -

This is a pretty interesting command. It records audio from your microphone and sends it to stdout. The second program reads the data from stdout and plays the data. You will observe a noticeable delay, which is precisely the issue we are exploring in this assignment.

1. Create a program that reads data from stdin and writes to stdout. Your program should read a block of M samples, where M is defined as a constant at the top of your program. We will refer to M as the frame duration in samples. Your program should also buffer M samples of data where M > N. We will refer to M as the window duration in samples. M will include the N most recent samples of the signal.

Your program should take arguments as follows:

my\_buffer\_program –sf 8000.0 –fdur 0.02 –wdur 0.03

where the sample frequency (-sf) is specified in Hz, and the frame (-fdur) and window (-wdur) durations are specified in seconds. Compute M and N from these arguments.

Test your program as follows:

cat x.raw | my\_buffer\_program –sf 8000 –fdur 0.02 –wdur 0.03 |
play -e signed -c 1 -b 16 -r 8000 -t raw -

1. Now you can put this all together to form a real-time sequence of operations:

rec -e signed -b 16 -r 8000 -c 1 -t raw - | my\_buffer\_program –sf 8000 –fdur 0.02 –wdur 0.03 |
play -e signed -c 1 -b 16 -r 8000 -t raw -

This will essentially record audio, pass it through your buffer program, and play the data. We will use this structure to implement and test many DSP programs in this class.