CS 454/654 Lab #4
Linux Timers, Signal Handlers, and DAQ

Spring 2025 (*Revision*: 2.0)

Demonstration due the week of Mar 24, 2025.

1 Overview

In this lab you will put into practice your knowledge of timers and real-time signal handlers under Linux and realate that to some basic notions of frequency-domain analysis. The objective is to generate square waves with the LabJack U3-LV USB data acquisition system (DAQ) and see their spectrum in the frequency domain. You must accomplish the following tasks.

1. Theory: Write a Python script to generate a 1 Hz square wave with amplitude between 0 and 5, and from time 0 to 10 seconds, and with a sampling rate of 10,000 Hz. Plot the wave using matplotlib. (5 points)
2. Theory: In the same Pyhton program, compute and appropriately post-process (normalize and shift) the Fast Fourier Transform (FFT) of the square wave signal generated at the previous step. Plot the FFT in a second sub-plot. (5 points)
3. Practice: Write a C program for the Linux system on the host machine to read two floating point voltages between 0 and +5 volts from the user. (5 points)
4. Practice: Compute and display the maximum frequency of the square wave that can be generated with the system-wide realtime clock (`CLOCK_REALTIME`). (10 points)
5. Practice: Read a floating point square wave frequency in Hz from the user. (10 points)
6. Practice: Use a timer, a `SIGRTMIN` handler function, and the LabJack U3-LV USB DAQ to generate a digital square wave. (20 points)
7. Practice: Use a second timer, a `SIGRTMAX` handler function, and the LabJack U3-LV USB DAQ to generate an analog square wave between the given voltages at the specified frequency. (30 points)
8. Practice: Provide a user prompt to terminate the program and gracefully release the USB device. (5 points)
9. Practice: Use the FFT function of your oscilloscope to compute the Fourier Transform of the digital square wave generated by your program. Compare & contrast the FFT produced by the scope with the theoretically generated one (step above) when the wave parameters are comparable. (10 points)

2 Procedure

1. Before getting started, read sections 6.4 through 6.6 in the CS-454/654 Laboratory Manual (available on the course website, as well as on Piazza).

2. Setup for Lab 4:

   (a) This lab will be performed entirely on the lab PC. We will not use the Amazing Ball System for this lab.

   (b) For the theoretical portion of the lab, take a look at the demo Python script that will be shared right after the lecture in which FFT is covered and live demoed.

   (c) For the practical part, download the auxiliary files for this lab from the Piazza page or the course website. Uncompress the *Lab4.tar.gz* archive on the development machine.

   (d) The auxiliary files archive contains four files: *u3.c* and *u3.h* are used to define and implement the communication functions with the LabJack U3-LV USB DAQ. You should not modify these files. A template of the square wave generator that you will implement is provided in *wave.c*. A makefile is also provided to compile your code with the required flags. Test that the template file will correctly compile with the provided makefile.

3. Next, update *wave.c* so that it fulfills the requirements specified in the Overview section. When doing so, please note:

   (a) Header files required by a library function or system call can be found at the top of the function's man page.

   (b) Use `clock_getres()` to determine the maximum frequency square wave that can be generated. (Note that you will need to link with the realtime library, `librt`. Check that this is the case in the provided Makefile.)

   (c) As specified in the template code's comments, the digital square wave will need to be produced in output on the `FIO2` line of the LabJack U3-LV DAQ. The analog square wave will need to be produced in output on the `DAC0` line of the LabJack U3-LV DAQ.

   (d) You should use the lab oscilloscope to verify your program is working correctly. The recommendation is to hook up the oscilloscope of your station in the followin way: (1) connect Channel 1's probe to `FIO2`; (2) connect Channel 2's probe to `DAC0`; (3) connect the ground end of one of the two probes to any DAQ's ground (GND) line. Follow the directions of the TF to make sure your connections are short-free and well secured.

   (e) The capability of the DAQ to generate a digital/analog square wave is limited to a certain range of frequencies. Also make sure your program can properly handle small frequencies, but also that the requested frequency is not too high to be correctly generated by the DAQ. For digital I/O, the DAQ should be able to handle up to 1 kHz waves.

   (f) Once everything is setup, the program's main should only wait for user's inputs. with a simple command prompt. If the "exit" command is entered by the user, the program

is terminated. Before termination, remember to appropriately release the USB device. See lab manual for directions.

(g) In order the to use the oscilloscope to compute the FFT, do the following. First, make sure that you have the scope configured correctly to display the digital wave in the time domain. Make sure that the time scale is set correctly to see about ten pulses of the wave on the display. Next, use the `Math` menu and add the `FFT` measurement, making sure to select the same channel (e.g., channel 1) used to display the digital wave as the `FFT Source` in the side menu. It might take a few seconds for the scope to update/display the FFT. It will appear as a red (very messy) line on the display.

(h) Once you have set the scope to acquire the FFT, use the *Single* mode of the scope to take a static snapshot of the square wave and its FFT. Then, zoom-in on the FFT to focus on the range from 0 Hz to 100 Hz. Qualitatively compare the shape of the real FFT to the one generated by your python script.

At the start of Lab5, each lab group will be asked to demonstrate and explain your Lab4 code to the TF.

3 Questions to Ponder

The following questions are provided for your lab group to think about. No written response is required.

1. Use the oscilloscope to determine the actual frequency of the digital wave generated when 500 Hz is requested. Repeat for 500.076019 Hz. Explain the difference.
2. What is the maximum achievable frequency for the analog wave. What impacts its frequency? And how does it compare to the digital wave?
3. What is the *width* of the FFT plot displayed by the oscilloscope and how is that related to the capabilities of the oscilloscope hardware?