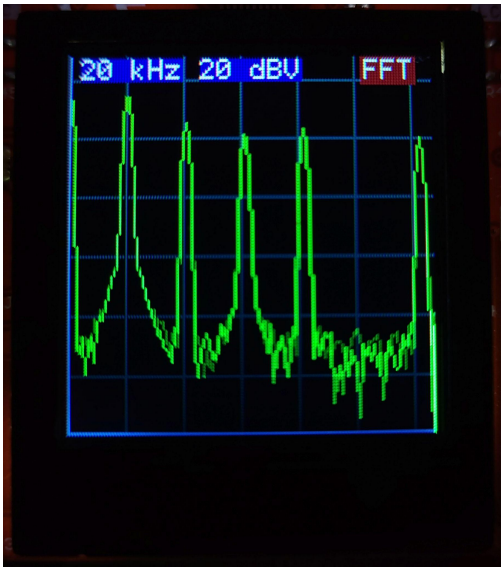
Slater Campbell
ECE3849 Lab 2
4/28/2021

Objective:

The objective of this lab was to port our existing 1-megasample oscilloscope to the TI-RTOS environment. Additionally, a spectrum analyzer component utilizing an FFT library was added to stress the RTOS implementation. I succeeded in porting the oscilloscope and adding the requested features. Button inputs are processed quickly and the FFT runs at a good framerate.



Discussion and Results:

Porting my existing software to the TI-RTOS framework was easy. First, I copied over the necessary includes, defines, and files from my original implementation. Then, I modified the settings in the rtos.cfg to port the ADC ISR as an Hwi. Once this was complete, I made empty functions for the waveform, processing and display tasks. I assigned the waveform task a priority of 15, which is the highest. I gave the processing task a priority of 1, which is the lowest. The display task initially had a priority of 14, but as I added more tasks I eventually changed this to 12. I placed the IntMasterEnable() line in the waveform task before the loop, and configured the semaphore associated with this task to start with a post. The waveform task posts to semaphore processSem0, which is pended on by the processing task. When the processing task finishes its operation, it posts to waveSem0 which is used by the waveform task, and displaySem0 which is used by the display task. After copying my code from Lab 1 into these tasks, I was able to get the original oscilloscope functions to work (minus user input).

Next, I added the button processing code. I removed all timer configuration and FIFO code from the button handling files I already had. Then, I configured the clock module as described. When the clock module activates, it calls a function which posts to the button scheduling semaphore. Another task pends on this semaphore, and then calls what used to be

the button ISR. This button read function determines whether there is an input, and if there is, it is debounced and put into a mailbox to be handled by the user input task. The user input task pends on the mailbox, and when it receives data, it updates global state variables accordingly. The waveform task, processing task, and display task all incorporate these global variables to determine what to do. After the user input task handles an input, it posts to displaySem0 to trigger a display update. The task that calls the button reading function has a priority of 14, which is the second highest priority. The user input task has a priority of 13.

Once I got button inputs working, I implemented the FFT code. This was relatively simple. I modified the starter code slightly, tested it with the the test sinusoid and verified the right values. Then I added the Blackman window function, and tested it, but the result was upside down and flipped. To fix this, I modified my display code to map the end of the buffer to the left edge of the display. I also changed the pixel height to be LCD_VERTICAL_MAX-(output of the FFT), in order to flip it top to bottom. I did not need to add any offset to fit the noise floor and peaks on screen.

Conclusion

I really enjoyed working through this lab and solving each issue as it came up. The most common issue was the stack size being too small. I enlarged it for the functions that needed it, but at one point even the larger stack size for the processing task wasn't enough. I realized that declaring a 1024 element array of floats would take a ton of memory, and when context switched to a different task, it would be put on the stack. I initially fixed this problem by declaring the array as static, but then I found a way to skip using it entirely.

In this lab, I gained experience implementing code in an RTOS environment. I learned how to port code over and how to write new code in this context. I also got a sense of the added flexibility that RTOS provides, at the expense of processing speed (button inputs and framerate are a little slower on this version). This lab took much less time than lab 1, and in the future I think it could be interesting to make the time domain extra credit from the last lab be required in this lab.