E E 474 Intro to Embedded Systems
Robyn Jung 1976925
Margot Enright-Down 1776361

Lab 4 Report: LCD Display, Touch Screen, and RTOs

Task 1: Using LCD driver to display temperature reading.

Procedure:

First, we had to complete the LCD_GPIOInit function in SSD2119_Display. The instruction was straightforward and it was an easy task. After checking the LCD with LCD_ColorFill(), we moved on to Task 1b, which was implementing the previous lab with ADC onto the LCD. To finish the LCD_PrintFloat function, I implemented the same method from LCD_PrintInteger(), but with a twist. I multiplied the passed float value by 100, saved the decimal places into a char array. Since I have decided to only print until the hundredth, printing out the value in the order was straight forward. The values saved in the char array were sent to LCD_PrintChar function, to be displayed on the screen.

Task 1 initializations are the same as the previous lab. Initializations and handlers for ADC0 SS3 and Port J were the same as the previous lab.

Nothing needed to be fixed in SSD2119_Touch. Using the provided functions in the SSD2119_Display, I hand calculated the coordinates from the LCD width and height, drew virtual buttons, and displayed the temperature reading on the top of the screen. It was tricky to evaluate the touch input. There was no information on the return values of Touch_ReadX & Y functions, so I had to find the minimum maximum coordinates by manually testing. Then, I made a nested if statements for each button presses.

Result:

Program performed as expected with a room for update. Because the frequency being used was quite low, polling took a while and I had to press the buttons for quite a long time. I am not sure if it is possible to use 2 PLLs but that could be a solution. Another thing that comes to mind is that my System clock was not 16MHz in the previous lab. Maybe it is configured to something else again and that was causing the delay in run speed.

Task 2a: Implementing Traffic System on the LCD.

Procedure:

I used the same drivers and Initialization for the PLL. I changed the frequency to 60MHz, got rid of the interrupts and decided to use polling. After displaying the basic setup with functions in SSD2119_Display, I adopted the same method from Lab 2 with the nested if statements to verify the touch input. I tried using interrupts with the touch input, but that did not work.

Result:

Traffic light system performed as expected. There is some delay between turning on and off the lights and for a brief period of time, both lights are on while one of them turns off.

Task 2b: Task 2a with RTOS.

Procedure:

After initializing all the drivers, the next thing to do was to create tasks. I gave Control the highest priority, followed by Pedestrian and StartStop. Then, I implemented the same method to evaluate the touch input. On/off flag and the pedestrian flag is set according to the button presses. Then, the flags are used in the Control function, along with timer. The skeleton of the FSM is from previous lab. In the FSM function is the state behaviors, turning on and off the lights.

Few other edits that are not in the main are in the FreeRTOS_Config. configUSE_PREEMPTION was defined to 0, and the CPU clock frequency to 60MHz.

Result:

The result would have been exactly the same as Task 2a. Unfortunately, the program did not work.