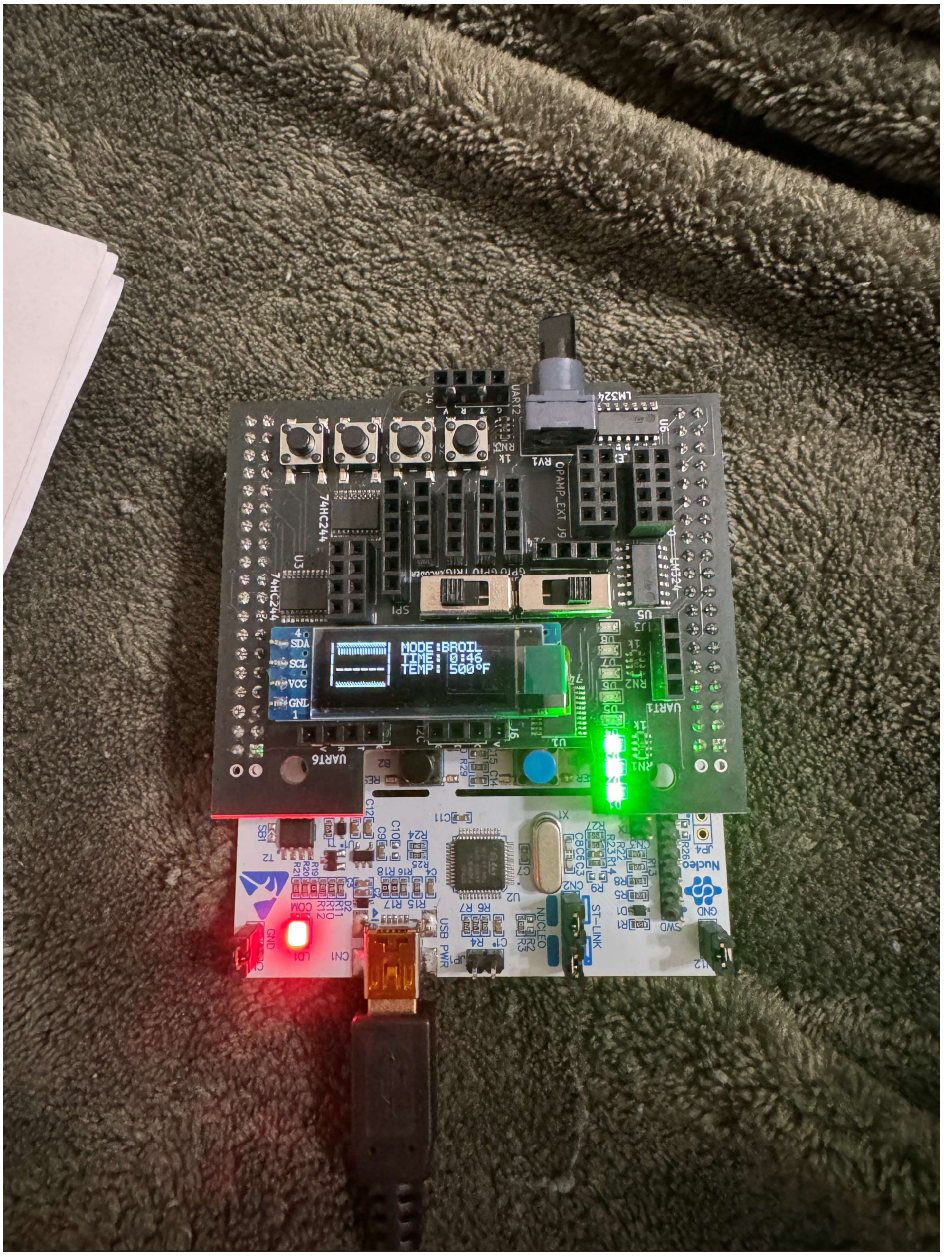
Lab 8 README

ECE 13

Sean Manger

Prof: Steve McGuire

F24



Author: <Sean Manger> (smanger@ucsc.edu)

Collaborators: For the toaster lab, I again collaborated with Donny Tang. The most notable thing we discussed was how to get the Buttons to register with some consistency. From my understanding, at least based on what I've read on our class discord server, it seems that most people in this lab had trouble getting the 100 Hz timer to register short button presses of less than a second or so. The TAs offered a few solutions on discord such as changing the clock speed of the Nucleo Board to register button pressed 20x faster than the standard time. However, I found that this didn't work for me and my button presses were still very slow and inconsistent. Donny informed me that he found out if he instead moved the button events checker to within the while loop in main the buttons started to work somewhat consistently. I applied this fix as well and was able to implement the lab plus the extra credit alarm phase. It is my understanding that the slow button presses are a bug of some sort within the Nucleo board. As a result, it has been announced that button inconsistency will be considered while grading. Besides this, I also discussed with Donny how to get the time right for the cooking phase. We know that the adc will return a value of between (0 – 255) + 1 for the final time value. This value is in seconds. Well 255 / 60 seconds / minute will give you 4 minutes. Likewise, if you do 255 % 60 you will get a remainder of 16 or 16 seconds. As such we could use the OLED function to display time in this way. Aside from this, Donny and I also helped each other perform some general debugging in regards to buttons, LEDs, and states. As always, my code is my own work and written entirely by me.

General Approach to Implementing the Toaster

For this lab, I thought that the best approach was simply to just dive straight into making the toaster. I read the manual as usual and followed two pieces of advice contained within. Firstly, I printed up a standalone copy of the toaster oven logic diagram contained within the lab manual and kept it close on hand. This was immensely helpful since I didn't have to waste time flipping back and forth between pages in the manual. The second thing I did was follow the advice to implement the updateOLED function first. This lab relied essentially entirely on what was displayed on the OLED and getting it right would mean an easier time later. To do this, I thought the simplest method would be to utilize a switch statement to output the desired graphics and information depending on what the user desired. This switch statement had 3 cases for Bake, Toast, Broil, and the optional Alarm, depending on what the user wanted to do. I also decided to prepare in advance several strings which would hold the various oven graphics such as the top being off or on. This made writing each of the print statements for the various modes much easier and far less

cluttered. After doing this, I tested my OLED function with some dummy structs and states and made sure it was outputting correctly.

After this, I went to work writing the heart of the toaster that is the run_toaster() function which controls all the FSM logic for our toaster. For this part, I found the easiest thing to do was follow the provided toaster diagram essentially verbatim. While the diagram looks quite intimidating at first, once you sit down, give it some thought, and mentally trace through it, it isn't so bad. I also didn't implement the states in any order and focused on what I wanted to do at a given time. This made it much more manageable for me and I think less stressful as well. The hardest part of writing the FSM logic for me was just managing the large amount of code and remembering what I was trying to do at any given time. With several states and many variables, it can be easy to lose track of progress. One annoyance during this phase was the inconsistent button presses but Donny's suggested fixed helped to mitigate that somewhat. While implementing the elapsed time for the various buttons I recognized that with a 5 Hz timer there would be 5 cycles per second or rather that one second would be equal to 5 timer ticks. Another difficulty I had was getting the appropriate 1/8 amount of time for the LEDs to turn off. What I initially did was divide the cooking time by 8 and multiply by 5 to convert it into an appropriate amount of timer ticks. This worked somewhat but it was slow and only about 5 – 6 of the LEDs would turn off by the end. After, fiddling with this somewhat I was able to get it closer but not exact. My ADC was also oscillating while I was setting it due to noise but once cooking began it remained a constant value. From what I understand all ADC's will have at least some amount of noise on them. As a result, the ADC and LEDs can be improved within my code. I would also like to be able to have more consistent button presses.

Conclusion

This was another very fun lab that I probably spent close to 30 or so hours coding. This was widely considered to be the 2nd most challenging lab in ECE 13 but when all was said and done it wasn't too bad. Besides a couple of improvements, I was able to implement everything successfully. If I could offer one improvement for the future, I would like to see an easier state machine lab before the toaster to perhaps mitigate some of the difficulty of this lab. Other than that, I really feel I have come a long way as a programmer now that we're in the tail end of the course.