Isabella Phung
CSE120
May 1, 2024

Lab 3: I2C LCD

3.1:

I had little difficulty completing the soldering portion of the assignment as I'm experienced in soldering a variety of components from home projects.

3.2:

This part proved the most difficult of all sections of the assignment. After looking into how Arduino's Wire and Print classes worked, I considered what would be the ESP-IDF equivalents. I went down a rabbit hole of searching through the examples provided in the esp-idf folder, specifically peripherals/lcd/i2c_oled folder, where they appeared to be utilizing some specialized drivers provided by esp-idf specifically for lcd screens. After some discussion with a TA, they explained that using these files was unnecessary and I could just use the same drivers written in lab 2. So, I copied over the driver setup code from my lab 2.2 project and used the same i2c command protocol to send commands to the LCD screen. After that, I did experience some difficulties when trying to compile my code as the driver code I had copied over followed C formatting, not C++ formatting. From this page, I just address each element individually when assigning them their values during driver initialization. I also changed my lab3_2.c file to a cpp file since it didn't recognise the "class" keyword and added extern "C" in the app_main() function declaration. After that, I found that the LCD did display the message, although I could not get the RGB to work no matter what. A TA showed me the slack message, which indicated changing the header file with a different RGB address and adjusted registers, but even after changing it, the LCD would not light up. I tried 0x2d, 0x2B, 0x60, which I had found on the DFRobot official github example project, as well as 0xC0, which I found on the official datasheet; nothing worked. After a few hours attempting to debug it, with further discussion with a TA, they accepted my project to be checked off regardless of the RGB functioning or not.

3.3:

After getting 3.2 to work, this part was fairly straightforward and just involved me copying code over from lab 2.2. I copied all of lab 2.2 into a file called "temp.c", removed the driver install code, then copied the app_main in temp over to the app_main in lab3_3.cpp. It was then as simple as adding an #include "temp.c" to the top of lab3_3.cpp to get everything working. In order to display the temperature and humidity data correctly formatted, I found I couldn't directly enter the values into the lcd.printstr as it was not designed for string formatting. So I used sprintf to format the values before displaying them.