5/9/2024

CPE 301.1001

Timmy Norris, Jackson Robertson, John Althoff

Final Project Technical Document

Overview

Basic Overview

For our final project, our group decided to not go with the schematics provided for a water cooler system. Instead, we decided to make our own project, and chose to do a smart lock system. For a smart lock system, we need: a keypad to get the user's passcode, a key fob reader to automatically unlock the lock, a system within the keypad to lock the lock, a buzzer to send off a noise to signal the door locking and unlocking, a spinner to replicate/act as the lock locking and unlocking, and an internal system that sets off after the temperature gets too high. There was also an LED screen that we used, a real time clock, and a button to activate an interrupt to shut the system off completely, which would be shown by a red and green LED.

Constraints

Some of the basic constraints that we were tasked to deal with were not using any of the excluded Arduino libraries such as Digital.Read(), Serial.Begin(), and delay(). To get around the use of Digital.Read(), we went back to lab 3 which taught us the use of GPIO, which required us to set the registers in the code, and in the setup function, set the pins to be either a read or write pin. This was crucial in getting the buzzer, button, and LEDs to work properly. The registers were also used for the LED screen, real time clock, RFID sensor for the key fob, and spinner. The temperature sensor was not digital, as it was an analog mechanic. For analog, we referred to Lab 7 which taught us to use ADC functions. Not only this, but we also needed to decide an appropriate temperature threshold to trigger our fire alert system, which we decided would be 120 degrees or higher. Next, we need to figure out how we could make an actual lock/passcode that functions with the user input. To do it we set the passcode outside the scope of both loops in a char array, and another char array to store the users input that is set inside the loop. The buzzer was taken from both labs 2 and 8, as those handled with making a piano using the buzzer. We took the my_delay() function from both, and that would replace not only the delay() function, but also trigger the timer for the buzzer to activate. The key fob reader we had to do actual research for as this was something we had not done in any of the labs. To get it to work, we needed to use an RFID reader that would get the ID that is associated with our key fob and check to see if it matches what we set in the outside of the loops. One of the specific constraints for the RFID reader is that it cannot take more than 3.3 volts of power, otherwise it will fry the component. To create our keypad, we referred to lab 6, which was the lab that involved creating a game on an LED screen. We had to make sure that the keypad could begin the process of unlocking and locking the smart lock. To do this, we had the # and * symbol on the keypad relate to locking and unlocking, respectively. Now for the LED screen, we had originally gone with the provided screen that we had used in Lab 6, but decided to use another, simpler screen to not create a wire clutter on our breadboard. The one we did use was an Adafruit_SSD1306. This allowed us to create an LED screen with text that could be shifted and font that could be increased or decreased. It did create more complicated code but was worth it as it made the project look cleaner and less cluttered. The spinner was another project that we had to do some research for to get working, specifically getting the wiring complete and get it to rotate clockwise and counterclockwise. Setting up the real time clock to properly determine the time of events of the smart lock was another component we had to do some research to get working. We ended up daisy-chaining the RTC with the Adafruit display to simplify the wiring. Lastly, the interrupt was the easiest function to implement, as we could use the interrupt library.

Research and Documentation

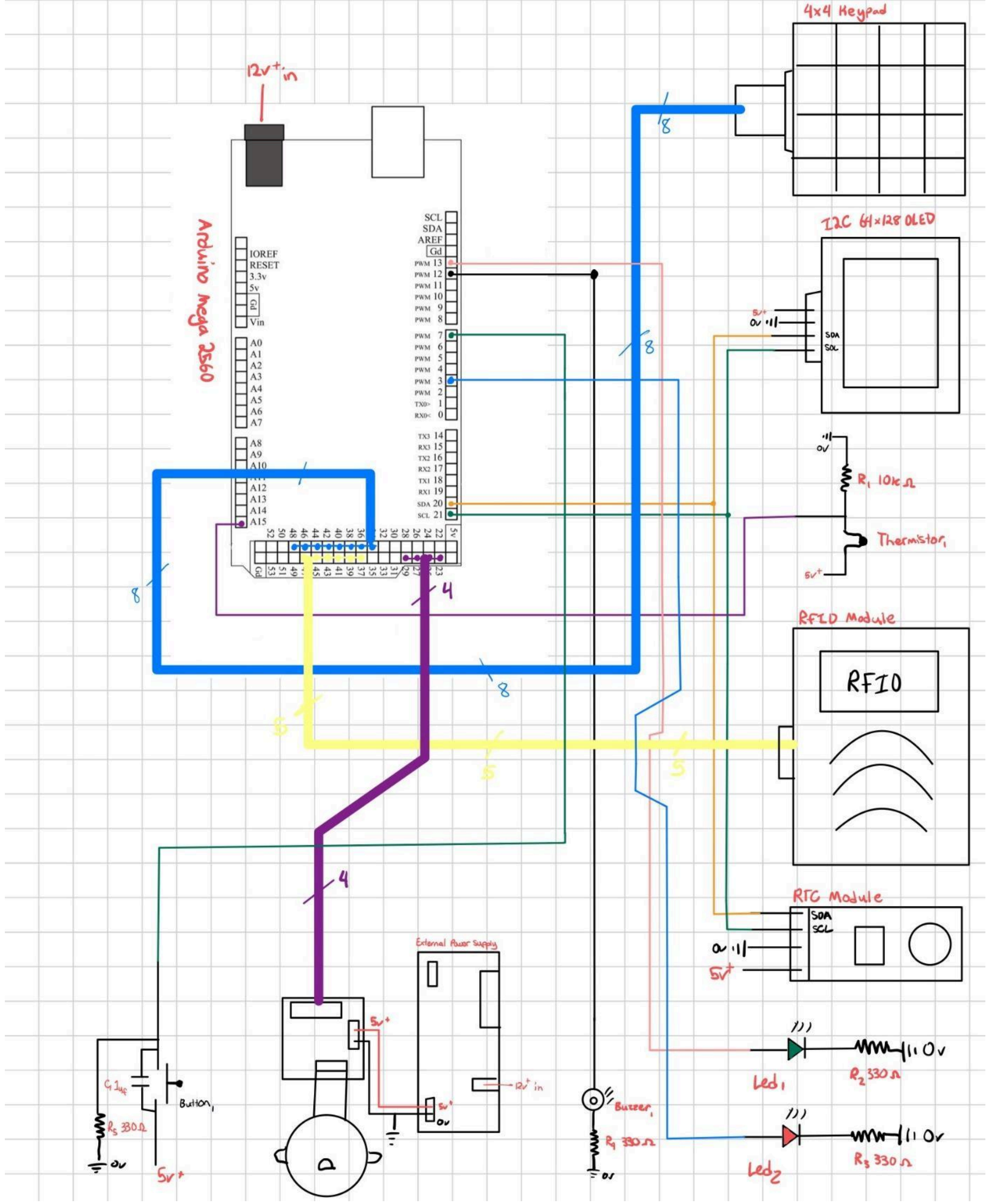
We began our research for this project by finding the documentation for each component we used based on the part number of each given component. Once we had all the datasheets, we compiled them into an excel able that we could use to pull information from. We then determined the operating voltage and current ranges for each part, most of our components call for 5v but specifically our RFID module called for 3v.

CPE 301 Final Project Component Datasheet Repository	
Component Name	Datasheet Link
Arduino Mega 2560	Datasheet
Atmel 2560	Datasheet
RC522 RFID	Datasheet
DS3231 RTC Module	Datasheet
SSD1306 128x64 I2C OLED Display	Datasheet
2x Led 3v	Datasheet
1x Pushbutton	Datasheet
3x 330ohm Resistor	Datasheet
2x 10kohm Resistor	Datasheet
Thermistor	Datasheet
Stepper Motor	Datasheet
ULN2003 Driver	Datasheet
Active Buzzer	Datasheet
0.1uf Capacitor	Datasheet

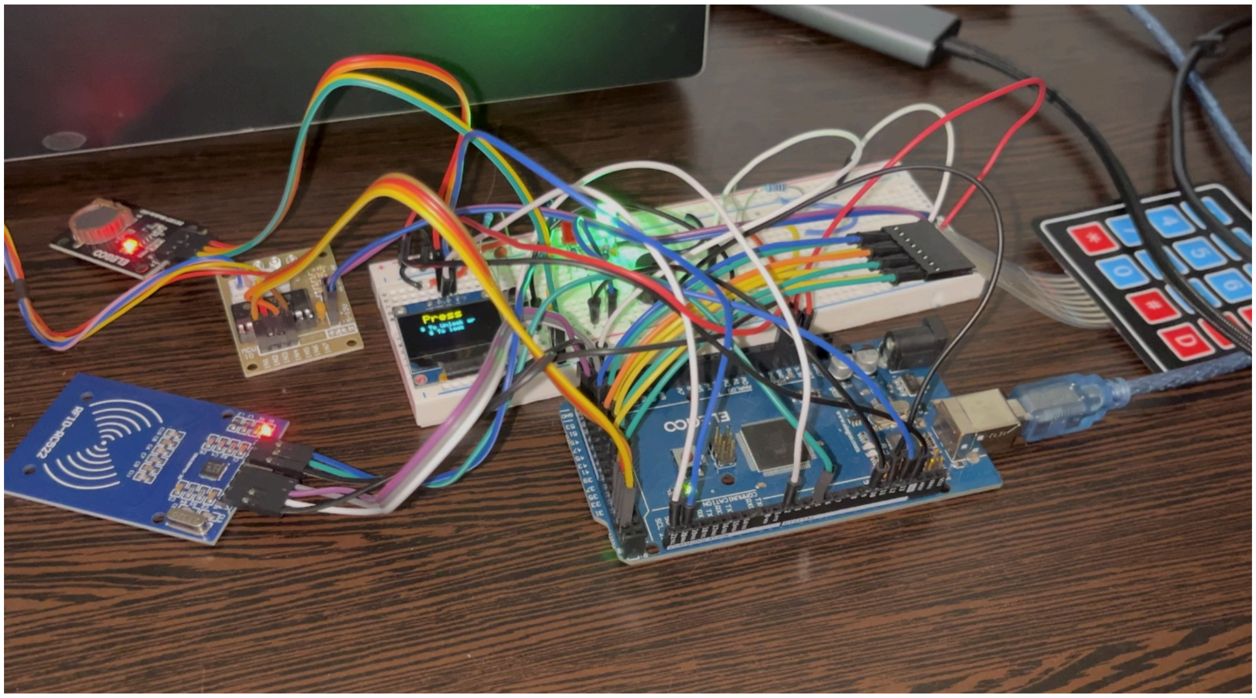
4x4 Membrane Keypad	Datasheet

Project Schematic

Cpe-301 Final project Schematic



Final system and Video



GitHub Repository

https://github.com/1103-Althoff-John/CPEGroup (NOTE: The file titled finalBuild.ino is the one that has the complete smart lock system code)