## EECE.4520/5520 Microprocessor System 2 and Embedded System

Fall 2021 v2020.08.31

1. **General Information**

**Lab 3: Controlling a Fan**

**Student Name: Elizabeth Brown**

**Student ID number: 01623721**

**Team Name/Number: The Blood Oranges**

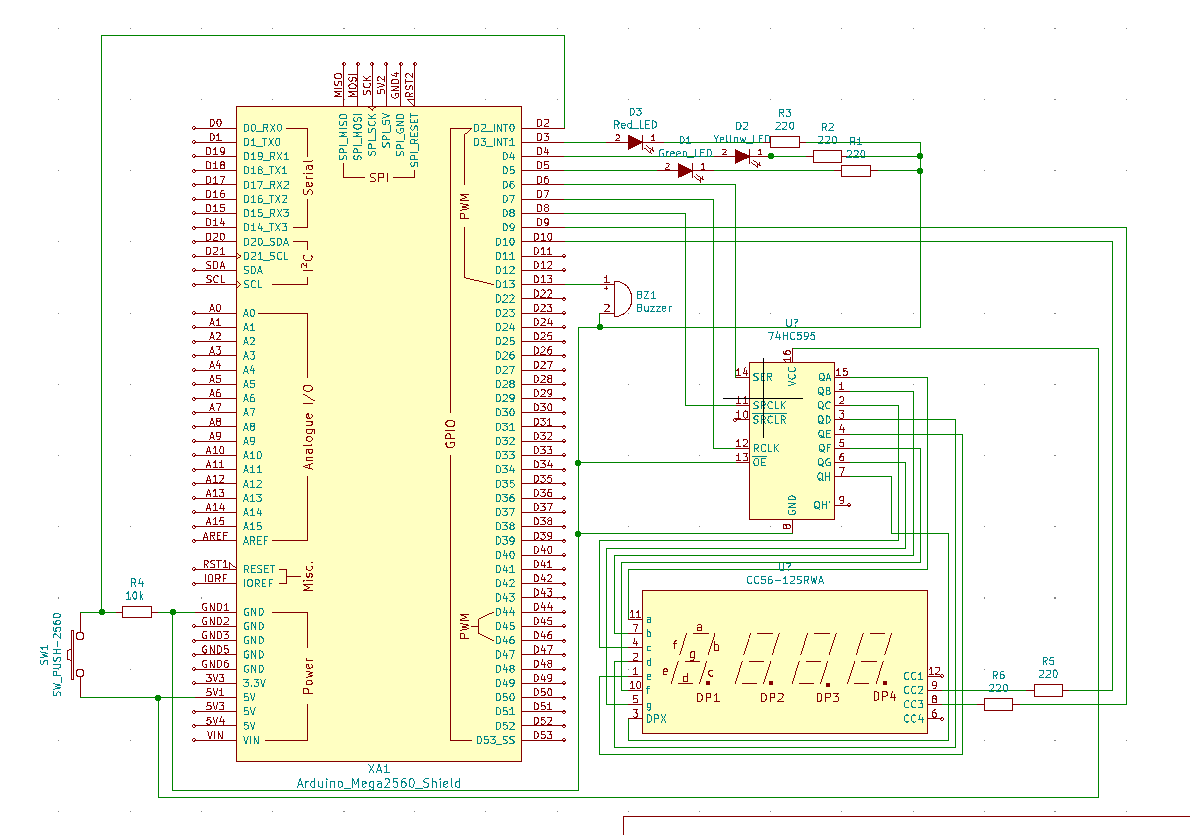
**Team member names: Elizabeth Brown, Joe Taylor, Cody Bellec**

**Date of completion: November XX, 2021**

**Demonstration method: Discord Call w/ Recorded Video**

**YouTube Link:** <https://youtu.be/jiMCETV5qKY>

1. **Design:**
   1. **Hardware Design**



The button is connected to 5V on one side, and a 10kΩ resister to ground and digital 2 on the Arduino on the other side. The buzzer is connected to digital 13 and ground. The red LED is connected to a 220Ω resister, digital 3 and ground. The yellow LED is connected to a 220Ω resister, digital 4 and ground. The green LED is connected to a 220Ω resister, digital 5 and ground. The shift register (74HC595) has an input voltage of 5V and ground is connected to Arduino's ground. The shift register’s 14th pin is connected to digital 6. The shift register’s 11th pin is connected to digital 8. The shift register’s 12th pin is connected to digital 7. Pin 13 of the shift register is connected to ground. The 7-segment's pin 8 is connected to a 220Ω resister and digital 9. The 7-segment's pin 9 is connected to a 220Ω resister and digital 10. Pin 3 of the 7-segment is connected to pin 7 of the shift register. The 7-segment pin 5 is connected to pin 6 of the shift register. Pin 10 of the 7-segment is connected to pin 5 of the shift register. The 7-segment's pin 1 is connected to pin 4 of the shift register. The 7-segment's pin 2 is connected to pin 3 of the shift register. The 7-segment's pin 4 is connected to pin 2 of the shift register. The 7-segment's pin 7 is connected to pin 1 of the shift register. The 7-segment's pin 11 is connected to pin 15 of the shift register.

* 1. **Software Design**

The code first declares the pins of the Arduino to initialize the hardware components. The variables are then initialized for future use. The interrupt service routine (ISR) was then initialized for the button and timer. Then the outputs and inputs were then initialized. The interrupts are then disabled in the code so the timer interrupt can be initialized. After the timer interrupt is declared, the interrupts are enabled again. The 7-segment outputs were disabled for the interrupt case which flashes the red LED until the flag toggles back. The 7-segment display then uses the shift register to display the remaining time. The colors of the LEDs are then initialized to red, yellow, and green. The 7-segment display uses the timer to display the remaining time and to play the buzzer when there are 3 seconds left on the timer. In the loop, the states are then rotated according to the timer.

Link to source code on GitHub: https://github.com/joetaylor010/micro2/blob/main/Lab%201/TrafficLight/TrafficLight.ino

* 1. **Results**

Photo of the model:

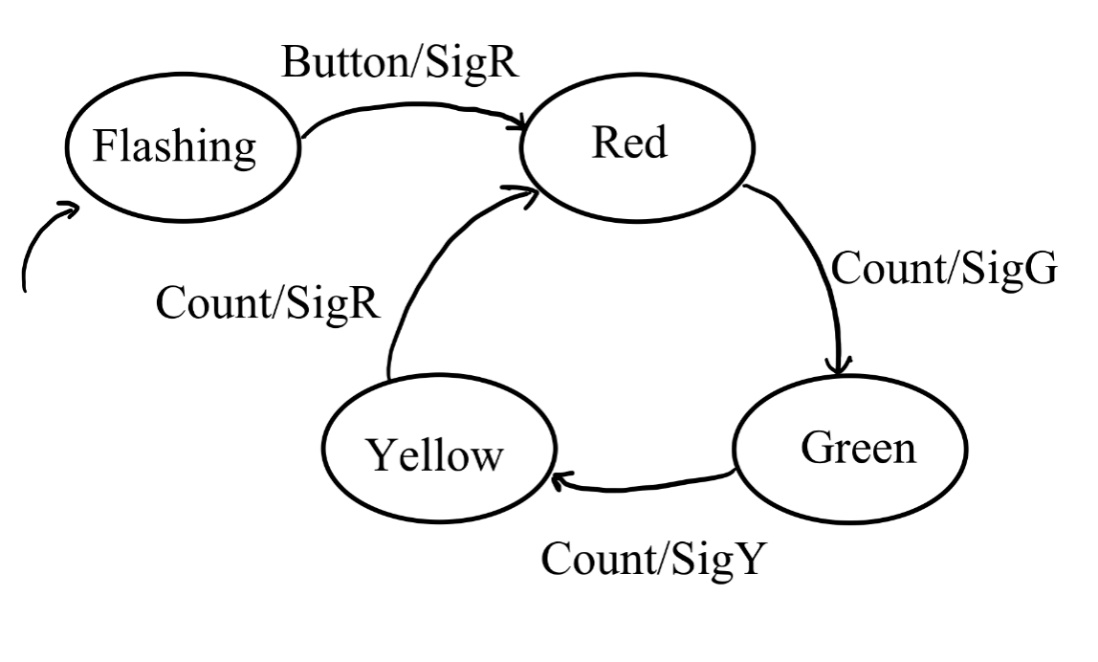


Photo of completed project:

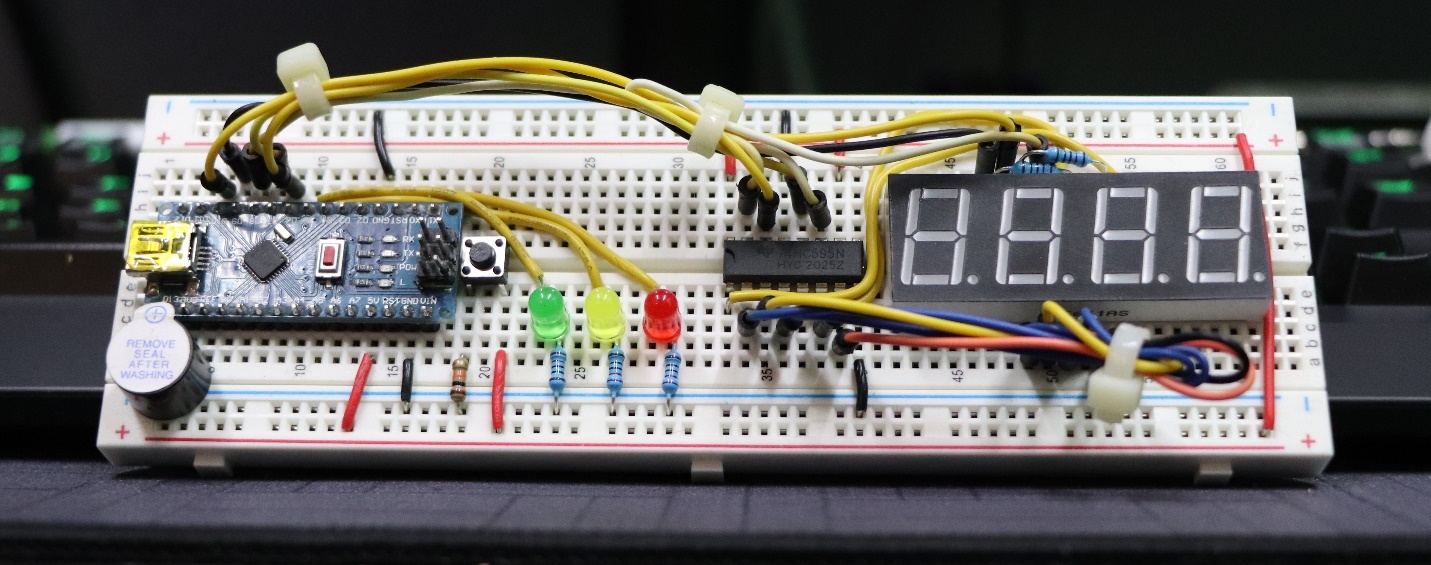
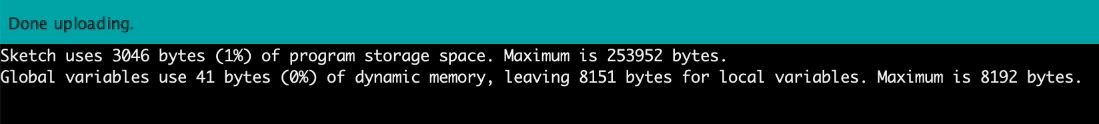


Photo of the board successfully uploading the code:



The board is initialized in the red flashing light stage. In this stage, the flashes on and off until otherwise told not to. By pressing the button, the circuit changes to the loop. Starting with red, the light is steady for 20 seconds and the 7-segment displays the remaining time. When the time reaches 3 seconds left, the buzzer sounds once on each remaining second. The red LED turns off, and the green LED turns on. The 7-segment display resets to 20 seconds, and in the remaining seconds, the buzzer makes sound like it did when the red LED was on. The green LED then turns off and the yellow LED turns on. This time, the timer is only set to 3 seconds, so the buzzer sounds on each second while the 7-segment counts down from 3. The loop then starts over again.

1. **Problems Encountered and Solved:**

The three problems we encountered during the lab were with interrupts, the switch, and the 7-segment displays.

The first of these issues dealt with the way interrupts are handled. The software was originally written to reset a loop index whenever the timer interrupt triggered, but for whatever reason this did not work and the index would ignore the reset. The solution was to use the ISR to set a timer flag, and perform all of our index resets via the standard code itself rather than the ISR.

The problem we had with the switch was that it was hypersensitive and would often times false trigger despite not being pressed. This was caused by the lack of pulldown resistor, which was causing the digital input pin to float. The solution was to add a 10K pulldown resistor to the digital pin which is the typical configuration and was notably missing from the ELEGOO directions.

The four digit 7-segment also presented some challenges. The first of these had to deal with the way it was wired; per the ELEGOO directions, all four cathodes for the display are wired through resistors to ground. For actually using the display, this pinout is incorrect as there is no way to change digits individually. To fix this, the cathodes for the middle two digits were connected via resistors to two digital pins of the Arduino. The outside digits of the display were left disconnected so that they are disabled. The code driving the software also presented issues, as digits were getting scrambled and writing nonsense. This ended up being caused by the digitalWrite() calls being backwards, turning the digits off when we wanted them on, and on when we wanted them off. Once these were flipped back to the way they were supposed to be the display worked like a charm.

1. **Personal Contribution to the Lab (Technical Details):**

Cody drew the schematic. Joe created and debugged the Arduino code that the board ran and modified the original pinouts Cody created to better fit the Arduino Nano used for the demonstration. Elizabeth created the module, documented the process, and wrote the report.

1. **Lessons Learned:**

In the first lab, we became familiar with how to use the Arduino to mimic a traffic light and used a 7-segment display to show the time remaining before changing LEDs. Applications of interrupts were learned during this lab. The lab also taught us how to use shift registers in real use applications. It taught us how to properly create a lab from the beginning brainstorming steps, all the way to execution.