SANTA CLARA UNIVERSITY	Mechatronics 2024	Andy Wolfe
Lab #5 – Color Sensing		

I. Objectives

- Measure color using a color sensor
- Experiment with the I2C interface
- Build a high-speed color sensor
- Map color coordinates into a color space to distinguish colors

II. Pre-Lab

Preparation:

- Review:
  - TCS3472 color sensor data sheet
  - Adafruit TCS34725 circuit schematic
  - LED data sheets for the CREE C4SMM-GJF-CT44Q7C2, C4SMA-RGF-CT34QBB2, and C5SMF-BJF-CR0U0351
  - Data sheet for the Vishay TEPT5700 phototransistor (Ambient Light Sensor)
- Review the build instructions for the DIY RGB color sensor.
- Read the lab instructions and think about algorithms you might use for the last demo.
- Bring something that can hold the sensors a fixed distance of somewhere between 2 and 10cm. from a piece of paper on the table. I will have tape, Velcro, and wire ties.

Pre-Lab Report:

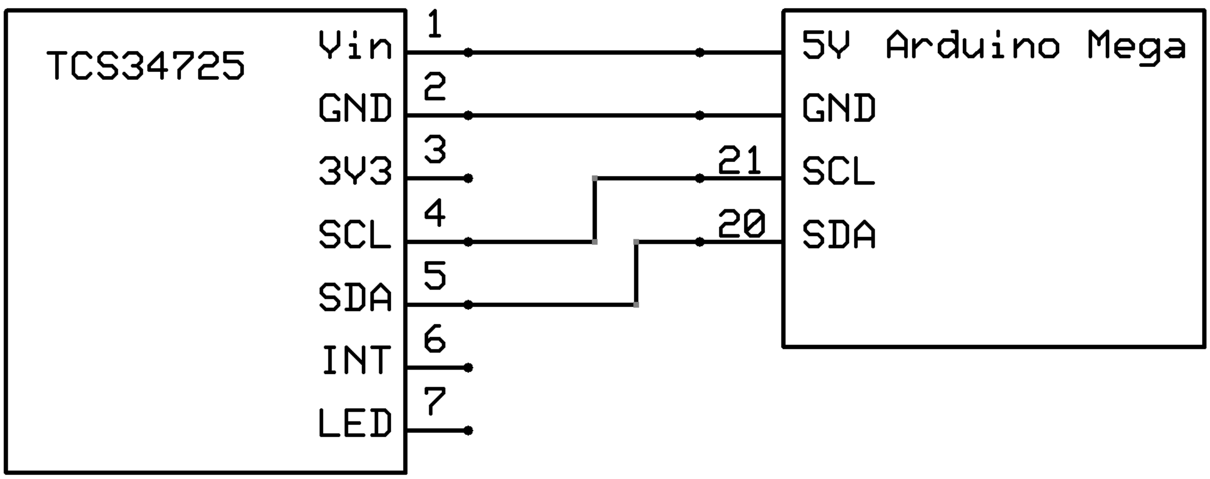
- Determine the range of A/D converter integration times that can be configured on the TCS3472.
- Determine the default A/D converter integration time that can be configured on the TCS3472.
- Determine the typical forward voltage drop at 20mA for the C5SMF-BJF-CR0U0351
- Determine the typical luminous intensity at 20mA for the C5SMF-BJF-CR0U0351
- Determine the typical forward voltage drop at 15mA for the C4SMA-RGF-CT34QBB2
- Determine the typical luminous intensity at 15mA for the C4SMA-RGF-CT34QBB2
- Determine what color light the Vishay TEPT5700 is most sensitive to.
- Determine the relative sensitivity of the Vishay TEPT5700 in response to the C5SMF-BJF-CR0U0351 as compared to the C4SMA-RGF-CT34QBB2.
- Include a selfie from your planning meeting of your group members along with a rainbow-colored picture from Jun 26, 2015.

III. Lab Procedure

Experiment 1: **TCS3472**

1. Find your TCS3472 RGB sensor.

2. Connect the following pins from the sensor to the Arduino (with the power off).



3. **Do not connect the 3V3 pin on the sensor!**

4. Copy the TCS3472 code library (Adafruit_TCS34725-1.4.2.zip) from the Google Drive.

5. In the Arduino IDE, go to Sketch\Include Library\Add .ZIP Library and select the Adafruit_TCS34725-1.4.2.zip file to install. (You may be able to install directly with the library manager as well)

6. Go to File\Examples\Adafruit TCS34725\ and open the demo program tcs34725. **Save the demo program under a new name**.

7. Run that program on your Arduino and make sure you are seeing data on the serial monitor.

8. Mount the sensor so that it is facing downwards 2-10cm. from the table. Place a piece of white paper under the sensor. If the sensor is not physically stable, your readings will be all over the place.

9. Take 10 consistent readings from the white paper. Record all of the data. (You can cut and paste from the Serial Monitor. You can stop scrolling to keep the data on the Serial Monitor screen.) Using the data from the Clear channel – determine the average reading value and also the noise level as a percentage of the average reading value. Include the data and calculations in your report. Also calculate the average readings from the R, G, and B channels. Demo 1 – show that this is working.

10. Replace the white paper with black. Take 10 consistent readings from the black paper. Calculate the average readings from the C, R, G, and B channels.

11. Test Red, Green, and Blue paper from Test Sample Set A and record important lab observations that may help in future use of this sensor.

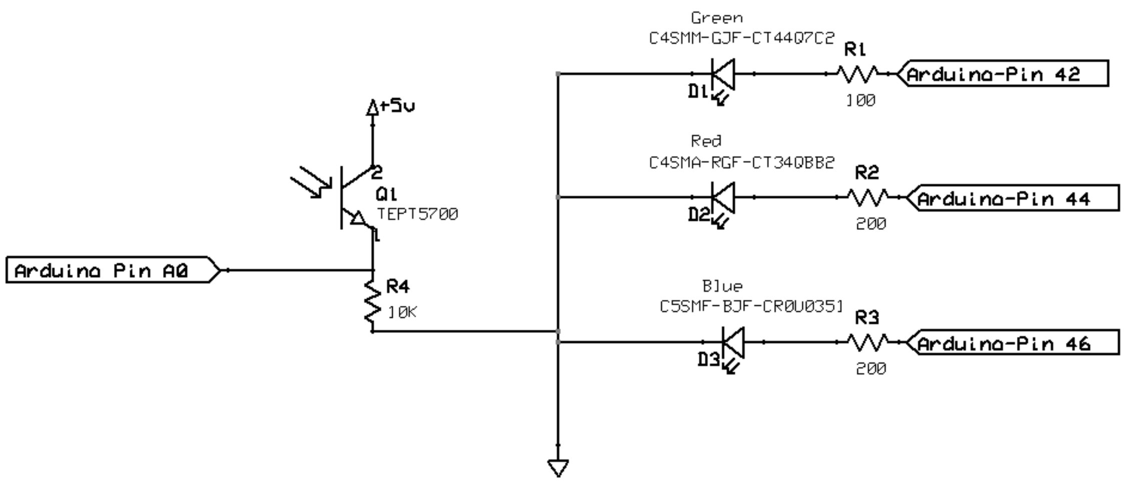
12. Modify the program to calculate and report the amount of time it takes to perform one iteration of loop(); If you have added anything to loop() other than timing measurement, comment it out first.

13. **Save the demo program under a new name**. Modify the demo program to set the measurement time to 24ms and the measurement gain to 16X. (Hint: Find the source code for Adafruit_TCS34725.h on your computer.) Record any observations as to how this changes the behavior and accuracy of the sensor.

Experiment 2: **Build Your Own RGB Sensor**

1. Build the DIY RGB sensor according to the directions in the separate handout. I will supply the LEDs and Phototransistor in the lab. They are different from the ones in your kit. Don't mix them up.

2. Connect the sensor using your proto-board and Arduino as shown below. Look at the resistor values carefully.



3. Load up the sketch "LED-response" from the lab Google folder. This sketch turns on the blue LED, waits a specific number of µs, then samples the phototransistor. It then turns off the LED for 100 ms. Depending on the delay, the results differ. It seems to take some time for the LED to reach full power and/or the phototransistor to reach full current output. Record the data and determine how long you should delay in order to get 90% of the maximum response and how long in order to get 99% of maximum response.

4. Modify the program to perform the same test for the Red and Green LEDs. Record the results and make the same determinations.

5. Write a program that every 100ms:

    a. Turns on the Red LED, waits until at least the 90% response time[1]; reads the phototransistor voltage level 4 times and adds the 4 results to produce a value r; turns off the red LED for at least 1ms.

[1] you can wait a fixed time based on your prior experiments.

b. Turns on the Green LED, waits until at least the 90% response time; reads the phototransistor voltage level 4 times and adds the 4 results to produce a value g; turns off the green LED for at least 1ms.

c. Turns on the Blue LED, waits until at least the 90% response time; reads the phototransistor voltage level 4 times and adds the 4 results to produce a value b; turns off the blue LED for at least 1ms.

d. Prints the values of r, g, and b to the Serial Monitor with labels.

6. Test and record the values for white paper, black paper, and red, green, and blue paper from Test Sample Set A. Demo 2 – show this program running.

7. Add a red, green, blue, and white LED to your design (from your kit). Connect each one to an Arduino digital output on one side and through a 330Ω resistor to ground on the other.

8. Create a discrimination function that determines if a paper sheet is:

   a. Red

   b. Green

   c. Blue

   d. Black

   e. White

   And turns on the corresponding LED to indicate which. (No LED for black)

9. Get samples from Test Sample Set B and test them on your program. They should provide correct answers. Demo 3 – show this program running.