# REAL-TIME EMBEDDED SYSTEM
## PROJECT DESIGN

# Dual-Mode Real-Time Cap Sorting System Using Arduino Uno

Name: NAVARRO, ROD GERYK C.

Course/Section: CPE161P-4/C1

Progress Report Number: 2

Date of Performance: 02/19/2025
Date of Submission: 02/24/2025

CYREL O. MANLISES, PH.D.
Instructor

# DISCUSSION

## Part I. Discuss in detail your progress here.

*This week progress, I reorganized and reconnected the components to make the circuit cleaner and more organized. I also received new components, including a transparent glass protector and a stand for the LCD, as well as a 4x4 keypad with labels, which I immediately integrated into the circuit. My main focus this week was learning how to use the TCS3200 color sensor and successfully integrating it into my system. In my last progress report, I had only set up the connections and created two modes: automatic and manual. Now, with the color sensor, I added a delay to the servo motors to ensure the sensor accurately detects the color of the bottle caps. When the top servo motor reaches a 90-degree angle, the sensor gets enough time to determine the cap's color before the servo moves it into position. After identifying the color, the servo transfers the cap to a second servo, which routes it to the correct container based on its color. I also added a monitoring feature that tracks the number of caps sorted into red, green, and blue categories. The LCD displays the count of each color in real-time, allowing users to easily monitor the sorting process. Additionally, I implemented a function that lets users return to the main menu by pressing the letter "D" in any mode, allowing them to select a different mode as needed. For the mechanism of the system I used a cup cake stand with 3 layers.*

*During this progress, I encountered several coding errors, with the most challenging issue being the timing of color detection. The delays in both servo motors initially caused conflicts with the color sensor, leading to incorrect readings. To fix this, I separated the movement of the first servo which picks up the caps from the movement of the second servo which sorts them. Now, the color sensor stops at the 90-degree position to accurately classify the cap's color in real time before the second servo moves it to the correct container.*

*Initializing Variables and Setting Up Components*

*In the beginning, I defined the necessary variables and constants to control the system efficiently. The LCD display was initialized using the LiquidCrystal_I2C library, allowing real-time monitoring of sorting progress. Pins were assigned to the TCS3200 color sensor to read color frequencies, and the keypad was mapped to allow user inputs for different modes. I also defined the servo motor pins and initialized control variables, such as automaticMode and modeSelected, to manage the operation flow. The variables for color detection, including redFrequency, greenFrequency, and blueFrequency, were set to store the detected values, while counters (rc, gc, bc) were used to keep track of sorted bottle caps. The setup() function was responsible for preparing the system before it started running. Serial communication was initiated to enable debugging, and the LCD displayed an initialization message to inform the user that the system was starting. The color sensor's frequency scaling was set to 20% to ensure accurate readings. I also configured the servos as output devices and mapped the keypad for user interactions. At the end of the setup process, the LCD prompted the user to select between automatic and manual mode, ensuring the system only started sorting once a mode was chosen. The loop() function continuously checked for keypad inputs and controlled the sorting process. If the user pressed 'A,' the system switched to automatic mode, while 'B' activated manual mode. Pressing 'D' reset the mode selection. The function then activated the color sensor by switching between different photodiodes to detect red, green, and blue frequencies. Based on the detected color, the servos moved the cap to the corresponding container. In automatic mode, the servos moved automatically after detection, while in manual mode, the user could control the servos manually using specific keys. This structured approach ensured that the system functioned smoothly and minimized errors in the sorting process.*

Part II. Provide screenshots as proof of your progress.

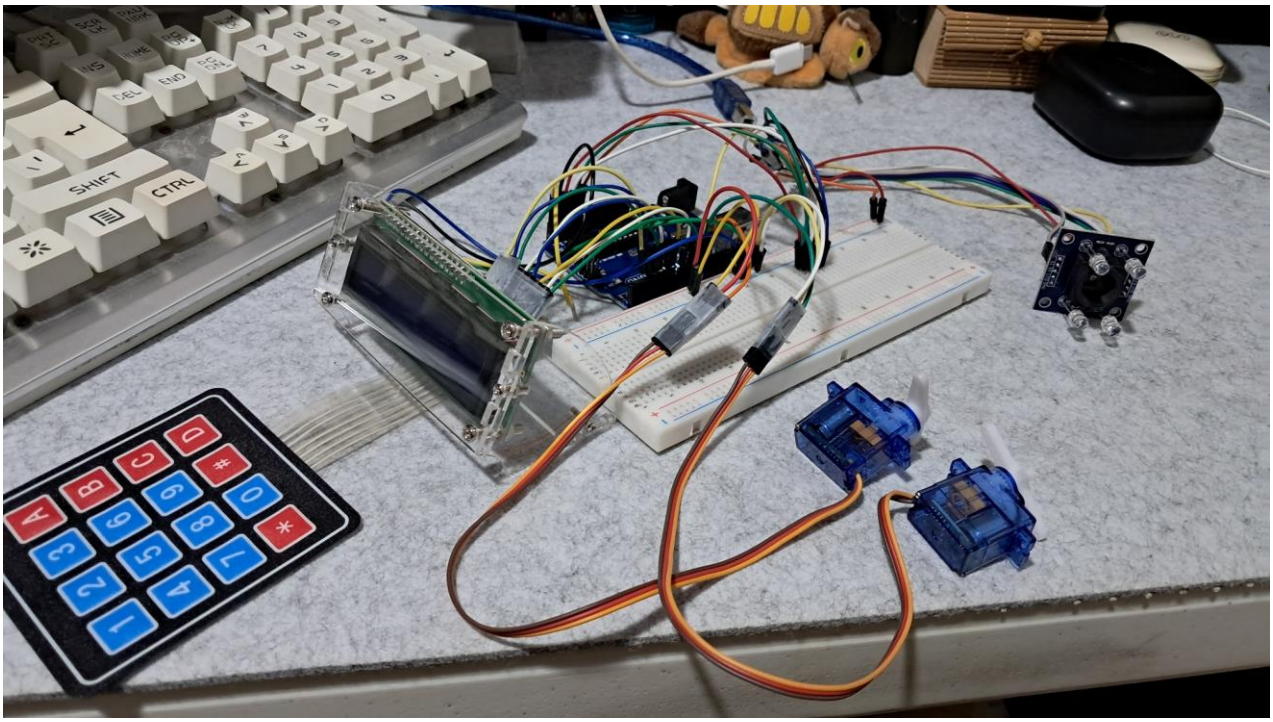


*Figure 1: The Connection of the 4x4 Keypad with Labels, Servo Motors, TCS3200 Color Sensor, and the LCD*

Navarro_ProgressReport2_CPE161P.ino

```
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27, 16, 2);

#define S0 A3
#define S1 A2
#define S2 A1
#define S3 A0
#define sensorOut 10

int redFrequency = 0;
int greenFrequency = 0;
int blueFrequency = 0;
int MIN_VALUE = 50;
int MAX_VALUE = 200;
int rc = 0;
int gc = 0;
int bc = 0;
int ic = 0;

const byte rows = 4;
const byte cols = 4;
char keys[rows][cols] = {
  {'D', '*', '0', '#'},
  {'C', '9', '8', '7'},
  {'B', '6', '5', '4'},
  {'A', '3', '2', '1'}
};

byte rowPins[rows] = {6,7,8,9};
byte colPins[cols] = {2,3,4,5};

// Servo setup
```

*Figure 2: The Initialization of Variables and Components*

Navarro_ProgressReport2_CPE161P.ino

```
50  void setup() {
51    Serial.begin(9600);
52    Serial.println("System Starting");
53
54    pinMode(S0, OUTPUT);
55    pinMode(S1, OUTPUT);
56    pinMode(S2, OUTPUT);
57    pinMode(S3, OUTPUT);
58    pinMode(sensorOut, INPUT);
59
60    // setting frequency scaling to 20%
61    digitalWrite(S0, HIGH);
62    digitalWrite(S1, LOW);
63
64    lcd.init();
65    lcd.backlight();
66    lcd.clear();
67    lcd.setCursor(0, 0);
68    lcd.print(" Initializing");
69    lcd.setCursor(0, 1);
70    lcd.print("System On");
71
72    for (int a = 12; a < 15; a++) {
73      lcd.setCursor(a, 1);
74      lcd.print(".");
75      delay(700);
76    }
77    lcd.clear();
78
79    pinMode(servoPin1, OUTPUT);
80    pinMode(servoPin2, OUTPUT);
81
82    for (byte i = 0; i < rows; i++) {
```

Navarro_ProgressReport2_CPE161P.ino

```
110 void displayColor(int r, int g, int b, char key) {
111
112   if (!modeSelected) return; // ensure mode is selected before displaying color
113
114   if ((r > MAX_VALUE && g > MAX_VALUE && b > MAX_VALUE) || (r < MIN_VALUE && g < MIN_VALUE && b < MIN_VALUE)) {
115     lcd.setCursor(0, 0);
116     lcd.print("Color Not Found");
117     Serial.println("Color Not Found");
118
119   } else {
120     lcd.setCursor(0, 0);
121     if (r < g && r < b) {
122       lcd.print("Color: Red     ");
123       Serial.println("Color: Red ");
124       if(automaticMode){
125         ic++;
126        if (servoPin1, 90 && ic >= 2) {
127             rc = rc + 1;
128             lcd.setCursor(0, 1);
129             lcd.print("RC:");
130             lcd.print(rc);
131        }
132
133       moveServo(servoPin2, 140);
134       delay(1000);
135       moveServo(servoPin1, 0);
136       delay(1000);
137       }
138     } else if (g < r && g < b) {
139       lcd.print("Color: Green     ");
140       Serial.println("Color: Green");
141       if(automaticMode){
142       ic++;
```

*Figure 3: The Setup of the Components*

Navarro_ProgressReport2_CPE161P.ino

```
203
204 void loop() {
205   char key = getKey();
206
207   if (key) {
208     if (key == 'A') {
209       automaticMode = true;
210       modeSelected = true;
211       lcd.clear();
212       lcd.setCursor(0, 0);
213       lcd.print("Mode: Automatic");
214       delay(1000);
215       lcd.clear();
216     } else if (key == 'B') {
217       automaticMode = false;
218       modeSelected = true;
219       lcd.clear();
220       lcd.setCursor(0, 0);
221       lcd.print("Mode: Manual");
222       delay(1000);
223       lcd.clear();
224     }else if (key == 'D') {
225       // Reset mode selection and return to start screen
226       modeSelected = false;
227       lcd.clear();
228       lcd.setCursor(0, 0);
229       lcd.print("Select Mode: ");
230       lcd.setCursor(0, 1);
231       lcd.print("A:Auto B:Manual");
232       return; // Exit the loop iteration to wait for user input
233     }
234   }
235
```

*Figure 4: Loop Function*

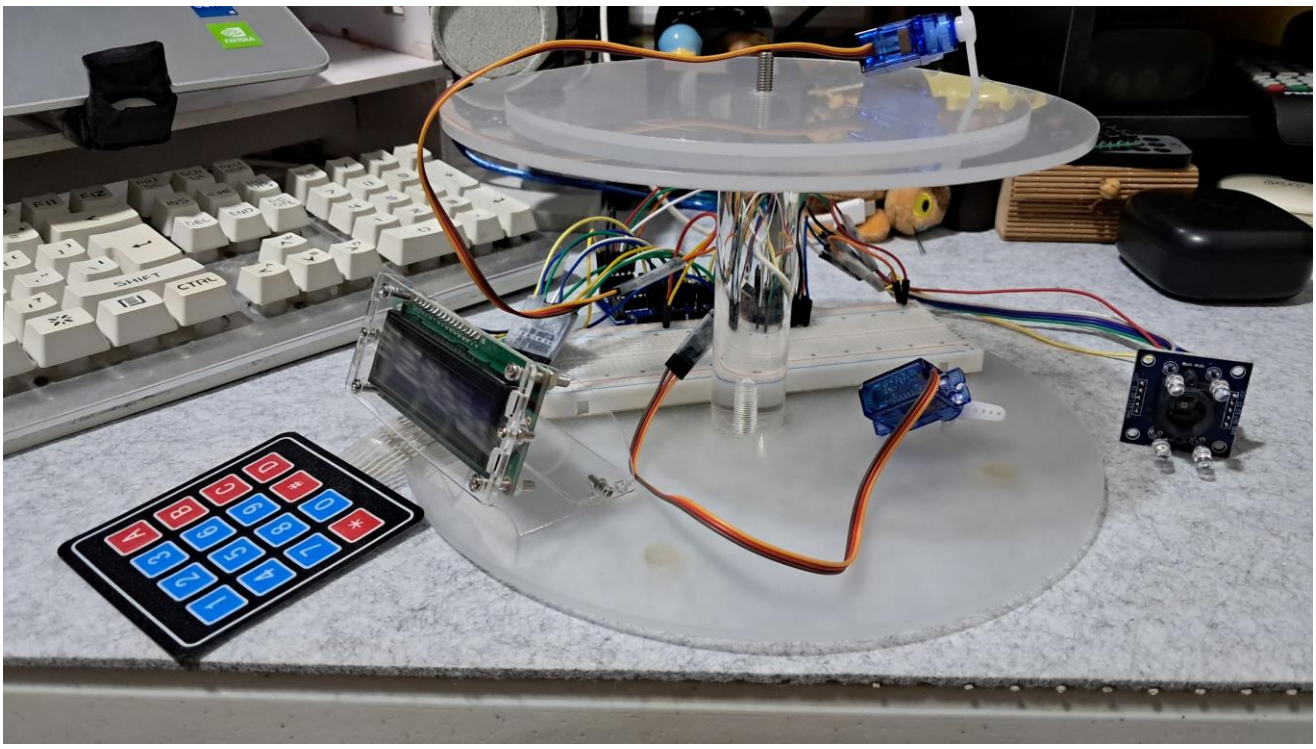

**Figure 5:** *The Circuit with the Cup Cake Stand for the Mechanism of the System*

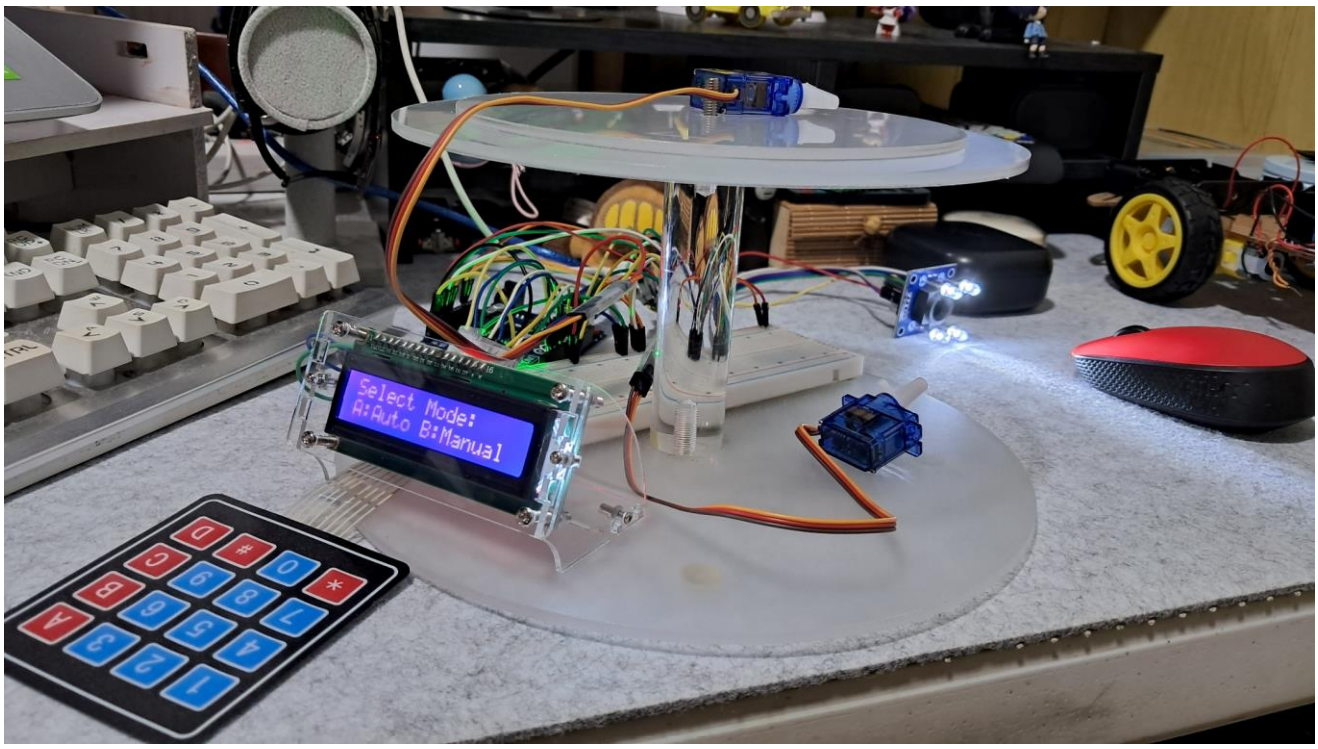


**Figure 6:** *Working Prototype*

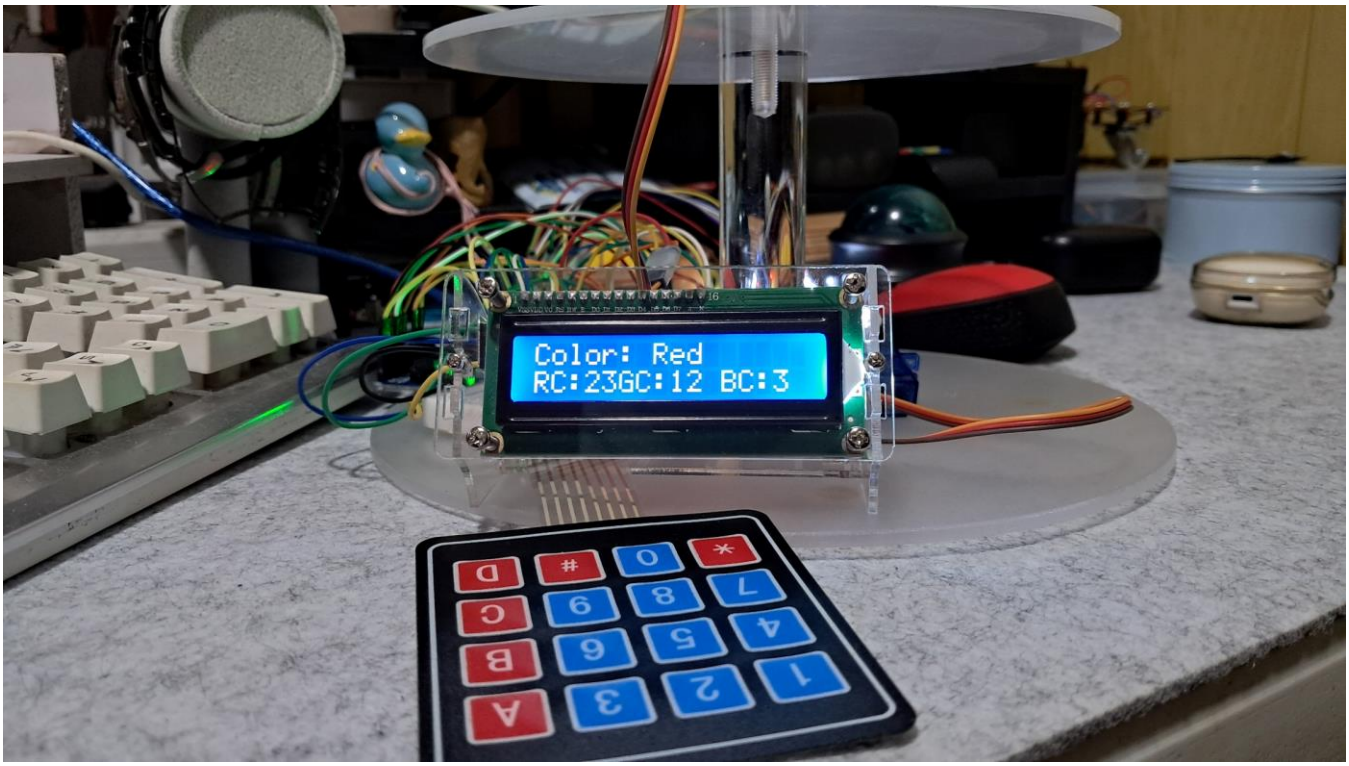


*Figure 6. 1: Automatic Mode*

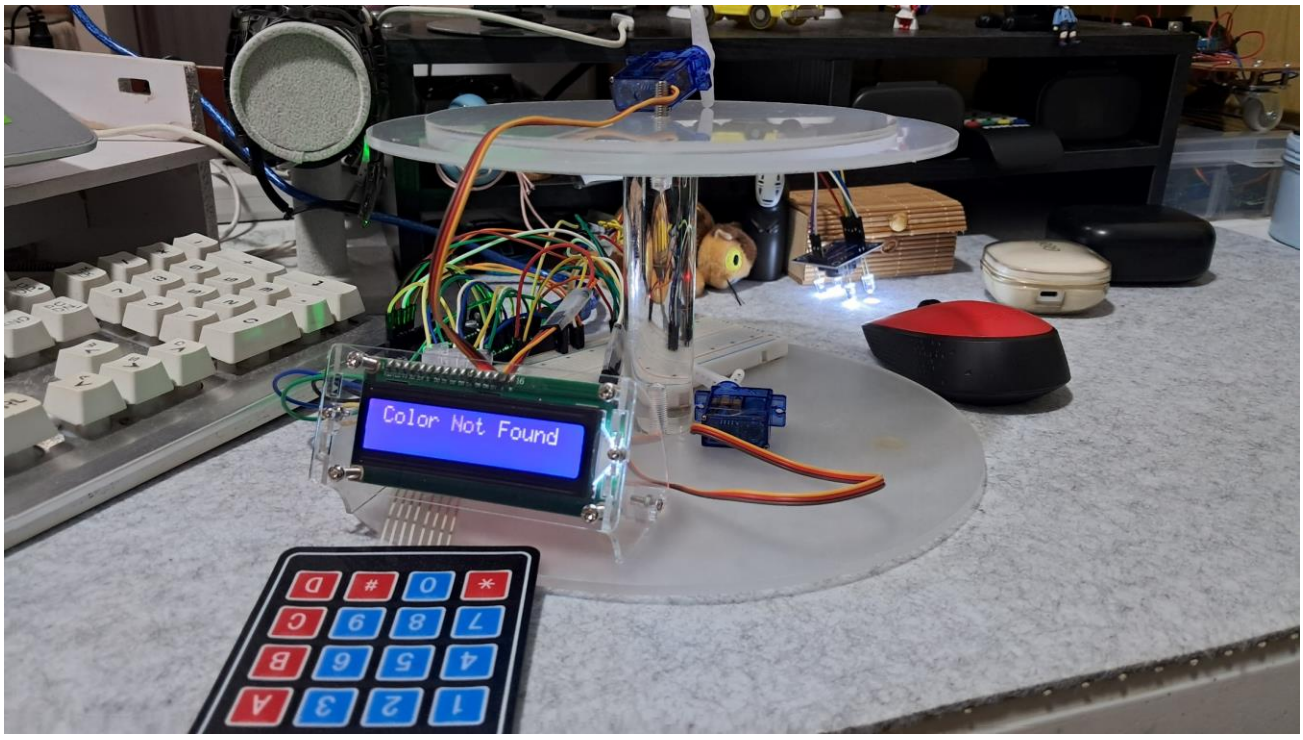


*Figure 6. 2: Manual Mode*

## Part III. List down your work load

- *Rearranged the connections to make the circuit cleaner and well organized.*
- *Created a counter that counts the number of caps in each color which is displayed in the LCD in real-time.*
- *Successfully integrate the color sensor in the system to determine the color of the caps.*
- *Created a function to return to the selection of modes.*
- *Configured the movement of the servo motors.*
- *Listed the pin connections for each component.*
- *Fix any issues in the connections.*

## Part IV. Gantt chart

| **Tasks** | **02/17/2025** | | **02/23/2025** | | **03/2/2025** | | **03/9/2025** | |
|---|---|---|---|---|---|---|---|---|
| Connection of the components and the creation of the two modes | | | | | | | | |
| Integrate the color sensor in the system, as well as modifying the code with the color sensor. | | | | | | | | |
| Create the mechanism for the whole system. | | | | | | | | |
| Debugging | | | | | | | | |
| Polishing the working prototype. | | | | | | | | |
| Defense/Presentation | | | | | | | | |