**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.

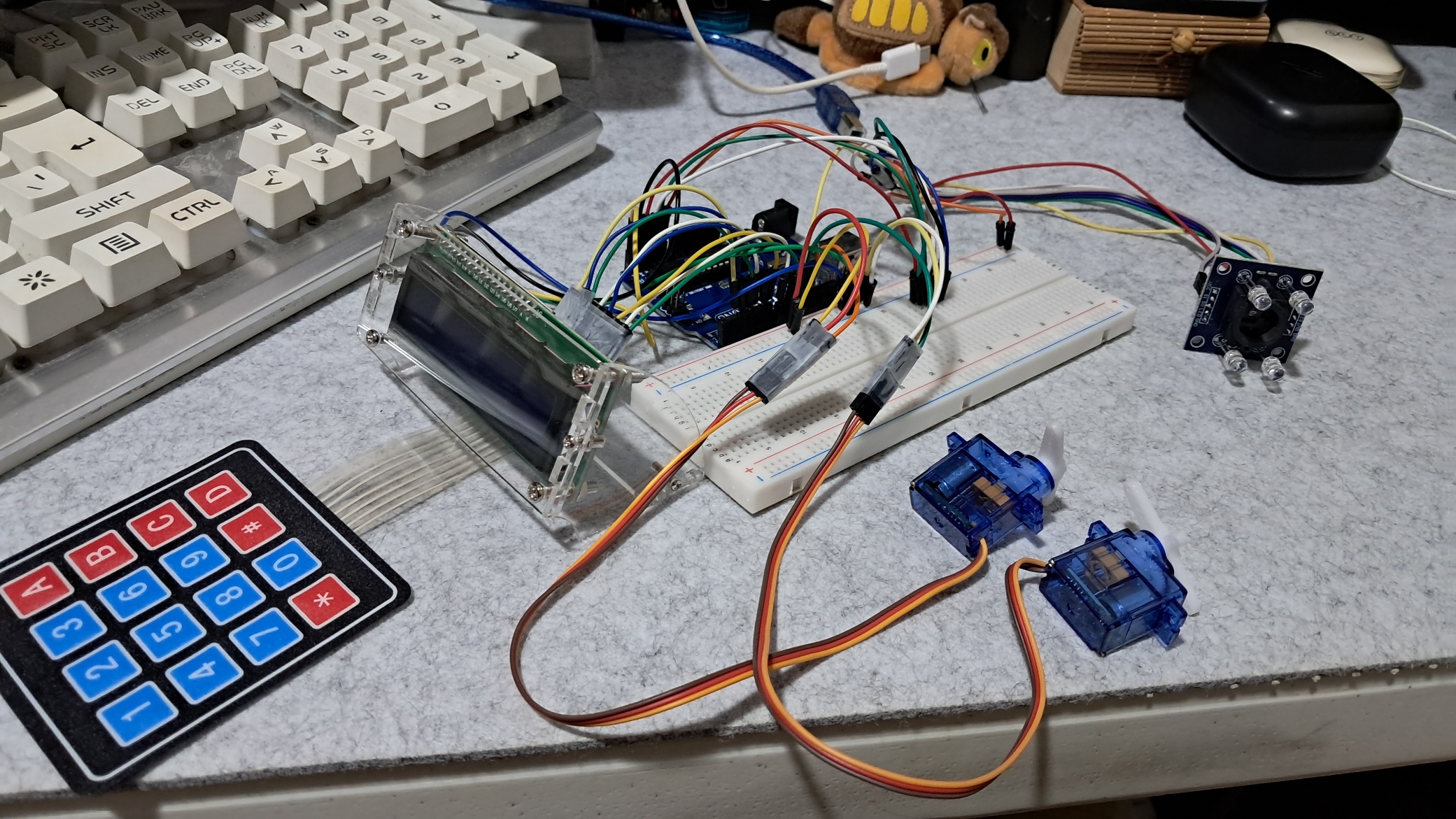
This week progress, I reorganized and reconnected the components to make the circuit cleaner and more organized. I also received new components, including a transparent glass protector and a stand for the LCD, as well as a 4x4 keypad with labels, which I immediately integrated into the circuit. My main 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.

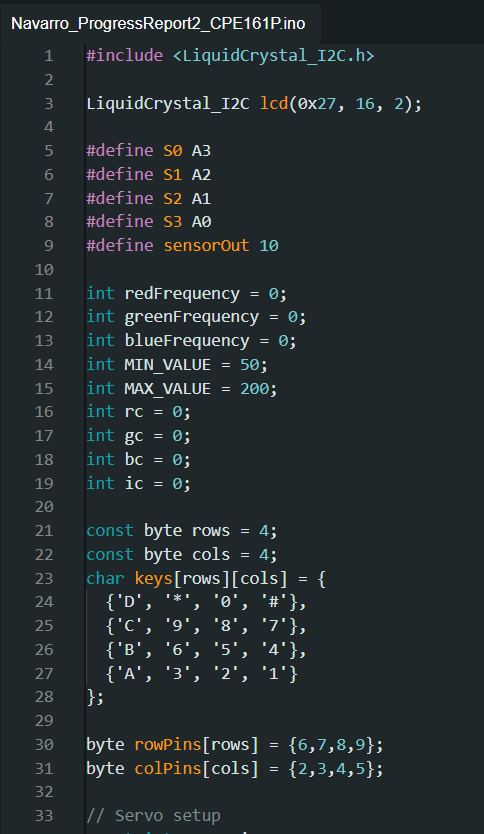
**Initializing Variables and Setting Up Components**

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



**Figure 2:**  The Initialization of Variables and Components

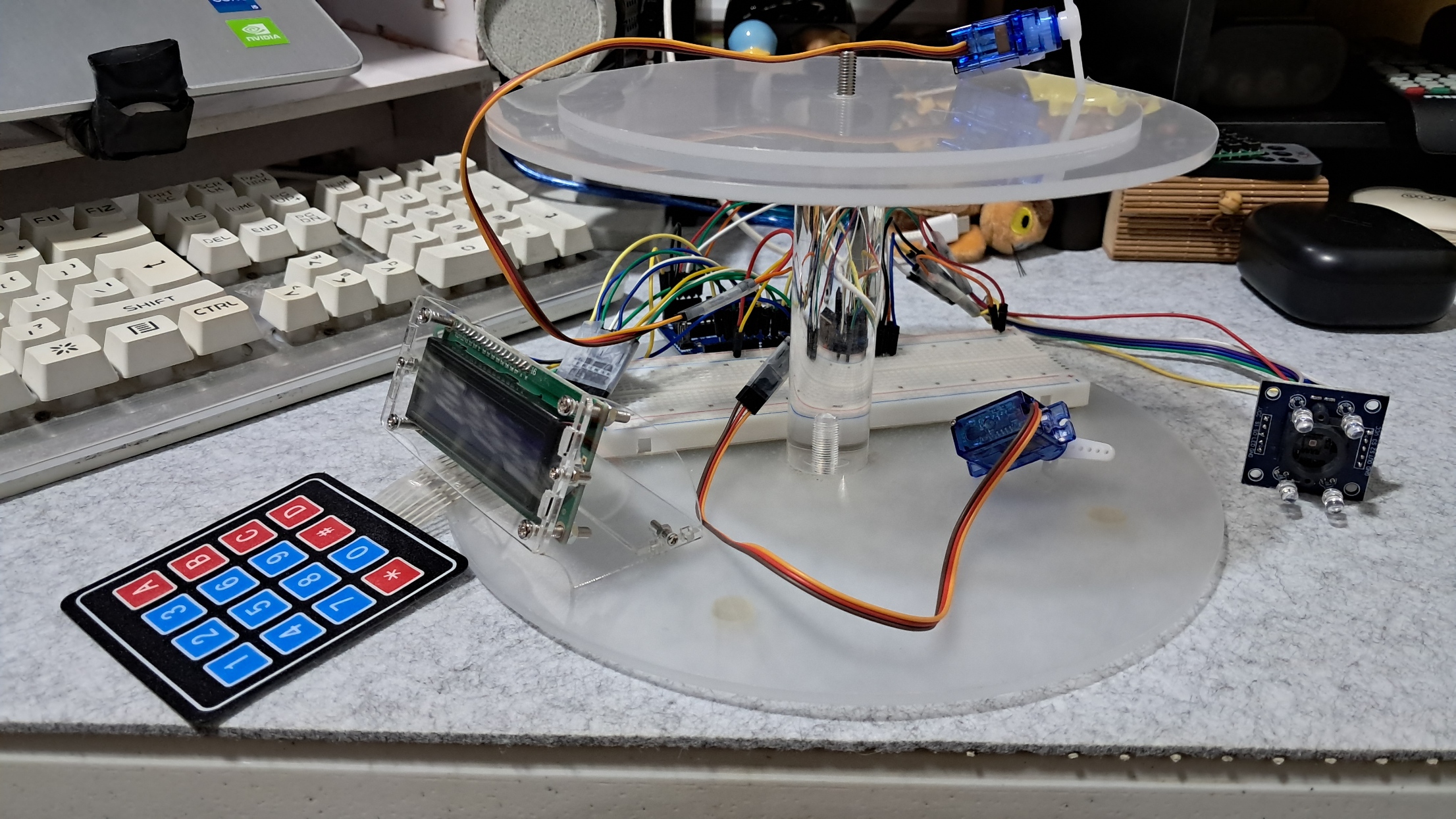
****

****

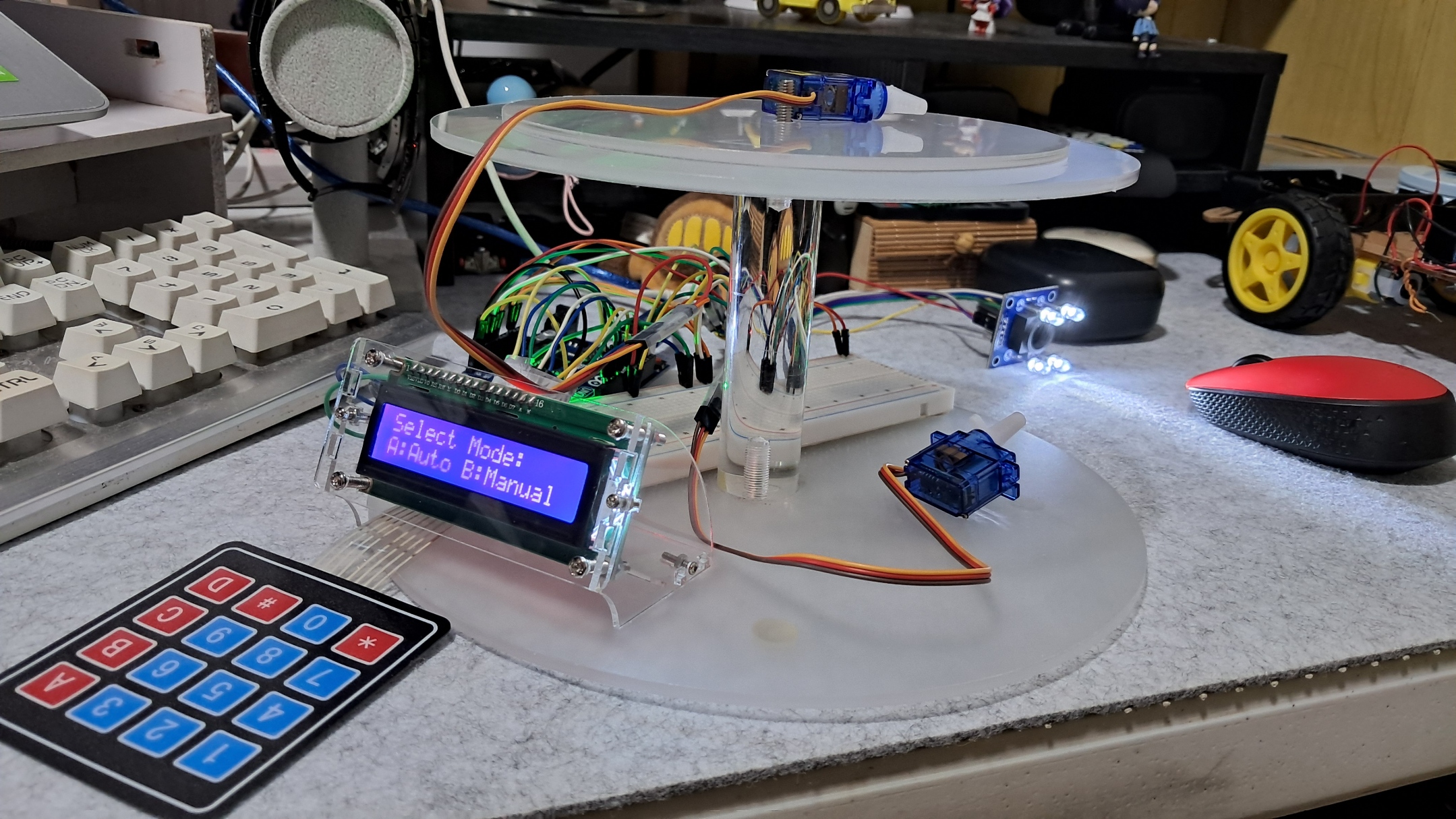
**Figure 3:**  The Setup of the Components

****

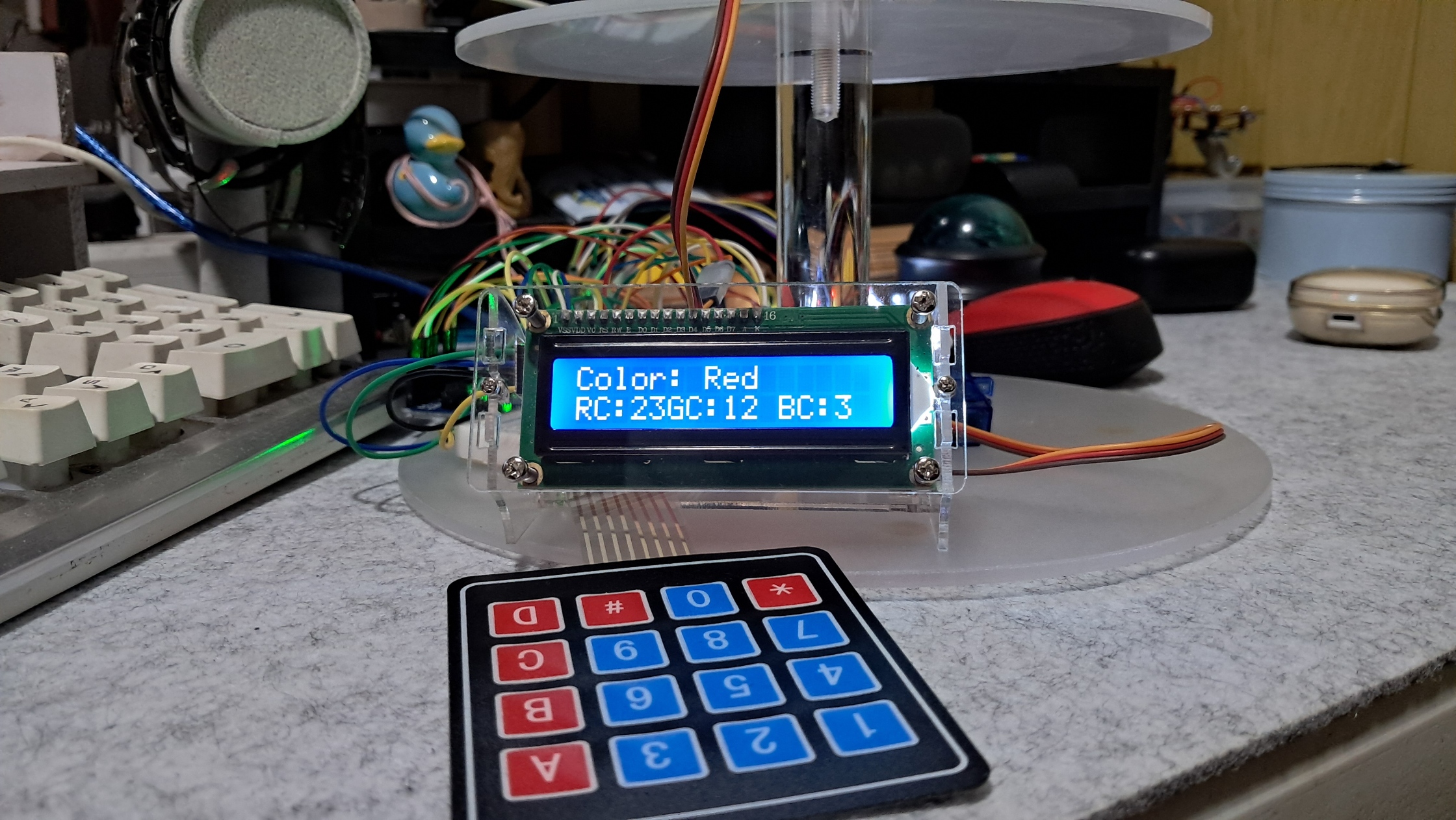
**Figure 4:**  Loop Function



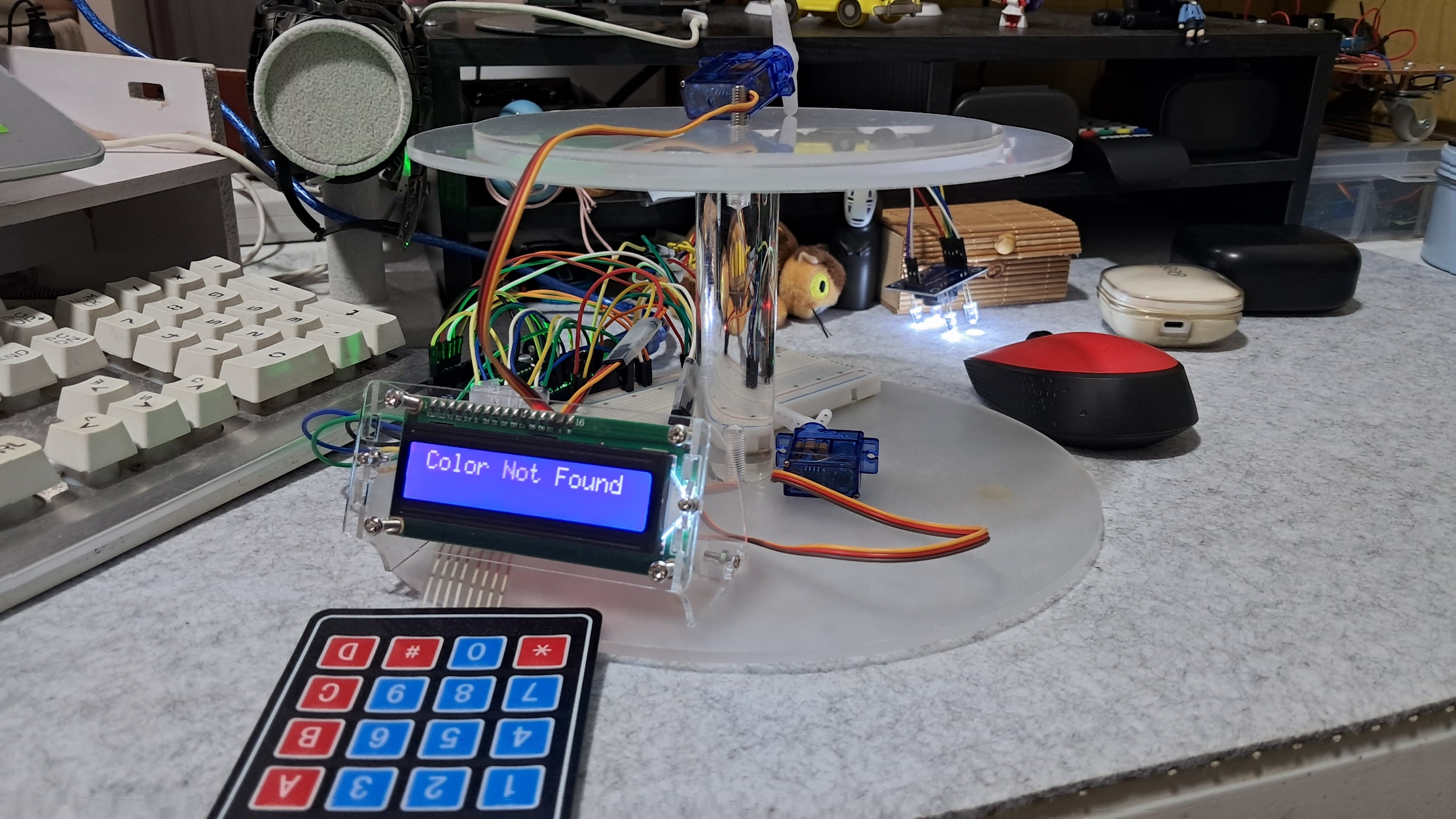
**Figure 5:**  The Circuit with the Cup Cake Stand for the Mechanism of the System



**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.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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 |  |  |  |  |  |  |  |  |

Part IV. Gantt chart