PAMANTASAN NG LUNGSOD NG MAYNILA
(University of the City of Manila)
Intramuros, Manila

Microprocessors (Laboratory)

Laboratory Activity No. 7

DC Motor and Temperature Sensor

Score

Submitted by:
Leader: Palacio, Leticia Mae
Baldanzo, Raizza Marrie C.
Baltes, Billy Renz C.
Belmonte, Jhade Loui M.
Magnabihon, Michael Lorenz M.
Tiu. Joshua Miguel Yaj A.

S 10:00am-1:00pm / CPE 0412.1-1

Submitted to:

Engr. Maria Rizette H. Sayo

Date Submitted:

09/12/2023

I. Objectives

This laboratory activity aims to implement the principles and techniques of hardware programming using Arduino through:

- Creating a ventilation system that senses room temperature. If temperature exceeds 25C, the system buzzes and turns on a dc motor simulating ventilation until it goes back to 25C. If the temperature is lower than 25C, the buzzer and motor is then turned off.
- Improve the ventilation system by adding status indicators. Ensure that when the temperature is below 26.0°C, the motor, buzzer, and red LED are deactivated. Activate the green LED for normal temperature. Validate with the provided Arduino code for synchronized control.

II. Methods

- Perform the tasks and problems presented in the presentation.
- Present a unique implementation representative of the objectives.

III. Results

To create the ventilation system, an Arduino development board is needed which controls the whole system, an h-bridge motor driver which drives and controls the dc motor, a dc motor, a 9v battery, a temperature sensor and a piezo buzzer.

Name	Quantity	Component
U1	1	H-bridge Motor Driver
U2	1	Arduino Uno R3
M1	1	DC Motor
BAT1	1	9V Battery
U3	1	Temperature Sensor [TMP36]
PIEZO1	1	Piezo

Table 1. List of Components for the Ventilation System

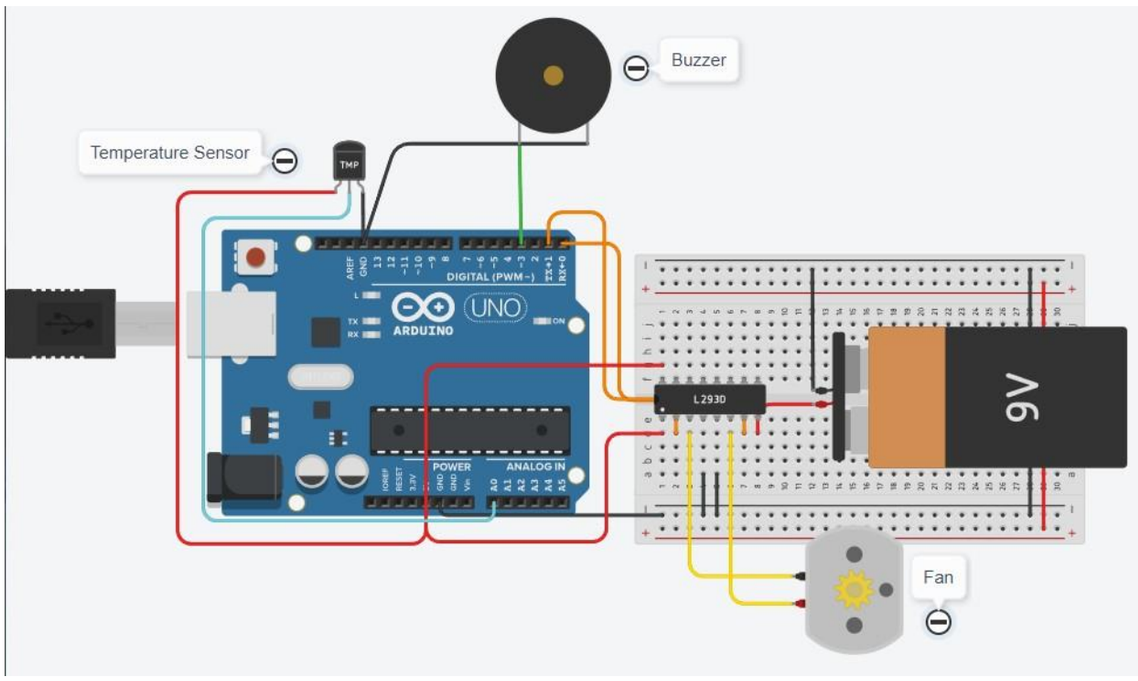

Figure 1. Circuit Diagram of the Ventilation System

The circuit diagram is made by connecting a DC motor which represents the fan into an H-bridge motor driver IC (L293D) which controls how the motor works. The input of the IC is connected as output of the Arduino which allow the Arduino to technically control the DC motor. The buzzer and temperature sensor are also connected to the Arduino. The buzzer serves as the output or indicator of exceeding temperatures while the sensor serves as the input of the whole system measuring the room temperature and gives data to the Arduino to decide how the system should react.

```
int motor1 = 0; //motor
int motor2 = 1; //motor
int sensor = A0; //temp sensor
int buzzer = 3; //buzzer

void setup()
{
    pinMode(motor1, OUTPUT);
    pinMode(motor2, OUTPUT);
    pinMode(sensor, INPUT);
    pinMode(buzzer, OUTPUT);
}

void motorOn()
{
  digitalWrite(motor1, HIGH); //rotate clockwise
  digitalWrite(motor2, LOW); //rotate clockwise
}

void motorOff()
{
  digitalWrite(motor1, LOW);
  digitalWrite(motor2, LOW);
}

void loop()
{
  int tmpValue = analogRead (sensor); //sensor analog value
  int temp = map(tmpValue, 20, 358, -40, 125); //converted analog value to temp

  if (temp > 25) //turn on motor and buzzer
  {
    motorOn();
    digitalWrite(buzzer, HIGH);
  }
  else if (temp == 25) //turn off motor but turn on buzzer
  {
    motorOff();
    digitalWrite(buzzer, HIGH);
  }
  else //turn off motor and buzzer (temp below 25)
  {
    motorOff();
    digitalWrite(buzzer, LOW);
  }
}
```

Figure 2. First Part of the Program Code of the Ventilation System

The first part of the code declares different variables respective to the pins of the Arduino and initializing the components (DC motor, temperature sensor, and buzzer). 2 functions are then made which turns the motor on and off. The main part of the program reads the analog sensor value and converts that value into the temperature value which is used as basis for the different conditions that determine how the system works. If temperature is greater than 25, the motor is then turned on and the buzzer is also turned on. If the temperature is equal to 25, the motor is turned off while the buzzer is turned on. Lastly, if the temperature is less than 25, the motor and buzzer is turned off.

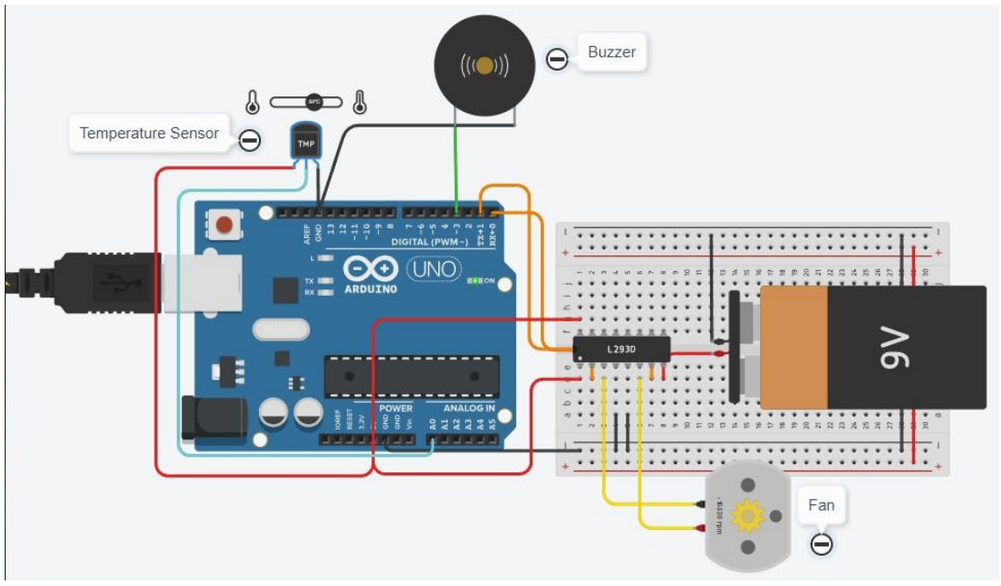

Figure 3. Simulation of the Ventilation System when Temperature is Greater than 25C

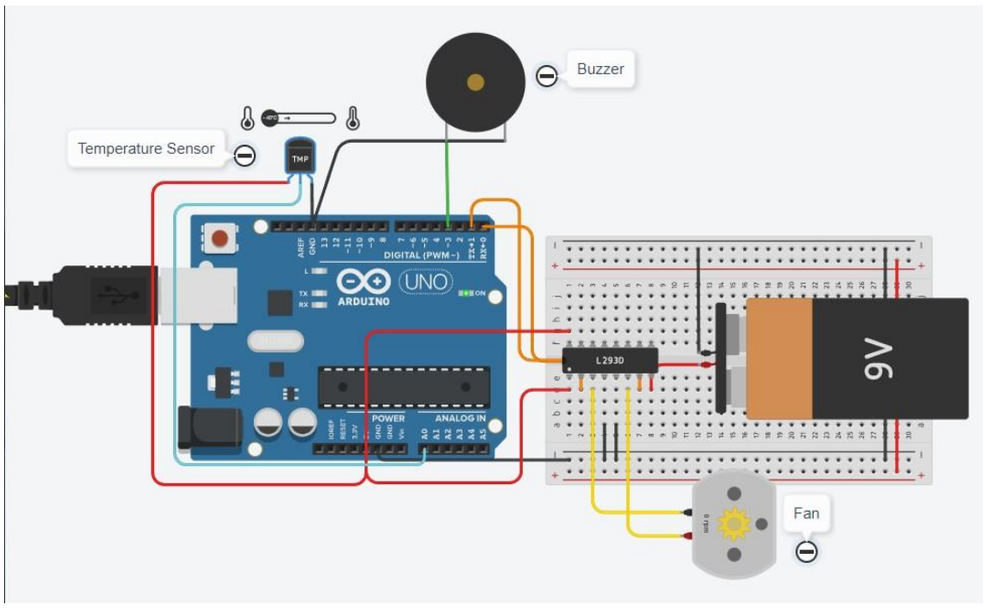

Figure 4. Simulation of the Ventilation System when Temperature is Less than 25C

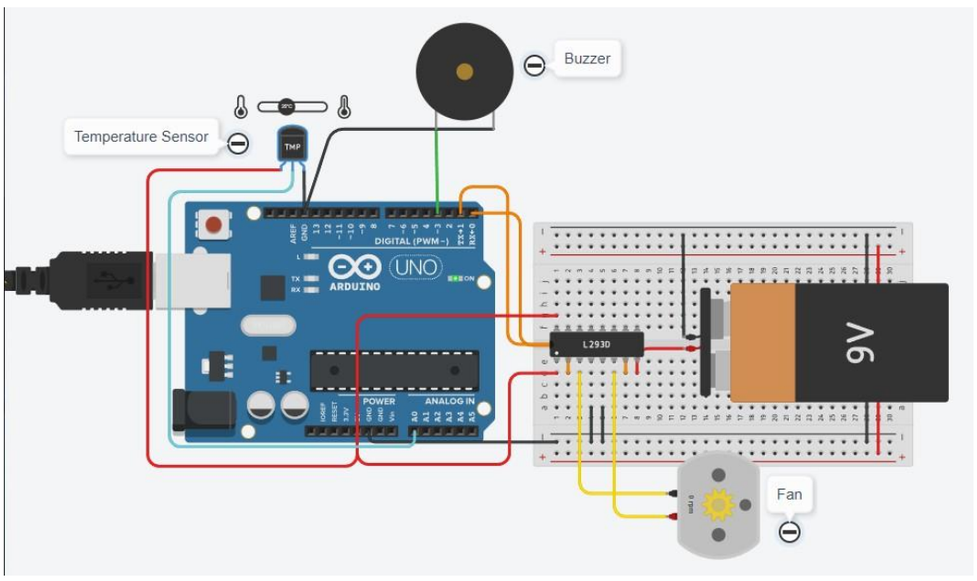

Figure 5. Simulation of the Ventilation System when Temperature is Equal than 25C

IV. Modification

Circuit Diagram:

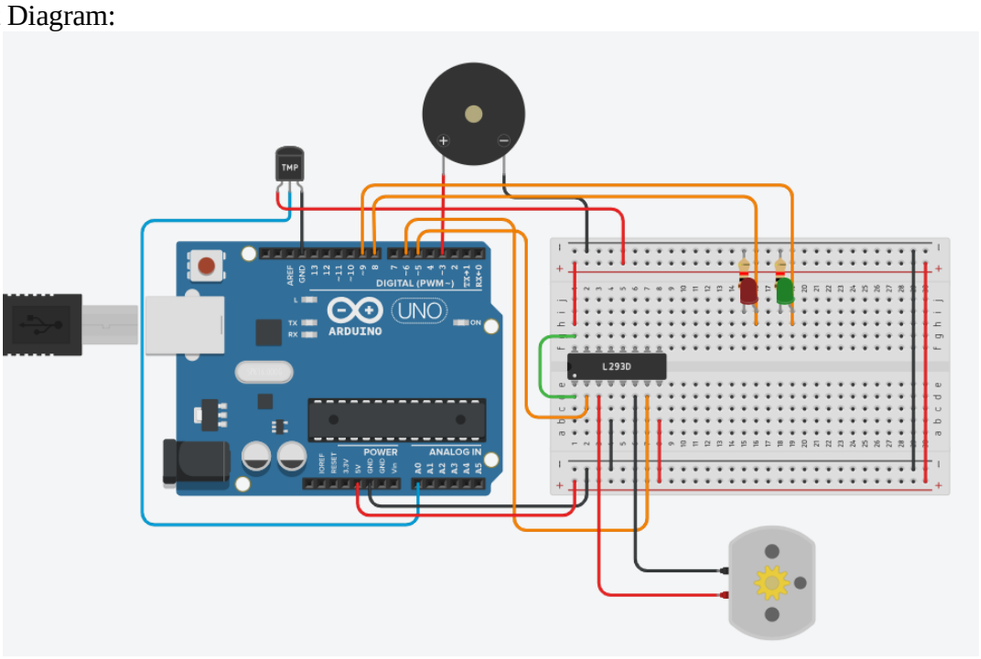

Figure 6. Circuit Diagram of Modified Ventilation System

Actual Implementation:

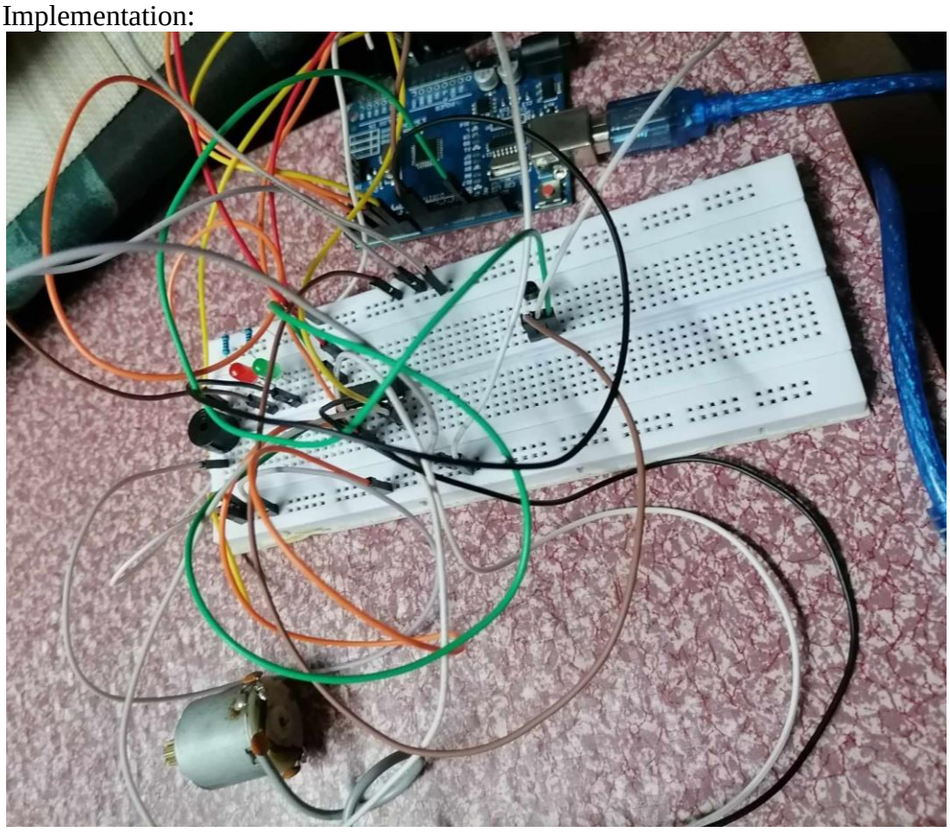

Figure 7. Actual Implementation of Modified Ventilation System

Modified Code:

```
// DC motor and temperature sensor (TMP36) using LEDs as indicator

// Pin configurations
const int motor1 = 5;           // Motor
const int motor2 = 6;           // Motor
const int sensorPin = A0;       // Temperature sensor
const int buzzerPin = 3;        // Buzzer
const int redLedPin = 8;        // Red LED indicator
```

```
const int greenLedPin = 9;      // Green LED indicator

// Constants
const float temperatureThreshold = 26.0; // Threshold temperature in Celsius

void setup() {
  // Pin mode setup
  pinMode(motor1, OUTPUT);
  pinMode(motor2, OUTPUT);
  pinMode(sensorPin, INPUT);
  pinMode(buzzerPin, OUTPUT);
  pinMode(redLedPin, OUTPUT);
  pinMode(greenLedPin, OUTPUT);

  // Serial communication setup
  Serial.begin(9600);
}

void motorOn() {
  // Turn on the motor (Rotate clockwise)
  digitalWrite(motor1, HIGH);
  digitalWrite(motor2, LOW);
}

void motorOff() {
  // Turn off the motor
  digitalWrite(motor1, LOW);
  digitalWrite(motor2, LOW);
}

void loop() {
  // Read sensor analog value
  int sensorValue = analogRead(sensorPin);

  // Convert sensor value to voltage
  float voltage = sensorValue * (5.0 / 1024.0);

  // Convert voltage to temperature using TMP36 formula
  float tempCelsius = (voltage - 0.5) * 100.0;

  // Print temperature to Serial Monitor
  Serial.print("Temperature in Celsius: ");
  Serial.println(tempCelsius);

  // Check if temperature exceeds the threshold
  if (tempCelsius > temperatureThreshold) {
    motorOn();
    digitalWrite(buzzerPin, HIGH);
    digitalWrite(redLedPin, HIGH);    // Red LED ON
    digitalWrite(greenLedPin, LOW);   // Green LED OFF
  } else {
    motorOff();
    digitalWrite(buzzerPin, LOW);
    digitalWrite(redLedPin, LOW);     // Red LED OFF
    digitalWrite(greenLedPin, HIGH);  // Green LED ON
  }
  delay(500); // Delay for stability
}
```

Tinkercad Simulation:

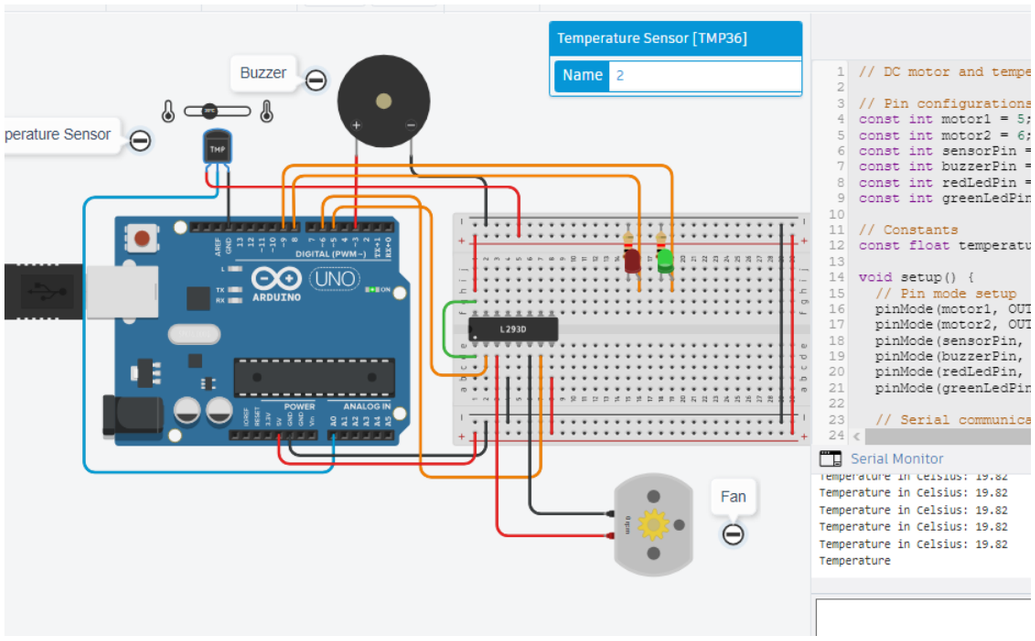

Figure 8. Simulation of Modified Ventilation System when Temperature is Less than 25C and the LED Green Indicator is ON

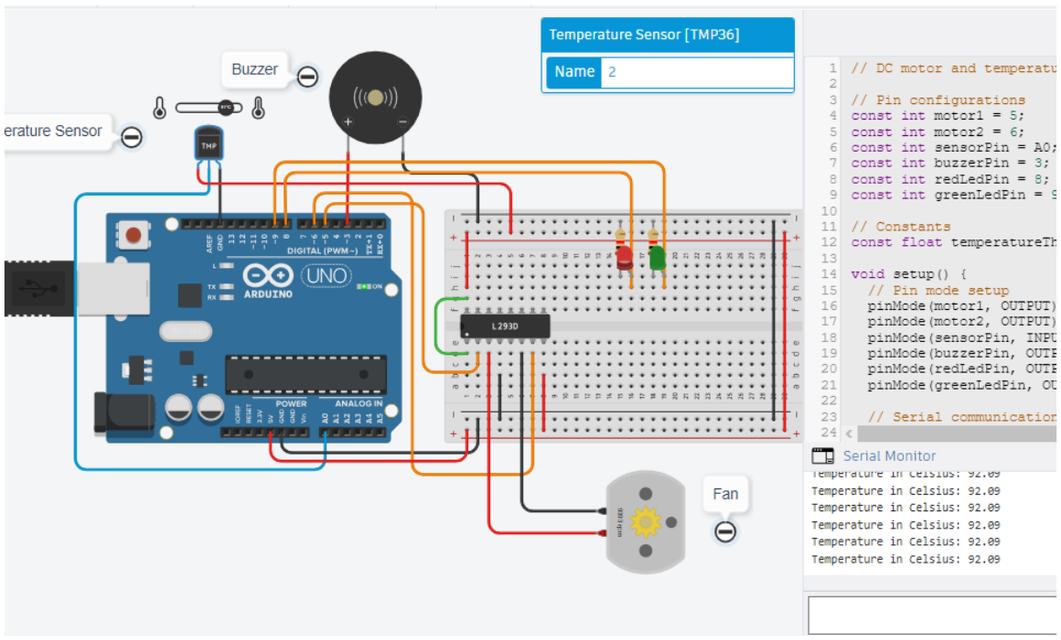

Figure 8. Simulation of Modified Ventilation System when Temperature is Greater than 25C and the LED Red Indicator is ON

V. Conclusion

In this laboratory activity, we successfully designed and implemented an Arduino system to monitor temperature using the TMP36 sensor and control a DC motor based on a predefined temperature threshold. The system also incorporates visual and auditory indicators through LEDs and a buzzer.

The system's key components include a DC motor, a TMP36 temperature sensor, a buzzer, and two LEDs (red and green) serving as temperature status indicators. The Arduino code reads the analog signal from the temperature sensor, converts it to Celsius using the TMP36 formula, and compares it to a predetermined threshold.

When the temperature exceeds the set threshold, the system activates the DC motor, sounds the buzzer, and turns on the red LED, indicating that the temperature is above the acceptable range. Conversely, when the temperature is below the threshold, the motor is turned off, the buzzer is silenced, and the green LED is illuminated, signaling that the temperature is within the desired range.

Integrating visual and auditory indicators enhances the user experience and makes it easier to identify the system's status at a glance. The Serial Monitor is also utilized to display the real-time temperature in Celsius, offering additional monitoring capabilities.

Through this laboratory activity, we gained practical experience in interfacing sensors and actuators with Arduino, implementing conditional logic for decision-making based on sensor readings, and incorporating multiple indicators for user feedback.

In summary, this activity has provided a hands-on opportunity to apply concepts related to sensor interfacing and actuation, reinforcing the understanding of Arduino programming and laying the groundwork for more complex projects involving sensor networks and environmental monitoring systems.

References

[1] Instructables. (2021, July 12). Temperature controlled DC motor | Arduino TINKERCAD. Instructables. https://www.instructables.com/Automatic-Food-Warmer-Arduino-TINDERCAD/

[2] Ramya, P. (2021, December 15). Speed control of a DC motor using temperature sensor. Medium. https://medium.com/@ramya.17c4c5/speed-control-of-a-dc-motor-using-temperature-sensor-80b65198a32f