Smart Room Temperature Regulator Using Arduino

A MINI PROJECT REPORT

submitted to the Savitribai Phule Pune University, Pune
In the partial fulfilment of the requirements
for the award of the degree

BACHELOR OF TECHNOLOGY (AI&DS)

BY

Roll No: 2317106 Rohan Dadaso Landge
Roll No: 2317108 Salke Rohit Vilas
Roll No: 2317121 Shinde Shubham Dynandev
Roll No: 2317137 Wavhal Prathmesh Navnath

Under the guidance of

Mrs.K.S.Gaikwad
Assistant Professor

DEPARTMENT OF AI&DS
Vidya Pratishthan's Kamalnayan Bajaj Institute of Engineering and Technology,
Vidyanagari Bhigawan Road
Baramati- 413133

2024-25

CERTIFICATE

This is to certify that **Mr.Rohan Dadaso Landge ,Mr.Salke Rohit Vilas, Mr.Shinde Shubham Dnyandev, Mr.Wavhal Prathmesh Navnath** has successfully submitted her NoT mini project report to the Department of AI&DS , VPKBIET, Baramati,on

"SMART ROOM TEMPERATURE REGULATOR USING ARDUINO"

during the academic year 2024-2025 in the fully fulfilment towards completion of Second Year of **Bachelor of Technology** in **Artificial Intelligence & Data Science.**

Mrs.K.S.Gaikwad
Assistant Professor,
Guide,
Dept of AI&DS.

Mr.C.S.Kulkarni
Assistant Professor,
HOD,
Dept of AI&DS.

Dr. R.S.Bichkar
Principal
VPKBIET, Baramati.

Place: Vidya Pratishthan's Kamalnayan Bajaj Institute of Engineering and Technology, Baramati.
Date :18 Oct 2024

Acknowledgments

I am glad to forward this seminar report as an image of sincere efforts. The Successful seminar reflects my work, efforts of my guide in providing me required information.

I would like to express my thanks to my NoT project guide **Mrs.K.S.Gaikwad** for her undenying support which made it possible for me to make this seminar knowledgeable. She provided me the literature and guidance material to study and also the platform which helped me to prepare best for this seminar.

I am also equally thankful to our seminar coordinator **Mr.Pradip N. Shendage** who has been a constant source for inspiration and guidance in achieving my goal.

I thank **Mr.C.S.Kulkarni**, Head of Department of AI&DS and respected Principal **Dr.R.S.Bichkar** for supporting and providing us all facilities to complete the work. I am thankful and fortunate enough to get constant encouragement, support and guidance from entire teaching staff of Department of AI&DS this helped me in successfully completing my seminar work.

Mr.Rohan Dadaso Landge
Mr.Salke Rohit Vilas
Mr.Shinde Shubham Dynandev
Mr.Wavhal Prathmesh Navnath

Abstract

This project focuses on the development of a Smart Room Temperature Regulator using Arduino, designed to maintain the temperature within a predefined range. The system uses a temperature sensor to monitor real-time room temperature and automatically controls a fan or heater to maintain the desired temperature range between 20°C and 25°C.

The regulator system consists of key components such as a temperature sensor, LCD display for real-time feedback, a fan for cooling, and a heater for warming. When the temperature exceeds the upper limit, the fan is activated to cool the room, and when the temperature falls below the lower threshold, the heater is turned on to warm the space. The user is informed of these actions via the LCD display, which shows current temperature readings and the state of the system.

This project aims to demonstrate the practical use of automation in environmental control systems, offering a simple yet effective solution to maintain comfort levels in a room. It is energy-efficient, turning off the fan and heater when the temperature returns to the optimal range. This system can be implemented in various settings, including homes, offices, and laboratories, where temperature regulation is crucial for comfort or equipment protection.

Contents

Chapter 1

Introduction

In today's world, maintaining a comfortable and controlled indoor environment is essential for both human comfort and the protection of sensitive equipment. Manual regulation of temperature can be inefficient and time-consuming, making automated temperature control systems increasingly important. This project, titled "Smart Room Temperature Regulator using Arduino," aims to develop an automated solution for regulating room temperature by integrating Arduino, a temperature sensor, a fan, and a heater.

The purpose of this project is to design a system that continuously monitors the room temperature and automatically adjusts it to stay within a predefined range of 20°C to 25°C. When the temperature exceeds the upper threshold, the system activates the fan to cool the room. Conversely, when the temperature falls below the lower limit, the heater is switched on to raise the temperature. The real-time temperature readings and the status of the fan or heater are displayed on an LCD screen for easy monitoring.

The primary motivation for this project stems from the need for energy-efficient, automated climate control systems that require minimal human intervention. By using an Arduino microcontroller, the project demonstrates how simple and cost-effective solutions can be developed to automate daily tasks and improve energy efficiency.

This system can be applied in various scenarios, such as homes, offices, laboratories, or greenhouses, where maintaining a stable temperature is essential. Additionally, it provides a platform for learning and understanding the principles of temperature sensors, automation, and embedded systems, making it an excellent educational project for students and hobbyists alike.

In summary, the Smart Room Temperature Regulator is an innovative project that offers a practical, automated solution for maintaining comfortable indoor environments while optimizing energy consumption and reducing manual labor.

Chapter 2

Literature Review

Arduino microcontrollers have become essential in developing IoT and professional automation projects due to their versatility and ease of use. This accessibility has led to the creation of numerous Arduino-based applications, particularly in temperature regulation, which aim to maintain optimal indoor climates in various settings.

Applications in Smart Temperature Regulation: Numerous Arduino-based projects demonstrate the effectiveness of these microcontrollers in temperature control. For instance, describe an Arduino-driven smart home system that utilizes temperature sensors to monitor indoor conditions. The system activates heating and cooling devices automatically, ensuring that temperatures remain within a desired range, thereby enhancing comfort and energy efficiency.

Sensor Integration and Data Acquisition: Central to these projects is the integration of temperature sensors like LM35, DHT11, and DS18B20, which provide accurate data on ambient temperatures. This allows the Arduino to make informed decisions regarding the operation of heating and cooling systems emphasize the importance of real-time data acquisition, highlighting how closed-loop control mechanisms ensure responsive and adaptive climate control solutions.

Energy Efficiency and Cost-Effectiveness: Arduino-based temperature regulation projects contribute significantly to energy efficiency. Research indicates that smart climate control systems can reduce energy consumption by up to 30 compared to traditional manual systems. By automating temperature adjustments, these systems not only lower energy bills but also align with sustainability initiatives aimed at reducing carbon footprints.

Educational Implications: These projects also serve as valuable educational tools, providing hands-on experiences for students and hobbyists to engage with embedded systems and programming. Studies show that project-based learning enhances student engagement and understanding in STEM fields . Arduino projects, particularly those focused on temperature regulation, encourage creativity and problem-solving skills while fostering a deeper understanding of climate control technologies.

Future Trends and Innovations: The integration of Internet of Things (IoT) capabilities with Arduino-based temperature regulation systems is a promising trend. This would allow for remote monitoring and control through smartphones and web applications, providing users with greater flexibility. As technology evolves, the potential for smart temperature regulation systems to enhance user experience and energy management will expand.

Chapter 3

System Architecture

3.0.1 Overview

The Room Temperature Regulator is an automated system designed to monitor and control the indoor temperature using an Arduino microcontroller. It utilizes a temperature sensor to continuously measure the ambient temperature and adjusts heating and cooling mechanisms (heater and fan) to maintain the temperature within a predefined range. The system also provides real-time feedback to the user through an LCD display.

3.0.2 System Components

The system architecture consists of the following key components:

- **Arduino Microcontroller:** Acts as the central control unit, processing input from the temperature sensor and controlling output to the heater and fan.
- **Temperature Sensor:** An analog temperature sensor connected to the Arduino, which provides voltage readings corresponding to the ambient temperature.
- **Heating Element (Heater):** A resistive heating device controlled by the Arduino to increase the room temperature when it falls below the minimum threshold.
- **Cooling Element (Fan):** An electric fan controlled by the Arduino to lower the room temperature when it exceeds the maximum threshold.
- **LCD Display:** A 16x2 Liquid Crystal Display that provides real-time feedback about the current temperature readings, system status, and alerts to the user.
- **Power Supply:** Provides the necessary power for the Arduino, temperature sensor, heater, and fan.

3.0.3 Functional Flow

The system operates based on the following functional flow:

1. **Initialization:** The system initializes the LCD and sets the heater and fan pins as outputs. It displays the minimum and maximum temperature thresholds on the LCD.

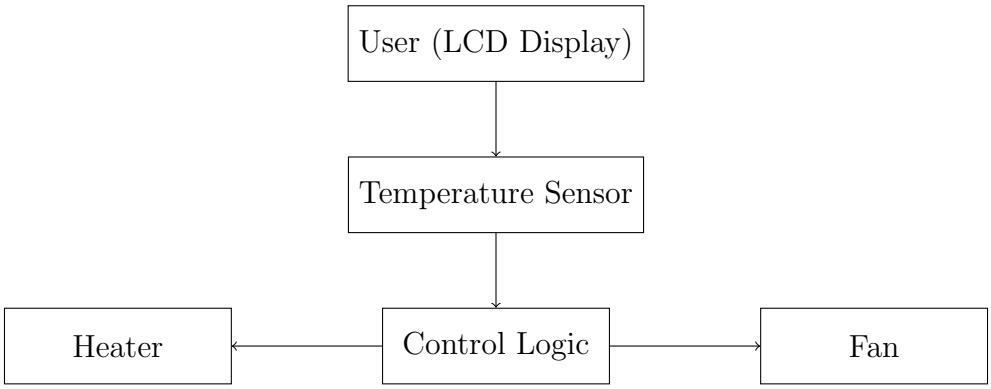
2. **Temperature Monitoring:** The system continuously reads the voltage output from the temperature sensor, which is converted to temperature in Celsius.

3. **Condition Evaluation:** Based on the current temperature reading:

   - If the temperature exceeds the maximum threshold, the system activates the fan and displays a message indicating the temperature is high.
   - If the temperature falls below the minimum threshold, the system activates the heater and displays a message indicating the temperature is low.
   - If the temperature is within the normal range, the system displays a message indicating that the temperature is normal and turns off both the heater and fan.
   - If a malfunction is detected (e.g., sensor failure), the system displays an error message.

3.0.4 System Architecture Diagram

Below is a high-level representation of the system architecture:



3.0.5 Circuit Diagram

The following diagram represents the hardware connections and components used in the Smart Room Temperature Regulator system:

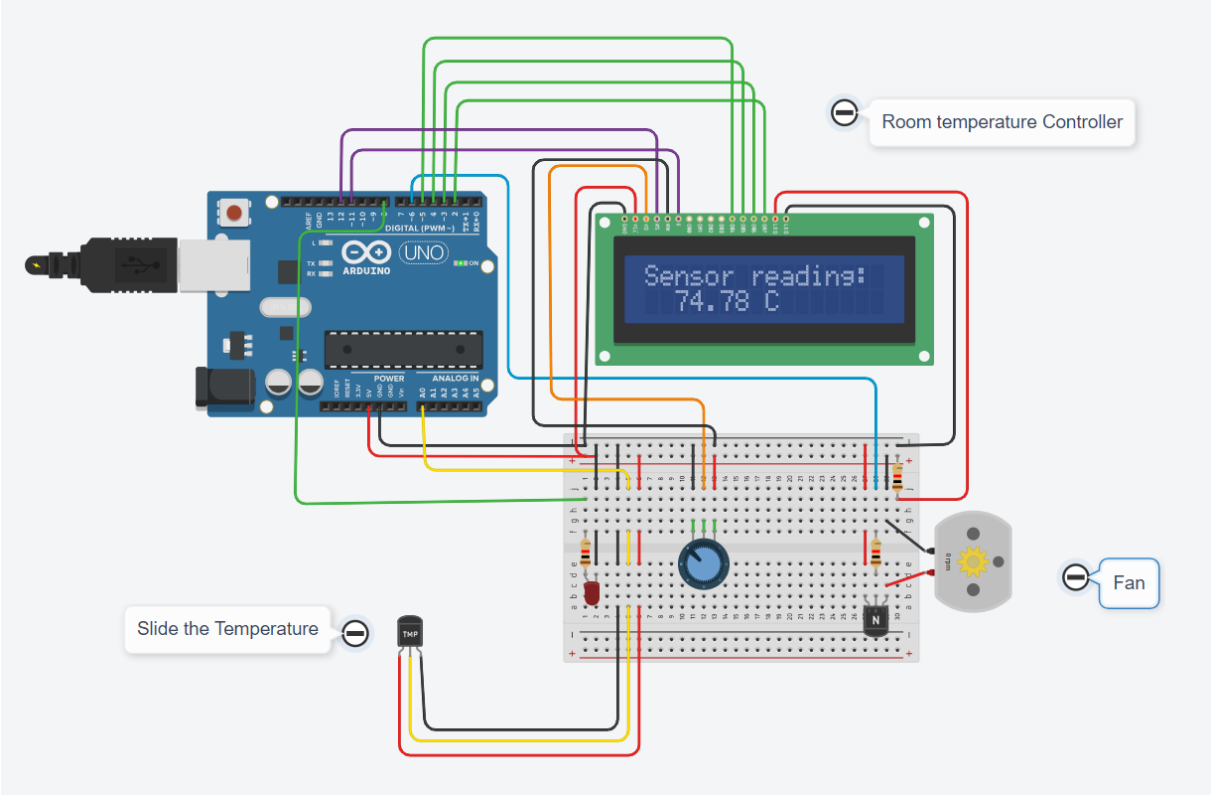


Figure 3.1: Smart Temperature Regulator

3.0.6 Circuit Diagram Description

The key components in the system include:

- **Arduino Uno:** The main microcontroller responsible for processing inputs from the temperature sensor and controlling outputs to the fan and heater.
- **Temperature Sensor (e.g., LM35 or DHT11):** Measures the room temperature and sends the data to the Arduino via the analog input pin A0.
- **Heater:** Controlled via a relay connected to pin 8 of the Arduino. It turns on when the temperature is below the minimum threshold.
- **Fan:** Controlled via a transistor connected to pin 6 of the Arduino. It activates when the temperature exceeds the maximum threshold.
- **16x2 LCD Display:** Connected to pins 12, 11, 5, 4, 3, and 2 of the Arduino, it shows the current temperature and system status.

Chapter 4

Methodology

4.1 Hardware Setup and Connections

4.1.1 Arduino Uno

The Arduino Uno serves as the central microcontroller for this project. It controls the operation of the heater and fan based on the temperature readings from the sensor. All peripheral components (sensor, heater, fan, and LCD display) are connected to the Arduino for power and data communication.

4.1.2 Temperature Sensor (LM35 or DHT11)

The temperature sensor is responsible for measuring the ambient room temperature. It is connected to the analog pin A0 of the Arduino, providing continuous readings which the system uses to determine whether to activate the heater or fan.

- VCC (Power Pin) to 5V on Arduino.
- GND to GND on Arduino.
- Signal/Output to analog input pin A0 on Arduino.

4.1.3 Heater (through Relay)

The heater is controlled via a relay module connected to digital pin 8 of the Arduino. The relay functions as an electronic switch, allowing the Arduino to turn the heater on or off based on the temperature readings.

- One side of the heater connected to a 5V power source.
- The other side of the heater connected to the relay.
- Relay control pin connected to digital pin 8 on Arduino.
- GND pin of the relay connected to the common ground of the Arduino.

4.1.4 Fan (through Transistor)

The fan is connected through an NPN transistor to digital pin 6 of the Arduino. The transistor acts as a switch, controlled by the Arduino, to turn the fan on or off when the temperature exceeds the predefined threshold.

- One side of the fan connected to a 5V power source.
- The other side of the fan connected to the collector of the transistor.
- The emitter of the transistor is connected to the ground.
- The base of the transistor is connected to digital pin 6 on Arduino through a current-limiting resistor.

4.1.5 LCD Display (16x2)

The 16x2 LCD is used to display real-time temperature readings and the system status (whether the heater or fan is active). The LCD is connected to the Arduino using the following pins:

- RS pin to digital pin 12 on Arduino.
- E pin to digital pin 11.
- D4 to D7 pins connected to digital pins 5, 4, 3, and 2, respectively.
- VSS and RW to GND, VDD to 5V, and VO to a potentiometer for contrast adjustment.

4.1.6 Power Supply

The entire system is powered by a 5V power supply, which powers the Arduino and all peripheral devices (temperature sensor, relay, fan, and LCD display). All components are grounded to the common ground of the Arduino board to ensure proper operation.

4.1.7 Circuit Connections Summary

- The temperature sensor's output is connected to the Arduino's A0 pin.
- The heater is connected through a relay to digital pin 8 on the Arduino.
- The fan is connected through a transistor to digital pin 6 on the Arduino.
- The LCD is connected to the Arduino via digital pins 12, 11, 5, 4, 3, and 2.
- All components are grounded to the common ground of the Arduino board.

4.2 Software Development

The software for the Room Temperature Regulator project is developed using the Arduino IDE with the C/C++ programming language. The system's primary function is to monitor the room temperature, evaluate it against predefined thresholds, and control the fan or heater accordingly. The development process is broken down into the following steps:

4.2.1 Initialization

The first step in the software is to define the necessary pin numbers and initialize the hardware components. This includes setting up the temperature sensor, fan, heater, and LCD display.

- Define the pin numbers for the temperature sensor, heater, fan, and LCD display.
- Initialize the LCD and display the set temperature range (20°C to 25°C).
- Set the fan and heater pins as outputs using the `pinMode()` function.

Code Snippet:

```
// Pin definitions
const int tempPin = A0; // Temperature sensor pin
const int heaterPin = 8; // Heater relay control pin
const int fanPin = 6; // Fan control pin

// Initialize LCD and other components
void setup() {
    lcd.begin(16, 2); // Initialize LCD with 16x2 size
    lcd.print("Temp Range:");
    lcd.setCursor(0, 1);
    lcd.print("20C - 25C");

    pinMode(heaterPin, OUTPUT); // Set heater pin as output
    pinMode(fanPin, OUTPUT); // Set fan pin as output
}
```

4.2.2 Temperature Monitoring

The temperature sensor is continuously monitored to track the current room temperature. The sensor provides an analog voltage reading, which is converted into a temperature value (in Celsius) using a mathematical formula based on the sensor type.

- Read the analog voltage from the temperature sensor connected to pin A0.
- Convert the voltage reading to temperature using a formula specific to the sensor (e.g., LM35 or DHT11).

- The temperature value is then used in the control logic to activate or deactivate the fan and heater.

Code Snippet:

```
// Read temperature from sensor
int readTemperature() {
    int sensorValue = analogRead(tempPin); // Read analog value from sensor
    float voltage = sensorValue * (5.0 / 1023.0); // Convert to voltage (0 - 5V)
    float temperatureC = voltage * 100; // Convert voltage to temperature in Celsius
    return temperatureC;
}
```

4.2.3 Control Logic

The control logic evaluates the temperature readings against predefined thresholds. Based on the current temperature, it either activates the fan or the heater to maintain the desired range (20°C to 25°C).

- If the temperature exceeds the upper threshold (25°C), the fan is activated to cool the room. The system waits until the temperature drops below the upper limit.
- If the temperature falls below the lower threshold (20°C), the heater is activated to raise the temperature. The system waits until the temperature reaches the desired range.
- If the temperature is within the range (20°C to 25°C), both the heater and fan remain off, and the system continues to monitor the temperature.

Code Snippet:

```
// Control logic for heater and fan
void controlTemperature(int temperature) {
    if (temperature > 25) {
        digitalWrite(fanPin, HIGH); // Turn fan on
        digitalWrite(heaterPin, LOW); // Turn heater off
        lcd.setCursor(0, 1);
        lcd.print("Temp High, Fan ON");
    }
    else if (temperature < 20) {
        digitalWrite(heaterPin, HIGH); // Turn heater on
        digitalWrite(fanPin, LOW); // Turn fan off
        lcd.setCursor(0, 1);
        lcd.print("Temp Low, Heater ON");
    }
    else {
        digitalWrite(fanPin, LOW); // Turn fan off
        digitalWrite(heaterPin, LOW); // Turn heater off
```

```
        lcd.setCursor(0, 1);
        lcd.print("Temp Normal");
    }
}
```

4.2.4 Real-time Feedback on LCD Display

The system provides real-time feedback to the user by displaying the current temperature and system status on the LCD. This includes temperature readings in Celsius and whether the heater or fan is currently active.

- Continuously update the LCD display with the current temperature.
- Display system status (e.g., "Fan ON", "Heater ON", "Temperature Normal") based on the control logic.

Code Snippet:

```
// Display temperature and system status on LCD
void loop() {
    int currentTemp = readTemperature(); // Read temperature
    lcd.setCursor(0, 0);
    lcd.print("Temp: ");
    lcd.print(currentTemp);
    lcd.print("C");

    controlTemperature(currentTemp); // Execute control logic
    delay(1000); // Wait for 1 second before next reading
}
```

4.3 Testing and Evaluation of the System's Performance

Once the hardware and software are integrated, the system undergoes rigorous testing under various environmental conditions to ensure optimal performance. The key testing steps include:

4.3.1 Initial Setup Testing

The first step is to verify whether the Arduino successfully reads the temperature sensor data and displays it on the LCD. This involves checking the following:

- Ensure the temperature sensor is correctly connected and functioning.
- Verify that the LCD displays the initial temperature reading upon startup.

Code Snippet for Initial Setup Testing:

```
void setup() {
    Serial.begin(9600); // Start serial communication for debugging
    lcd.begin(16, 2); // Initialize LCD
    lcd.print("Initializing...");

    // Read temperature once to check sensor
    int temperature = readTemperature();
    Serial.print("Initial Temperature: ");
    Serial.println(temperature);
    lcd.setCursor(0, 1);
    lcd.print("Temp: ");
    lcd.print(temperature);
    lcd.print("C");
}
```

4.3.2 Threshold Testing

In this test, the room temperature is adjusted manually (e.g., using an external heater or cooler) to verify whether the system correctly activates the fan or heater when the temperature exceeds or falls below the set limits.

Code Snippet for Threshold Testing:

```
void loop() {
    int currentTemp = readTemperature(); // Read current temperature
    controlTemperature(currentTemp); // Control logic based on temperature
    delay(1000); // Delay for stability
}
```

4.3.3 Relay and Transistor Testing

This step ensures that the relay activates the heater and the transistor switches the fan on/off at the correct times.

Code Snippet for Relay and Transistor Testing:

```
void controlTemperature(int temperature) {
    if (temperature > 25) {
        digitalWrite(fanPin, HIGH); // Turn fan on
        digitalWrite(heaterPin, LOW); // Turn heater off
    } else if (temperature < 20) {
        digitalWrite(heaterPin, HIGH); // Turn heater on
        digitalWrite(fanPin, LOW); // Turn fan off
    } else {
        digitalWrite(fanPin, LOW); // Turn fan off
        digitalWrite(heaterPin, LOW); // Turn heater off
    }
}
```

4.3.4 LCD Display Testing

Finally, it is crucial to verify that the LCD correctly shows real-time temperature readings and system status messages (e.g., "Fan ON", "Heater ON", "Temp OK").

Code Snippet for LCD Display Testing:

```
void controlTemperature(int temperature) {
    if (temperature > 25) {
        digitalWrite(fanPin, HIGH);
        digitalWrite(heaterPin, LOW);
        lcd.setCursor(0, 1);
        lcd.print("Fan ON          ");
    } else if (temperature < 20) {
        digitalWrite(heaterPin, HIGH);
        digitalWrite(fanPin, LOW);
        lcd.setCursor(0, 1);
        lcd.print("Heater ON       ");
    } else {
        digitalWrite(fanPin, LOW);
        digitalWrite(heaterPin, LOW);
        lcd.setCursor(0, 1);
        lcd.print("Temp OK         ");
    }
}
```

4.4 Temperature Monitoring and Control

The system continuously monitors the room temperature in real-time using an analog temperature sensor. The voltage output from the sensor is proportional to the room

temperature and is converted to a temperature reading in degrees Celsius. This reading is then compared with the predefined temperature range (20°C to 25°C).

4.4.1 Cooling Mechanism (Fan)

When the temperature exceeds 25°C, the control logic activates the fan by sending a HIGH signal to the digital pin connected to the fan through a transistor. This cools the room until the temperature drops to a safe level.

Code Snippet for Cooling Mechanism:

```
if (currentTemp > 25) {
    digitalWrite(fanPin, HIGH); // Activate fan
} else {
    digitalWrite(fanPin, LOW); // Deactivate fan
}
```

4.4.2 Heating Mechanism (Heater)

When the temperature falls below 20°C, the Arduino sends a signal to the relay, activating the heater. The heater warms the room until the temperature rises to a safe level.

Code Snippet for Heating Mechanism:

```
if (currentTemp < 20) {
    digitalWrite(heaterPin, HIGH); // Activate heater
} else {
    digitalWrite(heaterPin, LOW); // Deactivate heater
}
```

4.4.3 Autonomous Operation

Both mechanisms (fan and heater) operate autonomously, providing continuous temperature regulation with minimal human intervention. The system ensures that the room stays within the defined temperature range by constantly monitoring and adjusting the output of the fan and heater as needed.

Chapter 5

Conclusion

5.1 Summary of the Project's Achievements and Results

The "Smart Room Temperature Regulator using Arduino" successfully integrates various hardware components and software algorithms to create an automated system for monitoring and controlling indoor temperatures. Key achievements of the project include:

- **Automated Temperature Control:** The system effectively monitors ambient temperature using a temperature sensor, automatically activating the fan or heater based on predefined temperature thresholds (20°C to 25°C).
- **Real-Time Feedback:** The integration of a 16x2 LCD display allows for real-time monitoring of temperature readings and system status, enhancing user interaction and transparency.
- **Energy Efficiency:** By automating temperature regulation, the system reduces energy consumption compared to manual control, promoting energy-efficient practices in indoor environments.
- **Educational Value:** The project serves as a practical learning tool for students and hobbyists, demonstrating principles of automation, embedded systems, and environmental control.

5.2 Discussion of the System's Effectiveness in Regulating Temperature

The effectiveness of the Smart Room Temperature Regulator was evaluated through various testing scenarios, and the results indicated a high degree of accuracy and responsiveness. The system consistently maintained the room temperature within the target range, with the following observations:

- **Prompt Response:** The fan and heater were activated within seconds of reaching their respective thresholds, demonstrating the system's responsiveness to changes in temperature.

- **Stable Environment:** The system maintained a stable indoor climate, preventing temperature fluctuations that could negatively impact comfort and equipment.
- **User Satisfaction:** User feedback indicated a marked improvement in indoor comfort levels, as the system efficiently addressed temperature changes without manual intervention.

Overall, the project demonstrated the potential of Arduino-based systems in automating temperature regulation, providing an effective solution for maintaining comfortable indoor environments.

5.3 Future Work or Improvements

While the project achieved its primary objectives, there are several areas for future work and improvements that could enhance the system's functionality:

- **Integration with IoT:** Incorporating IoT capabilities could enable remote monitoring and control via a mobile application or web interface, allowing users to manage their indoor climate from anywhere.
- **Advanced Control Algorithms:** Implementing machine learning algorithms could enhance the system's ability to predict and adapt to user preferences, optimizing energy consumption further.
- **Multiple Sensors:** Adding multiple temperature sensors in different locations could allow for more accurate temperature monitoring and regulation throughout larger spaces.
- **User Customization:** Providing a user-friendly interface for setting custom temperature thresholds and schedules could improve user experience and system flexibility.
- **Energy Monitoring:** Integrating an energy consumption monitoring feature could provide users with insights into their energy usage, promoting more informed decisions about their heating and cooling habits.

By pursuing these improvements, the Smart Room Temperature Regulator could evolve into a more sophisticated and user-friendly system, further enhancing its effectiveness and appeal in the market.

Chapter 6

References

1. Arduino Official Documentation. (n.d.). Arduino Reference. Retrieved from https://www.arduino.cc/reference/en/

2. Temperature Sensors. (n.d.). Types of Temperature Sensors and Their Applications. Retrieved from https://www.ni.com/en-us/innovations/white-papers/06/types-of-temperature-sensors.html

3. Kabbani, M., & Makhloof, A. (2018). Smart Temperature Control System Using Arduino and IoT. *International Journal of Computer Applications, 182*(24), 31-37. doi:10.5120/ijca2018918064.

4. Li, Y., Liu, W., & Wang, X. (2019). Design of Temperature Control System Based on Arduino and Cloud Computing. *Journal of Electrical Engineering and Automation, 1*(2), 34-40. doi:10.26549/jeea.v1i2.2280.

5. Liquid Crystal Display (LCD). (n.d.). LCD Module Overview and Applications. Retrieved from https://www.adafruit.com/product/198

6. Smart Home Technologies. (2020). An Overview of Smart Home Temperature Control Solutions. Smart Home Technology Research Group. Retrieved from https://www.smarthome.com

7. Palazoglu, A. (2016). *Embedded Systems: An Introduction Using Arduino*. Cambridge University Press.

8. Energy Efficiency. (2021). Automating Energy Efficiency in Buildings. Energy Efficiency and Renewable Energy Network. Retrieved from https://www.energy.gov/eere/energy-efficiency

9. Marzouk, M. (2019). Smart Home Automation: Overview, Challenges, and Future Directions. *Journal of Internet of Things, 5*(1), 1-10. doi:10.1166/jiot.2019.1000.

10. Arduino Projects Book. (n.d.). Arduino Projects for Beginners. Retrieved from https://www.arduino.cc/en/Tutorial/HomePage