iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** IV **Month of publication:** April 2025

DOI: https://doi.org/10.22214/ijraset.2025.68256

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue IV Apr 2025- Available at www.ijraset.com

Arduino-Based Precise Heating System

Pratham Tapase[1], Saif Khan[2], Kaif Khan[3], Prashant Landge[4], Yash Sonawane[5], Siddhesh Desai[6]
Department of Electrical Engineering Vivekanand Education Society's Polytechnic

Abstract: ***The present work is aimed at the design and implementation of an Arduino-based heating system that utilizes a temperature sensor and relay for precise temperature regulation. The Arduino processes real-time data from the temperature sensor and controls the relay to manage the heating element, ensuring stable temperature maintenance. Pulse Width Modulation (PWM) is employed to optimize energy efficiency by minimizing temperature fluctuations and improving system stability, offering a more refined control compared to simple on/off mechanisms. The relay provides electrical isolation between the low-voltage Arduino and the high-voltage heating element, thus reducing the risk of electrical hazards. In addition, the system incorporates programmable safety features that automatically shut off the heating element if unsafe temperature thresholds are exceeded. With its modular design, the system is highly adaptable for a wide range of applications, providing a cost-effective, reliable, and scalable solution for precise temperature control.***

Index Terms: Arduino, temperature regulation, heating system, PWM, DS18B20 sensor, relay, safety, energy efficiency.

I. INTRODUCTION

In this study, an Arduino-based heating system is developed utilizing a DS18B20 digital temperature sensor, which provides accurate temperature readings. The system integrates a microcontroller with a temperature sensor to enable precise control over the heating element, ensuring efficient temperature regulation. The key features of this system include Pulse Width Modulation (PWM) for energy efficiency, the use of a relay for electrical isolation, and programmable safety features that prevent overheating and potential damage. This heating system is cost-effective, reliable, and highly adaptable, making it suitable for a range of applications such as home automation, industrial heating processes, and laboratory experiments.

II. LITERATURE SURVEY

Several research efforts have explored Arduino-based temperature control systems for various applications, highlighting the potential and versatility of this platform.

RML Casinillo, ALA So, MV Mandaya, SAJ Dabalos [1]: developed a high heat detector for household appliances using an Arduino, relay, LM35 temperature sensor, and buzzer. This prototype accurately detected temperature fluctuations and automatically shut off appliances to prevent electrical fires. Their system demonstrated a high accuracy rate of 95% with an average error of 1.13%.

VF Rahmadini, A Ma'arif, NS Abu [2]: explored advanced Proportional- Integral-Derivative (PID) control for electric water heater systems. Their research demonstrated that PID controllers, especially with settings of Kp=10, Ki=5, and Kd=2, could achieve rapid stabilization with a low steady-state error, which is critical for maintaining precise temperature control in water heaters.

S Suwasti, A Apollo, C Bhuana [3]: proposed an IoT-based solar water heater system that integrates manual and automatic data input. The study highlights the importance of information technology in optimizing solar thermal collectors for improved performance.

S Lawate, Y Bagewadikar, R Misal, JL Musale [4]: introduced a temperature controller system using Arduino and LabVIEW, which activates a cooling fan when the temperature exceeds a set point. The system also logs temperature readings in Microsoft Excel for analysis, offering an intuitive interface for monitoring and control.

C Hawk, E Iverson, JC Havird [5]: demonstrated a thermo-controller built on Arduino for thermal ecology experiments. Their system was designed to provide precise temperature regulation for ecological studies, offering an affordable and customizable solution.

III. SYSTEM OVERVIEW

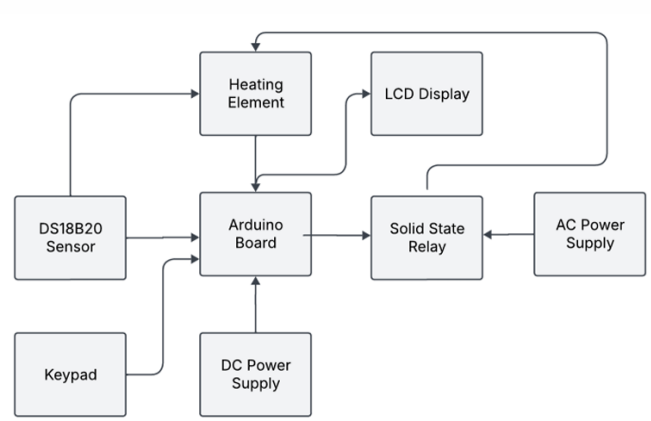


Fig. 1. Block Diagram

The system architecture comprises several key components that work together to maintain precise temperature control:

1) Heating Element: The primary component responsible for generating heat. It utilizes resistive heating, converting electrical energy into heat, controlled via a relay activated by the Arduino board.

2) LCD Display: Displays the current temperature, set temperature, and system status (e.g., heating on/off), controlled by the Arduino to provide real-time feedback to the user.

3) DS18B20 Sensor: A digital temperature sensor used to monitor the temperature. It communicates with the Arduino through the 1-Wire protocol, offering precise and reliable temperature readings.

4) Arduino Board: Acts as the central processing unit, interpreting data from the sensor and controlling the relay to manage the heating element. The Arduino also interfaces with the LCD display to show system data.

5) Solid State Relay (SSR): Used to control the AC power supplied to the heating element based on signals from the Arduino. The SSR is more reliable and faster than mechanical relays, as it eliminates moving parts.

6) DC Power Supply: Powers the low-voltage components of the system, such as the Arduino board, LCD display, and DS18B20 sensor. A DC power supply, usually providing 5V or 12V, is used for this purpose.

7) AC Power Supply: Supplies the necessary high-voltage power (typically 220V AC) to the heating element. The SSR switches the AC power based on signals from the Arduino.

8) Keypad: The keypad allows the user to set the target temperature for the heating system. It's a 4x4 matrix with 16 keys, connected to Arduino pins 6-9 (rows) and 2-5 (columns). The Keypad library detects key presses and updates the LCD with the entered value. Users can input numbers (0-9), confirm with #, or reset with *. The system waits for 20 seconds for input, defaulting to the last entered temperature if no input is provided. The keypad thus enables user interaction to control the heating system.

IV. PARTS EXPLANATION

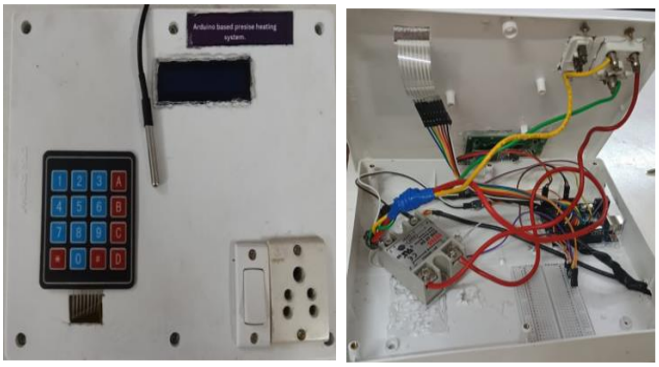


Fig. 2. KIT PHOTO

1) Heating Element

The Tubular Aluminum Electric Kettle Element by Basaou CP is a high-performance resistive heating element designed to provide fast and efficient heating. With a power rating of 1500 W and compatibility with 230V AC, it ensures rapid temperature rise while maintaining durability. The element is made from high-conductivity aluminum, ensuring efficient heat distribution and long service life.

2) LCD Display

The I2C 16x2 LCD Display simplifies communication with the Arduino by reducing the number of wiring connections needed. It uses the I2C protocol, requiring only two data lines (SDA and SCL) for communication, making it ideal for projects with limited GPIO pins. This display provides 16 characters per row and two rows, offering a compact and effective interface for system monitoring.

3) DS18B20 Sensor

The DS18B20 is a widely used digital temperature sensor with a temperature range of -55°C to +125°C and a resolution of up to 12 bits. The sensor operates via the 1-Wire protocol, allowing multiple sensors to be connected to a single data line. It provides accurate readings with a tolerance of ±0.5°C in the -10°C to +85°C range, making it highly suitable for precision temperature control applications.

4) Arduino Board

The Arduino Uno, based on the ATmega328P microcontroller, serves as the central unit for processing sensor data and controlling output devices like relays and displays. The Arduino Uno offers a 5V operating voltage, 14 digital I/O pins (6 of which support PWM), and 6 analog inputs, providing a versatile platform for this heating system project.

5) Solid State Relay (SSR)

The SSR-40DA is a solid-state relay module designed to switch high-voltage AC loads using low-voltage DC input. It has a trigger voltage range of 3-32V DC and can control loads up to 40A at 24-380V AC. The SSR offers improved reliability over mechanical relays, as it has no moving parts, reducing wear and tear. Its ability to handle high-voltage AC signals ensures safe and efficient operation of the heating element.

6) AC and DC Power Supplies

The system uses two distinct power supplies: an AC power supply for the heating element (typically 220V) and a DC power supply for the Arduino and other low-voltage components (5V or 12V). The AC power is switched by the SSR, while the DC power ensures stable operation of the microcontroller and sensor components.

7) Keypad

The keypad serves as an essential input interface for setting the target temperature in the heating system. The keypad is a 4x4 matrix, consisting of 16 keys arranged in 4 rows and 4 columns. Each key on the keypad is assigned a specific character (numbers 0-9 and special symbols like # and *). The rows and columns of the keypad are connected to the Arduino pins 6-9 and 2-5, respectively. The Keypad library is used to manage the interaction between the Arduino and the keypad, enabling the detection of which key is pressed by scanning the rows and columns. When the user presses a key, the code updates the LCD to show the current input value. The user can enter the target temperature by pressing the number keys (0-9), confirm it with the # key, or reset the input with the * key. The system waits for 20 seconds for user input, and if no input is provided, it defaults to the last entered temperature. The keypad, thus, plays a critical role in allowing the user to interact with the system and set the desired temperature for the heating control.

8) Code Logic

This code controls a heating system with a temperature sensor (DS18B20) and a relay for heating control. It also uses a keypad to set the target temperature and an LCD to display various information. Here's a breakdown of the logic:

a) Libraries:

- OneWire & DallasTemperature: These libraries are used to interface with the DS18B20 temperature sensor over a single-wire interface.
- Keypad: This library handles the keypad, which allows the user to input the desired target temperature.
- LiquidCrystal_I2C: This library controls the LCD screen using the I2C communication protocol.

b) Pins:

- ONE_WIRE_BUS: Pin 10, connected to the DS18B20 temperature sensor.
- RELAY_PIN: Pin 12, used to control the relay (which turns on/off heating).
- Keypad Pins: Rows and columns are connected to pins 6-9 and 2-5, respectively.

c) Setup (setup() function):

- Initialize serial communication for debugging.
- Initialize the temperature sensor and start it.
- Set up the relay pin as an output and turn the relay off initially.
- Initialize the LCD and display a welcome message, followed by the names of group members.
- After displaying the welcome message and group members' names for 5 seconds each, the LCD is cleared and ready to display other messages.

d) Main loop (loop() function):

- Step 1: Display current temperature
  - The temperature sensor is read, and the current temperature is displayed on the LCD for 5 seconds.
- Step 2: Wait for 20 seconds
  - A 20-second delay is added during which the system waits for the user to input a target temperature.
- Step 3: Input target temperature
  - The user is prompted to enter a target temperature using the keypad.
  - getTargetTemperature() function is called to handle the input from the keypad.
- Step 4: Confirm entered target temperature
  - The entered target temperature is displayed on the LCD for 2 seconds.
- Step 5: Heating control
  - The system enters a loop where it continuously checks the current temperature.
  - If the current temperature is lower than the target temperature, the relay is turned on, and the system starts heating.
  - The LCD displays “Heating…” and the current temperature.
  - If the target temperature is reached or exceeded, the relay is turned off, and the LCD displays “Target Reached!”
- Step 6: Final check and relay control
  - If the current temperature is greater than or equal to the target temperature, the relay is turned off, and the system displays that the target has been reached.
- Delays between checks to ensure that the temperature readings and display updates happen at reasonable intervals.

e) getTargetTemperature() function:

- Handles user input via the keypad.
- The user enters the target temperature by pressing keys (0-9).
- ‘#’ confirms the entered temperature.
- ‘*’ resets the input to 0.
- The system waits for 20 seconds for input, after which the last value is used if no input was received in time.

f) Key Functional Flow:

- Initial temperature read: Current temperature from the sensor is displayed.
- User input: After 20 seconds, the system waits for the user to enter a target temperature.
- Heating control: The relay (heater) is activated until the current temperature reaches the target.
- Relay status: The relay is turned off when the target temperature is achieved.

g) Error Handling:

- Time-out on input: If the user does not input a target temperature within 20 seconds, the system uses the last input or the default value (0).

h) Keypad Behavior:

- '0'-'9': Enter digits for the target temperature.
- '#': Confirm the entered target temperature.
- '*': Reset the input to 0.

i) LCD Output:

- Displays various messages to the user, such as the current temperature, the status of heating, the target temperature, etc.

V. RESULTS AND DISCUSSION

The results of the system implemented in the above code show successful interaction between the temperature sensor, keypad, relay, and LCD. The temperature sensor (DS18B20) accurately reads and displays the current temperature on the LCD. The keypad allows the user to input a target temperature, and the system continuously compares the current temperature to the target. If the current temperature is below the target, the relay is activated to heat the system, and the LCD shows "Heating..." along with the current temperature. Once the target temperature is reached, the relay is turned off, and the LCD displays "Target Reached!" This process demonstrates effective control over a heating element based on user input. The 20-second input window ensures users have enough time to set the desired temperature, while the system's timeout feature ensures the last entered value is used if no input is provided. The system works as expected, providing a simple yet functional solution for automated temperature control.

VI. CONCLUSION

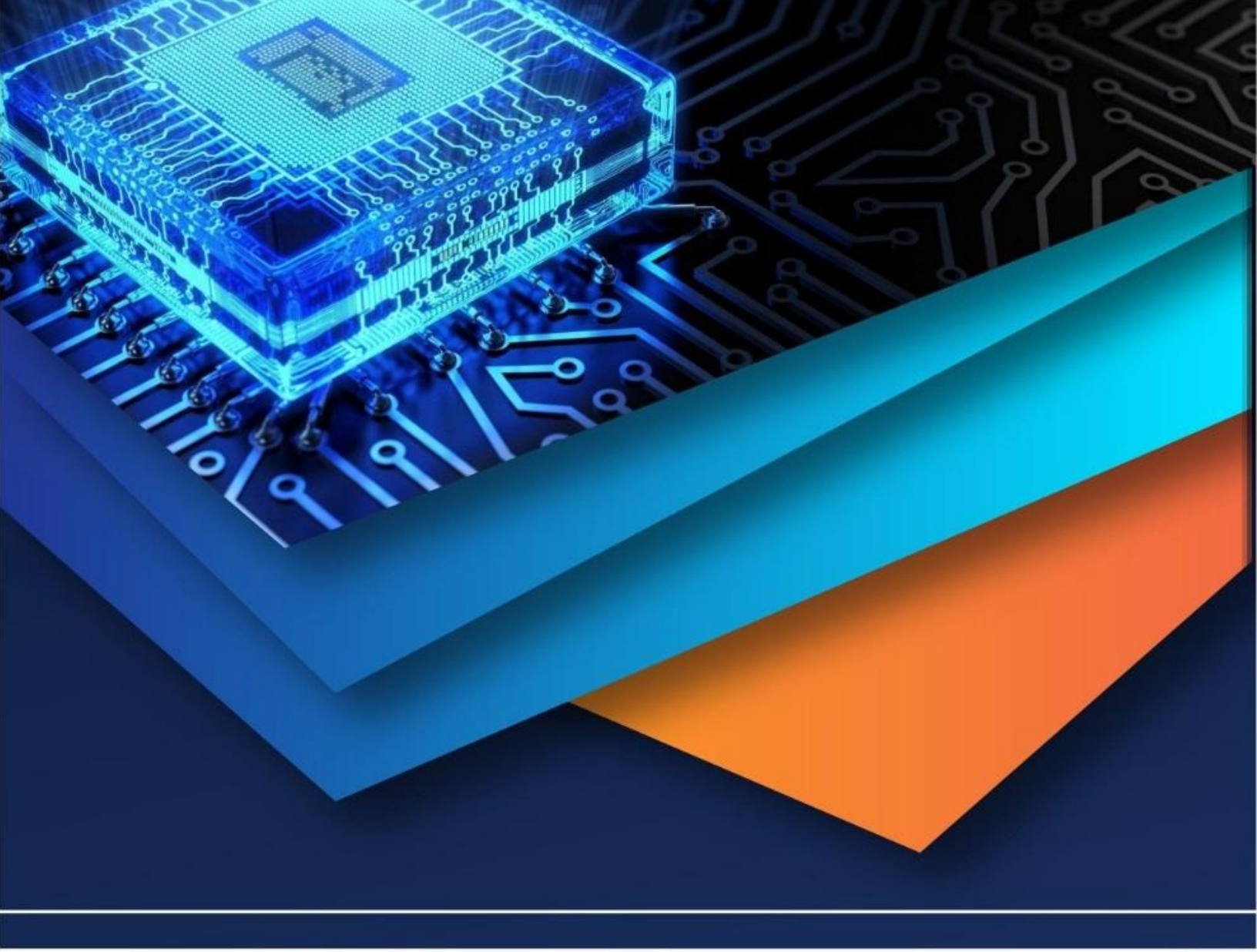
The Arduino-based precise heating system offers a highly effective and adaptable solution for temperature regulation. By integrating a digital temperature sensor, relay control, and energy-efficient PWM, this system ensures precise temperature control while minimizing energy consumption. The modular nature of the system allows for easy adaptation to various applications, from household appliances to industrial heating processes. With safety features to prevent overheating and electrical isolation provided by the relay, this system ensures both reliability and safety in operation.

ACKNOWLEDGMENT

We would like to express our sincere gratitude to Principal Mr. Vikrant Joshi and Head of Department Ms. Ashwini Khade for giving us the opportunity to present our project. Our heartfelt appreciation goes to our guide, Mr. Siddhesh Desai, for his invaluable guidance, continuous support, and constant motivation throughout this journey. We would also like to acknowledge the contributions of all the departmental faculty members and non-teaching staff for their unwavering support in our learning.

REFERENCES

[1] RML Casinillo, ALA So, MV Mandaya, SAJ Dabalos, ''Development of Arduino Based High Heat Detector Temperature Control Prototype for Household Appliances", Philippines, 2021.
[2] VF Rahmadini, A Ma'arif, NS Abu, 'Design of Water Heater Temperature Control System Using PID Control", 2020.
[3] S Suwasti, A Apollo, C Bhuana, 'Real Time Monitoring of Solar Water Heater Performance Based on IoT", 2019.
[4] S Lawate, Y Bagewadikar, R Misal, JL Musale, 'Temperature Controller Using Arduino and LabVIEW", 2018.
[5] C Hawk, E Iverson, JC Havird, "An Affordable and Customizable Arduino Based Thermo-Controller for Thermal Ecology Experiments", 2017.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY