ISSN: 1672 - 6553

JOURNAL OF DYNAMICS AND CONTROL

VOLUME 10 ISSUE 03: P242-248

SMART PID TEMPERATURE CONTROL SYSTEM WITH RELAY OUTPUT AND LCD DISPLAY

Jitendra Gaikwad, Pratima Patil, Vivek Patil,Punam Dhangar

Department of Instrumentation Engineering, Vishwakarma Institute of Technology, Pune, India

JOURNAL OF DYNAMICS AND CONTROL
https://doi.org/10.71058/jodac.v10i03009

SMART PID TEMPERATURE CONTROL SYSTEM WITH RELAY OUTPUT AND LCD DISPLAY

Jitendra Gaikwad[1], Pratima Patil[2], Vivek Patil[3],Punam Dhangar[4]
Department of Instrumentation Engineering, Vishwakarma Institute of Technology, Pune, India
jitendra.gaikwad@vit.edu, pratima.patil24@vit.edu, vivek.patil24@vit.edu, punam.dhangar24@vit.edu

ABSTRACT: *In this project, an ESP32 microcontroller is used to construct a PID-based temperature control system. In order to maintain a user-defined setpoint, the system automatically controls a relay-driven heater after continuously measuring the temperature using a DS18B20 digital sensor. While a Wi-Fi-enabled web interface enables live monitoring and change of setpoint and PID parameters (Kp, Ki, Kd), a 16x2 LCD display provides real-time feedback of temperature, setpoint, and relay condition. Smooth heating, little overshoot, and steady temperature control are guaranteed by the PID controller. System reliability is improved by safety features such a cutoff at 70°C and EEPROM storage for setpoints and PID parameters. The efficacy of PID implementation on a low-cost microcontroller platform is validated by experimental results that show accurate and responsive temperature management.*
Keywords- PID Controller, Temperature Control, ESP32, DS18B20, Relay Module, LCD Display, I2C, Embedded System, Microcontroller, Time-Proportional Control

I. INTRODUCTION

Many industrial, laboratory, and home uses, including ovens, incubators, water heaters, and chemical reactors, depend heavily on temperature control. Process quality, efficiency, and safety are all guaranteed by maintaining a specific temperature. Conventional ON/OFF controllers are straightforward; however, they frequently cause temperature fluctuations and overshoot. Because PID (Proportional-Integral-Derivative) control provides smooth, accurate, and steady control, it is frequently employed to get around these restrictions.

This project uses the ESP32 microcontroller to construct a PID-based temperature control system. In order to maintain the user-defined setpoint, the system automatically controls a relay-driven heater while continuously monitoring the temperature using a DS18B20 digital sensor. While a Wi-Fi-enabled web interface enables remote monitoring and real-time PID parameter adjustment, a 16x2 LCD display provides local feedback on temperature, setpoint, and relay status. Additionally, the system uses EEPROM storage to maintain setpoint and PID parameters throughout power cycles and includes safety features such an automatic shutdown when the temperature rises above 70°C. This method shows how PID control and microcontrollers may be used to precisely and easily regulate temperature.

II. LITERATURE REVIEW

Due to PID's simplicity, well-understood behavior, and ease of implementation on microcontrollers, temperature management using classical PID controllers has been extensively explored. PID temperature control systems based on microcontrollers have been shown in a number of studies in various applications.

- In "MCU-Based PID Temperature Control System for Linear Heating and Cooling," Hu (2023) created a heating/cooling controller using an STC12 microcontroller as its foundation. The system demonstrates that microcontroller-based PID can provide high precision and stability in difficult thermal settings by achieving precise temperature regulation over a wide range (room temperature to 400°C), with a steady-state error < 0.1°C and near-linear temperature ramping.
- In "Temperature Control System and its Control using PID Controller," Singh et al. (2016) created a PID controller, calculated the electric oven's transfer function, and modeled it mathematically. When compared to open-loop control, their experimental results demonstrated better step response characteristics (less overshoot, quicker settling), demonstrating the usefulness of classical PID for thermal processes like ovens and heaters.
- For applications like 3D printing, a study titled "FDM 3D printer temperature control system based on PID control" employed PID control to regulate the temperatures of the heated bed and extruder. The outcome demonstrated far faster heating and stabilization than non-PID (open-loop) control, proving that PID may greatly boost performance even in non-traditional thermal systems.

https://doi.org/10.71058/jodac.v10i03009

- In "The Use of Arduino and PID Control Approach for the Experimental Setup of HVAC Temperature Testing," The authors demonstrated that microcontroller-based PID may be utilized successfully even for air-temperature regulation in environmental control setups by using an Arduino-based PID system to manage an electric heater for an HVAC test rig.
- A recent work "Implementing PID Control on Arduino Uno for Air Temperature Optimization" (2024) used an Arduino Uno to study PID-based air temperature control and reported thorough testing of proportional, integral, and derivative parameters. The viability of classical PID for thermal regulation even in slow-responding systems is reinforced by the paper's emphasis on how the system achieves steady and accurate air temperature control by appropriately adjusting PID gains.

III. METHODOLOGY

The methodology of this project involves sensing, control computation, actuation, and user interaction. The DS18B20 temperature sensor continuously measures the current temperature and sends the data to the ESP32 using the One-Wire protocol. TheThis project's methodology includes user interaction, actuation, control computation, and sensing. The DS18B20 temperature sensor uses the One-Wire protocol to provide data to the ESP32 after continuously measuring the current temperature. The ESP32 uses the PID algorithm to handle this measured value after comparing it to the user-defined setpoint. Smooth and reliable temperature management is made possible by the PID controller's calculation of an output that indicates how long the relay should stay ON within a predetermined time window.

As the actuator, the relay module attached to the ESP32 turns the heater on or off based on the PID output. Easy local monitoring is made possible by a 16x2 LCD with an I2C interface that shows the temperature, setpoint, and relay condition in real time. Additionally, from any connected device, users may examine real-time temperature data, change the setpoint, and alter PID parameters via the ESP32's web-based interface. The setpoint and PID gains are saved in EEPROM storage, guaranteeing that the system maintains its configuration even when the power is turned off. This process guarantees precise, reliable, and easy-to-use temperature management.

A.Hardware Components

- **ESP32**

Fig. 1. ESP32 Microcontroller Module

The ESP32 is a potent, inexpensive microcontroller that combines Bluetooth and Wi-Fi, making it perfect for standalone and Internet of Things applications. It can readily communicate with sensors and actuators thanks to its dual-core processor, several GPIO pins, ADC channels, and different communication protocols. The ESP32 is the main controller in this project. It uses the PID algorithm to calculate the necessary control output after reading temperature data from the DS18B20 sensor. It then operates the relay module to turn the heater on or off. It also communicates with the 16x2 LCD through the I2C module to provide relay status, setpoint values, and temperature readings in real time. Because of its versatility and processing capacity, the ESP32 is an excellent choice for handling real-time temperature management duties. It also offers the possibility of future expansion, such as web-based monitoring or logging.

- **DS18B20 Temperature Sensor**

Fig. 2. DS18B20 Digital Temperature Sensor

A single data line can be shared by several digital temperature sensors thanks to the DS18B20's 1-Wire interface. It is appropriate for precise temperature management applications because it offers extremely

https://doi.org/10.71058/jodac.v10i03009

accurate temperature readings across a broad range. In this project, the inaccuracies frequently associated with analogue sensors are eliminated because the DS18B20 continuously measures the ambient or heater temperature and transmits digital data to the ESP32. The system can identify even minute temperature variations because to the sensor's high-resolution measurement capabilities. Since the PID algorithm depends on exact input values to control the relay and maintain the intended temperature setpoint, accuracy is essential for efficient control.

- **Relay Module 5V Single Channel**

Fig. 3. 5V Single-Channel Relay Module

The microcontroller may use low-voltage logic signals to control high-power devices thanks to the 5V single-channel relay module, which functions as an electrically powered switch. The relay in this temperature control system is linked to the heater and managed by the ESP32. To maintain the intended setpoint temperature, the relay turns the heater on or off based on the PID output. Because the module has an active LOW input, the relay is turned on by a LOW logic signal from the ESP32 and off by a HIGH signal. By using the relay, the low-power ESP32 may safely regulate a higher-power heating element without directly handling hazardous voltages, guaranteeing system functionality and safety.

- **LCD Display 16x2**

Fig. 4. 16×2 Alphanumeric LCD Display

The system's settings are shown locally and in real time on the 16x2 alphanumeric LCD. It can show the temperature, setpoint, and relay status all at once because it can display two lines of 16 characters each. The LCD in this project makes it simple for users to check whether the relay is ON or OFF, view the setpoint value, and watch the current temperature readings from the DS18B20 sensor. This visual feedback, which offers instantaneous insight into the system's functioning, is crucial for both development and demonstration. The LCD uses less GPIO pins on the ESP32 when paired with the I2C module, which simplifies the cabling and enables the microcontroller to effectively control additional peripherals.

- **I2C Module**

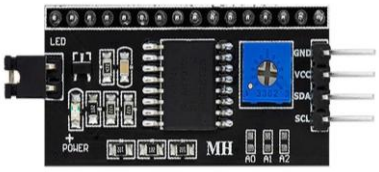


Fig. 5. I2C Module for LCD Interface

An interface adapter called the I2C module makes it easier for the ESP32 and the 16x2 LCD to communicate. An LCD typically needs several data pins to function, but the I2C module enables the LCD to communicate with just two pins—SDA and SCL. In addition to lowering the amount of connections needed, this frees up ESP32 pins for more sensors or actuators. The I2C module simplifies the hardware setup and increases overall system stability in this project by making it simple for the ESP32 to send commands and display data on the LCD. The I2C module guarantees a clean and manageable design by lowering wiring complexity, which is particularly advantageous for embedded system applications like PID-based temperature management.

B. Software Components

The Arduino IDE, which offers a user-friendly platform for programming the ESP32 microcontroller, was used to create the software for this project. The operation of this project depends on the Arduino IDE's support for C/C++ programming and its ability to seamlessly integrate additional libraries. The DS18B20 sensor's temperature reading, PID

https://doi.org/10.71058/jodac.v10i03009

computations, relay module control, and LCD display updating are the main software functions.

The OneWire and DallasTemperature libraries were utilised to read temperature data from the DS18B20 sensor. If necessary, several sensors can share a single data line thanks to the OneWire library, which facilitates communication over a single digital pin. By managing sensor commands and conversions automatically, the DallasTemperature library makes it easier to read temperature in degrees Celsius.

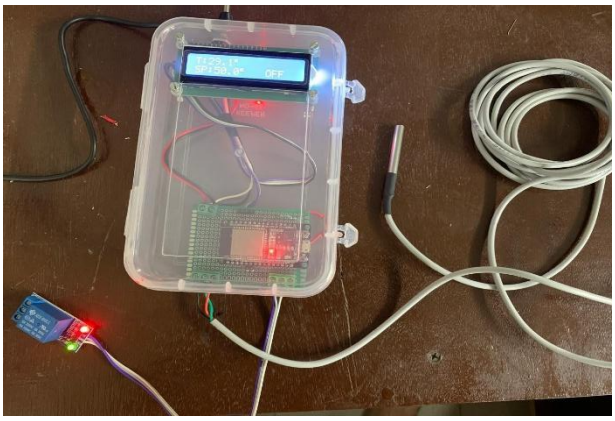
The PID_v1 library was utilised to implement the PID controller. This library offers a conventional, tried-and-true PID algorithm that calculates the output using the setpoint, the current temperature, and the derivative, integral, and proportional gains. The relay module is then controlled by the output in a time-proportional manner, enabling accurate heater ON/OFF switching.

The I2C module was utilised to interface with the 16x2 LCD using the LiquidCrystal_I2C library. Real-time temperature, setpoint, and relay status visualisation is made possible by this library, which lowers the number of GPIO pins needed and streamlines data and command transmission to the display.

The WiFi and WebServer libraries were used to optionally provide a basic web interface. By serving as an access point, the ESP32 enables real-time temperature monitoring, PID output, and relay status via a web browser. Additionally, users can remotely modify the setpoint and PID gains.

Reliable temperature regulation and transparent system parameter visualisation are made possible by the software's overall integration of sensor readings, PID control, relay actuation, and display updates into a coherent system.

C.Workflow:



Real-time temperature measurement, control, and display are all part of the system's operation. This is how the workflow can be explained.

Temperature Sensing: Using GPIO 4, the DS18B20 sensor continuously detects the temperature and transmits the results to the ESP32.

PID Control: The PID algorithm is used by the ESP32 to calculate the error between the measured temperature and the setpoint. The duration of the relay's ON state within a predetermined time window is determined by the output value that the PID controller computes.

Relay Control: Depending on the PID output, the relay, which is attached to GPIO 32 (Active LOW), turns the heater on or off. When the temperature falls below the setpoint, the heater turns on, and when the setpoint is reached, it turns off. If the temperature rises above 70°C, a safety cutoff makes sure the heater is switched off.

UI on the Web and Display: The temperature, setpoint, and relay status are all continuously shown on the 16x2 LCD. The ESP32 also functions as a Wi-Fi access point, hosting a webpage that allows users to adjust PID parameters, establish the setpoint, and monitor temperature in real-time. Every second, the website is updated.

EEPROM Storage: Even after a power cycle, the system maintains the setpoint and PID settings stored in EEPROM.

https://doi.org/10.71058/jodac.v10i03009

Note on PID Tuning:
The immediate reaction to the temperature mistake is determined by Kp (Proportional Gain). Although a larger Kp enhances responsiveness, oscillation may result.
Ki (Integral Gain): Removes steady-state error, but if it's too large, it can delay the system.
To minimize overshoot, Kd (Derivative Gain) forecasts future mistake; too much Kd might add noise.
PID Range Suggestion:
20–40 Kp
0.1–0.8 Ki
50–120 Kd
For slow heating control, the project defaults are Kp = 35, Ki = 0.25, and Kd = 80.

IV. Result

The temperature setpoint was successfully maintained with great precision and stability by the PID-based temperature control system that was put into place. The ESP32 used the PID algorithm to manage the relay-controlled heater based on precise and continuous temperature readings from the DS18B20 sensor. In a few of minutes, the system reached the setpoint with little oscillation or overshoot, exhibiting smooth heating behavior. While the web interface offered real-time updates and allowed users to remotely modify the setpoint and PID parameters, the 16x2 LCD displayed temperature, setpoint, and relay status in real-time, making local monitoring simple.Overall, the system demonstrated dependable and responsive temperature control, demonstrating the efficacy of PID implementation on the ESP32 platform in a useful heating application. Additionally, the EEPROM storage maintained the setpoint and PID parameters even after power cycles, allowing the system to resume normal operation without reconfiguration. The safety cutoff feature effectively prevented the temperature from exceeding 70°C, ensuring secure operation.

V. CONCLUSION

Using an ESP32 microcontroller, a PID-based temperature control system is successfully implemented in this project. A relay for heater control, a 16x2 LCD for local visualization, and a DS18B20 sensor for precise temperature monitoring are all integrated into the system. Usability is improved via a Wi-Fi-enabled web interface that enables real-time PID parameter monitoring and adjustment. Compared to basic ON/OFF controllers, the PID controller performs better by maintaining the intended setpoint with little overshoot and seamless changes. EEPROM storage guarantees that setpoints and PID gains are maintained throughout power cycles. This design can be expanded to a variety of heating applications that need accurate, dependable, and interactive temperature management.

VI. FUTURE SCOPE

The performance, functionality, and usability of the PID-based temperature control system described in this project can be improved in a number of ways. The integration of several temperature sensors, which enables dispersed temperature monitoring for larger or more complicated systems, is one possible enhancement. Multi-zone temperature control, in which several heaters or cooling components are individually controlled based on separate setpoints, can also be added to the system. Adding sophisticated control algorithms, like model predictive control or adaptive PID, could increase system effectiveness and lower energy usage. Better monitoring and analysis capabilities could be achieved by upgrading the online interface to incorporate data logging, graph plotting, and remote alerts via mobile notifications. The system is also appropriate for smart home and industrial applications since it can be coupled with cloud services or IoT platforms for remote management and real-time monitoring. Lastly, swapping out the relay for a solid-state relay or PWM-controlled heater could provide more precise heating power control and lessen mechanical wear, making the system more durable and energy-efficient. These improvements would increase the system's applicability and make it a flexible answer to contemporary temperature management needs.

VII. REFERENCES

1. Li, X., & Zhang, W., “Q-Learning Based PID Auto-Tuning for Temperature Regulation,” *Journal of Process Control*, 2019.

2. *Håland, J., Hovland, V. E., & Johansen, T. A., “Deep Reinforcement Learning-Based PID Controller Design,” IEEE Access, 2020..*

3. Park, S., & Lee, J., “Adaptive Temperature Control Using Actor–Critic Reinforcement Learning,” *Control Engineering Practice*, 2021.

4. Pannu, H. A., Gupta, A. K., & Singh, H., “Reinforcement Learning-Based Auto-Tuning of PID Controllers for Dynamic Systems,” *Neural Computing and Applications*, 2021..

https://doi.org/10.71058/jodac.v10i03009

5. Busoniu, D., & Babuska, R., "Q-Learning Based Temperature Control of a Heating Process," *IEEE Conference on Decision and Control (CDC)*, 2017

6. Chen, L., & Yu, Y., "Deep Q-Network Based PID Tuning for Nonlinear Temperature Systems," *Artificial Intelligence in Engineering Journal*, 2023..

7. Sutton, R. S., & Bello, M. G., "RL-Enhanced PID Controller for Real-Time Industrial Thermal Loops," *IEEE Transactions on Industrial Informatics*, 2022.