



Temperature PID Control on a Customized Trainer using Siemens S7-1200 PLC

Jianwei Ren

Department of Chemical Engineering, University of Pretoria, cnr Lynnwood Road and Roper Street, Hatfield 0028, South Africa.

jianwei.ren@up.ac.za

Abstract. This paper presents a stepwise guide for implementing temperature proportional-integral-derivative (PID) control on a customized model using a Siemens S7-1214C DC/DC/DC programmable logic controllers (PLC) workstation. The customized trainer includes an aluminium block, heating element, RTD PT-100 sensor, PT-100 transmitter, solid-state module, PLC-based PID controller, and 24 V power supply. The detailed implementation process includes initial setup, pretuning, fine-tuning, and data retention. The aim of this work is to provide guidance to practitioners in the field of temperature control and enable them to effectively apply PID control to various industrial processes.

Keywords: Temperature control; Proportional-integral-derivative; Siemens S7-1200 PLC

1 Introduction

Industrial production processes often require temperature control. A programmable logic controller (PLC) is a small computer capable of receiving data through its inputs and sending operating instructions through its outputs. The primary function of a PLC is to control a system's functions using the internal logic programmed into it. One commonly used method for controlling temperature or other system parameters is the proportional-integral-derivative (PID) algorithm [1-4]. **Fig. 1a** illustrates the algorithm of a typical PID controller, which sets the power using the formula 1 [5]:

$$\dot{q} = k_p(T_s - T) + k_I \int (T_s - T) dt + k_D \frac{d}{dt} (T_s - T) \dots \dots \dots (1)$$

Where T is the measured temperature, T_s is the desired temperature (or set temperature), and k_p , k_I and k_D are set by the user.

The PID controller generates a pulse-width modulation (PWM) signal based on the temperature input signal feedback. PWM techniques are commonly used in power electronics to control semiconductor devices, while simultaneously achieving output voltage control and harmonic elimination or reduction at the inverter's output [6]. PWM is an effective control technique that generates analog signals from digital devices such as microcontrollers, and the resulting signal is a train of pulses in the form of square waves, with each pulse being either high or low. By converting the signal into discrete parts, PWM reduces the average power delivered by an electrical signal. As shown in **Fig. 1b**, a comparator generates a PWM signal by comparing a modulating signal and

a non-sinusoidal or sawtooth wave. The width of the pulse generated at the output is determined by the magnitude of the modulating signal compared to the sawtooth wave. If the sawtooth signal is greater than the modulating signal, the output signal is in a high state. Meanwhile, the duty cycle is a critical parameter associated with PWM signals. **Fig. 1c** depicts that a PWM signal remains on for a specific duration and off for a certain time. The percentage of time for which the signal stays on is known as the duty cycle. A signal that is always on has a duty cycle of 100%. The duty cycle can be calculated using the following formula 2:

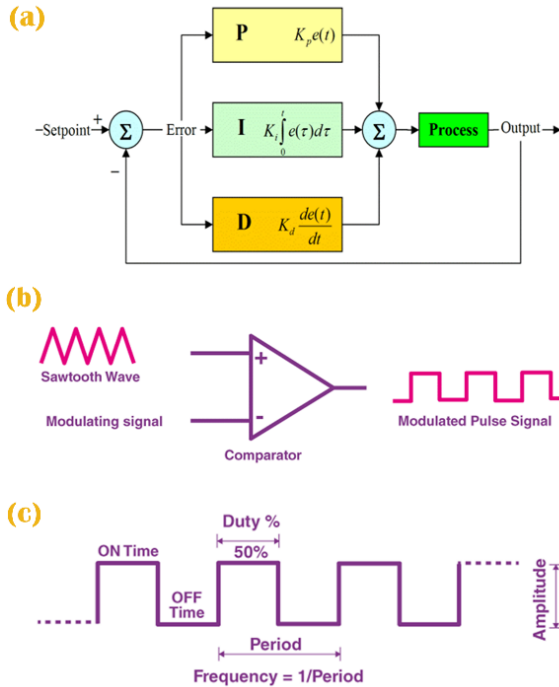


Fig. 1. (a) The algorithm of a typical PID controller. (b) The generation of a pulse width modulation (PWM) Signal. (c) Duty cycle of a PWM signal.

$$\text{Duty Cycle} = \frac{\text{Turn On Time}}{\text{Turn On Time} + \text{Turn Off Time}} \dots\dots\dots(2)$$

The average voltage value is dependent on the duty cycle, and therefore the average value can be adjusted by controlling the width of the pulse's on-time. The frequency of a PWM signal determines how quickly it completes a period. The frequency can be calculated using the formula:

- $\text{Frequency} = 1/\text{Time Period} \dots\dots\dots(3)$

- $\text{Time Period} = \text{On Time} + \text{OFF time} \dots\dots\dots(4)$

where the time is the sum of the on-time and off-time of the signal. The output voltage of a PWM signal is proportional to the duty cycle. For instance, if the operating voltage is 5 V and the duty cycle is 100%, then the output voltage will also be 5 V. On the other hand, if the duty cycle is 50%, then the output voltage will be 2.5 V.

In this work, the PWM will be used to digitally control power and implement a PID algorithm in a PLC, and explore how the proportional, integral, and derivative gains affect the controller's behavior. The readers will be demonstrated how to choose the optimal values for these parameters to achieve effective temperature control. More specifically, the the temperature PID control implementation process will be demonstrated using a Siemens S7-1200 PLC workstation. The system uses an analog input module to convert temperature changes into digital signals that the PLC can process. The PID controller compares the current temperature with the set temperature and adjusts the output power accordingly until the desired temperature is reached. If the current temperature exceeds the set value, the system will take appropriate action to bring the temperature back to the desired level.

2 Customized trainer hardware

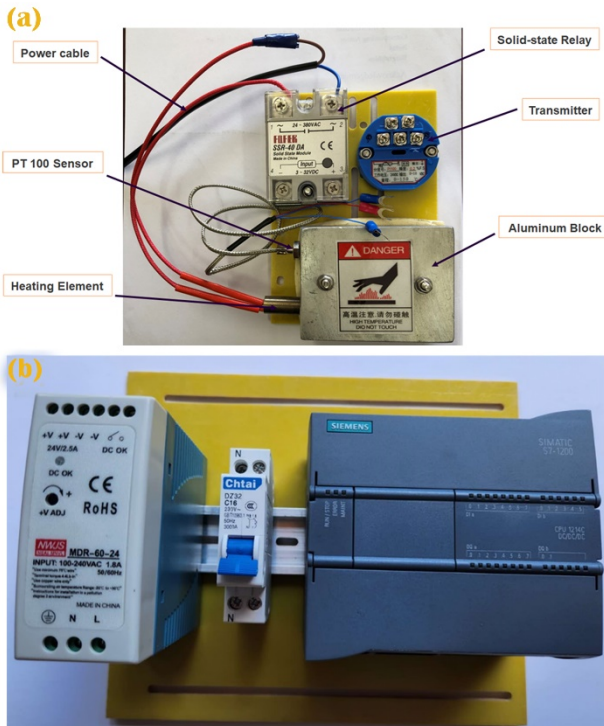


Fig. 2. (a) The proposed PID temperature control model. (b) Digital photos of 24 VDC power supply, switch and PLC1214c DC/DC/DC module on rail.

As depicted in **Fig. 2a**, the temperature control trainer hardware consists of an aluminum block whose temperature is monitored using a Transmitter/PT100 temperature sensor. The continuous signals from the sensor are fed back to the PLC PID controller, which in turn controls the solid-state relay (SSR-40DA). The SSR-40DA is responsible for regulating the power supplied to the heating element, and thereby enables precise temperature control of the aluminum block. Provided that the goal is to maintain the temperature of the aluminum block at a desired value such as 60 °C, as the block releases heat to the environment, a PID control model is necessary to regulate its temperature accurately. The RTD-100 sensor generates a signal proportional to the detected temperature, and the transmitter then converts this signal into a standard 0–10 V voltage output. The heating element is controlled by the SSR-40DA drive to maintain the desired temperature. Here, the SSR-40DA Module is a non-contact high-speed switching device that prioritizes reliability and user safety. It utilizes a lightweight DC voltage to safely switch high-voltage and high-current circuits without exposing the user to potential electrical hazards. Note that the SSR-40DA Terminal Type unit uses a Zero Cross Trigger control method to regulate between 24V and 380V AC circuits, which could cause serious harm to users if a leak or fault occurs. This not only ensures the safety of users but also allows devices and machinery to be operated remotely. In addition, the SIMATIC S7-1214C CPU in **Fig. 2b** has 14 digital inputs, 10 digital transistor outputs, and 2 analog inputs for efficient communication and workflow. It is also compatible with various communication protocols such as Ethernet, MODBUS, TCP/IP, PROFINET, and PROFIBUS. With built-in system diagnostics, it can detect errors and display them on the TIA portal web server. The digital output of the S7-1214C CPU will be used to drive the control side of the SSR-40DA instead of a mechanical relay due to the high cycling rates involved. The PLC code will be written to PID control a duty cycle for the On/Off time of the SSR-40DA, which will be controlled by PWM output from the S7-1214C CPU, where a range of 0 to 100% will control the heater power supply and maintain the desired temperature of the aluminum block.

3 Hardware wiring connections

As illustrated in **Fig. 3**, the hardware can be connected by following the steps below:

- (1) Connect the red wire of PT-100 sensor to the RT terminal of Transmitter.
- (2) connect the two blue wires to the negative (-) of RT terminal of the Transmitter.
- (3) Connect the positive (+) and negative (-) terminal of the 24 V power supply to the positive (+) and negative (-) terminal of the Transmitter.
- (4) Connect the Signal Out (+) of the Transmitter to the AIO terminal of the PLC 1214c module.
- (5) Connect the negative (-) terminal of the Transmitter to the 2M terminal of the PLC 1214c.
- (6) Connect one of the two wires of the Heating Element to the Neutral terminal of the 220 VAC Power Supply.
- (7) Connect the other wire of the Heating Element to the terminal #1 of the Solid-state Relay (SSR).
- (8) Connect the terminal #2 of the Solid-state Relay (SSR) to the Live terminal of the 220 VAC Power Supply.

- (9) Connect the terminal #3 of the Solid-state Relay (SSR) to the DO2 terminal of the PLC 1214c module.
- (10) Connect the terminal #4 of the Solid-state Relay (SSR) to the COM terminal of the negative (-) terminal of the 24 V power supply.
- (11) Connect the positive (+) terminal of the 24 V power supply to the 3L+ terminal of the PLC 1214c module.
- (12) Connect the negative (-) terminal of the 24 V power supply to the 3M terminal of the PLC 1214c module.
- (13) Plug the 24 V power supply, switch, and PLC 1214c module onto the rail, respectively.
- (14) Connect the PLC 1214c module to the computer via an internet cable to RJ45 port. The computer has been installed with TIA Portal Software.
- (15) For convenience, a push button will be connected to initialize the digital input into the PLC 1214c module.
- (16) Up to now, the hardware connections are done, as shown in Figure 8. The cables can be tied to make it look neater.

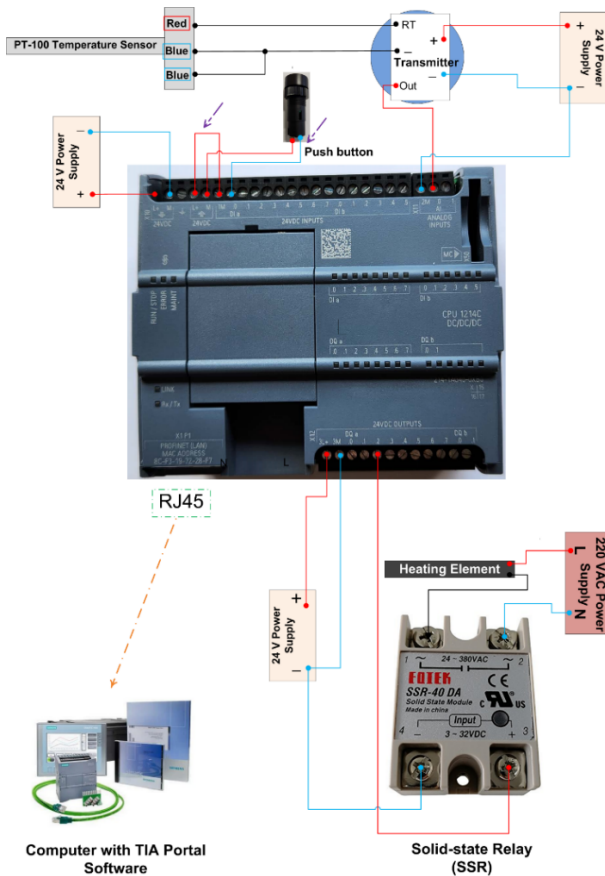


Fig. 3. Hardware wiring connections.

4 PLC Programming processes

The steps can be followed to achieve the temperature measurement and heating control of the aluminum block. follow. The first step is to establish communication between the PT-100 sensor and the corresponding transmitter through the PLC-1214c module to ensure temperature measurement is possible. To achieve PID control of the aluminum block at the desired temperature, the Q0.2 channel of the PLC-1214c module will output PWM using a counterOut task. The generated PWM signals pulse the SSR and vary the power to the heater rapidly to achieve fine control over the amount of power into the heater.

Afterwards, a PID function block will be applied to achieve temperature PID control on the aluminum block using the following control logic:

- Collect temperature data from the PT-100 sensor and use it as the process variable input.
- Continuously compare the process variable with the set temperature using the given PID gains.
- Use the PID output in percentage to generate the PWM signal.
- The PWM signal will turn the relay on/off to control the heating element. Note that for this task, an SSR must be used instead of a mechanical relay because the relay would cycle at high rates and wear the contacts over a short time.
- After PID autotuning, a new set of PID gains will be derived from the PID process. Save and apply these gains to the PID process to achieve the desired temperature control.

In programming the PID control, it is necessary to create the PLC tags and set their data type and address. As shown in **Fig. 4**, the tags required for this task include:

- **Desired Temperature:** This tag will be used to set the desired temperature setpoint for the PID control. The data type for this tag should be 'Real' and its address can be set as '%MD100'.
- **Measured Temperature:** This tag will be used to collect temperature data from the PT-100 temperature transmitter sensor. The data type for this tag should be 'Int' and its address should be '%IW64'.
- **PWM Output:** This tag will be used to generate the PWM signal that controls the SSR. The data type for this tag should be 'Bool' and its address should be '%Q0.2'.
- **Error:** This tag is to feedback any process errors. The data type for this tag should be 'Bool' and its address is put as '%I0.3'.
- **PID_State:** This tag is to show the state of the PID control. The data type for this tag should be 'Word' and its address is put as '%MW104'.
- **PLC_Manual:** This tag allows to put the PLC in a manual mode. The data type for this tag should be 'Bool' and its address is put as '%I0.0'.

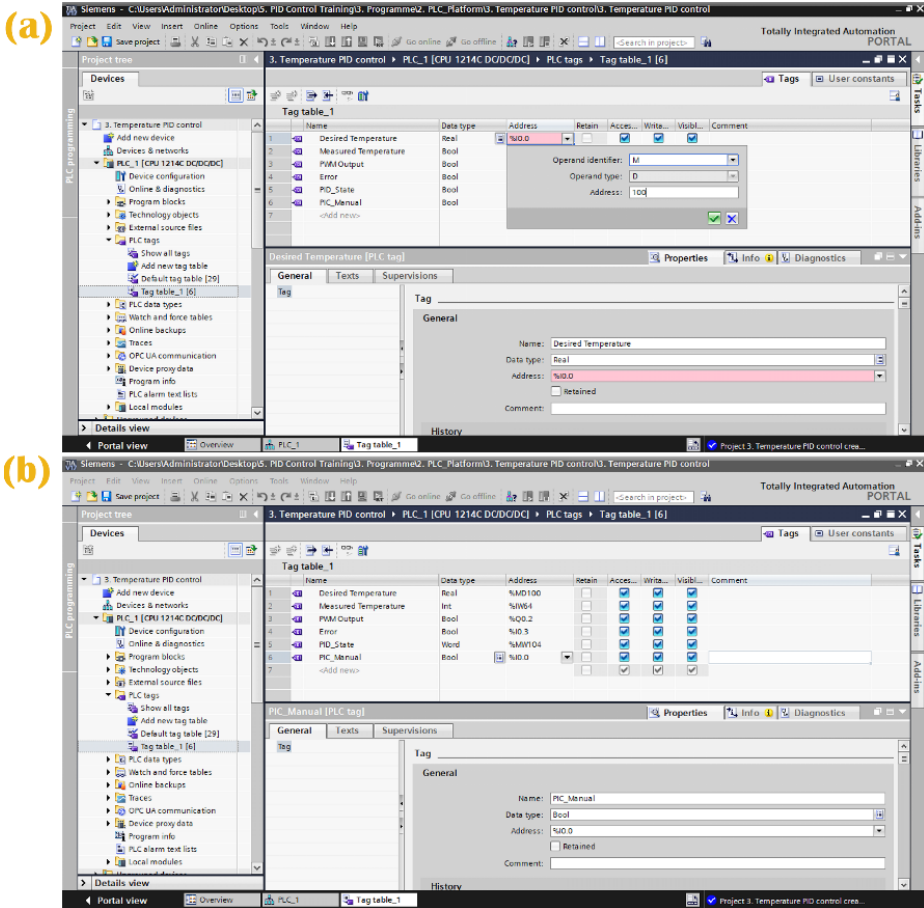


Fig. 4. Create the PLC tags, set their 'Data type' and 'Address'.

5. Remarks

It is important to note that the addresses provided in this paragraph are just examples and may vary depending on the specific configuration of the PLC and the programming environment. It is recommended to refer to the PLC manual and the programming software documentation for the correct addressing scheme.

Acknowledgments. The author thanks the financial support of Research Development Program (RDP) at University of Pretoria.

Disclosure of Interests. The author has no competing interests to declare that are relevant to the content of this article.

References

1. Ding, S.C., Li, W.H.: Temperature monitoring system based on PLC. TELKOMNIKA, 11, 7251–7258 (2013).
2. Kumar, P., Pathan, S.S., Mashilkar, B.: Liquid level control using PID controller based on Labview & Matlab software. Int. J. Eng. Res. Techno. 3, 111–114 (2014).

Open Access This chapter is licensed under the terms of the Creative Commons Attribution-NonCommercial 4.0 International License (<http://creativecommons.org/licenses/by-nc/4.0/>), which permits any noncommercial use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

