MIDDLE EAST TECHNICAL UNIVERSITY

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

PROCESS CONTROL LABORATORY

EXPERIMENT 2: TEMPERATURE CONTROL WITH ELECTRONIC PID CONTROLLER

I. Objective

The objective of this experiment is to investigate the operation of an electronic PID controller in a closed loop control system and to get familiar with the arrangement of various system elements such as thermocouple as a measuring element and TRIAC as a final control element in a temperature control loop.

II. Information

In this experiment, you will design a controller to keep a heating process at a desired temperature. Block diagram of the overall system is given in Fig. 1.

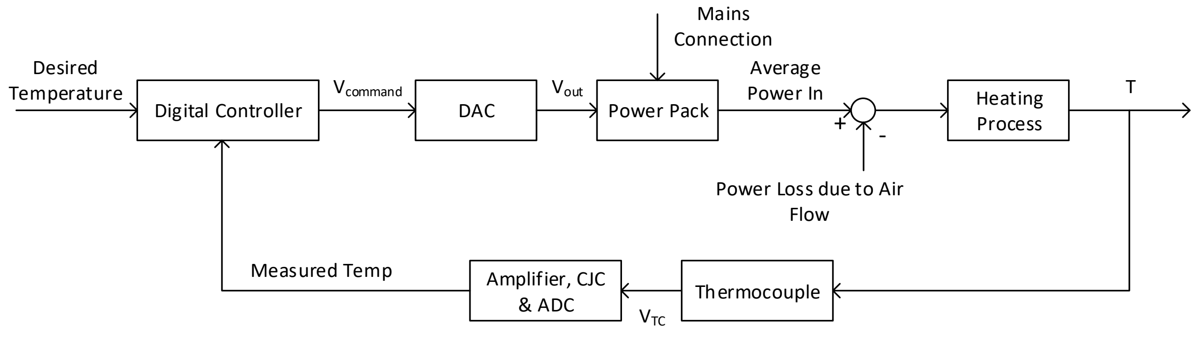

Figure 1. Block Diagram of the System

Brief descriptions of these elements and their input-output signals are as follows.

<u>**Elements**</u>:

Heating Process:

This is the process of heating a wire wound resistor that is in direct contact with the environment (air) and thermocouple (its measuring device) as depicted in Fig. 2. Average power dissipated by the resistor is the well-known Joule loss. Part of the dissipated power is accumulated in the iron mass and raises its temperature while the remaining part is lost as a result of heat transfer to the environment. Some portion of the dissipated power is also transferred to the thermocouple to raise its temperature.

Thermocouple:

A thermocouple is a measuring device which generates an analog emf that is proportional to the difference of temperature between its hot and cold junctions (refer to Exp. 1 for more information). To put it simply, it measures the temperature with some DC offset and multiplies this difference with some gain K (that is, $v_{TC} = K * (T_{Thermocouple} - T_{ref})$). The gain K is a nonlinear function of the temperature difference and depends on the material used in building the thermocouple; however, for a K-type thermocouple it can be approximated by 41 µV/°C for a temperature difference range of $0 - 1000$ °C.

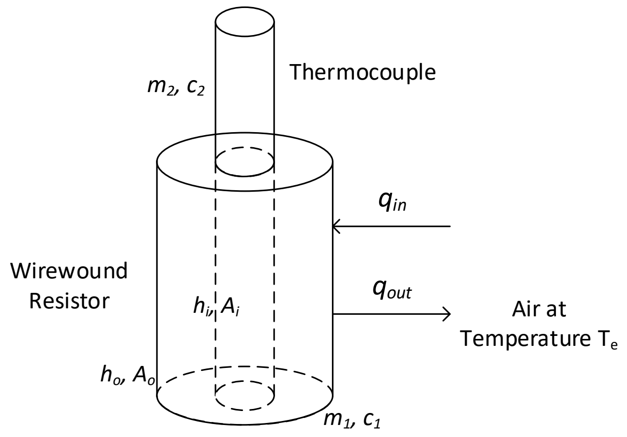

Figure 2. Connection of Wire Wound Resistor and Thermocouple

Signal Conditioning Circuit: (Amplifier, CJC & ADC block)

One of the problems observed with a thermocouple is that it is a relative temperature sensor as its output voltage has an offset voltage relative to its cold junction. In order to overcome this, its cold junction can be kept at a known temperature, which may be impractical in some cases. Alternatively, one could use an absolute temperature measurement device to eliminate this offset as a Cold Junction Compensator (CJC). Another problem caused by a thermocouple is that its resolution is quite low for most of the commercially available Analog to Digital Converters (ADCs). Letting an ADC have M-bit resolution, and letting $v_{refHigh}$ & v_{refLow} be its biasing voltages, voltage resolution of an ADC can be given as,

$$Q = \frac{v_{refHigh} - v_{refLow}}{2^M} = \frac{\text{full span}}{\text{number of intervals}}$$

Let $v_{refHigh} = 3.3\text{V}$, $v_{refLow} = 0\text{V}$ and $M = 13$ bits. Then, $Q = \frac{3.3V}{8192} \cong 0.403\text{mV}$. Without any signal amplification, one would need a temperature difference of ΔT, as given below, to be able to detect any temperature change.

$$\Delta T = \frac{Q\ (\text{V})}{K\ (\text{V}/°\text{C})} \cong 9.83°\text{C}$$

Above result clearly shows that one has to decrease ADC's resolution (by decreasing its full span or increasing its number of intervals). If that is not possible, one can instead amplify the output voltage to match desired resolution. In this experiment, you will be using an IC, MAX31855, which achieves all this conditioning and outputs a 13-bit-long digital word to your controller as your measured process variable in °C.

Digital Controller:

In this experiment, you will be using Arduino® Mega 2560 as your digital controller. It will be programmed using Simulink® Support Package for Arduino® Hardware, examples/getting started videos of which interested readers can find here. The controller will be run on real-time on the board, while the PC connected to it will merely be used to collect real-time data. It is to measure the temperature of your plant in °C and compare it to the (digital) desired temperature set point and send its controller output (digital voltage) to the Digital to Analog Converter to provide its corrective action.

You will be using a controller design procedure based on Controller Output (CO, v_{cmd}) and observe how the measured process value ($T_{measured}$) reacts to the change in CO, and you will be using data from a bump test to approximate the overall system as a First Order Plus Dead Time (FOPDT) model as given by the equation below. Here, plant gain is given as $K_p = \Delta T_{measured}/\Delta v_{cmd}$ (where the $\Delta T_{measured}$ is the temperature difference between the measured temperature just before the bump and the one at steady state), θ_p is the deadtime and T_p is the plant time constant.

$$G_p(s) = K_p \frac{e^{-s\theta_p}}{T_p s + 1}$$

In such a model, FOPDT has the unit of °C/ V and error signal $e(t) = T_{set}(t) - T_{measured}(t)$ has the unit of °C; thus, the controller has to have the unit of V/°C so that the overall transfer function is unitless (°C/°C). Note that this result is also consistent with the parameter units in Table 1.

Controller transfer input-output relationship is given as,

$$V_{cmd}(t) = K_c \left(1 + \frac{1}{T_i} \int e(t)dt + T_d \frac{de(t)}{dt}\right) + CO_{bias}$$

where K_c is the controller gain, T_i is the integral time and T_d is the derivative time. Different from what you are accustomed to in your previous feedback courses is the addition of CO_{bias} to the corrective action. As to why one may need this bias term, consider a P-only controller. This bias value is chosen manually by an operator so that when $e(t) = 0$, $v_{cmd}(t) = CO_{bias}$ achieves a predetermined PV. However, as set point changes (or the disturbance, or environmental temperature) P-only controller may not achieve 0 steady state error for a self-regulating process and thus it is insufficient.

The following table lists the optimal controller parameters for set point tracking, with the optimality criteria being the minimization of Integral of Time-Weighted Absolute Error ($\mathrm{ITAE} = \int_0^\infty t|e(t)|dt$), when a FOPDT plant is to be controlled.

	P	PI
K_c	$\frac{0.2}{K_p}\left(\frac{T_p}{\theta_p}\right)^{1.22}$	$\frac{0.586}{K_p}\left(\frac{T_p}{\theta_p}\right)^{0.916}$
T_i	X	$\frac{T_p}{1.03 - 0.165(\theta_p/T_p)}$

Table 1. Optimal ITAE Controller Parameters for Servo Control Problem of a FOPDT System

DAC and Power Pack:

Digital to Analog Converter used in this experiment is a single channel 12-bit DAC (MCP4725).

Power pack you will use in this experiment is Sauter ESL130 electronic power control unit. Power pack maps the range of input signals (in our case it is configured as $2 - 5\mathrm{V}$) to fully off and fully on pulses of mains AC voltage over a period T. Triac, the AC actuator, is controlled in burst firing mode. That is, t_{on} and t_{off} are always an integer multiple of half of mains' electricity period (integer multiple of $T_{mains} = 10$ msec in Turkey). The overall period of this firing mode is configured to be $T = 4\mathrm{sec}$. You can see the case of triggering for $v_{cmd} = 3.5V$ with an exaggerated mains frequency for clearer drawing. Roughly speaking, those of you familiar with PWM controlled DC motors may consider this as Pulse-Width-Modulated AC signal delivered to the load. The following is the description from the manual of ESL130.

"With the ESL, the heating output is controlled quasi-continuously, i.e. the heating coil is switched on/off in pulses. The control part and the output part are electrically isolated using an opto-coupler. The circuit-breaker is a TRIAC. The switching time is triggered by the zero crossing of the sine wave voltage."

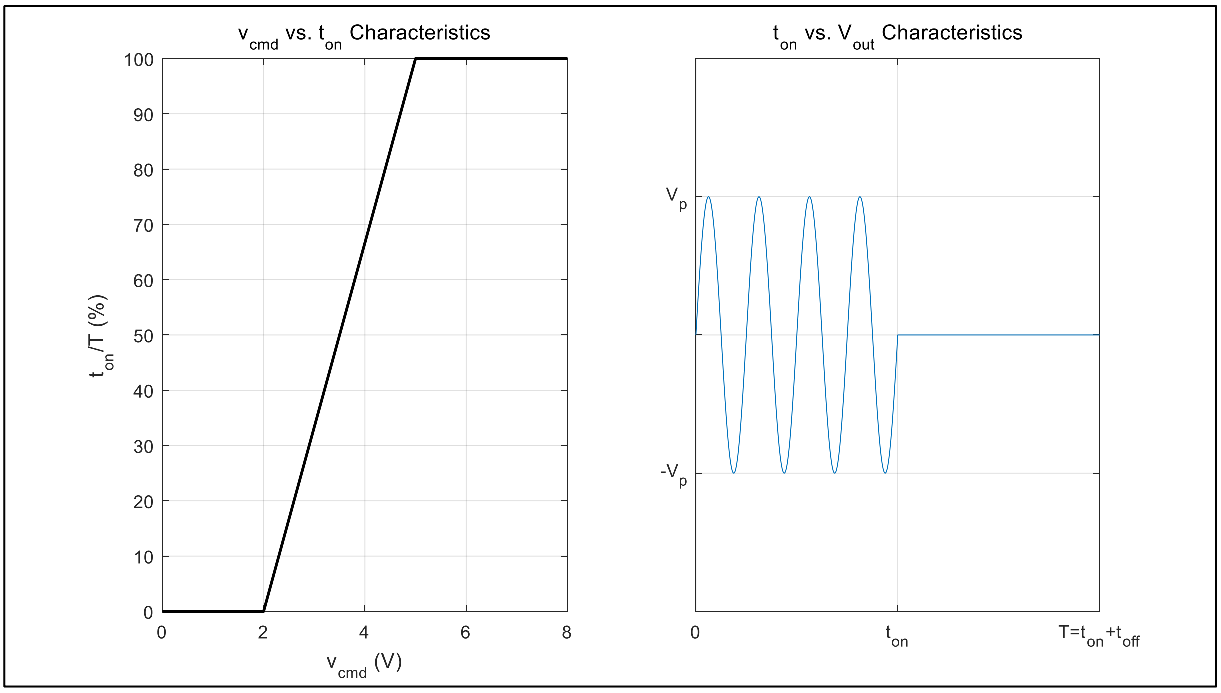

Figure 3. t_{on}/T (%) vs. v_{cmd} Plot and V_{out} vs. Time Plot for $t_{on}/T = 50\%$

It is straightforward to show that average voltage delivered to the load and average power dissipated at the resistive load are given by:

$$V_{rms} = \frac{V_p}{\sqrt{2}}\sqrt{\frac{t_{on}}{t_{on}+t_{off}}}$$

$$P_{avg} = \frac{V_p^2}{2R_L}\frac{t_{on}}{t_{on}+t_{off}} \rightarrow P_{avg} \propto (v_{cmd}-2) \text{ when } 2\text{V} \leq v_{cmd} \leq 5\text{V}$$

Fan:

The fan in this experiment is used to provide air flow around the wire wound resistor in order to cool it. It is used both to make process self-regulating and to act as a varying disturbance. Without the fan, heat loss to the environment is negligible and the heating process becomes an integrating one; hence, we need to provide some sort of airflow to make it lossy (and equivalently self-regulating) in order not to damage the equipment. Changing air flow speed has the equivalent effect of varying disturbance; hence, we can observe the effectiveness of the controller under different disturbance levels.

It is possible to obtain a lumped parameter model for the heating of two masses (the resistor & the thermocouple). Its transfer function would be equivalent to a second order RC circuit (with two time constants). However, it appears that switching period of the TRIAC is comparable to both of the time constants ($T \cong \tau_1$ and T is not much less than τ_2), resulting in an apparent initial dead time. That is why, a FOPDT model is to be used.

III. Preliminary Work

1. Redraw the block diagram in Fig. 1 showing the units of all signals. Also indicate whether a signal is digital (D) or analog (A).
2. What is our control objective? Explain what each of the following corresponds to in our system: Process Variable, Measurement Sensor, Measured Process Variable (PV), Set Point (SP), Controller Output (CO), Final Control Element (FCE), Manipulated Variable (PV), Disturbances (D).
3. Download SysIdDataExp2.mat from ODTUClass. It contains experimental data from a bump test when the system is at steady state at room temperature for a desired temperature of $T \cong 220°\text{C}$. Plot the input/output vs. time graphs. Estimate the FOPDT parameters of this system. Show your work.
4. **(BONUS)** Remove the initial steady state values from the input & the output. Start MATLAB® System Identification app (with the command *systemIdentification*). Import the time domain data (with starting time 0 and sampling time 0.1 sec) and then click on Estimate → Process Models. Enter your initial guess from the previous step, and some reasonable bounds to the variables. Then use this app to estimate the model parameters. Note that if parameter estimation gets stuck at a boundary value, you may have to tinker with that end point (decrease/increase for a lower/upper end bounds).
5. Simulate the identified plant. You may fix the simulation time step to 0.1 sec. Your system should closely match the behavior of the plant. That is: for $v_{cmd} = 2\text{V}$ it should remain at constant non-zero environmental temperature; it should treat $v_{cmd} > 5\text{V}$ as $v_{cmd} = 5V$ and step time/amplitude should match the experimental ones.
6. Plot the simulated and experimental data on the same figure. You may have to save your simulated (scope) data to workspace.
7. Calculate the optimal ITAE controller parameters for servo control (for both P and PI modes of control).
8. Simulate your plant with a P-only controller $K_c = K_{c,optimal}$ for $T_{set} = 220°\text{C}, 200°\text{C}$ and $240°\text{C}$ (T_{set} should change sequentially in one simulation, and changes should occur after reaching steady state for current set point). You may use the PID Controller block in Simulink®; however, it should not be sufficient by itself to achieve controller equation given in the information section. Comment on the results. What is the reason for observing an overshoot (not in the sense of an underdamped system) for $T_{set} = 180°\text{C}$?
9. Repeat part 7 by using a P-only controller with $K_c = 2K_{c,optimal}$. Comment on the differences.
10. Repeat part 7 with a PI controller ($K_c = K_{c,optimal},\ T_i = T_{i,optimal}$) and ($K_c = K_{c,optimal},\ T_i = 0.5T_{i,optimal}$). Comment on the results.

IV. Experimental Work

1. Start MATLAB. Navigate to User\Desktop\Exp2 and open the Simulink file Exp2.slx. You should see Simulink model shown in Fig. 4, which is configured to be run externally on Arduino Mega 2560. Once you click run, Simulink Coder will generate the C source files according to your model and then generate & deploy the executable application to run on target hardware. The values you will observe on the Scope will be those transferred from Arduino Board, and you will use Dashboard blocks to send commands to the board (which will enable you to alter variable values without recompilation).

   You can turn the controller on/off by toggling the top left switch, and update the setpoint/controller parameters during simulation by entering their values into Edit Field dashboard blocks. Use the Scope block to view measurements and use its Configuration Properties for proper time and temperature scaling.

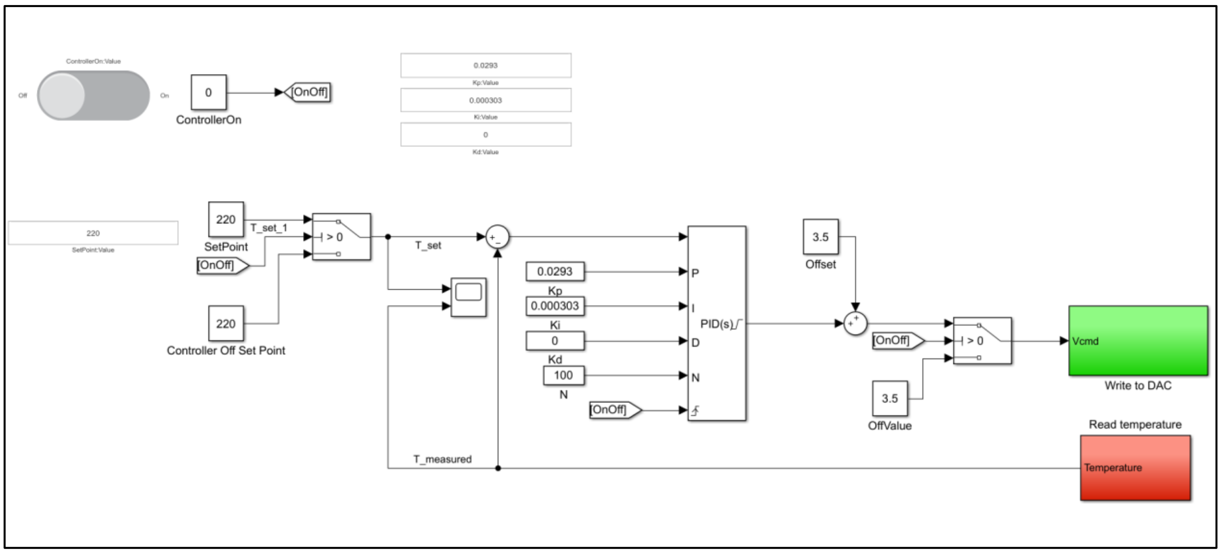

Figure 4. User Interface

2. Now, adjust the set point as 240 (℃), make sure the controller is off and the fan is configured to provide minimum air flow. Have your TA check the $K_{c,optimal}$ value you found in preliminary work step 7 and also enter that. Click on Run to deploy your application. When you start seeing live measurements from the system, turn on the controller using the toggle switch and **manually** power up the Power Pack and the fan & record when you start the controller.
3. Once the set point is reached, change the set point to 220 (℃) and then to 200 (℃).
4. **Manually** increase the amount of airflow to its maximum and wait for the temperature to reach its steady state. Then, adjust the set point to 220 (℃), and 240 (℃). Turn off the controller (with the slider switch), stop the simulation and rename the workspace variable named ScopeData1 into ResultsPart1. Bring the amount of airflow to its minimum.
5. Repeat steps 2-4 with $K_c = 2K_{c,optimal}$. This time rename the ScopeData1 variable into ResultsPart2.
6. Do these for ($K_c = K_{c,optimal},\ T_i = T_{i,optimal}$) and ($K_c = K_{c,optimal},\ T_i = 0.5T_{i,optimal}$).
7. Save the relevant workspace variables for your report.

V. Results and Discussion

1. Provide all the data you obtained from the experiment with proper figure naming & numbering, and axis labeling.
2. What is the main reason for obtaining non-zero steady state error when $T_{set} = 220$℃ with fan airflow at its minimum when you use P-only controllers? What about when the airflow is at its maximum?
3. Calculate the apparent time constants for two P only controllers. Also calculate the steady state error values (for each set point & P controller pair). Give your response in tabular form. Discuss how changing the proportional gain affects these.
4. What are time (i) rise time, (ii) settling time and (iii) percent overshoot of the two PI controllers? Which one of these controllers is more aggressive?
5. You may be familiar with the terminology that T_i is sometimes called as the reset time. Perform a web search for the terms manual & automatic reset for PID controllers and comment, on this setup, how the integral term achieves automatic reset.

VI. Conclusions

Write down your general conclusions and comments on temperature control with PI controllers.