Lab 4: Summary

<u>Tasks for Lab 4:</u>

1A-1C: Open-loop hardware experiment.

1D: Open-loop simulation (See Lab 1)

2A: PI Controller design based on a first-order plant [derive by yourself] (See Tutorial 2)

2A: PI Controller closed-loop simulation (Similar to Lab 2)

2B: PI Controller closed-loop hardware experiment.

<u>Outcomes:</u> With minimal starting material, you will: plan, design, simulate and implement a PI control system on hardware.

<u>Report Requirements:</u>

1. Labs 4 and 5 will be combined into a single report worth 15% of your ACS grade.

2. Hardware will not be available outside of scheduled ACS lab hours. Failure to complete the tasks will result in a substantial number of report marks being inaccessible.

3. Maximum of three students per group. All students must attend the same lab session. Groups must be fixed between Lab 4 and Lab 5. Due to the work required to complete the lab, students who do not attend the lab with their group will not receive a mark for the report.

Labs 4 and 5 will be tight on time. To support your group members, you must be punctual

Lab 4: Tasks

Task 1: Open-Loop Empirical Study

- Interact with the plant input and output.
- Estimate the plant transfer function by analysing the open-loop experimental results.

1A: Read the temperature sensor (LM335)

- The analog input will return a number [0 1023].

Arduino:

1. Connect jADC to a pin: A0-A5.
2. Connect jVCC to the 5V pin.

Simulink

- Simulink Support Package for Arduino Hardware: Common: Analog Input. Set pin number.
- Convert the analog input [0 1023] to a [0 5] Voltage.
- Convert Voltage to millivolt (mV).
- °K = mV / 10
- °C = °K – 273.15
- Finishing time: inf. External

- Check your COM port in Arduino IDE and change the COM port setting.
- Run the program. Is your measured temperature approximately (within a few degrees) what you expect?
- You are now able to interact with the plant output.

1B: Send an input to the Arduino

Arduino

1. Connect jPWM to a pin: PWM 2-13.

Simulink

The Arduino PWM accepts an input between [0-255] corresponding to a PWM duty cycle of [0 100] %.

1. Simulink Support Package for Arduino Hardware: Common: PWM. Set pin number.
2. Add a constant 0.5 (representing a 50% duty cycle).
3. Convert this to a PWM duty cycle.
4. Add the Math Operations: Rounding Function: Round and round the PWM duty cycle before connecting it to the Arduino PWM channel.

- Confirm your solutions with a lab demonstrator before proceeding.
- You are now able to interact with the plant input.

1C: Estimate the Plant

1. Study Figure:

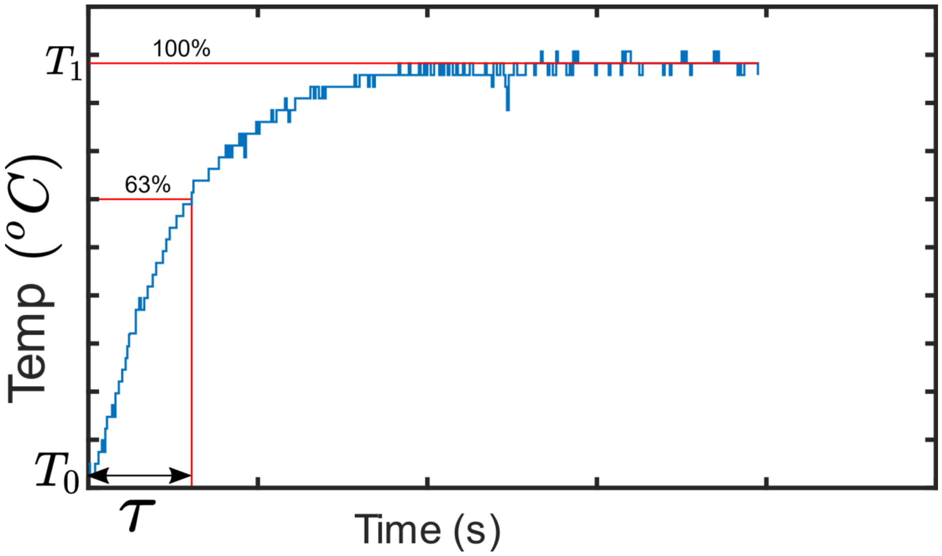

Figure: An example of step response output

2. Run the program and connect the power supply.
3. Collect the results:

Parameter	Measured value
T_0	
T_1	
τ	
a	$a = \frac{T_1 - T_0}{0.5} =$ *(0.5 = 50% Duty Cycle = 0.5*5V dropped across power resistor)

4. Scope Settings: Time: Time Span = 2000
5. Save your plot and your MATLAB files.
6. Calculate your empirical plant transfer function:

Plant Transfer Function G(s)	$G = \frac{a}{\tau s + 1} =$

7. Turn off the power at the wall – let the board cool back to room temperature.

1D: Open-Loop Simulation

1. Create a new Simulink file.
2. Simulate the open-loop response of the estimated plant using the same input.
3. Confirm that your simulation result is approximately what you expected to see.

- Save your MATLAB file.

Task 2: Closed-Loop PI Control System

- Design a PI Control System where the temperature of the heated resistor tracks a user-provided reference signal and satisfies performance criteria.
- Confirm closed-loop system performance and safety in simulation before proceeding to demonstrate the system in a hardware experiment.

2A: PI Controller Simulation on Simulink

1. Create a new Simulink file.
2. Add the plant that was estimated in Task 1C.
3. Choose a reference signal between [40°C, 70°C].
4. Derive the K_c and τ_i PI parameters required for your plant for a given tuning parameter, ω_n. Assume ξ = 0.707. Use a second-order desired polynomial: $s^2 + 2\xi\omega_n s + \omega_n^2$
5. Choose one of the two PI control structures:
    a. Proportional and Integral on E(s)
    b. Proportional on Y(s) and Integral on E(s).
6. Add the Simulink Block: Discontinuities: saturation block. Saturate U(s) between [0,1].
7. Implement this in a simulation and confirm that the results match your expectations.
8. Tune your control system by adjusting ω_n so that the maximum overshoot is <=20% and that the system takes <=600 seconds to reach steady-state.
9. Plot your reference, control signal, and output signal.
10. Confirm your solution with a lab demonstrator before proceeding.
11. Save your MATLAB files.

2B: PI Controller on Arduino

1. Create a new Simulink file.
2. Use the same reference signal as in Task 2A. Use the blocks you created in Task 1A and Task 1B.
3. Add the Simulink Block: Discontinuities: saturation block. Saturate U(s) between [0,1].
4. Use the same controller parameters and controller structure from Task 2A.
5. Confirm your solution with a lab demonstrator before proceeding.
6. Plot your reference, control signal and output signal.
7. Save your plots
8. **<u>Save your MATLAB, Simulink, and plot pictures on the network drive.</u>**
9. Once you have completed the Lab, in the Arduino IDE, select the Arduino Mega 2560 board and upload the blank sketch.