EE 407
Process Control Laboratory

Experiment 5
Heat Flow Characteristics and Temperature Control

Department of Electrical and Electronics Engineering
Middle East Technical University

November 29, 2018

1 Objectives

In this experiment, you will first identify the parameters of a mathematical model describing the dynamics of the Heat Flow setup shown in the cover page. To this end, you will rely on the bump test method. Subsequently, you will attempt to control temperature of a certain point in the chamber by making use of an on-off controller and a PID-based controller. Finally, you will investigate some practical process control concepts, such as integral anti-windup, to gain some insight about real world issues.

2 Prerequisites

To successfully carry out this laboratory, the prerequisites are

- to be familiar with the basics of bump test based system identification,
- to be familiar with the fundamentals of the design procedures for on-off and PID control algorithms,
- to be familiar with the wiring and operating procedure of the Heat Flow plant with the Q8-USB data acquisition device, as discussed in [1],[3].

3 References

[1] Heat Flow Setup - User Manual, Quanser.
[2] Student Workbook for Heat Flow Experiment, Quanser.
[3] Q8-USB Data Acquisition Device - User Manual, Quanser.
[4] N. Nise, *Control Systems Engineering, 7th edition*. John Wiley & Sons, 2015.
[5] K. Ogata, *Discrete-time Control Systems, 2nd edition*. Englewood Cliffs, N.J.: Prentice-Hall, 1987.

4 Experimental Setup

The experimental setup from Quanser Inc. basically comprises of a chamber equipped with three temperature sensors located equidistantly. There is a coil based heater and a blower at the one end of the chamber that is used to transfer heat conductively. The system has a built-in amplifier to deliver power to the heater and blower and the amount of power is controlled using analog signals. There is also a tachometer mounted on the blower to measure the speed of the fan.

4.1 Main Components

This laboratory is composed of the following hardware and software components:

- **A fiberglass chamber equipped with related components to examine heat flow:** Quanser Heat Flow
- **Data Acquistion Board:** Quanser Q8-USB
- **Real Time Control Software:** The QuaRC-Simulink configuration
- **A Computer to Run Matlab-Simulink and the QuaRC software**

Student Name:
Student ID:

5 Preliminary Work

5.1 Mathematical Modeling

1. The block diagram of the experimental setup demonstrated in Fig. 1 depicts the inputs and outputs of the system. The blower voltage, V_b, and the heater voltage, V_h, affects how the chamber temperature changes with respect to the ambient temperature, T_a.

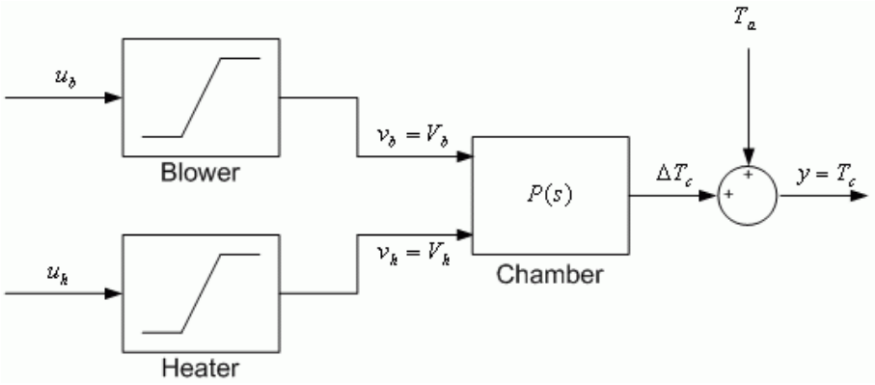


Figure 1: Dynamics of the Heat Flow setup

The temperature of the chamber at the n$^t h$ sensor can be described by

$$\frac{d}{dt}T_n(t) = f(V_h, V_b, T_a, x_n) \tag{1}$$

where x_n is the distance between the nth sensor and the heater.

Determining the function $f(\cdot)$ in (1) to account for the exact thermodynamic phenomenon is a challenge beyond the scope of this experiment. Therefore, we assume that a first-order dynamical model is sufficient for the purpose of designing a temperature controller. Hence, consider the following transfer function

$$T_n(s) = \frac{K_n}{\tau_n s + 1} V_h(s) \tag{2}$$

where K_n is the steady-state gain and τ_n is the time constant of the nth sensor.

5.2 Bump Test Identification

2. The bump test is a simple identification procedure based on investigating the step response of a stable system. It is carried out in the following way: a constant input is applied to the system then the stable system reaches to an equilibrium. Thereafter,

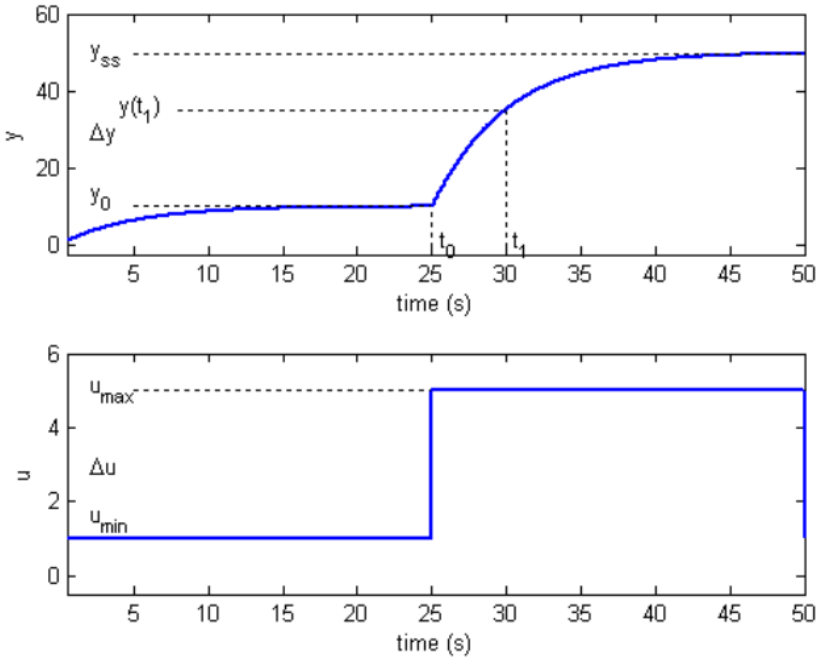


Figure 2: Input and output signals obtained in a bump-test

the input is changed rapidly to a new level while the output is being recorded. In Fig. 2, you can see the findings acquired during a typical bump test.

The step response shown in Fig. 2 is generated using the transfer function

$$\frac{Y(s)}{U(s)} = \frac{K}{\tau s + 1} \tag{3}$$

with $K = 10$ and $\tau = 5$.

The input signal, u, is a step that begins at time t_0. The input signal has a minimum value of u_{min} and a maximum value of u_{max}. The resulting output signal is initially at y_0. Once the step is engaged, the output eventually settles to its steady-state value y_{ss}. By considering the output and input signals, the steady-state gain can be computed as

$$K = \frac{\triangle y}{\triangle u} \tag{4}$$

where $\triangle y = y_{ss} - y_0$ and $\triangle u = u_{max} - u_{min}$.

In order to find the model time constant, τ, the output signal at $y(t_1)$ must be measured. It is defined as

$$y(t_1) = (1 - e^{-1})\triangle y + y_0 \tag{5}$$

where $t_1 = t_0 + \tau$. Hence, time constant can obviously be obtained by $\tau = t_1 - t_0$.

5.3 Time Response Characteristics of Second Order Systems

3. The block diagram shown in Fig. 3 is a general unit-gain feedback system with a controller, C(s), and a plant, P(s). The measured output, Y(s), is supposed to track the reference signal, R(s), and the tracking has to satisfy certain specifications.

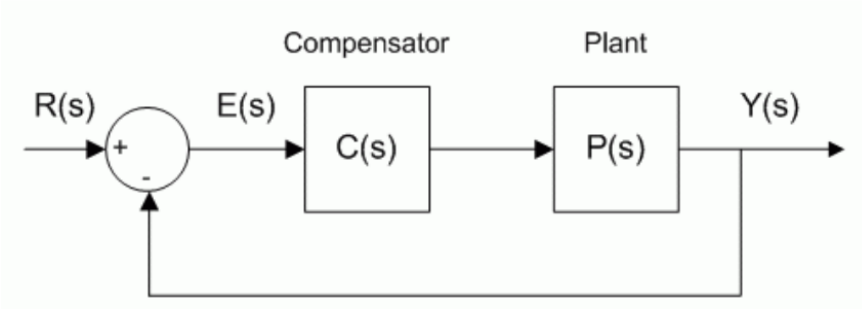


Figure 3: Block diagram of a negative unit feedback close loop system

The output of the given system can be written in Laplace domain as

$$Y(s) = \frac{C(s)P(s)}{1 + C(s)P(s)} R(s). \tag{6}$$

4. Now, find the error transfer function E(s) shown in Fig. 3 in terms the reference signal $R(s)$, the plant transfer function $P(s)$ and the controller C(s).

5. By using the final-value theorem, calculate $\lim_{t\to\infty} e_{ss}(t)$ when the controller is $G_c(s) = K_p$ and the input is a unit step of amplitude R_0. Recall that the transfer function of the plant is taken to be

$$P(s) = \frac{K}{\tau s + 1}. \tag{7}$$

6. Based on the steady-state error you found in the previous step state the type of the system (justify your answer).

7. When the first order plant in (7) is controlled by a PI controller, a zero appears in the closed-loop transfer function. However, if the closed-loop zero is far away from the closed-loop dominant poles, then the response approaches to that of a two-pole system (without a zero), [4] (Section 4.8, System Response with Zeros). Thus, the closed-loop transfer function can be casted into the standard form given by

$$\frac{Y(s)}{R(s)} = \frac{\omega_n^2}{s^2 + 2\omega_n \xi s + \omega_n^2} \tag{8}$$

where ω_n is the natural frequency and ξ is the damping ratio.

8. Assume we apply a unit step input to the closed-loop system in (8), formulate the percent overshoot (PO) and the peak time (t_p) in terms of the parameters in (8). (You don't need to derive the formulas.)

5.4 Performance Requirements for the Heat Flow Experiment

9. In this experiment, time-domain performance requirements are specified as follows.

$$e_{ss} = 0, \quad t_p = 15\,\text{sec}, \quad PO(\%) = 15.0. \tag{9}$$

5.5 Temperature Control

5.5.1 On-Off Control

10. An on-off control scheme can be implemented using a variety of switches and the design procedure does not rely on the mathematical model of the plant. The block diagram depicted in Fig. 4 shows the on-off controller that will be used to regulate the temperature in this experiment. This controller is implemented using a relay switch. Also shown in the diagram, is the nonlinear dynamics of the heater actuator, whose output is limited between 0 V and 5.0 V.

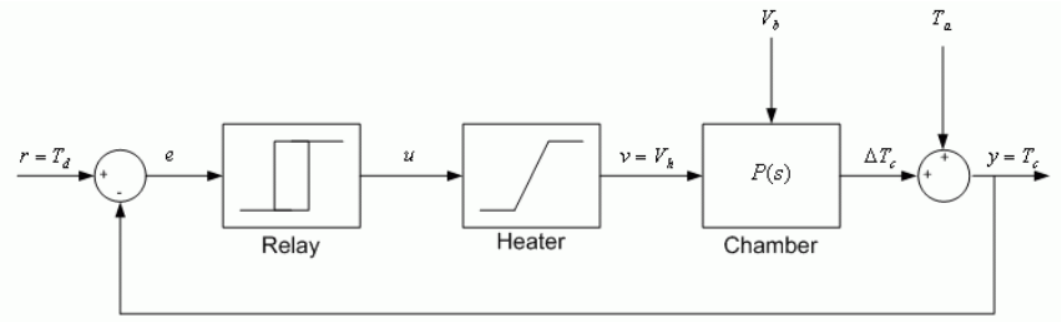


Figure 4: An on-off controller realized with a relay block

11. The relay characteristics is illustrated in Fig. 5. When ON, the relay switch outputs a voltage of $V_{h,on}$ to the heater. When OFF, it outputs a value of $V_{h,off}$ to the heater (typically this is set to 0 V). The hysteresis width, $\triangle T_h$, can be adjusted in accordance with the performance specifications. Changing the width of the hysteresis affects the performance of the control, as will be investigated in details in the lab.

5.6 Proportional-Integral Control

12. The proportional-integral (PI) controller to be utilized has the following structure

$$V_h(t) = K_p \left(b_{sp} T_d(t) - T(t)\right) + K_i \int \left(T_d(t) - T(t)\right) dt \tag{10}$$

where K_p is the proportional control gain, b_{sp} is the set-point weight, K_i is the integral control gain, $T_d(t)$ is the desired temperature, $T(t)$ is the measured chamber temperature at a particular sensor and $V_h(t)$ is the heater control voltage. The block diagram of the controller is illustrated in Fig. 6.

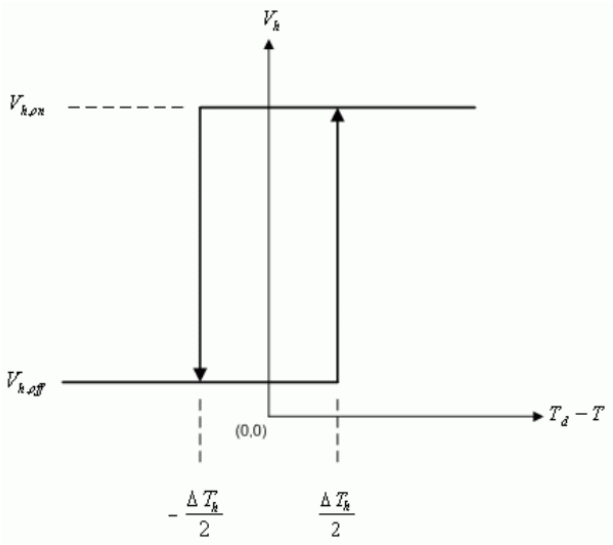


Figure 5: Input-output characteristics of the relay

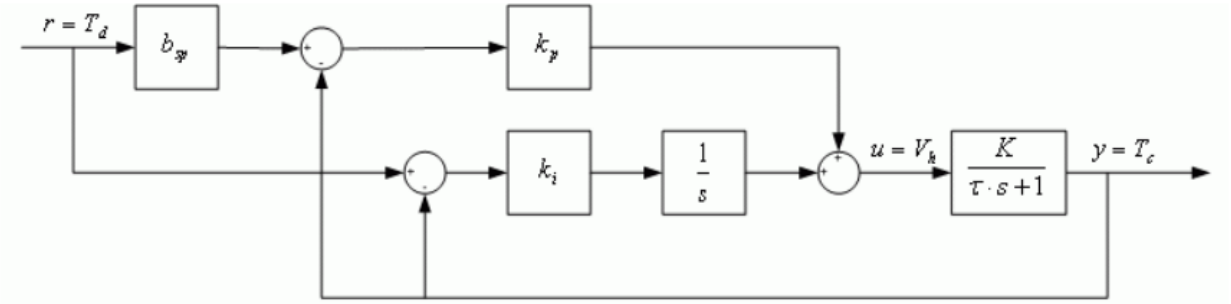


Figure 6: Block diagram of a close loop system with PI controller

Remark: In this experiment, the temperature is controlled about a single sensor (either 1, 2, or 3). From this point on, the T_n notation for measured temperature at the nth sensor will be dropped. The $T(t)$ denotes the temperature measured at a particular sensor. Typically this will be sensor 1 but it can be chosen for any.

13. Find the closed-loop transfer function, $T(s)/T_d(s)$, using the time-domain PI control in (10), the block diagram in Fig. 6, and the process model in (7).

14. The resulting closed-loop transfer function should have the same structure as the standard second-order system given in (8). The denominator of (8) is called the standard characteristic equation. Find the control gains K_p and K_i in terms of ω_n, ξ, τ, K. (Take $b_{sp} = 1$.)

15. Using the equations in Step 8 express the natural frequency and the damping ratio in terms of percentage overshoot and peak time.

16. Calculate the minimum damping ratio and natural frequency required to meet the requirements revealed in Step 9.

17. For the time being, you are able to analytically calculate the PI controller gains once you identify the plant model. As an exercise, assume that you conducted the bump test and observed the system parameters to be $K = 0.8$ and $\tau = 50sec$. Calculate PI controller parameters to satisfy time-domain requirements.

18. Find the error transfer function of the closed-loop system when the PI controller is employed.

19. Evaluate the steady-state error to a unit-step input.

5.6.1 Anti-Windup

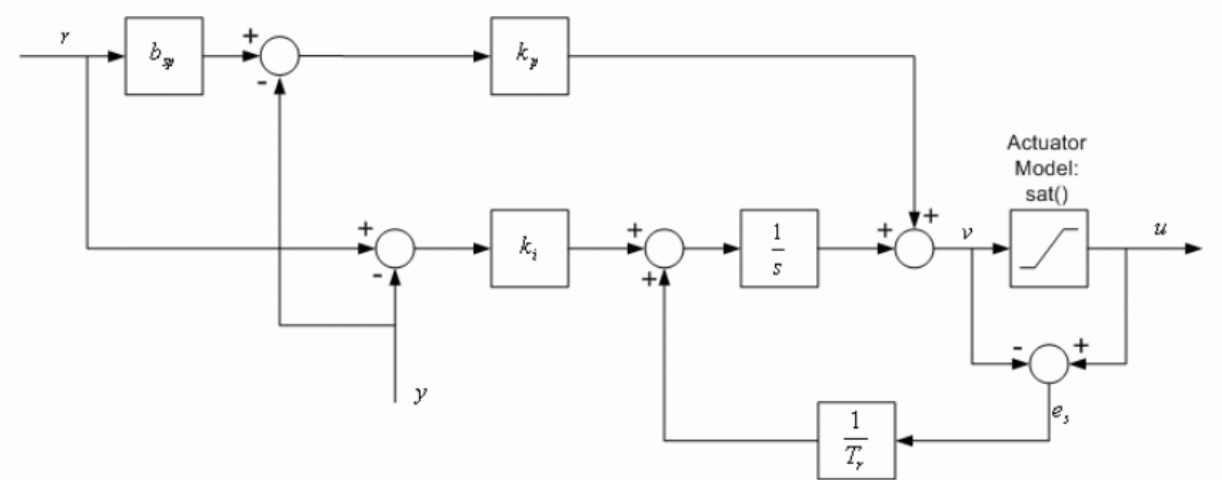


Figure 7: PI control scheme with anti-windup

20. The heater actuator on the setup can be modeled by the following piecewise continuous function

$$u(t) = \begin{cases} 5 & 5 < v(t) \\ v(t) & 0 < v(t) \leq 5 \\ 0 & v(t) \leq 0 \end{cases} \tag{11}$$

where $v(t)$ is the ideal control voltage and $u(t)$ is the actual voltage applied.

This nonlinearity can cause substantial problems when used in a control loop with an integral action. If the controller saturates, the integrator may begin to drift away yielding a degraded control performance. This phenomenon is called integrator wind up.

21. An integrator anti-windup scheme such as the one shown in Fig. 7 can be used to mitigate this issue. If the control signal does not saturate, then the extra feedback loop with the time constant T_r is inactive because $u = v$. When the controller output saturates, the extra feedback loop drives the saturation error, e_s, to zero and causes

the integrator to output a value just at the saturation limit. This means that the control signal will decrease from the saturation limit as soon as the control error goes negative.

The windup protection is governed by the integrator reset time, T_r. There is less protection against windup if T_r is made large (e.g. none if $T_r = \infty$).

22. In Fig 8, the effect of using an anti-windup scheme is demonstrated. The dashed blue line represents the response without windup protection and the solid red line is the response with anti-windup. When wind up occurs, the integrator builds up a lot of energy and causes there to be a larger overshoot in the response. However, with anti-windup the input to the integrator is decreased when the controller saturates and the overshoot is decreased significantly. Anti-windup becomes especially important in slower systems with large time constants, such as the Heat Flow, because the integrator has more time to wind up.

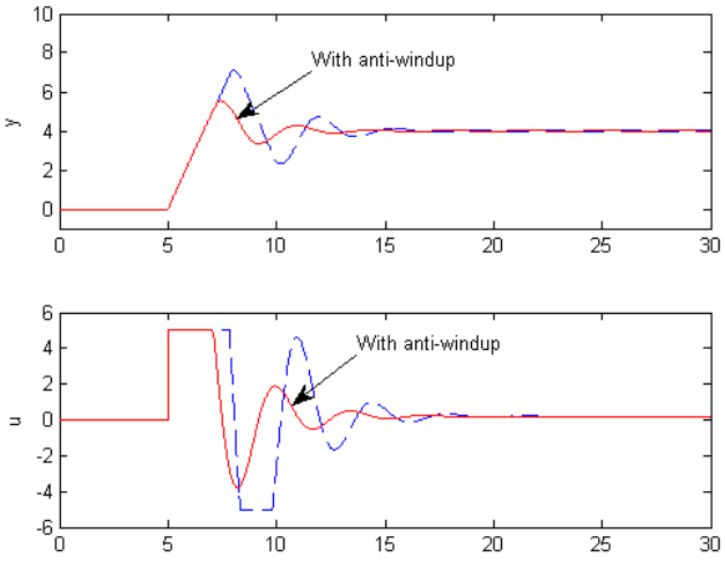


Figure 8: Effect of using anti-windup scheme on the output response

6 In-Lab Experimental Procedure

Before starting the experimental procedure, have your assistants check the below box.

Write down the K_p and K_i formulas you obtained in Step 14 (in terms of $\omega_n, \xi, \tau_n, K_n$):

Assistant's Signature

6.1 Bump Test Modeling

The *q_hfe_open_loop* Simulink diagram shown in Fig. 9 is used to feed open-loop voltages to the heater and blower and measure the corresponding chamber temperatures and fan speed. The *Heatflow* subsystem contains QUARC blocks that interface with the Heat Flow hardware. The *To Host File* block is used to save the temperature sensors readings and the blower and heater input voltages to a MATLAB .mat file. The experimental data saved in this file can then be accessed in MATLAB to plot the response and take measurements.

6.1.1 Getting Started

Follow this procedure to run the open-loop controller on the Heat Flow device:

1. Load the MATLAB software.
2. Browse through the *Current Directory* window in MATLAB and find the folder that contains the lab files, e.g. *q_hfe_open_loop.mdl*.
3. Double-click on the *q_hfe_open_loop.mdl* file to open the Simulink diagram shown in Fig. 9.
4. Go to the *Current Directory* window and open the *setup_hfe.m* file.
5. Run the script to load the appropriate sensor calibration gains in the MATLAB workspace.
6. From the model, open the *Heatflow* subsystem and investigate the structure shown Fig. 10. This subsystem implements the input-output operations between the Q8-USB Data Acquisition Board (DAQ) and the the rest of the experimental hardware.

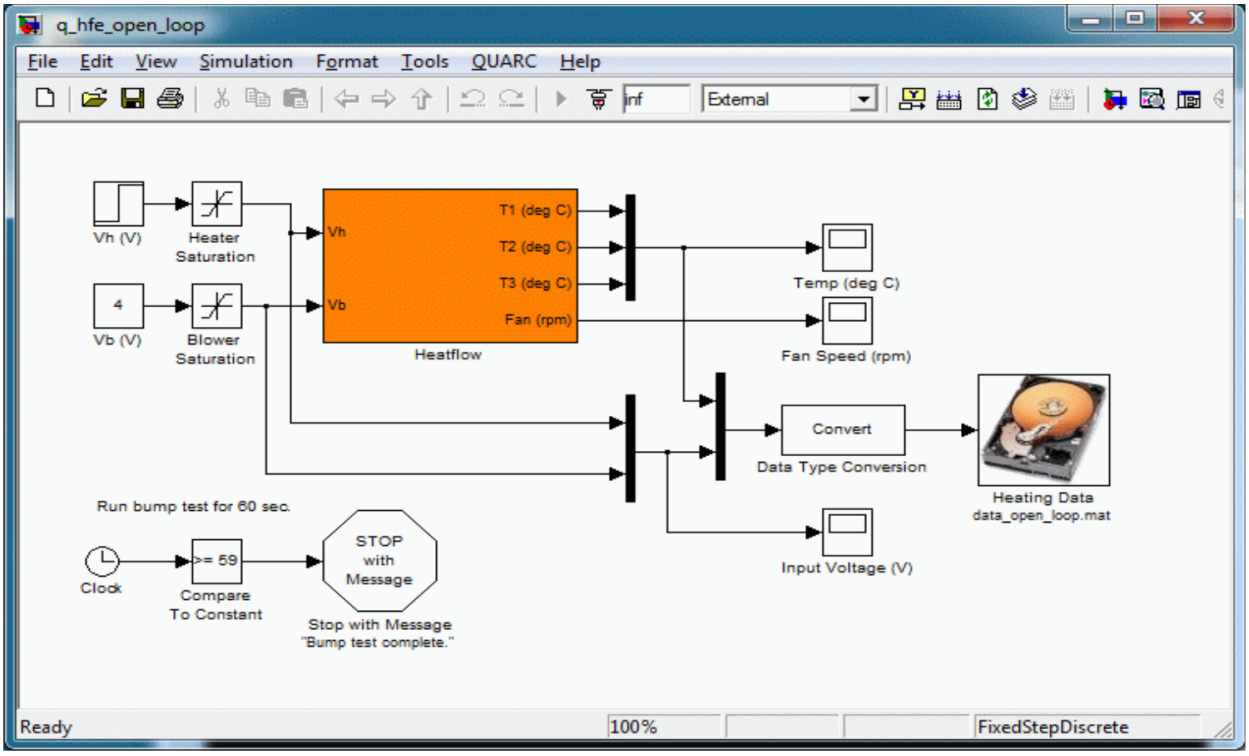


Figure 9: Simulink diagram used to run HFE in open-loop using QUARC.

7. **Configure DAQ:** Double-click on the HIL Initialize block inside the *Heatflow* subsystem and ensure it is configured for the DAQ device that is installed in your system which is Q8-USB.

8. Ensure that all of the connections have been made as instructed Reference Heatflow User Manual.

9. Power on the Heat Flow.
   Remark: The blower should start when the power is engaged.

10. You are now ready to build and run the Simulink model.

6.1.2 Running the Bump Test

This procedure demonstrates how to perform the bump test with the heater:

1. You've already setup the MATLAB workspace by the steps discussed in Section 6.1.1.

2. Click on *Quarc* |*Set default options*.

3. Click on *Quarc* |*Build* to compile the *q_hfe_open_loop* Simulink diagram.

4. Make sure Vb = 4 in the Simulink diagram.

5. Go to Quarc |Start to begin running the controller.

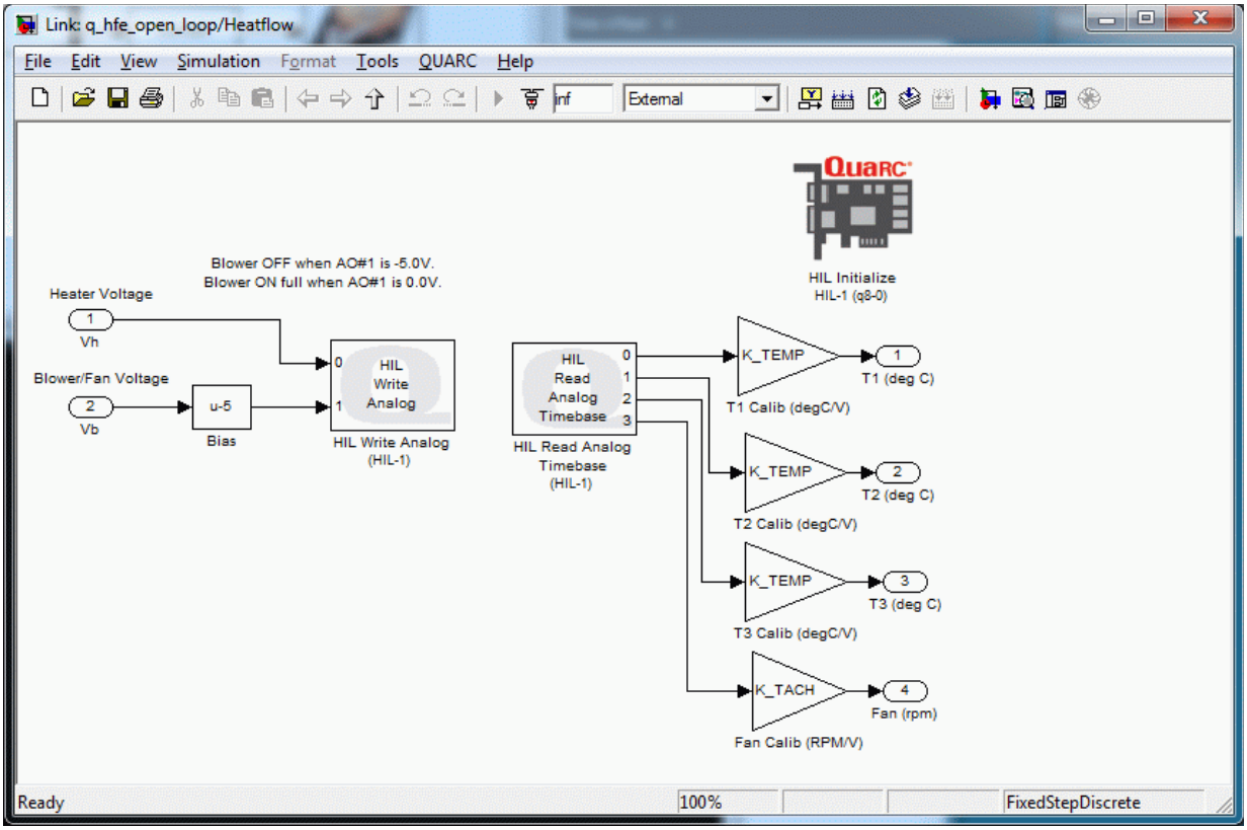


Figure 10: Heatflow subsystem

6. A 4V step is applied to the heater 5 seconds after the control begins running.

7. The scopes are expected to display signals similar to the ones in Figures 11a, 11b, and 11c. When the heater step is applied the temperature in the chamber begins to increase.

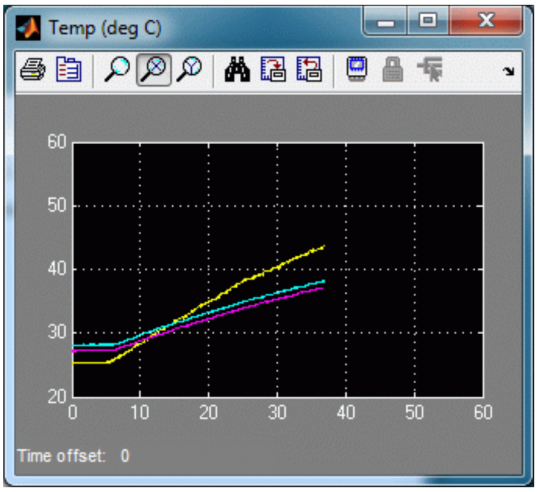
8. Wait for the model to stop automatically after 299 seconds.

9. The temperature sensor readings and blower and heater voltages are saved in the .mat file *data_open_loop.mat*. Copy this .mat file to a USB flash drive since you will examine this data in your report.

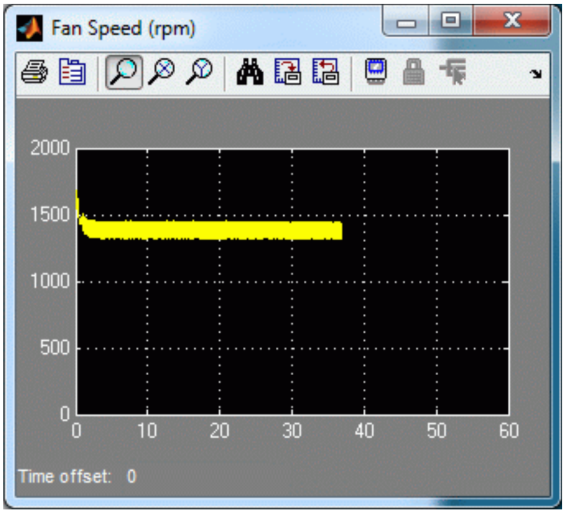
10. As you observe the step response of the Heat Flow setup resembles a first-order response. Hence, using the bump test method discussed in the Preliminary Work, find the steady-state gain and time constant observed at the T_1 sensor. (You may investigate the response by the relevant Scope block.)

6.2 On-Off Temperature Control

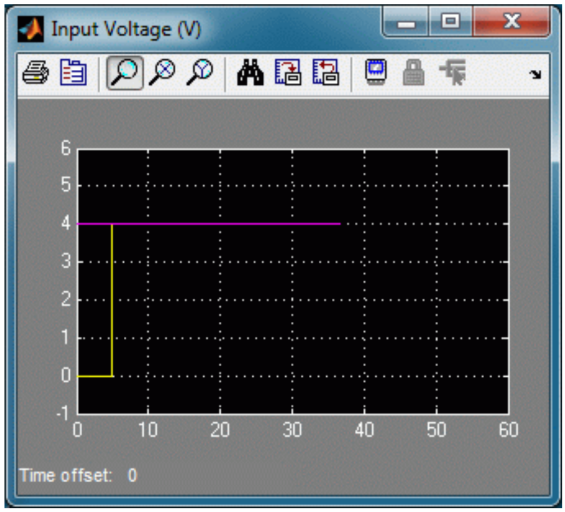
The *q_hfe_on_off* Simulink model shown in Figure 12, is used to run the on-off controller discussed in Section 5.5.1. The on-off controller is implemented using a Simulink *Relay* block.



(a) Temperature bump test response. T_1 is yellow, T_2 is purple, and T_3 is cyan.



(b) Fan speed in blower



(c) Heater (yellow) and blower (purple) voltage

Figure 11: Bump test results

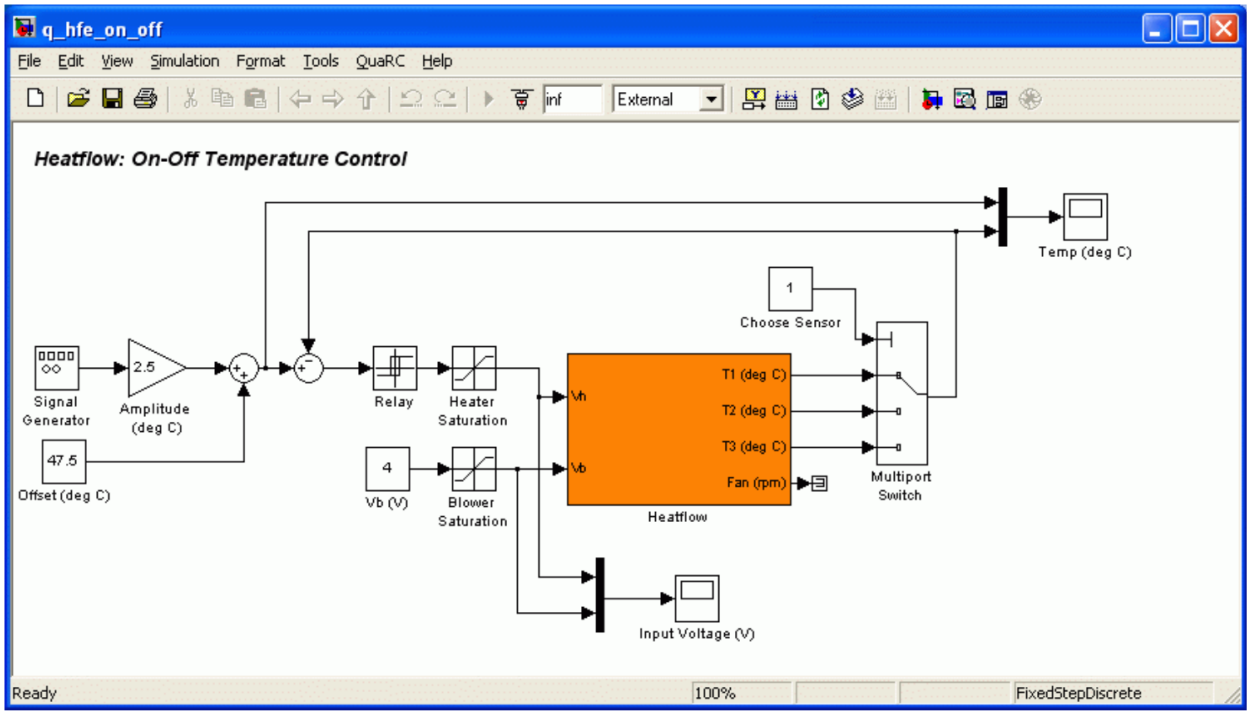
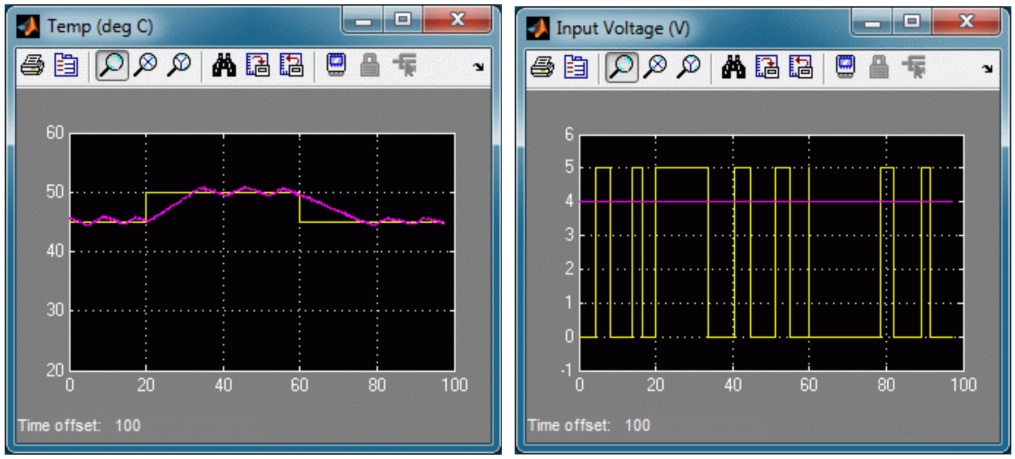


Figure 12: Simulink diagram to run on-off controller on Heat Flow using QUARC.

This procedure demonstrates how to run the on-off on the HFE:

1. Open the *q_hfe_on_off* Simulink model.
2. Click on *Quarc |Set default options.*
3. Click on *Quarc |Build* to compile the QUARC controller.
4. Ensure that the *Amplitude* (deg *C)* block is set to 2.5 and the *Offset (deg C)* to 47.5. This square setpoint signal will vary between 45.0 ^{o}C and 50.0 ^{o}C.
5. In the *Signal Generator* block, set the *Frequency* of the square wave to 0.02 Hz.
6. Set the *Vb (V)* block to 4.0 V. The blower will be running at a constant rate while we vary the voltage to the heater, using the on-off control, to regulate the temperature at the setpoint.
7. Set the *Choose Sensor* block to 1 in order to control the temperature about the T_1 sensor.
8. Ensure that the on-off controller has a hysteresis width of 1.0 ^{o}C, $V_{h,off} = 0$, and $V_{h,on} = 5.0$ V by clicking on the *Relay* block.
9. Go to Quarc |Start to begin running the controller.



(a) T_1 on-off controller response (b) On-off controller signals

Figure 13: Responses for on-off control

10. The typical on-off control temperature response for sensor T_1 is shown in Fig. 13a and the heater and blower voltages are shown in Fig. 13b. The heater voltage is the yellow trace and the purple plot is the blower voltage.

11. After each run, the last 100 seconds of the desired and measured temperature readings and the heater and blower voltages are saved to the MATLAB workspace under the variables *data_temp* and *data_u*, respectively. You can access the data using the the commands shown in Fig. 14.

```
% save into variables
t = data_temp(:,1);
T1d = data_temp(:,2);
T1 = data_temp(:,3);
Vh = data_u(:,2);
Vb = data_u(:,3);
```

Figure 14: Loading data from workspace variables *data_temp* and *data_u*.

12. After you observe the general characteristics of the performance, stop the model and save *data_temp* and *data_u* into a .mat file named *data_on_off_controller_hysteresisWidth1*. Copy this .mat file into your USB flash drive.

13. Reduce the hysteresis width to 0.2. Run the model again and once you observe a full period of response stop the model and save *data_temp* and *data_u* into a .mat file named *data_on_off_controller_hysteresisWidth0_2*. Copy this .mat file into your USB flash drive.

6.3 PI Temperature Control

The Simulink model called *q_hfe_pi*, depicted in Fig. 15, is used to implement the PI controller on the Heatflow using QUARC.

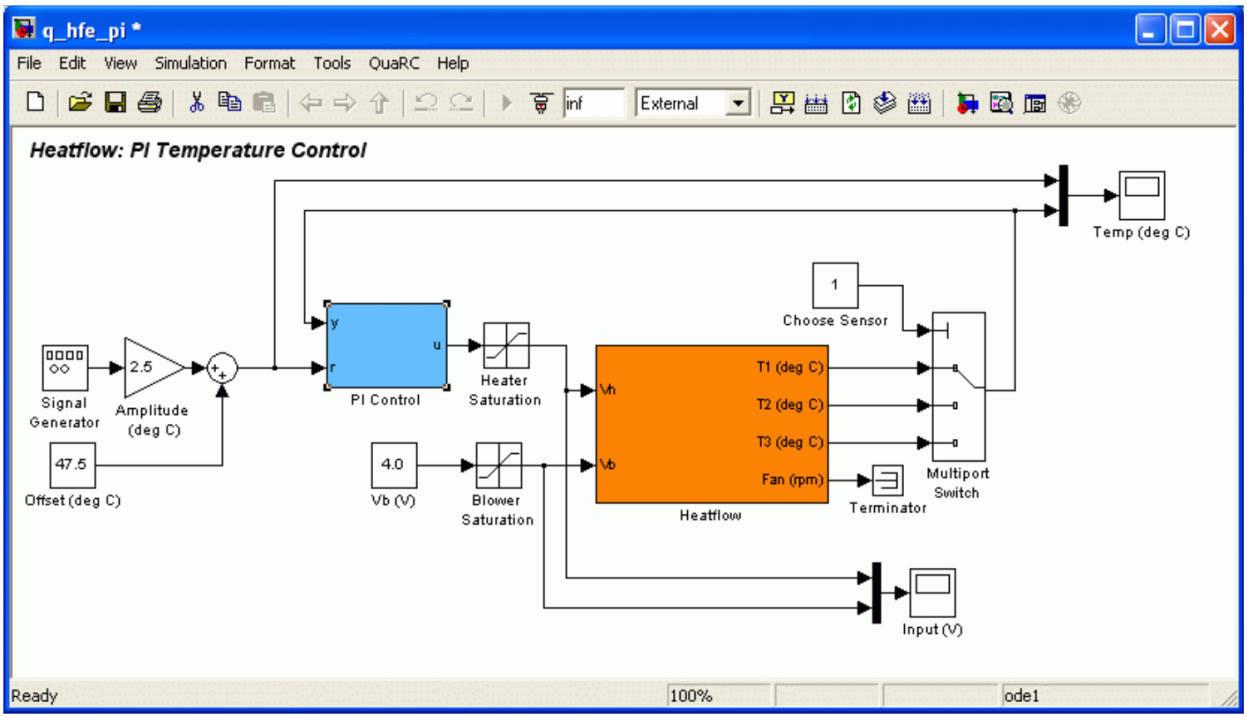


Figure 15: Simulink diagram used to run PI control on Heat Flow using QUARC

The *PI Control* subsystem is shown in Fig. 16. As depicted, the *kp* and *ki* Slider Gain blocks are used to change the proportional and integral gains, respectively.

1. Open the *q_hfe_pi* Simulink model.
2. Click on *Quarc |Set default options*.
3. Click on *Quarc |Build* to compile the QUARC controller.
4. Ensure that the *Amplitude (deg C)* block is set to 2.5 and the *Offset (deg C)* to 47.5. This square setpoint signal will vary between 45.0 ^{o}C and 50.0 ^{o}C.
5. In the *Signal Generator* block, set the *Frequency* of the square wave to 0.02 Hz.
6. Set the *Vb (V)* block to 4.0 V. The blower will be running at a constant rate while we vary the voltage to the heater.
7. Set the *Choose Sensor* block to 1 in order to control the temperature about the T_1 sensor.
8. An example temperature response for sensor T_1 is shown in Fig. 17a with the PI gains set to $k_p = 0.7$ and $k_i = 0.05$. The heater and blower voltages are given in Fig. 17b. The heater voltage is the yellow trace and the purple plot is the blower voltage.
9. To observe the response of a pure proportional control, set $k_p = 0.3$ and $k_i = 0$.

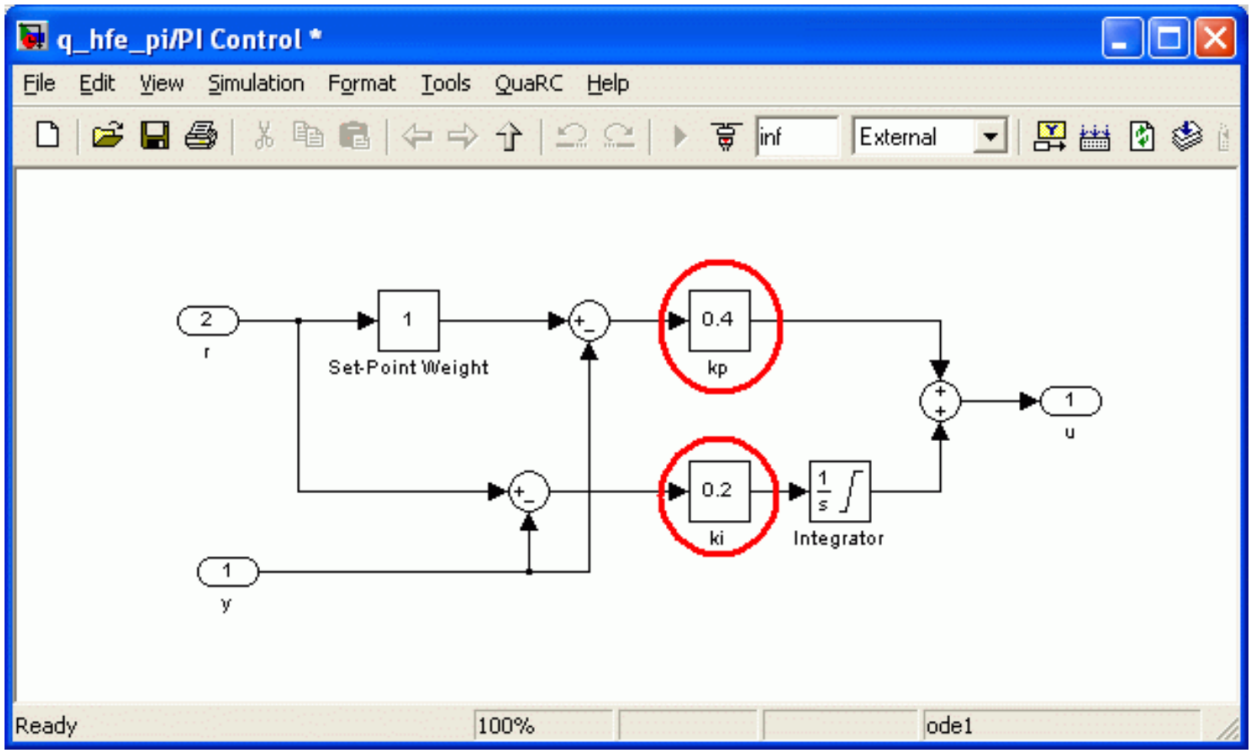


Figure 16: Inside the PI Control subsystem in *q_hfe_pi* model.

10. In the menu of the Simulink model, go to *QUARC |Start* to begin running the controller.

11. After you observe the general characteristics of the response, stop the model and save *data_temp* and *data_u* into a .mat file named *data_P_only_controller*. Copy this .mat file into your USB flash drive.

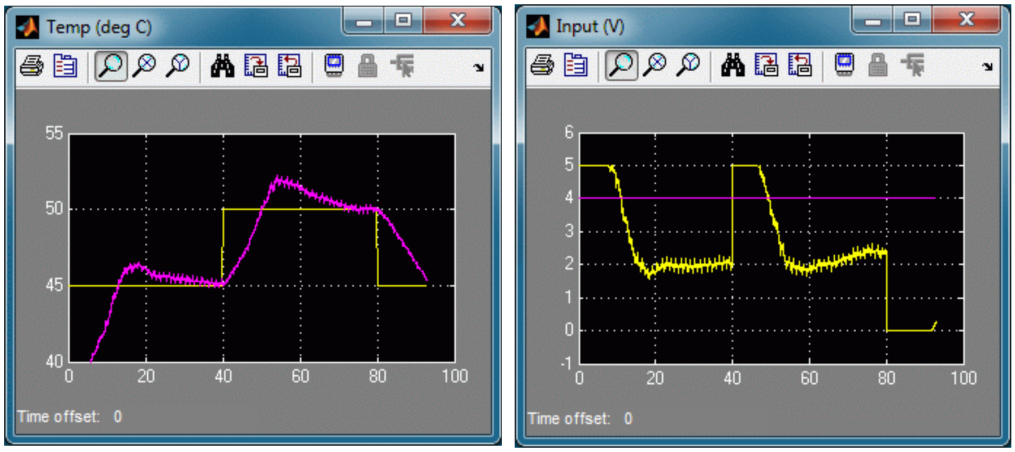
12. Click on QUARC |Stop to stop running the controller.

13. Now set $k_p = 0.5$ and $k_i = 0$ and run the model.

14. Slowly begin increasing the integral gain. Once you are satisfied with the response have your assistant check your design.

15. The integral anti-windup scheme is currently being used, as can be identified by the saturation marks on the *Integrator* block inside the *PI Control* subsystem (pictured in Fig. 16). The heater saturation is 0-5 V, so the lower and upper saturation limits of the integrator are set to 0 and 5, as shown in Fig. 18. The integrator resets every sample, so the reset time is automatically set to $T_r = 1$ when using this block.

16. To turn anti-windup action off, first stop the model. Then, go into the *PI Control* subsystem, double-click on the *Integrator* block, and uncheck the *Limit Output* parameter as illustrated in Fig. 18.



(a) PI temperature response (b) PI blower and heater voltages

Figure 17: Responses for PI control

17. Without changing the controller parameters (which were previously optimized) rebuild and start the model controller.

18. After you observe the response, stop the model and save *data_temp* and *data_u* into a .mat file named *data_PI_without_antiWindUp*. Copy this .mat file into your USB flash drive.

19. Now, reengage the anti-windup without changing the controller gains and run the experiment one more time. Your are going to be asked to interpret the effect of anti-windup scheme, so take your time to investigate the resulting response.

20. Stop the model.

6.3.1 PI Control Considering the Time-domain Specifications

1. Based on ω_n and ξ values (determined by the time-domain specifications) and the identified model parameters, K and τ, calculate the control gains k_p and k_i analytically.

2. Run the *q_hfe_pi* model with these controller parameters.

3. After you observe the response, stop the model and save *data_temp* and *data_u* into a .mat file named *data_PI_analytically_tuned*.

4. Calculate the peak time and percentage overshoot by inspecting the measured signals via appropriate Scope blocks.

5. If the response did not satisfy the requirements or just to enhance it, try to perform some fine tuning by adjusting the PI gains.

6. Once you satisfy the requirements stop the model and save *data_temp* and *data_u* into a .mat file named *data_PI_fine_tuned*. Don't forget to note the final values of the gains.

7. Stop the model.

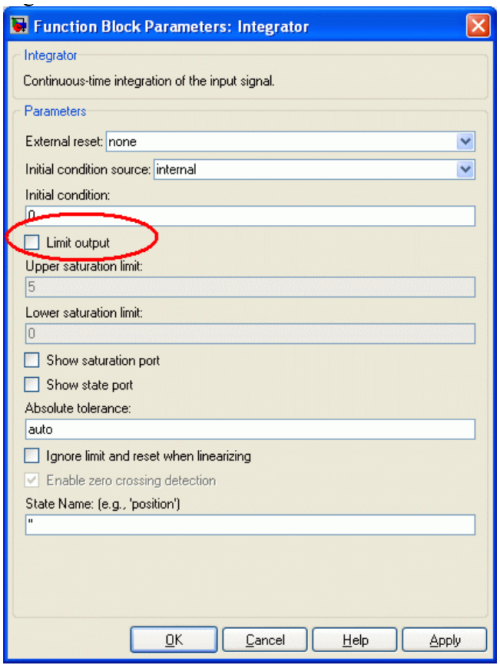


Figure 18: PI blower and heater voltages

7 Results

1. (**Bump test:**) Plot the temperature responses and the applied heater voltage that you recorded during the bump test with explanatory axis labels and legend. Notice that the relevant data can be accessed using the MATLAB commands shown in Fig. 19.

```
% load raw data from MAT file
load('data_open_loop.mat');
% save into variables
t = data_hfe_ol(1,:);
T1 = data_hfe_ol(2,:);
T2 = data_hfe_ol(3,:);
T3 = data_hfe_ol(4,:);
Vh = data_hfe_ol(5,:);
Vb = data_hfe_ol(6,:);
```

Figure 19: Loading data from *data_open_loop.mat*.

2. Does the temperature in different locations along the duct go up at the same rate? Explain.

3. (**On-Off Controller:**) Plot the T1 responses and heater voltages generated by two different on-off controllers (with $Hysteresis\ Width = \{1, 0.2\}$) (you may find the commands given in Fig. 14 useful). How does decreasing the hysteresis width affect the response? Do you see any potential drawbacks of making the width too small?

4. List possible advantages and disadvantages of the on-off controller scheme.

5. (**P-only Controller:**) Load the *data_P_only_controller.mat* file. Plot the T1 response and the heater voltage. Measure the steady-state error from the graph. Is the result consistent with your expectations?

6. Now, evaluate the steady-state error analytically given the amplitude of the step, the control gain, k_p, being 0.3, and the model parameters identified in Section 6.1.2. How is it compared to the measured value?

7. (**PI Manual Tuning:**) Discuss the effect of adding an integral component next to the proportional only controller regarding your tuning experience in the lab.

8. (**PI Anti Windup Effect:**) Load *data_PI_without_antiWindUp.mat*. Plot the T1 response and the heater voltage. Comment on the effect of not having a anti windup action in the PI controller.

9. (**PI Analytical Tuning:**) Indicate the values of the controller gains you analytically calculated to satisfy the specifications. Load *data_PI_analytically_tuned.mat*. Plot the T1 response and the heater voltage. Calculate the peak time and percentage overshoot of the response. Is there any discrepancy is the results? If so, state some of the possible reasons. Are these values in agreement with the design requirements?

10. (**PI Fine Tuning:**) Now, if you fine tuned the controller, load *data_PI_fine_tuned.mat* and plot the T1 response and the heater voltage. Measure the peak time and percentage overshoot. Are the requirements now satisfied?

8 Conclusions

Write down your general conclusions and comments on this experiment.