EEE 342 Feedback Control Systems
Spring 2022-2023

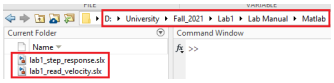
Lab 1 - System Identification and PI Controller Design

As in preliminary work, you will need to identify the system by using quantized step response of the open velocity loop with noise. To obtain step response of your kit, Simulink files and hardware communication need to be configured. The steps in part-1 explain the adjustments you need to do to get samples from the physical system properly. In the second part, you will use a first order approximation on received data to identify the DC motor. In third part, you will need to design 3 different PI controllers to observe effects of P and I coefficients on step response. Finally, you will implement your final PI controller design and obtain the closed loop response experimentally.

Part-1: Hardware Connection and Configuration

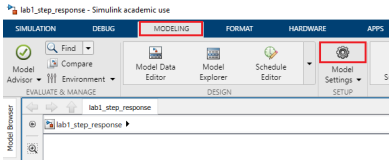
The sole purpose of this part is to assist you to configure Simulink diagrams and provide a proper connection between the kit and Matlab. Therefore, you can skip this part in your reports except the plot of position vs time data obtained from hardware.

1. Connect the power source to your kit. This will cause the Arduino to run the last installed code.
2. Please close Matlab and plug your USB cable for Arduino and wait until a COM port is assigned. Then, you can open Matlab.
3. Create a folder and download `lab1_step_response.slx` and `lab1_read_velocity.slx` files from moodle into this folder. Then, open MATLAB-R2021b and set your Matlab current workspace directory to the folder you created as shown below.

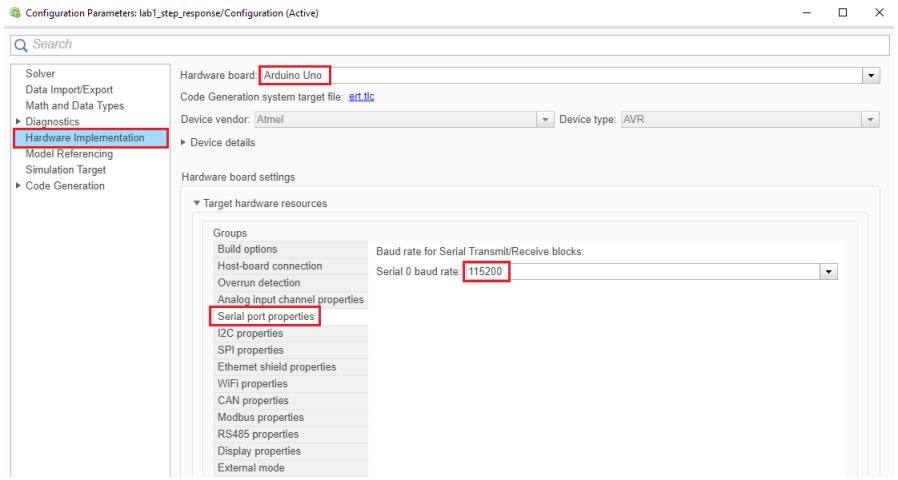


   Note that the path of the folder should only contain ASCII codes (space and Turkish special characters may return error during your work on Simuink).

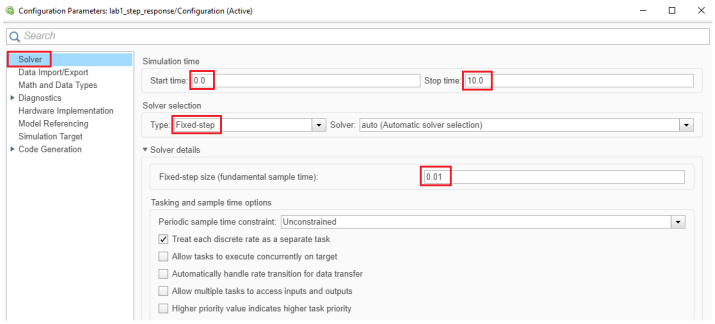
4. Open `lab1_step_response.slx` file and configure the voltage source (Step block): Step time: 0, Initial value: 0, Final Value: 12, Sample Time: 0.01. This block will now release $r(t) = 12u(t)\ V$.
5. The whole configuration part will be done in configuration window. You can open this window by pressing CTRL+E or by clicking on "Model Settings" button under Modelling tab.



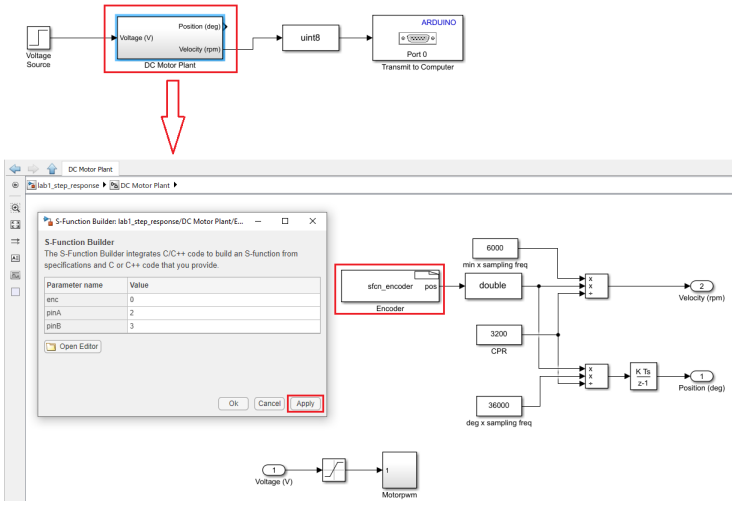
In order to provide a communication between Simulink and kit, you firstly need to identify the hardware you are using. This identification allows the Simulink to recognize pins of Arduino and compile properly. The kits contain Arduino Uno, therefore, in hardware implementation pane in configuration window, you need to select Arduino Uno as hardware board. Also, in the same pane, set Serial 0 baud rate to 115200, which can be found under Serial port properties. The figure provides visualization for this description.



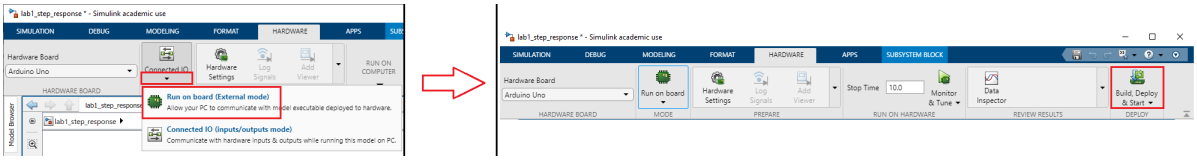
Now that we configured the communicated device, we need to set duration and sampling period. This can be done by opening the Solver pane in configuration window. Set Start Time to 0 and Stop Time to 10. Then, select the Type as Fixed-step in Solver selection panel. Lastly, open Solver details and set fixed-step size (fundamental sample time) to 0.01. The figure below provides visual content for this description.



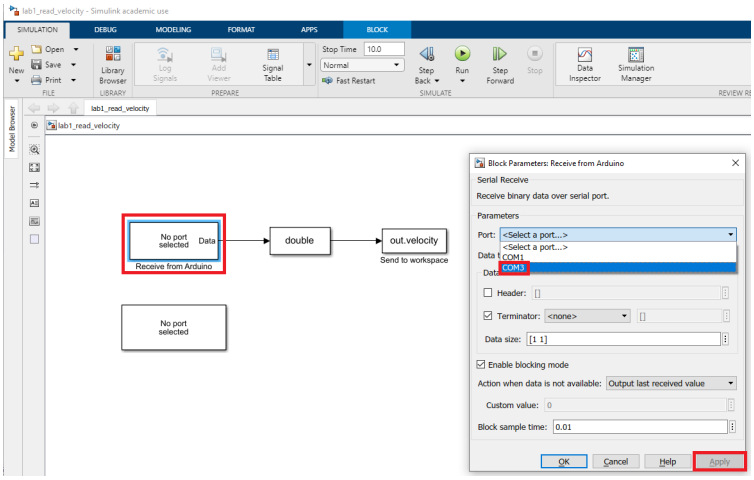
6. There is a final step to provide proper communication between Simulink and Arduino, that is to compile embedded code in `.slx` file. Open DC Motor Plant in `lab1_step_response.slx` and double click on sfcn_encoder block. Press the "Apply" button on bottom right of the last opened window. This action will compile the C code in this block. The following figure shows how this process is visualized on corresponding window.

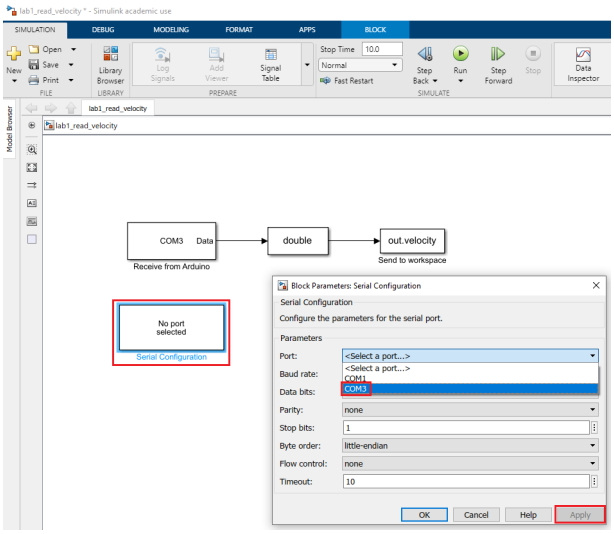


7. Close the last window and change the mode from "Connected IO" to "Run on Board (External mode)" `lab1_step_response.slx`. Press the button with title "Build, Deploy & Start" under Hardware tab, as shown below. Now, the embedded code is built within Arduino, and you will see that the DC motor rotates for 10 seconds. However, you will not be able to receive the output data yet.



8. Download `lab1_read_velocity.slx` to your folder and open it. This file will allow you to receive data. Open "Receive from Arduino" block and set Communication Port to **second** line (for example COM23). Click "No" if a window opens after you apply changes. Repeat the same procedure for "Serial Configuration Block". You also need to set duration to 10, as can be seen from the following figure.





9. Apply these changes and close these windows. Open `lab1_read_velocity.slx` again, enter 10 for simulation duration and click "Run". After 10-15 seconds, you will see a *SimulationOutput* data called "out". Save this data as `vel_1` (both by using `vel_1=out.velocity` to save on workspace, and by saving the variable `vel_1` on workspace to your current folder as `vel_1.mat`).

Check-1 Plot the received data and show the figure to one of the TAs for the first check.

Report Plot the raw data in your report. Comment on the received data.

Part-2: System Identification in Time Domain

1. Note that you need to save the previously received data to avoid repeating part-1 to receive lost data. As you have done in your preliminary work, you will need to estimate first order approximation of DC motor after passing the received data from an LPF, where

$$H_{LPF} = \frac{1}{0.01s + 1} \ .$$

   Show your calculations and sample points on figures in your reports.

2. Download `lab1_velocity_sim.slx` file to your folder. Open this file and define your first order approximation in DC motor block.

3. Plot the response of simulation on top of the data you received from hardware. You can use the following lines:

```
figure; plot(vel_1,'b','LineWidth',2);
hold on; plot(out.vel_sim,'r','LineWidth',2);
xlabel('Time (sec)'); ylabel('Velocity (rpm)');
title('Velocity vs Time');
legend({'Hardware result','Simulation result'},'Location','SouthEast');
```

Check-2 Plot the received data and output of your model on top of each other. Show the figure to one of the TA's for the second check.

Report Plot the figure and type the first order approximation of your DC motor, and the calculations you have made in your report. Explain the reasons of differences between approximation and raw data.

Part-3: PI Controller on Simulation System

Now that you approximately found a mathematical model of the physical system, you can design controllers in computational environment. You are asked to design a PI controller such that the output of the closed loop satisfies the following criteria.

a) Steady state error is 0.

b) Maximum percentage overshoot is less than 8

c) Settling time is less than 0.8 seconds (%2 error bound).

In order to design your controller, please use the following steps:

1. Download `lab1_closedLoop_sim.slx` from Moodle.
2. Make sure that the input is $r(t) = 120\ rpm$.
3. Adjust the parameters of first order approximation of DC motor:

$$G_{motor}(s) = \frac{K}{\tau s + 1}$$

4. Set PI controller coefficients K_p and K_i to $[0.5, 0.5]$.

$$C_{PI}(s) = \frac{K_p s + K_i}{s}$$

5. Run the simulation and observe the output. Change your PI coefficients according to specified criteria and what you have learned from the first preliminary work (for example consider how overshoot can be reduced or how settling time can be increased).
6. Simulate the closed loop and adjust your PI parameters until the output satisfies the criteria.
7. Finally, you should have **3** different PI controllers where one of them satisfies the criteria mentioned above, one of them has $K_p = 10K_i$ relation, and the last one satisfies $K_i = 20K_p$ equality.

Check-3 Plot outputs of 3 aforementioned controller designs on different figures. Show these figures to one of the TA's. You also need to show that the criteria are met in your final design (overshoot, settling time and steady state error).

Report Your report needs to include plots of responses of the closed loop with 3 different PI controller designs. Please indicate the overshoot, settling time and steady state error of all three responses. Also, output of the controller of final design should be shown in your report.

Part-4: PI Controller on Physical System

1. Download `lab1_closedLoop_hardware.slx` from Moodle.
2. Implement your final PI controller in PI controller block in Simulink.
3. Deploy this file to Arduino (follow the steps at part-1 to avoid any errors during data transmission).
4. Run `lab1_read_velocity.slx` to receive data.

Check-4 Plot the received data and your simulation output for the final controller on top of each other. Show the figure to one of the TAs for the final check.

Report Plot the data obtained from both simulation and hardware on top of each other (a single figure that contains both of the output responses) in your report. Comment on differences between hardware and simulation results in your report. Does the response of hardware satisfy criteria given in part-3? Also, please comment on how system identification and simulation was useful during controller design in your conclusion.

In your report, you are expected to explain the work done in order. It needs to include all plots you drew, all mathematical equations you did, and all the results you obtained in the lab. You also need to comment on each result you obtained between lab checks. Do not forget to use **report template** and write introduction and conclusion parts.