Exercise Manual
For

SC2104/CE3002
Sensors, Interfacing and Digital Control

Practical Exercise #4:
Digital Control
of
DC Motor

Venue: SCSE Labs

COMPUTER ENGINEERING COURSE

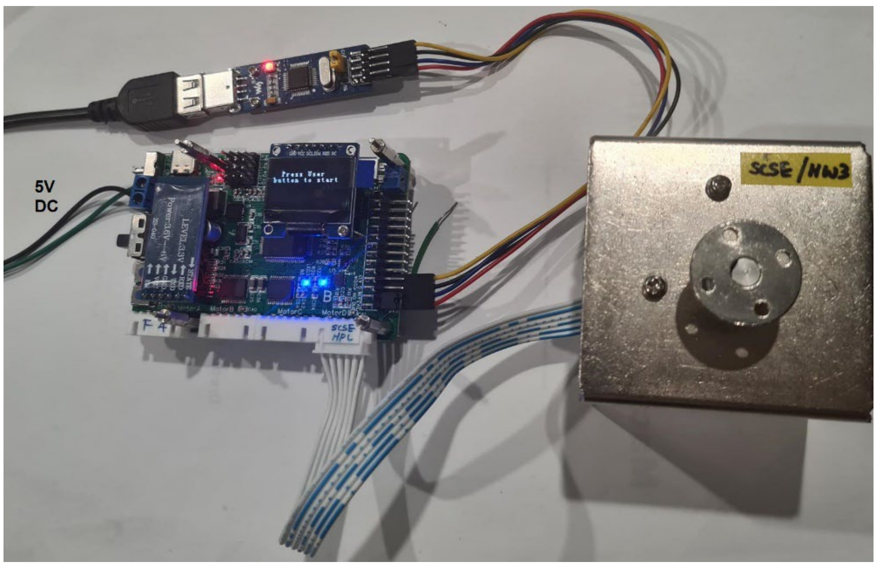
SCHOOL OF COMPUTER SCIENCE AND ENGINEERING
NANYANG TECHNOLOGICAL UNIVERSITY

Learning Objectives

These exercises are to introduce student to the implementation of a digital control system to operate a DC motor (using the STM32F4 board). Student will observe the control system's effects on the DC motor operation based on the P, I and D control algorithms.

Equipment and accessories required

i) One desktop computer installed with STM32CubeIDE and SerialPlot.
ii) One STM32F4 board
iii) One ST-LINK SWD board
iv) One 20:1 gear DC Motor (attached with 26-poles/13 pulses Hall Effect Encoder)



Introduction

Digital control is now the predominant mean for many control applications. In this exercise, you will go through a sequence of tasks designed to let you be familiarized with the concepts of digital PID control and use it to control the position of a DC motor and observe its behaviours with different parameter settings.

1. Setup

You will be using the STM32F4 board (used in Lab 1 to 3 exercises) and the STM32CubeIDE for this lab exercises, together with the SerialPlot software to observe the movement of the motor position. The program code required for the exercises are provided in a zip file available on NTULearn course site, which you will use to setup a new project.

1.1. Download the file

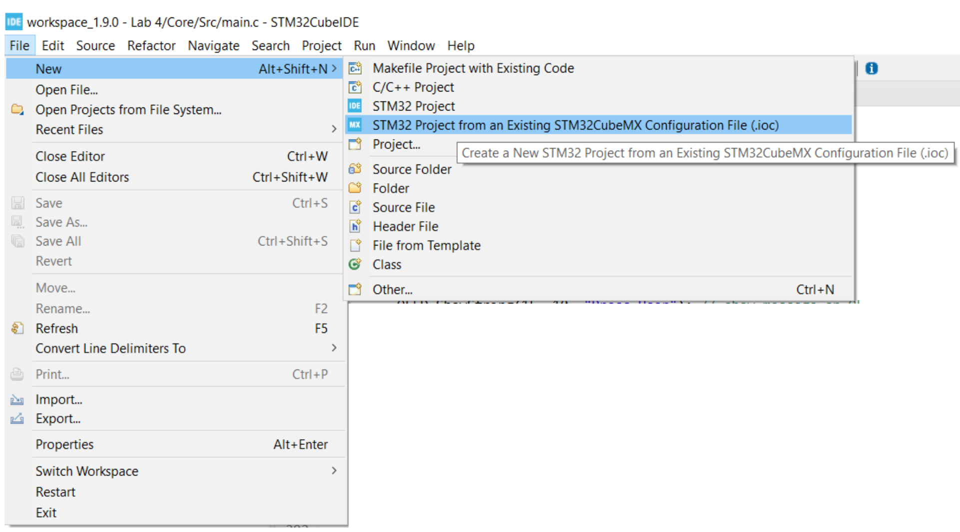
Download the zip file that contains the following files from NTULearn course site, extract and store them on the computer (or your thumb drive).

Lab 4.ioc
main.c
oled.c
stm32f4xx_it.c
main.h
oled.h
oledfont.h

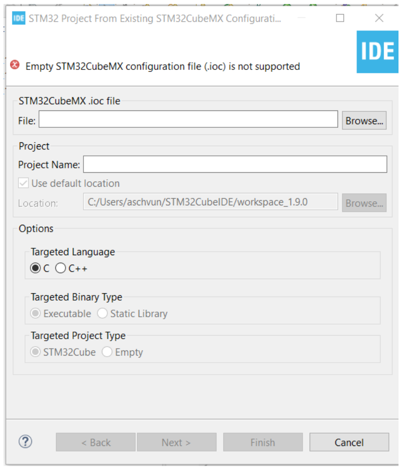
You will import these files into the project for this exercise.

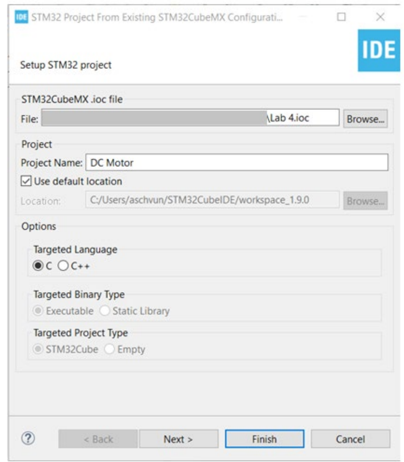
1.2. Setting up the STM32 Project

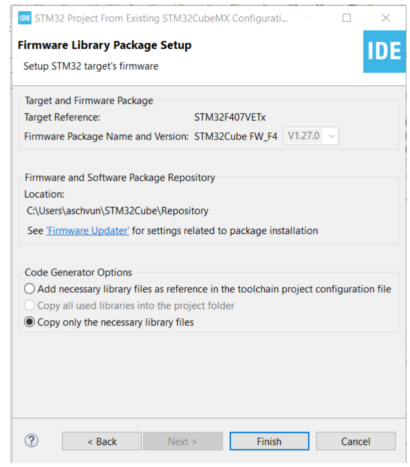
Launch the STM32CubeIDE on the computer and create a new STM32 project by choosing the option to import an existing configuration file.



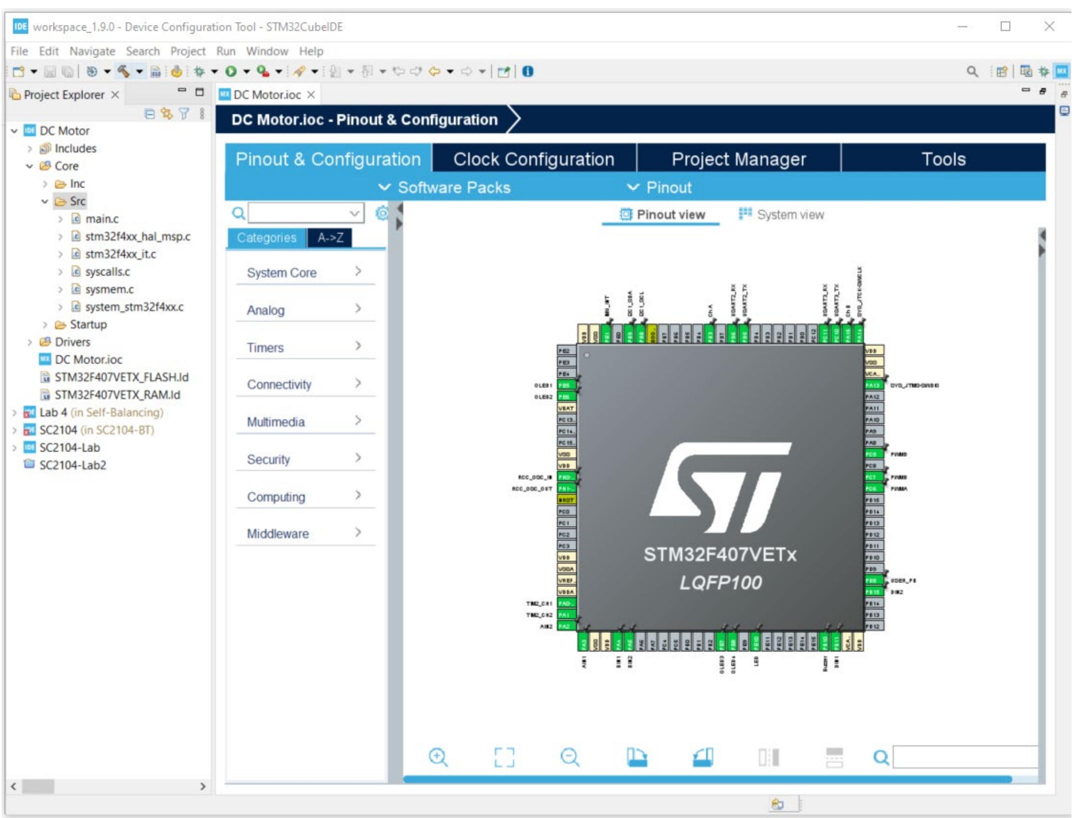
- Import the “Lab 4.ioc” configuration file shown below to setup a new project “DC Motor”.





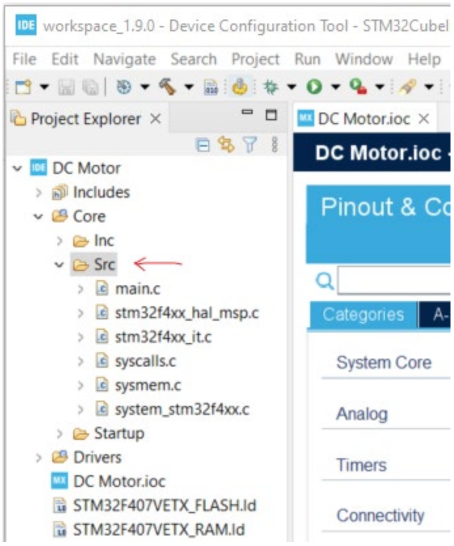


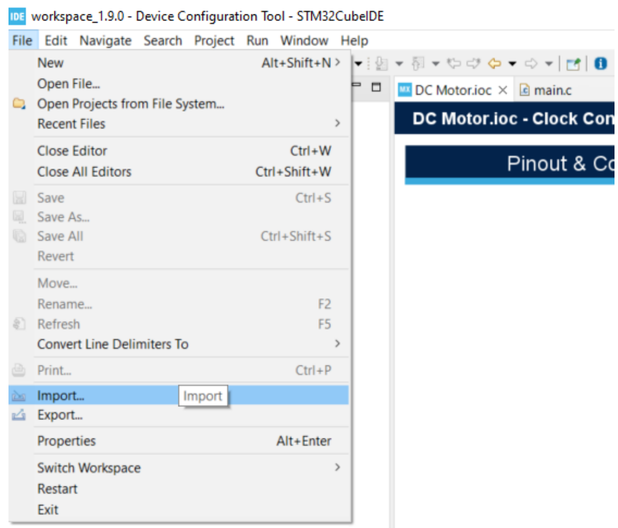
Once the configuration file has been imported and executed, the Microcontroller configuration will be shown as below, together with the auto generated program code (on the left panel) corresponds to the configuration.

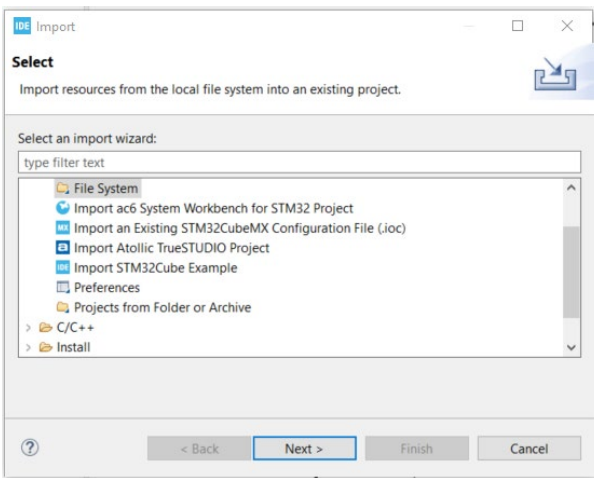


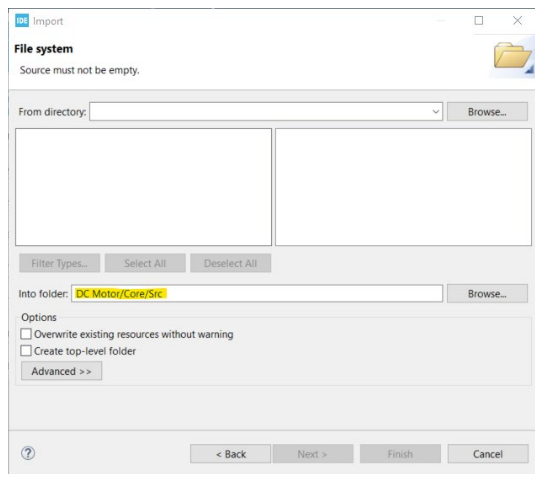
You will now import the program files extracted from the zip folder into this project.

- Select the 'Src' folder to import the given files main.c, oled.c and stm32f4xx_it.c

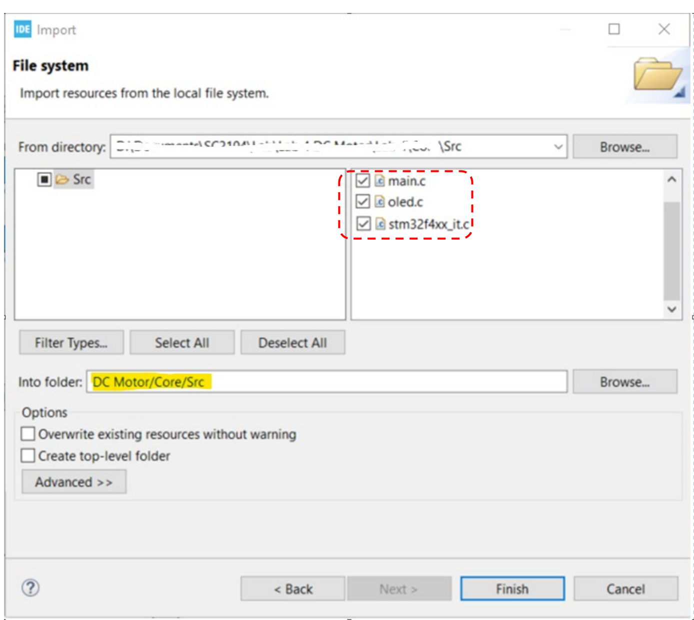




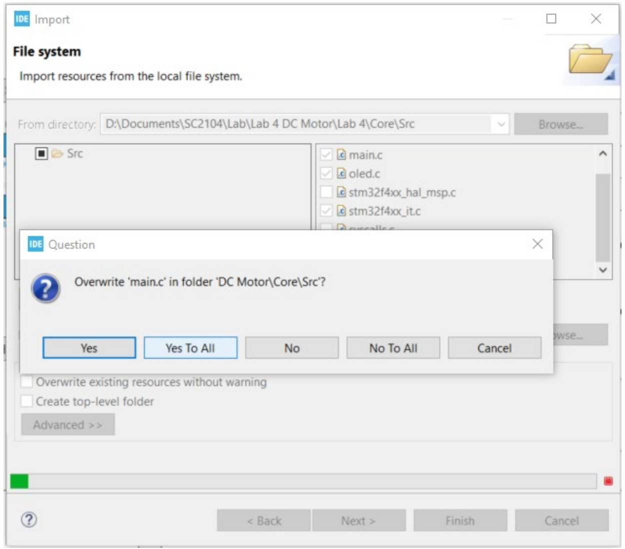




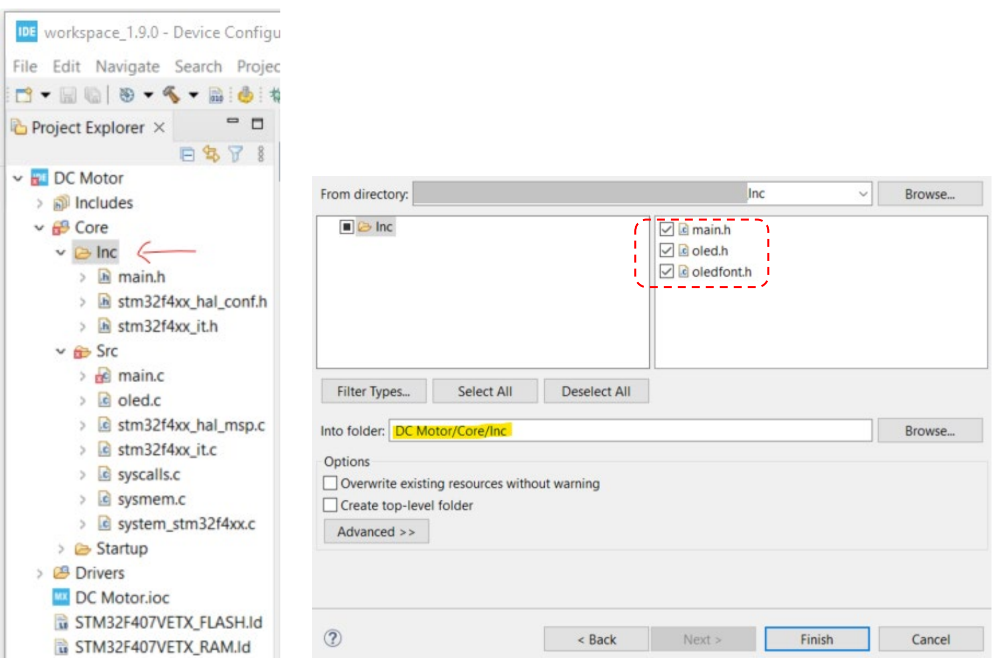
- Select the directory where the files are kept, and import them into the folder 'Src'



Note that the default main.c file is to be replaced by the provided version.



- Select the 'Inc' folder and similarly import the three header files provided as shown below.



- Build/compile the project and download the code onto the STM32F4 board. (The code should be compiled with no error)

- Connect the USB serial port 3 (the middle USB port) to the computer and run the SerialPort program (with 115200 baud rate and using 'comma' column delimiter) to observe the data sent by the program.

1.3. Program logic

The logic of the program is as follows:

1. Perform all the necessary initializations
2. Setup interrupt to detect User push button
3. Set the target angle (e.g., 720 degrees, i.e., 2 revolutions)
4. a. Execute the control loop
   b. Send the data to the serial port to be displayed in SerialPort program
   c. Repeat from 'a' until output angle has settled at/near the target angle
   d. Exit the control loop and stop the motor
   e. Wait for User push button to be pressed and go to 'a'

The data that are sent through the serial port are as follows (using 'comma' as column delimiter)

i) Output angle (Display this value on the SerialPlot)
ii) Target angle(Display this value on the SerialPlot)
iii) Error - difference between i) and ii)
iv) PWM value used to drive the motor (Maximum value is 7200, minimum value is ?)
v) Value of error integration - for PI control
vi) Number of times the output angle has settled within the target angle region (set to 5 to trigger 'd' above)
vii) Value of error rate - for PD control

The following are code snippets relevant to the exercises (note their line numbers)

a. Setting of Target angle

```
314   /* Infinite loop */
315   /* USER CODE BEGIN WHILE */
316   start = 0;
317   angle = 0;
318   target_angle = 720; // rotate 720 degree
319   error = target_angle - angle;
320   error_old = 0;
321   error_area = 0;
```

b. Setting of PID parameters

```
331
332   Kp = 10;        // 10
333   Ki = 0.000;    // 0.001
334   Kd = 0000;
```

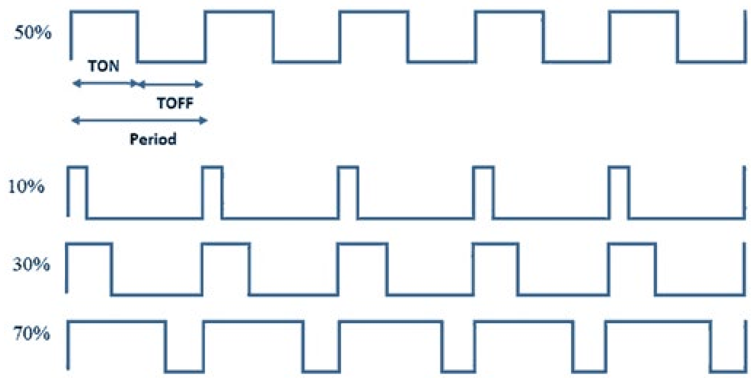
c. Control loop.
The PID control loop is called at line 360. But it can be ‘disabled’ by un-comment and set the PWM value (pwmVal) at line 361.

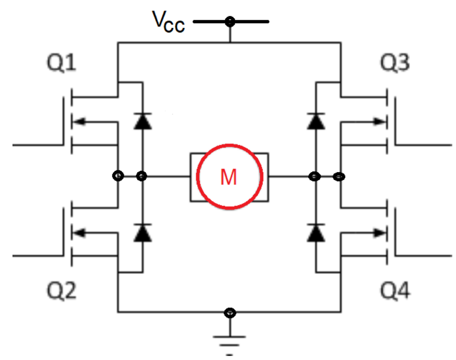
```
339   while (1)
340   {
341       if (start==0){ // reset and wait for the User PB to be pressed
342           pwmVal = 0;
343           __HAL_TIM_SetCompare(&htim8,TIM_CHANNEL_4,pwmVal);
344           HAL_GPIO_TogglePin(GPIOE, GPIO_PIN_10); // LED
345           //OLED_Clear(); // get display ready
346           OLED_ShowString(15, 40, "Press User"); // show message on OLED display at line 40)
347           OLED_ShowString(0, 50, "button to start"); // show message on OLED display at line 50)
348           OLED_Refresh_Gram();
349           err = 0;// for checking whether error has settle down near to zero
350           angle = 0;
351           error_old = 0;
352           error_area = 0;
353           error = target_angle - angle;
354           }
355       while (start==0){ //wait for the User PB to be pressed
356           HAL_Delay(500);
357           millisOld = HAL_GetTick(); // get time value before starting - for PID
358           }
359
360       pwmVal = PID_Control(); // call the PID control loop calculation
361       //pwmVal = 2000;          // overwrite PID control above, minimum pwmVal = 1000?
362       __HAL_TIM_SetCompare(&htim8,TIM_CHANNEL_4,pwmVal); // output PWM waveform to drive motor
363
364       if (abs(error) < 3){ // error is less than 3 deg
365           err++; // to keep track how long it has reached steady state
366           angle = (int)(position*360/260);  //calculate the angle
367           error = target_angle - angle; // calculate the error
368           }
369
370       serial_uart(); // send the various data to the serial port for display
371
```

2. Practical Exercises

2.1 Operating range of the PWM value

Depending on the DC voltage (V_{CC}) applied to the board, the minimum PWM value needed to start turning the motor will vary, while the maximum PWM value is to be limited to 7200 (i.e., its Period).





- Set the pwmVal at line 361 to various values (e.g., 1000, 800, 600 etc) to find the minimum value that will make the motor starts to turn. This minimum value will affect the proper operation of the motor control as will be observed later.

2.2 Step response – Fixed PWM control loop

In this part of the exercises, you will compare the effect of using various fixed PWM values to operate the control loop.

- Set the pwmVal at line 361 to the following values and observe their respective step responses – i.e., the shape and time taken to reach the target angle.
  i) 1000
  ii) 1500
  iii) 2000

- Run the SerialPlot program to display the Output Angle and Target angle sent by the microcontroller. (Note: Screen capture the waveforms such that you can later include them in your lab report for submission.)

- Try turning the motor shaft by hand and observe the response of the system. This is equivalent to introducing a disturbance to the motor.

2.3 Step response – PID control loop

In this part of the exercises, you will compare the effect of using P, I and D to operate the control loop for the DC motor.

a. Proportional Gain Kp
Set the Kp (at line 332) to various values and observe the change on the step response of the system.
- Kp = 10 (Compare this response to those observed in 2.2)
- Kp = 20
- Kp = 30
- Kp = 40

b. Integral Gain Ki
Add the integral gain Ki (at line 333) to form a PI control loop and observe its effect on the step response of the system
- Ki = 0.001, Kp = 10 (Compare this response to 'a' above)
- Ki = 0.001, Kp = 20

c. Differential Gain Kd
Add the Differential gain Ki (at line 333) to form a PID control loop and observe its effect on the step response of the system
- Ki = 0.001, Kp = 20, Kd = 500 (Compare the response to 'b' above)
- Ki = 0.001, Kp = 20, Kd = 1000

3. Report Submission

Submit a report (up to maximum of 5 pages) summarizing your observations and explanations of the responses using the different control strategies, accompanying with screenshots of the various waveforms.