Introduction to Mechatronics (ME/AE 6705)
Lab Assignment 10

Feedback Control of a DC Motor
Using Encoders and MSP432 LaunchPad

10.1 Objective

This main objective of this lab is to learn how to implement a feedback controller for a DC motor to drive it to a desired speed.

10.2 Deliverables and Grading

To get credit for this lab assignment you must:

1. Submit a written report as a PDF file to Canvas, answering the questions in Section 10.8. You must write your own Simulink and C code for this lab – no lab groups are allowed. Submit your report to Canvas before the due date specified on Canvas. (80 points)

2. Submit the commented final version of your code on Canvas (single .c file containing your main() function, do not submit header files, etc. You should only submit a single file). (Pass/Fail)

10.3 Setup

This lab requires Code Composer Studio, MSP432 LaunchPad, and the DC motor system and the motor driver circuit used for Lab 9. The lab uses onboard features of the MSP432 LaunchPad including GPIO, Timer A, PWM, and interrupts.

Note: To setup a new project for CCS with driverlib, create an "Empty Project (with main.c)" similar to lab 2, copy only the "driverlib" folder provided with lab 2 and paste it into the new project, and specify the search path for the C preprocessor.

10.4 Problem Statement

Write a program for the MSP432 that implements feedback control for the DC motor setup created in Lab 9. The goal of the feedback controller is to drive the speed of the motor to a desired value (which you select) using duty cycle as the control input.

You will implement a PID controller that satisfies the following performance specifications: rise time of less than **500 ms** and overshoot of less than **5%**.

10.5 Simulation Model

To begin, construct a simple simulation model of the DC motor system in Simulink. You may use the example Simulink models provided on Canvas as templates. The plant model for the DC motor system should be a first-order system with the time constant you measured in Lab 9:

$$G(s) = \frac{\omega(s)}{D(s)} = \frac{K}{\tau s + 1}$$

where τ is a constant. In the above equation, $D(s)$ represents duty cycle and $\omega(s)$ represents motor speed. The value of K is the gain you determine for the 100% duty cycle case in Lab 9.

The input to your block diagram should be desired speed. The current speed should be subtracted from this to form the error signal. The error signal should be input to the PID control block. The output of the PID control is the duty cycle value, which should be input to the first-order plant (you can eliminate any amplifiers used in the example Simulink templates). Make sure to limit the output of your PID controller to a minimum of 0 and maximum of 100 (100% duty cycle). This can be done by double clicking on the PID block, and going to the PID Advanced tab.

Select two desired reference speeds ω_1 and ω_2. These should be selected at some values between the steady-state speed you observed for your DC motor system in Lab 9 between 20% duty cycle and 100% duty cycle. Simulate the step response of your closed-loop system to each of these two desired speeds (this will require running two different simulations, each with a different desired speed). Tune the PID gains to achieve the performance specifications above. Plot the step responses and show that your response meets the required performance specifications for each case. In these simulations, note that $\omega = 0$ corresponds to the duty cycle = 0%, and thus ω_1 and ω_2 are measured as absolute values.

10.6 Hardware

The motor driver circuit and the DC motor/encoder used in the Lab 9 will be used for the experimental portion of this lab as well.

The MSP432 LaunchPad should be used to implement the PID controller for setting the PWM duty cycle that controls the speed of the DC motor.

10.7 Software

Implement a PID controller on your MSP432 that controls the DC motor speed using duty cycle as the control input. Use the rectangular integration method and first order finite differencing described in class. Make sure to enforce saturation on your duty cycle so it can never be set below 0 or above 100%.

Run the controller at 100 Hz and store the timer counts for the encoder pulses in a similar manner as in Lab 9. Convert these counts to velocity readings once the motor velocity reaches a steady state. In addition, output the velocity of the motor to the screen at 1Hz.

Hint: This can be done using a Timer A interrupt and a counter (volatile global variable) which is incremented during each interrupt. Every time the counter reaches 100, a printf statement is called and the counter is reset.

10.8 Requirements

1. Pick two motor velocities, ω_1 and ω_2, greater than 90 rpm. Use your closed loop Simulink model to simulate the step response for each velocity. On each plot highlight the rise time and overshoot to show that they meet the specified requirements (you might want to look at Matlab's stepinfo() function for this). State the PID gains you used for these simulations

2. Now using your *hardware* setup and the same PID gains used in simulation, control the DC motor to each speed of ω_1 and ω_2 starting from zero speed each time. Record the step response for each case. Plot the experimental step response (two plots, one for each case) and highlight the overshoot and rise time. Does it match the required performance specifications? Now create two new plots where you overlay the simulated and experimental response. How similar is the experimental response to the simulated one? If the simulated and experimental responses look noticeably different, explain why you think that is the case.

3. If your experimental response does not meet the required rise time and overshoot specifications, tune your PID gains until it does. You might want to start by setting all gains to 0 and tuning your integral gain first, then your proportional gain, and finally your derivative gain. Note, the final gains might differ significantly from your simulation gains.

4. Use the controller gains from Question 3 to control the motor speed to a value lower than 80 rpm. What do you notice in the step response? Explain the reason behind the behavior observed in your response.

5. Submit the commented final version of the MSP432 code on Canvas.