ME 133 Winter 2023
Lab 6: *DC and Stepper Motors*
March 2, 2023
Due: 3/10/2023

Submit a `zip` file named `yourFirstName-Lab1.zip` on Canvas with your code, a lab report (following the format in syllabus), and a short video proving the working hardware-software integration.

Exercise 1

Material required

- Arduino
- 1 x Transistor
- 1 x Diode
- 2 x Buttons
- 1 x 330 Ohm resistor
- 2 x 10k Ohm resistors

Coding together

We will learn about DC brushed motors.

Consider the circuit shown below to control a DC motor in a single direction. Most DC motors require current greater than 250mA. The maximum current of Arduino is 150mA. If motors are directly connected to the output of any pin, this can damage both the pin and the motor. To prevent damage, you need circuitry that acts as a bridge between your microcontroller and the motor. Some possibilities are transistors, H-bridges, and relays. In this lab we will use a transistor as shown below.

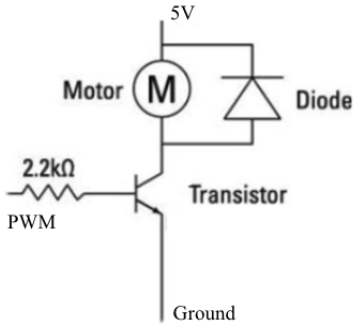

Figure 1: How to safely wire a DC motor to your microcontroller

The diode provides a safe path for the inductive kickback of the motor. If the current in the inductor is suddenly switched off, the diode will briefly supply the voltage necessary to keep the current flowing in the short term.

Exercise 2

Coding together

We will learn about DC Stepper motors.
From: https://lastminuteengineers.com/28byj48-stepper-motor-arduino-tutorial/
See also, for a good theoretical explanation: http://users.ece.utexas.edu/ valvano/Datasheets/Stepper-Basic.pdf

The motor

Stepper motors are great motors for position control. They are a special type of brushless motors that divides a full rotation into a number of equal "steps".

When the 28BYJ-48 motor runs in full step mode, each step corresponds to a rotation of 11.25°. That means there are 32 steps per revolution (360°/11.25° = 32). In addition, the motor has a 1/64 reduction gear set. (Actually its 1/63.68395 but for most purposes 1/64 is a good enough approximation) What this means is that there are actually 32*63.68395 steps per revolution = 2037.8864 (2038) steps!

The driver board

Your motor comes with a ULN2003 based **driver board**. The ULN2003 is one of the most common motor driver ICs, consisting of an array of 7 Darlington transistor pairs, each pair is capable of driving loads of up to 500mA and 50V. Four out of seven pairs are used on this board. The board has a connector that mates the motor wires perfectly which makes it very easy to connect the motor to the board. There are also connections for four control inputs as well as power supply connections.

The board has four LEDs that show activity on the four control input lines (to indicate stepping state). They provide a nice visual when stepping.

Note that it is possible to directly power the stepper motor from the Arduino. However, this is not recommended; as the motor may induce electrical noise onto its power supply lines and this could damage the Arduino. So, if you can, use a separate 5V power supply to power your stepper motors.

DC Motor Assignment

Part 1

Connect two pushbuttons to the Arduino. Wire a circuit and write a program that increases (+10) the speed of the motor when the first button is pressed and decreases (-10) the speed of the motor when the second button is pressed. Be sure that the speed remains between feasible values (0-255). Display the current value of the speed variable on the serial plotter.

Part 2

For a fixed load, the shaft speed of a DC motor is proportional to the applied voltage. Wire a circuit and write a program that continuously estimates the speed of the motor and visualizes it for the user through the serial plotter (basically monitor the potential drop across the motor). To change the speed of the motor use the configuration in Part 1. If you have some, you can use a large capacitor to filter out the slowly changing variations in your output.

Stepper Motor Assignment

Part 1

The motor has two phases: IN1 will activate positive phase A, IN2 will activate positive phase B, and so on. The LEDs on the driver board will light up accordingly. In order to activate a phase it is sufficient to put in high the respective digital pin from the Arduino.

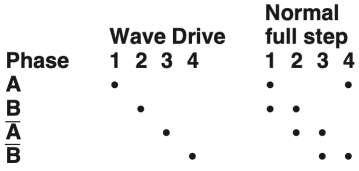

Figure 2: Two examples of driver modes

Design the circuit and write the code to drive the stepper motor, without using external libraries! Attach one external button to toggle between driving the motor in Normal full step mode and in Wave Drive mode. Attach a second external button to toggle the direction of motion of the motor, and a potentiometer to select the speed of the motor.

Show me on the video that the motor is working, make it obvious with the help of some tape. Also, when the motor is slow show me that the LEDs on the driven board light up in the correct order.

Part 2 (40 pt.)

Now, without changing your circuit, implement the same logic (except for changing the driver mode) using the `<Stepper.h>`. Use the following code as a reference but be sure to make the appropriate changes in the motor parameters!

```
/* Stepper Motor Control */

#include <Stepper.h>
const int stepsPerRevolution = 90;
// change this to fit the number of steps per revolution
// for your motor
// initialize the stepper library on pins 8 through 11:
Stepper myStepper(stepsPerRevolution, 8, 9, 10, 11);

void setup() {
   // set the speed at 60 rpm:
   myStepper.setSpeed(60);
   // initialize the serial port:
   Serial.begin(9600);
}

void loop() {
   // step one revolution in one direction:
   Serial.println("clockwise");
   myStepper.step(stepsPerRevolution);
   delay(500);
   // step one revolution in the other direction:
   Serial.println("counterclockwise");
   myStepper.step(-stepsPerRevolution);
   delay(500);
}
```