# EE 324
# Control Systems Lab
# Experiment 1
# DC Motor Position Control

**Tuesday Batch I**
**Group 6**

Abhin Shah - 140070013
Karan Chadha - 140070014
Kalpesh Krishna - 140070017

## AIM :-

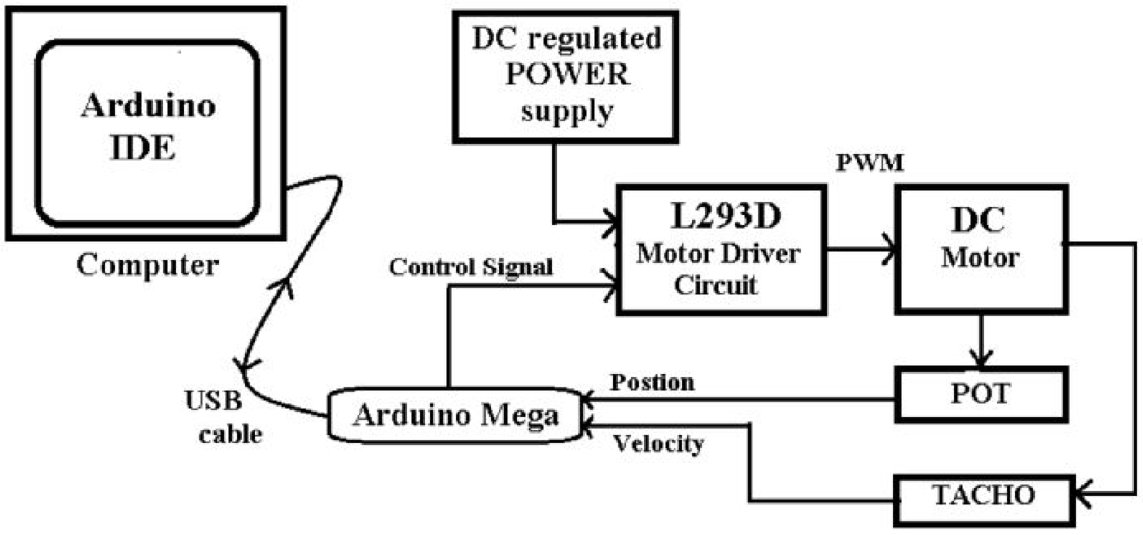
1. To design and implement an embedded Feedback control system using Arduino Mega
2. To design and implement a motor driver circuit and interface the Arduino Mega with this circuit using PID mechanism

This Feedback circuit is used to rotate the DC motor by a specified angle of 180 degrees with a less than 1 sec rise time, 1.5 sec settling time and 10% overshoot.

## CIRCUIT :-

The microcontroller used in the system is Arduino Mega, which gave the input control signal to the Motor Driver IC L293D circuit. Thus the Arduino controls the direction of rotation of the motor. The feedback output of the potentiometer that is connected to the motor which is an input to the Arduino specified its position.

**The circuit diagram is as shown:-**



**OBSERVATION & INFERENCES :-**

1. The Proportion Component P accounts for present values of the error. For example, if the error is large and positive, the control output will also be large and positive. It ensures a initial fast movement towards the final value to reduce the time taken and also slow moment near the steady state value.
2. The Integration Component I accounts for past values of the error. For example, if the current output is not sufficiently strong, the integral of the error will accumulate over time, and the controller will respond by applying a stronger action. It slows down the oscillations.
3. The Differentiation Component D accounts for possible future trends of the error, based on its current rate of change

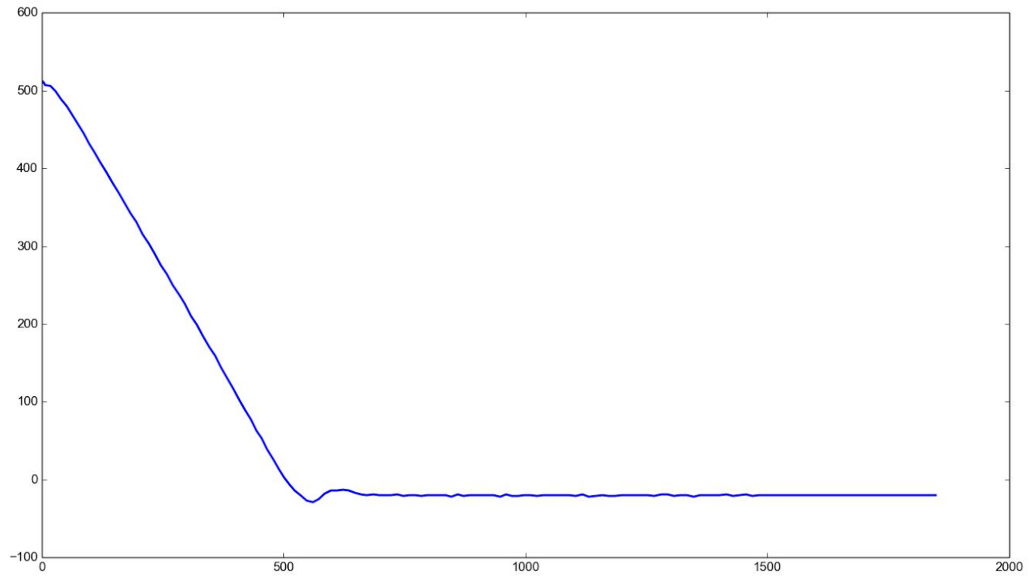
Values used in the system are:

1. Kp = 5
2. Ki = 0.001
3. Kd = 2

Time Response characteristics are:

1. Percentage Overshoot = 5.86%
2. Rise time = 0.375s
3. Settling time = 0.7s

**The time response plot is :-**



**The problems faces were:-**

1. We had not shorted the ground of the Arduino Mega and the L293D circuit
2. Our motor was moving only in one direction since we had only one connection from the motor POT to the Arduino.

3. Choosing the values of $K_p$, $K_I$ and $K_D$ properly:

On the first lab day, most of the time went into the correcting the circuit as said above. We randomly tried some values of $K_p$, $K_I$ and $K_D$ on that day, but it did not work out well. With just $K_P$, it was working well with the error within the given bounds but it was not a good solution. On adding other parameters, it was slowing down the convergence and there were small jerks after convergence. Before coming on the next day, we read up more on how to set the parameters here: http://electronics.stackexchange.com/questions/127524/how-to-calculate-kp-kd-and-ki.

The rule of thumb given here is as follows:

1.Raise Kp until the system's response (even with a little overshoot) is satisfyingly fast to track (quasi-) step changes in your setpoint. This proportional component of a PID defines the 'stiffness' of your control system's response.

2.Raise Kd until the system's response is adequately damped. You don't need this if you don't have an overshoot. This derivative component defines an artificial damping for your system.

3.Raise Ki until the steady-state error (which you will have) with respect to the setpoint is corrected fast enough, without affecting the initial dynamics too much.

4.Fine tune all gains.

Change all values slowly, as too large values may cause instabilities in the system.