AMERICAN INTERNATIONAL UNIVERSITY–BANGLADESH (AIUB)

FACULTY OF ENGINEERING

MICROPROCESSOR AND EMBEDDED SYSTEMS

Spring 2021-2022
Section: A
Group: 02
Lab Report No: 07

Submitted By:

Group Member's Name	ID
Hasan, Mahmud	17-33881-1
Islam, Md. Ariful	18-36842-1
Alam, Ifta khirul	18-36817-1
Mulk, MD. Abdullah Al Malikal	18-37803-2
Rahman, Md Ashikur	18-38519-2
Islam, Sheikh Md. Samiul	18-39261-3
Muhaiminul Islam	18-38920-3
Ema, Fahamida Tanjiya	18-38961-3

Title: Controlling a motor through the application of PWM.

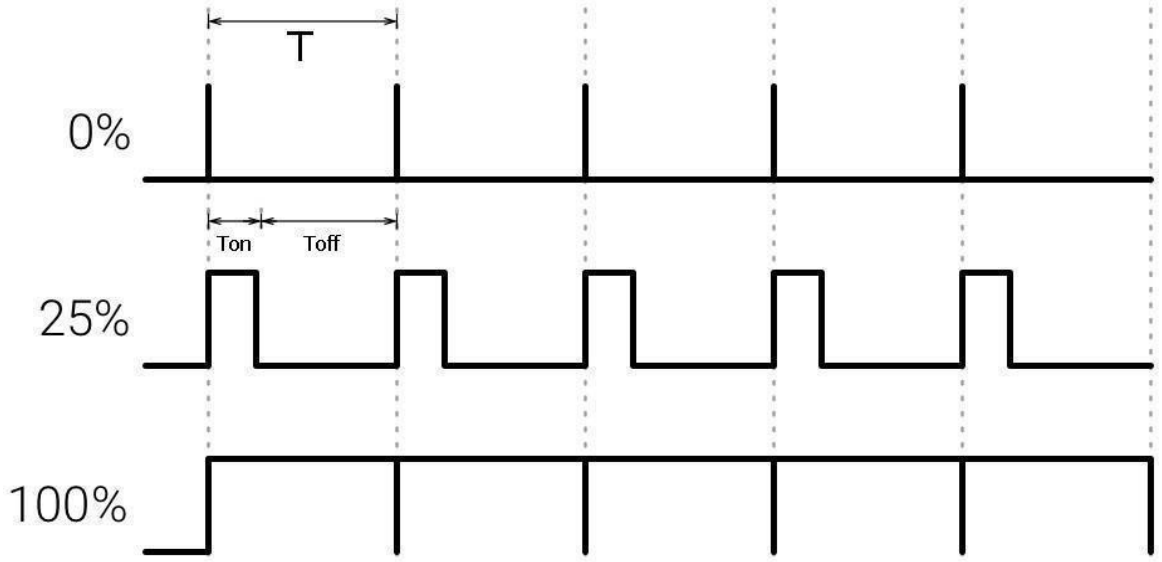
Introduction:

The objective of this experiment is to get familiarized with Microcontroller based motor speed control.

Theory and Methodology:

Microcontroller and Arduino are digital devices; they cannot give the analog output. Microcontroller gives Zero and ONE as output, where ZERO is logical LOW and ONE is logical HIGH. In our case, we are using 5 volt version of the Arduino. So it's logical ZERO is zero voltage, and logical HIGH is 5 voltage.

Digital output is good for digital devices but sometimes we need the analog output. In such a case the PWM is very useful. In the PWM, output signal switches between zero and one, on high and fixed frequency, as shown in the figure below.



Output Signal Of PWM

As shown in the above figure the ON time is T_{on} and the OFF time is T_{off}. **T is the sum of the T_{on} and T_{off} , which is called the Time Period.** In the concept of PWM, T is not varying and the T_{on} and the T_{off} can vary, in this way when T_{on} increase T_{off} will decrease and T_{off} increase when T_{on} decrease proportionally.

The duty cycle is the fraction of one Time period. Duty cycle is commonly expressed as a percentage or a ratio. A period is the time it takes for a signal to complete an on-and-off cycle. As a formula, a duty cycle may be expressed as:

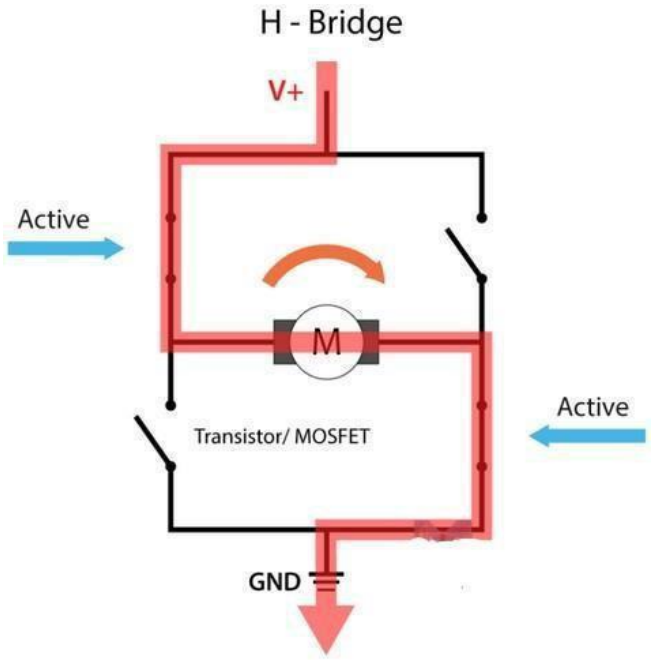
DUTY CYCLE = (Ton / T) x100 %

Now the motor speed varies according to duty cycle. Suppose the duty is zero, motor does not run and when duty cycle is 100 % the motor moves on maximum RPM**. But this concept is not always right because motor starts running after giving some fixed voltage that is called threshold voltage.**

Microcontroller and the Arduino can process signals and **consumes almost 20 to 40mA current but motors need high current and voltage, so we are using the transistor for driving the motor.** Transistor is connected in ***series*** with motor and ***transistor's base is connected to Arduino's PWM pin through a resistance.*** PWM signal is coming from Arduino and the transistor works as a switch and it **short circuits the Emitter (E) and Collector (C) when PWM signal is in High state** and ***normally opens when PWM signal is in LOW state***. This process works continuously and the motors runs at desired speed.

H-Bridge DC Motor Control

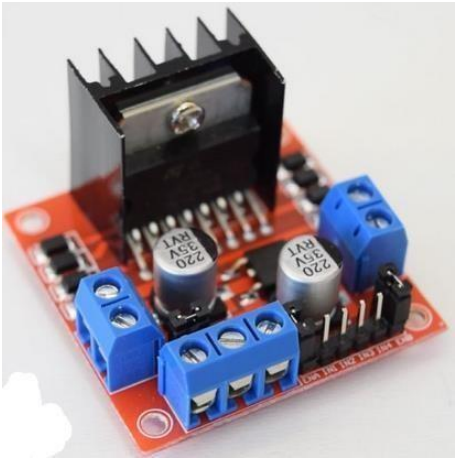
On the other hand, for controlling the rotation direction, we just need to inverse the direction of the current flow through the motor, and the most common method of doing that is by using an H-Bridge. An H-Bridge circuit contains four switching elements, transistors or MOSFETs, with the motor at the center forming an H-like configuration. **By activating two particular switches at the same time we can change the direction of the current flow, thus change the rotation direction of the motor.**



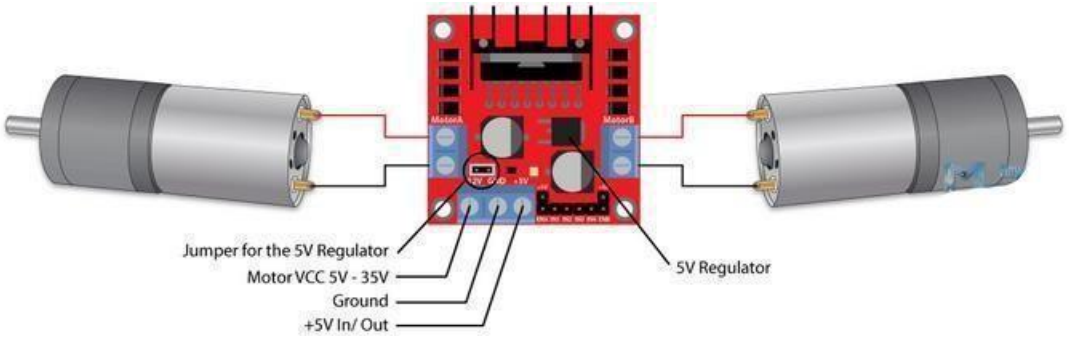
So if we combine these two methods, the PWM and the H-Bridge, we can have a complete control over the DC motor. There are many DC motor drivers that have these features and the L298N is one of them.

L298N Driver

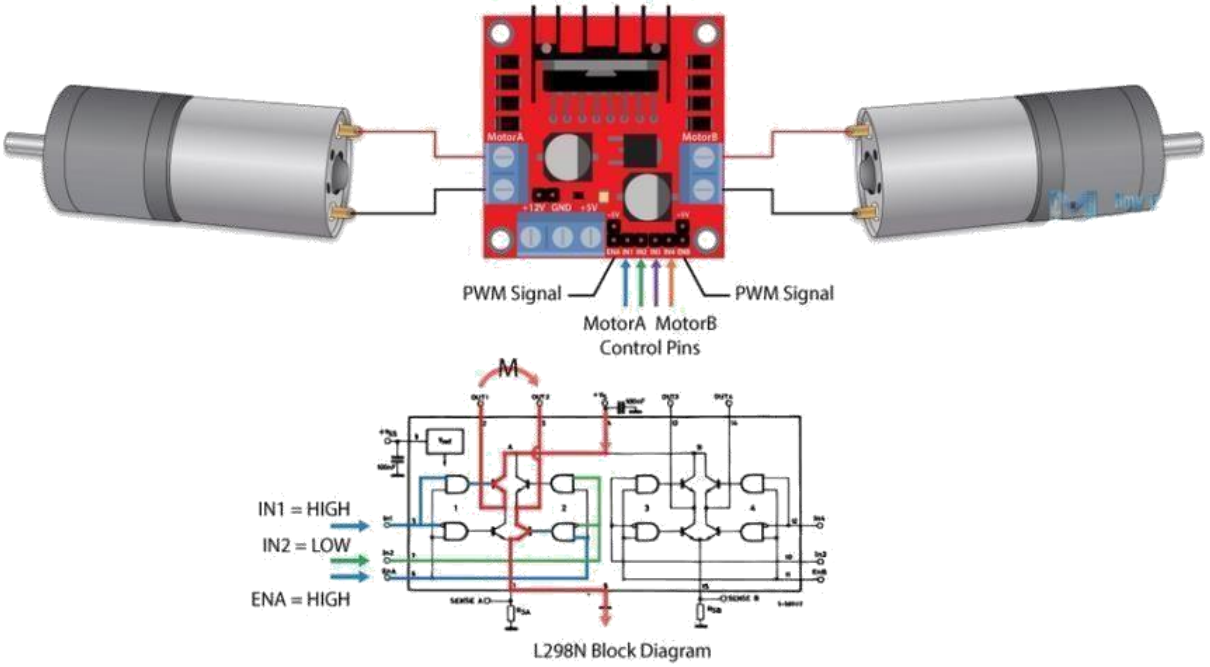
The L298N is a ***dual H-Bridge motor driver which allows speed and direction control of two DC motors at*** the same time. The module can drive DC motors that have voltages between 5 and 35V, with a peak current up to 2A.



Let's take a closer look at the pinout of L298N module and explain how it works. The module has two screw terminal blocks for the motor A and B, and another screw terminal block for the Ground pin, the VCC for motor and a 5V pin which can either be an input or output.



Next are the logic control inputs. The Enable A and Enable B pins are used for enabling and controlling the speed of the motor. ***If a jumper is present on this pin, the motor will be enabled and work at maximum speed, and if we remove the jumper we can connect a PWM input to this pin and in that way control the speed of the motor. If we connect this pin to a Ground the motor will be disabled.***

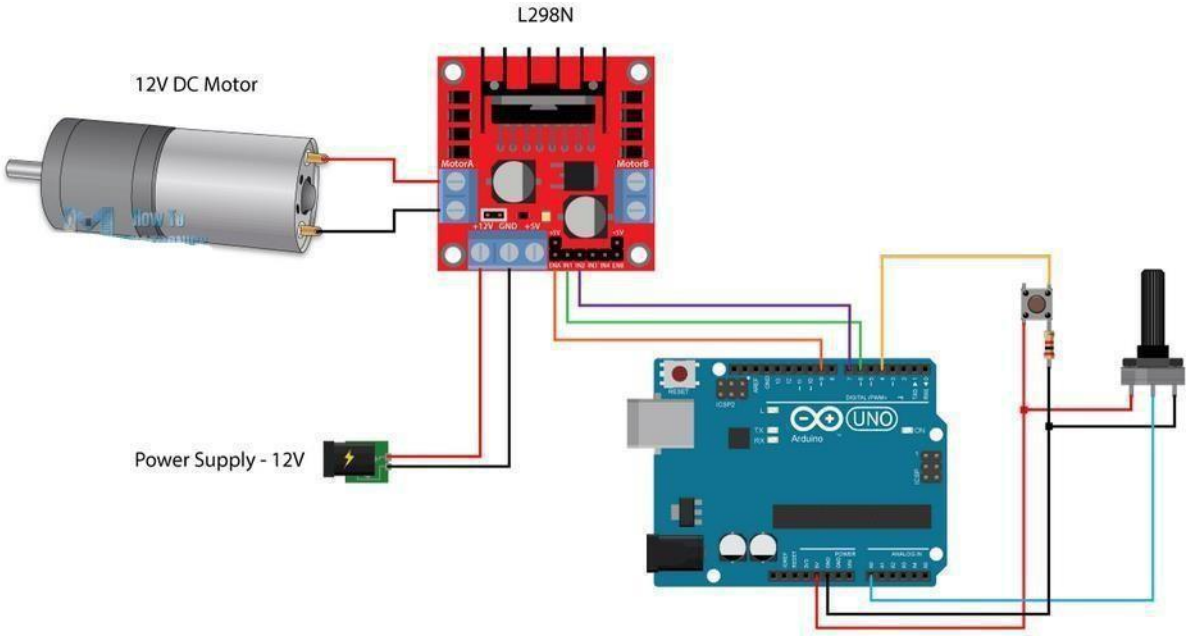


L298N Block Diagram

Next, the Input 1 and Input 2 pins are used for controlling the rotation direction of the motor A, and the inputs 3 and 4 for the motor B. Using these pins we actually control the switches of the H-Bridge inside the L298N IC. If input 1 is LOW and input 2 is HIGH the motor will move forward, and vice versa, if input 1 is HIGH and input 2 is LOW the motor will move backward. In case both inputs are same, either LOW or HIGH the motor will stop. The same applies for the inputs 3 and 4 and the motor B.

Arduino and L298N

Now let's make some practical applications. In the first example we will ***control the speed of the motor using a potentiometer*** and ***change the rotation direction using a push button***. Here's the circuit schematic diagram.

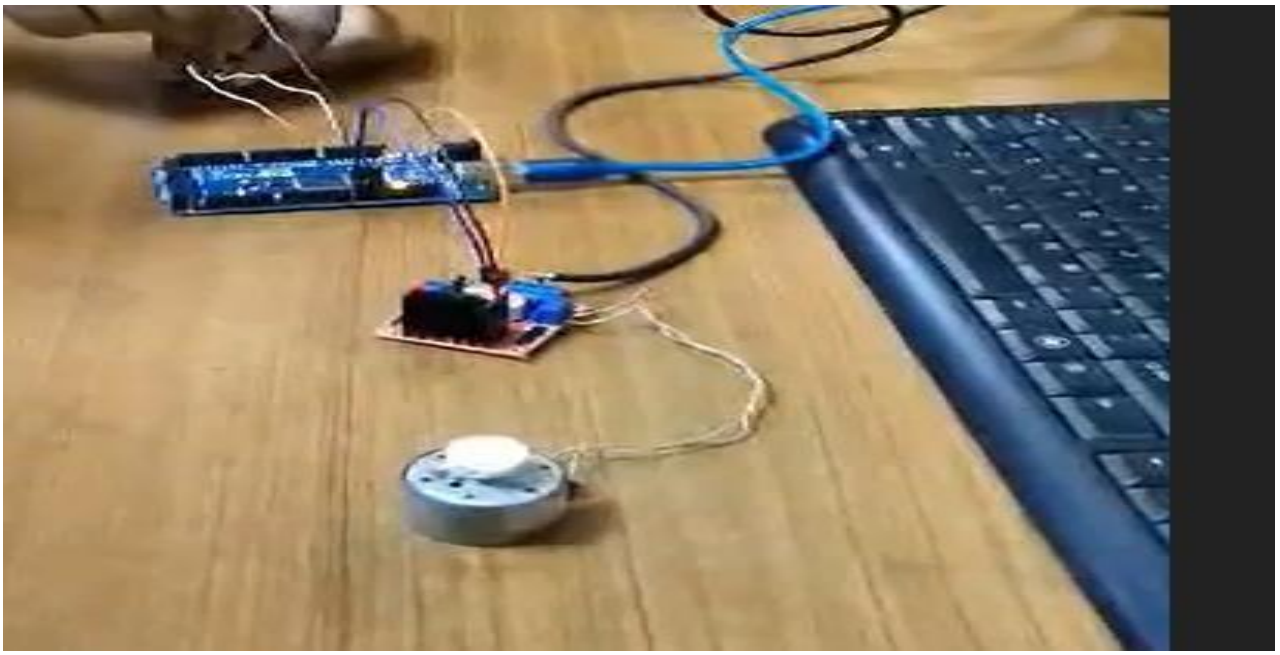


So we need an L298N driver, a DC motor, a potentiometer, a push button and an Arduino board.

Components List

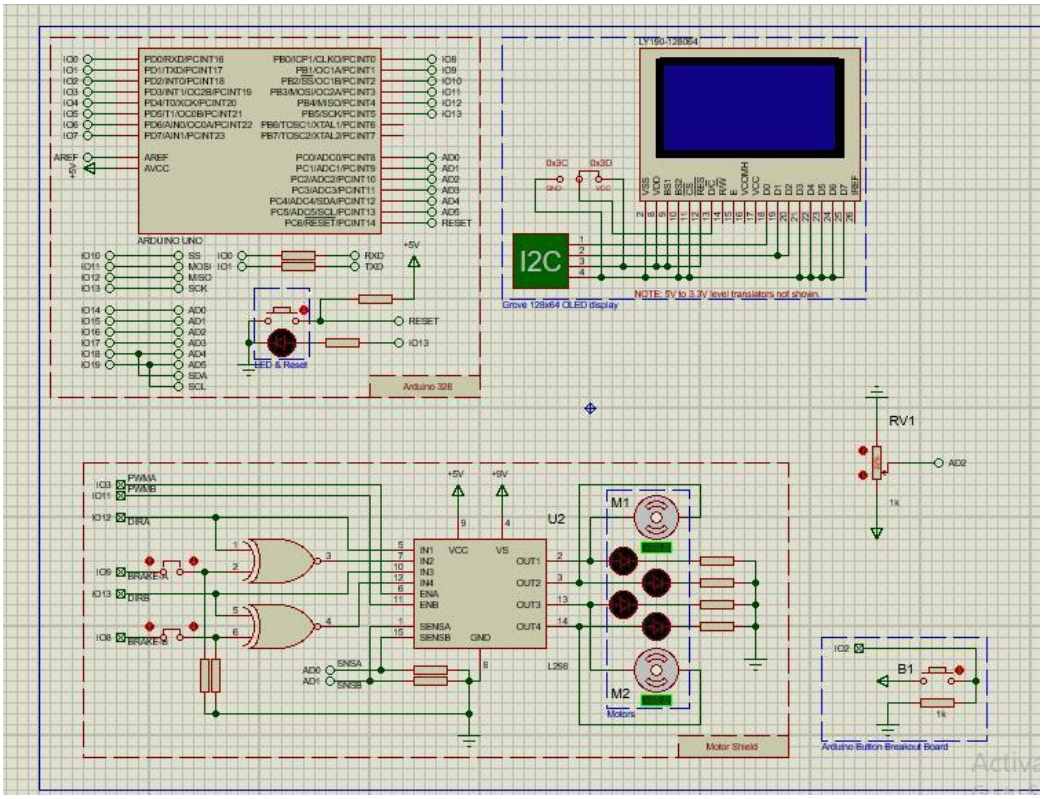
- L298N Driver
- 12V High Torque DC Motor
- Arduino Board
- Breadboard and Jump Wires

Experimental Figure:



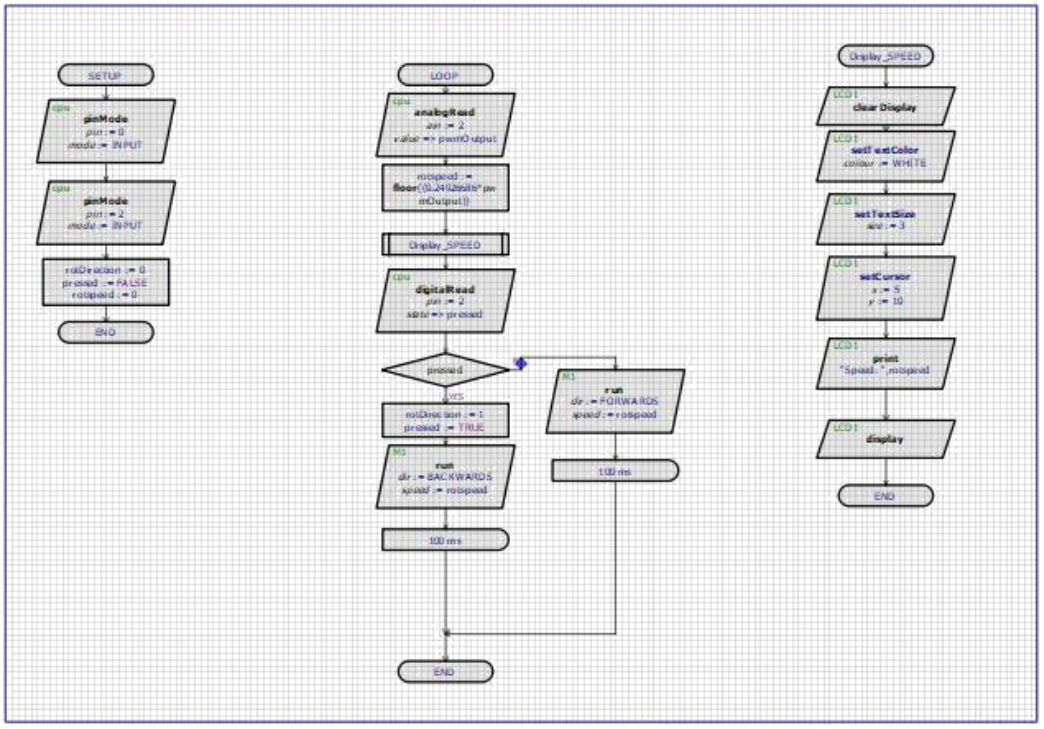
Experimental Procedure:

Simulation in Proteus:

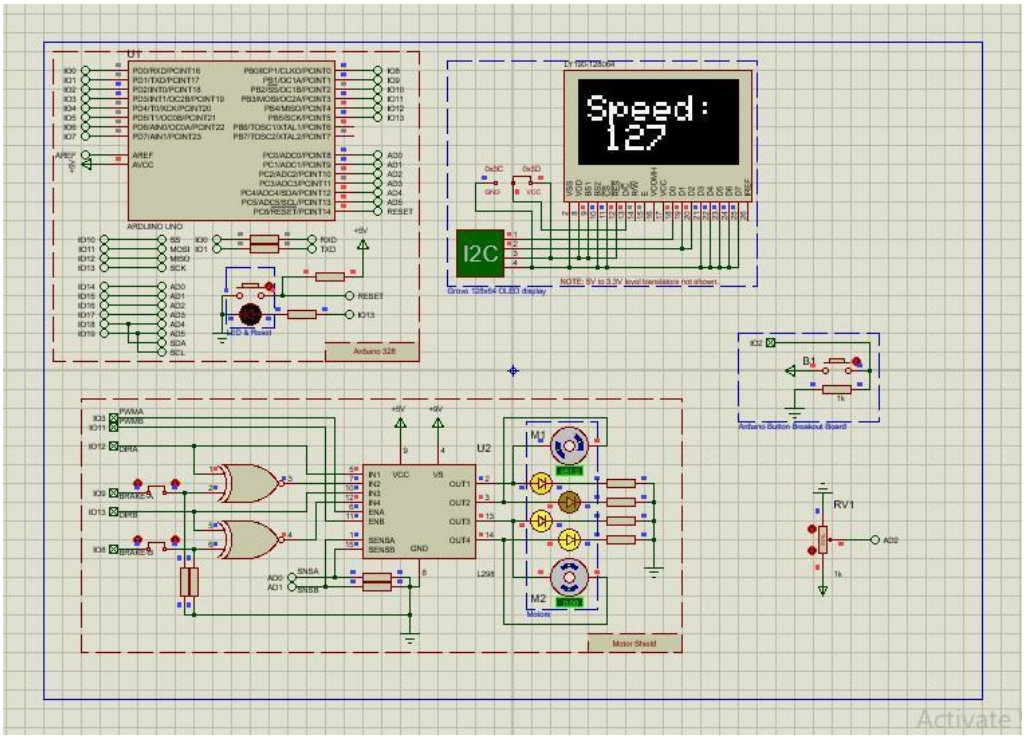


Flowchart for Proteus Visual Designer:

Thus the overall flowchart would look like as below:



Next build the project and click the play button in schematic capture. For duty cycle of 50% , motor speed and LCD output is shown below:



Thus, speed in LCD display along with speed of M1 motor (sign is +) can be varied by increasing or decreasing the percentage of duty cycle using POT-HG. When push button is pressed, the direction of Motor M1 gets altered with same speed (sign is minus(-)) .

Note that under the Motor Control heading there are actually two shields. These are identical except that the first has the board configured and populated with two DC motors and the second is configured and populated with one stepper motor. In the real hardware of course you would need to configure and populate the shield manually according to which project you were programming the Arduino with.

Conclusion: By knowing how components function, we were able to use them and grasp the purpose of each component's use as well as the associated values. We created a DC motor speed control system that is dependable, precise, and adaptive to a wide range of system ratting and reactions. It means that regardless of the load, the motor will run at a consistent speed. The speed does not alter when a particular amount of load is applied, and the software is designed to suit the needs of speed control. The designed method automatically manages the speed of the DC motor using the PWM approach. Basically, we tried to grasp the concept of PWM properly, and we used the Arduino analog output in this experiment.