**A**

**Mini – Project Report**

**on**

“Precision Speed Modulation of DC Motor Using Microcontroller 8051**”**

*Speed Control of DC motor using PWM*

**Submitted By**

|  |  |
| --- | --- |
| ADITYA PACHORE | 21410066 |
| VISHVAJEET JAGTAP | 21410059 |
| HARSH BANDGAR | 21410045 |
| TANUJ JOSHI | 21410051 |
| RAJWARDHAN VICHARE | 21410058 |

**Under the guidance of**

***Mr. S. R. Khedkar***



Department of Electronics Engineering

Walchand College of Engineering, Sangli

(Government-Aided Autonomous Institute)

MPI LAB mini Project A.Y. 2022-2023

**Abstract**

The aim of developing this project is to control the speed of DC motor. The main advantage in using a DC motor is that the Speed-Torque relationship can be varied to almost any useful form. To achieve the speed control an electronic technique called Pulse Width Modulation is used which generates High and Low pulses. These pulses vary the speed in the motor. For the generation of these pulses a microcontroller (AT89C51) is used. As a microcontroller is used setting the speed ranges as per the requirement is easy which is done by changing the duty cycles time period in the program. This project is practical and highly feasible in economic point of view, and has an advantage of running motors of higher ratings. This project gives a reliable, durable, accurate and efficient way of speed control of a DC motor.

The mini project "Speed Control of DC motor using PWM" involves the use of an 8051 microcontroller, an L298N motor driver module, a 12V DC motor, and four push buttons. As said, the objective of this project is to control the speed and direction of the DC motor using pulse-width modulation (PWM) technique.

The microcontroller is programmed to receive input signals from the push buttons and to generate the PWM signals for controlling the speed of the motor. The L298N module is used to provide the necessary current and voltage to the motor for its operation.

The four push buttons have different functions, such as increasing the speed of the motor, decreasing the speed, changing the direction of rotation, and stopping the motor. The PWM signals are adjusted based on the input from the push buttons, resulting in the desired speed and direction of the motor.

In addition to the components mentioned earlier, this mini project also includes an LCD interface. This interface displays the duty cycle of the PWM signal generated by the microcontroller, which corresponds to the speed of the motor.

The duty cycle represents the percentage of time that the PWM signal is high in one cycle. By adjusting the duty cycle, the average voltage applied to the motor can be controlled, thereby controlling its speed. The LCD interface provides real-time feedback on the duty cycle, allowing the user to accurately monitor and control the speed of the motor.

With the addition of the LCD interface, the project becomes more user-friendly and easier to operate. It enhances the overall functionality and usefulness of the project, making it a valuable tool for experimentation and learning in the field of microcontroller-based motor control.

Overall, this project demonstrates the practical application of PWM technique and microcontroller-based control of DC motors, which can be useful in various industrial and automation applications.

**Contents**

**Chapter 1**

* 1. **Introduction**
  2. **Project Idea**
  3. **Problem definition**
  4. **Objectives**
  5. **Expected outcomes**

**Chapter 2**

**Literature Survey**

**Chapter 3**

**3.1 System block diagram**

**3.2 Block description**

**3.3 Circuit diagram**

**3.4 List of Components**

**Chapter 4**

**4.1 Mathematical Modelling**

**4.2.1 Objective 1: Configure L298n and 8051 microcontroller.**

**4.1.2 Objective 2: Design algorithm to generate PWM for speed control of DC Motor.**

**4.1.3 Objective 3: Design and implement a user interface for speed control of the DC motor.**

**4.2 Final Program**

**4.3 Final Proteus simulation**

**Chapter 5**

**Result and Testing**

**5.1 Objective 1**

**5.2 Objective 2**

**5.3 Objective 3**

**5.4 System output**

**Chapter 6**

**Conclusion & Future Scope**

**Chapter 1**

**1.1 Introduction**

DC motor speed control using PWM (Pulse Width Modulation) method is a commonly used technique in many industrial and consumer applications. PWM is a way to vary the average voltage applied to the DC motor by varying the duty cycle of a square wave signal. In this project, we will be using an 8051 microcontroller and L298N motor driver to implement PWM-based speed control of a DC motor.

The 8051 microcontroller is a popular choice for many embedded system applications due to its low cost, ease of use, and availability of a wide range of development tools. The L298N motor driver is a commonly used driver IC for controlling DC motors as it can drive two motors simultaneously and can handle up to 2A of current per channel.

The PWM signal is generated by the 8051-microcontroller using one of its output pins. The duty cycle of the PWM signal is controlled by varying the value of a variable resistor connected to one of the input pins of the microcontroller. The PWM signal is then fed to the L298N motor driver which in turn drives the DC motor at the desired speed.

To display the duty cycle of the PWM signal, an LCD is interfaced with the microcontroller. The duty cycle is displayed as a percentage on the LCD screen. The LCD is connected to the microcontroller using the standard 8-bit data interface.

**1.2 Project Idea**

The project idea for DC motor speed control using PWM method in 8051-L298N interface is to build a system that can control the speed of a DC motor using Pulse Width Modulation (PWM) method. The system will use an 8051 microcontroller to generate the PWM signal using 8051 timer interrupt and an L298N motor driver to drive the DC motor. The duty cycle of the PWM signal will be controlled using switches connected to the microcontroller. An LCD will display the duty cycle percentage of the PWM signal. This project can be used as a learning tool for microcontroller-based motor control and interfacing with LCD displays. It can also be applied in real-world applications such as in robotics, automation, and industrial control systems.

**1.3 Problem definition**

The design of the 8051-L298N interface solves the problem of DC motor speed control by the PWM method, which is to accurately and efficiently control the DC motor speed in various applications.

DC motors are commonly used in many industrial and consumer applications, and their speed control is crucial to optimize their performance and efficiency. Traditional methods of DC motor speed control, such as varying the voltage or resistance, are less efficient and less precise compared to PWM-based control. Therefore, this project aims to develop a system that can precisely control the speed of a DC motor using PWM method, which will result in better performance, efficiency, and control accuracy.

**1.4 Objectives**

1. Configure L298n and 8051 microcontroller.
2. Design algorithm to generate PWM for speed control of DC Motor.
3. Design and implement a user interface for speed control of the DC motor.

**1.5 Expected Outcomes**

The LCD screen will show the current speed of the motor as well as the direction of rotation. The microcontroller is responsible for processing the input signal from the buttons, generating the appropriate PWM signal using the timer and sending the signal to the motor driver.

The outcomes of this project to control the speed of a DC motor using an 8051 microcontroller are significant. Here are some of the possible outcomes:

* A working system: The successful completion of the project would result in a working system that can control the speed and direction of a DC motor with a high degree of accuracy and precision.
* Increased efficiency: With the ability to control the speed of the motor, the system can achieve increased efficiency and reduce energy consumption.
* Versatility: The system can be used in a wide range of applications that require precise speed control of DC motors, such as robotics, automation, and manufacturing.
* Learning opportunity: This project can serve as a valuable learning opportunity for students, who are interested in microcontroller programming, motor control, and electronic circuit design.
* Customizability: The system can be customized to suit specific needs and requirements, such as changing the control algorithm, adding sensors for feedback control, or integrating with other systems.

**Chapter 2**

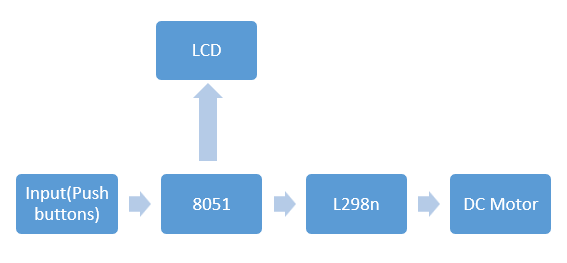
**Literature Survey**

* Y. S. E. Ali, S. B. M. Noor, S. M. Uashi and M. K Hassan” Microcontroller Performance for DC MotorSpeed Control” O-7803-8208©2003 IEEE.
* Speed Control Of DC Motor Using Analog PWMTechnique Nandkishor P. Joshi,Member ISTE1 , Ajay P.Thakare,Member IEEE,Member ISTE2
* N. Milivojevic, Mahesh Krishnamurthy and YusufGurkaynak,” Stability Analysis of FPGA- Based Control of Brushless DC Motors and Generators Using Digital PWMTechnique”, IEEE Transactions on Industrial Electronics,Vol. 59, no. 1, January 2012 [3] Hong Wong and Vikram Kapila, “Internet-Based Remote Control of a DC Motorusing an Embedded Ethernet Microcontroller”

**Chapter 3**

**Design Methodology**

**3.1 System Block Diagram**



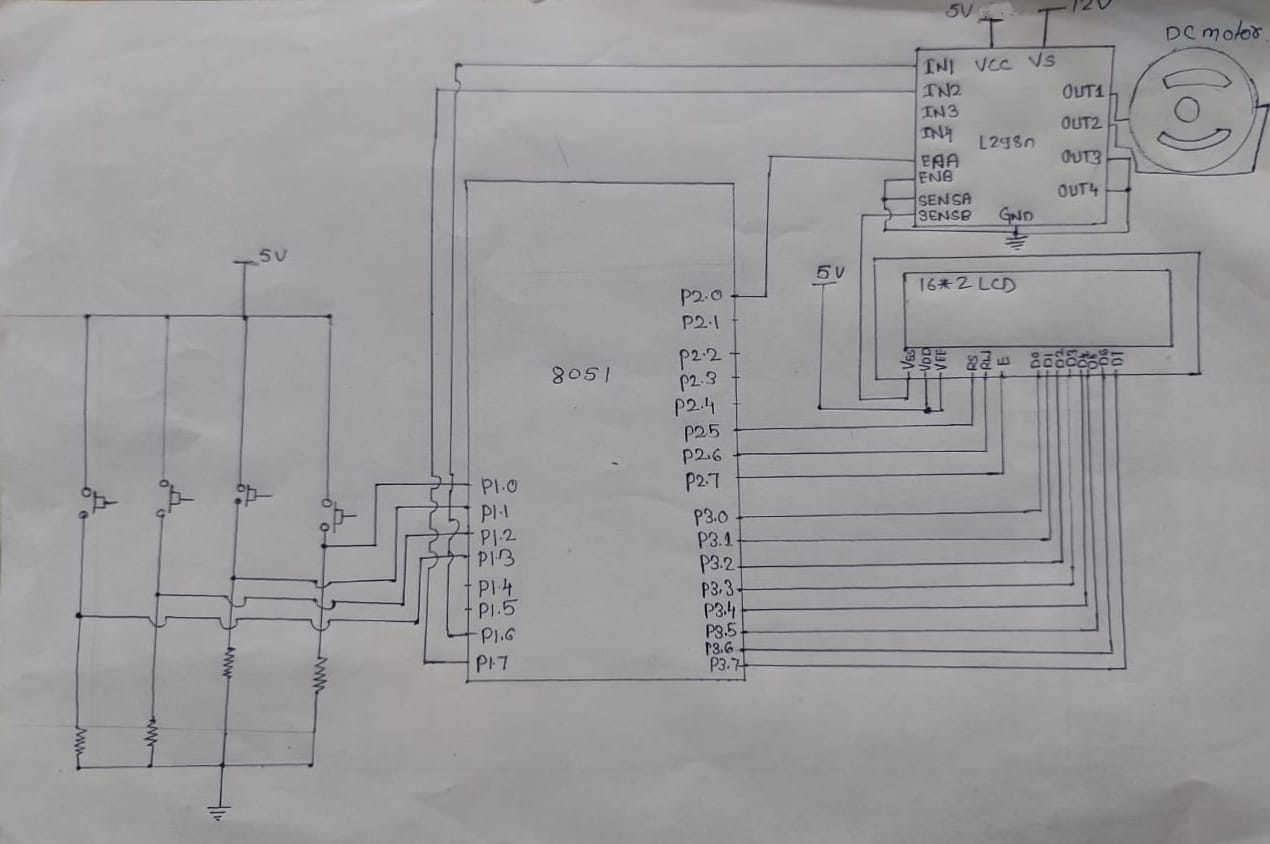
**3.2 Block Description**

1. Input (Push Buttons): First the input is provided to 8051 in the form of push buttons. There are 4 push buttons, each having different function. Push buttons are used for:

* Increasing speed
* Decreasing speed
* Changing direction of rotation
* Start/Stop of the motor

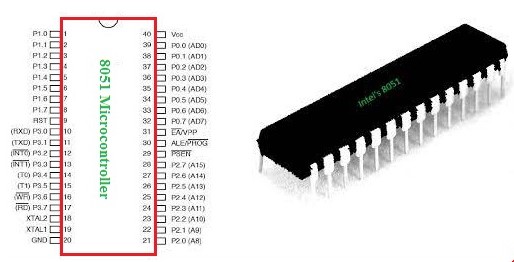
1. 8051: The main CPU or controlling unit of this project is 8051-microcontroller. It takes the input from push buttons, provides instructions to l298n and send data to LCD accordingly.
2. LCD: It takes the input from 8051. The data changes according to the instructions sent to the motor driver l298n.
3. L298n: It is a motor driver which provides variable power to the DC motor according to the instruction sent by 8051.
4. DC Motor: It rotates by taking the input power from L298n and its speed changes according to the power supplied by the L298n.

**3.3 Circuit Diagram**



**3.4 List of component**

1)Microcontroller-8051:



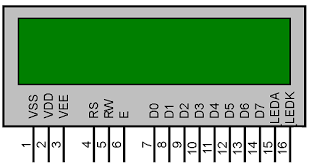
2) DC motor:



3) L298n Motor Driver :



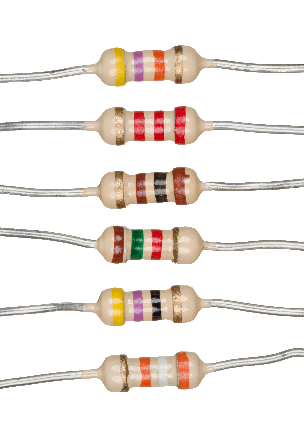
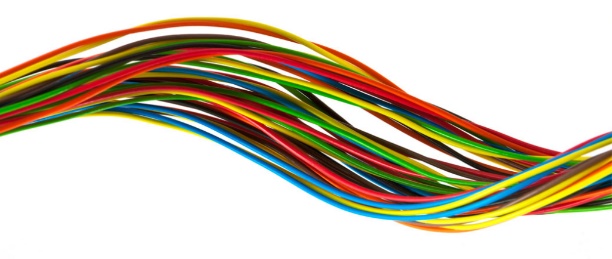
4) 16 \* 2 LCD:



5) PUSH BUTTON :



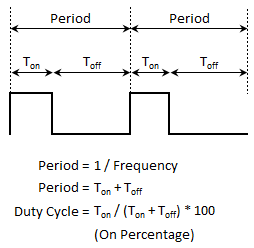
6) Resistors, wires and Power supply.

**Chapter 4**

**Implementation**

**4.1 Mathematical modelling**



Period = 1/Frequency

Period = TON + TOFF = T

Duty Cycle = (TON / TON + TOFF) \* 100

Period = 65535 – PWM Period

ON Period = ((period/100.0) \* duty cycle)

OFF Period = (period – ON Period)

ON Period = 65535 – ON Period

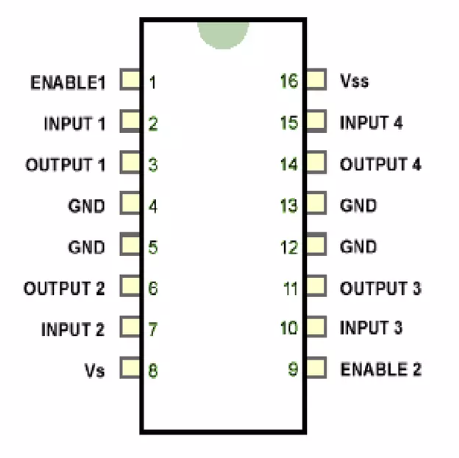
OFF Period = 65535 – OFF Period

**4.1.1** **Objective 1: Configure L298n and 8051 microcontroller.**

**Explanation:**

**L298n Motor Driver:**

**Pin configuration:**

****

|  |  |  |
| --- | --- | --- |
| **Pin No.** | **Pin Name** | **Description** |
| **Power Supply Pins** | | |
| 1 | **VCC** | VCC pin is used to supply power to the motor. Its input voltage is between 5 to 35V. |
| 2 | **GND** | GND is a ground pin. It needs to be connected to the power supply ground(negative). |
| 3 | **+5V** | +5V pin supplies power for the switching logic circuitry inside the L298N IC. If the 5V-EN jumper is in place, this pin acts as output and can be used to power up a microcontroller or other circuitry (sensor). If the 5V-EN jumper is removed, you need to connect it to the 5V power supply of the microcontroller. |
| **Control Pins** | | |
| 1 | **IN1** | These pins are input pins of **Motor A**. These are used to control the rotating direction of Motor A. When one of them is HIGH and the other is LOW, Motor A will start rotating in a particular direction. If both the inputs are either HIGH or LOW the Motor A will stop. |
| 2 | **IN2** |
| 3 | **IN3** | These pins are input pins of **Motor B**. These are used to control the rotating direction of Motor A. When one of them is HIGH and the other is LOW, Motor A will start rotating in a particular direction. If both the inputs are either HIGH or LOW the Motor A will stop. |
| 4 | **IN4** |
| **Speed Control Pins** | | |
| 1 | **ENA** | ENA pin is used to control the speed of **Motor A**. If a jumper is present on this pin, so the pin connected to +5 V and the motor will be enabled, then the Motor A rotates maximum speed.  if we remove the jumper, we need to connect this pin to a PWM input of the microcontroller. In that way, we can control the speed of Motor A. If we connect this pin to Ground the Motor A will be disabled. |
| 2 | **ENB** | ENB pin is used to control the speed of **Motor B**. If a jumper is present on this pin, so the pin connected to +5 V and the motor will be enabled, then the Motor B rotates maximum speed.  if we remove the jumper, we need to connect this pin to a PWM input of the microcontroller. In that way, we can control the speed of Motor B. If we connect this pin to Ground the Motor B will be disabled. |
| **Output Pins** | | |
| 1 | **OUT1** & **OUT2** | This terminal block will provide the output for **Motor A**. |
| 2 | **OUT3** & **OUT4** | This terminal block will provide the output for **Motor B** |

**How Motor Driver Module Works:**

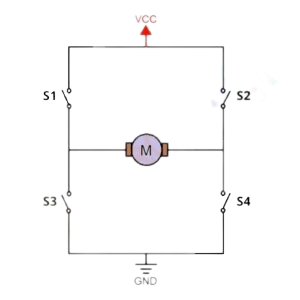
This module uses two techniques for the control speed and rotation direction of the DC motors. These are H-Bridge – For controlling rotation direction and PWM – For controlling the speed.

**H-Bridge Techniques:**  
L298n motor driver module uses the H-Bridge technique to control the direction of rotation of a DC motor. In this technique, H-Bridge controlled DC motor rotating direction by changing the polarity of its input voltage.

An H-Bridge circuit contains four switching elements, like transistors (BJT or MOSFET), with the motor at the center forming an H-like configuration. Input**IN1, IN2, IN3, and IN4** pins actually control the **switches** of the H-Bridge circuit inside L298N IC.  
We can change the direction of the current flow by activating two particular switches at the same time, this way we can change the rotation direction of the motor.

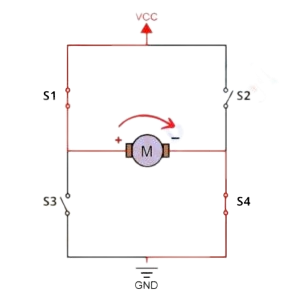
**Case 1**

When S1, S2, S3, and S4 all switches are open then no current goes to the Motor terminals. So, in this condition, the motor is stopped (not working).



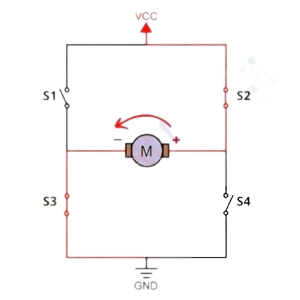
**Case 2**

When the switch S1 and S4 are closed, then the motor left terminal is getting a positive (+) voltage and the motor right terminal is getting a negative(-) voltage. So, in this condition motor start rotating in a particular direction (clockwise).



**Case 3**

When S2 and S3 switches are closed, then the right motor terminal is getting a positive (+) voltage and the left motor terminal is getting a negative (-) voltage. So, in this condition motor start rotating in a particular direction (anticlockwise).



**PWM (Pulse Width Modulation) Techniques**

L298n motor driver module uses the PWM technique to control the speed of rotation of a DC motor. In this technique, the speed of a DC motor can be controlled by changing its input voltage.

Pulse Width Modulation is a technique where the average value of the input voltage is adjusted by sending a series of ON-OFF pulses. The average voltage is proportional to the width of the pulses, these pulses known as Duty Cycle.

If the duty cycle higher, then the average voltage is applied to the DC motor (High Speed), and the lower the duty cycle, the less the average voltage being applied to the dc motor(Low Speed).

**Module Specifications & Features:**

|  |  |
| --- | --- |
| **Parameter** | **Value** |
| Operating Voltage | 5V – 46V |
| Operating Current | 2A |
| Logic Voltage | 5V |
| Logical Current | 0-36mA |
| Maximum Power (W) | 25W |
| Driver Chip | L298 dual-channel H-Bridge motor driver IC |
| LED lights indicators | Power-On LED indicator |
| Drives motor | Drives up to 4 motors (2 for each motor output terminal block) or One Stepper Motor |
| Module Dimensions | 44 x 44 x 28 (LxWxH)mm |

**Motor Driver Truth Tables**

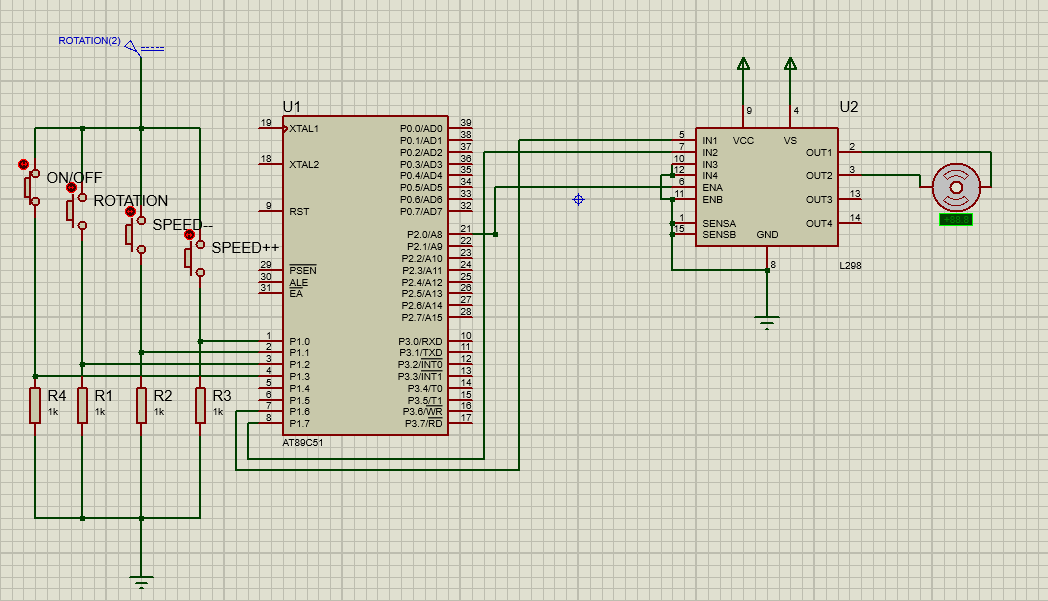
| **Motor A truth table** | | | |
| --- | --- | --- | --- |
| **ENA** | **IN1** | **IN2** | **Description** |
| **0** | **N/A** | **N/A** | **Motor A is off** |
| **1** | **0** | **0** | **Motor A is stopped (brakes)** |
| **1** | **0** | **1** | **Motor A is on and turning backwards** |
| **1** | **1** | **0** | **Motor A is on and turning forwards** |
| **1** | **1** | **1** | **Motor A is stopped (brakes)** |

| **Motor B truth table** | | | |
| --- | --- | --- | --- |
| **ENB** | **IN3** | **IN4** | **Description** |
| **0** | **N/A** | **N/A** | **Motor B is off** |
| **1** | **0** | **0** | **Motor B is stopped (brakes)** |
| **1** | **0** | **1** | **Motor B is on and turning backwards** |
| **1** | **1** | **0** | **Motor B is on and turning forwards** |
| **1** | **1** | **1** | **Motor B is stopped (brakes)** |

**Applications**

* Control DC motors.
* Control stepping motors
* In Robotics

**To configure L298n and 8051 Microcontroller, you need to follow these steps:**

****

1. Connect the L298n motor driver to the 8051 microcontroller:
2. Connect the pins of the L298n to the corresponding pins of the 8051 microcontroller.
3. Connect the power supply (5V or 12V) to the L298n motor driver.
4. Configure the L298n motor driver:
5. Set the enable pin of the L298n to high to enable the motor driver.
6. Set the direction pins (IN1 and IN2) of the L298n to control the direction of the motor.
7. Program the 8051 Microcontroller:
8. Write a program to control the direction and speed of the motor.
9. Use the GPIO pins of the 8051 microcontroller to control the L298n motor driver.
10. Test the circuit:
11. Apply a voltage to the motor driver and check the direction and speed of the motor.

**Algorithm:**

1. Define value to be loaded in timer for PWM period of 20 milli second
2. Assign pins of 8051 to IO pins of l298n
3. Use timer interrupt to generate square wave of different frequencies
4. Use ‘if’ statements to give instructions like increase, decrease of speed of change direction of rotation

**Program to achieve objective 1:**

#include <REG51.H>

#include <STDIO.H>

#include <intrins.h>

/\* Define value to be loaded in timer for PWM period of 20 milli second \*/

#define PWM\_Period 0xB7FE

sbit PWM\_Out\_Pin = P2^0; /\* PWM Out Pin for speed control \*/

sbit Speed\_Inc = P1^0; /\* Speed Increment switch pin \*/

sbit Speed\_Dec = P1^1; /\* Speed Decrement switch pin \*/

sbit Change\_Dir = P1^2; /\* Rotation direction change switch pin \*/

sbit CKT\_ON = P1^3; /\* Circuit ON/Off\*/

sbit M1\_Pin1 = P1^6; /\* Motor Pin 1 \*/

sbit M1\_Pin2 = P1^7; /\* Motor Pin 2 \*/

void delay(unsigned int count)

{

int i,j;

for(i=0; i<count; i++)

{

for(j=0; j<112; j++);

}

}

unsigned int ON\_Period, OFF\_Period, DutyCycle;

float Speed;

int rotation(int dir)

{

if(dir==1)

return 0;

else

return 1;

}

/\* Function to provide delay of 1ms at 11.0592 MHz \*/

void Timer\_init()

{

TMOD = 0x01; /\* Timer0 mode1 \*/

TH0 = (PWM\_Period >> 8);/\* 20ms timer value \*/

TL0 = PWM\_Period;

TR0 = 1; /\* Start timer0 \*/

}

/\* Timer0 interrupt service routine (ISR) \*/

void Timer0\_ISR() interrupt 1

{

PWM\_Out\_Pin = !PWM\_Out\_Pin;

if(PWM\_Out\_Pin)

{

TH0 = (ON\_Period >> 8);

TL0 = ON\_Period;

}

else

{

TH0 = (OFF\_Period >> 8);

TL0 = OFF\_Period;

}

}

void Set\_DutyCycle\_To(float duty\_cycle,int pin)

{

float period = 65535 - PWM\_Period;

ON\_Period = ((period/100.0) \* duty\_cycle);

OFF\_Period = (period - ON\_Period);

ON\_Period = 65535 - ON\_Period;

OFF\_Period = 65535 - OFF\_Period;

}

void Motor\_Init()

{

Speed = 0;

M1\_Pin1 = 1;

M1\_Pin2 = 0;

Set\_DutyCycle\_To(Speed,M1\_Pin1);

}

void main()

{

int count=1;

unsigned char a[20];

unsigned int i = 0;

Speed\_Inc =0;

Speed\_Dec =0;

Change\_Dir =0;

CKT\_ON = 0;

EA = 1; /\* Enable global \_interrupt \*/

ET0 = 1; /\* Enable timer0 \_interrupt \*/

Timer\_init();

Motor\_Init();

while(CKT\_ON==1){

{

M1\_Pin1 =1;

}

/\* Increment Duty cycle i.e. speed by 10% for Speed\_Inc Switch \*/

if(Speed\_Inc == 1)

{

if(Speed < 100)

Speed += 10;

Set\_DutyCycle\_To(Speed,M1\_Pin1);

while(Speed\_Inc == 1);

delay(200);

}

/\* Decrement Duty cycle i.e. speed by 10% for Speed\_Dec Switch \*/

if(Speed\_Dec == 1)

{

if(Speed > 0)

Speed -= 10;

Set\_DutyCycle\_To(Speed,M1\_Pin1);

while(Speed\_Dec == 1);

delay(200);

}

/\* Change rotation direction for Change\_Dir Switch \*/

if(Change\_Dir == 1)

{

M1\_Pin1 = !M1\_Pin1;

M1\_Pin2 = !M1\_Pin2;

while(Change\_Dir == 1);

delay(200);

}

}

while(CKT\_ON == 0){

M1\_Pin1 =0;

}

}

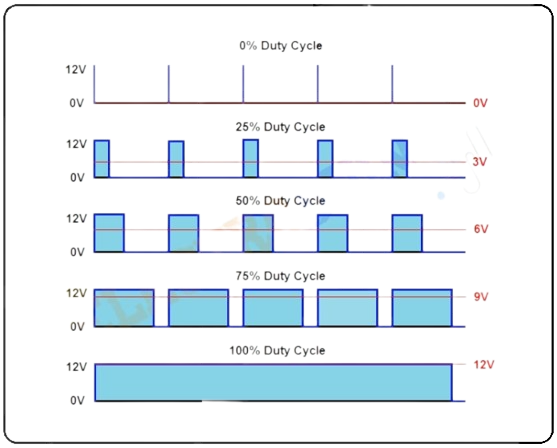
**4.1.2 Objective 2: Design algorithm to generate PWM for speed control of DC Motor.**

Explanation:

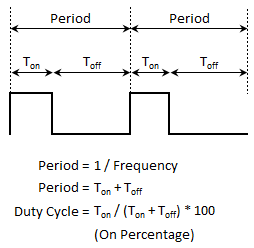
L298n motor driver module uses the PWM technique to control the speed of rotation of a DC motor. In this technique, the speed of a DC motor can be controlled by changing its input voltage.

Pulse Width Modulation is a technique where the average value of the input voltage is adjusted by sending a series of ON-OFF pulses. The average voltage is proportional to the width of the pulses, these pulses known as Duty Cycle.

If the duty cycle higher, then the average voltage is applied to the DC motor (High Speed), and the lower the duty cycle, the less the average voltage being applied to the dc motor(Low Speed).



Mathematical equation and its explanation:



Period = 1/Frequency

Period = TON + TOFF = T

Duty Cycle = (TON / TON + TOFF) \* 100

* The average voltage depends on the duty cycle in a single period of time
* Wider the ON time (i.e., large duty cycle), higher the speed of motor

Algorithm:

1. Declare a function having one float for duty cycle and one integer for setting ENA pin 1 / 0
2. For 8051:

Period = 65535 – PWM Period

ON Period = ((period/100.0) \* duty cycle)

OFF Period = (period – ON Period)

ON Period = 65535 – ON Period

OFF Period = 65535 – OFF Period

1. Now call this function in main function providing the values of float and integer

**4.1.3 Objective 3: Design and implement a user interface for speed control of the DC motor.**

Explanation:

Basic structure to implement any User Interface (UI)

The program initializes the necessary hardware components including the timer and LCD. The program enters a while loop that continuously reads the input from the switches to adjust the motor speed and direction.

Within the while loop, it checks the state of the Speed\_Inc switch. If it is pressed, the program increases the speed of the motor by 10% and updates the duty cycle of the PWM signal accordingly. And then waits until the switch is released before continuing.

Similarly, the program checks the state of the Speed\_Dec switch. If it is pressed, the program decreases the speed of the motor by 10% and updates the duty cycle of the PWM signal accordingly. It then waits until the switch is released before continuing.

The program also checks the state of the Change\_Dir switch. If it is pressed, the program changes the direction of the motor and updates the duty cycle of the PWM signal accordingly. The program then waits until the switch is released before continuing.

The program updates the LCD display to show the current duty cycle of the PWM signal and the current direction of the motor. It loops back to step 2 and continues reading the switch inputs to adjust the motor speed and direction.

Algorithm:

1. Load timer for PWM Period of 20 ms
2. Set pins of 8051 for functions like PWM output, speed increase/decrease, change direction, circuit on/off, LCD Pins
3. Define all the functions required to initialize LCD, motor and motor driver
4. Add ISR
5. In main function enable timer and call all the functions as per need
6. Use ‘if’ statements to give instructions like increase, decrease of speed of change direction of rotation

**4.2 Final Program**

#include <REG51.H>

#include <STDIO.H>

#include <intrins.h>

/\* Define value to be loaded in timer for PWM period of 20 milli second \*/

#define PWM\_Period 0xB7FE

sbit PWM\_Out\_Pin = P2^0; /\* PWM Out Pin for speed control \*/

sbit Speed\_Inc = P1^0; /\* Speed Increment switch pin \*/

sbit Speed\_Dec = P1^1; /\* Speed Decrement switch pin \*/

sbit Change\_Dir = P1^2; /\* Rotation direction change switch pin \*/

sbit CKT\_ON = P1^3;

sbit M1\_Pin1 = P1^6; /\* Motor Pin 1 \*/

sbit M1\_Pin2 = P1^7; /\* Motor Pin 2 \*/

sfr LCD\_DATA = 0xB0 ;

sbit RS = P2^5;

sbit RW = P2^6;

sbit E = P2^7;

void delay(unsigned int count)

{

int i,j;

for(i=0; i<count; i++)

for(j=0; j<112; j++);

}

void lcd\_cmd(unsigned char command)

{

LCD\_DATA = command;

RS = 0;

RW = 0;

E = 1;

delay(10);

E = 0;

}

void lcd\_data(unsigned char disp\_data)

{

LCD\_DATA = disp\_data;

RS = 1;

RW = 0 ;

E = 1;

delay(10);

E = 0;

}

void lcd\_init()

{

lcd\_cmd(0x38);

delay(10);

lcd\_cmd(0x0c);

delay(10);

lcd\_cmd(0x80);

delay(10);

}

unsigned int ON\_Period, OFF\_Period, DutyCycle;

float Speed;

int rotation(int dir)

{

if(dir==1)

return 0;

else

return 1;

}

/\* Function to provide delay of 1ms at 11.0592 MHz \*/

void Timer\_init()

{

TMOD = 0x01; /\* Timer0 mode1 \*/

TH0 = (PWM\_Period >> 8);/\* 20ms timer value \*/

TL0 = PWM\_Period;

TR0 = 1; /\* Start timer0 \*/

}

/\* Timer0 interrupt service routine (ISR) \*/

void Timer0\_ISR() interrupt 1

{

PWM\_Out\_Pin = !PWM\_Out\_Pin;

if(PWM\_Out\_Pin)

{

TH0 = (ON\_Period >> 8);

TL0 = ON\_Period;

}

else

{

TH0 = (OFF\_Period >> 8);

TL0 = OFF\_Period;

}

}

void display(int tik)

{

unsigned char a[20];

unsigned int i = 0;

int key = rotation(tik);

if(key == 0)

{

lcd\_cmd(0xc0);

lcd\_data('a');

}

if(key==1)

{

lcd\_cmd(0xc0);

lcd\_data('c');

}

sprintf(a,"DutyCycle=%0.1f",Speed);

lcd\_init();

while( a[i] != '\0')

{

lcd\_data(a[i]);

i++;

delay(3);

}

}

void Set\_DutyCycle\_To(float duty\_cycle,int pin)

{

float period = 65535 - PWM\_Period;

ON\_Period = ((period/100.0) \* duty\_cycle);

OFF\_Period = (period - ON\_Period);

ON\_Period = 65535 - ON\_Period;

OFF\_Period = 65535 - OFF\_Period;

display(pin);

}

void Motor\_Init()

{

Speed = 0;

M1\_Pin1 = 1;

M1\_Pin2 = 0;

Set\_DutyCycle\_To(Speed,M1\_Pin1);

}

void main()

{

int count=1;

unsigned char a[20];

unsigned int i = 0;

Speed\_Inc =0;

Speed\_Dec =0;

Change\_Dir =0;

CKT\_ON = 0;

EA = 1; /\* Enable global \_interrupt \*/

ET0 = 1; /\* Enable timer0 \_interrupt \*/

Timer\_init();

Motor\_Init();

while(CKT\_ON==1)

{

while(count==1){

sprintf(a,"PWM Generation");

lcd\_init();

lcd\_cmd(0x01);

while( a[i] != '\0')

{

lcd\_data(a[i]);

i++;

delay(30);

}

count--;

}

/\* Increment Duty cycle i.e. speed by 10% for Speed\_Inc Switch \*/

if(Speed\_Inc == 1)

{

if(Speed < 100)

Speed += 10;

Set\_DutyCycle\_To(Speed,M1\_Pin1);

while(Speed\_Inc == 1);

delay(200);

}

/\* Decrement Duty cycle i.e. speed by 10% for Speed\_Dec Switch \*/

if(Speed\_Dec == 1)

{

if(Speed > 0)

Speed -= 10;

Set\_DutyCycle\_To(Speed,M1\_Pin1);

while(Speed\_Dec == 1);

delay(200);

}

/\* Change rotation direction for Change\_Dir Switch \*/

if(Change\_Dir == 1)

{

M1\_Pin1 = !M1\_Pin1;

M1\_Pin2 = !M1\_Pin2;

while(Change\_Dir == 1);

display(M1\_Pin1);

delay(200);

}

}

while(CKT\_ON == 0){

sprintf(a,"!!!");

lcd\_init();

lcd\_cmd(0x01);

while( a[i] != '\0')

{

lcd\_data(a[i]);

i++;

delay(30);

}

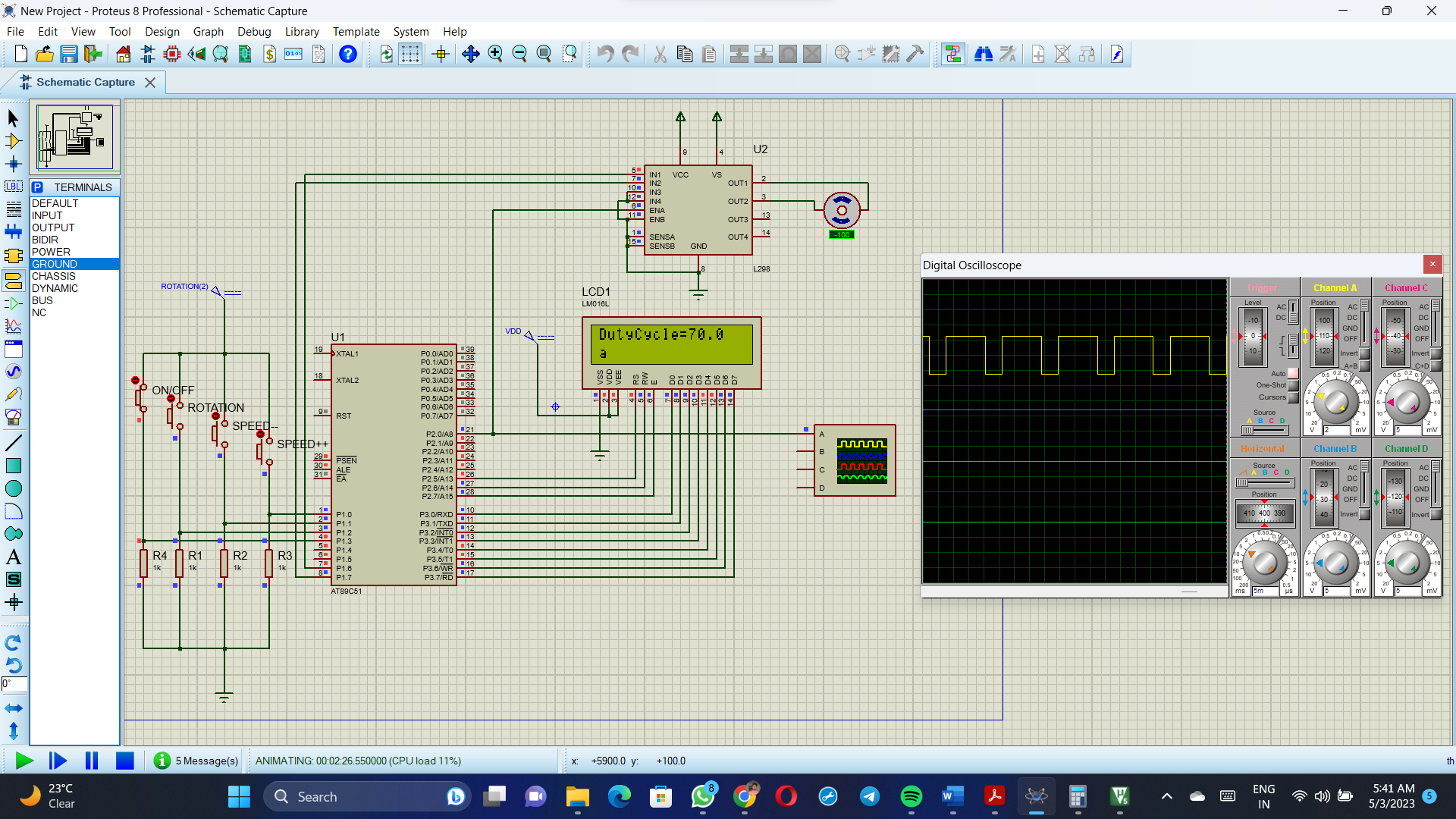
count++;

M1\_Pin1 =0;

}

}

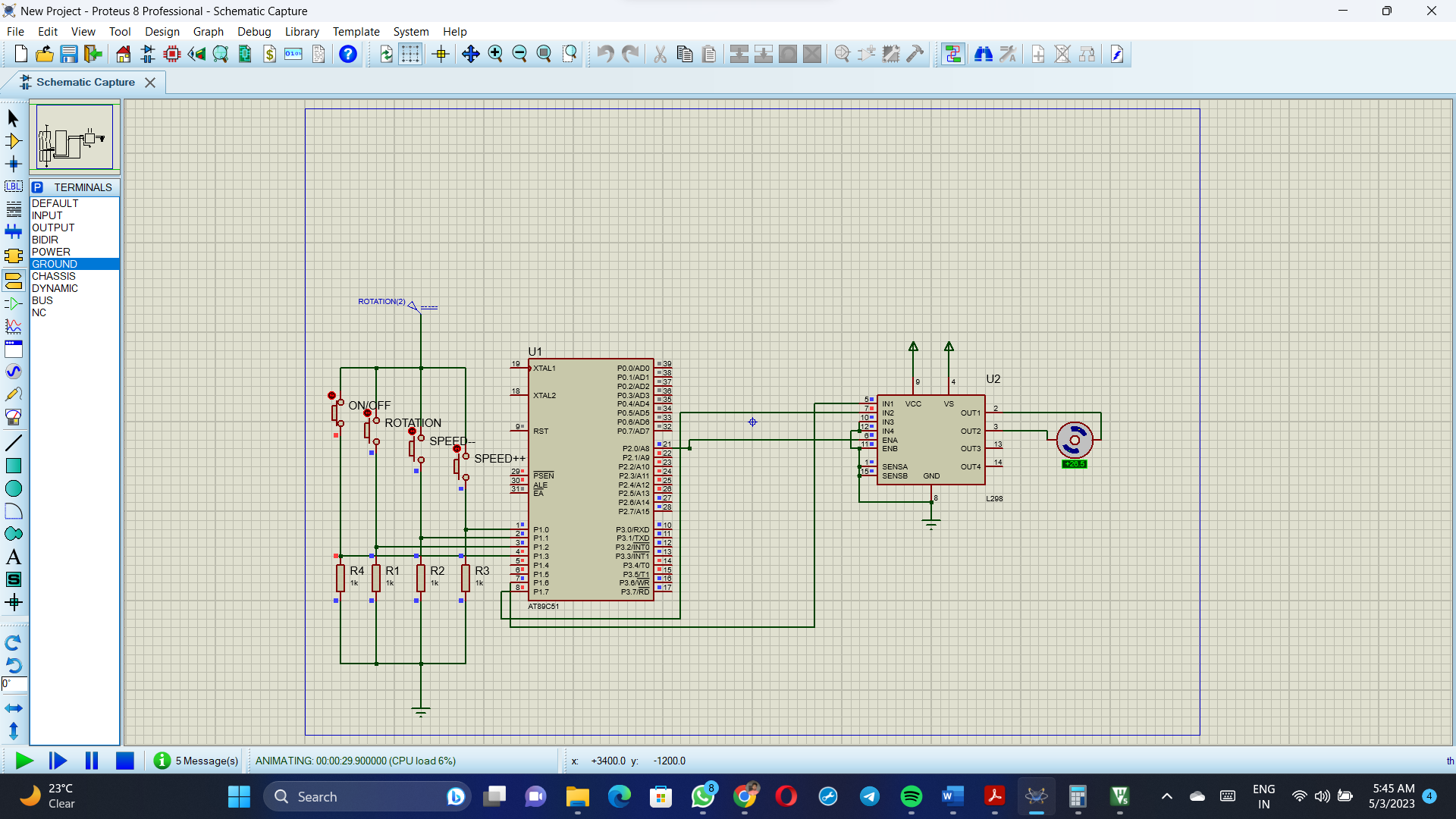
**4.3 Final Proteus simulation**



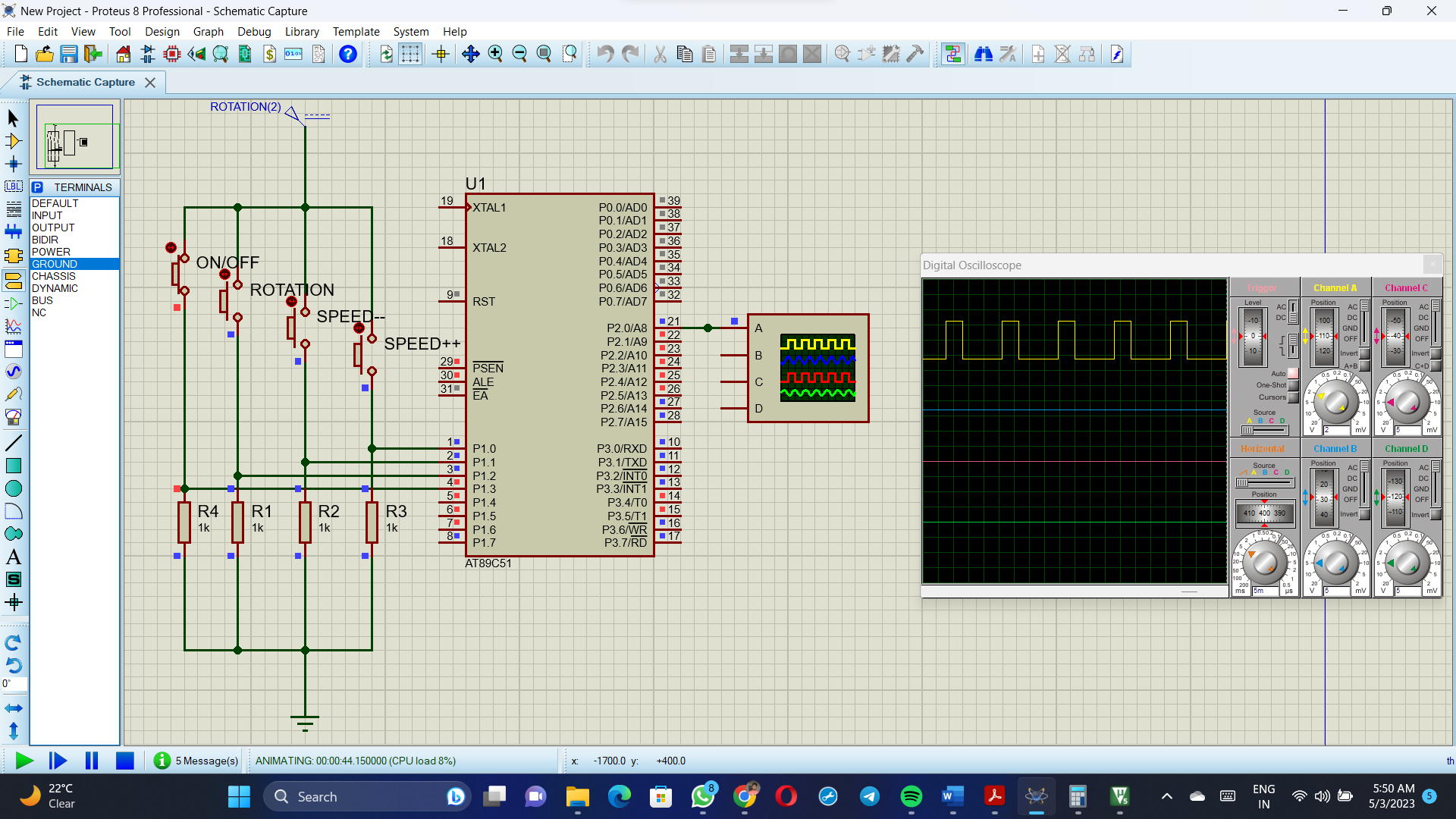
**Chapter 5**

**Result and Testing**

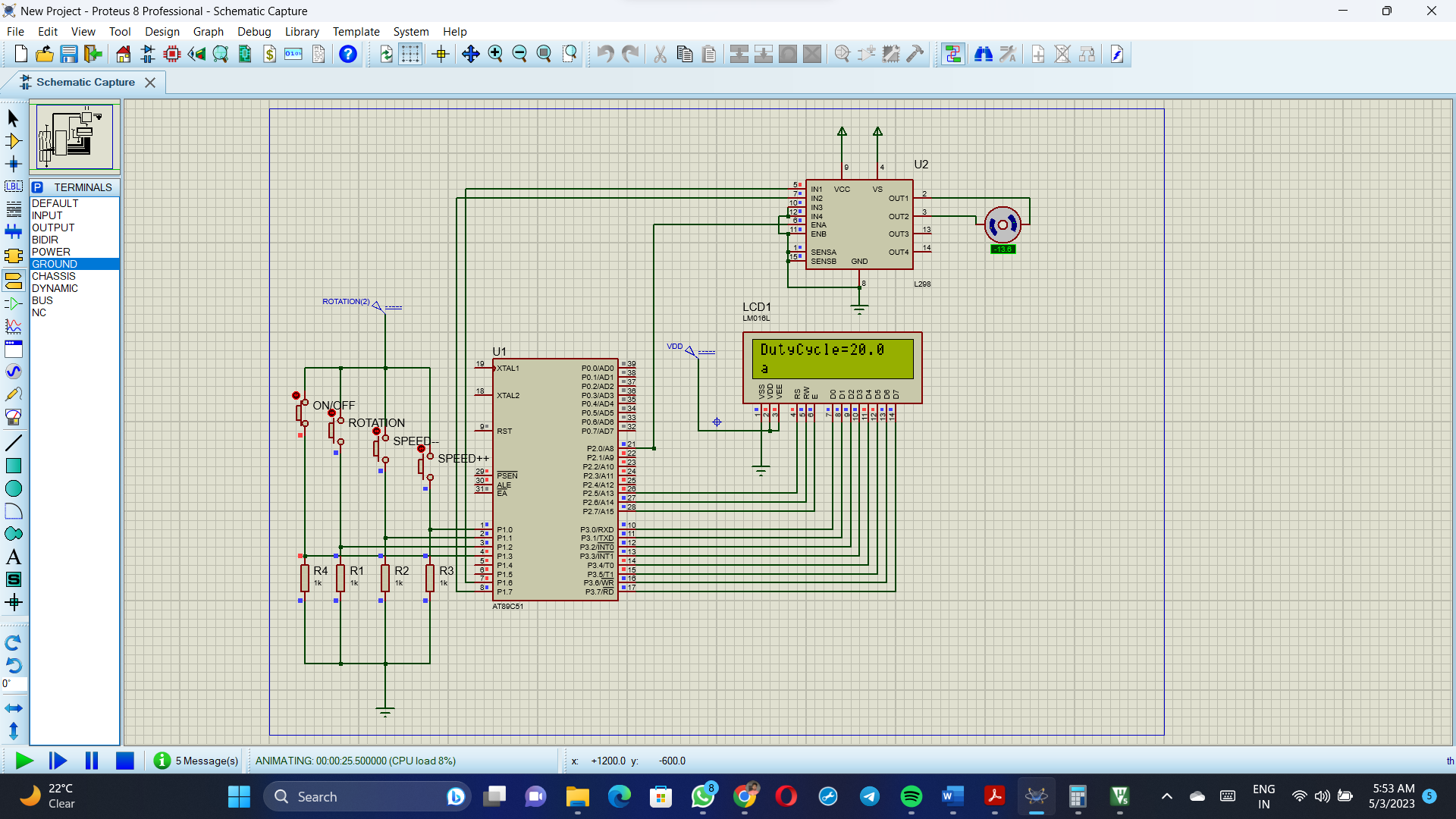
**5.1 Objective 1**



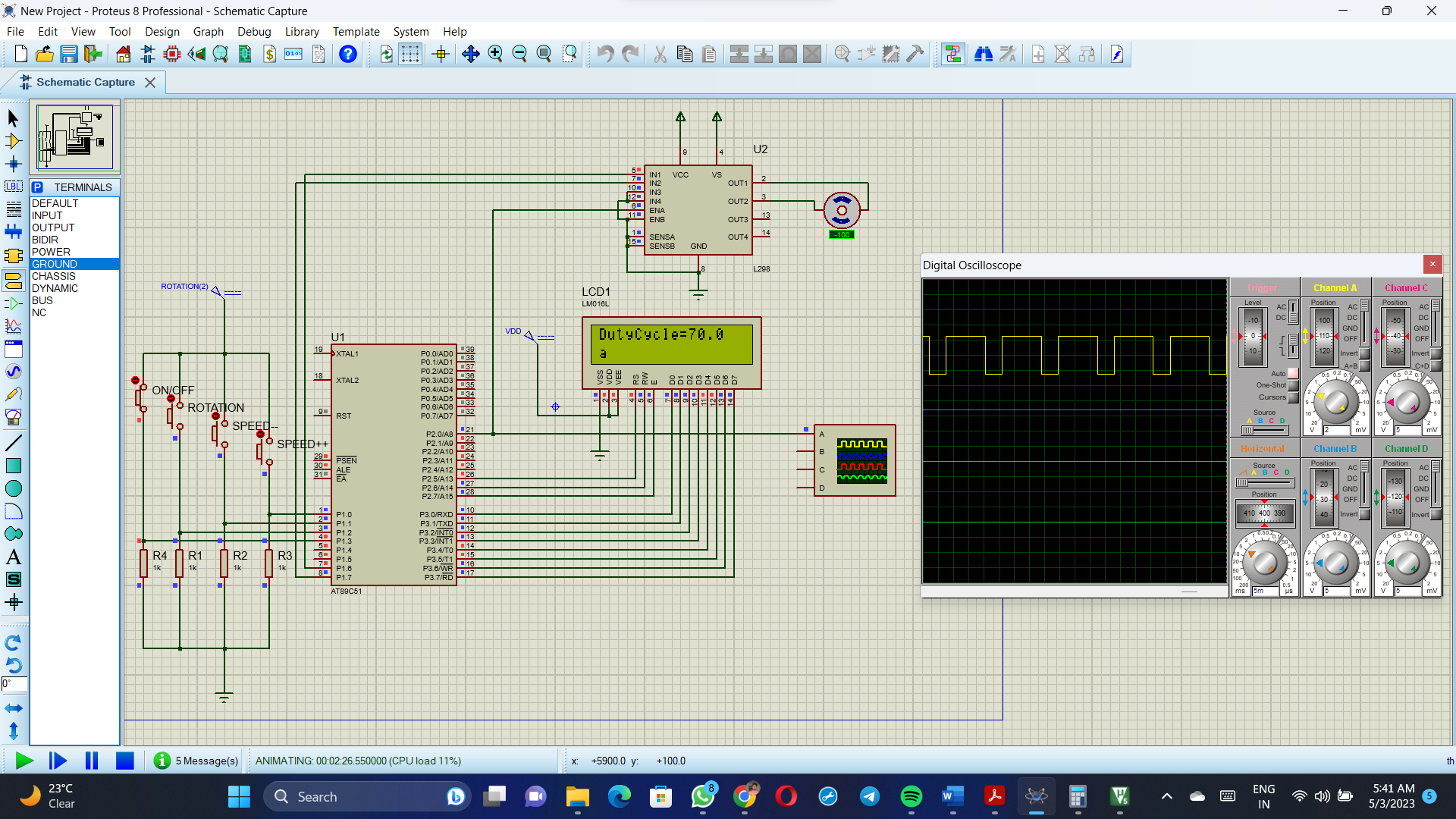
**5.2 Objective 2**



**5.3 Objective 3**



**5.4 System output**



**Chapter 6**

**Conclusion & Future Scope**

The mini project "Speed Control of DC motor using PWM" successfully demonstrated the use of pulse-width modulation technique and microcontroller-based control of DC motors. The project aimed to control the speed and direction of the DC motor using an 8051 microcontroller, an L298N motor driver module, a 12V DC motor, four push buttons, and an LCD interface. The project successfully achieved the objective by generating PWM signals based on the user input from the push buttons, adjusting the duty cycle to control the speed of the motor, and displaying the duty cycle on the LCD interface. Overall, the project serves as a valuable tool for learning and experimentation in the field of motor control and automation.

The mini project "Speed Control of DC motor using PWM" can be further enhanced and expanded in several ways. Here are some future scope possibilities:

1. Adding more push buttons: The project can be extended by adding more push buttons for additional functionalities, such as changing the acceleration rate or implementing different speed profiles.
2. Implementing feedback control: The project can be improved by incorporating a feedback system that measures the actual speed of the motor and adjusts the PWM signal accordingly to maintain a constant speed.
3. Integrating wireless communication: The project can be made more advanced by integrating wireless communication modules, such as Bluetooth or Wi-Fi, to remotely control the motor speed and direction.
4. Scaling up for industrial applications: The project can be scaled up for industrial applications by using higher voltage and current-rated motors and motor drivers, and implementing safety features like emergency stop and motor overload protection.

Overall, the mini project "Speed Control of DC motor using PWM" has immense potential for further development and expansion in various domains, including robotics, automation, and industrial control.