Tab 1

Autonomous Car Project Report

Student Name: Naman Gaur
Enrollment No: 220160205004

Autonomous Car Project Report submitted in partial fulfilment of the requirements for the Degree of BTech in Electrical(Electric Vehicle) Engineering on April 21st 2025.

Report Track: Project Report

Report Advisor: Assistant Prof. Aayush

GD Goenka University, Haryana

Student Declaration

I hereby declare that the work presented in the report entitled ”Autonomous Car Project” submitted by me for the partial fulfilment of the requirements for the degree of B.Tech. in Electrical Engineering(Electric Vehicle) at GD Goenka University, Haryana, is an authentic record of my work carried out under the guidance of Assistant Prof. Aayush . Due acknowledgements have been given in the report for all material used. This work has not been submitted elsewhere for the reward of any other degree.

Naman Gaur **Date:** April 21st, 2025

Certificate

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Assistant Prof. Aayush **Date:** April 21st, 2025

Abstract

This project presents the design and implementation of an autonomous differential-drive vehicle based on the Arduino Mega 2560 microcontroller, featuring three HC-SR04 ultrasonic sensors for environment sensing and an L298N dual H-bridge driver for motor control. The vehicle employs two DC motors arranged in a differential-drive configuration, allowing precise forward, reverse, and pivot maneuvers. Using timed laps of 10 seconds each and ultrasonic-triggered pivot turns at detected walls or corners, the platform autonomously navigates a rectangular track, counts completed laps, and halts after ten laps. This modular framework combines low-cost hardware with straightforward firmware logic, making it suitable for educational robotics, rapid prototyping, and further extensions such as closed-loop speed control or wireless telemetry.

Keywords: Differential Drive Vehicle, Ultrasonic Sensor, L298N Motor Driver, Arduino Mega 2560, ultrasonic-triggered pivot turns at detected walls or corners, autonomous navigation of a rectangular track, timed laps.

Acknowledgement

I would firstly like to thank my mentor for the project, Aayush Sir, for his immeasurable guidance and help with the building of the autonomous car. I would also like to thank my college, GD Goenka University, for providing this opportunity as well as materials for the project. I would also like to thank my mom and dad who provided great emotional support and stood as a pillar for me while duration of this project. I would also like to thank my sister for her help with purchase of the materials as well as great mentorship of the project. Finally I would like to thank my friend Abdulhadi for his help and advice regarding the construction of this project.

Contents

Student Declaration

Abstract

Acknowledgement

Table of Contents

Chapter 1: Autonomous Car Project Description

Chapter 2: Embedded Lab Experiments Concept Description

Chapter 3: Embedded Lab Experiments Implementation

Chapter 4: Integration

Chapter 1: Autonomous Car Project Description

1. Introduction

The project's objective is to build a low-cost, educational autonomous vehicle that can autonomously traverse a rectangular arena, detect its walls, and maintain lap timing without human intervention. By leveraging widely available components and straightforward firmware, the platform serves both as a teaching tool and a rapid-prototyping base for robotics experimentation.

2. Hardware Components

2.1 Arduino Mega 2560

The Arduino Mega 2560 features an ATmega2560 microcontroller, 54 digital I/O pins (15 PWM), 16 analog inputs, four hardware serial ports, and operates at 16 MHz, making it ideal for handling multiple sensors and outputs simultaneously.

2.2 Ultrasonic Sensors (HC-SR04)

Three HC-SR04 modules are used for obstacle detection on the front, left, and right sides. Each sensor emits an ultrasonic pulse and measures echo return time to estimate distance up to ~4 m with ~3 mm resolution.

2.3 Motor Driver (L298N)

The L298N IC contains two full H-bridge drivers, allowing independent bidirectional control of two DC motors up to 2 A per channel with a motor-supply range of 4.5–36 V and TTL-level inputs (4.5–7 V).

2.4 DC Motors

Two brushed DC motors form a differential-drive system. Speed and direction are independently controlled to allow straight motion, in-place pivot turns, and swerves.

2.5 Power Supply

A 12 V battery powers the motors through the L298N with its Vcc and GND terminals. Logic power (5 V) is supplied directly from the Arduino with 5V and GND terminals of both connected successfully. A 9V battery powers the Arduino for its operations.

3. Software & Control Logic

3.1 Differential-Drive Control

The firmware uses `setMotorSpeeds(left, right)` to apply signed PWM values (–255 to 255) for forward/reverse motion. Straight segments apply equal speeds; pivot turns swap one motor's polarity for in-place rotation.

3.2 Ultrasonic-Triggered Pivot Turns

In each loop, the code reads front, left, and right distances. If the front sensor reading drops below a threshold (e.g., 20 cm), it chooses a pivot direction based on which side sensor reads a larger distance, executing a swerving turn to re-align with the corridor.

3.3 Non-Blocking Lap Timing

Using `millis()`, the firmware tracks elapsed time to increment a lap count every 20 000 ms without halting execution. After ten laps, the motors halt to a stop.

4. Operational Flow

1. **Startup Delay**: A single-use startup `delay()` is added via a `firstRun` flag to allow operator readiness.
2. **Idle State**: The `delay()` function is run for 60 seconds to safely implement the connection of the motor driver to the battery.
3. **Navigation Loop**: Continuously read sensors, drive motors, and check lap timing.
4. **Lap Completion**: On every 20 s interval, increment `lapCount`, update display, and if `lapCount == 10`, call `stopMotors()` and return to Idle State.

5. Future Extensions

- **Closed-Loop Speed Control** via optical or magnetic encoders for precise RPM regulation.

- **Wireless Telemetry** using Bluetooth or Wi-Fi modules to log performance data.
- **Enhanced Path Planning** with additional sensors (IR, camera) for complex obstacle environments.
- **Multi-Lap Track Variations** with dynamic lap targets and user-settable intervals.

An image of the autonomous car in action within a well defined arena.

Chapter 2: Embedded Lab Experiments Concept Description

Experiment 1: LED Fading

1. **Description**

In this experiment we aim to fade an LED automatically using an Arduino Uno. This is done by attaching PWM capable pins 10 and 11 to the LED terminals and utilising the for loop to complete the fading of the LED. A delay of 10ms is also added within the loop to show the fading of the LED properly.

2. **Components Required**

- Arduino Uno
- LED
- Jumper wires

3. **Hardware Connections**

- Digital Pin 10 to positive terminal of the LED
- Digital Pin 11 to negative terminal of the LED

Experiment 2: Distance Measurement with Ultrasonic Sensor

1. **Description**

In this experiment we aim to measure the distance of an object using an Ultrasonic sensor. This is done by attaching PWM capable pins 10 and 11 of an Arduino Uno with the Trig and Echo pins of the Ultrasonic sensor respectively. We then connect the Vcc terminal of the sensor with the 5V Arduino pin and the GND pin of the sensor to the GND pin of the Arduino. We then calculate the distance from the object to the sensor by multiplying the time it took for the signal to return with the speed of the sound and dividing by 2, since the pulse travelled twice the distance. Next we print the distance using the Serial Monitor within our Arduino IDE.

2. **Components Required**

- Arduino Uno
- Ultrasonic Sensor
- A random object to be used for measurement
- Jumper Wires

3. **Hardware Connections**

- Digital pin 10 to Trig pin
- Digital pin 11 to Echo pin
- 5V pin to Vcc pin
- GND pin of Arduino to GND pin of sensor

Experiment 3: Ultrasonic Sensor with LED

1. **Description**

In this experiment we aim to glow an LED if the distance of the object is less than or equal to 8 cm. Our setup remains the same as the previous experiment. The code and hardware connections of the ultrasonic sensor also remain the same. Digital pins 8 and 9 are connected to the LED terminals. Here, we utilize for if else statements for the glowing of the LED. If the distance is less than or equal to 8 cm, the LED glows, else it remains the same.

2. **Components Required**

- Arduino Uno
- Ultrasonic sensor
- LED bulb
- A movable object
- Jumper Wires

3. **Hardware Connections**

- Digital pin 10 to Trig pin
- Digital pin 11 to Echo pin
- 5V pin to Vcc pin
- GND pin of Arduino to GND pin of sensor

- Digital pin 8 to positive terminal of LED
- Digital pin 9 to negative terminal of LED

Experiment 4: Display of text using LCD screen

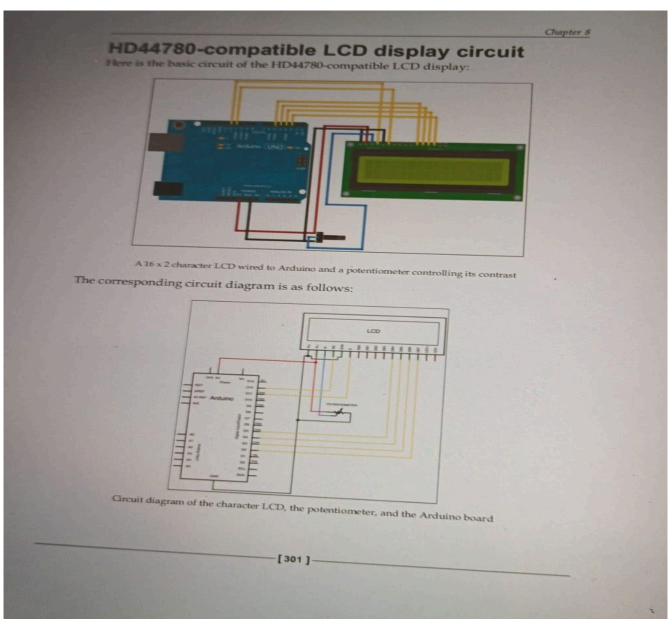
1. Description

The aim of this experiment is to display the message, “I am Arduino” upon the LCD screen. This is done by connecting the LCD to the Arduino and the Potentiometer as shown in figure. Here, we utilize the LiquidCrystal.h library and its functions. We first initialize the LCD and then write our message as a string to be printed later for a duration of 5 seconds using an if loop. We then print the time of operation below it using the `millis()` function.

2. Components Required

- Arduino Uno
- 16*2 LCD screen
- A Potentiometer
- Jumper wires

3. Hardware Connections


Chapter 8

HD44780-compatible LCD display circuit

Here is the basic circuit of the HD44780-compatible LCD display:

A 16 x 2 character LCD wired to Arduino and a potentiometer controlling its contrast

The corresponding circuit diagram is as follows:

Circuit diagram of the character LCD, the potentiometer, and the Arduino board

[301]

A schematic diagram displaying the connections for the LCD Screen

Experiment 5: LCD screen with Ultrasonic Sensor and LED

1. Description

The aim for this experiment is to display the distance measured by the ultrasonic sensor to the LCD. If the distance is less than or equal to 8 cm, the LED glows, else it remains the same. The setup for this project is the same as that of the previous 3 experiments. The distance will first be measured by the ultrasonic sensor, then it will be printed by the LCD Screen using the LiquidCrystal.h library and the condition for the LED bulb will be printed using if-else statements.

2. Components Required

- Arduino Uno
- A 16*2 LCD Screen
- An Ultrasonic sensor
- A Potentiometer
- A movable object
- Jumper Wires

3. Hardware Connections

- Digital pin 9 to Trig pin of sensor
- Digital pin 10 to Echo pin of sensor
- 5V pin to Vcc pin
- GND pin of Arduino to GND pin of sensor
- Digital pin 6 to positive terminal of LED
- Digital pin 7 to negative terminal of LED
- *The connections for the LCD Screen will remain the same as previous experiments.*

Experiment 6: Servo motor control with constant speed without using Servo.h library

1. **Description**

    The aim of this experiment is to control the rotation of a servo motor at a constant speed without the help of a library. We do this by connecting the PWM pin 9 of the Arduino Uno to the SIG pin of the servomotor and the 5V and GND pin with Vcc and GND of the servomotor respectively. We then add a delay for each step and use it in a for loop to rotate the servo motor at a constant speed. A delay of 15ms is added within the for loop to smoothen the rotation.

2. **Components Required**

    - Arduino Uno
    - A Servo motor
    - Jumper Wires

3. **Hardware Connections**

    - Digital pin 9 to SIG pin of servo motor
    - 5V pin to Vcc of servo motor
    - GND pin to GND pin of servo motor

Experiment 7: Servo motor control with variable speed without using Servo.h library

1. **Description**

    The aim of this experiment is to control the rotation of a servo motor at a variable speed as provided by a potentiometer without the help of a library. We do this by connecting the PWM pin 10 of the Arduino Uno to the SIG pin of the servomotor and the 5V and GND pin with Vcc and GND of the servomotor respectively. The SIG pin of the potentiometer is connected to analog pin A0 while the Vcc and GND are connected to 5V and GND of Arduino respectively.The value read from the potentiometer is mapped as a delay for each step which will then be used within the for loop for the rotation of the servo. A delay of 15ms is added within the for loop to smoothen the rotation.

2. **Components Required**

    - Arduino Uno
    - A Servo motor

- A Potentiometer
- Jumper Wires

Experiment 8: DC Motor Control

1. Description

This experiment aims to rotate a DC motor in clockwise direction for 3 seconds and anticlockwise direction for another 3 seconds. This is done by connecting the digital pins 10 and 11 of the Arduino Uno to the input pins of the DC motors. Then a code is written for the same using the `digitalWrite()` and the `delay()` functions.

2. Components Required

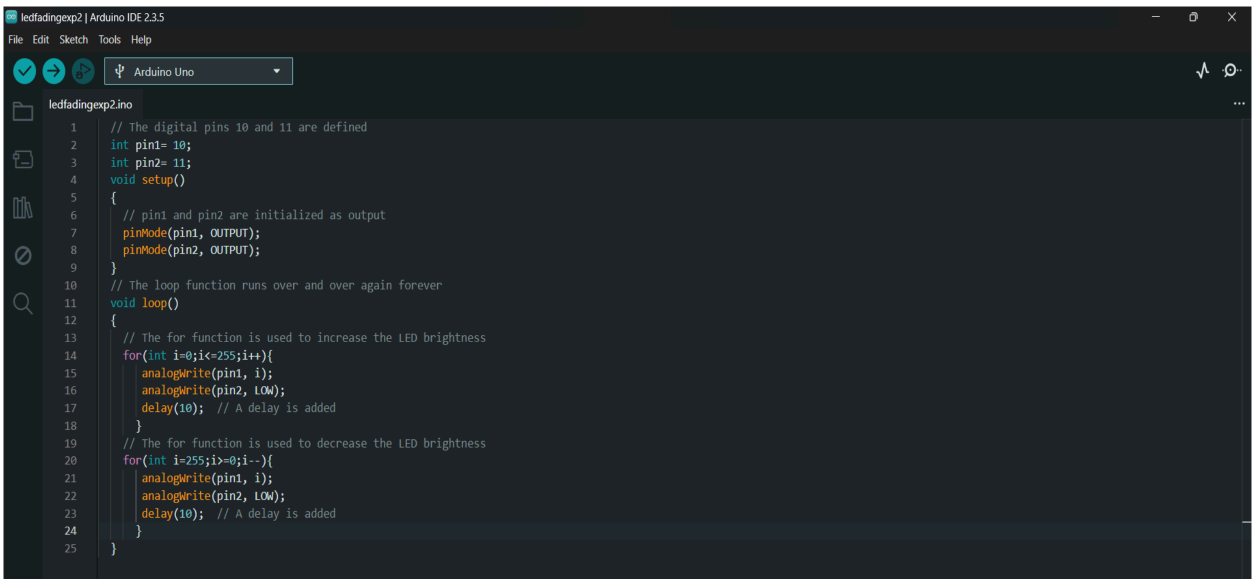
- Arduino Uno
- A DC Motor
- Jumper Wires

3. Hardware Connections

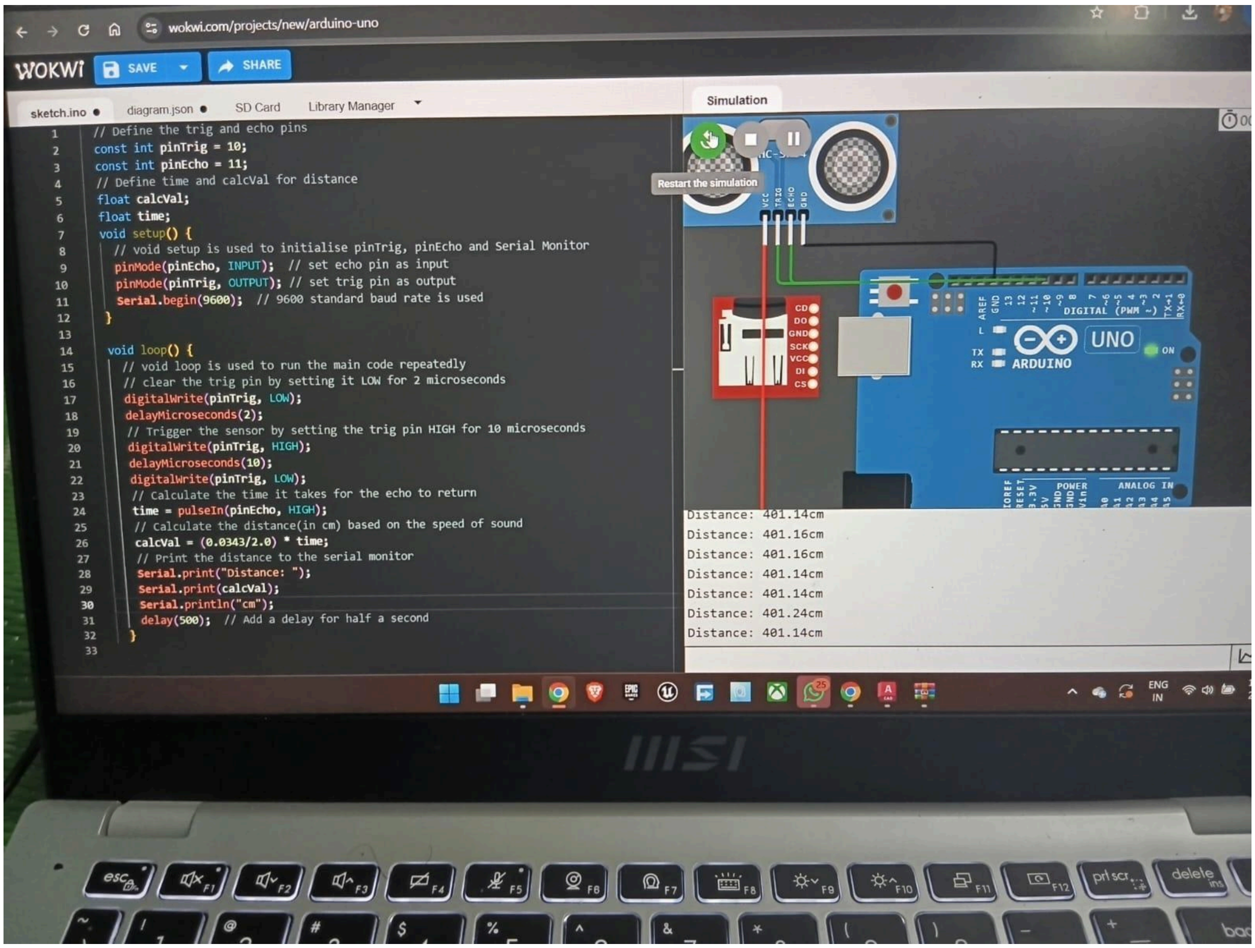
- Digital pin 10 to positive terminal of DC Motor
- Digital pin 11 to negative terminal of DC Motor

Chapter 3: Embedded Lab Experiments Implementation

Experiment 1: LED Fading

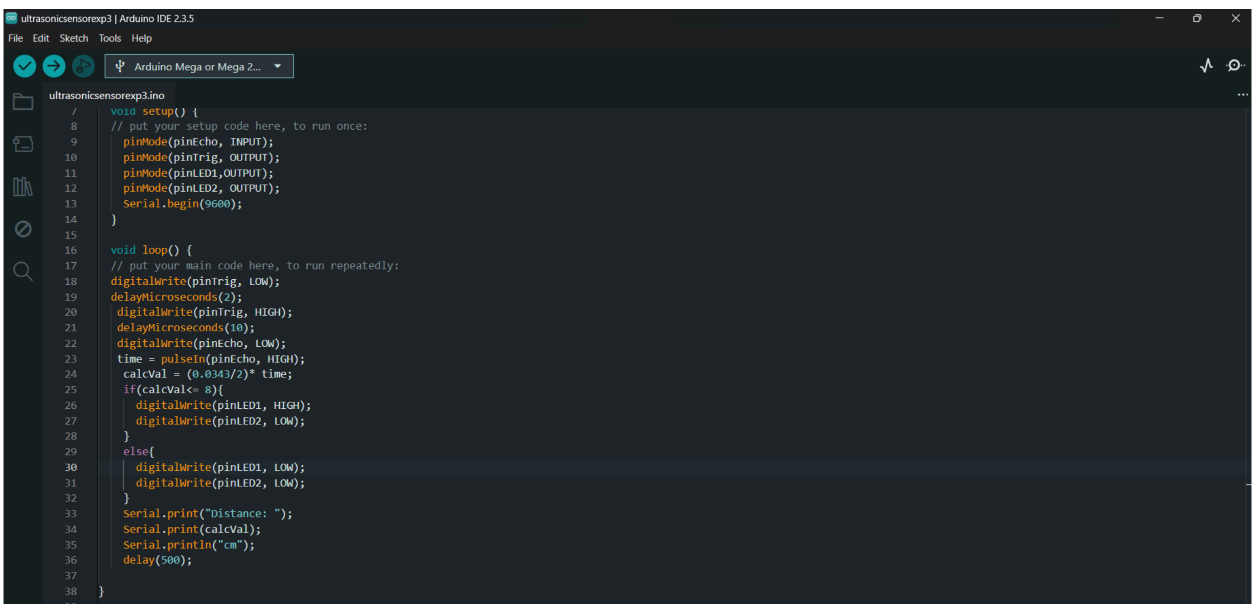


Experiment 2: Distance Measurement with Ultrasonic Sensor



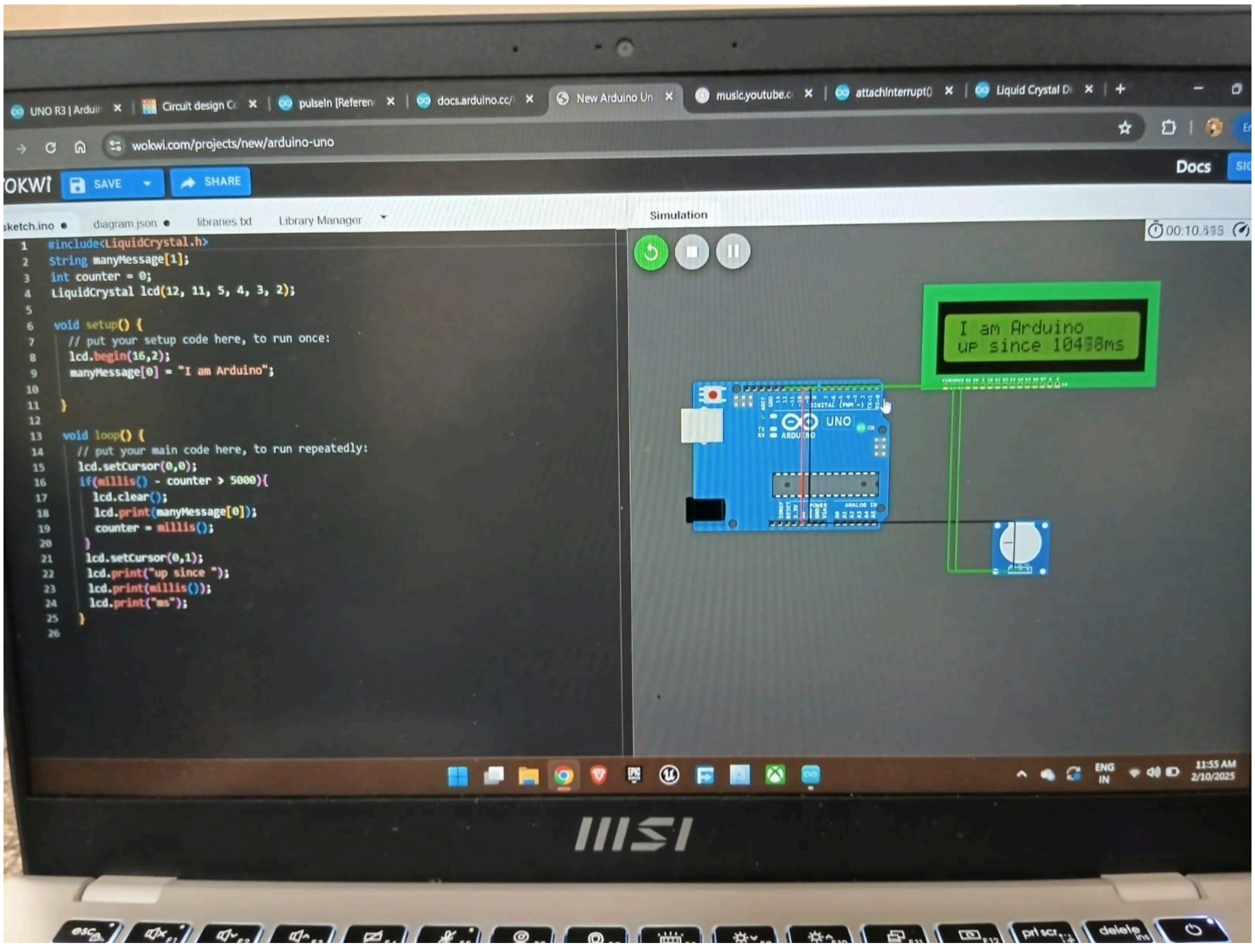
Experiment 3: Ultrasonic Sensor with LED


```
const int pinTrig= 10;
const int pinEcho= 11;
const int pinLED1 = 8;
const int pinLED2 = 9;
float calcVal;
float time;
void setup() {
// put your setup code here, to run once:
  pinMode(pinEcho, INPUT);
  pinMode(pinTrig, OUTPUT);
  pinMode(pinLED1,OUTPUT);
  pinMode(pinLED2, OUTPUT);
  Serial.begin(9600);
}

void loop() {
// put your main code here, to run repeatedly:
digitalWrite(pinTrig, LOW);
delayMicroseconds(2);
 digitalWrite(pinTrig, HIGH);
 delayMicroseconds(10);
 digitalWrite(pinEcho, LOW);
 time = pulseIn(pinEcho, HIGH);
  calcVal = (0.0343/2)* time;
  if(calcVal<= 8){
    digitalWrite(pinLED1, HIGH);
    digitalWrite(pinLED2, LOW);
  }
  else{
    digitalWrite(pinLED1, LOW);
    digitalWrite(pinLED2, LOW);
  }
  Serial.print("Distance: ");
```

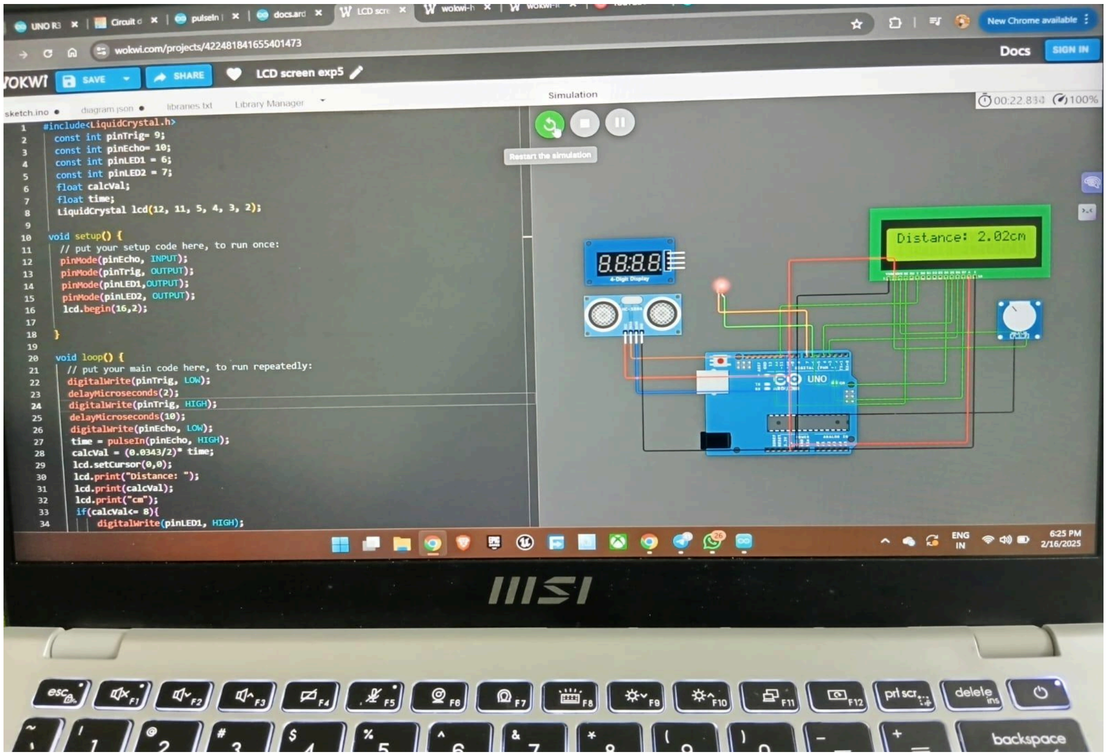


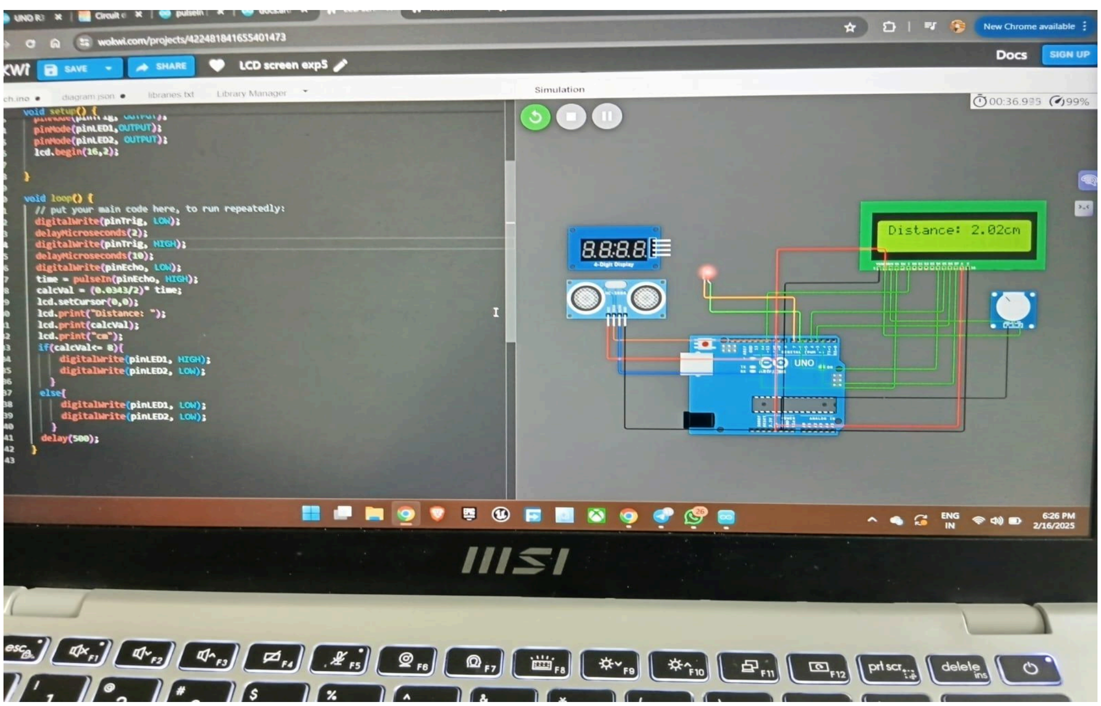

```
void setup() {
// put your setup code here, to run once:
  pinMode(pinEcho, INPUT);
  pinMode(pinTrig, OUTPUT);
  pinMode(pinLED1,OUTPUT);
  pinMode(pinLED2, OUTPUT);
  Serial.begin(9600);
}

void loop() {
// put your main code here, to run repeatedly:
digitalWrite(pinTrig, LOW);
delayMicroseconds(2);
 digitalWrite(pinTrig, HIGH);
 delayMicroseconds(10);
 digitalWrite(pinEcho, LOW);
 time = pulseIn(pinEcho, HIGH);
  calcVal = (0.0343/2)* time;
  if(calcVal<= 8){
    digitalWrite(pinLED1, HIGH);
    digitalWrite(pinLED2, LOW);
  }
  else{
    digitalWrite(pinLED1, LOW);
    digitalWrite(pinLED2, LOW);
  }
  Serial.print("Distance: ");
  Serial.print(calcVal);
  Serial.println("cm");
  delay(500);

}
```

Experiment 4: Display of text using LCD screen

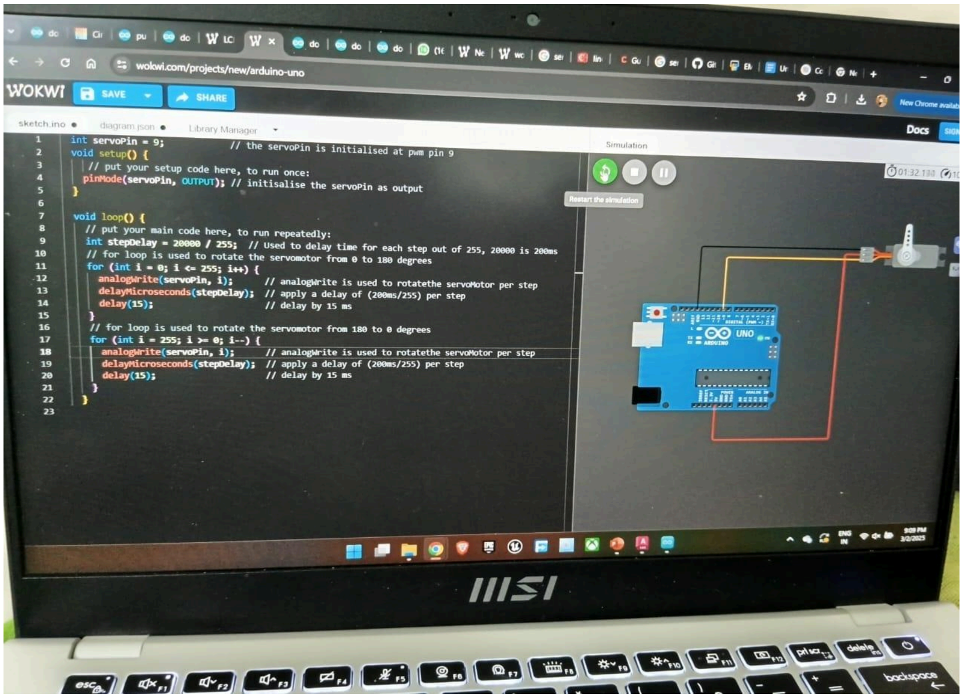


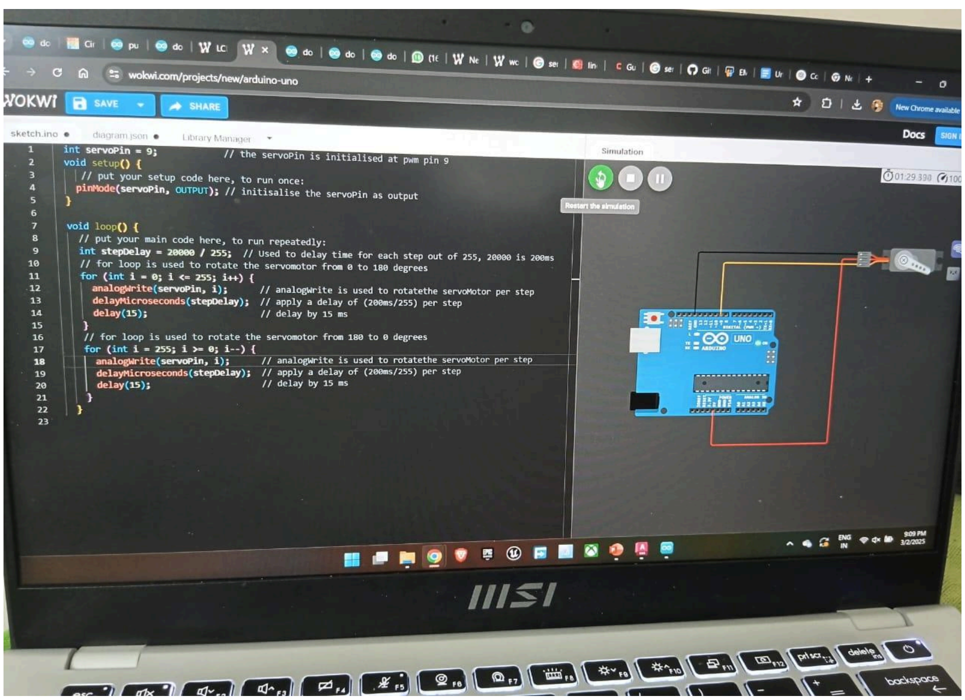
Experiment 5: LCD screen with Ultrasonic Sensor and LED



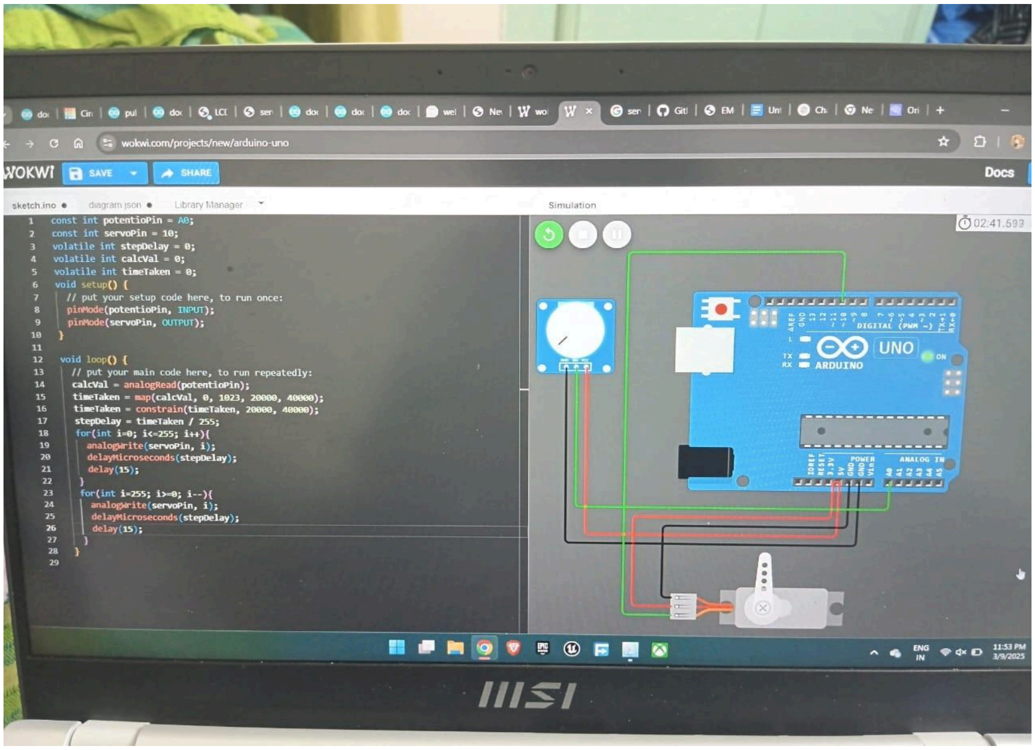


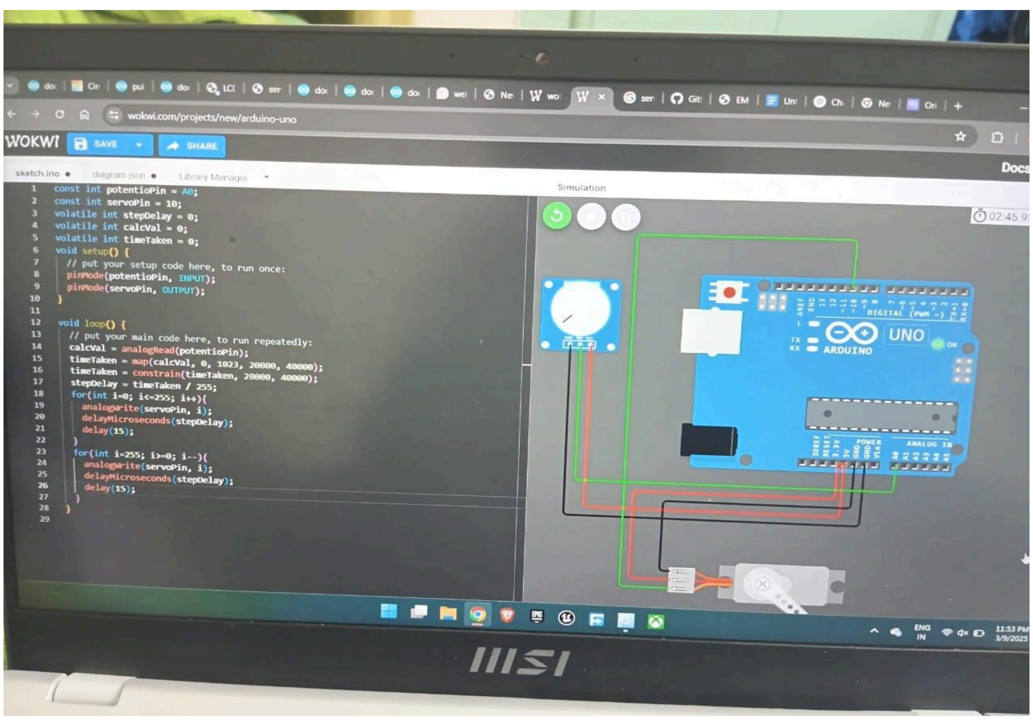
Experiment 6: Servo motor control with constant speed without using Servo.h library





Experiment 7: Servo motor control with variable speed without using Servo.h library





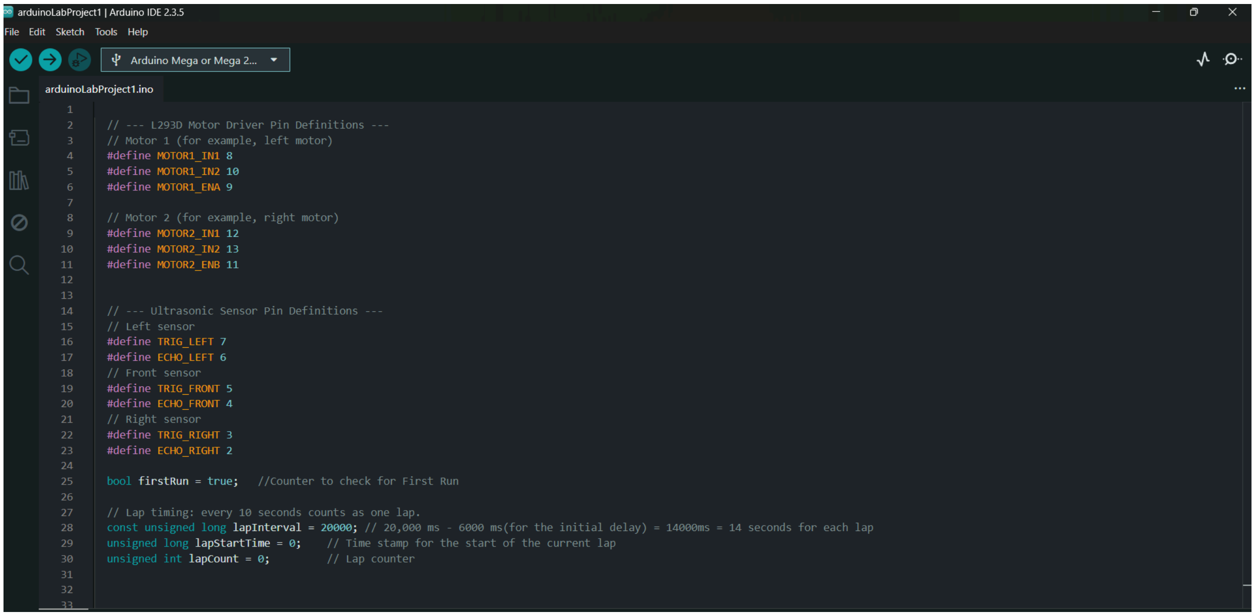
Experiment 8: DC Motor Control

WOKWI SAVE SHARE

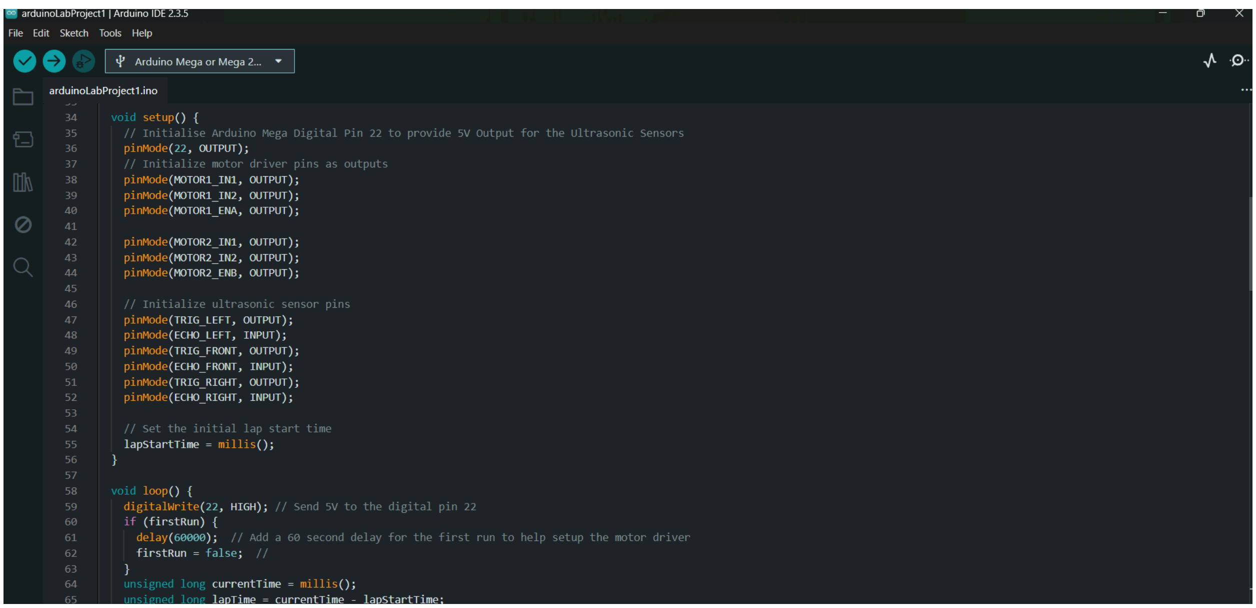
sketch.ino ● diagram.json ● Library Manager

```
// Define the both input pins of the DC Motor
#define MOTOR_IN1 10
#define MOTOR_IN2 11
  void setup() {
  // put your setup code here, to run once:
  // initialise both the input pins of the DC motor
    pinMode(MOTOR_IN1, OUTPUT);
    pinMode(MOTOR_IN2, OUTPUT);
  }

  void loop() {
  // put your main code here, to run repeatedly:
  // rotate the motor in clockwise direction for 3 seconds
    digitalWrite(MOTOR1_IN1, HIGH);
    digitalWrite(MOTOR1_IN2, LOW);
    delay(3000);  // A delay of 3 seconds is added
  // rotate the motor in anticlockwise direction for 3 seconds
    digitalWrite(MOTOR_IN1, LOW);
    digitalWrite(MOTOR_IN2, HIGH);
    delay(3000);  // A delay of 3 seconds is added
  }
```

Chapter 4: Integration




```cpp

// --- L293D Motor Driver Pin Definitions ---
// Motor 1 (for example, left motor)
#define MOTOR1_IN1 8
#define MOTOR1_IN2 10
#define MOTOR1_ENA 9

// Motor 2 (for example, right motor)
#define MOTOR2_IN1 12
#define MOTOR2_IN2 13
#define MOTOR2_ENB 11


// --- Ultrasonic Sensor Pin Definitions ---
// Left sensor
#define TRIG_LEFT 7
#define ECHO_LEFT 6
// Front sensor
#define TRIG_FRONT 5
#define ECHO_FRONT 4
// Right sensor
#define TRIG_RIGHT 3
#define ECHO_RIGHT 2

bool firstRun = true;   //Counter to check for First Run

// Lap timing: every 10 seconds counts as one lap.
const unsigned long lapInterval = 20000; // 20,000 ms - 6000 ms(for the initial delay) = 14000ms = 14 seconds for each lap
unsigned long lapStartTime = 0;    // Time stamp for the start of the current lap
unsigned int lapCount = 0;         // Lap counter
```




```cpp
void setup() {
  // Initialise Arduino Mega Digital Pin 22 to provide 5V Output for the Ultrasonic Sensors
  pinMode(22, OUTPUT);
  // Initialize motor driver pins as outputs
  pinMode(MOTOR1_IN1, OUTPUT);
  pinMode(MOTOR1_IN2, OUTPUT);
  pinMode(MOTOR1_ENA, OUTPUT);

  pinMode(MOTOR2_IN1, OUTPUT);
  pinMode(MOTOR2_IN2, OUTPUT);
  pinMode(MOTOR2_ENB, OUTPUT);

  // Initialize ultrasonic sensor pins
  pinMode(TRIG_LEFT, OUTPUT);
  pinMode(ECHO_LEFT, INPUT);
  pinMode(TRIG_FRONT, OUTPUT);
  pinMode(ECHO_FRONT, INPUT);
  pinMode(TRIG_RIGHT, OUTPUT);
  pinMode(ECHO_RIGHT, INPUT);

  // Set the initial lap start time
  lapStartTime = millis();
}

void loop() {
  digitalWrite(22, HIGH); // Send 5V to the digital pin 22
  if (firstRun) {
    delay(60000);  // Add a 60 second delay for the first run to help setup the motor driver
    firstRun = false;  //
  }
  unsigned long currentTime = millis();
  unsigned long lapTime = currentTime - lapStartTime;
```

arduinoLabProject1 | Arduino IDE 2.3.5

File Edit Sketch Tools Help

Arduino Mega or Mega 2...

arduinoLabProject1.ino

```
    // --- Lap Timing ---
    if (lapTime >= lapInterval) {
      lapCount++;

      // After 10 laps, stop the vehicle
      if (lapCount >= 10) {
        stopMotors();
        while (true) {
        // Halt further execution
        }
      }
      // Reset lap timer for next lap
      lapStartTime = currentTime;
    }

    // --- Read Ultrasonic Sensors ---
    int distanceFront = readUltrasonic(TRIG_FRONT, ECHO_FRONT);
    int distanceLeft  = readUltrasonic(TRIG_LEFT, ECHO_LEFT);
    int distanceRight = readUltrasonic(TRIG_RIGHT, ECHO_RIGHT);

    // --- Decision Making Using Sensor Readings ---
    // If a wall (or corner) is detected by the front sensor, perform a pivot turn.
    // The turning direction is chosen by comparing left and right sensor readings.
    if (distanceFront <= 20.5) {
      // Pivot turn using differential motor speeds
      if (distanceLeft > distanceRight) {
        // Turn (swerve) to the left: left motor rotates backward, right motor rotates forward.
        setMotorSpeeds(0, 1);
      }
      if (distanceLeft > 50 && distanceRight > 50) {
        // Stops both the motors rotation and halts the movement of the vehicle
        setMotorSpeeds(0, 0);
```

arduinoLabProject1 | Arduino IDE 2.3.5

File Edit Sketch Tools Help

Arduino Mega or Mega 2...

arduinoLabProject1.ino

```
      }
    } else {
       // Otherwise, continue moving straight ahead at the set speed.
       setMotorSpeeds(1, 1);
       if(distanceLeft < 3.7) {
         // Turn (swerve) to the right: left motor rotates forward, right motor rotates backward.
         setMotorSpeeds(1, 0);
       }
       if(distanceRight < 3.7) {
         // Turn (swerve) to the left: left motor rotates backward, right motor rotates forward.
         setMotorSpeeds(0, 1);
       }
      }
  }

  // --- Function to Read Distance from an Ultrasonic Sensor ---
  // Returns the distance in centimeters.
  int readUltrasonic(int trigPin, int echoPin) {
    long duration, distance;

    // Ensure the trigger is low
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);

    // Send a 10-microsecond pulse to trigger the sensor
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);

    // Read the echo time
    duration = pulseIn(echoPin, HIGH);
```

arduinoLabProject1 | Arduino IDE 2.3.5

File Edit Sketch Tools Help

Arduino Mega or Mega 2...

arduinoLabProject1.ino

```cpp
  // Calculate the distance in centimeters:
  // (duration in µs) * (speed of sound 0.034 cm/µs) / 2
  distance = duration * 0.0343 / 2;

  return distance;
}

// --- Motor Control Function ---
// The below function helps set the rotational direction of both motors 1 and 2 depending upon the variables speed1 and speed2
void setMotorSpeeds(int speed1, int speed2) {
  // Control Motor 1
  if (speed1 == 0) {
    // Motor 1(say the left motor) rotates backward at speed1 = 0
    digitalWrite(MOTOR1_IN1, LOW);
    digitalWrite(MOTOR1_IN2, HIGH);
    analogWrite(MOTOR1_ENA, 255);
  } else {
    // Motor 1(say the left motor) rotates forward at speed1 = 1
      digitalWrite(MOTOR1_IN1, HIGH);
      digitalWrite(MOTOR1_IN2, LOW);
      analogWrite(MOTOR1_ENA, 255);
    }

  // Control Motor 2
  if (speed2 == 0) {
    // Motor 2(say the right motor) rotates backward at speed2 = 0
    digitalWrite(MOTOR2_IN1, LOW);
    digitalWrite(MOTOR2_IN2, HIGH);
    analogWrite(MOTOR2_ENB, 255);
  } else {
      // Motor 2(say the right motor) rotates forward at speed2 = 1
      digitalWrite(MOTOR2_IN1, HIGH);
```

arduinoLabProject1 | Arduino IDE 2.3.5

File Edit Sketch Tools Help

Arduino Mega or Mega 2...

arduinoLabProject1.ino

```cpp
    analogWrite(MOTOR1_ENA, 255);
  } else {
    // Motor 1(say the left motor) rotates forward at speed1 = 1
      digitalWrite(MOTOR1_IN1, HIGH);
      digitalWrite(MOTOR1_IN2, LOW);
      analogWrite(MOTOR1_ENA, 255);
    }

  // Control Motor 2
  if (speed2 == 0) {
    // Motor 2(say the right motor) rotates backward at speed2 = 0
    digitalWrite(MOTOR2_IN1, LOW);
    digitalWrite(MOTOR2_IN2, HIGH);
    analogWrite(MOTOR2_ENB, 255);
  } else {
      // Motor 2(say the right motor) rotates forward at speed2 = 1
      digitalWrite(MOTOR2_IN1, HIGH);
      digitalWrite(MOTOR2_IN2, LOW);
      analogWrite(MOTOR2_ENB, 255);
    }
}


// --- Function to Stop Both Motors ---
void stopMotors() {
  // Both motor 1 and motor 2 stop their rotation as soon as this function is called within the loop
  digitalWrite(MOTOR1_IN1, LOW);
  digitalWrite(MOTOR1_IN2, LOW);
  analogWrite(MOTOR1_ENA, 0);
  digitalWrite(MOTOR2_IN1, LOW);
  digitalWrite(MOTOR2_IN2, LOW);
  analogWrite(MOTOR2_ENB, 0);
}
```