LINE FOLLOWING ROBOT

Group Members:

FAHAD MAHMOOD (221678)

SABEEH UR REHMAN (221672)

M USMAN (221756)

BE MECHATRONICS (Session 2022-2026)

Project Supervisor

ENGR. UMER FAROOQ

ASSISTANT PROFESSOR

SUBMISSION DATE: JUNE 11, 2024

DEPARTMENT OF MECHATRONICS ENGINEERING

FACULTY OF ENGINEERING

AIR UNIVERSITY, ISLAMABAD

CERTIFICATE OF APPROVAL

It is certified that the project work titled "***LINE FOLLOWER ROBOT***" is carried out at Air University, Islamabad. It is fully adequate in scope and in quality, as end semester project of course Micro-Controllers–Lab for the degree of BS. Mechatronics Engineering.

DECLARATION

It is declared that this is an original piece of our own work, except where otherwise acknowledged in text and references. This work has not been submitted in any other form for another degree or diploma at any University or other Institution for tertiary education and will not be submitted by us in future for obtaining any degree from this or any other University or Institution.

ABSTRACT

Robots can be fixed robots or mobile robots. Mobile Robots are robots with a mobile base which makes the robot move freely in the environment. One of the advanced mobile robots is the Line Follower Robot. It is basically a robot which follows a particular path or trajectory. The path can be a black line on the white floor (visible) or a magnetic field (invisible). This Robot can be integrated with more and more sensors to get full information of the movement, Voltage, Current, Speed, Obstacles and store them in SD Card. Its applications start from basic domestic uses to industrial uses, etc. The use of line following robotic vehicle is transport the materials from one place to another place in the industries. This robot movement completely depends on the track.

TABLE OF CONTENTS

CHAPTER 1- PROJECT PRELIMINARIES

1.1 PROPOSAL

Our proposed was to design a line following robot that can cover any given path in minimum possible time while being cost effective also. Microcontroller System for a line following robot that can go on binary path from Point A to Point B, upon reaching Point B which is a T junction with all black line and stop, which it uses IR sensors which detects the line and H bridge which controls the working of the wheels.

1.2 CONCEPT

Generally, the path is predefined and can be either visible like a black line on a white surface with a high contrasted color or it can be invisible like a magnetic field. Definitely, this kind of Robot should sense the line with its Infrared Ray (IR) sensors that installed under the robot. After that, the data is transmitted to Logic circuit. Hence, The Logic circuit is going to decide the proper commands and then it sends them to the driver and thus the path will be followed by the line follower robot. In this Report, we have illustrated the process of design, implementation and testing, a small line follower robot designed for the line follower robot's competition. Basic attributes of our project considered by us were:

1. Reliable construction.

2. Cost effective.

3. Collection of data of various types.

1.3 DESIGN

We have done our designing by working on the attributes of this project in detail.

Reliable construction:

1. Using button to turn ON or OFF the robot.
2. Implementing circuit on breadboard.

Cost Effective:

1. Purchasing the components from whole seller.
2. Using the minimally optimum components to complete the project.

Collection of data of various types:

1. Current required by motors.
2. Required speed.
3. Voltage required.
4. Weight of the system with batteries.
5. Lighting check for the working of IR-sensors.

1.4 WORK BREAK DOWN STRUCTURE

PHASES	WORK DISTRIBUTION
PHASE I	Simulation on Proteus, Programming
PHASE II	PCB designing and Making
PHASE III	Ordering components
PHASE IV	Implementation on breadboard and then on PCB
PHASE V	Complete Project and final testing

Table 1.1 Work distribution

1.5 ESTIMATED BUDGETS

COMPONENT	PRICE
1 chassis (2 tires, 2 motors, bolts, nuts, and base)	1200-1500
Wires (Male to female, male to male, female to female)	400
Soldering Wire	200
Sticker Sheet	120-150
Ultrasonic Sensor	250
Sil headers (male & female)	180-200
Vero board	0
Voltage Sensor	150
Current Sensor	230-250
Double sided tape	100-150
Ferric chloride	250
5 IR sensors	5x180
Adhesives	0
L298N motor driver	350-370
2 ARDUINO MEGA	2200

PCB	800
Bread board	140-200
12 battery cells	210
LCD Display	650
SD Card module	200
Servo motor	280
Ultrasonic sensor Holder	70-100
Nail Polish remover (Acetone)	80
	Total = 8,000-9,000

Table 1.2 Estimated Budgets

1.6 GANTT CHART

CONTENTS	25th APR	27rd APR	30th APR	5th MAY	13th MAY	17th MAY	20th MAY
SCHEMATIC							
EQUIPMENT PURCHASING							
BREADBOARD IMPLEMENTATION							
PCB DESIGNING ON SOFTWARE							
PCB IMPLEMENTATION							
ERROR ERADICATION AND FINAL TESTING							
REPORT WRITING							

Table 1.3 GANTT CHART

1.7 TEAM MEMBERS ROLES & DETAIL

Team Members	Roles

FAHAD MAHMOOD (ORANGE)	• **Schematic simulation and PCB designing**
	• **Report making (focusing on):**
	  ◦ PCB design details
	  ◦ Component selection
	  ◦ Manufacturing considerations

M USMAN (DARK RED)	• **Technical work**
	• **Assembling hardware**
	• **Debugging and updating code**
	• **Report making (focusing on):**
	  ◦ Integration of hardware and software
	  ◦ Testing procedures

SABEEH UR REHMAN (GREEN)	• **Buying components** • **Writing code** • **Schematic simulation on Proteus.**

Table 1.4 Team roles

1.8 Work Break down structure

Phase	Work distribution
Phase I	**Simulation on Proteus**
Phase II	**Buying Components**
Phase III	**PCB Making**
Phase IV	**Implementation on Breadboard**
Phase VI	**PCB Implementation**
Phase V	**Complete Project and final Testing**
Phase VII	**Report Making**

Table 1.5 Work breakdown

CHAPTER # 2-PROJECT CONCEPTION

2.1 INTRODUCTION

Robots are basically complete automatic machines. It starts working when sense something, decides to work according to the conditions and stop by its own due to other condition or senses. So, we can say robot are the replica of human being, as they work on phenomena as human being do. Robots are always made for the ease of human being. Now the robot may be **fixed or mobile robot**.

The **line follower robot** is one of the advance mobile robots, as we make its base moveable and it can move from one place to another. Line follower robot follows a particular path or trajectory and decides it decides its choices to interact with obstacle. Usually the path is a black line on the white floor, but in other cases it may be a magnetic field which can't be seen by naked eyes. They can be mostly use in industry and in domestic chores as they can carry the parcels or materials from one place to another place.

2.2 LITERATURE REVIEW:

2.2.1. ULTRASONIC SENSOR:

Ultrasonic transducers and ultrasonic sensors are devices that generate or sense ultrasound energy. An ultrasonic sensor sends a high pulse (signal) and then a low pulse (signal) in a continuous manner. Once these signals hit an obstacle, the signals reflect back and are received by ultrasonic sensor. The time taken by the signals to return is used to calculate the distance between the sensor and the obstacle. The closer the obstacle is to the sensor, the quicker these signals return.

Figure 2.1 ULTRASONIC SENSOR

Pin Configuration:

Pin Number	Pin Name	Description
1	Vcc	The Vcc pin powers the sensor, typically with +5V
2	Trigger	Trigger pin is an Input pin. This pin has to be kept high for 10us to initialize measurement by sending US wave.
3	Echo	Echo pin is an Output pin. This pin goes high for a period of time which will be equal to the time taken for the US wave to return back to the sensor.
4	Ground	This pin is connected to the Ground of the system.

Table 2.1 Pin Configuration

HC-SR04 Ultrasonic (US) sensor is a 4 pin module, whose pin names are Vcc, Trigger, Echo and Ground respectively. This sensor is a very popular sensor used in many applications where measuring distance or sensing objects are required. The module has two eyes like projects in the front which forms the Ultrasonic transmitter and Receiver. The sensor works with the simple high school formula that

$$Distance = Speed \times Time$$

The Ultrasonic transmitter transmits an ultrasonic wave, this wave travels in air and when it gets objected by any material it gets reflected back toward the sensor this reflected wave is observed by the Ultrasonic receiver module.

2.2.2. SERVO MOTOR:

A **servo motor** is a type of motor that can rotate with great precision. Normally this type of motor consists of a control circuit that provides feedback on the current position of the motor shaft, this feedback allows the servo motors to rotate with great precision. If you want to rotate an object at some specific angles or distance, then you use a servo motor. It is just made up of a simple motor which runs through a **servo mechanism**. If motor is powered by a DC power supply then it is called DC servo motor, and if it is AC-powered motor then it is called AC servo motor. For this tutorial, we will be discussing only about the **DC servo motor working.**

Figure 2.2 SERVO MOTOR

Servo motor can be rotated from 0 to 180 degrees, but it can go up to 210 degrees, depending on the manufacturing. This degree of rotation can be controlled by applying the **Electrical Pulse** of proper width, to its Control pin. Servo checks the pulse in every 20 milliseconds. The pulse of 1 ms (1 millisecond) width can rotate the servo to 0 degrees, 1.5ms can rotate to 90 degrees (neutral position) and 2 ms pulse can rotate it to 180 degrees.

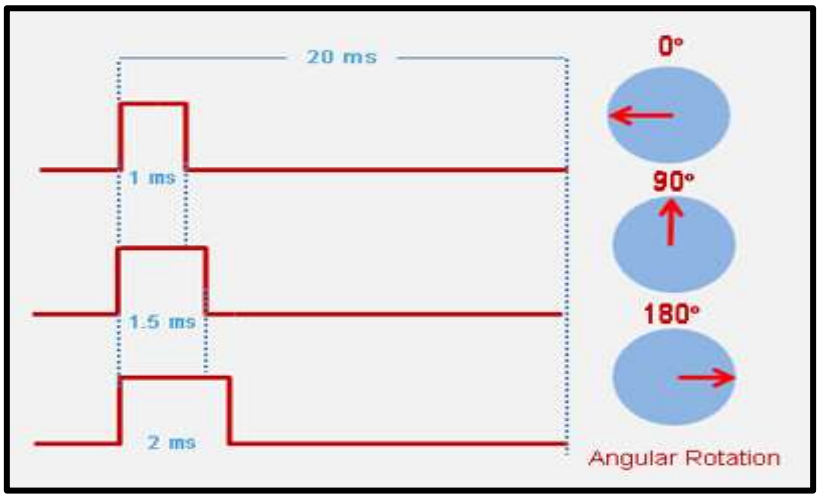


Figure 2.3 SERVO MOTOR ROTATION PULSE

Pin Configuration:

Wire Number	Wire Colour	Description
1	Brown	Ground wire connected to the ground of system
2	Red	Powers the motor typically +5V is used
3	Orange	PWM signal is given in through this wire to drive the motor

Table 2.2 Pin Configuration

ARDUINO MEGA 2560:

The **Arduino Mega 2560** is a microcontroller board based on the ATmega2560. It has 54 digital input/output pins (of which 15 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Mega 2560 board is compatible with most shields designed for the Uno and

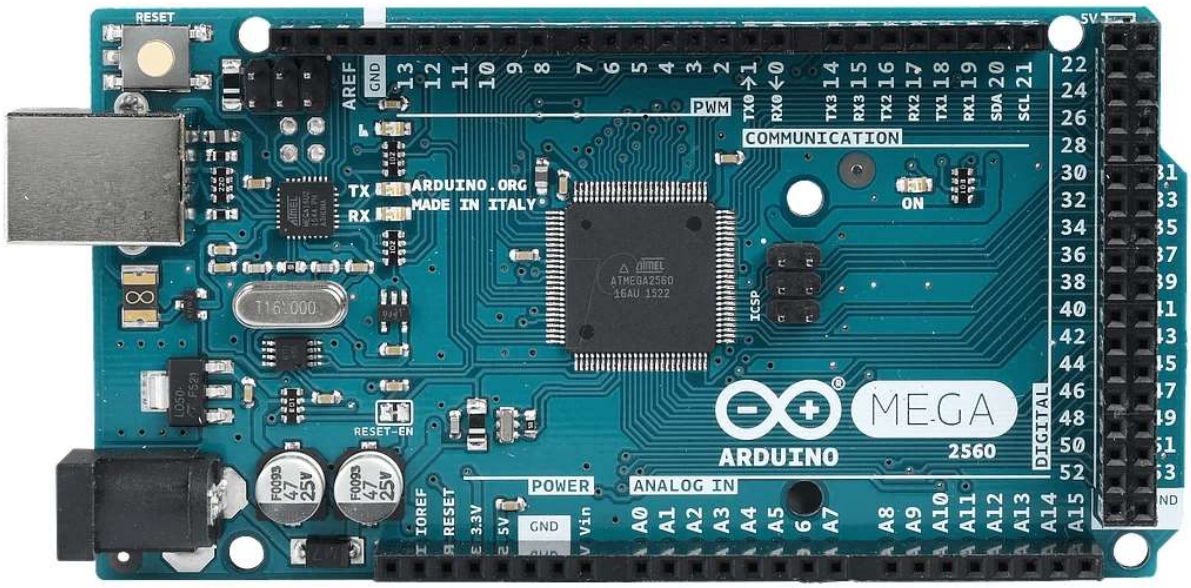


Figure 2.4 ARDUINO AtMega-2560

Pin Configuration:

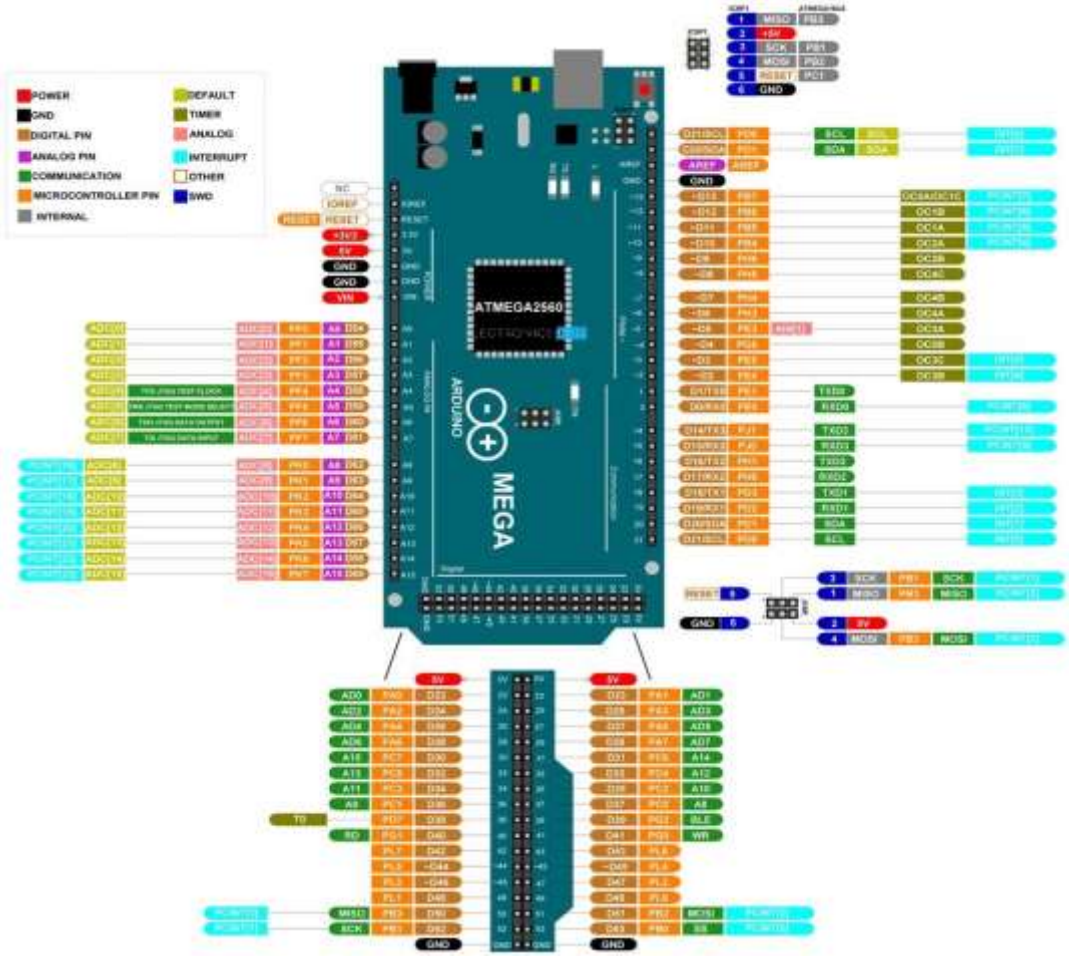


Figure 2.5 Pin Configuration

2.2.3. Robotic Chassis (2 Wheel with DC Motor):

This robotic chassis kit contains of an acrylic base with two gear motors, two compatible wheels, a ball caster, and other accessories.

Package Contains:

1. 1 x Rubber wires
2. 1 x Deceleration motors
3. 1 x Aluminum fasteners
4. 1 x Nylon all-direction wheel
5. 1 x Chassis
6. 1 x Battery box (4 x AA batteries, not included)
7. 1 x Screwdriver.

Figure 2.6 ROBOTIC CHASIS

2.2.4. ESP32

ESP32 is the name of the chip that was developed by Espressif Systems. This provides Wi-Fi (and in some models) dual-mode Bluetooth connectivity to embedded devices. While ESP32 is technically just the chip, modules and development boards that contain this chip are often also referred to as “ESP32” by the manufacturer. ESP32 can perform as a complete standalone system or as a slave device to a host MCU, reducing communication stack overhead on the main application processor. ESP32 can interface with other systems to provide Wi-Fi and Bluetooth functionality through its SPI / SDIO or I2C / UART interface.

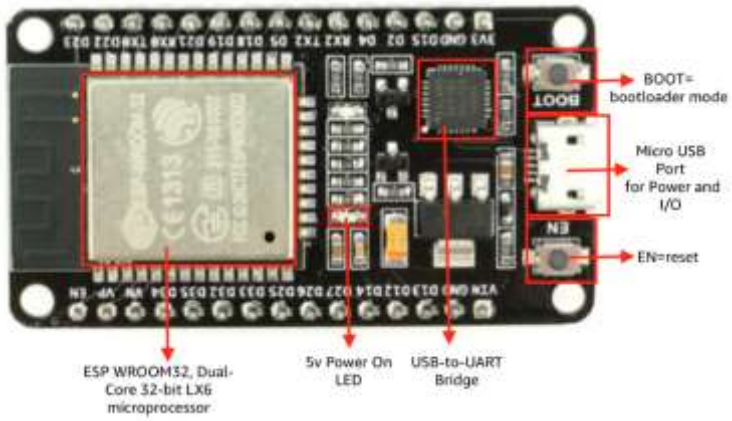


Figure 2.7 ESP 32

2.2.5. L298N Motor Driver

This **L298N Motor Driver Module** is a high power motor driver module for driving DC and Stepper Motors. This module consists of an L298 motor driver IC and a 78M05 5V regulator. **L298N Module** can control up to 4 DC motors, or 2 DC motors with directional and speed control.

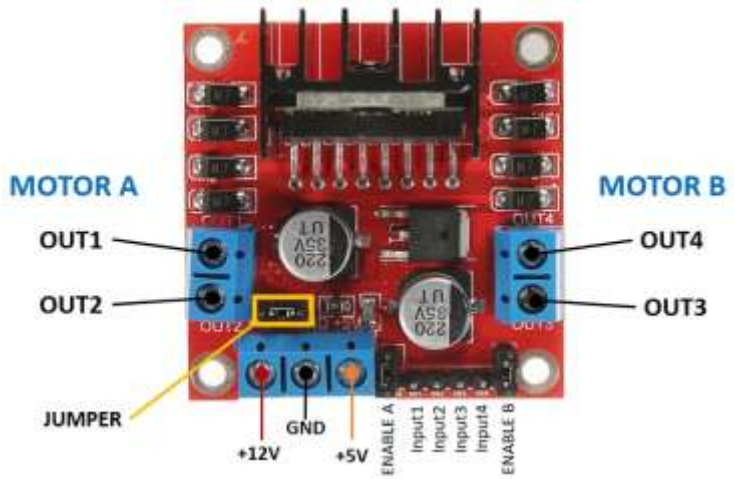


Figure 2.8 L298N

2.2.6. TT Gear Motor:

The controller i.e., L298N is basically used to control this DC motor. Its speed and direction are controlled by the l298N. D.C. motors are built to operate in the steady state in a speed range close to their no-load speed this speed is generally too high for applications like the robot we are making so that is why we merge the DC motor with gearbox to reduce its speed. The basic working principle of the DC motor is that whenever a current carrying conductor places in the magnetic field, it experiences a mechanical

force. The principle remains the same but using them both as a single component is very helpful in performing variety of functions.

Figure 2.9 TT Gear Motor

2.2.7. TCRT 5000 IR SENSOR

The TCRT5000 are reflective sensors which include an infrared emitter and phototransistor in a leaded package which blocks visible light. The package includes two mounting clips. The TCRT5000 are reflective sensors which include an infrared emitter and phototransistor in a leaded package which blocks visible light.

Figure 2.10 TCRT5000 IR Sensor

2.2.8. ROTARY INCREMENTAL ENCODER

An incremental rotary encoder is a type of electromechanical device that converts the angular motion or position of a rotary shaft into analogue or digital code that represents that motion or position. It can be used for motor speed and position feedback applications that include a servo control loop and for light- to heavy-duty industrial applications.

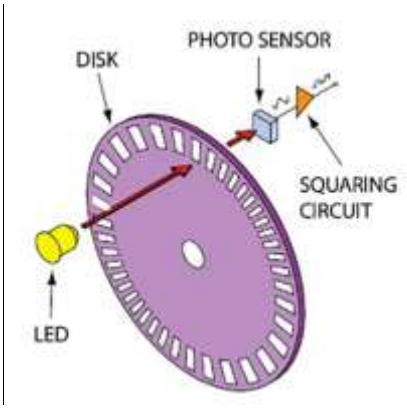


Figure 2.11 Rotary Incremental Encoder

2.2.9. SD Card Module

SD cards or Micro SD cards are widely used in various applications, such as data logging, data visualization, and many more. Micro SD Card Adapter modules make it easier for us to access these SD cards with ease. The Micro SD Card Adapter module is an easy-to-use module with an SPI interface and an on-board 3.3V voltage regulator to provide proper supply to the SD card.

Figure 2.12 SD CARD MODULE

2.2.10. 16 x 4 LCD Display

16X4 CHARACTER LCD 1604 GREEN LCD DISPLAY is a dot-matrix liquid crystal display module specially used for displaying letters, numbers, symbols, etc. Divided into 4-bit and 8-bit data transmission methods. 1604 Green Character LCD provides rich command settings: clear display; cursor return to origin; display on/o; cursor on/o; display character ashes; cursor shift; display shift, etc. It can be used in any embedded systems, industrial device, security, medical and hand-held equipment.

Figure 2.13 LCD DISPLAY

Pin Configuration:

Pin No.	Symbol	Description
1	V_{SS}	Ground
2	V_{DD}	Power supply for logic
3	V_O	Contrast Adjustment
4	RS	Data/ Instruction select signal
5	R/W	Read/Write select signal
6	E	Enable signal
7~14	DB0~DB7	Data bus line
15	A	Power supply for B/L +
16	K	Power supply for B/L -

Table 2.3 Pin Configuration

2.2.11. ACS712 CURRENT SENSOR

The **ACS712 Module** uses the famous **ACS712 IC** to **measure current** using the Hall Effect principle. The module gets its name from the IC (ACS712) used in the module, so for you final products use the IC directly instead of the module.

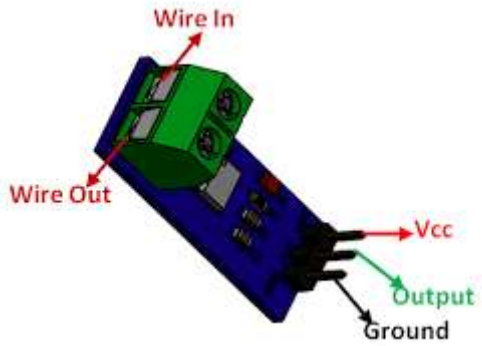


Figure 2.14 Current Sensor

2.2.12. VOLTAGE SENSOR

Voltage Sensor is a precise low-cost sensor for measuring voltage. It is based on the principle of resistive voltage divider design. It can make the red terminal connector input voltage to 5 times smaller.

Figure 2.15 Voltage Sensor

2.2.13. Potentiometer

A potentiometer is a three-terminal resistor with a sliding or rotating contact that forms an adjustable voltage divider. If only two terminals are used, one end and the wiper, it acts as a variable resistor or rheostat.

The measuring instrument called a potentiometer is essentially a voltage divider used for measuring electric potential (voltage); the component is an implementation of the same principle, hence its name.

Potentiometers are commonly used to control electrical devices such as volume controls on audio equipment. Potentiometers operated by a mechanism can be used as position transducers, for example, in a joystick. Potentiometers are rarely used to directly control significant power (more than a watt), since the power dissipated in the potentiometer would be comparable to the power in the controlled load.

Figure 2.16 Potentiometer

2.3 FLOW DIAGRAM

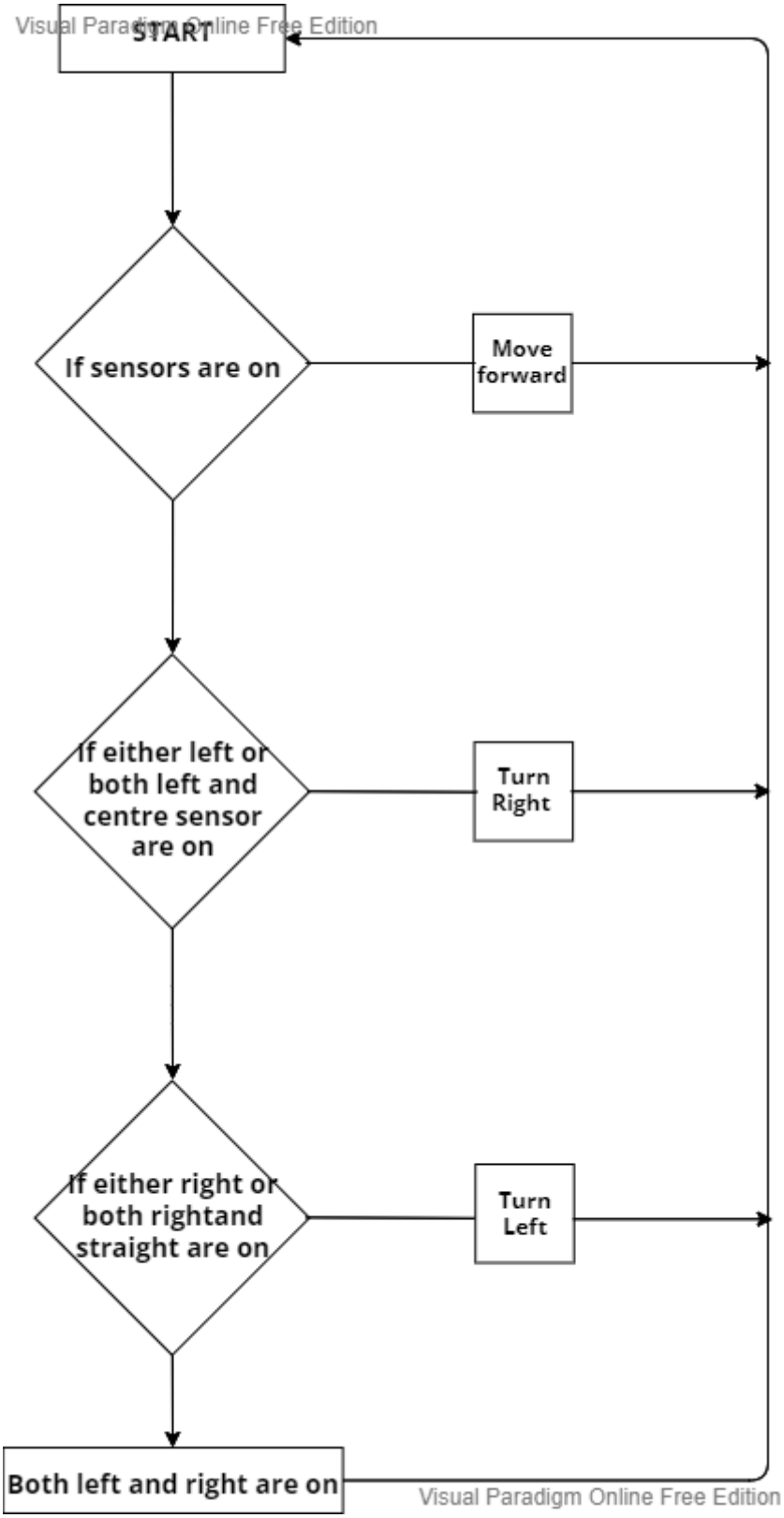


Figure 2.17 Flow Diagram

2.4 DETAILED BLOCK DIAGRAM:

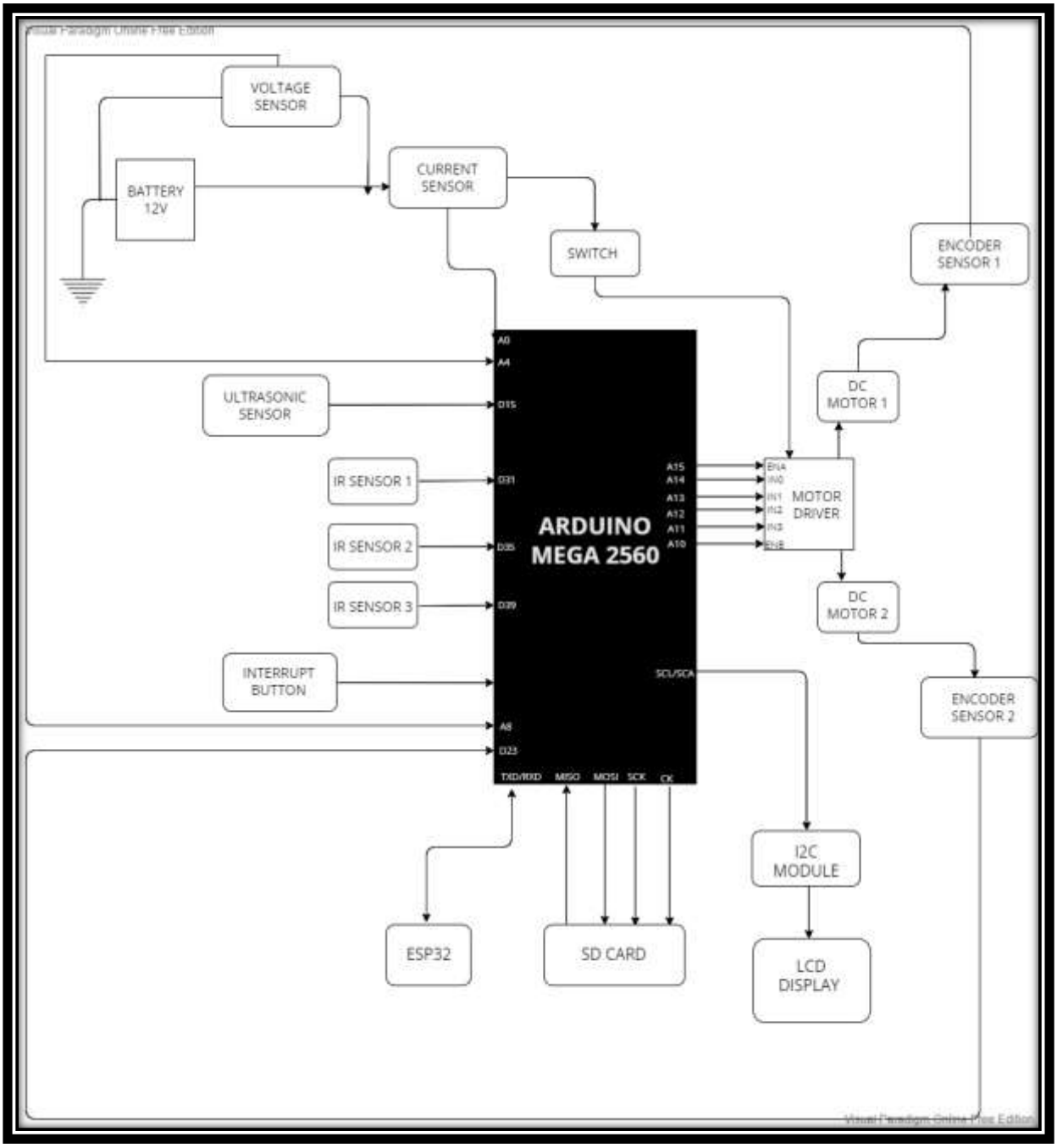


Figure 2.18 Detailed block diagram

2.5 STATE MACHINE DIAGRAM

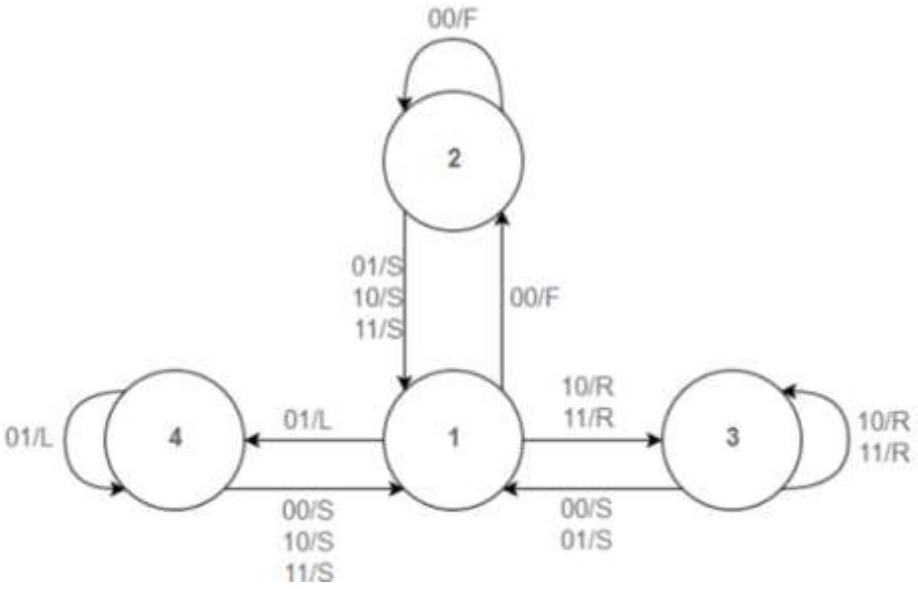


Figure 2.19 State Machine Diagram

CHAPTER # 3 PRODUCT DESIGN

3.1 SYSTEM CONSIDERATION FOR THE DESIGN

Our project was based on microcontroller use. And the sensors used in our project were based on logic implementation. For building up logic we used truth table. We like to simplify logic to a lowest cost form to save costs by elimination of components. We define lowest cost as being the lowest number of gates with the lowest number of inputs per gate.

3.2 Criteria for Component Selection:

Why DC motors?

Dc motors are most easy to control. One dc motor requires only 2 signals for its operation. If we want to change its direction just reverse the polarity of the power supply across it, we can vary speed by varying the voltage across motor. Mathematical interpretation: Rotation power is given by:

$$T = Pr / w$$

Rotational power is constant for DC motor for constant input electrical power. Thus, torque is inversely proportional speed.

Why 2 Motors?

By using 2 motors we can move our robot in any direction. This steering mechanism of the robot is called differential drive.

This means that in order to increase the power output of the motor, you can increase the voltage rating or increase the current. For example, a 12 volt DC motor can supply the same power as a six volt DC motor, but at 1/2 the current. This is important because most components are limited by the amount of current they can carry. If your robot will be extremely heavy, you may even want to look at 24 volt DC or even 90 volt DC motors. One of the trade offs for the higher voltage is safety. It is hard to shock yourself at 12 or 24 volts, but 90 volts can cause shock and possible injury. Another key property of DC motors is that the speed is controlled by changing the voltage.

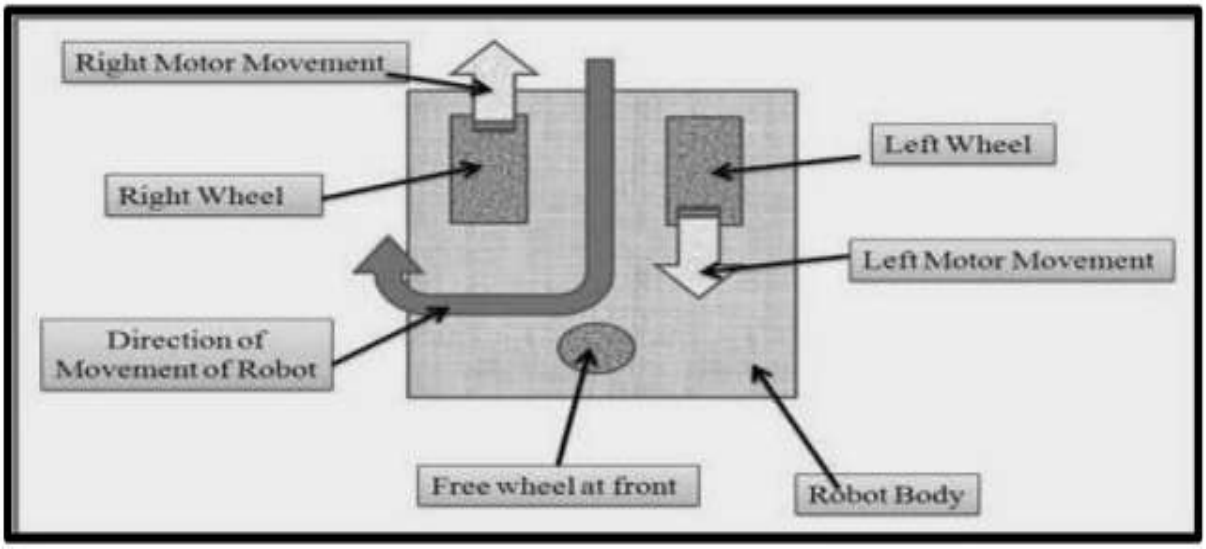


Fig 3.1: Description of various parts.

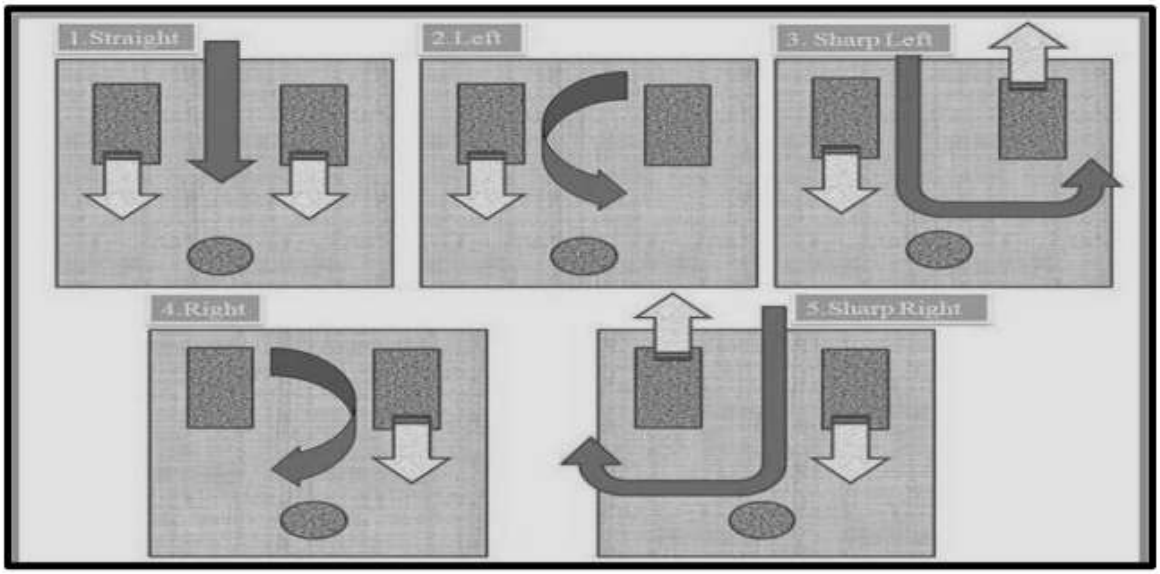


Figure 3.2 Description of various parts.

Left Motor	Right Motor	Robot Movement
Straight	Straight	Straight
Stop	Straight	Left
Reverse	Straight	Left
Straight	Stop	Right
Straight	Reverse	Right
Reverse	Reverse	Reverse

Sensors Selection

As we are asked to make the line follower robot without using any microcontroller, so the best selection of the sensor in this regard is IR sensors. In our case of arena, the sensor will detect black line as path while white area as obstacle, so we have to configure our circuit accordingly. The simplest case will be if we use 2 IR sensor and can work, even 1 IR sensor works if we have only black or white surface area. But as we are advised to used minimum 6 or 5 sensors, that will increase the complexity of the circuit and robot, but it will improve the accuracy of our robot. That is why we will use 5 sensors in our robot.

Working principle of IR sensor

Let we have a black line path and white surface area. So, the basic working principle of line follower robot is related to light. We will use behavior of light at the black and white surfaces. When light or IR rays falls on a white surface it is almost reflected. Now when light or IR rays falls on black surface, it is completely absorbed. This behaviour of light is used in building a line follower robot, and can be seen below in figure.

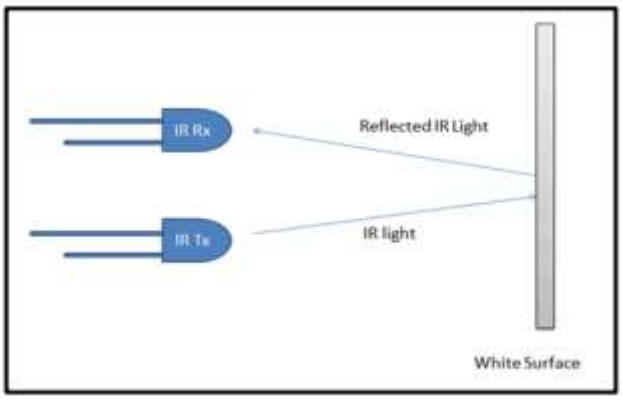


Figure 3.3 IR SENSOR WORKING

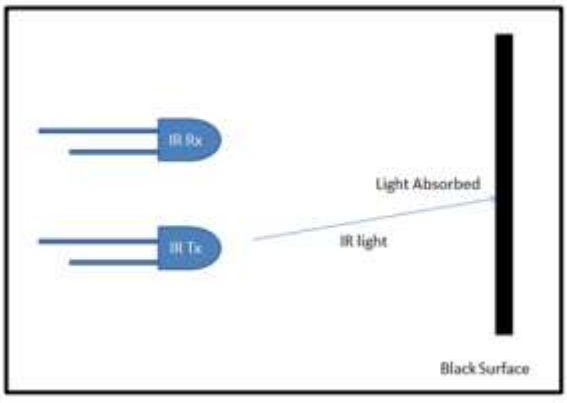


Figure 3.4 Working principle of the Sensors

Why should we use 16x4 LCD?

1. Slim profile
2. No radiation emission from the screen
3. Better under brighter conditions because of anti-glare technology
4. Lighter in weight with respect to screen size
5. Energy efficient because of lower power consumption

CHAPTER # 4 Mechanical Design

Mechanical design play very important rule in any project of engineering. Our line following robot is an engineering project that can be affected by a lame or bad model of mechanical design. We can say that the mechanical design is responsible for the sustainability, weight lifting, long lasting life of the robot. So, designing the robot in a most sufficient way so that external condition or internal condition on that material used and that design are minimum.

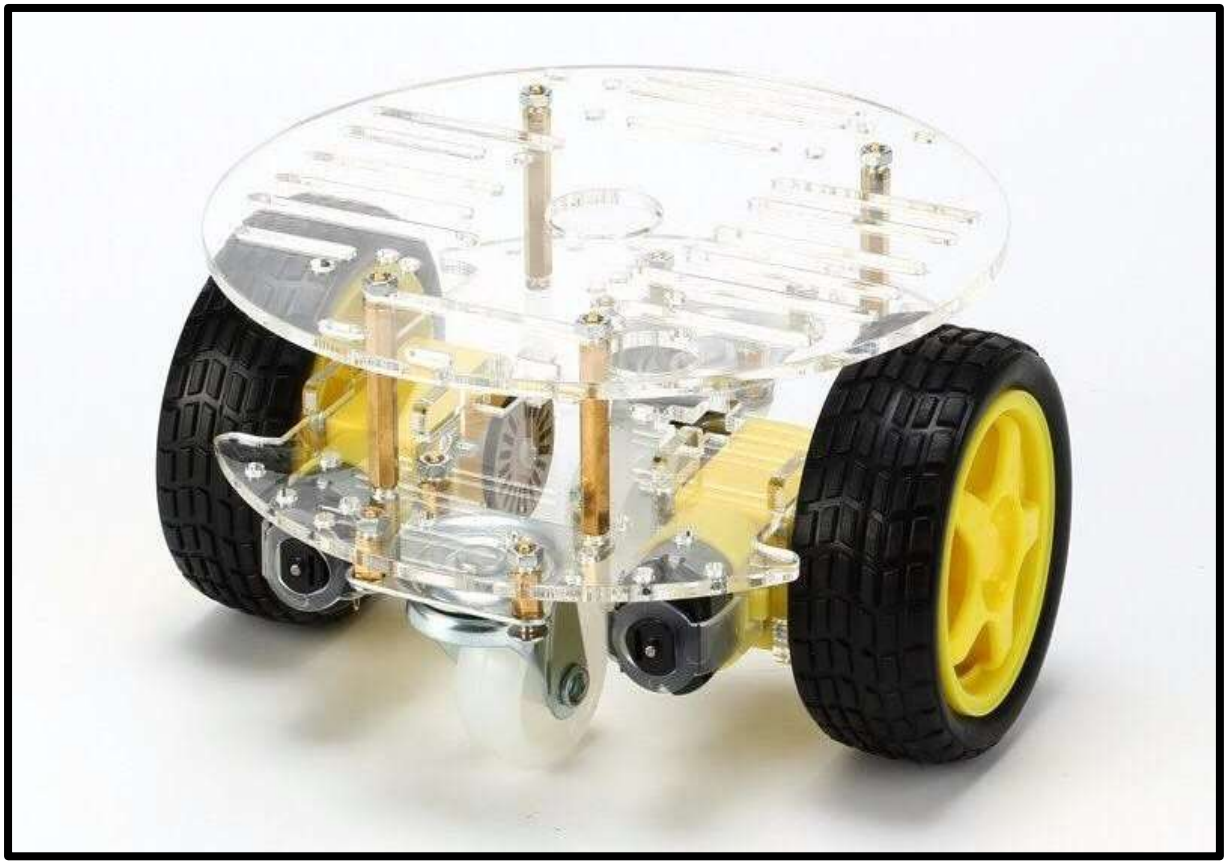


Figure 4.1 Base

4.1 Mechanism selection

Mechanism, we adopted for our line follower robot is a Base (Figure 1) that is the main body over which the PCB (Figure 2) is mounted by some Connecter (Figure 3) of some height. So, there is some space created between the PCB and Base. This Space is filled by placing the power supply/ Lithium Rechargeable Cells (Figure 4). The base's lower side is connected to 2 Motors (Figure 5) on each back side (Left and Right) by a rectangular shape connecter (Figure 6). These Motors Then connected to the Tires (Figure 7) which are responsible for moment (Front, Left or right). Now the front side of the line follower robot Base is directly connected to a freely moving wheel (Figure 8) in any direction according to the moment of

backward tires. Over the PCB, there will be the Circuit that is MUXs, Gates, Sil Headers, Diodes, Motor Driver. Bolts and Nuts (Figure 9) are used in different places for tightness. The 5 IR Sensors (Figure 10) are mounted on the front end of the Base according to the truth table condition.

Round 2 wheel Robot Car Chassis Mini Round Double-Deck kit comes with Round shape chassis, Motors, wheels and all required assembly parts to build a complete robot car. Unleash your inner Engineer and make your vehicle dreams a reality with the Mini Round Robot Chassis Kit.

Round 2 wheel Robot Car Chassis is a transparent platform for building a robot. The set includes two DC motors with wheels with a diameter of 65 mm. At the front and back supported by a metal rotating wheel. The chassis components are made of acrylic, have mounting holes allowing you to install all the sensors, controllers and others.

Transparent, universal platform, allowing you to build a robot, e.g. line follower or fighting robot (sumo). For control, you can use any controller, including Arduino with Motor Shield.

Round 2 wheel Robot Car Chassis gives you everything you need to build the shell of a 2-wheel-drive Mobile Platform Robot! You get the metal plates that make up the chassis, two DC drive motors with matching wheels, and a caster ball for balance. You'll fill in the rest with a power supply, microcontroller board, and motor controller.

The differential drive (two separately driven wheels) allows for a near zero turning radius while the high-strength aluminum alloy body plus high-quality high-speed motors make it suitable for flat indoor surfaces. Also it's adorably small.

Features:

1. Mechanical structure is simple,
2. it is easy to install.
3. This car is the tachometer encoder.
4. With 2 AA battery box.
5. Can be used for distance measurement, velocity.
6. Can use with other devices to realize function of tracing, obstacle avoidance, distance testing, speed testing, wireless remote control.

7. Gear Motor reduction radio: 1:48. Note: motor power supply is 3V~6V.

4.2 Platform Design

So, there are so many designs of which an individual can think of for a line follower robot. Some of them may be available in the market in easy cast or we can design some by yourself. The design maybe in rectangular form, maybe a triangular shape, maybe pentagonal shape or maybe a circular shape. So, it's just depends upon our feasibility. That is because the design wouldn't affect our project, this is because our project contains very simple configuration above the base. We just the Lithium Battery cells, a PCB and Motor Driver.

4.3 Material Selection and choices

Material selection is very important thing to do while doing an engineering project, especially when the project is of high level or there are a lot of load etc. over it. The commonly used materials used in different projects are aluminum, Glass, Plastic, Iron, Steel, Brass etc. different material have different properties according to which they are selected. Now in our case of project, we have chosen the base of plastic (Glass), this is because it has very less weight as compare to other and our project can be easily mounted over it without breakage etc. The bolts and nuts are of steel, so they can hold things together tightly for times. The connecters are also of plastic (Glass), as in our case, it can work easily without any problem.

CHAPTER 5- SOFTWARE/FIRMWARE DESIGN/THEORETICAL DESIGN

5.1 Explanation

Following are the inputs and outputs of the circuit.

Inputs

1. Left Sensor (LS)
2. Right Sensor (RS)
3. Right Centre Sensor (RC)
4. Left Centre Sensor (LS)
5. Centre Sensor (CS)
6. Ultrasonic Sensor
7. Voltage sensor
8. Current sensor
9. Rotary encoder (Tachometer)

Outputs

1. Left Motor
2. Right Motor
3. SD card
4. 16 X 4 LCD

5.2 SCHEMATIC COMPONENTS

1. Ultrasonic sensor
2. SD card module
3. Servo motor
4. TCRT5000 IR sensors
5. L298 motor driver
6. ESP 32
7. 16x4 LCD display
8. Motors

9. Current sensor
10. Voltage sensor
11. Potentiometer

5.2 PCB Making Procedure

1. To design PCB in proteus there is an ARES designing tool is exist that we will use to design PCB.

2. After that we will create our design through the usage of 2D graphics box Mode, 3. Press at the select layer and now choose board edge.

4. After that now construct the box shape in the working area. After that press mouse button then the color of green line will turn in yellow.

5. Now we can construct our circuit in that rectangle box.

6. After that press at the element and change its location if needed through the use of the rotation tab after that put it in the working area.

7. When you put all elements in the working area make a configuration of all these elements.

8. the location of any element can be varied through pressing at the selection mode and after that chose the element and move it to the new location.

9. After that link, all the elements of circuitry, Choose trach option and now we can vary the track width through choosing C or E option.

10. Choose the width of the screen according to layout of PCB circuitry.

11. After that link all elements used to press at the elements at endpoint there will be green colored line that will show move this line to another component where you have to link.

12. After making link green line will automatically diminish.

13. For a single layer, PCB designing put all elements of circuit at one side and make their links at the other side.

14. For two-layer PCB connection and interlinking of all components can be made at both sides.

15. If there is any fault exist in our design will show in red color circles.

16. To avoiding any fault, we can vary the track path. In the case of 2 layer PCB, we can vary track amongst layers through double-pressing the left button.

17. Red color track seen shown at upper portion and the track of blue-colored shown at lower portion among layers. It is 2 layer PCB.

18. Start tracking on the connections and use T50 for the track size. Also use square connectors of 5mm.When tracking is completed save the project in the similar file that we saved in proteus.

19. We can also select a three-dimensional display for a final look of our project.

20. We can analyze all joints of elements angles placements and other parameters.

5.3 Truth Table

IR1	IR2	IR3	L	R
1	1	1	0	0
1	1	1	0	1
1	1	1	0	1
1	1	1	0	1
1	1	0	1	0
1	1	0	0	1
1	1	0	0	1
1	1	0	0	1
1	0	1	1	0
1	0	1	1	1
1	0	1	1	1
1	0	1	1	1
1	0	0	1	0
1	0	0	1	0
1	0	0	1	0
1	0	0	0	1
0	1	1	1	0
0	1	1	1	1
0	1	1	1	1
0	1	1	1	1
0	1	0	1	0
0	1	0	1	0
0	1	0	0	1
0	1	0	0	1
0	0	1	1	0
0	0	1	1	1

0	0	1	1	1
0	0	1	1	1
0	0	0	1	0
0	0	0	1	0
0	0	0	0	0

Table 5.1 Truth Table

5.6 Controller Selections with features

Line following robot detects and follows a line. The line is black line on a white surface and vice versa. The line is sensed by sensor, proximity sensor and IR sensor. The proximity sensor used for path detection and IR sensor used for obstacle detection. These sensors mounted at front end give input to microcontroller (Arduino uno) which decides whether right motor or left motor will move to turn right, left, forward etc.

Chapter 6- Programming And Sensor Integration

6.1 Integration of LCD

RS pin of the LCD module is connected to digital pin 12 of the Arduino. R/W pin of the LCD is grounded. Enable pin of the LCD module is connected to digital pin 11 of the Arduino. In this project, the **LCD module and Arduino are interfaced in the 4-bit mode**. This means only four of the digital input lines (DB4 to DB7) of the LCD are used. This method is very simple, requires less connections and you can almost utilize the full potential of the LCD module. Digital lines DB4, DB5, DB6 and DB7 are interfaced to digital pins 5, 4, 3 and 2 of the Arduino. The 10K potentiometer is used for adjusting the contrast of the display. 560 ohm resistor R1 limits the current through the back light LED. The Arduino can be powered through the external power jack provided on the board. +5V required in some other parts of the circuit can be tapped from the 5V source on the Arduino board. The Arduino can be also powered from the PC through the USB port. The full program for interfacing LCD to Arduino is shown below.

CODE

```
/* This is a sketch to test 16x4 LCD:
#include LiquidCrystal lcd(8,9,4,5,6,7);
void setup() {
lcd.begin(16,4);
lcd.setCursor(0,0);
lcd.print("VOLTAGE");
lcd.setCursor(0,1);
lcd.print("CURRENT!");
lcd.setCursor(0,2);
lcd.print("P/N: ");
lcd.setCursor(0,3);
lcd.print("SD CARD");
}
void loop() {
}
```

6.2 Integration of Sd Card Module

Description	Command
Initializes the SD library and card. Enter the pin connected to the SS pin as a function's argument.	SD.begin(#sspin)
Tests whether a file or directory exists on the SD card.	SD.exists(filename)
Opens a file on the SD card in reading or writing mode. (If you leave the mode section blank, the file will open in reading mode by default) If the file is opened for writing, it will be created a file with this name if it doesn't already exist.	SD.open(filepath, mode)
Close the file and ensure that any data written to it is physically saved to the SD card.	file.close()*
Remove a file from the SD card.	SD.remove(filename)
Create a directory on the SD card	SD.mkdir(filename)
Remove a directory from the SD card.	SD.rmdir(filename
Returns the file name	file.name()*
Print data to the file	file.print(data)
Print data, followed by a carriage return and newline	file.println(data
Read from the file.	file.read()
Check if there are any bytes available for reading from the file.	file.available()

Storing data is one of the most important parts of every project. There are several ways to store data according to the data type and size. SD and micro SD cards are one of the most practical ones among the storage devices, which are used in devices such as mobile phones, minicomputers and etc.

CODE

```
#include <SPI.h>
#include <SD.h>
File myFile;
void setup() {
// Open serial communications and wait for port to open:
Serial.begin(9600);
while (!Serial) {
; // wait for serial port to connect. Needed for native USB port
only
}
Serial.print("Initializing SD card...");
if (!SD.begin(10)) {
Serial.println("initialization failed!");
while (1);
}
Serial.println("initialization done.");
// open the file. note that only one file can be open at a time,
```

```
// so you have to close this one before opening another.
myFile = SD.open("test.txt", FILE_WRITE);
// if the file opened okay, write to it:
if (myFile) {
Serial.print("Writing to test.txt...");
myFile.println("This is a test file :)");
myFile.println("testing 1, 2, 3.");
for (int i = 0; i < 20; i++) {
myFile.println(i);
}
// close the file:
myFile.close();
Serial.println("done.");
} else {
// if the file didn't open, print an error:
Serial.println("error opening test.txt");
}
}
void loop() {
// nothing happens after setup
}
```

6.3 Integration of Ultrasonic Sensor

An Ultrasonic Sensor is a device that measures distance to an object using **Sound Waves**. It works by sending out a sound wave at ultrasonic frequency and waits for it to bounce back from the object. Then, the time delay between transmission of sound and receiving of the sound is used to calculate the distance.

It is done using the formula **Distance = (Speed of sound * Time delay) / 2**

We divide the distance formula by 2 because the sound waves travel a round trip i.e from the sensor and back to the sensor which doubles the actual distance.

CODE

```
/*
* Ultrasonic Sensor HC-SR04 interfacing with Arduino.
*/
// defining the pins
const int trigPin = 9;
const int echoPin = 10;
// defining variables
long duration;
int distance;
```

```
void setup() {
pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
pinMode(echoPin, INPUT); // Sets the echoPin as an Input
Serial.begin(9600); // Starts the serial communication
}
void loop() {
// Clears the trigPin
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
// Sets the trigPin on HIGH state for 10 micro seconds
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
// Reads the echoPin, returns the sound wave travel time in
microseconds
duration = pulseIn(echoPin, HIGH);
// Calculating the distance
distance= duration*0.034/2;
// Prints the distance on the Serial Monitor
Serial.print("Distance: ");
Serial.println(distance);
}
```

6.4 Integration of Optical Encoder

The optical encoder used, its purpose here is to measure the speed of the robot.

CODE

```
int encoder_pin = 2;
unsigned int rpm = 0;
volatile byte pulses = 0;
unsigned long timeold = 0;
unsigned int pulsesperturn = 20;
static volatile unsigned long debounce = 0;
void setup(){
   Serial.begin(9600);
   pinMode(encoder_pin, INPUT);
   attachInterrupt(0, counter, RISING);
   pulses = 0;
```

```
   rpm = 0;
   timeold = 0;
  Serial.print("Seconds ");
  Serial.print("RPM ");
}
 void loop(){
   if (millis() - timeold >= 1000){
      noInterrupts();
      rpm = (60 * 1000 / pulsesperturn )/ (millis() -
timeold)* pulses;
      timeold = millis();
      Serial.print(millis()/1000); Serial.print("       ");
      Serial.print(rpm,DEC); Serial.print("   ");
      pulses = 0;
      interrupts(); // Restart the interrupt processing
   }
  }
 void counter(){
  if(  digitalRead (encoder_pin) && (micros()-debounce > 500)
&& digitalRead (encoder_pin) ) {
                              //Check again that the encoder
sends a good signal and then check that the time is greater
than 1000 microseconds and check again that the signal is
correct.
        debounce = micros(); // Stores the time to check that
we do not count the bounce in the signal.
        pulses++;}
        else ;
```

6.5 Integration of IR Sensor

The connections for the IR sensor with the Arduino are as follows: Connect the negative wire on the IR sensor to GND on the Arduino. Connect the middle of the IR sensor which is the VCC to 5V on the Arduino. Connect the signal pin on the IR sensor to pin 8 on the Arduino.

CODE

```
void setup() {
  // put your setup code here, to run once:
pinMode(4,INPUT);
pinMode(12,OUTPUT);//LED
}

void loop() {
  // put your main code here, to run repeatedly:
if(digitalRead(4)==LOW){
  digitalWrite(12,HIGH);
}
```

```
else{
  digitalWrite(12,LOW);
}
}
```

6.6 Integration of Current Sensor

First, the load. I have used a 12V DC Motor along with a 12V power supply. The screw terminals of the ASC712 Current Sensor Module board are connected in series with the motor and power supply. Then connect the VCC, GND and OUT of the ASC712 board to +5V, GND and A0 of Arduino. Now, in order to view the results, a 16×2 LCD is connected to Arduino. Its RS, E, D4-D7 pins are connected to Digital I/O Pins 7 through 2 of Arduino.A 10KΩ POT is connected to Pin 3 of LCD and its VCC and GND are connected to +5V and GND.

CODE

```
void setup() {
  Serial.begin(9600); //Start Serial Monitor to display
current read value on Serial monitor
}

void loop() {
unsigned int x=0;
float AcsValue=0.0,Samples=0.0,AvgAcs=0.0,AcsValueF=0.0;

  for (int x = 0; x < 150; x++){ //Get 150 samples
  AcsValue = analogRead(A0);     //Read current sensor values
  Samples = Samples + AcsValue;  //Add samples together
  delay (3); // let ADC settle before next sample 3ms
}
AvgAcs=Samples/150.0;//Taking Average of Samples

AcsValueF = (2.5 - (AvgAcs * (5.0 / 1024.0)) )/0.185;

Serial.print(AcsValueF);//Print the read current on Serial
monitor
delay(50);
}
```

6.7 Integration of Voltage Sensor

The Arduino Voltage Sensor Interface is pretty straight forward. Connect the voltage to be measured to the screw terminal of the Voltage Sensor, connected the output of the voltage divider to the Arduino. That's it.

CODE

```
const float voltage_sensor= A2;
float voltage ;
float value;
float r2 = 39000.00 ;
float r1 = 10000.00 ;
void setup()
{
  // put your setup code here, to run once:
  Serial.begin(9600);
  pinMode(voltage_sensor, INPUT);
}
void loop() {
  // put your main code here, to run repeatedly
  value= analogRead(voltage_sensor);
  voltage= (value*(5.0/682)*(r1 + r2))/r2 ;
  Serial.print("Voltage Value :  ");
Serial.println(voltage);
}
```

6.8 Integration of ESP 32

ESP32 is highly-integrated with in-built antenna switches, RF balun, power amplifier, low-noise receive amplifier, filters, and power management modules. ESP32 adds priceless functionality and versatility to your applications with minimal Printed Circuit Board (PCB) requirements.

CODE:

```
// Load Wi-Fi library
#include <WiFi.h>

// Replace with your network credentials
const char* ssid = "Partners";
const char* password = "Harami07";

// Set web server port number to 80
WiFiServer server(80);

// Variable to store the HTTP request
String header;

// Auxiliar variables to store the current output state
String output26State = "off";
String output27State = "off";

// Assign output variables to GPIO pins
const int output26 = 26;
const int output27 = 27;
```

```
// Current time
unsigned long currentTime = millis();
// Previous time
unsigned long previousTime = 0;
// Define timeout time in milliseconds (example: 2000ms = 2s)
const long timeoutTime = 2000;

void setup() {
  Serial.begin(115200);
  // Initialize the output variables as outputs
  pinMode(output26, OUTPUT);
  pinMode(output27, OUTPUT);
  // Set outputs to LOW
  digitalWrite(output26, LOW);
  digitalWrite(output27, LOW);

  // Connect to Wi-Fi network with SSID and password
  Serial.print("Connecting to Konain Haider  ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  // Print local IP address and start web server
  Serial.println("");
  Serial.println("WiFi connected.");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
  server.begin();
}

void loop(){
  WiFiClient client = server.available();   // Listen for
incoming clients

  if (client) {                             // If a new
client connects,
    currentTime = millis();
    previousTime = currentTime;
    Serial.println("New Client.");          // print a
message out in the serial port
    String currentLine = "";                // make a String
to hold incoming data from the client
    while (client.connected() && currentTime - previousTime
<= timeoutTime) {  // loop while the client's connected
      currentTime = millis();
      if (client.available()) {             // if there's
bytes to read from the client,
        char c = client.read();             // read a byte,
then
```

```
        Serial.write(c);                    // print it out
the serial monitor
        header += c;
        if (c == '\n') {                    // if the byte is
a newline character
          // if the current line is blank, you got two
newline characters in a row.
          // that's the end of the client HTTP request, so
send a response:
          if (currentLine.length() == 0) {
            // HTTP headers always start with a response code
(e.g. HTTP/1.1 200 OK)
            // and a content-type so the client knows what's
coming, then a blank line:
            client.println("HTTP/1.1 200 OK");
            client.println("Content-type:text/html");
            client.println("Connection: close");
            client.println();

            // turns the GPIOs on and off
            if (header.indexOf("GET /26/on") >= 0) {
              Serial.println("GPIO 26 on");
              output26State = "on";
              digitalWrite(output26, HIGH);
            } else if (header.indexOf("GET /26/off") >= 0) {
              Serial.println("GPIO 26 off");
              output26State = "off";
              digitalWrite(output26, LOW);
            } else if (header.indexOf("GET /27/on") >= 0) {
              Serial.println("GPIO 27 on");
              output27State = "on";
              digitalWrite(output27, HIGH);
            } else if (header.indexOf("GET /27/off") >= 0) {
              Serial.println("GPIO 27 off");
              output27State = "off";
              digitalWrite(output27, LOW);
            }

            // Display the HTML web page
            client.println("<!DOCTYPE html><html>");
            client.println("<head><meta name=\"viewport\"
content=\"width=device-width, initial-scale=1\">");
            client.println("<link rel=\"icon\"
href=\"data:,\">");
            // CSS to style the on/off buttons
            // Feel free to change the background-color and
font-size attributes to fit your preferences
            client.println("<style>html { font-family:
Helvetica; display: inline-block; margin: 0px auto; text-
align: center;}");
            client.println(".button { background-color:
#4CAF50; border: none; color: white; padding: 16px 40px;");
```

```
            client.println("text-decoration: none; font-size:
30px; margin: 2px; cursor: pointer;}");
            client.println(".button2 {background-color:
#555555;}</style></head>");

            // Web Page Heading
            client.println("<body><h1>ESP32 Web
Server</h1>");

            // Display current state, and ON/OFF buttons for
GPIO 26
            client.println("<p>GPIO 26 - State " +
output26State + "</p>");
            // If the output26State is off, it displays the
ON button
            if (output26State=="off") {
              client.println("<p><a href=\"/26/on\"><button
class=\"button\">ON</button></a></p>");
            } else {
              client.println("<p><a href=\"/26/off\"><button
class=\"button button2\">OFF</button></a></p>");
            }

            // Display current state, and ON/OFF buttons for
GPIO 27
            client.println("<p>GPIO 27 - State " +
output27State + "</p>");
            // If the output27State is off, it displays the
ON button
            if (output27State=="off") {
              client.println("<p><a href=\"/27/on\"><button
class=\"button\">ON</button></a></p>");
            } else {
              client.println("<p><a href=\"/27/off\"><button
class=\"button button2\">OFF</button></a></p>");
            }
            client.println("</body></html>");

            // The HTTP response ends with another blank line
            client.println();
            // Break out of the while loop
            break;
          } else { // if you got a newline, then clear
currentLine
            currentLine = "";
          }
        } else if (c != '\r') {  // if you got anything else
but a carriage return character,
          currentLine += c;      // add it to the end of the
currentLine
        }
      }
```

```
    }
    // Clear the header variable
    header = "";
    // Close the connection
    client.stop();
    Serial.println("Client disconnected.");
    Serial.println("");
  }
}
```

6.9 Final complete code of project

Microcontroller Code

```
#include <LiquidCrystal.h> //library for LCD
// initialize the library with the numbers of the interface pins
 LiquidCrystal lcd(8, 9, 10, 11, 12, 13);
// Voltage Sensor
int analogInput = A1;     //voltage sensor
float vout = 0.0;
float vin = 0.0;
float R1 = 30000.0; //30k
float R2 = 7500.0; //7500 ohm resistor, I tweaked this
int value = 0;
int sensor = A2; //u sensor
unsigned long start_time = 0;
unsigned long end_time = 0;
int steps=0;
float steps_old=0;
float temp=0;
float rps=0;
//Measuring Current Using ACS712
const int currentPin = A0; // current sensor is attached
int sensitivity = 66;  // 66-185 mv/A
int adcValue= 0; //
int offsetVoltage = 2500; // 0.25 mv 0v ki value ko remove karne k liye
double adcVoltage = 0;
double currentValue = 0;
//variables for battery level indicator
int f=2, e=3,d=4, c=5, b=6,a=7;
void setup() {
```

```
  //baud rate
  Serial.begin(9600);//baud rate at which arduino communicates with
Laptop/PC
  // set up the LCD's number of columns and rows
  lcd.begin(16, 4);  //LCD order
pinMode(sensor,INPUT_PULLUP);
//  lcd.setCursor(0,0);
//  lcd.print(" STEPS - 0");
//  lcd.setCursor(0,1);
//  lcd.print(" RPS   - 0.00");
pinMode(a,OUTPUT);
pinMode(b,OUTPUT);
pinMode(c,OUTPUT);
pinMode(d,OUTPUT);
pinMode(e,OUTPUT);
pinMode(f,OUTPUT);
digitalWrite(f,HIGH);
delay(500);
digitalWrite(e,HIGH);
delay(500);
digitalWrite(d,HIGH);
delay(500);
digitalWrite(c,HIGH);
delay(500);
digitalWrite(b,HIGH);
delay(500);
digitalWrite(a,HIGH);
delay(500);
digitalWrite(a,LOW);
delay(500);
digitalWrite(b,LOW);
delay(500);
digitalWrite(c,LOW);
delay(500);
digitalWrite(d,LOW);
delay(500);
digitalWrite(e,LOW);
delay(500);
```

```
digitalWrite(f,LOW);
delay(500);
  lcd.setCursor(0,0);//Setting cursor on LCD
  lcd.print("CHEAP THRILLS");//Prints on the LCD
  delay(1000);
  lcd.setCursor(1,1);
  lcd.print("~HASSAN 190979");
  lcd.setCursor(1,2);
  lcd.print("~MARIUM 190967");
  lcd.setCursor(1,3);
  lcd.print("~UZAIR 190970");
  delay(2000);//time delay for 3 sec
  lcd.clear();//clearing the LCD display
  lcd.display();//Turning on the display again
  lcd.setCursor(1,0);//setting LCD cursor
  lcd.print("Values from");//prints on LCD
  lcd.setCursor(1,1);
  lcd.print("Current Sensor");
  lcd.setCursor(2,2);
  lcd.print("ACS712 and");
  lcd.setCursor(1,4);
  lcd.print("Voltage sensor");
  lcd.clear();
  delay(500);//delay for 1 sec}
void loop() //method to run the source code repeatedly
{start_time=millis();
 end_time=start_time+1000;
 while(millis()<end_time)
 {if(digitalRead(sensor))
   { steps=steps+1;
    while(digitalRead(sensor));}
   lcd.setCursor(9,2);
   lcd.print(steps);
   lcd.print("   ");}
// voltage
   value = analogRead(analogInput);
   vout = (value * 5.0) / 1024.0;
```

```
   vin = vout / (R2/(R1+R2));
if(vin>11.88)
{ digitalWrite(a,HIGH);
   digitalWrite(b,HIGH);
    digitalWrite(c,HIGH);
     digitalWrite(d,HIGH);
      digitalWrite(e,HIGH);
       digitalWrite(f,HIGH);}
if (vin<=11.46 && vin>11.28)
{ digitalWrite(a,LOW);
   digitalWrite(b,HIGH);
    digitalWrite(c,HIGH);
     digitalWrite(d,HIGH);
      digitalWrite(e,HIGH);
       digitalWrite(f,HIGH); }
if (vin<=11.12 && vin>10.98)
{digitalWrite(a,LOW);
   digitalWrite(b,LOW);
    digitalWrite(c,HIGH);
     digitalWrite(d,HIGH);
      digitalWrite(e,HIGH);
       digitalWrite(f,HIGH);}
if (vin<=10.90 && vin>10.79)
{ digitalWrite(a,LOW);
   digitalWrite(b,LOW);
    digitalWrite(c,LOW);
     digitalWrite(d,HIGH);
      digitalWrite(e,HIGH);
       digitalWrite(f,HIGH);}
if (vin<=10.60 && vin>10.53)
{digitalWrite(a,LOW);
   digitalWrite(b,LOW);
    digitalWrite(c,LOW);
     digitalWrite(d,LOW);
      digitalWrite(e,HIGH);
       digitalWrite(f,HIGH);}
if  (vin<=10.53)
```

```
{ digitalWrite(a,LOW);
   digitalWrite(b,LOW);
    digitalWrite(c,LOW);
     digitalWrite(d,LOW);
      digitalWrite(e,LOW);
       digitalWrite(f,HIGH);}
// current
  adcValue = analogRead(currentPin);
  adcVoltage = (adcValue / 1024.0) * 5000;
  currentValue = ((adcVoltage - offsetVoltage) / sensitivity);
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("Current =       ");
  lcd.setCursor(10,0);
  lcd.print(currentValue,2); // upto two decimal points
  lcd.setCursor(14,0);
  lcd.print("A");
   lcd.setCursor(0,1);
 lcd.print("Voltage=");
 lcd.setCursor(8,1);
 lcd.print(vin,2);
 lcd.setCursor(12,1);
 lcd.print("V");
lcd.setCursor(0,2);
  lcd.print(" STEPS = 0");
  lcd.setCursor(0,3);
  lcd.print(" RPS   = 0.00");
      temp=steps-steps_old;
    steps_old=steps;
    rps=(temp/20);
    lcd.setCursor(9,3);
    lcd.print(rps);
    lcd.print("   ");}
```

CODE:

```
#include <SPI.h>
#include <SD.h>
#include <Servo.h> //Servo motor library. This is standard library
```

```
#include <NewPing.h> //Ultrasonic sensor function library. You must install this library

File myFile;

//sensor pins

#define trig_pin A4 //analog output 1

#define echo_pin A5 //analog input 2

#define maximum_distance 200

boolean goesForward = false;

int distance = 100;

NewPing sonar(trig_pin, echo_pin, maximum_distance); //sensor function

Servo servo_motor; //our servo name

void setup()

{

Serial.begin(9600);

SD.begin(10);

  Serial.println("initialization done.");

 servo_motor.attach(8); //our servo pin

 servo_motor.write(115);

 delay(100);

 distance = readPing();

 pinMode(2, INPUT);//left sensor 1

 pinMode(4, INPUT); //left sensor 2

 pinMode(5, INPUT); //middle sensor3

 pinMode(6, INPUT); //right sensor 4

 pinMode(7, INPUT); //right sensor5

 pinMode(3, OUTPUT);  // pwm

  pinMode(8, OUTPUT);

 pinMode(A0, OUTPUT); //LEFT MOTOR.

 pinMode(A1, OUTPUT);

 pinMode(A2, OUTPUT); //RIGHT MOTOR.

 pinMode(A3, OUTPUT);}

 // put your setup code here, to run once

void loop(){

 int distanceRight = 0;

 int distanceLeft = 0;

 delay(50);

 if (distance <= 25){

 distanceRight = lookRight();
```

```
 delay(0);
 distanceLeft = lookLeft();
 delay(0);}
 distance = readPing();
if (distance<=25){
STOP();}
else{
 int a = digitalRead(2);
 int b = digitalRead(4);
int c = digitalRead(5);
 int d = digitalRead(6);
 int e = digitalRead(7);
if ((a == LOW && b == LOW && c == HIGH && d==LOW && e== LOW)||(a == LOW &&
b == HIGH && c == HIGH &&
d==HIGH && e== LOW)||(a == HIGH && b == HIGH && c == HIGH && d==HIGH && e==
HIGH))//FORWARD
 { analogWrite(A0, 0);
 analogWrite(A1, 255);
 analogWrite(A2, 0);
 analogWrite(A3, 255);
 analogWrite(3, 100);
f(); }
if ((a == LOW && b == HIGH && c == LOW && d==LOW && e==LOW)||(a==HIGH &&
b== HIGH && c==LOW &&d==
LOW &&e==LOW)||(a==HIGH && b== HIGH && c==HIGH &&d== LOW
&&e==LOW)||(a==HIGH && b== LOW &&
c==LOW &&d== LOW &&e==LOW)||(a==LOW && b== HIGH && c==HIGH &&d== LOW
&&e==LOW)||(a==HIGH &&
b== HIGH && c==HIGH && d==HIGH && e==LOW)) //LEFT TURN
 {
 analogWrite(A0, 0);
 analogWrite(A1, 255);
 analogWrite(A2, 0);
 analogWrite(A3, 0);
 analogWrite(3, 100);
l(); }
if ((a == LOW && b == LOW && c == LOW && d==HIGH && e==LOW)||(a==LOW && b==
LOW && c==LOW &&d==
HIGH &&e==HIGH)||(a==LOW && b== LOW && c==HIGH &&d== HIGH
&&e==HIGH)||(a==LOW && b== LOW &&
```

```
c==LOW &&d== LOW &&e==HIGH)||(a==LOW && b== HIGH && c==HIGH &&d== HIGH
&&e==HIGH)||(a==LOW &&
b== LOW && c==HIGH &&d== HIGH &&e==LOW) ) //RIGHT TURN
 {
 analogWrite(A0, 0);
 analogWrite(A1, 0);
 analogWrite(A2, 0);
 analogWrite(A3, 255);
 analogWrite(3, 100);
 r(); }
if (a == LOW && b == LOW && c == LOW && d==LOW && e== LOW) // BACK {
 analogWrite(A0, 0);
 analogWrite(A1, 255);
 analogWrite(A2, 255);
 analogWrite(A3, 0);
 analogWrite(3, 100);
bs(); }}}
int lookRight(){
 servo_motor.write(50);
 delay(100);
 int distance = readPing();
 delay(100);
 servo_motor.write(115);
 return distance;}
int lookLeft(){
 servo_motor.write(170);
 delay(100);
 int distance = readPing();
 delay(100);
 servo_motor.write(115);
 return distance;
 delay(100);}
int readPing(){
 delay(70);
 int cm = sonar.ping_cm();
 if (cm==0)
 { cm=250;  }
 return cm; }
```

```
void STOP(){
 analogWrite(A0, 0);
 analogWrite(A1, 0);
 analogWrite(A2, 0);
 analogWrite(A3, 0);
 analogWrite(3, 100);
 obs();
 }
 void f(){
  // open the file. note that only one file can be open at a time,
  // so you have to close this one before opening another.
  myFile = SD.open("test.txt", FILE_WRITE);
  // re-open the file for reading:
  myFile = SD.open("test.txt");
  Serial.println("forward.txt:");
    Serial.println("forward.txt:");
    myFile.close(); }
void r(){
  // open the file. note that only one file can be open at a time,
  // so you have to close this one before opening another.
  myFile = SD.open("test.txt", FILE_WRITE);
  // re-open the file for reading:
  myFile = SD.open("test.txt");
  Serial.println("right.txt:");
    Serial.println("right.txt:");
    myFile.close(); }
  void l(){
  // open the file. note that only one file can be open at a time,
  // so you have to close this one before opening another.
  myFile = SD.open("test.txt", FILE_WRITE);
  // re-open the file for reading:
  myFile = SD.open("test.txt");
  Serial.println("left.txt:");
    Serial.println("left.txt:");
    myFile.close();  }
  void bs(){
  // open the file. note that only one file can be open at a time,
```

```
  // so you have to close this one before opening another.
  myFile = SD.open("test.txt", FILE_WRITE)
  // re-open the file for reading:
  myFile = SD.open("test.txt");
  Serial.println("back.txt:");
    Serial.println("back.txt:");
    myFile.close(); }
 void obs(){
  // open the file. note that only one file can be open at a time,
  myFile = SD.open("test.txt", FILE_WRITE)
  // re-open the file for reading:
  myFile = SD.open("test.txt");
  Serial.println("Obstacle.txt:");
    Serial.println("Obstacle.txt:");
    myFile.close();
  }
```

CHAPTER 7 SIMULATIONS AND FINAL INTEGRATIONS

7.1 SCHEMATIC CIRCUIT

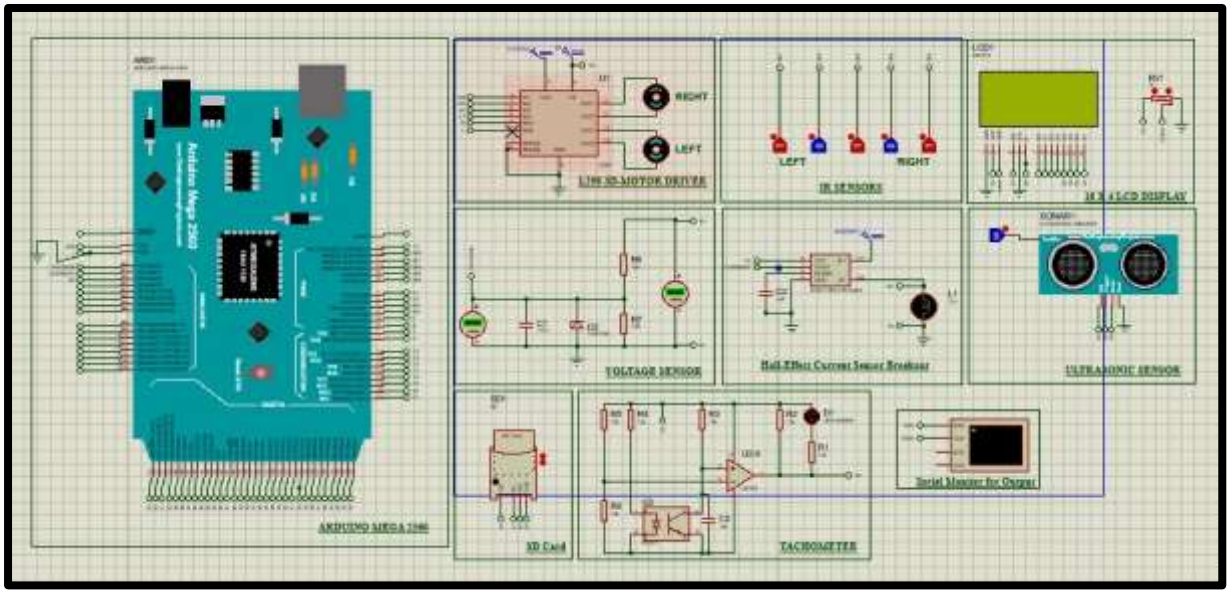


Figure 7.1 SCHEMATIC CIRCUIT

7.2 SIMULATION

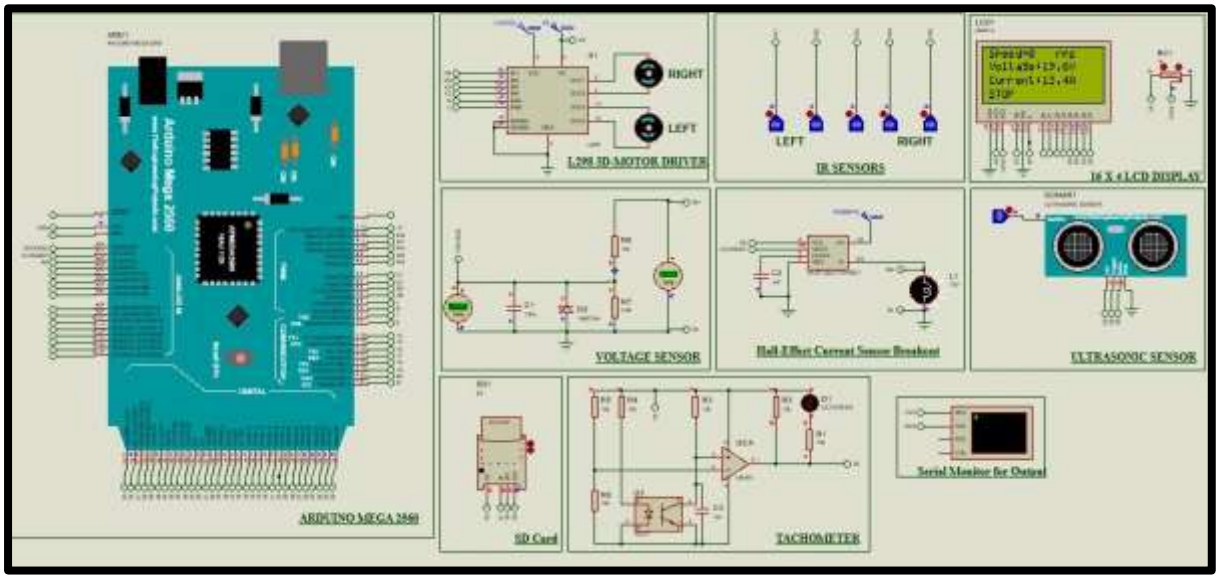


Figure 7.2 SIMULATION OF CONTROLLER 1

7.3 SCHEMATIC CIRCUIT OF PCB

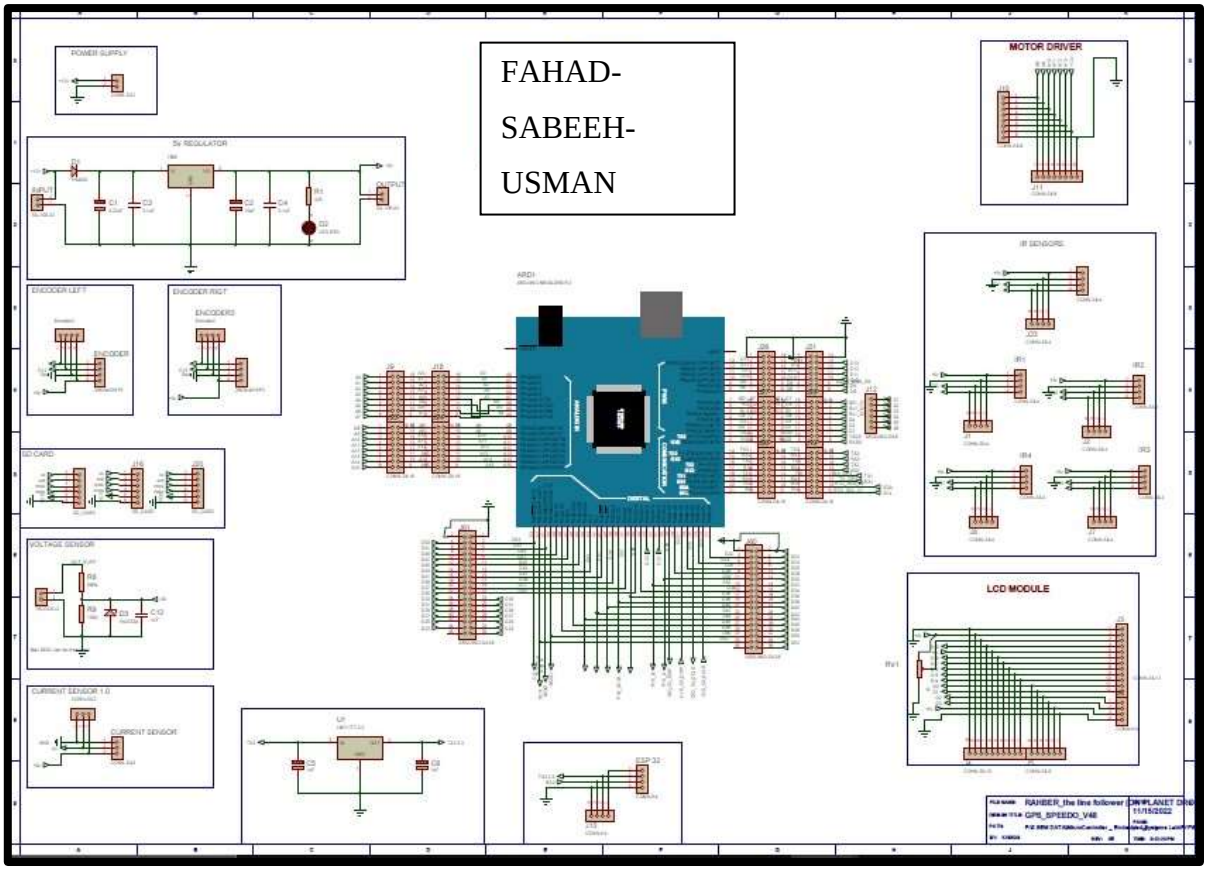


Figure 7.3 PCB SCHEMATIC

7.4 PCB ON PROTEUS

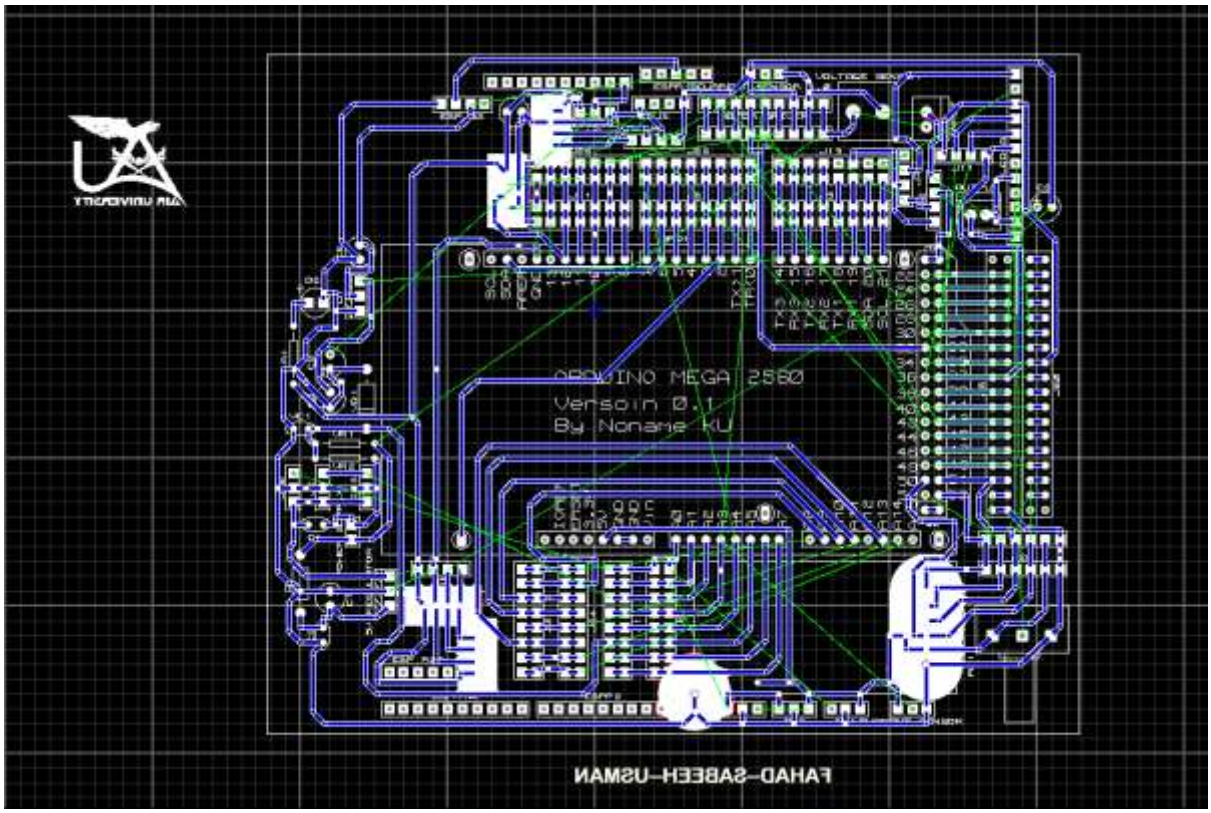


Figure 7.4 PCB LAYOUT

7.5 3D VISUALIZER

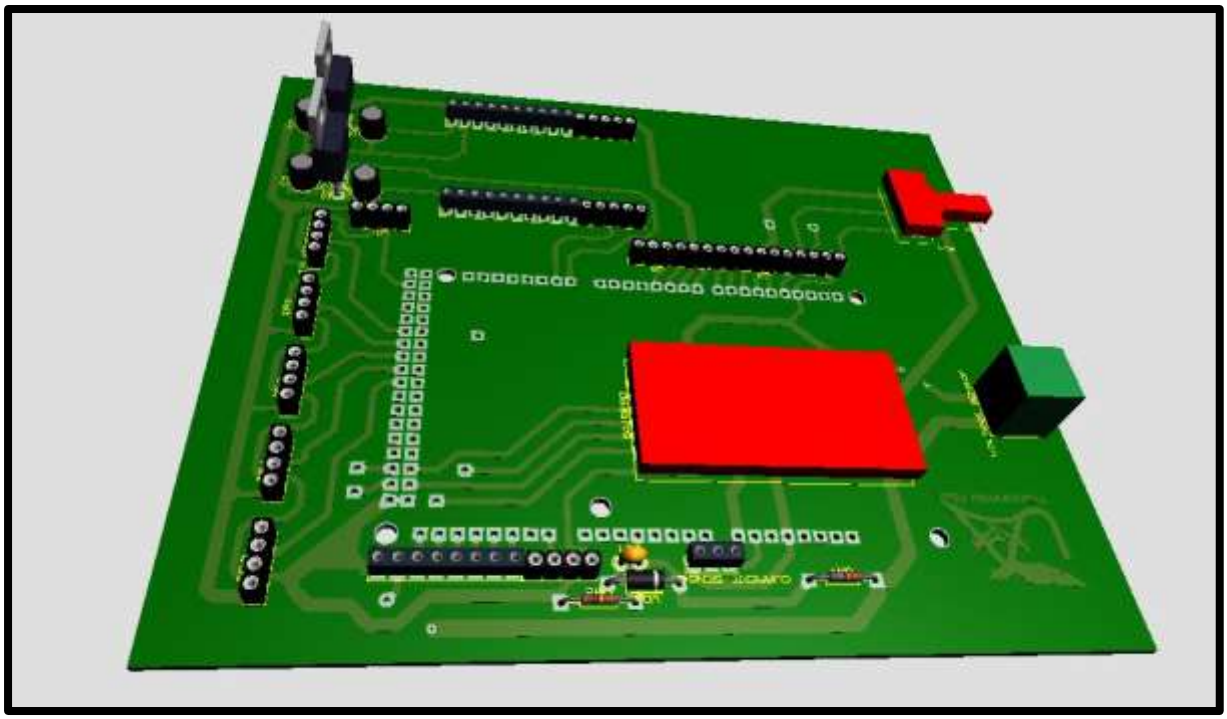


Figure 7.6 3D VISUALIZER

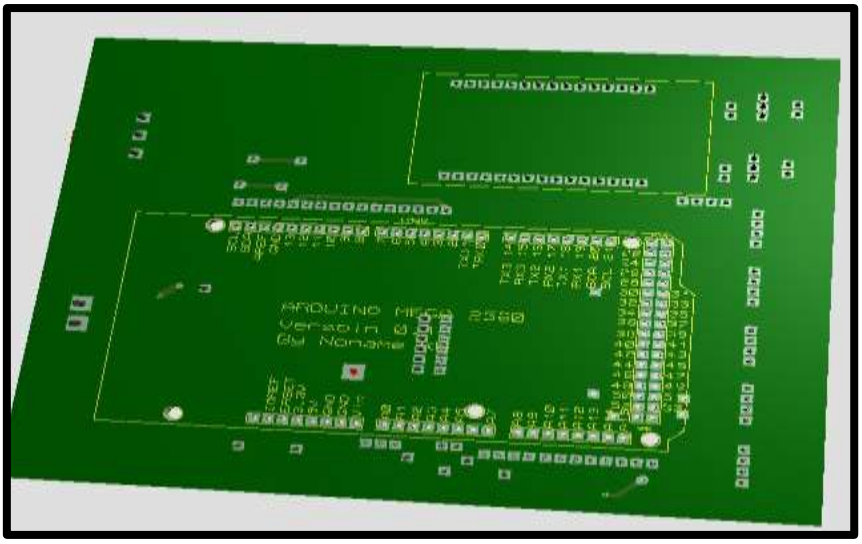


Figure 7.7 3D VISUALIZER

7.6 PCB DESIGNING CHALLENGES

ROUTING CHALLENGES

The PCB designing was not easy for us because it is our first time we are implementing such complex circuit. First of all the placement of all components were to be decided very carefully otherwise the circuit routing connections would be more and more complex. We were also adviced to make only one-layer PCB becaused double layer PCB is not easy to be implemented in real life. So what we have to do is to place our connection and do routing in a way that least no of layers would be made. We also faced a lot of DRC and CRC errors and we had to eradicate all of them. We made PCB about 10-15 times and in last there were only two to three jumper wires.

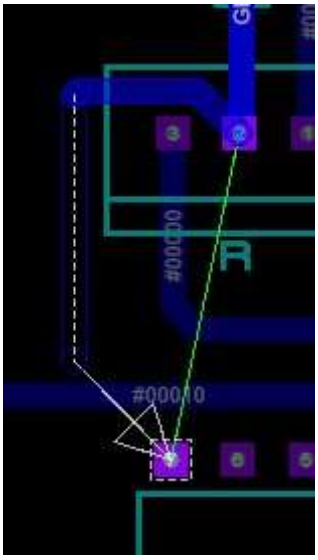


Figure 7.8 ERRORS

Figure 7.9 NO OF ERRORS

SQUARE CONNECTORS

We were also advised to use square connectors in place of circular connectors. But the size of square connectors was 50mm which were not easy to implement where there were routing which is very close to the connectors. So then we had to reduce the size of connectors.

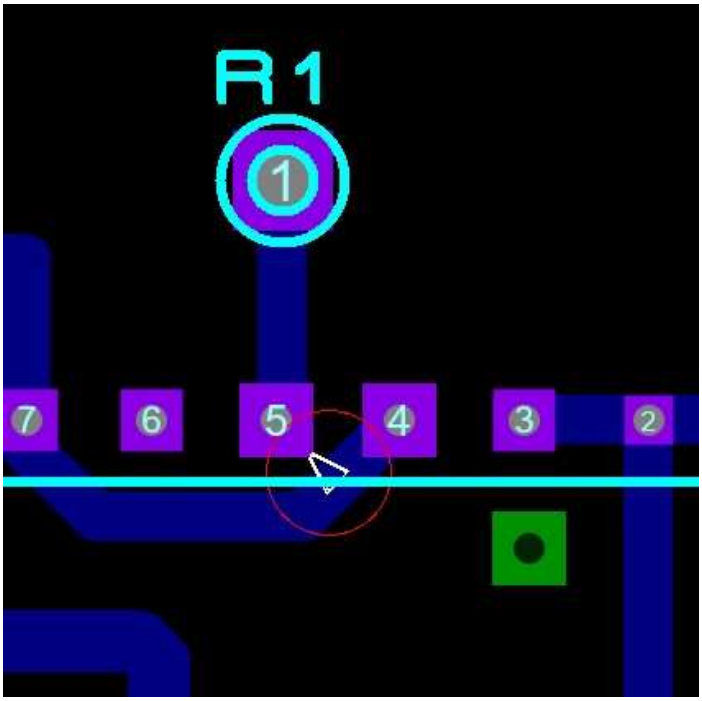


Figure 7.10 CONNECTOR ERROR

CHAPTER 8 HARDWARE IMPLEMENTATION

8.1 TESTING

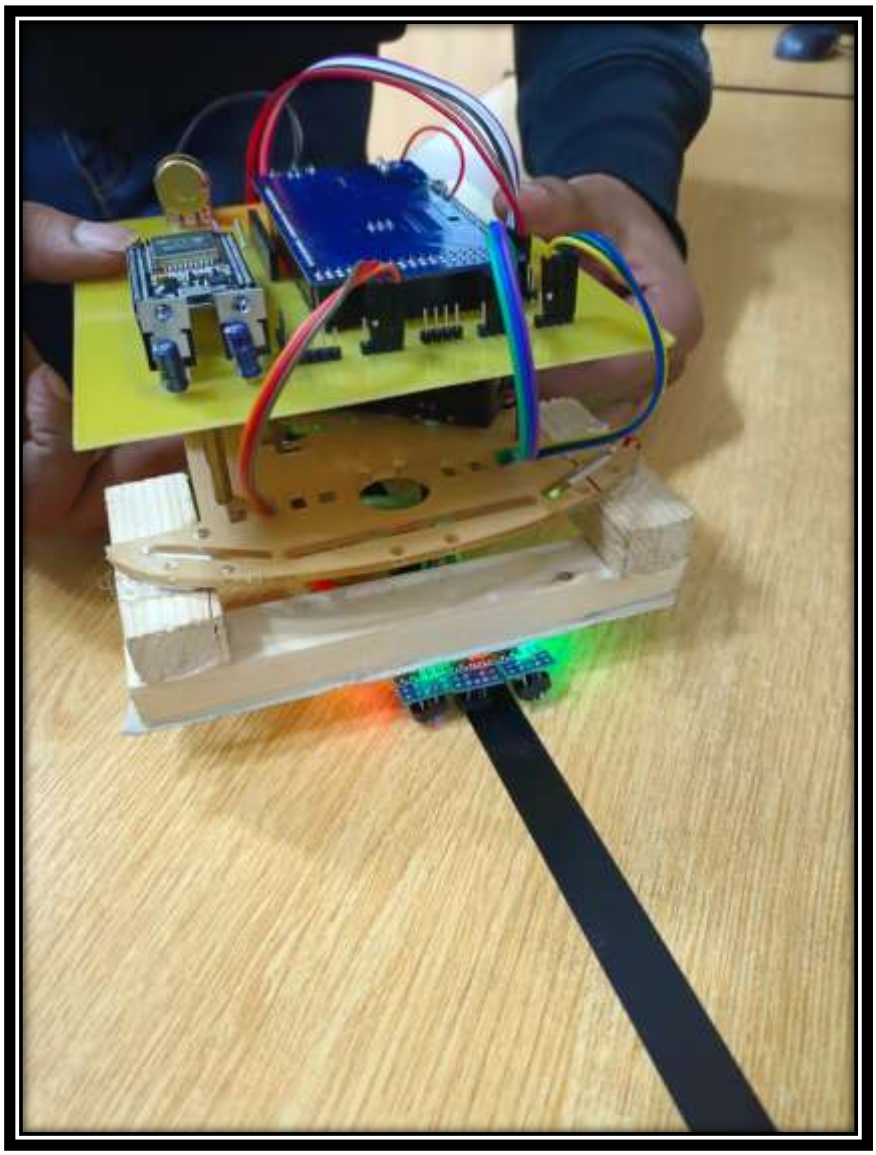


Figure 8.1 TESTING

8.2 IMPLEMENTATION ON PCB

1. For making PCB we have to do print our final PCB layout on sticky paper.
2. You should take the mirror print out.
3. Select the output in black both from the PCB design software and the printer driver settings.
4. Cut the copper board according to the size of the layout using a hacksaw or a cutter.
5. Next, rub the copper side of the PCB using steel wool or abrasive sponge scrubs. This removes the top oxide layer of copper as well as the photoresist layer. Sanded surfaces also allow the image from the paper to stick better.

6. Transfer the printed image (taken from a laser printer) from the photo paper to the board. Make sure to flip top layer horizontally. Put the copper surface of the board on the printed layout. Ensure that the board is aligned correctly along the borders of the printed layout and use tape to hold the board and the printed paper in the correct position.
7. Using the circuit as a reference, draw a basic sketch on the copper plate with a pencil. Once your sketch looks good, trace over it with a permanent black marker.
8. After printing on glossy paper, we iron it image side down to the copper side, then heat up the electric iron to the maximum temperature.
9. Put the board and photo paper arrangement on a clean wooden table (covered with a table cloth) with the back of the photo paper facing you.
10. Using pliers or a spatula, hold one end and keep it steady. Then put the hot iron on the other end for about 10 seconds. Now, iron the photo paper all along using the tip while applying a little pressure for about 5 to 15 mins.
11. Pay attention to the edges of the board – you need to apply pressure and do the ironing slowly.
12. Doing a long hard press seems to work better than moving the iron around.
13. The heat from the iron transfers the ink printed on the glossy paper to the copper plate.
14. After ironing, place the printed plate lukewarm water for about 10 minutes. The paper will dissolve, then you can remove the paper gently. Remove the paper by peeling it from a low angle.
15. Take a plastic box and fill it up with some water.
16. Dissolve 2-3 teaspoons of ferric chloride power in the water.
17. Dip the PCB into the etching solution (Ferric chloride solution, FeCl3) for approximately 30 mins.
18. The FeCl3 reacts with the unmasked copper and removes the unwanted copper from the PCB.
19. This process is called Etching. Use pliers to take out the PCB and check if the entire unmasked area has been etched or not. In case it is not etched, leave it in the solution for some more time.
20. A few drops of thinner (nail polish remover works well) on a pinch of cotton wool will remove completely the toner/ink on the plate, exposing the copper surface. Rinse carefully and dry with a clean cloth or kitchen paper. Trim to final size and smoothen edges with sandpaper.

21. Now, drill holes using a PCB driller like this PCB driller and solder all your cool components to the board.

PCB Board

Figure 8.2 PCB Board

PCB Board After soldering

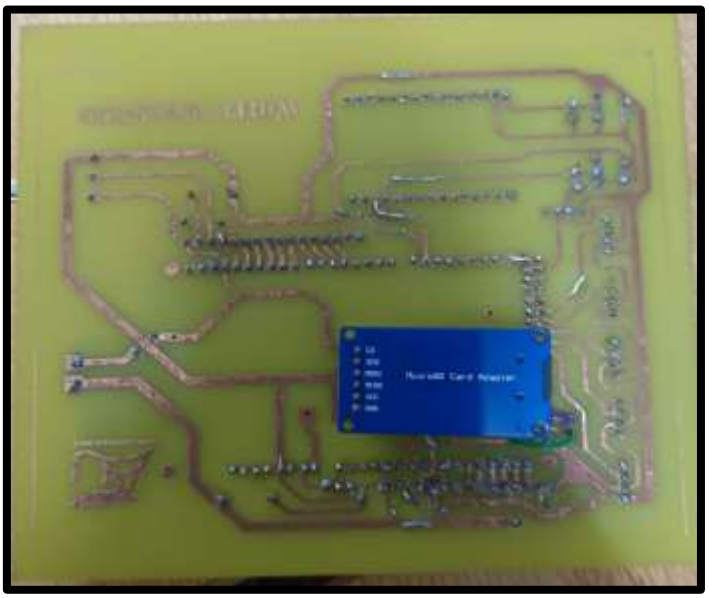
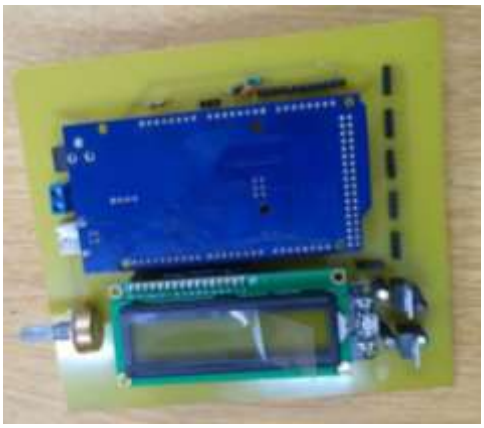


Figure 8.3 PCB Board After Soldering

Placing Components on PCB Board



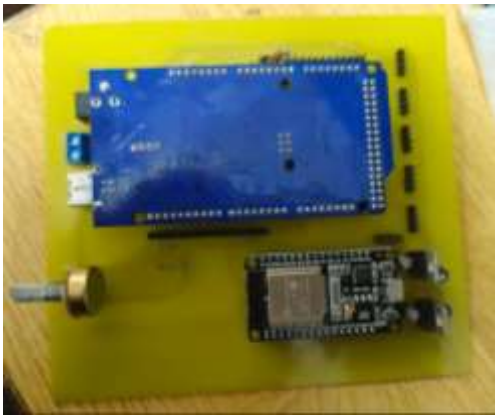


Figure 8.3 PCB Board After Placing Components

CHAPTER 9 SYSTEM TEST PHASE

9.1 FINAL TESTING

Final testing of our project was done this arena which is shown below:

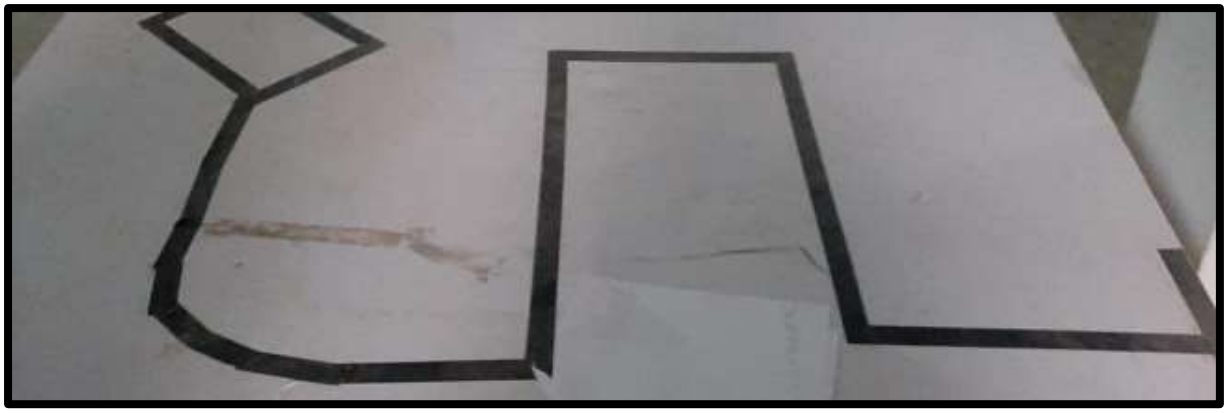


Figure 9.1 ARENA

The arena path contains straight and curved path for testing we have to connect our IR Sensors with 5V power supply motor driver with 15V power supply and infrared sensors with 15V power supply. After connecting all the power supply the robot motor will start rotating. Then we have to place our robot in the start of black line. The 6 infrared sensors attached in-front of robot will detect the black line and give binary output 1 to the motor driver. If the left sensor will detect black line then it will give 1 output to motor driver. Motor driver will rotate right motor and robot will take turn towards left. But if the sensors calibration are not done properly then we have to calibrate them first according to our path.

Steps to do calibration for black and white

1. Mount the IR sensor about 1 to 2 inches above a black surface.
2. Turn the potentiometer fully clockwise.
3. Slowly turn the potentiometer counterclockwise until the signal LED just turns off.
4. Do not change the distance between the IR sensor and the black surface. If you need tochange the distance, then you need to re-calibrate.
5. The signal LED will turn on when the sensor is above a white surface and off when it isabove a black surface.

After doing proper calibration we again tested the robot, and it was following the path perfectly. Thefinal testing was achieved successfully.

9.2 Project Actual Pictures

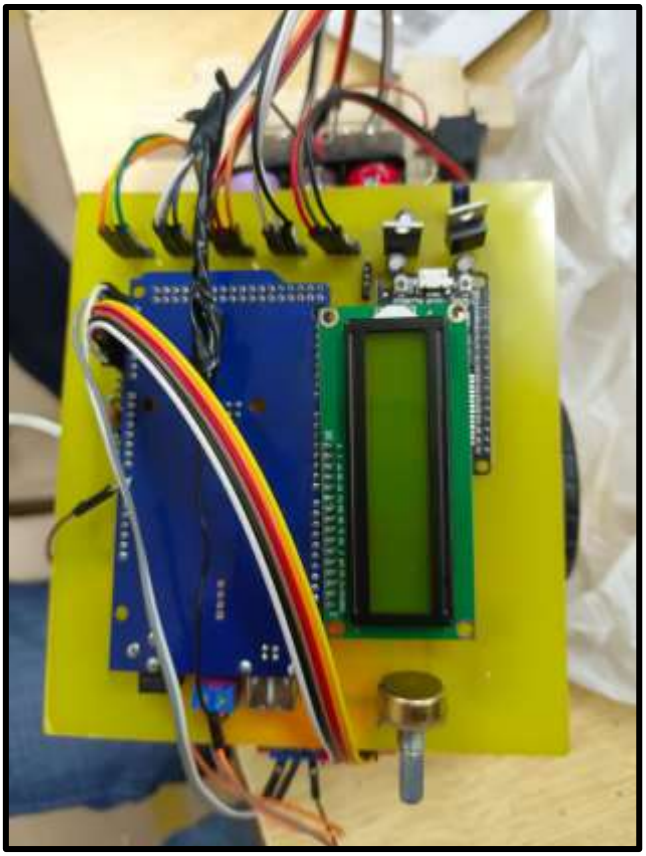
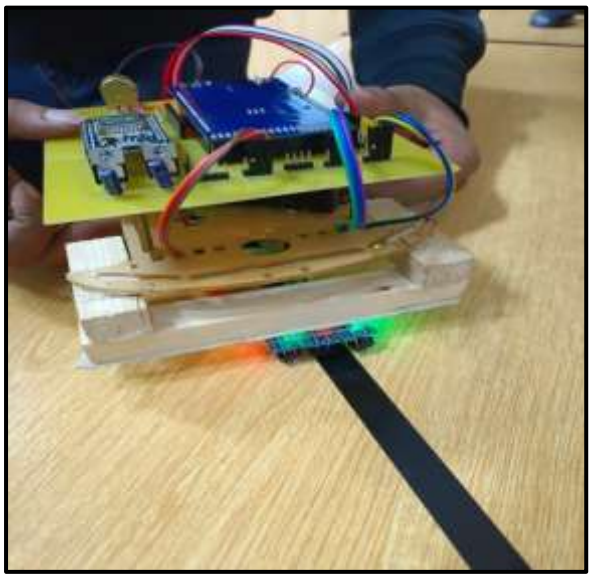
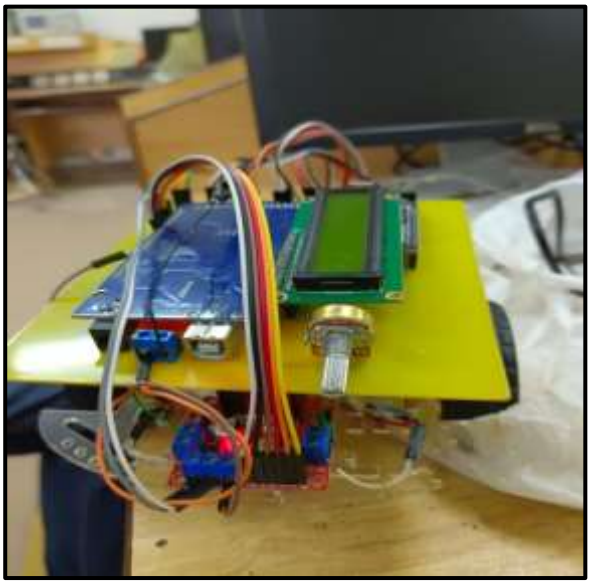


Figure 9.2 Our Project





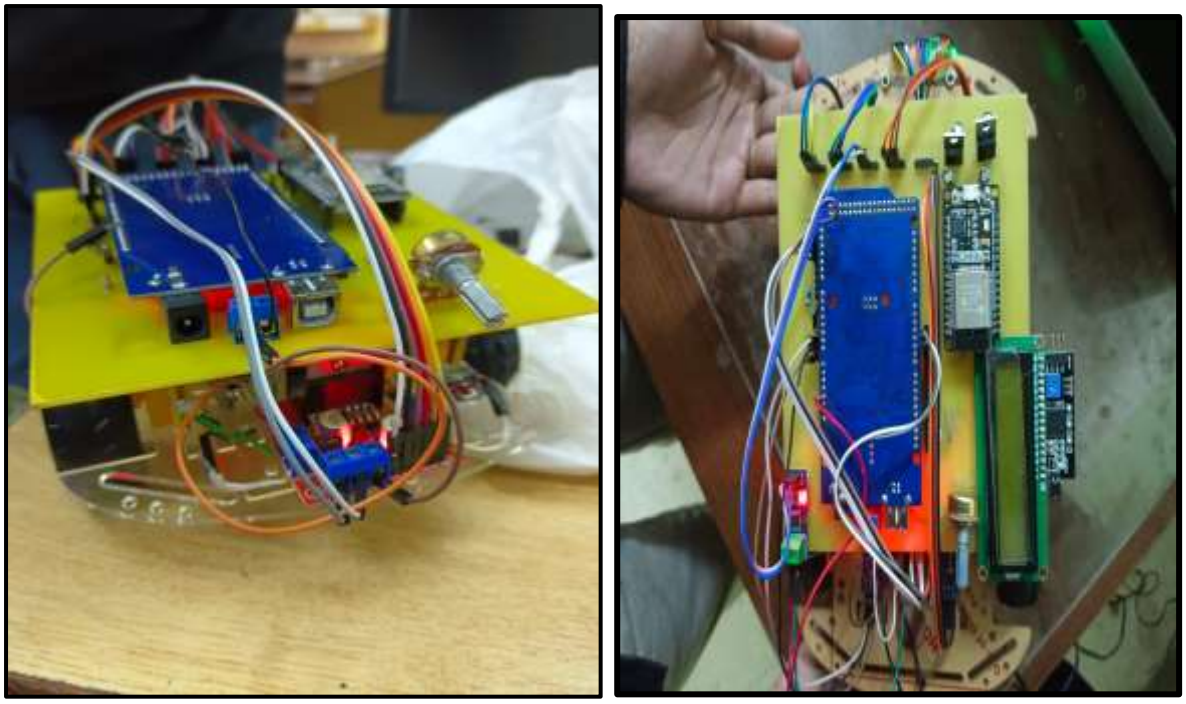


Figure 9.3 READINGS ON LCD DISPLAY OF VOLTAGE, CURRENT, TACHOMETER

CHAPTER 10 PROJECT MANAGEMENT

10.1 FINAL BILL OF MATERIAL LIST

COMPONENT	PRICE
1 chassis (2 tires, 2 motors, bolts, nuts, and base)	1200-1500
Wires (Male to female, male to male, female to female)	400
Soldering Wire	400
Sticker Sheet	150-210
Ultrasonic Sensor	250
ESP 32	950
Sil headers (male & female)	180-200
Voltage Sensor	150
Current Sensor	230-250
Double sided tape	100-150
Ferric chloride	250
5 TCRT5000 IR sensors	5x180
L298N motor driver	350-370
ARDUINO MEGA 2650	4000
PCB	800
Bread board	140-200
12 battery cells	210
LCD Display	650
SD Card module	200
Ultrasonic sensor Holder	70-100
Nail Polish remover (Acetone)	80
	Total= 8850

TABLE 10.1 TABLE OF CONTENT

10.3 WORD COUNT

Member	Tasks	Estimated Word Count
Fahad	Buying components, Writing code, Schematic simulation	High (20-30% of total report words)
Usman	Technical work, Assembling hardware, Debugging & updating code, Report (Integration & Testing)	Moderate-High (15-20% of total report words)
Sabeeh	Schematic simulation & PCB design, Report (PCB Design & Manufacturing)	Moderate-High (15-20% of total report words)

TABLE 10.3 WORD COUNT

10.2 Risk Management

In project management, it is the practice of identifying, evaluating, and preventing or risks to a project that have the potential to impact the desired outcomes. Project managers are typically responsible for overseeing the risk management process throughout the duration of a given project.

During this project, we have gone through several risks like PCB etching was a huge risk in which if etching is not done correctly, we would have to get a new one from store which was a big trouble.Another one is drilling of the PCB holes, soldering.

As our PCB was double layer so PCB alignment was a big game for us. Many components were not available in proteus, so we made them for ourselves.

But in all these risks we have gone through, by the grace of Allah almighty, we have got success in such pressurized environment.

10.3 Challenges Faced during Project

Number of Sensors

Digital IR sensor works if we have only black or white surface area. The simplest case will be if we use 2 IRsensor and can work if we have black path with white surface. But to get accuracy in robot, we are advised to used minimum 5 or 6 sensors, that will increase the complexity of the circuit. So, making conditions, truth tables, simulation on proteus, PCB Layout etc. tickling of these isn't an easy job. So that part is also challenging.

Possibilities in Arena Track

The tract, we will use in project is called arena. The arena is of different type, sometimes we have white line with black surface area, sometimes the opposite. Let consider the black one with white surface area. Here the track may be trajectory, may include u turn, may have left and right turn atsingle place. That's challenging, and we will try to see if what can we do with these conditions.

Components Cost

The Component cost is something that may be so challenging for some students. As Pakistan is apoor country and we a lot of friends, class fellows for which it is not easy to buy some expensive component. We can identify the quality of component with less price available. Also, if somewhere a somebodyfinds the old used component that can used again and have less price, can also be bought.

Design of Robot

How to design our robot in a most sufficient way so that external condition or internal condition on that robot is minimum. There are thousands of designs available in daily life. Adding 4 wheels or 2 wheels to robot, where to place the sensors for best configuration, where to place components and batteries etc. they all should be kept in mind for good design.

Circuit Patching on bread board

For this regard, we first use proteus software, to check if our circuit is working or not. So, making the circuit on proteus may be challenging. Now if we have done it then, implementing circuit on bread board is according to proteus work. But here we got the problem. On proteus, conditions are allideal. And it's easy to simulate, patch things. But it's when it comes to real life.

Making PCB

PCB is also made first in proteus, but here it's not easy to make on proteus. We must check each thing properly for the best design layout. Now if finally, it is made on proteus, so here is the real hardpart. Implementing PCB design practically,

Interfacing different sensors on Microcontrollers

Interfacing different sensors on microcontrollers may cause problem so tickling them was one of the good things we did during the project.

Desired Output

Now if all the things are done quite good. So, the robot got to be work according to desired output required. But in practical world, we can't say anything. Maybe the output is different from what we expected. That may be the challenging part.

10.4 Learning Outcomes

1. In the subject, we study about microcontroller and embedded systems. This project will help us more about understanding and tickling them.
2. This project will let us learn, the implementation of these microcontrollers in real life.
3. If we have given any simple real-life problem, then how to tickle them with microcontrollers and sensors
4. It will enhance our proteus schematic and PCB layout skills.
5. Making logic diagrams, truth table etc. skills will be enhanced.
6. We learn about sensors (IR, ultrasonic, current, voltage sensor), and how it works and how to use and implement them inreal time life. We will learn, how quantity of sensor effects the robot.
7. How to store data in SD card.
8. How to display data on LCD display.
9. How to build something that can work by itself without human interference (Automation).
10. We will understand how to collect data from a given problem and do processes to make therequired and desires product.

Chapter 11- Individual Role, Feedback & Conclusion

11.1 Individual Role in Project

As we have assigned to make a project based on the instructions and information and evaluation of the course in practical life, i.e., Microcontrollers and embedded systems. In this course, we have studied the very basic to the very peak and applicable procedures to solve the problem of logical circuit by creating their logical statements and codes and make them work accordingly.

This is the whole background scenario on which we must make the project on the given problemthat is:

Create and construct a robot which follows the line and track it along a specific path (LINE FOLLOWING ROBOT) using a microcontroller. Also use different sensors like voltage, current, speed sensors and display it on LCD display and save the data on SD card.

There are three procedures/phases of this whole project, as follows.

1 Circuit designing and simulation on Proteus.

2 Simulation and Proceedings of the circuit/logic on breadboard.

3 PCB Designing and Finalizing procedures.

GROUP:

Our group consists of THREE hardworking, friendly and cooperative students.

~ FAHAD MAHMOOD (GROUP LEAD / CODING)

~ SABEEH UR REHMAN(TECHNICAL LEAD/REPORT WRITING)

~ M USMAN (COMPONENTS BUYING/ TECHNICAL LEAD)

Our group lead had assigned each task equally to everyone at every start/beginning of eachphase.

The role that everyone executed to fulfill his own tasks assigned by the group leader are written below. Everyone has done mixture of tasks in every phase. In ever phase, the Main support was given by the group lead but other also had done their best so that the workload was divided equallyon each.

11.2 Tasks Performed by each member

1. Creating truth table on excel file
2. Review and revision of truth table with further iterations and improvement
3. Software programming contribution on ide
4. Proteus file placement contribution
5. Hardware and software interfacing of module
6. PCB trouble shooting
7. Bread board wiring
8. Rawalpindi tour for component
9. Karachi company tour for remaining component
10. G-11 tour for remaining component
11. PCB printing contribution
12. Robot testing on arena

CONCLUSION

✓ The overall expected results were to make the robot follow the line by using the sensors attached to it and to display voltage and current and tachometer sensors readings on LCD. Although the outcome was simple, as mentioned earlier, the project makes us familiar with Arduino, the working mechanism of it and future aspects of it in a simple and understandable way.

✓ Although the thesis project is very little about the robot's use in real world, with the help of guidelines and the abundance of resources the outcome, it could be very beneficial for many people and different sectors of the world depending on the sensors and features required as per necessity.

✓*The goal of this project is to get students interested in and excited about the fields of engineering, mechatronics, and software development as they design, construct, and program an autonomous robot. The guidelines provided are very simple to use and understand thus, making it very easy for the new students to build a foundation in their Robotics learning. Even being a powerful little device, microcontroller would be nothing without its hardware.*

✓*It is just liked a brain that operates a body whereas in this case, the body is made up of many components, including motors, lights, and sensors. The motors provide locomotion, the sensors act as triggers for the motors, and the lights are used primarily as warning devices.*

APPENDIX

Color code for appendix is dark blue.

ESP 32:

Appendix A – ESP32 Pin Lists

A.1. Notes on ESP32 Pin Lists

Table 24: Notes on ESP32 Pin Lists

No.	Description
1	In Table IO_MUX, the boxes highlighted in yellow indicate the GPIO pins that are input-only. Please see the following note for further details.
2	GPIO pins 34–39 are input-only. These pins do not feature an output driver or internal pull-up/pull-down circuitry. The pin names are: SENSOR_VP (GPIO36), SENSOR_CAPP (GPIO37), SENSOR_CAPN (GPIO38), SENSOR_VN (GPIO39), VDET_1 (GPIO34), VDET_2 (GPIO35).
3	The pins are grouped into four power domains: VDDA (analog power supply), VDD3P3_RTC (RTC power supply), VDD3P3_CPU (power supply of digital IOs and CPU cores), VDD_SDIO (power supply of SDIO IOs). VDD_SDIO is the output of the internal SDIO-LDO. The voltage of SDIO-LDO can be configured at 1.8 V or be the same as that of VDD3P3_RTC. The strapping pin and eFuse bits determine the default voltage of the SDIO-LDO. Software can change the voltage of the SDIO-LDO by configuring register bits. For details, please see the column "Power Domain" in Table IO_MUX.
4	The functional pins in the VDD3P3_RTC domain are those with analog functions, including the 32 kHz crystal oscillator, ADC, DAC, and the capacitive touch sensor. Please see columns "Analog Function 1~3" in Table IO_MUX.
5	These VDD3P3_RTC pins support the RTC function, and can work during Deep-sleep. For example, an RTC GPIO can be used for waking up the chip from Deep-sleep.
6	The GPIO pins support up to six digital functions, as shown in columns "Function 1~6" in Table IO_MUX. The function selection registers will be set as "N-1", where N is the function number. Below are some definitions: • SD_* is for signals of the SDIO slave. • HS1_* is for Port 1 signals of the SDIO host. • HS2_* is for Port 2 signals of the SDIO host. • MT* is for signals of the JTAG. • U0* is for signals of the UART0 module. • U1* is for signals of the UART1 module. • U2* is for signals of the UART2 module. • SPI* is for signals of the SPI01 module. • HSPI* is for signals of the SPI2 module. • VSPI* is for signals of the SPI3 module.

No.	Description
7	Each column about digital "Function" is accompanied by a column about "Type". Please see the following explanations for the meanings of "type" with respect to each "function" they are associated with. For each "Function-N", "type" signifies: • I: input only. If a function other than "Function-N" is assigned, the input signal of "Function-N" is still from this pin. • I1: input only. If a function other than "Function-N" is assigned, the input signal of "Function-N" is always "1". • I0: input only. If a function other than "Function-N" is assigned, the input signal of "Function-N" is always "0". • O: output only. • T: high-impedance. • I/O/T: combinations of input, output, and high-impedance according to the function signal. • I1/O/T: combinations of input, output, and high-impedance, according to the function signal. If a function is not selected, the input signal of the function is "1". For example, pin 30 can function as HS1_CMD or SD_CMD, where HS1_CMD is of an "I1/O/T" type. If pin 30 is selected as HS1_CMD, this pin's input and output are controlled by the SDIO host. If pin 30 is not selected as HS1_CMD, the input signal of the SDIO host is always "1".
8	Each digital output pin is associated with its configurable drive strength. Column "Drive Strength" in Table IO_MUX lists the default values. The drive strength of the digital output pins can be configured into one of the following four options: • 0: ~5 mA • 1: ~10 mA • 2: ~20 mA • 3: ~40 mA The default value is 2. The drive strength of the internal pull-up (wpu) and pull-down (wpd) is ~75 μA.
9	Column "At Reset" in Table IO_MUX lists the status of each pin during reset, including input-enable (ie=1), internal pull-up (wpu) and internal pull-down (wpd). During reset, all pins are output-disabled.
10	Column "After Reset" in Table IO_MUX lists the status of each pin immediately after reset, including input-enable (ie=1), internal pull-up (wpu) and internal pull-down (wpd). After reset, each pin is set to "Function 1". The output-enable is controlled by digital Function 1.
11	Table Ethernet_MAC is about the signal mapping inside Ethernet MAC. The Ethernet MAC supports MII and RMII interfaces, and supports both the internal PLL clock and the external clock source. For the MII interface, the Ethernet MAC is with/without the TX_ERR signal. MDC, MDIO, CRS and COL are slow signals, and can be mapped onto any GPIO pin through the GPIO-Matrix.
12	Table GPIO Matrix is for the GPIO-Matrix. The signals of the on-chip functional modules can be mapped onto any GPIO pin. Some signals can be mapped onto a pin by both IO-MUX and GPIO-Matrix, as shown in the column tagged as "Same input signal from IO_MUX core" in Table GPIO Matrix.

Arduino Mega 2560:

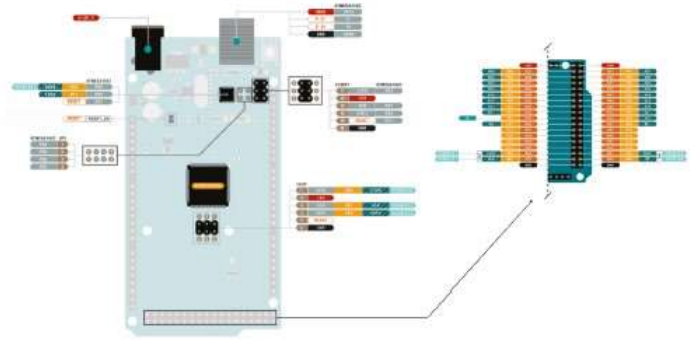
5.1 Analog

Pin	Function	Type	Description
1	NC	NC	Not Connected
2	IOREF	IOREF	Reference for digital logic V - connected to 5V
3	Reset	Reset	Reset
4	+3V3	Power	+3V3 Power Rail
5	+5V	Power	+5V Power Rail
6	GND	Power	Ground
7	GND	Power	Ground
8	VIN	Power	Voltage Input
9	A0	Analog	Analog input 0 /GPIO
10	A1	Analog	Analog input 1 /GPIO
11	A2	Analog	Analog input 2 /GPIO
12	A3	Analog	Analog input 3 /GPIO
13	A4	Analog	Analog input 4 /GPIO
14	A5	Analog	Analog input 5 /GPIO
15	A6	Analog	Analog input 6 /GPIO
16	A7	Analog	Analog input 7 /GPIO
17	A8	Analog	Analog input 8 /GPIO
18	A9	Analog	Analog input 9 /GPIO
19	A10	Analog	Analog input 10 /GPIO
20	A11	Analog	Analog input 11 /GPIO
21	A12	Analog	Analog input 12 /GPIO
22	A13	Analog	Analog input 13 /GPIO
23	A14	Analog	Analog input 14 /GPIO
24	A15	Analog	Analog input 15 /GPIO

5.2 Digital

Pin	Function	Type	Description
1	D21/SCL	Digital Input/I2C	Digital input 21/I2C Dataline
2	D20/SDA	Digital Input/I2C	Digital input 20/I2C Dataline
3	AREF	Digital	Analog Reference Voltage
4	GND	Power	Ground
5	D13	Digital/GPIO	Digital input 13/GPIO
6	D12	Digital/GPIO	Digital input 12/GPIO
7	D11	Digital/GPIO	Digital input 11/GPIO
8	D10	Digital/GPIO	Digital input 10/GPIO
9	D9	Digital/GPIO	Digital input 9/GPIO
10	D8	Digital/GPIO	Digital input 8/GPIO
11	D7	Digital/GPIO	Digital input 7/GPIO
12	D6	Digital/GPIO	Digital input 6/GPIO
13	D5	Digital/GPIO	Digital input 5/GPIO
14	D4	Digital/GPIO	Digital input 4/GPIO

Pin	Function	Type	Description
15	D3	Digital/GPIO	Digital input 3/GPIO
16	D2	Digital/GPIO	Digital input 2/GPIO
17	D1/TX0	Digital/GPIO	Digital input 1 /GPIO
18	D0/Tx1	Digital/GPIO	Digital input 0 /GPIO
19	D14	Digital/GPIO	Digital input 14 /GPIO
20	D15	Digital/GPIO	Digital input 15 /GPIO
21	D16	Digital/GPIO	Digital input 16 /GPIO
22	D17	Digital/GPIO	Digital input 17 /GPIO
23	D18	Digital/GPIO	Digital input 18 /GPIO
24	D19	Digital/GPIO	Digital input 19 /GPIO
25	D20	Digital/GPIO	Digital input 20 /GPIO
26	D21	Digital/GPIO	Digital input 21 /GPIO



Arduino Mega Pinout

L298N:

L298

DUAL FULL-BRIDGE DRIVER

- OPERATING SUPPLY VOLTAGE UP TO 46 V
- TOTAL DC CURRENT UP TO 4 A
- LOW SATURATION VOLTAGE
- OVERTEMPERATURE PROTECTION
- LOGICAL "0" INPUT VOLTAGE UP TO 1.5 V (HIGH NOISE IMMUNITY)

Multiwatt15

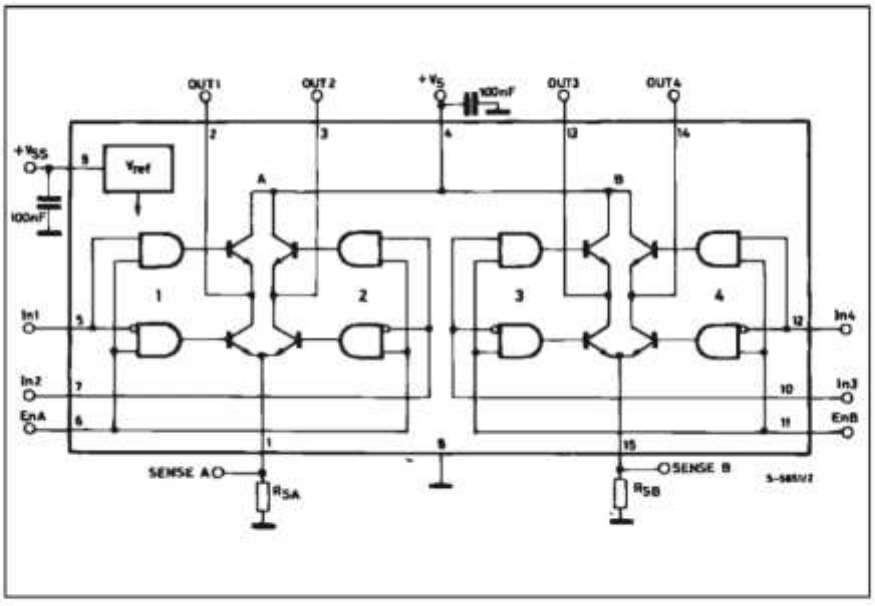
PowerSO20

ORDERING NUMBERS : L298N (Multiwatt Vert.)
L298HN (Multiwatt Horiz.)
L298P (PowerSO20)

DESCRIPTION

The L298 is an integrated monolithic circuit in a 15-lead Multiwatt and PowerSO20 packages. It is a high voltage, high current dual full-bridge driver designed to accept standard TTL logic levels and drive inductive loads such as relays, solenoids, DC and stepping motors. Two enable inputs are provided to enable or disable the device independently of the input signals. The emitters of the lower transistors of each bridge are connected together and the corresponding external terminal can be used for the connection of an external sensing resistor. An additional supply input is provided so that the logic works at a lower voltage.

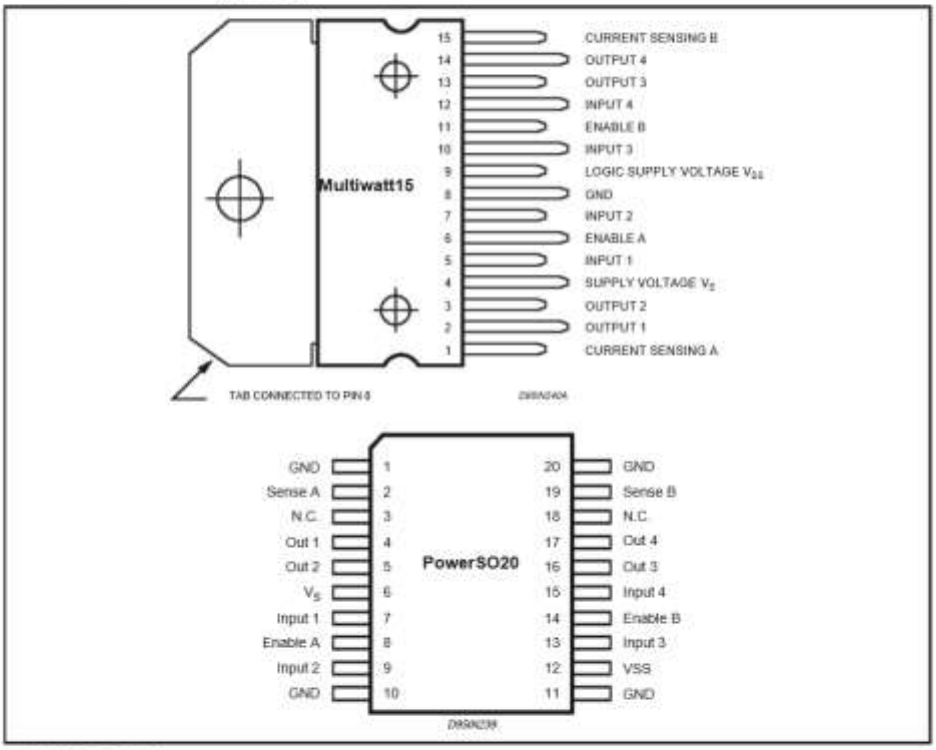
BLOCK DIAGRAM



ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_S	Power Supply	50	V
V_{SS}	Logic Supply Voltage	7	V
V_i, V_{en}	Input and Enable Voltage	–0.3 to 7	V
I_O	Peak Output Current (each Channel) – Non Repetitive (t = 100µs) –Repetitive (80% on –20% off; t_{on} = 10ms) –DC Operation	 3 2.5 2	 A A A
V_{sens}	Sensing Voltage	–1 to 2.3	V
P_{tot}	Total Power Dissipation (T_{case} = 75°C)	25	W
T_{op}	Junction Operating Temperature	–25 to 130	°C
T_{stg}, T_j	Storage and Junction Temperature	–40 to 150	°C

PIN CONNECTIONS (top view)



THERMAL DATA

Symbol	Parameter		PowerSO20	Multiwatt15	Unit
$R_{th\ j\text{-}case}$	Thermal Resistance Junction-case	Max.	–	3	°C/W
$R_{th\ j\text{-}amb}$	Thermal Resistance Junction-ambient	Max.	13 (*)	35	°C/W

(*) Mounted on aluminum substrate

CURRENT SENSOR:

ACS712

Fully Integrated, Hall Effect-Based Linear Current Sensor with 2.1 kVRMS Voltage Isolation and a Low-Resistance Current Conductor

Features and Benefits

- Low-noise analog signal path
- Device bandwidth is set via the new FILTER pin
- 5 µs output rise time in response to step input current
- 50 kHz bandwidth
- Total output error 1.5% at T_A = 25°C, and 4% at –40°C to 85°C
- Small footprint, low-profile SOIC8 package
- 1.2 mΩ internal conductor resistance
- 2.1 kV_{RMS} minimum isolation voltage from pins 1-4 to pins 5-8
- 5.0 V, single supply operation
- 66 to 185 mV/A output sensitivity
- Output voltage proportional to AC or DC currents
- Factory-trimmed for accuracy
- Extremely stable output offset voltage
- Nearly zero magnetic hysteresis
- Ratiometric output from supply voltage

Package: 8 pin SOIC (suffix LC)

Approximate Scale 1:1

Description

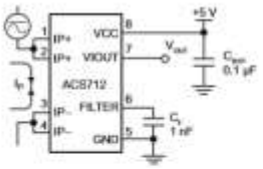
The Allegro® ACS712 provides economical and precise solutions for AC or DC current sensing in industrial, automotive, commercial, and communications systems. The device package allows for easy implementation by the customer. Typical applications include motor control, load detection and management, switched-mode power supplies, and overcurrent fault protection.

The device consists of a precise, low-offset, linear Hall sensor circuit with a copper conduction path located near the surface of the die. Applied current flowing through this copper conduction path generates a magnetic field which is sensed by the integrated Hall IC and converted into a proportional voltage. Device accuracy is optimized through the close proximity of the magnetic signal to the Hall transducer. A precise, proportional voltage is provided by the low-offset, chopper-stabilized BiCMOS Hall IC, which is programmed for accuracy after packaging.

The output of the device has a positive slope (>$V_{IOUT(Q)}$) when an increasing current flows through the primary copper conduction path (from pins 1 and 2, to pins 3 and 4), which is the path used for current sensing. The internal resistance of this conductive path is 1.2 mΩ typical, providing low power

Continued on the next page…

Typical Application



Application 1. The ACS712 outputs an analog signal, V_{OUT}, that varies linearly with the uni- or bi-directional AC or DC primary sensed current, I_P, within the range specified. C_F is recommended for noise management, with values that depend on the application.

Description (continued)

loss. The thickness of the copper conductor allows survival of the device at up to 5× overcurrent conditions. The terminals of the conductive path are electrically isolated from the sensor leads (pins 5 through 8). This allows the ACS712 current sensor to be used in applications requiring electrical isolation without the use of opto-isolators or other costly isolation techniques.

The ACS712 is provided in a small, surface mount SOIC8 package. The leadframe is plated with 100% matte tin, which is compatible with standard lead (Pb) free printed circuit board assembly processes. Internally, the device is Pb-free, except for flip-chip high-temperature Pb-based solder balls, currently exempt from RoHS. The device is fully calibrated prior to shipment from the factory.

Selection Guide

Part Number	Packing*	T_{OP} (°C)	Optimized Range, I_P (A)	Sensitivity, Sens (Typ) (mV/A)
ACS712ELCTR-05B-T	Tape and reel, 3000 pieces/reel	–40 to 85	±5	185
ACS712ELCTR-20A-T	Tape and reel, 3000 pieces/reel	–40 to 85	±20	100
ACS712ELCTR-30A-T	Tape and reel, 3000 pieces/reel	–40 to 85	±30	66

*Contact Allegro for additional packing options

Absolute Maximum Ratings

Characteristic	Symbol	Notes	Rating	Units
Supply Voltage	V_{CC}		8	V
Reverse Supply Voltage	V_{RCC}		–0.1	V
Output Voltage	V_{IOUT}		8	V
Reverse Output Voltage	V_{RIOUT}		–0.1	V
Output Current Source	$I_{IOUT(Source)}$		3	mA
Output Current Sink	$I_{IOUT(Sink)}$		10	mA
Overcurrent Transient Tolerance	I_P	100 total pulses, 250 ms duration each, applied at a rate of 1 pulse every 100 seconds.	60	A
Maximum Transient Sensed Current	I_R(max)	Junction Temperature, $T_J < T_J$(max)	60	A
Nominal Operating Ambient Temperature	T_A	Range E	–40 to 85	°C
Maximum Junction	T_J(max)		165	°C
Storage Temperature	T_{stg}		–65 to 170	°C

TÜV America
Certificate Number:
U8V 06 05 54214 010

Parameter	Specification
Fire and Electric Shock	CAN/CSA-C22.2 No. 60950-1-03 UL 60950-1:2003 EN 60950-1:2001

Allegro MicroSystems, Inc.
115 Northeast Cutoff, Box 15036
Worcester, Massachusetts 01615-0036 (508) 853-5000
www.allegromicro.com

IR SENSOR:

Arduino IR Infrared Obstacle Avoidance Sensor Module

The sensor module adaptable to ambient light, having a pair of infrared emitting and receiving tubes, transmitting tubes emit infrared certain frequency, when the direction of an obstacle is detected (reflection surface), the infrared reflected is received by the reception tube, After a comparator circuit processing, the green light is on, but the signal output interface output digital signal (a low-level signal), you can adjust the detection distance knob potentiometer, the effective distance range of 2 ~ 30cm, the working voltage of 3.3V- 5V. Detection range of the sensor can be obtained by adjusting potentiometer, with little interference, easy to assemble, easy to use features, can be widely used in robot obstacle avoidance, avoidance car, line count, and black and white line tracking and many other occasions.

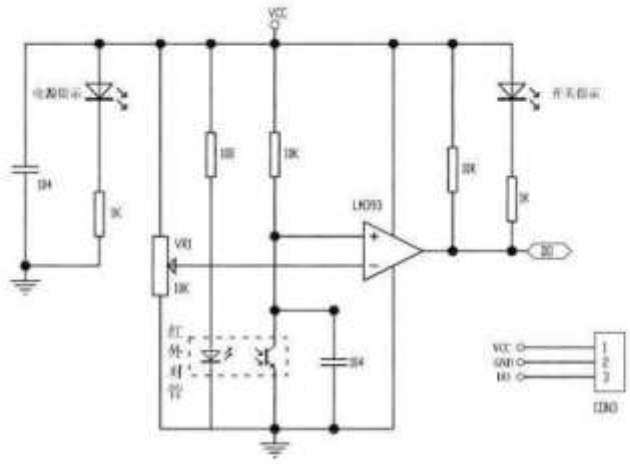
Specification

1. When the module detects an obstacle in front of the signal, the green indicator lights on the board level, while the OUT port sustained low signal output, the module detects the distance 2 ~ 30cm, detection angle 35 °, the distance can detect potential is adjusted clockwise adjustment potentiometer, detects the distance increases; counter clockwise adjustment potentiometer, reducing detection distance.
2. The sensor active infrared reflection detection, target reflectivity and therefore the shape is critical detection distance. Where the minimum detection distance black, white, maximum; small objects away from a small area, a large area from the Grand.
3. The sensor module output port OUT port can be directly connected to the microcontroller IO can also be directly drive a 5V relay; Connection: VCC-VCC; GND-GND; OUT-IO
4. Comparators LM393, stable;

5. The module can be 3-5V DC power supply. When the power is turned on, the red power indicator lights;
6. With the screw holes 3mm, easy fixed installation;
7. Board size: 3.2CM * 1.4CM
8. Each module has been shipped threshold comparator voltage adjusted by potentiometer good, non-special case, do not adjustable potentiometer.

Module Interface Description

1. VCC : 3.3V-5V external voltage (can be directly connected to 5v and 3.3v MCU)
2. GND : GND External
3. OUT : small board digital output interface (0 and 1)



VOLTAGE SENSOR:

Voltage Sensor / Divider Board for ARDUINO developments

Model: Voltage Sensor / 170640

Description: This module is designed based on the principle of resistor divider to reduce the input voltage of the terminal interface by 5 times. The maximum input voltage of the Arduino is 5V. The input voltage of the voltage detection module cannot be greater than 5V × 5 = 25V (if 3.3V is used) System, the input voltage can not be greater than 3.3Vx5 = 16.5V). Because the AVR chip used by Arduino is 10-bit AD, the analog resolution of this module is 0.00489V (5V/1023), so the voltage detection module detects that the input minimum voltage is 0.00489V×5=0.02445V.
parameter:
Voltage input range max: DC0-25V
Voltage detection range: DC0.02445V - 25V
Voltage simulation resolution: 0.00489V
DC input interface: terminal positive terminal is connected to VCC, negative terminal is connected to GND
Output interface: "+" is connected to 5/3.3V, "-" is connected to GND, and "s" is connected to the A0 pin of Arduino.

Reference Code:

```
#include

int val11;
int val2;

void setup()
{
 pinMode(LED1,OUTPUT);
 Serial.begin(9600);
 Serial.println("Emartee.Com");
 Serial.println("Voltage: ");
 Serial.print("V");
}
void loop()
{
 float temp;
 val11=analogRead(0);
 temp=val11/4.092;
 val11=(int)temp;//
 val2=((val11%100)/10);
 Serial.println(val2);
 delay(1000);
}
```

Shenzhen City, Guangdong Science and Technology Co., Ltd. Address: China's Shenzhen Futian District, Shenzhen Huaqiang North Road

ULTRASONIC SENSOR:

Tech Support: services@elecfreaks.com

Ultrasonic Ranging Module HC - SR04

Product features:

Ultrasonic ranging module HC - SR04 provides 2cm - 400cm non-contact measurement function, the ranging accuracy can reach to 3mm. The modules includes ultrasonic transmitters, receiver and control circuit. The basic principle of work:
(1) Using IO trigger for at least 10us high level signal,
(2) The Module automatically sends eight 40 kHz and detect whether there is a pulse signal back.
(3) IF the signal back, through high level , time of high output IO duration is the time from sending ultrasonic to returning.
Test distance = (high level time×velocity of sound (340M/S) / 2,

Wire connecting direct as following:

- 5V Supply
- Trigger Pulse Input
- Echo Pulse Output
- 0V Ground

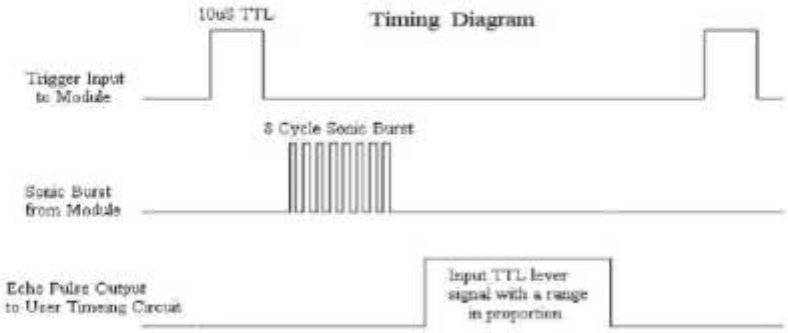
Electric Parameter

Working Voltage	**DC 5 V**
Working Current	**15mA**
Working Frequency	**40Hz**
Max Range	**4m**
Min Range	**2cm**
MeasuringAngle	**15 degree**
Trigger Input Signal	**10uS TTL pulse**
Echo Output Signal	**Input TTL lever signal and the range in proportion**
Dimension	**45*20*15mm**

Vcc Trig Echo GND

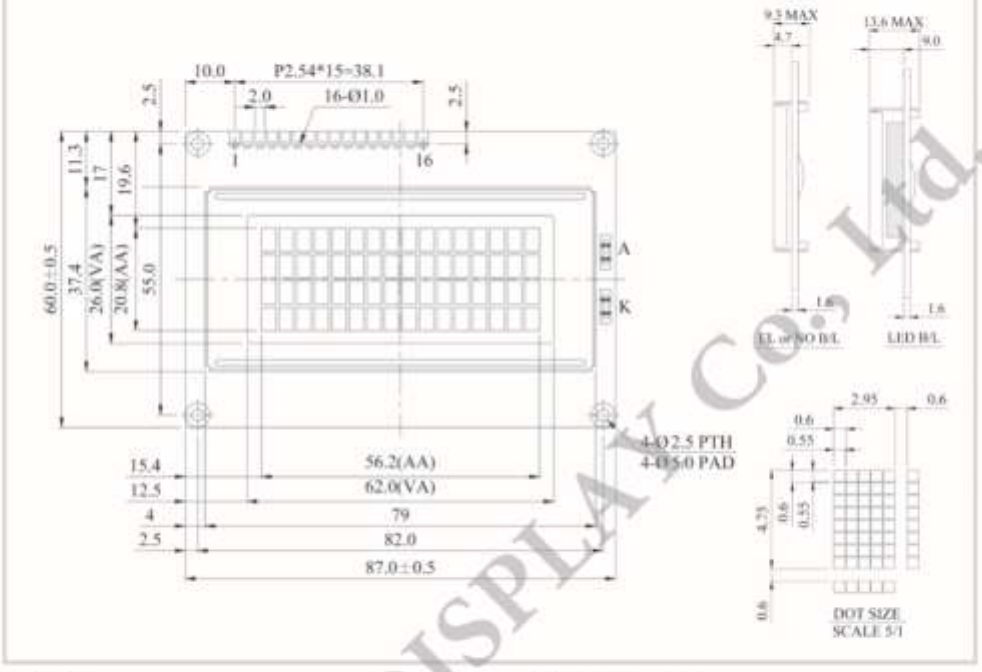
Timing diagram

The Timing diagram is shown below. You only need to supply a short 10uS pulse to the trigger input to start the ranging, and then the module will send out an 8 cycle burst of ultrasound at 40 kHz and raise its echo. The Echo is a distance object that is pulse width and the range in proportion .You can calculate the range through the time interval between sending trigger signal and receiving echo signal. Formula: uS / 58 = centimeters or uS / 148 =inch; or: the range = high level time * velocity (340M/S) / 2; we suggest to use over 60ms measurement cycle, in order to prevent trigger signal to the echo signal.



16X4 LCD:

WH1604A Character 16x4



Feature

1.5x8 dots includes cursor
2.Built-in controller (ST7066 or Equivalent)
3.5V power supply (Also available for 3V)
4.N.V. optional for 3V power supply
5.1/16 duty cycle
6.LED can be driven by PIN1, PIN2, PIN15, PIN16 or A and K
7.Interface : 6800, option SPI/I2C (RW1083 IC)

Pin No.	Symbol	Description
1	V_{SS}	Ground
2	V_{DD}	Power supply for logic
3	V_O	Contrast Adjustment
4	RS	Data/ Instruction select signal
5	R/W	Read/Write select signal
6	E	Enable signal
7	DB0	Data bus line
8	DB1	Data bus line
9	DB2	Data bus line
10	DB3	Data bus line
11	DB4	Data bus line
12	DB5	Data bus line
13	DB6	Data bus line
14	DB7	Data bus line
15	A	Power supply for B/L +
16	K	Power supply for B/L -

Mechanical Data

Item	Standard Value	Unit
Module Dimension	87.0 x 60.0	mm
Viewing Area	62.0 x 26.0	mm
Mounting Hole	82.0 x 55.0	mm
Character Size	2.95 x 4.75	mm

Electrical Characteristics

Item	Symbol	Standard Value typ.	Unit
Input Voltage	VDD	3/5	V
Recommended LCD Driving Voltage for Normal Temp Version module @25°C	VDD-VO	4.35	V

Display Character Address Code

Display position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
DD RAM Address	00	01														0F
DD RAM Address	40	41														4F
DD RAM Address	10	11														1F
DD RAM Address	50	51														5F

42

References:

[1] "IR Sensor : Circuit Diagram, Types Working with Applications." https://www.elprocus.com/infrared-ir-sensor-circuit-and-working/ (accessed Jul. 19, 2021).

[2] "How to Make a Printed Circuit Board (PCB) | PCB | Maker Pro." https://maker.pro/PCB/tutorial/how-to-make-a-printed-circuit-board-PCB (accessed Jul. 19,2021).

[3] P. Q. Anh, T. Duc Chung, T. Tuan, and M. K. A. A. Khan, "Design and Development of anObstacle Avoidance Mobile-controlled Robot," in 2019 IEEE Student Conference on Research and Development, SCOReD 2019, Oct. 2019, pp. 90–94, doi: 10.1109/SCORED.2019.8896296.

https://www.electronicsforu.com/technology-trends/learn-electronics/16x2-lcd-pinout-diagram

https://5.imimg.com/data5/VQ/DC/MY-1833510/lm393-motor-speed-measuring-sensor-module-for-arduino.pdf

https://5.imimg.com/data5/YP/WB/MY-1833510/micro-sd-card-module-for-arduino.pdf

www.amazon.in/MAEnt-Charger-Charging-Circuit-Indicator/dp/B07PJFFZZW

https://components101.com/sensors/ultrasonic-sensor-working-pinout-datasheet

https://components101.com/sites/default/files/component_datasheet/L298N-Motor-Driver-Datasheet.pdf

https://components101.com/sensors/ir-sensor-module

https://www.seeedstudio.com/blog/2020/02/15/acs712-current-sensor-features-how-it-works-arduino-guide/#:~:text=The%20ACS712%20is%20a%20fully,the%20amount%20of%20current%20applied.

https://www.etechnophiles.com/introduction-to-atmega328p-pinout-datasheet-specifications/

https://www.twinschip.com/7805-Voltage-Regulator

http://electrobist.com/product/0-1uf-25v-capacitor/

https://hallroad.org/100uf-25v-polar-capacitor-in-pakistan-en-2.html

https://www.majju.pk/product/22pf-ceramic-disc-capacitor/

https://www.addicore.com/10k-Ohm-1-4W-Metal-Film-Precision-Resistor-p/r5010ks.htm

https://www.twinschip.com/330_Ohm_Resistor

https://www.theengineeringprojects.com/2017/07/introduction-to-1n4007.html

https://www.indiamart.com/proddetail/16-mhz-frequency-crystal-oscillator-

20534374973.html

https://protosupplies.com/product/header-female-1x19-gold-2-pack/

https://sg.cytron.io/p-6x6x1-push-button-2-pins

https://create.arduino.cc/projecthub/rowan07/make-a-simple-led-circuit-ce8308

http://www.baymax-estore.com/product/2-pin-pcb-screw-terminals-block-2-54-mm/

https://electrobes.com/product/12-x-4-5-inch-copper-clad-fr4-fiber-glass-printed-circuit-board-pcb-sheet/

https://learn.sparkfun.com/tutorials/how-to-use-a-breadboard/all

https://www.cityelectronics.pk/shop/fecl3-ferric-chloride-100g/

https://tonomech.com/product/28-pin-dip-ic-socket-base-for-atmega328p-mcu/

https://www.sparkfun.com/products/11026

https://create.arduino.cc/projecthub/search?q=voltage+sensor

THE END