Design and Construction of IOT Based Fire Detection Robot.

Submitted by

SIDDHARTHA GHOSH	**BME-1602009082**
MD. MAHEDI HASAN	**BME-1602009077**
MD. HAKIMUDDIN HERA	**BME-1602009081**
BIVAS KUMAR DAS	**BME-1602009080**

Supervised by

MD. ALI AZAM
Lecturer
Department of Mechanical Engineering

A Project Submitted in Partial Fulfillment of the Requirements for the Degree of Bachelor of Science in Mechanical Engineering

Department of Mechanical Engineering
Sonargaon University
Dhaka-1205, Bangladesh.

February 2020.

DECLARATION OF AUTHORSHIP

We do hereby solemnly declare that, the work presented here in this project report of ***IOT Based Fire Detection Robot*** has been carried out by us and has not been previously submitted to any University/Organization for award of any degree or certificate

We hereby ensure that the works that has been prevented here does not breach any existing copyright.

We further undertake to indemnify the university against any loss or damage arising from breach of the foregoing obligation.

Signed:

SIDDHARTHA GHOSH **BME-1602009082**

……………………….

MD. MAHEDI HASAN **BME-1602009077**

……………………....

MD. HAKIMUDDIN HERA **BME-1602009081**

……………………….

BIVAS KUMAR DAS **BME-1602009080**

……………………….

Supervised by

MD. ALI AZAM
Lecturer
Department of Mechanical Engineering
Sonargaon University.

………………………………………..

ACKNOWLEDGEMENT

Fast of all we are very grateful to almighty Allah, who has blessed us with knowledge and ability to write this report successfully.

This thesis is accomplished under the supervision of MD. Ali Azam Lecturer, Department of Mechanical, Sonargaon University. It is a great pleasure to acknowledge our profound gratitude and respect to our supervisor for this consistent guidance, encouragement, helpful suggestion, constructive criticism and endless patience through the progress of this work. The successful completion of this thesis would not have been possible without his persistent motivation and continuous guidance.

The author are also grateful to Professor Md. Mostofa Hossain, Head of the Department of Mechanical Engineering and all respect teachers of the Mechanical Engineering Department for their co-operation and significant help for completing the thesis work successfully.

Thank you all.

ABSTRACT

"IOT Based Fire Detection Robot" is automatic robots which capable of receiving a set of command instructions in the form internet service and performs the necessary actions. We will be using an internet module node MCU at the receiver & transmitting data. The robot itself and send the data using internet service as per the required actions. Although, the appearance and capabilities of robot vary vastly, all robots share the feature of a mechanical, movable structure under some form of control. The control of robot involves three distant phases: perception, processing, action. Generally, the preceptors are sensors mounted on the robot, processing is done by the on board microcontroller and the task is performed using motors or with some other actuators.

TABLE OF CONTENTS

List of Figures

List of Tables

CHAPTER 1
INTRODUCTION

1.1 OVERVIEW

The number of embedded devices that can interact with environment and already connected to internet is astounding, and it is estimated that the number reaches 50 billion by 2020 [1]. The growth of such interacting objects achieved this staggering pace with the development of microcontroller based easy-to-use designed system which are replacing old systems designed with complicated electronic circuits. Thus, Internet of Things (IOT in short) is gradually taking over our daily needs.

IOT stands for Internet of things for internet Communications. IOT projects are based on one of the emerging technology of the century. It deals with design of a stand-alone embedded system that can monitor and control several devices remotely. Generally sending and receiving data is the concept followed in embedded domain. The system has two parts, namely; hardware and software. The hardware architecture consists of a stand-alone embedded system using an 8-bit micro-controller, several type of interface and driver circuits. [2]

The system software driver is developed using an interactive C programming language. The mobile unit which is dedicated at the robot is interfaced with an intellectual device called Micro controller so that it takes the responsibility of reading the received commands in the form of apps from the mobile unit and perform the corresponding predefined tasks. The micro controller is also interfaced with few DC gear motors in order to move the robot in different directions. The ON and OFF of the DC motors depends on the direction it has to move which is the complete responsibility of the controller to take those intelligent decisions.

The aim of this project is to function the robot in unstructured and dynamic environments and perform multiple tasks like sensing environmental changes and giving countermeasures, observing the surrounding activities through ultrasonic sensor.

1.2 Overall aim of the project (Enumerated)

The objective of our project is to function the robot in unstructured and dynamic environments.

1. To train the robot to work automatically and synthesize motion to achieve a given task by the user.
2. To design & implementation of android controlled robot through by internet.
3. To allow android app based monitoring from a remote location.
4. To examine the changes in the surroundings that come through Robot using ultrasonic sensor.
5. To detect interrupting in all the directions of the robot (Situational awareness).
6. To detect gas leak and fire, sensing surroundings, defense purpose.
7. To detect fire and give alert signal as well as it would be able to activate any fire defense system at that area.

1.3 Motivation

Detection of fire in homes is necessary to avoid destruction of property due to fire accidents both natural and induced. Detection of fire can prove to be very important as it could mean the difference between life and death. Fires can occur from anywhere and at any point of time, hence the presence of Fire Alarm System helps in keeping your family safe.

Fire is very deadly and it leads to loss of human life and property. Fire detection systems are necessary to reduce the destruction of personal belongings and caused by fire both man made and induced.

One of the most destructive properties of fire is that it spreads exponentially and with the right medium can spread uncontrollably. This is why timely detection of fire is necessary for avoiding a fire hazard.

Internet of Things is a collection of sensor, actuators, software, electronics embedded with home appliances, physical devices and vehicles which connect with each other to connect and exchange date which helps in increasing the efficiency of everyday appliances using computer based systems. Not only does it help in improving the efficiency of a device but also has economic benefits. Iot is just another way to make everyday life easier for humans by developing smart devices.

CHAPTER 2

SYSTEM ARCHITECTURE

2.1 Block Diagram:

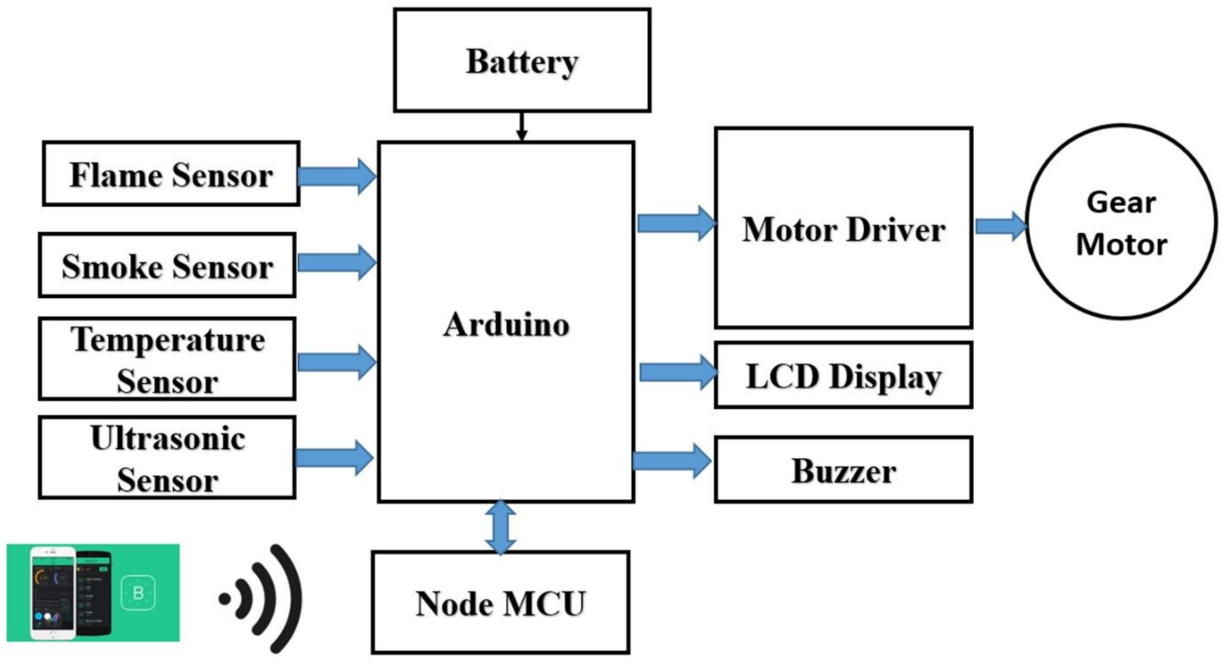


Figure 2.1: Block diagram of IOT Based Fire Detection Robot.

2.2 Circuit Diagram:

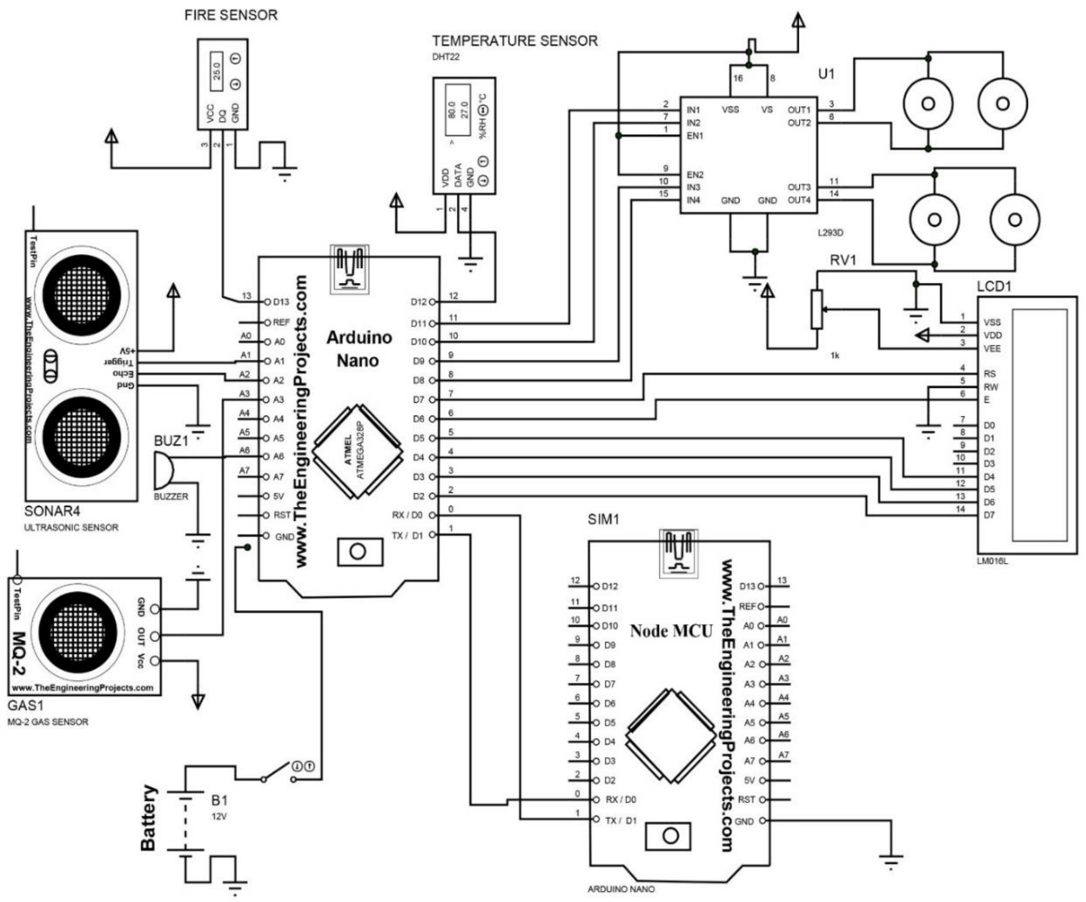


Figure 2.2: Circuit Diagram of IOT Based Fire Detection Robot

2.3 Working Principle:

In this project IOT controlled & self-obstacle avoiding robot. In our project we controlling our robot via mobile apps based Internet. We can send a signal to the Arduino through internet to start the robot car. After it turn ON it automatically move around and detect any object through Ultrasonic sensor and then it move to other ware. The robot is powered by battery. We implementing few sensors, Temperature sensor this sensor will detect the heat of the environment and we give an algorithm if the temperature gets increase from our normal temperature then it will warn by buzzer and also send a notification with the details of temperature. Also we implementing Fire sensor and Smoke sensor if this robot detects any fire or smoke environment it will instant alarm buzzer and send the condition of there. There is an LCD display on car so we can also see the condition of that car and other warning.

2.4 Methodology

This section describes the hardware connections in the project. Detailed description and diagrams are presented where required. The Connections can be discussed under the following headings.

- Connecting the temperature sensor DHT 11 to Arduino.
- Connecting the gas sensor MQ-2 to Arduino.
- Connecting the fire sensor LM-393 to Arduino.
- Connecting ultrasonic sensor HC-SR04 Arduino.
- Connecting the LCD 16*2 to Arduino.
- Connecting motor driver IC L293D to Arduino.
- Connecting DC gear motor to IC to L293D.
- Connecting the WiFi module Node MCU to Arduino.
- Construction of 4-wheel Chassis and connect with Arduino.
- Connecting Buzzer to Arduino.
- Connecting the Relay Arduino.
- Connecting voltage regulator 7805 and 7806 to Arduino.

Although the final connection description and as well as diagrams it is assumed that each and every component is connected to the Arduino Nano's original pins.

2.4.1 Connecting the temperature sensor DHT 11 to Arduino.

At first we connect the DHT11 sensor with Arduino. The signal pin of DHT11 is connected to an Arduino digital pin, the Vcc pin to a 5V power Source and Gnd to the Gnd pin of Arduino. [3][4]

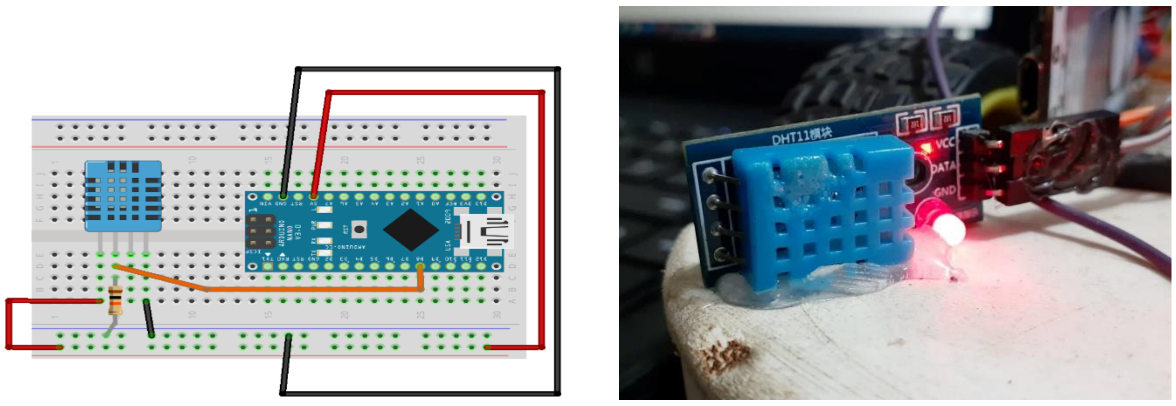


Figure 2.3: Connecting the DHT11

2.4.2 Connecting the gas sensor MQ-2 to Arduino.

Then we connect the MQ-2 Sensor to the Arduino. The Vcc pin again is connected to the 5V power pin and the Gnd pin to Arduino's Gnd pin. The Signal is analog. So we'll connect this pin to one of the Analog pins of Arduino like A0.

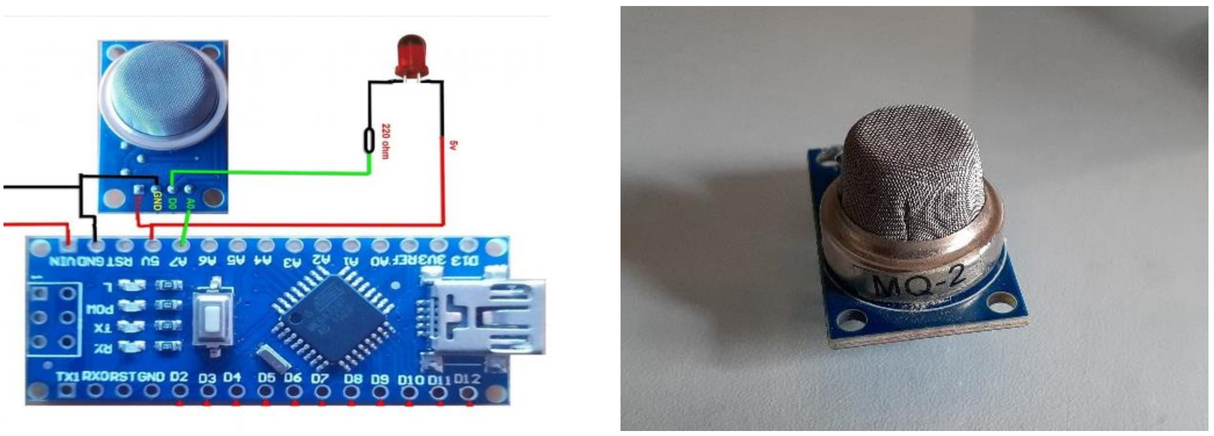

220 ohm
5v
GND
D0
A0
VIN GND RST 5V A7 A6 A5 A4 A3 A2 A1 A0 REF 3V3 D13
TX1 RX0 RST GND D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12
MQ-2

Figure 2.4 – Connecting MQ2 Gas Sensor.

2.4.3 Connecting ultrasonic sensor HC-SR04 Arduino.

The HC-SR04 ultrasonic sensor uses SONAR to determine the distance of an object just like the bats do.

Code to Note

The Ultrasonic sensor has four terminals - +5V, Trigger, Echo, and GND connected as follows −

1. Connect the +5V pin to +5V on your Arduino board.
2. Connect Trigger to digital pin 7 on your Arduino board.
3. Connect Echo to digital pin 6 on your Arduino board.
4. Connect GND with GND on Arduino.

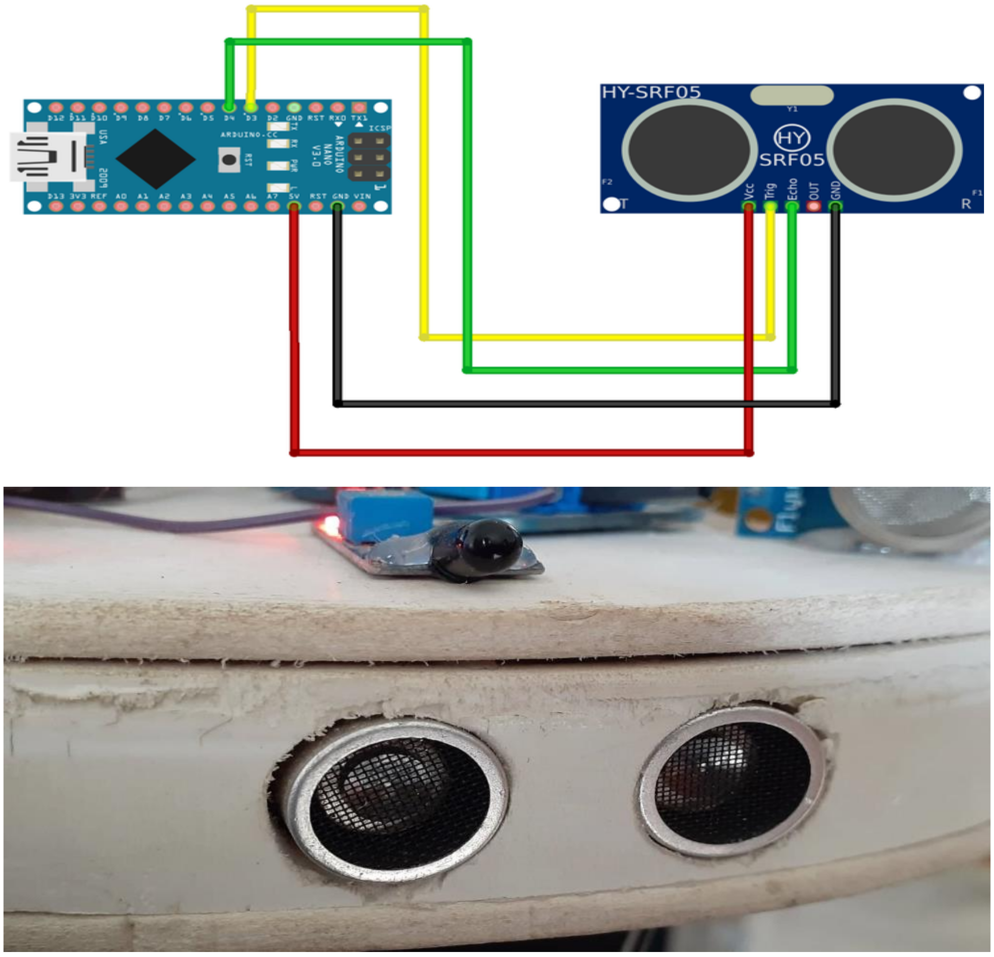


Figure 2.5 – Connecting Ultrasonic Sensor HC-SR04 to Arduino

2.4.4 Connecting the fire sensor LM-393 to Arduino.

- A Flame Sensor is a device that can be used to detect presence of a fire source or any other bright light sources. There are several ways to implement a Flame Sensor but the module used in this project is an Infrared Radiation Sensitive Sensor.

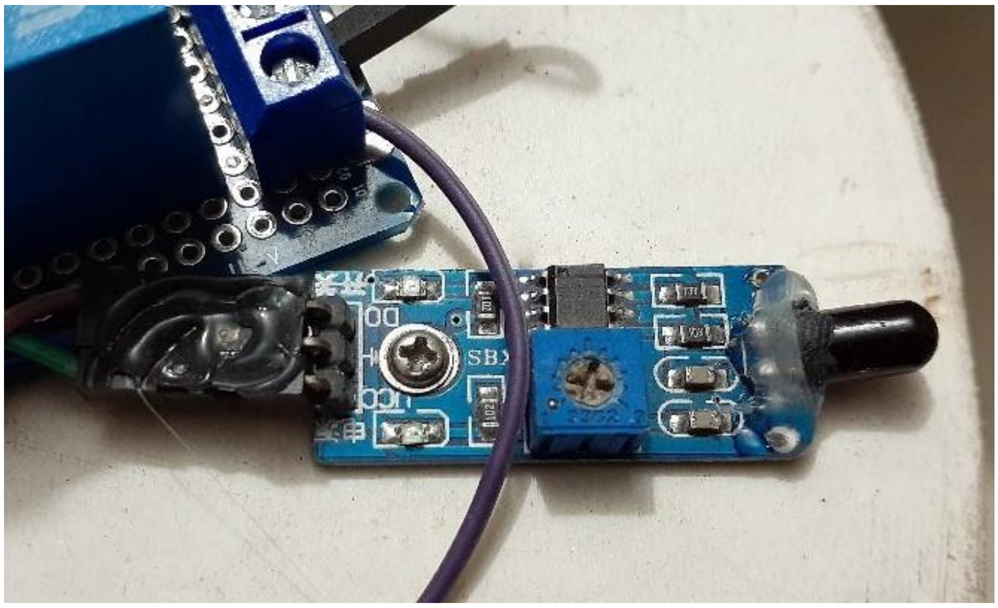


Figure 2.6 – Connecting the fire sensor LM-393 to Arduino.

- Flame Sensor has three pins (some may have four pins): VCC, GND and DO. Connect VCC and GND to +5V and GND of the power supply (can be connected to Arduino's +5V). the DO (short for Digital Output) is connected to Digital I/O Pin 11 of Arduino.

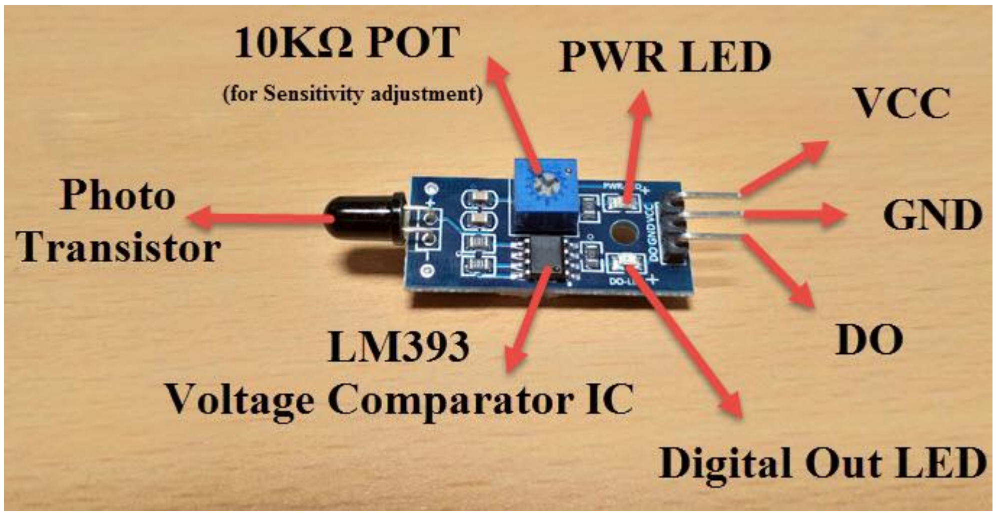


Figure 2.7 – Flame Sensor LM393 pin overview.

2.4.5 Connecting LCD Display to Arduino.

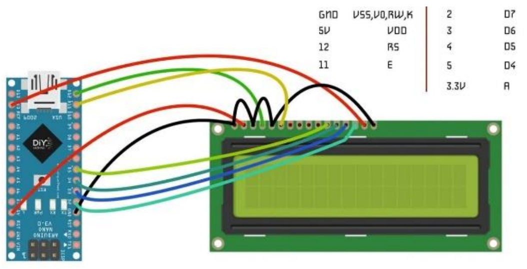
Hardware Required

- Arduino Uno
- 16x2 LCD Screen (compatible with Hitachi HD44780 driver)
- pin headers to solder to the LCD display pins
- 10k ohm potentiometer
- 220-ohm resistor
- hook-up wires
- breadboard

Before wiring the LCD screen to Arduino board we have to solder a pin header strip to the 16 pin count connector of the LCD screen. To wire our LCD screen to our board, we connect the following pins:

- LCD RS pin to digital pin 12
- LCD Enable pin to digital pin 11
- LCD D4 pin to digital pin 5
- LCD D5 pin to digital pin 4
- LCD D6 pin to digital pin 3
- LCD D7 pin to digital pin 2

Additionally, we wire a 10k pot to +5V and GND, with it's wiper (output) to LCD screens VO pin (pin3). A 330 ohm resistor is used to power the backlight of the display, usually on pin 15 and 16 of the LCD connector.[15]



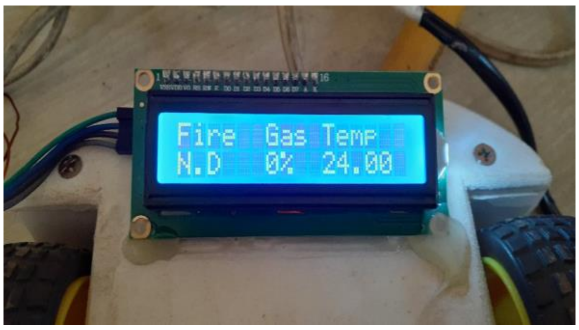


Figure 2.8 – Diagram of LCD Connection with Arduino

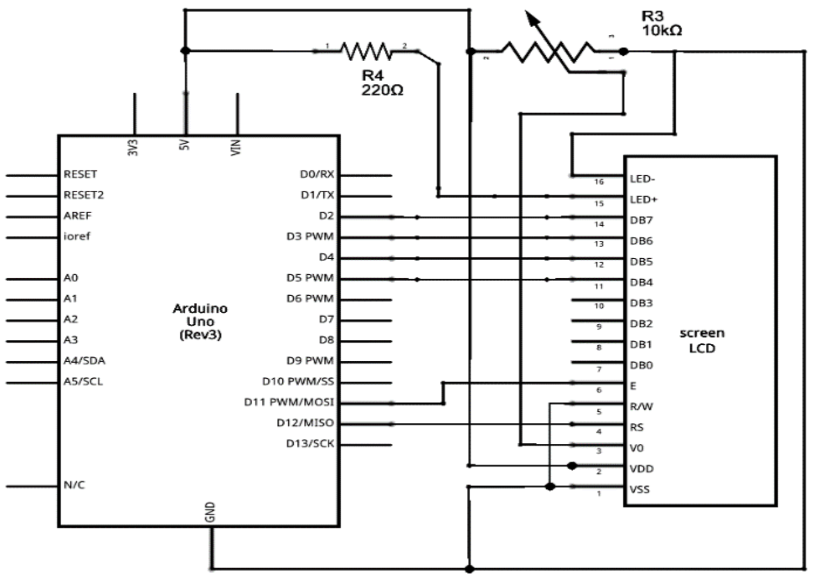


Figure 2.9 – Schematics of LCD Connection

2.4.6 Connecting Motor controller to Arduino

A motor driver is a small Current Amplifier whose function is to take a low-current control signal and then turn it into a higher-current signal that can drive a motor. The L293D is a typical Motor Driver which can drive 2 DC motors simultaneously.

Motor Driver ICs are primarily used in autonomous robotics only. Also most microprocessors operate at low voltages and require a small amount of current to operate while the motors require a relatively higher voltages and current. Thus current cannot be supplied to the motors from the microprocessor. This is the primary need for the motor driver IC.

The Connections to Arduino Nano

- All the Enable pins are connected to the 5V pin on the Arduino.
- Ground Pins should be shorted and connected to the Gnd pin on the Arduino.
- +ve side of the battery should be connected to pin 8 of the IC and the -ve to the Gnd of Arduino.

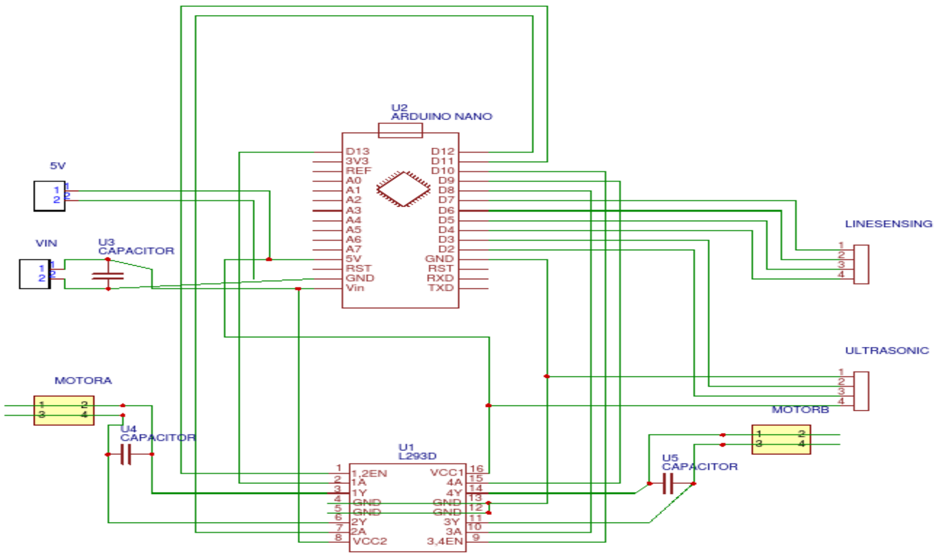


Figure 2.10 - Schematics of Arduino to L293D Connection

To set up the connection we need.

- L293D Motor Driver.
- Arduino NANO.
- DC motors.
- Bread Board.
- 9V Battery.
- Jumper Wires.
- Wheels.

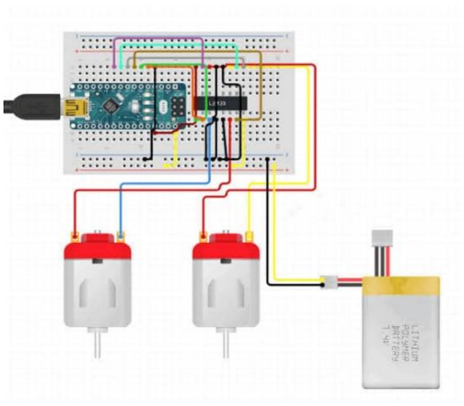
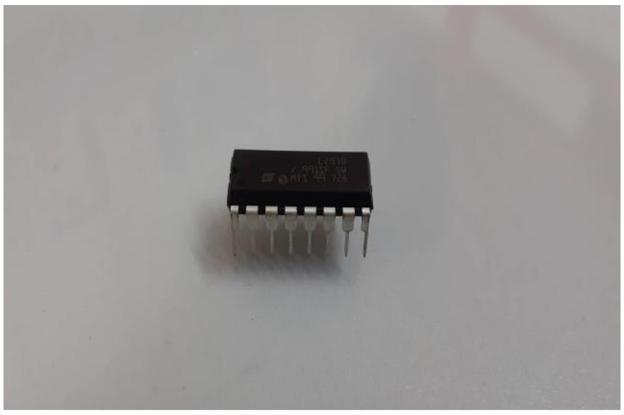



Figure 2.11- Arduino to L293D Connection

2.4.7 Connecting DC gear motor to IC L293D.

- Arduino DC motor connect to L293D:

  Place the L293D in the center of the breadboard, with half of the pins on either side of the breadboard. Connect 5V to Enable 1, Vss , and Vs on the L293D. Connect digital output pins (we're using 6 and 7) to input 1 and input 2 on the L293D. Connect your Arduino's GND to both GND pins on the same side of the L293D.

- Arduino Motor Control Setup:

  1. Connect 5V and ground of the IC to 5V and ground of Arduino.
  2. Connect the motor to pins 2 and 3 of the IC.
  3. Connect IN1 of the IC to pin 8 of Arduino.
  4. Connect IN2 of the IC to pin 9 of Arduino.
  5. Connect EN1 of IC to pin 2 of Arduino.
  6. Connect SENS A pin of IC to the ground.

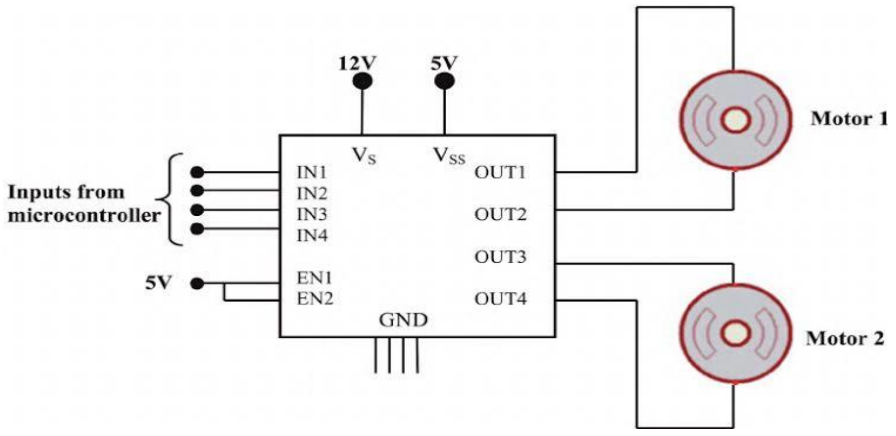


Figure 2.12: DC gear motor to IC L293D

2.4.8 Connecting the NodeMCU to Arduino.

Hardware Required:

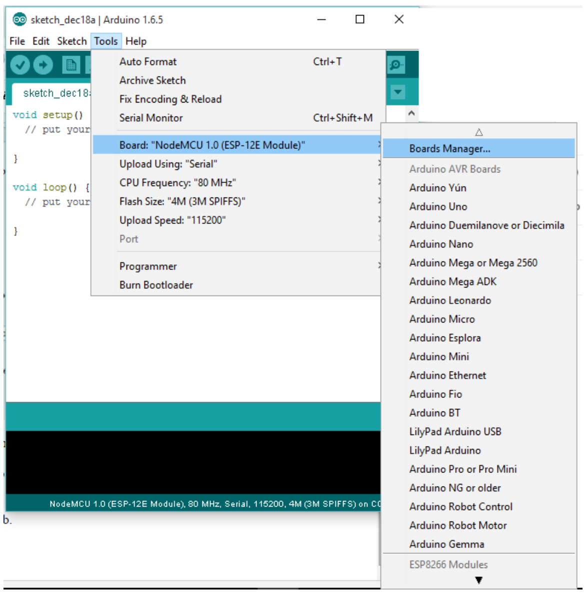
- Arduino Nano
- Node MCU

Connect Arduino IDE to Node MCU.

Step 1: Connect your NodeMCU to your computer. You need a USB micro B cable to connect the board.

Step 2: Open Arduino IDE. You need to have at least Arduino IDE version 1.6.4 to proceed with this.

Step 3: Make a LED blink using NodeMCU.



```
void setup()
  // put your

}

void loop() {
  // put your

}
```

Board: "NodeMCU 1.0 (ESP-12E Module)"
Upload Using: "Serial"
CPU Frequency: "80 MHz"
Flash Size: "4M (3M SPIFFS)"
Upload Speed: "115200"

NodeMCU 1.0 (ESP-12E Module), 80 MHz, Serial, 115200, 4M (3M SPIFFS) on CO

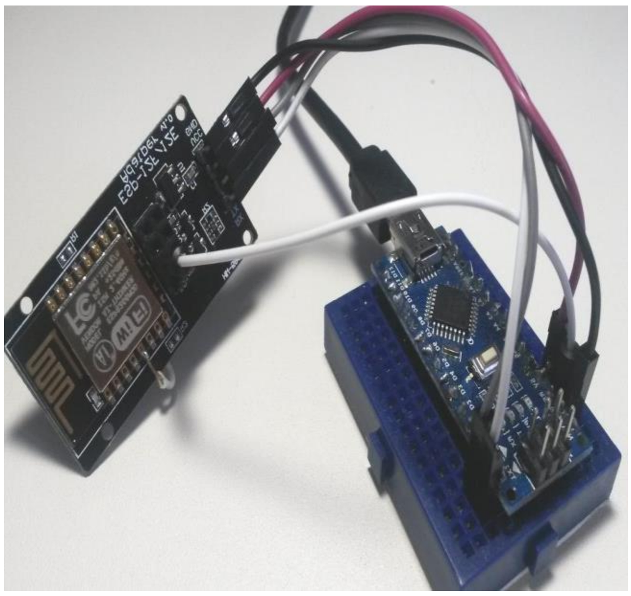


Figure 2.13- Connecting the NodeMCU to Arduino

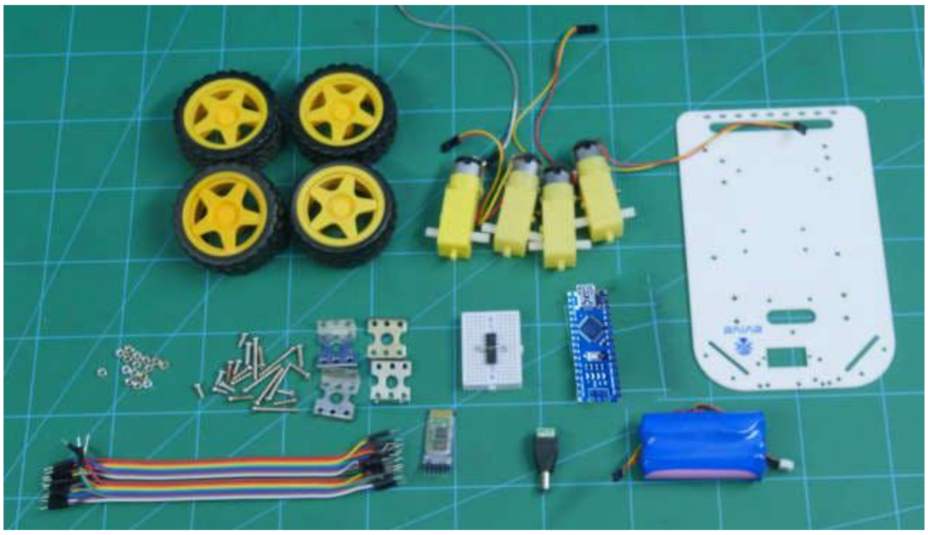
2.4.9 Construction of 4-wheel Chassis and connect with Arduino

Hardware Required:

- Arduino nano
- Chassis
- Motor Mounts
- DC Motors
- Wheel
- Motor Driver
- DC Terminal Block
- Battery
- Jumper Cables

The connection is made as following figure:

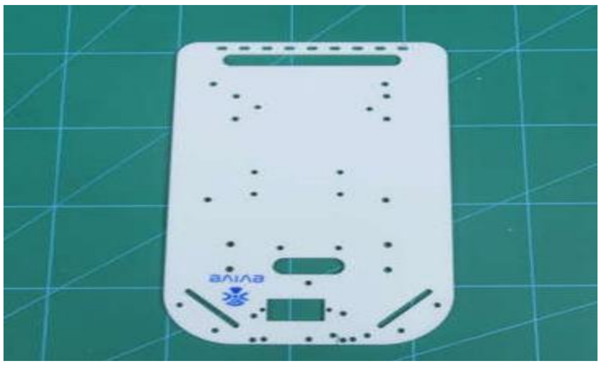
Step 1: Things You'll Need.



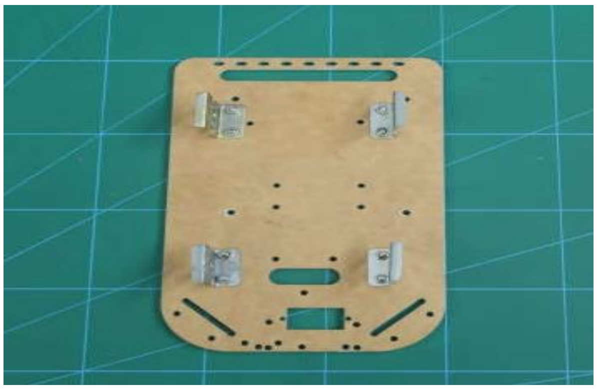
Step 2: Making of the Body

Firstly, we will start making the body of the 4 Wheel Robot.

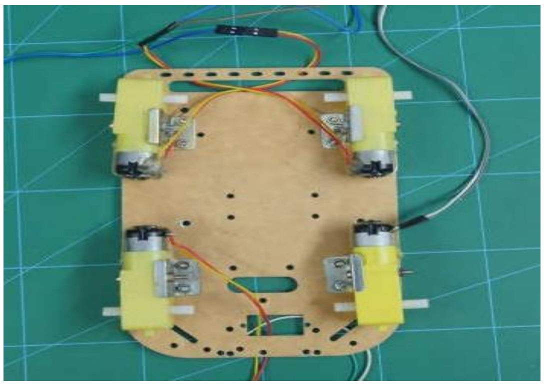
Take the chassis and turn it upside down.



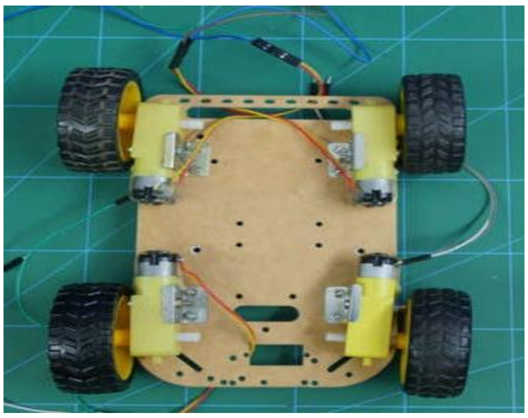
Step 3: Onto this chassis mount 4 motor mounts using M3 Bolts and nuts.



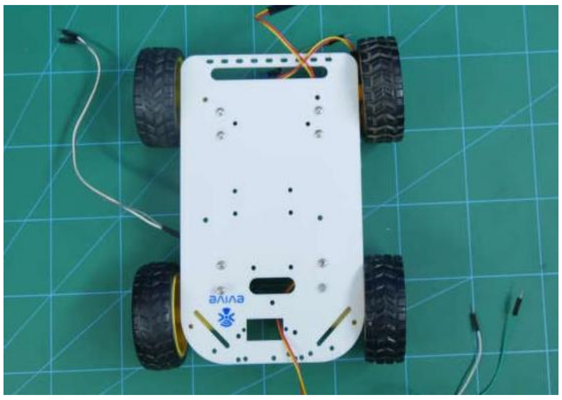
Step 4: Fix the DC Motors and fix them on the motor mounts using M2 Nuts and Bolts.



Step 5: Attach wheels on each shaft of the DC Motor.

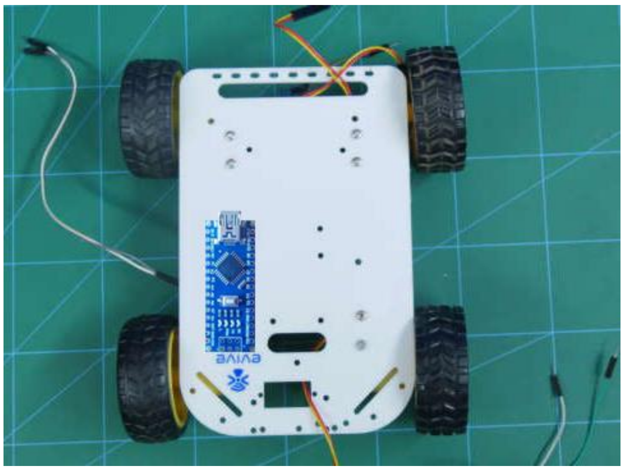


Step 6: Flip the assembly, thus your body is created.



Step 7: Adding the Brain

The microcontroller that we are going to use is Arduino Nano.



Step 8: Making Connections and Using Motor Driver.

Make the connection as shown in the below figure.

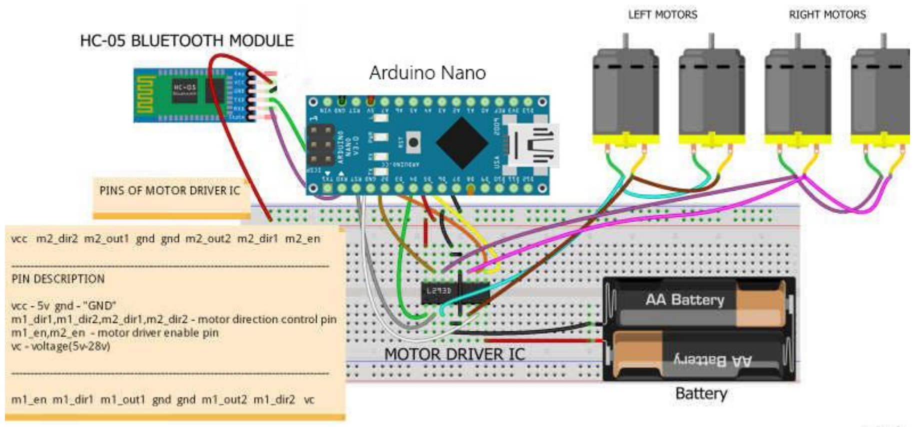

HC-05 BLUETOOTH MODULE
Arduino Nano
LEFT MOTORS
RIGHT MOTORS
PINS OF MOTOR DRIVER IC
vcc m2_dir2 m2_out1 gnd gnd m2_out2 m2_dir1 m2_en
PIN DESCRIPTION
vcc - 5v gnd - "GND"
m1_dir1,m1_dir2,m2_dir1,m2_dir2 - motor direction control pin
m1_en,m2_en - motor driver enable pin
vc - voltage(5v-28v)
m1_en m1_dir1 m1_out1 gnd gnd m1_out2 m1_dir2 vc
MOTOR DRIVER IC
AA Battery
AA Battery
Battery

Figure 2.14: Making Connections and Using Motor Driver

We are going to make use of the motor drivers because Arduino Uno does not provide sufficient power to run 4 Motors. Thus, we will be adding the motor driver, we will be able to give the robot needed energy.

The left two motors are connected in parallel. Similarly, the right two motors are too connected in parallel.

The connections are made as follows:

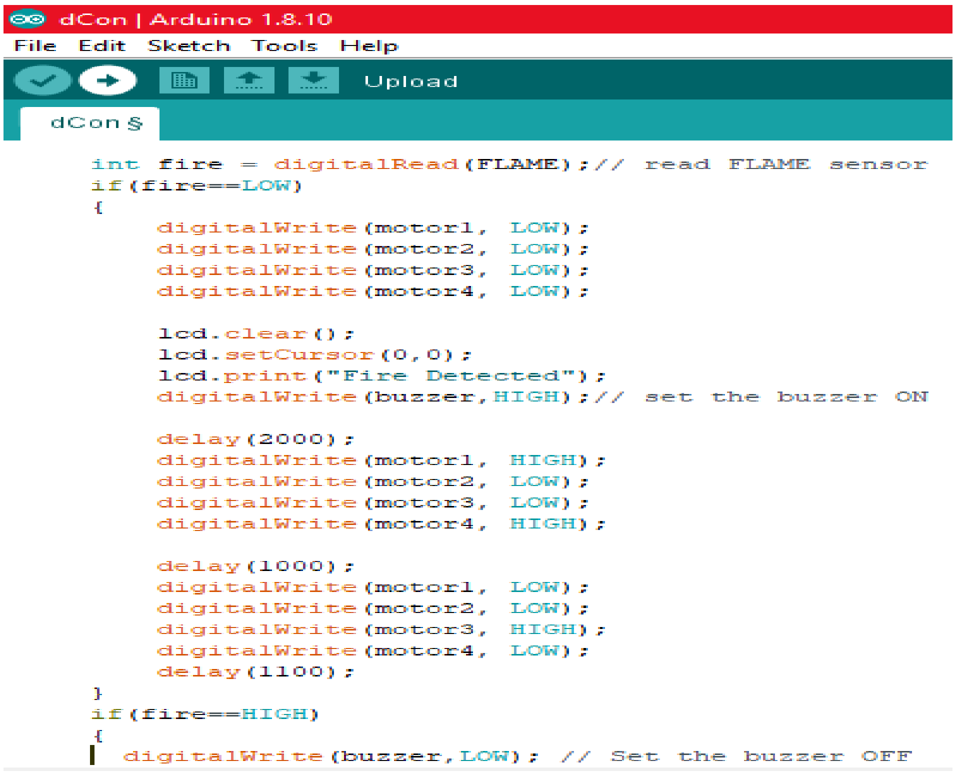
- Enable Pins – Digital Pin 10 and 11
- VCC – Arduino 5V
- m1_dir1, m1_dir2, m2_dir1, m2_dir2 – Digital Pin 4, 5, 6, and 7

- VC – External Battery
- GND – GND of Arduino and Battery Make sure we connect all the GND wires together.

Step 9: Adding the Battery.

We will need to add an extra power source to our robot. Also, to connect Arduino Uno with the battery we are going to use DC Terminal Block or DC Jack.

Step 10: Arduino Code.



```
int fire = digitalRead(FLAME);// read FLAME sensor
if(fire==LOW)
{
     digitalWrite(motor1, LOW);
     digitalWrite(motor2, LOW);
     digitalWrite(motor3, LOW);
     digitalWrite(motor4, LOW);

     lcd.clear();
     lcd.setCursor(0,0);
     lcd.print("Fire Detected");
     digitalWrite(buzzer,HIGH);// set the buzzer ON

     delay(2000);
     digitalWrite(motor1, HIGH);
     digitalWrite(motor2, LOW);
     digitalWrite(motor3, LOW);
     digitalWrite(motor4, HIGH);

     delay(1000);
     digitalWrite(motor1, LOW);
     digitalWrite(motor2, LOW);
     digitalWrite(motor3, HIGH);
     digitalWrite(motor4, LOW);
     delay(1100);
}
if(fire==HIGH)
{
  digitalWrite(buzzer,LOW); // Set the buzzer OFF
```

2.4.8 Connecting Buzzer to Arduino.

The Buzzer and LEDs are placed on the breadboard and all of their negative terminals are connected to the Arduino Gnd pin. The respective positive pins are connected to the individual digital pins on the Arduino nano.

This way when the digital pins are in High state a +5V voltage is applied and a suitable current is supplied to the components which can power them.

The following figures clarifies the connections:

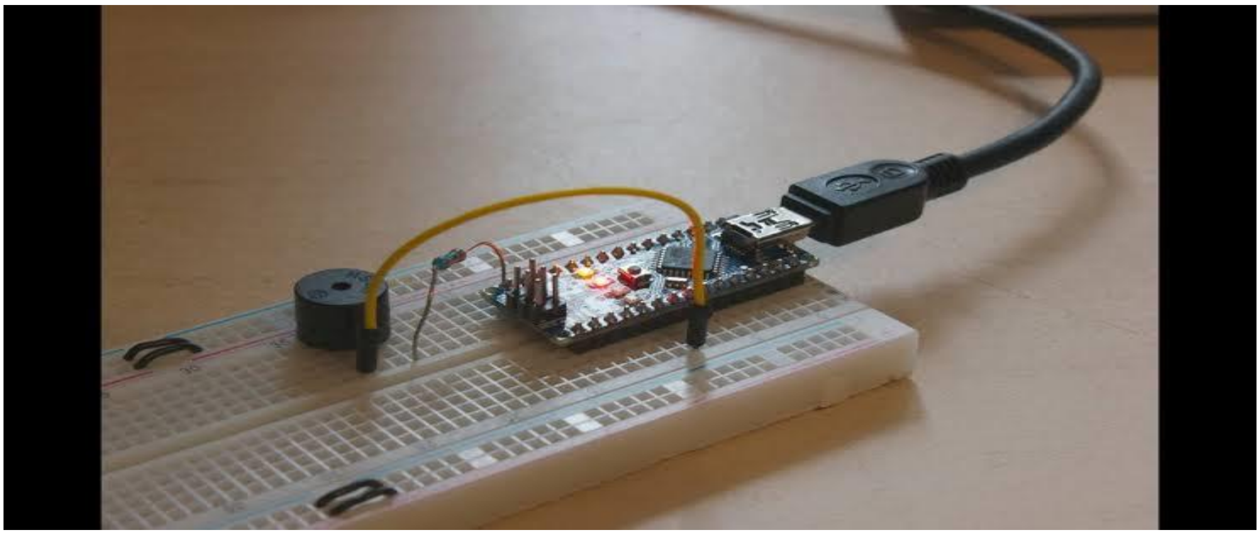


Figure 2.15 – Connecting Buzzer

2.4.9 Connecting the Relay to Arduino.

Instead, you use a low-voltage control signal from the Arduino to control a relay, which is capable of handling and switching high-voltage or high-power circuits. A relay consists of an electromagnet that, when energized, causes a switch to close or open. [5]

The connections between the relay module and the Arduino:

1) GND: goes to ground.
2) IN1: controls the first relay (it will be connected to an Arduino digital pin)
3) IN2: controls the second relay (it should be connected to an Arduino digital pin if you are using this second relay.
4) VCC: goes to 5V.

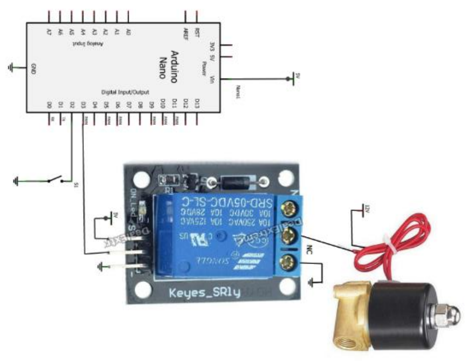


Figure 2.16: Connecting the Relay to Arduino

2.4.10 Connecting voltage regulator 7805 and 7806 to Arduino.

- 78xx voltage regulator work:

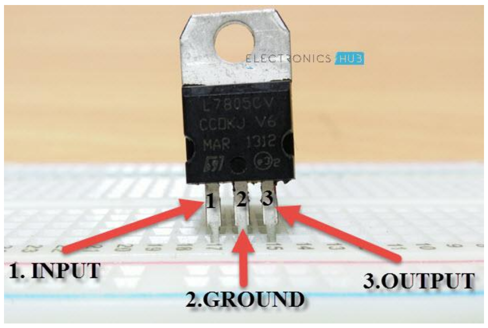
  Voltage sources in a circuit may have fluctuations resulting in not providing fixed voltage outputs. A voltage regulator IC maintains the output voltage at a constant value. The xx in 78xx indicates the output voltage it provides. 7805 IC provides +5 volts regulated power supply with provisions to add a heat sink.

- 7805 regulate voltage:

  For 7805 IC, it is +5V DC regulated power supply. This regulator IC also adds a provision for a heat sink. The input voltage to this voltage regulator can be up to 35V, and this IC can give a constant 5V for any value of input less than or equal to 35V which is the threshold limit.

- Pin Diagram of 7805 Voltage Regulator IC:

  As mentioned earlier, 7805 is a three terminal device with the three pins being 1. INPUT, 2. GROUND and 3. OUTPUT. The following image shows the pins on a typical 7805 IC in To-220 Package.



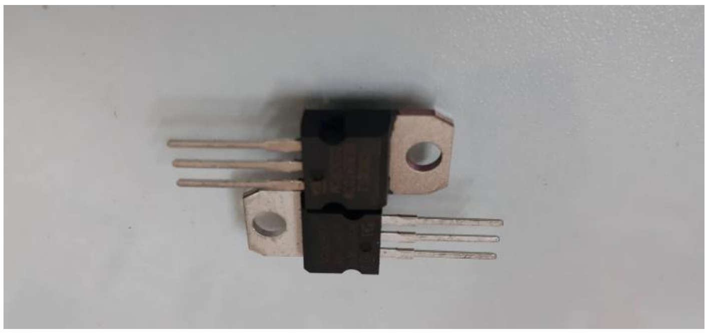
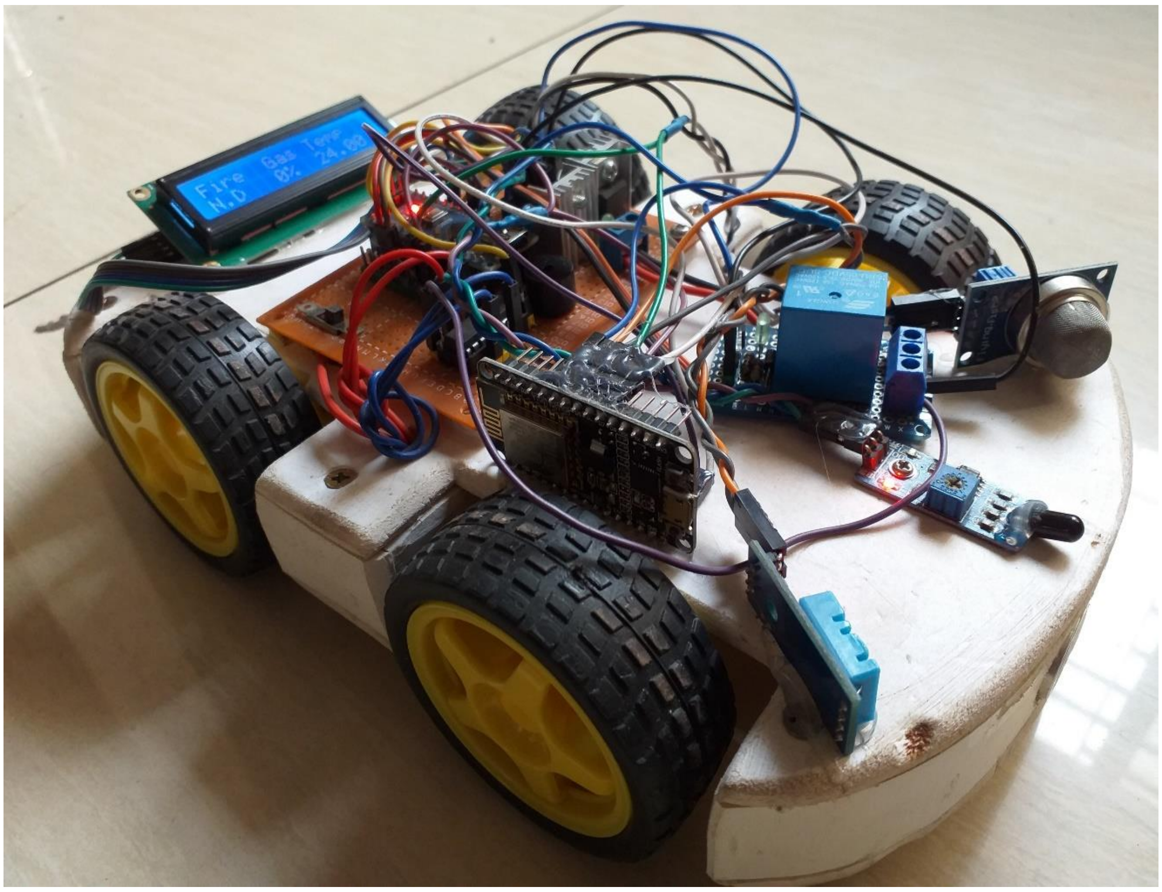


Figure 2.17: Connecting voltage regulator IC 7805 and 7806 to Arduino



IOT BASED FIRE DETECTION ROBOT.

2.5 List of Components with Price:

Sl.no	Particulars	Specification	Qty.	Unit Price (Taka)	Total Price(Taka)	Market Pric
1	Arduino Nano	Atmega 328p	1	280	280	
2	Node MCU		1	450	450	
3	Motor Driver IC		1	80	80	
4	Temperature Sensor		1	150	150	
5	Ultrasonic sensor		1	85	85	
6	Fire Sensor		1	40	40	
7	Smoke Sensor		1	120	120	
8	DC Gear Motor		4	310	1240	
9	Battery		1	450	450	
9	LCD Display		1	135	135	
10	Buzzer		1	15	15	
				Total	3045/=	3045/-

Table 1: List of Components with Price [6]

CHAPTER 3
HARDWARE ANALYSIS

3.1 Arduino Nano:

Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. It's intended for artists, designers, hobbyists, and anyone interested in creating interactive objects or environments. Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators.

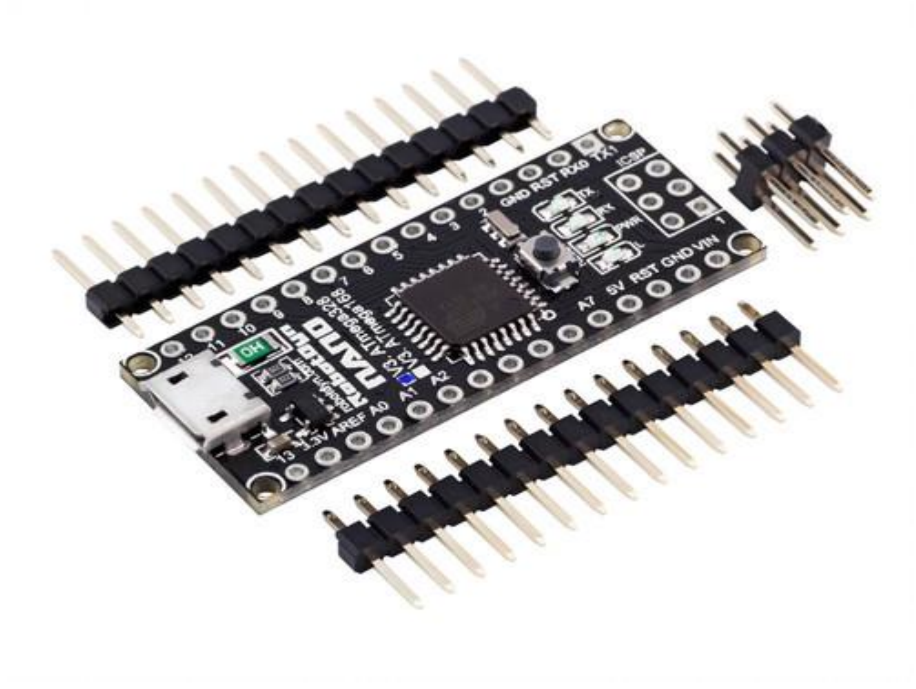


Figure 3.1:Arduino Nano

The microcontroller on the board is programmed using the Arduino programming language (based on Wiring) and the Arduino development environment (based on Processing). Arduino projects can be stand-alone or they can communicate with software on running on a computer (e.g. Flash, Processing, Max MSP). [7][8][9][10]

Arduino Nano is a surface mount breadboard embedded version with integrated USB. It is a smallest, complete, and breadboard friendly. It has everything that Decimal/Duemilanove has (electrically) with more analog input pins and onboard +5V AREF jumper. Physically, it is missing power jack. The Nano is automatically sense and switch to the higher potential source of power.

Nano's got the breadboard-ability of the Arduino and the Mini+USB with smaller footprint than either, so users have more breadboard space. It's got a pin layout that works well with the Mini or the Basic Stamp (TX, RX, ATN, and GND on one top, power and ground on the other). This new version 3.0 comes with ATMEGA328 which offer more programming and data memory space. It is two layers. That make it easier to hack and more affordable. [13]

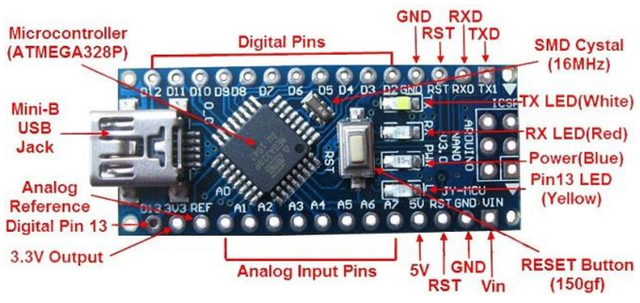


Figure 3.2: Section of Arduino Nano

Operating Voltage (logic level):5 V

Input Voltage (recommended):7-12 V

Input Voltage (limits):6-20 V

Digital I/O Pins: 14 (of which 6 provide PWM output)

Analog Input Pins: 8

DC Current per I/O Pin: 40 mA

Flash Memory: 32 KB (of which 2KB used by boot loader)

SRAM : 2 KB

EEPROM: 1 KB

Clock Speed: 16 MHz

Dimensions: 0.70" x 1.70"

Features:

• Automatic reset during program download
• Power OK blue LED
• Green (TX), red (RX) and orange (L) LED
• Auto sensing/switching power input
• Small mini-B USB for programming and serial monitor
• ICSP header for direct program download
• Standard 0.1 spacing DIP (breadboard friendly)
• Manual reset switch

Microcontroller IC ATmega328p:

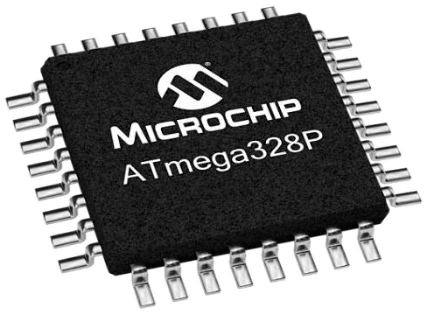


Figure 3.3: Microcontroller IC ATmega 328p

The high-performance Microchip Pico Power 8-bit AVR RISC-based microcontroller combines 32KB ISP flash memory with read-while-write capabilities, 1024B EEPROM, 2KB SRAM, 23 general purpose I/O lines, 32 general purpose working registers, three flexible timer/counters with compare modes, internal and external interrupts, serial programmable USART, a byte-oriented 2-wire serial interface, SPI serial port, a 6-channel 10-bit A/D converter (8-channels in TQFP and QFN/MLF packages), programmable watchdog timer with internal oscillator, and five software selectable power saving modes. The device operates between 1.8-5.5 volts. By executing powerful instructions in a single clock cycle, the device achieves throughputs approaching 1 MIPS per MHz, balancing power consumption and processing speed.

3.2 DC Gear Motor

A **DC Gear motor** is any of a class of rotary electrical machines that converts direct current electrical energy into mechanical energy. The most common types rely on the forces produced by magnetic fields. Nearly all types of DC motors have some internal mechanism, either electromechanical or electronic, to periodically change the direction of current flow in part of the motor.

DC motors were the first form of motor widely used, as they could be powered from existing direct-current lighting power distribution systems. A DC motor's speed can be controlled over a wide range, using either a variable supply voltage or by changing the strength of current in its field windings. Small DC motors are used in tools, toys, and appliances. The universal motor can operate on direct current but is a lightweight brushed motor used for portable power tools and appliances. Larger DC motors are currently used in propulsion of electric vehicles, elevator and hoists, and in drives for steel rolling mills. The advent of power electronics has made replacement of DC motors with AC motors possible in many applications.

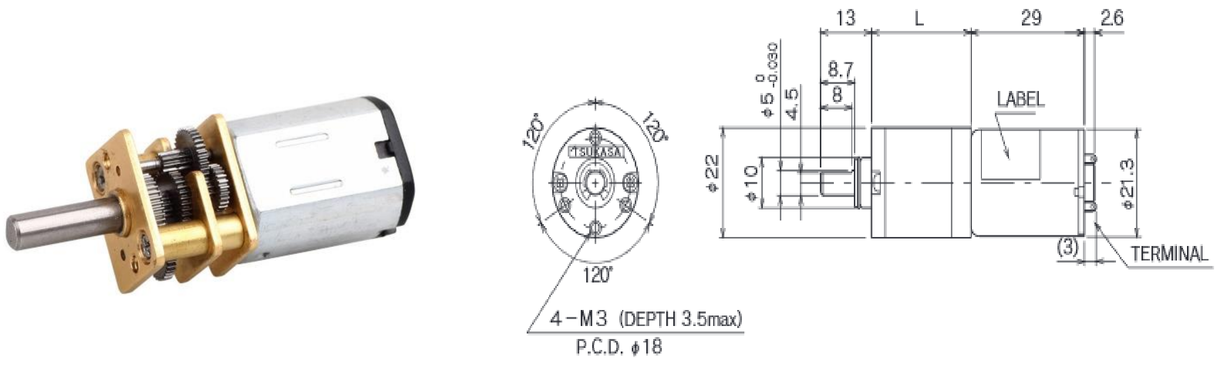


Figure 3.4: DC gear Motor

A coil of wire with a current running through it generates an electromagnetic field aligned with the center of the coil. The direction and magnitude of the magnetic field produced by the coil can be changed with the direction and magnitude of the current flowing through it.A simple DC motor has a stationary set of magnets in the stator and an armature with one or more windings of insulated wire wrapped around a soft iron core that concentrates the magnetic field. The windings usually have multiple turns around the core, and in large motors there can be several parallel current paths. The ends of the wire winding are connected to a commutator.

The commutator allows each armature coil to be energized in turn and connects the rotating coils with the external power supply through brushes. (Brushless DC motors have electronics that switch the DC current to each coil on and off and have no brushes.) The total amount of current sent to the coil, the coil's size and what it's wrapped around dictate the strength of the electromagnetic field created. A Direct Current (DC) motor is a rotating electrical device that converts direct current, of electrical energy, into mechanical energy. An Inductor (coil) inside the DC motor produces a magnetic field that creates rotary motion as DC voltage is applied to its terminal. Inside the motor is an iron shaft, wrapped in a coil of wire. This shaft contains two fixed, North and South, magnets on both sides which causes both a repulsive and attractive force, in turn, producing torque.

3.3 LCD Display:

LCD (Liquid Crystal Display) screen is an electronic display module and find a wide range of applications. A 16x2 LCD display is very basic module and is very commonly used in various devices and circuits. These modules are preferred over seven segments and other multi segment LEDs. The reasons being: LCDs are economical; easily programmable; have no limitation of displaying special & even custom characters (unlike in seven segments), animations and so on. A 16x2 LCD means it can display 16 characters per line and there are 2 such lines. In this LCD each character is displayed in 5x7 pixel matrix. This LCD has two registers, namely, Command and Data. The command register stores the command instructions given to the LCD. A command is an instruction given to LCD to do a predefined task like initializing it, clearing its screen, setting the cursor position, controlling display etc.

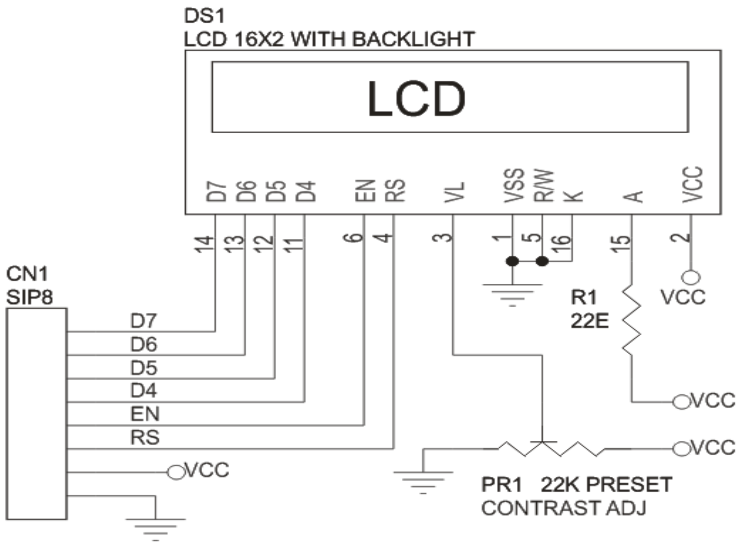


Figure 3.5: 16*2 LCD Display

Features of LCD Display

- 5 x 8 dots with cursor.
- Built-in controller (KS 0066 or Equivalent) + 5V power supply (Also available for + 3V) 1/16 duty cycle.
- B/L to be driven by pin 1, pin 2 or pin 15, pin 16 or A.K (LED) N.V. optional for + 3V power supply.

3.4 Motor Driver

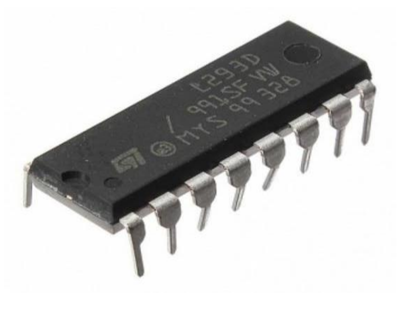


Figure 3.6 Motor Driver IC L293D

L293D is a typical Motor driver or Motor Driver IC which allows DC motor to drive on either direction. L293D is a 16-pin IC which can control a set of two DC motors simultaneously in any direction. It means that you can control two DC motor with a single L293D IC dual H-bridge *Motor Driver integrated circuit* (*IC*).

It works on the concept of H-bridge. H-bridge is a circuit which allows the voltage to be flown in either direction. As you know voltage need to change its direction for being able to rotate the motor in clockwise or anticlockwise direction, hence H-bridge IC are ideal for driving a DC motor.

In a single L293D chip there are two h-Bridge circuit inside the IC which can rotate two dc motor independently. Due its size it is very much used in robotic application for controlling DC motors. Given below is the pin diagram of a L293D motor controller.

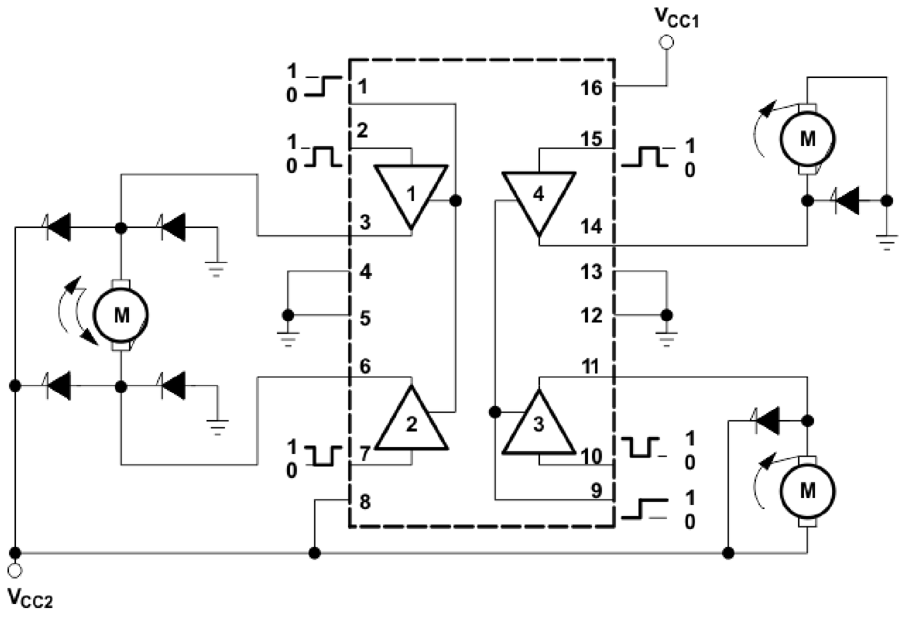


Figure 3.7: Schematic diagram of L293D Motor Driver IC

There are two Enable pins on l293d. Pin 1 and pin 9, for being able to drive the motor, the pin 1 and 9 need to be high. For driving the motor with left H-bridge you need to enable pin 1 to high. And for right H-Bridge you need to make the pin 9 to high. If anyone of the either pin 1 or pin 9 goes low then the motor in the corresponding section will suspend working. It’s like a switch.

There are 4 input pins for l293d, pin 2,7 on the left and pin 15,10 on the right as shown on the pin diagram. Left input pins will regulate the rotation of motor connected across left side and right input for motor on the right hand side. The motors are rotated on the basis of the inputs provided across the input pins as LOGIC 0 or LOGIC 1.
In simple you need to provide Logic 0 or 1 across the input pins for rotating the motor.

L293D Logic Table:

Let’s consider a Motor connected on left side output pins (pin 3,6). For rotating the motor in clockwise direction the input pins has to be provided with Logic 1 and Logic 0.
• **Pin 2** = **Logic 1** and **Pin 7** = **Logic 0** | Clockwise Direction
• **Pin 2** = **Logic 0** and **Pin 7** = **Logic 1** | Anticlockwise Direction
• **Pin 2** = **Logic 0** and **Pin 7** = **Logic 0** | Idle [No rotation] [Hi-Impedance state]
• **Pin 2** = **Logic 1** and **Pin 7** = **Logic 1** | Idle [No rotation]
In a very similar way the motor can also operate across input pin 15,10 for motor on the right hand side.
Circuit Diagram for l293d motor driver IC controller.

3.5 Ultrasonic Sensor:

The HC-SR04 ultrasonic sensor uses sonar to determine distance to an object like bats do. It offers excellent non-contact range detection with high accuracy and stable readings in an easy-to-use package. It comes complete with ultrasonic transmitter and receiver modules. [11][12]

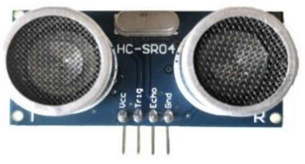


Figure 3.8: Ultrasonic Sensor

Features:

Here’s a list of some of the HC-SR04 ultrasonic sensor features and specs:

1. Power Supply :+5V DC
2. Quiescent Current : <2mA
3. Working Current: 15mA
4. Effectual Angle: <15°
5. Ranging Distance : 2cm – 400 cm/1″ – 13ft
6. Resolution : 0.3 cm
7. Measuring Angle: 30 degree
8. Trigger Input Pulse width: 10uS
9. Dimension: 45mm x 20mm x 15mm

As shown above the **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

10. **Distance = Speed × 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 as shown in the picture below

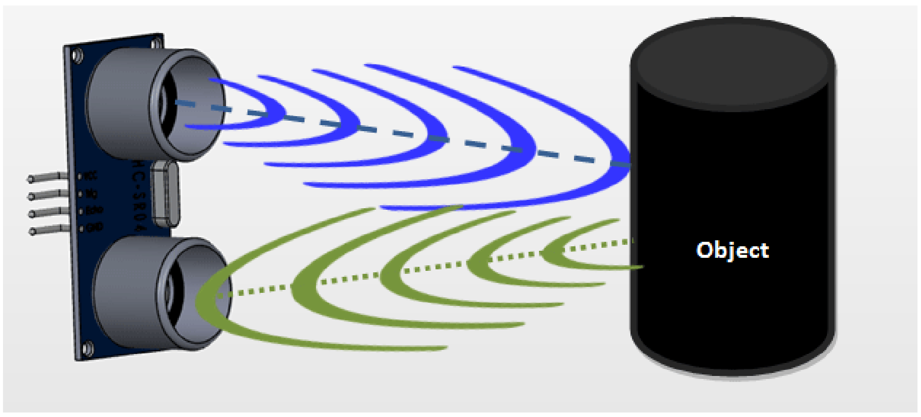


Figure 3.9: Ultrasonic Sensor detecting an object

Now, to calculate the distance using the above formulae, we should know the Speed and time. Since we are using the Ultrasonic wave we know the universal speed of US wave at room conditions which is 330m/s. The circuitry inbuilt on the module will calculate the time taken for the US wave to come back and turns on the echo pin high for that same particular amount of time, this way we can also know the time taken. Now simply calculate the distance using a microcontroller or microprocessor. [11][12]

3.6 DHT11 Temperature Sensor

Each DHT11 element is strictly calibrated in the laboratory that is extremely accurate on humidity calibration. The calibration coefficients are stored as programs in the OTP memory, which are used by the sensor's internal signal detecting process. The single-wire serial interface makes system integration quick and easy. Its small size, low power consumption and up-to-20-meter signal transmission making it the best choice for various applications, including those most demanding ones. The components 4-pin single row pin package. It is convenient to connect and special packages can be provided according to user's request. [3][14]

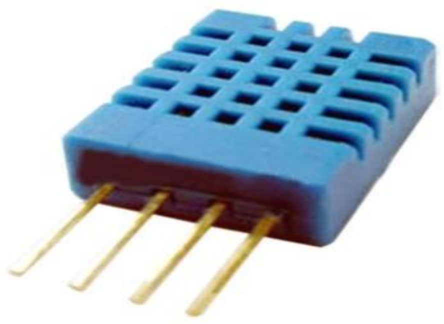


Figure 3.10: DHT11 sensor

DHT11 Specifications:

1. Operating Voltage: 3.5V to 5.5V
2. Operating current: 0.3mA (measuring) 60uA (standby)
3. Output: Serial data
4. Temperature Range: 0°C to 50°C
5. Humidity Range: 20% to 90%
6. Resolution: Temperature and Humidity both are 16-bit
7. Accuracy: ±1°C and ±1%

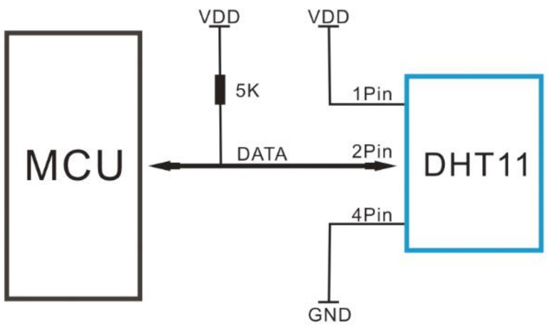


Figure 3.11: Typical Application of DHT11 sensor

3.7 Flame Sensor

The flame sensor can detect flame and infrared light sources with wavelengths ranging from 760 nm to 1100 nm. It uses the LM393 comparator chip, which gives a clean, stable digital output signal and driving ability of 15 mA. [11][12]

This flame detector that can be used in fire alarms and other fire detecting devices.

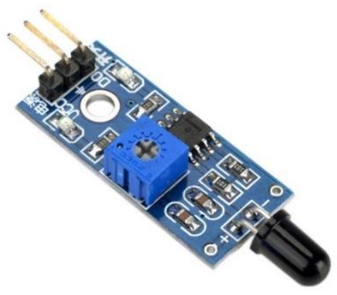


Figure 3.12: Fire Sensor

Specifications

- LM393 comparator chip
- Detection Range: 760 nm to 1100 nm
- Operating Voltage: 3.3 V to 5 V
- Maximum Output Current: 15 mA
- Digital Outputs: 0 and 1
- Detection Angle: about 60 degrees
- Adjustable sensitivity via potentiometer
- LED lights indicators: power (red) and digital switching output (green)
- Fixed bolt holes for easy installation
- PCB Size: 32 x 14 mm

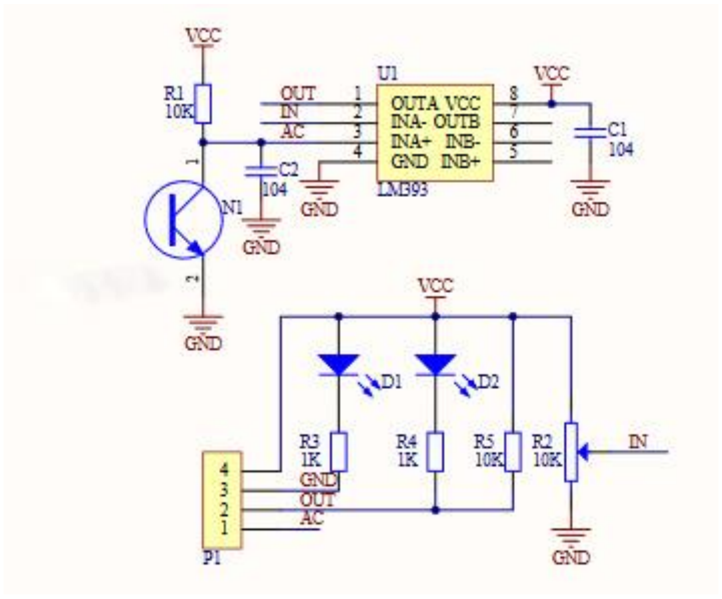


Figure 3.13: Schema Diagram of Flame Sensor

3.8 Node MCU

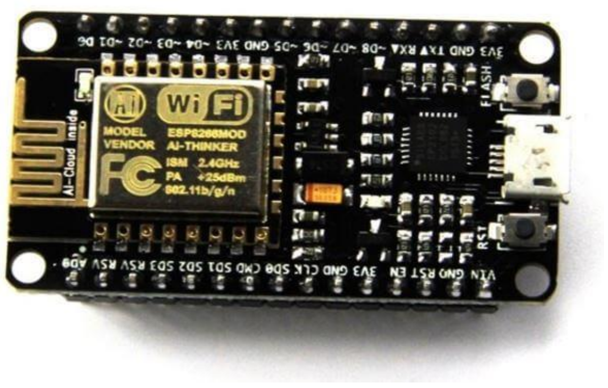


Figure 3.14: Node MCU

Node MCU is an open source IoT platform. It includes firmware which runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module. The term "Node MCU" by default refers to the firmware rather than the development kits. The firmware uses the Lua scripting language. It is based on the eLua project, and built on the Espressif Non-OS SDK for ESP8266.

Advanced API for hardware IO, which can dramatically reduce the redundant work for configuring and manipulating hardware. Code like arduino, but interactively in Lua script. Event-driven API for network applications, which facilitates developers writing code running on a 5mm*5mm sized MCU in Nodejs style. Greatly speed up your IOT application developing process.

Features

Wi-Fi Module – ESP-12E module similar to ESP-12 module but with 6 extra GPIOs.
USB – micro USB port for power, programming and debugging
Headers – 2x 2.54mm 15-pin header with access to GPIOs, SPI, UART, ADC, and power pins
Misc – Reset and Flash buttons
Power – 5V via micro USB port
Dimensions – 49 x 24.5 x 13mm
NodeMCU was created shortly after the ESP8266 came out. On December 30, 2013, Espressif Systems began production of the ESP8266. The ESP8266 is a Wi-Fi SoC integrated with a Tensilica Xtensa LX106 core, widely used in IoT applications (see related projects). NodeMCU started on 13 Oct 2014, when Hong committed the first file of nodemcu-firmware to GitHub. Two months later, the project expanded to include an open-hardware platform when developer Huang R committed the gerber file of an ESP8266 board, named devkit v0.9.[12] Later that month, Tuan PM ported MQTT client library from Contac to the ESP8266 SoC platform,[and committed to Node MCU project, then Node MCU was able to support the MQTT IoT protocol, using Lua to access the MQTT broker. Another important update was made on 30 Jan 2015, when Devsaurus ported the u8glib to Node MCU project, enabling Node MCU to easily drive LCD, Screen, OLED, even VGA displays.
In summer 2015 the creators abandoned the firmware project and a group of independent contributors took over. By summer 2016 the Node MCU included more than 40 different modules. Due to resource constraints users need to select the modules relevant for their project and build a firmware tailored to their needs.

ESP8266 Arduino Core

As Arduino.cc began developing new MCU boards based on non-AVR processors like the ARM/SAM MCU and used in the Arduino Due, they needed to modify the Arduino IDE so that it would be relatively easy to change the IDE to support alternate toolchains to allow Arduino C/C++ to be compiled for these new processors. They did this with the introduction of the Board Manager and the SAM Core. A "core" is the collection of software components required by the Board Manager and the Arduino IDE to compile an Arduino C/C++ source file for the target MCU's machine language. Some ESP8266 enthusiasts developed an Arduino core for the ESP8266 Wi-Fi SOC, popularly called the "ESP8266 Core for the Arduino IDE". This has become a leading software development platform for the various ESP8266-based modules and development boards, including Node MCUs.

3.9 Smoke Sensor

The MQ series of gas sensors use a small heater inside with an electrochemical sensor. They are sensitive to a range of gasses and are used indoors at room temperature. The output is an analog signal and can be read with an analog input of the Arduino.

The MQ-2 Gas Sensor module is useful for gas leakage detecting in home and industry. It can detect LPG, i-butane, propane, methane, alcohol, hydrogen and smoke.

Some modules have a built-in variable resistor to adjust the sensitivity of the sensor.

Note: The sensor becomes very hot after a while, don't touch it!

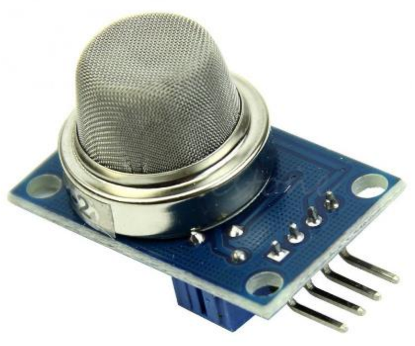


Figure 3.15: Smoke sensor

The connections are pretty easy:

- VCC to Arduino 5V pin
- GNG to Arduino GND pin
- Output to Arduino Analog A0 pin

3.10 Buzzer

- A buzzer is an audio signaling device.
- A piezo buzzer is connected to a digital output, which emits a beep tone when the output is high.
- When the piezo buzzer is connected to the analog pulse modulation output, it gives varying tones and effects.

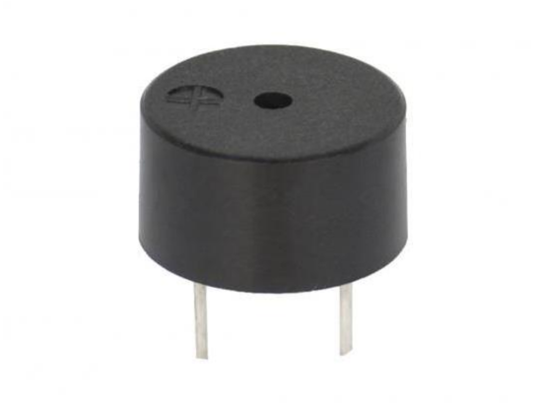


Figure 3.16**:** Buzzer

3.11 5V Relay

Relays are most commonly used switching device in electronics. Before we proceed with the circuit to drive the relay we have to consider two important parameter of the relay. Once is the Trigger Voltage, this is the voltage required to turn on the relay that is to change the contact from Common->NC to Common->NO. Our relay here has 5V trigger voltage, but you can also find relays of values 3V, 6V and even 12V so select one based on the available voltage in your project. The other parameter is your **Load** Voltage & Current, this is the amount of voltage or current that the NC, NO or Common terminal of the relay could withstand, in our case for DC it is maximum of 30V and 10A. Make sure the load you are using falls into this range

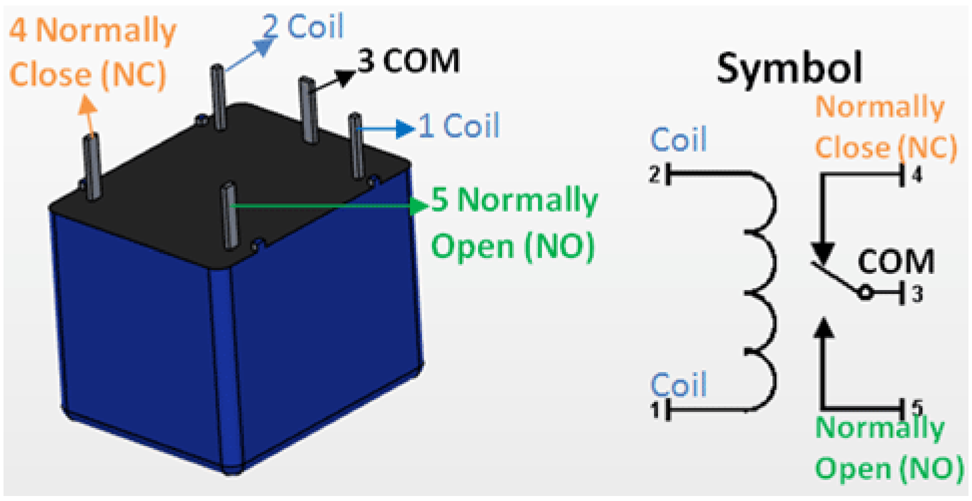


Figure 3.17: 5V Relay Connection Diagram

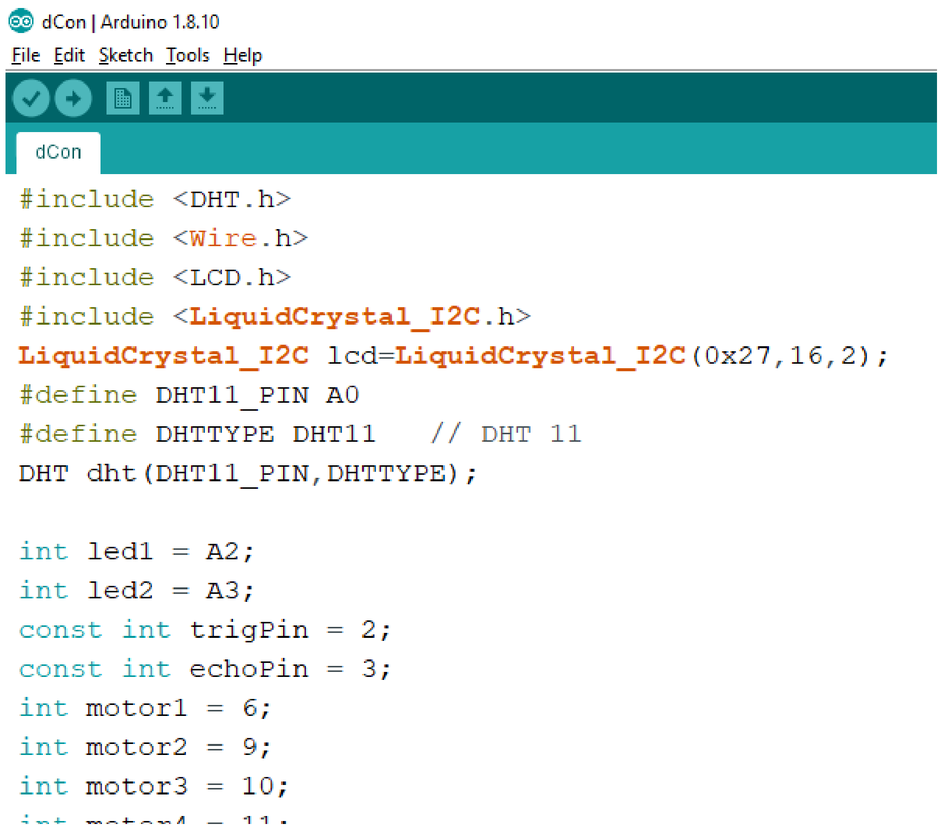
The above circuit shows a bare-minimum concept for a relay to operate. Since the relay has 5V trigger voltage we have used a +5V DC supply to one end of the coil and the other end to ground through a switch. This switch can be anything from a small transistor to a microcontroller or a microprocessor which can perform switching operating. You can also notice a diode connected across the coil of the relay, this diode is called the Fly Back Diode. The purpose of the diode is to protect the switch from high voltage spike that can produced by the relay coil. As shown one end of the load can be connected to the Common pin and the other end is either connected to NO or NC. If connected to NO the load remains disconnected before trigger and if connected to NC the load remains connected before trigger.

CHAPTER 4

SOFTWARE DESCRIPTION

4.1 Arduino software:

The smart microcontroller unit named as Arduino Nano can be programmed with the Arduino software. The Arduino Nano is a small, complete, and breadboard-friendly board based on the ATmega328P; offers the same connectivity and specs of the UNO board in a smaller form factor.



```
#include <DHT.h>
#include <Wire.h>
#include <LCD.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd=LiquidCrystal_I2C(0x27,16,2);
#define DHT11_PIN A0
#define DHTTYPE DHT11   // DHT 11
DHT dht(DHT11_PIN,DHTTYPE);

int led1 = A2;
int led2 = A3;
const int trigPin = 2;
const int echoPin = 3;
int motor1 = 6;
int motor2 = 9;
int motor3 = 10;
```

Figure 4.1: Arduino Software Interface IDE

The Arduino Nano is programmed using the Arduino Software (IDE), our Integrated Development Environment common to all our boards and running both online and offline. If you want to program your Arduino Nano while offline you need to install the Arduino Desktop IDE To connect the Arduino Nano to your computer, you'll need a Mini-B USB cable. This also provides power to the board, as indicated by the blue LED (which is on the bottom of the Arduino Nano 2.x and the top of the Arduino Nano 3.0).

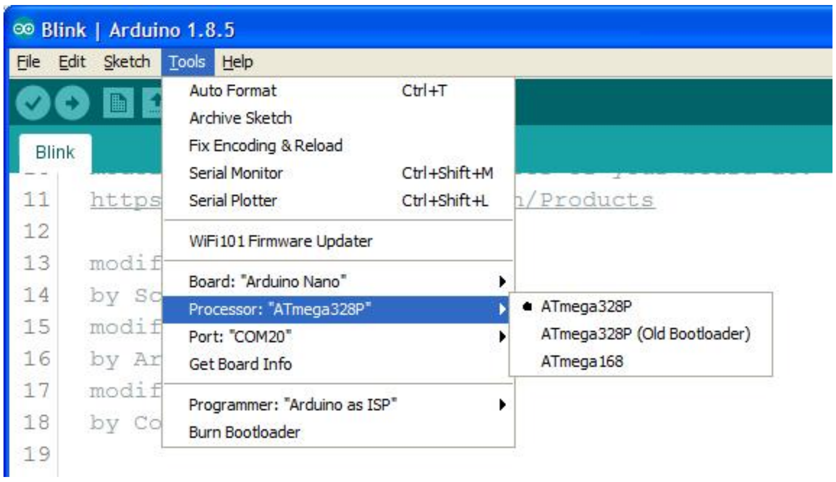
Open your first sketch

Open the LED blink example sketch: File > Examples >01.Basics > Blink.

Select your board type and port

You'll need to select the entry in the Tools > Board menu that corresponds to your Nano board.

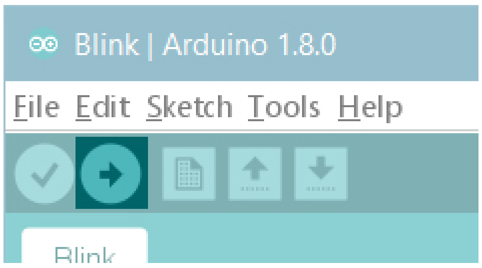
NOTE: We have updated the NANO board with a fresh bootloader. Boards sold from us from January 2018 have this new bootloader, while boards manufactured before that date have the old bootloader. First, make sure you have the Arduino AVR Core 1.16.21 or later looking at the Board Manager. Then, to program the NEW Arduino NANO boards you need to chose Processor > "ATmega328P". To program old boards you need to choose Processor > "ATmega328P (Old Bootloader)". If you get an error while uploading or you are not sure which bootloader you have, try each type of processor 328P until your board gets properly programmed.



Board: "Arduino Nano"
Processor: "ATmega328P"
Port: "COM20"
Programmer: "Arduino as ISP"
● ATmega328P

Upload and Run your first Sketch

To upload the sketch to the Arduino Nano, click the Upload button in the upper left to load and run the sketch on your board:

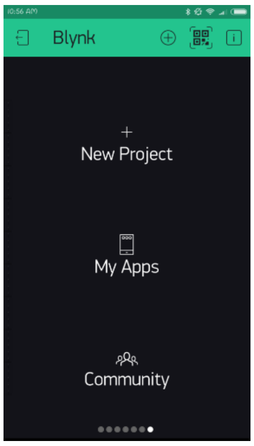


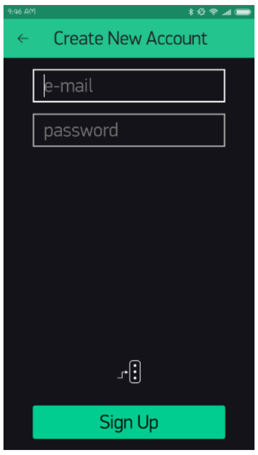
Wait a few seconds - you should see the RX and TX leds on the board flashing. If the upload is successful, the message "Done uploading." will appear in the status bar.

4.2 Blynk App

Blynk is a Platform with IOS and Android apps to control Arduino, Raspberry Pi and the likes over the Internet. It's a digital dashboard where you can build a graphic interface for your project by simply dragging and dropping widgets.

After downloading the app, create an account and log in. (If possible than log in with your real mail id for better connectivity later.)

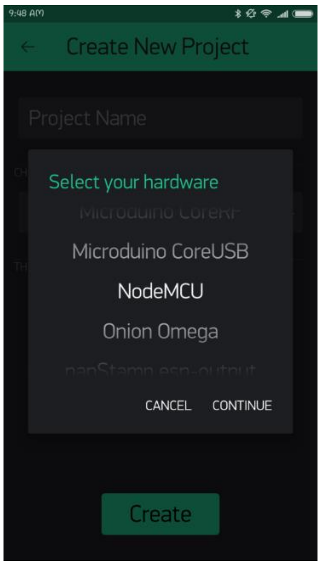


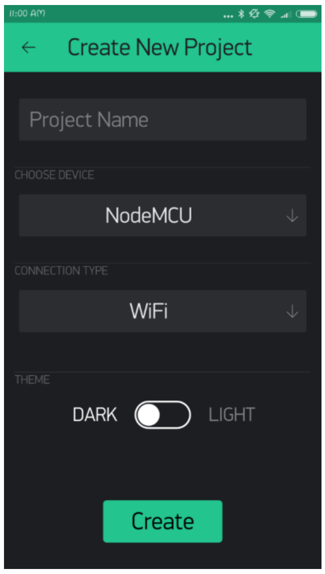


Create a Blynk Project:

Click the "Create New Project" in the app to create a new Blynk app. Give it any name.

Blynk works with hundreds of hardware models and connection types. Select the Hardware type. After this, select connection type. In this project we have select WiFi connectivity.





The Auth Token is very important – you'll need to stick it into your ESP8266's firmware. For now, copy it down or use the "E-mail" button to send it to yourself.

Add Widgets To The Project:-

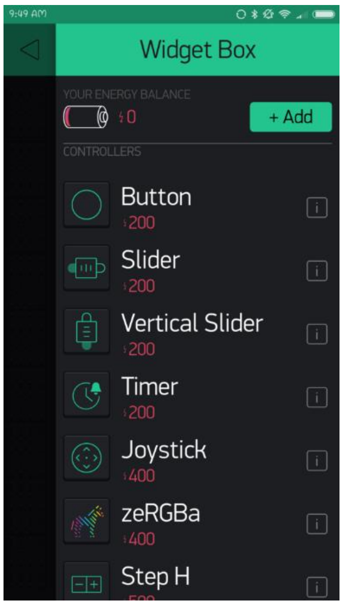
Then you'll be presented with a blank new project. To open the widget box, click in the project window to open.

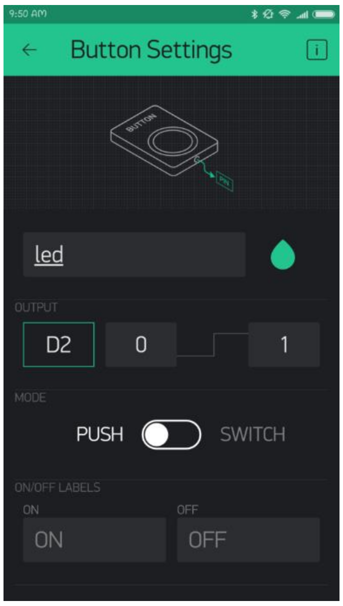
We are selecting a button to control Led connected with NodeMCU.

1. Click on Button.
2. Give name to Button say led.
3. Under OUTPUT tab- Click pin and select the pin to which led is connected to NodeMCU, here it is digital pin 2, hence select digital and under pin D2. And Click continue.

Under MODE tab- Select whether you want this button as "push button" or "Switch".

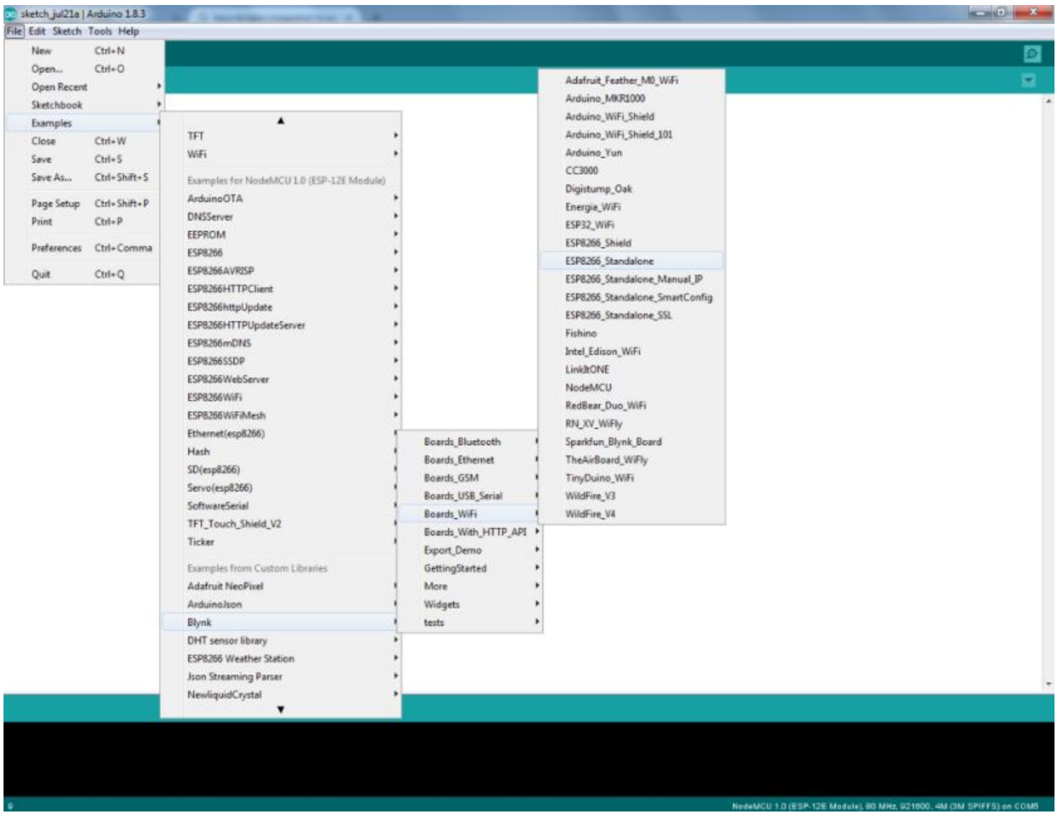
You have successfully created a GUI for Arduino.





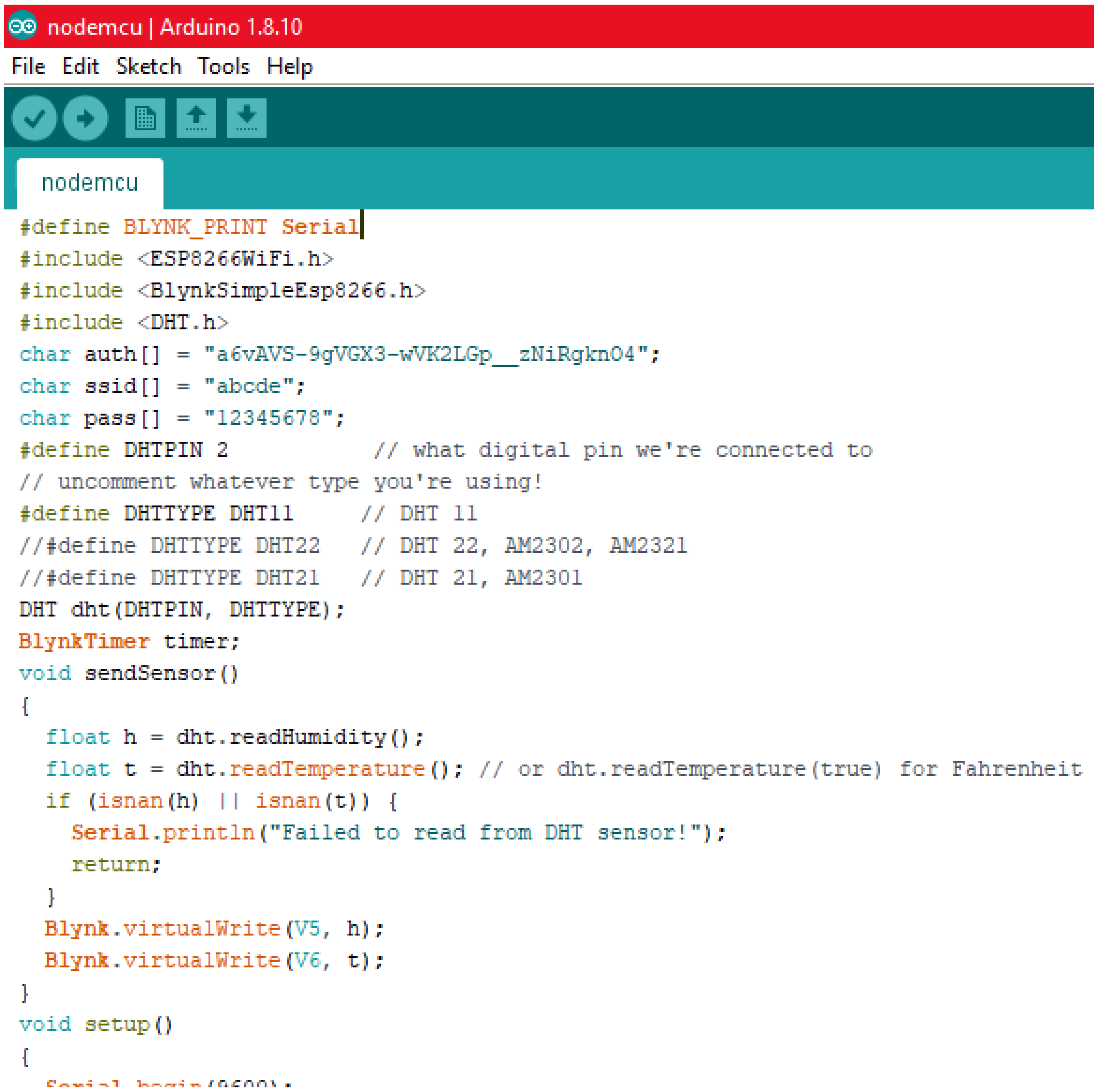
Upload The Firmware

Now that your Blynk project is set-up, open Arduino and navigate to the ESP8266_Standalone example in the File > Examples > Blynk > Boards_WiFi> ESP8266_Standalone menu.



Stand Alone Programming Code:

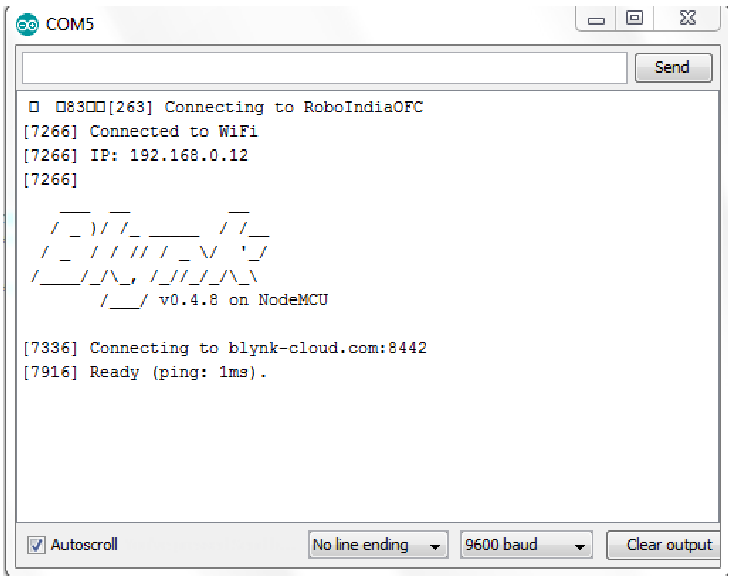
Before uploading, make sure to paste your authorization token into the auth [] variable. Also make sure to load your Wifi network settings into the Blynk.begin(auth, “ssid”, “pass”) function.



```
#define BLYNK_PRINT Serial
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#include <DHT.h>
char auth[] = "a6vAVS-9gVGX3-wVK2LGp__zNiRgkn04";
char ssid[] = "abcde";
char pass[] = "12345678";
#define DHTPIN 2          // what digital pin we're connected to
// uncomment whatever type you're using!
#define DHTTYPE DHT11     // DHT 11
//#define DHTTYPE DHT22   // DHT 22, AM2302, AM2321
//#define DHTTYPE DHT21   // DHT 21, AM2301
DHT dht(DHTPIN, DHTTYPE);
BlynkTimer timer;
void sendSensor()
{
  float h = dht.readHumidity();
  float t = dht.readTemperature(); // or dht.readTemperature(true) for Fahrenheit
  if (isnan(h) || isnan(t)) {
    Serial.println("Failed to read from DHT sensor!");
    return;
  }
  Blynk.virtualWrite(V5, h);
  Blynk.virtualWrite(V6, t);
}
void setup()
{
  Serial.begin(9600);
```

Execution:

After the app has uploaded, open the serial monitor, setting the baud rate to 9600. Wait for the “Ready” message.



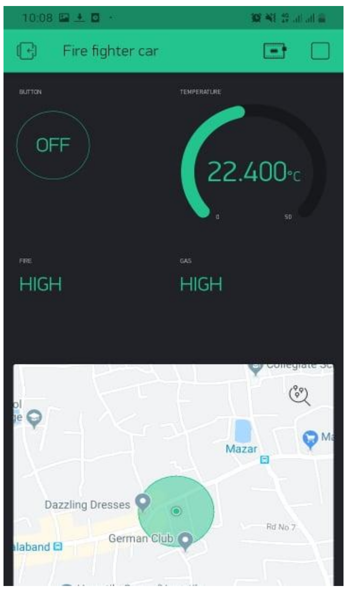
```
□  □83□□[263] Connecting to RoboIndiaOFC
[7266] Connected to WiFi
[7266] IP: 192.168.0.12
[7266]
    ___  __          __
   / _ )/ /_ _____  / /__
  / _  / / // / _ \/  '_/
 /____/_/\_, /_//_/_/\_\
        /___/ v0.4.8 on NodeMCU

[7336] Connecting to blynk-cloud.com:8442
[7916] Ready (ping: 1ms).
```

Then click the “Run” button in the top right corner of the Blynk app. Press the button and watch the LED

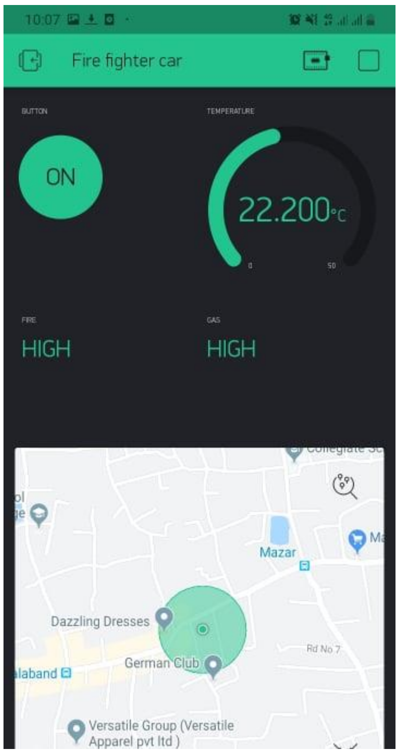
Then add more widgets to the project. They should immediately work on the ESP8266 without uploading any new firmware.

Output

- After input our code to Node MCU it will able to build the connection with Blynk App.



- Our robot is ready to launch with all included feature.



CHAPTER 5

Result and Analysis

5.1 Overview

This chapter illustrates the test results of the embedded device under a few different conditions with graphical representation when required.

In this chapter, the performance analysis is also given and the accuracy of the system is also analyzed.

5.2 Temperature, Gas and Fire Detection Tests

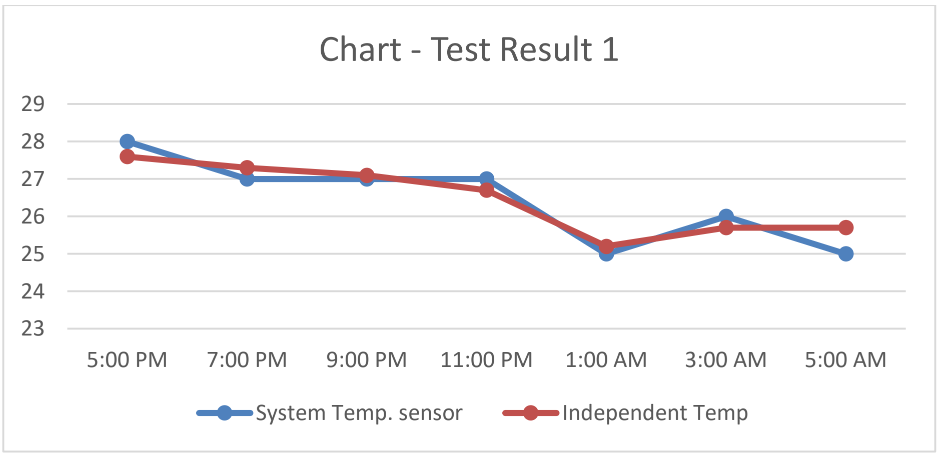
This section details two separate tests. One is for Temperature monitoring and other is for gas detection test.

5.2.1 Temperature Monitoring

Tests were performed in two separate conditions. Both the tests were performed for 12 hours. The system sensor readings were recorded every hour and a separate commercial electronic thermometer was used to record temperature independently.

Table 2 – Test Result for indoor Testing

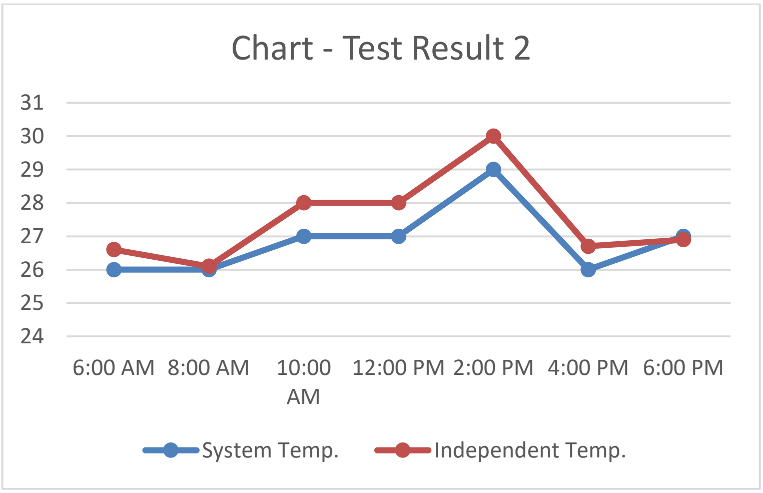
Hours	System Temp. (Celsius)	Independent Temp.
5 pm	28	27.60
6 pm	28	28.10
7 pm	27	27.30
8 pm	28	27.90
9 pm	27	27.10
10 pm	27	27.00
11 pm	27	26.70
12 am	26	26.10
1 am	25	25.20
2 am	26	25.70
3 am	26	25.70
4 am	25	25.70
5 am	25	25.70



Test 1 was in indoor condition. The test started at 5 pm in the evening and ended at 5 am in the morning. Table xx shows the system reading and independent temperature reading.

Table 3 – Test Result for outdoor testing

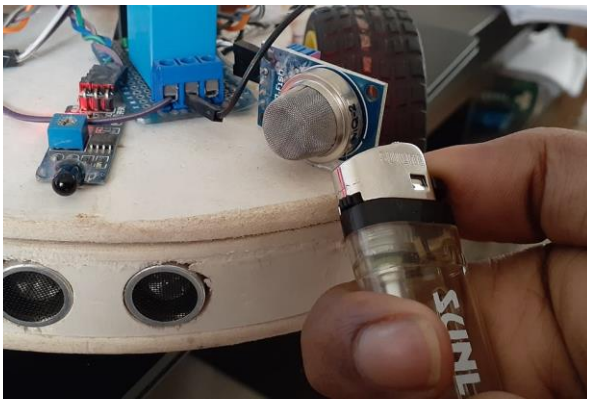
Hours	System Temp. (Celsius)	Independent Temp.
6 am	26	26.60
7 am	26	26.10
8 am	26	26.30
9 am	26	26.90
10 am	27	27.10
11 am	27	27.30
12 pm	27	28.00
1 pm	28	29.10
2 pm	29	30.00
3 pm	26	27.20
4 pm	26	26.70
5 pm	27	26.70
6 pm	27	26.90



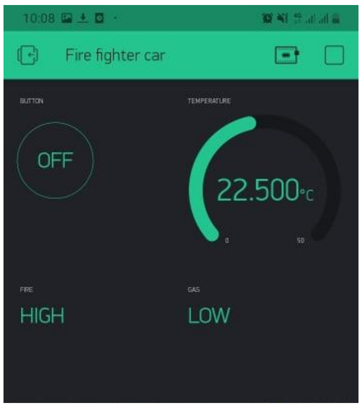
Test 2 was outdoor test. Test started at 6 AM in the morning and ended at 6 PM in the evening. Table xy shows the test results.

5.2.2 Gas Detection Test

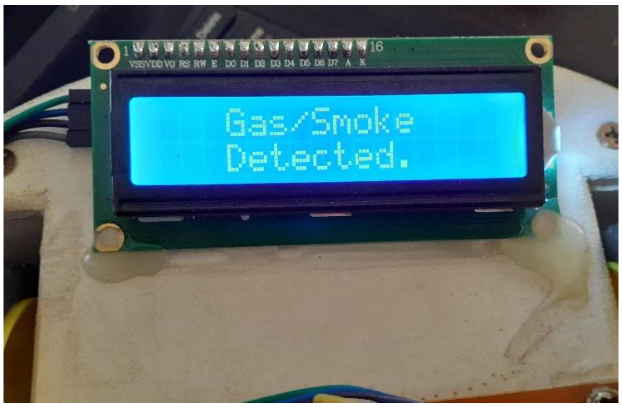
For gas detection test a gas cigarette lighter was used. In normal room environment the sensor was set to trigger above 300. The gas from the lighter quickly triggered the buzzer alarm and able to activated any fire defiance system.



Physical Input



Software Output

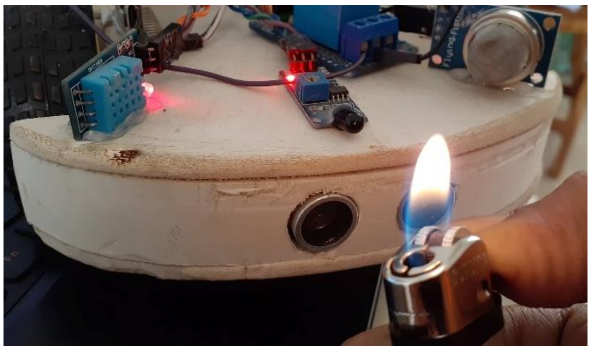


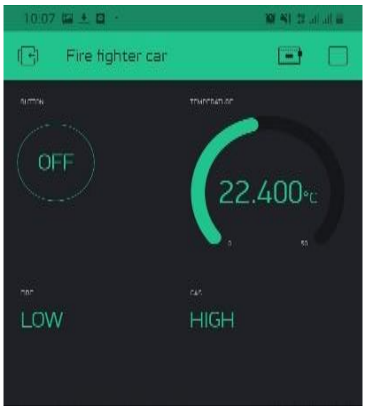
Gas/Smoke
Detected.

Hardware Output

5.2.2 Fire Detection Test

Flame sensors for fire detection and flame monitoring applications can be activated and tested. Most optical flame sensors respond to InfraRed (IR) radiation emitted from flames during combustion. The test unit simulates the flickering flame signal by modulating the output of a cigarette lighter or filament lamp.


Physical Input

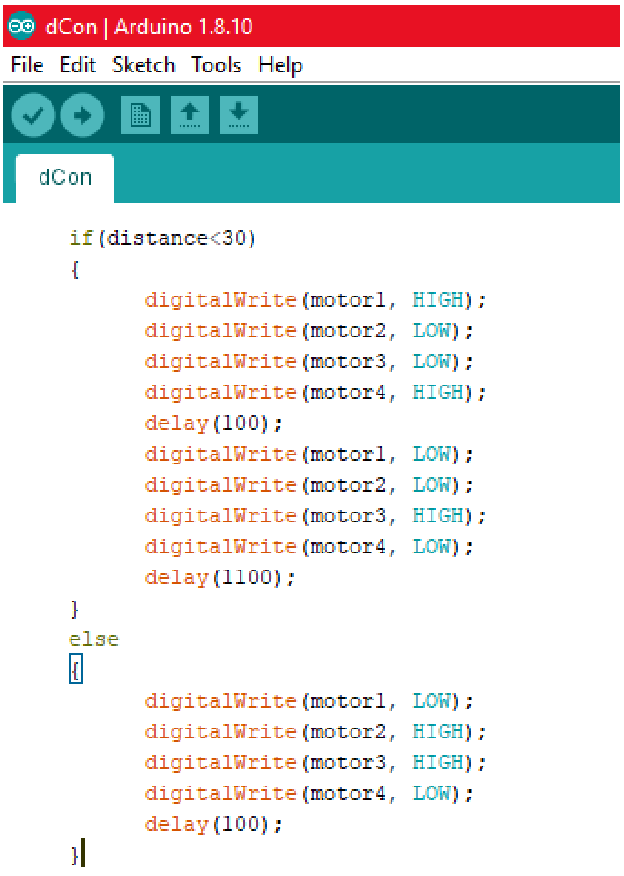


Software Output

Hardware Output

5.3 Detecting Obstacle

HC SR-04 Ultrasonic detector module has transmitter and receiver section. The sensor is interfaced with the micro controller with one output as the trigger and the other input as echo. After detecting the input echo signal by the receiver section. Then Microcontrollers triggers sent the signal to motor driver IC. If the echo signal with a high width is detected in the predefined range, then the obstacle is considered and at the same time the motor driver IC L293D commanded to LOW (motor 2 and 3) then it turn to right and find another way to ahead.



```
if(distance<30)
{
        digitalWrite(motor1, HIGH);
        digitalWrite(motor2, LOW);
        digitalWrite(motor3, LOW);
        digitalWrite(motor4, HIGH);
        delay(100);
        digitalWrite(motor1, LOW);
        digitalWrite(motor2, LOW);
        digitalWrite(motor3, HIGH);
        digitalWrite(motor4, LOW);
        delay(1100);
}
else
{
        digitalWrite(motor1, LOW);
        digitalWrite(motor2, HIGH);
        digitalWrite(motor3, HIGH);
        digitalWrite(motor4, LOW);
        delay(100);
}
```

Figure 5.1: Obstacle Detection Code

At any instant when an obstacle is found in the range of ultrasonic sensors then the firmware activated the L293D. The Ultrasonic Detector HC-SR04 circuit is a circuit which gives a low output in the absence of Ultrasonic signal when some obstacle come in path Ultrasonic signal is reflected back and fall onto the Ultrasonic detector. In such a way that obstacle is detected. Ultrasonic ranging module HC - SR04
Provides 2cm - 400cm non-contact.
Test distance = (high level time*velocity of sound (340M/S) / 2

5.4 Result Analysis

After the analysis of test results the following conclusion can be drawn-

1) Temperature was measured and error is within the tolerance limit.
2) Gas presence was accurately detected.
3) Ultrasonic Sensation value was measured and error is within the tolerance limit.
4) Our wifi module properly behaved with Arduino.
5) According to code the precision movement of our robot within the tolerance limit.
6) Android App based monitoring and control was successful.
7) Buzzer was on and LED were lit in case of abnormal temperature / gas / smoke / fire presence.

Chapter 6

FUTURE RECOMMENDATION

6.1 Future Improvements and Applications

This project is highly adaptable. However, it still has rooms for improvement. Following improvements are suggested –

- We can add camera to detect and get visible interface of any object.
- We can connect this system with fire extinguisher.
- We can add GSM module.
- Speed of the system can be increased accounting to the speed of production.
- We can also implement voice command to give some instruction
- The DC motor can be replaced with stepper motor.
- A touch screen could be added instead of the keypad shield and LCD display.
- TPM36 temperature sensor is more accurate than DHT11.
- The whole setup can be printed on PCB.

Moreover, the future application of this project is huge. This project demonstrates a Arduino based automation system. Although the scope of this project is limited, it can be broadened significantly.

- With some modification this can be used for fire defense.
- With some modification it can be used to detect weather condition.
- By increasing its sensing capacity, it can be used in any defensive actions.

Chapter 7

Discussion

7.1 Discussion

The project was completed in 13 weeks. Chart below shows the time allocation for the different steps of the project.

This project reflects our academic knowledge we gathered during our undergraduate studies. Our understanding about the hardware and coding skills were tested, refreshed during this ordeal.

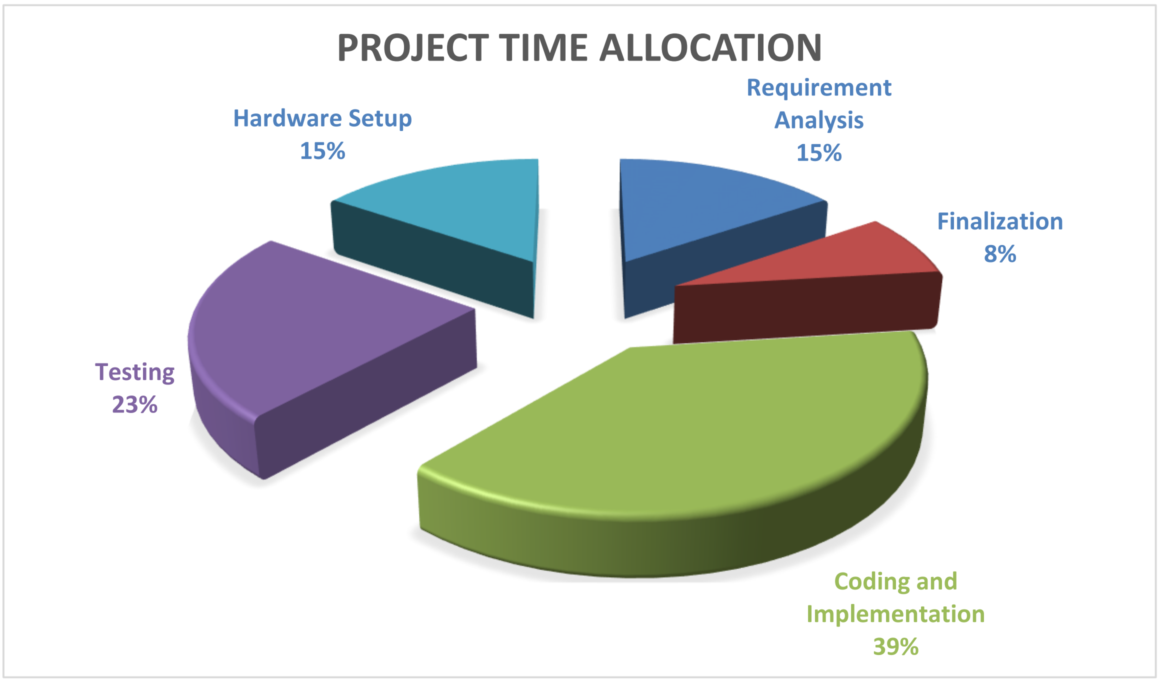


Figure 7.1: Project Time Allocation

Hardware based projects are often delayed by the lack of available funds, complications arising from underestimated design methodologies and poor hardware to software interfacing. During our project works, we faced all those difficulties and tried to overcome most of them. But in the end we are quite confident that we have presented an essential, modular and versatile system which can be used everywhere in a relatively low cost setup.

7.2 Conclusion

In this project IOT based Fire Detection robot we have successfully completed & the test results of this project are quite good in now days' robotics technology are becoming more and more efficient to complete our daily task. And it's become more usable in industry purpose. We were tried to make an implement on robotics which may can help us or may give us some idea to implement or improve our goals. We are hopping and want to make more improve on this project in future. As this project can help to detect fire and gas lick so this kind of robot can usable for home and also in defense area by some modification. We are looking for a great invention with our little dream project.

Reference

[1] Kouhia, E.-P., 2016. Development of an Arduino based Embedded system, s.l.: s.n.

[2] Banzi, M., n.d. How Arduino is open-sourcing imagination. s.l.: TED Talk.

[3] Vleeschouver, K. D., Loey, Van, A. & hendrickx, M. E., 2017. The Effect of High Pressure-High Temperature Processing Conditions on Acrylamide Formation and Other maillard Reaction Compounds. Journal of Agricultural and food chemistry, Volume 2010, 58(22), pp. 11740-11748.

[4] Akami, P., Oke, A. & Akpomiemie, O., 2015. Impact of environmental factors on building project. ScienceDirect, 11(1), pp. 91-97.

[5] E. Harshavardhan Goud , A. Harshika, G. Akhil, D. Charishma, K. Bhupathi, I. Kumara Swamy.
Real Time Based Temperature Control Using Arduino. International Journal of Innovations in Engineering and Technology (IJIET). http://dx.doi.org/10.21172/ijiet.82.030 (Last accessed 20-09-2018)

[6] Techshop Bangladesh. https://www.techshopbd.com/tutorial-categories/arduino/49/getting-started-with-arduino-techshop-bangladesh (Last accessed 14-09-2018)

[7] Anon., 2017. Arduino-Wikipedia. [Online] Available at:https://en.wikipedia.org/wiki/Arduino [Accessed 10 10 2018].

[8] kushner, D., 2011. The Making of Arduino-IEEE Spectrum. [Online] Available at: https://spectrum.ieee.org/geek-life/hands-on/the-making-of-arduino [Accessed 11 10 2018].

[9] Sandhu, R., 2016. What is Arduino? A full definition. [Online] Available at: https://www.lifewire.com/what-is-arduino-2495652 [Accessed 12 10 2018].

[10] Anon.,2017. ArduinoUnoRev3. [Online] Available at: https://store.arduino.cc/arduino-uno-rev3 [Accessed 12 10 2018].

[11] Kenny, D. T., 2005. Basic Sensor Technology. In: J. S. Wilson, ed. Sensor Technology Handbok. s.l.:s.n.
[12] Wilson, J. S., 2005. Sensor Technology Handbook. Oxford: Elsevier Inc.
[13] Blaauw, F. et al., 2016. Let's get Physical-An intuitive and generic methodd to combine sensor technology with ecological momentary assessments. SciencceDirecct, Volume 63, pp. 141-149.
[14] Anon.,2017. DHT11.pdf. [Online] Available at: https://akizukidenshi.com/download/ds/aosong/DHT11.pdf [Accessed 12 10 2018].
[15] Arduino Inc, 2018. [Online] Available at: https://www.arduino.cc/en/Reference/LiquidCrystal [Accessed 13 10 2018]

APPENDIX A

LIBRARIES AND CODE

Local Control Sketch for Arduino Nano (local.ino)

```
#include <DHT.h>
#include <Wire.h>
#include <LCD.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd=LiquidCrystal_I2C(0x27,16,2);
#define DHT11_PIN A0
#define DHTTYPE DHT11   // DHT 11
DHT dht(DHT11_PIN,DHTTYPE);

int led1 = A2;
int led2 = A3;
const int trigPin = 2;
const int echoPin = 3;
int motor1 = 6;
int motor2 = 9;
int motor3 = 10;
int motor4 = 11;
int FLAME =7;
int smoke= 5;
int buzzer = 12;
int sensorThres = 200;
int chk = 0;
int fire= 0;
int smoke1=0;
long duration;
int distance;
int cur_dis;
```

```
void setup()
{
   lcd.init();
   lcd.backlight();
   pinMode(smoke, INPUT);
   pinMode(buzzer, OUTPUT);
   pinMode(FLAME, INPUT);
   pinMode(trigPin, OUTPUT);
   pinMode(echoPin, INPUT);
   digitalWrite(6 , OUTPUT);
   digitalWrite(9 , OUTPUT);
   digitalWrite(10 , OUTPUT);
   digitalWrite(11 , OUTPUT);
   Serial.begin(9600);
   lcd.begin(16,2);
   lcd.clear();
   delay(1000);
   dht.begin();
   delay(1000);
   pinMode(led1,OUTPUT);
   digitalWrite(led1,LOW);
   cur_dis = 0;
}
void loop()
{
   float chk = dht.readTemperature();
   delay(1000);
   lcd.clear();
   lcd.setCursor(10,0);
   lcd.print("Temp");
   lcd.setCursor(10,1);
   lcd.print(chk);

   lcd.setCursor(0,0);
   lcd.print("Fire");
   lcd.setCursor(0,1);
   lcd.print("N.D");

   lcd.setCursor(6,0);
   lcd.print("Gas");
   lcd.setCursor(6,1);
   lcd.print("0%");

   digitalWrite(trigPin, LOW);
   delayMicroseconds(500);
   digitalWrite(trigPin, HIGH);
   delayMicroseconds(500);
   digitalWrite(trigPin, LOW);
   duration = pulseIn(echoPin, HIGH);

   distance= duration*0.034/2;
   Serial.print("Distance: ");
   Serial.println(distance);

   int fire = digitalRead(FLAME);// read FLAME sensor
   if(fire==LOW)
   {
      digitalWrite(motor1, LOW);
      digitalWrite(motor2, LOW);
      digitalWrite(motor3, LOW);
      digitalWrite(motor4, LOW);

      lcd.clear();
      lcd.setCursor(0,0);
```

```
    lcd.print("Fire Detected");
    digitalWrite(buzzer,HIGH); // set the buzzer ON

    delay(2000);
    digitalWrite(motor1, HIGH);
    digitalWrite(motor2, LOW);
    digitalWrite(motor3, LOW);
    digitalWrite(motor4, HIGH);

    delay(1000);
    digitalWrite(motor1, LOW);
    digitalWrite(motor2, LOW);
    digitalWrite(motor3, HIGH);
    digitalWrite(motor4, LOW);
    delay(1100);
}
if(fire==HIGH)
{
    digitalWrite(buzzer,LOW); // Set the buzzer OFF
    Serial.println("Peace");
}

int smoke1 = digitalRead(smoke);

if(smoke1==LOW)
{
    lcd.clear();
    lcd.setCursor(3,0);
    lcd.print("Gas/Smoke");
    lcd.setCursor(3,1);
    lcd.print("Detected.");
    digitalWrite(buzzer,HIGH); // Set the buzzer on
    delay(1000);
```

```
}

if (smoke1==HIGH)
{
   digitalWrite(buzzer,LOW); // Set the buzzer OFF
}

int x = cur_dis-distance;
if(cur_dis==distance || (x>=-5 && x<=5))
{
   digitalWrite(motor1, HIGH);
   digitalWrite(motor2, LOW);
   digitalWrite(motor3, LOW);
   digitalWrite(motor4, HIGH);
   delay(1000);

   digitalWrite(motor1, HIGH);
   digitalWrite(motor2, LOW);
   digitalWrite(motor3, LOW);
   digitalWrite(motor4, HIGH);
   delay(200);
   digitalWrite(motor1, LOW);
   digitalWrite(motor2, LOW);
   digitalWrite(motor3, HIGH);
   digitalWrite(motor4, LOW);
   delay(1100);
}
cur_dis = distance;

if(distance<30)
{
   digitalWrite(motor1, HIGH);
   digitalWrite(motor2, LOW);
```

```
        digitalWrite(motor3, LOW);
        digitalWrite(motor4, HIGH);
        delay(100);
        digitalWrite(motor1, LOW);
        digitalWrite(motor2, LOW);
        digitalWrite(motor3, HIGH);
        digitalWrite(motor4, LOW);
        delay(1100);
    }
    else
    {
         digitalWrite(motor1, LOW);
         digitalWrite(motor2, HIGH);
         digitalWrite(motor3, HIGH);
         digitalWrite(motor4, LOW);
         delay(100);
    }
}
```

Local Control Sketch for Node MCU (local_mcu.ino)

```
#define BLYNK_PRINT Serial
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#include <DHT.h>
char auth[] = "a6vAVS-9gVGX3-wVK2LGp__zNiRgknO4";
char ssid[] = "abcde";
char pass[] = "12345678";
#define DHTPIN 2          // what digital pin we're connected to

// uncomment whatever type you're using!
#define DHTTYPE DHT11     // DHT 11
//#define DHTTYPE DHT22   // DHT 22, AM2302, AM2321
//#define DHTTYPE DHT21   // DHT 21, AM2301
DHT dht(DHTPIN, DHTTYPE);
BlynkTimer timer;

void sendSensor()
{
  float h = dht.readHumidity();
  float t = dht.readTemperature(); // or dht.readTemperature(true) for Fahrenheit
  if (isnan(h) || isnan(t))
  {
    Serial.println("Failed to read from DHT sensor!");
    return;
  }
  Blynk.virtualWrite(V5, h);
  Blynk.virtualWrite(V6, t);
}
void setup()
{
  Serial.begin(9600);
  Blynk.begin(auth, ssid, pass);
  dht.begin();
  timer.setInterval(1000L, sendSensor);
}
void loop()
{
  Blynk.run();
  timer.run();
}
```

APPENDIX B

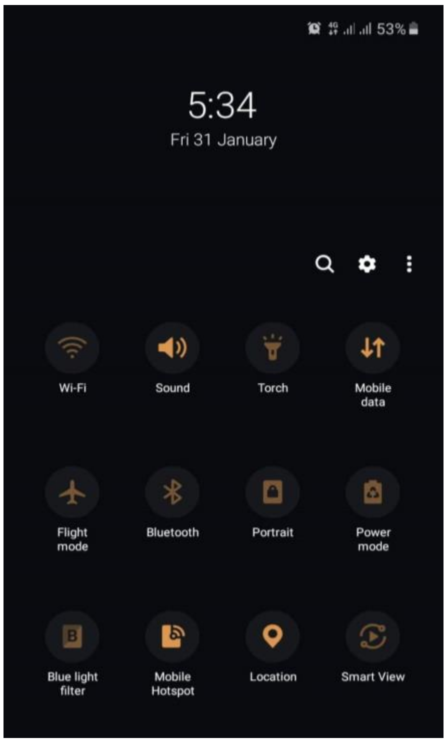
User Manual

To run this robot we need to do these following:

The first thing we have to do that we have to put users hotspot or if user uses wifi then wifi name and password into the robots Node MCU.

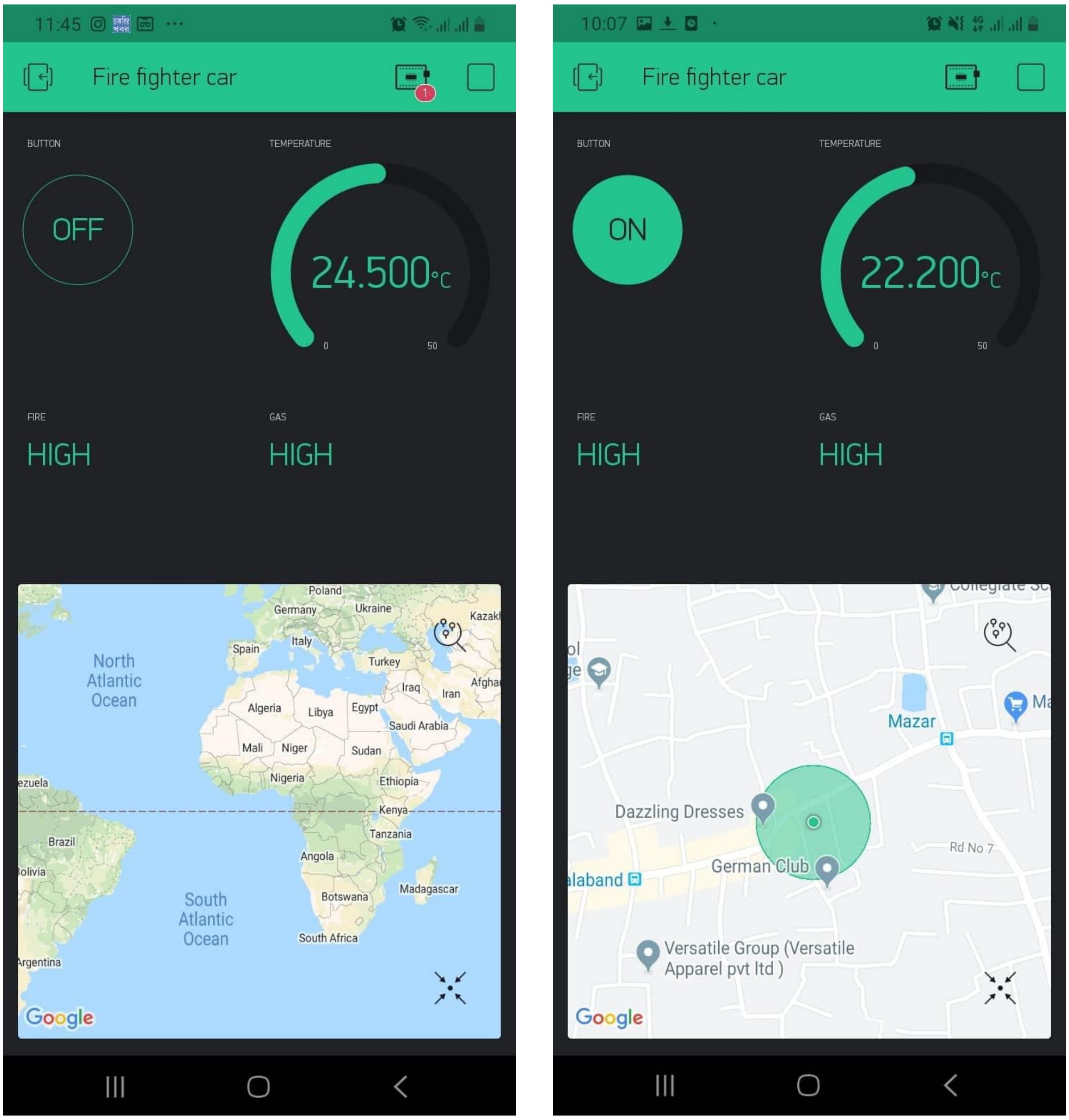
Then we have to follow steps given below:

1. First we have to install Blynk application on users phone to connect the robot.
2. Then to connect this robot with phone we have to turn on hotspot and mobile-data/wifi and location.



Connections Interface

3. After connecting with the robot through Blynk you can turn on and turn of the robot with Blynk and output of some sensors will be visible in the home screen of Blynk.



11:45

Fire fighter car

BUTTON

OFF

TEMPERATURE

24.500°C

0 50

FIRE

HIGH

GAS

HIGH

10:07

Fire fighter car

BUTTON

ON

TEMPERATURE

22.200°C

0 50

FIRE

HIGH

GAS

HIGH

Robot Controlling Interface

…The End…