EXPERIMENT-1

Digital I/O Interface – Blynk LED,Multicolour LED.

❖ AIM:

- **To interface and control RGB LED with Arduino.**

❖ APPARATUS REQUIRED:

S.No	Name of the equipment	Specifications	Quantity
1	Arduino	UNO	1
2	RGB common cathode LED		1
3	Jumper wires		required
4	Breadboard		1
5	Computer with Arduino IDE software	IDE 1.8.14	1

❖ PROCEDURE:

1. Build the circuit according to the schematic diagram.
2. Use Arduino Uno on the Arduino Desktop IDE.
3. Select board type and port.
4. Write and upload the program in the Arduino – IDE.

❖ BLYNK LED:

- **PROGRAMME:**

```
void setup(){
    //Initialize digital pin LED_BUILTIN as an output
    pinMode (LED_BUILTIN,OUTPUT);
}
// The loop function runs over and over again forever.
void loop(){
    digitalWrite (LED_BUILTIN,HIGH); //turn LED ON
    delay(1000);  //Wait for a second
    digitalWrite(LED_BUILTIN,LOW);  //turn LED OFF
    delay(1000);  //wait for a second
}
```

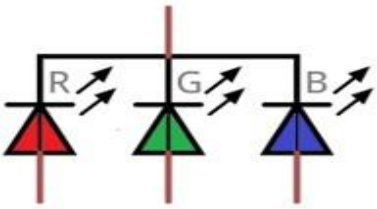
THEORY:

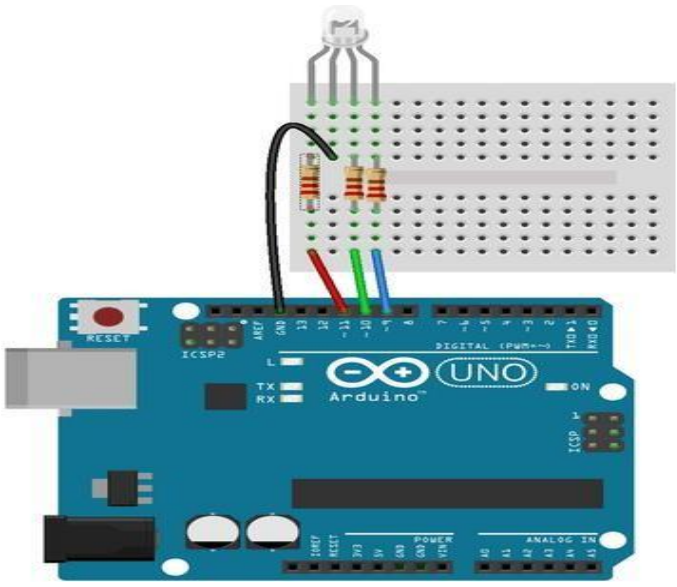
The RGB led consists of three different led's, red, green and blue. Many other colors can be obtained by mixing up these colors. The Arduino has aa analog write function which is used to obtain different colors for Arduino RGB led.

RGB LED Schematic

There are two types of RGB led's; the common cathode one and the common anode. In the common cathode RGB led, the cathode of all the led's is common and a PWM signals is fed to the anode of led's while in the common anode RGB led, the anode of all the led's is common and a PWM signals is fed to the cathode ofled's.

SCHEMATIC DIAGRAM:





❖ PROGRAMME:

```
int red = 2;
int green = 3;
int blue = 4;
void setup(){
        // initialize led pins as an output
```

```
        pinMode(2, OUTPUT);
        pinMode(3, OUTPUT);
        pinMode(4, OUTPUT);
}

// the loop function runs over and over again forever

void loop(){

        digitalWrite(red , 1); // turn the RED LED on (HIGH is the voltage level)

        digitalWrite(green, 0);

        digitalWrite(blue, 0);

        delay(1000); // wait for a second

        digitalWrite(red, 0);

        digitalWrite(green, 1);// turn the GREEN LED on (HIGH is the voltage level)

        digitalWrite(blue, 0);

        delay(1000);

        digitalWrite(red, 0);

        digitalWrite(green, 0);

        digitalWrite(blue, 1);// turn the BLUE LED on (HIGH is the voltage level)

        delay(1000);

}
```

❖ OBSERVATIONS:

S.No.	RGB Color Value	LED Color
1.	1, 0, 0	Red
2.	0, 1, 0	Green
3.	0, 0, 1	Blue

❖ RESULT:

Hence different Digital I/O Interfaces like blink led, Multicolor LED are connected and controlled withArduino board and corresponding outputs are observed.

EXPERIMENT-2

Digital I/O Interface - IR Sensor, Slot Sensor

AIM:

- **To Interface a IR with Arduino to detect motion and display "Stop" or "Go" depending on the outputfrom the sensor.**
- **To Interface a slot sensor with Arduino to detect object and control LED based on sensor output.**

APPPARATUS REQUIRED:

S.No	Name of the equipment	Specifications	Quantity
1	Arduino	UNO	1
2	IR Sensor		1
3	Slot Sensor		1
4	Jumper wires		required
5	Breadboard		1
6	Computer with Arduino IDE software	IDE 1.8.14	1

PROCEDURE:

1) Build the circuit according to the schematic diagram
2) Use Arduino Uno on the Arduino Desktop IDE.
3) Select board type and port.
4) Write and upload the program in the Arduino-IDE

IR SENSOR:

An Infrared sensor is an electronic module which is used to sense certain physical appearance of its surroundings by either emitting and/or detecting infrared radiation. IR sensors are also capable of determining the heat being emitted by an object and detecting motion.

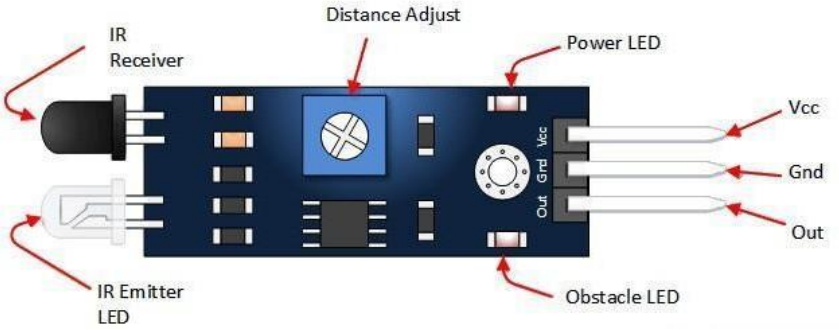
IR SENSOR WORKING:

Here an IR sensor is used for detecting obstacles. IR transmitter transmits IR signal, as that signal detects any obstacle in its path, the transmitted IR signal reflects back from the obstacle and received by the receiver.

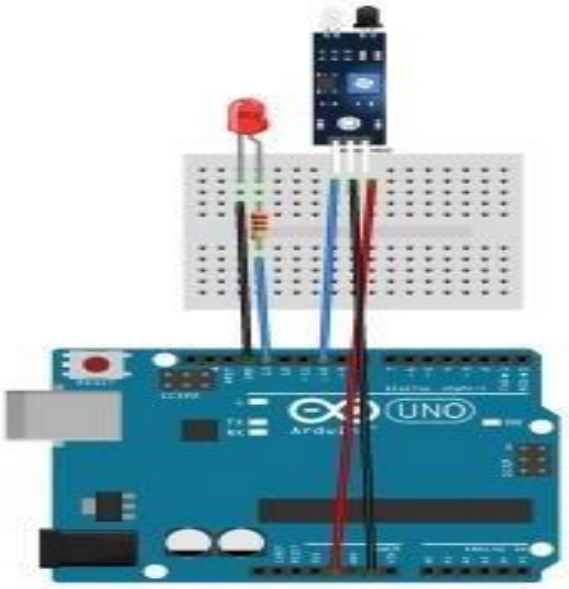
PINOUT:

1. VCC: 3.3V-5V power input pin
2. GND: 0V power pin

3. OUT: Digital Output Pin



SCHEMATIC DIAGRAM:



PROGRAMME:

```
int ir = 2;  // connect ir sensor to arduino pin 2
int led = 3;  // connect Led to arduino pin 3
void setup(){
        pinMode(ir, INPUT);  // sensor pin INPUT
        pinMode(led, OUTPUT);  // Led pin OUTPUT
}
void loop(){
        int statusSensor = digitalRead(ir);
        if(statusSensor == 1){
                digitalWrite(led, 0);  // LED LOW
        }
```

```
        else{
            digitalWrite(led, 1);  // LED HIGH
        }
    }
```

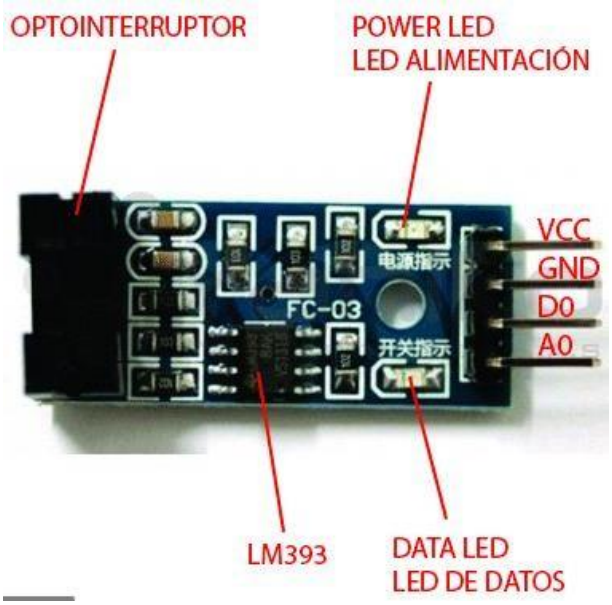
OBSERVATIONS:

S.No.	OBSTACLE	LED Status
1.	YES	ON
2.	NO	OFF

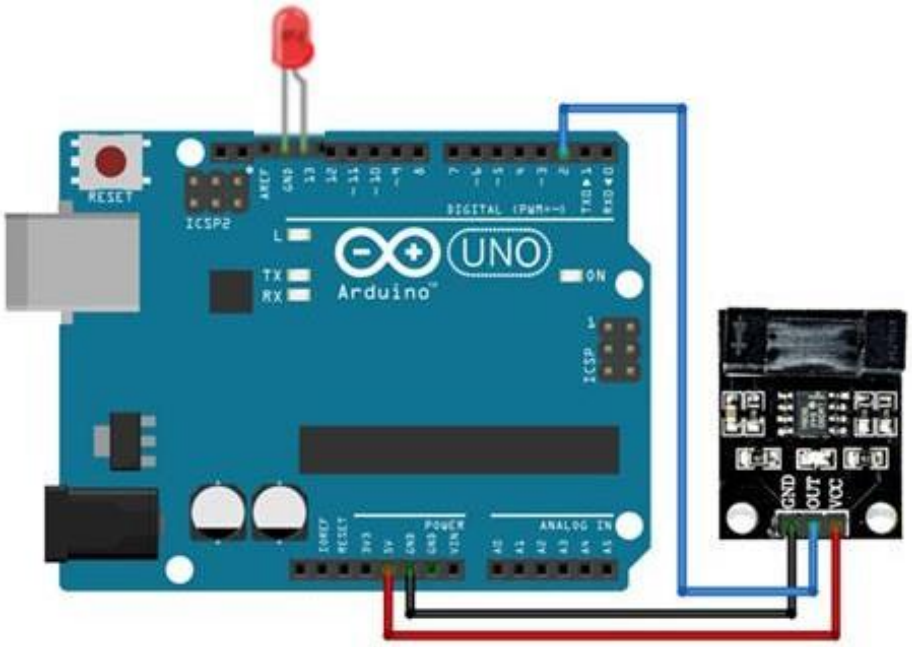
LM393 Speed Sensor Module (H206)

H206 speed sensor module is composed of infrared light sensor and LM393 voltage comparator IC, soit is named as LM393 speed sensor. The module also includes a grid plate, which must be mounted on the rotating shaft of the motor.

The infrared light sensor is composed of infrared LED and phototransistor, separated by a small gap. As shownbelow, the entire sensor device is placed in a black housing. The grid plate is composed of grooves, and the plate is arranged between the gaps of the infrared light sensors so that the sensor can sense the gap in the grid plate. Each gap in the grid plate triggers the IR sensor when passing through the gap; a comparator is then used to convert these triggers into a voltage signal. The comparator uses ON Semiconductor's LM393 IC. The module has three pins, two of which are used to power the module, and one output pin is used to count the number of triggers.



SCHEMATIC DIAGRAM:



PROGRAM:

```
int slot Sensor = 2;
int led = 13;
void setup() {
        // put your setup code here to run once:
        pinMode(slotSensor,INPUT);
        pinMode(led, 13);
}
void loop() {
        // put your main codes here, to run repeatedly:
        int value = digitalRead(slotSensor);
        if(value == 1){
                digitalWrite(led, 1);
        }
        else{
                digitalWrite(led, 0);
        }
}
```

OBSERVATIONS:

S.No.	OBSTACLE	**LED Status**
1.	YES	ON
2.	NO	OFF

RESULT:

Hence different Digital I/O Interfaces like IR Sensor and Slot Sensor connected and controlled withArduino board and corresponding outputs are observed.

EXPERIMENT-3

Analog Read and Write - Potentiometer, Led Brightness Control.

AIM:

- **To control LED brightness using potentiometer with Arduino**

APPARATUS REQUIRED:

S.No	Name of the equipment	Specifications	Quantity
1	Arduino	UNO	1
2	Potentio Meter	1K Ω	1
3	LED		1
4	Resistor	220ohms	1
5	Jumper wires		required
6	Breadboard		1
7	Computer with Arduino IDE software	IDE 1.8.14	1

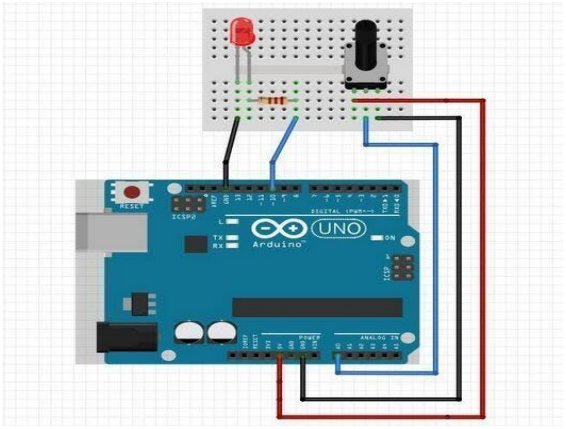
THEORY:

The analog input is a pin with ADC (Analog-to-Digital Converter) function. It can convert the externally input analog signal into a digital signal that can be recognized during chip operation, so as to realize the function of reading in the analog value. The Arduino analog input function has 10-bit precision, that is, it can convert a voltage signal of 0 to 5V into an integer form of 0 to 1024. Utilize the analogRead() function to read input voltage values by the potentiometer, and then use the analogWrite() function to control the brightness of the LED light.

PROCEDURE:

1) Build the circuit according to the schematic diagram
2) Use Arduino Uno on the Arduino Desktop IDE.
3) Select board type and port.
4) Write and upload the program in the Arduino-IDE.
5) By varying the potentiometer observe the LED intensity.

SCHEMATIC DIAGRAM:



PROGRAMME:

//POTENTIOMETER

```
void setup(){
        // put your setup code here, to run once:
        Serial.begin(9600);
}
void loop(){
        // put your main code here, to run repeatedly:
        int value = analogRead(A0);
        Serial.println(value);
        delay(10);
}
```

// LED BRIGHTNESS CONTROL

```
int led = 11;
int value = 0;
int ledvalue = 0;
void setup() {
        // put your setup code here, to run once:
        pinMode(led, OUTPUT);
}
void loop() {
        // put your main code here, to run repeatedly:
        Value = analogRead(A0);
        ledvalue = map(value, 0, 1024, 0, 255);
        analogWrite(led, ledvalue);
}
```

OBSERVATIONS:

S.No.	Potentiometer Value (K Ω)	LED Status
1.	Min	ON with high brightness
2.	Mid Value	ON with less brightness
3.	Max	OFF

RESULT:

Hence

➢ Observed the resistance values of potentiometer and display in serial monitor.

➢ LED brightness has been controlled using potentiometer.

EXPERIMENT- 4

Analog Read and Write - Temperature Sensor

Aim:

Analog Read and Write - Temperature Sensor

APPARATUS REQUIRED:

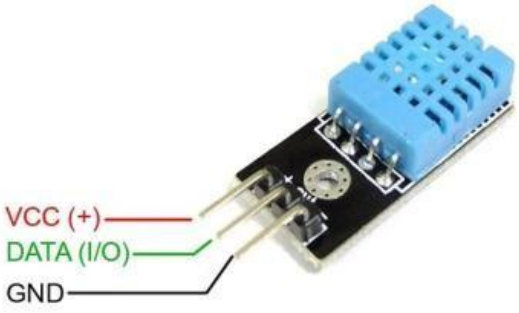
S.No	Name of the equipment	Specifications	Quantity
1	Arduino	UNO	1
2	Temperature Sensor	DTH11	1
3	Jumper wires		required
4	Breadboard		1
5	Computer with Arduino IDE software	IDE 1.8.14	1

THEORY:

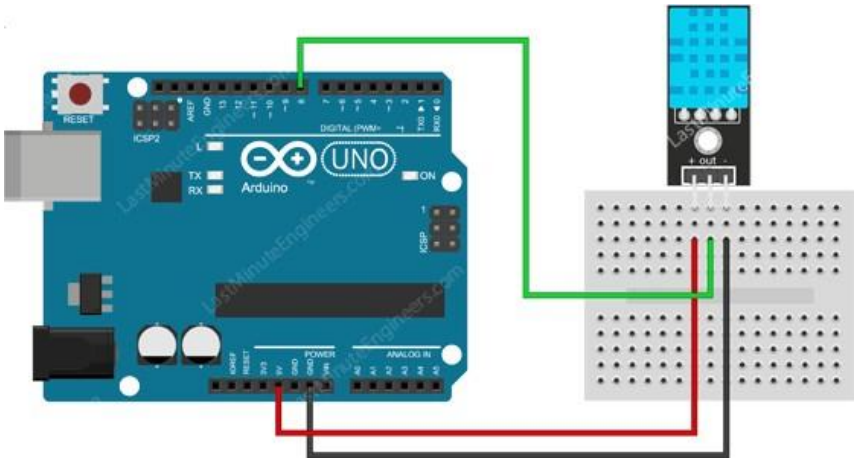
The analog input is a pin with ADC (Analog-to-Digital Converter) function. It can convert the externally input analog signal into a digital signal that can be recognized during chip operation, so as to realize the function of reading in the analog value. The Arduino analog input function has 10-bit precision, that is, it can convert a voltage signal of 0 to 5V into an integer form of 0 to 1024. Utilize the analogRead() function to read input voltage values by the potentiometer, and then use the analogWrite() function to control the brightness of the LED light.

Temperature sensor DTH11:

The digital temperature and humidity sensor DHT11 is a composite sensor that contains a calibrated digital signal output of temperature and humidity. The technology of a dedicated digital modules collection and the temperature and humidity sensing technology are applied to ensure that the product has high reliability and excellent long-term stability. The sensor includes a resistive sense of wet component and an NTC temperature measurement device, and is connected with a high-performance 8-bit microcontroller.



SCHEMATIC DIAGRAM:



PROGRAMME:

```
#include<dht.h>
dht DHT;
#define DHT11_PIN 8
void setup() {
        Serial.begin(9600);
}
void loop() {
        int chk = DHT.read11(DHT11_PIN);
        Serial.print("Temperature= ");
        Serial.println(DHT.temperature);
        Serial.print("Humidity= ");
        Serial.println(DHT.humidity);
        delay(1000);
}
```

OBSERVATIONS:

S.No.	Temperature	Humidity
1		
2		
3		

RESULT:

➢ Temperature and Humidity has been measured using Temperature sensor and displayed using serial monitor.

EXPERIMENT- 5

DC MOTOR CONTROL - DC MOTOR SPEED AND DIRECTION CONTROL

AIM:

To control DC Motor Speed and Direction using Arduino.

APPARATUS REQUIRED:

S.No	Name of the equipment	Specifications	Quantity
1	Arduino	UNO	1
2	DC Motor	12V High Torque	1
3	L298N Motor driver		1
4	Potentio Meter	10K Ω	1
5	Resistor	220ohms	1
6	Sparkfun push button Switch		1
7	Rechargeable Battery	5V	1
8	Jumper wires		required
9	Breadboard		1
10	Computer with Arduino IDE software	IDE 1.8.14	1

THEORY:

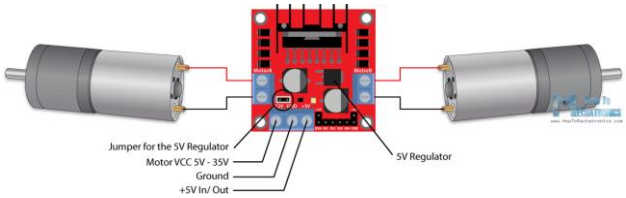
The L298N is a dual-channel H-Bridge motor driver capable of driving a 2x DC motors, making it ideal for building two-wheel robots.

Power Supply: From *'Vs' pin* the H-Bridge gets its power for driving the motors which can be 5 to 35V. *'Vss'* is used for driving the logic circuitry which can be 5 to 7V. And they both sink to a common ground named *'GND'*.

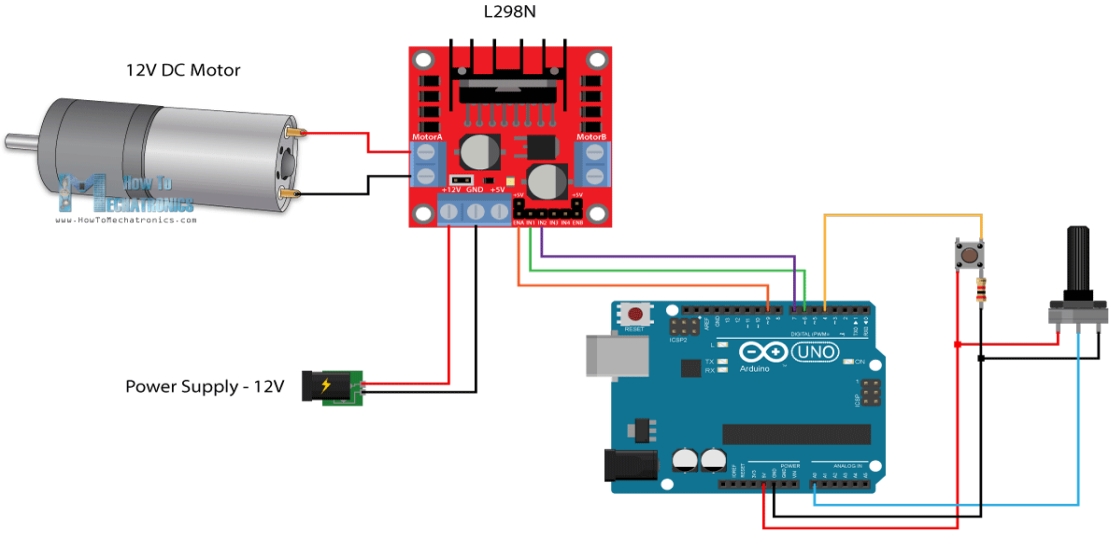
Output Pins: The L298N motor driver's output channels for the motor A and B are broken out to the edge of the module with two 3.5mm-pitch screw terminals.

Direction Control Pins: The IN1 and IN2 pins control the direction of the motor A while IN3 and IN4 control the direction of the motor B.

Speed Control Pins: ENA and ENB are used to turn the motors ON, OFF and control its speed. The module usually comes with a jumper on these pins. When this jumper is in place, the motor is enabled and spins at maximum speed. If you want to control the speed of motors, you need to remove the jumpers and connect themto PWM-enabled pins on Arduino.



SCHEMATIC DIAGRAM:



PROCEDURE:

1) Build the circuit according to the schematic diagram
2) Use Arduino Uno on the Arduino Desktop IDE.
3) Select board type and port.
4) Write and upload the program in the Arduino-IDE.
5) Control the speed and direction of DC motor by varying potentiometer and switch.

PROGRAMME:

```
//DC motor direction and speed control
int in1 = 6;
int in2 = 7;
int enA = 9;
int Speed = 0;
int val = 0;
void setup() {
        Serial.begin(9600);
        pinMode(enA, OUTPUT);
        pinMode(in1, OUTPUT);
        pinMode(in2, OUTPUT);
        Serial.begin(9600);
}

//Enter value 0 or 1 to select motor direction, 0 for Backward, 1 for Forward
```

```
void loop(){
        val = analogRead(A0); // Read potentiometer value to change the motor speed
        Speed = map(val, 0, 1023, 0, 255);
        // Anti clockwise rotation
        if(Serial.available() > 0){
                char state = Serial.read();
                //FORWARD Rotation
                if(state == '1'){
                        analogWrite(enA, Speed);
                        digitalWrite(in1, 1);
                        digitalWrite(in2, 0);
                        Serial.println("FORWARD");
                        delay(1000);
                }
                //BACKWARD Rotation
                if(state == '0'){
                        analogWrite(enA, Speed);
                        digitalWrite(in1, 0);
                        digitalWrite(in2, 1);
                        Serial.println("BACKWARD");
                        delay(1000);
                }
        }
}
```

OBSERVATIONS:

By varying the potentiometer, speed of the DC motor is controlled and each time button is pressed, itchanges the rotation direction of the motor.

RESULT:

Hence DC Motor Speed and Direction has been controlled using Arduino.

EXPERIMENT-6

SERIAL COMMUNICATION - DEVICE CONTROL

AIM:

To control LED using a serial monitor in Arduino IDE.

APPARATUS REQUIRED:

S.No	Name of the equipment	Specifications	Quantity
1	Arduino	UNO	1
2	LED		1
3	Jumper wires		required
4	Breadboard		1
5	Computer with Arduino IDE software	IDE 1.8.14	1

THEORY:

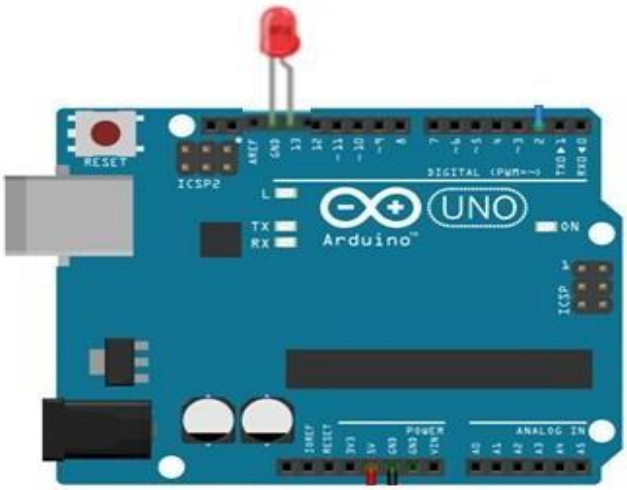
A Serial communication is a method of transmitting data as a train of 1s and 0s, i.e. only one bit for a clock cycle of binary coded data. Or one bit at a time, sequentially on a communication channel. It is a contrast to parallel communication, whereas in parallel system multiple bits are sent through several parallel paths. In a parallel system number of bits transmitted per clock is equal to the number of parallel paths. That is, a 4-bit channel can transmit any of the 16 binary states for a single clock cycle. In a serial communication to transfer abyte, the data is transferred as a sequence of 8 bits as one by one. Each bit is either high state 1s or low state 0s.

In Arduino boards, the serial connection can be made either via serial port (type B USB) or by digital pins 0 (RX) and 1 (TX). An Arduino IDE includes a serial monitor, a built-in terminal to communicate with an Arduino board.

PROCEDURE:

1) Build the circuit according to the schematic diagram
2) Use Arduino Uno on the Arduino Desktop IDE.
3) Select board type and port.
4) Write and upload the program in the Arduino-IDE.
5) open the serial monitor by,
    a. Press ctrl+shift+M
    b. Tools/Serial Monitor
    c. Or click the magnifier icon in the top right corner of the Arduino IDE.
6) Use Decimal 1 for LED ON and decimal 0 for LED OFF

SCHEMATIC DIAGRAM:



PROGRAMME:

```
//Serial LED control
void setup() {
        pinMode(13, OUTPUT);
        Serial.begin(9600);
}
//Enter value 0 or 1
void loop() {
        if(Serial.available() > 0){
                char state =  Serial.read();
                if(state == '1'){
                        digitalWrite(13, HIGH);
                        Serial.println("LED ON");
                }
                if(state == '0'){
                        digitalWrite(13, LOW);
                        Serial.println("LED OFF");
                }
        }
        delay(100);
}
```

OBSERVATIONS:

S.No.	Input from Serial Monitor	LED Status
1.	0	OFF
2.	1	ON

RESULT:

Hence LED has been controlled using commands from serial monitor.

EXPERIMENT-7

FABRICATION AND DIRECTION CONTROL OF WHEELED ROBOT USING ARDUINO

AIM:

To fabricate wheeled robot and control its direction using Arduino.

APPPARATUS REQUIRED:

S.No	Name of the equipment	Specifications	Quantity
1	Arduino	UNO	1
2	Motor driver	LM298	1
3	Wheel Robo set		1
4	Jumper wires		required
5	Rechargeable Battery 9-12 v		1
6	Bread board		1
9	Computer with Arduino IDE software	IDE 1.8.14	1

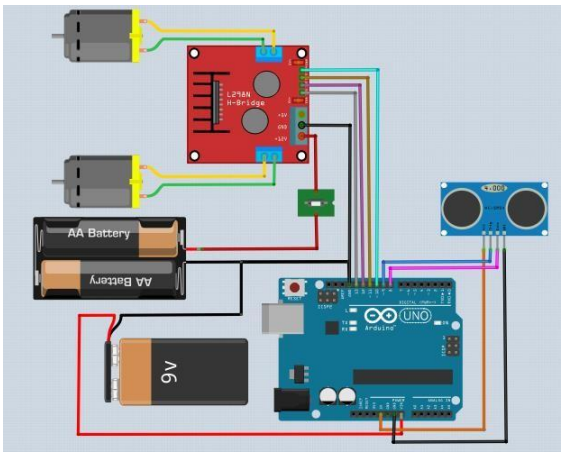
THEORY:

Obstacle Avoiding Robot is an intelligent device that can automatically sense the obstacle in front of it and avoid them by turning itself in another direction. This design allows the robot to navigate in an unknown environment by avoiding collisions.

PROCEDURE:

1) Assemble the circuit as shown in the schematic given below.
2) Use Arduino Uno on the Arduino Desktop IDE.
3) Select board type and port.
4) Write and upload the program in the Arduino-IDE.
5) Turn on the toggle switch and watch the robot avoiding obstacles.

SCHEMATIC DIAGRAM:



PROGRAMME:

```
int in1 = 3;
int in2 = 4;
int in3 = 5;
int in4 = 6;
int val = 0;
void setup() {
        // put your setup code here, to run once:
        Serial.begin(9600);
        pinMode(in1, OUTPUT);
        pinMode(in2, OUTPUT);
        pinMode(in3, OUTPUT);
        pinMode(in4, OUTPUT);
}
void loop() {
        // put your main code here, to run repeatedly:
        if(Serial.avaiable() > 0){
                val = Serial.read();
                delay(5);
                if(val == 'w'){
                        digitalWrite(3,1);
                        digitalWrite(4,0);
                        digitalWrite(5,0);
                        digitalWrite(6,1);
                        Serial.println("Forward");
                }
                if(val == 'a'){
                        digitalWrite(3,0);
                        digitalWrite(4,1);
                        digitalWrite(5,0);
                        digitalWrite(6,1);
                        Serial.println("Left");
                }
                if(val == 'd'){
                        digitalWrite(3,1);
                        digitalWrite(4,0);
                        digitalWrite(5,1);
```

```
                digitalWrite(6,0);
                Serial.println("Right");
        }
        if(val == 's'){
                digitalWrite(3,0);
                digitalWrite(4,1);
                digitalWrite(5,1);
                digitalWrite(6,0);
                Serial.println("Backward");
        }
        if(val == 'x'){
                digitalWrite(3,1);
                digitalWrite(4,1);
                digitalWrite(5,1);
                digitalWrite(6,1);
                Serial.println("Stop");
        }
    }
}
```

OBSERVATIONS:

w- Forward direction

a- Left direction

d- right direction

s- Backward direction

RESULT:

Hence wheeled robot is fabricated and controlled its direction using Arduino to avoid collisions.

EXPERIMENT-8

WIRELESS MODULE INTERFACE –Wi-Fi

AIM:

- control LED using Wi-Fi module interface with Arduino.

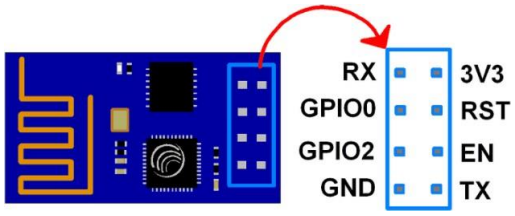
APPARATUS REQUIRED:

S.No	Name of the equipment	Specifications	Quantity
1	Arduino	UNO	1
2	Wi-Fi Module	ESP8266	2
3	LED		1
4	Jumper wires		required
5	Breadboard		1
6	Computer with Arduino IDE software	IDE 1.8.14	1
7	Mobile with Bluetooth Terminal App		

WI-FI MODULE INTERFACE:

WI-FI (ESP8266) MODULE:

Wireless Fidelity is a term used for products which uses any type of 802.11 technologies. Wi-Fi network operate with 11 Mbps or 54 Mbps data rate in the unlicensed 2.4 GHz and 5GHz radio frequency band. Devices which have Wi-Fi enabled can sent and receive data wirelessly from locations which are equipped with wireless access. Access points which are located in a Wi-Fi location transmit RF signal for the Wi-Fi enabled devices. These Wi-Fi enabled devices can receive the signal if they are located within access point range. The speed of data transmission depends upon the speed of pipeline fed into the access point.



3V3: - 3.3 V Power Pin.

GND: - Ground Pin.

RST: - Active Low Reset Pin.

EN: - Active High Enable Pin.

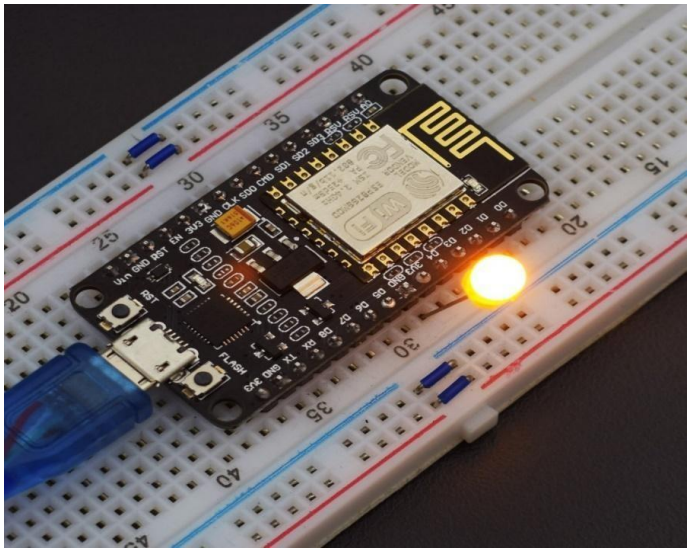
TX: - Serial Transmit Pin of UART.

RX: - Serial Receive Pin of UART.

GPIO0 & GPIO2: - General Purpose I/O Pins. These pins decide what mode (boot or normal) the module starts up in. It also decides whether the TX/RX pins are used for Programming the module or for serial I/O purpose.

To program the module using UART, Connect GPIO0 to ground and GPIO2 to VCC or leave it open. To use UART for normal Serial I/O leave both the pins open (neither VCC nor Ground).

SCHEMATIC DIAGRAM:



PROCEDURE:

1) Build the circuits according to the schematic diagrams
2) Use Arduino Uno on the Arduino Desktop IDE.
3) Select board type and port for transmitter and receiver.
4) Write and upload the code to the Arduino-IDE and then to Arduino board.
5) Control LED by using mobile through wifi.

PROGRAMME:

```
#include<ESP8266WiFi.H>
const char* ssid = " ";  // Your WiFi Name
const char* password = " ";  // Your WiFi password
int LED = 2;  // led connected to GPIO2(D4)
WiFiServer server(80);
void setup() {
        Serial.begin(115200);  //Default Baudrate
        pinMode(LED, OUTPUT);
        digitalWrite(LED, LOW);
        Serial.print("Connecting to the Network");
```

```
        WiFi.begin(ssid, password);
        while(WiFi.status() != WL_CONNECTED){
                delay(1000);
                Serial.print(".");
        }
        Serial.println("WiFi connected");
        Server.begin();  //Starts the server
        Serial.println("Server started");
        Serial.print("IP Address of network: ");  // will IP address on Serial Monitor
        Serial.println(WiFi.localIP());
        Serial.print("Copy and paste the following URL: https://");  //will print IP address URL
        Serial.print(WiFi.localIP());
        Serial.println("/");
}
void loop() {
        WiFiClient client = server.available();
        if (!client){
                return;
        }
        Serial.println("Waiting for new client");
        while(!client.available()){
                delay(1);
        }
        String request = client.readStringUntil('\r');
        Serial.println(request);
        Client.flush();
        int value = LOW;
        if(request.indexOf("LED = ON") != -1){
                digitalWrite(LED, HIGH); // Turn LED ON
                value = HIGH;
        }
        if(request.indexOf("LED = OFF") != -1){
                digitalWrite(LED, LOW); // Turn LED OFF
                value = LOW;
```

```
        }
        //*-----------------HTML Page Code----------------*//
        client.println("HTTP/1.1 200 OK");
        client.println("Content-Type: text/html");
        client.println(" ");
        client.println("<!DOCTYPE HTML>");
        client.println("html");
        client.print("CONTROL LED: ");
        if(value == HIGH){
            client.print("ON");
        }
        else{
            client.print("OFF");
        }
        client.println("<br><br>");
        client.println("<a href=\"/LED=ON\"\"><button>ON</button></a>");
        client.println("<a href=\"/LED=OFF\"\"><button>OFF</button></a><br/>");
        client.println("</html>");
        delay(1);
        Serial.println("Client disconnected");
        Serial.println(" ");
    }
```

OBSERVATION :

copy and paste the URL on your mobile, your light **ON or OFF** is controlled by mobile.

RESULT:

Hence

- LED has been controlled using Wi-Fi module interface with Arduino.

EXPERIMENT-9

BASIC ANDROID APP DEVELOPMENT USING MIT APP INVENTOR

AIM:

- To develop Basic Android App Development using MIT App Inventor

APPARATUS REQUIRED:

S.No	Name of the equipment	Specifications	Quantity
1	NodeMCU	ESP8266	1
2	Computer with Arduino IDE software	IDE 1.8.14	1
3	Mobile phone with wi-fi connectivity		1

PROCEDURE:

1. First go to the **MIT Application Inventor** website: http://ai2.appinventor.mit.edu/
2. Then click '**Create Applications**' in the top left corner
3. Now click on '**Projects**'on the next screen and then '**Start a new project**'.
4. Now click on '**Button**'and drag and drop two buttons on the main screen. You can enter your favoritename in the button from the options on the right.
5. Then click '**Connectivity**'and drag and drop the web component to the main screen.
6. Click '**Blocks** 'now to add blocks to your application.
7. Now click on **button 1 in the block menu** and then **click on the marked red option**.
8. After this click on **Web 1**. Scroll down and select the red marked block.
9. Now click on the **text menu and choose the first option**. Enter your **URL** in the text menu.
10. Then click on **Web 1 again and then select the marked red option.**
11. Follow the same procedure for '**Button 2′**.
12. Now that the **app** is ready to download, click on 'Build' to get the simple apk file. Also, there are twooptions to **download the app APK, by QR code and directly on PC, then install it on Android.**
13. Now your app is ready, and you can control the lighting using the **ON-OFF** button presented in the app.
14. Now we have to upload the code to NodeMCU to create a simple HTTP web server for controlling home applications. We will use the HTTP GET method for communicating between ESP8266 and Android applications.

PROGRAMME:

```
#include<ESP8266WiFi.h>
const char* ssid = "xxxxxxxx";
const char* password = "xxxxxxxxxx";
```

Serial Monitor is started at the default Baud Rate for NodeMCU

```
Serial.begin(115200);
```

Relay Pin is defined to NodeMCU D4 pin i.e. GPIO pin 2.

```
pinMode(2, OUTPUT);
digitalWrite(2, 0);
```

In the void setup function, the function will try to connect to WiFi. This process executes in a loop, whichmeans it runs until there is a connection to WiFi. So be careful before entering your WiFi SSID and Password.

```
void setup() {
        // Connect to WiFi network
        Serial.println();
        Serial.println();
        Serial.print("Connecting to ");
        Serial.println(ssid);
        WiFi.begin(ssid, password);
        while(WiFi.status() != WL_CONNECTED){
                delay(500);
                Serial.print(".");
        }
        Serial.println(" ");
        Serial.println(" WiFi connected");
}
```

In void loop, it check if a client has connected. It Wait until the client sends some data and performs tasksaccording to input.

```
void loop() {
        WiFiClient client = server.available();
        if(!client){
                return ;
        }
        Serial.println("new client");
        while(!client.available()){
                delay(1);
        }
        String req = client.readStringUntil('\r');
}
```

Now navigate through your browser to check if your web server is working perfectly. Use the following URLsto turn your light **ON or OFF**.

http://192.168.1.40/gpio/1

http://192.168.1.40:/gpio/0

***Note: 192.168.1.40** is the **IP address of NodeMCU**. You can find the IP address of your NodeMCU on **Serial Monitor**. When you **run the code on the Arduino IDE**, it prints your device's IP address on the serial monitor.Therefore, **it will confirm whether the webserver is working or not**.*

OBSERVATION:

RESULT:

Hence Android App is developed using MIT App Inventor

EXPERIMENT-10

SMART HOME ANDROID APP DEVELOPMENT USING APP INVENTOR AND ARDUINO

AIM:

To develop Home Automation with MIT App Inventor and ESP8266.

APPARATUS REQUIRED:

S. No	Name of the equipment	Specifications	Quantity
1	NodeMCU	ESP8266	1
2	RGB common cathode LED/Lamp		1
3	5V relay		1
4	Jumper wires		
5	Breadboard		1
6	Computer with Arduino IDE software	IDE 1.8.14	1
7	Mobile phone with wi-fi connectivity		1

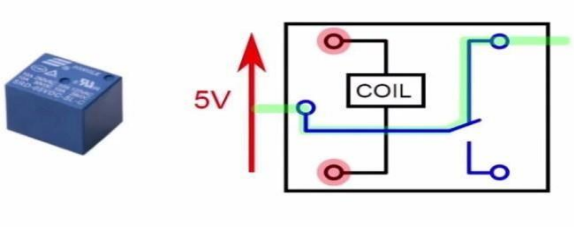
PROCEDURE:

1) Build the circuit according to the schematic diagram
2) Use Arduino Uno on the Arduino Desktop IDE.
3) Select board type and port.
4) Write and upload the program in the Arduino-IDE.

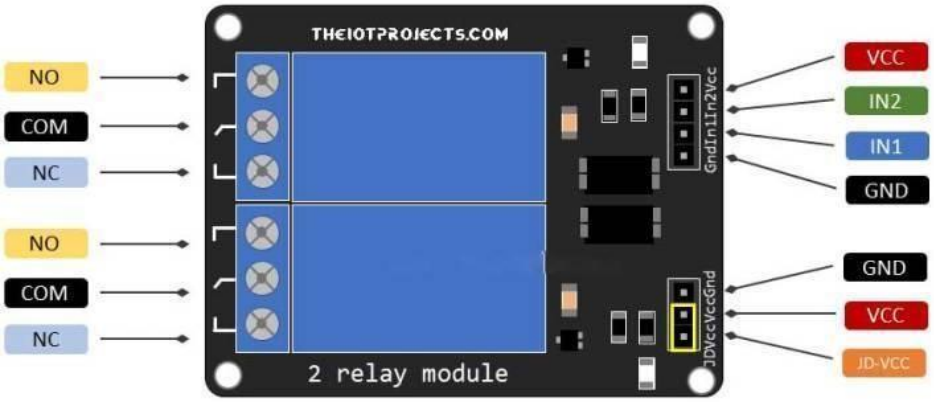
THEORY:

Relay Module

A relay is an electrically operated switch. Relays are used for controlling multiple circuits by one signal. So using relays we can turn the circuit “ON” and “OFF” with electricity. The relay is controlled by a small current (5V) and can switch “ON” and “OFF” the large current. Typically, the relay has five terminals asshown below:



When no voltage is applied to the coil, the COM terminal will be connected to the NC (Normally Closed) terminal. And when voltage is applied to the coil, an electromagnetic field is produced that attracts the armature, and a COM and NO (Normally Open) terminal connection is made, which allows for very large currents.



A small driver circuit with a **transistor, diode, and a resistor** is used to configure relay. The transistor is used to amplify the current, the resistor is used to provide **BIAS** to the transistor, and if the transistor is off, the **diode** is used to stop the reverse current flow. Here we have used the 5V relay module for demonstration.

SCHEMATIC DIAGRAM:

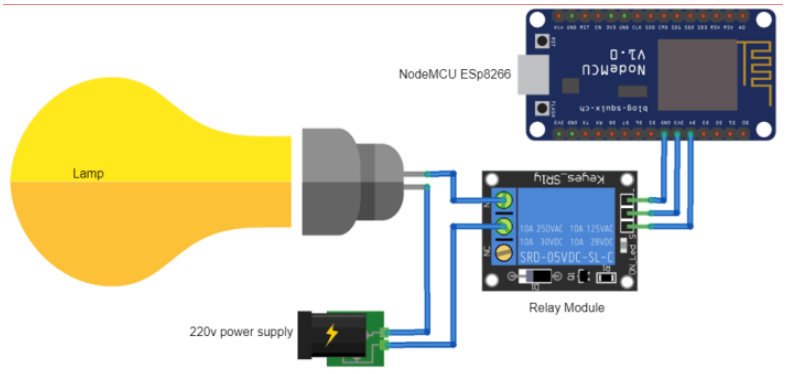


Fig: Wiring of Relay with Node MCU to Control Home Appliance.

NodeMCU	Relay
Vcc	Vcc
GND	GND
D4	Input

Pin connection of relay with NodeMCU ESP8266

Now we have to make an Android APP using MIT App Inventor for controlling home appliances.

PROGRAMME:

```
#include<ESP8266WiFi.h>
const char* ssid = “Enter Your Wifi Name”;
const char* password = “Enter Your WiFi password”;
WiFiServer server(80);
void setup() {
        Serial.begin(115200);  // Default Baud Rate for NodeMCU
        delay(10);
        pinMode(2, OUTPUT);  // Connect Relay to NodeMCU’s D4 pin
        digitalWrite(2, 0);
        //Connect to WiFi network
        Serial.println();
        Serial.println();
        Serial.print(“Connecting to ”);
        Serial.println(ssid);
        WiFi.begin(ssid, password);
        while(WiFi.status() != WL_CONNECTED){
                delay(500);
                Serial.print(“.”);
        }
        Serial.println(“”);
        Serial.println(“WiFi connected”);
        //Start the server
        server.begin();
        Serial.println(“Server started”);
        // Print the IP address
        Serial.println(WiFi.localIP());
}
void loop() {
        //Check if a client has connected
        WiFiClient client = server.available();
        if(!client){
```

```
            return;
        }
        //Wait until the client sends some data
        Serial.println("new client");
        while(!client.available()){
            delay(1);
        }
        String req = client.readStringUntil('\r');
        Serial.println(req);
        Client.flush();
        int val;
        if(req.indexOf("/gpio/0") != -1){
            val = 0;
        }
        else if(req.indexOf("/gpio/1") != -1){
            val = 1;
        }
        else{
            Serial.println("invalid request");
            Client.stop();
            return;
        }
        //Set GPIO2 according to the request
        digitalWrite(2, val);
        client.flush();
        String s = "HTTP/1.1 200 OK\r\nContent-Type: text/html\r\n\r\n<!DOCTYPE
        HTML>\r\n<html>\r\nGPIO is now ";
        s += (val)? "high": "low";
        s += "</html>"
}
```

OBSERVATION:

RESULT:

Hence developed Smart Home Android App Development using App Inventor and Arduino.

EXEPERIMENT-11

WIRELESS MODULE INTERFACE – BLUETOOTH

AIM:

- **Transfer data using Bluetooth with Arduino**
- **Control LED at Receiver through Switch at Transmitter using Wi-Fi module interface with Arduino.**

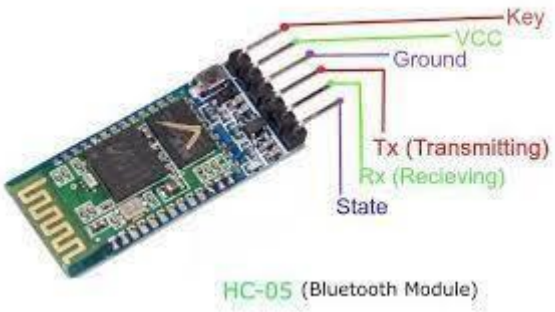
APPPARATUS REQUIRED:

S.no	Name of the equipment	Specifications	Quantity
1	**Arduino**	**UNO**	1
2	**Bluetooth Module**	**HC-05**	1
3	**Sparkfun push button Switch**		1
4	**LED**		1
5	**Jumper wires**		**required**
6	**Breadboard**		1
7	**Computer with Arduino IDE software**	**IDE 1.8.14**	
8	Mobile with Blutooth Terminal App		

BLUETOOTH MODULE INTERFACE:

BLUETOOTH:

Bluetooth is a one of the wireless technology. HC-05 is a Bluetooth module whichcancommunicate in two way. It is full-duplex. It be used with most micro controllers. Becauseitoperates Serial Port Protocol (SSP). The module communicate with the help of USART(Universal Synchronous/ Asynchronous Receiver/Transmitter) at the baud rate of 9600anditalso support other baud rate. This module can be interfaced with any microcontroller whichsupports USART. The HC-05 operates in two modes. One is Data mode and other isATcommand mode. When the Enable/Key pin is "LOW" the HC-05 is in Data Mode. If that pinset as "HIGH" the module is in AT command mode.



HC-05 (Bluetooth Module)

Enable - This pin is used to set the Data Mode or and AT command mode (set high).

VCC - This is connected to +5V power supply.
Ground - Connected to ground of powering system.
Tx (Transmitter) - This pin transmits the received data Serially.
Rx (Receiver) - Used for broadcasting data serially over bluetooth.
State -Used to check if the bluetooth is working properly.

PROCEDURE:

1) Build the circuit according to the schematic diagram

2) Use Arduino Uno on the Arduino Desktop IDE.

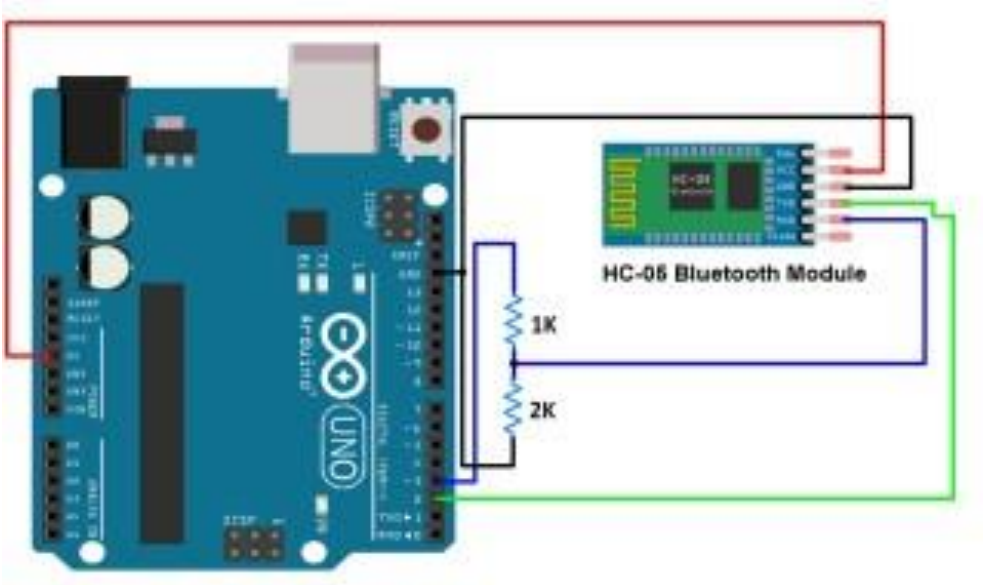
3) Select board type and port.

4) Write and upload the program in the Arduino-IDE.

5) Download and install a Bluetooth terminal application on phone and use it to connect to the HC-05Bluetooth module.

6) Data is sent from the Smartphone using the Bluetooth terminal application.

SCHEMATIC DIAGRAM:



PROGRAMME:

```
#include<softwareSerial.h>
/* Create object named bt of the class SoftwareSerial */
SoftwareSerial bt(2, 3);    /* (Rx,Tx) */
void setup() {
        bt.begin(9600);  /* Define baud rate for software serial communication */
        Serial.begin(9600);   /* Define baud rate for serial communication */
}
void loop() {
        if(bt.available()){      /* If data is available on serial port  */
                Serial.write(bt.read());  /* Print character received on to the serial monitor */
        }
}
```

OBSERVATION:

RESULT:

A message WELCOME is transmitted from Smartphone via Bluetooth to theArduino Uno and displayed it on Serial Monitor of PC.

EXEPERIMENT-12

Digital I/O Interface – PIR

AIM:

To Interface a PIR sensor with Arduino to detect motion and display "Stop" or "Go" depending on the output from the sensor.

APPPARATUS REQUIRED:

S.N o	Name of the equipment	Specifications	Quantity
1	**Arduino**	**UNO**	1
2	**RGB common cathode LED**		1
3	**Passive Infrared (PIR)Sensor**		1
4	**Jumper wires**		required
5	**Beardboard**		1
6	**Computer with Arduino IDE software**	IDE 1.8.14	1

PROCEDURE:

1) Build the circuit according to the schematic diagram

2) Use Arduino Uno on the Arduino Desktop IDE.

3) Select board type and port.

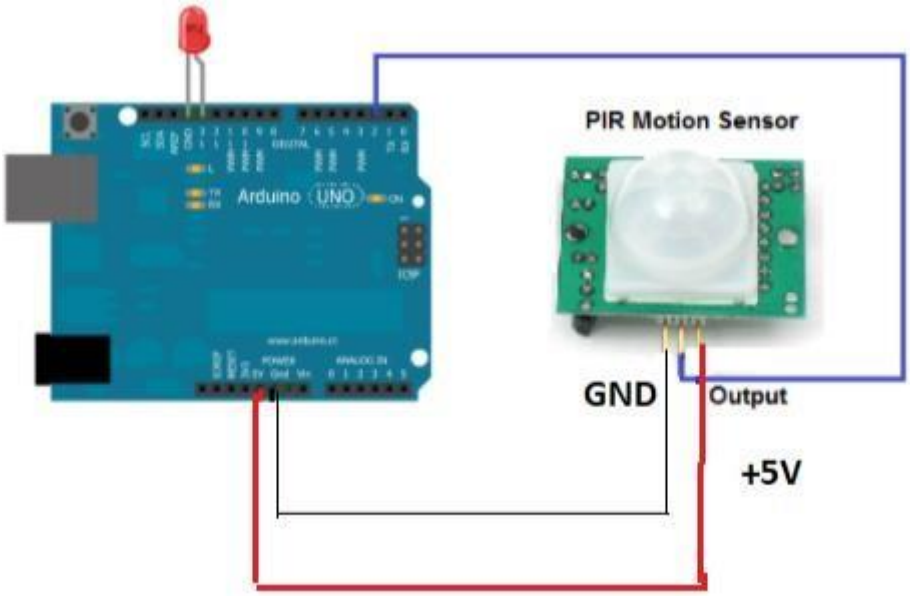
4) Write and upload the program in the Arduino-IDE

Passive Infrared (PIR) Sensor

THEORY:

The PIR sensor consists of 4 pins: GND, VCC, OUT, and EN. It can measure movement up to 30′. Within thedome-shaped lens (used to focus the incoming light onto the detectors), there are two slots that can detect different levels of infrared (IR) light or radiation. The sensor does not detect motion when the detectors sensethe same amount of IR. The motion is detected when the detectors measure two different levels of IR. The levels of IR can vary from object to object since each one emits different amounts of heat energy.

SCHEMATIC DIAGRAM:



PROGRAMME:

```
int LEDpin = 13;  //Connect RED LED at 13 pin
int obstaclePin = 2;
int hasObstacle = LOW;  // LOW MEANS NO OBSTACLE
void setup() {
        pinMode(LEDpin, OUTPUT);
        pinMode(obstaclePin, INPUT);
        Serial.begin(9600);
}
void loop() {
        hasObstacle = digitalRead(obstaclePin);
        if(hasObstacle == HIGH){
                Serial.println("Stop");
                digitalWrite(LEDpin,HIGH);
        }
        else{
                Serial.println("Go");
                digitalWrite(LEDpin, LOW);
        }
        delay(200);
```

}

OBSERVATIONS:

S.no	OBSTACLE	LED status
1	YES	ON
2	NO	OFF

RESULT:

Hence Digital I/O Interface like PIR is connected and controlled with Arduino board and correspondingoutputs are observed.

EXEPERIMENT-13

WIRELESS CONTROL OF WHEELED ROBOT USING WIFI.

AIM:

- To built a simple Robot (robotic car) that can be controlled over WiFi Network i.e. the user inputs for direction of the movement of the Robot are provided through the WiFi (with the help of asimple HTML Page).

APPPARATUS REQUIRED:

S.No	Name of the equipment	Specifications	Quantity
1	NodeMCU		1
2	LM298		1
3	Wheel Robo set		1
4	Jumper wires		required
5	Rechargeable Battery 9-12 v		1
6	Bread board		
7	Computer with Arduino IDE software	IDE 1.8.14	1

PROGRAMME:

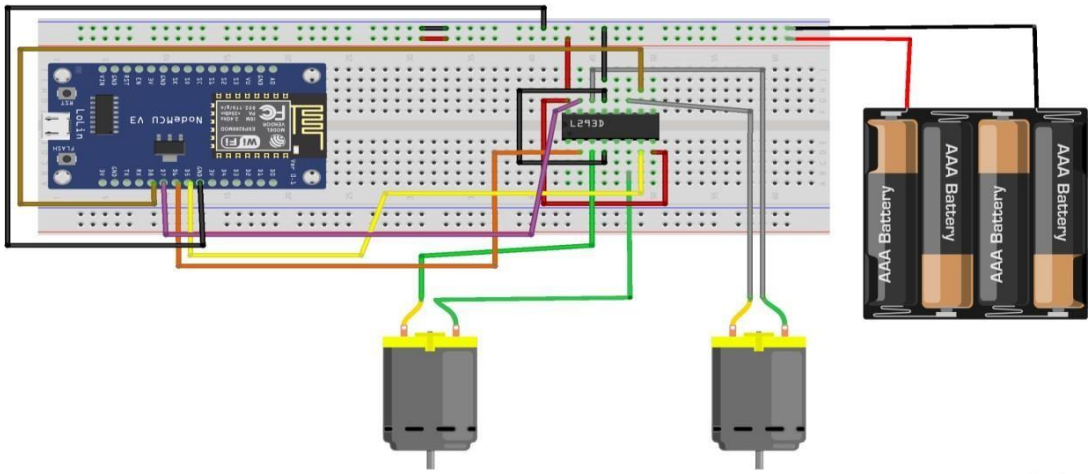
```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>
// Replace with your network credentials
const char* ssid = "TILAB";
const char* password = "Arduinonano";
ESP8266WebServer server(80); //instantiate server at port 80 (http port)
String page =" "; //For the Web Server
String page2=" "; //For updating Status of motor 1
String page3=" "; //For updating status of motor 2
void setup(void){
        //the HTML of the web page
        page = "<center><h1>Motor Control Web Server</h1><body><p><a
        href=\"Forward\"><button>Forward</button></a><p><a
        href=\"Backward\"><button>Backward</button></a></p><p><a href
  =\"Left\"><button>Left</button></a> <a href=\"Stop\"><button>Stop</button></a><a
```

```
href=\"Right\"><button>Right</button></a></p></body></center>";
pinMode(D5, OUTPUT); // inputs for motor 1
pinMode(D6,OUTPUT);
pinMode(D7,OUTPUT);        // inputs for motor 2
pinMode(D8,OUTPUT);
 pinMode(LED_BUILTIN,OUTPUT); // For status of WiFi connection
delay(1000);
 Serial.begin(115200);
WiFi.begin(ssid, password); //begin WiFi connection
Serial.println("");
// Wait for connection
 while(WiFi.status() != WL_CONNECTED) {
            delay(500);
           Serial.print(".");
 }
digitalWrite(LED_BUILTIN,HIGH);// when connected turns high
Serial.println("");
Serial.print("Connected to ");
Serial.println(ssid);
Serial.print("IP address: ");
Serial.println(WiFi.localIP()); //provides IP address
server.on("/", [](){server.send(200, "text/html", page+page2);});
server.on("/Forward",Forward);
server.on("/Backward",Backward);
server.on("/Left",Left);
server.on("/Right",Right);
server.on("/Stop",[](){            // turns all the motor input pins low
page2="<center><p> motor 1 Status : Off</p></center>";
 page3="<center><p> motor 2 Status : off</p></center>";
server.send(200,"text/html",page+page2+page3);
digitalWrite(D5,LOW);
digitalWrite(D6,LOW);
digitalWrite(D7,LOW);
digitalWrite(D8,LOW);
```

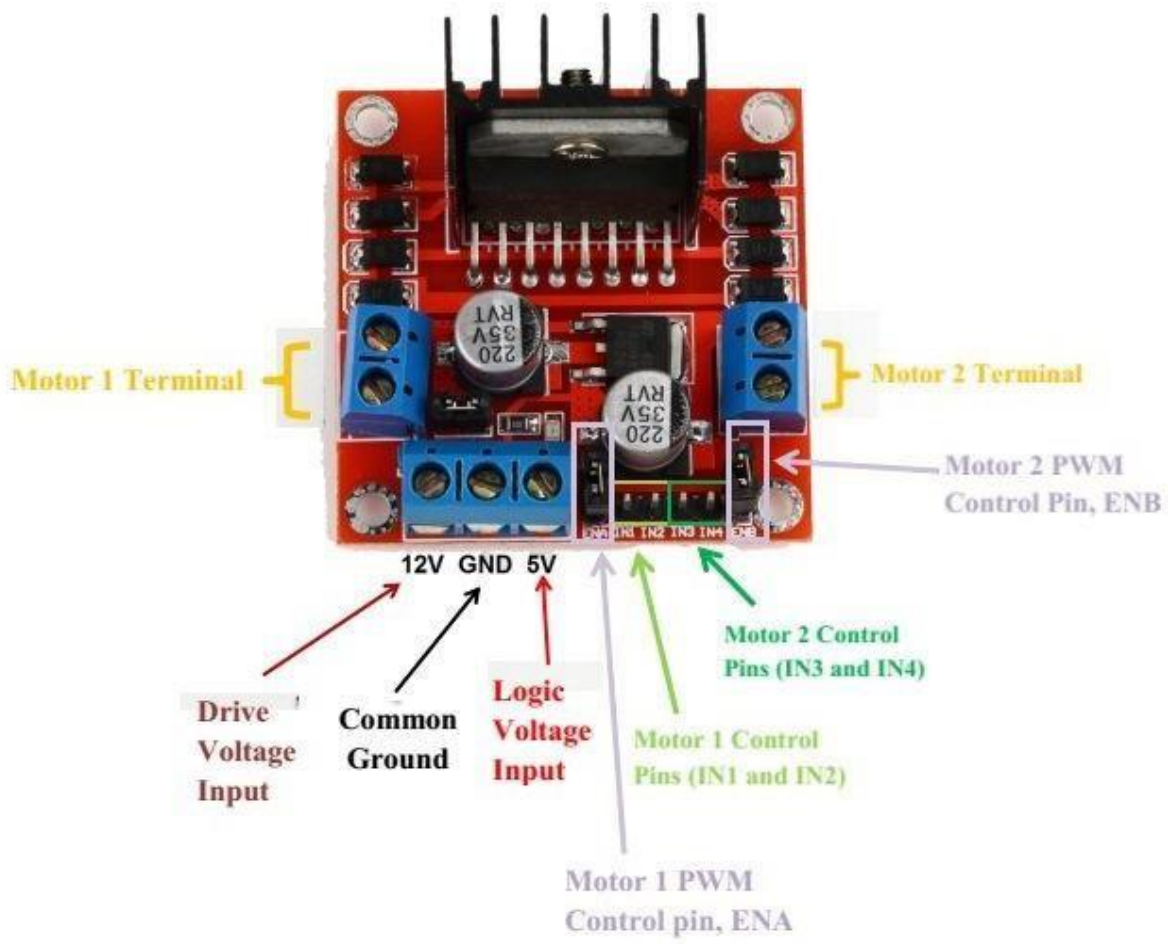
```
        delay(200);
        });
        server.begin();
        Serial.println("Web server started!");
}
void loop(void){
        server.handleClient();
}
void Forward(){
        digitalWrite(D5,HIGH);
        digitalWrite(D6,LOW);
        page2="<center><p> motor 1 Status : Forward </p></center>";
        server.send(200,"text/html", page+page2+page3);
        delay(200);
}
void Left(){
        page3="<center><p> motor 2 Status : Left</p></center>";
        server.send(200,"text/html",page+page2+page3);
        digitalWrite(D7,HIGH);
        digitalWrite(D8,LOW);
        delay(200);
}
void Right(){
        page3="<center><p> motor 2 Status : Right</p></center>";
        server.send(200,"text/html",page+page2+page3);
        digitalWrite(D8,HIGH);
        digitalWrite(D7,LOW);
        delay(200);
}
void Backward(){
        page2="<center><p> motor 1 Status : Backward</p></center>";
        server.send(200, "text/html", page+page2+page3);
        digitalWrite(D6, HIGH);
        digitalWrite(D5,LOW);
```

```
        delay(200);
    }
```

SCHEMATIC DIAGRAM:



LM298-Working



OBSERVATIONS:

RESULT:

Hence the WiFi Controlled Robot is controlled with the help of an HTML Web Page (whichcan be accessed using any web browser on a computer/Mobile phone that isconnected to the same WiFi Network).