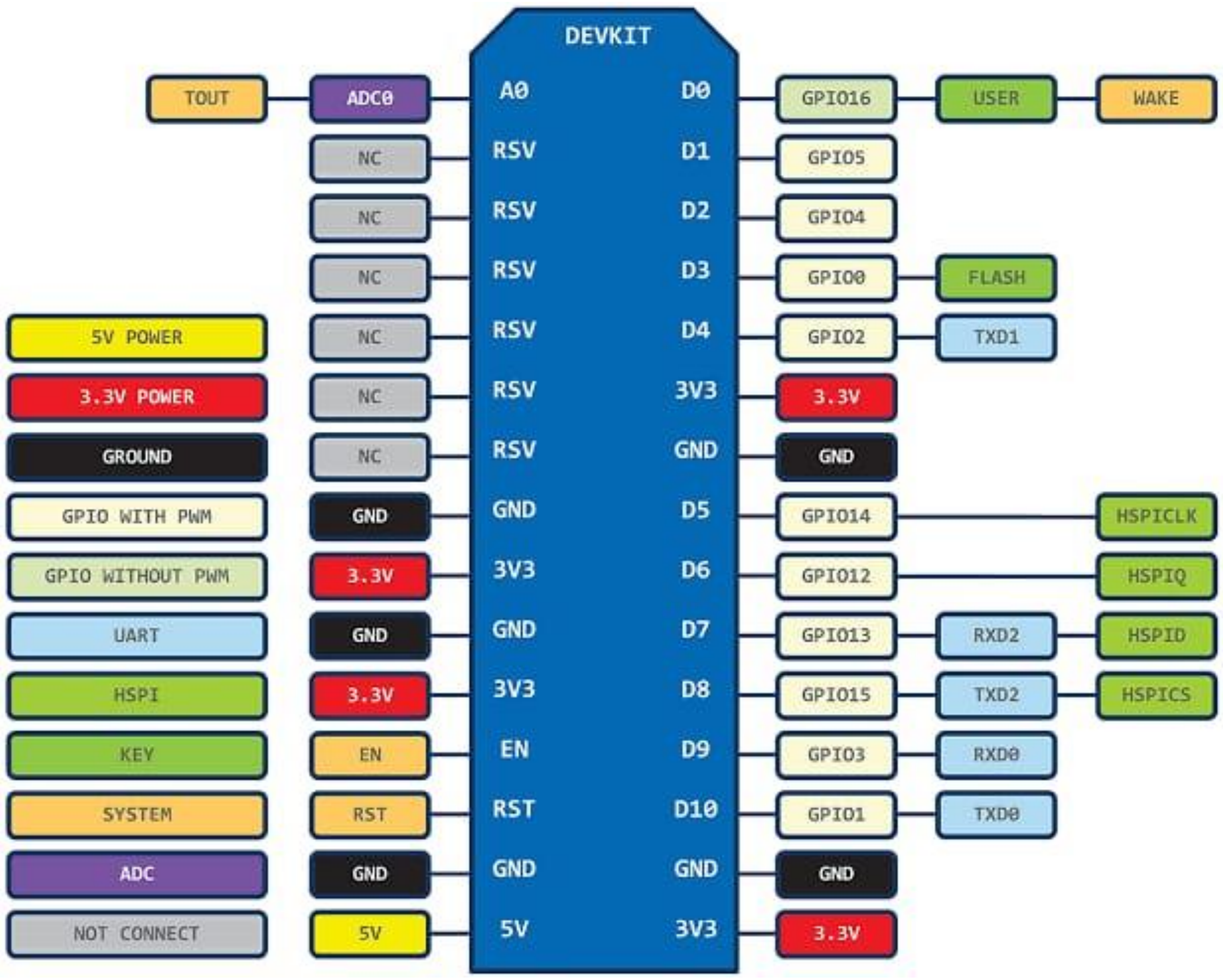
INDEX

EXPERIMENT – 1

AIM 1 : NODEMCU ESP8266 PINOUT Diagram



NodeMCU is an open source IoT platform.It includes firmware which runs on the ESP8266 Wi- FiSoC From Espressif Systems, and hardware which is based on the ESP- 12 module. The term "NodeMCU" by default refers to the firmware rather than the development kits.

NodeMCU is an open source development board and firmware based in the widely used ESP8266 - 12E WiFi modu le. It allows you to program the ESP8266 WiFi module with the simple and powerful LUA programming language or Arduino IDE.

With its USB-TTL , the nodeMCU Dev board supports directly flashing from USB port. It combines features of WIFI accesspoint and station + microcontroller. These features make the NodeMCU extremly powerful tool for Wifi networking. It can be used as access point and/or station, host a webserver or connect to internet to fetch or upload data.

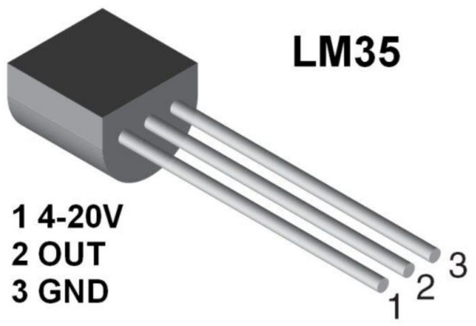
AIM 2 : Note down what each sensor does, its applications and the pins of each sensor.

<u>Sensor</u>

A device which gives an output by detecting the changes in quantities or events can be defined as a sensor. Generally, sensors produce an electrical signal or optical output signal corresponding to the changes in the inputs. There are different types of sensors, for example, consider a thermocouple which can be considered as temperature sensor that produces an output voltage based on the input temperature changes.

1) Temperature Sensor

- A device used to measure amount of heat energy that allows to detect a physical change in temperature from a particular source and converts the data for a device or user, is known as temperature sensor.
- Temperature sensor LM35 has 3 legs, the first leg is V_{cc}, which can be converted to 5V (ESP8266 board's output is 3.3V), middle leg is V_{out} (where the temperature is read from), right leg should be connected to ground.



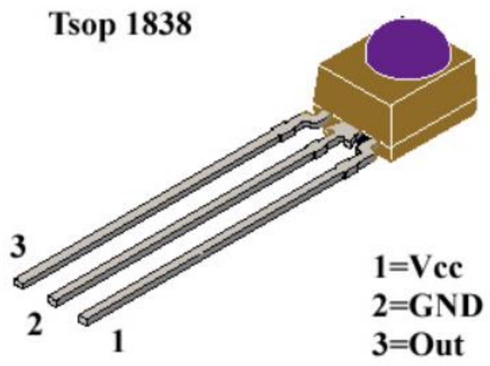
2) IR Sensor

IR sensor is a sensor which is used to sense certain characteristics of its surroundings by either emitting or detecting its radiation. It is also capable of measuring the heat being emitted by the objects and detecting motion.

Infrared sensors can be active or passive and they can be split into two main types:

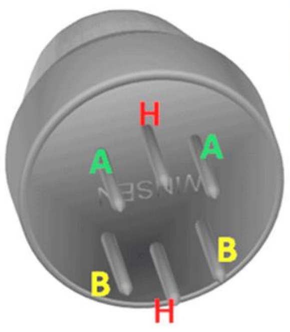
- **Thermal infrared sensors** – use infrared energy as heat. Their photo sensitivity is independent of the wavelength being detected. Thermal detectors do not require cooling but do have slow response times and low detection capabilities.

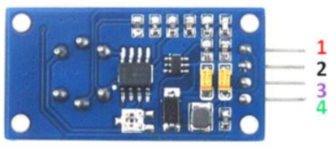
- **Quantum infrared sensors** – provide higher detection performance and faster response speed. Their photo sensitivity is dependent on wavelength. Quantum detectors have to be cooled in order to obtain accurate measurements.

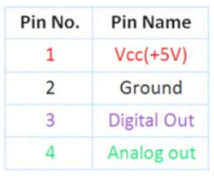


3) Gas Sensor

- A GAS sensor or a GAS Detector is a type of chemical sensor which detects/measures the concentration of gas in its vicinity. Gas sensor interacts with a gas to measure in concentration.
- They are used in various industries ranging from medicine to aerospace.
- Various technologies are used to measure Gas concentration such as semiconductors, oxidation, catalytic, infrared, etc.
- The most common types are as follows:
  - Metal Oxide Based GAS Sensor
  - Capacitance Based GAS Sensor
  - Acoustic Based GAS Sensor
  - Calorimetric GAS Sensor
  - Optical GAS Sensor
  - Electrochemical GAS Sensor



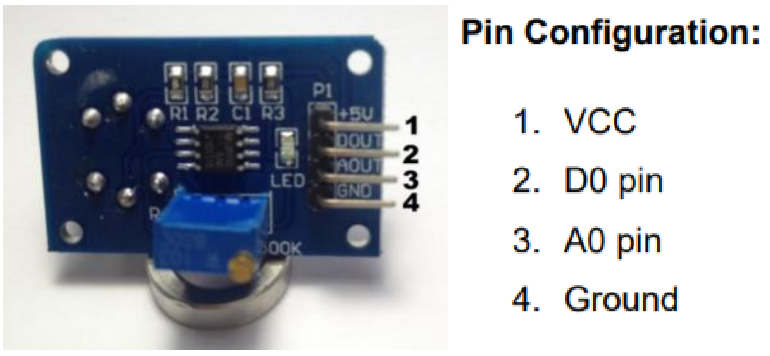




Pin No.	Pin Name
1	Vcc(+5V)
2	Ground
3	Digital Out
4	Analog out

4) Smoke Sensor

- A **smoke detector** is a device that senses smoke, typically as an indicator of fire. It provides a signal to a fire alarm system in a large building or produces an audible and visual signal locally in a room or a home.
- Smoke detectors are usually housed in a small, round shaped plastic case, and placed at the roof where there are risks of fire or fire hazards.
- There are two main types of smoke detectors: photoelectric and ionization.
- When smoke enters the detector chamber, a photoelectric type detects sudden scattering of light, whereas an ionization type detects the change of electrical current flow that triggers the signal - indicating the presence of smoke.
- Smoke detectors have an average life of about 10 years.



Pin Configuration:

1. VCC
2. D0 pin
3. A0 pin
4. Ground

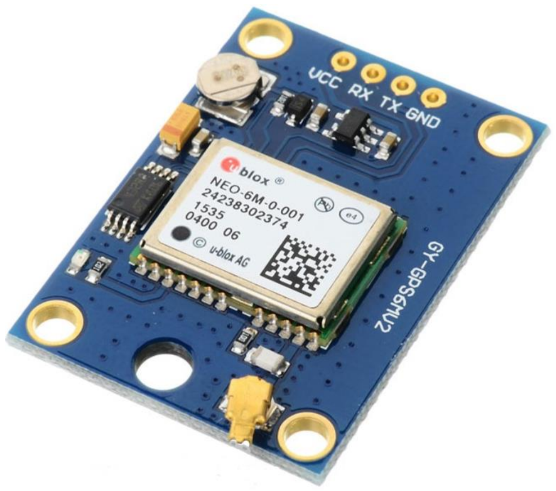
- The voltage that the sensor outputs changes according to smoke/gas.

Voltage ∞ concentration of gas/smoke

5) GPS Module

- GPS receiver module gives output in standard (National Marine Electronics Association) NMEA string format. It provides output serially on Tx pin with default 9600 Baud rate.
- This NMEA string output from GPS receiver contains different parameters separated by commas like longitude, latitude, altitude, time etc. Each string starts with '$' and ends with carriage return/line feed sequence.

- GPS receiver receives information signals from GPS satellites and calculates its distance from satellites. This is done by measuring the time required for the signal to travel from satellite to the receiver.



VCC: Power Supply 3.3 – 6 V

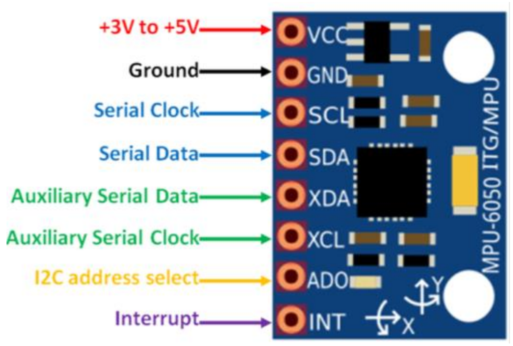
GND: Ground

TX: Transmit data serially which gives information about location, time etc.

RX: Receive Data serially. It is required when we want to configure GPS module.

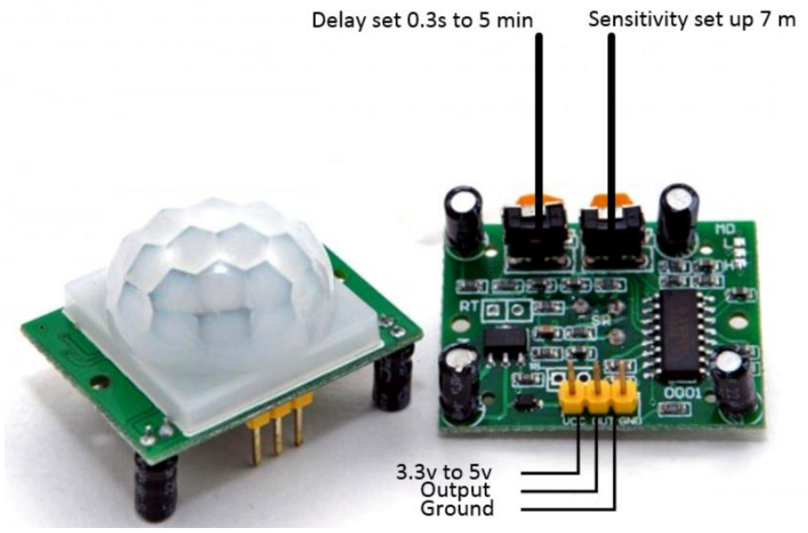
6) Gyroscope sensors

- A sensor or device which is used to measure the angular rate / angular velocity is known as gyro sensors. It is primarily used navigation and measurement of angular and rotational velocity in 3-axis directions. It uses Earth’s gravity to help determine orientation.
- There are three basic types of gyroscope Rotary (classical) gyroscopes, Vibrating Structure Gyroscope and Optical Gyroscopes.
- Main Applications:
  - Car navigation systems
  - Game controllers
  - Camera devices
  - Drone control, etc



7) PIR Sensor

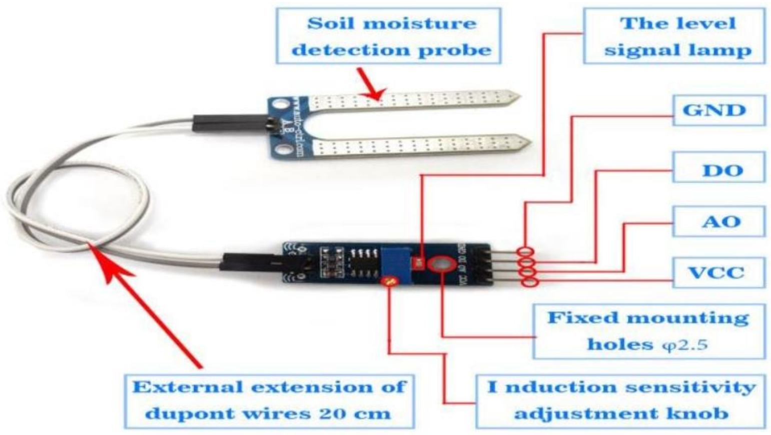
- PIR sensors are used to sense motion, almost always whether a human has moved in or out of the sensors range. They are small, inexpensive, low-power, easy to use and don't wear out.
- They are commonly found in appliances and gadgets used in homes or businesses. They are often referred to as PIR, "Passive Infrared", "Pyroelectric", or "IR motion" sensors.
- PIRs are basically made of a pyroelectric sensor, which can detect levels of infrared radiation. Everything emits some low-level radiation, and the hotter something is, the more radiation is emitted.
- The sensor in a motion detector is actually split in two halves. The reason for that is that we are looking to detect motion (change) not average IR levels. The two halves are wired up so that they cancel each other out. If one half sees more or less IR radiation than the other, the output will swing high or low.



8) Soil Moisture Sensor

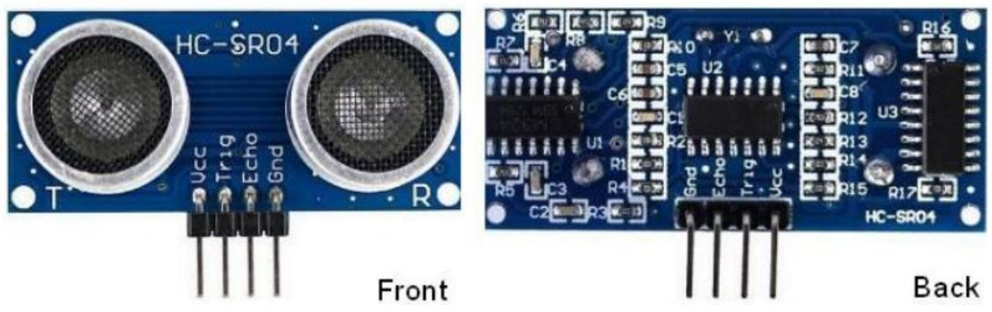
- Soil moisture sensor consist of two conducting plates which function as a probe and acting as a variable resistor together.
- When the sensor is inserted into the water, the resistance will decrease and get better conductivity between plates.
- First plate is connected to the +5Volt supply through series resistance of 10K ohm and second plate is connected directly to the ground.
- It simply acts as a voltage divider bias network, and output is taken directly from the first terminal of the sensor pin, which is shown in figure above.

- The output will change in the range of 0 – 5 Volt, in proportion with change in content of water in the soil.
- Ideally, when there is zero moisture in soil, the sensor acts as open circuit i.e. infinite resistance. For this condition, we get 5V at the output.


Soil moisture detection probe
The level signal lamp
GND
DO
AO
VCC
Fixed mounting holes φ2.5
External extension of dupont wires 20 cm
I nduction sensitivity adjustment knob

9) Ultrasonic Module

- The ultrasonic sensor works on the principle of SONAR and RADAR system which is used to determine the distance to an object.
- An ultrasonic sensor generates the high-frequency sound (ultrasound) waves. When this ultrasound hits the object, it reflects as echo which is sensed by the receiver.
- By measuring the time required for the echo to reach to the receiver, we can calculate the distance. This is the basic working principle of Ultrasonic module to measure distance.



VCC - +5 V supply

TRIG – Trigger input of sensor. Microcontroller applies 10 us trigger pulse to the HC-SR04 ultrasonic module.

ECHO–Echo output of sensor. Microcontroller reads/monitors this pin to detect the obstacle or to find the distance.

GND – Ground

10) GSM Module

- A GSM module or a GPRS module is a chip or circuit that will be used to establish communication between a mobile device or a computing machine and a GSM or GPRS system.
- GSM stands for Global System for Mobile communication.
- It is used to send, receive SMS/MMS or make a GPRS data connection.
- GSM systems have following advantages over basic land line telephony systems:
  1. Mobility
  2. Easy availability
  3. High uptime

11) Pressure Sensor

A pressure sensor is a device which senses pressure and converts it into an analog electric signal whose magnitude depends upon the pressure applied. Since they convert pressure into an electrical signal, they are also termed as pressure transducers.

Types of pressure sensors:

- Absolute pressure sensor

This sensor measures the pressure relative to perfect vacuum.

- Gauge pressure sensor

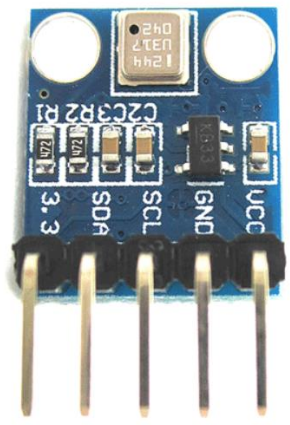
This sensor measures the pressure relative to atmospheric pressure.

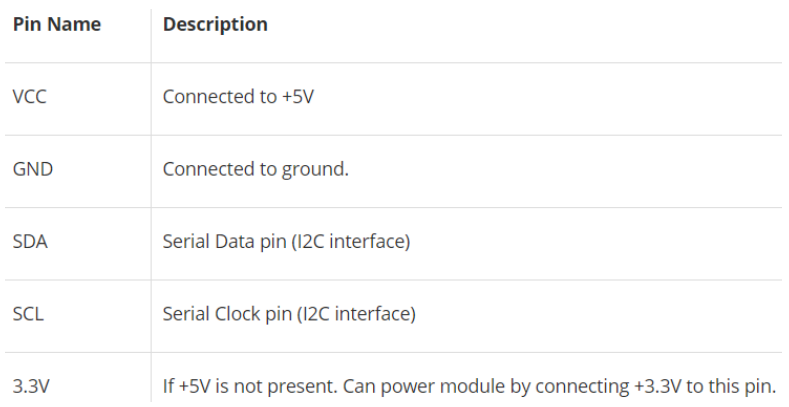
- Differential pressure sensor

This sensor measures the difference between two or more pressures introduced as input for the sensing unit.

- Sealed pressure sensor

This sensor is similar to a gauge pressure sensor except that it measures pressure relative to some fixed pressure rather than the ambient atmospheric pressure (which varies according to the location and the weather).





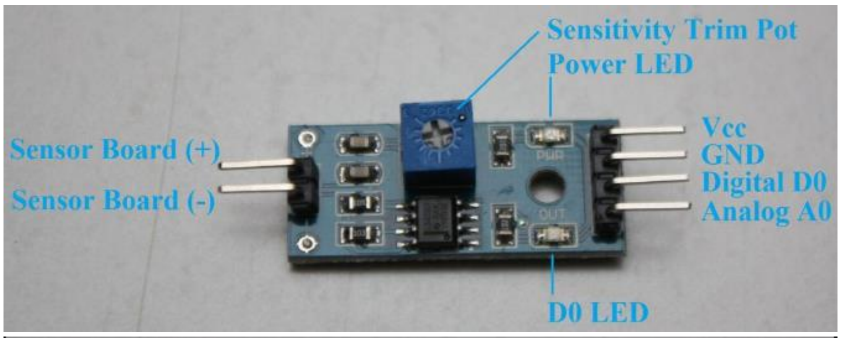
Pin Name	Description
VCC	Connected to +5V
GND	Connected to ground.
SDA	Serial Data pin (I2C interface)
SCL	Serial Clock pin (I2C interface)
3.3V	If +5V is not present. Can power module by connecting +3.3V to this pin.

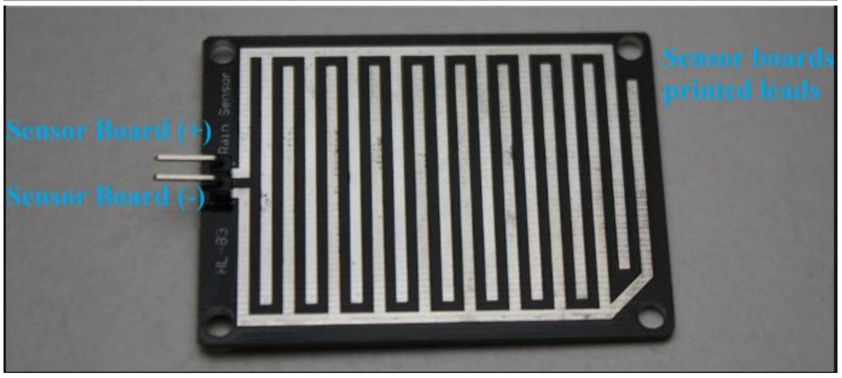
(BMP180 Sensor)

12) Rain Sensor

A rain sensor or rain switch is a switching device activated by rainfall. There are two main applications for rain sensors:

- The first is a water conservation device connected to an automatic irrigation system that causes the system to shut down in the event of rainfall.
- The second is a device used to protect the interior of an automobile from rain and to support the automatic mode of windscreen wipers.



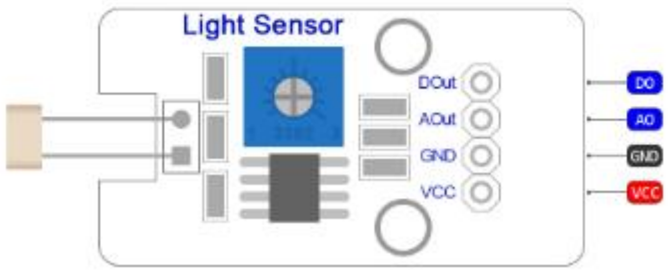


13) **Water quality sensor**

Water quality sensor is used to monitor quality of water with the help of information sensed by its sensors immersed in water, here pH sensors and turbidity sensors are used to measure quality of water.

14) **Light Sensor**

- Analog signals that are used for detecting the amount of light striking the sensor are called as light sensors.
- LDR (Light Dependant Resistors) are used as analog light sensor which can be used to switch on and off automatically based on the day light incident on LDR.
- The resistance of LDR increases with decrease in light and decreases with increase in light.

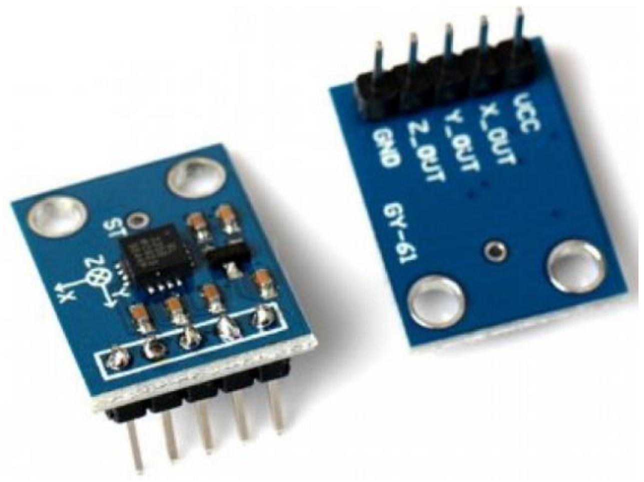


15) **Proximity Sensor**

- A proximity sensor is a sensor able to detect the presence of nearby objects without any physical contact.
- A proximity sensor often emits an electromagnetic field or a beam of electromagnetic radiation (infrared, for instance), and looks for changes in the field or return signal. The object being sensed is often referred to as the proximity sensor's target.
- Different proximity sensor targets demand different sensors. For example, a capacitive proximity sensor or photoelectric sensor might be suitable for a plastic target; an inductive proximity sensor always requires a metal target.
- Proximity sensors can have a high reliability and long functional life because of the absence of mechanical parts and lack of physical contact between the sensor and the sensed object.
- Proximity sensors are also used in machine vibration monitoring to measure the variation in distance between a shaft and its support bearing. This is common in large steam turbines, compressors, and motors that use sleeve-type bearings.

16) Accelerometer

- Accelerometers are integrated circuits or modules used to measure the acceleration of an object to which they are attached.
- They are used in applications including: vehicle dynamics, mobile phone orientation detection, image stability, tilt, tap detection and anti-theft devices. Accelerometers are available in a number of technologies.
- Accelerometers are commonly made of either piezoelectric, piezoresistive or capacitive elements, which are used to convert the mechanical motion into an electrical signal.
- The piezoelectric is the most common form of accelerometer that uses microscopic crystal structures. When the static crystal structure is deformed due to physical force or bending, it creates a voltage from the stress and the accelerometer interprets the voltage to determine velocity and orientation.



EXPERIMENT - 2

AIM 1: Configuring the ARDUINO IDE (Version 1.8.8) To NODEMCU ESP8266 In Windows

DESCRIPTION: Since the Arduino IDE's default board is set to Arduino board, we need to change a few settings/ configurations in the IDE to map it to NODEMCU ESP8266 (the board we are using)

STEP 1: Type the below link the address bar. Click on the Just Download button.

https://www.arduino.cc/en/Main/Donate

STEP 2: Once the Arduino IDE exe file is downloaded, run it and install the software to your

computer.

STEP 3: Open the IDE, Go to File Preferences. In the Additional Boards Manager URL text box

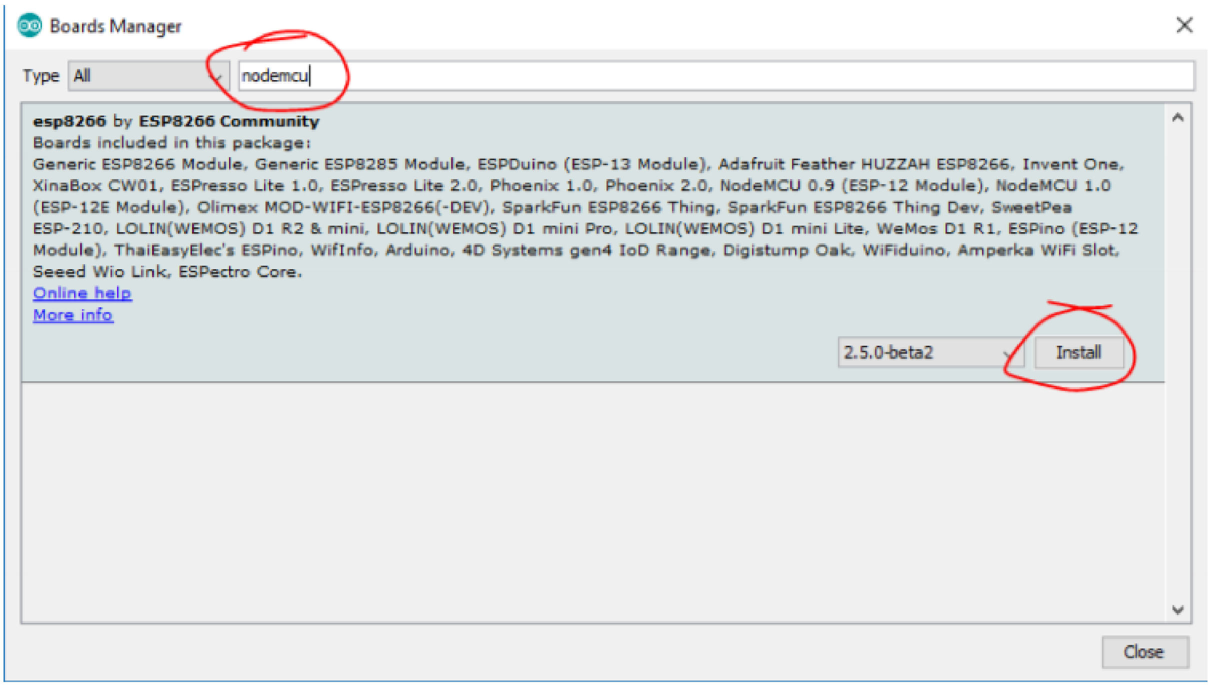
paste the below link and Click on OK:

http://arduino.esp8266.com/stable/package_esp8266com_index.json

STEP 4: From the Menu bar go to Tools Board : "Arduino/ Genuine Uno" Boards Manager

STEP 5: In the search bar type nodemcu. The manager lists the ESP8266 board. Click the install

button. After installation close the window.



esp8266 by **ESP8266 Community**
Boards included in this package:
Generic ESP8266 Module, Generic ESP8285 Module, ESPDuino (ESP-13 Module), Adafruit Feather HUZZAH ESP8266, Invent One, XinaBox CW01, ESPresso Lite 1.0, ESPresso Lite 2.0, Phoenix 1.0, Phoenix 2.0, NodeMCU 0.9 (ESP-12 Module), NodeMCU 1.0 (ESP-12E Module), Olimex MOD-WIFI-ESP8266(-DEV), SparkFun ESP8266 Thing, SparkFun ESP8266 Thing Dev, SweetPea ESP-210, LOLIN(WEMOS) D1 R2 & mini, LOLIN(WEMOS) D1 mini Pro, LOLIN(WEMOS) D1 mini Lite, WeMos D1 R1, ESPino (ESP-12 Module), ThaiEasyElec's ESPino, WifInfo, Arduino, 4D Systems gen4 IoD Range, Digistump Oak, WiFiduino, Amperka WiFi Slot, Seeed Wio Link, ESPectro Core.
Online help
More info

NOTE: Steps 3,4 and 5 are done to install the nodemcu esp8266 board configurations and interfaces

to our Arduino IDE because the default board will the Arduino board.

STEP 6: Go to Tools Board, scroll down ,find NODEMCU 1.0 (ESP-12E module) and select it.

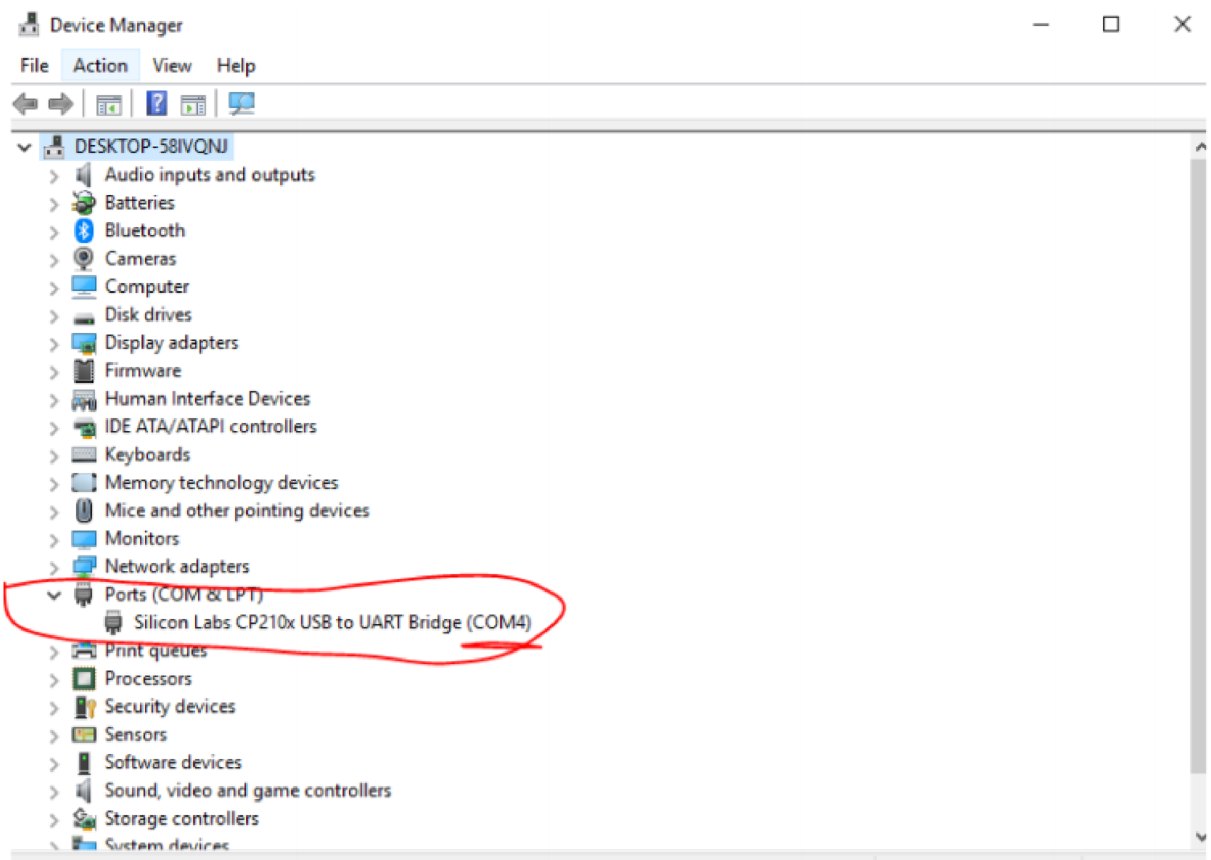
STEP 7: Go to Tools PORT and select the port related to NODEMCU.

HOW TO FIND THE PORT RELATED TO NODEMCU?

In Windows Search, search for Device Manager. In the Device Manager window, from the list of

devices displayed, select Ports (COM & LPT). Under it NODEMCU's PORT Number will be displayed.

Select the same in your Arduino IDE. Refer to below screenshot (varies from system to system, generally COM3/COM4)



This is done only once. The next time you use the system just check if the Board is set to NODEMCU 1.0 (ESP-12E module) and port is set to the respective port in the Tools Menu of your arduino IDE.

AIM 2: Blinking the On-Board LED Light

DESCRIPTION: To ON the blue LED present on the NODEMCU board for one second and after one second delay OFF the LED. This is repeated continuously.

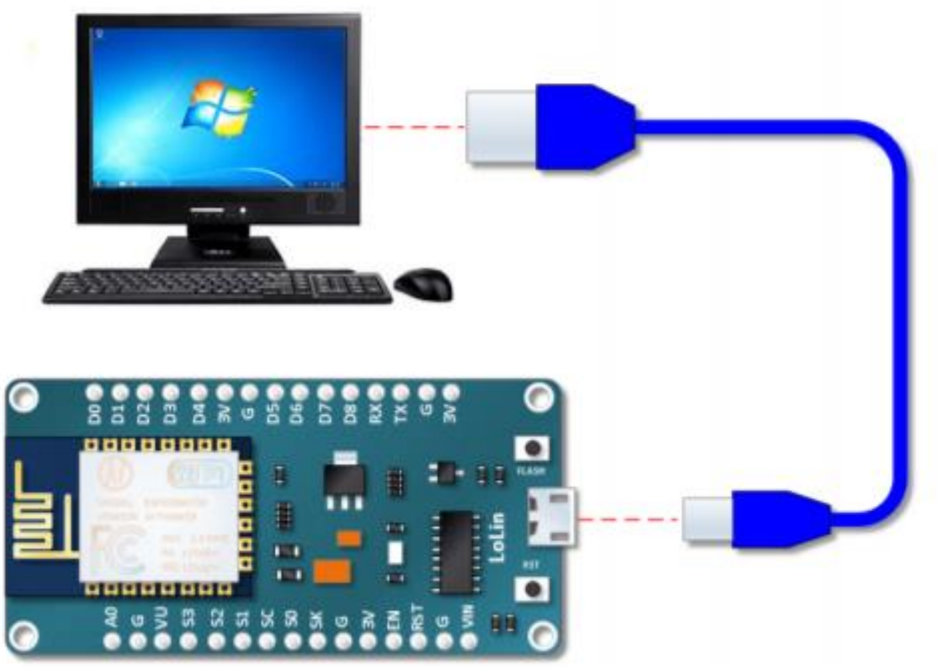
HARDWARE REQUIREMENTS:

- NODEMCU ESP8266 Board
- USB Cable to connect NODEMCU Board to CPU

SOFTWARE REQUIREMENTS:

- Arduino IDE v1.8.8 Windows.

BOARD/ CONNECTION DIAGRAM:



CODE:

```
void setup()
{
        pinMode(LED, OUTPUT);
}

void loop()
{
        digitalWrite(LED, HIGH);
        delay(1000);
        digitalWrite(LED, LOW);
        delay(1000);
}
```

RUNNING THE ARDUINO CODE/SKETCH:

- Save your code in This PC -> Documents -> Arduino. A folder with the same name as your file name will be created in the Arduino folder and the code file is stored inside it.
- Compile the Sketch (Code file is called sketch)
- Upload the Sketch (Uploads the code to NODEMCU).
- Everytime the code is modified, it has to be compiled and uploaded.

RESULTS/OBSERVATIONS/ANALYSIS :
The blue LED on the NodeMCU board blinks for 1sec after every 1sec time interval.

AIM 3: BLINKING AN EXTERNAL LED LIGHT

DESCRIPTION: To ON an external RED colour LED present on the breadboard for one second and

after one second delay OFF the LED. This is repeated continuously.

a) WITHOUT RESISTOR

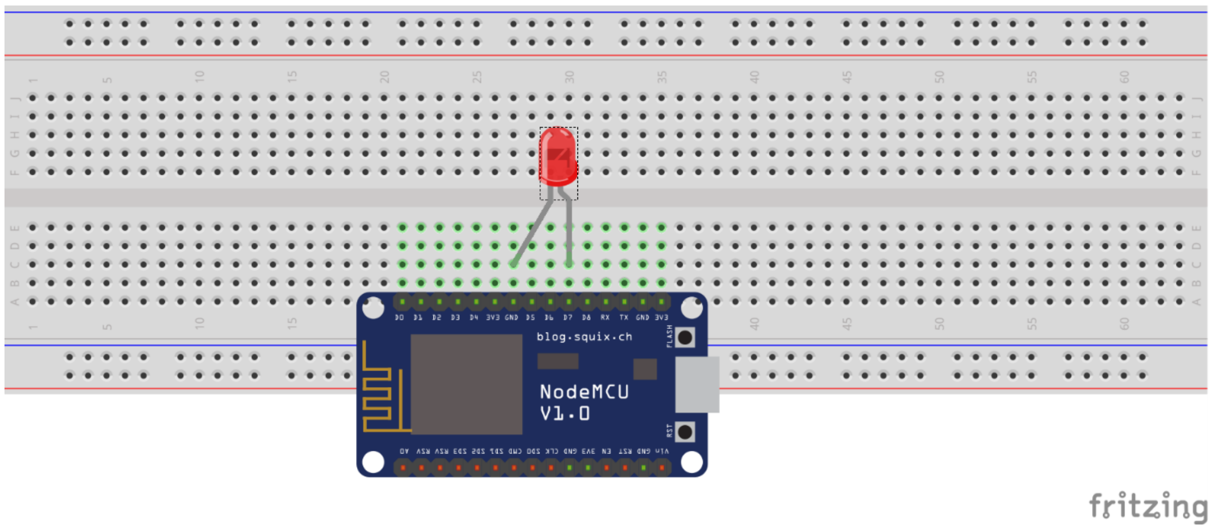
HARDWARE REQUIREMENTS:

- NODEMCU ESP8266 Board
- USB Cable to connect NODEMCU Board to CPU
- Bread Board
- LED
- Connecting wires

SOFTWARE REQUIREMENTS:

- Arduino IDE v1.8.8 Windows.

BOARD/ CONNECTION DIAGRAM:



CODE:

```
#define LED D7 (or 13)
void setup() {
        pinMode(LED, OUTPUT);
}
void loop() {
        digitalWrite(LED, HIGH);
        delay(1000);
        digitalWrite(LED, LOW);
        delay(2000);
}
```

RESULTS AND ANALYSIS:
The red LED blinks

b) WITH RESISTOR

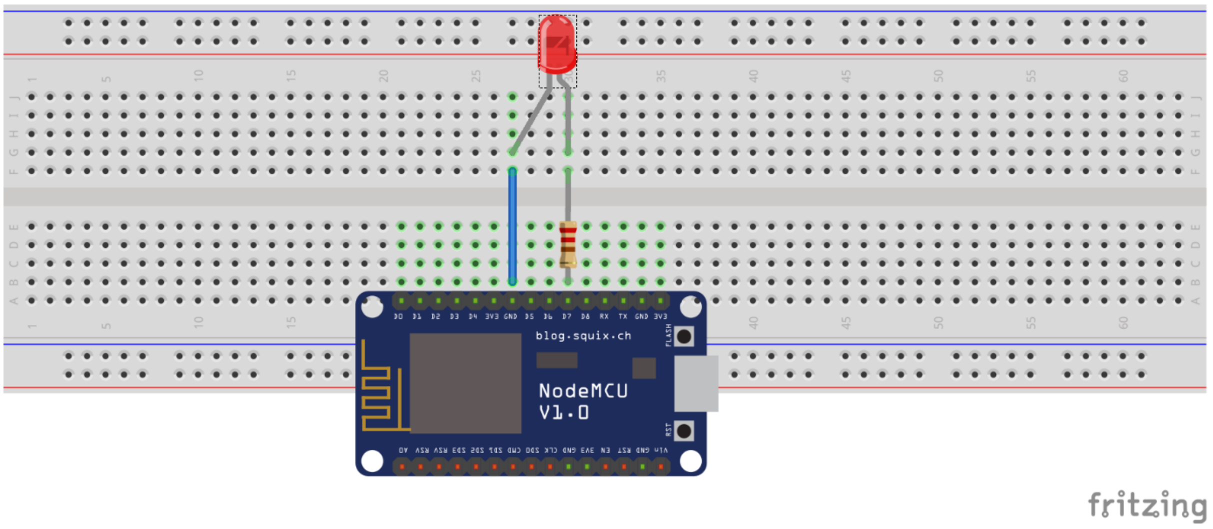
HARDWARE REQUIREMENTS:

- NODEMCU ESP8266 Board
- USB Cable to connect NODEMCU Board to CPU
- Bread Board
- LED
- Connecting wires
- resistor

SOFTWARE REQUIREMENTS:

- Arduino IDE v1.8.8 Windows.

BOARD/ CONNECTION DIAGRAM:



CODE:

```
#define LED D7 (or 13)
void setup() {
        pinMode(LED, OUTPUT);
}
void loop() {
        digitalWrite(LED, HIGH);
        delay(1000);
        digitalWrite(LED, LOW);
        delay(2000);
}
```

RESULTS AND ANALYSIS: The red LED blinks

AIM 4: CONTROLLING AN LED LIGHT USING WIFI AND BLYNK APP

DESCRIPTION: Since ESP8266 is a WIFI module, we can use it to operate the output devices (such as LED, motors etc.) wirelessly. To ON an external RED color LED present on the breadboard for one second and after one second delay OFF the LED. This is repeated continuously.

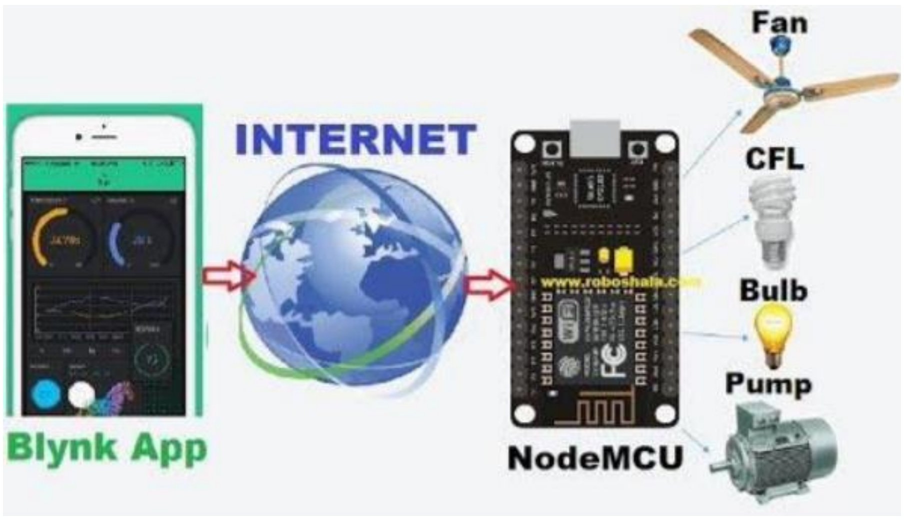
HARDWARE REQUIREMENTS:

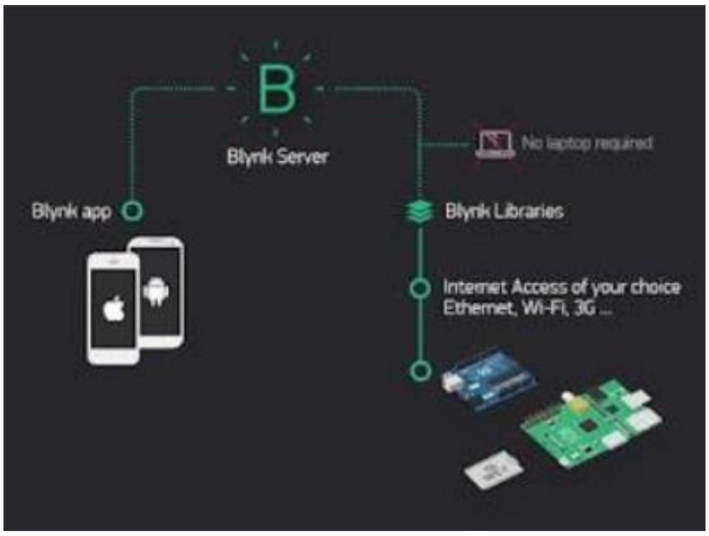
- NODEMCU ESP8266 Board
- USB Cable to connect NODEMCU Board to CPU
- Breadboard
- LED
- RESISTOR 330 OHM

SOFTWARE REQUIREMENTS:

- Arduino IDE v1.8.8 Windows.
- Blynk App

INSTALLING BLYNK APP IN MOBILE PHONE AND BLYNK LIBRARIES TO ARDUINO IDE





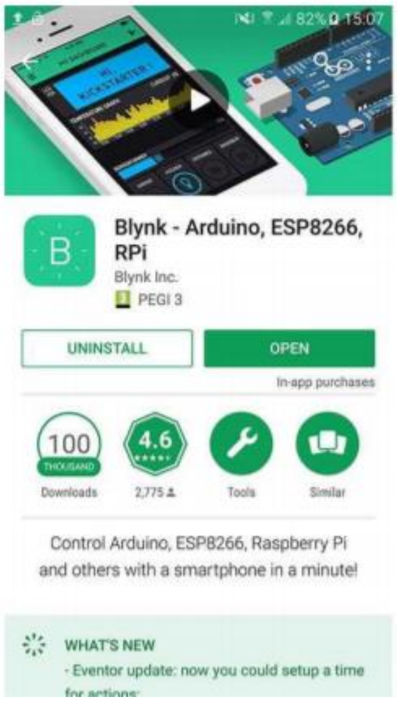
Step 1: Install Blynk app in your phone.

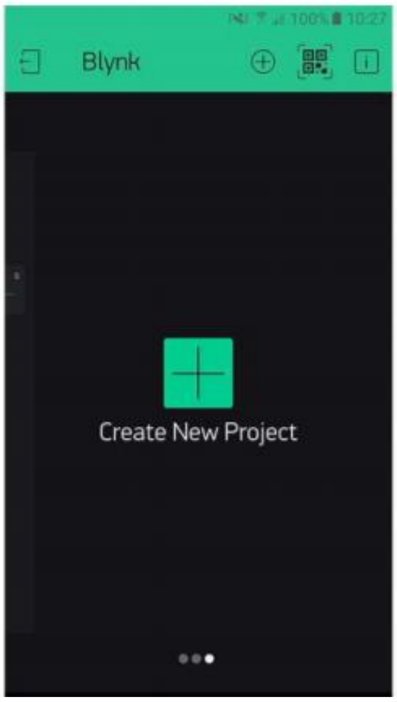
Step 2: Register with your email account.

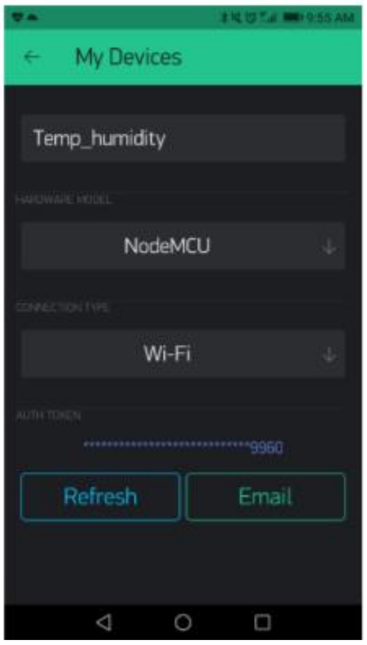
Step 3: Create New Project and give a Project name to it.

Step 4: Set Choose device to NodeMCU , set Connection Type to WI-FI and click the Create button.

Step 5: An Authentication Token is generated by the app and sent to the registered email account. (This authentication token is unique to every Blynk Project that you create in your phone. It helps link the phone project to the Code written in your Arduino IDE).


Blynk - Arduino, ESP8266, RPi
Blynk Inc.
PEGI 3
UNINSTALL
OPEN
In-app purchases
100 THOUSAND Downloads
4.6 2,775
Tools
Similar
Control Arduino, ESP8266, Raspberry Pi and others with a smartphone in a minute!
WHAT'S NEW
- Eventor update: now you could setup a time for actions





Temp_humidity

HARDWARE MODEL
NodeMCU

CONNECTION TYPE
Wi-Fi

AUTH TOKEN
****************************9960

Step 6: In your project home screen slide left. You will get a widget box.

Step 7: Select Button. A Button is created on the project home screen.

Step 8. Clicking on the Button will take you to Button Settings.

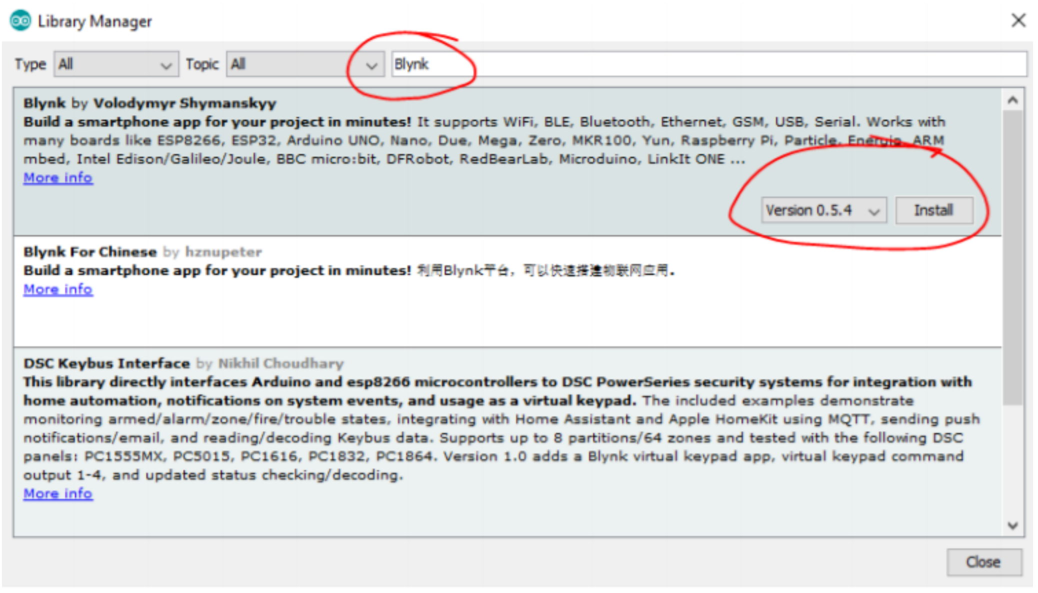
Step 9: For the Output select Digital from the left side menu and D7 from the right-side menu (D7 because we are connecting the Anode of the LED to D7. If the pin is changed on the board, it should be changed here as well).

Step 10: For the Mode, select Switch Mode. (Also, try to put it in Push Mode and see how it works).

Step 11: Now we will add the Blynk libraries to our Arduino IDE. In the Arduino IDE software, go to

Sketch -> Include Libraries -> Manage Libraries.

Step 12: Search for Blynk in the library manager and click on install.



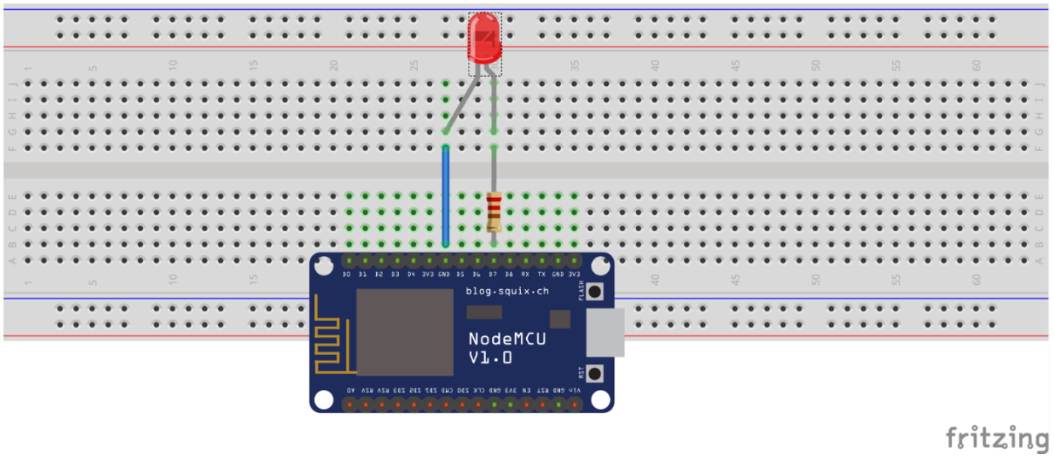
Blynk

Blynk by **Volodymyr Shymanskyy**
Build a smartphone app for your project in minutes! It supports WiFi, BLE, Bluetooth, Ethernet, GSM, USB, Serial. Works with many boards like ESP8266, ESP32, Arduino UNO, Nano, Due, Mega, Zero, MKR100, Yun, Raspberry Pi, Particle, Energia, ARM mbed, Intel Edison/Galileo/Joule, BBC micro:bit, DFRobot, RedBearLab, Microduino, LinkIt ONE ...
More info

Blynk For Chinese by hznupeter
Build a smartphone app for your project in minutes! 利用Blynk平台，可以快速搭建物联网应用。
More info

DSC Keybus Interface by Nikhil Choudhary
This library directly interfaces Arduino and esp8266 microcontrollers to DSC PowerSeries security systems for integration with home automation, notifications on system events, and usage as a virtual keypad. The included examples demonstrate monitoring armed/alarm/zone/fire/trouble states, integrating with Home Assistant and Apple HomeKit using MQTT, sending push notifications/email, and reading/decoding Keybus data. Supports up to 8 partitions/64 zones and tested with the following DSC panels: PC1555MX, PC5015, PC1616, PC1832, PC1864. Version 1.0 adds a Blynk virtual keypad app, virtual keypad command output 1-4, and updated status checking/decoding.
More info

BOARD/ CONNECTION DIAGRAM:



CODE:

```
#define BLYNK_PRINT Serial
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
char auth[] = "4e732d4bb5ea4bac8016f1955a5f82c6";
char ssid[] = "network name";
char pass[] = "network password";
void setup()
{
        Serial.begin(9600);
        Blynk.begin(auth, ssid, pass);
        // You can also specify server:
        //Blynk.begin(auth, ssid, pass, "blynk-cloud.com", 8442);
        //Blynk.begin(auth, ssid, pass, IPAddress(192,168,1,100), 8442);
}
```

```
void loop()
{
	Blynk.run();
}
```

Once the code is uploaded successfully to your NODEMCU module, go to your BLYNK phone app project that you created and click the PLAY BUTTON on the top right corner. This will run your project and connect to the NODEMCU.

There should not be any errors shown on your phone in red colour (Generally shown on the top right corner of phone app when phone is not yet connected to NODEMCU).

If no such errors are displayed, use the button in your App to control the LED else keep refreshing your Phone App till it shows a message that you are online.

RESULTS/OBSERVATIONS/ANALYSIS:

The red LED bulb glows when it is switched on from the Blynk Application.

AIM 5: Changing the brightness of external LED.

DESCRIPTION: To increase and decrease the brightness of the external red LED.

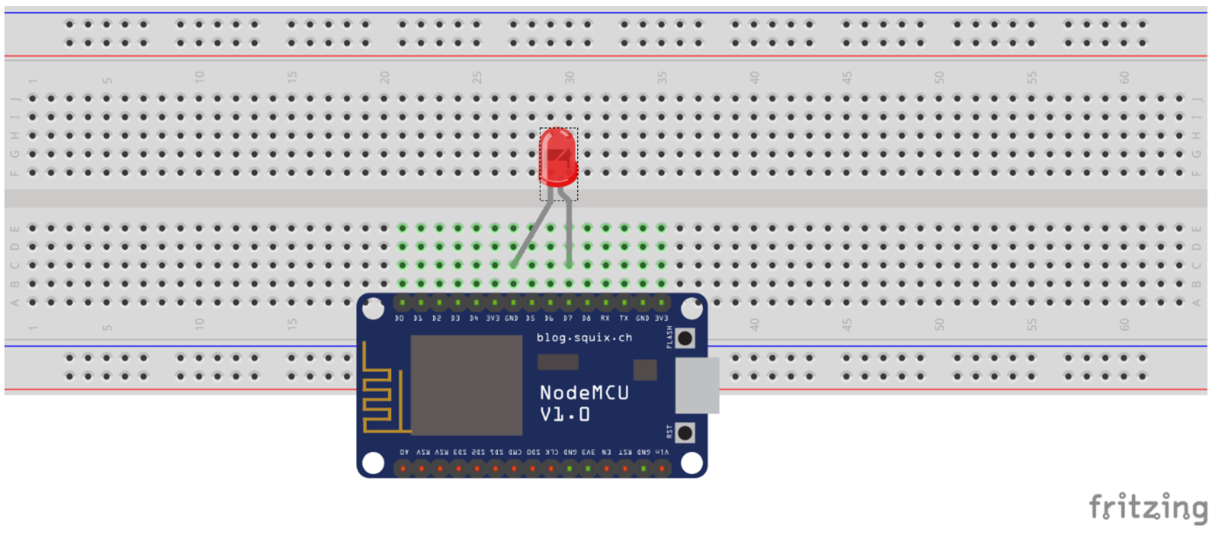
HARDWARE REQUIREMENTS:

- NODEMCU ESP8266 Board
- USB Cable to connect NODEMCU Board to CPU
- Breadboard
- LED

SOFTWARE REQUIREMENTS:

- Arduino IDE v1.8.8 Windows

BOARD/ CONNECTION DIAGRAM:



CODE:

```
#define LED D7
void setup() {
        pinMode(LED, OUTPUT);}
void loop() {
        for(int i=0; i<200; i++)
        {
                analogWrite(LED,i);
                delay(10);
        }
        for(int i=200; i>0; i--)
        {
                analogWrite(LED,i);
                delay(10);
        }
}
```

RESULTS AND ANALYSIS : The brightness of the LED increases and decreases with a delay 10 milliseconds.

Experiment – 3

AIM 1: Connect the rain sensor to NODEMCU and output on the serial monitor: 1 – when there is rain ; 0- when there is no rain

DESCRIPTION: To display 1 or 0 on the serial monitor, when there is rain or no rain respectively. Rain sensor which is connected to rain shield is used to detect rain. When rain shield is exposed to rain, rain sensor gives positive output.

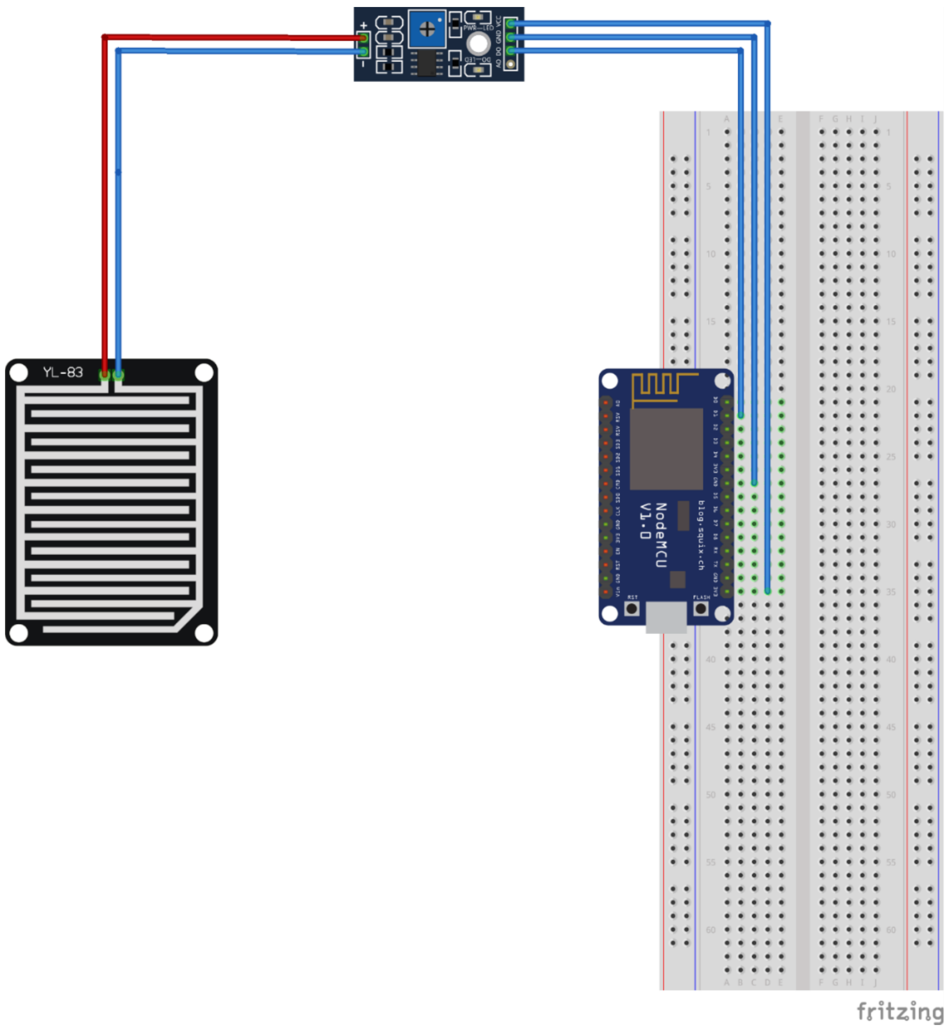
HARDWARE REQUIREMENTS:

- NODEMCU ESP8266 Board
- USB cable to connect NODEMCU board to CPU
- Connecting cables
- Rain shield
- Rain sensor

SOFTWARE REQUIREMENTS:

- Arduino IDE v1.8.8 Windows

BOARD DIAGRAM / CONNECTION DIAGRAM:



CODE:

```
#define pin D1
void setup()
{
	pinMode(pin, INPUT);
	Serial.begin(9600);
}
int v;
void loop()
{
	V=digitalRead(pin);
	Serial.println(v);
}
```

RESULTS AND ANALYSIS:

The serial monitor displays:
0 – initially when there is no rain
1- when there is rain (water on rain shield)

AIM 2: Connect an LED to the above circuit and let the LED glow whenever there is rain.

DESCRIPTION: To ON the external RED colour LED present on the breadboard when there is rain.

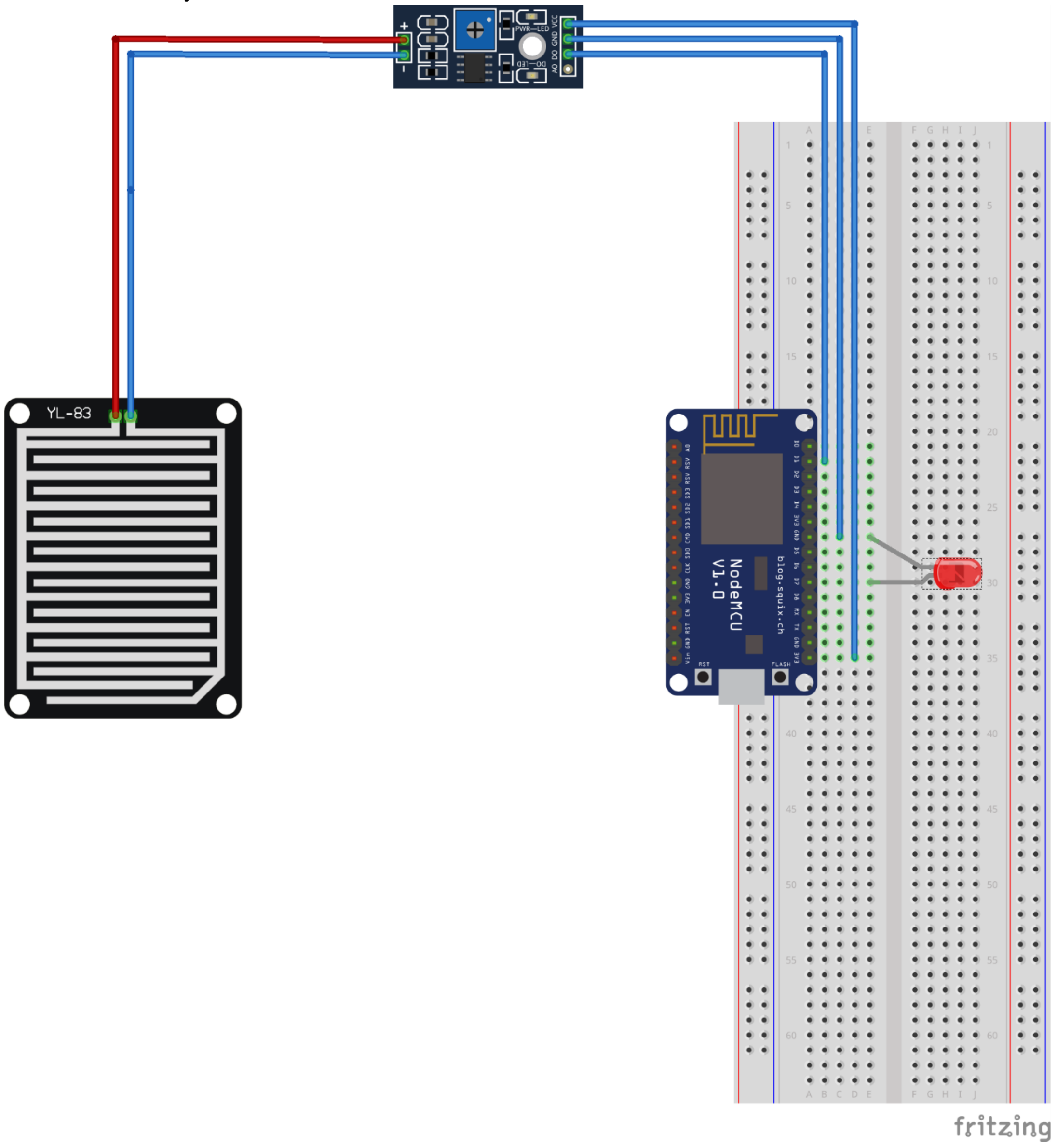
HARDWARE REQUIREMENTS:

- NODEMCU ESP8266 Board
- USB cable to connect NODEMCU board to CPU
- Connecting cables
- Rain shield
- Rain sensor
- RED colour LED

SOFTWARE REQUIREMENTS:

- Arduino IDE v1.8.8 Windows

BOARD DIAGRAM / CONNECTION DIAGRAM:



CODE:

```
#define pin D1
#define led D7
void setup()
{
	pinMode(pin,INPUT);
	pinMode(led,OUTPUT);
	Serial.begin(9600);
}
int v;
void loop()
{
	V=digitalRead(pin);
	Serial.println(v);
	if(v==0)
		digitalWrite(led,HIGH);
	else
		digitalWrite(led,LOW);
}
```

RESULTS AND ANALYSIS:

The serial monitor displays:
0 – initially when there is no rain, led doesn’t glow
1- when there is rain (water on rain shield), led glows

AIM 3: Send a notification to your mobile phone's Blynk app whenever there is rain.

DESCRIPTION: To display notification in Blynk app when it rains.

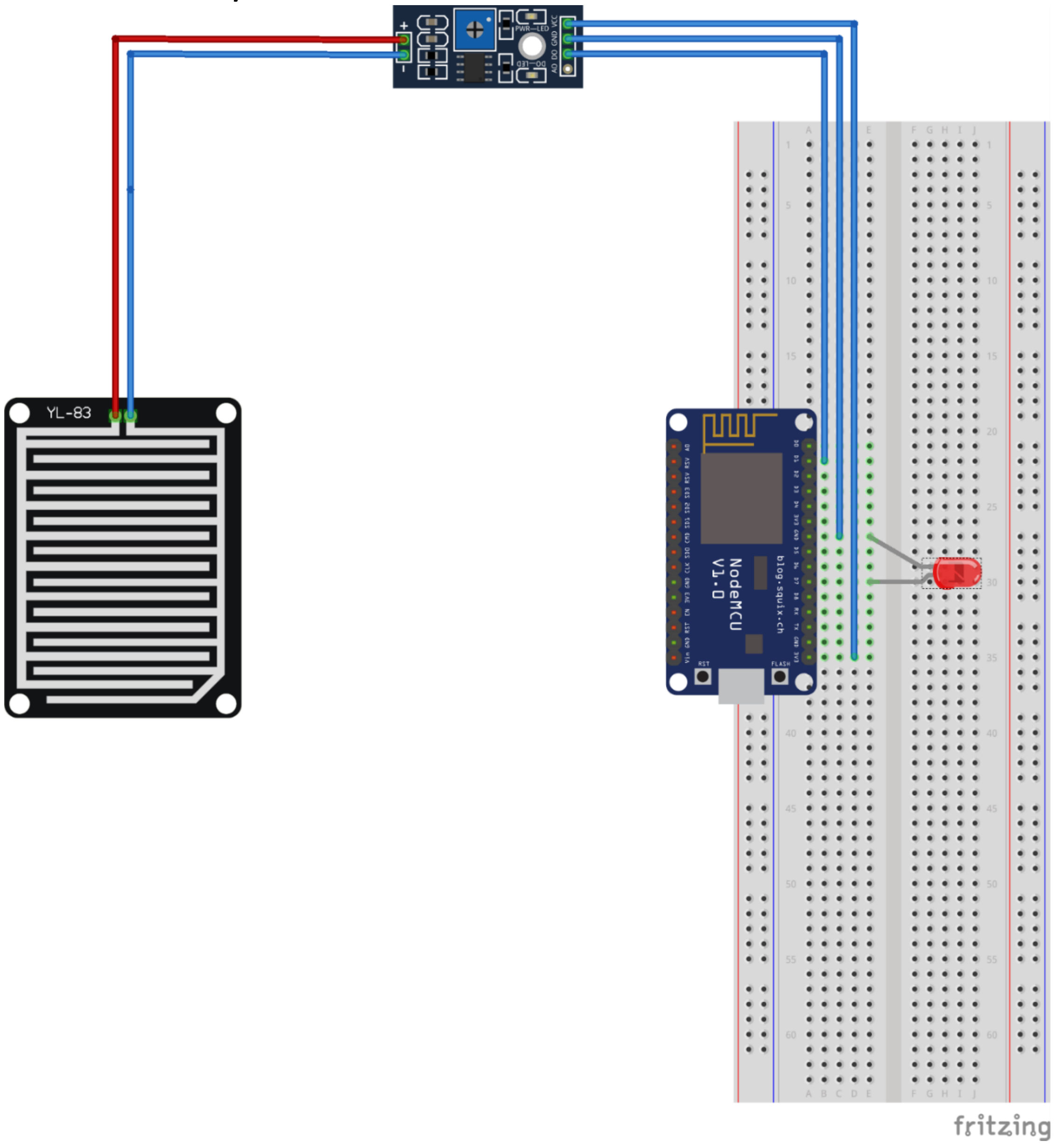
HARDWARE REQUIREMENTS:

- NODEMCU ESP8266 Board
- USB cable to connect NODEMCU board to CPU
- Connecting cables
- Rain shield
- Rain sensor

SOFTWARE REQUIREMENTS:

- Arduino IDE v1.8.8 Windows
- Blynk app

BOARD DIAGRAM / CONNECTION DIAGRAM:



CODE:

```
#include<Blynk.h>
#define BLYNK_PRINT serial
#include<ESP8266WIFI.h>
#include<BlynkSimpleESP8266.h>
#define pin D1
#define led D7

char auth[] = “89sd654wqr154897985jy8”;
char ssid[] = “network_name”;
char pass[] = “network_password”;

void setup()
{
        pinMode(pin,INPUT);
        pinMode(led,OUTPUT);
        Serial.begin(9600);
        Blynk.begin(auth,ssid,pass);
}
int v;
void loop()
{
        V=digitalRead(pin);
        Serial.println(v);

        if(v==0)
        {
                digitalWrite(led,HIGH);
                Blynk.notify(“It’s raining!, Happy New Year!!);
        }
        else
                digitalWrite(led,LOW);
        Blynk.run();
```

RESULTS AND ANALYSIS:

When it rains:

- Serial monitor displays 1
- LED glows
- Sends notification on Blynk app with a message “It’s raining!, Happy New Year!!

When it doesn’t rain:

- Serial monitor displays 0

Blynk app settings:

- Add notification
- Priority : high
- Enable notofications

EXPERIMENT – 4

AIM : To use ultrasonic sensor to get the distance between obstacle and sensor.

DESCRIPTION:

Ultrasonic sensors measure distance by using ultrasonic waves. The sensor head emits an ultrasonic wave and receives the wave reflected from the target. Ultrasonic Sensors measure the distance to the target by measuring the time between the emission and reception.

An Ultrasonic sensor is a device that can measure the distance of an object by using sound waves. It measures distance by sending out a sound wave at a specific frequency and waits for that sound wave to bounce back. By recording the time taken between the sound wave being generated and the sound wave bouncing back, it is possible to calculate the distance between the sensor and the object.

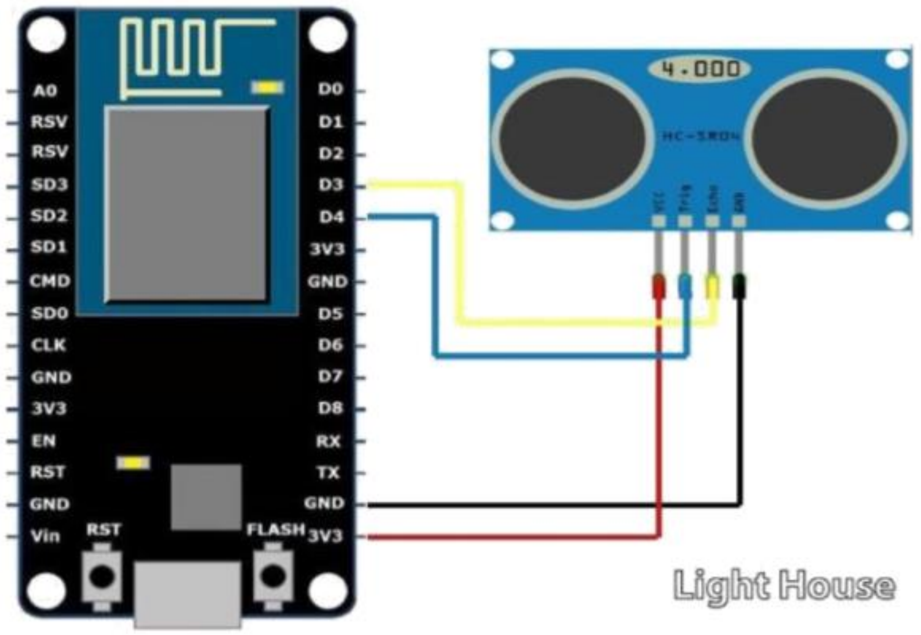
HARDWARE REQUIREMENTS:

- NODEMCU ESP8266 Board
- Bread Board
- LED
- Connecting wires
- Ultrasonic Sensor

SOFTWARE REQUIREMENTS:

- Arduino IDE v1.8.8 Windows.

BOARD/ CONNECTION DIAGRAM:

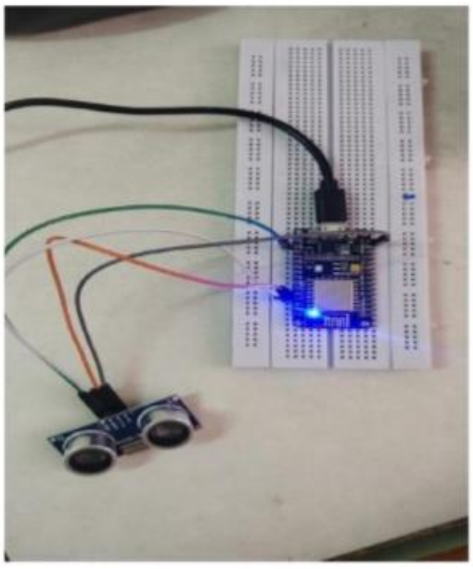


CODE:

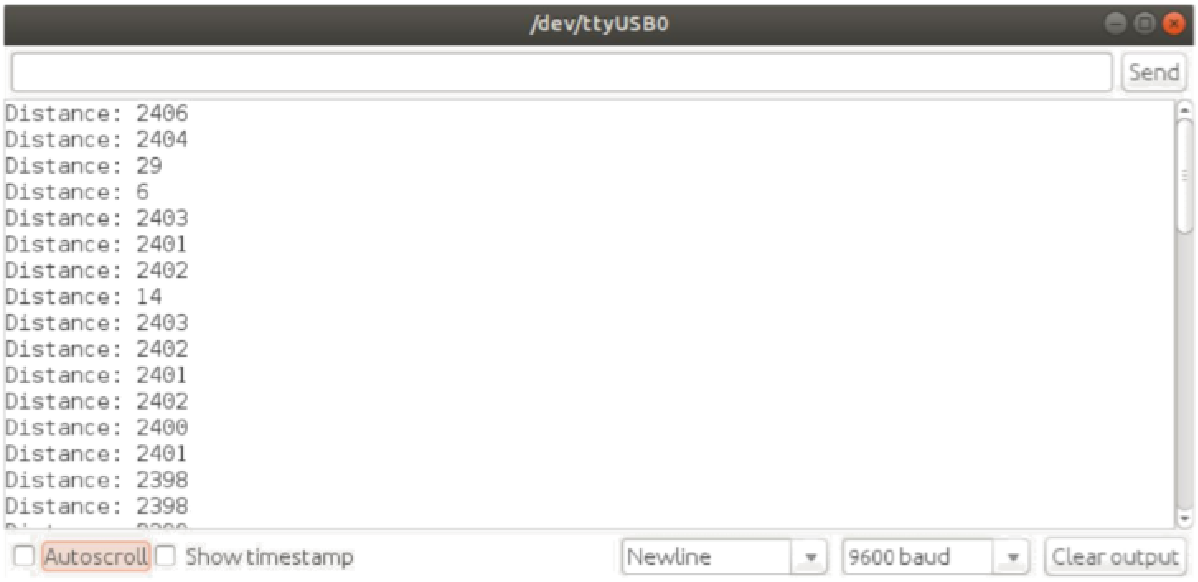
```
#define TRIGGER 5
#define ECHO 4

void setup()
{
        Serial.begin(9600);
        pinMode(TRIGGER,OUTPUT);
        pinMode(ECHO,INPUT);
}

void loop()
{
        long distance, duration;
        digitalWrite(TRIGGER, LOW);
        delayMicroseconds(2);
        digitalWrite(TRIGGER, HIGH);
        delayMicroseconds(10);
        digitalWrite(TRIGGER, LOW);
        duration = pulseIn(ECHO, HIGH);
        distance = ((duration/2)/29.1);
        Serial.print(distance);
        Serial.print("cms");
        delay(1000);
}
```

RESULTS AND ANALYSIS:

The serial monitor displays the distance between sensor and obstacle. Eg: 72 cms



Serial Monitor:



```
Distance: 2406
Distance: 2404
Distance: 29
Distance: 6
Distance: 2403
Distance: 2401
Distance: 2402
Distance: 14
Distance: 2403
Distance: 2402
Distance: 2401
Distance: 2402
Distance: 2400
Distance: 2401
Distance: 2398
Distance: 2398
```

AIM 2: To display the distances on the Blynk app using LCD display.

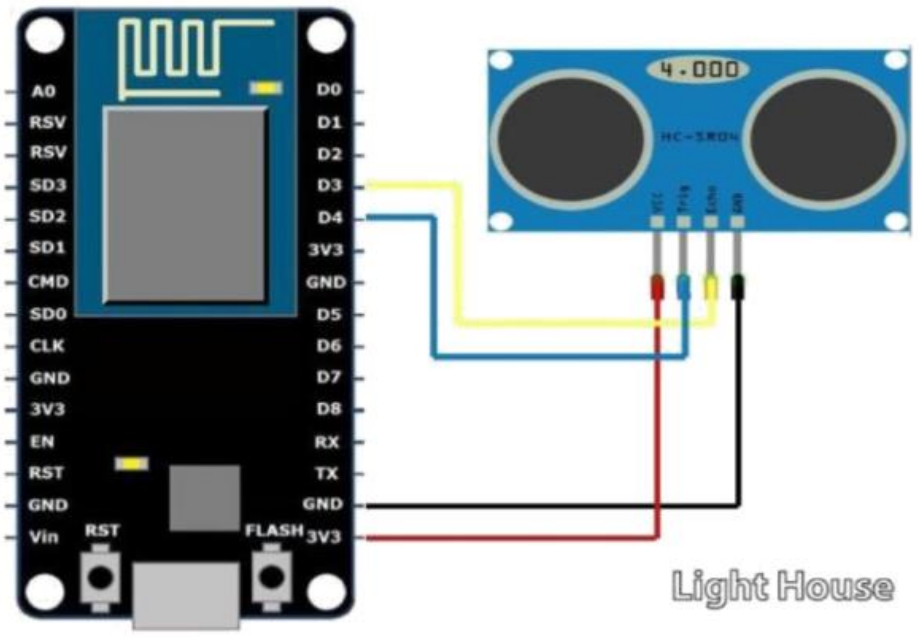
HARDWARE REQUIREMENTS:

- NODEMCU ESP8266 Board
- Bread Board
- LED
- Connecting wires
- Ultrasonic Sensor

SOFTWARE REQUIREMENTS:

- Arduino IDE v1.8.8 Windows.

BOARD/ CONNECTION DIAGRAM:



CODE:

```
#include<Blynk.h>
#define BLYNK_PRINT Serial
#include<ESP8266WIFI.h>
#define TRIGGER D1
#define ECHO D2
#define LED D7

char auth[] = “auth token number”
char ssid[] = “network name”
char pass[] = “network password”

WidgetLCD lcd(V1);
```

```
void setup()
{
        Serial.begin(9600);
        pinMode(TRIGGER, OUTPUT);
        pinMode(ECHO, INPUT);
        pinMode(LED, OUTPUT);
        Blynk.begin(auth, ssid, pass);
        lcd.char();
        lcd.print(0, 0, “Distance in cm”);
}

void loop()
{
        lcd.clear();
        lcd.print(0, 0, “Distance in cm”);
        long distance, duration;
        digitalWrite(TRIGGER, LOW);
        delayMicroseconds(2);
        digitalWrite(TRIGGER, HIGH);
        delayMicroseconds(10);
        digitalWrite(TRIGGER, LOW);
        duration = pulseIn(ECHO, HIGH);
        distance = ((duration/2)/29.1);

        if(distance>1000)
                digitalWrite(LED, HIGH);
        else
                digitalWrite(LED, LOW);
        Serial.print(distance);
        Serial.print(“cms”);
        lcd.print(7, 1, distance);
        Blynk.run();
        delay(3500);
}
```

Blynk App Settings:

1. Add “LCD” widget
2. Input : Advanced
    a. Input (select pin) : Virtual = V1

RESULTS AND ANALYSIS:

The distance between the sensor and obstacle is displayed on LCD (Blynk App)

AIM 3: Interfacing accelerometer with NodeMCU

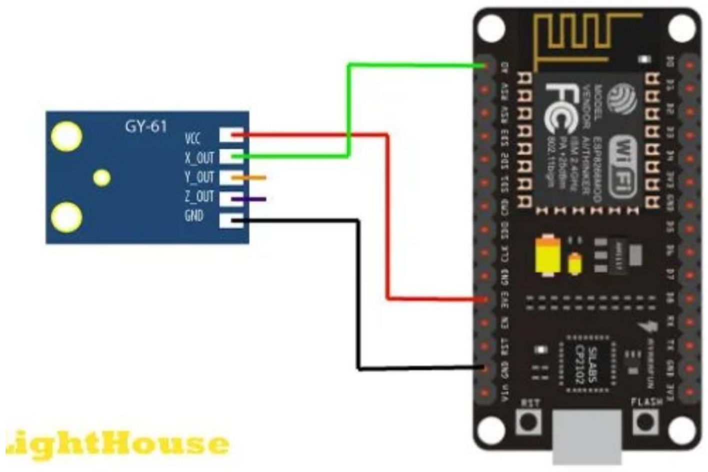
HARDWARE REQUIREMENTS:

- NODEMCU ESP8266 Board
- Bread Board
- LED
- Connecting wires
- Accelerometer

SOFTWARE REQUIREMENTS:

- Arduino IDE v1.8.8 Windows.

BOARD/ CONNECTION DIAGRAM:



CODE:

```
const int xPin = A0;
void setup()
{
        Serial.begin(9600);
}

void loop()
{
        int x = analogRead(xPin);
        delay(100);
        Serial.print(“X axis :”);
        Serial.println(x);
}
```

RESULTS AND ANALYSIS:

Since there is only one analog pin (A0) in NodeMCU, we can find results for a single axis, i.e. x-axis or y-axis or z-axis at a time.

OUTPUT (on serial monitor) :

```
X axis : 7
X axis : 10
X axis : 220
X axis :348
```

EXPERIMENT – 5

AIM 1 : Interfacing smoke sensor with NodeMCU

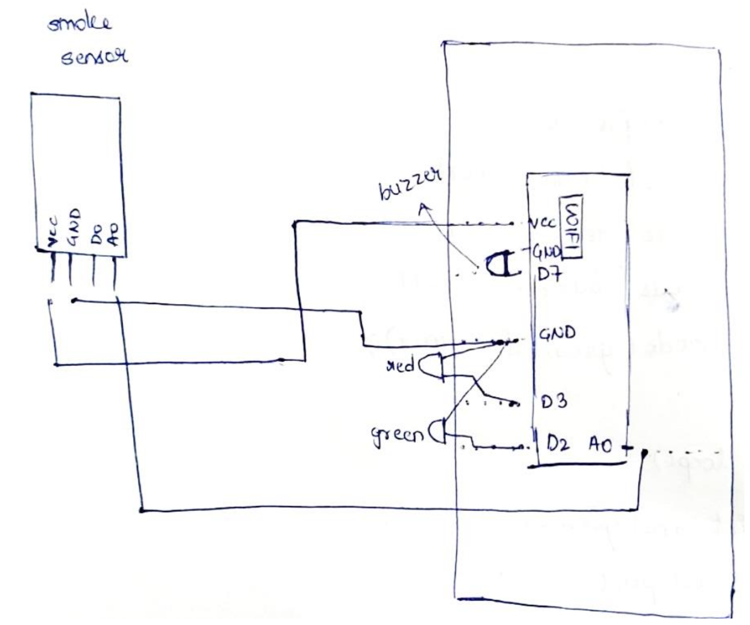
HARDWARE REQUIREMENTS:

- NODEMCU ESP8266 Board
- Bread Board
- LED – green and red
- Connecting wires
- Smoke Sensor
- Buzzer

SOFTWARE REQUIREMENTS:

- Arduino IDE v1.8.8 Windows.

BOARD/ CONNECTION DIAGRAM:



CODE:

```
int smoke = A0;
int redLed = 0;
int greenLed = 4;
int buzzer = 13;
int sensorThreshold = 300;
void setup()
{
        Serial.begin(9600);
        pinMode(smoke, INPUT);
        pinMode(redLed, OUTPUT);
        pinMode(greenLed, OUTPUT);
        pinMode(buzzer, OUTPUT);
}
void loop()
{
        int analogSensor = analogRead(smoke);
        Serial.print(" Print A0");
        Serial.println(analogSensor);
        if(analogSensor > sensorThreshold)
        {
                digitalWrite(redLed,HIGH);
                digitalWrite(greenLed,LOW);
                tone(buzzer, 1000, 200);
        }
        else
        {
                digitalWrite(redLed,LOW);
                digitalWrite(greenLed,HIGH);
                notone(buzzer);
        }
```

RESULTS AND ANALYSIS:

When value detected by the smoke sensor is greater than the threshold value:

1. Red led glows
2. Green led fades
3. Buzzer buzzes , and vice-versa

AIM 2: Interfacing PIR sensor with NodeMCU

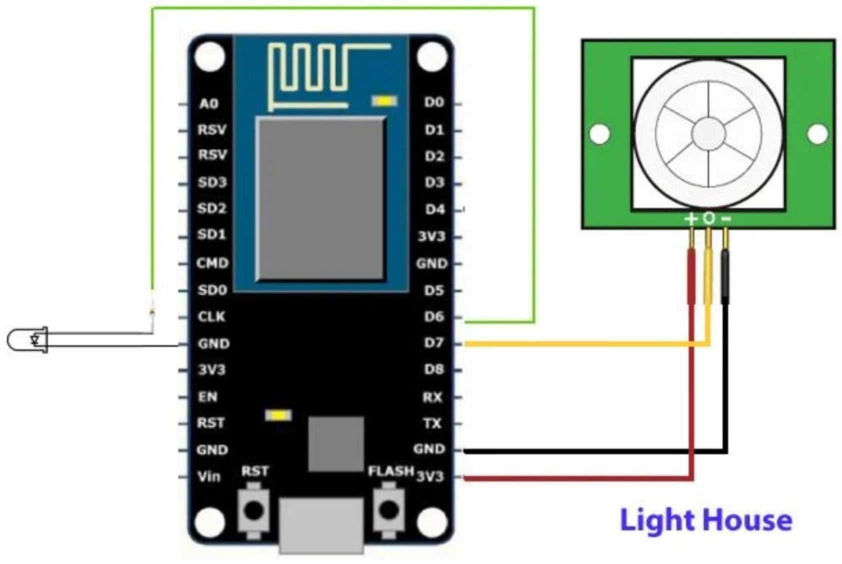
HARDWARE REQUIREMENTS:

- NODEMCU ESP8266 Board
- Bread Board
- LED
- Connecting wires
- PIR Sensor

SOFTWARE REQUIREMENTS:

- Arduino IDE v1.8.8 Windows.

BOARD/ CONNECTION DIAGRAM:



CODE:

```
int led = 13;
int sensor = 2;
int state = LOW;
int value = 0;
void setup()
{
        pinMode(led,OUTPUT);
        pinMode(sensor,INPUT);
        Serial.begin(9600);
}
void loop()
{
        value.digitalRead(sensor);
        Serial.println(value);
        if(value == HIGH)
        {
```

```
                    digitalWrite(led, HIGH);
                    delay(100);

                    if(state ==LOW)
                    {
                              Serial.println(“Motion detected”);
                              state=HIGH;
                    }
          }
          else
          {
                    digitalWrite(led,LOW);
                    delay(600);
                    if(state ==LOW)
                    {
                              Serial.println(“Motion detected”);
                              state=HIGH;
                    }
          }
}
```

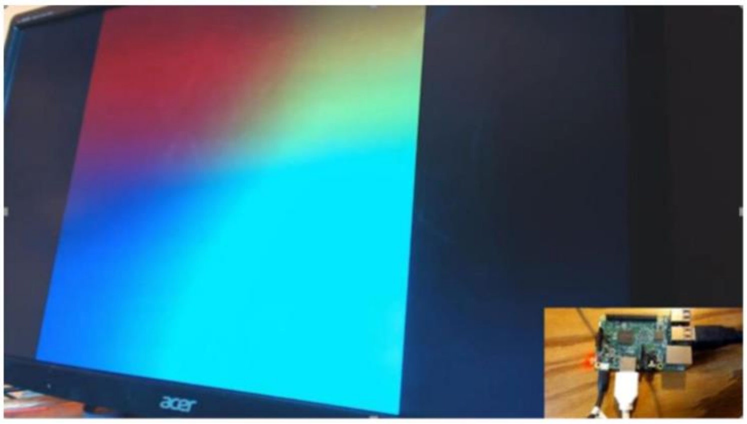
RESULTS AND ANALYSIS:
Values displayed on serial monitor is either 0 or 1.

EXPERIMENT – 7

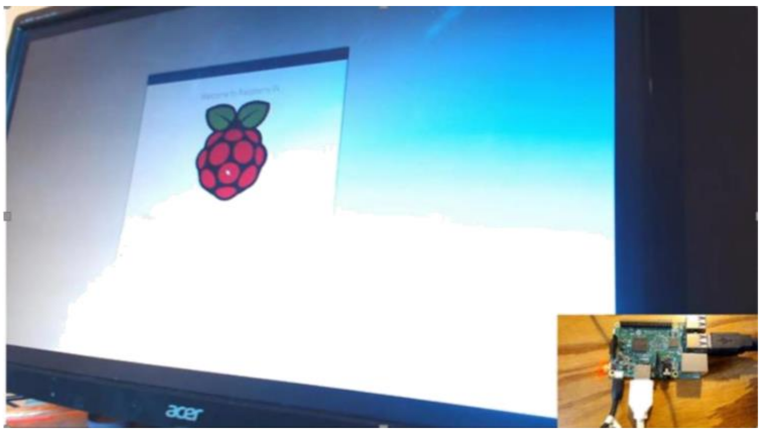
AIM 1: Raspberry Pi installation

Raspberry Pi installation

1.) Download raspbian from the below site: https://www.raspberrypi.org/downloads/
2.) Unzip the downloaded folder. You will find an image (.img file) in it.
3.) We need to write the image file to an SD card using Etcher software. This SD card will be plugged into the raspberry pi module.
4.) So first, to install the image file onto the SD card plug the SD card into a card reader and format it.
5.) Download Etcher software from the below site and install it in on your PC https://www.balena.io/etcher/
6.) Open Etcher
7.) Select the OS image file that was downloaded and unzipped earlier.
8.) Add select Flash.
9.) This will install the Raspbian OS onto the SD card.
10.)Plug the SD card into the raspberry pi module, connect the HDMI cable of monitor to pi module and connect the power cable.
11.)The screen will appear as below:



E



Simple program using Raspberry Pi and Python
1.) Connect to the internet through WiFi or Ethernet cable
2.) Open the terminal
3.) Update the packages using: sudo apt-get upgrade
4.) Use the below link for reference.
https://pythonprogramming.net/introduction-raspberry-pi-tutorials/

AIM 2: Make an LED blink

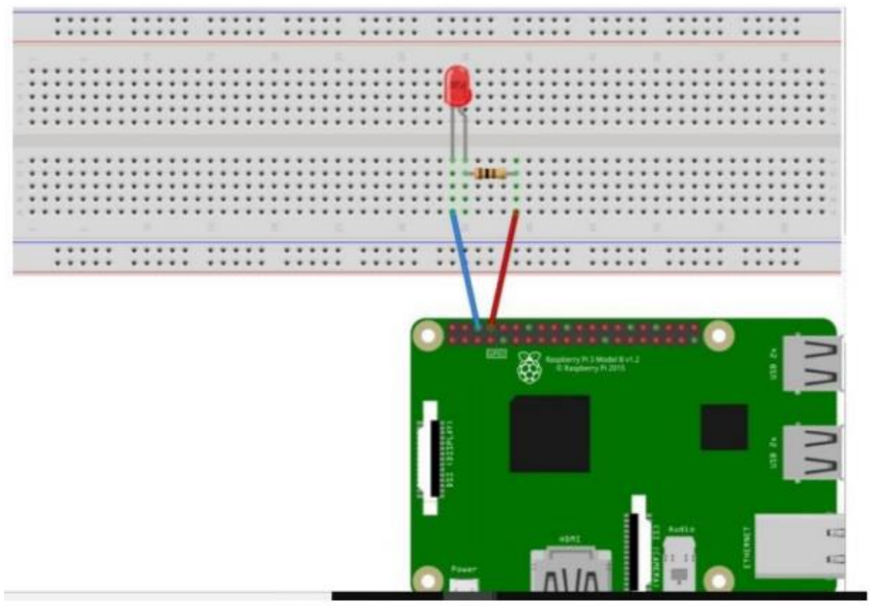
DESCRIPTION: On/OFF an LED using Raspberry Pi module and Python Programming
HARDWARE REQUIREMENTS:

- Raspberry Pi 3 B+ module
- Bread Board
- LED
- Connecting wires

SOFTWARE REQUIREMENTS:

- Raspbian OS
- Python

BOARD/ CONNECTION DIAGRAM:



PROGRAMMING:
Install the python library for pi using:

```
$ sudo apt-get install python-rpi.gpio python3-rpi.gpio
```

CODE: (file name: blinking_led.py):

```
import RPi.GPIO as GPIO                 # Import Raspberry Pi GPIO library
from time import sleep                  # Import the sleep function from the time module

GPIO.setwarnings(False)                 # Ignore warning for now
GPIO.setmode(GPIO.BOARD)                # Use physical pin numbering
# Set pin 8 to be an output pin and set initial value to low (off)
GPIO.setup(8, GPIO.OUT, initial=GPIO.LOW)
while True:                                                   # Run forever
        GPIO.output(8, GPIO.HIGH)                             # Turn on
        sleep(1)                                            # Sleep for 1 second
        GPIO.output(8, GPIO.LOW)                            # Turn off
        sleep(1)                                            # Sleep for 1 second
```

RUN THE PROGRAM:

```
$ python blinking_led.py
```

EXPERIMENT – 8

AIM 1: To use rain sensor to detect the rain and to notify the user.

HARDWARE REQUIREMENTS:

- Rain sensor
- Rain shield
- Raspberry pi
- Buzzer

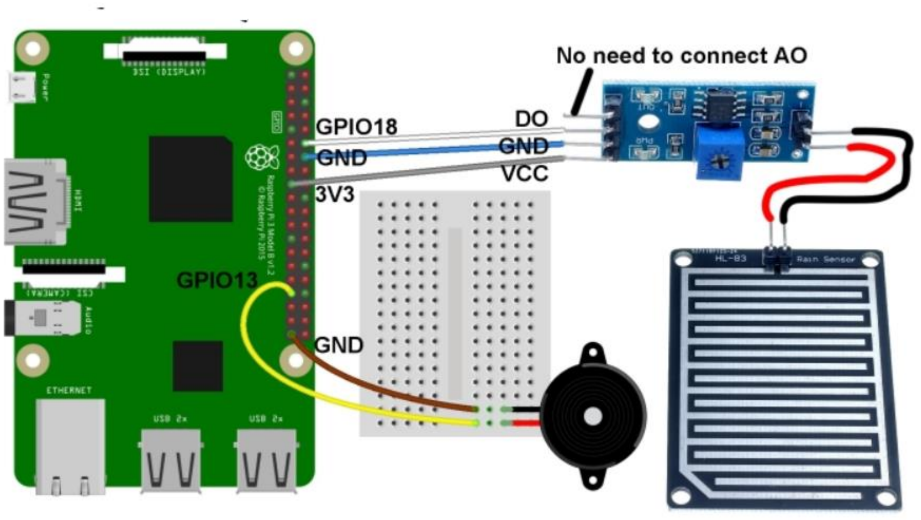
SOFTWARE REQUIREMENTS

- Raspbian OS
- Python

DESCRIPTION

We use the rain shield to sense the rain and we notify the user in case of heavy rain. The buzzer rings in case the threshold value is reached or exceeded.

CIRCUIT DIAGRAM:



CODE:

```
from time import sleep
from gpiozero import Buzzer, InputDevice
buzz=Buzzer(13)
no_rain=InputDevice(18)
def buzz_now(iterations):
        for x in range(iterations):
                buzz.on()
                sleep(1)
                buzz.off()
                sleep(1)
        while(True):
                if  not  no_rain  is_active:
                        print(“Rainin”)
                        buzz_now(5)
                        sleep(1)
```

RESULTS ANS ANALYSIS:

Thus the buzzer rings when it rains and alerts the user.

AIM 2: To demonstrate traffic lights using led and Raspberry pi

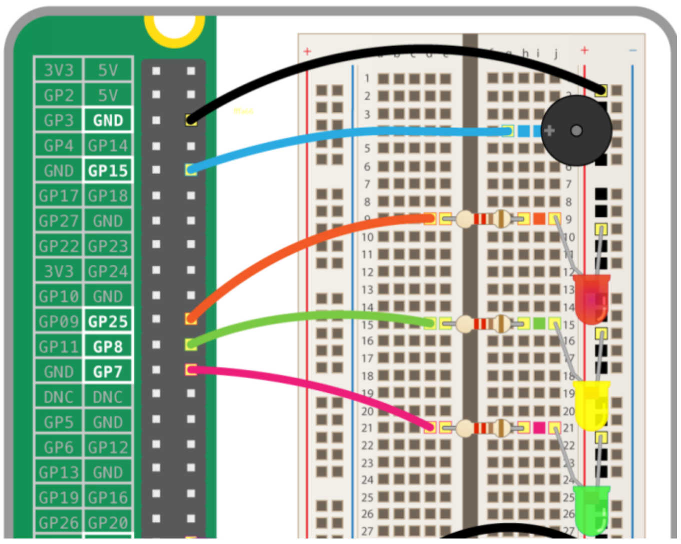
HARDWARE REQUIREMENTS:

- Rain sensor
- Rain shield
- Raspberry pi
- Buzzer

SOFTWARE REQUIREMENTS

- Raspbian OS
- Python

CIRCUIT DIAGRAM:



CODE:

```
import RPI.GPIO as GPIO
from  time import sleep
from gpiozero import Button, LED, TrafficLights, Buzzer
GPIO.setMode(GPIO.BCM)

lights=TrafficLights(25,8,7)
GPIO_TRIGGER=18
GPIO_ECHO=24

GPIO.setup(GPIO_TRIGGER,GPIO.OUT)
GPIO.setup(GPIO_ECHO,GPIO.IN)

def distance():
        #set trigger to high
        GPIO.output(GPIO_TRIGGER,True)
        #set trigger after 0.01ms to LOW
        Time.sleep(0.00001)
        GPIO.output(GPIO_TRIGGER,False)

        startTime = time.time()
        stopTime = time.time()

        #save start time
        while GPIO.input(GPIO_ECHO) == 0:
                startTime = time.time()

        #save time of arrival
        while GPIO.input(GPIO_ECHO) == 1:
                stopTime = time.time()

        #time difference between start and arrival
        timeElapsed = stopTime – startTime

        #multiply with sonic speed (34300 cm/s) and divide by 2 (to and fro)
        Distance = (timeElapsed * 34300)/2
        return distance

__name__ == “__main__”
try:
        while True:
                dist=distance()
                print(“measured distance = %.1f cm” % dist)
                if(dist<10):
                        lights.amber.on()
                        time.sleep(3)
                        lights.off()
                        lights.green.on()
                        time.sleep(3)
                        lights.off()
                else:
                        lights.red.on()
                        time.sleep(3)
                        lights.off()
        time.sleep(8)
except KeyboardInterrupt
        print(“Measurement stopped by user”)
        GPIO.cleanup()
```

RESULTS AND ANALYSIS:

Thus, the red, green and yellow leds glow at different intervals and stimulate actual traffic lights

EXPERIMENT – 9

AIM 1: To detect smoke using the mq-2 sensor and raspbian OS

DESCRIPTION: We use the mq-2 sensor to sense the smoke and we notify the user in case of heavy smoke. The buzzer rings in case the threshold value is reached or exceeded.

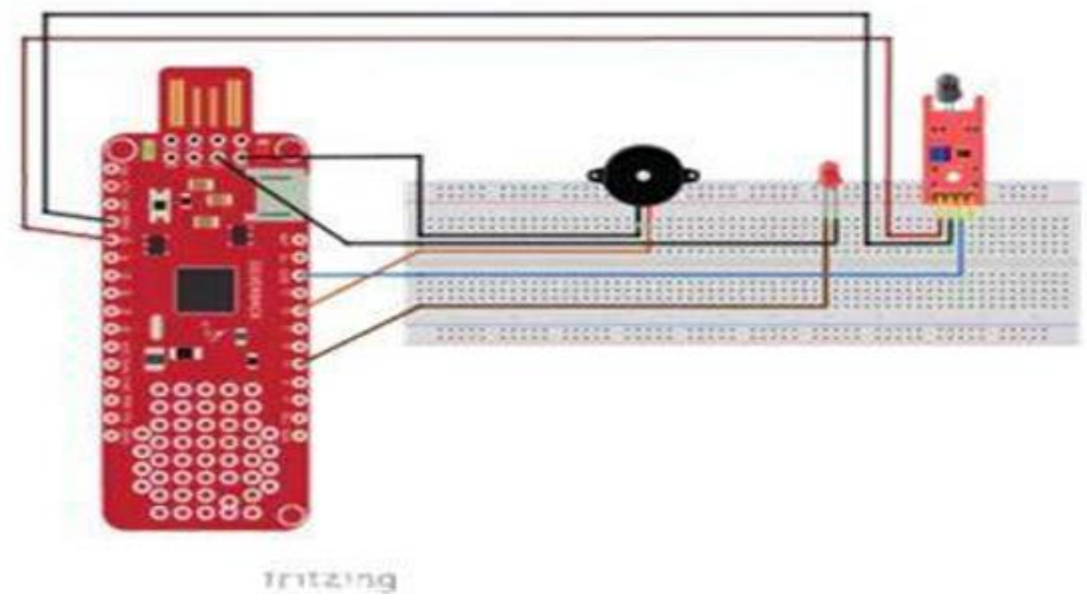
HARDWARE REQUIREMENTS:

- Mq-2 gas sensor
- raspberry pi
- buzzer
- jumper wires

SOFTWARE REQUIREMENTS:

- Raspbian OS
- Python

CIRCUIT DIAGRAM:



CODE:

```
import gpiozero
import time
import botbook_mcp3002  as mcp #
buzz=Buzzer(15)
smoke=0
def readval():
        global smoke
        smoke=mcp.readAnalog()
def main():
        while True:
                readval()
                print(“Current smoke level”,smoke)
        if smoke>120:
                print(“Heavy smoke”)
                buzz.beep()
        time.sleep(1)
if__name__==”__main__”:
        main()
```

RESULTS AND ANALYSIS:

Thus the smoke level is detected using smoke sensor and the value is displayed on the monitor. Also, if the detected value is greater than the threshold the user is alerted using a buzzer.

AIM 2: To test the quality of the soil using the soil moisture sensor

DESCRIPTION: The soil moisture is tested to give the right amount of water to the plants. Thus, we find the moisture using moisture sensor.

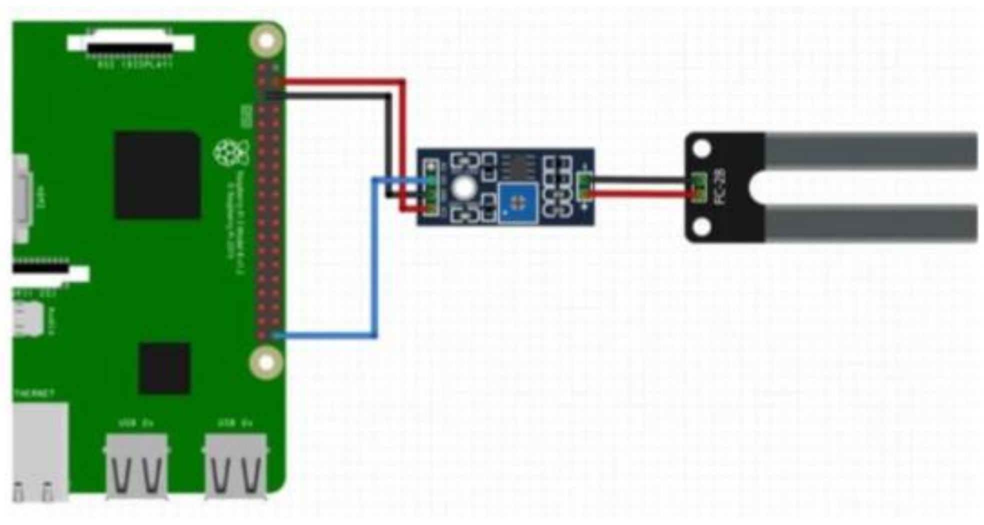
HARDWARE REQUIREMENTS:

- Soil moisture sensor
- raspberry pi
- buzzer
- jumper wires

SOFTWARE REQUIREMENTS:

- Raspbian OS
- Python

CIRCUIT DIAGRAM:



CODE:

```
import RPi. GPIO as GPIO
from gpioxero import inputDevice
from tie import sleep
GPIO.setmode(GPIO.BCM)
ip=inputDevice(7)
while True:
 if ip.is_active:
        print(“Water is not detected”)
        sleep(1)
else:
        print(“Water is detected”)
        sleep(1)
```

RESULTS AND ANALYSIS: Thus the moisture content in the soil is detected using the soil moisture sensor.