VARDHAMAN COLLEGE OF ENGINEERING

(AUTONOMOUS)

Shamshabad - 501 218, Hyderabad.

INDEX

Mr/~~Ms~~. ..... Roll Number 2028IA1270

S. No.	Date	Title of the Experiment	Page No.	Marks	Sign. of the Staff
1.		Raspberry Pi - LED Blink	1		
2.		NodeMCU - LED Blink	4		
3.		Raspberry Pi - Weather Monitoring	7		
4.		NodeMCU - Temperature Monitoring	10		
5.		Raspberry Pi - Automatic Street Lighting	13		
6.		NodeMCU - Automatic Street Lighting	16		
7.		NodeMCU - Distance Measurement	19		
8.		Raspberry Pi - Distance Measurement	22		
9.		Study Arduino, Types; Install Arduino IDE	25		
10.		Arduino - RGB LED	28		
12.		Study Raspberry Pi.	31		

Page No. 1	1. Implement LED Blink using Raspberry Pi.	Date
		Exp. No. 1

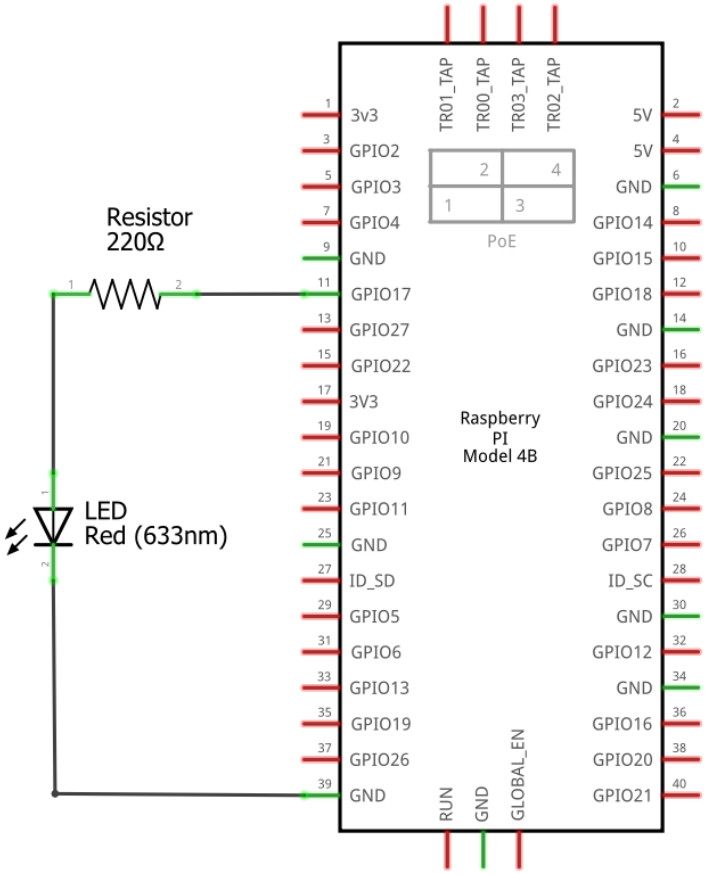
Aim: To write a program using Raspberry Pi for LED Blink.

Components:

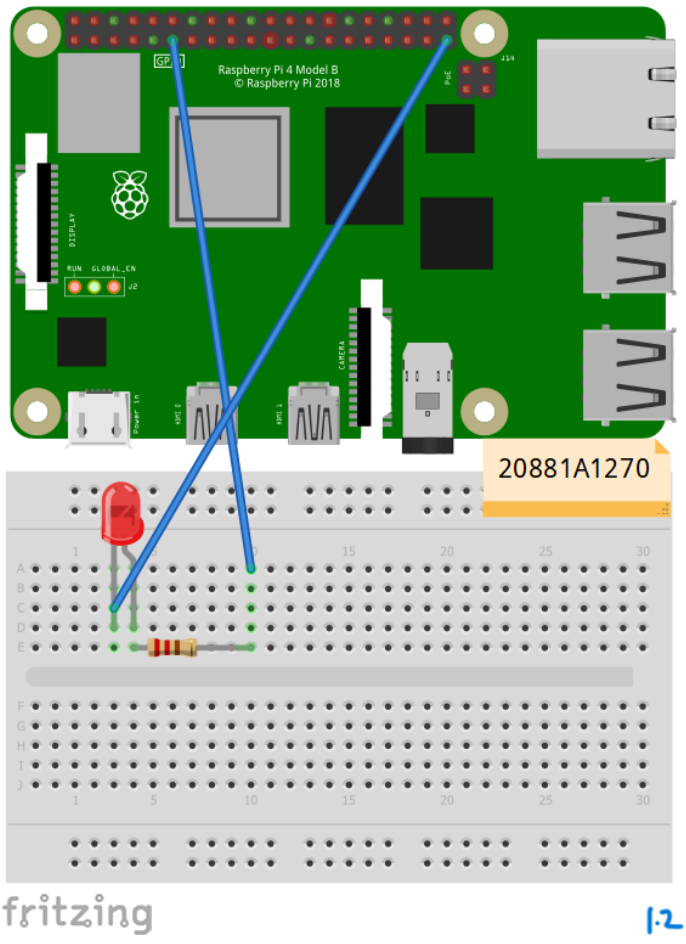
1. LED ×1
2. Resistor (220 Ω) ×1
3. Bread board ×1
4. Raspberry Pi 4 Model B ×1
5. Jumper wires (M-F) ×2
6. Misc: SD Card (with preloaded raspbian OS), Power Cable (Type-C), Micro HDMI to VGA Adapter, Monitor, Mouse, Keyboard.

Procedure:

- Place the LED on the Breadboard such that both the legs go into different rails. as shown in Fig. 1.2.
- Place the resistor on the breadboard such that the anode (+ve) side of LED and one leg of the resistor go into the same rail. as in the Fig 1.2
- Connect the cathode (-ve) side of LED to GND pin on the Raspberry Pi.
- Connect the other end of the resistor to the GPIO 17 (11) of the Raspberry Pi as shown in Fig. 1.1.


Resistor
220Ω
LED
Red (633nm)
Raspberry
PI
Model 4B
3v3
GPIO2
GPIO3
GPIO4
GND
GPIO17
GPIO27
GPIO22
3V3
GPIO10
GPIO9
GPIO11
GND
ID_SD
GPIO5
GPIO6
GPIO13
GPIO19
GPIO26
GND
5V
5V
GND
GPIO14
GPIO15
GPIO18
GND
GPIO23
GPIO24
GND
GPIO25
GPIO8
GPIO7
ID_SC
GND
GPIO12
GND
GPIO16
GPIO20
GPIO21
TR01_TAP
TR00_TAP
TR03_TAP
TR02_TAP
PoE
RUN
GND
GLOBAL_EN
fritzing
20881A1270

1.1


Raspberry Pi 4 Model B
© Raspberry Pi 2018
20881A1270
fritzing

1.2

Source code:

```
print (" This program is written by : 20881A1270 ")
import RPi.GPIO as GPIO
import time

led = 11

GPIO. setmode (GPIO. BOARD)
GPIO. setwarnings (False)
GPIO. setup (led, GPIO.OUT)

while True:
    GPIO . output (led, GPIO.HIGH)
    print ("LED ON")
    time.sleep (1)
    GPIO. output ( led, GPIO. LOW)
    print ( "LED OFF")
    time.sleep (1).
```

Working Principle:

- When the circuit is connected according to the Fig. 1.2, the LED is connected to a GPIO pin via a resistor and other end of LED is connected to GND.

- When the loop begins, the GPIO pin connected to the LED is set to high. i-e, LED will be on.
- after a delay of 1 second, the GPIO pin will be set to low. i-e, LED will be off.
- The loop will repeat infinitely until the power is removed to the Raspberry Pi.

u.py | 1-LED_Blink.py | 2-LDR.py | ldr.py

```
1  # LED_Blink
2  print("This program is written by : [redacted]- 20881A1270, [redacted]- 20881A1277")
3
4  import RPi.GPIO as GPIO
5  import time
6
7  led = 11
8
9  GPIO.setmode(GPIO.BOARD)
10 GPIO.setwarnings(False)
11 GPIO.setup(led, GPIO.OUT)
12
13 while True:
14     GPIO.output(led,GPIO.HIGH)
15     print("LED on")
16     time.sleep(1)
17     GPIO.output(led,GPIO.LOW)
18     print("LED off")
19     time.sleep(1)
```

Shell

```
This program is written by : [redacted]- 20881A1270, [redacted]- 20881A1277
LED on
LED off
LED on
LED off
LED on
```

2. Implement LED Blink using NodeMCU

Date

Exp. No. 2

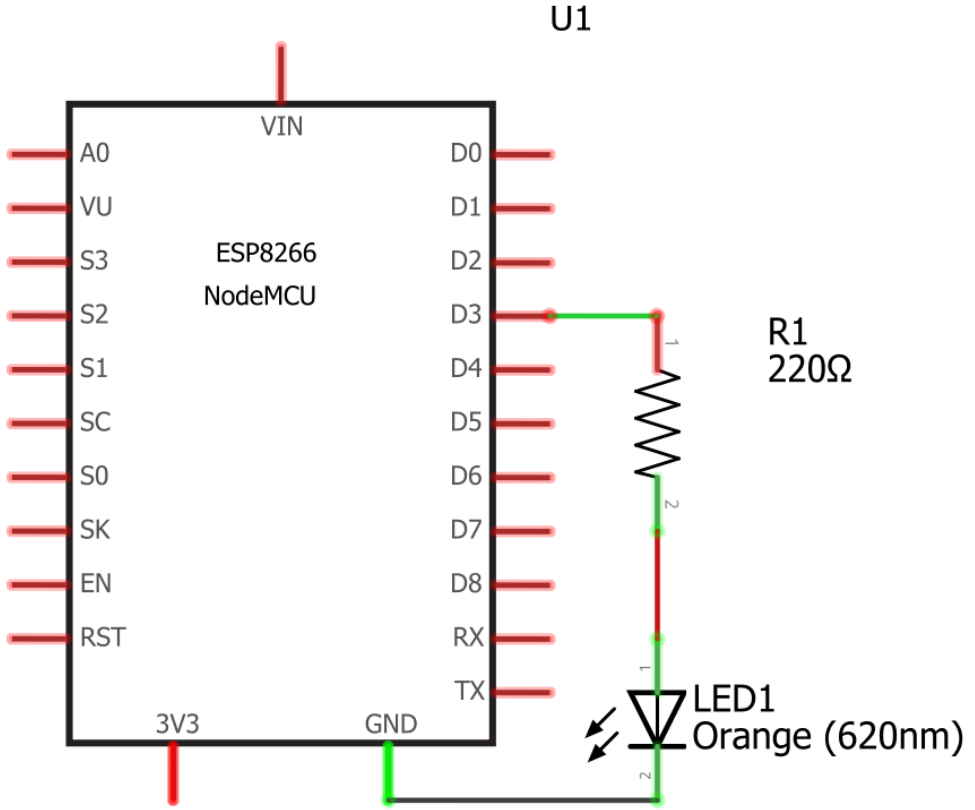
Aim: To write a program using Arduino IDE for NodeMCU to Blink LED.

Components:

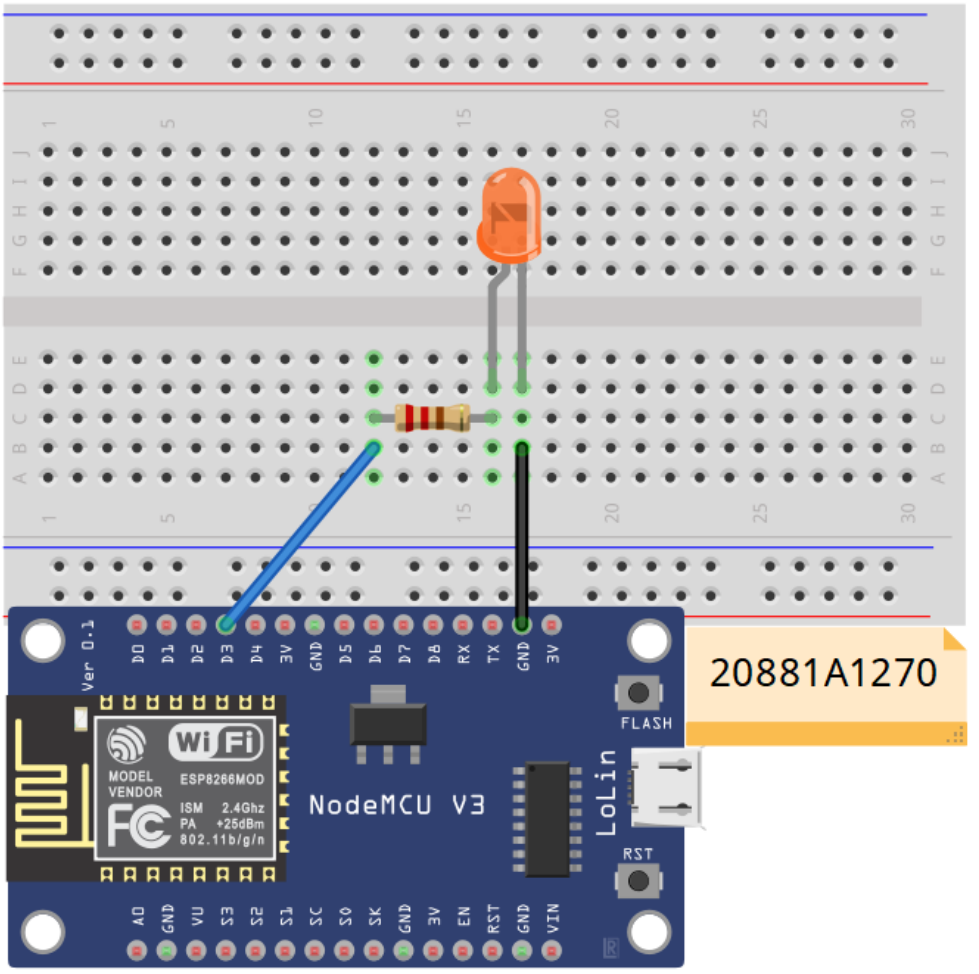
1. LED
2. Resistor (220Ω)
3. Breadboard
4. ESP8266 NodeMCU.
5. Jumper wires (M-F)
6. Misc: Computer (installed with Arduino IDE), USB Connecter (Type-A to micro USB).

Procedure:

- Place the NodeMCU on the bread board such that every pin goes into different rail. as shown in Fig 2.2.
- Place the LED on the breadboard such that both legs go into different rails and connect the resistor to the anode side of the LED as shown in Fig 2.2.
- Connect the cathode side of LED to GND and other side of the resistor to D3 digital pin of the NodeMCU as shown in Fig 2.1.


U1
VIN
A0
VU
S3
S2
S1
SC
S0
SK
EN
RST
ESP8266
NodeMCU
D0
D1
D2
D3
D4
D5
D6
D7
D8
RX
TX
3V3
GND
R1
220Ω
LED1
Orange (620nm)
fritzing
20881A1270

2.1


20881A1270
NodeMCU V3
LoLin
FLASH
RST
fritzing

2.2

Source code :

```
void setup()
{
    pinMode (D3, OUTPUT);
    Serial.begin (74880);
}
void loop()
{
    Serial.print("This program is written by : 20881A1270");
    digitalWrite (D3, HIGH);
    delay(100);
    digitalWrite (D3, LOW);
    delay(100);
}
```

Working principle :

- When the circuit is connected as shown in the Fig 2.2, as the LED is connected to a digital pin D3, it can be digitally written to HIGH and LOW.
- When the power is supplied to the NodeMCU and code is uploaded correctly, the loop starts with setting the LED to HIGH and waits for 100 ms and sets it to LOW again and waits for 100 ms.

- The process is repeated infinitely until the power supply is removed and simulates LED Blink.

LED_BLINK | Ardui

File Edit Sketch Tools Help

LED_BLINK

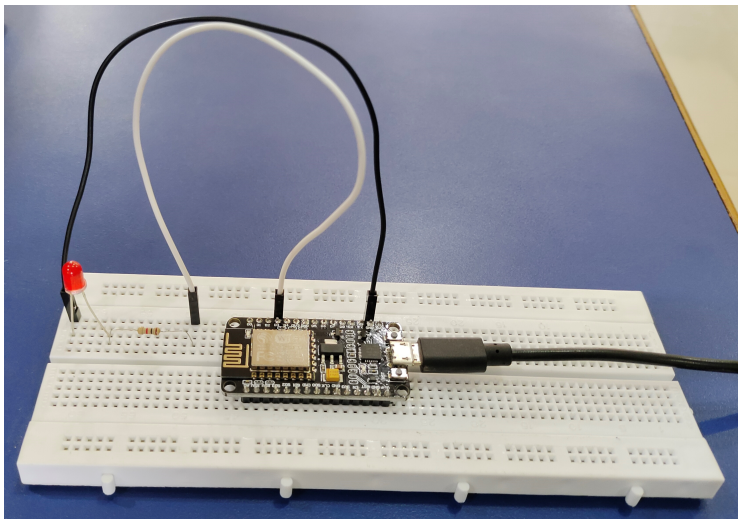
```
void setup()
{
  pinMode(D3,OUTPUT);
  Serial.begin(74880);
}
void loop()
{
  Serial.print("This program is written by: 20881A1262, 20881A1270, 20881A1277");
  digitalWrite(D3,HIGH);
  delay(100);
  digitalWrite(D3,LOW);
  delay(100);
}
```

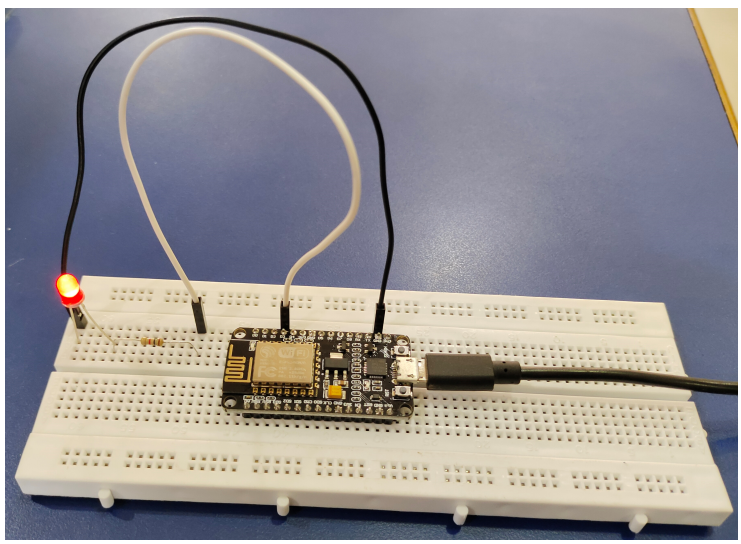
COM3

Send

This program is written by: 20881A1262, 20881A1270, 20881A1277This program is writt

Autoscroll Show timestamp | No line ending | 74880 baud | Clear output





Page No. 7	3. Implement IoT based weather monitoring system using Raspberry Pi.	Date
		Exp. No. 3

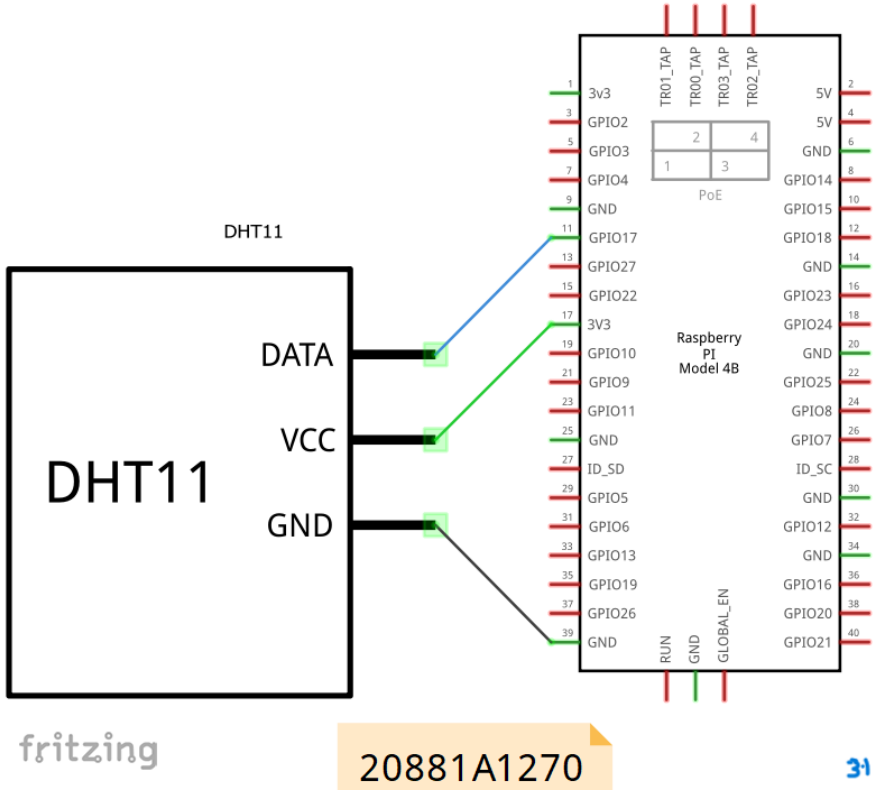
Aim: To write program for weather monitoring system using Raspberry Pi and implement it under IoT.

Components:

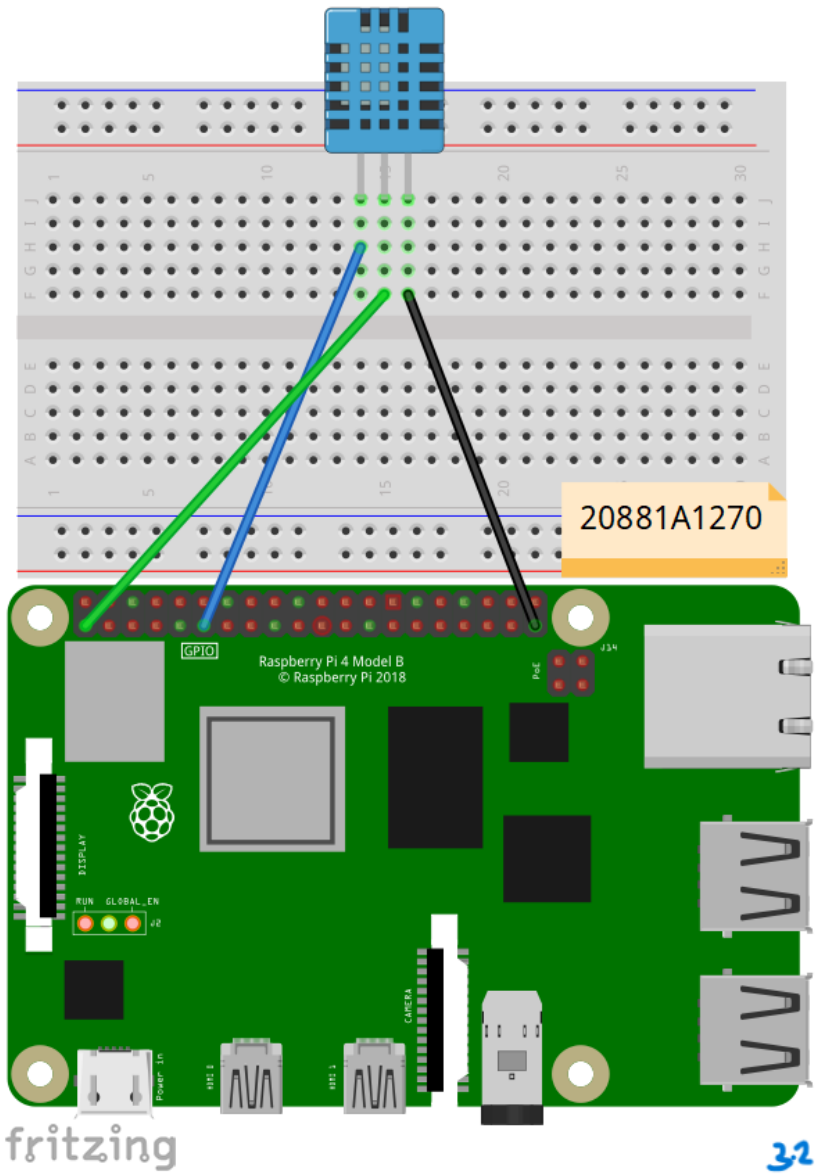
1. DHT11 Sensor ×1
2. Resistor (220 Ω) ×1
3. Breadboard ×1
4. Raspberry Pi 4 Model B ×1
5. Jumper Wires (M-F) ×3
6. Misc: SD Card (preloaded with Raspbian OS), Power Cable (Type C), Ethernet, micro HDMI to VGA Converter, Keyboard, Mouse, Monitor etc.

Procedure:

- Place the DHT11 Sensor on the breadboard such that the three pins of the sensor go into different rails.
- Connect the DATA pin of the sensor to the GPIO17 pin of the Raspberry Pi as shown in Fig 3.1
- Connect the VCC pin of the sensor to +5V and GND pin of the sensor to GND pins of the Raspberry Pi as shown in Fig 3.1


DHT11
DATA
VCC
GND
DHT11
Raspberry
PI
Model 4B
3v3
GPIO2
GPIO3
GPIO4
GND
GPIO17
GPIO27
GPIO22
3V3
GPIO10
GPIO9
GPIO11
GND
ID_SD
GPIO5
GPIO6
GPIO13
GPIO19
GPIO26
GND
5V
5V
GND
GPIO14
GPIO15
GPIO18
GND
GPIO23
GPIO24
GND
GPIO25
GPIO8
GPIO7
ID_SC
GND
GPIO12
GND
GPIO16
GPIO20
GPIO21
TR01_TAP
TR00_TAP
TR03_TAP
TR02_TAP
PoE
RUN
GND
GLOBAL_EN
fritzing
20881A1270

31


20881A1270
GPIO
Raspberry Pi 4 Model B
© Raspberry Pi 2018
PoE
DISPLAY
RUN
GLOBAL_EN
CAMERA
Power in
fritzing

32

Source code:

```
print("This program is written by : 2088IA1270");

import time

import adafruit_dht

from board import *

SENSOR_PIN = D17

dht11 = adafruit_dht.DHT11(SENSOR_PIN, use_pulseio=False)

i = 1

while (i <= 2):

    temperature = dht11.temperature

    humidity = dht11.humidity

    print(f"Humidity = {humidity:.02f}%")

    print(f"Temperature = {temperature:.02f}C")

    i += 1
```

Pre requisite module installation:

```
# sudo apt-get update

# sudo pip3 install adafruit_circuitpython_dht.
```

Working principle:

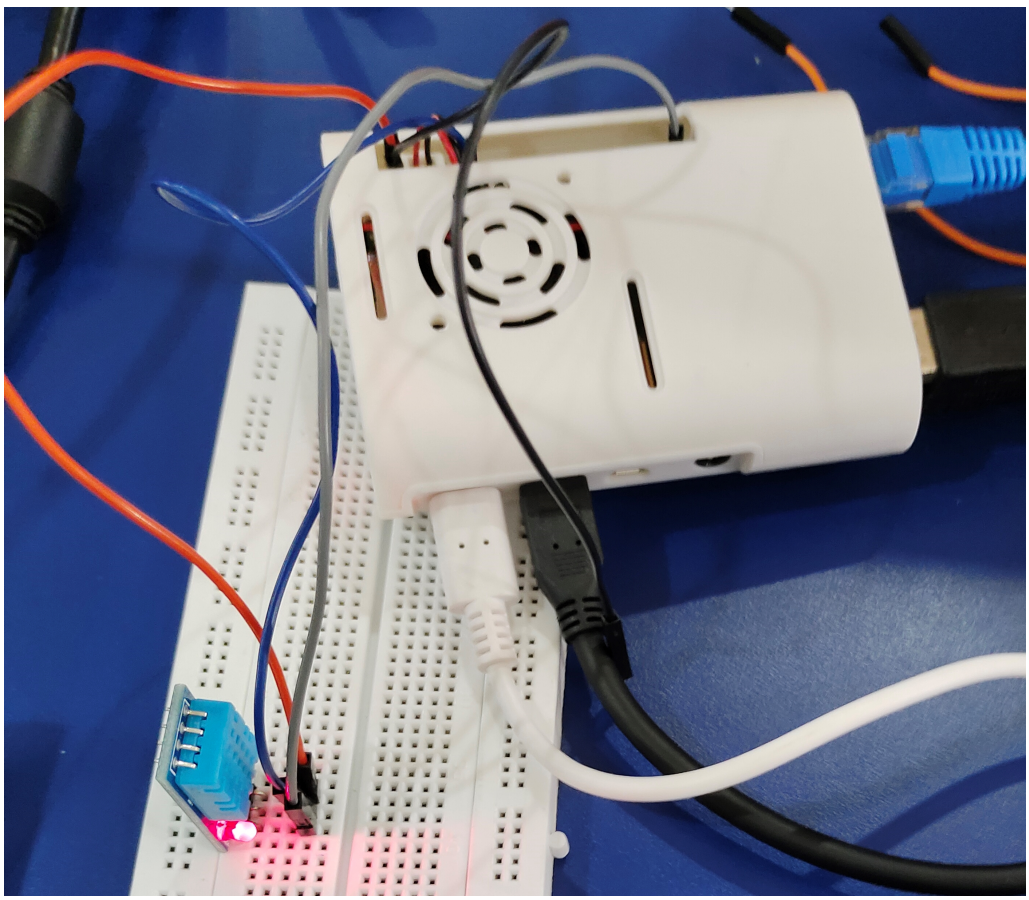
- When the circuit is connected according to the steps and as in Fig 3.2, the DHT11 sensor is connected to the GPIO17 pin.
- When the loop starts, the temperature and humidity values are fetched from the DHT11 sensor with the help of the adafruit_dht library.

2-LDR.py ✖ | ultra.py * ✖ | 3-DHT.py * ✖

```
1  print("This program is written by : [redacted] - 20881A1270, [redacted] - 20881A1277")
2  import time
3  import adafruit_dht
4  from board import *
5  SENSOR_PIN = D17
6  dht11=adafruit_dht.DHT11(SENSOR_PIN,use_pulseio=False)
7  i=1
8  while(i<=2):
9      temperature=dht11.temperature
10     humidity=dht11.humidity
11     print(f"Humidity = {humidity:.02f}")
12     print(f"Temperature = {temperature:.02f}C")
13     i+=1
14
15
16
17
18
19
20
```

Shell

```
>>> %Run 3-DHT.py

  This program is written by : [redacted] - 20881A1270, [redacted] - 20881A1277
  Humidity = 78.23
  Temperature = 25.89C
  Humidity = 79.01
  Temperature = 25.75C
```



4. Write program for monitoring temperature using DHT sensor and NodeMCU.

Date

Exp. No. 4

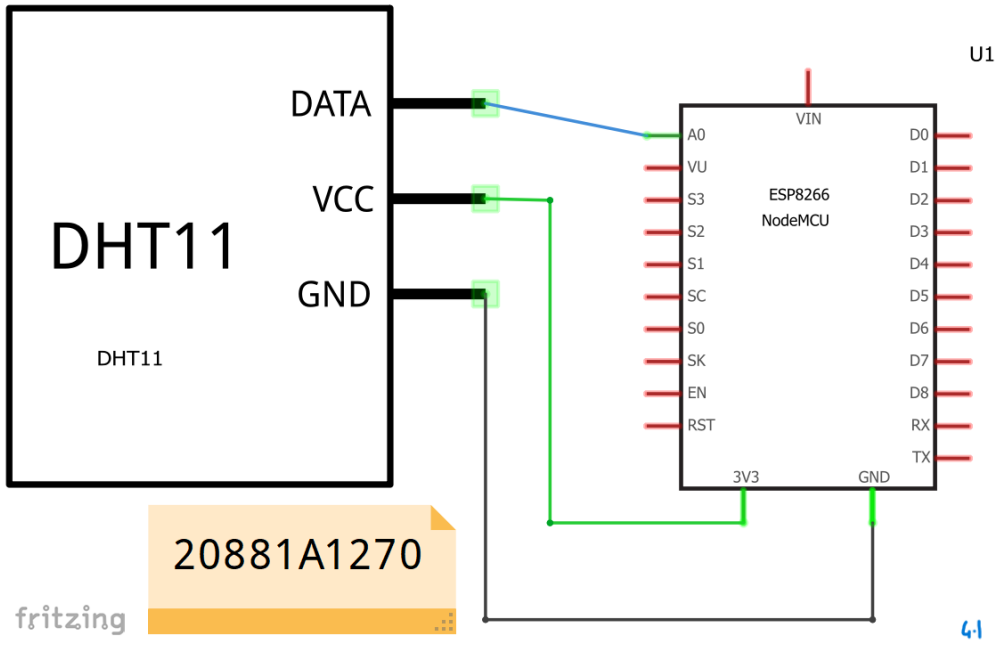
Aim: To write program for NodeMCU using Arduino IDE to monitor temperature using DHT11 Sensor.

Components:

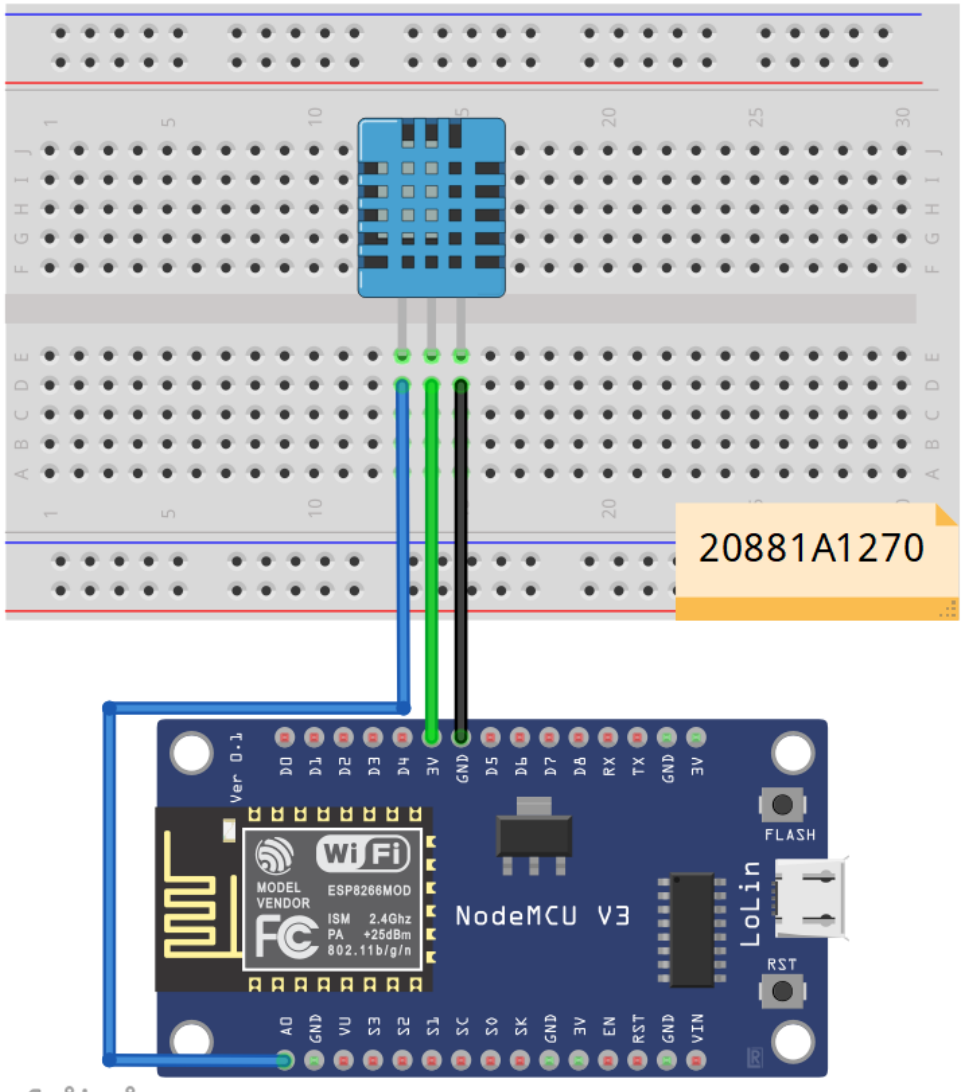
1. DHT11 Sensor ×1
2. Resistor (220Ω) ×1
3. Breadboard ×1
4. ESP8266 NodeMCU ×1
5. Jumper wires (M-M)
6. Misc: Computer (with Arduino IDE installed), USB power cable (USB Type-A to MicroUSB).

Procedure:

- Place the DHT Sensor on the breadboard such that 3 legs go into different rails.
- Connect the DATA pin to analog pin A0. as in fig. 4.1
- Connect the VCC pin to 3V3 as in fig. 4.1
- Connect the GND pin to GND as in Fig 4.1


U1
DATA
VCC
GND
DHT11
DHT11
VIN
A0
VU
S3
S2
S1
SC
S0
SK
EN
RST
ESP8266
NodeMCU
D0
D1
D2
D3
D4
D5
D6
D7
D8
RX
TX
3V3
GND
20881A1270
fritzing

4-1


20881A1270
Ver 0.1
FLASH
RST
NodeMCU V3
LoLin
MODEL ESP8266MOD
VENDOR
ISM 2.4Ghz
PA +25dBm
802.11b/g/n
fritzing

4-2

Source Code:

```
int sv=10;
int i=0;
float t;
double temp;
void setup()
{
    pinMode (A0, INPUT_PULLUP);
    Serial.begin(9600);
}
void loop()
{
    i++;
    if(i %10 == 0)
        Serial.println("This program is done by 20881A1270");
    sv = analogRead(A0); Serial.println("temp: ");
    Serial.println(sv);
    temp = (sv - 500)/10;
    Serial.println("Temperature in °: ");
    Serial.print(temp);
    Serial.println("°C");
    delay(1000);
}
```

Working Principle :

- According to the circuit in Fig 4.2, as the VCC of the DHT11 sensor is connected to 3V3, it will be set to power on as soon as the nodemcu is supplied with power.
- The analog value is read from the A0 pin and value is stored in the sv.
- the temperature is calculated using the input value and is printed onto the serial monitor.

NodeMCU_DHT11 | Arduino 1.8.19 (Windows Store 1.8.57.0)

File Edit Sketch Tools Help

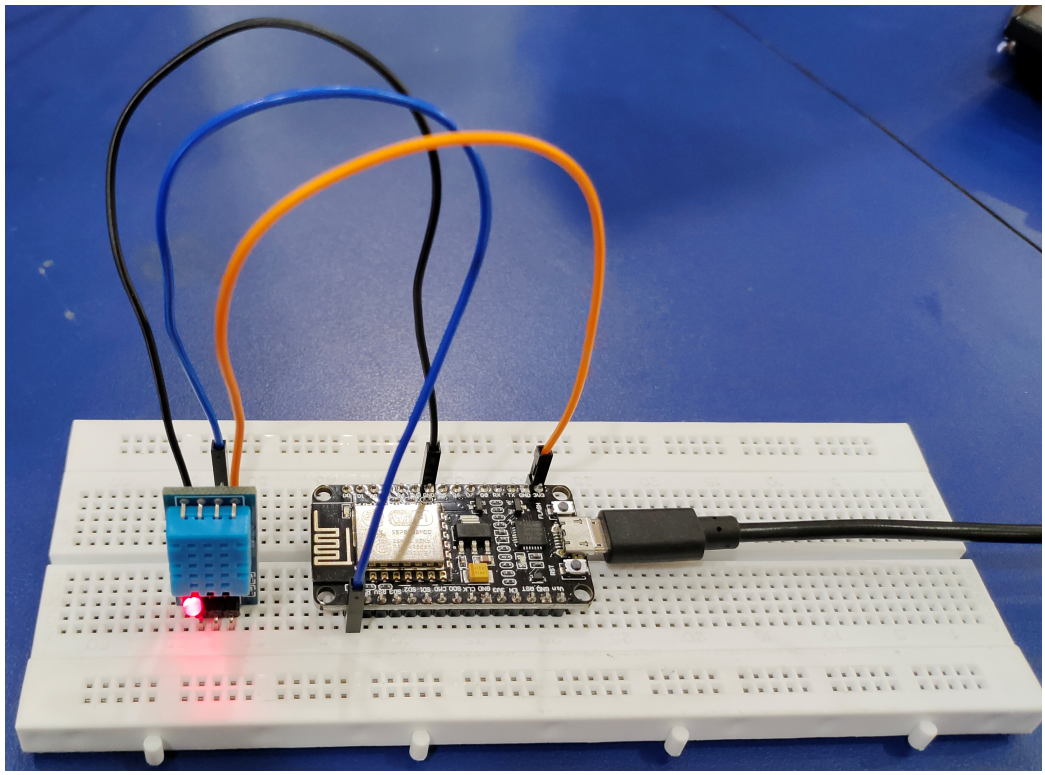
NodeMCU_DHT11

```
int sv=10;
int i=0;
float t;
double temp;
void setup()
{
pinMode(A0,INPUT_PULLUP);
  Serial.begin(9600);
}

void loop()
{
  i++;
  if(i % 10 == 0)
  Serial.println("This program is done by 20881A1270, 20881A1277.");
  sv=analogRead(A0);  Serial.println("temp: ");
  Serial.println(sv);

  temp=(sv-500)/10;
  Serial.println("Temparature in o:");
  Serial.print(temp);
  Serial.println("°C");
  delay(1000);
}
```

COM5

```
This program is done by 20881A1270, 20881A1277.
temp:
1024
Temparature in o:
52.00°C
temp:
1024
Temparature in o:
52.00°C
temp:
1024
Temparature in o:
52.00°C
temp:
1024
Temparature in o:
52.00°C
temp:
1024
Temparature in o:
52.00°C
```

Autoscroll Show timestamp



5. Implement Raspberry Pi based Automated Street Lighting system.

Date

Exp. No. 5

Aim: To write program for automated street lighting system using Raspberry Pi

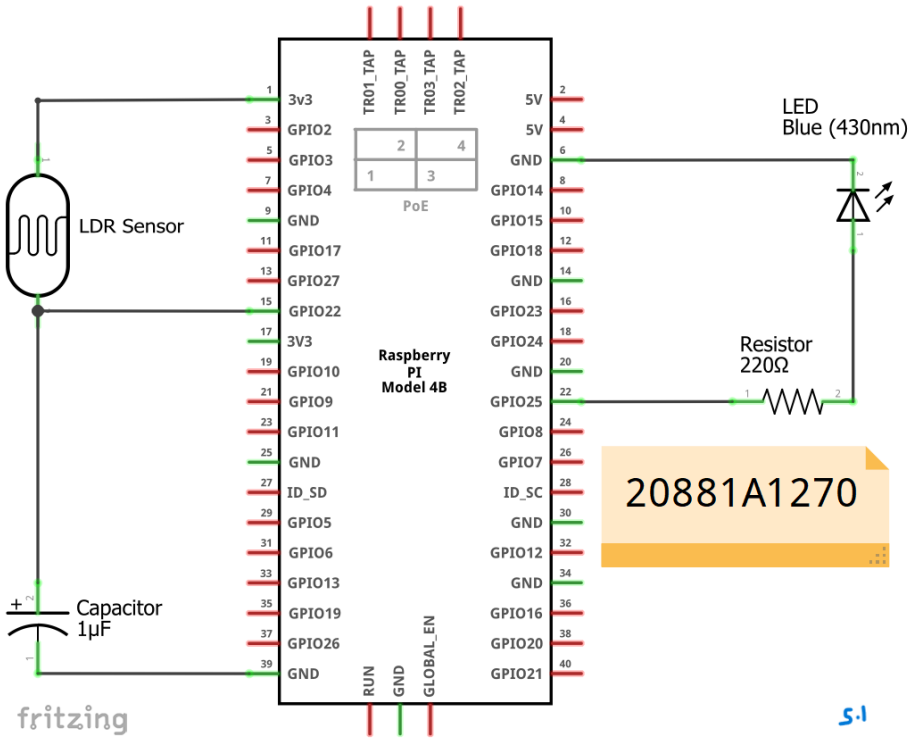
Components:

1. LDR Sensor ×1
2. LED ×1
3. Capacitor ×1
4. Resistor (220 Ω) ×1
5. Bread board ×1
6. Raspberry Pi 4 Model B ×1
7. Jumper Wires (M-F) ×5
8. Misc: micro SD Card (with preloaded Raspbian OS), Ethernet, Power Cable (Type C), Monitor, Keyboard, Mouse.

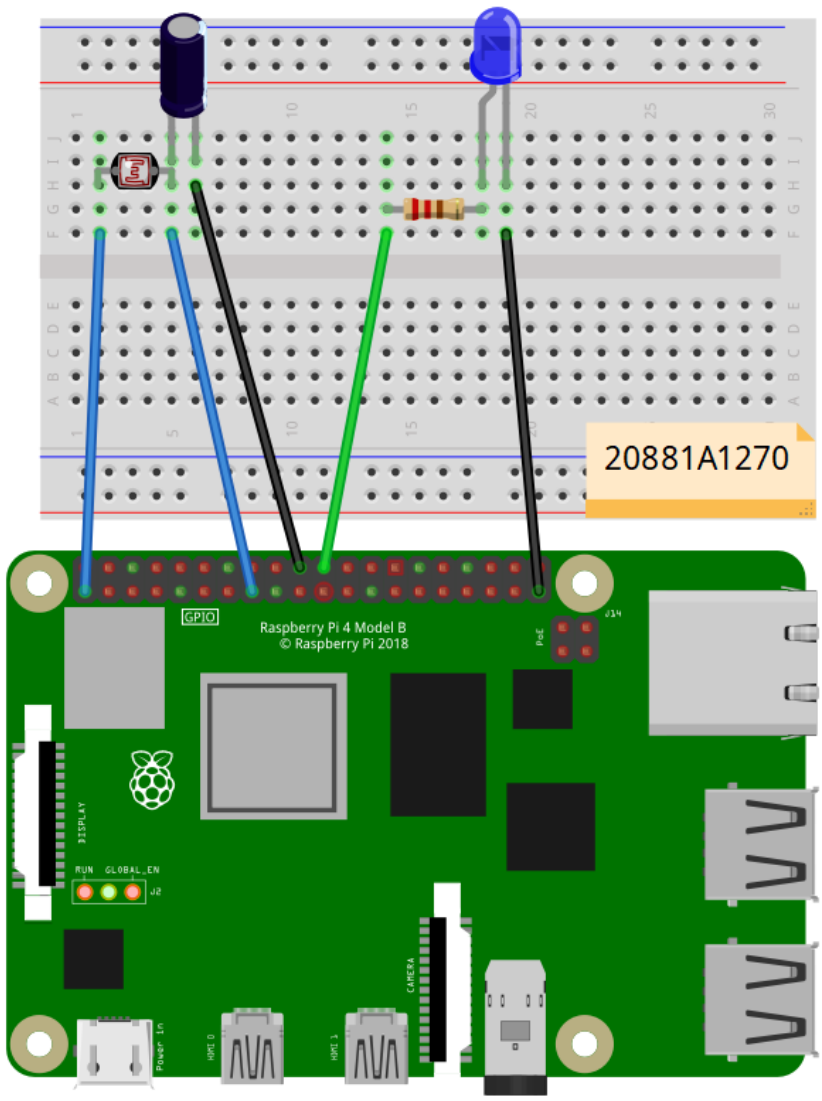
Procedure:

- Place the LDR Sensor on the breadboard such that both legs go into different rails. Connect one end of the LDR to +3.3 V and place a capacitor in series with the other leg as shown in Fig 5.2.
- Connect the rail containing LDR and capacitor to D15. pin and the other end of capacitor to GND. as in the Fig. 5.1

Mod1


TR01_TAP
TR00_TAP
TR03_TAP
TR02_TAP
3v3
GPIO2
GPIO3
GPIO4
GND
GPIO17
GPIO27
GPIO22
3V3
GPIO10
GPIO9
GPIO11
GND
ID_SD
GPIO5
GPIO6
GPIO13
GPIO19
GPIO26
GND
5V
5V
GND
GPIO14
GPIO15
GPIO18
GND
GPIO23
GPIO24
GND
GPIO25
GPIO8
GPIO7
ID_SC
GND
GPIO12
GND
GPIO16
GPIO20
GPIO21
RUN
GND
GLOBAL_EN
PoE
Raspberry PI Model 4B
LDR Sensor
Capacitor 1µF
LED Blue (430nm)
Resistor 220Ω
20881A1270
fritzing

S.1


20881A1270
GPIO
Raspberry Pi 4 Model B
© Raspberry Pi 2018
PoE
DISPLAY
RUN
GLOBAL_EN
J2
CAMERA
Power In
HDMI 0
HDMI 1
fritzing

S.2

- Place an LED on the breadboard such that both its legs are in different rails which are free as Fig 5.1
- Place a resistor connecting the anode side of the LED and connect other end of the resistor to GPIO25 as shown in Fig. 5.1.
- Finally connect other end of LED to GND.

Source Code:

```
print("This program is written by : 20881A1270").
import RPi.GPIO as GPIO
import time
led = 22
lda = 15
GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)
GPIO.setup(led, GPIO.OUT)
GPIO.output(led, GPIO.LOW)
while True:
    GPIO.setup(lda, GPIO.OUT)
    GPIO.output(lda, GPIO.LOW)
    time.sleep(1)
    GPIO.setup(lda, GPIO.IN)
    count = 0
    while (GPIO.input(lda) == GPIO.LOW):
        count += 1
    print(count)
```

```
if ( count >= 1)
        GPIO.output (led, GPIO.HIGH)
        print ("Dark")
    else:
        GPIO.output( led, GPIO.LOW)
        print ("Day")
    time.sleep(1)
```

Working Principle:

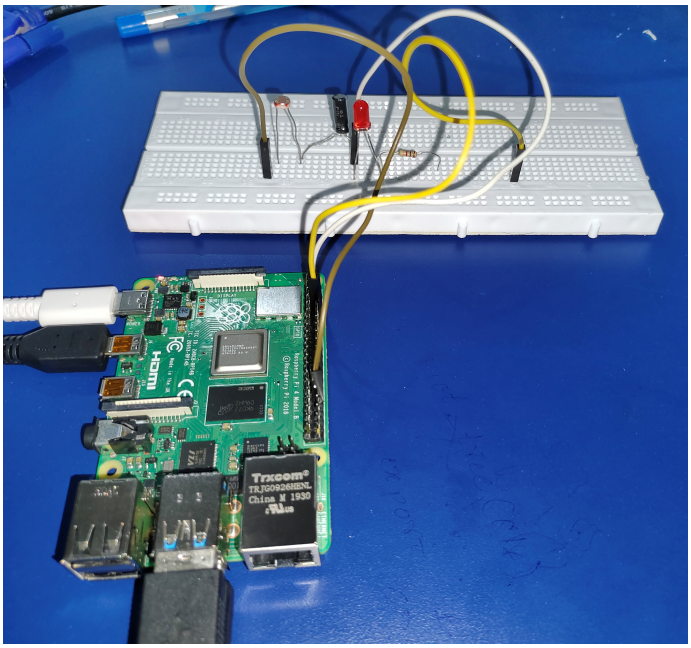
- As the LDR sensor is connected to 3v3, when the loop starts, the GPIO pin connected to the LDR is set as input and while loop is used to calculate the amount of light from the LDR sensor.
- The count is calculated and sets the LED to High if the count is greater than or equal to 1. i.e, surrounding light is low. and vice versa.

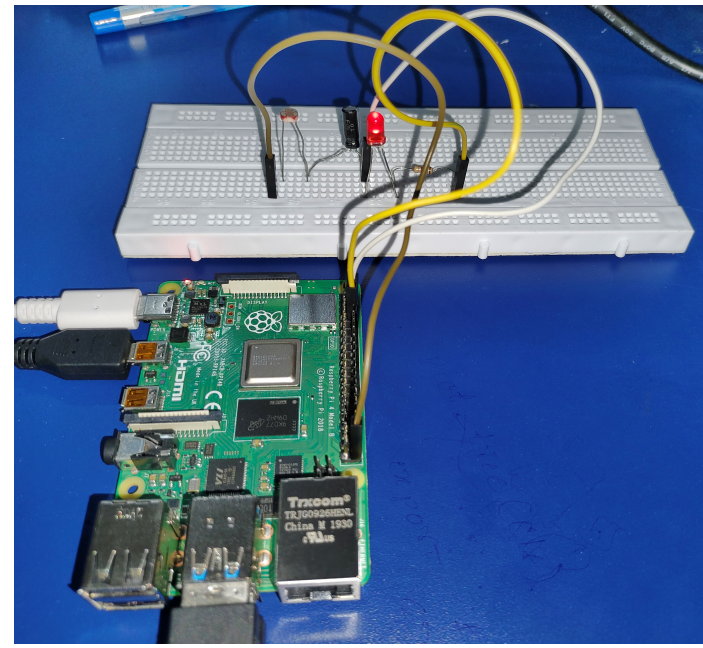
2-LDR.py ultra.py

```python
print("This program is written by : [redacted] - 20881A1270, [redacted] - 20881A1277")
import RPi.GPIO as GPIO
import time
led=22
ldr=15
GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)
GPIO.setup(led,GPIO.OUT)
GPIO.output(led,GPIO.LOW)
while True:
    GPIO.setup(ldr,GPIO.OUT)
    GPIO.output(ldr,GPIO.LOW)
    time.sleep(1)
    GPIO.setup(ldr,GPIO.IN)
    count=0
    while(GPIO.input(ldr)==GPIO.LOW):
        count+=1
    print(count)
    if(count>=1):
        GPIO.output(led,GPIO.HIGH)
        print("Dark")
    else:
        GPIO.output(led,GPIO.LOW)
        print("Day")
    time.sleep(1)
```

Shell

```
This program is written by : [redacted] - 20881A1270, [redacted] - 20881A1277
446
Dark
659
Dark
0
Day
3885
Dark
```





6. Implement NodeMCU based Automated street Lighting system.

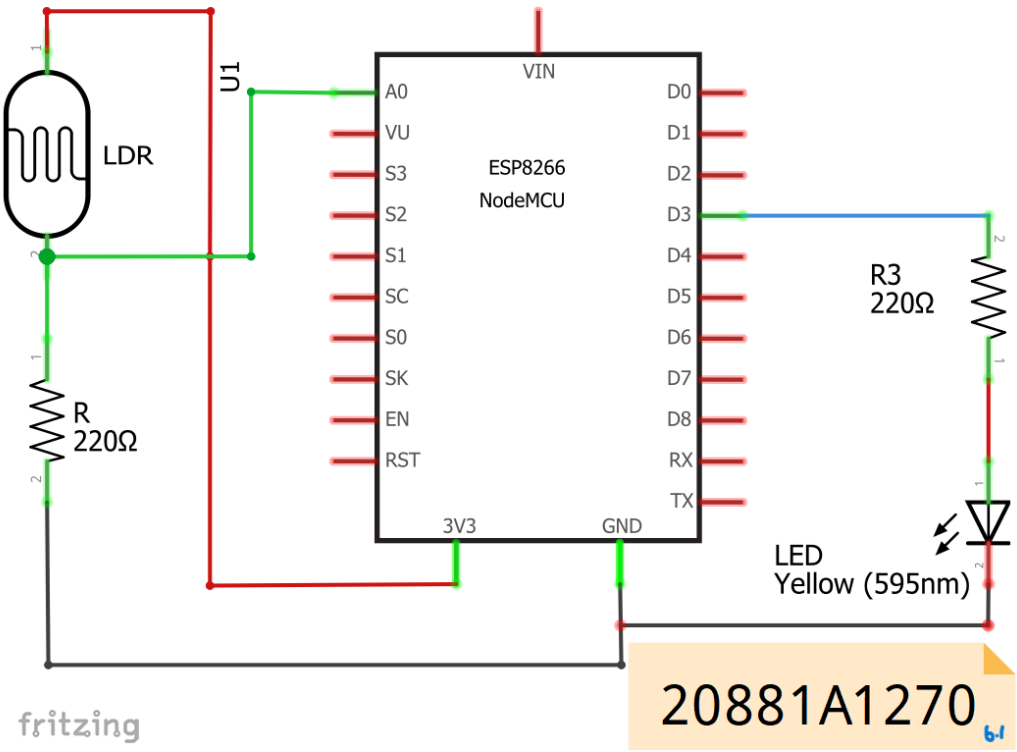
Aim: To write program using Arduino IDE for NodeMCU to implement automated street lighting system.

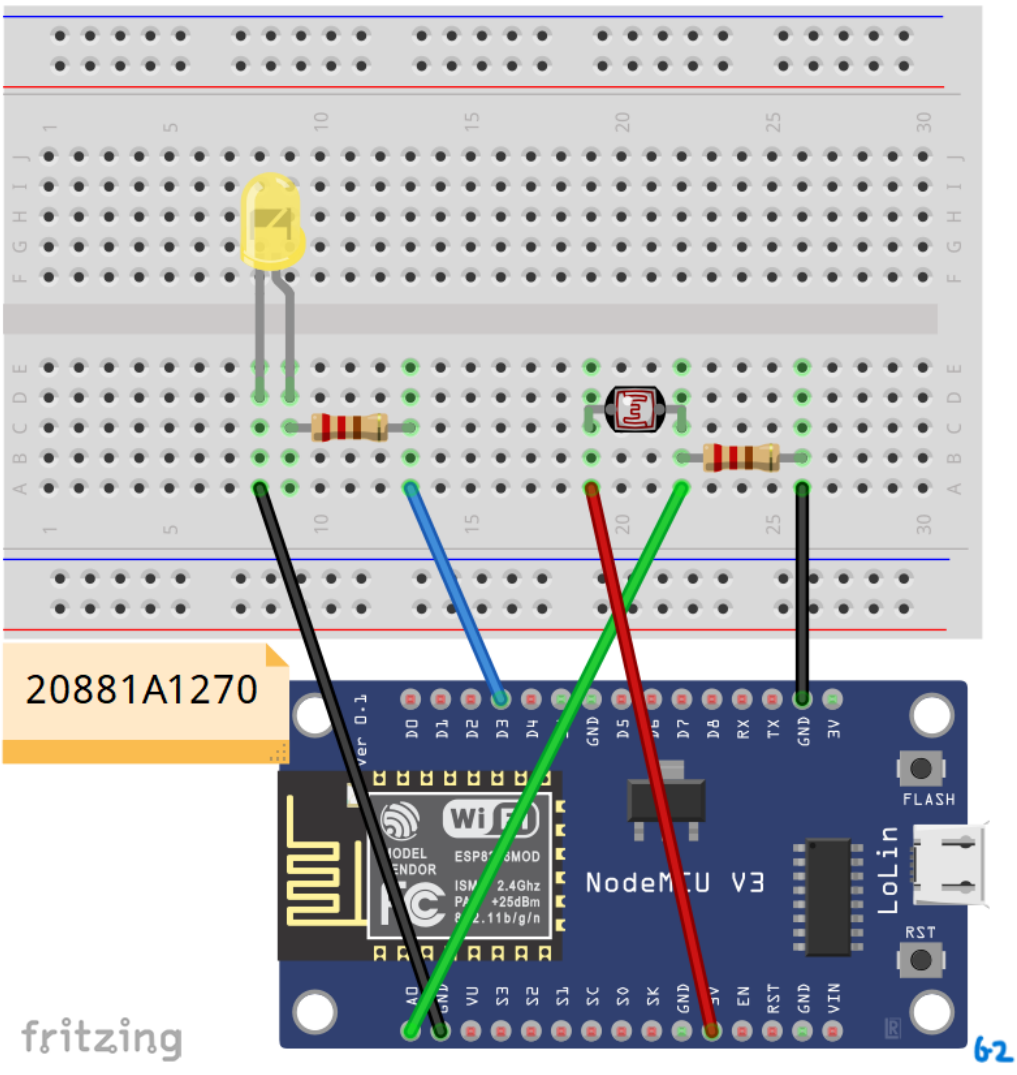
Components:

1. LDR Sensor.
2. LED
3. Resistor (220 Ω)
4. Breadboard
5. ESP8266 NodeMCU
6. Jumper Wires (M-F)
7. Misc: Computer (Installed with Arduino IDE), USB Connector (USB Type-A to micro USB).

Procedure:

- Place the LDR sensor on the breadboard such that both legs go into different rails. Connect one end of the LDR and one end of the resistor in a rail as shown in Fig 6.2.
- Conect the other end of the LDR sensor to 3V, connect the common rail for LDR and resistor to analog pin A0, connect the other end of the resistor to GND as in Fig. 6.2.
- Place the LED on the breadboard such that both legs go into different rails.


LDR
U1
VIN
A0
VU
S3
S2
S1
SC
S0
SK
EN
RST
ESP8266
NodeMCU
D0
D1
D2
D3
D4
D5
D6
D7
D8
RX
TX
3V3
GND
R
220Ω
R3
220Ω
LED
Yellow (595nm)
fritzing
20881A1270
6-1


20881A1270
NodeMCU V3
FLASH
RST
LoLin
fritzing
6-2

- Place the resistor on the breadboard such that one end is connected to the anode side of the LED. as shown in Fig 6.2.
- Connect the cathode side of LED to GND and other side of resistor to digital pin D3 as shown in Fig 6.1

Source code:

```
int ldr = 0;
void setup() {
    pinMode (D3, OUTPUT);
    pinMode (A0, INPUT);
    Serial.begin (9600);
}
void loop() {
    Serial.print ("This program is written by : 20881A1270");
    ldr = analogRead (A0);
    Serial.println ("Analog Reading :"); Serial.println (ldr);
    if ( ldr <= 50) {
        digitalWrite ( D3, HIGH);
        Serial.println ("LED ON");
        delay (1000);
    }
    else {
        digitalWrite (D3, LOW);
        Serial.println ("LED OFF");
        delay (1000);
    }
}
```

Working Principle:

- If the code is uploaded correctly and power is given to the NodeMCU, as the LDR is connected to 3V, the power is supplied right away.
- As the other end of LDR is connected to analog input pin A0, the light intensity is fetched from the LDR sensor.
- After that according to the conditions, the LED will be set to high for 1 second before another check or will be set to low if the light intensity value from LDR is greater than 50.

LDR_LED_BLINK | Arduino 1.8.20 Hourly Build 2021/12/20 07:33

File Edit Sketch Tools Help

Upload

LDR_LED_BLINK

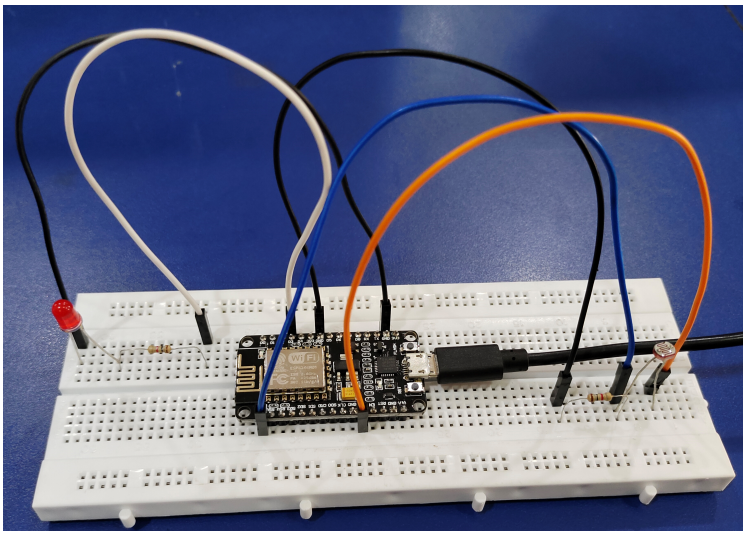
```
int ldr=0;
void setup() {
  pinMode(D3,OUTPUT);
  pinMode(A0,INPUT);
  Serial.begin(9600);
}
void loop() {
  Serial.print("This program is written by: 20881A1262, 20881A1270, 20881A1277");
  ldr=analogRead(A0);
  Serial.println("Analog Reading");
  Serial.println(ldr);
  if(ldr<=50)
  {
    digitalWrite(D3,HIGH);
    Serial.println("LED ON");
    delay(1000);
  }
  else
  {
    digitalWrite(D3,LOW);
    Serial.println("LED OFF");
    delay(1000);
  }
}
```

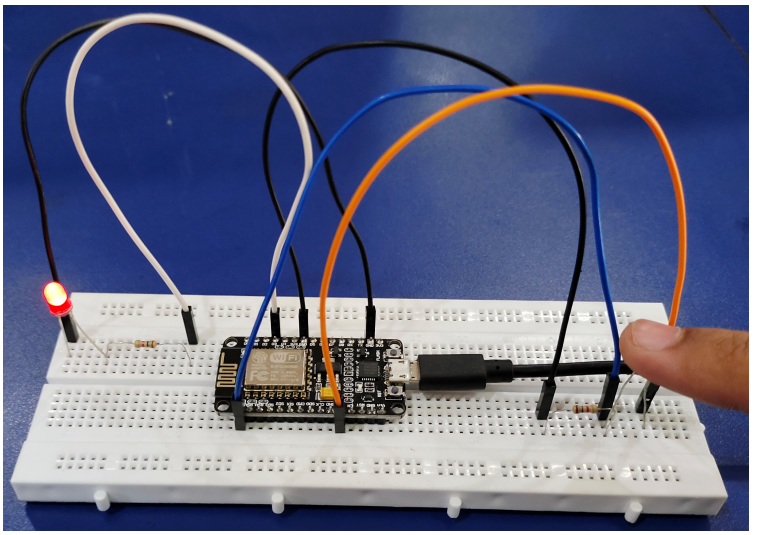
COM3

Send

```
This program is written by: 20881A1262, 20881A1270, 20881A1277Analog Reading
90
LED OFF
This program is written by: 20881A1262, 20881A1270, 20881A1277Analog Reading
89
LED OFF
This program is written by: 20881A1262, 20881A1270, 20881A1277Analog Reading
89
LED OFF
This program is written by: 20881A1262, 20881A1270, 20881A1277Analog Reading
42
LED ON
This program is written by: 20881A1262, 20881A1270, 20881A1277Analog Reading
31
LED ON
This program is written by: 20881A1262, 20881A1270, 20881A1277Analog Reading
```

Autoscroll Show timestamp No line ending 9600 baud Clear output





7. Write an Arduino Program for Distance Measurement using ultrasonic sensor.

Date

Exp. No. 7

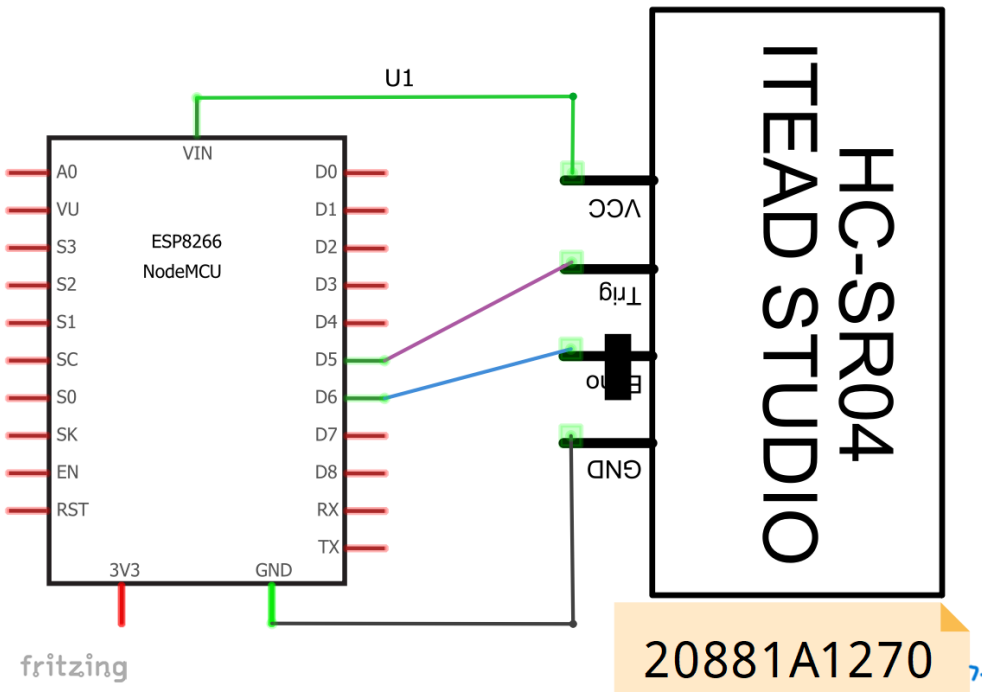
Aim: To write a program in Arduino IDE for NodeMCU to measure distance of an object using ultrasonic sensor.

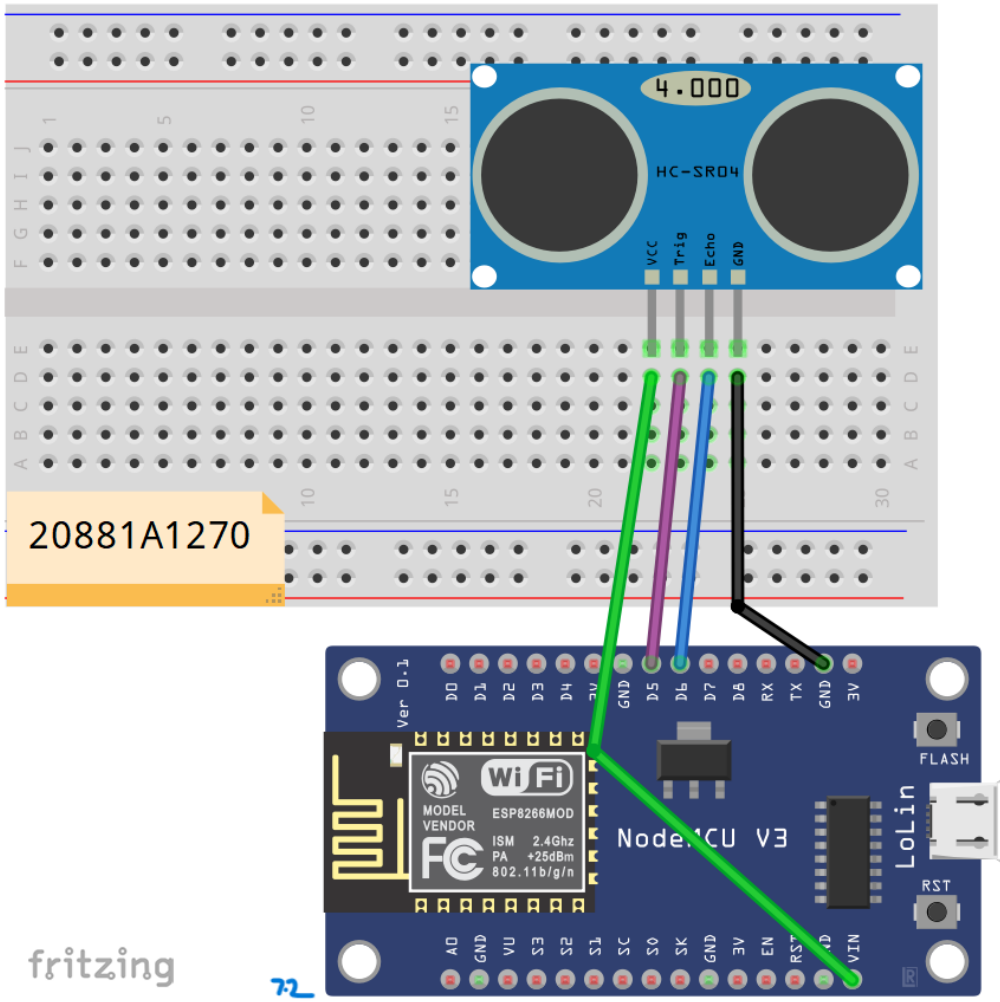
Components:

1. Ultrasonic sensor
2. ESP8266 NodeMCU
3. Breadboard
4. Connecting wires (M-M)
5. Misc: Computer (Installed with Arduino IDE), USB Power Cable (USB Type-A to MicroUSB).

Procedure:

- Place the ultrasonic sensor in the breadboard such that each leg goes into different rails.
- As shown in the figure 7.1, Connect,
  - → VCC Pin to VIN on NodeMCU.
  - → Trig Pin to Digital pin D5.
  - → Echo Pin to Digital pin D6.
  - → GND Pin to GND on NodeMCU


U1
VIN
A0
VU
S3
S2
S1
SC
S0
SK
EN
RST
3V3
GND
D0
D1
D2
D3
D4
D5
D6
D7
D8
RX
TX
ESP8266
NodeMCU
VCC
Trig
Echo
GND
HC-SR04
ITEAD STUDIO
20881A1270
fritzing


4.000
HC-SR04
VCC
Trig
Echo
GND
20881A1270
Ver 0.1
FLASH
RST
LoLin
MODEL
VENDOR
ESP8266MOD
ISM 2.4Ghz
PA +25dBm
802.11b/g/n
fritzing

Source Code:

```
int trigPin = 12, echoPin = 14, i=0;
long duration;
float distanceCm, distanceInch;
void setup() {
    Serial.begin (9600);
    pinMode (trigPin, OUTPUT);
    pinMode ( echoPin, INPUT);
}
void loop()
{
    i++;
    if (i%10==0)   Serial.println("This program is done by 20881A1270");
    digitalWrite (trigPin, LOW);
    delayMicroseconds (2);
    digitalWrite (trigPin, HIGH);
    delayMicroseconds (10);
    digitalWrite (trigPin, LOW);

    duration = pulseIn (echoPin, HIGH);
    distanceCm = duration * 0.034/2;
    distanceInch = distanceCm * 0.393701;

    Serial.print (" Distance (cm) : "); Serial.println ( distanceCm);
    Serial.print (" Distance (inch) : "); Serial.println ( distanceInch);

    delay (1000);
}
```

Working principle:

- If the circuit is connected as shown in the figure 7.2, As soon as the power is given to the NodeMCU, the NodeMCU Ultrasonic sensor will be powered from the VIN.
- When the trig pin is set to high from the digital pin D5, it will transmit a high frequency signal to the surrounding.
- The trig pin is then set to low after 10 ms and the echo pin will receive the echo of the high frequency signal.
- The value is fetched from the echoPin into duration.
- Then the distances are calculated based on the time duration of the travel of high frequency wave.

NodeMCU_Ultrasonic | Arduino 1.8.19 (Windows Store 1.8.57.0)

File Edit Sketch Tools Help

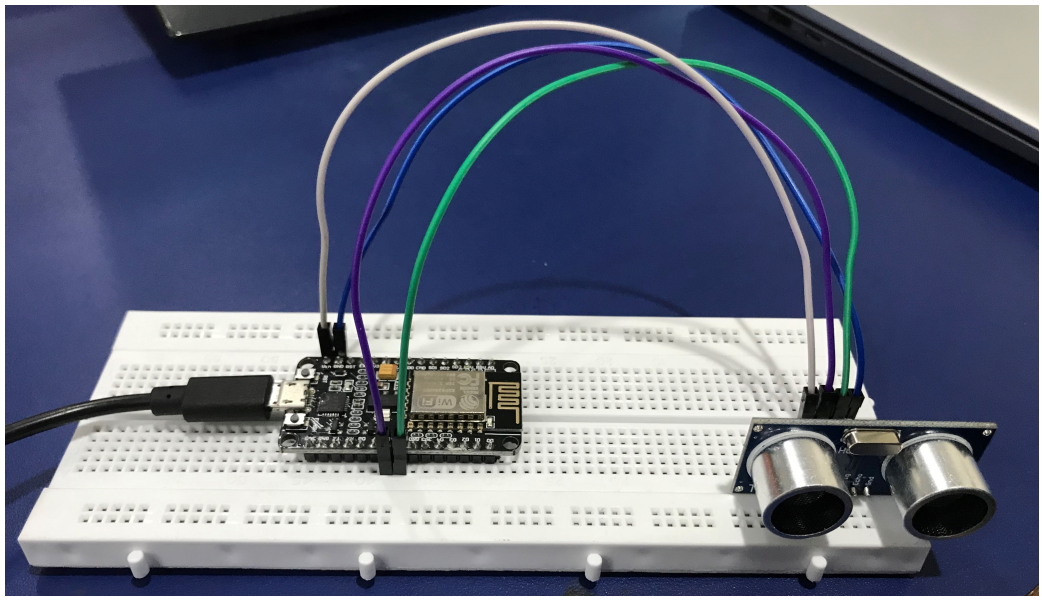
NodeMCU_Ultrasonic

```
int trigPin = 12;
int echoPin = 14;
int i=0;

long duration;
float distanceCm;
float distanceInch;

void setup()
{
  Serial.begin(9600);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
}

void loop()
{
  i++;
  if(i % 10 == 0)
    Serial.println("This program is done by 20881A1270, 20881A1277.");
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  duration = pulseIn(echoPin, HIGH);

  distanceCm = duration * 0.034/2;

  distanceInch = distanceCm * 0.393701;

  Serial.print("Distance (cm): ");
  Serial.println(distanceCm);
  Serial.print("Distance (inch): ");
  Serial.println(distanceInch);

  delay(1000);
}
```

COM5

```
This program is done by 20881A1270, 20881A1277.
Distance (cm): 198.30
Distance (inch): 78.07
Distance (cm): 197.86
Distance (inch): 77.90
Distance (cm): 198.73
Distance (inch): 78.24
Distance (cm): 9.47
Distance (inch): 3.73
Distance (cm): 22.92
Distance (inch): 9.02
Distance (cm): 35.02
Distance (inch): 13.79
Distance (cm): 203.59
Distance (inch): 80.15
Distance (cm): 5.61
Distance (inch): 2.21
Distance (cm): 2.91
Distance (inch): 1.14
Distance (cm): 12.39
Distance (inch): 4.88
```

Autoscroll Show timestamp



Page No. 22	8. Implement Raspberry Pi program for Distance Measurement using Ultrasonic sensor	Date Exp. No. 8

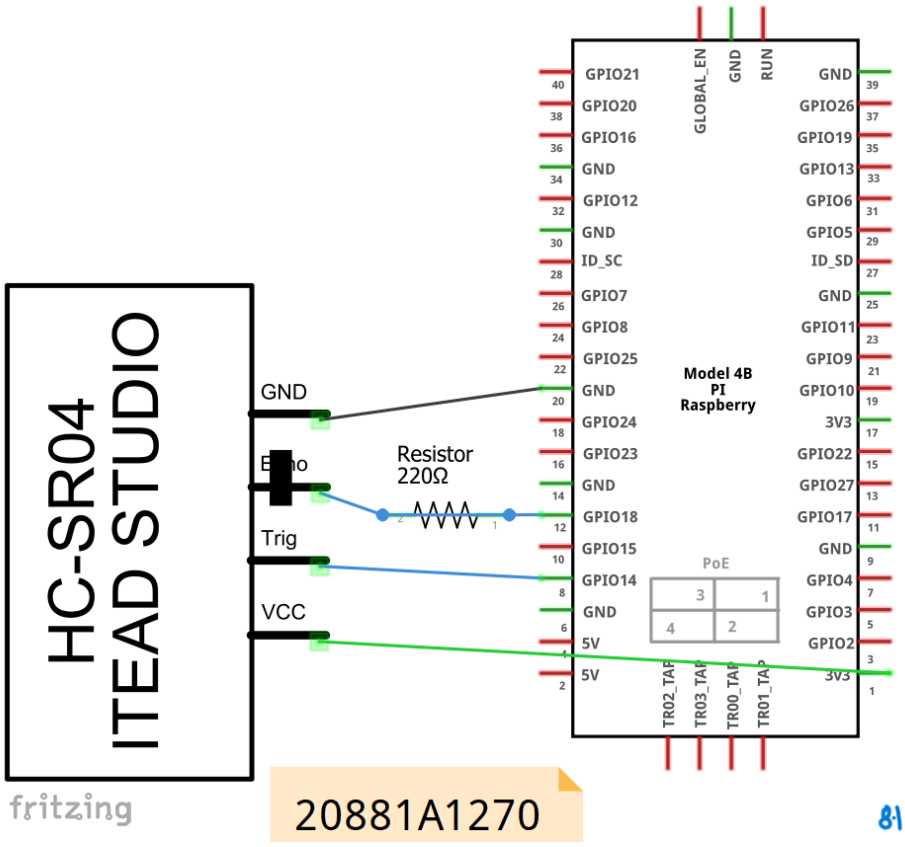
Aim: To write a program for Distance measurement with ultrasonic sensor using Raspberry Pi.

Components:

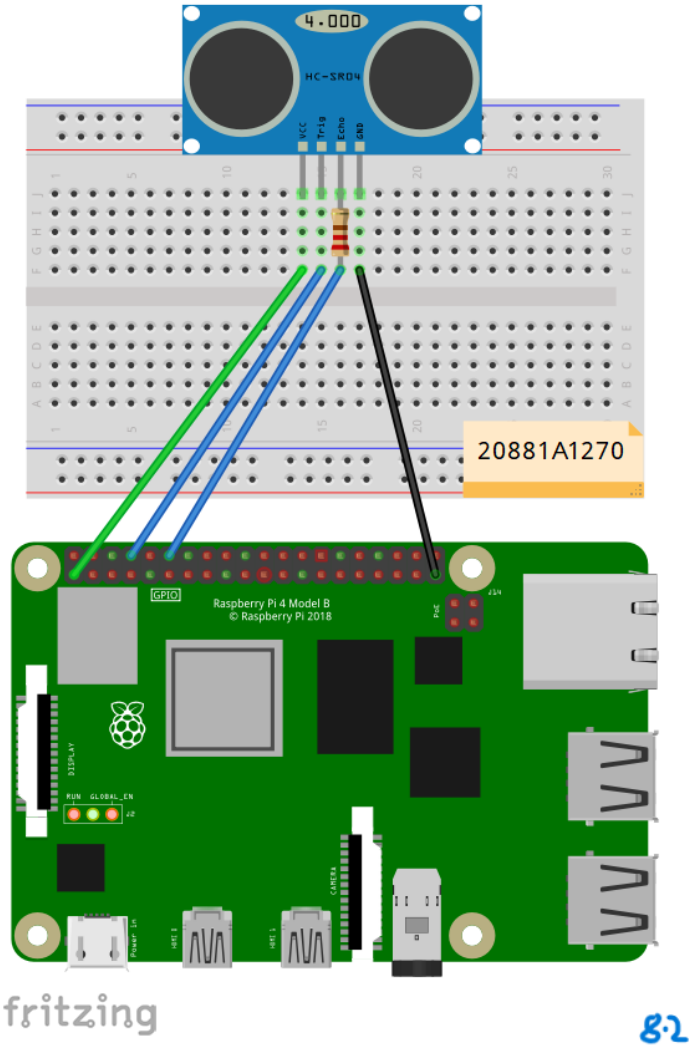
1. Ultrasonic sensor (HC-SR04) ×1
2. Resistor (220 Ω) ×1
3. Breadboard ×1
4. Raspberry Pi 4 Model B ×1
5. Jumper wires ×4
6. Misc: microSD Card (preloaded with Raspbian OS), power cable (TypeC) Monitor, Keyboard, Mouse, microHDMI to VGA converter.

Procedure:

- Place the Ultrasonic sensor on the breadboard such that all 4 legs on different rails and a resistor connected to the Echo pin rail as shown in the Fig 8.2
- Connect Vcc to +5V of Raspberry Pi, Trig to GPIO pin(8).
- Connect other end of resistor to GPIO pin (12) and GND to GND as shown in Fig 8.1.


HC-SR04
ITEAD STUDIO
GND
Echo
Trig
VCC
Resistor
220Ω
Model 4B
PI
Raspberry
GPIO21
GPIO20
GPIO16
GND
GPIO12
GND
ID_SC
GPIO7
GPIO8
GPIO25
GND
GPIO24
GPIO23
GND
GPIO18
GPIO15
GPIO14
GND
5V
5V
GLOBAL_EN
GND
RUN
GND
GPIO26
GPIO19
GPIO13
GPIO6
GPIO5
ID_SD
GND
GPIO11
GPIO9
GPIO10
3V3
GPIO22
GPIO27
GPIO17
GND
GPIO4
GPIO3
GPIO2
3V3
PoE
fritzing
20881A1270

8.1


4.000
HC-SR04
VCC
Trig
Echo
GND
20881A1270
GPIO
Raspberry Pi 4 Model B
© Raspberry Pi 2018
fritzing

8.2

Source Code:

```
import RPi.GPIO as US
import time
import signal
import sys
US.setmode (US.BOARD)
trig = 8
echo = 12
signal.signal (signal.SIGINT, False)
US.setup (trig, US.OUT)
US.setup (echo, US.IN)
while True:
    US.output (trig, 1)
    time.sleep (0.01)
    US.output (trig, 0)
    start = time.time()
    stop = time.time()
    while (US.input(echo) == 0):
        start = time.time()
    while (US.input(echo) == 1):
        stop = time.time()
    diff = stop - start
    dist = diff * 34300/2
    print ("Distance = %.2f" % (dist))
    time.sleep(1)
```

Working Principle:

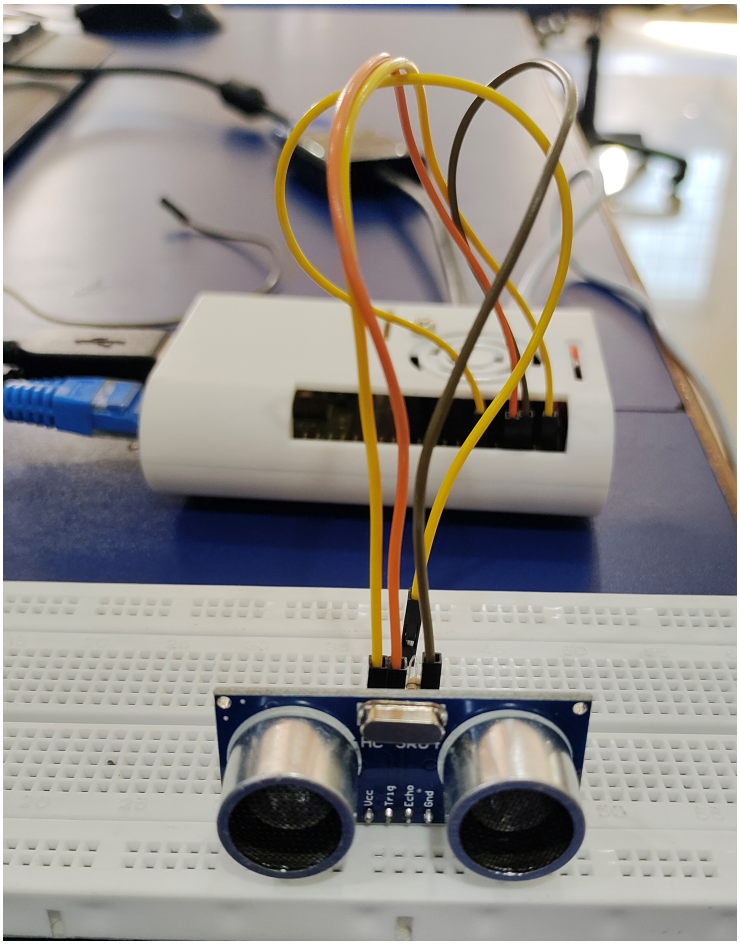
- If connected as in the Fig. 8.2, the ultrasonic sensor will be powered by +5V from the Raspberry Pi
- With the trig pin set to high, the sensor transmits high frequency signal to the surrounding.
- After transmitting the signal, trig pin is set to low and start will be initiated using time().
- When the echo pin gives high value as input, we stop the recording of time.
- Difference is calculated between start time and stop time and then distance is calculated.

Ultrasonic.py *

```python
# Distance Measurement using Ultrasonic sensor - 20881A1277,20881A1270

import RPi.GPIO as US
import time
import signal
import sys
US.setmode(US.BOARD)
trig = 8
echo = 12
signal.signal(signal.SIGINT,False)
US.setup(trig, US.OUT)
US.setup(echo, US.IN)

while True:
    US.output(trig, 1)
    time.sleep(0.01)
    US.output(trig, 0)
    start = time.time()
    stop = time.time()
    while (US.input(echo) == 0):
        start = time.time()
    while (US.input(echo) == 1):
        stop = time.time()
    diff = stop - start
    dist = diff*34300/2
    print("Distance = %.2f" %(dist))
    time.sleep(1)
```

Shell

```
Distance = 7.00
Distance = 22.63
Distance = 23.06
Distance = 21.71
Distance = 22.23
```



9. Study & Install IDE of Arduino and different types of Arduino.

Arduino: Arduino is an open source electronics platform based on easy-to-use hardware and software.

Arduino creates programmable circuit boards also called as microcontrollers which can read inputs from different sensors and give outputs to different actuators to make the work to complete a specific task.

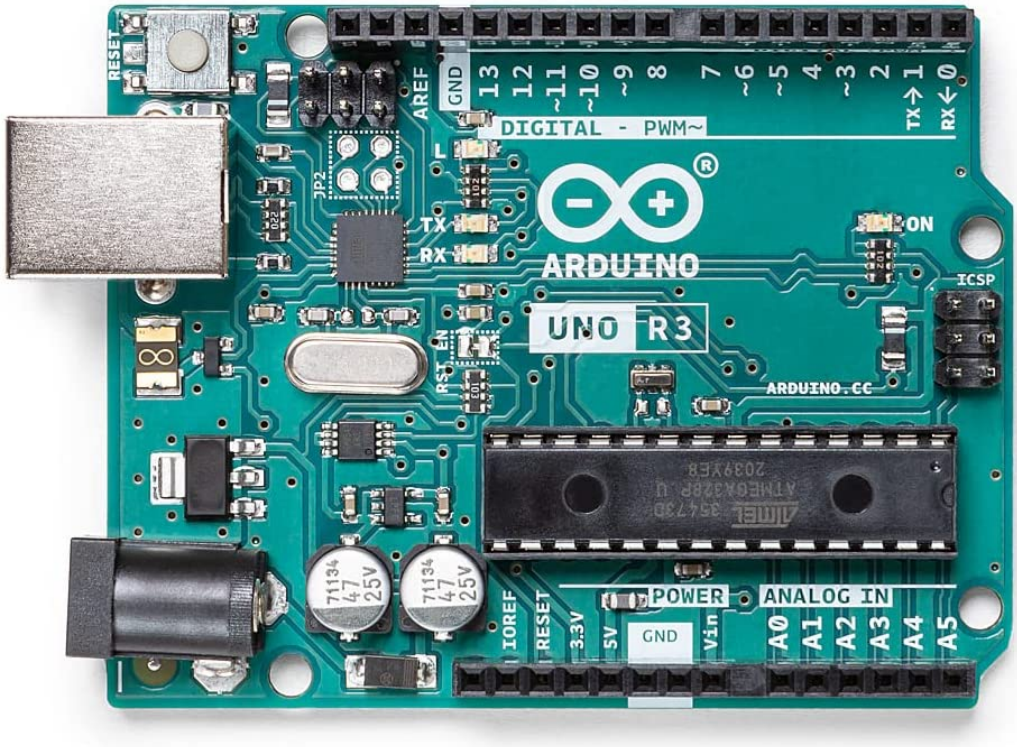
Arduino boards are of different types based on their usecase.

Types of Arduino boards include:

1. **Arduino Uno:**
   - → Used for simple IoT based applications, robot controlling etc.
   - → Has 14 Digital Input/Output pins, 6 Analog Input pins.

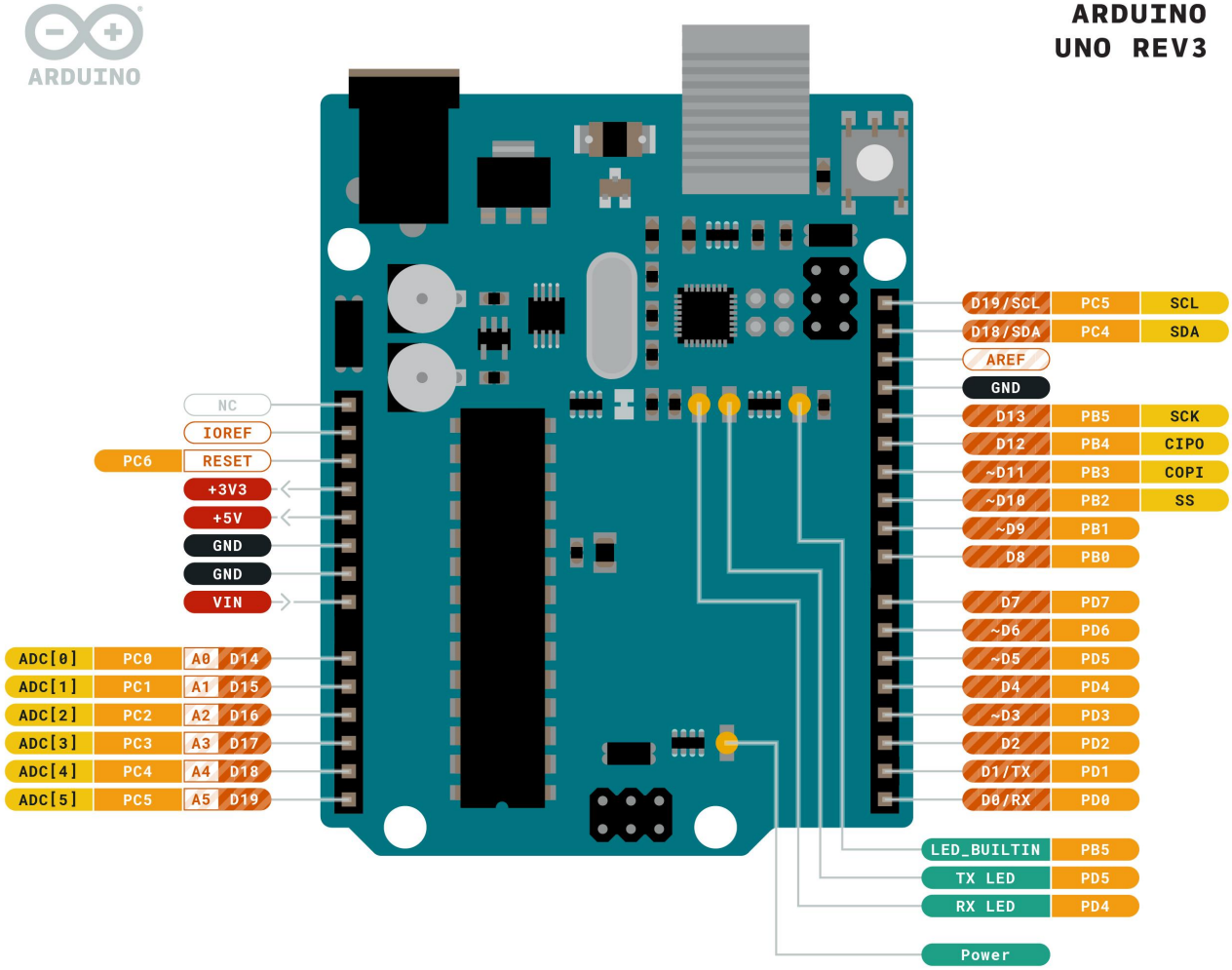
2. **Arduino Nano:**
   - → Used for small and portable electronics.
   - → Has 14 Digital Input/Output pins, 6 Analog Input pins.

3. **Arduino Mega:**
   - → Used for high I/O requirements with more memory space.
   - → Has 54 Digital Input/Output pins, 16 Analog Input pins.



ARDUINO

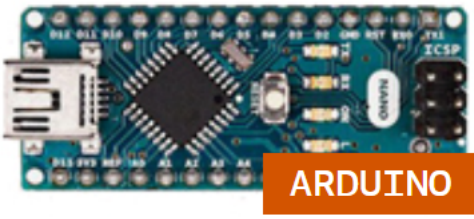
ARDUINO UNO REV3

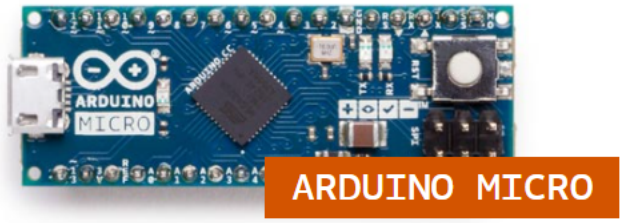


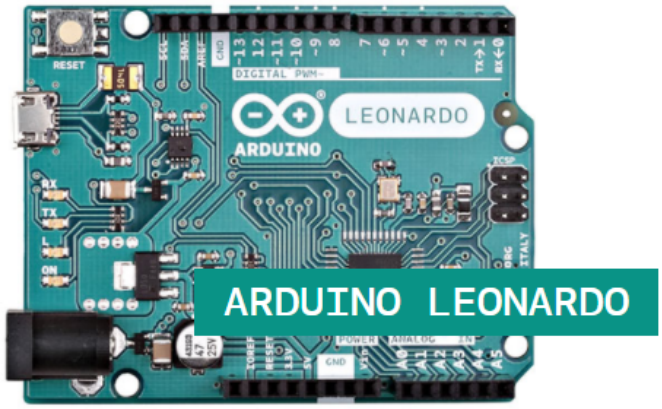
Ground
Power
LED
Internal Pin
SWD Pin
Other Pin
Digital Pin
Analog Pin
Default
Microcontroller's Port

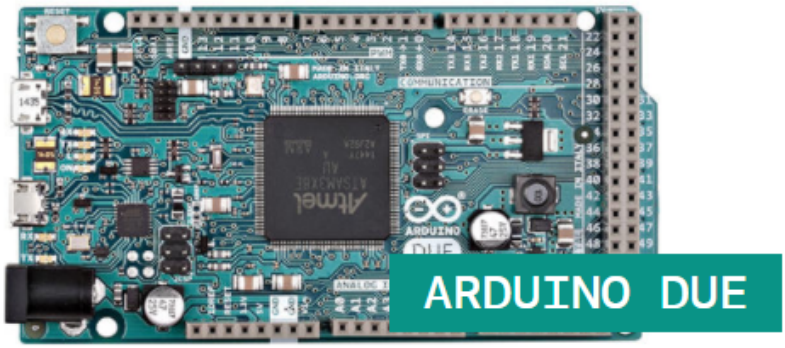
ARDUINO.CC

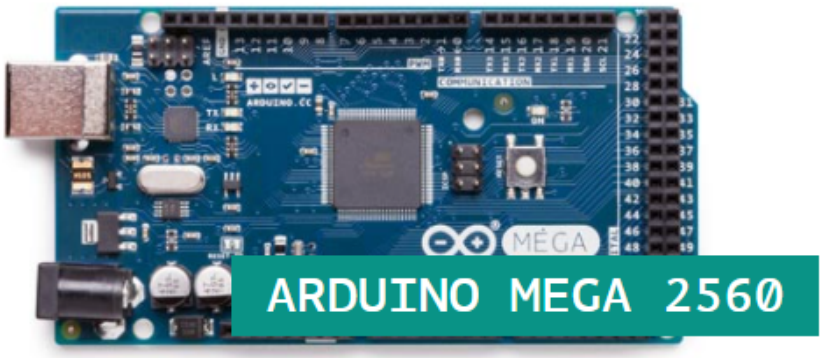
Types of Arduino Boards:











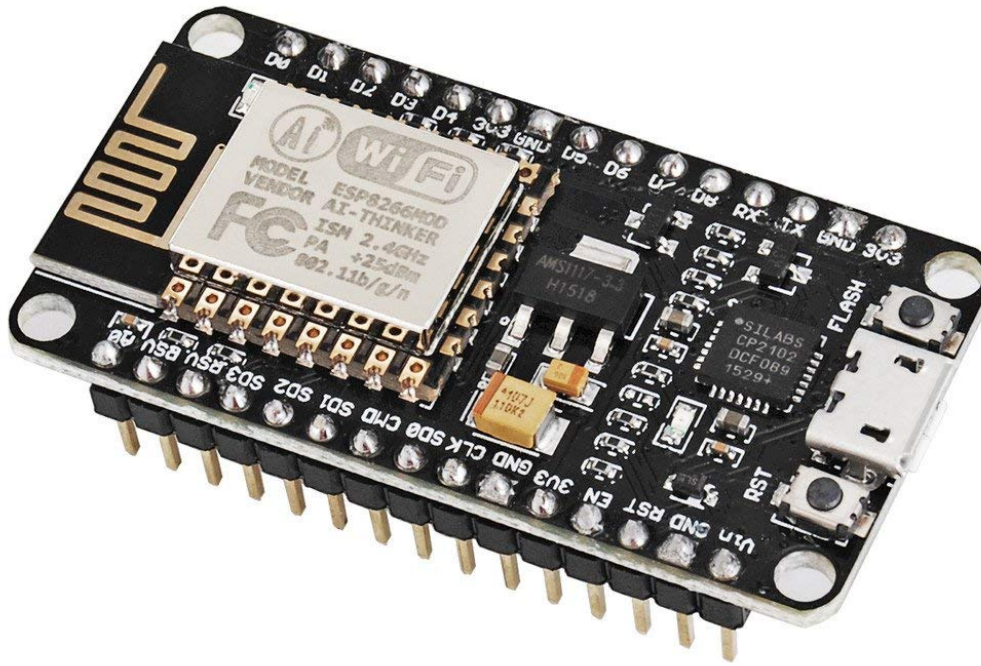
NodeMCU: NodeMCU [node + MCU (Micro controller unit)] is a low cost open source IoT platform developed by the ESP8266 opensource community

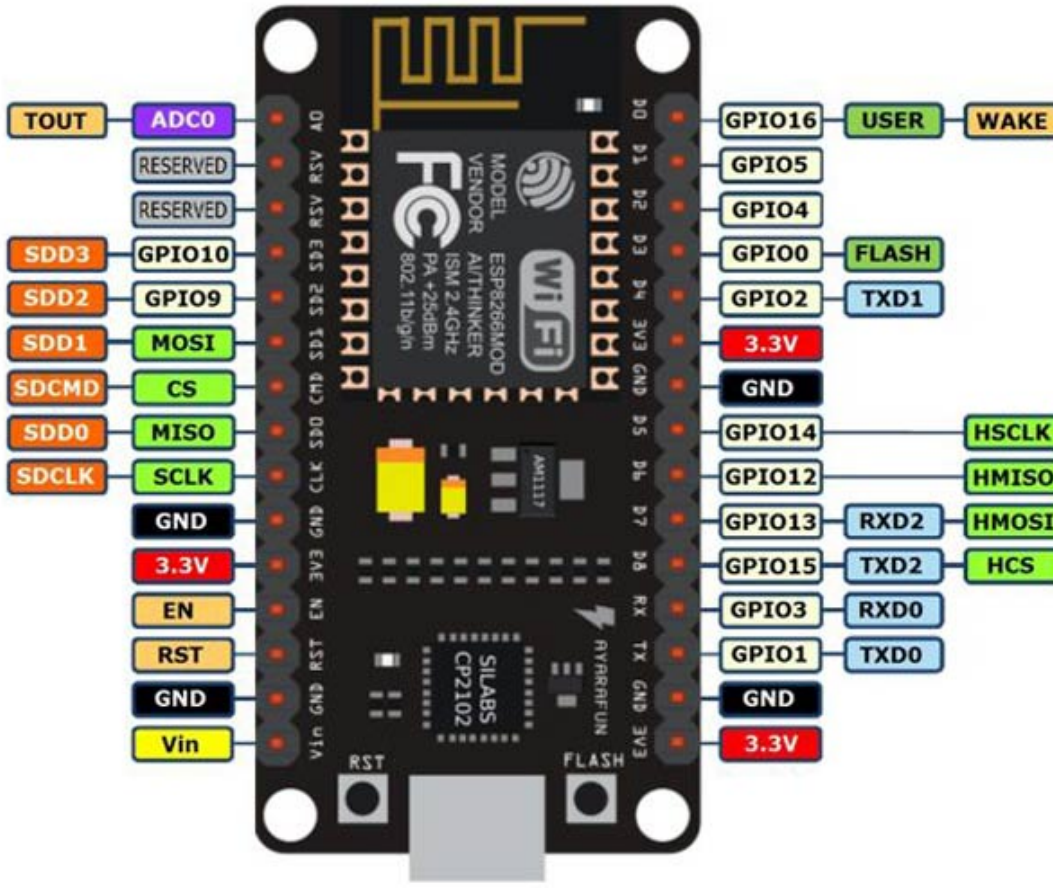
NodeMCU runs on the ESP8266 WiFi enabled microprocessor Arduino Uno run on the ATmga328p microcontroller.

NodeMCU works on 3.3V while Arduino boards work on 5V.

The NodeMCU contains 17 GPIO pins and 1 Analog input pin.

The ESP8266 based NodeMCU has a low power consumption, high memory and Wi-Fi built in. So, it is mostly used for prototyping IoT devices, Network related projects and low power operated applications while the Arduino boards are used for multiple I/O interfaces, beginner level projects and prototyping electronic products and systems.


MODEL ESP8266MOD
VENDOR AI-THINKER
ISM 2.4GHz
PA +25dBm
802.11b/g/n
AMS1117-3.3
H1518
SILABS
CP2102
FLASH
RST


TOUT
ADC0
RESERVED
RESERVED
SDD3
GPIO10
SDD2
GPIO9
SDD1
MOSI
SDCMD
CS
SDD0
MISO
SDCLK
SCLK
GND
3.3V
EN
RST
GND
Vin
GPIO16
USER
WAKE
GPIO5
GPIO4
GPIO0
FLASH
GPIO2
TXD1
3.3V
GND
GPIO14
HSCLK
GPIO12
HMISO
GPIO13
RXD2
HMOSI
GPIO15
TXD2
HCS
GPIO3
RXD0
GPIO1
TXD0
GND
3.3V
MODEL
VENDOR
ESP8266MOD
AITHINKER
ISM 2.4GHz
PA +25dBm
802.11b/g/n
SILABS
CP2102
AYRARFUN
RST
FLASH

To work with the Arduino and ESP boards, we need a software IDE which can be used to connect and upload the code for a project onto the boards.

Arduino IDE installation :

1. Download and Install the Arduino IDE from

   https://www.arduino.cc/en/software.

   → The drivers for Arduino boards are included by default.

   → For ESP based boards, follow below steps.

2. Goto File → Preferences

   Paste the following link in the Additional Board Manager URLs :

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

3. Goto Tools → Board → Boards Manager..

   Search for esp8266 and install the latest version.

10. Write program for RGB LED using Arduino.

Date

Exp. No. 10

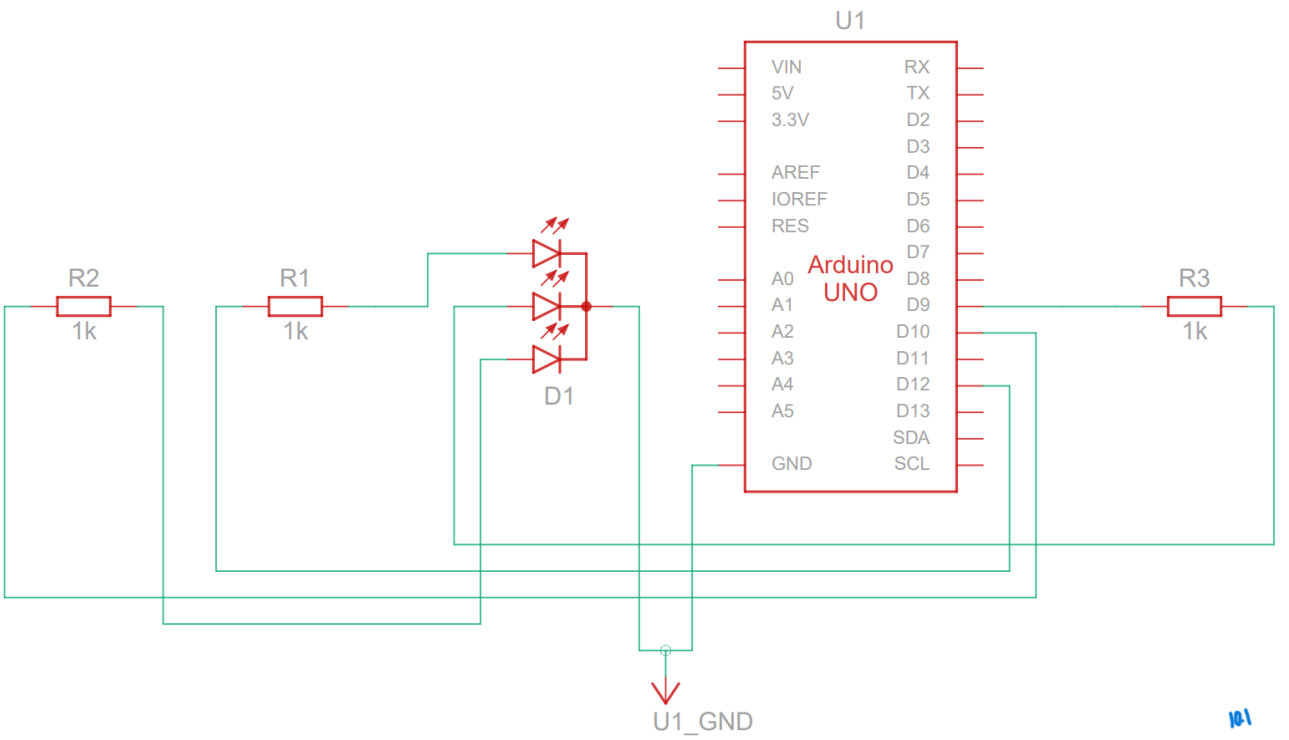
Aim: To write a program using TinkerCAD for Arduino to implement RGB LED Blink.

Components:

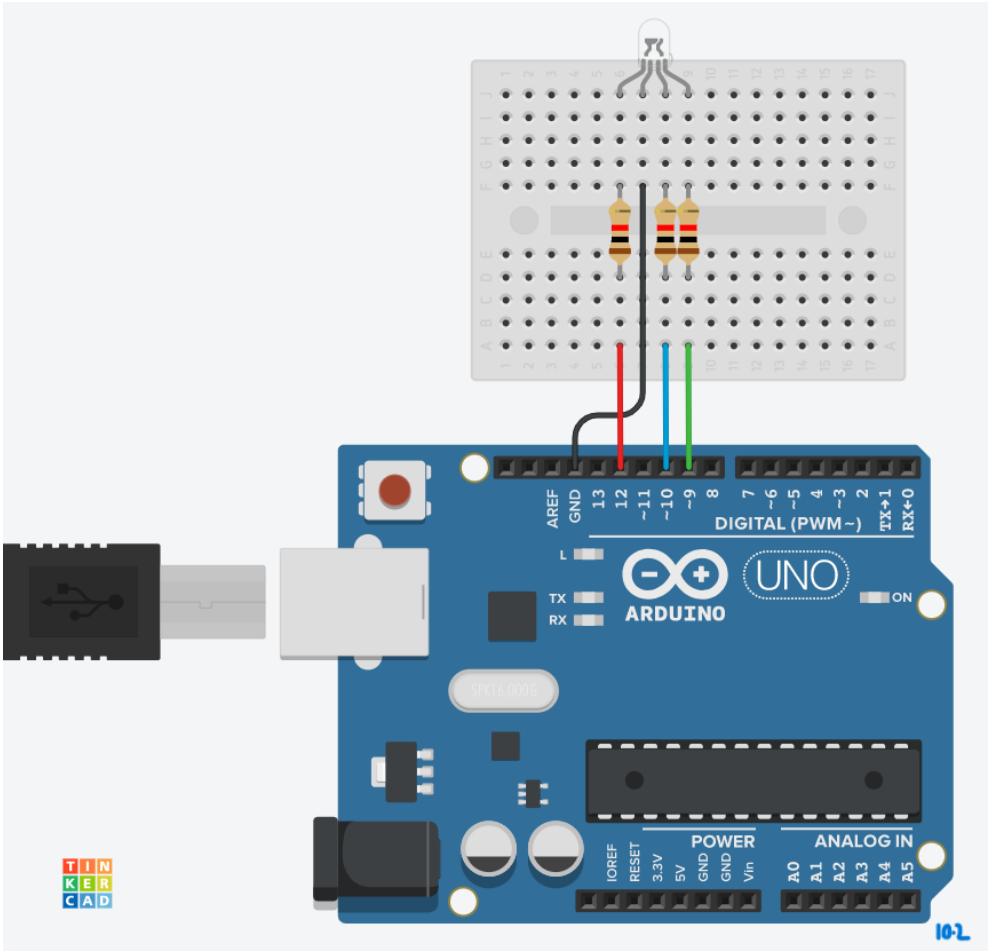
1. RGB LED
2. Resistor (220 Ω)
3. Breadboard
4. Arduino Uno R3
5. Jumper wires (M-M)
6. Misc: Computer (with TinkerCAD online software), USB power Cable (USB Type-A to Micro USB)

Procedure:

- Place the RGB LED in the breadboard such that 4 legs go into different rails.
- Connect the 3 resistors in series to the Red, Green, Blue pins as shown in the figure 10.2.
- As shown in the Fig. 10.1, Connect
  - → Red pin resistor's other end to digital pin D12.
  - → Blue pin resistor's other end to digital pin D10.
  - → Green pin resistor's other end to digital pin D9.


U1
VIN
5V
3.3V
AREF
IOREF
RES
A0
A1
A2
A3
A4
A5
GND
RX
TX
D2
D3
D4
D5
D6
D7
D8
D9
D10
D11
D12
D13
SDA
SCL
Arduino
UNO
R2
1k
R1
1k
D1
R3
1k
U1_GND

10-1


AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
L
TX
RX
ARDUINO
UNO
ON
POWER
ANALOG IN
IOREF
RESET
3.3V
5V
GND
GND
Vin
A0
A1
A2
A3
A4
A5
TIN
KER
CAD

10-2

Source Code :

```
int r = 12, b = 10, g = 9 ;

void setup()
{
    pinMode ( r, OUTPUT) ;    pinMode ( g, OUTPUT) ;   pinMode ( b, OUTPUT) ;
    Serial. begin (9600) ;
    Serial. println ( " This program is done by : 20881A1270 " ) ;
}
 void loop()
 {
     for ( int i=0 ;        i <= 255 ;  i += 64 )
     {
        analogWrite (r,i) ;  analogWrite( g, 255-i) ;  analogWrite ( b, 0) ;
        Serial.print (i) ;  Serial.print ( " , " ) ;  Serial. print (255 -i) ;
        Serial. print ( " , " ) ;  Serial. print (0) ;  Serial. print ( " \n " ) ;
         delay (1000) ;
        analogWrite ( g,i) ;  analogWrite( b, 255-i) ;  analogWrite (r,0) ;
         Serial. print (0) ;  Serial. print ( " , " ) ;  Serial. print ( i) ;
         Serial. print ( " , " ) ;  Serial. print ( 255-i) ;  Serial.print ( " \n " ) ;
         delay (1000) ;
         analogWrite ( b, i) ;  analogWrite (r, 255 -i) ;  analogWrite (g,0) ;
         Serial. print ( 255-i) ;  Serial. print ( " , " ) ;  Serial. print (0) ;
         Serial. print ( " , " ) ;  Serial. print ( i) ;  Serial. print ( ) ;
         delay (1000) ;
     }
     Serial. println ( " End of the loop. " ) ;
     delay (5000) ;
 }
```

Working Principle:

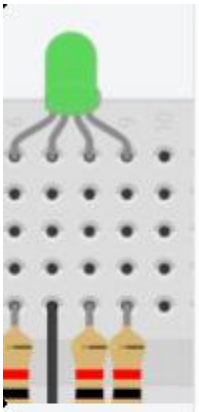
- When the circuit is connected as shown in the figure 10.1, the 3 digital pins (9, 10, 12) are set as output.
- When the loop starts, the 3 pins of the RGB LED are set to custom voltage values using the analogWrite function and differentiating i values using i, 255-i, 0 as inputs.
- As i is increased by 64 in every iteration from 0 to 255, there will be 4 iterations giving different combinational colors. are displayed in the Fig. 10.4.

1 (Arduino Uno R3)

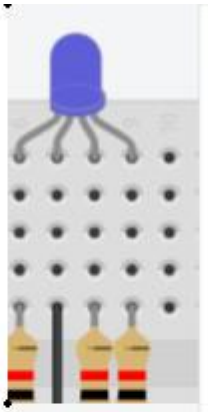
```
int r = 12, b = 10, g = 9;
void setup()
{
  pinMode(r, OUTPUT);
  pinMode(g, OUTPUT);
  pinMode(b, OUTPUT);
  Serial.begin(9600);
  Serial.println("This program is done by : 20881A1270");
}

void loop()
{
  for(int i = 0; i <= 255; i+=64)
  {
    analogWrite(r, i); analogWrite(g, 255-i); analogWrite(b, 0);
    Serial.print(i); Serial.print(","); Serial.print(255-i);
    Serial.print(","); Serial.print(0); Serial.print("\n");
    delay(1000);
    analogWrite(g, i); analogWrite(b, 255-i); analogWrite(r, 0);
    Serial.print(0); Serial.print(","); Serial.print(i);
    Serial.print(","); Serial.print(255-i); Serial.print("\n");
    delay(1000);
    analogWrite(b, i); analogWrite(r, 255-i); analogWrite(g, 0);
    Serial.print(255-i); Serial.print(","); Serial.print(0);
    Serial.print(",");  Serial.print(i); Serial.print("\n");
    delay(1000);
  }
  Serial.println("End of one loop.");
  delay(5000);
}
```

Serial Monitor

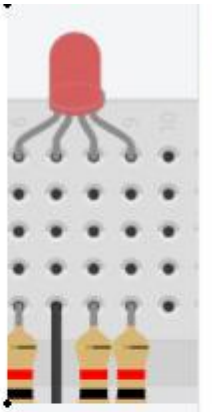
```
This program is done by : 20881A1270
0,255,0
0,0,255
255,0,0
64,191,0
0,64,191
191,0,64
128,127,0
0,128,127
127,0,128
192,63,0
0,192,63
63,0,192
End of one loop.
```



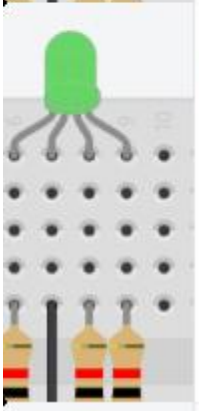
0,255,0



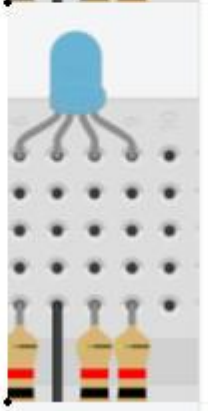
0,255,0
0,0,255



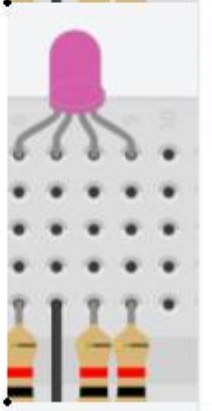
0,255,0
0,0,255
255,0,0



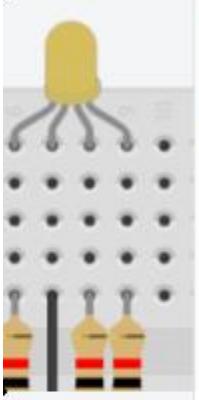
0,255,0
0,0,255
255,0,0
64,191,0



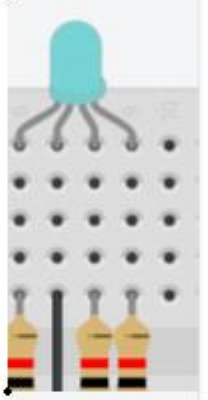
0,255,0
0,0,255
255,0,0
64,191,0
0,64,191



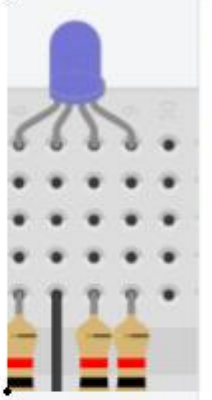
0,255,0
0,0,255
255,0,0
64,191,0
0,64,191
191,0,64



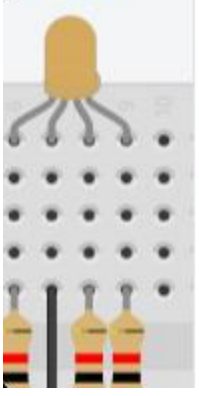
0,255,0
0,0,255
255,0,0
64,191,0
0,64,191
191,0,64
128,127,0



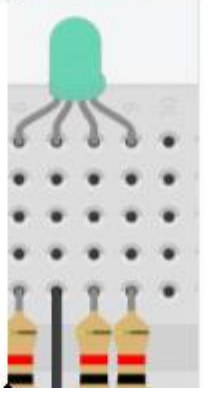
0,255,0
0,0,255
255,0,0
64,191,0
0,64,191
191,0,64
128,127,0
0,128,127



0,255,0
0,0,255
255,0,0
64,191,0
0,64,191
191,0,64
128,127,0
0,128,127
127,0,128



0,255,0
0,0,255
255,0,0
64,191,0
0,64,191
191,0,64
128,127,0
0,128,127
127,0,128
192,63,0



0,255,0
0,0,255
255,0,0
64,191,0
0,64,191
191,0,64
128,127,0
0,128,127
127,0,128
192,63,0
0,192,63

0,255,0
0,0,255
255,0,0
64,191,0
0,64,191
191,0,64
128,127,0
0,128,127
127,0,128
192,63,0
0,192,63
63,0,192
End of one loop.

12. Study and Configure Raspberry Pi.

Date

Exp. No. 12

Raspberry Pi : The Raspberry Pi is a low cost, credit card sized computer that plugs to a monitor or TV and used a keyboard and mouse. It is developed for teaching based on computer science and robotics by the Raspberry Pi Foundation.

Different versions and models of the Raspberry Pi include.

Raspberry Pi. [A, B, A+, B+].

Raspberry Pi 2 [B].

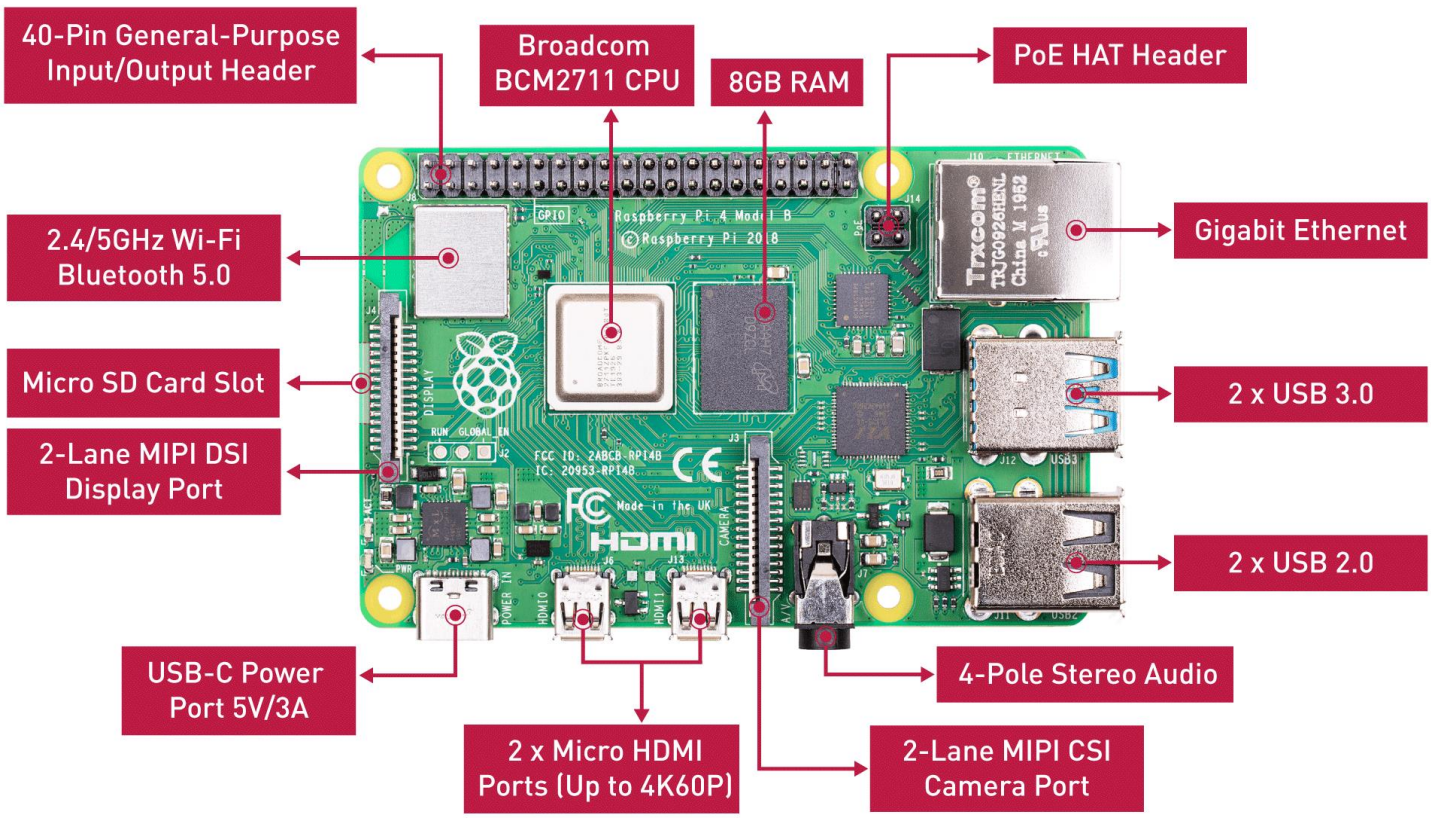
Raspberry Pi Zero

Raspberry Pi 3 [B, A+, B+]

Raspberry Pi 4 [B, 400].

Raspberry Pi Pico.

Raspberry Pi 4 Model B is the latest product in the popular Raspberry Pi range of computers. It offers ground-breaking increases in processing speed, multimedia performance, memory, and connectivity compared to the prior generation Raspberry Pi 3 Model B+, while retaining backwards compability and similar power combination. For the end user, Raspberry Pi 4 Model B provides desktop performances comparable to an entry level x86 PC.


40-Pin General-Purpose Input/Output Header
Broadcom BCM2711 CPU
8GB RAM
PoE HAT Header
2.4/5GHz Wi-Fi Bluetooth 5.0
Gigabit Ethernet
Micro SD Card Slot
2 x USB 3.0
2-Lane MIPI DSI Display Port
2 x USB 2.0
USB-C Power Port 5V/3A
4-Pole Stereo Audio
2 x Micro HDMI Ports (Up to 4K60P)
2-Lane MIPI CSI Camera Port

The Raspberry Pi 4 offers 40 GPIO pins some of which have multiple connection support.