IoT Introduction

What is the history of the connected objects?

The history of the Internet of Things began in the early 1980s with the first connected objects. A group of students at the Carnegie Melon University created a way to get Coca-Cola vending machine to report on its content through a network. They installed micro-switches into the machine to report on how many Coke cans were available and if they were cold. So cool!

But, it was only in 1999 that the Term of "Internet of Things" was coined by Kevin Ashton. Kevin Ashton was born in 1968. He is a British technology pioneer who created a global standard system for Radio-Frequency Identification (RFID) and other sensors. RFID uses electromagnetic fields to automatically identify and track tags attached to object. These bar code are essential to the Internet of Things, which allow computers to manage all individual things.

And that same year (1999), MIT professor Neil Gershenfeld's book "When Things Start to Think" was published, explaining the concept of Internet of Things and envisioning its future role, but without using the term. Prof. Neil Gershenfeld is the director of MIT's Center for Bits and Atoms where his unique laboratory is breaking down

boundaries between the digital and physical worlds, from pioneering quantum computing to digital fabrication to the Internet of Things.

Definition of IoT:

A dynamic global network infrastructure with self-configuring capabilities based on standard and interoperable communication protocols where physical and virtual "things" have identities, physical attributes, and virtual personalities and use intelligent interfaces and are seamlessly integrated into the information network, often communicate data associated with users and their environments.

Characteristics of IoT:

1) Dynamic & Self-Adapting

2) Self-Configuring

3) Interoperable Communication Protocols

4) Unique Identity

5) Integrated into Information Network.

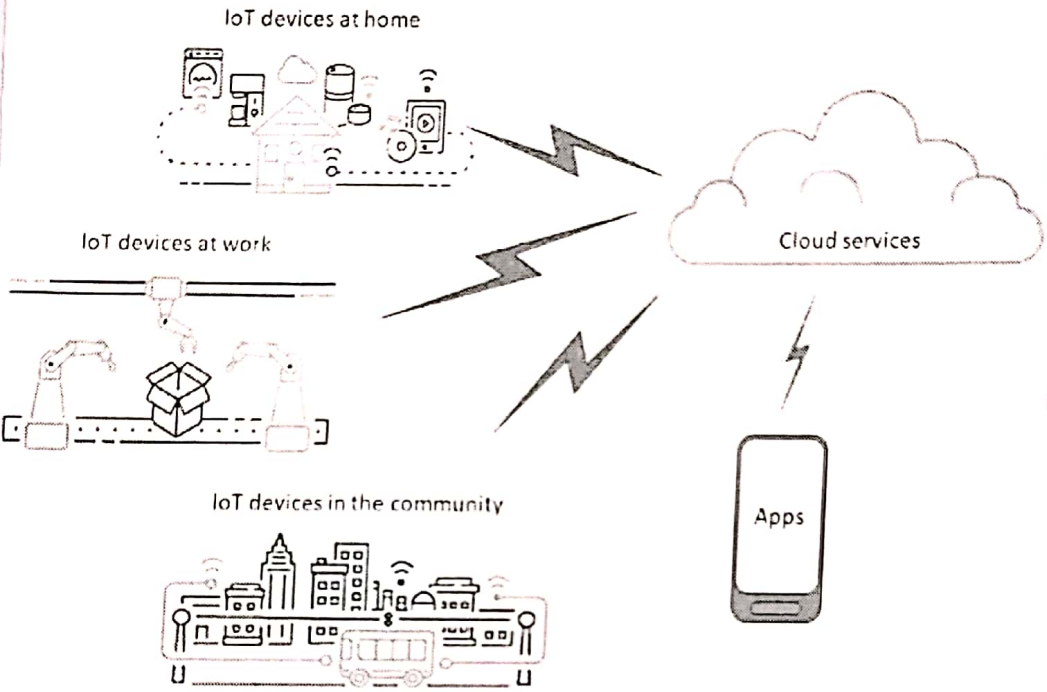
How does the IoT work?

The Internet of Things consists of ordinary devices that can connect to the Internet and communicate with each other via the cloud and connect to the Internet through a WiFi network, a cellular connection (3G or 4G and 5G now!) or Bluetooth.

One thing is really important and without this, the objects cannot communicate. There are the sensors. Sensors are added to objects such as washing machines, heaters, watches and almost anything else.

Some objects use these sensors to collect and relay information like the vending machine discussed above. Other objects can receive information and then perform an action. For example, the smart door locks receive a signal that we want to open the door and open the door.

But some objects can do both. For example, in an industrial setting, it may be monitoring machine looking for possible problems, which triggers an alarm when it is detected. On the other hand, in the home environment, it could be a smart thermostat, which collects information about temperature preferences and habits, and then acts accordingly to heat or cool the home to the good temperature


IoT devices at home
IoT devices at work
Cloud services
IoT devices in the community
Apps

What are advantages of IoT?

The sensors in the connected objects could improve their performance and efficiency. The "Things" in IoT ecosystem and respond immadiately to an event., and sate time, effort and money. Altogether, IoT can significantly contribute to the well being of the systems and people who use it. IoT can help to provide good Quality of Service (QoS) of the systems, and can help to improve the overall quality of life of the people.

Use Cases: Examples of IoT:

1) The Internet of Things can bring home and commercial security with smart locks, alarms and video surveillance systems.

2) For the farmers, the Internet of Things help them like automate farming techniques, take informed decisions to improve quality and quantity, minimize risk and waste, and reduce the effort required to manage crops. The main goal is to help them increase farm productivity, reduce crops.

3) The Internet of Things could have an impact on the environment for example with forest fire detection, air pollution etc.

Task - 1

RaspberryPi Introduction

Raspberry Pi is a mono-board computing platform that's as tiny as a credit card. Initially it was developed for Computer Science Education, but due to its reliability, performance, open design, modularity and low cost it is now being widely used in many areas such as weather forecasting, robotics, instrumentation etc.

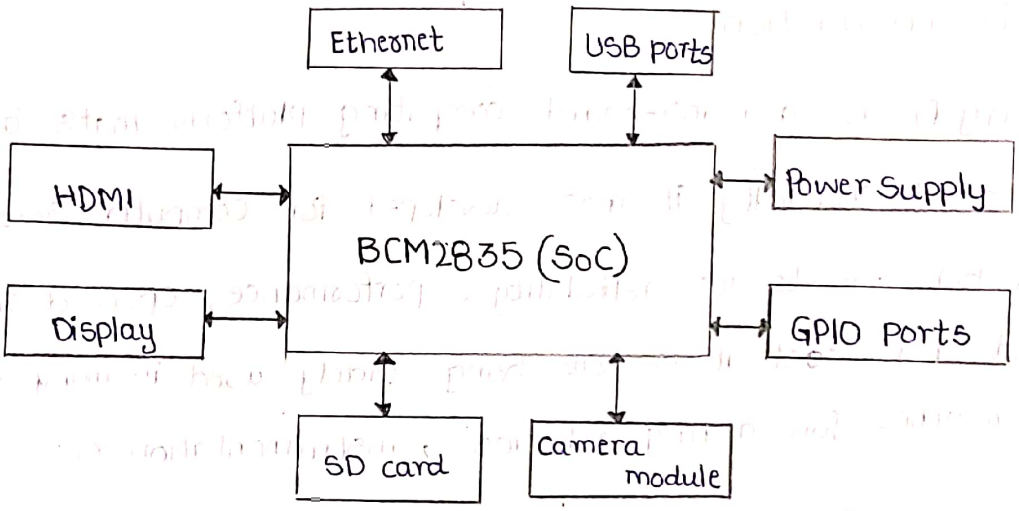
The Raspberry Pi hardware has evolved through several versions that feature variations in the type of the central processing unit, amount of memory capacity, networking support, and peripheral-device support.

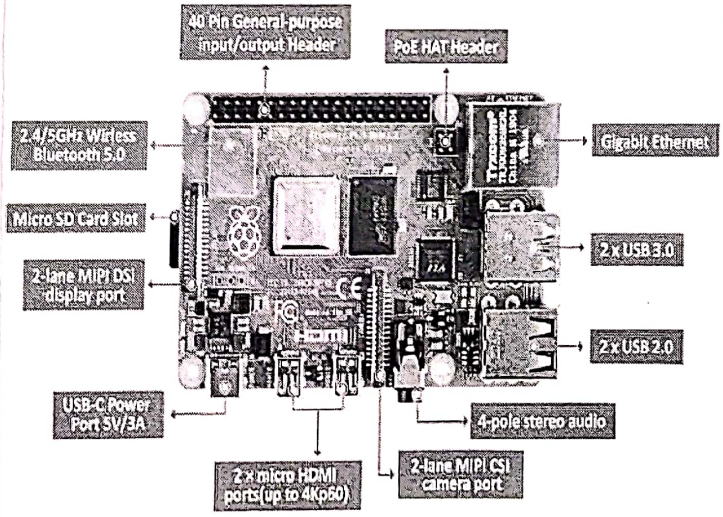
The Raspberry Pi Foundation in association with Broadcom, UK, has developed many small size single board Raspberry Pi computers.

Raspberry Pi Family

- Raspberry Pi 5.
- Raspberry Pi 4 Model B.
- Raspberry Pi 43 Model B+
- Raspberry Pi 3 Model B
- Raspberry Pi 2 Model B
- Raspberry Pi 1 Model B+
- Raspberry Pi 3 Model A+

Block Diagram of Raspberry Pi





- Raspberry Pi 1 Model A+
- Raspberry Pi - Zero 2 W
- Raspberry Pi Zero W
- Raspberry Pi - Zero

Raspberry Pi Features:

1) Processor (CPU) : Raspberry Pi models use ARM-based processors. All models use Broadcom processors and typically have the prefix "BCM" (eg. BCM2835, BCM2836, BCM 2837 or BCM2711).

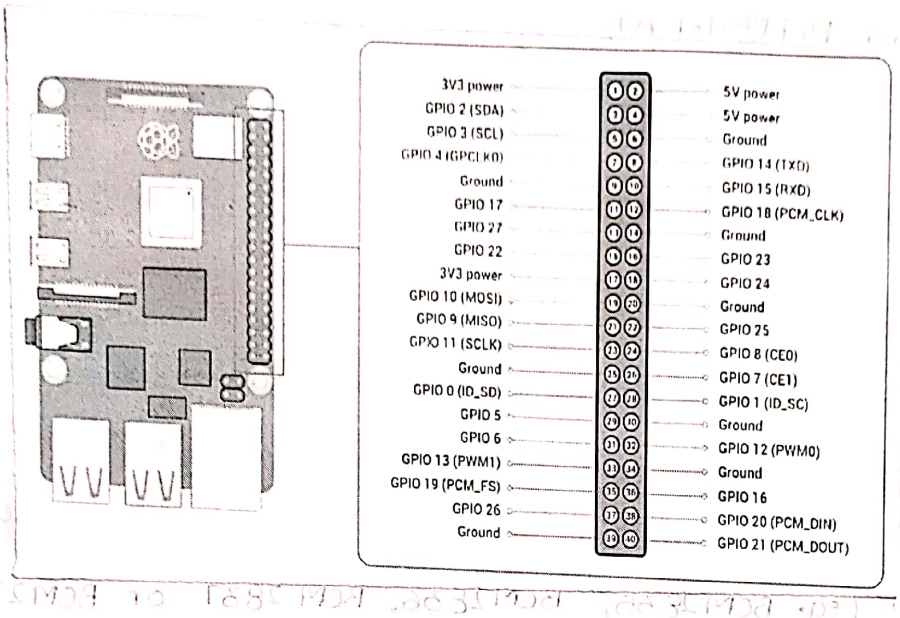
2) RAM (Memory) : The amount of RAM on a Raspberry Pi varies across models. Older models have lower RAM compared to newer ones. The lowest is 256MB while the newest has 8GB.

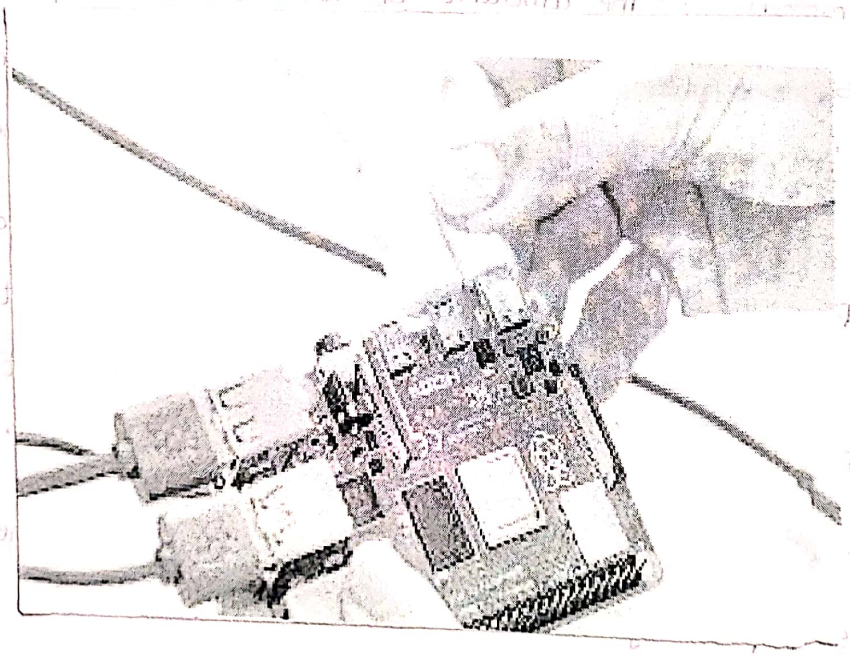
3) GPIO (General Purpose Input/Output): Raspberry Pi boards include a set of GPIO pins that allow for interfacing with external devices and components. The latest RPi models have standard 40 Pin GPIO, while the older models had 26 pin GPIO.

4) USB ports : The Raspberry Pi boards have one to five USB ports depending on the model.

5) Video Output : Most Raspberry Pi models have an HDMI port for connecting to monitors or TVs.

06(a)


3V3 power
GPIO 2 (SDA)
GPIO 3 (SCL)
GPIO 4 (GPCLK0)
Ground
GPIO 17
GPIO 27
GPIO 22
3V3 power
GPIO 10 (MOSI)
GPIO 9 (MISO)
GPIO 11 (SCLK)
Ground
GPIO 0 (ID_SD)
GPIO 5
GPIO 6
GPIO 13 (PWM1)
GPIO 19 (PCM_FS)
GPIO 26
Ground
5V power
5V power
Ground
GPIO 14 (TXD)
GPIO 15 (RXD)
GPIO 18 (PCM_CLK)
Ground
GPIO 23
GPIO 24
Ground
GPIO 25
GPIO 8 (CE0)
GPIO 7 (CE1)
GPIO 1 (ID_SC)
Ground
GPIO 12 (PWM0)
Ground
GPIO 16
GPIO 20 (PCM_DIN)
GPIO 21 (PCM_DOUT)



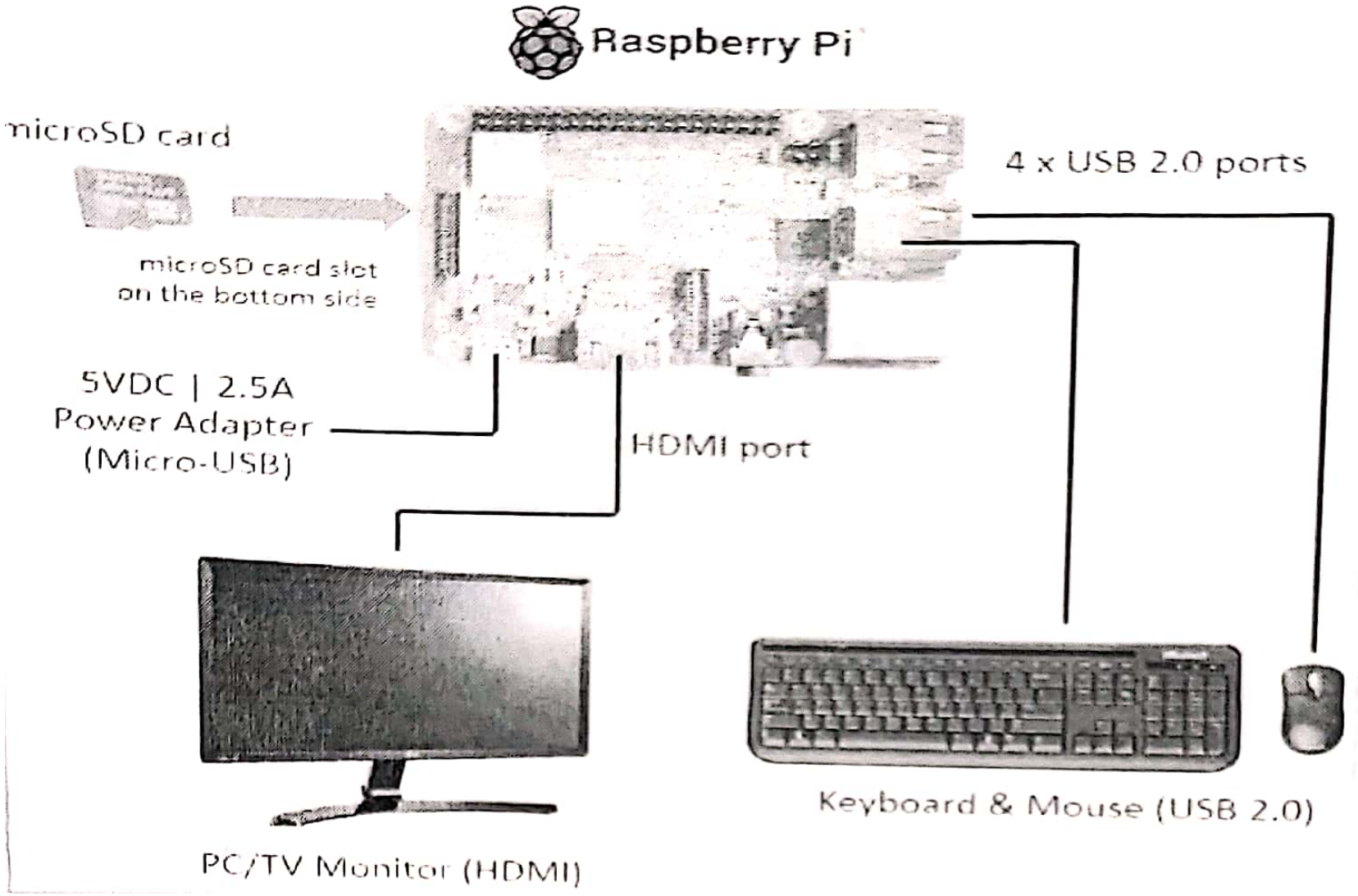
6) Camera and Display Ports : Certain Raspberry Pi models have dedicated ports for connecting the Raspberry Pi Camera Module and the Raspberry Pi Touchscreen Display.

7) Ethernet Port : Raspberry Pi Ø modules include an Ethernet Port for wired network connectivity.

8) WiFi and Bluetooth : Latest Raspberry Pi models come with built-in WiFi and Bluetooth capabilities, enabling wireless network connectivity and communication with Bluetooth enabled devices.

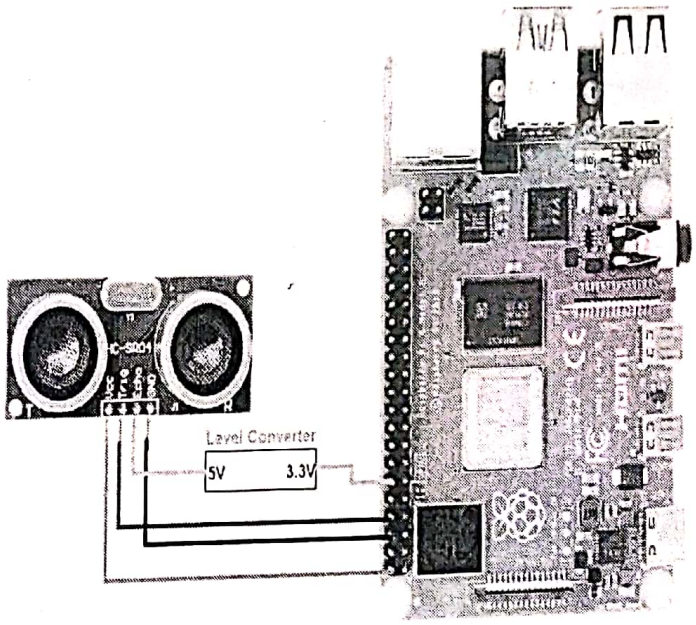
9) Storage : Raspberry Pi boards do not have built-in storage but support micro SD cards for permanent storage.

10) OS support : Raspberry Pi supports a variety of operating systems, including Raspbian (now called Raspberry Pi OS), ubuntu and other Linux distributions.



1) Interfacing RPi and Distance Sensor

```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
GPIO_TRIG = 11
GPIO_ECHO = 18
GPIO.setup(GPIO_TRIG, GPIO.OUT)
GPIO.setup(GPIO_ECHO, GPIO.IN)
GPIO.output(GPIO_TRIG, GPIO.LOW)
Time.sleep(2)
GPIO.output(GPIO_TRIG, GPIO.HIGH)
Time.sleep(0.00001)
GPIO.output(GPIO_TRIG, GPIO.LOW)
while GPIO.input(GPIO_ECHO)==0:
        Start_time = time.time()
while GPIO.input(GPIO_ECHO)==1:
    Bounce_back_time = time.time()
pulse_duration = Bounce_back_time - Start_time
distance = round(pulse_duration * 17150, 2)
print(f"Distance: {distance} cm")
GPIO.cleanup()
```


Level Converter
5V
3.3V

2) Interfacing RPi with LED

What is an LED?

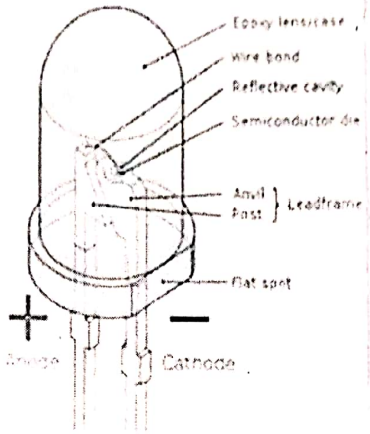
- LED - light emitting Diode
- A semiconductor component similar to a transistor or an integrated circuit.
- Electrical current through the semiconductor chip produces light
- Semiconductor materials used define the color of light produced.

LED program - Simple

```
from gpiozero import LED
from time import sleep
led = LED(17)
while True:
    led.on()
    sleep(1)
    led.off
    sleep(1)
```

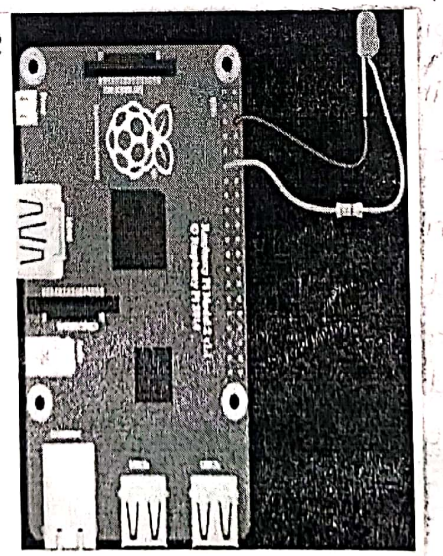
What is an LED?

- LED - Light Emitting Diode
- A semiconductor component similar to a transistor or an integrated circuit
- Electrical current through the semiconductor chip produces light
- Semiconductor devices



LED Program -Simple

```
from gpiozero import LED
from time import sleep

led=LED(17)

while True:
    led.on()
    sleep(1)
    led.off()
    sleep(1)
```



LED program - Standard

```
#program to blink the LED three times
    import RPi.GPIO as GPIO
    from time import sleep
# To dis able warnings
    GPIO.setwarnings(False)
# Set the mode of the GPIO pins (Board/BCM)
    GPIO.setmode(GPIO.BCM)
# Initialize
    Blink_count = 3
    count = 0
    LEDPin = 17
# Setup the pin the LED is connected to
    GPIO.setup(LEDPin, GPIO.OUT)

    try:
        while count < Blink_count:
            GPIO.output(LEPin, True)
            print("LED on")
            sleep(3)
            GPIO.output(LEDPin, False)
            print("LED off")
            Sleep(1)
            count += 1

    finally:
        # Reset the GPIO pins to safe state
        GPIO.cleanup()
```

TASK -2

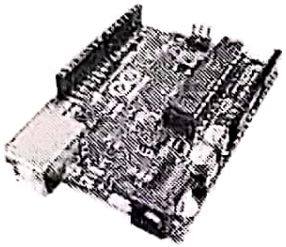
Introduction to ARDUINO:

Arduino is a prototype platform (open-source) based on an easy-to-use hardware and software. It consists of a circuit board, which can be programmed (referred to as a microcontroller) and a ready-made software called Arduino IDE (Integrated Development Environment), which is used to write and upload the computer code to the physical board.

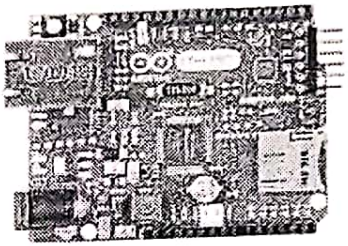
The key features are:

- Arduino boards are able to read analog or digital input signals from different sensors and turn it into an output such as activating a motor, turning LED on/off, connect to the cloud and many other actions.
- You can control your board functions by sending a set of instructions to the micro controller on the board via Arduino IDE (referred to as uploading software).
- Unlike most previous programmable circuit boards, Arduino does not need an extra piece of hardware (called a programmer) in order to load a new code onto the board. You can simply use a USB cable

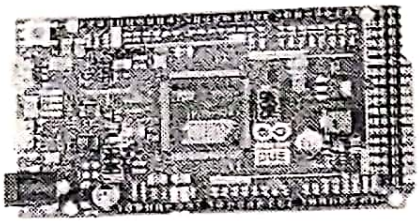
TYPES OF ARDUINO BOARD...



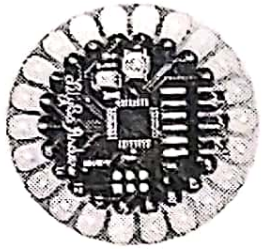
Arduino Uno – R3



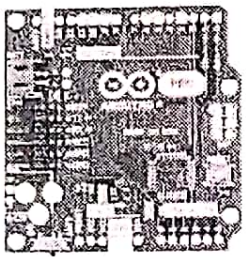
Arduino Ethernet



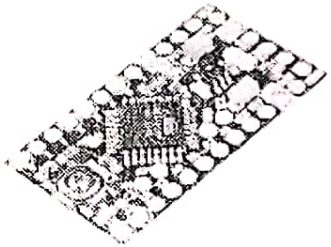
Arduino Due



LilyPad Arduino 328

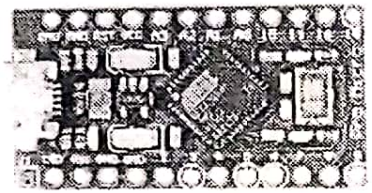


Arduino Pro

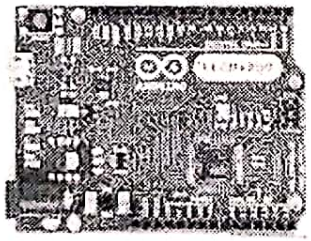


Arduino Pro Mini

Arduino Mini

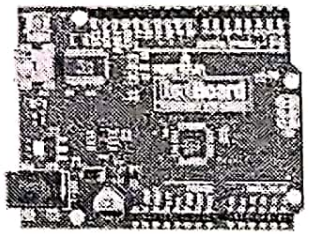


Arduino Pro Micro

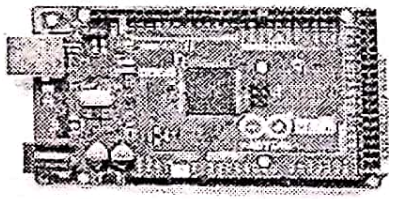


Arduino Leonardo

Arduino Fio



Arduino RedBoard



Arduino Mega

DO YOU KNOW ANY OTHER VERSION OF AURDUINO?

• Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program.

Arduino-Pin Details

• Vin - This is the input voltage pin of the Arduino board used to provide input supply from an external power source.

• 5V - This pin of the Arduino board is used as a regulated power supply voltage and it is used to give supply to the board as well as onboard components.

• 3.3V - This pin of the board is used to provide a supply of 3.3V which is generated from a voltage regulator on the board.

• GND - This pin of the board is used to ground the Arduino board.

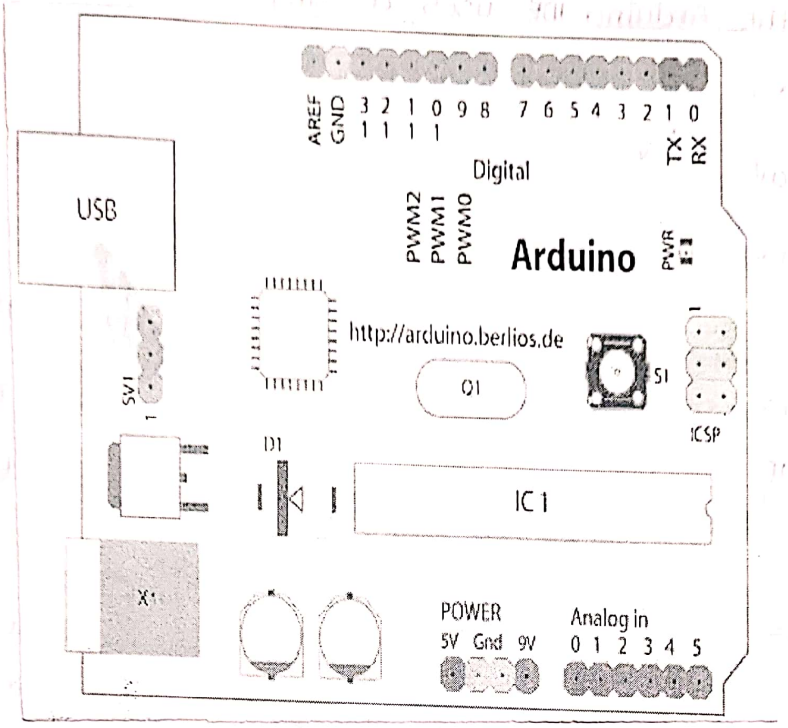
• Reset - This pin of the board is used to reset the microcontroller. It is used to Resets the microcontroller.

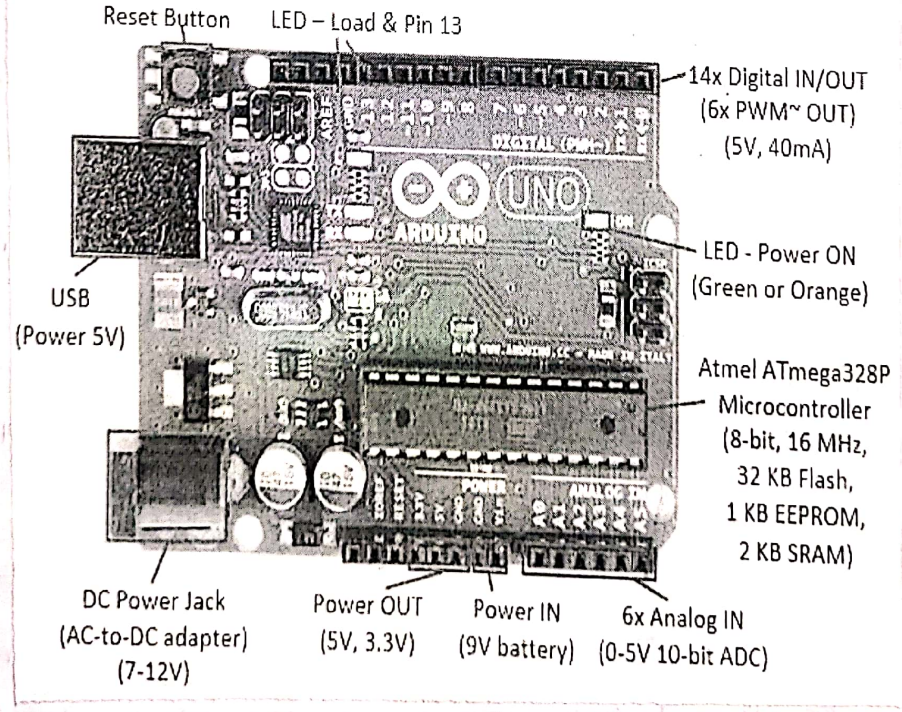
• Analog Pins - The pins A0 to A5 are used as an analog input and it is in the range of 0-5V

• Digital Pins - The Pins 0 to 13 are used as a digital input or output for the Arduino board.

• Serial Pins - These pins are also known as a UART pin. It is used for communication between the Arduino board and a computer or other devices. The transmitter pin number 1 and receiver pin number 0 is used to transmit and receive data

12 (a)



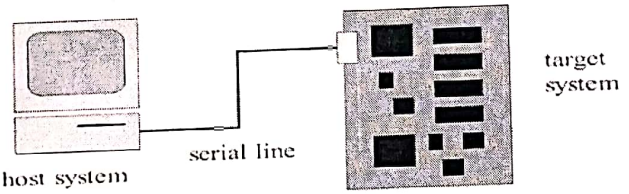


- External Interrupt Pins - This Pins of the Arduino board is used to produce the External interrupt and it is done by pin numbers 2 and 3.
- PWM pins - This pins of the board is used to convert the digital Siginal into an analog by varying the width of the pulse. The pin numbers 3, 5, 6, 9, 10 and 11 are used as a PWM pin.
- SPI pins - This is the Serial Peripheral Interface pin, it is used to maintain SPI communication with the help of the SPI library. SPI pins include:
  - SS : Pin number 10 is used as a Slave Select
  - MOSI : Pin number 11 is used as a Master Out Slave IN
  - MISO : Pin number 12 is used as a Master In Slave Out
  - SCK : Pin number 13 is used as a Serial Clock.
- LED Pin - The board has an inbuilt LED using digital pin-13. The LED glows only when the digital pin becomes high.
- AREF Pin - This is an analog reference pin of the Arduino board. It is used to provide a reference voltage from external power supply.

13(a)

Host/target design

Use a host system to prepare software for target system:



Steps to Build a program with Arduino IDE and Deploy on the Arduino Board.

Go to Arduino website and install latest version 2.3 or (older ver 1.8) on the Host Machine.

Host Machine

Step 1: Open the UI workspace

Step 2: Open a new sketch

Step 3: Type the Arduino Program

Step 4: Save in your folder with file n.ion extension

Step 5: Compile / verify Program

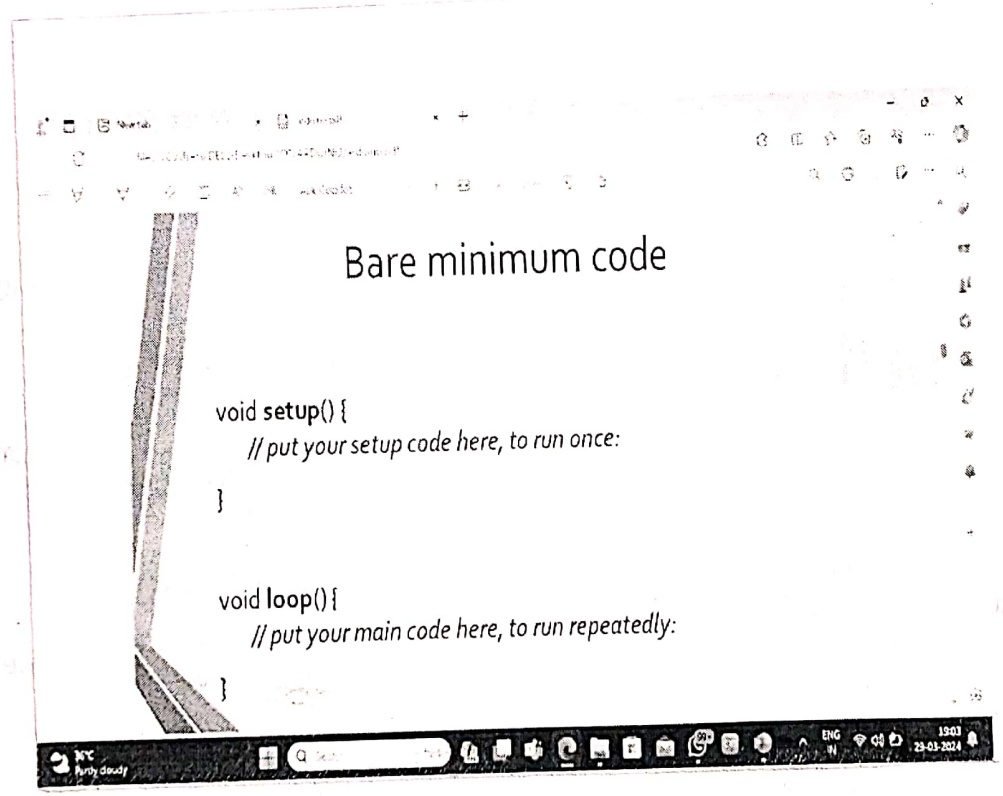
Step 6: After successful compilation - Import Library, Set the port

Target Machine

Step 7: Upload the program from Host Machine to Target Machine

Step 8: Open Serial Monitor or any other UI to see the Output.

14(a)



Structure of Arduino Code

Bare Minimum Code

```
void setup() {
    //put your setup code here, to run once:
}
void loop() {
    //put your main code here, to run repeatedly:
}
```

Distance Sensor (Ultrasonic Sensor - HC-SR04)

1. HC-SR04 stands for High-Conductance Ultrasonic Sensor. HC-SR04 is a Digital Sensor.
2. The HC-SR04 is an easy to use distance measuring sensor which has a range from 2cm to 400cm (about an inch to 13 feet). It's basically a SONAR which is used in submarines for detecting underwater objects.
3. The sensor measures how far things are without touching them, and it uses sound waves to get the measurements right.
4. The Sensor is composed of two ultrasonic transducers:
   - One is transmitter which outputs ultrasonic sound pulses and
   - the other is receiver which listens for reflected waves.

How the HC-SR04 Ultrasonic Distance Sensor Works?

- It emits an ultrasound waves at 40000 Hz which travels through the air at 343m/sec.
- If there is an object or obstacle on its path it will bounce back to the module.
- Considering the travel time and the speed of the sound you can calculate the distance.

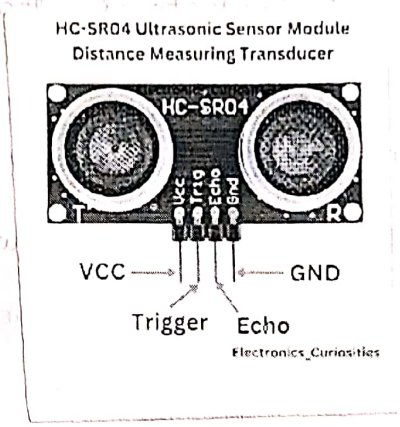
Distance Sensor (Ultrasonic Sensor - HC-SR04)

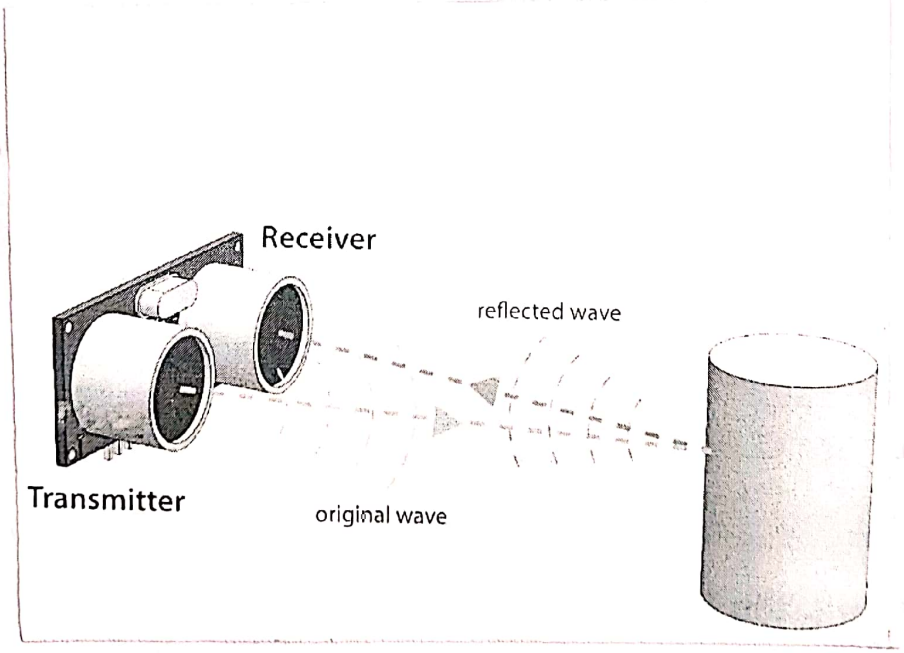
1. HC-SR04 stands for High-Conductance Ultrasonic Sensor. HC-SR04 is a Digital Sensor.
2. The HC-SR04 is an easy to use distance measuring sensor which has a range from 2cm to 400cm (about an inch to 13 feet). It's basically a SONAR which is used in submarines for detecting underwater objects.
3. The sensor measures how far things are without touching them, and it uses sound waves to get the measurements right.
4. The Sensor is composed of two ultrosonic transducers:
   - One is transmitter which outputs ultrasonic sound pulses and
   - the other is receiver which listens for reflected waves.

How the HC-SR04 Ultrasonic Distance Sensor Works?

- It emits an ultrasound waves at 40000 Hz which travels through the air at 343m/sec.
- If there is an object or obstacle on its path it will bounce back to the module.
- Considering the travel time and the speed of the sound you can calculate the distance.

16(a)





HC-SR04 Functioning

- **Trig Pin:**

  In order to generate the ultrasound we need to set the Trig pin on a High state for 10 µs. (That will send out an ultrasonic burst which will travel at the speed of sound).

- **Echo Pin:**

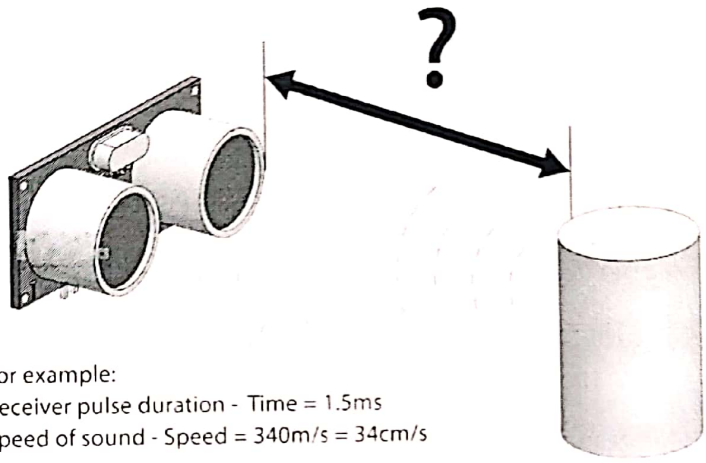
  The Echo pins goes high immediately (right after that ultrasonic burst is send,) and it starts listening or waiting for that wave to be Reflected from the object.

  - If there is no object or reflected pulse, the echo pin will time-out after 38ms and get back to low state.
  - If we receive a reflected pulse, the Echo pin will go down sooner than those 38ms. According to the amount of time the Echo pin was HIGH, we can determine the distance the sound wave travelled, thus the distance from the sensor to the object.

HC-SR04 - Distance Calculation

- For that purpose we are using the following basic formula for calculating distance:
  - Distance = Speed x Time
  - The time is the amount of time the Echo pin was HIGH, and
  - The speed is the speed of sound which is 340 m/s (approx).

17(a)



For example:
Receiver pulse duration - Time = 1.5ms
Speed of sound - Speed = 340m/s = 34cm/s

Distance = (Speed x Time) / 2
Distance = (34cm/ms x 1.5ms) / 2 = 25.5cm

- There's one additional step we need to do, and that's divide the end result by 2. and that's because we are measuring the duration the sound wave needs to travel to the object and bounce back.

Eg: Let's say the Echo Pin was HIGH for 2ms.

Distance = (speed x Time) / 2 = (34 cm/ms x 2ms) / 2 = 34cm

so, if the Echo pin was HIGH for 2ms (which we measure using the pulseIn() function), the distance from the sensor to the object is 34cm.

Calculation : in millisec

340m = 340 x 100 cm

1sec = 1000ms

340m/s = 34000 / 1000 cm/ms = 34cm/ms

Calculation : in microsec

The pulseIn() gives time in microsec

1sec = 1000000 µs

340 m/sec = 340 x 100cm / 1000000 microsec = 0.340 cm/µs

ARDUINO programming

setup: set the Arduino pins to interact with HC-SR04

Pin 12 - Output pin , Pin 11 - Input pin

Loop:

Activate the Trigger - to send sound waver Burst

make trigger pin Low for 2ms

Then set trig pin High for 10ms

Again make trig pin low for 2ms

Echo: Receives the sound waves after some time

Record that time duration using pulseIn() function, time in microsec

Calculate the Distance.

Convert the distance in Cm, inch.

Display the values on Desktop.

Program: Interfacing Arduino with Distance sensor - Ultrasonic Sensor.

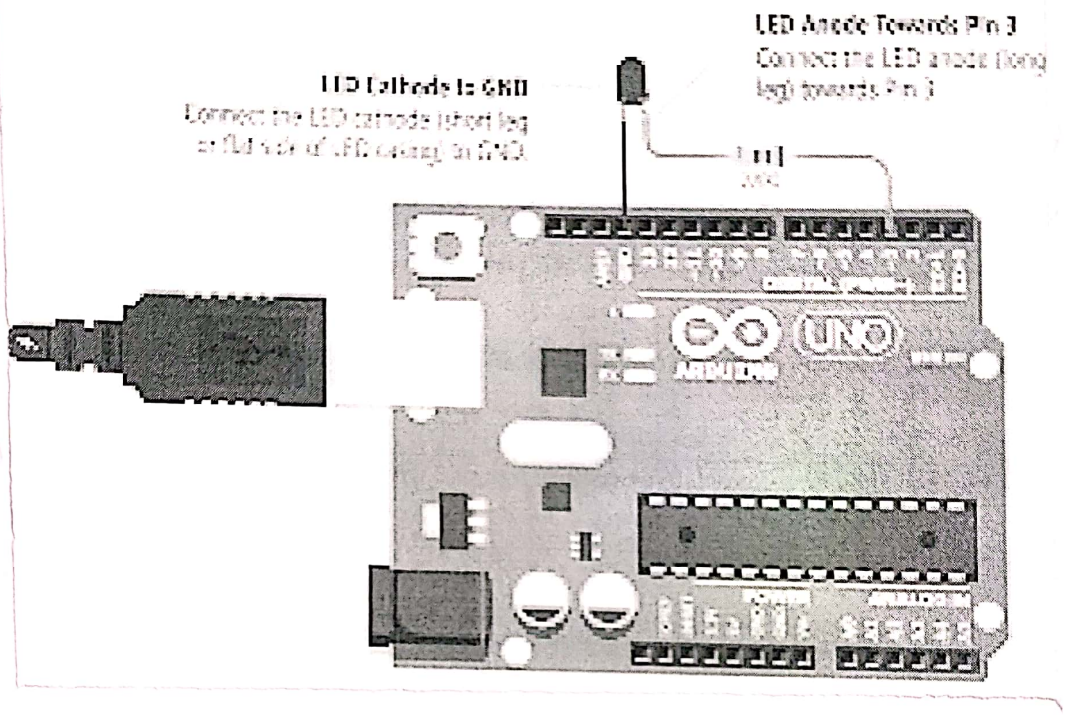
```
int trig =12;
int echo = 11;
int time_microsec;
int time_ms;
int dist_cm;
void setup () {
    //put your setup code here, to run once:
    Serial.begin(9600);
    pinMode(12, OUTPUT);
    pinMode(11, INPUT);
}
void loop () {
    //put your main code here, to run repeatedly:
    digitalWrite(trig, LOW);
    delayMicroSeconds(2);
    digitalWrite(trig, HIGH);
    delayMicroseconds(10);
    digitalWrite(trig, LOW);
    //pulseIn gives time in micro seconds, 1sec = 1000000 microseconds
    time_microsec = pulseIn(echo, HIGH);
    time_ms = time_microsec/1000;
    dist_cm = (34 * time_ms)/2;
```

```
Serial.print("Time to travel: ");
Serial.println(time_ms,1);
Serial.print("ms | ");
Serial.print("Distance: ");
Serial.print(dist_cm, 1);
Serial.print("cm");
delay(2000);
}
```

Program : Interfacing Arduino with LED

```
void setup() {
  // initialize digital pin LED_BUILTIN as an OUTPUT.
  pinMode (LED_BUILTIN, OUTPUT);
}
void loop() {
  digitalWrite(LED_BUILTIN , HIGH); // turn the LED on
  delay (1000); // wait for a second.
  digital Write (LED_ BUILTIN, LOW); // turn the LED off
  delay(1000); //wait for a second.
}
```

22(a)



program: Interfacing Arduino with DHT11 - Temperature and Humidity Sensor

```
#include "DHT.h"
#define DHTPIN 2
#define DHTTYPE DHT11
//DHT class dht Object with parameters - assigned pin and DHT Type
DHT dht (DHTPIN, DHTType);
void setup() {
    Serial.begin(9600);
    Serial.println(F("DHTxx test!"));
    dht.begin();
}
void loop() {
   delay(2000);
   float h = dht.readHumidity();
   Float t = dht.readTemperature();
   float f = dht.readTemperature(true);
   if (isnan(h) || isnan(t) || isnan(f)) {
       Serial.println(F("Failed to read from DHT sensor!"));
       return;
   }
```

```
Serial.print(F("Humidity: "));
Serial.print(h);
Serial.print(F(" % Temperature: "));
Serial.print(t);
Serial.print(F(" °C "));
Serial.print(f);
Serial.print(F("°F "));
Serial.println(" ");
}
```

TASK-3

Node MCU

Node MCU is a microcontroller development board with wifi capability

- On the Node MCU board - Besides the ESP12 Module, it contains other elements such as crystal oscillator, voltage regulator, power connector, Led indicators etc.
- ESP12 Module - The module contains ESP8266 SOC, networking (WiFi), and even a modern operating system and SDK.
- The ESP8266 is designed and manufactured by Espressif Systems.

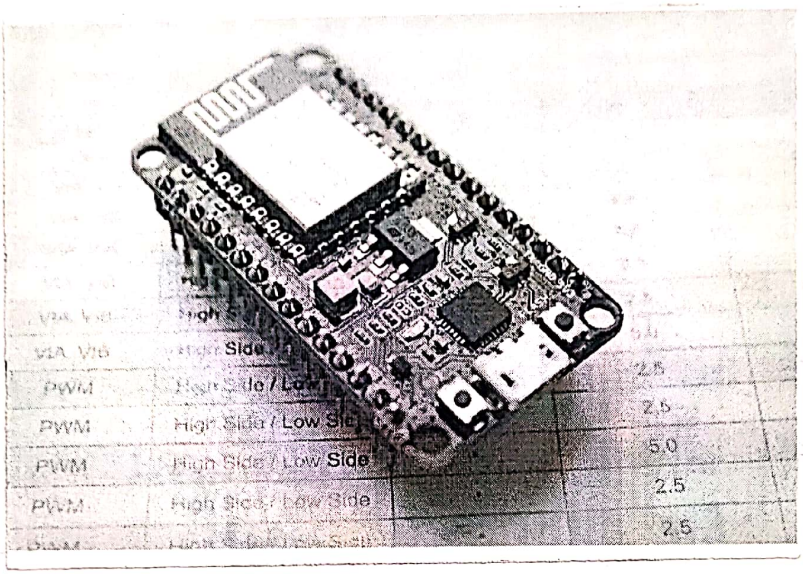
ESP12 Module

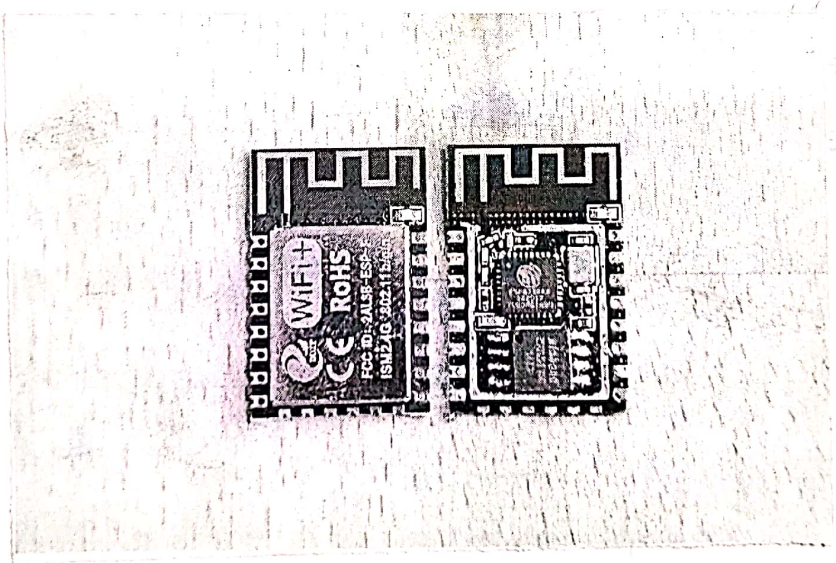
- The ESP is an RF module containing the ESP8266 SoC and other necessary components, like the antenna. With a module like this, it's a lot easier to design products with WiFi capabilities.

ESP8266 - The SoC

- The ESP8266 itself, as its core, is a microcontroller with built-in WiFi in a single chip.
- This type of chip is also known as a System-on-a-chip (or SoC for short). It means that all or most components of a computer are integrated into a single chip.

25 (a)





Node MCU Specifications

- clock Speed - 80MHz
- USB connector - Micro USB connector
- Operating Voltage - 3.3V
- Input Voltage - 4.5 - 10V
- SRAM - 64KB
- Flash Memory - 4MB
- Digital I/O pins + PWM pins - 11 + 5 = 16
- Analog pins - 1
- WiFi - 802.11 b/g/n

Node MCU pin Description

Power pins - 4 power pins, 4 GND pins

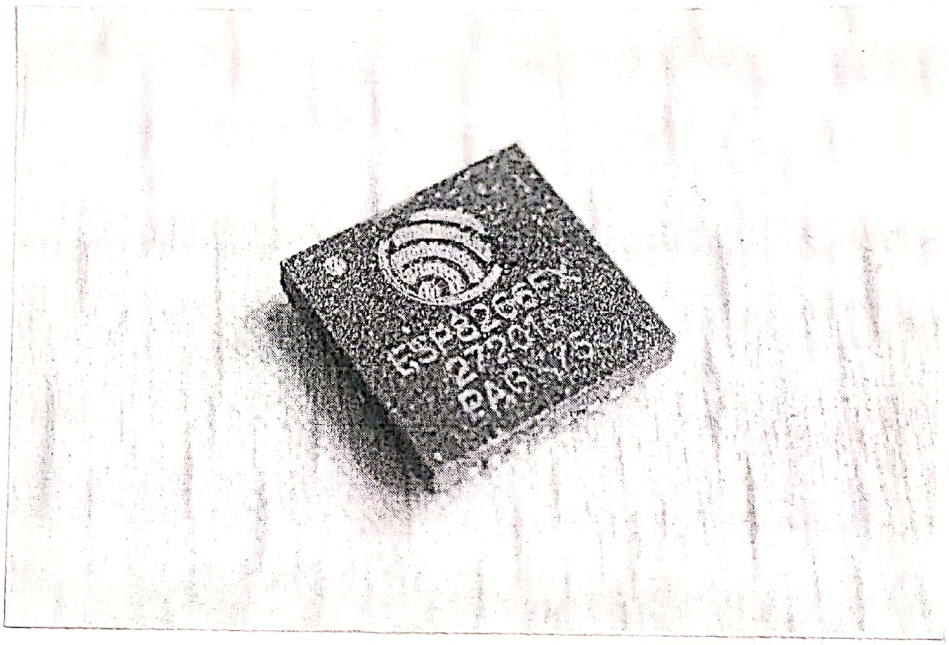
1 Vin pin - for power supply to Node MCU

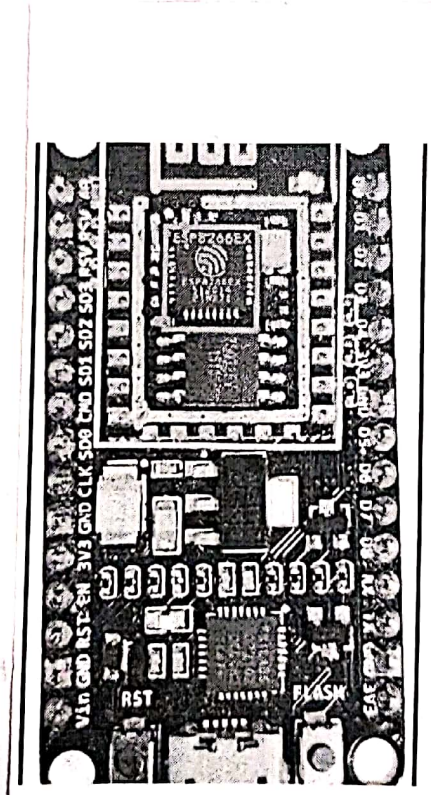
3 Vout Pins - for devices/sensors connected to Node MCU.

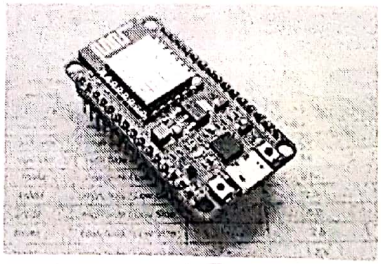
GPIO pins - 17 GPIO pins

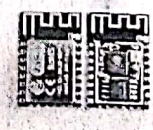
Functions such as I2C, I2S, UART, PWM, IR Remote Control, LED Light and Button etc.

- One Analog Pin (ADC pin): The Node MCU is embedded with a 10-bit precision SAR ADC.



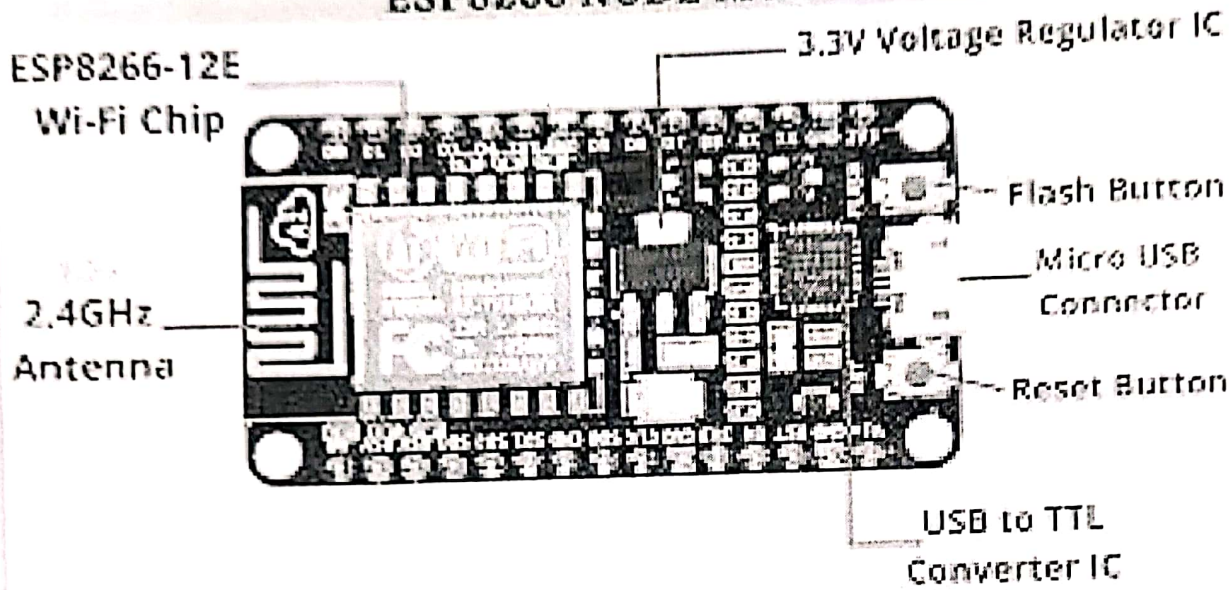






- I2C pins : used to connect I2C sensors and peripherals.
- UART pins : Node MCU / ESP8266 has 2 UART interfaces (UART0 and UART1) which provide asynchronous communication (RS232 and RS485), and can communicate at up to 4.5 Mbps. UART0 (TXD0, RXD0, RST0 & CTS0 pins) can be used for communication.
- SPI pins : Node MCU / ESP8266 features two SPIs (SPI and HSPI) in slave and master modes.
- SDIO pins : Node MCU / ESP8266 features Secure Digital Input / Output Interface (SDIO) which is used to directly interface SD cards.
- Control pins: EN, RST, WAKE pins are used to control the NodeMCU / ESP8266.
- PWM pins : are used to control the Node MCU / ESP8266.

ESP8266 NODE MCU



Configurring Arduino IDE for NodeMCU

Step 1: From File ... Select preferences

In the field "Additional Boards Manager URLS:" enter

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

Select OK button

Step 2: Selecting the NodeMCU Board Driver

From the Tools Select Board Manager

From the Boards Manager window, enter esp8266

Now select Install

Step 3: selecting Board version

From the Tools ... Select Board

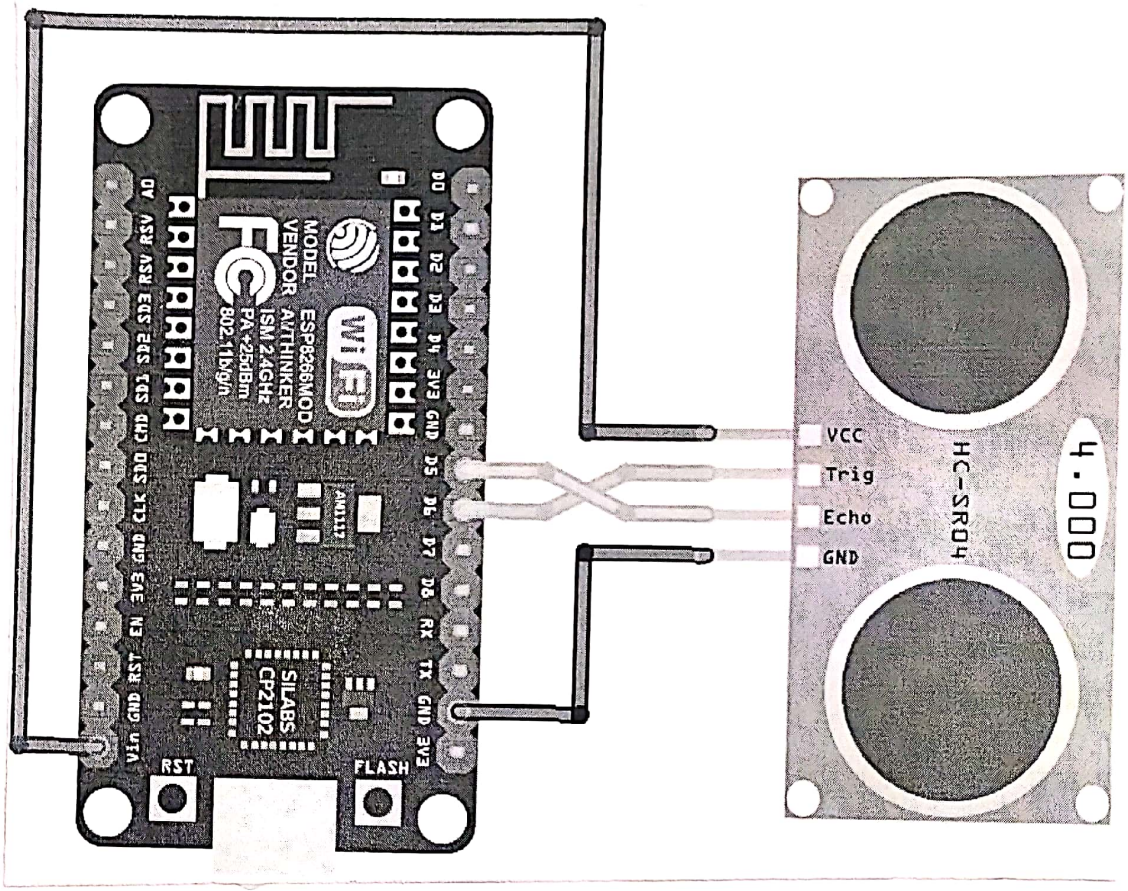
Next Select NodeMCU 1.0 (ESP-12E Module)

Step 4: Defining the Communications Port

Program : Interfacing NodeMCU with Distance - Sensor

```
int trig = 12;
int echo = 14;
int time_microsec;
int time_ms;
int dist_cm;

void setup() {
    Serial.begin(9600);
    pinMode(12, OUTPUT);
    pinMode(14, INPUT);
}
void loop() {
    digitalWrite(trig, LOW);
    delayMicroseconds(2);
    digitalWrite(trig, HIGH);
    delayMicroseconds(10);
    digitalWrite(trig, LOW);
    time_microsec = pulseIn(echo, HIGH);
    time_ms = time_microsec/1000;
    dist_cm = (34 * time_ms)/2;
```

29(a)

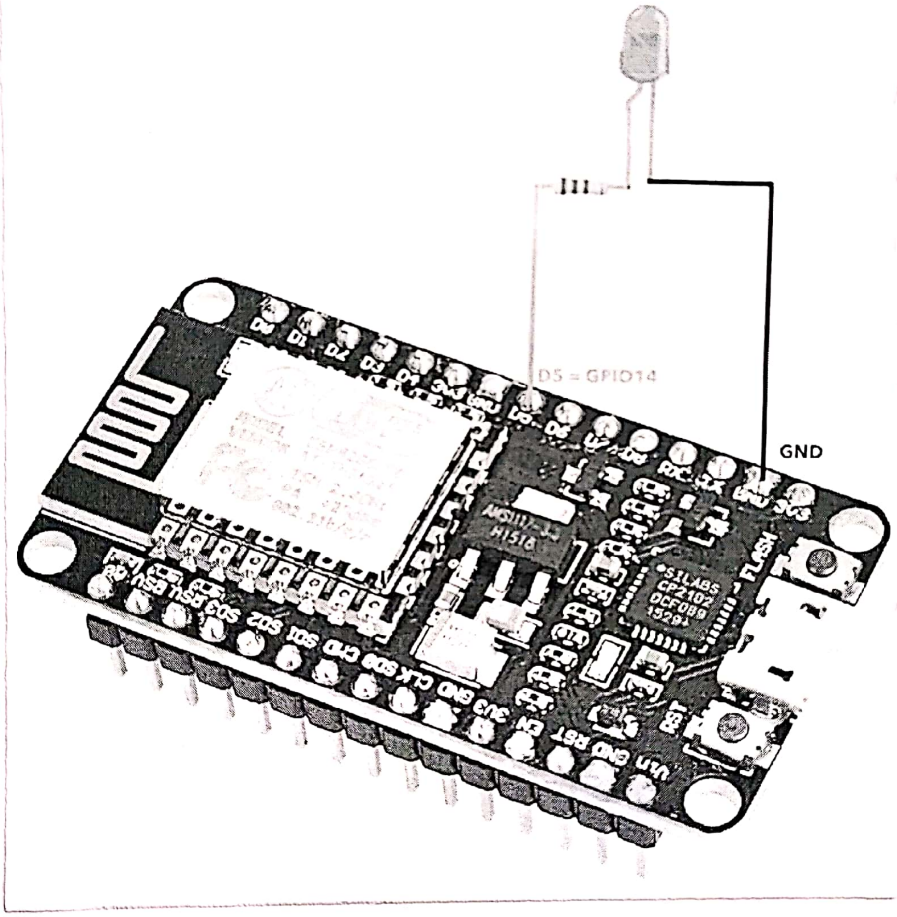

VCC
Trig
Echo
GND
HC-SR04
4.000
MODEL ESP8266MOD
VENDOR AI-THINKER
ISM 2.4GHz
PA +25dBm
802.11b/g/n
SILABS
CP2102
RST
FLASH

```
Serial.print("Time to travel: ");
Serial.println(time_ms, 1);
Serial.print("ms | ");
Serial.print("Distance: ");
Serial.print(dist_cm, 1);
Serial.print("cm");
delay(2000);
}
```

Program: Inter facing NodeMCU with LED

```
ind LEdPin = 2; //D4 Pin also refereed as pin no2 in NodeMCU
void setup() {
    pinMode(LedPin, OUTPUT);
    Serial.begin(9600);
}
void loop() {
    digitalWrite(LedPin, HIGH);
    delay(1000);
    digitalWrite(LedPin, LOW);
    delay(1000);
}
```

31(a)

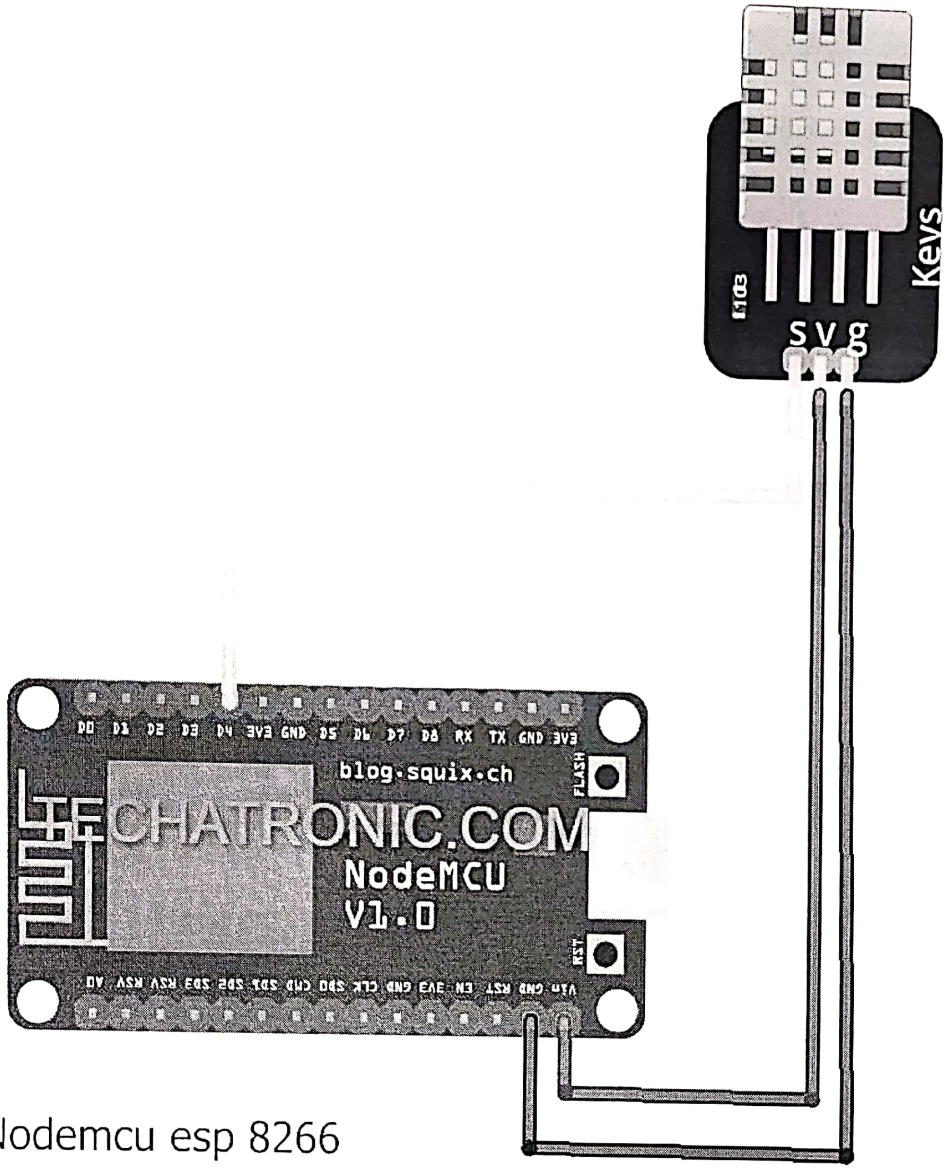


Program : Interfacing NodeMCU with DHT11

```
#include "DHT.h"
#define DHTPIN 2 // GPIO pin04
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);
void setup() {

    Serial.begin(9600);
    Serial.println(F("DHTxx Test!"));
    dht.begin();
}
void loop() {

    delay(2000);
    float h = dht.readHumidity();
    float t = dht.readTemperature();
    float f = dht.readTemperature(true);
    if (isnan(h) || isnan(t) || isnan(f)) {

        Serial.println(F("Failed to read from DHT sensor"));
        return;
    }
```

32(a)

DHT11 Temperature & Humidity Sensor



Nodemcu esp 8266

```
Serial.print(F("Humidity: "));
Serial.print(h);
Serial.print(F(" % Temperature: "));
Serial.print(t);
Serial.print(F("°C "));
Serial.print(f);
Serial.print(F(" °F "));
Serial.println(" ");
}
```

DHT 11 - Temperature and Humidity Sensor

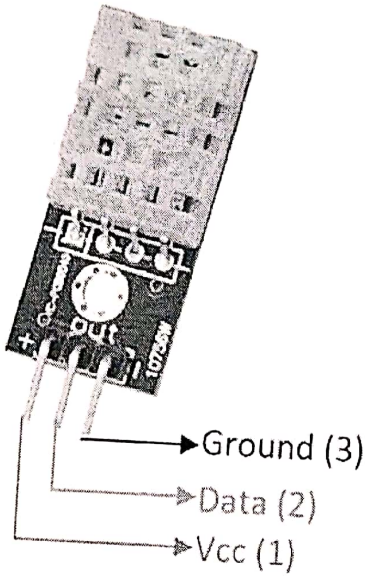
DHT stands for Digital Humidity and Temperature. The DHT sensor is a low-cost digital sensor for sensing temperature and humidity. This sensor can be easily interfaced with any micro-controller such as Arduino, Raspberry Pi to measure humidity and temperature instantaneously.

The DHT sensors are made of two parts, a capacitive humidity sensor and a thermistor. There is also a very basic chip inside that does some analog to digital conversion and spits out a digital signal with the temperature and humidity. The digital signal is fairly easy to read using any microcontroller.

DHT11 Specifications

- Ultra low cost
- 3 to 5V power and I/O
- 2.5mA max current use during conversion (while requesting data)
- Good for 20-80% humidity reading with 5% accuracy
- Good for 0-50°C temperature readings ±2°C accuracy
- No more than 1Hz sampling rate (once every second)
- Body size 15.5mm x 12mm x 5.5mm
- 4 pins with 0.1" spacing

DHT11 Pinout



www.Circuits-DIY.com

TASK-4

Raspberry Pi OS (Previously called Raspbian) is the officially supported Raspberry Pi OS.

Step 1 : Type "Download Raspberry Pi Installer" in google search

Step 2 : Click on https://www.raspberrypi.com/software/

Step 3 : From the screen... Select Download for windows

The actio initiates a download of Raspberry Pi Imager Software

Install the imager software.

Step 4 : After Download click on the imager file. You will get a pop up with options -

- Raspberry Pi Device OS Storage
- First choose - OS - select Raspberry Pi OS other
- Then select - Raspberry Pi OS Full (2.5 GB) or Raspberry Pi OS Lite 64 bit (0.5 GB)

Step 5 : Select Type of Device (Raspberry Pi 2 or 3 or 4)

Step 6 : Next select the Storage - Sd card and install the OS in the SD card

Step 7 : Next put the card in the Raspberry Pi

Step 8 : You get a pop-up "Welcome to the Raspberry Pi OS"

Step 9 : Next select - country, language

Step 10: Next - set Password

Interfacing OP: with LED

36(a)