S

Sri Indu
College of Engineering & Technology
UGC Autonomous Institution
Recognized under 2(f) & 12(B) of UGC Act 1956,
NAAC, Approved by AICTE &
Permanently Affiliated to JNTUH

Estd.2001

NAAC
NATIONAL ASSESSMENT AND ACCREDITATION COUNCIL

INTERNET OF THINGS LABORATORY (R22CSO2128)

LAB MANUAL

II Year I Semester

DEPARTMENT OF INFORMATION TECHNOLOGY

SRI INDU COLLEGE OF ENGINEERING & TECHNOLOGY

B. TECH –INFORMATION TECHNOLOGY

INSTITUTION VISION

To be a premier Institution in Engineering & Technology and Management with competency, values and social consciousness.

INSTITUTION MISSION

IM1 Provide high quality academic programs, training activities and research facilities.

IM2 Promote Continuous Industry-Institute Interaction for Employability, Entrepreneurship, Leadership and Research aptitude among stakeholders.

IM3 Contribute to the Economical and technological development of the region, state and nation.

DEPARTMENT VISION

To be a recognized knowledge centre in the field of Information Technology with self - motivated, employable engineers to society.

DEPARTMENT MISSION

The Department has following Missions:

DM1 To offer high quality student centric education in Information Technology.

DM2 To provide a conducive environment towards innovation and skills.

DM3 To involve in activities that provide social and professional solutions.

DM4 To impart training on emerging technologies namely cloud computing and IOT with involvement of stake holders.

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

PEO1: **Higher Studies:** Graduates with an ability to apply knowledge of Basic sciences and programming skills in their career and higher education.

PEO2: **Lifelong Learning:** Graduates with an ability to adopt new technologies for ever changing IT industry needs through Self-Study, Critical thinking and Problem solving skills.

PEO3: **Professional skills:** Graduates will be ready to work in projects related to complex problems involving multi-disciplinary projects with effective analytical skills.

PEO4: **Engineering Citizenship:** Graduates with an ability to communicate well and exhibitsocial, technical and ethical responsibility in process or product.

PROGRAM OUTCOMES (POs) & PROGRAM SPECIFIC OUTCOMES (PSOs)

PO	Description
PO 1	**Engineering Knowledge:** Apply knowledge of mathematics, natural science, computing, engineering fundamentals and an engineering specialization as specified in WK1 to WK4 respectively to develop to the solution of complex engineering problems.
PO 2	**Problem Analysis:** Identify, formulate, review research literature and analyze complex engineering problems reaching substantiated conclusions with consideration for sustainable development. (WK1 to WK4)
PO 3	**Design/Development of Solutions:** Design creative solutions for complex engineering problems and design/develop systems/components/processes to meet identified needs with consideration for the public health and safety, whole-life cost, net zero carbon, culture, society and environment as required. (WK5)
PO 4	**Conduct Investigations of Complex Problems:** Conduct investigations of complex engineering problems using research-based knowledge including design of experiments, modelling, analysis & interpretation of data to provide valid conclusions. (WK8).
PO 5	**Engineering Tool Usage:** Create, select and apply appropriate techniques, resources and modern engineering & IT tools, including prediction and modelling recognizing their limitations to solve complex engineering problems. (WK2 and WK6)
PO 6	**The Engineer and The World:** Analyze and evaluate societal and environmental aspects while solving complex engineering problems for its impact on sustainability with reference to economy, health, safety, legal framework, culture and environment. (WK1, WK5, and WK7).
PO 7	**Ethics:** Apply ethical principles and commit to professional ethics, human values, diversity and inclusion; adhere to national & international laws. (WK9)
PO 8	**Individual and Collaborative Team work:** Function effectively as an individual, and as a member or leader in diverse/multi-disciplinary teams.
PO 10	**Project Management and Finance:** Apply knowledge and understanding of engineering management principles and economic decision-making and apply these to one's own work, as a member and leader in a team, and to manage projects and in multidisciplinary environments.
PO 11	**Life-Long Learning:** Recognize the need for, and have the preparation and ability for i) independent and life-long learning ii) adaptability to new and emerging technologies and iii) critical thinking in the broadest context of technological change. (WK8)
Program Specific Outcomes	
PSO 1	**Software Development:** To apply the knowledge of Software Engineering, Data Communication, Web Technology and Operating Systems for building IOT and Cloud Computing applications.
PSO 2	**Industrial Skills Ability:** Design, develop and test software systems for world-wide network of computers toprovide solutions to real world problems.
PSO 3	**Project implementation:** Analyze and recommend the appropriate IT Infrastructure required for the implementationof a project.

GENERAL LABORATORY INSTRUCTIONS

1. Students are advised to come to the laboratory at least 5 minutes before (to the starting time),those who come after 5 minutes will not be allowed into the lab.
2. Plan your task properly much before to the commencement, come prepared to the lab with thesynopsis / program / experiment details.
3. Student should enter into the laboratory with:
    a) Laboratory observation notes with all the details (Problem statement, Aim, Algorithm,Procedure, Program, Expected Output, etc.,) filled in for the lab session.
    b) Laboratory Record updated up to the last session experiments and other utensils (if any)needed in the lab.
    c) Proper Dress code and Identity card.
4. Sign in the laboratory login register, write the TIME-IN, and occupy the computer system allotted to you by the faculty.
5. Execute your task in the laboratory, and record the results / output in the lab observation notebook, and get certified by the concerned faculty.
6. All the students should be polite and cooperative with the laboratory staff, must maintain thediscipline and decency in the laboratory.
7. Computer labs are established with sophisticated and high end branded systems, which should be utilized properly.
8. Students / Faculty must keep their mobile phones in SWITCHED OFF mode during the lab sessions. Misuse of the equipment, misbehaviors with the staff and systems etc., will attract severe punishment.
9. Students must take the permission of the faculty in case of any urgency to go out ; if anybody found loitering outside the lab / class without permission during working hours will be treated seriously and punished appropriately.
10. Students should LOG OFF/ SHUT DOWN the computer system before he/she leaves the labafter completing the task (experiment) in all aspects. He/she must ensure the system / seat is kept properly.

Head of the Department **Principal**

SRI INDU COLLEGE OF ENGINEERING & TECHNOLOGY

(An Autonomous Institution under UGC, New Delhi)

B.Tech. - II Year – I Semester

L	T	P	C
0	0	3	1.5

(R22CSO2128) INTERNET OF THINGS LAB

Course Objectives:

- To introduce the raspberry PI platform, that is widely used in IoT applications
- To introduce the implementation of distance sensor on IoT devices

Course Outcomes:

- Ability to introduce the concept of M2M (machine to machine) with necessary protocols andget awareness in implementation of distance sensor
- Get the skill to program using python scripting language which is used in many IoT devices

List of Experiments

1. Using raspberry pi
   a. Calculate the distance using a distance sensor.
   b. Basic LED functionality.
2. Using Arduino
   a. Calculate the distance using a distance sensor.
   b. Basic LED functionality.
   c. Calculate temperature using a temperature sensor.
3. Using Node MCU
   a. Calculate the distance using a distance sensor.
   b. Basic LED functionality.
   c. Calculate temperature using a temperature sensor.
4. Installing OS on Raspberry Pi
   a) Installation using PiImager
   b) Installation using image file
      - Downloading an Image
      - Writing the image to an SD card
        - using Linux
        - using Windows
      - Booting up Follow the instructions given in the URL https://www.raspberrypi.com/documentation/computers/getting-started.html
5. Accessing GPIO pins using Python
   a) Installing GPIO Zero library.
      First, update your repositories list:
      sudo apt update
      Then install the package for Python 3:
      sudo apt install python3-gpiozero
   b) Blinking an LED connected to one of the GPIO pin
   c) Adjusting the brightness of an LED Adjust the brightness of an LED (0 to 100, where 100 means maximum brightness) using the in-built PWM wavelength.
6. Collecting Sensor Data
   a) DHT Sensor interface
      - Connect the terminals of DHT GPIO pins of Raspberry Pi.
      - Import the DHT library using import Adafruit_DHT
      - Read sensor data and display it on screen.

INDEX

Experiment 1

1A. Installing OS on Raspberry Pi

Aim: To install an operating system on Raspberry Pi

To use your Raspberry Pi, you'll need an operating system. By default, Raspberry Pis check for an operating system on any SD card inserted in the SD card slot.

Depending on your Raspberry Pi model, you can also boot an operating system from other storage devices, including USB drives, storage connected via a HAT, and network storage.

To install an operating system on a storage device for your Raspberry Pi, you'll need:

- a computer you can use to image the storage device into a boot device
- a way to plug your storage device into that computer

Most Raspberry Pi users choose microSD cards as their boot device.

We recommend installing an operating system using Raspberry Pi Imager.

Raspberry Pi Imager is a tool that helps you download and write images on macOS, Windows, and Linux. Imager includes many popular operating system images for Raspberry Pi. Imager also supports loading images downloaded directly from Raspberry Pi or third-party vendors such as Ubuntu. You can use Imager to preconfigure credentials and remote access settings for your Raspberry Pi.

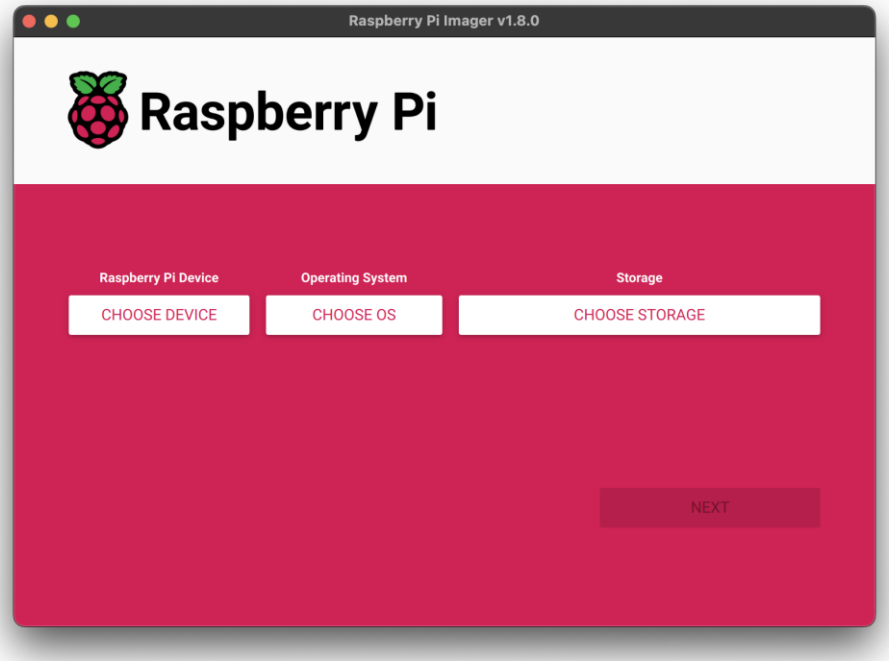
Imager supports images packaged in the .img format as well as container formats like .zip.

Install using Imager

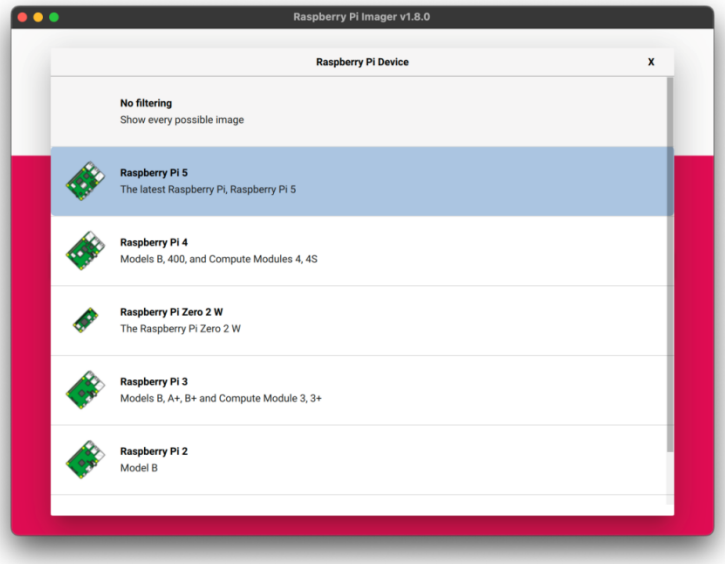
You can install Imager in the following ways:

- Download the latest version from raspberrypi.com/software and run the installer.
- Install it from a terminal using your package manager, e.g. sudo apt install rpi-imager.

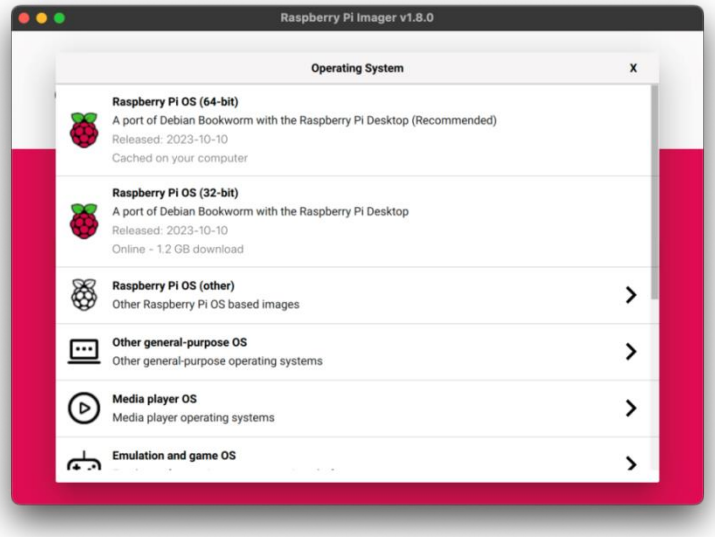
Once you've installed Imager, launch the application by clicking the Raspberry Pi Imager icon or running rpi-imager.



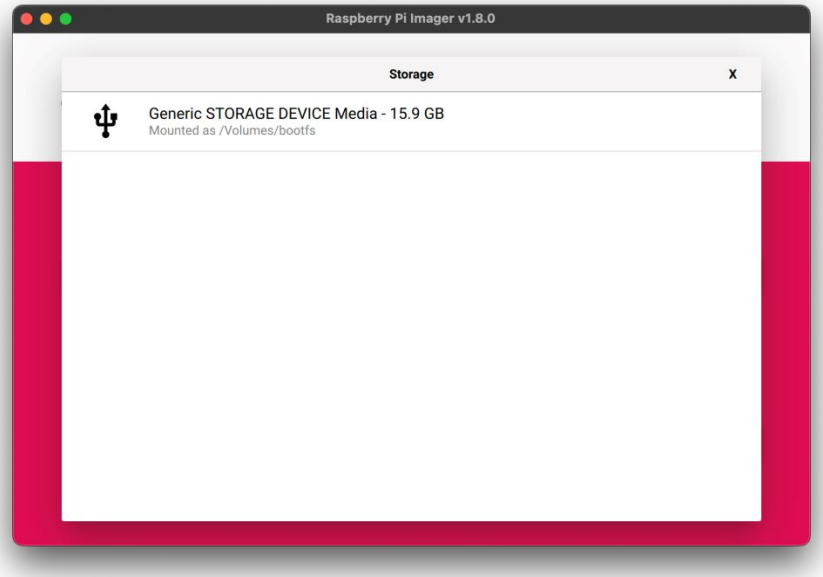
Click **Choose device** and select your Raspberry Pi model from the list.



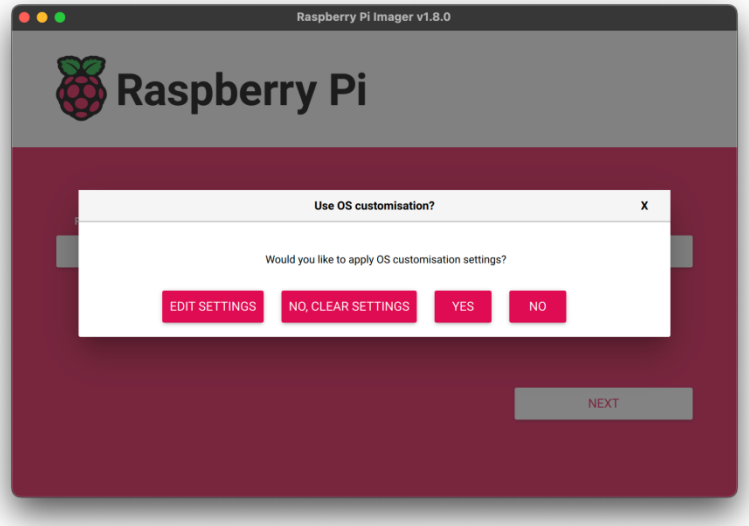
Next, click **Choose OS** and select an operating system to install. Imager always shows the recommended version of Raspberry Pi OS for your model at the top of the list.



Connect your preferred storage device to your computer. For example, plug a microSD card in using an external or built-in SD card reader. Then, click **Choose storage** and select your storage device



Next, click **Next**.



In a popup, Imager will ask you to apply OS customisation. We strongly recommend configuring your Raspberry Pi via the OS customisation settings. Click the **Edit Settings** button to open OS customisation.

If you don't configure your Raspberry Pi via OS customisation settings, Raspberry Pi OS will ask you for the same information at first boot during the configuration wizard. You can click the **No** button to skip OS customisation.

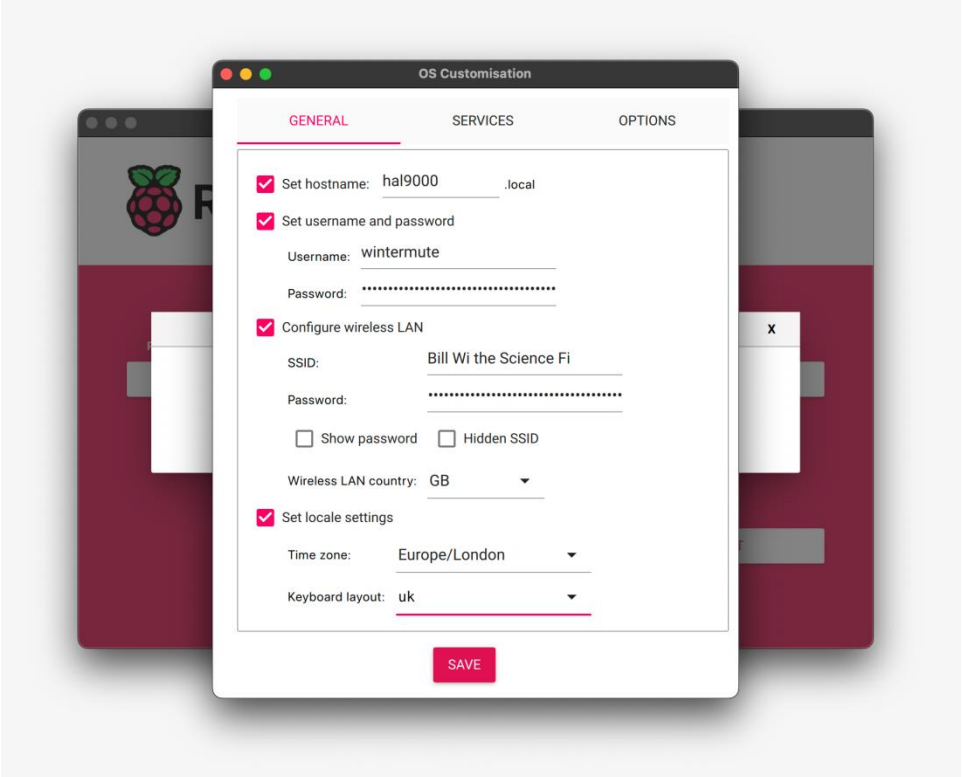
OS customisation

The OS customisation menu lets you set up your Raspberry Pi before first boot. You can preconfigure:

- a username and password
- Wi-Fi credentials
- the device hostname
- the time zone
- your keyboard layout
- remote connectivity

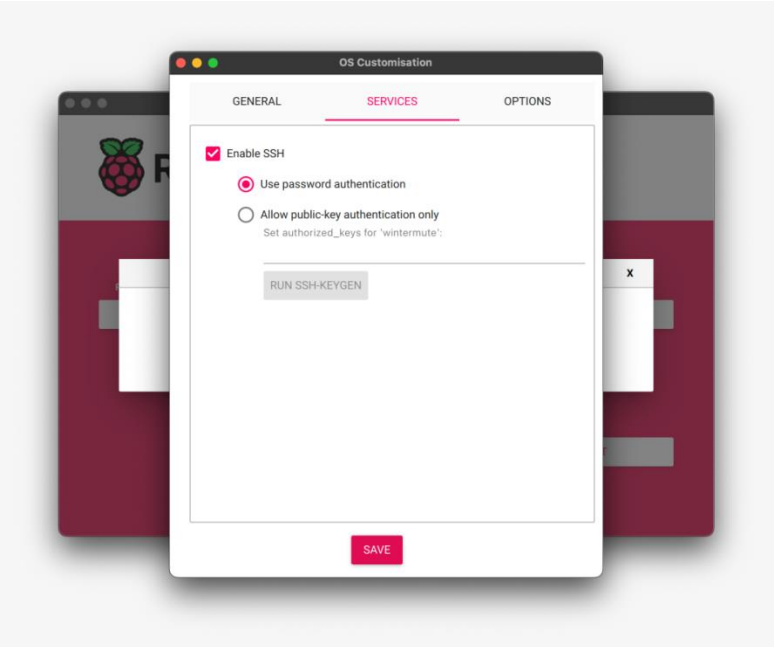
When you first open the OS customisation menu, you might see a prompt asking for permission to load Wi-Fi credentials from your host computer. If you respond "yes", Imager will prefill Wi-Fi credentials from the network you're currently connected to. If you respond "no", you can enter Wi-Fi credentials manually.

The **hostname** option defines the hostname your Raspberry Pi broadcasts to the network using mDNS. When you connect your Raspberry Pi to your network, other devices on the network can communicate with your computer using <hostname>.local or <hostname>.lan.
The **username and password** option defines the username and password of the admin user account on your Raspberry Pi.
The **wireless LAN** option allows you to enter an SSID (name) and password for your wireless network. If your network does not broadcast an SSID publicly, you should enable the "Hidden SSID" setting. By default, Imager uses the country you're currently in as the "Wireless LAN country". This setting controls the Wi-Fi broadcast frequencies used by your Raspberry Pi. Enter credentials for the wireless LAN option if you plan to run a headless Raspberry Pi.
The **locale settings** option allows you to define the time zone and default keyboard layout for your Pi.

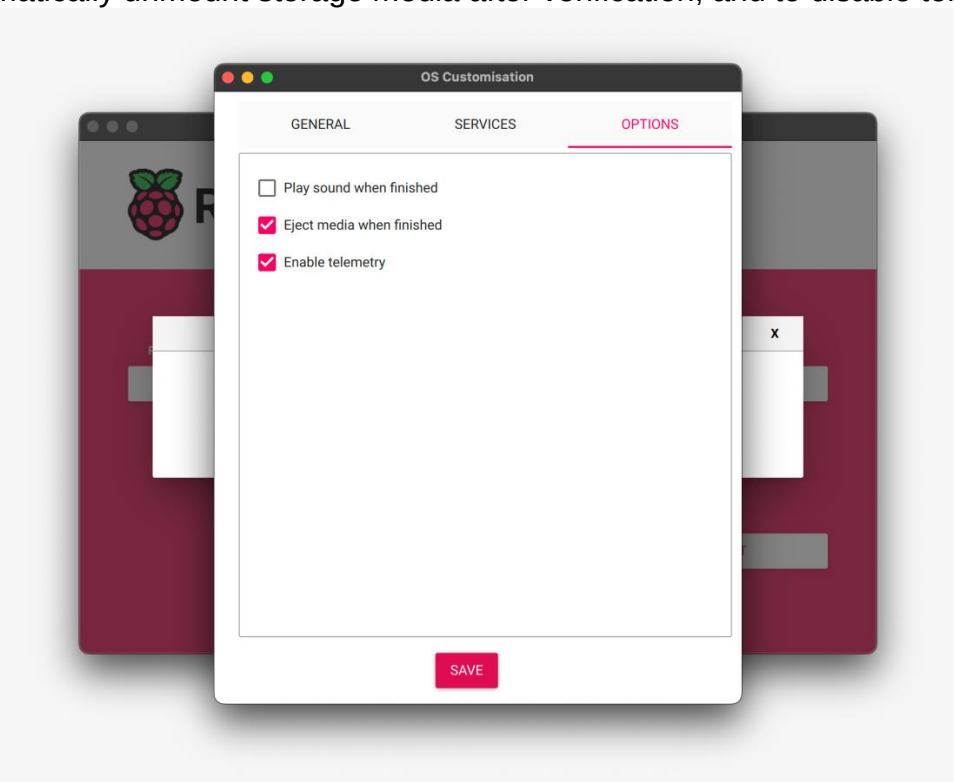


The **Services** tab includes settings to help you connect to your Raspberry Pi remotely.
If you plan to use your Raspberry Pi remotely over your network, check the box next to **Enable SSH**. You should enable this option if you plan to run a headless Raspberry Pi.

- Choose the **password authentication** option to SSH into your Raspberry Pi over the network using the username and password you provided in the general tab of OS customisation.
- Choose **Allow public-key authentication only** to preconfigure your Raspberry Pi for passwordless public-key SSH authentication using a private key from the computer you're currently using. If already have an RSA key in your SSH configuration, Imager uses that public key. If you don't, you can click **Run SSH-keygen** to generate a public/private key pair. Imager will use the newly-generated public key.

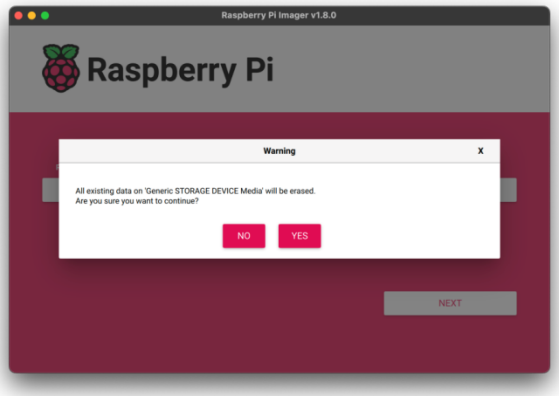


OS customisation also includes an **Options** menu that allows you to configure the behaviour of Imager during a write. These options allow you to play a noise when Imager finishes verifying an image, to automatically unmount storage media after verification, and to disable telemetry.



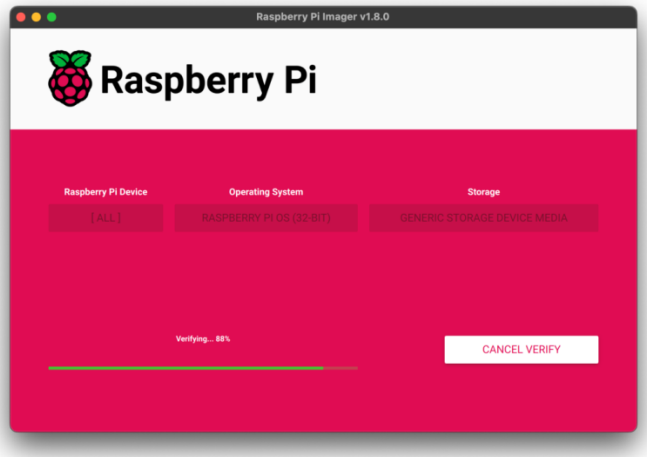
Write

When you've finished entering OS customisation settings, click **Save** to save your customisation. Then, click **Yes** to apply OS customisation settings when you write the image to the storage device.
Finally, respond **Yes** to the "Are you sure you want to continue?" popup to begin writing data to the storage device.



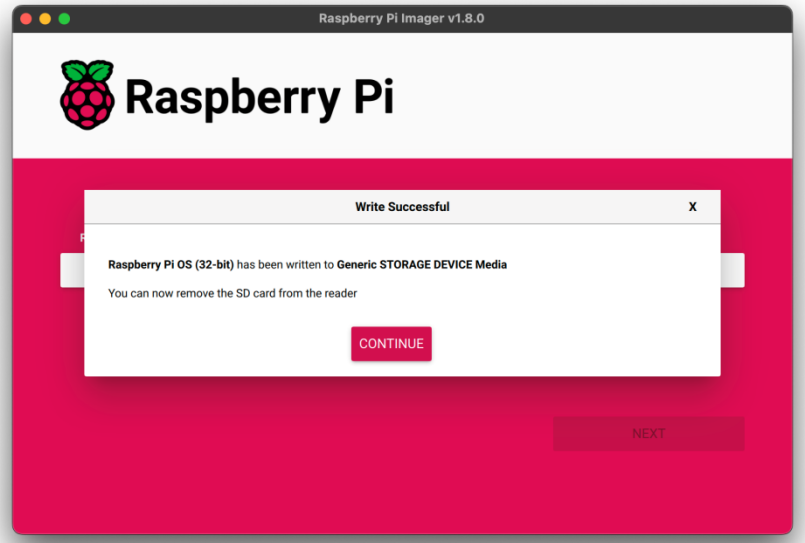
If you see an admin prompt asking for permissions to read and write to your storage medium, grant Imager the permissions to proceed.

Grab a cup of coffee or go for a walk. This could take a few minutes.



If you want to live especially dangerously, you can click **cancel verify** to skip the verification process.

When you see the "Write Successful" popup, your image has been completely written and verified. You're now ready to boot a Raspberry Pi from the storage device!



Next, proceed to the first boot configuration instructions to get your Raspberry Pi up and running.

Install over the network

Network Install enables a Raspberry Pi to install an operating system on a storage device using a version of Raspberry Pi Imager downloaded over the network. With Network Install, you can get an operating system installed on your Raspberry Pi with no separate SD card reader and no computer other than your Raspberry Pi. You can run Network Install on any compatible storage device, including SD cards and USB storage.

Network Install only runs on Raspberry Pi 4, 400, and 5. If your Raspberry Pi runs an older bootloader, you may need to update the bootloader to use Network Install.

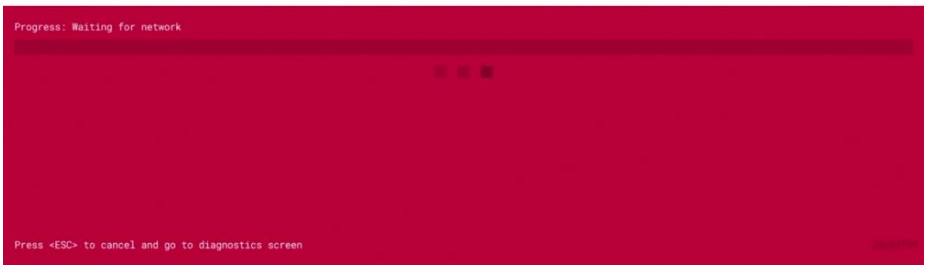
Network Install requires the following:

- a compatible Raspberry Pi model running firmware that supports Network Install
- a monitor
- a keyboard
- a wired internet connection

To launch Network Install, power on your Raspberry Pi *while pressing and holding the **SHIFT** key* in the following configuration:

- no bootable storage device
- attached keyboard
- attached compatible storage device, such as an SD card or USB storage

If you haven't already connected your Raspberry Pi to the internet, connect it with an Ethernet cable.



Once you're connected to the internet, your Raspberry Pi will download Raspberry Pi installer. If the download fails, you can repeat the process to try again.

Once you finish downloading Raspberry Pi Installer, your Raspberry Pi will automatically start Raspberry Pi Imager. For more information about running Raspberry Pi Imager, see install an operating system.

For more information about Network Install configuration, see HTTP boot.

Set up your Raspberry Pi

After installing an operating system image, connect your storage device to your Raspberry Pi.

First, unplug your Raspberry Pi's power supply to ensure that the Raspberry Pi is powered down while you connect peripherals. If you installed the operating system on a microSD card, you can plug it into your Raspberry Pi's card slot now. If you installed the operating system on any other storage device, you can connect it to your Raspberry Pi now.

Then, plug in any other peripherals, such as your mouse, keyboard, and monitor.

Finally, connect the power supply to your Raspberry Pi. You should see the status LED light up when your Pi powers on. If your Pi is connected to a display, you should see the boot screen within minutes.

Configuration on first boot

If you used OS customisation in Imager to preconfigure your Raspberry Pi, **congratulations!** Your device is ready to use. Proceed to next steps to learn how you can put your Raspberry Pi to good use.

If your Raspberry Pi does not boot within 5 minutes, check the status LED. If it's flashing, see the LED warning flash codes for more information. If your Pi refuses to boot, try the following mitigation steps:

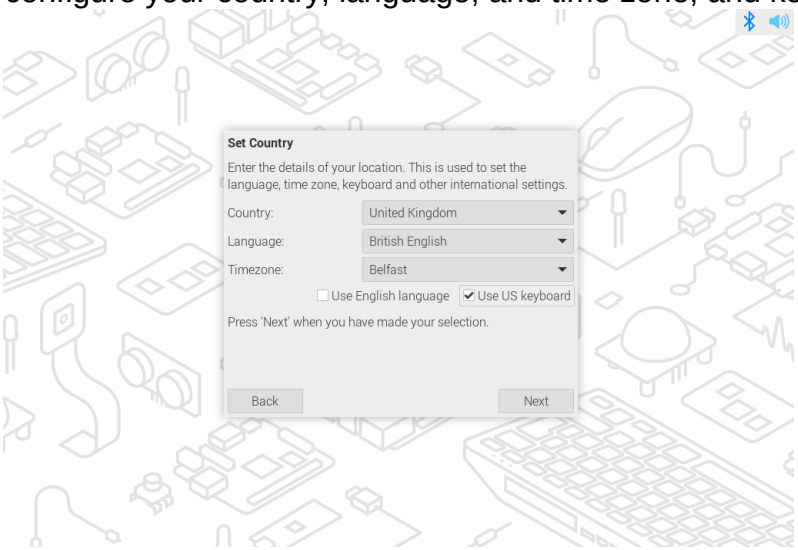
- if you used a boot device other than an SD card, try booting from an SD card
- re-image your SD card; be sure to complete the entire verify step in Imager
- update the bootloader on your Raspberry Pi, then re-image your SD card

Bluetooth

If you’re using a Bluetooth keyboard or mouse, this step will walk you through device pairing. Your Raspberry Pi will scan for pairable devices and connect to the first device it finds for each item.
This process works with built-in or external USB Bluetooth adapters. If you use a USB adapter, plug it in before booting your Raspberry Pi.

Locale

This page helps you configure your country, language, and time zone, and keyboard layout.

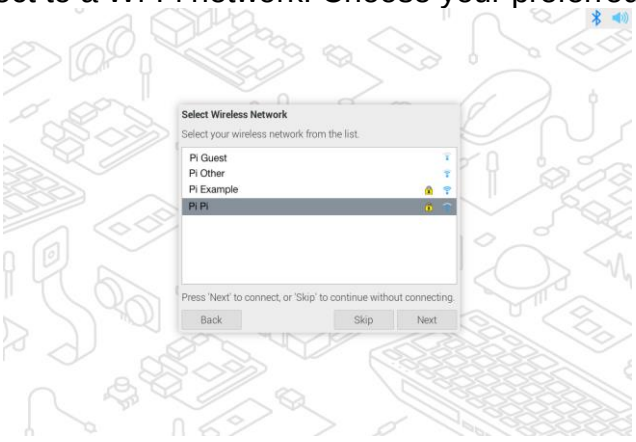


User

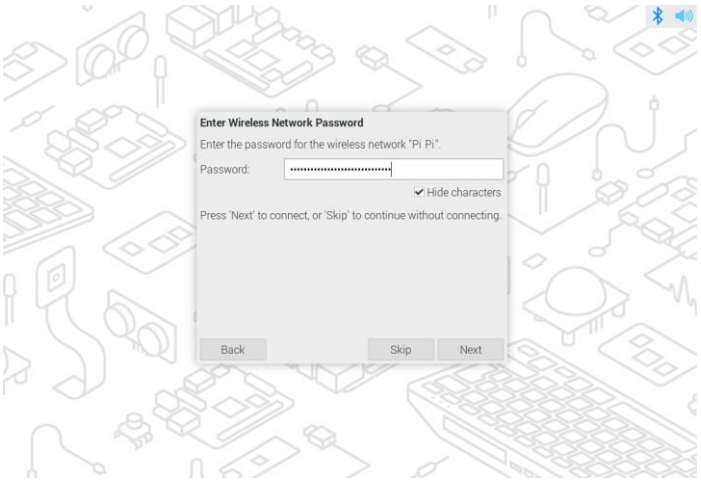
This page helps you configure the username and password for the default user account.
By default, older versions of Raspberry Pi OS set the username to "pi". If you use the username "pi", avoid the old default password of "raspberry" to keep your Raspberry Pi secure.

Wi-Fi

This page helps you connect to a Wi-Fi network. Choose your preferred network from the list.

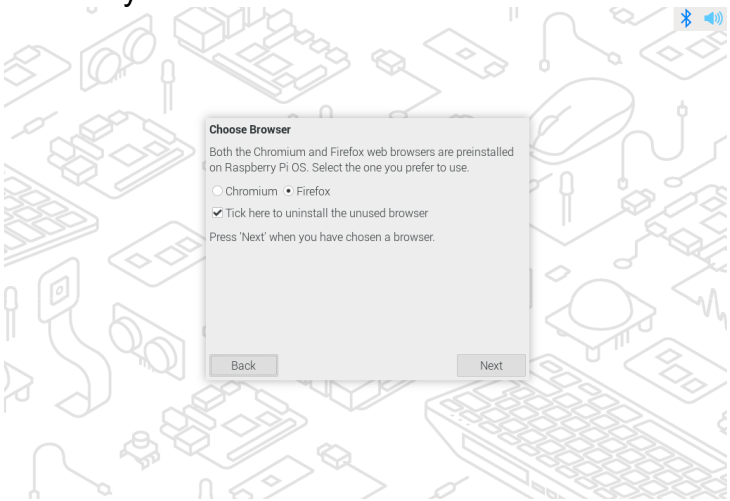


If your network requires a password, you can enter it here.



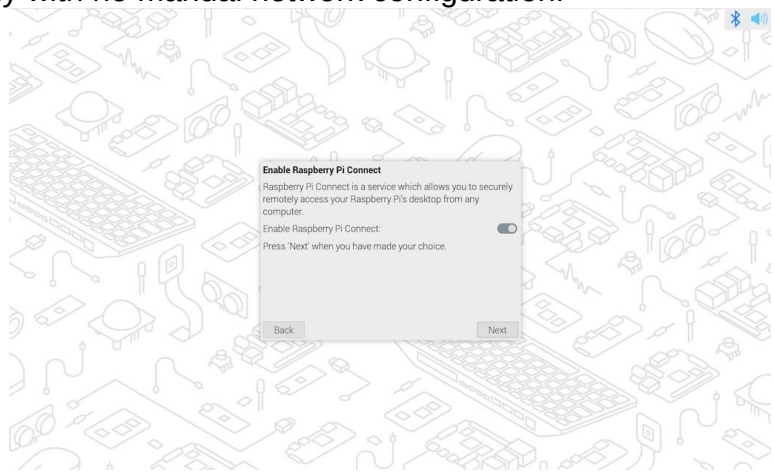
Browser

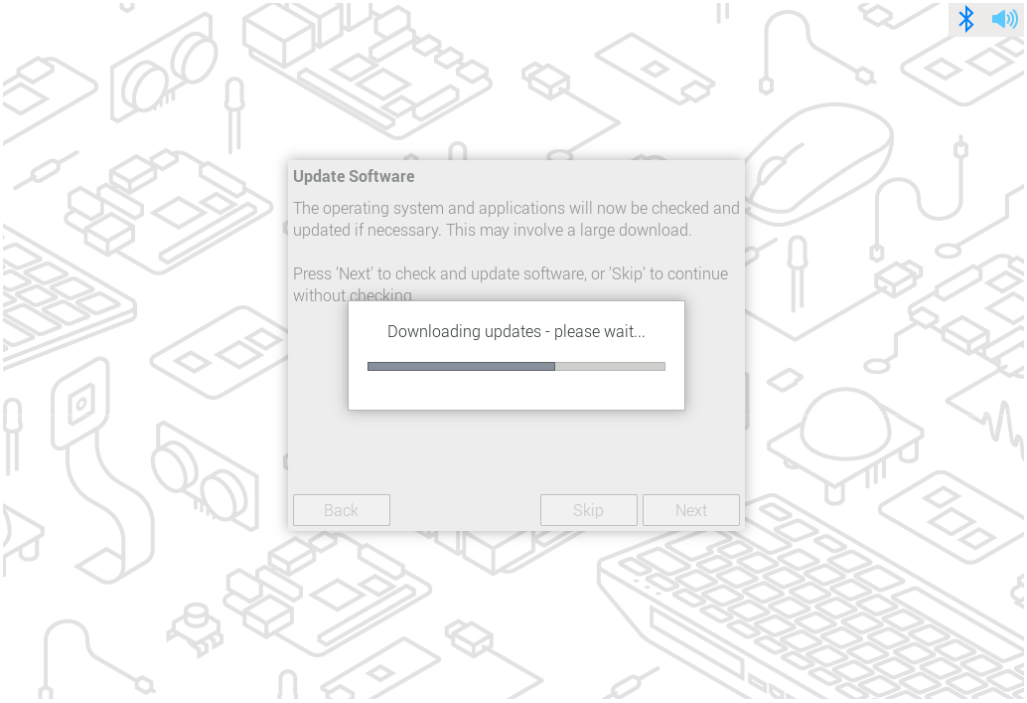
This page lets you select Firefox or Chromium as your default internet browser. You can optionally uninstall the browser you don't set as default.



Raspberry Pi Connect

This page lets you enable Raspberry Pi Connect, which provides the ability to access your Raspberry Pi remotely with no manual network configuration.

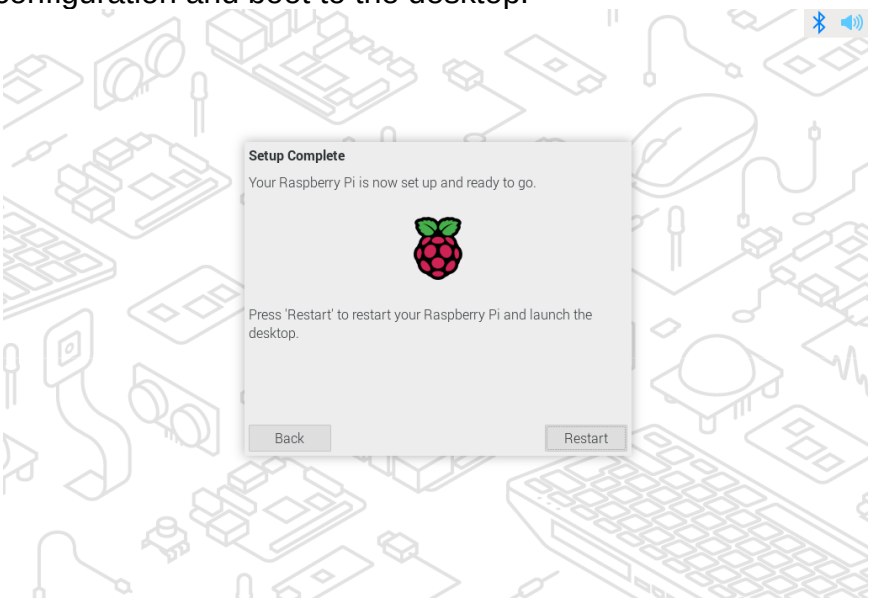




When you see a popup indicating that your system is up to date, click **OK** to proceed to the next step.

Finish

At the end of the configuration wizard, click **Restart** to reboot your Raspberry Pi. Your Raspberry Pi will apply your configuration and boot to the desktop.



* Raspberry Pi :

To get started with your Raspberry Pi, you'll need the following:

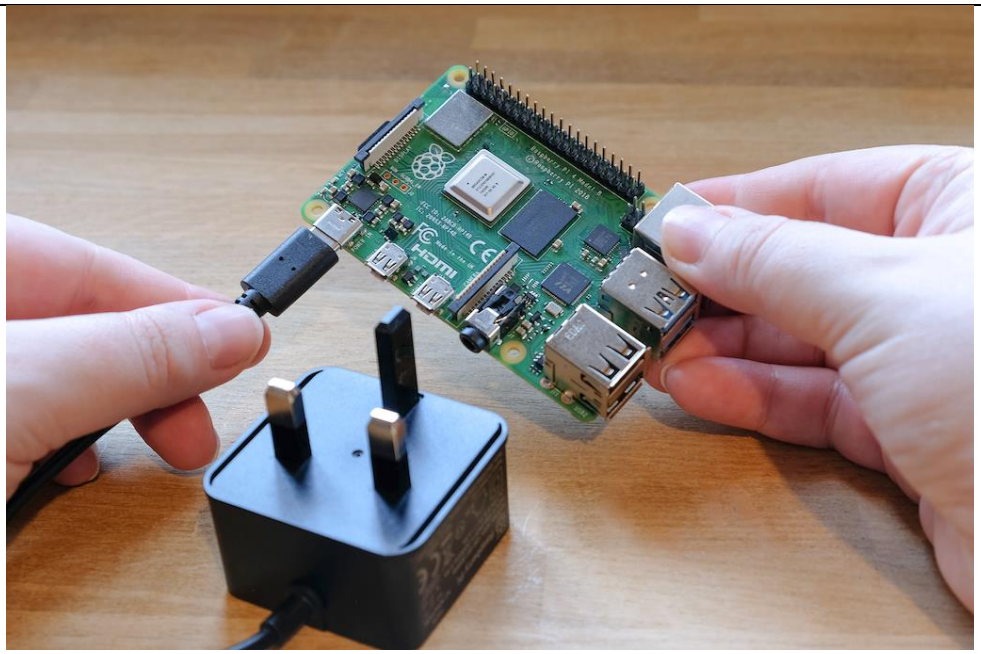
- a power supply
- boot media (e.g. a microSD card with ample storage and speed)

You can set up your Raspberry Pi as an interactive computer with a desktop, or as a *headless* computer accessible only over the network. To set your Raspberry Pi up headless, you don't need any additional peripherals: you can preconfigure a hostname, user account, network connection, and SSH when you install an operating system. If you want to use your Raspberry Pi directly, you'll need the following additional accessories:

- a display
- a cable to connect your Raspberry Pi to your display
- a keyboard

- a mouse
- **Power supply**
- The following table shows the USB-PD power mode required to power various Raspberry Pi models. You can use any high-quality power supply that provides the correct power mode.

Model	**Recommended power supply (voltage/current)**	**Raspberry Pi power supply**
Raspberry Pi 5	5V/5A, 5V/3A limits peripherals to 600mA	27W USB-C power supply
Raspberry Pi 4 Model B	5V/3A	15W USB-C power supply
Raspberry Pi 3 (all models)	5V/2.5A	12.5W Micro USB power supply
Raspberry Pi 2 (all models)	5V/2.5A	12.5W Micro USB power supply
Raspberry Pi 1 (all models)	5V/2.5A	12.5W Micro USB power supply
Raspberry Pi Zero (all models)	5V/2.5A	12.5W Micro USB power supply

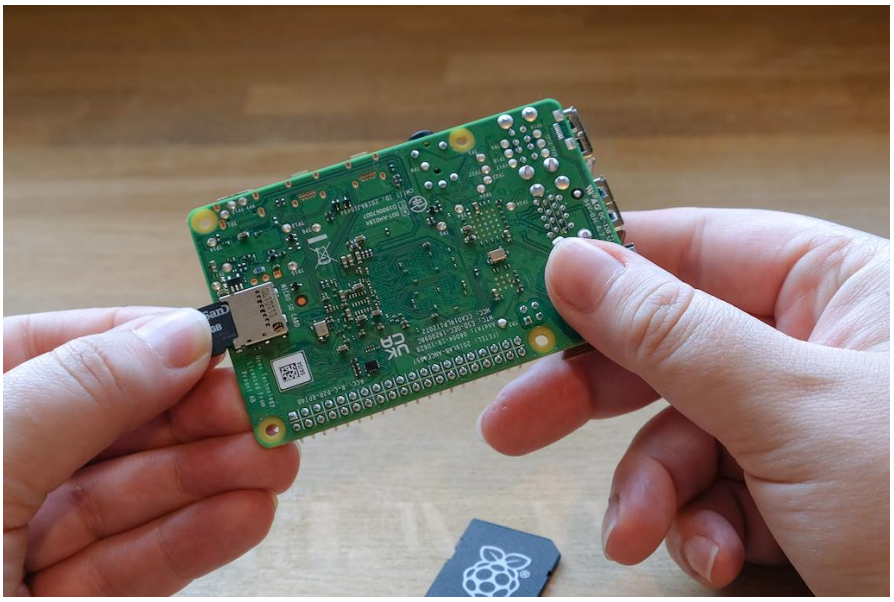


- Plug your power supply into the port marked "POWER IN", "PWR IN", or "PWR". Some Raspberry Pi models, such as the Zero series, have output USB ports with the same form factor as the power port. Be sure to use the correct port on your Raspberry Pi!

Boot Media

Raspberry Pi models lack onboard storage, so you have to supply it. You can boot your Raspberry Pi from an operating system image installed on any supported media: microSD cards are used commonly, but USB storage, network storage, and storage connected via a PCIe HAT are also available. However, only recent Raspberry Pi models support all of these media types.

All Raspberry Pi consumer models since the Raspberry Pi 1 Model A+ feature a microSD slot. Your Raspberry Pi automatically boots from the microSD slot when the slot contains a card.



Recommended SD cards

We recommend using an SD card with at least 32GB of storage for Raspberry Pi OS installations. For Raspberry Pi OS Lite, we recommend at least 16GB. You can use any SD card with a capacity of less than 2TB. Capacities above 2TB are currently not supported due to limitations in the MBR. As with any other boot media, you'll see improved performance on SD cards with faster read and write speeds.

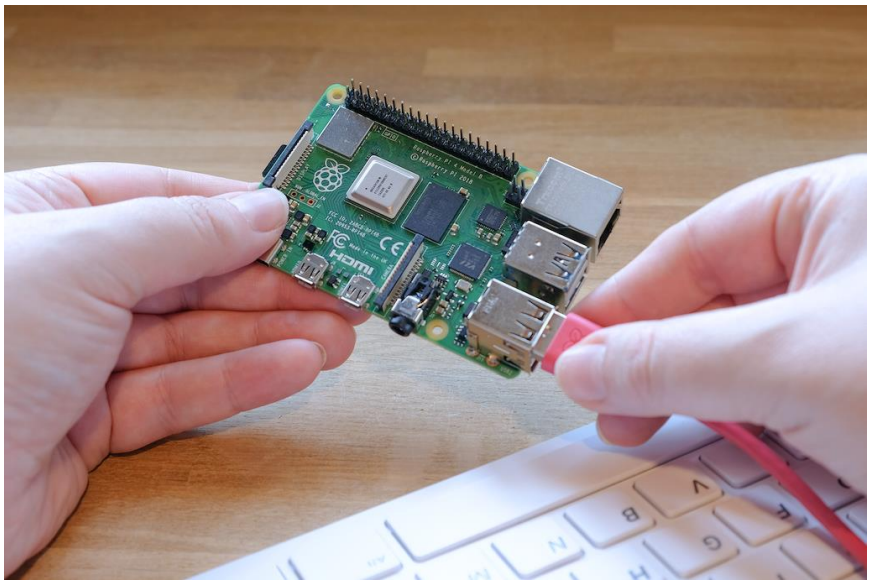
Because of a hardware limitation, the following devices will only boot from a boot partition of 256GB or less:

- Raspberry Pi Zero
- Raspberry Pi 1
- early Raspberry Pi 2 models with the BCM2836 SoC

Other operating systems have different requirements. Check the documentation for your operating system for capacity requirements.

Keyboard

You can use any of the USB ports on your Raspberry Pi to connect a wired keyboard or USB Bluetooth receiver.



Mouse

You can use any of the USB ports on your Raspberry Pi to connect a wired mouse or USB Bluetooth receiver.

Display

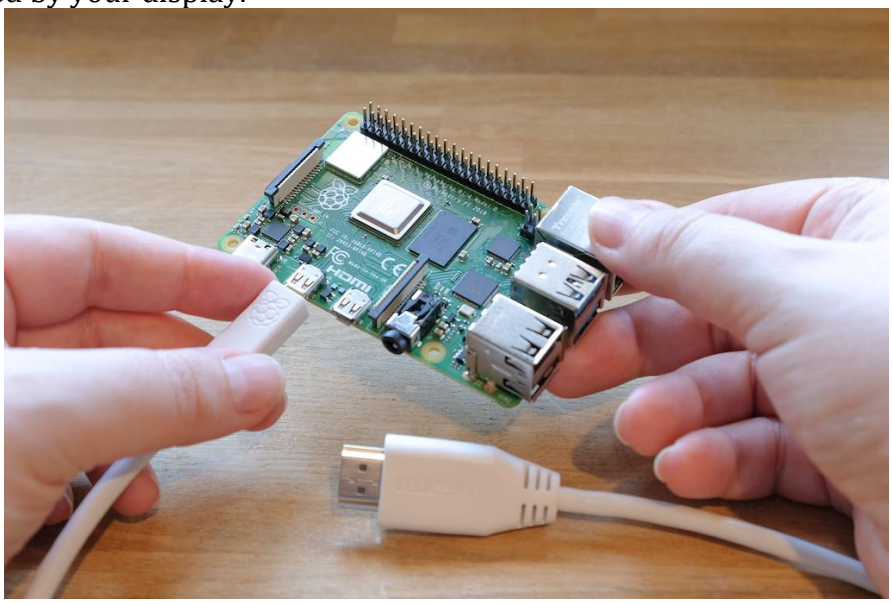
Raspberry Pi models have the following display connectivity:

Model	**Display outputs**
Raspberry Pi 5	2× micro HDMI
Raspberry Pi 4 (all models)	2× micro HDMI, audio and composite out via 3.5mm TRRS jack
Raspberry Pi 3 (all models)	HDMI, audio and composite out via 3.5mm TRRS jack
Raspberry Pi 2 (all models)	HDMI, audio and composite out via 3.5mm TRRS jack
Raspberry Pi 1 Model B+	HDMI, audio and composite out via 3.5mm TRRS jack
Raspberry Pi 1 Model A+	HDMI, audio and composite out via 3.5mm TRRS jack
Raspberry Pi Zero (all models)	mini HDMI

Note No Raspberry Pi models support video over USB-C (DisplayPort alt mode).

If your Raspberry Pi has more than one HDMI port, plug your primary monitor into the port marked HDMI0.

Most displays don't have micro or mini HDMI ports. However, you can use a micro-HDMI-to-HDMI cable or mini-HDMI-to-HDMI cable to connect those ports on your Raspberry Pi to any HDMI display. For displays that don't support HDMI, consider an adapter that translates display output from HDMI to a port supported by your display.



Audio

All Raspberry Pi models with HDMI, micro HDMI, or mini HDMI support audio output over HDMI. All Raspberry Pi models support audio over USB. All Raspberry Pi models equipped with Bluetooth

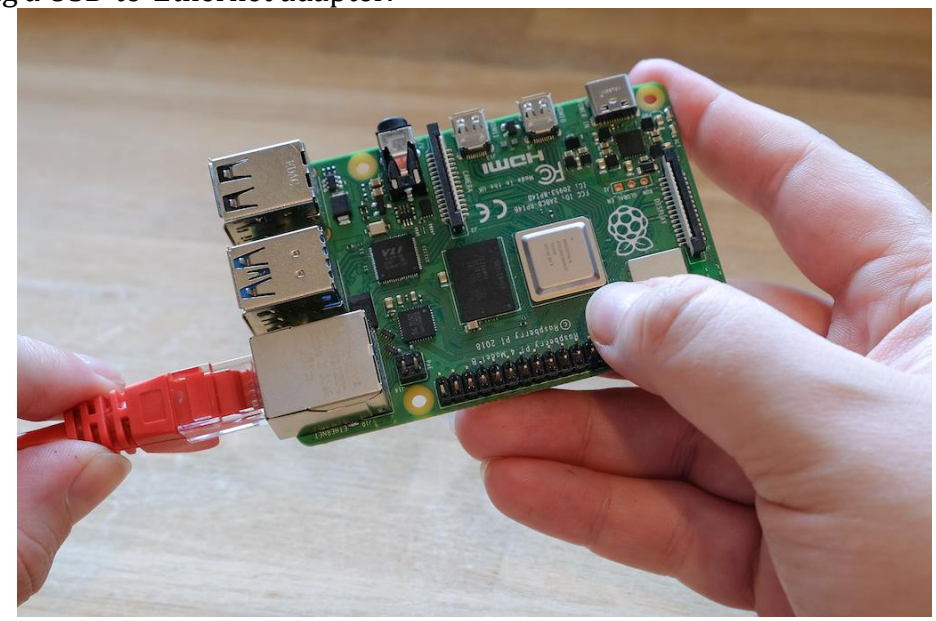
support Bluetooth audio. All variants of the Raspberry Pi 1, 2, 3, and 4 include a 3.5mm auxiliary TRRS jack, which may require amplification for sufficient output volume.

Networking

The following Raspberry Pi models come with Wi-Fi and Bluetooth connectivity:

- Raspberry Pi 5
- Raspberry Pi 4
- Raspberry Pi 3B+
- Raspberry Pi 3
- Raspberry Pi Zero W
- Rsapberry Pi Zero 2 W

The "Model B" suffix indicates variants with an Ethernet port; "Model A" indicates no Ethernet port. If your Raspberry Pi doesn't have an Ethernet port, you can still connect to a wired internet connection using a USB-to-Ethernet adapter.



Experiment 1

1B. Installing Arduino IDE Software

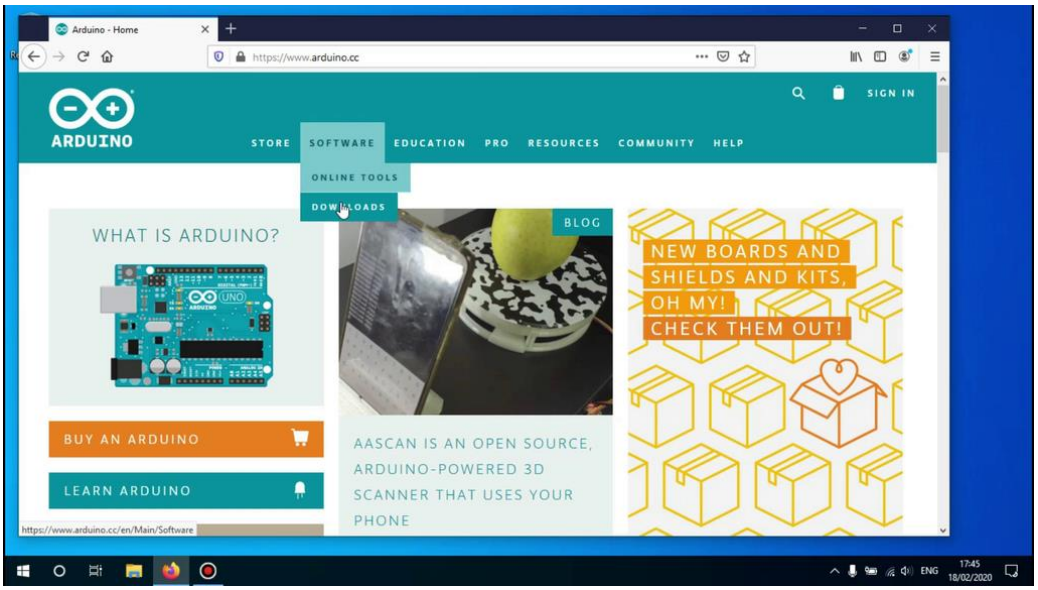
Aim: To install the Arduino IDE on Windows 10

Installation instructions:

1. Download the latest release (The download will start after you click this link. ...
2. Double-click the executable (.exe) file.
3. Follow the instructions in the installation guide.
4. When completing the setup, leave Run Arduino IDE ticked to launch the application, or launch it later from the Start Menu.

Installation Steps:

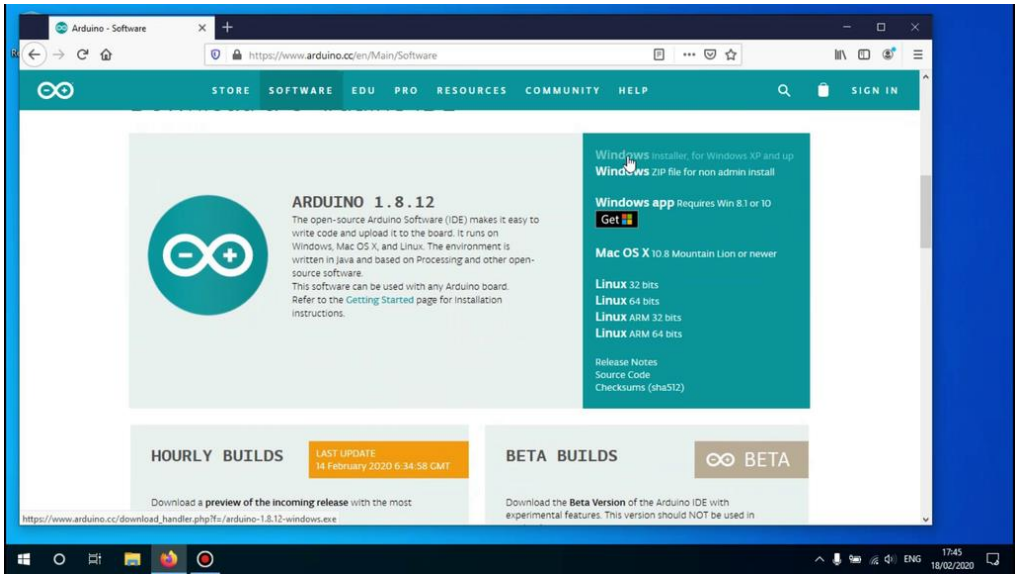
Step 1: Go to the Arduino.cc Website



Go to the website www.arduino.cc in order to download the software.

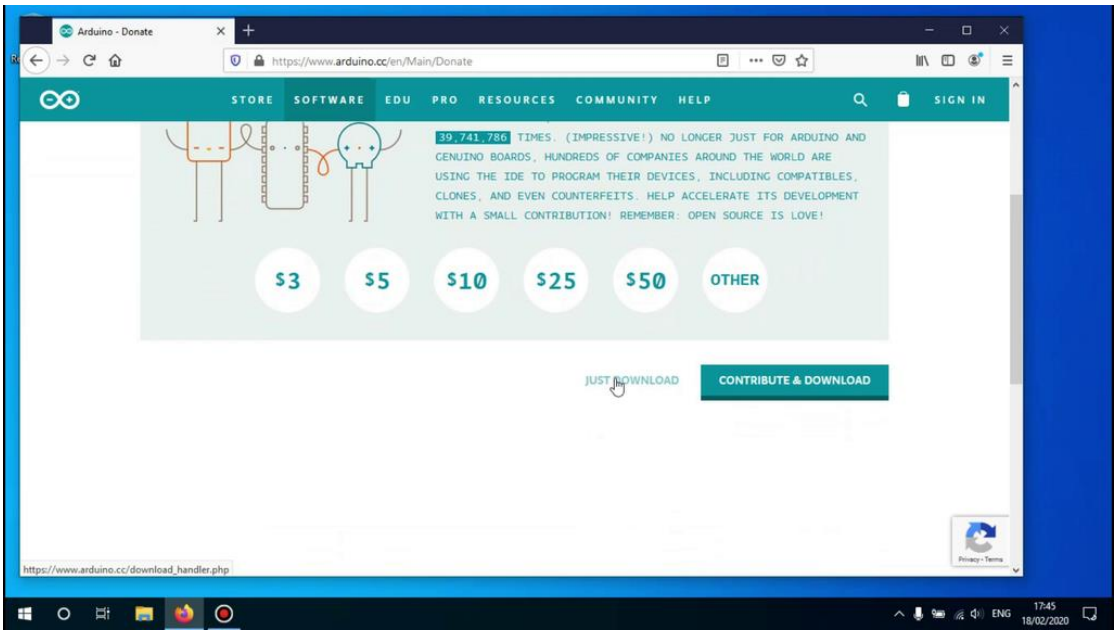
Hover over the 'Software' tab and click on 'Downloads'.

Step 2: Click on the Download Link



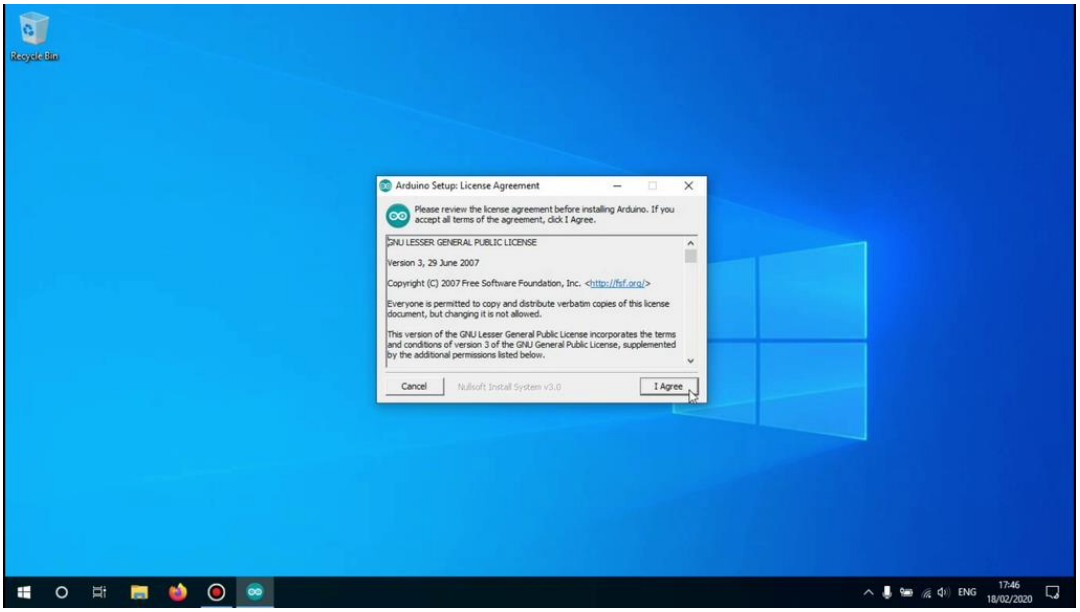
Scroll down until you see the link that says 'Windows installer' and click on it.

Step 3: Begin the Download



After clicking on the download link you'll be redirected to the donation page, here you can donate or skip it if you like by clicking on the 'Just download' link.

Step 4: Begin the Installation Process

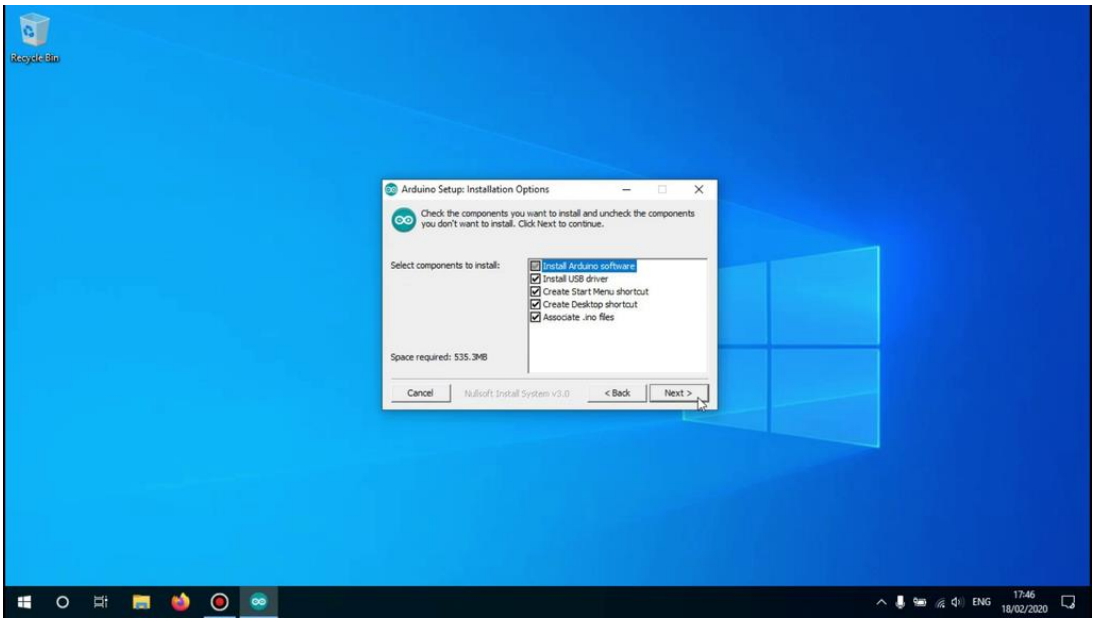


Open the downloaded file.

A new window will open asking you to agree to the license agreement.

Click on 'I agree' to continue.

Step 5: Select What to Install

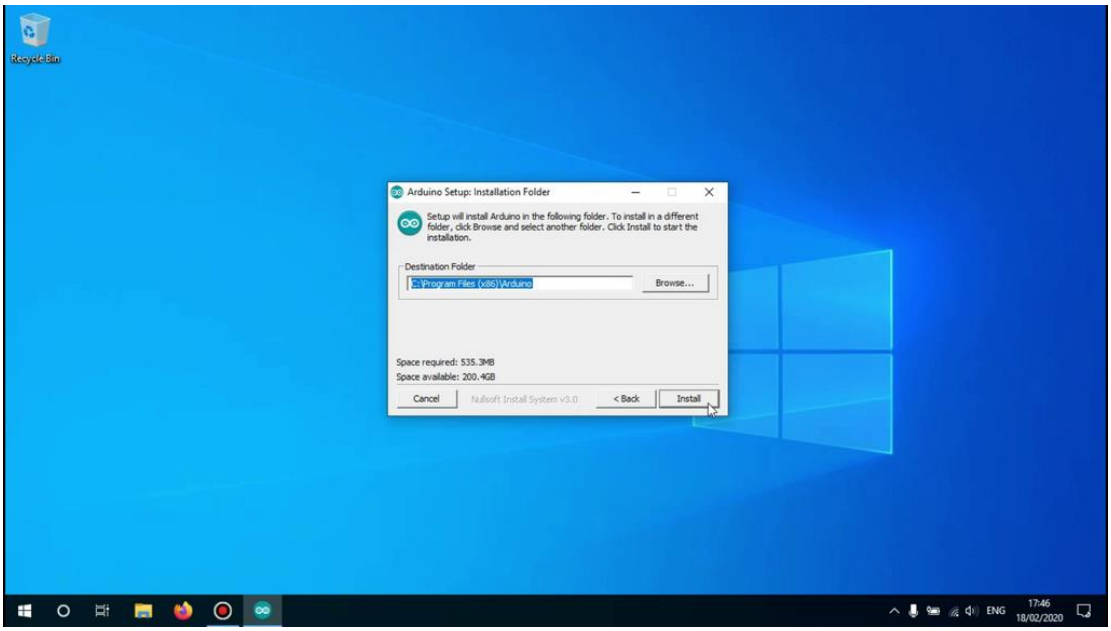


Now you'll see all the available options to install the software with.

If you don't know what you need, it is best to keep everything checked as you can change it later when the installation has finished.

Click on 'Next' to continue.

Step 6: Choose the Installation Path

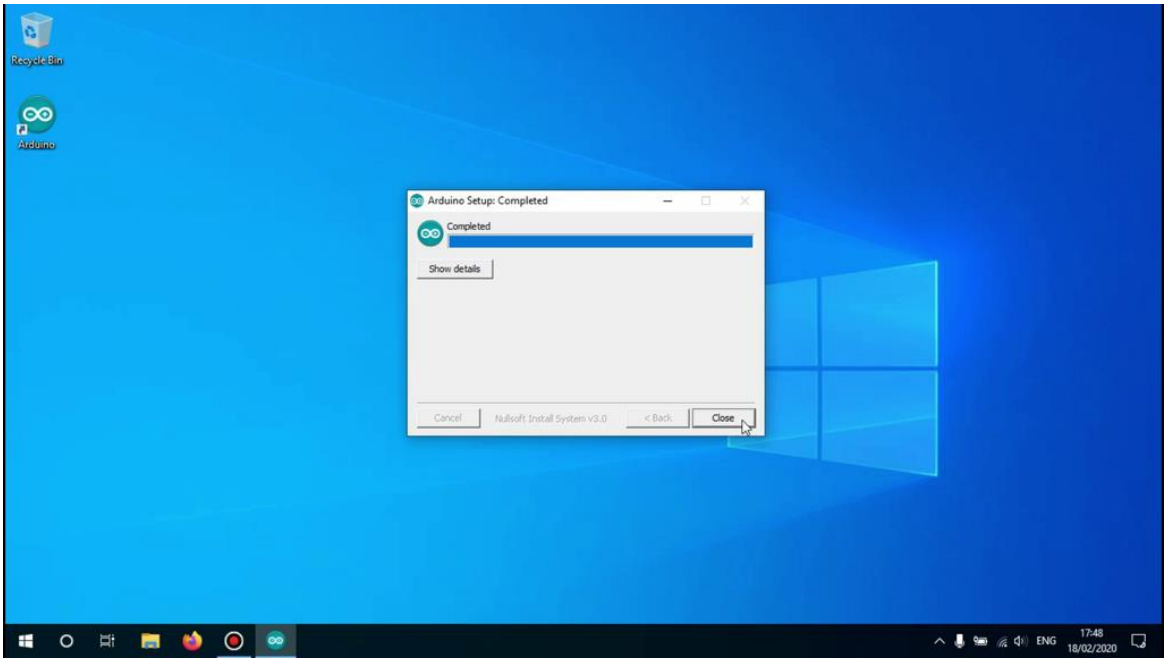


Now you have to choose the path the software will be installed in.

it is fine to leave it at the configured location but if you want the Arduino IDE somewhere else installed you can change that here.

Click on 'Install' to begin the installation

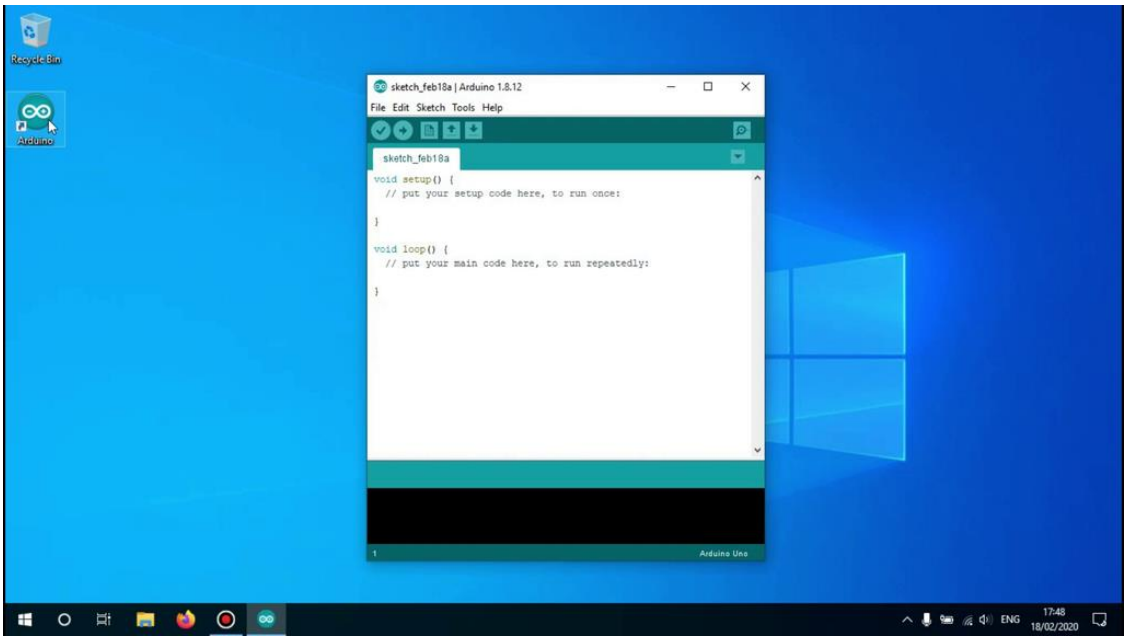
Step 7: Finish the Installation



Wait until the installation is finished, it shouldn't take very long.

When the installation is finished you may click on 'close' to end the setup wizard.

Step 8: Launch the Arduino IDE



The Arduino IDE has now been successfully installed.

To launch the IDE you can click on the Desktop icon that was created for you, or by searching for it in the start menu

Experiment 2

2A. Calculate the distance using a Ultrasonic sensor with ArduinoUno Board

Aim: Write a Arduino program and to calculate the distance of the object using a ultrasonic sensor with ArduinoUno Board

Components Required:

- Arduino Board
- USB Cable
- Ultrasonic Sensor (HC-SR04)
- Jumper Wires

Theory:

This experiment utilizes an ultrasonic sensor to measure the distance of an object from the sensor. The sensor emits ultrasonic waves and calculates the time it takes for the waves to bounce back from the object. Based on this time duration, the distance to the object is determined.

Pin Connections:

Ultrasonic Sensor:

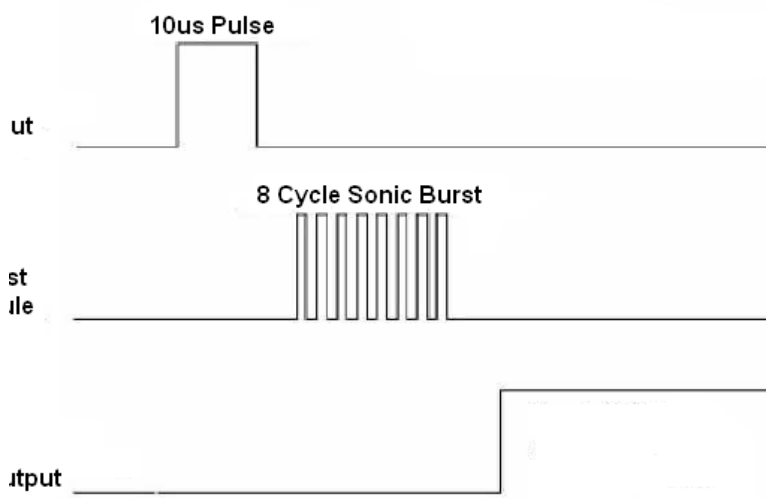
Trig Pin: Connect to digital pin 9 (trigPin) on the Arduino Uno.

Echo Pin: Connect to digital pin 10 (echoPin) on the Arduino Uno.

VCC: Connect to Arduino 5V.

GND: Connect to Arduino GND.

Working of HC-SR04:



We have to figure out the distance because the sensor itself simply holds it's "ECHO" pin HIGH for a duration of time corresponding to the time it took to receive the reflection (echo) from a wave it sent.

1. The module sends out a burst of sound waves, at the same time it applies voltage to the echo pin.

2. The module receives the reflection back from the sound waves and removes voltage from the echo pin.

On the base of the distance a pulse is generated in the ultrasonic sensor to send the data to NodeMCU or any other micro-controller.

The starting pulse is about 10us and the PWM signal will be 150 us-25us on the base of the distance. If no obstacle is there, then a 38us pulse is generated for NodeMCU to confirm that there are not objects detected.

FORMULA:

$$D = 1/2 \times T \times C$$

where D is the distance, T is the time between the Emission and Reception, and C is the sonic speed.

(The value is multiplied by 1/2 because T is the time for go-and-return distance.)

Instructions:

Set Up the Circuit:

- Connect the trig pin of the ultrasonic sensor to digital pin 9 (trigPin) on the Arduino Uno.
- Connect the echo pin of the ultrasonic sensor to digital pin 10 (echoPin) on the Arduino Uno.
- Connect the VCC pin of the ultrasonic sensor to Arduino 5V and GND pin to Arduino GND.

Initialize the System:

- Set trigPin as an output and echoPin as an input in the setup function.
- Initialize serial communication at a baud rate of 9600.

Measure Distance:

Send a 10 microsecond pulse to the trig pin of the sensor to initiate the measurement.

Measure the duration of the pulse received on the echo pin using the pulseIn() function.

Calculate the distance using the formula: distance = (duration * 0.0343) / 2, where 0.0343 is the speed of sound in centimeters per microsecond.

Output Results:

Print the calculated distance to the Serial Monitor for real-time monitoring.

Adjustments:

Modify the delay time according to the desired frequency of distance measurements.

Ensure proper alignment and calibration of the ultrasonic sensor for accurate distance readings.

Applications

- **Obstacle Detection:** Use the setup for obstacle detection in robotics projects or smart devices.
- **Parking Assistance Systems:** Implement distance sensing in parking assistance systems for vehicles.
- **Proximity Sensing**: Utilize the project for proximity sensing applications in automation and security systems.

Notes:

- Keep the ultrasonic sensor's orientation stable and ensure it is facing towards the object whose distance is being measured.
- Adjust the speed of sound value in the calculation based on environmental factors if necessary.

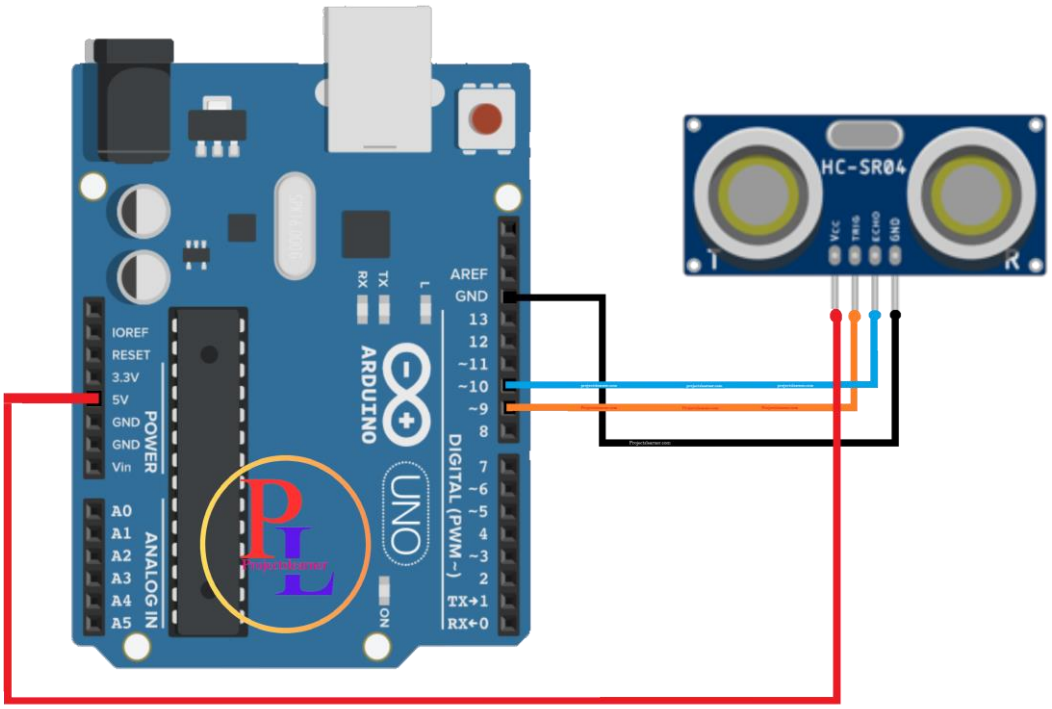


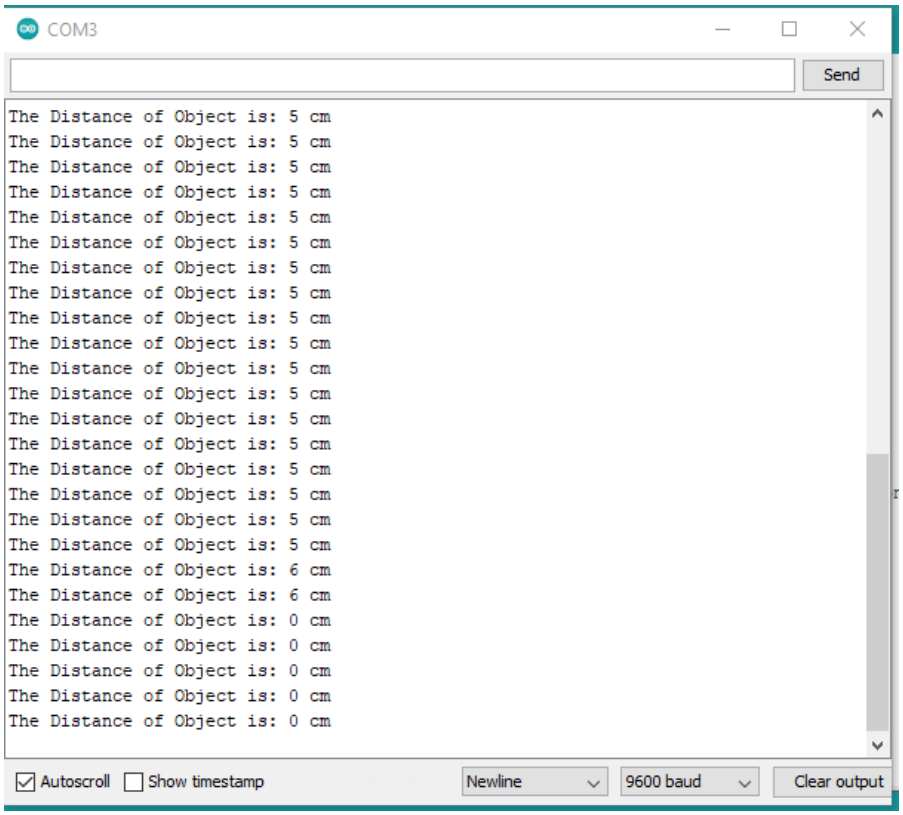
Figure: Distance using a Ultrasonic sensor with ArduinoUno Board

Arduino Code:

```
#define trigPin 9 // Trig pin of the sensor connected to Arduino pin 9
#define echoPin 10 // Echo pin of the sensor connected to Arduino pin 10


void setup() {
  Serial.begin(9600); // Initialize serial communication
  pinMode(trigPin, OUTPUT); // Set trigPin as an output
  pinMode(echoPin, INPUT); // Set echoPin as an input
}
void loop() {
  long duration, distance; // Variables to hold the duration and calculated distance
```

```
  // Clear the trigger pin
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);

  // Send a 10 microsecond pulse to the trigger pin
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Measure the duration of the pulse on the echo pin
  duration = pulseIn(echoPin, HIGH);

 // Calculate the distance based on the speed of sound (343 m/s or 0.0343 cm/microsecond)
  distance = (duration * 0.0343) / 2;

  // Print the distance to the Serial Monitor
  Serial.print("The Distance of Object is: ");
  Serial.print(distance);
  Serial.println(" cm");

  delay(1000); // Delay before taking the next measurement
}
```

Output:



Viva Questions:

1. What is the principle of ultrasonic sensor?
2. What are ultrasonic sensors used for?
3. What signal is used in an ultrasonic sensor?
4. What is the range of ultrasonic sensors?
5. What is the frequency of ultrasonic sensor?
6. What are three applications of ultrasonic sensors?
7. What is the wavelength of ultrasonic sensor?
8. Who invented the ultrasonic sensor?

Experiment 2B

2B. Basic LED functionality with ArduinoUno Board

Aim: Write a Arduino program for LED blinking with ArduinoUno Board

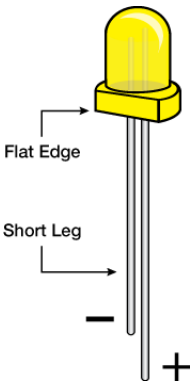
Components Required:

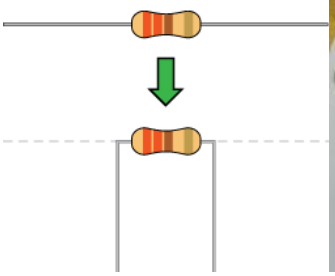
- Arduino controller board
- USB connector
- Bread board
- LED
- 1.4Kohm resistor
- connecting wires
- Arduino IDE

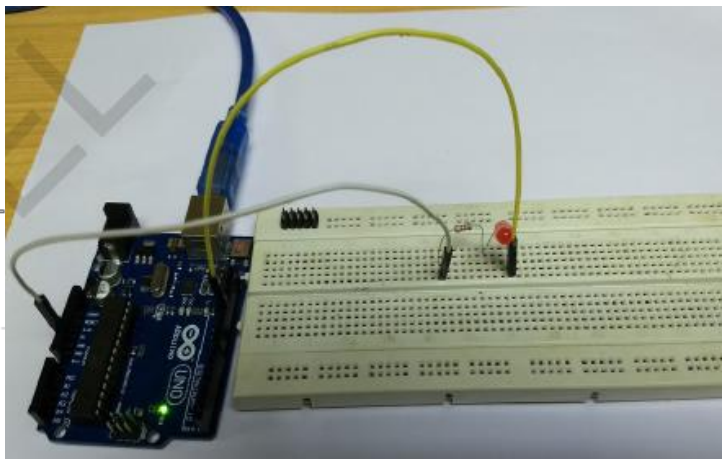
Procedure:

Follow the circuit diagram and hook up the components on the breadboard as shown in the image given below.

Note − To find out the polarity of an LED, look at it closely. The shorter of the two legs, towards the flat edge of the bulb indicates the negative terminal.







Components like resistors need to have their terminals bent into 90° angles in order to fit the breadboard sockets properly. You can also cut the terminals shorter.

- Connect the LED to the Arduino using the Bread board and the connecting wires
- Connect the Arduino board to the PC using the USB connector
- Select the board type and port
- Write the sketch in the editor, verify and upload.
- Connect the positive terminal of the LED to digital pin 12 and the negative terminal to the ground pin (GND) of Arduino Board

- Set the pin mode as output which is connected to the led, pin 12 in this case.
- Use digitalWrite() function to set the output as HIGH and LOW
- Delay() function is used to specify the delay between HIGH-LOW transition of the output
- Connect he board to the PC
- Set the port and board type
- Verify the code and upload, notice the TX – RX led in the board starts flashing as the code is uploaded.

Algorithm:

STEP 1: Start the process.

STEP 2: Start ->Arduino IDE -1.8.8

STEP 3: Then enter the coding in Arduino Software.

STEP 4: Compile the coding in Arduino Software.

STEP 5: In Arduino board, connect VCC to power supply 5V and connect to ground as in PIN gnd and connect trig and echo pins using jumper wires.

STEP 6: Connect the Arduino board with USB cable to the system.

STEP 7: Select tools -> select board ->Arduino Nano -> select processor -> AT Mega 328 p and the select port.

STEP 8: Upload the coding in Arduino board and now for the LED to blink.

STEP 9: Then, the output will be displayed in the serial monitor.

STEP 10: Stop the process.

Arduino Code:

```
/*
   Blink
   Turns on an LED on for one second, then off for one second,
repeatedly.
*/
# define ledpin 12;
void setup() {
pinMode(ledpin, OUTPUT); // set the pin mode
}
void loop() {
digitalWrite(ledpin, HIGH); // Turn on the LED
delay(1000);
digitalWrite(ledpin, LOW); //Turn of the LED
```

```
delay(1000);
}
```

Code to Note:

pinMode(ledpin, OUTPUT) − Before you can use one of Arduino's pins, you need to tell Arduino Uno R3 whether it is an INPUT or OUTPUT. We use a built-in "function" called pinMode() to do this.

digitalWrite(ledpin, HIGH) − When you are using a pin as an OUTPUT, you can command it to be HIGH (output 5 volts), or LOW (output 0 volts).

Result:

Viva Questions:

1. What is the function of LED in Arduino?
2. What is the LED on the Arduino UNO board?
3. How to use LED on Arduino UNO?
4. How many LEDs can Arduino UNO control?
5. What resistor to use with LED?
6. What is the use of LED function?

Experiment 2C

2C. Calculate temperature using DHT/LM35 sensor with ArduinoUno Board

Aim: Write a Arduino program and to Calculate temperature using a DHT/LM 35 sensor with ArduinoUno Board

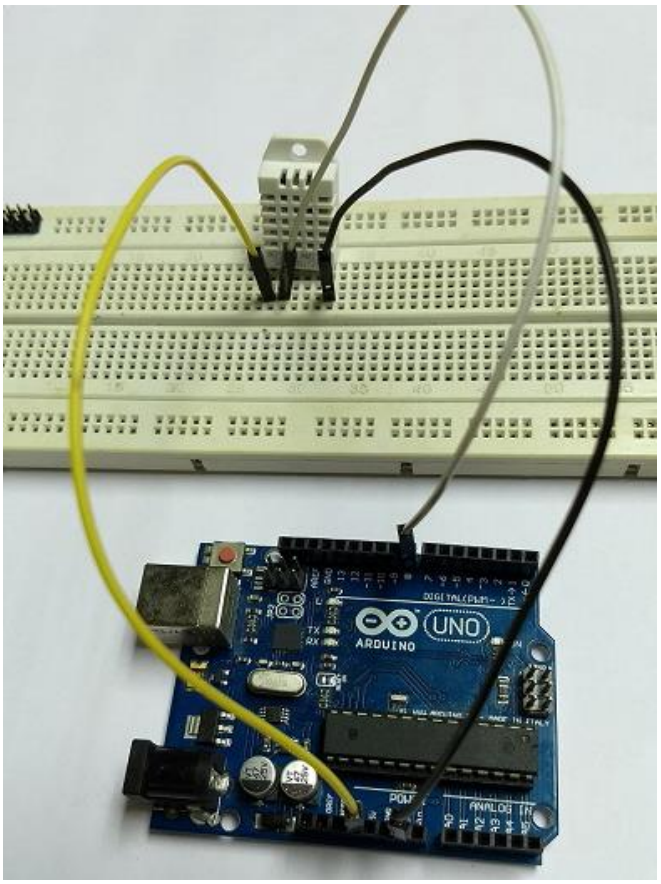
Components Required

- 1 × Breadboard
- 1 × Arduino Uno R3
- 1 × DHT / LM35 sensor
- Jumper wires
- Connectivity cable or USB cable.

Procedure:

Sensor Interface with Arduino:

- Digital Humidity and Temperature Sensor (DHT)
- PIN 1, 2, 3, 4 (from left to right)
    - *PIN 1- 3.3V-5V Power supply*
    - *PIN 2- Data*
    - *PIN 3- Null*
    - *PIN 4- Ground*



DHT Sensor Library:

- Arduino supports a special library for the DHT11 and DHT22 sensors
- Provides function to read the temperature and humidity values from the data pin

```
dht.readHumidity()
dht.readTemperature()
```

Connections:

Connect pin 1 of the DHT to the 3.3 V supply pin in the board

Data pin (pin 2) can be connected to any digital pin, here 12

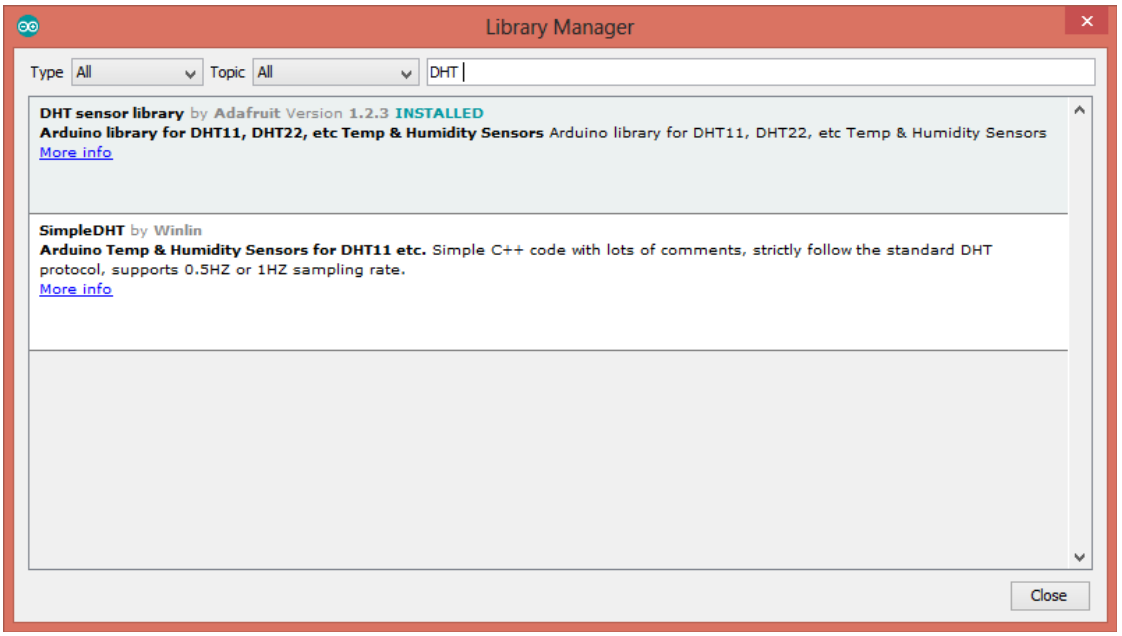
Connect pin 4 to the ground (GND) pin of the board

Install the DHT Sensor Library:

Go to Sketch -> Include Library ->Manage Library

Search for DHT SENSOR

Select the “DHT sensor library” and install it



Connect the board to the PC

Set the port and board type

Verify and upload the code

Algorithm:

STEP 1: Start the process.

STEP 2: Start the Arduino 1.8.8

STEP 3: Include the DHT library to the Arduino software.

STEP 4: Then enter the coding in Arduino software.

STEP 5: Complete the coding in Arduino.

STEP 6: In Arduino board connect VCC to the power supply 5V and connect SIG to digital signal DT and connect SND to ground GND using jumper wires.

STEP 7: Connect the arduino board with USB cable to the system.

STEP 8: Select tools □ Selected.

STEP 9: Upload the coding to arduino board. Then the output will be displayed in the serial monitor.

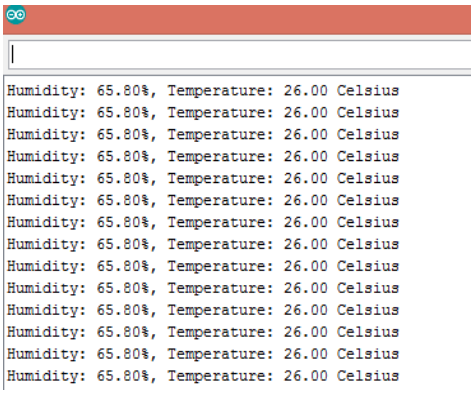
STEP 10: Stop the process.

Arduino Code:

```
#include <DHT.h>;
DHT dht(8, DHT22);//Initialize DHT sensor
float humidity; //Stores humidity value
float temperature; //Stores temperature value
void setup()
{
Serial.begin(9600);
dht.begin();
}
void loop()
{
//Read data from the sensor and store it to variables humidity and temperature
humidity = dht.readHumidity();
temperature= dht.readTemperature();
//Print temperature and humidity values to serial monitor
Serial.print("Humidity: ");
Serial.print(humidity);
Serial.print("%, Temperature: ");
Serial.print(temperature);
Serial.println(" Celsius");
delay(2000); //Delay of 2 seconds
}
```

Output:

The readings are printed at a delay of 2 seconds as specified by the delay() function



Result:
Thus the output to get temperature notification using Arduino has successfully executed.

Viva Questions:

1. What is the principle of DHT sensor?
2. What does DHT stand for in sensors?
3. What is the voltage of DHT sensor?
4. What is the temperature of DHT11 and DHT22?
5. What is the humidity value of the DHT sensor?
6. What is DHT library?
7. What is current rating(Operating Voltage and Current) of DHT11?

Experiment 3

3A. Calculate the distance using a distance sensor with NodeMCU Board

Aim: Write a Arduino program and to calculate the distance of the object using a ultrasonic sensor on NodeMCU Board

Hardware Requirements:

- NodeMCU
- HC-SR04 (Ultra-sonic Sensor)
- Bread Board
- Jumper Wires
- Micro USB Cable

Software Requirements:

- Arduino IDE

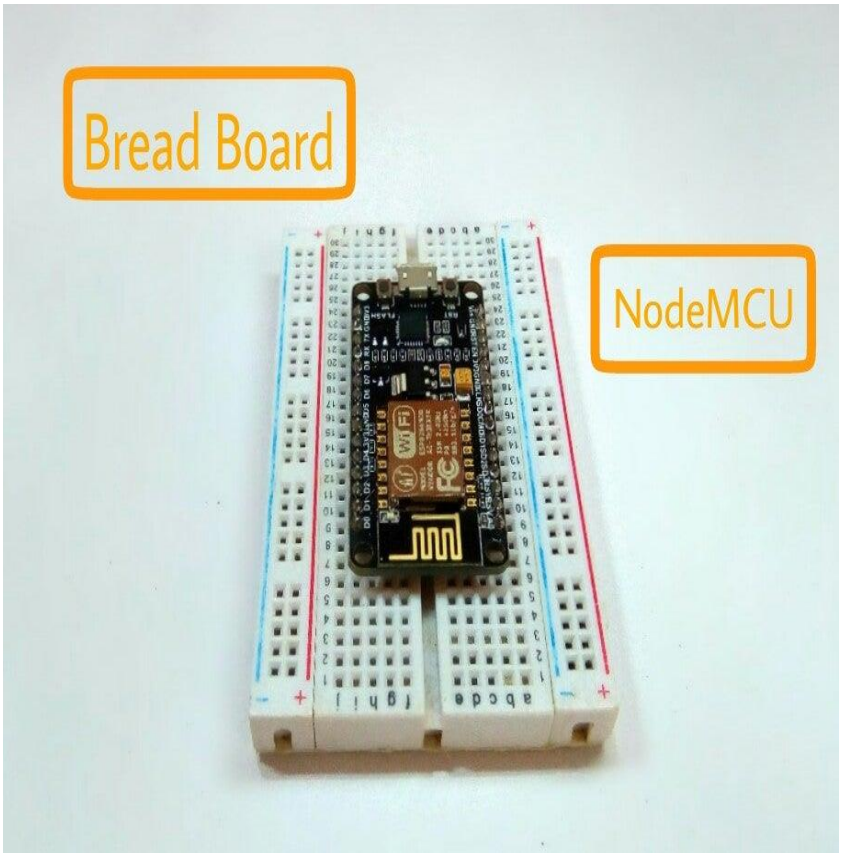
Description:

Specification of HC-SR04:

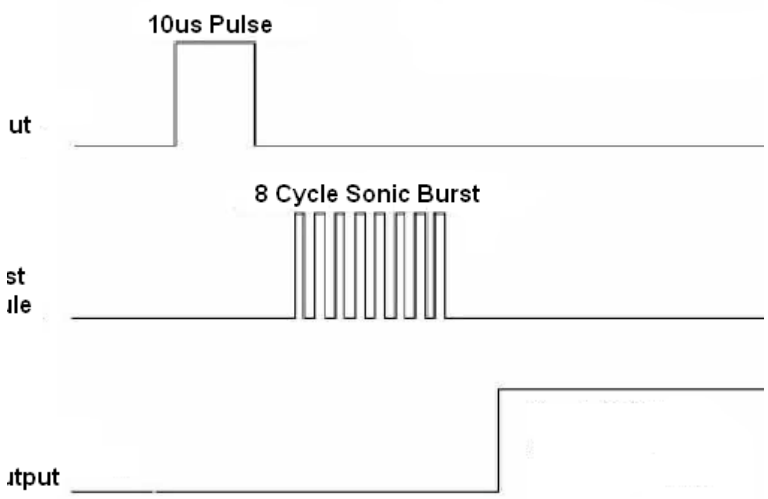
Power supply: 5v DC

Ranging distance: 2cm – 500 cm

Ultrasonic Frequency: 40k Hz



Working of HC-SR04:



We have to figure out the distance because the sensor itself simply holds it's "ECHO" pin HIGH for a duration of time corresponding to the time it took to receive the reflection (echo) from a wave it sent.

3. The module sends out a burst of sound waves, at the same time it applies voltage to the echo pin.
4. The module receives the reflection back from the sound waves and removes voltage from the echo pin.

On the base of the distance a pulse is generated in the ultrasonic sensor to send the data to NodeMCU or any other micro-controller.

The starting pulse is about 10us and the PWM signal will be 150 us-25us on the base of the distance. If no obstacle is there, then a 38us pulse is generated for NodeMCU to confirm that there are not objects detected.

FORMULA:

$$D = 1/2 \times T \times C$$

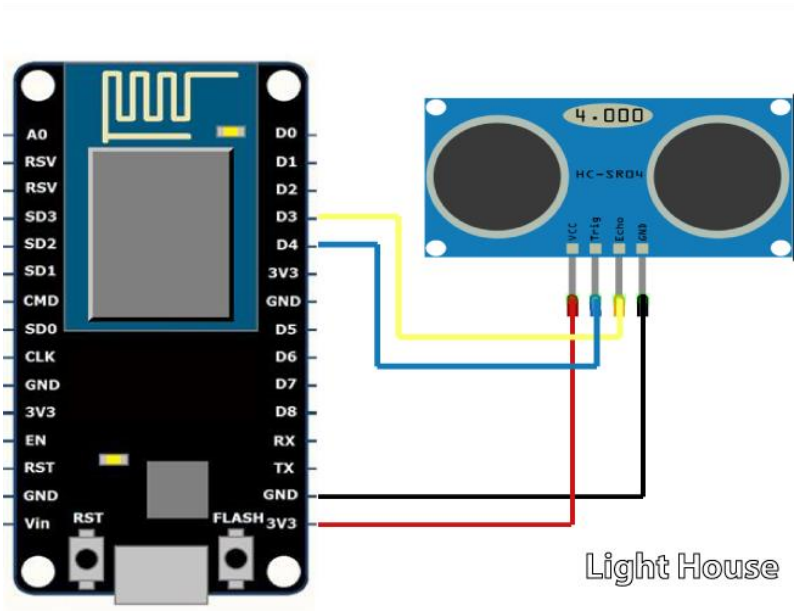
where D is the distance, T is the time between the Emission and Reception, and C is the sonic speed.

(The value is multiplied by 1/2 because T is the time for go-and-return distance.)

Interface HC-SR04:

- The **circuit connections** are made as follows:
- The HC-SR04 sensor attach to the Breadboard
- The sensor **Vcc** is connected to the NodeMCU **+3.3v**
- The sensor **GND** is connected to the NodeMCU **GND**
- The sensor **Trigger Pin** is connected to the NodeMCU Digital I/O **D4**
- The sensor **Echo Pin** is connected to the NodeMCU Digital I/O **D3**
- Before you get started with coding you need Arduino IDE.

- To download Arduino IDE and for NodeMCU setup, you can check my previous instructacle.



Code:

```
// defines pins numbers
 const int trigPin = 2;  //D4
 const int echoPin = 0;  //D3

 // defines variables
 long duration;
 int distance;

 void setup() {
 pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
 pinMode(echoPin, INPUT); // Sets the echoPin as an Input
 Serial.begin(9600); // Starts the serial communication
}
void loop() {

 // Clears the trigPin
     digitalWrite(trigPin, LOW);
     delayMicroseconds(2);

 // Sets the trigPin on HIGH state for 10 micro seconds
      digitalWrite(trigPin, HIGH);
     delayMicroseconds(10);
     digitalWrite(trigPin, LOW);
```

```
    // Reads the echoPin, returns the sound wave travel time in
    microseconds
          duration = pulseIn(echoPin, HIGH);

    // Calculating the distance
          distance= duration*0.034/2;

    // Prints the distance on the Serial Monitor
          Serial.print("Distance: ");
          Serial.println(distance);
          delay(2000);
  }
```

Output:

Result:

Viva Questions:

1. What is the purpose of NodeMCU?
2. Which IC is used in NodeMCU?
3. Which processor is used in NodeMCU?
4. Power Required By NodeMCU
5. Which software is used for NodeMCU?
6. Which cable is used for NodeMCU?
7. What is ESP8266 full form?

Experiment 3B

3B. Basic LED functionality with NodeMCU Board

Aim: Write a Arduino program for LED blinking with NodeMCU Board

Hardware Preparation:
NodeMCU x 1

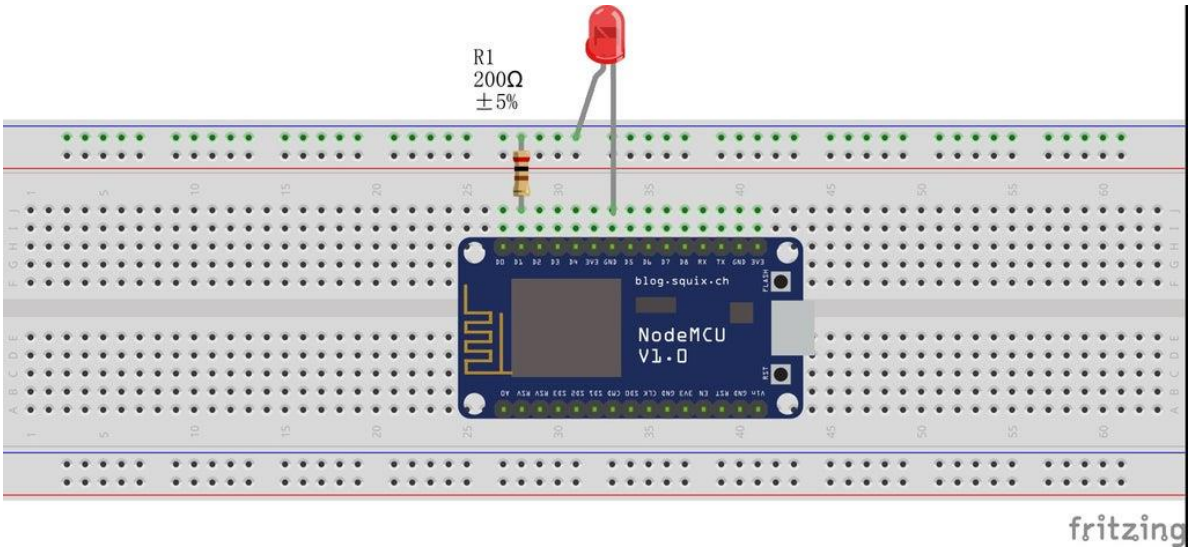
LED x 1

200 ohm resistor x 1

Micro USB cable x 1

PC x 1

Software Arduino IDE(version 1.6.4+)



Connect one end of the resistor to the digital pin correspondent to the LED_BUILTIN constant.

Connect the long leg of the LED (the positive leg, called the anode) to the other end of the resistor.

Connect the short leg of the LED (the negative leg, called the cathode) to the GND. In the diagram we show a NodeMCU that has D1 as the LED_BUILTIN value.

The value of the resistor in series with the LED may be of a different value than 200 ohm; the LED will lit up also with values up to 1K ohm.

Code:

```
#define LED D1 // Led in NodeMCU at pin GPIO16 (D0).

void setup() {
```

```
pinMode(LED, OUTPUT); // set the digital pin as output.
}
void loop() {

digitalWrite(LED, HIGH);// turn the LED off.(Note that LOW is the
//voltage level but actually
 //the LED is on; this is because it is acive low on the ESP8266.
delay(1000);          // wait for 1 second.
digitalWrite(LED, LOW); // turn the LED on.
delay(1000);         // wait for 1 second.
}
```

Output:

Result:

Viva Questions:

1. What is GPIO in NodeMCU?
2. How many bits is a NodeMCU?
3. How many types of NodeMCU are there?
4. What is the platform of NodeMCU?
5. What protocol does NodeMCU use?
6. What is the another name for NodeMCU?
7. Which IDE is used for NodeMCU?
8. Does NodeMCU have IP address?
9. Is ESP8266 analog or digital?
10. What is the application of NodeMCU?
11. Does NodeMCU have Bluetooth?
12. How many sensors can be connected to NodeMCU?

Experiment 3C

3C. Calculate temperature using a temperature sensor with NodeMCU Board

Aim: Write a Arduino program and to Calculate temperature using a temperature sensor with NodeMCU Board

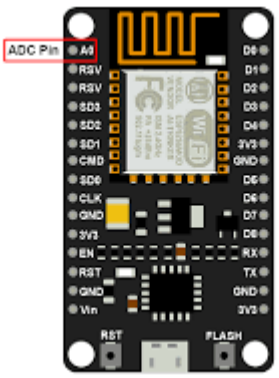
https://roboindia.com/tutorials/ds18b20-temp-sensor-nodemcu/

Digital Thermometer and monitor temperature over internet from anywhere.This instructable is a basic one to start tinkering with IoT. We will be interfacing temperature sensor LM35 with NodeMCU 1.0(ESP-12E).

LM35 is a temperature sensor which can measure temperature in the range of -55°C to 150°C.It is a 3-terminal device that provides analog voltage proportional to the temperature. NodeMCU ADC can be used to measure the analog voltage from LM35 and hence calculate the temperature which is in proportion to the analog voltage.

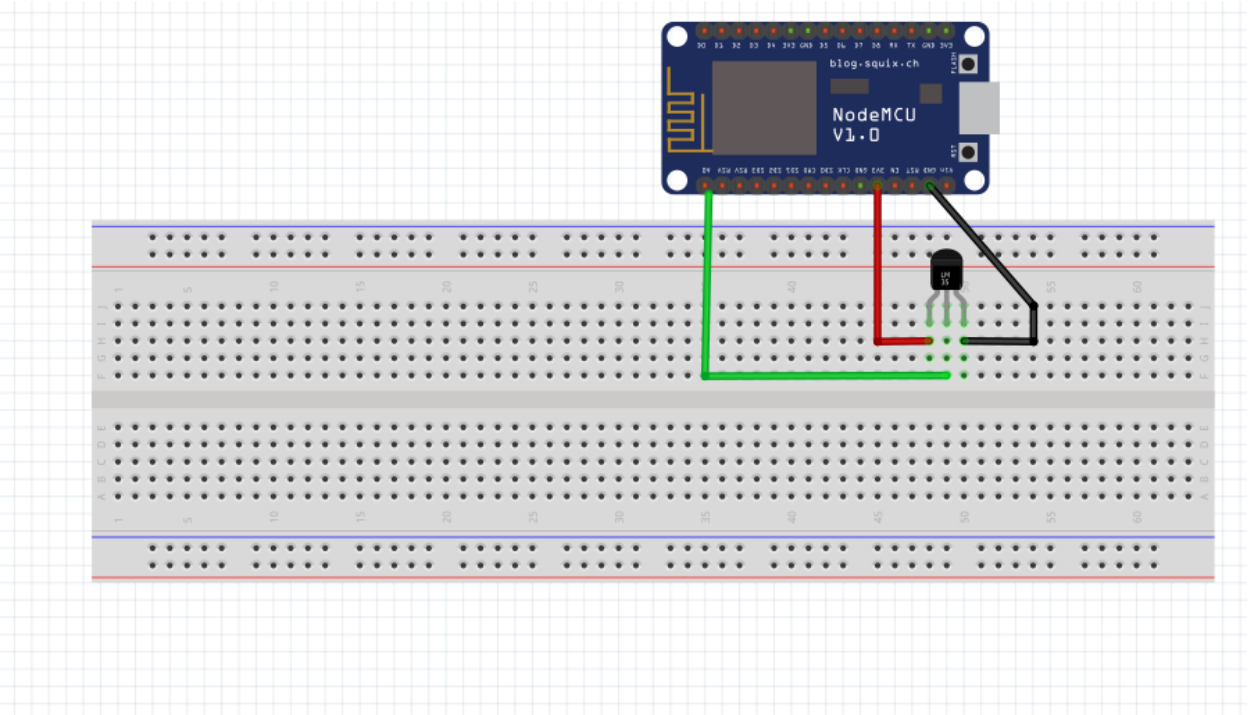
Components Required

- LM35 Temperature Sensor
- NodeMCU 1.0(ESP 12-E Module)
- Connecting Wires
- Breadboard
- Arduino IDE



Make Connections

1. Make connections as shown in the circuit diagram.
2. Connect the Vcc pin of LM35 to 3V pin of NodeMCU.
3. Connect the Analog pin of LM35 to A0 of NodeMCU.
4. Connect the GND pin of LM35 to GND of NodeMCU.



Conversion of Analog Value to Digital Value

- The Analog to Digital Converter (ADC) converts analog values into a digital approximation based on the formula:

  ADC Value = sample * 1024 / reference voltage

- To convert analog values to Celsius we have 3.3 V on our board and we know that output voltage of LM35 varies by 10 mV to every degree Celsius rise/fall.

  temp_celsius = ((analogRead(A0) * 330.0)/1024.0);

- To convert Celsius to Fahrenheit

  temp_fahrenheit = (temp_celsius * 1.8)+32.0;

Interfacing LM35 and NodeMCU

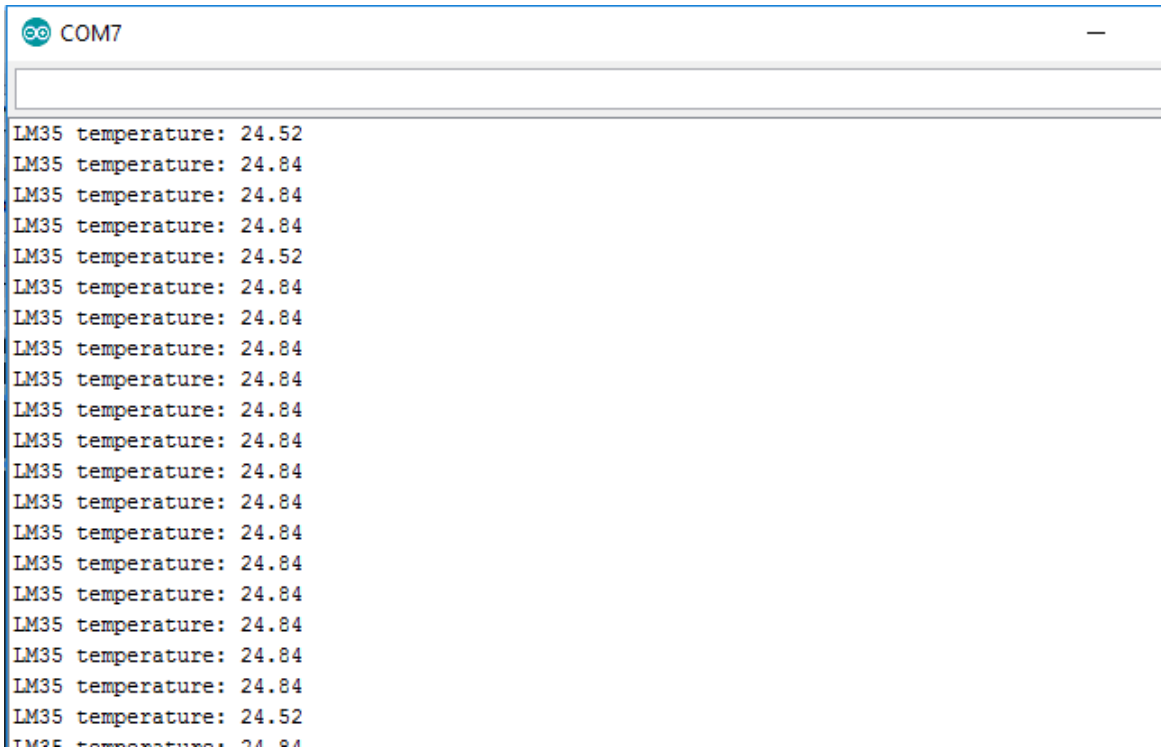
1. Open Arduino IDE.
2. Go to Tools -> Boards -> NodeMCU 1.0 (ESP 12-E Module).
3. Copy the code. (The Code has been attached below).
4. Compile it.
5. Upload it to NodeMCU.

Code

```
float vref = 3.3;

float resolution = vref/1023;


void setup() {
```

```
 Serial.begin(9600);  /* Define baud rate for serial
communication */
}

void loop() {
 float temperature = analogRead(A0);
 temperature = (temperature*resolution);
 temperature = temperature*100;
 Serial.println(temperature);
 delay(1000);
}
```

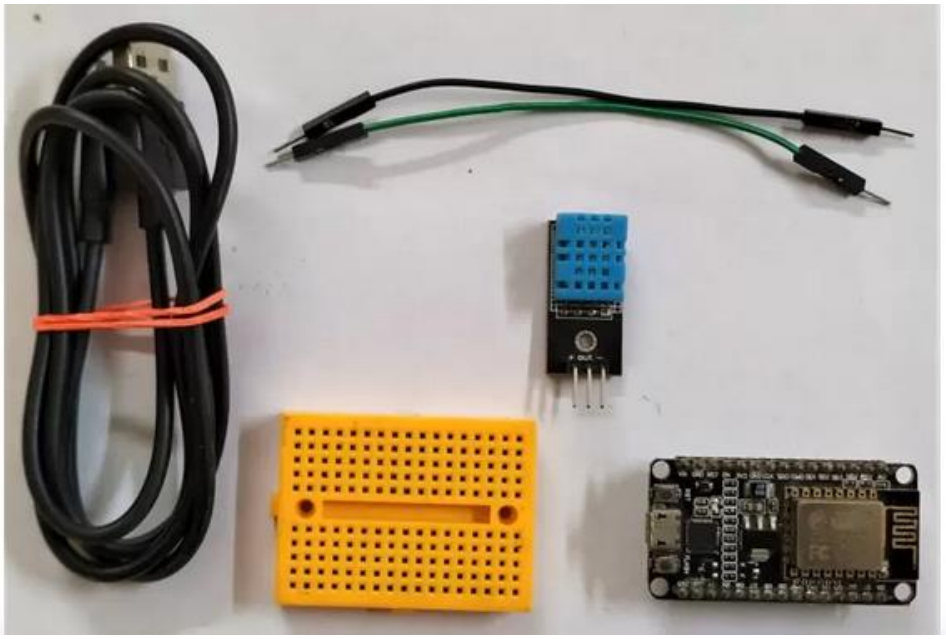
Output:



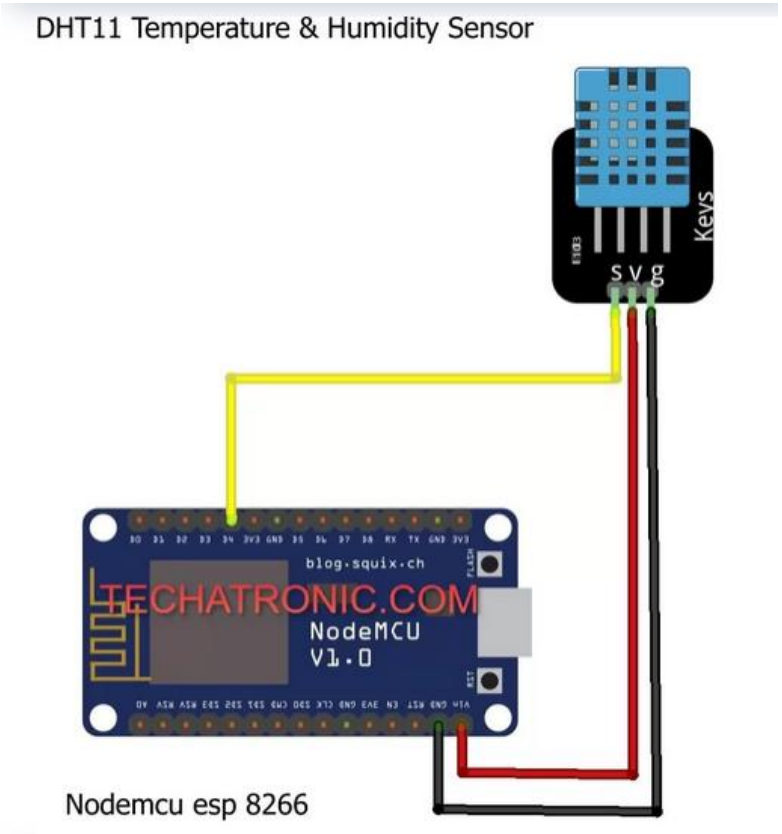
Interfacing of DHT11 Sensor with ESP8266 NodeMCU

Components Required

- ESP8266 nodemcu board
- DHT-11 sensor
- Jumper wires and a breadboard
- USB cable for uploading the code



Circuit Diagram



Connections:

- There are three pins in the DHT-11 sensor out of which two are for power and one is for the output data transmission.
- You have to connect all three pins to the nodemcu.
- Connect the VCC pin of the sensor with the VIN pin of the nodemcu.
- Join the GND pin of the sensor to the GND pin of the nodemcu.
- At last, connect the remaining pin that is OUT pinned with the digital-4 pin of the nodemcu.

Code:

install <DHT.h> library to the IDE software.

```
#include "DHT.h"
 DHT dht2(2,DHT11);
 void setup()
 {
  Serial.begin(9600);
 }
 void loop()
 {
   Serial.println("Temperature in C:");
   Serial.println((dht2.readTemperature( )));
   Serial.println("Humidity in C:");
   Serial.println((dht2.readHumidity()));
   delay(1000);
 }
```

Output:

Result:

Viva Questions:

1. Does NodeMCU have IP address?
2. Is ESP8266 analog or digital?
3. What is the application of NodeMCU?
4. Does NodeMCU have Bluetooth?
5. How many sensors can be connected to NodeMCU?

Experiment 4

4A. Calculate the distance using a distance sensor with Raspberry Pi board.

Aim: Write a python program and to Calculate the distance of the object using a ultrasonic sensor with Raspberry Pi board

Components Required:

- Raspberry Pi with pre-installed OS
- HC-SR04 Ultrasonic Sensor
- Power supply (5v)
- 1KΩ resistor (3 pieces)
- 1000uF capacitor
- 16*2 character LCD

Circuit Explanation:
Connections between Raspberry Pi and LCD are given in the below table:

LCD connection	Raspberry Pi connection
GND	GND
VCC	+5V
VEE	GND
RS	GPIO17
R/W	GND
EN	GPIO27
D0	GPIO24
D1	GPIO23
D2	GPIO18
D3	GPIO26
D4	GPIO5
D5	GPIO6
D6	GPIO13
D7	GPIO19

In this circuit, we used 8bit communication (D0-D7) to connect LCD with Raspberry Pi, however this is not a compulsory, we can also use 4-bit communication (D4-D7), but with 4 bit communication program becomes a bit complex for beginners so just go with 8 bit communication. Here we have connected 10 pins of LCD to Raspberry Pi in which 8 pins are data pins and 2 pins are control Pins.
Below is the circuit diagram for **connecting HC-SR04 sensor and LCD with Raspberry Pi for measuring the distance**.

As shown in the figure, **HC-SR04 Ultrasonic Sensor has four pins**,

1. PIN1- VCC or +5V
2. PIN2- TRIGGER (10us High pulse given to tell the sensor to sense the distance)
3. PIN3- ECHO (Provides pulse output whose width represents distance after trigger)
4. PIN4- GROUND

Echo pin provides +5V output pulse which cannot be connected to Raspberry Pi directly. So we will be using Voltage Divider Circuit (built using R1 and R2) to get +3.3V logic instead of +5V logic.

Working Explanation:

Complete working of **Raspberry Pi Distance Measure** goes as,

1. Triggering the sensor by pulling up the trigger pin for 10uS.
2. Sound wave is sent by the sensor. After receiving the ECHO, sensor module provides an output proportional to distance.
3. We will record the time when the output pulse goes from LOW to HIGH and when again when its goes form HIGH to LOW.
4. We will have start and stop time. We will use distance equation to calculate the distance.
5. The distance is displayed in 16x2 LCD display.

Accordingly we have written the **Python Program** for Raspberry Pi to do the following functions:

1. To send trigger to sensor
2. Record start and stop time of pulse output from sensor.
3. To Calculate the distance by using START and STOP time.
4. To Display the result obtained on the 16*2 LCD.

Python Code:

```
import time
import RPi.GPIO as IO    #calling for header file which helps in using GPIOs of PI
string_of_characters = 0
IO.setwarnings(False) #do not show any warnings
IO.setmode (IO.BCM)   #programming the GPIO by BCM pin numbers. (like PIN29 as GPIO5)
IO.setup(17,IO.OUT)   #initialize GPIO17,27,24,23,18,26,5,6,13,19,21 as an output
IO.setup(27,IO.OUT)
IO.setup(24,IO.OUT)
IO.setup(23,IO.OUT)
IO.setup(18,IO.OUT)
IO.setup(26,IO.OUT)
IO.setup(5,IO.OUT)
```

```
IO.setup(6,IO.OUT)
IO.setup(13,IO.OUT)

IO.setup(19,IO.OUT)
IO.setup(21,IO.OUT)
IO.setup(16,IO.IN)      #initialize GPIO16 as an input
def send_a_command (command):  #steps for sending a command to 16x2 LCD

    pin=command
    PORT(pin);
    IO.output(17,0)
    #PORTD&= ~(1<<RS);
    IO.output(27,1)
    #PORTD|= (1<<E);
    time.sleep(0.001)
    #_delay_ms(50);
    IO.output(27,0)
    #PORTD&= ~(1<<E);
    pin=0
    PORT(pin);
def send_a_character (character):  #steps for sending a character to 16x2 LCD
    pin=character
    PORT(pin);
    IO.output(17,1)
    #PORTD|= (1<<RS);
    IO.output(27,1)
    #PORTD|= (1<<E);
    time.sleep(0.001)
    #_delay_ms(50);
    IO.output(27,0)
    #PORTD&= ~(1<<E);
    pin=0
    PORT(pin);
def PORT(pin):   #assigning level for PI GPIO for sending data to LCD through D0-D7

    if(pin&0x01 == 0x01):
        IO.output(24,1)
    else:
        IO.output(24,0)
    if(pin&0x02 == 0x02):
        IO.output(23,1)
```

```
    else:
        IO.output(23,0)
    if(pin&0x04 == 0x04):
        IO.output(18,1)
    else:
        IO.output(18,0)
    if(pin&0x08 == 0x08):
        IO.output(26,1)
    else:
        IO.output(26,0)
    if(pin&0x10 == 0x10):
        IO.output(5,1)
    else:
        IO.output(5,0)
    if(pin&0x20 == 0x20):
        IO.output(6,1)
    else:
        IO.output(6,0)
    if(pin&0x40 == 0x40):
        IO.output(13,1)
    else:
        IO.output(13,0)
    if(pin&0x80 == 0x80):
        IO.output(19,1)
    else:
        IO.output(19,0)
  def send_a_string(string_of_characters):
  string_of_characters = string_of_characters.ljust(16," ")
  for i in range(16):
    send_a_character(ord(string_of_characters[i])) #send characters one by one through data port

while 1:
    send_a_command(0x38);  #16x2 line LCD
    send_a_command(0x0E);  #screen and cursor ON
    send_a_command(0x01);  #clear screen
    time.sleep(0.1)        #sleep for 100msec

    IO.setup(21,1)
    time.sleep(0.00001)
    IO.setup(21,0)        #sending trigger pulse for sensor to measure the distance
```

```
while (IO.input(16)==0):
    start = time.time()  #store the start time of pulse output
while (IO.input(16)==1):
    stop = time.time()   #store the stop time
distance = ((stop - start)*17150)  #calculate distance from time
distance = round(distance,2)       #round up the decimal values

if(distance<400):    #if distance is less than 400 cm, display the result on LCD
    send_a_command(0x80 + 0);
    send_a_string ("Dist=%s cm"% (distance));
    time.sleep(0.15)

if(distance>400):    #If distance is more than 400cm, just print 400+ on LCD
    send_a_command(0x80 + 0);
    send_a_string ("Dist= 400+ cm");
    time.sleep(0.15)
```

Result:

Output:

Viva Questions:

1. What is Raspberry Pi?
2. What is Raspberry Pi?
3. What is Raspberry Pi?
4. What is Raspberry Pi?
5. What is Raspberry Pi?
6. What is Raspberry Pi?
7. What is Raspberry Pi?
8. What is Raspberry Pi?

Experiment 4B

4B. Basic LED functionality with Raspberry Pi board

Aim: Write a python program for Turning on an LED with your Raspberry Pi's GPIO Pins

Components:

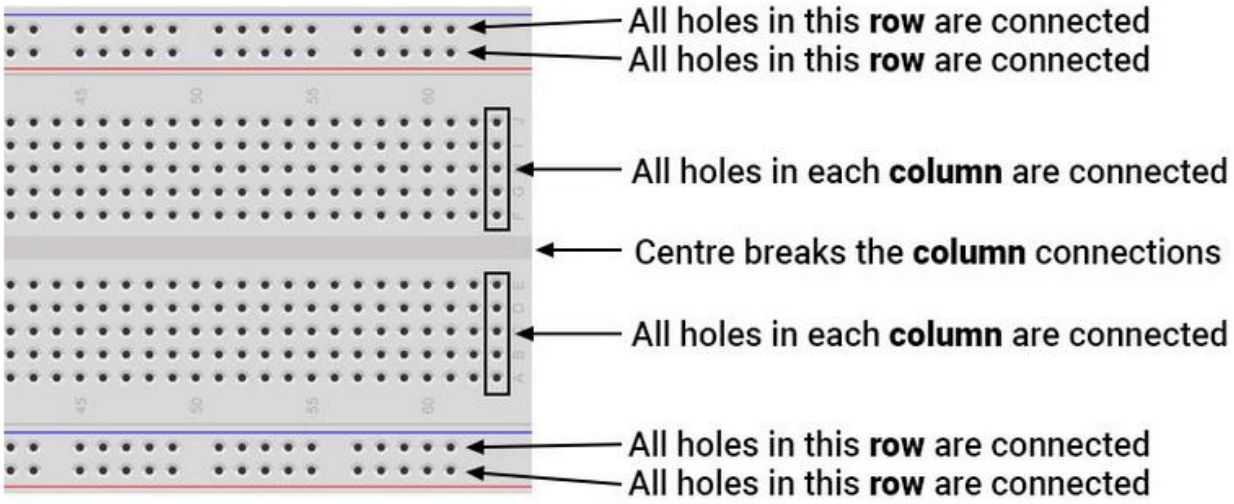
In addition to a Raspberry Pi running Raspberry Pi OS, you will need:

- A Breadboard
- An LED
- A 330 ohm resistor
- 2x Male to Female jumper wires

Theory:

The Breadboard

The breadboard is a convenient way to connect electronic components to each other without having to solder them together.



The LED

The longer leg (known as the '**anode**'), is always connected to the positive supply of the circuit. The shorter leg (known as the '**cathode**') is connected to the negative side of the power supply, known as 'ground'.

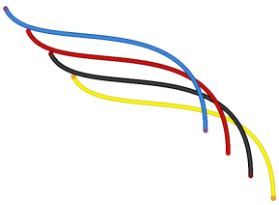
LED stands for **L**ight **E**mitting **D**iode, and glows when electricity (current) is passed through it.

The Resistor

You must **ALWAYS use resistors** to connect LEDs up to the GPIO pins of the Raspberry Pi. The Raspberry Pi GPIO pins can only supply a small current (about 60mA). The LEDs will want to draw more, and if allowed to they may damage your Raspberry Pi or the pins used. Therefore adding resistors to the circuit will ensure that only this small amount of current will flow.

Resistors are a way of limiting the amount of electricity going through a circuit; specifically, they limit the amount of 'current' that is allowed to flow. The measure of resistance is called the **Ohm (Ω)**, and the larger the resistance, the more it limits the current. The value of a resistor is marked with coloured bands along the length of the resistor body.

Jumper Wires

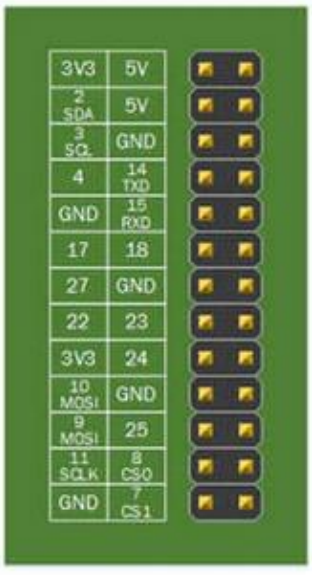


Jumper wires are used on breadboards to 'jump' from one connection to another. The ones you will be using in this circuit have different connectors on each end

The Raspberry Pi GPIO Pins

GPIO stands for General Purpose Input Output. It's a way the Raspberry Pi can control and monitor the outside world by being connected to electronic circuits.

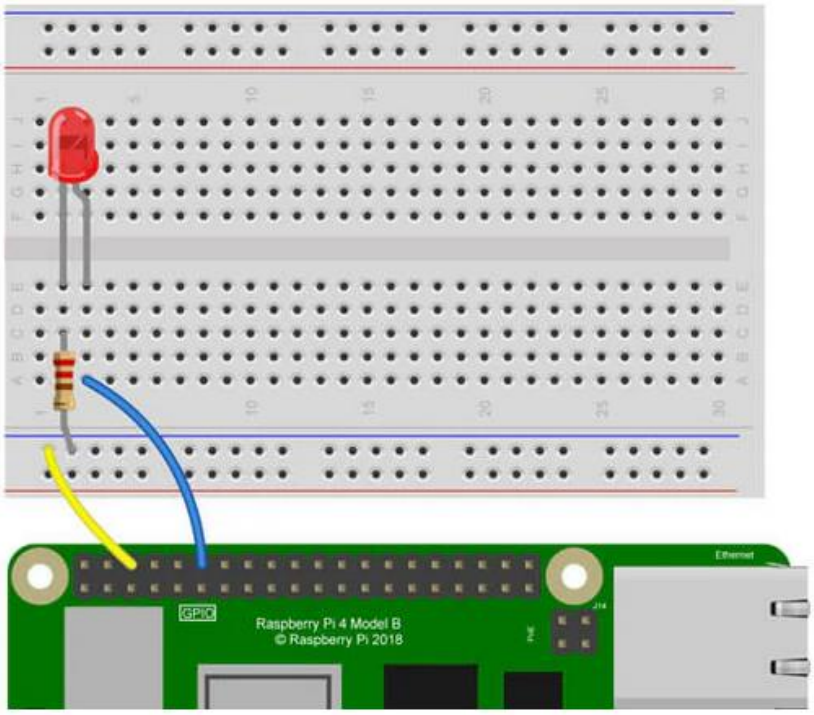
The diagram below left shows the pin layout for the older 26-pin Raspberry Pi Models which are no longer sold. The latest 40-pin Raspberry Pi's share the same layout of pins for the top 13 rows as the old model.



Building the Circuit:

The circuit consists of a power supply (the Raspberry Pi), an LED that lights when the power is applied, and a resistor to limit the current that can flow through the circuit:

- You will be using one of the ‘ground’ (GND) pins to act like the ‘negative’ or 0 volt ends of a battery.
- The ‘positive’ end of the battery will be provided by a GPIO pin. Here we will be using pin **GPIO18** (which is *physical* pin 12).
- When these pins are ‘taken high’, which means it outputs 3.3 volts, the LED will light.



- Use one of the jumper wires to connect a ground pin to the rail, marked with blue, on the breadboard. The female end goes on the Raspberry Pi's pin, and the male end goes into a hole on the breadboard.
- Then connect the resistor from the same row on the breadboard to a column on the breadboard, as shown above.
- Next, push the LEDs legs into the breadboard, with the long leg (with the kink) on the right.
- Lastly, complete the circuit by connecting the right hand leg of the LED to GPIO18. This is shown here with the blue wire.

Python Code:

Turn on your Raspberry Pi and open the terminal window.

Create a new text file “**LED.py**” by typing the following:

```
nano LED.py
```

```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(18,GPIO.OUT)
print "LED on"
GPIO.output(18,GPIO.HIGH)
time.sleep(1)
print "LED off"
GPIO.output(18,GPIO.LOW)
```

Running the Code

To run this code type:

```
sudo python LED.py
```

Output:

Result:

Viva Questions:

1. What is Raspberry Pi?
2. How does the Raspberry Pi work?
3. Tell about any interesting project with Raspberry Pi.
4. How is Raspberry Pi used in IoT?
5. What are the different components of a Raspberry Pi board?
6. What is the NOOBS software all about?
7. What are GPIO Pins used in Raspberry Pi boards?
8. Can Raspberry Pi be used as a server?
9. What is the language used by Raspberry Pi?
10. How is Raspberry Pi different from Arduino?

Experiment 5

5. Accessing GPIO pins using Python

Aim: To access GPIO pins using Python

Components Required:

Here we are using **Raspberry Pi 2 Model B with Raspbian Jessie OS**. All the basic Hardware and Software requirements are previously discussed, you can look it up in the Raspberry Pi Introduction, other than that we need:

- Connecting pins
- 220Ω or 1KΩresistor
- LED
- Bread Board

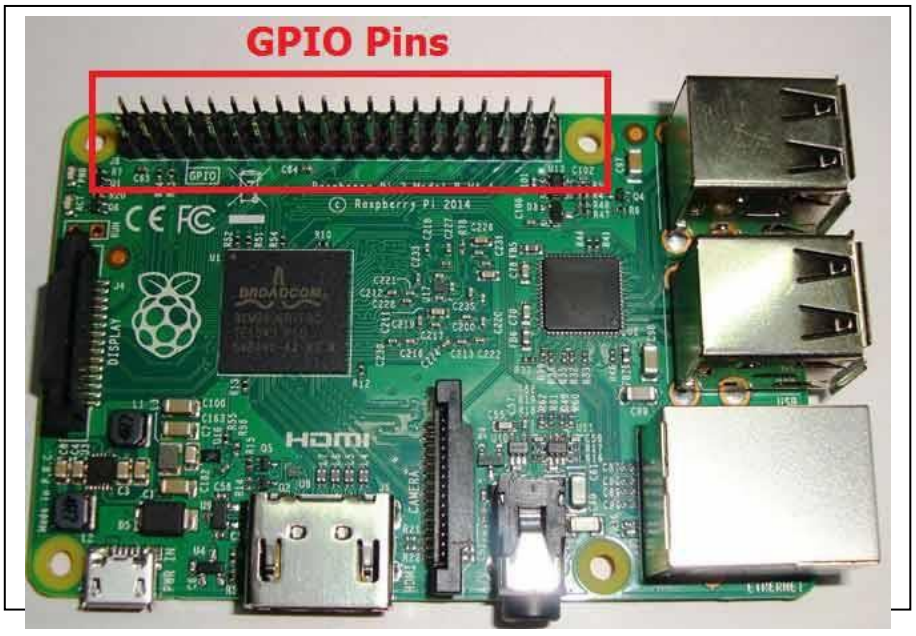
GPIO Programming:

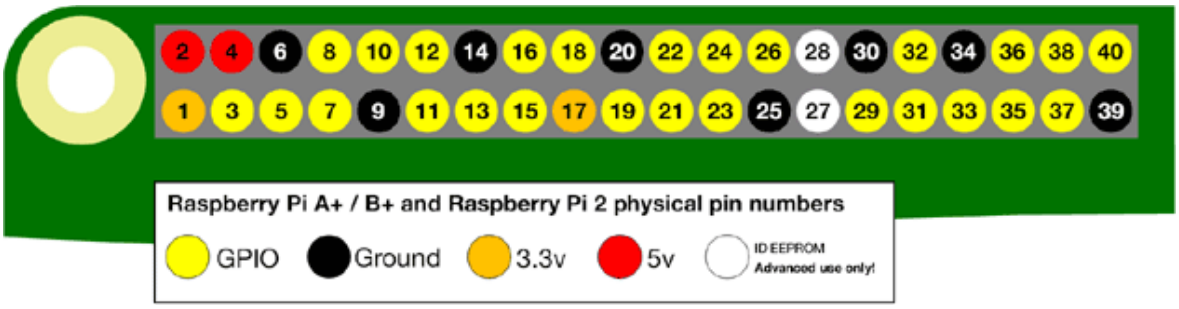
Programming of available GPIO pins of the corresponding device using native programming language. Interfacing of I/O devices like LED/Switch etc., and testing the functionality.

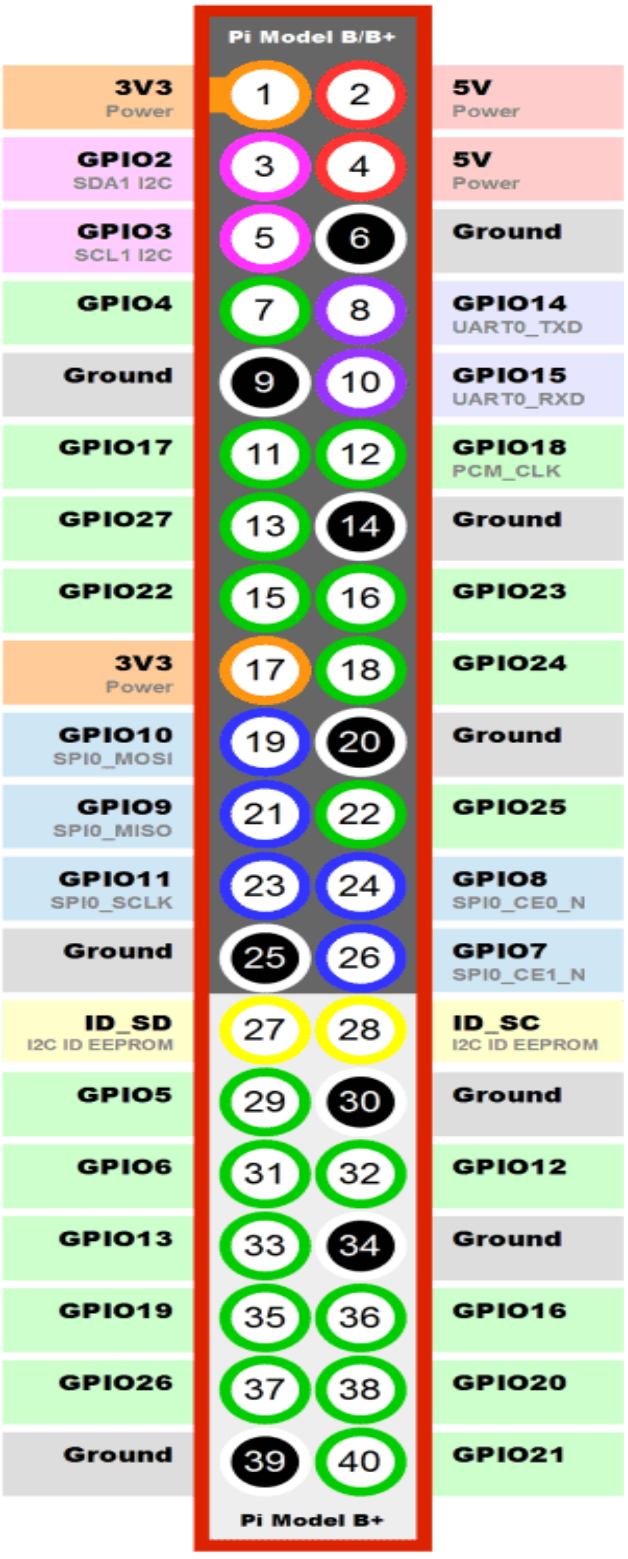
GPIO Pins:

As shown in above figure, there are 40output pins for the PI. But when you look at the second figure, you can see not all 40 pin out can be programmed to our use. These are only 26 GPIO pins which can be programmed. These pins go from **GPIO2 to GPIO27**.

These **26 GPIO pins can be programmed** as per need. Some of these pins also perform some special functions, we will discuss about that later. With special GPIO put aside, we have 17 GPIO remaining (Light green Cirl).



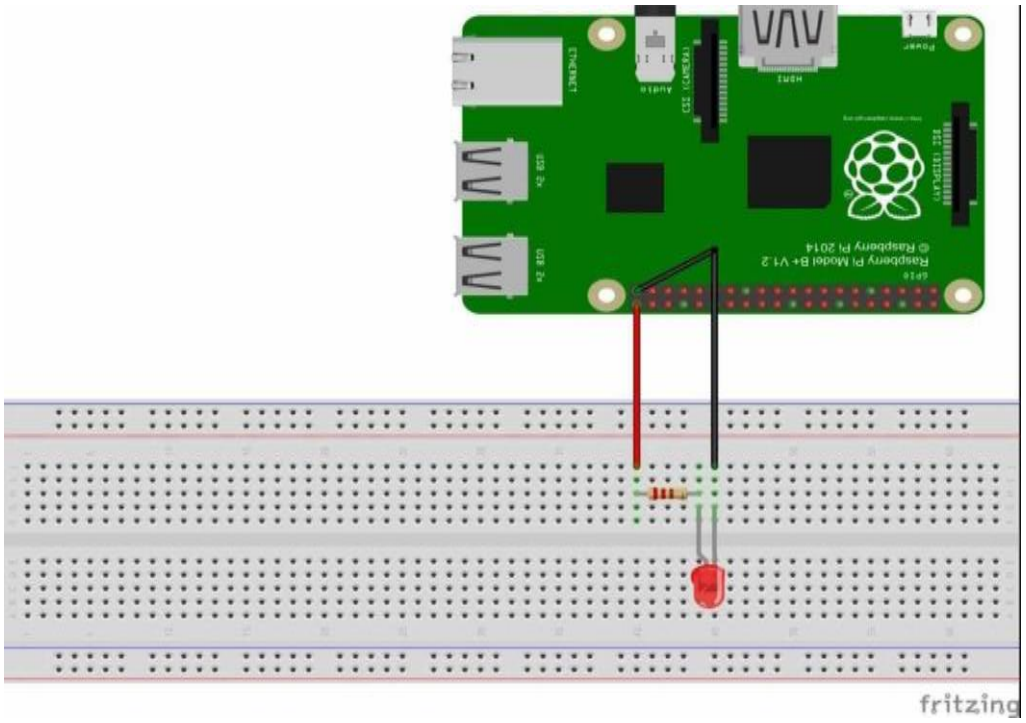

2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39
Raspberry Pi A+ / B+ and Raspberry Pi 2 physical pin numbers
GPIO
Ground
3.3v
5v
ID EEPROM
Advanced use only!


Pi Model B/B+
3V3 Power
1
2
5V Power
GPIO2 SDA1 I2C
3
4
5V Power
GPIO3 SCL1 I2C
5
6
Ground
GPIO4
7
8
GPIO14 UART0_TXD
Ground
9
10
GPIO15 UART0_RXD
GPIO17
11
12
GPIO18 PCM_CLK
GPIO27
13
14
Ground
GPIO22
15
16
GPIO23
3V3 Power
17
18
GPIO24
GPIO10 SPI0_MOSI
19
20
Ground
GPIO9 SPI0_MISO
21
22
GPIO25
GPIO11 SPI0_SCLK
23
24
GPIO8 SPI0_CE0_N
Ground
25
26
GPIO7 SPI0_CE1_N
ID_SD I2C ID EEPROM
27
28
ID_SC I2C ID EEPROM
GPIO5
29
30
Ground
GPIO6
31
32
GPIO12
GPIO13
33
34
Ground
GPIO19
35
36
GPIO16
GPIO26
37
38
GPIO20
Ground
39
40
GPIO21
Pi Model B+

Each of these 17 GPIO pins can deliver a maximum of **15mA current**. And the sum of currents from all GPIO cannot exceed 50mA. So we can draw a maximum of 3mA in average from each of these GPIO pins. So one should not tamper with these things unless you know what you are doing

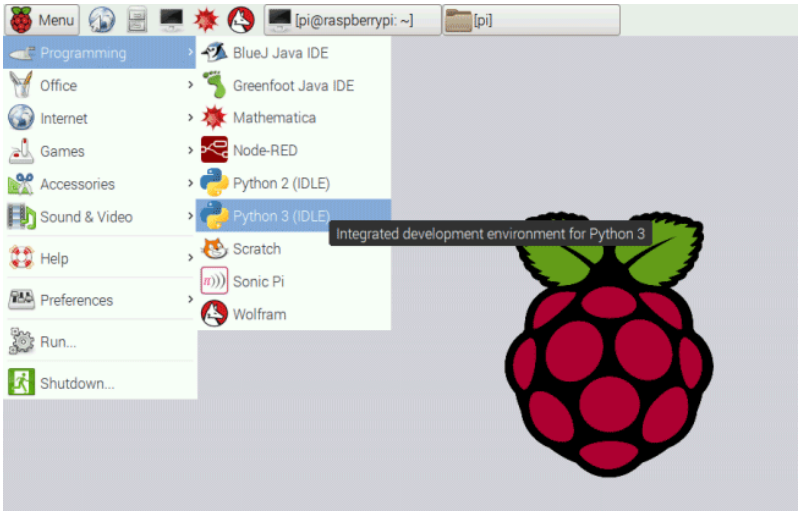
from each of these GPIO pins. So one should not tamper with these things unless you know what you are doing.

Circuit Diagram:

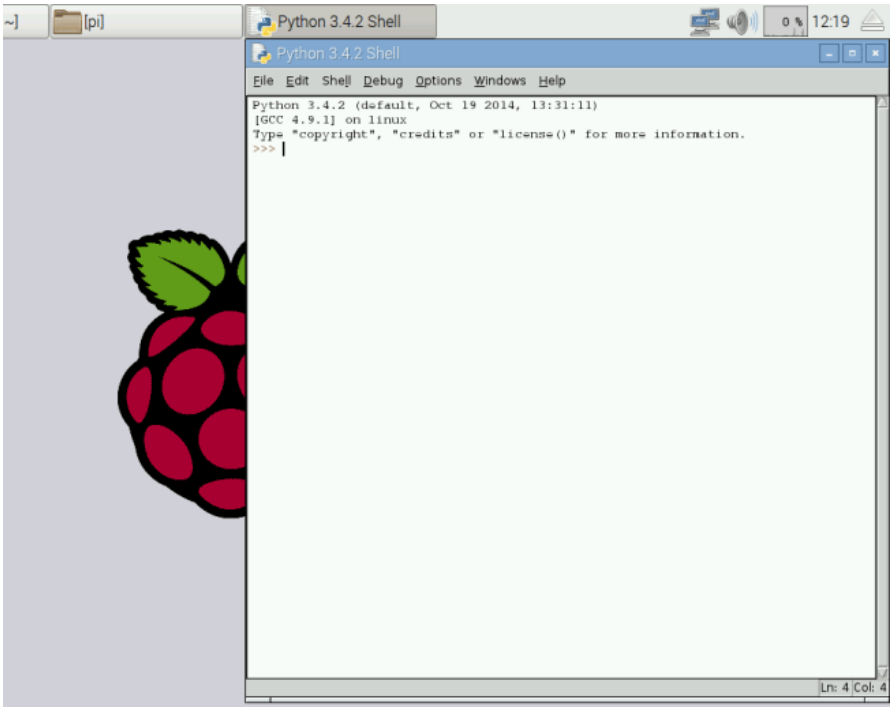


Steps:

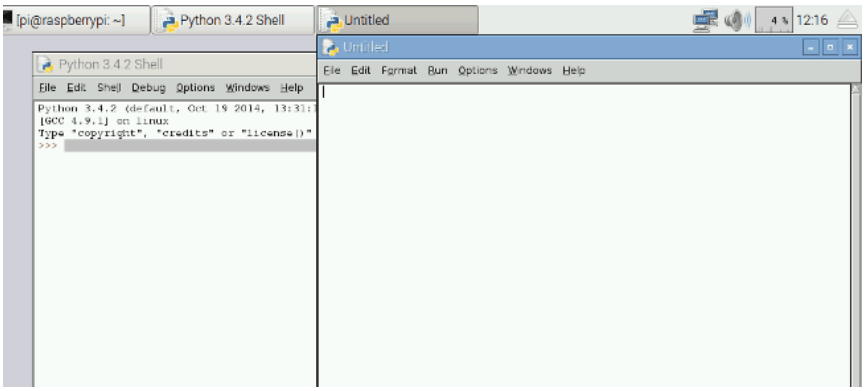
On the desktop, go the Start Menu and choose for the **PYTHON 3**, as shown in figure below.



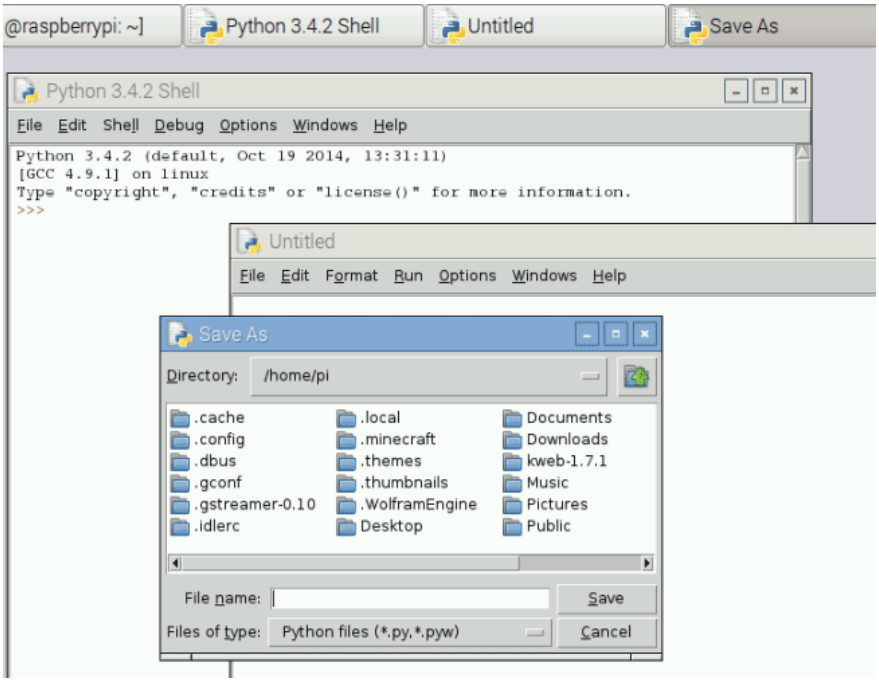
1. After that, PYHON will run and you will see a window as shown in below figure.



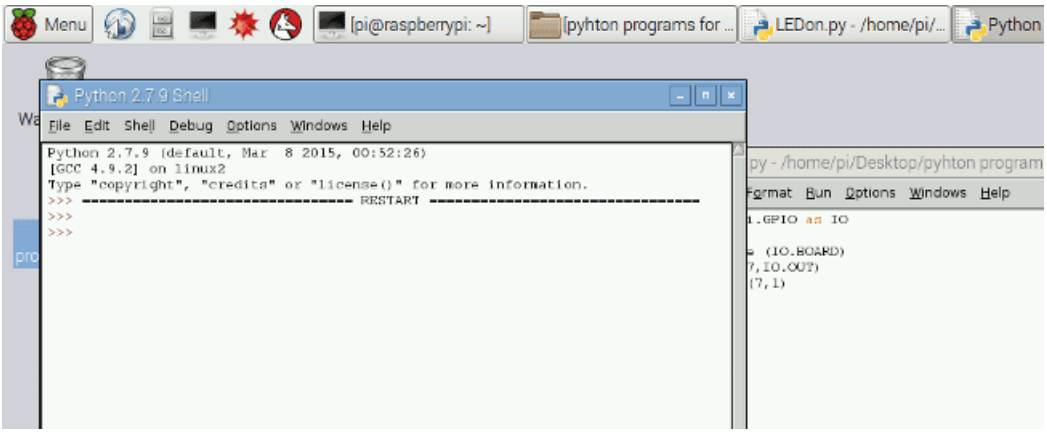
2. After that, click on *New File* in *File* Menu, You will see a new Window open,



3. Save this file as *blinky* on the desktop,



After that write the program for *blinky* as given below and execute the program by clickingon “RUN” on ‘DEBUG’ option.



If the program has no errors in it, you will see a “>>>”, which means the program is executed successfully. By this time you should see the LED blinking three times. If there were any errors in the program, the execution tells to correct it. Once the error is corrected execute the program again.

Program:

```
from
gpiozero
import LED
from time
import sleep

led =
LED(17)
```

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

Output:

Result:

Viva Questions:

1. Why do you think we must use the Raspberry Pi?
2. Are we likely to confront any overheating problems with Raspberry Pi?
3. How can you measure power consumption used by Raspberry Pi?
4. What are the generations of Raspberry Pi available?
5. What kind of projects have you done with Raspberry Pi?

Experiment 6

6. Collecting Sensor Data from DHT sensor interface

Aim: To Interface DHT11 Using Arduino Uno board and upload sensor data
.

Apparatus:

- Universal Board - 1
- Arduino board - 1
- Any sensor (DHT11) - 1
- WIFI Module - 1
- LCD - 1
- 12V Adaptor - 1
- Power jack - 1
- USB Cable - 1
- Jumper Wires Required

Hardware Procedure:

- LCD pins connected to Arduino Uno pin 2,3, 4, 5, 6, and 7.
- DHT11 pin connected to the 10 pin of Arduino board.
- Wifi module pins RX & TX are connected to 8 and 9 pin of Arduino (RX = 8, TX = 9).
- USB connector is connected to Arduino Uno to monitor.
- Place Wifi Module in IOT development Board.
- Connect the 12V power supply to development board.
- Power jack is connected to the Arduino Uno.
- Check the output from the development board.

Software Procedure:

1. Click on Arduino IDE
2. Click on file
3. Click on New
2. Write a Program as per circuit Pin connections
2. Click on Save
3. Create an Account in Things Speak, then create a channel.
4. Go to API keys in that Channel and then copy “Write API key”.
5. Go to Arduino code and paste Write API key.
6. Give your mobile hotspot name and password in Arduino code.
7. Click on save & Click on Verify.
8. Click on Upload the code into Arduino Uno by using USB cable.
9. After that open things Speak account and click on private view
10. DHT11 Sensor data will be uploaded and it will be shown as graph in Private view on ThingsSpeak account.

Program:

```
#include <dht.h>

#include <LiquidCrystal.h>

#include <SoftwareSerial.h>

LiquidCrystallcd(2,3, 4, 5, 6, 7);

SoftwareSerialwifi(8, 9); // TX, RX
```

```
String apiKey = "TRNIC1L9BXBXT322"; /// Write API Key
dhtDHT;
#define DHT11_PIN 10
const int buzzer = 13;

void setup(){
lcd.begin(16, 2);
pinMode(buzzer, OUTPUT);
digitalWrite(buzzer, 0);
project_Name();
Serial.begin(9600);
Serial.println("AT");
delay(1000);
Serial.println("AT+CMGF=1");
delay(1000);
Serial.println("AT+CNMI=2,2,0,0,0");
delay(1000);
lcd.setCursor(0,0);
lcd.print("WiFi module ");
lcd.setCursor(0,1);
lcd.print("Initilizing.... ");
wifi.begin(115200);
wifi.println("AT+RST");
delay(4000);
wifi.println("AT+CWMODE=3");
delay(4000);
wifi.print("AT+CWJAP=");
wifi.write('"');
wifi.print("STTMANI");
wifi.write('"');
wifi.write(',');
wifi.write('"');
wifi.print("hailucky123,./");
wifi.write('"');
wifi.println();
```

```
delay(1000);
lcd.setCursor(0,0);
lcd.print("WiFi module ");
lcd.setCursor(0,1);
lcd.print("Initilized..... ");
delay(1000);
lcd.clear();
}
void loop()
{
int chk = DHT.read11(DHT11_PIN);
//SendWiFi_Data();
//delay(1000);
lcd.setCursor(0,0);
lcd.print("Temperature: ");
lcd.setCursor(0,1);
lcd.print("Humidity: ");
lcd.setCursor(12,0);
lcd.print(DHT.temperature);
lcd.setCursor(9,1);
lcd.print(DHT.humidity);
delay(500);
/* Tempurature Data Process*/
if(DHT.temperature> 45)
{
buzzer_sound();
}
/* Humidity Data Process*/
if(DHT.humidity< 30)
{
buzzer_sound();
}
lcd.setCursor(15,1);
lcd.write(0x20);
SendWiFi_Data();
```

```
delay(1000);
}
void SendWiFi_Data(){
String cmd = "AT+CIPSTART=\"TCP\",\"";
cmd += "184.106.153.149"; // api.thingspeak.com
cmd += "\",80";
wifi.println(cmd);
delay(1500);
String getStr ="GET /update?api_key=";
getStr += apiKey;
getStr +="&field1=";
getStr += String(DHT.temperature);
getStr +="&field2=";
getStr += String(DHT.humidity);
getStr += "\r\n\r\n";
// send data length
cmd = "AT+CIPSEND=";
cmd += String(getStr.length());
wifi.println(cmd);
delay(1500);
wifi.println(getStr);
delay(1000);
}
void buzzer_sound()
{
digitalWrite(buzzer, HIGH);
delay(600);
digitalWrite(buzzer, LOW);
delay(400);
digitalWrite(buzzer, HIGH);
delay(600);
digitalWrite(buzzer, LOW);
delay(400);
}
void project_Name(){
```

```
lcd.setCursor(0,0);
lcd.print(" ESP8266 ");
lcd.setCursor(0,1);
lcd.print(" Interfacing ");
delay(3000);
lcd.clear();
}
```

Precautions:

1. Take care about given power supply (12V)
2. Jumper wires given carefully whenever given circuit connection

Output:

Result:

DHT11 sense the surrounding temperature and measure humidity in surrounding air that temperature and humidity shown by LCD display and Sensor data will be successfully uploaded on Things Speak account.

Viva Questions:

1. What is the purpose of the DHT11 sensor?
2. What is the working range of DHT11?
3. Which protocol is used for the DHT11 sensor?
4. What is the test of DHT11 sensor?