

Internet of Things



GUJARAT TECHNOLOGICAL UNIVERSITY



Government Engineering College, Bhavnagar

Subject: IOT and Applications (Internet of Things)

B.E. C.E. Semester-6th
(Computer Branch)

Submitted By:

Name: Vankani Arjun BakulBhai

Enrollment: 180210107060

Prof. Vishal Andodariya
(Faculty Guide)

Prof. KARSHAN KANDORIYA
(Head of the Department)
Academic Year (2020-21)

Index

SR No.	Algorithm (Lab work)	Page No
1	Study of Arduino and Raspberry-Pi Circuit	03
2	Study of Different Operating System for Raspberry-Pi/Arduino Board	08
3	Connect Raspberry Pi with your existing system components.	17
4	Controlling an LED Using Raspberry Pi.	23
5	Connect Arduino with your existing system components.	26
6	Controlling an LED using Arduino.	36
7	Write Program for RGB LED using Raspberry Pi.	43
8	Study the Temperature sensor and Write Program for monitor temperature using Arduino	46
9	Study and Implement RFID, NFC using Arduino.	50
10	Study and Implement Zigbee Protocol using Raspberry Pi	54

IOT and Applications (Internet of Things)

Practical-1: Study of Arduino and Raspberry-Pi Circuit

- Arduino and Raspberry are using for hardware to run programs whichever things we have to perform.
- Now we discussed both the topic below where and when we used this circuits as applicable.

1) Arduino



- Arduino is Hardware and Software company, project and user community that designs and manufactures single board Micro-Controllers and kits for building digital devices.
- Arduino board designs use a variety of Microprocessors and controllers.
- It's hardware products are licensed under a Creative commons license, while software is licensed under the GNU Lesser General Public License (LGPL) permitting the manufacture of Arduino boards and software distribution by anyone.
- The boards are equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards or breadboards and other circuits.
- The boards feature serial communications interfaces, including Universal Serial Bus (USB) on some models, which are also used for loading programs from personal computers.



Internet of Things

- The microcontrollers can be programmed using the C and C++ programming languages, using a standard API which is also known as the "Arduino language".
- In addition to using traditional compiler toolchains, the Arduino project provides an integrated development environment (IDE) and a command line tool (Arduino- cli) developed in Go.

➤ Application of Arduino:

- 1 Security System
- 2 Traffic Light Count Down Timer
- 3 Parking Lot Counter
- 4 Weighing Machines
- 5 Medical Instrument
- 6 Emergency Light for Railways

And many more IOT device that's uses sensor to program using Arduino.

➤ Advantage

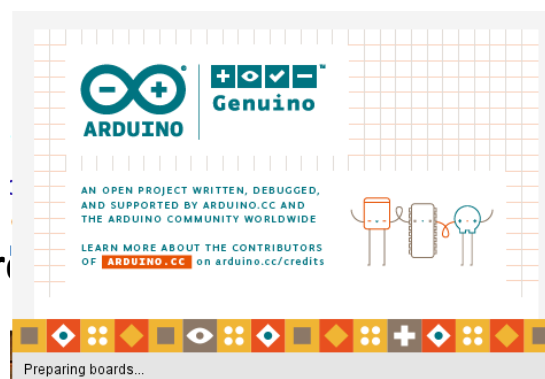
The biggest advantage of Arduino is its ready to use structure.

As Arduino comes in a complete package form which includes the 5V regulator, a burner, an oscillator, a micro-controller, serial communication interface, LED and headers for the connections.

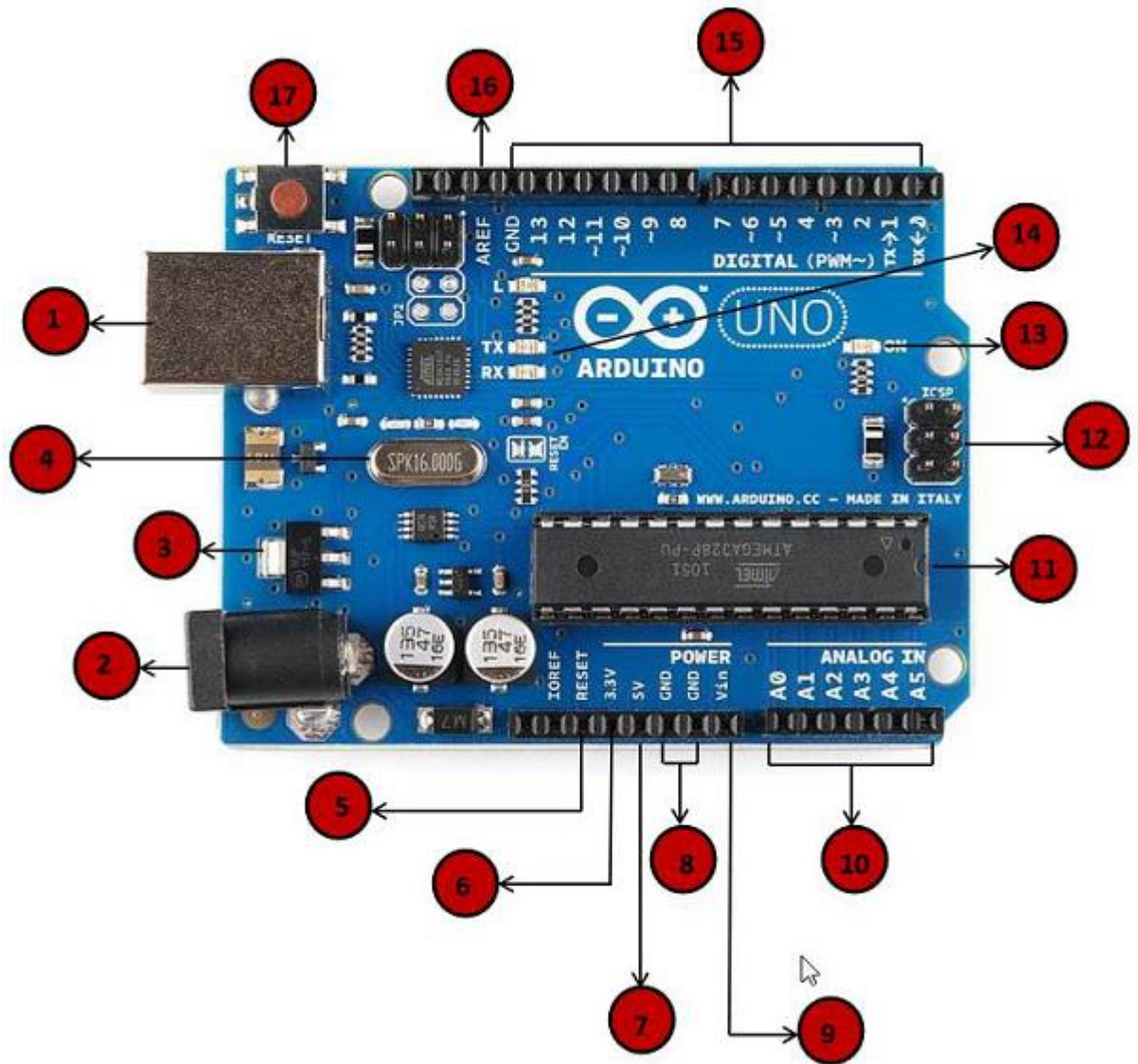
You don't have to think about programmer connections for programming or any other interface.

IDE

Arduino Circ



Internet of Things



Label -> Instructions

1-> Power USB

2-> Power (Barrel Jack)

3-> Voltage Regulator

4-> Crystal Oscillator

5,17-> Arduino Reset

6,7,8,9 -> Pins (Supply 3.3 output volt, Supply 5 output volt, (Ground), AC mains)

10-> Analog pins

Internet of Things

11-> Main microcontroller

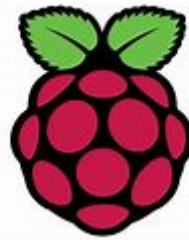
12-> ICSP pin

13-> Power LED indicator

14-> TX and RX LEDs

15-> Digital I/O

16-> AREF stands for Analog Reference



2) Raspberry Pi (Mini Computer)

- Raspberry Pi is a series of small single-board computers developed in the United Kingdom by the Raspberry Pi Foundation in association with Broadcom.
- Early on, the Raspberry Pi project leaned towards the promotion of teaching basic computer science in schools and in developing countries.
- Later, the original model became far more popular than anticipated, selling outside its target market for uses such as robotics.
- It is now widely used in many areas, such as for weather monitoring, because of its low cost, modularity, and open design.
- After the release of the second board type, the Raspberry Pi Foundation set up a new entity, named Raspberry Pi.
- The Foundation was rededicated as an educational charity for promoting the teaching of basic computer science in schools and developing countries.



Applications of Raspberry Pi

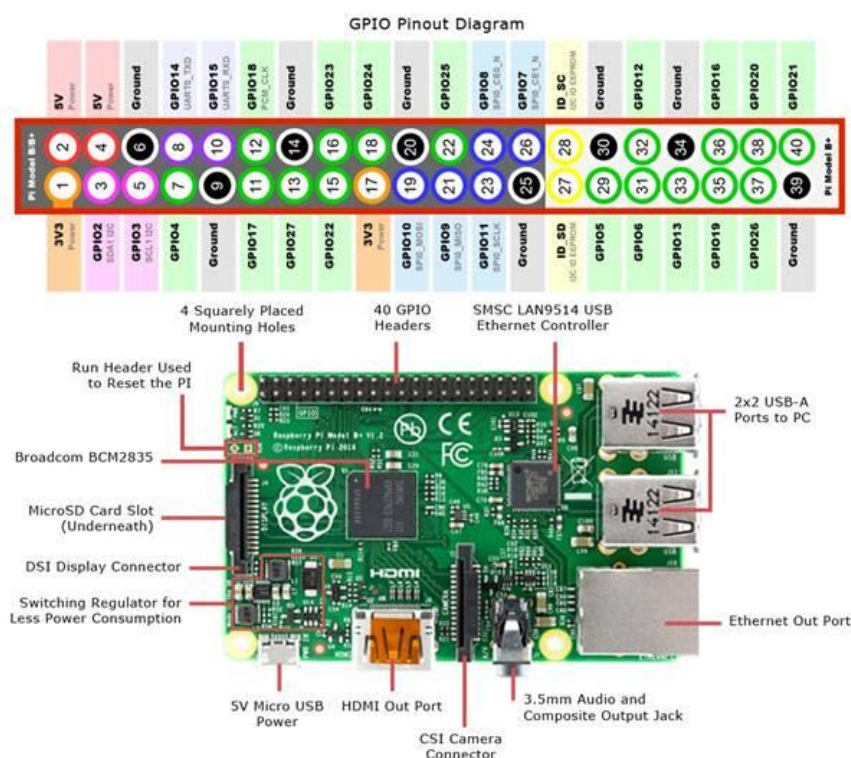
Internet of Things

- The raspberry pi boards are used in many applications like Media streamer, Arcade machine, Tablet computer, Home automation, Carputer, Internet radio, controlling robots, Cosmic Computer, hunting for meteorites, Coffee and also in raspberry pi-based projects.

Advantage of Raspberry Pi

- Low cost
- Huge processing power in a compact board
- Many interfaces (HDMI, multiple USB, Ethernet, onboard Wi-Fi and Bluetooth, many GPIOs, USB powered, etc.)
- Supports Linux, Python (making it easy to build applications)
- Readily available examples with community support
- Developing such an embedded board is going to cost a lot of money and effort.

➤ Raspberry Pi Circuit



IOT and Applications (Internet of Things)

Lab session-2

Practical-2: Study of Different Operating System for Raspberry-Pi/Arduino Board. Understanding the process of OS Installation on Raspberry-Pi/Arduino Board.

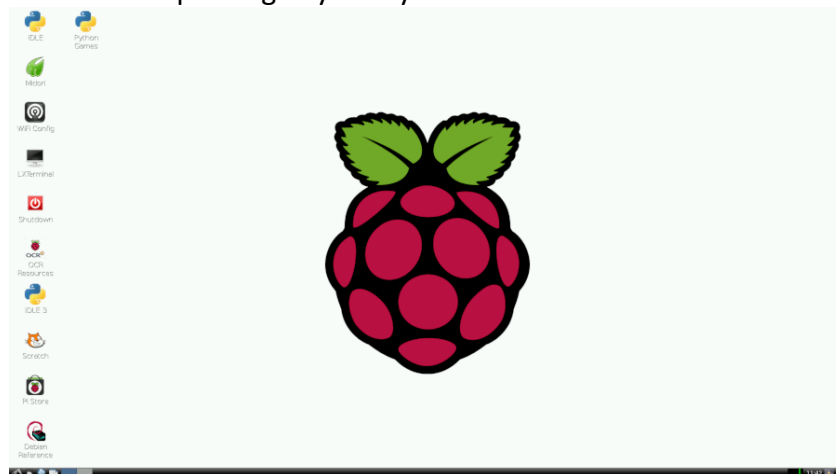
Raspberry-Pi Board

- The Raspberry Pi supports several OSES and as such usually comes without one. Most of the time, however, it ships with an SD card that includes NOOBS (New Out of the Box Software) – an OS that includes of a variety of Operating Systems from which you can choose which to or you to choose which to run on your Raspberry Pi setup.

❖ Different Operating System for Raspberry-Pi Board:

1) Raspbian

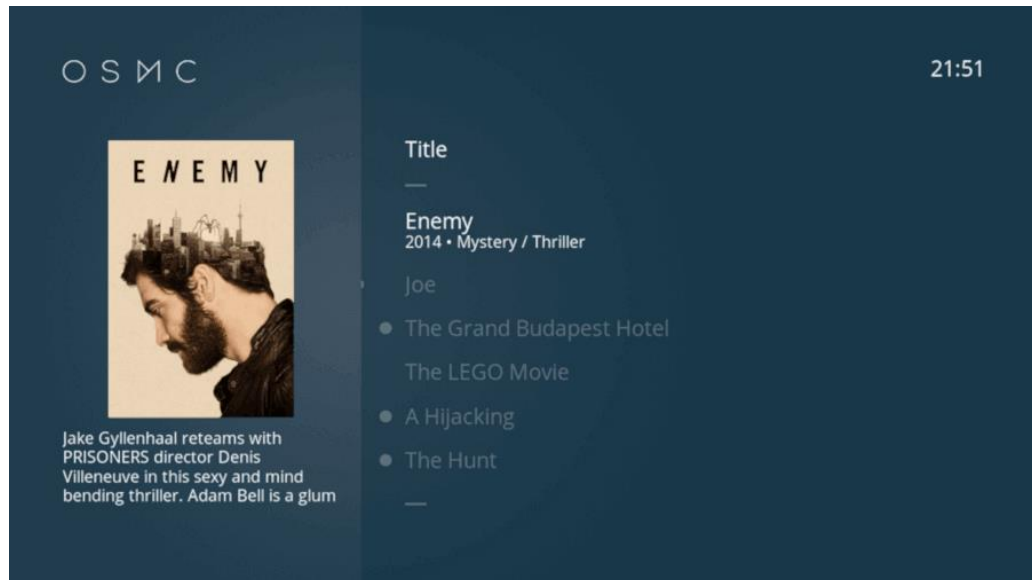
- Raspbian is a Debian-based engineered especially for the Raspberry Pi and it is the perfect general-purpose OS for Raspberry users.
- It employs the **Open box** stacking window manager and the **Pi Improved Xwindows Environment Lightweight** coupled with a number of pre-installed software which includes **Minecraft Pi**, **Java**, **Mathematica**, and **Chromium**.
- **Raspbian** is the Raspberry foundation's official supported OS and is capable of accomplishing any task you throw at it.



Internet of Things

2) OSMC

- OSMC (Open-Source Media Center) is a free, simple, open-source, and easy-to-use standalone Kodi OS capable of playing virtually any media format.
- It features a modern beautiful minimalist User Interface and is completely customizable thanks to the several built-in images that it comes with. Choose OSMC if you run the Raspberry Pi for managing media content.



3) Open ELEC

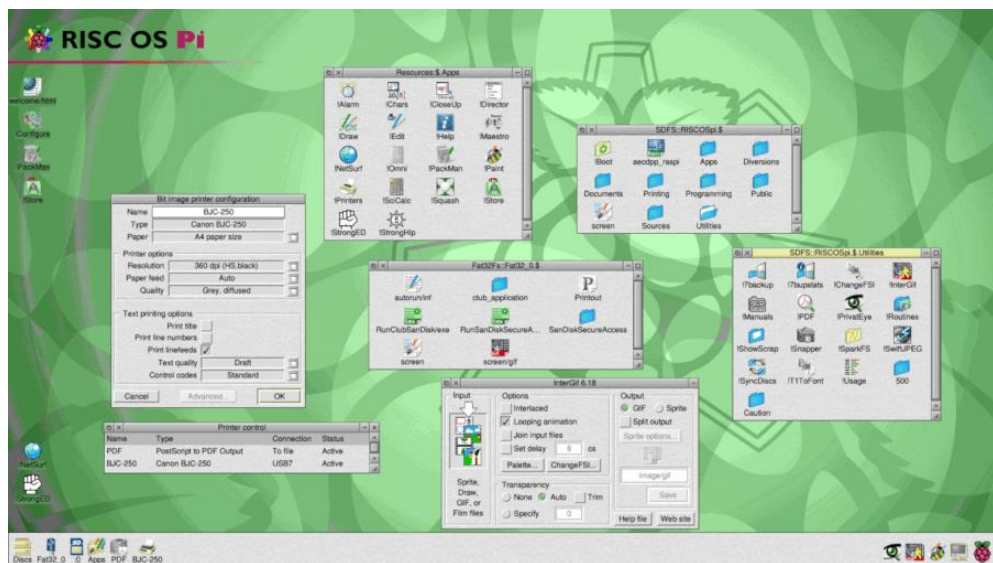
- Open ELEC (Open Embedded Linux Entertainment Center) is a small Linux-based **JeOS (Just enough Operating System)** developed from scratch to turn PCs into a **Kodi** media center.
- You can think of Open ELEC as a barebones Kodi as it has fewer customization options and limits access to certain areas e.g. SSH and it is more complex to customize.
- Nevertheless, Open ELEC is a powerful media center that might suit your needs if OSMC doesn't.



Internet of Things

4) RISC OS

- RISC OS is a unique open-source OS designed specifically for ARM processors by the creators of the original ARM. It is neither related to Linux nor Windows and is being maintained by a dedicated community of volunteers.
- If you want to choose RISC OS, you should know that it is very different from any Linux distro or Windows OS you have used so it will take some getting used to.



5) Windows IOT Core

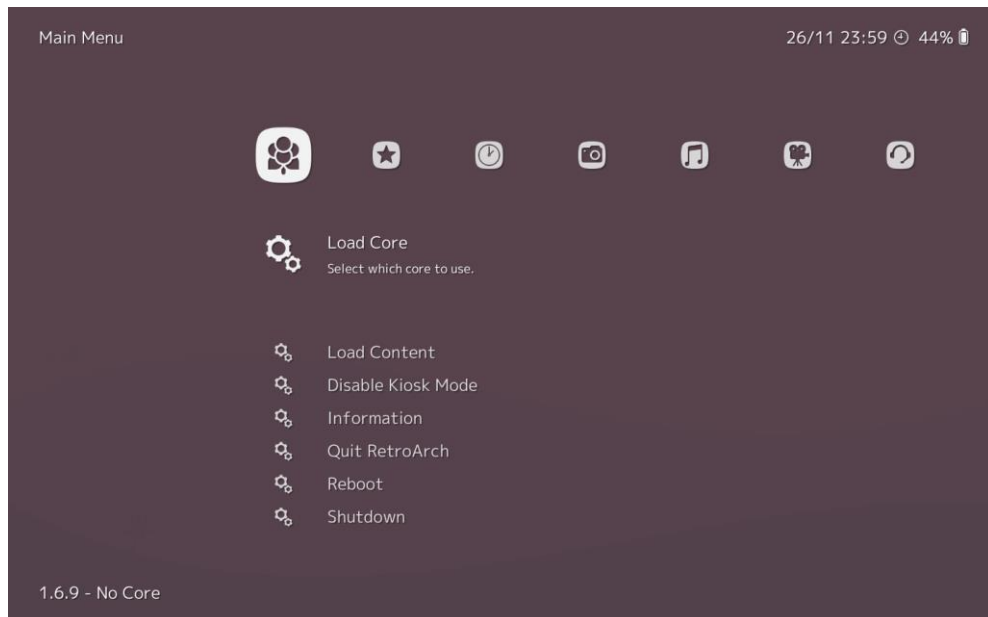
Windows IoT Core is a Windows OS built especially for the Raspberry Pi as a development platform for programmers and coders. Its aim is for programmers to use it to build prototypes of IoT devices using the **Raspberry Pi** and **Windows 10**.



Internet of Things

6) Lakka

- Lakka is a free, lightweight, and open-source distro with which you can turn even the smallest PC into a full-blown game console without the need for a keyboard or mouse.
- It features a beautiful User Interface and so many customization options you might get overwhelmed. Its PS4-like UX brings style to the Raspberry Pi so pick it if you're a gamer.



7) Rasp BSD

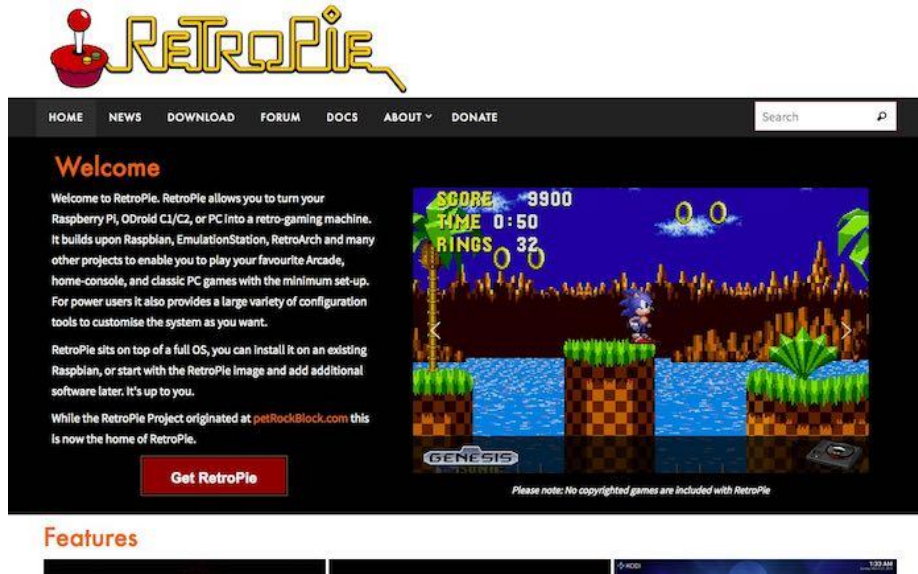
- Rasp BSD is a free and open-source image of **FreeBSD 11** that has been preconfigured in 2 images for Raspberry Pi computers.
- If you didn't know, **FreeBSD** isn't Linux, but it works in pretty much the same way as it is a descendant of the research by the **Berkeley Software Distribution** and it is among the world's most broadly used Operating Systems today with its code existing in-game consoles e.g. **PlayStation 4**, **macOS**, etc.



Internet of Things

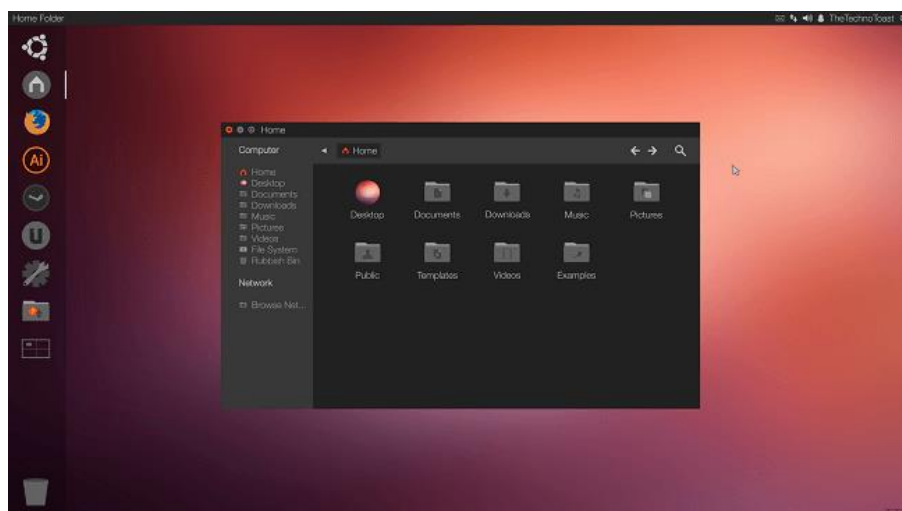
8) Retro Pie

- RetroPie is an open-source Debian-based software library with which you can emulate retro games on your **Raspberry Pi**, **PC**, or **ODroid C1/C2** and it currently stands as the most popular option for that task.
- **RetroPie** used the **Emulation Station** frontend and **SBC** to offer users a pleasant retro gaming experience so you can't go wrong with it.



9) Ubuntu Core

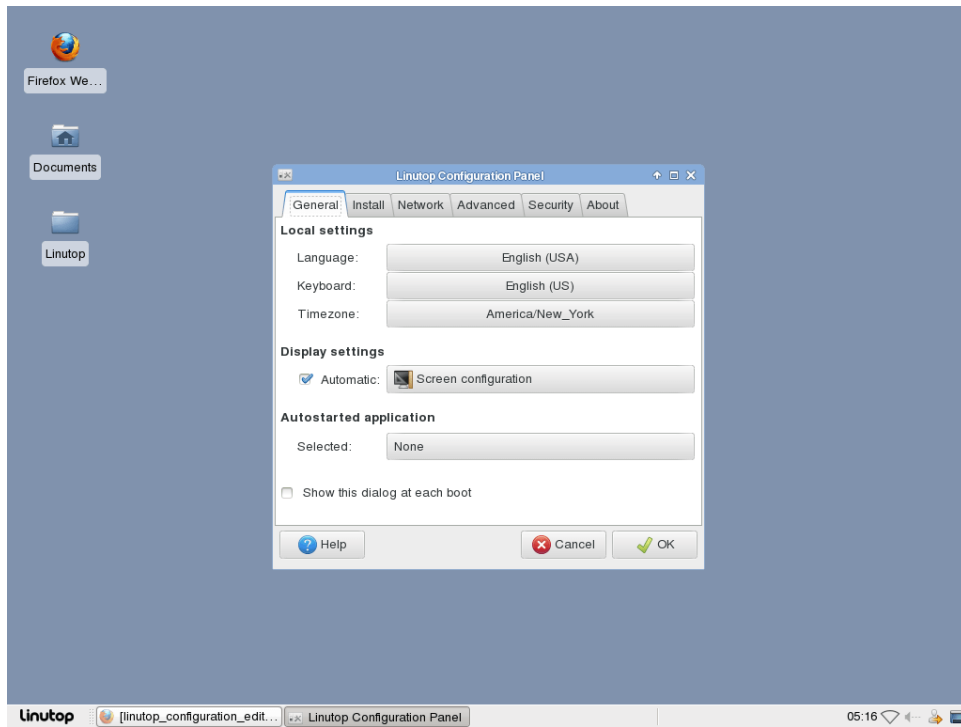
- Ubuntu Core is the version of Ubuntu designed for Internet of Things applications. Ubuntu is the most popular Linux-based Operating System in the world with over 20+ derivatives and given that it has an active and welcoming forum, it will be easy to get up and running with Ubuntu Snappy Core on your Raspberry Pi.



Internet of Things

10) Linutop

- Linutop OS is a secure Raspbian-based Web Kiosk and digital signage player. It is dedicated to professionals with the need to deploy public Internet stalls and digital signage solutions using Raspberries.
- This OS is perfect if you run hotels, restaurants, shops, city halls, offices, museums, etc. and it is compatible with Raspberry Pi B, B+ and 2.



Arduino Board



Arduino IDE is designed to run well on Windows 10, macOS, and Linux. However, in contrast to Raspberry Pi, which is a fully-fledged computer, Arduino runs as a single-board microcontroller. Therefore, a real time

Internet of Things

operating system (RTOS) is preferred in actual Arduino projects since it has a smaller footprint, better control over the tiny peripherals, and no buffering delays.

❖ Different Operating System for Arduino Board:

1) Free RTOS

- Some of the advantages of Free RTOS include over the air (OTA) updates, an exhaustive collection of IoT libraries, managing data sharing and hardware resources across multiple tasks, and more predictable memory use.

```

StructQueue | Arduino 1.8.9
File Edit Sketch Tools Help

StructQueue

/*
 * Example of a basic FreeRTOS queue
 * https://www.freertos.org/Embedded-RTOS-Queues.html
 */

// Include Arduino FreeRTOS library
#include <Arduino_FreeRTOS.h>

// Include queue support
#include <queue.h>

// Define a struct
struct pinRead {
    int pin;
    int value;
};

/*
 * Declaring a global variable of type QueueHandle_t
 */
QueueHandle_t structQueue;

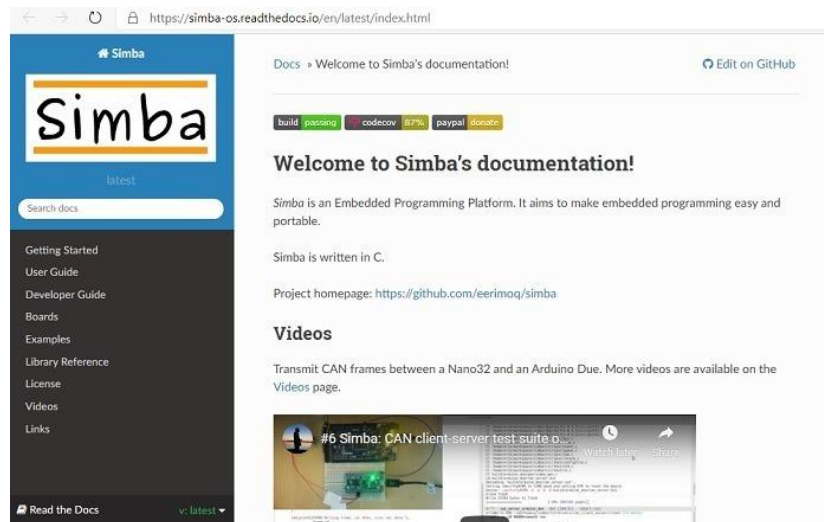
void setup() {

    /*
     * Create a queue.
     * https://www.freertos.org/a00116.html
     */
  
```

2) Simba

- If you want extensive support for all kinds of Arduino boards on a dedicated embedded platform, Simba offers a brilliant option. Not only does it support Arduino Uno and Mega, but also Zero, Due, Nano32, Pro Micro, and more.
- Among the advantages of Simba are a simple shell design, fast debugging, and a fairly extensive standard library that is comprised of a vast range of functions from USB to Math, sensors, and global navigation satellite systems.

Internet of Things



3) Trampoline

- In embedded industry parlance, a trampoline refers to short snippets of code which execute other lines of code. This has inspired a no-frills static RTOS for small embedded systems called Trampoline. It is available as a GitHub project and runs on Arduino Uno and Mega boards.
- The main advantages of Trampoline include real time predictability, support for very low RAM (32 kB), ROM (128 kB), and CPU (16 bit). What this means is that Trampoline has been designed for tiny cyber physical systems such as miniature drones and digital controllers such as brakes.

4) Duin OS

Based on Free DOS, an open-source operating system, Duin OS is an RTOS designed exclusively for Arduino boards. It currently supports Free RTOS-based embedded projects which are available as a GitHub repository.

The advantages of Duin OS include basic multitasking and multi-threading, low RAM and CPU requirements and efficient signaling.

Understanding the process of OS Installation on Raspberry-pi/Arduino Board.

➤ HOW TO INSTALL RASPBIAN OS IN YOUR RASPBERRY PI:

Step 1: Download the Required Software and Files.

Step 2: Get the SD Card and the Card Reader.

Step 3: Check the Drive in Which the SD Card Is Mounted.

Internet of Things

Step 4: Format the SD Card.

Step 5: Write the **OS** on the SD Card.

Step 6: Eject the SD Card.

➤ **HOW TO INSTALL OS IN YOUR ARDUINO:**

Step 1: On your host computer, goto <https://create.arduino.cc/>. Click Getting Started.

Step 2: Click Set up a generic Intel® IoT Platform.

Step 3: If you already have an Arduino Create account, provide your login credentials. If you don't, see the note below to create a new account.

Step 4: Click I'M UNSURE IF AN OS IS CURRENTLY INSTALLED ON MY DEVICE.

Step 5: Enter a new user name and password to use to log in to your target platform.

Step 6: If you need to specify proxy settings, click Set it up here and provide any required proxy information. Click DONE.

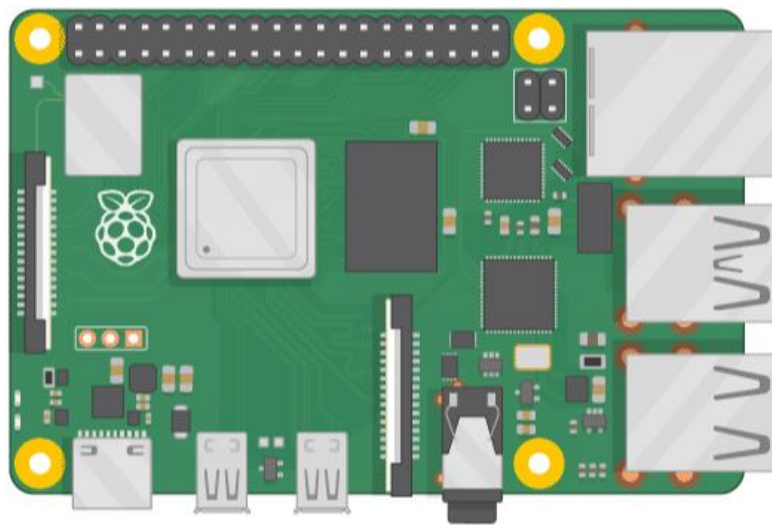
Step 7: When finished, click NEXT to continue with the setup.

Step 8: Select Ubuntu to install on your target platform, then click Download OS.

Internet of Things

+ Practical 3: Connect Raspberry Pi with your existing system components.

- The Raspberry Pi is a small computer that can do lots of things. You plug it into a monitor and attach a keyboard and mouse.



- We list down all the hardware can be joined with Raspberry Pie circuit and Used OS within the SD card.

Hardware

- A Raspberry Pi computer with an SD card or micro-SD card
- A monitor with a cable (and, if needed, an HDMI adaptor)
- A USB keyboard and mouse
- A power supply
- Headphones or speakers (optional)

Internet of Things

- An ethernet cable (optional)

Software

- Raspberry Pi OS, installed using the Raspberry Pi Imager

❖ Install Raspberry Pi OS on your SD card with the Raspberry Pi Imager

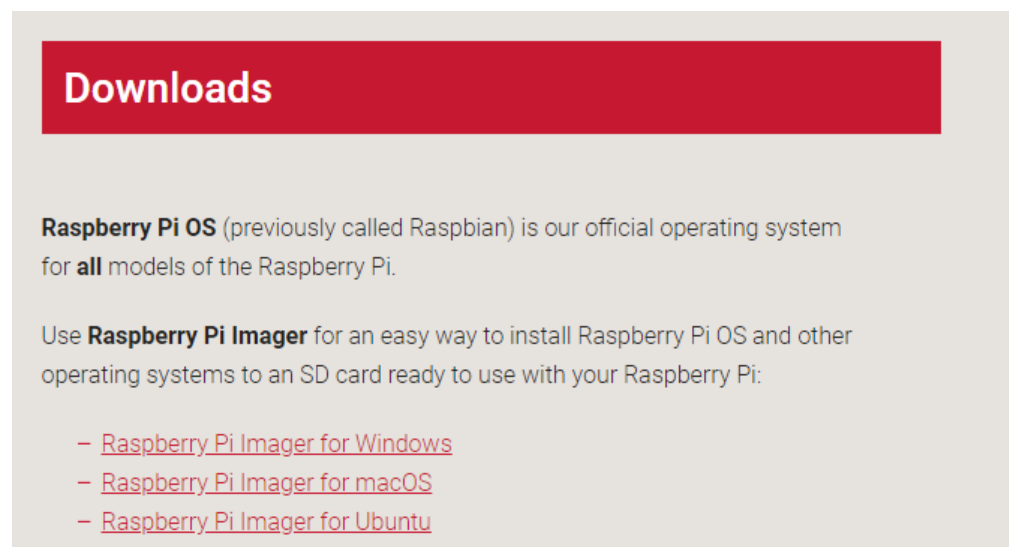
Many vendors sell SD cards with a simple Raspberry Pi OS installer called NOOBS preinstalled but you can really easily install Raspberry Pi OS yourself using a computer that has an SD card port or using an SD card reader.

Using the Raspberry Pi Imager is the easiest way to install Raspberry Pi OS on your SD card.

installing operating system images

Download and launch the Raspberry Pi Imager

- Visit the [Raspberry Pi downloads page](#).
- Click on the link for the Raspberry Pi Imager that matches your operating system.

A screenshot of the Raspberry Pi Downloads page. At the top, there is a red header with the word "Downloads" in white. Below this, the text reads: "Raspberry Pi OS (previously called Raspbian) is our official operating system for all models of the Raspberry Pi." This is followed by: "Use Raspberry Pi Imager for an easy way to install Raspberry Pi OS and other operating systems to an SD card ready to use with your Raspberry Pi:". At the bottom, there are three links, each preceded by a red dash: "[Raspberry Pi Imager for Windows](#)", "[Raspberry Pi Imager for macOS](#)", and "[Raspberry Pi Imager for Ubuntu](#)".

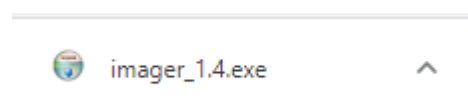
Downloads

Raspberry Pi OS (previously called Raspbian) is our official operating system for **all** models of the Raspberry Pi.

Use **Raspberry Pi Imager** for an easy way to install Raspberry Pi OS and other operating systems to an SD card ready to use with your Raspberry Pi:

- [Raspberry Pi Imager for Windows](#)
- [Raspberry Pi Imager for macOS](#)
- [Raspberry Pi Imager for Ubuntu](#)

- When the download finishes, click on it to launch the installer.



Internet of Things

Using the Raspberry Pi Imager

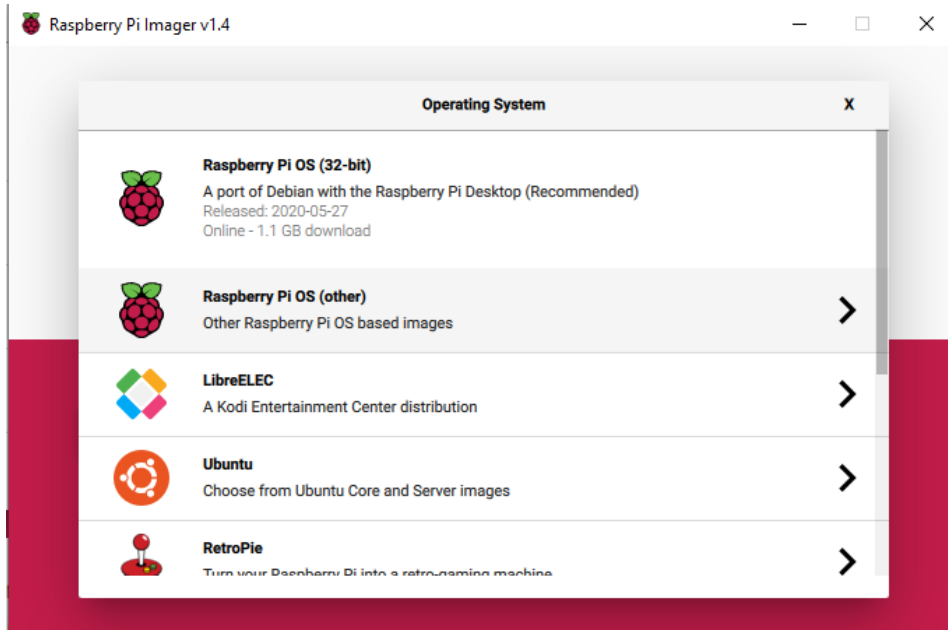
All data stored on the SD card will be overwritten during formatting and lost permanently, so make sure that you back up the card or any files you want to keep before running the installer.

When you launch the installer, your operating system may try to block you from running it. For example, Windows may give the following message:



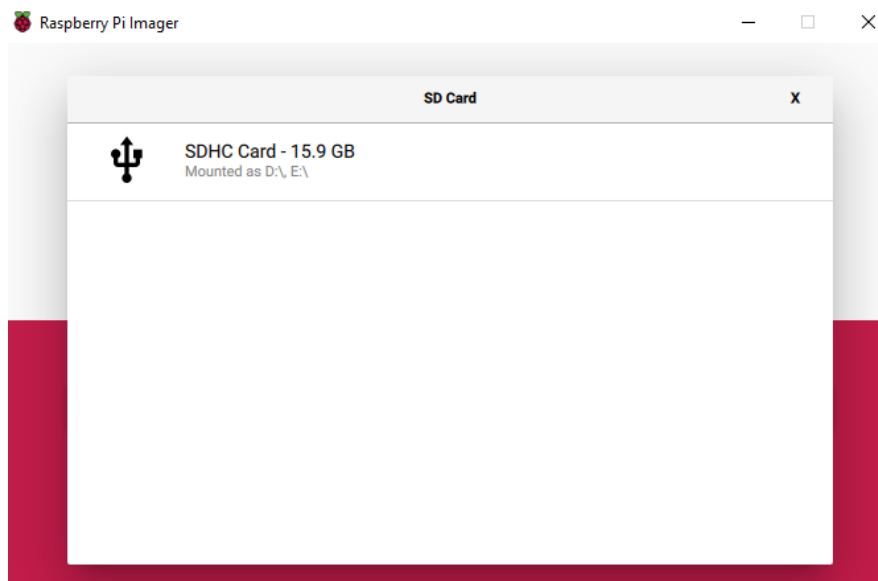
- If you get this, click on **More info** and then **Run anyway**.
- Insert your SD card into the computer or laptop's SD card slot.
- In the Raspberry Pi Imager, select the OS that you want to install. The first option, Raspberry Pi OS, is the recommended OS.

Internet of Things



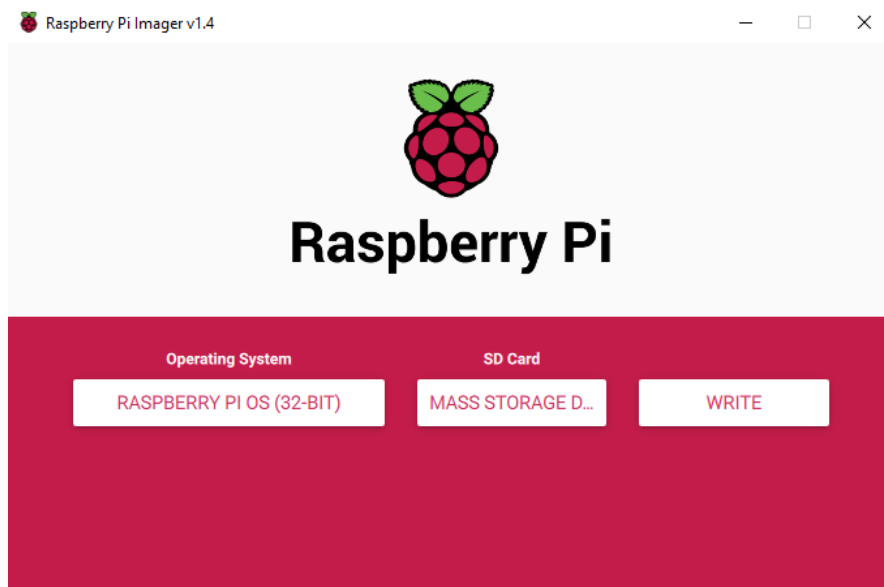
- Select the SD card you would like to install it on. Different platforms will display the drives in different ways. Mac OS, for example, will show you all drives including you main operating system.

Note: Make sure you are selecting the correct drive. The drives memory capacity can be a useful indication of which drive you are selecting.

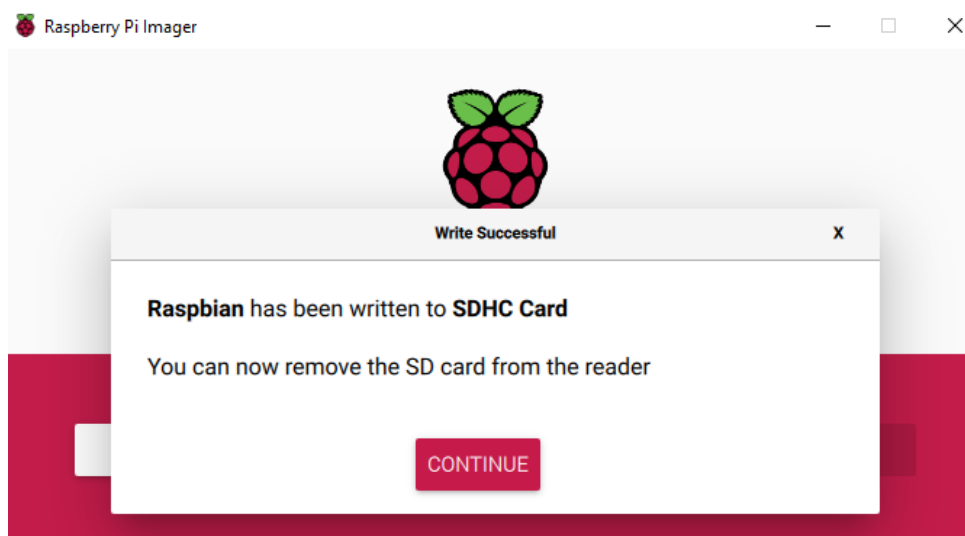


Once you have selected both the OS and the SD card, a new **WRITE** button will appear.

Internet of Things



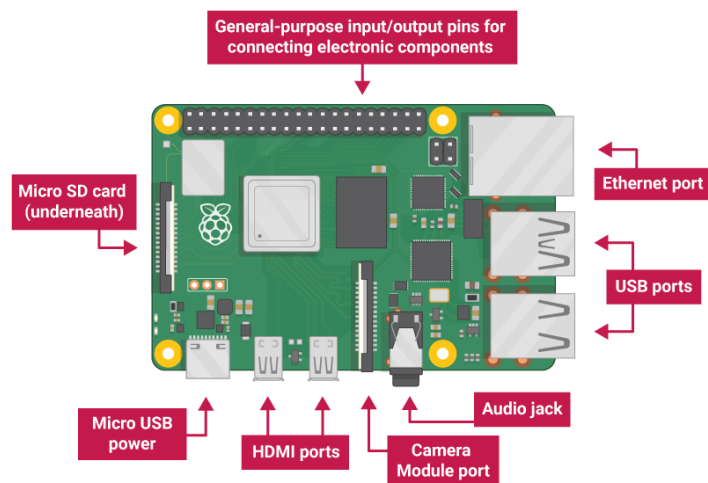
- Then simply click the **WRITE** button.
- Wait for the Raspberry Pi Imager to finish writing.
- Once you get the following message, you can eject your SD card.



Internet of Things



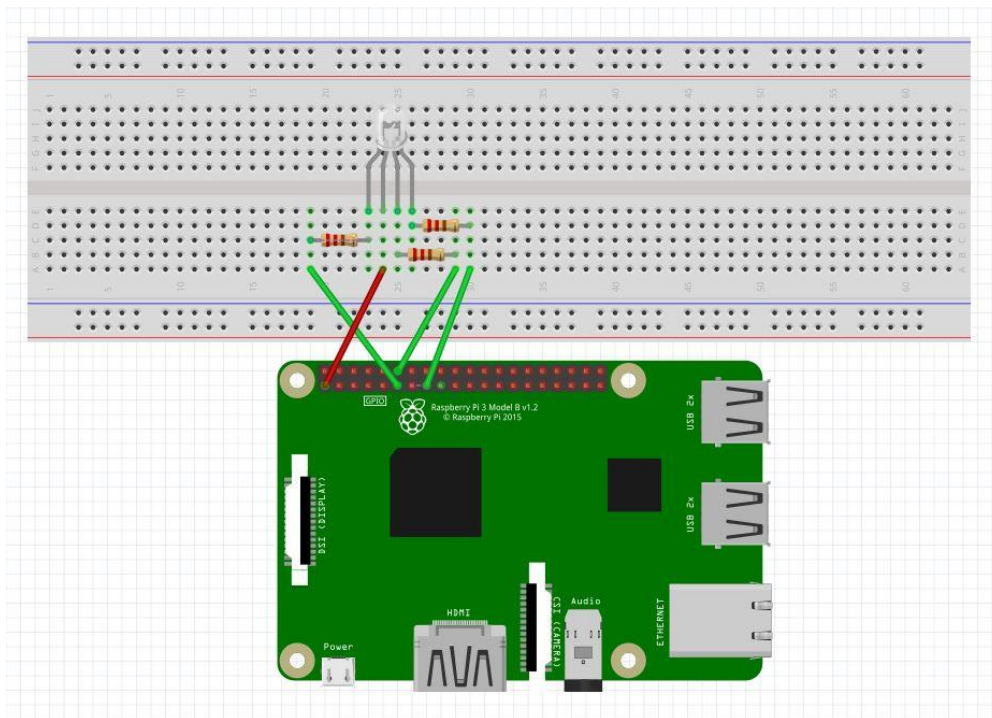
Circuit Diagram



Internet of Things

✚ Practical No 4: Controlling an LED Using Raspberry Pi.

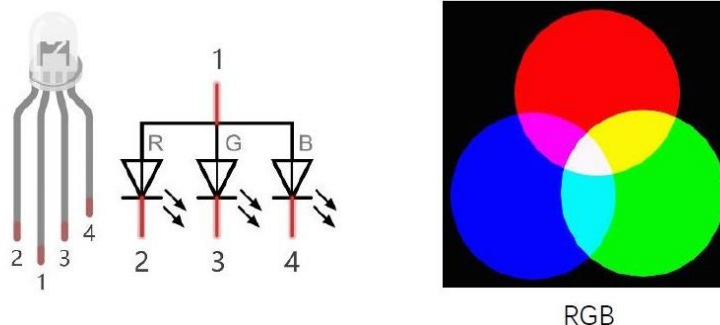
- We have to connect the Light Emitted Diode (LED) with bread-board and hard wired with Programmable Raspberry circuit. It will perform our code and blinks at interval like 5 seconds.



Main architecture

- As figure show, we connect register and LED light and Raspberry input port and Ground and Circuit switches.

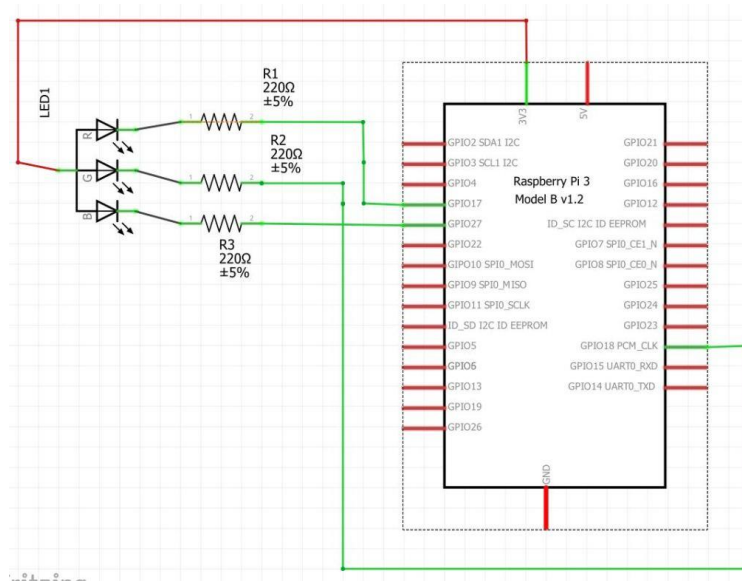
○ I want to see RGB Colour as(RED GREEN BLUE)



■ RGB LED

Internet of Things

- After this we Show pins of Raspberry GPIO17 and GPIO27 to switch with Register



Circuit Diagram

Python Script to Control RGB LED with Raspberry Pi

- Since each of the LED in an RGB LED is controlled via PWM, we will control 3 GPIO pins to generate PWM Signal.
- First, initialize all of the utilized GPIO resources as a PWM channel.
We **defined a tuple named pins** to abstract the GPIO pins that correspond to the RGB pins of the LED. Also, since this is a Common Anode RGB LED, **make sure you set the initial signal too HIGH to turn off the LED.**
- We will be displaying different colours for each second in this script. Its also a great practice to keep your code simple by defining functions such as **setColor()**. This function sets the duty cycle for each of the PWM channels.

```

From gpiozero import LED # It is for just one colour LED
Import time
K = input ("Sleep Time:")
Led = LED (18)
While True:
    led.toggle()
    time.sllep(k)
    led.toggle()
    time.sleep(k)
  
```

CODE

```

import RPi.GPIO as GPIO

import time
import random

pins = (11,12,13) # R = 11, G = 12, B = 13

def setup():
    global pwmR, pwmG, pwmB
    GPIO.setmode(GPIO.BOARD)
    for i in pins: # iterate on the RGB pins, initialize each and set to
HIGH to turn it off (COMMON ANODE)
        GPIO.setup(i, GPIO.OUT)
        GPIO.setup(i, GPIO.HIGH)
    pwmR = GPIO.PWM(pins[0], 2000) # set each PWM pin to 2 KHz
    pwmG = GPIO.PWM(pins[1], 2000)
    pwmB = GPIO.PWM(pins[2], 2000)
    pwmR.start(0) # initially set to 0 duty cycle
    pwmG.start(0)
    pwmB.start(0)

def setColor(r, g, b): # 0 ~ 100 values since 0 ~ 100 only for duty cycle
    pwmR.ChangeDutyCycle(r)
    pwmG.ChangeDutyCycle(g)
    pwmB.ChangeDutyCycle(b)

def displayColors():
    setColor(100, 0, 0) # red color
    time.sleep(1) # 1s
    setColor(0, 100, 0) # green
    time.sleep(1) # 1s
    setColor(0, 0, 100) # blue
    time.sleep(1) # 1s
    setColor(100, 100, 0) # yellow
    time.sleep(1) # 1s
    setColor(0, 100, 100) # cyan
    time.sleep(1) # 1s
    setColor(100, 0, 100) # magenta
    time.sleep(1) # 1s
    setColor(50, 0, 0) # maroon
    time.sleep(1) # 1s
    setColor(50, 0, 50) # purple
    time.sleep(1) # 1s
    setColor(0, 0, 50) # navy
    time.sleep(1) # 1s

def destroy():
    pwmR.stop()
    pwmG.stop()
    pwmB.stop()
    GPIO.cleanup()

if __name__ == '__main__':
    setup()
    displayColors()
    destroy()

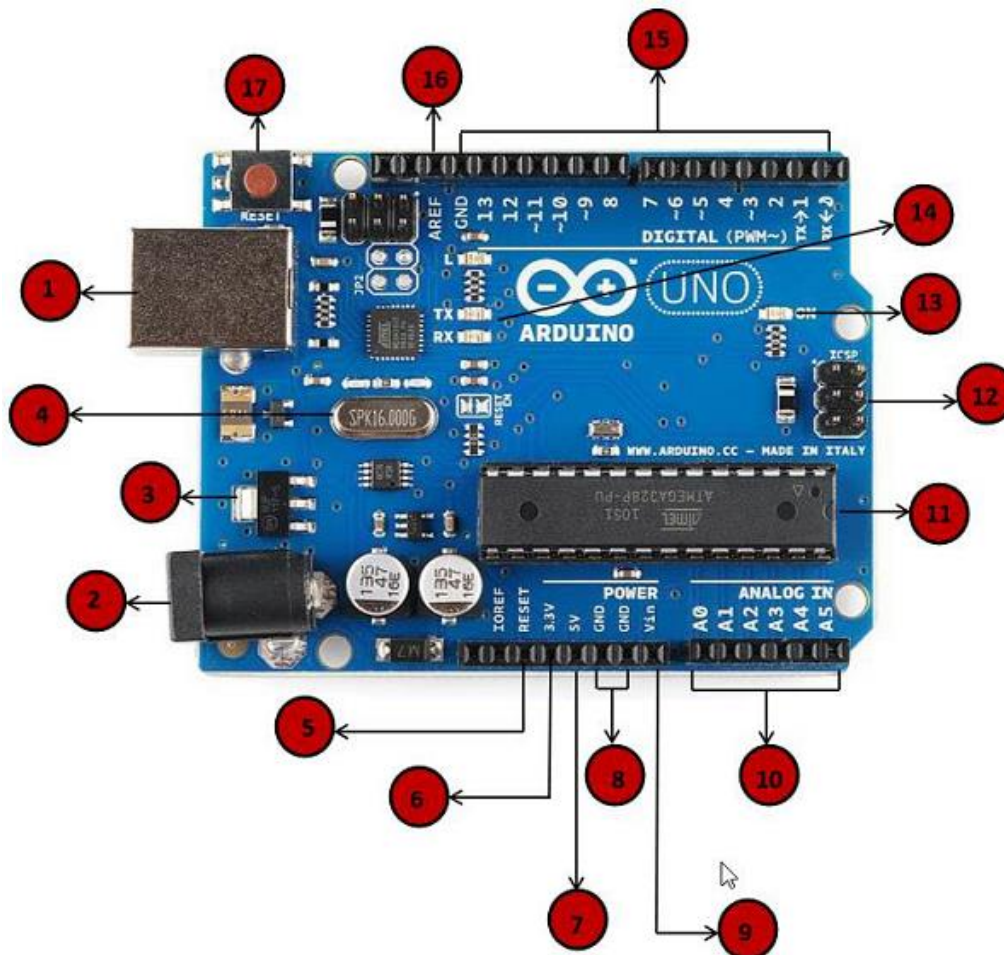
```

Internet of Things

✚ Practical No 5: Connect Arduino with your existing system components.

➤ Circuit

- There are different components on the Arduino board. Arduino UNO board is the most popular board in the Arduino board family.
- It is the best board to get started with electronics and coding. Some boards look a bit different from the one given below, but most Arduinos have majority of these components in common.



1) Power USB(No.1):

- Arduino board can be powered by using the USB cable from your computer. All you need to do is connect the USB cable to the USB connection (1).

Internet of Things

2) Power (Barrel Jack) (No.2):

- Arduino boards can be powered directly from the AC mains power supply by connecting it to the Barrel Jack (2).

3) Voltage Regulator (No.3):

- The function of the voltage regulator is to control the voltage given to the Arduino board and stabilize the DC voltages used by the processor and other elements.

4) Crystal Oscillator (No.4):

- The crystal oscillator helps Arduino in dealing with time issues. How does Arduino calculate time? The answer is, by using the crystal oscillator. The number printed on top of the Arduino crystal is 16.000H9H. It tells us that the frequency is 16,000,000 Hertz or 16 MHz

5) Arduino Reset (No.5,17):

- You can reset your Arduino board, i.e., start your program from the beginning. You can reset the UNO board in two ways. First, by using the reset button (17) on the board. Second, you can connect an external reset button to the Arduino pin labeled RESET (5).

6) Pins (3.3, 5, GND, V_{in}) (No.6,7,8,9):

- 3.3V (Pin-6) – Supply 3.3 output volt
- 5V (Pin-7) – Supply 5 output volt
- Most of the components used with Arduino board works fine with 3.3 volt and 5 volt.
- GND (Pin-8)(Ground) – There are several GND pins on the Arduino, any of which can be used to ground your circuit.
- V_{in} (Pin-9) – This pin also can be used to power the Arduino board from an external power source, like AC mains power supply.

7) Analog pins (No.10):

- The Arduino UNO board has six analog input pins A0 through A5. These pins can read the signal from an analog sensor like the humidity sensor

Internet of Things

or temperature sensor and convert it into a digital value that can be read by the microprocessor.

8) Main Microcontroller (No.11):

- Each Arduino board has its own microcontroller (11). You can assume it as the brain of your board. The main IC (integrated circuit) on the Arduino is slightly different from board to board. The microcontrollers are usually of the ATMEL Company. You must know what IC your board has before loading up a new program from the Arduino IDE. This information is available on the top of the IC. For more details about the IC construction and functions, you can refer to the data sheet.

9) ICSP pin (No.12):

- Mostly, ICSP (12) is an AVR, a tiny programming header for the Arduino consisting of MOSI, MISO, SCK, RESET, VCC, and GND. It is often referred to as an SPI (Serial Peripheral Interface), which could be considered as an "expansion" of the output. Actually, you are slaving the output device to the master of the SPI bus.

10) Power LED indicator (No.13):

- This LED should light up when you plug your Arduino into a power source to indicate that your board is powered up correctly. If this light does not turn on, then there is something wrong with the connection.

11) TX and RX LEDs (No.14):

- On your board, you will find two labels: TX (transmit) and RX (receive). They appear in two places on the Arduino UNO board. First, at the digital pins 0 and 1, to indicate the pins responsible for serial communication. Second, the TX and RX led (13). The TX led flashes with different speed while sending the serial data. The speed of flashing depends on the baud rate used by the board. RX flashes during the receiving process.

12) Digital I/O(No.15):

- The Arduino UNO board has 14 digital I/O pins (15) (of which 6 provide PWM (Pulse Width Modulation) output. These pins can be configured to work as input digital pins to read logic values (0 or 1) or as digital output

Internet of Things

pins to drive different modules like LEDs, relays, etc. The pins labeled “~” can be used to generate PWM.

13) AREF(No.16):

- AREF stands for Analog Reference. It is sometimes, used to set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pins.

○ Installation:

- After learning about the main parts of the Arduino UNO board, let's learn how to set up the Arduino IDE.

➤ Steps:

1. First you must have your Arduino board (you can choose your favorite board) and a USB cable. In case you use Arduino UNO, Arduino Duemilanove, Nano, Arduino Mega 2560, or Decimal, you will need a standard USB cable (A plug to B plug), the kind you would connect to a USB printer as shown in the following image.



- In case you use Arduino Nano, you will need an A to Mini-B cable instead as shown in the following image.

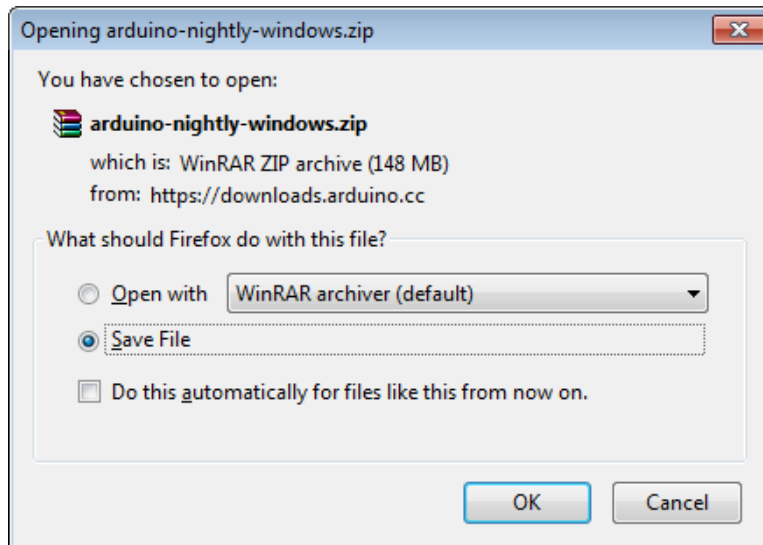


2. **Download Arduino IDE Software:**

- You can get different versions of Arduino IDE from the Download page on the Arduino Official website. You must select your

Internet of Things

software, which is compatible with your operating system (Windows, IOS, or Linux). After your file download is complete, unzip the file.



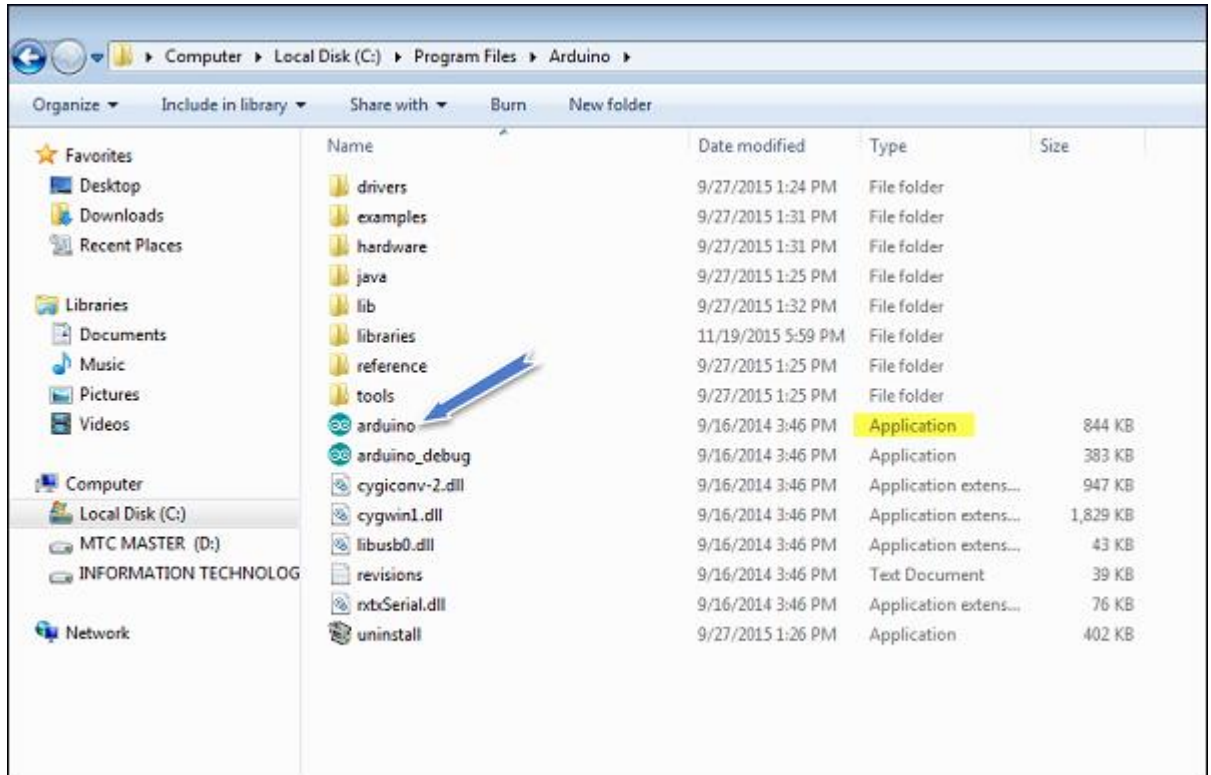
3. Power up your board:

- The Arduino Uno, Mega, Duemilanove and Arduino Nano automatically draw power from either, the USB connection to the computer or an external power supply. If you are using an Arduino Diecimila, you have to make sure that the board is configured to draw power from the USB connection. The power source is selected with a jumper, a small piece of plastic that fits onto two of the three pins between the USB and power jacks. Check that it is on the two pins closest to the USB port.
- Connect the Arduino board to your computer using the USB cable. The green power LED (labelled PWR) should glow.

4. Launch Arduino IDE:

- After your Arduino IDE software is downloaded, you need to unzip the folder. Inside the folder, you can find the application icon with an infinity label (application.exe). Double-click the icon to start the IDE.

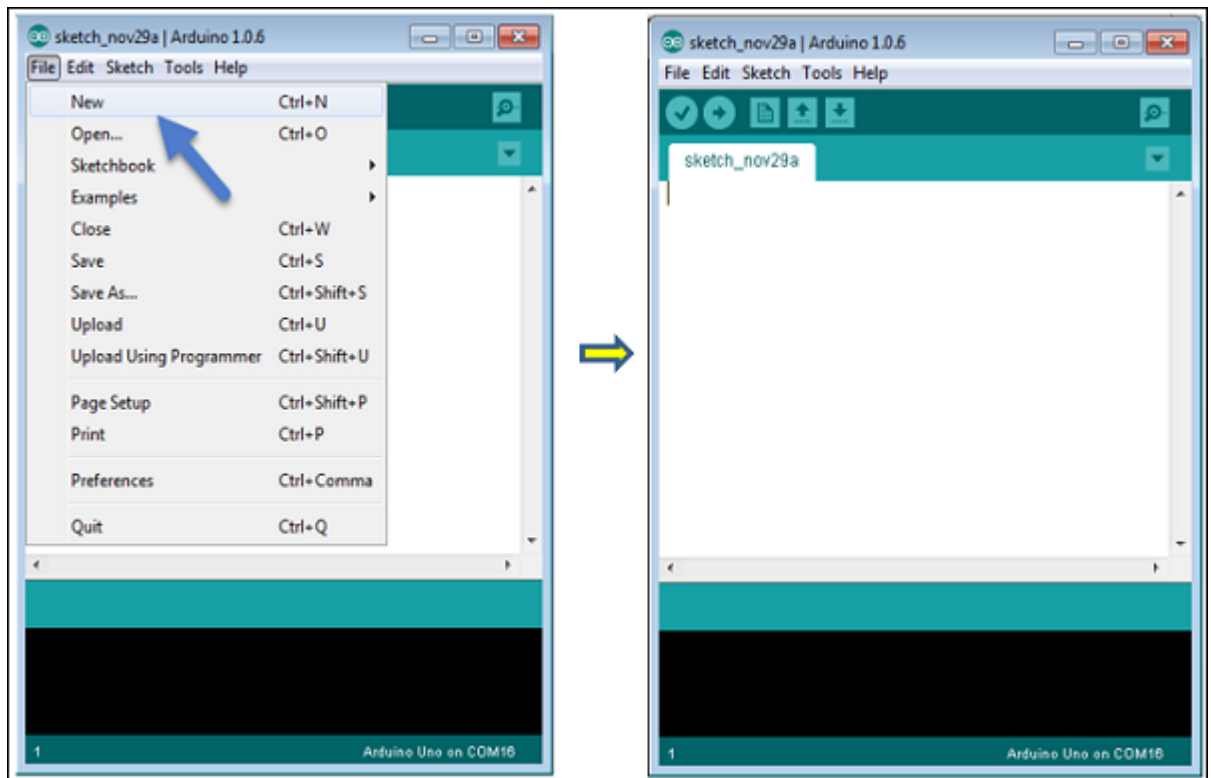
Internet of Things



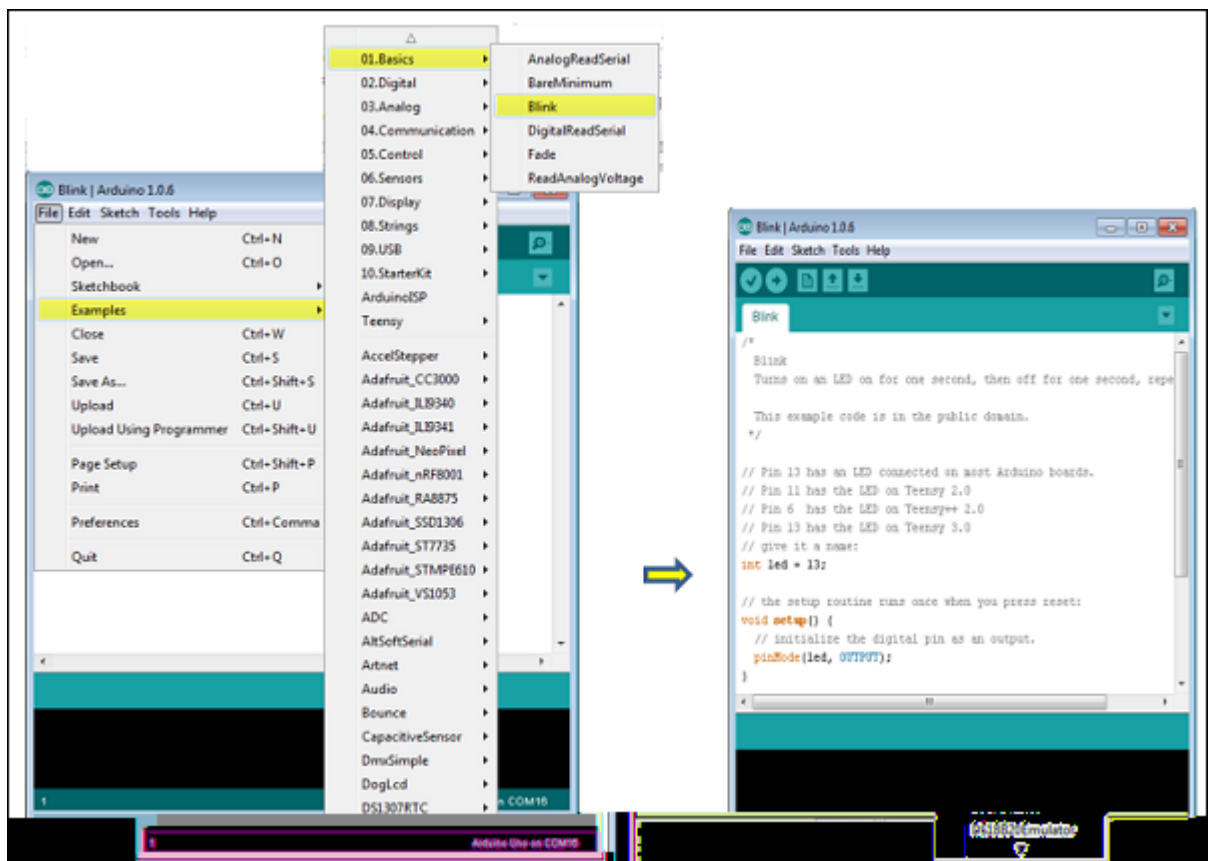
5. Open your first project:

- Once the software starts, you have two options –
 - Create a new project.
 - Open an existing project example.
- To create a new project, select File → **New**.

Internet of Things



- To open an existing project example, select File → Example → Basics → Blink.

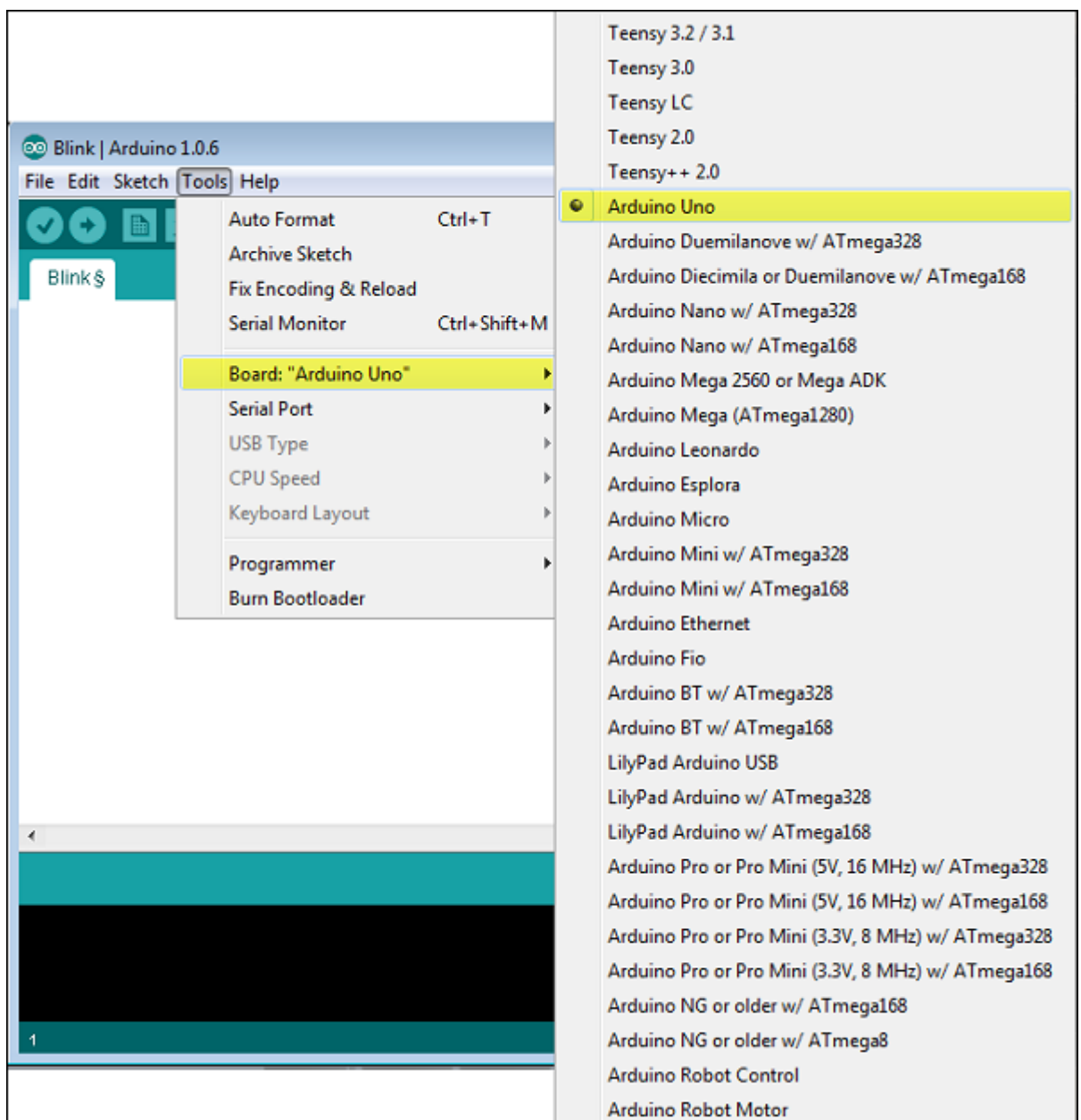


Internet of Things

- Here, we are selecting just one of the examples with the name **Blink**. It turns the LED on and off with some time delay. You can select any other example from the list.

6. Select your Arduino board:

- To avoid any error while uploading your program to the board, you must select the correct Arduino board name, which matches with the board connected to your computer.
- Go to Tools → Board and select your board.

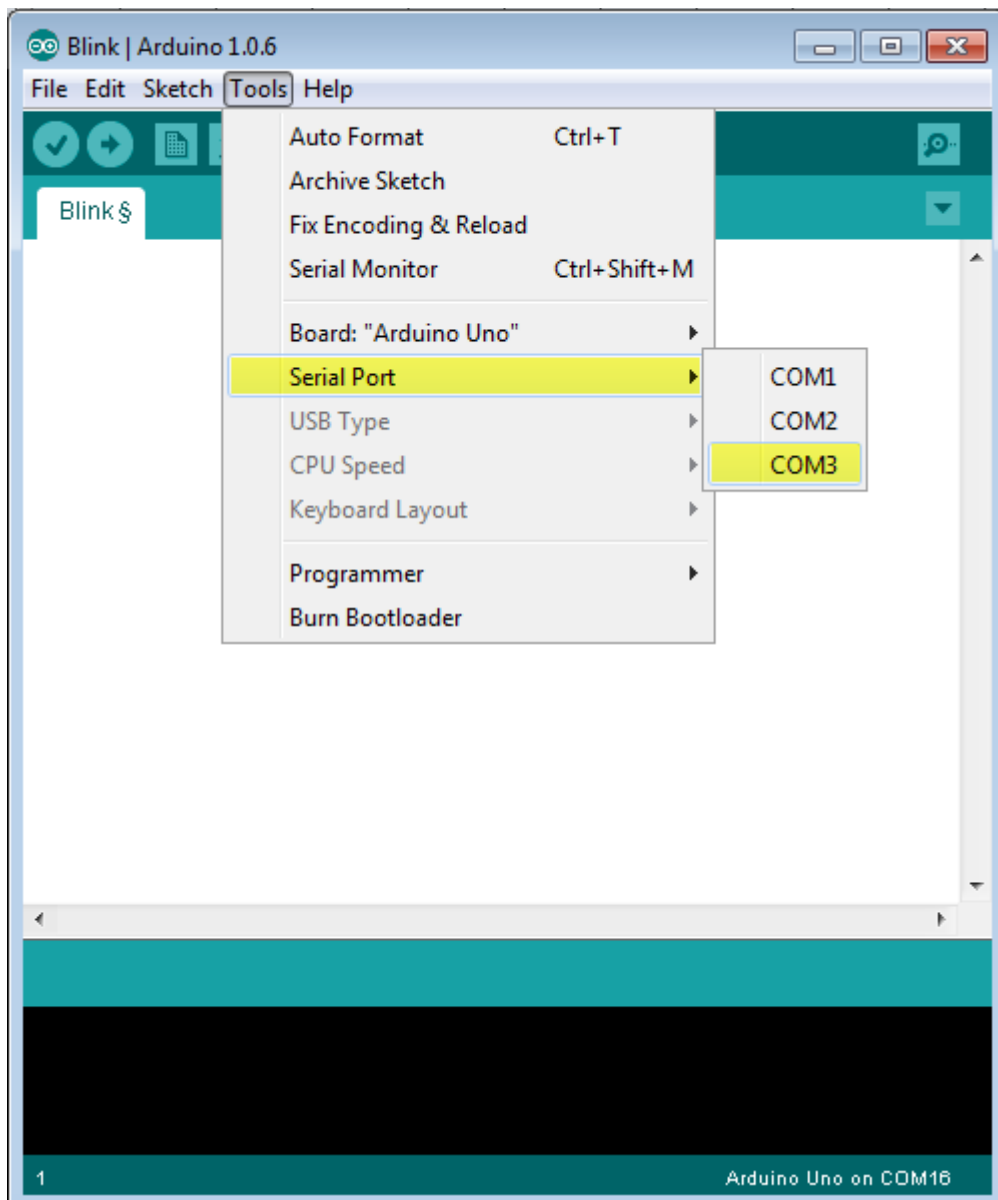


Internet of Things

- Here, we have selected Arduino Uno board according to our tutorial, but you must select the name matching the board that you are using.

7. Select your serial port:

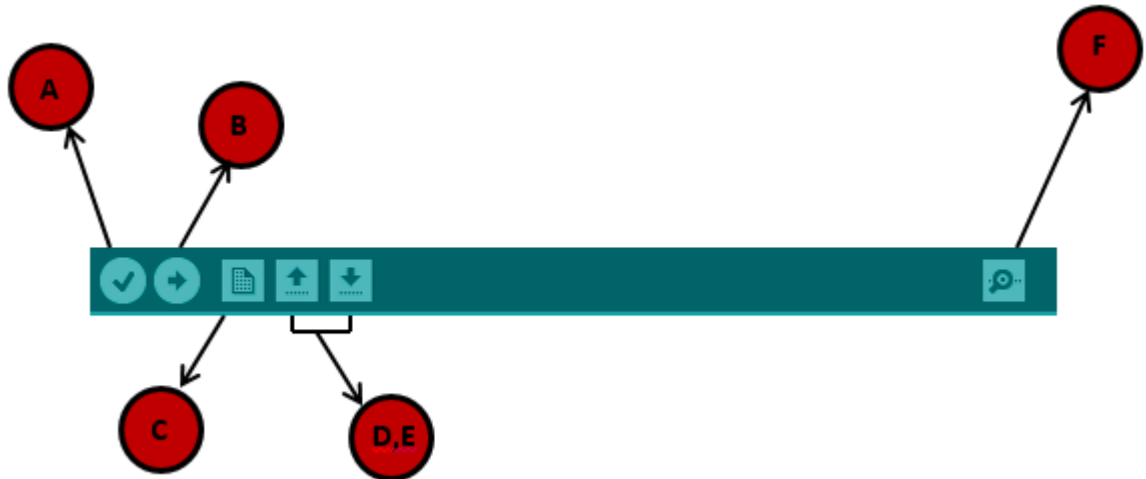
- Select the serial device of the Arduino board. Go to **Tools** → **Serial Port** menu. This is likely to be COM3 or higher (COM1 and COM2 are usually reserved for hardware serial ports). To find out, you can disconnect your Arduino board and re-open the menu, the entry that disappears should be of the Arduino board. Reconnect the board and select that serial port.



Internet of Things

8. Upload the program to your board:

- Before explaining how we can upload our program to the board, we must demonstrate the function of each symbol appearing in the Arduino IDE toolbar.

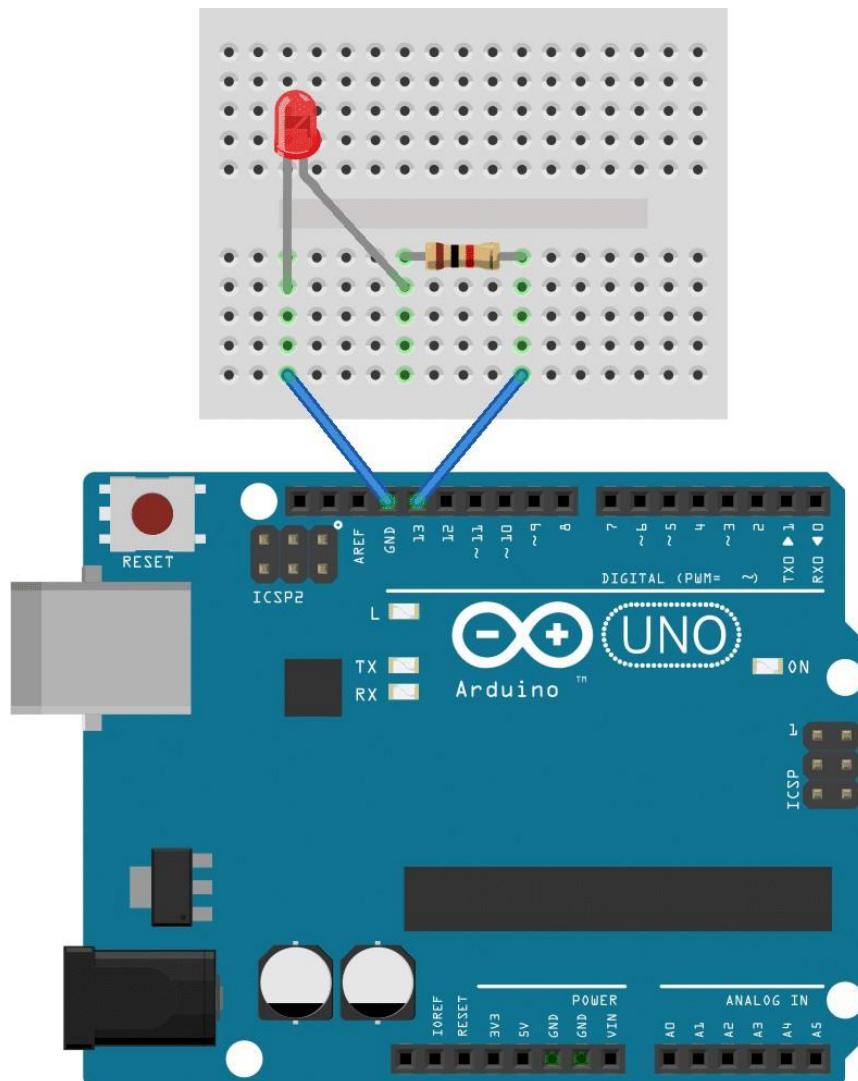


- A.** Used to check if there is any compilation error.
 - B.** Used to upload a program to the Arduino board.
 - C.** Shortcut used to create a new sketch.
 - D.** Used to directly open one of the example sketch.
 - E.** Used to save your sketch.
 - F.** Serial monitor used to receive serial data from the board and send the serial data to the board.
- Now, simply click the "Upload" button in the environment. Wait a few seconds; you will see the RX and TX LEDs on the board, flashing. If the upload is successful, the message "Done uploading" will appear in the status bar.

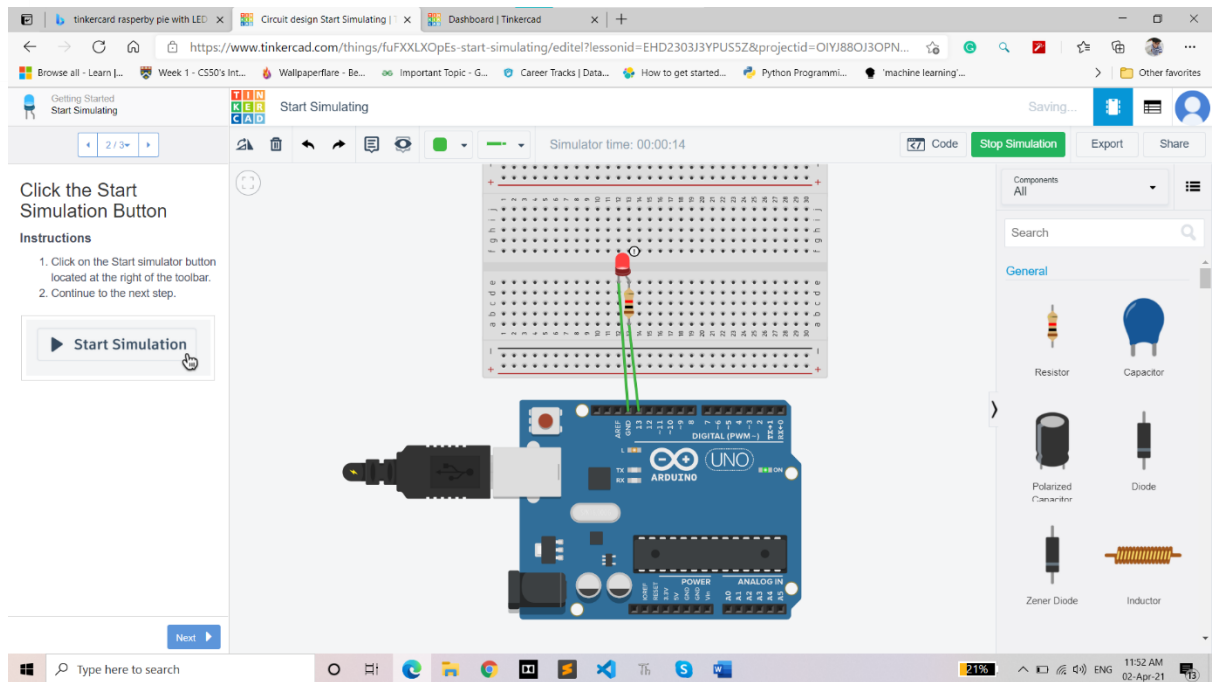
Internet of Things

Practical No 6: Controlling an LED using Arduino.

- Let's see how to turn on and off the Arduino's on-board LED. Then will see that how to turn on and off an LED connected to one of the Arduino's digital pins. Understand how to change the LEDs flashing rate, and how to change the pin that powers the LED. You should have the Arduino IDE software installed on your computer.



Internet of Things



➤ Controlling the Arduino's LED:

- To turn on an LED, the Arduino needs to send a HIGH signal to one of its pins. To turn off the LED, it needs to send a LOW signal to the pin. You can make the LED flash by changing the length of the HIGH and LOW states.
- The Arduino has an on-board surface mount LED that's hard wired to digital pin 13. It's the one with an "L" next to it:



- To get this LED flashing, upload the "Blink" program to your Arduino:

```
void setup() {
```

Internet of Things

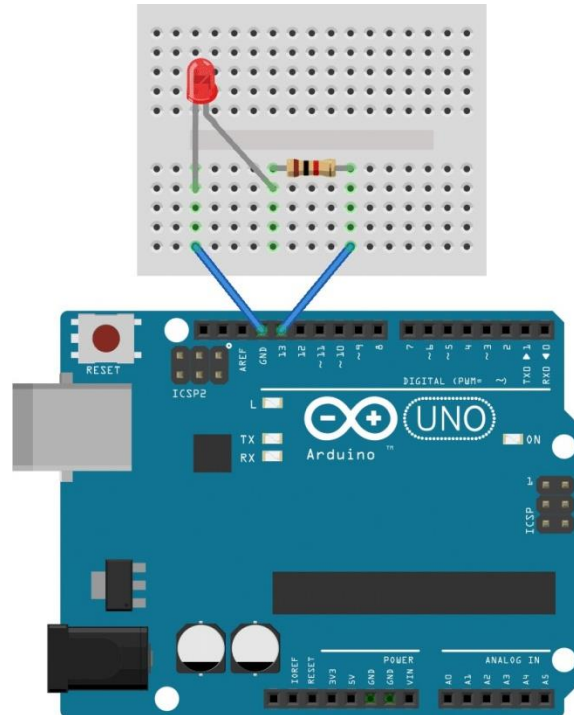
```
pinMode(13, OUTPUT);  
}  
void loop() {  
    digitalWrite(13, HIGH);  
    delay(1000);  
    digitalWrite(13, LOW);  
    delay(1000);  
}
```

- The LED should now be blinking on and off at a rate of 1000 milliseconds (1 second).
- The delay () function on line 6 tells the Arduino to hold the HIGH signal at pin 13 for 1000 ms. The delay () function on line 8 tells it to hold the LOW signal at pin 13 for 1000 ms. You can change the blinking speed by changing the number inside the parentheses of the delay () functions.

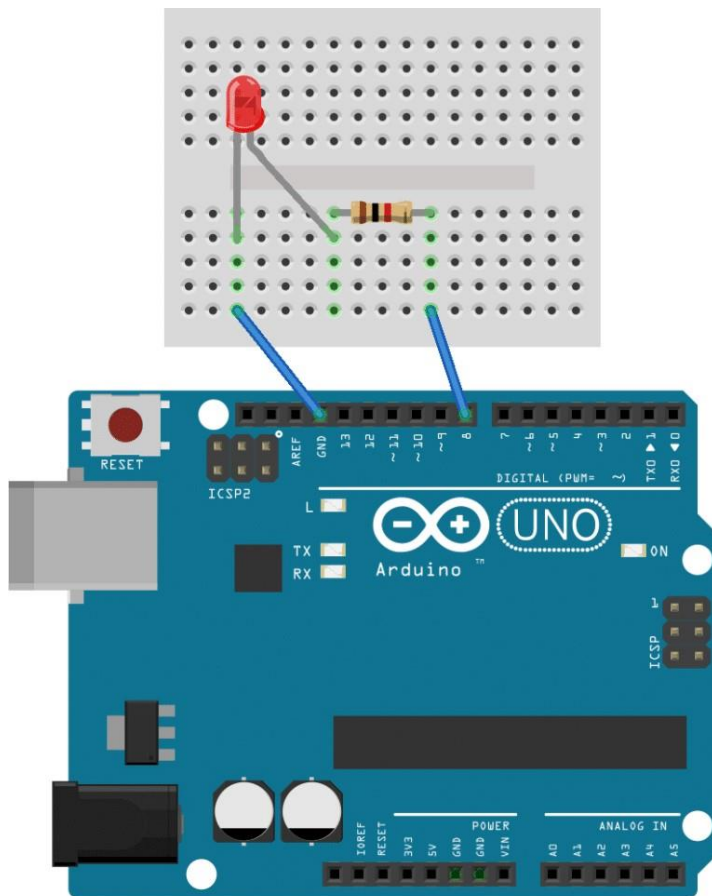
➤ Controlling an external LED:

- LEDs need to have a resistor placed *in series* (in-line) with it. Otherwise, the unrestricted current will quickly burn out the LED. The resistor can be any value between 100 Ohms and about 10K Ohms. Lower value resistors will allow more current to flow, which makes the LED brighter. Higher value resistors will restrict the current flow, which makes the LED dimmer.
- Also, most LED's have polarity, which means that they need to be connected the right way around. Usually, the LED's shortest lead connects to the ground side.
- If you connect the LED to pin 13 as shown in the image below, you can use the same code we used above to make the LED flash on and off.

Internet of Things



- If you want to use a different pin to power the LED, it's easy to change it. For example, say you want to use pin 8 instead of pin 13. First move the signal wire from pin 13 over to pin 8:



Internet of Things

- Now you'll need to edit a line of code in the program so the Arduino knows which pins to use as output pins. That's done on line 2 of the code above, where it says:

```
pinMode (13, OUTPUT);
```

- To use pin 8, you just have to change the 13 to an 8:

```
pinMode (8, OUTPUT);
```

- Next, change the code that tells the Arduino which pins will get the HIGH and LOW output signals. That's done everywhere there's a digitalWrite() function. In the program above, there's one on line 5 and one on line 7:

```
digitalWrite(13, HIGH);  
digitalWrite(13, LOW);
```

- To specify that pin 8 should get the HIGH and LOW signals, you just need to change the 13's to 8's:

```
digitalWrite(8, HIGH);  
digitalWrite(8, LOW);
```

- The finished program should look like this:

```
void setup() {  
  pinMode(8, OUTPUT);  
}  
void loop() {  
  digitalWrite(8, HIGH);  
  delay(1000);  
  digitalWrite(8, LOW);  
  delay(1000);  
}
```

- After uploading, the LED should flash just like it did when it was connected to pin 13.

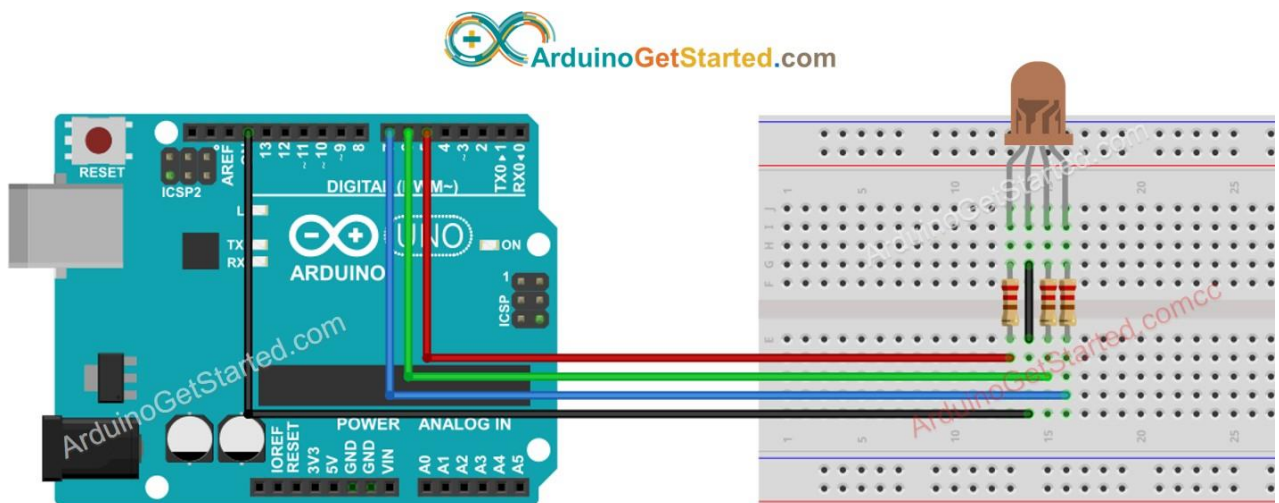
➤ RGB LED connections

Mainly three colours with RGB Pin

Internet of Things



- In the nature of physics, a colour is composed of three colour values: Red (R), Green (G) and Blue (B). Each colour value ranges from 0 to 255.
- The mix of three values creates $256 \times 256 \times 256$ colours in total.
- If we provide PWM signals (with duty cycle from 0 to 255) to R, G, B pins, we can make RGB LED displays any colour we want.
- The duty cycle of PWM signals to R, G and B pins correspond to colour values of Red (R), Green (G) and Blue (B)



CODE:

```
const int PIN_RED = 5;
const int PIN_GREEN = 6;
const int PIN_BLUE = 7;

void setup() {
  pinMode(PIN_RED, OUTPUT);
```

```
pinMode(PIN_GREEN, OUTPUT);
pinMode(PIN_BLUE, OUTPUT);
}

void loop() {
  // color code #00C9CC (R = 0, G = 201, B = 204)
  analogWrite(PIN_RED, 0);
  analogWrite(PIN_GREEN, 201);
  analogWrite(PIN_BLUE, 204);

  delay(1000); // keep the color 1 second

  // color code #F7788A (R = 247, G = 120, B = 138)
  analogWrite(PIN_RED, 247);
  analogWrite(PIN_GREEN, 120);
  analogWrite(PIN_BLUE, 138);

  delay(1000); // keep the color 1 second

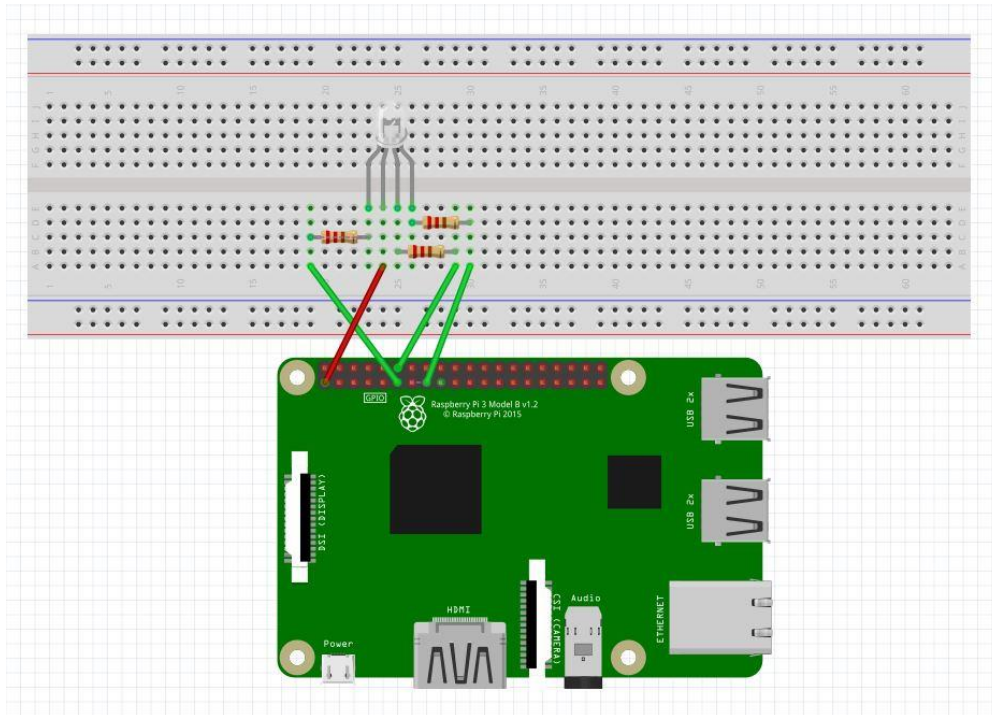
  // color code #34A853 (R = 52, G = 168, B = 83)
  analogWrite(PIN_RED, 52);
  analogWrite(PIN_GREEN, 168);
  analogWrite(PIN_BLUE, 83);

  delay(1000); // keep the color 1 second
}
```

Internet of Things

✚ Practical No 7: Write Program for RGB LED using Raspberry Pi.

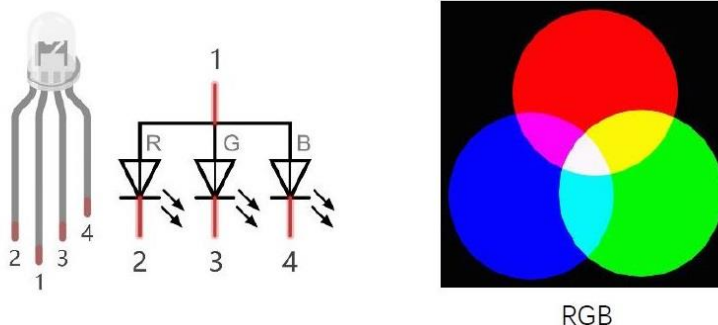
- We have to connect the Light Emitted Diode (LED) with bread-board and hard wired with Programmable Raspberry circuit. It will perform our code and blinks at interval like 5 seconds.



Main architecture

- As figure show, we connect register and LED light and Raspberry input port and Ground and Circuit switches.

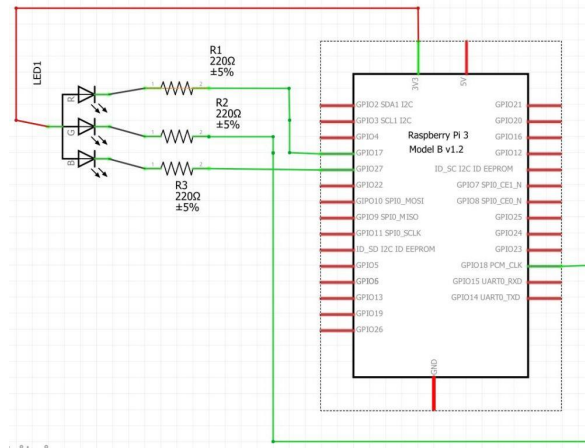
○ I want to see RGB Colour as (RED GREEN BLUE)



■ RGB LED

Internet of Things

- After this we Show pins of Raspberry GPIO17 and GPIO27 to switch with Register



Circuit Diagram

Python Script to Control RGB LED with Raspberry Pi

- Since each of the LED in an RGB LED is controlled via PWM, we will control 3 GPIO pins to generate PWM Signal.
- First, initialize all of the utilized GPIO resources as a PWM channel. We **defined a tuple named pins** to abstract the GPIO pins that correspond to the RGB pins of the LED. Also, since this is a Common Anode RGB LED, **make sure you set the initial signal too HIGH to turn off the LED**.
- We will be displaying different colours for each second in this script. Its also a great practice to keep your code simple by defining functions such as **setColor()**. This function sets the duty cycle for each of the PWM channels.

```

From gpiozero import LED # It is for just one colour LED
Import time
K = input ("Sleep Time:")
Led = LED (18)
While True:
    led.toggle()
    time.sllep(k)
    led.toggle()
    time.sleep(k)

```

CODE

```
import RPi.GPIO as GPIO
```

```
import time
import random

pins = (11,12,13) # R = 11, G = 12, B = 13

def setup():
    global pwmR, pwmG, pwmB
    GPIO.setmode(GPIO.BOARD)
    for i in pins: # iterate on the RGB pins, initialize each and set to HIGH to turn it off (COMMON ANODE)
        GPIO.setup(i, GPIO.OUT)
        GPIO.setup(i, GPIO.HIGH)
    pwmR = GPIO.PWM(pins[0], 2000) # set each PWM pin to 2 KHz
    pwmG = GPIO.PWM(pins[1], 2000)
    pwmB = GPIO.PWM(pins[2], 2000)
    pwmR.start(0) # initially set to 0 duty cycle
    pwmG.start(0)
    pwmB.start(0)

def setColor(r, g, b): # 0 ~ 100 values since 0 ~ 100 only for duty cycle
    pwmR.ChangeDutyCycle(r)
    pwmG.ChangeDutyCycle(g)
    pwmB.ChangeDutyCycle(b)

def displayColors():
    setColor(100, 0, 0) # red color
    time.sleep(1) # 1s
    setColor(0, 100, 0) # green
    time.sleep(1) # 1s
    setColor(0, 0, 100) # blue
    time.sleep(1) # 1s
    setColor(100, 100, 0) # yellow
    time.sleep(1) # 1s
    setColor(0, 100, 100) # cyan
    time.sleep(1) # 1s
    setColor(100, 0, 100) # magenta
    time.sleep(1) # 1s
    setColor(50, 0, 0) # maroon
    time.sleep(1) # 1s
    setColor(50, 0, 50) # purple
    time.sleep(1) # 1s
    setColor(0, 0, 50) # navy
    time.sleep(1) # 1s

def destroy():
    pwmR.stop()
    pwmG.stop()
    pwmB.stop()
    GPIO.cleanup()

if __name__ == '__main__':
    setup()
    displayColors()
    destroy()
```

Internet of Things

✚ Practical No 8: Study the Temperature sensor and Write Program for monitor temperature using Arduino

➤ We will take Humidity from DHT11 & DHT22 Sensor with LCD display

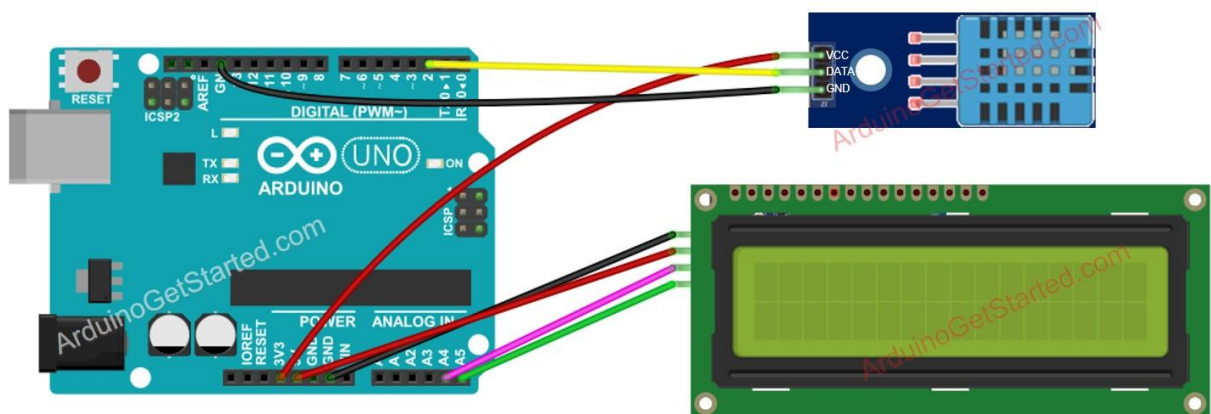
Hardware Required

- 1) Arduino Uno or Genuino Uno
- 2) USB 2.0 cable type A/B
- 3) LCS I2C
- 4) Temperature and Humidity Sensor DHT11 & DHT22
- 5) Jumper Wires

The differences between DHT11 and DHT22

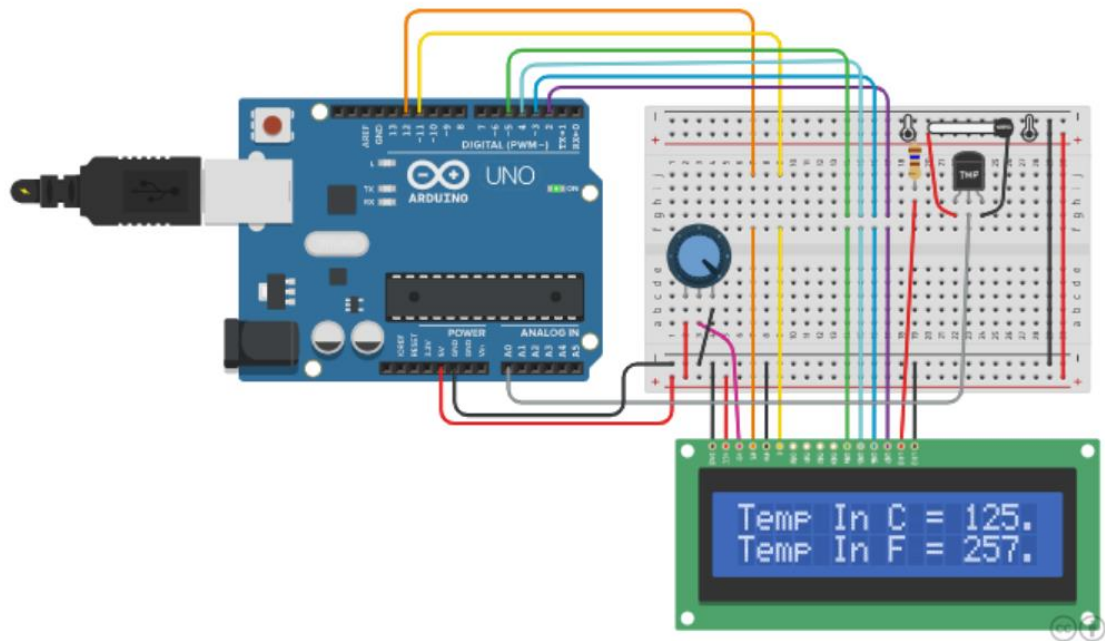
	DHT11	DHT22
Price	ultra low cost	low cost
Temperature Range	0°C to 50°C	-40°C to 80°C
Temperature Accuracy	± 2°C	± 0.5°C
Humidity Range	20% to 80%	0% to 100%
Humidity Accuracy	5%	± 2 to 5%
Reading Rate	1Hz (once every second)	0.5Hz (once every 2 seconds)
Body size	15.5mm x 12mm x 5.5mm	15.1mm x 25mm x 7.7mm

Tinker card Implement

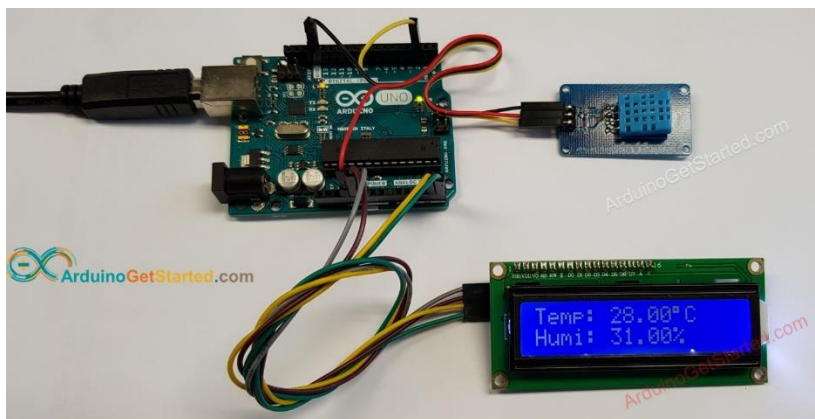


Internet of Things

Displaying Temperature Sensor Values on LCD Display Tinker cad



Real-time Application



CODE

```
#include <LiquidCrystal_I2C.h>
#include "DHT.h"
#define DHTPIN 2
#define DHTTYPE DHT11
```

Internet of Things

```
LiquidCrystal_I2C lcd(0x3F, 16, 2); // I2C address 0x3F, 16 column and 2 rows
DHT dht(DHTPIN, DHTTYPE);

void setup()
{
    dht.begin(); // initialize the sensor
    lcd.init(); // initialize the lcd
    lcd.backlight(); // open the backlight
}

void loop()
{
    delay(2000); // wait a few seconds between measurements

    float humi = dht.readHumidity(); // read humidity
    float tempC = dht.readTemperature(); // read temperature

    lcd.clear();
    // check if any reads failed
    if (isnan(humi) || isnan(tempC)) {
        lcd.setCursor(0, 0);
        lcd.print("Failed");
    } else {
        lcd.setCursor(0, 0); // start to print at the first row
        lcd.print("Temp: ");
        lcd.print(tempC); // print the temperature
        lcd.print((char)223); // print ° character
        lcd.print("C");

        lcd.setCursor(0, 1); // start to print at the second row
```

Internet of Things

```
lcd.print("Humi: ");  
lcd.print(humi);    // print the humidity  
lcd.print("%");  
}  
}
```

Output:



Internet of Things

Practical No 9: Study and Implement RFID, NFC using Arduino.

- In this tutorial, we are going to learn how to use RFID/NFC with Arduino. The RFID/NFC system includes two components: reader and tag. There are two popular RFID/NFC readers: RC522 and PN532 RFID/NFC reader.
- This tutorial focuses on RC522 RFID/NFC reader. PN532 RFID/NFC reader will be presented in an upcoming tutorial.
- RC522 RFID/NFC reader (also called RFID-RC522 Module) can:
 - Read the UID of RFID/NFC tag
 - Change the UID of RFID/NFC tag (only if the tag is UID-writable)
 - Read data from RFID/NFC tag
 In above capabilities, for Arduino, reading the UID is the most widely-used. This tutorial focuses on reading the UID of RFID/NFC tag. The other will be present in next tutorials

Hardware Required:

- 1) Arduino Uno or Genuino Uno
- 2) USB 2.0 cable type A/C
- 3) RFID/NFC RC522 kit (Reader/Tags)
- 4) Jumper Wires

About RFID-RC522 Module:

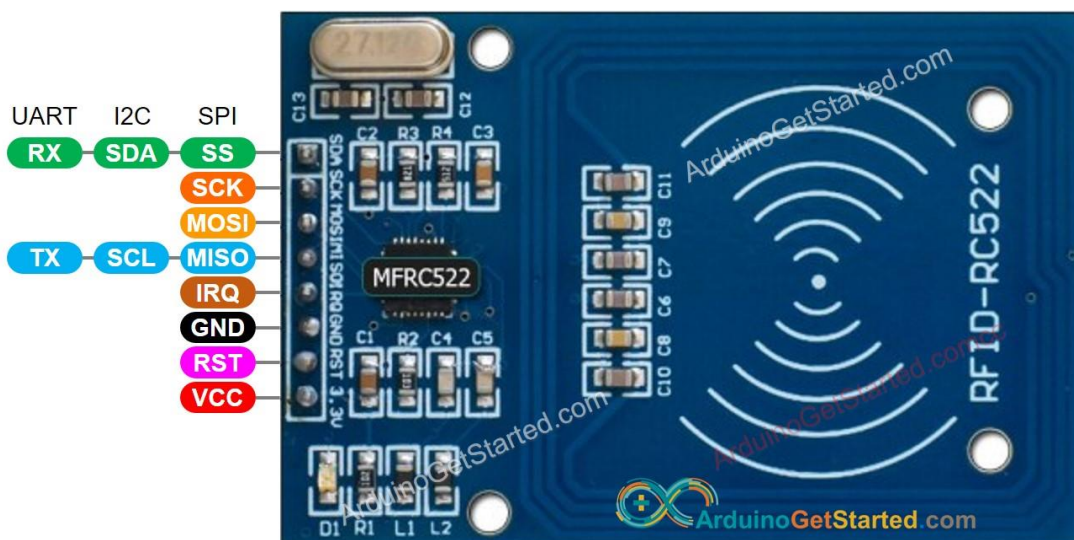
- PinoutRFID-RC522 has 8 pins, some of them are common pins, the others are shared among three communication modes: SPI, I2C, UART. At a time, only one communication mode can be used.

The pin is:

- **GND pin:** needs to connected to Gnd(0v)
- **VCC pin:** needs to connected to vcc(3.3)
- **RST pin:** is a pin for reset and power-down,when this pin goes low,hard power doen is enable,the module is reset.

Internet of Things

- **IRQ pin:** is an interrupt pin that can alert the microcontroller when RFID tag comes into its vicinity.
- **MISO/SCL/TX PIN:** acts as MISO when spi interface is enabled, act as SCL when i2c interface is enabled and acts as TX when UART interface is enabled.
- **MOSI pin :** acts as mosi when spi interface is enabled.
- **SCK pin:** acts as Sck when spi interface is enabled.
- **SS/SDA/RX pin:** acts as SS when spi interface is enabled, acts as SDA when I2C interface is enabled, acts as RX when UART interface is enabled.



The reader consists of a radio frequency module and an antenna which generates high frequency electromagnetic field

The tag is usually a passive device, which doesn't need to have power source. The tag contains a microchip that stores and processes information, and an antenna to receive and transmit a signal. The tag is used to store the information: UID (Unique ID) and data.

Internet of Things



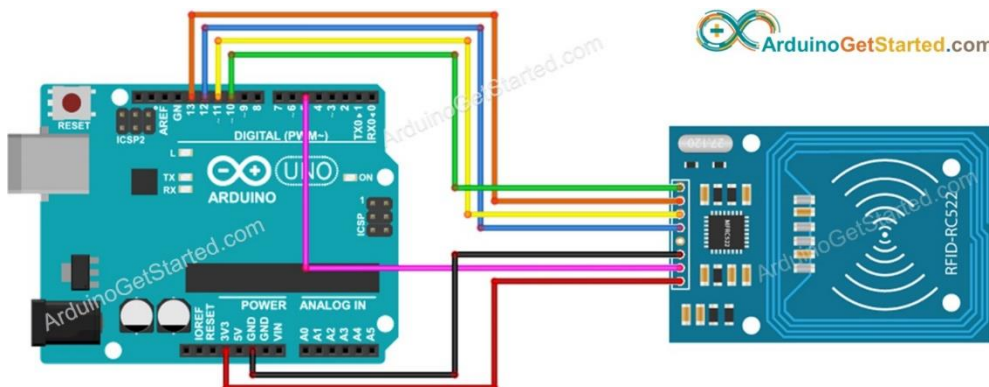
RFID/NFC Reader



RFID/NFC Tag

- To read the information on a tag, the tag must be in close proximity to the reader (does not require the direct line-of-sight). The reading processes:
- The reader generates an electromagnetic field which causes electrons to move through the tag's antenna and subsequently power the chip.
- The chip inside the tag then responds by sending the requested information back to the reader in the form of another radio signal.
- The reader detects the signal and transform the signal into data

Arduino reads the data from reader.



- If you use the male-to-female jumper wires, you can connect Arduino UNO directly to RFID-RC522 module. If you use the male-to-male jumper wires, you need to connect Arduino UNO to RFID-RC522 module via a breadboard.

CODE:

```
#include <SPI.h>
#include <MFRC522.h>

#define SS_PIN 10
#define RST_PIN 5
```

```

MFRC522 rfid(SS_PIN, RST_PIN);

void setup() {
  Serial.begin(9600);
  SPI.begin(); // init SPI bus
  rfid.PCD_Init(); // init MFRC522

  Serial.println("Tap RFID/NFC Tag on reader");
}

void loop() {
  if (rfid.PICC_IsNewCardPresent()) { // new tag is available
    if (rfid.PICC_ReadCardSerial()) { // NUID has been readed
      MFRC522::PICC_Type piccType = rfid.PICC_GetType(rfid.uid.sak);
      //Serial.print("RFID/NFC Tag Type: ");
      //Serial.println(rfid.PICC_GetTypeName(piccType));

      // print NUID in Serial Monitor in the hex format
      Serial.print("UID:");
      for (int i = 0; i < rfid.uid.size; i++) {
        Serial.print(rfid.uid.uidByte[i] < 0x10 ? " 0" : " ");
        Serial.print(rfid.uid.uidByte[i], HEX);
      }
      Serial.println();

      rfid.PICC_HaltA(); // halt PICC
      rfid.PCD_StopCrypto1(); // stop encryption on PCD
    }
  }
}

```

OUTPUT:

See UID on Serial Monitor.



Internet of Things

Practical-10: Study and Implement Zigbee Protocol using Raspberry Pi

➤ How to Interface ZigBee Module with Raspberry Pi

Interface XBee module with Raspberry Pi which will act as a receiver and make it communicate wirelessly with another XBee module (XBee explorer board) which is serially connected with the laptop.

➤ Hardware Requirements

1. 1 x Raspberry Pi with Raspbian Installed in it
2. 2 x XBee Pro S2C modules (any other model can be used)
3. 1 x XBee explorer board (optional)
4. 1 x Xbee Breakout board (optional)
5. USB cables
6. LEDs

➤ Configuring XBee Modules using XCTU

the XBee module can act as a Coordinator, Router or an End device but it needs to be configured to work in desired mode. So, before using the **XBee modules with Raspberry Pi**, we have to configure these modules using XCTU software.

To connect XBee module with the laptop, a USB to serial converter or specifically designed explorer board is used. Just hook up the XBee module to the Explorer board and plug it with the laptop using USB cable.

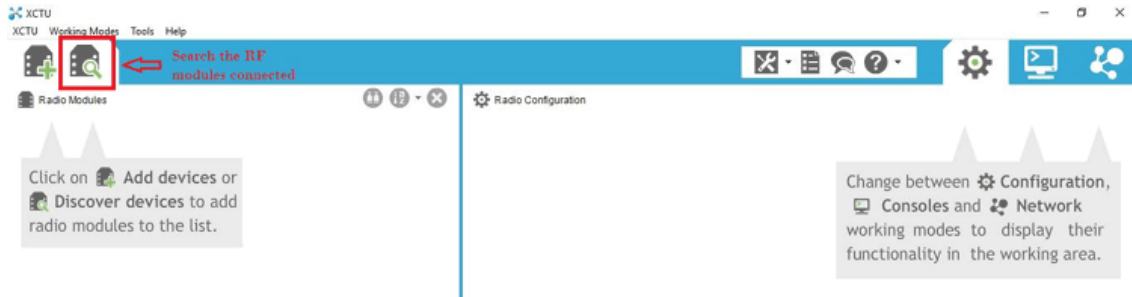
If you don't have any converter or explorer board, then an **Arduino board can be used as a USB to serial device** which can easily communicate with the XBee and laptop. Just upload blank sketch in Arduino board and now it can behave like a USB to Serial converter.

➤ Configuring XBee Modules:

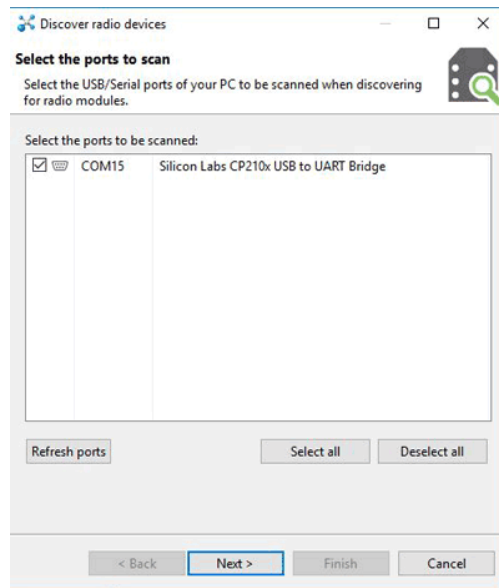
Download the [XCTU software from this link](#) and install it. After downloading and installing the XCTU software, open it and make sure your XBee module is properly connected. Check the COM port of the Arduino board in device manager.

1. Now, click on the search button. This will show you all the RF devices connected with your laptop. In our case, it will show only one XBee module.

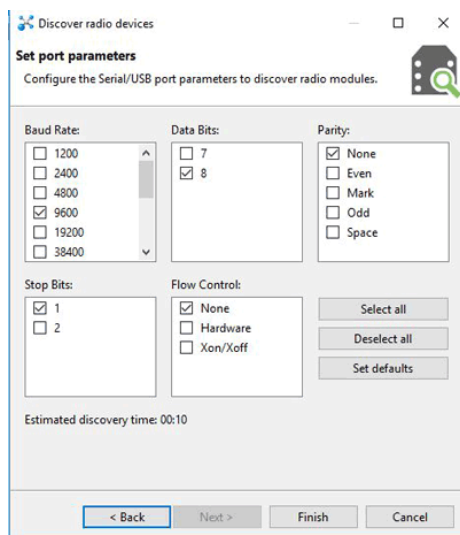
Internet of Things



2. Select the Serial port of the Explorer board/Arduino board and click on Next.

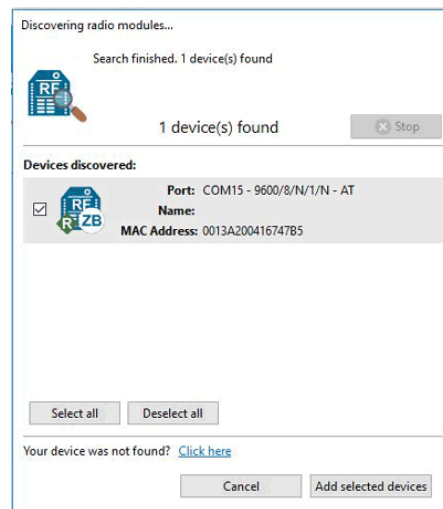


3. In the next window, set the USB port parameters as shown below and click on Finish.



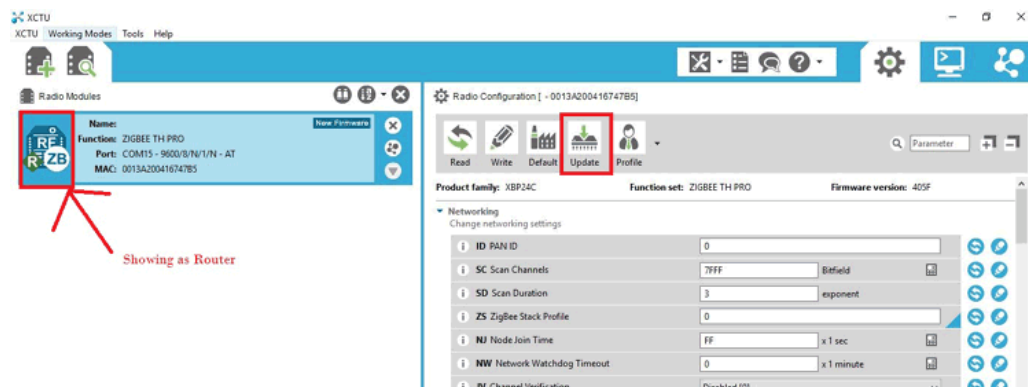
4. Select the Discovered device and click on **Add selected device**. This process will add your XBee module to XCTU dashboard.

Internet of Things

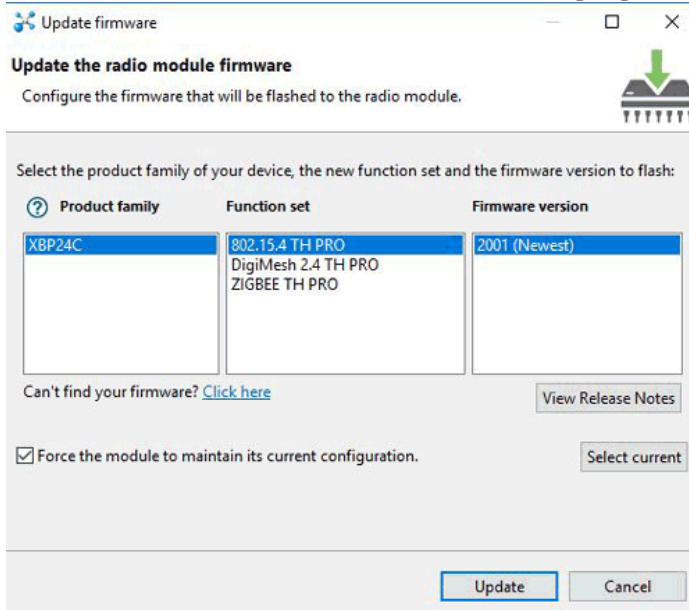


5. Now, you can configure your XBee module in this window. You can use either AT commands or put the data manually. As you can see, there is **R** showing on the left panel which means XBee is in router mode. We have to make it Coordinator for the transmitter part.

First, update the Firmware by clicking on the Update firmware.

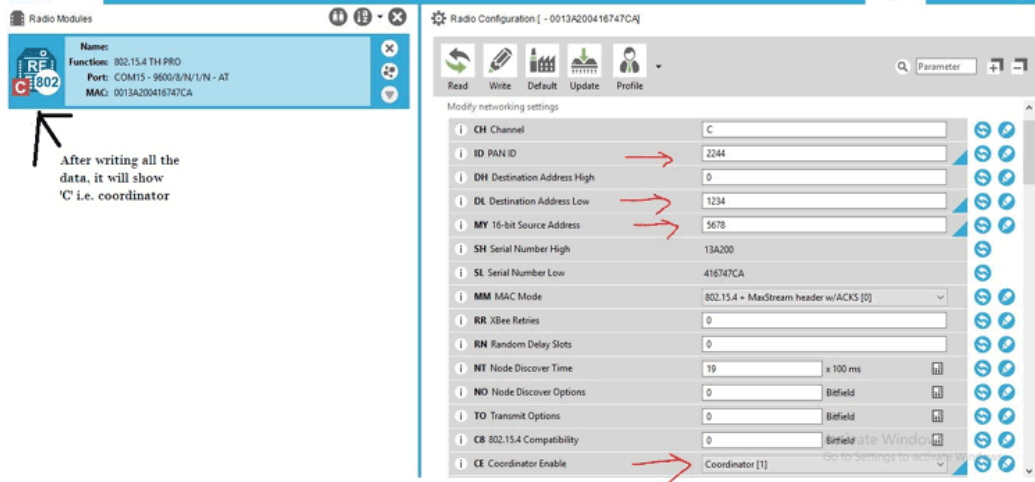


6. Choose the Product family of your device which is available on back of your XBee module. Select function set and firmware version as highlighted below and **click on Update**.



Internet of Things

7. Now, you have to give ID, MY and DL data to make connection with other XBee. ID remains same for both the modules. Only MY and DL data interchange i.e. **MY for the receiver XBee becomes DL of the transmitter XBee (coordinator)** and **DL for the receiver XBee becomes MY of the transmitter XBee**. Make CE as **Coordinator** and then hit the **Write** button. As shown below.



	<i>ATDL</i>	<i>ATMY</i>	<i>ATID</i>
<i>XBee 1 coordinator</i>	1234	5678	2244
<i>XBee 2 end device</i>	5678	1234	2244

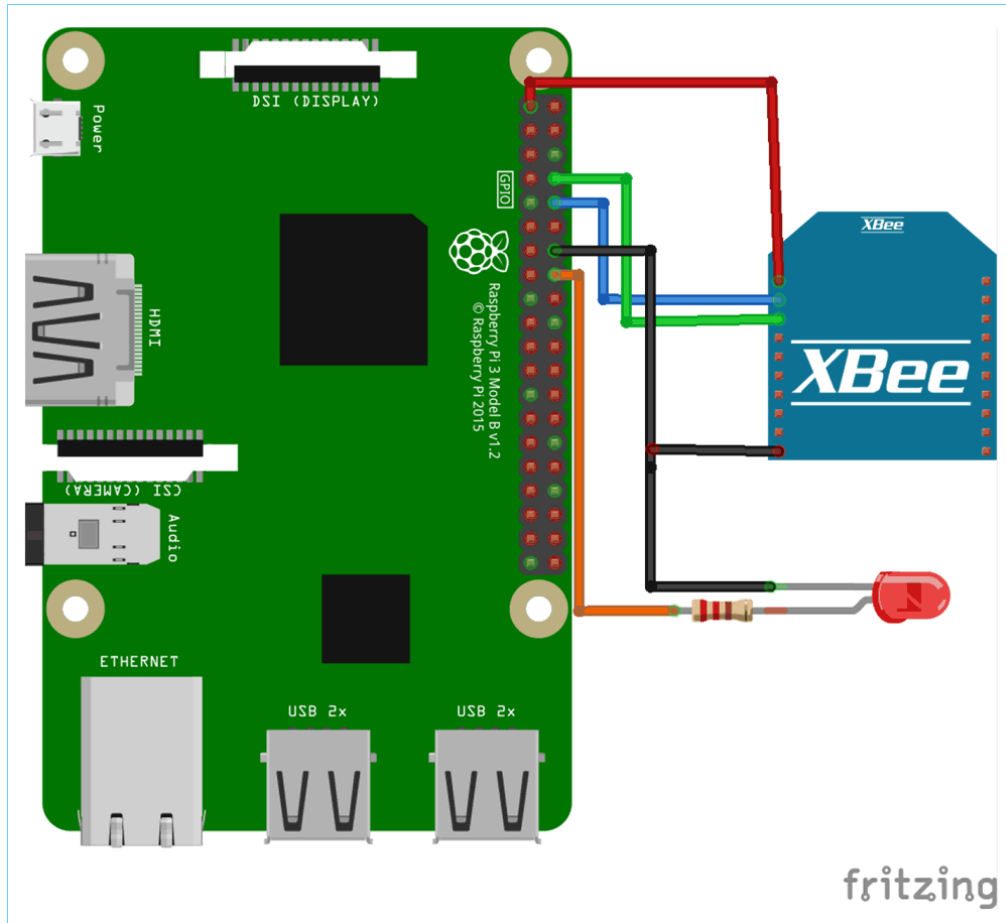
8. After writing the above data to the transmitter part, plug out it from the explorer board and plug in the second XBee module in it. Repeat the same process as above only changes are the DL, MY, and CE. **As we will make the second XBee as End device so in CE drop down menu, select the End device and hit the Write button.**

9. Now, our XBee modules are ready to interface with the Raspberry Pi. We will **connect the transmitter XBee to the laptop and receiver XBee with the Raspberry Pi**. Then give commands to the receiver part using laptop. laptop.

➤ Circuit Diagram for Receiver Part

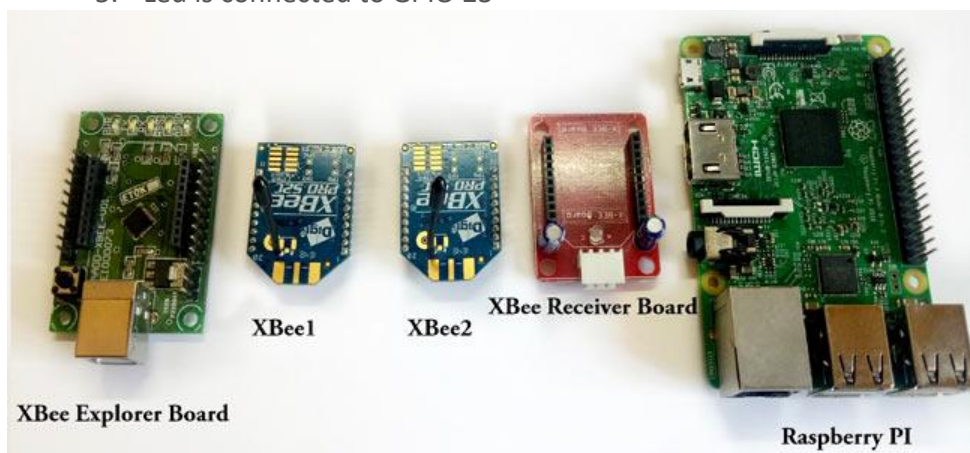
Connections for **interfacing ZigBee module with Raspberry PI** are shown in the circuit diagram.

Internet of Things



➤ Connections:

1. Tx (pin2) of XBee -> Tx of pin Raspberry Pi
2. Rx (pin3) of XBee -> Rx of pin Raspberry Pi
3. Gnd (pin10) of XBee -> GND of pin Raspberry Pi
4. Vcc (Pin1) of XBee -> 3.3v of pin Raspberry Pi
5. Led is connected to GPIO 23



Internet of Things

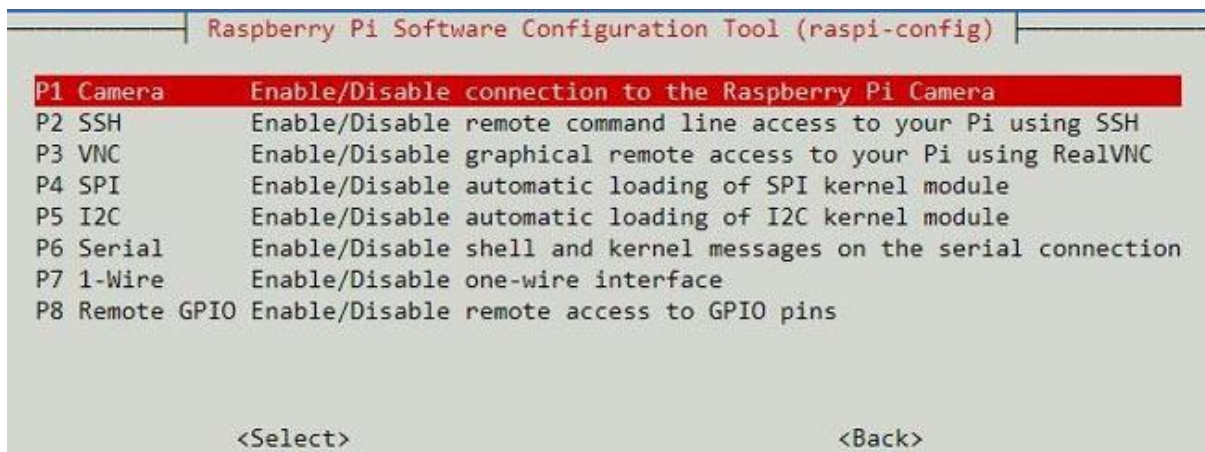
➤ Setup Raspberry Pi for Serial communication

Now, we will setup the Raspberry Pi for the Serial communication. By default, the hardware serial port of Pi is disabled. So, we have to enable it before starting the connection.

1. In the terminal, run the command *raspi-config*.



2. Go to option 5 *Interfacing options* and hit the enter. Now, select the *P6 Serial* option and Enable it and then save.



Exit the terminal and you are all set to make the connection between Raspberry Pi and XBee. GPIO14 and 15 will act as Tx and Rx respectively and these are available at /dev/ttySO port of raspberry pi.

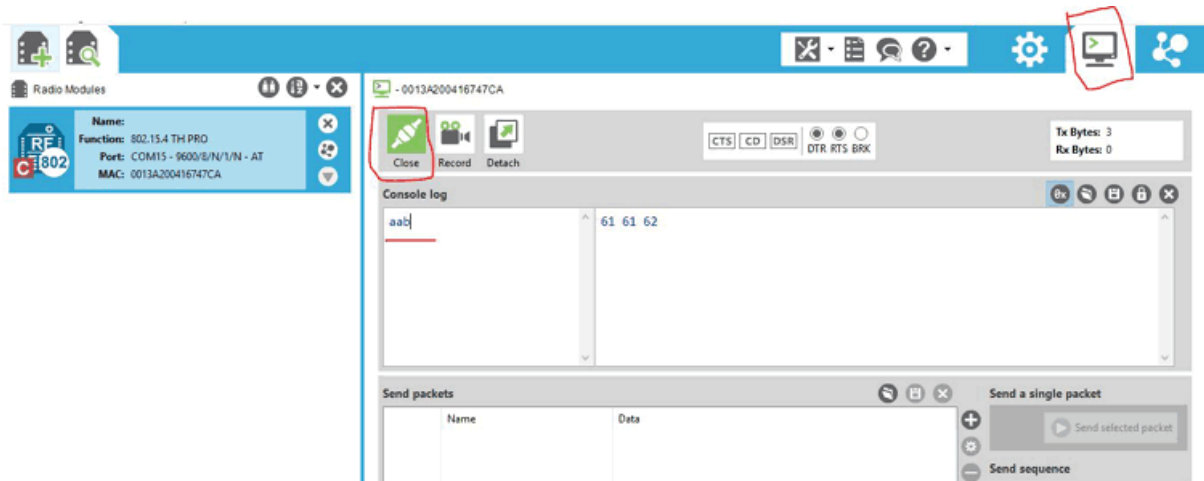
Now, we will write a python script to ON the LED whenever we receive 'a' from the transmitter side XBee.

➤ Testing the wireless XBee communication using Raspberry Pi

Now, we all set to **test our XBee transmitter and receiver**. To give command to the transmitter part, we will use XCTU's console terminal. Click on the Console icon near the settings option. Then, click on Open button to connect the XBee to the laptop.

Internet of Things

Enter 'a' in Console log. You will see that LED will turn ON for 3 seconds and then it turn OFF.



➤ Code:

```
#!/usr/bin/env python
import time
import serial
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(23,GPIO.OUT)
ser = serial.Serial(
    port='/dev/ttyS0',
    baudrate = 9600,
    parity=serial.PARITY_NONE,
    stopbits=serial.STOPBITS_ONE,
    bytesize=serial.EIGHTBITS,
    timeout=1
)
counter=0

while 1:
    #ser.write(str.encode('Write counter: %d \n'%(counter)))
    #time.sleep(1)
    #counter += 1
    x=ser.readline().strip()
    print(x)
    if x == 'a':
        GPIO.output(23,GPIO.HIGH)
        time.sleep(3)
    else:
        GPIO.output(23,GPIO.LOW)
```