Dr. Vishwanath Karad
MIT WORLD PEACE UNIVERSITY | PUNE
TECHNOLOGY, RESEARCH, SOCIAL INNOVATION & PARTNERSHIPS

School of Electronics and Communication Engineering

Third Year B. Tech. (ECE)

IoT Architectures and Protocols

Course Code: ECE3007B

Laboratory Manual

2022-23

Dr. Vishwanath Karad
MIT WORLD PEACE UNIVERSITY | PUNE
TECHNOLOGY, RESEARCH, SOCIAL INNOVATION & PARTNERSHIPS

Dr. Vishwanath Karad
MIT WORLD PEACE UNIVERSITY | PUNE
TECHNOLOGY, RESEARCH, SOCIAL INNOVATION & PARTNERSHIPS

Internet of Things

I N D E X

Sr. No.	Name of the Experiment	Page	Date of Checking	Signature of Batch I/C
1	To introduce various hardware platforms with respective GPIO pins/IDE/OS installation for IoT based design.			
2	To interface Sensors and actuators with hardware platform.			
3	To Study and implement RFID/NFC based application			
4	To Study and implement ZigBee/BLE protocol using hardware platform.			
5	To Study and implement MQTT/AMQP/HTTP protocol using hardware platform			
6	To design a simple IoT based system using cloud infrastructure for connecting IoT devices			
7	Data visualization and Analysis for IoT application.			
8	PBL- Title Finalization of Project based on real life IoT applications using cloud platform,			
9	PBL- Project Implementation			
10	PBL- Final Project Demonstration, Presentation and Report submission			

CERTIFICATE

Certified that Mr./Ms.____ of Class **Third Year B. Tech. (ECE)** Division_____ Roll No._____________ has completed the laboratory work in the subject **IoT Architectures and Protocols** during the **Semester VI** in the School of Electronics and Communication Engineering during the Academic Year 2022-2023.

Signature of the Faculty **Head of the Department**

Third Year B. Tech (ECE)

Semester: VI **Subject:** IoT Architectures and Protocols

Name:	**Class:**
Roll No:	**Batch:**

Experiment No: 01

Name of the Experiment: To introduce various hardware platforms with respective GPIO pins/IDE/OS installation for IoT based design.

Performed on:

Submitted on:

Marks	Teacher's Signature with date

Aim: To introduce various hardware platforms with respective GPIO pins/IDE/OS installation for IoT based design.

Pre-requisite: Raspberry Pi board layout

Objective:

1. To introduce Raspberry Pi.
2. To understand steps required to install operating system on Raspberry Pi.
3. To prepare the Raspberry-Pi development board ready for experiments

Components and equipment required:

Raspberry Pi Board, SD Card, Monitor

Theory:

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. Raspberry Pi is an indispensable single-board credit card size computer that comes in handy for a lot of work.

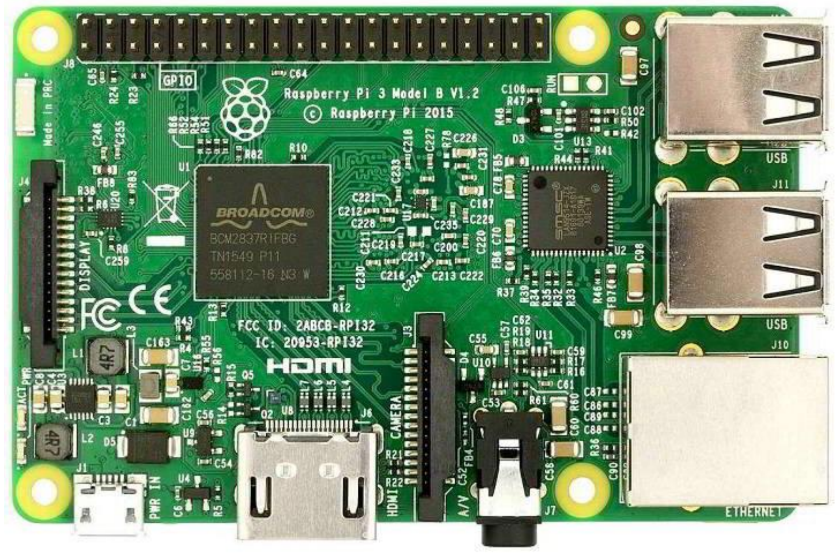


Figure 1.1: Raspberry Pi 3 Board

Raspberry Pi is popularly used for real time Image/Video Processing, IoT based applications and Robotics applications. Raspberry Pi is slower than laptop or desktop but is still a computer which can provide all the expected features or abilities, at a low power consumption. Raspberry Pi Foundation officially provides Debian based Raspbian OS. Also, they provide NOOBS OS for Raspberry Pi. We can install several Third-Party versions of OS like Ubuntu, Archlinux, RISC OS, Windows 10 IOT Core, etc.

Raspbian OS is official Operating System available for free to use. This OS is efficiently optimized to use with Raspberry Pi. Raspbian have GUI which includes tools for Browsing, Python programming, office, games, etc. We should use SD card (minimum 8 GB recommended) to store the OS (operating System). Raspberry Pi is more than computer as it provides access to the on-chip hardware i.e. GPIOs for developing an application. By accessing GPIO, we can connect devices like LED, motors, sensors, etc and can control them too.

It has ARM based Broadcom Processor SoC along with on-chip GPU (Graphics Processing Unit). The CPU speed of Raspberry Pi varies from 700 MHz to 1.2 GHz. Also, it has on-board SDRAM that ranges from 256 MB to 1 GB. Raspberry Pi also provides on-chip SPI, I2C, I2S and UART modules.

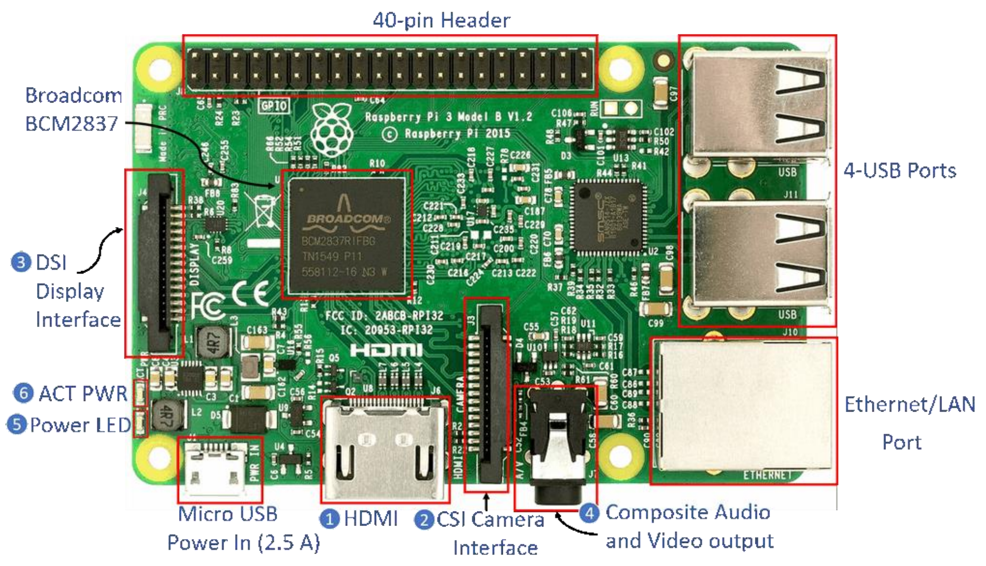


Figure 1.2: Raspberry Pi 3 Model B Hardware

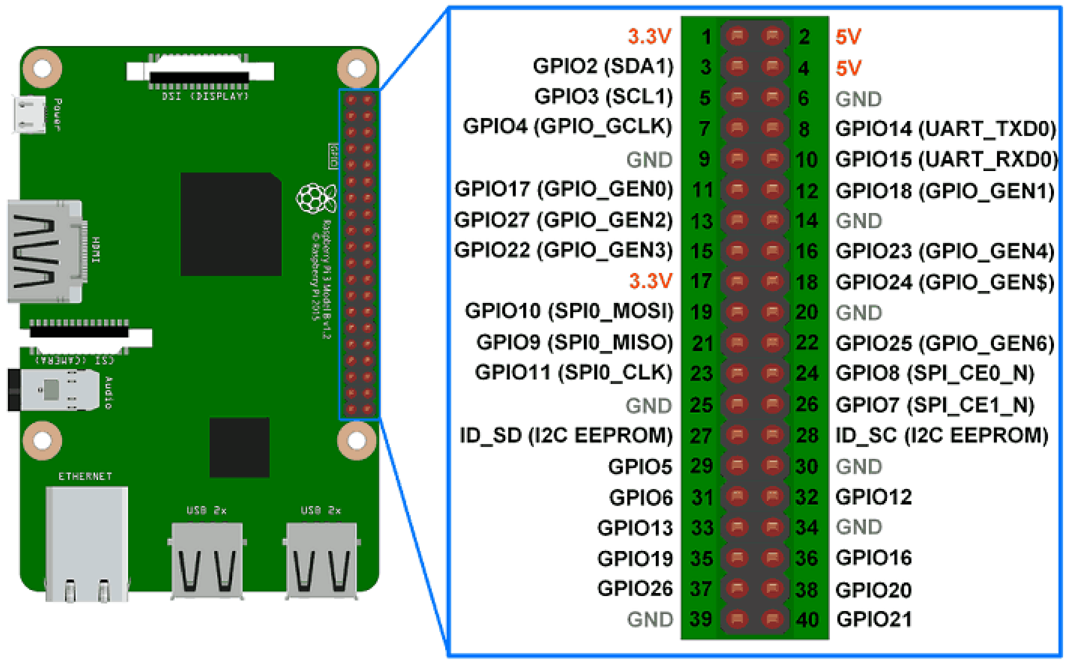

DSI (DISPLAY)
Power
HDMI
CSI (CAMERA)
Audio
ETHERNET
USB 2x
USB 2x
GPIO
Raspberry Pi 3 Model B v1.2 © Raspberry Pi 2015
3.3V 1 2 5V
GPIO2 (SDA1) 3 4 5V
GPIO3 (SCL1) 5 6 GND
GPIO4 (GPIO_GCLK) 7 8 GPIO14 (UART_TXD0)
GND 9 10 GPIO15 (UART_RXD0)
GPIO17 (GPIO_GEN0) 11 12 GPIO18 (GPIO_GEN1)
GPIO27 (GPIO_GEN2) 13 14 GND
GPIO22 (GPIO_GEN3) 15 16 GPIO23 (GPIO_GEN4)
3.3V 17 18 GPIO24 (GPIO_GEN$)
GPIO10 (SPI0_MOSI) 19 20 GND
GPIO9 (SPI0_MISO) 21 22 GPIO25 (GPIO_GEN6)
GPIO11 (SPI0_CLK) 23 24 GPIO8 (SPI_CE0_N)
GND 25 26 GPIO7 (SPI_CE1_N)
ID_SD (I2C EEPROM) 27 28 ID_SC (I2C EEPROM)
GPIO5 29 30 GND
GPIO6 31 32 GPIO12
GPIO13 33 34 GND
GPIO19 35 36 GPIO16
GPIO26 37 38 GPIO20
GND 39 40 GPIO21

Figure 1.3: Raspberry Pi 3 Model B GPIO Pinout

1. **HDMI (High-Definition Multimedia Interface):** It is used for transmitting uncompressed video or digital audio data to the Computer Monitor, Digital TV, etc. Generally, this HDMI port helps to connect Raspberry Pi to the Digital television.
2. **CSI Camera Interface:** CSI (Camera Serial Interface) interface provides a connection in between Broadcom Processor and Pi camera. This interface provides electrical connections between two devices.

3. **DSI Display Interface:** DSI (Display Serial Interface) Display Interface is used for connecting LCD to the Raspberry Pi using 15-pin ribbon cable. DSI provides fast High-resolution display interface specifically used for sending video data directly from GPU to the LCD display.
4. **Composite Video and Audio Output:** The composite Video and Audio output port carries video along with audio signal to the Audio/Video systems.
5. **Power LED:** It is a RED coloured LED which is used for Power indication. This LED will turn ON when Power is connected to the Raspberry Pi. It is connected to 5V directly and will start blinking whenever the supply voltage drops below 4.63V.
6. **ACT PWR:** ACT PWR is Green LED which shows the SD card activity.

Procedure:

- To get started with Raspberry Pi, we have to store required OS on SD card.
- Now to store OS on SD card we need to install OS on SD card. If you want to know how to install/store OS on SD card you can refer Installing Operating System Image on SD card.
- Here, we installed the Raspbian OS on SD card.
- Now, we have an SD card with installed OS and Raspberry Pi Board.
- Initially to use raspberry Pi we need computer monitor or Digital Display.
- We can directly connect Raspberry Pi to the Digital Display using HDMI cable.

Figure 1.4: HDMI Cable

- But, if we have a computer monitor (VGA Display), then we need an HDMI to VGA converter along with a VGA cable for connecting Raspberry Pi with monitors. HDMI to VGA converter and VGA cable is shown below.

Figure 1.5: HDMI to VGA Converter

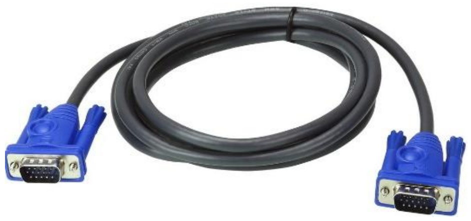
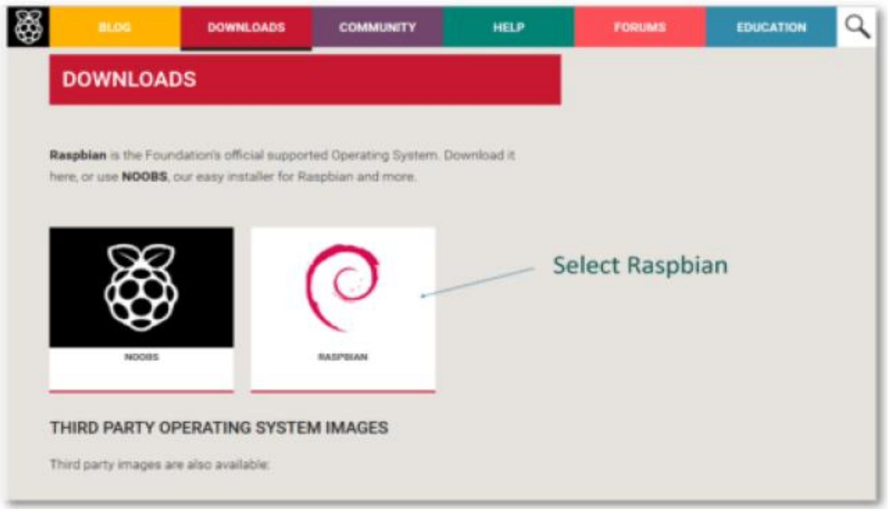


Figure 1.6: VGA Cable

Configuring Raspberry Pi

1. The first step in configuring the raspberry pi would be to install the raspbian operating system. Go to https://www.raspberrypi.org/downloads/ and select the raspbian O.S.

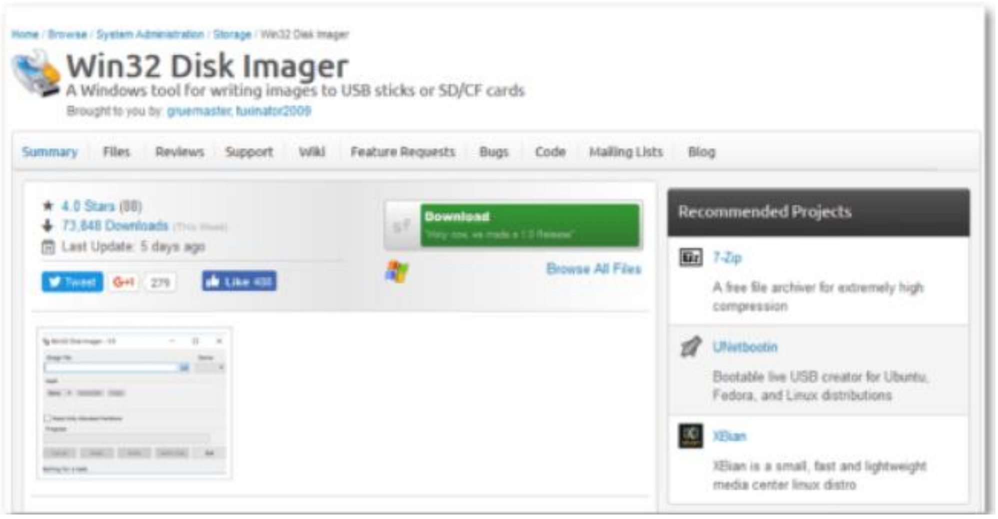


Raspbian is the Foundation's official supported Operating System. Download it here, or use **NOOBS**, our easy installer for Raspbian and more.

THIRD PARTY OPERATING SYSTEM IMAGES

Third party images are also available:

2. Once you download the raspbian operating system, you would need to format the S.D card and flash the raspbian O.S onto your S.D card. So, you would require a disk imaging software.



Win32 Disk Imager

A Windows tool for writing images to USB sticks or SD/CF cards

Brought to you by: gruemaster, tuxinator2009

4.0 Stars (88)

73,848 Downloads (This Week)

Last Update: 5 days ago

7-Zip

A free file archiver for extremely high compression

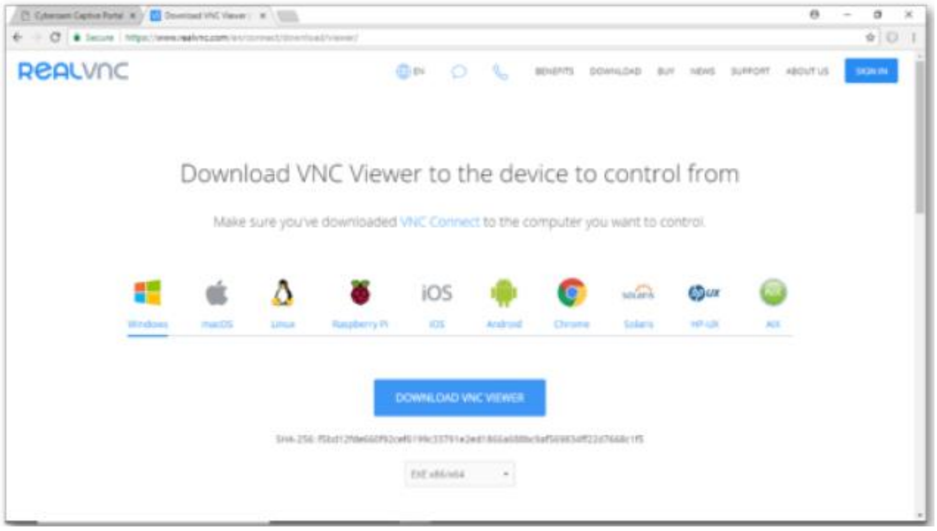
UNetbootin

Bootable live USB creator for Ubuntu, Fedora, and Linux distributions

XBian

XBian is a small, fast and lightweight media center linux distro

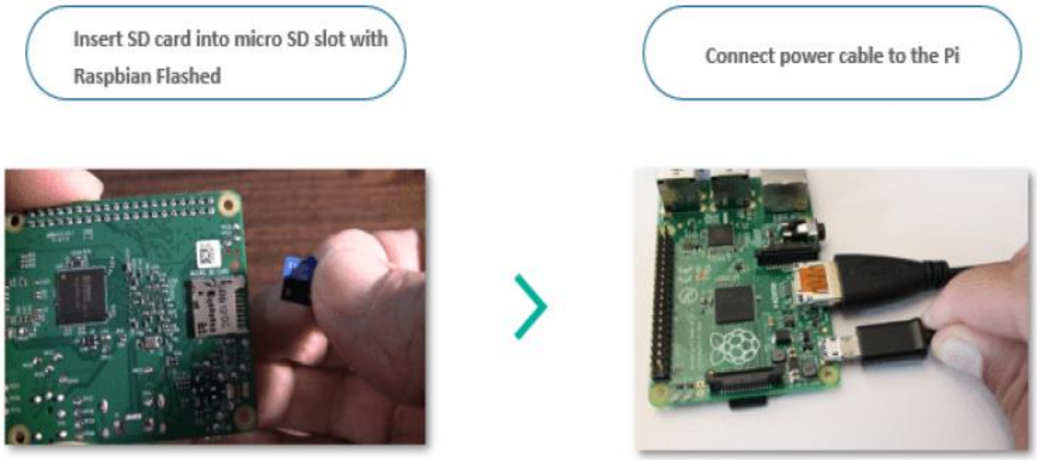
3. We would also require a graphical desktop sharing system, so that we can control the Raspberry Pi with a graphical user interface. One such graphical desktop sharing software is VNC Viewer. So, go ahead and download VNC Viewer.


RealVNC
BENEFITS DOWNLOAD BUY NEWS SUPPORT ABOUT US SIGN IN
Windows macOS Linux Raspberry Pi iOS Android Chrome Solaris HP-UX AIX
DOWNLOAD VNC VIEWER
EXE x86/x64

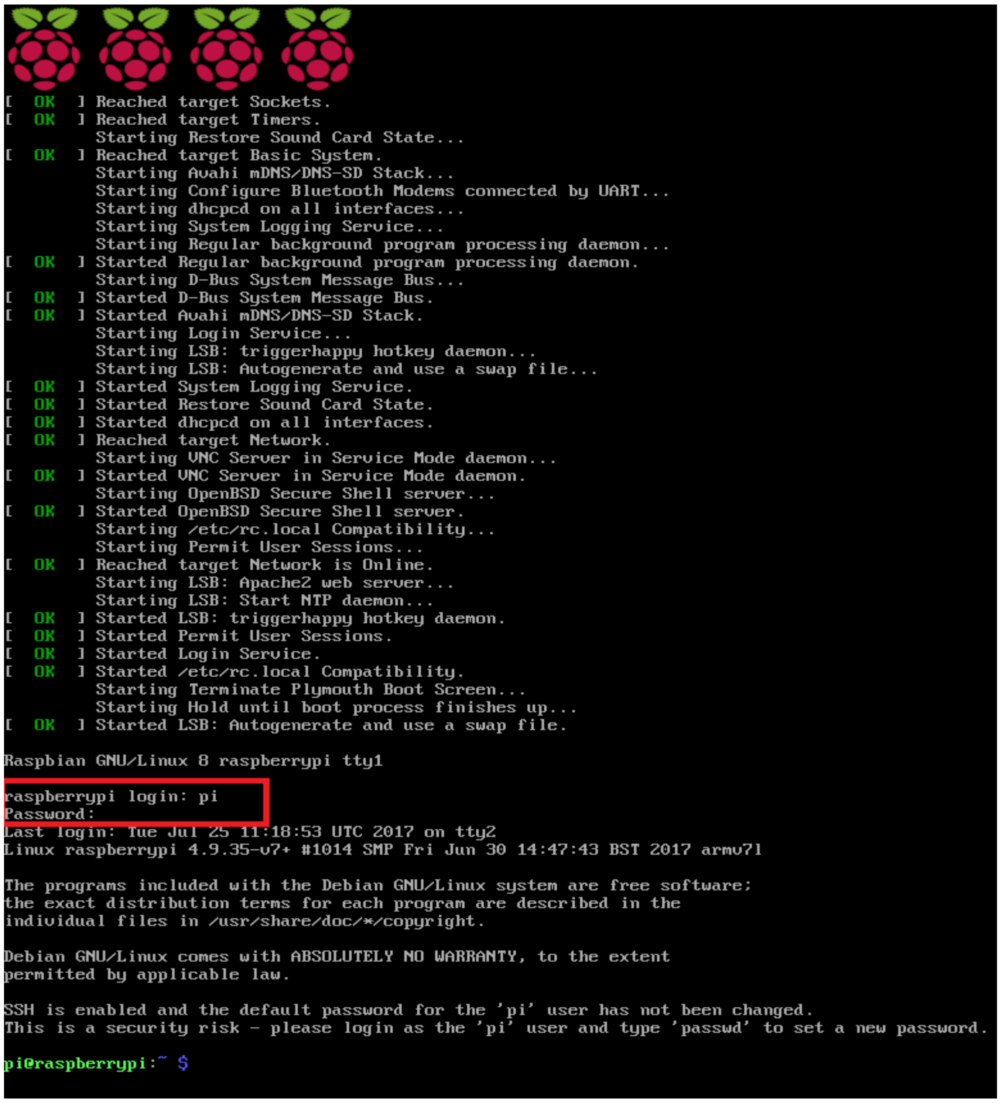
Download VNC Viewer to the device to control from

Make sure you've downloaded VNC Connect to the computer you want to control.

4. After completing all of the above steps, finally insert the SD card into the micro SD slot and connect the power cable to the Pi.


Insert SD card into micro SD slot with Raspbian Flashed
Connect power cable to the Pi

5. Next, connect the Raspberry Pi to the monitor & plug-in the power cable. Click on install Raspbian and follow the instructions to install the Raspbian OS on Raspberry Pi.
6. Next time when you connect the power source to the Raspberry Pi, it'll start searching for open Wi-Fi networks. So, we'll turn on the mobile hot-spot in our system and let it connect. Once, the raspberry pi is connected to our Wi-Fi, we'll take the dynamic IP of the Raspberry Pi and feed it into the VNC viewer dialog box, and we can start working with Pi.
7. Now, connect the Raspberry Pi to the Display/monitor and Power-On Raspberry Pi. We will get a Black command window asking for Login and Password as shown below



```
[  OK  ] Reached target Sockets.
[  OK  ] Reached target Timers.
         Starting Restore Sound Card State...
[  OK  ] Reached target Basic System.
         Starting Avahi mDNS/DNS-SD Stack...
         Starting Configure Bluetooth Modems connected by UART...
         Starting dhcpcd on all interfaces...
         Starting System Logging Service...
         Starting Regular background program processing daemon...
[  OK  ] Started Regular background program processing daemon.
         Starting D-Bus System Message Bus...
[  OK  ] Started D-Bus System Message Bus.
[  OK  ] Started Avahi mDNS/DNS-SD Stack.
         Starting Login Service...
         Starting LSB: triggerhappy hotkey daemon...
         Starting LSB: Autogenerate and use a swap file...
[  OK  ] Started System Logging Service.
[  OK  ] Started Restore Sound Card State.
[  OK  ] Started dhcpcd on all interfaces.
[  OK  ] Reached target Network.
         Starting VNC Server in Service Mode daemon...
[  OK  ] Started VNC Server in Service Mode daemon.
         Starting OpenBSD Secure Shell server...
[  OK  ] Started OpenBSD Secure Shell server.
         Starting /etc/rc.local Compatibility...
         Starting Permit User Sessions...
[  OK  ] Reached target Network is Online.
         Starting LSB: Apache2 web server...
         Starting LSB: Start NTP daemon...
[  OK  ] Started LSB: triggerhappy hotkey daemon.
[  OK  ] Started Permit User Sessions.
[  OK  ] Started Login Service.
[  OK  ] Started /etc/rc.local Compatibility.
         Starting Terminate Plymouth Boot Screen...
         Starting Hold until boot process finishes up...
[  OK  ] Started LSB: Autogenerate and use a swap file.

Raspbian GNU/Linux 8 raspberrypi tty1

raspberrypi login: pi
Password:
Last login: Tue Jul 25 11:18:53 UTC 2017 on tty2
Linux raspberrypi 4.9.35-v7+ #1014 SMP Fri Jun 30 14:47:43 BST 2017 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set a new password.

pi@raspberrypi:~ $
```

8. Then, use the following login name and password

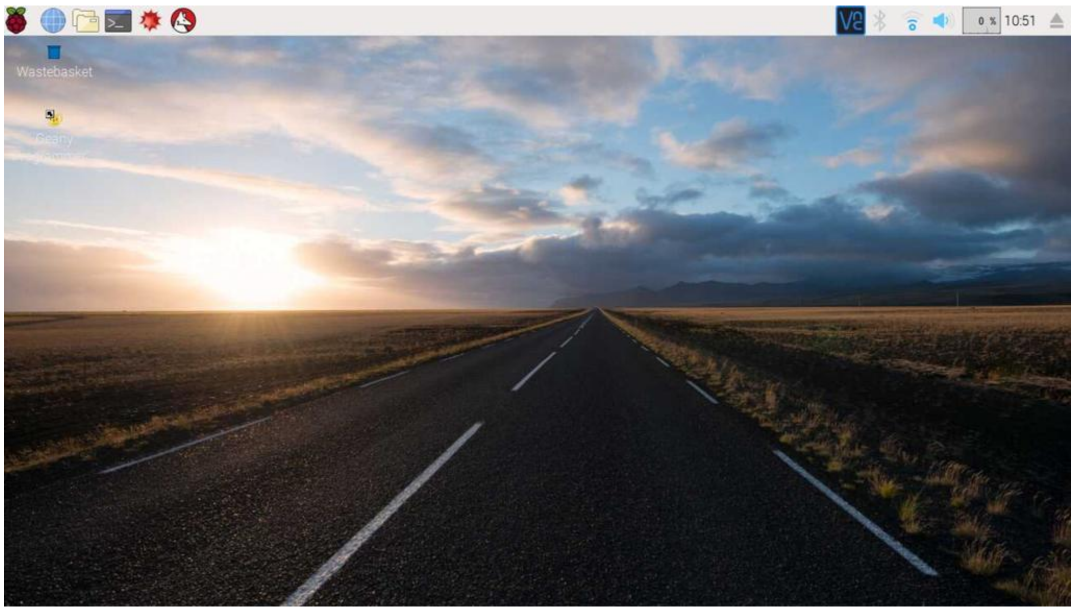
raspberrypi Login: pi

Password: raspberry

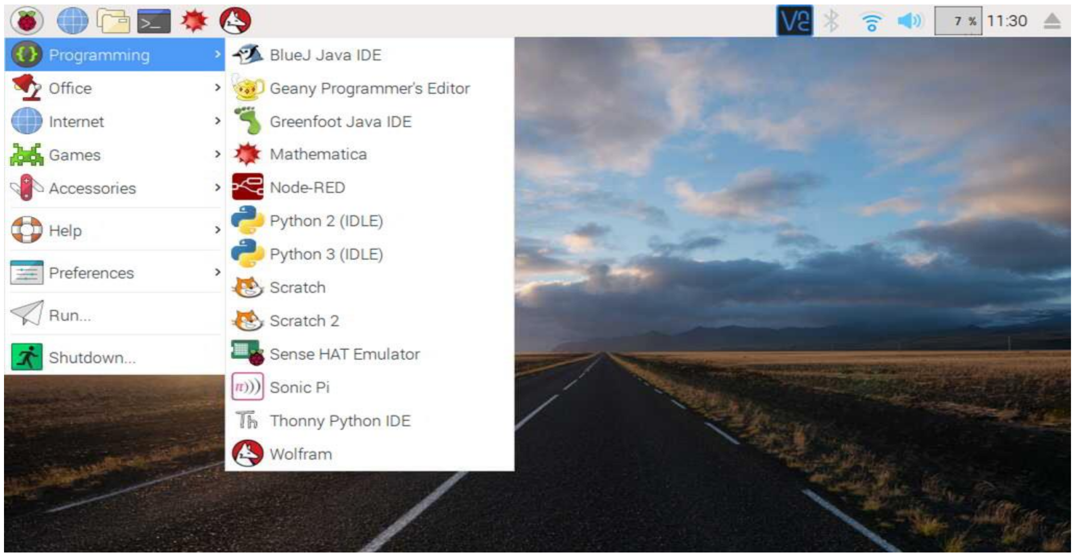
9. This is the default user name and password. You can change the password after the first login. The above command window can be used to operate Raspberry Pi.
10. To get GUI environment on Raspberry Pi, use below command,

```
startx
```

11. And we will get Home Screen of Raspberry Pi as shown below:



12. On display, there is a symbol of raspberry to the top-left corner of display. After clicking on it, we will get menu as shown below,



13. As we can see, the Raspbian OS has installed Python 2 & 3. It also has different programming IDE like Geany, BlueJ Java IDE, etc. As raspberry pi 3 has On-chip WiFi, we can connect it to the network and will get access over Internet.
14. We can also change password of "Pi" user.

Conclusion:

Post Lab Questions:

1. State briefly IoT platforms available with short note.
2. List and State IoT Cloud platforms.
3. What are the main components of an IoT network?
4. List out various versions of Raspberry-Pi available
5. Compare more prominently used Raspberry Pi with their features

Additional links for more information:

1. Raspberry Pi Beginner's Guide: Install and Setup NOOBS
   https://www.youtube.com/watch?v=wvxCNQ5AYPg
2. Getting started with Raspberry-pi installation
   https://www.raspberrypi.org/blog/getting-started-raspberry-pi/
3. Building IoT with Raspberry Pi
   https://www.blemobileapps.com/blog/building-internet-things-iot-raspberry-pi/

Third Year B. Tech (ECE)

Semester: VI **Subject:** IoT Architectures and Protocols

Name: **Class:**

Roll No: **Batch:**

Experiment No: 02

Name of the Experiment: Interfacing Sensors and Actuators to Raspberry Pi

Performed on:

Submitted on:

Marks	Teacher's Signature with date

Aim: To Interface sensors and actuators to Raspberry pi

Prerequisite: Specifications of various sensors and actuators, Raspberry Pi 3 Model Board layout

Objective:

1. To understand sensor interfacing with Raspberry pi
2. To understand actuator interfacing with Raspberry pi
3. To sense the physical quantity using sensor
4. Control the actuator depending on the sensor data

Components and equipment required:

Raspberry Pi 3 Model, LED, Resistors, Sensors, Actuators, USB Cable, HDMI to VGA converter cable, Breadboard etc.

Theory:

DHT11 Interfacing with Raspberry Pi

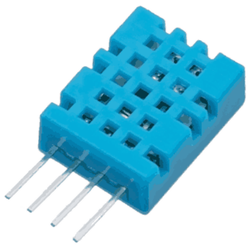


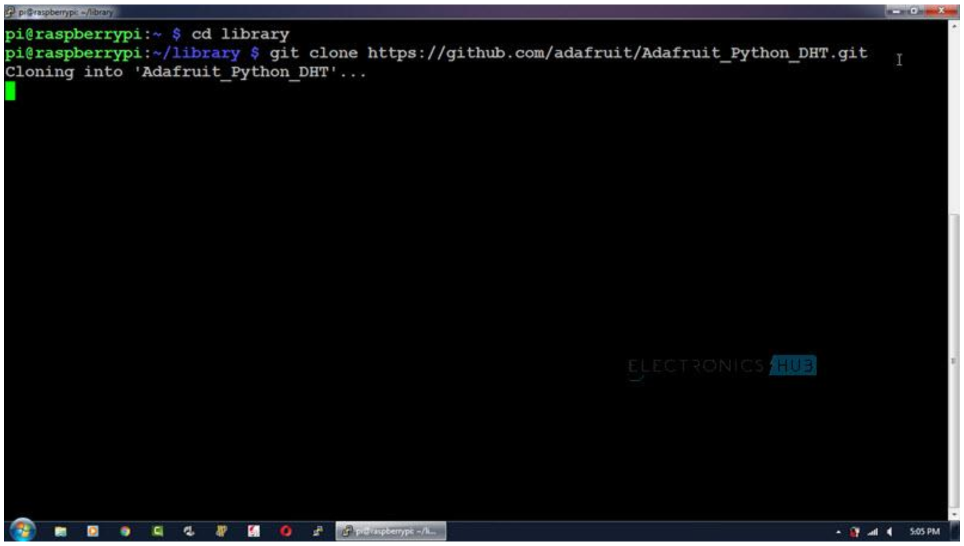
Figure 2.1: DHT11 Sensor

Figure 2.1 shows the DHT11 Sensor. DHT11 sensor measures and provides humidity and temperature values serially over a single wire. It can measure relative humidity in percentage (20 to 90% RH) and temperature in degree Celsius in the range of 0 to 50°C. It has 4 pins; one of which is used for data communication in serial form. Pulses of different TON and TOFF are decoded as logic 1 or logic 0 or start pulse or end of a frame. DHT11 is a Digital Sensor consisting of two different sensors in a single package. The sensor contains an NTC (Negative Temperature Coefficient) Temperature Sensor, a Resistive-type Humidity Sensor and an 8-bit Microcontroller to convert the analog signals from these sensors and produce a Digital Output.

Installing DTH11 Library

Since we are using a library called Adafruit. DHT provided by Adafruit for this project, we need to first install this library into Raspberry Pi. First step is to download the library from GitHub. But before this, I have created a folder called 'library' on the desktop of the Raspberry Pi to place the downloaded files. Now, enter the following command to download the files related to the Adafruit_DHT library.

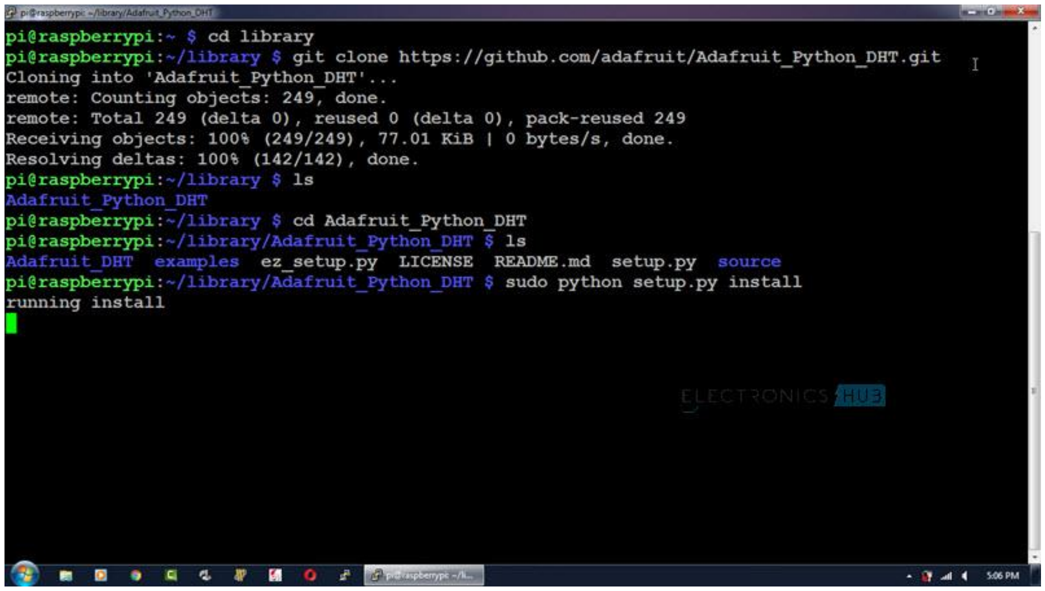
git clone https://github.com/adafruit/Adafruit_Python_DHT.git



```
pi@raspberrypi:~ $ cd library
pi@raspberrypi:~/library $ git clone https://github.com/adafruit/Adafruit_Python_DHT.git
Cloning into 'Adafruit_Python_DHT'...
```

All the contents will be downloaded to a folder called 'Adafruit_Python_DHT'. Open this directory using cd Adafruit_Python_DHT. To see the contents of this folder, use 'ls' command. In that folder, there is file called 'setup.py'. We need to install this file using the following command.

sudo python setup.py install



```
pi@raspberrypi:~ $ cd library
pi@raspberrypi:~/library $ git clone https://github.com/adafruit/Adafruit_Python_DHT.git
Cloning into 'Adafruit_Python_DHT'...
remote: Counting objects: 249, done.
remote: Total 249 (delta 0), reused 0 (delta 0), pack-reused 249
Receiving objects: 100% (249/249), 77.01 KiB | 0 bytes/s, done.
Resolving deltas: 100% (142/142), done.
pi@raspberrypi:~/library $ ls
Adafruit_Python_DHT
pi@raspberrypi:~/library $ cd Adafruit_Python_DHT
pi@raspberrypi:~/library/Adafruit_Python_DHT $ ls
Adafruit_DHT  examples  ez_setup.py  LICENSE  README.md  setup.py  source
pi@raspberrypi:~/library/Adafruit_Python_DHT $ sudo python setup.py install
running install
```

Interfacing Diagram

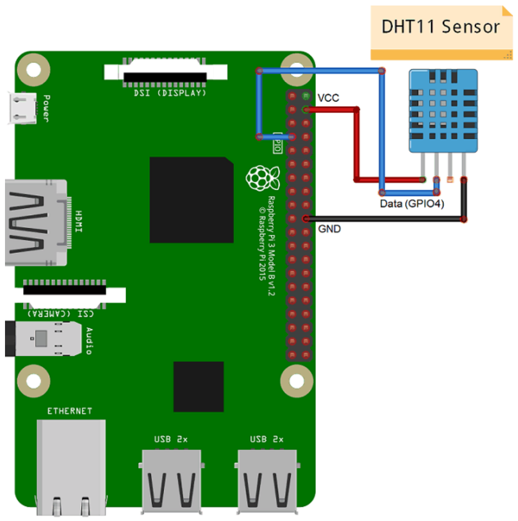


Figure 2.2: DHT11 Sensor interfacing with Raspberry Pi

Procedure:

1. Use the DHT Sensor Python library by Adafruit from GitHub.
2. The Adafruit Python DHT Sensor library is created to read the Humidity and Temperature on raspberry Pi or Beaglebone Black.
3. It is developed for DHT series sensors like DHT11, DHT22 or AM2302.
4. Download Adafruit DHT Sensor library from here.
5. Extract the library and install it in the same root directory of downloaded library by executing following command,
   sudo python setup.py install

6. Once the library and its dependencies has been installed, open the example sketch named simpletest from the library kept in examples folder.
7. In this code, raspberry Pi reads Humidity and Temperature from DHT11 sensor and prints them on terminal. However it read and display the value only once. So, here we made change in the program to print value continuously.
8. Assign proper sensor type to the sensor variable in this library. Here, we are using DHT11 sensor.
   sensor = Adafruit_DHT.DHT11
9. If anyone is using sensor DHT22 then we need to assign Adafruit_DHT. DHT22 to the sensor variable shown above.
10. Also, comment out Beaglebone pin definition and uncomment pin declaration for Raspberry Pi.
11. Then assign pin no. to which DHT sensor's data pin is connected. Here, data out of DHT11 sensor is connected to GPIO4. As shown in above interfacing diagram.

.

Conclusion:

Post Lab Questions:

1. List out various sensors and explain any one in brief
2. List out various actuators and explain any one in brief
3. Explain in detail Adafruit Python Library
4. State the applications of any one sensors and one actuator
5. What are different operating systems used with Raspberry Pi?

Additional links for more information:

1. https://www.electronicshub.org/raspberry-pi-dht11-humidity-temperature-sensor-interface/

Third Year B. Tech (ECE)

Semester: VI **Subject:** IoT Architectures and Protocols

Name: **Class:**

Roll No: **Batch:**

Experiment No: 03

Name of the Experiment: To Study and implement RFID/NFC based application

Performed on:

Submitted on:

Marks	Teacher's Signature with date

Aim: To Study and implement RFID/NFC based application

Prerequisite: RFID System, Raspberry Pi 3 Model layout and Pin configuration

Objective:

1. To identify pin configuration of Raspberry Pi to interface with RFID
2. To analyse the steps to interface RFID with Raspberry Pi.
3. To verify the reception of data

Components and equipment required:

Raspberry Pi 3, Micro SD Card, Power Supply, RC522 RFID Reader with tags, Breadboard, Connecting Wires

Theory:

RFID system is made up of two parts: a tag or label and a reader. RFID tags or labels are embedded with a transmitter and a receiver. The RFID component on the tags have two parts: a microchip that stores and processes information, and an antenna to receive and transmit a signal. The tag contains the specific serial number for one specific object.

To read the information encoded on a tag, a two-way radio transmitter-receiver called an interrogator or reader emits a signal to the tag using an antenna. The tag responds with the information written in its memory bank. The interrogator will then transmit the read results to an RFID computer program.

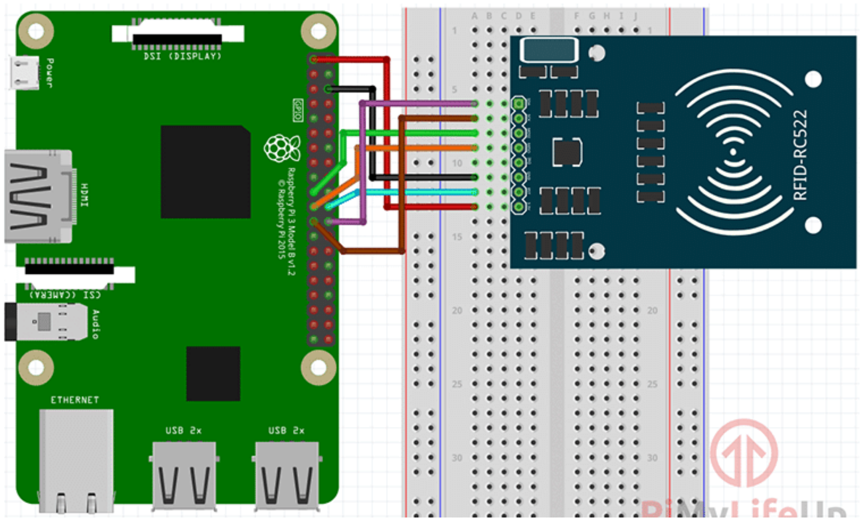
Wiring the RFID RC522

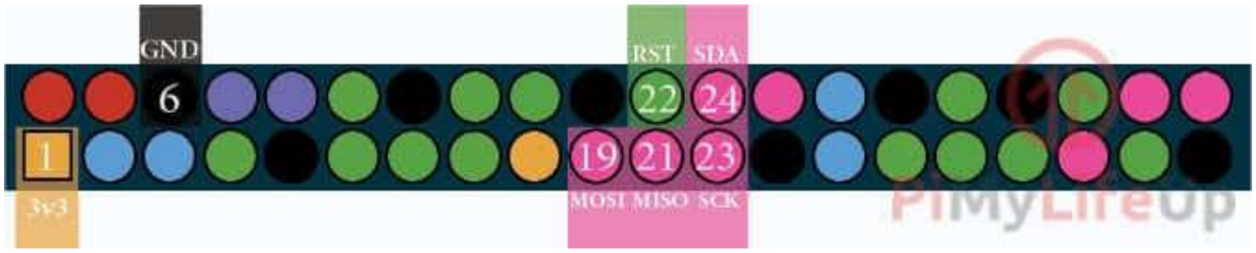
On your RFID RC522 you will notice that there are 8 possible connections on it, these being SDA (Serial Data Signal), SCK (Serial Clock), MOSI (Master Out Slave In), MISO (Master In

Slave Out), IRQ (Interrupt Request), GND (Ground Power), RST (Reset-Circuit) and 3.3v (3.3v Power In). We will need to wire all of these but the IRQ to our Raspberry Pi's GPIO pins. Either wire these directly to the GPIO Pins or like we did in this tutorial, plug the RFID RC522 into our Breadboard then wire from there to our Raspberry Pi's GPIO Pins.
Wiring your RFID RC522 to your Raspberry Pi is fairly simple, with it requiring you to connect just 7 of the GPIO Pins directly to the RFID reader. Follow the table below, and check out our GPIO guide to see the positions of the GPIO pins that you need to connect your RC522 to.

- **SDA** connects to **Pin 24**.
- **SCK** connects to **Pin 23**.
- **MOSI** connects to **Pin 19**.
- **MISO** connects to **Pin 21**.
- **GND** connects to **Pin 6**.
- **RST** connects to **Pin 22**.
- **3.3v** connects to **Pin 1**.

By default, the Raspberry Pi has the SPI (Serial Peripheral Interface) disabled, which is a bit of a problem as that is what our RFID reader circuit runs through.





Procedure:

1. Let's begin by first opening the raspi-config tool, and we can do this by opening the terminal and running the following command.

```
sudo raspi-config
```

2. This tool will load up a screen showing a variety of different options. If you want a more in-depth look into these options, you can check out our raspi-config guide.
3. Now on this next screen, you want to use your arrow keys to select "P4 SPI", again press Enter to select the option once it is highlighted.
4. You will now be asked if you want to enable the SPI Interface, select Yes with your arrow keys and press Enter to proceed. You will need to wait a little bit while the raspi-config tool does its thing in enabling SPI.
5. Once the SPI interface has been successfully enabled by the raspi-config tool you should see the following text appear on the screen, "The SPI interface is enabled".
6. Before the SPI Interface is fully enabled we will first have to restart the Raspberry Pi. To do this first get back to the terminal by pressing Enter and then ESC.
7. Type the following Linux command into the terminal on your Raspberry Pi to restart your Raspberry Pi.
   ```
   sudo reboot
   ```
8. Once your Raspberry Pi has finished rebooting, we can now check to make sure that it has in fact been enabled. The easiest way to do this is to run the following command to see if spi_bcm2835 is listed.
   ```
   lsmod | grep spi
   ```
9. If for some reason the SPI module has not activated, we can edit the boot configuration file manually by running the following command on our Raspberry Pi.
   ```
   sudo nano /boot/config.txt
   ```
10. Within the configuration file, use Ctrl + W to find "dtparam=spi=on".
    If found it, check to see if there is a # in front of it. If there is, remove it as this is commenting out the activation line. If can't find the line at all, add "dtparam=spi=on" to the bottom of the file. Once made the changes, press Ctrl + X then pressing Y and then Enter to save the changes. Now proceed from Step 5 again, rebooting Raspberry Pi then checking to see if the module has been enabled.

- ## Getting Python ready for the RFID RC522

Now wired-up RFID RC522 circuit to the Raspberry Pi, now power it on and begin the process of programming simple scripts in Python to interact with the chip. Let's check how to read data from the RFID chips and how to write to them.

1. First need to update Raspberry Pi to ensure it's running the latest version of all the software. Run the following two commands on your Raspberry Pi to update it.
   ```
   sudo apt-get update
   sudo apt-get upgrade
   ```
2. Now the final thing we need before we can proceed is to install python3-dev, python-pip and git packages. Simply run the following command on your Raspberry Pi to install all of the required packages for this guide on setting up RFID reader.
   ```
   sudo apt-get install python3-dev python3-pip
   ```

3. To begin, first install the Python Library spidev to Raspberry Pi using the python "pip" tool that has been downloaded in the previous step. The spidev library helps handle interactions with the SPI and is a key component to this tutorial as we need it for the Raspberry Pi to interact with the RFID RC522. Run the following command on Raspberry Pi to install spidev to Raspberry Pi through pip. sudo is used here to ensure that the package is installed so that all users can utilize it and not just the current user.

```
sudo pip3 install spidev
```

4. Now installed the spidev library to Raspberry Pi, now proceed to installing the MFRC522 library using pip as well. There are two files that are included within our MFRC522 library that we make use of:

MFRC522.py which is an implementation of the RFID RC522 interface, this library handles all the heavy lifting for talking with the RFID over the Pi's SPI Interface.

SimpleMFRC522.py that takes the MFRC522.py file and greatly simplifies it by making you only have to deal with a couple of functions instead of several.

To install the MFRC522 library to your Raspberry Pi using pip go ahead and run the following command.

```
sudo pip3 install mfrc522
```

With the library now saved to Raspberry Pi, start programming for RFID RC522.

Writing with the RFID RC522

1. Now let's start off by making a folder where we will be storing our couple of scripts. Let's call this folder "pi-rfid", create it by running the following command.

```
mkdir ~/pi-rfid
```

2. Begin by changing directory into newly cloned folder, and begin writing Write.py Python script.

```
cd ~/pi-rfid
sudo nano Write.py
```

3. Within this file, write the following lines of code. This code will basically ask you for text to input and then write that text to the RFID Tag.

```
#!/usr/bin/env python
import RPi.GPIO as GPIO
from mfrc522 import SimpleMFRC522
```

The first line of this segment of code helps tell the terminal how to interpret the file, and it lets it know that it should use Python when executing it and not something else such as Bash.

```
reader = SimpleMFRC522()
```

This line creates a copy of the SimpleMFRC522 as an object, runs its setup function then stores it all in our reader variable.

```
try:
        text = input('New data:')
        print("Now place your tag to write")
        reader.write(text)
        print("Written")
finally:
        GPIO.cleanup()
```

4. Save the file by pressing **Ctrl + X** then pressing **Y** and then finally hitting **Enter**.
5. Before testing out the script make sure that you have an RFID tag handy. Once ready, type the following command into your Raspberry Pi’s terminal.
   ```
   sudo python3 Write.py
   ```
6. With that done, simply place your RFID Tag on top of your RFID RC522 circuit. As soon as it detects it, it will immediately write the new data to the tag. You should see “Written” appear in your command line if it was successful.

Reading with the RFID RC522

Now that we have written our script to write to RFID tags using our RC522 we can now write a script that will read this data back off the tag.

1. Let’s start off by changing the directory to make sure we are in the right place, and then we can run nano to begin writing our Read.py script.
   ```
   cd ~/pi-rfid
   sudo nano Read.py
   ```
2. Within this file, write the following lines of code. This script will basically sit and wait till you put your RFID tag on the RFID RC522 reader, it will then output the data it reads off the tag.
   ```
   reader = SimpleMFRC522()
   #!/usr/bin/env python
   import RPi.GPIO as GPIO
   from mfrc522 import SimpleMFRC522
   reader = SimpleMFRC522()

   try:
        id, text = reader.read()
     print(id)
     print(text)
   finally:
           GPIO.cleanup()
   ```
3. Now finished Read.py script and need to test it out. Before we test out the script, grab one of the RFID tags that you want to read. Once that you are ready, type the following command into your Raspberry Pi’s terminal.
   ```
   sudo python3 Read.py
   ```
4. With the script now running, all need to do is place your RFID Tag on top of your RFID RC522 circuit. As soon as the Python script detects the RFID tag being placed on top, it will immediately read the data and print it back out to you.
   An example of what a successful output would look like is displayed below.
   ```
   pi@raspberrypi:~/pi-rfid $ sudo python3 Read.py
   827843705425
   ```

`Pimylifeup`

5. If you successfully receive data back from your Read.py script with the text that you pushed to the card using your Write.py script then you have successfully set up your Raspberry Pi to connect with your RFID RC522 Circuit.

Conclusion:

Post Lab Questions:

1. What is RFID? How do RFID system work?
2. What sort of RFID technology used in library?
3. Does the use of RFID pose any health problems? Explain
4. Explain the backscatter concept

Additional links for more information:

1. https://pimylifeup.com/raspberry-pi-rfid-rc522/
2. https://www.elprocus.com/rfid-basic-introduction-simple-application/
3. https://www.atlasrfidstore.com/rfid-beginners-guide/

Third Year B. Tech (ECE)

Semester: VI **Subject:** IoT Architectures and Protocols

Name: **Class:**

Roll No: **Batch:**

Experiment No: 04

Name of the Experiment: To Study and implement ZigBee/BLE protocol using hardware platform.

Performed on:

Submitted on:

Marks	Teacher's Signature with date

Aim: To Study and implement ZigBee/BLE protocol using hardware platform.

Prerequisite: Bluetooth and BLE technology, Python programming, Basics of Raspberry Pi Model 3

Objectives:

1. To setup on-board Bluetooth of Raspberry Pi
2. Transfer files between Raspberry pi and device

Components and equipment required:

Raspberry Pi 3 Model with 5V USB power supply, BLE device, smartphone.

Theory:

Bluetooth Low Energy (aka BLE/Bluetooth 4.0/Bluetooth Smart) is the most recent incarnation of Bluetooth technology developed by the Bluetooth SIG (the organization that maintains the specification). This communication protocol is designed for applications where data needs to be transferred in small amounts at relatively low speed while consuming low amounts of power (e.g., heart rate monitor, step counter, wireless keyboard). This latest version of the protocol is not compatible with its predecessor (Bluetooth classic), as an upside, long gone are the days where pairing devices was necessary!

The goal of this Instructable is to demonstrate how you can setup your Raspberry Pi to read and write data from Bluetooth Low Energy (BLE) devices nearby. Whether you want to read the number of steps from your fitbit, or your heart rate from your iWatch, or read/write any type of data from/to BLE devices.

The Raspberry Pi is a popular platform because of its low cost and high integration. In addition to Wi-Fi and Ethernet, this board also has integrated Bluetooth which support BLE. The Rapsberry Pi support for Bluetooth depends on specifically on which Raspberry Pi you have:

Board	Bluetooth Chipset	Bluetooth Supported
Raspberry Pi 3 Model A+	**Broadcom BCM43438**	**Bluetooth 4.1**
Raspberry Pi 3 Model B	**Broadcom BCM43438**	**Bluetooth 4.1**
Raspberry Pi Model 3B+	**Cypress CYW43455**	**Bluetooth 4.2**
Raspberry Pi 4 Model B	**Cypress CYW43455**	**Bluetooth 5.0**

Raspberry Pi supports Bluetooth Low Energy because they integrate a combo Wi-Fi + Bluetooth chipset. The exact chipset supported varies depending on the board being used. Raspberry Pi 3A used BCM43438 chipset from Broadcom, while it moved to a CYW43455 chipset with support for 802.11ac and dual band (2.4GHz and 5GHz).

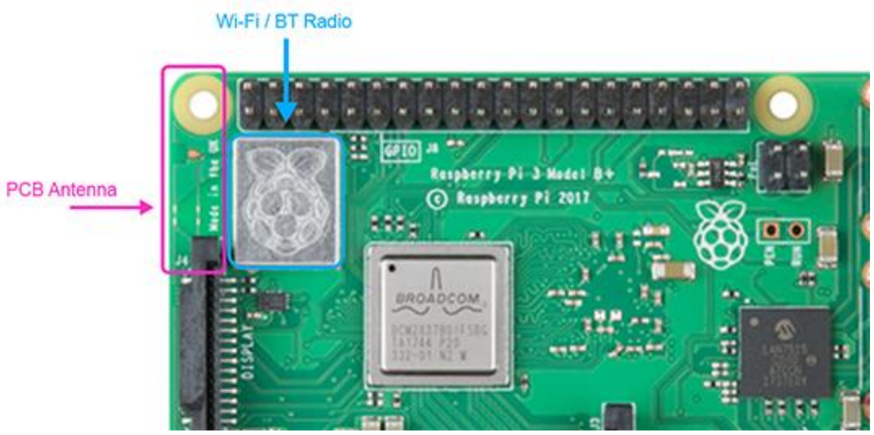
Each version of the Raspberry Pi uses Linux and so leverages the Open Source BlueZ Bluetooth stack. This stack has been used for many years, and although it supports both Bluetooth Classic and BLE, its support for BLE is more recent and there are some limitations and sometimes bugs because it does not get as much testing. If you're thinking of deploying BlueZ, you should most definitively test it extensively in a real-world application.

Bluetooth 5.0 support on the Raspberry Pi is limited to the mandatory features. The main one that is missing is the Long-Range support, so if you were thinking of leveraging the Coded PHY for extra-long range, you won't be able to. This feature is normally not found on combo Wi-Fi + Bluetooth chipsets.

- **Raspberry Pi BLE Performance**

Raspberry Pi has pretty good chipsets from an RF performance and feature perspective, leveraging Broadcom and now Cypress. Some of devices have support for 5GHz Wi-Fi band reduces interference on the 2.4GHz band used by BLE. Raspberry Pi 3 has come up with a great on-board feature i.e. the on-board Bluetooth. So, there is no need to for external Bluetooth dongle. This frees up a USB port for other uses, which would otherwise be used up by the Bluetooth dongle.
Raspberry Pi 3 has BCM43438 highly integrated single chip which includes 2.4GHz WLAN, Bluetooth and FM receiver. Given that the system is a single chip solution, there is also support for Coexistence to reduce interference. The biggest issue the Raspberry Pi has a relatively small

antenna. This antenna is surrounded somewhat by the GPIO connector pin connector. Both of these have some impact on performance, though the RPI foundation does not provide detailed antenna radiation information.

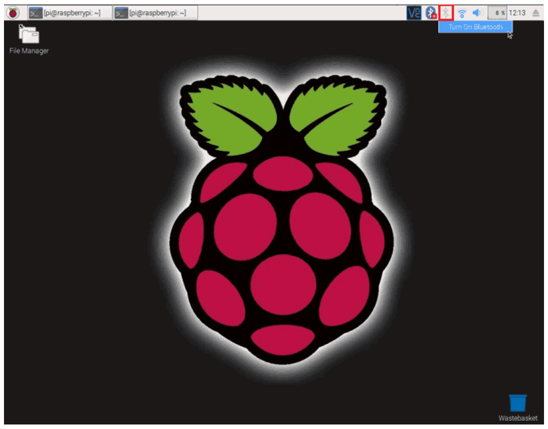


Ultimately, the Bluetooth LE range of the Raspberry Pi devices is decent but limited, and it won't approach what you can get with an external antenna and a more sophisticated system. If you're looking to control devices over any reasonable distances beyond a room or two, adding another radio or finding another solution would be best. It can make a good solution for indoor control of devices.

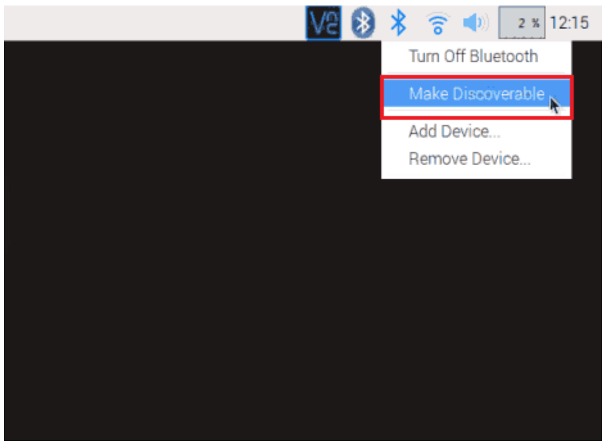
Procedure:

1. How to set up the On-board Bluetooth of Raspberry Pi

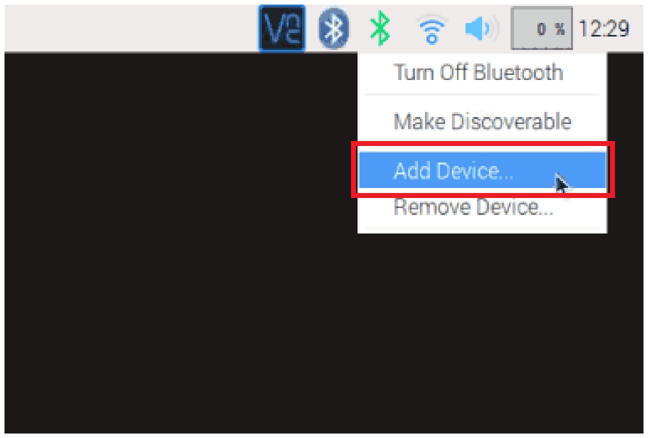
Raspberry Pi has on-board Bluetooth which can be used for communication or sending/receiving files. Before establishing communication between Raspberry Pi and a Bluetooth enabled device, we need to pair them. Pairing a Bluetooth device on Raspberry Pi is same as that on a mobile or Laptop. Turn-ON Bluetooth as follows,



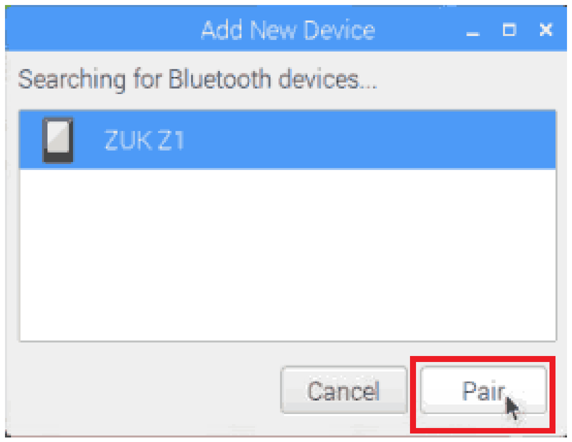
Then make it discoverable.



Now, select option Add Device…



After selecting Add device… pop-up window appear which will be as follows,



In above window, we can see mobile Bluetooth device named “ZUK Z1”. Select device and then click on pair. This will prompt for pairing, and displays the pin (the pin that was randomly generated by raspberry pi) from Raspberry Pi to Bluetooth device, which should cause the device to ask you to confirm the connection.

After the device accepts the connection by using the pair option, our raspberry pi and the Bluetooth device will be paired and connection can be established between them. Now our Raspberry Pi is ready to communicate with the paired device.

2. How to transfer files between Raspberry Pi and a device via Bluetooth?

There are two ways for communicating between Raspberry Pi and a device via Bluetooth.

- Bluetooth GUI (Graphical User Interface)
- Command Line Interface (CLI)

Before using GUI or CLI for accessing Bluetooth based transfer, we need to install the following packages,

Blueman–It is a full featured Bluetooth manager. It provides GUI based setting panel **Bluetooth manager.**

Bluez–This package provides the Bluetooth protocol stack and the bluetoothctl utility.

Bluetooth–This package provides all the plugins supported by BluezBluetooth stack.

Now to use Bluetooth service install above packages using following command,

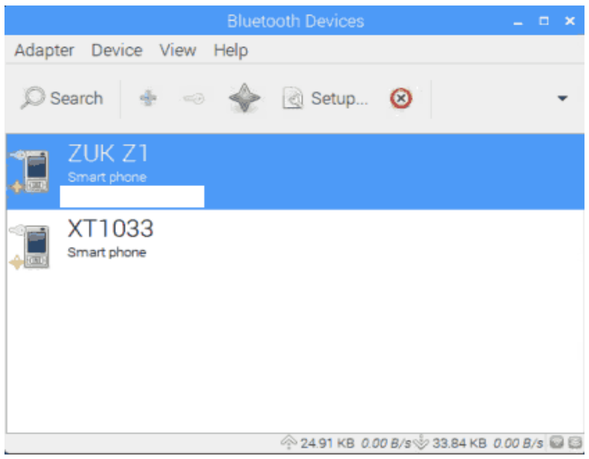
```
sudo apt-get install bluemanbluez Bluetooth
```

After successful installation of above packages, reboot Raspberry Pi.

3. Using Graphical User Interface (GUI) for Bluetooth Services

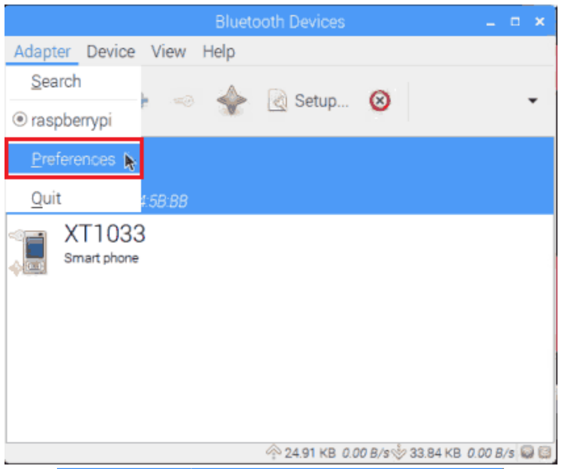
Raspberry Pi don't have GUI for accessing Bluetooth services like in laptop/mobile. To get GUI for Bluetooth, we installed packages mentioned above.

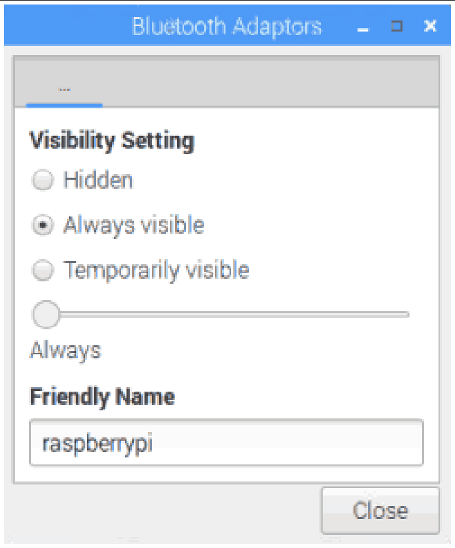
Now, we can access Bluetooth services (send/receive) using GUI. For this, go to the menu and select preferences. In preferences, select Bluetooth Manager. Now, we can see Bluetooth manager window with all the visible Bluetooth devices listed as shown below.



ZUK Z1
Smart phone
XT1033
Smart phone
24.91 KB 0.00 B/s 33.84 KB 0.00 B/s

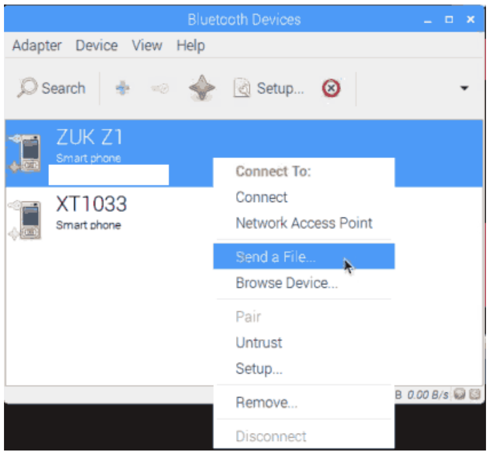
We need to make Bluetooth visible so that it is discoverable to other devices. To do this, select Adapter à preferences as shown below,





Visibility Setting
Hidden
[✓ Always visible]
Temporarily visible
Always
Friendly Name
raspberrypi

We can transfer files from Raspberry Pi to selected Bluetooth device. For sending file, right click on selected device and select send a file as shown below,



Browse a file and send it. We can receive a file too on Raspberry Pi.

4. Command Line Interface (CLI) for Bluetooth Services

The packages which we have installed earlier are also necessary for command line interface.

Pairing a Bluetooth device from shell is the simplest and most reliable option.

- **Bluetooth Connection using bluetoothctl**

Follow the steps given below to pair and connect Raspberry Pi's Bluetooth with another Bluetooth device. We are using bluetoothctl utility for pairing and establishing connection between Raspberry Pi and Bluetooth devices.

Enter following command to start bluetoothctl

```
bluetoothctl
```

turn the power to the controller on using following command

```
power on
```

enter **devices** to get the mac address of device with which we want to pair. This command gives a list of available devices.

```
devices
```

if no device is in the list then search for devices as follows,

```
scan on
```

turn the agent on

```
agent on
```

pair the Bluetooth device

```
pair <mac address>
```

e.g. pair 01:02:03:04:05:06

If using a device without PIN then we need to manually mark the device as a trust device as follows,

```
trust <mac address>
```

Finally establish a connection between two Bluetooth enabled devices.

```
Connect <mac address>
```

To know more about commands, you can enter **help** command**.**

- **Send/Receive Data over Bluetooth using CLI**

We can send/receive data from/on Raspberry Pi over Bluetooth using CLI. This communication is helpful when we need to control any device connected to Raspberry Pi using smartphone/laptop over Bluetooth. For such applications CLI is the most efficient way of communication. Here, we are using RFCOMM (Radio Frequency Communication) Bluetooth protocol.

Conclusion:

Post Lab Questions:

1. What is BLE and how does BLE technology works?
2. Explain role of BLE in IoT.
3. List out applications of BLE, Explain any one in detail

Additional links for more information:

1. https://circuitdigest.com/microcontroller-projects/turn-your-raspberry-pi-into-bluetooth-beacon-using-eddystone-ble-beacon
2. https://www.instructables.com/id/Control-Bluetooth-LE-Devices-From-A-Raspberry-Pi/
3. https://www.electronicwings.com/raspberry-pi/using-raspberry-pi-3-on-board-bluetooth-for-communication

Third Year B. Tech (ECE)

Semester: VI **Subject:** IoT Architectures and Protocols

Name: **Class:**

Roll No: **Batch:**

Experiment No: 05

Name of the Experiment: To Study and implement MQTT/AMQP/HTTP protocol using hardware platform

Performed on:

Submitted on:

Marks	Teacher's Signature with date

Aim: Install MQTT protocol on Raspberry Pi to establish communication between connected objects.

Prerequisite: Concept of MQTT, Raspberry Pi model 3.

Objectives:

1. To understand how sensor data will get published and subscribed
2. Installing Mosquitto, the open-source broker on Raspberry Pi
3. Send / receive your first MQTT messages

Components and equipment required:

Raspberry Pi Board with Micro SD Card, HDMI to VGA Converter, Power Supply.

Theory:

1. Introduction of MQTT protocol

MQTT (Message Queuing Telemetry Transport) is an ISO standard publish-subscribe-based messaging protocol. It works on top of the TCP/IP protocol. It is designed for connections with remote locations where a "small code footprint" is required or the network bandwidth is limited. The publish-subscribe messaging pattern requires a message broker. An MQTT system consists of clients communicating with a server, often called a "broker". A client may be either a publisher of information or a subscriber. Each client can connect to the broker.

Information is organized in a hierarchy of topics. When a publisher has a new item of data to distribute, it sends a control message with the data to the connected broker. The broker then distributes the information to any clients that have subscribed to that topic. The publisher does not need to have any data on the number or locations of subscribers, and subscribers in turn do not have to be configured with any data about the publishers. If a broker receives a topic for which there are no current subscribers, it will discard the topic unless the publisher indicates that the topic is to be retained. This allows new subscribers to a topic to receive the most current value rather than waiting for the next update from a publisher.

When a publishing client first connects to the broker, it can set up a default message to be sent to subscribers if the broker detects that the publishing client has unexpectedly disconnected from the broker. Clients only interact with a broker, but a system may contain several broker servers that exchange data based on their current subscribers' topics. MQTT relies on the TCP protocol for data transmission. A variant, MQTT-SN, is used over other transports such as UDP or Bluetooth.

2. Message Types:

Connect

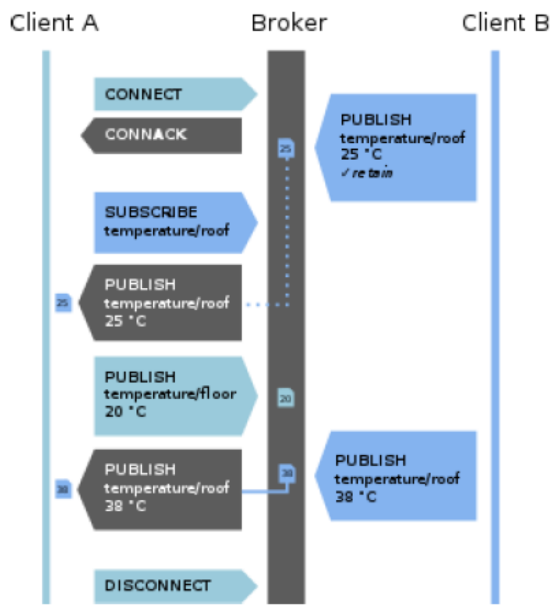
Waits for a connection to be established with the server and creates a link between the nodes.

Disconnect

Waits for the MQTT client to finish any work it must do, and for the TCP/IP session to disconnect.

Publish

Returns immediately to the application thread after passing the request to the MQTT client.


Client A
Broker
Client B
CONNECT
CONNACK
PUBLISH temperature/roof 25 °C ✓retain
SUBSCRIBE temperature/roof
PUBLISH temperature/roof 25 °C
PUBLISH temperature/floor 20 °C
PUBLISH temperature/roof 38 °C
PUBLISH temperature/roof 38 °C
DISCONNECT

Publish/Subscribe

The MQTT protocol is based on the principle of publishing messages and subscribing to topics, or "pub/sub". Multiple clients connect to a broker and subscribe to topics that they are interested in. Clients also connect to the broker and publish messages to topics. Many clients may subscribe to the same topics and do with the information as they please. The broker and MQTT act as a simple, common interface for everything to connect to. This means that you if you have clients that dump subscribed messages to a database, to Twitter, Cosm or even a simple text file, then it becomes very simple to add new sensors or other data input to a database, Twitter or so on.

Topics/Subscriptions

Messages in MQTT are published on topics. There is no need to configure a topic, publishing on it is enough. Topics are treated as a hierarchy, using a slash (/) as a separator. This allows sensible arrangement of common themes to be created, much in the same way as a filesystem.

3. Raspberry Pi Implementation

I) MQTT Broker Setup and Subscription

- The important step is to make sure your MQTT broker is set up, and ready to receive and send messages. MQTT broker can be used in following ways

Existing broker on either a local network or on the Internet (Cloud): You will use an existing broker on your network or on the Internet (Cloud). This broker is ready to send and receive messages and you have the ability to add a topic to this broker if a suitable topic is not already available.

New broker on a local network using Paho Python:

You will establish an MQTT broker on your Development Computer using Paho Python libraries. The Development Computer connects to the Gateway through a local network.

New broker on a local network using mosquitto: You will establish an MQTT broker on your Development Computer using mosquitto. The Development Computer connects to the Gateway through a local network

II) MQTT implementation using Mosquitto

Mosquitto is an open source message broker (or server) that implements MQTT protocols. With its good community support, documentation, and ease of installation it has become one of the most popular MQTT brokers. Mosquitto is an open source iot.eclipse.org project. It implements the MQTT protocol versions 3.1 and 3.1.1. For more details, please refer to http://mosquitto.org/.

a. Prerequisites

- An Ubuntu 16.04 server with root access
- Open port TCP:1883 on firewall (Check if port is available or not)

b. Mosquitto Broker Installation

Step One: Install Mosquitto

Update Ubuntu's package list and install the latest Mosquitto Broker available from it

```
$sudo apt-get update
$sudo apt-get install mosquito
```

The Mosquitto service will start after installation.

Step Two:

Install MQTT clients

```
$sudo apt-get install mosquitto-clients
```

Mosquitto clients help us easily test MQTT through a command line utility. We will use two command windows, one to subscribe to a topic named "test" and one to publish a message to it.

Topics are labels used by the broker to filter messages for each connected client. A client program subscribed to a topic "Home1/BedroomTemp" will only listen to messages published to the same topic by other clients. Subscribe to topic "test"
mosquitto_sub -t "test"
Mosquito_sub is a subscribe client we installed in the previous command. Here specify "-t" followed by a topic name.
Publish a message to topic "test"
Login to the terminal as a second instance and publish a message to the "test" topic.
$mosquitto_pub -m "message from mosquitto_pub client" -t "test"
Here the additional parameter "–m" is followed by the message we want to publish. Hit "Enter" and you should see a message from mosquitto_pub client displayed in other terminal where mosquito_sub client is running.
Step Three:
Test Mosquitto MQTT Broker with MQTT Client:
For testing you can use any MQTT Client. However, if you have Python 2.7 Installed on your machine, you can test it with following sample Python scripts. To execute these Scripts, you must have Paho MQTT Client installed on your machine. You can install it with pip command –
$pip install paho-mqtt
Once Paho Client Library is installed, you can execute following Python scripts (Don't forget to change "MQTT_BROKER" IP Address) –

- Publisher.py (Code is available at the end for reference)
- Subscriber.py (Code is available at the end for reference)

c. Uninstall Mosquitto MQTT Broker:
To uninstall Mosquitto you can use following command –
$sudo apt-get purge mosquitto
If you want to completely remove Mosquitto with its associated configuration files, use following command –
$sudo apt-get --purge remove mosquito
d. Test MQTT implementation using Mosquitto
MQTT using an Existing MQTT Broker on the Internet/cloud based on Paho Python

Procedure:

Perform these steps on your Development Computer.
1. Connect the Development Computer to the Internet.
2. Open a new Linux terminal window.
3. Go to the Paho examples directory:
cd org.eclipse.paho.mqtt.python-1.1/examples
4. Modify the Subscriber.py script to change the mqttc.connect and mqttc.subscribe lines:
mqttc.connect("test.mosquito.org",1883, 60)

mqttc.subscribe("mytopic", 0)

In this script, the following apply:

- mqttc.connect
- test.mosquitto.org: The Internet broker URL.
- 1883: Network port used for MQTT messages.
- 60: Timeout in seconds.
- mqttc.subscribe
- mytopic: Name of your new MQTT topic. The topic is established automatically on the broker if it does not already exist.
- 0: Quality of Service.

5. Run the sub.py script:

python subscriber.py

Subscribing to a new topic called mytopic on the test.mosquitto.org broker automatically establishes that topic. Now, when the Gateway (or any other computer) publishes an MQTT message to mytopic on the test.mosquitto.org broker, the message will be sent to the Linux terminal where the sub.py script is running. Continue to Gateway Setup.

- Publisher.py (Code is available at the end for reference)
- Subscriber.py (Code is available at the end for reference)
- **Set up an MQTT Broker on a Local Network Using Mosquitto**

Use the steps in this section if your goal is to use your Development Computer as a local network test broker using the open source application called mosquitto.

Note: For details about mosquitto use and syntax, see http://mosquitto.org, or use <mosquitto_command> --help, where <mosquitto_command> is the command that you need help with.

To start a broker process on the Development Computer:

1. Open a new Linux terminal window.

The broker will run in this window. You can still receive MQTT messages on this Development Computer. The MQTT messages will be received in a different terminal window.

2. Install mosquitto if you do not already have it installed.

sudo apt-get install mosquito

3. Create a topic and subscribe to it with the mosquitto_sub command.

mosquitto_sub -d -h localhost -t mytopic

In this script, the following apply:

- -d: Enable debug messages.
- -h localhost: An alias to the development computer's local loopback IP address (typically 127.0.0.1).
- -t mytopic: Name of the MQTT topic. The topic is established automatically on the broker if it does not already exist.

4. Check that the Development Computer is connected to the same network as the Gateway. Subscribing to a new topic called mytopic on the local host automatically starts the local broker

and establishes that topic for other subscribers. Now, when the Gateway publishes an MQTT message to mytopic at the Development Computer's IP address, the message will be sent to the Linux terminal where the mosquitto_sub command is running.

Conclusion:

Post Lab Questions:

1. What is MQTT? Write history of MQTT
2. Different ways of using Mosquitto broker.
3. What are the principles of MQTT?
4. Differentiate Transfer protocols

Additional links for more information:

1. https://diyprojects.io/mqtt-mosquitto-communicating-connected-objects-iot/#.W2AMH9UzbDe
2. http://test.mosquitto.org/gauge/
3. https://cs.iupui.edu/~xiaozhon/course_tutorials/

Third Year B. Tech (ECE)

Semester: VI **Subject:** IoT Architectures and Protocols

Name: **Class:**

Roll No: **Batch:**

Experiment No: 06

Name of the Experiment: To design a simple IoT based system using cloud infrastructure for connecting IoT devices

Performed on:

Submitted on:

Marks	Teacher's Signature with date

Aim: To sense the data from sensors and send it to cloud system in simple text files, excel sheets or databases system

Prerequisite: Cloud concepts, Raspberry Pi 3 Model

Objectives:

1. To understand how sensor data is sent to the cloud
2. To learn various public cloud platforms

Components and equipment required:

Raspberry Pi 3 Model, DHT22, 10k Resistor, Jumper cables etc

Theory:

Temperature sensor: It is a device, a thermocouple or RTD, that provides temperature measurement through an electrical signal.

Thermocouple: It is made from two dissimilar metals that generate electrical voltage in direct proportion to changes in temperature. The wires are joined together to form measuring junction and reference junction.

RTD: Resistor temperature detection is variable resistor that will change its electrical resistance in direct proportion to changes in temperature in precise, repeatable & linear manner.

Temperature and Humidity Sensor DHT22 & Raspberry Pi Interfacing

The DHT22 sensor is used to measure the temperature and humidity. It is also known as AM2302. This sensor is cheap and also has better accuracy.

Specifications of DHT22

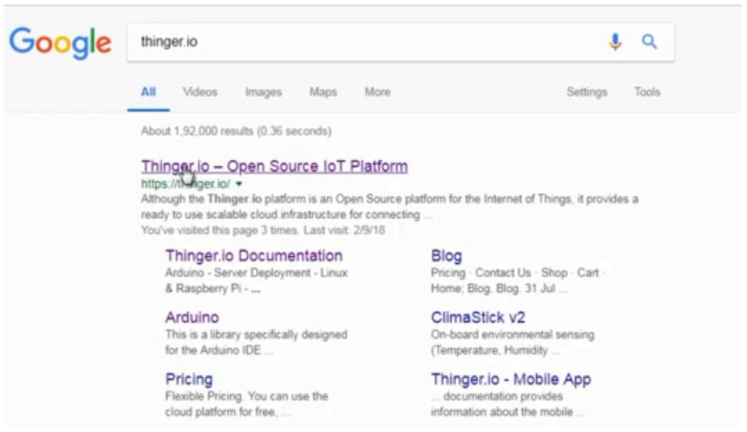
The specifications of the temperature and humidity sensor DHT22 are as follows:

- Temperature range is from -40 to 125 degree Centigrade with accuracy of ±0.5 °C.
- Humidity range is from 0 to 100% with accuracy of ± 2-5%.
- Sampling rate is 0.5 Hz.
- Operating Voltage is 3-5V.
- Maximum Current while measuring is 2.5mA.
  - Connect DHT sensor to raspberry pi
  - Read the sensor data
  - Upload the data to Thingspeak.
  - Analyze the data

Setting up the Thinger.io Account:

Thinger.io is a cloud IoT Platform that provides every needed tool to prototype, scale and manage connected products in a very simple way.

- **Free IoT platform:** Thinger.io provides a lifetime freemium account with only few limitations to start learning and prototyping when your product becomes ready to scale, you can deploy a Premium Server with full capacities within minutes.
- **Simple but Powerful:** Just a couple code lines to connect a device and start retrieving data or controlling its functionalities with our web-based Console, able to connect and manage thousands of devices in a simple way.
- **Hardware agnostic:** Any device from any manufacturer can be easily integrated with Thinger.io's infrastructure.
- **Extremely scalable & efficient infrastructure:** thanks to our unique communication paradigm, in which the IoT server subscribes device resources to retrieve data only when it is necessary, a single Thinger.io instance is able to manage thousands of IoT devices with low computational load, bandwidth and latencies.
- **Open-Source:** most of the platform modules, libraries and APP source code are available in our Github repository to be downloaded and modified with MIT license.



thinger.io

About 1,92,000 results (0.36 seconds)

Thinger.io – Open Source IoT Platform
https://thinger.io/
Although the Thinger.io platform is an Open Source platform for the Internet of Things, it provides a ready to use scalable cloud infrastructure for connecting ...
You've visited this page 3 times. Last visit: 2/9/18

Thinger.io Documentation
Arduino - Server Deployment - Linux & Raspberry Pi - ...

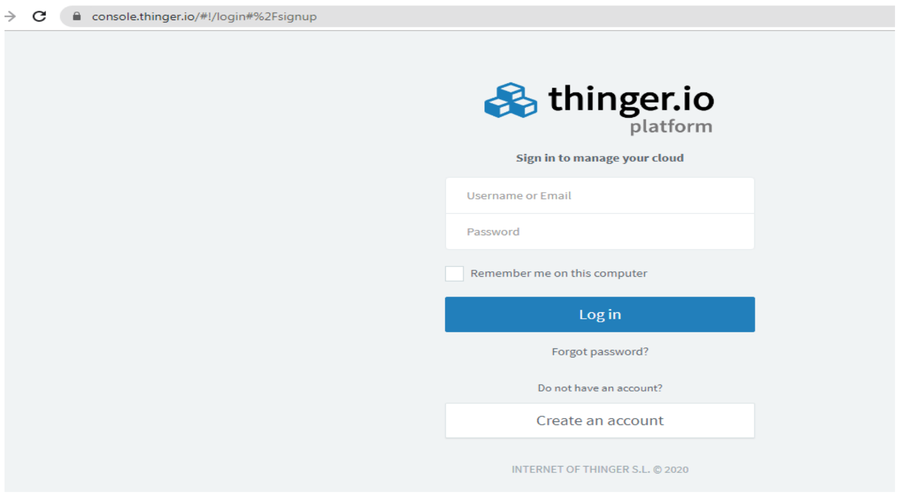
Blog
Pricing · Contact Us · Shop · Cart · Home; Blog. Blog. 31 Jul ...

Arduino
This is a library specifically designed for the Arduino IDE ...

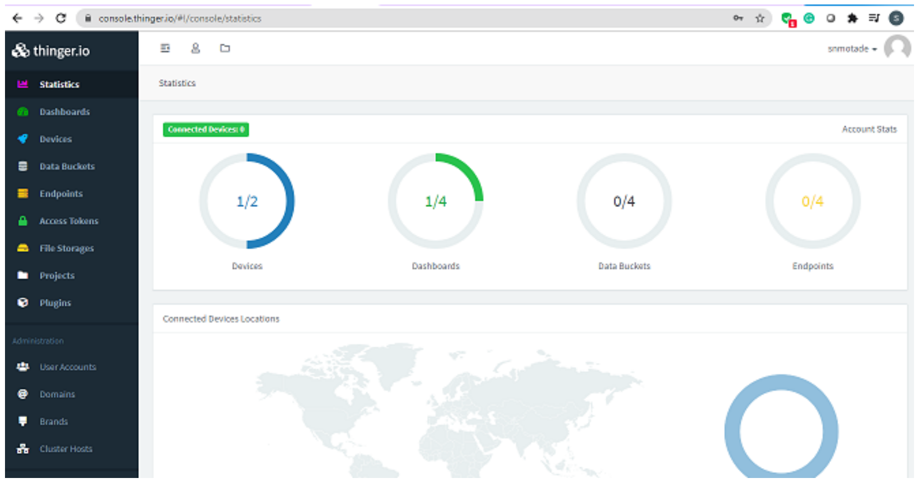
ClimaStick v2
On-board environmental sensing (Temperature, Humidity ...

Pricing
Flexible Pricing. You can use the cloud platform for free, ...

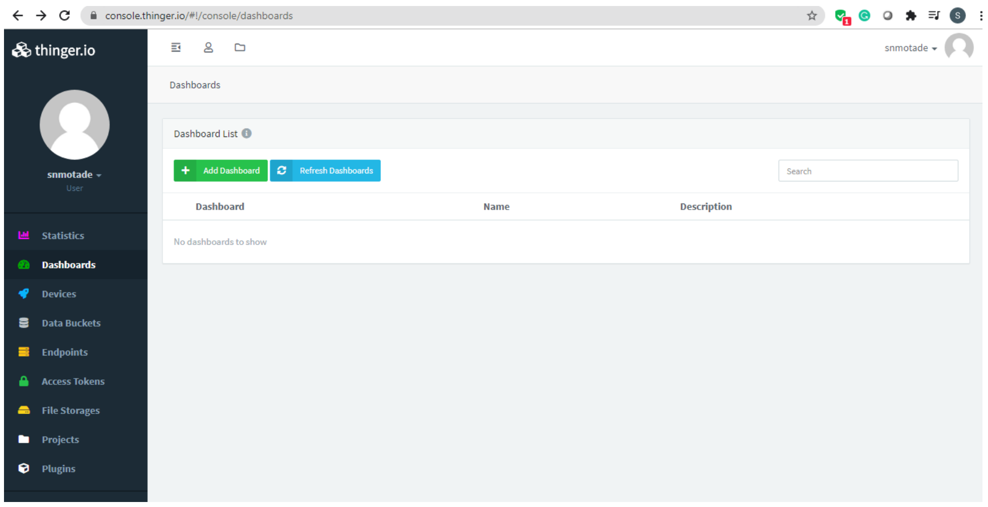
Thinger.io - Mobile App
... documentation provides information about the mobile ...


console.thinger.io/#!/login#%2Fsignup
thinger.io platform
Sign in to manage your cloud
Username or Email
Password
Remember me on this computer
Log in
Forgot password?
Do not have an account?
Create an account
INTERNET OF THINGER S.L. © 2020

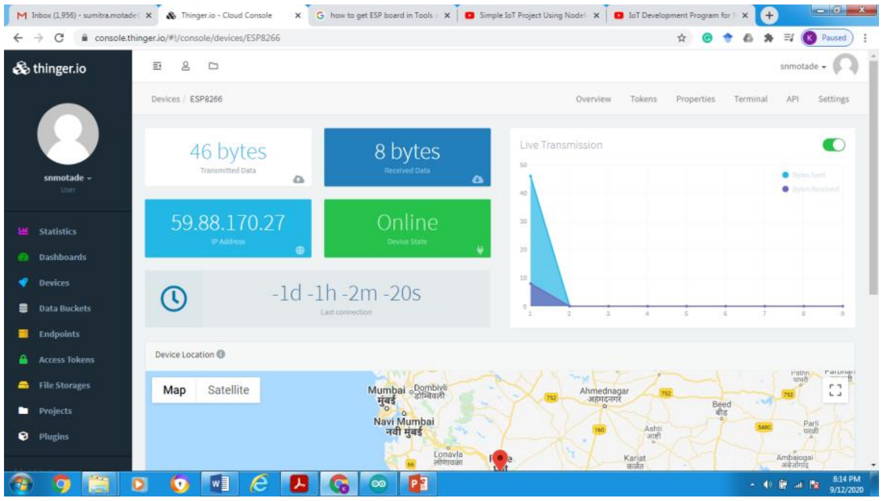
Create the device:



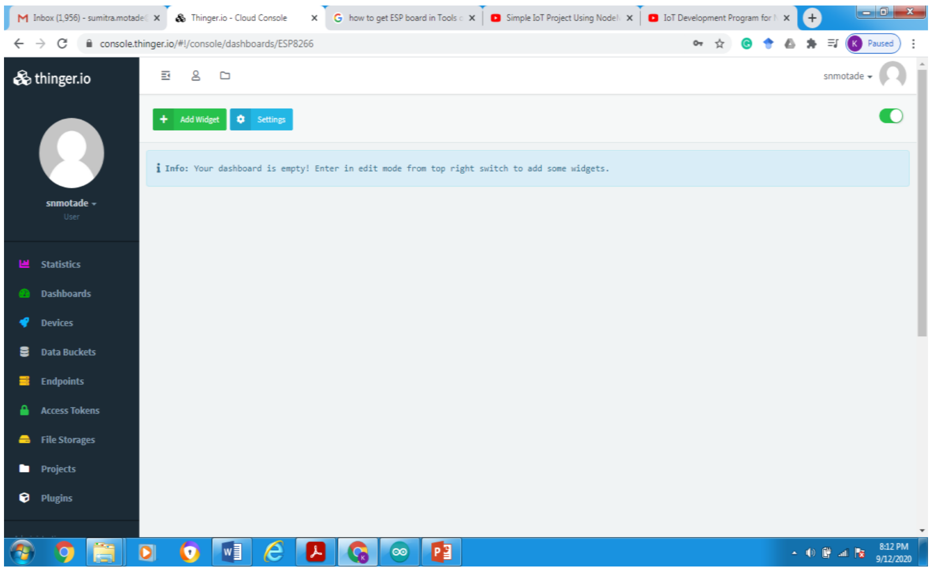
Add Dashboard:



Device Connection:



Add Widgets:



```
Info: Your dashboard is empty! Enter in edit mode from top right switch to add some widgets.
```

Installing the DHT22 Library:

1. Enter the below command to clone the library
git clone https://github.com/adafruit/Adafruit_Python_DHT.git

2. Then enter in to the installed directory using the below command
cd Adafruit_Python_DHT

3. Now download the required modules using the below command
sudo apt-get install build-essential python-dev

4. Then install the library using the below command
sudo python setup.py install

Conclusion:

Post Lab Questions:

1. List and write features of IoT Cloud platforms.
2. What is IBM Watson?
3. What is Amazon Web Service?
4. What is the role of cloud in IoT?

Additional links for more information:

1. https://www.youtube.com/watch?v=fEJDKjDbPR0
2. https://www.youtube.com/watch?v=qq2VnxGIFv0https://in.mathworks.com/solutions/internet-of-things.html
3. https://docs.thinger.io/features/endpoints-1
4. https://static-pt-assets.s3.amazonaws.com/tutorials71.htm#stub

Third Year B. Tech (ECE)

Semester: VI **Subject:** IoT Architectures and Protocols

Name: **Class:**

Roll No: **Batch:**

Experiment No: 07

Name of the Experiment: Data visualization and Analysis for IoT application.

Performed on:

Submitted on:

Marks	Teacher's Signature with date

Aim: To sense the data from sensors and send it to cloud system for analysis.

Prerequisite: Cloud concepts, Basics of Raspberry Pi 3 Model/ NodeMCU, Basics of DHT11 sensor

Objectives:

1. Interfacing Input and output devices to Raspberry pi/ NodeMCU
2. Read data from the sensor and display on Serial Monitor
3. To understand how sensor data is sent to the cloud (ThingSpeak)
4. Display real time sensor data on cloud platform in various form

Components and equipment required:

Raspberry pi, DHT11 sensor, Relay, HDMI cable, Python, Breadboard, LED, Resistors jumper wires etc.

Theory:

ThingSpeak is an IoT platform. It uses channels to store data sent from apps or devices. With the settings described in Channel Configurations, it is possible to create a channel, and then send and retrieve data to and from the channel. The channels can be made public to share data. What makes ThingSpeak different and special is that is uses simple HTTP Protocol to transfer, store and retrieve information from different sensors.

ThingSpeak is an open data platform for monitoring your data online. You can set the data as private or public depending on your choice. ThingSpeak takes minimum of 15 seconds to update your readings. It's a great platform for building your IoT projects. We will read the temperature and humidity from the DHT22 and then we will send it to the API of the ThingSpeak channel. We will get the API after creating the channel.

Also, the ThingSpeak Application allows us to log the sensor data, track locations and even social networking of things. Another important thing (or rather a unique feature) about ThingSpeak is its support from MATLAB. The close relationship between ThingSpeak and MATLAB has led to integrate several key features of MATLAB into the ThingSpeak Application. One such feature is to analyse and visualize the user data i.e. the sensor data in a graphical way without the MATLAB License.

Using the Representational state transfer (REST) which is an architectural style designed as a request-response model that communicates over HTTP. ThingSpeak, uses the REST API calls GET, POST, PUT, and DELETE to create and delete channels, read and write channel data, and clear the data in a channel. A web browser or client sends a request to the server, which responds with data in the requested format. Web browsers use this interface to retrieve web pages or to send data to remote servers.

You can also use the MQTT Publish method to update a channel feed and MQTT Subscribe to receive messages when a channel updates.

It is possible to upload data from the web or send data from devices to a Thing Speak IoT platform channel. Use these apps to transform and visualize data on Thing Speak channels or trigger an action or can use your web browser to complete GET HTTP requests to the RESTful API for ThingSpeak.

Copy the URL to the address bar of your web browser, changing <write_api_key> to your user API Key, which is found in Account > My Profile.

https://api.thingspeak.com/update.json?api_key=<write_api_key>&field1=123

The response is a JSON object of the public channel list.

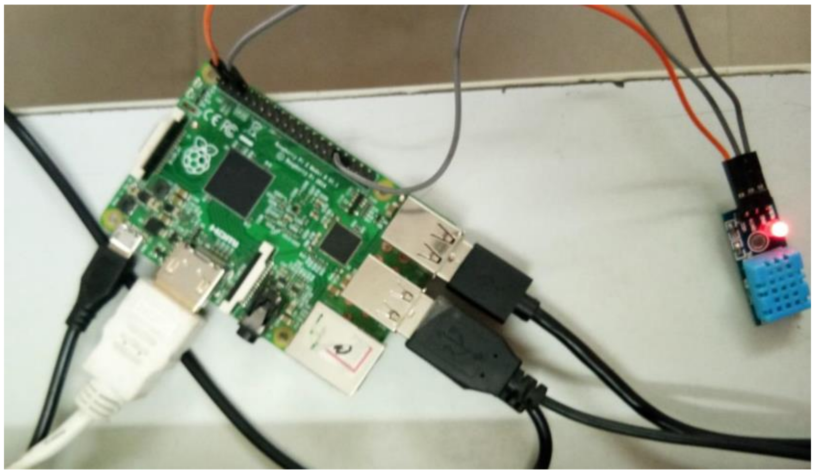
Procedure:

Part A

- Create an account on ThingSpeak https://thingspeak.com/
- Create a new channel over ThingSpeak
- Edit the channel settings, add two fields first one for temperature and second for humidity
- Save the channel.
- Review the "Update a Channel Feed" URL thingSpeak is insanely easy to use.
- On your channel and using key make an HTTP request to https://api.thingspeak.com/update?api_key=YOUR_KEY_HERE&field1=4
- Send different value into field1 and field2.

Part B:

- Connect DHT sensor to Raspberry Pi



- The DHT11 sensor consists of three connecting pins:
  1. VCC: This pin used to connect with the voltage. Connect this pin with the pin number2 at Raspberry Pi board via connecting wire.
  2. GND: This is called as 'ground'. Connect it at Pin number 6 of pi board.
  3. DATA: It is used as an output data port connect this at pin named GPIO 23 at Pi board.
- Save the code as 'dht_thingspeak.py'.
- Use Adafruit DHT Library to make this code run.
- Now run the following command at the terminal to execute the code:

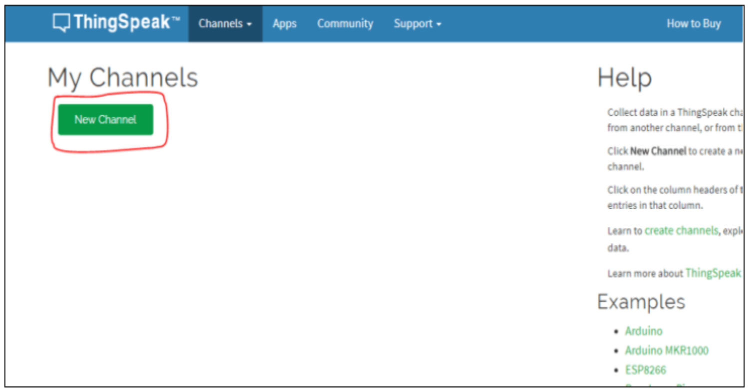
  sudopython dht_thingspeak.py 'writeAPI Key'.

- Here, mention the write API key of your thingspeak channel at theplace 'writeAPIKey'.
- Upload the data to Thingspeak.
- Analyze the data using MATLAB

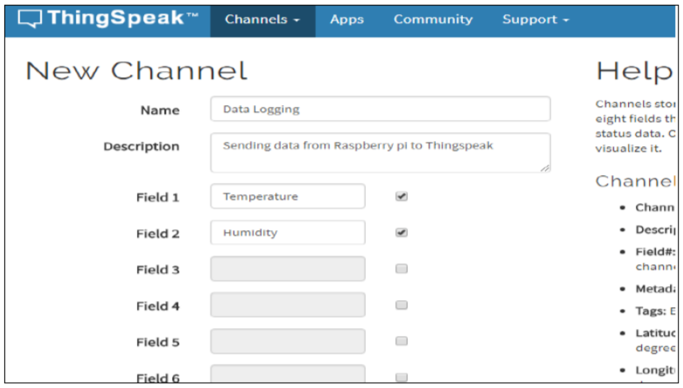
Setting up the ThingSpeak Account:

1. First of all, go to the following link and sign up to ThingSpeak. If you already have an account, then sign in. https://thingspeak.com/

2. After creating the account or logging in, click on new channel.

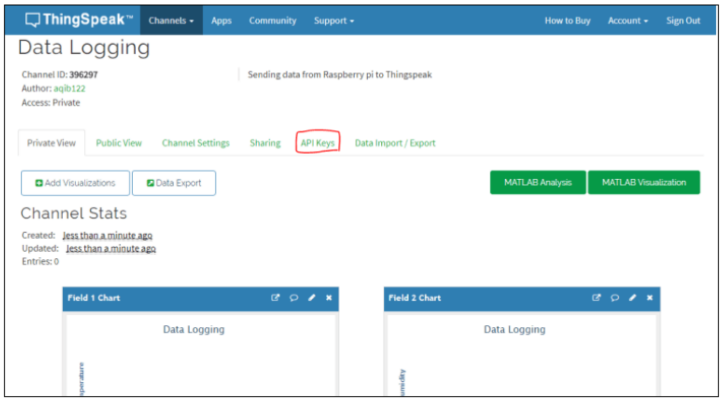


3. Fill the information about the channel. Select two fields because we will be sending the data for the two fields from the raspberry Pi. Leave the other information as it is and save the channel.



Name	Data Logging	
Description	Sending data from Raspberry pi to Thingspeak	
Field 1	Temperature	✓
Field 2	Humidity	✓
Field 3		☐
Field 4		☐
Field 5		☐
Field 6		☐

4. Go to the API keys tab.



Data Logging

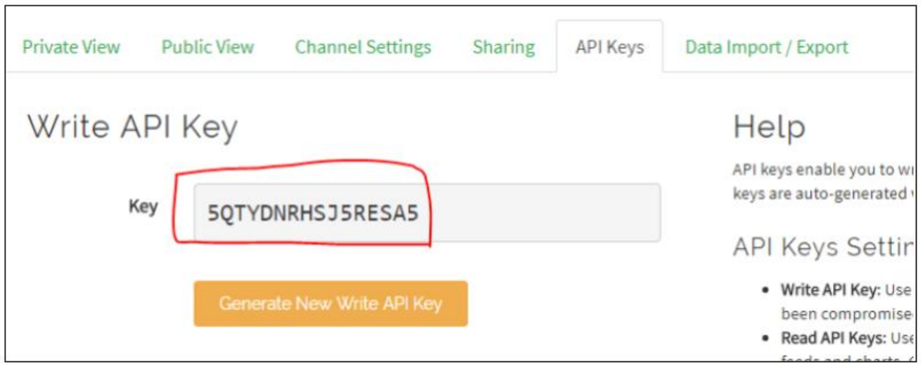
Channel ID: 396297
Author: aqib122
Access: Private

Sending data from Raspberry pi to Thingspeak

Channel Stats

Created: less than a minute ago
Updated: less than a minute ago
Entries: 0

5. In the API keys tab, copy the write API key. This is the API key at which we will send the data from the Raspberry Pi.



Write API Key

Key 5QTYDNRHSJ5RESA5

Help

API keys enable you to wi
keys are auto-generated

API Keys Settir

- **Write API Key:** Use
been compromise
- **Read API Keys:** Use

Conclusion:

____

____

____

____

Post Lab Questions:

1. What are the key features of ThingSpeak cloud platform?
2. List out various five temperature sensors and write features of all
3. Explain the IoT system Application which uses Cloud platform and also role of Cloud platform

Additional links for more information:

1. https://thingspeak.com
2. https://in.mathworks.com/solutions/internet-of-things.html
3. https://in.mathworks.com/matlabcentral/fileexchange/52456-analyzing-traffic-using-a-webcam-a-raspberry-pi-and-thingspeak