PRESIDENCY UNIVERSITY

GAIN MORE KNOWLEDGE REACH GREATER HEIGHTS

PRESIDENCY GROUP OVER 40 YEARS OF ACADEMIC WISDOM

Private University Estd. in Karnataka State by Act No. 41 of 2013

Department of Computer Science and Engineering

www.presidencyuniversity.in

Laboratory Manual

GAIN MORE KNOWLEDGE
REACH GREATER HEIGHTS

Internet of Things (IoT)

CSE 220

For

III year Students (CSE)

Itgalpur Rajanakunte, Yelahanka, Bengaluru, Karnataka 560064

FOREWORD

It is my great pleasure to present this laboratory manual for third year engineering students for the subject Internet of Things (IoT).

As a student, many of you may be wondering with some of the questions in your mind regarding the subject and exactly what has been tried is to answer through this manual.

As you may be aware that presidency university has already been awarded with best emerging university in south India by ASSOCHAM, INDIA and it's our endure to technically equip our students take the advantage of the procedural aspects of this certification.

Faculty members are also advised that covering these aspects in initial stage itself, will greatly relieve them in future as much of the load will be taken care by the enthusiasm energies of the students once they are conceptually clear.

Mohammed Mujeer
Assistant Professor,
Department of CSE,
Presidency University

LABORATORY MANUAL CONTENTS

This manual is intended for the third year students of computer science and engineering in the subject of Internet of Things (IoT). This manual typically contains practical/lab sessions related raspberry pi and android implemented in python programming covering various aspects related the subject to enhanced understanding.

Students are advised to thoroughly go through this manual rather than only topics mentioned in the syllabus as practical aspects are the key to understanding and conceptual visualization of theoretical aspects covered in the books.

Good luck for your enjoyable laboratory sessions

Prof. Mohammed Mujeer

Instructor In charge

PROGRAME OBJECTIVES

B.Tech. Computer Science Engineering graduates of Presidency University, will be able to:

1. Direct and coordinate activities concerned with software/hardware design, development, and testing.
2. Design and develop products and systems for various applications.
3. Monitor and analyze computing system performance for network traffic, security, and capacity.
4. Order and maintain inventory of computing equipment for customer premises equipment, facilities, access networks and backbone network
5. Operate and trouble shoot computer-assisted engineering, design software and equipment to perform various engineering tasks.

PROGRAMME OUTCOMES

On successful completion of B.Tech. Computer Science Engineering programme from Presidency University a student will be able to

1. **Engineering knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write

effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

DOs and DON'Ts in Laboratory:

1. Make entry in the Log Book as soon as you enter the Laboratory.
2. All the students should sit according to their roll numbers starting from their left to right.
3. All the students are supposed to enter the terminal number in the log book.
4. Do not change the terminal on which you are working.
5. All the students are expected to get at least the algorithm of the program/concept to be implement.
6. Strictly follow the instructions given by the teacher/Lab Instructor.

LAB INDEX

Design, develop and implement following using arduino, raspberry pi compiler and python language.

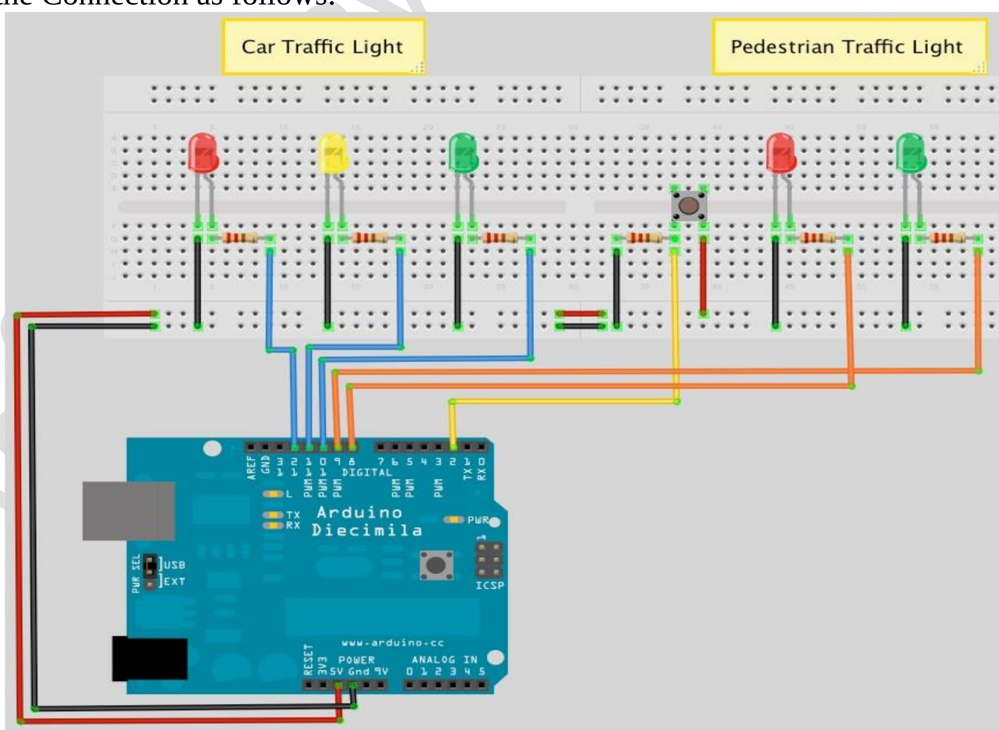
	Arduino Experiments
1	Arduino program to demonstrate traffic control system and to simulate using tinker CAD.
2	Arduino program to demonstrate usage of servo motor with potentio meter and to simulate using tinker CAD.
3	Arduino program to demonstrate Bluetooth controlled LED
4	Arduino program to demonstrate IR sensor module
5	Arduino program to demonstrate PIR Motion Sensor
6	Arduino program to demonstrates Security access using RFID Reader
	Raspberry pi Experiments
7	Installation of Raspberry pi software
8	Working basic commands on Raspberry pi & to demonstrate remote logging in raspberry pi
9	Raspberry pi program to implement blinking LED
10	Raspberry pi program to implement camera module for video
11	Raspberry pi program to obtain the temperature using DHT sensors
12	Using a Raspberry Pi with distance sensor (ultrasonic sensor HC-SR04)
13	Raspberry pi program to implement Garage spot light

Experiment No: 1

Arduino program to implement traffic control system

In this experiment we are going to make a traffic light with pedestrian light along with button, to request to cross the road. The Arduino will execute when the button is pressed by changing the state of light to make the cars stop and allow the pedestrian to cross safely.

Sl no.	Equipment's Needed
1	Arduino board(1)
2	Power cable(1)
3	Breadboard(1)
4	LED(6) 2x Red LED's 1x Yellow LED 2x Green LED's
5	220 Ω Resistor(6)
6	Jumper Wires(7)
7	Tactile Switch

Step 1:	Connect the Arduino board to your computer using the USB cable.
Step 2:	Select the board and serial port as outlined in earlier section.
Step 3:	Make the Connection as follows: 
Step 4:	Open project code to type following program

	`` // Interactive Traffic Lights int carRed = 12; // To assign the car lights int carYellow = 11; int carGreen = 10; int pedRed = 9; // To assign the pedestrian lights int pedGreen = 8; int button = 2; // Tactile button pin int crossTime = 5000; // time allowed to cross unsigned long changeTime; // time since button pressed void setup() { pinMode(carRed, OUTPUT); pinMode(carYellow, OUTPUT); pinMode(carGreen, OUTPUT); pinMode(pedRed, OUTPUT); pinMode(pedGreen, OUTPUT); pinMode(button, INPUT); // button on pin 2 // turn on the green light digitalWrite(carGreen, HIGH); digitalWrite(pedRed, HIGH); } void loop() { int state = digitalRead(button); /* check if button is pressed and it is over 5 seconds since last button press */ if (state == HIGH && (millis() - changeTime) > 5000) { ```

```
// Call the function to change the lights
changeLights();
}
}
void changeLights() {
digitalWrite(carGreen, LOW); // green off
digitalWrite(carYellow, HIGH); // yellow on
delay(2000); // wait 2 seconds
digitalWrite(carYellow, LOW); // yellow off
digitalWrite(carRed, HIGH); // red on
delay(1000); // wait 1 second till its safe
digitalWrite(pedRed, LOW); // ped red off
digitalWrite(pedGreen, HIGH); // ped green on
delay(crossTime); // wait for preset time period
// flash the ped green
for (int x=0; x<10; x++) {
digitalWrite(pedGreen, HIGH);
delay(250);
digitalWrite(pedGreen, LOW);
delay(250);
}
// turn ped red on
digitalWrite(pedRed, HIGH);
delay(500);
digitalWrite(carYellow, HIGH); // Yellow will switch on
digitalWrite(carRed, LOW); // red will switch off
delay(1000);
digitalWrite(carGreen, HIGH);
digitalWrite(carYellow, LOW); // Yellow will switch off

// record the time since last change of lights
changeTime = millis();
```

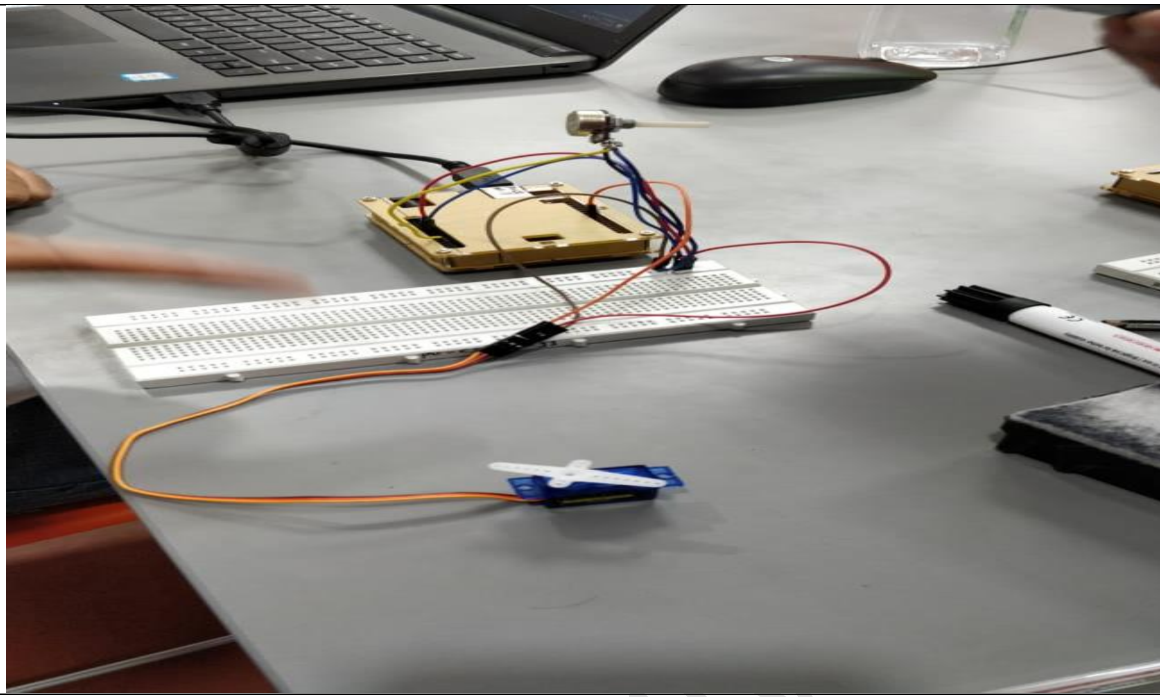
	// Retun / Loop }

Experiment No: 2

Arduino program to demonstrate control of servo motor using potentio meter

Sl no.	Equipment's Needed
1	Arduino board
2	Power cable
3	Servo Motor
4	Potentio meter

Step 1:	Connect the Arduino board to your computer using the USB cable.
Step 2:	Select the board and serial port as outlined in earlier section.
Step 3:	Lots of IoT applications with respevt to robotics use servo motor. Servo motor comes with 3 different arms with 2 variants full arm and half arm and runs 1800. Black pin->Gnd pin of arduino Red Pin-> +5V of arduino Orange ->Used to send pulse width modulation, and is connected to pin no 09 from arduino Potentio meter- is a three-terminal resistor with a sliding or rotating contact that forms an adjustable voltage divider. If only two terminals are used, one end and the wiper, it acts as a variable resistor or rheostat. 1st Nob->Gnd pin of arduino 2nd Nob->A0 pin of arduino 3rd Nob->+5v pin of arduino
Step 4:	Make the Connection as follows:



Step 5: Open project code to type following program

```
/*
 Controlling a servo position using a potentiometer (variable resistor)
 by Michal Rinott <http://people.interaction-ivrea.it/m.rinott>

 modified on 8 Nov 2013
 by Scott Fitzgerald
 http://www.arduino.cc/en/Tutorial/Knob
*/

#include <Servo.h>

Servo myservo;  // create servo object to control a servo

int potpin = 0;  // analog pin used to connect the potentiometer
int val;    // variable to read the value from the analog pin

void setup()
{
  myservo.attach(9);  // attaches the servo on pin 9 to the servo object
}

void loop()
{
val = analogRead(potpin); // reads the value of the potentiometer (value between 0 and
1023)
val = map(val, 0, 1023, 0, 180);// scale it to use it with the servo (value between 0 and 180)
myservo.write(val);                // sets the servo position according to the scaled value
delay(15);
}                      // waits for the servo to get there
```

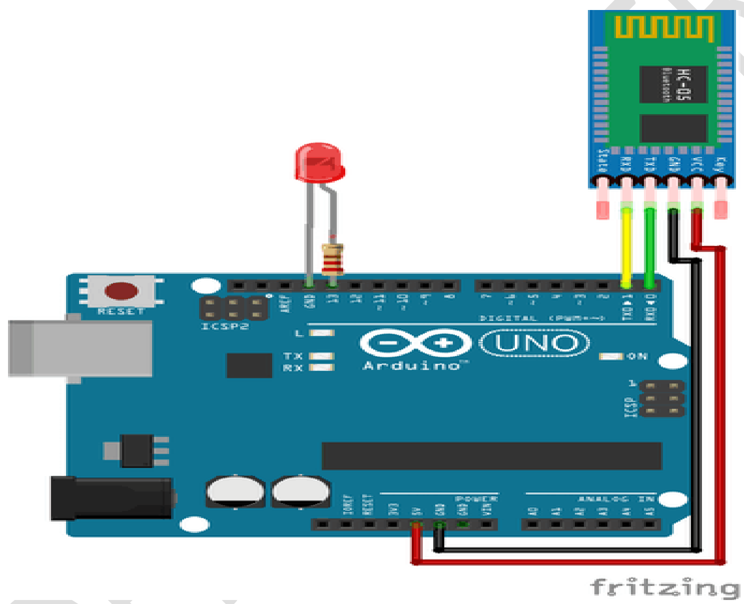
Experiment No: 3

Bluetooth controlled LED

Sl no.	Equipment's Needed
1	Arduino board
2	Power cable
3	Servo Motor
4	Potentio meter

This experiment is going to show how to make Bluetooth LED control

Step#1: Connection.



Step#2:

Let's Start Building

The circuit is so simple and small, there are only a few connections to be made

Arduino Pins Bluetooth Pins

RX (Pin 0) ———> RX

TX (Pin 1) ———> TX

5V ———> VCC

GND ———> GND

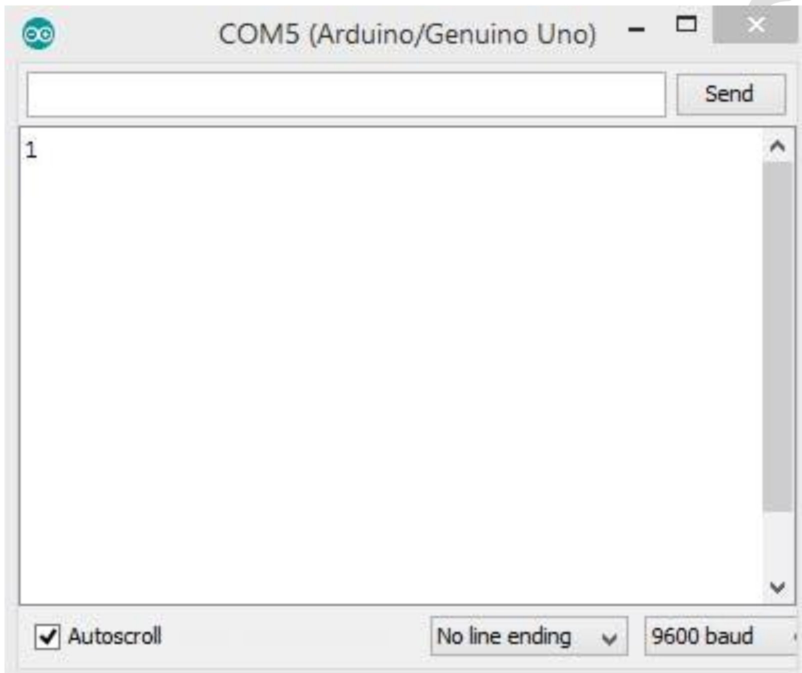
Connect a LED negative to GND of Arduino and positive to pin 13 with a resistance valued between 220Ω – 1KΩ. And you're done with the circuit

Step#3:

How Does it Work?

HC 05/06 works on serial communication.here the android app is designed to send serial data to the Bluetooth module when a certain button is pressed. The Bluetooth module at the other end receives the data and sends it to Arduino through the TX pin of the Bluetooth module(RX pin of Arduino). The Code fed to Arduino checks the received data and compares it.If received data is 1 the LED turns on turns OFF when received data is 0

Open the serial monitor and watch the received data



How to use the App?

Watch in video how to pair to Bluetooth module

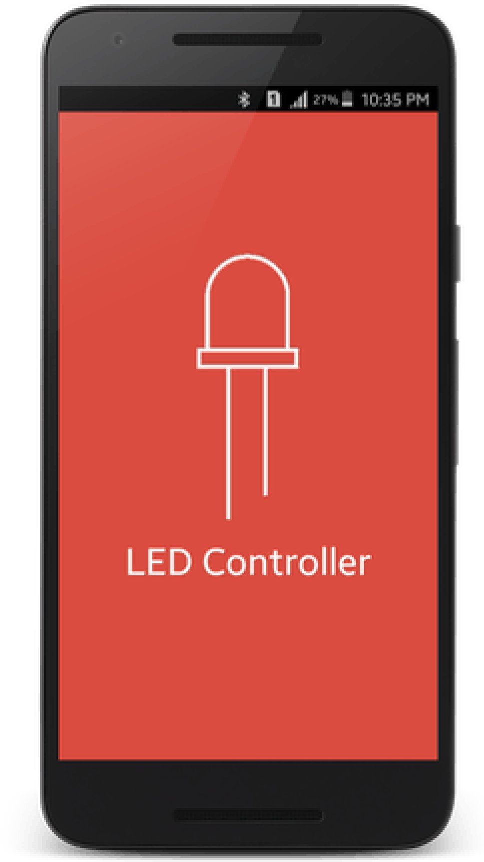
- Download the Application form here or here
- Pair your device with HC 05/06 Bluetooth module

1) Turn ON HC 05/06 Bluetooth module

2) Scan for available device

3) Pair to HC 05/06 by entering default password 1234 OR 0000

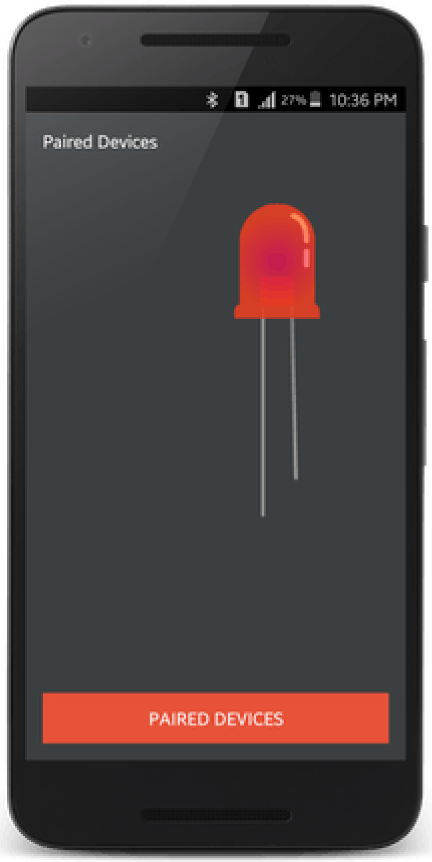
- Install LED application on your android device
- Open the Application

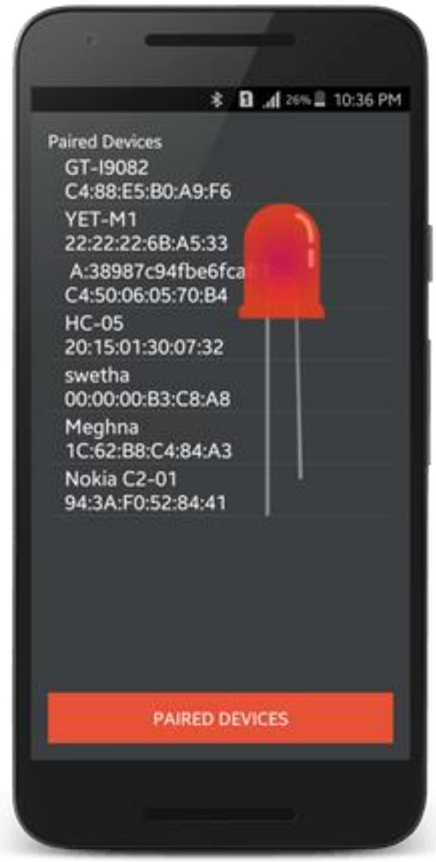

10:35 PM
LED Controller

•

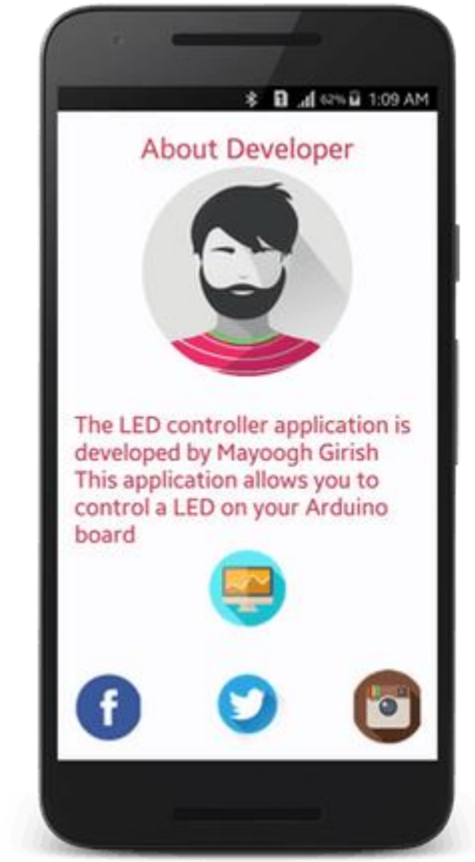
Press paired devices

Select your Bluetooth module from the List (HC 05)





- After connecting successfully
- Press ON button to turn ON LED and OFF button to turn OFF the LED



- Disconnect button to disconnect from Bluetooth module

Experiment No: 4

IR SENSOR MODULE

What is an infrared sensor ?

An infrared sensor is an electronic device, that emits in order to sense some aspects of the surroundings. An IR sensor can measure the heat of an object as well as detects the motion.These types of sensors measures only infrared radiation, rather than emitting it that is called as a passive IR sensor.

Can an IR sensor detect humans?

The Passive Infrared (PIR) sensor is used to detect the presence of human.... The Grid-EYE sensor detects the human using the infrared radiation radiated by the human body. Every human radiates the infrared energy of specific wavelength range. The absorbed incident radiation changes the temperature of a material.IR sensors? 2.8V at 15cm to 0.4V at 150cm with a supply voltage between 4.5 and 5.5 VDC.

Components Required

1. Arduino Uno

2. IR sensor

3. LED

4.Arduio cable

5.Jumper wire

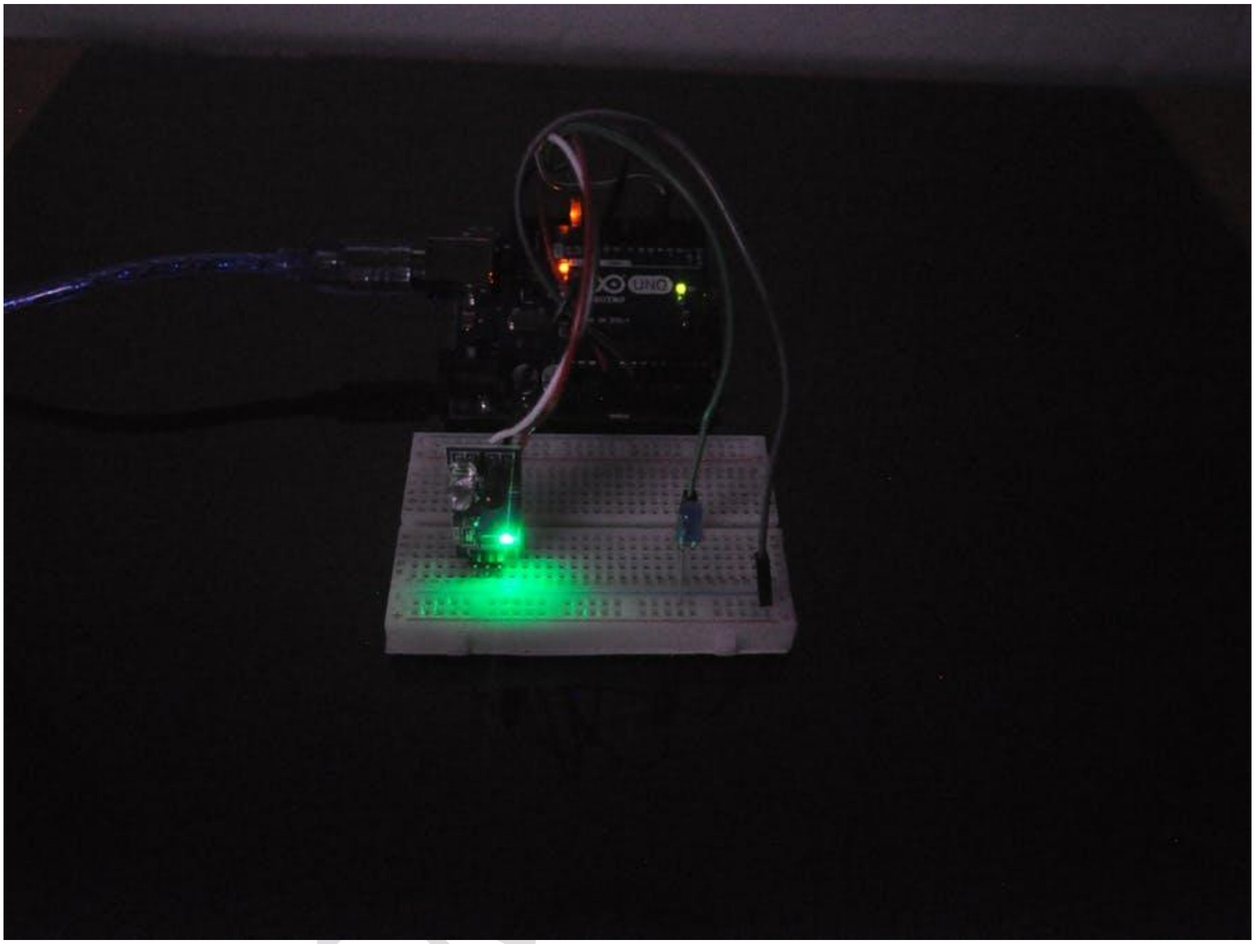
IR Sensor Module

The connections for the **IR sensor with** the **Arduino** are as follows:

Connect the negative wire on the **IR sensor** to GND on the **Arduino**. Connect the middle of the **IR sensor** which is the VCC to 5V on the **Arduino**. Connect the signal pin on the **IR sensor** to pin 8 on the **Arduino**.

```
void setup() {
  // put your setup code here, to run once:
pinMode(8,INPUT);
pinMode(12,OUTPUT);//LED
}

void loop() {
  // put your main code here, to run repeatedly:
if(digitalRead(8)==LOW){
  digitalWrite(12,HIGH);
}
```

```
else{
  digitalWrite(12,LOW);
}
}
```



Experiment No: 5

Arduino with PIR Motion Sensor

PIR sensor detects a human being moving around within approximately 10m from the sensor. This is an average value, as the actual detection range is between 5m and 12m.PIR are fundamentally made of a pyro electric sensor, which can detect levels of infrared radiation. For numerous essential projects or items that need to discover when an individual has left or entered the area. PIR sensors are incredible, they are flat control and minimal effort, have a wide lens range, and are simple to interface with.

Most PIR sensors have a 3-pin connection at the side or bottom. One pin will be ground, another will be signal and the last pin will be power. Power is usually up to 5V. Sometimes bigger modules don't have direct output and instead just operate a relay which case there is ground, power and the two switch associations. Interfacing PIR with microcontroller is very easy and simple. The PIR acts as a digital output so all you need to do is listening for the pin to flip high or low. The motion can be detected by checking for a high signal on a single I/O pin. Once the sensor warms up the output will remain low until there is motion, at which time the output will swing high for a couple of seconds, then return low. If motion continues the output will cycle in this manner until the sensors line of sight of still again. The PIR sensor needs a warm-up time with a specific end goal to capacity fittingly. This is because of the settling time included in studying nature's domain. This could be anyplace from 10-60 seconds.


```
int led = 13;                // the pin that the LED is atteched to
int sensor = 2;              // the pin that the sensor is atteched to
int state = LOW;             // by default, no motion detected
int val = 0;                 // variable to store the sensor status (value)

void setup() {
  pinMode(led, OUTPUT);      // initalize LED as an output
  pinMode(sensor, INPUT);    // initialize sensor as an input
  Serial.begin(9600);        // initialize serial
}

void loop(){
  val = digitalRead(sensor);   // read sensor value
  if (val == HIGH) {           // check if the sensor is HIGH
    digitalWrite(led, HIGH);   // turn LED ON
    delay(500);                // delay 100 milliseconds

    if (state == LOW) {
      Serial.println("Motion detected!");
      state = HIGH;        // update variable state to HIGH
    }
  }
  else {
      digitalWrite(led, LOW); // turn LED OFF
      delay(500);             // delay 200 milliseconds

      if (state == HIGH){
        Serial.println("Motion stopped!");
        state = LOW;        // update variable state to LOW
    }
  }
}
```

Experiment No: 6

Security access using RFID Reader

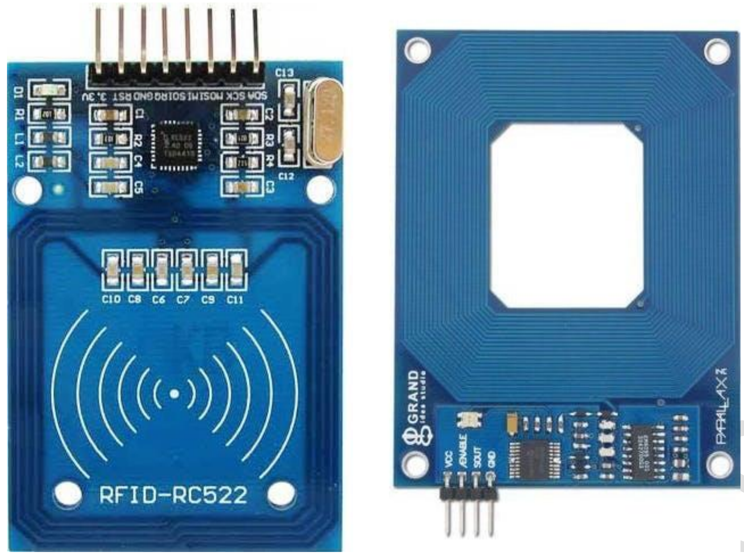
What is an RFID reader?

RFID tagging is an ID system that uses small radio frequency identification devices for identification and tracking purposes. An RFID tagging system includes the tag itself, a read/write device, and a host system application for data collection, processing, and transmission. In simple words an RFID uses electromagnetic fields to transfer data over short distances. RFID is useful to identify people, to make transactions, etc…You can use an RFID system to open a door. For example, only the person with the right information on his card is allowed to enter.

An RFID system uses:

- Tags attached to the object to be identified, in this example we have a keychain and an electromagnetic card. Each tag has his own identification (UID).

- Two-way radio transmitter-receiver, the reader, that sends a signal to the tag and read its response



Basic Specifications:

Input voltage: 3.3V

Frequency: 13.56MHz

Now, before typing out the necessary code, you need to download the necessary library for this sensor from this repository.

Extract the contents from the zip folder "rfid-master" and add this library folder under the existing libraries of Arduino.

After doing so, restart your ArduinoIDE.

Now, our Arduino is ready to take commands and execute accordingly.

The Arduino Code has been uploaded at the end of this tutorial. Compile the code and eliminate "typo" errors (if any).

Now, its time to connect our Arduino with the RFID reader. Refer to the PIN wiring below,as well as the Connection schematic diagram for easy reference.

PinWiring to Arduino Uno

SDA------------------------Digital 10

SCK------------------------Digital 13

MOSI----------------------Digital 11

MISO----------------------Digital 12

IRQ------------------------unconnected

GND-----------------------GND

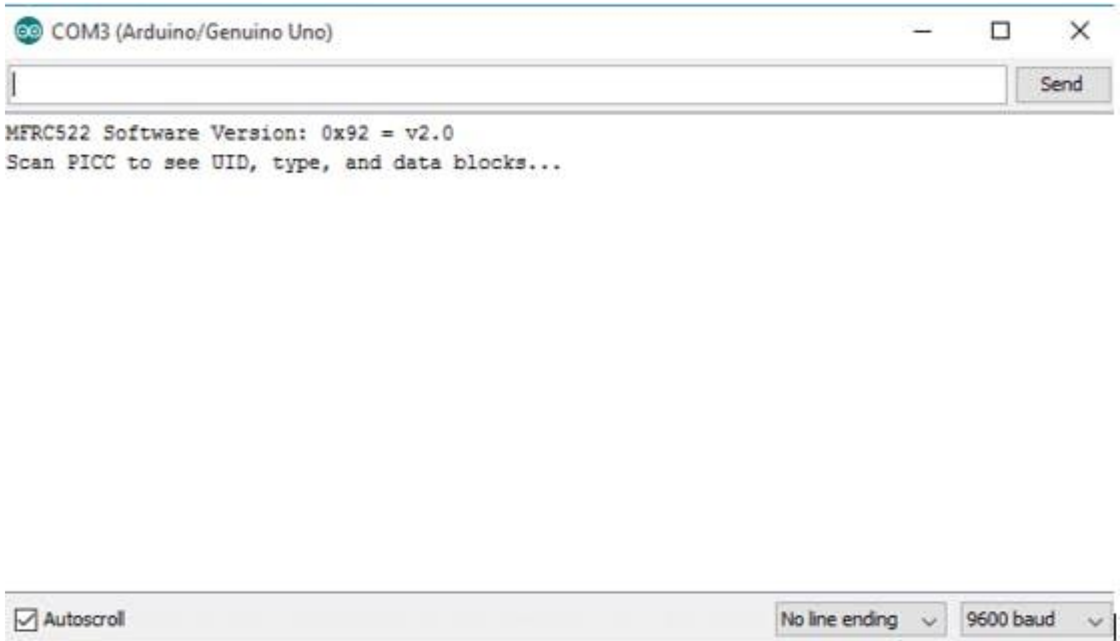
RST------------------------Digital 9

3.3V------------------------3.3V (DO NOT CONNECT TO 5V)

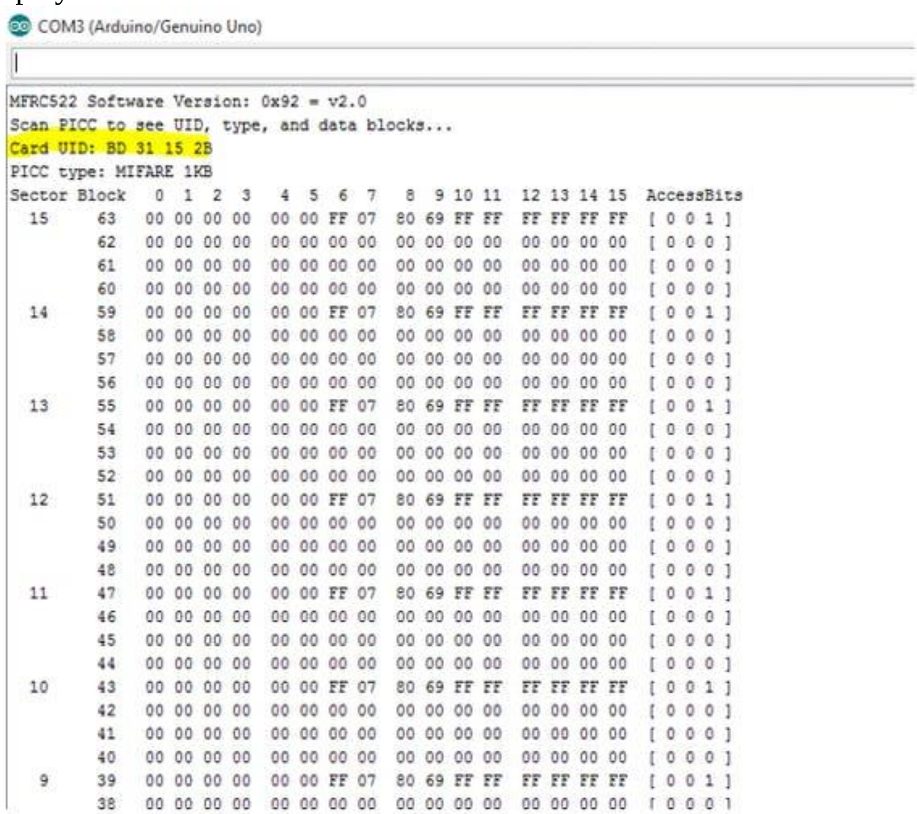
Reading data from an RFID tag

After having the circuit ready, go to File > Examples > MFRC522 > DumpInfo and upload the code. This code will be available in Arduino IDE (after installing the RFID library).

Then, open the serial monitor. You should see something like the figure below:



Approximate the RFID card or the keychain to the reader. Let the reader and the tag closer until all the information is displayed.



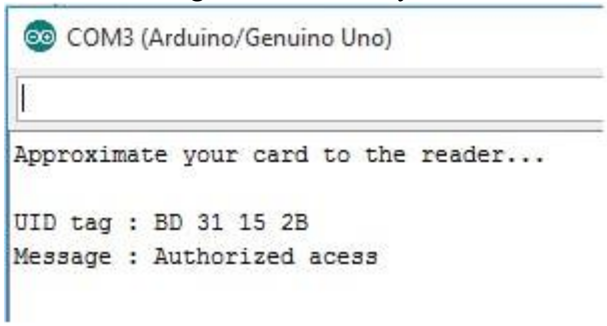
This is the information that you can read from the card, including the card UID that is highlighted in yellow. The information is stored in the memory that is divided into segments and blocks as you can see in the previous picture.

You have 1024 bytes of data storage divided into 16 sectors and each sector is protected by two different keys, A and B.

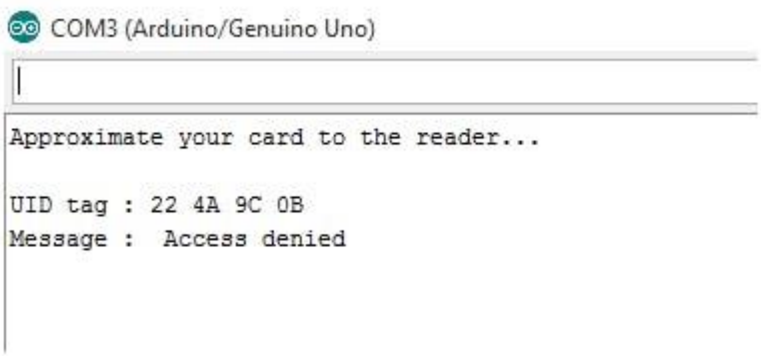
Write down your UID card because you'll need it later. Upload the Arduino code that has been suffixed here.

Demonstration

Approximate the card you've chosen to give access and you'll see:



If you approximate another tag with another UID, the denial message will show up




```
/*
 *
 * All the resources for this project: https://www.hackster.io/Aritro
 * Modified by Aritro Mukherjee
 *
 *
 */

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

#define SS_PIN 10
#define RST_PIN 9
MFRC522 mfrc522(SS_PIN, RST_PIN);   // Create MFRC522 instance.

void setup()
{
  Serial.begin(9600);   // Initiate a serial communication
  SPI.begin();      // Initiate  SPI bus
  mfrc522.PCD_Init();   // Initiate MFRC522
```

```
  Serial.println("Approximate your card to the reader...");
  Serial.println();

}
void loop()
{
  // Look for new cards
  if ( ! mfrc522.PICC_IsNewCardPresent())
  {
    return;
  }
  // Select one of the cards
  if ( ! mfrc522.PICC_ReadCardSerial())
  {
    return;
  }
  //Show UID on serial monitor
  Serial.print("UID tag :");
  String content= "";
  byte letter;
  for (byte i = 0; i < mfrc522.uid.size; i++)
  {
     Serial.print(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");
     Serial.print(mfrc522.uid.uidByte[i], HEX);
     content.concat(String(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " "));
     content.concat(String(mfrc522.uid.uidByte[i], HEX));
  }
  Serial.println();
  Serial.print("Message : ");
  content.toUpperCase();
  if (content.substring(1) == "BD 31 15 2B") //change here the UID of the
card/cards that you want to give access
  {
    Serial.println("Authorized access");
    Serial.println();
    delay(3000);
  }

 else   {
    Serial.println(" Access denied");
    delay(3000);
  }
}
```

Experiment No: 6

Installation of Raspberry pi software

Sl no.	Equipment's Needed
1	Raspberry Pi
2	VGA cable
3	Power Cable

Introduction: The Raspberry Pi is a fully-fledged mini computer, capable of doing whatever you might do with a computer. It comes with 4x USB, HDMI, LAN, built-in Bluetooth/WiFi support, 1GB RAM, 1.2GHz quad-core ARM CPU, 40 GPIO (General Purpose Input Output) pins, audio and composite video output, and more.

One can use Raspberry Pis as home security cameras, server monitoring devices, cheap headless machines (basically running low-weight scripts 24/7 with a low cost-to-me)... others have used them for media centers and even for voice-enabled IoT devices. The possibilities are endless, but first we need to get acquainted!

Make sure that, if you do get a case, it has openings for the GPIO pins to be connected, you will also need a 1000mA+ mini usb power supply and at least an 8GB micro SD card, but suggested is 16 GB micro SD card or greater.

You will also want to have a spare monitor (HDMI), keyboard, and mouse handy to make things easier when first setting up. You wont will eventually be able to control your Pi remotely, so you wont always need a separate keyboard, mouse, and monitor. If you don't have a monitor with HDMI input, you can buy something like an HDMI to DVI converter.

If you're using an older version board, please see what you might need to change, for example, the older Rasbperry Pis take a full-sized SD card, but the latest model requires a micro SD card. Also, the Raspberry Pi 3 Model B has built-in wifi, where the older models will require a wifi dongle.

A typical Raspberry Pi shopping list, assuming you have a mouse, keyboard, and HDMI monitor that you can use temporarily while setting up is:

1. Raspberry Pi -
2. 1000mA+ mini usb power supply -
3. 16 GB micro SD card –

Additionally, if you plan to join us on the initial GPIO (General Purpose Input Output pins) tutorials, you will also want to pick up:

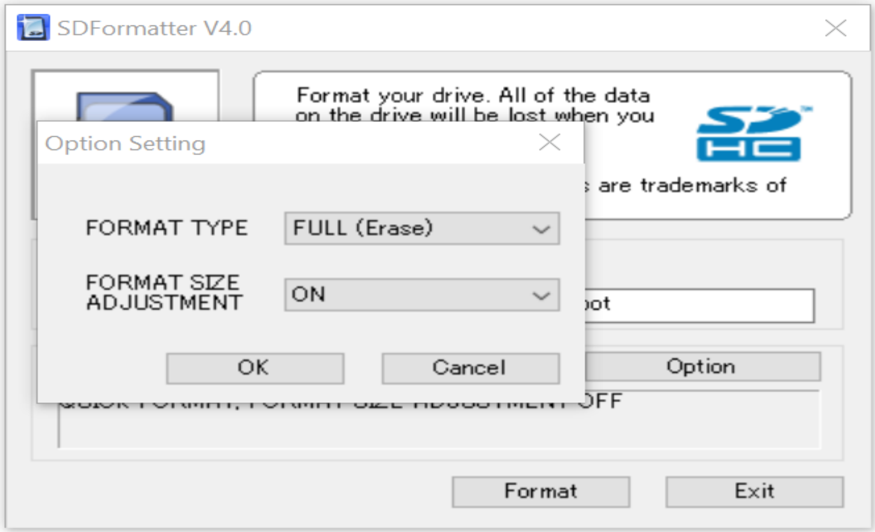
1. 10 x Male-to-Female jumper wires (you should consider just buying a bunch of these so you have plenty in the future).
2. 1 x Breadboard (You may also want multiples of these)
3. 3 x LED light (...more wouldn't hurt)
4. ~6 x Resistors (between 300 and 1K Ohm). You will need at least 1K and 2K ohms for the distance sensor, then ~300-1K resistance per LED bulb. You probably should just buy a kit, they're super cheap.
5. 1 x HC-SR04 Ultrasonic Distance Sensor (...you know what I'm going to say...think about maybe a few.)
6. 1 x Raspberry Pi camera module. You only need one of these!

For the jumpers, breadboard, and leds, you could also just buy a kit, something like: this GPIO starter kit.

There are also a few ways to install and use an operating system on the Raspberry Pi. The most user-friendly method is to use the NOOBS (New Out of Box Software) installer. If you're comfortable enough, you can just simply download the operating system ISO, format the SD card, mount the ISO, and boot the Pi. If that sounds like gibberish to you, then follow along with the NOOBS installation option.

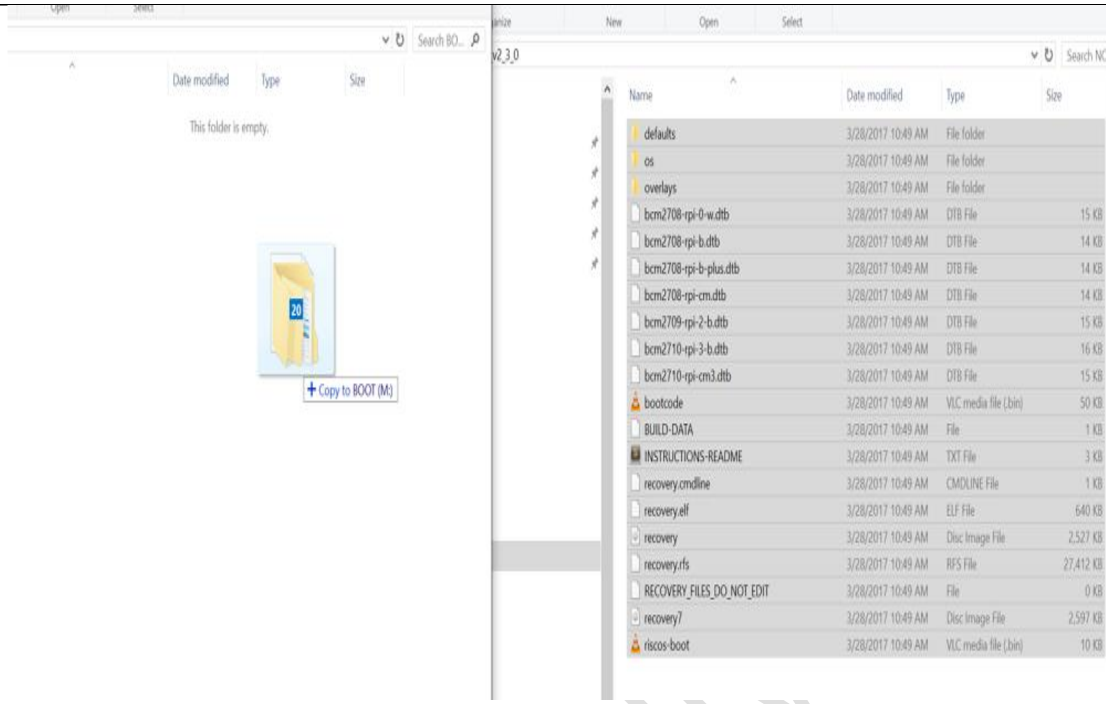
While we're working with the SD card, let's go ahead and Download NOOBS, which is just over 1GB.

First, we must format the SD card. If you are on Windows, you can use SD Formatter. Mac users can also use SD Formatter, but they have a built in formatter, and Linux users can use GParted. Whatever the case, you need to format the SD card, do not do a "quick format" and do make sure you have the "resize" option on. Using SDFormatter on Windows, and chosing options:

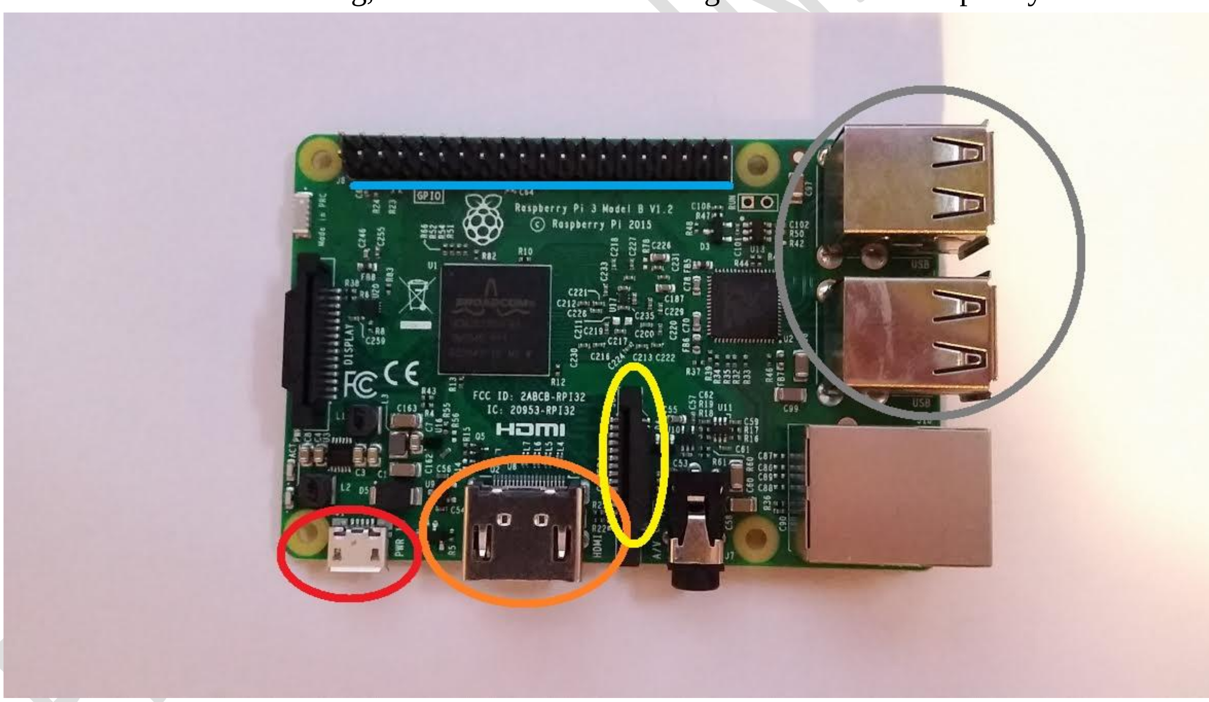


This should go without saying, but do make sure you're formatting the right drive. This will format any flash drive, in alphabetical order. If you had something plugged in already, like your favorite USB drive, and forgot about it, that'd likely be the default choice to format, and then you'd spend all afternoon trying to recover your data rather than enjoying playing with your Raspberry Pi.

Now, assuming you've downloaded the NOOBS package, let's go ahead and extract that. Now, we want to copy all these NOOBS contents to our SD Card. Do not drag the directory, but rather the contents:



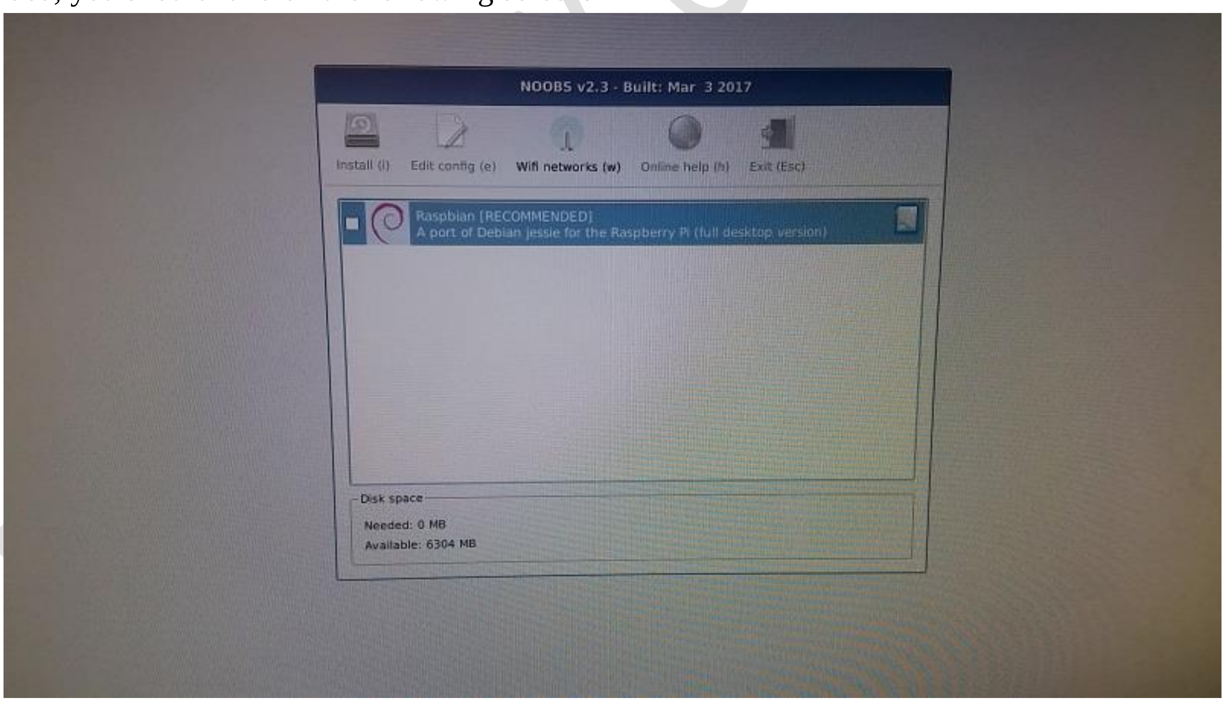
While that's transferring, let's talk about a few things on the actual Raspberry Pi board:



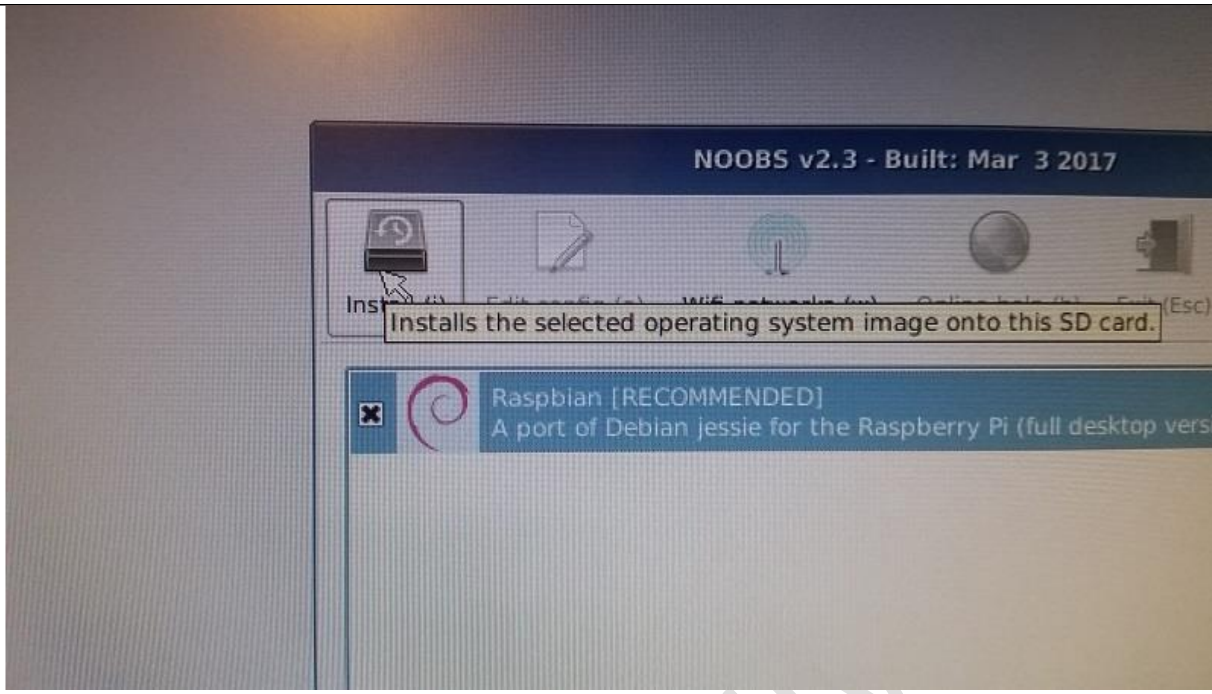
The GPIO (General Purpose Input/Output) pins are underlined in blue. We can use these to control peripheral devices like motors, servos, and more. Circled in red is the micro usb power input for the board. In orange, the HDMI output port. The yellow is where you can plug in the Raspberry Pi camera module. The grey circle has the USB ports. This is obviously not everything, but these are the main things to note.

Once everything is transferred to the micro SD card, you can put it in the Raspberry Pi. The slot is on the bottom side of the board, circled in yellow here:

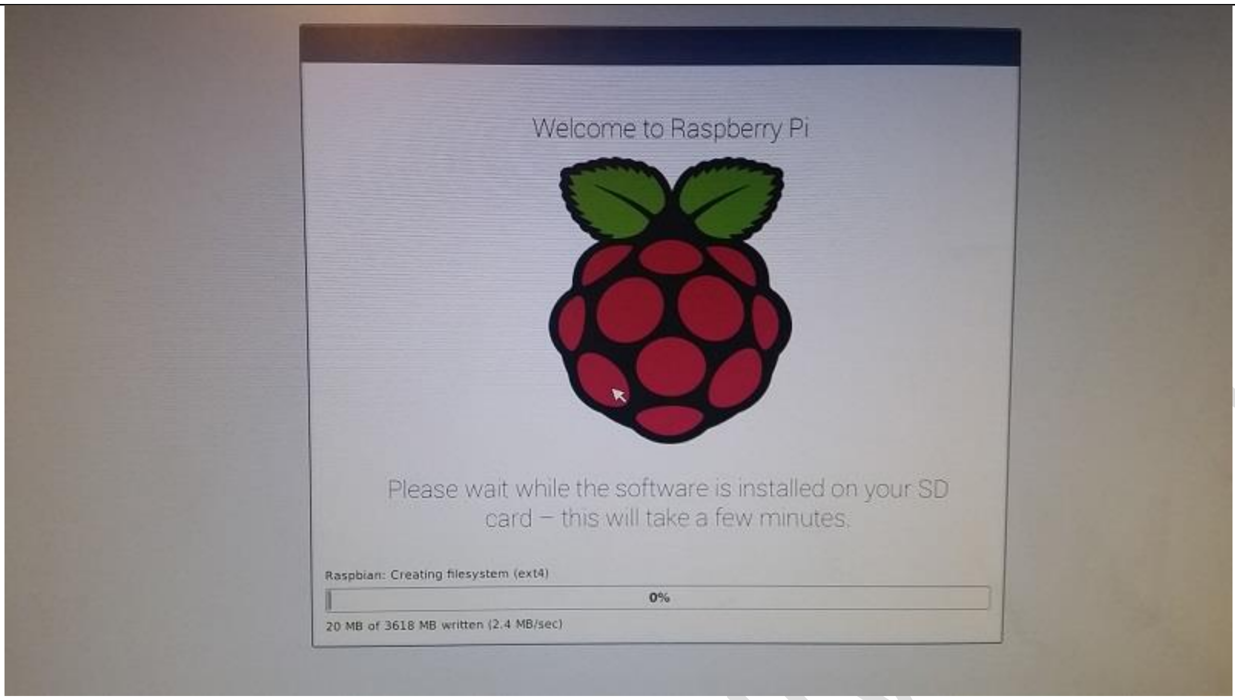
Once you've got the SD card plugged in, go ahead and plug in your keyboard, mouse, and HDMI cable to your monitor. Finally, plug in the power, and this will start up the Raspberry Pi. Once fully loaded, you should land on the following screen:



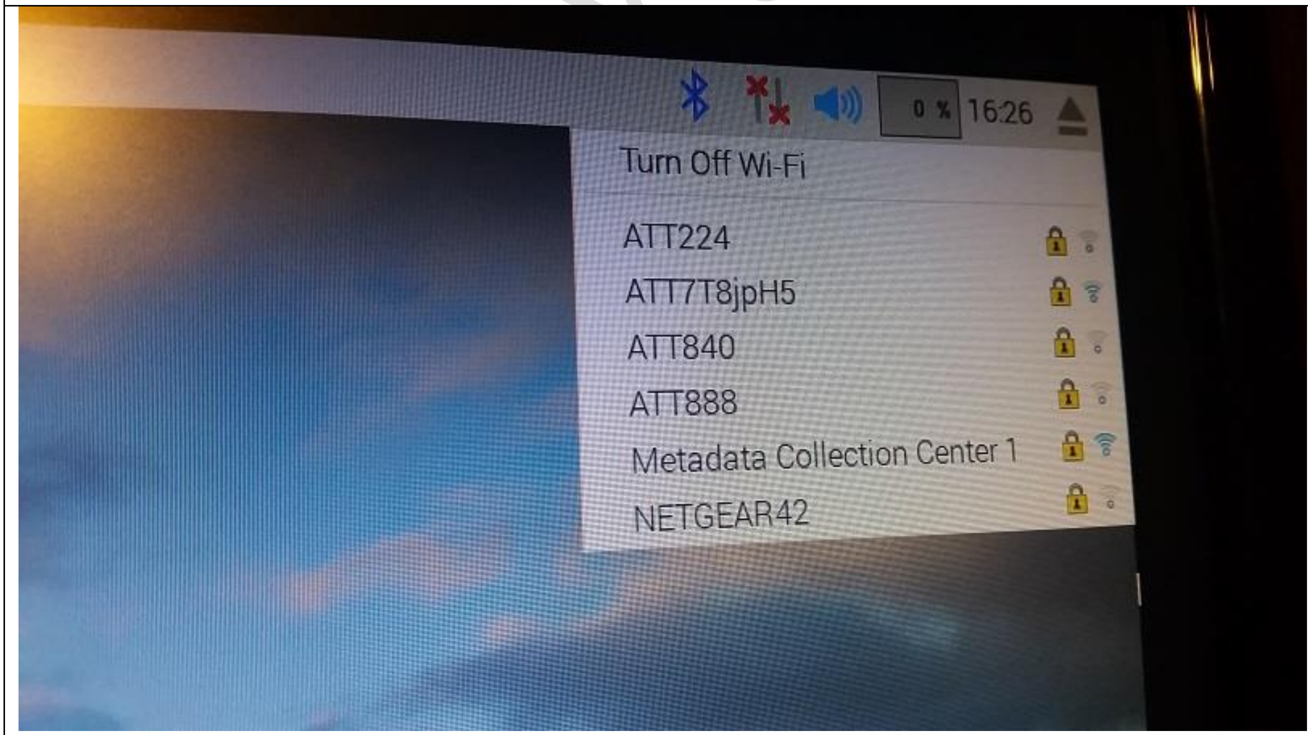
Now you can choose the operating system. In my case, the only option is Raspbian, so I will check that box, then click "install."



Let the process go, this will take a while, something like 20-30 minutes or so.



Once that's done, hit okay and the device should reboot to desktop. While on the desktop, wait for a moment for wifi to start up and find available connections. Connect to your wifi network if possible. You can also plug directly in with an ethernet cable if you don't have wifi. You can also just continue interacting directly with the Raspberry Pi with the mouse and keyboard connected to it if you like, but I prefer to access it remotely.



Once we've connected to our network, we'd like to actually interface with the Pi. First, we want to update. Open a terminal by either right clicking on the desktop and opening terminal that way, or by doing control+alt+t. Now, in the terminal, do:

$sudo apt-get update
and then
$ sudo apt-get upgrade
You do not type the $ sign, it's there to denote when you're typing something in the command line. The upgrade might take a minute. While we wait, your Raspberry Pi's default credentials are: username: pi password: raspberry. For some reason, the apt-get upgrade for me was taking absurdly long. You need to be connected to your network, and have internet access, so make sure you have those things first before doing this, but still was having trouble. One can solve this by doing:

$ sudo nano /etc/apt/sources.list

Then replace everything here with:

deb http://archive.raspbian.org/raspbian jessie main contrib non-free
deb-src http://archive.raspbian.org/raspbian jessie main contrib non-free

control+x, y, enter

$ sudo apt-get dist-upgrade

$ sudo apt-get update

$ sudo apt-get upgrade

To save some space you can also do: $ sudo apt-get purge wolfram-engine and then $ sudo apt-get autoremove. This alone freed up almost 700mb of space for me.

This is going to conclude the first part of this tutorial series. In the next tutorial, we're going to cover how we can remotely access our Raspberry Pi.

Experiment No: 8

Working basic commands on Raspberry pi

Sl no.	Equipment's Needed
1	Raspberry Pi
2	VGA cable

Introduction:

In this Experiment, we're going to have a quick crash course for using the terminal. Most of our interactions with the Raspberry Pi will be via shell, since this is the simplest and most lightweight to keep your Pi accessible.

To begin, let's assume you've just logged in. The location that you will usually log in to will be the root directory for your user. In this case, our user is "pi," so we log in to /home/pi. We can confirm this by using the command: pwd (print working directory)

pi@raspberrypi:~ $ pwd
/home/pi

Often times, to denote the terminal, you will see people use the $ sign. Since, most of the time, people wont be using the same usernames and hostnames, this is the standard to denote that we're operating in the terminal, so, instead, if you googled how you get your current directory in linux, you'd probably see: $ pwd. This doesn't mean that you should actually type the $ sign, it's just meant to denote that you're in the terminal.

Next, many times you might see a "permission denied" error when trying to do something. For many tasks, you need to act as the super user, like the administrator account. The command to act as the super user is sudo, which is short for "super user do." An example of this was when we wanted to update and upgrade, we used sudo. Be careful when using sudo to create files...etc. The creator is the owner, so if you use sudo to create the file, it can only be further edited by the super user. Many files will require sudo to edit, however.

To move around the system, you can use cd, which stands for "change directory." At any time, we can use ~ to reference the current user's home. In our case ~ is short for /home/pi. We can change directories into there with $ cd ~.

We were already there, but, just in case you weren't, you are now!

Next, to discover the contents of the directory you are in, you can use ls to list directory contents:

$ ls

pi@raspberrypi:~ $ ls
Desktop Documents Downloads Music Pictures Public python_games Templa

It wont always be the case, but often you will see different colors for different types of files. In our case, we just have directories, so they are all the same color.

Let's assume we want to make our own directory, to do this, we can use the mkdir command, short for make dir.

$ mkdir example

We can list the contents again:

$ ls

Seeing our new directory:

pi@raspberrypi:~ $ ls
Desktop Documents Downloads example Music Pictures Public python_games

Then we can change directories into our new directory:

$ cd example

Maybe we don't want to be here. We can move backwards with: $ cd ..

pi@raspberrypi:~/example $ cd ..
pi@raspberrypi:~ $

We can also move back a few directories at a time: $cd ../../

pi@raspberrypi:~ $ cd ../../
pi@raspberrypi:/ $ ls
bin boot debian-binary dev etc home lib lost+found man media mnt opt proc root run sbin srv sys tmp usr var

Let's go into our example dir now:

$ cd example

Next, we can create a simple file with any of the built in editors. There are many to choose from, I think nano is the easiest, so I will use that:

$ nano test.py

Nano can be used to both create or edit files. If the file doesn't yet exist, it will be created. If it does exist, it will allow you to save a new one.

Let's just add: print("hi") to this file. When done, we can exit with control+x, then press yto save, and then enter to keep the name.

To run this file, we can do: $ python test.py

```
pi@raspberrypi:~/example $ python test.py
hi
```

Next, let's make another directory:

$ mkdir downloads

$ cd downloads

Experiment No: 9

To demonstrate remote logging in raspberry pi

Sl no.	Equipment's Needed
1	Raspberry Pi
2	VGA cable

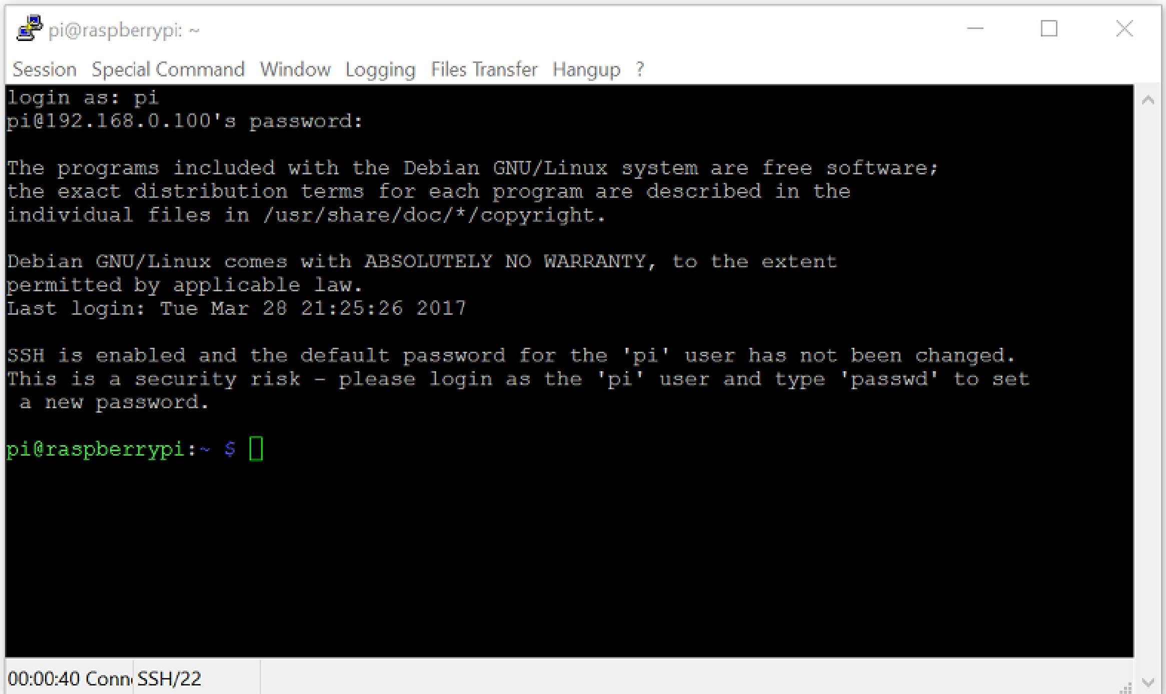
Introduction :

In this Experiment, we're going to cover how we can remotely access our Raspberry Pi, both with SSH and with a remote desktop client. We want to eventually be able to remotely access our Raspberry Pi because much of the "value" of the Raspberry Pi is its size, and that it can be put in a variety of places that we might not want to have a keyboard, mouse, and monitor attached to it at all times and we probably don't want to have to carry over all of this stuff when we do want to access it.

First, let's connect via shell (SSH). Open the terminal on the Raspberry Pi (control+alt+t), and type ifconfig. If you're connected via wifi, then go under the wlan section, and look for your inet address. This will be your local ip, something like 192.168.XX.XXX. We can use this to connect via SSH (user: pi, pass: raspberry), BUT we first have to enable the SSH server. To do this, type sudo raspi-config. Here, we can do quite a few things, but let's head into option #5 interfacing options, next choose the 2nd option forSSH and enable the server. Once this is done, you can shell into the Raspberry Pi.

On Windows, you will need to use an SSH client. But the recommended is PuTTY . Once downloaded, you can open PuTTy, fill in "host name" field with your Pi's local ip, hit enter, and then you will be asked for a username and password.

When successful, you should have something like:



This is identical to the terminal you accessed earlier from the Pi's desktop.

Now, sometimes, you might really want to get access to the desktop instead of just the terminal. To do this, you need the "host" to have remote desktop capabilities, as well as whatever remote PC you attempt to use to access it. First, let's deal with the Raspberry Pi. We're going to install xrdp.

sudo apt-get remove xrdp vnc4server tightvncserver

seudo apt-get update

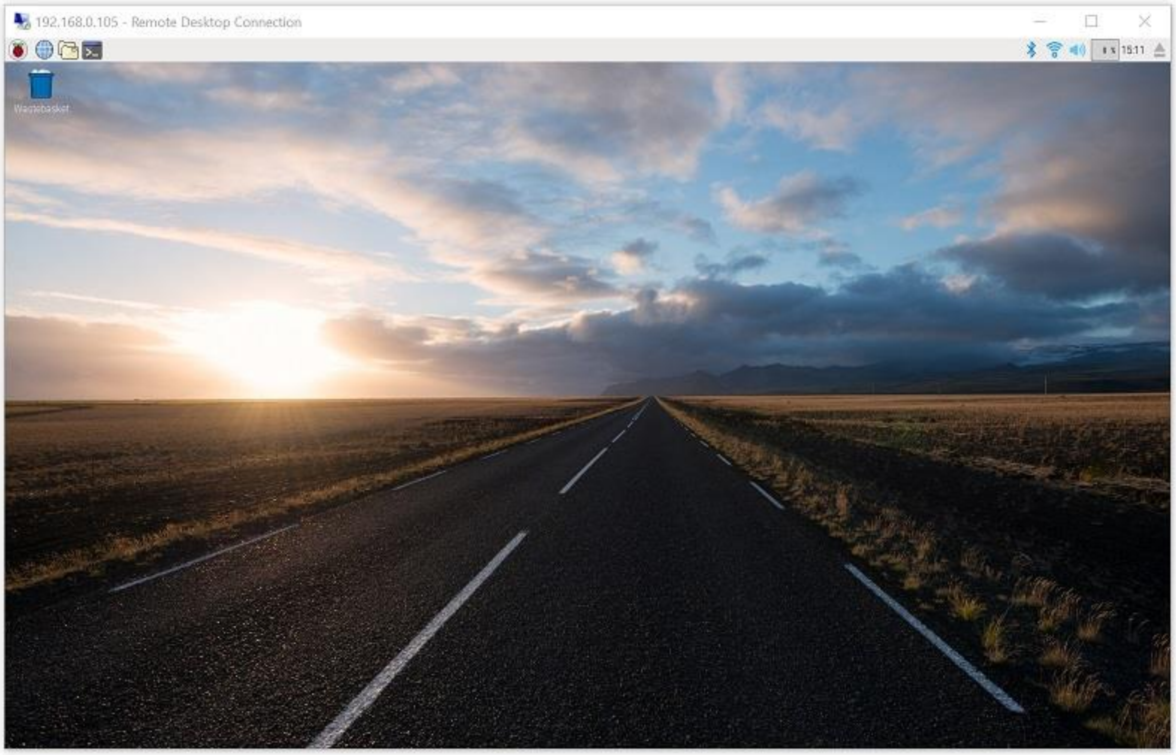
sudo apt-get install xrdp

sudo apt-get install tightvncserver

Now let's deal with the computer from which you plan to connect your Raspberry Pi.

For Mac and Windows, you can use the Microsoft application called Remote Desktop. On linux, you can use grdesktop (sudo apt-get install grdesktop). All three of these examples are going to act the same way. Run them, fill in the IP address of the Raspberry pi, the username, the password, and connect!

When done, you should have something like:

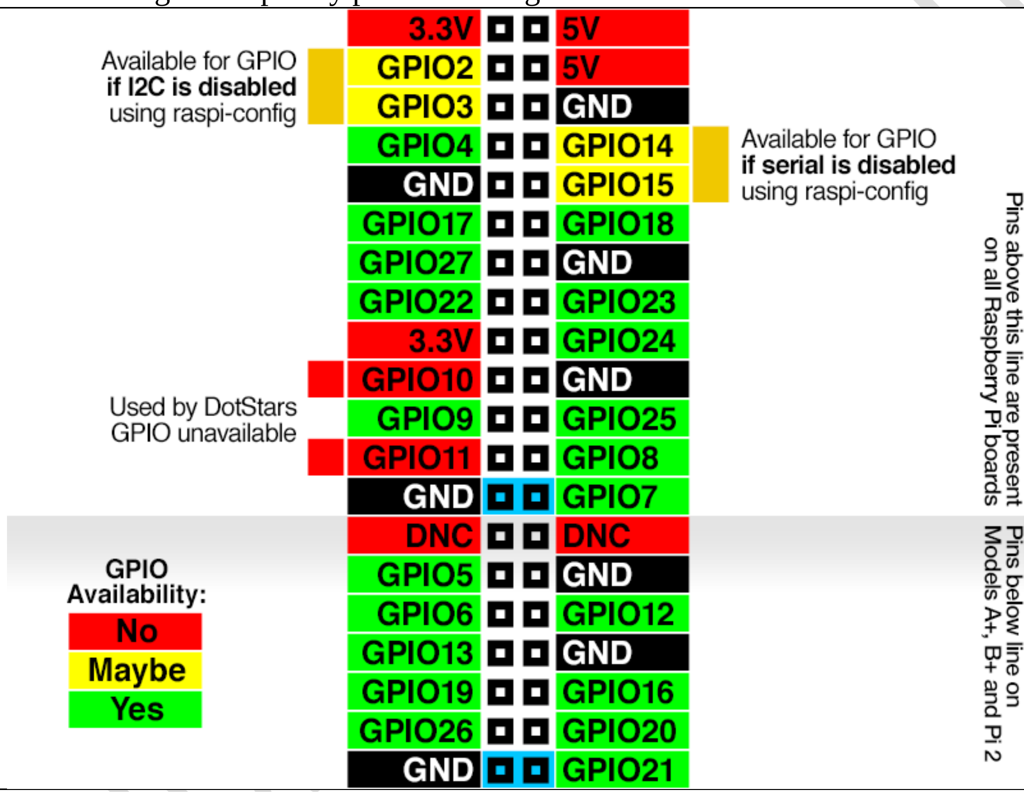


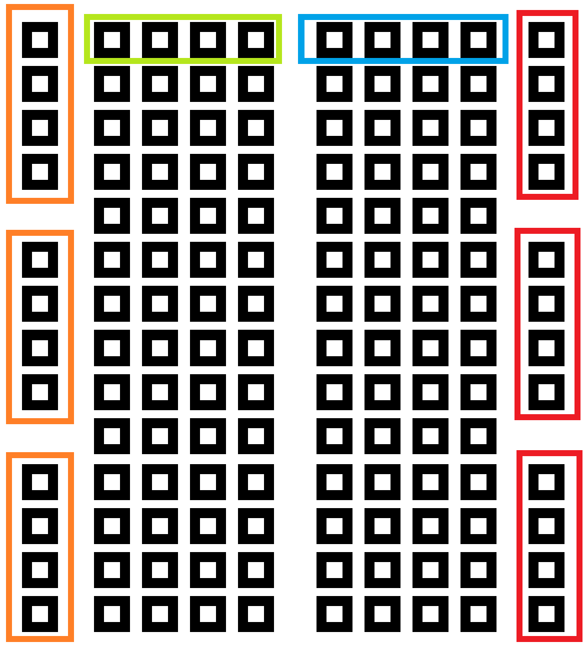
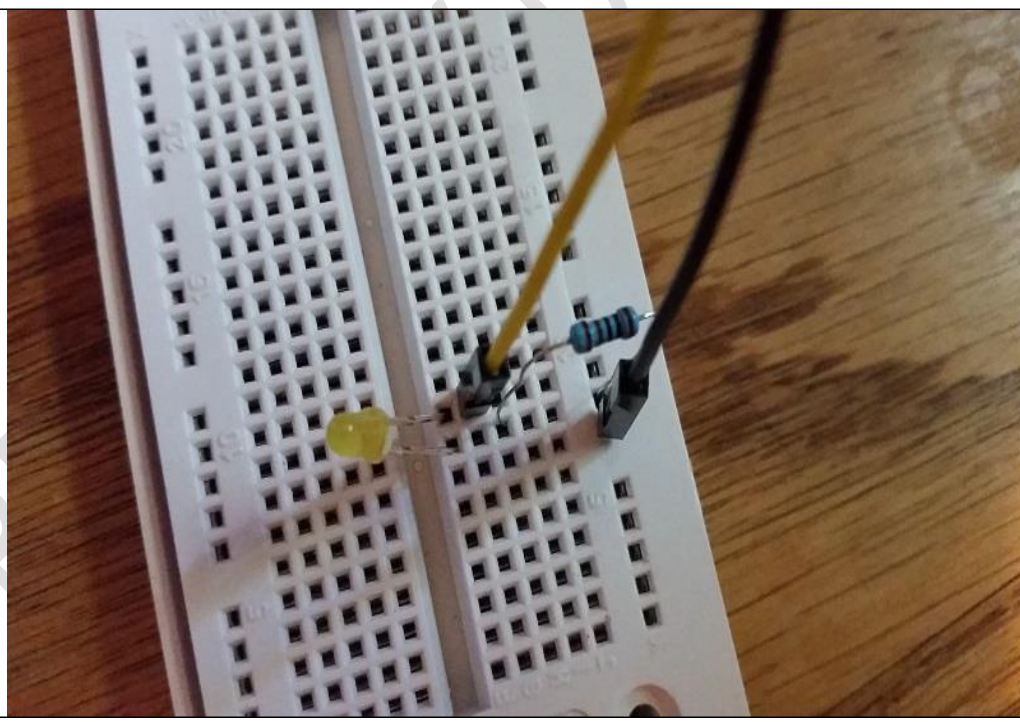
With the SSH server turned on, you will likely be seeing warnings every time you log in that your password is still the default password. The SSH server is turned off by default because, as the Pi has gained popularity, people who might not realize the security risk are using it in places like their homes and businesses. All it takes is someone to connect to your wifi, scan for other local ips, and try them all with default usernames and passwords for various devices, like the Raspberry Pi, and boom they're in. If you want to change your password, you can do sudo raspi-config, and it's the first option.

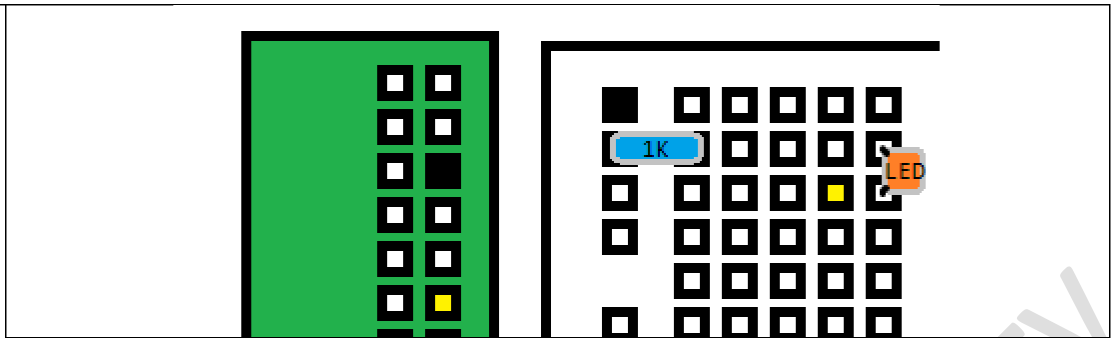
Experiment No: 10

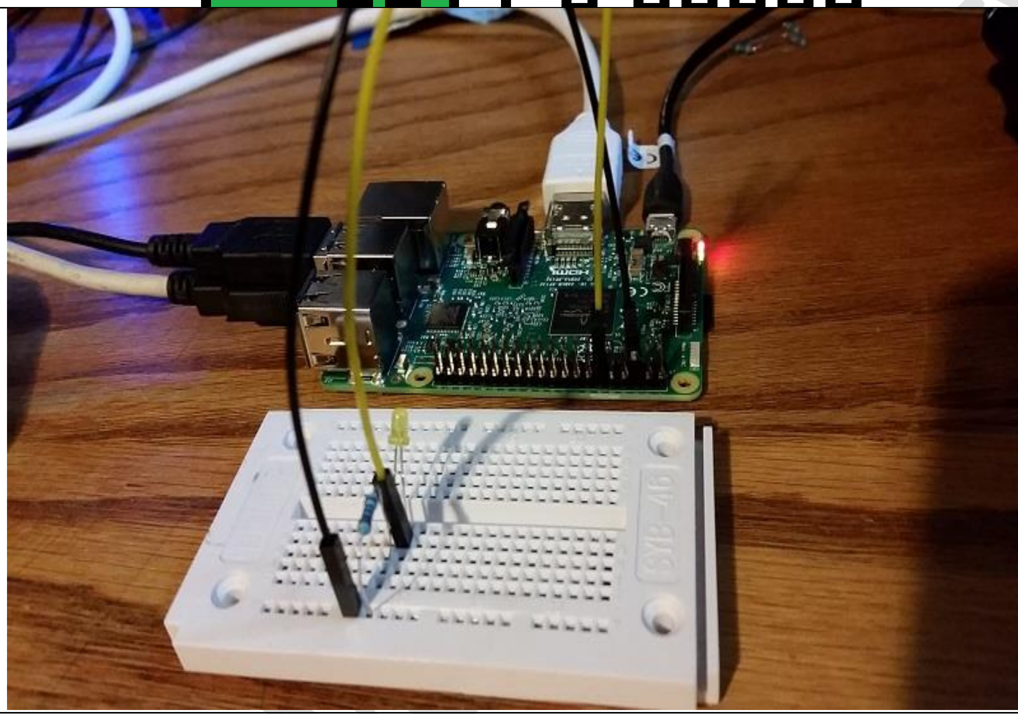
Raspberry pi program to implement blinking LED

Sl no.	Equipment's Needed
1	Raspberry Pi
2	VGA cable
3	Power Cable
4	Jumper Cables

Step 1:	Understanding of Raspberry pi Pin out diagram and Bread board connections
	

	
Step 2:	Make the connections as follows
	





Step 3:	Write the python program to implement blinking LED

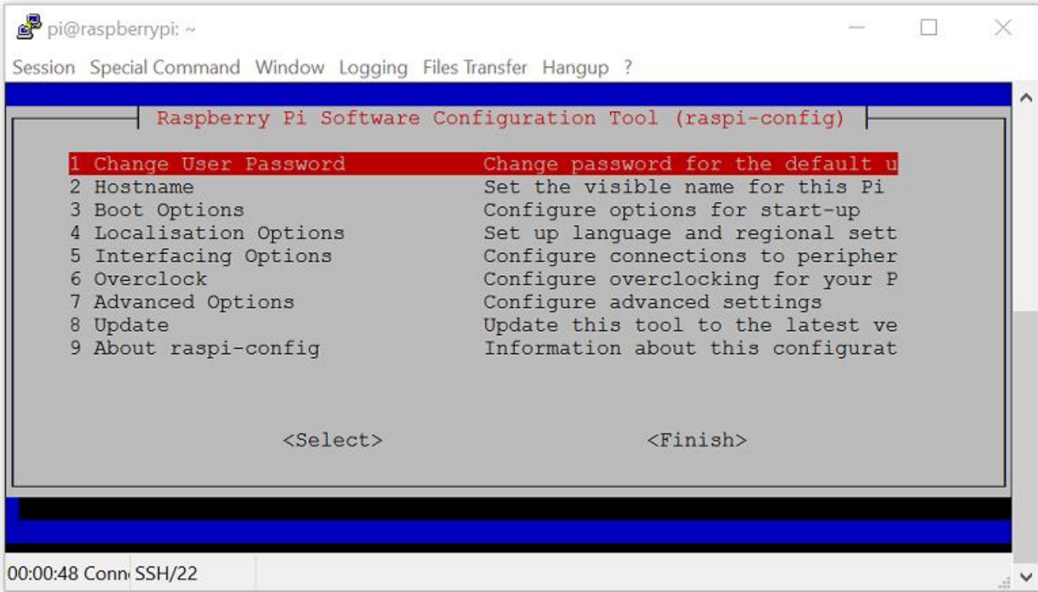
```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)

GPIO.setup(18, GPIO.OUT)
GPIO.output(18, GPIO.HIGH)

time.sleep(3)

GPIO.output(18, GPIO.LOW)
GPIO.cleanup()
```

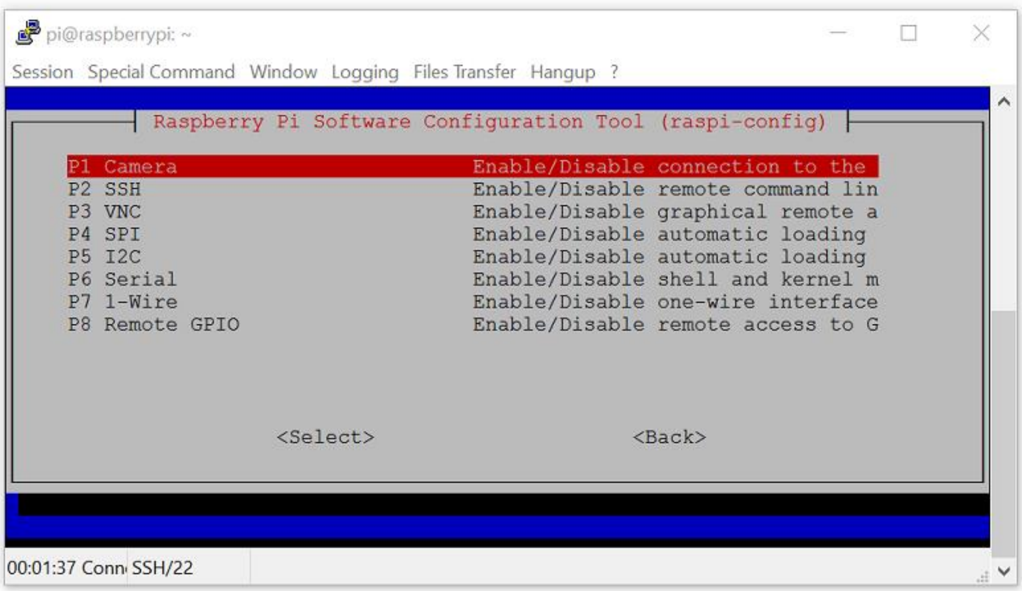
Experiment No: 11

Raspberry pi-camera module

Sl no.	Equipment's Needed
1	Raspberry Pi
2	Raspberry Pi Camera Module
3	VGA cable
4	Power Cable

Step 1:	Open the notch **"Connect camera"** module to raspberry pi Blue should be facing towards Ethernet port. 
Step 2:	**pi@raspberrypi:~** $cd Desktop/
Step 3:	Enable Camera pi@raspberrypi:~ /Desktop$ sudo raspi-config Interfacing option→ Enable camera





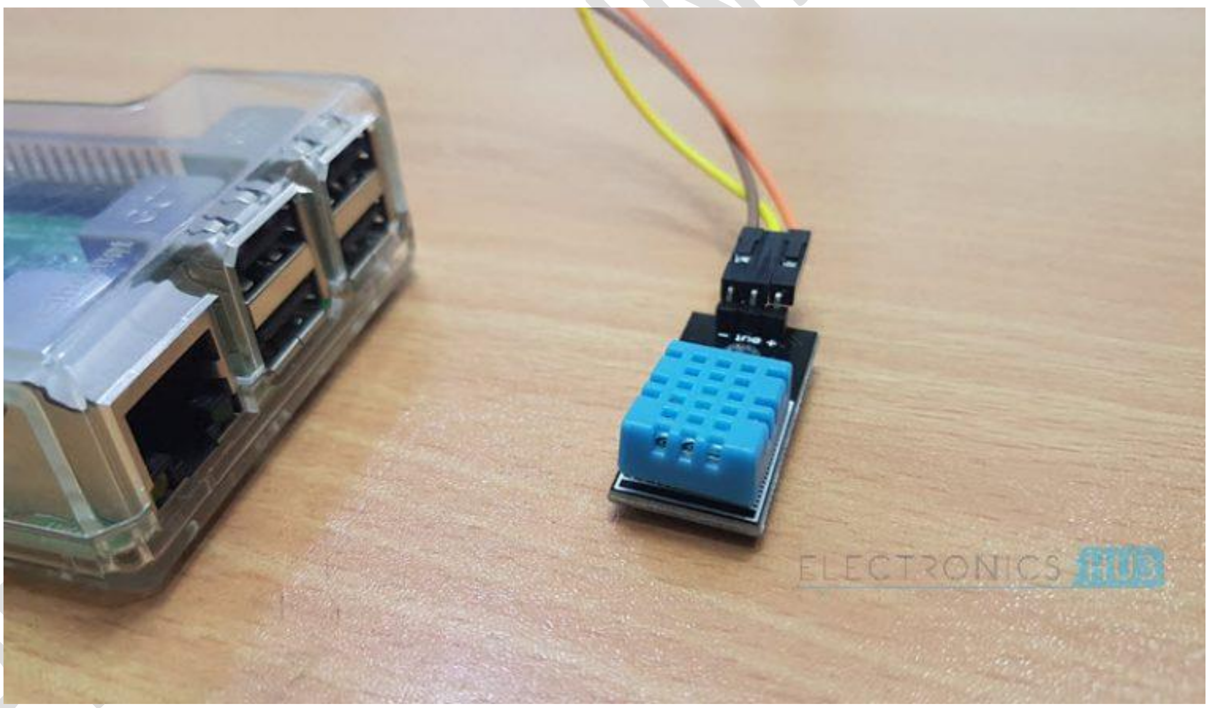
Yes to enable, and then go ahead and reboot:

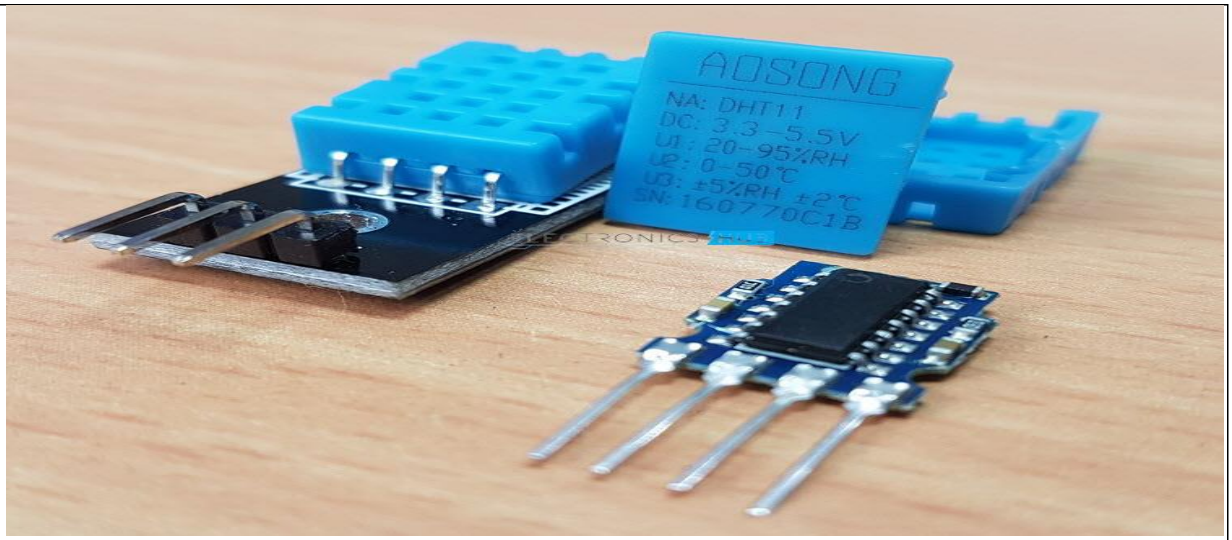
	
Step4:	Let's open the terminal next (control+alt+t) and do a $ sudo apt-get install python3-picamera. You will likely find that you already have it, but we want to be sure. Now, in our cameraexample.py file:
Step 5:	Write the Phython program pi@raspberrypi:~ /Desktop$ nano cameraexample.py import picamera import time camera = picamera.PiCamera() camera.capture('example.jpg') camera.vflip = True camera.capture('example2.jpg') camera.start_recording('examplevid.h264') time.sleep(5) camera.stop_recording()
Step 6:	Command to view recorded video $ omxplayer examplevid.h264.

Experiment No: 12

Raspberry pi program to implement temperature sensor

Sl no.	Equipment's Needed
1	Raspberry Pi Model 3
2	Raspberry Pi Camera Module
3	Digital humidity and temperature sensor-(DHT 22)
4	VGA cable
5	Power Cable

Step No.	Raspberry Pi DHT11 Humidity and Temperature Sensor Interface
1	**Raspberry Pi DHT11 Humidity and Temperature Sensor Interface** In this project, we will learn about DHT11 Humidity and Temperature Sensor and how the Raspberry Pi DHT11 Humidity Sensor interface works. By Interfacing DHT11 Temperature and Humidity Sensor with Raspberry Pi, you can implement a basic IoT Project like a simple Weather Station.  Overview 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.



We know that the output from the DHT11 Sensor is Digital. But how exactly we can read this digital data?

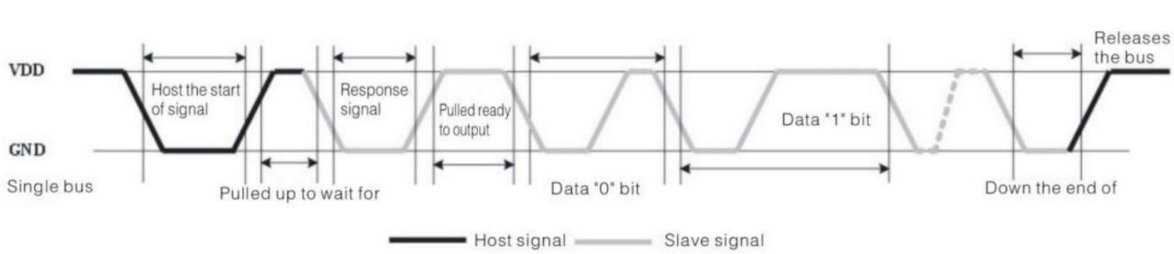
Reading Digital Output from DHT11

DHT11 uses a Single bus data format for communication. Only a single data line between an MCU like Arduino or Raspberry Pi and the DHT11 Sensor is sufficient for exchange of information.

In this setup, the Microcontroller acts as a Master and the DHT11 Sensor acts as a Slave. The Data OUT of the DHT11 Sensor is in open-drain configuration and hence it must always be pulled HIGH with the help of a 5.1KΩ Resistor.

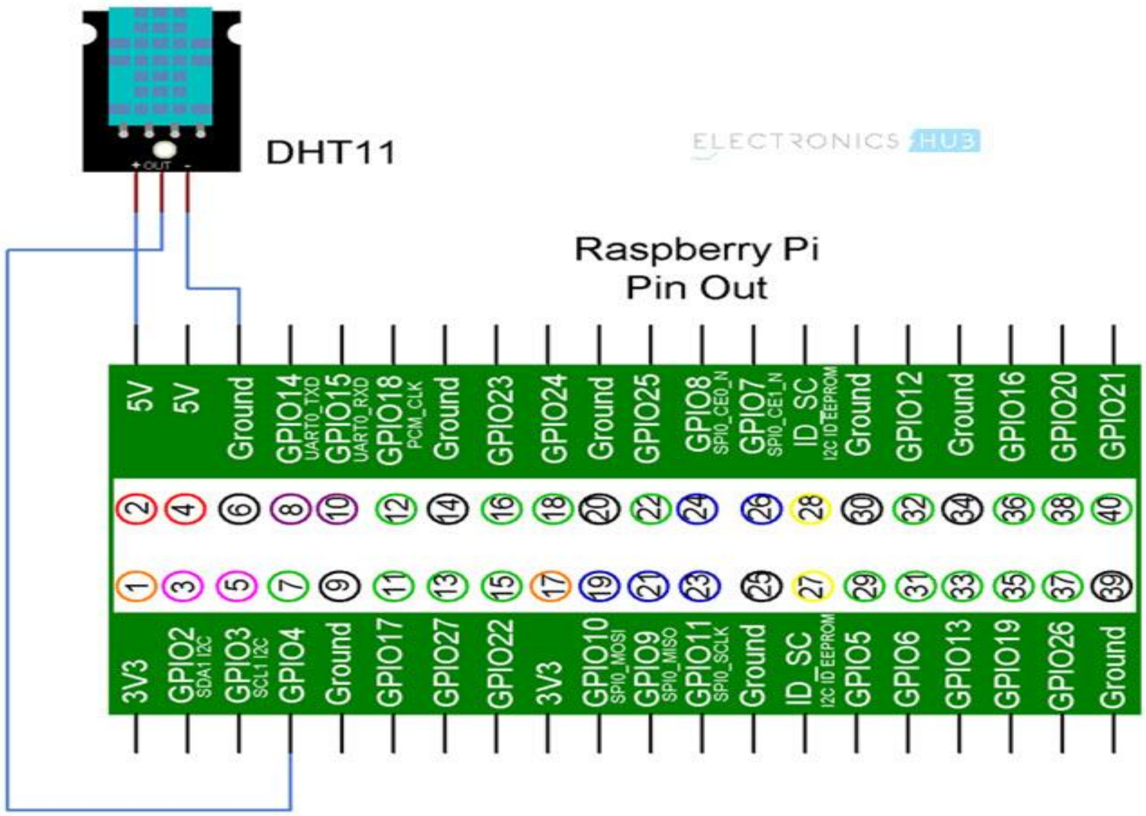
This pull-up will ensure that the status of the Data is HIGH when the Master doesn't request the data (DHT11 will not send the data unless requested by the Master).

Now, we will the how the data is transmitted and the data format of the DHT11 Sensor. Whenever the Microcontroller wants to acquire information from DHT11 Sensor, the pin of the



Microcontroller is configured as OUTPUT and it will make the Data Line low for a minimum

	time of 18ms and releases the line. After this, the Microcontroller pin is made as INPUT. The data pin of the DHT11 Sensor, which is an INPUT pin, reads the LOW made by the Microcontroller and acts as an OUTPUT pin and sends a response of LOW signal on the data line for about 80µs and then pulls-up the line for another 80µs. After this, the DHT11 Sensor sends a 40 bit data with Logic ‘0’ being a combination of 50µs of LOW and 26 to 28µs of HIGH and Logic ‘1’ being 50µs of LOW and 70 to 80µs of HIGH. After transmitting 40 bits of data, the DHT11 Data Pin stays LOW for another 50µs and finally changes its state to input to accept the request from the Microcontroller. **NOTE:** We have implemented this logic while programming the Arduino. But for Raspberry Pi, we used a library that takes care of all these things. Raspberry Pi DTH11 Humidity and Temperature Sensor Interface By interfacing the DHT11 Sensor with Raspberry Pi, you can build your own IoT Weather Station. All you need to implement such IoT Weather is a Raspberry Pi, a DHT11 Humidity and Temperature Sensor and a Computer with Internet Connectivity.
2	Circuit Diagram The following is the circuit diagram of the DHT11 and Raspberry Pi Interface.

<table>
<tr><td></td><td>
</td></tr>
<tr><td>3</td><td>Components Required

• Raspberry Pi 3 Model B
• DHT11 Temperature and Humidity Sensor
• Connecting Wires
• Power Supply
• Computer</td></tr>
<tr><td>4</td><td>Circuit Design

If you observe the circuit diagram, there is not a lot of stuff going on with respect to the connections. All you need to do is to connect the VCC and GND pins of the DHT11 Sensor to +5V and GND of Raspberry Pi and then connect the Data OUT of the Sensor to the GPIO4 i.e. Physical Pin 7 of the Raspberry Pi.</td></tr>
<tr><td>5</td><td>Installing DTH11 Library

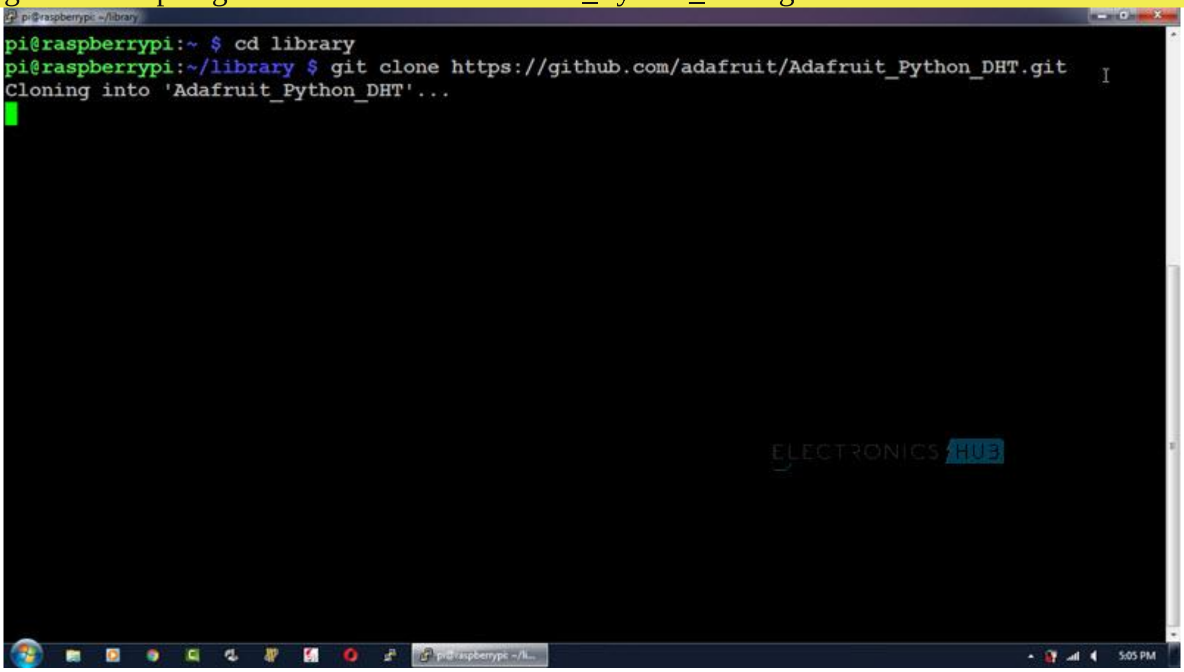
Since we are using a library called Adafruit_DHT provided by Adafruit for this project, we need to first install this library into Raspberry Pi.

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. You don’t</td></tr>
</table>

have to do that.

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



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

Code

As we are using the library Adafruit_DHT for this project, there is nothing much to do in the Python Programming part. All you need to do is to invoke the library with the Sensor and GPIO Pin and print the values of Temperature and Humidity.

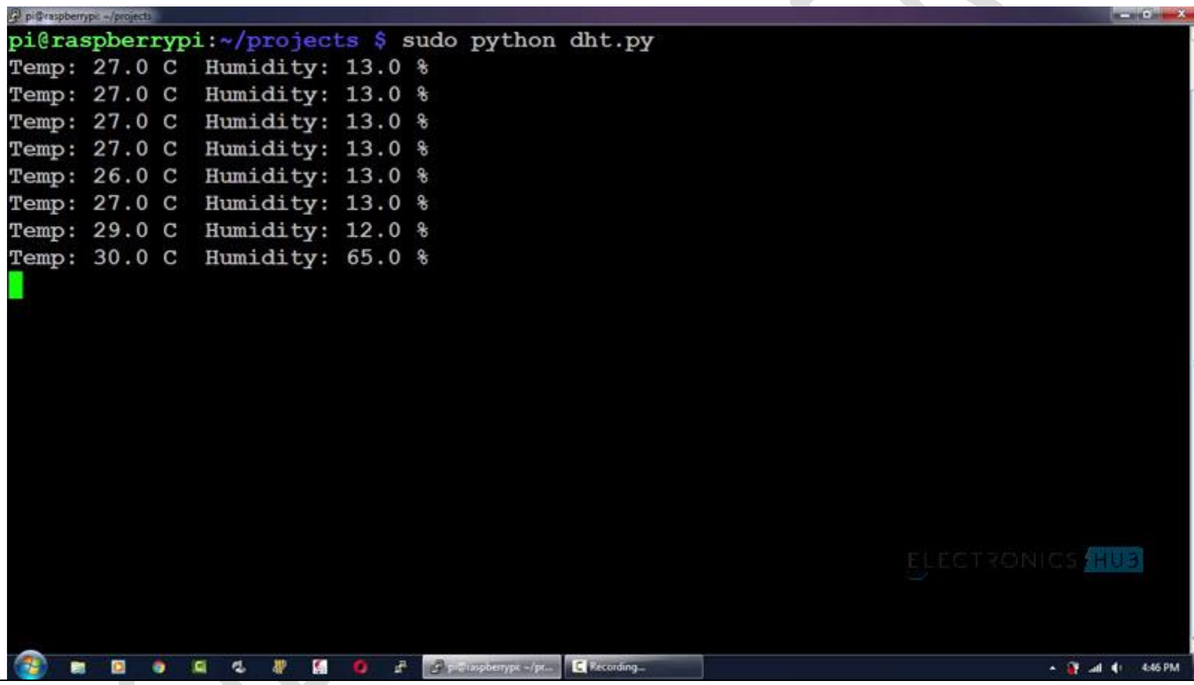
```
import sys
import Adafruit_DHT
import time
```

```
    while True:

        humidity, temperature = Adafruit_DHT.read_retry(22, 4)

        print 'Temp: {0:0.1f} C  Humidity: {1:0.1f} %'.format(temperature, humidity)


        time.sleep(1)
```

view rawRaspberry_Pi_DHT_11.py hosted with by GitHub

Working

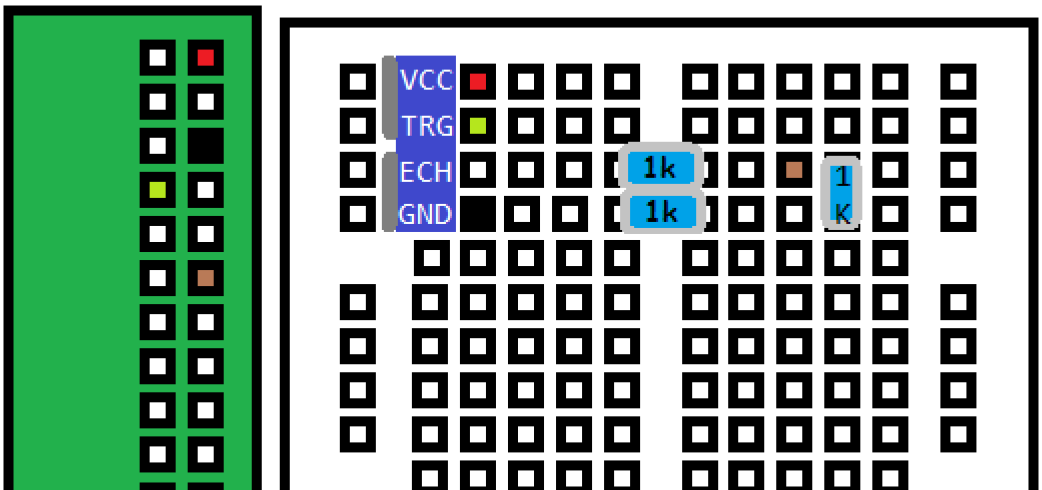
Make the connections as per the circuit diagram and install the library. Use the above python program to see the results.



Experiment No: 13

Demonstration of Raspberry Pi with distance sensor (ultrasonic sensor HC-SR04)

Sl no.	Equipment's Needed
1	Raspberry Pi Model 3
2	Raspberry Pi ultrasonic sensor HC-SR04
3	VGA cable
4	Power Cable

Step 1:	Make the connections as follows
	
	
Step 2:	Open the terminal CTRL+ALT+T and type $ Sudo nano Filename.py to open empty notepad
Step 3:	Write the python program to implement use of raspberry pi with temperature sensor
	`import RPi.GPIO as GPIO`

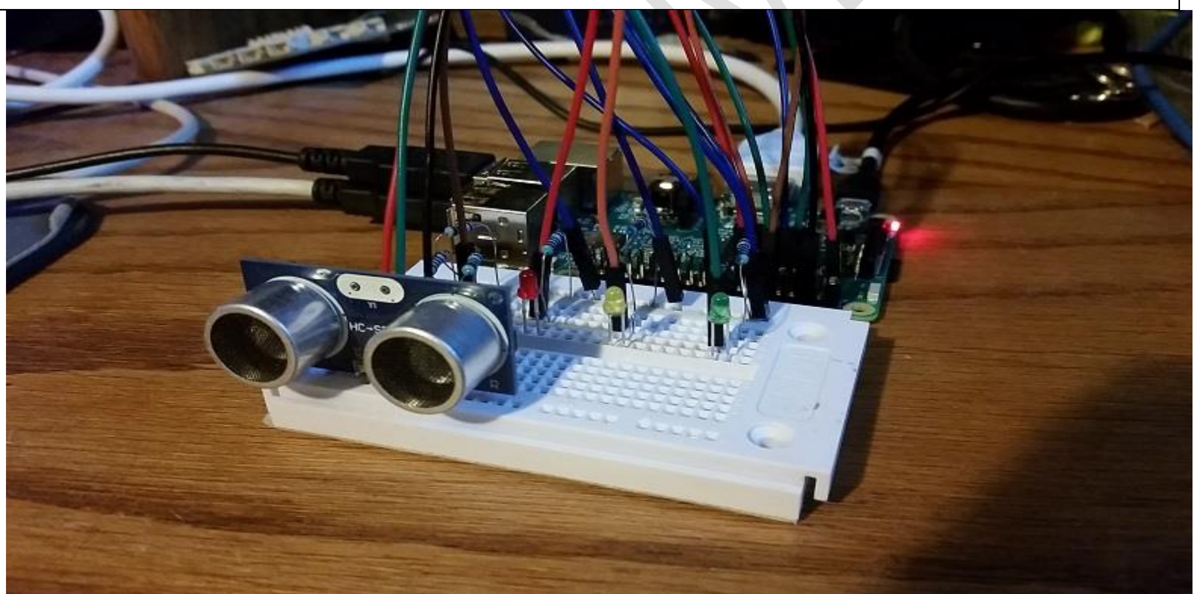
```
import time

GPIO.setmode(GPIO.BCM)

TRIG = 4
ECHO = 18

GPIO.setup(TRIG,GPIO.OUT)
GPIO.setup(ECHO,GPIO.IN)

GPIO.output(TRIG, True)
time.sleep(0.00001)
GPIO.output(TRIG, False)

while GPIO.input(ECHO) == False:
    start = time.time()

while GPIO.input(ECHO) == True:
    end = time.time()

sig_time = end-start

#CM:
distance = sig_time / 0.000058

#inches:
#distance = sig_time / 0.000148

print('Distance: {} centimeters'.format(distance))

GPIO.cleanup()
```

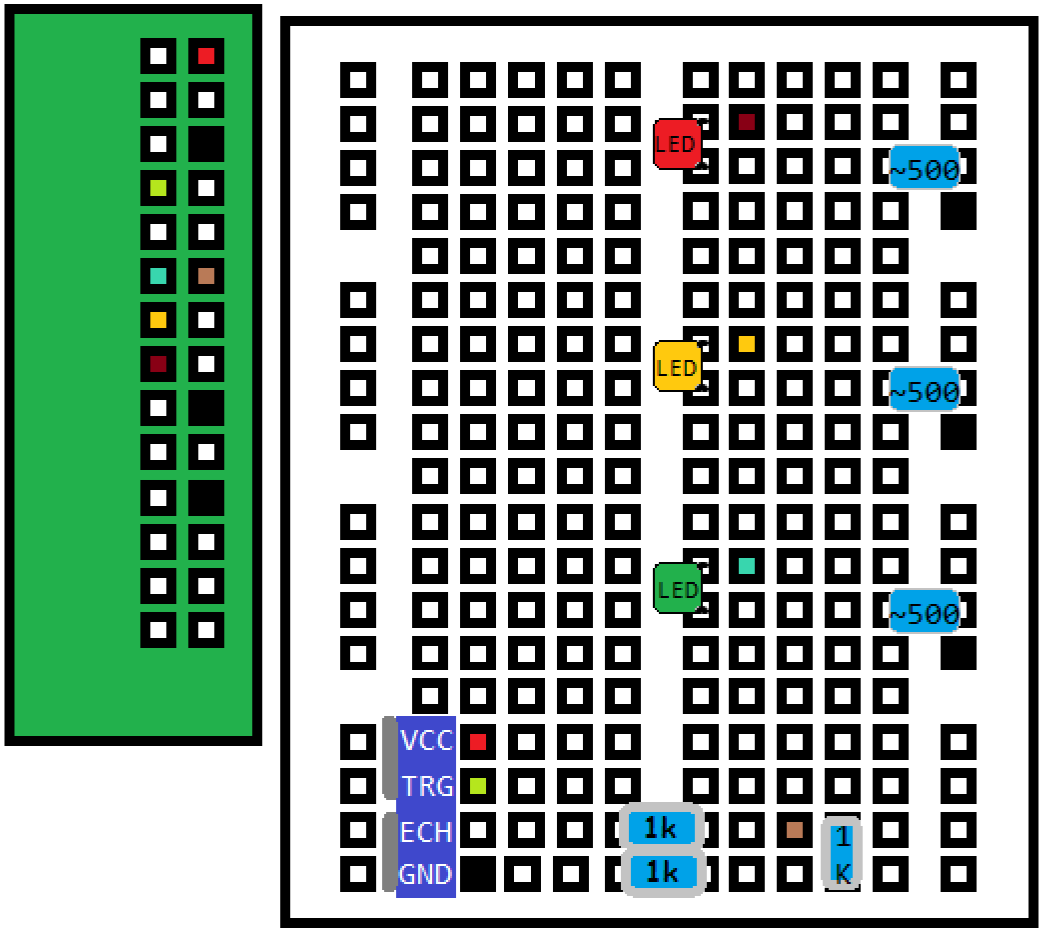
Step 3:	Compile the program using command $ Python Filename.py And verify the output

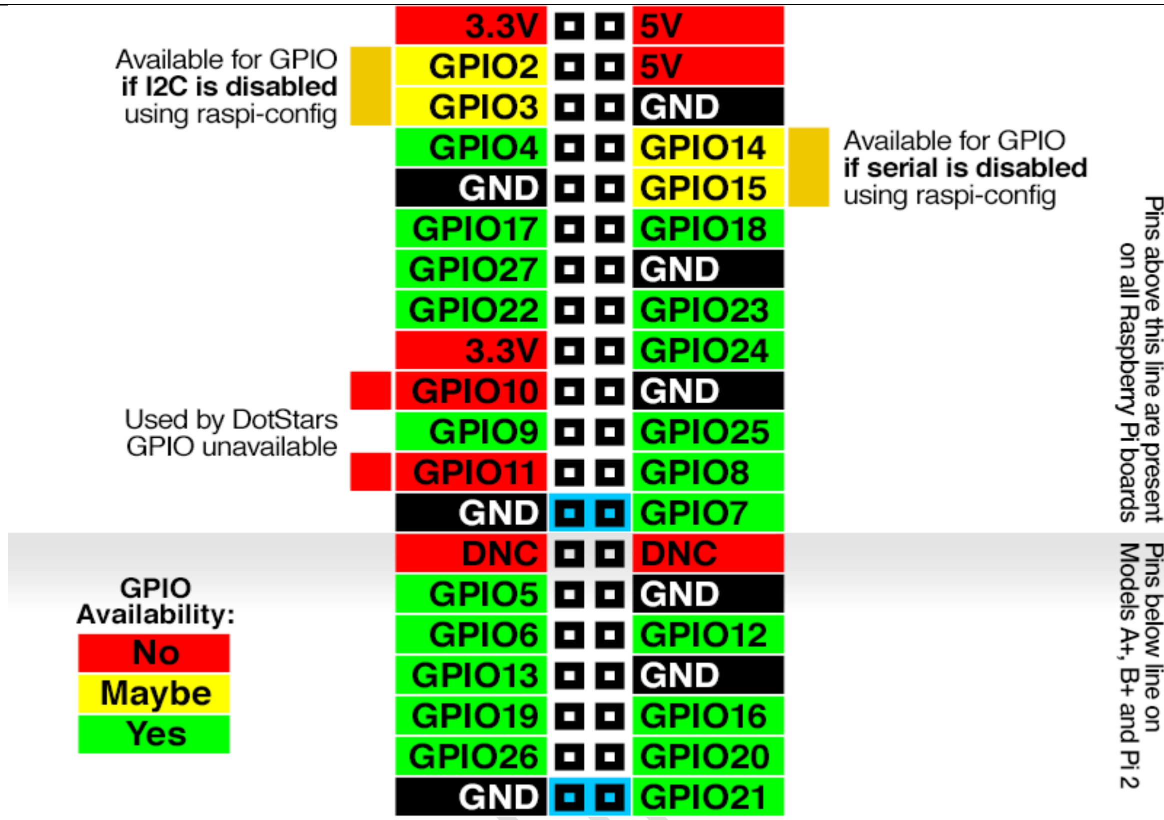
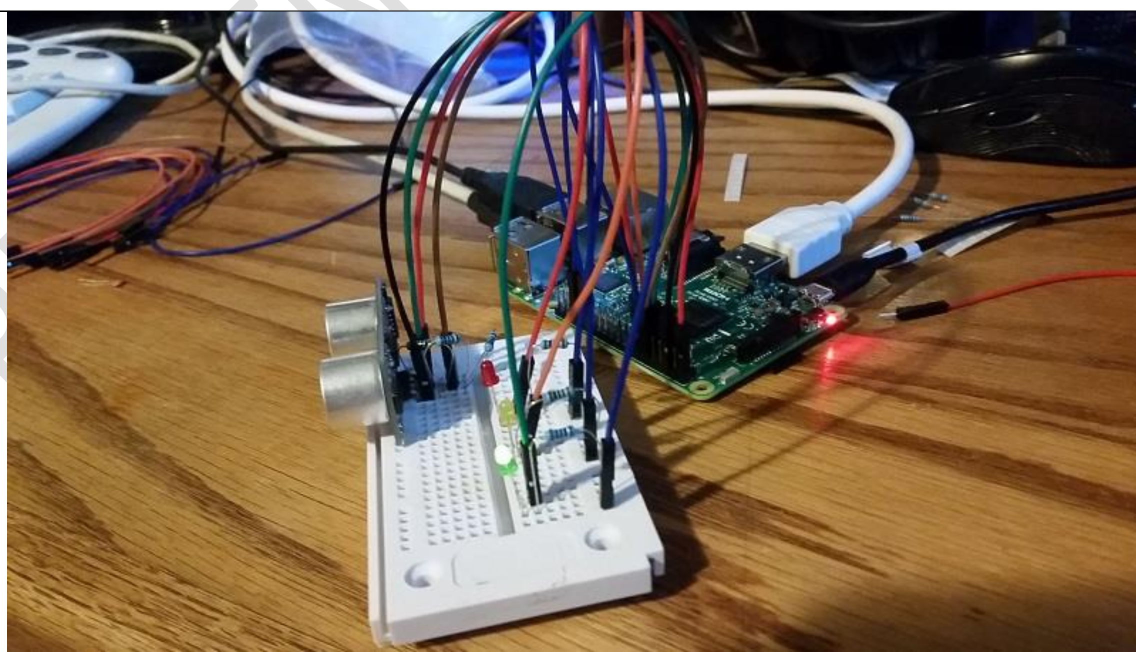
Experiment No: 14

Raspberry pi program to implement Garage spot light.

Sl no.	Equipment's Needed
1	Raspberry Pi Model 3
2	Raspberry Pi ultrasonic sensor HC-SR04
3	VGA cable
4	Power Cable
5	Jumper cables- Male to Female
6	3 different colored LEDs-RGB

Step 1:	The idea of a garage stop-light is to show green when you have plenty of room to pull your car forward in your garage, and then turn yellow as you approach the fully forward position, and then red when you should stop. We're going to build this system with our Raspberry Pi, and use some distances that we can easily test.
	
Step 2:	To start, we'll just leave the distance sensor hooked up as is, but now we're going to add the light circuits, only this time we have three. Here's how I set mine up:

	
Step 3:	Once you have everything hooked up and you're confident you've not messed anything up, you can power on your Raspberry Pi. You may want to re-reference the Adafruit image shared earlier:

	
Step 4:	Garage spot light script
	Output
	

	Hand getting close: