Project 6

Light control using Pi Bluetooth and Smart phone

Introduction

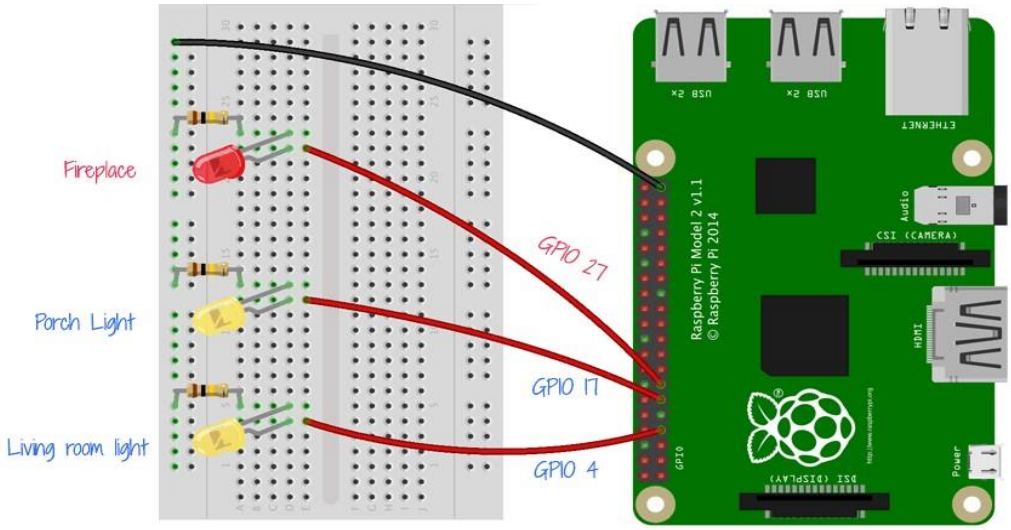
In this project you will learn how to control LED light with an android phone through Bluetooth connection with the raspberry pi. In place of LED light, you can use a greater number of lights or appliances using Relay as per your requirement.

Hardware Required

- Raspberry Pi
- 3 LED
- Jumpers
- Breadboard
- Blue Term Android App

Hardware Setup

Connect as per the given diagram
Living Room Light larger leg – Physical PIN No----------7 GPIO 4
Porch Light larger leg – Physical PIN No- ----------11 GPIO 17
Fireplace Room Light larger leg – Physical PIN No------- 13 GPIO 27



Pairing Devices with Raspberry Pi over Bluetooth:

Pairing Bluetooth Devices, like mobile phone, with Raspberry Pi is very easy. Here we have **paired our Android Smart phone with Raspberry Pi**. We have previously installed BlueZ in Pi, which provides a command line utility called ***"bluetoothctl"*** to manage our Bluetooth devices. Now open the *bluetoothctl* utility by below command:

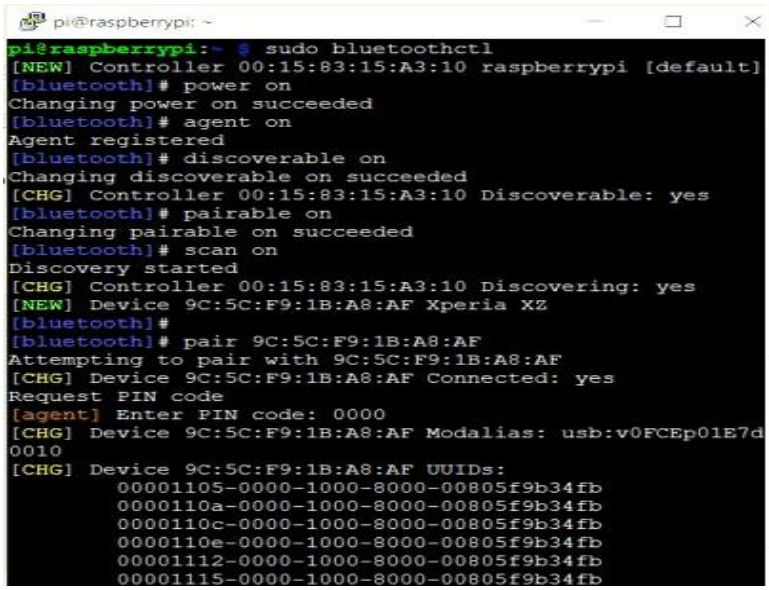
```
sudo bluetoothctl
```

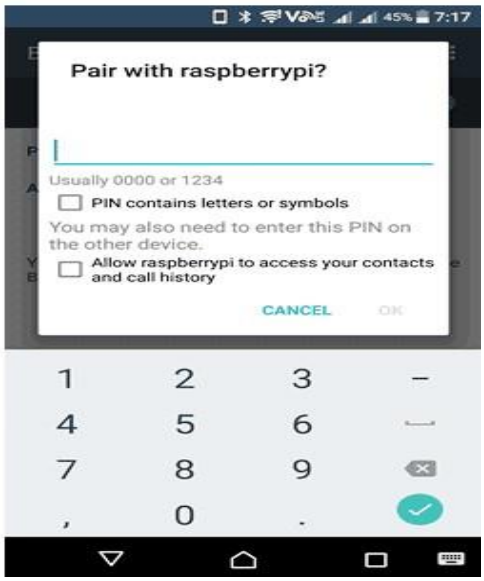
You can check all the commands of *bluetoothctl* utility by typing *'help'*. For now we need to enter below commands in given order:

```
[bluetooth]# power on

[bluetooth]# agent on

[bluetooth]# discoverable on

[bluetooth]# pairable on

[bluetooth]# scan on
```

After the last command *"scan on",* you will see your Bluetooth device (Mobile phone) in the list. Make sure that your mobile has Bluetooth turned on and visible by nearby devices. Then copy the MAC address of your device and pair it by using given command:

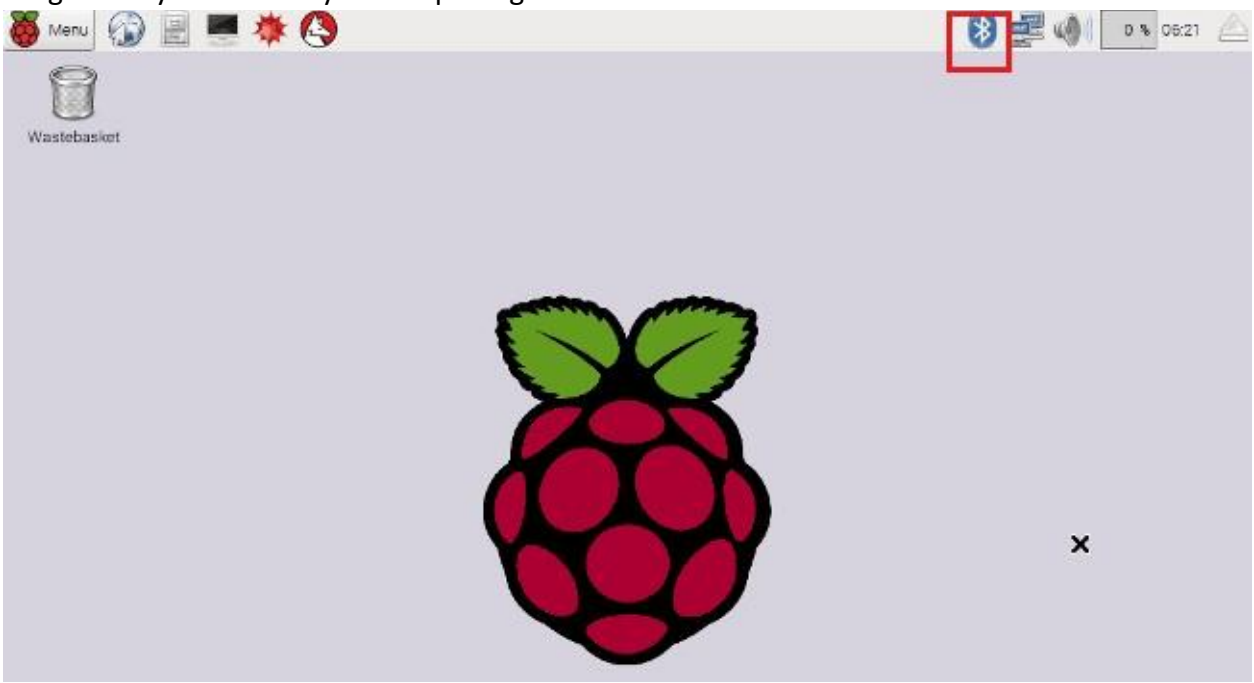
```
pair <address of your phone>
```





Then you will be prompted for Passcode or Pin in your Terminal console then type passcode there and press enter. Then type the same passcode in your mobile phone when prompted and you are now successfully paired with Raspberry Pi.

As told earlier, you can also use **Desktop interface to pair the Mobile phone**. After installing Blueman, you will see a Bluetooth icon in right side of your Raspberry Pi desktop as shown below, using which you can easily do the pairing.

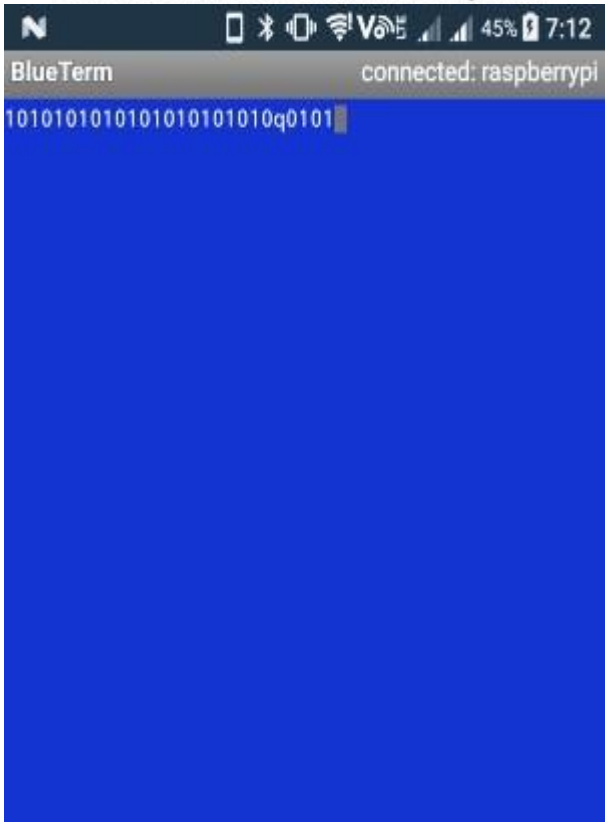


Controlling LED Remotely with Android App BlueTerm:

Now after setting up all the things and successfully try paring the Smart Phone over bluetooth, we need to install an**Android App for communicating with Raspberry Pi using a Bluetooth Serial Adapter,** so that we can control the GPIO pins of Raspberry Pi. As told earlier RFCOMM/SPP protocol emulates serial communication over Bluetooth, so we installed here **BlueTerm App** which supports this protocol.

You can also use any other Bluetooth Terminal App which supports communication via RFCOMM socket.

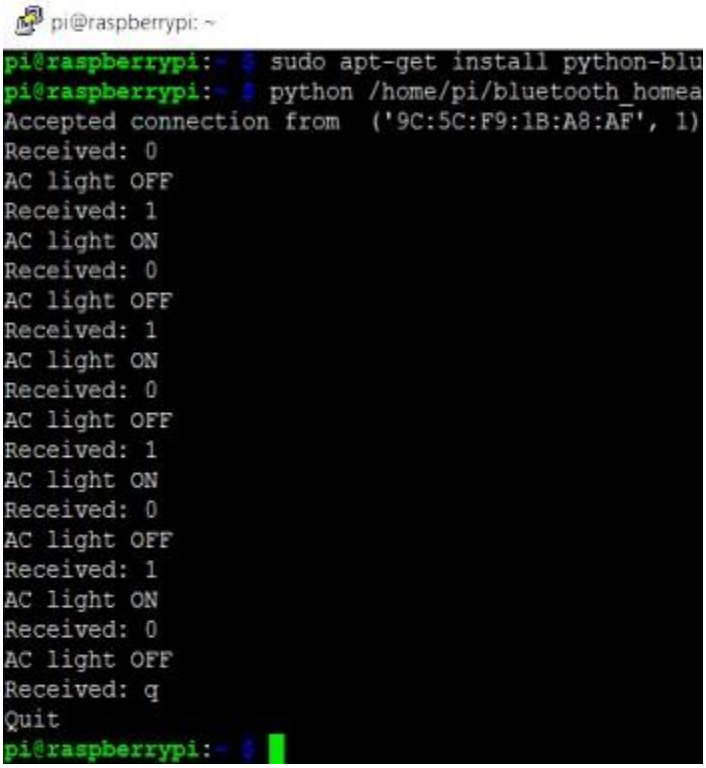
Now after down loading and installing the BlueTerm App, **run the below given Python Program** from the terminal and connect the paired *raspberrypi* device from the BlueTerm App at the same time.



After successful connection you will see *connected:raspberrypi* at the top right corner of the App as shown below:

Now you can just **enter the following commands from the BlueTerm app to make the light ON and OFF**. Press ‘q’ to exit the program.

Commands:
10 – Living Room Light OFF
11 – Living Room Light ON
20– Pouch Light OFF
21– Pouch Light ON
30 – Fireplace Light OFF
30 – Fireplace Light ON
Q – Quit



Python Coding

```
import RPi.GPIO as GPIO        #calling for header file which helps in using GPIOs of PI

LED1=4

LED2=17

LED3=27

GPIO.setmode(GPIO.BCM)     #programming the GPIO by BCM pin numbers. (like PIN40 as GPIO21)

GPIO.setwarnings(False)

GPIO.setup(LED1,GPIO.OUT)

GPIO.setup(LED2,GPIO.OUT)

GPIO.setup(LED3,GPIO.OUT)

GPIO.output(LED1,0)

GPIO.output(LED2,0) GPIO.output(LED3,0)
server_socket=bluetooth.BluetoothSocket( bluetooth.RFCOMM
)
```

```
port = 1 server_socket.bind(("",port))
server_socket.listen(1) client_socket,address =
server_socket.accept() print "Accepted
connection from ",address while 1:
        data = client_socket.recv(1024)
        print "Received: %s" % data
                                                If (data == "10"):    #if '10' is sent from the Android App,
turn OFF the LED
                print ("GPIO 4 LOW, LED OFF")
        GPIO.output(LED,0)
                                                if (data == "11"):    #if '11' is sent from the Android App,
turn ON the LED
                print ("GPIO 4 HIGH, LED ON")
        GPIO.output(LED,1)
                                                if (data == "20"):    #if '20' is sent from the Android App,
turn OFF the LED
                print ("GPIO 17 LOW, LED OFF")
        GPIO.output(LED,0)
                                                if (data == "21"):    #if '21' is sent from the Android App,
turn ON the LED
                 print ("GPIO 17 HIGH, LED ON")
        GPIO.output(LED,1)
                                                if (data == "30"):    #if '30' is sent from the Android App,
turn OFF the LED
                print ("GPIO 27 LOW, LED OFF")
        GPIO.output(LED,0)
                                                if (data == "31"):    #if '31' is sent from the Android App,
turn ON the LED
                print ("GPIO 27 HIGH, LED ON")
GPIO.output(LED,1)
```

```
if (data == "q"):
        print ("Quit")
```

```
                    break

client_socket.close() server_socket.close()
```

Output

In this way we can control Raspberry Pi GPIO with android phone through Bluetooth connection, and also at place of LED you can connect Relays to connect AC appliances.