1. Blinking of LED using Raspberry Pi.

Circuit Diagram:

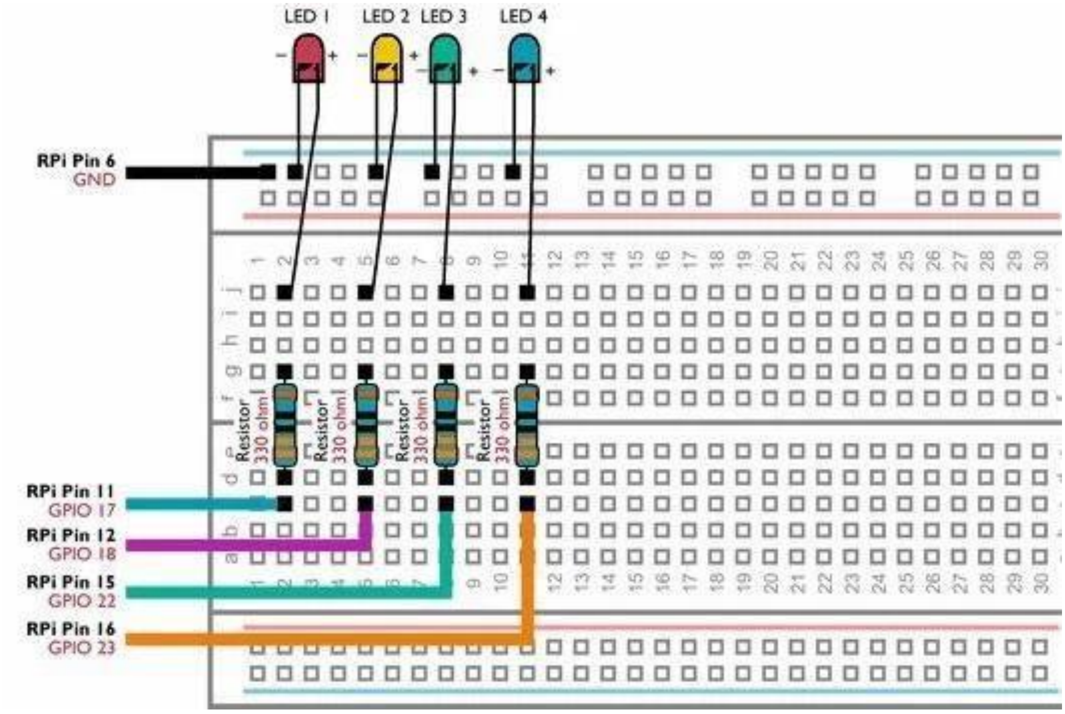
connect positive terminal of LED in to pin no 7 connect
negative terminal of LED into pin no 9

Code: import RPi.GPIO

```
as GPIO import time

GPIO.setmode(GPIO.BOARD)
GPIO.setup(7,GPIO.OUT)

   GPIO.output(7,True)
print("LED ON")     time.sleep(1)
GPIO.output(7,False)
   print("LED OFF")
   time.sleep(1)
GPIO.cleanup()
```

B) Blink 4 LEDs in pattern.

Circuit Diagram:



Code:

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BOARD)
GPIO.setup(11,GPIO.OUT)
GPIO.setup(12,GPIO.OUT)
GPIO.setup(15,GPIO.OUT)
GPIO.setup(16,GPIO.OUT)

for i in range(5):
GPIO.output(11,True)
   GPIO.output(12,False)
   GPIO.output(15,False)
GPIO.output(16,False)
   time.sleep(0.5)

   GPIO.output(11,False)
   GPIO.output(12,True)
   GPIO.output(15,False)
GPIO.output(16,False)
   time.sleep(0.5)

   GPIO.output(11,False)
   GPIO.output(12,False)
   GPIO.output(15,True)
GPIO.output(16,False)
   time.sleep(0.5)

   GPIO.output(11,False)
   GPIO.output(12,False)
   GPIO.output(15,False)
GPIO.output(16,True)
   time.sleep(0.5)

GPIO.cleanup()
```

2. Displaying Time over 4-Digit 7-Segment Display Using Raspberry Pi.

Hardware Requirements:-

- Raspberry Pi Model A/B/B+
- 4 digit 7 Segment Display
- Jumper wires (Female to Female)

Pin Configuration:-

Connect your 4 digit 7 segment display with Raspberry Pi's GPIO Pins.

Board Pin	**Function**	**RPI Physical Pin**	**Raspberry Function**
GND	Ground	14	GND
VCC	+ 5V Power	4	5V
DI0	Data In	18	GPIO 24
CLK	Clock	16	GPIO 23

Step 1: Open terminal and Download Python Script.

```
wget https://raspberrytips.nl/files/tm1637.py
```

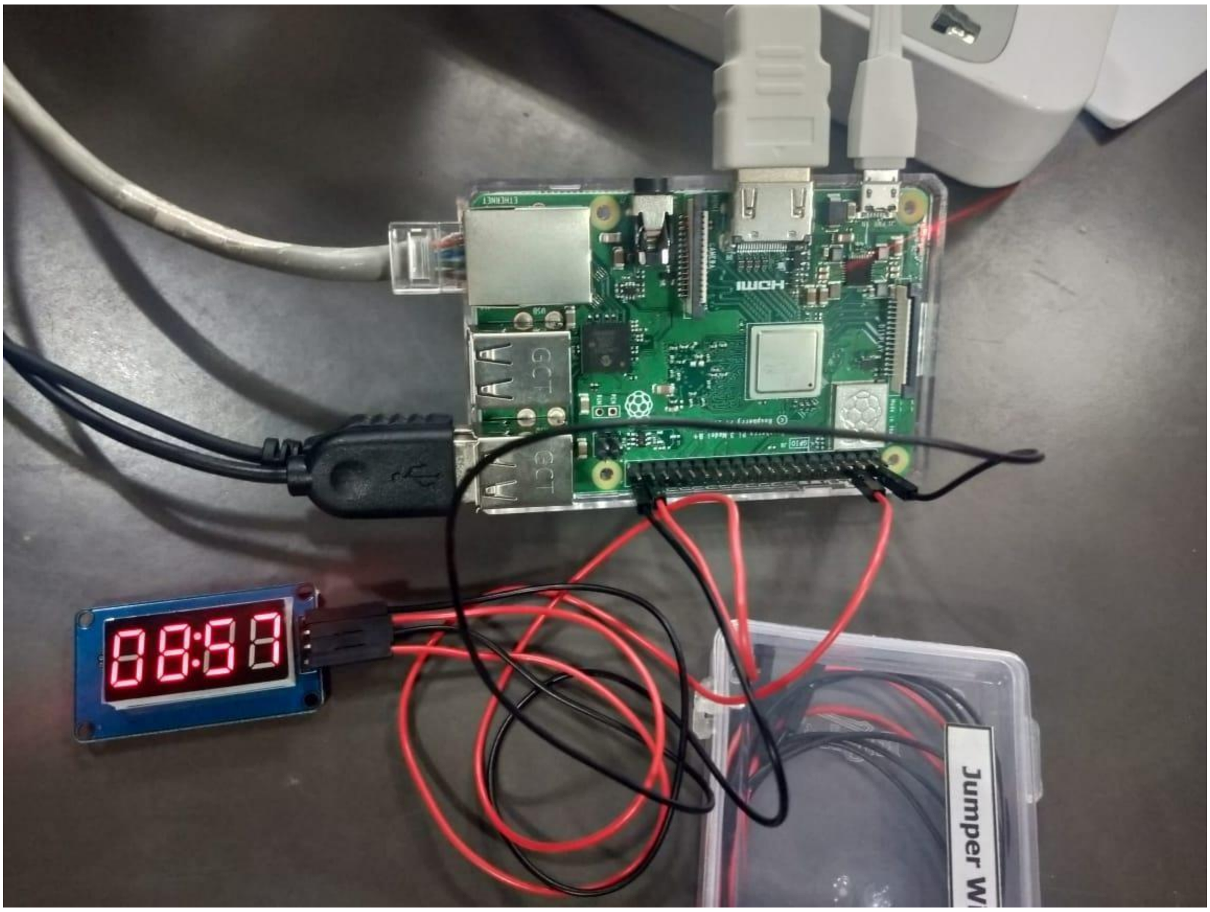
Step 2: Write Python Script to display Time and save as 7seg.py

```
import sys
import time
import tm1637
import datetime
import RPi.GPIO as GPIO

#clock->GPIO21(pin 40)
#DI->GPIO20(pin 38)

Display=tm1637.TM1637(21,20,tm1637.BRIGHT_TYPICAL)
Display.Clear()
Display.SetBrightnes(1)
```

```
while(True):     now =
datetime.datetime.now()
hour = now.hour
minute = now.minute
second = now.second
    currenttime = [ int(hour / 10), hour % 10, int(minute / 10), minute % 10 ]
Display.Show(currenttime)
    Display.ShowDoublepoint(second % ff5)
time.sleep(1)
```

Step 3: Start the script with following command.

```
python 7seg.py
```



3. Sensor interfacing with Raspberry Pi.

Hardware Requirements:-

- Raspberry Pi Model A/B/B+
- Fingerprint Module
- Serial USB Converter
- Jumper Wires

Pin Configuration:-

Fingerprint Module	**USB Serial Converter**
5V	5.0 V
GND	GND
TXD	RXD
RXD	TXD

Step 1: Connect fingerprint module to Raspberry Pi USB port by using USB to Serial converter.

Step 2: To install this library, root privileges are required. So login with root user.

```
sudo bash
```

Step 3: Download some required packages using wget command.

```
wget -O - http://apt.pm-codeworks.de/pm-codeworks.de.gpg | apt-key add -
wget http://apt.pm-codeworks.de/pm-codeworks.list -P /etc/apt/sources.list.d/
```

Step 4: Update the Raspberry Pi

```
apt-get update
```

Step 5: Install the downloaded finger print sensor library

```
apt-get install python-fingerprint –yes
```

Step 6: To return to the normal shell (under the Pi user).

```
exit
```

Step 7: Now go to the examples directory.

```
cd   /usr/share/doc/python-fingerprint/examples/
```

Step 8: Run example_enroll.py script to store new fingerprint

```
sudo python example-enroll.py
```

Note:

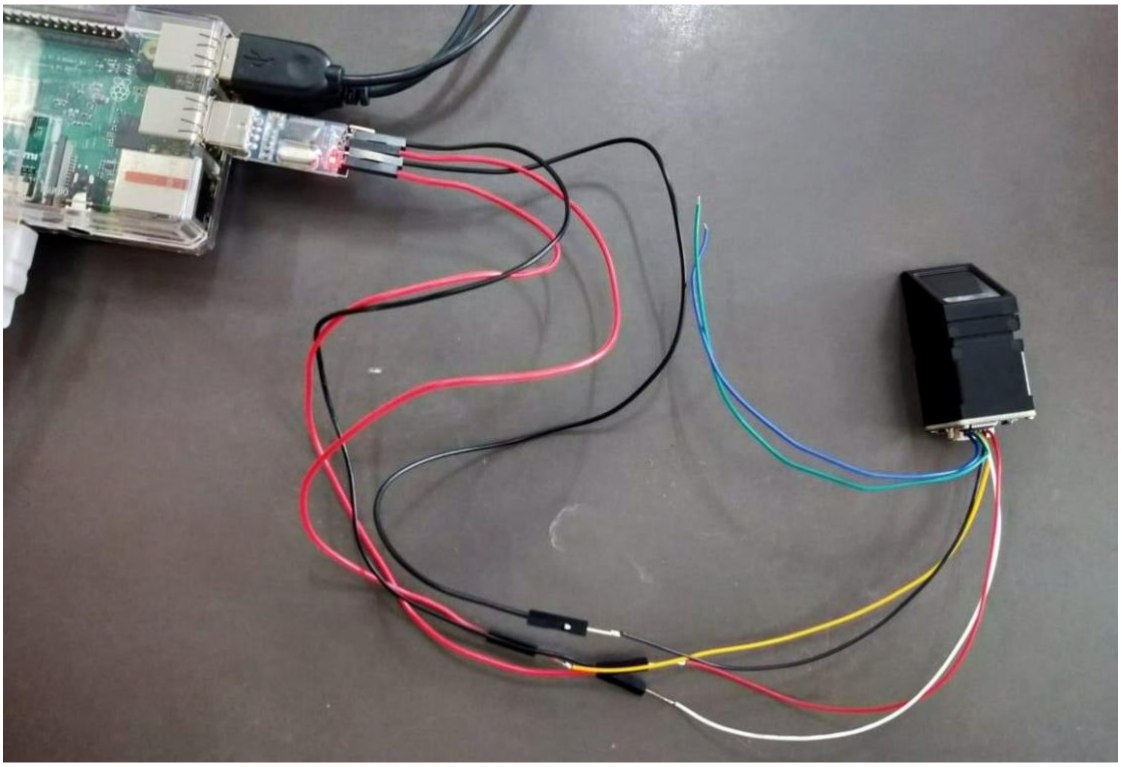
Put your finger on the fingerprint sensor, wait for the instruction in the terminal and remove meyour finger as soon as it is written there. Afterwards you have to put your finger a second time for the verification and the imprint is stored in the next number.

Step 9: Run example_search.py script to see whether our finger is recognized.

```
sudo python example-search.py
```

Note:

Put the same finger on glass surface. If the fingerprint is detected, it displays found message. If fingerprint is not detected, then gives "No match Found" message.

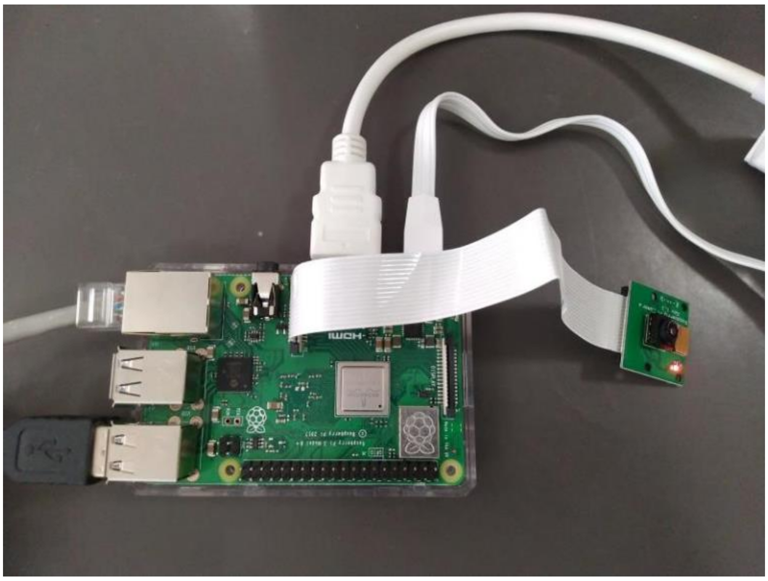


4. Image with Raspberry Pi and Pi Camera.

Hardware Requirement:

- Raspberry Pi
- Pi Camera

Connection:



Step 1: Write a Python Script to capture images and save them in folder.

from time import sleep from picamera import PiCamera

```
camera=PiCamera()
camera.resolution=(1280,720)
camera.start_preview() sleep(10)
camera.capture('/home/pi/Pictures/img.jpg')
camera.stop_preview()
```

Methods used:

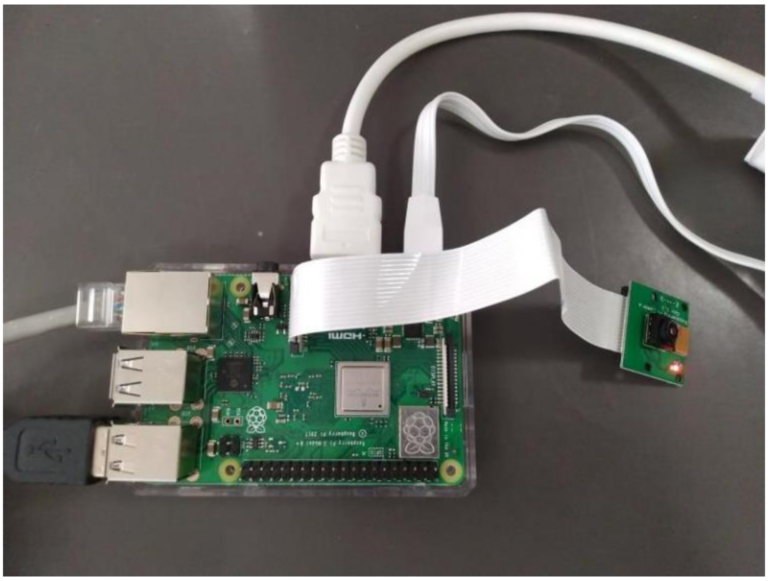
- camera.capture() is used to capture the still pictures. Provide Path (where to save) and image name as parameter.
- camera.start_preview() is used to start live display of the camera’s input.

5. Record Videos with Raspberry Pi and Pi Camera

Hardware Requirement:

- Raspberry Pi
- Pi Camera

Connection:



Step 1: Write a Python Script to record video and save them in folder.

```
from time import sleep from picamera import PiCamera

camera=PiCamera() camera.resolution=(1280,720)
camera.start_preview()
camera.start_recording('/home/pi/Videos/video.h264') sleep(10)
camera.stop_recording()
camera.stop_preview()
```

To play the video, need to open a terminal window. Type the command omxplayer videonme.h264

And Press Enter to play the video.

Methods used:

- start_recording() is used to start the recording of the videos. Provide Path (where to save) and video name as parameter. Raspberry Pi' video supports .ht64 as extension.
- stop_recording() is used to stop the recording.

6. Raspberry Pi using Telegram

Hardware Requirements:

- Raspberry Pi Model A/B/B+
- LED
- Breadboard
- Jumper Wires

Steps:-

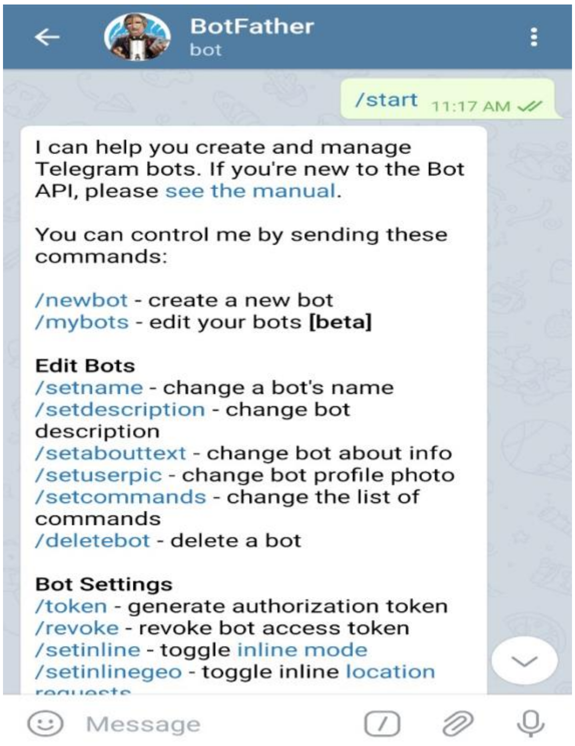
1. Connect LED with Raspberry Pi's GPIO Pins.

LED terminal	**Pin Number**	**GPIO Number**
+ve Terminal	Pin 31	GPIO 6
-ve Terminal	Pin 9	GND

2. Install Telegram App in Mobile. Follow process to obtain access token:

a) Open Telegram. Request Bot Father to create a new Bot

b) Search “Bot Father” and Click on Start.
c) Create new bot using /newbot
d) Provide a name for your bot (E.g. raspberry pi)
e) Then, provide username for your bot (E.g. RPI_Tele_Bot)
f) After this process the BotFather will give you a Token for access.



/start 11:17 AM

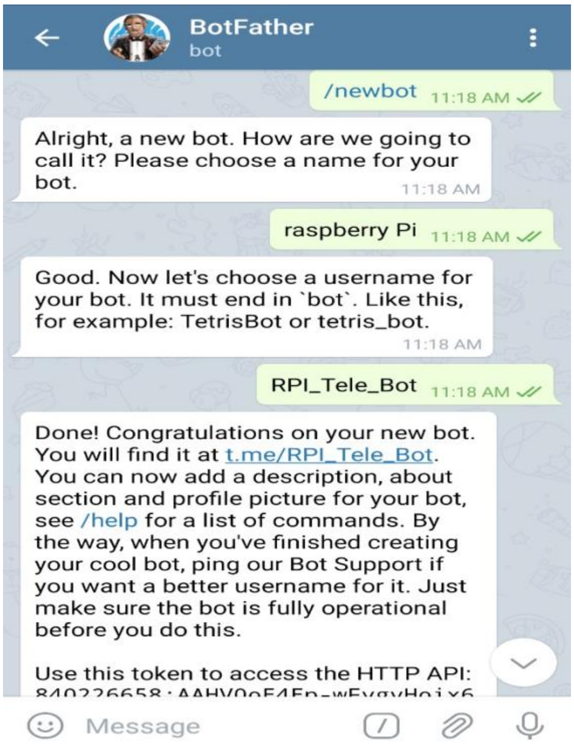
I can help you create and manage Telegram bots. If you're new to the Bot API, please see the manual.

You can control me by sending these commands:

/newbot - create a new bot
/mybots - edit your bots **[beta]**

Edit Bots
/setname - change a bot's name
/setdescription - change bot description
/setabouttext - change bot about info
/setuserpic - change bot profile photo
/setcommands - change the list of commands
/deletebot - delete a bot

Bot Settings
/token - generate authorization token
/revoke - revoke bot access token
/setinline - toggle inline mode
/setinlinegeo - toggle inline location requests



/newbot 11:18 AM

Alright, a new bot. How are we going to call it? Please choose a name for your bot. 11:18 AM

raspberry Pi 11:18 AM

Good. Now let's choose a username for your bot. It must end in `bot`. Like this, for example: TetrisBot or tetris_bot. 11:18 AM

RPI_Tele_Bot 11:18 AM

Done! Congratulations on your new bot. You will find it at t.me/RPI_Tele_Bot. You can now add a description, about section and profile picture for your bot, see /help for a list of commands. By the way, when you've finished creating your cool bot, ping our Bot Support if you want a better username for it. Just make sure the bot is fully operational before you do this.

Use this token to access the HTTP API:
840226658:AAHV0oF4Fn-wFvgvHoix6

3. Install Telegram Bot on Raspberry Pi sudo apt-get install python-pip

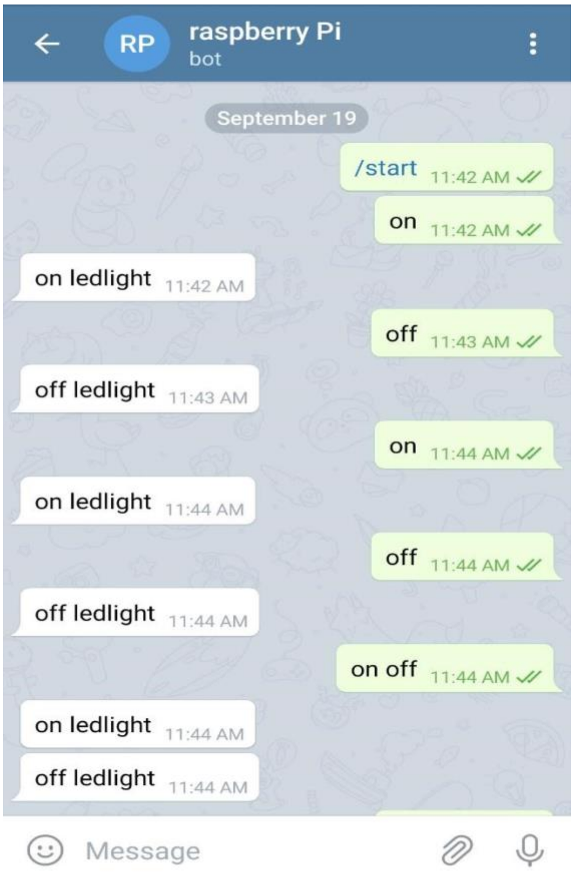
sudo pip install telepot

4. Write Python Script to blink LED with Telegram Bot (E.g. tele.py)

```
    import time,datetime import RPi.GPIO as GPIO import telepot
from telepot.loop import MessageLoop

led=6

now=datetime.datetime.now() GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

GPIO.setup(led,GPIO.OUT)
GPIO.output(led,0)

def action(msg):
char_id=msg['chat']['id']
cmd=msg['text']     print('Received:
%s' %cmd)
```

```
    if 'on' in cmd:         message
= "on "
message=message+"led"
GPIO.output(led,1)
message=message+"light"
        telegram_bot.sendMessage(char_id,message)

    if 'off' in cmd:
        message = "off "
message=message+"led"
GPIO.output(led,0)
message=message+"light"
        telegram_bot.sendMessage(char_id,message)

telegram_bot=telepot.Bot(' *** ACCESS KEY TOKEN *** ')
print(telegram_bot.getMe())
MessageLoop(telegram_bot,action).run_as_thread()
print('Up and Running..')

while 1:
  time.sleep(10)
```

5. Now, run the example code as follows: python
tele.py



/start 11:42 AM

on 11:42 AM

on ledlight 11:42 AM

off 11:43 AM

off ledlight 11:43 AM

on 11:44 AM

on ledlight 11:44 AM

off 11:44 AM

off ledlight 11:44 AM

on off 11:44 AM

on ledlight 11:44 AM

off ledlight 11:44 AM

7. Pi based Oscilloscope

Hardware Requirements

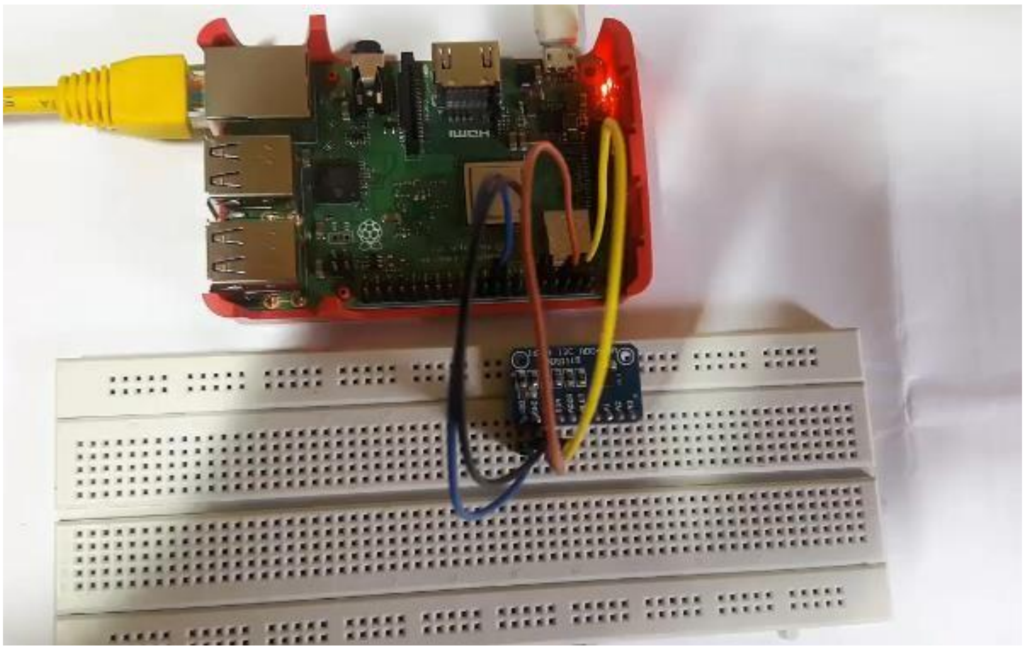
- Raspberry Pi Model A/B/B+
- ADS1115 ADC
- Breadboard
- Jumper Wires

Software Requirements

1. Raspbian Stretch OS
2. Adafruit module for interfacing with the ADS1115 ADC chip
3. Python Module matplotlib used for data visualization

Connect ADC with Raspberry Pi’s GPIO

ADS1115 ADC	**Pin Number**	**GPIO Number**
VDD	Pin 17	3.3v
GND	Pin 9	GND
SCL	Pin 5	GPIO 3
SDA	Pin 3	GPIO 2



1. Update and Upgrade Raspberry Pi

sudo apt-get update sudo apt-get upgrade

2. Enable Raspberry Pi I2C interface

sudo raspi-config

3. Install the Adafruit ADS1115

library for ADC sudo apt-get install build-essential python-dev python-smbus git git clone https://github.com/adafruit/Adafruit_Python_ADS1x15.git sudo python setup.py install

5. Install Matplotlib

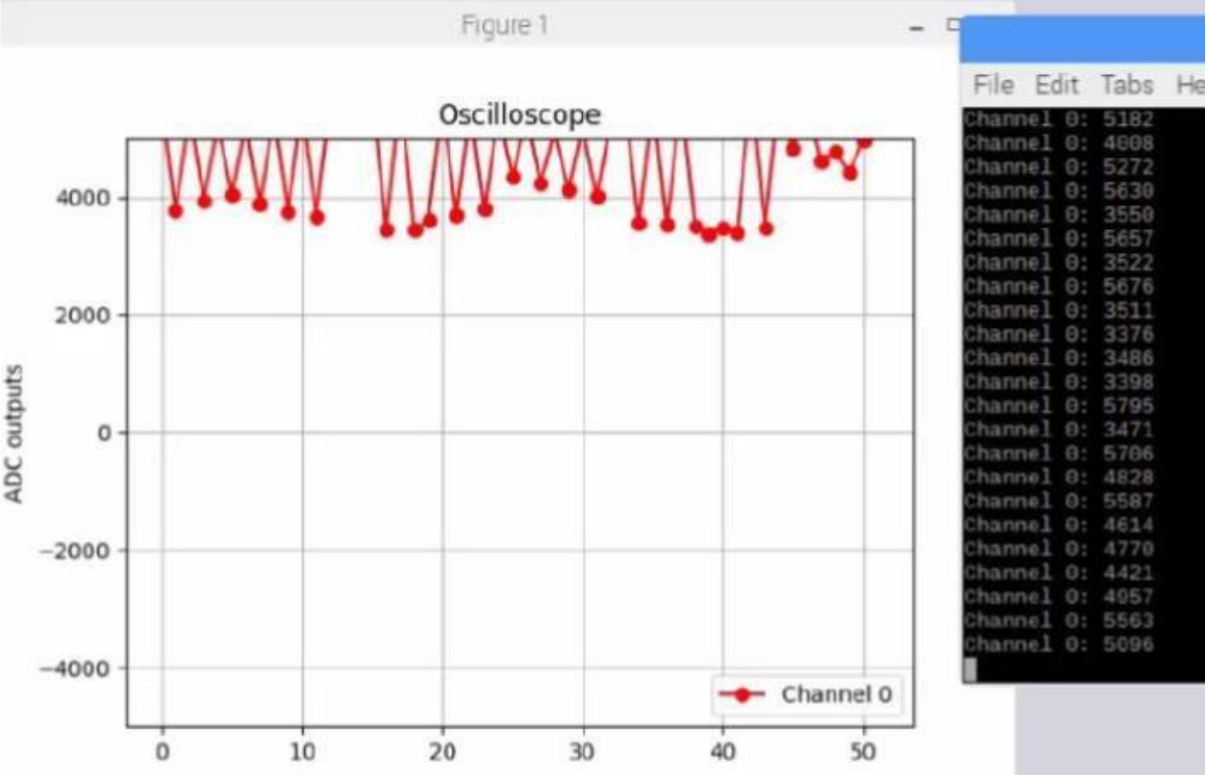
sudo apt-get install python-matplotlib

6. Write python code (Eg- osci.py) import matplotlib.pyplot as plt

```
from matplotlib.animation import FuncAnimation import
Adafruit_ADC1x15

# Create an ADS1115 ADC (16-bit) instance.
adc= Adafruit_ADC1x15.ADS1115()

GAIN=1
val=[]

# Start continuous ADC conversions on channel 0 using the previous gain value.
adc.start_adc(0,gain=GAIN) print(“Reading ADC1x15 Channel 0”)

ax=plt.subplots() ax.set_ylim(-5000,5000)
ax.set_ylabel(“ADC Output”)
ax.set_title(“Oscilloscope”) ax.grid(True)


line = ax.plot([], ‘ro-’, lable=’Channel 0’) ax.legend(loc=’lower
right')

def update(cnt):
        # Read the last ADC conversion value and print it
        out.  value = adc.get_last_result()  print('Channel 0:
        {0}'.format(value))
        # Set new data to line
        line.set_data(list(range(len(val))),
        val)  ax.relim()  ax.autoscale_view()
        #Store values for later
        val.append(int(value))  if(cnt>50):
                val.pop(0)
```

ani = FuncAnimation(fig, update, interval=500) plt.show()

7. Save the code and run using python osci.py



```
Channel 0: 5182
Channel 0: 4808
Channel 0: 5272
Channel 0: 5630
Channel 0: 3550
Channel 0: 5657
Channel 0: 3522
Channel 0: 5676
Channel 0: 3511
Channel 0: 3376
Channel 0: 3486
Channel 0: 3398
Channel 0: 5795
Channel 0: 3471
Channel 0: 5706
Channel 0: 4828
Channel 0: 5587
Channel 0: 4614
Channel 0: 4770
Channel 0: 4421
Channel 0: 4957
Channel 0: 5563
Channel 0: 5096
```