Fingerprint Sensor interfacing with Raspberry Pi

In this lesson we will learn to interface fingerprint sensor with raspberry pi. For this we willing be using the USB port of raspberry pi.

Hardware Guide:

For completing this lesson, you will require the following things along with your initial raspberry pi setup

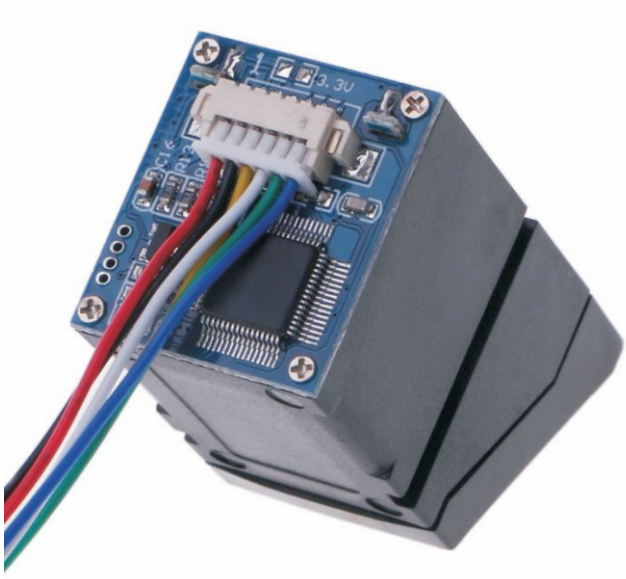
1. Fingerprint Sensor
2. USB to TTL/UART converter
3. Connecting wires
4. Push Buttons
5. 16x2 LCD
6. LED
7. Breadboard

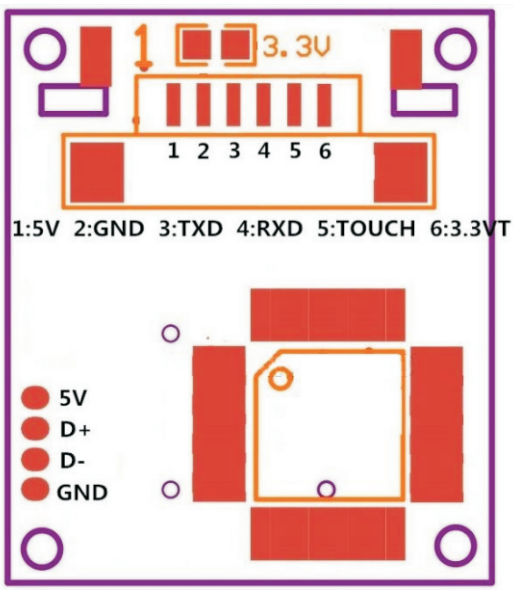
Fingerprint Sensor:

It is an intelligent module which can freely get fingerprint, image processing, verified fingerprint, search and storage, and it can work normally without upper monitor's participatory management.

Fingerprint processing includes two parts: fingerprint enrolment and fingerprint matching (the matching can be 1:1 or 1: N). Enrolling fingerprint, user needs to enter the finger 2-4 times for every one finger, process finger images with many times, store generate templates on module. When fingerprint matching, enrol and process verified fingerprint image and then matching with

PIn 1 = +5V, Pin2 = GND, PIN 3 = TXD , Pin 4 = RXD

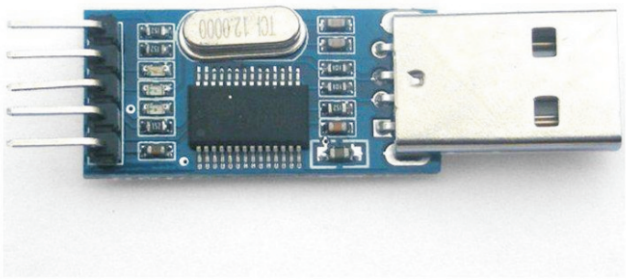




module (if match with appoint templates on the module, named fingerprint verification, for 1:1 matching method; if match with many templates on the module, named fingerprint search method also named 1: N) system will return the matching result, success or failure.

USB to TTL converter:

Here we have used a fingerprint module which works on UART. So here we have interfaced this fingerprint module with Raspberry Pi using a USB to Serial converter.

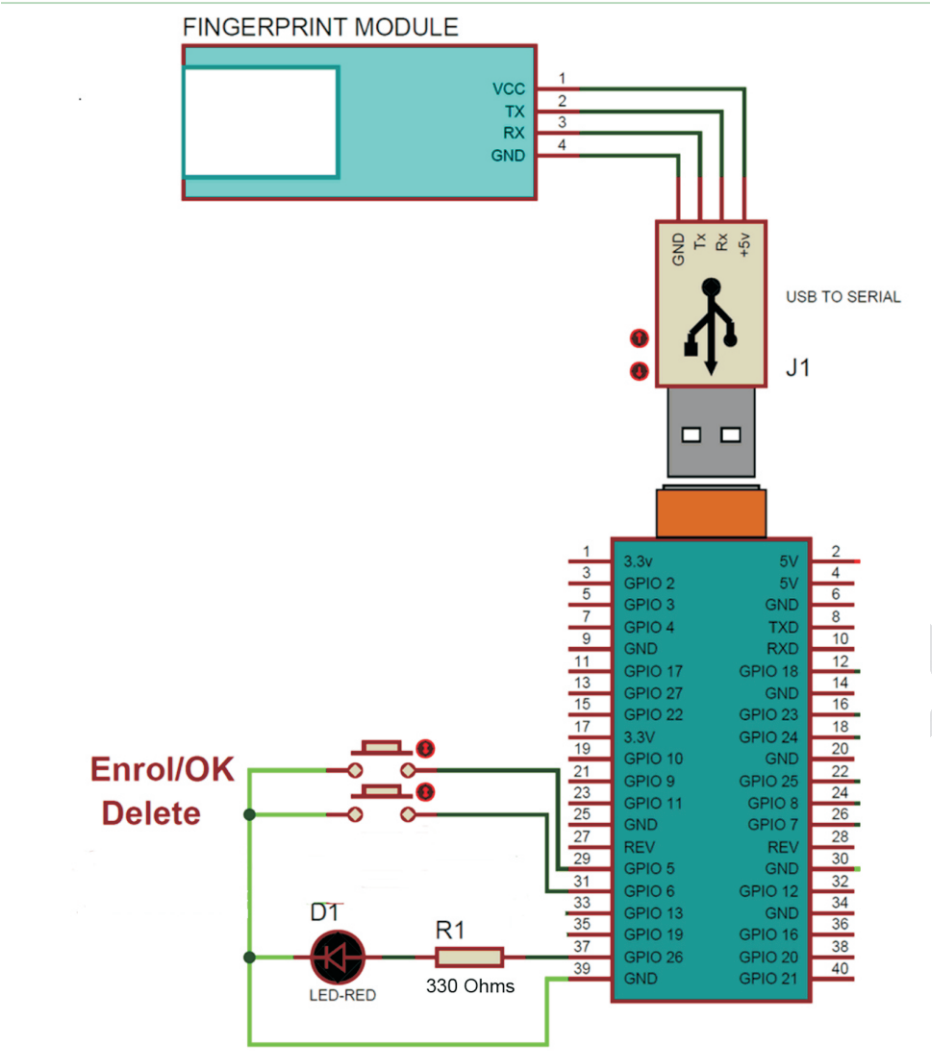


Wiring up your circuit:

In this Raspberry Pi Finger Print sensor interfacing project, we have used a 2 push buttons: one for enrolling the new finger print, one for deleting the already fed finger prints positions.
A LED is used for indication that fingerprint sensor is ready to take finger for matching. Here we have used a fingerprint module which works on UART. So here we have interfaced this fingerprint module with Raspberry Pi using a USB to Serial converter.

So, first of all, we need to make the all the required connection as shown below.
Connections are simple, we have just connected fingerprint module to Raspberry Pi USB port by using USB to Serial converter.

1. Connect the VCC Pin of Finger Print Module (Pin 1) Red Wire to 5V Pin of USB to TTL converter
2. Connect the GND Pin of Finger Print Module (Pin 2) Black Wire to GND Pin of USB to TTL converter
3. Connect the TXD Pin of Finger Print Module (Pin 3) Yellow Wire to RXD Pin of USB to TTL converter
4. Connect the RXD Pin of Finger Print Module (Pin 4) White Wire to TXD Pin of USB to TTL converter
5. Lastly connect the USB to TTL converter to USB port of Raspberry Pi.
6. Connect one push button switch to pin 29 of Raspberry Pi for Enroll finger print.
7. Connect another push button switch to pin 31 of Raspberry Pi for Deleting already fed finger print.
8. Connect LED to pin 37 of Raspberry Pi for indication that finger print is ready to take finger for matching.
9. Finally connect ground pin 39(GND) of Raspberry pi to circuit as shown in Circuit diagram.



Software Guide:

After making all the connections we need to power up Raspberry Pi and get it ready with terminal open. Now we need to install fingerprint library for Raspberry Pi in python language by following the below steps.

1. To install this library, root privileges are required. So first we enter in root by given command: sudo bash
2. Then download some required packages by using given commands:
    a. wget –O – http://apt.pm-codeworks.de/pm-codeworks.de.gpg | apt-key add –
    b. wget http://apt.pm-codeworks.de/pm-codeworks.list -P /etc/apt/sources.list.d/
3. After this, we need to update the Raspberry pi and install the downloaded finger print sensor library:
    a. sudo apt-get update
    b. sudo apt-get install python-fingerprint
    c. now exit root by typing exit
4. After installing library now, we need to check USB port on which your finger print sensor is connected, by using given the command: ls /dev/ttyUSB* (or lsusb)

Now in your python code replace the USB port number with the one you got on the screen after executing the command in step4.

Code:

```
import time
from pyfingerprint.pyfingerprint import PyFingerprint
import RPi.GPIO as gpio


enrol=5
delet=6
led=26

HIGH=1
LOW=0

gpio.setwarnings(False)
gpio.setmode(gpio.BCM)

gpio.setup(enrol, gpio.IN, pull_up_down=gpio.PUD_UP)
gpio.setup(delet, gpio.IN, pull_up_down=gpio.PUD_UP)
gpio.setup(led, gpio.OUT)

try:
    f = PyFingerprint('/dev/ttyUSB0', 57600, 0xFFFFFFFF, 0x00000000)
    if ( f.verifyPassword() == False ):
        raise ValueError('The given fingerprint sensor password is wrong!')
except Exception as e:
    print('Exception message: ' + str(e))
    exit(1)

print('Currently used finger templates: ' + str(f.getTemplateCount()) +'/'+ str(f.getStorageCapacity()))


def enrollFinger():
    print('Waiting for finger...')
    while ( f.readImage() == False ):
        pass
    f.convertImage(0x01)
    result = f.searchTemplate()
    positionNumber = result[0]
    if ( positionNumber >= 0 ):
        print('Finger already exists at position #' + str(positionNumber))
        time.sleep(2)
        return
    print('Remove finger...')
    time.sleep(2)


    print('Waiting for same finger again...')
    while ( f.readImage() == False ):
        pass
    f.convertImage(0x02)
    if ( f.compareCharacteristics() == 0 ):
        print ("Fingers do not match")
        time.sleep(2)
        return
```

```
    f.createTemplate()
    positionNumber = f.storeTemplate()
    print('Finger enrolled successfully!')

    print('New template position #' + str(positionNumber))
    time.sleep(2)

def searchFinger():
    try:
        print('Waiting for finger...')
        while( f.readImage() == False ):
            #pass
            time.sleep(.5)
            return
        f.convertImage(0x01)
        result = f.searchTemplate()
        positionNumber = result[0]
        accuracyScore = result[1]
        if positionNumber == -1 :
            print('No match found!')
            time.sleep(2)
            return
        else:
            print('Found finger at position #' + str(positionNumber))
            time.sleep(2)

    except Exception as e:
        print('Operation failed!')
        print('Exception message: ' + str(e))
        exit(1)

def deleteFinger():
    positionNumber = 0
    count=0
    while gpio.input(delet) == True:   # here delet key means ok
      positionNumber = input('Please enter the template position you want to delete: ')
      positionNumber = int(positionNumber)
      if f.deleteTemplate(positionNumber) == True :
        print('Template deleted!')
        time.sleep(1)
        print('Currently used finger templates: ' + str(f.getTemplateCount()) +'/'+ str(f.getStorageCapacity()))
        time.sleep(1)
        return


print ("Edkits Electronics Welcomes You ")
time.sleep(3)
flag=0
```

```
while 1:
    gpio.output(led, HIGH)

    if gpio.input(enrol) == 0:
        gpio.output(led, LOW)
        enrollFinger()
    elif gpio.input(delet) == 0:
        gpio.output(led, LOW)
        while gpio.input(delet) == 0:
            time.sleep(0.1)
        deleteFinger()
    else:
        searchFinger()
```