Interfacing GPS module with Raspberry Pi
Find Your Current Location on Google Maps

Installation Manual

- **GPS** stands for Global Positioning System and used to detect the Latitude and Longitude of any location on the Earth, with exact UTC time (Universal Time Coordinated).
- GPS module is the main component in our vehicle tracking system.
- This device receives the coordinates from the satellite for each and every second, with time and date.
- **GPS module** sends the data related to tracking position in real time, and it sends so many data in NMEA format.
- NMEA format consists of sentence starts from **$GPGGA,** the coordinates, time and other useful information.
- **GPGGA** is referred to **Global Positioning System Fix Data**.
- GPGGA string contains following co-ordinates separated by commas.

Sr No	Identifier	Description	Sr No	Identifier	Description
1	$GPGGA	Global Positioning system fix data	7	FQ	Fix Quality Data
2	HHMMSS.SSS	Time in hour minute seconds and milliseconds format.	8	NOS	No. of Satellites being Used
3	Latitude	Latitude (Coordinate)	9	HPD	Horizontal Dilution of Precision
4	N	Direction N=North, S=South	10	Altitude	Altitude from sea level
5	Longitude	Longitude(Coordinate)	11	M	Meter
6	E	Direction E= East, W=West	12	Height	Height
			13	Checksum	Checksum Data

Hardware Requirements	Software Requirements
• Raspberry Pi Model B/B+ , SD Card • Ethernet Cable / Wi-Fi • Power Supply to Pi • Neo 6m v2 GPS Module • 4 Jumper Wires (Female to Female)	• Raspbian Stretch OS

Neo 6m v2 GPS Module

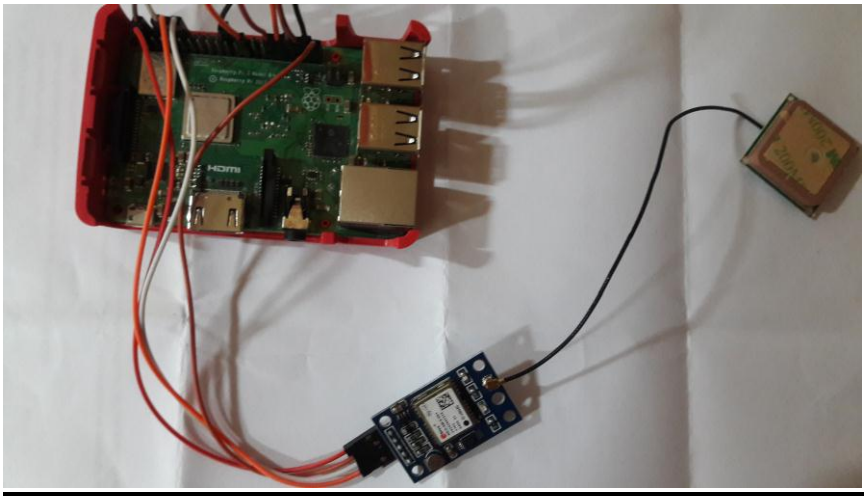
- This board features the u-blox NEO-6M GPS module with antenna and built-in EEPROM. This is compatible with various flight controller boards designed to work with a GPS module.
- EEPROM is used for saving the configuration data when powered off.
- Power Supply Range: 3 V to 5 V
- Default Baud Rate: 9600 bps

Connect GPS Module to RPi

Connection is very simple. Requires 4 Female to Female Jumper Wires

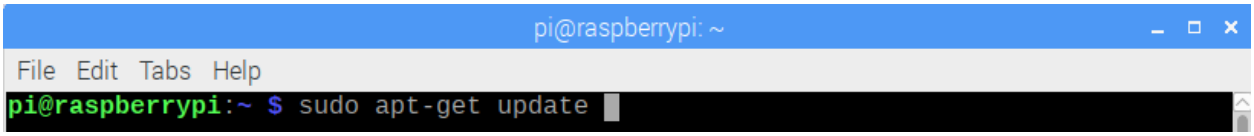
Neo 6m V2 GPS Board Pin	Details	Raspberry Pi Physical Pin	Raspberry Pi Function
VCC	Power	Pin 1	3.3V Power
GND	Common Ground	Pin 39	GND
TXD	Data Output	Pin 10	(UART_RXD0) GPIO15
RXD	Data Input	Pin 8	(UART_TXD0) GPIO14

Connect GPS module to Raspberry Pi's GPIO Pins by using Female to Female Jumper wires

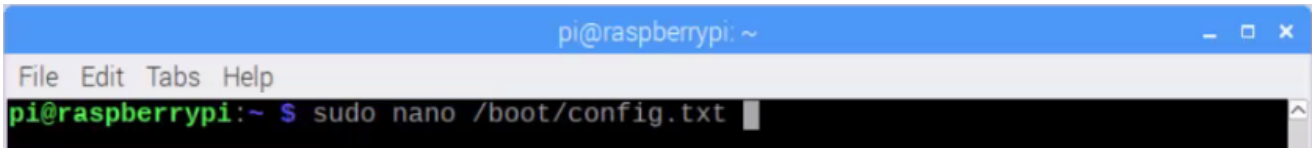


- Once the connection are made, Power on Raspberry Pi and GPS module's red led also gets on.
- Make sure that the GPS module is kept in a place where the GPS has a clear view to the sky or at least near a window for reliable signal integrity.
- We may find problem receiving signals in indoor area, the GPS signal can get blocked by metal, deep forest, mountains etc.

Step 1: Update Raspberry Pi



Step 2: edit the /boot/config.txt file

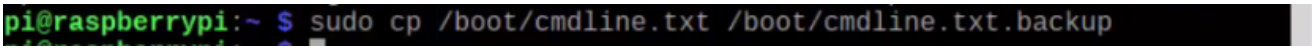


At the bottom of this file, add the above lines

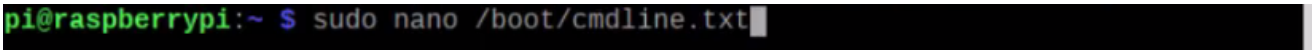
The ***dtoverlay=pi3-disable-bt*** disconnects the bluetooth from the *ttyAMA0*, this is to allow us access to use the full UART power available via *ttyAMAO* instead of the mini UART ttyS0.

Save with Ctrl+X, yes and press Enter.

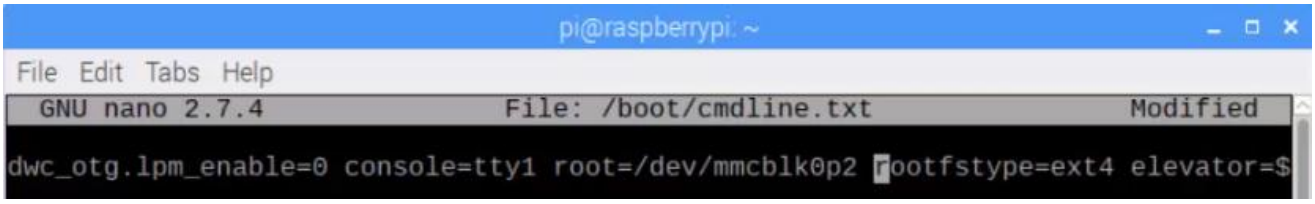
Step 3: Before proceeding, make a copy of cmdline.txt file



Now, edit file cmdline.txt



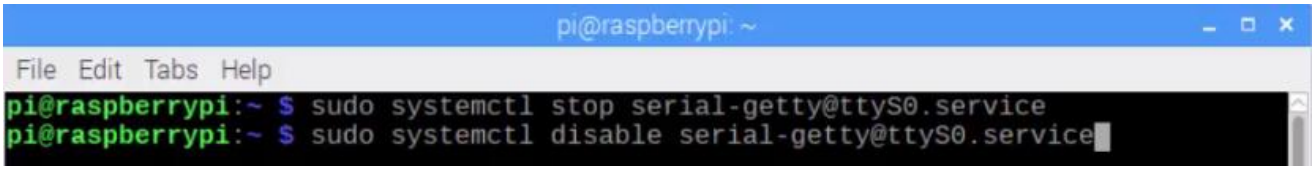
Remove console=serial0,115200 **and Modify** root=/dev/mmcblk0p2



Save with Ctrl+X, yes and press Enter.

Step 4: Reboot Raspberry Pi using the command *sudo reboot*

Step 5: Stop and disable the Pi's serial ttyS0 service

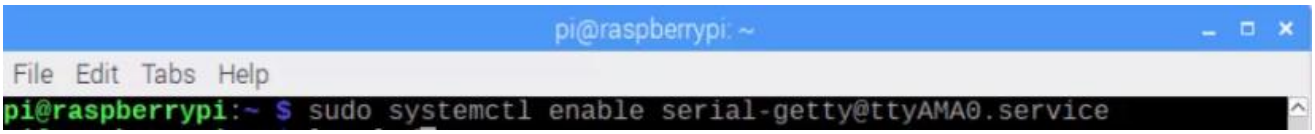


The following commands can be used to enable it again if needed

```
sudo systemctl enable serial-getty@ttyS0.service

sudo systemctl start serial-getty@ttyS0.service
```

Step 6: Reboot Raspberry Pi using the command *sudo reboot*

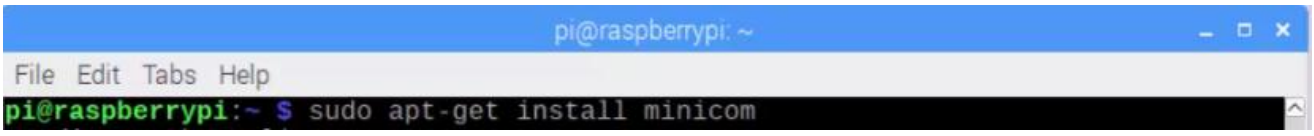
Step 7: Now, Enable the ttyAMA0 service



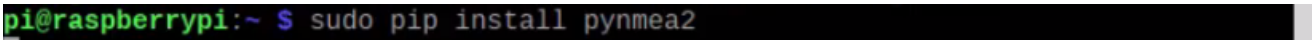
Verify it using *ls –l /dev* command

Step 8: Install minicom and pynmea2

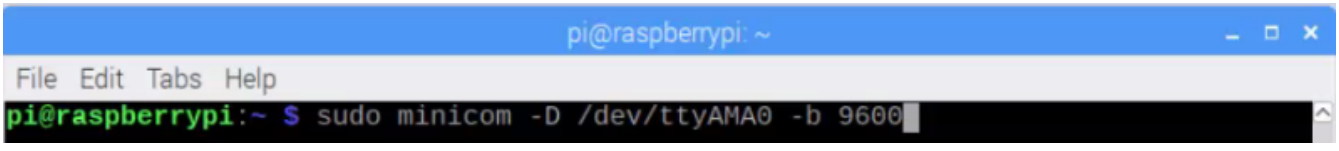
Install minicom package which is used to connect to the GPS module and make sense of the data.



Install pynmea2 library which is used to parse the received data.

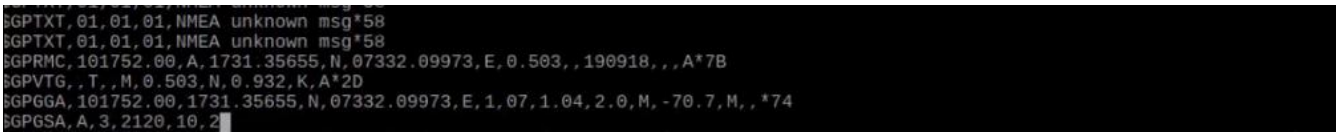


Step 9: Use minicom command to test our GPS module is working fine.



9600 represents the baud rate at which the GPS module communicates.

Here, we can see NMEA sentences .**NMEA format** consist several sentences, in which we only need one sentence. This sentence starts from **$GPGGA** and contains the coordinates

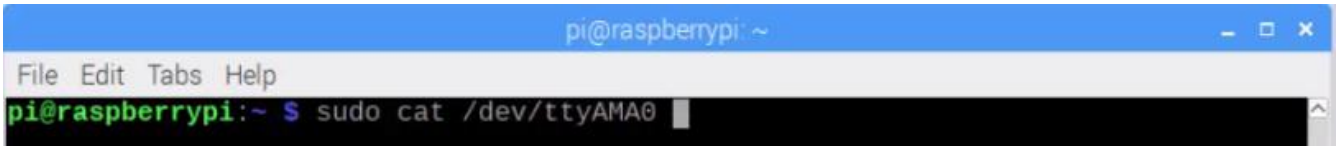


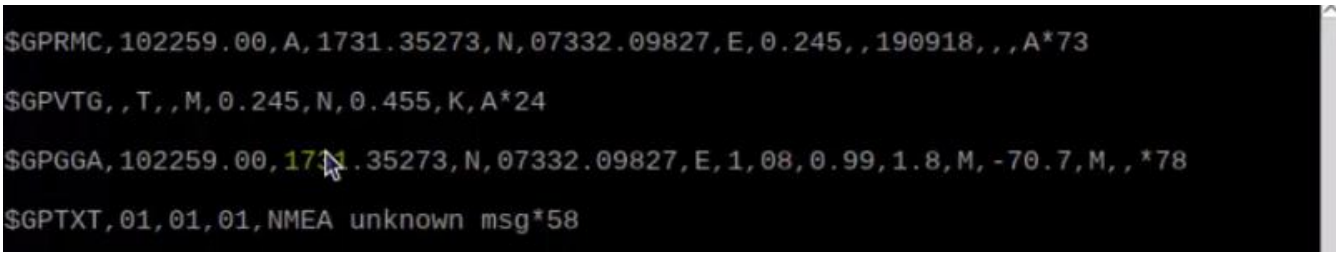
Sometimes we received sentences which contains **unknown msg*58,** but this is not an error, actually it may takes some time to track your GPS module. (Even for the first time more than 20-30 minutes.)

I suggest keep your GPS module's antenna in open space (e.g. near the window)

To exit from above window, Press Ctrl+A, and Press x and Enter Key.

Step 10: The above same test can also be done using cat command





This sentence gives you Latitude found after two commas and Longitude found after four commas.

Step 11: Write a Python script to receive the serial data from the GPS module.

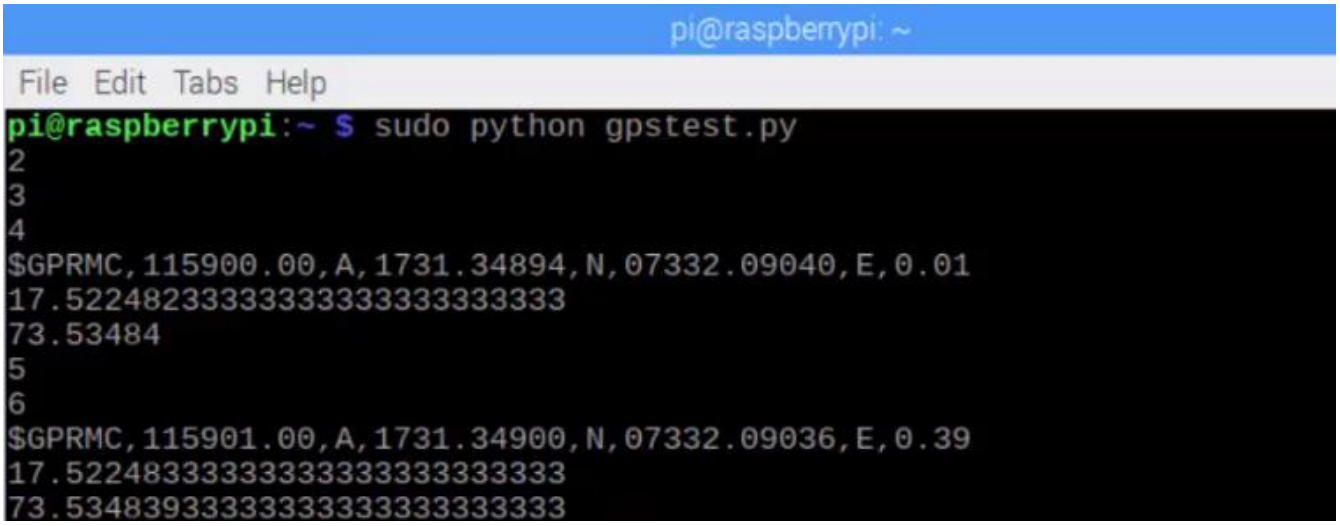
Python Script (gpstest.py)

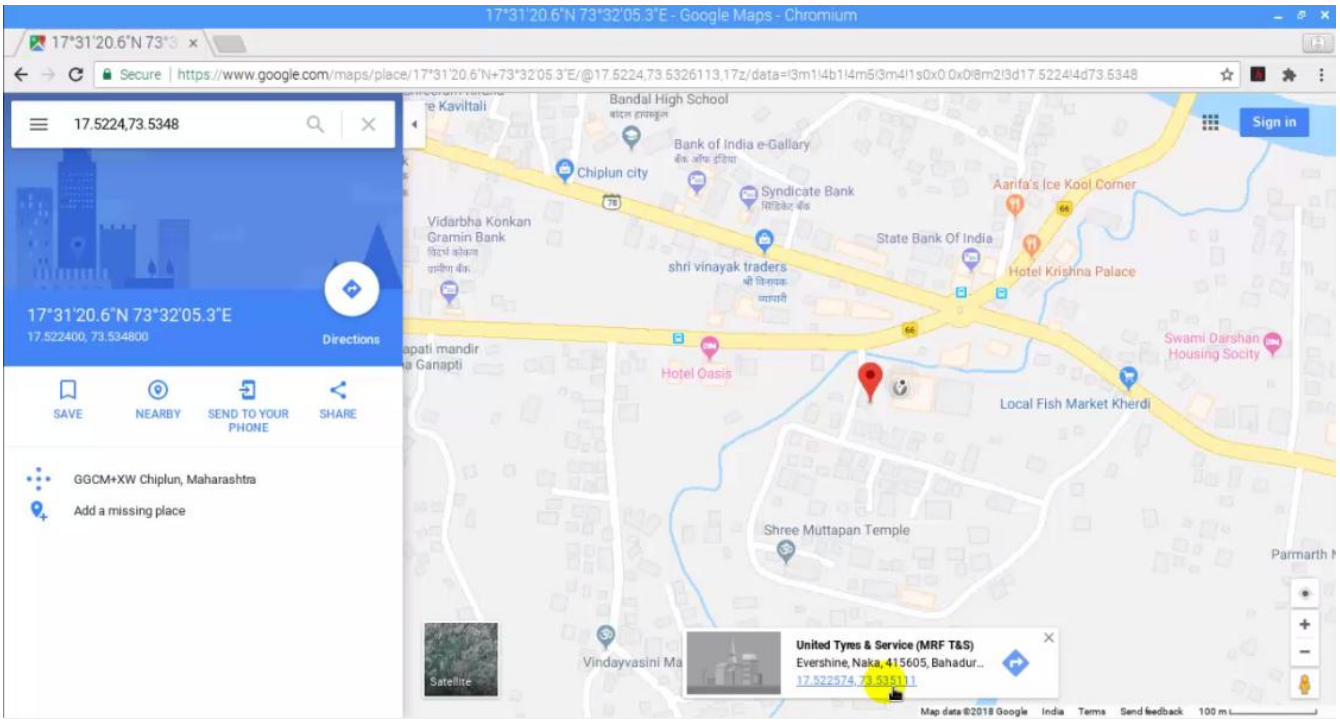
```
import serial
#import RPi.GPIO as GPIO
import os, time
from decimal import *

delay = 1

#GPIO.setmode(GPIO.BOARD)

def find(str, ch):
    for i, ltr in enumerate(str):
        if ltr == ch:
            yield i

port = serial.Serial("/dev/ttyAMA0", baudrate=9600, timeout=1)
```

```
cd=1
try:
   while cd <= 50:
      ck=0
      fd="
      while ck <= 50:
         rcv = port.read(10)

         fd=fd+rcv
         ck=ck+1

      if '$GPRMC' in fd:
         ps=fd.find('$GPRMC')
         dif=len(fd)-ps

         if dif > 50:
            data=fd[ps:(ps+50)]
            print(data)
            p=list(find(data, ","))
            lat=data[(p[2]+1):p[3]]
            lon=data[(p[4]+1):p[5]]

            s1=lat[2:len(lat)]
            s1=Decimal(s1)
            s1=s1/60
            s11=int(lat[0:2])
            s1=s11+s1

            s2=lon[3:len(lon)]
            s2=Decimal(s2)
            s2=s2/60
            s22=int(lon[0:3])
            s2=s22+s2

            print("Latitude:",s1)
            print("Longitude:",s2)

      cd=cd+1
      print(cd)
except KeyboardInterrupt:
   print("Thank You")
```

- Run the above script on terminal using **sudo python filename.py command.**
- So this code helps to locate your Pi. Once the Pi is on with this code, it will receive the location coordinates from the GPS and the same can viewed in terminal window.
- To plot our location on Google map, we need to call URL link for **Google map.** We can use following link for opening Google map with our extracted longitude and latitude coordinates,

  http://maps.google.com/?q=<latitude>,<longitude>





That's all!!!

Thank you….