MOBILE AS GPS

V2X Graduation Project Summary

By: Mahmoud Mohamed Kamal
https://www.linkedin.com/in/mahmoudfierro98/

UART Configuration

Enable UART in the Raspberry Pi, Open the Raspberry Pi terminal and insert the following commands.

Step 1: Open the config.txt file in the nano editor using the following command.

sudo nano /boot/config.txt

Add the following lines to the end of the file.

# Enable UART
enable_uart=1

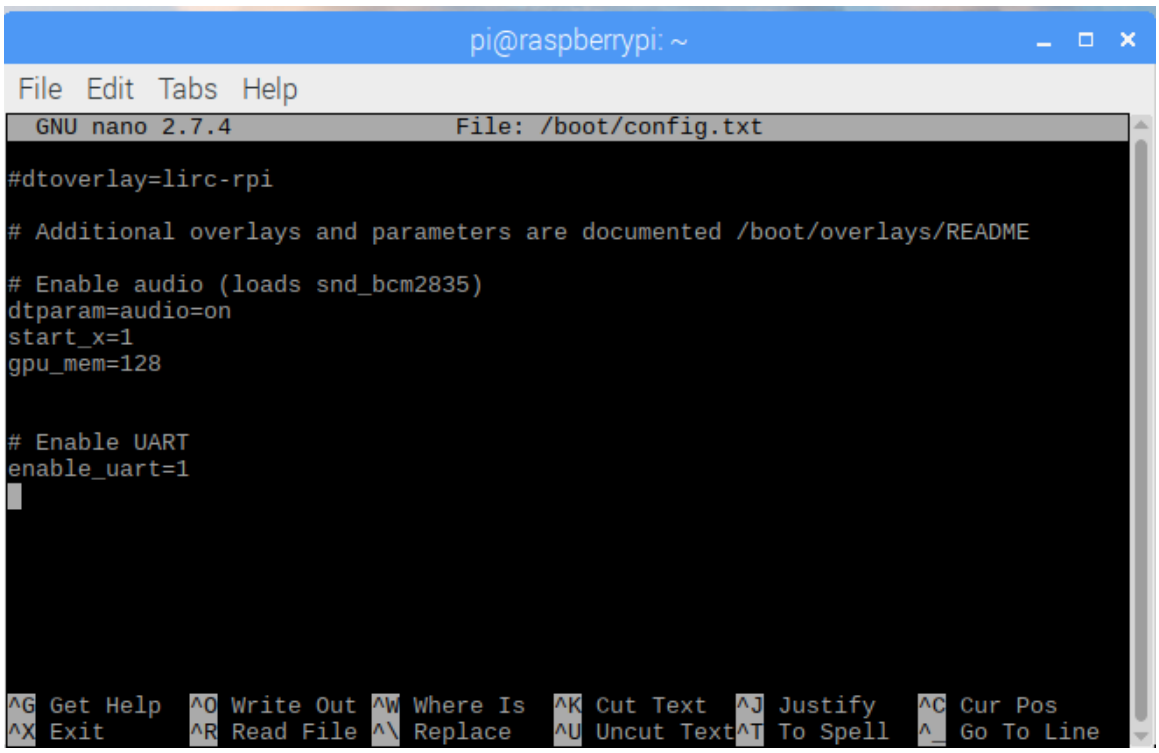

Figure 01: UART Configuration Step 1.

Then press Ctrl+x and press Y to save the file and close it.

Step 2: Disconnect the serial communication between the Raspberry Pi and the Bluetooth module.

sudo systemctl disable serial-getty@ttyS0.service

Step 3: Open cmdline.txt in nano editor.

Delete the “console=serial0,115200” line and save the file.

sudo nano /boot/cmdline.txt

Step 4: Reboot your Raspberry Pi.

sudo reboot

Step 5: Install python-serial.

sudo apt-get install python-serial

sudo apt-get install python3-serial

Step 5: Use python to receive data from Microcontrollers.

UART.py *

```
1  import serial
2
3  ser = serial.Serial("/dev/ttyS0",baudrate=9600,timeout=0.5)
4  ser.write('Connected\r\n')
5
6  while True:
7      data = ser.readline()
8      print(data)
```

Bluetooth Configuration

We used it for GPS.

Enable Bluetooth in the Raspberry Pi, Open the Raspberry Pi terminal and insert the following commands.

Step 1: Add the SP profile to the Pi. Edit this file:

sudo nano /etc/systemd/system/dbus-org.bluez.service

Step 2: Add the compatibility flag, ' -C', at the end of the 'ExecStart=' line. Add a new line after that to add the SP profile. The two lines should look like this:

```
ExecStart=/usr/lib/bluetooth/bluetoothd -C
ExecStartPost=/usr/bin/sdptool add SP
NotifyAccess=main
```

Figure 02: Bluetooth Configuration Step 2.

Step 3: Save the file and reboot.

Sudo reboot

Step 4: Pair and trust your Pi and phone with bluetoothctl or with GUI.

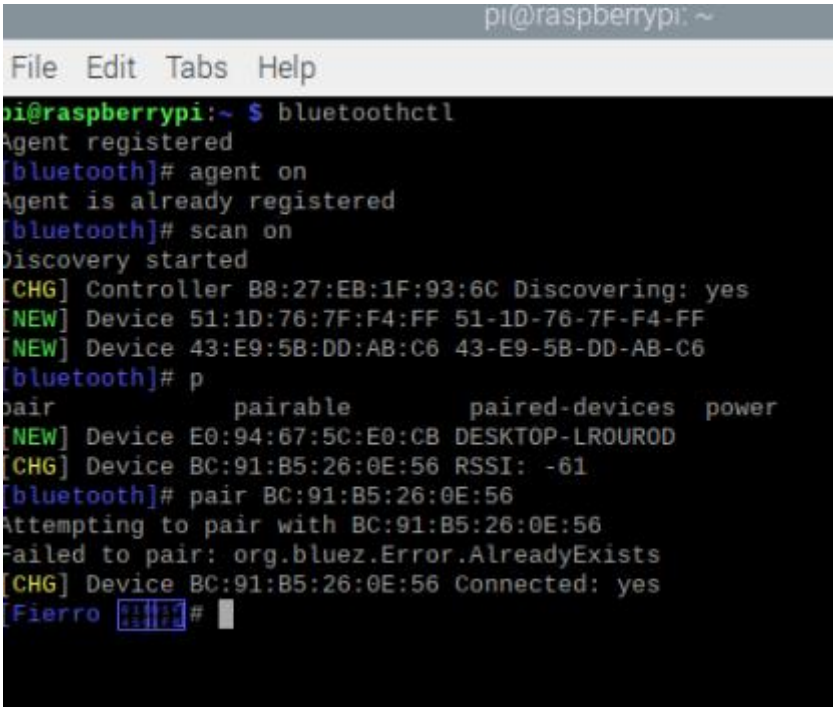

Figure 03: Bluetooth Configuration Step 4 - Terminal.

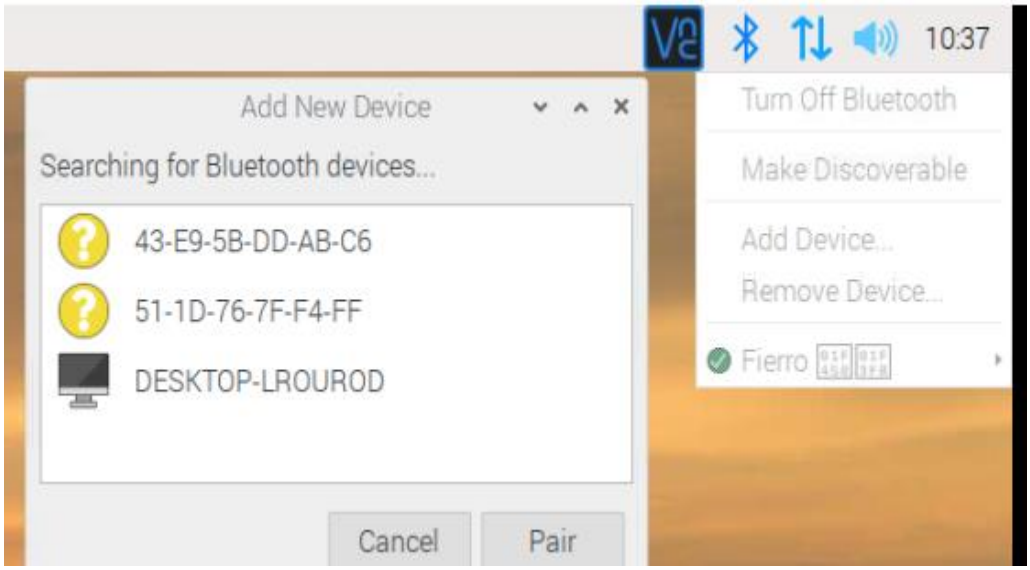

Figure 04: Bluetooth Configuration Step 4 - GUI.

Share GPS via Bluetooth:

Step 1: Pair your mobile phone with Raspberry pi.

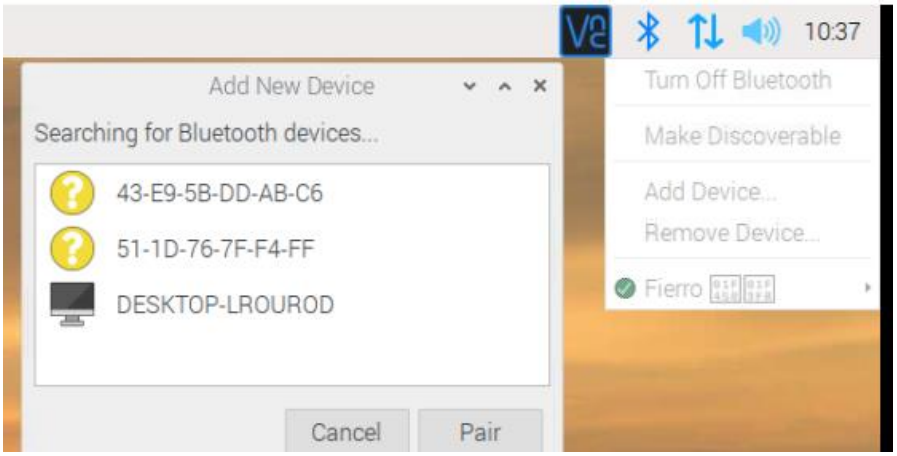

Figure 05: Share GPS via Bluetooth Step 1.

Step 2: Install Share GPS on your phone.

Figure 06: Share GPS via Bluetooth Step 2.

Step 3: Turn on Location on your phone and the app will start getting your latitude and longitude.

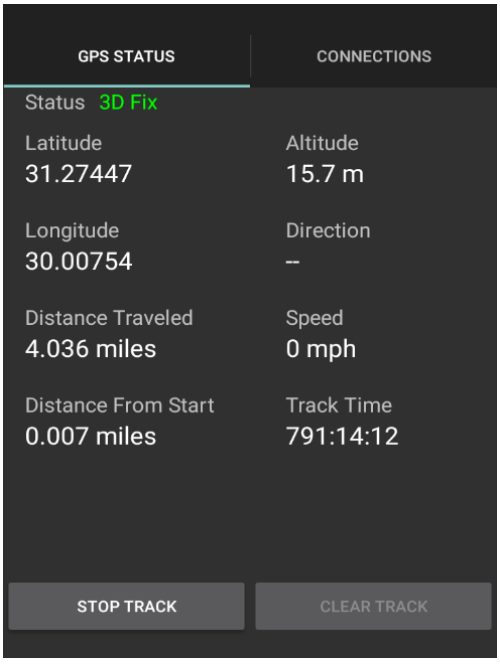

Figure 07: Share GPS via Bluetooth Step 3.

Step 4: wait serial connection by using this command:

sudo rfcomm watch hci0

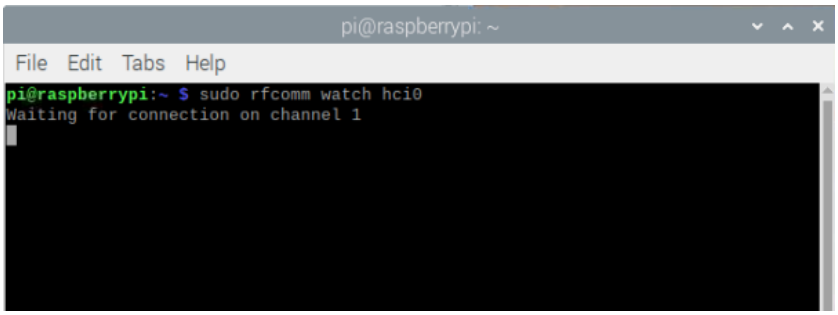

Figure 08: Share GPS via Bluetooth Step 4.

Step 5: Connect the app with raspberry pi.

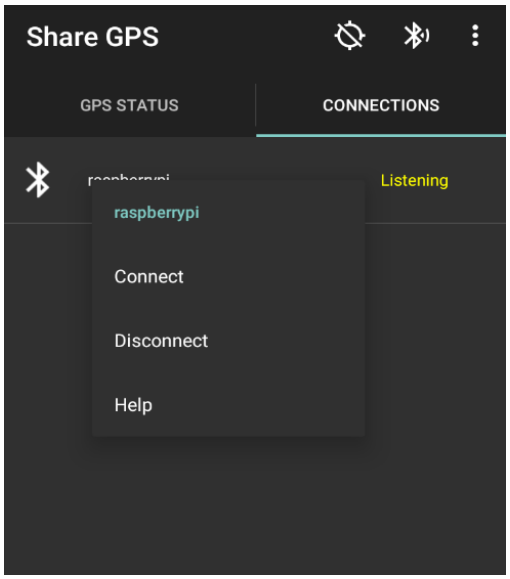

Figure 09: Share GPS via Bluetooth Step 5.

Step 6: Now app starts to connect and share location to raspberry pi.
And you can receive the reads by serial communication of /dev/rfcomm0.

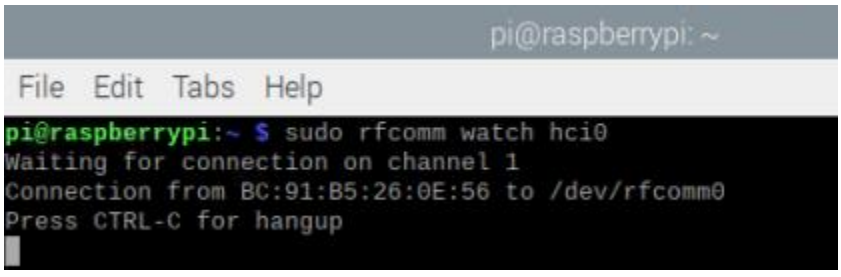

Figure 10: Share GPS via Bluetooth Step 6 a.

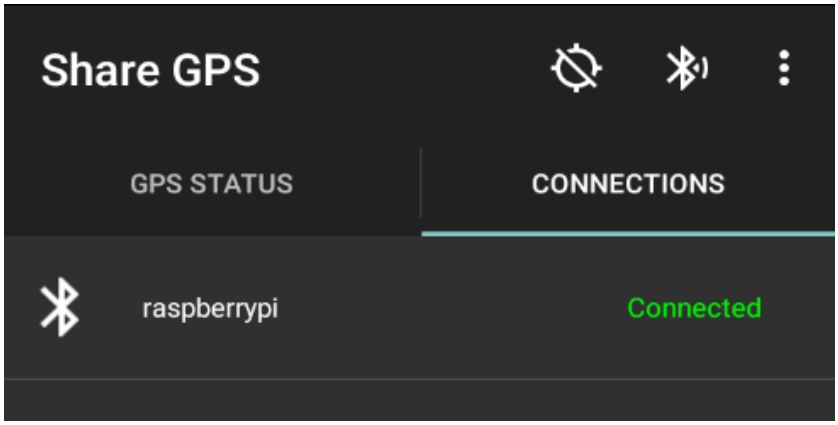

Figure 11: Share GPS via Bluetooth Step 6 b.

Step 7: By python and serial module we can get NMEA code that had your latitude and longitude.

GPS.py

```
1  import serial
2
3  ser = serial.Serial("/dev/rfcomm0",baudrate=9600,timeout=0.5)
4
5
6  while True:
7      data = ser.readline()
8      print(data)
```

```
pi@raspberrypi: ~/Desktop
File  Edit  Tabs  Help
pi@raspberrypi:~ $ cd Desktop/
pi@raspberrypi:~/Desktop $ python GPS.py
```

```
pi@raspberrypi: ~/Desktop
File  Edit  Tabs  Help
$GNGSA,A,2,05,20,29,,,,,,,,,,3.7,3.5,1.0,1*3F
$GNGSA,A,2,68,85,,,,,,,,,,,3.7,3.5,1.0,2*33
$PQGSA,A,2,11,13,,,,,,,,,,,3.7,3.5,1.0,4*3C
$GNVTG,,T,,M,0.0,N,0.0,K,A*3D
$GNRMC,102446.00,A,3116.461976,N,03000.451996,E,0.0,,130621,3.0,E,A*0F
$GNGGA,102446.00,3116.461976,N,03000.451996,E,1,07,3.5,17.6,M,20.0,M,,*4F
$GPGSV,3,1,12,05,21,046,18,20,23,046,19,29,47,054,16,10,14,187,*78
$GPGSV,3,2,12,12,02,142,,15,04,105,,16,16,319,,18,74,310,*7A
$GPGSV,3,3,12,23,40,165,,25,34,149,,26,42,309,,31,26,247,*7E
$GLGSV,3,1,09,85,17,272,15,68,21,067,19,74,55,277,,73,39,196,*63
$GLGSV,3,2,09,80,00,000,,69,19,120,,84,48,337,,83,19,045,*64
$GLGSV,3,3,09,67,01,023,*5D
```

Figure 12: Share GPS via Bluetooth Step 7.

NMEA:

At the end we received a message that called NMEA code or NMEA message.

```
$GNRMC,055557.00,A,3116.459486,N,03000.462794,E,0.0,,170721,3.0,E,A*0D
```

All we need is this line that has GMT, your latitude, and your longitude.

- GMT: `055557` This means the time of GMT 05:55:57.
- Position status: `A,` (A = data valid, V = data invalid)
- Latitude:

  `,3116.461976,N,`

  This means $31°16.461976'N = 31.27438°N$
- Longitude:

  `03000.451996,E`

  This means $30°00.451996'E = 30.00755°E$

So, the app all his works is sharing GPS of mobile with Raspberry pi and sending NMEA code with your time and position.

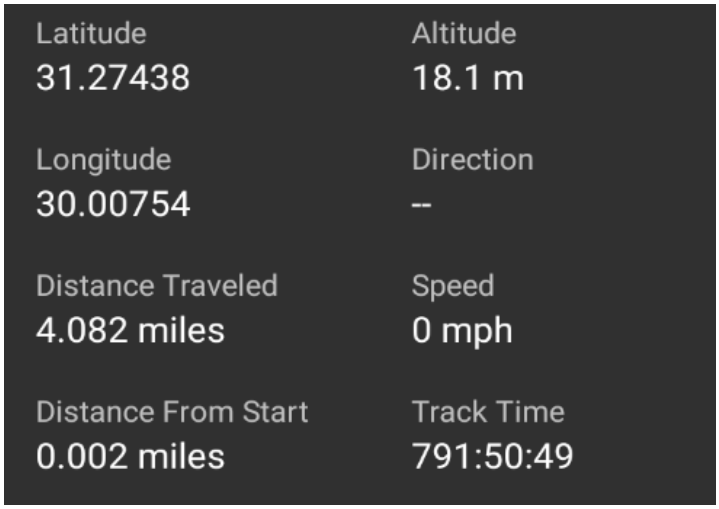

Figure 13: NMEA Checking.

Advantages

- So accurate using many stiles to get your location.
- Indoor.
- Suitable for our project.
- Using Bluetooth of Raspberry pi.

Software

Using serial module to get data from Bluetooth of mobile and the parsing NMEA code to get specific line start with “$GNRMC” sprite it to time, latitude and longitude.

We can send the data after parsing to another microcontroller through UART.

GPS.py

```
import serial

GPS  = serial.Serial("/dev/rfcomm0",baudrate=9600,timeout=0.5)
#UART = serial.Serial("/dev/ttyS0",baudrate=9600,timeout=0.5)
#UART.write("ddd")
while True:
    line = GPS.readline()
    data = line.split(",")
    ''' Parsing NMEA Code '''
    if data[0] == "$GNRMC":
        #UART.write(line)
        #UART.write("\n")
        ''' Get UTC of postion '''
        time = data[1]
        time = time.split(".")
        utc  = time[0]
        utc2 = str(int(utc[1])+2)
        hour = utc[0] + utc2
        mins = utc[2:4]
        secs = utc[4:6]
        print(line)
        print ("Time(UTC+2): "+hour+":"+mins+":"+secs)
        #UART.write("Time(UTC+2): "+hour+":"+mins+":"+secs)
        #UART.write("\n")
        ''' Check Position status (A = data valid, V = data invalid) '''
        if data[2] == "A":
            ''' Get UTC of Latitude '''
            Latitude_deg = data[3][0:2]
            Latitude_min = data[3][2::]
            Latitude_dir = data[4]
            Latitude     = Latitude_deg+Latitude_min+Latitude_dir
            print("Latitude: "+Latitude)
            #UART.write("Latitude: "+Latitude)
            #UART.write("\n")
            ''' Get UTC of Longtitude '''
            Longtitude_deg = data[5][0:3]
            Longtitude_min = data[5][3::]
            Longtitude_dir = data[6]
            Longtitude     = Longtitude_deg+Longtitude_min+Longtitude_dir
            print("Longtitude: "+Longtitude)
            # UART.write("Longtitude: "+Longtitude)
            #UART.write("\n")
        else:
            print "Not Fix - Data Invalid"
```

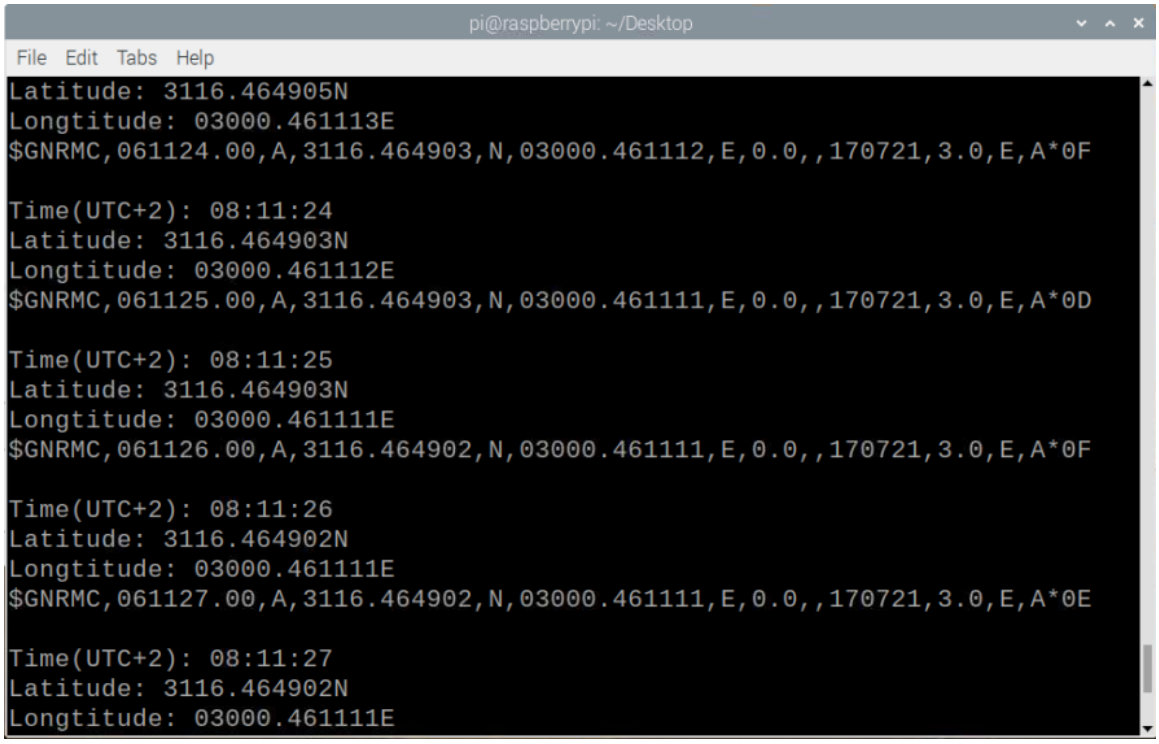

Figure 14: GPS code output.

As shown refresh data every 1 second.