**Mobil bázisállomás dokumentáció**

**Hardver:**

* Raspberry PI 4
* RPI PowerPack V1.2
* Sirius RTK GNSS Rover (F9P)
* RTK F9P GPS Antenna

**Információk:**

Operációs rendszer: Ubuntu 20.04

Használt python csomag: <https://github.com/semuconsulting/pygnssutils>

Csomag telepítési helye: home/base/.local/lib/python3.10/site-packages/pygnssutils

Egyéb hasznos információk: https://www.semuconsulting.com/pygnssutils/py-modindex.html

**Használt parancsok:**

Ezzel a parancsal tekinthető meg, pontosan melyik porton van a vevő: *ls /dev/tty\**

Pygnssutils telepítése: *python3 -m pip install --upgrade pygnssutils*

NTRIP szerver indítása CLI-ből: *gnssserver --inport "/dev/ttyACM0" --hostip 192.168.1.142 --outport 2101 --ntripmode 1 --protfilter 4 --format 2*

Az NTRIP client indítása, amely automatikusan kapcsolódik, ha létezik a mount point (ez nincsen használatban, de teszteléshez ezt használtam): *gnssntripclient -S hostip -P 2101 -M pygnssutils --user anon --password password*

Default beállítások a mountpointhoz való csatlakozáshoz:

user=anon

password=password

mountpoint=pygnssutils

Openvpn telepítése: *sudo apt-get install openvpn*

VPN-re való csatlakozás: *sudo openvpn –config \*vpn-file neve\**

Hogy a bash file ne kérjen a sudo-nál jelszót (így semminél sem fog):

*sudo visudo*

Utolsó sornak hozzáadni:

*<username> ALL=(ALL) NOPASSWD: ALL*

*pl.: dani ALL=(ALL) NOPASSWD: ALL*

Hogy a shell script magától elinduljon a rendszer indulásakor:

*crontab -e*

A filehoz hozzáadni: *@reboot /path/to/your/base\_startup.sh*

base\_startup.sh tartalma, magyarázattal:

#!/bin/bash

sleep 50

#rendszergazda jog adása a portnak

sudo chmod a+rw /dev/ttyACM0

#vpn csatlakozás

sudo openvpn --config /home/base/Desktop/vpnkey/test2.ovpn > /dev/null 2>&1 &

sleep 5

#ntrip server indítása

sudo gnssserver --inport "/dev/ttyACM0" --hostip 172.20.10.13 --outport 2101 --ntripmode 1 --protfilter 4 --format 2 > ~/Desktop/gnssserver\_output.txt 2>&1 &

sleep 10

#survey-in állapot olvasása

sudo python3 /home/base/Desktop/read\_survey\_in\_status.py &

read\_survey\_in\_status.py tartalma:  
Alapvetően a lényeg az, hogy a NAV-SVIN üznetet olvasom, aholis az active és a valid property a fontos. Amikor a SURVEY-IN folyamatban van akkor az active 1 értéket, a valid 0 értéket vesz fel. Amikor a SURVEY-IN befejeződőtt, akkkor az active 0 és a valid 1 értéket vesz fel. Ezt vizsgálva amikor a valid 1 értéket felvette akkor pirosan fog világítani a bázisállomáshoz kapcsolt led.

import time

import lgpio

from serial import Serial

from pyubx2 import UBXReader

SURVEY\_DURATION = 60

LED\_PIROS = 12

LED\_ZOLD = 13

h = lgpio.gpiochip\_open(0)

lgpio.gpio\_claim\_output(h, LED\_PIROS)

lgpio.gpio\_claim\_output(h, LED\_ZOLD)

stream = Serial('/dev/ttyACM0', 38400, timeout=5)

ubr = UBXReader(stream)

for raw, parsed in ubr:

    if parsed.identity == "NAV-SVIN":

        print(f"SVIN duration {parsed.dur} ({parsed.dur \* 100/SURVEY\_DURATION}%), valid? {parsed.valid}, active? {parsed.active}")

        if parsed.valid == 1:

            try:

                while True:

                    lgpio.gpio\_write(h, LED\_ZOLD, 1)

            except KeyboardInterrupt:

                lgpio.gpio\_write(h, LED\_PIROS, 0)

                lgpio.gpio\_write(h, LED\_ZOLD, 0)

                lgpio.gpiochip\_close(h)