Logging Receiver Data with GPS Coordinates

Ray WA1CYB

Stow

Beta Build

What is a Logging Receiver? Receives SDR signals and GPS Information and stores it in a file

Why would I use this? To show signal coverage for your station ***or*** a repeater station ***or*** ……

How is this implemented? GPS program on laptop uses gpsd (I use Ubuntu 22.04.1). Receiver created with GNU radio and a custom python block

What is the output file format?: A Comma Separated Variable format. Read by many programs

What SDR does it use: I have 2 versions, one for the RTL-SDR Dongle and 1 for the Ettus B205mini
From GNU Radio Companion it is easy to change the SDR type if desired.

Can the program be run from the command line?: Yes. Display not needed IFF you are sure it's tuned in correctly and doesn't drift. Recommend setting up in GNU Radio Companion 1st.

Any other features?: Real time RF monitor of the frequency with choice of demodulation. S-meter. Offset averaging S-meter for logging. Logging Bandwidth is programmable. You can type on the keyboard while running and the note will show up in the CSV file in the 6th column

Slides

- Screen Shots while running GNU Radio Companion
  2 Spectrum plots to tune in the signal for maximum strength
  1 Spectrum waterfall or Input Spectrum or time plot

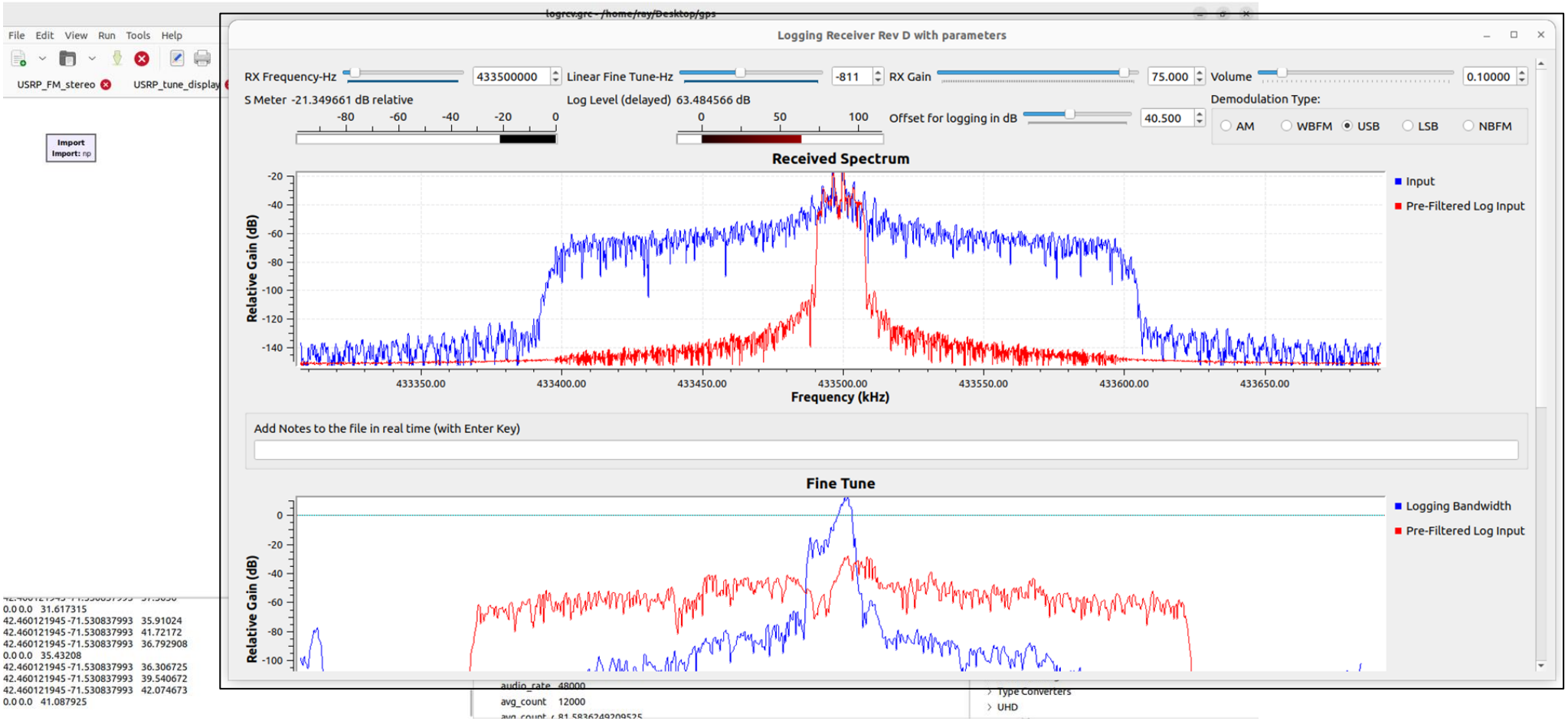
- Screens while running from command line
  2 windows: 1 with the graphical plots, the other the current location and signal level

- Control Description, CSV File Description

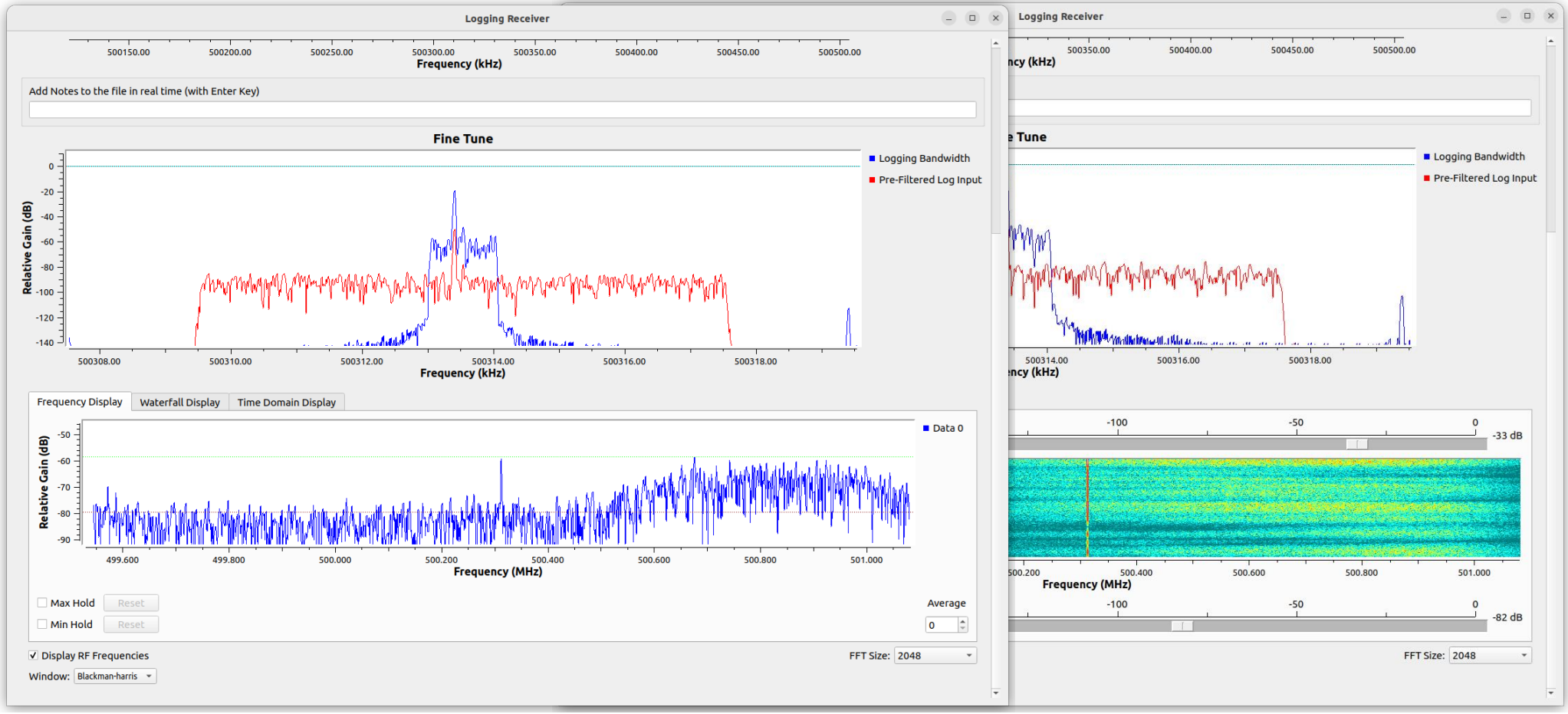
- Example Output Plotted by othe programs:
  - Excel my house to Bj's and back
  - Microsoft Excel: 3D Maps plugin plots data on the map directly, tile for signal value
  - Excel with 70cm coverage around my house when trees are wet (duty cycle controlled)
  - Converted a run with ***RouteConverterLinux*** and plotted on Google Earth
    - Modified the csv file so the Speed = Signal Level and new altitude = 100*Signal Level

- Location of Files on Github: **https://github.com/WA1CYB/.....**

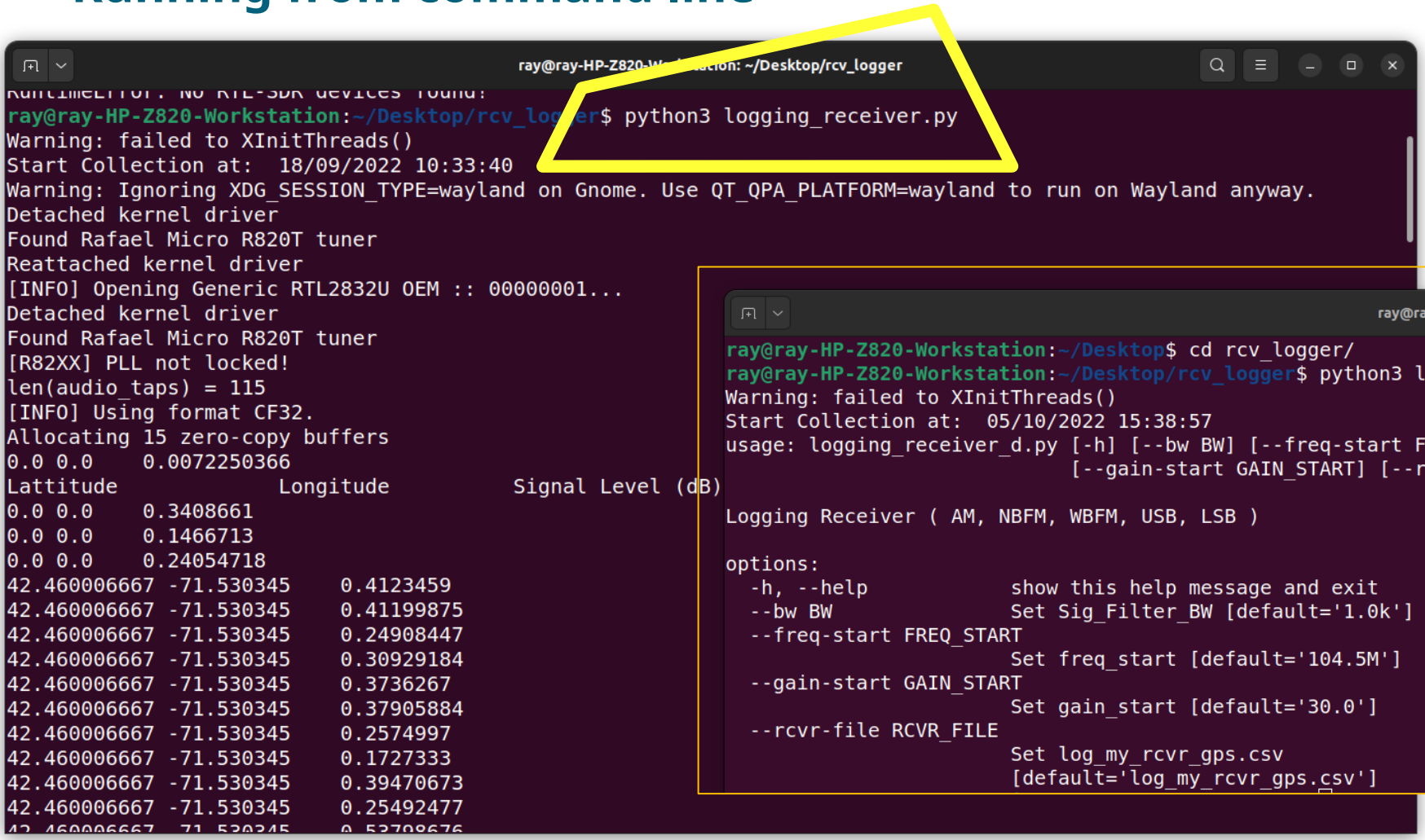
Typical Screen While Running GRC (GNU Radio Companion)



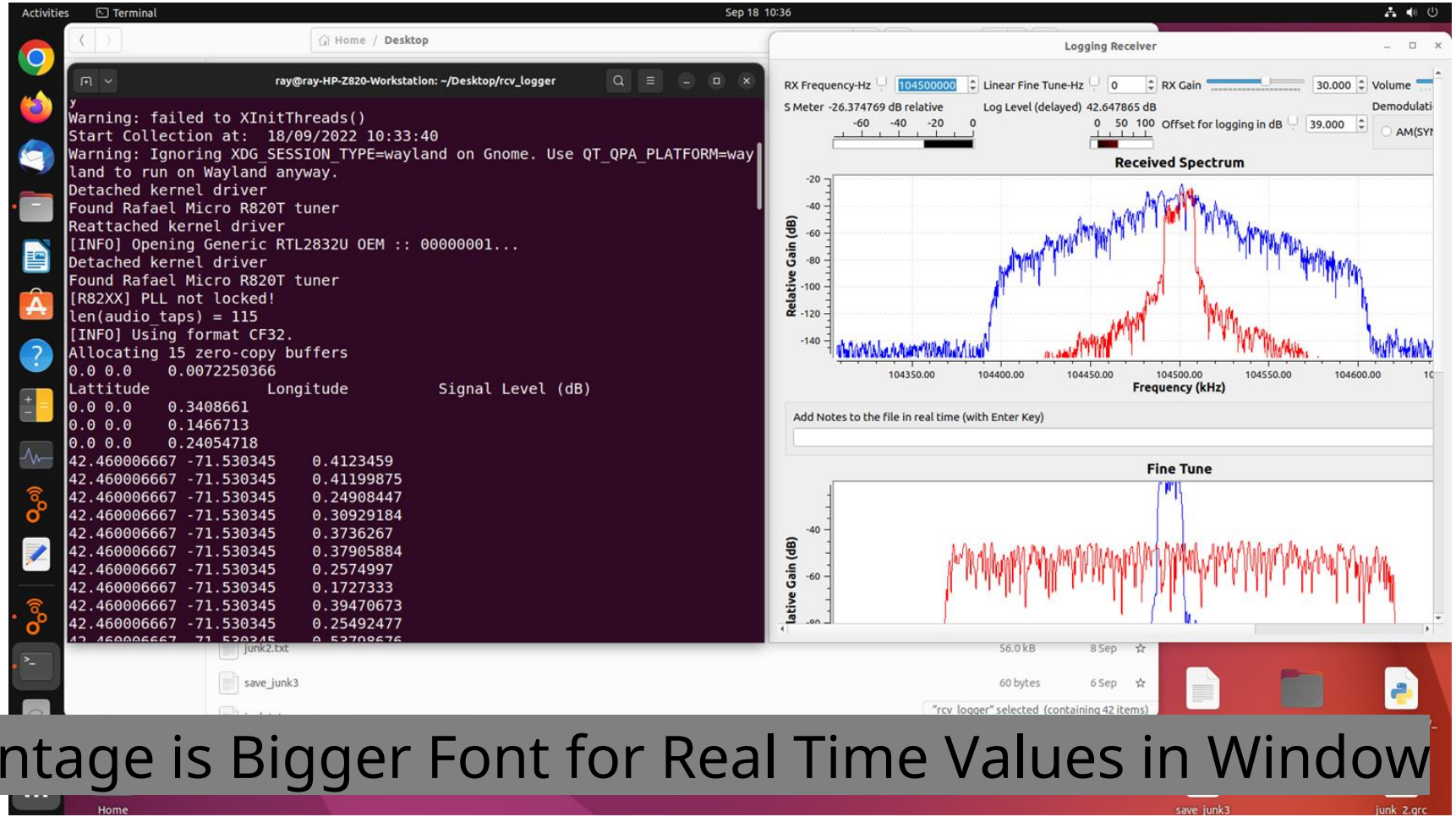
Bottom Portion of Screen When Scrolled Down



Running from command line

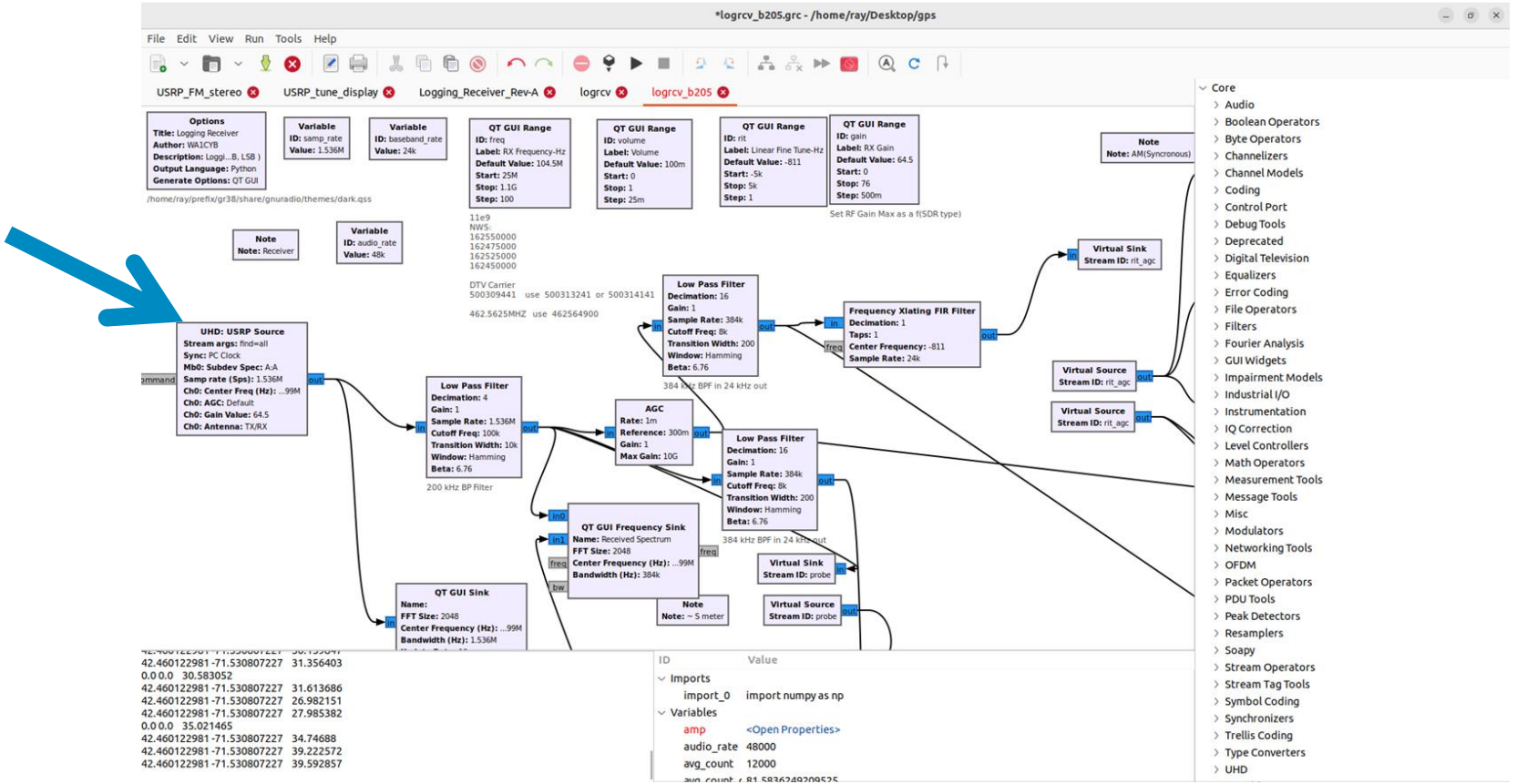


Running from command line – Typical Screen Output

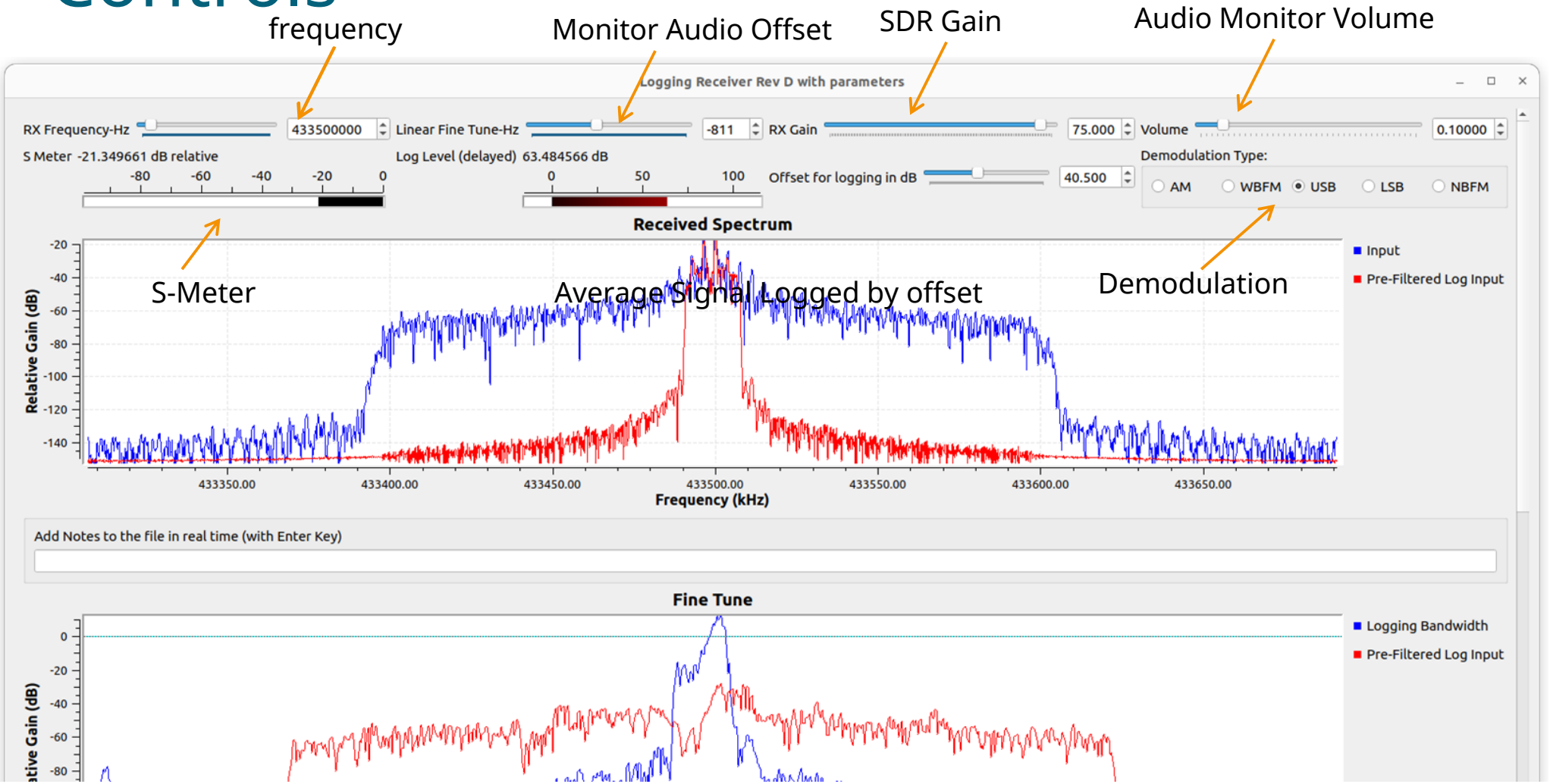


Advantage is Bigger Font for Real Time Values in Window

B205mini ‘version’



Controls


frequency
Monitor Audio Offset
SDR Gain
Audio Monitor Volume
Logging Receiver Rev D with parameters
RX Frequency-Hz
433500000
Linear Fine Tune-Hz
-811
RX Gain
75.000
Volume
0.10000
S Meter -21.349661 dB relative
-80 -60 -40 -20 0
Log Level (delayed) 63.484566 dB
0 50 100
Offset for logging in dB
40.500
Demodulation Type:
AM WBFM USB LSB NBFM
S-Meter
Received Spectrum
Average Signal Logged by offset
Demodulation
Relative Gain (dB)
-20 -40 -60 -80 -100 -120 -140
433350.00 433400.00 433450.00 433500.00 433550.00 433600.00 433650.00
Frequency (kHz)
Input
Pre-Filtered Log Input
Add Notes to the file in real time (with Enter Key)
Fine Tune
0 -20 -40 -60 -80
Logging Bandwidth
Pre-Filtered Log Input

Comma Seperated File Output log_my_rcvr_gps.csv

* Parameters that were run: Freq, gain, offset, date&time

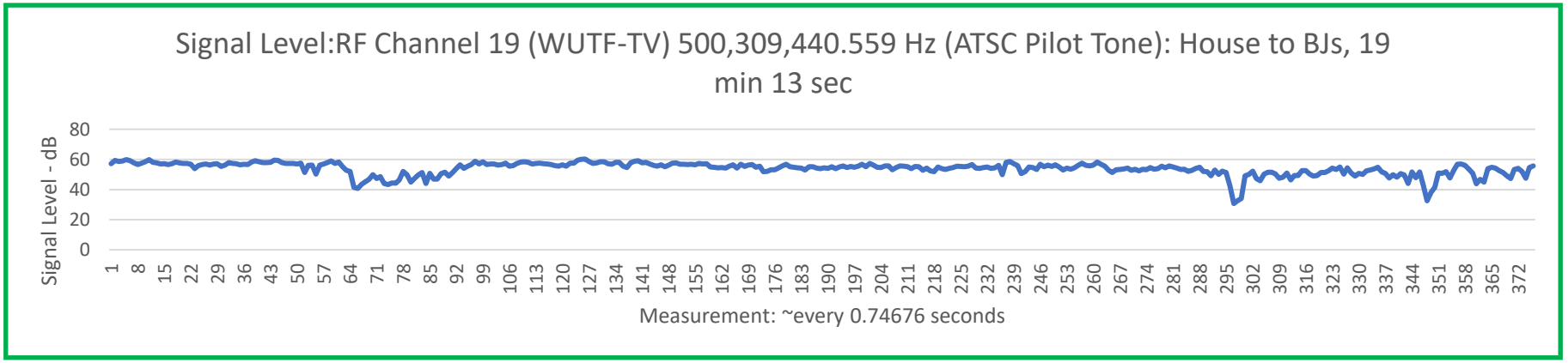
*Lattitude, Longitude, GPS_Time, Notes manually typed in while running

	A	B	C	D	E	F	G	H	I	J	K	L
1						logFREQ:433500000.0 Gain:77.0 Offset:44 start time:02/10/2022 16:16:24						
2	Lattitude	Longitude	Signal Level (dB)	Elevation	gps Time	notes						
3	0	0	56.212105	nan								
4						Lattitude	Longitude	Signal Level (dB)				
5	0	0	54.802338	nan								
6	0	0	54.54431	nan								
7	0	0	55.803623	nan	2022-10-02T20:16:40.460Z							
8	42.45983724	-71.53065748	56.35217	114.8786	2022-10-02T20:16:40.460Z							
9	42.45984108	-71.5306421	55.69851	114.4126	2022-10-02T20:16:41.150Z							
10	42.45984108	-71.5306421	55.405666	114.4126	2022-10-02T20:16:42.000Z							
11	0	0	55.38034	nan	2022-10-02T20:16:43.000Z							
12	42.45984021	-71.53063824	55.49604	113.0378	2022-10-02T20:16:43.000Z							
13	42.45984789	-71.53064592	56.128246	112.1043	2022-10-02T20:16:44.000Z							
14	42.45984702	-71.53064207	-0.5215416	110.7295	2022-10-02T20:16:45.000Z							
15	0	0	-0.26473618	nan	2022-10-02T20:16:46.000Z							
16	42.45984614	-71.53063822	-0.38463974	109.3547	2022-10-02T20:16:46.000Z							
17	42.4598519	-71.53063437	-0.42312622	108.6549	2022-10-02T20:16:47.000Z							
18	42.45984719	-71.5306459	48.71967	107.7461	2022-10-02T20:16:48.000Z							
19	0	0	50.535152	nan	2022-10-02T20:16:49.000Z							
20	42.45985295	-71.53064205	52.455	107.0464	2022-10-02T20:16:49.000Z							
21	42.45985295	-71.53064205	63.748	107.0464	2022-10-02T20:16:50.000Z							
22	42.45985872	-71.5306382	73.03367	106.3466	2022-10-02T20:16:51.000Z							
23	0	0	72.65481	nan	2022-10-02T20:16:52.000Z							
24	42.45985207	-71.5306382	72.5972	105.6715	2022-10-02T20:16:52.000Z							
25	42.45985784	-71.53063435	72.57312	104.9718	2022-10-02T20:16:53.000Z							
26	42.45985784	-71.53063435	72.34866	104.9718	2022-10-02T20:16:54.000Z							
27	0	0	72.45337	nan	2022-10-02T20:16:55.000Z							
28	42.45985976	-71.53064588	59.629234	104.7381	2022-10-02T20:16:55.000Z							

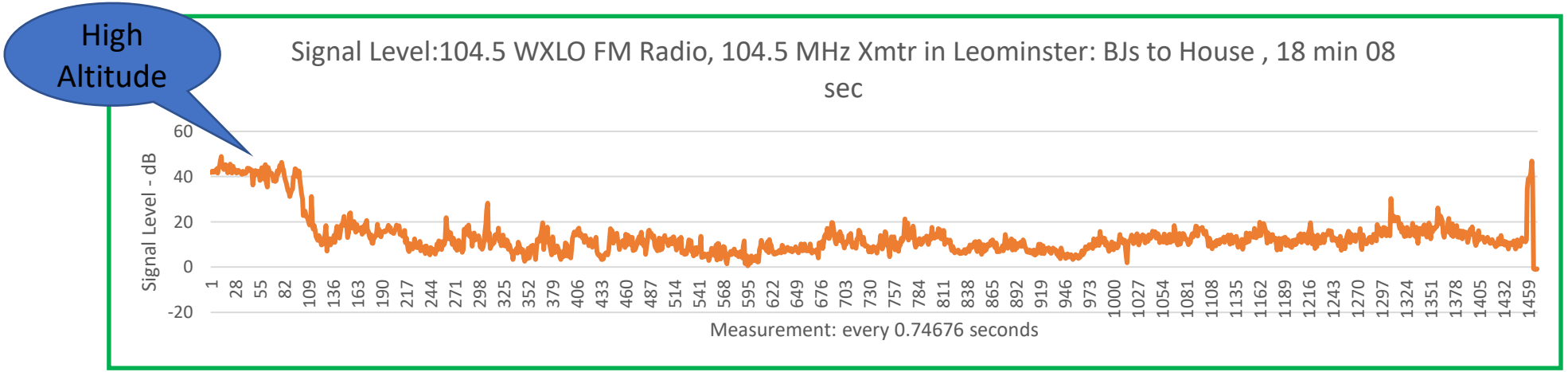
Notes Show up in this column

If you look at the data in Microsoft Excel or LibreOffice:

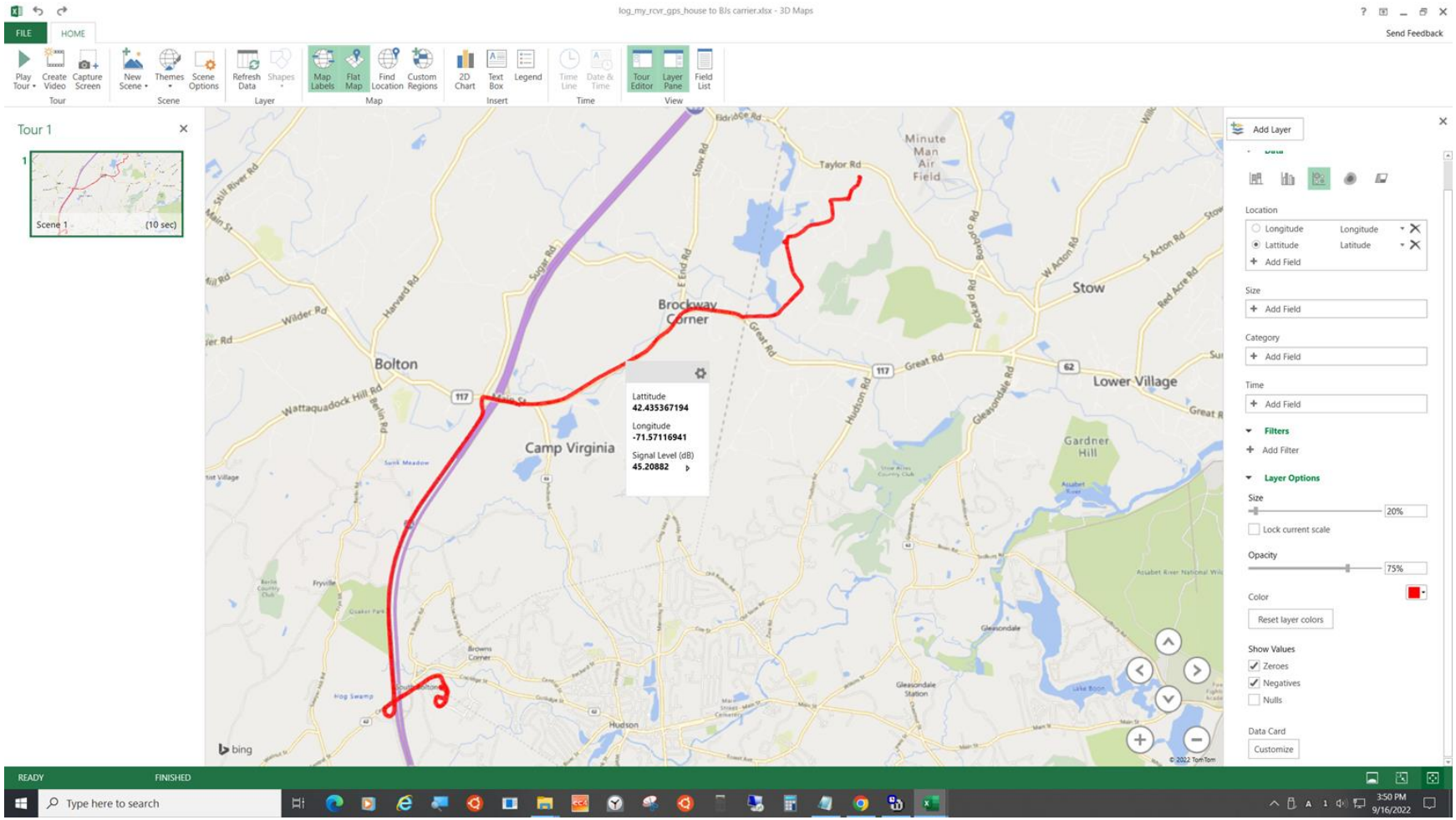
Examples



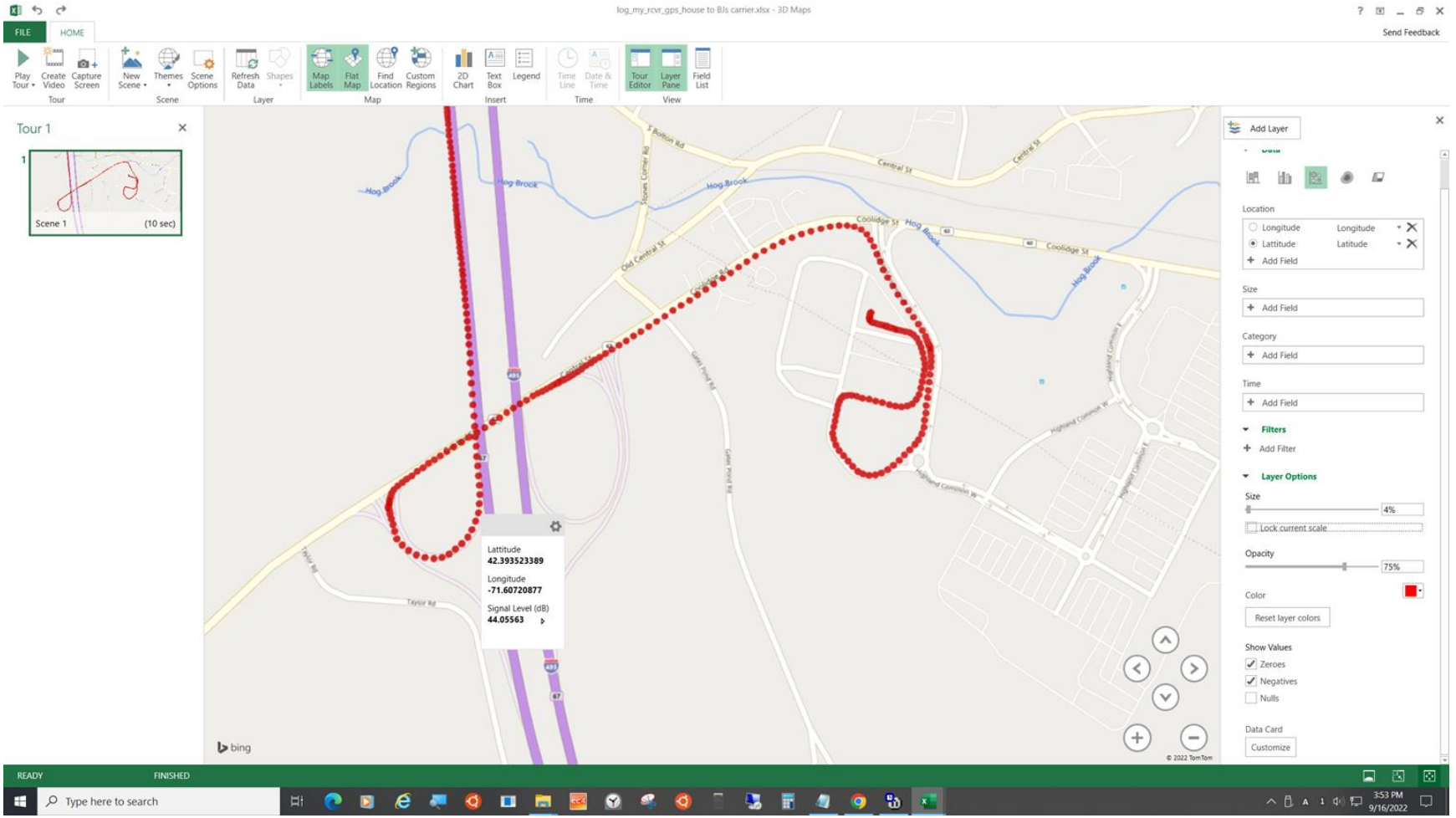
High Altitude



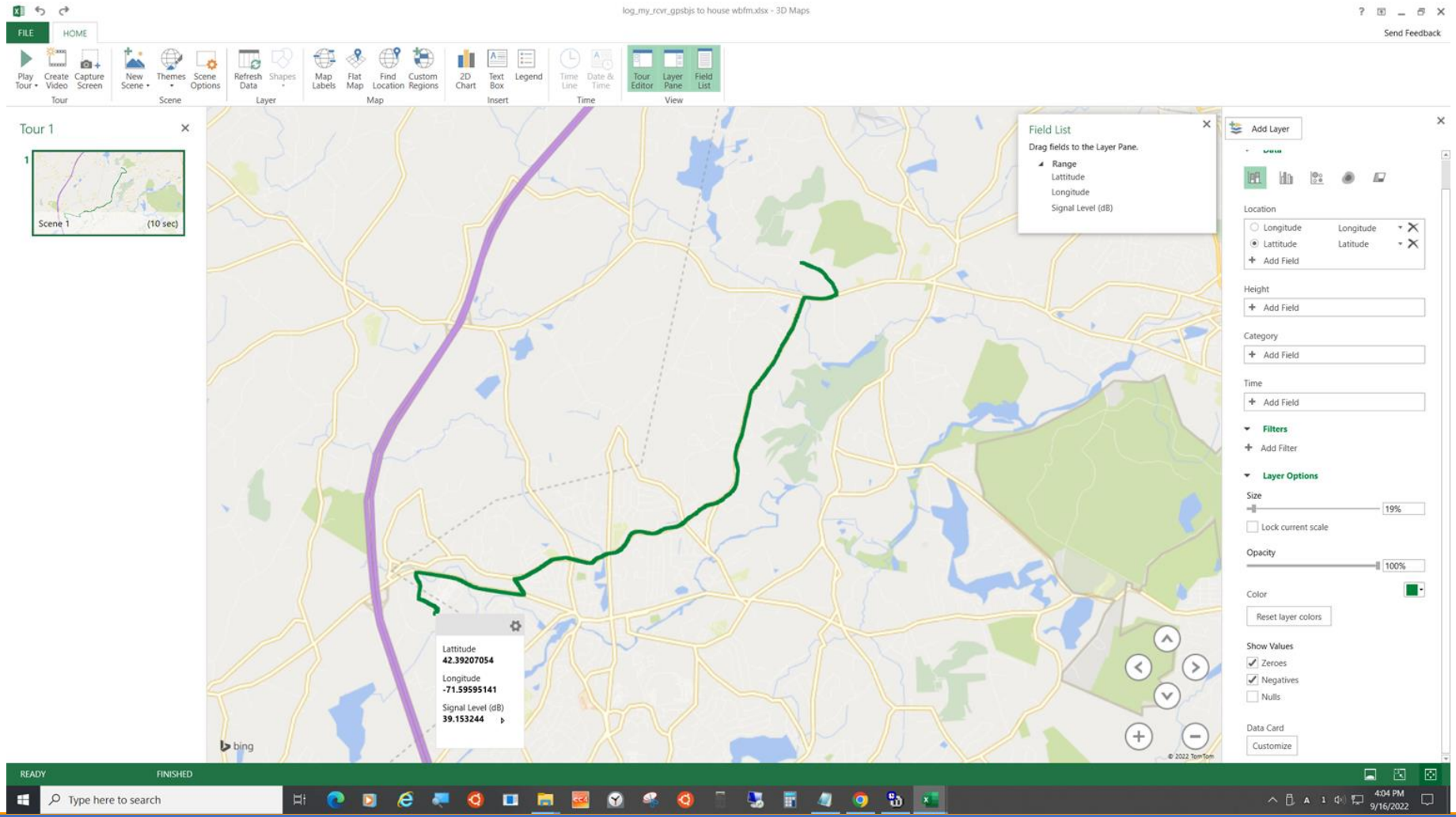
If you look at the data in Microsoft Excel: 3D Maps



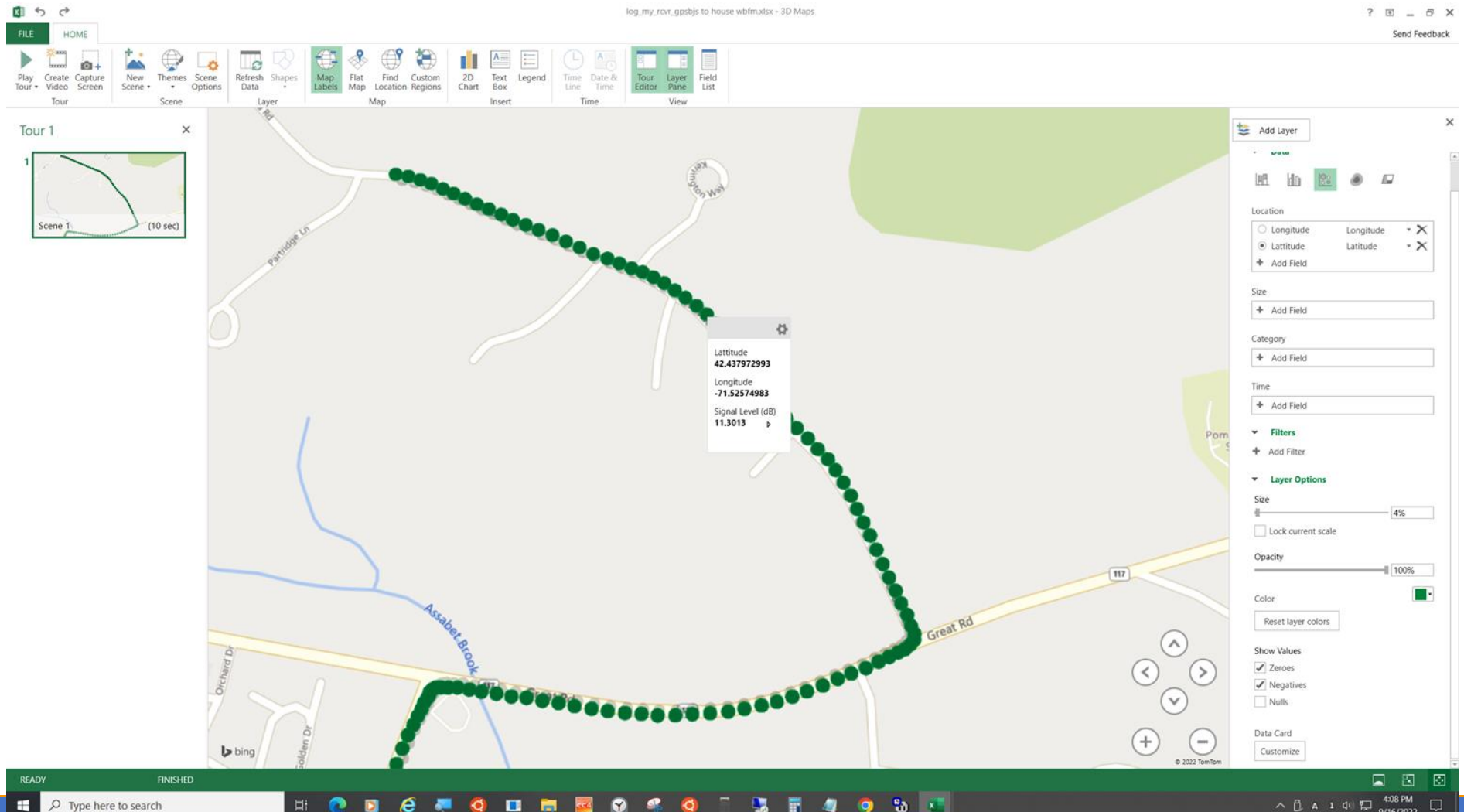
Cursor Can Show Signal Level (From Data Card)



Shows Route Back With FM Radio

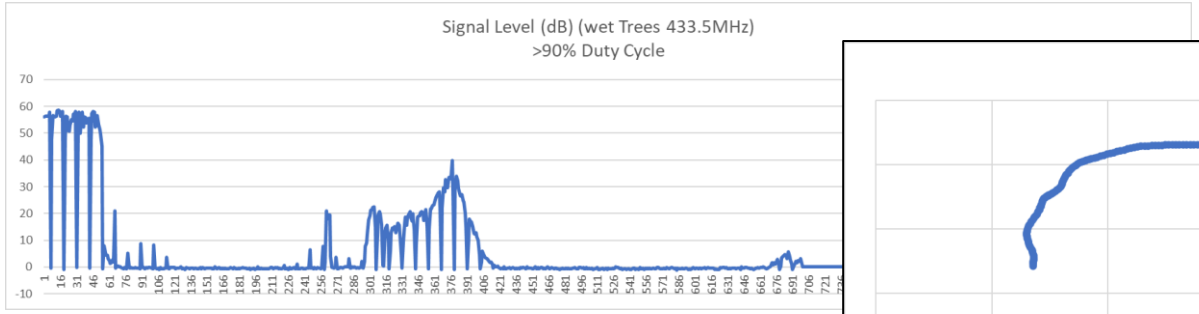


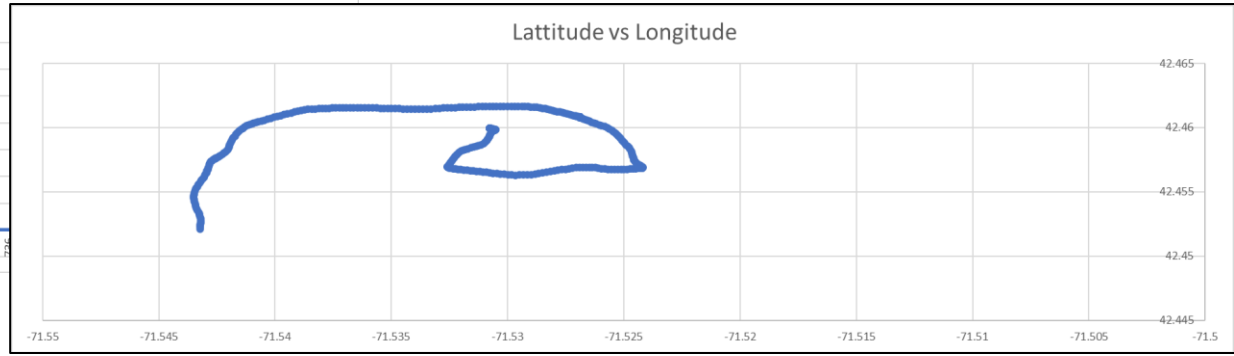
Cursor can show signal level (from data card)

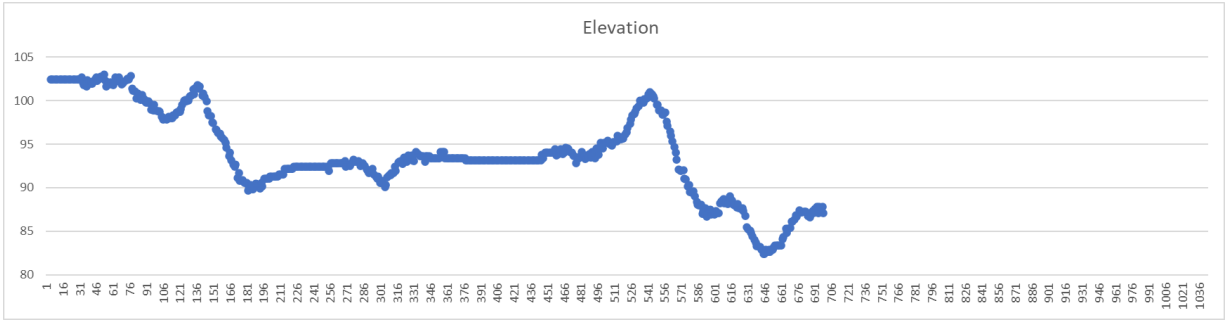


433.5 MHz Test From My Ground Plane To A Rubber Duckie On my Car

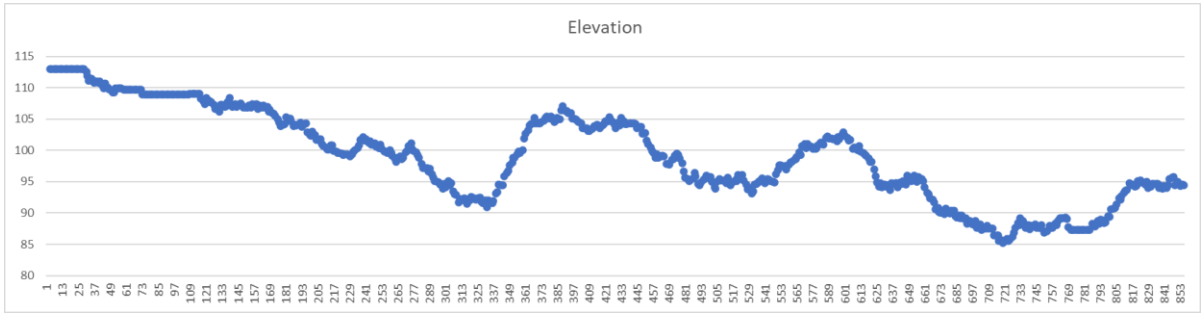
- Needed to modulate Transmitter with a duty cycle to prevent shutdown
- I used 66% to 90% on/off for most tests
- PTT “Modulation” circuit on github (Raspberry PI pico, transistor, relay)

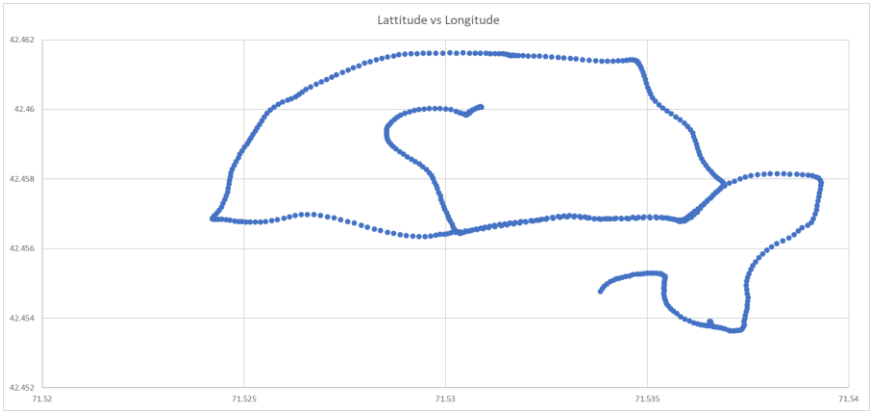


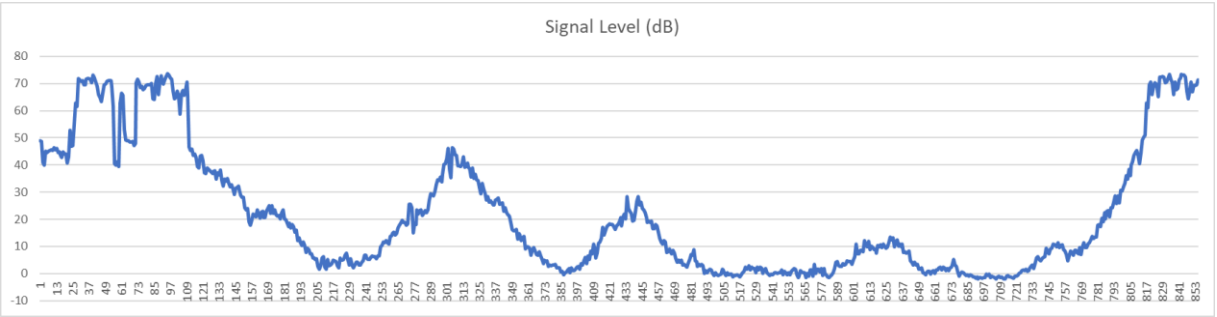


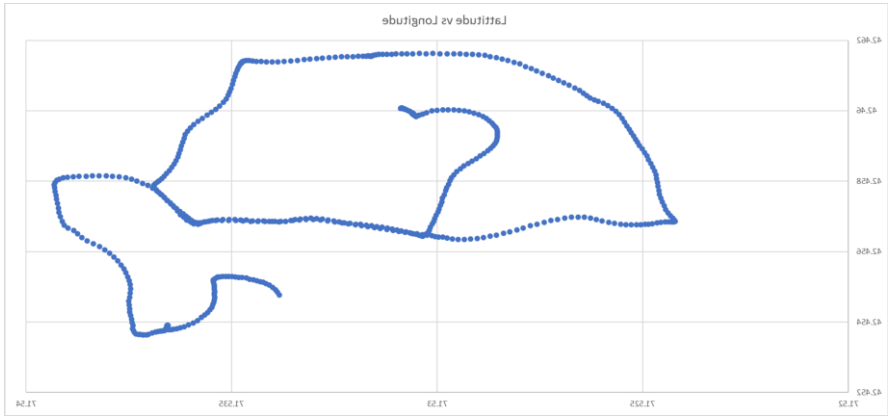


433.5 MHz Test From My Ground Plane To A Rubber Duckie On my Car ~10 watts ERP



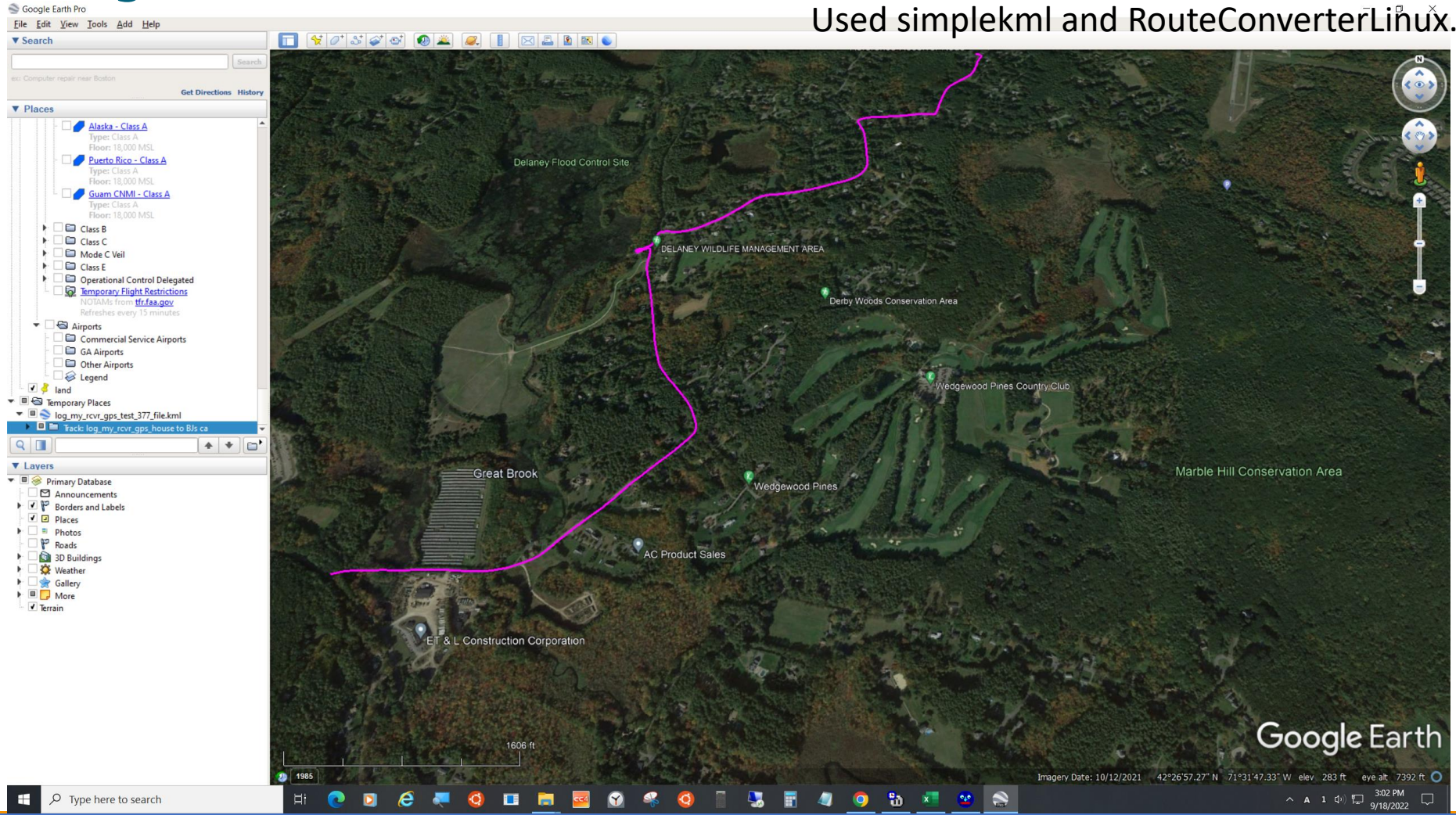




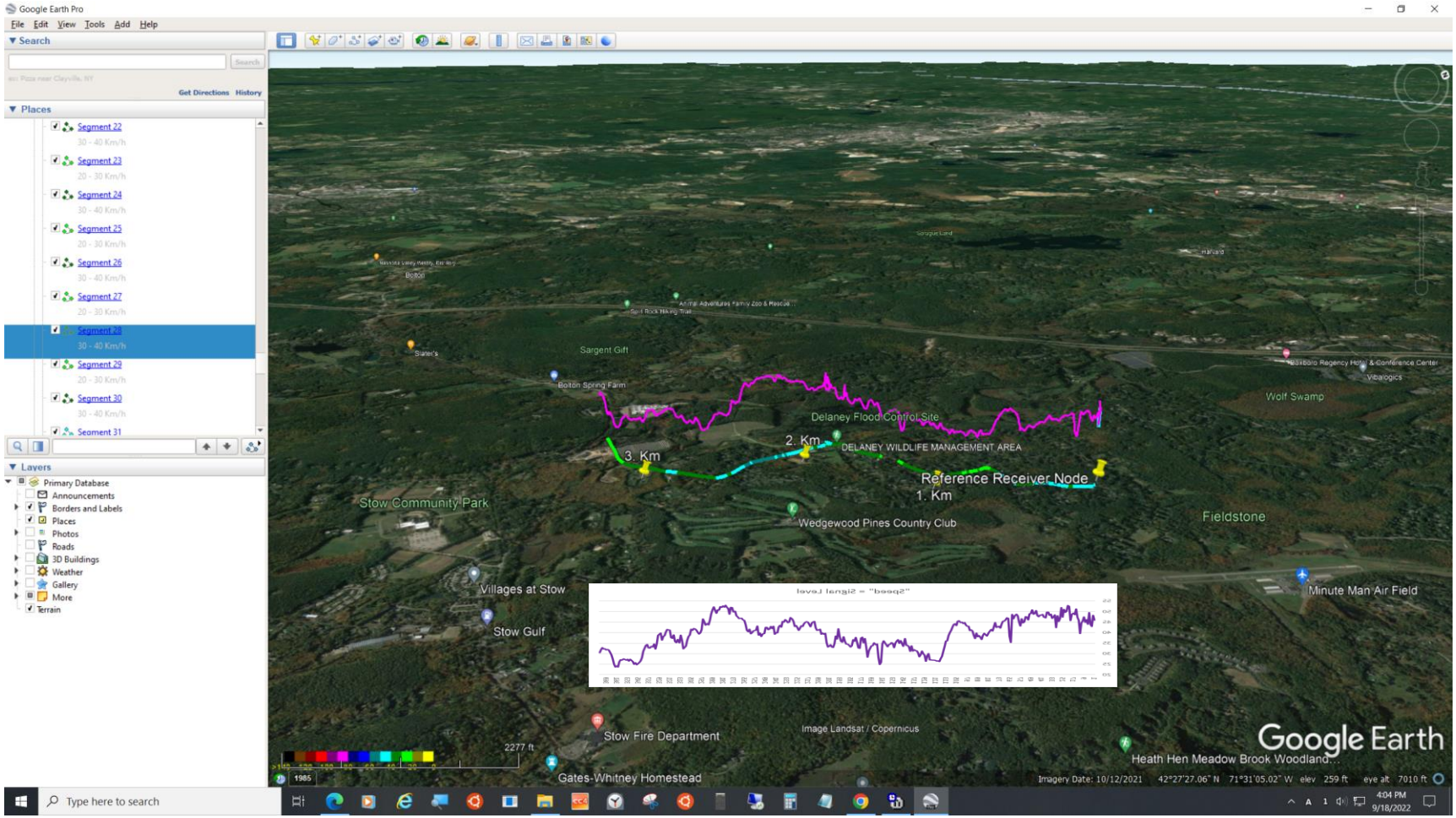


Google Earth Pro – Shows Path

Used simplekml and RouteConverterLinux.jar



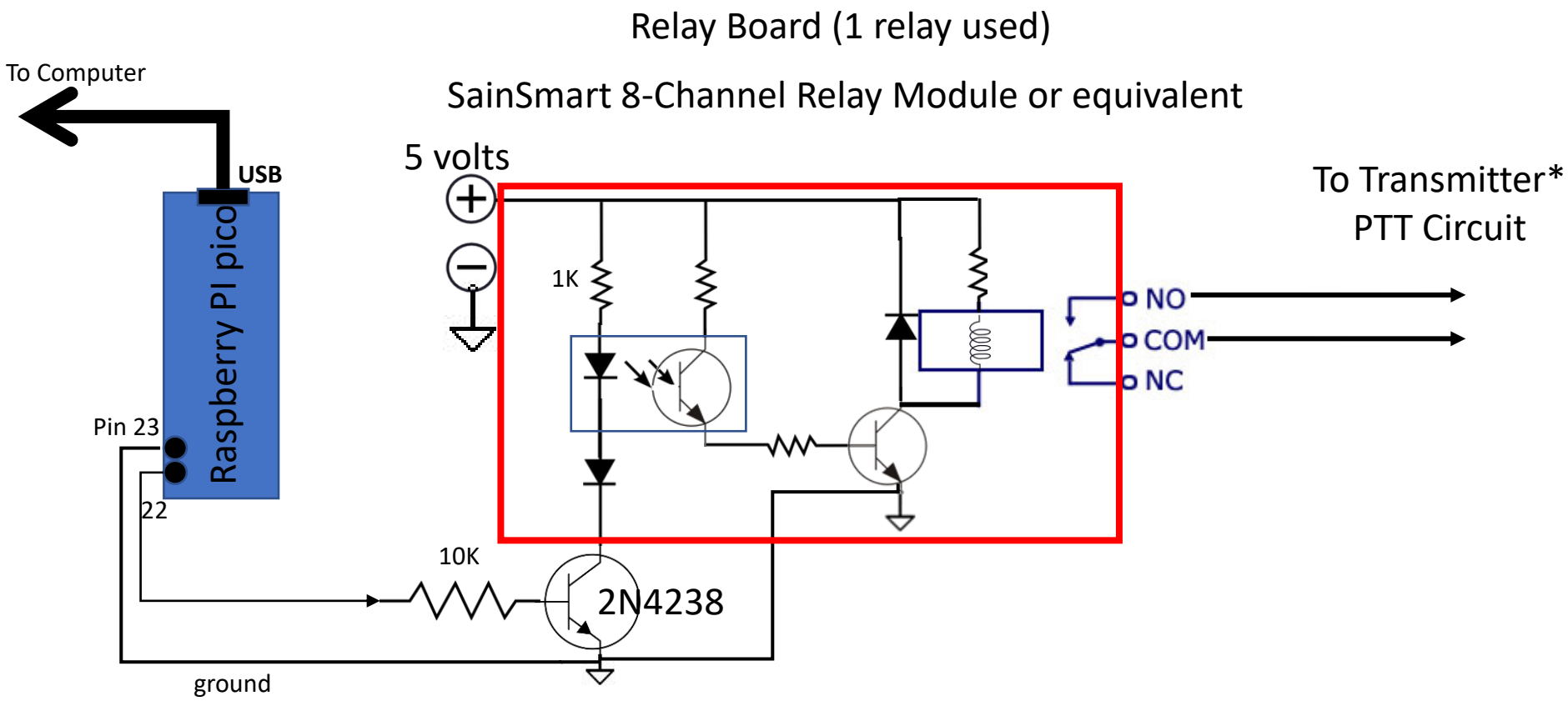
Google Earth Pro – Shows Speed = Signal Level: Alt = 100*Signal Level versus Excel (reversed)



Software Location for Download:
https://github.com/WA1CYB/.....

Next: Raspberry PI version

PTT Controller To Control The Duty Cycle Of The Transmitter



* PTT Circuit for a UV25X4 are pins 4 and 5

PTT Controller To Control The Duty Cycle Of The Transmitter
*** Software is a simple modification of the LED Blink Program ***

```
from machine import Pin, Timer
from time import sleep
from machine import Pin

tim = Timer()

led = Pin(25, Pin.OUT)
bfradio = Pin(17, Pin.OUT)
# PINS are GPIO Numbers not PIN Numbers
# REAL Pin(22) = GPIO(17)
# Ground = Pin(23)

myled = led

def tick(timer):
    global led, bfradio
    myled.on()
    bfradio.off()
    sleep(5.000)
    myled.off()
    bfradio.on()
    sleep(10.000)
# polling time must be <2 seconds. Set to 0.2 sec (5 Hz)
tim.init(freq=5.0, mode=Timer.PERIODIC, callback=tick)
```