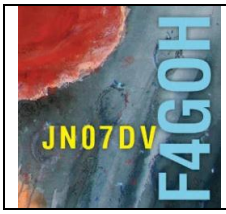
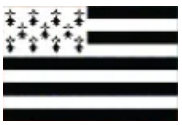


Raspberry Pi for HAM Radio
Part 1



Summary:

Part 1: Presentation

Ver 1.2, 4/09/2020

1.1 Introduction

Since 2012, the Raspberry Pi nano computer has become an increasingly important part of the DIY and "maker" community. The increase in power of the Raspberry Pi over the years offers very interesting possibilities for radio amateurs. Indeed it allows not to permanently monopolize a PC in the decoding of frames with software like WSJT-X, FLDIGI, etc..., without forgetting the possibility to control the Raspberry Pi remotely and thus to be able to work outside the radio shack as I can sometimes do on my couch. Moreover, this nano computer is now widely used in any Hotspot (DMR or D-STAR).

So why in a club, so few OM use this tool?

I have often asked myself this question and I think it comes from the use of the operating system (OS). When you power on a Raspberry PI, the OS, called Raspbian is stored on a microSD card. This is a Linux distribution specially designed for the Raspberry.

When you want to install a software under Linux, it is no longer a matter of double clicking on a downloaded file like "setup.exe" like under Windows. Indeed, it is often necessary to use the console and thus type "command lines" to install or configure a program. This can seem off-putting and frankly very tedious. We are so used to using a graphical environment that we feel like we are back to the early 80's micro-computing.

However, the mastery of some basic commands in Linux console allows us to consider the installation of different programs and hardware such as:

- WSJT-X
- FLDIGI
- QSSTV
- DIREWOLF (APRS)
- SDR receivers (rtl-sdr, Lime SDR, SDRplay, Airspy, etc.)
- GNU-RADIO
- ADS-B Décoding
- Programming
- Etc.

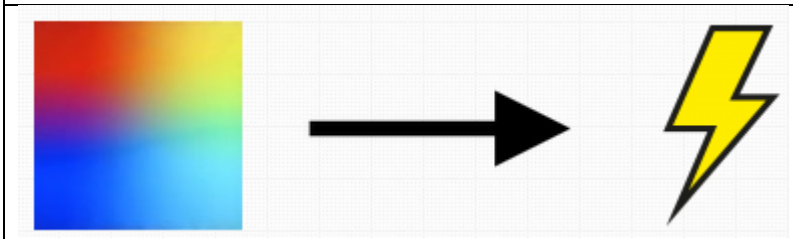
In the appendix on page 21, the main characteristics of the Raspberry Pi.

1.2 Required material

There are many kits containing the Raspberry PI with a case, an mSD card and a power supply from resellers such as https://www.elektor.fr , https://www.kubii.fr/40-les-cartes-raspberry-pi or amazon

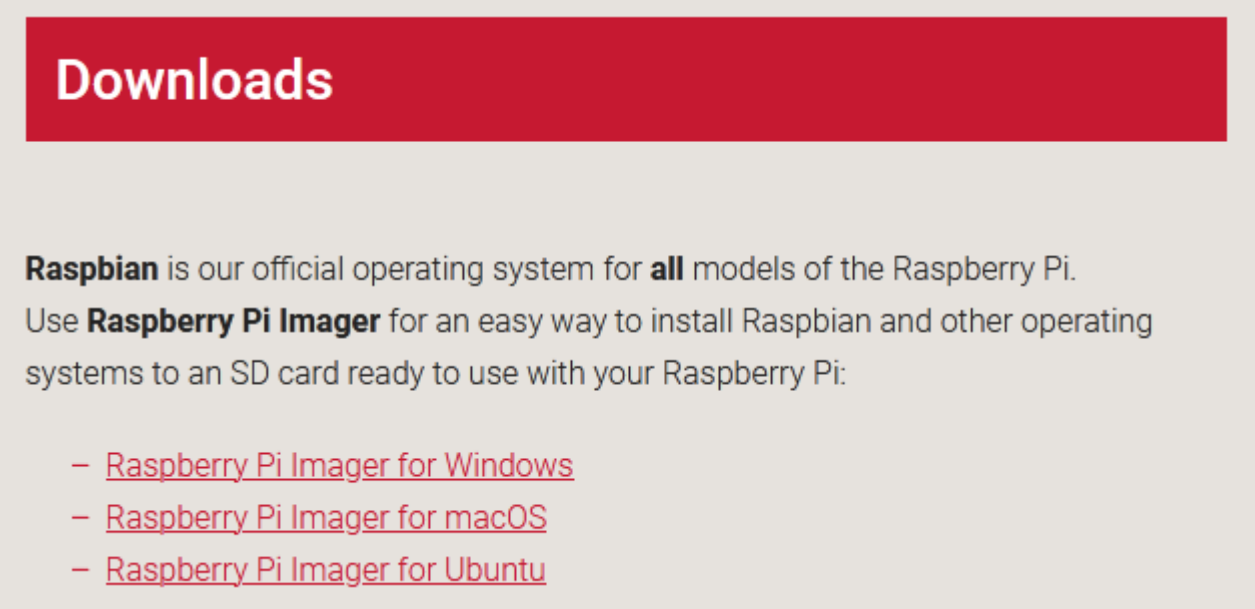
It will be necessary to pay attention to the choice of the power supply: 2 Amperes mini for a Raspberry Pi 3 and 3 Amperes for the Raspberry Pi 4.

Raspberry Pi 4 Kit	Mouse, USB Keyboard and display with HDMI input.
	

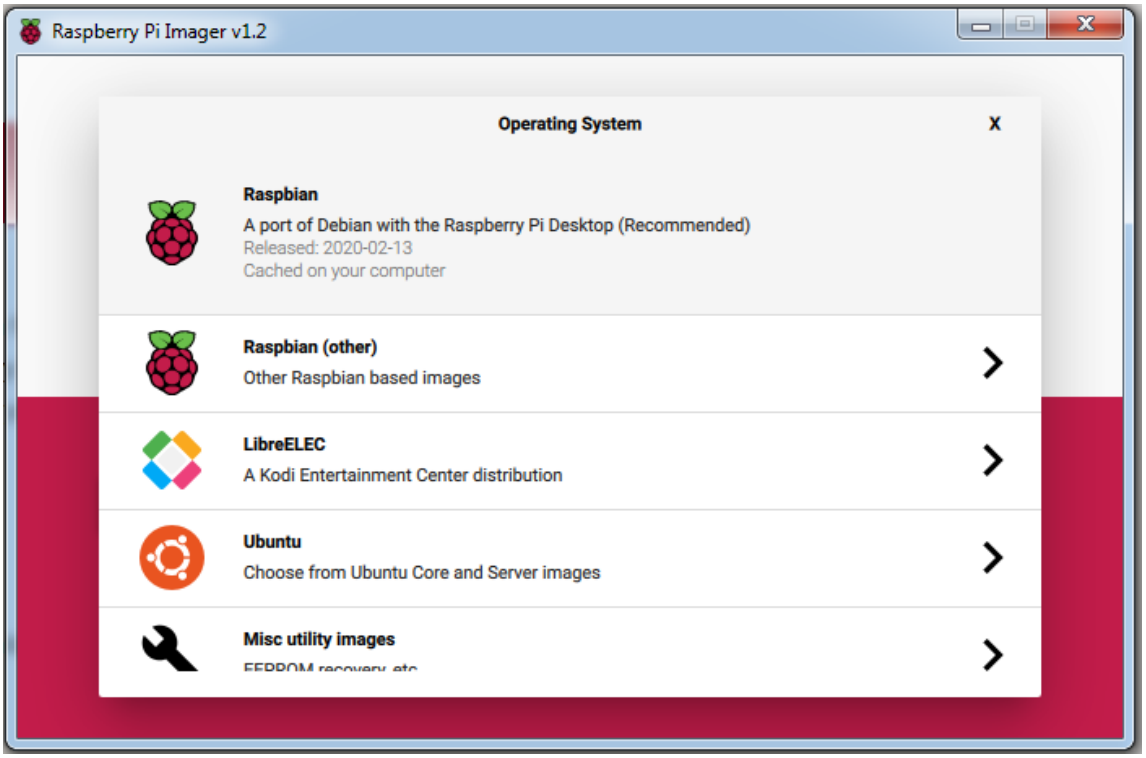
If the power supply current is too low, a yellow lightning flash will appear in the upper right corner of the screen when the power is turned on. For projects requiring few resources (PiAware) and a power supply on battery and solar panel, the Raspberry pi zero W is a good compromise with a consumption of 160mA. However, the network connection will be done by WIFI.	
	

1.3 Preparation of the operating system

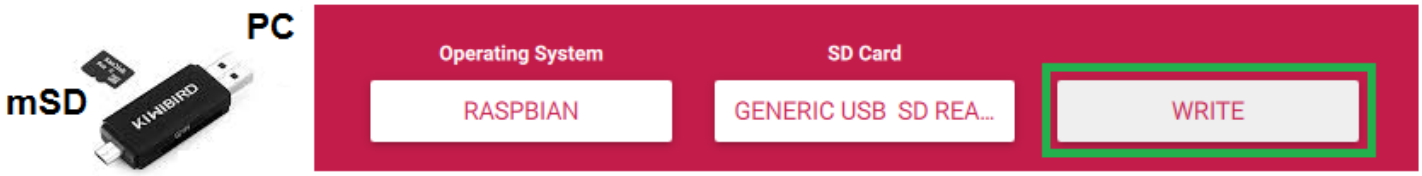
With your Computer, download the new Pi Imager operating system management utility for the Raspberry PI. https://www.raspberrypi.org/downloads/



After downloading and installing, select the Raspbian with Raspberry Pi Desktop OS.



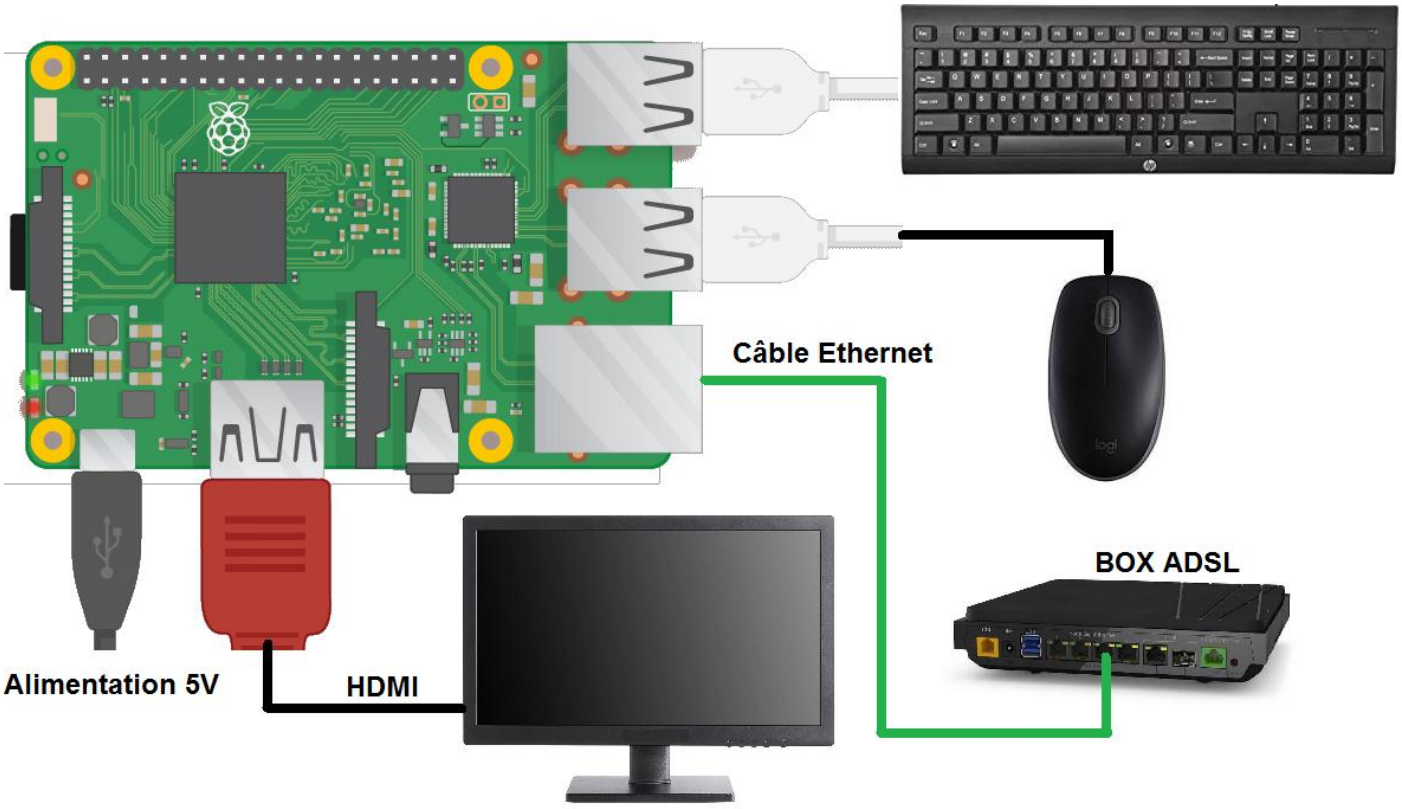
Insert the micro Sd card into a USB reader on the PC, select the SD card reader here GENERIC USB Reader, then click on WRITE.



The Pi Imager utility will automatically download and copy the selected operating system to the mSd card. The time of the operation depends on the Internet connection, you have to be patient.

1.4 Power on RPI

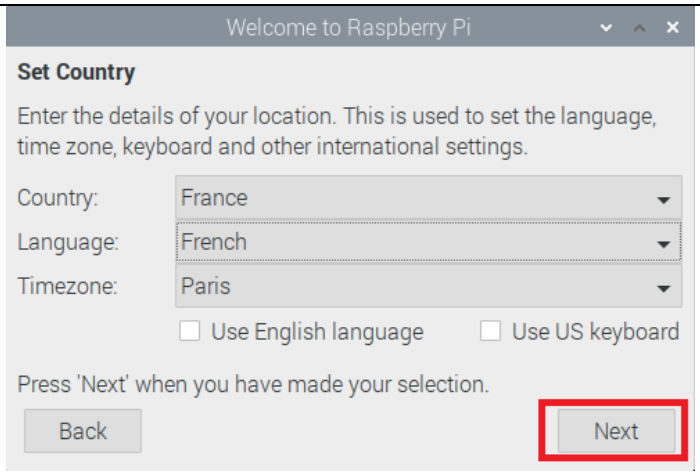
Carry out the following wiring, ending with the 5V power supply of the Raspberry.



By default, the language at startup is English, a configuration window appears in order to configure the user's country. (for me in French)

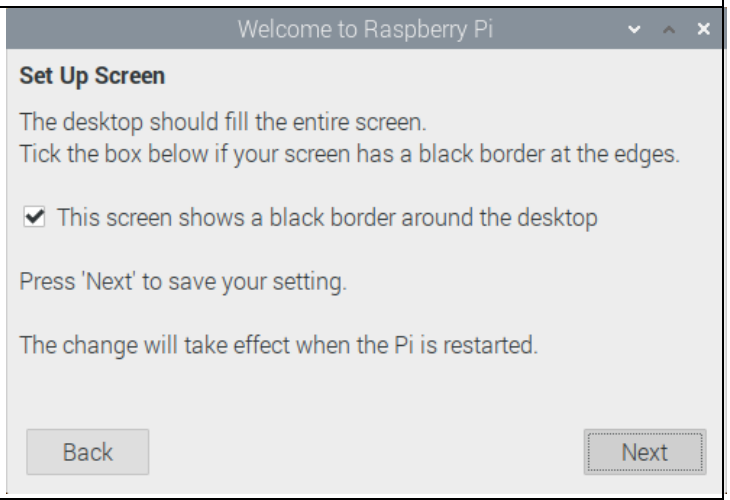
Welcome window after startup

Select your language

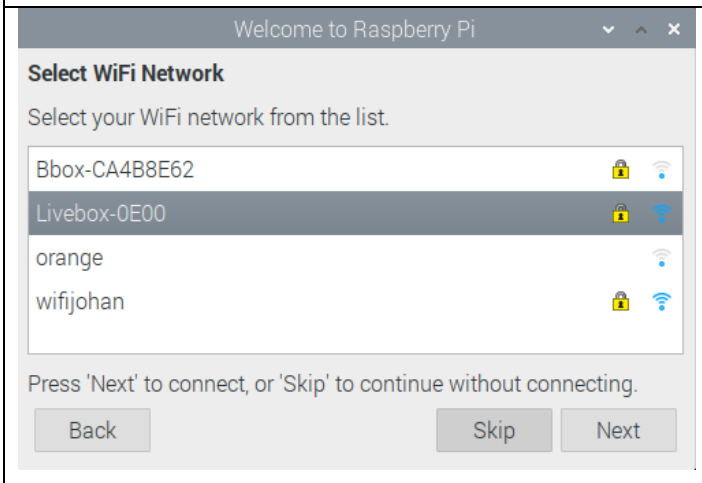


Change Password
(I have kept the same one for practical reasons in this document)

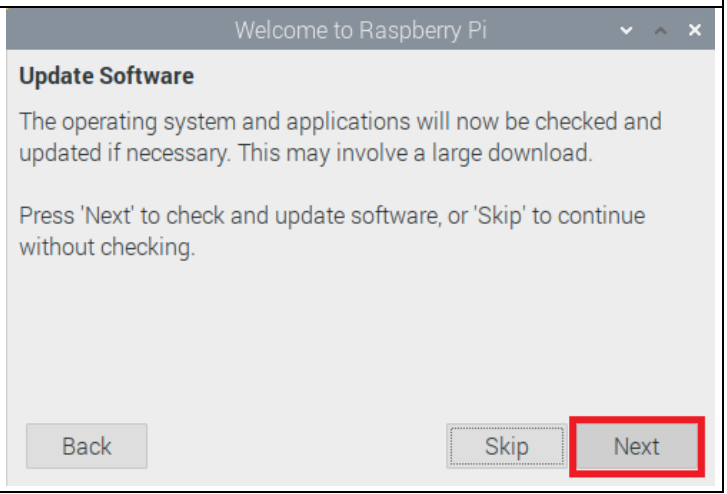
Enhances the image if there is a black border around the screen.

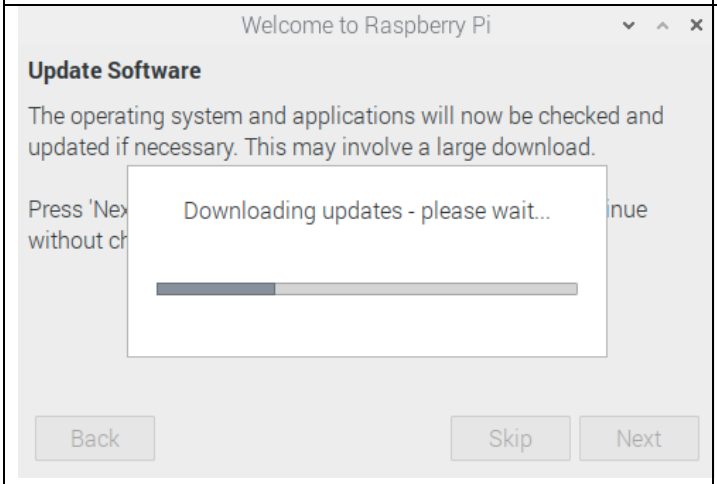
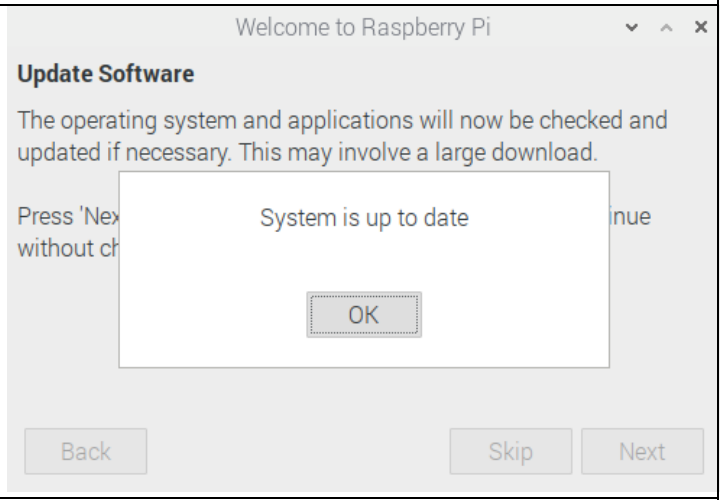
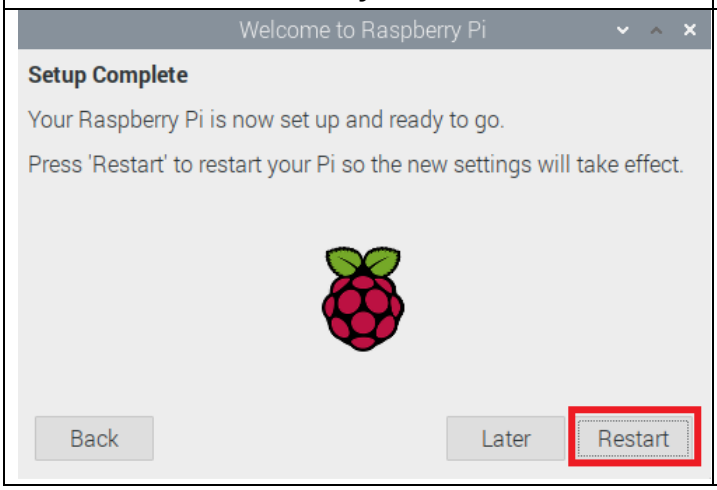
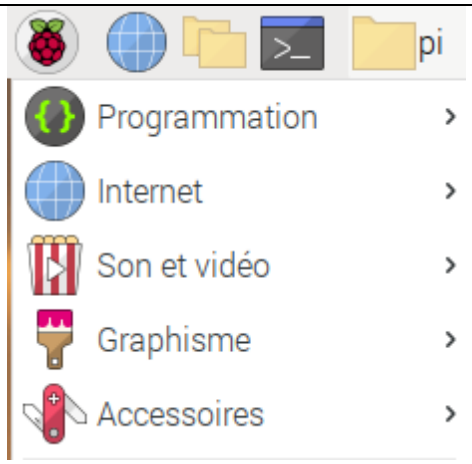


Setting up a WIFI connection (optional)



Updating is recommended



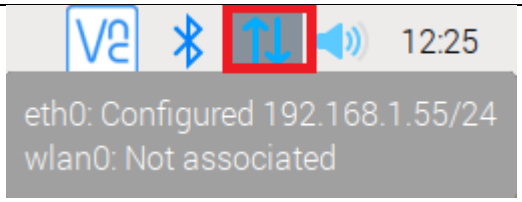
Download	Update completed
	
A restart is necessary	**The menu is now in French**
	

1.5 Find the IP address of my Raspberry Pi

In many applications, it is necessary to know the IP address of your Raspberry PI connected to the network, for example to connect remotely, to perform maintenance operations.

For the 1st power on, a screen is necessary. But once the Raspberry Pi is configured, there will be no need for a screen, keyboard or mouse. The management will be done using your usual PC.

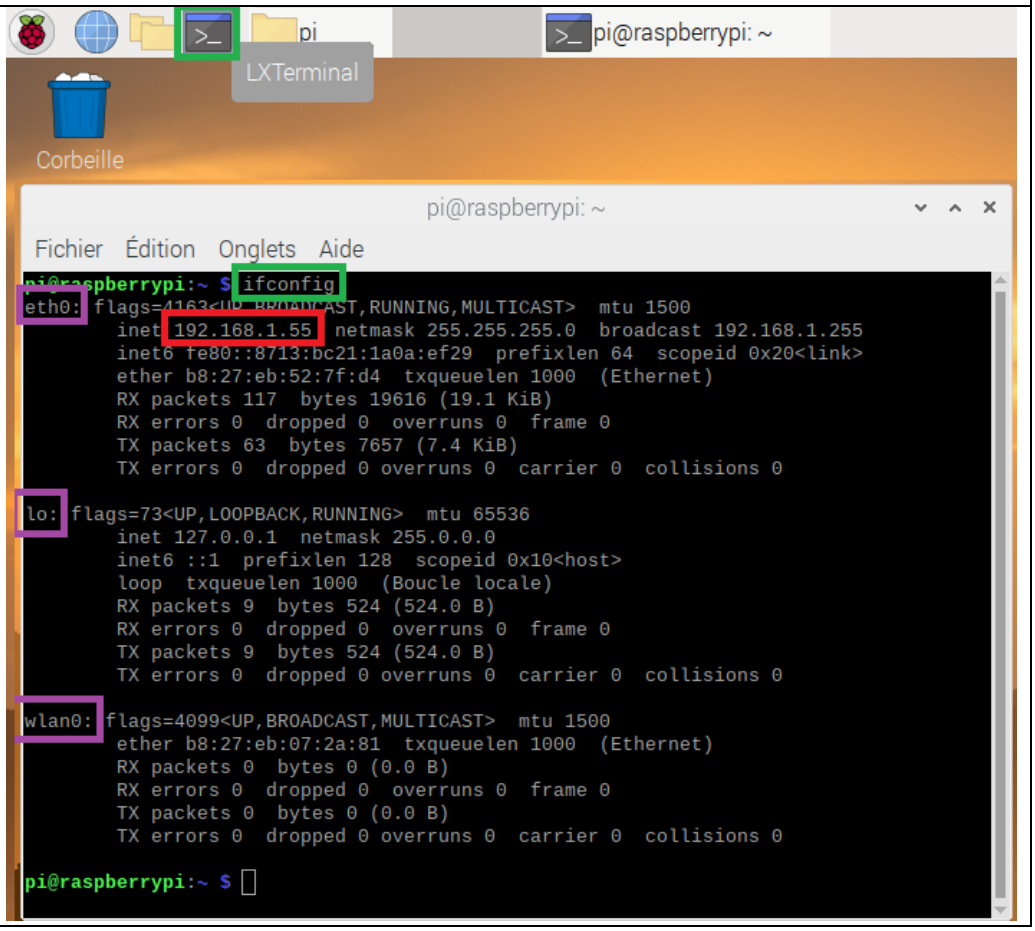
Method 1 :
Move the mouse to the top right of the screen on the blue arrows without clicking. The IP address will appear after 2 seconds.



Méthode 2 :
Open the LXTerminal command prompt using the desktop icon
Or the keys **CTRL+Alt+t** simultaneously.

Then type: ifconfig followed by the enter key.

The ip address of my RPI is 192.168.1.55 (Red frame)



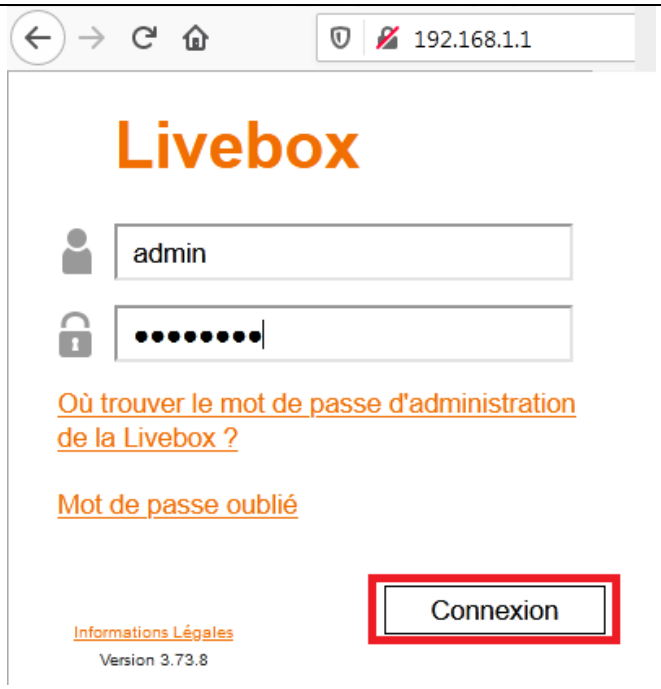
Note :

eth0 corresponds to the wired interface by Ethernet cable ;
lo : is the IP address of the local loop always 127.0.0.1 ;
wlan0: is the WIFI interface.

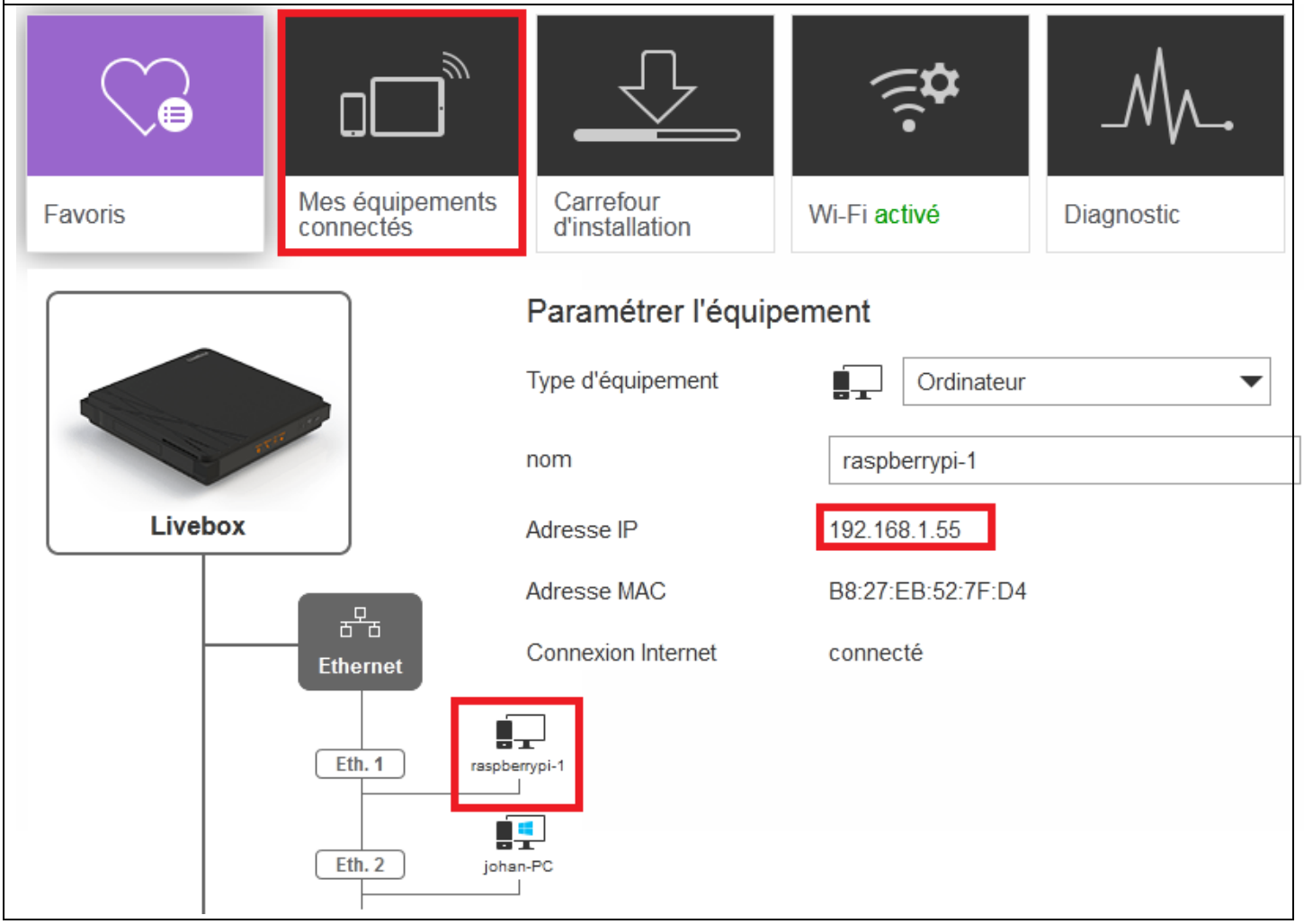
Method 4 :

With a browser, connect to the administration menu of your ADSL box.

Example for a French ADSL Box



Rechercher un équipement dans le réseau, puis identifier l'adresse IP.

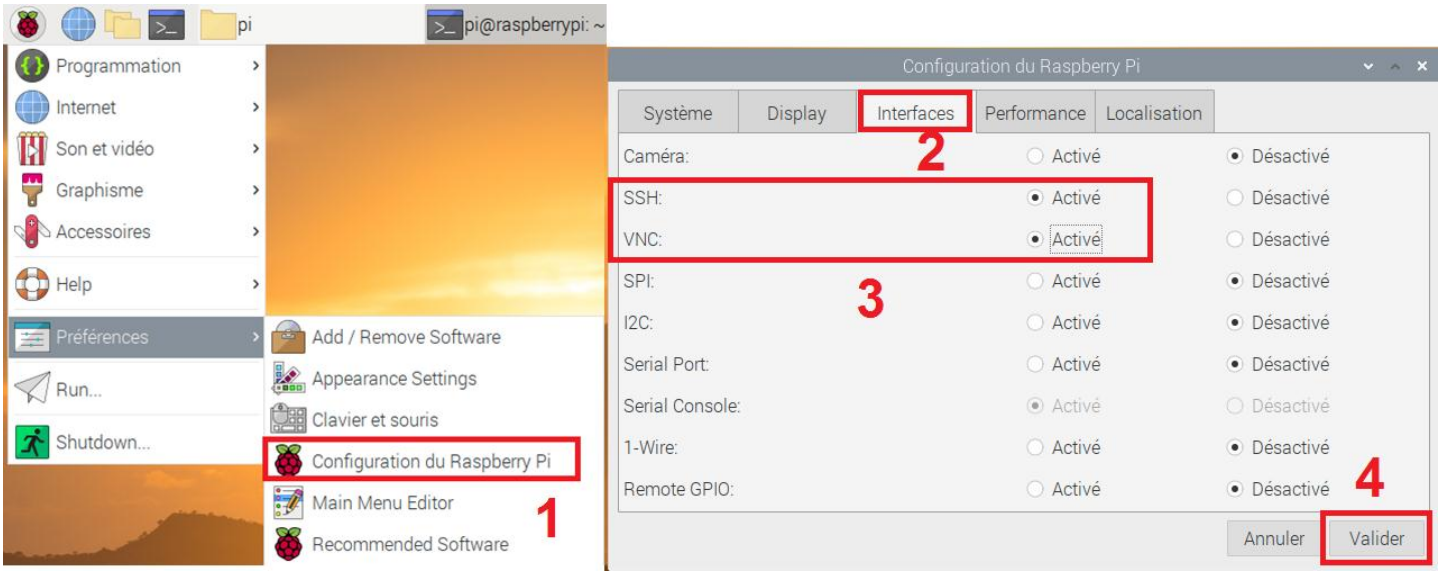


1.6 Take control of the Raspberry Pi remotely with a PC

It is not necessarily necessary to have an additional monitor for the Raspberry PI. Most of the time, especially for the command lines, I only use the console. For this, there is only the power supply and the Ethernet cable connected to the Raspberry PI.

1.6.1 SSH and VNC

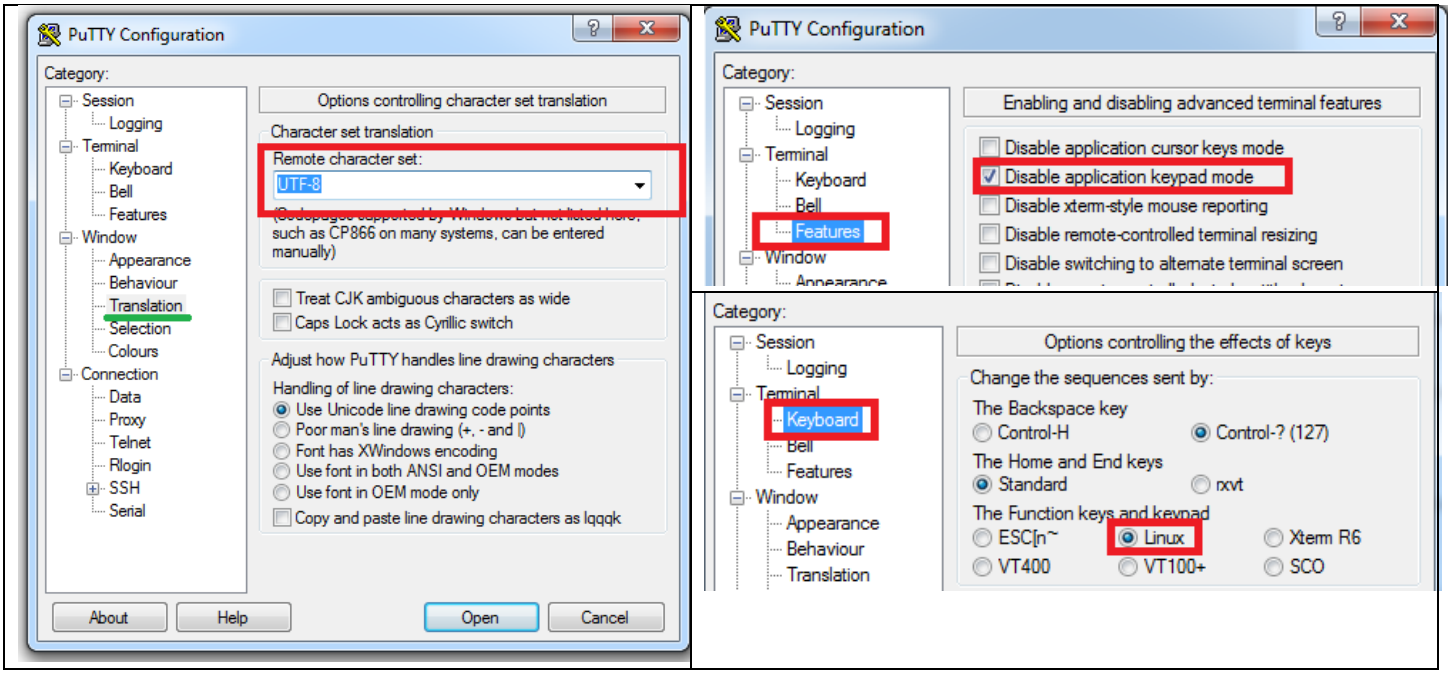
By default, the connection is disabled. To enable these options, go to the Configuration menu of the Raspberry PI. Follow steps 1 to 4.



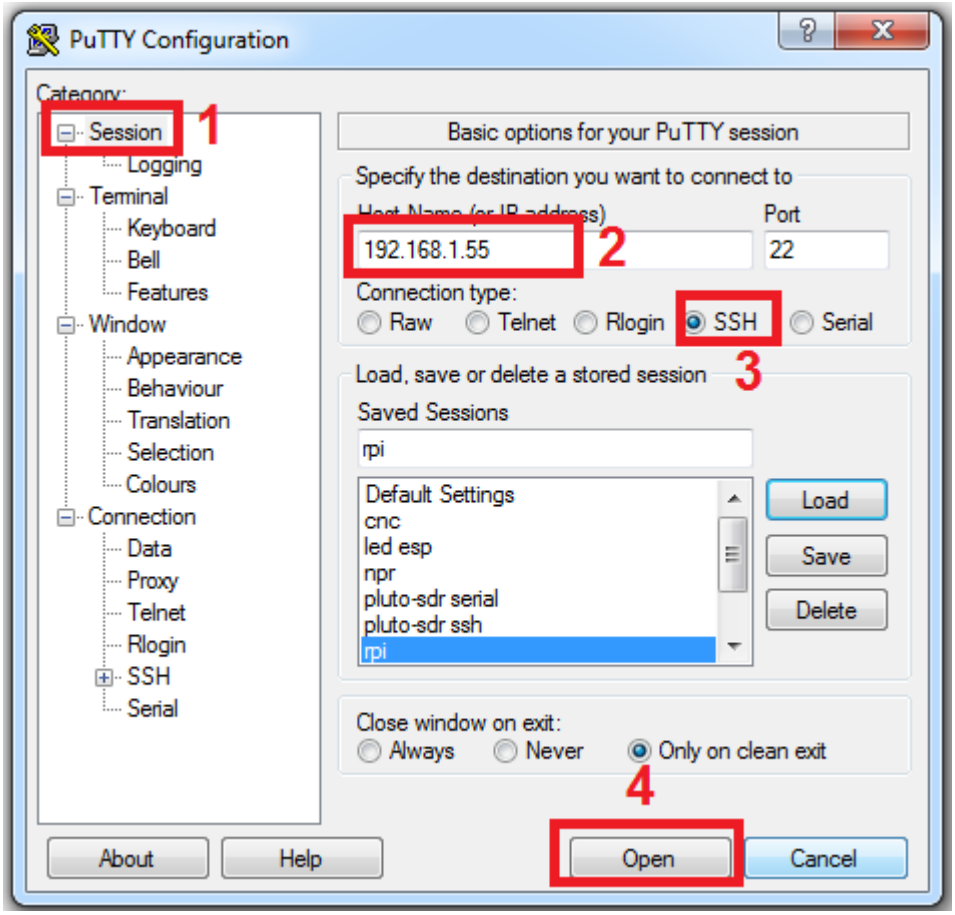
1.6.2 Putty

Download and install putty https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html

In the Translation menu, check the UTF-8 character set, disable the keypad mode application, set the keyboard to Linux. This configuration allows to use the keypad.



Type the IP address of the RPI in SSH, then Open : steps (1 to 3)



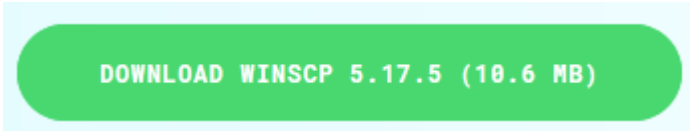
Enter the login and password entered during configuration on page 6.

Login :	**pi**
Password :	**raspberry**

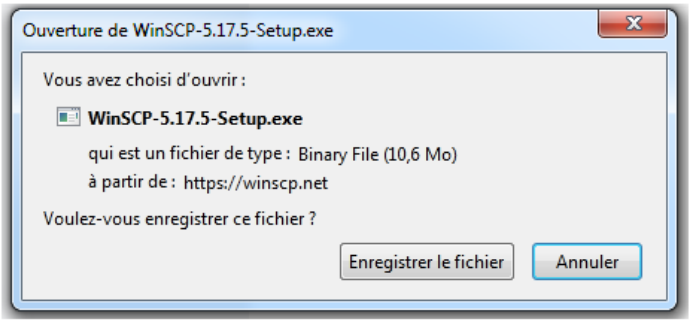
The command prompt appears as before with LXTerminal

1.6.3 WinSCP

WinSCP is a utility that installs in a PC to access the files of the Raspberry PI
Download and install WinSCP https://winscp.net/eng/download.php

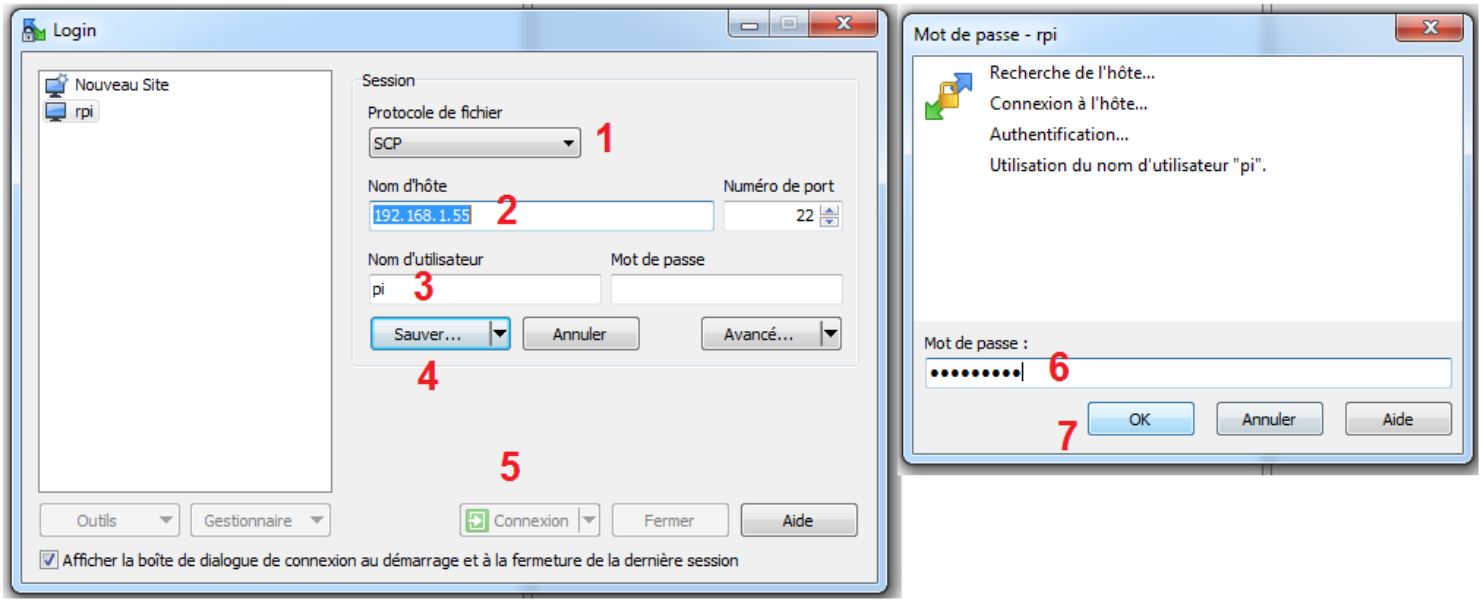


Your download of WinSCP-5.17.5-Setup.exe will start shortly...

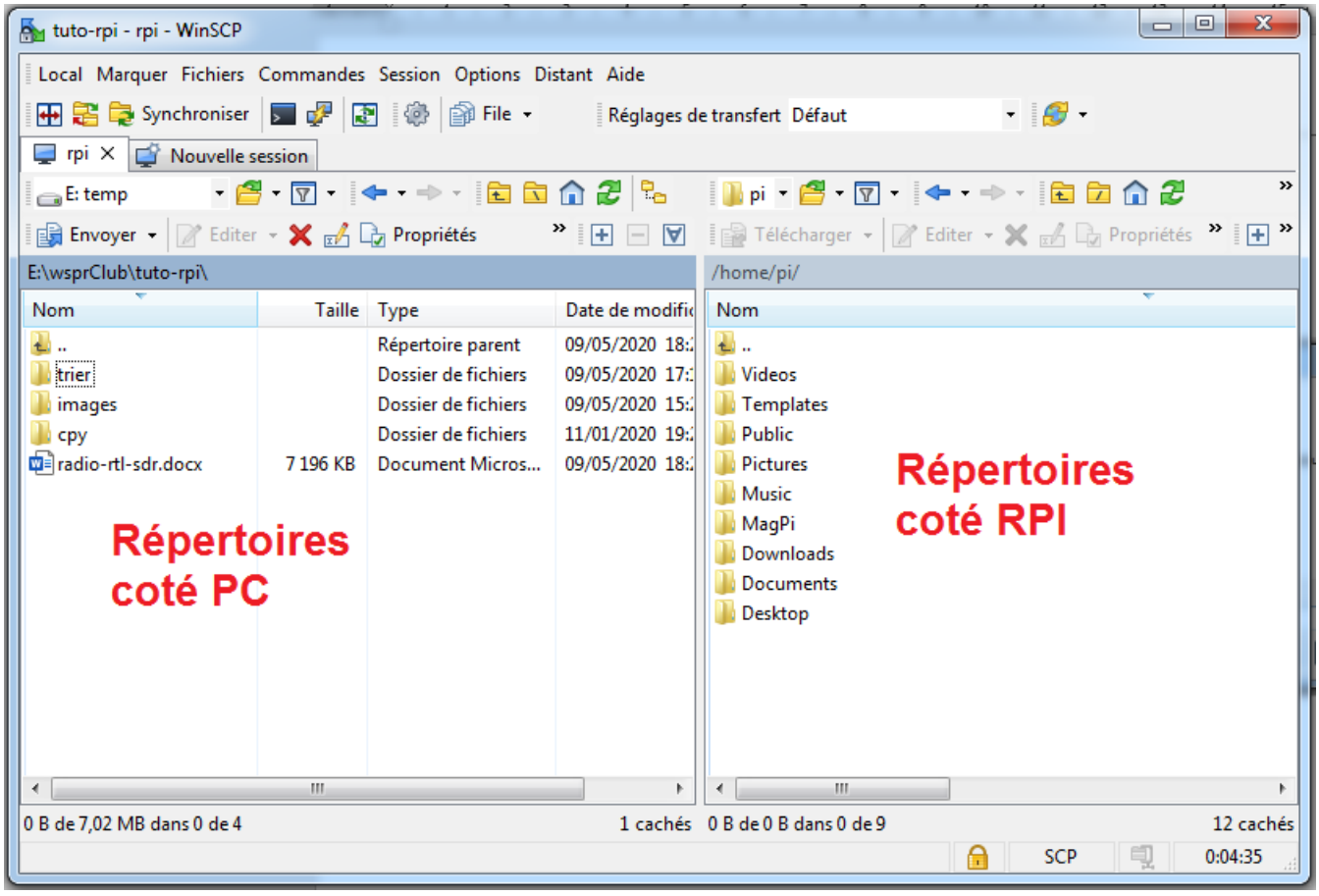


Follow steps (1 to 6) to connect to the Raspberry Pi

password: `raspberry`



To copy the files, simply move them with the mouse between the 2 PC and RPI explorer windows. (The user directory of the RPI is /home/pi)



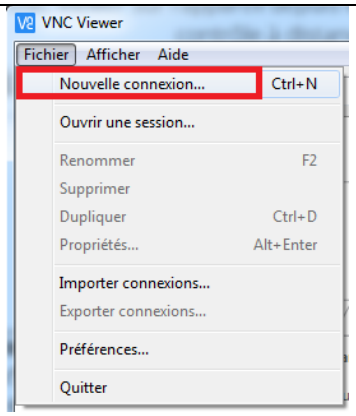
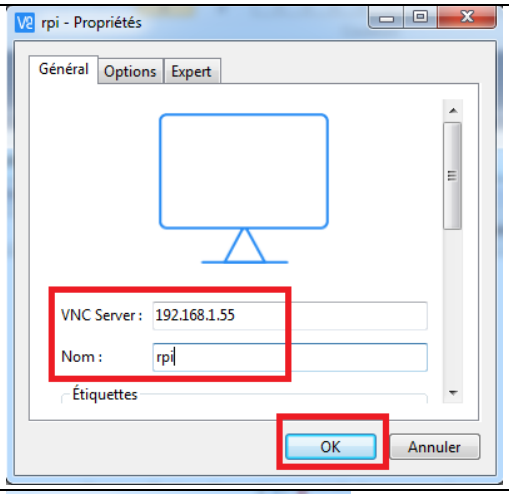
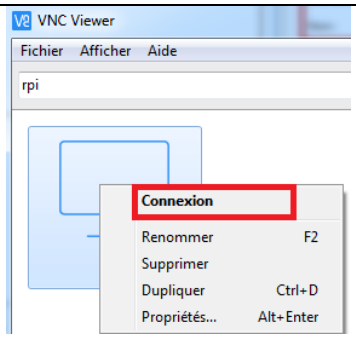
1.6.4 VNC

VNC is a system for visualizing and controlling the desktop environment of a remote computer. It allows the VNC client software to transmit keyboard and mouse input information to the remote computer with VNC server software across a computer network.https://www.realvnc.com/fr/connect/download/viewer/

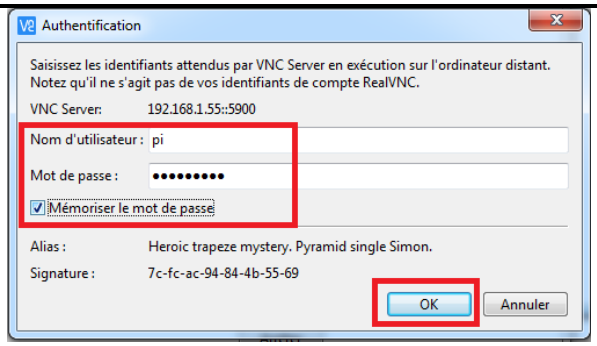
Download and install VncViewer

Téléchargez VNC Viewer sur l'appareil depuis lequel vous souhaitez exercer le contrôle à distance.

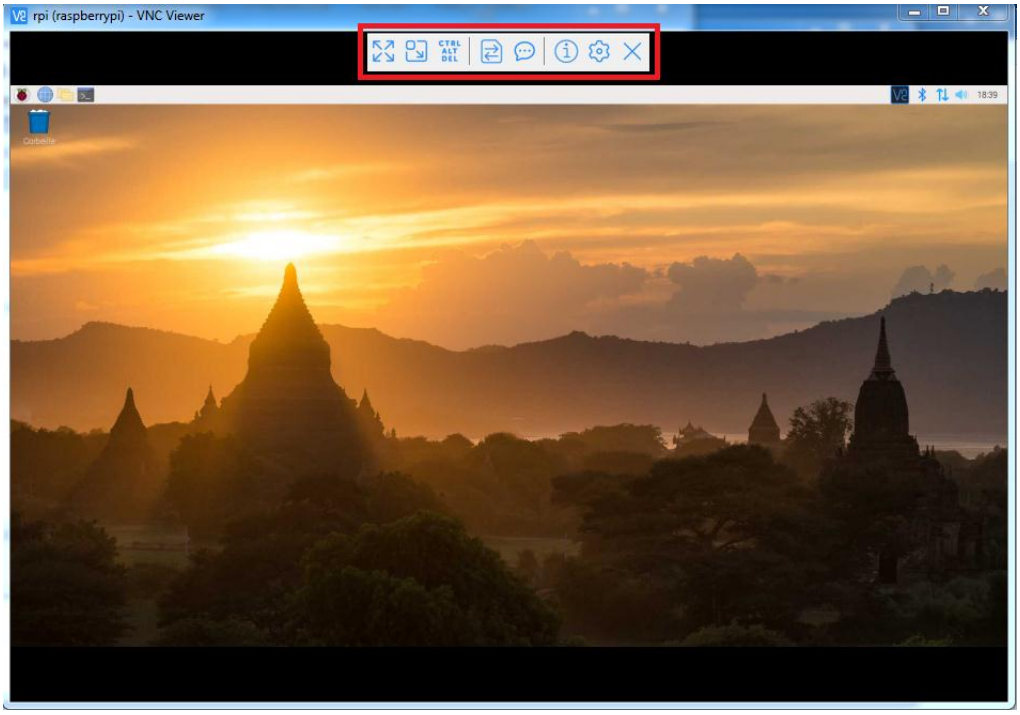
Télécharger VNC Viewer

When starting VNC Viewer, you have to create a new connection	
Fill in the IP address of the raspberry.	
Right click then, Login.	

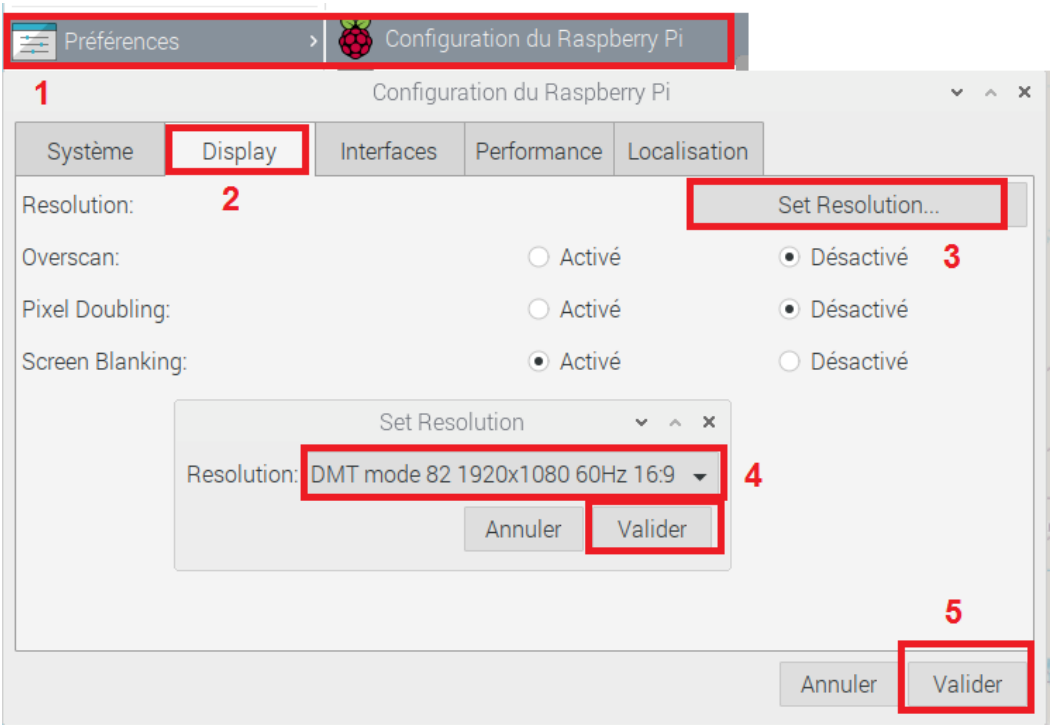
Re-enter the usual login and password.



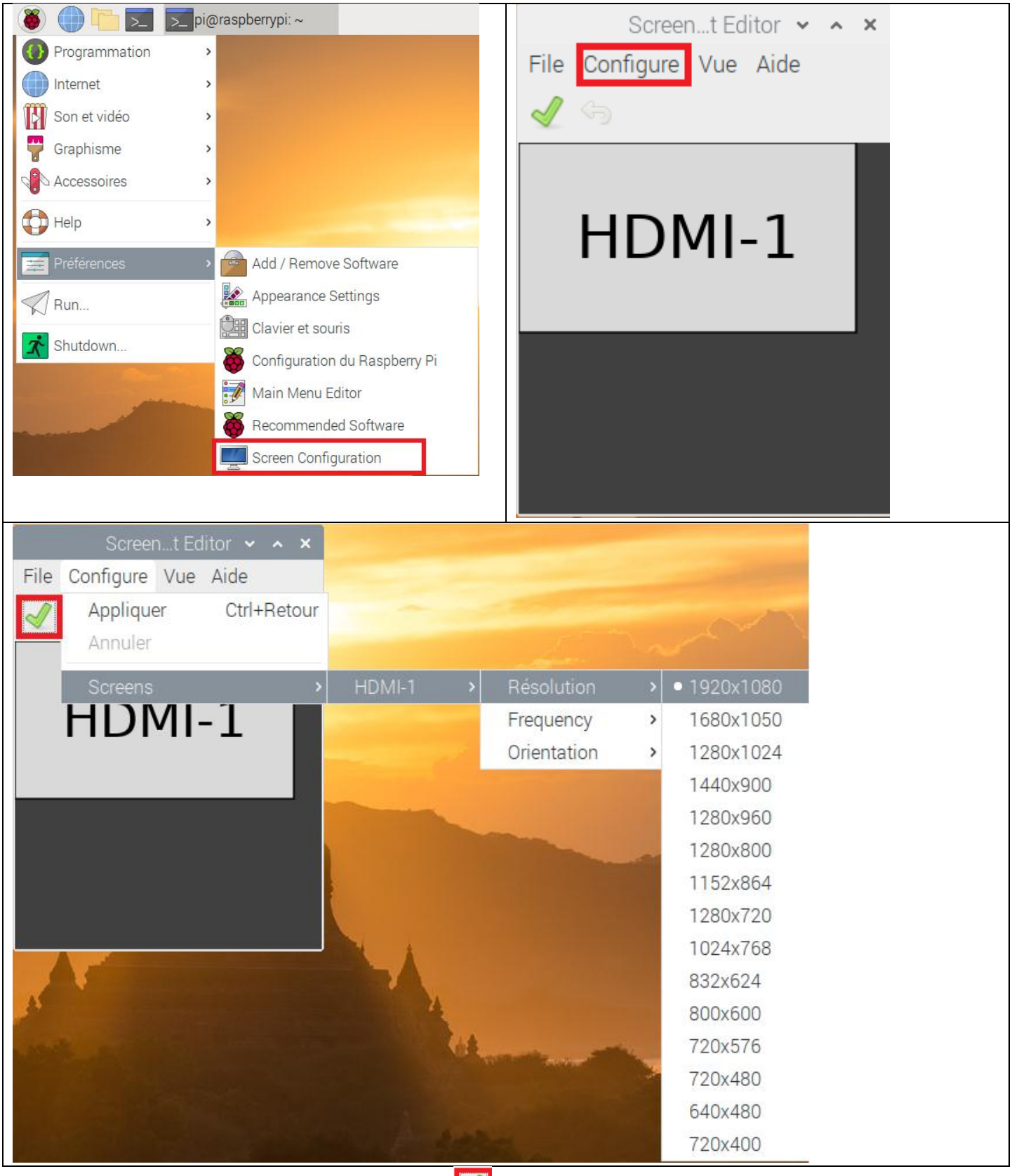
The icons at the top of the window are used to configure the VNC viewer and modify the image quality. No more need for a monitor connected to the RPI.



If necessary, change the resolution by going to the RPI configuration menu.

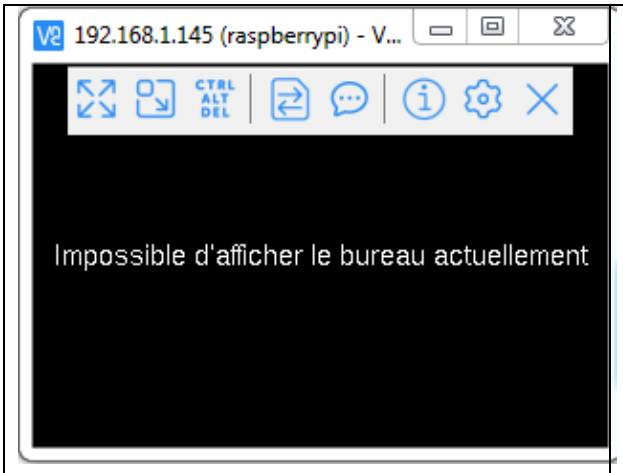


In the case of the Raspberry PI 4, the set resolution option is absent. It is then necessary to go to the Screen Configuration menu and choose its resolution.

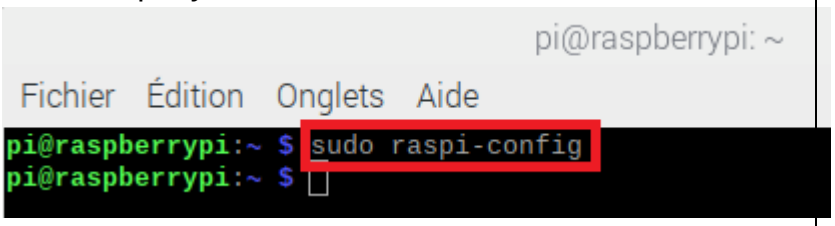


Don't forget to validate the resolution change 

However, if there is no screen connected to the Raspberry PI 4 at startup, the following message may occur with VNC viewer:



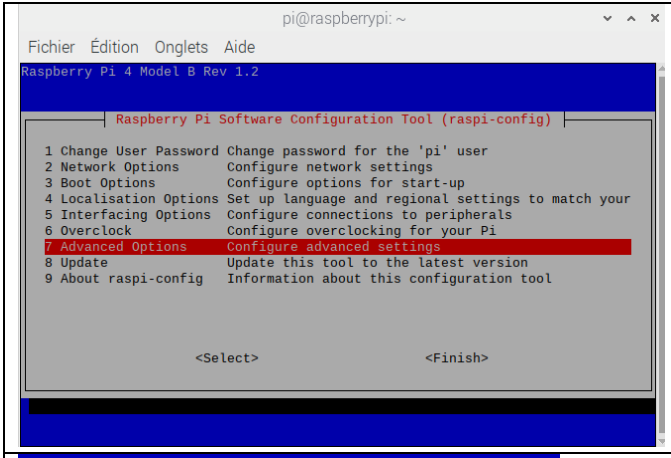
In this case, it is necessary to access the console in SSh with putty.

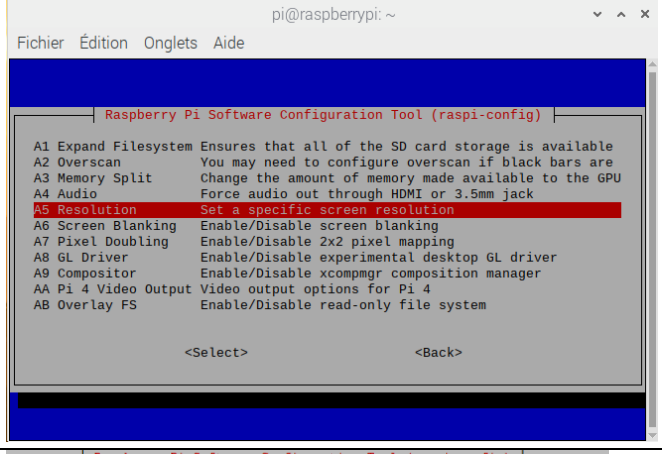


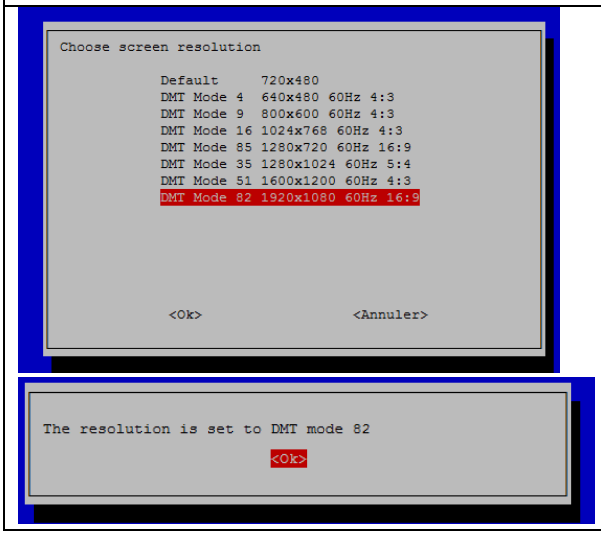
Puis taper :

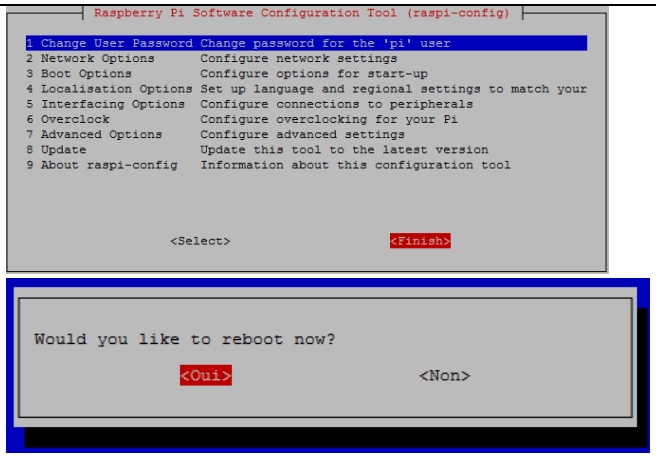
```
sudo raspi-config
```

From the Advanced Option menu, select the resolution you want.







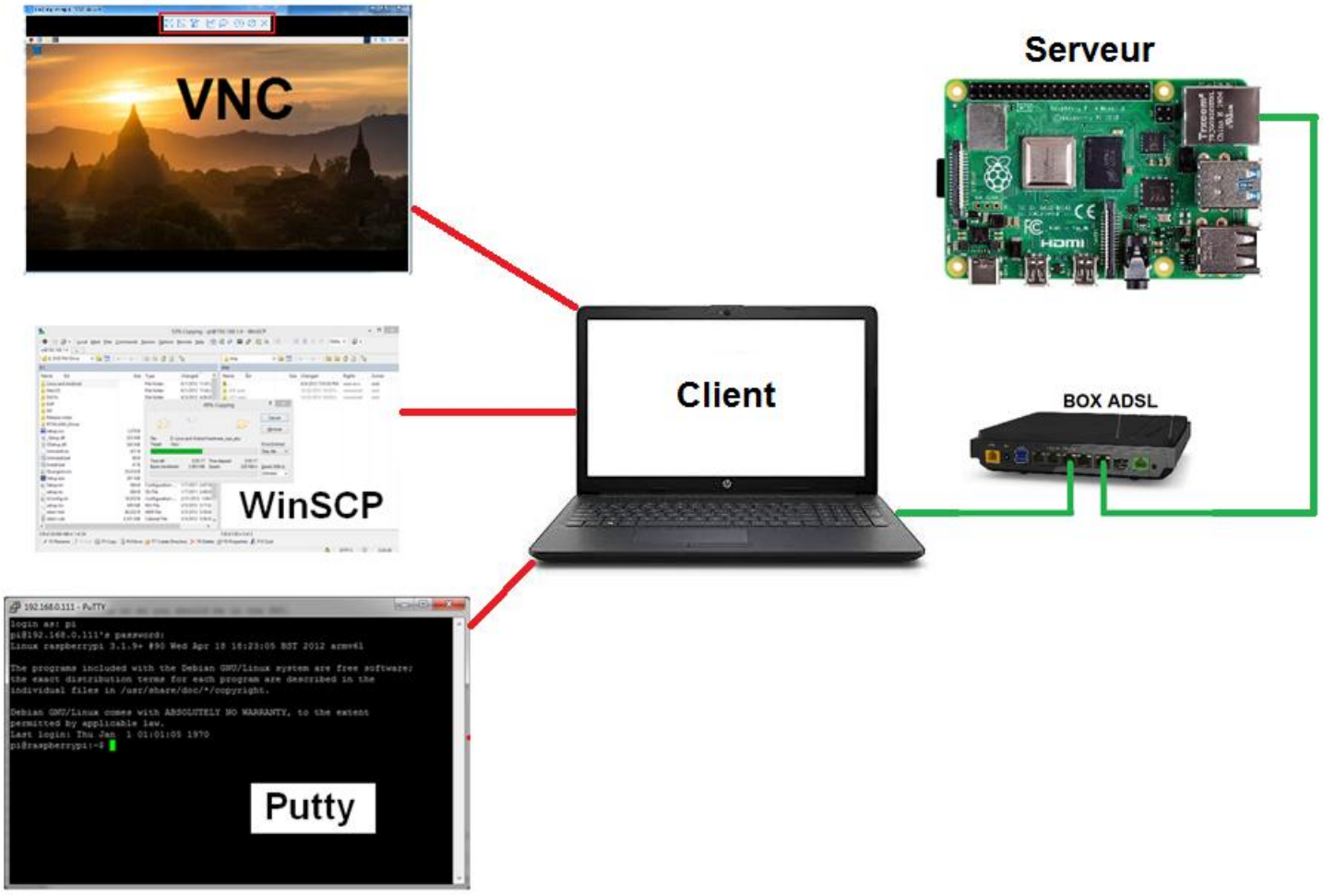


Redémarrer la Raspberry.

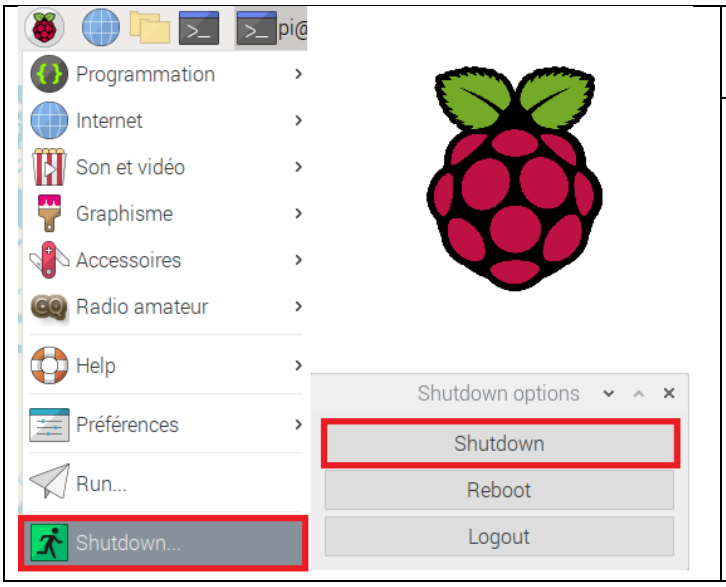
It's thanks to this utility (VNC viewer) that I work with WSJT-X installed in the RPI from the couch.

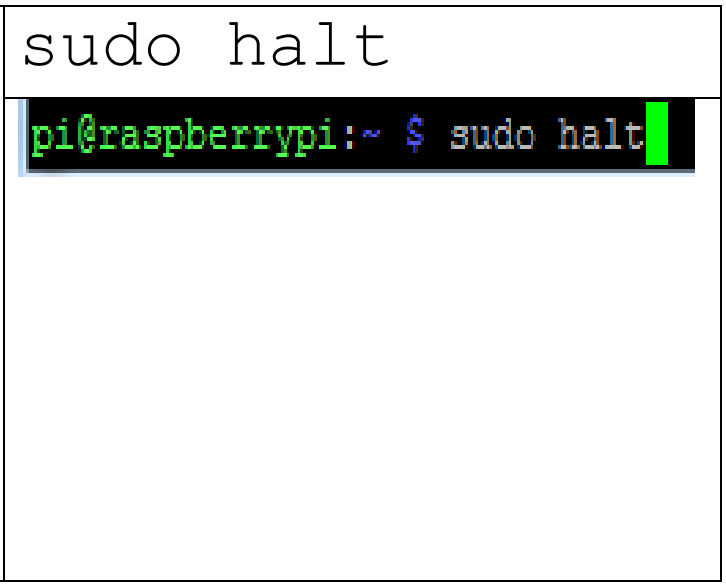
1.7 Conclusion:

It is important to be familiar with the utilities that will allow you to control the Raspberry remotely.
In part 2, we will see how to install and use WSJT-X. In the appendix on page 19 some information about the command lines.



Never turn off the RPI abruptly by coupling the power, always use the graphic menu or command line.





Appendix 1
Console: command line

The basic commands will be reviewed in part 3:

Try in order (from left to right)

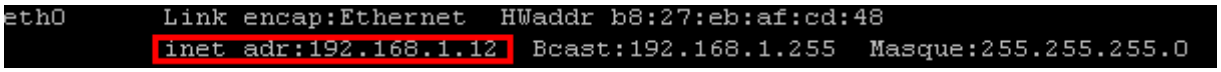
Lists the contents of the folder	Create a trial folder
`ls -l`	**`mkdir essai`**
Change folder (goes to the “essai” folder)	Going up from a folder
`cd essai`	**`cd ..`**
Creates a text file	Type a random text CTRL + O save file CTRL + X quit nano
`nano test`	
Displays the content of the test file	Lists again the contents of the folder
`cat test`	**`ls -l`**
Changes the access rights of the file	Lists again the contents of the folder
`chmod 777 test`	**`ls -l`**

Before chmod
`-rw-r--r-- 1 pi pi 11 mars 30 16:05 test`
After Chmod
`-rwxrwxrwx 1 pi pi 11 mars 30 16:05 test`	

Delete a file	Lists again the contents of the folder
`rm test`	**`ls -l`**

You can press the up arrow at any time to avoid retyping the command.

The RPI console, as on most Unix systems, offers auto-completion, i.e. with the TAB key you can complete what you type, this is very useful to navigate through the tree structure or to avoid typos!

Displays available memory	Displays the available space on the disk in bytes.s
free	**df /**
Displays connected USB devices	
lsusb	
Displays the ip address of the Raspberry PI	
ifconfig **ip addr**  	
Displays file/folder type	Ping command
Sort essai	**ping 192.168.1.1**
Super user command, grants administrator rights.	Command advanced packaging tool, for software installation.
sudo	**apt-get**
Displays the manual of the package management command, you can at any time display the manual of the different previous commands.	
man apt-get **man ls** **man chmod** etc..	

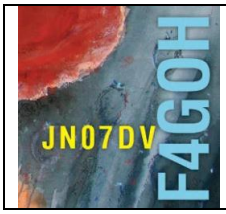
Appendix 2

Few models of Raspberry PI:

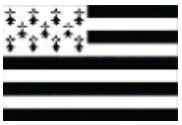
Model	GPIO	Ethernet	WIFI	Bluetooth	USB2	USB3	RAM	CPU Clock
Pi-B	26	O	N	N	2	N	512MB	700 MHz
Pi-2B	40	O	N	N	4	N	1GB	900 MHz
Pi-3B	40	O	2.4 Ghz	O	4	N	1GB	1.2 Ghz
Pi-4B	40	Gigabits	2.4+5 Ghz	O	2	2	4GB(max)	1.5 Ghz
Pi-Zero-W	40	N	2.4 GHz	O	1	N	512MB	1 Ghz

Raspberry PI consumption in mA:

Model	Main board	HDMI & Ethernet	HDMI & Wifi	Max current available on USB ports
Pi-B	500	nc	nc	500
Pi-2B	180	290	nc	600 à 1200
Pi-3B+	390	490	520	1200
Pi-4B	560	680	710	1200
Pi-Zero-W	160	nc	180	Depending on the power supply



Raspberry Pi for HAM Radio
Part 2



Prerequisite: HAM Radio tutorial with the Raspberry Pi Part 1

Summary:

Part 2 : Traffic software Installation FLDIGI, WSJT-X etc.

Version du 6/03/2021 V1.2

2.1 Introduction.

In this description, I tried to be as clear as possible with as many screenshots as possible. There are different possibilities to install a software on a Linux environment. For each radio utility installed, I chose what seems to me the easiest, limiting as much as possible the command lines for beginners.

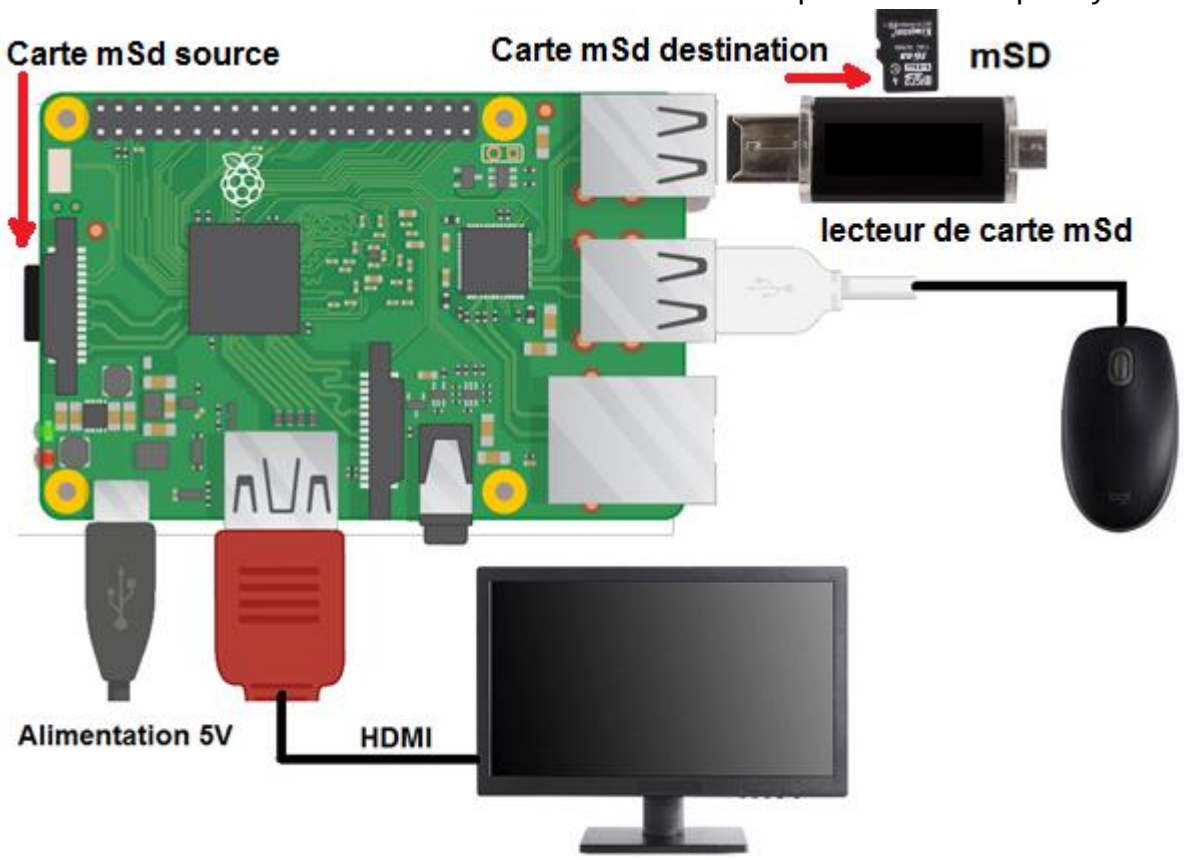
As far as possible, I will try to keep this description up to date once a year, because the evolution of the Raspberry and the operating system requires constant modifications.

Installation always takes some time. Before any modifications or additions of programs, remember to save the mSD card so that you can go back very easily. Also think about the ferrite cores on the connection cables around the Raspberry Pi.

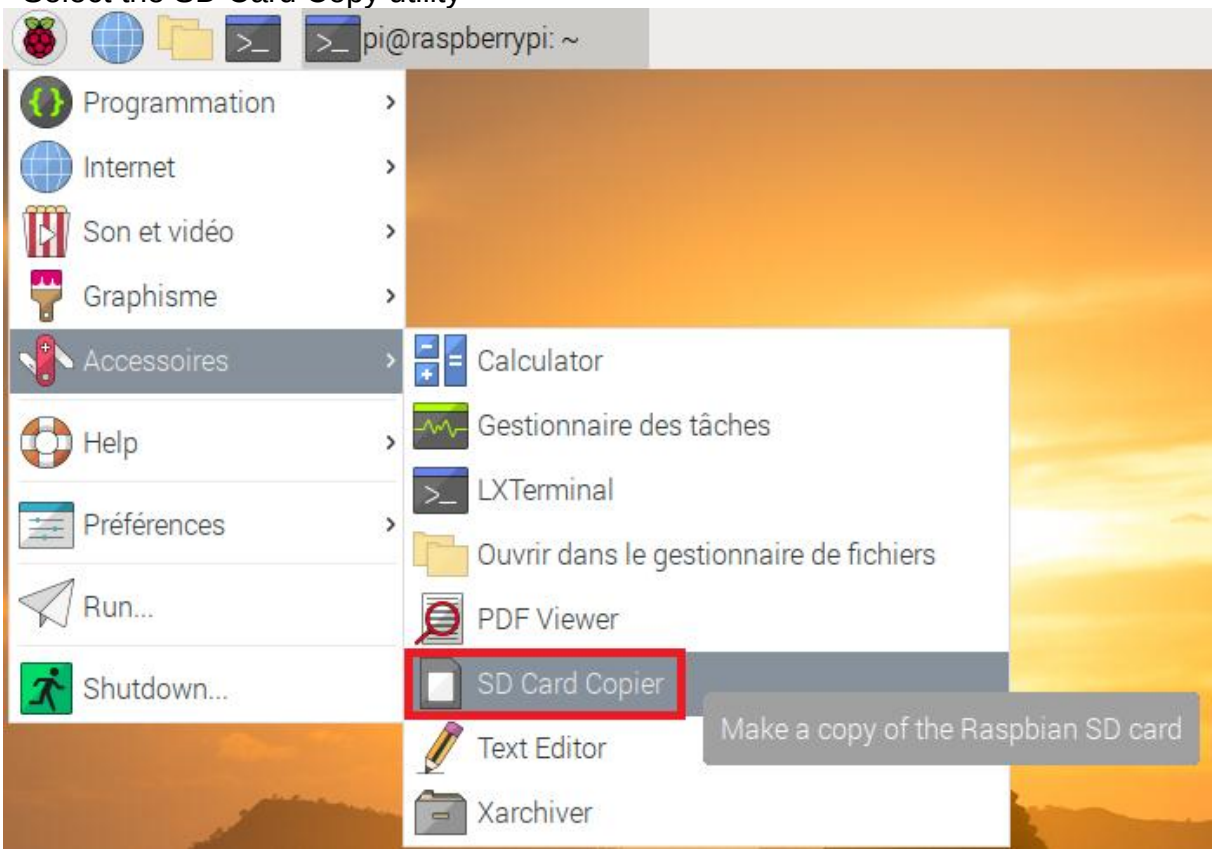
Having more and more free Open Source Radio utilities is really in the Radio Amateur spirit. I hope this document will make other OMs want to make detailed explanatory tutorials and not just a briefing note listing incomplete command lines in GitHub or in a blog.

2.2 Sauvegarde de la carte mSD.

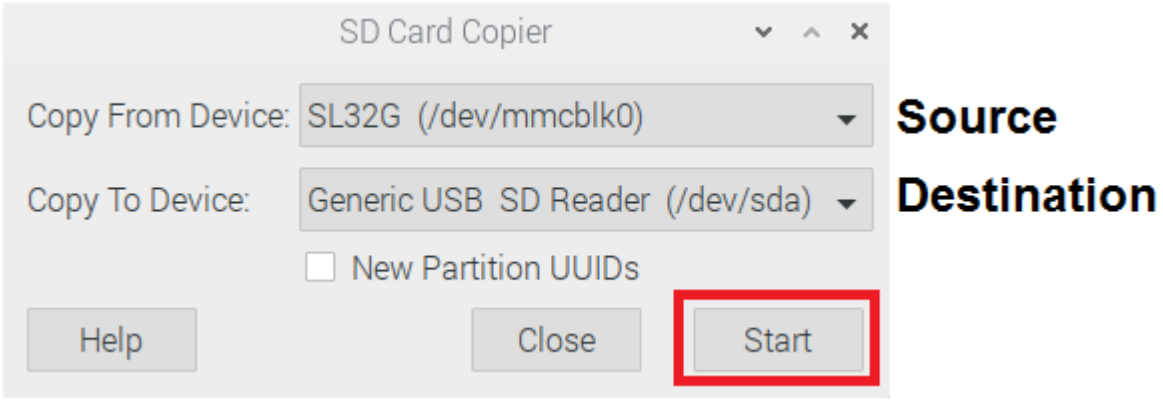
Take advantage of a break or lunch to make a backup of the operating system (This operation can be done at any time)
- Insert a reader with its blank mSD card into a free USB port on the Raspberry PI.

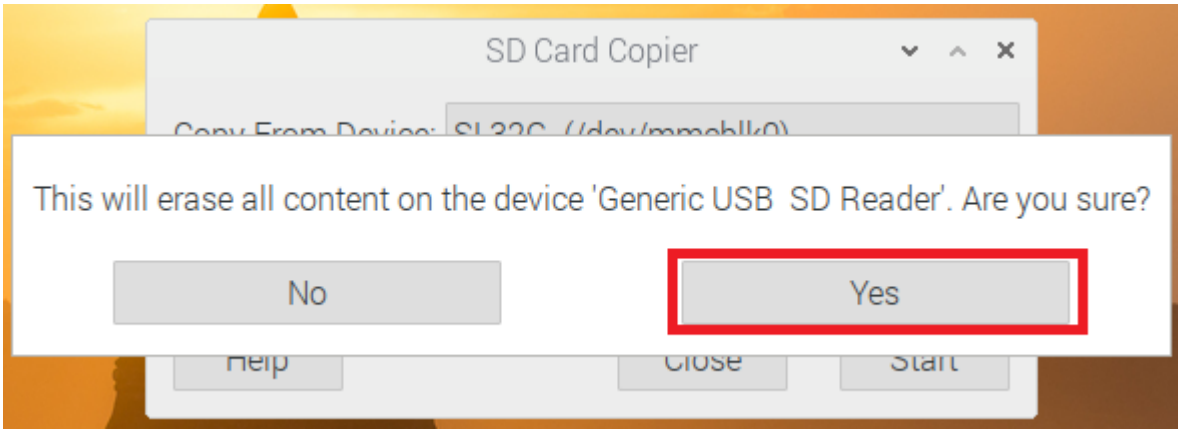


- Select the SD Card Copy utility



Select the source and destination, then start the copy.





2.3 Update repositories

The **`sudo apt update`** command updates the list of files available in the APT repositories present in the configuration file /etc/apt/sources.list. Running this command regularly is a good practice, in order to keep your list of available packages up to date and hope to have the latest software versions. The same goes for the upgrade command to update obsolete packages.

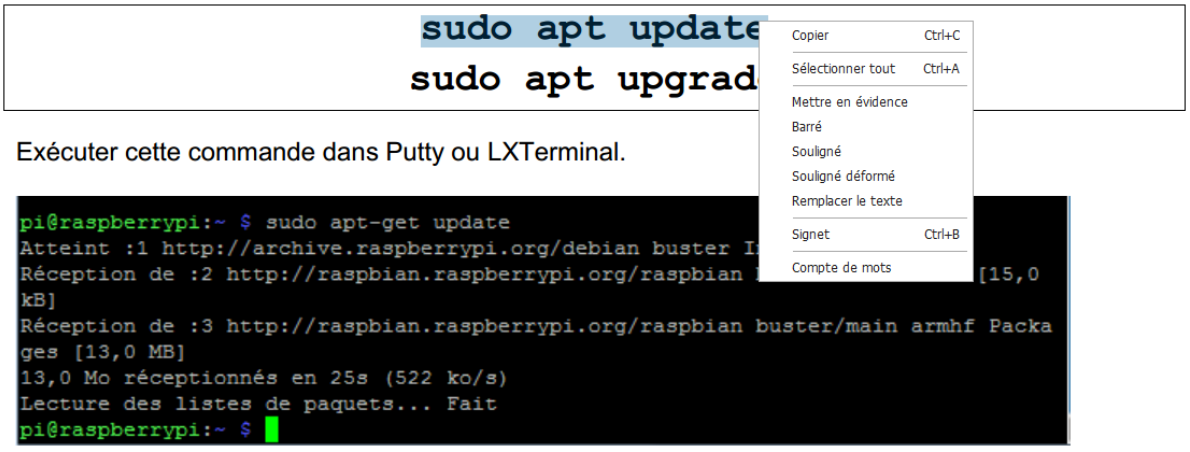
```
sudo apt update
sudo apt upgrade
```

Execute this command in Putty or LXTerminal.

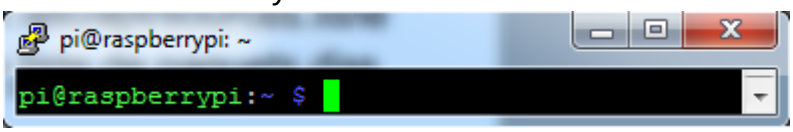
```
pi@raspberrypi:~ $ sudo apt-get update
Atteint :1 http://archive.raspberrypi.org/debian buster InRelease
Réception de :2 http://raspbian.raspberrypi.org/raspbian buster InRelease [15,0
kB]
Réception de :3 http://raspbian.raspberrypi.org/raspbian buster/main armhf Packa
ges [13,0 MB]
13,0 Mo réceptionnés en 25s (522 ko/s)
Lecture des listes de paquets... Fait
pi@raspberrypi:~ $
```

Note: It is possible to copy-paste the sudo apt update command from this tutorial to Putty without retyping it manually in the console.

- Select the command with the mouse, then right-click Copy or CTRL+C



- Back to Putty



- A simple right click with the mouse and the command appears.

```
pi@raspberrypi:~ $ sudo apt update
```

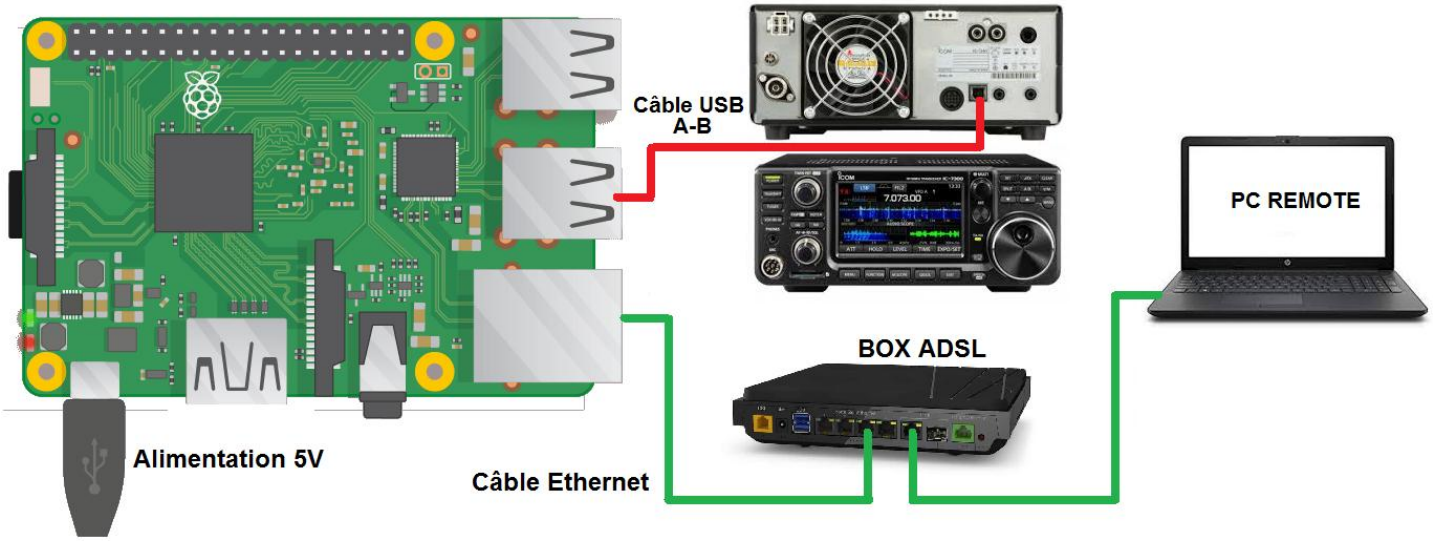
- All you have to do is press the enter key on the keyboard to execute the command.

2.4 Transceiver connection

I can't mention all the possibilities of connection with TRX. I will only retain two cables, one with an ICOM7300 and the other with an Elecraft KX3.

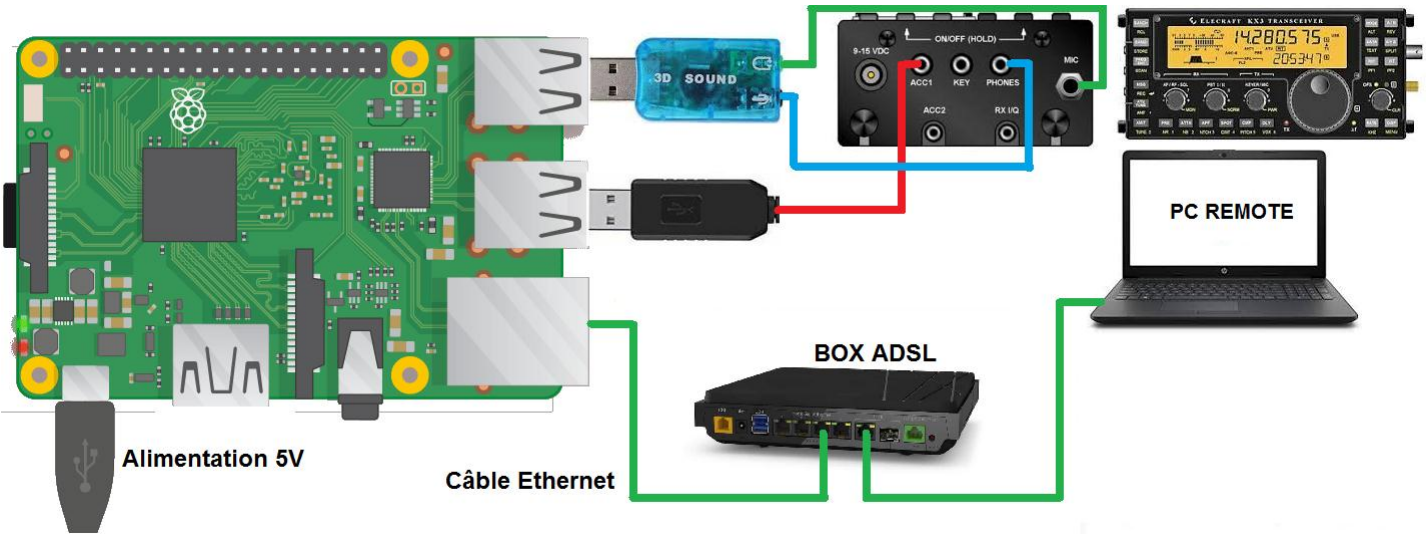
2.4.1 RPI with an ICOM 7300

The TRX has a USB connector allowing its total management with a PC. The connection is made with a simple USB cable type A-B. This is the easiest configuration. It is possible to insert a USB Isolator (galvanic isolation) between the TRX and the Raspberry PI.



2.4.2 RPI with an Elecraft KX3

As the Raspberry PI does not have a sound input, you will need to add an external USB sound card to connect the audio inputs/outputs of the TRX. It will also be necessary to connect the USB / RS232 cable reserved for the CAT-System.



I deliberately used a cheap sound card. But there are other more powerful models. It is also possible to use a USB signalink style box instead of the USB sound card.

2.5 Connect a USB device to the Raspberry PI

It is interesting to know the exact name of the USB devices you are using in order to configure the radio software correctly.

- What happens when you connect a USB device to the Raspberry PI?
- Visually nothing at all (Except for USB memory sticks).
- So how do you know if the Raspberry PI has detected something?
- It's simple, use the `lsusb` command

`lsusb`

This command lists all USB devices connected to the Raspberry PI.

Examples (on the Raspberry Pi 3) :

No external devices are connected to the Raspberry.

```
pi@raspberrypi:~ $ lsusb
Bus 001 Device 003: ID 0424:ec00 Standard Microsystems Corp. SMSC9512/9514 Fast Ethernet Adapter
Bus 001 Device 002: ID 0424:9514 Standard Microsystems Corp. SMC9514 Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

A keyboard and a mouse are connected to the Raspberry PI (first 2 lines)

```
pi@raspberrypi:~ $ lsusb
Bus 001 Device 004: ID 413c:2106 Dell Computer Corp. Dell QuietKey Keyboard
Bus 001 Device 005: ID 093a:2510 Pixart Imaging, Inc. Optical Mouse
Bus 001 Device 003: ID 0424:ec00 Standard Microsystems Corp. SMSC9512/9514 Fast Ethernet Adapter
Bus 001 Device 002: ID 0424:9514 Standard Microsystems Corp. SMC9514 Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

The ICOM 7300 transceiver is connected, we see the PCM2901 sound card internal to the TRX, the CAT-System CP2102 interface and its HUB. (first 3 lines)

```
pi@raspberrypi:~ $ lsusb
Bus 001 Device 008: ID 08bb:2901 Texas Instruments PCM2901 Audio Codec
Bus 001 Device 007: ID 10c4:ea60 Cygnal Integrated Products, Inc. CP2102/CP2109 UART Bridge Controller [CP210x family]
Bus 001 Device 006: ID 0451:2046 Texas Instruments, Inc. TUSB2046 Hub
Bus 001 Device 003: ID 0424:ec00 Standard Microsystems Corp. SMSC9512/9514 Fast Ethernet Adapter
Bus 001 Device 002: ID 0424:9514 Standard Microsystems Corp. SMC9514 Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
pi@raspberrypi:~ $
```

The inexpensive sound card CM108 and the CAT-System FT232 interface for the KX3 are connected.

```
pi@raspberrypi:~ $ lsusb
Bus 001 Device 009: ID 0403:6001 Future Technology Devices International, Ltd FT232 Serial (UART) IC
Bus 001 Device 010: ID 0d8c:013c C-Media Electronics, Inc. CM108 Audio Controller
Bus 001 Device 003: ID 0424:ec00 Standard Microsystems Corp. SMSC9512/9514 Fast Ethernet Adapter
Bus 001 Device 002: ID 0424:9514 Standard Microsystems Corp. SMC9514 Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

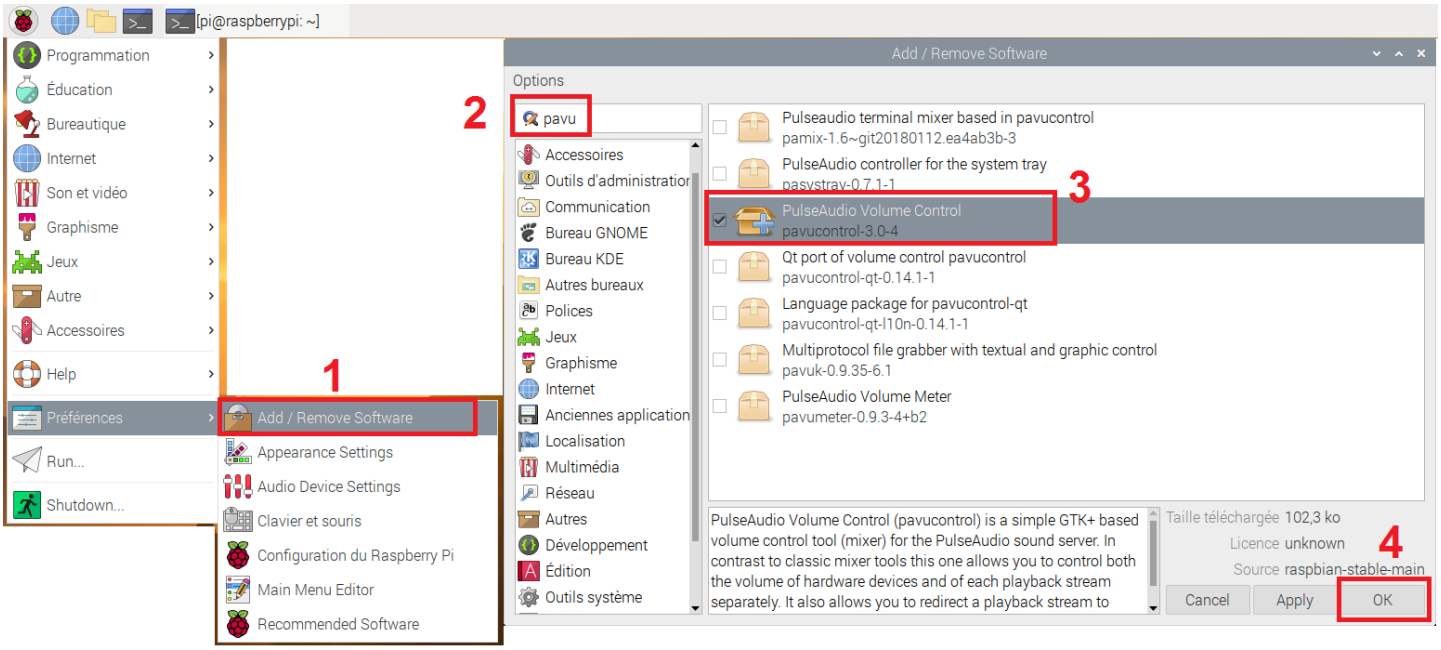
Usually the Raspberry PI already has the drivers for the most common devices. But some transceivers (especially SDR) require specific drivers (not discussed here). You should then refer to the manufacturer's website, or search with the VID (vendor identifier) and PID (product identifier). Below example with the FT232:

VID	PID
0403	**6001**

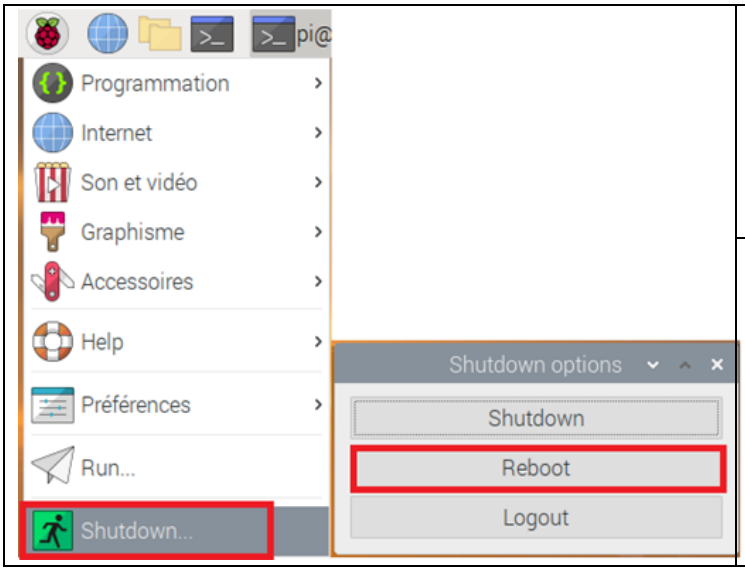
2.6 Sound card management

By default, the sound card control panel is not installed in the Raspberry PI.

Install Pulse Audio Volume Control by following steps 1 to 4.

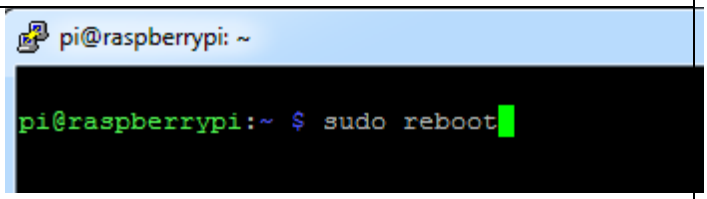


Restart the Raspberry PI (2 solutions)



Graphic or in the console

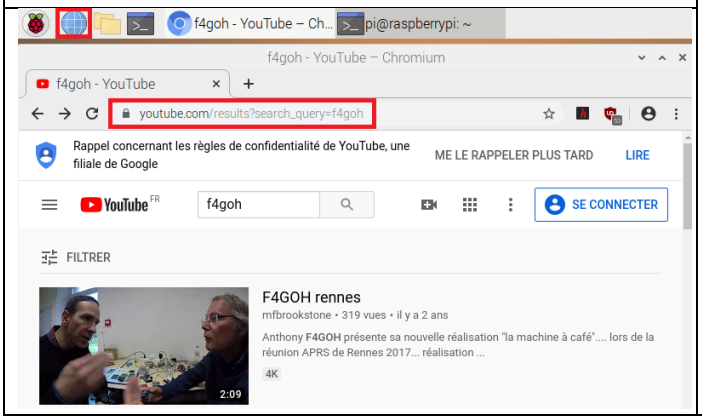
```
sudo reboot
```

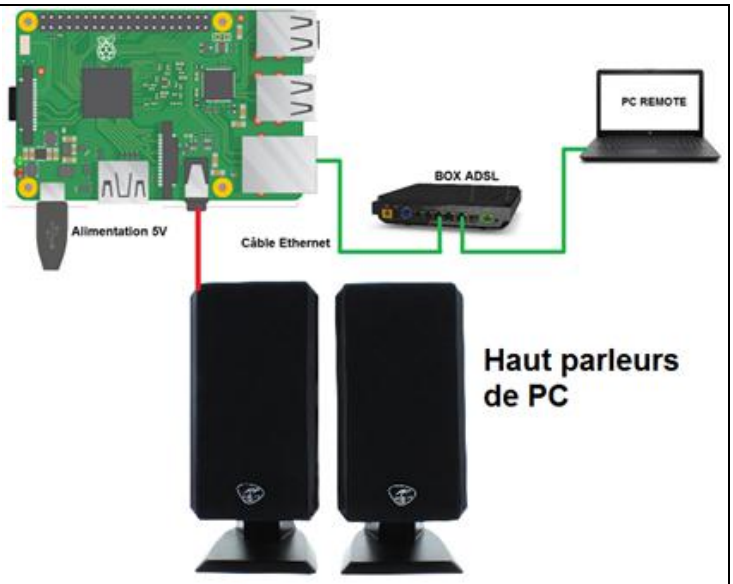


Sound output check via 3.5 jack jack

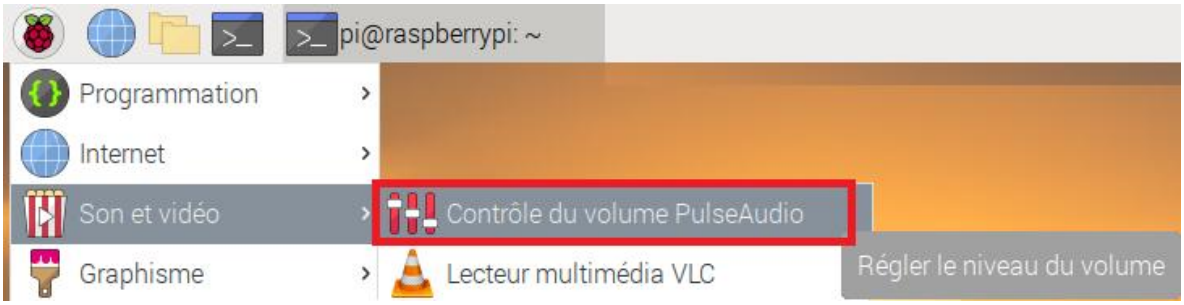
Do not connect a transceiver or external USB sound card.

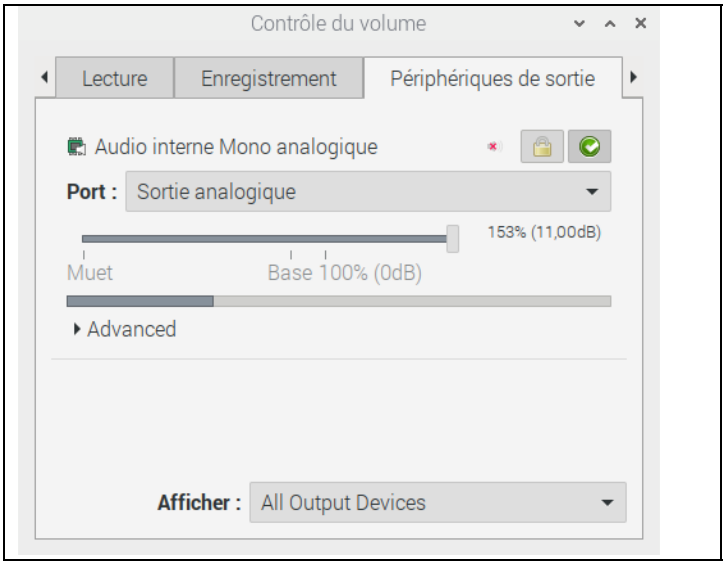
Connect an amplified speaker or headphones to the 3.5 jack output. With the Chromium browser, open a page on YouTube to play a video.

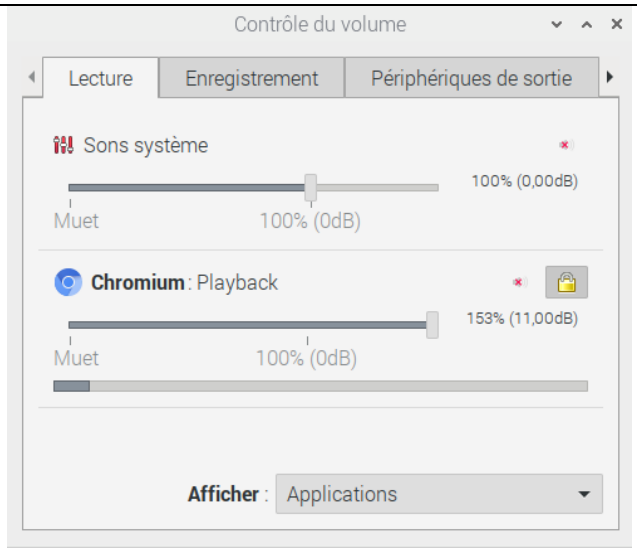




Select PulseAudio volume control







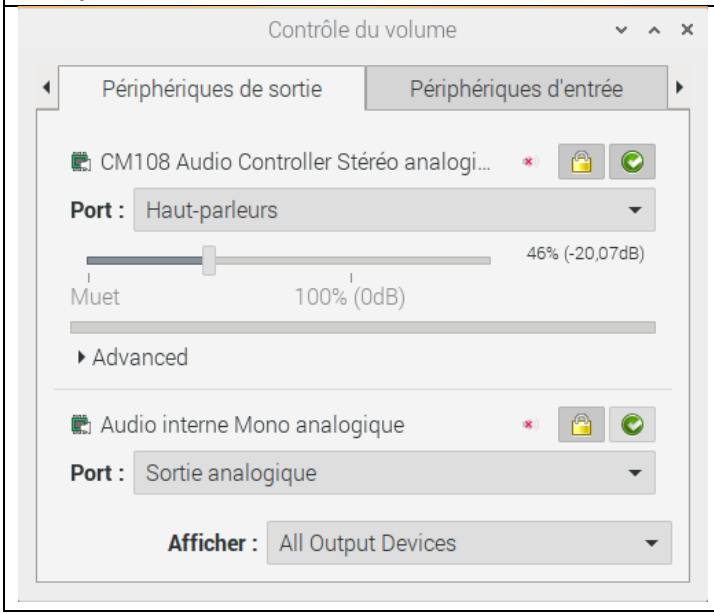
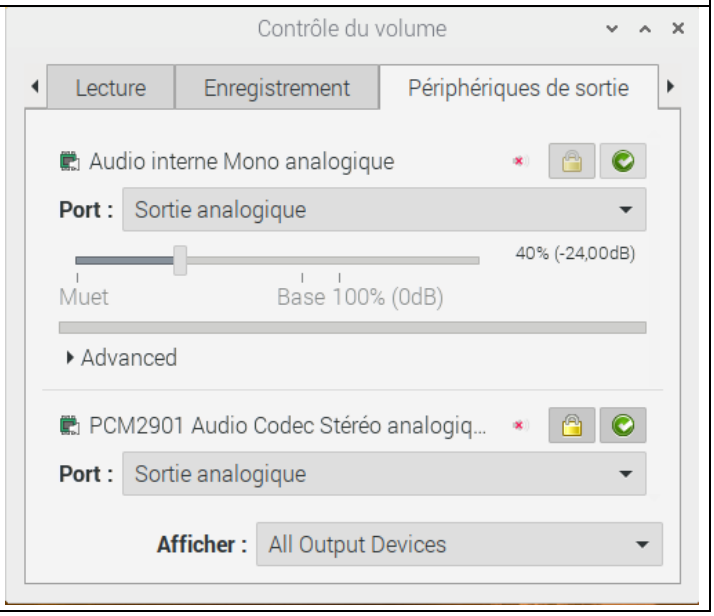
However, if there was no sound, a manual configuration must be done in order to correctly switch the sound output. Use LXTerminal or Putty and type the following command:

```
sudo amixer cset numid=3 1
```

```
pi@raspberrypi:~ $ sudo amixer cset numid=3 1
numid=3,iface=MIXER,name='PCM Playback Route'
  ; type=INTEGER,access=rw------,values=1,min=0,max=3,step=0
  : values=1
pi@raspberrypi:~ $
```

The last digit "1" (in red) corresponds to the jack 3.5 output, if you put in the place of the number "2", it corresponds to the HDMI output.

Now connect a transceiver or a USB sound card to identify its reference with PulseAudio. Locating the sound card is now easier with the USB device.

Inexpensive sound card **CM108**	IC7300 internal **PCM2901** sound card
	

2.7 CAT-System interface management

The control of a transceiver is mostly done via a serial link. The time of DB9 sockets being over, USB / RS232 adapters are now used. In the Windows operating system, access to the communication port is made by knowing the COM port number. On Linux, the following command line must be used:

```
ls /dev/tty*
```

This command lists all tty devices and is also used for serial link.

The origin of the name tty comes from the English teletypewriter.

Here is the list when no USB serial communication device is connected.
We mark /dev/ttyAMA0 (the serial link of the Raspberry PI is located on the GPIO pins of the 40-pin connector)

```
pi@raspberrypi:~ $ ls /dev/tty*
/dev/tty    /dev/tty19  /dev/tty3   /dev/tty40  /dev/tty51  /dev/tty62
/dev/tty0   /dev/tty2   /dev/tty30  /dev/tty41  /dev/tty52  /dev/tty63
/dev/tty1   /dev/tty20  /dev/tty31  /dev/tty42  /dev/tty53  /dev/tty7
/dev/tty10  /dev/tty21  /dev/tty32  /dev/tty43  /dev/tty54  /dev/tty8
/dev/tty11  /dev/tty22  /dev/tty33  /dev/tty44  /dev/tty55  /dev/tty9
/dev/tty12  /dev/tty23  /dev/tty34  /dev/tty45  /dev/tty56  /dev/ttyAMA0
/dev/tty13  /dev/tty24  /dev/tty35  /dev/tty46  /dev/tty57  /dev/ttyprintk
/dev/tty14  /dev/tty25  /dev/tty36  /dev/tty47  /dev/tty58
/dev/tty15  /dev/tty26  /dev/tty37  /dev/tty48  /dev/tty59
/dev/tty16  /dev/tty27  /dev/tty38  /dev/tty49  /dev/tty6
/dev/tty17  /dev/tty28  /dev/tty39  /dev/tty5   /dev/tty60
/dev/tty18  /dev/tty29  /dev/tty4   /dev/tty50  /dev/tty61
pi@raspberrypi:~ $
```

If the USB / RS232 cable reserved for the CAT-System is connected, a new line appears.
It is this name /dev/ttyUSB0 that you will have to use when configuring the software.

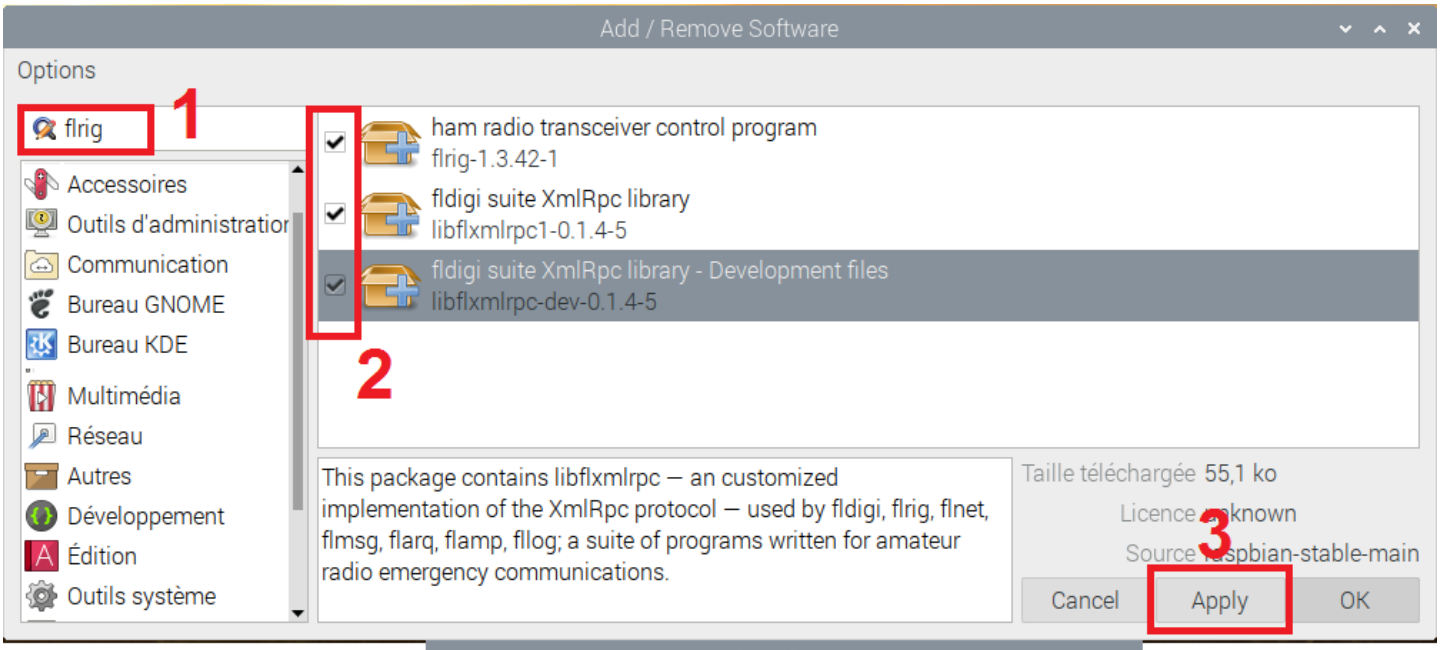
```
pi@raspberrypi:~ $ ls /dev/tty*
/dev/tty    /dev/tty19  /dev/tty3   /dev/tty40  /dev/tty51  /dev/tty62
/dev/tty0   /dev/tty2   /dev/tty30  /dev/tty41  /dev/tty52  /dev/tty63
/dev/tty1   /dev/tty20  /dev/tty31  /dev/tty42  /dev/tty53  /dev/tty7
/dev/tty10  /dev/tty21  /dev/tty32  /dev/tty43  /dev/tty54  /dev/tty8
/dev/tty11  /dev/tty22  /dev/tty33  /dev/tty44  /dev/tty55  /dev/tty9
/dev/tty12  /dev/tty23  /dev/tty34  /dev/tty45  /dev/tty56  /dev/ttyAMA0
/dev/tty13  /dev/tty24  /dev/tty35  /dev/tty46  /dev/tty57  /dev/ttyprintk
/dev/tty14  /dev/tty25  /dev/tty36  /dev/tty47  /dev/tty58  /dev/ttyUSB0
/dev/tty15  /dev/tty26  /dev/tty37  /dev/tty48  /dev/tty59
/dev/tty16  /dev/tty27  /dev/tty38  /dev/tty49  /dev/tty6
/dev/tty17  /dev/tty28  /dev/tty39  /dev/tty5   /dev/tty60
/dev/tty18  /dev/tty29  /dev/tty4   /dev/tty50  /dev/tty61
pi@raspberrypi:~ $
```

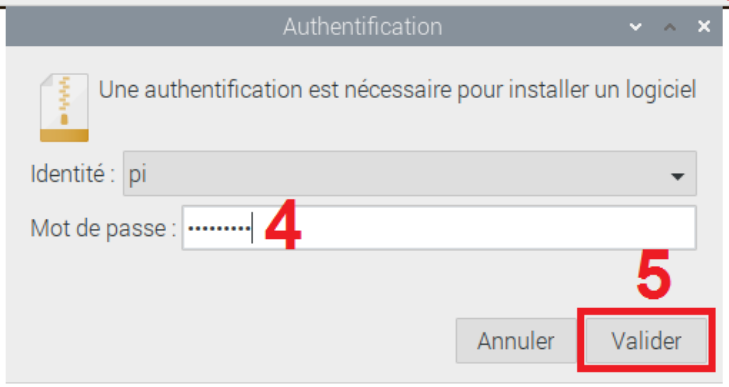
Always check the name of the device used to configure the CAT-System. Sometimes, depending on the driver, the name may be /dev/ttyACM0.

The explanations of the basic concepts are now complete. Let's move on to the installation and configuration of radio software.

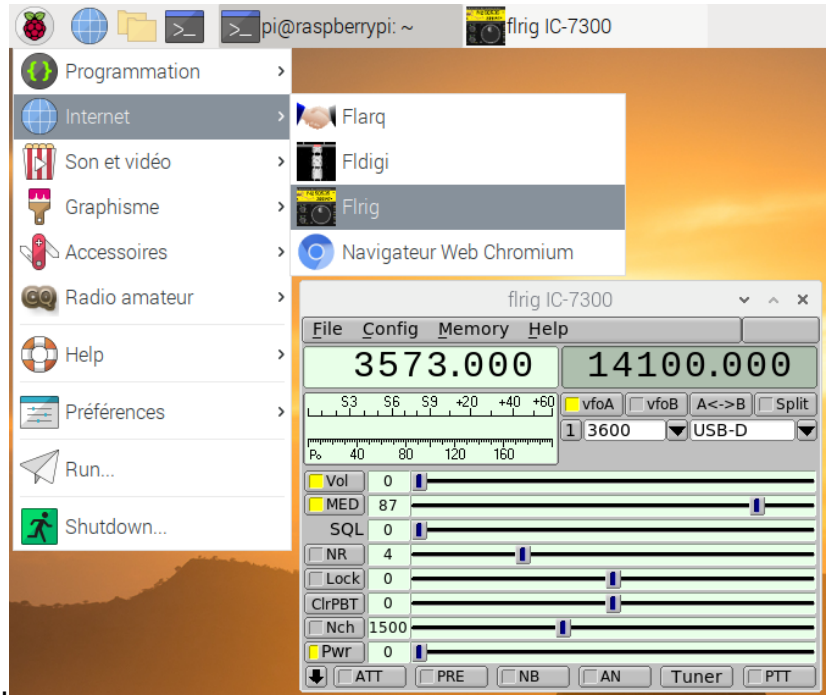
2.8 Fldigi

Start by installing flrig. Go to the Preferences Menu, Add / Remove Software

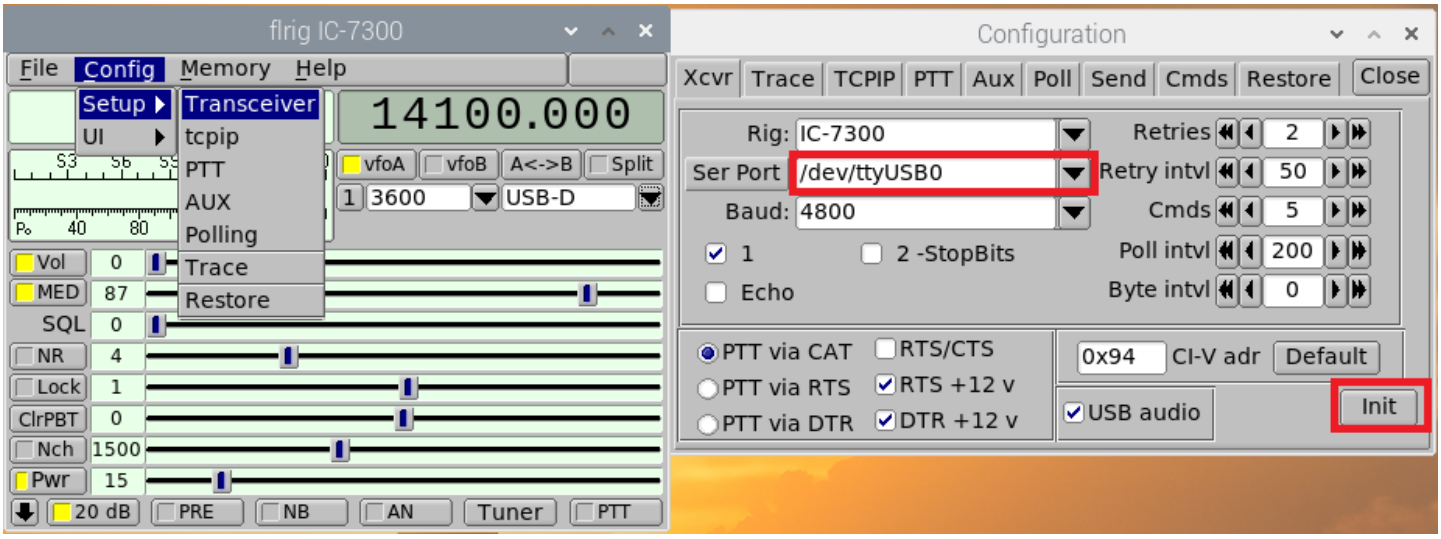




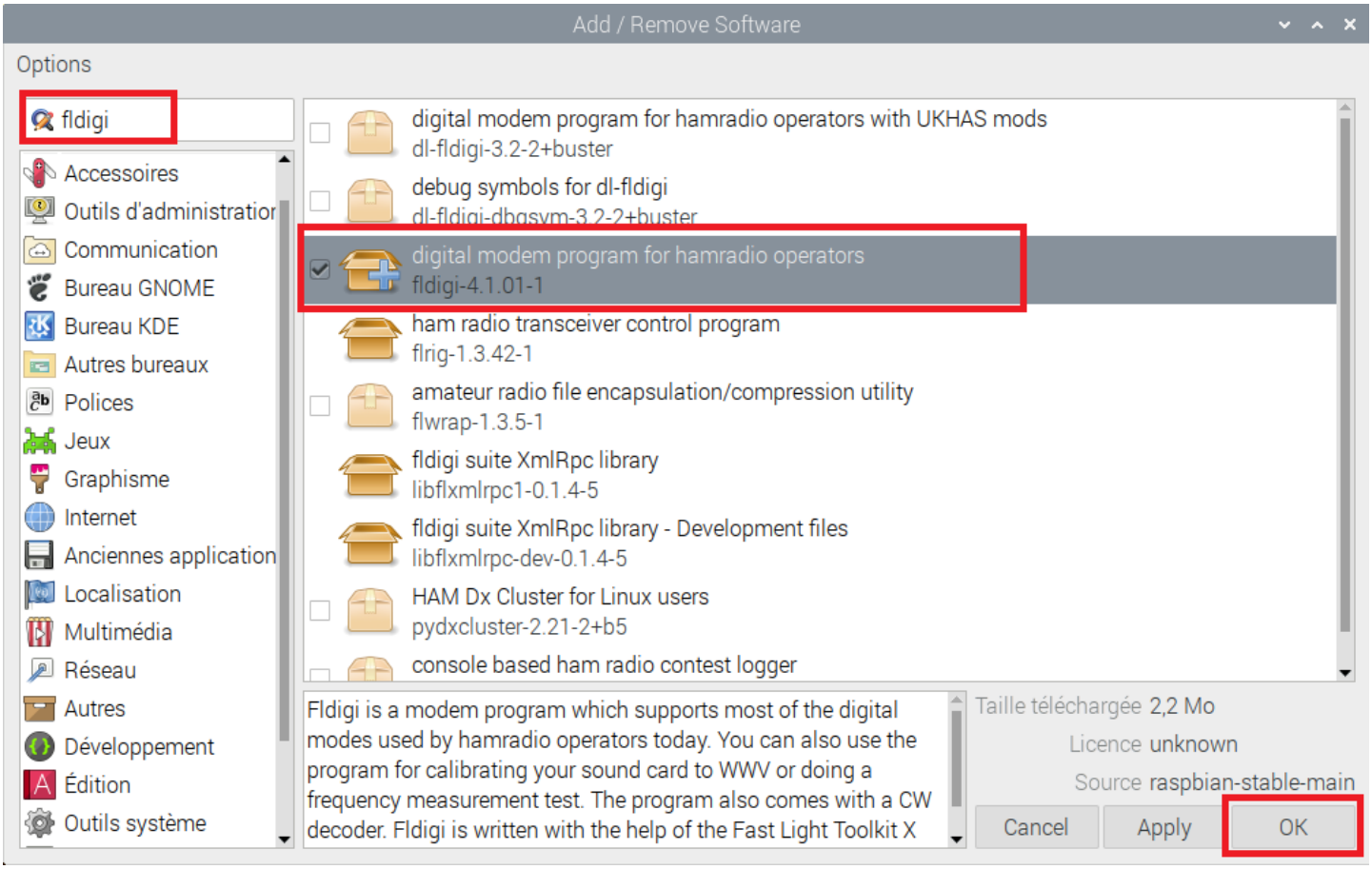
When testing the RPI4 flrig can be found in the Internet menu. The amateur radio menu will appear after fldigi installation.

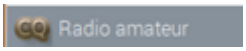


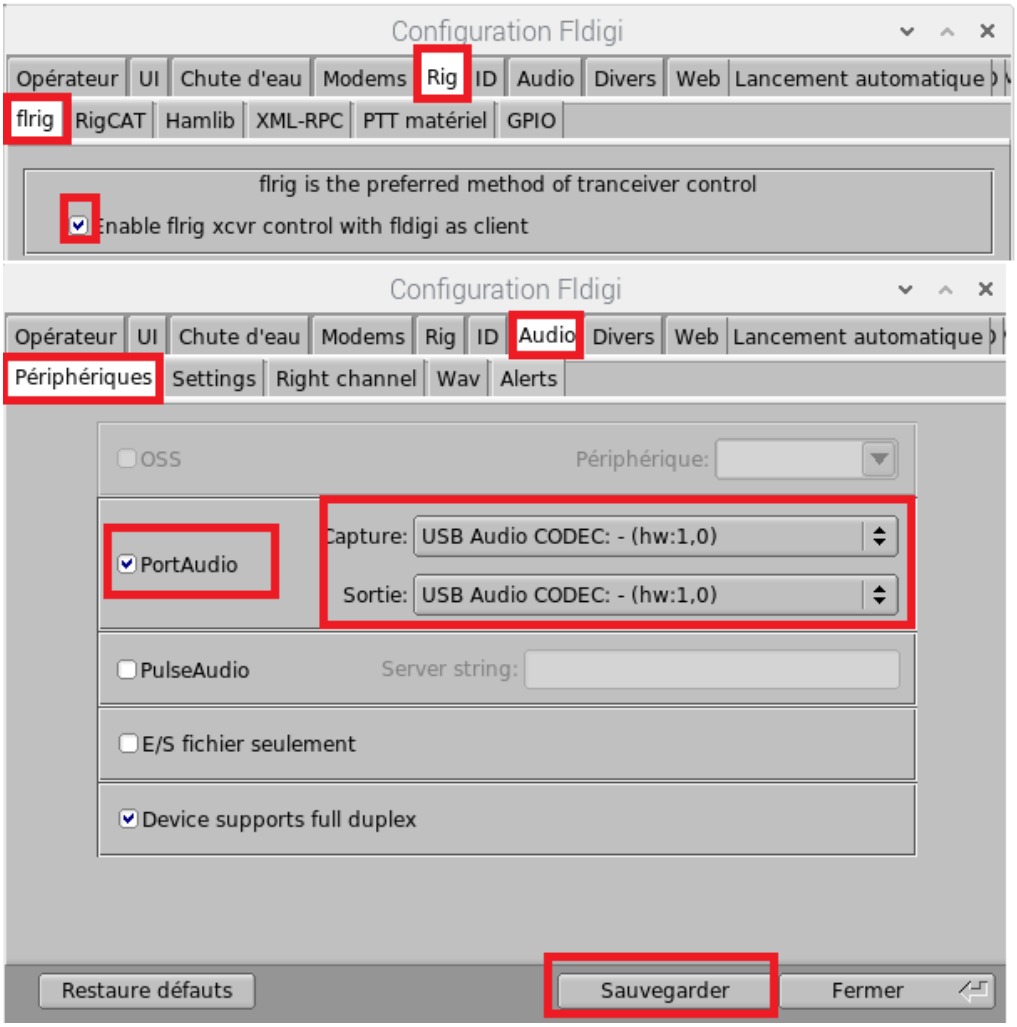
Configuration test with the IC-7300



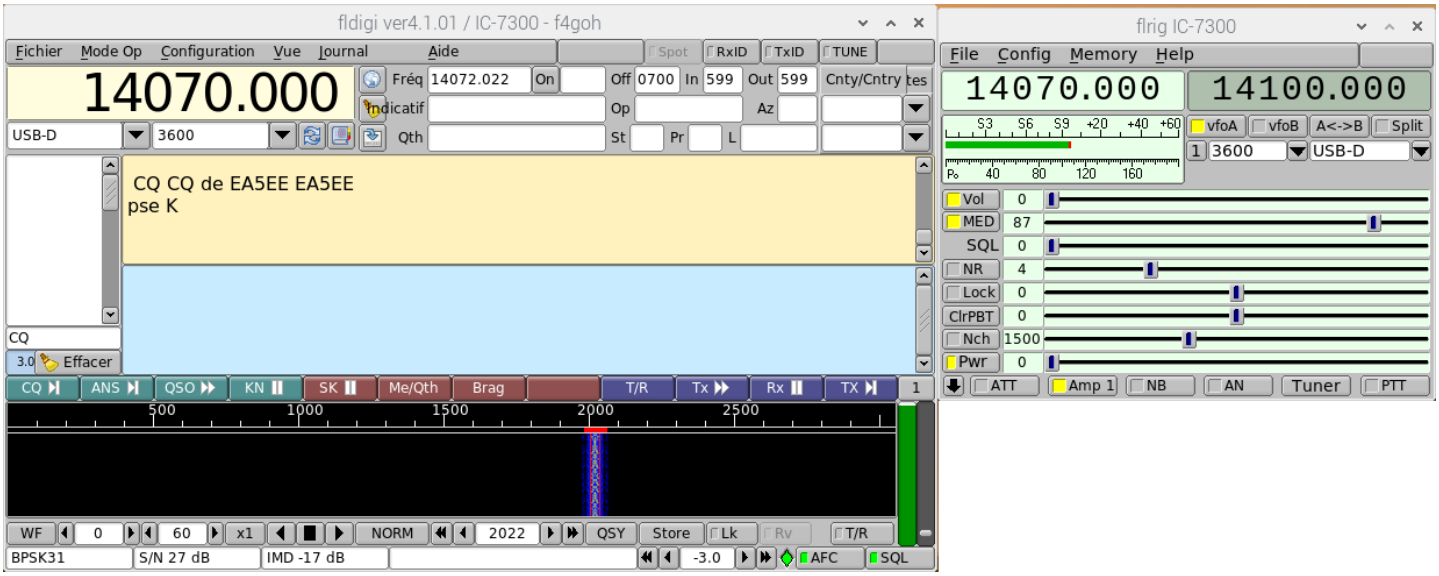
Continue by installing fldigi: Preferences Menu, Add / Remove Software



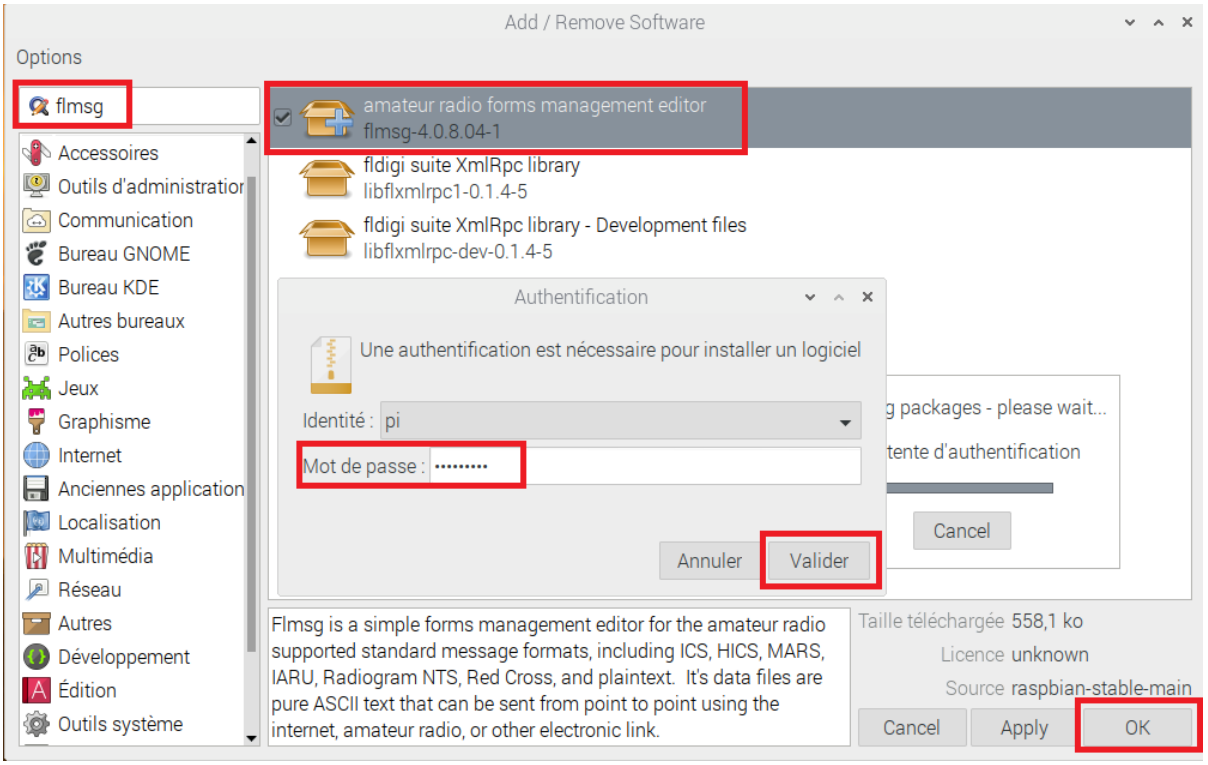
Run Fldigi  at the same time as flrig from the Amateur Radio menu 
Enable flrig and the audio port in connection with the audio card on the USB port



Check the reception e.g. in PSK31 over 20 meters.

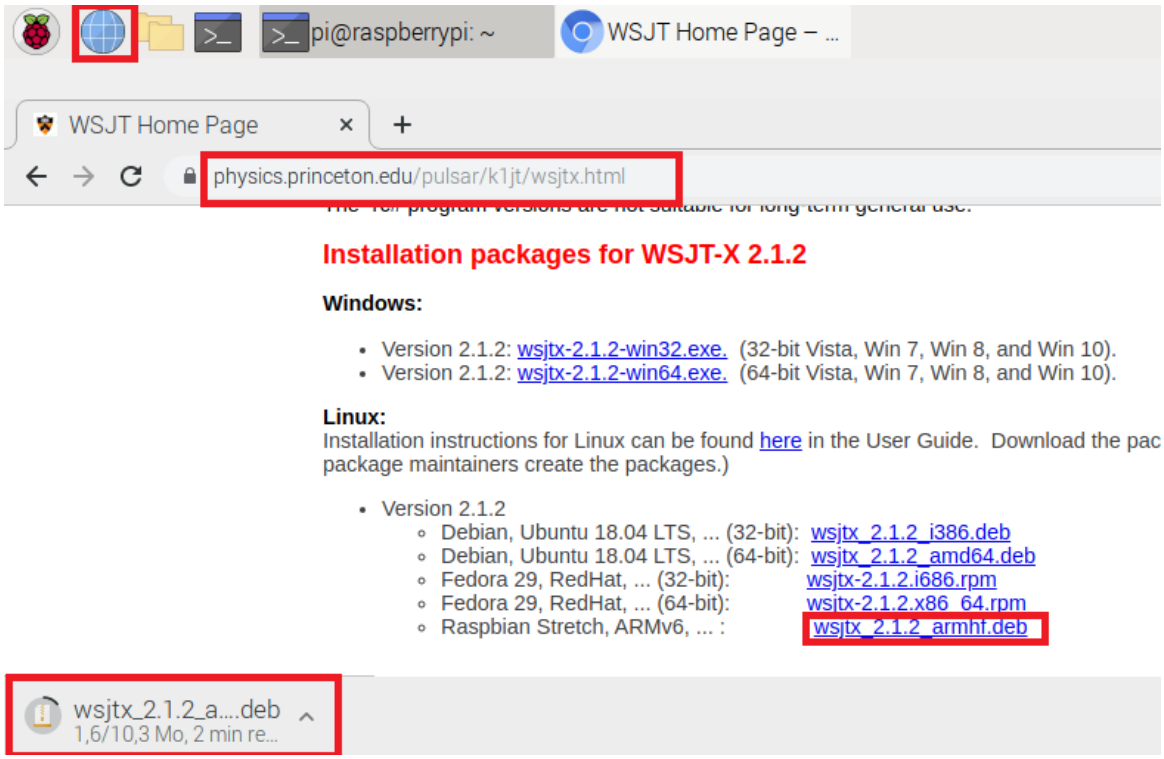


flmsg is also available and easy to install

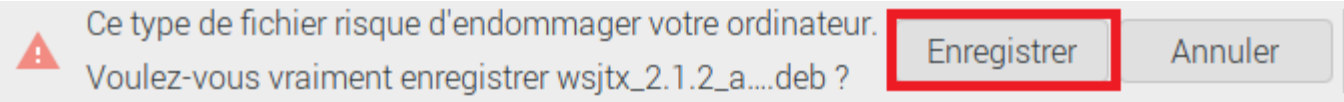


2.9 Wsjt-x

With the Chromium browser, go to the main page of wsjt-x https://physics.princeton.edu/pulsar/K1JT/wsjtx.html, then download **wsjtx_2.1.2_armhf.deb**



Please note that the version of wsjt-x is constantly evolving.

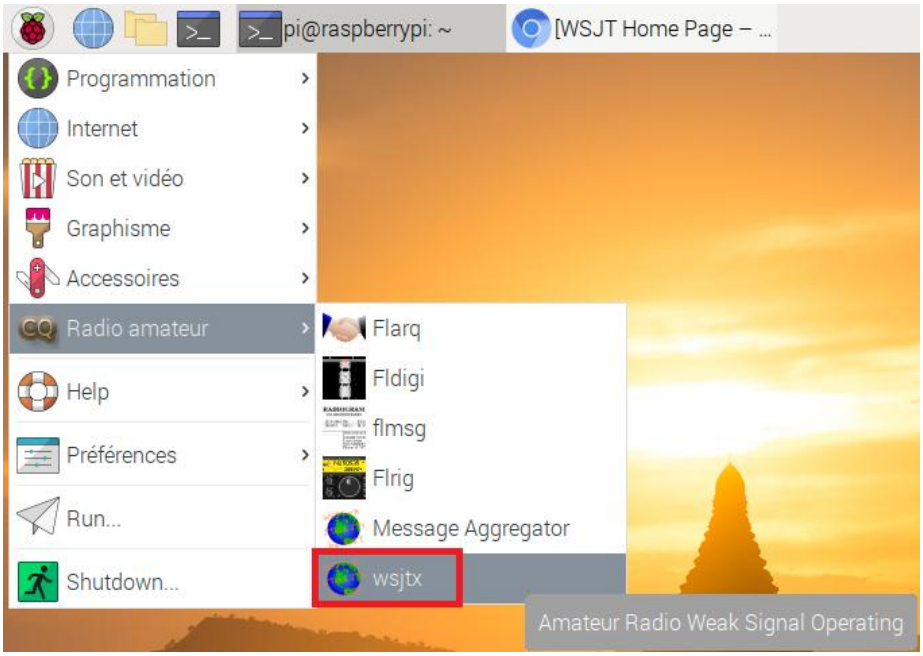


Using LXTerminal or putty, type the following commands in order

cd Downloads		pi@raspberrypi:~ $ cd Downloads/ pi@raspberrypi:~/Downloads $ Go to the Downloads directory
ls	`ls` Lima Sierra letters	pi@raspberrypi:~/Downloads $ ls wsjtx_2.1.2_armhf.deb The file is present
sudo dpkg -i wsjtx_2.1.2_armhf.deb		wsjt-x software installation
sudo apt-get install -f		Libraries are missing, this command allows you to find and install the missing files.
continue ? [O/n]		O

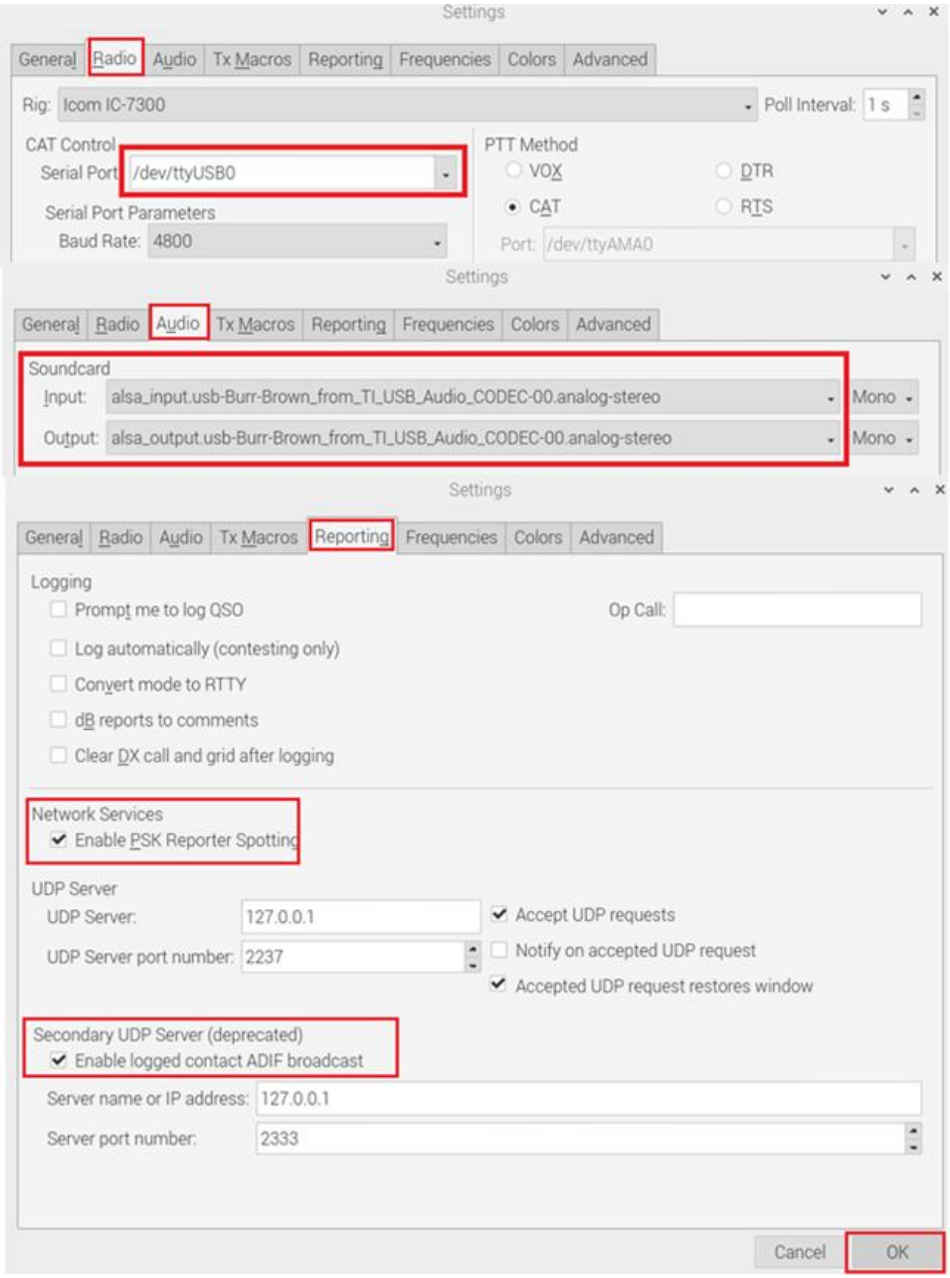
For the latest version of wsjt-x, if the error "**locale :: facet :: _ S_create_c_locale name not valid**" Go to the raspberry configuration from the command line (tutorial 4, page 5). Go to localization option, then change locale add in addition to your language the line en_US.UTF8. Use the usual language by default.

wsjtx appears in the main menu

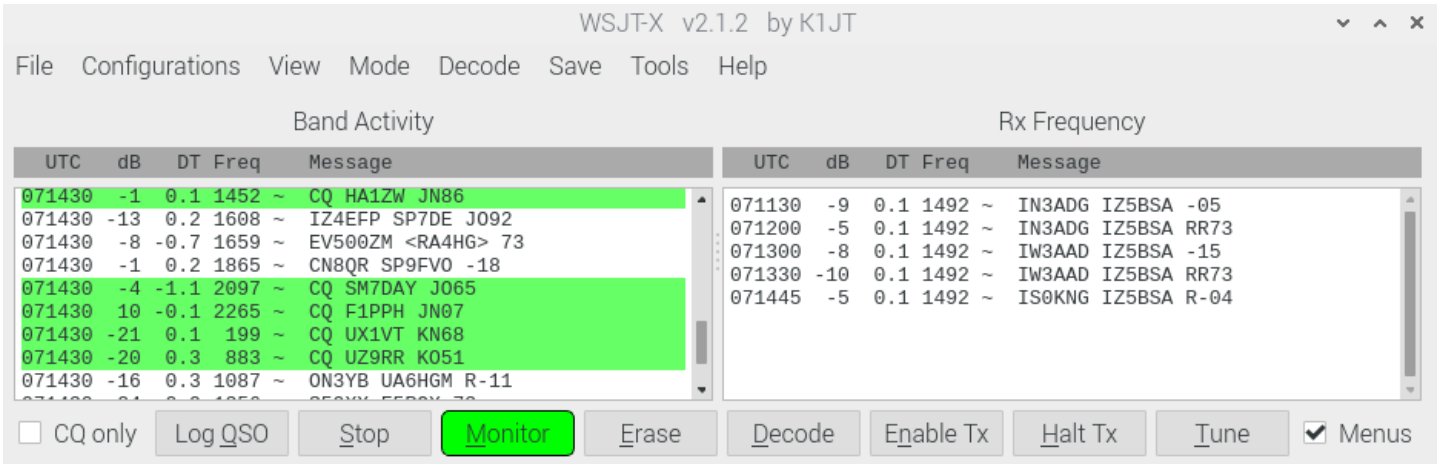


Note: If you download the file with a remote computer, it is possible to copy the file to the Raspberry PI with WinSCP. (See part 1)

You can recognize the CAT-System command on /dev/ttyUSB0, and the internal sound card of the IC7300 from Texas Instruments.

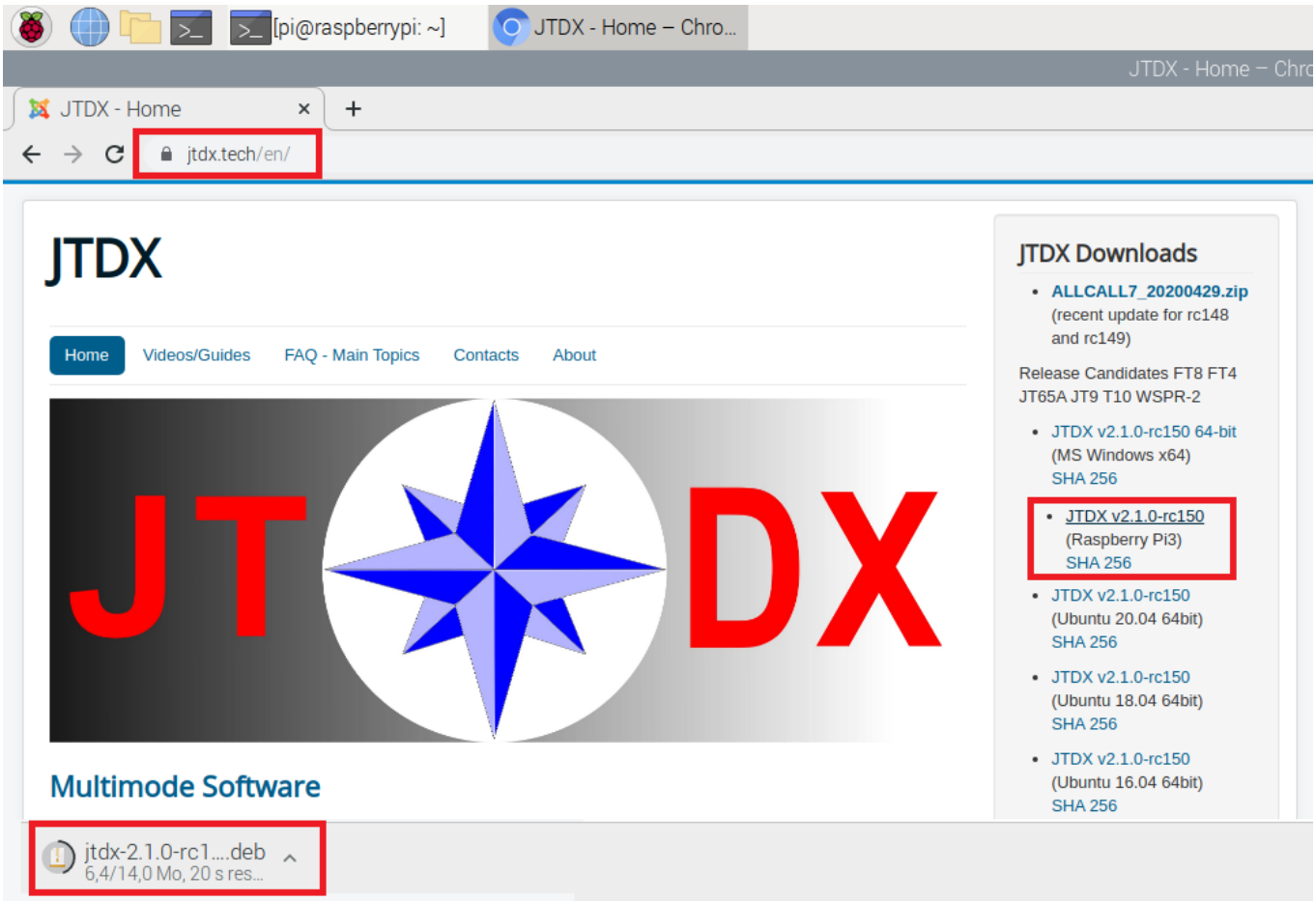


Check the reception e.g. in FT8 over 20 meters.



2.10 JTDX

With the Chromium browser, go to the main page of jtdx https://www.jtdx.tech/en/, then download **jtdx-2.1.0-rc150_u_armhf.deb**



Using LXTerminal or putty, type the following commands in order

cd Downloads		`pi@raspberrypi:~ $ cd Downloads/`
		`pi@raspberrypi:~/Downloads $`
		Go to the Downloads directory

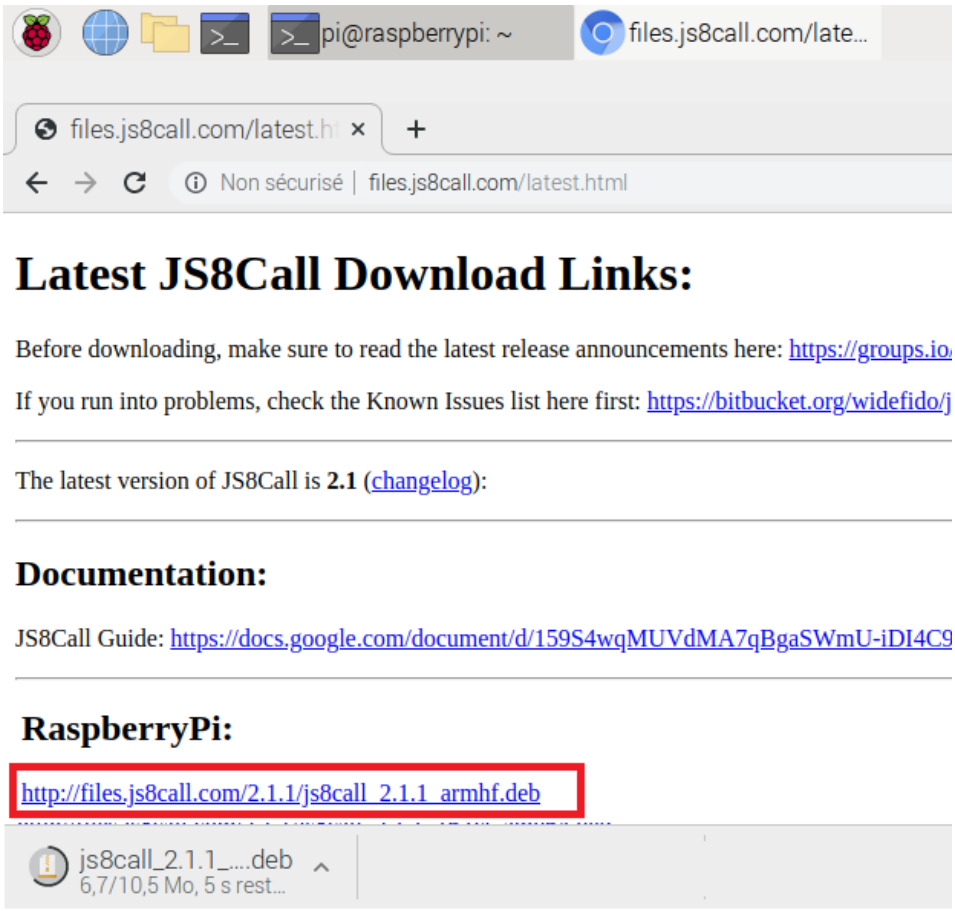
ls	`ls` for Lima Sierra letters	`pi@raspberrypi:~/Downloads $ ls`
		`jtdx-2.1.0-rc150_u_armhf.deb`
		`pi@raspberrypi:~/Downloads $`
		The file is present

sudo dpkg -i jtdx-2.1.0-rc150_u_armhf.deb		Jtdx installation
sudo apt-get install -f		Libraries are missing, this command allows you to search and install the missing files.
continue ? [O/n]		O

It is possible to run JTDX manually with LXTterminal and typing **jtdx**.	pi@raspberrypi: ~ Fichier Édition Onglets Aide `pi@raspberrypi:~ $ jtdx`

2.11 JS8CALL

With the Chromium browser, go to js8call's main page, http://files.js8call.com/latest.html, then download **js8call_2.1.1_armhf.deb**

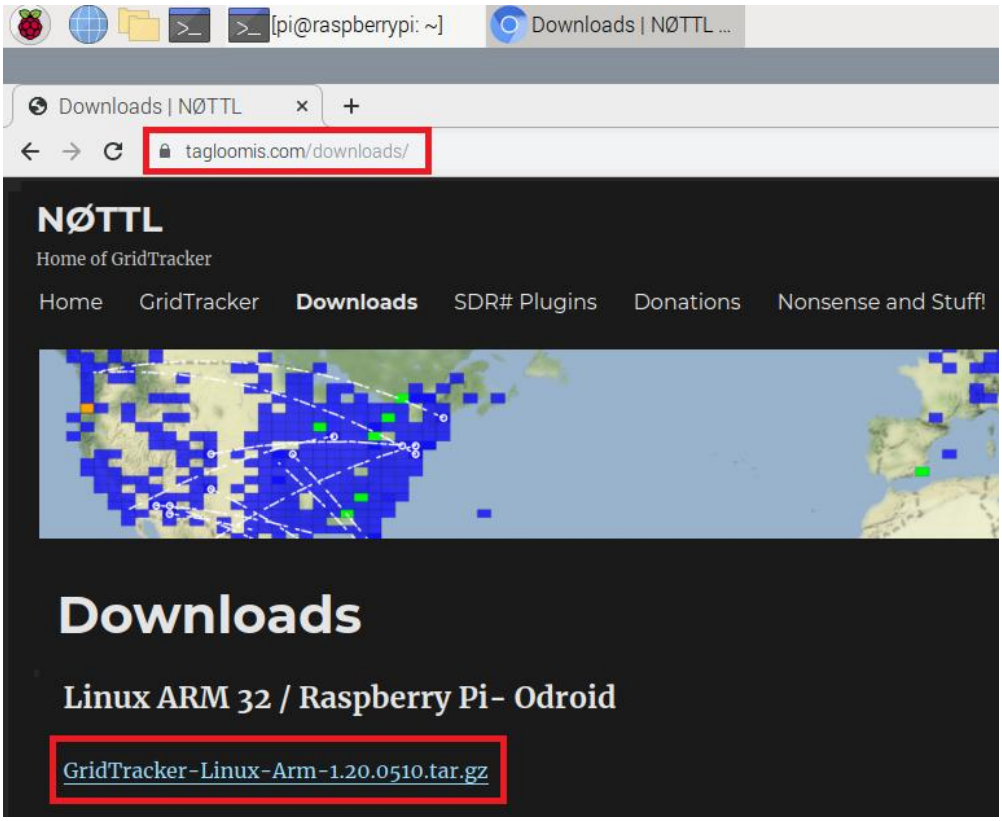


A l'aide de LXTerminal ou de putty, taper les commandes suivantes dans l'ordre

cd Downloads		pi@raspberrypi:~ $ cd Downloads/ pi@raspberrypi:~/Downloads $ Go to the Downloads directory
ls	`ls` Lima Sierra letters	pi@raspberrypi:~/Downloads $ ls js8call_2.1.1_armhf.deb jtdx-2.1 pi@raspberrypi:~/Downloads $ The file is present
sudo dpkg -i js8call_2.1.1_armhf.deb		Jtdx installation
sudo apt-get install -f		If libraries are missing, this command allows to search and install the missing files (If wsjt-x and jtdx were installed before js8call, this command is not required)
continue ? [O/n]		O

2.12 GridTracker (Real-time tracking of Locators during a QSO)

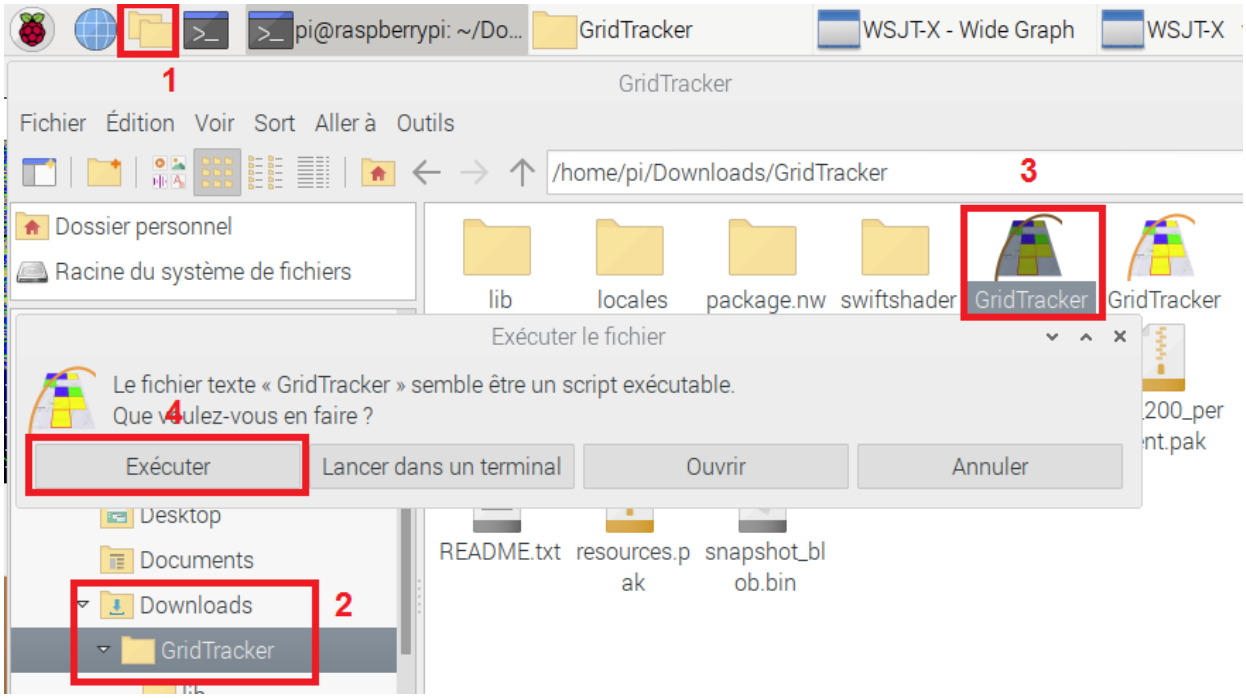
With the Chromium browser, go to the following page: https://tagloomis.com/downloads/, then download **GridTracker-Linux-Arm-1.20.0510.tar.gz**



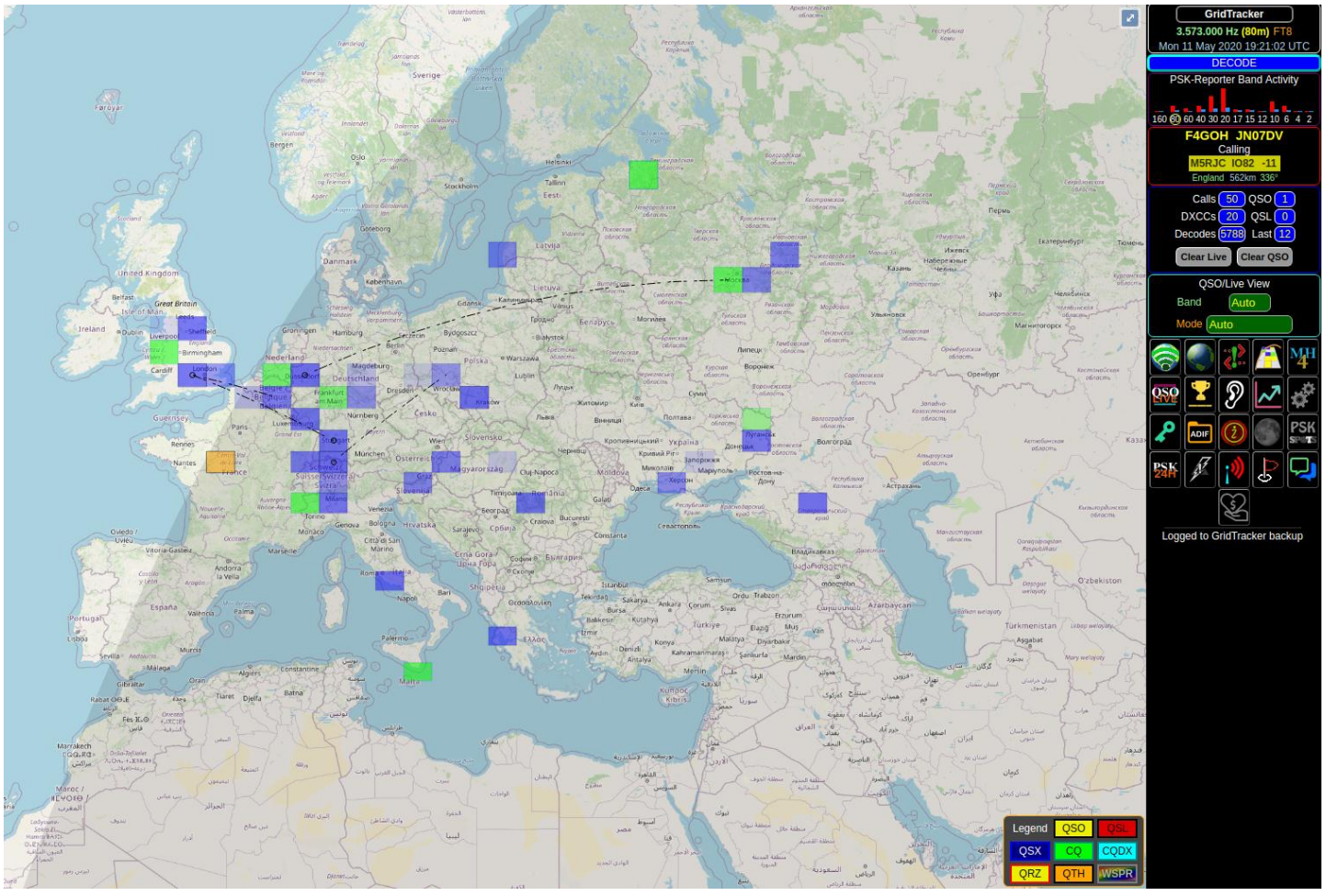
Using LXTerminal or putty, type the following commands in order

cd Downloads		pi@raspberrypi:~ $ cd Downloads/ pi@raspberrypi:~/Downloads $ Go to the Downloads directory
ls	`ls` for Lima Sierra letters	pi@raspberrypi:~/Downloads $ ls GridTracker-Linux-Arm-1.20.0510.tar.gz The file is present
`tar -zxvf GridTracker-Linux-Arm-1.20.0510.tar.gz`		Decompresses the archive (can be done graphically with the file manager)
sudo apt-get install libgconf-2-4		Adds a missing library
Souhaitez-vous continuer ? [O/n]		O
cd GridTracker		Go to the Gridtracker directory
./GridTracker		Executes the Gridtracker program.

For this program, there is no special installation process, you need to use the file manager (1), browse to download (2), then run Gridtracker(3 and 4).
I advise you to move the directory in order not to delete it accidentally. To create an icon on the desktop, follow the instructions in the README.txt file



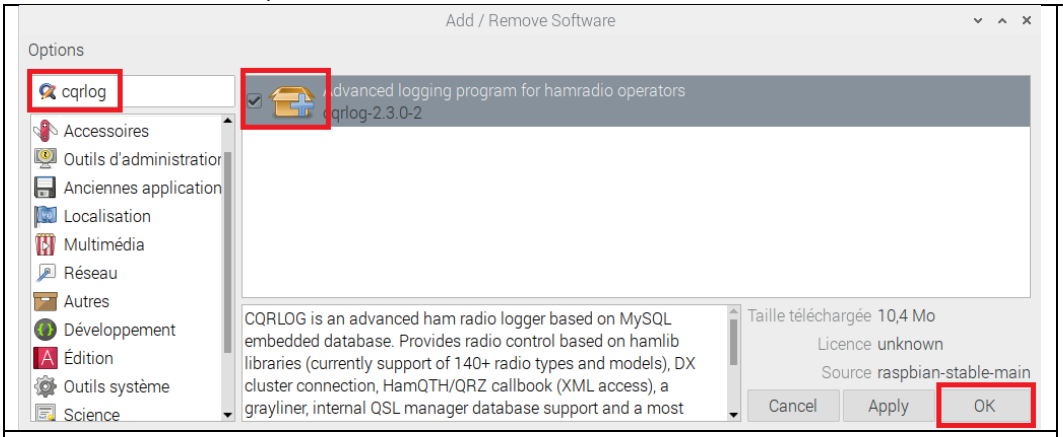
GridTracker works with WSJT-X

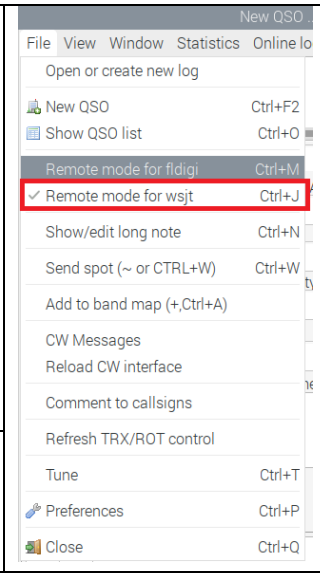


2.13 Cqrlog

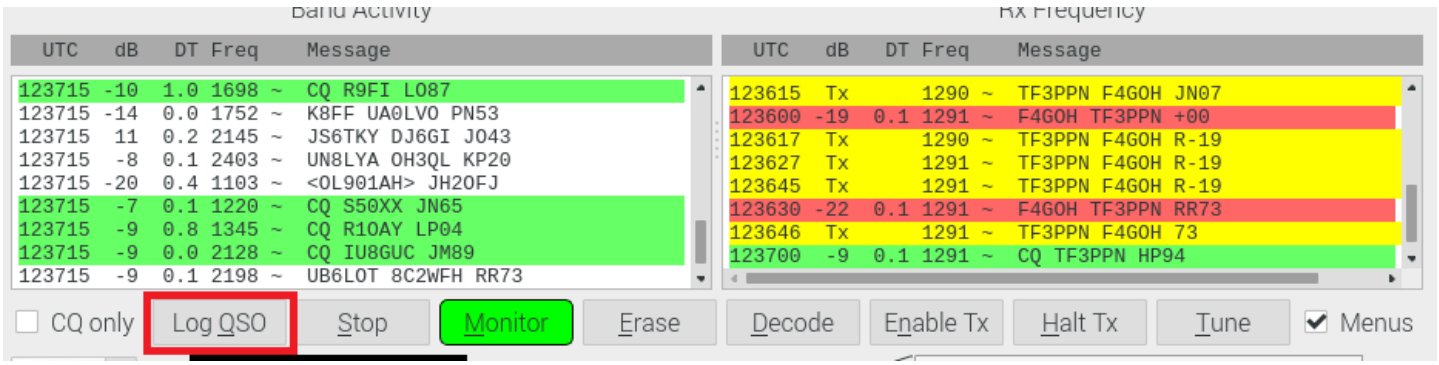
Manages the traffic log in connection with wsjt-x and automatically sends logs to EQSL, LOTW, HRDlog.net, etc., provided that the server in WSTJ-X is active (see page 15 Reporting menu)

Preferences menu, Add / Remove Software.

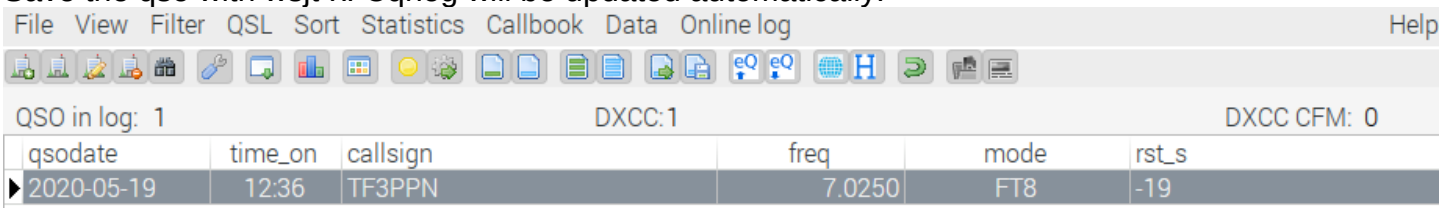




As soon as you launch cqrlog, don't forget to create the database. Watch the K0PIR cqrlog videos to activate the "remote" mode wsjt-x.

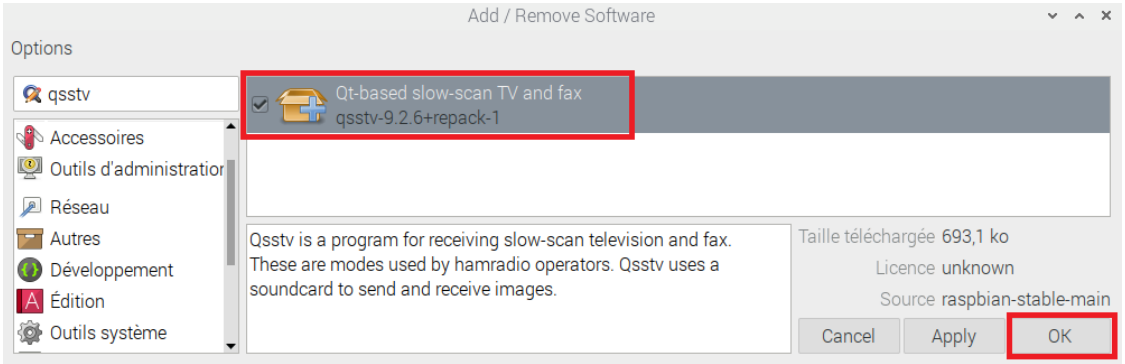


Save the qso with wsjt-x. Cqrlog will be updated automatically.

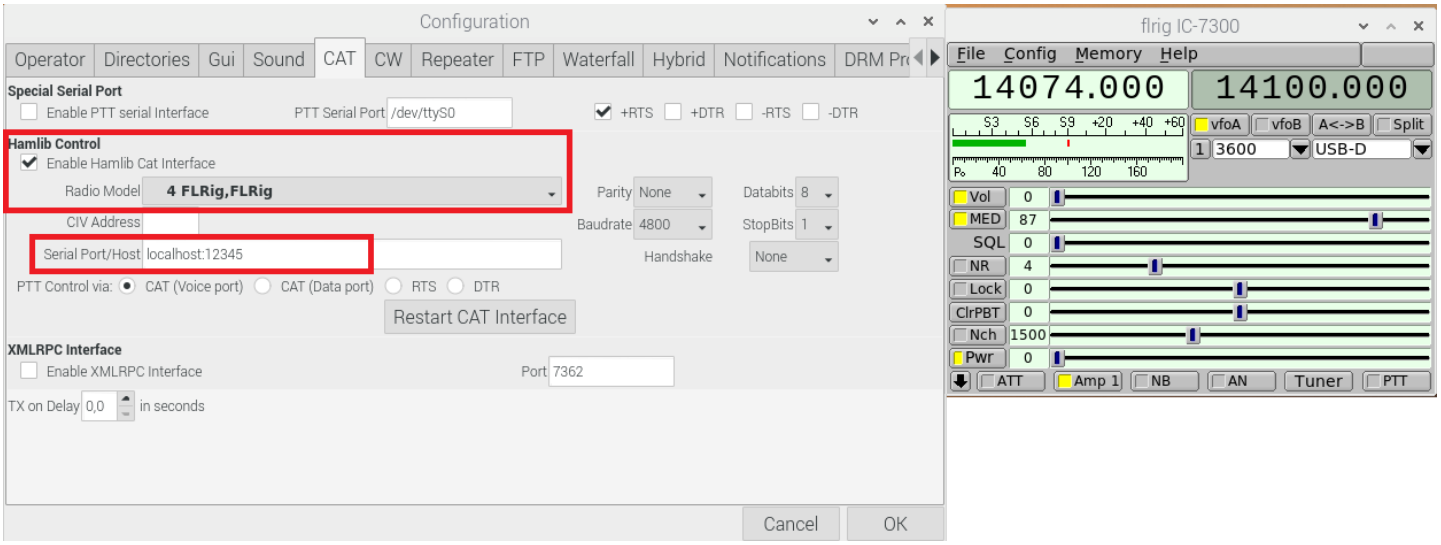


2.14 Installation de Qsstv

Preferences menu, Add / Remove Software



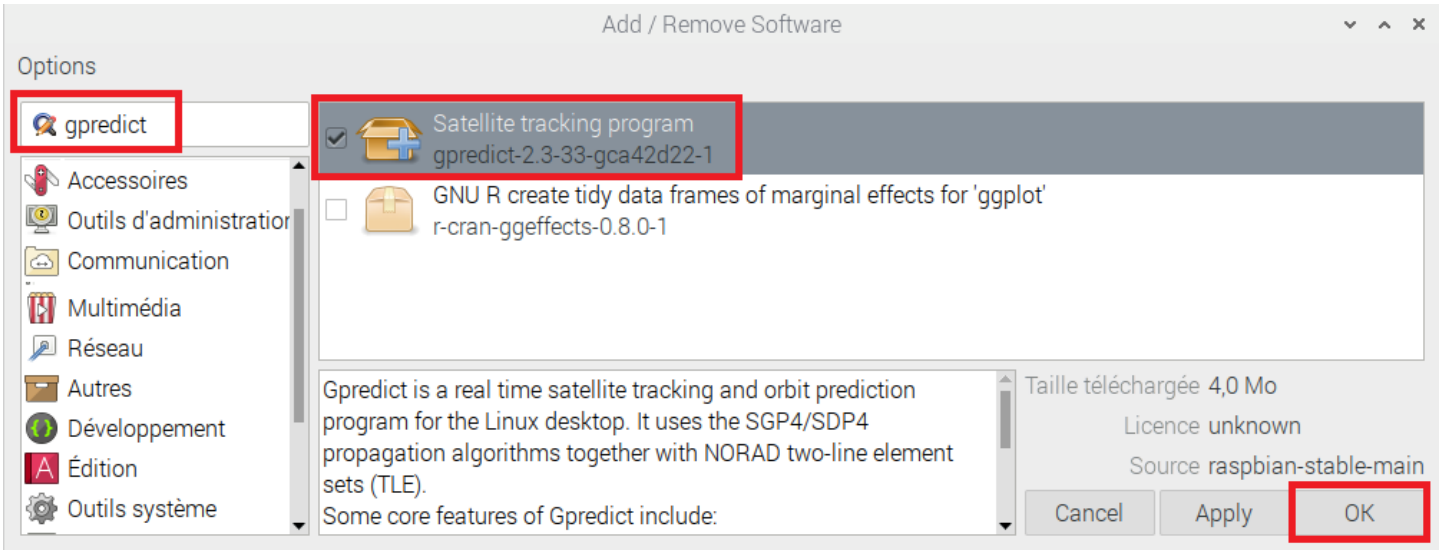
To control the transceiver with qsstv, use flig, as shown in the screenshot below



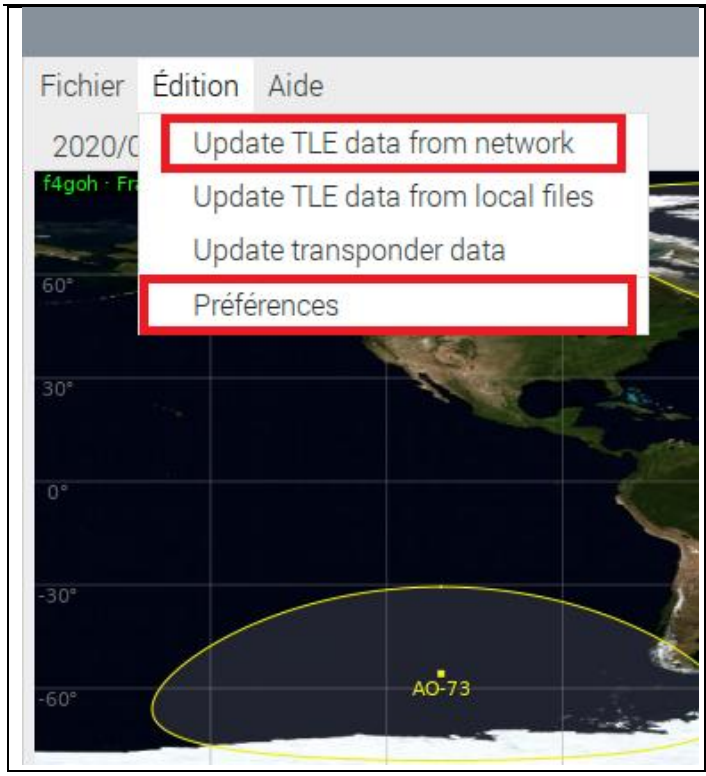
2.15 Gpredict

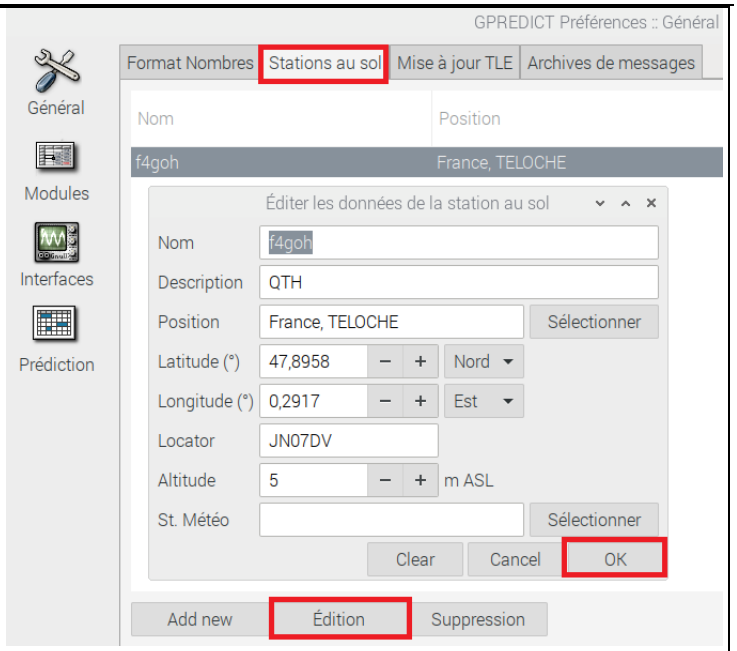
Satellite passage prediction software.

Preferences menu, Add / Remove Software



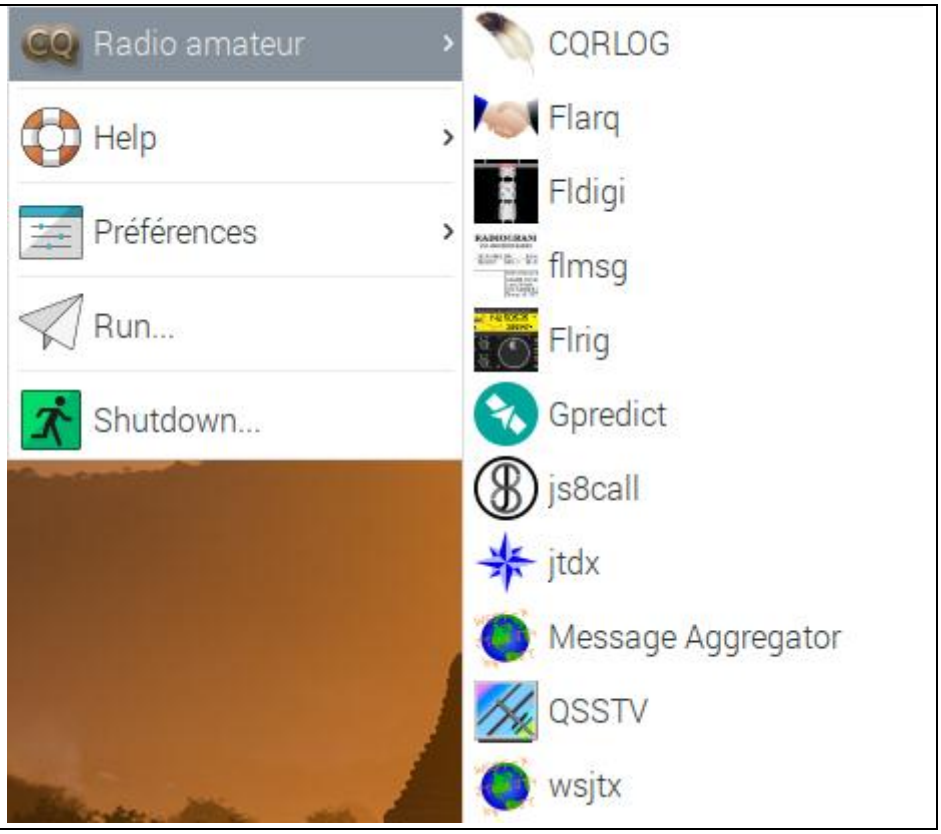
Start by updating the TLEs (Two-Line Elements: "Two-Line Orbital Parameters"), then change the position of the station on the GROUND (preferences menu).



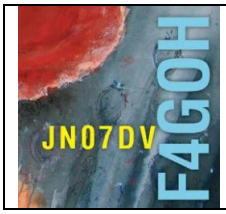


The installation result:

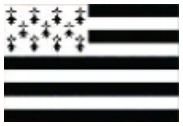
Don't forget to make a mSD card backup.
(See page 2)



Next part: Using an rtl sdr receiver key.



Raspberry Pi for HAM Radio
Part 3



Prerequisite: HAM Radio tutorial with the Raspberry Pi Part 1 and 2.

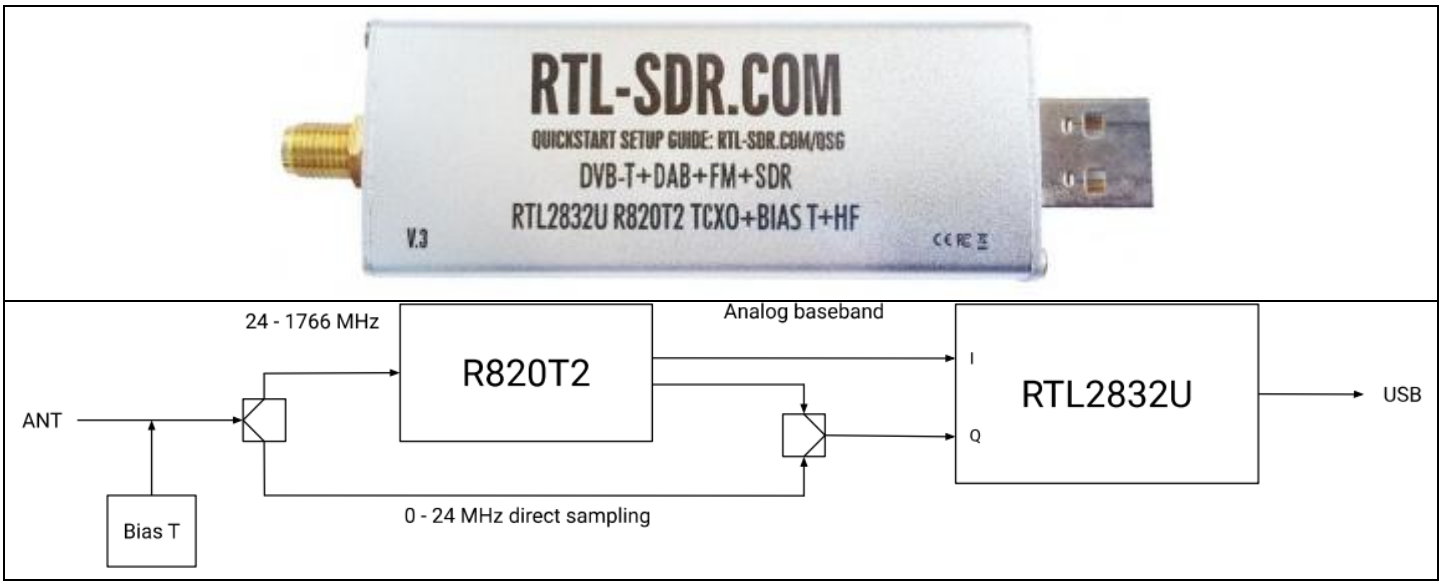
Summary:

Part 3: Using RTL-SDR Key

Version du 16/02/2021 V1.3

3.1 Introduction.

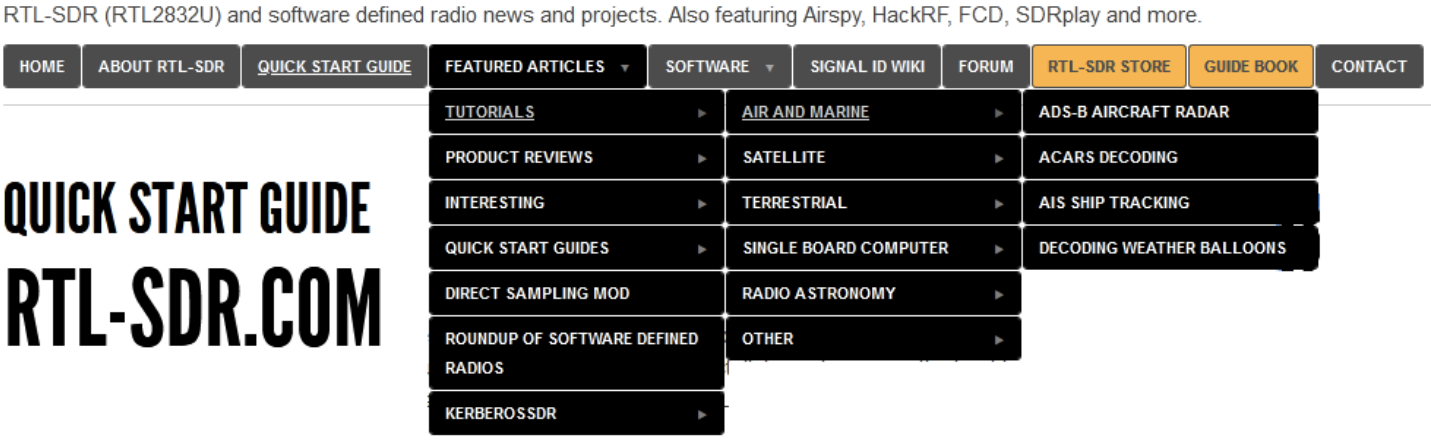
In this last tutorial about Raspbian in graphical mode, I explain the installation procedure of common software using the RTL-SDR receiver key. For several years, this famous key has not stopped evolving. The latest version is the [RTL-SDR V3](RTL-SDR V3).



https://gitlab.com/librespacefoundation/sdrmakerspace/sdreval/-/raw/master/Report/pdf/Evaluation_of_SDR_Boards-1.0.pdf

The internal structure shows the 2 main integrated circuits as well as two "switches" to switch between HF and VHF/UHF band. In addition, the key now has a Bias T avoiding the need to use an external power supply for a preamplifier.

Do not hesitate to consult the official website of the key and browse the different menus.
https://www.rtl-sdr.com/rtl-sdr-blog-v-3-dongles-user-guide/



Despite this, I never found this site very clear. Some explanations are well done and others are really sloppy. There are a lot of videos with demonstrations of operation, showing the decoding of this or that signal. At that time, we ask ourselves the following question:
"How do you install these programs in the computer or the Raspberry Pi? Then starts a real police investigation in order to cross various sources of information to succeed in installing the programs.

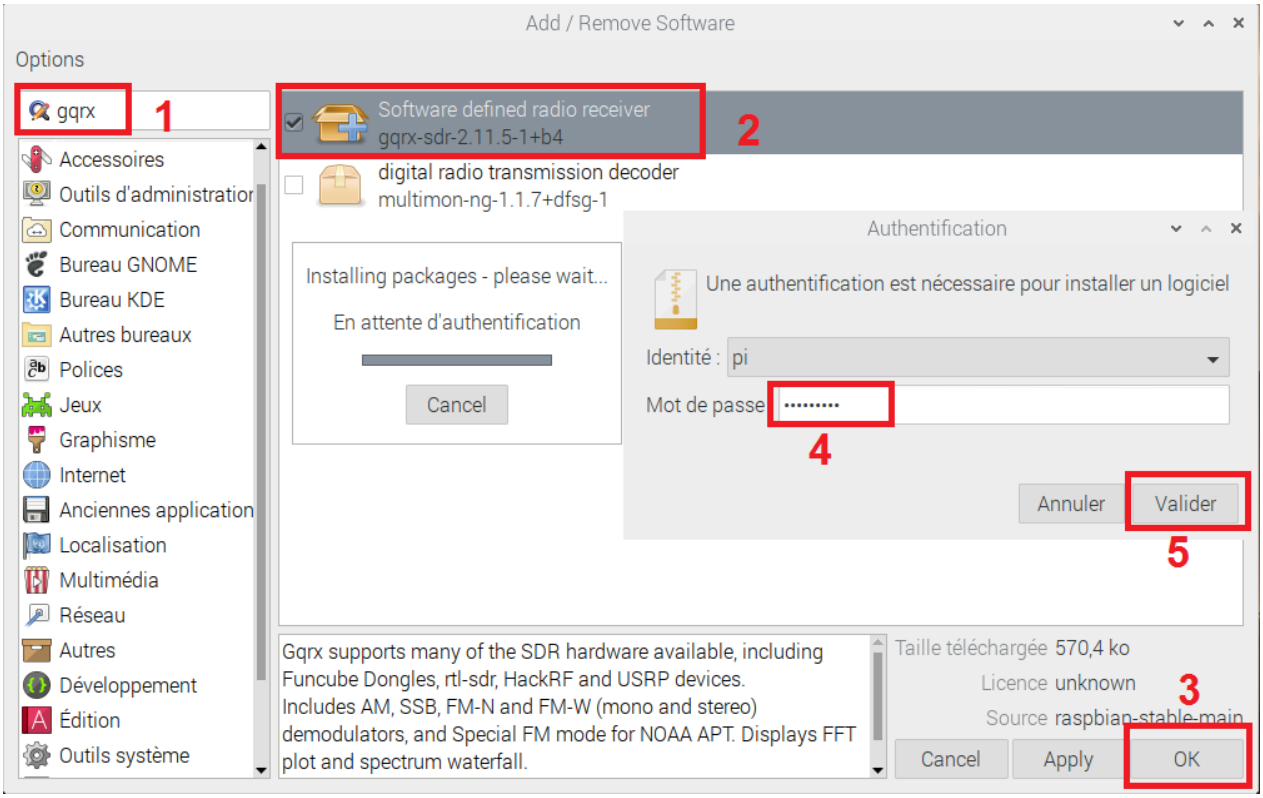
It's a real shame that this environment is exclusively the business of IT specialists, because it deserves to be more democratized. I have often tested installation procedures from websites or even very recent books with often very mixed success. I imagine that an OM alone in its shack may have difficulty to succeed. Hence the interest of radio clubs where someone who has already done the installation can provide help. Without the information remaining "word of mouth", consider making a complete PDF installation sheet by publishing it on the site of the club concerned.

3.2 GQRX

Gqrx is an open source SDR receiver software using GNU radio and the Qt graphical toolkit. There are two possible command line installations with either LXTerminal or Putty:

```
sudo apt update
sudo apt upgrade
sudo apt install gqrx-sdr
```

Or in the Preferences menu, Add / Remove Software.
Search on the keyword "gqrx" (1), select the package to be installed (2), then ok (3), enter again the password created during installation (4 and 5) (See part 1).



In both cases, you will have to wait at least 20 minutes, because there are more than 500MB of files to download and install.

3.3 RTL-SDR Drivers

Remove the rtl-sdr drivers that were set by default with gqrx-sdr and install the latest version of the software manually from the command line with LXTerminal or Putty.

```
sudo apt-get remove rtl-sdr
```

```
pi@raspberrypi:~ $ sudo apt-get remove rtl-sdr librtlsdr0 librtlsdr-dev
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Le paquet « librtlsdr-dev » n'est pas installé, et ne peut donc être supprimé
Le paquet « librtlsdr0 » n'est pas installé, et ne peut donc être supprimé
Le paquet « rtl-sdr » n'est pas installé, et ne peut donc être supprimé
Les paquets suivants ont été installés automatiquement et ne sont plus nécessair
es :
  libmicrodns0 libqt5charts5 xlog-data
Veuillez utiliser « sudo apt autoremove » pour les supprimer.
0 mis à jour, 0 nouvellement installés, 0 à enlever et 0 non mis à jour.
pi@raspberrypi:~ $
```

Retrieve the official files.

```
git clone https://github.com/osmocom/rtl-sdr.git
```

```
pi@raspberrypi:~ $ git clone https://github.com/osmocom/rtl-sdr.git
Clonage dans 'rtl-sdr'...
remote: Enumerating objects: 65, done.
remote: Counting objects: 100% (65/65), done.
remote: Compressing objects: 100% (38/38), done.
remote: Total 1863 (delta 25), reused 55 (delta 23), pack-reused 1798
Réception d'objets: 100% (1863/1863), 448.80 KiB | 1.11 MiB/s, fait.
Résolution des deltas: 100% (1317/1317), fait.
pi@raspberrypi:~ $ ls
audio      Downloads    Music     qsstv      Templates  Videos
Desktop    GridTracker  Pictures  rtl-sdr    tx_drm
Documents  MagPi        Public    templates  tx_stock
pi@raspberrypi:~ $
```

Install the libraries.

```
sudo apt install build-essential cmake usbutils libusb-1.0-0-dev
```

```
pi@raspberrypi:~/rtl-sdr/build $ sudo apt install build-essential cmake usbutils
 libusb-1.0-0-dev
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
build-essential est déjà la version la plus récente (12.6).
usbutils est déjà la version la plus récente (1:010-3).
Les paquets suivants ont été installés automatiquement et ne sont plus nécessair
es :
  libmicrodns0 libqt5charts5 xlog-data
Veuillez utiliser « sudo apt autoremove » pour les supprimer.
Les paquets supplémentaires suivants seront installés :
  cmake-data libjsoncpp1 librhash0 libusb-1.0-doc libuv1
Paquets suggérés :
  cmake-doc ninja-build
Les NOUVEAUX paquets suivants seront installés :
  cmake cmake-data libjsoncpp1 librhash0 libusb-1.0-0-dev libusb-1.0-doc
  libuv1
0 mis à jour, 7 nouvellement installés, 0 à enlever et 0 non mis à jour.
Il est nécessaire de prendre 4□578 ko dans les archives.
Après cette opération, 23,6 Mo d'espace disque supplémentaires seront utilisés.
Souhaitez-vous continuer ? [O/n] O
```

Type these commands in the order :

```
cd rtl-sdr

mkdir build

cd build

cmake -DINSTALL_UDEV_RULES=ON -DDETACH_KERNEL_DRIVER=ON ../
```

```
pi@raspberrypi:~ $ cd rtl-sdr/
pi@raspberrypi:~/rtl-sdr $ mkdir build
pi@raspberrypi:~/rtl-sdr $ cd build/
pi@raspberrypi:~/rtl-sdr/build $ cmake -DINSTALL_UDEV_RULES=ON -DDETACH_KERNEL_D
RIVER=ON ../
-- The C compiler identification is GNU 8.3.0
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compiler: /usr/bin/cc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Build type not specified: defaulting to release.
-- Extracting version information from git describe...
-- Looking for pthread.h
-- Looking for pthread.h - found
```

sudo make install

```
pi@raspberrypi:~/rtl-sdr/build $ sudo make install
Scanning dependencies of target convenience_static
[  3%] Building C object src/CMakeFiles/convenience_static.dir/convenience/conve
nience.c.o
[  6%] Linking C static library libconvenience_static.a
[  6%] Built target convenience_static
Scanning dependencies of target rtlsdr
[  9%] Building C object src/CMakeFiles/rtlsdr.dir/librtlsdr.c.o
[ 12%] Building C object src/CMakeFiles/rtlsdr.dir/tuner_e4k.c.o
[ 15%] Building C object src/CMakeFiles/rtlsdr.dir/tuner_fc0012.c.o
[ 18%] Building C object src/CMakeFiles/rtlsdr.dir/tuner_fc0013.c.o
[ 21%] Building C object src/CMakeFiles/rtlsdr.dir/tuner_fc2580.c.o
[ 25%] Building C object src/CMakeFiles/rtlsdr.dir/tuner_r82xx.c.o
[ 28%] Linking C shared library librtlsdr.so
```

sudo ldconfig

```
-- Set runtime path of "/usr/local/bin/rtl_fm" to ""
-- Installing: /usr/local/bin/rtl_eeprom
-- Set runtime path of "/usr/local/bin/rtl_eeprom" to ""
-- Installing: /usr/local/bin/rtl_adsb
-- Set runtime path of "/usr/local/bin/rtl_adsb" to ""
-- Installing: /usr/local/bin/rtl_power
-- Set runtime path of "/usr/local/bin/rtl_power" to ""
pi@raspberrypi:~/rtl-sdr/build $ sudo ldconfig
pi@raspberrypi:~/rtl-sdr/build $
```

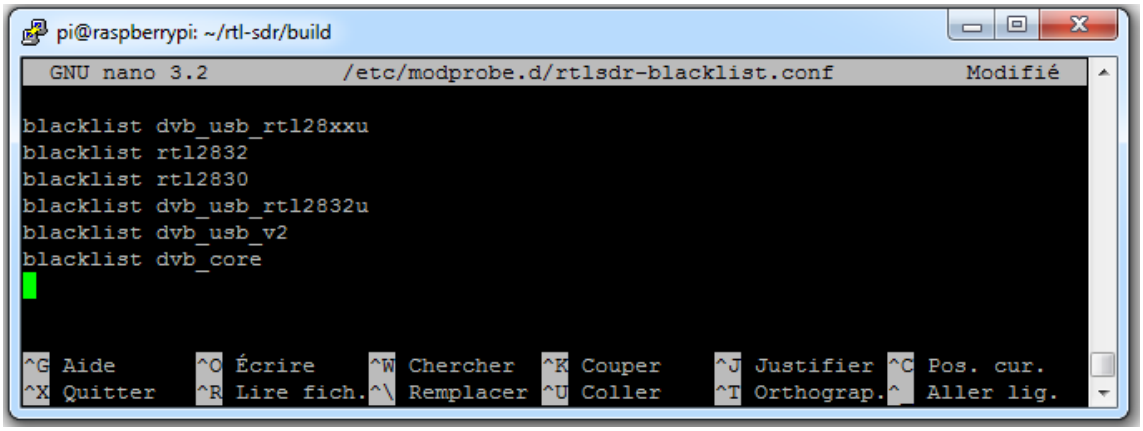
sudo nano /etc/modprobe.d/rtlsdr-blacklist.conf

```
pi@raspberrypi:~/rtl-sdr/build $ sudo nano /etc/modprobe.d/rtlsdr-blacklist.conf
```

```
blacklist dvb_usb_rtl28xxu
blacklist rtl2832
blacklist rtl2830
blacklist dvb_usb_rtl2832u
blacklist dvb_usb_v2
blacklist dvb_core
```

Copy and paste the above list into the nano editor

Reminder right click to paste the text in the nano editor



Save the file using the Ctrl + o keys, then the enter key to validate the recording.	**Exit the editor using Ctrl + x.**
	

Restart the Raspberry PI

```
pi@raspberrypi:~/rtl-sdr/build $ sudo reboot
```

```
sudo reboot
```

Verification: Plug the rtl-sdr key into a USB port on the Raspberry Pi.
With LXTerminal or Putty, type the following command line:

```
lsusb
```

Locate the rtl-sdr key: **RTL238 DVB-T**

Case of the Raspberry Pi 3 :

```
pi@raspberrypi:~ $ lsusb
Bus 001 Device 004: ID 0bda:2838 Realtek Semiconductor Corp. RTL2838 DVB-T
Bus 001 Device 003: ID 0424:ec00 Standard Microsystems Corp. SMSC9512/9514 Fast
Ethernet Adapter
Bus 001 Device 002: ID 0424:9514 Standard Microsystems Corp. SMC9514 Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

Case of the Raspberry Pi 4 :

```
pi@raspberrypi:~ $ lsusb
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 003: ID 0bda:2838 Realtek Semiconductor Corp. RTL2838 DVB-T
Bus 001 Device 002: ID 2109:3431 VIA Labs, Inc. Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

With LXTerminal or Putty, type the following command line : (be careful, underscore key)

```
rtl_test
```

The rtl-sdr key should be recognized, if not, disconnect and reconnect the key and restart the test.

```
pi@raspberrypi:~ $ rtl_test
Found 1 device(s):
  0:  Realtek, RTL2838UHIDIR, SN: 00000001

Using device 0: Generic RTL2832U OEM
Found Rafael Micro R820T tuner
Supported gain values (29): 0.0 0.9 1.4 2.7 3.7 7.7 8.7 12.5 14.4 15.7 16.6 19.7
 20.7 22.9 25.4 28.0 29.7 32.8 33.8 36.4 37.2 38.6 40.2 42.1 43.4 43.9 44.5 48.0
 49.6
[R82XX] PLL not locked!
Sampling at 2048000 S/s.

Info: This tool will continuously read from the device, and report if
samples get lost. If you observe no further output, everything is fine.

Reading samples in async mode...
^CSignal caught, exiting!
ctrl+c

User cancel, exiting...
Samples per million lost (minimum): 0
pi@raspberrypi:~ $ ^C
```

Exit the program by pressing the Ctrl key and the c key (**Ctrl+c**) at the same time.

If **`rtl_test`** displays this message over and over again `lost at least 112 bytes`, reinstall the program using the manual method.

Calculation optimization:

The program **`volk_profile`** creates a profile to calculate the FFT (*) optimized for the system on which it runs. This will take some time (20 minutes), but will improve graphics performance. With LXTerminal or Putty, type the following command line: (beware underscore key)

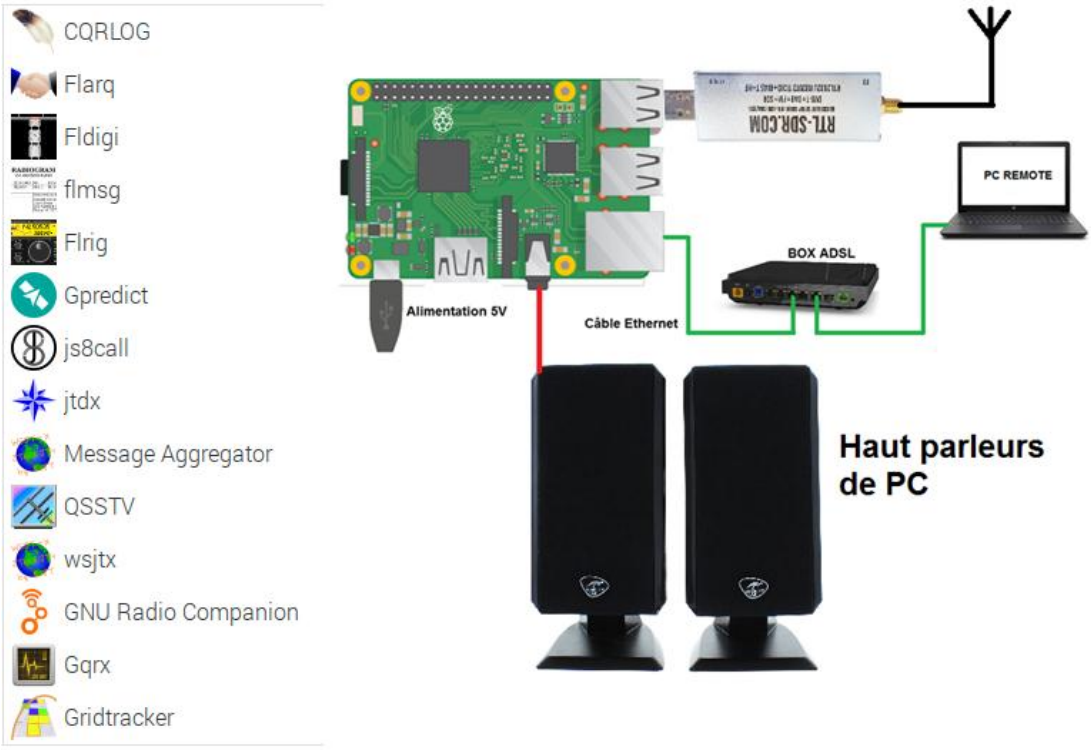
```
sudo volk_profile
```

```
pi@raspberrypi:~ $ sudo volk_profile
RUN_VOLK_TESTS: volk_64u_popcntpuppet_64u(131071,1987)
no architectures to test
RUN_VOLK_TESTS: volk_64u_popcntpuppet_64u(131071,1987)
no architectures to test
```

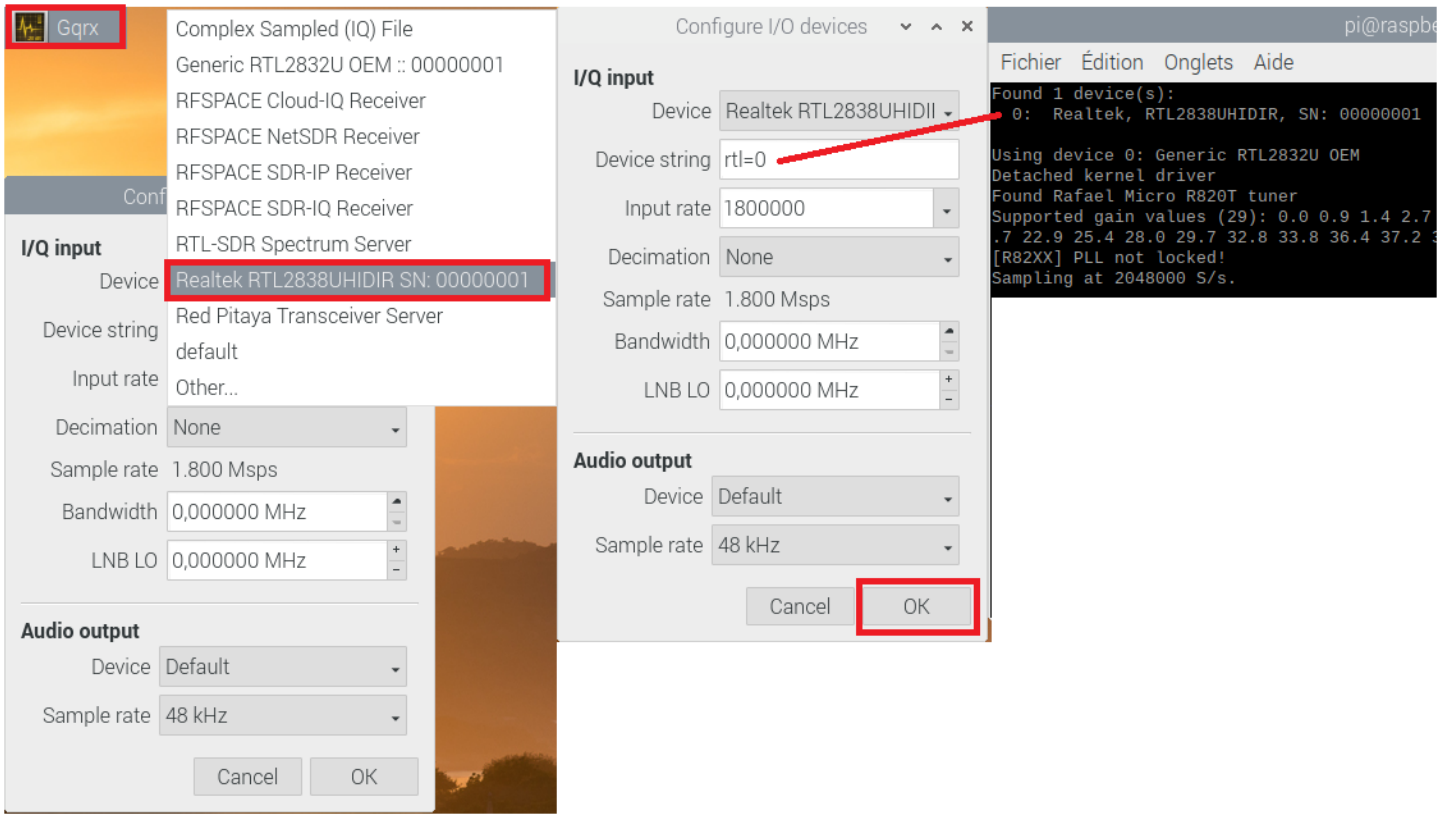
(*) FFT : https://en.wikipedia.org/wiki/Fast_Fourier_transform

3.4 Listen VHF with GQRX

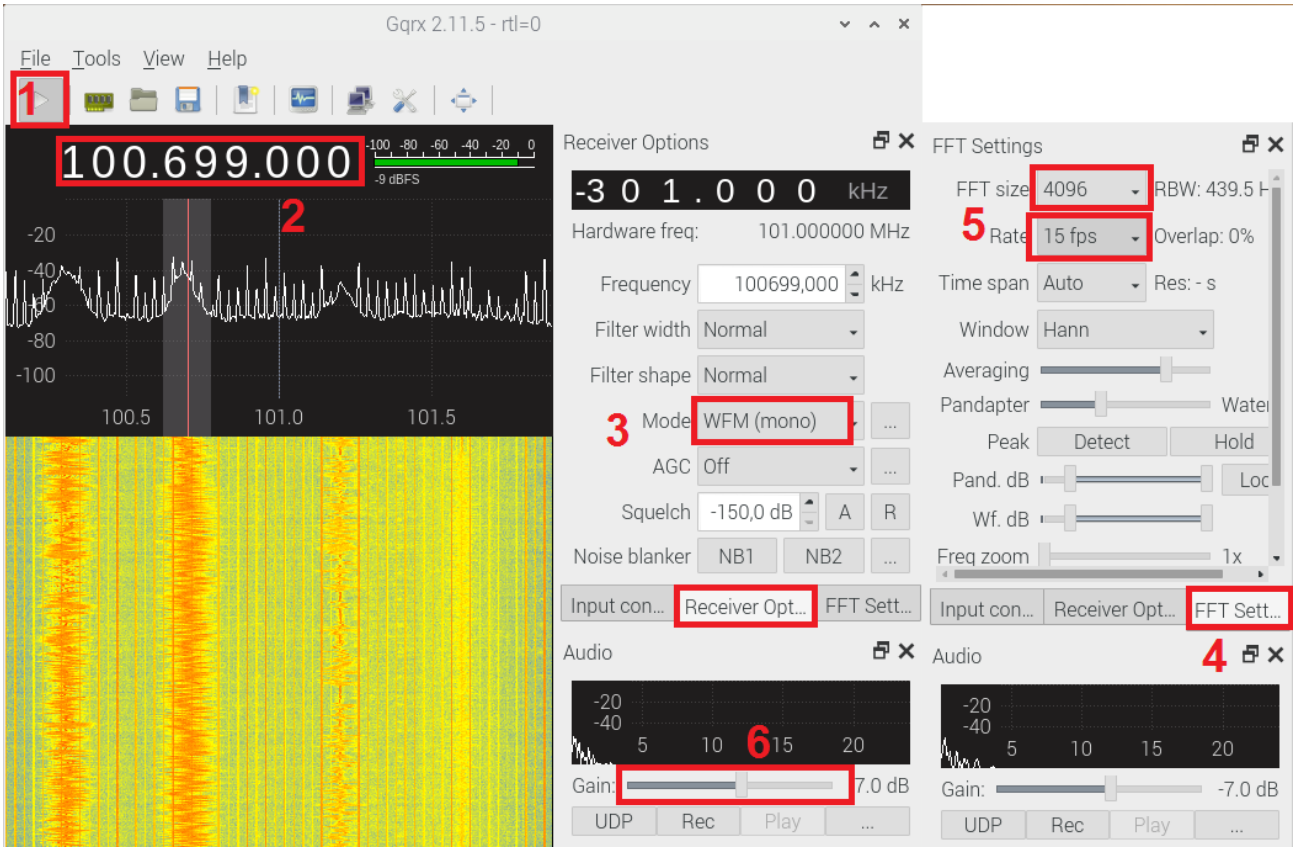
The amateur radio menu is starting to look good



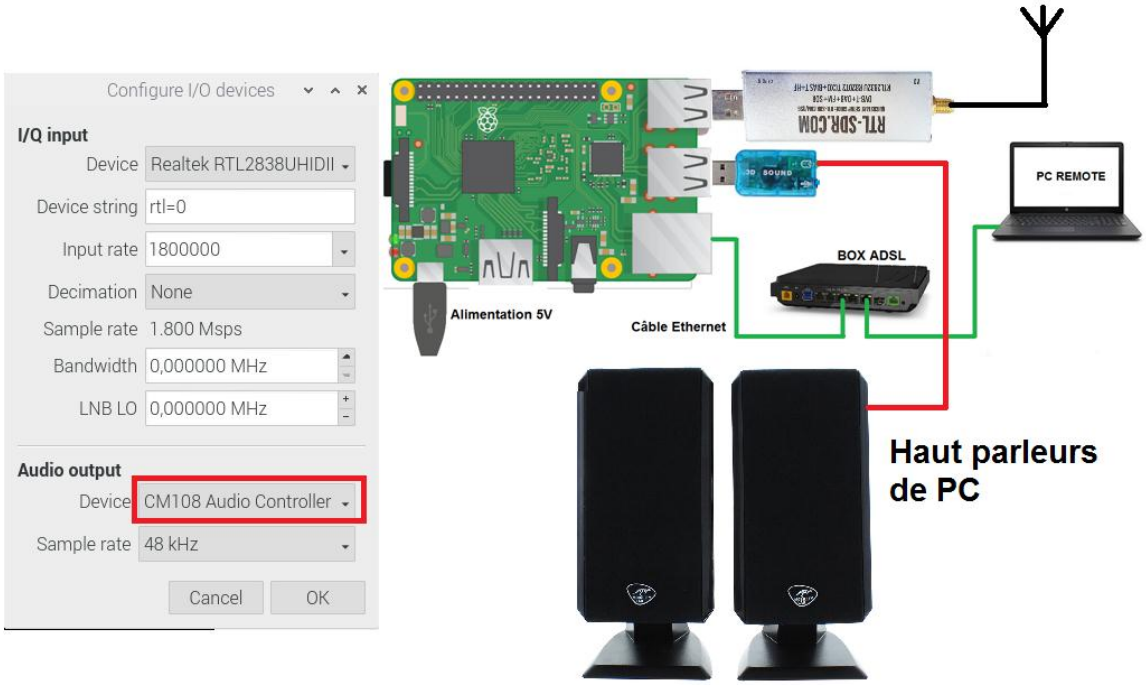
After connecting PC speakers to the 3.5 jack on the Raspberry PI, run the GQRX program. Select the rtl-sdr key. Note the link between rtl=0 and the test performed with `rtl_test`.



Before any specific decoding, it is vital to check the correct operation of the unit.
Start the software (1), adjust the frequency on the FM broadcast band (2), select the WFM mode (mono). If the sound is jerky, adjust the FFT size and Rate (5), finally adjust the sound output level (6).



In case of use with an external sound card, select Audio Output from the DEVICE drop-down menu.



If however the sound is very jerky, lower the screen resolution and avoid using VNC Viewer which takes a lot of processor resources.

➔ Menu Preferences, RPI configuration, Display, set resolution.

3.5 Listen HF with GQRX

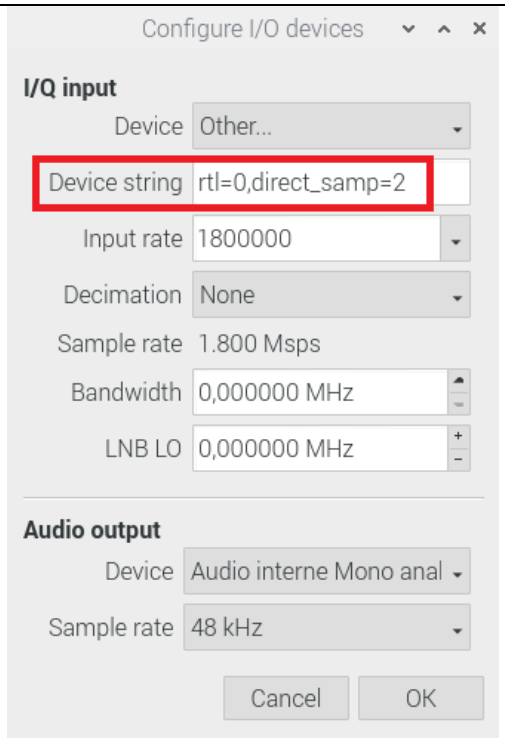
To switch from the VHF to HF band, it is best to exit QQRX and restart the program.

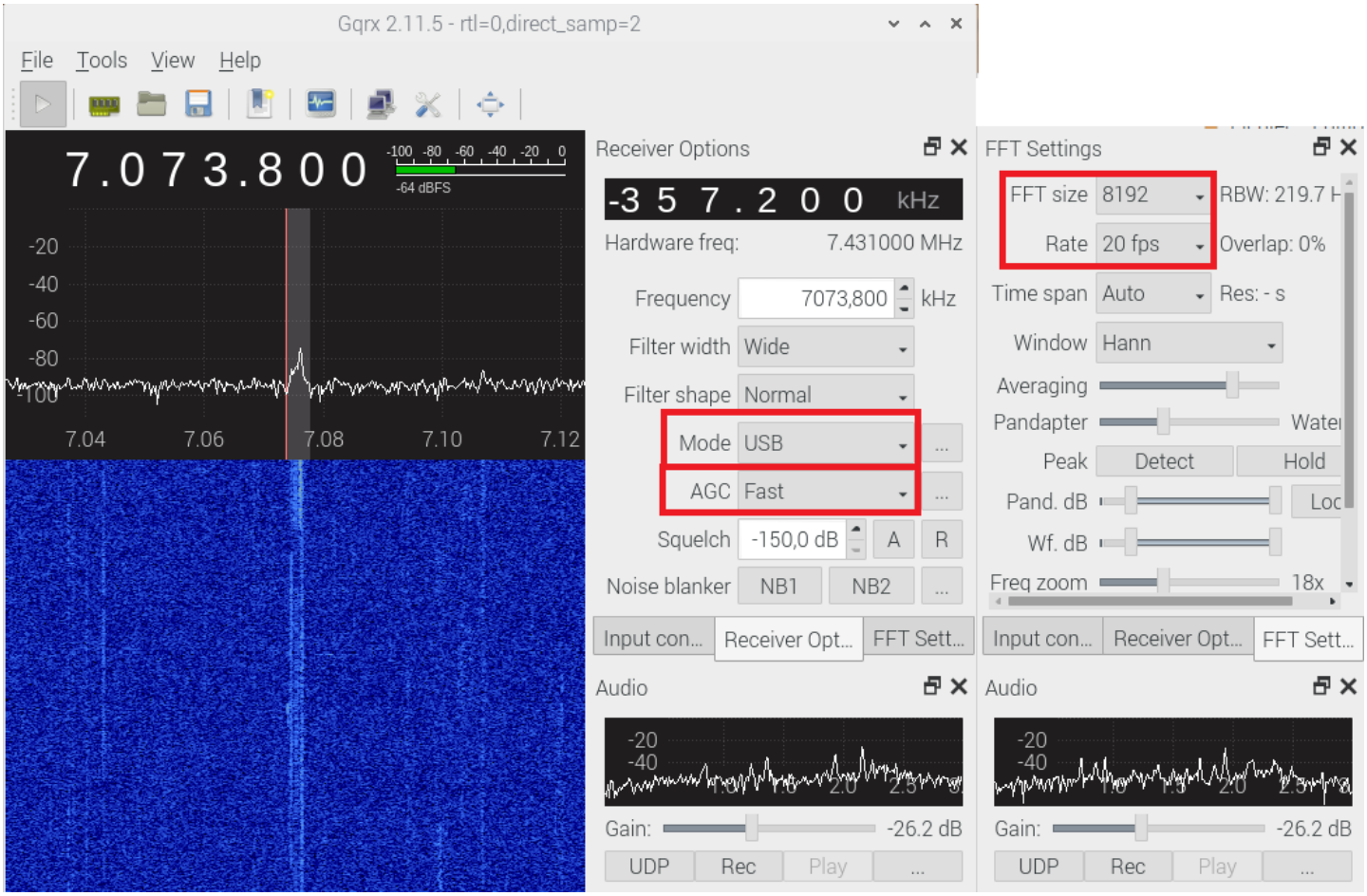
Add the parameter direct_samp=2 (beware underscore key)

`rtl=0,direct_samp=2`

After use, the Device drop-down menu automatically switches to Other.

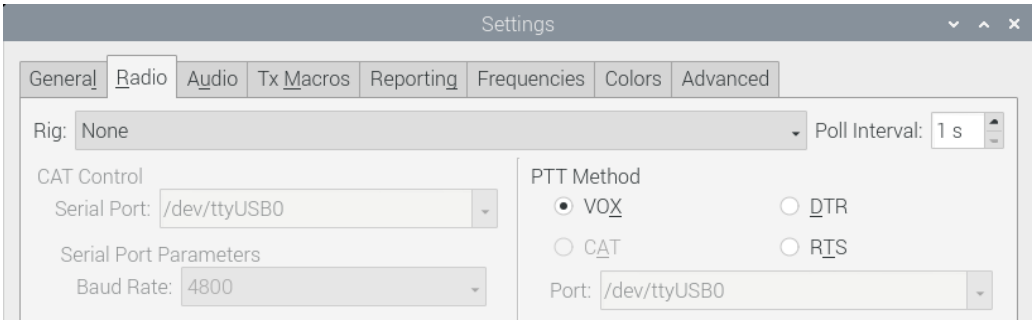
Listen to the band 40 meters on the frequency range of the FT8 digital mode, because there are now always people.





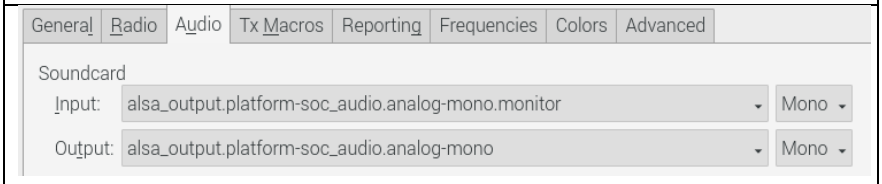
3.6 FT8 decoding with GQRX

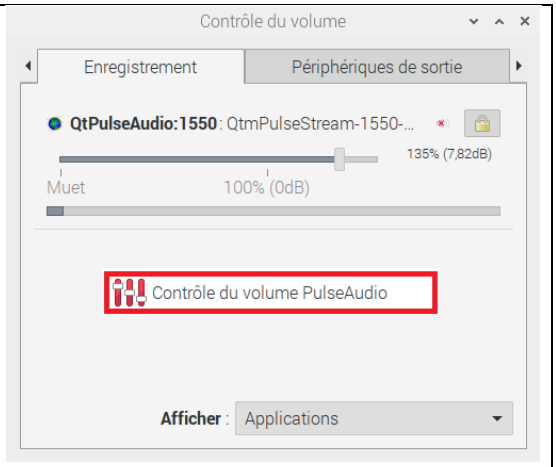
Remember to disable the CAT system if you haven't already done so:



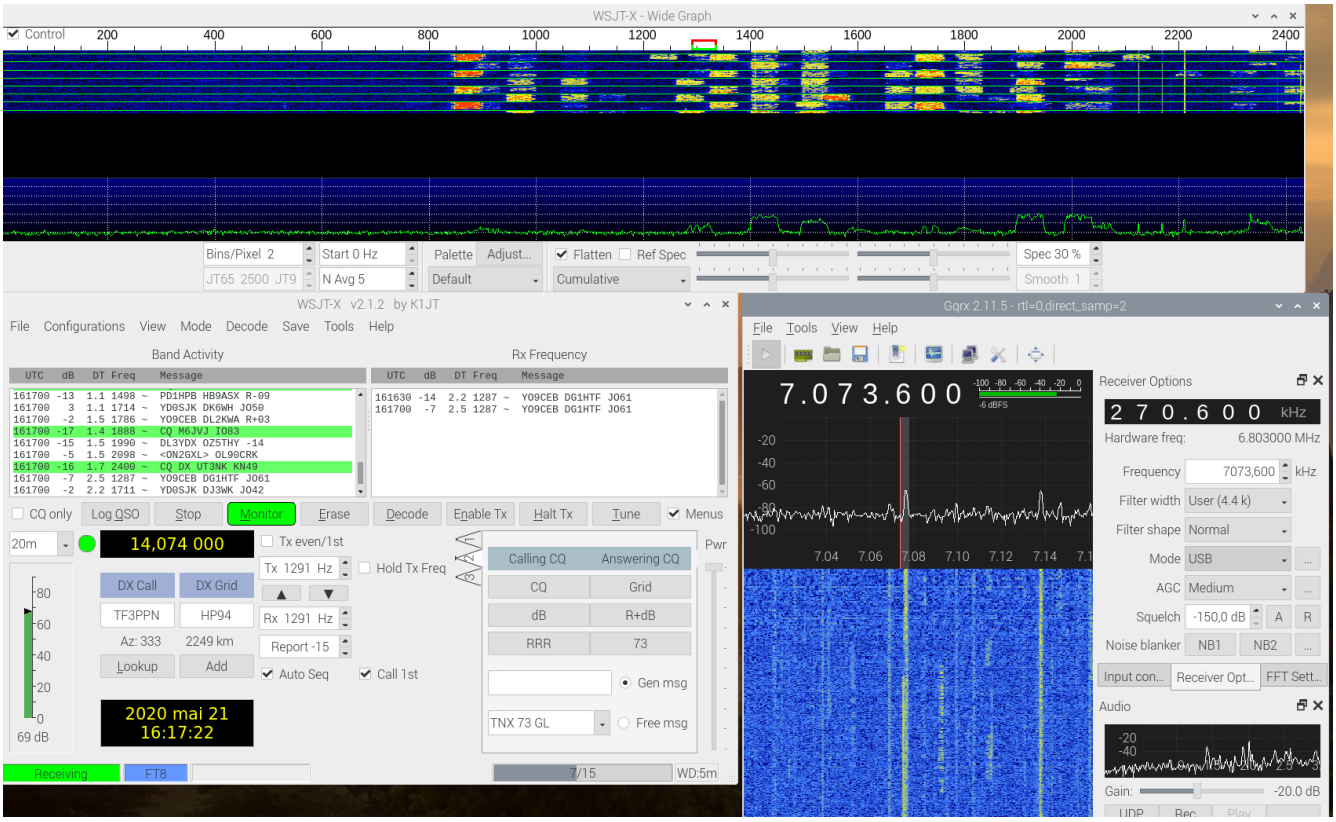
Run the WSJT-X program at the same time as GQRX, then check the audio configuration.

You can also see the activation of recording (monitoring) by the WSJT-X software in the PulseAudio volume control.





Decoding is immediate. A good way to use the Raspberry PI for listening without hogging a PC.

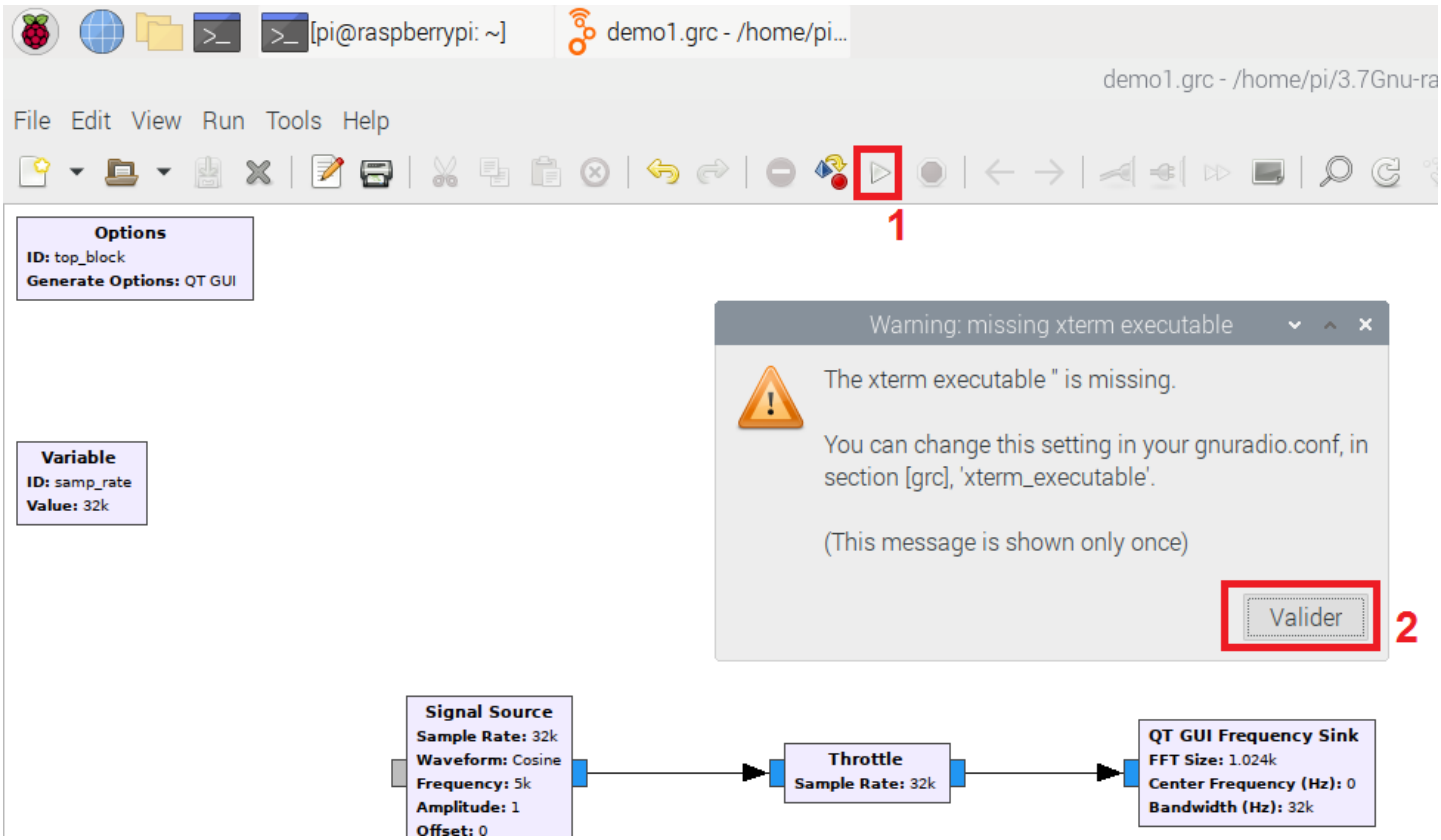


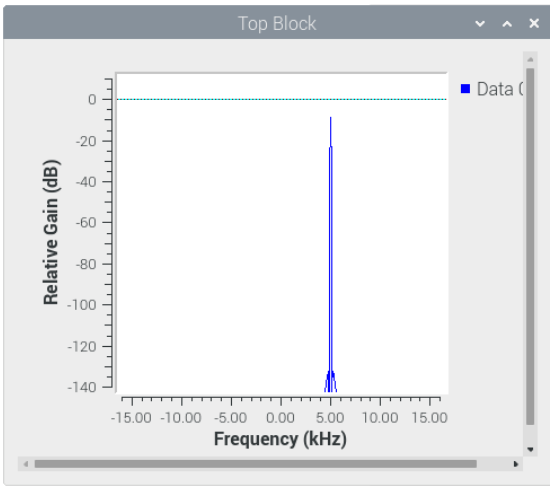
3.7 First step with GNU Radio

GNU Radio is a software suite dedicated to the implementation of sdr radios. The signal processing is done with GNU companion using blocks in a graphical way.

Execute the GNU companion program located in the Radio amateur menu. There is a lot of video on Youtube

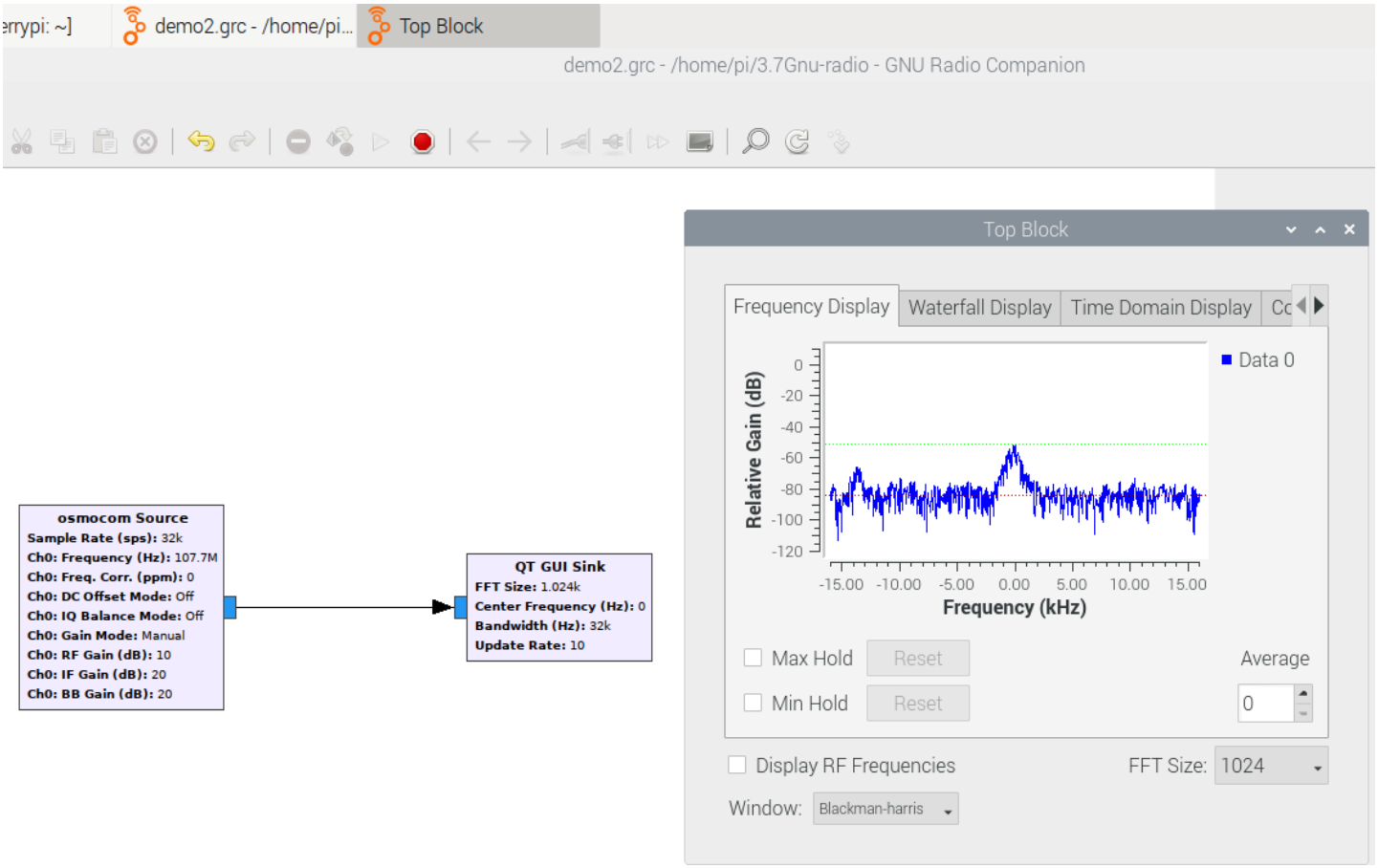
After saving the file, run the script by clicking on the arrow (1). An error message is displayed (2). Ignore it





Watch also the videos of David Haworth WA9ONY and especially the introduction of the rtl-sdr key with GNU Radio : (RTL SDR Raspberry Pi 4 GNU Radio Tutorial 1)

I set the frequency to 107.7Mhz in osmocom Source



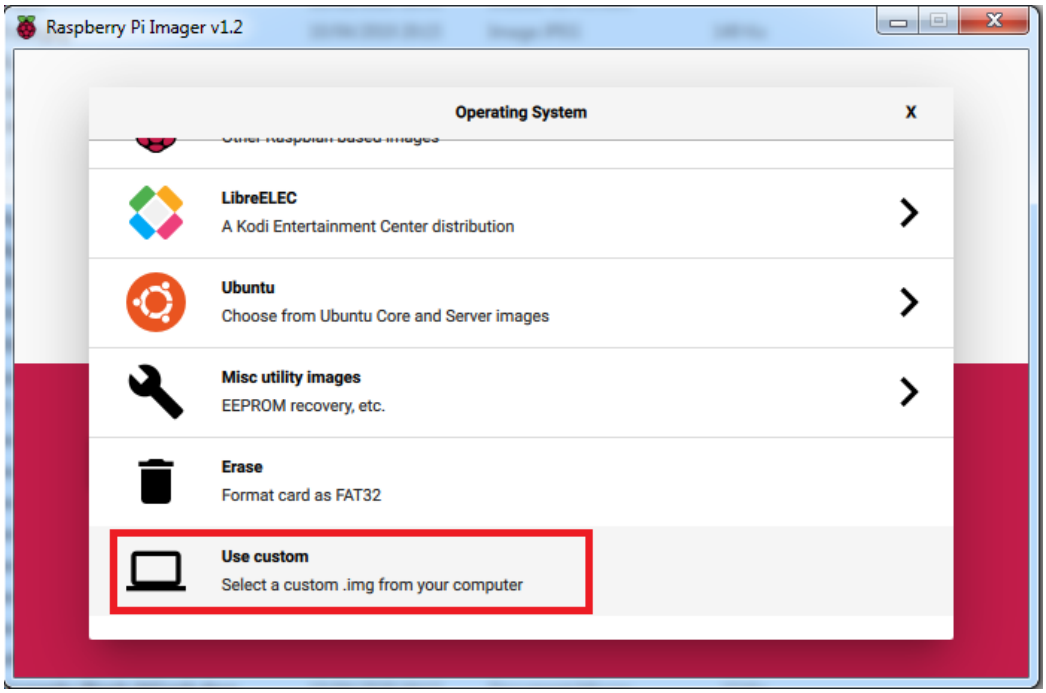
3.8 Using ready pisdr iso image

There is a distribution of Raspbian called **pisdr** with many software already preinstalled.

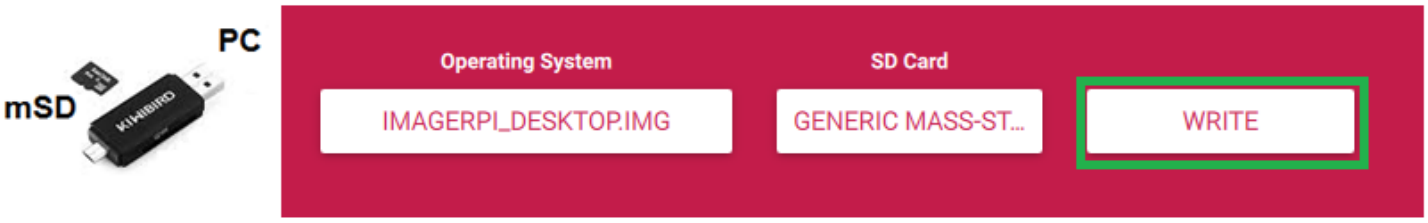
https://github.com/luigifcruz/pisdr-image

Download the file **2020-03-08-PiSDR-vanilla.img.xz** by clicking on download latest Image.

With the disk utility Pi imager, select the use custom menu

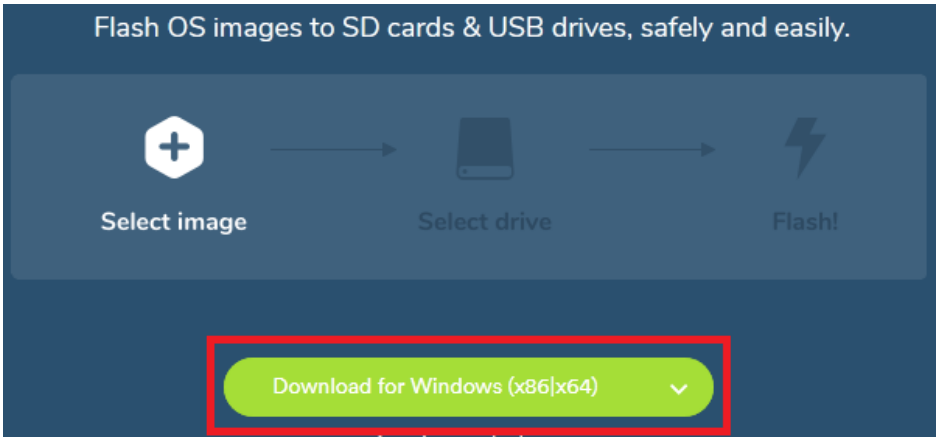


Insert the micro Sd card into a USB reader on the PC, select the SD card reader here GENERIC USB Reader, then click on WRITE.



In case of difficulty to copy the disk image, use the etcher utility, based on the same principle as Pi Imager.

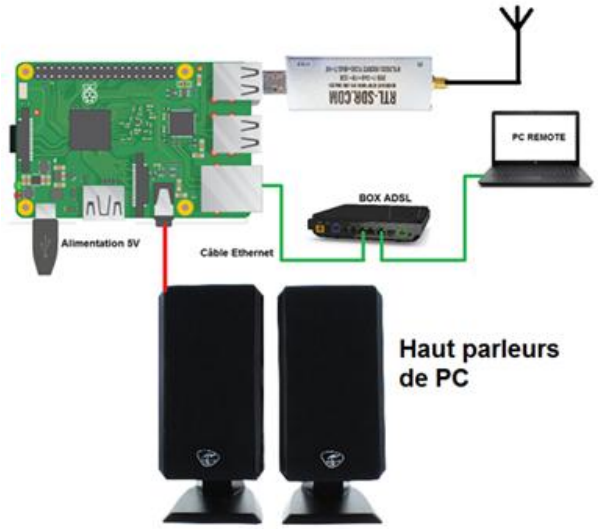
https://www.balena.io/etcher/



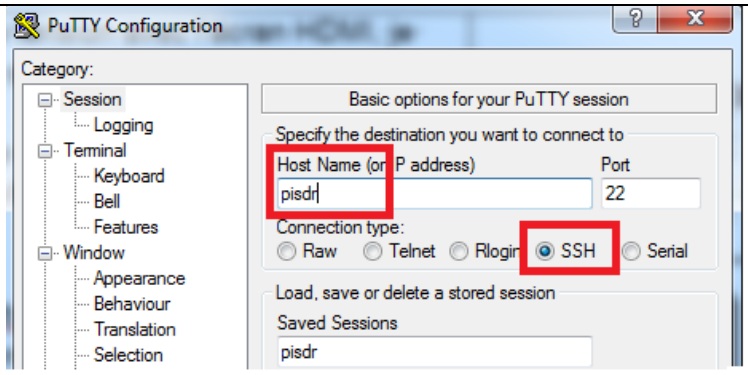
When powering up with the HDMI display, I did not see any sound on the 3.5 jack output.

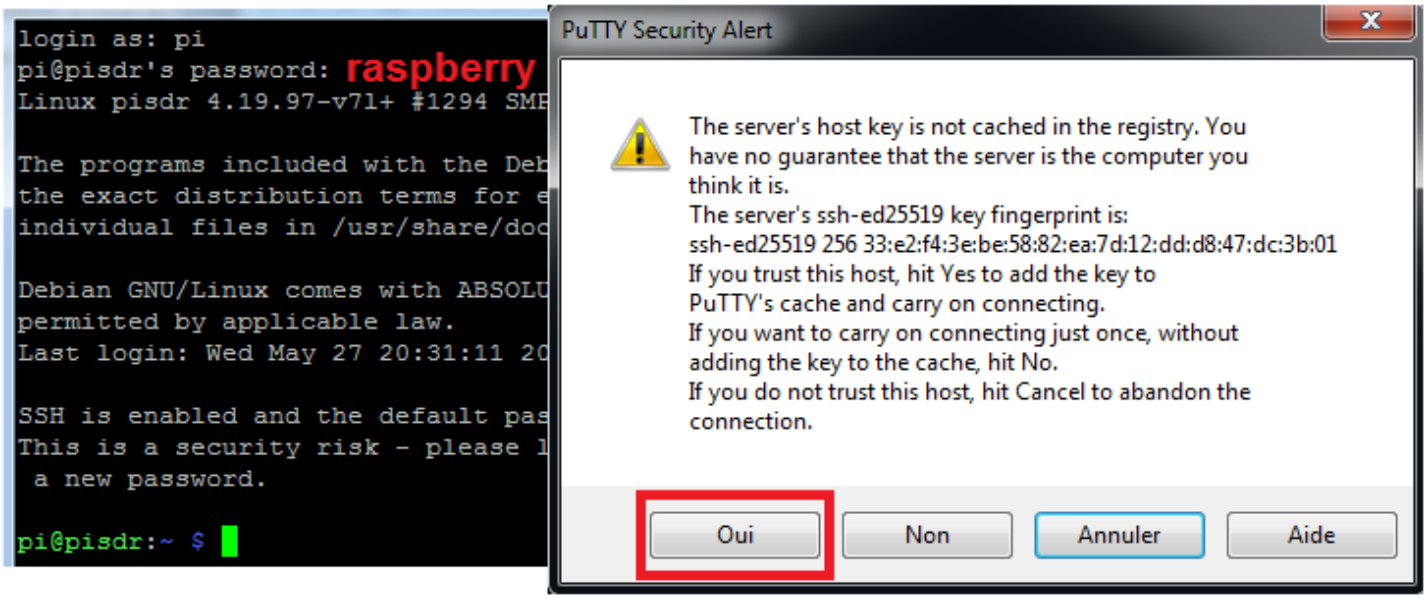
Perform the first power-up with the **HDMI display** disconnected to check the sound output on the 3.5 jack.

Use the Putty utility to connect the Raspberry Pi with an Ethernet link.



The domain name is **pisdr**, no need to search for the IP address of your Raspberry PI.





- Start by installing pavucontrol (for later)

```
sudo apt-get install pavucontrol
```

- Redirect the sound to the 3.5 jack socket

```
sudo amixer cset numid=3 1
```

```
pi@pisdr:~ $ sudo amixer cset numid=3 1
numid=3,iface=MIXER,name='PCM Playback Route'
  ; type=INTEGER,access=rw------,values=1,min=0,max=3,step=0
  : values=1
```

- Set the volume to 100%.

```
amixer sset 'Master' 100%
```

```
pi@pisdr:~ $ amixer sset 'Master' 100%
Simple mixer control 'Master',0
  Capabilities: pvolume pvolume-joined pswitch pswitch-joined
  Playback channels: Mono
  Limits: Playback 0 - 65536
  Mono: Playback 65536 [100%] [on]
```

- Check the sound output by generating a test noise.

```
aplay /usr/share/sounds/alsa/Noise.wav
```

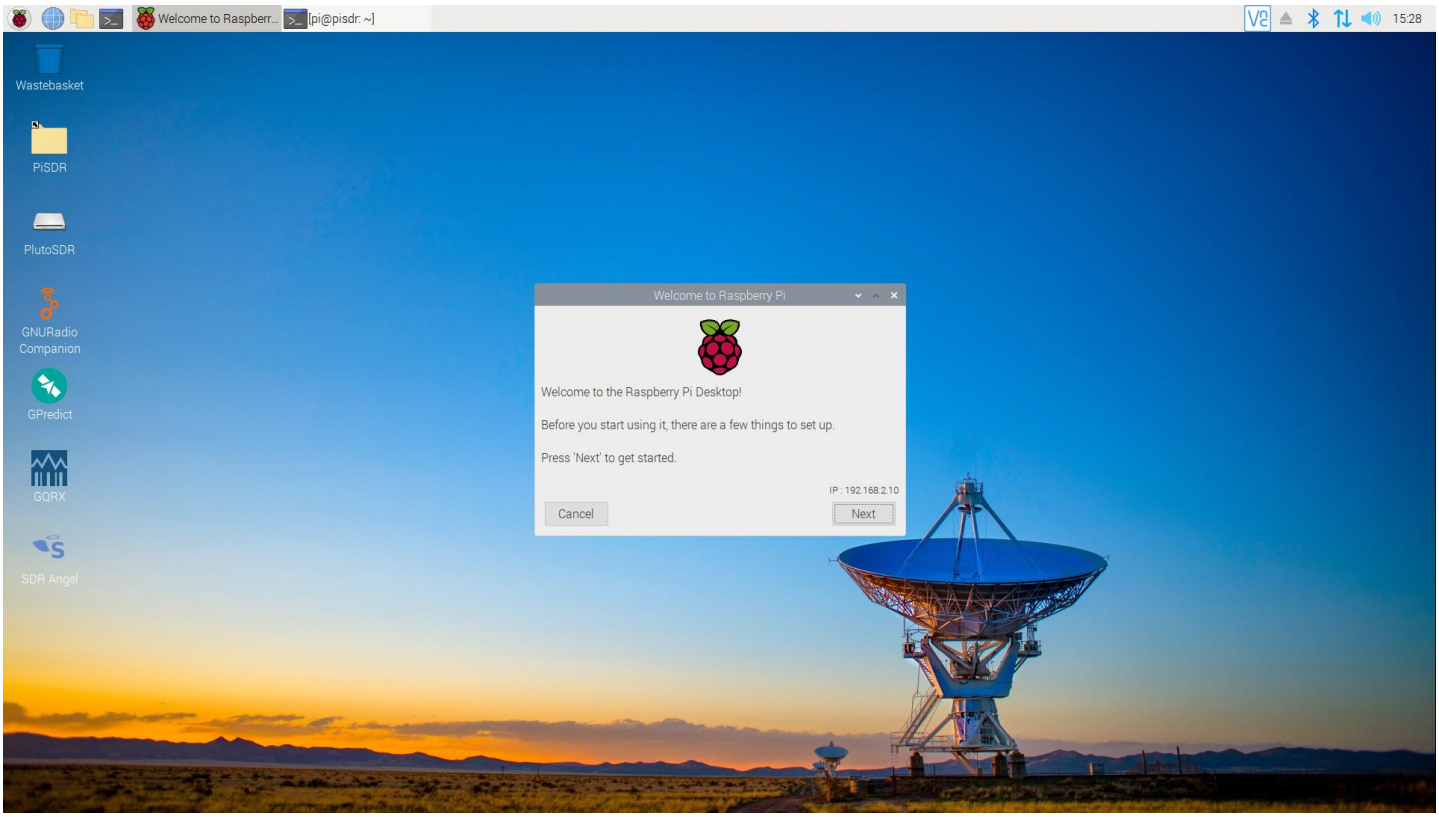
```
pi@pisdr:~ $ aplay /usr/share/sounds/alsa/Noise.wav
Lecture WAVE '/usr/share/sounds/alsa/Noise.wav' : Signed 16 bit Little Endian,
réquence 48000 Hz, Mono
```

Normally, sound should be present on the 3.5 jack output. Then stop the Raspberry Pi

```
sudo halt
```

Turn off the power while you reconnect the HDMI cable. Turn the Raspberry Pi back on using the keyboard and mouse.

You have to start by configuring the country, like the tutorial in part 1.



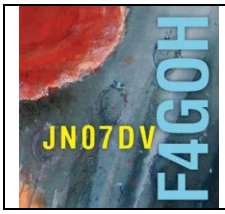
List of supported SDR Receivers	List of Installed Software
• Airspy HF+ Discovery • Airspy Mini • Airspy R2 • LimeNET Micro • LimeSDR Mini • LimeSDR USB • PlutoSDR • RTL-SDR • HackRF	• GNURadio • GQRX • LimeSuite • LimeVNA • SDRAngel • Soapy Remote • LeanSDR • VIRGO • CygnusRFI • rpitx • rtl_433 • acarsdec • Gpredict • multimon-ng
Reminder: Always connect the SDR device to the Raspberry Pi **before** running listening software.	

In the list, the SDRAngel software deserves special attention. Feel free to take the software in hand with the help of a vidéo or other internet resource. Fldigi and WSJT-X is not present in the list, you will just have to add these programs as explained in part 2.

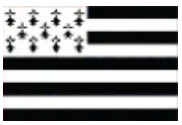
Useful link on the RTL SDR key :

https://www.surviveuk.com/wp-content/uploads/2016/07/The-Hobbyists-Guide-To-RTL-SDR-Carl-Laufer.pdf

Next part: Using the Raspbian Lite operating system without GUI. Command lines will be more important.



Raspberry Pi for HAM Radio
Part 4



Prerequisite: Discovery of Raspberry PI Part 1 (pages 2 and 3)

Summary:

Part 4: Using the Raspbian Lite operating system without GUI.

Version du 5/09/2020

4.1 Introduction.

As a HAM radio, I use the Raspberry Pi in 2 situations:

- As described in the first 3 tutorials, the Raspberry PI is connected directly to a transceiver. Using VNC viewer, I control the Raspberry remotely. This allows me to perform QSO in digital mode in a remote way, elsewhere than in the radio shack.

- The Raspberry pi is exclusively dedicated to a specific application and is switched on 24 hours a day. It can be an application managing an sdr listening server like openwebrx, an APRS igate with direwolf, the reception of weather images, or simply hosting a WEB server at home.

In the 2nd situation, the permanent use of an HDMI monitor has no interest, since the management of the Raspberry PI will be done systematically on the command line via putty or, depending on the application used, via an administration WEB page hosted in the mSd card.

As a result, there is a "light" version of Raspbian that does not manage the graphical desktop environment (all installation and configuration is done on the command line). This has the advantage of not monopolizing processor resources for the management of the graphic desktop and saving the electrical energy of the nano computer.

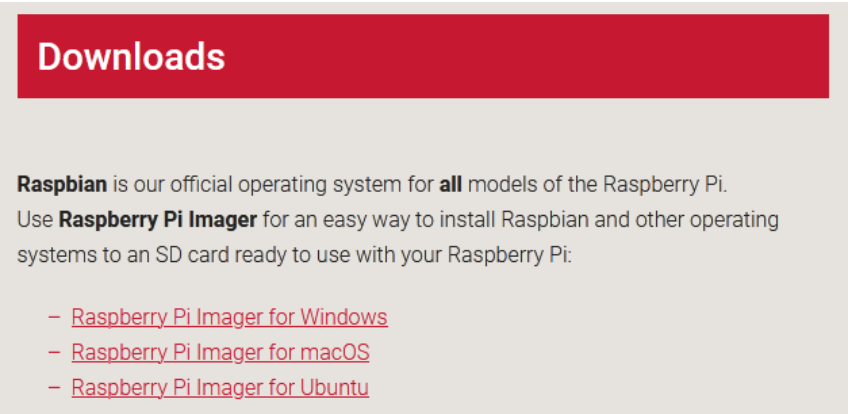
Some developers such as MW0MWZ with pistar, (DMR hotspot management) have the excellent idea of offering a "picture" ready to use with the operating system and all the software already installed. The user only has to use a copy software like etcher, without having to perform all the tedious tasks of a complete installation.

However, not all applications using RPI are ready to use in a disk image. A large majority of applications require manual installation with the prerequisite of having a Raspbian operating system already installed and configured in the correct language.

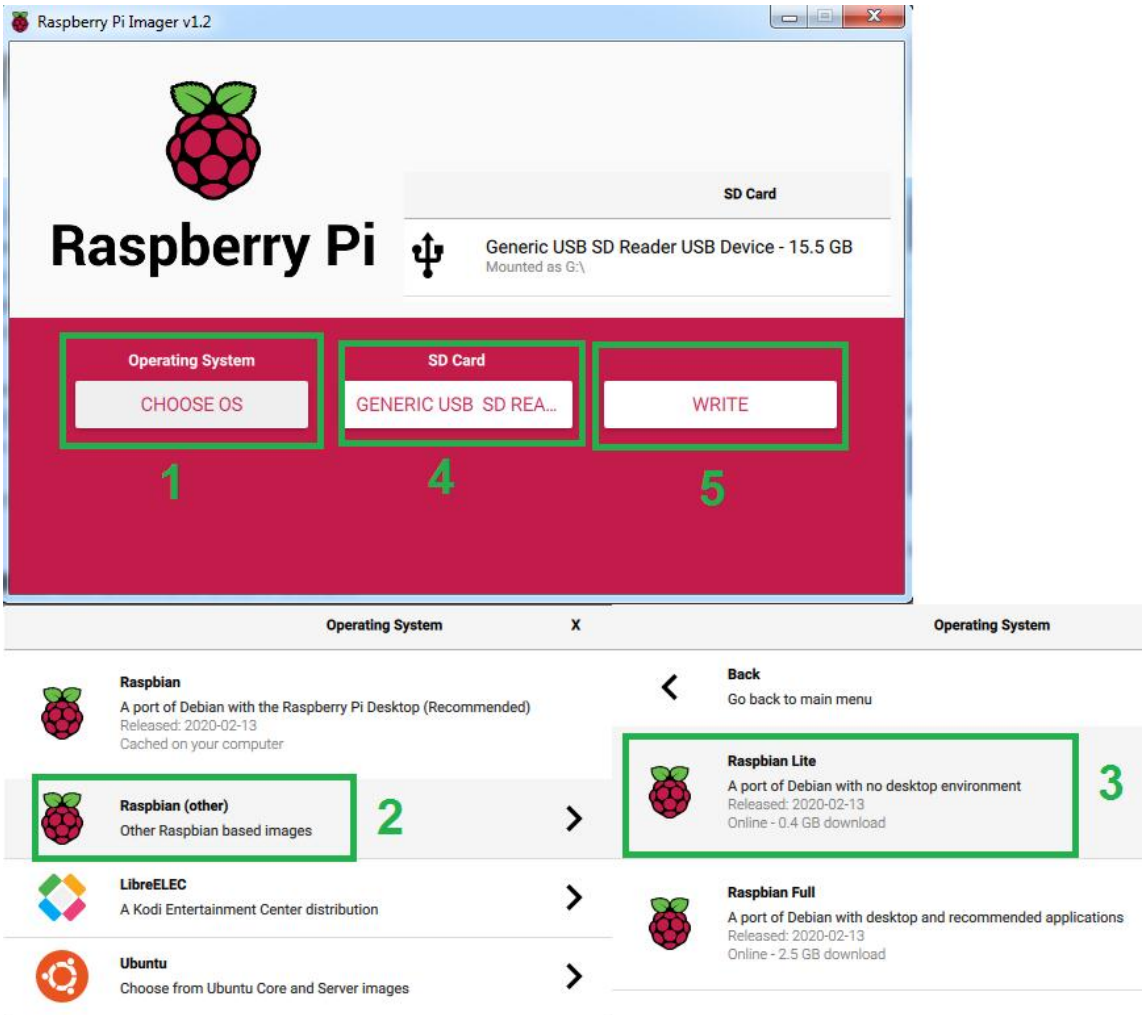
This document will explain how to install and configure Raspbian with no Desktop environment and remind the remote connection procedure with putty. Not forgetting the procedure to save the mSd card.

4.2 Operating system setup

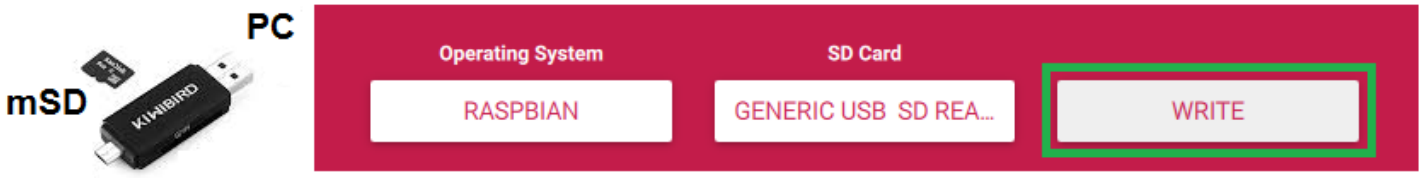
With your Computer, download the new Pi Imager operating system management utility for the Raspberry PI. https://www.raspberrypi.org/downloads/



After downloading and installing, select the **Raspbian OS with no Desktop environment**



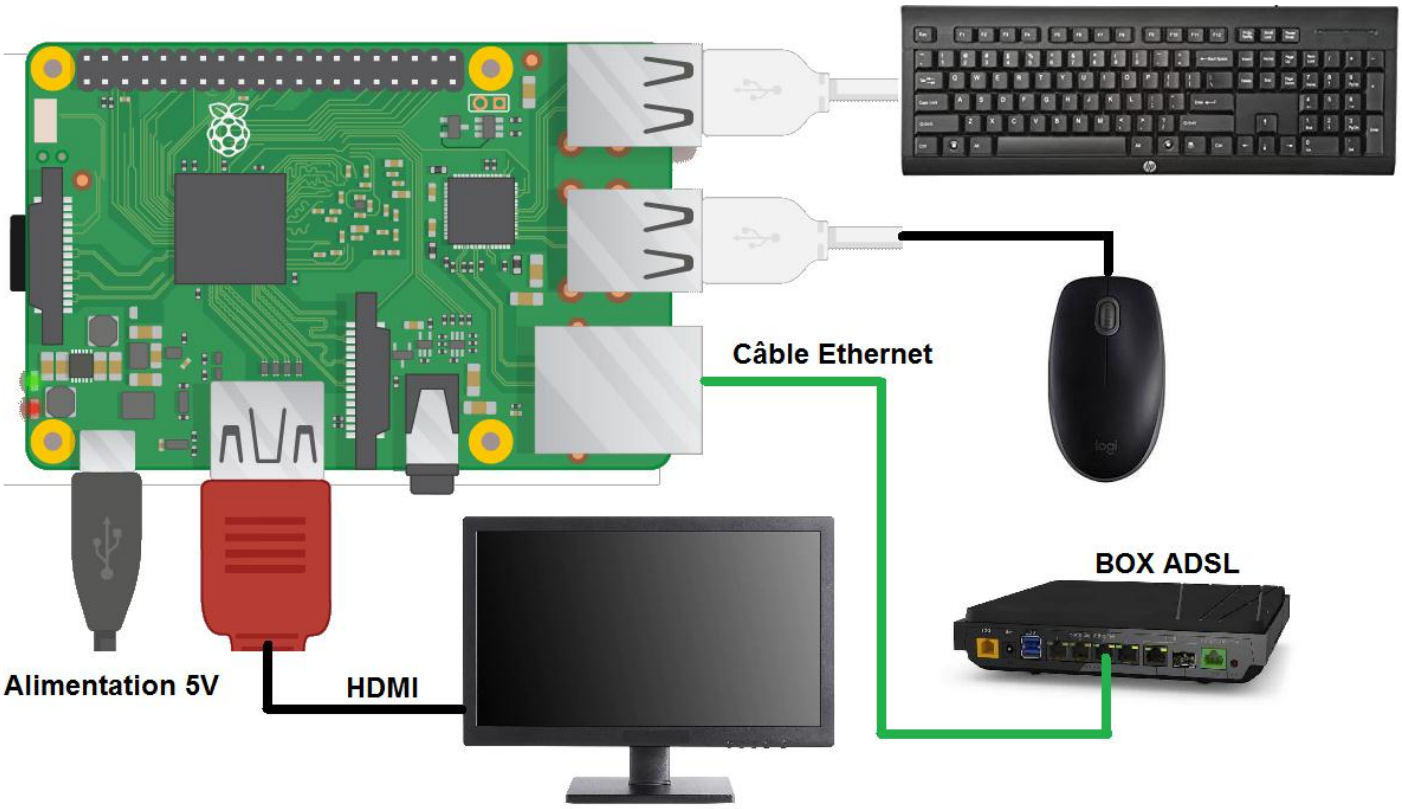
Insert the micro Sd card into a USB reader on the PC, select the SD card reader here GENERIC USB Reader, then click on WRITE.



The Pi Imager utility will automatically download and copy the selected operating system to the mSd card. The time of the operation depends on the Internet connection, you have to be patient.

4.3 Power on

Carry out the following wiring, ending with the 5V power supply of the Raspberry.

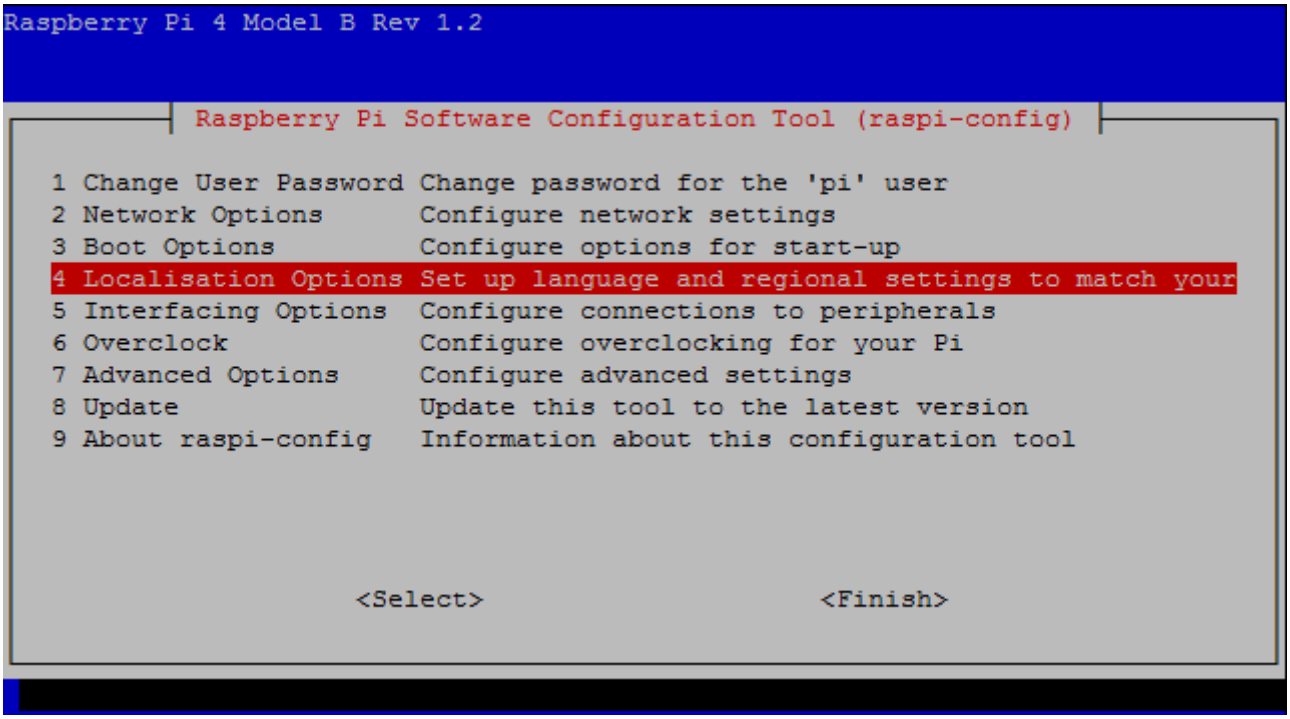


4.4 Raspi-config use **(beware french description configuration azerty keyboard)**

By default, the language at startup is English, the console appears and asks you to enter the login and password.

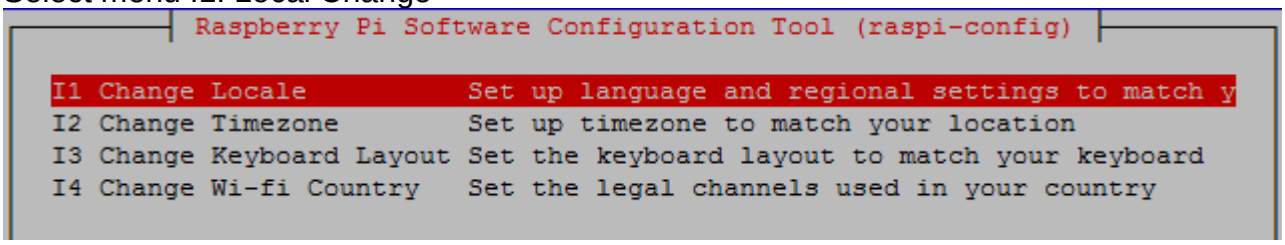
Beware the keyboard configuration is initially in qwerty. Then run the `raspi-config` configuration utility.

It will be necessary to configure your own country, the language with the right character set, the keyboard, activate the SSh, and possibly configure the wifi. (here for french setup)

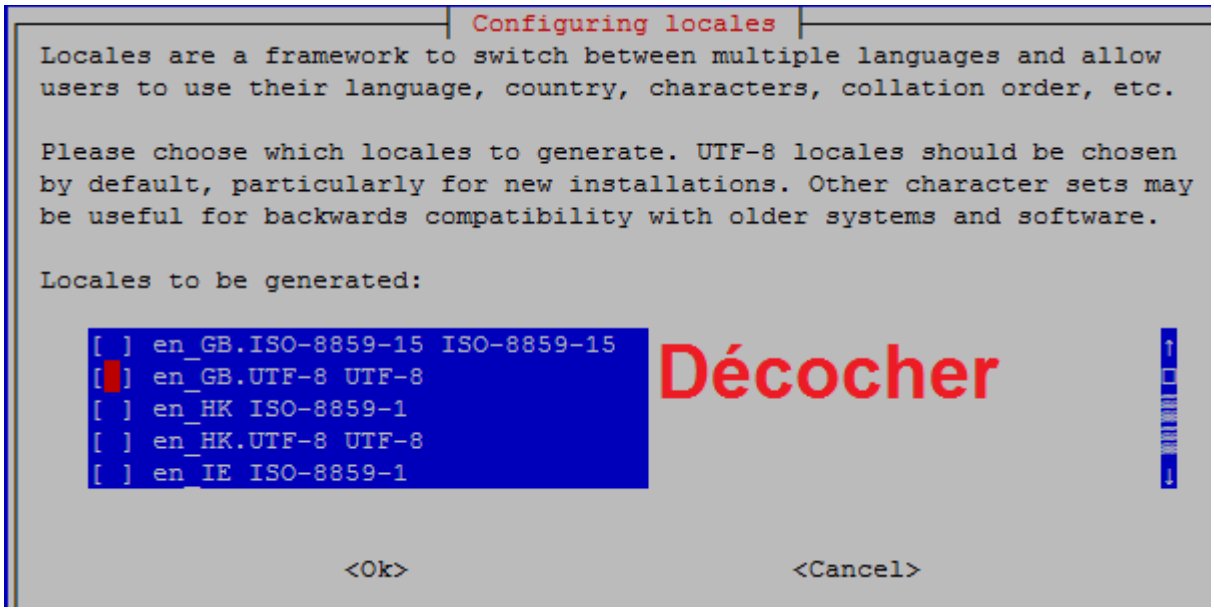


Start by selecting menu 4: Location Options. (Down arrow, then enter key)

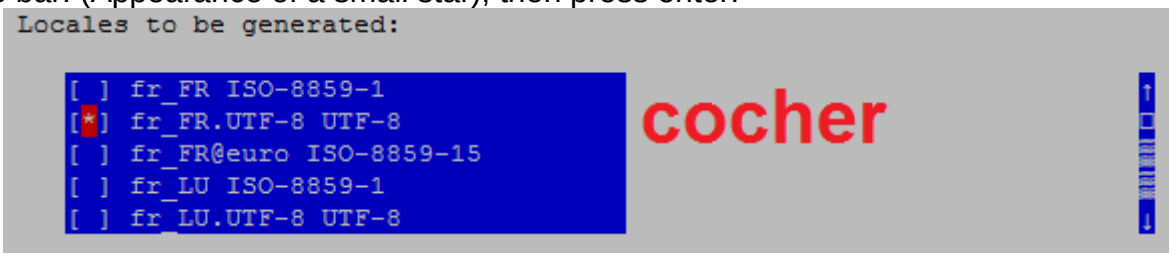
Select menu I1: Local Change



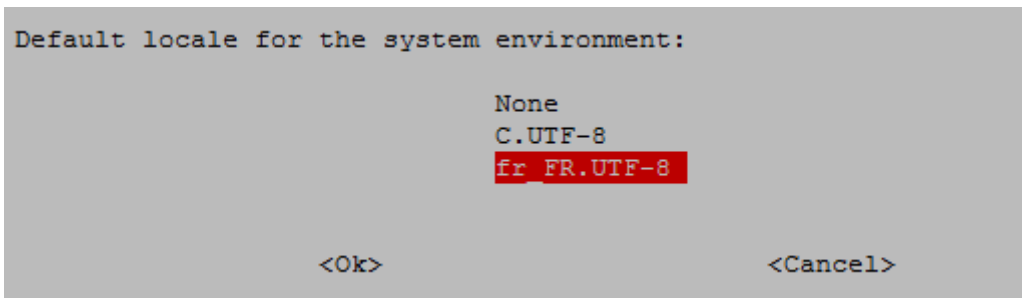
Find the line **in_GB.UTF-8 UTF-8** with the down arrow or pageDown key, uncheck it by pressing the space bar.



Find the line **fr_EN.UTF-8 UTF-8** with the arrow at the bottom of the keyboard, check it by pressing the space bar. (Appearance of a small star), then press enter.



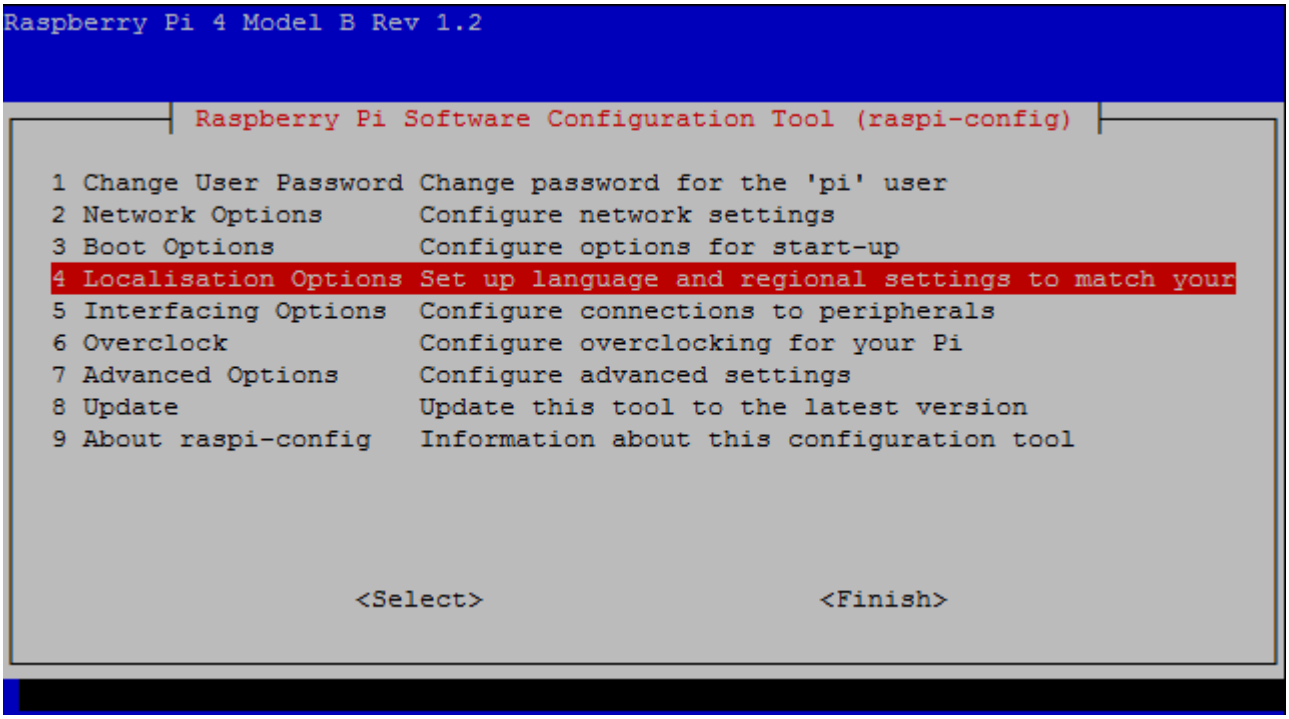
Select **fr_EN.UTF-8 UTF-8**, then press enter to return to the main menu.



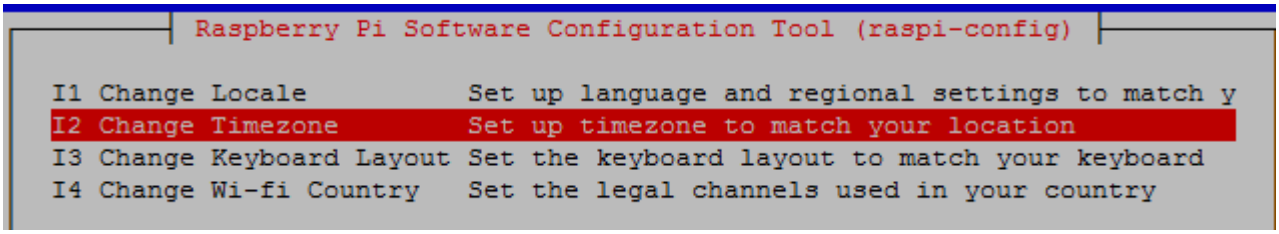
In case of false manipulation, press the tab key on the keyboard, this allows you to scan the different elements of the window and thus easily access <ok> and <Cancel>.

Note: The screenshots presented in this document are sometimes in French while the configuration on the HDMI monitor will be done in English. This does not change the following procedure.

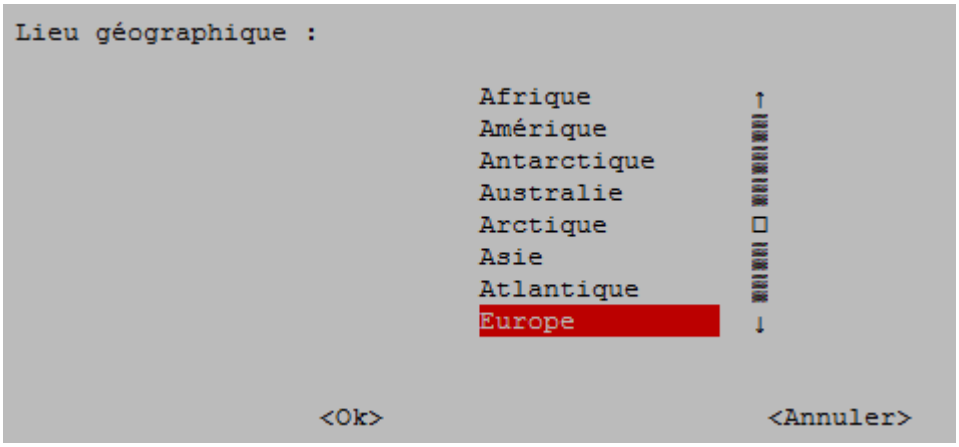
Select Menu 4: Localization Options again. (down arrow, then press enter)



Select menu I1: Local Change (down arrow, then enter key)



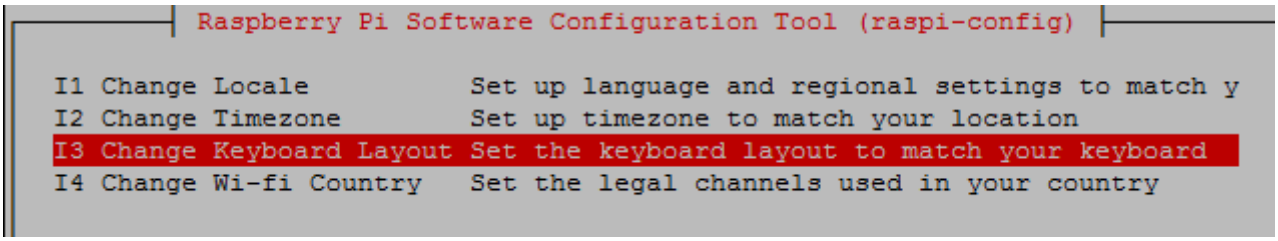
Select Europe (then press enter)



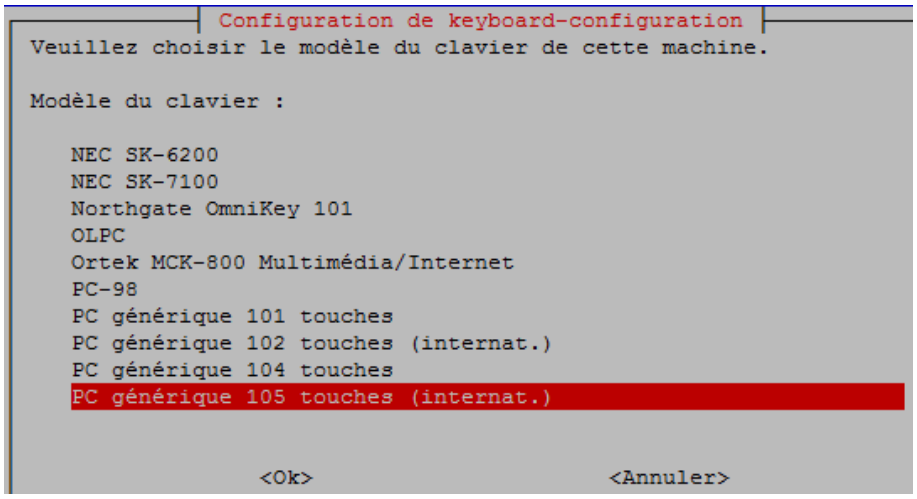
Select Paris (pressing the letter P on the keyboard goes faster) as time zone (then enter key)

Select menu 4: Location Options again. (down arrow, then enter key)

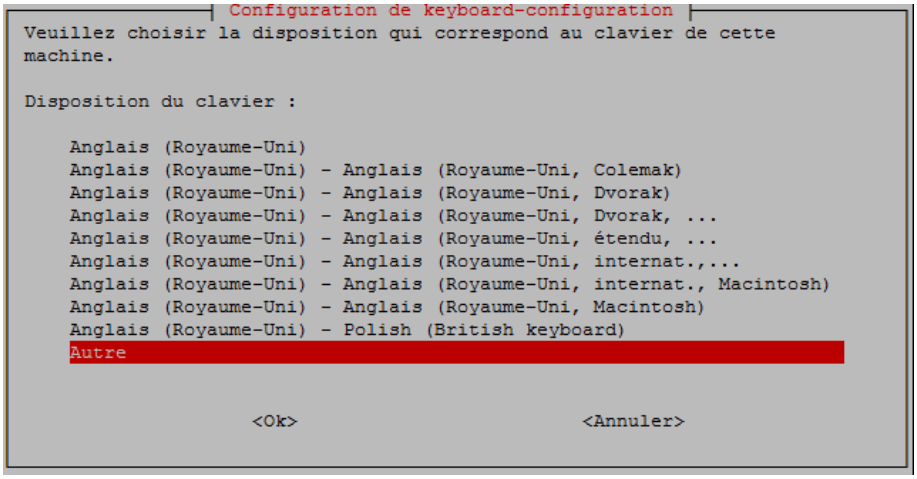
This time you need to configure the keyboard in azerty.



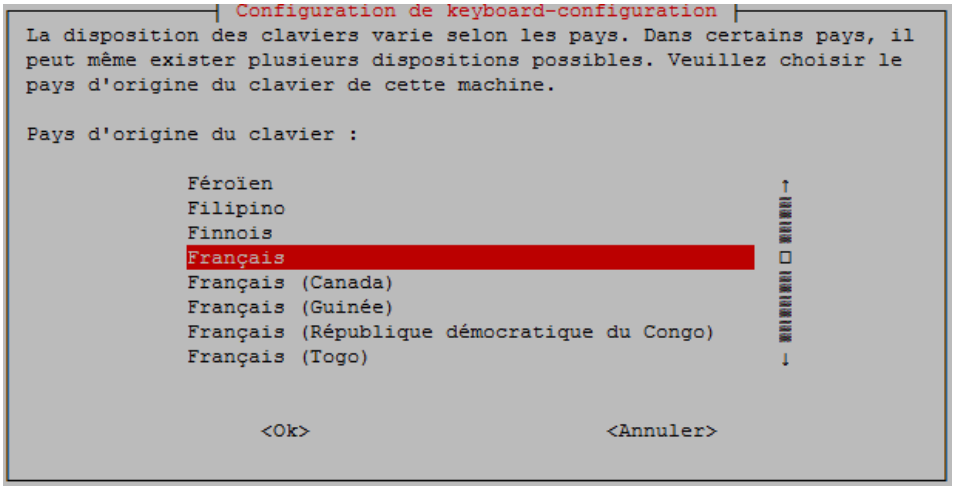
Select generic PC 105 keys. (Down arrow, then enter key)



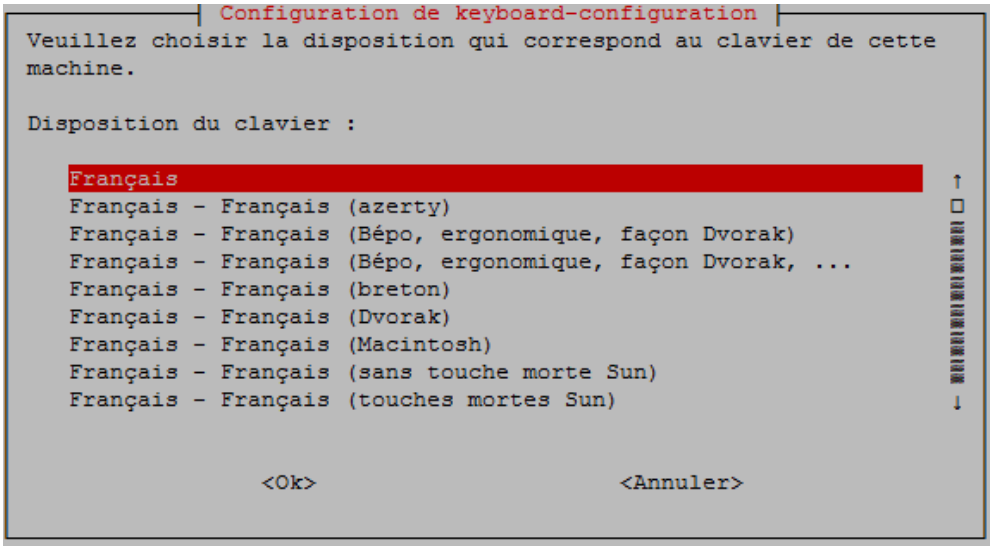
Select Other. (Then press enter)

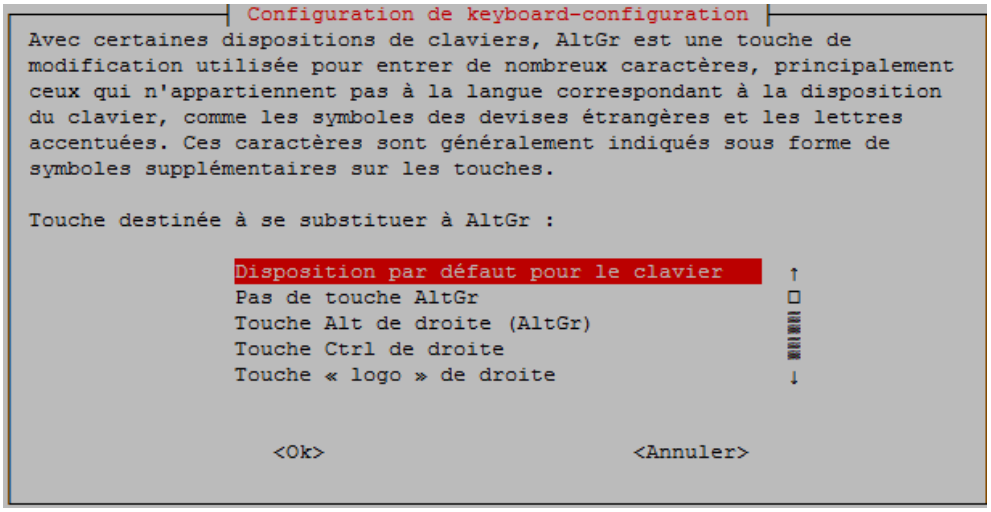


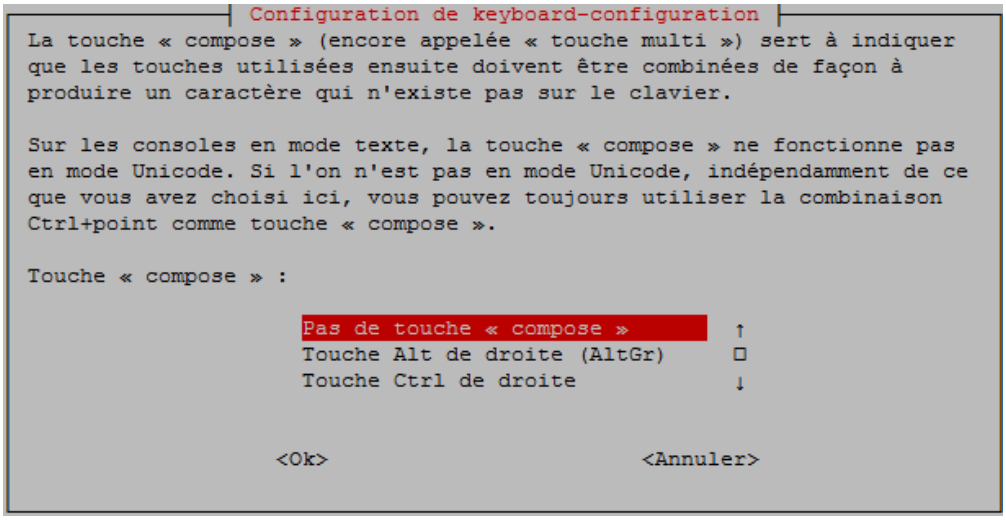
Select English. (Then press enter)



Select English again, and confirm the following steps with the enter key:

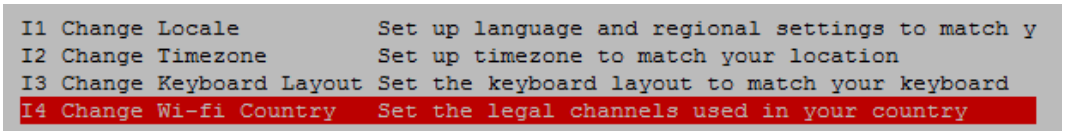


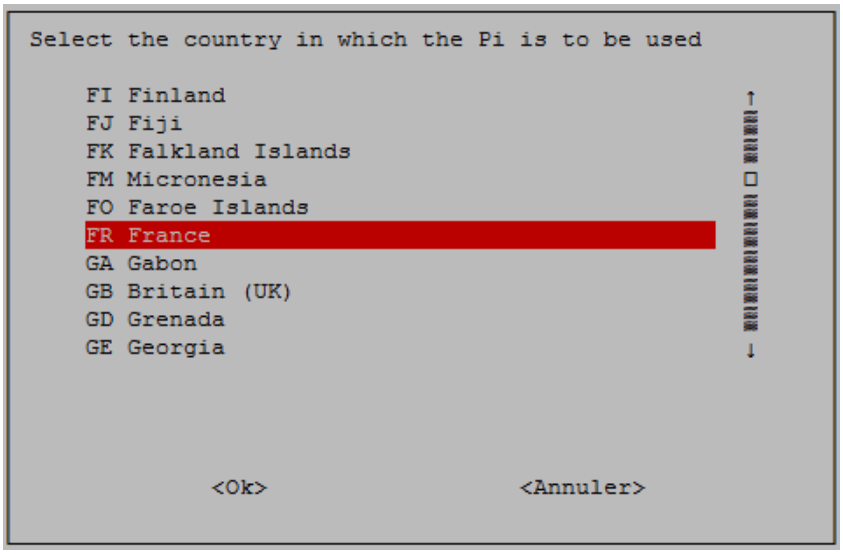

Configuration de keyboard-configuration
Avec certaines dispositions de claviers, AltGr est une touche de
modification utilisée pour entrer de nombreux caractères, principalement
ceux qui n'appartiennent pas à la langue correspondant à la disposition
du clavier, comme les symboles des devises étrangères et les lettres
accentuées. Ces caractères sont généralement indiqués sous forme de
symboles supplémentaires sur les touches.
Touche destinée à se substituer à AltGr :
Disposition par défaut pour le clavier
Pas de touche AltGr
Touche Alt de droite (AltGr)
Touche Ctrl de droite
Touche « logo » de droite
<Ok>
<Annuler>


Configuration de keyboard-configuration
La touche « compose » (encore appelée « touche multi ») sert à indiquer
que les touches utilisées ensuite doivent être combinées de façon à
produire un caractère qui n'existe pas sur le clavier.
Sur les consoles en mode texte, la touche « compose » ne fonctionne pas
en mode Unicode. Si l'on n'est pas en mode Unicode, indépendamment de ce
que vous avez choisi ici, vous pouvez toujours utiliser la combinaison
Ctrl+point comme touche « compose ».
Touche « compose » :
Pas de touche « compose »
Touche Alt de droite (AltGr)
Touche Ctrl de droite
<Ok>
<Annuler>

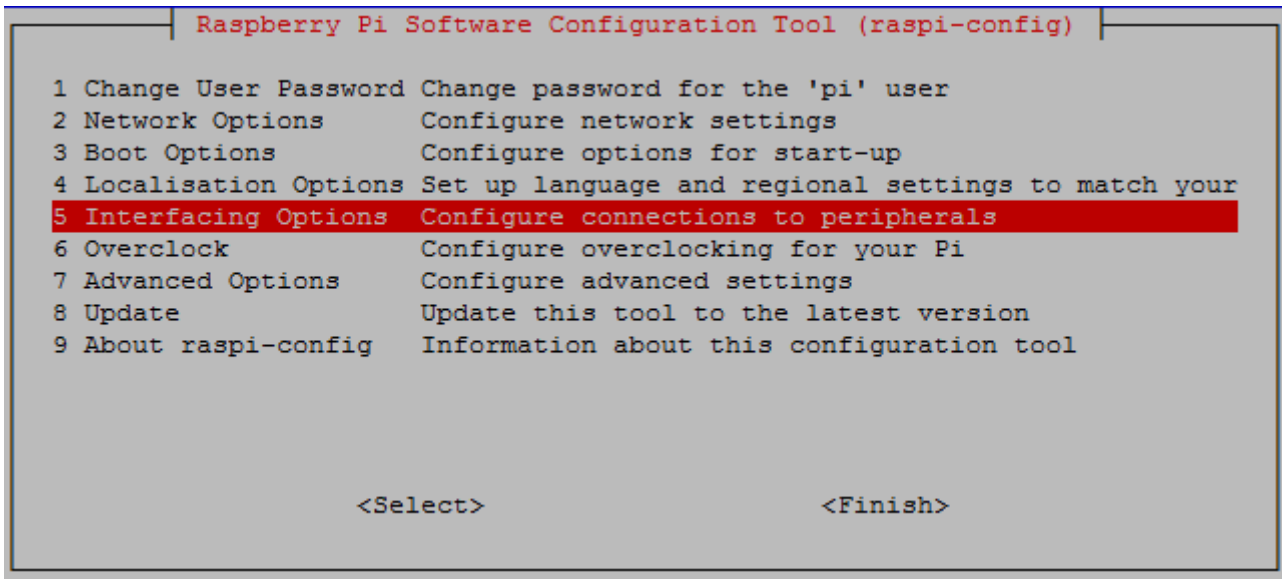
Back to the main menu, select menu 4: Location Options again.

This is the moment to configure the country of use of the WIFI

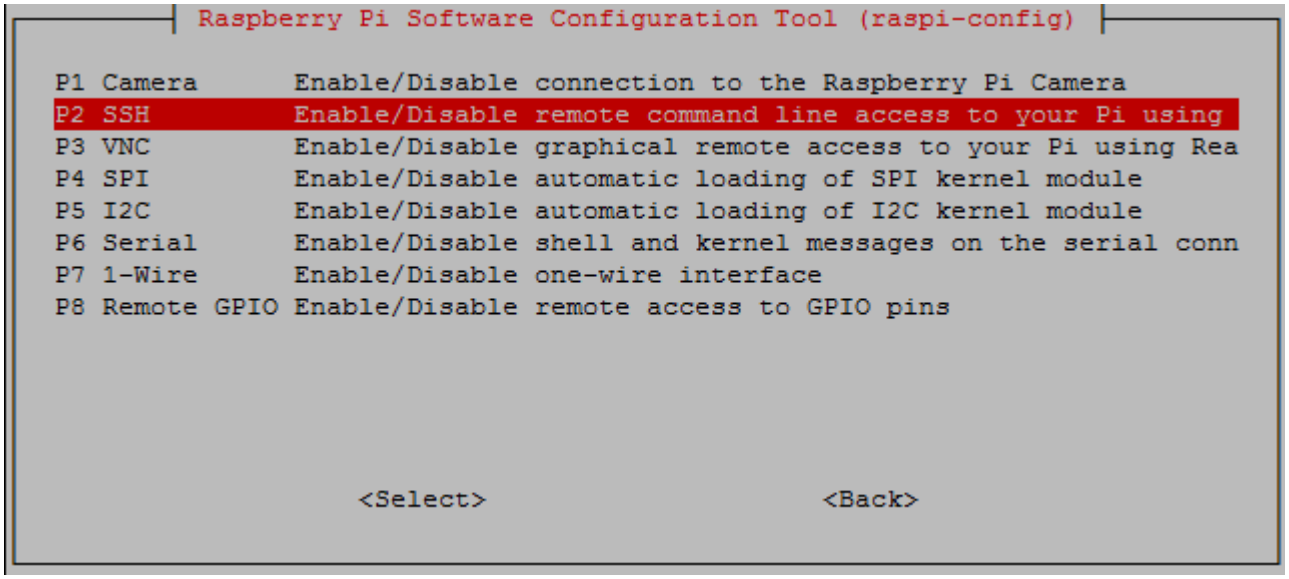




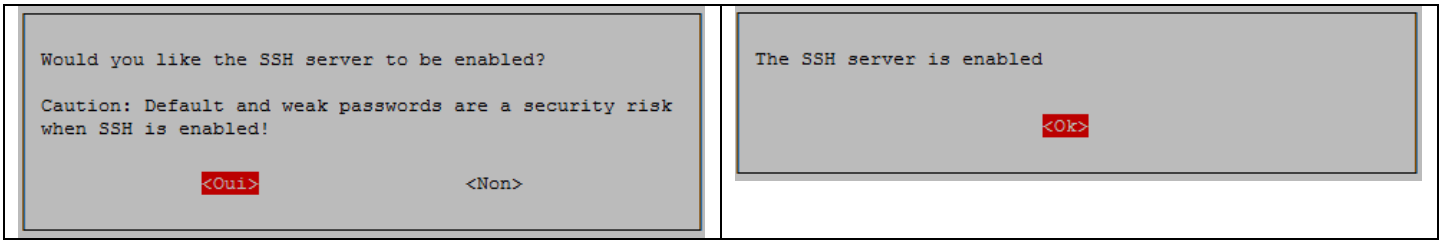
Back to the main menu, select Menu 5: Interfacing Options again.



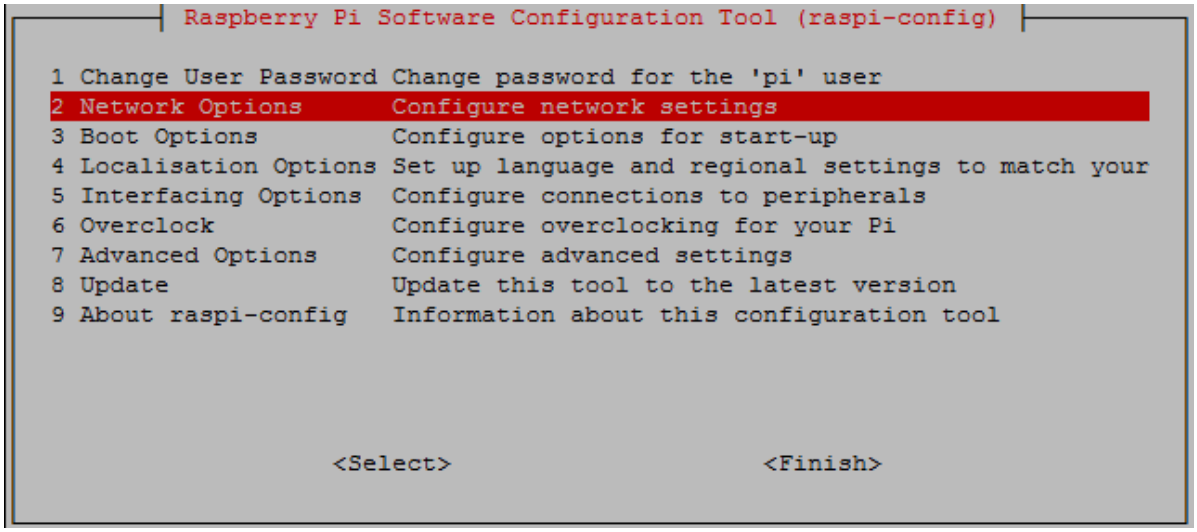
Activate SSH by selecting the line, then press enter.



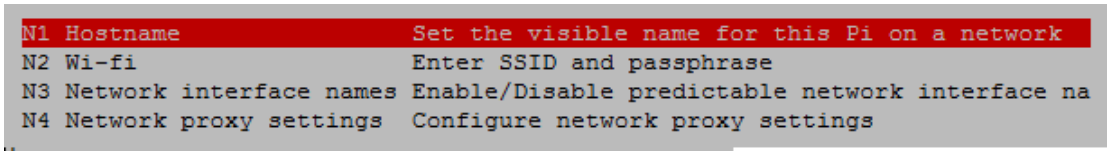
Note: When using sensors or actuators connected directly to the 40-pin GPIO interface, you often have to return to this menu to activate the communication BUS for the project. This can be for example a mini weather station using a BME280 sensor, or simply the camera on its dedicated connector.



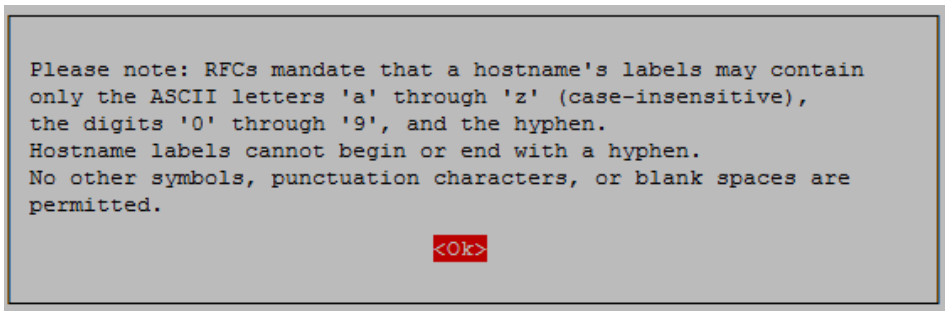
Back to the main menu, select menu 2: Network Options.



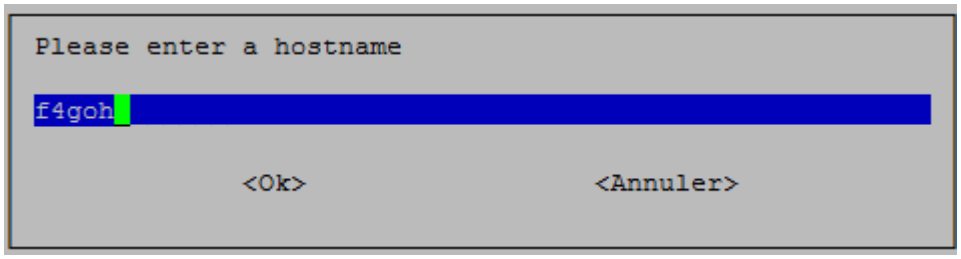
Select Hostname. (Then enter key)

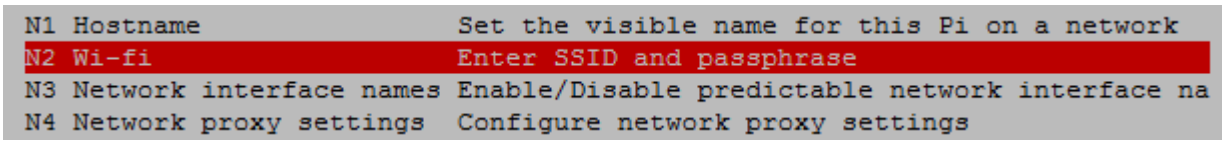


Appears an alert: lowercase characters, no spaces, etc.

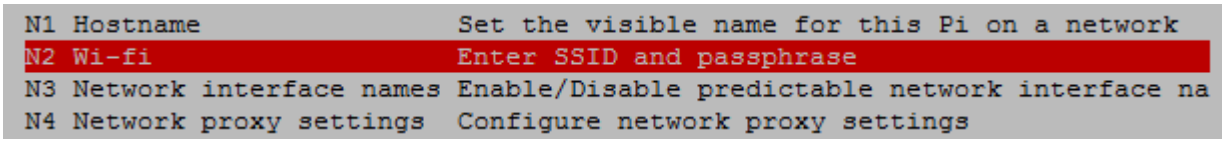


Simple, I took my callsign as a domain name



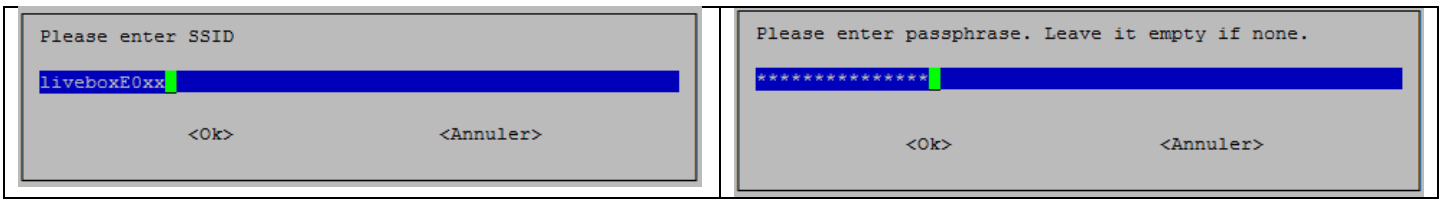


Back to the main menu, select again menu 2: Network Options, then Wi-fi,

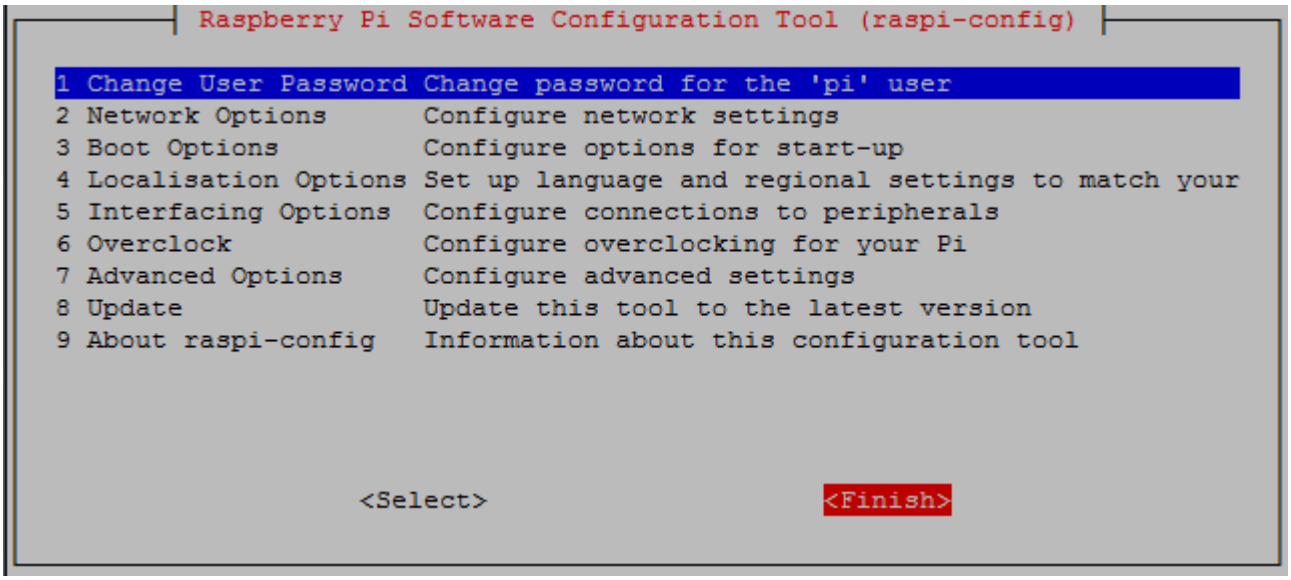


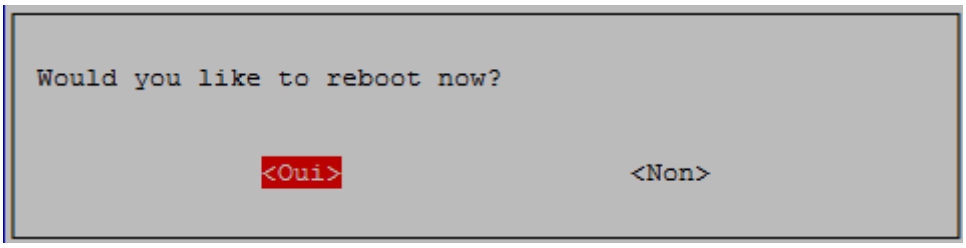
This step is optional if the internet connection is via Wi-fi.

It is necessary to know the name of your access point (ADSL Box), its password precisely and with the character

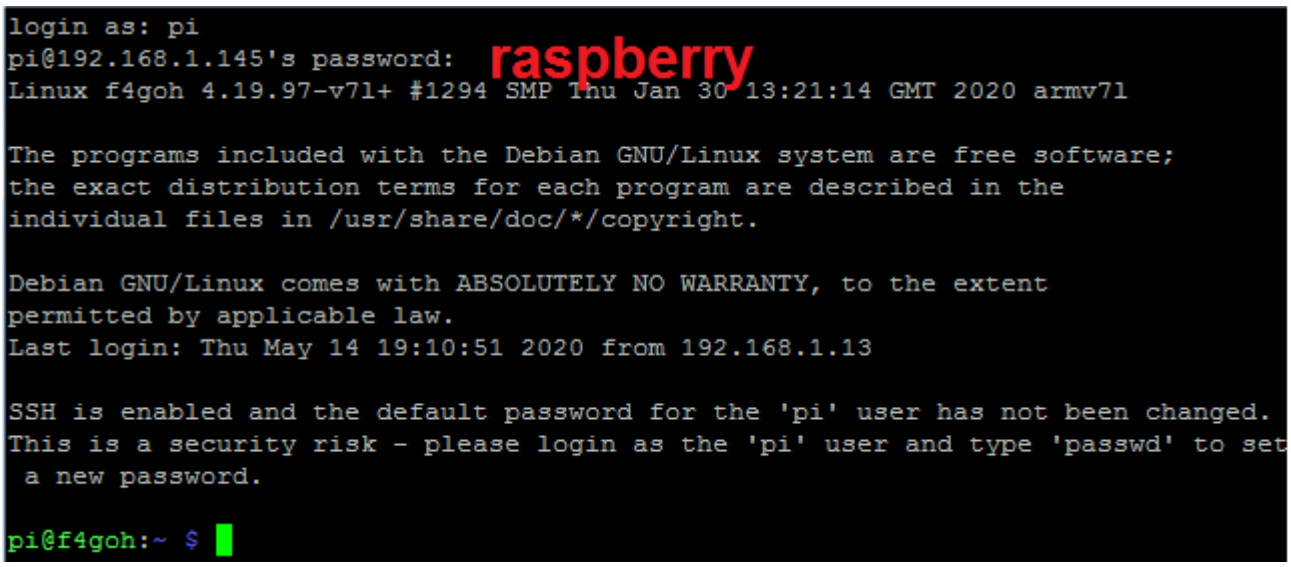


The basic configuration is finally completed, a restart is required.





The console appears and asks you to enter the login and password.



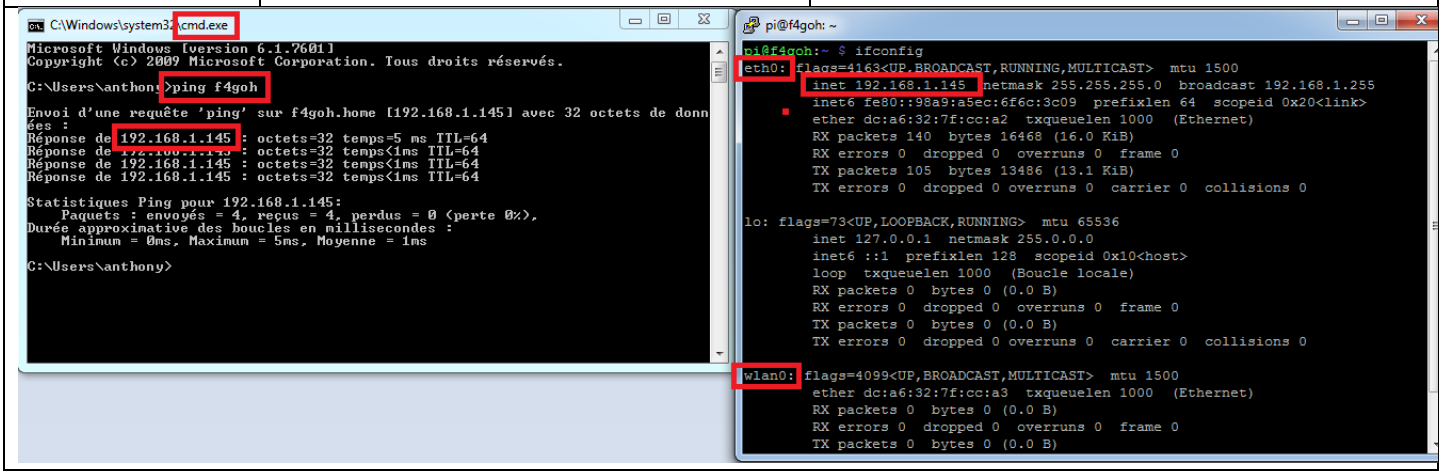
To change the default password, simply go to menu 1 of raspi-config (`Change User Password`).

4.5 Check IP address

On the PC, open the command prompt, type the `ping` command followed by the domain name chosen on page 13.

With the Raspberry Pi, display the IP address with the `ifconfig` command. The IP address must match (e.g. 192.168.1.145).

PC		**Raspberry Pi**
`ping f4goh`	Accessoires Bloc-notes Calculatrice Explorateur Windows Invite de commandes	Type: **ifconfig** followed by the enter key

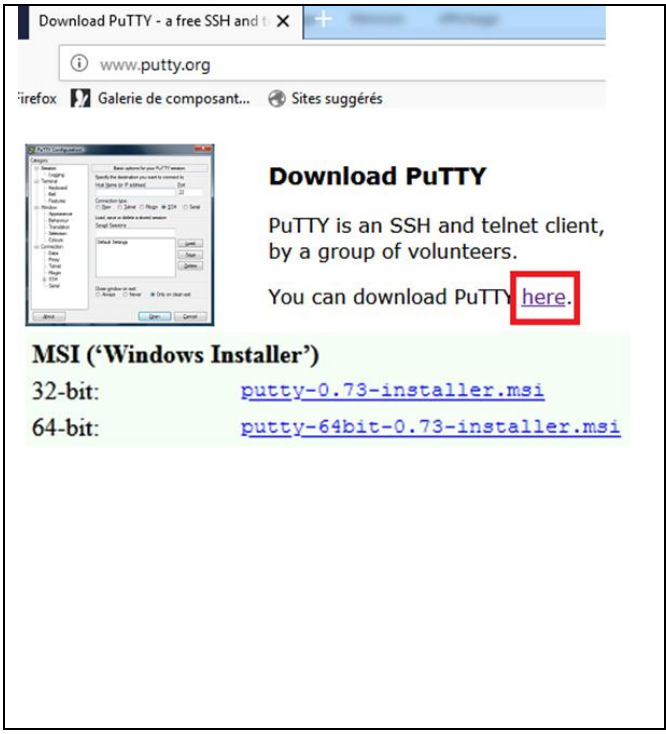


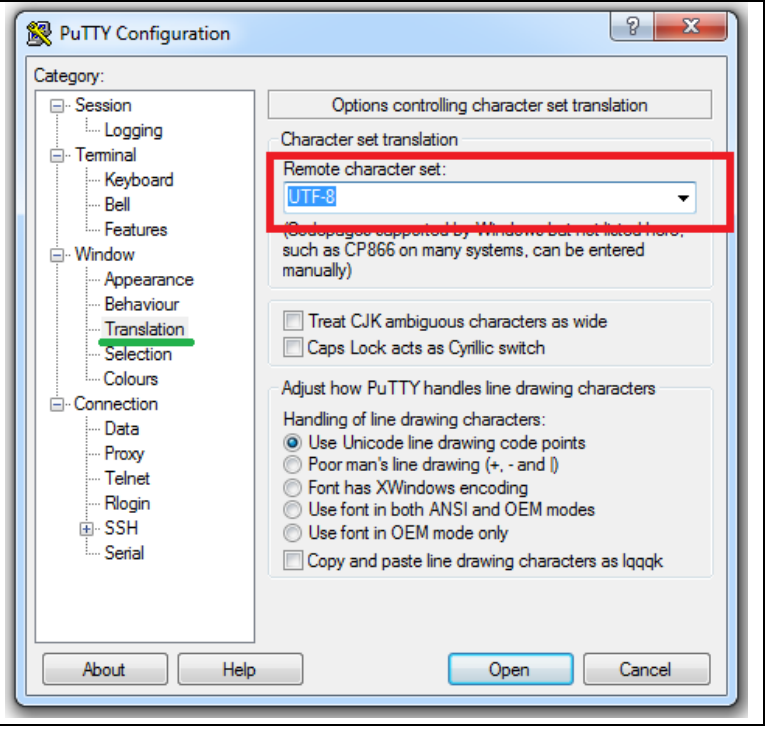
Note :

eth0 corresponds to the wired interface by Ethernet cable ;
lo : is the IP address of the local loop always 127.0.0.1 ;
wlan0: is the WIFI interface.

4.6 Putty

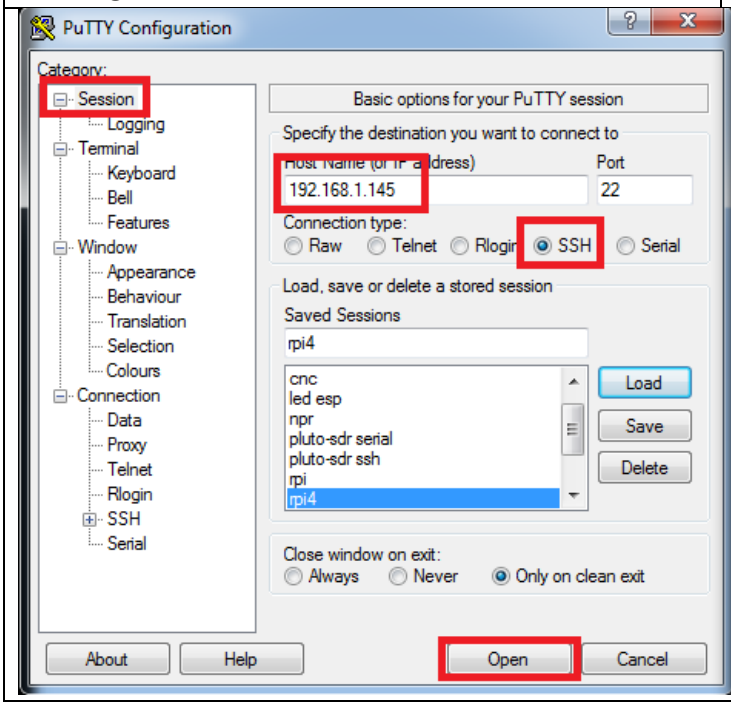
Download and install putty https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html
Then, in the Translation menu, check the **UTF-8** character set.

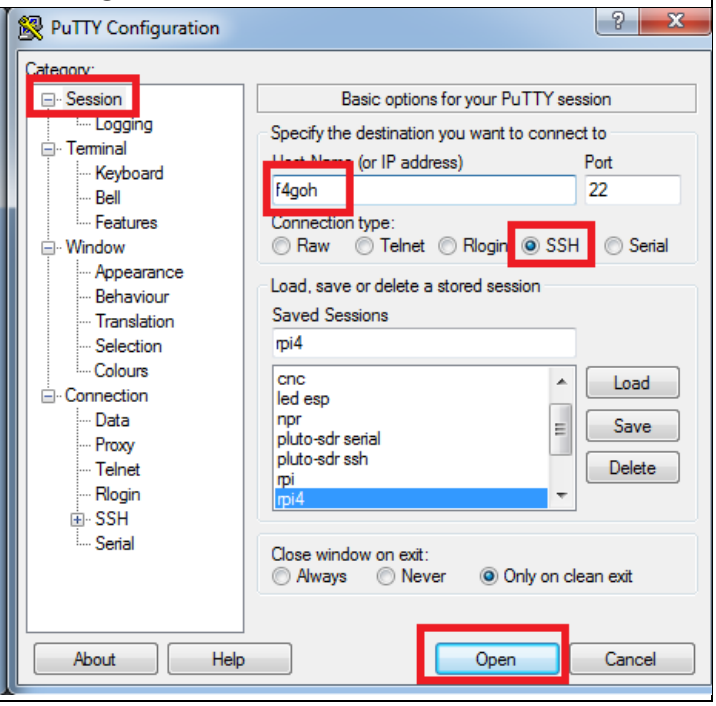


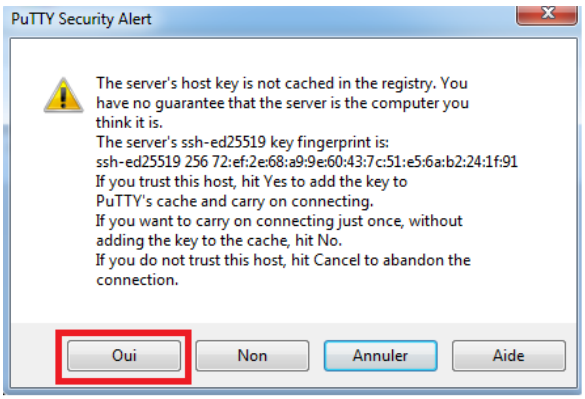


Type the domain name or IP address of the RPI, then Open

Configuration avec adresse IP	Configuration avec nom de domaine

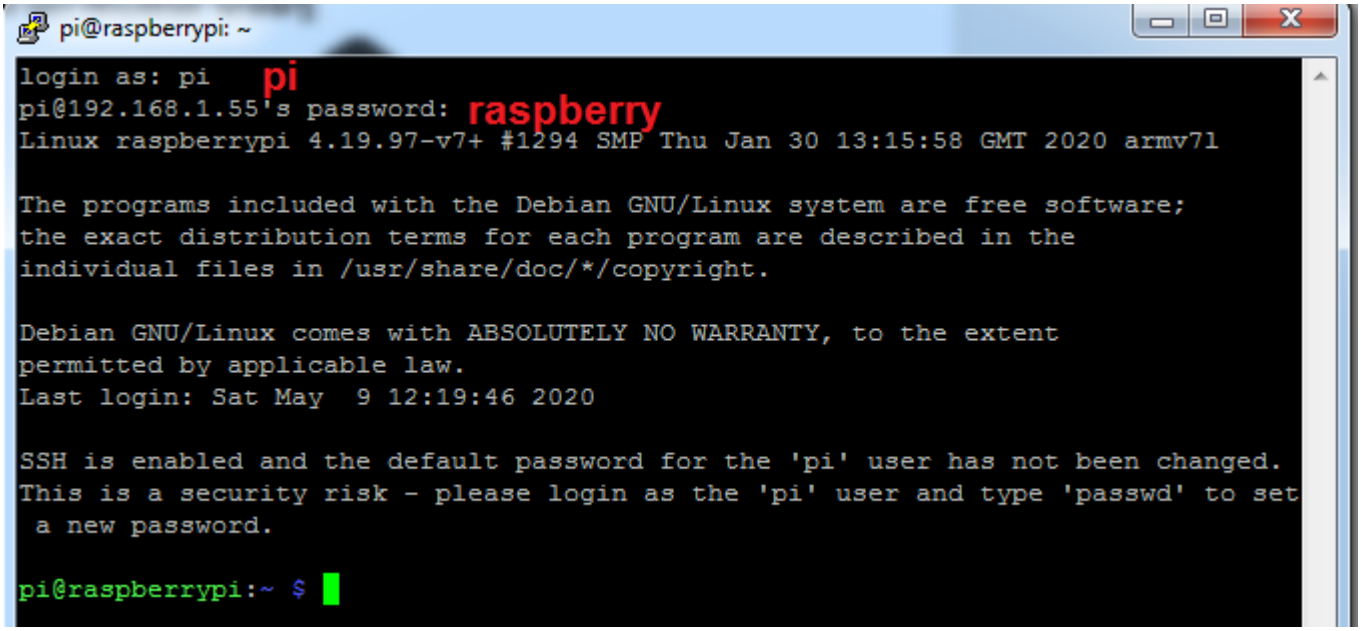






Enter the usual login and password.

Login :	**pi**
Password :	**raspberry**

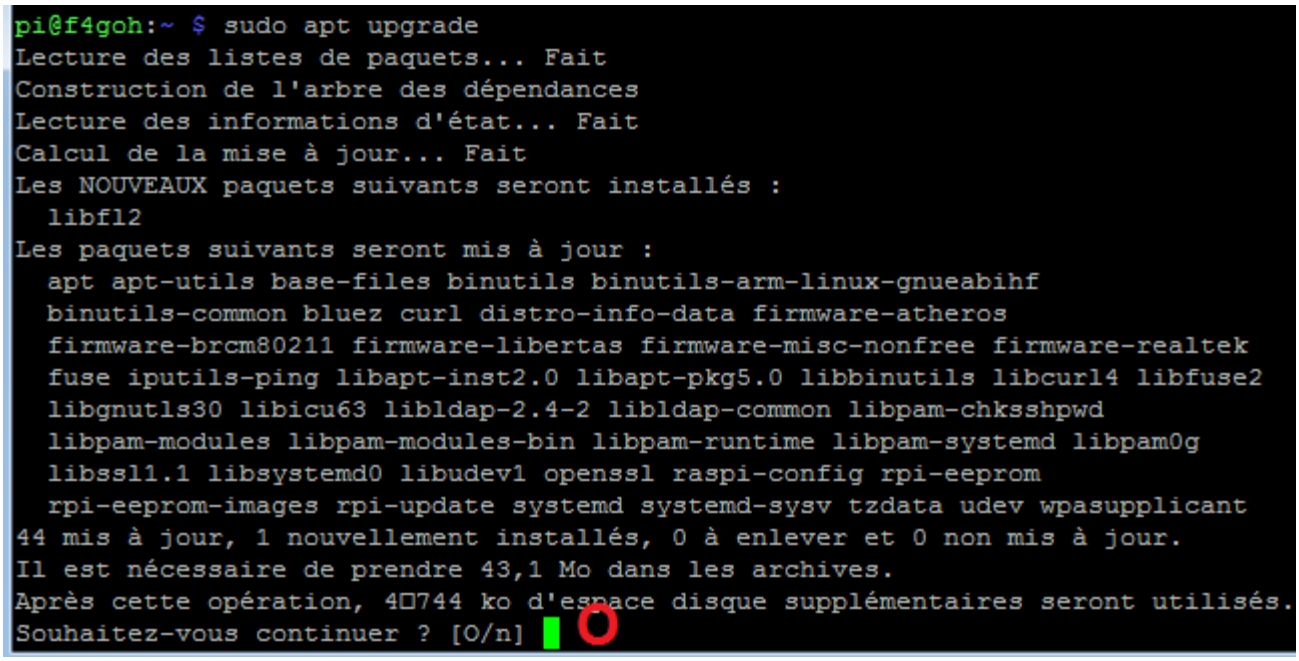


4.7 Repositories update

The **`sudo apt-get update`** command updates the list of files available in the APT repositories present in the configuration file /etc/apt/sources.list. Running this command regularly is a good practice, in order to keep your list of available packages up to date and hope to have the latest software versions. The same goes for the upgrade command to update obsolete packages in the Rasberry Pi.

```
sudo apt update
sudo apt upgrade
```

Execute this command in Putty



4.8 Backup mSD card

Take advantage of a break or lunch to make a backup of the operating system. This time there is no longer the ease of the graphic desktop to backup the operating system. However, it is possible to make the backup in command line, according to the site: framboise314

Start by installing the git utility for retrieving computer projects on github

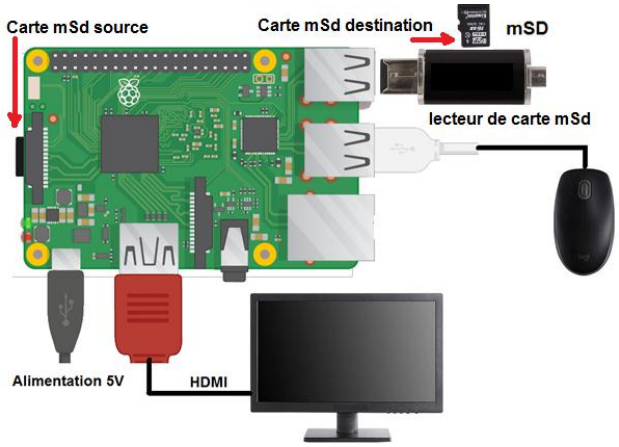
```
sudo apt install git
```

```
pi@f4goh:~ $ sudo apt install git
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
  git-man libcurl3-gnutls liberror-perl
Paquets suggérés :
  git-daemon-run | git-daemon-sysvinit git-doc git-el git-email git-gui gitk
  gitweb git-cvs git-mediawiki git-svn
Les NOUVEAUX paquets suivants seront installés :
  git git-man libcurl3-gnutls liberror-perl
0 mis à jour, 4 nouvellement installés, 0 à enlever et 0 non mis à jour.
Il est nécessaire de prendre 6□137 ko dans les archives.
Après cette opération, 32,9 Mo d'espace disque supplémentaires seront utilisés.
Souhaitez-vous continuer ? [O/n] O
```

```
git clone https://github.com/billw2/rpi-clone.git
cd rpi-clone
ls
```

```
pi@f4goh:~ $ git clone https://github.com/billw2/rpi-clone.git
Clonage dans 'rpi-clone'...
remote: Enumerating objects: 176, done.
remote: Total 176 (delta 0), reused 0 (delta 0), pack-reused 176
Réception d'objets: 100% (176/176), 86.39 KiB | 574.00 KiB/s, fait.
Résolution des deltas: 100% (68/68), fait.
pi@f4goh:~ $ ls
rpi-clone
pi@f4goh:~ $ cd rpi-clone/
pi@f4goh:~/rpi-clone $ ls
LICENSE  README.md  rpi-clone  rpi-clone-setup
```

- Insert a reader with its blank mSD card into a free USB port on the Raspberry PI.



The following command starts the backup.

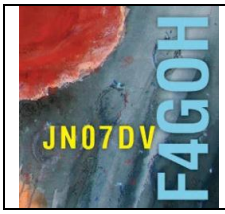
```
sudo ./rpi-clone sda
```

```
pi@f4goh:~/rpi-clone $ sudo ./rpi-clone sda

Booted disk: mmcblk0 15.5GB               Destination disk: sda 15.5GB
--------------------------------------------------------------------------
Part      Size    FS     Label           Part   Size    FS     Label
1 /boot   256.0M  fat32  --              1       14.4G  fat32  --
2 root     14.2G  ext4   rootfs
--------------------------------------------------------------------------
== Initialize: IMAGE partition table - partition number mismatch: 2 -> 1 ==
1 /boot               (52.0M used)    : MKFS  SYNC to sda1
2 root                (1.3G used)     : RESIZE  MKFS  SYNC to sda2
--------------------------------------------------------------------------
Run setup script       : no.
Verbose mode           : no.
-----------------------:
** WARNING **          : All destination disk sda data will be overwritten!
-----------------------:

Initialize and clone to the destination disk sda?  (yes/no): yes
Optional destination ext type file system label (16 chars max):

Initializing
  Imaging past partition 1 start.
  => dd if=/dev/mmcblk0 of=/dev/sda bs=1M count=8 ...
  Resizing destination disk last partition ...
    Resize success.
  Changing destination Disk ID ...
  => mkfs -t vfat -F 32  /dev/sda1 ...
  => mkfs -t ext4  /dev/sda2 ...

Syncing file systems (can take a long time)
Syncing mounted partitions:
  Mounting /dev/sda2 on /mnt/clone
  => rsync // /mnt/clone with-root-excludes ...
  Mounting /dev/sda1 on /mnt/clone/boot
  => rsync /boot/ /mnt/clone/boot  ...

Editing /mnt/clone/boot/cmdline.txt PARTUUID to use 290628bd
Editing /mnt/clone/etc/fstab PARTUUID to use 290628bd
================================
Done with clone to /dev/sda
   Start - 19:34:27    End - 19:37:33    Elapsed Time - 3:06

Cloned partitions are mounted on /mnt/clone for inspection or customizing.

Hit Enter when ready to unmount the /dev/sda partitions ...
  unmounting /mnt/clone/boot
  unmounting /mnt/clone
================================

pi@f4goh:~/rpi-clone $
```

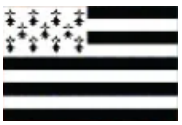
Touche entrée

Next part: Recall some elementary commands in the console, then decode a WSPR signal with an rtl-sdr key without using WSJT-X.

Next part: Installation and configuration of a sdr listening server. (openwebrx)



Raspberry Pi for HAM Radio
Part 5



Prerequisite: Raspberry PI Discovery Part 4

Summary:

Part 5: Basic commands, file access rights and WSPR.

Version du 5/09/2020 V1.0

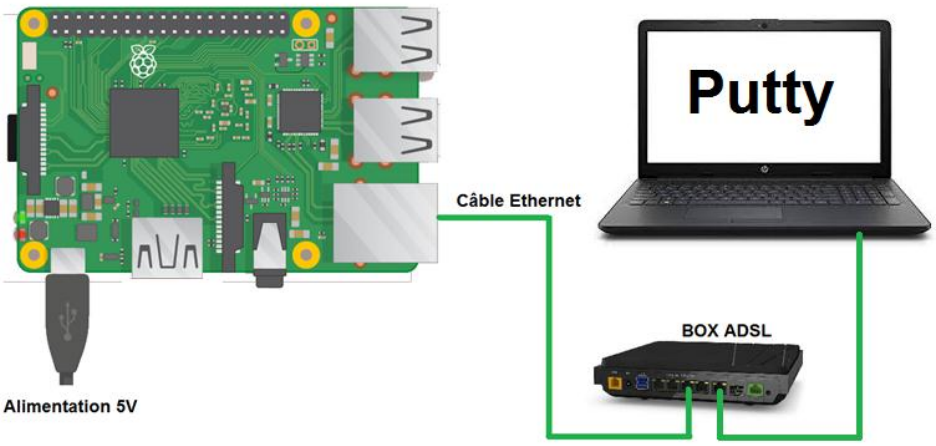
5.1 Introduction.

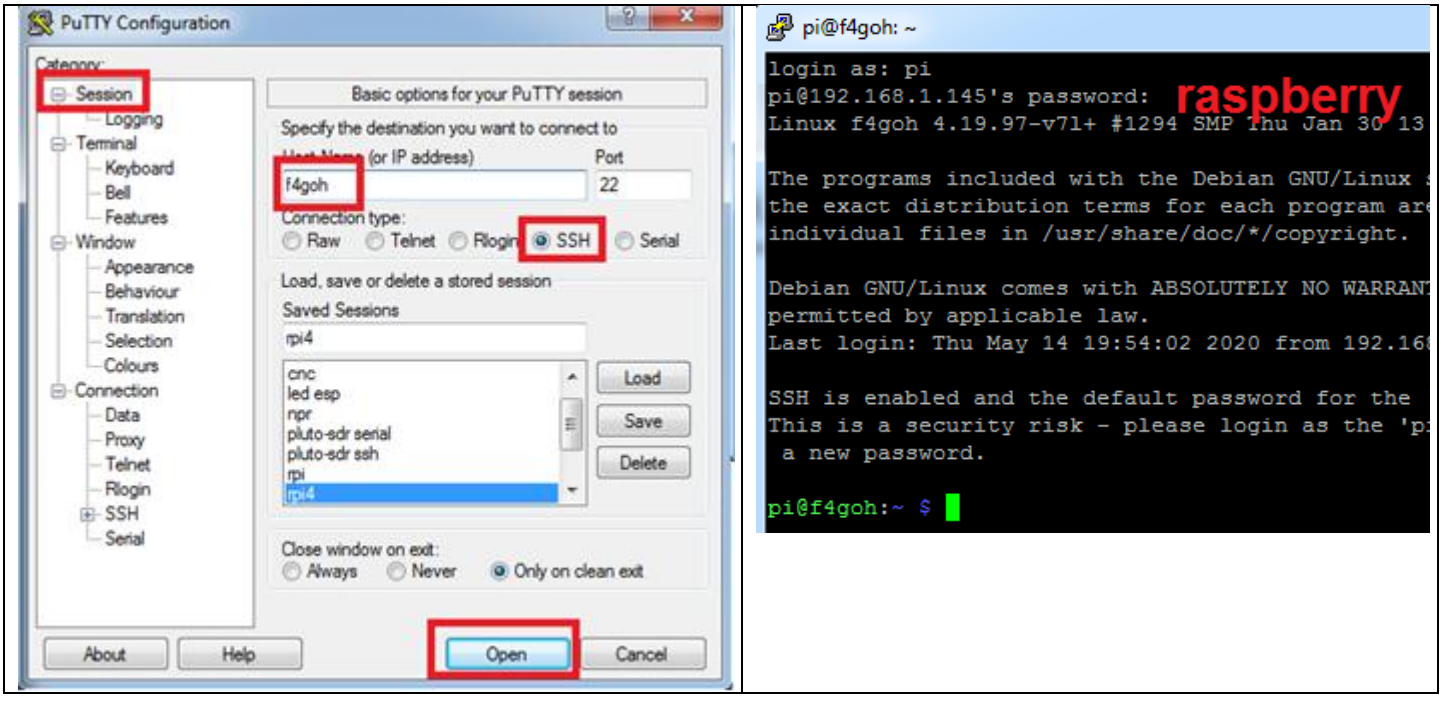
The objective of this tutorial is to be able to get by with a minimum of Linux command lines, i.e. :

- Access and explore the different directories ;
- Create a directory,
- Create, modify,
- Copy, move or delete a file,
- Changing the rights of a file
- Make a file executable and know how to execute it ;
- Switch to super user mode.

These commands will be seen through concrete examples. This can serve as a basis for the following, the interest being not to feel helpless when installing projects around the rtl-sdr receiver key.

All the following manipulations will be done in SSh connection with PuTTY. No more need for HDMI screen or keyboard/mouse.

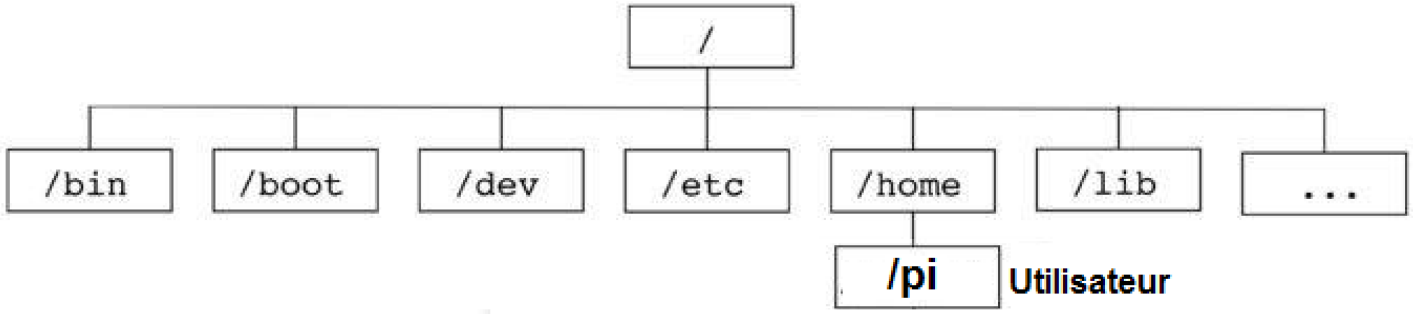




5.2 Linux structure folders

When we talk about tree structure, we are talking about hierarchy and the way files and directories are organized on an operating system. The organization of folders on a computer is often compared to a tree. The base of your tree is what is called the "root".
For example, on Windows, the root is often "C:\" which corresponds to the hard drive "C".

On Linux, the root is "/". A sequence of files starting with "/" starts from the base of your tree structure.




```
pi@f4goh:~ $ ls /
bin   dev  home  lost+found  mnt  proc  run   srv  tmp  var
boot  etc  lib   media       opt  root  sbin  sys  usr
```

When installing Raspbian, the user pi is the default login. It is always located in home. This allows you to create other users.

- / => Root, it contains the main directories
- /bin => Executables essential to the system, usable by all users (ls pwd cp)
- /boot => files allowing Linux to boot
- /dev => Entry point for all peripherals (hard disk, screen, partition, TTY consoles)
- /etc => contains the commands and files needed by the system administrator
- /home => User's personal directory
- /lib => contains the shared libraries essential to the system at boot time
- /media => contains mounting points for temporary partitions (USB sticks, data partitions)
- /opt => Generic directory for the installation of programs compiled by the administrator (specific software not present in the repositories)
- /proc => does not physically exist on a disk, it is created by the kernel in memory. This partition is used to give information about the system.
- /root => Administrator's home directory.
- /sbin => Contains essential system programs that can be used by the admin only.
- /srv => This is a data directory for various services (storage of FTP account documents, or web site pages).
- /tmp => Temporary files directory
- /usr => Contains installed programs (/usr/bin) with their libraries (/usr/lib)
- /var => contains variable data (log files) but sometimes databases (/var/lib/mysql) and website pages (/var/www/html)

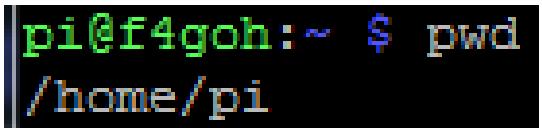
In the rest of the tutorial we will always use pi as user.

5.3 The basic commands

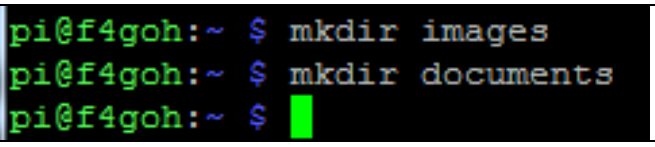
Unix commands are a word or a phrase that indicates a sequence of commands to execute on the computer, they consist of a name, can take one or more options and parameters.

pwd displays the path to the directory where the user is located. Its name in English means "print working directory". This command is very useful when you don't know which directory you are in.

pwd

	Name of the current directory:	pi
	Absolute path: (from \)	/home/pi

`mkdir` allows to create directories. The command is an abbreviation of make directory.

	Crée un répertoire image Crée un répertoire documents

`ls` is a command to list the contents of a directory (abbreviation of list in English).

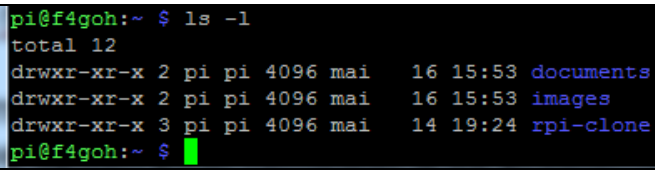
It is used in the form: ls {options} {parameters}

ls

	Lists the contents of the current directory (pi directory). The two previously created directories and the rpi-clone backup utility installed in part 4.

Nevertheless this command lists the files in my current directory without any additional information, so I need to give it an option to fix this problem. I will give it the "-l" option (Lima hyphen). Note that all options in all commands always start with a '-'.

ls -l

	Lists the contents of the current directory with more details.

`cd` (abbreviation for change directory) is a command to change the current directory.

cd documents

Note :
In the command line, you must use the auto-completion with the TAB key to the maximum. In the example below, the goal is to move in the documents directory.

`pi@f4goh:~ $ cd d`	I start by typing cd, space key, letter d then the tab key.	
`pi@f4goh:~ $ cd documents/`	The word "documents" appears directly without having to type it.	
`pi@f4goh:~/documents $`	Simply confirm with the enter key.	

pwd

`pi@f4goh:~/documents $ pwd`	documents
`/home/pi/documents`	/home/pi/documents
`pi@f4goh:~/documents $`	

To go back, always use the command cd, space key, point point.

cd ..

`pi@f4goh:~/documents $ cd..`	J'ai oublié un espace entre cd et les deux points !!
`-bash: cd.. : commande introuvable`	
`pi@f4goh:~/documents $ cd ..`	Voila, cela fonctionne.
`pi@f4goh:~ $`	

I now move to the images directory

cd images

`pi@f4goh:~ $ cd images/`	cd i " TAB key, followed by the enter key ".
`pi@f4goh:~/images $`	

Now I want to go directly back to the documents directory.

1st possibility -> relative path :

```
cd ../documents
```

`pi@f4goh:~/images $ cd ../documents/` `pi@f4goh:~/documents $`	I go back to the previous directory, then I move to the documents directory in a single line.

2nd possibility -> absolute path :

```
cd /home/pi/documents
```

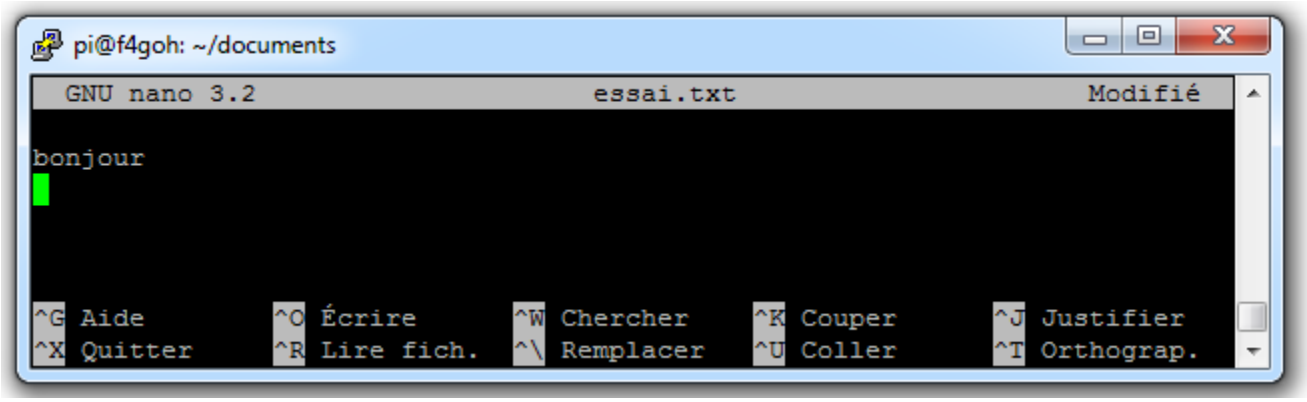
`pi@f4goh:~/images $ cd /home/pi/documents/` `pi@f4goh:~/documents $`	I move in the documents directory taking as reference the root /.

We will now create a text file in the documents directory with the nano utility.
Nano is a basic text editor that allows you to edit plain text files, without formatting (bold, italic, underline...). Under Windows, an identical text editor is available, the Notepad.

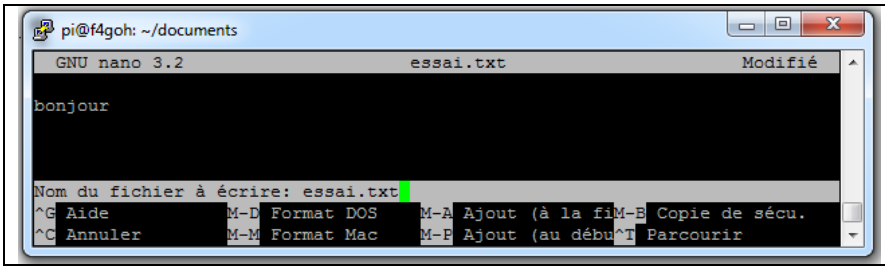
```
nano essai.txt
```

`pi@f4goh:~/documents $ nano essai.txt`	Creates and edits the essai.txt file

Type any text, e.g. hello (bonjour).



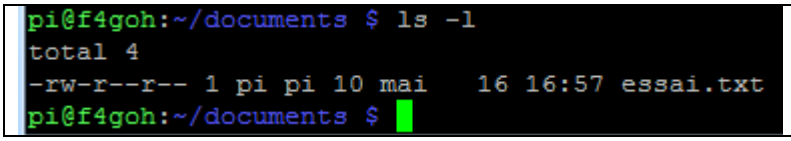
Save the file using the Ctrl + o keys.

	Enter key to confirm the recording.

Exit the editor using Ctrl + x.

Lists the contents of the directory, the file essai.txt appears.

```
ls -l
```

	Lists the contents of the current directory with more details.

`cp` (in reference term copy) is a command to copy files and directories.

I want to copy the essai.txt file to the “images” directory

```
pi@f4goh:~/documents $ cp essai.txt ../images
pi@f4goh:~/documents $ ls ../images/
essai.txt
```

Copy the `essai.txt` file indicating the destination directory.
Lists the contents of the image directory to see if the file is well copied.

`mv` (in reference to the term move) allows you to move files and directories. It also allows to rename a file or a directory.

I want to move the essai.txt file to the /home/pi directory using the absolute path.

```
pi@f4goh:~/documents $ mv essai.txt /home/pi/
pi@f4goh:~/documents $ ls
pi@f4goh:~/documents $
```

Move the `essai.txt` file indicating the destination directory.
Lists the contents of the image directory to see if the file is missing

I now go back to the /home/pi directory.

To go back to the base directory "/home/pi", always use the command `cd`, space key, point point.

```
cd ..
```

OR

```
cd /home/pi
```

Or go back to the default directory

```
cd ~
```

`rm` (in reference to remove) is a command to delete files and directories.

I delete the documents and images directory

```
rm -r documents
rm -r images
```

```
pi@f4goh:~ $ rm -r documents/
pi@f4goh:~ $ rm -r images/
pi@f4goh:~ $ ls
essai.txt  rpi-clone
pi@f4goh:~ $
```

Deleting both directories
Lists the image content to see if both directories are deleted.

Note: the dash `-r` option is mandatory for a directory. For a file there is no need for a dash -r. The file essai.txt is used for the rest and will be deleted later.

5.4 Access rights on files

On a Linux system, every file and folder have an owner and rights. Users are grouped into 3 groups: Owner (u), group (g) and other users (o). These groups make it possible to give different rights to each user. The different rights that can be granted are: read (r, 4), write (w, 2) and execute (x, 1).

.

Les différents droits		
	File	**Directory**
Read	See the content	List content
Write	Modify the content	Add or delete an item
Execute	Execute	Getting through

Let's observe the access rights of the file essai.txt

```
pi@f4goh:~ $ ls -l essai.txt
-rw-r--r-- 1 pi pi 10 mai   16 16:57 essai.txt
pi@f4goh:~ $
```

- (File)	**read**	**write**	**not execute**	**read**	**not write**	**not execute**	**read**	**not write**	**not Execute**
	r	w	-	r	-	-	r	-	-
d (Folder)	**4**	**2**	**0**	**4**	**0**	**0**	**4**	**0**	**0**
	Owner User **(u)**			**Group (g)**			Other users **(o)**		

The file essai.txt is in read and write access for its owner, i.e. the user whose login is pi. On the other hand, the file is read-only for the group and the other users.

The modification of the rights of a file or a directory is done in two ways: Absolute or relative.

Example of absolute change:

```
chmod 660 essai.txt
```

```
pi@f4goh:~ $ ls -l essai.txt
-rw-r--r-- 1 pi pi 10 mai   16 16:57 essai.txt
pi@f4goh:~ $ chmod 660 essai.txt
pi@f4goh:~ $ ls -l essai.txt
-rw-rw---- 1 pi pi 10 mai   16 16:57 essai.txt
pi@f4goh:~ $
```

Each digit of the number 660 is encoded in octal.

- (File)	read	write	not execute	read	write	not execute	Not read	Not write	not execute
	r	w	x	r	w	x	-	-	x
d (Folder)	**4**	**2**	**0**	**4**	**2**	**0**	**0**	**0**	**0**
	Owner User **(u)**			**Group (g)**			Other users **(o)**		

Example of relative change: (we add "+" execute rights to the owner)

```
chmod u+x essai.txt
```

```
pi@f4goh:~ $ ls -l essai.txt
-rw-rw---- 1 pi pi 10 mai   16 16:57 essai.txt
pi@f4goh:~ $ chmod u+x essai.txt
pi@f4goh:~ $ ls -l essai.txt
-rwxrw---- 1 pi pi 10 mai   16 16:57 essai.txt
pi@f4goh:~ $
```

The test file has become executable. When using the `ls` command, it is colored green (even if it can't actually run, it's still a text file, this is just taken as an example).

Note: It often happens that a file is downloaded and cannot be executed because it does not have the necessary rights. The `chmod` command solves the problem.

Delete the essai.txt file

```
rm essai.txt
```

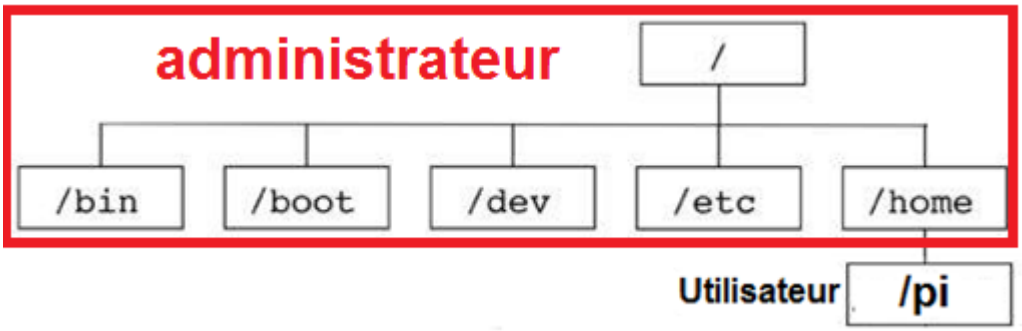
Exercise: From the table below, give the associated linux commands in order to modify the rights of the .txt file.

- (File)	read	write	not execute	read	write	not execute	Not read	Not write	not execute
	r	w	x	r	w	x	-	-	x
d (Folder)	**4**	**2**	**1**	**4**	**0**	**1**	**0**	**0**	**0**
	Owner User **(u)**			**Group (g)**			Other users **(o)**		

chmod 750 fichier.txt

5.5 The super user

On the Raspberry Pi, when the user uses the "pi" login, he only has access to the /home/pi directory and the subdirectories he created himself. The user pi does not have access to other directories (for example /bin, /boot, etc...).



To modify files for example in the /bin directory or to install new programs, the user "pi" will have to enter the sudo command systematically. sudo (abbreviation of substitute user do) is a command allowing a user to run a command as an administrator.

However, it is possible to switch to super user mode easily and there will be no need to type the sudo command systematically.

sudo su

`pi@f4goh:~ $ sudo su`
`root@f4goh:/home/pi# ls`
`rpi-clone`
`root@f4goh:/home/pi# exit`
`exit`
`pi@f4goh:~ $`
	In super user or root mode, it has no more colors and the command prompt ends with # instead of $.
	To exit super user mode, **exit** cmd.

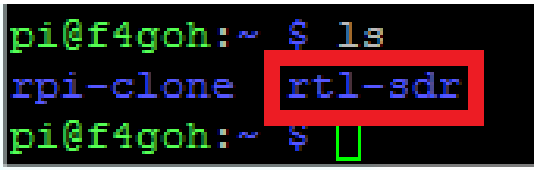
exit

Warning: In super user mode, you really need to know what you are doing. A file modification or deletion is irreversible.

5.6 RTL-SDR key Installation

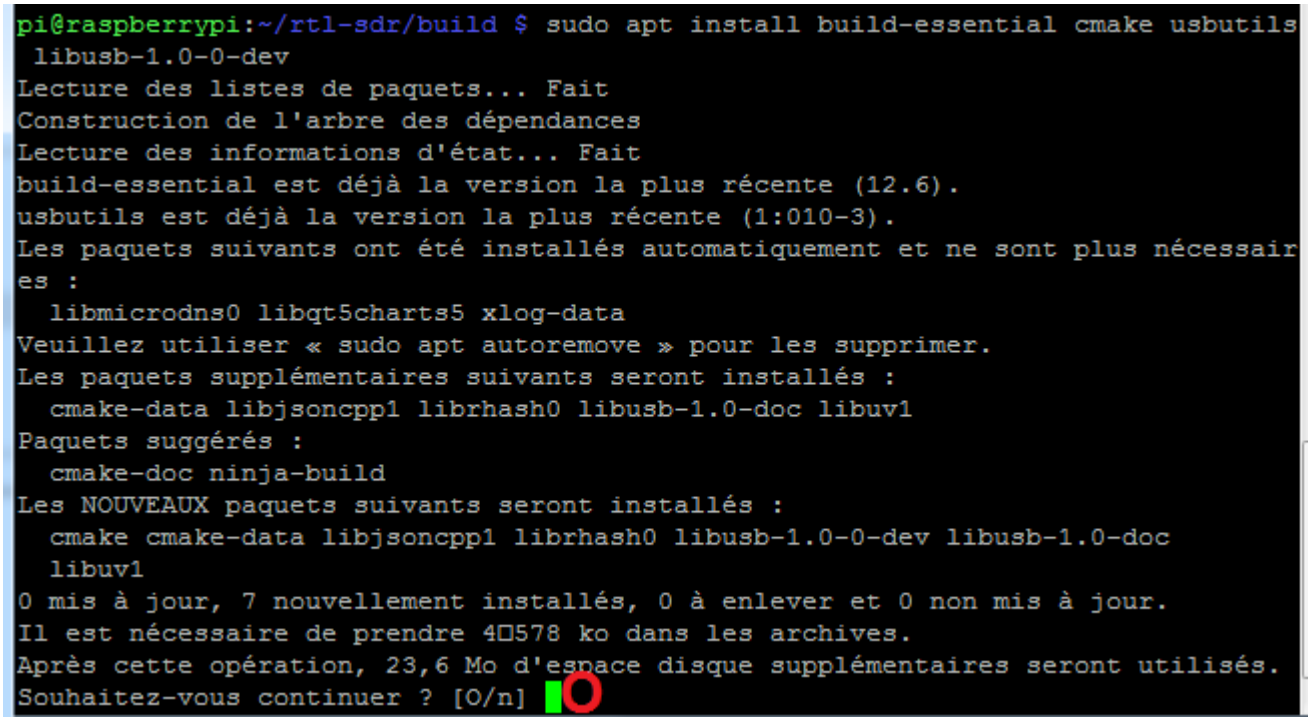
Retrieve the official files:

```
git clone https://github.com/osmocom/rtl-sdr.git
```



Install the libraries.

```
sudo apt install build-essential cmake usbutils libusb-1.0-0-dev
```



Type these commands in the order :

```
cd rtl-sdr

mkdir build

cd build

cmake -DINSTALL_UDEV_RULES=ON -DDETACH_KERNEL_DRIVER=ON ../
```

```
pi@raspberrypi:~ $ cd rtl-sdr/
pi@raspberrypi:~/rtl-sdr $ mkdir build
pi@raspberrypi:~/rtl-sdr $ cd build/
pi@raspberrypi:~/rtl-sdr/build $ cmake -DINSTALL_UDEV_RULES=ON -DDETACH_KERNEL_D
RIVER=ON ../
-- The C compiler identification is GNU 8.3.0
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compiler: /usr/bin/cc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Build type not specified: defaulting to release.
-- Extracting version information from git describe...
-- Looking for pthread.h
-- Looking for pthread.h - found
```

sudo make install

```
pi@raspberrypi:~/rtl-sdr/build $ sudo make install
Scanning dependencies of target convenience_static
[  3%] Building C object src/CMakeFiles/convenience_static.dir/convenience/conve
nience.c.o
[  6%] Linking C static library libconvenience_static.a
[  6%] Built target convenience_static
Scanning dependencies of target rtlsdr
[  9%] Building C object src/CMakeFiles/rtlsdr.dir/librtlsdr.c.o
[ 12%] Building C object src/CMakeFiles/rtlsdr.dir/tuner_e4k.c.o
[ 15%] Building C object src/CMakeFiles/rtlsdr.dir/tuner_fc0012.c.o
[ 18%] Building C object src/CMakeFiles/rtlsdr.dir/tuner_fc0013.c.o
[ 21%] Building C object src/CMakeFiles/rtlsdr.dir/tuner_fc2580.c.o
[ 25%] Building C object src/CMakeFiles/rtlsdr.dir/tuner_r82xx.c.o
[ 28%] Linking C shared library librtlsdr.so
```

sudo ldconfig

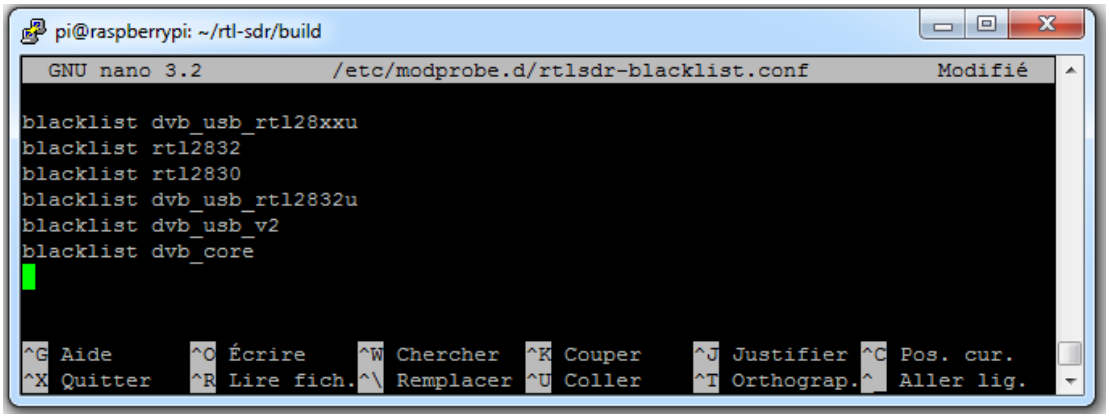
```
-- Set runtime path of "/usr/local/bin/rtl_fm" to ""
-- Installing: /usr/local/bin/rtl_eeprom
-- Set runtime path of "/usr/local/bin/rtl_eeprom" to ""
-- Installing: /usr/local/bin/rtl_adsb
-- Set runtime path of "/usr/local/bin/rtl_adsb" to ""
-- Installing: /usr/local/bin/rtl_power
-- Set runtime path of "/usr/local/bin/rtl_power" to ""
pi@raspberrypi:~/rtl-sdr/build $ sudo ldconfig
pi@raspberrypi:~/rtl-sdr/build $
```

sudo nano /etc/modprobe.d/rtlsdr-blacklist.conf

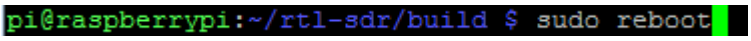
```
pi@raspberrypi:~/rtl-sdr/build $ sudo nano /etc/modprobe.d/rtlsdr-blacklist.conf
```

```
blacklist dvb_usb_rtl28xxu
blacklist rtl2832
blacklist rtl2830
blacklist dvb_usb_rtl2832u
blacklist dvb_usb_v2
blacklist dvb_core
```

Copy and paste the above list in the nano editor. Reminder right click: to paste the text in the nano editor.

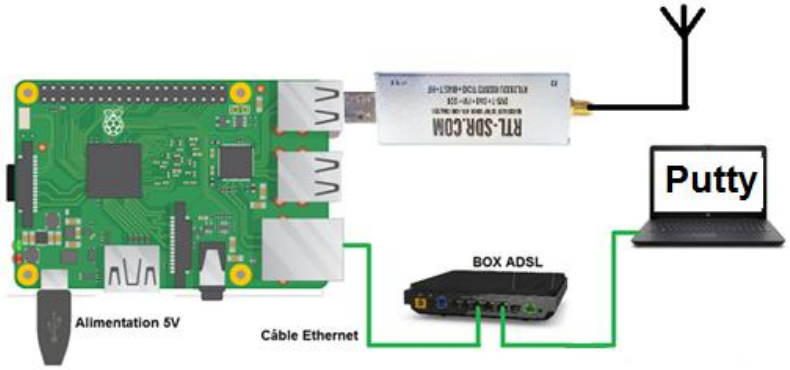


Save the file using the Ctrl + o keys, then the enter key to validate the recording.	**Exit the editor using Ctrl + x.**
	

Restart the Raspberry PI: 


```
sudo reboot
```

Verification: Plug the rtl-sdr key into a USB port on the Raspberry Pi. Connect the rtl-sdr key to the Raspberry Pi on a free USB port.



Check the presence of the rtl-sdr key:

With LXTerminal or Putty, type the following command line :

```
lsusb
```

Locate the rtl-sdr key: **RTL238 DVB-T**

Raspberry Pi 3 :

```
pi@raspberrypi:~ $ lsusb
Bus 001 Device 004: ID 0bda:2838 Realtek Semiconductor Corp. RTL2838 DVB-T
Bus 001 Device 003: ID 0424:ec00 Standard Microsystems Corp. SMSC9512/9514 Fast
Ethernet Adapter
Bus 001 Device 002: ID 0424:9514 Standard Microsystems Corp. SMC9514 Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

Raspberry Pi 4 :

```
pi@raspberrypi:~ $ lsusb
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 003: ID 0bda:2838 Realtek Semiconductor Corp. RTL2838 DVB-T
Bus 001 Device 002: ID 2109:3431 VIA Labs, Inc. Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

With LXTerminal or Putty, type the following command line : (be careful, underscore key)

```
rtl_test
```

The rtl-sdr key should be recognized, if not, disconnect and reconnect the key and restart the test.

```
pi@raspberrypi:~ $ rtl_test
Found 1 device(s):
  0:  Realtek, RTL2838UHIDIR, SN: 00000001

Using device 0: Generic RTL2832U OEM
Found Rafael Micro R820T tuner
Supported gain values (29): 0.0 0.9 1.4 2.7 3.7 7.7 8.7 12.5 14.4 15.7 16.6 19.7
 20.7 22.9 25.4 28.0 29.7 32.8 33.8 36.4 37.2 38.6 40.2 42.1 43.4 43.9 44.5 48.0
 49.6
[R82XX] PLL not locked!
Sampling at 2048000 S/s.

Info: This tool will continuously read from the device, and report if
samples get lost. If you observe no further output, everything is fine.

Reading samples in async mode...
^CSignal caught, exiting!
ctrl+c

User cancel, exiting...
Samples per million lost (minimum): 0
pi@raspberrypi:~ $ ^C
```

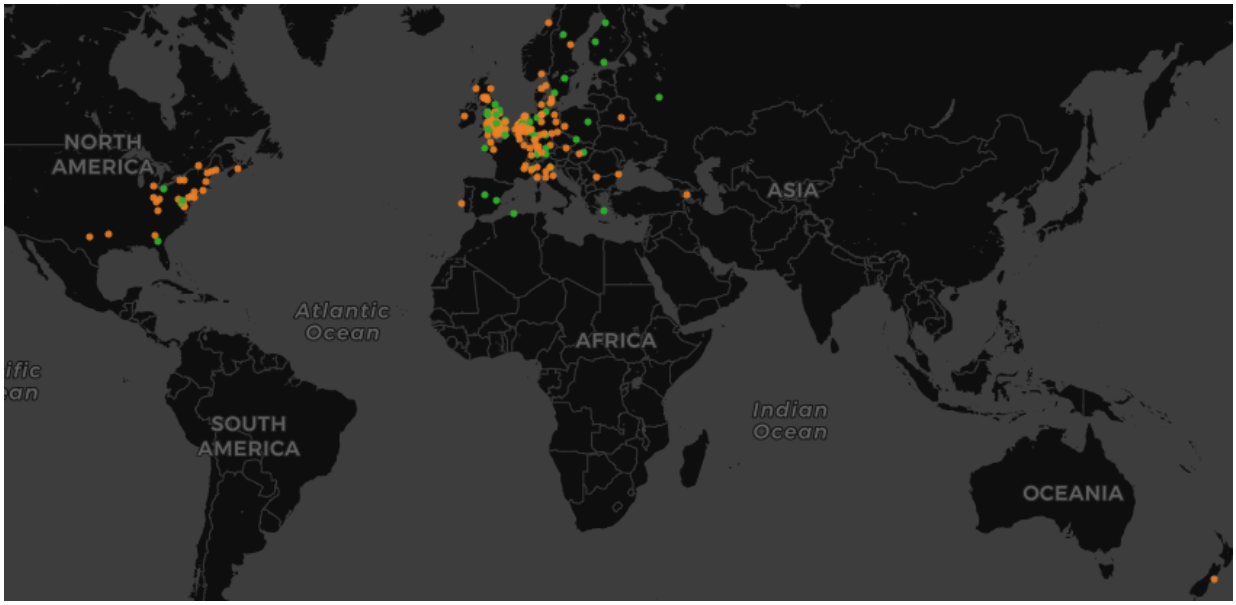
Exit the program by pressing the Ctrl key and the c key simultaneously (Ctrl+c)

If **`rtl_test`** displays this message `lost at least 112 bytes` repeatedly, reinstall the program using the manual method.

5.7 WSPR decoding

The objective is to decode the frames received in WSPR over 40 meters and to transfer them to the WSPR server, without using the WSJT-X software.

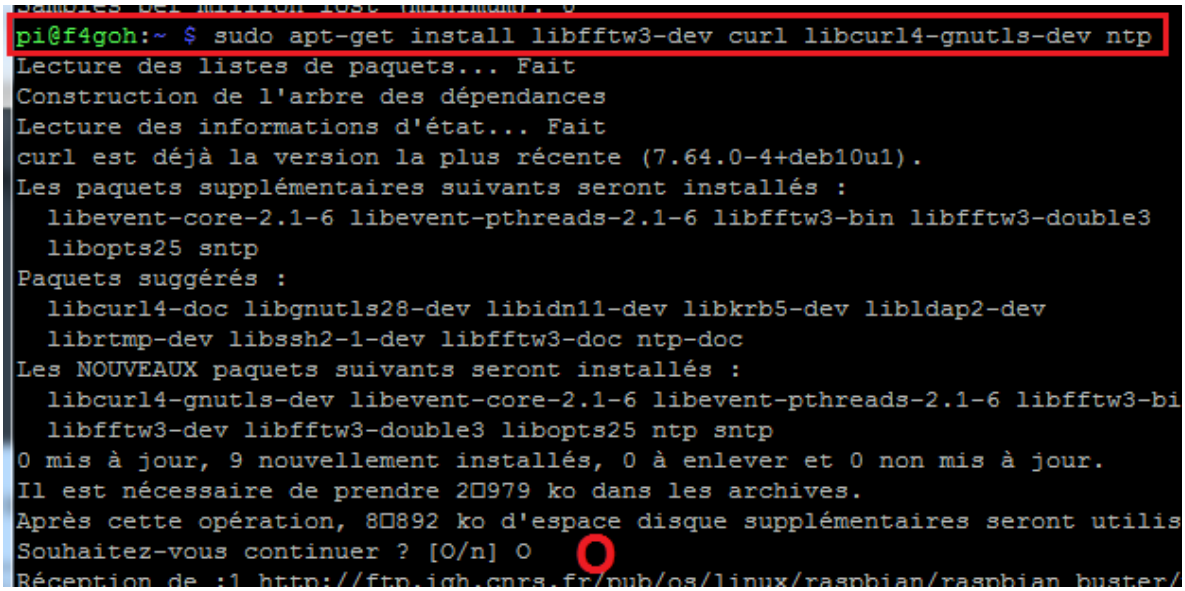
Below is an example of reception over a 24-hour period.



Prerequisite: To have installed the drivers of the rtl-sdr key previously.

```
sudo apt-get install libfftw3-dev curl libcurl4-gnutls-dev ntp
```

It's better to copy and paste than to retype everything. (See part 2, page 4)



Download with git the program of Guenael VA2GKA.

```
git clone https://github.com/Guenael/rtlsdr-wsprd
```

```
pi@f4goh:~ $ git clone https://github.com/Guenael/rtlsdr-wsprd
Clonage dans 'rtlsdr-wsprd'...
remote: Enumerating objects: 123, done.
remote: Total 123 (delta 0), reused 0 (delta 0), pack-reused 123
Réception d'objets: 100% (123/123), 217.19 KiB | 774.00 KiB/s, fait.
Résolution des deltas: 100% (73/73), fait.
pi@f4goh:~ $
```

```
cd rtlsdr-wsprd
```

What are the files in the rtlsdr-wsprd directory?

```
ls
```

There is no executable file otherwise it would be green.

```
pi@f4goh:~ $ cd rtlsdr-wsprd/
pi@f4goh:~/rtlsdr-wsprd $ ls
fano.c           nhash.c         rtlsdr_wsprd.h  wsprd_utils.c
fano.h           nhash.h         tab.c           wsprd_utils.h
Makefile         README.md       wsprd.c         wsprsim_utils.c
metric_tables.h  rtlsdr_wsprd.c  wsprd.h         wsprsim_utils.h
pi@f4goh:~/rtlsdr-wsprd $
```

Compilation of the program.

```
make
```

```
pi@f4goh:~/rtlsdr-wsprd $ make
gcc -Wall -O3 -ffast-math -std=gnu99 -c rtlsdr_wsprd.c -o rtlsdr_wsprd.o
rtlsdr_wsprd.c: In function 'main':
rtlsdr_wsprd.c:750:34: warning: '%02d' directive writing between 2 and 11 bytes int
o a region of size 7 [-Wformat-overflow=]
         sprintf(rx_options.date,"%02d%02d%02d", gtm->tm_year - 100, gtm->tm_mon +
1, gtm->tm_mday);
```

Is an executable file generated?

```
ls
```

Yes, the file is in green (rtlsdr_wsprd)

```
pi@f4goh:~/rtlsdr-wsprd $ ls
fano.c           nhash.c       rtlsdr_wsprd.c  wsprd.c        wsprd_utils.o
fano.h           nhash.h       rtlsdr_wsprd.h  wsprd.h        wsprsim_utils.c
fano.o           nhash.o       rtlsdr_wsprd.o  wsprd.o        wsprsim_utils.h
Makefile         README.md     tab.c           wsprd_utils.c  wsprsim_utils.o
metric_tables.h  rtlsdr_wsprd  tab.o           wsprd_utils.h
pi@f4goh:~/rtlsdr-wsprd $
```

Running the file: point, slash, rtlsdr_wsprd

```
./rtlsdr_wsprd
```

```
pi@f4goh:~/rtlsdr-wsprd $ ./rtlsdr_wsprd
rtlsdr_wsprd, a simple WSPR daemon for RTL receivers

Use:    rtlsdr_wsprd -f frequency -c callsign -l locator [options]
        -f dial frequency [(,k,M) Hz], check http://wsprnet.org/ for freq.
        -c your callsign (12 chars max)
        -l your locator grid (6 chars max)
Receiver extra options:
        -g gain [0-49] (default: 29)
        -a auto gain (default: off)
        -o frequency offset (default: 0)
        -p crystal correction factor (ppm) (default: 0)
        -u upconverter (default: 0, example: 125M)
        -d direct dampling [0,1,2] (default: 0, 1 for I input, 2 for Q input)
        -n max iterations (default: 0 = infinite loop)
        -i device index (in case of multiple receivers, default: 0)
Decoder extra options:
        -H use the hash table (could caught signal 11 on RPi)
        -Q quick mode, doesn't dig deep for weak signals
        -S single pass mode, no subtraction (same as original wsprd)
Example:
        rtlsdr_wsprd -f 144.489M -c A1XYZ -l AB12cd -g 29 -o -4200
pi@f4goh:~/rtlsdr-wsprd $
```

Usual frequencies :

Frequencies

USB dial (MHz): 0.136, 0.4742, 1.8366, 3.5686, 5.2872, 5364.7, 7.0386, 10.1387, 14.0956, 18.1046, 21.0946, 24.9246, 28.1246, 50.293, 70.091, 144.489, 432.300, 1296.500

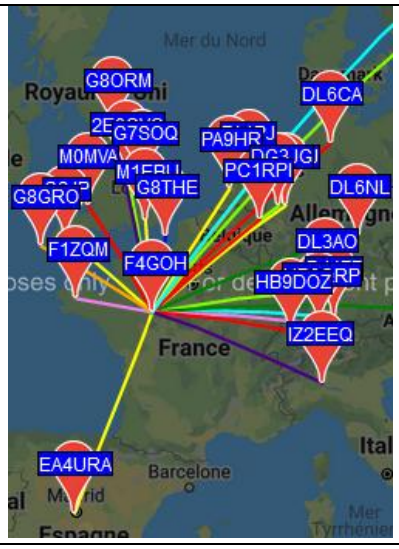
You have to pass parameters to the program in order to run it with a callsign

Example for the 40 meters band, frequency 7.0386MHz, callsign f4goh, locator jn07dv, gain 29 db, frequency offset 10 hertz, direct sampling for HF (-d 2) Q input.

```
sudo ./rtlsdr_wsprd -f 7.0386M -c F4GOH -l JN07DV -g 29 -o -10 -d 2
```

Résultat :

```
pi@f4goh: ~/rtlsdr-wsprd
pi@f4goh:~/rtlsdr-wsprd $ sudo ./rtlsdr_wsprd -f 7.0386M -c F4GOH -l JN07DV -g 2
9 -o -10 -d 2
Found 1 device(s):
  0:  Realtek, RTL2838UHIDIR, SN: 00000001

Using device 0: Generic RTL2832U OEM
Found Rafael Micro R820T tuner
Enabled direct sampling mode, input 2

Starting rtlsdr-wsprd (2020-05-22, 18:53z) -- Version 0.2
  Callsign      : F4GOH
  Locator       : JN07DV
  Dial freq.    : 7038599 Hz
  Real freq.    : 7038589 Hz
  PPM factor    : 0
  Gain          : 29 dB
Wait for time sync (start in 43 sec)

Spot : -11.27 -0.42   7.040109  0  <...> JN47VQ 37
Spot : -12.00 -2.09   7.040065  0  <...> JO31OL 23
Spot : -12.58 0.60    7.040140  1  G8THE JO00 20
Spot : -20.83 -0.31   7.040119  0  <...> JO50IP 23
Spot : -21.66 1.67    7.040058  0  G7SOQ IO92 23
Spot : 0.47 -1.49     7.040102  0  G2JP IO70 37
```



5.8 Weather stations decoding

Prerequisite: To have installed the drivers of the rtl-sdr key previously.

Download with git the rtl_433 program.

```
git clone https://github.com/merbanan/rtl_433.git
```

```
pi@f4goh:~ $ git clone https://github.com/merbanan/rtl_433.git
Clonage dans 'rtl_433'...
remote: Enumerating objects: 14, done.
remote: Counting objects: 100% (14/14), done.
remote: Compressing objects: 100% (14/14), done.
remote: Total 14841 (delta 3), reused 1 (delta 0), pack-reused 14827
Réception d'objets: 100% (14841/14841), 5.08 MiB | 1.47 MiB/s, fait.
Résolution des deltas: 100% (11480/11480), fait.
pi@f4goh:~ $ ls
rpi-clone  rtl_433  rtl-sdr  rtlsdr-wsprd
```

```
cd rtl_433/
mkdir build
cd build
cmake ../
```

```
pi@f4goh:~ $ ls
rpi-clone  rtl_433  rtl-sdr  rtlsdr-wsprd
pi@f4goh:~ $ cd rtl_433/
pi@f4goh:~/rtl_433 $ mkdir build
pi@f4goh:~/rtl_433 $ cd build/
pi@f4goh:~/rtl_433/build $ cmake ../
-- The C compiler identification is GNU 8.3.0
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compiler: /usr/bin/cc -- works
```

```
make
```

```
pi@f4goh:~/rtl_433/build $ make
Scanning dependencies of target r_433
[  1%] Building C object src/CMakeFiles/r_433.dir/abuf.c.o
[  1%] Building C object src/CMakeFiles/r_433.dir/am_analyze.c.o
[  2%] Building C object src/CMakeFiles/r_433.dir/baseband.c.o
[  2%] Building C object src/CMakeFiles/r_433.dir/bitbuffer.c.o
[  3%] Building C object src/CMakeFiles/r_433.dir/compat_paths.c.o
```

sudo make install

```
pi@f4goh:~/rtl_433/build $ sudo make install
[ 88%] Built target r_433
[ 89%] Built target rtl_433
[ 92%] Built target data
[ 93%] Built target style-check
[ 94%] Built target baseband-test
[ 95%] Built target test_bitbuffer
[ 96%] Built target data-test
[ 97%] Built target test_fileformat
[ 98%] Built target test_optparse
[100%] Built target test_util
Install the project...
```

Connect a VHF/UHF antenna to the rtl-sdr key and run the program. (beware underscore key)

rtl_433

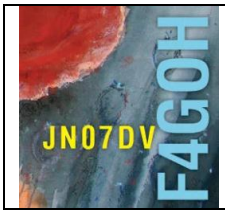
Here are some weather stations in my neighborhood.

```
pi@f4goh:~/rtl_433/build $ rtl_433
rtl_433 version 20.02-56-gd4ce64b branch master at 202005201829 inputs file rtl_tcp RTL-SDR
Use -h for usage help and see https://triq.org/ for documentation.
Trying conf file at "rtl_433.conf"...
Trying conf file at "/home/pi/.config/rtl_433/rtl_433.conf"...
Trying conf file at "/usr/local/etc/rtl_433/rtl_433.conf"...
Trying conf file at "/etc/rtl_433/rtl_433.conf"...
Registered 124 out of 152 device decoding protocols [ 1-4 8 11-12 15-17 19-21 23 25-26 29-36 38
-60 63 67-71 73-100 102-105 108-116 119 121 124-128 130-149 151-152 ]
Found Rafael Micro R820T tuner
Exact sample rate is: 250000.000414 Hz
[R82XX] PLL not locked!
Sample rate set to 250000 S/s.
Tuner gain set to Auto.
Tuned to 433.920MHz.

time      : 2020-05-22 21:48:13
model     : Nexus-T      House Code: 162
Channel   : 1            Battery   : 0            Temperature: 21.10 C

time      : 2020-05-22 21:48:17                   brand     : OS
model     : Oregon-THGR122N                       House Code: 26
Channel   : 1            Battery   : 1            Temperature: 22.00 C
Humidity  : 35 %

time      : 2020-05-22 21:48:17                   brand     : OS
model     : Oregon-THGR122N                       House Code: 26
Channel   : 1            Battery   : 1            Temperature: 22.00 C
Humidity  : 35 %
^CSignal caught, exiting!
pi@f4goh:~/rtl_433/build $
```

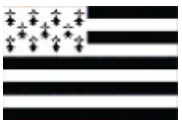
ctrl+c pour sortir

Next part: Installation and configuration of a sdr listening server (openwebrx)

Next: https://github.com/projecthorus/radiosonde_auto_rx/wiki



Raspberry Pi for HAM Radio
Part 6



Prerequisite: Discovery of Raspberry PI Part 4

Summary:

Part 6: OpenwebRX.

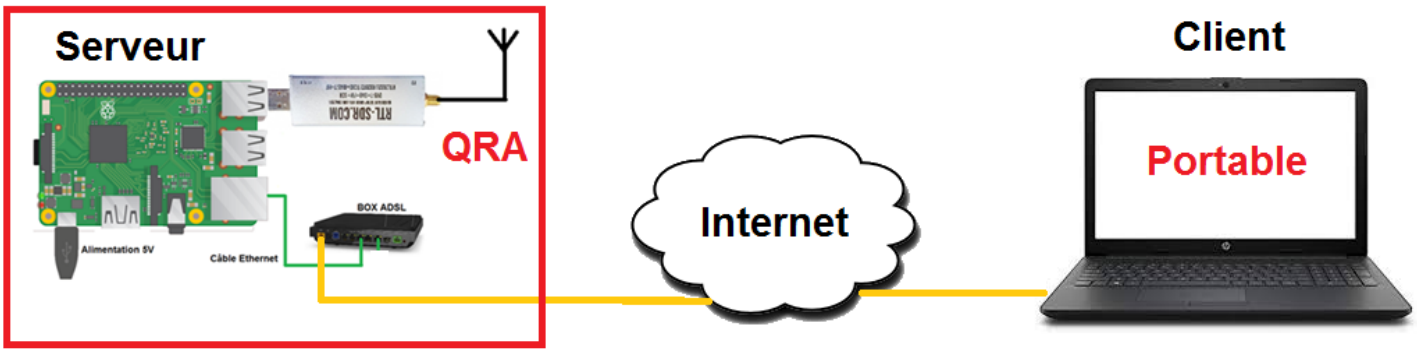
This tutorial only comments on the openWebRX instructions.

Version du 22/05/2021 V3

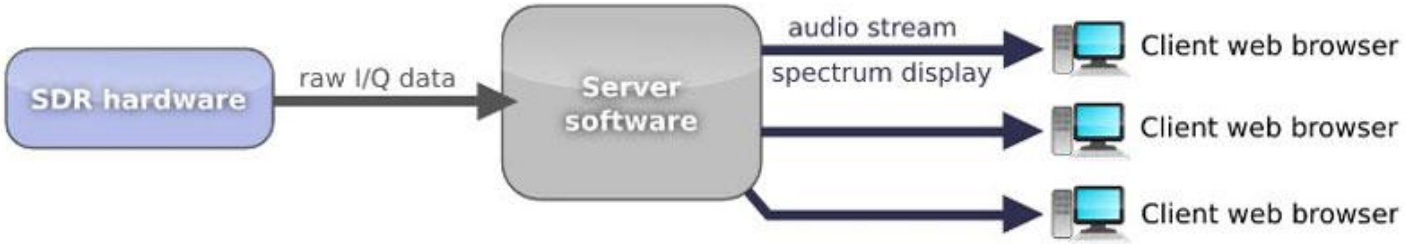
6.1 Introduction.

The objective of this tutorial is to install in his radio shack, the openwebRX software in order to create his own SDR web.

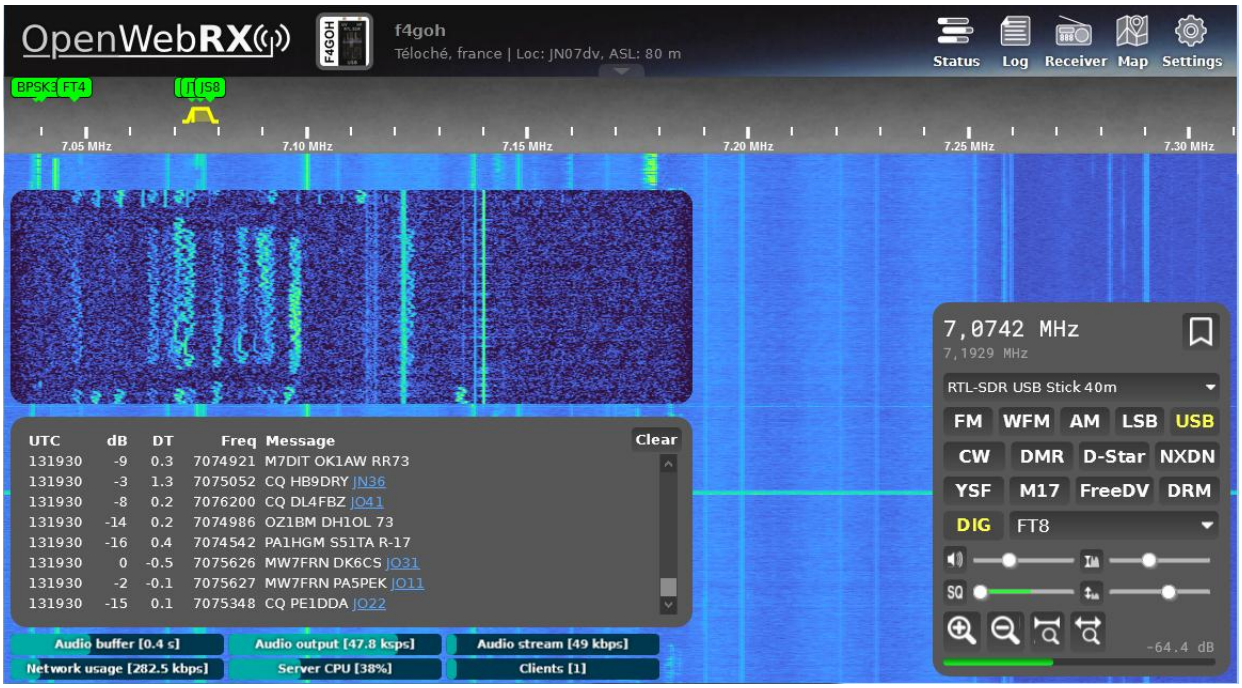
When you are on the move, you just have to connect to your laptop at home to listen to the desired frequency band using your own antennas.



OpenwebRX is a solution for remote spectrum monitoring and has AM / FM / SSB / CW / BPSK31 demodulators. It uses a lightweight signal processing (DSP) library called libcsdr. The spectrum display is like any waterfall radio software; however, the display can also be viewed in 3D.

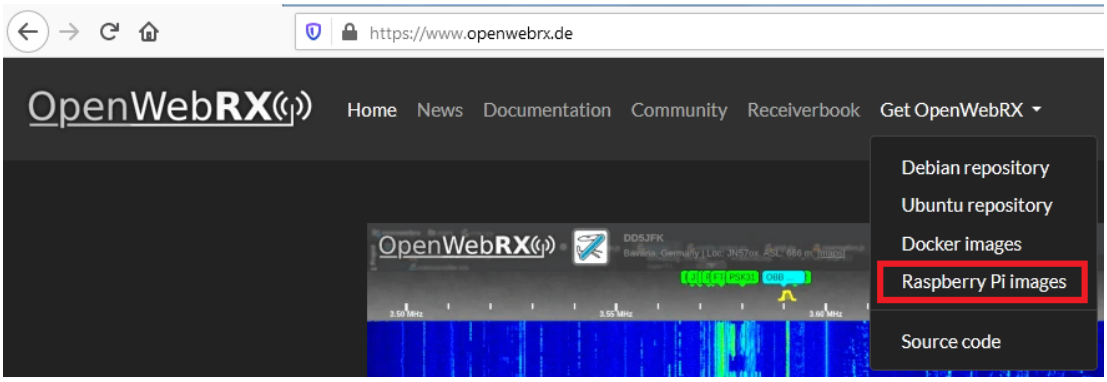


Since the upload speed of a private Internet connection is relatively low, care should be taken to ensure that there are not too many people connected to the same server.

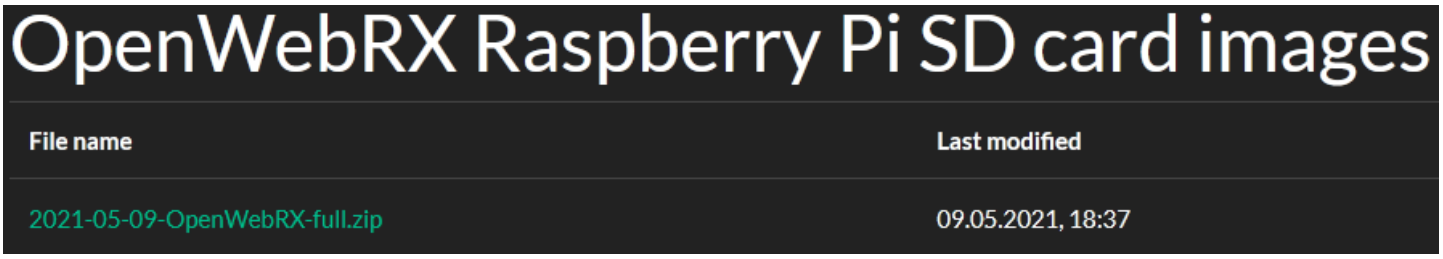


6.2 Installation of the ready-to-use image

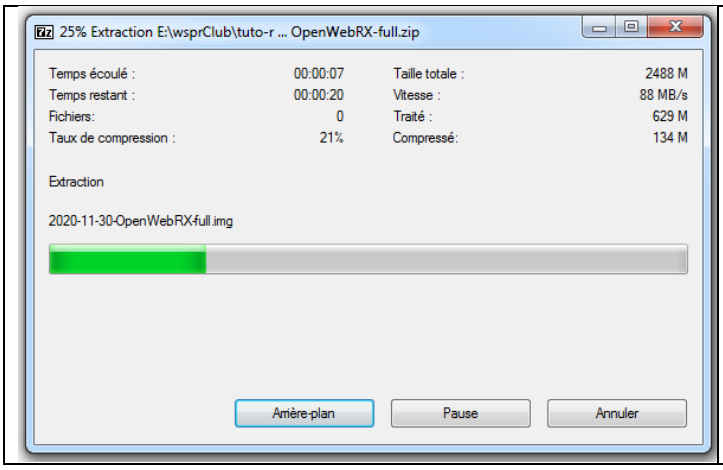
The project initially created by Andras, HA7ILM was taken over by Jakob DD5JFK. Jakob had the good idea to propose a ready to use Raspberry Pi image.

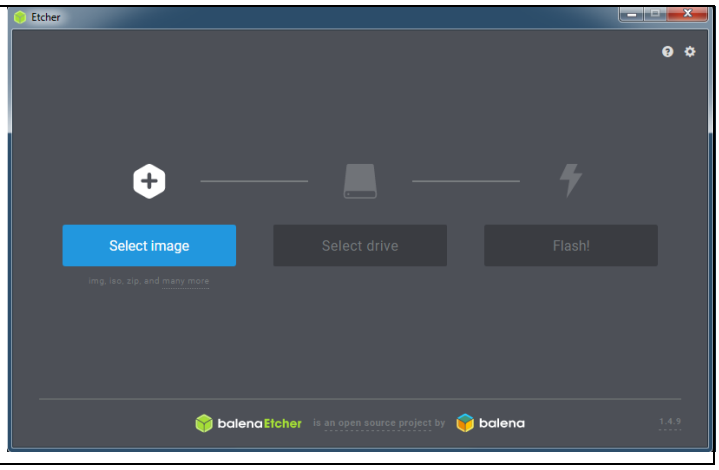


Start by downloading the zip file `2020-11-30-OpenWebRX-full.zip` or later.



Unzip the file, then use https://www.balena.io/etcher to copy the image to an mSD card.

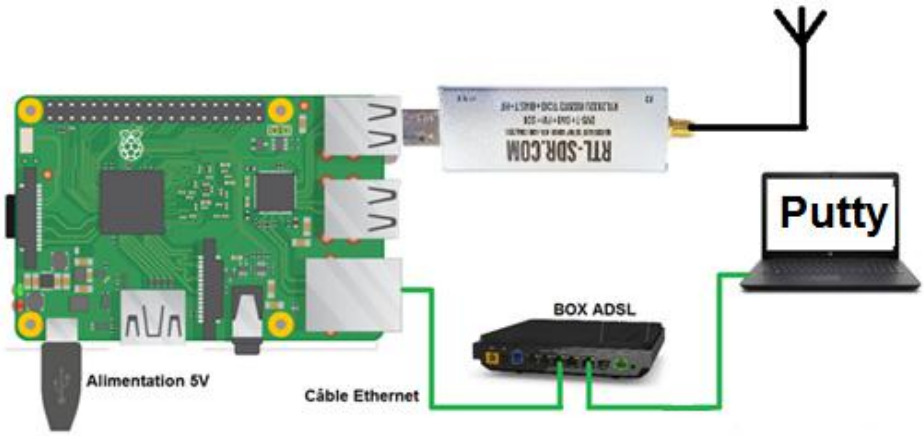




Use **tutorial 4** to configure the keyboard in **your language** and enable the **ssh connection** more convenient to configure openwebrx with the **Putty** utility.

Attention the keyboard configuration is initially in qwerty. Then run the `raspi-config` and follow the tutorial 4.

Once the board is inserted into the Raspberry PI and powered up **with the rtl-sdr dongle plugged in**, perform a connectivity test between a fixed PC and the RPI using the Windows console.

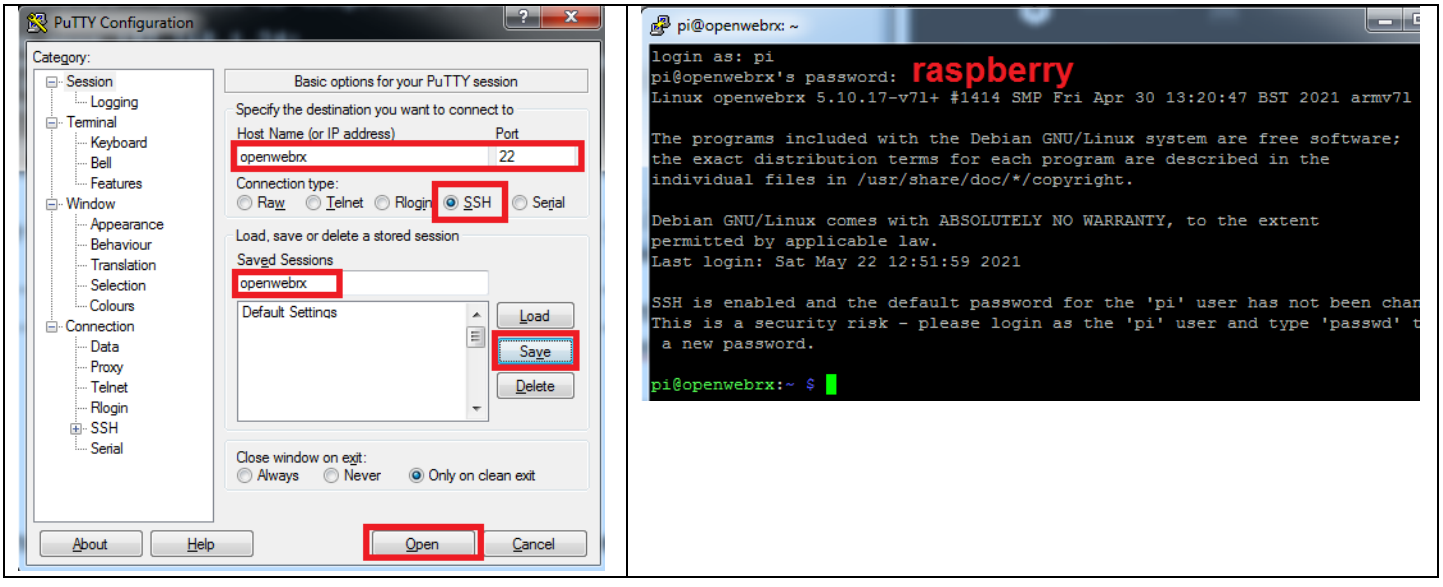



```
ping openwebrx
```

Invite de commandes

```
C:\Users\anthony>ping openwebrx

Envoi d'une requête 'ping' sur openwebrx.home [192.168.1.24]
onnées :
Réponse de 192.168.1.24 : octets=32 temps=1 ms TTL=64
Réponse de 192.168.1.24 : octets=32 temps<1ms TTL=64
Réponse de 192.168.1.24 : octets=32 temps<1ms TTL=64
Réponse de 192.168.1.24 : octets=32 temps<1ms TTL=64

Statistiques Ping pour 192.168.1.24:
    Paquets : envoyés = 4, reçus = 4, perdus = 0 (perte 0%),
Durée approximative des boucles en millisecondes :
    Minimum = 0ms, Maximum = 1ms, Moyenne = 0ms

C:\Users\anthony>_
```

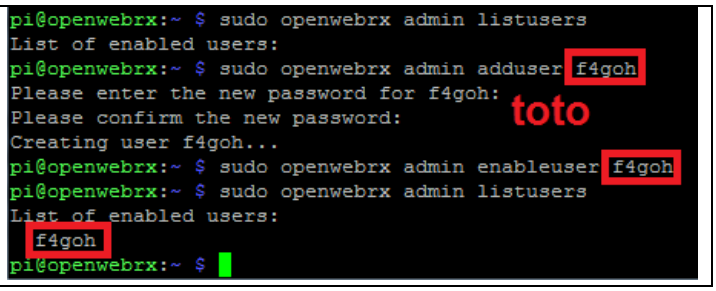
With Putty, establish an **SSh** connection with the Raspberry. Please note that the **SSh** connection must be enabled as explained in **tutorial 4**.



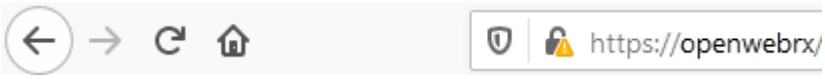
6.3 Creating a user and minimum requirements

With Putty, create a user as explained in the guide here
https://github.com/jketterl/openwebrx/wiki/User-Management

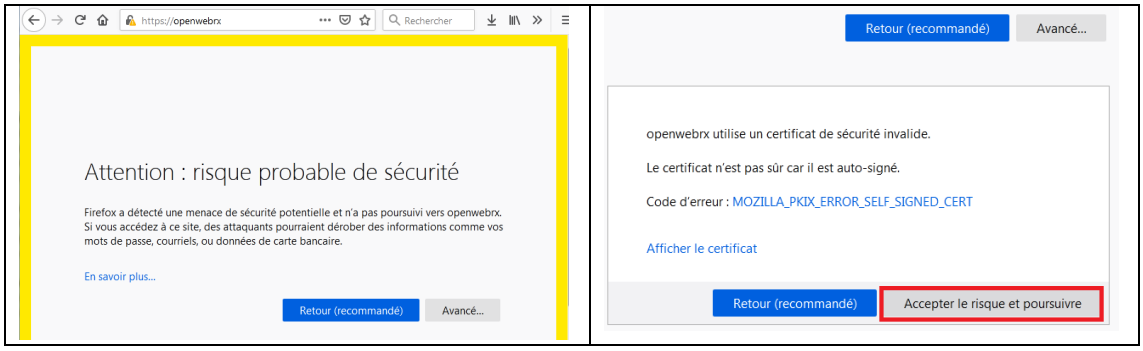
A minimum of 2 commands must be entered to create a user. Replace the username with your callsign. (I used a simple password for the tests: toto)

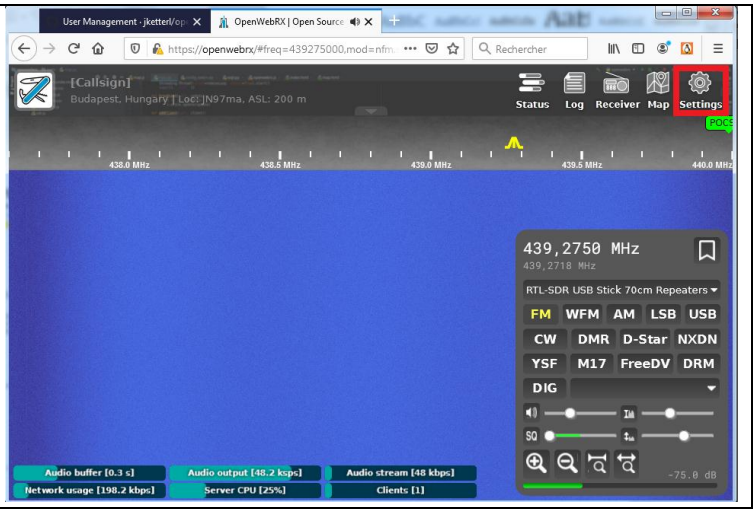
sudo openwebrx admin adduser **f4goh** sudo openwebrx admin enableuser **f4goh** sudo openwebrx admin listusers	

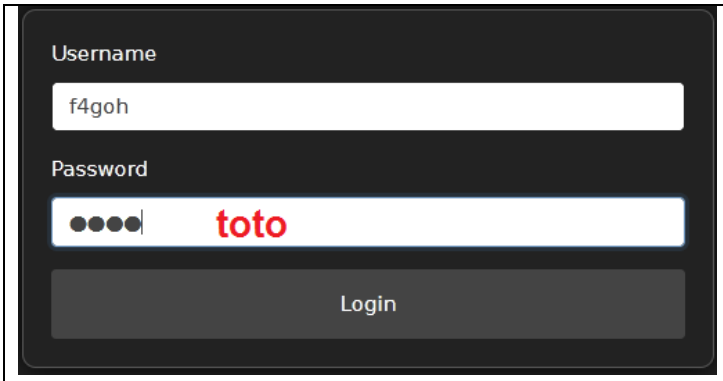
In the URL bar of firefox enter the following address https://openwebrx/ or openwebrx :8073

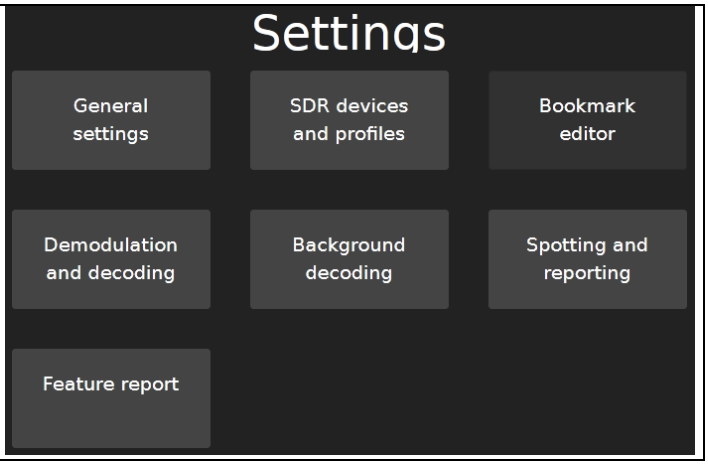


Add security exception



A web page should appear with a default frequency of around 430Mhz. A background sound (noise) should be heard. Go to the setting menu at the top right of the screen.	





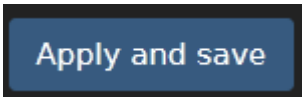
Configuration minimale de la station réceptrice

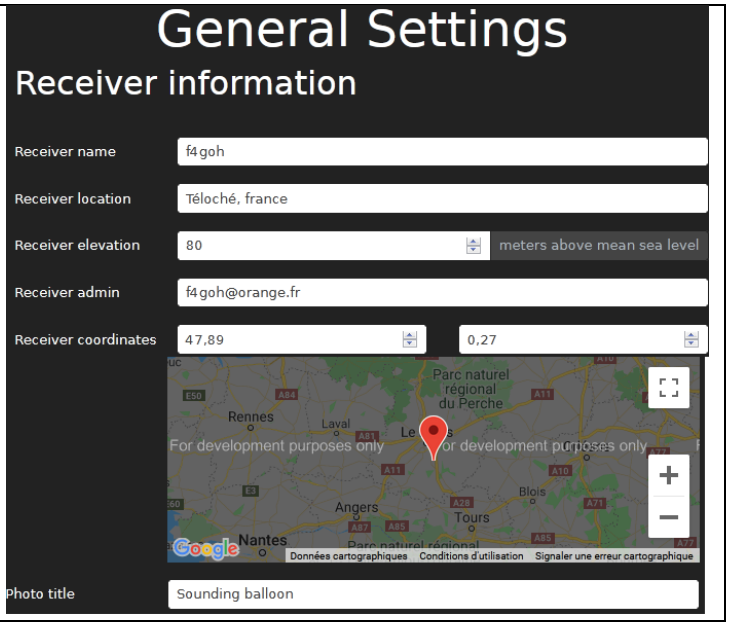
Fill in the data fields

- Code ;
- Location ;
- Altitude ;
- Geographical position ;

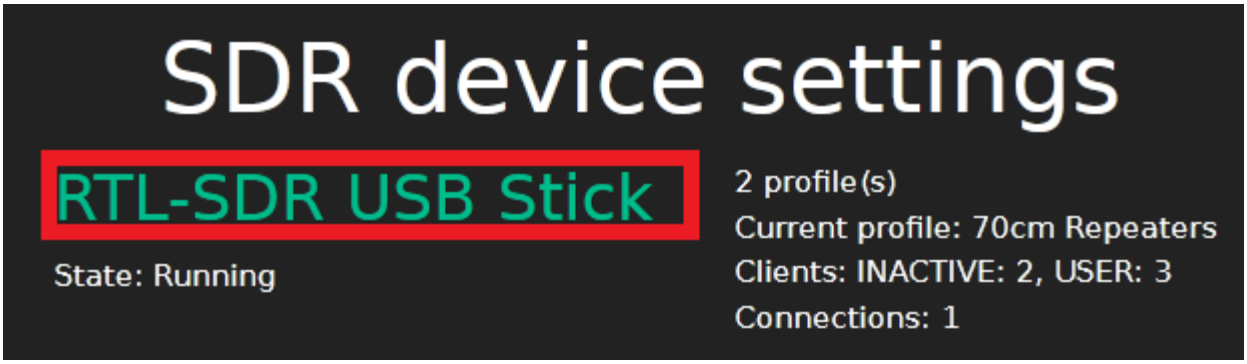
It is possible to change the background image and the icon.

Finish by applying and saving the changes.



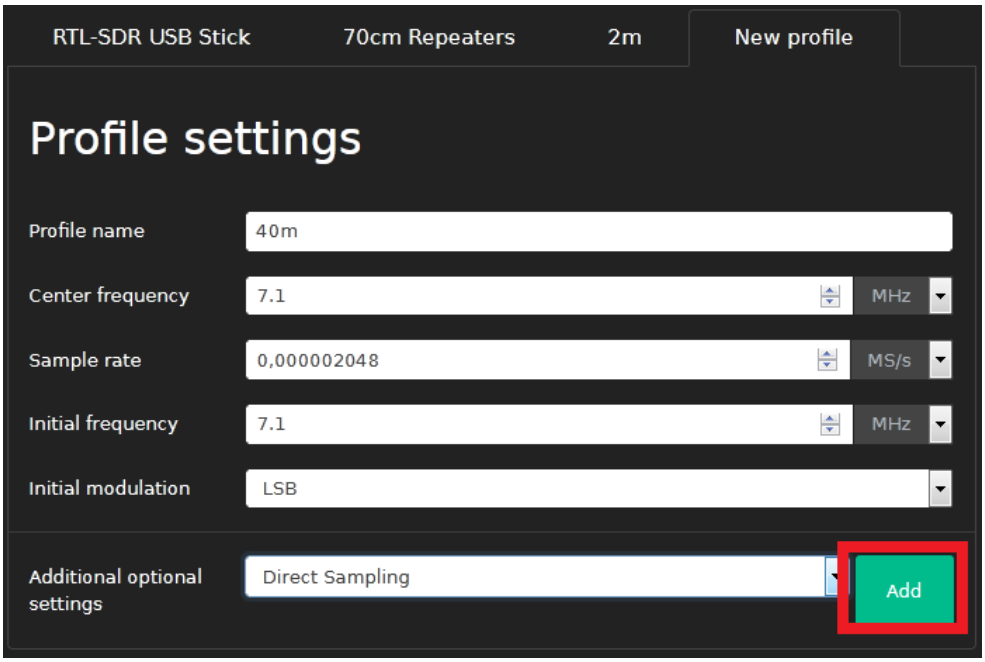


Receiver band additions (SDR devices and profiles menu)

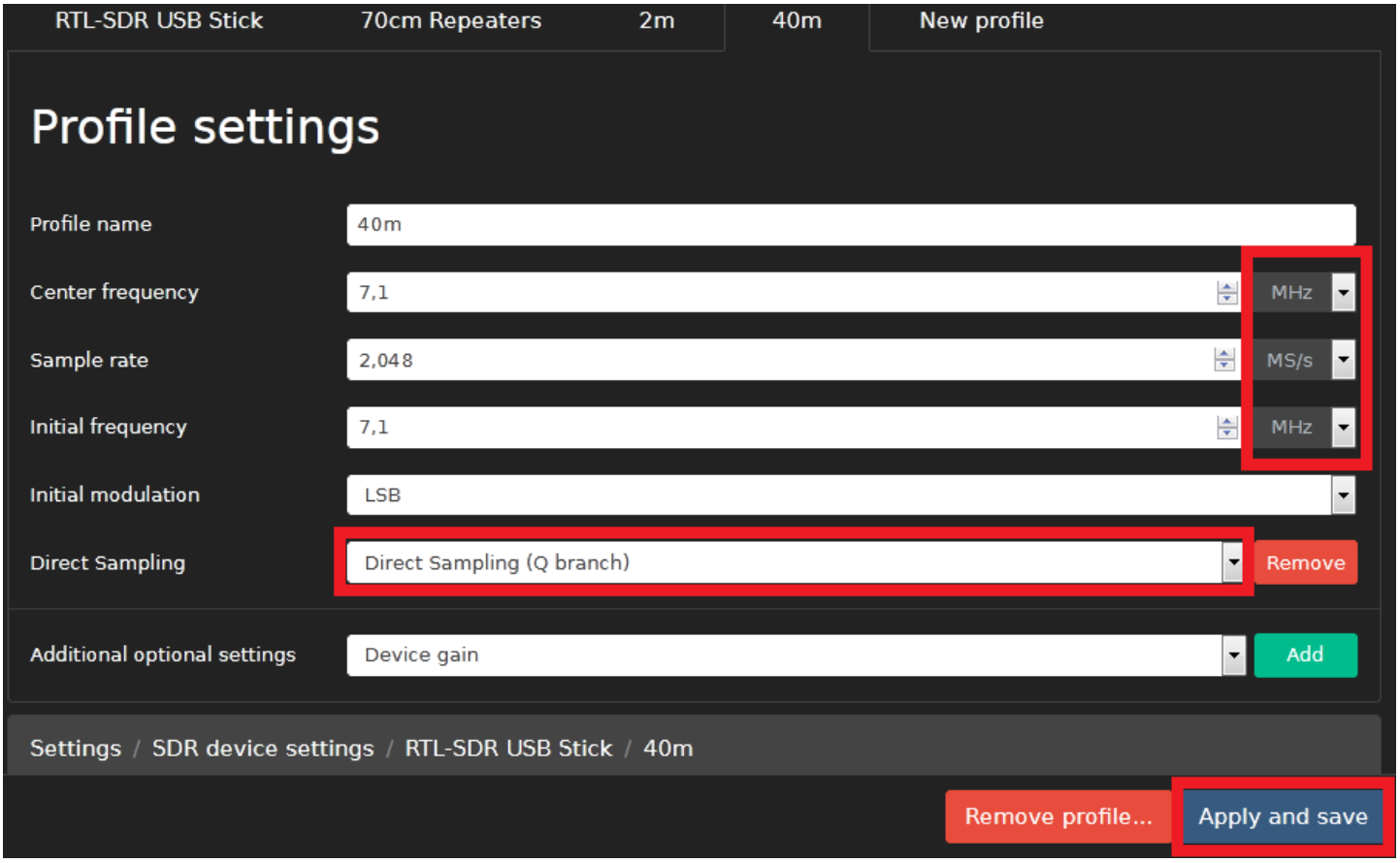


The rtl-sdr key is well detected. There are two default profiles, one in UHF and one in VHF. Click on the link RTL-SDR USB Stick

Addition of a new HF reception profile on 40m.

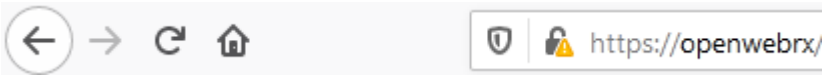


Add Direct Sampling mode

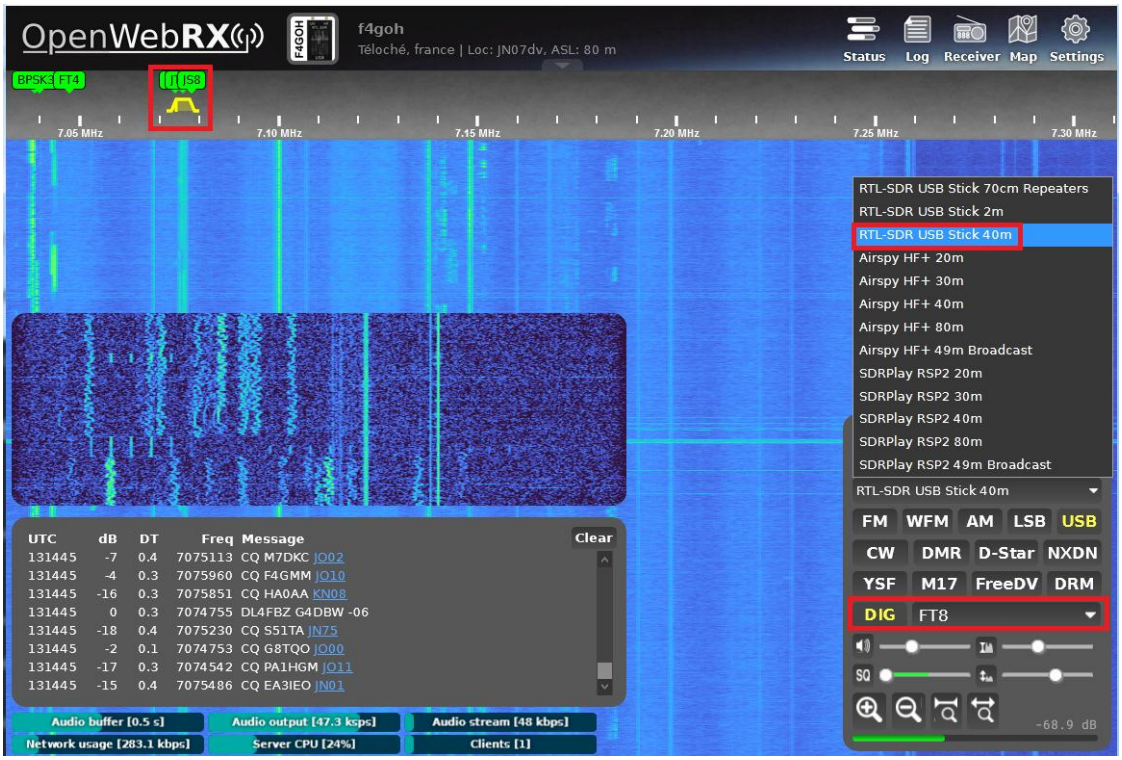


Select the Q branch option for HF, then save the configuration.

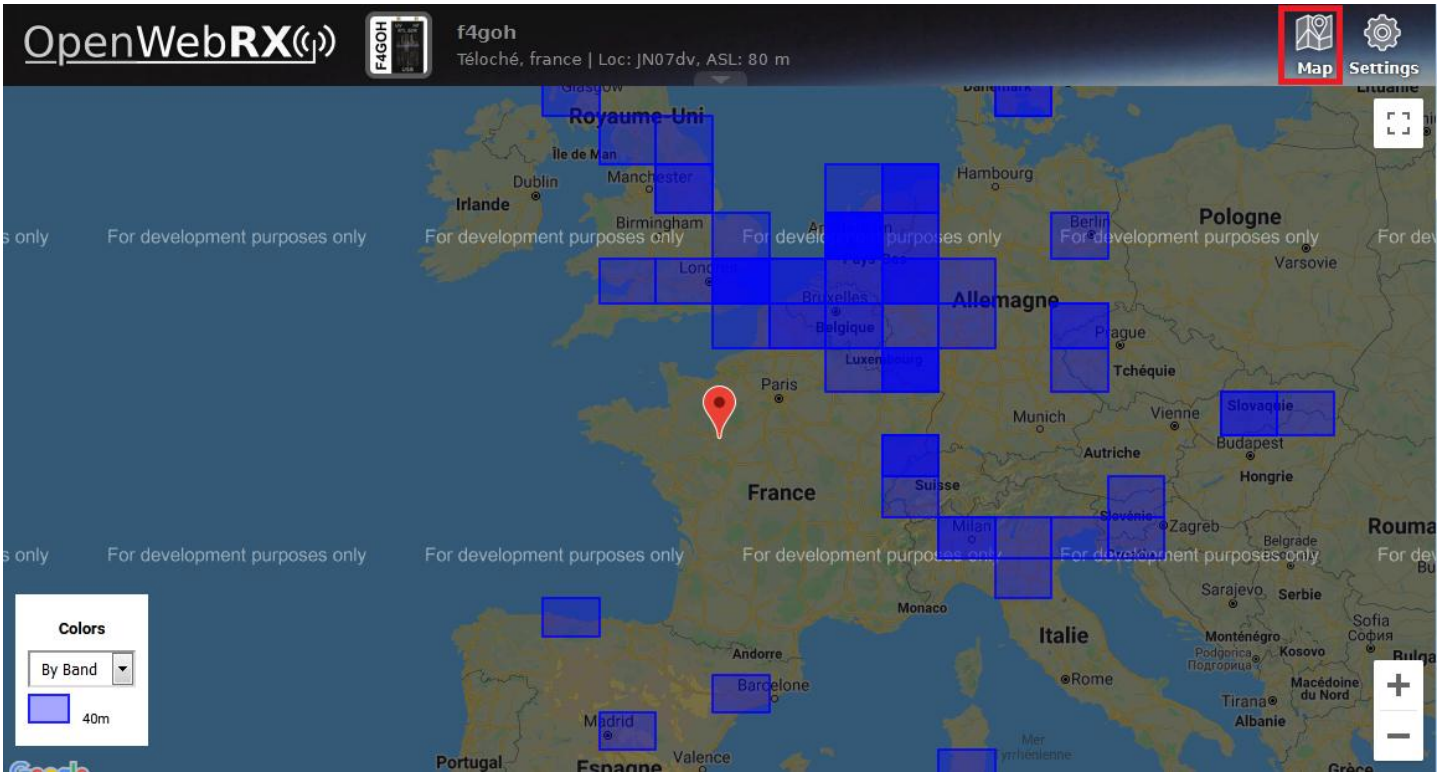
In the URL bar of firefox enter the following address https://openwebrx/ or openwebrx :8073



Select the previously created profile 40 from the strip menu.

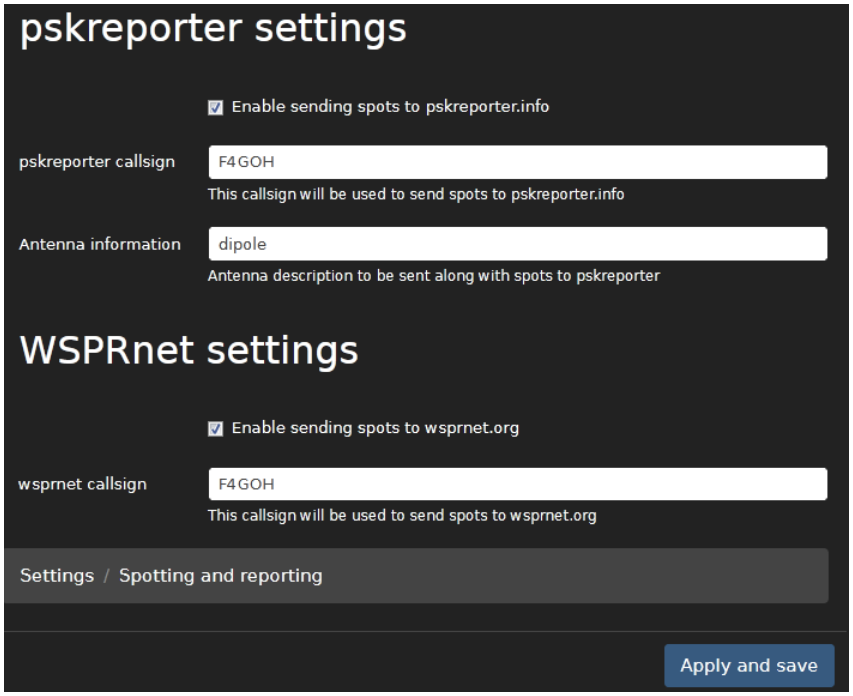


It is possible to decode in real time different digital modes such as FT8 and to have the mapping of the locators.

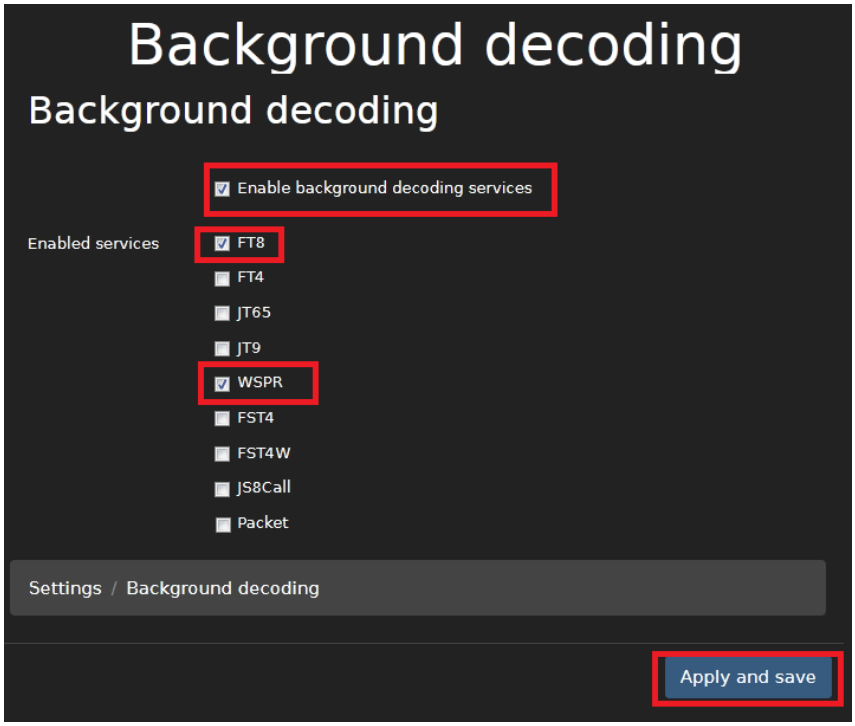


6.4 Spots Reporting on pskreporter or in WSPR

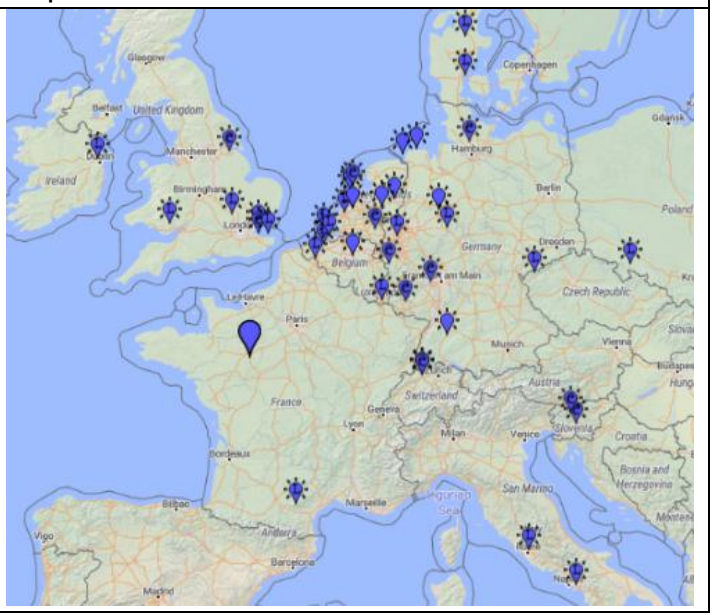
In the **Spotting and reporting menu**, it is possible to activate the reporting of stations in APRS, PSK and WSPR.



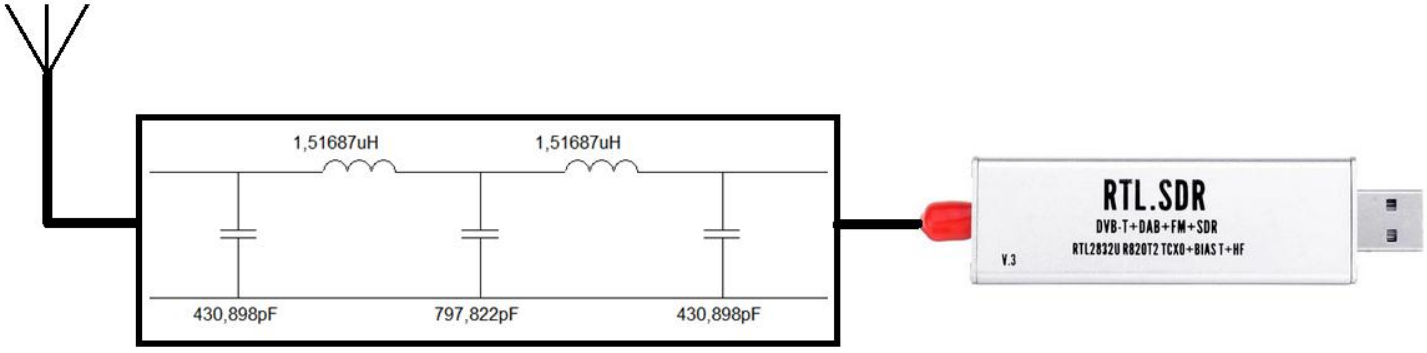
Don't forget to activate background decoding (Background decoding menu)



15 minutes later the reports were completed in FT8 and WSPR.

Reports en WSPR	Reports en FT8
	

Think about adding an HF filter at the input, it improves listening.



It is possible to register on the following site https://www.receiverbook.de in order to declare your webradio or to listen to a station. If you want to make your webradio public, If you want to make your web radio public, you will need to get a software key and change the configuration in general settings.

The openwebRX software is much more powerful with a RaspberryPI 4. The new version of openwebRX from DD5JFK is exceptional and very well done. Unfortunately, the documentation provided by Jacob is not as good and lacks a lot of explanations.

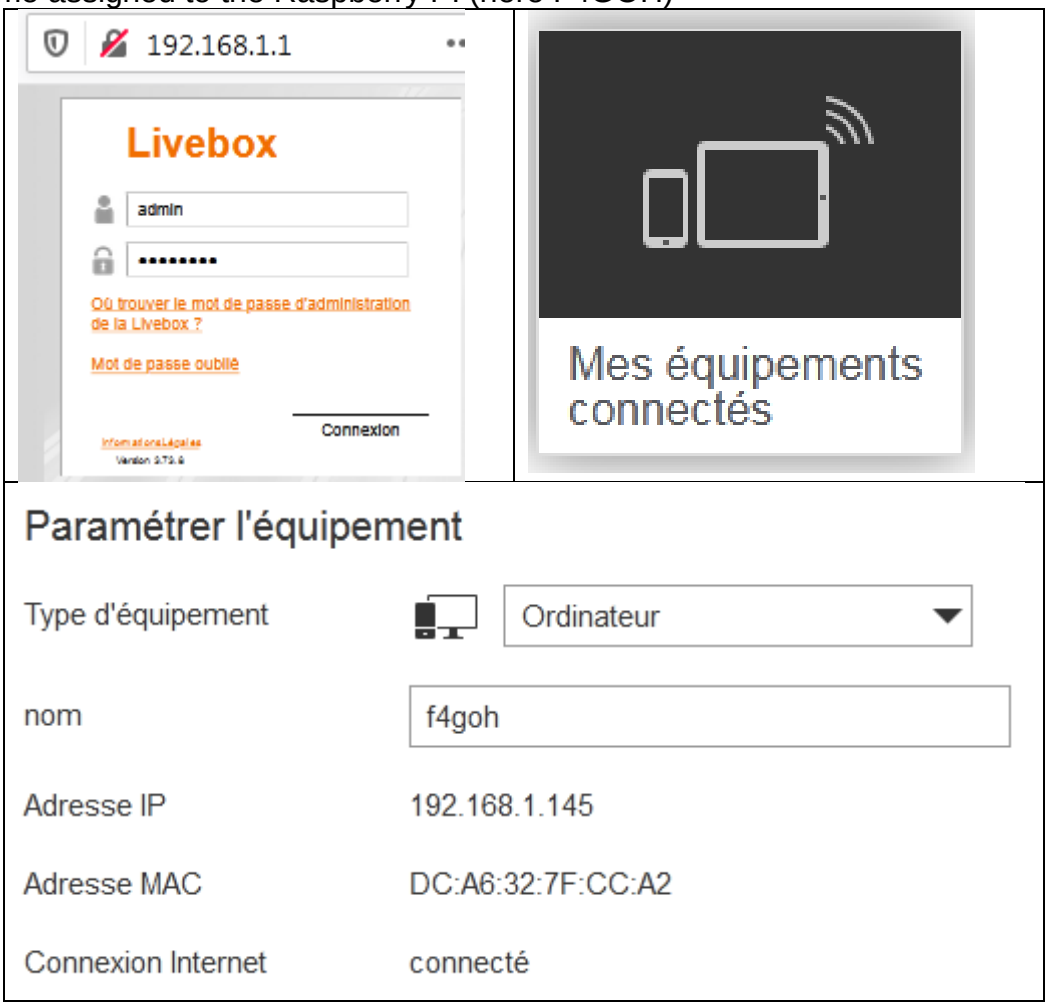
6.5 Openwebrx access out of the QRA

To access the Raspberry Pi from outside your QRA, you need to set up the ADSL box or the fiber so that it directs the connection request to the Raspberry PI on the local network. Not having at my disposal all the boxes of the different Internet service providers, I will only detail the configuration of the Orange LiveBox. For the other providers, it will be necessary to do an Internet research on port forwarding and domain name configuration.

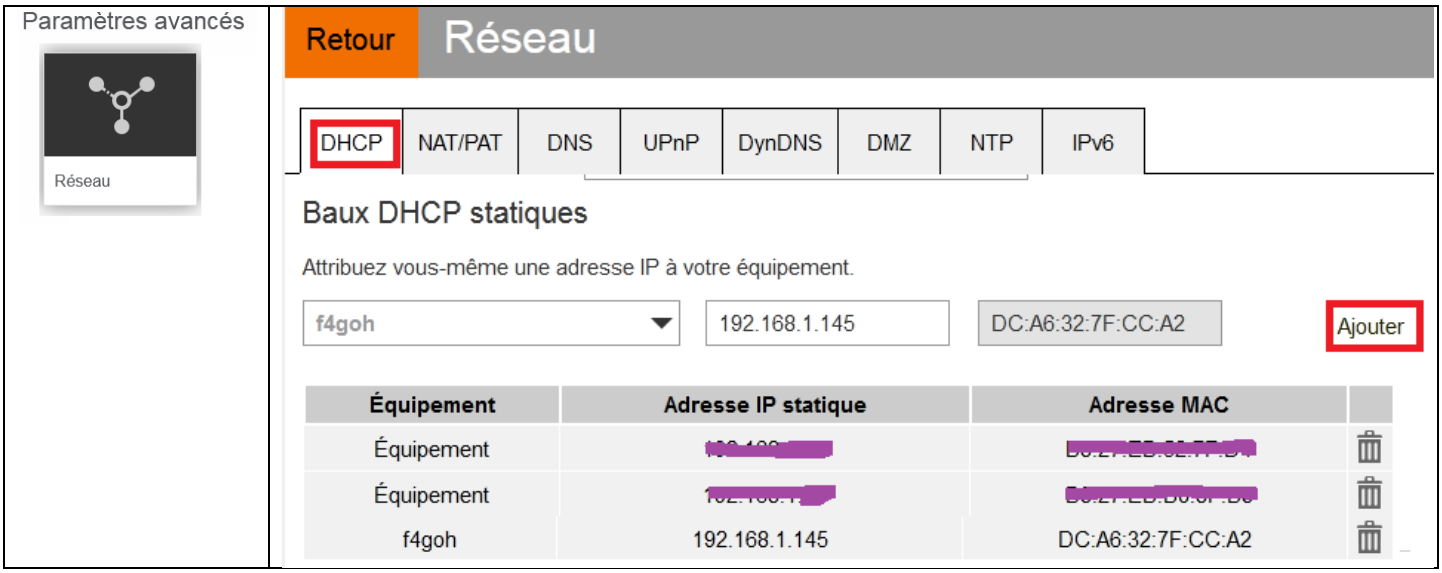
The configuration is done in 2 steps:

- Perform a port forwarding (8073) to the Raspberry PI with the IP address 192.168.1.145 for my case.
- Get a free domain name so that I don't remember by heart the IP address provided by the service provider.

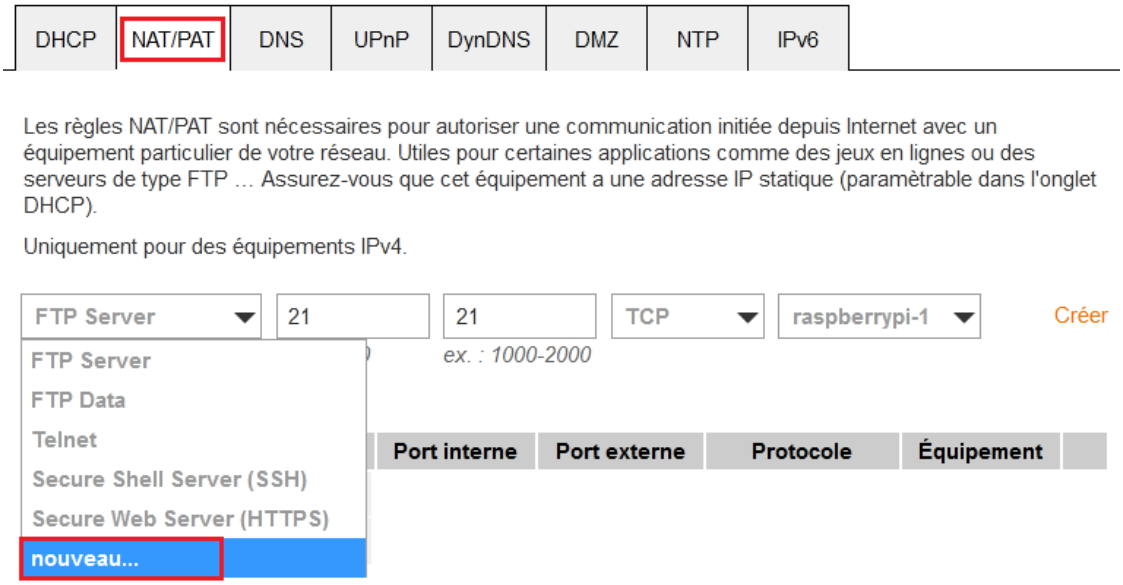
Check the name assigned to the Raspberry PI (here F4GOH)

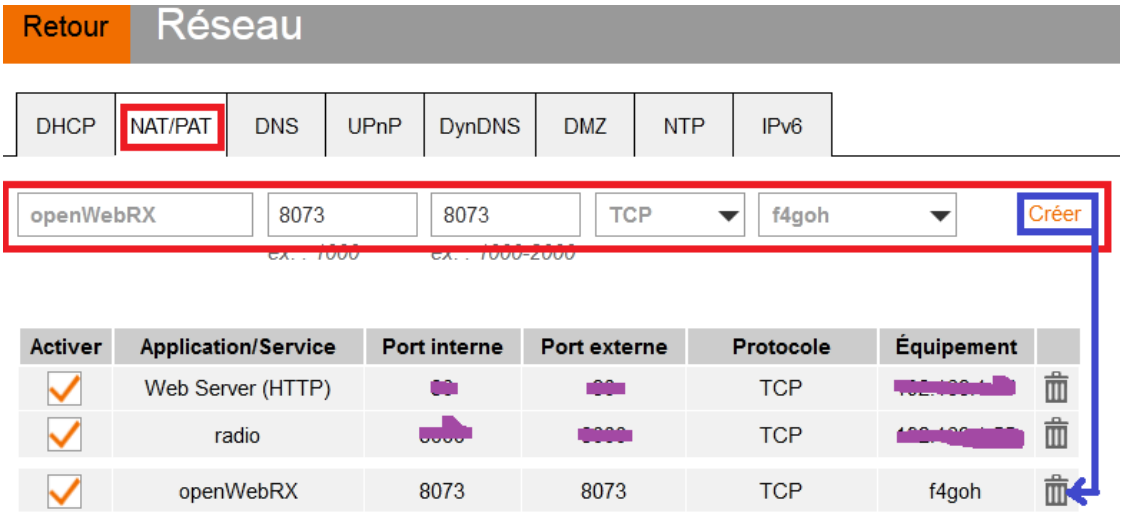


In the advanced settings, network menu, then DHCP, add your Raspberry Pi as static IP address.

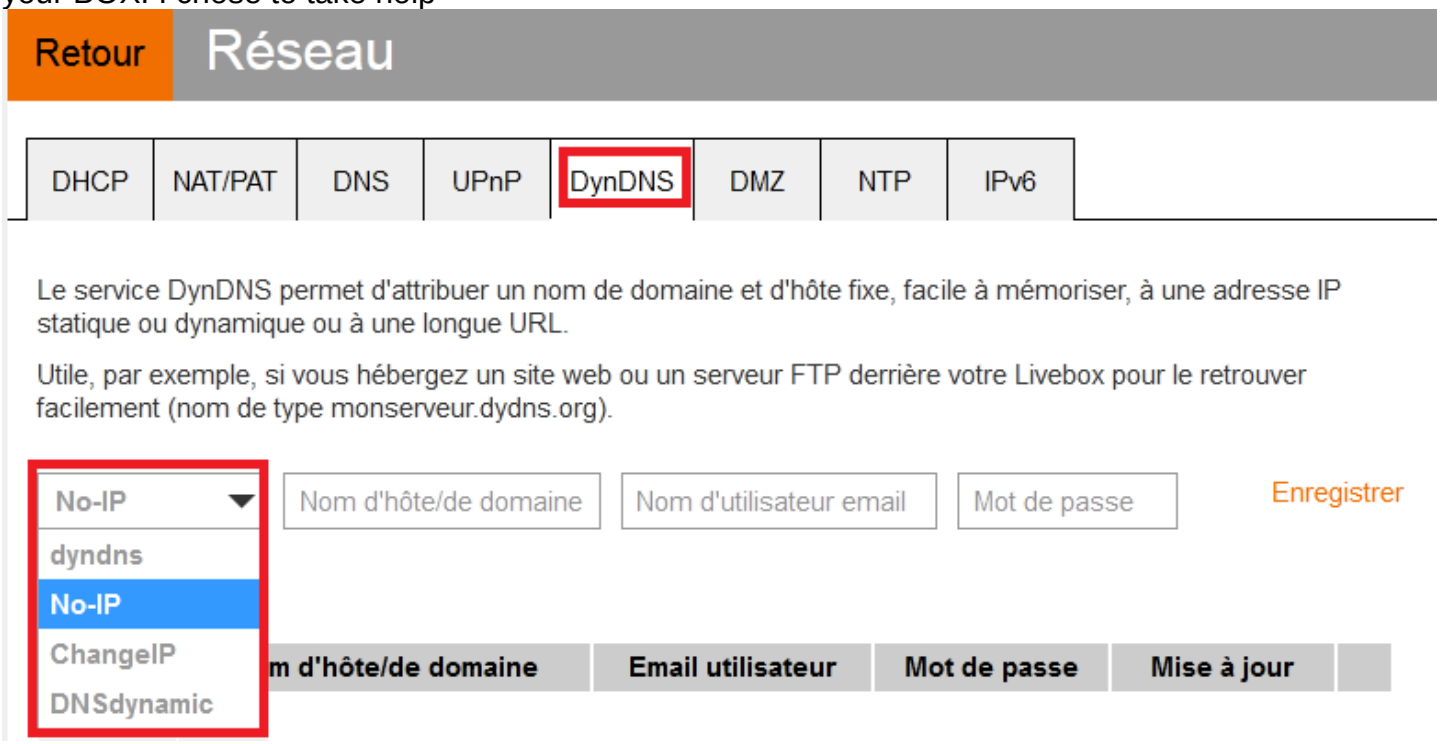


In the advanced settings, network menu, then NAT/PAT, add a new port forwarding to the Raspberry PI on port 8073.

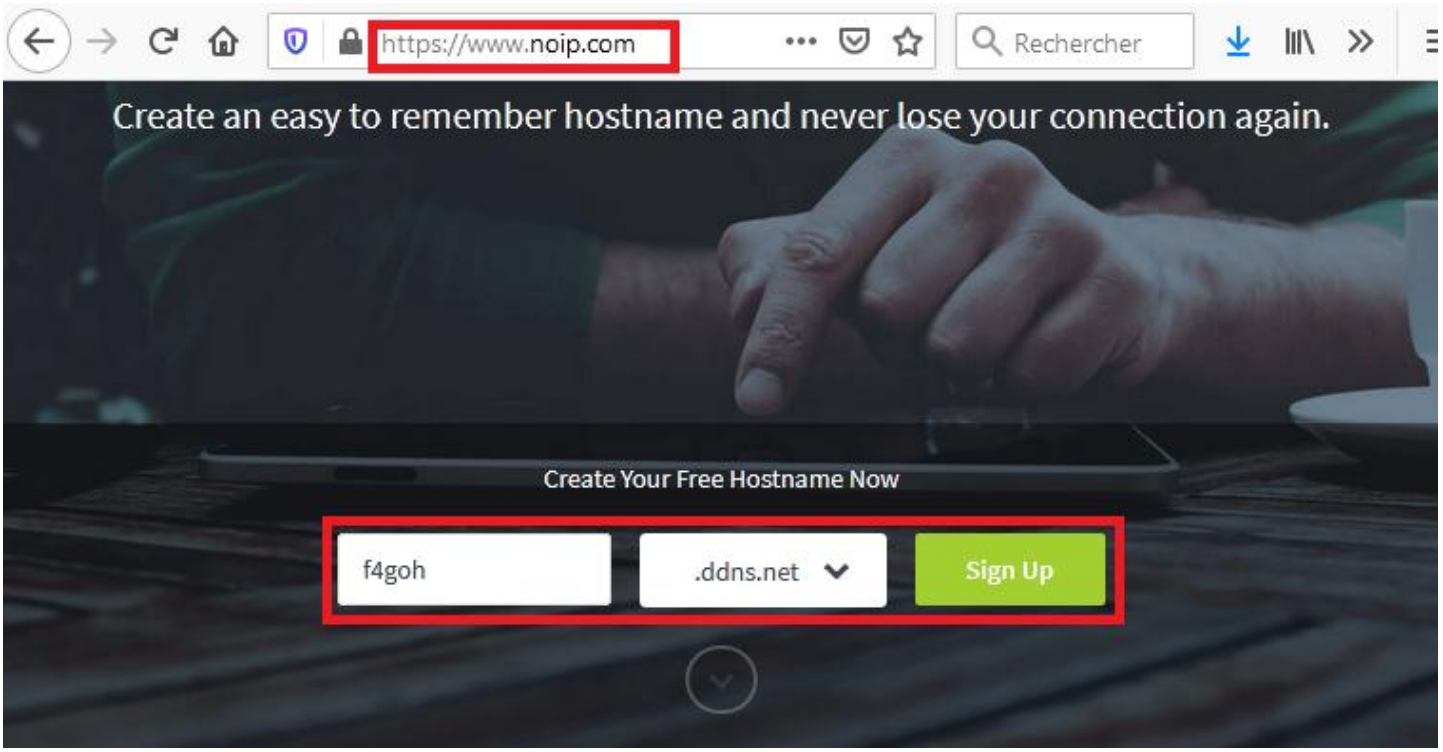




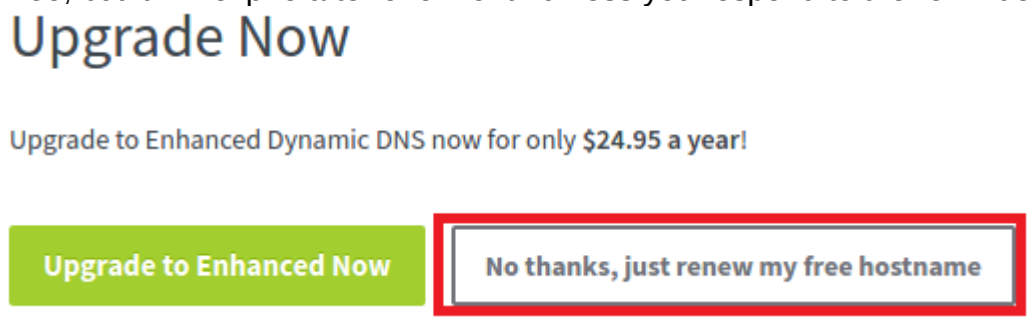
Still in the advanced settings, network menu, then DynDNS, consult the list of providers available for your BOX. I chose to take noip



Register on the https://www.noip.com/ website by selecting a domain name. Remember the password for future use.

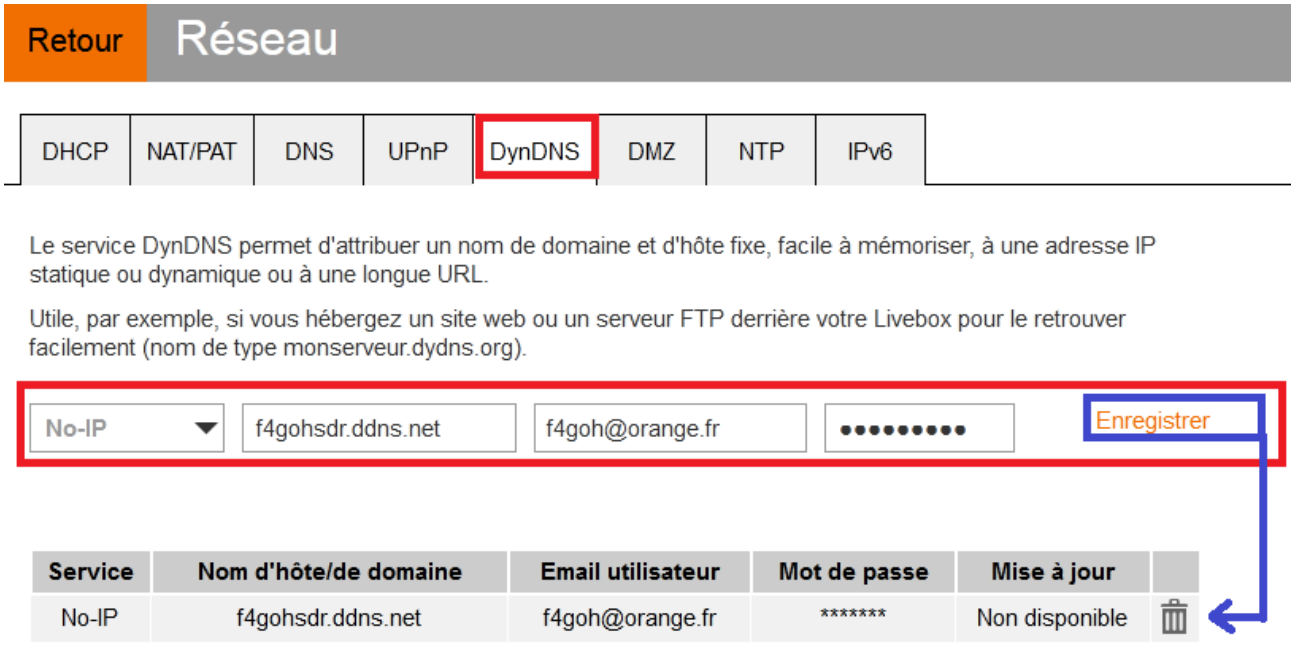


Registration is free, but it will expire after one month unless you respond to the reminder email.



You will have to do this every month or pay a subscription.

Back in the advanced settings, network menu, then DynDNS, enter the domain name previously chosen with the email account used for registration and its password.

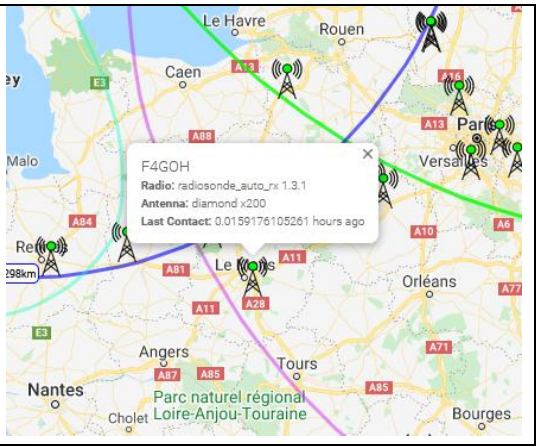


It is now possible to connect to openWebRX with the following address from outside the QRA:

Raspberry Pi for HAM Radio
Part 7

Summary:

Part 7: r2could and radiosonde auto rx



This tutorial only comments on the user manual or the existing wiki with additional screenshots.
Version 5/09/2020 V1.0

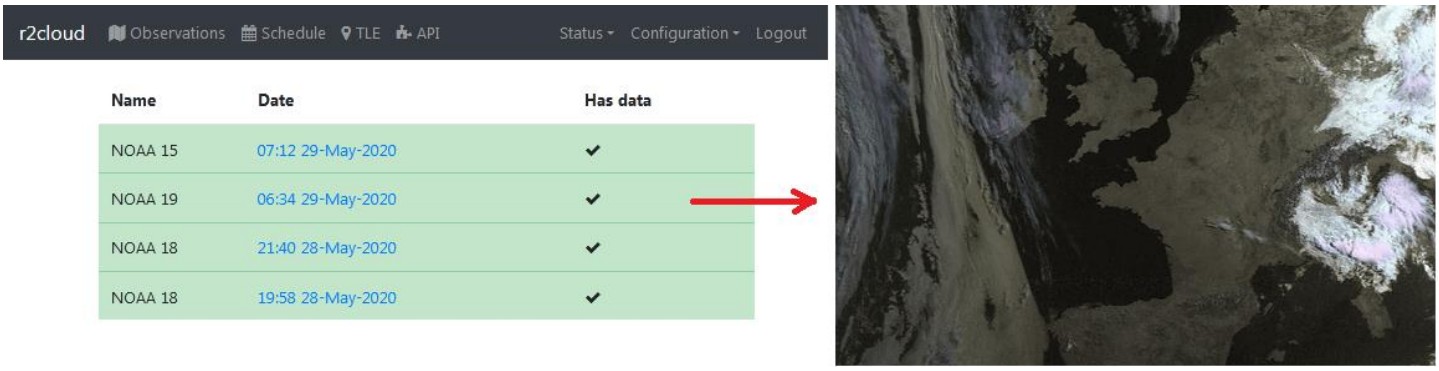
7.1 R2could introduction.

The NOAA (National Oceanic and Atmospheric Administration) satellites have been in orbit for a long time. They emit continuous weather images on 137Mhz. Currently there are 3 (NOAA15,18 and 19), they have been joined by a Russian satellite Meteor-M2 which broadcasts color images.

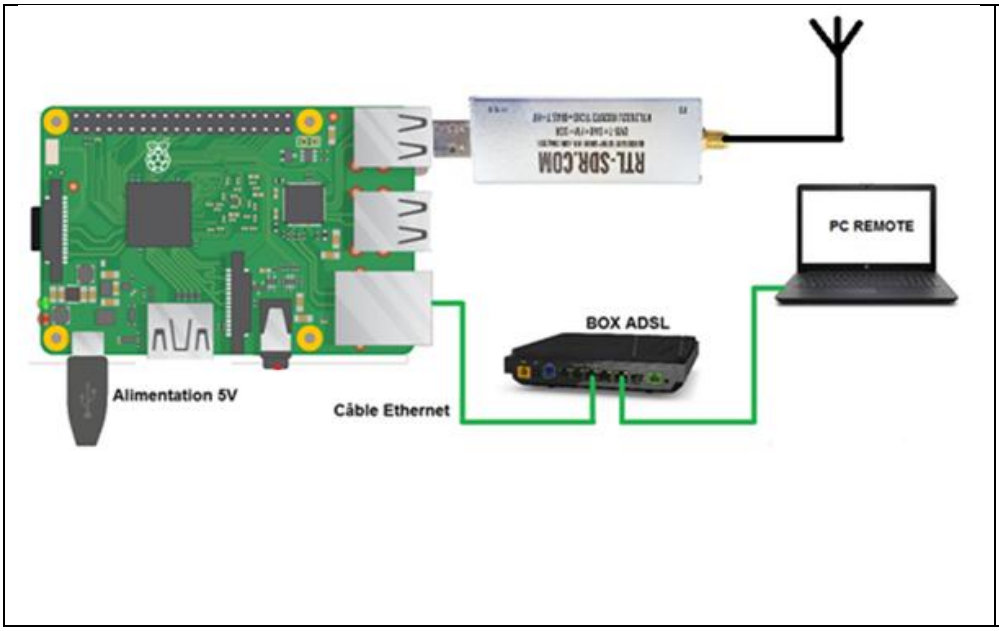
The advantage of using a Raspberry Pi to receive weather images is obvious, we have no regrets about leaving it on 24 hours a day. Its software automatically updates the orbital parameters of the satellites (TLE: "Two-Line Elements") and manages the reception and decoding of the images.

Moreover, r2could also decodes the telemetry of cubesats. As soon as a new cubesat is in service, the update is automatic.

For aerials, the use of a dual-band antenna is enough to start. But if you want to get the best possible reception, a 137 Mhz four-wire QFH antenna is easy to build.



On the hardware side, a Raspebrry Pi and an rtl-sdr key are sufficient. No monitor or keyboard connected to the Raspberry PI is required, even for commissioning.



7.2 R2Could Installation

https://github.com/dernasherbrezon/r2cloud

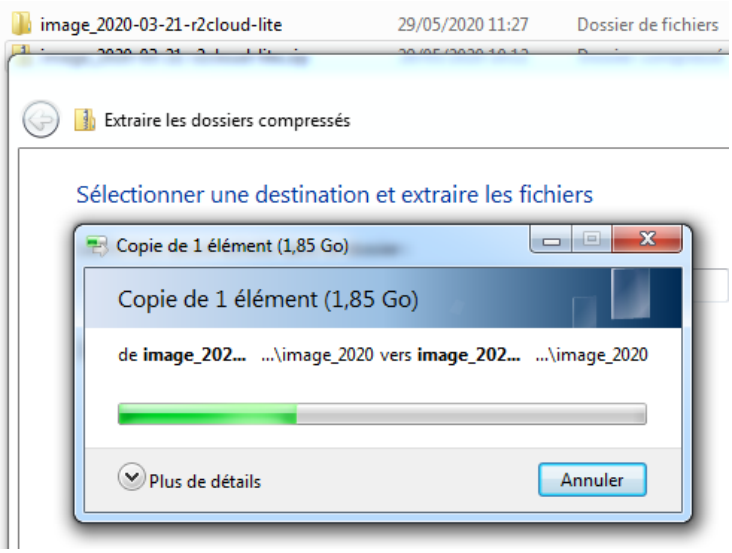
From the link above, download the file image_2020-03-21-r2cloud-lite.zip and choose the easy installation mode.

Installation

1. Install r2cloud

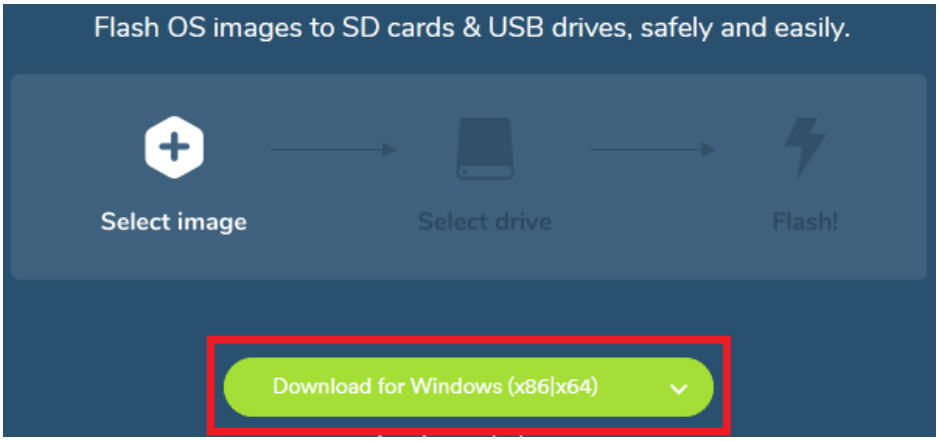
- From the image. This is simpliest way and require brand new SD card
  - Download the latest official image
  - Insert SD card into the card reader and flash it. You could use Etcher to do this
  - Insert SD card into the card reader and create file `r2cloud.txt` in the root directory. This file should contain any random string. This string is a login token. This token will be used during initial setup.

Extraire le fichier compressé

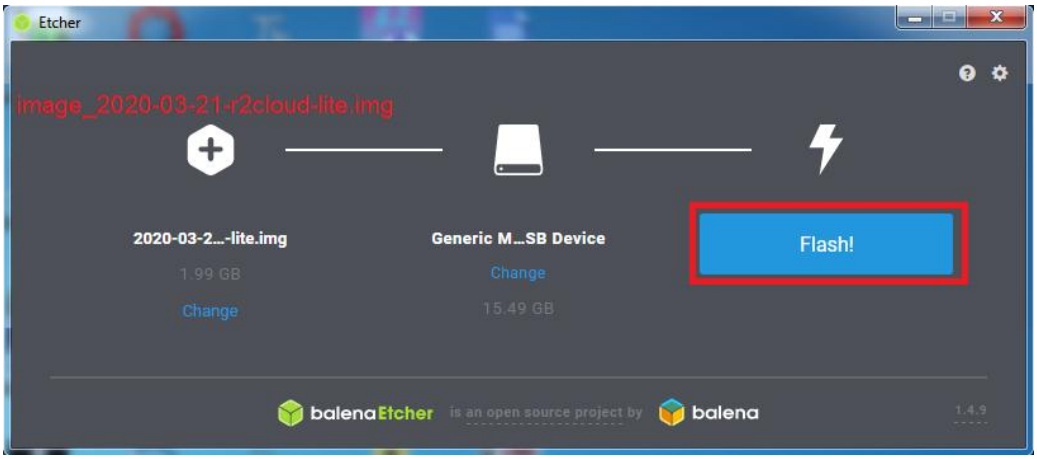


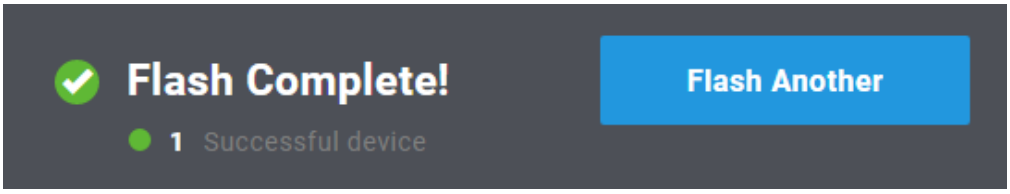
Download and install the utility and check to copy the image to an mSD card.

https://www.balena.io/etcher/



Select the file image_2020-03-21-r2cloud-lite.img, the reader of the mSD card, then click on Flash.

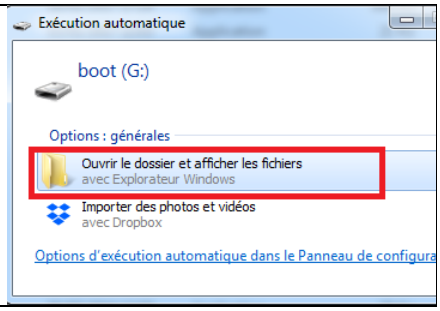




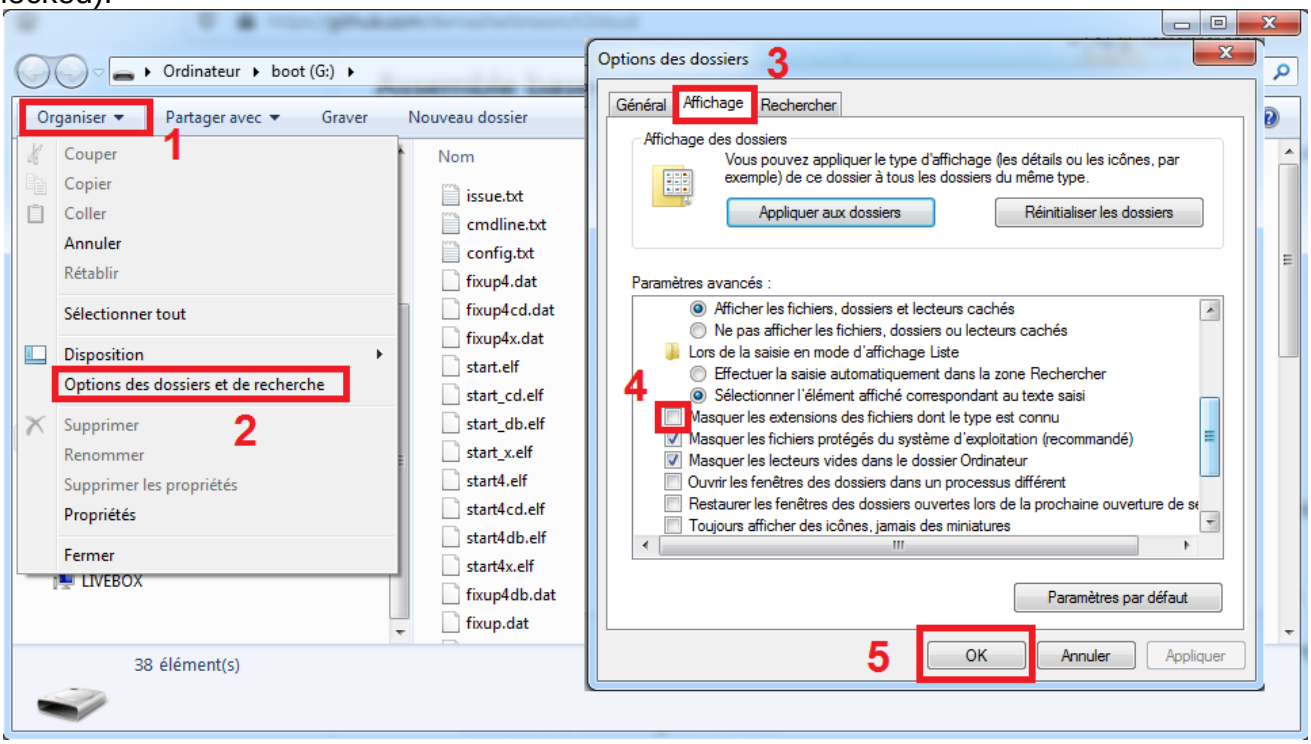
Once the copy is complete, unplug and plug the mSd card reader.

In Windows, the window below will appear.

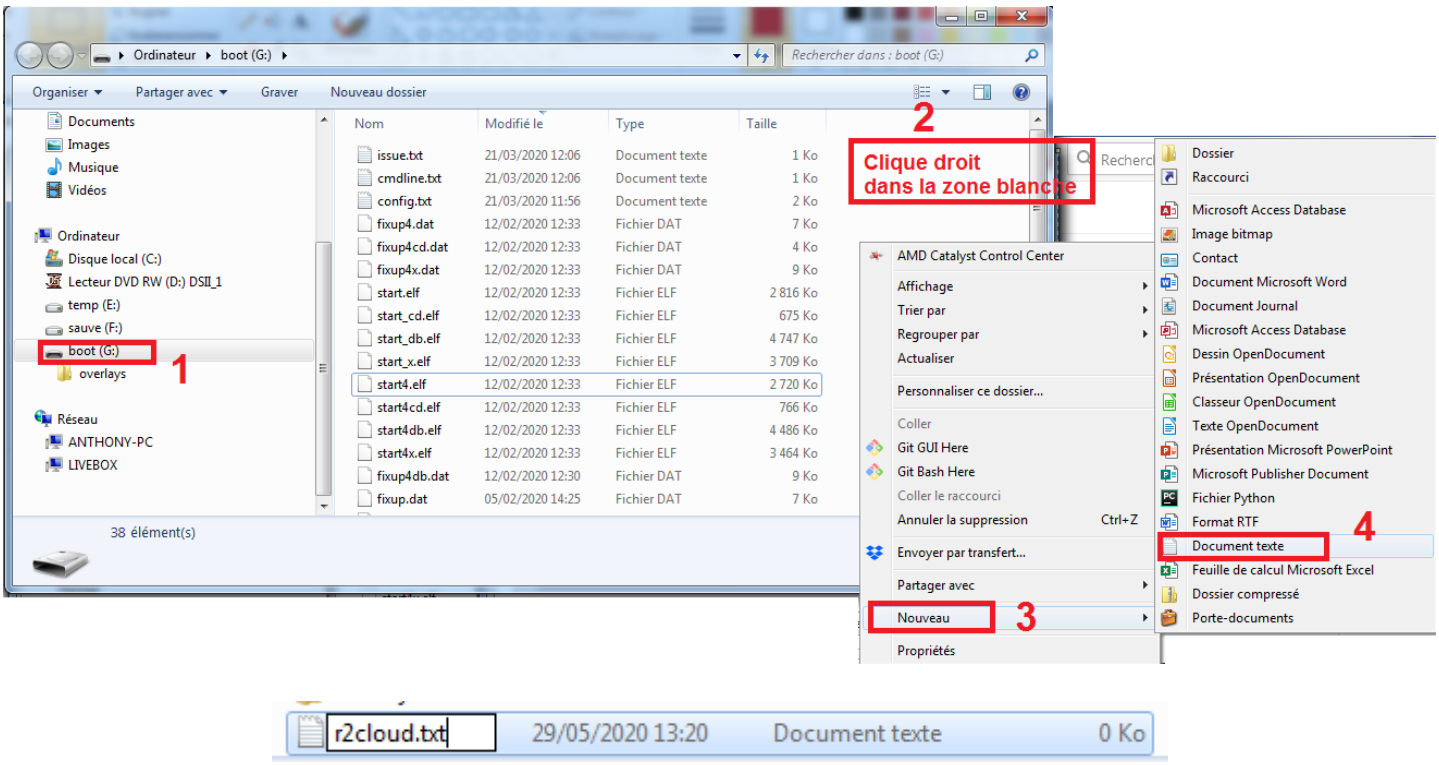
Use the file explorer in the partition named boot.



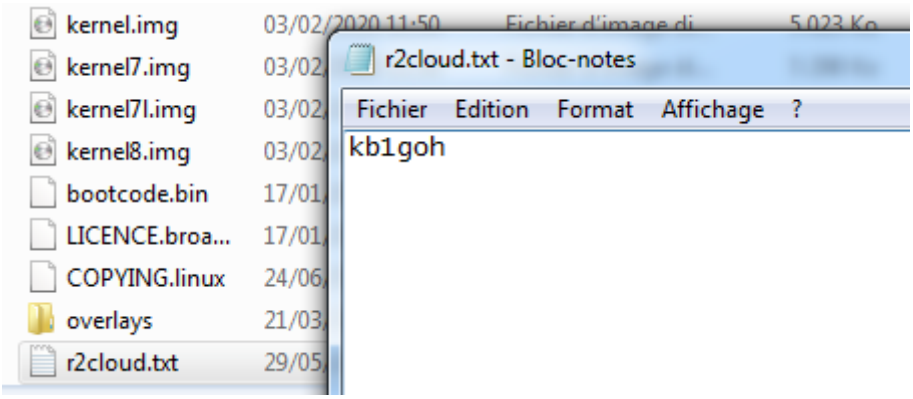
In the explorer, make sure that you can see the file extensions by following steps 1 to 5 (box 4 unchecked).



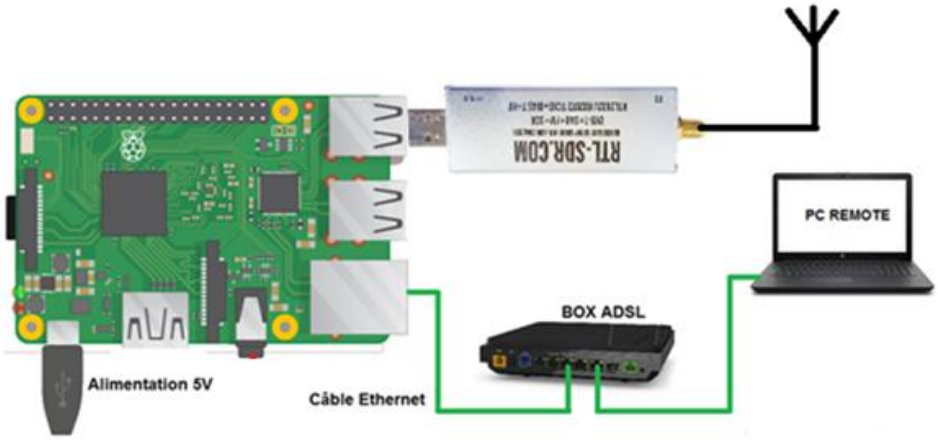
Create a new file named **`r2could.txt`**



Open the **`r2could.txt`** file and indicate his login. To make it simple, use his callsign. Don't forget to save the file.



Insert the mSd card into the Raspebrry Pi, then turn on the power with the rtl-sdr key connected:

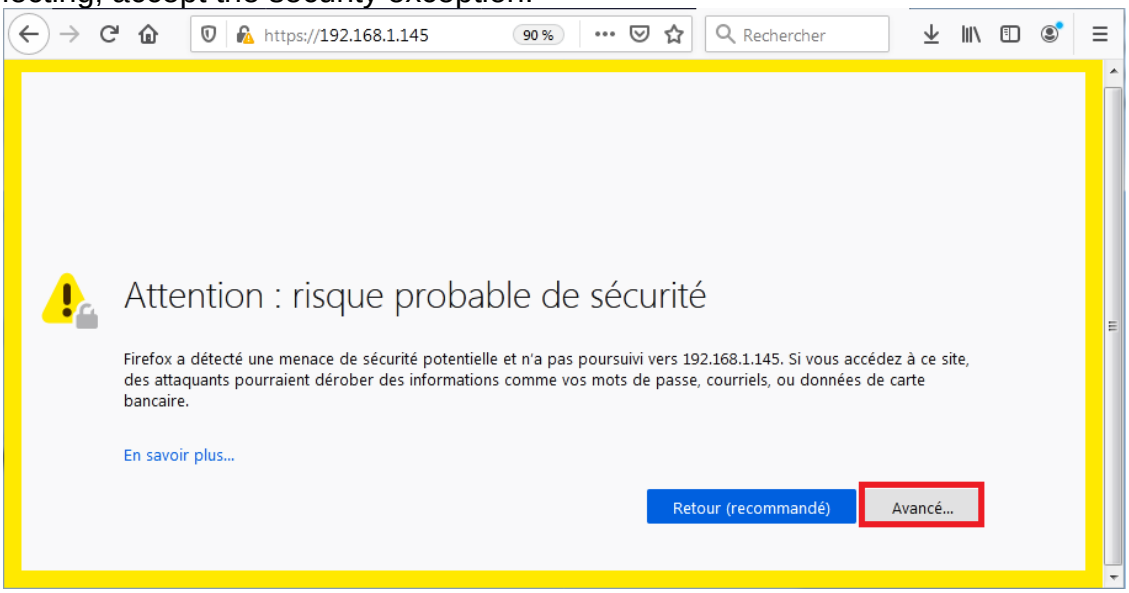


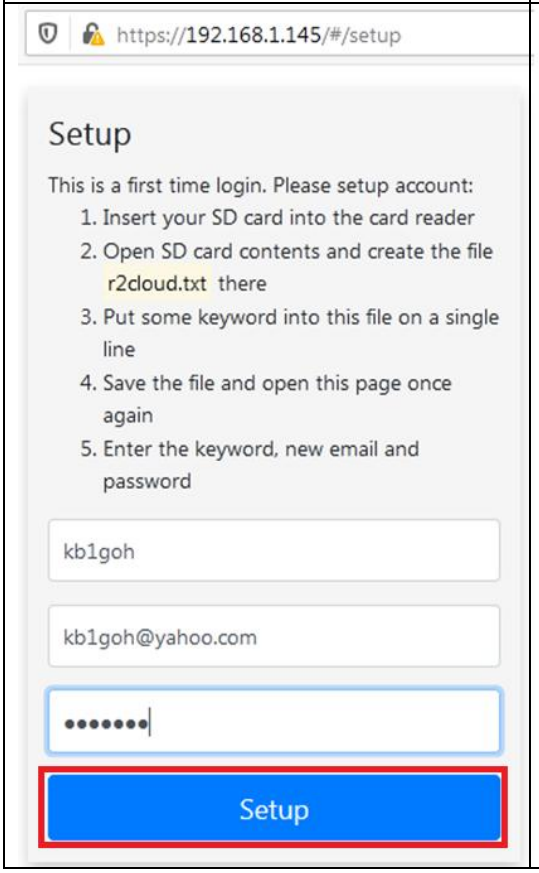
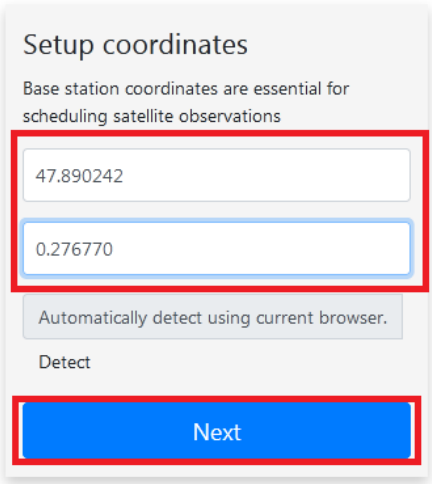
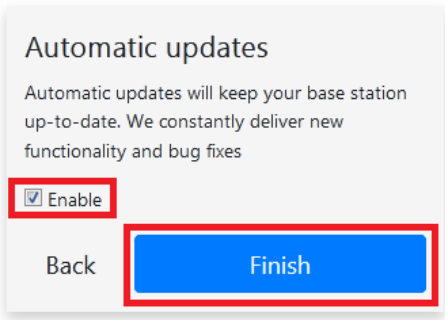
7.3 Start r2could

Use the domain name already preconfigured in the Raspberry PI https://raspberrypi.local, or use the IP address of its Raspberry PI connected on the local network.

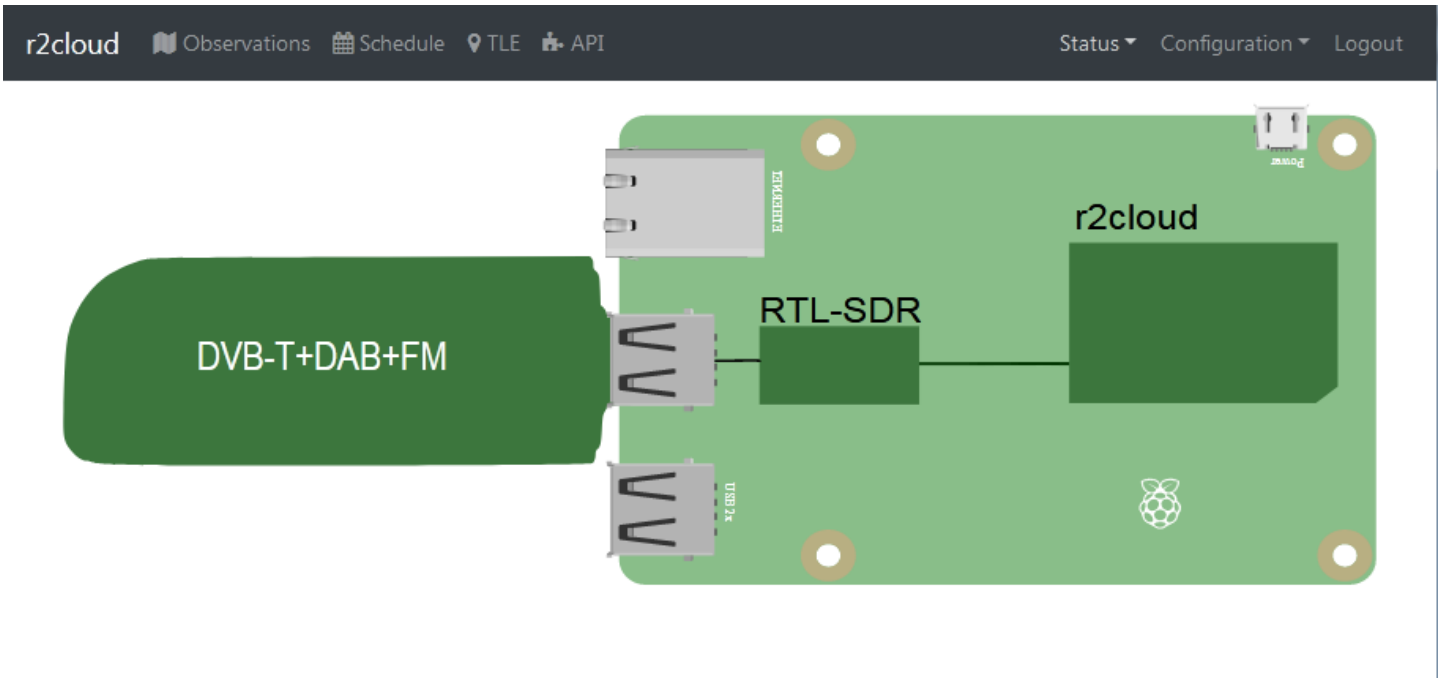
Reminder: To know the IP address of the Raspberry Pi, please refer to the tutorial Part 1.

When connecting, accept the security exception.

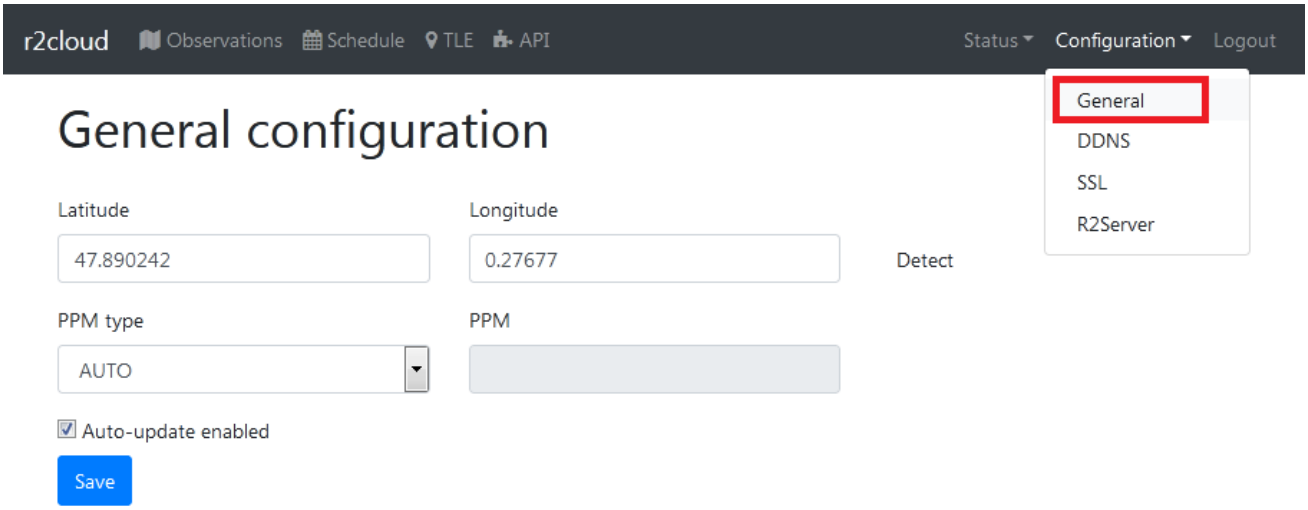


Enter the previously chosen login in the text file. Also enter an email and a password.	Enter its longitude and latitude coordinates. (Coordinates in decimal degrees)	Accept automatic updates.
		

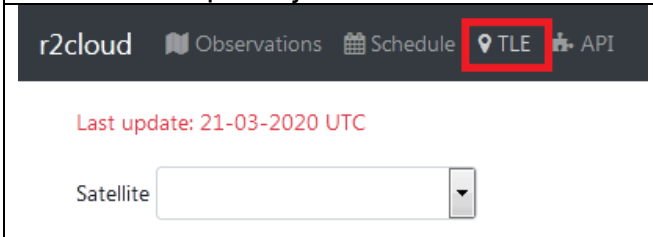
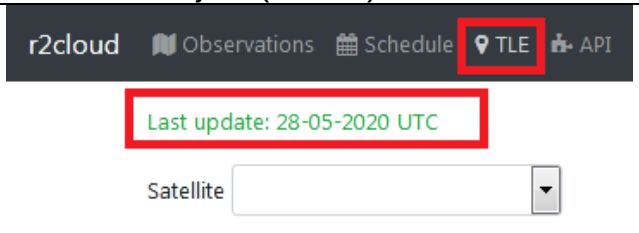
The main page must appear



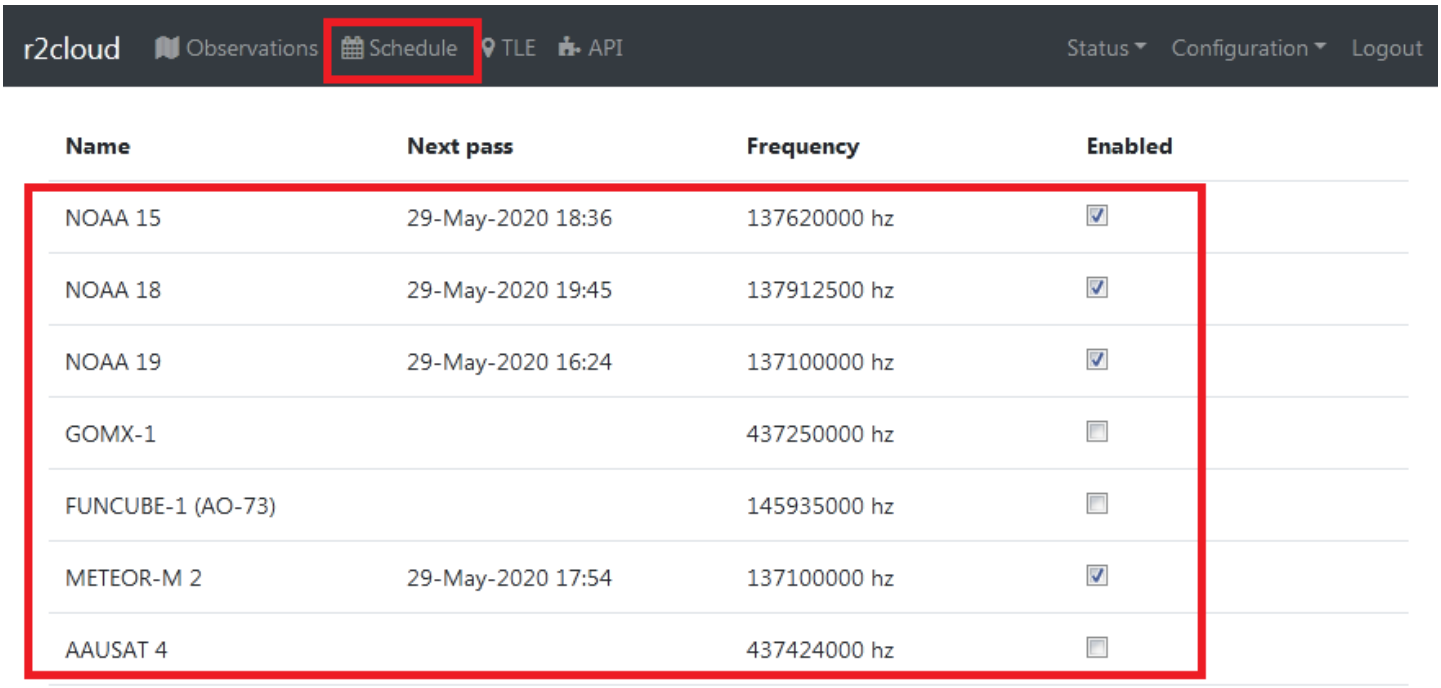
In the Configuration menu, General, check the accuracy of the geographic coordinates of the receiving site.



In the TLE menu, the update is not done yet. You will have to wait several hours.

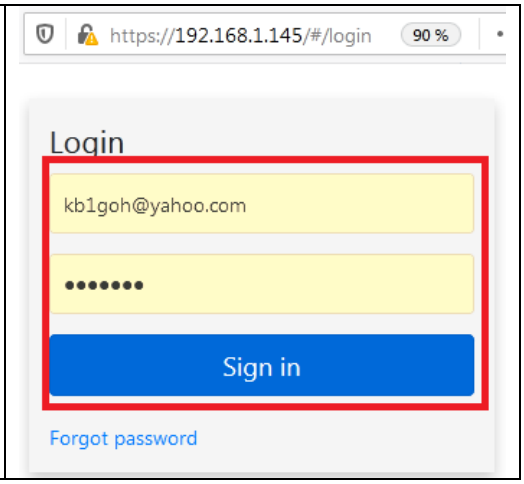
Le TLE n'est pas à jour	Le TLE est a jour (en vert)
	

In the Schedule menu, select the satellites to be received.

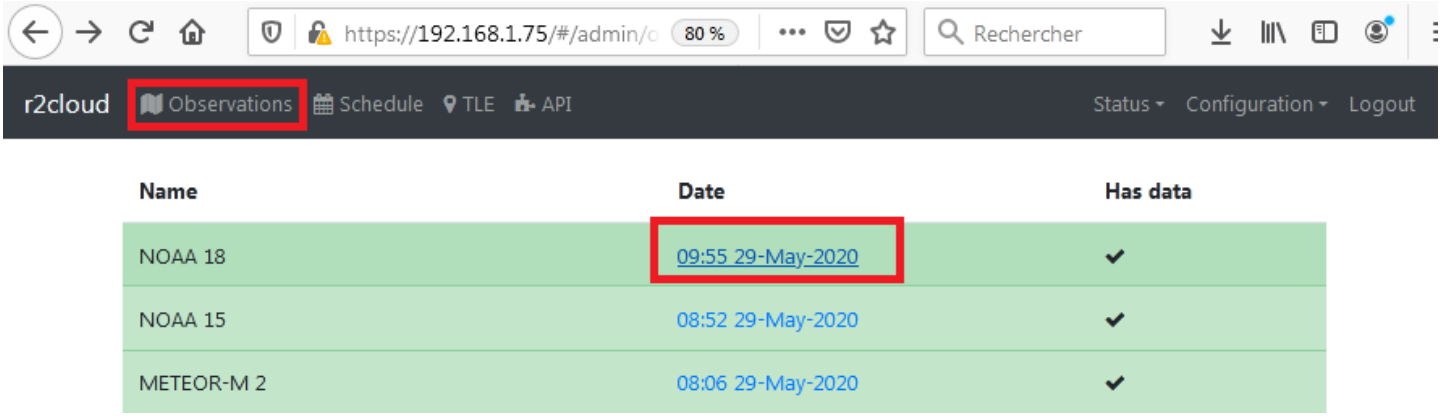


Leave the Raspberry Pi on and wait 24 to 48 hours.

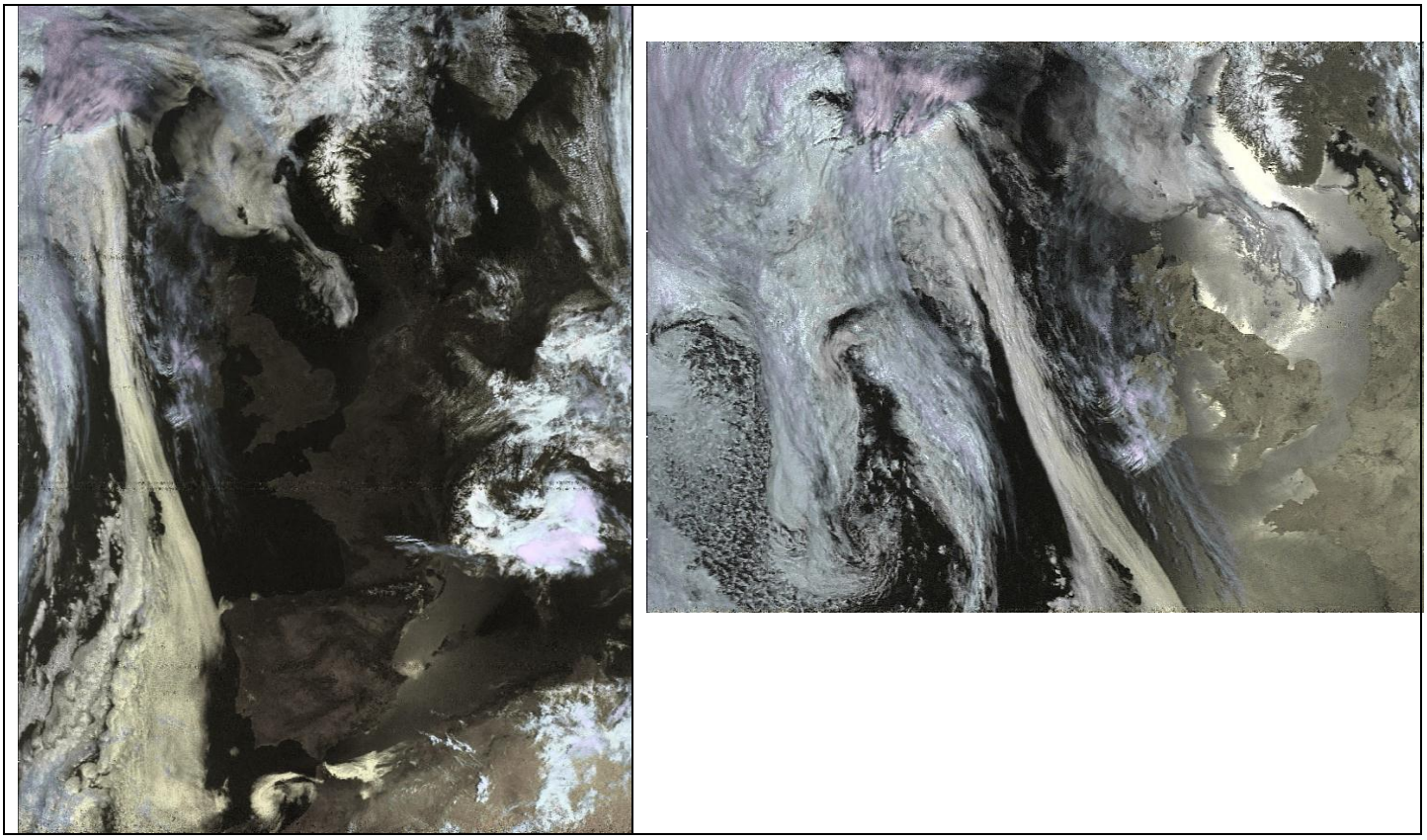
To see the images, you will have to log in.



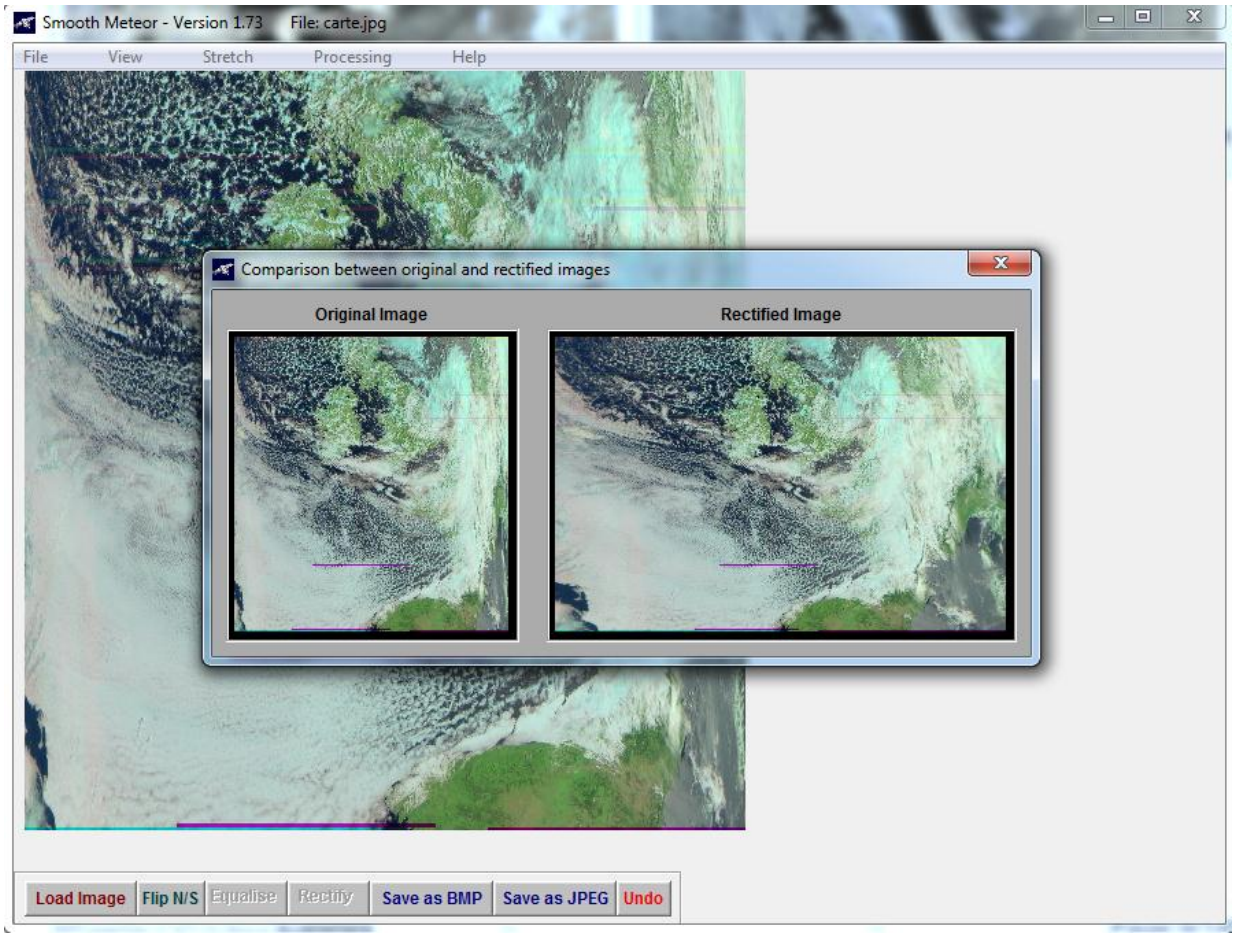
In the Observations menu, click on the date of the concerned satellite (if data have been received, the line appears in green color).



The received image will then appear.



The color images of the meteor satellite are distorted on the sides. This is normal, but the SmoothMeteor program can straighten these images.



7.4 Troubleshooting

If after 48 hours no image is received, the antenna and the connectivity between the rtl-sdr key and the Raspberry Pi must be checked.

The ssh connection with the Putty utility and the Raspberry PI is not possible. It will be necessary to connect a monitor and a keyboard on the Raspberry Pi.

`Raspberrypi login` **: pi**
`Password` **: raspberry**

Check rtl_sdr key :

rtl_test

```
pi@raspberrypi:~ $ rtl_test
Found 1 device(s):
  0:  Realtek, RTL2838UHIDIR, SN: 00000001

Using device 0: Generic RTL2832U OEM
Found Rafael Micro R820T tuner
Supported gain values (29): 0.0 0.9 1.4 2.7 3.7 7.7 8.7 12.5 14.4 15.7 16.6 19.7
 20.7 22.9 25.4 28.0 29.7 32.8 33.8 36.4 37.2 38.6 40.2 42.1 43.4 43.9 44.5 48.0
 49.6
[R82XX] PLL not locked!
Sampling at 2048000 S/s.

Info: This tool will continuously read from the device, and report if
samples get lost. If you observe no further output, everything is fine.

Reading samples in async mode...
^CSignal caught, exiting!
User cancel, exiting...
Samples per million lost (minimum): 0
pi@raspberrypi:~ $ ^C
```

ctrl+c

7.5 Radiosonde auto rx introduction

Like r2could for satellites, the radiosonde auto rx software allows to receive and display on a map the position of weather radiosondes sent regularly.

https://github.com/projecthorus/radiosonde_auto_rx/wiki

Currently, the models of radiosondes supported by the software are as follows:

- Vaisala RS92 (experimental support for the RS92-NGP)
- Vaisala RS41
- Graw DFM06/DFM09/DFM17/PS-15
- Meteomodem M10
- Intermet iMet-4 (and 'narrowband' iMet-1 sondes)
- Lockheed Martin LMS6, 400 MHz and 1680 MHz variants (including the new 'LMS-X' type)
- Meisei iMS-100

7.6 Radiosonde auto rx Installation

There is no ready-to-use image file. To install the radiosonde auto rx software, you will need to prepare an mSD card with "Raspbian Lite" and install the rtl-sdr libraries.

Preliminary Installation Summary:

- HAM Radio with the Raspberry Pi Part 4 (Complete installation except the backup of the mSd card which can be done later);
- HAM Radio with Raspberry Pi Part 5.6 Installation of the RTL-SDR key as summarized in the following commands:

Updating repositories and software: (it is recommended to use the PuTTY utility)

```
sudo apt update
sudo apt upgrade
```

Installing the git utility:

```
sudo apt install git
```

Installation of dependencies for radiosonde auto rx

```
sudo apt-get install python-numpy python-setuptools python-crcmod python-
requests python-dateutil python-pip sox git build-essential cmake usbutils
libusb-1.0-0-dev rng-tools libsamplerate-dev
```

```
pi@f4goh:~ $ sudo apt-get install python-numpy python-setuptools python-crcmod p
ython-requests python-dateutil python-pip sox git build-essential cmake usbutils
 libusb-1.0-0-dev rng-tools libsamplerate-dev
```

Installation of libraries for python programming software

```
sudo pip install flask flask-socketio
```

```
pi@f4goh:~ $ sudo pip install flask flask-socketio
Looking in indexes: https://pypi.org/simple, https://www.piwheels.org/simple
Collecting flask
  Downloading https://files.pythonhosted.org/packages/f2/28/2a03252dfb9ebf377f40
fba6a7841b47083260bf8bd8e737b0c6952df83f/Flask-1.1.2-py2.py3-none-any.whl (94kB)
    100% |                                | 102kB 141kB/s
Collecting flask-socketio
  Downloading https://files.pythonhosted.org/packages/f9/01/2090f37ca90e0020008e
```

You will find the installation of the rtl-sdr radio receiver key as in part 5.6 of the tutorial.

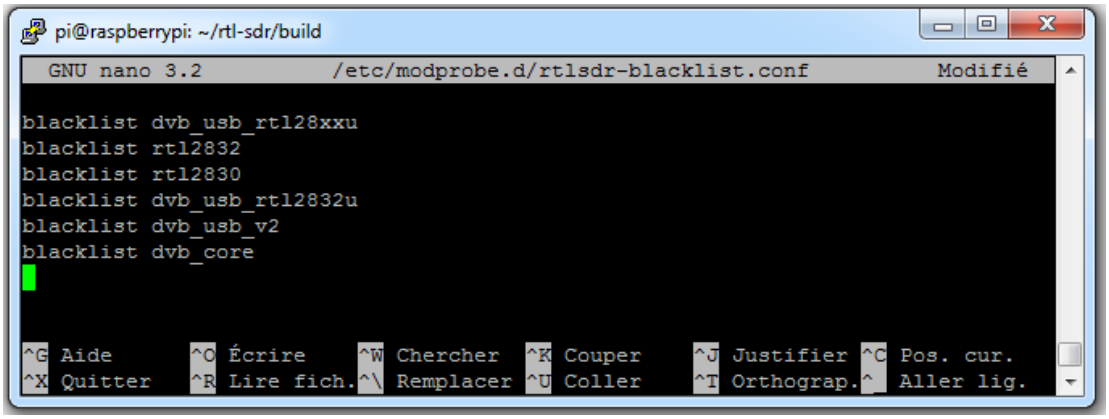
```
git clone https://github.com/osmocom/rtl-sdr.git
cd rtl-sdr
mkdir build
cd build
cmake -DINSTALL_UDEV_RULES=ON -DDETACH_KERNEL_DRIVER=ON ../
sudo make install
sudo ldconfig
```

```
pi@raspberrypi:~/rtl-sdr/build $ sudo nano /etc/modprobe.d/rtlsdr-blacklist.conf
```

```
sudo nano /etc/modprobe.d/rtlsdr-blacklist.conf
```

```
blacklist dvb_usb_rtl28xxu
blacklist rtl2832
blacklist rtl2830
blacklist dvb_usb_rtl2832u
blacklist dvb_usb_v2
blacklist dvb_core
```

Copy and paste the above list into the nano editor. Right click reminder: to paste the text in the nano editor.

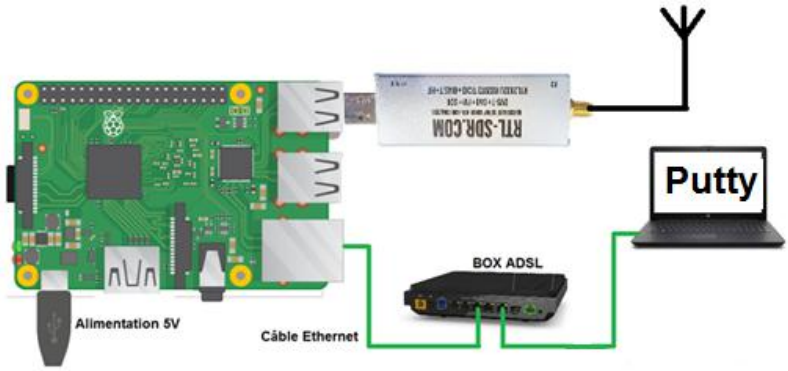


Save the file using the Ctrl + o keys, then the enter key to validate the recording.	**Exit the editor using Ctrl + x.**
	

Restart the Raspberry PI: `pi@raspberrypi:~/rtl-sdr/build $ sudo reboot`

```
sudo reboot
```

Verification: Plug the rtl-sdr key into a USB port on the Raspberry Pi. Connect the rtl-sdr key to the Raspberry Pi on a free USB port.



Check the presence of the rtl-sdr key:

With LXTerminal or Putty, type the following command line:

```
lsusb
```

Find rtl-sdr key : **RTL238 DVB-T**

Raspberry Pi 3 :

```
pi@raspberrypi:~ $ lsusb
Bus 001 Device 004: ID 0bda:2838 Realtek Semiconductor Corp. RTL2838 DVB-T
Bus 001 Device 003: ID 0424:ec00 Standard Microsystems Corp. SMSC9512/9514 Fast
Ethernet Adapter
Bus 001 Device 002: ID 0424:9514 Standard Microsystems Corp. SMC9514 Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

Raspberry Pi 4 :

```
pi@raspberrypi:~ $ lsusb
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 003: ID 0bda:2838 Realtek Semiconductor Corp. RTL2838 DVB-T
Bus 001 Device 002: ID 2109:3431 VIA Labs, Inc. Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

With LXTerminal or Putty, type the following command line: (be careful, underscore key)

```
rtl_test
```

The rtl-sdr key should be recognized, if not, disconnect and reconnect the key and restart the test.

```
pi@raspberrypi:~ $ rtl_test
Found 1 device(s):
  0:  Realtek, RTL2838UHIDIR, SN: 00000001

Using device 0: Generic RTL2832U OEM
Found Rafael Micro R820T tuner
Supported gain values (29): 0.0 0.9 1.4 2.7 3.7 7.7 8.7 12.5 14.4 15.7 16.6 19.7
 20.7 22.9 25.4 28.0 29.7 32.8 33.8 36.4 37.2 38.6 40.2 42.1 43.4 43.9 44.5 48.0
 49.6
[R82XX] PLL not locked!
Sampling at 2048000 S/s.

Info: This tool will continuously read from the device, and report if
samples get lost. If you observe no further output, everything is fine.

Reading samples in async mode...
^CSignal caught, exiting!
User cancel, exiting...
Samples per million lost (minimum): 0
pi@raspberrypi:~ $ ^C
```

ctrl+c

Radiosonde Auto RX software Installation

```
git clone https://github.com/projecthorus/radiosonde_auto_rx.git
```

```
ls
```

```
pi@f4goh:~ $ git clone https://github.com/projecthorus/radiosonde_auto_rx.git
Clonage dans 'radiosonde_auto_rx'...
remote: Enumerating objects: 37, done.
remote: Counting objects: 100% (37/37), done.
remote: Compressing objects: 100% (27/27), done.
remote: Total 5202 (delta 17), reused 18 (delta 10), pack-reused 5165
Réception d'objets: 100% (5202/5202), 12.45 MiB | 1008.00 KiB/s, fait.
Résolution des deltas: 100% (3551/3551), fait.
pi@f4goh:~ $ ls
radiosonde_auto_rx  rtl-sdr
pi@f4goh:~ $
```

Don't forget the tab key TAB. The reflex for typing the 2 command lines without copying and pasting in PuTTY must be: cd ra TAB a TAB ./b TAB	

```
cd radiosonde_auto_rx/auto_rx
./build.sh
```

```
pi@f4goh:~ $ cd radiosonde_auto_rx/auto_rx/
pi@f4goh:~/radiosonde_auto_rx/auto_rx $ ./build.sh
Building dft_detect
Building RS92/RS41/DFM/LMS6/iMS Demodulators
Building LMS6-1680 Demodulator.
Building iMet Demodulator.
Building fsk-demod utils from codec2
Copying files into auto_rx directory.
Done!
pi@f4goh:~/radiosonde_auto_rx/auto_rx $
```

Copy the configuration file

```
cp station.cfg.example station.cfg
```

Edit the configuration file **station.cfg** with the nano editor

```
pi@f4goh:~/radiosonde_auto_rx/auto_rx $ nano station.cfg
```

https://github.com/projecthorus/radiosonde_auto_rx/wiki/Configuration-Settings

Below is the minimum configuration:

Modification	Avant	Après
```# RADIOSONDE SEARCH SETTINGS #``` Reception frequencies min and max	```# the sondes are min_freq = 400.05 max_freq = 403.0```	```# the sondes are min_freq = 400.05 max_freq = 406.0```
```# STATION LOCATION #``` Define the geographical position of the listening station. Coordinates in decimal degrees.	```# Used by the Habitat [location] station_lat = 0.0 station_lon = 0.0 station_alt = 0.0```	```# Used by the Habitat Upl [location] station_lat = 47.890242 station_lon = 0.276770 station_alt = 80.0```

```# HABITAT UPLOAD SETTINGS #```
Identification on the tracker.habhub.org website
	``habitat enabled = False
	# Uploader callsign, as shown above.
	uploader callsign = CHANGEME AUTO RX
	# Upload listener position to Habitat
	upload_listener_position = True
	# Uploader Antenna Description.
	# If upload_listener_position is enab
	uploader_antenna = 1/4 wave monopole```
		``habitat_enabled = True
		# Uploader callsign, as shown a
		uploader callsign = F4GOH
		# Upload listener position to H
		upload_listener_position = True
		# Uploader Antenna Description.
		# If upload_listener_position i
		uploader_antenna = diamond x200```

7.7 Start radiosonde auto rx

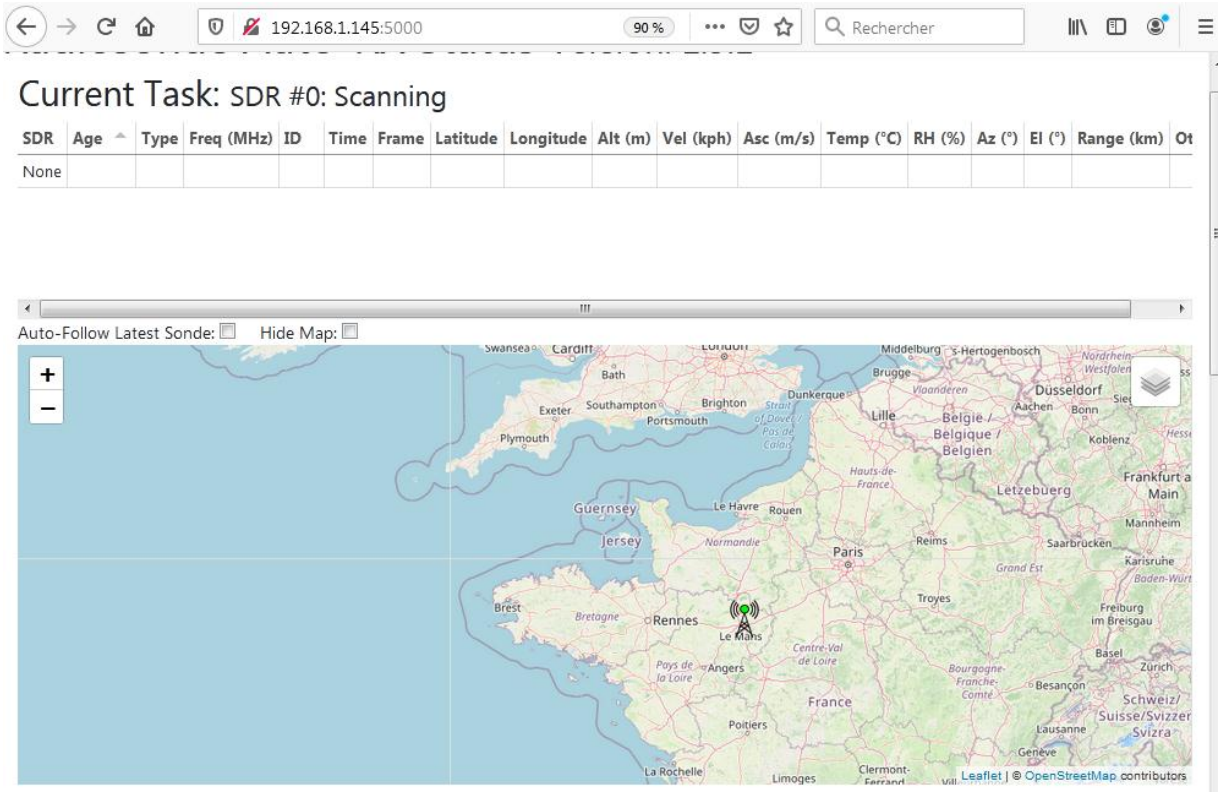
Run the program with python. (Make sure to run the program from the radiosonde_auto_rx/auto_rx directory)

```
python auto_rx.py
```

```
pi@f4goh:~/radiosonde_auto_rx/auto_rx $ python auto_rx.py
2020-05-30 09:43:36,434 INFO:Reading configuration file...
2020-05-30 09:43:39,303 INFO:Config - Tested SDR #0 OK
2020-05-30 09:43:39,309 INFO:Started Flask server on http://0.0.0.0:5000
 * Serving Flask app "autorx.web" (lazy loading)
 * Environment: production
2020-05-30 09:43:39,313 INFO:Telemetry Logger - Started Telemetry Logger Thread.
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
2020-05-30 09:43:39,315 INFO:OziMux - Started OziMux / Payload Summary Exporter
2020-05-30 09:43:39,755 INFO:Version - Local Version: 1.3.1  Current Master Version: 1.3.1
2020-05-30 09:43:39,756 INFO:SDR #0 has been allocated to Scanner.
2020-05-30 09:43:42,608 INFO:Scanner #0 - Starting Scanner Thread
2020-05-30 09:43:42,613 INFO:Scanner #0 - Running frequency scan.
```

Once the program radiosonde auto rx is executed, open with a browser the web page with the IP address of the Raspberry PI on port 5000.

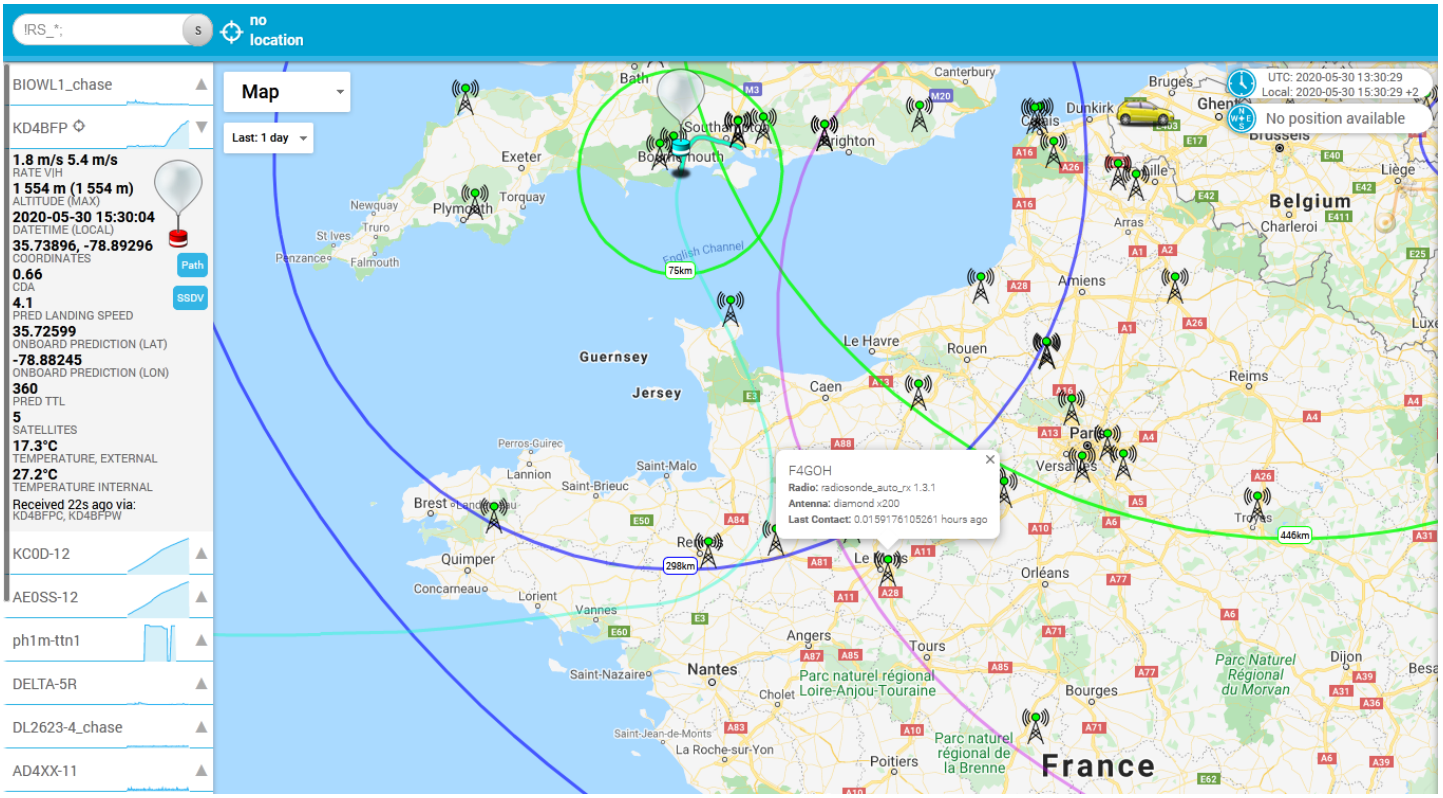
http://192.168.1.145:5000/ or with your domain name http://f4goh:5000/



A map should appear with, if any, the position of the balloons.

With the browser, go to the following WEB page:

http://tracker.habhub.org/



Your station should be listed. This site also indicates the presence of probe balloons and the possibility for your station to decode them.

To exit the program, press ctrl+c

```
403.05 402.98 402.92]
^C2020-05-30 10:07:11,924 INFO:Web - Flask Server Shutdown.
2020-05-30 10:07:11,925 INFO:Starting shutdown of all threads.
2020-05-30 10:07:11,926 INFO:Scanner #0 - Waiting for current scan to finish...
2020-05-30 10:07:23,134 INFO:Scanner #0 - Scanner Thread Closed.
2020-05-30 10:07:23,402 INFO:Telemetry Logger - Stopped Telemetry Logger Thread.
pi@f4goh:~/radiosonde_auto_rx/auto_rx $
```

ctrl+c

Automatically start "radiosonde auto rx" when starting the Raspberry PI

(From the radiosonde_auto_rx/auto_rx folder)

```
cd ~/radiosonde_auto_rx/auto_rx/
```

Copy the startup file

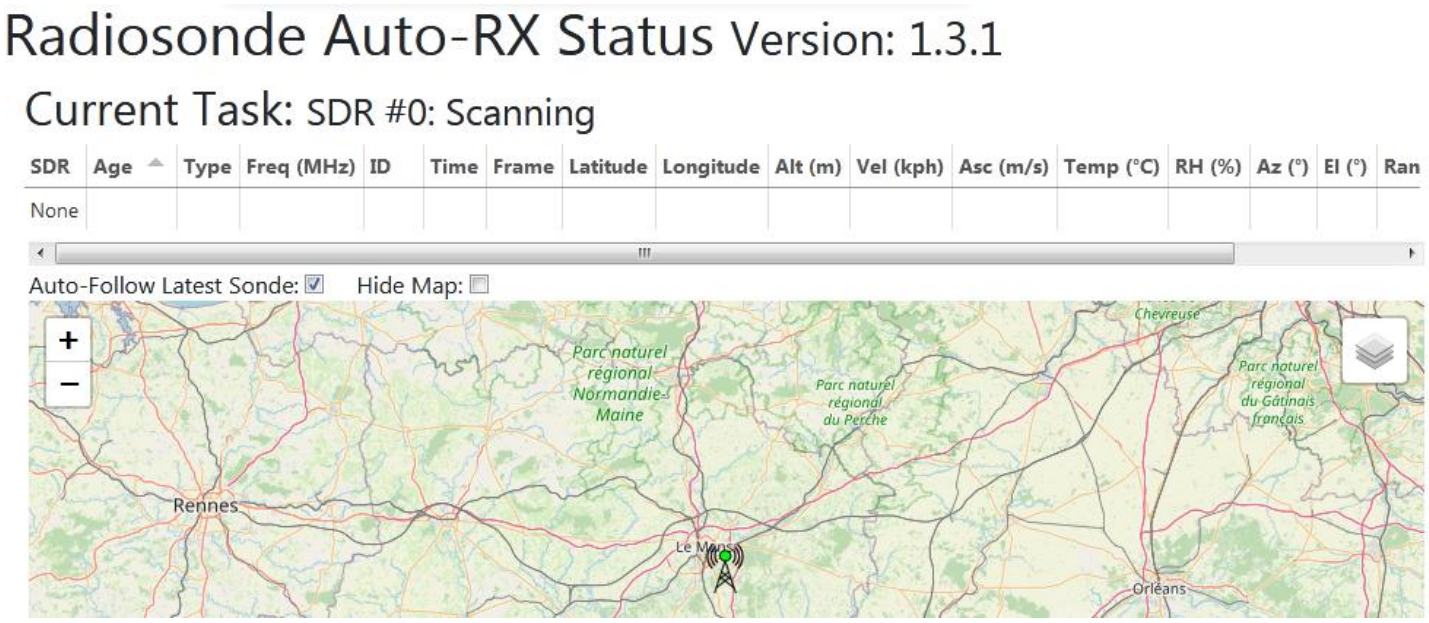
```
sudo cp auto_rx.service /etc/systemd/system/
```

Restart the Raspberry PI

```
sudo reboot
```

Once the program radiosonde auto rx is executed, open with a browser the web page with the IP address of the Raspberry PI on port 5000.

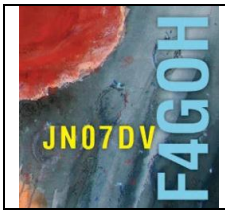
http://192.168.1.145:5000/ or with your domain name http://f4goh:5000/



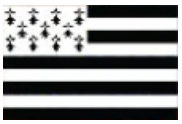
To remove "auto rx radiosonde" on startup of the Raspberry PI, perform the following command and then restart.

```
sudo rm /etc/systemd/system/auto_rx.service
```

To be continued:
Flight tracking of ADS-B aircraft: https://flightaware.com/adsb/piaware/build

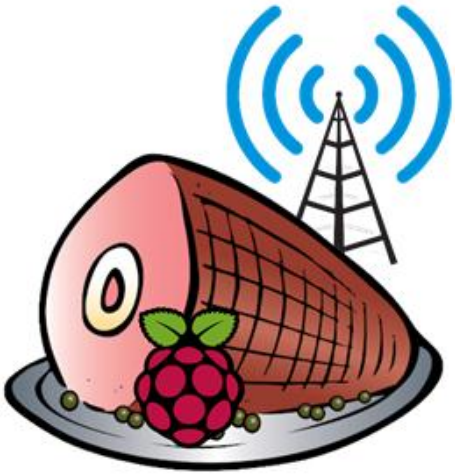


Raspberry Pi for HAM Radio
Part 8



Summary:

Part 8 : HamPi and RadioPI



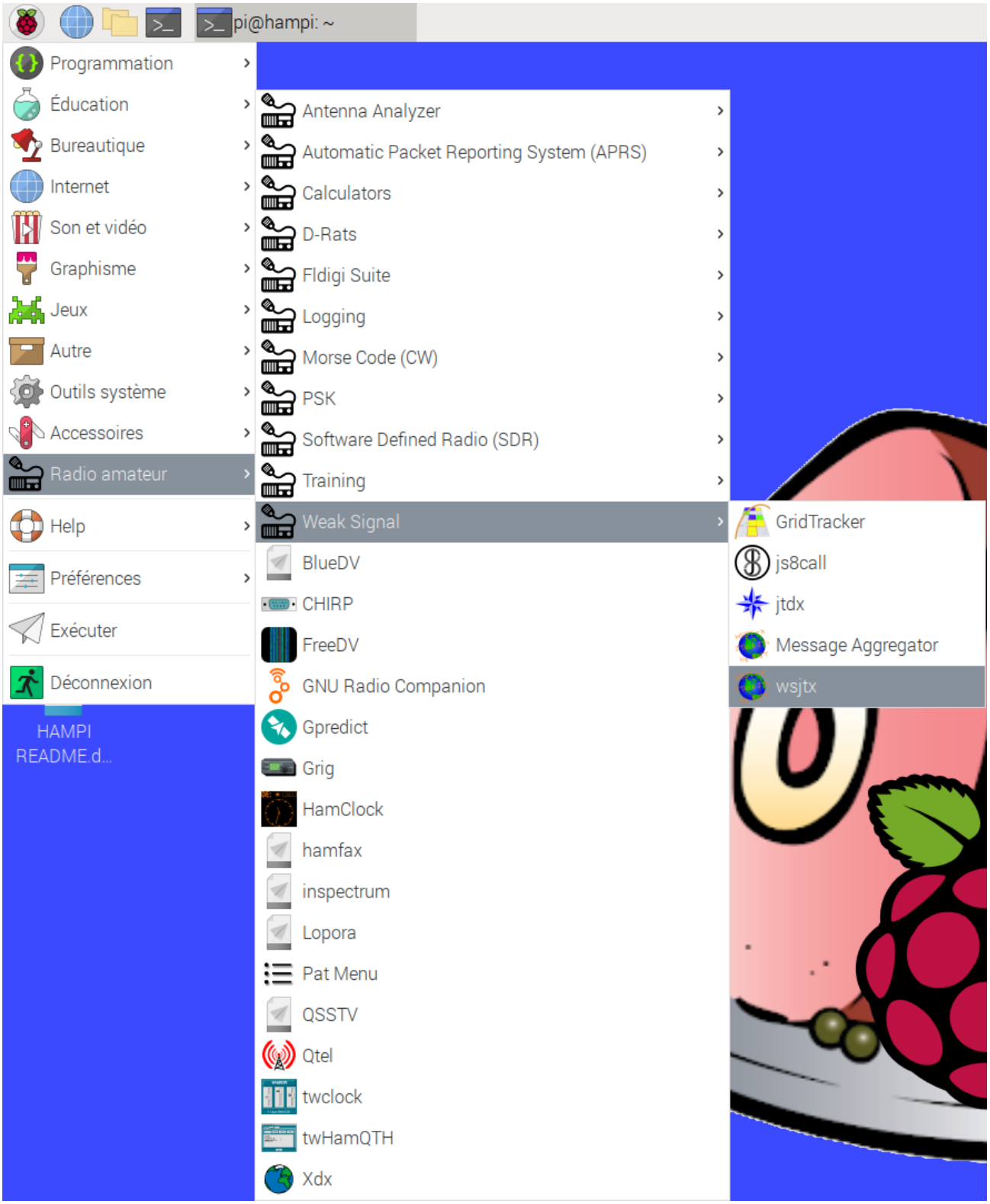
Prerequisites: Discovery of Raspberry Pi part 1

Version du 25/09/2021 V1.3

8.1 HamPi introduction

HamPi 1.x is a Raspbian distribution containing more than 100 preinstalled software created by Dave Slotter, W3DJS. The list of preinstalled software is in the appendix on page 9.

The organization of the menu containing the software is very well done.



8.2 Image disk installation

Start by visiting the official website.

https://github.com/dslotter/HamPi

To find out what's new in the latest version.

But the download link is here

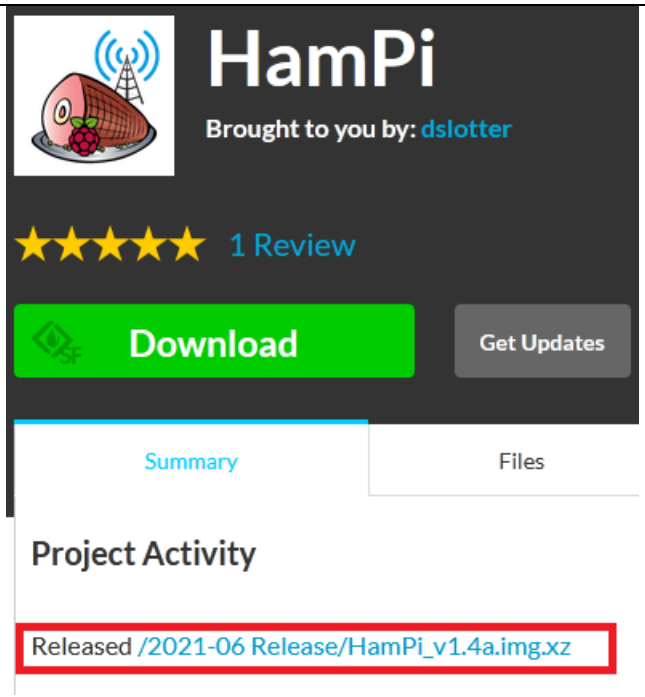
https://sourceforge.net/projects/hampi/

Then download the file

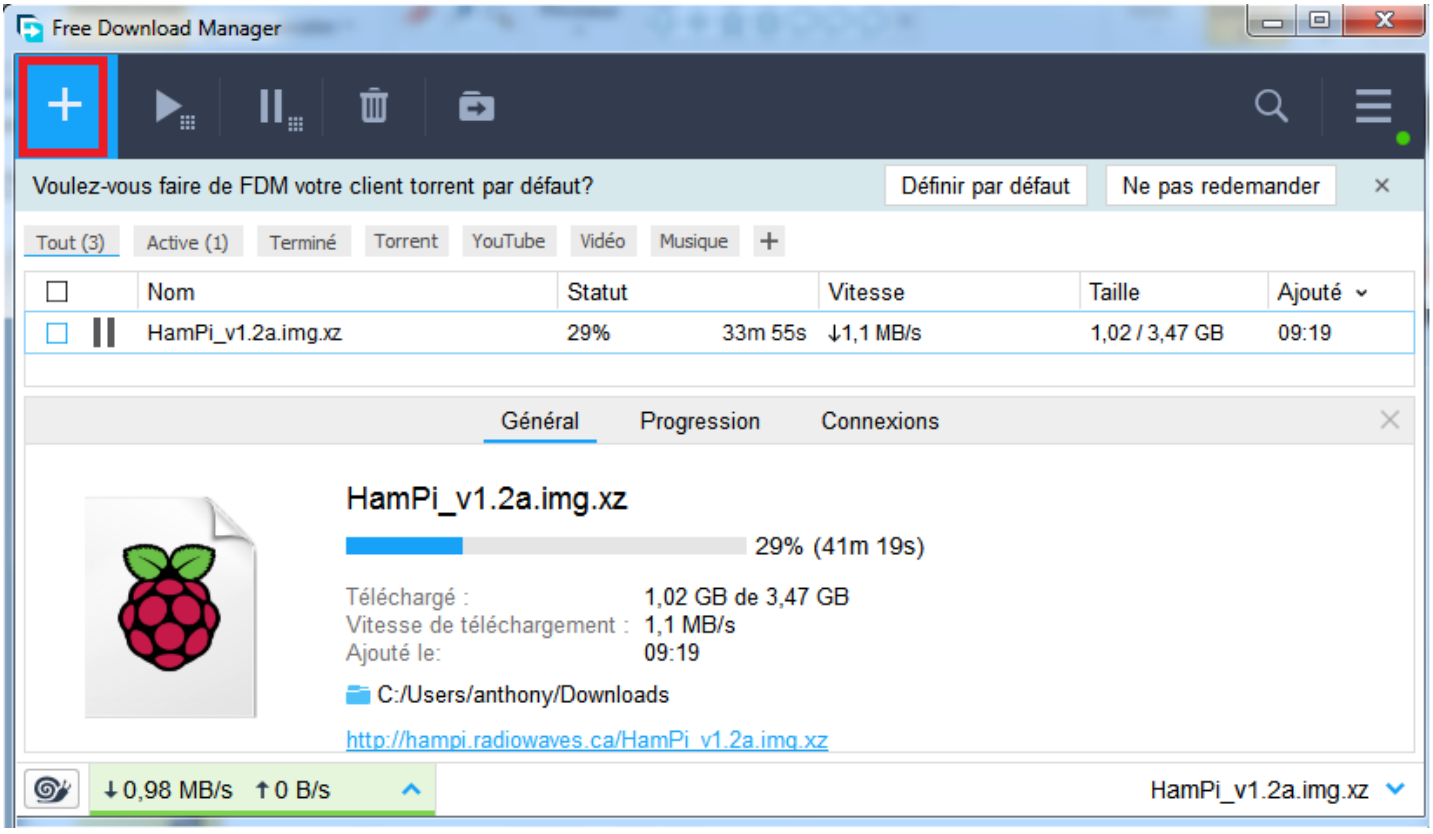
HamPi_vx.x.img.xz

Image size is about 2 or 3 Go
A 32GB mSD card will be required.

The screenshots show the installation of version v1.2a, the process is identical for the more recent versions.

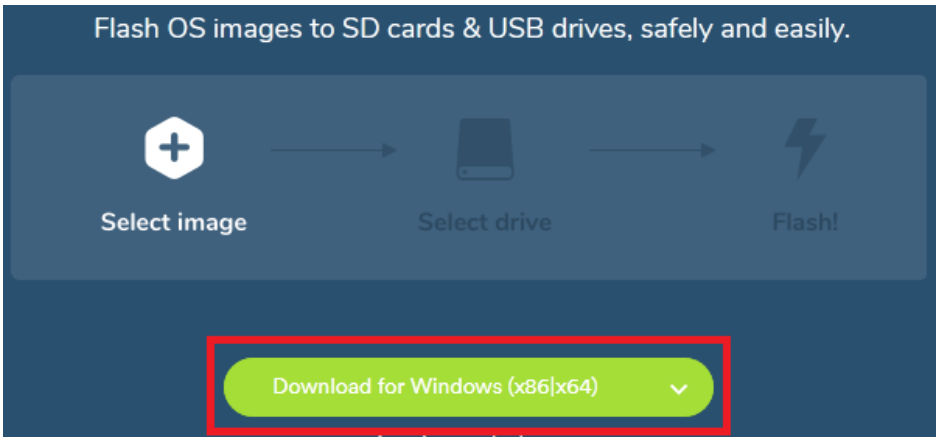


To make downloading easier, use Free Download Manager utility. This will resume the download in case of interruption. After clicking on the + icon, paste the URL of the file to download.



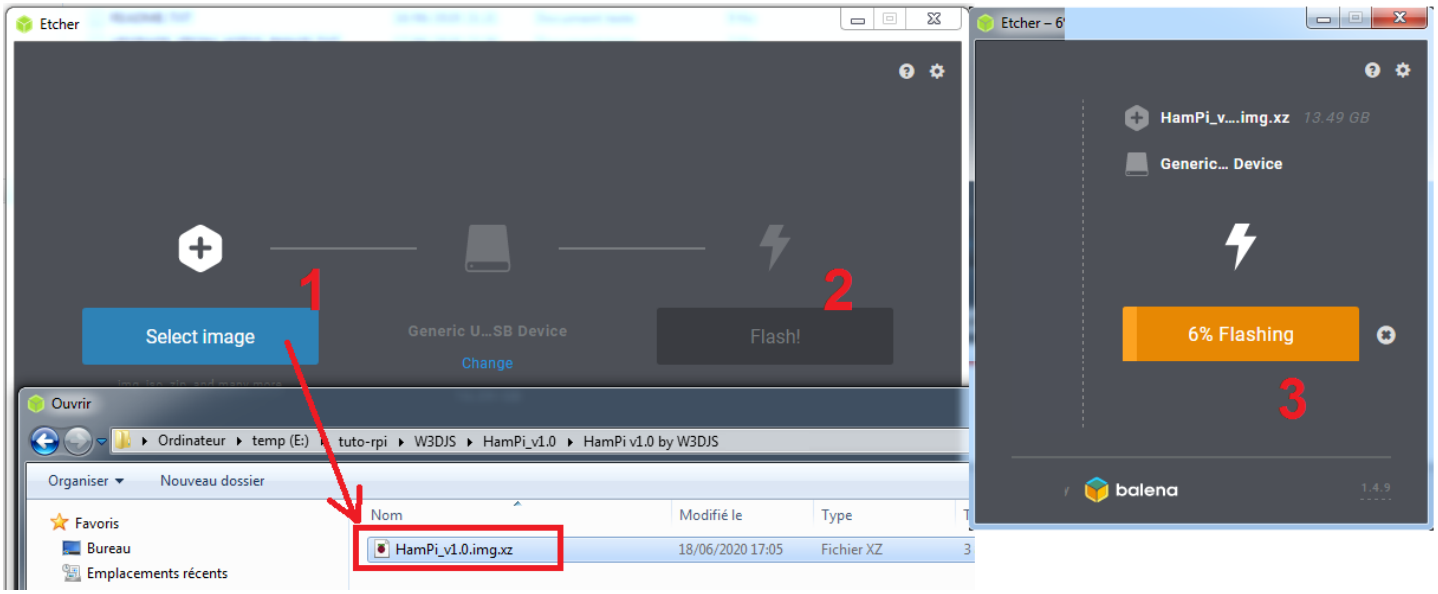
Download and install the utility and check to copy the image to an mSD card.

https://www.balena.io/etcher/



Select the file **`HamPi_vx.x.img.xz`**, the player of the mSD card, then click on Flash.

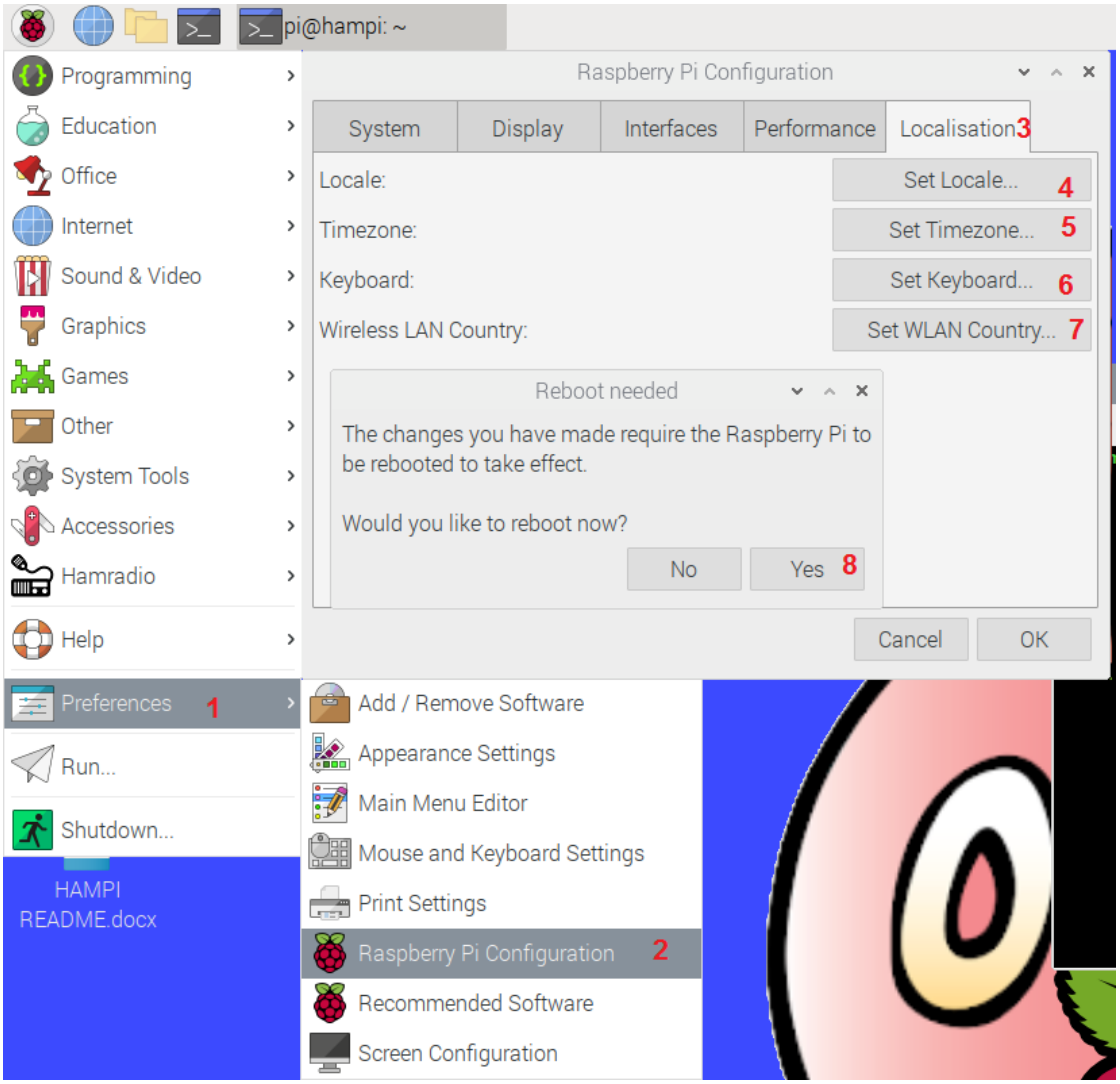
Please note that a **32GB** mSD card is required.



It will take a good quarter of an hour for the writing and verification to be completed.

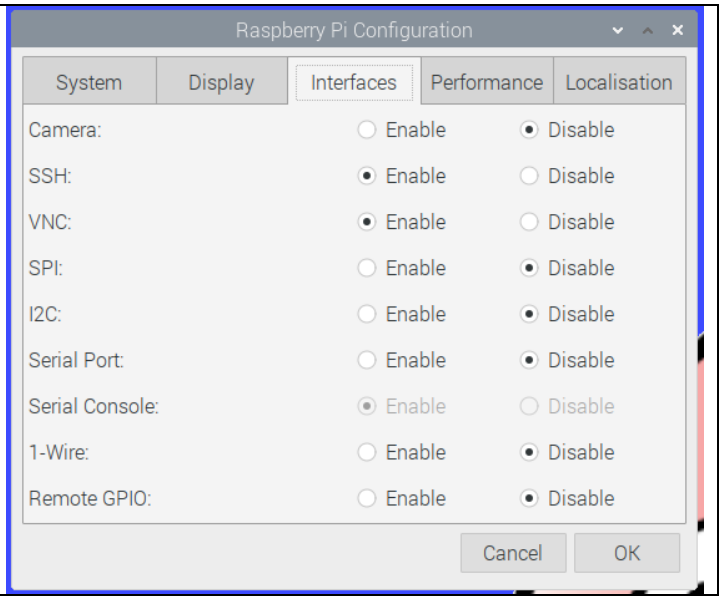
8.3 Start HamPi

As soon as the power is turned on, the country and language must be configured as explained in part 1.



By default the SSh and VNC connection is enabled.

The control of the Raspberry via PuTTY in SSh and the use of VNC viewer are explained in the 1st part of my tutorials.



I first plugged in the **rtl-sdr** key, but the **`rtl_test`** command did not work. I had to do the manipulation explained in part 3 of my tutorials, page 6 with the command line terminal.

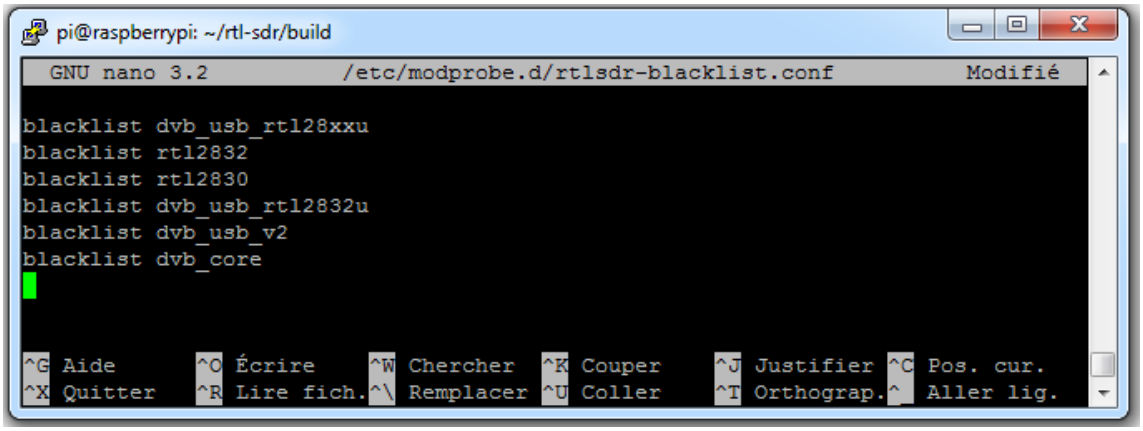
```
sudo nano /etc/modprobe.d/rtlsdr-blacklist.conf
```

```
pi@raspberrypi:~/rtl-sdr/build $ sudo nano /etc/modprobe.d/rtlsdr-blacklist.conf
```

```
blacklist dvb_usb_rtl28xxu
blacklist rtl2832
blacklist rtl2830
blacklist dvb_usb_rtl2832u
blacklist dvb_usb_v2
blacklist dvb_core
```

Copy and paste the above list into the nano editor

Reminder right click to paste the text in the nano editor



Save the file using the Ctrl + o keys, then the enter key to validate the recording.	**Exit the editor using Ctrl + x.**
	

Restart the Raspberry PI.

```
pi@raspberrypi:~/rtl-sdr/build $ sudo reboot
```

```
sudo reboot
```

Verification: Plug the rtl-sdr key into a USB port on the Raspberry Pi.
With LXTerminal or Putty, type the following command line :

```
lsusb
```

Locate the rtl-sdr key : **RTL238 DVB-T**

Cas de la Raspberry Pi 4 :

```
pi@raspberrypi:~ $ lsusb
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 003: ID 0bda:2838 Realtek Semiconductor Corp. RTL2838 DVB-T
Bus 001 Device 002: ID 2109:3431 VIA Labs, Inc. Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

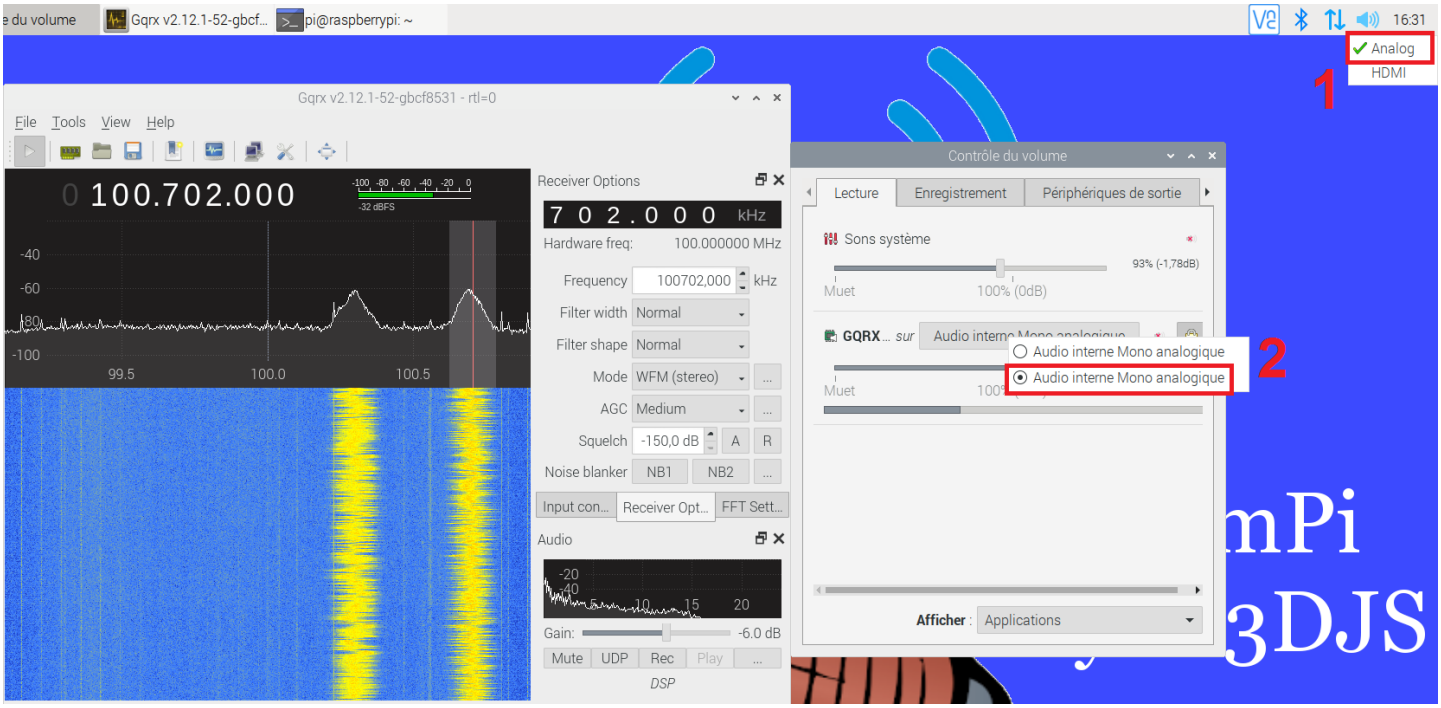
With LXTerminal or Putty, type the following command line : (be careful, underscore)

```
rtl_test
```

The rtl-sdr key should be recognized, if not, disconnect and reconnect the key and restart the test.

```
pi@raspberrypi:~ $ rtl_test
Found 1 device(s):
  0:  Realtek, RTL2838UHIDIR, SN: 00000001

Using device 0: Generic RTL2832U OEM
Found Rafael Micro R820T tuner
Supported gain values (29): 0.0 0.9 1.4 2.7 3.7 7.7 8.7 12.5 14.4 15.7 16.6 19.7
 20.7 22.9 25.4 28.0 29.7 32.8 33.8 36.4 37.2 38.6 40.2 42.1 43.4 43.9 44.5 48.0
 49.6
[R82XX] PLL not locked!
Sampling at 2048000 S/s.

Info: This tool will continuously read from the device, and report if
samples get lost. If you observe no further output, everything is fine.

Reading samples in async mode...
^CSignal caught, exiting!
ctrl+c

User cancel, exiting...
Samples per million lost (minimum): 0
pi@raspberrypi:~ $ ^C
```

With the GQRX software, I tested the correct reception of the 88-108Mhz FM band.



But the sound did not work on the 3.5 jack output.

1- Then right-click on the speaker icon and check **Analog.**
2- With the "audio pulse volume" control utility, right-click on "internal mono analog audio" and select the 2nd line.

I don't have time to test all the software. Each OM will have to carry out its own tests.

There is a help page on GitHub (registration required) where you can ask questions and a group of Raspberry Pi users around the radio.

https://github.com/dslotter/HamPi/issues

https://groups.io/g/RaspberryPi-4-HamRadio/

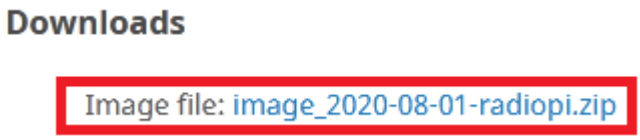
8.4 RadioPi Introduction

RadioPi is a Raspbian distribution created by BG6LH, BG1TPT, BI1EIH containing 15 pre-installed software packages. The advantage of this image disk is to propose a startup wiki (user manual) which is not the case for HamPi. On the other hand, this image has much less preinstalled software than HamPi. Hopefully RadioPi will benefit from the same number of software packages as HamPI in the future.

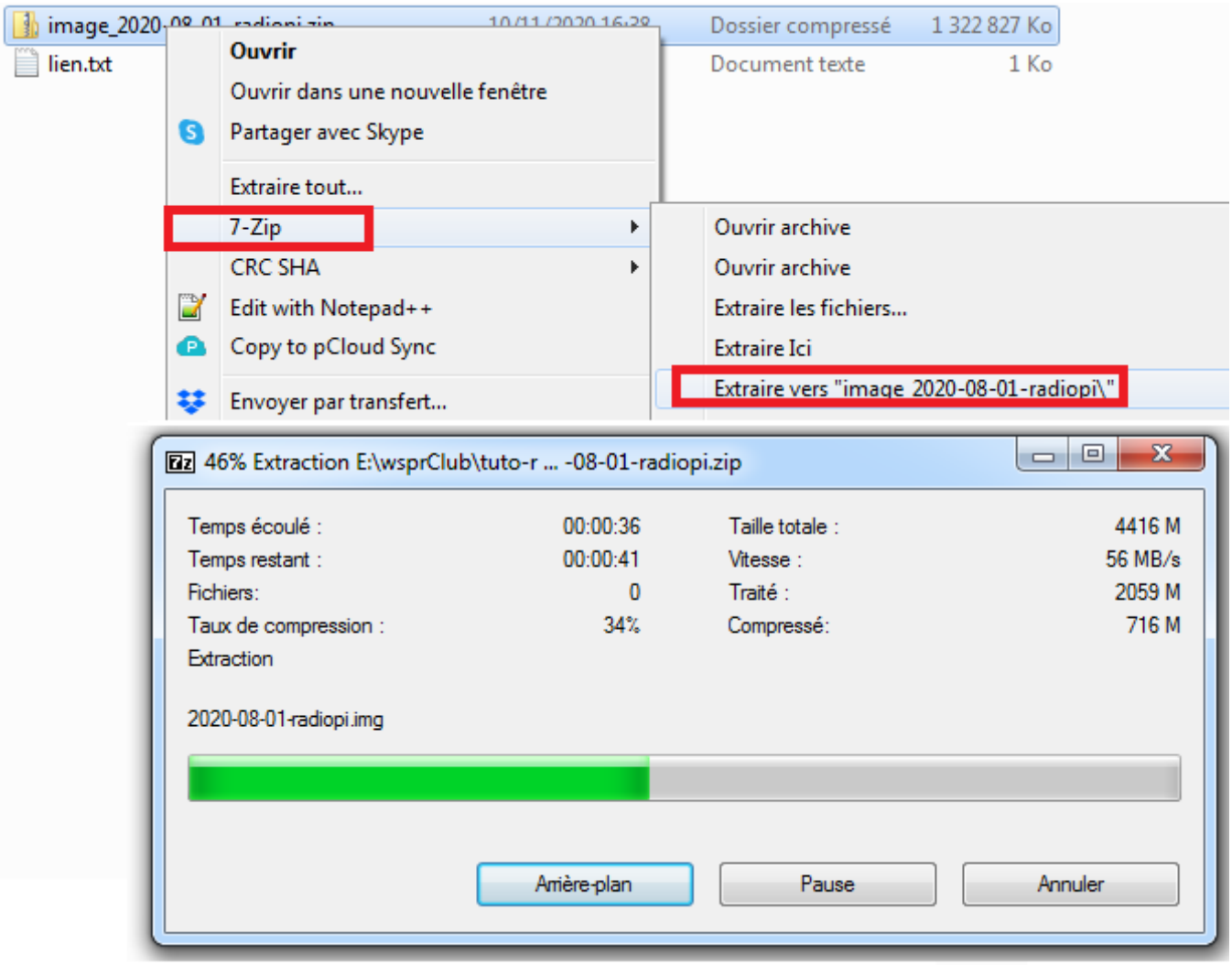
- **WSJT-X**, 2.2.2, The extreme weak-signal communication software
- **JTDX**, 2.1.0-RC150, More features than WSJT-X
- **TQSL**, 2.4.3, The ARRL LoTW QSL client
- **Xlog**, 2.0.14, A light weight logging program
- **Fldigi**, 4.1.01, Amateur Radio in Digital Modes
- **CHIRP**, daily20190104, Programming amateur radios, supporting a large number of manufacturers and models, such as Yaesu, Icom, Kenwood, Baofeng, TYT and so on.
- **QSSTV**, 9.2.6, Receiving and transmitting SSTV and HAMDRM(DSSTV)
- **GNU Radio Companion**, 3.7.13.4, GNU Radio is a free & open-source software development toolkit that provides signal processing blocks to implement software radios.
- **GQRX**, 2.11.5, Gqrx is an open source software defined radio receiver (SDR) powered by the GNU Radio and the Qt graphical toolkit.
- **CubicSDR**, 0.2.5, Cross-platform SDR application
- **Direwolf**, 1.4, A software "soundcard" modem/TNC and APRS en/decoder
- **Gpredict**, 2.3-33-gca42d22-1, A real-time satellite tracking and orbit prediction application
- **VNC Server**, 6.7.2, To remote control your rig remotely by VNC
- **PulseAudio Preferences**, 1.1, For simultaneous line-output, and transmit monitoring.

8.5 Image disk installation

From the https://radiopi.club/ website, download the back version of the image.



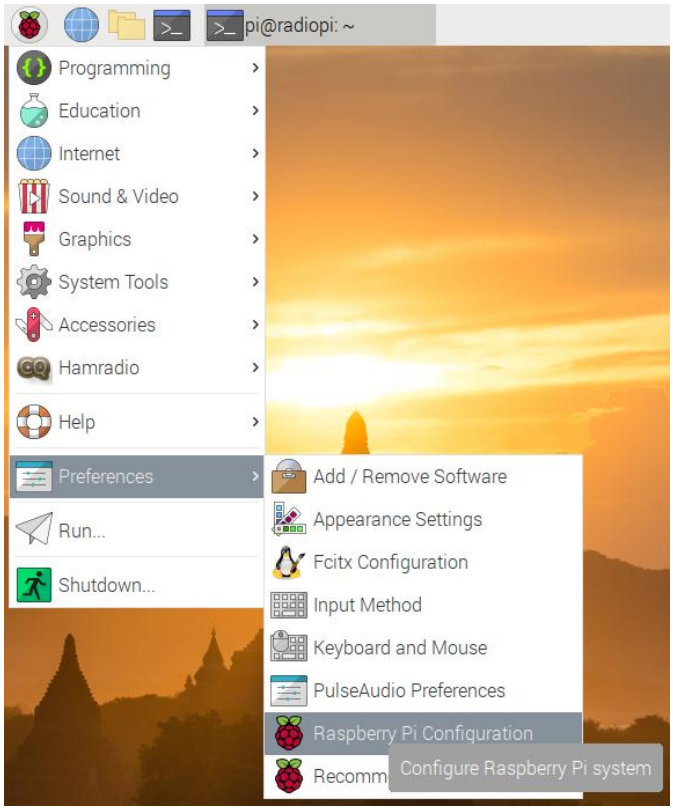
Then decompress the archive with the 7zip utility.

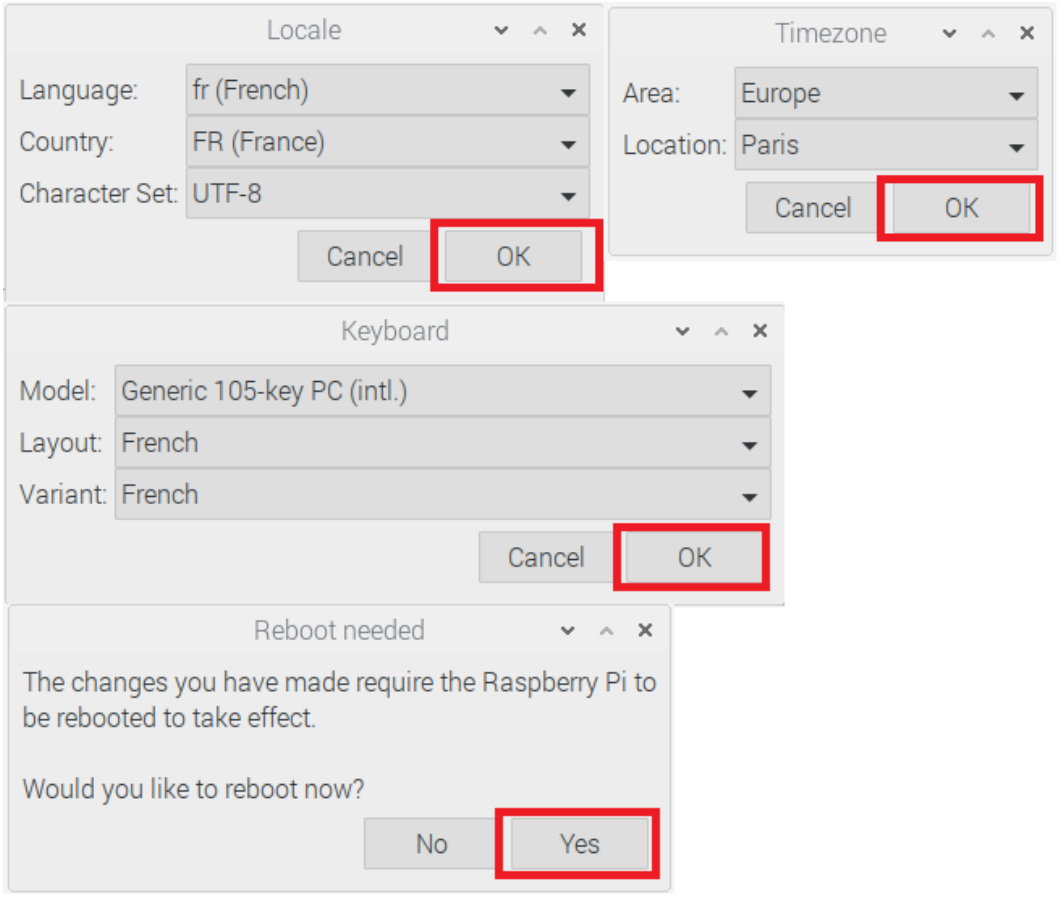


To copy the image to an mSD card, use the same procedure as for HamPi (page 4 of the same document) with the Etcher utility.

8.6 Prise en main de radioPi

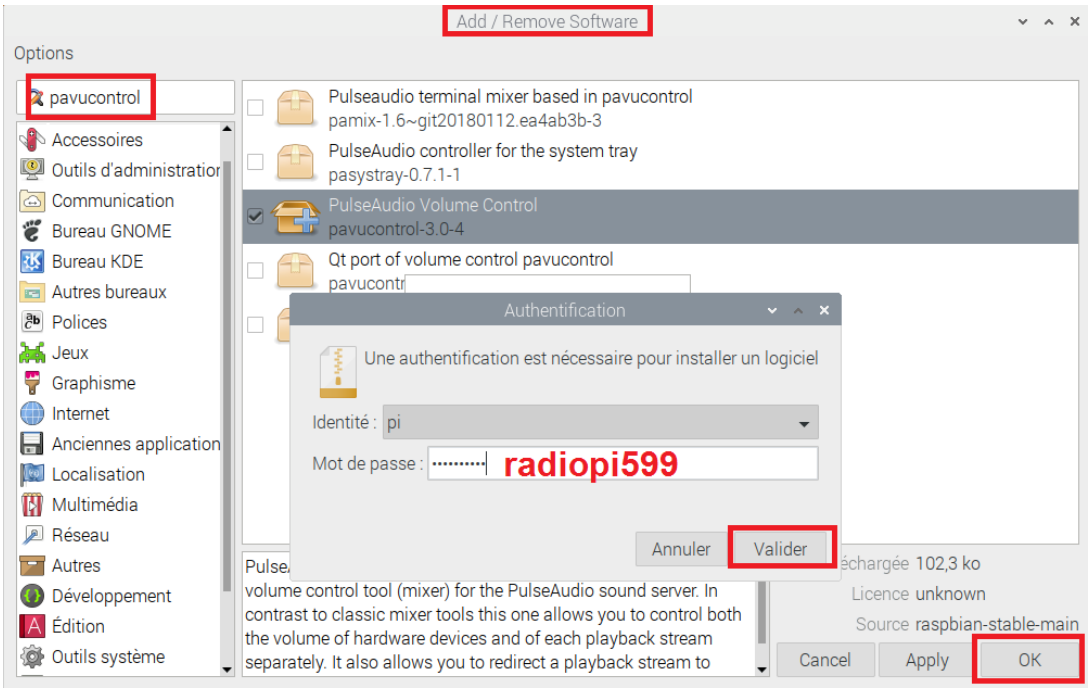
Start by configuring the country and keyboard by going to the "RaspberryPi configuration" menu.



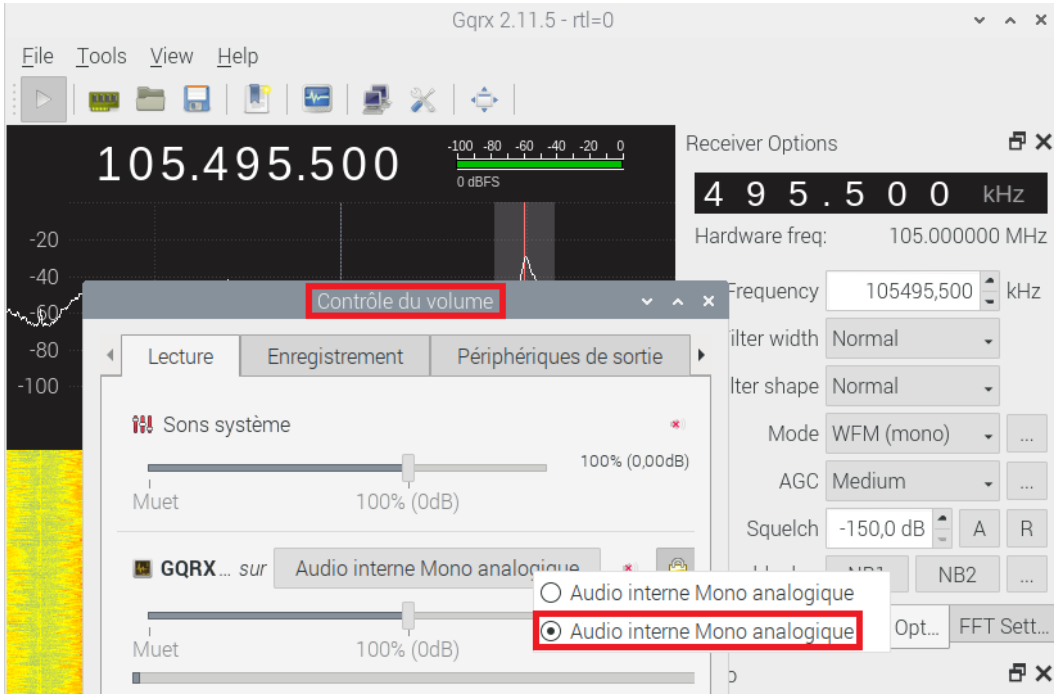


Perform the following operations to remove the fcitx utility that interferes with the use of the French keyboard.

```
sudo apt purge fcitx
sudo apt autoremove
sudo apt update
sudo reboot
```



GQRX with the rtl-sdr key works the first time.



Appendix

HamPi 1.2 – The Raspberry Pi Ham Radio Image by W3DJS

General Ham Radio Applications

HamLib - Ham Radio Control Libraries
grig - graphical user interface to the Ham Radio Control Libraries
CHIRP - Radio Programming Software
APRS Message App for JS8Call - GUI to send APRS messages via JS8Call
QTel - EchoLink client
QSSTV - Slow Scan TV (e.g. "Fax")
Gpredict - Satellite prediction
FreeDV - Free digital voice vocoder
BlueDV - Client for D-Star and DMR
WsprryPi - WSPR software
ADS-B Flight Tracking Software
Pi3/4 Stats Monitor - by W1HKJ
VOACAP - HF propagation prediction
GPS Support
wxtoimg - NOAA weather imaging software
twHamQTH - an online callsign look up program
twclock - a world clock and automatic ID for amateur radio operators
acfax - Receive faxes using your radio and sound card
colrconv - convers client with sound and ncurses color support
D-Rats 0.3.9 (by new maintainer Maurizio Andreotti) - A communication tool for D-STAR
fbb - Packet radio mailbox and utilities
gcb - Utility to calculate long and short path to a location
glfer - Spectrogram display and QRSS keyer
Xdx is a DX-cluster client
DXSpider - DX Cluster Server
fccexam - Study tool for USA FCC commercial radio license exams.
gnuais / gnuaisgui - GNU Automatic Identification System receiver
hamexam - Study guide for USA FCC amateur radio (ham radio) license examinations.
hamfax - Qt based shortwave fax
inspectrum - tool for visualising captured radio signals
predict-gsat - Graphical Predict client
splat - analyze point-to-point terrestrial RF communication links
wwl - Calculates distance and azimuth between two Maidenhead locators
AX.25 – Packet Radio drivers for ax.25 protocol
linpac - terminal for packet radio with mail client
PyBOMBS - GNU Radio install management system
AMBEServer – AMBE vocoder chip support
HamClock – GUI HamClock by WBOEW
Adifmerg – command-line ADIF conversion utility

Lopora – QRSS Beacon Reception
Universal Ham Radio Remote (UHRR) – UHRR provides remote radio operation
RpiTx -- Turns Raspberry Pi into low power transmitter
ACARS Decoder – for tracking aircraft transponders
CygnusRFI – RFI analysis tool for ground stations and radio telescopes)
Update Scripts -- to update Fldigi suite and WSJT-X (so one doesn't have to wait for new HamPi release.)

Antenna Ham Radio Applications

antennavis - Antenna Visualization Software
gsmc - A GTK Smith Chart Calculator for RF impedance matching
nec2c - Translation of the NEC2 FORTRAN source code to the C language
xnecview - NEC structure and gain pattern viewer
yagiuda - software to analyse performance of Yagi-Uda antennas

Digital Mode Ham Radio Applications

WSJT-X - Weak Signal (FT8, FT4, etc.) by W1JT
GridTracker - Graphical mapping companion program for WSJT-X or JTDX
JTDX - Alternate client for Weak Signal (FT8, FT4, etc.)
JS8Call - Messaging built on top of FT8 protocol by KN4CRD
JS8CallTools - Get Grid coordinates using GPS
(FLDigi is in its own section below.)
gnss-sdr - GLONASS satellite system Software Defined Receiver
linpsk - amateur radio PSK31/RTTY program via soundcard
multimon - multimon - program to decode radio transmissions
multimon-ng - digital radio transmission decoder
psk31lx - a terminal based ncurses program for psk31
twpsk - a psk program

Software Defined Radio

CubicSDR - Software Defined Radio receiver
cutesdr - Simple demodulation and spectrum display program
GQRX - Software defined radio receiver
LeanSDR – Lightweight, portable software defined radio
SDRAngel - SDR player
lysdr - Simple software-defined radio
SoapyAudio - Soapy SDR plugin for Audio devices
SoapyHackRF - SoapySDR HackRF module
SoapyMultiSDR - Multi-device support module for SoapySDR
SoapyNetSDR - Soapy SDR module for NetSDR protocol

SoapyRemote - Use any Soapy SDR remotely
SoapyRTLSDR - Soapy SDR module for RTL SDR USB dongle
SoapySDR - Vendor and platform neutral SDR support library
SoapySDRPlay - Soapy SDR module for SDRPlay
Support for RTL-SDR
Support for SDRPlay SDR
Support for HackRF SDR
Support for AirSpy and AirSpy HF
SoapySDRAirSpy- Soapy SDR module for AirSpy SDR
SoapySDRFUNcube Dongle Pro+- Soapy SDR module for FUNCube Dongle Pro+
SoapySDRPlutoSDR- Soapy SDR module for Pluto SDR
SoapySDROsmoSDR- Soapy SDR module for Osmo SDR
SoapySDRRedPitaya- Soapy SDR module for Red Pitaya SDR
SoapyUHD- Soapy SDR module for Ettus ResearchUHD SDR
SoapySDRVOLKConverters - Support for VOLK-based type converters

APRS Applications

Xastir - APRS GUI client / Digipeater / Igate
YAAC - Yet Another APRS Client
DireWolf - Software "soundcard" AX.25 packet modem/TNC and APRS encoder/decoder
aprsdigi - digipeater for APRS
aprx - APRS Digipeater and iGate
soundmodem - Sound Card Amateur Packet Radio Modems

FLDigi Application Suite from W1HKJ

flrig - Rig Control program which interfaces with fldigi
fldigi - Digital Modes Communications
flaa - RigExpert Antenna Analyzer Control Program
flamp - File transmissions via Amateur Multicast Protocol
flarq - ARQ data transfer utility for fldigi
flcluster - Telnet client to remote DX Cluster Servers
fllog - Logbook application which can use same data file as fldigi
flmsg - Editor for ICS 213 Forms
flnet - Net Control Assistant for Net Activities (Check-In Application)
flpost - NBEMs post office
flwrap - File encapsulation and compression for transmission over amateur radio
flwkey - Winkeyer (or clone) control program for K1EL Winkeyer series

Logging Applications

TrustedQSL - LotW client
CQRlog - Ham Radio Logging Application
PyQSO - Logging software (written in Python)
klog - The Ham Radio Logging program
tlf - console based ham radio contest logger
tucnak2 - VHF/UHF/SHF Hamradio contest log version 2
twlog - basic logging program for ham radio
upload_adif_log – Upload only new log entries to LotW, eQSL.cc and ClubLog
wsjtx_to_n3fjp - Logging adapter to allow WSJT-X to log to N3FJP
xlog - GTK+ Logging program for Hamradio Operators

WinLink Applications

Pat WinLink - WinLink for Raspberry Pi (and other platforms)
ARDOP support for Pat WinLink
ARDOP-GUI - Provides graphical representation of ARDOP connections
Find ARDOP - Retrieves local ARDOP sources by KM4ACK
Pat Menu 2 – Menu for Pat by KM4ACK
PMON - a PACTOR® Monitoring Utility for Linux

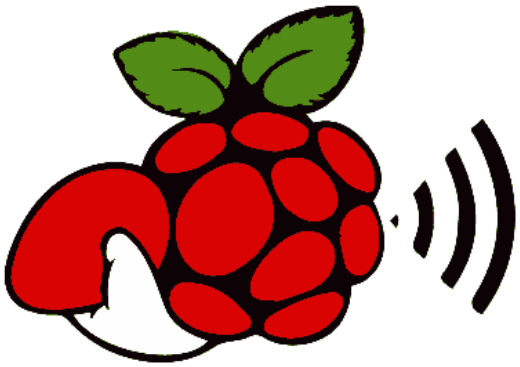
Morse Code Applications

aldo - Morse code training program
cw - sound characters as Morse code on the soundcard or console speaker
cwcp - Text based Morse tutor program
xcwcp - Graphical Morse tutor program
cwdaemon - morse daemon for the serial or parallel port
ebook2cw - convert ebooks to Morse MP3s/OGGs
ebook2cwgui - GUI for ebook2cw
morse - training program about morse-code for aspiring radio hams
morse2ascii - tool for decoding the morse codes from a PCM WAV file
morsegen - convert file to ASCII morse code
qrq - High speed Morse telegraphy trainer
xdemorse - decode Morse signals to text

Raspberry Pi for HAM Radio
Part 9

Summary:

Part 9: Radio test with Rpitx



Prerequisites: Part 4, Using the Raspbian Lite operating system without GUI.

Version du 10/10/2020 V1.0

9.1 Introduction

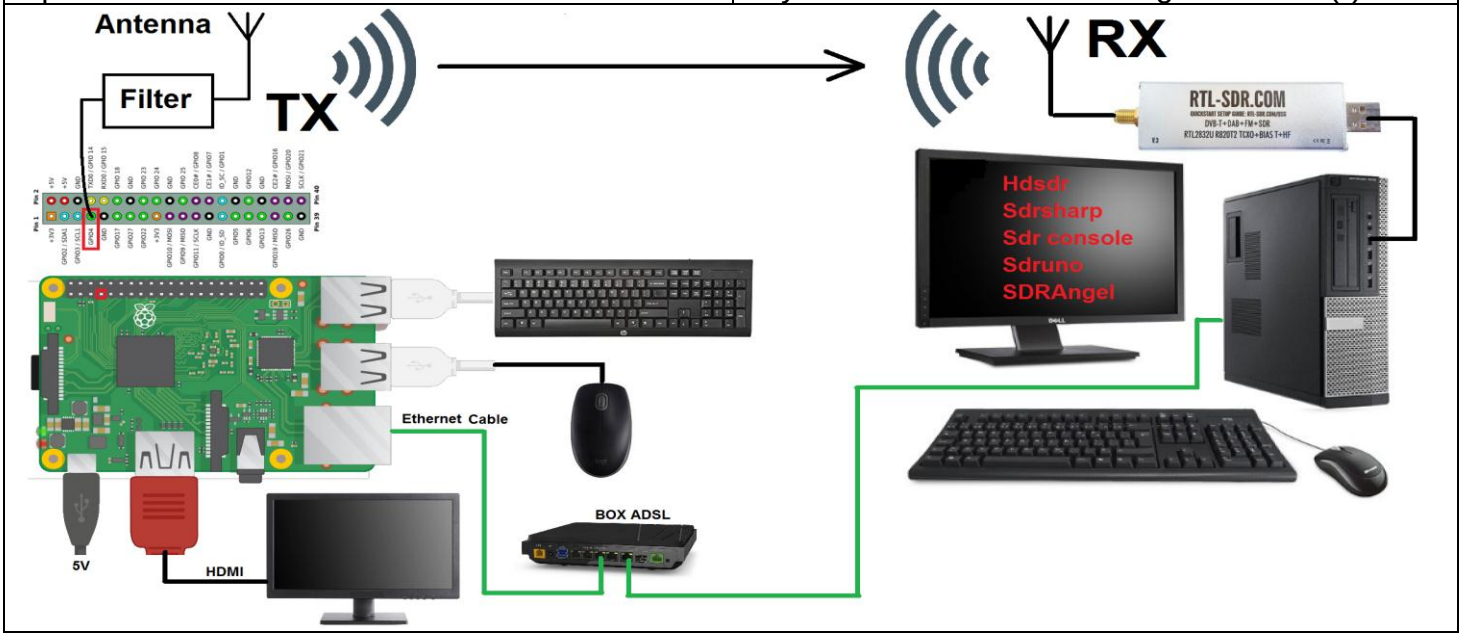
In this tutorial, the aim is to implement a test bench to receive the radio frequency signals emitted by the Raspberry PI. Indeed, it has a specific output (GPIO 4) that can be used directly on an antenna. The Raspberry Pi can manage frequencies from 5 KHz to 1500 MHz.

The software suite managing the Raspberry PI internal RF transmitter called rpitx, is designed by Evariste Courjaud F5OEO.

Thus, the Raspberry Pi will be able to directly transmit a carrier, and various signals such as :
- A signal FM, SSB, code OOK ;
- An SSTV image, an OPERA signal, FT8;
- Etc...

However, to receive signals from the Raspberry PI, you will also need to set up a listening station. This one will be composed of a PC and an rtl-sdr key, without forgetting the installation of a reception software (a YouTube video is proposed for each SDR reception program).

Raspberry Pi transmitter with the software suite Rpitx.	The receiving part will be a PC with the rtl-sdr key and one of the 5 receiving softwares (*).



(*) It is possible to use a 2nd Raspberry PI with the GQRX software as explained in part 3: Standard use of the RTL-SDR key.

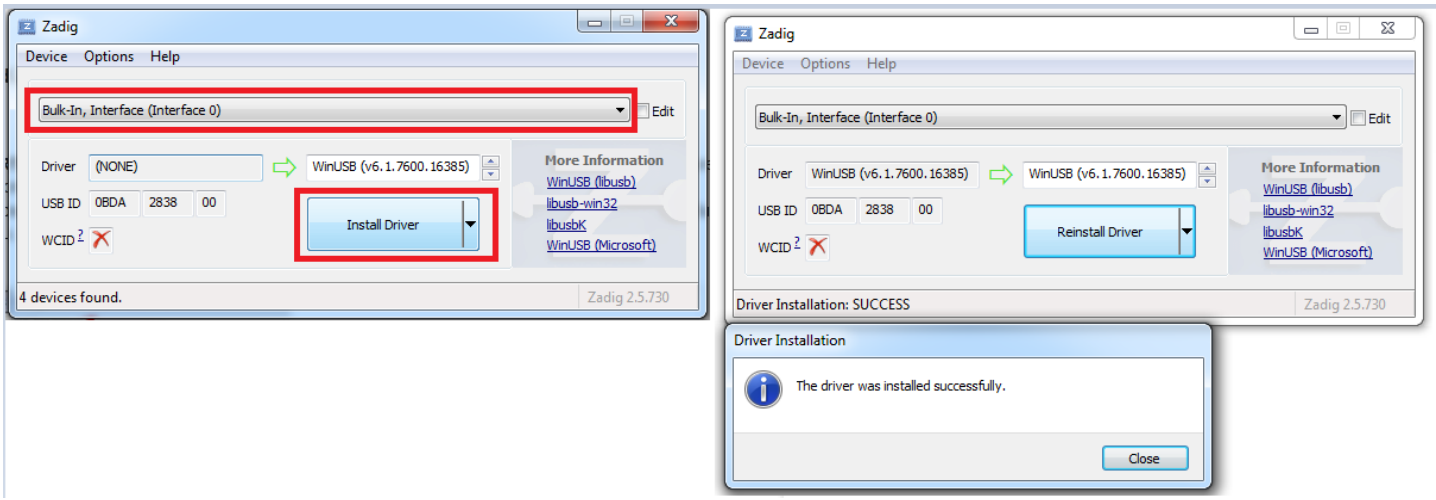
9.2 Rx Installation software on PC

There are many PC reception software programs that use the rtl-sdr key. I can't list them all, but I'll go back to the installation of 5 of them. Most OMs that regularly use SDR reception have their preferred software. But those who have never installed and used SDR software can at least make a choice. The description above explains the installation of the software under Windows. I don't detail the in-depth use of the 5 software.

9.2.1 Zadig driver

The common point of the 5 softwares is the specific driver for the rtl-sdr key.
Go to https://zadig.akeo.ie/, then download the **`Zadig 2.5.exe`** file (4.9 MB)

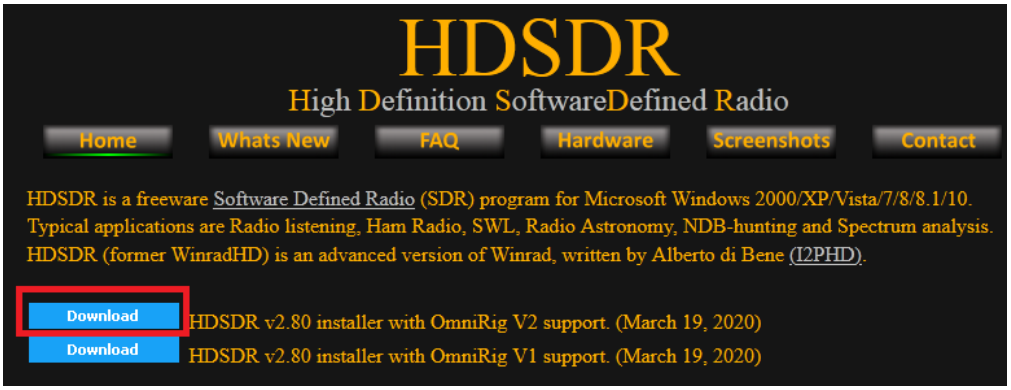
Connect the rtl-sdr key to a USB port on the PC. Make sure to always use the same **USB** port for the rtl-sdr key. Run the file zadig-2.5.exe , select Bulk-in, (Interface 0), then Install Driver.



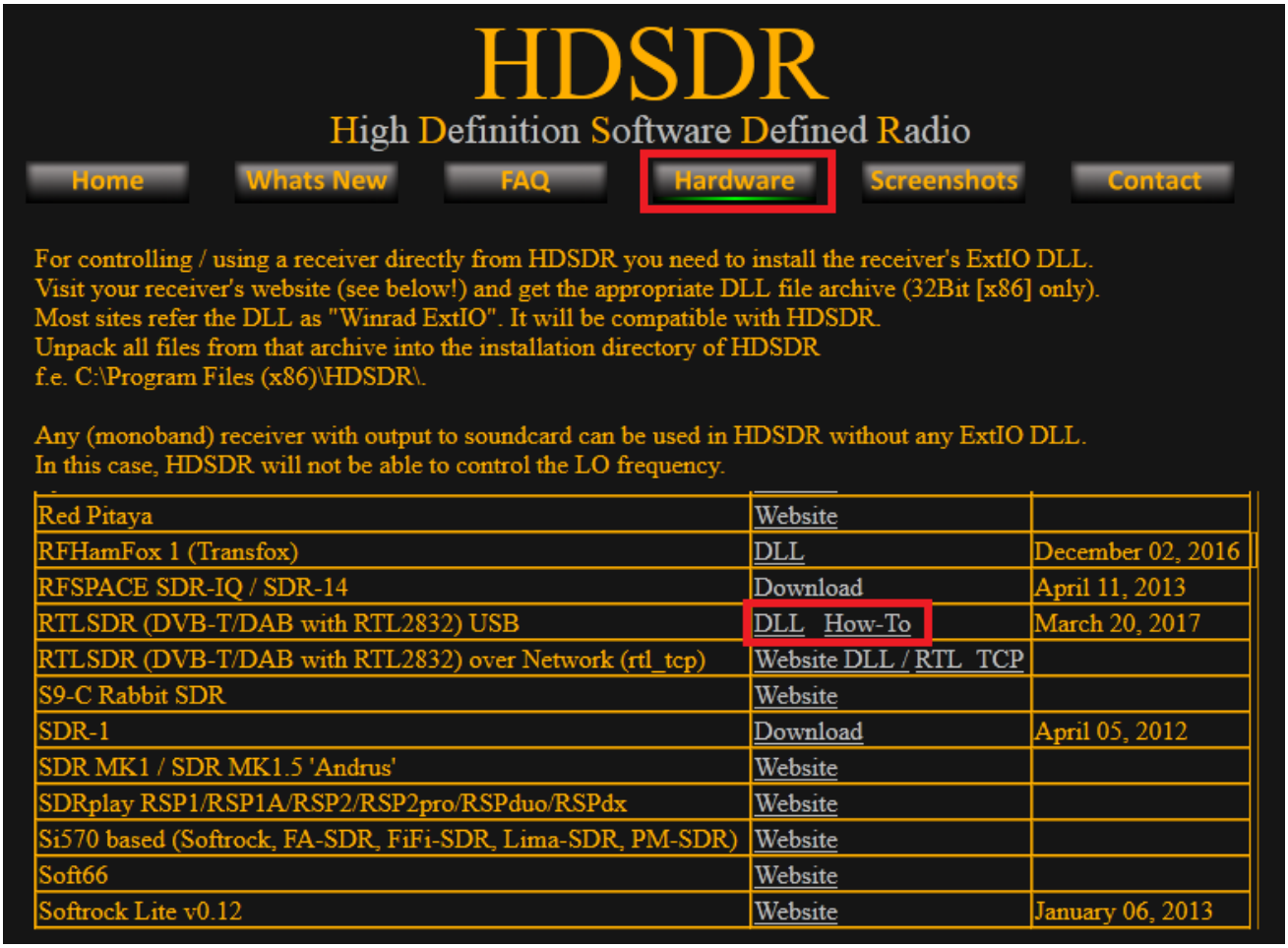
9.2.2 Hdsdr

Hdsdr is certainly the best-known software. It is very easy to install. An installation guide is available (How-to).

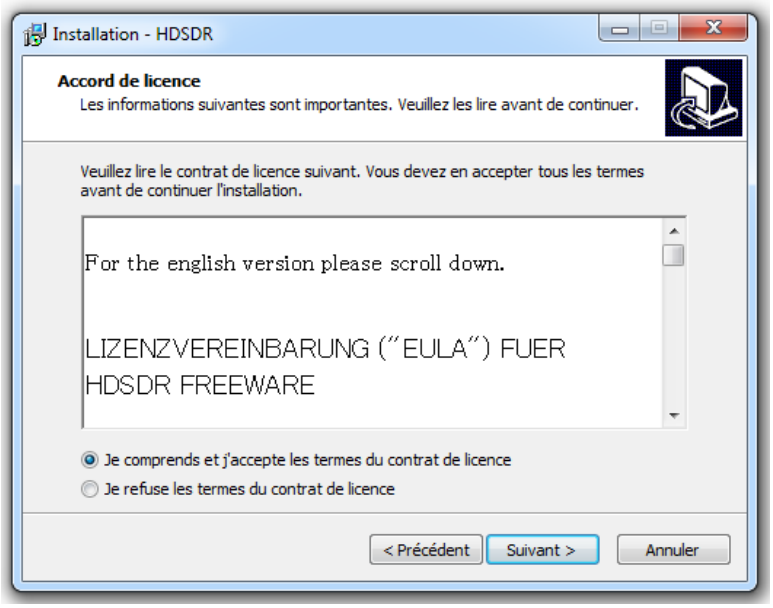
Go to http://www.hdsdr.de/, then download the HDSDR_install.exe file.



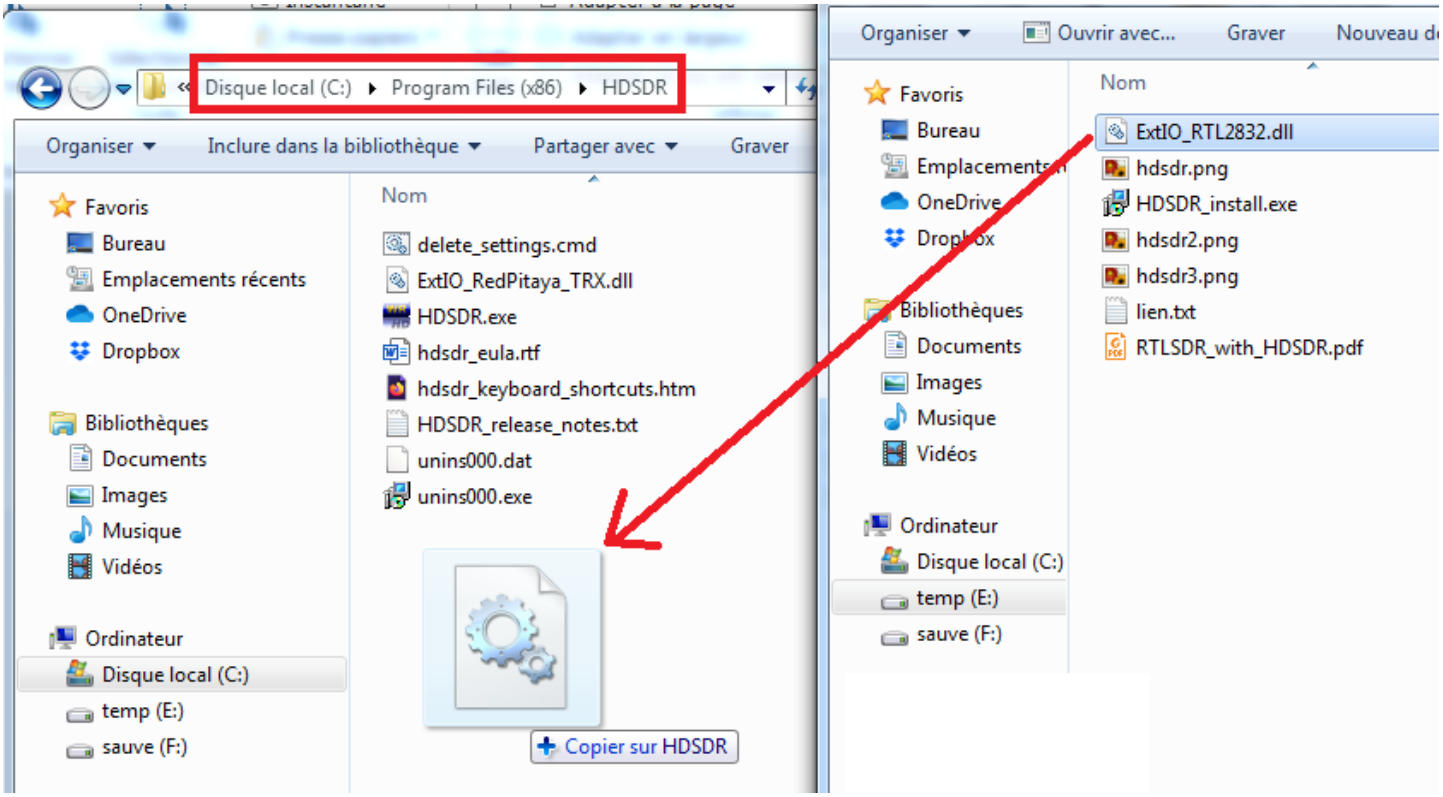
Then download the file `ExtIO_RTL2832.dll`.
http://hdsdr.de/download/ExtIO/ExtIO_RTL2832.dll



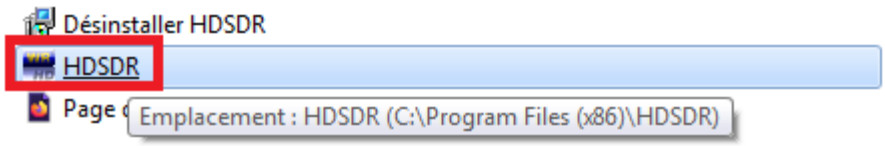
Then install HDSDR (**`HDSDR_install.exe`**).



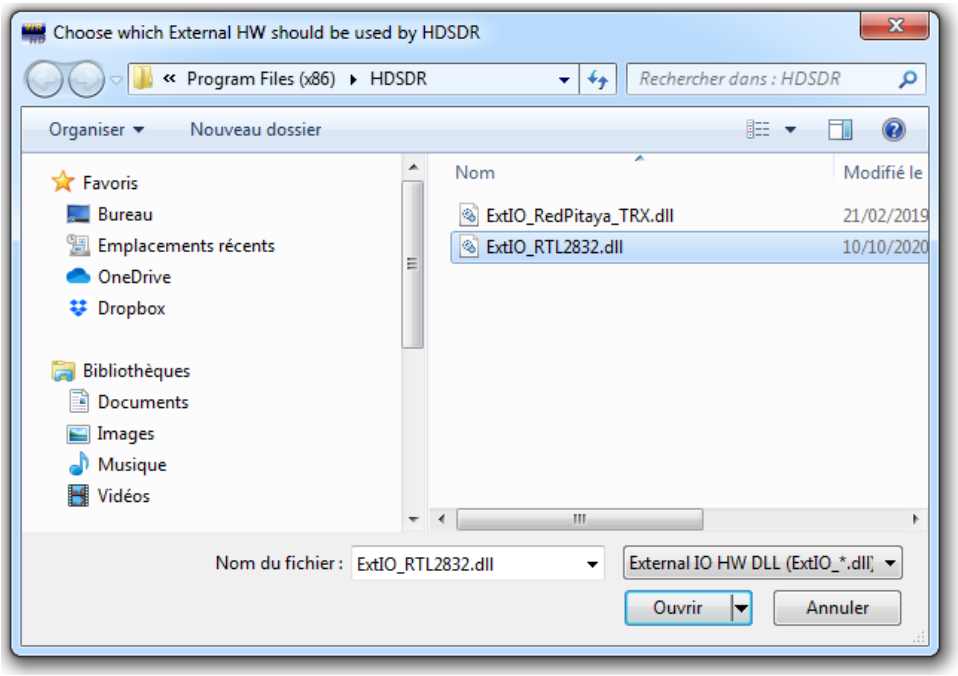
Copy the `ExtIO_RTL2832.dll` file to the HDSDR installation directory.



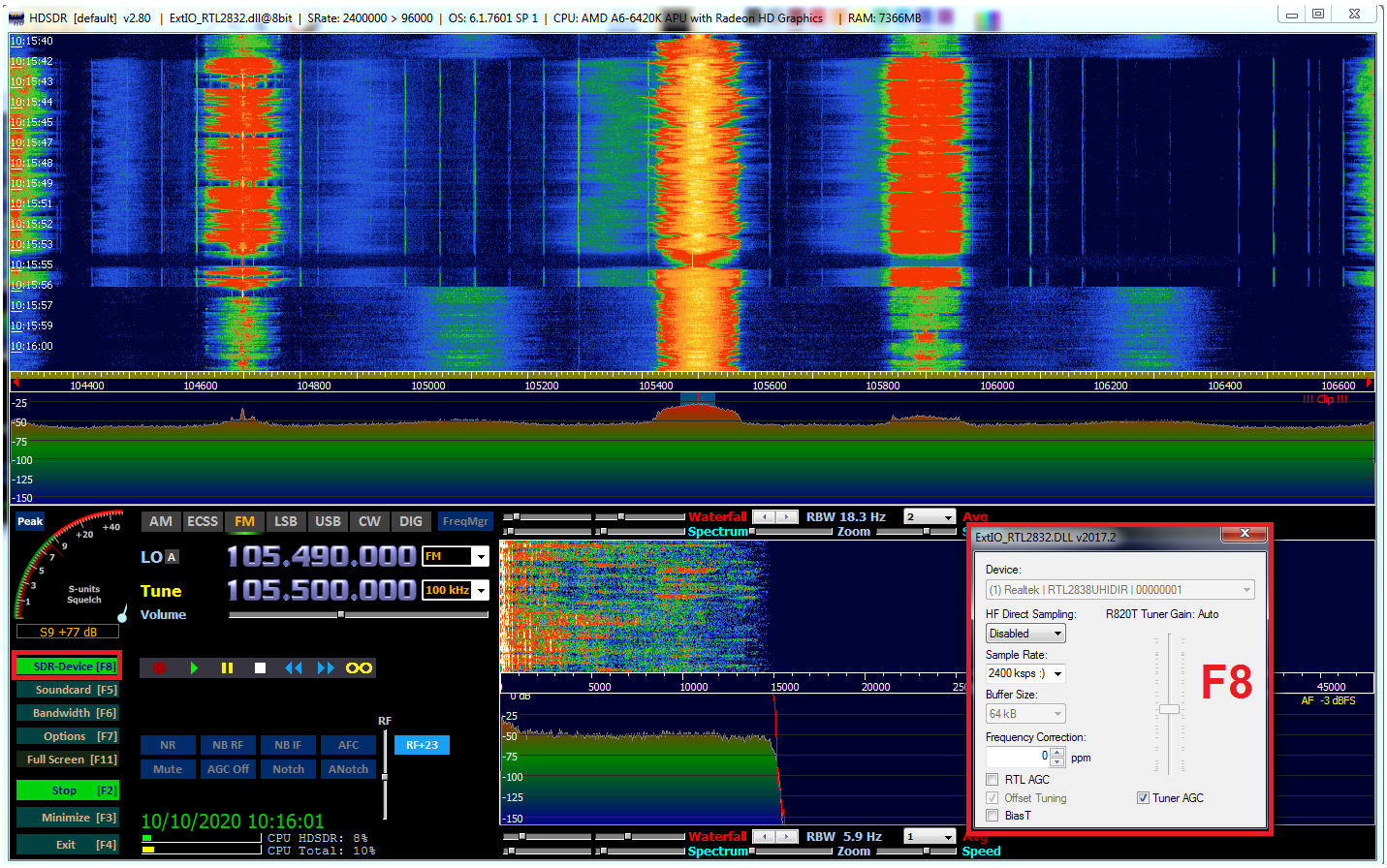
Then Run the HDSDR program.



Select the `ExtIO_RTL2832.dll` file

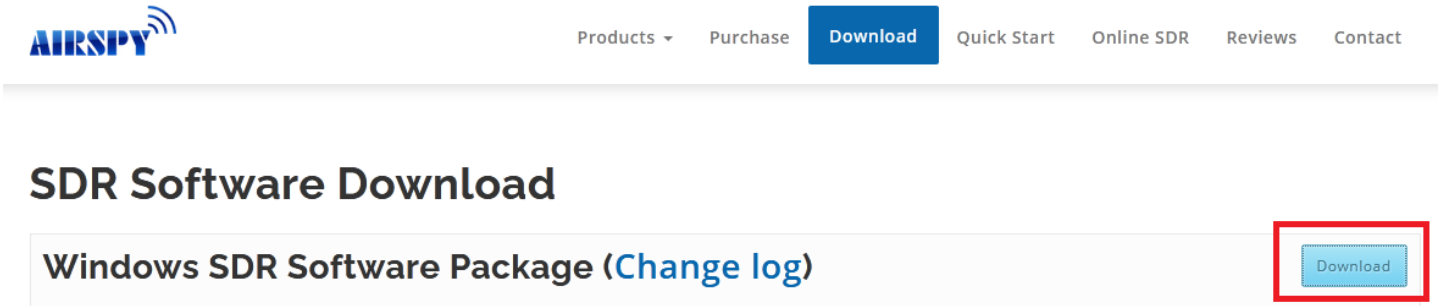


Test reception by listening to the [FM band](#) (88-108Mhz)

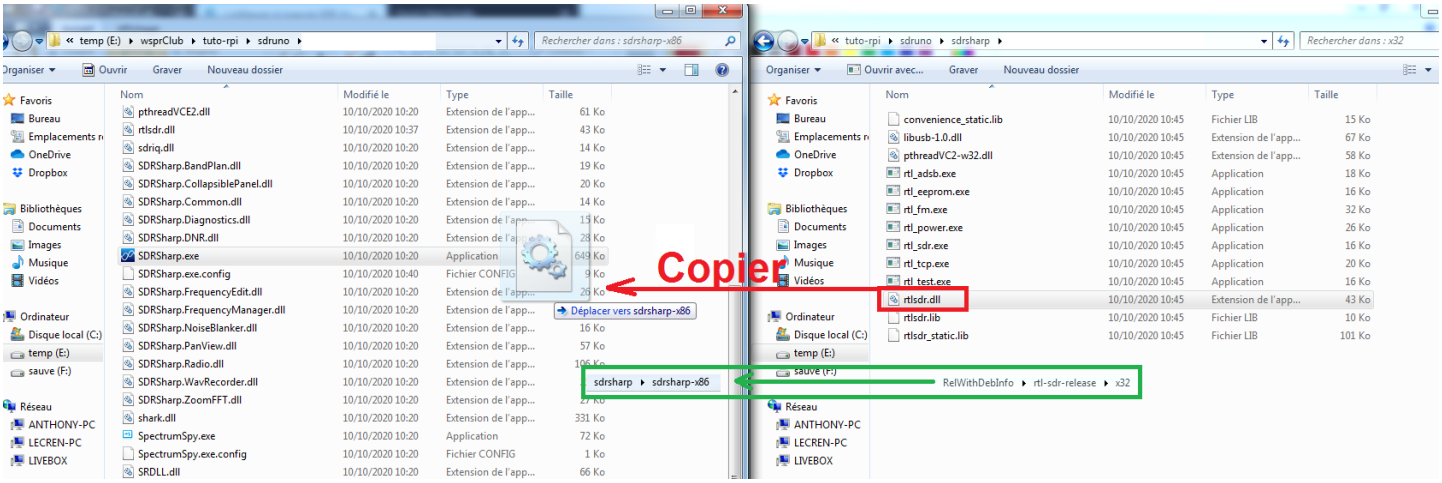


9.2.3 Sdrsharp

Go to https://airspy.com/download/, then download the file `sdrsharp-x86.zip`
Then unzip the zip. There is no installation program like hdsdr.

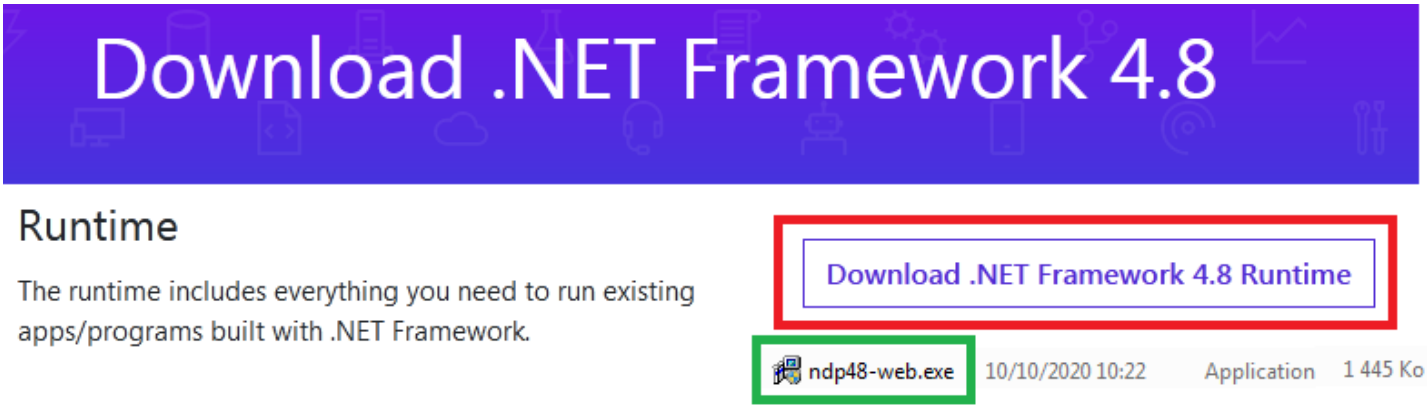


Download the file `RelWithDebInfo.zip`
http://osmocom.org/attachments/download/2242/RelWithDebInfo.zip
Unzip the zip again. Copy the file **`rtlsdr.dll`** located in RelWithDebInfo\rtl-sdr-release**x32** to the directory containing SDRsharp.

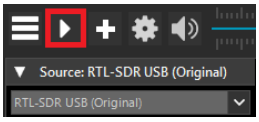


Run SDRSharp as administrator (right click)

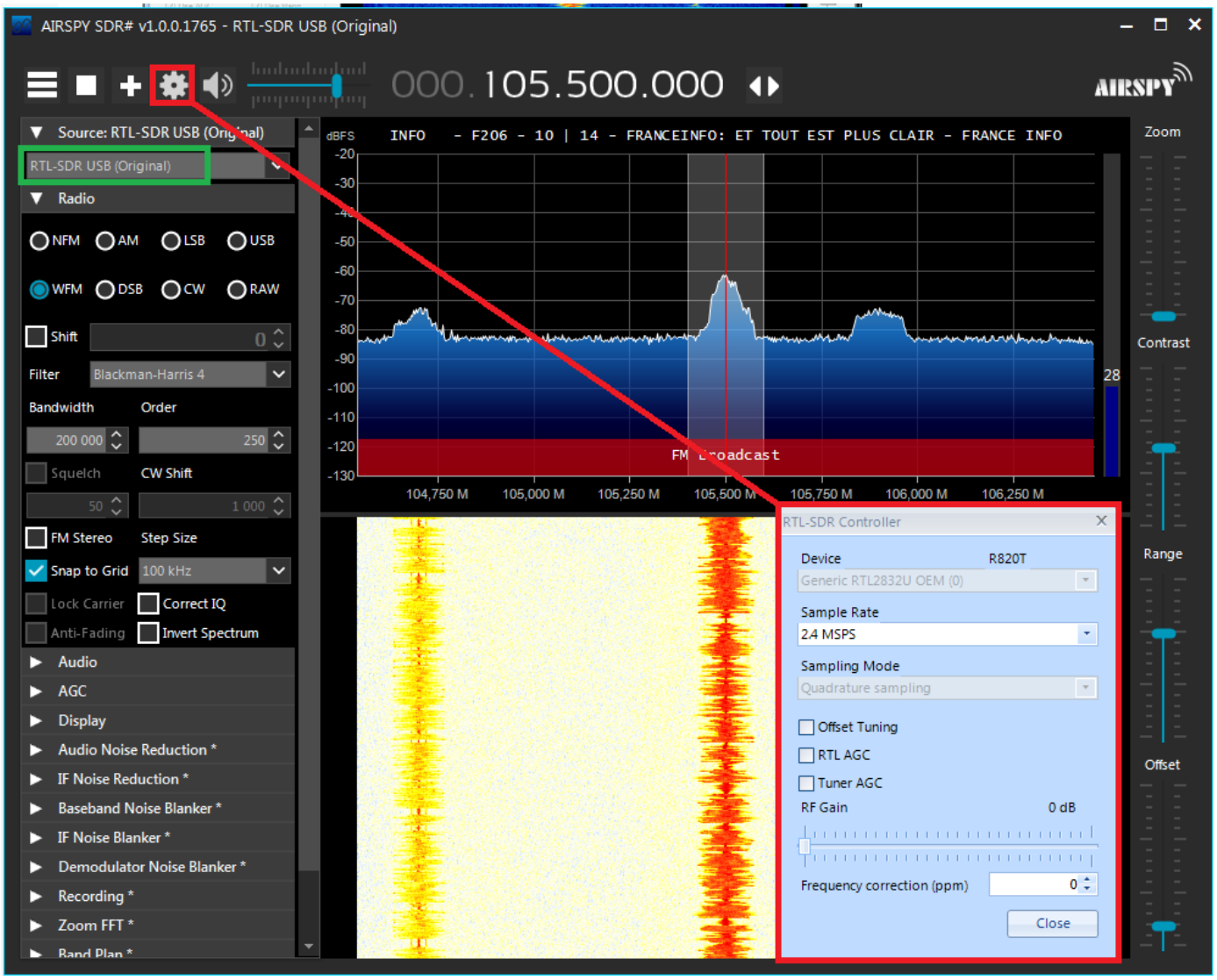
The first time SDRSharp is launched, the NET Framework may not be installed. In this case a WEB page opens automatically. Download the **`ndp48-web.exe`** file, then install the framework.



Select the rtl-sdr key in the list, then start listening by clicking on the triangle (play).



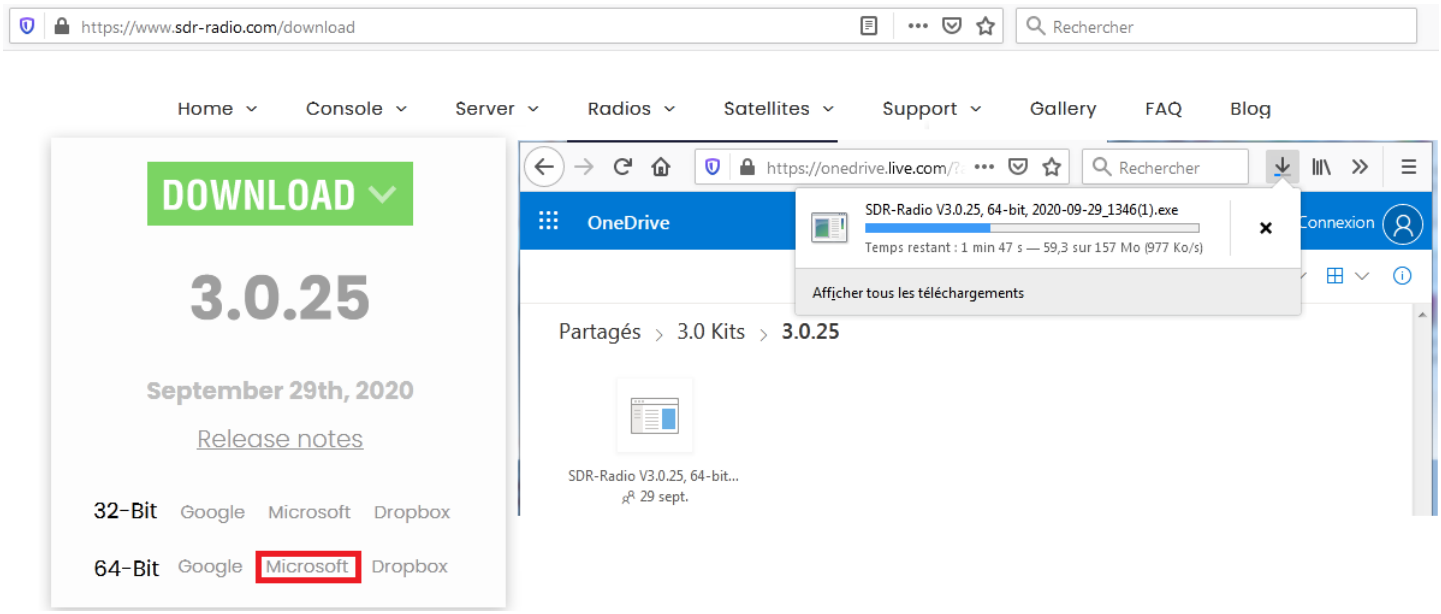
Test reception by listening to the [FM band](#) (88-108Mhz)



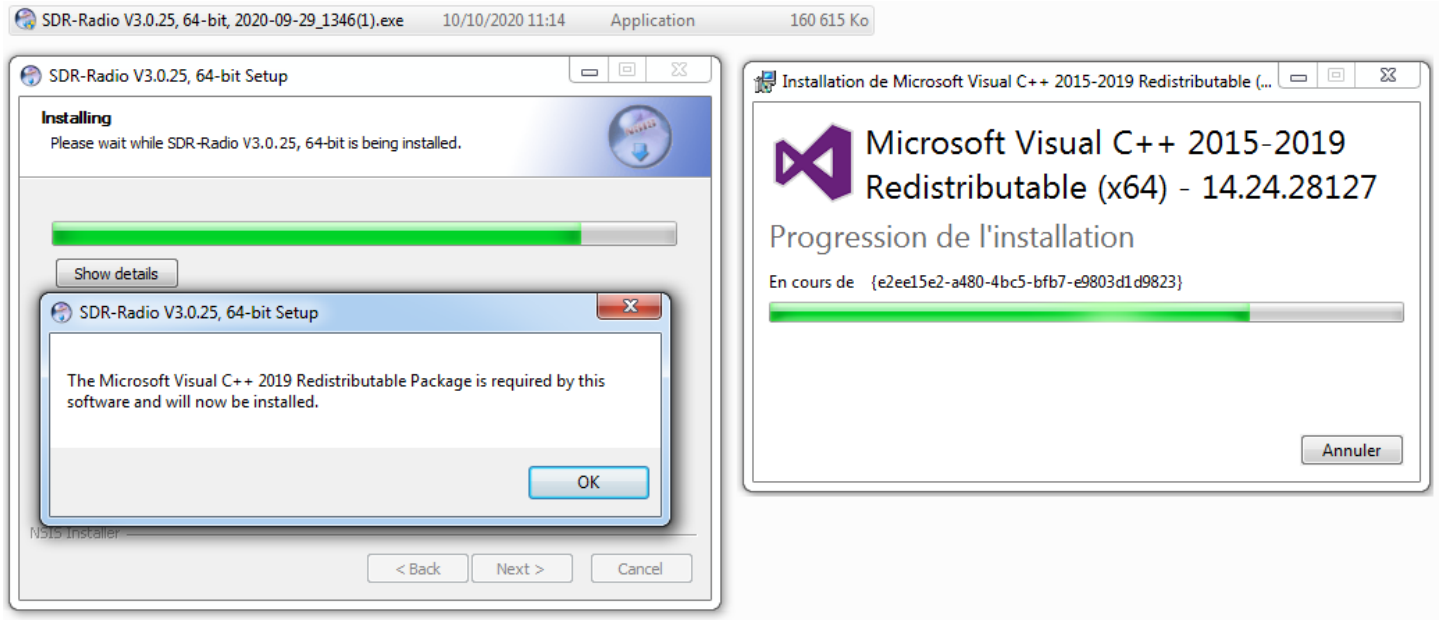
9.2.4 Sdr console

Go to https://www.sdr-radio.com/download and download the SDR-Radio `V3.0.25, 64-bit, 2020-09-29_1346.exe` file or a more recent version.

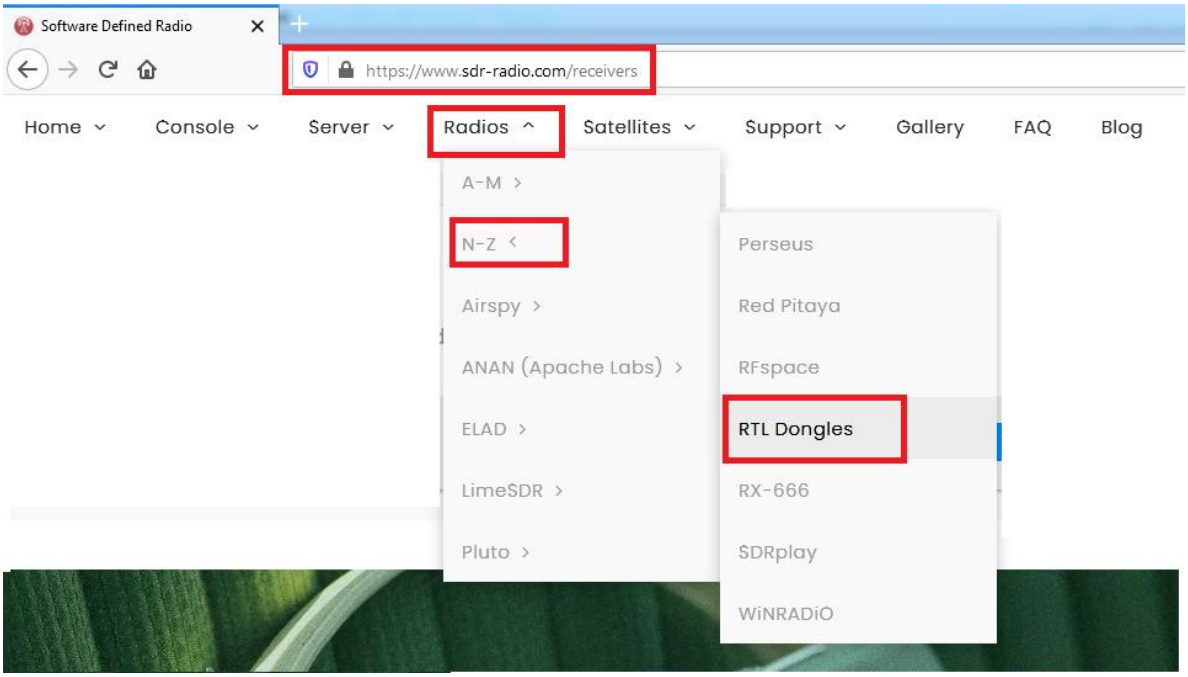
Choose the Microsoft 64-bit link.



When running the installer (setup), the Visual C++ Redistributable libraries may not be installed. In this case accept this additional installation.

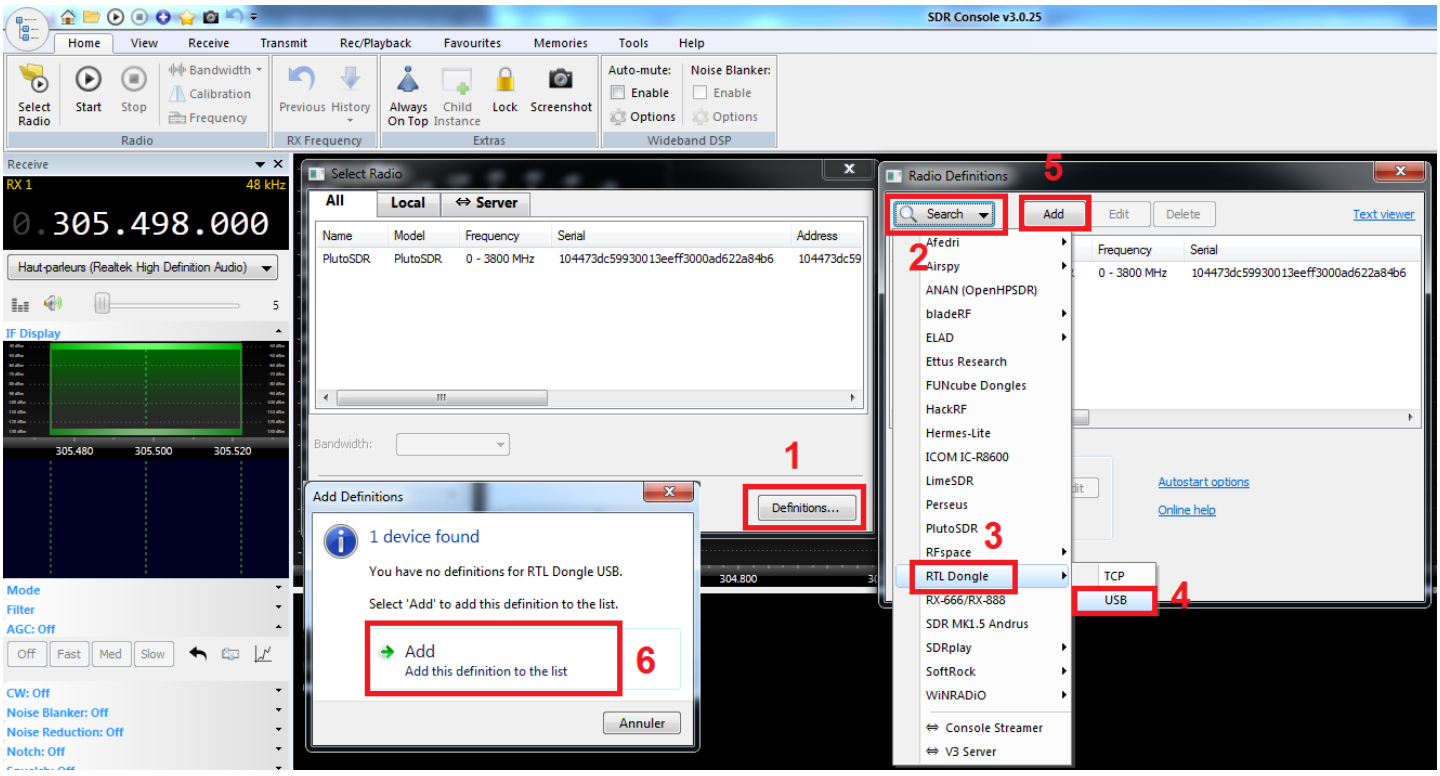


Help for configuring the rtl-sdr key can be found in the Radio RTL Dongles menu.
https://www.sdr-radio.com/rtl-dongles

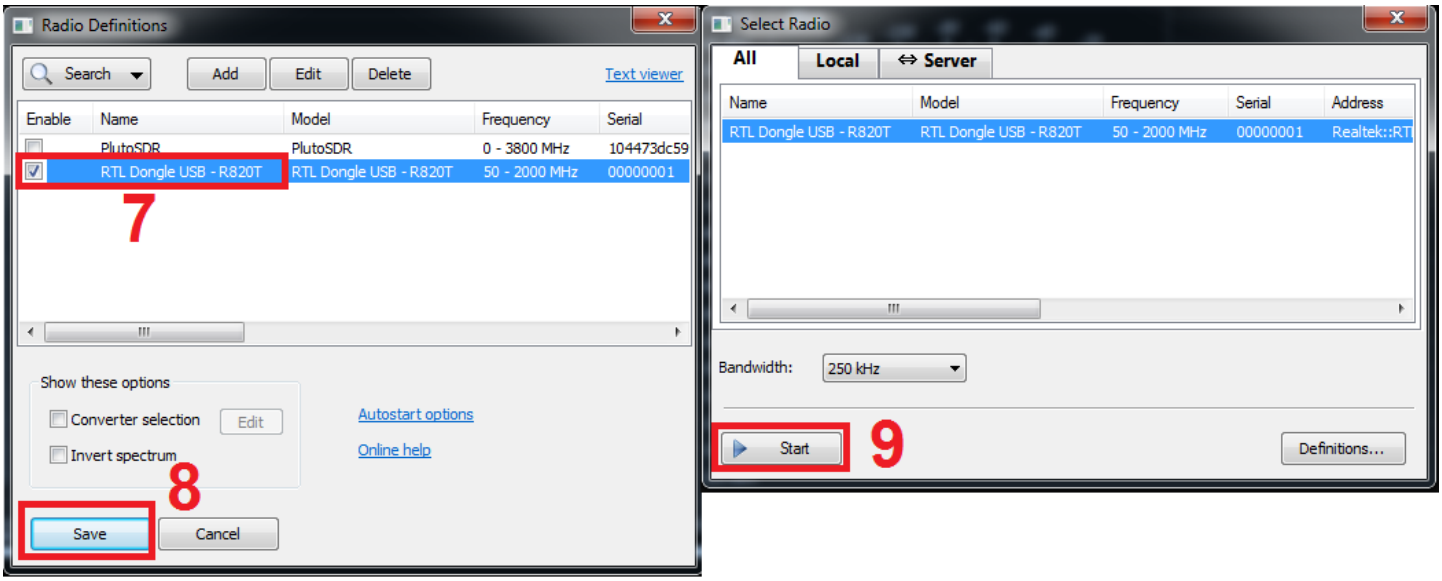


Run the sdr program called **`console`** Console

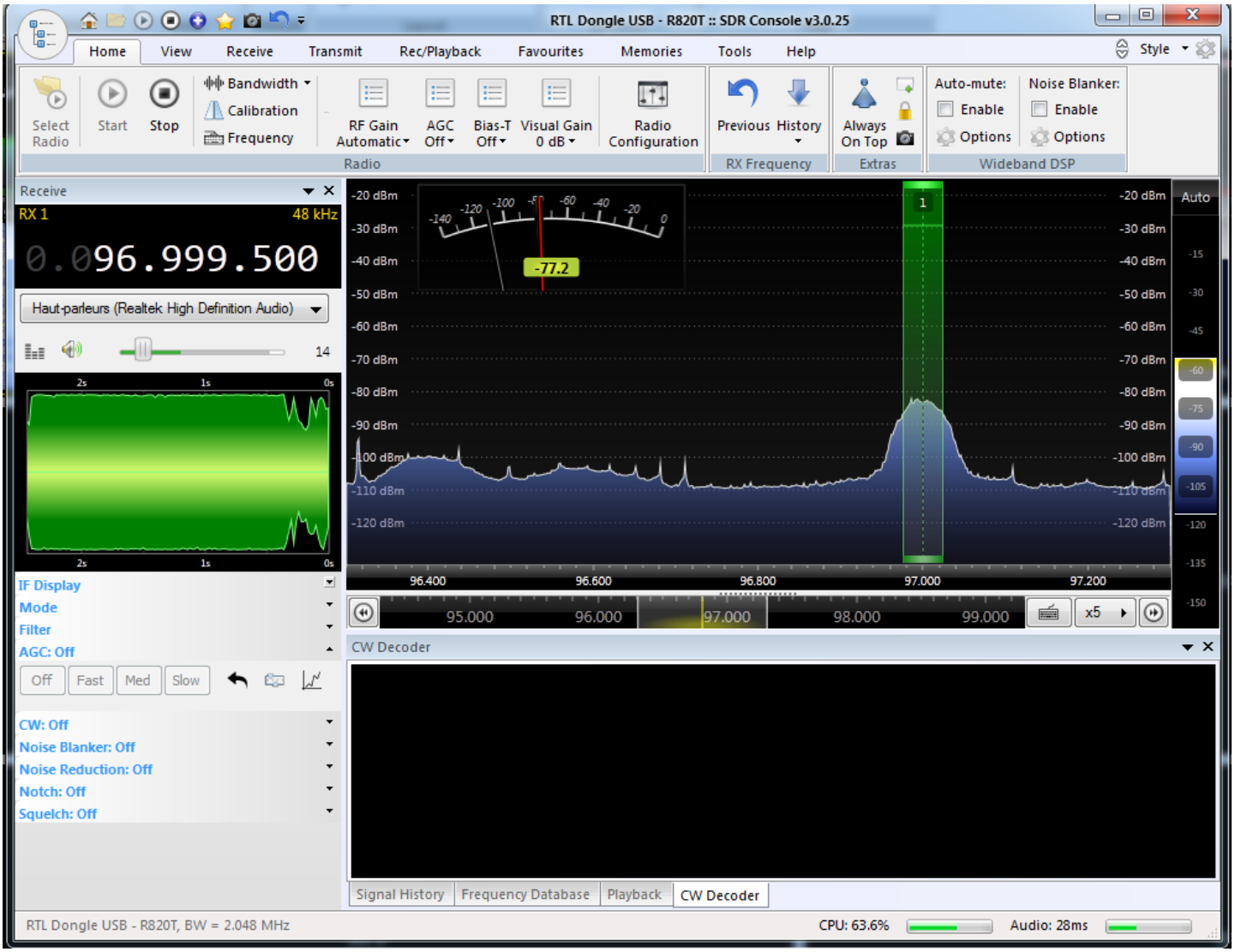
Select the rtl-sdr key by following steps 1 to 6.



Continue with steps 7 to 9.

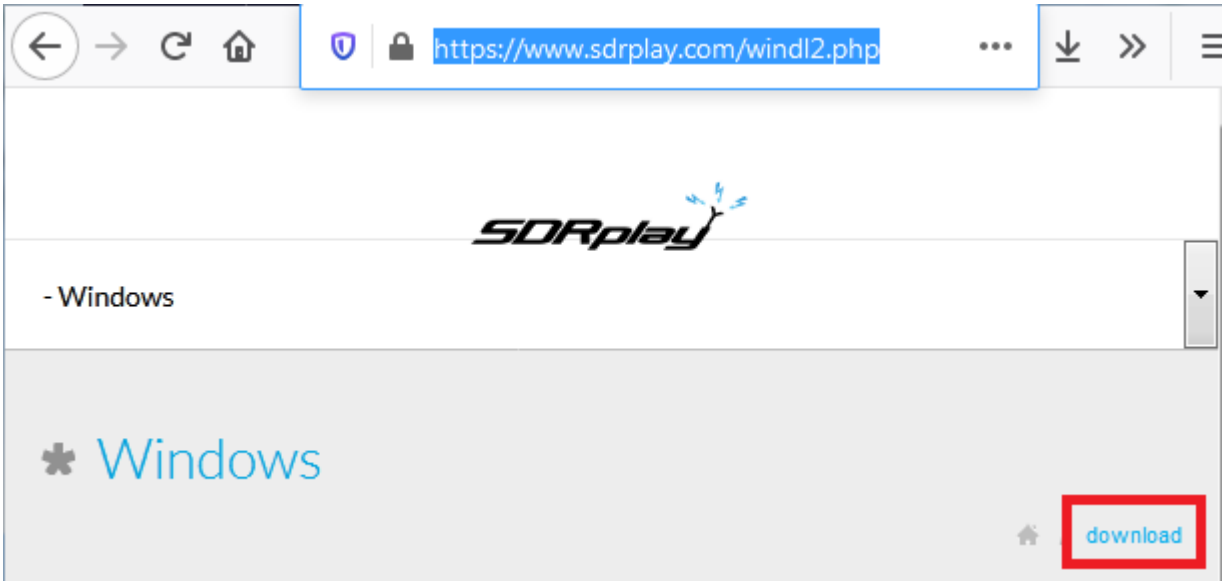


Test reception by listening to the [FM band](#) (88-108Mhz)

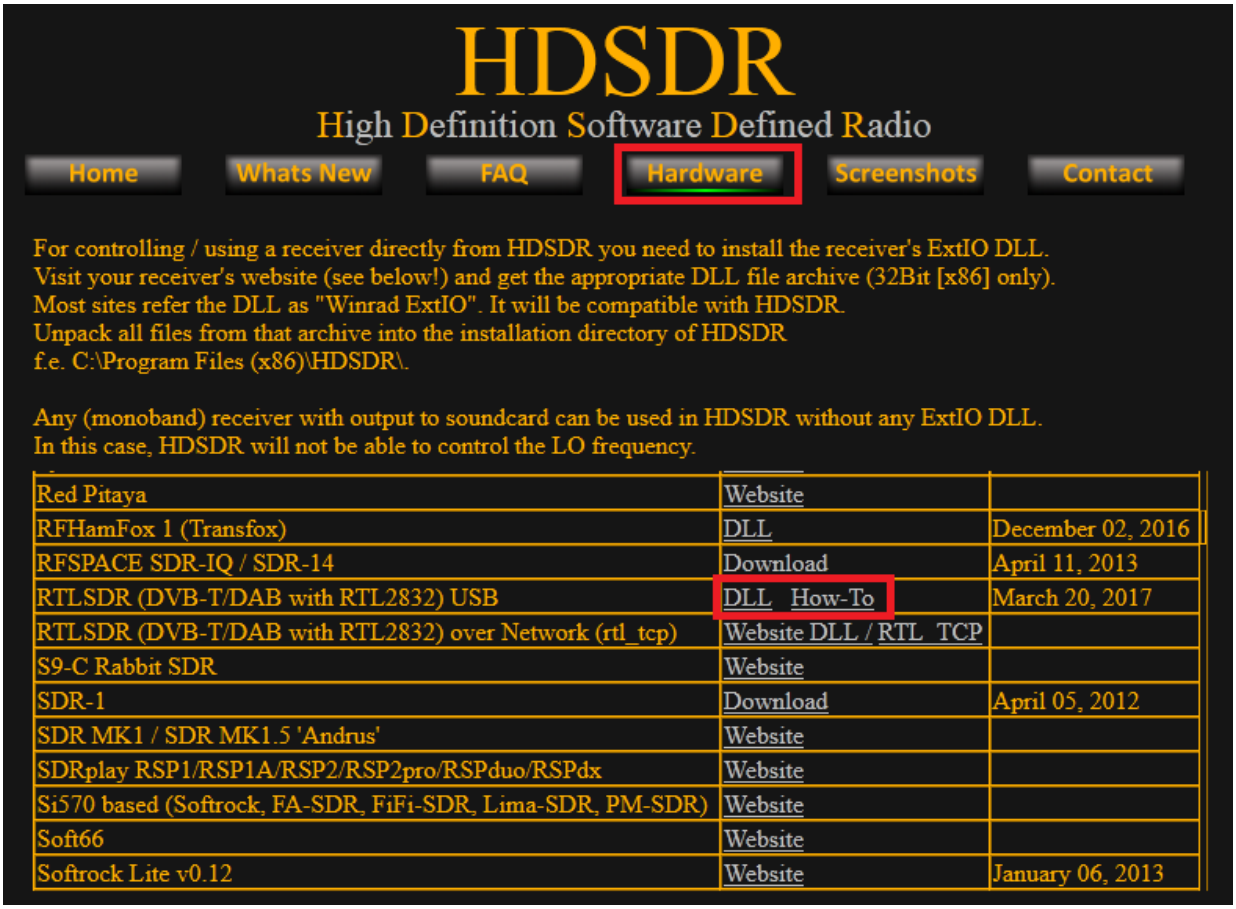


9.2.5 Sdruno

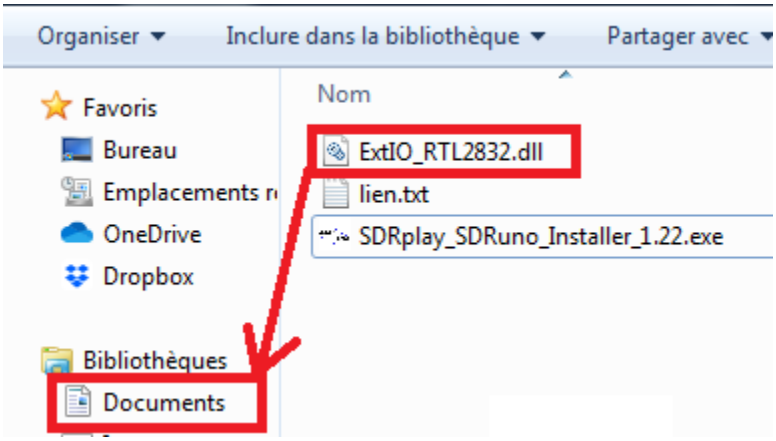
Go to https://www.sdrplay.com/windl2.php, then download the `SDRplay_SDRuno_Installer_1.22.exe` file or a more recent version.



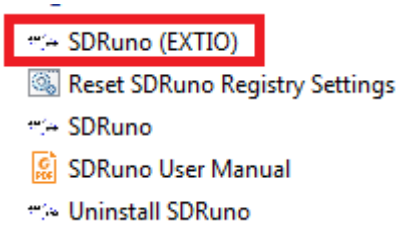
Then download the file **`ExtIO_RTL2832.dll`** the same file as for Hdsdr.
http://hdsdr.de/download/ExtIO/ExtIO_RTL2832.dll



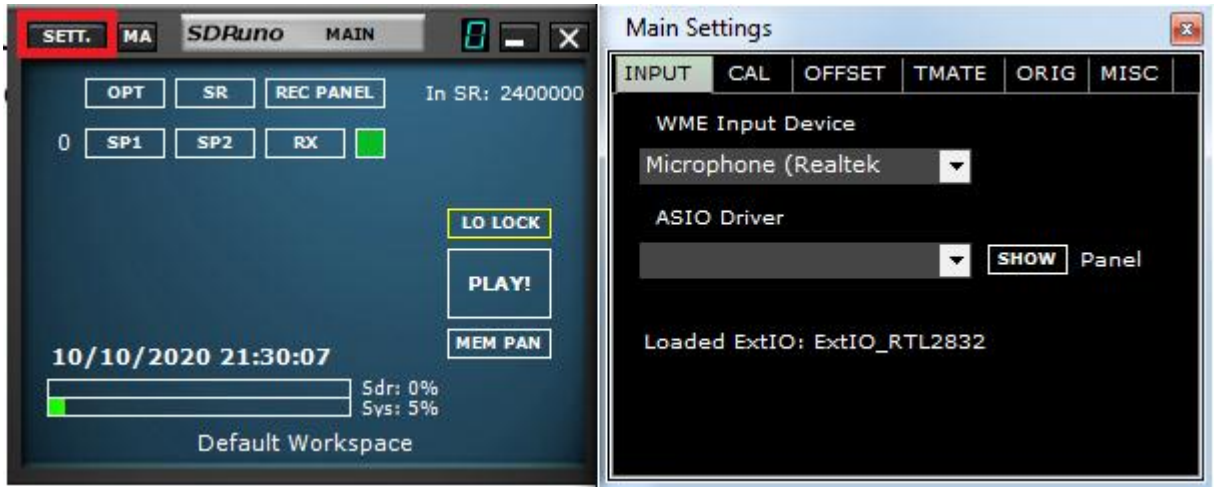
Copy the **`ExtIO_RTL2832.dll`** file to the Documents directory



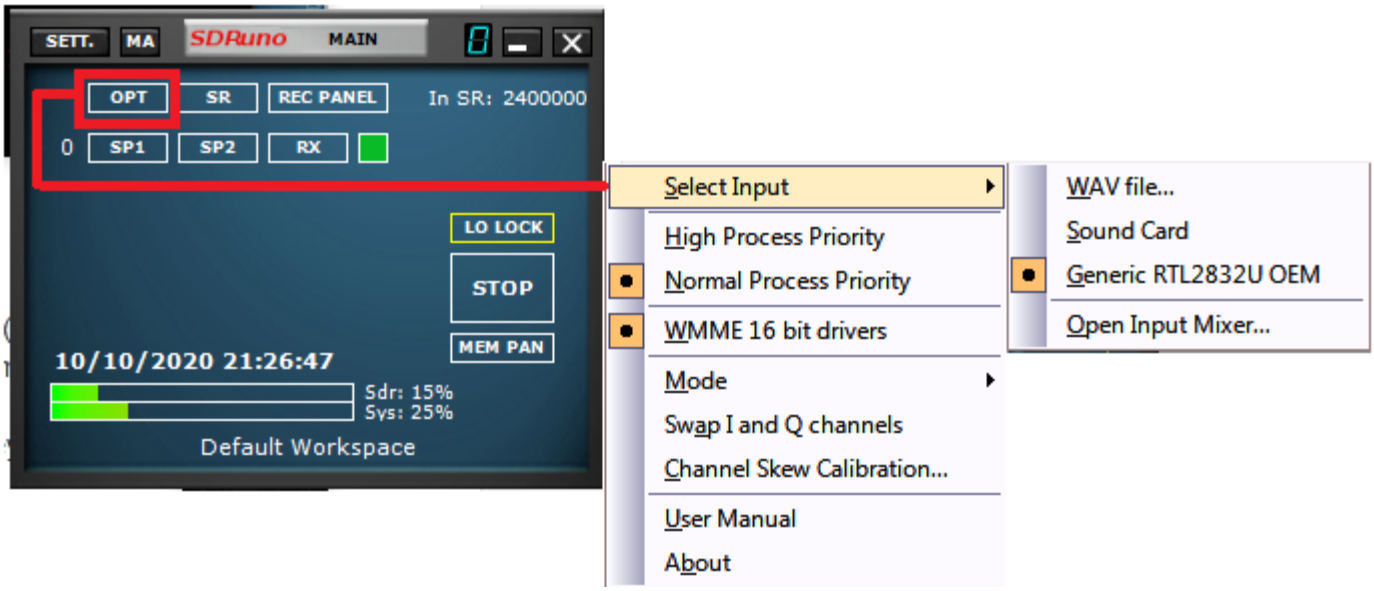
Execute the SDRuno (Extio) file, installed on the hard disk. (Search when you don't know where the program was installed). Note that there is a manual in pdf (SDRuno User manual) and it is the only manual installed with the program among the 5 softwares presented here.



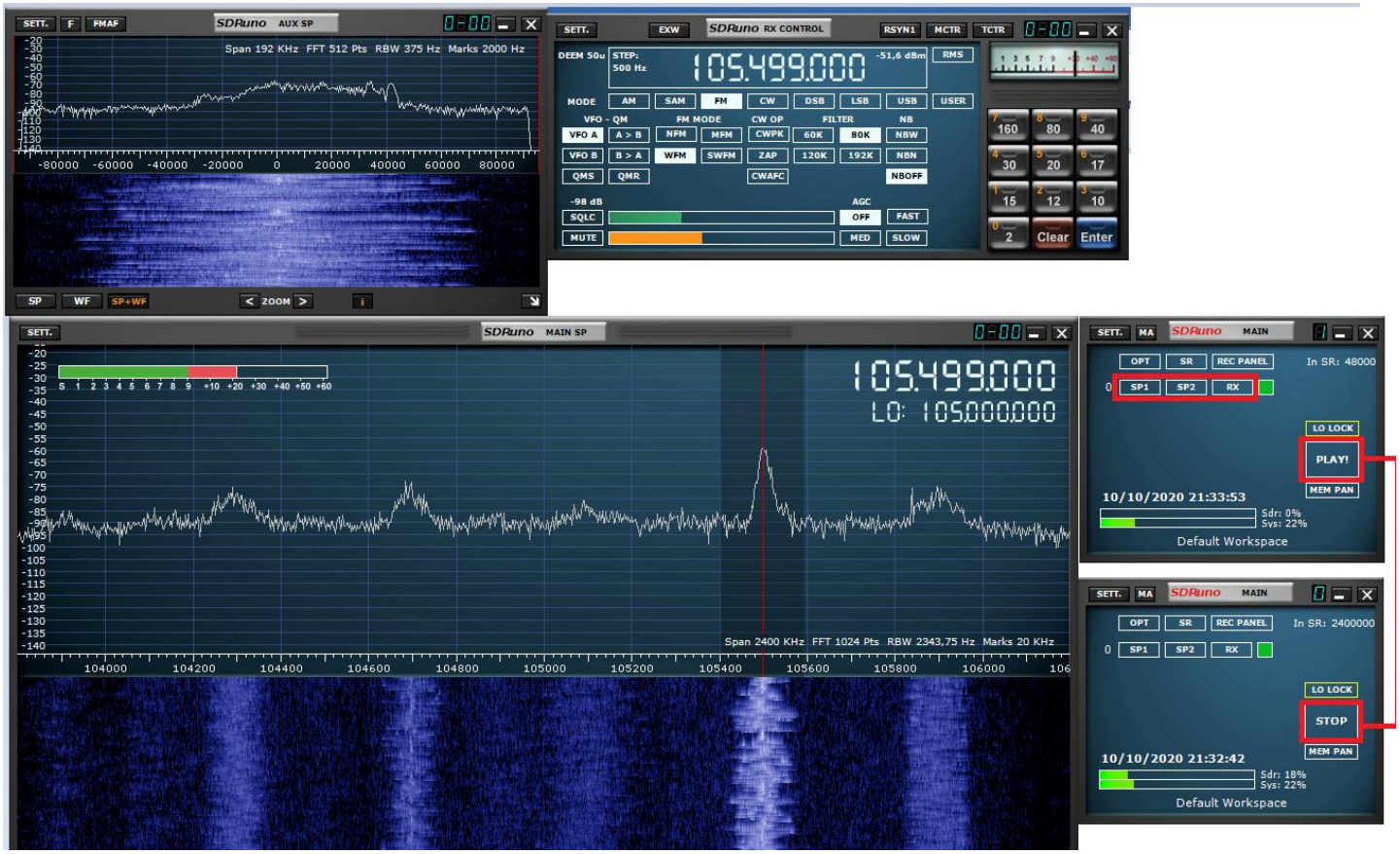
Check that the `ExtIO_RTL2832.dll` file is correctly taken into account so that the software can access the rtl-sdr key.



2nd check with the OPT menu



The SP1, SP2 and RX buttons display the 3 windows to set the frequency and reception mode.

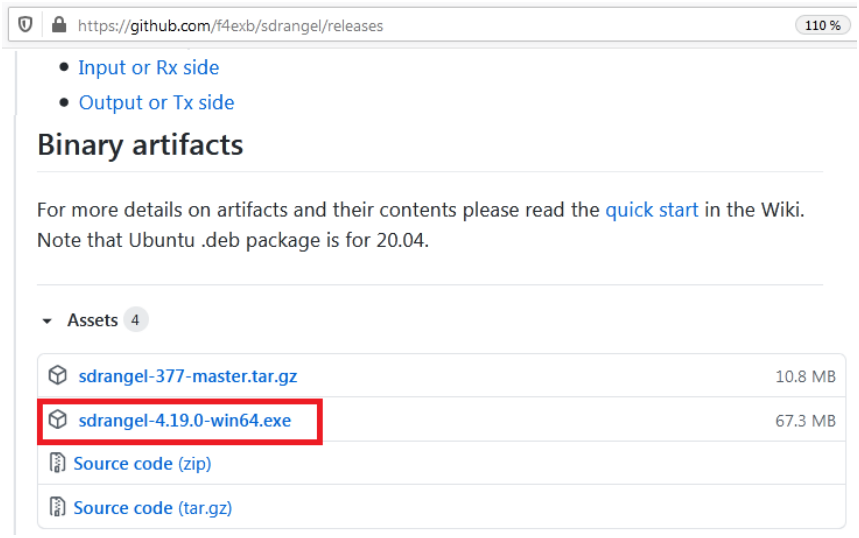


The configuration described here comes from the site radioforeveryone.com

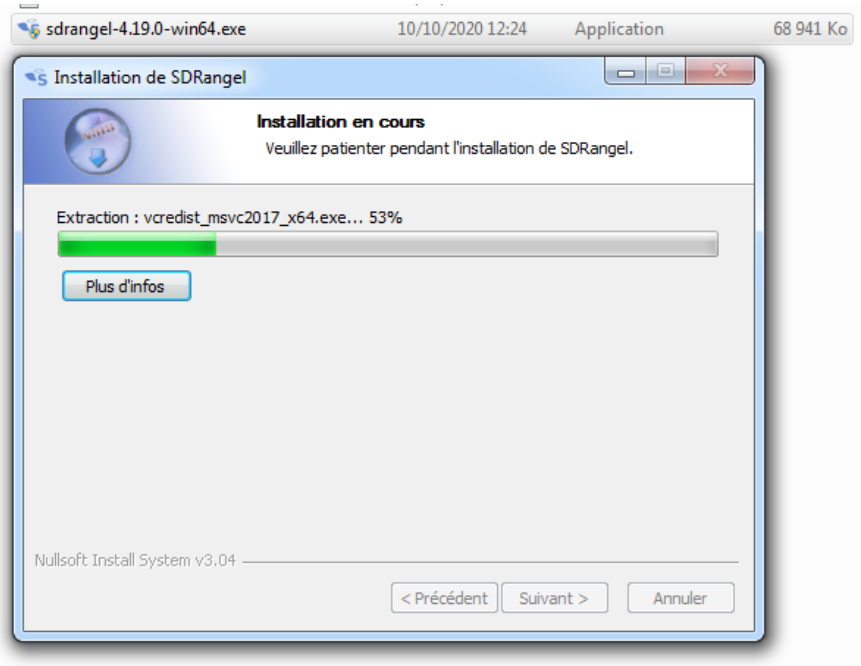
A video describing the configuration process is also available on youtube

9.2.6 SDRAngel

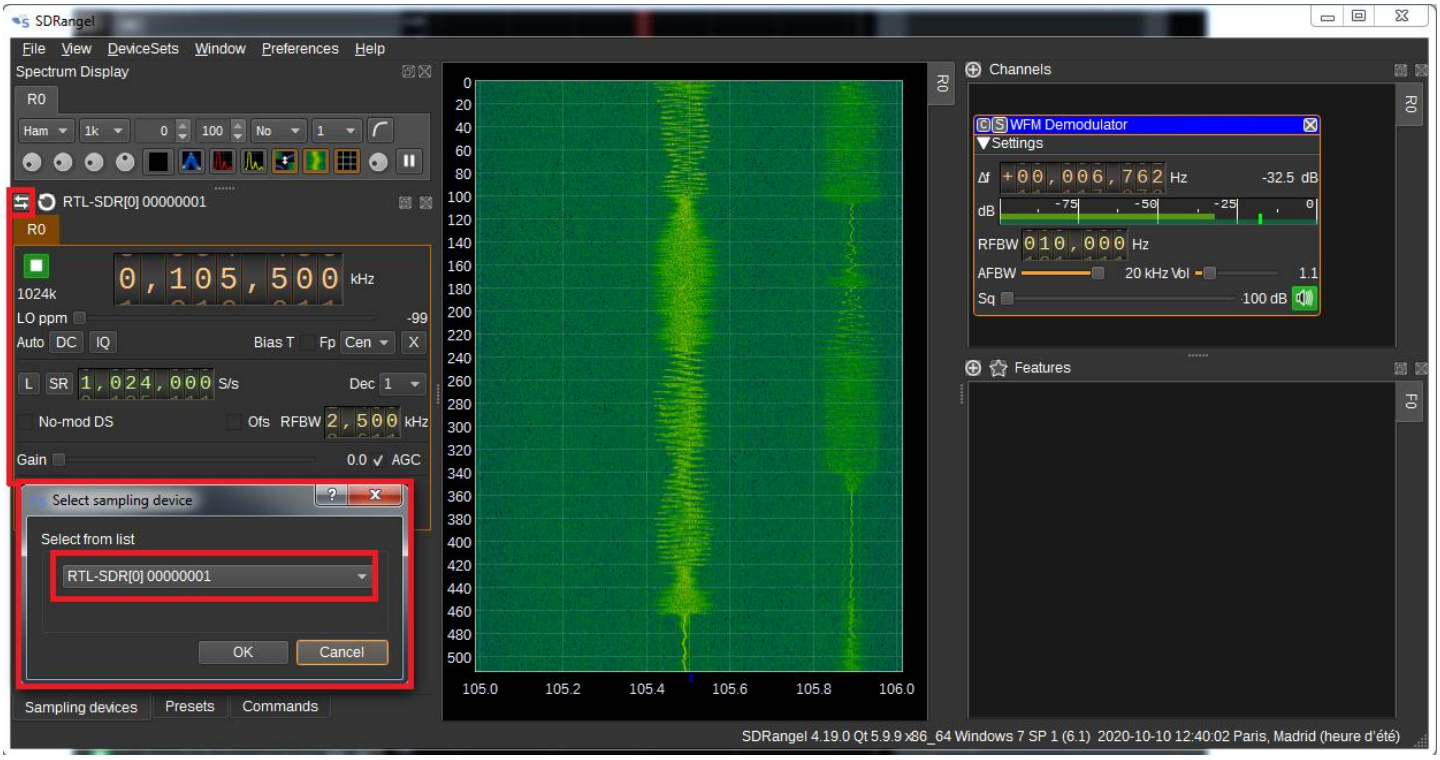
Go to https://github.com/f4exb/sdrangel/releases, then download the file `sdrangel-4.19.0-win64.exe` or a more recent version.



Run Installer



Once the program SDRangel is launched, select the menu with the arrows and choose the rtl-sdr key. Then click on the triangle.



I am not a specialist of this software. To tell the truth, I don't really know how to use it. The F4EXB author has made a wiki. Maybe an internet search on Youtube will also help you.

9.3 Installation de rpitx

As F5OEO indicates in its project, Raspbian Lite must be used. To do so, please refer to part 4 of my tutorials. The installation of command line rpitx does not pose any problem.

```
sudo apt-get update
sudo apt-get install git

git clone https://github.com/F5OEO/rpitx
cd rpitx
./install.sh
```

Allow 3 to 5 minutes to download the dependencies (depending on the speed rate) and install for a Raspberry Pi 3.

```
pi@raspberrypi:~ $ git clone https://github.com/F5OEO/rpitx
Clonage dans 'rpitx'...
remote: Enumerating objects: 65, done.
remote: Counting objects: 100% (65/65), done.
remote: Compressing objects: 100% (50/50), done.
remote: Total 1459 (delta 24), reused 49 (delta 15), pack-reused 1394
Réception d'objets: 100% (1459/1459), 10.57 MiB | 727.00 KiB/s, fait.
Résolution des deltas: 100% (818/818), fait.
pi@raspberrypi:~ $ cd rpitx/
pi@raspberrypi:~/rpitx $ ./install.sh
```

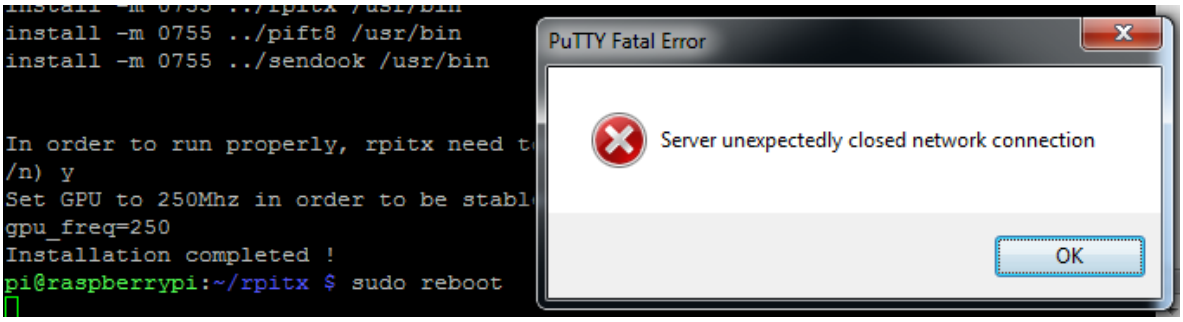
You have to accept the modification of the /boot/config.txt file, otherwise rpitx will be unstable when tx.

```
In order to run properly, rpitx need to modify /boot/config.txt. Are you sure (y
/n) y
Set GPU to 250Mhz in order to be stable
gpu_freq=250
Installation completed !
pi@raspberrypi:~/rpitx $
```

I was able to test rpitx on an RPI3 without any problem. You will have to restart the Raspberry Pi before any use.

```
sudo reboot
```

Putty's classic message when restarting RPI

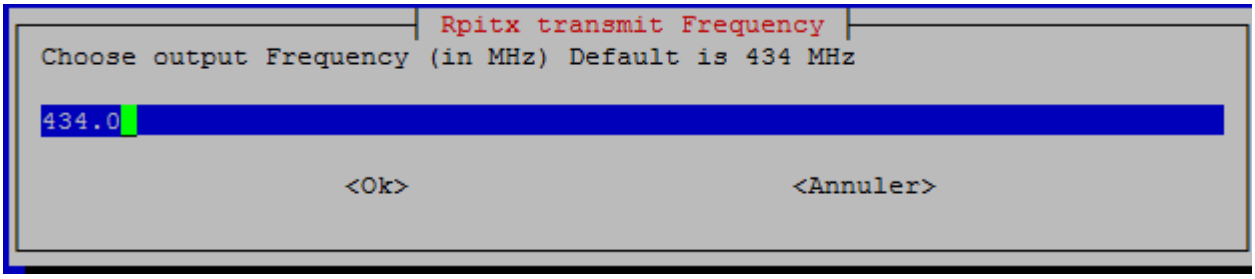


9.4 Rpitx 1st test

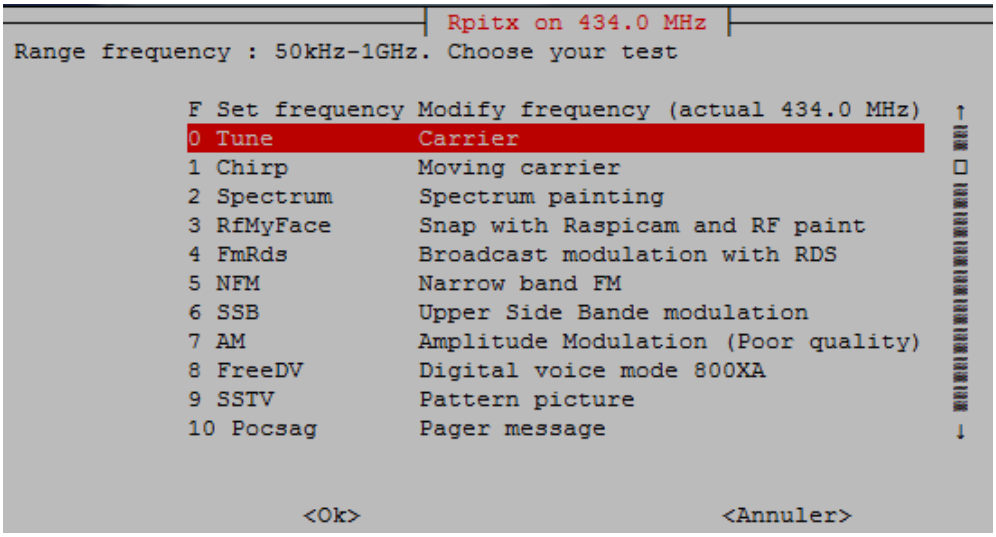
F5OEO had the great idea to add a quick test menu called `./easytest.sh` in its software suite.

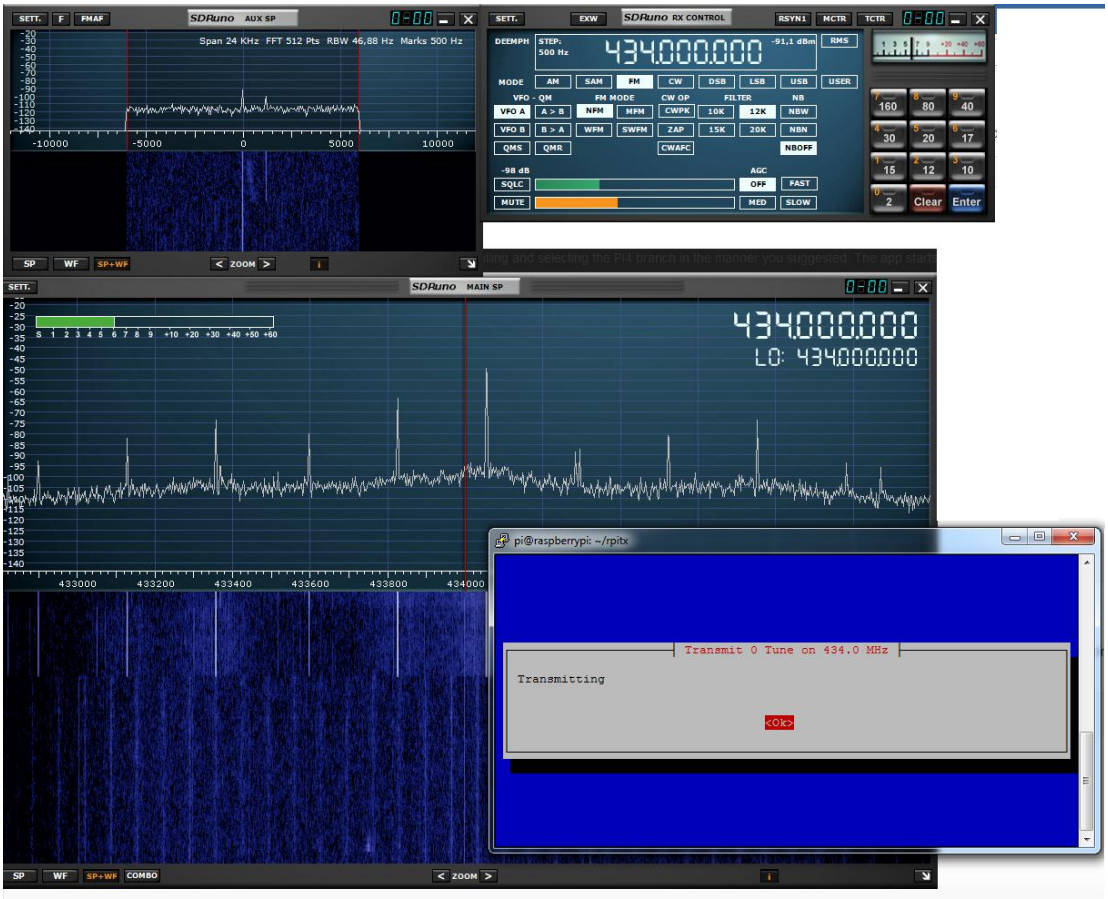
```
login as: pi
pi@192.168.1.10's password:
Linux raspberrypi 5.4.51-v7l+ #1333 SMP Mon Aug 10 16:51:40 BST 2020 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sun Oct 11 09:32:47 2020

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set
 a new password.

pi@raspberrypi:~ $ cd rpitx/
pi@raspberrypi:~/rpitx $ ./easytest.sh
```

Choose the base frequency for testing.

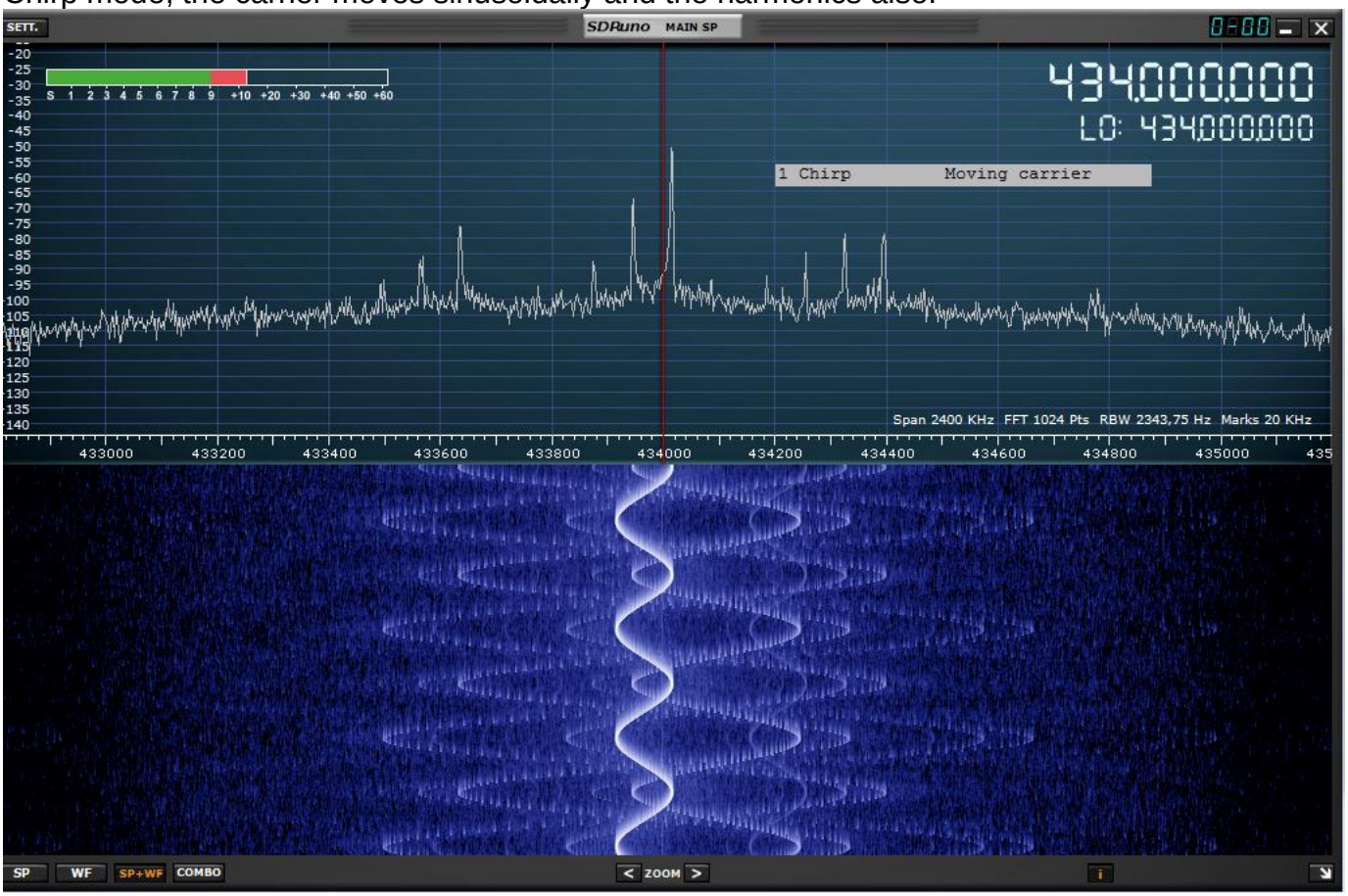


The first very classical test is to transmit a carrier and to check in the windows of the SDR software the good reception of the signal.





Chirp mode, the carrier moves sinusoidally and the harmonics also.



9.5 Others tests

Some programs can be accessed individually.

```
pi@raspberrypi:~/rpitx $ ls
BBC.jpg           morse                README.md          testfmrds.sh
core18            Not_working_with_v2  rpitx              testfoxhunt.sh
csdr              pichirp              rtlmenu.sh         testfreedv.sh
csdrpizero.diff   picture.rgb          sampleaudio.wav    testfsq.sh
doc               picture.U            sendiq             testnfm.sh
dvbrf             picture.V            sendook            testopera.sh
easytest.sh       picture.Y            snap2spectrum.sh   testpocsag.sh
fm2ssb.sh         pifmrds              snapsstv.sh        testspectrum.sh
foxhunt           pifsq                spectrumpaint      testssb.sh
freedv            pift8                src                testsstv.sh
ft8menu.sh        piopera              sv1afnfilter.sh    testvfo.sh
install.sh        pisstv               testam.sh          transponder.sh
LICENCE           pocsag               testchirp.sh       tune
pi@raspberrypi:~/rpitx $
```

9.5.1 Fox hunt


```
pi@raspberrypi:~/rpitx $ ./foxhunt
usage : foxhunt frequency(Hz) frequency shift(Hz)
pi@raspberrypi:~/rpitx $ sudo ./foxhunt 434000000 500
^CCaught signal - Terminating 2 ->touches CTRL+C
pi@raspberrypi:~/rpitx $
```

9.5.2 On-off keying TXing

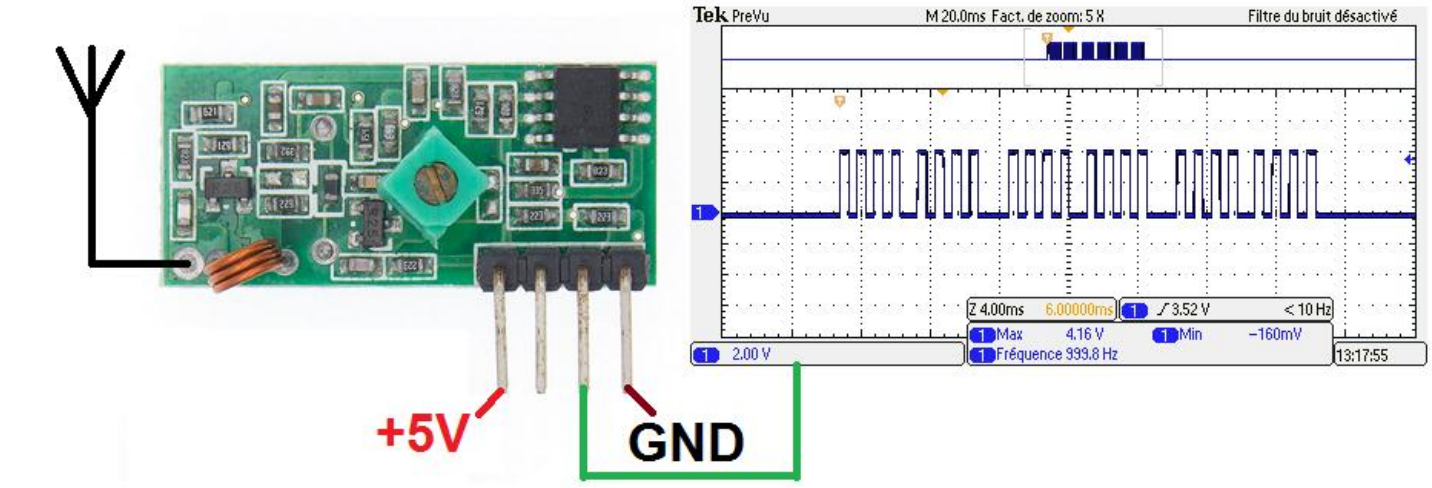
On-off keying is an 1-or-0 transmission used in home automation systems such as garage door control.

```
pi@raspberrypi:~/rpitx $ ./sendook -h
sendook : a program to send On-Off-Keying with a Raspberry PI.
usage : sendook [options] "binary code"
Options:
-h : this help
-v : verbose (-vv : more verbose)
-d : dry run : do not send anything
-f freq : frequency in Hz (default : 433.92MHz)
-0 nb : duration in microsecond of 0 bit (by default : 500us). Use integer only.
-1 nb : duration in microsecond of 1 bit (by default : 250us)
-r nb : repeat nb times the message (default : 3)
-p nb : pause between each message (default : 1000us=1ms)

"binary code":
  a serie of 0 or 1 char (space allowed and ignored)

Examples:
  sendook -f 868.3M -0 500 -1 250 -r 3 1010101001010101
    send 0xaa55 three times (with the default pause of 1ms) on 868.3MHz. A 0 is a gap of
500us, a 1 is a pulse of 250us
pi@raspberrypi:~/rpitx $
```

Fortunately the help is present with the **-h**

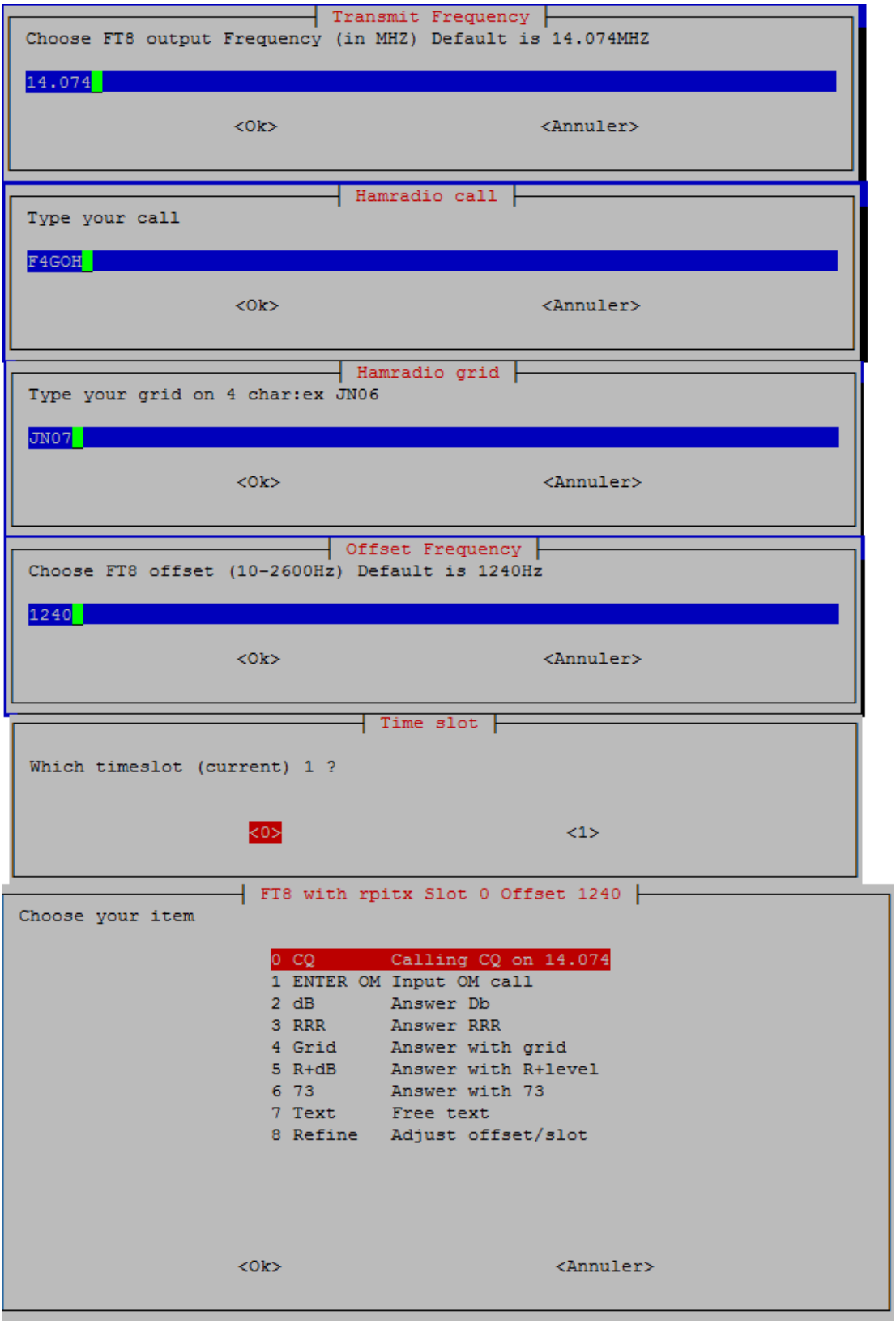
```
pi@raspberrypi:~/rpitx $ sudo ./sendook 1010101001010101
Frequency set to : 433920000Hz
Bit duration 0 : 500us ; 1 : 500us
Send message 3 times with a pause of 1000us
Message successfuly transmitted
pi@raspberrypi:~/rpitx $
```

It is not necessary to specify the frequency. By default, the reception frequency of cheap modules is 433.92 MHz. That's good, the result displayed with the oscilloscope corresponds perfectly to the data in console.



9.5.3 FT8 transmission

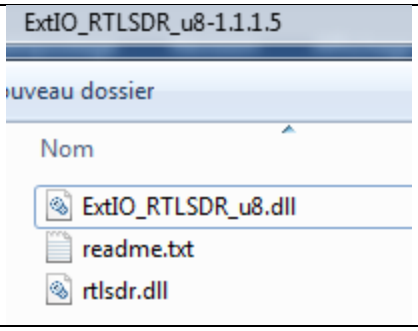
A specific menu has been created: `pi@raspberrypi:~/rpitx $ ./ft8menu.sh`



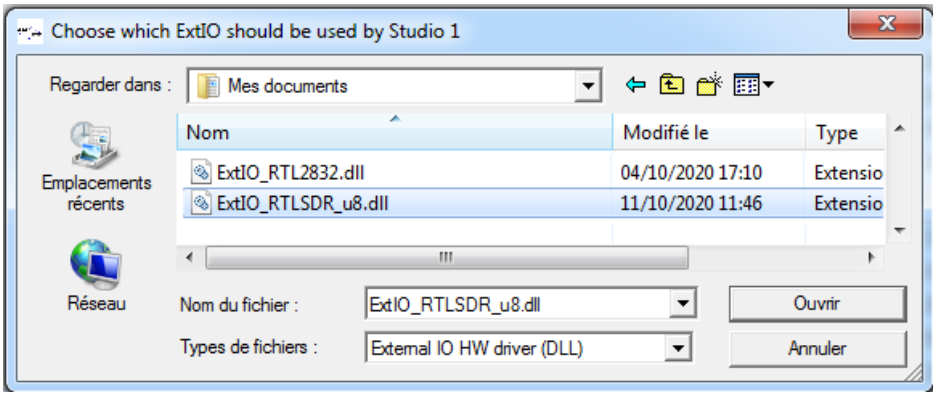
With SDRuno you will have to use the Direct Sampling mode with another dll file by using the configuration on page 13.

Link of the rtl-sdr web page specifying the page of the forum proposing the file **`ExtIO_RTLSDR_u8.dll`**

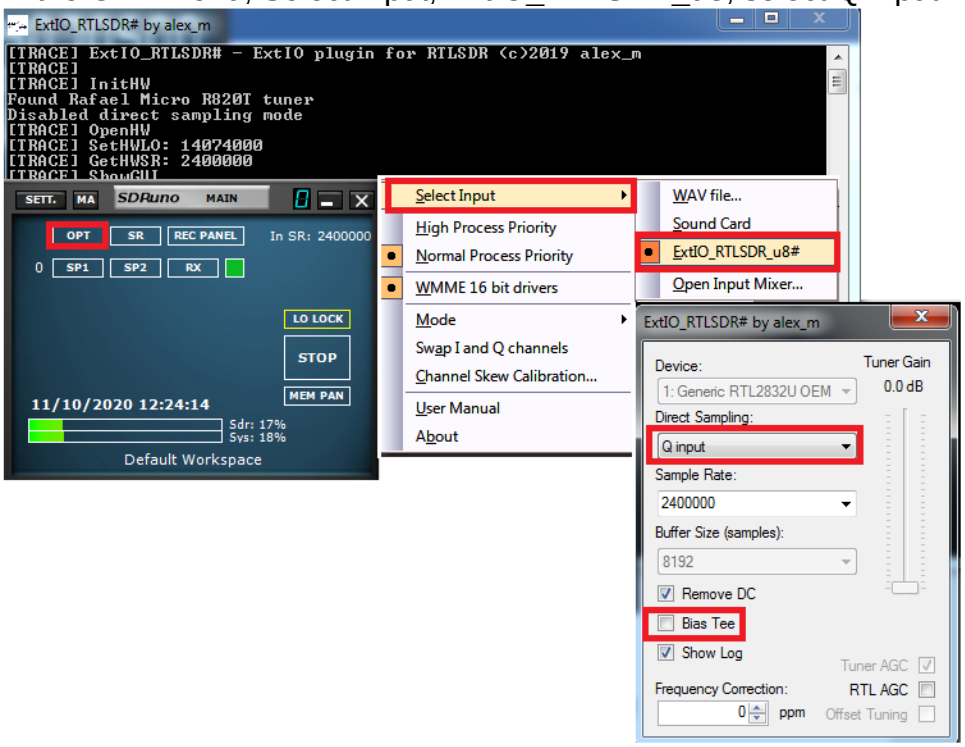
https://www.rtl-sdr.com/forum/download/file.php?id=926

Il faudra copier les 2 fichiers dll dans le répertoire Documents **`ExtIO_RTLSDR_u8.dll`** et **`rtlsdr.dll`**	

As soon as you start SDRuno (EXTIO) , select the **`ExtIO_RTLSDR_u8.dll`** file, in order to be able to listen to the HF bands.



In the OPT menu, Select input, ExtIO_RTLSDR_u8, select Q input.

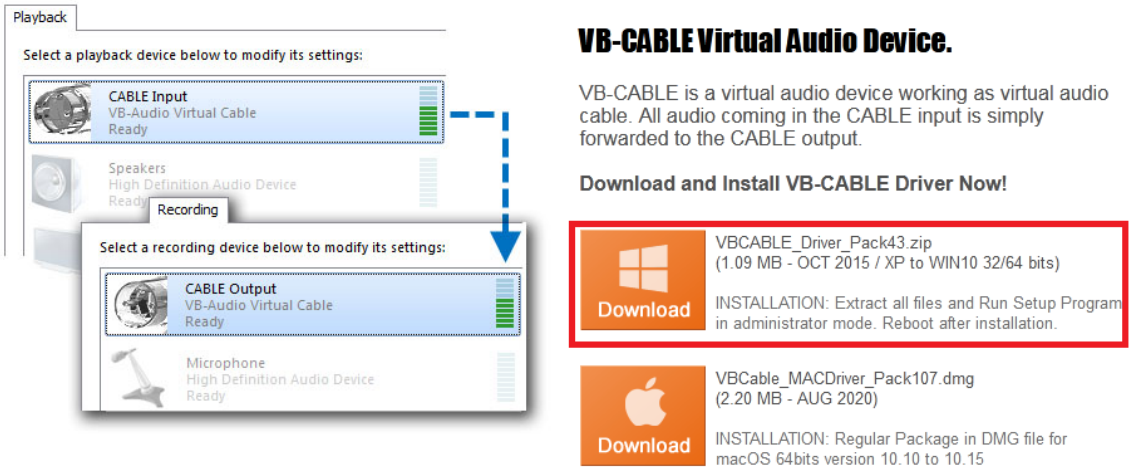


Starting a QC from the FT8 rpitx menu

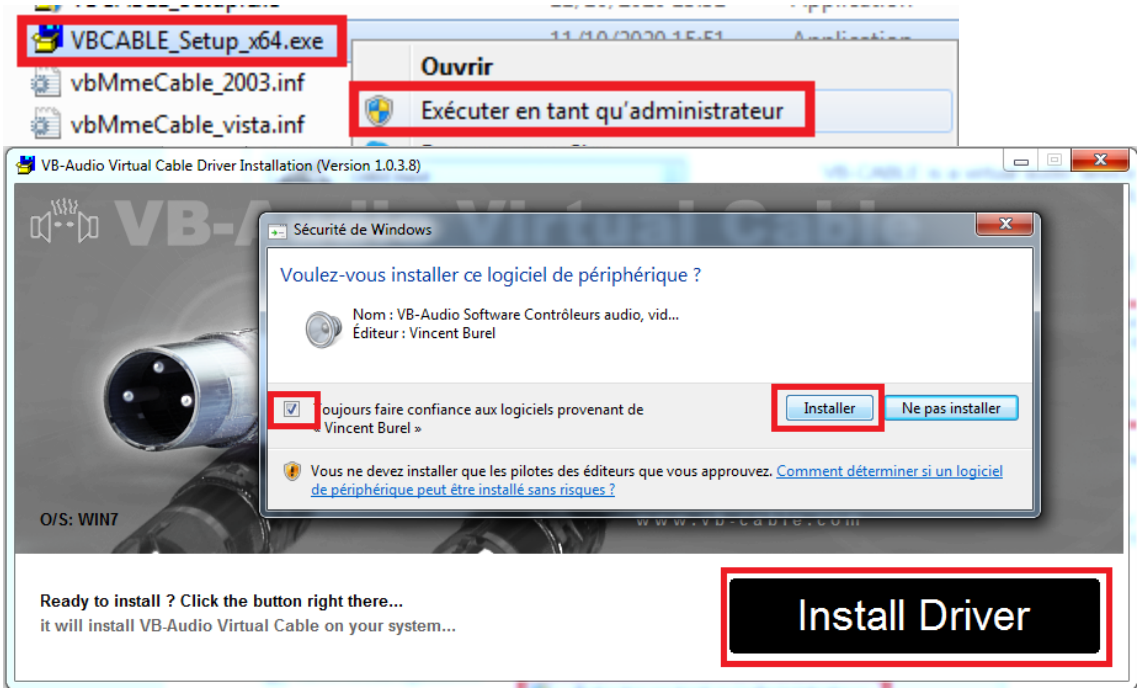
The signal must appear in the "waterfall".

To decode the FT8 transmission, a "virtual audio bridge" (Virtual audio cable) will have to be installed.
An explanation can be found on the leradioscope.fr site, or YouTube video

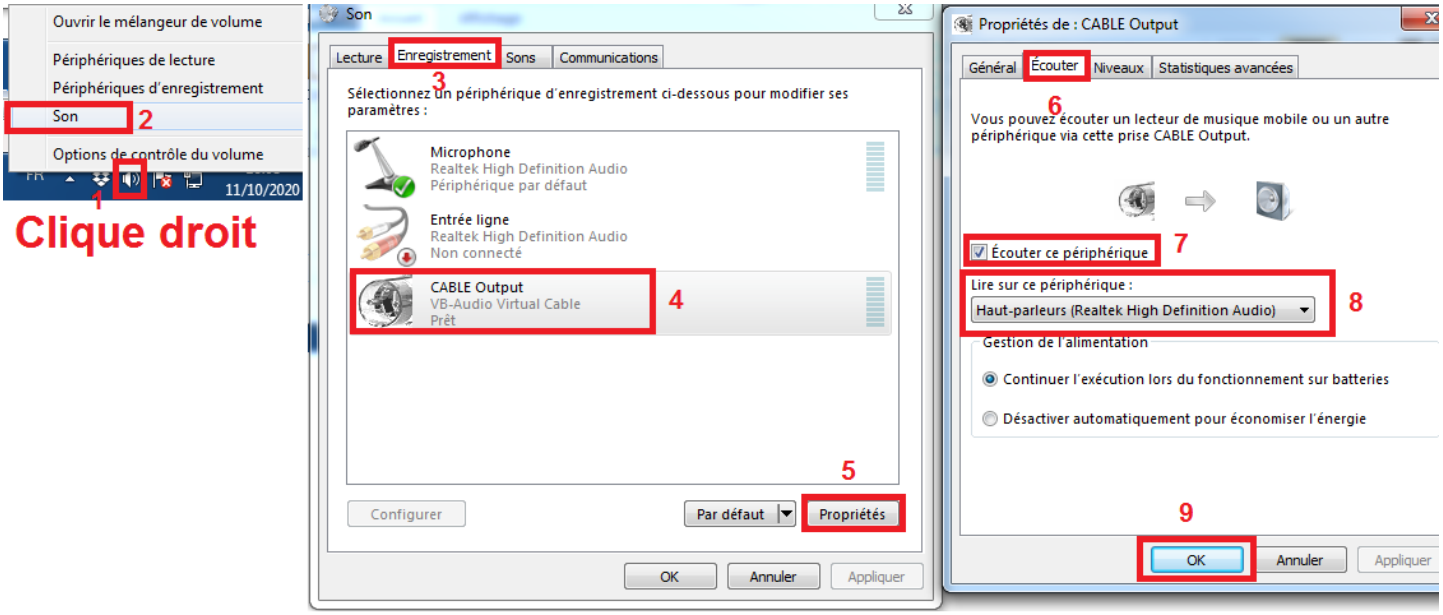
Start by downloading the file VBCABLE_Driver_Pack43.zip



Installer les Drivers en mode administrateur (clique droit)

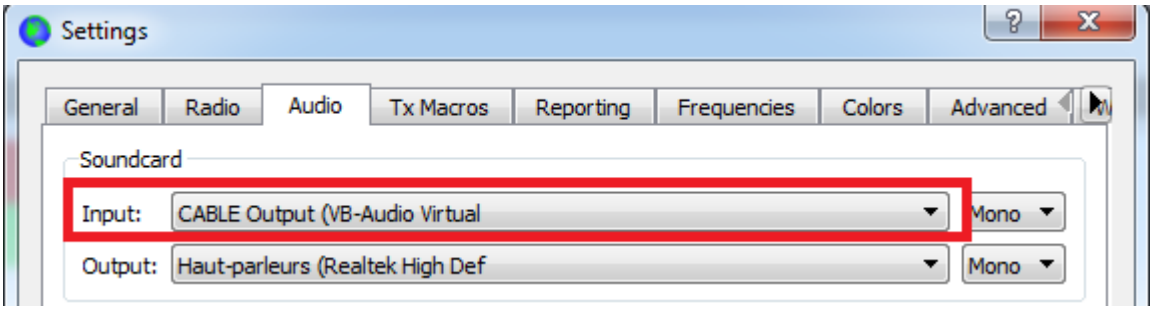


Some screenshots from the video

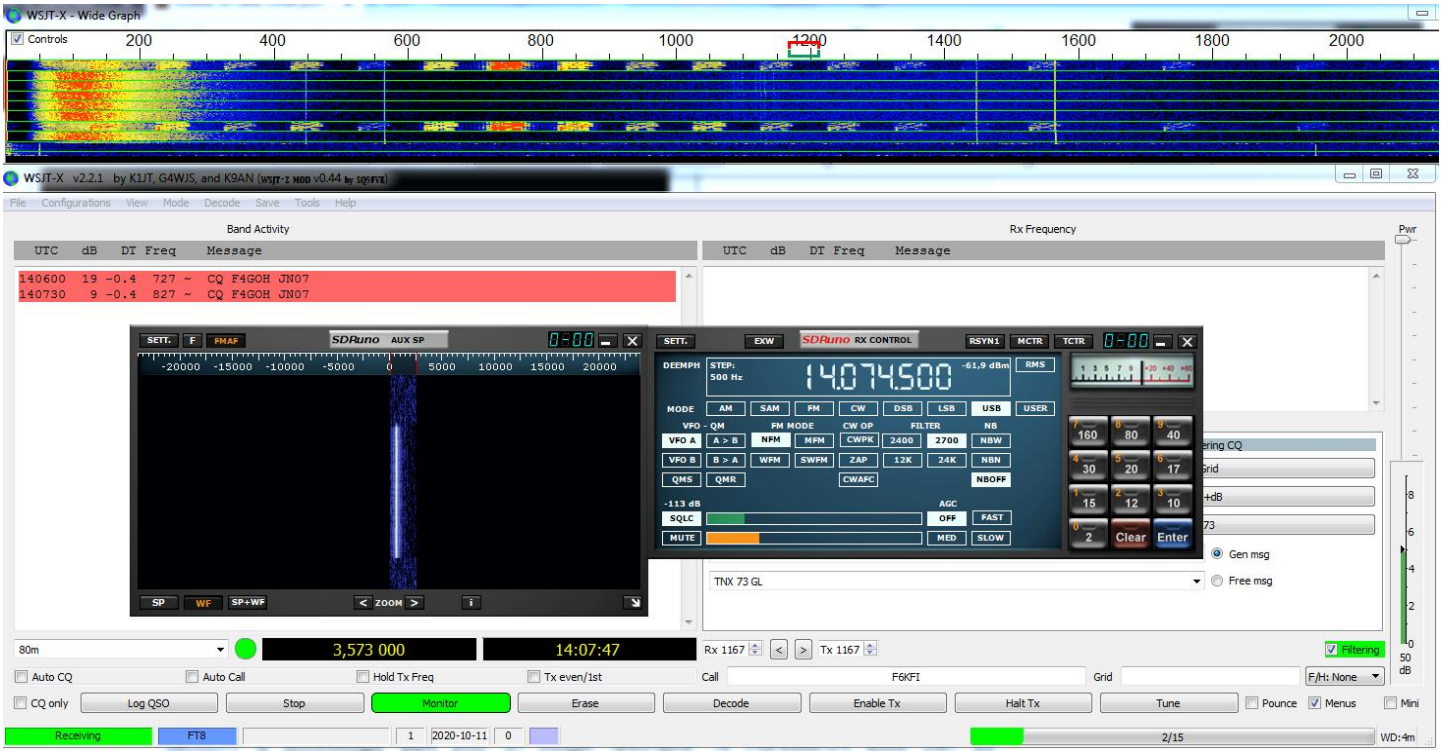


Then configure SDRuno

Finish by configuring the reception source in the WSJT-X software.

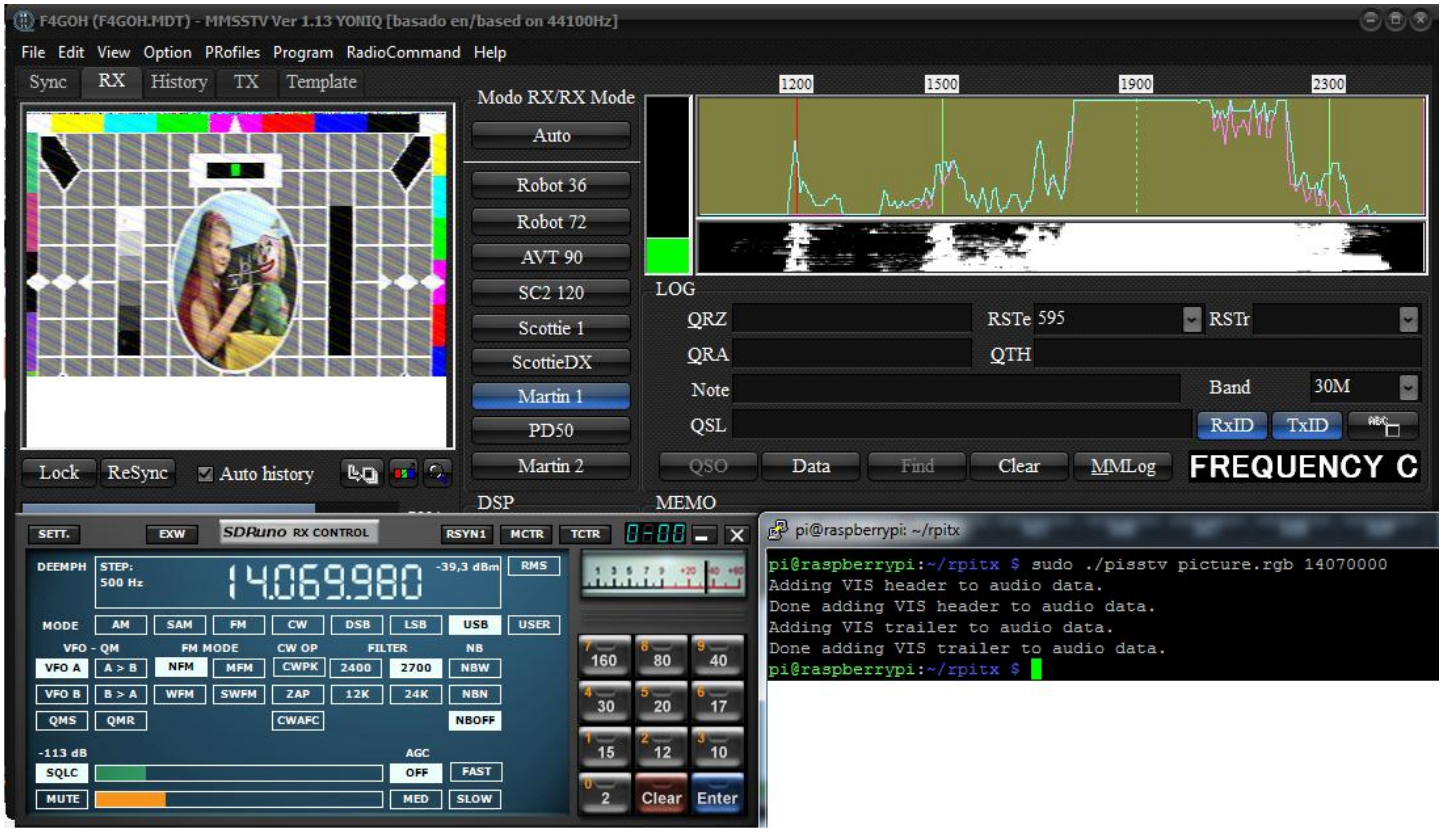


As soon as you switch to repeat transmission, the HF signal is present in both waterfalls (SDRuno and Wsjtx).



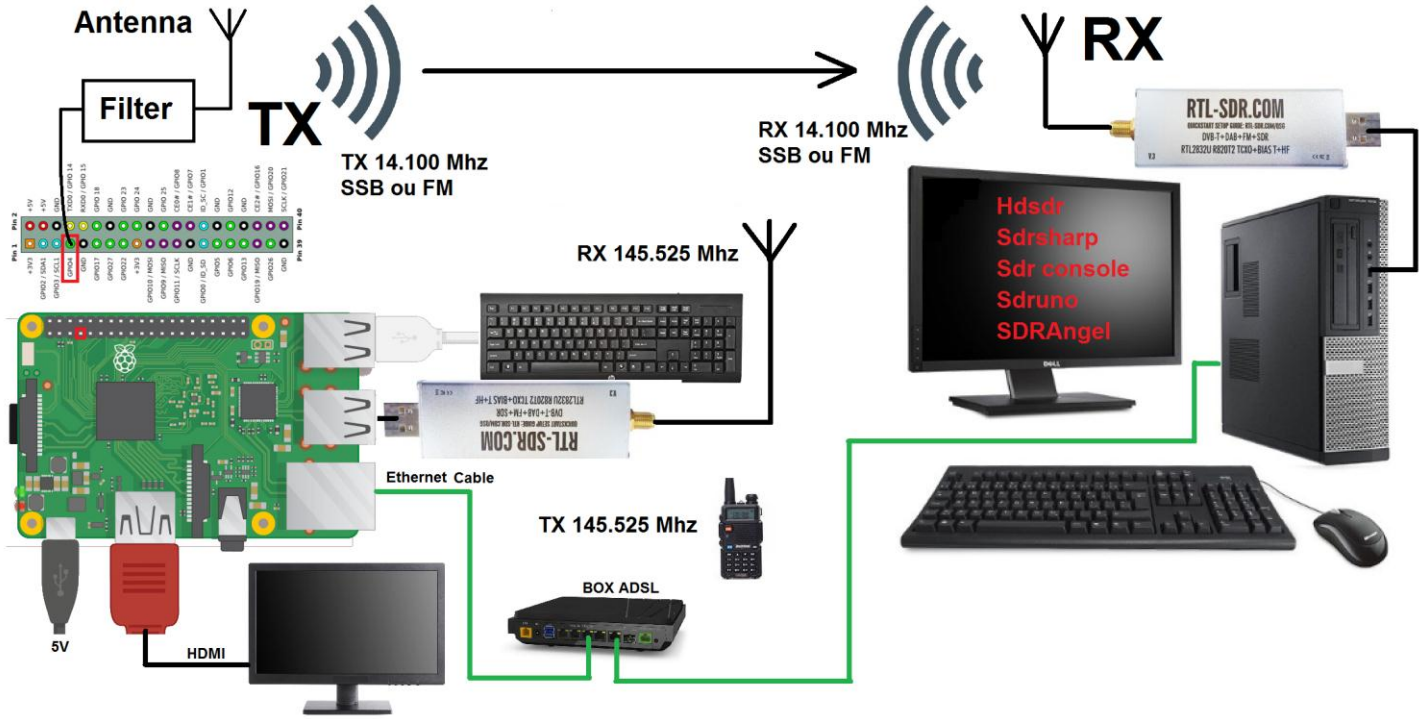
It is the same for the decoding of an SSTV image (MMSSTV installation help).

```
sudo ./pisstv picture.rgb 14070000
```



9.5.4 Transponder

Objective: A Pocket PC in FM broadcasts on 145.525 Mhz. Rpitx equipped with an rtl-sdr key receives the transmission and copies the signal in Fm or SSB on 14.100Mhz.



To access the transponder function, just use the following specific script.

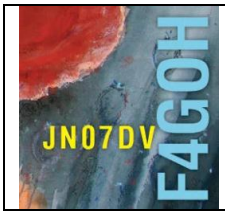
```
pi@raspberrypi:~/rpitx $ ./rtlmenu.sh
```

```
                    Rpitx with RTLSDR
Choose your test

          0 Record        Record spectrum on 145.525
          1 Play          Replay spectrum
          2 Transponder   Transmit 145.525 to 14.100MHZ
          3 Fm->SSB       Transcode FM 145.525 to 14.100MHZ
          4 Set frequency Modify frequency (actual 145.525 Mhz)
```

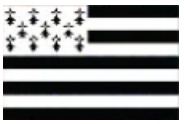
Menu 4 allows you to choose the input and output frequency.

9.5 Conclusion

The **rpitx** project is very interesting and will allow to understand and experiment with different data transmissions without spending anything. This project also allowed me to make a synthesis of SDR software installation under Windows to have a quick guide when I will have forgotten everything in 6 months. If an amplifier is used, it will be necessary to think about the low-pass filter to eliminate the harmonics generated by the RF output of the Raspberry.



Raspberry Pi for HAM Radio
Part 10



Summary:

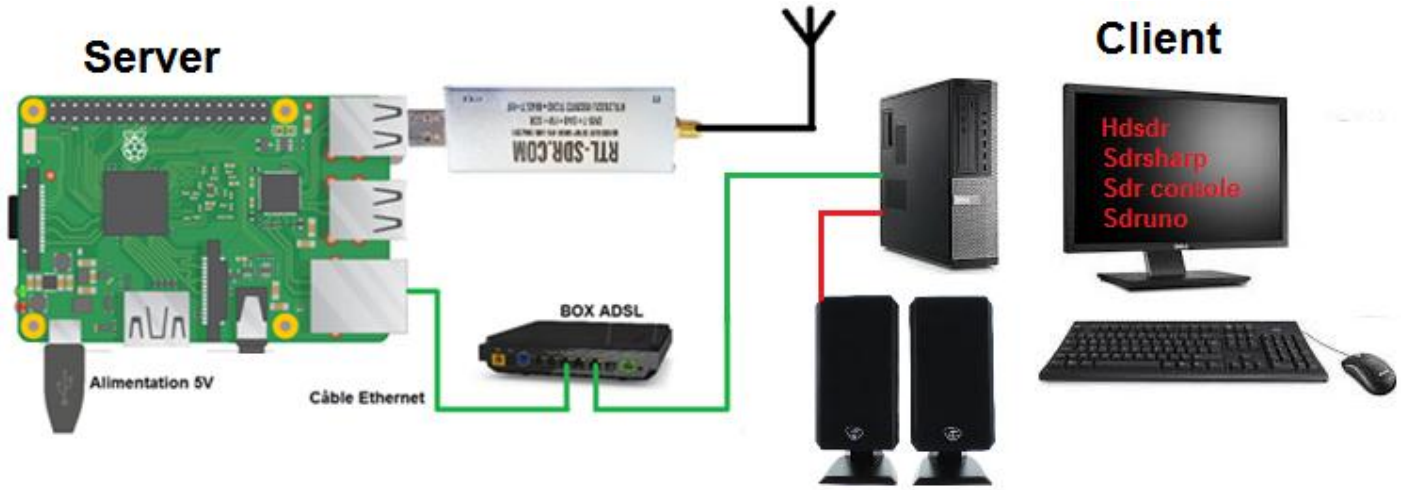
Part 10: Streaming server with Rtl sdr.

Prerequisite: Part 4: Using the Raspbian Lite operating system.

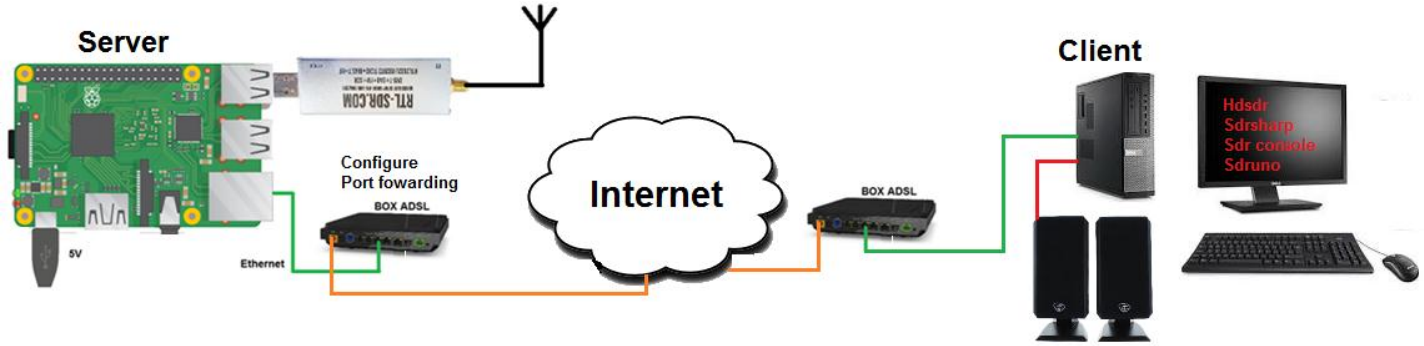
Version du 31/10/2020 V1.0 on RPI3

10.1 Presentation of the server / client set

In this tutorial, the idea is to implement a Raspberry Pi as a server for remote listening on a PC. Unlike the tutorial 6 on OpenWebRX, the client (PC) uses one of the 4 softwares (Hdsdr, Sdrsharp, Sdr console, Sdruno). The advantage is that the listening will be much more fluid than with OpenWebRX, while having all the features of the SDR reception software. The communication between the client and the server will be done in TCP/IP.



Of course, it is possible to have a connection between the server and the client via the Internet. In this case it will be necessary to configure the ADSL BOX as described in tutorial 6.8: Accessing the web SDR from outside the QRA.



10.2 Rtl-sdr drivers Installation

This time, I used the two following links to install drivers for the RTL-SDR V3 key. This version has improvements compared to the driver described in tutorial 3.

https://www.rtl-sdr.com/rtl-sdr-blog-v-3-dongles-user-guide/

https://github.com/rtlsdrblog/rtl-sdr-blog

Use Raspbian Lite without GUI. Remember to execute these commands if it is not already done.

```
sudo apt update
sudo apt upgrade
sudo apt install git
```

Recovering the project:

```
git clone git://github.com/rtlsdrblog/rtl-sdr-blog.git
```

```
pi@raspberrypi:~ $ ls
rtl-sdr-blog
pi@raspberrypi:~ $
```

Install the libraries.

```
sudo apt install build-essential cmake usbutils libusb-1.0-0-dev
```

```
pi@raspberrypi:~/rtl-sdr/build $ sudo apt install build-essential cmake usbutils
 libusb-1.0-0-dev
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
build-essential est déjà la version la plus récente (12.6).
usbutils est déjà la version la plus récente (1:010-3).
Les paquets suivants ont été installés automatiquement et ne sont plus nécessair
es :
  libmicrodns0 libqt5charts5 xlog-data
Veuillez utiliser « sudo apt autoremove » pour les supprimer.
Les paquets supplémentaires suivants seront installés :
  cmake-data libjsoncpp1 librhash0 libusb-1.0-doc libuv1
Paquets suggérés :
  cmake-doc ninja-build
Les NOUVEAUX paquets suivants seront installés :
  cmake cmake-data libjsoncpp1 librhash0 libusb-1.0-0-dev libusb-1.0-doc
  libuv1
0 mis à jour, 7 nouvellement installés, 0 à enlever et 0 non mis à jour.
Il est nécessaire de prendre 4□578 ko dans les archives.
Après cette opération, 23,6 Mo d'espace disque supplémentaires seront utilisés.
Souhaitez-vous continuer ? [O/n] O
```

Type these commands in the order:

```
cd rtl-sdr-blog/

mkdir build

cd build

cmake -DINSTALL_UDEV_RULES=ON -DDETACH_KERNEL_DRIVER=ON ../
```

```
pi@raspberrypi:~ $ cd rtl-sdr-blog/
pi@raspberrypi:~/rtl-sdr-blog $ mkdir build
pi@raspberrypi:~/rtl-sdr-blog $ cd build/
pi@raspberrypi:~/rtl-sdr-blog/build $ cmake -DINSTALL_UDEV_RULES=ON -DDETACH_KER
NEL_DRIVER=ON ../
-- The C compiler identification is GNU 8.3.0
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compiler: /usr/bin/cc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Build type not specified: defaulting to release.
-- Extracting version information from git describe...
-- Found PkgConfig: /usr/bin/pkg-config (found version "0.29")
-- Checking for module 'libusb-1.0'
--   Found libusb-1.0, version 1.0.22
```

```
sudo make install
```

```
pi@raspberrypi:~/rtl-sdr-blog/build $ sudo make install
Scanning dependencies of target rtlsdr_shared
[  3%] Building C object src/CMakeFiles/rtlsdr_shared.dir/librtlsdr.c.o
[  6%] Building C object src/CMakeFiles/rtlsdr_shared.dir/tuner_e4k.c.o
[  9%] Building C object src/CMakeFiles/rtlsdr_shared.dir/tuner_fc0012.c.o
[ 12%] Building C object src/CMakeFiles/rtlsdr_shared.dir/tuner_fc0013.c.o
[ 15%] Building C object src/CMakeFiles/rtlsdr_shared.dir/tuner_fc2580.c.o
[ 18%] Building C object src/CMakeFiles/rtlsdr_shared.dir/tuner_r82xx.c.o
[ 21%] Linking C shared library librtlsdr.so
[ 21%] Built target rtlsdr_shared
Scanning dependencies of target convenience_static
[ 25%] Building C object src/CMakeFiles/convenience_static.dir/convenience/conve
nience.c.o
[ 28%] Linking C static library libconvenience_static.a
[ 28%] Built target convenience_static
```

```
sudo cp ../rtl-sdr.rules /etc/udev/rules.d/
sudo ldconfig
```

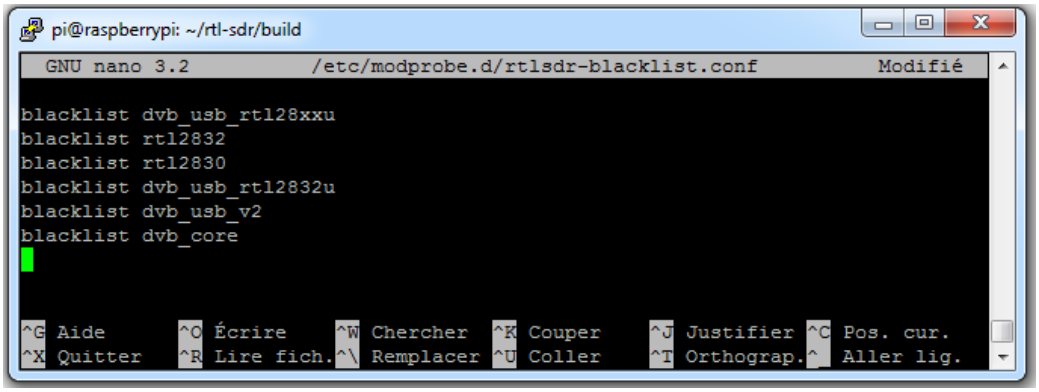
```
-- Set runtime path of "/usr/local/bin/rtl_eeprom" to ""
-- Installing: /usr/local/bin/rtl_adsb
-- Set runtime path of "/usr/local/bin/rtl_adsb" to ""
-- Installing: /usr/local/bin/rtl_power
-- Set runtime path of "/usr/local/bin/rtl_power" to ""
-- Installing: /usr/local/bin/rtl_biast
-- Set runtime path of "/usr/local/bin/rtl_biast" to ""
pi@raspberrypi:~/rtl-sdr-blog/build $ sudo cp ../rtl-sdr.rules /etc/udev/rules.d
/
pi@raspberrypi:~/rtl-sdr-blog/build $ sudo ldconfig
pi@raspberrypi:~/rtl-sdr-blog/build $
```

```
sudo nano /etc/modprobe.d/rtlsdr-blacklist.conf
```

```
pi@raspberrypi:~/rtl-sdr-blog/build $ sudo nano /etc/modprobe.d/rtlsdr-blacklist.conf
```

```
blacklist dvb_usb_rtl28xxu
blacklist rtl2832
blacklist rtl2830
blacklist dvb_usb_rtl2832u
blacklist dvb_usb_v2
blacklist dvb_core
```

Copy and paste the above list into the nano editor. Reminder right click : to paste the text in the nano editor.



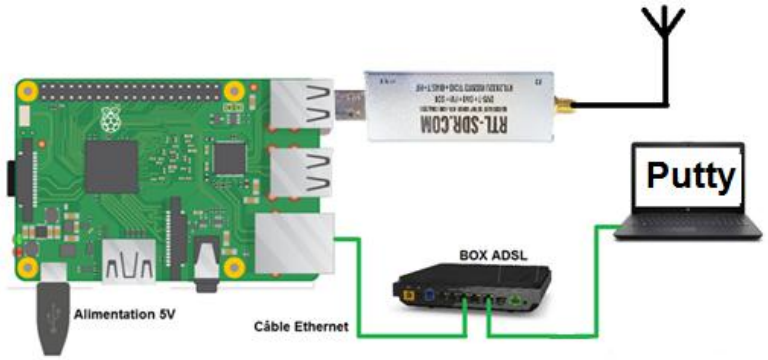
Save the file using the Ctrl + o keys, then the enter key to validate the recording.	**Exit the editor using Ctrl + x.**
	

Restart the Raspberry PI:

```
pi@raspberrypi:~/rtl-sdr-blog/build $ sudo reboot
```

```
sudo reboot
```

Verification: Plug the rtl-sdr key into a USB port on the Raspberry Pi. Connect the rtl-sdr key **to the Raspberry Pi on a free USB port.**



Check the presence of the rtl-sdr key. With LXTerminal or Putty, type the following command line:

```
lsusb
```

Locate the rtl-sdr key: **RTL238 DVB-T**

Raspberry Pi 3 :

```
pi@raspberrypi:~ $ lsusb
Bus 001 Device 004: ID 0bda:2838 Realtek Semiconductor Corp. RTL2838 DVB-T
Bus 001 Device 003: ID 0424:ec00 Standard Microsystems Corp. SMSC9512/9514 Fast
Ethernet Adapter
Bus 001 Device 002: ID 0424:9514 Standard Microsystems Corp. SMC9514 Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

Raspberry Pi 4 :

```
pi@raspberrypi:~ $ lsusb
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 003: ID 0bda:2838 Realtek Semiconductor Corp. RTL2838 DVB-T
Bus 001 Device 002: ID 2109:3431 VIA Labs, Inc. Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

With LXTerminal or Putty, type the following command line : (be careful, underscore key)

```
rtl_test
```

The rtl-sdr key should be recognized, if not, disconnect and reconnect the key and restart the test.

```
pi@raspberrypi:~ $ rtl_test
Found 1 device(s):
  0:  Realtek, RTL2838UHIDIR, SN: 00000001

Using device 0: Generic RTL2832U OEM
Found Rafael Micro R820T tuner
Supported gain values (29): 0.0 0.9 1.4 2.7 3.7 7.7 8.7 12.5 14.4 15.7 16.6 19.7
 20.7 22.9 25.4 28.0 29.7 32.8 33.8 36.4 37.2 38.6 40.2 42.1 43.4 43.9 44.5 48.0
 49.6
[R82XX] PLL not locked!
Sampling at 2048000 S/s.

Info: This tool will continuously read from the device, and report if
samples get lost. If you observe no further output, everything is fine.

Reading samples in async mode...
^CSignal caught, exiting!
User cancel, exiting...
Samples per million lost (minimum): 0
pi@raspberrypi:~ $ ^C
```

ctrl+c

Exit the program by pressing the Ctrl key and the c key at the same time. (Ctrl+c)

10.3 Rtl_tcp server with 4 SDR softwares

10.3.1 Sdrsharp

Install the Sdrsharp software described in tutorial 9.2.3, page 7. This time the RTL_SDR receiver key is connected to the Raspberry pi. The goal is to access it via the local network. It is used in two steps:

- Running the rtl_tcp server on the Raspberry PI;
- Connection on the Raspberry PI with Sdrsharp software.

Identify the IP address of the Raspberry PI:

ifconfig

```
pi@raspberrypi:~ $ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.55  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::8a7a:b574:ec19:dbce  prefixlen 64  scopeid 0x20<link>
        ether b8:27:eb:52:7f:d4  txqueuelen 1000  (Ethernet)
        RX packets 1055  bytes 146517 (143.0 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 238  bytes 32568 (31.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

My RPI IP address is: **192.168.1.55**

This command with a "dummy" option allows you to obtain all possible server configurations related to the Rtl-sdr key.

rtl_tcp -h

```
pi@raspberrypi:~ $ rtl_tcp -h
rtl_tcp: invalid option -- 'h'
rtl_tcp, an I/Q spectrum server for RTL2832 based DVB-T receivers

Usage:  [-a listen address]
        [-p listen port (default: 1234)]
        [-f frequency to tune to [Hz]]
        [-g gain (default: 0 for auto)]
        [-s samplerate in Hz (default: 2048000 Hz)]
        [-b number of buffers (default: 15, set by library)]
        [-n max number of linked list buffers to keep (default: 500)]
        [-d device index (default: 0)]
        [-P ppm_error (default: 0)]
        [-T enable bias-T on GPIO PIN 0 (works for rtl-sdr.com v3 dongles)]
        [-D enable direct sampling (default: off)]
pi@raspberrypi:~ $
```

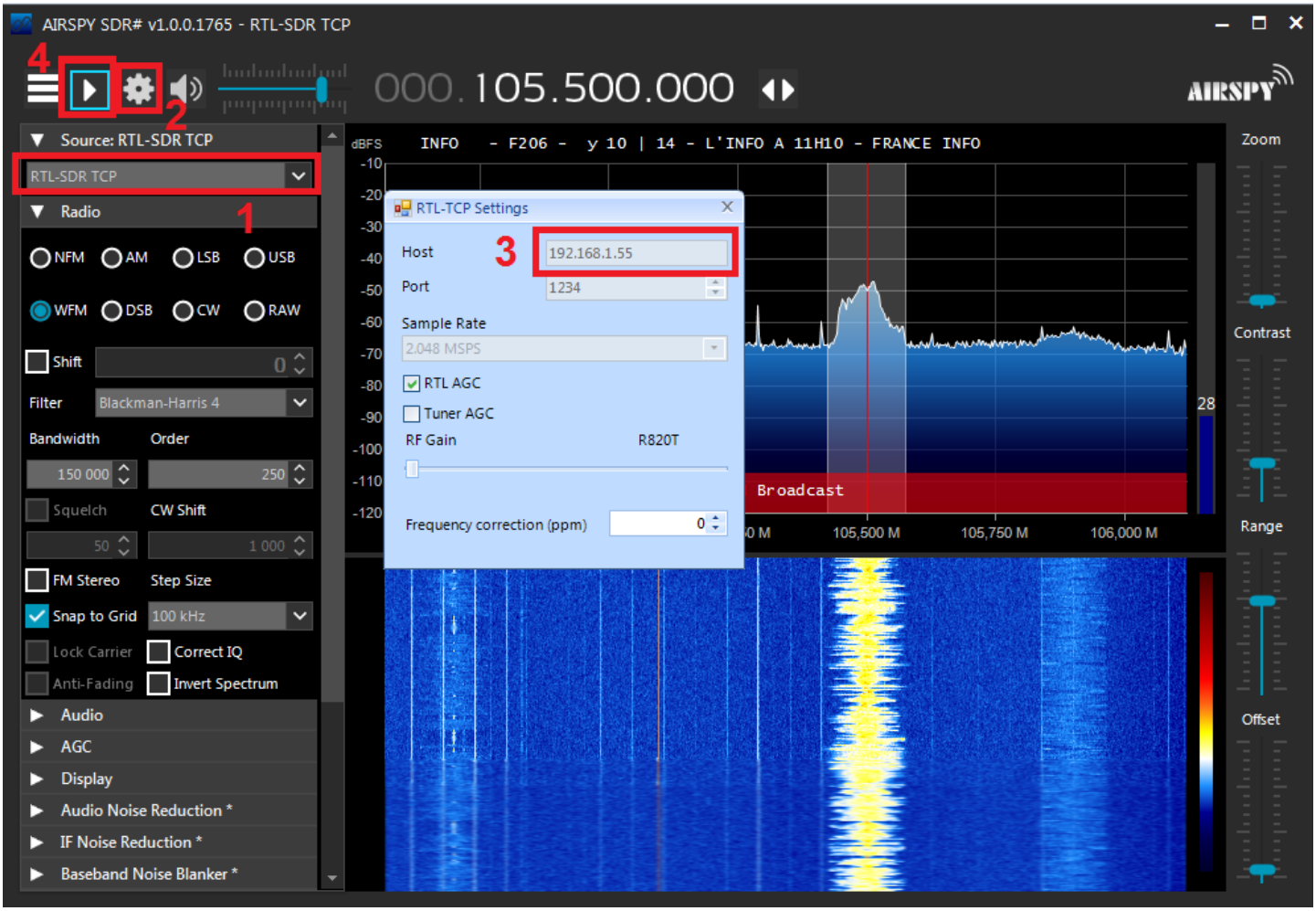
Listening in VHF and UHF

Execute the following command, making sure to replace the IP address of the Raspberry Pi with your own.

```
rtl_tcp -a 192.168.1.55
```

```
pi@raspberrypi:~ $ rtl_tcp -a 192.168.1.55
Found 1 device(s):
  0:  Realtek, RTL2838UHIDIR, SN: 00000001

Using device 0: Generic RTL2832U OEM
Found Rafael Micro R820T tuner
[R82XX] PLL not locked!
Tuned to 100000000 Hz.
listening...
Use the device argument 'rtl_tcp=192.168.1.55:1234' in OsmoSDR (gr-osmosdr) source
to receive samples in GRC and control rtl_tcp parameters (frequency, gain, ...).
```

Run the Sdrsharp software and perform the following 4 steps in order.



Listening in HF

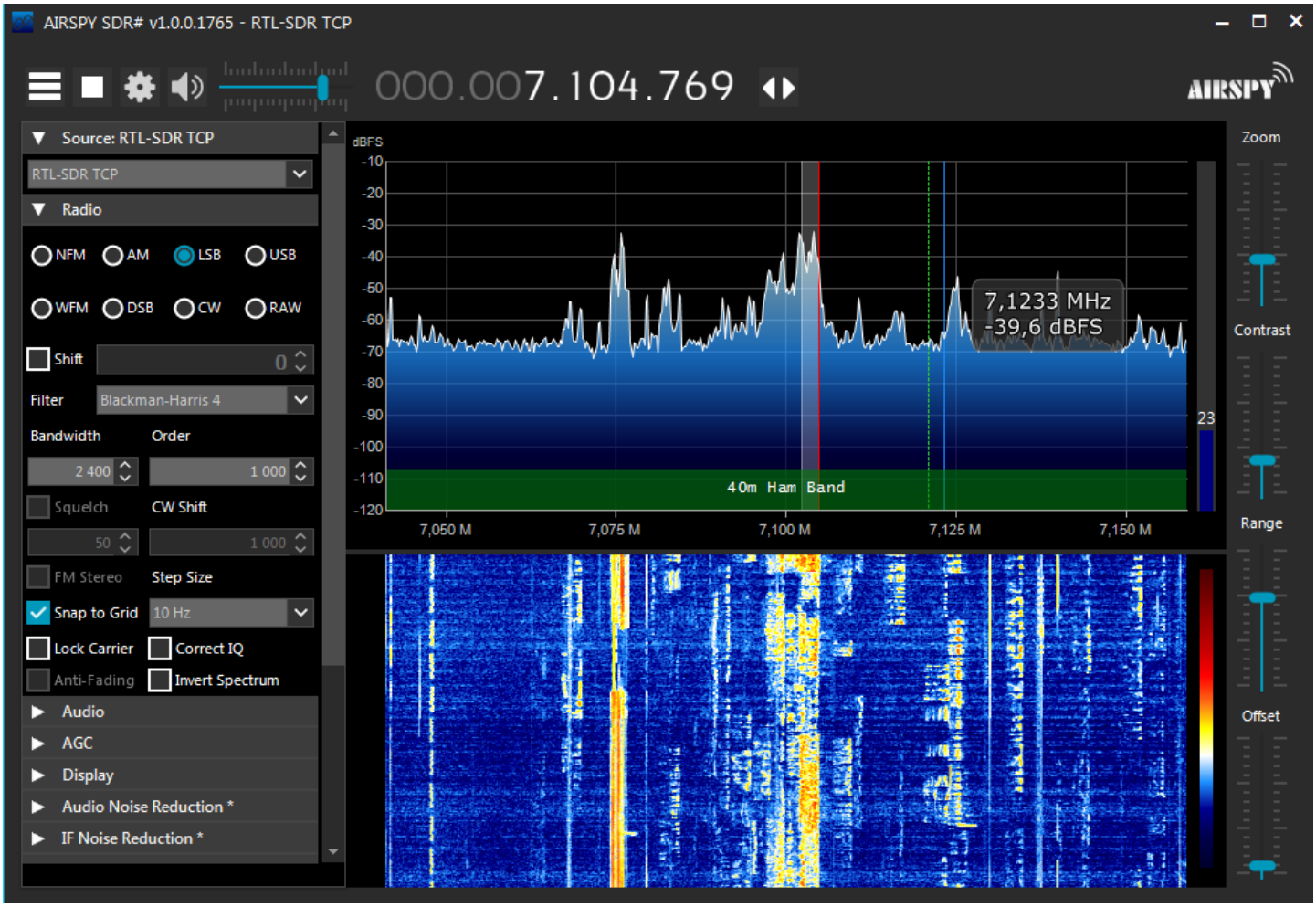
Stop listening on Sdrsharp  . Then stop the server with CTRL+C.

```
Use the device argument 'rtl_tcp=192.168.1.55:1234' in OsmoSDR (gr-osmosdr) source
to receive samples in GRC and control rtl_tcp parameters (frequency, gain, ...).
^CSignal caught, exiting!
bye!
pi@raspberrypi:~ $
```

ctrl+c

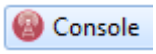
Run the following command on the RPI server (direct conversion mode), making sure to replace the Raspberry Pi IP address with your own.

```
rtl_tcp -a 192.168.1.55 -D
```

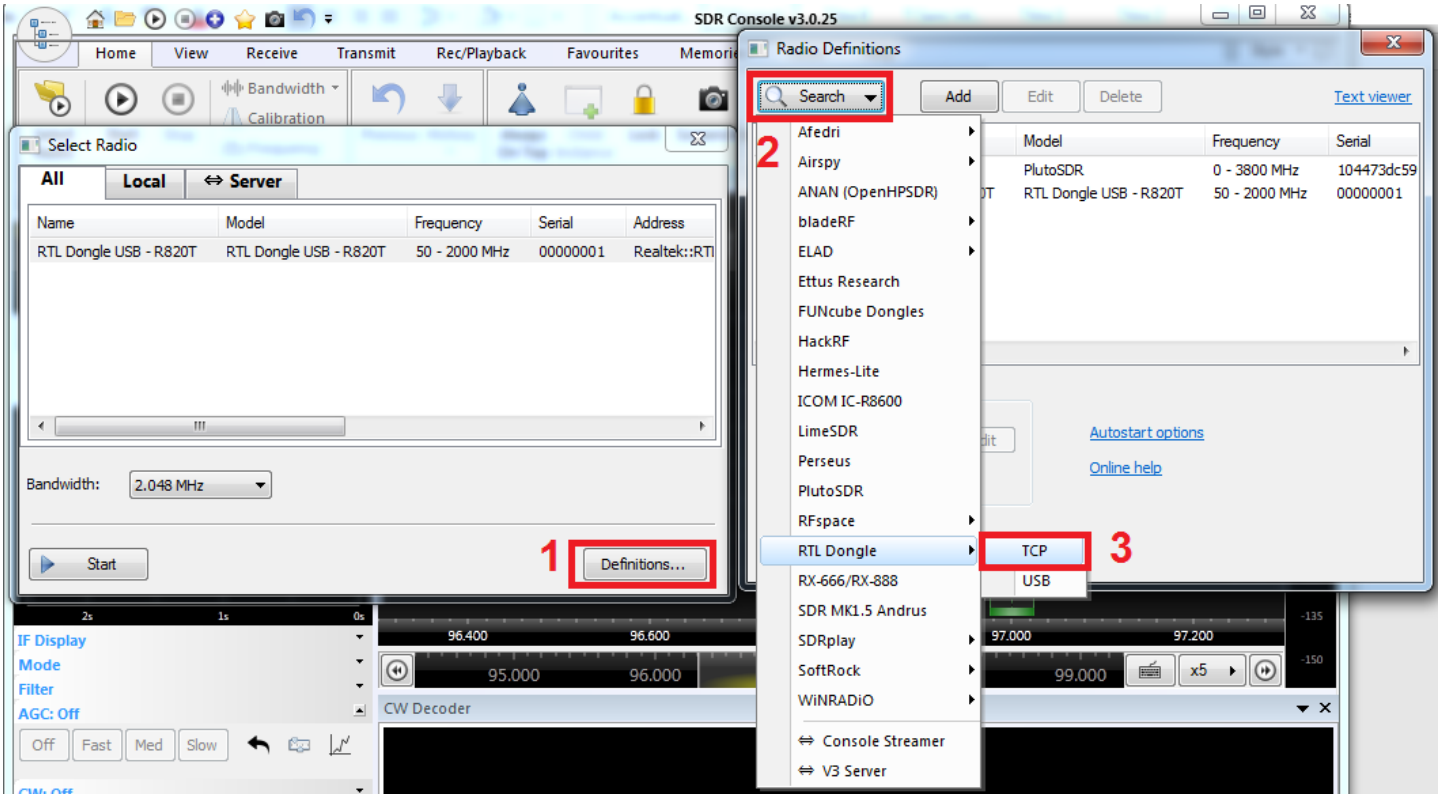


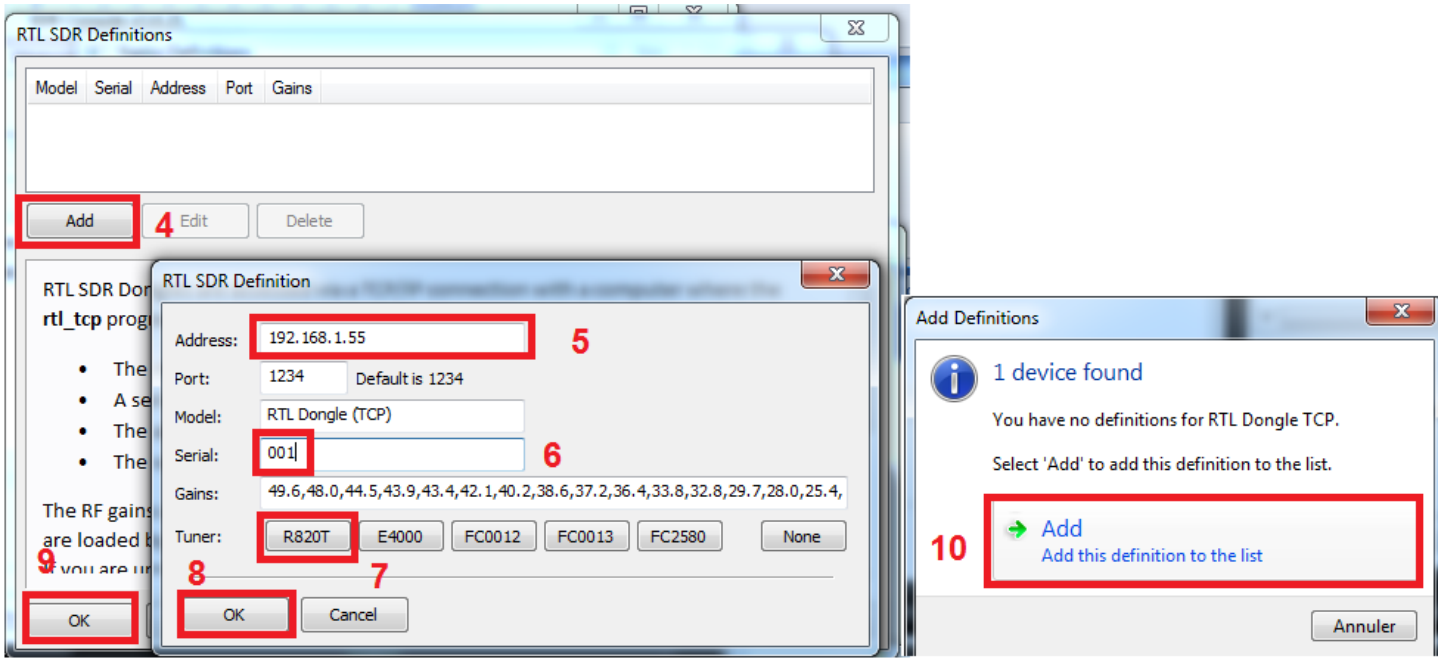
10.3.2 Sdr console

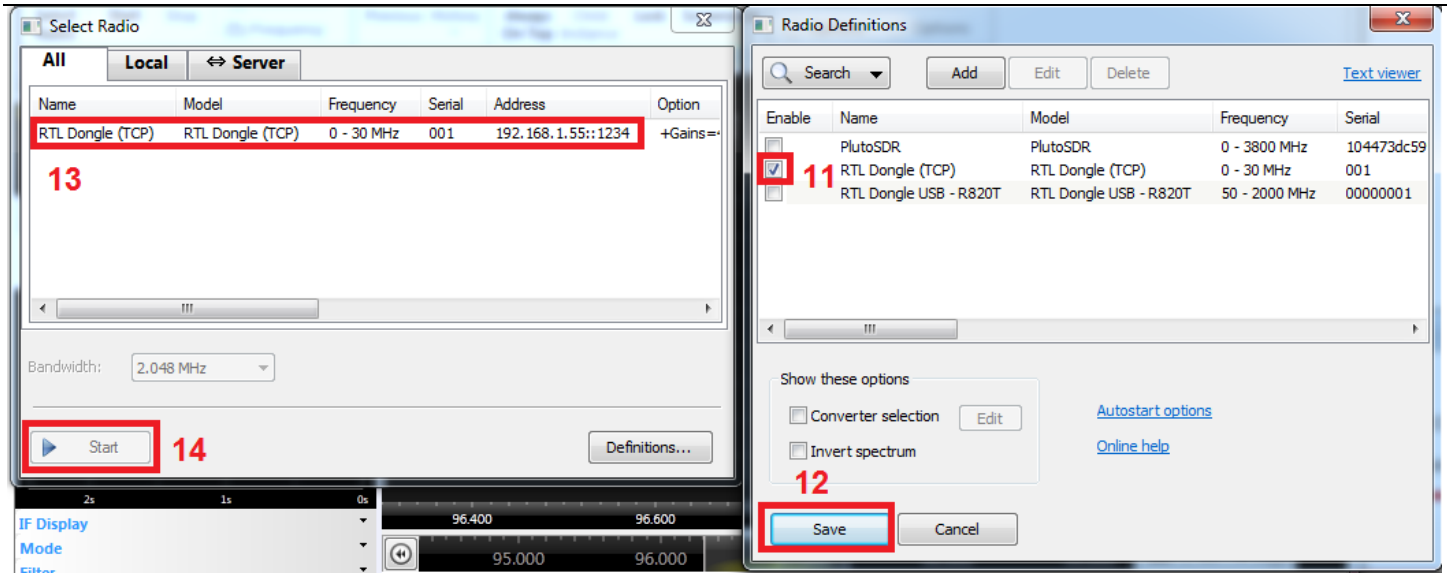
Install the Sdr console software described in tutorial 9.2.4, page 9.

Execute Sdr console 

, then choose RTL Dongle TCP







Run the following command on the RPI server, making sure to replace the IP address of the Raspberry Pi with your own.

```
rtl_tcp -a 192.168.1.55
```

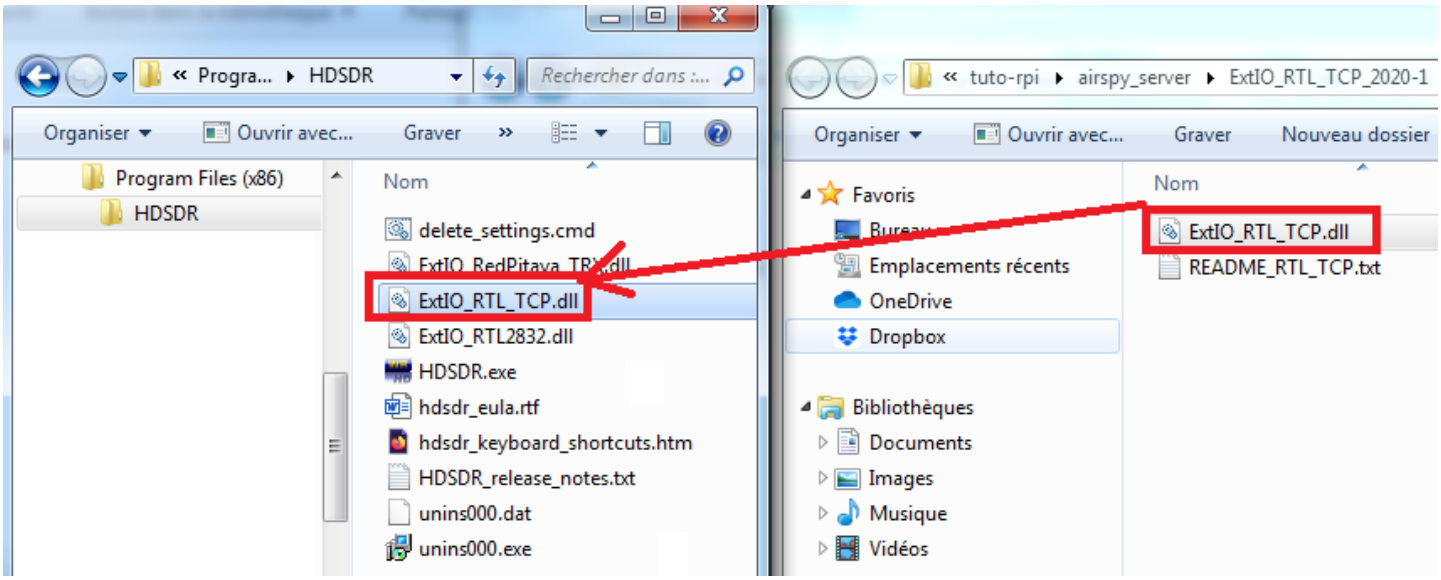
Click Start to start listening. The procedure is identical for a listening in HF with the option **-D**

10.3.3 Hdsdr

Install the Sdr console software described in tutorial 9.2.2, page 4.

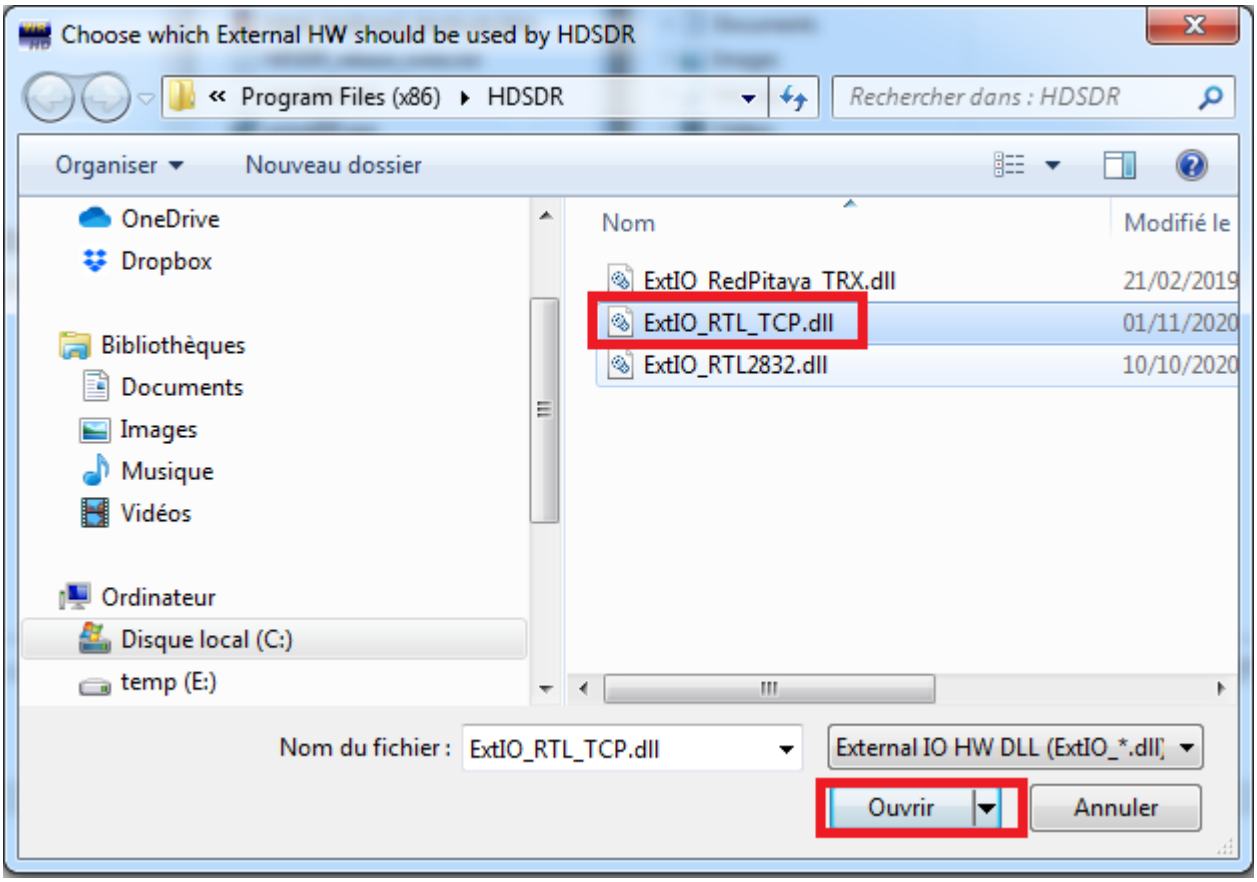
Download the file [ExtIO_RTL_TCP_2020-1.zip]

Copy the ExtIO_RTL_TCP.dll file to the HDSDR installation directory.

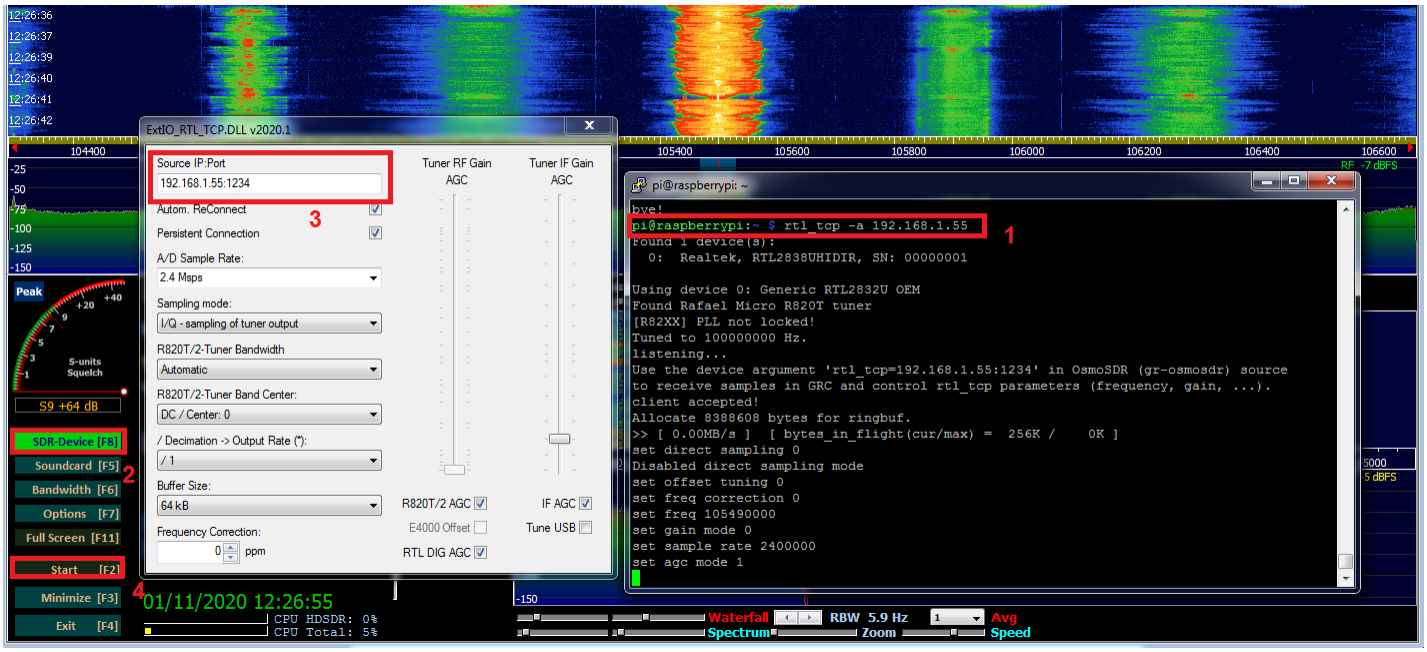


Then Run the HDSDR program HDSDR

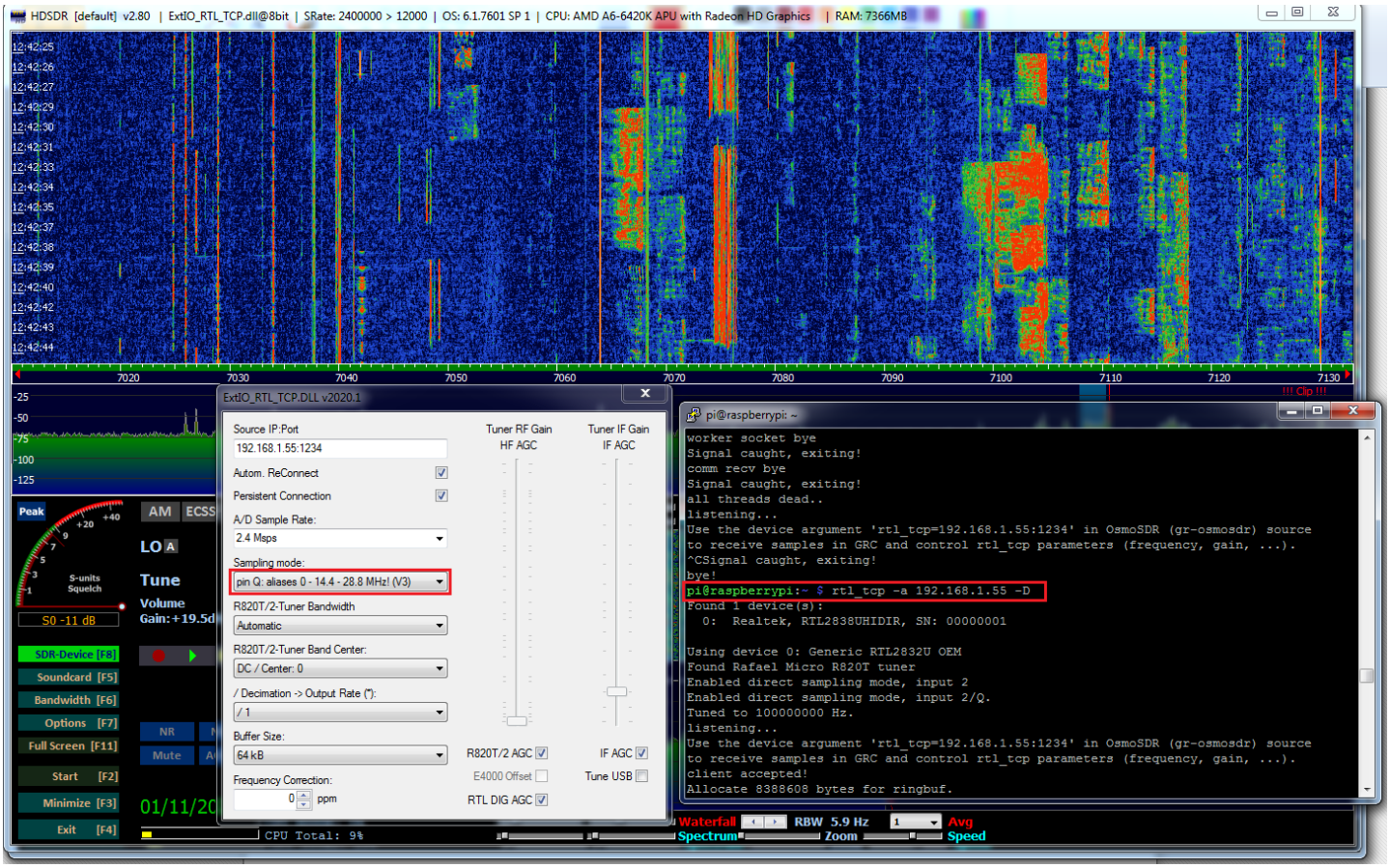
Select the file ExtIO_RTL_TCP.dll



Stop listening by pressing Stop [F2], then enter the correct IP address in SDR Device [F8]. Once configured press Start [F2].



For an **HF listening**, it will be necessary to change the sampling mode to Q

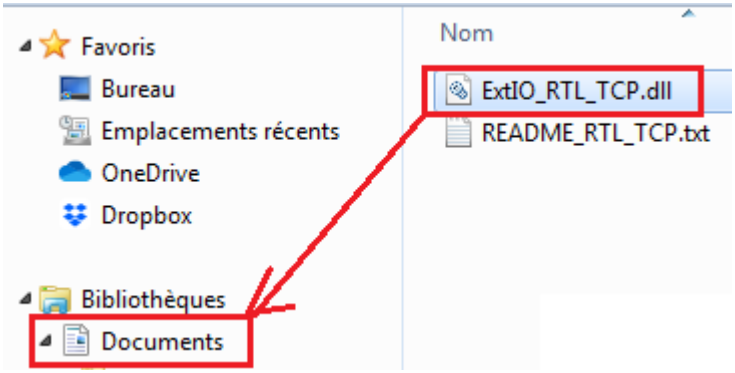


10.3.4 Sdruno

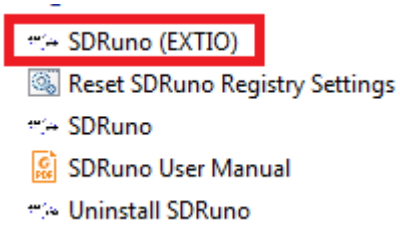
Install the Sdruno software described in Tutorial 9.2.5, page 12.

Download the file [ExtIO_RTL_TCP_2020-1.zip]

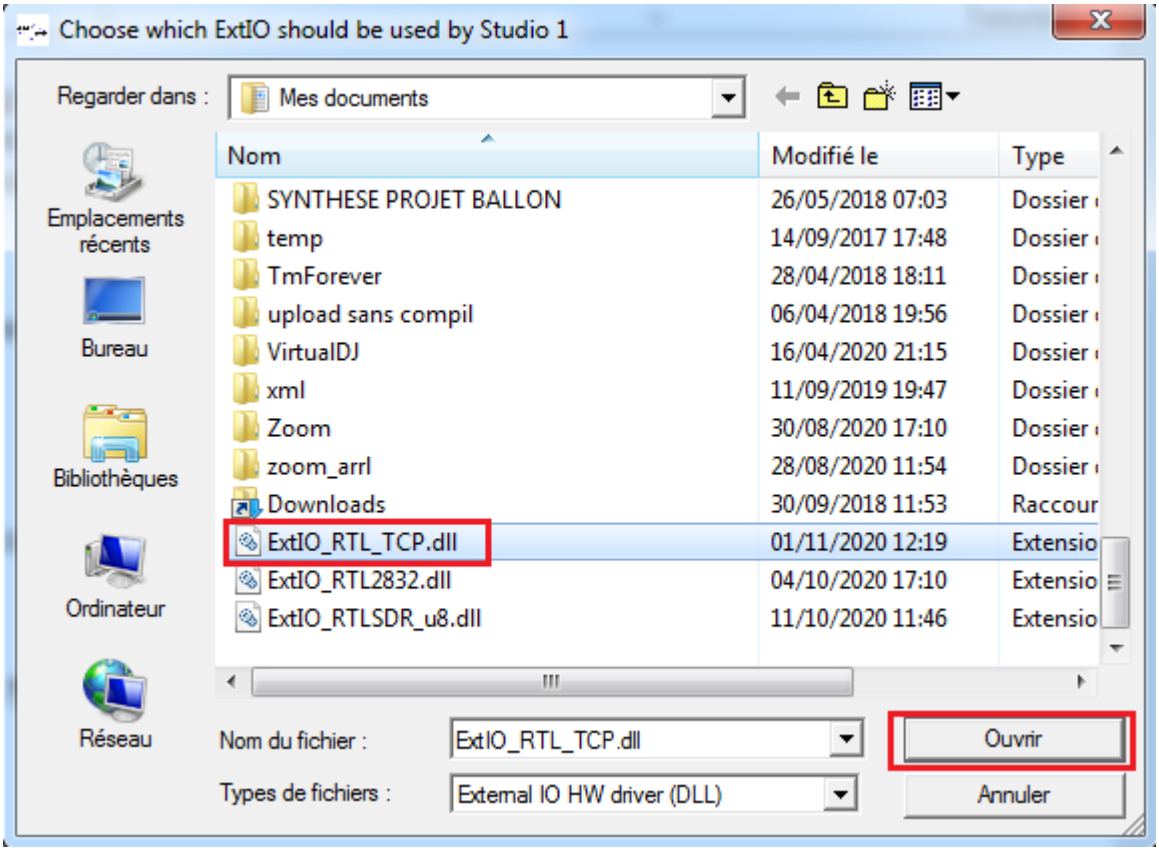
Copy the file `ExtIO_RTL_TCP.dll` to the Documents directory

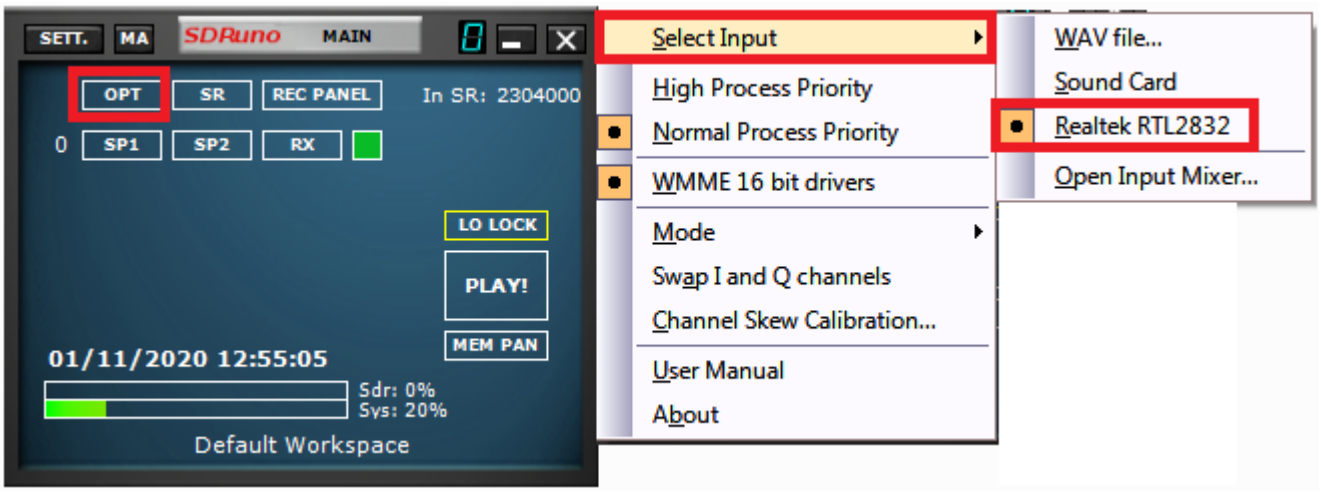


Execute the SDRuno (Extio) file, installed on the hard disk. (Search when you don't know where the program was installed). Note that there is a manual in pdf (SDRuno User manual) and it is the only manual installed with the program among the 4 softwares presented here.

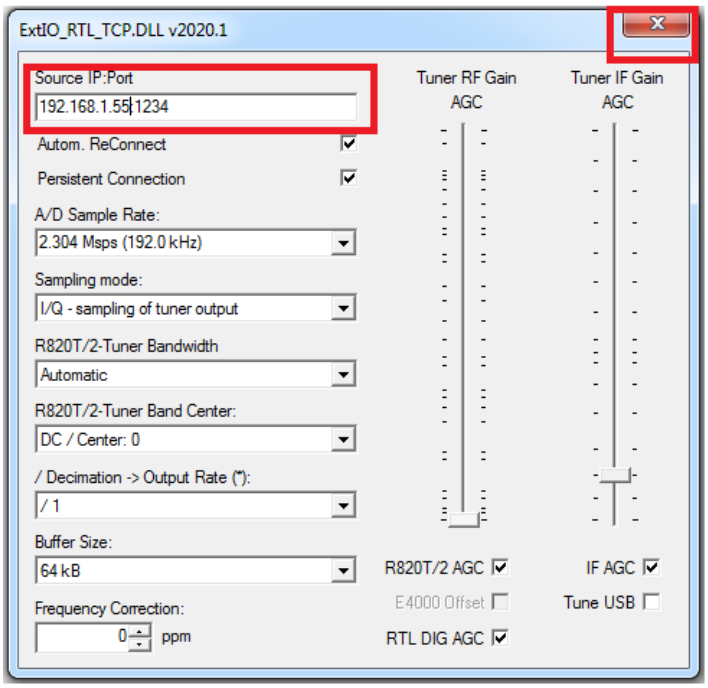


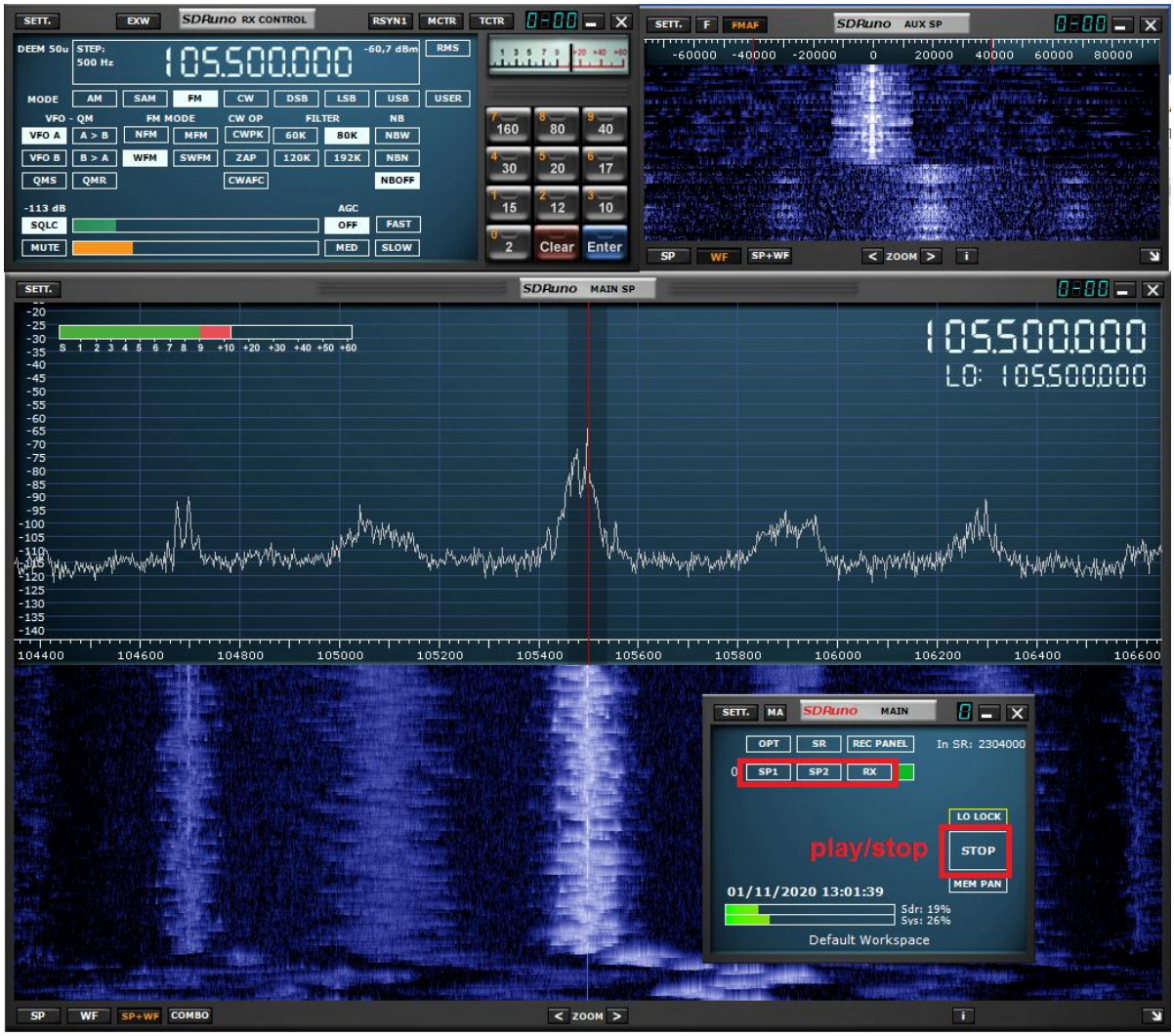
Check that the `ExtIO_RTL_TCP.dll` file is correctly taken into account so that the software can access the rtl-sdr key.





Change the IP address, making sure to use yours.

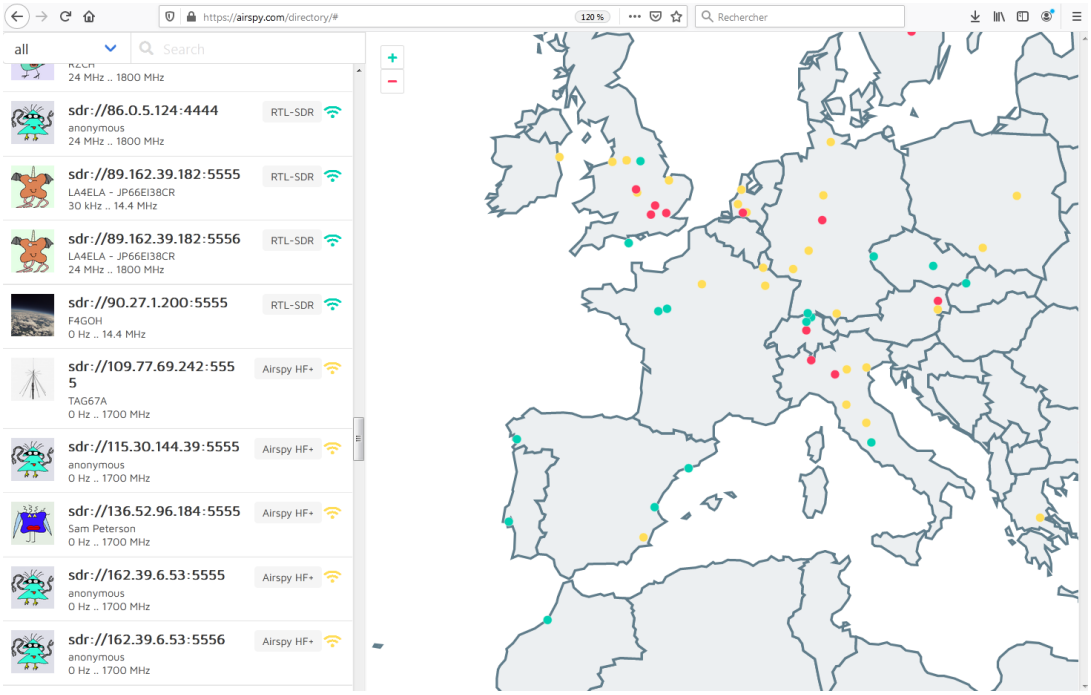




To date I have not been able to use rtl_tcp with SDRuno in HF reception (direct sampling mode).

10.4 Installing the SPY Server

Spyserver works a little bit like rtl_tcp, but with the addition of the ability to share your listening station. This one appears on the world map. All you have to do is copy the IP address and port into your favorite SDR software and listen to it.



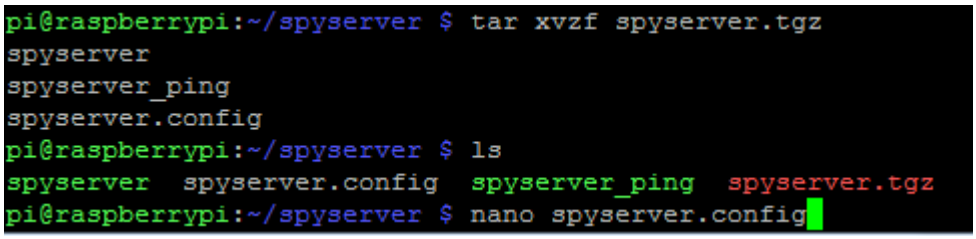
With the Putty utility, type these commands in the following order:

```
mkdir spyserver
cd spyserver/
wget -O spyserver.tgz http://airspy.com/?ddownload=4247
ls
```

```
pi@raspberrypi:~ $ mkdir spyserver
pi@raspberrypi:~ $ cd spyserver/
pi@raspberrypi:~/spyserver $ wget -O spyserver.tgz http://airspy.com/?ddownload=4247
--2020-11-01 15:33:24--  http://airspy.com/?ddownload=4247
Résolution de airspy.com (airspy.com)… 172.67.138.28, 104.27.167.24, 104.27.166.24
Connexion à airspy.com (airspy.com)|172.67.138.28|:80… connecté.
requête HTTP transmise, en attente de la réponse… 301 Moved Permanently
Emplacement : https://airspy.com/?ddownload=4247 [suivant]
--2020-11-01 15:33:24--  https://airspy.com/?ddownload=4247
Connexion à airspy.com (airspy.com)|172.67.138.28|:443… connecté.
requête HTTP transmise, en attente de la réponse… 302 Found
Emplacement : /downloads/spyserver-arm32.tgz [suivant]
--2020-11-01 15:33:26--  https://airspy.com/downloads/spyserver-arm32.tgz
Réutilisation de la connexion existante à airspy.com:443.
requête HTTP transmise, en attente de la réponse… 200 OK
Taille : 63778 (62K) [application/x-gzip]
Sauvegarde en : « spyserver.tgz »

spyserver.tgz         100%[=========================>]  62,28K  --.-KB/s    ds 0,1s

2020-11-01 15:33:26 (563 KB/s) – « spyserver.tgz » sauvegardé [63778/63778]

pi@raspberrypi:~/spyserver $ ls
spyserver.tgz
pi@raspberrypi:~/spyserver $
```

Unzip the file, then edit the configuration file `spyserver.config`

```
tar xvzf spyserver.tgz
ls
nano spyserver.config
```



Configure your server using the example below:

```
# SPY Server Configuration File

# TCP Listener

bind_host = 192.168.1.55
bind_port = 5555-6666

# List Server in Airspy Directory

list_in_directory = 1

# Owner Name
# For example: John Doe L8ZEE
owner_name =f4goh

# Owner email
# For example: john@doe.com
owner_email =f4goh@orange.fr

# Antenna
# For example: Random Wire/Magnetic Loop/Mini-Whip/Inverted V/etc.
antenna_type =dipole

# Antenna Location
# For example: 48.858332, 2.294560
antenna_location =47.89,0.27

# General Description

general_description = test rx

# User sessions

maximum_clients = 1

# Maximum session duration
# In minutes. 0 for no limit

#maximum_session_duration = 30

# Allow clients to retune and change of gain of the device
#
allow_control = 1

# Device Type
# Possible Values:
#   AirspyOne (R0, R2, Mini)
#   AirspyHF+
#   RTL-SDR
#   Auto      (Scans for the first available device)

device_type = RTL-SDR

# Device Serial Number as 64bit Hex
# For example: 0xDD52D95C904534AD
# A value of 0 will acquire the first available device.
#
device_serial = 0

# Device Sample Rate
# Possible Values:
#   Airspy R0, R2 : 10000000 or 2500000
#   Airspy Mini   :  6000000 or 3000000
#   Airspy HF+    :   768000
#   RTL-SDR       :   500000 to 3200000
# Comment to use the device's default

#device_sample_rate = 2500000

# Force 8bit Compression Mode
# The 8bit Compression mode has proven sufficiently good for most streaming use cases.
# Use it to same some internet bandwidth.
#
#force_8bit = 1

# Maximum Bandwidth
# Limits the maximum IQ bandwidth the clients can set
# Recommended value for WFM is 200000
# Recommended value for narrow band modes is 15000
#
#maximum_bandwidth = 15000

# FFT Frames Per Second
fft_fps = 15
# FFT Bins
# Bins = 2^fft_bin_bits

fft_bin_bits = 16

# Initial Center Frequency

initial_frequency = 7100000

# Minimum Tunable Frequency
# Comment if using the device default
#
#minimum_frequency = 0

# Maximum Tunable Frequency
# Comment if using the device default

#maximum_frequency = 35000000

# Frequency Correction in PPB
#
#frequency_correction_ppb = 0

# Initial Gain
#
#initial_gain = 5

# RTL-SDR Sampling mode
# Quadrature = 0,
# Direct Sampling I Branch = 1
# Direct Sampling Q Branch = 2
rtl_sampling_mode = 2

# Converter Offset
# Set to -120000000 to enable the SpyVerter offset
#converter_offset = -120000000

# Bias-Tee
# For AirspyOne only - Useful for LNA's and SpyVerter
#enable_bias_tee = 0

# Buffer Size (in milliseconds)

buffer_size_ms = 50

# Buffer Count

buffer_count = 10
```

In the configuration example rtl_sampling_mode = 2 for HF listening. For VHF/UHF listening, just add the # character to comment the line. (#rtl_sampling_mode = 2).

Run the server.

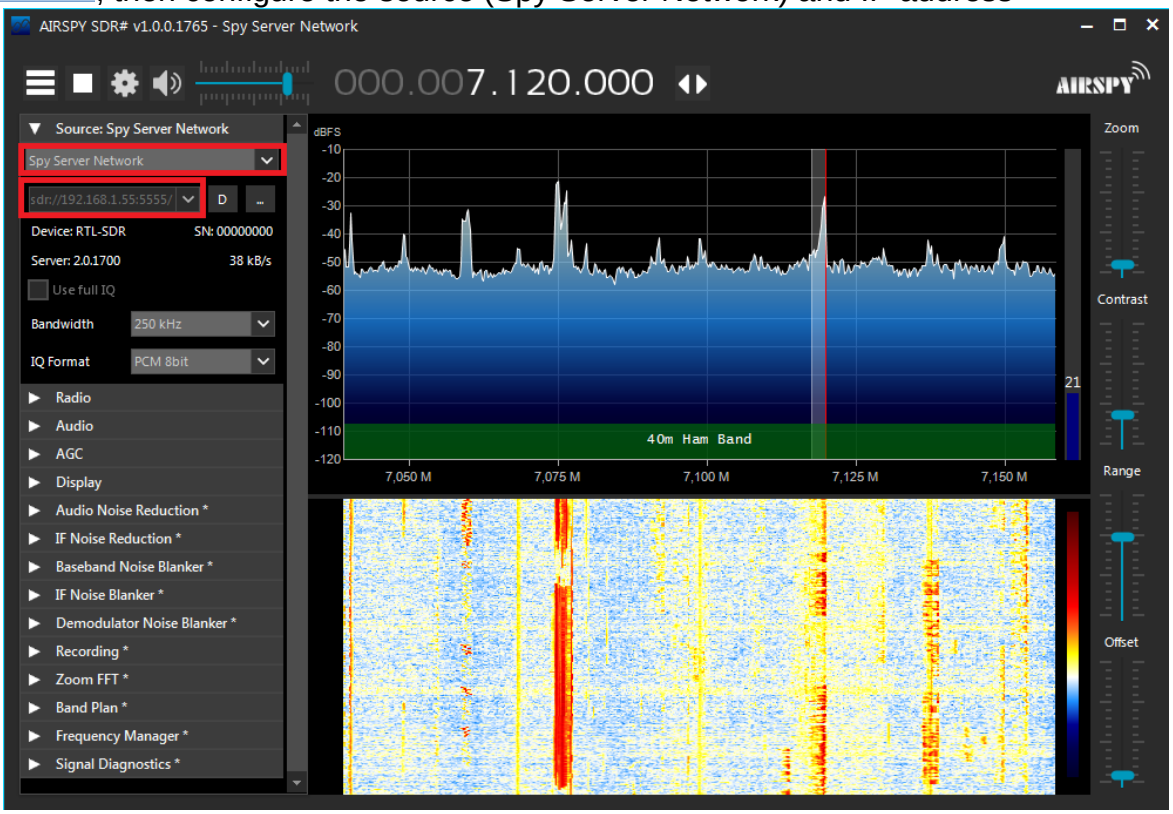
```
./spyserver
```

```
pi@raspberrypi:~/spyserver $ ./spyserver
SPY Server v2.0.1700
Copyright (C) 2016-2018 Youssef Touil - https://airspy.com
Reading the configuration file: spyserver.config
Listening for connections on 192.168.1.55:5555
Found Rafael Micro R820T tuner
[R82XX] PLL not locked!
Enabled direct sampling mode, input 2
Accepted client 192.168.1.13:26511 running SDR# v1.0.0.1765 on Microsoft Windows
 NT 6.1.7601 Service Pack 1
Device was sleeping. Wake up!
Found Rafael Micro R820T tuner
[R82XX] PLL not locked!
Enabled direct sampling mode, input 2
Acquired an RTL-SDR device
```

10.5 Test with 2 client softwares

10.5.1 Sdrsharp

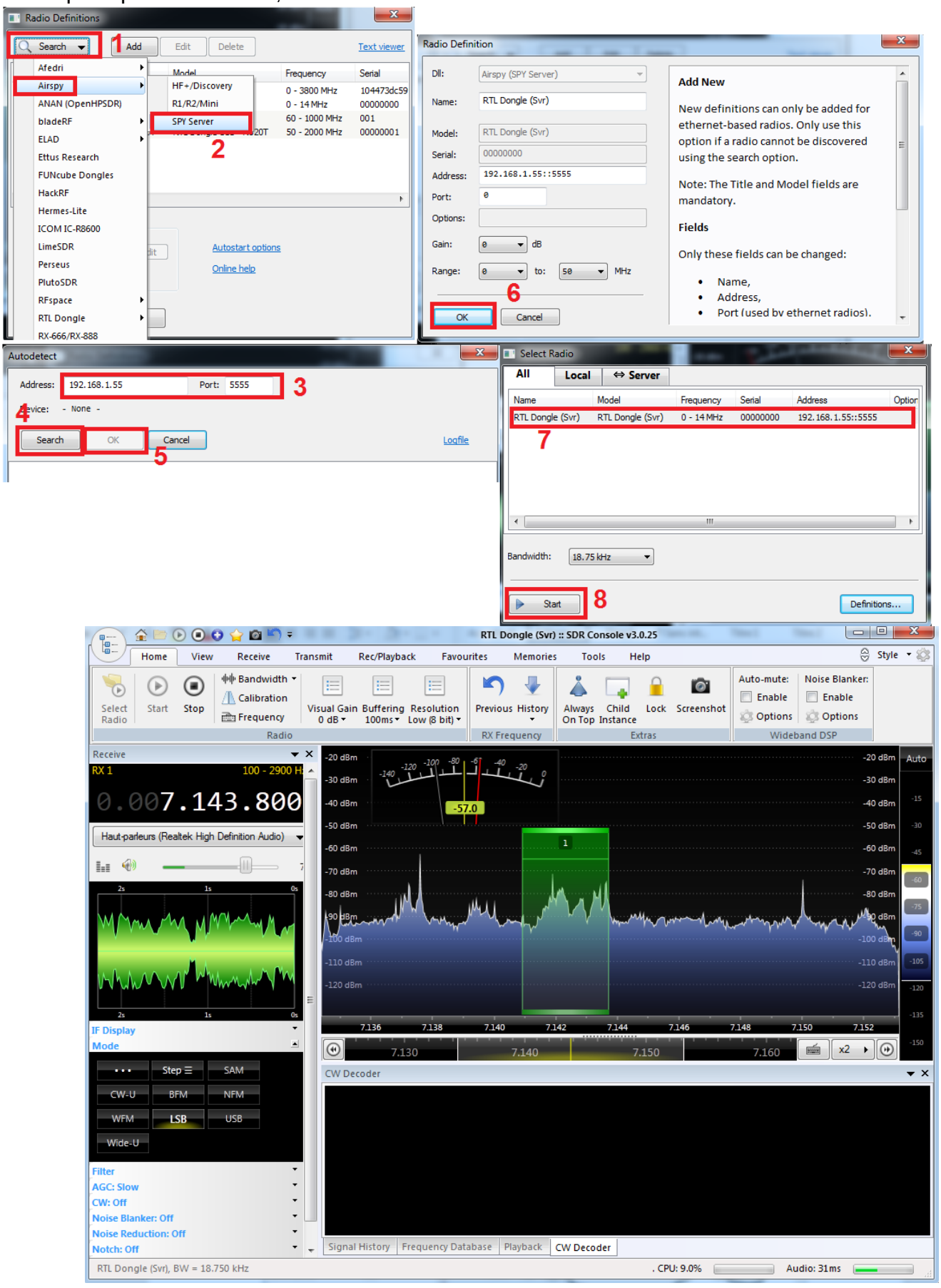
Run SDRSharp.exe, then configure the source (Spy Server Network) and IP address



In this configuration the listening is on the local network. To be able to access from the Internet it is necessary to configure the ADSL BOX.

10.5.2 Sdr console

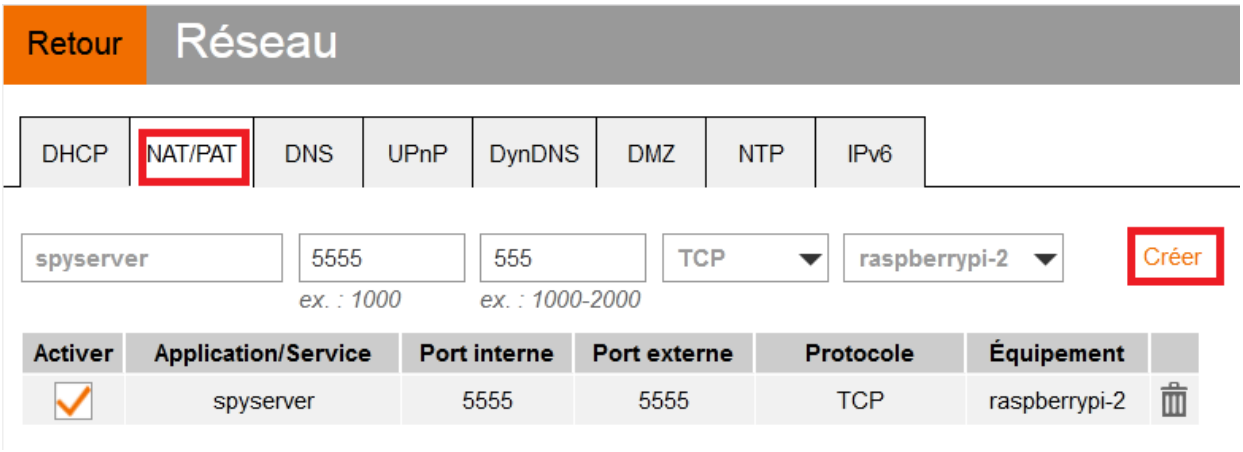
Same principle as in 10.3.2, but with SPY Server.



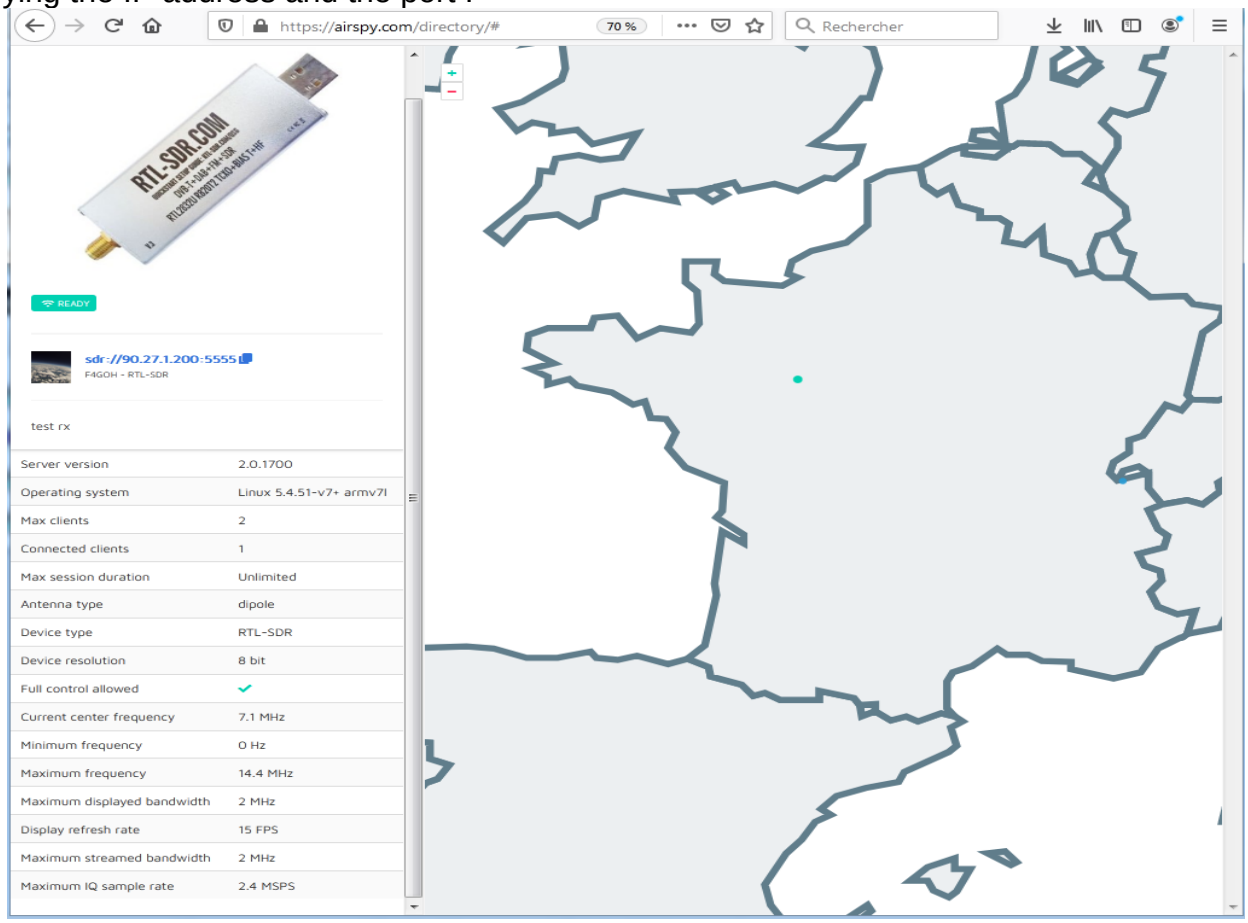
10.6 ADSL box Configuration

The configuration is identical to the tutorial 6.8: Accessing the web SDR from outside the QRA.

The difference is in the port number (5555).

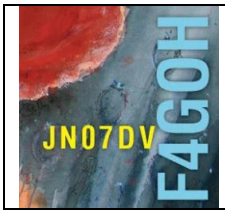


On the world map, my station appears, the "ready" icon is in green, it is possible to access my station by copying the IP address and the port :

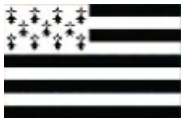


10.7 Conclusion

SDR reception sharing is very easy to achieve with a Raspberry pi. This will be able to render many services in various test situations. I thank Ladislav **OK1UNL** for giving me the idea of this tutorial. Linux users will be able to use GQRX as a client for the server.



Raspberry Pi for HAM Radio
Part 11



Summary:

Part 11: Remote station with HamVoIP

Reminder: **Raspberry Pi = RPI**

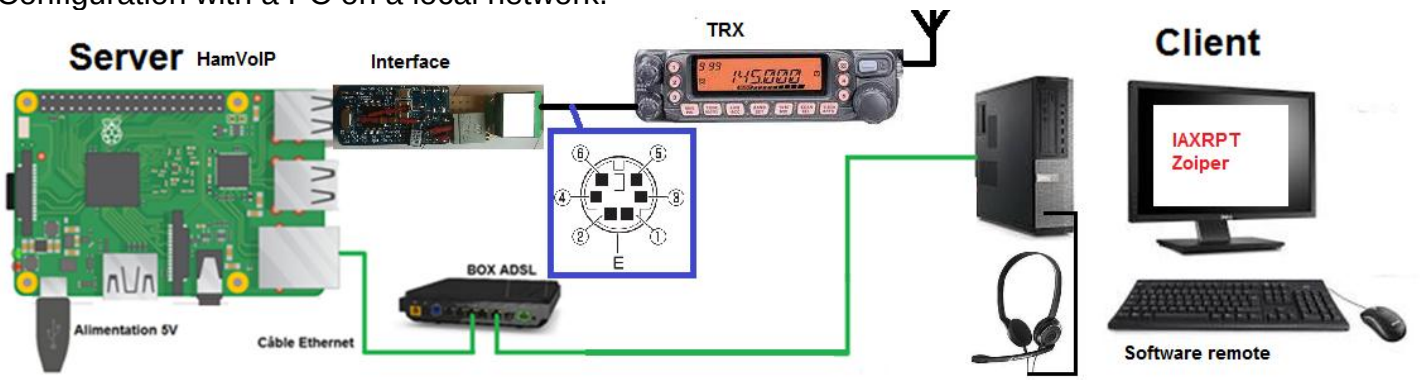
Prerequisite: Part 8: image disk creation on mSd card.

Version du 9/5/2021 V1.0 on RPI4

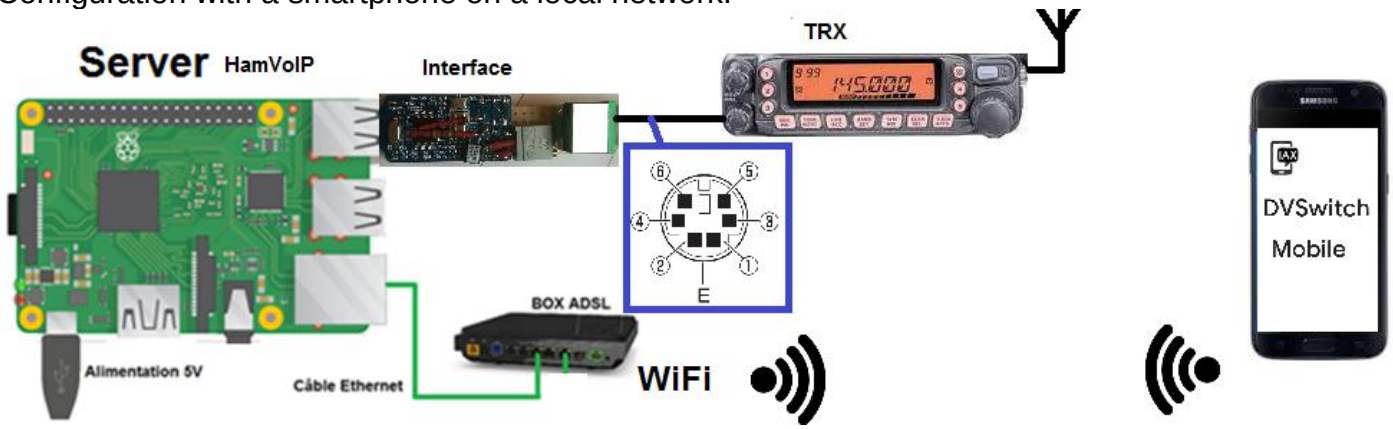
11.1 Server/client set Presentation

The objective is to be able to perform a remote QSO with a classic mobile TRX located at the QRA via a local network or the Internet in a private setting. This low-cost solution only requires a modified sound card located between the Rapsberry Pi and the TRX and the pre-installed HAMVOIP Asterisk Allstar software suite. The connection with the server will be done with a PC or a smartphone.

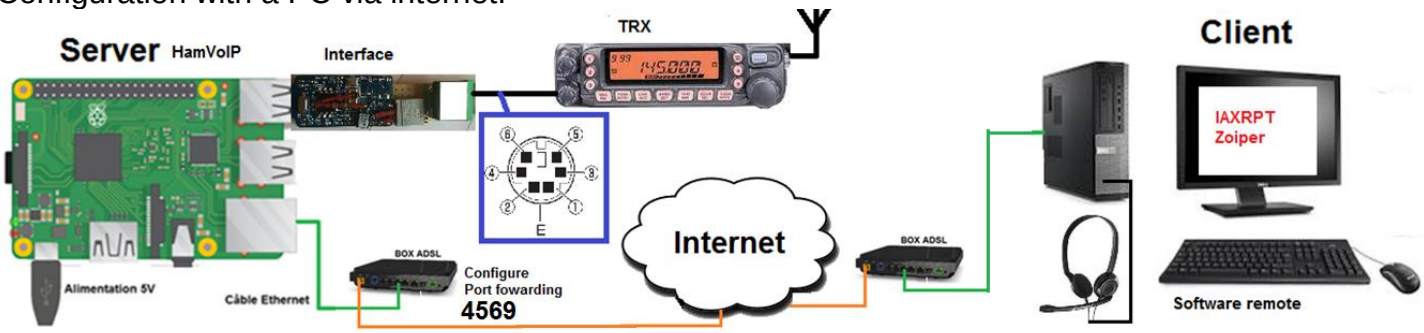
Configuration with a PC on a local network.



Configuration with a smartphone on a local network.



Configuration with a PC via internet.



However, you should remember to switch on the TRX and RPI before leaving the station.
It is not possible to change the frequency remotely. (Especially on a mobile TRX!).

For OMs who want to chain QSOs in HF (voice or digital), I recommend the https://doc.webradiocontrol.tech/ software, which has to be paid for, but which offers many possibilities with a very good man-machine interface (Possibility of testing over a period of 15 days).

11.2 VoIP protocols and software

VoIP stands for Voice over IP. VoIP is a computer technology that allows voice to be transmitted over IP-enabled networks, via the Internet or private networks.

There are several protocols that support VoIP, including:

- SIP : Session Initiation Protocol,
- IAX : Inter-Asterisk_eXchange

It is the second protocol that we use in our remote station case because HAMVOIP uses Asterisk open source software for Linux.

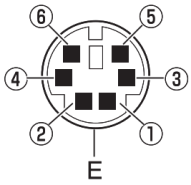
Note :

The client software installed on the PC is called iaxRpt and has the same name as the protocol.

11.3 Making the interface

The connection between the TRX and the interface is made using the mini DIN 6 pin. The pins used are 1, 2, 3, 5 and 6. A mini DIN 6 female connector for printed circuit boards can be used as shown in the print on the right.

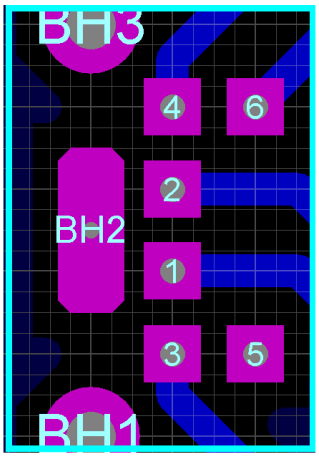
Broches de terminal de données :



Kenwood

N°	Nom	I/O	Fonction
①	PKD	Entrée	Signal audio pour transmission par paquets
②	DE	—	Mise à la terre de la borne PKD
③	PKS	Entrée	'L' est transmis et le microphone est en sourdine
④	PR9	Sortie	Signal de répétition de 9600 (bps)
⑤	PR1	Sortie	Signal de répétition de 1200 (bps)
⑥	SQC	Sortie	Signal de contrôle du silencieux; Fermé : H, Ouvert : L (Les réglages par défaut peuvent être modifiés dans Menu 520)
	E	—	Mise à la terre commune

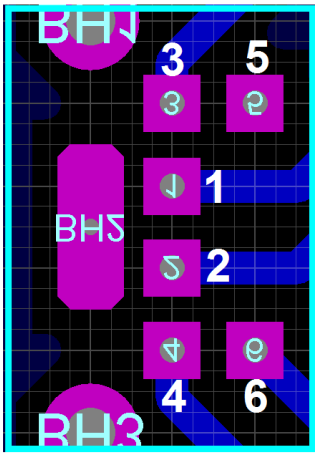
Vue de dessus



Yaesu

Goupille	Étiquette	Note	Couleur du fil CT-39
1	PKD (Données d'entrée)	Données d'entrée Packet *Impédance :* 10 Ohm-k *Niveau d'entrée maximum :* 40 pp mV pour 1200 bps 2,0 V pp pour 9600 bps	Marron
2	GND	Signal de terre	Rouge
3	PTT	Terrain à transmission	Orange
4	RX9600	Données de sortie Packet de 9600 bps *Impédance :* 10 Ohm-k, *Sortie maximale :* 500 mV pp	Jaune
5	RX1200	Données de sortie Packet de 1200 bps *Impédance :* 10 Ohm-k *Sortie maximale :* 300 mV pp	Vert
6	PSK (SQL)	Contrôle du silencieux *Silencieux ouvert :* +5 V, *Silencieux fermé :* 0 V	Bleu

Vue de dessous

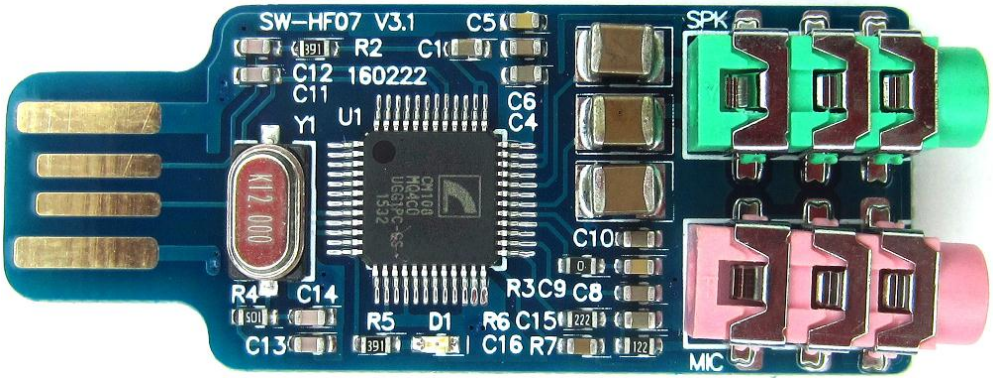


The Yaesu FT7900 documentation is much clearer than the Kenwood TM-V71.

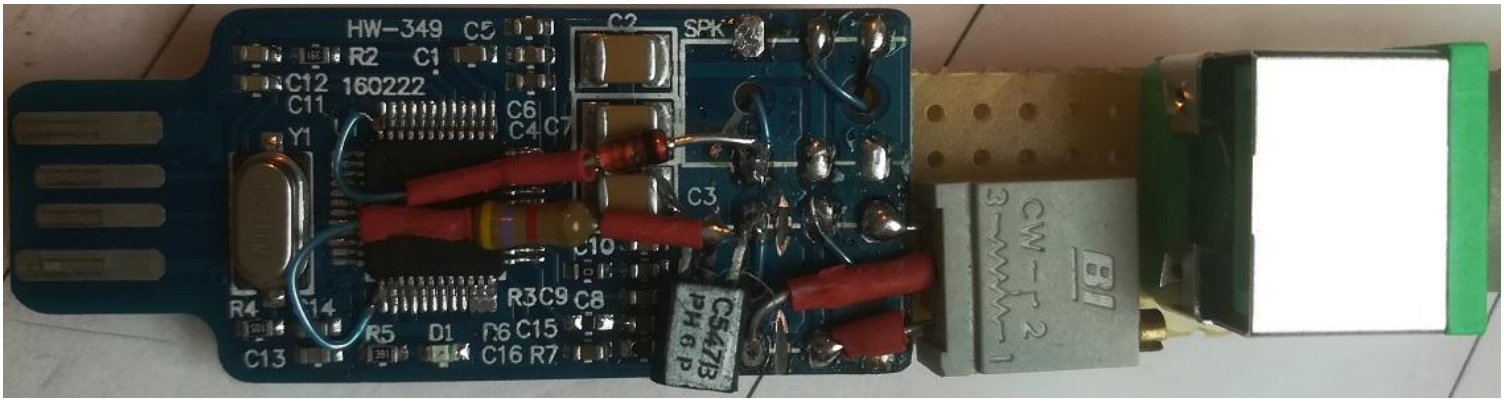
Before installing HAMVOIP, you will have to buy the CM108 USB sound card, sold without a case. Then modify this "key" by adding an NPN transistor (BC547B) to control the TRX's PTT as well as a diode (1N4148) and an adjustable 50K or 100K multi-turn resistor.

Before modifying the USB sound card, it is recommended that you test it on a PC. To do this, use headphones on the green output while listening to music, for example.

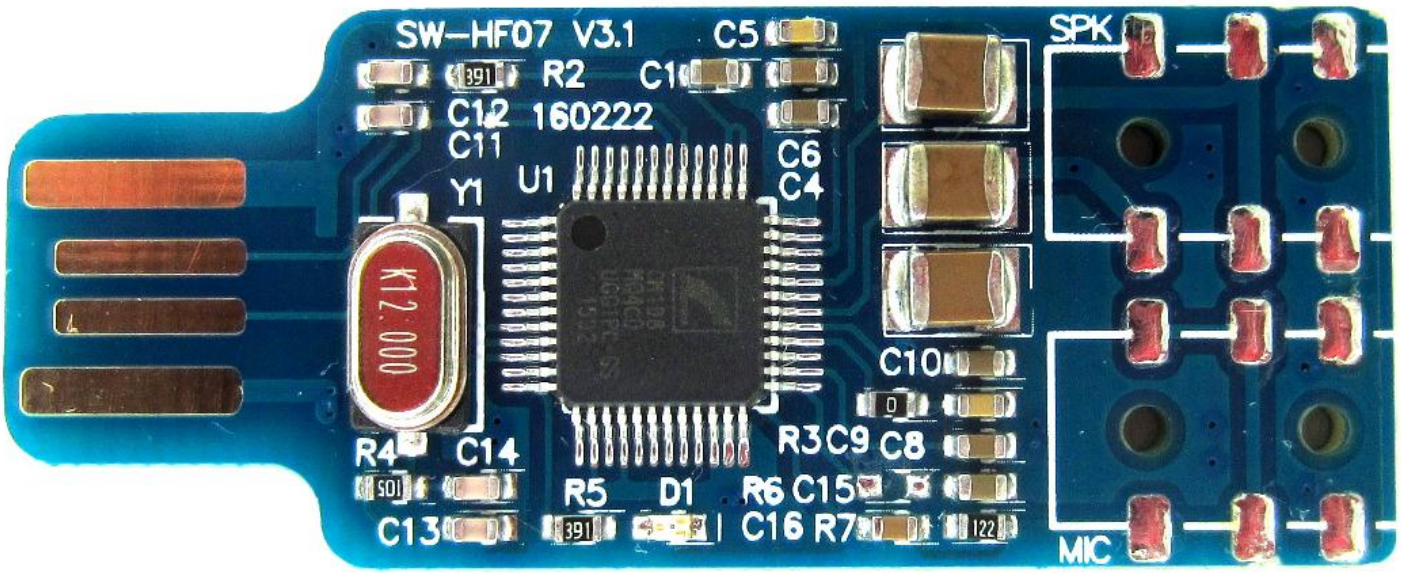
Sound card before modification:



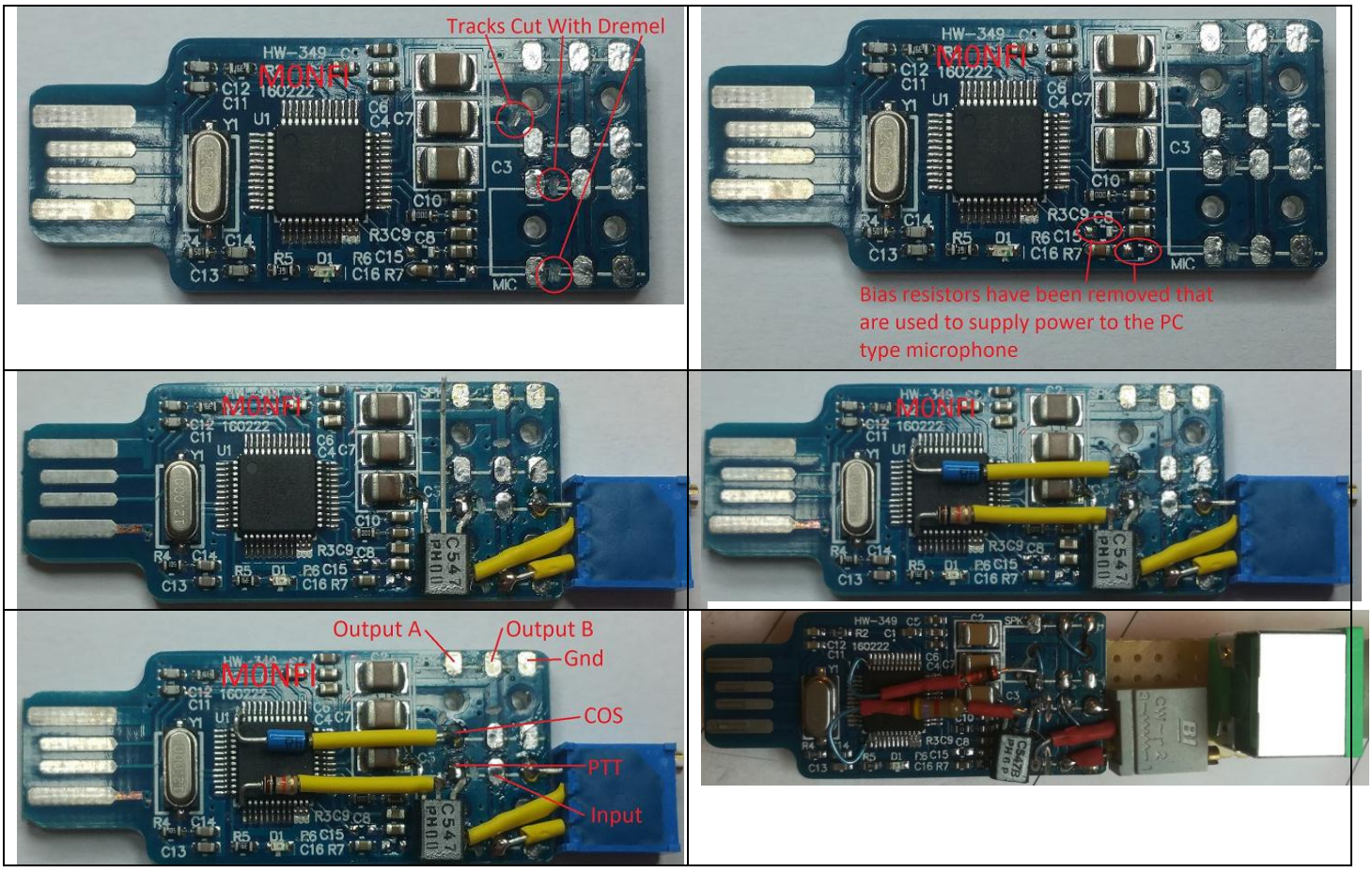
Sound card after modification :



- Start by unsoldering the 2 3.5 jack connectors



Follow the modification of the "sound card" from the blog at the following address
Tip: Preset the potentiometer to halfway before implementation.
https://www.marrold.co.uk/2018/04/converting-cm108-usb-fob-for-amateur.html



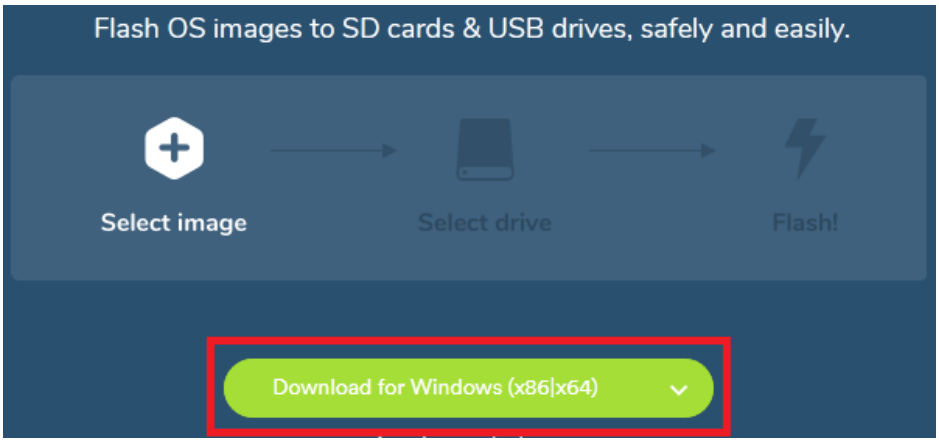
Finish by connecting the wires to the 6-pin mini din connector

11.4 Creating the disk image

Download the disk image (green box) using this link https://hamvoip.org/#download

- **RPi2-3-4 includes Pi 2,3B,3B+,3A+,and 4B Image Version 1.6-14 - Self Extracting exe for Windows (LATEST)Be sure to update when setting up! -**
  RPi2/RPi2-3-4_V1.6-14_Allstar.exe
  IMPORTANT - Download this README for the V1.6 version
- **RPi2-3-4 includes Pi 2,3B+,3A+,and 4B Image Version 1.6-14 - xz archive (LATEST) Be sure to update when setting up! -**
  RPi2/RPi2-3-4_V1.6-14_Allstar.img.xz
  IMPORTANT - Download this README for the V1.6 version
- **BBB Image (Final version) NOT RECOMMENDED FOR NEW INSTALLS! -**
  BBB/BBB_Allstar_V1.2.2.img.zip

Download and install the Etcher utility to copy the image to an MSD card
https://www.balena.io/etcher/



Select the RPi2-3-4_Vx.x-x_Allstar.img.xz file, the SD card reader, and then click on Flash. Same procedure as in tutorial 8.

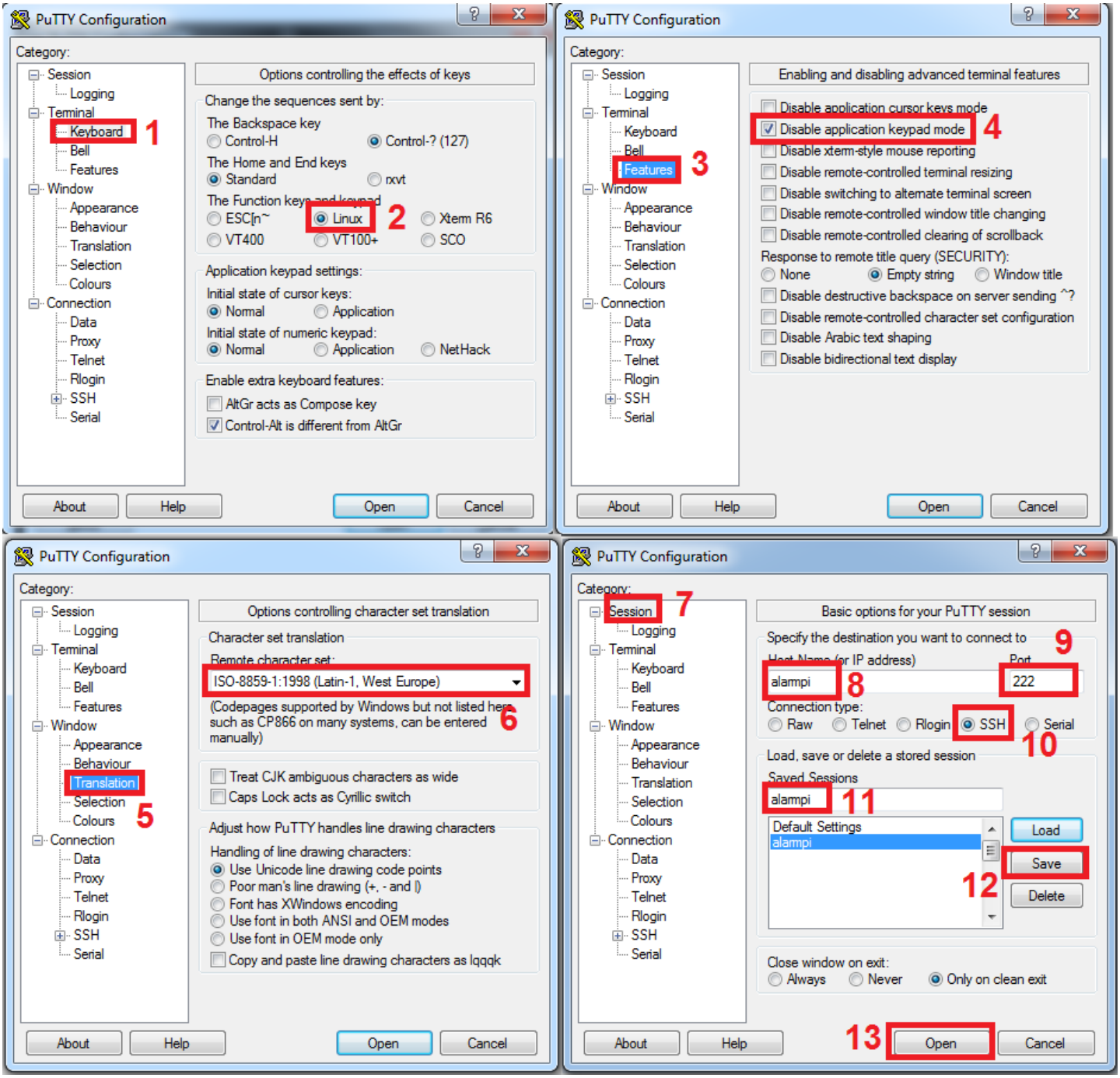
11.5 Basic Setup

There is no need to use a monitor on the Raspberry PI (RPI), in fact all the configuration can be done remotely with Putty as explained in tutorial 1.
Insert the microSd card into the RPI and the sound card. Turn on the RPI. Do not forget the network cable.

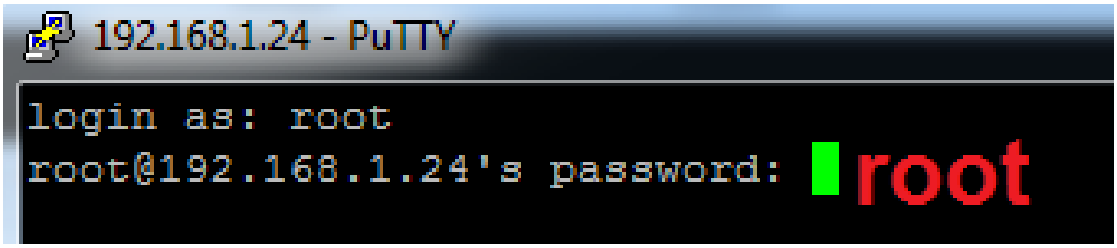
Start the command prompt  Invite de commandes and then with the ping alarmpi command, perform a connectivity test between the PC and the Raspberry PI as shown in the following screenshot.

```
C:\Users\anthony>ping alarmpi

Envoi d'une requête 'ping' sur alarmpi.home [192.168.1.24] av
nées :
Réponse de 192.168.1.24 : octets=32 temps<1ms TTL=64
Réponse de 192.168.1.24 : octets=32 temps<1ms TTL=64
Réponse de 192.168.1.24 : octets=32 temps<1ms TTL=64
Réponse de 192.168.1.24 : octets=32 temps<1ms TTL=64

Statistiques Ping pour 192.168.1.24:
    Paquets : envoyés = 4, reçus = 4, perdus = 0 (perte 0%),
Durée approximative des boucles en millisecondes :
    Minimum = 0ms, Maximum = 0ms, Moyenne = 0ms
```

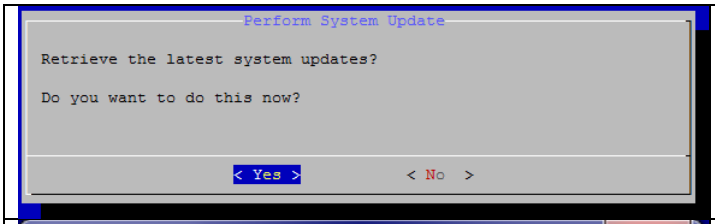
Configure Putty as follows before connecting to the RPI via ssh. (Don't forget to save the session, so you don't have to do this configuration work again)

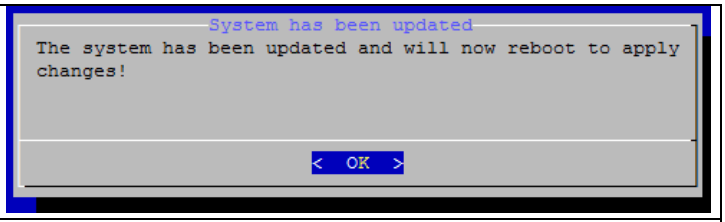


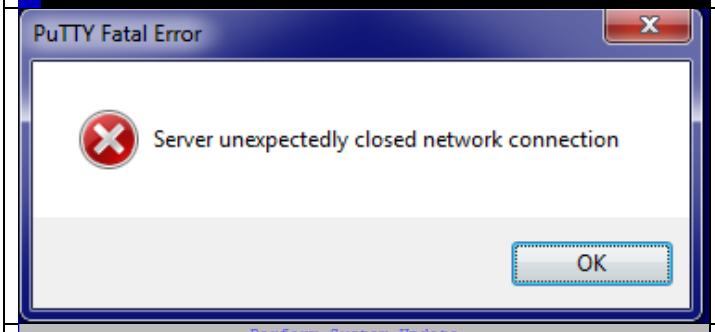
Then enter the login **`root`** and password **`root`**



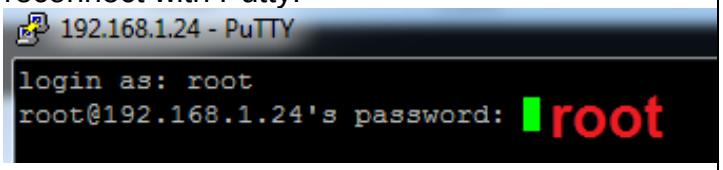
The configuration starts automatically. Follow the screenshots below:

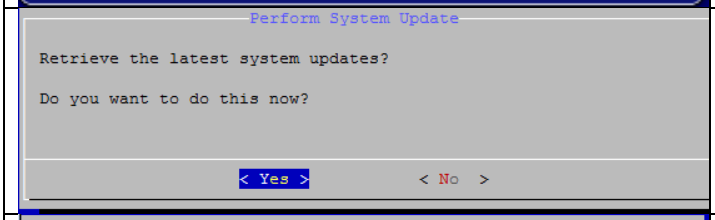


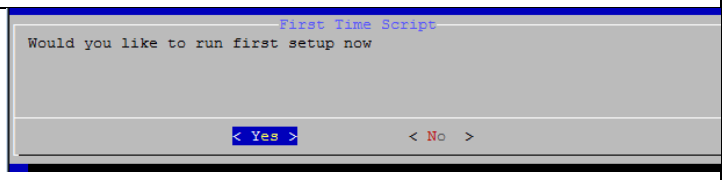


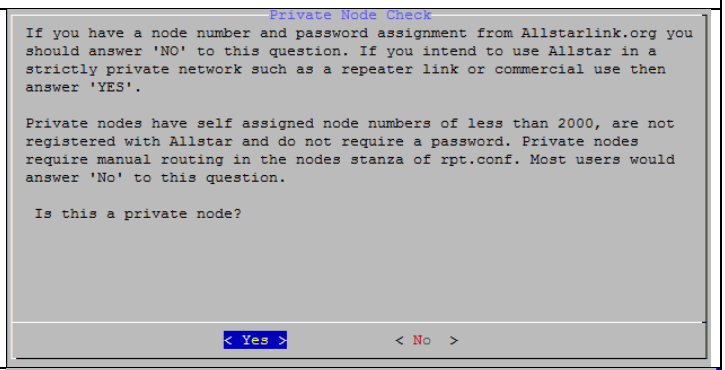


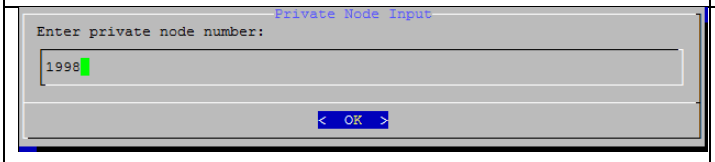
After the update and reboot you will need to reconnect with Putty.

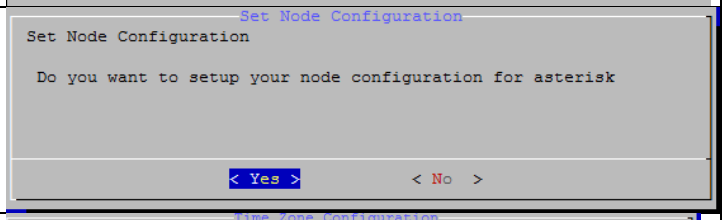


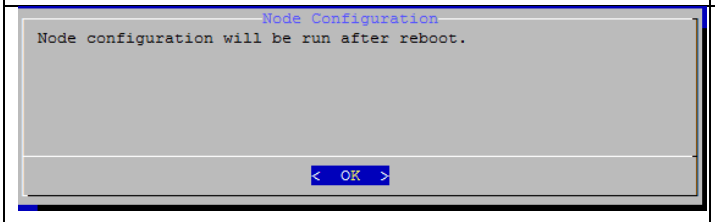


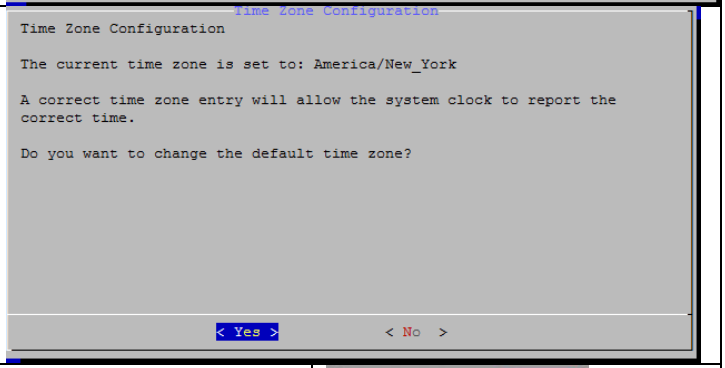




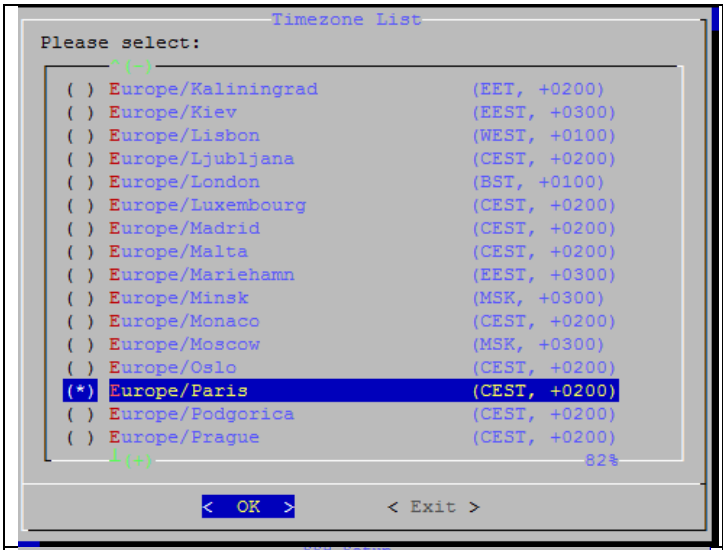


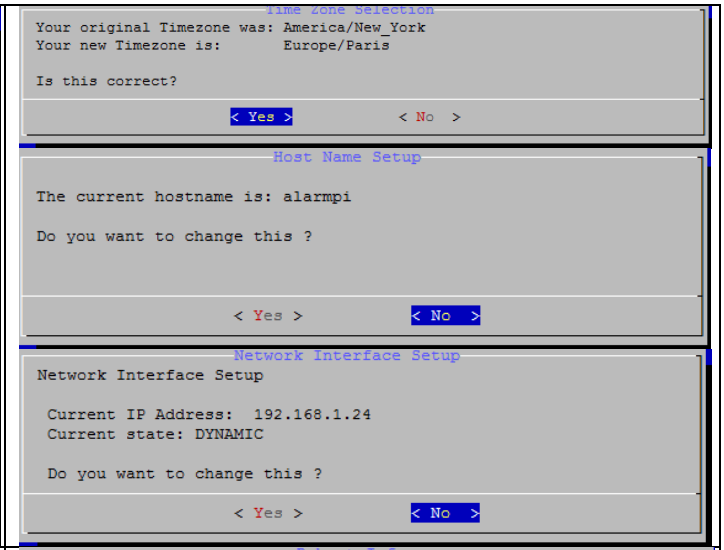


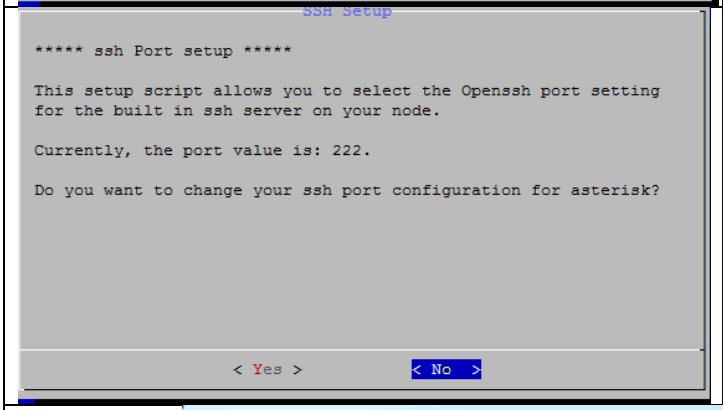


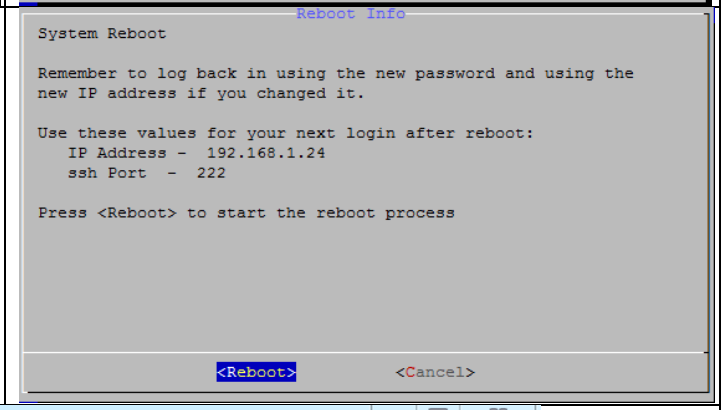


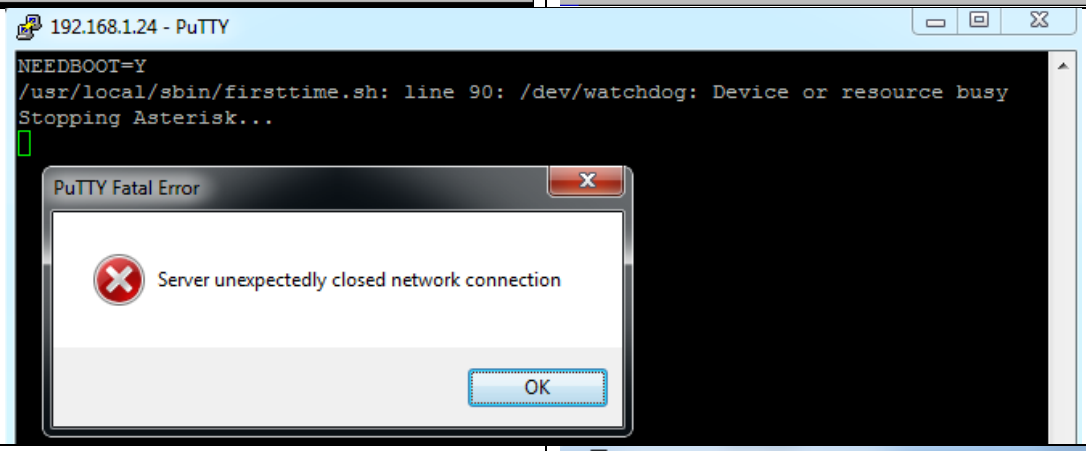
Use the scroll keys to move faster, then validate the country with the space bar before pressing the enter key.



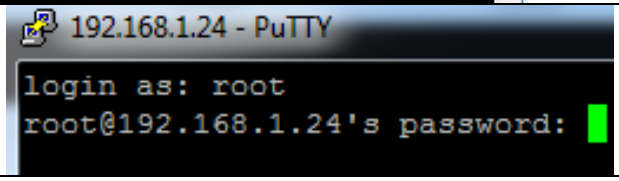


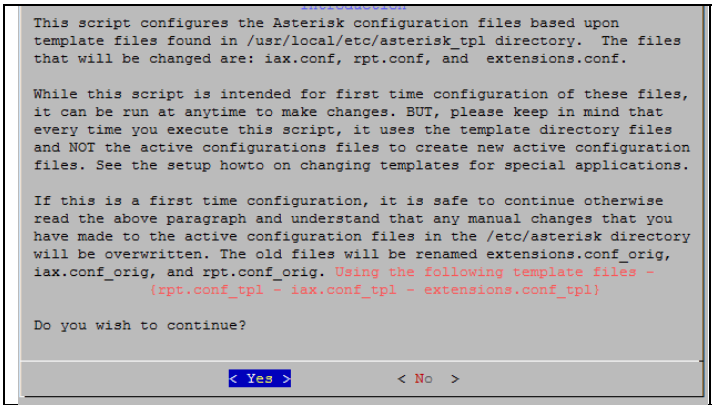


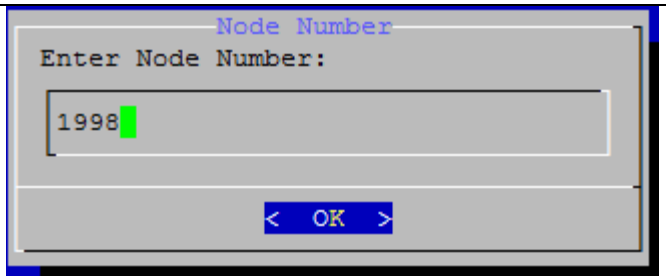




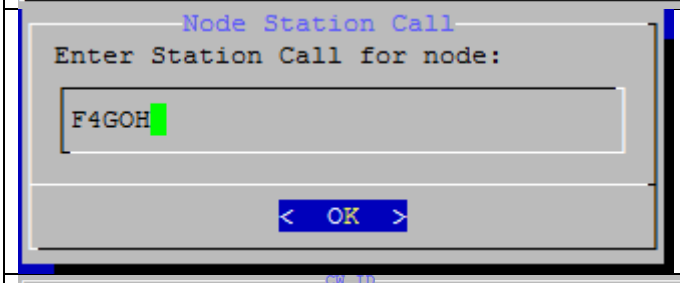
After the restart you will have to log in again with Putty, but with your new password of at least 8 characters. If you forget, you will have to start all over again.

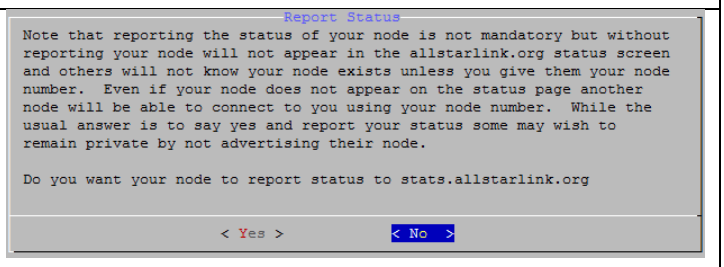


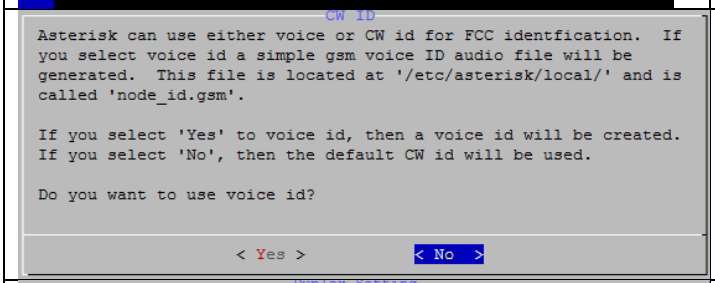


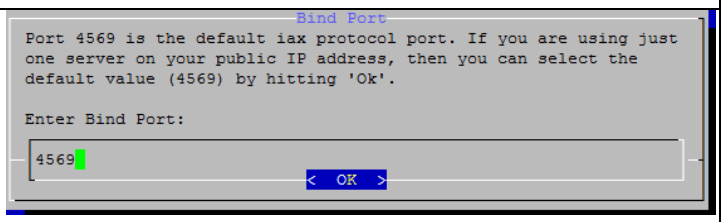


Keep 1998

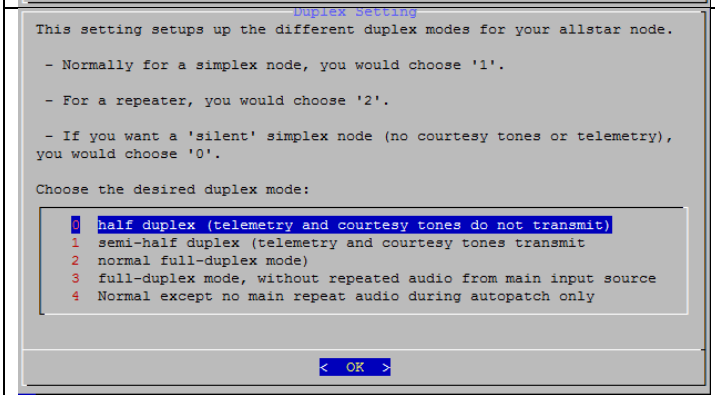


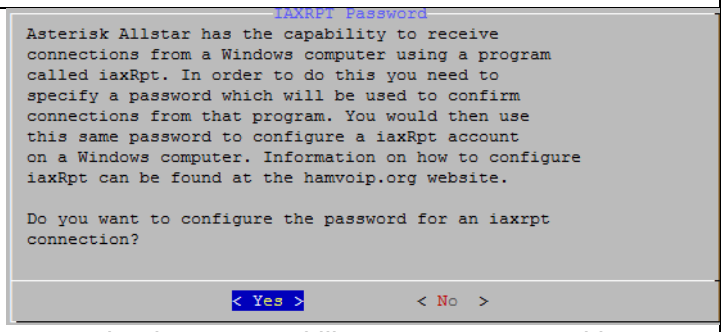




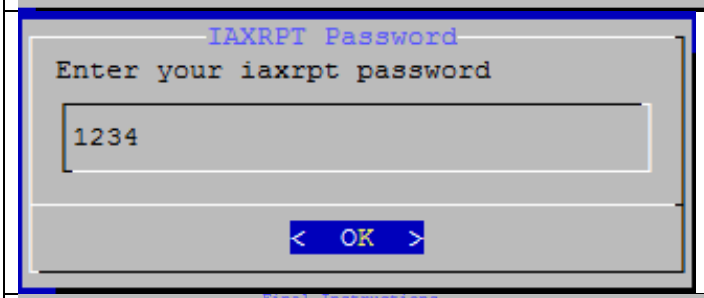


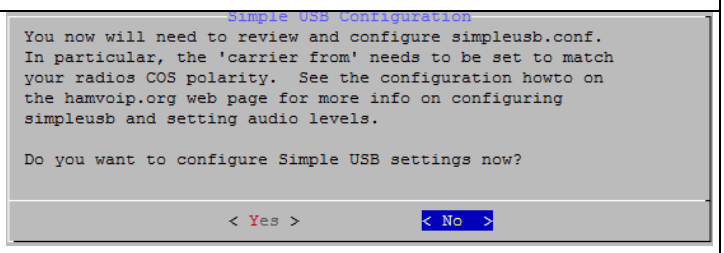
Keep 4569

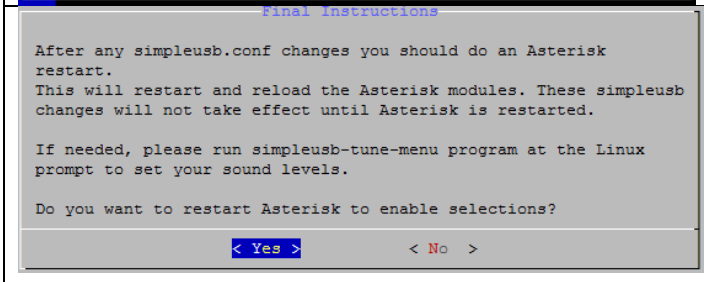


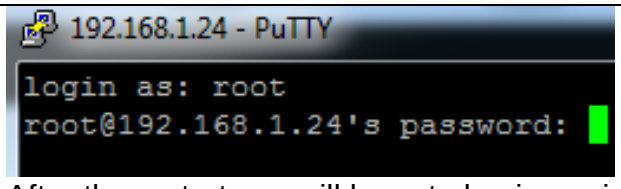


Use a simple password like "1234" to start with.





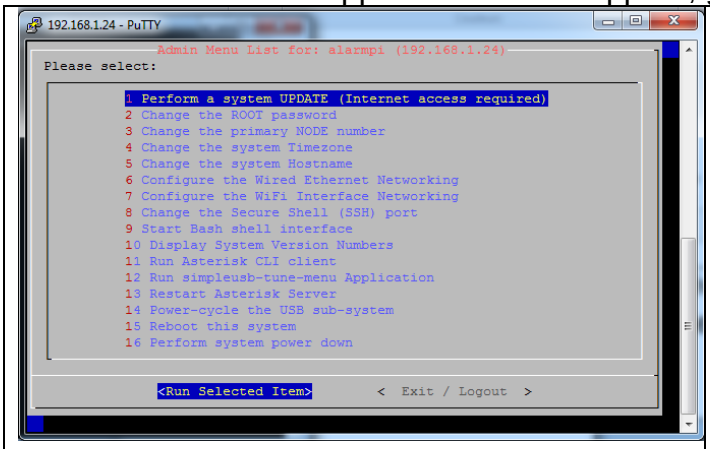


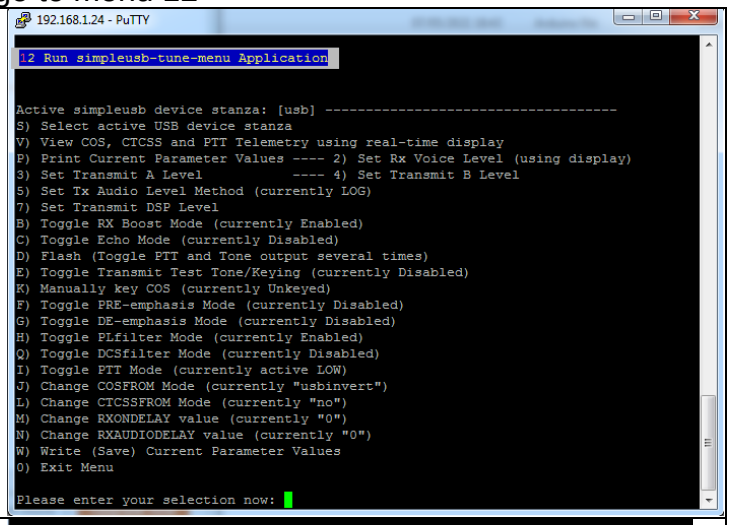


After the restart you will have to log in again with Putty, but with your new password of minimum 8 characters

11.6 TRX Settings

The main menu of the application should appear, go to menu 12





Menu **2**:
Change the sound level (100) of the reception.

```
Please enter your selection now: 2
RX VOICE DISPLAY:
                              v -- 3KHz        v -- 5KHz
|>                                                       |
Current setting on Rx Channel is 500
Enter new value (0-999, or CR for none): 100
Changed setting on RX Channel to 100
RX VOICE DISPLAY:
                              v -- 3KHz        v -- 5KHz
|>                                                       |
Current setting on Rx Channel is 100
Enter new value (0-999, or CR for none):
Rx voice setting not changed
```

Menu **3** :
As well as transmission (100)

```
Please enter your selection now: 3
Current setting on Tx Channel A is 500
Enter new Tx A Level setting (0-999, or C/R for none): 100
Changed setting on TX Channel A to 100
```

Menu J :
Change COSFROM mode to 2 [usb]

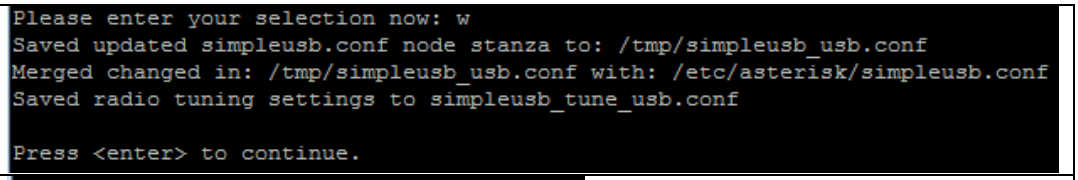
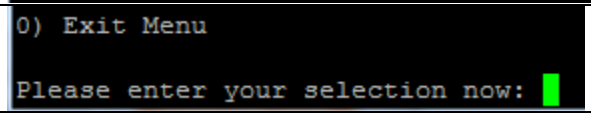
```
Please enter your selection now: j

Please select from the following telemetry methods:
1) [no]
2) [usb]
3) [usbinvert] (currently selected)
4) [pp]
5) [ppinvert]

Press <enter> to exit with no change
Enter make your selection now: 2
```

The **P** menu provides an overview of the configuration

```
Please enter your selection now: p
USB device String:       1-1.3:1.0

Card is:          0
Name is:          usb
Rx Level:         100
Rx ondelay:       0
Rx audio-delay:   0
Tx A Level:       100
Tx B Level:       500
Tx DSP Level:     999
preemphasis:      no
deemphasis:       no
plfilter:         yes
dcsfilter:        no
rxboost:          yes

PTT:              Active LOW
PTT status:       Un-Keyed

CTCSS (input):    Ignored
COS (input):      CM108/CM119 Active HIGH
COS (test):       Un-Keyed
COS (composite): Un-Keyed

Press <enter> to continue.
```

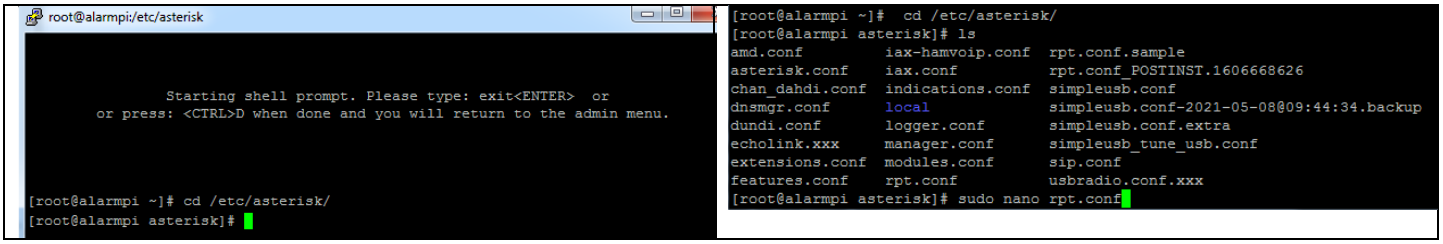
```
Please enter your selection now: v


Press <enter> to exit from this feature.

REALTIME RX TELEMETRY DISPLAY:
    COS      |    COS    |   COS    |   CTCSS   |    CTCSS     |  PTT
(composite) | (input) | (test) | (input) | (override) |
   CLEAR     |   CLEAR   | CLEAR  | IGNORE  |    CLEAR     | CLEAR
```

The **v** menu gives an overview of the status of the COS and PTT signals.
Very useful menu for detecting a failure.

Finish by saving the configuration (key **w**)	
Exit the menu (zero key: **0**)	

11.7 Advanced configuration

For command line configuration specialists, it is possible to do manual configuration in the 9 Start bash shell interface



The 3 configuration files are :

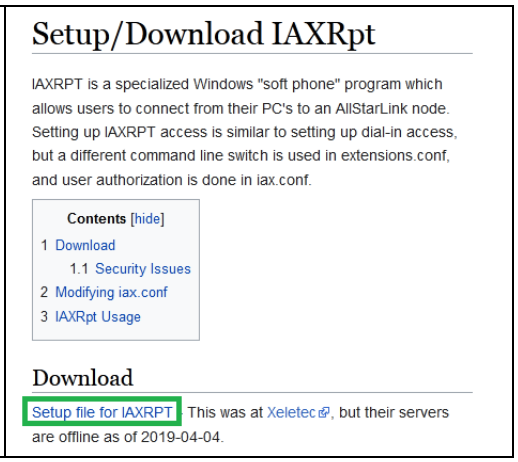
- **`rpt.conf`**
- **`iax.conf`**
- **`extensions.conf`**

https://wiki.allstarlink.org/wiki/Main_Page

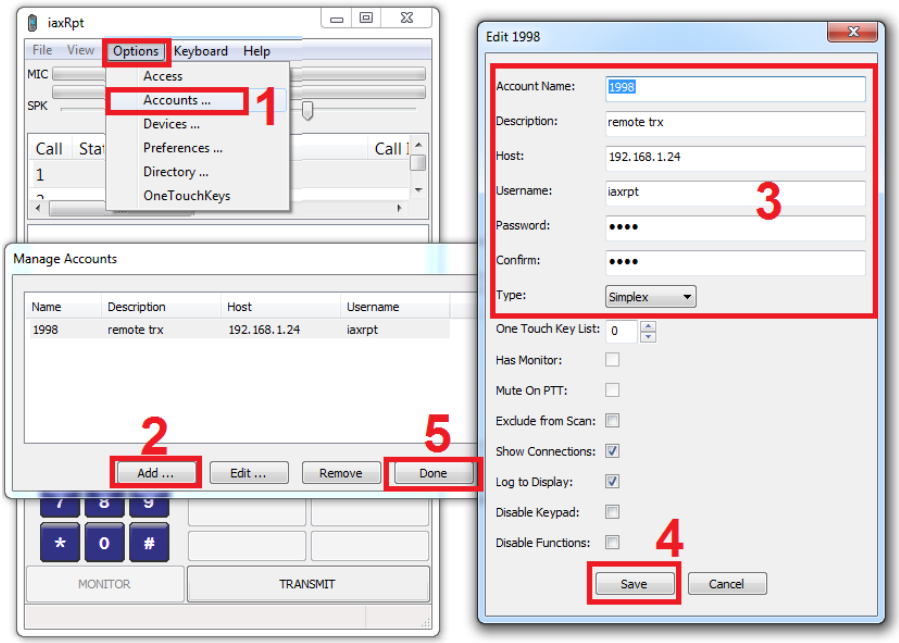
The problem is that the official documentation is complete but confusing, mixed with obsolete screenshots rather like 1000 post-it notes on a desktop.

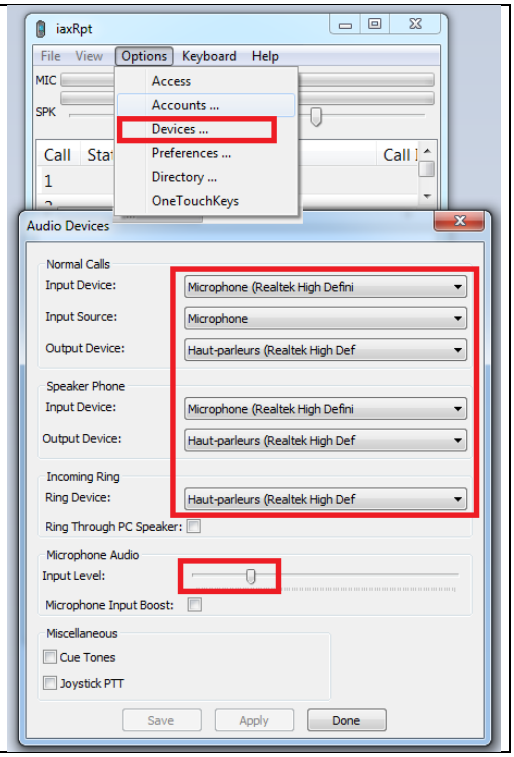
The following is the default configuration using the "node" number **1998** and the password "iaxrpt" **1234**.

11.8 Using iaxRpt on Windows

Download the iaxRpt utility (green box) using the link below: https://wiki.allstarlink.org/wiki/Setup/Download_IAXRpt	

Once the software is installed, perform the following configuration by adjusting the IP address of the RPI, and then check the sound configuration of the PC provided that the headset and microphone are properly connected.



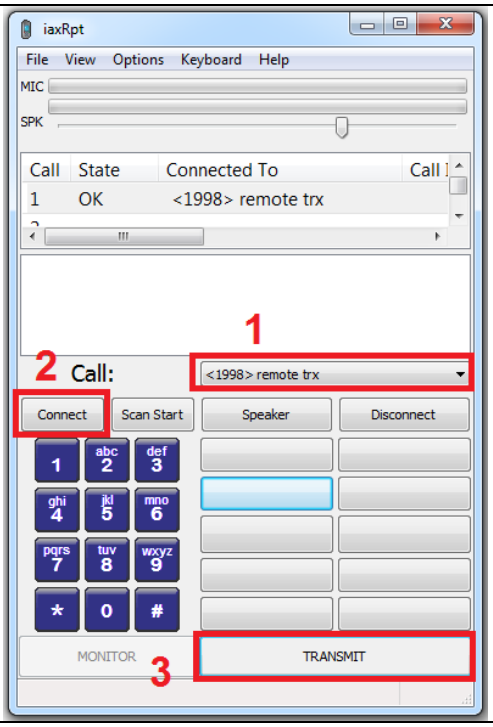


The password used here is **1234**, as previously configured.

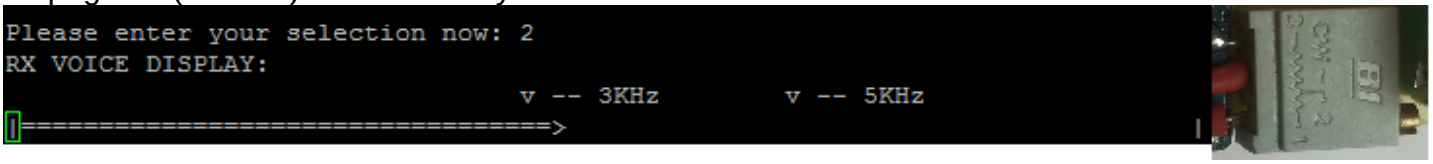
Operation:

1 Select "**<1998> remote trx**" from the drop-down menu;
2 Click on "**connect**";
3 Click on "**transmit**" to go to transmit;

Click on "**disconnect**" to leave the connection;

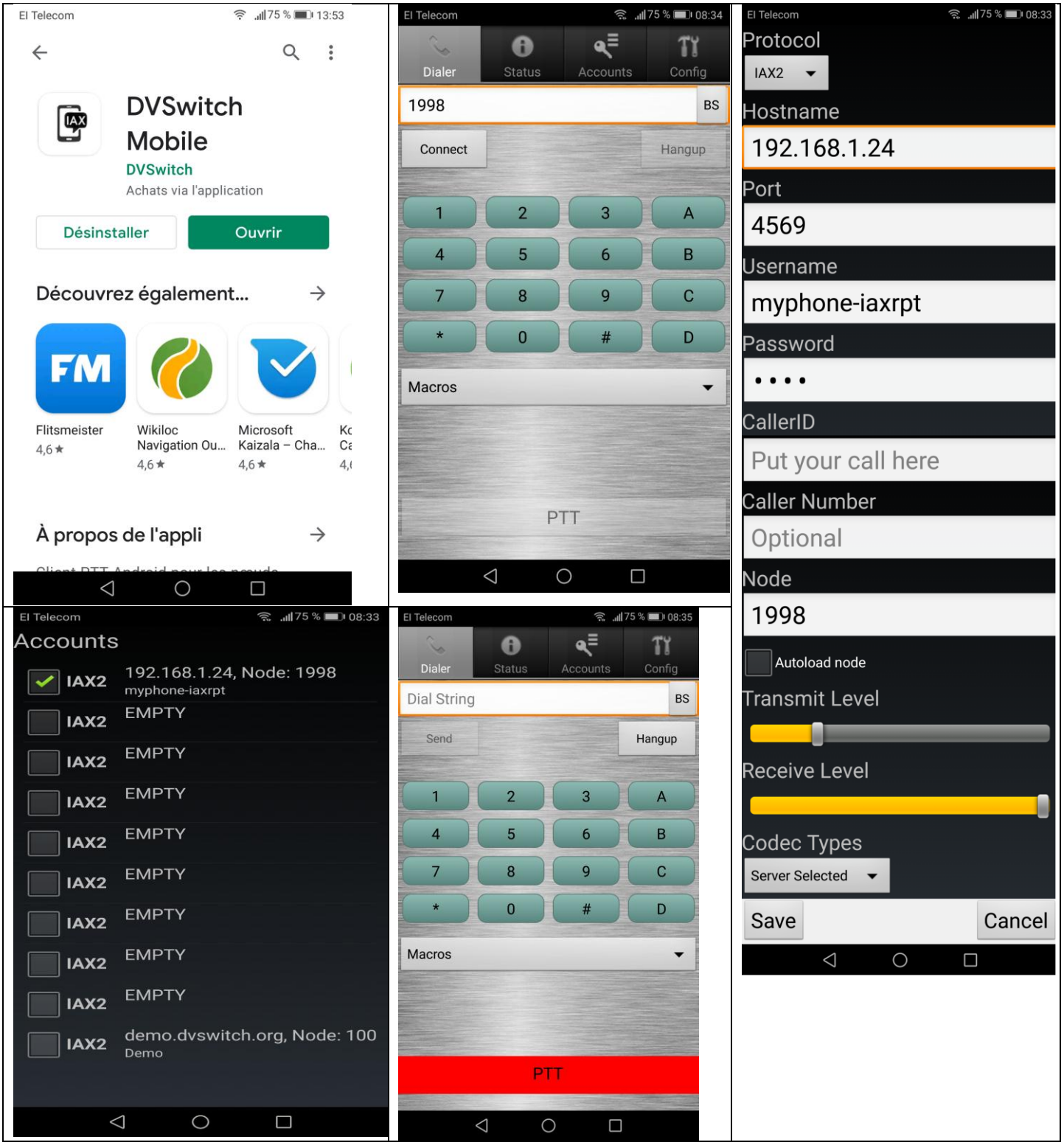


Adjust the sound levels with a screwdriver on the adjustable potentiometer if necessary as explained on page 11 (menu 2) to remove any sound saturation.



11.9 Using DVSwitch mobile for smartphone

Install DVSwitch Mobile, then go to the Account menu. Select the 1st line. Fill in the fields as shown in the example by adapting the IP address of the RPI. Do not forget to lower the Transmit Level
https://play.google.com/store/apps/details?id=org.dvswitch&hl=fr&gl=US



It is possible to use the Zoiper client software, but this does not have a PTT command.

11.10 Configuration overview

File **iax.conf**	File **extensions.conf**
Configuration for access with iaxRpt on PC	

`; IAXRPT Example - THis stanza is used for Windows IAXRPT`
 	
`[iaxrpt]`
`username=iaxrpt`
`type=user`
`context=radio-iaxrpt`
`auth=md5`
`secret=1234 ; PASSWORD`
`host=dynamic`
`disallow=all`
`allow=ulaw`
`allow=gsm`
`transfer=no`
	`[radio-iaxrpt]`
	`exten=1998,1,Rpt,1998|X`
	`exten=1999,1,Rpt,1999|X`

Configuration for access with DVSwitch Mobile on Smartphone	

`; Example for Android/Iphone Zoiper/IAXRPT phone connections`
 	
`[myphone-iaxrpt]`
`type=friend`
`context=phone-iaxrpt`
`host=dynamic`
`auth=md5`
`secret=1234 ; PASSWORD`
`disallow=all`
`allow=ulaw`
`transfer=no`
`calltokenoptional=0.0.0.0/0.0.0.0`
`requirecalltoken=no`
	`[phone-iaxrpt]`
	`exten => 1998,1,Answer`
	`exten => 1998,n,Playback,rpt/node`
	`exten => 1998,n,Playback,digits/1`
	`exten => 1998,n,Playback,digits/9`
	`exten => 1998,n,Playback,digits/9`
	`exten => 1998,n,Playback,digits/8`
	`exten => 1998,n,Set(CALLERID(num)=0)`
	`exten => 1998,n,Rpt,1998|P|${CALLERID(name)} ;;;`
	`The "CallerID" from IAXRpt`

Note the links between the two configuration files as well as the password 1234 and the "node" **1998**. There is a difference in configuration between using Windows and Smartphone.

11.11 Conclusion

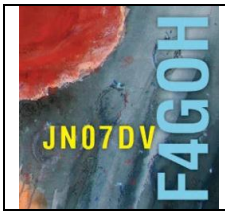
Once you have all the elements in hand, the basic configuration is fairly easy to do, but you shouldn't really rely on the official documentation. Fortunately, there are many examples on Youtube illustrating "bits" of configuration on both the server and the client.

In any case, the investment remains minimal with the sound USB key. Of course, you need a Raspberry PI and a TRX with the COS output that is essential for this project.

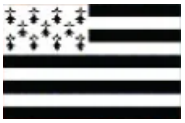
HamVoIP is an interesting project that avoids investing in expensive and licensed hardware. It is also possible to create an allstarlink account allowing other OMs to use your access point (this is not developed here).

The modified key will be used in another APRS tutorial soon. So this little interface can be used more.

73 and good remote.



Remote Station with an SDR receiver Part 12



Sommaire :

Version 15/9/2021 V1.0 on RPI4

12.1 Server/client set Presentation

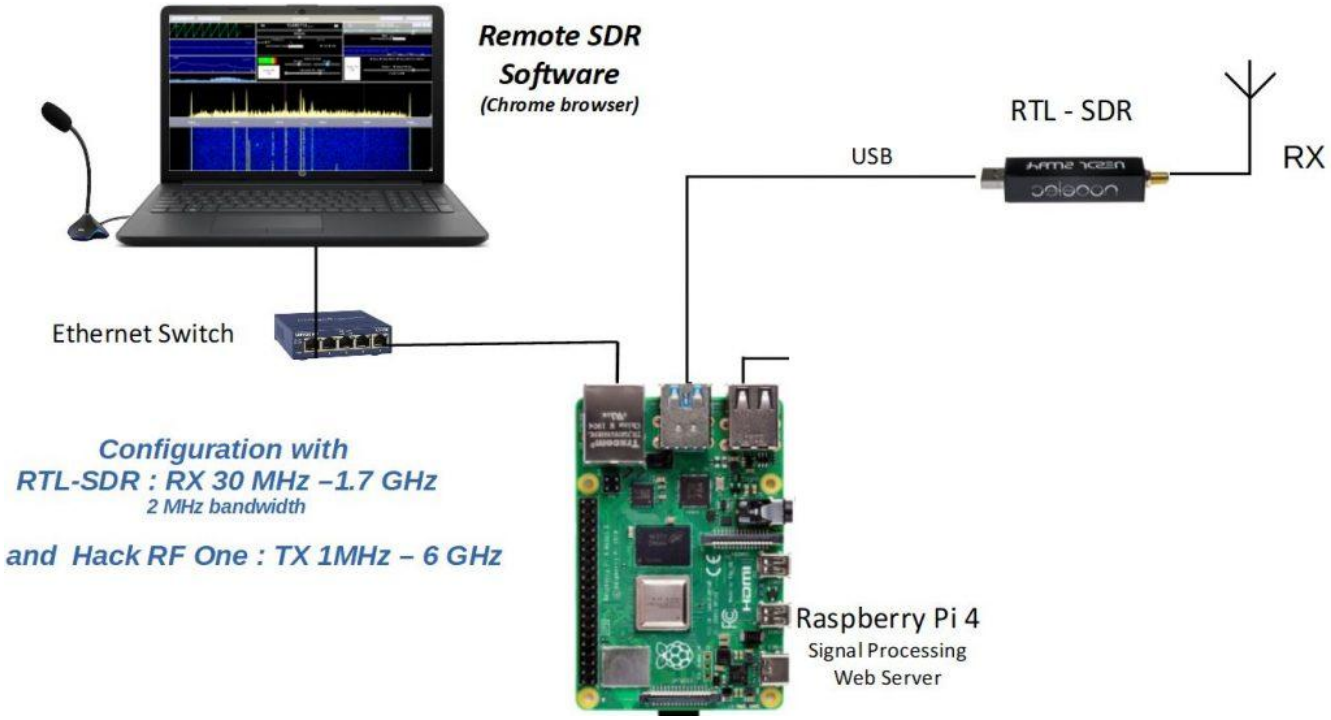
The goal is to be able to use an SDR key remotely using a WEB browser.
Using the Chrome browser (mandatory), the user accesses the Raspberry PI on their local network to which the RTL-SDR key is connected.

Note that it is possible to use with the same environment an Adalm-Pluto or a HackRF and thus use the transmission part which differs from the openwebRX project (tutorial 6.)

The software was produced by f1atb. His work is remarkable. The interface is user-friendly and intuitive.

https://f1atb.fr

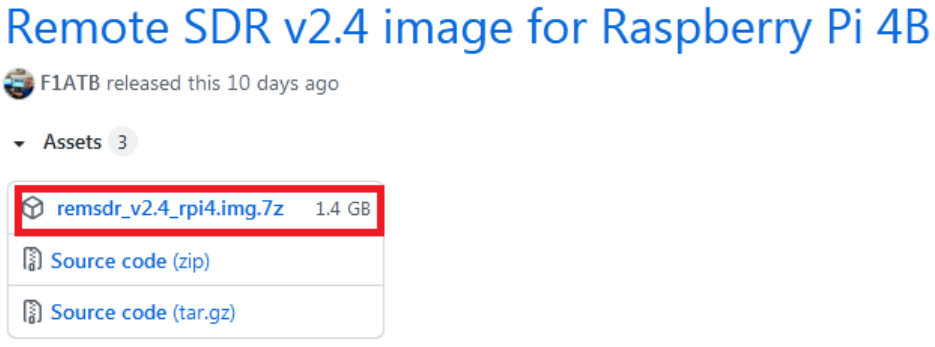
Example of configuration with a PC on a local network (f1atb image). Only the use of the RTL-SDR key is explained in this document.



12.2 Creating the disk image

Download the disk image (framed in red) using this link

https://github.com/F1ATB/Remote-SDR/releases

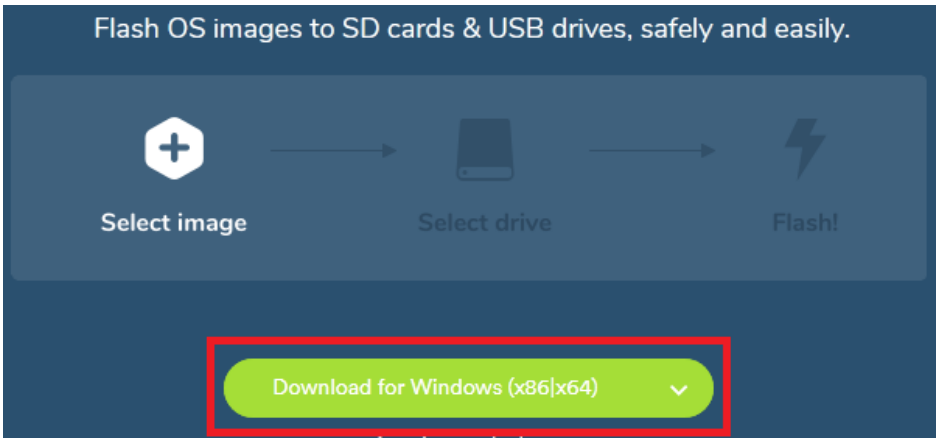


Use 7-zip software to decompress the image

Lien	Type	Windows	Taille
Téléchargez	.exe	32 bits x86	1,2 Mo
Téléchargez	.exe	64 bits x64	1,4 Mo

Download and install the Etcher utility to copy the image to an mSD card

https://www.balena.io/etcher/

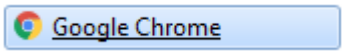


Select the remsdr_v2.4_rpi4.img file, the mSD card reader, then click on Flash. Same procedure as tutorial 8.

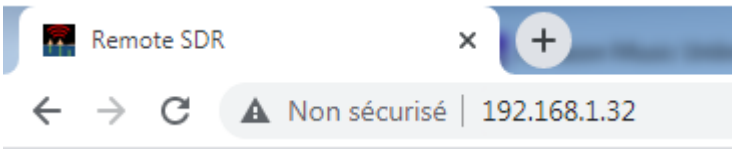
12.3 Basic Setup

Identify the IP address of your Raspberry PI.

The procedure is explained in tutorial 1 chapter 5 page 7.

Install if you haven't already done so, the mandatory Chrome browser. 

Then enter the IP address of the Raspberry PI in the URL bar, in my case 192.168.1.32

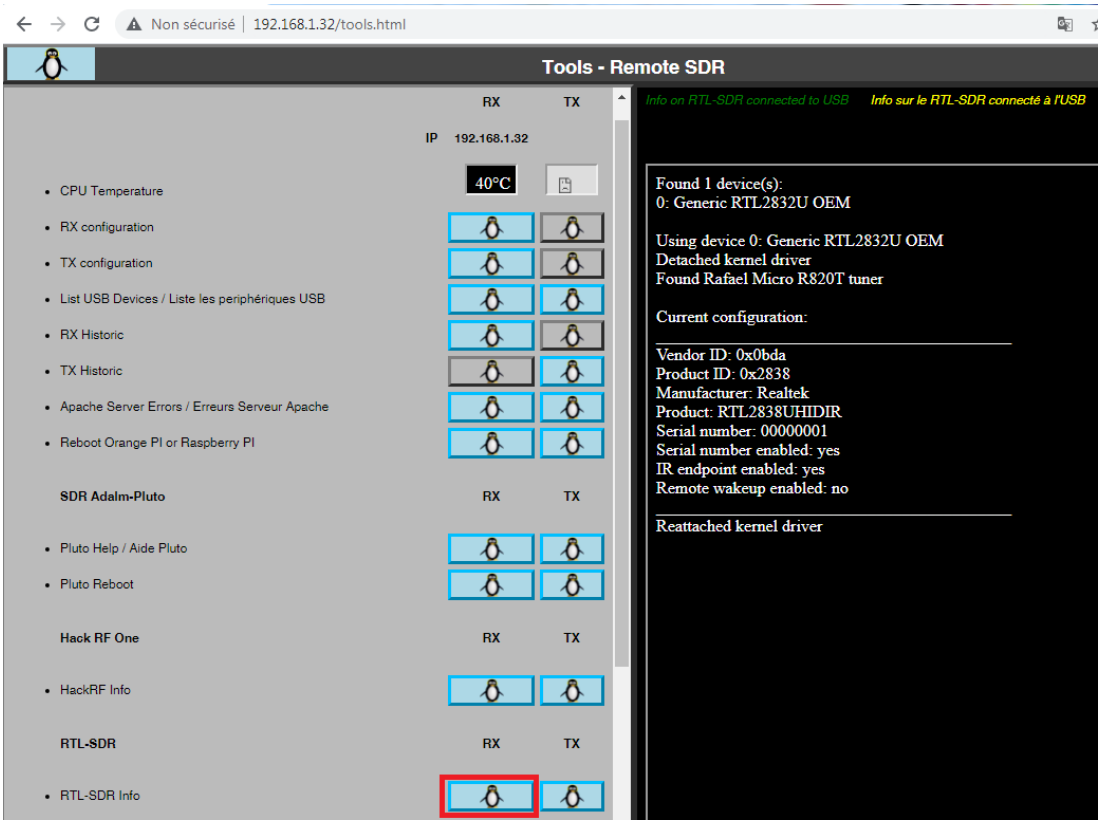


12.4 Basic configuration

There is virtually nothing to configure. You will still have to check if the RTL-SDR key is properly connected to the Raspberry PI.

Click on Tools

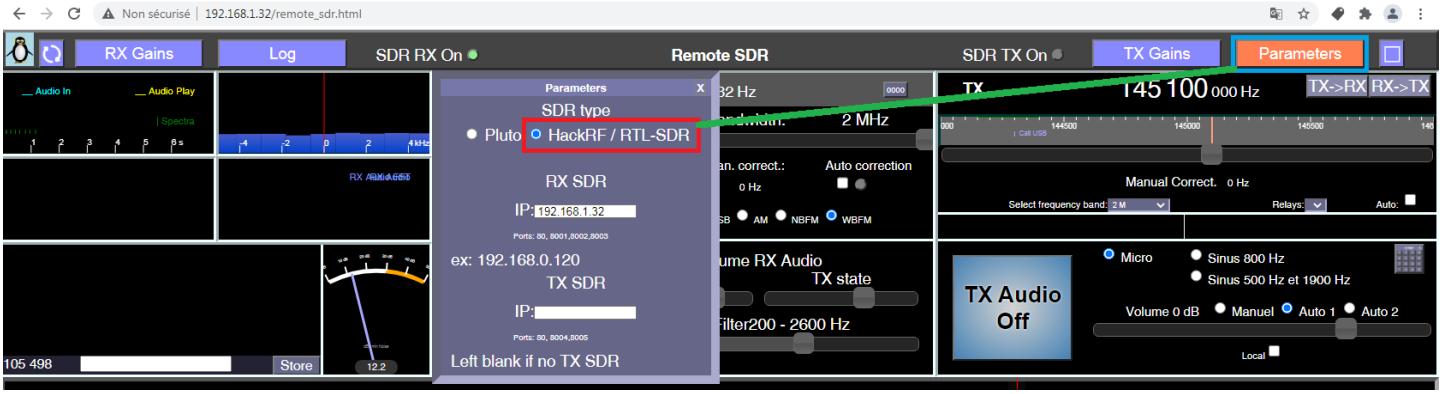
Then check the detection of the RTL-SDR key as indicated below in red.



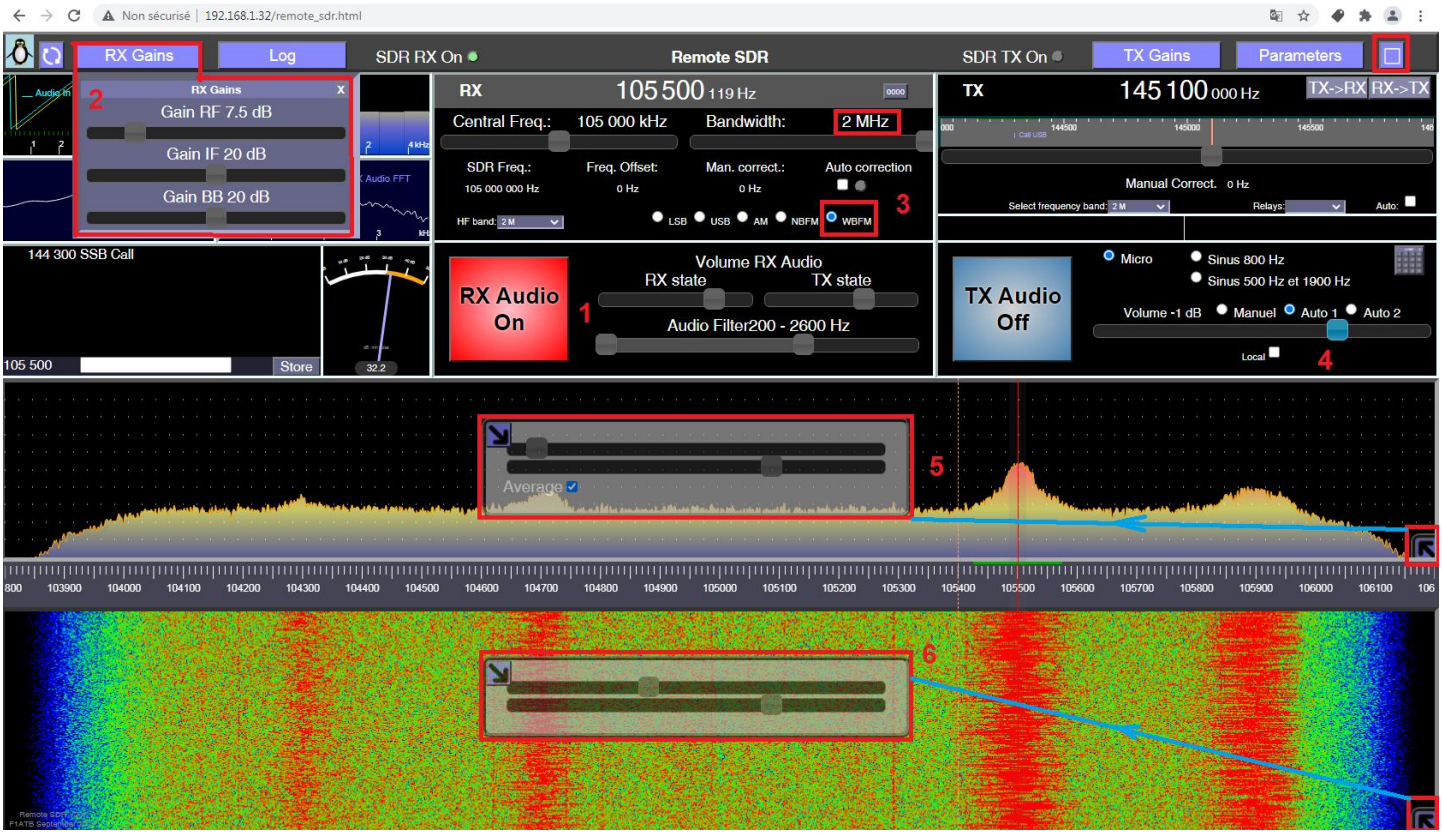
Return to the main menu (penguin icon at the top right)

Click on Remote SDR

Check the parameter selection for the RTL-SDR key. The IP address must match the Raspberry PI.



Test the reception of the RTL-SDR key using a broadcast FM station as usual.

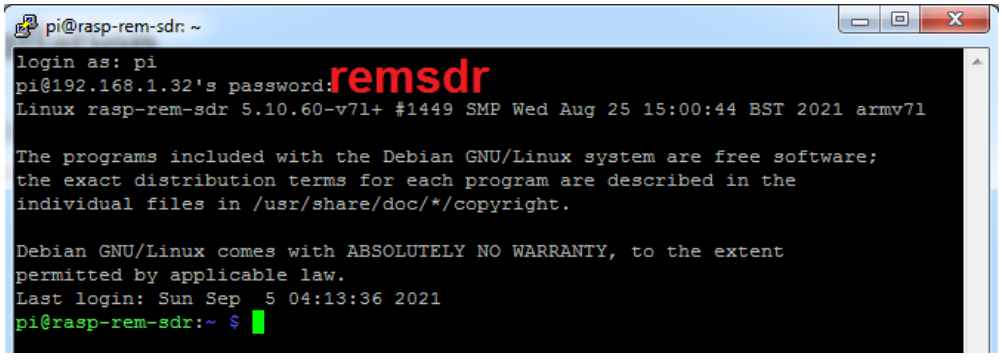


1 - activate reception,
2 - Adjust the gain on reception,
3 - Select WBFM and adjust the listening frequency. Increase the bandwidth to 2 MHz,
4 - Adjust the volume,
5/6 - Adjust the levels to correctly view the stations.

All you have to do is click in the waterfall to listen to the selected station.

12.5 Ssh connexion

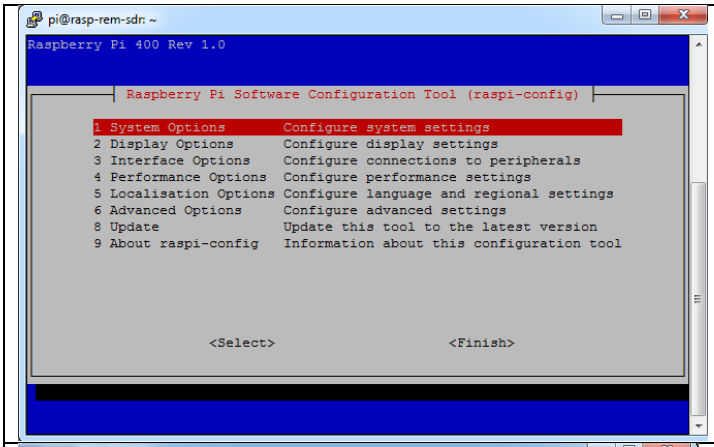
To access the Raspberry PI in ssh with putty (tutorial 1 page 10), enter the following login and password:

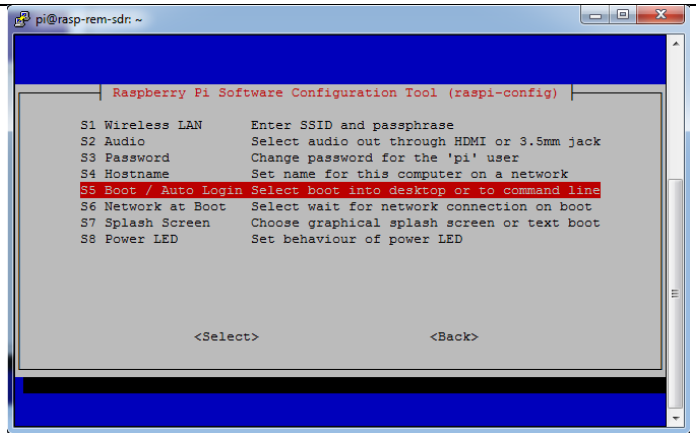


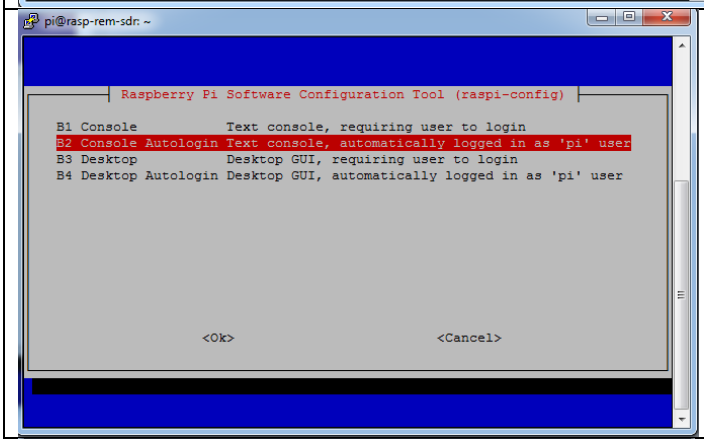
I disabled the desktop environment when starting the Raspberry PI.

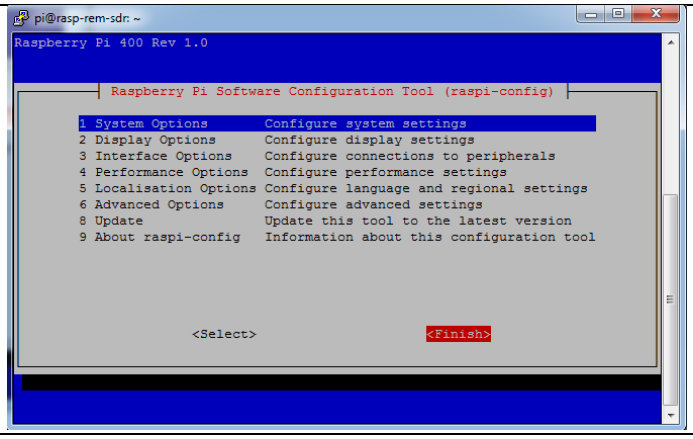
```
sudo raspi-config
```

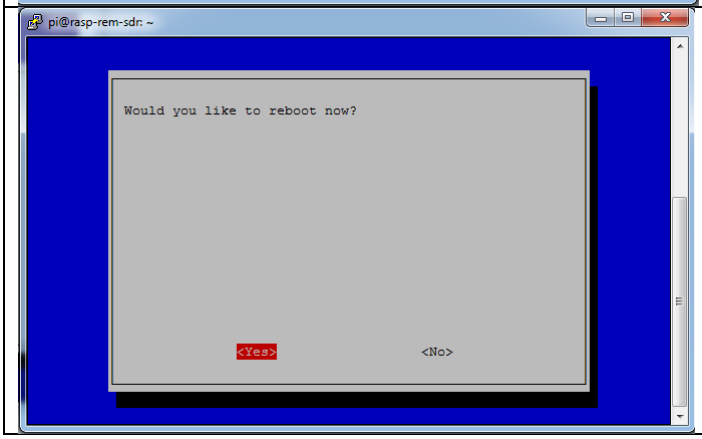
```
pi@rasp-rem-sdr:~ $ sudo raspi-config
pi@rasp-rem-sdr:~ $
```





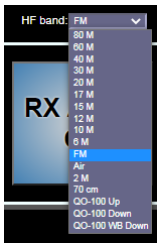






It is possible to change the predetermined frequencies by editing the following file in the console:

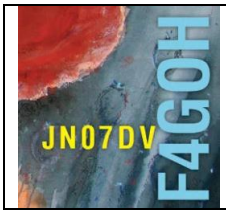
```
nano  /var/www/html/configurationRX.js
```



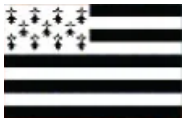
12.6 Conclusion

Bravo André (F1ATB) for this magnificent achievement. The graphical interface is very ergonomic and will delight OMs wishing to control their station remotely.

73 and good remote.



Universal HamRadio Remote Part 13



Summary :

Part 13 : Universal HamRadio Remote

Prerequisite: Part 4: Using the Raspbian Lite operating system without a graphical interface.

This document contains the information at the following link with screenshots.

https://github.com/F4HTB/Universal_HamRadio_Remote_HTML5/wiki

Version du 30/10/2021 V1.0 sur RPI4

13.1 Server/client set Presentation

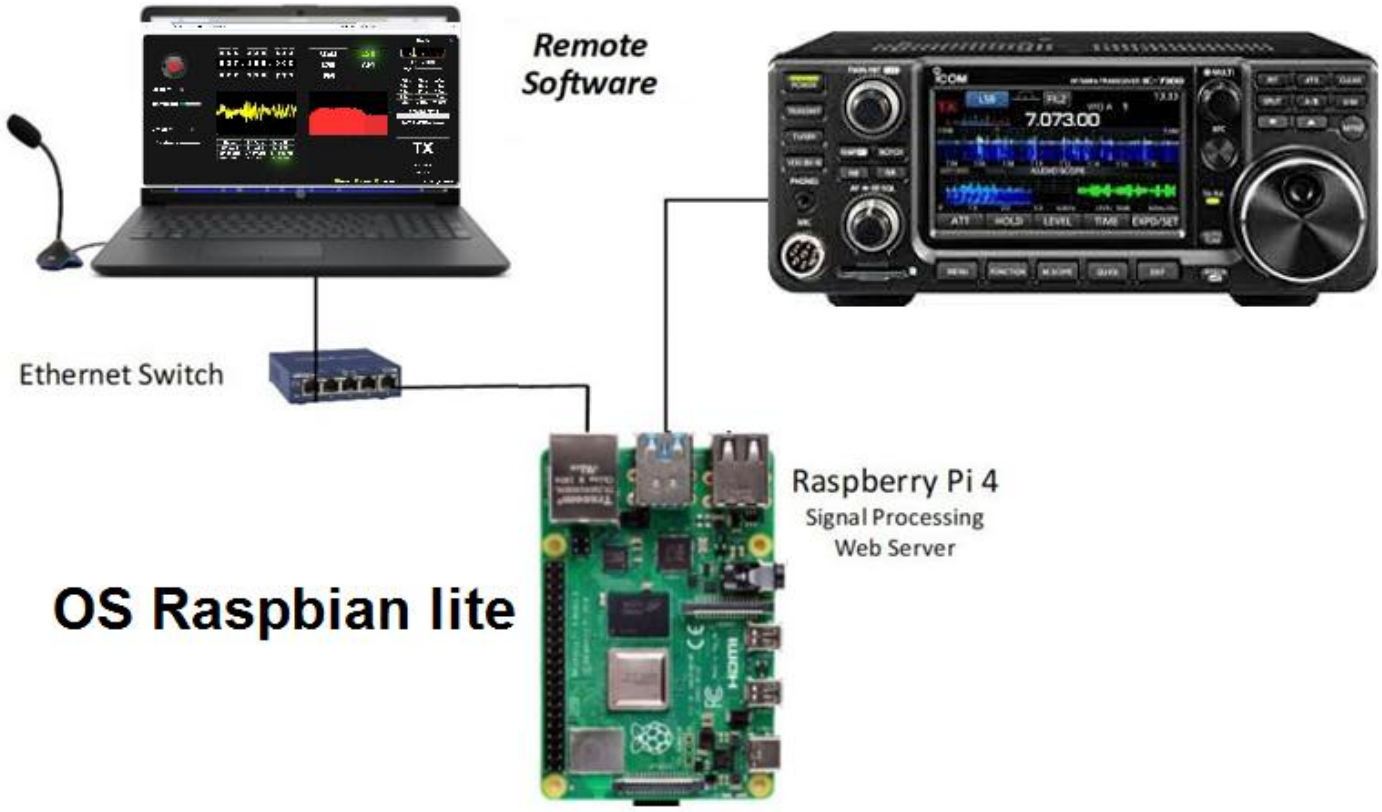
The goal is to be able to use a transceiver remotely using a WEB browser.

Using the Firefox browser (for example), the user accesses the Raspberry PI on their local network to which the transceiver is connected.

Note that it is possible to use with the same environment, the transceivers supported by the Hamlib library, (which is considerable).

The software was produced by F4HTB. The interface is user-friendly and intuitive.

Example of a configuration with a PC on a local network with an IC7300. The operator will be able to traffic remotely by voice with a PC connected to his local network. The Raspberry PI is used as a WEB server.



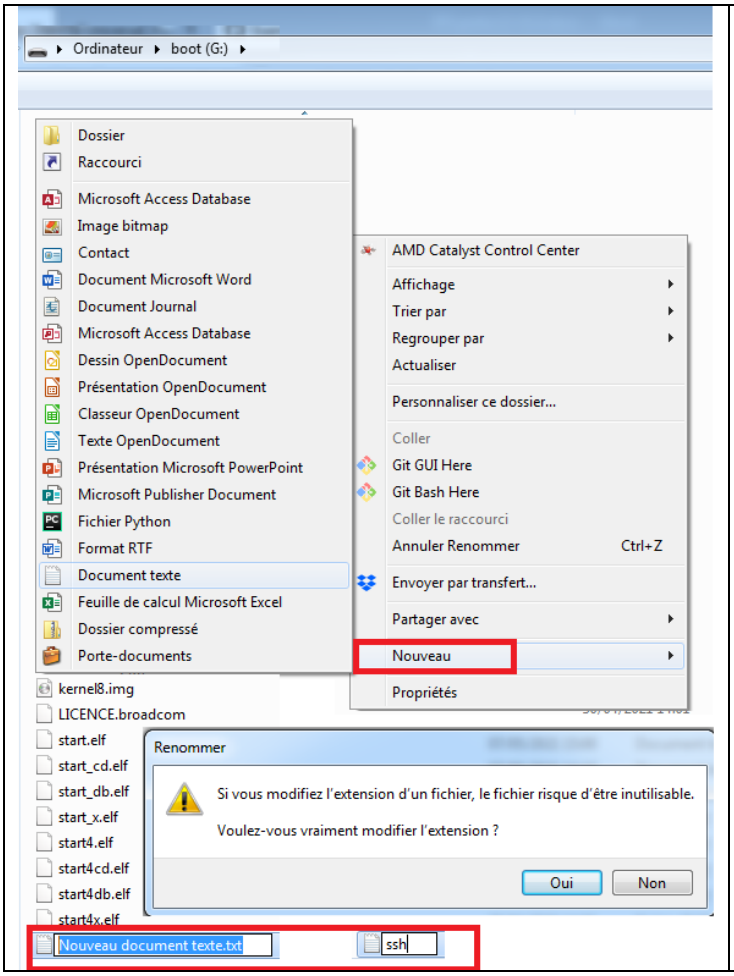
13.2 Installation

F4HTB does not offer a ready-made disc image. You will have to install the software manually.

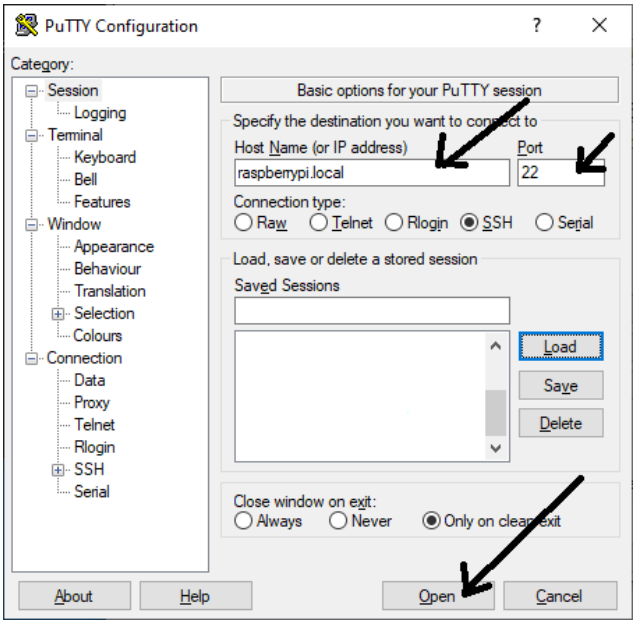
Start by loading the Raspbian Lite operating system without a graphical interface onto an mSD card, as shown in tutorial 4 or using the F4HTB guide.

https://github.com/F4HTB/Universal_HamRadio_Remote_HTML5/wiki/Example-of-complete-installation

Once the image with the Raspbian Lite operating system is created, add the empty "ssh" file without extension in the root of the mSD card named boot.
This directly activates the ssh connection without going through the `raspi-config` menu explained in tutorial 4.



Start the raspberry with the mSD card. Then establish an ssh connection with putty.



Update the operating system

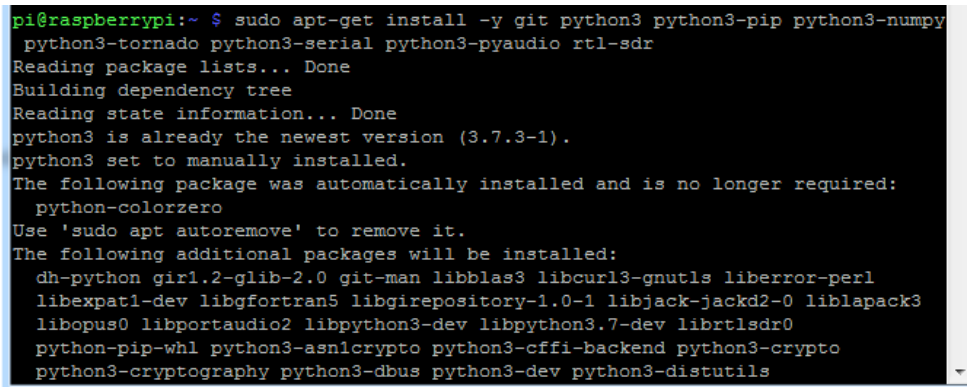
```
sudo apt-get update && sudo apt-get upgrade -y && sudo reboot
```

Allow 15 minutes to wait.

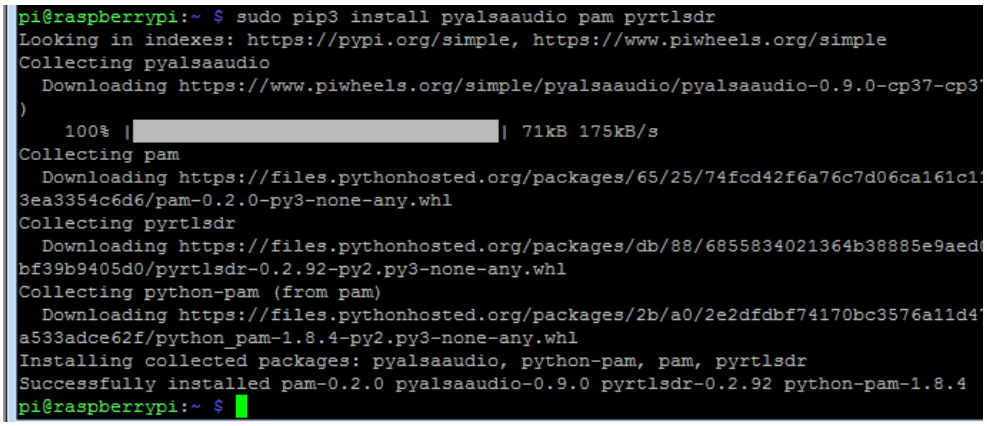
After restarting, reconnect in ssh with putty.

Install the Python 3 environment and its dependencies

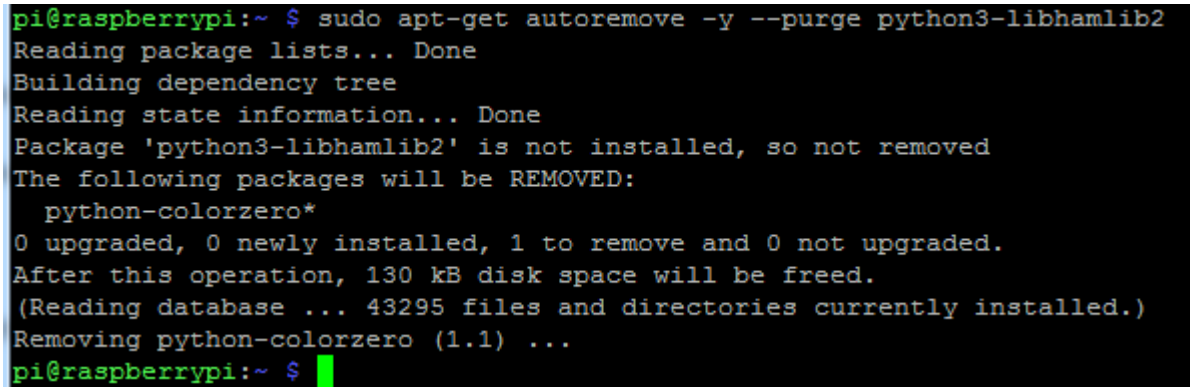
```
sudo apt-get install -y git python3 python3-pip python3-numpy
python3-tornado python3-serial python3-pyaudio rtl-sdr
```




```
sudo pip3 install pyalsaaudio pam pyrtlsdr
```




```
sudo apt-get autoremove -y --purge python3-libhamlib2
```



```
sudo apt-get install -y autoconf automake libtool swig
```

```
pi@raspberrypi:~ $ sudo apt-get install -y autoconf automake libtool swig
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  autotools-dev libltdl-dev libltdl7 libsigsegv2 m4 swig3.0
Suggested packages:
  autoconf-archive gnu-standards autoconf-doc gettext libtool-doc gfortran
  | fortran95-compiler gcj-jdk m4-doc swig-doc swig-examples swig3.0-examples
  swig3.0-doc
The following NEW packages will be installed:
  autoconf automake autotools-dev libltdl-dev libltdl7 libsigsegv2 libtool m4
  swig swig3.0
0 upgraded, 10 newly installed, 0 to remove and 0 not upgraded.
Need to get 4,030 kB of archives.
After this operation, 12.6 MB of additional disk space will be used.
Get:1 http://ftp.igh.cnrs.fr/pub/os/linux/raspbian/raspbian buster/main armhf li
bsigsegv2 armhf 2.12-2 [32.3 kB]
Get:2 http://ftp.igh.cnrs.fr/pub/os/linux/raspbian/raspbian buster/main armhf m4
 armhf 1.4.18-2 [185 kB]
Get:3 http://ftp.igh.cnrs.fr/pub/os/linux/raspbian/raspbian buster/main armhf au
toconf all 2.69-11 [341 kB]
Get:4 http://ftp.igh.cnrs.fr/pub/os/linux/raspbian/raspbian buster/main armhf au
totools-dev all 20180224.1 [77.0 kB]
Get:5 http://ftp.igh.cnrs.fr/pub/os/linux/raspbian/raspbian buster/main armhf au
tomake all 1:1.16.1-4 [771 kB]
```

Install the Hamlib library

```
cd ~/
git clone https://github.com/Hamlib/Hamlib.git
```

```
pi@raspberrypi:~ $ cd ~/
pi@raspberrypi:~ $ git clone https://github.com/Hamlib/Hamlib.git
Cloning into 'Hamlib'...
remote: Enumerating objects: 47578, done.
remote: Counting objects: 100% (4863/4863), done.
remote: Compressing objects: 100% (1784/1784), done.
remote: Total 47578 (delta 3542), reused 4235 (delta 3068), pack-reused 42715
Receiving objects: 100% (47578/47578), 18.30 MiB | 333.00 KiB/s, done.
Resolving deltas: 100% (37673/37673), done.
pi@raspberrypi:~ $ cd Hamlib/
pi@raspberrypi:~/Hamlib $
```

```
cd Hamlib/
./bootstrap
```

```
pi@raspberrypi:~/Hamlib $ ./bootstrap
Running 'autoreconf -i' to process configure.ac
and generate the configure script.
aclocal: installing 'macros/libtool.m4' from '/usr/share/aclocal/libtool.m4'
aclocal: installing 'macros/ltoptions.m4' from '/usr/share/aclocal/ltoptions.m4'
aclocal: installing 'macros/ltsugar.m4' from '/usr/share/aclocal/ltsugar.m4'
aclocal: installing 'macros/ltversion.m4' from '/usr/share/aclocal/ltversion.m4'
aclocal: installing 'macros/lt~obsolete.m4' from '/usr/share/aclocal/lt~obsolete.m4'
aclocal: installing 'macros/pkg.m4' from '/usr/share/aclocal/pkg.m4'
libtoolize: putting auxiliary files in AC_CONFIG_AUX_DIR, 'build-aux'.
libtoolize: copying file 'build-aux/ltmain.sh'
configure.ac:95: installing 'build-aux/ar-lib'
configure.ac:26: installing 'build-aux/compile'
configure.ac:126: installing 'build-aux/config.guess'
configure.ac:126: installing 'build-aux/config.sub'
configure.ac:35: installing 'build-aux/install-sh'
configure.ac:35: installing 'build-aux/missing'
amplifiers/elecraft/Makefile.am: installing 'build-aux/depcomp'
bindings/Makefile.am:106: installing 'build-aux/py-compile'
parallel-tests: installing 'build-aux/test-driver'
pi@raspberrypi:~/Hamlib $
```

```
./configure --with-python-binding PYTHON=$(which python3)
```

```
pi@raspberrypi:~/Hamlib $ ./configure --with-python-binding PYTHON=$(which python3)
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C89... none needed
checking whether gcc understands -c and -o together... yes
checking how to run the C preprocessor... gcc -E
checking for grep that handles long lines and -e... /usr/bin/grep
checking for egrep... /usr/bin/grep -E
checking for ANSI C header files... yes
checking for sys/types.h... yes
checking for sys/stat.h... yes
checking for stdlib.h... yes
checking for string.h... yes
checking for memory.h... yes
checking for strings.h... yes
checking for inttypes.h... yes
checking for stdint.h... yes
checking for unistd.h... yes
checking minix/config.h usability...
```

```
make all && sudo make install && cd bindings && make && sudo make install && sudo ldconfig
```

The compilation is quite long, wait about 10 minutes

```
Hamlib Version 4.4~git configuration:

Prefix          /usr/local
Preprocessor    gcc -E
C Compiler      gcc -g -O2
C++ Compiler    g++ -std=c++11 -g -O2

Package features:
   With C++ binding               yes
   With Perl binding              no
   With Python binding            yes
   With TCL binding               no
   With Lua binding               no
   With rigmem XML support        no
   With Readline support          no
   With INDI support              no

   Enable HTML rig feature matrix  no
   Enable WinRadio                yes
   Enable USRP                    no
   Enable USB backends            no
   Enable shared libs             yes
   Enable static libs             yes

----------------------------------------------------------------------
pi@raspberrypi:~/Hamlib $ make all && sudo make install && cd bindings && make && sudo make install && sudo l
dconfig
Making all in macros
make[1]: Entering directory '/home/pi/Hamlib/macros'
make[1]: Nothing to be done for 'all'.
make[1]: Leaving directory '/home/pi/Hamlib/macros'
Making all in include
make[1]: Entering directory '/home/pi/Hamlib/include'
make  all-am
make[2]: Entering directory '/home/pi/Hamlib/include'
make[2]: Leaving directory '/home/pi/Hamlib/include'
make[1]: Leaving directory '/home/pi/Hamlib/include'
Making all in lib
make[1]: Entering directory '/home/pi/Hamlib/lib'
  CC       termios.lo
  CC       dummy.lo
```

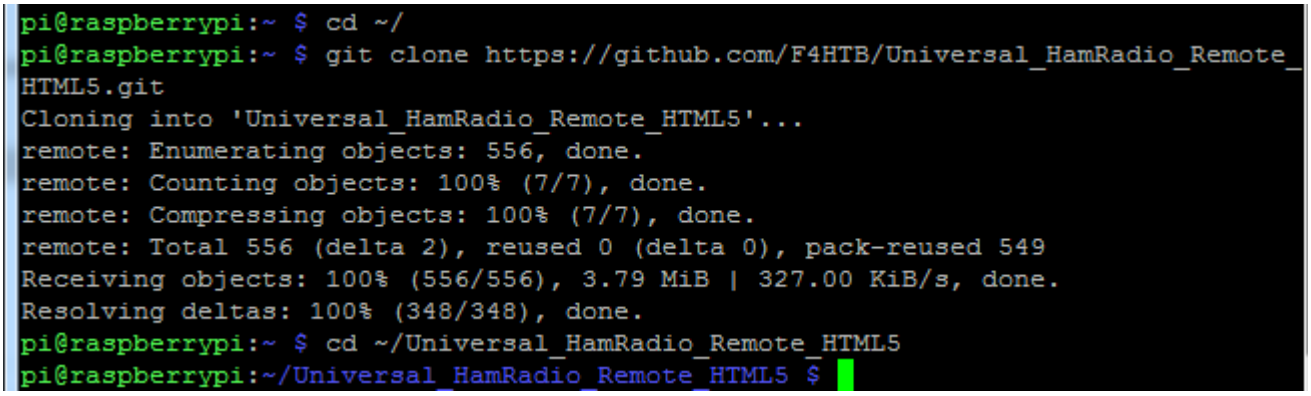
Restart the Raspberry PI, then reconnect in ssh with putty.

```
sudo reboot
```

```
   - have your system administrator add LIBDIR to '/etc/ld.so.conf'

See any operating system documentation about shared libraries for
more information, such as the ld(1) and ld.so(8) manual pages.
----------------------------------------------------------------------
 /usr/bin/mkdir -p '/usr/local/share/doc/hamlib/examples'
 /usr/bin/install -c -m 644 py3test.py '/usr/local/share/doc/hamlib/examples'
make[2]: Leaving directory '/home/pi/Hamlib/bindings'
make[1]: Leaving directory '/home/pi/Hamlib/bindings'
pi@raspberrypi:~ $ sudo reboot
```

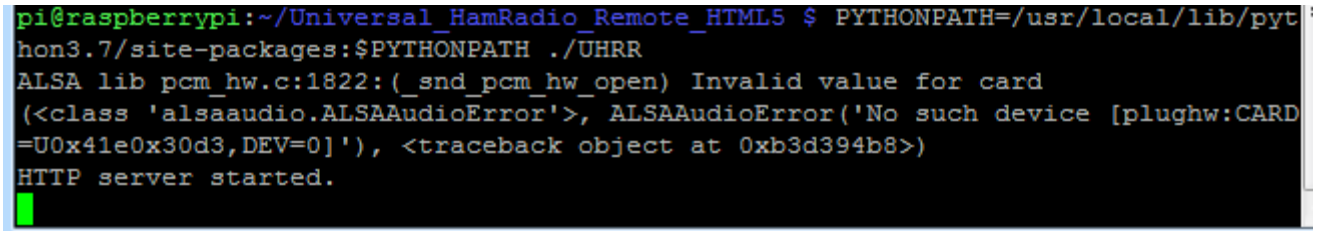
Install F4HTB software

```
cd ~/
git clone https://github.com/F4HTB/Universal_HamRadio_Remote_HTML5.git
```



Finish by starting the UHRR server

```
cd ~/Universal_HamRadio_Remote_HTML5
PYTHONPATH=/usr/local/lib/python3.7/site-packages:$PYTHONPATH ./UHRR
```



Console summary :

```
sudo apt-get update && sudo apt-get upgrade -y && sudo reboot
sudo apt-get install -y git python3 python3-pip python3-numpy python3-tornado python3-serial python3-pyaudio
rtl-sdr
sudo pip3 install pyalsaaudio pam pyrtlsdr
sudo apt-get autoremove -y --purge python3-libhamlib2
sudo apt-get install -y autoconf automake libtool swig
cd ~/
git clone https://github.com/Hamlib/Hamlib.git
cd Hamlib/
./bootstrap
./configure --with-python-binding PYTHON=$(which python3)
make all && sudo make install && cd bindings && make && sudo make install && sudo ldconfig
sudo reboot

cd ~/
git clone https://github.com/F4HTB/Universal_HamRadio_Remote_HTML5.git


cd ~/Universal_HamRadio_Remote_HTML5
PYTHONPATH=/usr/local/lib/python3.7/site-packages:$PYTHONPATH ./UHRR
```

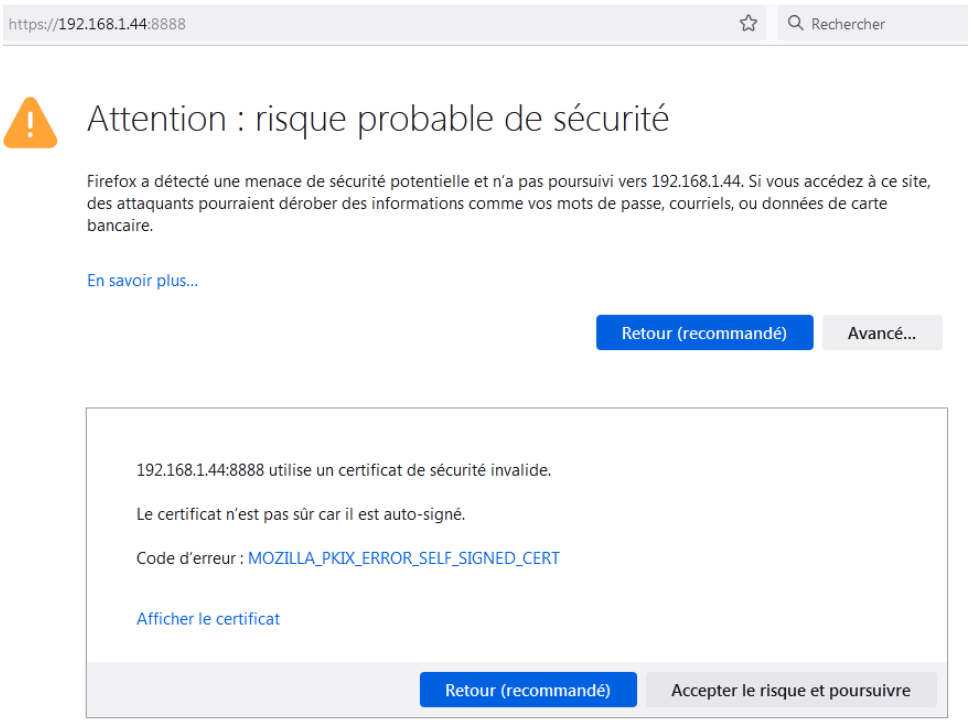
13.3 Basic Setup

With a browser (Firefox for example) access the server using the URL.

https://raspberrypi.local:8888/
ou:
https://192.168.1.44:8888/

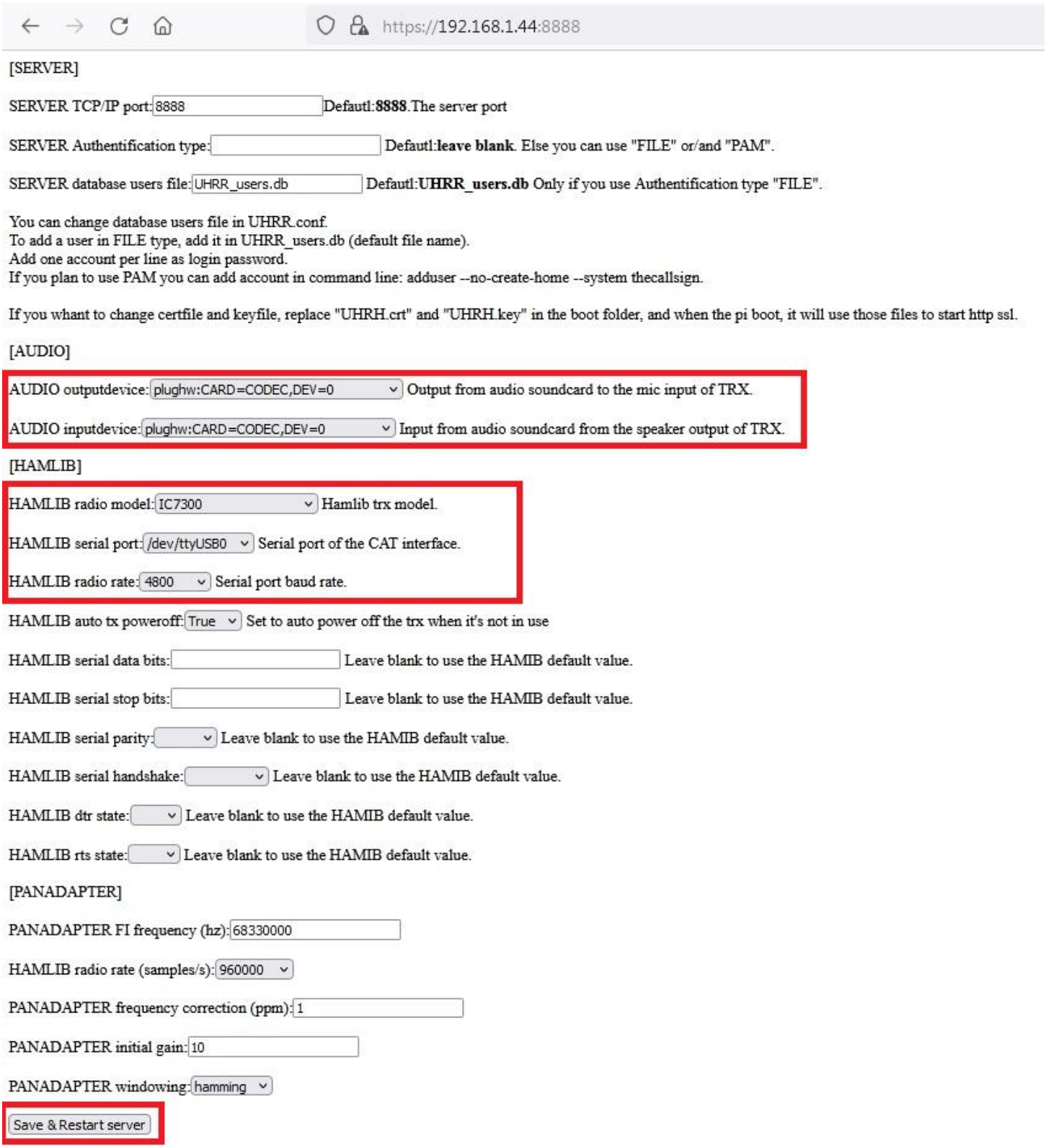
By specifying the IP address of your Raspberry pi, 192.168.1.44 is given as an example.

Add the security exception

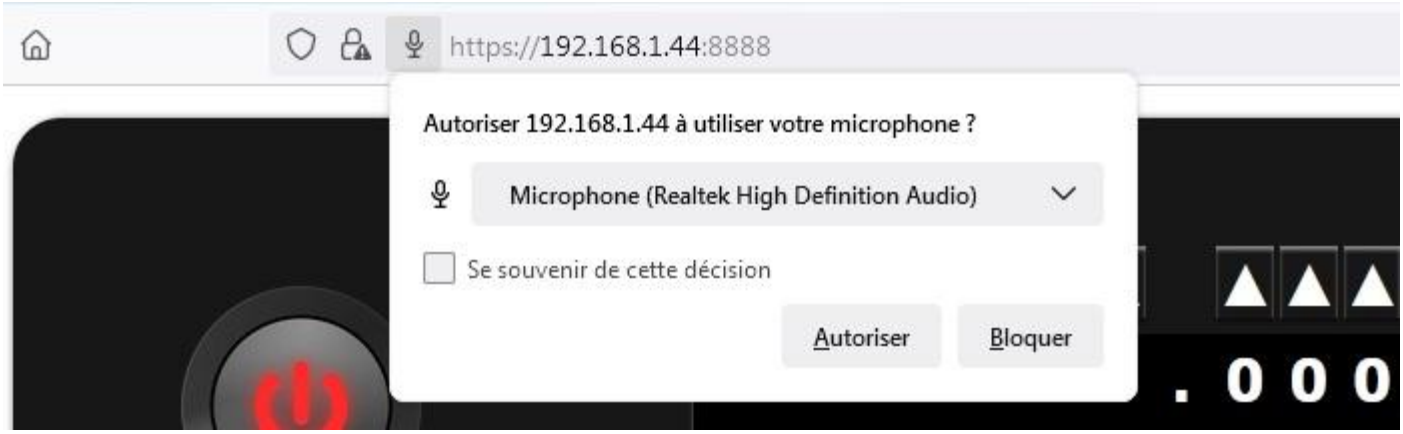


A configuration page appears. Configure the various menus concerning the sound card and the transceiver used.

Do not forget to click on "save and restart server" to save the configuration.

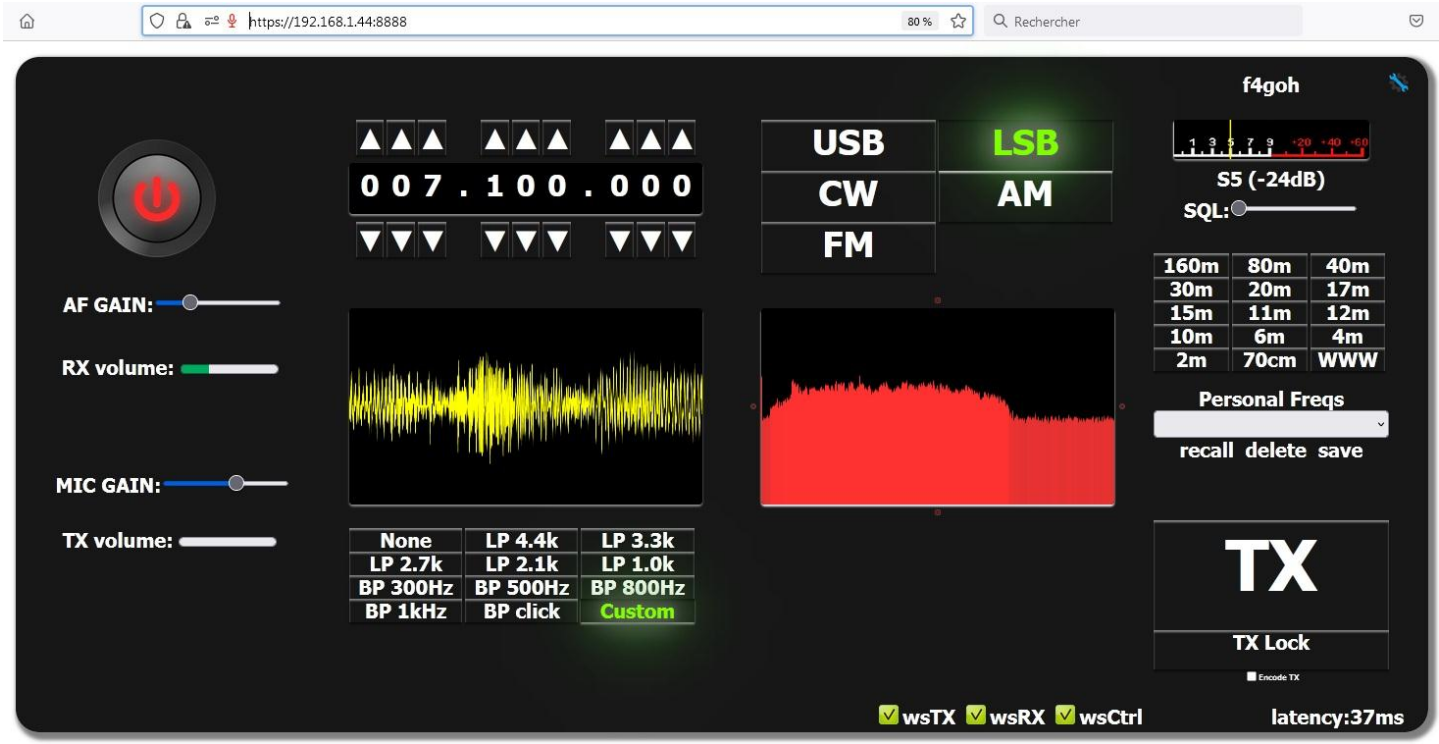


Do not forget to connect a headset microphone to the PC to allow the use of the microphone or other audio source.



Finally the long-awaited interface

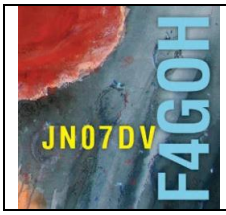
Click on the start button. The sound from the transceiver must be heard through the headphones. Otherwise, click on "the blue adjustable wrench" at the top right to return to the configuration menu.



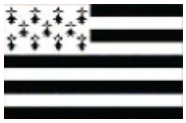
13.4 Conclusion

Congratulation Olivier (F4HTB) for this magnificent achievement. The graphical interface is very ergonomic and will delight OMs wishing to control their station remotely.

73 and good remote.



Igate APRS
Part 14



Summary :

Part 14 : Igate APRS

Prerequisite: Part 4: Using the Raspbian Lite operating system without a graphical interface.

The creator of the APRS system is Bob Bruninga, WB4APR.

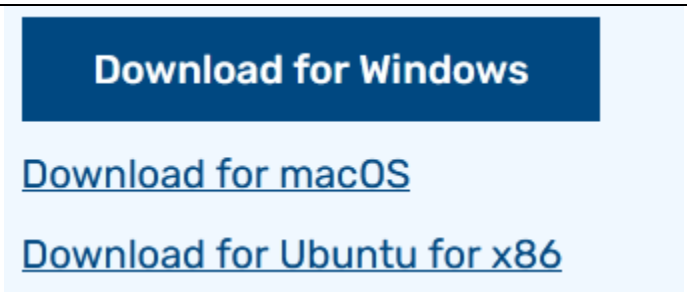
This document contains the information at the following link with screenshots.

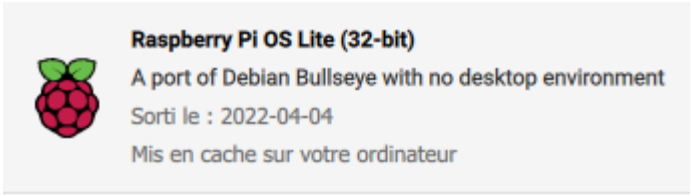
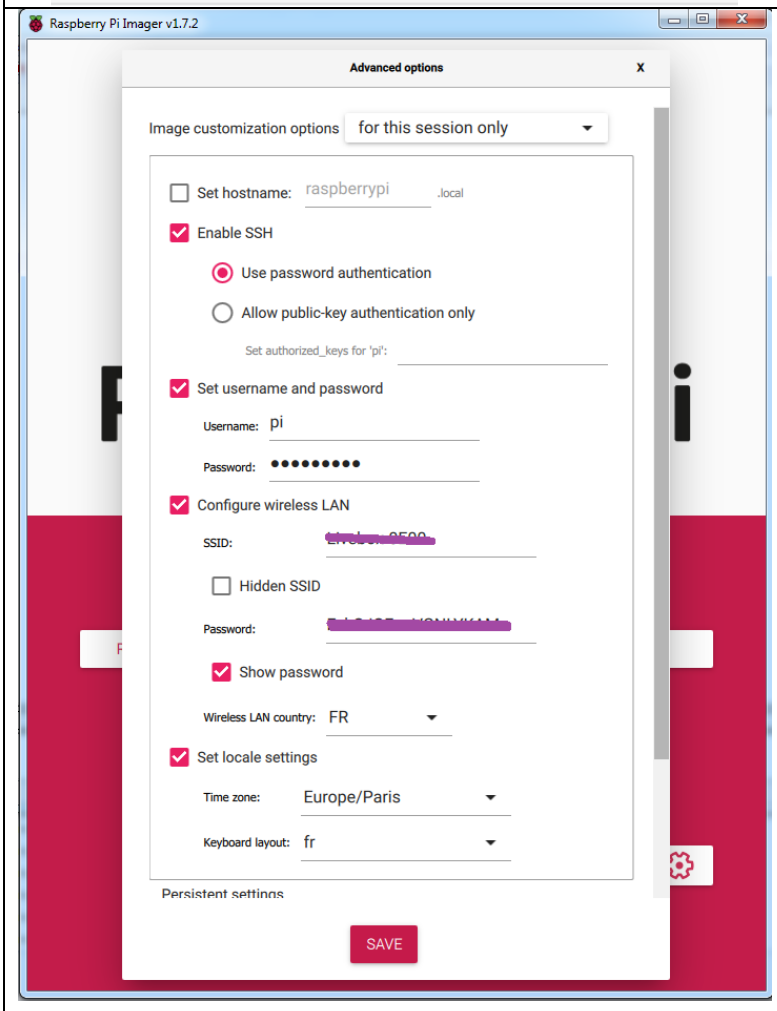
https://github.com/wb2osz/direwolf/blob/master/doc/Raspberry-Pi-SDR-IGate.pdf
https://github.com/wb2osz/direwolf/blob/master/doc/User-Guide.pdf

Version du 24/04/2022 V1.0 sur RPI4

14.1 Raspbian Lite operating system

Download the latest version of Pi Imager https://www.raspberrypi.com/software/

<table>
<tr><td colspan="2"> 

https://downloads.raspberrypi.org/imager/imager_latest.exe</td></tr>
<tr><td> 
</td><td>Choose Raspberry PI OS Li</td></tr>
<tr><td> 
</td><td>It is recommended to pre-configure the disk image:

- Enable SSH
Username: pi
Password: raspberry

- Configure WIFI access

- Change the time zone and keyboard configuration according to the country.</td></tr>
</table>

Save the configuration, then start writing to the mSd memory card.

14.2 Updating the operating system

Once the card has been inserted into the Raspberry Pi and powered up, identify the IP address of the Raspberry PI connected to your Wifi network (box in red).

Reminder: In the console window :

```
ifconfig
```

```
pi@raspberrypi:~ $ ifconfig
eth0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether dc:a6:32:e6:dc:dc  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 10  bytes 1600 (1.5 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 10  bytes 1600 (1.5 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.44  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::7bed:6545:85b3:3ef9  prefixlen 64  scopeid 0x20<link>
        ether dc:a6:32:e6:dc:dd  txqueuelen 1000  (Ethernet)
        RX packets 11530  bytes 15366617 (14.6 MiB)
        RX errors 0  dropped 55  overruns 0  frame 0
        TX packets 5973  bytes 594867 (580.9 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Performing an update of the packages :

```
sudo apt update
```

```
pi@raspberrypi:~ $ sudo apt update
Get:1 http://raspbian.raspberrypi.org/raspbian bullseye InRelease [15.0 kB]
Get:2 http://archive.raspberrypi.org/debian bullseye InRelease [23.7 kB]
Get:3 http://raspbian.raspberrypi.org/raspbian bullseye/main armhf Packages [13.
2 MB]
Get:4 http://archive.raspberrypi.org/debian bullseye/main armhf Packages [279 kB
]
Fetched 13.5 MB in 17s (782 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
10 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

Reminder: To copy and paste a line from the pdf file into putty :
- Select the line with the mouse;
- Right click and copy, or Ctrl+c ;
- Select the putty window with a left click;
- Select the Putty window with a left click; Then right click with the mouse.

```
sudo apt upgrade
```

```
pi@raspberrypi:~ $ sudo apt upgrade
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
The following packages will be upgraded:
  gzip liblzma5 libraspberrypi-bin libraspberrypi-dev libraspberrypi-doc
  libraspberrypi0 pi-bluetooth raspi-config userconf-pi xz-utils
```

14.3 RTL-SDR key installation (already described in part 3)

Start by installing git :

```
sudo apt-get install git
```

```
pi@raspberrypi:~ $ sudo apt-get install git
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  git-man liberror-perl
Suggested packages:
  git-daemon-run | git-daemon-sysvinit git-doc git-el git-email git-gui gitk gitweb git-cvs
  git-mediawiki git-svn
The following NEW packages will be installed:
  git git-man liberror-perl
0 upgraded, 3 newly installed, 0 to remove and 0 not upgraded.
Need to get 6,583 kB of archives.
After this operation, 33.1 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
```

```
git clone https://github.com/osmocom/rtl-sdr.git
```

```
pi@raspberrypi:~ $ git clone https://github.com/osmocom/rtl-sdr.git
Clonage dans 'rtl-sdr'...
remote: Enumerating objects: 65, done.
remote: Counting objects: 100% (65/65), done.
remote: Compressing objects: 100% (38/38), done.
remote: Total 1863 (delta 25), reused 55 (delta 23), pack-reused 1798
Réception d'objets: 100% (1863/1863), 448.80 KiB | 1.11 MiB/s, fait.
Résolution des deltas: 100% (1317/1317), fait.
pi@raspberrypi:~ $ ls
audio      Downloads    Music     qsstv      Templates  Videos
Desktop    GridTracker  Pictures  rtl-sdr    tx_drm
Documents  MagPi        Public    templates  tx_stock
pi@raspberrypi:~ $
```

Install the libraries.

```
sudo apt install build-essential cmake usbutils libusb-1.0-0-dev
```

```
pi@raspberrypi:~/rtl-sdr/build $ sudo apt install build-essential cmake usbutils
 libusb-1.0-0-dev
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
build-essential est déjà la version la plus récente (12.6).
usbutils est déjà la version la plus récente (1:010-3).
Les paquets suivants ont été installés automatiquement et ne sont plus nécessair
es :
  libmicrodns0 libqt5charts5 xlog-data
Veuillez utiliser « sudo apt autoremove » pour les supprimer.
Les paquets supplémentaires suivants seront installés :
  cmake-data libjsoncpp1 librhash0 libusb-1.0-doc libuv1
Paquets suggérés :
  cmake-doc ninja-build
Les NOUVEAUX paquets suivants seront installés :
  cmake cmake-data libjsoncpp1 librhash0 libusb-1.0-0-dev libusb-1.0-doc
  libuv1
0 mis à jour, 7 nouvellement installés, 0 à enlever et 0 non mis à jour.
Il est nécessaire de prendre 4□578 ko dans les archives.
Après cette opération, 23,6 Mo d'espace disque supplémentaires seront utilisés.
Souhaitez-vous continuer ? [O/n] O
```

Type these commands in order:

```
cd rtl-sdr

mkdir build

cd build

cmake -DINSTALL_UDEV_RULES=ON -DDETACH_KERNEL_DRIVER=ON ../
```

```
pi@raspberrypi:~ $ cd rtl-sdr/
pi@raspberrypi:~/rtl-sdr $ mkdir build
pi@raspberrypi:~/rtl-sdr $ cd build/
pi@raspberrypi:~/rtl-sdr/build $ cmake -DINSTALL_UDEV_RULES=ON -DDETACH_KERNEL_D
RIVER=ON ../
-- The C compiler identification is GNU 8.3.0
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compiler: /usr/bin/cc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Build type not specified: defaulting to release.
-- Extracting version information from git describe...
-- Looking for pthread.h
-- Looking for pthread.h - found
```

```
sudo make install
```

```
pi@raspberrypi:~/rtl-sdr/build $ sudo make install
Scanning dependencies of target convenience_static
[  3%] Building C object src/CMakeFiles/convenience_static.dir/convenience/conve
nience.c.o
[  6%] Linking C static library libconvenience_static.a
[  6%] Built target convenience_static
Scanning dependencies of target rtlsdr
[  9%] Building C object src/CMakeFiles/rtlsdr.dir/librtlsdr.c.o
[ 12%] Building C object src/CMakeFiles/rtlsdr.dir/tuner_e4k.c.o
[ 15%] Building C object src/CMakeFiles/rtlsdr.dir/tuner_fc0012.c.o
[ 18%] Building C object src/CMakeFiles/rtlsdr.dir/tuner_fc0013.c.o
[ 21%] Building C object src/CMakeFiles/rtlsdr.dir/tuner_fc2580.c.o
[ 25%] Building C object src/CMakeFiles/rtlsdr.dir/tuner_r82xx.c.o
[ 28%] Linking C shared library librtlsdr.so
```

```
sudo ldconfig
```

```
-- Set runtime path of "/usr/local/bin/rtl_fm" to ""
-- Installing: /usr/local/bin/rtl_eeprom
-- Set runtime path of "/usr/local/bin/rtl_eeprom" to ""
-- Installing: /usr/local/bin/rtl_adsb
-- Set runtime path of "/usr/local/bin/rtl_adsb" to ""
-- Installing: /usr/local/bin/rtl_power
-- Set runtime path of "/usr/local/bin/rtl_power" to ""
pi@raspberrypi:~/rtl-sdr/build $ sudo ldconfig
pi@raspberrypi:~/rtl-sdr/build $
```

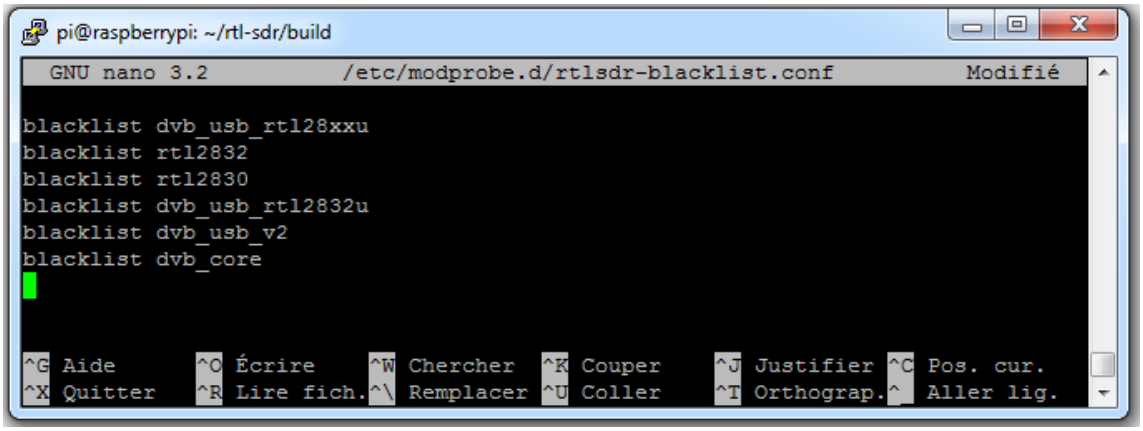
```
sudo nano /etc/modprobe.d/rtlsdr-blacklist.conf
```

```
pi@raspberrypi:~/rtl-sdr/build $ sudo nano /etc/modprobe.d/rtlsdr-blacklist.conf
```

```
blacklist dvb_usb_rtl28xxu
blacklist rtl2832
blacklist rtl2830
blacklist dvb_usb_rtl2832u
blacklist dvb_usb_v2
blacklist dvb_core
```

Copy and paste the above list into the nano editor

Right click to paste the text into the nano editor



Save the file using the Ctrl + o keys, then the enter key to validate the saving.	**Exit the editor with Ctrl + x.**
	

Reboot the Raspberry PI

```
pi@raspberrypi:~/rtl-sdr/build $ sudo reboot
```

```
sudo reboot
```

Verification: Plug the rtl-sdr into a USB port on the Raspberry Pi.
With LXTerminal or Putty, type the following command line:

```
lsusb
```

Locate the rtl-sdr key: **RTL238 DVB-T**

Case of the Raspberry Pi 3 :

```
pi@raspberrypi:~ $ lsusb
Bus 001 Device 004: ID 0bda:2838 Realtek Semiconductor Corp. RTL2838 DVB-T
Bus 001 Device 003: ID 0424:ec00 Standard Microsystems Corp. SMSC9512/9514 Fast
Ethernet Adapter
Bus 001 Device 002: ID 0424:9514 Standard Microsystems Corp. SMC9514 Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

Case of the Raspberry Pi 4 :

```
pi@raspberrypi:~ $ lsusb
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 003: ID 0bda:2838 Realtek Semiconductor Corp. RTL2838 DVB-T
Bus 001 Device 002: ID 2109:3431 VIA Labs, Inc. Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

With LXTerminal or Putty, type the following command line: (beware ofundescore key)

```
rtl_test
```

The rtl-sdr key should be recognised, if not, disconnect and reconnect the key and repeat the test.

```
pi@raspberrypi:~ $ rtl_test
Found 1 device(s):
  0:  Realtek, RTL2838UHIDIR, SN: 00000001

Using device 0: Generic RTL2832U OEM
Found Rafael Micro R820T tuner
Supported gain values (29): 0.0 0.9 1.4 2.7 3.7 7.7 8.7 12.5 14.4 15.7 16.6 19.7
 20.7 22.9 25.4 28.0 29.7 32.8 33.8 36.4 37.2 38.6 40.2 42.1 43.4 43.9 44.5 48.0
 49.6
[R82XX] PLL not locked!
Sampling at 2048000 S/s.

Info: This tool will continuously read from the device, and report if
samples get lost. If you observe no further output, everything is fine.

Reading samples in async mode...
^CSignal caught, exiting!
ctrl+c

User cancel, exiting...
Samples per million lost (minimum): 0
pi@raspberrypi:~ $ ^C
```

Exit the program by pressing the Ctrl key and the c key simultaneously (Ctrl+c)

If **`rtl_test`** displays this message in a loop `lost at least 112 bytes`, reinstall the program using the manual method.

14.4 Direwolf installation

The best linux software to manage APRS is Direwolf, designed by John, wb2osz. This one works very well with an RTL-SDR key.

Checking the prerequisites: Normally these utilities are already installed in the operating system, but it is still prudent to check.

```
sudo apt-get install gcc
sudo apt-get install g++
sudo apt-get install make
sudo apt-get install cmake
```

```
pi@raspberrypi:~ $ sudo apt-get install gcc
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
gcc est déjà la version la plus récente (4:8.3.0-1+rpi2).
gcc passé en « installé manuellement ».
Le paquet suivant a été installé automatiquement et n'est plus nécessaire :
  python-colorzero
Veuillez utiliser « sudo apt autoremove » pour le supprimer.
0 mis à jour, 0 nouvellement installés, 0 à enlever et 0 non mis à jour.
pi@raspberrypi:~ $ sudo apt-get install g++
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
g++ est déjà la version la plus récente (4:8.3.0-1+rpi2).
g++ passé en « installé manuellement ».
Le paquet suivant a été installé automatiquement et n'est plus nécessaire :
  python-colorzero
Veuillez utiliser « sudo apt autoremove » pour le supprimer.
0 mis à jour, 0 nouvellement installés, 0 à enlever et 0 non mis à jour.
pi@raspberrypi:~ $ sudo apt-get install make
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
make est déjà la version la plus récente (4.2.1-1.2).
make passé en « installé manuellement ».
Le paquet suivant a été installé automatiquement et n'est plus nécessaire :
  python-colorzero
Veuillez utiliser « sudo apt autoremove » pour le supprimer.
0 mis à jour, 0 nouvellement installés, 0 à enlever et 0 non mis à jour.
pi@raspberrypi:~ $ sudo apt-get install cmake
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
cmake est déjà la version la plus récente (3.16.3-3~bpo10+1).
Le paquet suivant a été installé automatiquement et n'est plus nécessaire :
  python-colorzero
Veuillez utiliser « sudo apt autoremove » pour le supprimer.
0 mis à jour, 0 nouvellement installés, 0 à enlever et 0 non mis à jour.
pi@raspberrypi:~ $
```

Installing sound libraries

`sudo apt-get install libasound2-dev`

```
pi@raspberrypi:~ $ sudo apt install libasound2-dev
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Le paquet suivant a été installé automatiquement et n'est plus nécessaire :
  python-colorzero
Veuillez utiliser « sudo apt autoremove » pour le supprimer.
Paquets suggérés :
  libasound2-doc
Les NOUVEAUX paquets suivants seront installés :
  libasound2-dev
0 mis à jour, 1 nouvellement installés, 0 à enlever et 0 non mis à jour.
Il est nécessaire de prendre 145 ko dans les archives.
Après cette opération, 546 ko d'espace disque supplémentaires seront utilisés.
Réception de :1 http://archive.raspberrypi.org/debian buster/main armhf libasoun
d2-dev armhf 1.1.8-1+rpt1 [145 kB]
145 ko réceptionnés en 1s (155 ko/s)
Sélection du paquet libasound2-dev:armhf précédemment désélectionné.
(Lecture de la base de données... 44317 fichiers et répertoires déjà installés.)
Préparation du dépaquetage de .../libasound2-dev_1.1.8-1+rpt1_armhf.deb ...
Dépaquetage de libasound2-dev:armhf (1.1.8-1+rpt1) ...
Paramétrage de libasound2-dev:armhf (1.1.8-1+rpt1) ...
pi@raspberrypi:~ $
```

```
sudo apt-get install libudev-dev
```

```
pi@raspberrypi:~ $ sudo apt install libudev-dev
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Le paquet suivant a été installé automatiquement et n'est plus nécessaire :
  python-colorzero
Veuillez utiliser « sudo apt autoremove » pour le supprimer.
Les NOUVEAUX paquets suivants seront installés :
  libudev-dev
0 mis à jour, 1 nouvellement installés, 0 à enlever et 0 non mis à jour.
Il est nécessaire de prendre 110 ko dans les archives.
Après cette opération, 202 ko d'espace disque supplémentaires seront utilisés.
Réception de :1 http://ftp.igh.cnrs.fr/pub/os/linux/raspbian/raspbian buster/main a
mhf libudev-dev armhf 241-7~deb10u8+rpi1 [110 kB]
110 ko réceptionnés en 1s (79,5 ko/s)
Sélection du paquet libudev-dev:armhf précédemment désélectionné.
(Lecture de la base de données... 44388 fichiers et répertoires déjà installés.)
Préparation du dépaquetage de .../libudev-dev_241-7~deb10u8+rpi1_armhf.deb ...
Dépaquetage de libudev-dev:armhf (241-7~deb10u8+rpi1) ...
Paramétrage de libudev-dev:armhf (241-7~deb10u8+rpi1) ...
Traitement des actions différées (« triggers ») pour man-db (2.8.5-2) ...
pi@raspberrypi:~ $
```

Download the Direwolf source code using git

```
git clone https://www.github.com/wb2osz/direwolf
```

```
pi@raspberrypi:~ $ git clone https://www.github.com/wb2osz/direwolf
Clonage dans 'direwolf'...
warning: redirection vers https://github.com/wb2osz/direwolf.git/
remote: Enumerating objects: 2945, done.
remote: Counting objects: 100% (105/105), done.
remote: Compressing objects: 100% (74/74), done.
remote: Total 2945 (delta 43), reused 67 (delta 30), pack-reused 2840
Réception d'objets: 100% (2945/2945), 134.50 MiB | 154.00 KiB/s, fait.
Résolution des deltas: 100% (1998/1998), fait.
pi@raspberrypi:~ $
```

Go to the Direwolf directory, then create a build directory.

```
cd direwolf
git checkout dev
mkdir build && cd build
```

```
pi@raspberrypi:~ $ cd direwolf
pi@raspberrypi:~/direwolf $ git checkout dev
Branch 'dev' set up to track remote branch 'dev' from 'origin'.
Switched to a new branch 'dev'
pi@raspberrypi:~/direwolf $ mkdir build && cd build
pi@raspberrypi:~/direwolf/build $
```

Compile and install the program

cmake ..

```
pi@raspberrypi:~/direwolf/build $ cmake ..
-- The C compiler identification is GNU 8.3.0
^[[D-- The CXX compiler identification is GNU 8.3.0
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compiler: /usr/bin/cc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Check for working CXX compiler: /usr/bin/c++
-- Check for working CXX compiler: /usr/bin/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Found Git: /usr/bin/git (found version "2.20.1")
-- Dire Wolf Version: 1.7.0-b66c21d
-- Build type set to: Release
CMake system: Linux
-- Target architecture: ARM
-- Use NEON SIMD instructions
-- Looking for strlcpy
-- Looking for strlcpy - not found
-- Looking for strlcat
-- Looking for strlcat - not found
```

make -j4

```
pi@raspberrypi:~/direwolf/build $ make -j4
Scanning dependencies of target misc
Scanning dependencies of target geotranz
[  1%] Building C object external/misc/CMakeFiles/misc.dir/strlcat.c.o
[  1%] Building C object external/misc/CMakeFiles/misc.dir/strlcpy.c.o
[  2%] Building C object external/geotranz/CMakeFiles/geotranz.dir/error_string.c.o
[  2%] Building C object external/geotranz/CMakeFiles/geotranz.dir/mgrs.c.o
[  2%] Building C object external/geotranz/CMakeFiles/geotranz.dir/polarst.c.o
[  3%] Linking C static library libmisc.a
[  4%] Building C object external/geotranz/CMakeFiles/geotranz.dir/tranmerc.c.o
[  4%] Built target misc
Scanning dependencies of target text2tt
[  4%] Building C object src/CMakeFiles/text2tt.dir/tt_text.c.o
[  4%] Building C object external/geotranz/CMakeFiles/geotranz.dir/ups.c.o
[  5%] Building C object external/geotranz/CMakeFiles/geotranz.dir/usng.c.o
[  5%] Building C object external/geotranz/CMakeFiles/geotranz.dir/utm.c.o
Scanning dependencies of target atest
[  5%] Building C object src/CMakeFiles/atest.dir/atest.c.o
[  6%] Building C object src/CMakeFiles/atest.dir/ais.c.o
[  8%] Linking C executable text2tt
[ 96%] Building C object src/CMakeFiles/direwolf.dir/xmit.c.o
[ 97%] Building C object src/CMakeFiles/direwolf.dir/dwgpsd.c.o
[ 97%] Building C object src/CMakeFiles/direwolf.dir/dwgpsnmea.c.o
[ 98%] Building C object src/CMakeFiles/direwolf.dir/mheard.c.o
[ 98%] Building C object src/CMakeFiles/direwolf.dir/audio.c.o
[100%] Building C object src/CMakeFiles/direwolf.dir/cm108.c.o
[100%] Linking C executable direwolf
[100%] Built target direwolf
pi@raspberrypi:~/direwolf/build $
```

```
sudo make install
```

```
pi@raspberrypi:~/direwolf/build $ sudo make install
[  4%] Built target geotranz
[  6%] Built target misc
[  9%] Built target ll2utm
[ 10%] Built target text2tt
[ 12%] Built target utm2ll
[ 28%] Built target atest
[ 31%] Built target aclients
[ 40%] Built target decode_aprs
[ 44%] Built target kissutil
[ 45%] Built target log2gpx
[ 47%] Built target tt2text
[ 55%] Built target gen_packets
[ 91%] Built target direwolf
[ 93%] Built target cm108
[ 95%] Built target ttcalc
[100%] Built target appserver
Install the project...
-- Install configuration: "Release"
-- Installing: /usr/local/share/doc/direwolf/CHANGES.md
-- Installing: /usr/local/share/doc/direwolf/LICENSE
-- Installing: /usr/local/share/doc/direwolf/external/LICENSE
-- Installing: /usr/local/share/applications/direwolf.desktop
-- Installing: /usr/local/share/pixmaps/direwolf_icon.png
```

```
make install-conf
```

```
pi@raspberrypi:~/direwolf/build $ make install-conf
Scanning dependencies of target install-conf
Built target install-conf
pi@raspberrypi:~/direwolf/build $
```

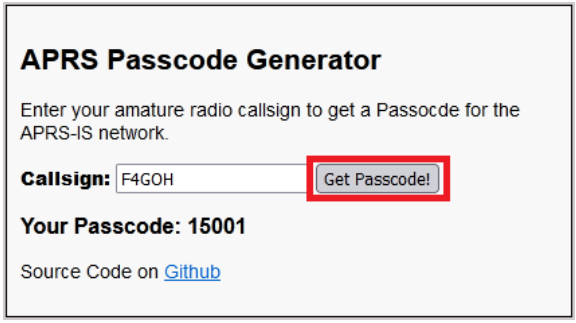
14.5 Direwolf setup with the RTL-SDR

Return to the /home/pi directory
You can see that sample Direwolf configurations have been created. These files end with the **`.conf`** extension.

```
cd ~
pwd
ls
```

```
pi@raspberrypi:~/direwolf/build $ cd ~
pi@raspberrypi:~ $ pwd
/home/pi
pi@raspberrypi:~ $ ls
direwolf       dw-start.sh  sdr.conf           telem-m0xer-3.txt
direwolf.conf  rtl-sdr      telem-balloon.conf  telem-volts.conf
pi@raspberrypi:~ $
```

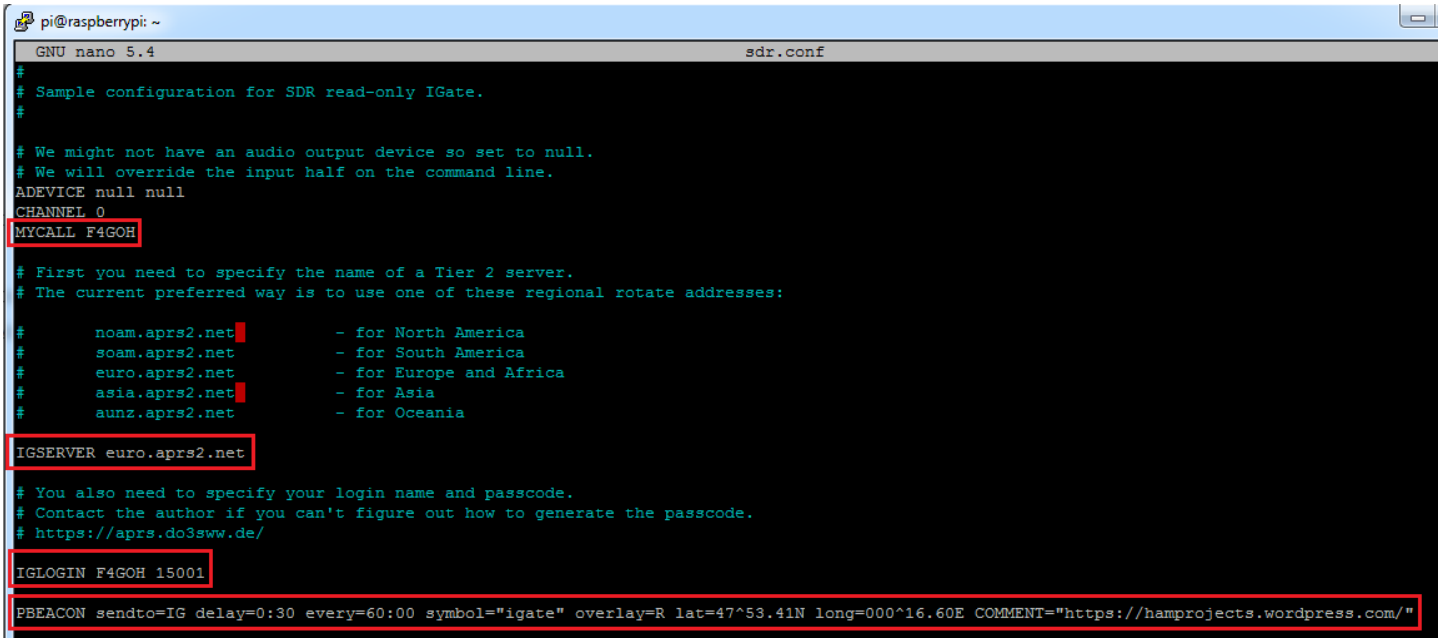
Obtain an encrypted password for the APRS server using the following address
https://aprs.do3sww.de/



Edit the sdr.conf file

```
nano sdr.conf
```

Change the lines according to your callsign and geographical location.



It is possible to add the command PBEACON which will report the position of the igate on
https://aprs.fi

```
MYCALL F4GOH

IGSERVER euro.aprs2.net

IGLOGIN F4GOH 15001

PBEACON sendto=IG delay=0:30 every=60:00 symbol="igate" overlay=R lat=47^53.41N
long=000^16.60E COMMENT="https://hamprojects.wordpress.com/"
```

PBEACON must be on the same line!

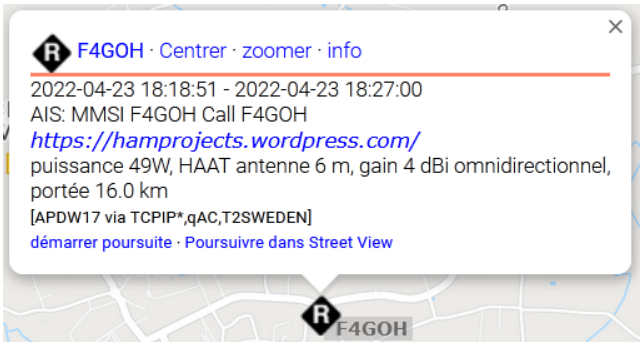
14.6 Configuration test

Run the following command:

```
rtl_fm -f 144.80M - | direwolf -c sdr.conf -r 24000 -D 1 -
```

```
pi@raspberrypi:~ $ rtl_fm -f 144.80M - | direwolf -c sdr.conf -r 24000 -D 1 -
Dire Wolf DEVELOPMENT version 1.7 E (Apr 23 2022)
Includes optional support for:  cm108-ptt

Reading config file sdr.conf
Audio input device for receive: stdin  (channel 0)
Audio out device for transmit: null  (channel 0)
Channel 0: 1200 baud, AFSK 1200 & 2200 Hz, A+, 24000 sample rate.
Note: PTT not configured for channel 0. (Ignore this if using VOX.)
Ready to accept AGW client application 0 on port 8000 ...
Ready to accept KISS TCP client application 0 on port 8001 ...
Found 1 device(s):
  0:  Realtek, RTL2838UHIDIR, SN: 00000001

Using device 0: Generic RTL2832U OEM
Found Rafael Micro R820T tuner
Tuner gain set to automatic.
Tuned to 145052000 Hz.
Oversampling input by: 42x.
Oversampling output by: 1x.
Buffer size: 8.13ms
Exact sample rate is: 1008000.009613 Hz
Sampling at 1008000 S/s.
Output at 24000 Hz.

Now connected to IGate server euro.aprs2.net (95.216.197.75)
Check server status here http://95.216.197.75:14501

[ig] # aprsc 2.1.10-gd72a17c
[ig] # logresp F4GOH verified, server T2SWEDEN
[ig] F4GOH>APDW17:!4753.41NR00016.60E&https://hamprojects.wordpress.com/
```

Then check that the igate appears on https://aprs.fi



If the command line is mistyped or copied and pasted incorrectly, the program will not run correctly as the following example shows:

```
pi@raspberrypi:~ $ rtl_fm -f 144.80M - | direwolf -c sdr.conf -r 24000 -D 1 -
Dire Wolf DEVELOPMENT version 1.7 E (Apr 23 2022)
Includes optional support for:  cm108-ptt

Reading config file sdr.conf
Audio input device for receive: -  (channel 0)
Audio out device for transmit: null  (channel 0)
ALSA lib pcm.c:2660:(snd_pcm_open_noupdate) Unknown PCM -
Could not open audio device - for input
No such file or directory
Pointless to continue without audio device.
Found 1 device(s):
  0:  Realtek, RTL2838UHIDIR, SN: 00000001

Using device 0: Generic RTL2832U OEM
Found Rafael Micro R820T tuner
Tuner gain set to automatic.
Tuned to 145052000 Hz.
Oversampling input by: 42x.
Oversampling output by: 1x.
Buffer size: 8.13ms
Exact sample rate is: 1008000.009613 Hz
Sampling at 1008000 S/s.
Output at 24000 Hz.
Signal caught, exiting!

User cancel, exiting...
```

The other solution is to download a ready-made script, change the execution rights and then run it.

```
wget https://github.com/f4goh/DRAPRS/raw/master/config_direwolf/sdr/runsdr.sh
chmod +x runsdr.sh
./runsdr.sh
```

```
pi@raspberrypi:~ $ wget https://github.com/f4goh/DRAPRS/raw/master/config_direwo
lf/sdr/runsdr.sh
--2022-04-24 11:19:30--  https://github.com/f4goh/DRAPRS/raw/master/config_direw
olf/sdr/runsdr.sh
Resolving github.com (github.com)... 140.82.121.3
```

```
pi@raspberrypi:~ $ cat runsdr.sh
rtl_fm -f 144.80M - | direwolf -c sdr.conf -r 24000 -D 1 -
```

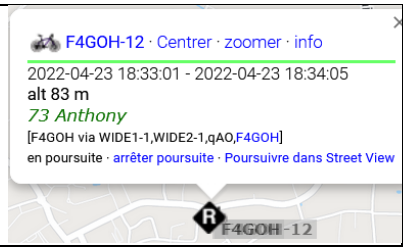
```
pi@raspberrypi:~ $ chmod +x runsdr.sh
pi@raspberrypi:~ $ ./runsdr.sh
```

Use an [APRS Tracker] to send a test frame to confirm correct decoding.

Use to send a test frame, to confirm correct decoding.

My F4GOH-12 tracker appears well in the console and on [https://aprs.fi]

```
[ig] # aprsc 2.1.10-gd72a17c
[ig] # logresp F4GOH verified, server T2SWEDEN
[ig] F4GOH>APDW17:!4753.41NR00016.60E&https://hamprojects.wordpress.com/

F4GOH-12 audio level = 29(6/6)    [NONE]    ||||||___
[0.2] F4GOH-12>F4GOH,WIDE1-1,WIDE2-1:/163230h4753.39N/00016.61Eb/A=-00156/73 Anth
Position with time, BIKE
N 47 53.3900, E 000 16.6100
/A=-00156/73 Anthony
```



14.7 Run Direwolf at the RPI startup démarrage du RPI

Exit the program with Ctrl + c.

```
F4GOH-12 audio level = 29(6/6)    [NONE]    |||||____
[0.2] F4GOH-12>F4GOH,WIDE1-1,WIDE2-1:/163630h4753.41N/00016.61Eb/A=000273/73 Anthony
Position with time, BIKE
N 47 53.4100, E 000 16.6100, alt 83 m (273 ft)
/73 Anthony
^C
QRT
Signal caught, exiting!

User cancel, exiting...

End of file on stdin.  Exiting.
pi@raspberrypi:~ $
```

Ctrl+c

Install the screen software

```
sudo apt install screen
```

```
pi@raspberrypi:~ $ sudo apt install screen
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libutempter0
Suggested packages:
  byobu | screenie | iselect
The following NEW packages will be installed:
  libutempter0 screen
0 upgraded, 2 newly installed, 0 to remove and 0 not upgraded.
Need to get 560 kB of archives.
After this operation, 983 kB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://mirrors.ircam.fr/pub/raspbian/raspbian bullseye/main armhf
 1.2.1-2 [8,596 B]
```

Rename the existing `dw-start.sh` file to dw-start.sh.bak, so as not to lose it.
Download the configured `dw-start.sh` script and change the execution rights.

```
mv dw-start.sh dw-start.sh.bak
wget https://github.com/f4goh/DRAPRS/raw/master/config_direwolf/sdr/dw-start.sh
chmod +x dw-start.sh
```

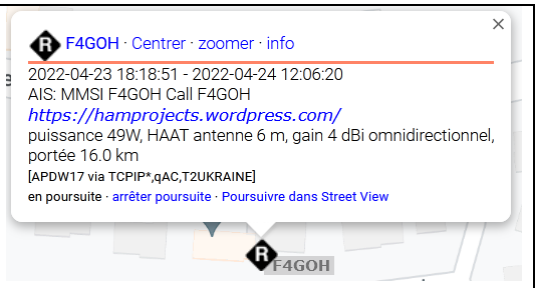
Run the `dw-start.sh` file, then wait 30 seconds

```
./dw-start.sh
```

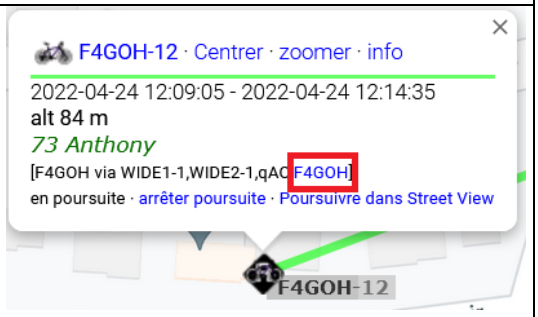
Once the script is active, check again at https://aprs.fi to see if the beacon frame has been sent.

Once the script is active, check again on sending the tag frame.

```
pi@raspberrypi:~ $ ./dw-start.sh
Direwolf in GUI mode start up
DISPLAY=:0
Did not find an X terminal emulator.  Reverting to CLI mode
-----------------------
Direwolf in CLI mode start up
There is a screen on:
        805.direwolf    (23/04/22 19:03:22)     (Detached)
1 Socket in /run/screen/S-pi.
-----------------------
pi@raspberrypi:~ $
```



Vérifier également le report sur l'igate avec un tracker.



Edit the crontab task scheduler by choosing option 1 (nano editor)

```
crontab -e
```

```
pi@raspberrypi:~ $ crontab -e
no crontab for pi - using an empty one

Select an editor.  To change later, run 'select-editor'.
  1. /bin/nano        <---- easiest
  2. /usr/bin/vim.tiny
  3. /bin/ed

Choose 1-3 [1]: 1
```

At the end of the file add the following line

```
* * * * * /home/pi/dw-start.sh >/dev/null 2>&1
```

```
pi@raspberrypi: ~
  GNU nano 5.4                          /tmp/crontab.ujt4x6/crontab
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command

* * * * * /home/pi/dw-start.sh >/dev/null 2>&1

^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute     ^C Location    M-U Undo
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify     ^_ Go To Line  M-E Redo
```

Reminder:

Save the file using the Ctrl + o keys, then the enter key to validate the saving.	**Exit the editor using Ctrl + x..**
	

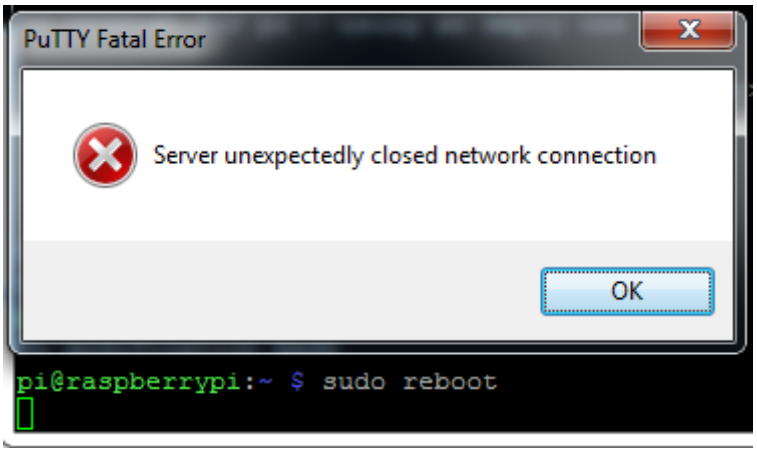
A message appears after exiting the editor "`installing new crontab`".

```
pi@raspberrypi:~ $ crontab -e
no crontab for pi - using an empty one

Select an editor.  To change later, run 'select-editor'.
  1. /bin/nano        <---- easiest
  2. /usr/bin/vim.tiny
  3. /bin/ed

Choose 1-3 [1]: 1
crontab: installing new crontab
```

Reboot the Raspberry

```
sudo reboot
```

Direwolf will run in the background when the Raspberry PI is rebooted.



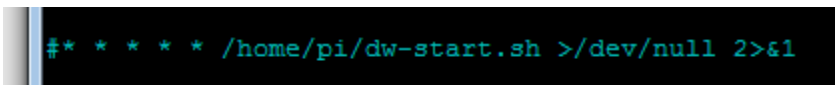
Wait one minute, then check again on https://aprs.fithat the sending of the beacon and tracker frames is working properly.

2022-04-24 12:24:05 CEST: **F4GOH-12**>F4GOH,WIDE1-1,WIDE2-1,qAO,F4GOH:/102400h4753.41N/00016.60Eb/A=000276/73 Anthony
2022-04-24 12:24:35 CEST: **F4GOH-12**>F4GOH,WIDE1-1,WIDE2-1,qAO,F4GOH:/102430h4753.41N/00016.60Eb/A=000276/73 Anthony
2022-04-24 12:25:05 CEST: **F4GOH-12**>F4GOH,WIDE1-1,WIDE2-1,qAO,F4GOH:/102500h4753.41N/00016.60Eb/A=000276/73 Anthony

To stop the task scheduler :

```
crontab -e
```

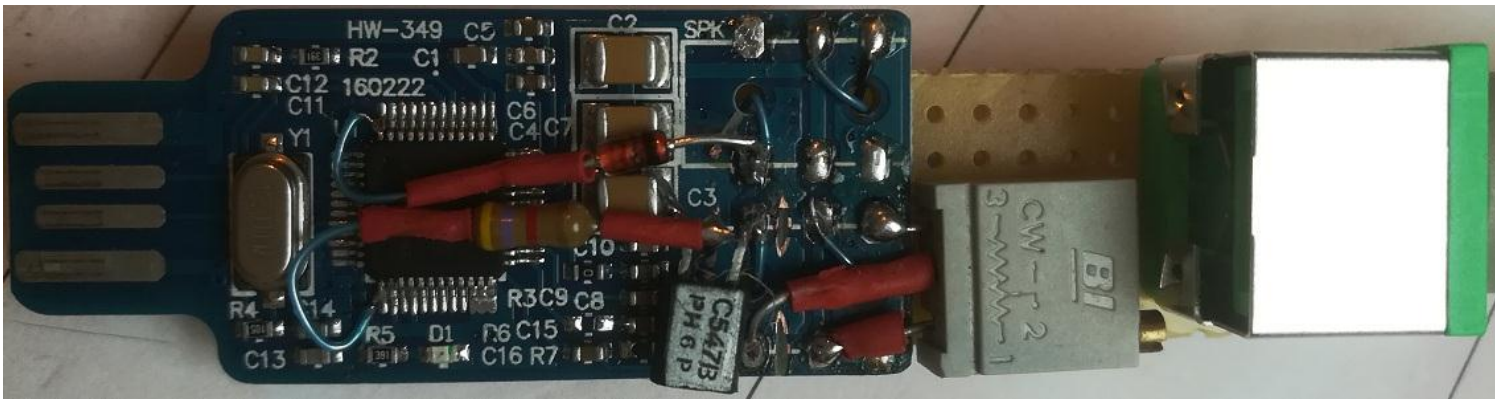
Add a # to the beginning of the line, then restart the RPI.




```
sudo reboot
```

14.8 Direwolf setup with an external USB sound card

The hardware used for the following will be the modified cm108 USB sound card described in tutorial 11.



Plug the sound card into the Raspberry PI, then check for PTT control compatibility (1) and note the number of the recognised sound card in the RPI (2).

```
crontab -e
```

```
pi@raspberrypi:~ $ cm108
    VID  PID   Product                         Sound                ADEVICE  2   ADEVICE           HID [ptt]
    ---  ---   -------                         -----                -------      -------           ---------
**  0d8c 000c  C-Media USB Headphone Set       /dev/snd/pcmC3D0c    plughw:3,0   plughw:Set,0      /dev/hidraw0
**  0d8c 000c  C-Media USB Headphone Set       /dev/snd/pcmC3D0p    plughw:3,0   plughw:Set,0      /dev/hidraw0
**  0d8c 000c  C-Media USB Headphone Set       /dev/snd/controlC3                                  /dev/hidraw0

** = Can use Audio Adapter GPIO for PTT.   1

Notice that each USB Audio adapter is assigned a number and a name.  These are not predictable so you could
end up using the wrong adapter after adding or removing other USB devices or after rebooting.  You can assign a
name to each USB adapter so you can refer to the same one each time.  This can be based on any characteristics
that makes them unique such as product id or serial number.  Unfortunately these devices don't have unique serial
numbers so how can we tell them apart?  A name can also be assigned based on the physical USB socket.
Create a file like "/etc/udev/rules.d/85-my-usb-audio.rules" with the following contents and then reboot.

SUBSYSTEM!="sound", GOTO="my_usb_audio_end"
ACTION!="add", GOTO="my_usb_audio_end"
DEVPATH=="/devices/platform/scb/fd500000.pcie/pci0000:00/0000:00:00.0/0000:01:00.0/usb1/1-1/1-1.4/1-1.4:1.0/sound/card?", ATTR{id}="Fred"
LABEL="my_usb_audio_end"

pi@raspberrypi:~ $
```

Rename the original direwolf.conf file, then upload the preconfigured direwolf.conf file

```
mv direwolf.conf direwolf.conf.bak
wget https://github.com/f4goh/DRAPRS/raw/master/config_direwolf/soundcard/direwolf.conf
```

```
pi@raspberrypi:~ $ wget https://github.com/f4goh/DRAPRS/raw/master/config_direwolf/soundcard/direwolf.conf
--2022-04-24 14:20:37--  https://github.com/f4goh/DRAPRS/raw/master/config_direwolf/soundcard/direwolf.conf
Resolving github.com (github.com)... 140.82.121.3
Connecting to github.com (github.com)|140.82.121.3|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://raw.githubusercontent.com/f4goh/DRAPRS/master/config_direwolf/soundcard/direwolf.conf [following]
--2022-04-24 14:20:38--  https://raw.githubusercontent.com/f4goh/DRAPRS/master/config_direwolf/soundcard/direwolf.conf
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.110.133, 185.199.111.133, 185.199.109.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.110.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 16114 (16K) [text/plain]
Saving to: 'direwolf.conf'

direwolf.conf                   100%[=============================================================>]  15.74K  --.-KB/s    in 0.01s

2022-04-24 14:20:38 (1.17 MB/s) - 'direwolf.conf' saved [16114/16114]
```

Custom configuration of the direwolf.conf file.

Edit the direwolf.conf file.

nano direwolf.conf

Indicate the number of the sound card. (Here in the RPI4, the external sound card is number 3)

```
# and a card number reported by "arecord -l" command.  Example:

ADEVICE  plughw:3,0

# You can also use "-" or "stdin" to pipe stdout from
```

Changing the callsign

```
# Example (don't use this unless you are me):  MYCALL   WB2OSZ-5
#

MYCALL F4GOH
```

Activate PTT control via the cm108 sound card

```
# If using a C-Media CM108/CM119 or similar USB Audio Adapter,
# you can use a GPIO pin for PTT control.  This is very convenient
# because a single USB connection is used for both audio and PTT.
# Example:

PTT CM108
```

Send a signal transmitting the position of the igate. (Optional and useful if the RPI is not connected to the internet)

```
# This results in a broadcast once every 10 minutes.
# Every half hour, it can travel via two digipeater hops.
# The others are kept local.
#

PBEACON delay=1  every=30 overlay=S symbol="digi" lat=47^53.42N long=000^16.60E power=50 height=20 gain=4 comment="Teloche" via=WIDE1-1,WIDE2-1
#PBEACON delay=11 every=30 overlay=S symbol="digi" lat=42^37.14N long=071^20.83W power=50 height=20 gain=4 comment="Chelmsford MA"
#PBEACON delay=21 every=30 overlay=S symbol="digi" lat=42^37.14N long=071^20.83W power=50 height=20 gain=4 comment="Chelmsford MA"
```

Activate the digipeater. (Radio transmission of heard frames)

```
#
# For most common situations, use something like this by removing
# the "#" from the beginning of the line below.
#

DIGIPEAT 0 0 ^WIDE[3-7]-[1-7]$|^TEST$ ^WIDE[12]-[12]$ TRACE
```

Activate the igate and send the position of the igate (PBEACON) to the internet every 60 minutes.

```
IGSERVER euro.aprs2.net

# You also need to specify your login name and passcode.
# Contact the author if you can't figure out how to generate the passcode.

IGLOGIN F4GOH 15001

# That's all you need for a receive only IGate which relays
# messages from the local radio channel to the global servers.

# Some might want to send an IGate client position directly to a server
# without sending it over the air and relying on someone else to
# forward it to an IGate server.  This is done by using sendto=IG rather
# than a radio channel number. Overlay R for receive only, T for two way.

PBEACON sendto=IG delay=0:30 every=60:00 symbol="igate" overlay=R lat=47^53.42N long=000^16.60E COMMENT="https://hamprojects.wordpress.com/"
#PBEACON sendto=IG delay=0:30 every=60:00 symbol="igate" overlay=T lat=42^37.14N long=071^20.83W
```

Save the file Ctrl+o, Ctrl+x

Run the direwolf program, it loads the configuration file direwolf.conf by default

direwolf

```
pi@raspberrypi:~ $ direwolf
Dire Wolf DEVELOPMENT version 1.7 E (Apr 23 2022)
Includes optional support for:  cm108-ptt

Reading config file direwolf.conf
Audio device for both receive and transmit: plughw:3,0  (channel 0)
Channel 0: 1200 baud, AFSK 1200 & 2200 Hz, A+, 44100 sample rate / 3.
Using /dev/hidraw0 GPIO 3 for channel 0 PTT control.
Ready to accept AGW client application 0 on port 8000 ...
Ready to accept KISS TCP client application 0 on port 8001 ...

Now connected to IGate server euro.aprs2.net (44.141.143.40)
Check server status here http://44.141.143.40:14501

[ig] # aprsc 2.1.5-g8af3cdc
[ig] # logresp F4GOH verified, server T2NORWAY
[ig] F4GOH>APDW17:!4753.42NR00016.60E&https://hamprojects.wordpress.com/
[0L] F4GOH>APDW17,WIDE1-1,WIDE2-1:!4753.42NS00016.60E#PHG7140Teloche

F4GOH-12 audio level = 24(6/4)    [NONE]    __|||||__
[0.4] F4GOH-12>F4GOH,WIDE1-1,WIDE2-1:/123000h4753.41N/00016.61Eb/A=000229/73 Anthony
Position with time, BIKE
N 47 53.4100, E 000 16.6100, alt 70 m (229 ft)
/73 Anthony
[0H] F4GOH-12>F4GOH,F4GOH*,WIDE2-1:/123000h4753.41N/00016.61Eb/A=000229/73 Anthony
```

Check on https://aprs.fi that the sending of the beacon and tracker frames is working properly.

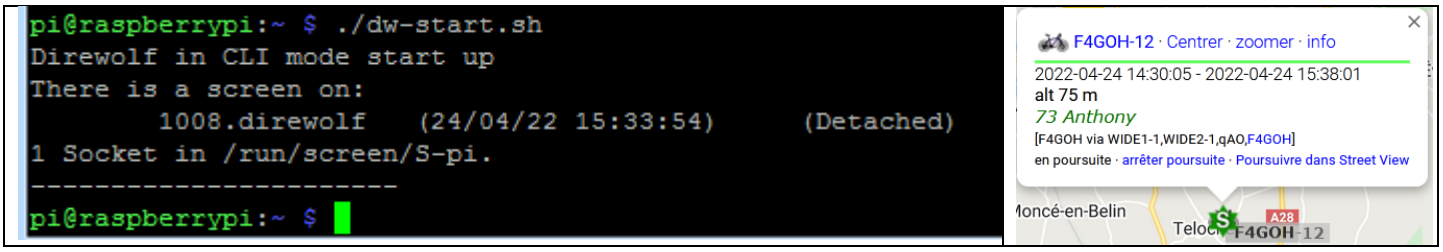
Rename the existing `dw-start.sh` file to `dw-start.sh.bak`, so as not to lose it.
Download the configured `dw-start.sh` script and change the execution rights.

```
mv dw-start.sh dw-start.sh.bak
wget https://github.com/f4goh/DRAPRS/raw/master/config_direwolf/soundcard/dw-start.sh
chmod +x dw-start.sh
```

Run the `dw-start.sh` file, then wait 30 seconds

```
./dw-start.sh
```

Once the script is active, check again at https://aprs.fi to see if the beacon frame has been sent.



Repeat the configuration of the "crontab" task scheduler described on pages 17, 18 and 19.

14.9 Conclusion

The APRS protocol is still widely used today. Its implementation requires few components and there is a lot of source code on the net. The Direwolf program is a good example, even if its configuration is sometimes a bit complex. However, the configuration possibilities are numerous. With an RTL-SDR key or a modified sound card, it will be easy to dedicate an igate at low cost.

Direwolf has supported the FX25 for several years.

https://en.wikipedia.org/wiki/FX.25_Forward_Error_Correction

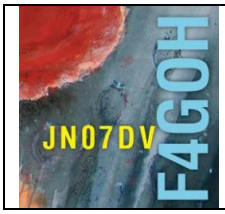
The source code for a probe balloon tracker supporting the FX25 is available at the following address

https://github.com/PhilippeSimier/Radiocommunication/tree/master/projet_ballon

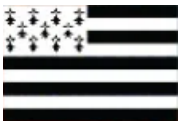
This tracker based on an (ESP32+DRA818) was programmed by Philippe and Anthony (F4GOH)

Direwolf is able to decode the FX25 by adding the `-X 1` option to the "`dw-start.sh`" file

```
# 2. FX.25 Forward Error Correction (FEC) will allow your signal to
#    go farther under poor radio conditions.  Add "-X 1" to the command
line.

DWCMD="$DIREWOLF -a 100 -X 1"
```



ADS-B with Piaware
Part 15



Prerequisite: Discovering Raspberry PI Part 1

Summary :

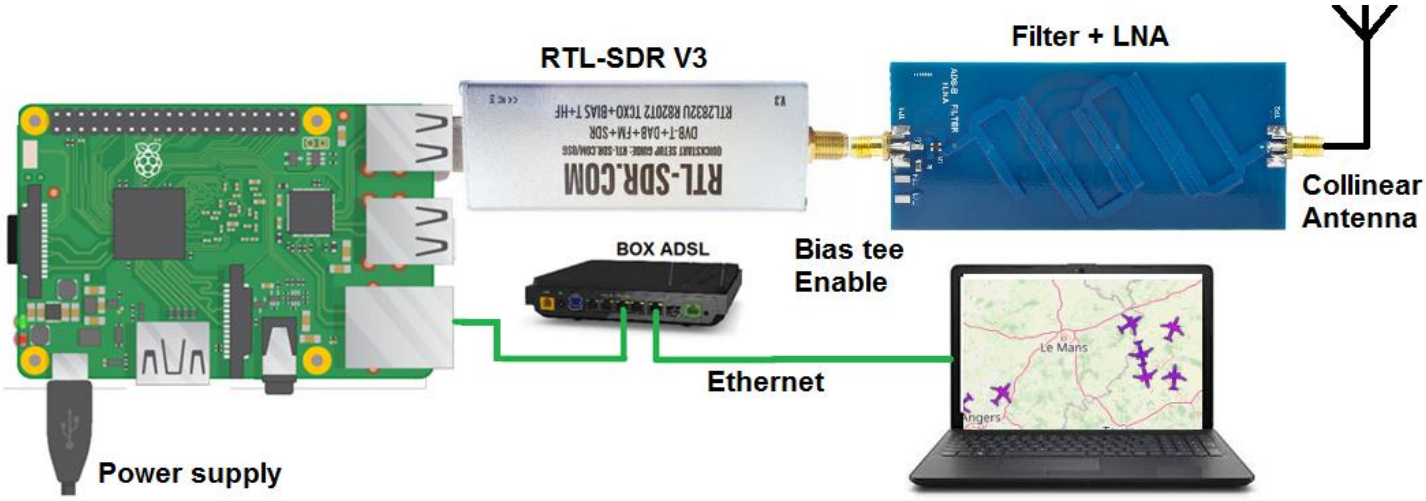
Part 15: ADS-B with Piaware.

Version du 02/08/2022 V1.0

15.1 Introduction.

Automatic Dependent Surveillance-Broadcast (ADS-B) is a cooperative surveillance system for air traffic control. An ADS-B equipped aircraft determines its position via a satellite positioning system and periodically sends this position and other information to ground stations and other ADS-B equipped aircraft operating in the area.
ADS-B is mandatory for large aircraft with a departure weight of more than 5.7 tons or a cruising speed of more than 250 KTAS (Knots True Airspeed).

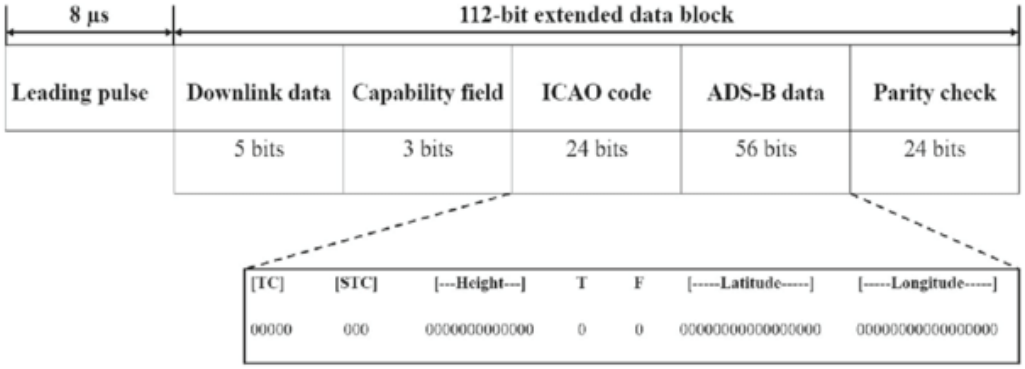
Watch the video introduction: https://youtu.be/BDLFHdq540g



In order to simplify the software installation as much as possible, I use the Piaware environment provided by flightaware.com

Transmission protocol:

The aircraft sends a series of pulses on 1090 Mhz. After a preamble, the useful message is coded in Manchester format.



The rate is 1Mbits/s and the packet contain either 56 or 112 bits.

For more information on decoding a frame:

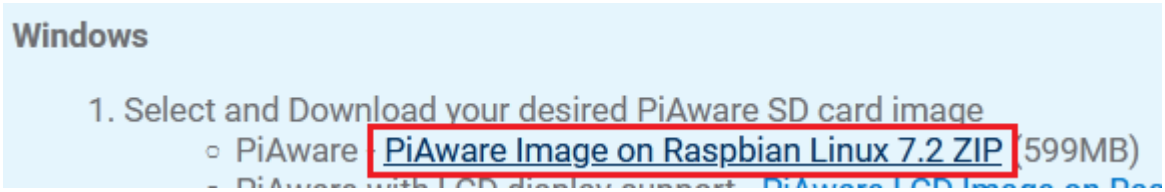
https://mode-s.org/decode/content/ads-b/1-basics.html

15.2 Downloading and installing the Piaware image.

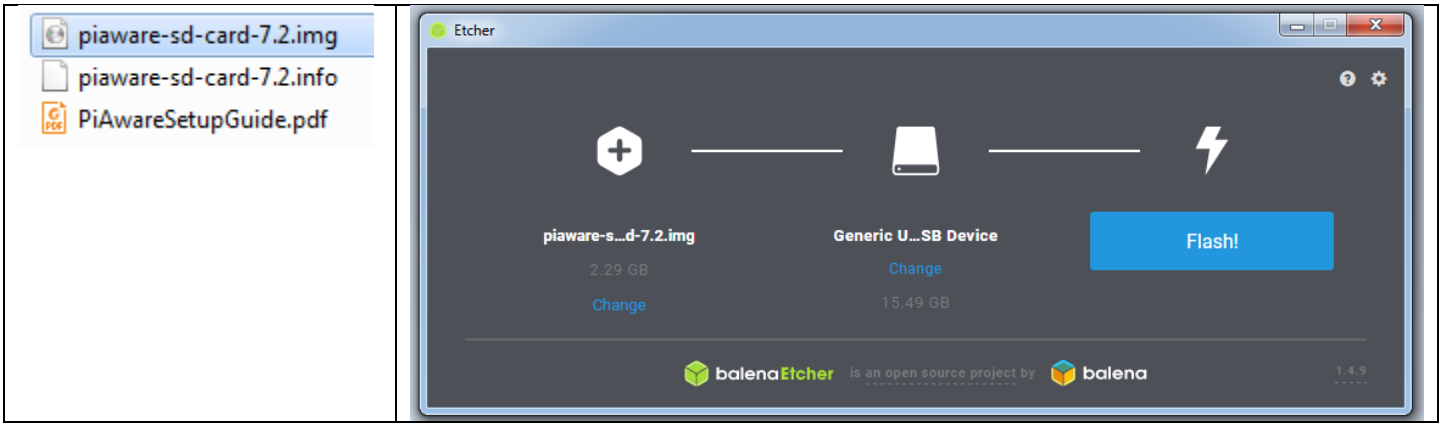
Go to the download page:

https://fr.flightaware.com/adsb/piaware/build/ or https://flightaware.com/adsb/piaware/build/

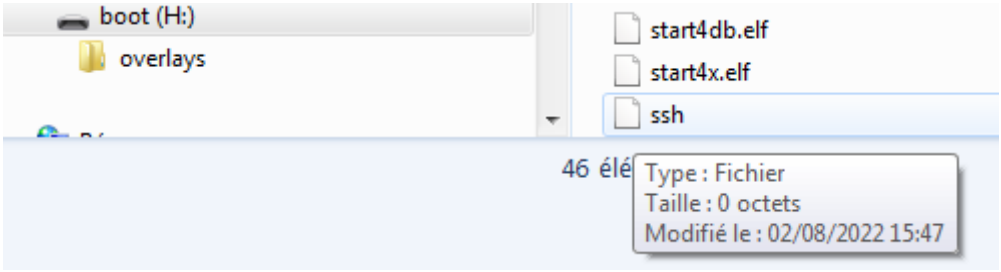
Then download the zip file without LCD support.



Unzip the zip file and locate the **`piaware-sd-card-7.2.img`**, then use https://www.balena.io/etcher to copy the image to an MSD card.

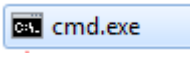


Finish by enabling SSh by creating an empty file on the /boot partition of the SD card with the filename "ssh" only (no file extension). When this file is present, SSh will be automatically activated.

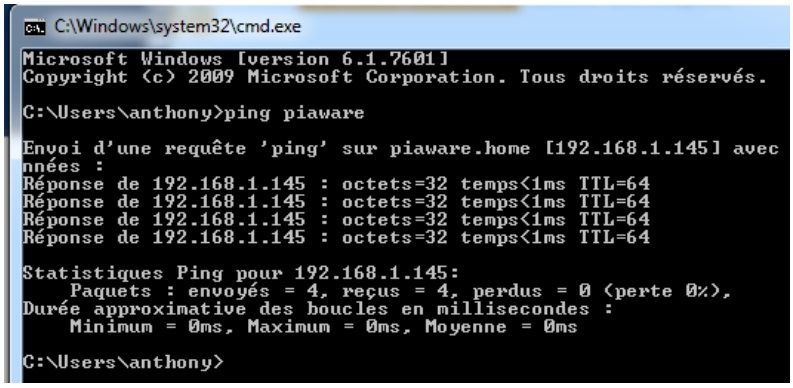


15.3 Basic Configuration

Boot the Raspberry Pi with the rtl-sdr key connected. Start the command prompt on your machine, then type:




```
ping piaware
```



This allows you to know the IP address of the Raspberry Pi (RPI) on the local network.

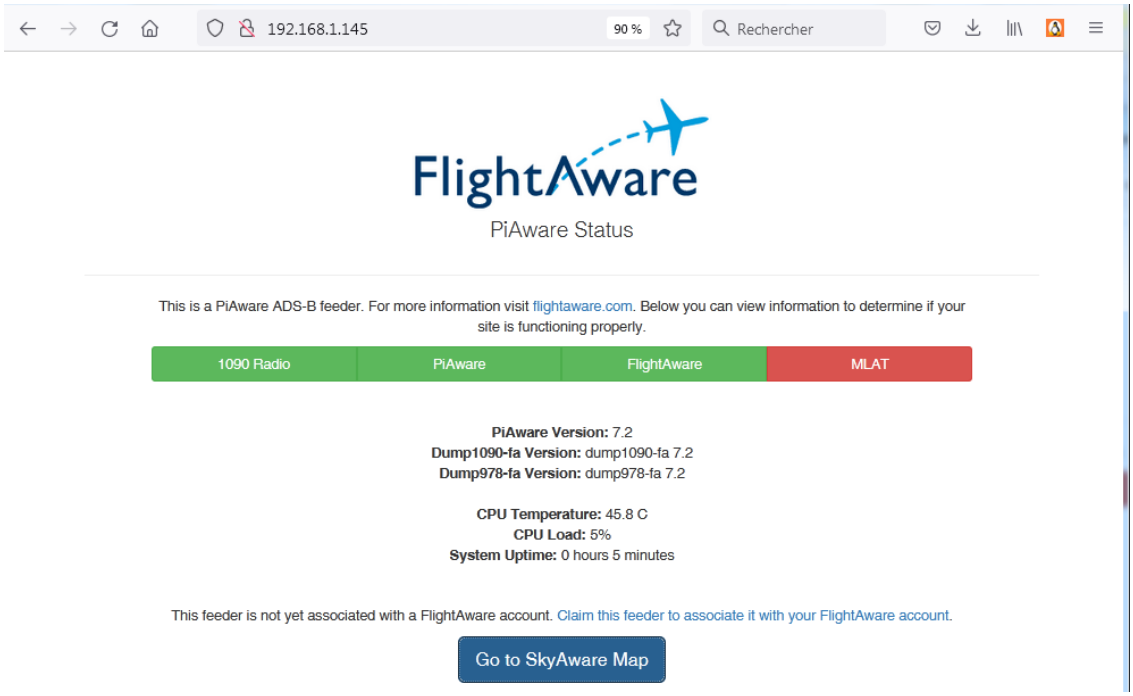
With a browser such as Firefox, enter the following URL:

http://piaware/

Or if that doesn't work, the IP address of the RPI. (IP 192.168.1.145 is given as an example).

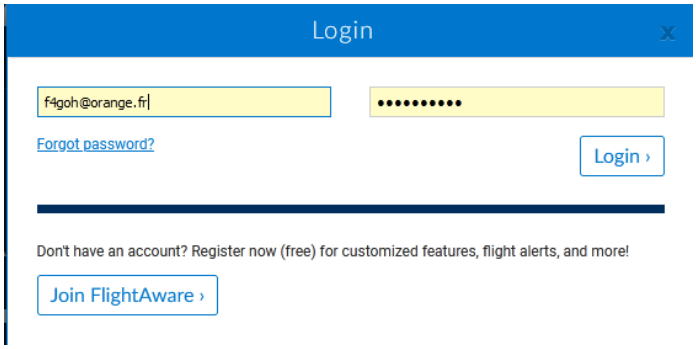
http://192.168.1.145/

The following homepage should appear with 3 green blocks and one red block.



Link your receiving station by creating an account with FlightAware. To do this click on:

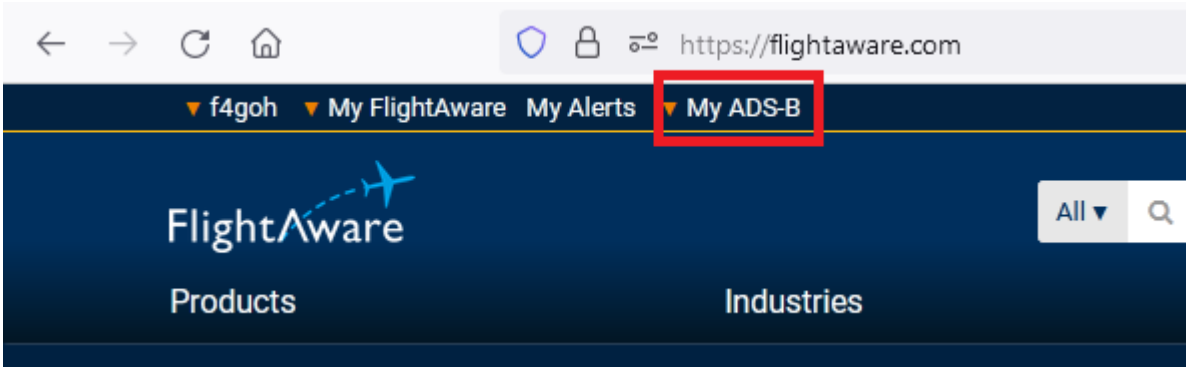
Claim this feeder to associate it with your flight account.



Once the registration is complete, this page should appear along with a confirmation email.

A link between the RPI and your account will be automatically created.

Check your account (main page) for a new menu "My ADS-B".

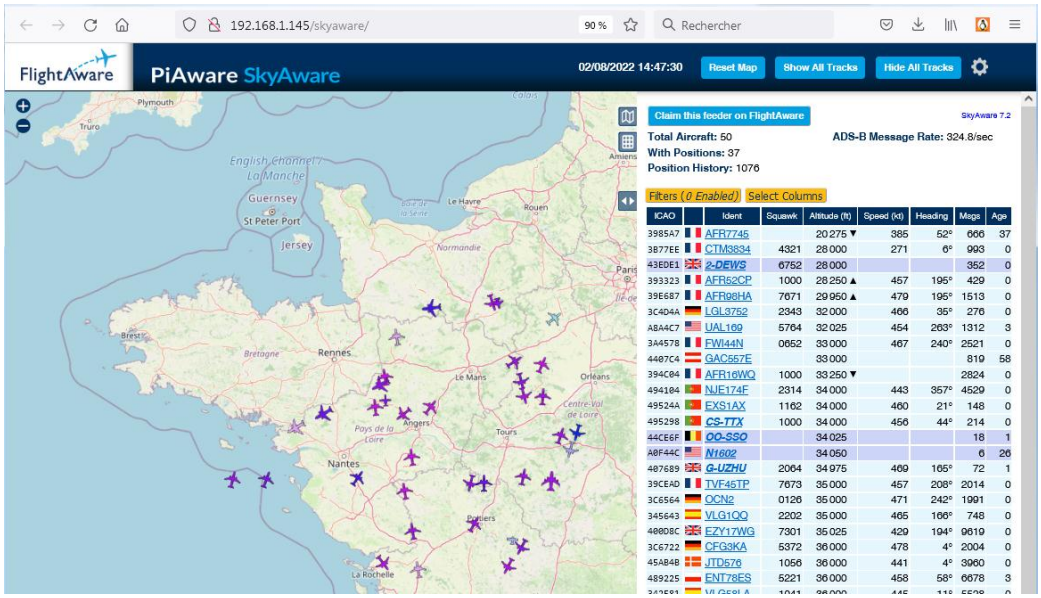


15.4 Checking reception

Return to the local RPI page (page 4), then click on 

.

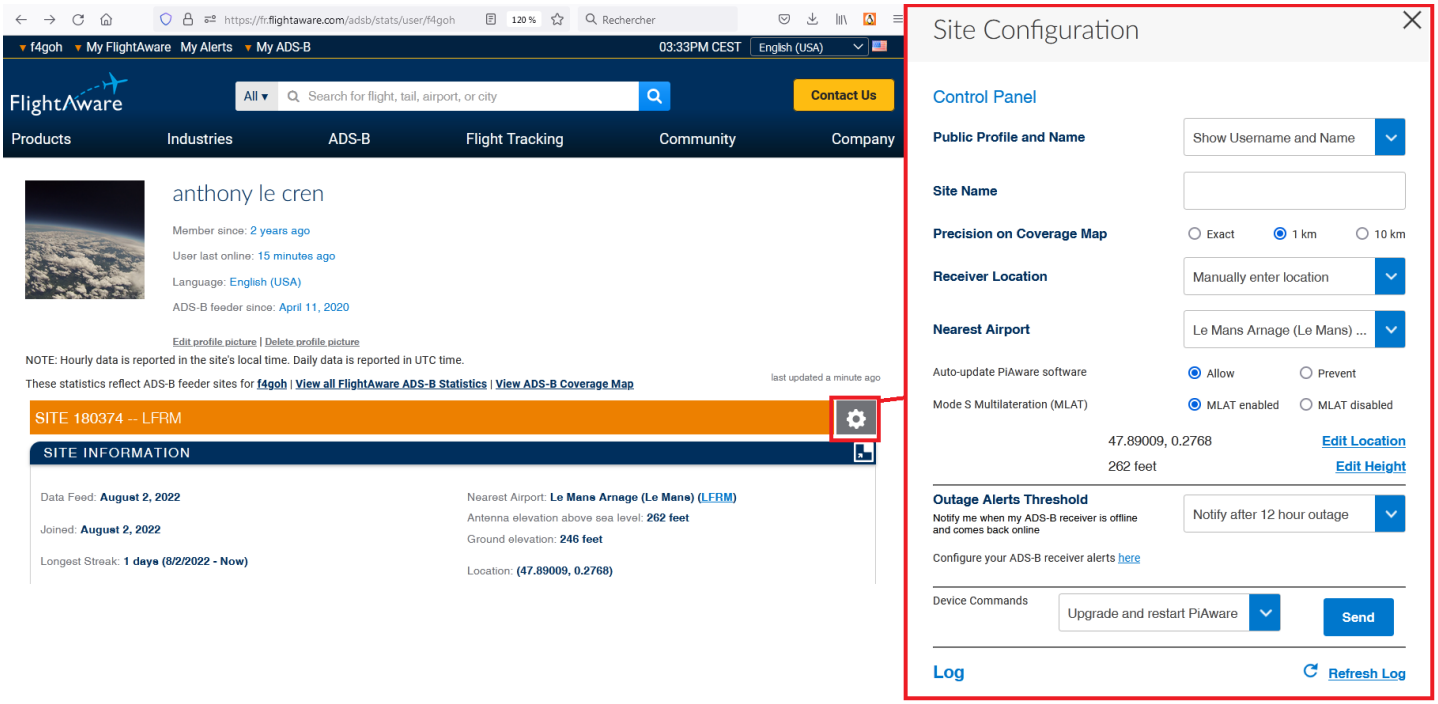
The position of the aircraft should be displayed together with the flight information in the right sidebar.



15.5 Locating the listening station

Go to the statistics page (of course change f4goh to your login :).

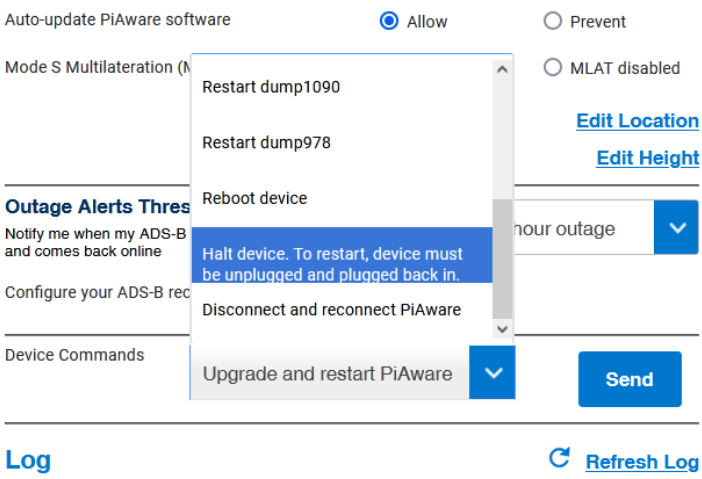
https://flightaware.com/adsb/stats/user/f4goh



Change the geographical coordinates in the "site configuration" menu (Edit Location...).

Note:

This menu also has an option to remotely shut down or restart the RPI.



Back on the local RPI page http://piaware/ or http://192.168.1.145/, all features are in green.

15.6 Build the antenna

I will not describe in this document the construction of the 1090Mhz collinear antenna, but rather the URL links that allowed me to build it with coaxial cable (RG58 cable is to be avoided, too small in diameter and too much loss).

http://f4ffd.hamstation.eu/Antenne.Colineaire.ADS-B.htm

https://youtu.be/TkUYdCPFXXs

15.7 Adding the filter + LNA

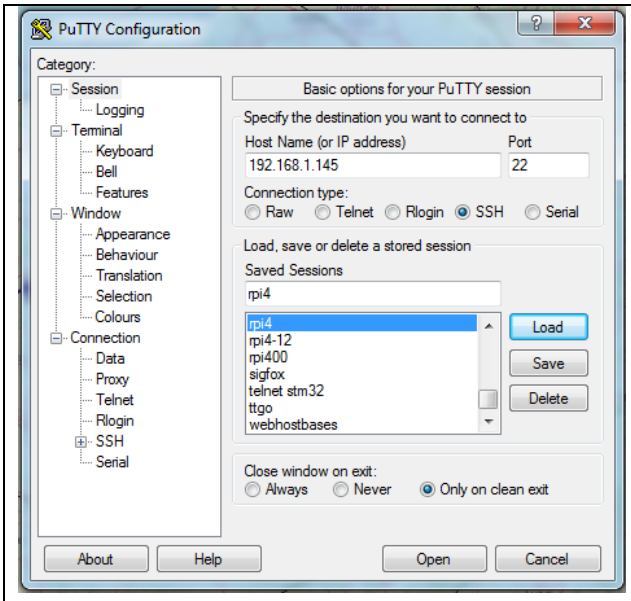
In order to improve the reception quality, it is often necessary to add a band filter and a low noise amplifier (LNA).

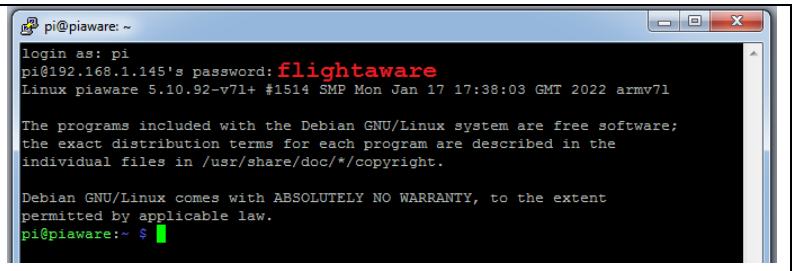
I opted for a simple solution by using a ready-to-use, low-cost circuit.

The circuit must be supplied with 5V from an external source. After some modifications, it is possible to use the Bias-T integrated in the Rtl-sdr **V3**.

Simply remove the capacitor C1 and move the resistor R1 in place of C1.

We now need to activate the bias-T of the Rtl-sdr key using a few command lines with Putty in SSh.





Login : `pi`

Password : `flightaware`

https://www.rtl-sdr.com/getting-the-v3-bias-tee-to-activate-on-piaware-ads-b-images/

Dependencies installation:

```
sudo apt-get update
sudo apt install git
sudo apt install cmake
sudo apt install build-essential
sudo apt install libusb-1.0-0-dev
```

rtl_biast utility installation:

```
git clone https://github.com/rtlsdrblog/rtl_biast
ls
cd rtl_biast/
mkdir build
cd build/
cmake .. -DDETACH_KERNEL_DRIVER=ON
make
```

```
pi@piaware:~ $ cd rtl_biast/
pi@piaware:~/rtl_biast $ mkdir build
pi@piaware:~/rtl_biast $ cd build/
pi@piaware:~/rtl_biast/build $ cmake .. -DDETACH_KERNEL_DRIVER=ON
-- The C compiler identification is GNU 10.2.1
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Check for working C compiler: /usr/bin/cc - skipped
-- Detecting C compile features
-- Detecting C compile features - done
-- Build type not specified: defaulting to release.
-- Extracting version information from git describe...
-- Found PkgConfig: /usr/bin/pkg-config (found version "0.29.2")
-- Checking for module 'libusb-1.0'
--   Found libusb-1.0, version 1.0.24
-- Looking for libusb_handle_events_timeout_completed
-- Looking for libusb_handle_events_timeout_completed - found
-- Looking for libusb_error_name
```

T-bias test:

Stop reception:

```
sudo service dump1090-fa stop
```

Activate Bias-T

```
cd src/
ls
./rtl_biast -b 1
```

Check with a **DC voltmeter** that there is a DC voltage of 5V on the Rtl-sdr SMA plug.

Restart the reception:

```
sudo service dump1090-fa start
```

Check reception on the local RPI page http://piaware/skyaware/

Note:

To disable Bias-T:

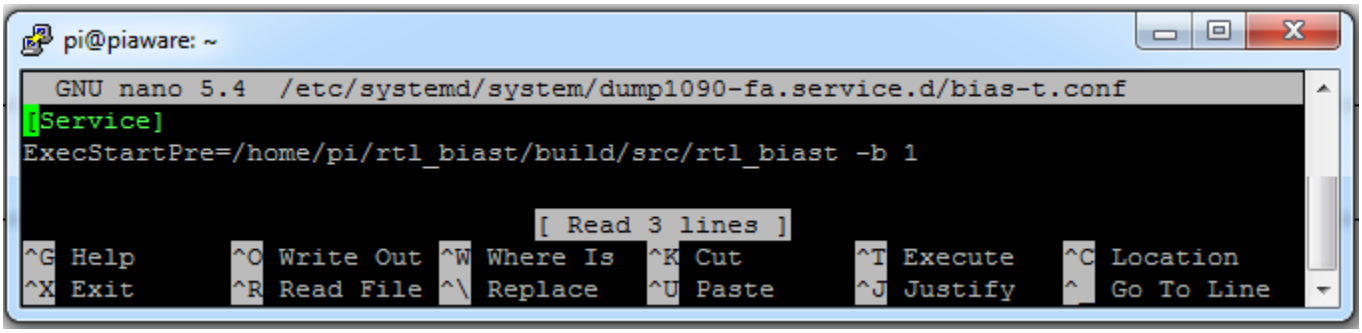
```
./rtl_biast -b 0
```

The Bias-T must now be activated automatically when the Raspberry pi starts up:

```
sudo mkdir /etc/systemd/system/dump1090-fa.service.d
sudo nano /etc/systemd/system/dump1090-fa.service.d/bias-t.conf
```

Paste the following two lines into the nano editor:

```
[Service]
ExecStartPre=/home/pi/rtl_biast/build/src/rtl_biast -b 1
```



Do not forget to save ctrl+o before quitting ctrl+x.

Restart the RPI, then check the reception again.

```
sudo reboot
```

15.8 Conclusion

The solution offered by FlightAware is very practical and will delight ADS-B fans. There will be more tutorials to follow regarding the listening of airliners.