LIBRE SDR

Open-Source SDR for Radio Amateurs

DISCOVERING LIBRESDR

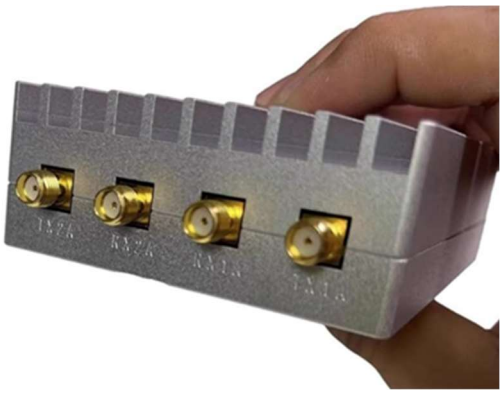
I first heard about the **LibreSDR** during a Sunday morning **QO-100 AMSAT net**, where several radio amateurs were discussing their experiences using this compact yet capable software-defined radio. The conversation quickly turned from casual mention to genuine enthusiasm, suggesting that LibreSDR likely to gain a good reputation among satellite operators and SDR experimenters.

Curiosity piqued, I began researching the platform to better understand what made it so appealing.

WHAT IS LIBRESDR?

The LibreSDR is a **software-defined radio communication experiment platform**, closely aligned with the well-known PlutoSDR design philosophy. It is typically based on:

- **AD9363 / AD9361 RF transceiver**
- **ZYNQ-7020 System-on-Chip**
- Open, experiment-friendly firmware support



This hardware combination provides impressive flexibility for both terrestrial and satellite applications, making it particularly suitable for QO-100 operations and general RF experimentation.

VARIANTS: KNOW WHAT YOU'RE BUYING

One important point for potential buyers is that **not all LibreSDR units are identical**. Several variants exist, and differences may include component choice, RF performance, and firmware compatibility. As a result, selecting a trusted supplier is crucial to avoid disappointment.

SOURCE & COST

Based on recommendations from others, I chose to purchase my unit from **Open Source SDR Lab**, a supplier frequently mentioned as a reliable supplier.

- **Supplier:** Open Source SDR Lab
- **Price:** Approximately **£120.00**

At this price point, LibreSDR sits comfortably in the affordable SDR category while offering serious experimentation potential.

Link to Open Source SDR Lab: https://opensourcesdrlab.com

Below is what I received when mine arrived.

In the box was the following

- LibreSDR
- Four antennas
- One OTG
- Two C type USB leads
- One Network Cable
- SD Card with some software pre-installed.

Product description from Open Source SDR Lab website.

Four antenna inputs (two RX and two TX ports)
An Ethernet port
A debug connection with serial ports for console access
An SD card slot

Inspired by Pluto, the design refers to Pluto, and the firmware is transplanted to support Pluto.

1) Using XC7Z020-2CLG400I chip 7z020-2I

2) The AD9363 chip is configured as the AD9361 chip, the frequency range is 70MHz~6GHz, and the maximum sampling rate is 61.44MHz.

3) PS 1GB DDR3 32bit bit width, frequency 1066M

4) 32MB QSPI Flash, support TF card

5) Onboard JTAG circuit and USB to serial port, one line for power supply and debugging

6) Support USB OTG, can be master or slave, all USB interfaces use TYPE-C interface, OTG mode is automatically recognized, and the USB port is turned on for power supply

7) Adopt VCXO, support PPS or 10MHz tame

8) Gigabit Ethernet interfaces

9) Adopt 10-layer PCB wiring, impedance matching, signal integrity design

10) The product adopts CNC shell to strengthen heat dissipation and ensure strength

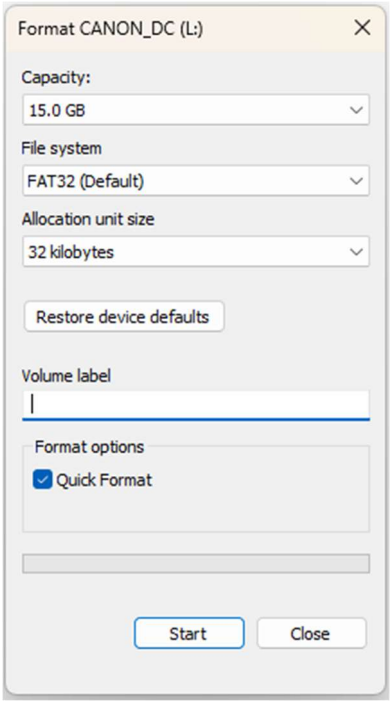
Now you have you LibreSDR how do you use it?

LibreSDRs appeal lies in its versatility. How you use it will depend entirely on which part of the hobby excites you the most.
From now on I am going to show you how I am using my LibreSDR for use on the narrow and wide band transponders on QO-100 satellite for transmission and reception of voice and data along with DATV transmission and reception.

To begin using the LIBRESDR, you need to prepare an SD card with the necessary software. Follow these steps:

Format the SD Card: Connect your SD card to your computer and format it using a FAT32 file system. Ensure that you do not have any files on this SD Card as they will all be deleted during the format process.

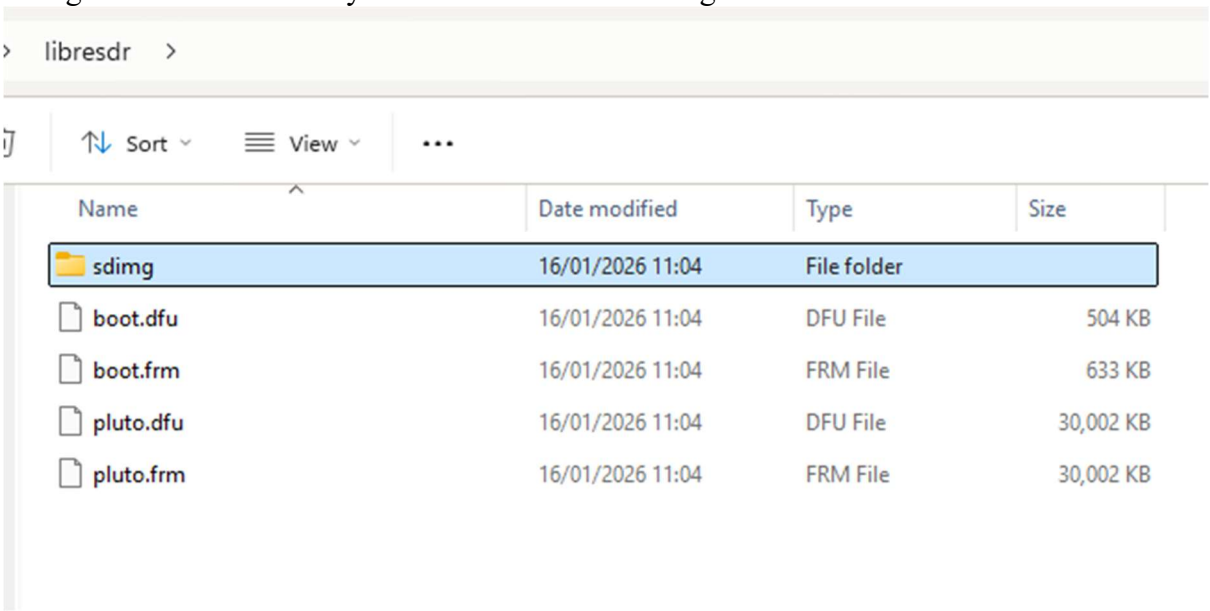
1) Maximum size of SD card supported by the LibreSDR is 32GB any bigger won't work.
2) Format the card in your PC using a Fat 32 file system.



I use the Tezuka firmware from Evariste F5OEO.

Download the Required Files: Visit the GitHub repository at https://github.com/F5OEO/tezuka_fw/releases to download the firmware file. **Please** select the correct version of the firmware for your Libre SDR, the firmware that you require is called **libresdr.zip** download this file to you PC.

When the file has downloaded extract the file to a folder.
Navigate to the folder and you should see the following.



Copy Files to the SD Card: Copy the boot.dfu, boot.frm, pluto.dfu and pluto.frm to you prepared SD Card.

You can also place the firmware directly into the memory of the LibreSDR the same as a PlutoSDR but this is not recommended.

This will get your LibreSDR working the files in the sdimg folder you can play with later. In this folder you will have the files to overclock your LibreSDR this will increase the bandwidth up to around 26 MHz

Connecting the LIBRESDR:

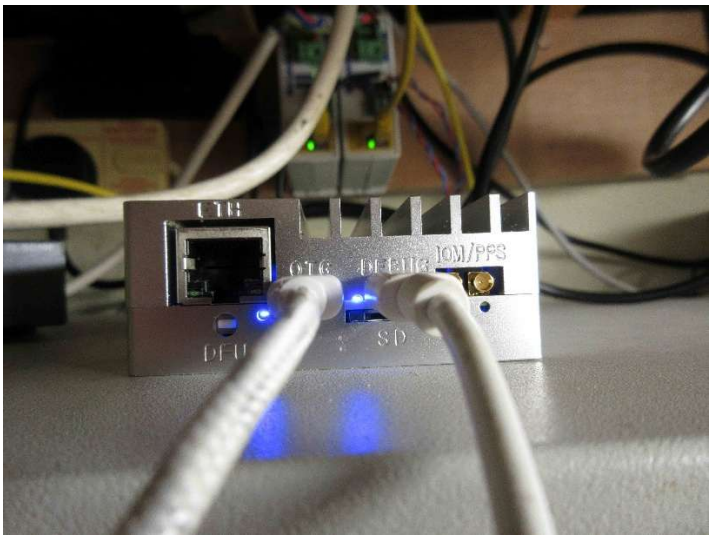
Insert the SD card into the LibreSDR.

Power the device using the OTG port.

You will need a good quality 5 Volt 3Amp power supply and I use the Raspberry Pi 15W USB-C Power Supply

Connect the LibreSDR to your computer via USB using the USB C type socket next to the RJ45 port.

Wait for the LibreSDR to power up, be patient you can't rush this when the LibreSDR has booted you will have these two blue led illuminated.



Accessing the Device:

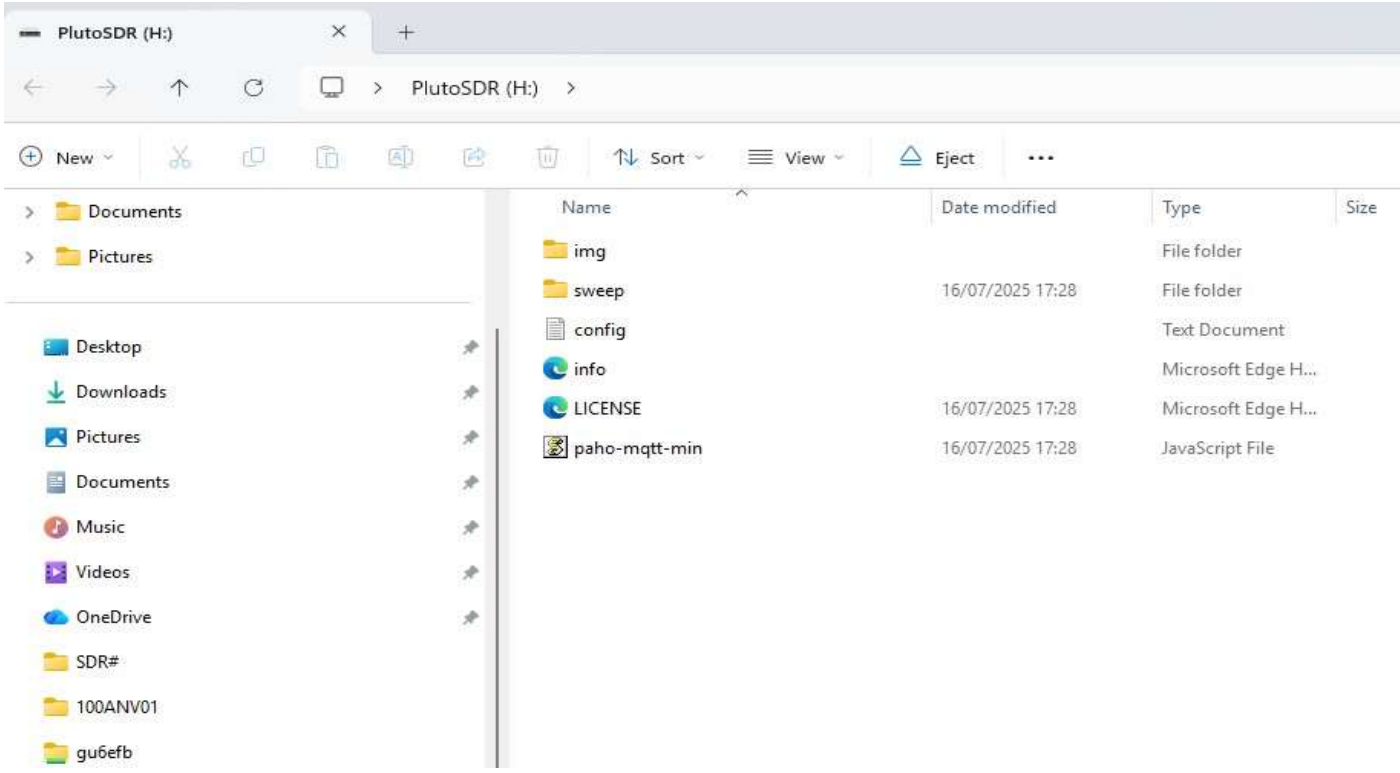
I found that I needed to install the ADALM-PLUTO driver to be able to connect to the LibreSDR.

Driver link: https://github.com/analogdevicesinc/plutosdr-m2k-drivers-win/releases

Console Access: Connect to the debug port using a serial connection.

You may need to try different COM ports to find the correct one

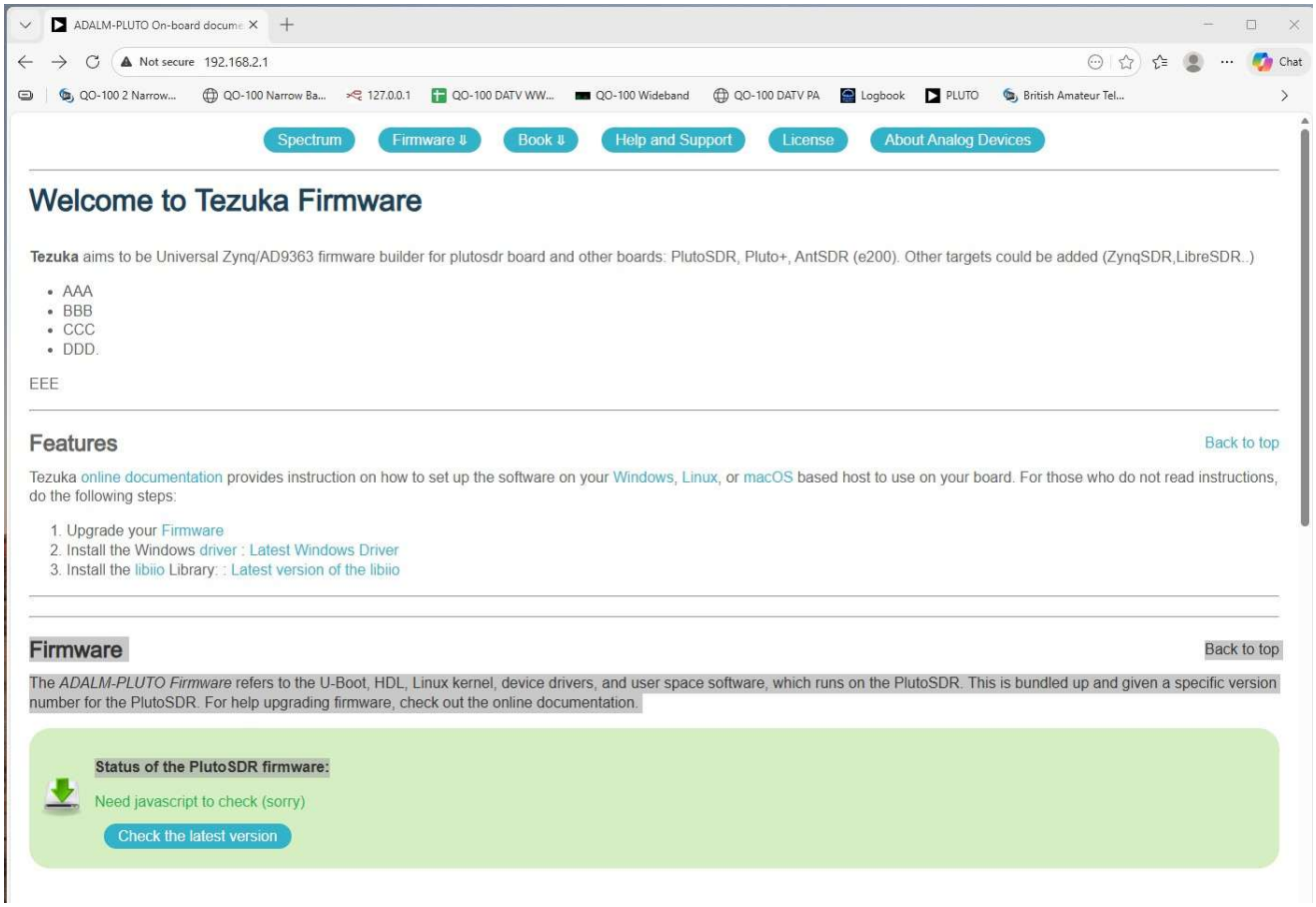
You should then be greeted with an information page displayed on your PC.



To access the Libre SDR, you can use the USB comport, Web page or SSH a console connection:

Web Browser or SSH Access: Use the default IP address 192.168.2.1 to connect.

The default username is **root** and the password is **analog.**



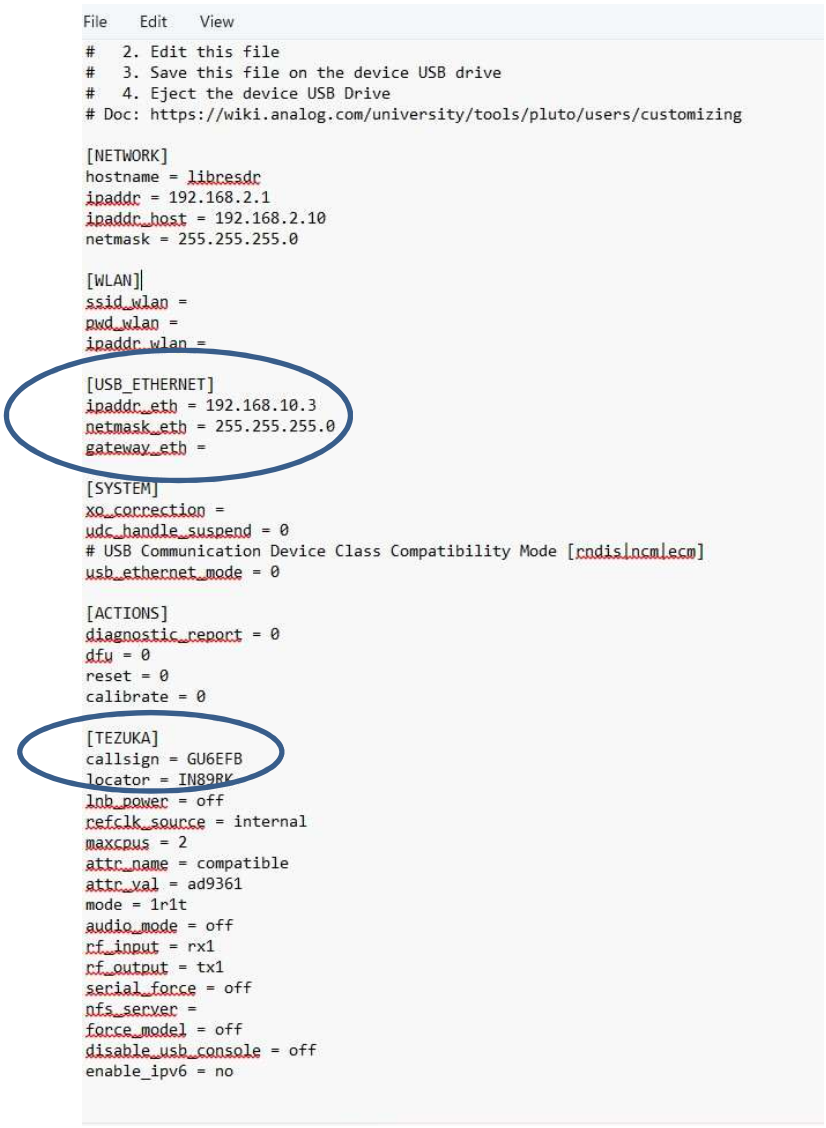
Troubleshooting Common Issues

If you encounter issues during setup, consider the following troubleshooting tips:

Ensure the SD card is formatted correctly and contains all necessary files.

If the device does not boot, verify that the firmware files are correctly named and placed in the right directory.

Open the LiberSDR Config File: and you see something like this. Please bear in mind that you will need to enter the following setting to suit your requirements.



```
File    Edit    View
#   2. Edit this file
#   3. Save this file on the device USB drive
#   4. Eject the device USB Drive
# Doc: https://wiki.analog.com/university/tools/pluto/users/customizing

[NETWORK]
hostname = libresdr
ipaddr = 192.168.2.1
ipaddr_host = 192.168.2.10
netmask = 255.255.255.0

[WLAN]
ssid_wlan =
pwd_wlan =
ipaddr_wlan =

[USB_ETHERNET]
ipaddr_eth = 192.168.10.3
netmask_eth = 255.255.255.0
gateway_eth =

[SYSTEM]
xo_correction =
udc_handle_suspend = 0
# USB Communication Device Class Compatibility Mode [rndis|ncm|ecm]
usb_ethernet_mode = 0

[ACTIONS]
diagnostic_report = 0
dfu = 0
reset = 0
calibrate = 0

[TEZUKA]
callsign = GU6EFB
locator = IN89RK
lnb_power = off
refclk_source = internal
maxcpus = 2
attr_name = compatible
attr_val = ad9361
mode = 1r1t
audio_mode = off
rf_input = rx1
rf_output = tx1
serial_force = off
nfs_server =
force_model = off
disable_usb_console = off
enable_ipv6 = no
```

Enter your Callsign and QRA locator in the TEZUKA section.

Do not change the [NERWORK] IP address 192.168.2.1

If you want to access the LibreSDR over a IP network enter your network setting in the section labelled [USB_ETHERNET] Mine is 192.168.10.3

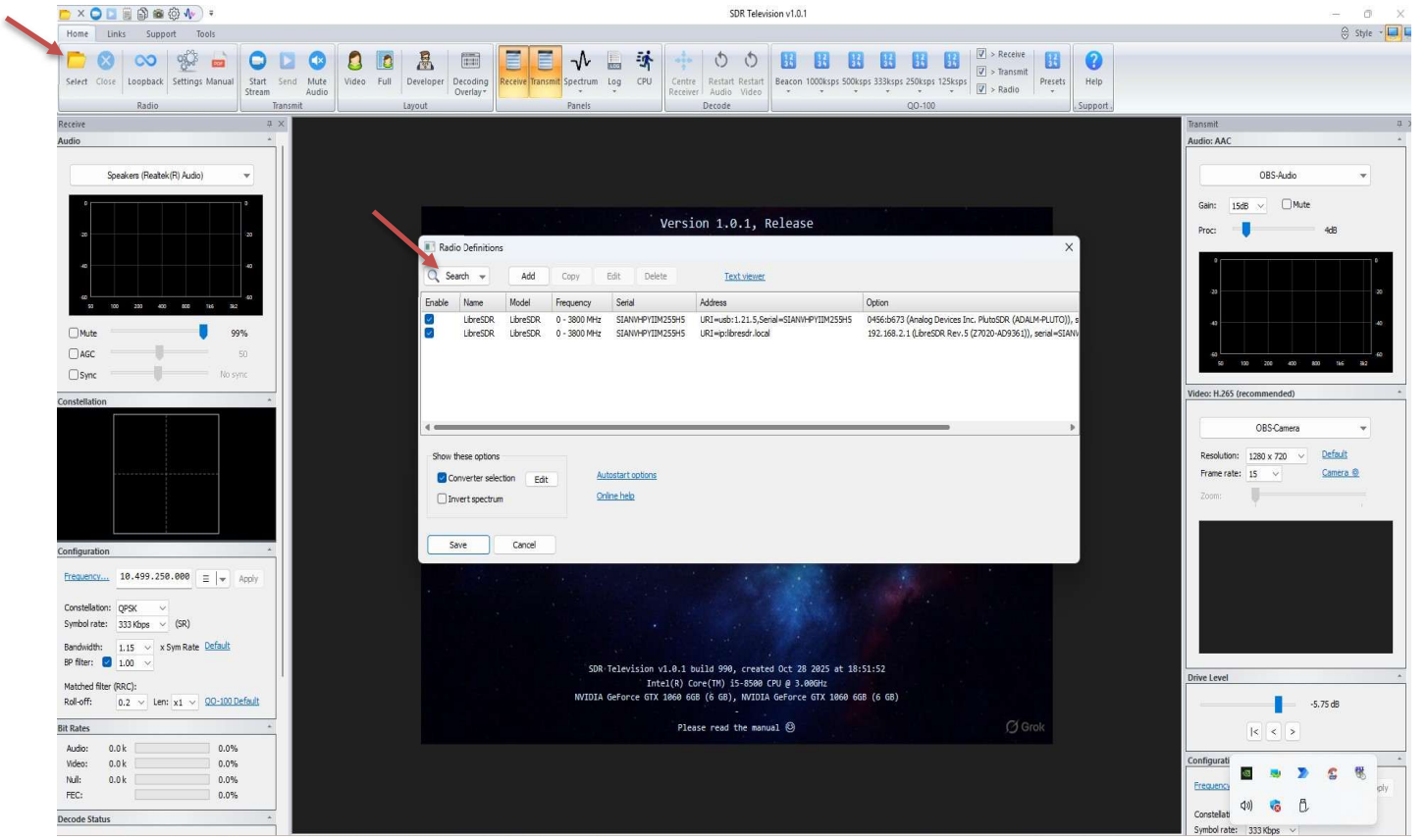
Leave all other settings at default.

Save the settings and eject the LiberSDR by using the Windows function to safely eject the USB connected LibreSDR this will write the config file into the LibreSDR's memory.

Wait for the LibreSDR to reboot. At this point you should be able to use your LibreSDR with many of software programs that support the Pluto. You will need to experiment as not all may work.

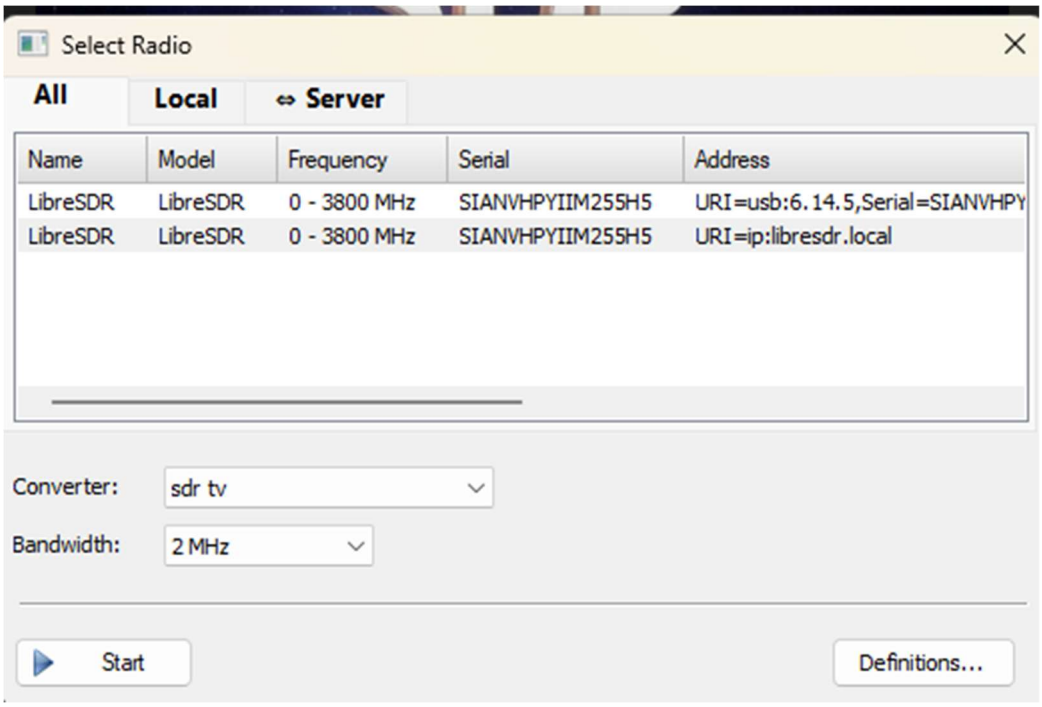
Setting SDR Television to allow use with the LibreSDR:

Download and install SDR Television from Simon G4ELI at https://www.sdr-radio.com/sdr-television-v1-0-1-bugfix this is the current release.



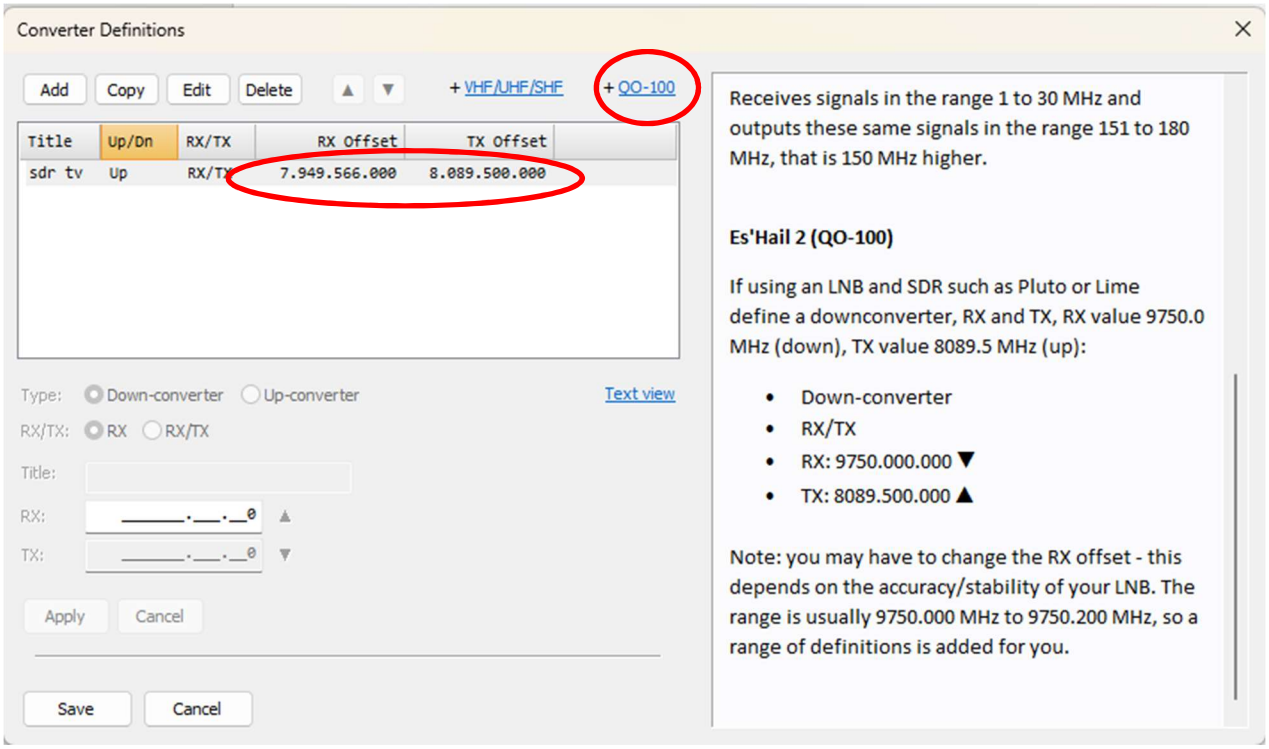
To enable the LibreSDR to be found by SDRTelevision follow the steps below

- 1 Click on select in the ribbon bar
- 2 Click on Definitions
- 3 Use the search option and from the dropdown menu and select PlutoSDR
  This should find your LiberSDR and populate the box with two entries as shown below.



Select the Bandwidth for 2 MHZ or 2.5 MHz

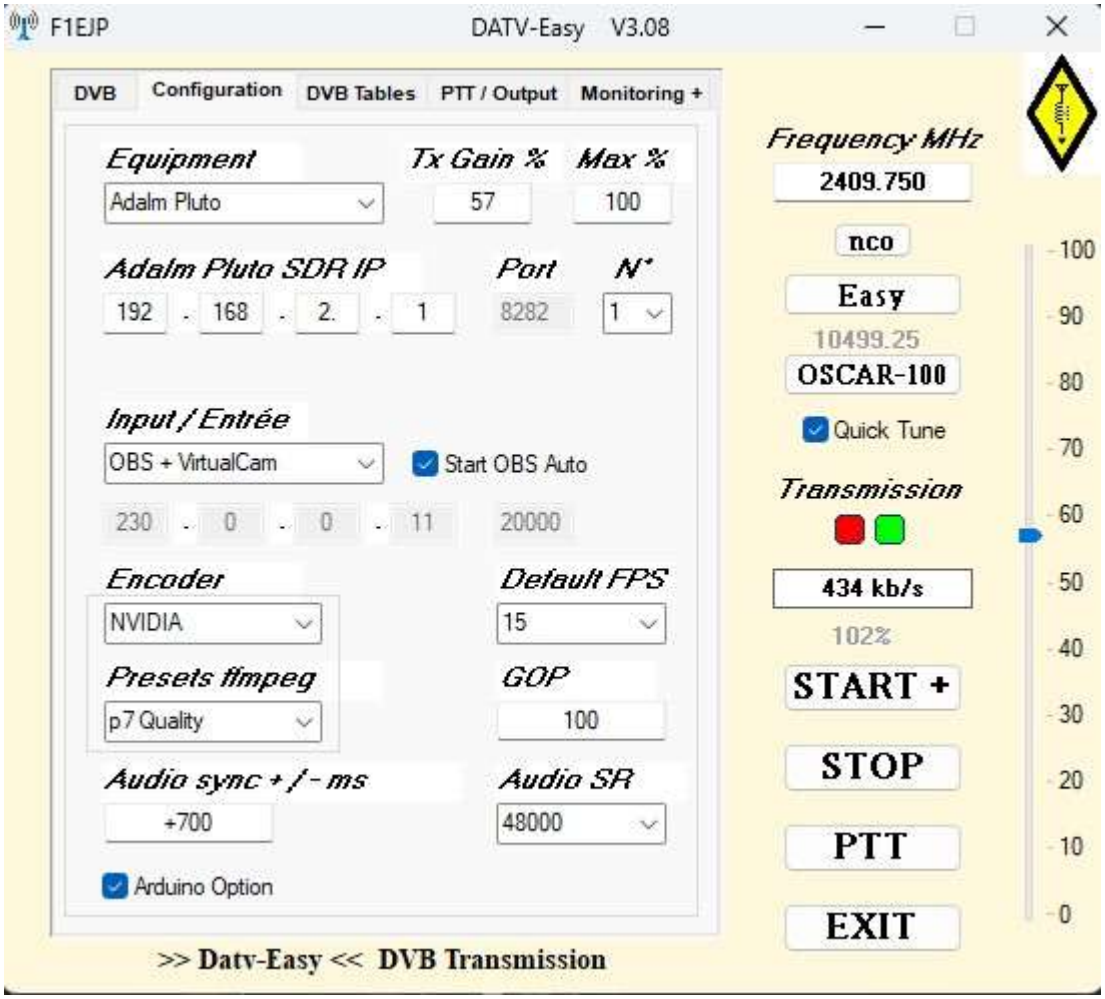
In converter select option enter your LNB offset as the LibreSDR can also receive the DATV transmissions from the satellite, without the need for any further equipment. Use the default TX and RX to get you started, click on +QO-100 button to set your RX and TX offset. Clicking on "+QO-100" button in the converter definitions gets you a good start (in fact it is quite similar to SDRConsole)



Please note that the transmission from the satellite uses horizontal polarization so you will need to provide the LNB with 18 Volts via a Bias-T

It is a very similar procedure to setup a LiberSDR for use with SDRConsole just follow the same steps used to install your LibreSDR in SDRTelevision but using SDR Console. Remember that you will need to supply the LNB with 12 volts for use with the narrowband QO-100 Transponder.

The LiberSDR also works well when using DTV-Easy using the following settings



FINAL THOUGHTS

The above is what I have been using with my LiberSDR, they work for me.

Use the above information at your own risk.

LibreSDR should work with most software that works with an Adlam Pluto but you need to experiment.

LibreSDR support and firmware is in a state of continual development and this is very much an experimental use of the LibreSDR. It's a reminder that powerful radio experimentation doesn't have to come with a premium price tag.

Good luck and have some fun.

Updates: 17/01/2026

I have now been made aware of some updated firmware for the LibreSDR at

https://gitlab.com/libre-sw/40mhz-sync

One of the benefits of this firmware is the Synchronization of the 40MHz clock with an external 10MHz reference now works. I have yet to try this firmware so please read all of the instructions before proceeding.