Debian 12 Bookworm

Recipe and build log for configurations

[bluetooth](#_Section:_Bluetooth)---------------------------------------------------- ------------------------------🡪 [To start](#_Section:_Initial_settings)

[librespot](#_Section:_Install_librespot)---------------------------------------------- ------------------------------------🡪 [To image](#_Create_an_image)

[shairport-sync](#_Section:_install_Shairport-sync)----------------------------------- ------------------------------------🡪 [To apt](#_All_about_apt)

------------------------------------------------------------------- ------------------------------------🡪 [To test](#_Section:_basic_testing)

Table of Contents

[Section: Initial settings and configurations 7](#_Toc211527084)

[Format the SSD 7](#_Toc211527085)

[Use the Raspberry Imager 7](#_Toc211527086)

[Set the “enhetsbeteckning” again 8](#_Toc211527087)

[Edit the file config.txt in the boot sector 8](#_Toc211527088)

[First boot of new OS on new SSD 8](#_Toc211527089)

[No boot? 8](#_Toc211527090)

[Useful disk commands for SSD 8](#_Toc211527091)

[Install PuTTY and WinSCP on Windows desktop 8](#_Toc211527092)

[Become root 9](#_Toc211527093)

[Enable root login for ssh for WinSCP 9](#_Toc211527094)

[Size of file system 9](#_Toc211527095)

[Extend file system 9](#_Toc211527096)

[Make SUDO work 10](#_Toc211527097)

[Check host name 10](#_Toc211527098)

[Fix the root prompter so it becomes colourful 10](#_Toc211527099)

[First update of the Raspberry Pi firmware 10](#_Toc211527100)

[Update the Raspberry Pi operating system 10](#_Toc211527101)

[Edit config.txt in /boot/firmware 11](#_Toc211527102)

[Alsa checking after editing config.txt 12](#_Toc211527103)

[Look at cmdline.txt in /boot/firmware 15](#_Toc211527104)

[Section: Images 16](#_Toc211527105)

[Create an image with Win32 Image Writer DEPRECATED 16](#_Toc211527106)

[Create an image with image-utils 16](#_Toc211527107)

[Flash an image to SSD 18](#_Toc211527108)

[SSD is in read-only mode 18](#_Toc211527109)

[First boot of an image 19](#_Toc211527110)

[Expand the filesystem 19](#_Toc211527111)

[wlan1 is gone… 20](#_Toc211527112)

[Change host name 20](#_Toc211527113)

[Update the Raspberry Pi firmware and bootloader 20](#_Toc211527114)

[shutdown now -h 21](#_Toc211527115)

[Images 22](#_Toc211527116)

[Status development SSDs 23](#_Toc211527117)

[All about apt 24](#_Toc211527118)

[Section: More basic configurations 25](#_Toc211527119)

[Expand file system on USB SATA SSD 25](#_Toc211527120)

[Some OS trim details 25](#_Toc211527121)

[Check version 25](#_Toc211527122)

[Get git capabilities 26](#_Toc211527123)

[All about systemd 27](#_Toc211527124)

[Section: Bluetooth 29](#_Toc211527125)

[Bluetooth system parts’ versions 29](#_Toc211527126)

[Bluez – the Bluetooth stack 29](#_Toc211527127)

[bluetoothd daemon definitions for systemd 30](#_Toc211527128)

[Configure Bluetooth – main.conf and input.conf 30](#_Toc211527129)

[Start bluetooth.service with Systemd 30](#_Toc211527130)

[Disable LE - only BR/EDR is wanted… 30](#_Toc211527131)

[BlueALSA - Bluetooth Audio ALSA Backend 32](#_Toc211527132)

[Compile BlueALSA 32](#_Toc211527133)

[Installed BlueALSA codecs 35](#_Toc211527134)

[BlueALSA service - daemon bluealsad for systemd 35](#_Toc211527135)

[bluealsad daemon definition for systemd 36](#_Toc211527136)

[BlueALSA player service - bluealsa-aplay for systemd 36](#_Toc211527137)

[bluealsa-aplay daemon definition for systemd 36](#_Toc211527138)

[Bluez bt-agent service –for systemd 37](#_Toc211527139)

[bluez-agent daemon definition for systemd 37](#_Toc211527140)

[bt-agent work around 38](#_Toc211527141)

[Start the services with systemd 38](#_Toc211527142)

[Journald commands for bluetooth 39](#_Toc211527143)

[Group membership for stream and root 41](#_Toc211527144)

[Some bluetoothctl as commands 41](#_Toc211527145)

[Correct start and stop systemd sequence 41](#_Toc211527146)

[More on systemd on bluetooth and blue\* 42](#_Toc211527147)

[HCI trace and btmon and hcitop and some debugging commands 42](#_Toc211527148)

[More information about Bluetooth signal 43](#_Toc211527149)

[All about bt-commands 44](#_Toc211527150)

[Section: Install raspotify 45](#_Toc211527151)

[Install raspotify 45](#_Toc211527152)

[Librespot version 45](#_Toc211527153)

[Configuration 45](#_Toc211527154)

[raspotify daemon definition for systemd 45](#_Toc211527155)

[Stop Raspotify 46](#_Toc211527156)

[Completely uninstall Raspotify 46](#_Toc211527157)

[Section: Install librespot 47](#_Toc211527158)

[Librespot version 47](#_Toc211527159)

[Sources and tools for librespot 47](#_Toc211527160)

[Install Cargo and Rust 47](#_Toc211527161)

[1. GITHUB FORK: Compile librespot from github with cargo 47](#_Toc211527162)

[Fork librespot from Github and check the ssh key 48](#_Toc211527163)

[Clone the fork to .cargo directory and get the Cargo.toml file right 48](#_Toc211527164)

[Do cargo build 49](#_Toc211527165)

[The librespot bin file can be found here 50](#_Toc211527166)

[Remove librespot files 51](#_Toc211527167)

[Librespot basic test 51](#_Toc211527168)

[2. CRATE: Compile librespot from Crate with cargo 53](#_Toc211527169)

[Where did the essential CRATE files go? 53](#_Toc211527170)

[Create the systemd librespot service file 55](#_Toc211527171)

[Detect librespot events 56](#_Toc211527172)

[Some journalctl for librespot 57](#_Toc211527173)

[Disable Ivp 6? 57](#_Toc211527174)

[librespot server error 2025-08-06 57](#_Toc211527175)

[Section: install Shairport-sync 60](#_Toc211527176)

[Shairpoint-syncVersion 60](#_Toc211527177)

[Do not install the shairport-sync package 60](#_Toc211527178)

[Airplay 2 60](#_Toc211527179)

[Prepare for installation with Airplay 2 60](#_Toc211527180)

[Compile and install NQPTP – Not Quite PTP 60](#_Toc211527181)

[Compile and install shairport-sync 61](#_Toc211527182)

[The shairport-sync bin file can be found here 61](#_Toc211527183)

[Configuration file for Shairport-sync 62](#_Toc211527184)

[Detect Shairport-sync events 63](#_Toc211527185)

[Start shairport-sync.service in systemd 63](#_Toc211527186)

[Use Cloudplayer for testing 63](#_Toc211527187)

[Some journalctl for shairport-sync 64](#_Toc211527188)

[Section: Wi-Fi and hotspot 65](#_Toc211527189)

[Network Manager 65](#_Toc211527190)

[Configurations at directory /etc/NetworkManager/system-connections 65](#_Toc211527191)

[The Wi-Fi needs to be activated 65](#_Toc211527192)

[Connect to Wi-Fi 66](#_Toc211527193)

[Connect/disconnect sequences Wi-Fi network 67](#_Toc211527194)

[Force the 2.4 GHz band 68](#_Toc211527195)

[Change country code for wireless conf files 68](#_Toc211527196)

[p2p-dev-wlan0? 68](#_Toc211527197)

[ip link commands on wlan interface 68](#_Toc211527198)

[iw commands on wlan interfaces 68](#_Toc211527199)

[Journald for nm 68](#_Toc211527200)

[Create a hotspot with nmcli 69](#_Toc211527201)

[Enable and Disable hotspot with nmcli 69](#_Toc211527202)

[Hotspot connection file IP-address: 10.0.0.1 69](#_Toc211527203)

[Make the hotspot persistent 70](#_Toc211527204)

[Install the USB Wi-Fi antenna 71](#_Toc211527205)

[Attach the USB Wi-Fi antenna 71](#_Toc211527206)

[More info about the Wi-Fi signal 71](#_Toc211527207)

[Section: memory management 73](#_Toc211527208)

[No more RAM swapping to disk 73](#_Toc211527209)

[Changing commit times – No? 73](#_Toc211527210)

[Temporary filesystem - yes 74](#_Toc211527211)

[Set up tmpfs - for logs and temporary files 74](#_Toc211527212)

[Read-only root file systems - No 75](#_Toc211527213)

[Overlay file system - No 76](#_Toc211527214)

[Section: basic testing 77](#_Toc211527215)

[IQaudio sound card and alsa 77](#_Toc211527216)

[Bluetooth 77](#_Toc211527217)

[Librespot 78](#_Toc211527218)

[Shairport-sync 78](#_Toc211527219)

[Mount an USB stick 78](#_Toc211527220)

[Wi-Fi and AP 79](#_Toc211527221)

[Memory 79](#_Toc211527222)

[Web server 79](#_Toc211527223)

[Boot phase 80](#_Toc211527224)

[Disk volume check on OS SSD 80](#_Toc211527225)

[Throttled? 80](#_Toc211527226)

[Section: set up backend 81](#_Toc211527227)

[Create the directory structure 81](#_Toc211527228)

[Look at /player directories from 2022 82](#_Toc211527229)

[Install nodejs and npm 83](#_Toc211527230)

[Get the right version of nodejs 83](#_Toc211527231)

[Create package.json for npm in /streamer 84](#_Toc211527232)

[Install Express to node.js 84](#_Toc211527233)

[Install socket.io to node.js 84](#_Toc211527234)

[Install Notyf 84](#_Toc211527235)

[Popper 85](#_Toc211527236)

[Bootstrap 85](#_Toc211527237)

[jQuery 85](#_Toc211527238)

[Install node-pty (and strip-ansi) 85](#_Toc211527239)

[Optional npm packages to consider 85](#_Toc211527240)

[npm packages versions 85](#_Toc211527241)

[npm 87](#_Toc211527242)

[Section: Create a systemd service for player 88](#_Toc211527243)

[Section: pishrink minimized image DEPRECATED 89](#_Toc211527244)

[Make an image [optional] 89](#_Toc211527245)

[Shrink the image - pishrink script 89](#_Toc211527246)

[Section: all about log files 90](#_Toc211527247)

[Reduce the size of logs 90](#_Toc211527248)

[Journald log files 90](#_Toc211527249)

[Use journald and configure journals 91](#_Toc211527250)

[Turn off journald with the config file [optional] 91](#_Toc211527251)

[Turn on logs again - if needed [optional] 91](#_Toc211527252)

[Check logs and journal with journalctl 92](#_Toc211527253)

[Check out daemon.log - when Player is running as a service 92](#_Toc211527254)

[Clean out logs before customer image 92](#_Toc211527255)

[Section: systemctl and systemd.service 93](#_Toc211527256)

[SSD memory read & write 94](#_Toc211527257)

[Section: Add ons, additions and trimming 95](#_Toc211527258)

[Investigate another image 95](#_Toc211527259)

[Which version?... 96](#_Toc211527260)

[Section: SSD memory read & write 98](#_Toc211527261)

[Section: image logs 99](#_Toc211527262)

[Section: Docker 101](#_Toc211527263)

# Section: Initial settings and configurations

## Format the SSD

Connect the SSD to the PC. Open up the command line by enter “kommandotolken” (Swedish version of Windows) in the search box.

Enter: diskmgmt.msc

Like this:

Microsoft Windows [Version 10.0.22631.5472]

(c) Microsoft Corporation. Med ensamrätt.

C:\Users\kalle>diskmgmt.msc

### Initiate the disk

The Disk Management tool opens in a new window. Scroll down to find the disk. Right click on the leftmost box. Choose “Initiera disk”, välj **MBR**. If the tool refuses to initiate the disk put it in **offline** and then **online** again.

1. MBR (Master Boot Record), is used by 32-bit PCs, older PCs, and removable drives such as memory cards.
2. GPT (GUID Partiopn Table), use for storage larger than 2 TB and not used for boot.

### Format and create a new volume

Right click on the second box that represents the size of the disk. Choose “Ny enkel volym” and continue with instructions that comes up. Choose file type **exFAT**. Now the disk should be visible in “Utforskaren” with a unique “enhetsbeteckning”. If not put in **offline** and **online** again, or detach the ssd and attach it again.

## Use the Raspberry Imager

Flash the Raspberry OS lite 32-bit on to the SSD. The “Enhetsbeteckning” gets lost.

### OS customization in Raspberry imager

Name: Streamer

Username: stream

Password: fajKvam8\*

Also enable Wi-Fi and SSH.

**---- after the image is flashed/burned**

Disconnect the SSD and connect it again – the “Enhetsbeteckning” should come back. Check with the Disk Management tool that all three partitions are in place and that the ssd has a “Enhetsbeteckning”.

## Set the “enhetsbeteckning” again

In the Disk Manager right click on the boot partition and choose “Ändra enhetsbeteckning och sökvägar…”. Add a “enhetsbeteckning” for the SSD and its boot partition will be accessible in “utforskaren”. If it does not work put it in offline and online again, or detach the ssd and attach it again.

It is important that the ssd has a “Enhetsbetckning” since the boot sector is a FAT32 file type and must work fully in Windows (be accessible in “utforskaren”) before the ssd is attached to the Raspberry.

## Edit the file config.txt in the boot sector

Find the file config.txt in the boot partion - bootfs. Open it with “Anteckningar”. Add the lines below to secure continues power to the SSD via USB.

# Enable USB 5V 5A

usb\_max\_current\_enable=1

## First boot of new OS on new SSD

The first boot takes a longer time. For example, the filesystem is expanded, which requires an extra boot. Led light will be both green and red during this. Network led lights will come and go. Wait!

sudo df -Th -- should show an extended filesystem

sudo sfdisk -F -- should show no unpartioned space

## No boot?

Detach the ssd and attach it to the Windows laptop to see what is going on. If that doe not work, boot the Raspberry Pi with a working disk, wait and then attach the ssd to a USB 2.0 port. Find out if the ssd is dead or not.

## Useful disk commands for SSD

sudo df -Th -- mounted partitions

sudo fdisk -l -- mounted disks

sudo blkid -- filesystems on disks

sudo lsusb -- overview of USB ports

sudo lsblk -- detailed disk structure

sudo partprobe –summary -- partions

sudo fsck -A --check and repair all or /dev/sdb1

d

**--- after first boot of the Raspberry Pi**

## Install PuTTY and WinSCP on Windows desktop

Get the ip address and then open PuTTY. Access the Raspberry Pi through ssh.

### No SSH service

Create an empty file named ssh in the boot partition and not in the overlay folder. Connect the SSD to the Raspberry Pi 5 and boot. If not set in the customization part of the imager.

## Become root

Using PuTTY, login as user: **stream** and enter password: **fajskvam8\***.

sudo passwd -- set password to: **fajskvam88\*\*!!** for **root**.

sudo passwd root -- for Trixie

After that do:

su -- in order to become root

Prompter: **root@raspberrypi:/#**, or check with env or whoami that user now is root.

## Enable root login for ssh for WinSCP

Edit file: nano /etc/ssh/sshd\_config and set the following config parameter:

(under Authentication:)

PermitRootLogin prohibit-password has to be changed

PermitRootLogin yes -- uncomment, enter ‘yes’ + save with ctrl-x and Y.

and then:

sudo systemctl restart ssh

or do a:

reboot

## Size of file system

sudo du -Th –total

Will yield this:

Filesystem Size Used Avail Use% Mounted on

udev 953M 0 953M 0% /dev

tmpfs 199M 4.4M 195M 3% /run

/dev/sda2 220G 2.0G 207G 1% / - **Note**. Partion in windows

tmpfs 993M 0 993M 0% /dev/shm

tmpfs 5.0M 16K 5.0M 1% /run/lock

/dev/sda1 510M 99M 412M 20% /boot/firmware - **Note**: bootfs in Windows

tmpfs 199M 0 199M 0% /run/user/1000

total 223G **2.1G** 210G 1% -

99 MB + 2 GB seems to be the Raspberry Pi OS = **2,1 GB** in total.

du -hs to see disk usage at present directory

df -Th --total display disk system space

fdisk -l shows disk partitions

free -hw shows RAM usage

## Extend file system

Can be done through:

sudo raspi-config

Or from the command line run:

sudo raspi-config --expand-rootfs  +

reboot

## Make SUDO work

sudo visudo

Below # User privilege specification add the line:

stream ALL=(ALL) NOPASSWD: ALL

See file /etc/sudoers.tmp - the file can also be found [here](file:///D:\10%20Player%20Code\Player%20Configs\sudoers).

## Check host name

If this was not done in the imager tool, edit the host name here

sudo nano /etc/hostname

… and the file

sudo nano /etc/hosts

## Fix the root prompter so it becomes colourful

Add this line last in /root/.bashrc – or copy from /home/rad/.bashrc

export PS1="\[$(tput bold)\]\[\033[38;5;6m\]\u\[$(tput sgr0)\]\[\033[38;5;45m\]\w\[$(tput sgr0)\]\[\033[38;5;14m\]\\$\[$(tput sgr0)\]"

Close PuTTy and open again.

### Set time zone

Do that in raspi-config…

Or…

sudo date --set '2025-07-12 15:07:30' -- no does not work

## First update of the Raspberry Pi firmware

sudo rpi-update

sudo reboot

### Check bootloader

To check if the bootloader needs an update, you can execute:

sudo rpi-eeprom-update

If the output shows that you need to update your bootloader, you can do so by executing:

sudo rpi-eeprom-update -a

## Update the Raspberry Pi operating system

To completely update your Raspberry Pi operating system, you can execute:

sudo apt update

sudo apt full-upgrade

sudo reboot

**[from now on - login as root] -----------------------------------------------------------------------------------------**

## Edit config.txt in /boot/firmware

### Added and disabled general configs

max\_usb\_current=1 -- added before first boot

#dtparam=audio=on -- disabled

dtoverlay=vc4-kms-v3d,noaudio -- noaudio added

### Added IQaudIO DigiAMP+ dependent configs

The Raspberry Pi (IQaudIO ) audio board must be the primary audio device by disabling the Raspberry Pi’s on-board audio devices. Note: there is no need for a .asoundrc in root’s home folder. Check with ls -a to confirm that the file does not exist.

#### Get the right configurations

Use the following command to check the version of the board:

grep -a . /proc/device-tree/hat/\*

If the last line is /proc/device-tree/hat/vendor:IQaudIO Limited www.iqaudio.com then add in config.txt:

#Specify the model

dtoverlay=rpi-digiampplus -- should be iqaudio-digiampplus????

If the vendor string says "Raspberry Pi Ltd." then no further action is needed.

#### This twoconfigurations are added to config txt

# A). Prevents the normal HAT overlay from being loaded

dtoverlay=

# B). IQaudIO DigiAMP+ defines the sound card

dtoverlay=iqaudio-digiampplus,unmute\_amp

For Raspberry Pi boards: replace B). with dtoverlay=rpi-digiampplus,unmute\_amp.

#### Legacy

dtoverlay=rpi-digiampplus or dtoverlay=iqaudio-digiampplus??

**Note**: sudo shutdown now is required - this is important

### More about IQaudio configs from overlays/**README** file

Name: iqaudio-dacplus

Info: Configures the IQaudio DAC+ audio card

Load: dtoverlay=iqaudio-dacplus,<param>=<val>

Params:

**24db\_digital\_gain** Allow gain to be applied via the PCM512x codec Digital volume control. Enable with "dtoverlay=iqaudio-dacplus,24db\_digital\_gain" (The default behaviour is that the Digital volume control is limited to a maximum of 0dB. ie. it can attenuate but not provide gain. For most users, this will be desired as it will prevent clipping. By appending the 24db\_digital\_gain parameter, the Digital volume control will allow up to 24dB of gain. If this parameter is enabled, it is the responsibility of the user to ensure that the Digital volume control is set to a value (that does not result in clipping/distortion!)

**auto\_mute\_amp** If specified, unmute/mute the IQaudIO amp when starting/stopping audio playback.

**unmute\_amp** If specified, unmute the IQaudIO amp once when the DAC driver module loads.

## Alsa checking after editing config.txt

Enter:

sudo alsamixer -- choose IQaudIO card. Press **<F6>**.

Adjust analogue bars from 0% to 100% by using the arrow key. Both mixers ANALOGUE and ANALOGUE P1 should be set to 100%. Set DIGITAL to max as well. **<Esc>** for exit.

Then do:

sudo alsactl store -- in order to save settings

### Check if there is sound

sudo amixer -Mc 0 set Digital 25% -- -c is card, might set volume

sudo speaker-test -c 2 -t wav - listen to sound check, stop with **ctlr-<c>.**

-c is number of channels, -t is test NOTE: **HIGH VOLUME**

There is no sound – there is an error: (because an .asoundrc file was added)

*Playback device is default*

*Stream parameters are 48000Hz, S16\_LE, 2 channels*

*WAV file(s)*

*ALSA lib confmisc.c:165:(snd\_config\_get\_card) Cannot get card index for Zero*

*Playback open error: -19,No such device*

sudo aplay -l -- should render only one card (otherwise see next page)

\*\*\*\* List of PLAYBACK Hardware Devices \*\*\*\*

*--- other cards first bcm2835 vc4hdmi0 and vc4hdmi1 …*

card 3: IQaudIODAC [IQaudIODAC], device 0: IQaudIO DAC HiFi pcm512x-hifi-0 []

Subdevices: 1/1

Subdevice #0: subdevice #0

Check also:

sudo cat /proc/asound/modules

0 snd\_soc\_iqaudio\_dac

sudo cat /proc/asound/cards

0 [IQaudIODAC ]: IQaudIODAC - IQaudIODAC

IQaudIODAC

sudo aplay -L -- a longer lists of devices

Volume is adjusted for amixer scontrol ‘Digital’ from now on.

### Alsa fixing

sudo alsactl --help

sudo amixer --help

sudo amixer contents -- a lot of information

#### More about proc/asound/modules

Originally cat /proc/asound/modules will show:

0 vc4

1 vc4

2 snd\_soc\_iqaudio\_dac

This is bad. Three sound cards are defined and the last one is the wanted one. Either the order can be changed - which is hard to figure out at the present (try with editing /proc/asound/modules and be sure to change the permission as well). Or get it right in config.txt - see previous pages. I.e. comment out #dtparam=audio=on and the overlays: #dtoverlay=vc4-fkms\*…

### Manually change the order of the sound cards

Instead of doing it in config.txt fix by creating the file /etc/modprobe.d/blacklist.conf and enter:

blacklist snd\_soc\_hdmi\_codec

blacklist snd\_bcm2835

Correct cat /proc/asound/modules file content:

0 snd\_soc\_iqaudio\_dac

sudo lsmod|grep snd -- lists all snd\_<modules>

Before:

…

snd\_soc\_hdmi\_codec 20480 2 #get rid of this std hdmi device

snd\_bcm2835 24576 0 #get rid of headphone jack…

…

After:

…

snd\_soc\_iqaudio\_dac 16384 0 #correct!!

…

#### Option: amixer

amixer numid=3 <value<

Reference: <https://www.sigmdel.ca/michel/ha/rpi/sound_output_03_en.html>

#### vcgencmd

Get settings that are set:

sudo vcgencmd get\_config <config> --for example arm\_freq

sudo vcgencmd get\_config int -- non-zero integer values

sudo vcgencmd get\_config str -- non-null setting

### What is a .asoundrc file? When are they needed?

The .asoundrc file (in user home directory) and /etc/asound.conf (for system-wide settings) are the configuration files for ALSA drivers. Neither file is required for ALSA to work properly. **Most applications will work without them.** The main use of these two configuration files is to add functionality such as routing and sample-rate conversion. It allows you to create "virtual devices" that pre or post-process audio streams. Any properly written ALSA program can use these virtual devices as though they were normal devices.

#### Default PCM device

Using aplay -L you can get a List of existing PCM output devices. If you want the default to be, for example, a USB Device instead of the onboard sound, you can place a pcm.!default line in the .asoundrc.

<https://alsa.opensrc.org/Asoundrc>

### alsa utils

## Look at cmdline.txt in /boot/firmware

It is a one command line represented as a plain text file used by the Raspberry Pi to pass parameters to the kernel during system boot. Parameters must be space separate. They are kernel parameters. Nothing special. There are a few pi-specific options, but they're still for the kernel. Do not edit.

Original Raspberry Pi:

console=serial0,115200 console=tty1 root=PARTUUID=21a4a857-02 rootfstype=ext4 fsck.repair=yes rootwait ipv6.disable=1

Knowledge gained about this mysterious file:

* console=serial0,115200. The kernel creates a serial console on GPIOs, enable\_uart=1 is required in config.txt. Which makes me think that it should be removed?
* console=tty1 is a virtual console.
* root= is the root filesystem and the PARTUUID varies, point outs the internal location.
* rootfstype=ext4 must be the linux file format.
* fsck.repair=yes indicates auto repair attempts on boot partition.
* rootwait. Wait indefinitely for root devices to show up. Useful for devices that are detected asynchronously (e.g. USB and MMC devices - SD cards).

Might add ipv6.disable=1, disable internet version 6. Makes librespot work better.???

# Section: Images

## Create an image with Win32 Image Writer DEPRECATED

### Win 32 Image writer – move the disk to the windows side

**Warning: makes a huge image**

1. In the tool Winscp create an image file named **Xxxx PAn.img** in the directory **E:/7 Player Prod/Debian 12 – Bookworm 2025/**.
2. Open the *Win32 Image Writer* and then find the blue folder button, click.
3. The folder-button launches *MS File Explorer*, find the created empty image file (use \*.\*).
4. Add the file path to the newly created **Xxxx PAn.img** in the long box.
5. ~~Click on “Read Only Allocated Partitions”.~~
6. Click on the **Read** button. **The image will be huge.**

## Create an image with image-utils

### RonR’s image-utils – reside on the linux side

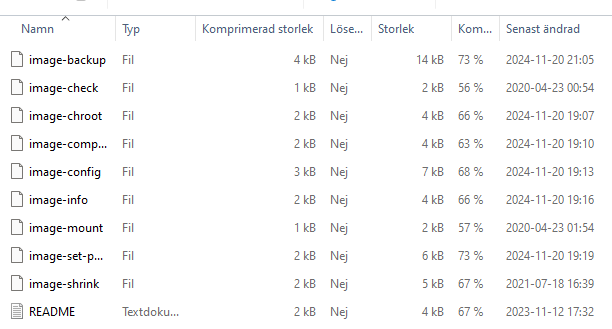
#### i. Get the scripts

Downloaded:

<https://forums.raspberrypi.com/viewtopic.php?t=332000&sid=8a3cd217c59e88889bb1e008765bc3e1>

image.backup is from 2024-11-20 – downloaded 2025-07-10.

Check version by downloading the zip and in Windows Explorer look at the dates…



#### ii. Install the scripts

Simply extract **image-utils.zip** to the Pi and make a directory for the files (/root/image-utils).

sudo unzip -- be sure to be in the script directory

sudo chmod +x image-\* -- make executable, and then add .sh extensions.

### ------------------ after scripts are installed, START HERE:

#### c. Mount external USB ssd that is not the OS usb

Attach the external SSD device to the other USB 3.0 port.

1. Always make the directory /media/usb-ssd – the mount point.  
   sudo mkdir /media/usb-ssd
2. Check that the USB is attached:   
   sudo lsblk

sda 8:0 0 223.6G 0 disk

├─sda1 8:1 0 512M 0 part /boot/firmware

└─sda2 8:2 0 223.1G 0 part /

sdb 8:16 0 931.5G 0 disk

└─**sdb1** 8:17 0 931.5G 0 part -- here is the WD external SSD, it is huge…

Optional:

sudo blkid -o export

: -- a lot of information

DEVNAME=/dev/**sdb1**

LABEL=**Seagate**

UUID=332B-916B

BLOCK\_SIZE=512

TYPE=**exfat**

PARTUUID=332b916b-01

1. Mount the SSD with type exfat. It must be mounted at /media/usb-ssd   
   sudo mount -t exfat /dev/**sdb1** /media/usb-ssd  
   sudo systemctl daemon-reload -- not always needed
2. Check if it got mounted:sudo   
   ls -a /media/usb-ssd   
   …and check with   
   sudo df -Th

#### d. Finally make the image

1. Goto   
   cd /root/image-utils
2. ~~sudo ./image-backup.sh -- starts interactive process~~

There is a problem that the SSD gets in read only mode, always use **no** auto expand?

1. Option **-n** means no auto expand of file system.   
   sudo ./image-backup.sh -n -- image process starts
2. the script becomes interactive, asks for 'Image file to create?',   
   do enter   
   **/media/usb-ssd/images/streamer-nnnn.img**
3. and it then asks 'Initial image file ROOT filesystem size MB [3287]??'   
   add 100 MB more   
   **3387**
4. then it asks 'Added space... etc..' Add 100 MB more,   
   **100**
5. then 'Create /media/usb-ssd/streamer-nnnn.img (y/n)? press 'y'.   
   **y**  
   Wait… and then: Starting full backup… and it is executing the script.
6. Go to   
   cd /. …and check out the content  
   sudo ls -a /media/usb-ssd/images
7. Remember to unmount the external SSD:  
   sudo umount /media/usb-ssd

Detach external SSD with the new image and attach it to the Windows desktop.

Rename the image file if needed and copy to the Windows disk.

### Finally

sudo poweroff or sudo shutdown -h now -- h option means halt or power off

## Flash an image to SSD

* Use balenaEtcher. The tool can discover SSDs without “Enhetsbeteckning”.
* Use an image that have been physically moved to windows (external SSD reattached to Windows desktop), instead of copied from Raspberry pi to Windows over ssh.

## SSD is in read-only mode

If the SSD suddenly flipped into write-protected mode and DiskPart, Registry edits, and formatting tools all fail, it probably tripped its internal controller into read-only fail-safe mode. Some drives do this when they detect flash wear or write issues—even if the drive is brand new.

Open in Windows “kommandotolken”, be sure to be administrator. Enter diskpart. A new window with the name C:\WINDOWS\system32\diskpart.exe opens.

DISKPART>List disk -- choose the right disk number from the list

DISKPART> select DISK=4 -- diskmgmt has the numbers too

Disk 4 is now the selected disk.

DISKPART> attributes disk clear readonly -- removes readonly

Disk attributes cleared successfully.

DISKPART> select DISK=4 -- again…

Disk 4 is now the selected disk.

DISKPART> attributes disk -- check that read only is gone

Current Read-only State : No

Read-only : No

Boot Disk : No

Pagefile Disk : No

Hibernation File Disk : No

Crashdump Disk : No

Clustered Disk

Now the disk can be erased in the diskmgmt tool.

### SSD file system suddenly becomes read-only

The partition sda2 is mounted at root, /.This turn the SSD into r/w again:

mount -o remount,rw /dev/sda2 /

Option? mount -o remount,uid=1000,gid=46,rw /dev/sda2 /

On command mount, the option remount is used on an already-mounted filesystem. This is used to change the mount flags for a filesystem, especially to make a read-only filesystem writable again. It does not change device or mount point.

dmesg | grep sda2

[ 2.870285] sda: sda1 sda2

[ 3.915733] EXT4-fs (sda2): mounted filesystem 748cd48b-29b4-4897-8fc8-ac55438d9000 **ro** with ordered data mode. Quota mode: none.

[ 4.816309] EXT4-fs (sda2): re-mounted 748cd48b-29b4-4897-8fc8-ac55438d9000 **r/w**.

On the Linux side sometimes, the OS can write or format a “locked” SSD that Windows refuses to touch. Use gparted or dd which should be able to wipe it.

## First boot of an image

The first boot takes a longer time if the filesystem is expanded, which requires an extra boot. Led light will be both green and red during this. Network led lights will come and go

sudo df -Th --total -- before expansion of a 240GB SSD

Filesystem Type Size Used Avail Use% Mounted on

udev devtmpfs 955M 0 955M 0% /dev

tmpfs tmpfs 199M 4.7M 194M 3% /run

/dev/sda2 ext4 5.7G 4.8G 660M 88% / -- added 100 MB

tmpfs tmpfs 993M 4.0K 993M 1% /dev/shm

tmpfs tmpfs 5.0M 8.0K 5.0M 1% /run/lock

tmpfs tmpfs 993M 0 993M 0% /tmp

tmpfs tmpfs 993M 8.0K 993M 1% /var/log

tmpfs tmpfs 993M 0 993M 0% /var/tmp

/dev/sda1 vfat 510M 81M 430M 16% /boot/firmware -- added 100 MB

tmpfs tmpfs 199M 0 199M 0% /run/user/0

total - 12G 4.9G 6.3G 44% -

sudo sfdisk -F -- unpartioned space

## Expand the filesystem

sudo raspi-config --expand-rootfs  +

reboot -- note: that the reboot will take longer time than normal.

Do this again:

df -Th --total -- much better…

## wlan1 is gone…

:

12:51:48:986 Boot Phase: resets-------------------[2] started

Device "wlan1" does not exist.

nmcli dev status **-- shows no wlan1! what is this?**

It seems like the Raspberry do not recognise the wlan1 USB after file expansion of a new image.

Solution: Pull out the USB, check the antenna and push it in again - should fix the problem.

Check other CLI’s like /sbin/iw dev wlan0 set ~~power\_save off~~

## Change host name

Edit the files for network:

/etc/hostname

/etc/hosts

Edit Spoitify Connect:

/usr/lib/systemd/system/librespot.service

### Additional stuff

Optional: Read-only mode – Raspberry Pi Specific

This optional step is applicable to a Raspberry Pi only. Run sudo raspi-config and then choose Performance Options > Overlay Filesystem and choose to enable the overlay filesystem, and to set the **boot partition** to be write-protected. (The idea here is that this offers more protection against files being corrupted by the sudden removal of power.)

If it's a Raspberry Pi and you have optionally enabled the read-only mode, you must take the device out of Read-only mode:  
Run sudo raspi-config and then choose Performance Options > Overlay Filesystem and choose to disable the overlay filesystem and to set the boot partition not to be write-protected. This is so that changes can be written to the file system; you can make the filesystem read-only again later. Save the changes and reboot the system.

Special SW <https://github.com/mikebrady/dacquery>

This is the dmesg to watch:

dmesg -HTwx

## Update the Raspberry Pi firmware and bootloader

sudo rpi-update -- updates firmware and bootloader only!

sudo reboot

To check if the bootloader needs an update, one can execute:

sudo rpi-eeprom-update

If the output shows that you need to update your bootloader, you can do so by executing:

sudo rpi-eeprom-update -a

To **completely update** the Raspberry Pi operating system, go to [All about apt](#_All_about_apt) (below)

## shutdown now -h

A slow Raspberry Pi 5 shutdown is often due to the default power settings, which keep the system in a powered-on state even when halted. To speed up the shutdown, modify the EEPROM configuration to enable the POWER\_OFF\_ON\_HALT setting, which will completely shut down the system instead of just halting it, resulting in much lower power consumption and a faster shutdown.

Edit EEPROM config by running

sudo rpi-eeprom-config -e

and make sure the following settings are configured:

[all]

BOOT\_UART=1

WAKE\_ON\_GPIO=0

POWER\_OFF\_ON\_HALT=1

BOOT\_ORDER=0xf461

…and then:

reboot

Check with:

sudo vcgencmd bootloader\_config

sudo raspi-config

Advanced Options -> Shutdown Behaviour -> Full power off

### Otherwise

A slow Linux shutdown is often caused by unresponsive processes that delay the shutdown process, or by a long default timeout in the systemd service manager. To fix this, either manually stop all applications before shutting down, or reduce the DefaultTimeoutStopSec in the /etc/systemd/system.conf or /etc/systemd/user.conf files to a lower value, such as 5 or 10 seconds. Another way to diagnose the issue is to press **Esc** during shutdown to see which process is causing the delay.

sudo systemd-analyze cat-config systemd/system.conf

Check this:

Add the device tree overlay named gpio-poweroff to your /boot/config.txt file:

dtoverlay=gpio-poweroff

## Images

streamer-5001 to 5003 was initial and basic settings, huge sizes

streamer-5003 to 5005 failed…

streamer-5006L bluetooth, image-backup made, [L made on linux side]

*Window side copying is now default*

streamer-5007w bluetooth, moved to windows [w for window side]

streamer-5008w updated OS, streamer

streamer-5009w librespot installed, raspotify too

*No auto expand of filesystem*

streamer-5010 shairport-sync installed, only librespot, no auto expand

streamer-5011 Wi-Fi and AP in place, memory managed

streamer-5012 nodejs, folder structure and npm-package are in place

streamer-5013 journald management

*Software development begins*

streamer-5120 finished start and wi-fi-pages. librespot in error state.

streamer-5124 Network, Bluetooth and librespot works

streamer-5125 Added reset, about and model pages

streamer-5126 Systemd streamer.service, fixed undervoltage

streamer-5127 Redirect console, minor fixes

## Status development SSDs

SSD

|  |  |  |
| --- | --- | --- |
|  | 240 GB | src 5.006/5.007w|5.008w|src 5.009|src 5.012| src 5.125 |
|  | 240 GB | 5.006 linux |src 5.013|inside Tesla |
|  | 240 GB | 5.006 linux |5.007w| src 5.008w|src 5.011|src5.124|src 5.127 |
|  | 240 GB | 5.006 linux |5.009w|src 5010|src 5.120|src 5.126 |

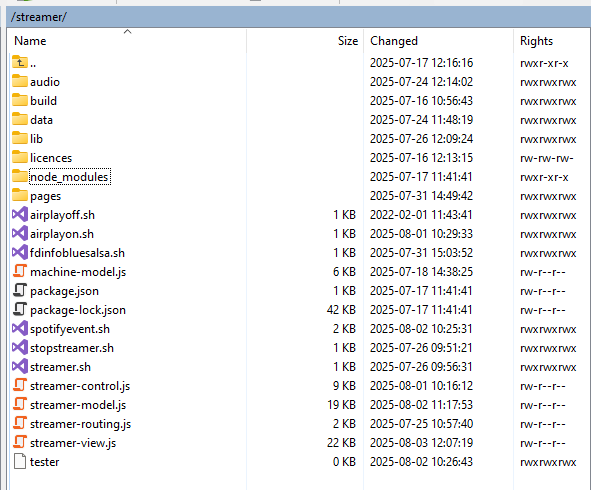
USB

|  |  |  |
| --- | --- | --- |
|  | 8 GB | Player 4.016| |
|  | 8 GB | 5.006 linux |5.007w| |
|  | 16 GB | moode |

### Cleaning up:

sudo rm -r \*.\*

2025-08-07, 5.013+



# All about apt

sudo apt update updates only system package list, no updating yet!!!!!

sudo apt list --upgradable

i) sudo apt full-upgrade -y for a **full update** - might take a long time **recommended**

ii) sudo apt dist-upgrade upgrade all installed packages to their latest version

iii) sudo apt clean cleaning up archives after many upgrades

Special:

apt-get update --allow-releaseinfo-change in order to get the right repository

followed by a reboot and apt update again…

sudo dpkg -s <packagename> check if a package is installed

reboot to make packages work and conf files to be correct

sudo apt remove xxxxx to remove a package

sudo apt remove -y --purge xxxxxx more brutal purge of a package **recommended**

sudo rm -v /etc/apt/sources.list.d/xxxxxxx follow up with this line if loaded externally

sudo apt autoremove to cleanup… left over packages

sudo apt autoclean to cleanup… temp files

sudo apt-mark xxxxx hold on to the current version

sudo apt show -a xxxxx returns all possible versions available by apt

sudo apt list xxxxx prints full version

sudo apt-cache policy xxxxx full version + available information

sudo apt-cache show xxxxx available information

sudo apt-cache search search engine

sudo apt info verbose list of packages

sudo apt install xxxxx install package

sudo dpkg --force-depends -r turn dependency errors into warnings, install whatever…

Also be aware that downloaded package files (.deb files) are kept in /var/cache/apt/archives. You can remove these in order to free up space with sudo apt clean (sudo apt-get clean in older releases of apt).

# Section: More basic configurations

## Expand file system on USB SATA SSD

This should already been done, check with:

sudo df -Th --total

sudo raspi-config --expand-rootfs  +

reboot

du -hs to see disk usage at present directory

df -Th --total display disk system space

fdisk -l shows disk partitions

free -hw shows RAM usage

### OPTIONAL Get Wi-fi up and running

Wi-Fi is now set in the imager before burning the image.

Wi-Fi is initially blocked by rfkill. Use raspi-config to set the country before use.

## Some OS trim details

### Get the target right - disable attach display [p68 in book *Exploring Raspberry Pi*

Remove graphical display mode:

systemctl isolate multi-user.target

(Can be brought back by systemctl set-default multi-user.target)

sudo systemctl get-default should yield multi-user.target (run level 3)

Otherwise do:   
systemctl set-default multi-user.target

### Disable some services in order to speed up boot time [Moode]

In a headless environment no need for keyboard setup--

sudo systemctl disable keyboard-setup.service

Unless there are external joystick or something that uses `/dev/input/` disable this:

sudo systemctl disable triggerhappy.service

### Disable raspi-config or do it last [optional]

sudo systemctl disable raspi-config.service

## Check version

sudo uname -a

[2022] Linux Player 5.10.92-v7l+ #1514 SMP Mon Jan 17 17:38:03 GMT 2022 armv7l GNU/Linux

[2025] Linux Streamer 6.12.25+rpt-rpi-v8 #1 SMP PREEMPT Debian 1:6.12.25-1+rpt1 (2025-04-30) aarch64 GNU/Linux

sudo cat /etc/os-release -- for some longer texts…

Interesting check, maybe…:

sudo vcgencmd measure\_temp

### Remove samba - smbd.service

sudo apt remove -y samba --purge + sudo apt autoremove

## Get git capabilities

sudo apt install -y git -- used for librespot for example…

**All about systemd**

<https://systemd.io/> man systemctl

systemctl list-units --type=service --all

systemctl start .service

systemctl stop .service

systemctl enable .service

systemctl disable .service

systemctl restart .service

systemctl reload .service

systemctl status .service

systemctl is-active .service

systemctl daemon-reload

systemctl edit bluealsa-aplay

Will open the system service file in use.

systemctl show *SERVICE* -- service information

systemctl reload-or-restart SERVICE -- start OR reload

systemctl cat SERVICE -- shows content of unit file

systemctl is-enabled SERVICE

systemctl is-system-running

systemctl list-unit-files

systemctl list-dependencies *UNIT\_FILE*

systemctl log-level

journalctl all journal entries

journalctl -u SERVICE entries for a specific service

journalctl -b entries from the current boot

### Swap space?

swapon --show

If your system doesn’t show one existing, here’s how to create a 2GB swap:

sudo fallocate -l 2G /swapfile

sudo chmod 600 /swapfile

sudo mkswap /swapfile

sudo swapon /swapfile

echo '/swapfile swap swap defaults 0 0' | sudo tee -a /etc/fstab

### Wi-Fi Speed?

Run this command on the Raspberry Pi to start listening:

iperf3 -s

(You’ll need to install the iperf3 package first.)

Run this command on your client device to test the link speed:

iperf3 -c 192.168.1.100

If you pull up the system process list in long format, you can see the ‘niceness’ of a process:

ps -el

The lower the nice value, the higher priority the process gets. A value of 0 has normal priority, but a value of -20 represents the highest priority, meaning the system will dedicate the most resources to do this task quickly. To do so, you’ll need to figure out its process ID (PID).

You can get it by filtering the process list with grep:

and then:

sudo renice -n -20 -p <PID>

# **Section: Bluetooth**

The Bluez package is preloaded and runs with systemd. Configure Bluez. Install the Bluez tools package and finally compile and install Bluez-Alsa.

### Overview

BlueZ is the Linux protocol stack. Streamer uses the userspace management daemon, bluetoothd, that exposes higher-level functionality (via D-Bus APIs) in the interactive tool bluetoothctl. It opens a text prompt where specific actions can be typed to manage everything related to Bluetooth and devices. In addition, the userspace part also includes the important CLI tools for configuration and monitoring. That is the package bluez-tools that gives the CLI-commands.

* Bluez consists of the daemon bluetoothd which gives the tool bluetoothctl (interactive/CLI) and its bt CLI commands.
* BlueALSA consists of the daemon bluealsad, CLI commands as bluealsa-aplay, and bluealsactl

## Bluetooth system parts’ versions

Bluetooth 5.0

Version: bluez 5.66-1+rpt1+deb12u2 [2025-07-02]

(Legacy version for Player 4.xxx – bluez 5.55-3.1+rpt1 all [2022-01-28] (Bluetooth 4.2))

Version bt-commands: version: 2.0~20170911.0.7cb788c-4 [2025-07-02]

Version Blue-Alsa v4.3.1-68-gbc3c2b4-dirty [2025-07-02]

## Bluez – the Bluetooth stack

### Initial steps to set up Bluez – the bt stack

Bluez is the Linux Bluetooth stack and the package bluez should therefore already been installed already. Check with;

sudo dpkg -s bluez

### rfkill

Do not forget: …or just check with rfkill

sudo rfkill unblock bluetooth

### Install bt- command line tools for Bluez

Install bt-tools with CLI functions like bt-adapter, bt-agent, bt-device and bt-network.

sudo apt install bluez-tools -y

## bluetoothd daemon definitions for systemd

Change the file: /etc/systemd/system/bluetooth.target.wants/bluetooth.service

#Below the right configuration; add the --noplugin=sap

ExecStart=/usr/libexec/bluetooth/bluetoothd --noplugin=sap

## Configure Bluetooth – main.conf and input.conf

**1.** Open file /etc/bluetooth/main.conf and uncommented the following entries:

Class = 0x00041C #add new value (Moode has 0x2c041c)

04 = Major Device Class: Audio/Video

1c = Minor Device Class: Portable Audio (speakers/headphones)

DiscoverableTimeout = 0 --stays discoverable forever

PairableTimeout = 0

FastConnectable = true

ReconnectAttempts=7

ReconnectIntervals=1,2,4,8,16,32,64

**2.** Open file /etc/bluetooth/input.conf and add the following entries:

[General]

Enable=Source,Sink,Media,Socket

## Start bluetooth.service with Systemd

Reload and restart:

sudo systemctl daemon-reload

sudo service bluetooth restart – the daemon bluetoothd restarts

Note: there is no need for sudo systemctl enable bluetooth, it is already enabled.

Then check with:

sudo systemctl status bluetooth --no-pager --full

### Error message in the journal part of systemctl status bluetooth

There are errors here -often old errors.

## Disable LE - only BR/EDR is wanted…

Disable low energy connections:

sudo btmgmt le off

hci0 Set Low Energy complete, settings: powered fast-connectable ssp br/edr secure-conn

Check with:

sudo btmgmt info

Index list with 1 item

hci0: Primary controller

addr 2C:CF:67:75:F6:A1 version 9 manufacturer 305 class 0x00041c

supported settings: powered connectable fast-connectable discoverable bondable link-security ssp br/edr le advertising secure-conn debug-keys privacy configuration static-addr phy-configuration

current settings: powered fast-connectable ssp br/edr secure-conn

name Streamer

short name

hci0: Configuration options

supported options: public-address

missing options:

### Also check the journald service

journalctl -u bluetooth -- Note: sap should have been fixed in service file

journalctl | fgrep -i blue -- sap plug in is not used anymore

journalctl -u bluetooth -b

### Error message in the journal d service

There are errors here -often old errors.

## BlueALSA - Bluetooth Audio ALSA Backend

BlueALSA is designed for audio systems where the high-level audio systems like Streamer. PulseAudio or PipeWire are not required. The target system application is interfacing directly with ALSA, with only one application at a time using each Bluetooth audio stream. In such systems. BlueALSA adds Bluetooth audio to the ALSA sound card support. BlueALSA acts as a proxy between BlueZ and ALSA.

BlueALSA consists of the daemon bluealsad, CLI commands as bluealsa-aplay, bluealsactl and in addition ALSA plug-ins. Moode supports Bluetooth aptX|LDAC|SBC XQ+

## Compile BlueALSA

<https://github.com/Arkq/bluez-alsa/wiki/Installation-from-source> - this is the wiki.

### A. Install required packages:

sudo apt-get install git automake build-essential libtool pkg-config python3-docutils

sudo apt-get install libasound2-dev libbluetooth-dev libdbus-1-dev libglib2.0-dev libsbc-dev -- libglib2.0-dev was found later

sudo apt-get install libdbus-1-3 -- libdbus-1-3 already installed

### B. Get the source code:

Make a working directory under root.

root~#pwd -- check where you are

/root

root~#mkdir bluez-alsa -- create the directory

root~#ls -a -- look for the directory

. .. .bash\_history .bashrc **bluez-alsa** .lesshst .local .profile .ssh

Clone the BlueALSA source code from Github

git clone https://github.com/arkq/bluez-alsa.git

cd bluez-alsa

#### Auto configuration problems – both in 2022 and 2025

Prepare the auto configuration. It is very important to do this step after the essential requirements installation of the step above. In the /bluez-alsa directory run the automatic reconfiguration as follows:

autoreconf --install --force

warning: couldn't open directory 'm4': No such file or directory

Set the right access to the file /root/bluez-alsa/configure.ac – 0777.

This worked better with the –verbose

autoreconf -ivf

Create a new directory:

mkdir build

cd build

### C. Choose BlueALSA options

#### Codecs of interest

--enable-aac -- not a package

*It's a successor to MP3. AAC is the default audio format for Apple and YouTube, and some Android*

--enable-aptx

aptX is like ACC, but for various of devices other than Apple.

--enable-aptx-hd

High-resolution aptX.

--with-libopenaptx

Required for aptX and aptX HD.

--enable-faststream

For headphones.

--enable-lc3plus -- not a package

*Also, for headphones and LE headphones. Designed for Bluetooth.*

--enable-ldac -- missing header file ldacBT.h

*Sony’s high-resolution audio codec.*

--enable-mp3lame

Mp3 support.

#### More options

--enable-a2dpconf

The CLI command a2dpconf presents the fields of the given A2DP codec CONFIG in a human-readable format.

--enable-hcitop

A tool that provides a dynamic real-time view of activity statistics for each HCI interface.

--enable-manpages

Man pages for bluealsad daemon, the BlueALSA ALSA plugins, and the enabled utilities.

--enable-systemd

Install default systemd service unit files for bluealsad and bluealsa-aplay.

#### Install these dependencies

sudo apt-get install libopenaptx-dev -y -- aptX lib installed

sudo apt-get install libmp3lame-dev -y -- mp3 lib installed

#### Packagse missing

sudo apt-get install libasound2-dev was the trick

sudo apt-get install libbluetooth-dev was required.

sudo apt-get install libdbus-1-dev was missing.

sudo apt-get install libsbc-dev

sudo apt-get install libsbc1 were missing

sudo apt-get install libncurses-dev -y

### D. Configure with options

Be in the right directory:

root~/bluez-alsa/build#

In the build directory

../configure --enable-aptx --enable-aptx-hd --with-libopenaptx --enable-faststream --enable-mp3lame --enable-a2dpconf --enable-hcitop --enable-manpages --enable-systemd

### E. Compile

In the build directory

make

sudo make install

#### Options not choosen

* acc - since the packages is not in repository, requires <https://github.com/mstorsjo/fdk-aac>
* lc3plus – no package
* LDAC missing file ldacBT.h
* HSP/HFP - no phone headphones… Device should only be connected for audio!

### F. Runtime dependencies

bash-completion apt-get install bash-completion ---in place, all below

dbus apt-get install dbus -y

ibasound apt-get install libasound2

libbluetooth apt-get install libbluetooth3

libbsd apt-get install libbsd0

libopenaptx apt-get install libopenaptx0

libsamplerate apt-get install libsamplerate0

libsbc apt-get install libsbc1

libunwind apt-get install libunwind8

mp3lame apt-get install libmp3lame0

ncurses **apt-get install libncurses5** -- installed

**apt-get install libncurses6** -- installed

readline apt-get install libreadline7 -- non existing!

apt-get install libreadline8 -- in place

spandsp **apt-get install libspandsp2** -- installed

## Installed BlueALSA codecs

sudo bluealsad –help -- codecs are listed at the bottom

… shows at the bottom available BT audio codecs:

**a2dp-source**: SBC\*, MP3, aptX, aptX-HD, FastStream

**a2dp-sink**: SBC\*, MP3, aptX, aptX-HD, FastStream

**hfp**-...: CVSD -- of no interest

\*) SBC (Sub-band Codec) is a mandatory Bluetooth audio codec used in the Advanced Audio Distribution Profile (A2DP)

Note: (this is change from older versions of bluealsad)

Native A2DP volume control is the default (they dropped --a2dp-volume option).

--initial-volume=NUM initial volume level [0-100] --- tried this – no effect

Reference: installed codecs Debian 11 – Player 4.xxxx from 2022

Available BT audio codecs:

a2dp-source: SBC, MP3, aptX

a2dp-sink: SBC, MP3, aptX

hfp-\*: CVSD

## BlueALSA service - daemon bluealsad for systemd

### General information

The main component of BlueALSA (bluez-alsa) is bluealsad. It has to run as a. It will register org.bluealsa service in the D-Bus system bus. The daemon acts as a proxy between BlueZ and ALSA. bluealsad makes the Streamer to be a sink to phones that are streaming bt and as a source to a bt speaker.

<https://github.com/Arkq/bluez-alsa/wiki/Systemd-integration>

Legacy: v3.1.0-77-gac5706f (2021-11-16).

Version: v4.3.1-68-gbc3c2b4-dirty (2025-07-03).

Visit ALSA PCM configuration files for BluezAlsa at: (not BluezALSA system configs)

Directory: /etc/alsa/conf.d/ -- BlueALSA local config file (20-bluealsa.conf)

### Group memberships for services

sudo adduser --system --group --no-create-home bluealsa

sudo adduser --system --group --no-create-home bluealsa-aplay

sudo adduser --system --group --no-create-home bluez-agent

sudo adduser bluealsa-aplay audio

## bluealsad daemon definition for systemd

In the preexisting file /usr/lib/systemd/system/bluealsa.service add the following:

[service]

User=root

ExecStart=/usr/bin/bluealsad -p a2dp-source -p a2dp-sink --all-codecs

Tried to fix the volume control problem with this:

ExecStart=/usr/bin/bluealsad -p a2dp-source -p a2dp-sink --all-codecs --initial-volume=0 – nothing happened, might be override by bluealsa-aplay!

**No worries.**

### Moode’s Exec-line

ExecStart=/usr/bin/bluealsa -p a2dp-source -p a2dp-sink -c aptx -c aptx-hd -c ldac --sbc-quality=xq+

Reload and restart:

sudo systemctl daemon-reload

sudo service bluealsa restart

sudo systemctl status bluealsa

#### Older versions

*1) ExecStart=/usr/bin/bluealsa -p a2dp-sink -p a2dp-source*

*2) ExecStart=/usr/bin/bluealsa -p a2dp-source -p a2dp-sink -p hfp-hf -p hsp-hs*

As a reference see this issue: <https://github.com/Arkq/bluez-alsa/issues/339>

## BlueALSA player service - bluealsa-aplay for systemd

Bluealsa-aplay captures bt streaming into alsa for analogue output (default) or stream to a bt speaker (requires /etc/asound.conf file).

## bluealsa-aplay daemon definition for systemd

In the preexisting file /usr/lib/systemd/system/bluealsa-aplay.service add the mac address, the name and run as root

[service]

User=root

ExecStart=/usr/bin/bluealsa-aplay 00:00:00:00:00:00 --volume=software

Fixed the native volume control problem with volume=software - this works for aptX!

There is a native volume control problem with aptX. BlueZ release 5.65 or later (2025-07-07 version is 5.66) should be used and then (maybe) native A2DP volume control will always be available (with those devices which provide it). Using aptX requires software control and alsa card should be at 100% volume, do sudo amixer -c 0 set Digital 100%

Sound check

amixer -c 0 get Digital toggle --get volume

amixer -c 0 set Digital toggle -- mute and unmute

amixer -c 0 set Digital 25% -- -c is card, set volume

When streaming check:

sudo amixer -D bluealsa scontrols

Simple mixer control 'Galaxy S21 Ultra 5G A2DP',0

### Streamer is the sink and connected to phone

Yields – **aptX** is used:

bluealsa:DEV=8C:DE:E6:25:C5:8C,PROFILE=a2dp,SRV=org.bluealsa

Galaxy S21 Ultra 5G, phone, capture

A2DP (aptX): S16\_LE 2 channels 44100 Hz

#### As a reference:

Player 4.xxx connected to phone [2022] yields – **SBC** is used:

bluealsa:SRV=org.bluealsa,DEV=8C:DE:E6:25:C5:8C,PROFILE=a2dp

Galaxy S21 Ultra 5G, trusted phone, capture

A2DP (SBC): S16\_LE 2 channels 44100 Hz

### Streamer is the source and connected to Eneby bt-speaker

bluealsa-aplay -L

Yields – **SBC** is used:

bluealsa:DEV=FC:58:FA:ED:57:60,PROFILE=a2dp,SRV=org.bluealsa

ENEBY30, audio-headphones, playback

A2DP (SBC): S16\_LE 2 channels 48000 Hz

Reload and restart:

sudo systemctl daemon-reload

sudo service bluealsa-aplay restart

sudo systemctl status bluealsa-aplay

## Bluez bt-agent service –for systemd

### bt-agent as it was supposed to work

The CLI for bluetooth was installed before (see the chapter Bluez tools above). One of quite many drawbacks of using Bluetoothctl is that closing the tool actually unregisters the agent - which is problematic. The solution is to run an agent that can be daemonized with systemd.

## bluez-agent daemon definition for systemd

Create a new file /usr/lib/systemd/system/bluez-agent.service

[Unit]

Description=Bluetooth Auth Agent

After=bluetooth.service

PartOf=bluetooth.service

[Service]

Type=simple

ExecStart=/usr/bin/bt-agent -c NoInputNoOutput

User=root

[Install]

WantedBy=bluetooth.target

### Moode added

KillSignal=SIGINT

ExecStartPost=/bin/sleep 1

ExecStartPost=/bin/hciconfig hci0 sspmode 1 -- Simple Pairing Mode

## bt-agent work around

Even worse, for bluez version 5.50 the agent NoInputNoOutput do not work as it should. It is really a big mess. Instead, the agent type DisplayOnly is choosen. That agent requires pin codes - which have to be avoided (an unwanted behaviour that the former agent took care of). The workaround is to use a file with the allowed (pattern of) pins and set it to \* \* which makes the agent accept any pin or no pin. Well, it works. … and for the moment [in 2022] stuck with 5.55 since every bluez package in Raspberry has to have an armf specific customization. Compile the latest version directly from source will not work.

Create a new file: /usr/lib/systemd/system/bluez-agent.service

[Unit]

Description=Bluetooth Auth Agent

After=bluetooth.service

PartOf=bluetooth.service

[Service]

Type=simple

#ExecStart=/usr/bin/bt-agent -c NoInputNoOutput --NOTE: does not work anymore!!!

ExecStart=/usr/bin/bt-agent -c DisplayOnly -p /streamer/data/pins

User=root

[Install]

WantedBy=bluetooth.target

**Note** -c is capability and -p is pin

Create the file /streamer/data/pins and add: \* <space> \*

\* \*

Set permission to 0777 which systemd says: Warning! /streamer/data/pins is world readable!

## Start the services with systemd

systemctl start bluealsa

systemctl status bluealsa

systemctl enable bluealsa

systemctl start bluealsa-aplay

systemctl status bluealsa-aplay

systemctl enable bluealsa-aplay

systemctl start bluez-agent

systemctl status bluez-agent

systemctl enable bluez-agent

Note: If the tab is pressed twice after agent when using bluetoothctl there will be a list of available agents and commands. simple-agent is bluetoothctl’s own agent.

DisplayOnly KeyboardDisplay NoInputNoOutput on

DisplayYesNo KeyboardOnly off

#### bthelper - no action

Note: the special help service config file can be found here (no need for changes for now): /usr/lib/systemd/system/bthelper@.service

#### [Optional] Set bluetoothctl agent using command bt.agent

sudo bt-agent -c NoInputNoOutput -d -- optional, but have to be done every time

sudo bt-agent --capability=NoInputNoOutput -- another way

There is a -d flag for daemon, maybe use that? [option, but systemd is better]

#### [Optional] Reset bluetoothctl agent manually

Note requires to systemctl disable blues-agent, followed by:

sudo systemctl disable bluez-agent

sudo bluetoothctl agent off

sudo bluetoothctl agent NoInputNoOutput

sudo systemctl enable bt-agent

sudo bluez-agent --capability=NoInputNoOutput -- another way

## Journald commands for bluetooth

sudo journalctl -xeu bluealsa.service

sudo journalctl -xeu bluealsa-aplay.service

sudo journalctl -u bluetooth

sudo journalctl | grep -i blue

sudo journalctl -u blue -b

### Some errors 2022-01-28 and how they were fixed

1 **bluealsa-aplay**[509]: /usr/bin/bluealsa-aplay: W: Couldn't get BlueALSA PCM list: The name org.bluealsa was not provided by any .service files

**FIX**: added Name=org.bluealsa under [service]

**2 bluetoothd**[521]: Failed to set privacy: Rejected (0x0b) ERROR

**Known error**. Persistant.

**3 bluealsa**[510]: /usr/bin/bluealsa: W: Couldn't register hands-free profile: GDBus.Error:org.bluez.Error.AlreadyExists: Already Exists

**4 bluealsa**[510]: /usr/bin/bluealsa: W: Couldn't register media endpoint: GDBus.Error:org.bluez.Error.AlreadyExists: Already Exists

**FIX**: a new systemd file was put in place, ref: <https://github.com/Arkq/bluez-alsa/wiki/Systemd-integration>

**5.** **bluealsa**[522]:W ../../../gio/gdbusobjectmanagerserver.c:1098: Error registering manager at /org/bluez/hci0/battery: An object is already exported for the interface org.freedesktop.DBus.ObjectManager at /org/bluez/

FIX: unknown??

For now these error and warnings might be okay. . .

### Some errors 2025-07-06 - bluetoothd in journalctl -u bluetooth

profiles/audio/vcp.c:vcp\_init() D-Bus experimental not enabled

src/plugin.c:plugin\_init() Failed to init vcp plugin

**No worries**: Volume Control Profile (VCP) controls the volume of Bluetooth Low Energy (BLE) streaming.

profiles/audio/mcp.c:mcp\_init() D-Bus experimental not enabled

src/plugin.c:plugin\_init() Failed to init mcp plugin

**No worries**: voice-to-text transcription

profiles/audio/bap.c:bap\_init() D-Bus experimental not enabled

src/plugin.c:plugin\_init() Failed to init bap plugin

**No worries**: Bom Bap… some music sound stuff???

profiles/sap/server.c:sap\_server\_register() Sap driver initialization failed.

sap-server: Operation not permitted (1)

**No worries:** "SAP" (SIM Access Profile) driver manages SIM cards and is not required.

### Check with aplay

sudo aplay -L

null

Discard all samples (playback) or generate zero samples (capture)

bluealsa

Bluetooth Audio

hw:CARD=IQaudIODAC,DEV=0

IQaudIODAC, IQaudIO DAC HiFi pcm512x-hifi-0

Direct hardware device without any conversions

plughw:CARD=IQaudIODAC,DEV=0

IQaudIODAC, IQaudIO DAC HiFi pcm512x-hifi-0

Hardware device with all software conversions

default:CARD=IQaudIODAC

IQaudIODAC, IQaudIO DAC HiFi pcm512x-hifi-0

Default Audio Device

sysdefault:CARD=IQaudIODAC

IQaudIODAC, IQaudIO DAC HiFi pcm512x-hifi-0

Default Audio Device

dmix:CARD=IQaudIODAC,DEV=0

IQaudIODAC, IQaudIO DAC HiFi pcm512x-hifi-0

Direct sample mixing device

### More tests for the bluetooth system

amixer -D bluealsa sset " IQaudIODAC" 50%

aplay --verbose --vumeter=stereo -c 2 <sound file>

aplay -vv -D bluealsa:DEV=XX:XX:XX:XX:XX:XX <sound file>

## Group membership for stream and root

All groups:

sudo less /etc/group -- many groups

Find out members of a group:

sudo getent group bluetooth

bluetooth:x:112:pi,root

Check with:

sudo groups -- yields just root

The benefit is that adding to groups helps to use audio files from default stream user, without having to elevate privileges for each bluetooth action. The following command adds stream (=U) user to bluetooth group:

sudo adduser U bluetooth -- works as well, but shorter

sudo adduser U audio -- works as well, but shorter

sudo adduser root bluetooth -- works as well, but shorter

sudo adduser root audio -- works as well, but shorter

sudo usermod -G bluetooth -a pi -- optional way to do the same as above

sudo reboot -- eventually

### Turn off and on Bluetooth use rfkill

sudo rfkill list -- lists soft blocked wired devices

sudo rfkill block bluetooth

sudo rfkill unblock bluetooth

## Some bluetoothctl as commands

### Scan 15 seconds - without finding the PID

sudo bluetoothctl --timeout 15 scan on>/dev/null

### Manual procedure to make Streamer discoverable

bluetoothctl

default-agent >>>

discoverable on > /dev/null >>>

Note: this should be automated by bt-agent, but the interface is used by bluetoothctl.js

### Information about devices in bluetoothctl

Here is a variant:

sudo bluetoothctl devices | cut -f2 -d' ' | while read uuid; do bluetoothctl info $uuid; done|grep -e "Device\|Connected\|Name"

## Correct start and stop systemd sequence

sudo systemctl stop bluetooth

sudo systemctl start bluetooth

sudo systemctl restart bluealsa

sudo systemctl restart bluealsa-aplay

## More on systemd on bluetooth and blue\*

There are two bluetooth daemons running by systemd:

1. bluetooth.target - Bluetooth. Docs: man:systemd.special(7)
2. bluetooth.service - Bluetooth service. Docs: man:bluetoothd(8)

### All bluetooth related services

sudo systemctl --all | grep -i -E 'blue|hci|bt' -- has to be grep here!

**..sys-devices-platform-soc-fe201000.serial-tty-ttyAMA0-hci0.device**

loaded active plugged /sys/devices/platform/soc/fe201000.serial/tty/ttyAMA0/hci0

**sys-subsystem-bluetooth-devices-hci0.device**

loaded active plugged /sys/subsystem/bluetooth/devices/hci0

**bluealsa-aplay.service**

loaded active running BlueALSA player service

**bluealsa.service**

loaded active running Bluealsa daemon

**bluetooth.service**

loaded active running Bluetooth service

**bt-agent.service**

loaded active running Bluetooth Auth Agent

**bthelper@hci0.service**

loaded active exited Raspberry Pi bluetooth helper

**hciuart.service**

loaded active running Configure Bluetooth Modems connected by UART

**system-bthelper.slice**

loaded active active system-bthelper.slice

**bluetooth.target**

loaded active active Bluetooth

### Further reading - internal RAD documents

Bluealsa install and compile rev B

Bluetoothspkr rev C

## HCI trace and btmon and hcitop and some debugging commands

Trace on screen only:

sudo btmon

sudo btmon 2>&1

Save to file:

sudo btmon --write ./hcitrace.snoop | tee ./hcitrace.txt

sudo dmesg -H | grep <nnn> -- kernel messages

sudo hcitop -- a simple dynamic view of HCI activity  
<https://github.com/Arkq/bluez-alsa/blob/master/doc/hcitop.1.rst>

cat /dev/input/event0 | xxd -- hex conversion tool, might be useful

## More information about Bluetooth signal

sudo hcitool lq 8C:DE:E6:25:C5:8C

Link quality: 255 -- Value range is 0 to 255

### Some known problems (2002-03-20)

journalctl -u bluetooth [Player cannot find bt speaker]

-- Journal begins at Thu 2022-03-10 14:14:19 EST, ends at Thu 2022-03-10 14:19:52 EST. --

:

Mar 10 14:19:30 Player bluetoothd[514]: profiles/audio/avdtp.c:avdtp\_connect\_cb() connect to FC:58:FA:ED:57:60: Host is down (112)

… more from journalctl bluetooth when it refuses to connect to speaker, error:

Mar 10 13:47:37 Player bluetoothd[504]: profiles/audio/avdtp.c:avdtp\_connect\_cb() connect to FC:58:FA:ED:57:60: Permission denied (13)

Mar 10 13:52:31 Player bluetoothd[504]: src/service.c:btd\_service\_connect() a2dp-sink profile connect failed for FC:58:FA:ED:57:60: Protocol not available

A restart of the bluetooth service was done, it stalled, but came up again. That fixed the connection failure.

journalctl -u bluealsa

At the beginning especially no missing RTP packets, but then burst of missing RTP packets - 4 per second and minor stuttering.

:

Mar 10 14:08:13 Player bluealsa[940]: /usr/bin/bluealsa: W: Missing RTP packet: 57652 != 57651

:

journalctl -u bluealsa-aplay

-- Journal begins at Thu 2022-03-10 11:50:51 EST, ends at Thu 2022-03-10 12:33:34 EST. --

Mar 10 11:51:02 Player systemd[1]: /lib/systemd/system/bluealsa-aplay.service:20: Unknown key name 'Name' in section 'Service', ignoring.

Mar 10 11:51:02 Player systemd[1]: Started BlueALSA player service.

Mar 10 11:51:02 Player bluealsa-aplay[493]: /usr/bin/bluealsa-aplay: W: Couldn't get BlueALSA PCM list: The name org.bluealsa was not provided by any .service files

Mar 10 13:54:16 Player bluealsa-aplay[493]: /usr/bin/bluealsa-aplay: W: Couldn't open mixer: Mixer element not found

Another problem is that bluetoothctl is always in discovering mode and scans all the time.

# All about bt-commands

bt-adapter

* List available adapters
* Show information about adapter (incl properties)
* Discover remote devices (with remote device name resolving)
* Change adapter properties (eg. Name, Discoverable, Pairable, etc)

bt-agent

* Manage incoming Bluetooth requests (eg. request of pincode, request of authorize a connection/service request, etc)

bt-audio

* Connecting to audio devices

bt-device

* List added devices
* Connect to the remote device by his MAC, retrieve all SDP records and then initiate the pairing
* Disconnect the remote device
* Remove device (and also the pairing information)
* Show information about device (incl properties)
* Service discovery
* Change device properties (eg. Name, Trusted, Blocked, etc)

bt-input

* Connecting to input devices

bt-monitor

* Capturing D-Bus signals of bluetoothd

bt-network

* Connect to the network device
* Register network server for the provided UUID (gn/panu/nap)

bt-serial

* Connects to a specific RFCOMM based service on a remote device and then creates a RFCOMM TTY device for it

bt-obex

* Agent (to accept/reject incoming bluetooth object push requests) for OBEXD (OPP/FTP profile)
* Send local file to the specified remote device using object push profile
* Start FTP session with remote device

# Section: Install raspotify

Give raspotify a new chance? Before it could not handle events and the librespot version is always old.

## Install raspotify

sudo apt-get -y install curl && curl -sL https://dtcooper.github.io/raspotify/install.sh | sh

sudo systemctl status raspotify

…it is already up and running.

Check out the wiki: <https://github.com/dtcooper/raspotify/wiki>

## Librespot version

librespot 0.6.0 383a6f6 (Built on 2025-01-17). [timestamp: 2025-07-11 05:50:04].

* librespot can be found in /bin – which means that it could be started directly, like in Player 4.016. Or compile a later version and run it in Raspotify.

## Configuration

This is the file: /etc/raspotify/conf

LIBRESPOT\_AUTOPLAY=on -- Automatically play similar songs.

LIBRESPOT\_NAME="Streamer"

LIBRESPOT\_BITRATE="320"

LIBRESPOT\_INITIAL\_VOLUME="35"

LIBRESPOT\_VOLUME\_CTRL="cubic"

#TMPDIR=/tmp

sudo systemctl restart raspotify

As a reference, this is how librespot was started with systemd:

ExecStart=/root/.cargo/librespot/target/release/librespot  
-n Player -b 320 --initial-volume 35 --volume-ctrl cubic --disable-audio-cache --enable-volume-normalisation --backend alsa --onevent /player/spotifyevent.sh

## raspotify daemon definition for systemd

/var/lib/systemd/deb-systemd-helper-enabled/raspotify.service

/etc/systemd/system/multi-user.target.wants/raspotify.service

/lib/systemd/system/raspotify.service

## Stop Raspotify

sudo systemctl stop raspotify.service

sudo systemctl disable raspotify.service

Removed "/etc/systemd/system/multi-user.target.wants/raspotify.service".

sudo systemctl status raspotify.service

Rename /bin/librespot to /bin/librespotRASP

## Completely uninstall Raspotify

sudo apt purge -y raspotify

Remove the repo:

sudo rm -f /etc/apt/sources.list.d/raspotify.list

Delete the repo's key:

sudo rm -f /usr/share/keyrings/raspotify\_key.asc

# Section: Install librespot

Compile and then install librespot from source found in Crate (the Rust depository of crates), with Cargo (the package manager of Rust) and Rust itself, the general-purpose programming language.

## Librespot version

Version: librespot 0.6.0 VERGEN\_IDEMPOTENT\_OUTPUT [2025-07-11]

Legacy: librespot version: 0.3.1 [2021-11-23]

## Sources and tools for librespot

Install librespot, as described on Github, using Crate and Cargo:

<https://crates.io/crates/librespot>

References: <https://crates.io/> “The Rust community’s crate registry”

<https://rustup.rs/> <https://doc.rust-lang.org/cargo/index.html>

<https://github.com/librespot-org/librespot>

## Install Cargo and Rust

curl https://sh.rustup.rs -sSf | sh

Rust is installed now. Great!

To get started you may need to restart your current shell.

This would reload your PATH environment variable to include

Cargo's bin directory ($HOME/.cargo/bin).

To configure your current shell, you need to source

the corresponding env file under $HOME/.cargo.

This is usually done by running one of the following (note the leading DOT):

. "$HOME/.cargo/env" # For sh/bash/zsh/ash/dash/pdksh

source "$HOME/.cargo/env.fish" # For fish

. "$HOME/.cargo/env"

Restart shell now - important:

sudo logout -- open up with PuTTy again

… then compile using option 1. or 2. below….

## 

## 1. GITHUB FORK: Compile librespot from github with cargo

### First try to update rustup, if it is installed - maybe not?

sudo rustup update

sudo rustup component add rustfmt -- check not needed

sudo rustup component add clippy -- check not needed

After installation do:

sudo source $HOME/.cargo/env -- add cargo to PATH

sudo echo $PATH -- check PATH

/root/.cargo/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin

Then add the required packages: [should already be in placed since bluealsa-aplay was built)

sudo apt install libasound2-dev pkg-config

## Fork librespot from Github and check the ssh key

Fork the repo, so that there will be a copy with read/write access available, but first check if the ssh key is in the Callenberg Github account is in place, pretty complicated… (see the document *All about git 2025* )

Keys should be present in the Github account and in /root/.ssh/. But the permission for file /root/.ssh/id\_ed26619, the secret key, must be 0600, otherwise Github will not accept the key.

e-mail from Github regarding the keys when they are created:

The following SSH key was added to your account:

player 4  
SHA256:……………..CgC0+OD6rUWplh8dNJQy+O9K5GCIol/Z………

### Test that there is connection to Github

ssh -T git@github.com

The authenticity of host 'github.com (140.82.114.4)' can't be established.

ECDSA key fingerprint is SHA256:p2QAMXNIC1TJYWeIOttrVc98/R1BUFWu3/LiyKgUfQM.

Are you sure you want to continue connecting (yes/no/[fingerprint])? Y

Please type 'yes', 'no' or the fingerprint: yes

Warning: Permanently added the ECDSA host key for IP address '140.82.113.4' to the list of known hosts.

Expected and correct response

Hi Callenberg! You've successfully authenticated, but GitHub does not provide shell access.

Reference: document “*All about git 2025.doc*”, can be found in ***7. Player Prod***.

## Clone the fork to .cargo directory and get the Cargo.toml file right

### 0. In case, **REMOVE** any old librespot in the .cargo directory

Here is how to do that - [fix .cargo](#_Remove_librespot_files)

### 1. First **FORK** to github Callenberg -- **IMPORTANT**:

Remove any existing fork - go to *Setting*, continue to *Danger Zone* and delete the repository.

Fork librespot to github Callenberg.

The update box should read: ‘*This branch is up to date with librespot-org:dev.*’.

sudo cd .cargo -- important, otherwise the Cargo.toml cannot be easily found

root~/.cargo# -- this is the directory to be in…

### 2. …and now **CLONE**: (the directory librespot should not exists before cloning)

Be sure that the right version is cloned - not and old one.

sudo git clone git@github.com:Callenberg/librespot.git

Cloning into 'librespot'...

Warning: Permanently added the ECDSA host key for IP address '140.82.113.3' to the list of known hosts.

remote: Enumerating objects: 9489, done.

remote: Counting objects: 100% (115/115), done.

remote: Compressing objects: 100% (86/86), done.

remote: Total 9489 (delta 46), reused 72 (delta 29), pack-reused 9374

Receiving objects: 100% (9489/9489), 3.36 MiB | 4.46 MiB/s, done.

Resolving deltas: 100% (5848/5848), done.

### 3. Find the clone

Now there is a new directory under .cargo called librespot, go there:

sudo cd /root/.cargo/librespot -- this directory that will be used for building

This is the right directory to be in: root~/.cargo/librespot

Note: This avoids the ”could not find `Cargo.toml` in `/root` or any parent directory” problem. The wanted file Cargo.toml is now in the librespot directory. Find the version line:   
version = "0.7.1" - this is the latest in September 2025.

sudo apt-get install build-essential

sudo apt-get install libasound2-dev pkg-config

## Do cargo build

The default for librespot is to build with native-tls (system TLS[[1]](#footnote-1)), **Rodio for audio** playback, and libmdns for discovery, but we do not want **Rodio**! We must then specify at least one **TLS backend** along with **alsa** and **discovery backends**. Note that librespot requires at least one TLS backend to function. Building with --no-default-features alone will fail compilation.

TLS backend: use native-tls for maximum compatibility.

Audio: use alsa-backend directly - no Rodio!

Discovery: use with-libmdns, works on all platforms.

Also use --release since this is production not development.

Finally, now build librespot in the directory /root/.cargo/librespot - it will require 414 packages expect 34 mins! **No sudo below…**

cargo build --no-default-features --features " native-tls alsa-backend with-libmdns" --release

In case there is no TLS:

error: Either feature "native-tls" (default), "rustls-tls-native-roots" or "rustls-tls-webpki-roots" must be enabled for this crate.

Note: --no-default-features -- means no Rodio backend

Reference: <https://github.com/librespot-org/librespot/blob/dev/COMPILING.md>

<https://github.com/librespot-org/librespot/wiki/Compiling#addition-features>

### Backup the keys - both of them

Be sure to copy the keys to PC. (ssh folder under Bookworm in ***7. Player Prod*** ).

## The librespot bin file can be found here

Here is where to find the bin:   
/root/.cargo/librespot/target/release/librespot

[Quick command: cd ./target/release/librespot ]

~~Create a soft link of the file in /root/.cargo/bin :~~

~~ln -s /root/.cargo/librespot/target/release/librespot /root/.cargo/bin/librespot~~

~~(/usr/local/rust/cargo might be a better candidate following Linux practises and traditions.)~~

No, copy the bin file in to the /bin, then check with:

which librespot

/usr/bin/librespot

Check build version with:

librespot -V

: :

**September 2025**

librespot 0.7.1 df5f957 (Built on 2025-09-17, Build ID: 5U2hcPB9, Profile: release)

**2025-08-06:**

librespot 0.6.0-dev be37402 (Built on 2025-08-06, Build ID: CJxJJDOC, Profile: release)

**2025 July - Crate version**

librespot 0.6.0 VERGEN\_IDEMPOTENT\_OUTPUT (Built on 2025-07-11, Build ID: DTSg4t5r, Profile: release) librespot 0.3.1 f4be9bb (Built on 2022-01-29, Build ID: xt908EGz, Profile: release)

Check path as well:

echo $PATH

/root/.cargo/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin

### Backends

/root/.cargo/librespot/target/release/librespot --backend ?

[2025-09-17T16:19:56Z INFO librespot] librespot 0.7.1 df5f957 (Built on 2025-09-17, Build ID: 5U2hcPB9, Profile: release)

Available backends:

- alsa (default)

- pipe

- subprocess

## Remove librespot files

cargo clean is the primary way to remove files generated by Cargo. It removes the target directory and its contents.

root~/.cargo/librespot# cargo clean -- use -n for dry run first

Removed 9659 files, 3.7GiB total

The target directory in /librespot is where Cargo stores compiled artifacts, including executables, object files, and other intermediate files. Deleting this directory forces Cargo to rebuild the project from scratch on the next build.

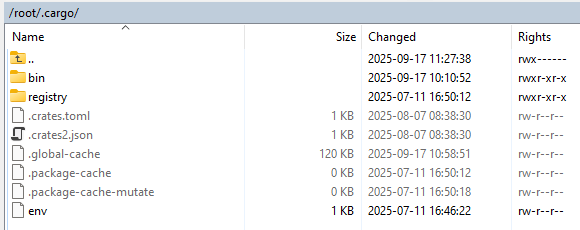
Then delete the /librespot directory - otherwise github will complain.

To uninstall a package in Rust installed with cargo install, use the cargo uninstall. Works only if the bin file is in the /.cargo/bin directory.

cargo uninstall librespot

Removing /root/.cargo/bin/librespot

The **WinSCP** view of /root/.cargo/ should now look like this:



## Librespot basic test

Simplest:

/bin/librespot –name Streamer

Verbose one:

/bin/librespot -v -n Streamer --initial-volume 35

Systemctl:

/bin/librespot -n Streamer --initial-volume 35 --disable-audio-cache --enable-volume-normalisation --onevent=/streamer/spotifyevent.sh

## 2. CRATE: Compile librespot from Crate with cargo

**Warning: do not use since librespot holds on to the alsa forever.**

Compiling with create is easier - no forks and no .ssh keys, but it is compiled with Rodio as backend - that does not work in August 2025 because librespot does not close the alsa as long as it is running.

<https://crates.io/crates/librespot>

sudo apt install cmake -- dependency

sudo apt install clang -- dependency

sudo cargo install librespot -- takes a long time…

:…

Installing /root/.cargo/bin/librespot **-- it ended up here!**

Installed package `librespot v0.6.0` (executable `librespot`)

Timestamp of bin file: 2025-07-11 17:12:10

## Where did the essential CRATE files go?

Both of them exists in the /root/.cargo directory

Here is the bin file: … it has to be copied.

/root/.cargo/bin/librespot

Copy the binary file and also put in directory /bin.

Here is the systemd service file provided by rust:

/root/.cargo/registry/src/index.crates.io-1949cf8c6b5b557f/librespot-0.6.0/contrib/librespot.service

Save a copy in directory /usr/lib/systemd/system

### More about where things are placed

Rustup metadata and toolchains will be installed into the Rustup

home directory, located at: **/root/.rustup**

This can be modified with the RUSTUP\_HOME environment variable.

The Cargo home directory located at: **/root/.cargo**

This can be modified with the CARGO\_HOME environment variable.

The cargo, rustc, rustup and other commands will be added to

Cargo's bin directory, located at: **/root/.cargo/bin**

This path will then be added to your PATH environment variable by

modifying the profile files located at:

/root/.profile /root/.bashrc

-------------------

You can uninstall at any time with rustup self uninstall and

these changes will be reverted.

Check path as well:

echo $PATH

/root/.cargo/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin

*Here is now where to access the binary file librespot:   
/root/.cargo/librespot/target/release/librespot*

*Quick access command:*

*cd ./target/release/librespot*

Check version: *crate version*

librespot -V

librespot 0.6.0 VERGEN\_IDEMPOTENT\_OUTPUT (Built on 2025-07-11, Build ID: DTSg4t5r, Profile: release)

librespot 0.3.1 f4be9bb (Built on 2022-01-29, Build ID: xt908EGz, Profile: release)

### Maybe check with curl?

curl", "http://localhost:65444"

interval: 1800s # Run the health check every 30 minutes

timeout: 5s # Wait up to 5 seconds for a response

retries: 3 # Mark as unhealthy after 3 failed checks

start\_period: 10s # Allow 10 seconds for the service to initialize

### More about environment variables [no use of sudo here]

root~# STREAMER\_LIB=/streamer/lib -- creates a variable

root~# export STREAMER\_LIB -- defines the variable into system

root~# echo $STREAMER\_LIB

/streamer/lib

## Create the systemd librespot service file

Must be placed in: /usr/lib/systemd/system/librespot.service

[Unit]

Description=Librespot (an open source Spotify client)

Documentation=https://github.com/librespot-org/librespot

Documentation=https://github.com/librespot-org/librespot/wiki/Options

Wants=network.target sound.target

After=network.target sound.target

[Service]

#DynamicUser=yes -- **this creates chaos, cannot write events on file**

User=root -- **must be root**!

Group=audio -- add this too

SupplementaryGroups=audio -- what is this, remove?

Restart=always

RestartSec=10

ExecStart=/bin/librespot -n Streamer --initial-volume 35 --disable-audio-cache --enable-volume-normalisation --onevent=/streamer/spotifyevent.sh

~~ExecStart=/bin/librespot -n Streamer -b 320 --initial-volume 35 --volume-ctrl linear --disable-audio-cache --enable-volume-normalisation --backend alsa --device "default:CARD=IQaudIODAC" --onevent /streamer/spotifyevent.sh~~

[Install]

WantedBy=multi-user.target

Then do; copy paste into console:

sudo systemctl daemon-reload

sudo systemctl enable librespot.service

sudo systemctl start librespot.service

sudo systemctl status librespot.service

sudo systemctl is-active librespot.service -- inactive or active?

sudo systemctl stop librespot.service

sudo journalctl -u librespot -- if errors, check this out

### lsof is not needed anymore [Optional]

sudo apt install lsof

Check cpu load of lsof with, copy paste into console:

ps aux | grep "/usr/bin/lsof -POlLwnp" -- crude

ps -Ao user,comm,pid,pcpu,stat --sort=-pcpu | head -n 6 -- gives top 5

ps -eo pcpu,args --sort=-%cpu | head -- compact

watch "ps -e -o pid,stat,cmd,%cpu,%mem --sort=-%cpu | head -n 6" --ultimate 2 s

top -bn1 |sed -n '7,12'p --maybe?

Look at STAT, where: R running or runnable (on run queue)

S interruptible sleep (waiting for an event to complete)

## Detect librespot events

<https://github.com/librespot-org/librespot>

First create an empty file or copy the var/log/spotifystream.log the [file](file:///D:\10%20Player%20Code\Player%20Source\Player%20on%20PC\spotifystream.log) which will be used for detecting changes of the Spotify Connect. Note: set the right permission for this collecting file which should be 0777 - check. That goes for the vital script: /player/spotifyevent.sh as well - [here](file:///D:\10%20Player%20Code\Player%20Source\Player%20on%20PC\spotifyevent.sh).

sed -i -e 's/\r$//' *“the script”*

### How /player/spotifyevent.sh works

if [[ $PLAYER\_EVENT == "playing" ]]; then

echo "spot:start" > /var/log/streamsensor.log

sync -d /var/log/streamsensor.log

fi

##This is when the user connects Spotify to Streamer -nothing streamed yet.

#if [[ $PLAYER\_EVENT == "started" ]]; then

# echo "spot:start started" > /var/log/streamsensor.log

# sync -d /var/log/streamsensor.log

#fi

if [[ $PLAYER\_EVENT == "paused" ]]; then

echo "spot:stop" > /var/log/streamsensor.log

sync -d /var/log/streamsensor.log

fi

if [[ $PLAYER\_EVENT == "stopped" ]]; then

echo "spot:stop" > /var/log/streamsensor.log

sync -d /var/log/streamsensor.log

fi

Possible $PLAYER\_EVENT:

{"VOLUME": "22937", "PLAYER\_EVENT": "volume\_set"}

{"TRACK\_ID": "4QXRhWSVeU3Vmwtj38Curv", "**PLAYER\_EVENT": "started**"}

{"POSITION\_MS": "66687242", "**PLAYER\_EVENT": "playing**", "TRACK\_ID": "4QXRhW….

{"PLAYER\_EVENT": "changed", "TRACK\_ID": "3z4yuRyycQGV7bE3tDnDWh", "OLD\_TRA

{"PLAYER\_EVENT": "preloading", "TRACK\_ID": "3PJWrzeVy0zQIEN1WlyCxo"}

{"DURATION\_MS": "223266", "**PLAYER\_EVENT": "paused**", "TRACK\_ID": "3PJWr

{"**PLAYER\_EVENT": "stopped**", "TRACK\_ID": "3PJWrzeVy0zQIEN1WlyCxo"}

**playing** the track has started to play, play symbol shown [optional]

**paused** the track has stopped playing, pause symbol shown [optional]

changed new track

**stopped** leaving the Spotify connect device

**started** connected to the Spotify connect device

preload initiation of track

volume\_set set volume

*sink value: running or closed/temporarily\_closed, note this is a sink event*

Reference. source code for events

<https://github.com/librespot-org/librespot/blob/dev/src/player_event_handler.rs>

## Some journalctl for librespot

Journalctl commands:

journalctl -u librespot

journalctl | grep -i librespot

journalctl -u librespot -b

journalctl -xeu librespot.service

## Disable Ivp 6?

Might be an ivp 6 problem? Sort of,

[2022-01-30T17:31:44Z WARN libmdns::fsm] error sending packet Os { code: 99, kind: AddrNotAvailable, message: "Cannot assign requested address" }

New message after ipv6 is disables:

Failed to register IPv6 receiver: Os { code: 19, kind: Uncategorized, message: "No such device" } -- … we can live with that.

### i) cmdline.txt

In cmdline.txt add: ipv6.disable=1

shutdown now -h

### ii) /etc/sysctl.conf

To disable ipv6, you have to open /etc/sysctl.conf using any text editor and insert the following lines at the end:

net.ipv6.conf.all.disable\_ipv6 = 1

net.ipv6.conf.default.disable\_ipv6 = 1

net.ipv6.conf.lo.disable\_ipv6 = 1

…and reboot. If IPv6 is still not disabled, then the problem is that sysctl.conf is still not activated. To solve this, type the command:

sudo sysctl -p

### Install lsof - for analyzing open files per process

apt install lsof

Another option but for interactive commands, not related to lsof.

apt install expect

## librespot server error 2025-08-06

Root cause: bad CDN (Content Delivery Network) URLs and ContextError.

In the librespot log this shows up pretty early on:

…Unable to load audio item: Error { kind: Unavailable, error: StatusCode(500)

…couldn't load context info because: context is not available. type: Default

…Skipping to next track, unable to load track <SpotifyId("spotify: track:…

It is generally believed that it seems to correspond with the sunsetting of the Spotify's V2 Ads API: <https://developer.spotify.com/documentation/ads-api>.

A possible fix:

Add an entry to /etc/hosts file as  
0.0.0.0 apresolve.spotify.com  
…towards the end of the file.

Fix: (to be put somewhere):

spclient: Specify base url for metadata requests

Dockerfile:

# Install pre-requisites for a build

RUN apt-get update && apt-get install -y --no-install-recommends \

libasound2-dev \

libssl-dev \

pkg-config \

cmake \

clang \

llvm-dev \

libclang-dev \

build-essential \

git \

libavahi-compat-libdnssd-dev \

alsa-utils \

libasound2 \

&& rm -rf /var/lib/apt/lists/\*

# Create app directory

WORKDIR /app

# 2025-08-12 - latest commit

RUN git clone https://github.com/librespot-org/librespot.git .

# Build release binary

RUN cargo build --release --features "alsa-backend" && strip target/release/librespot

# Stage 2: Runtime

FROM debian:bookworm-slim

LABEL org.opencontainers.image.title="dockerized-librespot" \

RUN apt-get update && apt-get install -y --no-install-recommends \

libasound2 \

ca-certificates \

curl \

&& rm -rf /var/lib/apt/lists/\*

# Copy the built binary from builder

COPY --from=builder /app/target/release/librespot /usr/local/bin/librespot

# Configure the entry point

COPY entrypoint.sh /usr/local/bin/entrypoint.sh

RUN chmod +x /usr/local/bin/entrypoint.sh

# Create user and set permissions if you want

RUN useradd -m -s /bin/bash librespot && usermod -a -G audio librespot

USER librespot

# Variables default values

ENV DEVICE\_NAME="librespot (Docker)"

ENV BITRATE=320

ENTRYPOINT ["/usr/local/bin/entrypoint.sh"]

# Section: install Shairport-sync

Compile and then install librespot from source found in <https://github.com/mikebrady/shairport-sync>

## Shairpoint-syncVersion

Version: 4.3.7-4-g188aa527-AirPlay2-smi10-OpenSSL-Avahi-ALSA-soxr-sysconfdir:/etc [2025-07-12]

Legacy: Version: 3.3.9-OpenSSL-Avahi-ALSA-soxr-sysconfdir:/etc [2022-01-30]

## Do not install the shairport-sync package

~~apt install shairport-sync~~

This seems to be a really old package - Version: 3.3.8

## Airplay 2

Shairport Sync is an AirPlay 2 audio player Shairport Sync offers full audio synchronisation, a feature of AirPlay that previous implementations do not provide. Full audio synchronisation means that audio is played on the output device at exactly the time specified by the audio source.

References:

<https://github.com/mikebrady/shairport-sync> -- general information, *Building and Installing* section

<https://github.com/mikebrady/shairport-sync/blob/master/INSTALL.md> -- install, compile, build

<https://hedenberg.org/carsetup/> -- interesting example

Shairport Sync, if built for **AirPlay** **2** operation, is not compatible with Apple Music or iTunes on Windows 11. The apps are not compatible with the Airplay 2 version of AMPLibraryAgent.

## Prepare for installation with Airplay 2

sudo apt update

sudo apt upgrade

sudo apt install --no-install-recommends build-essential git autoconf automake libtool

sudo apt install --no-install-recommends libpopt-dev libconfig-dev libasound2-dev avahi-daemon libavahi-client-dev libssl-dev libsoxr-dev

sudo apt install --no-install-recommends libplist-dev libsodium-dev libavutil-dev libavcodec-dev libavformat-dev uuid-dev libgcrypt-dev xxd

apt install --no-install-recommends systemd-dev -- no does not exist = **okay**

## Compile and install NQPTP – Not Quite PTP

nqptp is a daemon that monitors timing data from PTP clocks it sees on ports 319 and 320. It maintains records for one clock, identified by its Clock ID. It is a companion application to Shairport Sync and provides timing information for AirPlay 2 operation.

sudo git clone https://github.com/mikebrady/nqptp.git

cd nqptp

sudo autoreconf -fi

sudo ./configure --with-systemd-startup

sudo make

sudo make install systemctl enable nqptp

sudo systemctl start nqptp

sudo systemctl start nqptp

sudo systemctl status nqptp

## Compile and install shairport-sync

sudo git clone https://github.com/mikebrady/shairport-sync.git

cd shairport-sync

autoreconf -fi

./configure --sysconfdir=/etc --with-alsa --with-soxr --with-avahi --with-ssl=openssl --with-systemd --with-airplay-2

make

make install

## The shairport-sync bin file can be found here

which shairport-sync

/usr/local/bin/shairport-sync

## Configuration file for Shairport-sync

Edit config file in /etc/shairport-sync.conf the file can also be found [here](file:///D:\10%20Player%20Code\Player%20Configs\shairport-sync.conf).

### 1. Configure by adding in the General Section :-

general = {

name = "%H" //"Streamer";

interpolation = "soxr"; //"soxr" only for fast processor

volume\_range\_db = 60 ;

//use only usable part of built-in audio card mixer's attenuation range

Mike Brady says: “A problem with the built-in DAC that it declares itself to have a very large mixer volume control range – all the way from -102.38dB up to +4dB, a range of 106.38 dB. In reality, only the top 60 dB of it is in any way usable. To help get the most from it, consider using the volume\_range\_db setting in the general group to instruct Shairport Sync to use the top of the DAC mixer's declared range. For example, if you set the volume\_range\_db figure to 60, the top 60 dB of the range will the used. With this setting on the Raspberry Pi, maximum volume will be +4dB and minimum volume will be -56dB, below which muting will occur.” <https://github.com/mikebrady/shairport-sync> [scroll down – way down to see this]

audio\_backend\_latency\_offset\_in\_seconds = **-0.75**;

// -0.23= old value This is added to the latency requested by the player to delay or advance //the output by a fixed amount.

//Use it, for example, to compensate for a fixed delay in the audio back end.

//I.e. if the output device, e.g. a soundbar, takes 100 ms to process audio, set //this to -0.1 to deliver the audio

//to the output device 100 ms early, allowing it time to process the audio and //output it perfectly in sync.

audio\_backend\_buffer\_desired\_length\_in\_seconds = **1.5**; // 1.5/0.2 before;

// If set too small, buffer underflow occurs on low-powered machines.

//Too long and the response time to volume changes becomes annoying.

//Default is 0.2 seconds in the alsa backend, 0.35 seconds in the pa backend and //1.0 seconds otherwise. **1.5 is required for Bluetooth Speaker**

};

### 2. Configure by adding event file paths in the General Section:-

sessioncontrol = {

run\_this\_before\_play\_begins = "/player/airplayon.sh";

run\_this\_after\_play\_ends = "/player/airplayoff.sh";

wait\_for\_completion = "yes"; //wait for events script to finish

allow\_session\_interruption = "no"; };

### 3. Configure by adding in the Alsa Section :-

alsa = {

output\_device = "default"; **-- only this line is needed, always use default audio**

~~output\_device = "hw:0"; -- //card is hardwired~~

~~mixer\_control\_name = "PCM"; -- //PCM is okay here~~  };

Moode’s version

audio\_backend\_latency\_offset\_in\_seconds = 0.0;

audio\_backend\_buffer\_desired\_length\_in\_seconds = 0.2;

active\_state\_timeout = 10.0;

allow\_session\_interruption = "no"; -- can this solve the conflict with librespot?

session\_timeout = 120;

wait\_for\_completion = "yes";

output\_rate = 44100;

output\_format = "S16";

disable\_standby\_mode = "auto";

### Or start with a minimized config file: (has to be completed with additional lines…)

general = {

name = "Streamer";

~~interpolation = "soxr";~~

~~password = "secret";~~

~~output\_backend = "alsa";~~ };

alsa = {

output\_device = "hw:0"; //card is hardwired!

~~mixer\_control\_name = "PCM";~~  };

## Detect Shairport-sync events

First create an empty file or copy the var/log/airplaystream.log the [file](file:///D:\10%20Player%20Code\Player%20Configs\airplaystream.log) which will be used for detecting changes of Airplay. The value inside the file should be false.

Note: set the right permission for this collecting file which should be 0777 - check. That goes for the vital scripts: /player/airplayon.sh and ./airplayoff.sh as well - can be found [here](file:///D:\10%20Player%20Code\Player%20Source\Player%20on%20PC\airplayon.sh) and [here](file:///D:\10%20Player%20Code\Player%20Source\Player%20on%20PC\airplayoff.sh). See also example: *(from* [*https://hedenberg.org/carsetup/*](https://hedenberg.org/carsetup/) *), interesting example!*

### Groups and the user shairport-sync

Add shairport-sync user to the group bluetooth:

sudo adduser shairport-sync bluetooth -- users should have been added too

sudo adduser shairport-sync audio -- not needed, already added

(shairport-sync does not have access to the bluetooth device unless added to the bluetooth group.)

## Start shairport-sync.service in systemd

There is usually no need to edit the shairpoint-sync.service file since shairport-sync is configured in the configure file on previous pages.

sed -i -e 's/\r$//' *“the script”*

Start shairport-sync and see if it is working:

systemctl start shairport-sync

systemctl enable shairport-sync

systemctl status shairport-sync

### …and reboot

reboot - in order for the onevent to be working; script and file, or…

shutdown now -h

## Use Cloudplayer for testing

Download **Cloudplayer** from Google Play Store. It can stream both Airplay 1 and Airplay 2.

<https://www.doubletwist.com/cloudplayer/>

That might be the reason why Cloudplayer shows two available Streamers?

## Some journalctl for shairport-sync

Journalctl commands:

journalctl -u shairport-sync

journalctl | grep -i shairport-sync

journalctl -u shairport-sync -b

journalctl -xeu shairport-sync.service

### OPTIONAL – how to remove existing shairport-sync and its scripts

Check to see if shairport-sync is already installed –use the command sudo which shairport-sync to find where it is located, if installed. If it is installed it should be deleted After deleting, check again in case further copies are installed elsewhere.

Will most likely be found in: /usr/local/bin/shairport-sync

Also remove the startup script files /lib/systemd/system/shairport-sync.service (exists) or /etc/systemd/system/shairport-sync.service or /etc/init.d/shairport-sync if they exist – new ones will be installed if necessary.

# Section: Wi-Fi and hotspot

The nice and robust wpa\_supplicant procedures are gone…

## Network Manager

Debian 12 Bookworm uses Network Manager (nm) for network configuration. nmtui is the raspi-config like user interface, nmcli is the commands.

### Initial check devices and network connections

sudo nmcli device status -- or just d or dev

DEVICE TYPE STATE CONNECTION

eth0 ethernet connected Wired connection 1

lo loopback connected (externally) lo

wlan0 wifi unavailable --

sudo nmcli connection show -- or just c or con

NAME UUID TYPE DEVICE

Wired connection 1 9c153676-82c8-390f-9178-43f75847d52c ethernet eth0

lo 648670d0-dc27-42e3-82d4-005a4f9f6d98 loopback lo

Wired Connection 1 - the default wired connection. It can be written it 'Wired Connection 1' (surrounded by quotes) or Wired\ Connection\ 1

## Configurations at directory /etc/NetworkManager/system-connections

nm maintains connection information in /etc/NetworkManager/system-connections/\*.nmconnection which are read when nm starts up. Network Manager configures the network accordingly. If edited a connection file the connection needs to bring the connection down and up or restart the system.

Example: /etc/NetworkManager/system-connections/BELL462.nmconnection

## The Wi-Fi needs to be activated

### Check and enable Wi-Fi radio

sudo rfkill -- looks good

ID TYPE DEVICE SOFT HARD

0 bluetooth hci0 unblocked unblocked

1 wlan phy0 unblocked unblocked

sudo nmcli radio wifi -- or just r

disabled

sudo nmcli radio wifi on

sudo nmcli radio wifi

enabled -- looks good

sudo nmcli d s -- no connection yet = Wi-Fi is disconnected

DEVICE TYPE STATE CONNECTION

eth0 ethernet connected Wired connection 1

lo loopback connected (externally) lo

wlan0 wifi **disconnected** --

## Connect to Wi-Fi

### Scan..,

sudo nmcli dev wifi list -- scan for Wi-Fi networks

IN-USE BSSID SSID MODE CHAN RATE SIGNAL BARS SECURITY

D8:44:89:5B:0B:6E BELL462 Infra 6 130 Mbit/s 100 ▂▄▆█ WPA2

DE:44:89:5B:0B:6E -- Infra 6 130 Mbit/s 100 ▂▄▆█ WPA2

D8:44:89:5B:0B:6D BELL462 Infra 48 270 Mbit/s 100 ▂▄▆█ WPA2

F0:A7:31:81:21:EB BELL462 Infra 6 130 Mbit/s 75 ▂▄▆\_ WPA2

12:A7:31:81:21:EB -- Infra 6 130 Mbit/s 70 ▂▄▆\_ WPA2

F0:A7:31:81:21:EA BELL462 Infra 48 270 Mbit/s 69 ▂▄▆\_ WPA2

24:2F:D0:2B:1B:55 BELL462 Infra 6 130 Mbit/s 65 ▂▄▆\_ WPA2

84:D8:1B:20:2F:B4 BELL462 Infra 6 195 Mbit/s 64 ▂▄▆\_ WPA2

24:2F:D0:2B:1B:54 BELL462 Infra 48 270 Mbit/s 34 ▂▄\_\_ WPA2

2A:2F:D0:2B:1B:54 -- Infra 48 270 Mbit/s 34 ▂▄\_\_ WPA2

### …connect

sudo nmcli dev wifi connect "BELL462" password "4DFD4EFC155C" ifname wlan1

Device 'wlan0' successfully activated with 'd8848859-d784-4885-96c7-2efc599d1115'.

The ifname option defines which wlan.

sudo nmcli device show

DEVICE TYPE STATE CONNECTION

eth0 ethernet connected Wired connection 1

lo loopback connected (externally) lo

wlan0 wifi **connected** **BELL462**

sudo nmcli connection show

NAME UUID TYPE DEVICE

Wired connection 1 9c153676-82c8-390f-9178-43f75847d52c ethernet eth0

BELL462 d8848859-d784-4885-96c7-2efc599d1115 wifi wlan0

lo 648670d0-dc27-42e3-82d4-005a4f9f6d98 loopback lo

sudo curl -Is http://www.google.com | head -n 1 -- to check internet access

HTTP/1.1 200 OK

### The connection is defined in the file:

/etc/NetworkManager/system-connections/BELL462.nmconnection

**Note**: the connection profile still exists after disconnect.

[connection]

id=BELL462

uuid=d8848859-d784-4885-96c7-2efc599d1115

type=wifi

interface-name=wlan0

[wifi]

mode=infrastructure

ssid=BELL462

[wifi-security]

auth-alg=open

key-mgmt=wpa-psk

psk=4DFD4EFC155C

[ipv4]

method=auto

[ipv6]

addr-gen-mode=default

method=auto

[proxy]

### Delete connection profile

nmcli connection delete BELL462 -- deletes a connection file, but UUID should be used

When a Wi-Fi connection is defined, nm sets it to be autoconnected when the network starts up. However, if there are two Wi-Fi connections, define which one will be autoconnected, and make the other connection(s) NOT autoconnect:

sudo nmcli c mod <no-auto-connection-name> autoconnect false

### Disconnect

By network interface (device)

sudo nmcli con delete BELL462 -- disconnects and deletes connection profile

sudo nmcli device disconnect wlan0 -- connection profile still exists

By SSID (connection name):

sudo nmcli connection down id BELL462 --use UUID

If there are multiple connections with the same SSID, use the UUID.

nmcli con delete BELL462 -- deletes a connection profile, use UUID

nmcli con up BELL462 -- activates a saved connection, use UUID

nmcli con down BELL462 -- deactivates a connection, use UUID

.

## Connect/disconnect sequences Wi-Fi network

sudo nmcli radio wifi on

sudo nmcli device wifi connect "BELL462" password "4DFD4EFC155C" ifname wlan0

sudo nmcli device disconnect wlan0

## Force the 2.4 GHz band

sudo nmcli connection modify BELL462 802-11-wireless.band bg

Connection name is BELL462 in this case. …but does it work? Use a 2.4 GHz antenna instead (see below).

## Change country code for wireless conf files

sudo iw reg get -- settings information

sudo iw reg set CA --set to Canada

## p2p-dev-wlan0?

This is the Wi-Fi Direct or peer-to-peer connections, which is not used by Streamer. It is persistent and seems difficult to remove. Let it be.

## ip link commands on wlan interface

sudo ip link show wlan0 -- show wlan interface

3: wlan0: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 1500 qdisc pfifo\_fast state **UP** mode DORMANT group default qlen 1000

link/ether 2c:cf:67:75:f6:a0 brd ff:ff:ff:ff:ff:ffip link set wlan0 down

sudo ip link set wlan0 up

sudo ip link set dev wlan0 {up|down}

## iw commands on wlan interfaces

sudo iw list -- of less interest

sudo iw dev wlan0 info -- channel

sudo iw dev wlan0 link -- signal

### More commands…

sudo lspci

0002:00:00.0 PCI bridge: Broadcom Inc. and subsidiaries BCM2712 PCIe Bridge

0002:01:00.0 Ethernet controller: Raspberry Pi Ltd RP1 PCIe 2.0 South Bridge

sudo iwconfig wlan0

sudo netstat -tulanp -- shows internet connections

## Journald for nm

sudo journalctl -b | grep NetworkManager

sudo journalctl -b | grep brcmfmac -- to see if power management is on!

Player kernel: brcmfmac: brcmf\_cfg80211\_set\_power\_mgmt: power save enabled

## Create a hotspot with nmcli

To create a hosted wireless network on the command line, run the following command

sudo nmcli device wifi hotspot ssid Streamer password **retrostreamer**

nm named the connection Hotspot-1

Set the right IP-address range. Otherwise, it will more random like 10.42.0.1

sudo nmcli con down Hotspot-1

sudo nmcli con modify Hotspot ipv4.addresses 10.0.0.1/24 ipv4.gateway 10.0.0.254

sudo nmcli con up Hotspot

sudo nmcli dev wifi show-password -- generates QNR-code too

Password: **retrostreamer**

## Enable and Disable hotspot with nmcli

To disable the hotspot network, run the following command:

sudo nmcli connection down Hotspot-1

sudo nmcli connection up Hotspot-1

## Hotspot connection file IP-address: 10.0.0.1

/etc/NetworkManager/system-connections/Hotspot-1-313554ed-8f82-4f08-aa14-f0ca9f91e354.nmconnection

[connection]

id=Hotspot-1

uuid=313554ed-8f82-4f08-aa14-f0ca9f91e354

type=wifi

autoconnect=false

interface-name=wlan0

timestamp=1760208265

[wifi]

mode=ap

ssid=Streamer

[wifi-security]

group=ccmp;

key-mgmt=wpa-psk

pairwise=ccmp;

proto=rsn;

psk=retrostreamer

[ipv4]

address1=10.0.0.1/24,10.0.0.254

method=shared

[ipv6]

addr-gen-mode=default

method=ignore

[proxy]

UUID is: 313554ed-8f82-4f08-aa14-f0ca9f91e354

## Make the hotspot persistent

To make a Raspberry Pi hotspot permanent, you need to configure it to start automatically on boot by setting a high connection priority and enabling autoconnect.

sudo nmcli connection modify <hotspot UUID> connection.autoconnect yes connection.autoconnect-priority 100

Which in October 2025 would be:

sudo nmcli connection modify 313554ed-8f82-4f08-aa14-f0ca9f91e354 connection.autoconnect yes connection.autoconnect-priority 100

## Install the USB Wi-Fi antenna

[Mediatek/Ralink MT7601U - with antenna and 150 Mb/s]

### Check if the driver module is in place:

modinfo mt7601u

dmesg | grep mt76 -- debugging

### Update and install firmware

apt update

apt install firmware\*ralink\* firmware\*misc\*

**Note**: it is believed that this has to be repeated after a kernel update

### Update Wi-Fi local chip software update

apt update

apt install firmware-brcm80211

## Attach the USB Wi-Fi antenna

Does it work? Let’s check out:

sudo nmcli dev status

wlan1 wifi disconnected --

sudo nmcli dev wifi connect "BELL462" password "4DFD4EFC155C" ifname wlan1

Device 'wlan1' successfully activated with '690601c7-c16c-472f-b166-703d6d40e7da'.

sudo iwconfig wlan1

wlan1 IEEE 802.11 ESSID:"BELL462"

Mode:Managed Frequency:2.437 GHz Access Point: D8:44:89:5B:0B:6E

Bit Rate=72.2 Mb/s Tx-Power=30 dBm

Retry short limit:7 RTS thr:off Fragment thr:off

Encryption key:off

Power Management:off

Link Quality=70/70 Signal level=-29 dBm

Rx invalid nwid:0 Rx invalid crypt:0 Rx invalid frag:0

Tx excessive retries:0 Invalid misc:33 Missed beacon:0

## More info about the Wi-Fi signal

sudo iwconfig wlan1 | grep "Link Quality"

Link Quality=70/70 Signal level=-35 dBm

sudo iwconfig wlan0 | grep -i --color quality --more fun

Link Quality=70/70 Signal level=-35 dBm -- dev rigg

Link Quality=48/70 Signal level=-62 dBm -- playertest

Measured in dBm, a signal of greater than -70 dBm is considered an excellent signal in all networks, e.g -35 dBm is really good and -62 dBm is okay. The closer dBm number is to 0, the stronger signal.

Levels:

**-30 dBm**: This is the maximum signal strength. If you have this measurement, you are likely standing right next to the access point.

**-50 dBm**: This is considered an excellent signal strength.

**-60 dBm**: This is a good signal strength.

**-67 dBm**: This is a reliable signal strength. This is the minimum for any online services that require a reliable connection and Wi-Fi signal strength.

**-70 dBm**: This is not a strong signal strength. You may be able to check your email.

**-80 dBm**: This is an unreliable signal strength. You may be able to connect to your network, but you will not support most online activity.

**-90 dBm**: This is a bad signal strength. You are not likely to connect to internet at this level.

Ideally, you are looking to fall in the **-60 dBm** **to -50 dBm** range (since **-30 dBm** is unlikely). A signal strength at **-67 dBm** will work for a while, but this is the minimum strength you will want before your connection starts to suffer.

sudo cat /proc/net/wireless -- add watch first for monitoring

Inter-| sta-| Quality | Discarded packets| Missed | WE

face | tus | link level noise | nwid crypt frag retry misc | beacon | 22

wlan1: 0000 70. -35. -256 0 0 0 0 1056 0

shutdown now -h

# Section: memory management

du -hs to see disk usage at present directory

df -Th –total display disk system space

fdisk -l shows disk partitions

free -hw shows RAM usage

## No more RAM swapping to disk

Initially it looks like this:

sudo free -hw

Swap: 511Mi 0B 511Mi

First do:

sudo systemctl disable dphys-swapfile -- makes change persistent over reboots

Manually turn off swap space:

sudo swapoff --all

Check:

sudo free -hw

Swap: 0B 0B 0B

The swap line should now contain 0’s as above.

### More on CLIs for memory

df -Th –total -- -T gives filesystem type

df -hT | grep ext4 | awk '{print " Name: " $1 "\nMount: " $7 "\n Type: " $2 "\nUsage: " $6 "\n Size: " $3 "\n Used: " $4 "\n Free: " $5}'

Name: /dev/sda2

Mount: /

Type: ext4

Usage: 3%

Size: 220G

Used: 5.3G

Free: 206G

mount | grep vfat | awk -F'[ ()]+' '{print " Name: " $1 "\n Mount: " $3 "\n Type: " $5 "\nOptions: " $6}'

Name: /dev/sda1

Mount: /boot/firmware

Type: vfat

Options: rw,relatime,fmask=0022,dmask=0022,codepage=437,iocharset=ascii,shortname=mixed,errors=remount-ro

## Changing commit times – No?

The commit time for a file system is the amount of time for which it caches data before it copies it all to storage. Increasing this time improves performance by batching up lots of writes, but can lead to data loss if there is a corruption event before the data is written. Reducing the commit time will mean less chance of a corruption event leading to data loss, although it does not prevent it completely. Default is five seconds.

sudo nano /etc/fstab

Add commit=<commit time in seconds> to the EXT4 entry for the root file system, an example of shortening the time:

PARTUUID=50a7611a-02 / ext4 defaults,noatime,commit=3 0 1

A longer commit time, like 900 seconds (15 minutes), reduces write cycles, potentially extending SSD life, but increases the risk of data loss in case of power off (which is normal) failure or system crash. Conversely, a shorter commit time like 3 seconds reduces the risk of data loss but increases write cycles. If Streamer is a read-heavy application - no need to change the commit time. If it's write-intensive, adjusting the commit time could be beneficial.

No action for now – maybe extend the commit time ?...

## Temporary filesystem - yes

When the temporary files are stored in RAM, it can really speed up operations that rely on temporary files; and most important it reduces disk wear. Even if an application does not actually write any data, in the background Linux will constantly be making writes to the SSD

storage, mostly writing logging information.

On Raspberry Pi OS, /var/log is one of the biggest culprits for SDD wear. To deal with this, do either:

* Move it entirely to RAM with a tmpfs, as described below.
* Use log2ram to log to RAM and occasionally flush to disk.

Examples:

tmpfs /tmp tmpfs defaults,noatime 0 0 -- basic

tmpfs /var/log tmpfs defaults,noatime,size=16m 0 0 -- limits to 16MB

## Set up tmpfs - for logs and temporary files

Add in /etc/fstab

tmpfs /var/log tmpfs defaults,noatime,mode=0777 0 0

tmpfs /tmp tmpfs defaults,noatime,mode=0777 0 0

tmpfs /var/tmp tmpfs defaults,noatime,nosuid,mode=0777 0 0

tmpfs /var/run tmpfs defaults,noatime,nosuid,mode=0755 0 0

The noatime and nosuid parameters are also recommended for security and performance, and mode= (along with gid=) matches the permissions and group of the original filesystem. The noatime option avoids SSD write every time the disk is read.

... continue with:

sudo mount -a -- mount all fs in fstab

sudo systemctl daemon-reload

Check with df:

sudo df -Th --total

Filesystem Type Size Used Avail Use% Mounted on

udev devtmpfs 955M 0 955M 0% /dev

tmpfs tmpfs 199M 6.5M 193M 4% /run

/dev/sda2 ext4 220G 5.3G 206G 3% /

tmpfs tmpfs 993M 4.0K 993M 1% /dev/shm

tmpfs tmpfs 5.0M 8.0K 5.0M 1% /run/lock

/dev/sda1 vfat 510M 81M 430M 16% /boot/firmware

**tmpfs tmpfs 199M 0 199M 0% /run/user/0** -- the new ones

**tmpfs tmpfs 993M 0 993M 0% /var/log**

**tmpfs tmpfs 993M 0 993M 0% /tmp**

**tmpfs tmpfs 993M 0 993M 0% /var/tmp**

Also keep an eye on /var/spool

This will give a total overview, check it out..,

sudo mount |column -t

### References for tmpfs

Volumio’s tmpfs entries:

tmpfs /var/log tmpfs size=20M,nodev,uid=1000,mode=0777,gid=4, 0 0

tmpfs /var/spool/cups tmpfs defaults,noatime,mode=0755 0 0

tmpfs /var/spool/cups/tmp tmpfs defaults,noatime,mode=0755 0 0

**tmpfs** /tmp tmpfs defaults,noatime,mode=0755 0 0

tmpfs /dev/shm tmpfs defaults,nosuid,noexec,nodev 0 0

tmpfs /var/log tmpfs defaults,noatime,mode=0777 0 0

Another set-up; <https://www.zdnet.com/article/raspberry-pi-extending-the-life-of-the-sd-card/>

tmpfs /tmp tmpfs defaults,noatime,nosuid,size=100m 0 0

tmpfs /var/tmp tmpfs defaults,noatime,nosuid,size=30m 0 0

tmpfs /var/log tmpfs defaults,noatime,nosuid,mode=0755,size=100m 0 0

tmpfs /var/run tmpfs defaults,noatime,nosuid,mode=0755,size=2m 0 0

tmpfs /var/spool/mqueue tmpfs defaults,noatime,nosuid,mode=0700,gid=12,size=30m 0 0

piCorePlayer is built on a very small linux distro which is only about 12 MB, known as piCore Linux. It boots very fast and it is running entirely in RAM, it doesn't write to the SD-card unless you want to save a new setting. Therefore, pCP is very robust and you can simply pull the power without any risk of corruption of your SD card.

## Read-only root file systems - No

The *root file system* (rootfs) is the file system on the disk partition on which the root directory is located, and it is the file system on which all the other file systems are mounted as the system is booted up. On the Raspberry Pi it is /, and by default it is located on the SD card as a fully read/write EXT4 partition. There is also a boot folder, which is mounted as /boot and is a read/write FAT partition. Making the rootfs *read ONLY* prevents any sort of write accesses to it, making it much more robust to corruption events.However, unless other actions are taken, this means nothing can write to the file system at all, so…

No action so far.

## Overlay file system - No

An overlay file system (overlayfs) combines two file systems, an upper file system and a lower file system. When a name exists in both file systems, the object in the upper file system is visible while the object in the lower file system is either hidden or, in the case of directories, merged with the upper object.

Not for now.

Find the largest files in /var/log via:

sudo du -hs /var/log/\* | sort -rh | head -n 5

# Section: basic testing

shutdown now -h -- start again gives a stable and well-known state

## IQaudio sound card and alsa

### Standard checks

aplay -l

aplay -L

alsamixer

amixer scontrols

. . .

Simple mixer control 'Digital',0 #use Digital

. . .

### Sound check

amixer -c 0 get Digital toggle --get volume

amixer -c 0 set Digital toggle -- mute and unmute

amixer -c 0 set Digital 25% -- -c is card, set volume

Play sound:

speaker-test -c2

aplay -q -c2 -Ddefault /player/startup.wav

## Bluetooth

### Standard checks

systemctl status blue\*

Or check with:

systemctl status bluetooth

journalctl -u bluetooth

journalctl | fgrep -i blue -- -i means ignore case

journalctl -u bluetooth -b

bluealsa-aplay -L -- when bt spkr is connected

bluealsa-aplay -l

amixer scontrols

Bluetooth connecting source and streaming works [2022-01-31].

## Librespot

### Standard checks

sudo systemctl daemon-reload

sudo systemctl enable librespot.service

sudo systemctl start librespot.service

sudo systemctl status librespot.service

sudo systemctl is-active librespot.service -- inactive or active?

sudo systemctl stop librespot.service

sudo journalctl -u librespot -- if errors, check this out

journalctl -u librespot

journalctl | fgrep -i libre -- -i means ignore case

journalctl -u liberspot -b

Check file /var/log/streamsensor.log

### Check without systemctl service and configuration file

Full

/bin/librespot -n Streamer -b 320 --initial-volume 35 --volume-ctrl **cubic** --disable-audio-cache --enable-volume-normalisation --onevent=/streamer/spotifyevent.sh

Test

/bin/librespot -n Streamer -b 320 --initial-volume 35 --volume-ctrl linear --disable-audio-cache --enable-volume-normalisation --onevent=/streamer/spotifyevent.sh

## Shairport-sync

### Standard checks

systemctl status shairport-sync

systemctl stop shairport-sync

systemctl daemon-reload

systemctl start shairport-sync

journalctl -u shairport-sync

journalctl | grep -i shairport -- -i means ignore case

journalctl -u shairport-sync -b

Check file /var/log/streamsensor.log -- works

From console

/usr/local/bin/shairport-sync --log-to-syslog

## Mount an USB stick

fdisk -l | fgrep sd\* -- or even sdb since it is USB boot

Disk /dev/sdb: 1.88 GiB, 2021654528 bytes, 3948544 sectors

/dev/sdb1 \* 63 3948543 3948481 1.9G 6 FAT16

ls -l /dev/disk/by-uuid/| fgrep sdb -- uuid is 44D7-2AD9

total 0

lrwxrwxrwx 1 root root 10 Jan 31 16:06 **44D7-2AD9** -> ../../sdb1

mount /dev/sdb1 /mnt/usb -o uid=pi,gid=pi

The options uid and gid sets the owner and group of the root of the filesystem on the USB.

umount /mnt/usb -- remove usb

## Wi-Fi and AP

sudo nmcli dev status

## Memory

du -hs to see disk usage at present directory

df - h display disk system space

df -Th --total

fdisk -l shows disk partitions

free -hw shows RAM usage

systemctl status dphys-swapfile

systemctl disable dphys-swapfile

swappoff --all

## Web server

1. Check open ports for active servers:   
   netstat -tlnp
2. Check with this:   
   wget --verbose localhost:nnnn   
   ,where nnnn is the port.
3. Checking ports in general:   
   netstat -an
4. Or more specific   
   sudo netstat -tnl | grep nnnn,  
   where nnnn is the port.

Check with web browser on PC: <ip address> and:

<http://player.local> (http://player.local has to be typed in the first time)   
and: player.local.

sed -i -e 's/\r$//' *“the script”*

Use script ./port for checking. Start the player with script ./player. Stop player with the script ./rad.

In case: remove Window CR from script files:

sed -i -e 's/\r$//' *<script>*

## Boot phase

Open file /var/log/daemon.log

## Disk volume check on OS SSD

sudo fsck /dev/sda2

#fsck /dev/sda2

fsck from util-linux 2.38.1

fsck.fat 4.2 (2021-01-31)

There are differences between boot sector and its backup.

This is mostly harmless. Differences: (offset:original/backup)

65:01/00

1) Copy original to backup

2) Copy backup to original

3) No action

[123?q]? 3

Dirty bit is set. Fs was not properly unmounted and some data may be corrupt.

1) Remove dirty bit

2) No action

[12?q]? 1

\*\*\* Filesystem was changed \*\*\*

The changes have not yet been written, you can still choose to leave the

filesystem unmodified:

1) Write changes

2) Leave filesystem unchanged

[12?q]? 1

/dev/sda1: 427 files, 41010/261115 clusters

## Throttled?

vcgencmd get\_throttled

throttled=0x50000 -- anything other than 0x0 then it is, or has been, throttled.

0x50000 means throttled has occurred since the last reboot. 0x5000 indicates that the system has detected conditions requiring a reduction in CPU frequency to prevent damage or maintain stability. This usually points to either thermal issues (overheating) or power supply issues (undervoltage). One can also use watch vcgencmd get\_throttled to continuously monitor the status.

### Reset

vcgencmd get\_throttled 0x05 --might not work

throttled=0x0

vcgencmd commands

commands="commands, set\_logging, bootloader\_config, bootloader\_version, cache\_flush, codec\_enabled, get\_mem, get\_rsts, measure\_clock, measure\_temp, measure\_volts, get\_hvs\_asserts, get\_config, get\_throttled, pmicrd, pmicwr, read\_ring\_osc, version, readmr, otp\_dump, pmic\_read\_adc, power\_monitor"

# Section: set up backend

## Create the directory structure

Create the main JavaScript file and name it index.js. Some like to call their main file server.js or app.js. The file name index.js matches the default value of the “*main*” field in the package.json file, which identifies the entry point of the application player.

An example of directory structure

node-starter (project folder)

index.js (JavaScript file)

package.json (JSON file)

public - (folder) -- Static web files, HTML, CSS, JS and fonts

routes - (folder) -- JS for certain routes

controllers - (folder) -- JS functions for flow, references in router

views - (folder) -- JS for dynamic content

models - (folder) -- JS for interfacing DB (data)

This is more frontend

├── index.html

├── js/

│ ├── main.js

│ ├── models/

│ ├── views/

│ ├── collections/

│ ├── templates/

│ └── libs/

│ ├── backbone/

│ ├── underscore/

│ └── ...

├── css/

└── ...

Streamer main folder structure, even if there are more folders there… One main difference is the use of socket.io which kind of replace routers and those things.

├── streamer/

│ ├── data/

│ ├── lib/

│ └── pages/

│ ├── css/

│ ├── webfonts/

│ ├── machine.js

Create a folder streamer and change the permissions to 0777. Make them in

mkdir /streamer

mkdir /streamer/audio

mkdir /streamer/build

mkdir /streamer/data

mkdir /streamer/lib

mkdir /streamer/pages

mkdir /streamer/pages/ccs

mkdir /streamer/pages/webfonts

## Look at /player directories from 2022

## Install nodejs and npm

### Get the architecture

sudo uname -m

aarch64 -- also known as ARM64

### Do not install the generic nodejs package

~~sudo apt update~~

~~sudo apt install nodejs~~

**In the future us the right directory:**

**root@raspberrypi:/streamer#**

~~sudo nodejs -v -- version 18.19.0~~ – old and not supported!

### Remark

**Everything ended up (by mistake) in the home directory of root!**

## Get the right version of nodejs

<https://nodesource.com/products/distributions>

Choose: [Linux distribution – distribution version – Nodejs version]

Debian Debian Bookworm 22

sudo apt-get install -y curl

curl -fsSL https://deb.nodesource.com/setup\_22.x | sudo -E bash –

2025-07-16 10:58:12 - Repository configured successfully.

2025-07-16 10:58:12 - To install Node.js, run: **apt-get install nodejs -y**

2025-07-16 10:58:12 - You can use N|solid Runtime as a node.js alternative

2025-07-16 10:58:12 - To install N|solid Runtime, run: apt-get install nsolid -y

sudo apt-get install -y nodejs

nodejs -v

v22.15.0

sudo npm -v

10.9.2

To update run: npm install -g npm@11.4.2

sudo which -a nodejs

/usr/bin/nodejs

/bin/nodejs

### Why node and nodejs?

node -v

v22.15.0 -- correctly installed

-bash: /usr/local/bin/node: cannot execute: required file not found **– indicates error**

While Node might sometimes be used as a shorthand or in command-line contexts (e.g., node your\_file.js), Node.js is the official and more complete name for the runtime environment. There is no functional difference between node and nodejs in the context of this software.

### Runtime version N|Solid or Node.js?

N|Solid Runtime is an enhanced, enterprise-grade version of the open-source Node.js runtime, specifically designed for mission-critical applications. While it is fully compatible with standard Node.js applications and can run them without code modifications, it offers significant additions in terms of monitoring, security, and performance diagnostics.

**Right directory: root/streamer#**

## Create package.json for npm in /streamer

Get the package. json dependencies right, be sure to be in the folder /player:

npm init

package name: (root) streamer

/root/package.json -- better if it was in directory streamer, ah well…

## Install Express to node.js

npm install express --save

added 67 packages, and audited 68 packages in 3s

<https://expressjs.com/>

For npm; use the --save option!

## Install socket.io to node.js

npm install socket.io --save

added 25 packages, and audited 93 packages in 2s

<https://socket.io/>

**Note**: remember to copy socket.io.js from folder /root/node\_modules/socket.io/client-dist/socket.io.js to folder /streamer/pages, otherwise socket errors.

## Install Notyf

npm install notyf --save

added 1 package, and audited 94 packages in 909ms

Find the notyf.min.js and notyf.min.css files in folder /root/node\_modules/notyf and copy them to /streamer/pages.

<https://www.npmjs.com/package/notyf>

### [optional] pidof - for getting process id in node.js

npm install pidof --save

<https://www.npmjs.com/package/pidof>

### [optional] Install album-art

npm install album-art --save -- does not work at the moment (March 2022)

<https://github.com/lacymorrow/album-art>

npm install -g album-art -- in order to get CLI

There were vulnerabilities and they were fixed with: [**This is questionable**]

npm audit fix --force -- might cause problems

Updating album-art to 1.0.4,which is a SemVer major change.

(If problems occur, maybe copy the old album-art from Player 3.221F)

### [optional] Install usb detection

Check these first:

apt install -y build-essential -- should be in place

apt install libudev-dev -- might load packages. This is a must !!!!

npm install usb-detection --save --with libudev-dev installed first!

<https://www.npmjs.com/package/@bithighlander/usb-detection>

## Popper

Note: popper.min.js must be included before bootstrap.js  in the streamerxxx.html.

## Bootstrap

The bootstrap version 4.3.1 is used. It is old, but works.

## jQuery

The jquery version is 3.3.1.

## Install node-pty (and strip-ansi)

npm install node-pty --save

added 2 packages, and audited 96 packages in 6s

<https://www.npmjs.com/package/node-pty>

npm install strip-ansi --save

added 2 packages, and audited 98 packages in 874ms

<https://www.npmjs.com/package/strip-ansi>

## Optional npm packages to consider

### Install node-watch

npm install node-watch --save

<https://www.npmjs.com/package/node-watch>

### Options; use nodejs Javascript fs-module functions:

fs.watchFile('/var/log/streamsensor.log', (curr, prev) => { … body });

or fs.watch('/var/log/streamsensor.log', (eventType) => { … body });

## npm packages versions

The versions can be found in /root/package-lock.json, as of 2025-07-16:

"express": "^5.1.0",

"node-pty": "^1.0.0",

"notyf": "^3.10.0",

"socket.io": "^4.8.1",

"strip-ansi": "^7.1.0"

As of 2022-03-01 for Player 4.016:

"dependencies": {

"album-art": "^1.0.4",

"express": "^4.17.2",

"node-pty": "^0.10.1",

"node-watch": "^0.7.3",

"notyf": "^3.10.0",

"socket.io": "^4.4.1",

"strip-ansi": "^7.0.1",

"usb-detection": "^4.13.0"

## npm

npm is the package manager for the Node JavaScript platform.

npm init <package-spec>

If the <package-spec> is omitted (by just calling npm init), it will fall back to legacy init behavior. Then npm will ask you a bunch of questions, and then write a package. json for you. It will attempt to make reasonable guesses based on existing fields, dependencies, and options selected. It is strictly additive, so it will keep any fields and values that were already set.

npm install [<package-spec> ...]

npm update [<pkg>...]

npm uninstall [<@scope>/]<pkg>...

npm outdated [<package-spec> ...]

# Section: Create a systemd service for player

Start script is /streamer/streamer.sh Script content:

#!/bin/bash

# wrong CR characters, remove with: sed -i -e 's/\r$//' player.sh

echo "Player service starting... calling node"

Start /usr/local/bin/node /player/machine.js #no ‘&’ at end here

------------------------------------------------------------------------

Stop /usr/bin/killall -9 node #this is /player/stopplayer.sh

Often times, the instructions say to put systemd unit file into the /lib/systemd/system directory but this is bad practice. Don’t do it. Files that come in packages downloaded from a distribution repository go into /usr/lib/systemd. Local modifications (e.g. Streamer) that are manually placed for software installations that are not in the form of an apt package go into /etc/systemd/system.

Reference: <https://www.thedigitalpictureframe.com/ultimate-guide-systemd-autostart-scripts-raspberry-pi/>

Add the service definitions, systemd unit file, below to /etc/systemd/system/player.service

[Unit]

Description=Player machine

After=multi-user.target

[Service]

ExecStart=/bin/bash /player/player.sh

ExecStop=/bin/bash /player/stopplayer.sh

User=root

Restart=on-failure

RestartSec=60

[Install]

WantedBy=multi-user.target

The permission on the unit file needs to be set to 644 :

sudo chmod 644 /etc/<file>.service

### Set up systemctl

Check that the service starts:

systemctl daemon-reload

systemctl start player.service

systemctl status player.service

When things go wrong: systemctl reset-failed player

Finally:

systemctl enable player.service

### When testing: do not forget to disable!

… otherwise player might be started twice.

systemctl disable player.service

### Check boot phase

Open file /var/log/daemon.log

# Section: pishrink minimized image DEPRECATED

## Make an image [optional]

Using a smaller USB memory stick with the candidate Player system, typically 8 GB, make an image.

1. Check that an empty wpa\_supplicant.conf is in the /boot/ directory. [Optional]
2. Create new image file, insert the USB in the PC, open the ***Win32 Image Writer*** and then in ***MS File Explorer*** go to the directory **D:/11 Player Prod/nnnnnn/x.xxx month year/** and there create a file in Notes (or similar) with the file name **Player xxxxN full rev Y.img** .

## Shrink the image[[2]](#footnote-2) - pishrink script

1. Reboot the Player **with a larger USB memory stick**, typically 16 GB, and then copy the 8GB image from the **D:/11 Player/Prod/xxx** to the directory /player/data.  
   [Otherwise the file system will not be big enough for the transfer and transformation.]
2. Expand the file system of the bigger USB  
   sudo systemctl enable raspi-config.service  
   After enabling run:

sudo raspi-config --expand-rootfs

sudo reboot

1. The script pishrink.sh has to have the right permissions 0777. Check and fix.
2. sudo /player/lib/pishrink.sh "/player/data/Player 3125H rev full.img" <copy-to-file>\* -- last file name is optional
3. Rename the shrunken file to **Player xxxxN min rev Y.img**
4. **Copy the file back into D:/11 Player Prod/nnnnnn/x.xxx month year/**

Sourced from: <https://github.com/Drewsif/PiShrink>

# Section: all about log files

## Reduce the size of logs

All logs should be in ram by now.

### How big are they?

sudo du -hs /var/log/\* | sort -rh | head -n 5

4.0K /var/log/wtmp --since boot

4.0K /var/log/lastlog

0 /var/log/README

0 /var/log/private

0 /var/log/btmp

journalctl --disk-usage – max is set to 10 MB.

Archived and active journals take up 3.3M in the file system.

Archived and active journals take up 2.4M in the file system. -- later on…

## Journald log files

By default, systemd-journald logs are stored in the /run/log/journal/ directory that exists in RAM, which is a volatile location that is cleared upon reboot, or in /var/log.

Check where the journal (log) is:

journalctl --verify

PASS: /run/log/journal/9ce8b8c108644e1ca9e62cc84f7a5dc1/system.journal

### Minimize archived logs

journalctl --vacuum-files=1 -- reduces number of archived files manually

Vacuuming done, freed 864.0K of archived journals from /run/log/journal/

journalctl --vacuum-size=10M -- set the size of the archive

journalctl --rotate -- rotates into archive

### Rotate configurations

/etc/systemd/journald.conf

All logs are in /var/log are tmpfs (only in RAM) but limit the amount of space journald tries to use as well. In the file /etc/systemd/journald.conf. Uncomment the SystemMaxUse=... set it to 10M.

## Use journald and configure journals

Do the settings below in /etc/systemd/journald.conf

Storage=auto -- which is default actually.

If "volatile", journal log data will be stored only in memory, i.e. below the /run/log/journal hierarchy.

If "persistent", data will be stored preferably on disk.

"auto" behaves like "persistent" if the /var/log/journal directory exists, and "volatile" otherwise (the existence of the directory controls the storage mode).

"none" turns off all storage

SystemMaxUse=10M

RuntimeMaxUse=10M

/etc/logrotate.conf

The file /etc/logrotate.conf contains some default settings and sets up rotation for a few logs that are not owned by any system packages change to:

daily

rotate 0

/etc/logrotate.d/

This directory is where any packages installed will place their logrotate configuration.

apt btmp dpkg ppp wtmp -- logrotate.conf for packages

All logrotate configurations for the packages above must be checked and set to new rotate values: daily, rotate, size 100k, compress

Any options not set in these configuration blocks will inherit the default values or those set in /etc/logrotate.conf. Any setting in a specific logrotate file will override logrotate’s default values.

## Turn off journald with the config file [optional]

Configure the journald services in the config [file](file:///D:\10%20Player%20Code\Player%20Configs\journald.conf): /etc/systemd/journald.conf

Turn off this journal thing:

Storage=auto

change to Storage=none

[This Stack Exchange post](https://raspberrypi.stackexchange.com/a/186?ref=dzombak.com) discuss disabling journaling for the Pi’s filesystem. Don’t do this. While this change might reduce SD card wear, it makes your Pi more likely to face filesystem corruption in the event of a crash or power outage.

Rrsyslog does not exist.

## 

## Turn on logs again - if needed [optional]

/etc/systemd/journald.conf Storage=none change to Storage=auto

Restart and reload systemd service.

## Check logs and journal with journalctl

The log for service xxxxxx is reached by entering:

journalctl -b xxxxx

journalctl -u xxxxx - u is unit long version

journalctl -fu xxxxx - f show actual real time, ctrl-<c> to stop

journalctl -f -o cat - check the journal

journalctl -n1000 - to check last 1000 lines

journalctl --list-boots - boot log data

Check where the journals are and how many:

journalctl --verify -- run/log/journal/ is the directory

### Reference: [cheat sheet and man page]

<https://gist.github.com/sergeyklay/f401dbc8286f732783e05072f03ecb61>

<https://man7.org/linux/man-pages/man1/journalctl.1.html>

## Check out daemon.log - when Player is running as a service

cat /var/log/daemon.log | grep bash -- player log messages

cat /var/log/daemon.log | grep systemd -- systemd log messages

Others: librespot, bluetoothd, bluealsa-aplay, avahi-daemon, . . .

### Legacy: journald configuration

1. Config values for the size of the journal if stored on disk:

Specifies the ***maximum disk space that can be used*** in volatile storage (within the /run filesystem).

RuntimeMaxUse=1%

Specifies the amount of space to be set aside for ***other uses*** when writing data to volatile storage (within the /run filesystem).

RuntimeKeepFree=99%

Specifies the amount of space that an ***individual journal file can take up*** in volatile storage (within the /run filesystem) ***before being rotated***.

RuntimeMaxFileSize=102400 -- 100Kb is enough…

RuntimeMaxFiles=10 -- max number of files, default was 100

2. No logging in syslog

ForwardToSyslog=no -- hell no

ForwardToWall=no -- no general messages to all users

## Clean out logs before customer image

Clearing log content in /var/log

sudo cat /dev/null > /var/log/xxxxxx

Clearing the content of the file /root/.bash\_history

sudo cat /dev/null > /root/.bash\_history

# Section: systemctl and systemd.service

Reference: <https://man7.org/linux/man-pages/man1/systemctl.1.html>

To stop a currently running service:

systemctl stop xxxxxx

Start a service that has stopped and executes instructions in the service’s unit file:

systemctl start xxxxxx

Restart, restarts an already running service:

systemctl restart xxxxxx

Reload the configuration file without restarting (e.g. mpd.config):

systemctl reload alternatively: systemctl reload-or-restart

Reload the services’ configuration files for all of systemctl: (e.g. mpd.service and others):

systemctl daemon-reload

systemctl enable

systemctl disable - for starting at boot time

Services on a nice tree format:

systemctl status xxxxxx.service -- .service is optional

systemctl list-unit-files -- to see status like static, enabled or disabled

systemctl list-units --all -- list all services loaded

systemctl list-units --all --state=inactive - the ones that are not running

systemctl list-units --type=service

systemctl is-failed xxxxxx -- if one has failed

systemctl cat xxxxxx - to look at a specific service config file

systemctl list-dependencies xxxxx

systemctl show XXXXX --shows all configurable data, incl PID and much more

# SSD memory read & write

1. Single-level cell **(SLC)** **NAND** offers the highest endurance, typically sustaining around 100,000 write cycles.
2. Multi-level cell **(MLC) NAND** has a lower endurance, with planar NAND generally lasting up to 10,000 write cycles and **3D NAND** up to 35,000.
3. Triple-level cell **(TLC) NAND** has even lower endurance, ranging from 300 to 3,000 write cycles depending on the type of NAND.
4. Quadruple-level cell **(QLC) NAND** has the lowest endurance, typically supporting 1,000 write cycles.

From: <https://www.reddit.com/r/buildapc/comments/18dzseu/is_lexar_brand_reliable/>

# Section: Add ons, additions and trimming

### bash shell aliases

Check out: <https://www.cyberciti.biz/tips/bash-aliases-mac-centos-linux-unix.html>

Example:

ls --color=auto

### Playing silent sounds

aplay -N /dev/zero

aplay -N -r 8000 -f S16\_LE /dev/zero

### In shells when no stdout

[linux command] > /dev/null 2>&1

Investigate another image

### ------------------ after scripts are installed, START HERE:

#### c. Mount external USB device that is not the OS USB SSD

1. Attach the external USB device to the other USB 3.0 port.
2. Always make the directory /media/usb-ssd = the mount point.  
   sudo mkdir /media/usb-source

Check that the USB is attached:

sudo lsblk

sda 8:0 0 223.6G 0 disk

├─sda1 8:1 0 512M 0 part /boot/firmware

└─sda2 8:2 0 223.1G 0 part /

sdb 8:16 1 14.8G 0 disk

├─sdb1 8:17 1 512M 0 part

└─**sdb2** 8:18 1 4.1G 0 part -- this is the Linux partition, filesys type ext4

1. Mount the SSD with type exfat. It must be mounted with   
   sudo mount -t ext4 /dev/**sdb2** /media/usb-source
2. sudo systemctl daemon-reload -- not really needed

Check if it got mounted:

sudo ls -a /media/usb-source

bin boot dev etc home lib lost+found media mnt opt proc root run sbin srv sys tmp usr var

sudo umount /media/usb-ssd

### image-mount.sh

sudo chmod +x image-mount -- make executable, and then add .sh extensions.

Find the name of the image file:

Goto cd /root/image-utils

sudo ./image-mount.sh <imagefile> mountpoint Linux

Usage

image-mount imagefile mountpoint [W95|Linux] where W95 mounts the BOOT partition and Linux mounts the ROOT partition. If neither is specified, Linux is assumed. image-mount mounts a standard 'raw' image file to allow it to be read or written as if it were a device.

Go to cd /.

sudo ls -a /media/usb-ssd/images

Rename the image file to the correct version

Go to cd /.   
sudo ls -a /media/usb-ssd/images  
sudo umount /media/usb-ssd

Detach external SSD with the new image and attach it to the Windows desktop.

Rename the image file if needed and copy to the Window disk

### Optimization - as Moode does it…

~~sudo dphys-swapfile swapoff~~ -- swap is off

~~sudo update-rc.d dphys-swapfile remove~~

~~sudo rm /var/swap~~

sudo systemctl disable cron.service -- a possibility!

~~sudo systemctl enable rpcbind~~ --converts rpc calls somehow…

sudo systemctl set-default multi-user.target -- already done

sudo apt-get -y purge triggerhappy -- a possibility!

### Upgrade packages Moode style:

sudo apt-get update --allow-releaseinfo-change

sudo apt-get -y upgrade

### Change country code for wireless conf files

Replace country CA with DE

sed -i 's/country\_code=CA/country\_code=DE/' <conf file>

## Which version?...

cat /proc/version

Linux version 6.12.34+rpt-rpi-v8 (serge@raspberrypi.com) (aarch64-linux-gnu-gcc-12 (Debian 12.2.0-14+deb12u1) 12.2.0, GNU ld (GNU Binutils for Debian) 2.40) #1 SMP PREEMPT Debian 1:6.12.34-1+rpt1~bookworm (2025-06-26)

sudo hostnamectl

Static hostname: Player

Icon name: computer

Machine ID: 45677fd33dca4b97838c0214e7692911

Boot ID: bc4e5b4dd1e8452c93f962e7aae22d6c

Operating System: Raspbian GNU/Linux 11 (bullseye)

Kernel: Linux 5.10.92-v7l+

Architecture: arm

sudo uname -a

Linux Streamer 6.12.34+rpt-rpi-v8 #1 SMP PREEMPT Debian 1:6.12.34-1+rpt1~bookworm (2025-06-26) aarch64 GNU/Linux

Of less importance

… the first comments

### Checking CPU load

top -n 1 | grep "%Cpu"

### Root priviligies

Log in as root:

~$ su

Password: fajskvam88\*\*!!]

Or sudo su

# Section: SSD memory read & write

1. Single-level cell **(SLC)** **NAND** offers the highest endurance, typically sustaining around 100,000 write cycles.
2. Multi-level cell **(MLC) NAND** has a lower endurance, with planar NAND generally lasting up to 10,000 write cycles and **3D NAND** up to 35,000.
3. Triple-level cell **(TLC) NAND** has even lower endurance, ranging from 300 to 3,000 write cycles depending on the type of NAND.
4. Quadruple-level cell **(QLC) NAND** has the lowest endurance, typically supporting 1,000 write cycles.

From: <https://www.reddit.com/r/buildapc/comments/18dzseu/is_lexar_brand_reliable/>

# Section: image logs

root~/image-utils#./image-backup.sh -n 2025-08-16, 5.120 image

Image file to create? /media/usb-ssd/images/streamer-5120.img

Initial image file ROOT filesystem size (MB) [7368]? 7468

Added space for incremental updates after shrinking (MB) [0]? 100

Create /media/usb-ssd/images/streamer-5120.img (y/n)? y

Starting full backup (for incremental backups, run: ./image-backup.sh /media/usb-ssd/images/streamer-5120.img)

e2fsck 1.47.0 (5-Feb-2023)

Pass 1: Checking inodes, blocks, and sizes

Pass 2: Checking directory structure

Pass 3: Checking directory connectivity

Pass 4: Checking reference counts

Pass 5: Checking group summary information

rootfs: 226163/478608 files (0.1% non-contiguous), 1626694/1911552 blocks

resize2fs 1.47.0 (5-Feb-2023)

Resizing the filesystem on /dev/loop1p2 to 1756041 (4k) blocks.

The filesystem on /dev/loop1p2 is now 1756041 (4k) blocks long.

resize2fs 1.47.0 (5-Feb-2023)

Resizing the filesystem on /dev/loop1p2 to 1753190 (4k) blocks.

The filesystem on /dev/loop1p2 is now 1753190 (4k) blocks long.

resize2fs 1.47.0 (5-Feb-2023)

Resizing the filesystem on /dev/loop1p2 to 1753185 (4k) blocks.

The filesystem on /dev/loop1p2 is now 1753185 (4k) blocks long.

e2fsck 1.47.0 (5-Feb-2023)

Pass 1: Checking inodes, blocks, and sizes

Pass 2: Checking directory structure

Pass 3: Checking directory connectivity

Pass 4: Checking reference counts

Pass 5: Checking group summary information

rootfs: 226163/438048 files (0.1% non-contiguous), 1624149/1753185 blocks

e2fsck 1.47.0 (5-Feb-2023)

Pass 1: Checking inodes, blocks, and sizes

Pass 2: Checking directory structure

Pass 3: Checking directory connectivity

Pass 4: Checking reference counts

Pass 5: Checking group summary information

rootfs: 226163/438048 files (0.1% non-contiguous), 1624149/1753185 blocks

resize2fs 1.47.0 (5-Feb-2023)

Resizing the filesystem on /dev/loop0p2 to 1778785 (4k) blocks.

The filesystem on /dev/loop0p2 is now 1778785 (4k) blocks long.

e2fsck 1.47.0 (5-Feb-2023)

Pass 1: Checking inodes, blocks, and sizes

Pass 2: Checking directory structure

Pass 3: Checking directory connectivity

Pass 4: Checking reference counts

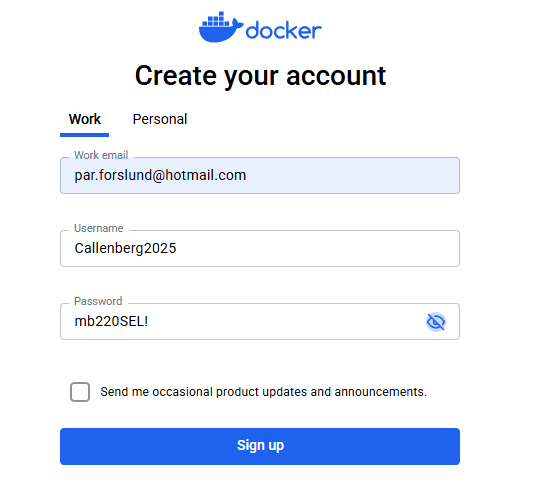
Pass 5: Checking group summary information

rootfs: 226163/446160 files (0.1% non-contiguous), 1624658/1778785 blocks

**touch: failed to get attributes of '': No such file or directory**

# Section: Docker

Account opened 2025-08-07 when librespot was broken (error 500).



Built with alsa:

<https://hub.docker.com/r/pienczyk/dockerized-librespot>

1. Transport Layer Security, use of the operating system's secure communication features for network connections. Like OpenSSL on Unix-like systems, SChannel on Windows, and Secure Transport on macOS. [↑](#footnote-ref-1)
2. PB8 shrunk from 7,864,300 to 3,371,631 kB. The file system seems to be of the size 2,454 MB. [↑](#footnote-ref-2)