

# Raspberry Pi Internet Radio

## Technical Reference Manual



A comprehensive guide to building Internet radios using the Raspberry Pi and MPD (Music Player Daemon)

**Bob Rathbone Computer Consultancy**

[www.bobrathbone.com](http://www.bobrathbone.com)

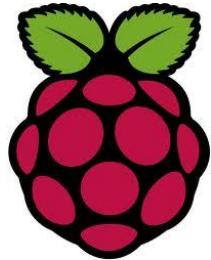
12th of February 2025

Version 8.0

## Contents

Chapter 1 - Introduction.....	3
Chapter 2 Installing radio components.....	4
Chapter 3 Manual installation and configuration .....	25
Chapter 4 - Internet Security .....	70
Chapter 5 - Frequently asked questions (FAQs).....	75
Chapter 6 Advanced topics.....	80
Chapter 7 – Configuration.....	125
Chapter 9 – Operation .....	148
Chapter 8 -Troubleshooting.....	193
Chapter 9 - Source files .....	251
Chapter 10 - Streaming to other devices using Icecast2.....	269
Acknowledgements .....	278
Glossary.....	289
Appendix A - System Files used by the Radio Program .....	294
Appendix B – Technical specification and other notes.....	304
Appendix C - SD Cards for Raspberry Pi .....	312
Appendix D – Explanation of Vcc, Vdd, Vss and Vee .....	313
Appendix E Raspberry Pi Model 1 wiring .....	314
Index .....	317

## Chapter 1 - Introduction



This manual is a supplement to the **Raspberry Pi Internet Radio Constructors Manual** which describes how to create one of the most popular Internet Radios using the Raspberry PI educational computer. This manual provides a supplemental information about the product particularly for those who are interested in maintaining the software.

The source and basic construction details are available from the following Web sites:

[https://bobrathbone.com/raspberrypi/pi\\_internet\\_radio.html](https://bobrathbone.com/raspberrypi/pi_internet_radio.html)

<https://github.com/bobrathbone/piradio6>

## Chapter 2 Installing radio components

### Contents chapter 2      Page


## Configure the SPI Kernel Module

Skip this section unless installing PiFace CAD or devices using the ST7789 OLED display.

If installing the radio on PiFace CAD or the ST7789 OLED display it is necessary to enable the SPI interface. For example, if PiFace CAD was selected the following message will be seen:

```
The chosen display PiFace CAD requires the
SPI kernel module to be loaded at boot time.
The program will call the raspi-config program
Select the following options on the next screens:
  5 Interfacing options
  P4 Enable/Disable automatic loading of SPI kernel module

Press enter to continue:
```

Press enter and select option 1:

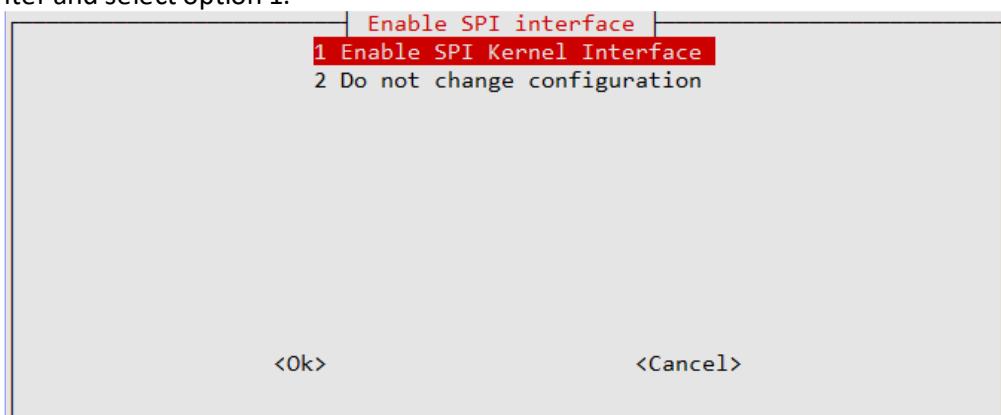


Figure 1 Enable SPI Kernel Module

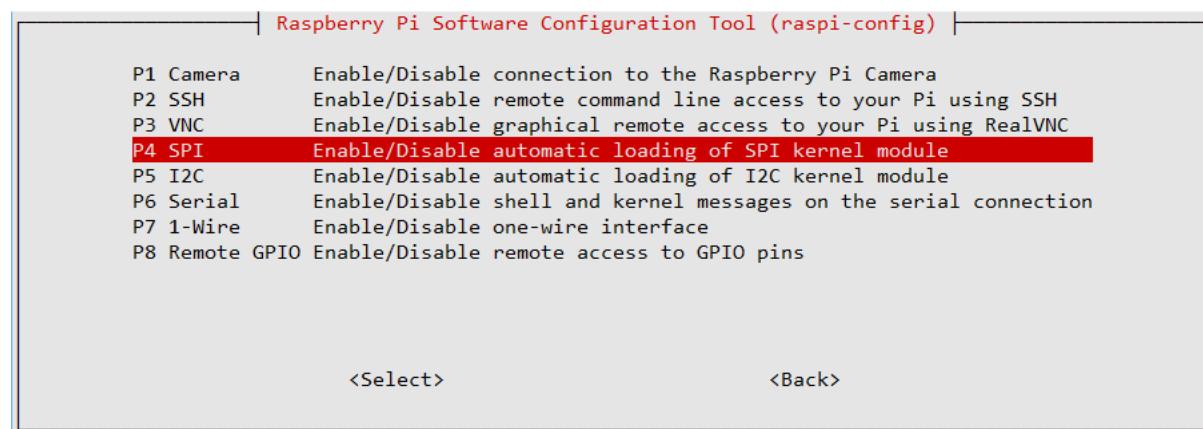


Figure 2 Enable SPI kernel module option

Select the option P4 SPI. The following screen is displayed:

```
Would you like the SPI interface to be enabled?
```

<Yes>

<No>

Select Yes to enable the SPI kernel module. Select “Finish” to exit.

```
Would you like the ARM I2C interface to be enabled?
```

<Yes>

<No>

## Connecting a Bluetooth audio device

### Install the Bluetooth software

Always update the system with the following commands even if you have done it before.

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

Install the necessary Bluetooth software by running the following:

```
$ sudo apt-get install pulseaudio pulseaudio-module-bluetooth
```

Add user pi (or the user you have configured) to the Bluetooth group:

```
$ sudo usermod -G bluetooth -a pi
```

## Install Blue Alsa utilities

```
$ sudo apt install bluez-alsa-utils
```

Do not install pipewire. In fact, remove it as it interferes with pulseaudio.

```
$ sudo apt remove pipewire
```

Reboot the Raspberry Pi.

```
$ sudo reboot
```

## Pairing a Bluetooth device

Switch on the Bluetooth speakers or headphones. Reboot the Raspberry Pi.

To pair your Bluetooth device run **bluetoothctl**. This will enter its own shell.

```
$ bluetoothctl  
Agent registered  
[bluetooth]#
```

Do not mistake the # prompt for the root (super-user) prompt. Put scanning on.

Make sure that the Bluetooth interface is powered on

```
[bluetooth]# power on
```

If you see the following message:

```
$ bluetoothctl  
[bluetooth]# scan on  
No default controller available
```

See *Bluetooth device connection problems* on page 237.

For the bookworm version of the OS display the Bluetooth address of the controller.

```
Agent registered  
[CHG] Controller B8:27:EB:A3:E4:52 Pairable: yes
```

Switch on the Bluetooth agent and set as default

```
[bluetooth]# agent on  
Agent registered  
[bluetooth]# default-agent  
Default agent request successful
```

Switch on scanning.

```
[bluetooth]# scan on  
Discovery started
```

```
[CHG] Controller DC:A6:32:05:36:9D Discovering: yes  
[NEW] Device C0:48:E6:73:3D:FA [TV] Samsung Q7 Series (65)  
[NEW] Device 00:75:58:41:B1:25 SP-AD70-B
```

When you see your Bluetooth speaker or headphones switch scan back off.

```
[bluetooth]# scan off  
:  
[CHG] Controller DC:A6:32:05:36:9D Discovering: no  
Discovery stopped
```

In this example the device name is **SP-AD70-B** and has a Bluetooth ID of **00:75:58:41:B1:25**.

Now pair the device using its ID:

```
[bluetooth]# pair 00:75:58:41:B1:25  
Attempting to pair with 00:75:58:41:B1:25  
[CHG] Device 00:75:58:41:B1:25 Connected: yes  
[CHG] Device 00:75:58:41:B1:25 UUIDs: 0000110b-0000-1000-8000-00805f9b34fb  
[CHG] Device 00:75:58:41:B1:25 UUIDs: 0000110e-0000-1000-8000-00805f9b34fb  
[CHG] Device 00:75:58:41:B1:25 UUIDs: 0000111e-0000-1000-8000-00805f9b34fb  
[CHG] Device 00:75:58:41:B1:25 ServicesResolved: yes  
[CHG] Device 00:75:58:41:B1:25 Paired: yes  
Pairing successful  
[CHG] Device 00:75:58:41:B1:25 ServicesResolved: no  
[CHG] Device 00:75:58:41:B1:25 Connected: no
```

Now connect and trust the device:

```
[bluetooth]# connect 00:75:58:41:B1:25  
Attempting to connect to 00:75:58:41:B1:25  
[CHG] Device 00:75:58:41:B1:25 Connected: yes  
Connection successful  
[CHG] Device 00:75:58:41:B1:25 ServicesResolved: yes
```

Trust the new Bluetooth device

```
[SP-AD70-B]# trust 00:75:58:41:B1:25  
[CHG] Device 00:75:58:41:B1:25 Trusted: yes  
Changing 00:75:58:41:B1:25 trust succeeded
```

Check the newly paired device

```
[SP-AD70-B]# info 00:75:58:41:B1:25  
Device 00:75:58:41:B1:25 (public)  
Name: SP-AD70-B  
Alias: SP-AD70-B  
Class: 0x00240404  
Icon: audio-headset  
Paired: yes  
Bonded: yes  
Trusted: yes  
Blocked: no  
Connected: yes  
LegacyPairing: no  
UUID: Audio Sink (0000110b-0000-1000-8000-00805f9b34fb)
```

```
UUID: A/V Remote Control      (0000110e-0000-1000-8000-  
00805f9b34fb)  
      UUID: Handsfree        (0000111e-0000-1000-8000-  
00805f9b34fb)  
[SP-AD70-B]#
```

Note that the Bluetooth prompt displays the name of the connected device. Now exit **bluetoothctl**.

```
[SP-AD70-B]# exit  
$
```

You can also use **bluetoothctl** with commands following it from the normal pi user prompt.

**For Bullseye:**

```
$ bluetoothctl paired-devices  
Device 00:75:58:41:B1:25 SP-AD70-B
```

**For bookworm run**

```
$ bluetoothctl devices Paired  
Device 00:75:58:41:B1:25 SP-AD70-B
```

The following displays all available commands:

```
$ bluetoothctl help
```

## Testing Bluetooth

Once Bluetooth is running switch on the Bluetooth Speaker or device. The Bluetooth device should indicate that it is connected to the Raspberry Pi by some means (See your device manual).

Display Bluetooth devices with bluealsa-aplay

```
$ bluealsa-aplay -l  
**** List of PLAYBACK Bluetooth Devices ****  
hci0: 00:75:58:41:B1:25 [SP-AD70-B], trusted audio-headset  
      A2DP (SBC) : S16_LE 2 channels 48000 Hz  
**** List of CAPTURE Bluetooth Devices ***
```

Note: Your display will vary depending upon the Bluetooth device just paired

Stop the radio software (Important).

```
$ sudo systemctl stop radiod
```

Now run the following:

```
$ aplay -D bluealsa:SRV=org.bluealsa,DEV=00:75:58:41:B1:25,PROFILE=a2dp  
/usr/share/sounds/alsa/Front_Center.wav
```

Note: The above is all one line. Substitute the above Bluetooth address with your own address. If things are working correctly, a spoken “Front Center” should be heard from the Bluetooth device. If not see *Bluetooth device no sound* on page 193 in Chapter 8 -Troubleshooting.

### Configuring the radio and MPD software to use Bluetooth

Now re-run the **configure\_audio.sh** configuration script and select Bluetooth.

```
$ cd /usr/share/radio/  
$ ./configure_audio.sh
```

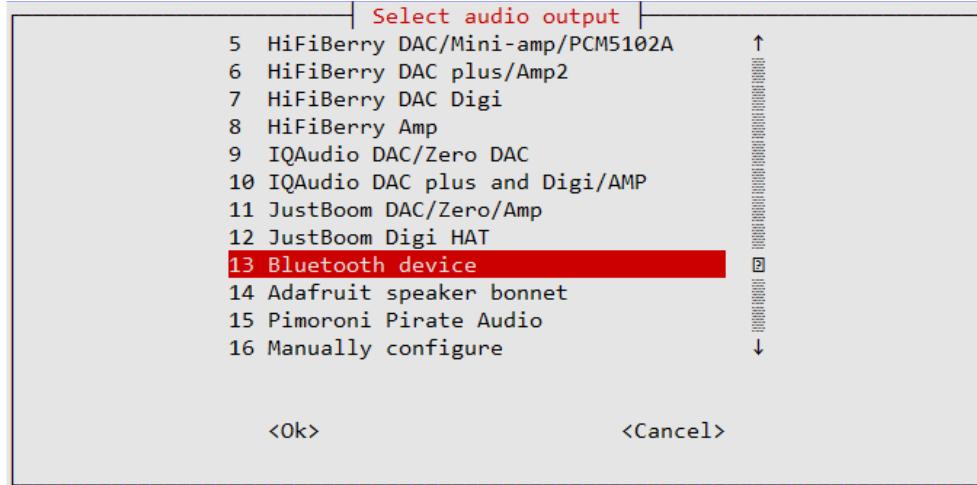


Figure 3 Configuring bluetooth devices

Reboot the Raspberry Pi when prompted.

```
$ sudo reboot
```

Run the following power on command:

```
$ bluetoothctl power on  
[CHG] Controller AA:AA:AA:AA:AA:AA Class: 0x006c0000  
Changing power on succeeded
```

Re-connect your device as shown in the following example.

```
$ bluetoothctl connect 00:75:58:41:B1:25  
Attempting to connect to 00:75:58:41:B1:25  
[CHG] Device 00:75:58:41:B1:25 Connected: yes  
[NEW] Endpoint /org/bluez/hci0/dev_00_75_58_41_B1_25/sep1  
[NEW] Transport /org/bluez/hci0/dev_00_75_58_41_B1_25/sep1/fd0  
Connection successful
```

### Using the alsamixer with Bluetooth devices

After rebooting run the alsamixer software and make sure that the volume is at full.

```
$ alsamixer -D bluealsa
```

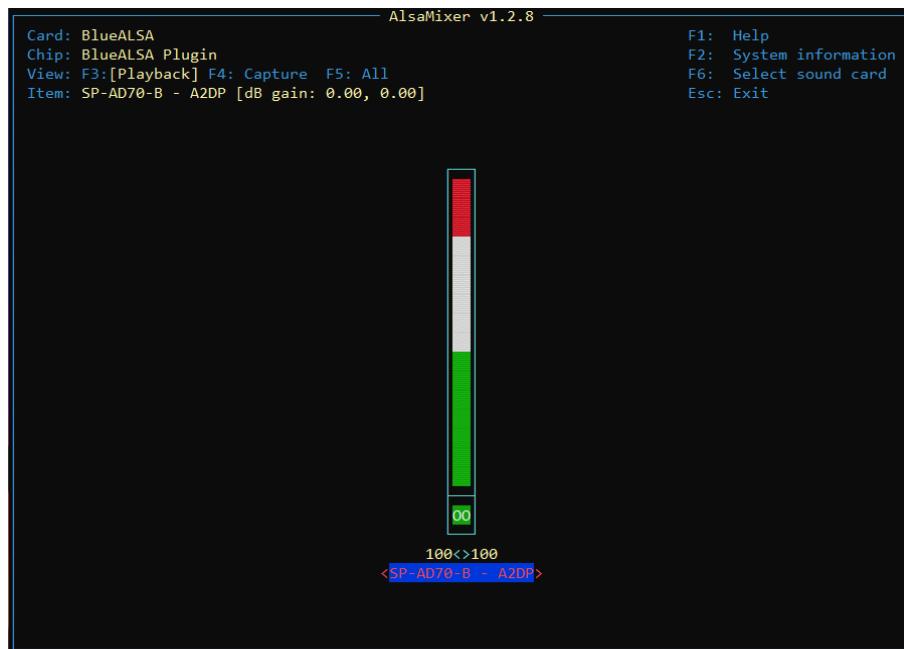


Figure 4 Alsamixer using Bluetooth devices

#### Retest the Bluetooth speakers

```
$ aplay -D bluealsa /usr/share/sounds/alsa/Front_Center.wav
```

Note that unlike the previous **aplay** instruction, there is no need to specify either the Bluetooth address or profile definition as these are all now specified in **/etc/asound.conf** which has set up by running the **configure\_audio.sh** script and selecting the Bluetooth option.

## Installing the IR remote control software

Before starting, the IR sensor needs to be wired to the correct GPIO pin. The following table shows the correct GPIO pin assignment for the IR receiver depending upon the hardware being used. Configuration commands shown later use the GPIO number shown in bold in the table below.

**Table 1** IR Sensor Pin outs

Radio Type	Pin	GPIO	Type of Raspberry PI
Two- or Four-line LCD with Push Buttons	21	<b>9</b>	Any (No DAC)
Two- or Four-line LCD with Rotary encoders	21	<b>9</b>	Any (No DAC)
Two- or Four-line LCD with I2C backpack	21	<b>9</b>	Any (No DAC)
Adafruit RGB plate with push buttons	36	<b>16</b>	40-pin version only
All versions using DAC sound cards	22	<b>25</b>	40-pin version only
IQaudio Cosmic Controller and OLED display	22	<b>25</b>	40-pin version only



Note: If you have wired your IR sensor to a different IR sensor other than shown in the above table, simply use any of the above GPIO numbers and then later amend the `dtoverlay= gpio-ir,gpio_pin=x` in `/boot/firmware/config.txt` where `x` is the GPIO number you have used.

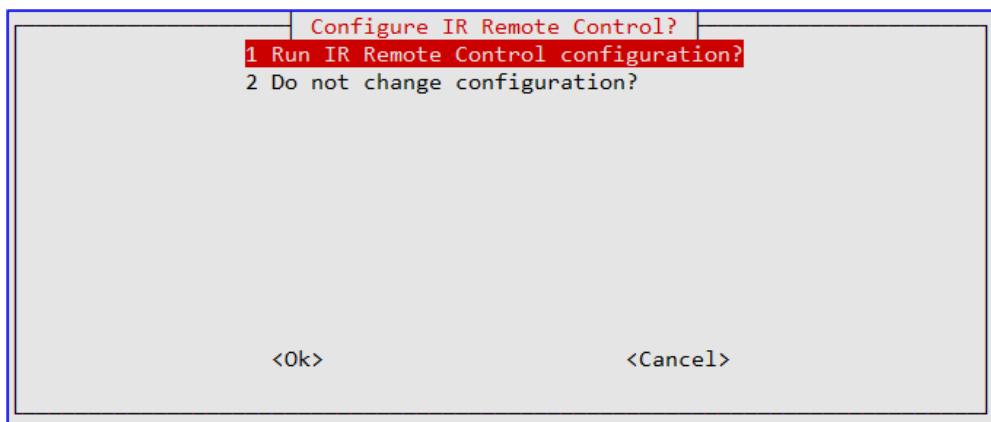
If you haven't already done so update the operating system first.

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

Run the IR remote installation program:

```
$ cd /usr/share/radio  
$ ./configure_ir_remote.sh
```

The following screen will be displayed. Select option 1 – Run IR Remote Control Configuration.



**Figure 5** IR Remote Installation program

It is now necessary to select which GPIO is to be used for the IR sensor. This is either 9, 16 or 25.

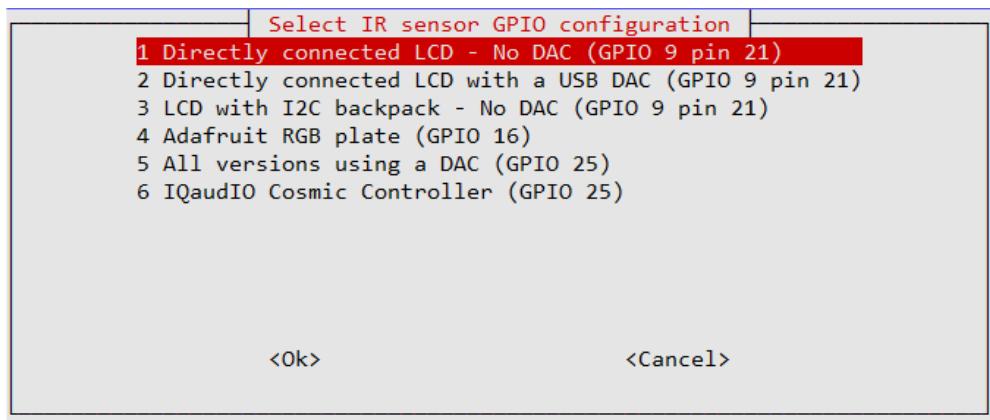


Figure 6 IR configuration IR sensor GPIO selection

Now select the Remote Activity LED GPIO. This is either GPIO 11, 13, 14 or 16.

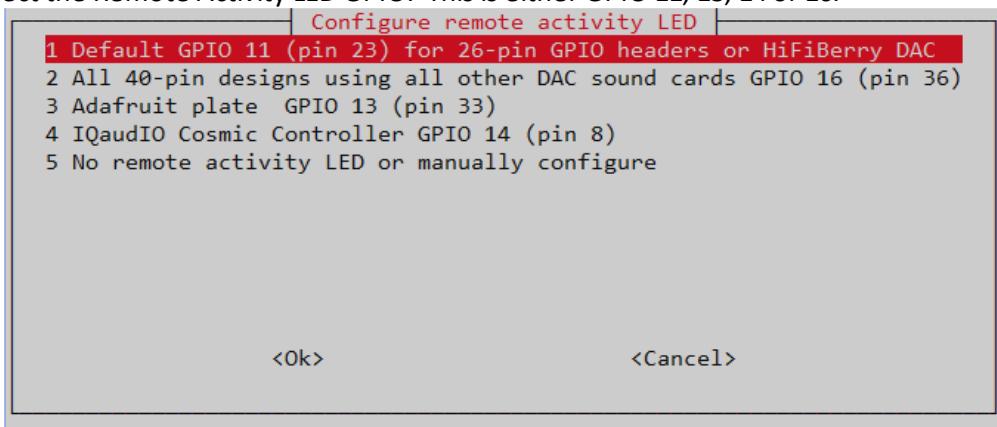


Figure 7 IR configuration Activity LED GPIO selection

Once the Activity LED selection has been made the program will install the Kernel Event components and configure the `/boot/firmware/config.txt` file. It installs `ir-keytable` and `python3-evdev` packages.

The program will display the following instructions to complete the set-up process. In the following it gives the example for the Kernel events service.

The following will be displayed:

```
./configure_ir_remote.sh configuration log, Fri 20 Dec 07:59:43 GMT 2024
Boot configuration in /boot/firmware/config.txt
Selected GPIO is 25
Remote activity LED is GPIO 16

Added following line to /boot/firmware/config.txt:
dtoverlay= gpio-ir,gpio_pin=25
Configured remote_led=16 in /etc/radiod.conf
Using kernel event device rc2
```

At the end the program will display the following instructions to complete the set-up process.

```
Configuration of Kernel Event completed OK
Reboot the system and then run the following
to configure your IR remote control
Create myremote.toml using the scan codes from the ir-keytable program
output
```

```
See the example in /usr/share/radio/myremote.mytoml  
Then copy your configuration file (myremote.toml) to /etc/rc_keymaps  
    sudo cp myremote.toml /etc/rc_keymaps/.
```

Reboot the Raspberry Pi

A log of this run will be found in /usr/share/radio/install\_ir.log

The program adds the **gpio-ir** dtoverlay to the **/boot/firmware/config.txt** file. The **gpio\_pin** varies with the selection you made. In the following example **GPIO25** is being used.

```
dtoverlay= gpio-ir, gpio_pin=25
```

Copy **rc6\_mce.toml** to **/etc/rc\_keymaps**

```
$ sudo cp /lib/udev/rc_keymaps/rc6_mce.toml /etc/rc_keymaps
```

Reboot the radio:

```
$ sudo reboot
```

Check that **gpio\_ir\_recv** module is loaded

```
$ lsmod | grep gpio_ir  
gpio_ir_recv           16384  0
```

After reboot check that the correct IR daemon is running. Use Ctr-C to exit

If the Kernel Events interface was configured check **ireventd**.

```
$ systemctl status ireventd.service  
● ireventd.service - Radio remote control daemon  
   Loaded: loaded (/lib/systemd/system/ireventd.service; enabled; vendor  
   preset: enabled)  
     Active: active (running) since Sat 2023-06-24 12:22:28 BST; 1h 51min  
       ago  
       Main PID: 3055 (python3)  
         Tasks: 1 (limit: 1599)  
        CPU: 336ms  
      CGroup: /system.slice/ireventd.service  
              └─3055 python3 /usr/share/radio/ireventd.py nodaemon  
:
```



Note that some remote such as the Samsung remote have a select button or buttons (for example VCR and TV) which change the protocol or coding or both of the IR signal transmitted. Make sure that you use the same mode when setting up the remote control and using it avoid confusion.



**Important:** A problem with the IR remote control software (No devices) has recently been reported by a number of constructors. This seems to be a problem with the latest version of Bookworm. See workaround below.

## Configuring the IR remote control.

The Radio software uses the **ir-keytable** program to handle IR remote control events. You should also have already installed the general IR software as shown in *Error! Reference source not found.* on page **Error! Bookmark not defined..**

The configuration files for **ir-keytable** have the **toml** extension. These can be listed with the following command:

```
$ ls /lib/udev/rc_keymaps/
adstech_dvb_t_pci.toml    encore_enltv.toml    pixelview_002t.toml
af9005.toml                evga_indtube.toml   pixelview_mk12.toml
:
```

This will list over **430** different remote controls and if you are lucky, you may find one that matches the one you are using. If you cannot find a definition for your remote control then it will be necessary to create your own **toml** definition file (myremote.toml). First run **ir-keytable**:

If you find a suitable keymap for your remote control copy it to **/etc/rc\_keymaps** directory.

```
$ cp /lib/udev/rc_keymaps/<your toml file> /etc/rc_keymaps/.
```

## Configure ir-keytable for Kernel Protocol

```
$ sudo cp /lib/udev/rc_keymaps/rc6_mce.toml /etc/rc_keymaps
```

## Creating an IR remote control definition

If you are using the Raspberry Pi Mini IR Remote Control you can skip this section and go to *Using the Raspberry Pi Mini IR Remote Control* on page 18, otherwise carry out the following instruction.

```
$ ir-keytable
```

This will list the available input devices of which there are several. These devices can change depending what drivers and in what order they are specified.

```
Found /sys/class/rc/rc1/ with:
  Name: vc4-hdmi-1
  Driver: cec
  Default keymap: rc-cec
  Input device: /dev/input/event1
  Supported kernel protocols: cec
  Enabled kernel protocols: cec
  bus: 30, vendor/product: 0000:0000, version: 0x0001
  Repeat delay = 0 ms, repeat period = 125 ms
Found /sys/class/rc/rc2/ with:
  Name: gpio_ir_recv
```

```

Driver: gpio_ir_recv
Default keymap: rc-rc6-mce
Input device: /dev/input/event2
LIRC device: /dev/lirc0
Attached BPF protocols: Operation not permitted
Supported kernel protocols: lirc rc-5 rc-5-sz jvc sony nec sanyo
mce_kbd rc-6 sharp xmp imon
Enabled kernel protocols: lirc nec
bus: 25, vendor/product: 0001:0001, version: 0x0100
Repeat delay = 500 ms, repeat period = 125 ms
Found /sys/class/rc/rc0/ with:
Name: vc4-hdmi-0
Driver: cec
:

```

The one that is of interest is **gpio\_ir\_recv** which uses **/sys/class/rc/rc2/**. Information only: the **ireventd.py program** locates the correct device (**rc2** in this case) for the **gpio\_ir\_recv** device driver when it is loaded.

Now run the following command specifying the correct device name with the **-s** flag:

```
$ sudo ir-keytable -s rc2 -t -p rc-5,rc-5-sz,jvc,sony,nec,sanyo,mce_kbd,rc-6,sharp,xmp
```

Note the above is all one line. This will display the protocols available.

```

Found device /sys/class/rc/rc0/
:
Opening /dev/input/event0
Input Protocol version: 0x00010001
Protocols changed to rc-5 rc-5-sz jvc sony nec sanyo mce_kbd rc-6 sharp xmp

```

Now press a button on the remote control such as Volume Up. This will display the **scancode** for that button. In this case it is the code for **KEY\_VOLUMEUP**.

```

228.396048: lirc protocol(necx): scancode = 0x833356
228.396069: event type EV_MSC(0x04): scancode = 0x833356
228.396069: event type EV_SYN(0x00).
228.456056: lirc protocol(necx): scancode = 0x833356 repeat
228.456073: event type EV_MSC(0x04): scancode = 0x833356
228.456073: event type EV_SYN(0x00).
228.564074: lirc protocol(necx): scancode = 0x833356 repeat
228.564098: event type EV_MSC(0x04): scancode = 0x833356
228.564098: event type EV_SYN(0x00).

```

Now create file called **myremote.toml** as shown in the example below. The scancode in the above example is for **KEY\_VOLUMEUP**.The name can be anything you wish, but it must end in the **toml** extension.

```

[[protocols]]
name = "myremote"
protocol = "nec"
variant = "nec"
[protocols.scancodes]
0x833356 = "KEY_VOLUMEUP"

```

```

0x833357 = "KEY_VOLUMEDOWN"
0x833395 = "KEY_MUTE"
0x833358 = "KEY_CHANNELUP"
0x833359 = "KEY CHANNELDOWN"
0x833396 = "KEY_MENU"
0x833342 = "KEY_UP"
0x833343 = "KEY_DOWN"
0x833344 = "KEY_LEFT"
0x833345 = "KEY_RIGHT"
0x833341 = "KEY_OK"
0x833360 = "KEY_NUMERIC_0"
0x833361 = "KEY_NUMERIC_1"
0x833362 = "KEY_NUMERIC_2"
0x833363 = "KEY_NUMERIC_3"
0x833364 = "KEY_NUMERIC_4"
0x833365 = "KEY_NUMERIC_5"
0x833366 = "KEY_NUMERIC_6"
0x833367 = "KEY_NUMERIC_7"
0x833368 = "KEY_NUMERIC_8"
0x833369 = "KEY_NUMERIC_9"
0x833373 = "KEY_EXIT"

```

The name field should be **myremote**. The protocol should be that shown above (**necx** in this example). Assign the scan code for each key as shown in the above event. The variant field is normally the same as the protocol field but this will be indicated in the event output if it is different. There is an example **myremote.toml** file in the **/usr/share/radio/remotes** directory.



Note: There wasn't a **necx** protocol. The nearest one was **nec** so this was used.



Note: The remote-control power on/off button must be called **KEY\_EXIT** and not **KEY\_POWER**. If **KEY\_POWER** is used the event system issues its own system shutdown command instead of letting the radio decide the exit action, either system shutdown or stop radio. See the **exit\_action** parameter in **/etc/radiod.conf** file.

Now write the new remote-control definition (myremote.toml) to the key table again specifying the correct device:

```

$ sudo ir-keytable -s rc2 -c -w myremote.toml
Read myremote table
Old keytable cleared
Wrote 11 keycode(s) to driver
Protocols changed to rc-5

```

Display the new table:

```

$ ir-keytable -r
scancode 0x833341 = KEY_OK (0x160)
scancode 0x833342 = KEY_UP (0x67)
scancode 0x833343 = KEY_DOWN (0x6c)
scancode 0x833344 = KEY_LEFT (0x69)
scancode 0x833345 = KEY_RIGHT (0x6a)
scancode 0x833356 = KEY_VOLUMEUP (0x73)
scancode 0x833357 = KEY_VOLUMEDOWN (0x72)
scancode 0x833358 = KEY_CHANNELUP (0x192)
scancode 0x833359 = KEY_CHANNELDOWN (0x193)
scancode 0x833360 = KEY_NUMERIC_0 (0x200)
scancode 0x833361 = KEY_NUMERIC_1 (0x201)
scancode 0x833362 = KEY_NUMERIC_2 (0x202)

```

```

scancode 0x833363 = KEY_NUMERIC_3 (0x203)
scancode 0x833364 = KEY_NUMERIC_4 (0x204)
scancode 0x833365 = KEY_NUMERIC_5 (0x205)
scancode 0x833366 = KEY_NUMERIC_6 (0x206)
scancode 0x833367 = KEY_NUMERIC_7 (0x207)
scancode 0x833368 = KEY_NUMERIC_8 (0x208)
scancode 0x833369 = KEY_NUMERIC_9 (0x209)
scancode 0x833373 = KEY_EXIT (0x74)
scancode 0x833395 = KEY_MUTE (0x71)
scancode 0x833396 = KEY_MENU (0x8b)
Enabled kernel protocols: lirc nec

```

Now test again with the correct -s <dev> flag :

```

$ ir-keytable -s rc2 -t
7427.808043: lirc protocol(necx): scancode = 0x833356
7427.808079: event type EV_MSC(0x04): scancode = 0x833356
7427.808079: event type EV_KEY(0x01) key_down: KEY_VOLUMEUP(0x0073)
7427.808079: event type EV_SYN(0x00).
7427.864040: lirc protocol(necx): scancode = 0x833356 repeat
7427.864075: event type EV_MSC(0x04): scancode = 0x833356
7427.864075: event type EV_SYN(0x00).

```



If you do not specify the correct device using the -s flag you will see the settings for the default protocol **rc6\_mce** and not the **myremote.toml** that you have just created.

If all is OK it is now necessary to make the changes permanent when the RPi is rebooted.

Copy the newly created **myremote.toml** to **/etc/rc\_keymaps/**

```
$ sudo cp myremote.toml /etc/rc_keymaps/myremote.toml
```

## Using the Raspberry Pi Mini IR Remote Control

If you are using the **Raspberry Pi Mini IR Remote Control** carry out the following instruction instead of the one above.

Copy the **mini.toml** definition to **/etc/rc\_keymaps/**

```
$ sudo cp /usr/share/radio/remotes/mini.toml /etc/rc_keymaps/myremote.toml
```

Reboot the Raspberry pi

```
$ sudo reboot
```

After reboot check that the IR key table is OK

```

$ ir-keytable -s rc2 -r
scancode 0x833341 = KEY_OK (0x160)
scancode 0x833342 = KEY_UP (0x67)
scancode 0x833343 = KEY_DOWN (0x6c)
scancode 0x833344 = KEY_LEFT (0x69)
scancode 0x833345 = KEY_RIGHT (0x6a)
scancode 0x833356 = KEY_VOLUMEUP (0x73)
scancode 0x833357 = KEY_VOLUMEDOWN (0x72)
scancode 0x833358 = KEY_CHANNELUP (0x192)

```

```
scancode 0x833359 = KEY_CHANNELDOWN (0x193)
scancode 0x833395 = KEY_MUTE (0x71)
scancode 0x833396 = KEY_MENU (0x8b)
: {Rest of output not shown}
```



Note: If you called your configuration anything else other than **myremote.toml** you not only need to change the previous instructions to use that name but also edit it in **/etc/radiod.conf** and change the **keytable** parameter to match the name you chose.

Edit **/etc/radiod.conf** and find the keytable entry and change the name to match the name you used.

```
# ireventd daemon keytable name
keytable=myremote.toml
```

Note: The **ireventd.service** (**ireventd.py**) reads this entry to load they key table at the start of its run. Now reboot the system and test the remote control:

```
$ reboot
```

## IR Remote Control Operation

The keys should work as expected, for example, Volume Up/Down, Mute and Channel Up/down. Pressing any of the numeric keys will cause the radio to play the station or track number entered. For example, press 5 and the 5<sup>th</sup> station in the playlist will play. Press 128 within two seconds and the radio will jump to the 128<sup>th</sup> station or track (if it exists). If it doesn't exist it will jump to station 1.

You can check the status of the **ireventd.service** using **systemctl**.

```
$ systemctl status ireventd.service
● ireventd.service - Radio remote control daemon
  Loaded: loaded (/lib/systemd/system/ireventd.service; enabled; vendor
  preset: enabled)
    Active: active (running) since Wed 2023-06-14 10:44:28 BST; 4h 43min
      ago
      Main PID: 541 (python3)
        Tasks: 1 (limit: 1599)
        CPU: 686ms
      CGroup: /system.slice/ireventd.service
              └─541 python3 /usr/share/radio/ireventd.py nodaemon

Jun 14 10:44:28 bobrath systemd[1]: Started Radio remote control daemon.
:
```

## Troubleshooting IR remote problems

See the section called *IR remote control problems* on page 231 for further information.

## Testing the ireventd daemon

If the **ireventd.service** isn't running then it is possible to test it as follows:

First stop the service.

```
$ sudo systemctl stop ireventd.service
```

Run the **ireventd.py** software from the command line. For example:

```
$ cd /usr/share/radio  
$ sudo ./irreventd.py
```

This will display the usage instructions:

```
This program must be run with sudo or root permissions!  
Usage: sudo ./irreventd.py start|stop|status|nodaemon|flash|config|send  
<KEY>
```

The following command flashes the activity LED six times if fitted.

```
$ sudo ./irreventd.py flash
```

If the service is crashing you can test it using the following command to run it as a foreground process:

```
$ sudo ./ireventd.py nodaemon
```

If you see the following:

```
pidfile /var/run/ireventd.pid already exist. Daemon already running?
```

Run

```
$ sudo ./ireventd.py stop
```

If the program is running normally, it displays the following (pid and GPIO number will vary):

```
IR Remote control listener running pid 1847  
Using IR events architecture  
Flashing LED on GPIO 16  
Read myremote table  
Old keytable cleared  
Wrote 22 keycode(s) to driver  
Protocols changed to nec  
:  
/dev/input/event0 gpio_ir_recv gpio_ir_recv/input0  
Listening for IR events:
```

Pressing a key on the remote-control will display the key that was pressed. The radio program will respond with 'OK'. For example, the following will be displayed if the Channel UP key is pressed.

```
KEY_CHANNELUP  
OK
```

It can also be used to send a valid test key or command to the radio program.

```
$ sudo ./ireventd.py send
Usage: ./ireventd.py send <KEY>
Where <KEY> is a valid IR_KEY
    KEY_VOLUMEUP,KEY_VOLUMEDOWN,KEY_CHANNELUP,KEY CHANNELDOWN,KEY_MENU
    KEY_UP,KEY_DOWN,KEY_LEFT,KEY_RIGHT,KEY_OK,KEY_INFO,KEY_MUTE,KEY_EXIT
    PLAY_<n> Where <n> is 1 to 999, play station/track n
```

For example, to play station 12.

```
$ sudo ./ireventd.py send PLAY_12
```

The status command:

```
$ sudo ./ireventd.py status
Remote control running pid 1975
```

Finally, the IR configuration can be displayed.

```
$ sudo ./ireventd.py config
Remote Control daemon configuration
-----
LED = GPIO 16
HOST = localhost
PORT = 5100
LISTEN = 5100
dtoverlay= gpio-ir, gpio_pin=25
```

Key map: /etc/rc\_keymaps/myremote

## Airplay Installation

If you have not already done so, carry out a system and firmware upgrade, as shown in **Error! Reference source not found.** on page **Error! Bookmark not defined.. Airplay** uses a program called **shairport-sync** from Mike Brady.

Airplay is based upon the procedure in the following link:

<http://www.redsilico.com/multiroom-audio-raspberry-pi>



Do not use the procedure in the above link. Use the procedure described below as it has been greatly modified to work with the radio software.

Airplay is installed using an installation script called **install\_airplay.sh**.

```
$ cd /usr/share/radio
$ ./install_airplay.sh
```

This will install both **shairport-sync** and configure the radio to use it.

If the script fails with a dependency error for PHP. Run the following:

```
$ sudo apt -f install
```

Re-run **install\_airplay.sh**.

## Configuring the Airplay feature

Below are the configuration parameters found in the Airplay section of **/etc/radiod.conf** affecting the Airplay (shairport-sync) function in the radio.

```
[AIRPLAY]
# Airplay activation yes or no
airplay=no

# Mixer preset volume for radio and media player if using sound card
# Set to 0 if using onboard audio or USB sound dongle.
# If using a sound card set to 100% initially and adjust as necessary
# Old name was mixer_volume
mixer_preset=100
```



If upgrading from an earlier version of the radio and you selected “Do not update existing configuration” during installation then the [AIRPLAY] section will be missing from **/etc/radiod.conf**. If this is the case then copy the above lines to the end of the file.

Check that Airplay is enabled. This should already be configured by **/etc/radiod.conf**

```
airplay=yes
```

## Airplay service check

Check that all is well with D-Bus. The following

```
$ sudo systemctl start shairport-sync
Failed to get D-Bus connection: Unknown error -1
```

If the above error message is seen install **systemd-sysv**:

```
$ sudo apt install systemd-sysv
```

Re-start and check status.

```
$ sudo systemctl start shairport-sync
$ sudo systemctl status shairport-sync
● shairport-sync.service - ShairportSync AirTunes receiver
   Loaded: loaded (/lib/systemd/system/shairport-sync.service; disabled;
   vendor preset: enabled)
     Active: active (running) since Sat 2021-03-20 11:25:15 GMT; 15s ago
```

Reboot the Raspberry Pi.

```
$ sudo reboot
```

See the following section on how to use Airplay.

## Using Airplay on the radio

Using Airplay on a HDMI/Touchscreen is described in the section called **Running Airplay on the HDMI touchscreen** on page 159. LCD versions of the radio are described here.



Figure 8 Airplay source selection

Press the menu button until **Input Source:** is displayed.

Turn the channel button (or Up/Down switches on a push-button radio) until **Airplay receiver** appears.

Press the menu button one more time. The word Airplay will be displayed on the bottom line along with 'Unknown artist' and 'Unknown title'.

Now use the Airplay device to connect to the raspberry PI (varies according to device software). Start playing the music tracks and this should start being heard on the radio which also displays the Artist, Track and Album on the LCD display. The volume is adjustable if correctly set-up. The mute also works in the normal way but does not pause or stop the Airplay stream as this can only be done from the device running Airplay.



Figure 9 Running an Airplay device on the radio with Cloudbreak

The above example is using an evaluation copy of CloudBreak running on an Android mobile telephone.

Unfortunately, Cloudbreak is no longer available however there are a number of Airplay Apps available for Android telephones such as “Double twist”.

## Chapter 3 Manual installation and configuration

<b>Contents chapter 4</b>	<b>Page</b>
	26
	26
The GPIOconverter program	64

## Radio software pre-requisites

### Upgrading from previous versions and other changes

If upgrading from version **6.x** or earlier it is necessary to replace your existing configuration file with the new one supplied in the upgrade. First take a note of any specific configuration items that are not standard or make a copy of your existing **/etc/radiod.conf** configuration file.



**Important:** The name format of Radio playlists has changed from version 7.3 onwards. See Creating and Maintaining Playlist files on page 169. Also, in this release, it is now possible to add, move and delete entries in the play list using any MPD client that is capable of doing so. See **Maintaining playlists using external MPD clients** on page 173.



**Note:** The location of the log file has changed from **/var/log/radio.log** to **/var/log/radiod/radio.log**. Also, its ownership has been changed from **root** to user **pi**. This has been done to reduce the number of programs that need to run with **sudo**.



**IMPORTANT:** In **Bookworm** the location for the boot configuration file **/boot/config.txt** has been changed to **/boot/firmware/config.txt**. Nearly all instructions in this manual refer to **/boot/firmware/config.txt** however if using **Bullseye** change the instruction to use **/boot/config.txt**. The installation scripts will choose the correct location.

### Installing the Music player daemon

If you haven't already done so upgrade the system packages as shown in **Error! Reference source not found.** on page **Error! Bookmark not defined..**

After reboot install the [Music Player Daemon](#) (mpd) and its client (mpc) along with the Python3, MPD and **python3-rpi.gpio** libraries. Note that you must install **python3-mpd** and not **python.mpd**.

```
$ sudo apt install mpd mpc python3-mpd python3-rpi.gpio
```

Answer yes 'y' when asked to continue.

If the installation says it cannot find the above packages, then re-run the update command to update the library list and retry installing the above packages.

```
$ sudo apt update
```



**IMPORTANT:** Unfortunately, the configuration file (**/etc/mpd.conf**) supplied with MPD version 0.22.6 in **Bookworm** and onwards is corrupt. It is necessary to correct this as shown below.

Find the following entries in **/etc/mpd.conf** somewhere around line 232.

#### Corrupt /etc/mpd.conf file

```
#audio_output {
#    type          "alsa"
#    name          "My ALSA device"
#    device        "hw:0,0"
#    mixer_type   "software"
##    mixer_device "default"      # optional
##    mixer_control "PCM"         # optional
##    mixer_index   "0"           # optional
```

```
# }
```

Change the above entries to the following:

#### Corrected /etc/mpd.conf file

```
audio_output {
    type      "alsa"
    name     "My ALSA Device"
    device   "hw:0,0"
    mixer_type "software"
    # mixer_device   "default"    # optional
    # mixer_control  "PCM"        # optional
    # mixer_index "0"          # optional
}
```



If installing on **Bookworm Lite** this will take quite a long time as there are a lot of software libraries to be installed.



At this stage there are no playlists configured so the music daemon won't play anything. The play lists are created when the **radiod** Debian Package is installed in the next section.



The version of **MPD** released with **Bookworm** is **0.22.6** which is reasonably up-to-date. You should install MPD version **0.22.6** first. When it is all working if you are having problems playing certain stations, it is well worth installing the latest version of MPD as shown in the section called *Compiling and installing the latest Music Player Daemon* on page 118.

#### Installing pulseaudio

The **pulseaudio** package may or may not needed to be installed. This has been different between various releases of the operating system. Currently the **pulseaudio** package causes a lot of problems with the radio software, in particular **espeak** and the Alsa sound system. If you do not intend to use espeak and need pulseaudio, for example, for blue-tooth devices or pHat BEAT then install it. The following table shows the options:

Table 2 PulseAudio installation options

Type of radio installation	Install pulseaudio
Using espeak	No
Pimoroni Pirate Radio with pHat BEAT	Yes (Installed by phatbeat)
Pimoroni Pirate Audio/mini-speaker	No
Adafruit speaker bonnet	No
LCD display radio	No unless using above DACs
HDMI or touch-screen displays	No unless using above DACs
IQaudIO, HiFiBerry or JustBoom DACs	No
Bluetooth sound devices	Yes

To install **pulseaudio**:

```
$ sudo apt install pulseaudio
```

### To remove **pulseaudio**:

```
$ sudo apt remove pulseaudio
```

If installing on Bookworm disable the `pulseaudio-enable-autospawn.service`.

```
$ sudo systemctl stop pulseaudio-enable-autospawn.service
```



At the time of writing there was a fault with the **lxplug-volumepulse** package which stops any of the menus on the Raspberry Pi desktop from being displayed. Remove the **lxplug-volumepulse** package with the following command:

```
$ sudo apt remove lxplug-volumepulse
```

## Installing OLED TFT driver software and libraries

If you are not using any OLED TFT display then skip this section.



**IMPORTANT NOTE:** The following devices and programs use **PIP3** to install the drivers.

- Olimex OLED or SSD1306 OLED TFTs
- Pimoroni Pirate Audio Radio
- SH1106 SPI OLED TFT
- Adafruit 2.5 and 3.5-inch TFT touchscreen
- Monochrome OLEDs which use the LUMA TFT driver
- The weather programs
- The IQaudIO controller

If you attempt to use **PIP3** (The name is a recursive acronym for "Pip Installs Packages") on the **Bookworm OS** it will fail with the following message:

```
To install Python packages system-wide, try apt install
python3-xyz, where xyz is the package you are trying to
install.
```

```
If you wish to install a non-Debian-packaged Python package,
create a virtual environment using python3 -m venv path/to/venv.
Then use path/to/venv/bin/python and path/to/venv/bin/pip. Make
sure you have python3-full installed.
```

```
For more information visit http://rptl.io/venv
```

To allow **pip3** to be used on **Bookworm OS** carry out the following instruction:

```
$ sudo mv /usr/lib/python3.11/EXTERNALLY-MANAGED
/usr/lib/python3.11/EXTERNALLY-MANAGED.old
```

Note that the above is all one line. There is no need to carry out the above instruction on system running **Bullseye OS**.

You will now be able to use **pip3** to install the driver software and programs shown in this manual, however you will see the following warning message:

WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package manager. It is recommended to use a virtual environment instead: <https://pip.pypa.io/warnings/venv>

The above message can be safely ignored for the installation procedures in this manual. However, if installing other author's software with **pip3**, you should seek advice from that author.

If using **Raspbian Lite** also install the **python3-pil** package:

```
$ sudo apt -y install python3-pil
```

### Install display drivers



Note: If you are installing the software for any of the products listed below then first carry out the instructions in the appropriate section. Do this before installing the radio software.

**Table 3 Display driver software installation**

Device	Section/Devices	Page
Pirate Radio	<b>Installing Pimoroni Pirate Radio (pHat BEAT)</b>	34
Pirate Audio	<b>Installing the Pimoroni Pirate Audio</b>	44
SSD1306 0.9-inch OLED	Installing the SSD1306 OLED driver libraries	49
SH1106 SPI (joystick)	Installing the SH1106 SPI OLED driver libraries	53
MHS RPi displays	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
SH1106 1.3-inch OLED	<b>Installing LUMA monochrome OLEDs</b>	59
LUMA OLED devices	SSD1306, SSD1309, SSD1325, SSD1331, SH1106, WS0010	59
Waveshare TFTs	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
Grove LCD RGB	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
Adafruit TFT	<b>Installing the Adafruit 2.5 and 3.5-inch TFT touchscreen</b>	54
PiFace CAD	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
Waveshare SH1106 SPI	Installing the SH1106 SPI OLED driver libraries	53

### Installing the Radio Daemon



**Note:** The `python-configparser` package has been dropped in **Bookworm** and appears to be satisfied with the package `python3-iniparse`. See:  
<https://packages.debian.org/bookworm/mips64el/python3-iniparse>

The latest version of the **radiod** (Version 7.9) setup.sh script has been amended to skip installation of **python-configparser** if using **Bookworm**.

If running **Bullseye** first install the **python-configparser** package. Skip this set if running **Bookworm** or later:

```
$ sudo apt install python-configparser
```

The Raspberry PI Internet Radio software is distributed as a Debian package. This can be downloaded from [http://www.bobrathbone.com/raspberrypi/pi\\_internet\\_radio.html](http://www.bobrathbone.com/raspberrypi/pi_internet_radio.html)  
Either download it to your PC or Macintosh and copy it to the **/home/pi** directory or get it directly using the **wget** facility.

To use the **wget** facility first copy the download link from the above page (Right click on the link). Log into the Raspberry PI. Now use **wget** to the software package.

#### For 32-bit systems

```
$ wget http://www.bobrathbone.com/raspberrypi/packages/radiod_7.9_armhf.deb
```

Run **dpkg** to install the package.

```
$ sudo dpkg -i radiod_7.9_armhf.deb
```

#### For 64-bit systems:

```
$ wget http://www.bobrathbone.com/raspberrypi/packages/radiod_7.9_arm64.deb
```

Run **dpkg** to install the package.

```
$ sudo dpkg -i radiod_7.9_armhf.deb
```

If upgrading or re-installing the radio software use the following to overwrite the existing package download file (Substitute correct file name):

```
$ curl -L -O http://bobrathbone.com/raspberrypi/packages/radiod_7.9_arm64.deb
```

The **dpkg** program will install the files.

```
(Reading database ... 131542 files and directories currently
installed. Preparing to unpack radiod_7.9_armhf.deb ...
Raspberry PI internet radio installation
Stopping radiod service
```

If updating from an earlier version of the radio software you may see the following message.  
Enter 'I' or 'Y' to install the new version.

```
Configuration file '/etc/logrotate.d/radiod'
==> Deleted (by you or by a script) since installation.
==> Package distributor has shipped an updated version.
What would you like to do about it ? Your options are:
  Y or I  : install the package maintainer's version
  N or O  : keep your currently-installed version
  D      : show the differences between the versions
  Z      : start a shell to examine the situation
The default action is to keep your current version.
*** radiod (Y/I/N/O/D/Z) [default=N] ? I
```

## Installing the speech facility (espeak)

It is possible to configure speech for visually impaired and blind persons who cannot read the display. As channels are changed or stepping through the menu the radio will “speak” to you. This excellent idea came from one of the project contributors, see *Chapter 11 - Setting up Spotify*

<b>Contents chapter 11</b>	<b>Page</b>
Setting up Spotify	279
Spotify hardware requirements	279
Spotify installation	279
Spotify operation	282
Troubleshooting Raspotify	284
Airplay Installation	21

## Setting up Spotify



The radio can also be set up as a Spotify receiver. You will still need a Spotify App on your telephone, PC or Tablet. You will also need a Premium Spotify account and not just a free or trial version.

More information at <https://www.spotify.com>

### Spotify hardware requirements

Spotify works with the on-board audio output or HDMI audio. However, if using a DAC, the card must be capable of hardware volume control. When the radio is running the volume is software controlled by MPD. When Spotify is running, MPD is stopped and the volume is controlled by the standard **amixer** command. The volume control for Spotify is coded in the **volume\_class.py** file. For example, the following command sets the volume to 100%.

```
sudo -u pi amixer cset numid=1 100%
```

The **numid** must match that of the hardware volume control. This is typically numid 1, 2 or 6. The **amixer controls** command will display all the hardware controls for a sound card. To check if your DAC supports Hardware Volume control run the **amixer controls** command.

```
$ amixer controls
:
numid=1,iface=MIXER,name='Digital Playback Volume'
```

DACs using the PCM5122a chip should all support hardware volume control. However, DACs based upon the PCM5102 chip do not have any hardware volume control so the volume cannot be set by **amixer** commands. It is still possible to use these DACs however volume can only be done from the Spotify client App running on a mobile telephone, tablet or PC.

### Spotify installation

The radio software also supports **Raspotify** originally from Dave Cooper which is Spotify for Raspberry Pi OS. First carry out a system update and upgrade as shown in **Error! Reference source not found.** on page **Error! Bookmark not defined..**

Install the **apt-transport-https** package and download and install the **Raspotify** software with the **curl** command into the pi home directory.

```
$ cd
$ sudo apt install apt-transport-https
$ curl -sL https://dtcooper.github.io/raspotify/install.sh | sh
```

This will both download and install **Raspotify**. In the **/etc/default/raspotify** configuration file you will see the **OPTIONS** line for Spotify account details. It is not necessary to configure your account details as these will be picked up from the connecting **Spotify** App on your PC or Mobile.

```
#OPTIONS="--username <USERNAME> --password <PASSWORD>"
```

More information on Raspotify can be found at <https://dtcooper.github.io/raspotify>

Using `sudo edit /lib/systemd/system/raspotify.service` and disable the restart options.

```
#Restart=always  
#RestartSec=10
```

Set the correct device ID to the `ExecStart` in the same `raspotify.service` file. This must match the `device` setting in the `audio_output` definition in `/etc/mpd.conf`

```
ExecStart=/usr/bin/librespot --device=hw:0,0
```

The configuration in `/etc/mpd.conf` is set up by the `configure_audio.sh` script

```
audio_output {  
    type          "alsa"  
    name          "IQAudio DAC/Zero DAC"  
    device        "hw:0,0"
```

Save the file and update `systemd` with the following command:

```
$ sudo systemctl daemon-reload
```

Finally run the `set_mixer_id.sh` script:

```
$ cd /usr/share/radio  
$ sudo ./set_mixer_id.sh  
mixer_volume_id=1
```

The above `mixer_volume_id` output may vary.

The radio will automatically stop and start Raspotify but it may be started and with the following commands:

```
$ sudo systemctl start raspotify  
$ sudo systemctl stop raspotify
```

You can also check the status of `raspotify` with `systemctl`.

```
$ sudo systemctl status raspotify  
● raspotify.service - Raspotify (Spotify Connect Client)  
   Loaded: loaded (/lib/systemd/system/raspotify.service; enabled; vendor  
   preset: enabled)  
     Active: active (running) since Fri 2022-02-18 10:12:25 GMT; 6min ago  
       Docs: https://github.com/dtcooper/raspotify  
              https://github.com/librespot-org/librespot  
              https://github.com/dtcooper/raspotify/wiki  
              https://github.com/librespot-org/librespot/wiki/Options  
     Main PID: 13033 (librespot)  
        Tasks: 1 (limit: 2054)  
         CPU: 123ms  
        CGroup: /system.slice/raspotify.service  
                  └─13033 /usr/bin/librespot
```

```
Feb 18 10:12:25 bookworm2 systemd[1]: Started Raspotify (Spotify Connect Client).
```

Disable Raspotify from starting at boot time. Starting and stopping Raspotify is done by the radio.

```
$ sudo systemctl disable raspotify
```

It is necessary to modify **/etc/raspotify/conf** to allow the track title to be displayed.

```
sudo vi /etc/raspotify/conf
```

Comment out the **LIBRESPOT\_QUIET** parameter in **/etc/raspotify/conf**. You will need to use your editor with **sudo**. For example, **sudo nano /etc/raspotify/conf**

```
# Only log warning and error messages.  
#LIBRESPOT_QUIET=
```

Failure to do so will mean that track titles will not be displayed.

Raspotify messages during operation can be observed with the following command:

```
$ journalctl --lines 0 --follow _SYSTEMD_UNIT=raspotify.service
```

Finally stop the Raspotify service and restart radiod.

```
$ sudo systemctl stop raspotify  
$ sudo systemctl restart radiod
```

**Note:** The authors of Raspotify have stated that it is not possible to display the artist's name and never will be. Only the track title can be displayed.

## Spotify operation

To start the Radio as a Spotify receiver either select Spotify from the Playlists (LCD or OLED versions) or from the Sources window in the touchscreen version:



Figure 64 Starting the Spotify Receiver

The following window will appear however a different message may appear on the second line of the display window:



Figure 65 The radio in Spotify mode

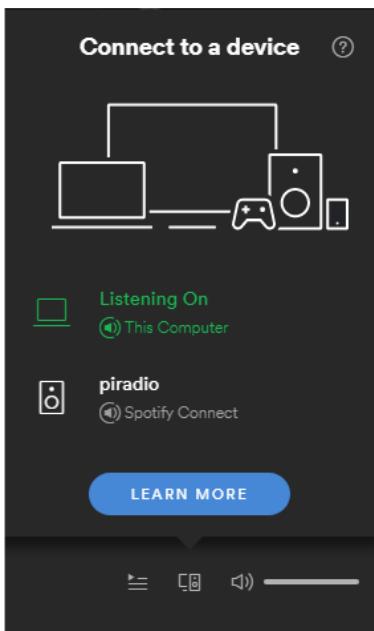


Figure 66 Spotify connecting to the radio

To use Spotify you will need a Spotify App on your telephone, PC or Tablet.

As previously mentioned you will need a Premium Spotify account and not just a free or trial version.

On the radio, press the Menu button until you come to the sources selection. Press channel Up or Down until you see Spotify displayed.

Press the Menu button again and the radio will stop the MPD player and start raspotify.

Now click on *Connect to a device* in the Spotify application. You should see an entry called Spotify Connect with the hostname of your radio.

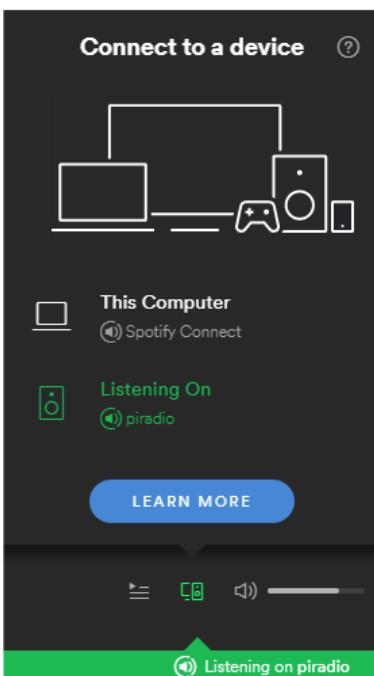


Figure 67 Listening to Spotify on the radio

Click on the *Spotify Connect* device for the radio (piradio in this example).

The Spotify application will switch from the current device (PC, mobile phone or tablet) to the Raspberry Pi radio.

You can control the volume either from the Spotify Application or using the volume control on the radio.

To exit Raspotify on the Raspberry Pi press the Menu button and then select another source such as the radio.

The Spotify application will connect the Raspotify application on the Radio.

**Note:** If you don't hear any sound then turn the volume up to full.

In the case of the touchscreen version of the program the following screen will be displayed:



Figure 68 Spotify playing a music track

In the case of the LCD or OLED version of the radio the title line above will be displayed on the second line of LCD or OLED screen.

**Note:** Raspotify unfortunately does not supply the Artist information. Only the track name is supplied by **librespot** which is used by Raspotify. There is currently no solution for this.

### Exiting Spotify

For the LCD and OLED versions press the Menu button until the Select source: window is displayed. Select any other source (playlist) to exit.

In the case of the touchscreen version of the radio, press the “Exit Spotify”button at the bottom of the screen.

### Troubleshooting Raspotify

#### Installation problems

If the curl command to install the Raspotify software fails then carry out a system update and upgrade as shown in **Error! Reference source not found.** on page **Error! Bookmark not defined..** Retry the curl command.

#### Raspotify exits with a 101 error code

This is almost certainly an authentication fault. Check your user name and password are correctly set up in **/etc/default/raspotify** and retry. However, it isn't actually necessary to put a username and password in **/etc/default/raspotify** as Raspotify will be using a Raspotify Premium account running on a PC, tablet or mobile phone.

#### The client connects to Raspotify but no sound heard

Check that the device ID has been added to the **ExecStart** statement in the same **raspotify.service** file.

```
ExecStart=/usr/bin/librespot . . . . . --device=hw:0,0
```

**Note:** The author does not directly support Raspotify.

Report Raspotify issues to <https://github.com/dtcooper/raspotify/issues>

Raspotify comes with an MIT licence. See <https://opensource.org/licenses/MIT>

### Cannot change Raspotify volume

In some cases, this is normal as many cheaper sound cards (DACs) do not have mixer controls. Mixer controls are required to change the volume on the Raspberry Pi when running Spotify or Airplay. In the case of no mixer controls the sound when running Spotify must be controlled from the App on either your mobile device or from the PC application.

However, it is possible to set up a software mixer (SoftVol).

### Raspotify initial sound too loud

Edit the **/lib/systemd/system/raspotify.service** file. Locate the following line

```
Environment="VOLUME_ARGS=--enable-volume-normalisation --volume-ctrl linear  
--initial-volume 100"
```

Change to the desired volume setting.

```
--initial-volume 75
```

Acknowledgements on page 278). This facility requires installation of the **espeak** package.  
See [https://elinux.org/RPi\\_Text\\_to\\_Speech\\_\(Speech\\_Synthesis\)](https://elinux.org/RPi_Text_to_Speech_(Speech_Synthesis))

The speech facility makes use of the **/var/lib/radio/language** file as already described in the section called **Creating a new language file** on page 134

Install the **espeak** package:

```
$ sudo apt install espeak
```

Enable the speech facility in **/etc/radiod.conf** and restart the radio.

```
# Speech for visually impaired or blind listeners, yes or no  
speech=yes
```

The verbose setting speaks the station or track details every time it is changed. However, it can take a long time to move through the tracks or stations whilst speaking. Usually set this to no.

```
verbose=no
```

To get the right balance between speech volume and the normal radio volume adjust the **speech\_volume** parameter percentage (10-100%)

```
speech_volume=75
```

Speak hostname and IP address when in the Information Menu.

```
speak_info=no
```

### The /var/lib/radiod/voice file

The **/var/lib/radiod/voice** file contains the **espeak** command (or part of it).

```
$ espeak -ven+f2 -k5 -s130 -a
```

Where **-v** is the voice (**en+f2** = English female voice 2), **-k** is capitals emphasis, **-s** is the voice speed and **-a** is amplitude (0-200), the **-a** parameter is filled in by the radio program.

### Testing espeak

You can test **espeak** with the following command (Stop the radio first).

```
$ espeak -ven+f2 -k5 -s130 -a20 "Hello Bob" --stdout | aplay
```

To see the capabilities of **espeak** see the Web site <http://espeak.sourceforge.net/> or run:

```
$ espeak -h
```

If no sound is heard then test using the **aplay** program. The **espeak** system will not work if **aplay** is not working. Test with **aplay** and a suitable wav file.

```
$ sudo mpc pause  
$ aplay /usr/share/sounds/alsa/Noise.wav
```

Or if Scratch is installed

```
$ aplay /usr/share/scratch/Media/Sounds/Vocals/Singer2.wav
```

If still no sound check what devices are configured using **aplay**.

```
# aplay -l  
**** List of PLAYBACK Hardware Devices ****  
card 0: ALSA [bcm2835 ALSA], device 0: bcm2835 ALSA [bcm2835 ALSA]  
Subdevices: 8/8  
Subdevice #0: subdevice #0  
Subdevice #1: subdevice #1  
Subdevice #2: subdevice #2  
Subdevice #3: subdevice #3  
Subdevice #4: subdevice #4  
Subdevice #5: subdevice #5  
Subdevice #6: subdevice #6  
Subdevice #7: subdevice #7  
card 0: ALSA [bcm2835 ALSA], device 1: bcm2835 ALSA [bcm2835 IEC958/HDMI]  
Subdevices: 1/1  
Subdevice #0: subdevice #0  
card 1: Device [Generic USB Audio Device], device 0: USB Audio [USB Audio]  
Subdevices: 0/1  
Subdevice #0: subdevice #0
```

In the above example there are two devices namely *bcm2835 ALSA* (normal audio jack output) and *Generic USB Audio Device*. If using either a HiFiBerry DAC or IQaudIO device then create the **/etc/asound.conf** file using nano:

```
$ sudo nano /etc/asound.conf
```

Add the following lines:

```
ctl.!default {
    type hw
    card 1
}

pcm.!default {
    type plug
    slave {
        pcm "plughw:0,0"
        format S32_LE
    }
}
```

The format S16\_LE is an alternative format but does not work with HiFiBerry DAC. The above statements set up the default mixer and PCM sound device respectively to use card 1.

If using the USB (Card 2 device 1) then change the device definition in the above file.

```
pcm "plughw:1,0"
```

Retest with **aplay** (No need to reboot).



Note: This author does not provide direct support for **espeak**.

See: <https://sourceforge.net/p/espeak/discussion/> for general support issues.

### Speech Operation

The espeak facility can be switched on and off with the remote control which has a button for toggling sound on and off and another button to speak information about the station or track as well as speaking the time. These buttons are set up in the section called **Error! Reference source not found.** on page **Error! Bookmark not defined..**

The Rotary encoder version of the radio is the best implemented. The MUTE switch is now the “Speak information” switch. To mute the radio, hold the button in for two seconds and release.

### Suppressing an individual message

It is possible to suppress speech of an individual message by adding an exclamation mark (!) to the beginning of the message string in the language file. For example if you do not wish to hear the time when speaking information then change **the\_time** parameter by adding an ! character to the beginning of the text to be spoken as shown in the example below:

```
the_time: !The time is
```

The exclamation message is removed if the message is displayed on a display. Only speech is affected.

## Installing Pimoroni Pirate Radio (pHat BEAT)

If not using the Pimoroni Pirate Radio then skip this section

To use the Pimoroni Pirate radio with **pHat BEAT** it is necessary to install the Pimoroni software pHat BEAT first. Do this before installing the Rathbone radio software.

Once the Pimoroni software is installed and tested it is then necessary to install the Rathbone radio software as shown in **Error! Reference source not found.** on page **Error! Bookmark not defined..**. Only the VU meter and pHat audio software is used by the Rathbone software. pHat uses the VLC radio not MPD.

The following instructions are based on the following link:

<https://github.com/pimoroni/phat-beat#full-install-recommended>

Run the following commands from the pi user home directory:

```
$ cd  
$ curl https://get.pimoroni.com/phatbeat | bash
```

This displays the following:

```
% Total     % Received % Xferd  Average Speed   Time     Time     Time  
Current                                         Dload  Upload   Total Spent  Left  
Speed  
100 35982  100 35982    0      0   113k      0 --::-- --::-- --::--  
114k  
  
This script will install everything needed to use  
pHAT Beat  
  
Always be careful when running scripts and commands copied  
from the internet. Ensure they are from a trusted source.  
  
This script should -- only be run on a Raspberry Pi with RPi OS --  
other systems and SBCs are not supported and may explode!  
  
If you want to see what this script does before running it,  
you should run: 'curl https://get.pimoroni.com/phatbeat'  
  
Note: pHAT Beat uses the I2S interface  
bash: line 494: [: !=: unary operator expected  
  
Do you wish to continue? [y/N] y
```

Answer yes (y).

The following is displayed:

```
pHAT Beat comes with examples and documentation that you may wish to  
install.  
Performing a full install will ensure those resources are installed,  
along with all required dependencies. It may however take a while!  
Do you wish to perform a full install? [y/N] y
```

Again, answer yes(y).

The installation will take quite some time as it does a system upgrade and then builds the software as well as installing any other required packages so be patient.

Eventually the installation program displays:

```
All done!
```

## Install the Rathbone Internet radio software

Do the following:

1. Now carry out the instructions shown in section *Radio* software pre-requisites
2. Upgrading from previous versions and other changes

If upgrading from version **6.x** or earlier it is necessary to replace your existing configuration file with the new one supplied in the upgrade. First take a note of any specific configuration items that are not standard or make a copy of your existing **/etc/radiod.conf** configuration file.

**Important:** The name format of Radio playlists has changed from version 7.3 onwards. See *Creating and Maintaining Playlist files* on page 169. **Also, in this release,** it is now possible to add, move and delete entries in the play list using any MPD client that is capable of doing so. See Maintaining playlists using external MPD clients **on page 173.**

**Note:** The location of the log file has changed from **/var/log/radio.log** to **/var/log/radiod/radio.log**. Also, its ownership has been changed from **root** to user **pi**. This has been done to reduce the number of programs that need to run with **sudo**.

**IMPORTANT:** In **Bookworm** the location for the boot configuration file **/boot/config.txt** has been changed to **/boot/firmware/config.txt**. Nearly all instructions in this manual refer to **/boot/firmware/config.txt** however if using **Bullseye** change the instruction to use **/boot/config.txt**. The installation scripts will choose the correct location.

## Installing the Music player daemon

If you haven't already done so upgrade the system packages as shown in **Error! Reference source not found.** on page **Error! Bookmark not defined..**

After reboot install the Music Player Daemon (mpd) and its client (mpc) along with the Python3, MPD and **python3-rpi.gpio** libraries. Note that you must install **python3-mpd** and not **python.mpd**.

```
$ sudo apt install mpd mpc python3-mpd python3-rpi.gpio
```

Answer yes 'y' when asked to continue.

If the installation says it cannot find the above packages, then re-run the update command to update the library list and retry installing the above packages.

```
$ sudo apt update
```

**IMPORTANT:** Unfortunately, the configuration file (**/etc/mpd.conf**) supplied with MPD version 0.22.6 in **Bookworm** and onwards is corrupt. It is necessary to correct this as show below.

Find the following entries in **/etc/mpd.conf** somewhere around line 232.

### Corrupt /etc/mpd.conf file

```
#audio_output {
#    type          "alsa"
#    name          "My ALSA device"
#    device        "hw:0,0"
#    mixer_type    "software"
##    mixer_device  "default"      # optional
##    mixer_control "PCM"         # optional
##    mixer_index   "0"           # optional
#}
```

Change the above entries to the following:

### Corrected /etc/mpd.conf file

```
audio_output {
    type          "alsa"
    name          "My ALSA Device"
    device        "hw:0,0"
    mixer_type    "software"
#    mixer_device  "default"      # optional
#    mixer_control "PCM"         # optional
#    mixer_index   "0"           # optional
}
```

If installing on **Bookworm Lite** this will take quite a long time as there are a lot of software libraries to be installed.

At this stage there are no playlists configured so the music daemon won't play anything. The play lists are created when the **radiod** Debian Package is installed in the next section.

The version of **MPD** released with **Bookworm** is **0.22.6** which is reasonably up-to-date. You should install MPD version **0.22.6** first. When it is all working if you are having problems playing certain stations, it is well worth installing the latest version of MPD as shown in the section called *Compiling and installing the latest Music Player Daemon* on page 118.

### Installing pulseaudio

The **pulseaudio** package may or may not needed to be installed. This has been different between various releases of the operating system. Currently the **pulseaudio** package causes a lot of problems with the radio software, in particular **espeak** and the Alsa sound system. If you do not intend to use espeak and need pulseaudio, for example, for blue-tooth devices or pHat BEAT then install it. The following table shows the options:

**Table 2 PulseAudio installation options**

Type of radio installation	Install pulseaudio
Using espeak	No
Pimoroni Pirate Radio with pHat BEAT	Yes (Installed by phatbeat)
Pimoroni Pirate Audio/mini-speaker	No
Adafruit speaker bonnet	No

<b>LCD display radio</b>	No unless using above DACs
<b>HDMI or touch-screen displays</b>	No unless using above DACs
<b>IQaudIO, HiFiBerry or JustBoom DACs</b>	No
<b>Bluetooth sound devices</b>	Yes

To install **pulseaudio**:

```
$ sudo apt install pulseaudio
```

To remove **pulseaudio**:

```
$ sudo apt remove pulseaudio
```

If installing on Bookworm disable the `pulseaudio-enable-autospawn.service`.

```
$ sudo systemctl stop pulseaudio-enable-autospawn.service
```

At the time of writing there was a fault with the **lxplug-volumepulse** package which stops any of the menus on the Raspberry Pi desktop from being displayed. Remove the **lxplug-volumepulse** package with the following command:

```
$ sudo apt remove lxplug-volumepulse
```

## Installing OLED TFT driver software and libraries

If you are not using any OLED TFT display then skip this section.

**IMPORTANT NOTE:** The following devices and programs use **PIP3** to install the drivers.

- Olimex OLED or SSD1306 OLED TFTs
- Pimoroni Pirate Audio Radio
- SH1106 SPI OLED TFT
- Adafruit 2.5 and 3.5-inch TFT touchscreen
- Monochrome OLEDs which use the LUMA TFT driver
- The weather programs
- The IQaudIO controller

If you attempt to use **PIP3** (The name is a recursive acronym for "Pip Installs Packages") on the **Bookworm OS** it will fail with the following message:

```
To install Python packages system-wide, try apt install
python3-xyz, where xyz is the package you are trying to
install.
```

```
If you wish to install a non-Debian-packaged Python package,
create a virtual environment using python3 -m venv path/to/venv.
Then use path/to/venv/bin/python and path/to/venv/bin/pip. Make
sure you have python3-full installed.
```

```
For more information visit http://rptl.io/venv
```

To allow **pip3** to be used on **Bookworm OS** carry out the following instruction:

```
$ sudo mv /usr/lib/python3.11/EXTERNALLY-MANAGED  
/usr/lib/python3.11/EXTERNALLY-MANAGED.old
```

Note that the above is all one line. There is no need to carry out the above instruction on system running **Bullseye OS**.

You will now be able to use **pip3** to install the driver software and programs shown in this manual, however you will see the following warning message:

```
WARNING: Running pip as the 'root' user can result in broken permissions and  
conflicting behaviour with the system package manager. It is recommended to  
use a virtual environment instead: https://pip.pypa.io/warnings/venv
```

The above message can be safely ignored for the installation procedures in this manual. However, if installing other author's software with **pip3**, you should seek advice from that author.

If using **Raspbian Lite** also install the **python3-pil** package:

```
$ sudo apt -y install python3-pil
```

### Install display drivers

Note: If you are installing the software for any of the products listed below then first carry out the instructions in the appropriate section. Do this before installing the radio software.

**Table 3 Display driver software installation**

Device	Section/Devices	Page
Pirate Radio	Installing Pimoroni Pirate Radio (pHat BEAT)	34
Pirate Audio	Installing the Pimoroni Pirate Audio	44
SSD1306 0.9-inch OLED	Installing the SSD1306 OLED driver libraries	49
SH1106 SPI (joystick)	Installing the SH1106 SPI OLED driver libraries	53
MHS RPi displays	Error! Reference source not found.	Error! Bookmark not defined.
SH1106 1.3-inch OLED	Installing LUMA monochrome OLEDs	59
LUMA OLED devices	SSD1306, SSD1309, SSD1325, SSD1331, SH1106, WS0010	59
Waveshare TFTs	Error! Reference source not found.	Error! Bookmark not defined.
Grove LCD RGB	Error! Reference source not found.	Error! Bookmark not defined.
Adafruit TFT	Installing the Adafruit 2.5 and 3.5-inch TFT touchscreen	54
PiFace CAD	Error! Reference source not found.	Error! Bookmark not defined.
Waveshare SH1106 SPI	Installing the SH1106 SPI OLED driver libraries	53

3. Installing the Radio Daemon on page 26.
  - a. Select option 5 Pimoroni pHat BEAT with own push buttons
  - b. Select option 1 40-pin wiring
  - c. No display used/Pimoroni Pirate radio
  - d. Audio configuration – Select 4 Pimoroni Pirate/pHAT with PiVumeter
  
4. Since the Pirate Radio does not have a screen, you can optionally install **espeak** as shown in the section called ***Radio software pre-requisites***
5. ***Upgrading from previous*** versions and other changes

If upgrading from version **6.x** or earlier it is necessary to replace your existing configuration file with the new one supplied in the upgrade. First take a note of any specific configuration items that are not standard or make a copy of your existing **/etc/radiod.conf** configuration file.

**Important:** The name format of Radio playlists has changed from version 7.3 onwards. See ***Creating and Maintaining Playlist files*** on page 169. **Also, in this release**, it is now possible to add, move and delete entries in the play list using any MPD client that is capable of doing so. See **Maintaining playlists using external MPD clients** on page 173.

**Note:** The location of the log file has changed from **/var/log/radio.log** to **/var/log/radiod/radio.log**. Also, its ownership has been changed from **root** to user **pi**. This has been done to reduce the number of programs that need to run with **sudo**.

**IMPORTANT:** In **Bookworm** the location for the boot configuration file **/boot/config.txt** has been changed to **/boot/firmware/config.txt**. Nearly all instructions in this manual refer to **/boot/firmware/config.txt** however if using **Bullseye** change the instruction to use **/boot/config.txt**. The installation scripts will choose the correct location.

### Installing the Music player daemon

If you haven't already done so upgrade the system packages as shown in **Error! Reference source not found.** on page **Error! Bookmark not defined..**

After reboot install the Music Player Daemon (mpd) and its client (mpc) along with the Python3, MPD and **python3-rpi.gpio** libraries. Note that you must install **python3-mpd** and not **python.mpd**.

```
$ sudo apt install mpd mpc python3-mpd python3-rpi.gpio
```

Answer yes 'y' when asked to continue.

If the installation says it cannot find the above packages, then re-run the update command to update the library list and retry installing the above packages.

```
$ sudo apt update
```

**IMPORTANT:** Unfortunately, the configuration file (**/etc/mpd.conf**) supplied with MPPD version 0.22.6 in **Bookworm** and onwards is corrupt. It is necessary to correct this as show below.

Find the following entries in **/etc/mpd.conf** somewhere around line 232.

#### Corrupt /etc/mpd.conf file

```
#audio_output {
#    type          "alsa"
#    name          "My ALSA device"
#    device        "hw:0,0"
```

```

#      mixer_type      "software"
##    mixer_device     "default"      # optional
##    mixer_control    "PCM"        # optional
##    mixer_index      "0"          # optional
#

```

Change the above entries to the following:

### Corrected /etc/mpd.conf file

```

audio_output {
    type      "alsa"
    name      "My ALSA Device"
    device    "hw:0,0"
    mixer_type "software"
#   mixer_device     "default"      # optional
#   mixer_control    "PCM"        # optional
#   mixer_index      "0"          # optional
}

```

If installing on **Bookworm Lite** this will take quite a long time as there are a lot of software libraries to be installed.

At this stage there are no playlists configured so the music daemon won't play anything. The play lists are created when the **radiod** Debian Package is installed in the next section.

The version of **MPD** released with **Bookworm** is **0.22.6** which is reasonably up-to-date. You should install MPD version **0.22.6** first. When it is all working if you are having problems playing certain stations, it is well worth installing the latest version of MPD as shown in the section called *Compiling and installing the latest Music Player Daemon* on page 118.

### Installing pulseaudio

The **pulseaudio** package may or may not needed to be installed. This has been different between various releases of the operating system. Currently the **pulseaudio** package causes a lot of problems with the radio software, in particular **espeak** and the Alsa sound system. If you do not intend to use espeak and need pulseaudio, for example, for blue-tooth devices or pHat BEAT then install it. The following table shows the options:

**Table 2 PulseAudio installation options**

Type of radio installation	Install pulseaudio
<b>Using espeak</b>	No
<b>Pimoroni Pirate Radio with pHat BEAT</b>	Yes (Installed by phatbeat)
<b>Pimoroni Pirate Audio/mini-speaker</b>	No
<b>Adafruit speaker bonnet</b>	No
<b>LCD display radio</b>	No unless using above DACs
<b>HDMI or touch-screen displays</b>	No unless using above DACs
<b>IQaudIO, HiFiBerry or JustBoom DACs</b>	No
<b>Bluetooth sound devices</b>	Yes

To install **pulseaudio**:

```
$ sudo apt install pulseaudio
```

To remove **pulseaudio**:

```
$ sudo apt remove pulseaudio
```

If installing on Bookworm disable the `pulseaudio-enable-autospawn.service`.

```
$ sudo systemctl stop pulseaudio-enable-autospawn.service
```

At the time of writing there was a fault with the **lxplug-volumepulse** package which stops any of the menus on the Raspberry Pi desktop from being displayed. Remove the **lxplug-volumepulse** package with the following command:

```
$ sudo apt remove lxplug-volumepulse
```

## Installing OLED TFT driver software and libraries

If you are not using any OLED TFT display then skip this section.

**IMPORTANT NOTE:** The following devices and programs use **PIP3** to install the drivers.

- Olimex OLED or SSD1306 OLED TFTs
- Pimoroni Pirate Audio Radio
- SH1106 SPI OLED TFT
- Adafruit 2.5 and 3.5-inch TFT touchscreen
- Monochrome OLEDs which use the LUMA TFT driver
- The weather programs
- The IQaudIO controller

If you attempt to use **PIP3** (The name is a recursive acronym for "Pip Installs Packages") on the **Bookworm OS** it will fail with the following message:

```
To install Python packages system-wide, try apt install
python3-xyz, where xyz is the package you are trying to
install.
```

```
If you wish to install a non-Debian-packaged Python package,
create a virtual environment using python3 -m venv path/to/venv.
Then use path/to/venv/bin/python and path/to/venv/bin/pip. Make
sure you have python3-full installed.
```

```
For more information visit http://rptl.io/venv
```

To allow **pip3** to be used on **Bookworm OS** carry out the following instruction:

```
$ sudo mv /usr/lib/python3.11/EXTERNALLY-MANAGED
/usr/lib/python3.11/EXTERNALLY-MANAGED.old
```

Note that the above is all one line. There is no need to carry out the above instruction on system running **Bullseye OS**.

You will now be able to use **pip3** to install the driver software and programs shown in this manual, however you will see the following warning message:

WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package manager. It is recommended to use a virtual environment instead: <https://pip.pypa.io/warnings/venv>

The above message can be safely ignored for the installation procedures in this manual. However, if installing other author's software with **pip3**, you should seek advice from that author.

If using **Raspbian Lite** also install the **python3-pil** package:

```
$ sudo apt -y install python3-pil
```

### Install display drivers

Note: If you are installing the software for any of the products listed below then first carry out the instructions in the appropriate section. Do this before installing the radio software.

**Table 3 Display driver software installation**

Device	Section/Devices	Page
Pirate Radio	<b>Installing Pimoroni Pirate Radio (pHat BEAT)</b>	34
Pirate Audio	<b>Installing the Pimoroni Pirate Audio</b>	44
SSD1306 0.9-inch OLED	Installing the SSD1306 OLED driver libraries	49
SH1106 SPI (joystick)	Installing the SH1106 SPI OLED driver libraries	53
MHS RPi displays	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
SH1106 1.3-inch OLED	<b>Installing LUMA monochrome OLEDs</b>	59
LUMA OLED devices	SSD1306, SSD1309, SSD1325, SSD1331, SH1106, WS0010	59
Waveshare TFTs	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
Grove LCD RGB	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
Adafruit TFT	<b>Installing the Adafruit 2.5 and 3.5-inch TFT touchscreen</b>	54
PiFace CAD	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
Waveshare SH1106 SPI	Installing the SH1106 SPI OLED driver libraries	53

### Installing the Radio Daemon

**Note:** The `python-configparser` package has been dropped in **Bookworm** and appears to be satisfied with the package `python3-iniparse`. See:  
<https://packages.debian.org/bookworm/mips64el/python3-iniparse>

The latest version of the `radiod` (Version 7.9) setup.sh script has been amended to skip installation of `python-configparser` if using **Bookworm**.

If running **Bullseye** first install the `python-configparser` package. Skip this set if running **Bookworm** or later:

```
$ sudo apt install python-configparser
```

The Raspberry PI Internet Radio software is distributed as a Debian package. This can be downloaded from [http://www.bobrathbone.com/raspberrypi/pi\\_internet\\_radio.html](http://www.bobrathbone.com/raspberrypi/pi_internet_radio.html)  
Either download it to your PC or Macintosh and copy it to the `/home/pi` directory or get it directly using the `wget` facility.

To use the `wget` facility first copy the download link from the above page (Right click on the link). Log into the Raspberry PI. Now use `wget` to the software package.

#### For 32-bit systems

```
$ wget http://www.bobrathbone.com/raspberrypi/packages/radiod_7.9_armhf.deb
```

Run `dpkg` to install the package.

```
$ sudo dpkg -i radiod_7.9_armhf.deb
```

#### For 64-bit systems:

```
$ wget http://www.bobrathbone.com/raspberrypi/packages/radiod_7.9_arm64.deb
```

Run `dpkg` to install the package.

```
$ sudo dpkg -i radiod_7.9_armhf.deb
```

If upgrading or re-installing the radio software use the following to overwrite the existing package download file (Substitute correct file name):

```
$ curl -L -O http://bobrathbone.com/raspberrypi/packages/radiod_7.9_arm64.deb
```

The `dpkg` program will install the files.

```
(Reading database ... 131542 files and directories currently  
installed.Preparing to unpack radiod_7.9_armhf.deb ...  
Raspberry PI internet radio installation  
Stopping radiod service
```

```
Unpacking radiod (7.9)
```

If updating from an earlier version of the radio software you may see the following message.  
Enter 'I' or 'Y' to install the new version.

```
Configuration file '/etc/logrotate.d/radiod'
==> Deleted (by you or by a script) since installation.
==> Package distributor has shipped an updated version.
What would you like to do about it ? Your options are:
 Y or I  : install the package maintainer's version
 N or O  : keep your currently-installed version
 D      : show the differences between the versions
 Z      : start a shell to examine the situation
The default action is to keep your current version.
*** radiod (Y/I/N/O/D/Z) [default=N] ? I
```

6. Installing the speech facility on page **Error! Bookmark not defined.** to hear choices when using the menu button.

Finally reboot the Raspberry Pi to start the radio.

```
$ sudo reboot
```



If no sound is heard from the Pirate radio, then use the volume up (+) button to increase the volume until sound is heard.

## Installing the Pimoroni Pirate Audio

Install the packages required by the Pimoroni Pirate Audio .

```
$ sudo apt -y install python3-rpi.gpio python3-spidev python3-pip python3-pil python3-numpy python3-pip git
```

Uninstall any old **numpy** package

```
$ sudo pip3 uninstall numpy
```

Install setup tools

```
$ sudo pip3 install setuptools
```

Install the st7789 library if using the Pimoroni Pirate Audio. Not needed for the SSD1306 OLED.

```
$ sudo pip3 install st7789
```

Now carry out the instructions shown in **Error! Reference source not found.** on page **Error! Bookmark not defined..** Select the following.

1. User interface - Select option 8 Pimoroni Audio with four push buttons
2. Select option 1 40-pin wiring
3. Display type - Select Pimoroni Audio ST7789 TFT

When configuring the audio output, select option 14 *Pimoroni Pirate Audio (HiFiBerry DAC)*. This will configure **/boot/firmware/config.txt** (for **Bookworm**) or **/boot/config.tx** (for **Bullseye**) with the following lines.

```
dtoverlay=hifiberry-dac
```

and

```
dtparam=audio=off
```

More details about the configuration for the Pirate Audio are found in section **Error! Reference source not found.** on page **Error! Bookmark not defined..**

**Note 1:** Button Y (Volume up) can be GPIO 24 instead of 20 on versions of the Pirate Audio card produced before January 2020.

The new settings in **/etc/radiod.conf** are normally:

```
up_switch=16
down_switch=5
left_switch=6
right_switch=20
```

or on some variants.

```
right_switch=24
```

Manually edit **/etc/radiod.conf** if your Pirate Audio is using GPIO24.

Now carry out the instructions shown in section *Radio* software pre-requisites

Upgrading from previous versions and other changes

If upgrading from version **6.x** or earlier it is necessary to replace your existing configuration file with the new one supplied in the upgrade. First take a note of any specific configuration items that are not standard or make a copy of your existing **/etc/radiod.conf** configuration file.

**Important:** The name format of Radio playlists has changed from version 7.3 onwards. See *Creating and Maintaining Playlist files* on page 169. **Also, in this release**, it is now possible to add, move and delete entries in the play list using any MPD client that is capable of doing so. See Maintaining playlists using external MPD clients on page 173.

**Note:** The location of the log file has changed from **/var/log/radio.log** to **/var/log/radiod/radio.log**. Also, its ownership has been changed from **root** to user **pi**. This has been done to reduce the number of programs that need to run with **sudo**.

**IMPORTANT:** In **Bookworm** the location for the boot configuration file **/boot/config.txt** has been changed to **/boot/firmware/config.txt**. Nearly all instructions in this manual refer to **/boot/firmware/config.txt** however if using **Bullseye** change the instruction to use **/boot/config.txt**. The installation scripts will choose the correct location.

## Installing the Music player daemon

If you haven't already done so upgrade the system packages as shown in **Error! Reference source not found.** on page **Error! Bookmark not defined..**

After reboot install the Music Player Daemon (mpd) and its client (mpc) along with the Python3, MPD and **python3-rpi.gpio** libraries. Note that you must install **python3-mpd** and not **python.mpd**.

```
$ sudo apt install mpd mpc python3-mpd python3-rpi.gpio
```

Answer yes 'y' when asked to continue.

If the installation says it cannot find the above packages, then re-run the update command to update the library list and retry installing the above packages.

```
$ sudo apt update
```

**IMPORTANT:** Unfortunately, the configuration file (**/etc/mpd.conf**) supplied with MPD version 0.22.6 in **Bookworm** and onwards is corrupt. It is necessary to correct this as show below.

Find the following entries in **/etc/mpd.conf** somewhere around line 232.

### Corrupt /etc/mpd.conf file

```
#audio_output {
#    type          "alsa"
#    name          "My ALSA device"
#    device        "hw:0,0"
#    mixer_type    "software"
##    mixer_device  "default"      # optional
##    mixer_control "PCM"         # optional
##    mixer_index   "0"           # optional
#}
```

Change the above entries to the following:

### Corrected /etc/mpd.conf file

```
audio_output {
    type          "alsa"
    name          "My ALSA Device"
    device        "hw:0,0"
    mixer_type    "software"
#    mixer_device  "default"      # optional
#    mixer_control "PCM"         # optional
#    mixer_index   "0"           # optional
}
```

If installing on **Bookworm Lite** this will take quite a long time as there are a lot of software libraries to be installed.

At this stage there are no playlists configured so the music daemon won't play anything. The play lists are created when the **radiod** Debian Package is installed in the next section.

The version of **MPD** released with **Bookworm** is **0.22.6** which is reasonably up-to-date. You should install MPD version **0.22.6** first. When it is all working if you are having problems playing certain stations, it is well worth installing the latest version of MPD as shown in the section called *Compiling and installing the latest Music Player Daemon* on page 118.

### Installing pulseaudio

The **pulseaudio** package may or may not need to be installed. This has been different between various releases of the operating system. Currently the **pulseaudio** package causes a lot of problems with the radio software, in particular **espeak** and the Alsa sound system. If you do not intend to use espeak and need pulseaudio, for example, for blue-tooth devices or pHat BEAT then install it. The following table shows the options:

**Table 2 PulseAudio installation options**

Type of radio installation	Install pulseaudio
Using espeak	No
Pimoroni Pirate Radio with pHat BEAT	Yes (Installed by phatbeat)
Pimoroni Pirate Audio/mini-speaker	No
Adafruit speaker bonnet	No
LCD display radio	No unless using above DACs
HDMI or touch-screen displays	No unless using above DACs
IQaudIO, HiFiBerry or JustBoom DACs	No
Bluetooth sound devices	Yes

To install **pulseaudio**:

```
$ sudo apt install pulseaudio
```

To remove **pulseaudio**:

```
$ sudo apt remove pulseaudio
```

If installing on Bookworm disable the `pulseaudio-enable-autospawn.service`.

```
$ sudo systemctl stop pulseaudio-enable-autospawn.service
```

At the time of writing there was a fault with the **lxplug-volumepulse** package which stops any of the menus on the Raspberry Pi desktop from being displayed. Remove the **lxplug-volumepulse** package with the following command:

```
$ sudo apt remove lxplug-volumepulse
```

### Installing OLED TFT driver software and libraries

If you are not using any OLED TFT display then skip this section.

**IMPORTANT NOTE:** The following devices and programs use **PIP3** to install the drivers.

- Olimex OLED or SSD1306 OLED TFTs
- Pimoroni Pirate Audio Radio
- SH1106 SPI OLED TFT

- Adafruit 2.5 and 3.5-inch TFT touchscreen
- Monochrome OLEDs which use the LUMA TFT driver
- The weather programs
- The IQaudIO controller

If you attempt to use **PIP3** (The name is a recursive acronym for "Pip Installs Packages") on the **Bookworm OS** it will fail with the following message:

```
To install Python packages system-wide, try apt install
python3-xyz, where xyz is the package you are trying to
install.

If you wish to install a non-Debian-packaged Python package,
create a virtual environment using python3 -m venv path/to/venv.
Then use path/to/venv/bin/python and path/to/venv/bin/pip. Make
sure you have python3-full installed.

For more information visit http://rptl.io/venv
```

To allow **pip3** to be used on **Bookworm OS** carry out the following instruction:

```
$ sudo mv /usr/lib/python3.11/EXTERNALLY-MANAGED
/usr/lib/python3.11/EXTERNALLY-MANAGED.old
```

Note that the above is all one line. There is no need to carry out the above instruction on system running **Bullseye OS**.

You will now be able to use **pip3** to install the driver software and programs shown in this manual, however you will see the following warning message:

```
WARNING: Running pip as the 'root' user can result in broken permissions and
conflicting behaviour with the system package manager. It is recommended to
use a virtual environment instead: https://pip.pypa.io/warnings/venv
```

The above message can be safely ignored for the installation procedures in this manual. However, if installing other author's software with **pip3**, you should seek advice from that author.

If using **Raspbian Lite** also install the **python3-pil** package:

```
$ sudo apt -y install python3-pil
```

### Install display drivers

Note: If you are installing the software for any of the products listed below then first carry out the instructions in the appropriate section. Do this before installing the radio software.

**Table 3 Display driver software installation**

Device	Section/Devices	Page
Pirate Radio	Installing Pimoroni Pirate Radio (pHat BEAT)	34
Pirate Audio	Installing the Pimoroni Pirate Audio	44

<b>SSD1306 0.9-inch OLED</b>	Installing the SSD1306 OLED driver libraries	49
<b>SH1106 SPI (joystick)</b>	Installing the SH1106 SPI OLED driver libraries	53
<b>MHS RPi displays</b>	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
<b>SH1106 1.3-inch OLED</b>	<b>Installing LUMA monochrome OLEDs</b>	59
<b>LUMA OLED devices</b>	SSD1306, SSD1309, SSD1325, SSD1331, SH1106, WS0010	59
<b>Waveshare TFTs</b>	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
<b>Grove LCD RGB</b>	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
<b>Adafruit TFT</b>	<b>Installing the Adafruit 2.5 and 3.5-inch TFT touchscreen</b>	54
<b>PiFace CAD</b>	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
<b>Waveshare SH1106 SPI</b>	Installing the SH1106 SPI OLED driver libraries	53

Installing the Radio Daemon on page 26.

### Installing the SSD1306 OLED driver libraries

SSD1306 devices can either use the **ssd1306\_class** driver or the **luma\_class** driver configured for SSD1306. The Luma driver supports multiple devices. See [Installing LUMA monochrome OLEDs](#) on page 59. The 0.9-inch SSD1306 OLED display works with either the **ssd1306\_class** or **luma\_class** driver.

To use the *Luma* driver see [Installing LUMA monochrome OLEDs](#) on page 59.

Otherwise follow the instructions below to use the SSD1306 driver. These install the `Adafruit_Python_SSD1306` driver software.

```
$ sudo apt install git
$ git clone https://github.com/adafruit/Adafruit_Python_SSD1306.git
$ cd Adafruit_Python_SSD1306
$ sudo python3 setup.py install
```

Now carry out the instructions shown in section *Radio* software pre-requisites

Upgrading from previous versions and other changes

If upgrading from version **6.x** or earlier it is necessary to replace your existing configuration file with the new one supplied in the upgrade. First take a note of any specific configuration items that are not standard or make a copy of your existing `/etc/radiod.conf` configuration file.

**Important:** The name format of Radio playlists has changed from version 7.3 onwards. See [Creating and Maintaining Playlist files](#) on page 169. Also, in this release, it is now possible to add, move and delete entries in the play list using any MPD client that is capable of doing so. See [Maintaining playlists using external MPD clients](#) on page 173.

**Note:** The location of the log file has changed from `/var/log/radio.log` to `/var/log/radiod/radio.log`. Also, its ownership has been changed from **root** to user **pi**. This has been done to reduce the number of programs that need to run with **sudo**.

**IMPORTANT:** In **Bookworm** the location for the boot configuration file `/boot/config.txt` has been changed to `/boot/firmware/config.txt`. Nearly all instructions in this manual refer to `/boot/firmware/config.txt` however if using **Bullseye** change the instruction to use `/boot/config.txt`. The installation scripts will choose the correct location.

## Installing the Music player daemon

If you haven't already done so upgrade the system packages as shown in **Error! Reference source not found.** on page **Error! Bookmark not defined..**

After reboot install the Music Player Daemon (mpd) and its client (mpc) along with the Python3, MPD and `python3-rpi.gpio` libraries. Note that you must install `python3-mpd` and not `python.mpd`.

```
$ sudo apt install mpd mpc python3-mpd python3-rpi.gpio
```

Answer yes 'y' when asked to continue.

If the installation says it cannot find the above packages, then re-run the update command to update the library list and retry installing the above packages.

```
$ sudo apt update
```

**IMPORTANT:** Unfortunately, the configuration file (`/etc/mpd.conf`) supplied with MPD version 0.22.6 in **Bookworm** and onwards is corrupt. It is necessary to correct this as show below.

Find the following entries in `/etc/mpd.conf` somewhere around line 232.

### Corrupt /etc/mpd.conf file

```
#audio_output {
#    type          "alsa"
#    name          "My ALSA device"
#    device        "hw:0,0"
#    mixer_type    "software"
##    mixer_device  "default"      # optional
##    mixer_control "PCM"         # optional
##    mixer_index   "0"           # optional
#}
```

Change the above entries to the following:

### Corrected /etc/mpd.conf file

```
audio_output {
    type          "alsa"
    name          "My ALSA Device"
    device        "hw:0,0"
    mixer_type    "software"
#    mixer_device  "default"      # optional
#    mixer_control "PCM"         # optional
#    mixer_index   "0"           # optional
}
```

If installing on **Bookworm Lite** this will take quite a long time as there are a lot of software libraries to be installed.

At this stage there are no playlists configured so the music daemon won't play anything. The play lists are created when the **radiod** Debian Package is installed in the next section.

The version of **MPD** released with **Bookworm** is **0.22.6** which is reasonably up-to-date. You should install MPD version **0.22.6** first. When it is all working if you are having problems playing certain stations, it is well worth installing the latest version of MPD as shown in the section called *Compiling and installing the latest Music Player Daemon* on page 118.

### Installing pulseaudio

The **pulseaudio** package may or may not needed to be installed. This has been different between various releases of the operating system. Currently the **pulseaudio** package causes a lot of problems with the radio software, in particular **espeak** and the Alsa sound system. If you do not intend to use espeak and need pulseaudio, for example, for blue-tooth devices or pHat BEAT then install it. The following table shows the options:

**Table 2 PulseAudio installation options**

Type of radio installation	Install pulseaudio
Using espeak	No
Pimoroni Pirate Radio with pHat BEAT	Yes (Installed by phatbeat)
Pimoroni Pirate Audio/mini-speaker	No
Adafruit speaker bonnet	No
LCD display radio	No unless using above DACs
HDMI or touch-screen displays	No unless using above DACs
IQaudIO, HiFiBerry or JustBoom DACs	No
Bluetooth sound devices	Yes

To install **pulseaudio**:

```
$ sudo apt install pulseaudio
```

To remove **pulseaudio**:

```
$ sudo apt remove pulseaudio
```

If installing on Bookworm disable the `pulseaudio-enable-autospawn.service`.

```
$ sudo systemctl stop pulseaudio-enable-autospawn.service
```

At the time of writing there was a fault with the **lxplug-volumepulse** package which stops any of the menus on the Raspberry Pi desktop from being displayed. Remove the **lxplug-volumepulse** package with the following command:

```
$ sudo apt remove lxplug-volumepulse
```

## Installing OLED TFT driver software and libraries

If you are not using any OLED TFT display then skip this section.

**IMPORTANT NOTE:** The following devices and programs use **PIP3** to install the drivers.

- Olimex OLED or SSD1306 OLED TFTs
- Pimoroni Pirate Audio Radio
- SH1106 SPI OLED TFT
- Adafruit 2.5 and 3.5-inch TFT touchscreen
- Monochrome OLEDs which use the LUMA TFT driver
- The weather programs
- The IQaudIO controller

If you attempt to use **PIP3** (The name is a recursive acronym for "Pip Installs Packages") on the **Bookworm OS** it will fail with the following message:

```
To install Python packages system-wide, try apt install
python3-xyz, where xyz is the package you are trying to
install.

If you wish to install a non-Debian-packaged Python package,
create a virtual environment using python3 -m venv path/to/venv.
Then use path/to/venv/bin/python and path/to/venv/bin/pip. Make
sure you have python3-full installed.

For more information visit http://rptl.io/venv
```

To allow **pip3** to be used on **Bookworm OS** carry out the following instruction:

```
$ sudo mv /usr/lib/python3.11/EXTERNALLY-MANAGED
/usr/lib/python3.11/EXTERNALLY-MANAGED.old
```

Note that the above is all one line. There is no need to carry out the above instruction on system running **Bullseye OS**.

You will now be able to use **pip3** to install the driver software and programs shown in this manual, however you will see the following warning message:

```
WARNING: Running pip as the 'root' user can result in broken permissions and
conflicting behaviour with the system package manager. It is recommended to
use a virtual environment instead: https://pip.pypa.io/warnings/venv
```

The above message can be safely ignored for the installation procedures in this manual. However, if installing other author's software with **pip3**, you should seek advice from that author.

If using **Raspbian Lite** also install the **python3-pil** package:

```
$ sudo apt -y install python3-pil
```

## Install display drivers

Note: If you are installing the software for any of the products listed below then first carry out the instructions in the appropriate section. Do this before installing the radio software.

**Table 3 Display driver software installation**

Device	Section/Devices	Page
Pirate Radio	<b>Installing Pimoroni Pirate Radio (pHat BEAT)</b>	34
Pirate Audio	<b>Installing the Pimoroni Pirate Audio</b>	44
SSD1306 0.9-inch OLED	Installing the SSD1306 OLED driver libraries	49
SH1106 SPI (joystick)	Installing the SH1106 SPI OLED driver libraries	53
MHS RPi displays	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
SH1106 1.3-inch OLED	<b>Installing LUMA monochrome OLEDs</b>	59
LUMA OLED devices	SSD1306, SSD1309, SSD1325, SSD1331, SH1106, WS0010	59
Waveshare TFTs	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
Grove LCD RGB	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
Adafruit TFT	<b>Installing the Adafruit 2.5 and 3.5-inch TFT touchscreen</b>	54
PiFace CAD	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
Waveshare SH1106 SPI	Installing the SH1106 SPI OLED driver libraries	53

Installing the Radio Daemon on page 26.

## Installing the SH1106 SPI OLED driver libraries

The SH1106 SPI driver used by the 1.3" OLED with joystick and 3-button interface requires the following commands to install the prerequisite libraries. The following instructions are only for the SPI interface and not the I2C interface.

```
$ sudo apt-get install python3-pip  
$ sudo pip3 install RPi.GPIO  
$ sudo apt-get install python3-smbus  
$ sudo pip3 install spidev
```

When you run the radio configuration program later on it will configure **/etc/radiod.conf** with the following switch settings.

```
menu_switch=13  
mute_switch=21  
up_switch=6  
down_switch=19  
left_switch=5  
right_switch=26
```

## Installing the Adafruit 2.5 and 3.5-inch TFT touchscreen



If using the Adafruit TFT display this procedure needs to be carried out before attempting to install the Radio software. This procedure requires the desktop version of the Raspberry Pi OS



**Note:** There are two types of TFT touch-screen available screen available from Adafruit namely Capacitive or Resistive. The one used in this project is the Resistive type.

Plug the TFT screen into the Raspberry Pi 40 pin header. Depending upon the type of display the TFT screen will have either a 26-pin or 40-pin female header which plugs into the GPIO header.

The basic installation instructions can be found on the Adafruit Web site:

<https://cdn-learn.adafruit.com/downloads/pdf/adafruit-pitft-3-dot-5-touch-screen-for-raspberry-pi.pdf>

Or from the Bob Rathbone Web site:

<http://bobrathbone.com/raspberrypi/documents/Adafruit/adafruit-pitft-3-dot-5-touch-screen-for-raspberry-pi.pdf>

Software installation is also covered in the above manual.

Look at the section called *FBCP Install Commands* (HDMI Mirroring).

Below is the easy installation for the Adafruit TFT displays:

Install pre-requisite python3-pip (May already be installed)

```
$ cd ~  
$ sudo apt install -y git python3-pip
```

Download the Adafruit installer scripts:

```
$ sudo pip3 install --upgrade adafruit-python-shell click  
$ git clone https://github.com/adafruit/Raspberry-Pi-Installer-Scripts.git
```

Now run the setup scripts.

```
$ cd Raspberry-Pi-Installer-Scripts/  
$ sudo python3 adafruit-pitft.py --display=35r --rotation=90 --install-type=fbcsp
```

Replace 35r with 28r for the 2.8-inch TFT. To flip the display, change the rotation from 90 to 270 and re-run the python3 script.

The above command adds the following to **/boot/firmware/config.txt** (for Bookworm) or **/boot/config.txt** (for Bullseye).

```
# --- added by adafruit-pitft-helper Sun Jul 11 11:15:00 2021 ---  
hdmi_force_hotplug=1  
dtparam=spi=on  
dtparam=i2c1=on  
dtparam=i2c_arm=on
```

```
dtoverlay=pitft35-resistive,rotate=90,speed=2000000,fps=20
# --- end adafruit-pitft-helper Sun Jul 11 11:15:00 2021 ---
hdmi_cvt=720 480 60 1 0 0 0
```

Make sure that the resolution specified matches your device. This is 720x480 in the above example. The supported resolutions are 480x320 or 720x480 pixels for the 2.8-inch and 3.5-inch TFT respectively. Smaller resolutions are not supported.

Edit **/boot/firmware/config.txt** (for Bookworm) or **/boot/config.txt** (for Bullseye) and force the console to the same size with the same values.

```
framebuffer_width=720
framebuffer_height=480
```



It is very important that the resolution set in the above instructions matches your device and that they have been set to the same value. If not, the results will be unpredictable.

The TFT screen should already be calibrated but if calibration required refer to the section called *Resistive Touchscreen Manual Install & Calibrate* in the Adafruit in the Adafruit manual.

There is also an interactive installation script:

```
sudo python3 adafruit-pitft.py
```

See the Adafruit manual for the full procedure.

Reboot the Raspberry Pi.

```
$ sudo reboot
```

If everything has been correctly configured the Raspberry Pi desktop should be seen on the TFT display. If a HDMI monitor is also plugged in then the RPi desktop should also be seen mirrored on it.

Now carry out the instructions shown in section *Radio* software pre-requisites

Upgrading from previous versions and other changes

If upgrading from version **6.x** or earlier it is necessary to replace your existing configuration file with the new one supplied in the upgrade. First take a note of any specific configuration items that are not standard or make a copy of your existing **/etc/radiod.conf** configuration file.

**Important:** The name format of Radio playlists has changed from version 7.3 onwards. See *Creating and Maintaining Playlist files* on page 169. **Also, in this release**, it is now possible to add, move and delete entries in the play list using any MPD client that is capable of doing so. See Maintaining playlists using external MPD clients **on page 173**.

**Note:** The location of the log file has changed from **/var/log/radio.log** to **/var/log/radiod/radio.log**. Also, its ownership has been changed from **root** to user **pi**. This has been done to reduce the number of programs that need to run with **sudo**.

**IMPORTANT:** In **Bookworm** the location for the boot configuration file **/boot/config.txt** has been changed to **/boot/firmware/config.txt**. Nearly all instructions in this manual refer to **/boot/firmware/config.txt** however if using **Bullseye** change the instruction to use **/boot/config.txt**. The installation scripts will choose the correct location.

## Installing the Music player daemon

If you haven't already done so upgrade the system packages as shown in **Error! Reference source not found.** on page **Error! Bookmark not defined..**

After reboot install the Music Player Daemon (mpd) and its client (mpc) along with the Python3, MPD and **python3-rpi.gpio** libraries. Note that you must install **python3-mpd** and not **python.mpd**.

```
$ sudo apt install mpd mpc python3-mpd python3-rpi.gpio
```

Answer yes 'y' when asked to continue.

If the installation says it cannot find the above packages, then re-run the update command to update the library list and retry installing the above packages.

```
$ sudo apt update
```

**IMPORTANT:** Unfortunately, the configuration file (**/etc/mpd.conf**) supplied with MPD version 0.22.6 in **Bookworm** and onwards is corrupt. It is necessary to correct this as show below.

Find the following entries in **/etc/mpd.conf** somewhere around line 232.

### Corrupt /etc/mpd.conf file

```
#audio_output {
#    type          "alsa"
#    name          "My ALSA device"
#    device        "hw:0,0"
#    mixer_type    "software"
##    mixer_device  "default"      # optional
##    mixer_control "PCM"         # optional
##    mixer_index   "0"           # optional
#}
```

Change the above entries to the following:

### Corrected /etc/mpd.conf file

```
audio_output {
    type          "alsa"
    name          "My ALSA Device"
    device        "hw:0,0"
    mixer_type    "software"
#    mixer_device  "default"      # optional
#    mixer_control "PCM"         # optional
#    mixer_index   "0"           # optional
}
```

If installing on **Bookworm Lite** this will take quite a long time as there are a lot of software libraries to be installed.

At this stage there are no playlists configured so the music daemon won't play anything. The play lists are created when the **radiod** Debian Package is installed in the next section.

The version of **MPD** released with **Bookworm** is **0.22.6** which is reasonably up-to-date. You should install MPD version **0.22.6** first. When it is all working if you are having problems playing certain stations, it is well worth installing the latest version of MPD as shown in the section called *Compiling and installing the latest Music Player Daemon* on page 118.

### Installing pulseaudio

The **pulseaudio** package may or may not need to be installed. This has been different between various releases of the operating system. Currently the **pulseaudio** package causes a lot of problems with the radio software, in particular **espeak** and the Alsa sound system. If you do not intend to use espeak and need pulseaudio, for example, for blue-tooth devices or pHat BEAT then install it. The following table shows the options:

**Table 2 PulseAudio installation options**

Type of radio installation	Install pulseaudio
Using espeak	No
Pimoroni Pirate Radio with pHat BEAT	Yes (Installed by phatbeat)
Pimoroni Pirate Audio/mini-speaker	No
Adafruit speaker bonnet	No
LCD display radio	No unless using above DACs
HDMI or touch-screen displays	No unless using above DACs
IQaudIO, HiFiBerry or JustBoom DACs	No
Bluetooth sound devices	Yes

To install **pulseaudio**:

```
$ sudo apt install pulseaudio
```

To remove **pulseaudio**:

```
$ sudo apt remove pulseaudio
```

If installing on Bookworm disable the `pulseaudio-enable-autospawn.service`.

```
$ sudo systemctl stop pulseaudio-enable-autospawn.service
```

At the time of writing there was a fault with the **lxplug-volumepulse** package which stops any of the menus on the Raspberry Pi desktop from being displayed. Remove the **lxplug-volumepulse** package with the following command:

```
$ sudo apt remove lxplug-volumepulse
```

### Installing OLED TFT driver software and libraries

If you are not using any OLED TFT display then skip this section.

**IMPORTANT NOTE:** The following devices and programs use **PIP3** to install the drivers.

- Olimex OLED or SSD1306 OLED TFTs
- Pimoroni Pirate Audio Radio
- SH1106 SPI OLED TFT

- Adafruit 2.5 and 3.5-inch TFT touchscreen
- Monochrome OLEDs which use the LUMA TFT driver
- The weather programs
- The IQaudIO controller

If you attempt to use **PIP3** (The name is a recursive acronym for "Pip Installs Packages") on the **Bookworm OS** it will fail with the following message:

```
To install Python packages system-wide, try apt install
python3-xyz, where xyz is the package you are trying to
install.

If you wish to install a non-Debian-packaged Python package,
create a virtual environment using python3 -m venv path/to/venv.
Then use path/to/venv/bin/python and path/to/venv/bin/pip. Make
sure you have python3-full installed.

For more information visit http://rptl.io/venv
```

To allow **pip3** to be used on **Bookworm OS** carry out the following instruction:

```
$ sudo mv /usr/lib/python3.11/EXTERNALLY-MANAGED
/usr/lib/python3.11/EXTERNALLY-MANAGED.old
```

Note that the above is all one line. There is no need to carry out the above instruction on system running **Bullseye OS**.

You will now be able to use **pip3** to install the driver software and programs shown in this manual, however you will see the following warning message:

```
WARNING: Running pip as the 'root' user can result in broken permissions and
conflicting behaviour with the system package manager. It is recommended to
use a virtual environment instead: https://pip.pypa.io/warnings/venv
```

The above message can be safely ignored for the installation procedures in this manual. However, if installing other author's software with **pip3**, you should seek advice from that author.

If using **Raspbian Lite** also install the **python3-pil** package:

```
$ sudo apt -y install python3-pil
```

### Install display drivers

Note: If you are installing the software for any of the products listed below then first carry out the instructions in the appropriate section. Do this before installing the radio software.

**Table 3 Display driver software installation**

Device	Section/Devices	Page
Pirate Radio	Installing Pimoroni Pirate Radio (pHat BEAT)	34
Pirate Audio	Installing the Pimoroni Pirate Audio	44

<b>SSD1306 0.9-inch OLED</b>	Installing the SSD1306 OLED driver libraries	49
<b>SH1106 SPI (joystick)</b>	Installing the SH1106 SPI OLED driver libraries	53
<b>MHS RPi displays</b>	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
<b>SH1106 1.3-inch OLED</b>	<b>Installing LUMA monochrome OLEDs</b>	59
<b>LUMA OLED devices</b>	SSD1306, SSD1309, SSD1325, SSD1331, SH1106, WS0010	59
<b>Waveshare TFTs</b>	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
<b>Grove LCD RGB</b>	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
<b>Adafruit TFT</b>	<b>Installing the Adafruit 2.5 and 3.5-inch TFT touchscreen</b>	54
<b>PiFace CAD</b>	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
<b>Waveshare SH1106 SPI</b>	Installing the SH1106 SPI OLED driver libraries	53

Installing the Radio Daemon on page 26.

### Installing LUMA monochrome OLEDs

The LUMA device driver (`luma_class.py`) interfaces monochrome OLEDs which are using SSD1306, SSD1309, SSD1325, SSD1331, SH1106 or WS0010 interface chips. In this release only SSD1206 and SH1106 OLEDs have been tested.

The devices test used 64x64 or 128x64 pixels resolution. Other resolutions may also work. If using any of the above devices then first install the necessary dependencies.

Install dependencies:

#### For Bookworm

```
$ sudo apt install libtiff5-dev
```

#### For Bullseye

```
$ sudo apt install libtiff5
```

#### For both Bookworm and Bullseye

```
$ sudo apt install python3-pip
$ sudo apt install python3 python3-pil libjpeg-dev zlib1g-dev libfreetype6-dev liblcms2-dev libopenjp2-7 -y
$ sudo -H pip3 install --upgrade luma.oled
$ sudo pip3 install pathlib
$ sudo pip3 install luma.core
```

For more information on the Luma driver see:

<https://luma-oled.readthedocs.io/en/latest/install.html>

<https://luma-oled.readthedocs.io/en/latest/api-documentation.html>

When the **configure\_radio.py** program is run and the LUMA OLED driver is selected the **display\_type** parameter in **/etc/radiod.conf** is set to **LUMA.<DEVICE>** where **<DEVICE>** is the chip being used.

For example, LUMA.SH1106 for OLEDs using the sh1106 chip

```
display_type=LUMA.SH1106
```

Now carry out the instructions shown in section *Radio* software pre-requisites

Upgrading from previous versions and other changes

If upgrading from version **6.x** or earlier it is necessary to replace your existing configuration file with the new one supplied in the upgrade. First take a note of any specific configuration items that are not standard or make a copy of your existing **/etc/radiod.conf** configuration file.

**Important:** The name format of Radio playlists has changed from version 7.3 onwards. See ***Creating and Maintaining Playlist files*** on page 169. **Also, in this release**, it is now possible to add, move and delete entries in the play list using any MPD client that is capable of doing so. See **Maintaining playlists using external MPD clients** on page 173.

**Note:** The location of the log file has changed from **/var/log/radio.log** to **/var/log/radiod/radio.log**. Also, its ownership has been changed from **root** to user **pi**. This has been done to reduce the number of programs that need to run with **sudo**.

**IMPORTANT:** In **Bookworm** the location for the boot configuration file **/boot/config.txt** has been changed to **/boot/firmware/config.txt**. Nearly all instructions in this manual refer to **/boot/firmware/config.txt** however if using **Bullseye** change the instruction to use **/boot/config.txt**. The installation scripts will choose the correct location.

### Installing the Music player daemon

If you haven't already done so upgrade the system packages as shown in **Error! Reference source not found.** on page **Error! Bookmark not defined..**

After reboot install the Music Player Daemon (mpd) and its client (mpc) along with the Python3, MPD and **python3-rpi.gpio** libraries. Note that you must install **python3-mpd** and not **python.mpd**.

```
$ sudo apt install mpd mpc python3-mpd python3-rpi.gpio
```

Answer yes 'y' when asked to continue.

If the installation says it cannot find the above packages, then re-run the update command to update the library list and retry installing the above packages.

```
$ sudo apt update
```

**IMPORTANT:** Unfortunately, the configuration file (**/etc/mpd.conf**) supplied with MPD version 0.22.6 in **Bookworm** and onwards is corrupt. It is necessary to correct this as show below.

Find the following entries in **/etc/mpd.conf** somewhere around line 232.

### Corrupt /etc/mpd.conf file

```
#audio_output {
#    type          "alsa"
#    name          "My ALSA device"
#    device        "hw:0,0"
#    mixer_type    "software"
##    mixer_device  "default"      # optional
##    mixer_control "PCM"         # optional
##    mixer_index   "0"           # optional
#}
```

Change the above entries to the following:

#### Corrected /etc/mpd.conf file

```
audio_output {
    type          "alsa"
    name          "My ALSA Device"
    device        "hw:0,0"
    mixer_type    "software"
#    mixer_device  "default"      # optional
#    mixer_control "PCM"         # optional
#    mixer_index   "0"           # optional
}
```

If installing on **Bookworm Lite** this will take quite a long time as there are a lot of software libraries to be installed.

At this stage there are no playlists configured so the music daemon won't play anything. The play lists are created when the **radiod** Debian Package is installed in the next section.

The version of **MPD** released with **Bookworm** is **0.22.6** which is reasonably up-to-date. You should install MPD version **0.22.6** first. When it is all working if you are having problems playing certain stations, it is well worth installing the latest version of MPD as shown in the section called *Compiling and installing the latest Music Player Daemon* on page 118.

#### Installing pulseaudio

The **pulseaudio** package may or may not needed to be installed. This has been different between various releases of the operating system. Currently the **pulseaudio** package causes a lot of problems with the radio software, in particular **espeak** and the Alsa sound system. If you do not intend to use espeak and need pulseaudio, for example, for blue-tooth devices or pHat BEAT then install it. The following table shows the options:

**Table 2 PulseAudio installation options**

Type of radio installation	Install pulseaudio
Using espeak	No
Pimoroni Pirate Radio with pHat BEAT	Yes (Installed by phatbeat)
Pimoroni Pirate Audio/mini-speaker	No
Adafruit speaker bonnet	No
LCD display radio	No unless using above DACs

<b>HDMI or touch-screen displays</b>	No unless using above DACs
<b>IQaudio, HiFiBerry or JustBoom DACs</b>	No
<b>Bluetooth sound devices</b>	Yes

To install **pulseaudio**:

```
$ sudo apt install pulseaudio
```

To remove **pulseaudio**:

```
$ sudo apt remove pulseaudio
```

If installing on Bookworm disable the `pulseaudio-enable-autospawn.service`.

```
$ sudo systemctl stop pulseaudio-enable-autospawn.service
```

At the time of writing there was a fault with the **lxplug-volumepulse** package which stops any of the menus on the Raspberry Pi desktop from being displayed. Remove the **lxplug-volumepulse** package with the following command:

```
$ sudo apt remove lxplug-volumepulse
```

## Installing OLED TFT driver software and libraries

If you are not using any OLED TFT display then skip this section.

**IMPORTANT NOTE:** The following devices and programs use **PIP3** to install the drivers.

- Olimex OLED or SSD1306 OLED TFTs
- Pimoroni Pirate Audio Radio
- SH1106 SPI OLED TFT
- Adafruit 2.5 and 3.5-inch TFT touchscreen
- Monochrome OLEDs which use the LUMA TFT driver
- The weather programs
- The IQaudio controller

If you attempt to use **PIP3** (The name is a recursive acronym for "Pip Installs Packages") on the **Bookworm OS** it will fail with the following message:

```
To install Python packages system-wide, try apt install
python3-xyz, where xyz is the package you are trying to
install.
```

```
If you wish to install a non-Debian-packaged Python package,
create a virtual environment using python3 -m venv path/to/venv.
Then use path/to/venv/bin/python and path/to/venv/bin/pip. Make
sure you have python3-full installed.
```

```
For more information visit http://rptl.io/venv
```

To allow **pip3** to be used on **Bookworm OS** carry out the following instruction:

```
$ sudo mv /usr/lib/python3.11/EXTERNALLY-MANAGED  
/usr/lib/python3.11/EXTERNALLY-MANAGED.old
```

Note that the above is all one line. There is no need to carry out the above instruction on system running **Bullseye OS**.

You will now be able to use **pip3** to install the driver software and programs shown in this manual, however you will see the following warning message:

```
WARNING: Running pip as the 'root' user can result in broken permissions and  
conflicting behaviour with the system package manager. It is recommended to  
use a virtual environment instead: https://pip.pypa.io/warnings/venv
```

The above message can be safely ignored for the installation procedures in this manual. However, if installing other author's software with **pip3**, you should seek advice from that author.

If using **Raspbian Lite** also install the **python3-pil** package:

```
$ sudo apt -y install python3-pil
```

### Install display drivers

Note: If you are installing the software for any of the products listed below then first carry out the instructions in the appropriate section. Do this before installing the radio software.

**Table 3 Display driver software installation**

Device	Section/Devices	Page
Pirate Radio	Installing Pimoroni Pirate Radio (pHat BEAT)	34
Pirate Audio	Installing the Pimoroni Pirate Audio	44
SSD1306 0.9-inch OLED	Installing the SSD1306 OLED driver libraries	49
SH1106 SPI (joystick)	Installing the SH1106 SPI OLED driver libraries	53
MHS RPi displays	Error! Reference source not found.	Error! Bookmark not defined.
SH1106 1.3-inch OLED	Installing LUMA monochrome OLEDs	59
LUMA OLED devices	SSD1306, SSD1309, SSD1325, SSD1331, SH1106, WS0010	59
Waveshare TFTs	Error! Reference source not found.	Error! Bookmark not defined.
Grove LCD RGB	Error! Reference source not found.	Error! Bookmark not defined.
Adafruit TFT	Installing the Adafruit 2.5 and 3.5-inch TFT touchscreen	54
PiFace CAD	Error! Reference source not found.	Error! Bookmark not defined.
Waveshare SH1106 SPI	Installing the SH1106 SPI OLED driver libraries	53

Installing the Radio Daemon on page 26.

## The GPIOconverter program

The **Raspberry Pi Model 5** was introduced at the end of 2023. It only works with **Raspberry Pi Bookworm OS** or later.

However, the biggest impact for most developers is that the standard **RPi.GPIO** input/output library does not work on the **Raspberry Pi model 5**. This is because the **RPi Model 5** now has a separate chip called **RP1** for controlling I/O including the pins on the GPIO header (**j8**). This means that hundreds of thousands of programs or maybe even millions of programs need to be modified to use one of the newer libraries such as **gpiod** or **Igpio**. The **RP1** chip also controls USB ports, Gigabyte Ethernet, MIPI Camera Controllers and Low Speed Peripherals compatible with earlier versions of the Raspberry Pi.

The **Raspberry Pi Internet Radio** is also such a program and would have meant a lot of work to convert all the GPIO routines to say **GPIOD** which does run on the RPi Model 5. So, it was decided to write a simple interface called **GPIOconverter** which converts **RPi GPIO** calls to one of the newer GPIO interfaces. This is a so-called **software shim**. See the following link for more information: [https://en.wikipedia.org/wiki/Shim\\_\(computing\)](https://en.wikipedia.org/wiki/Shim_(computing)). Although **GPIOD** was advocated as being the best way forward it was poorly documented and there didn't seem to be any examples of how to handle interrupts. The choice was made to use the excellent **python3-Igpio** library for the **GPIOconverter** software. The architecture of the interface is shown below:

**OUTPUT: User Program --> GPIO calls --> GPIOconverter --> LGPIO**  
**INPUT: LGPIO events --> GPIOconverter --> User Program**

The following illustration shows the location of the RP1 I/O chip.

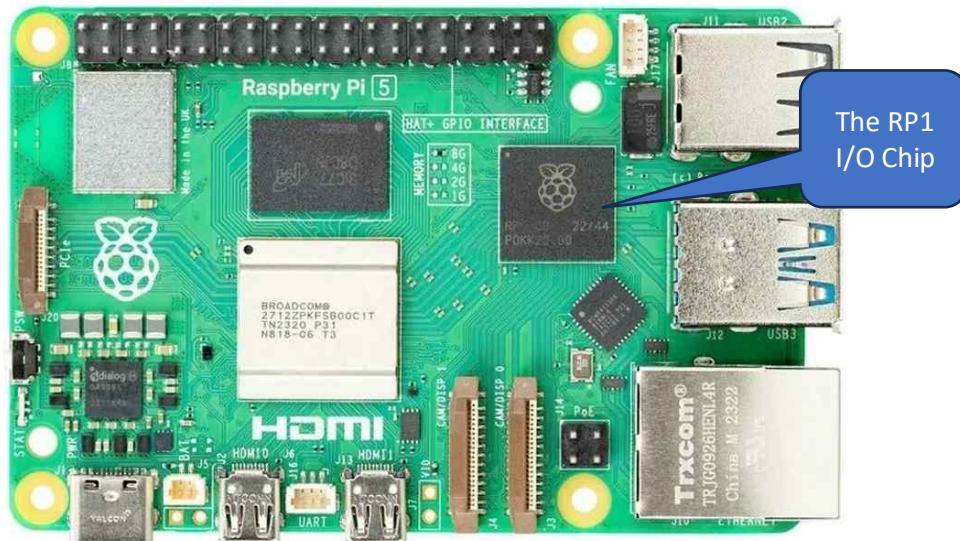


Figure 10 The Raspberry Pi Model 5 RP1 I/O chip

A further problem occurred with Bookworm OS in early 2024 when a new release of the kernel meant that RPi/GPIO no longer works on even earlier models such as the 3B or 4B. So, if installing

the radio software on the Bookworm OS the **GPIOconverter** shim is enabled during installation to correct this problem.

The **GPIOconverter** software called GPIO.py is located in the **/usr/share/radio/RPi** directory.

### Enabling GPIO.py

To enable **GPIOconverter** in the case of either a Raspberry Pi model 5 or if running on Bookworm only:

```
touch /usr/share/radio/RPi/__init__.py
```

This creates an empty file called **\_\_init\_\_.py**. The instruction above will cause the code in the **/usr/share/radio/** directory using the GPIO calls to see directory RPi as a package instead of using the system wide GPIO package.

For earlier models such as the 3B or 4 or if running on **Bullseye** disable the package

```
rm /usr/share/radio/RPi/__init__.py
```

### Inbuilt MPD HTTP streamer

The MPD daemon can be configured to use its own inbuilt streamer. However, this requires a special MPD client such as **gmpc** on the PC. It cannot be easily accessed from a Web browser. If you wish to use the inbuilt streamer see the following URL:

[http://mpd.wikia.com/wiki/Built-in\\_HTTP\\_streaming\\_part\\_2](http://mpd.wikia.com/wiki/Built-in_HTTP_streaming_part_2)

### Installing Icecast streamer

You may wish to play the output of the Radio through your PC speakers or a mobile device such as a tablet or telephone. This is possible to do this using **Icecast**. For more information on **Icecast** see the following Wikipedia article: <http://en.wikipedia.org/wiki/Icecast>.

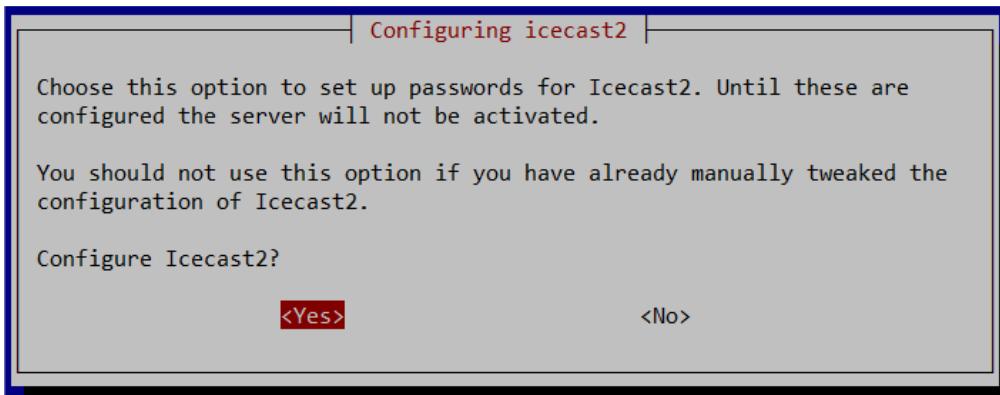
Please also refer to **Error! Reference source not found.** on page **Error! Bookmark not defined..**

### Installing Icecast

Install **icecast2** using the **install\_streaming.sh** script.

```
$ cd /usr/share/radio
$ sudo ./install_streaming.sh
Starting Icecast2 integration with the Music Player Daemon
The Icecast2 installation program will ask if you wish to configure Icecast2.
Answer 'yes' to this. Configure Icecast as follows:
Icecast2 hostname: localhost
Icecast2 source password: mympd
Icecast2 relay password: mympd
Icecast2 administration password: mympd
Continue y/n: y
```

Enter 'y' to continue. The Icecast2 installation program will ask if you wish to configure Icecast2:



**Figure 11 Configuring Icecast2**

Answer '**yes**' to this. Configure Icecast as follows:

Icecast2 hostname: **localhost** (or the hostname of the Raspberry Pi)

Icecast2 source password: **mympd**

Icecast2 relay password: **mympd**

Icecast2 administration password: **mympd**

It is important that you replace the default password 'hackme' with 'mympd'. The installation program continues configuration. The `icecast2` server will be started:

```

Done Configuring icecast2..
Processing triggers for libc-bin (2.19-18+deb8u6) ...
Processing triggers for systemd (215-17+deb8u5) ...
Configuring Icecast2
Copying /etc/icecast2/icecast.xml to /etc/icecast2/icecast.xml.orig

```

Check that the PI Radio stream (Output 2) is enabled

```

$ mpc outputs
Output 1 (My ALSA Device) is enabled
Output 2 (PI Radio MPD Stream) is enabled

```

If not enable it and restart **mpd** and **icecast2**. In this case it is output 2

```

$ mpc enable 2
$ sudo systemctl restart mpd.service icecast2.service

```

Check that MPD has established a connection with the `icecast2` server

```

$ netstat -tn | grep :8000
tcp        0      0 127.0.0.1:59096      127.0.0.1:8000          ESTABLISHED
tcp        0      0 127.0.0.1:8000      127.0.0.1:59096          ESTABLISHED

```

The `Icecast2 install_streaming.sh` script sets the **streaming\_on** parameter in `/etc/radiod.conf` to yes.

```
streaming_on=yes
```

It also adds the following configuration to `/etc/mpd.conf`.

```

# MPD Radio Stream
audio_output {
    type          "shout"
    name          "PI Radio MPD Stream"
    description   "MPD stream on Raspberry Pi Radio"
    host          "localhost"
    port          "8000"
    mount         "/mpd"
    password      "mympd"
    bitrate       "128"
    format        "44100:16:2"
    encoding      "mp3"
}

```

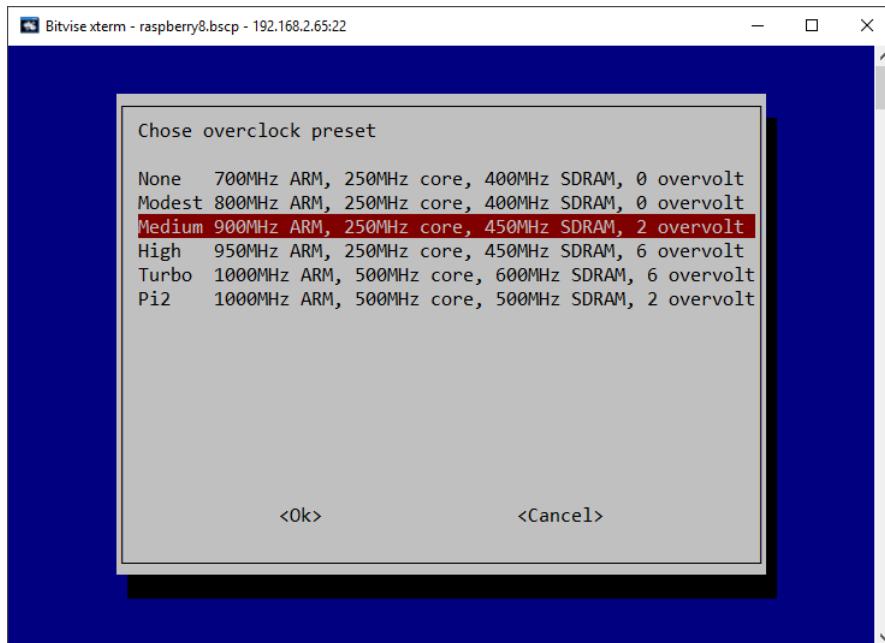
This completes the installation of Icecast2 however you may need to configure the clock speed if your Raspberry Pi is an earlier version (See *Overclocking older Raspberry PI's* on page 67)

Now go to *Icecast2 Operation* on page 272.

### Overclocking older Raspberry PI's

With older versions of the Raspberry Pi it will almost certainly be necessary to over-clock to handle Icecast2 streaming using the **raspi-config** program. Medium over-clocking seems to be sufficient. Note that the later versions of the Raspberry Pi cannot be overclocked and are fast enough anyhow.

Run **raspi-config**. Select option 'Overclock'. After a warning screen about over-clocking has been displayed, the following screen will be displayed:



**Figure 12 Over-clocking the Raspberry PI**

Select 'Medium' to start with. Reboot the Raspberry PI when prompted. Re-test the radio with streaming switched on.

## Software maintenance

### Apply patches to the radio software

From version 7.7 onwards patches will not be normally used, instead a complete new build will be issued. Patches will be announced on X (Formerly Twitter) at: [https://twitter.com/bob\\_rathbone](https://twitter.com/bob_rathbone) or [https://x.com/bob\\_rathbone](https://x.com/bob_rathbone) and should always be applied to the current software release. For technical reasons the BR Web site will still use the original blue Twitter logo for the time being.

Follow this X (Formerly Twitter) feed for announcements about new patches. Patches can be viewed at [http://www.bobrathbone.com/raspberrypi/pi\\_internet\\_radio.html](http://www.bobrathbone.com/raspberrypi/pi_internet_radio.html)

Patches take the form:

**radiod-patch-<version>-<patch-number>.tar.gz**

Where; <version> is the package version number, 7.9 in this case.

<patch-number> is the patch number from 1 onwards.

For example: **radiod-patch-7.9-1.tar.gz**



Always check for the latest patches on the Web site. They will not be listed in this document.

To apply this patch (if it exists) run the following commands:

```
$ cd /usr/share/radio  
$ wget http://www.bobrathbone.com/raspberrypi/packages/radiod-patch-7.9-  
1.tar.gz  
$ tar -xvf radiod-patch-7.9-1.tar.gz
```

Modify the above command as necessary.

To see the details of the patch run the following command:

```
$ cat README.patch
```

Restart the radio software to activate the patches.



Any patch greater than 1 will also include all previous patches where relevant so it is not necessary to install previous patches. So for example patch 3 will include patches 1 and 2.



Do not apply any patches from a previous version of the software to the current version. This will most likely cause the current software to malfunction.

All relevant patches in a particular version will normally be included in the next version of the software.

## Keeping the radio software up-to-date



The Radio software may be updated from time to time especially if a new version of the operating system is released or a new feature is added to the software. To keep up to date follow the author on **X** (Formerly Twitter) at: [https://twitter.com/bob\\_rathbone](https://twitter.com/bob_rathbone) or [https://x.com/bob\\_rathbone](https://x.com/bob_rathbone)

For technical reasons the BR Web site will continue to use the original blue Twitter logo for the time being.

## Backing up the SD card

Having spent a lot of time and effort installing and configuring the Radio software it is a very good idea to create a backup of the SD card should it ever become corrupted. There are various ways of doing this. Search the Web for “Raspberry Pi backup SD card”

One of the easiest ways of backing up the SD card on a Windows machine using Windows Disk Imager (Win32DiskImager) described in the following link.

<https://www.raspberrypi.org/forums/viewtopic.php?t=26463>

This allows you to create a copy of the SD card in an image (.img) file. This can then be compressed using **winzip/Zzip** or any other zip utility to reduce the space on disk.

## Chapter 4 - Internet Security

<b>Contents chapter 4</b>	<b>Page</b>
Some golden Internet Security rules	71
SSH keys installation	72
Generate a client key	73
Add client public key to Raspberry Pi authorised keys	73
Firewall set-up	73



This is a section that probably will not concern most people as their Raspberry Pi is not exposed to the internet. However, with more and more cases of such devices such as Web cams and other Internet connected devices being hacked by unscrupulous hackers, Internet Security is an aspect of home computing that must be taken seriously. These incursions can be used to mount Phishing (harvesting bank details etc.) or Distributed Denial of Service attacks (**DDOS**) on the wider community as a whole. More and more Internet providers are choosing to block compromised user's systems from access to the Internet until the infection is removed.

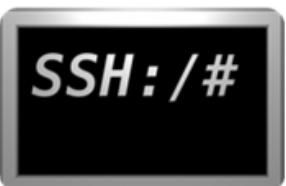
### Some golden Internet Security rules



Always change the user **pi** password from the system installation default or use another log in name such as 'bobrath' (Only possible since version 7.5 of the software). When installed the password for user **pi** is 'raspberry'. It will be the first password that will be attempted by a hacker. The password can easily be changed using the **raspi-config** program (See **Error! Reference source not found.** on page **Error! Bookmark not defined.**). The user **pi** is very dangerous if hacked as with the command **sudo bash** the hacker then has user root privileges and can do anything they want including installing **Phishing** or **DDOS** software. Don't give them the chance!



Never ever use insecure protocols/programs such as **Telnet**, **Rexec** or **FTP** across the Internet. The problem with all such programs is that the login username and password are unencrypted and can be discovered by a hacker using eavesdropping software. The use of such software to access the Raspberry Pi will attract hackers like flies around a honey-pot.



If access to the Raspberry Pi across a network is required (for example a headless RPi) then use Secure Shell (**SSH**) for terminal access and Secure Copy (**SCP**) for file transfer. On the latest releases of Raspbian, **SSH** is disabled for security reasons. For extra security use **SSH keys** (explained later)



Install **firewall** software such as **fail2ban**. The **Raspbian Bullseye** operating system has a firewall called **iptables** which can be configured to block or allow access to specific ports from a specific **IP** address or range. The **fail2ban** software is an enhancement to **iptables** which monitors certain ports for hacking attempts and adds a blocking rule to the **iptables** configuration. More on this later. The fail2ban software is a good defence against so-called brute-force dictionary attacks.

## SSH keys installation

### Raspberry Pi ssh keys

If using **SSH** across the Internet, changing the password for user **pi** will afford some limited protection. However, a hacker can still access the system with SSH and try to log in as user **pi**. They can still try (often using software) to try and guess the password. By using SSH keys a greater level of protection is afforded as person logging in must be in possession of the SSH keys.

Log in as user **pi** and then run the **ssh-keygen** program. Just press enter when asked any questions.

```
$ ssh-keygen -t rsa -C "raspberrypi"
Generating public/private rsa key pair.
Enter file in which to save the key (/home/pi/.ssh/id_rsa):
Created directory '/home/pi/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/pi/.ssh/id_rsa.
Your public key has been saved in /home/pi/.ssh/id_rsa.pub.
The key fingerprint is:
c0:54:96:d9:2d:a5:d0:d0:7c:68:91:8a:5f:e2:a5:9d pi@pixelpi
The key's randomart image is:
+---[RSA 2048]---+
| ..=O.*. |
| o .o.X.o |
| .o. o.o |
| ..o o |
| oS* . |
| + E |
| |
| |
+-----+
```

Two keys are generated, one public and one private in the **.ssh** directory. The key **id\_rsa.pub** is the public key. The **id\_rsa** file is the private key.

```
$ ls -la .ssh/
total 16
drwx----- 2 pi pi 4096 Feb 16 12:40 .
drwxr-xr-x 19 pi pi 4096 Feb 16 12:40 ..
-rw----- 1 pi pi 1675 Feb 16 12:40 id_rsa
-rw-r--r-- 1 pi pi 392 Feb 16 12:40 id_rsa.pub
```

## Generate a client key

It is also necessary to generate SSH keys on Typically **Putty** and **Bitvise** are a very popular choice for **SSH** clients. There is already so much documentation on the Internet on how to generate SSH keys for both **Putty** and **Bitvise** that it is not repeated here. Search the Internet for instructions.

## Add client public key to Raspberry Pi authorised keys

On the Raspberry Pi create or edit the **/home/pi/.ssh/authorized\_keys**.

```
$ cd /home/pi/.ssh/  
$ vi authorized_keys
```

Paste or copy the Public key created on the PC to the **authorized\_keys** file. (Some output in the following text omitted).

```
ssh-rsa  
AAAAAB3NzaC1yc2EAAAQEAuX+NEQoQECPN2d+Lu+qL2exMT/ICYbrNax6DVWBtKGzTxFOb  
:  
LeiaFbI3tWyi+ZPXg8Swhr1OaPN612E/fnAQPbG12S+YMtcIXknNiwVGL8RB3D8N/Q== rsa-  
key-20170216
```

Finally connect to the Raspberry Pi from the PC client using the **publickey** method. If this works OK disable password login method. Edit `vi /etc/ssh/sshd_config` and disable PAM and Password authentication.

```
PasswordAuthentication no  
:  
UsePAM no
```

## Firewall set-up

Linux has a firewall facility built-in to the kernel called **iptables**. Rules may be added to **iptables** to allow or deny access to system services as required. However, **iptables** is static and has to be configured with any new rules. The **fail2ban** program enhances **iptables** by dynamically adding rules as required. For example, if a hacker attempts five unsuccessful logins using SSH then **fail2ban** blocks that IP address from connecting to the SSH port by adding a blocking rule to **iptables**. The following commands install and enable **fail2ban**.

```
$ sudo apt install fail2ban  
$ sudo systemctl enable fail2ban
```

Below is an example of a blocked host (195.22.126.242)

```
$ sudo iptables -L  
Chain INPUT (policy ACCEPT)  
:  
Chain FORWARD (policy ACCEPT)  
:  
Chain OUTPUT (policy ACCEPT)  
:  
Chain fail2ban-ssh (2 references)  
target prot opt source destination
```

```
REJECT    all    --  195.22.126.242 anywhere reject-with icmp-port-unreachable
```

However, rules added by fail2ban will be lost if the Raspberry Pi is rebooted. An additional package called **iptables-persistent** can be added so that rules added by fail2ban can be made permanent. The following command allows installation.

```
$ sudo apt install iptables-persistent
```

The following command can be used to make the **iptables** rules persistent after a reboot. Run the following command before rebooting.

```
$ iptables-save
```

Search the internet for more information on **iptables** and **fail2ban**.

## Chapter 5 - Frequently asked questions (FAQs)

<b>Contents chapter 5</b>	<b>Page</b>
What is the login name and password?	76
How do I change the order of the radio stations?	76
Why are some station names not being displayed in the Web interface?	76
Why doesn't the Web interface display URLs until a station is selected?	77
Why are music tracks played randomly when loaded?	77
Can the volume be displayed as Volume nn instead of as blocks?	77
Why do I see a station number on LCD line 3?	77
Is it possible to change the date format?	77
Is there a pause & resume function?	77
Is there a reboot or shutdown option?	77
Why do I see a different station name from the one in the playlist?	77
What Rotary Encoder can I use for this project?	78
Can this code or documentation be re-used in other projects?	78
Can I use an Electronic Ink display?	78
Can you make or sell me a radio?	78
Can you recommend the hardware for my project?	78
Can you make a change to the software for my project?	79
What if I want to try different hardware?	79
How do I configure network roaming	79

## What is the login name and password?

The default login name is: pi

The default password is: raspberry

You should change this at the earliest opportunity. See *Chapter 4 - Internet Security* on page 70.

## How do I change the order of the radio stations?

Playlists are loaded by the radio daemon in alphabetic order using the playlist name.

When loading an individual playlist, MPD loads the stations in the order that they are defined in each individual playlist in the **/var/lib/radiod/stationlist** file.

It helps greatly to group stations of the same type into a single playlist. For example, group all BBC radio stations into a single playlist.

The only way to get all of the radio stations in the order that you define them is to define a single playlist, for example **myplaylist**:

```
(myplaylist)
#
# United Kingdom
[BBC Radio 1] http://bbc.co.uk/radio/listen/live/r1.aspx
[BBC Radio 2] http://bbc.co.uk/radio/listen/live/r2.aspx
[BBC Radio 3] http://bbc.co.uk/radio/listen/live/r3.aspx
:
:
[RAIradio3] http://www.listenlive.eu/rai3.m3u
```

This will produce a single playlist called **myplaylist.m3u** with the stations loading in the order that they have been defined in the **/var/lib/radiod/stationlist** file. Make sure there are no blank lines between station definitions otherwise this terminates the playlist. All remaining stations will end up in their own single playlist file.

## Why are some station names not being displayed in the Web interface?

The reason for this is that some stations don't send the name with the stream. If you run the **mpc playlists** command you will see that some radio stations show only the station URL and not the name:

```
$ mpc playlist
RAIradio2
:
BBC Radio 4 extra
http://icestreaming.rai.it/1.mp3
BBC Radio 3
BBC Radio 6
BBC Radio 5 live
```

The only way around this is to complain directly to the radio station to ask them to amend their stream to include the station name and title details. The only way reason that the station name is seen with the radio program is that it picks up the names out of the station list file.

The snoopy Web interface can't do this however.

## Why doesn't the Web interface display URLs until a station is selected?

When the Snoopy Web interface is loaded it loads the playlists found in the `/var/lib/mpd/playlists` directory. Snoopy displays the URLs but doesn't appear to use any titles defined in the playlists. It only displays the radio station information (if present) once it starts streaming from a particular radio station. Snoopy is third party software over which this author has no control.

## Why are music tracks played randomly when loaded?

This is the default behaviour when the music library is loaded. Random always defaults to "off" when the radio is selected. However, when the media library is selected the value stored in the `/var/lib/radiod/random` file is used which is either 1 (Random on) or 0 (Random off). This value can be changed in the selection menu by toggling "Random on/off". So, the radio software will remember the desired random setting for the music library when it is restarted.

## Can the volume be displayed as Volume nn instead of as blocks?

Yes, it can. Volume is displayed by default as blocks on character LCD's or as a volume ribbon on OLED devices. However, it can also be configured to display as a value "Volume *nn*" where *nn* is 1 to 100. This can be changed as shown in section **Configuring the volume display** on page 132

## Why do I see a station number on LCD line 3?

If no song information is available then the station playlist number followed by the stream speed. In the following example Radio 1 is not transmitting any song information. It is number 37 in the play list. The speed from the stream is 96 Kilobit. The displayed stream speed can also continuously change for some radio stations where the stream speed is variable.

12:01 23/08/2015  
Radio1  
Station 37 96K  
Volume 75

## Is it possible to change the date format?

Yes. Please see the section called *Changing the date format* on page 129.

## Is there a pause & resume function?

Yes, but it is called mute and un-mute. The mute function also stops or pauses the MPD player. If playing a radio station, a "stop" command is carried out. If playing a media track a "pause" is carried out. The reason these are different is that media may be safely paused but in the case of a radio station pausing causes buffering and jumping to the next station when the radio resumes normal playing.

## Is there a reboot or shutdown option?

There is only a shutdown option. Hold the menu button in for at least three seconds. The radio will stop and should display "Radio stopped" on the display. Wait ten seconds and then power off the Raspberry Pi. Powering back on achieves the same effect as a reboot. Also see **Error! Reference source not found.** on page **Error! Bookmark not defined..**

## Why do I see a different station name from the one in the playlist?

The station information displayed comes from the stream itself. The name entered in the playlist definition is only used in the search function. This was a design decision because the station

information is only available once a particular radio station is selected so only the playlist name can be initially used. If the station transmits the station title this is used instead.

Run the `display_current.py` program to see all the information that comes from the stream (It is quite interesting).

### What Rotary Encoder can I use for this project?

It is possible to use any so called “Incremental Rotary Encoder” such as the COM-09117 12-step rotary encoders from sparkfun.com or the KY-040 encoder. An incremental rotary encoder provides cyclical outputs (only) when the encoder is rotated. The other type is an “absolute rotary encoder” and maintains position information even when switched off. These do not work with the radio software.

The cheaper smaller rotary encoders are usually incremental encoders. Absolute rotary encoders are usually bigger and more expensive as they house more electronics. If unfortunately, the seller doesn't provide a specification then there is a small risk that they may not run with this software.



**Note:** Not all manufacturers' rotary encoders will work with this project. If they work then fine if not regrettably you will need to purchase the recommended rotary encoders. You can also try the alternative rotary class which may just work with your encoders.

### Can this code or documentation be re-used in other projects?

Yes, it can. You can even use it commercially, provided that you do so under the terms of the licence distributed with this package. The software and documentation for this project is released under the GNU General Public Licence and may be re-used in other projects. See **Error! Reference source not found.** on page **Error! Bookmark not defined..**

You do not need to ask permission to re-use this code or documentation as it is already permitted in the licenses.

### Can I use an Electronic Ink display?

The answer is not at the moment. There are a number of electronic ink displays on the market such as the Pimoroni Inky pHat and the Waveshare ink display. These displays appear to only suitable for static display of information and do not handle dynamically changing screens well. However, looking at on-line demo's the Waveshare device, it can handle partial screen updates however continuous updates appeared to damage the screen after a time. Maybe this will improve over time.

### Can you make or sell me a radio?

The answer is NO. I regret I do not make any radios for sale nor do I make radios for other people. The whole purpose of this project is to help hobby and enthusiasts to learn computing. If you want one of the radios shown in this manual you need to build it yourself.

### Can you recommend the hardware for my project?

No - The author never makes recommendations as to which hardware someone should use for their project. There is a detailed overview, in this manual, of the possible solutions which are supported by the software.

No one can then ever say “well you recommended it” if there is a problem with the hardware selection. The choice of hardware is totally down to the constructor as only they know their exact

requirements. Also, the author wishes to remain neutral when it comes to recommending any manufacturer's hardware offering.

### Can you make a change to the software for my project?

Many of the ideas and features in this project have come from the community of enthusiast's and constructors. Their ideas and contributions have greatly enhanced both the usability and functionality of the project. These suggestions are always welcome and where possible, sooner or later, find their way into the product but not always.

What seems to be a simple change will mean months of testing of a new release. There are at least twelve basic designs plus variations all of which must be validated before the new software can be released.

However, if a constructor wishes to make software changes, help is available although this does not extend to either debugging their software or teaching Python.

### What if I want to try different hardware?

Often a constructor comes across other hardware other than that published in this manual and ask whether or not it can be used. This often because it is cheaper, more locally available or better than the current options. The author will give an opinion as to whether or not it might work in their situation but will not take responsibility for the choice made for any given project. The choice of hardware is ultimately the responsibility of the constructor.

Sometimes there are cheaper versions a popular piece of hardware but prove to be disappointing. One such case was a cheap Chinese version of Adafruit's RGB backplate. This has an RGB backlight. The cheap alternative simply put an RGB LED on the board which did not act as a backlight but rather as an indicator LED. Other cases of cheap alternatives to using the official Raspberry Pi 7-inch touch screen have also displayed start-up problems. So, caution is advised.

Sometimes the suggested hardware would require new or modified software and the author will consider this depending upon the merits of introducing such hardware and has done so many times in the past, thus enhancing the project.

Again, such suggestions and ideas are always welcome but do not guarantee that these suggestions will be supported by the Radio software.

### How do I configure network roaming between different locations?

See the section called *Configuring network roaming on a Raspberry Pi* on page 115

## Chapter 6 Advanced topics

<b>Contents chapter 6</b>	<b>Page</b>
Booting from a USB drive or stick	81
Repairing a non-bootable SD card	99
Building your own package	102
Using PWM GPIOs for sound output on a Pi Zero	103
Fitting a Pimoroni On-Off Shim power switch	103
Adding a Real Time Clock (RTC)	105
Accessing the Raspberry Pi from a PC or MAC using VNC	106
Installing an earlier release of Bookworm OS	113
Configuring network roaming on a Raspberry Pi	115
X-Windows configuration for the radio	119

## Booting from a USB drive or stick

The Raspberry Pi is normally booted from an SD card; however, it is also possible to boot from a USB 3.0 disk drive or stick. The drive can be either a USB drive with a spindle or a Solid-State Drive (SSD) or a USB stick. This procedure works with Raspberry Pi's with USB 2.x or 3.x ports such as models 3B, 3B+ or 4B+. The Pi 400 and 4B may already be USB boot enabled and can usually boot from a USB drive without modifying the firmware.

Before starting this procedure, it is important to understand that there are two types of USB drive protocols of importance with respect to the driver software/hardware/firmware:

1. USB drives that support the **USB Attached SCSI (UAS)** protocol
2. USB drives that support USB Mass Storage **Bulk-Only Transport (BOT)** protocol

**UAS** drivers generally provide faster transfers when compared to the older **BOT** protocol drivers. When used with an SSD, **UAS** is considerably faster than **BOT** for random reads and writes. For UAS to be enabled the USB Drive, hardware controller, USB driver software and firmware must all support **UAS** for it to work.



Regrettably **Bookworm** does not appear to be properly supporting non-UAS drives and this procedure currently only works with **Bullseye** for these types of USB drive. This may be a configuration issue and is still being investigated by the author.



Figure 13 Raspberry Pi booting from an Intenso USB 3.0 2.5-inch disk drive

The advantages of booting from a USB disk drive are:

- A USB disk drive is a lot less susceptible to corruption than a SD card
- A USB disk drive generally has greater capacity than a SD card for media files and the like
- USB disk drives are a lot faster than a SD card. SSD disks are even faster but limited to USB 3.0 (or 2.x) speeds.

Disadvantages:

- Drives using a spindle are theoretically prone mechanical failure, however modern solid-state drives or USB sticks don't suffer from this problem.

To make a bootable USB disk drive you will need:

1. A Raspberry Pi with USB 3.0 ports or newer RPi's (3B+) with USB 2.0 ports.
2. A spare micro-SD Card
3. A USB disk drive or stick (For example a 2.5-inch USB 3.0 SSD)
4. Raspberry Pi imager software



Note: After carrying out this routine it will still be possible to boot from an SD card. Simply power off, disconnect the USB disk drive, insert a SD card with an OS image and power back on. The Raspberry Pi will now boot from the SD card.



Note: During testing USB sticks were found to be just as vulnerable as SD-cards. It is better to use USB 3.0 2.5-inch SSD drives.

### Pi Zero W booting from a USB drive

Booting from a USB drive on a Pi Zero W is not supported. The Pi Zero W uses a BCM2835, which contains the old SD card-only bootloader. Secondly the USB ports have very limited power availability (Standard 600 mA).

### USB Disk Drive power consumption

A small USB disk drive with spindle (250 GByte) takes about 260 Milliwatts. The allowed USB power load for the Raspberry Pi is 1.2A across all four USB ports so there is plenty of power available for other USB devices. Solid State Drives (SSD) take significantly less power and do not suffer from mechanical failure. When the Raspberry Pi is shut down, the power is removed from the USB ports and the disk drive is completely powered off.

### Enabling the Raspberry Pi to boot from a USB drive

Making a Raspberry Pi boot from a USB drive or stick depends upon the Raspberry Pi model.

- Method 1 - Models with USB 2.0 ports such as the RPi model 3B+
- Method 2 - Models with USB 3.x ports such as the RPi model 4B

#### *Method 1 - Enabling USB booting on models with USB 2.0 ports*

This method Raspberry Pi's which have USB 2.0 ports such as the RPi model 3B+.

To enable the Raspberry Pi's USB boot mode, it is necessary to add a configuration option to the **/boot/firmware/config.txt** file. When the RPi is rebooted, this will set a bit in the Raspberry Pi's OTP (One Time Programmable) memory, allowing the device to be booted from a USB mass storage device. After that, we won't need the SD card anymore.

To enable USB boot mode, open a terminal session and run the following command:

```
$ echo program_usb_boot_mode=1 | sudo tee -a /boot/firmware/config.txt
```

This adds the config option **program\_usb\_boot\_mode=1** to the end of **/boot/firmware/config.txt** file. Reboot the Pi with the SD card still inserted.

```
$ sudo reboot
```

After the reboot, check that boot mode has been enabled with the following command:

```
$ vcgencmd otp_dump | grep 17  
17:3020000a
```



Note: If you are going to use your microSD card with a different Raspberry Pi later on, you might wish to remove the **program\_usb\_boot\_mode=1** line from **config.txt** file, so that the boot mode won't be programmed to that device as well. Edit the file using the command **sudo nano /boot/firmware/config.txt**. and remove the above line.

Now power off and proceed to the section called Now carry out the instructions shown in section **Radio software pre-requisites**

#### Upgrading from previous versions and other changes

If upgrading from version **6.x** or earlier it is necessary to replace your existing configuration file with the new one supplied in the upgrade. First take a note of any specific configuration items that are not standard or make a copy of your existing **/etc/radiod.conf** configuration file.

**Important:** The name format of Radio playlists has changed from version 7.3 onwards. See *Creating and Maintaining Playlist files* on page 169. **Also, in this release**, it is now possible to add, move and delete entries in the play list using any MPD client that is capable of doing so. See *Maintaining playlists using external MPD clients* on page 173.

**Note:** The location of the log file has changed from **/var/log/radio.log** to **/var/log/radiod/radio.log**. Also, its ownership has been changed from **root** to user **pi**. This has been done to reduce the number of programs that need to run with **sudo**.

**IMPORTANT:** In **Bookworm** the location for the boot configuration file **/boot/config.txt** has been changed to **/boot/firmware/config.txt**. Nearly all instructions in this manual refer to **/boot/firmware/config.txt** however if using **Bullseye** change the instruction to use **/boot/config.txt**. The installation scripts will choose the correct location.

#### Installing the Music player daemon

If you haven't already done so upgrade the system packages as shown in **Error! Reference source not found.** on page **Error! Bookmark not defined..**

After reboot install the Music Player Daemon (mpd) and its client (mpc) along with the Python3, MPD and **python3-rpi.gpio** libraries. Note that you must install **python3-mpd** and not **python.mpd**.

```
$ sudo apt install mpd mpc python3-mpd python3-rpi.gpio
```

Answer yes 'y' when asked to continue.

If the installation says it cannot find the above packages, then re-run the update command to update the library list and retry installing the above packages.

```
$ sudo apt update
```

**IMPORTANT:** Unfortunately, the configuration file (**/etc/mpd.conf**) supplied with MPD version 0.22.6 in **Bookworm** and onwards is corrupt. It is necessary to correct this as shown below.

Find the following entries in **/etc/mpd.conf** somewhere around line 232.

#### Corrupt /etc/mpd.conf file

```
#audio_output {
#    type          "alsa"
#    name          "My ALSA device"
#    device        "hw:0,0"
#    mixer_type    "software"
##    mixer_device  "default"      # optional
##    mixer_control "PCM"         # optional
##    mixer_index   "0"           # optional
#}
```

Change the above entries to the following:

#### Corrected /etc/mpd.conf file

```
audio_output {
    type          "alsa"
    name          "My ALSA Device"
    device        "hw:0,0"
    mixer_type    "software"
#    mixer_device  "default"      # optional
#    mixer_control "PCM"         # optional
#    mixer_index   "0"           # optional
}
```

If installing on **Bookworm Lite** this will take quite a long time as there are a lot of software libraries to be installed.

At this stage there are no playlists configured so the music daemon won't play anything. The play lists are created when the **radiod** Debian Package is installed in the next section.

The version of **MPD** released with **Bookworm** is **0.22.6** which is reasonably up-to-date. You should install MPD version **0.22.6** first. When it is all working if you are having problems playing certain stations, it is well worth installing the latest version of MPD as shown in the section called *Compiling and installing the latest Music Player Daemon* on page 118.

#### Installing pulseaudio

The **pulseaudio** package may or may not need to be installed. This has been different between various releases of the operating system. Currently the **pulseaudio** package causes a lot of problems with the radio software, in particular **espeak** and the Alsa sound system. If you do not intend to use espeak and need pulseaudio, for example, for blue-tooth devices or pHat BEAT then install it. The following table shows the options:

**Table 2 PulseAudio installation options**

Type of radio installation	Install pulseaudio
Using espeak	No

<b>Pimoroni Pirate Radio with pHat BEAT</b>	Yes (Installed by phatbeat)
<b>Pimoroni Pirate Audio/mini-speaker</b>	No
<b>Adafruit speaker bonnet</b>	No
<b>LCD display radio</b>	No unless using above DACs
<b>HDMI or touch-screen displays</b>	No unless using above DACs
<b>IQaudIO, HiFiBerry or JustBoom DACs</b>	No
<b>Bluetooth sound devices</b>	Yes

To install **pulseaudio**:

```
$ sudo apt install pulseaudio
```

To remove **pulseaudio**:

```
$ sudo apt remove pulseaudio
```

If installing on Bookworm disable the `pulseaudio-enable-autospawn.service`.

```
$ sudo systemctl stop pulseaudio-enable-autospawn.service
```

At the time of writing there was a fault with the **lxplug-volumepulse** package which stops any of the menus on the Raspberry Pi desktop from being displayed. Remove the **lxplug-volumepulse** package with the following command:

```
$ sudo apt remove lxplug-volumepulse
```

## Installing OLED TFT driver software and libraries

If you are not using any OLED TFT display then skip this section.

**IMPORTANT NOTE:** The following devices and programs use **PIP3** to install the drivers.

- Olimex OLED or SSD1306 OLED TFTs
- Pimoroni Pirate Audio Radio
- SH1106 SPI OLED TFT
- Adafruit 2.5 and 3.5-inch TFT touchscreen
- Monochrome OLEDs which use the LUMA TFT driver
- The weather programs
- The IQaudIO controller

If you attempt to use **PIP3** (The name is a recursive acronym for "Pip Installs Packages") on the **Bookworm OS** it will fail with the following message:

```
To install Python packages system-wide, try apt install
python3-xyz, where xyz is the package you are trying to
install.
```

```
If you wish to install a non-Debian-packaged Python package,
create a virtual environment using python3 -m venv path/to/venv.
Then use path/to/venv/bin/python and path/to/venv/bin/pip. Make
sure you have python3-full installed.
```

For more information visit <http://rptl.io/venv>

To allow **pip3** to be used on **Bookworm OS** carry out the following instruction:

```
$ sudo mv /usr/lib/python3.11/EXTERNALLY-MANAGED  
/usr/lib/python3.11/EXTERNALLY-MANAGED.old
```

Note that the above is all one line. There is no need to carry out the above instruction on system running **Bullseye OS**.

You will now be able to use **pip3** to install the driver software and programs shown in this manual, however you will see the following warning message:

```
WARNING: Running pip as the 'root' user can result in broken permissions and  
conflicting behaviour with the system package manager. It is recommended to  
use a virtual environment instead: https://pip.pypa.io/warnings/venv
```

The above message can be safely ignored for the installation procedures in this manual. However, if installing other author's software with **pip3**, you should seek advice from that author.

If using **Raspbian Lite** also install the **python3-pil** package:

```
$ sudo apt -y install python3-pil
```

### Install display drivers

Note: If you are installing the software for any of the products listed below then first carry out the instructions in the appropriate section. Do this before installing the radio software.

**Table 3 Display driver software installation**

Device	Section/Devices	Page
Pirate Radio	<b>Installing Pimoroni Pirate Radio (pHat BEAT)</b>	<b>34</b>
Pirate Audio	<b>Installing the Pimoroni Pirate Audio</b>	<b>44</b>
SSD1306 0.9-inch OLED	<b>Installing the SSD1306 OLED driver libraries</b>	<b>49</b>
SH1106 SPI (joystick)	<b>Installing the SH1106 SPI OLED driver libraries</b>	<b>53</b>
MHS RPi displays	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
SH1106 1.3-inch OLED	<b>Installing LUMA monochrome OLEDs</b>	<b>59</b>
LUMA OLED devices	SSD1306, SSD1309, SSD1325, SSD1331, SH1106, WS0010	<b>59</b>
Waveshare TFTs	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
Grove LCD RGB	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
Adafruit TFT	<b>Installing the Adafruit 2.5 and 3.5-inch TFT touchscreen</b>	<b>54</b>

PiFace CAD	Error! Reference source not found.	Error! Bookmark not defined.
Waveshare SH1106 SPI	Installing the SH1106 SPI OLED driver libraries	53

Installing the Radio Daemon on page 26.

#### *Method 2 - Enabling USB booting on models with USB 3.x ports*

This method Raspberry Pi's which have two USB 3.0 ports and two USB 2.x such as the RPi model 4B or Pi 400. The model 4B and Pi 400 running the latest firmware come already enabled to boot from USB devices. This can be checked with the following command:

```
$ vcgencmd otp_dump | grep 17
17:3020000a
```

The value shown must be **17:3020000a**. If this is the case then proceed to the section called Now carry out the instructions shown in section **Radio software pre-requisites**

#### **Upgrading from previous versions and other changes**

If upgrading from version **6.x** or earlier it is necessary to replace your existing configuration file with the new one supplied in the upgrade. First take a note of any specific configuration items that are not standard or make a copy of your existing **/etc/radiod.conf** configuration file.

**Important:** The name format of Radio playlists has changed from version 7.3 onwards.

See **Creating and Maintaining Playlist files** on page 169. Also, in this release, it is now possible to add, move and delete entries in the play list using any MPD client that is capable of doing so. See Maintaining playlists using external MPD clients **on page 173**.

**Note:** The location of the log file has changed from **/var/log/radio.log** to **/var/log/radiod/radio.log**. Also, its ownership has been changed from **root** to user **pi**. This has been done to reduce the number of programs that need to run with **sudo**.

**IMPORTANT:** In **Bookworm** the location for the boot configuration file **/boot/config.txt** has been changed to **/boot/firmware/config.txt**. Nearly all instructions in this manual refer to **/boot/firmware/config.txt** however if using **Bullseye** change the instruction to use **/boot/config.txt**. The installation scripts will choose the correct location.

#### *Installing the Music player daemon*

If you haven't already done so upgrade the system packages as shown in **Error! Reference source not found.** on page **Error! Bookmark not defined..**

After reboot install the Music Player Daemon (mpd) and its client (mpc) along with the Python3, MPD and **python3-rpi.gpio** libraries. Note that you must install **python3-mpd** and not **python.mpd**.

```
$ sudo apt install mpd mpc python3-mpd python3-rpi.gpio
```

Answer yes 'y' when asked to continue.

If the installation says it cannot find the above packages, then re-run the update command to update the library list and retry installing the above packages.

```
$ sudo apt update
```

**IMPORTANT:** Unfortunately, the configuration file (**/etc/mpd.conf**) supplied with MPD version 0.22.6 in **Bookworm** and onwards is corrupt. It is necessary to correct this as shown below.

Find the following entries in **/etc/mpd.conf** somewhere around line 232.

#### Corrupt /etc/mpd.conf file

```
#audio_output {
#    type          "alsa"
#    name          "My ALSA device"
#    device        "hw:0,0"
#    mixer_type    "software"
##    mixer_device  "default"      # optional
##    mixer_control "PCM"         # optional
##    mixer_index   "0"           # optional
#}
```

Change the above entries to the following:

#### Corrected /etc/mpd.conf file

```
audio_output {
    type          "alsa"
    name          "My ALSA Device"
    device        "hw:0,0"
    mixer_type    "software"
#    mixer_device  "default"      # optional
#    mixer_control "PCM"         # optional
#    mixer_index   "0"           # optional
}
```

If installing on **Bookworm Lite** this will take quite a long time as there are a lot of software libraries to be installed.

At this stage there are no playlists configured so the music daemon won't play anything. The play lists are created when the **radiod** Debian Package is installed in the next section.

The version of **MPD** released with **Bookworm** is **0.22.6** which is reasonably up-to-date. You should install MPD version **0.22.6** first. When it is all working if you are having problems playing certain stations, it is well worth installing the latest version of MPD as shown in the section called *Compiling and installing the latest Music Player Daemon* on page 118.

#### **Installing pulseaudio**

The **pulseaudio** package may or may not need to be installed. This has been different between various releases of the operating system. Currently the **pulseaudio** package causes a lot of problems with the radio software, in particular **espeak** and the Alsa sound system. If you do not intend to use espeak and need pulseaudio, for example, for blue-tooth devices or pHat BEAT then install it. The following table shows the options:

**Table 2 PulseAudio installation options**

Type of radio installation	Install pulseaudio
Bob Rathbone   Raspberry PI Internet Radio - Chapter 6 Advanced topics	95

<b>Using espeak</b>	No
<b>Pimoroni Pirate Radio with pHat BEAT</b>	Yes (Installed by phatbeat)
<b>Pimoroni Pirate Audio/mini-speaker</b>	No
<b>Adafruit speaker bonnet</b>	No
<b>LCD display radio</b>	No unless using above DACs
<b>HDMI or touch-screen displays</b>	No unless using above DACs
<b>IQaudIO, HiFiBerry or JustBoom DACs</b>	No
<b>Bluetooth sound devices</b>	Yes

To install **pulseaudio**:

```
$ sudo apt install pulseaudio
```

To remove **pulseaudio**:

```
$ sudo apt remove pulseaudio
```

If installing on Bookworm disable the `pulseaudio-enable-autospawn.service`.

```
$ sudo systemctl stop pulseaudio-enable-autospawn.service
```

At the time of writing there was a fault with the **lxplug-volumepulse** package which stops any of the menus on the Raspberry Pi desktop from being displayed. Remove the **lxplug-volumepulse** package with the following command:

```
$ sudo apt remove lxplug-volumepulse
```

## Installing OLED TFT driver software and libraries

If you are not using any OLED TFT display then skip this section.

**IMPORTANT NOTE:** The following devices and programs use **PIP3** to install the drivers.

- Olimex OLED or SSD1306 OLED TFTs
- Pimoroni Pirate Audio Radio
- SH1106 SPI OLED TFT
- Adafruit 2.5 and 3.5-inch TFT touchscreen
- Monochrome OLEDs which use the LUMA TFT driver
- The weather programs
- The IQaudIO controller

If you attempt to use **PIP3** (The name is a recursive acronym for "Pip Installs Packages") on the **Bookworm OS** it will fail with the following message:

To install Python packages system-wide, try `apt install python3-xyz`, where xyz is the package you are trying to install.

If you wish to install a non-Debian-packaged Python package, create a virtual environment using `python3 -m venv path/to/venv`. Then use `path/to/venv/bin/python` and `path/to/venv/bin/pip`. Make sure you have `python3-full` installed.

For more information visit <http://rptl.io/venv>

To allow **pip3** to be used on **Bookworm OS** carry out the following instruction:

```
$ sudo mv /usr/lib/python3.11/EXTERNALLY-MANAGED  
/usr/lib/python3.11/EXTERNALLY-MANAGED.old
```

Note that the above is all one line. There is no need to carry out the above instruction on system running **Bullseye OS**.

You will now be able to use **pip3** to install the driver software and programs shown in this manual, however you will see the following warning message:

```
WARNING: Running pip as the 'root' user can result in broken permissions and  
conflicting behaviour with the system package manager. It is recommended to  
use a virtual environment instead: https://pip.pypa.io/warnings/venv
```

The above message can be safely ignored for the installation procedures in this manual. However, if installing other author's software with **pip3**, you should seek advice from that author.

If using **Raspbian Lite** also install the **python3-pil** package:

```
$ sudo apt -y install python3-pil
```

### Install display drivers

Note: If you are installing the software for any of the products listed below then first carry out the instructions in the appropriate section. Do this before installing the radio software.

**Table 3 Display driver software installation**

Device	Section/Devices	Page
Pirate Radio	<b>Installing Pimoroni Pirate Radio (pHat BEAT)</b>	<b>34</b>
Pirate Audio	<b>Installing the Pimoroni Pirate Audio</b>	<b>44</b>
SSD1306 0.9-inch OLED	<b>Installing the SSD1306 OLED driver libraries</b>	<b>49</b>
SH1106 SPI (joystick)	<b>Installing the SH1106 SPI OLED driver libraries</b>	<b>53</b>
MHS RPi displays	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
SH1106 1.3-inch OLED	<b>Installing LUMA monochrome OLEDs</b>	<b>59</b>
LUMA OLED devices	SSD1306, SSD1309, SSD1325, SSD1331, SH1106, WS0010	<b>59</b>
Waveshare TFTs	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
Grove LCD RGB	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
Adafruit TFT	<b>Installing the Adafruit 2.5 and 3.5-inch TFT touchscreen</b>	<b>54</b>

PiFace CAD	Error! Reference source not found.	Error! Bookmark not defined.
Waveshare SH1106 SPI	Installing the SH1106 SPI OLED driver libraries	53

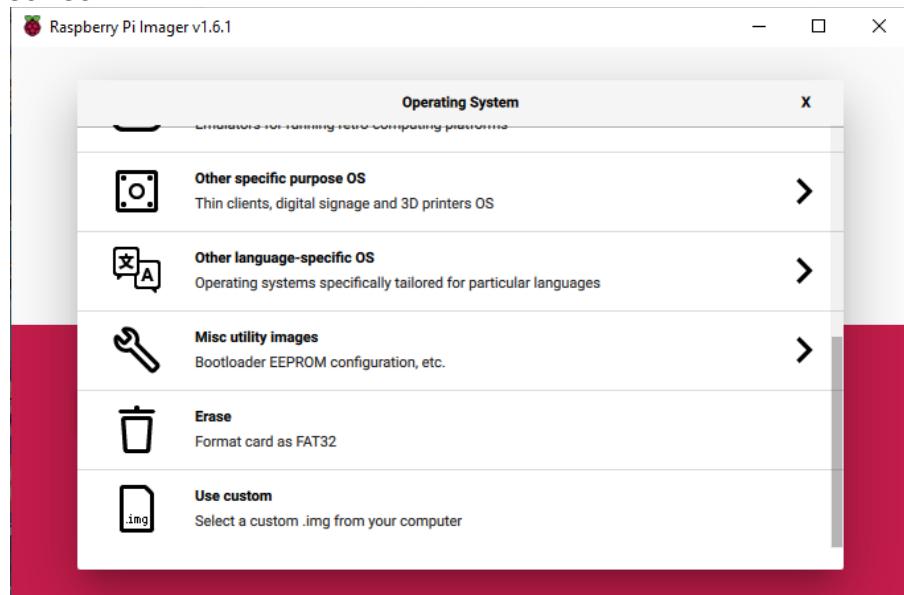
Installing the Radio Daemon on page 26. If not continue below.

The SD card is loaded with some special software to overwrite the EEPROM code to boot from USB. Download the Raspberry Imager software from <https://www.raspberrypi.org/software/>

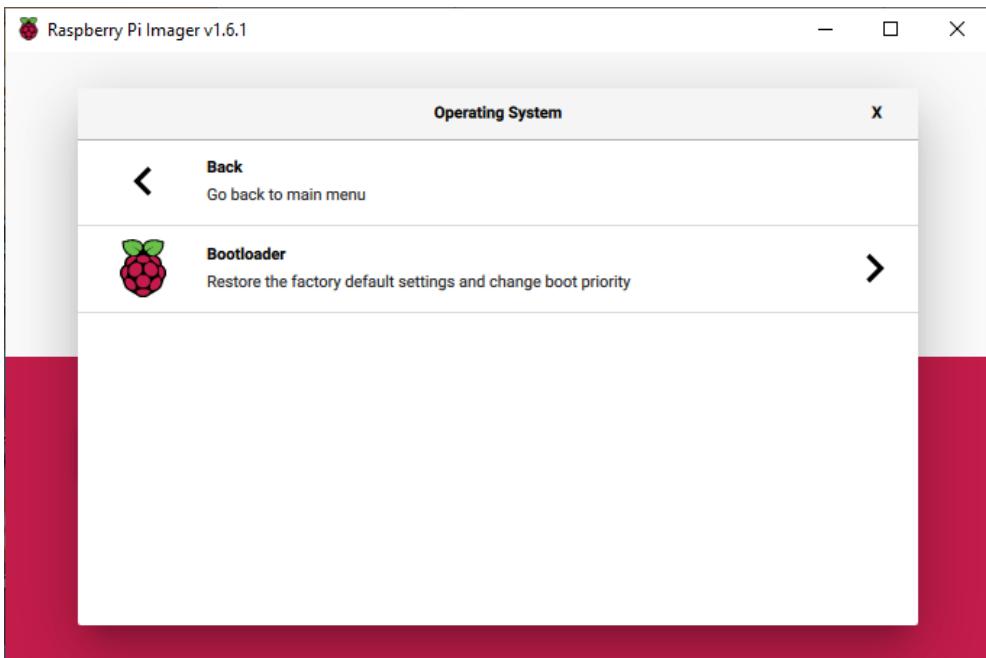
Insert the spare SD card into the PC and start the Raspberry Pi Imager software.



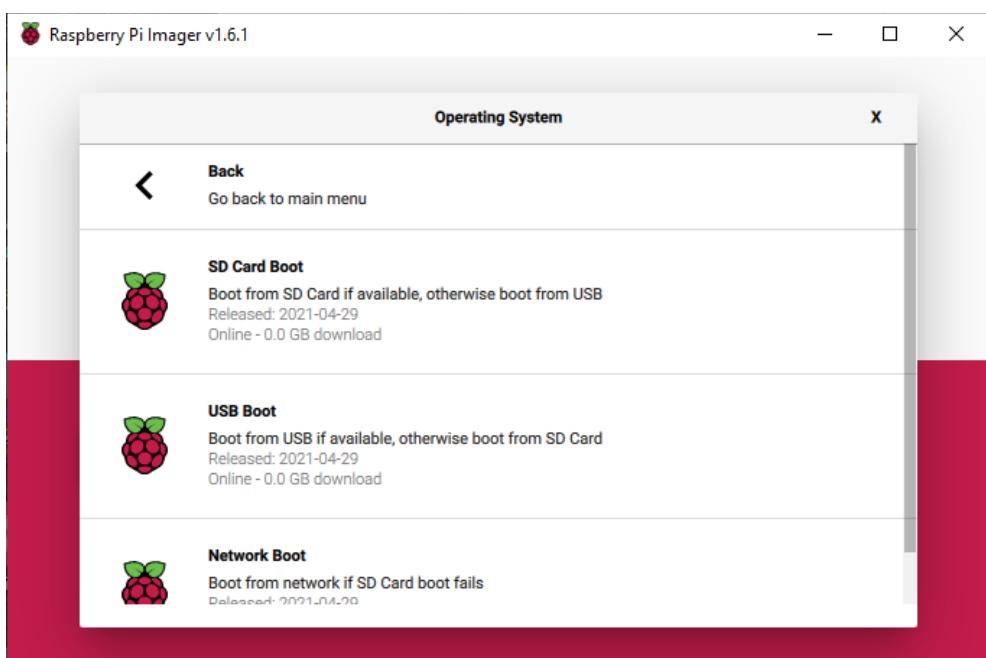
Select "CHOOSE OS"



Use the scroll down and select "Misc Images – Bootloader EEPROM configuration"



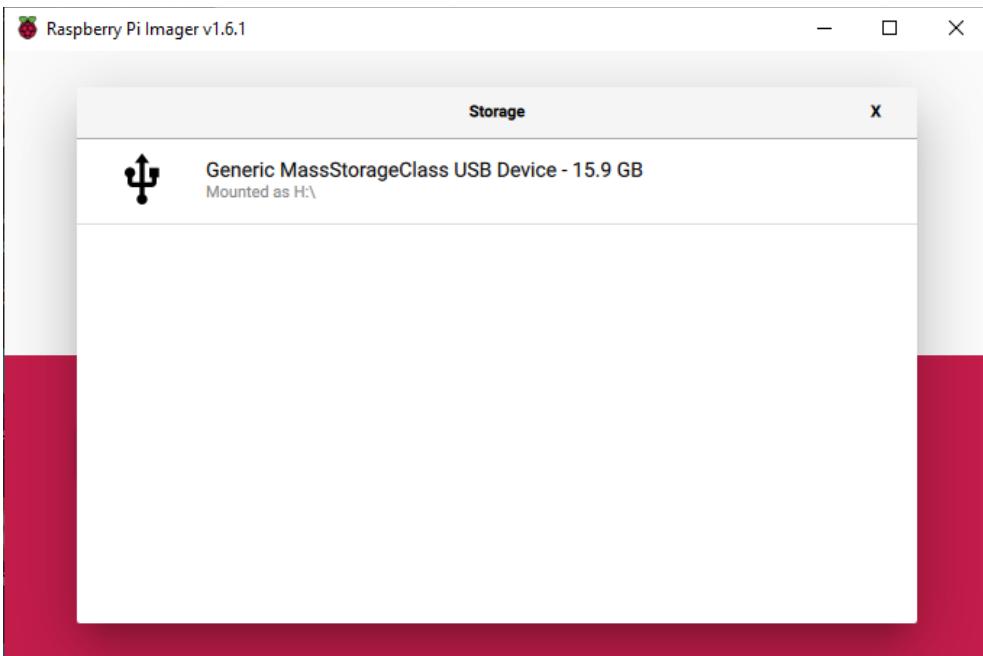
Click the right arrow ">" on "Bootloader" to show the options below.



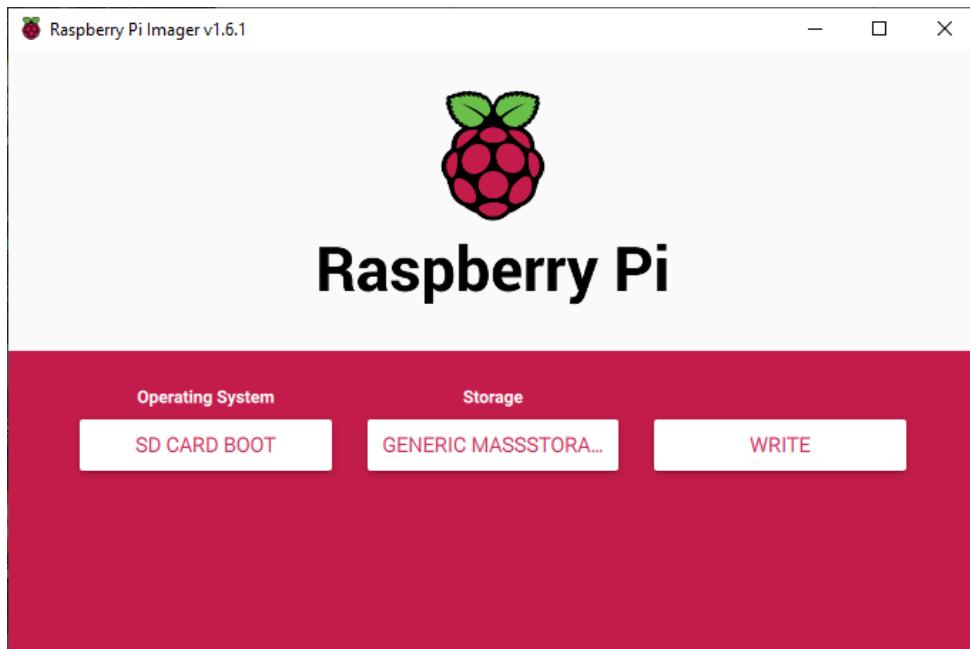
Select the first or second option depending upon your operational requirements. Typically, this will be the second option "Boot from USB if available, otherwise boot from SD card".



Click on “Choose storage”



Select your SD card:



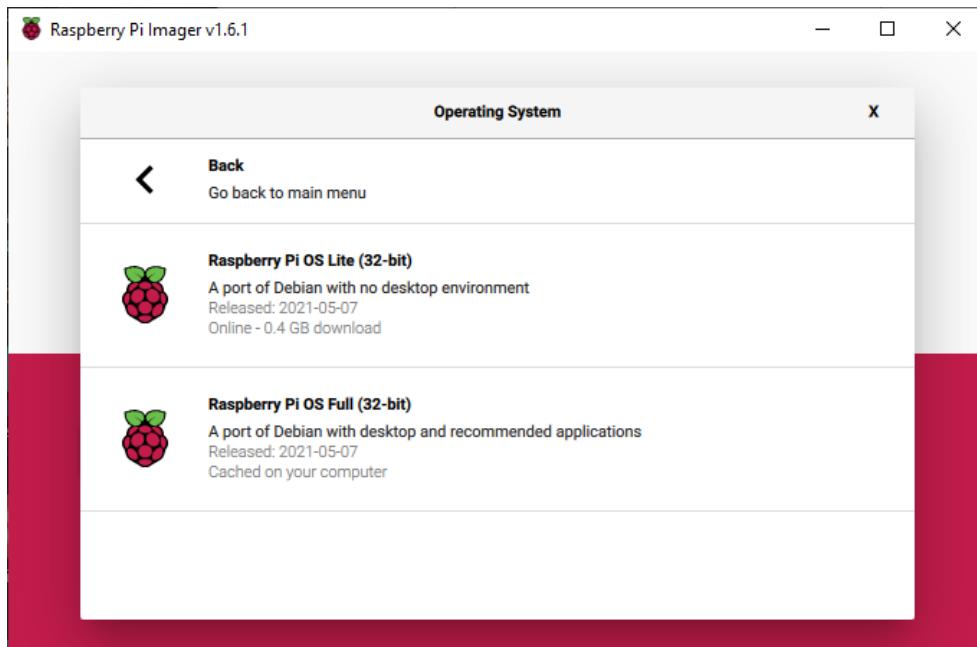
Now press “Write” to create the SD Card image. This will be very quick as there isn’t much to write. Once written, remove the SD card from the PC and label it as USB boot. You will need this later.

### [Creating the bootable USB drive](#)

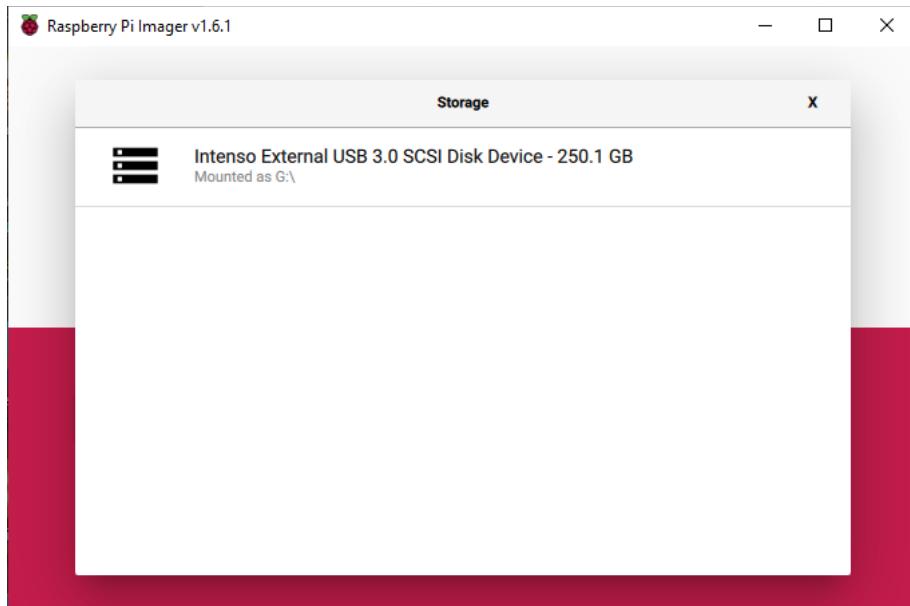
Plug your new USB disk drive into the PC.

Now create the Operating system on the USB disk drive (Not a USB stick). Restart the Raspberry OS imager.

Select the required operating system. Recommended is the “Raspberry Pi OS full (32 bit)”



Now select the USB drive. In the example below this is a USB 3.0 250 Mbyte 2.5-inch disk drive which should be more than enough for most purposes.



Now write to the USB disk as previously shown. Once complete, remove the USB drive from the PC.

#### Modifying the EEPROM boot order

Take the SD card that you created and labelled as “USB boot” and put it into the Raspberry Pi SD card slot. Power on the Raspberry Pi.

After a few seconds you will see the LED on the Raspberry Pi repeatedly flashing. This means that it has finished writing to the RPi EEPROM. Power off the Raspberry Pi and remove the SD card and put it somewhere safe so that you don’t accidentally put it into any other Raspberry Pi.

#### USB 2.x SSD disk drive not booting

Disable the VC4 DRM driver.

Edit the `/boot/cmdline.txt` file using sudo. Disable VC4 driver by commenting it out.

```
#dtoverlay=vc4-kms-v3d
```

Reboot the Raspberry Pi.

#### USB 3.x SSD disk drive slow booting

Skip this section if you are using a normal disk drive with a spindle.



Note: At the time of writing there is a known problem with some USB 3.0 SSD drives which do not support the UAS protocol and cause them to operate very slowly to a point of being unusable. It is necessary to modify the `/boot/cmdline` file to correct this.

For more information on this problem see the following article:

<https://www.raspberrypi.org/forums/viewtopic.php?f=28&t=245931>

### *Find the SDD Vendor and Product ID*

If using a USB SSD disk drive, it is necessary to first identify the Vendor and Product ID of the USB SDD disk drive and its device. Remove any USB storage devices and boot up the Raspberry Pi from an SD card. Once fully booted up, insert the SDD USB drive into the USB 3.0 port.

Log into the Raspberry Pi and then run the dmsg (ring buffer display) program:

```
$ dmesg
```

This should display details of the newly inserted SSD disk. The following example shows the details for an Intenso 250 Gigabyte SDD drive.

Note the Vendor and Product ID. In this example it is **152d** and **0579** respectively. Also note the device name of the first partition (/boot) of the SSD drive. This is **sda1** in this case.

```
[ 235.244594] usb 2-1: new SuperSpeed Gen 1 USB device number 3 using xhci_hcd
[ 235.276190] usb 2-1: New USB device found, idVendor=152d, idProduct=0579,
bcdDevice= 5.06
[ 235.276213] usb 2-1: New USB device strings: Mfr=1, Product=2,
SerialNumber=3
[ 235.276231] usb 2-1: Product: Portable SSD
[ 235.276249] usb 2-1: Manufacturer: Intenso
[ 235.276266] usb 2-1: SerialNumber: 2020090810405
[ 235.315953] scsi host1: uas
[ 235.317600] scsi 1:0:0:0: Direct-Access      Intenso  Portable SSD
0506 PQ: 0 ANSI: 6
[ 235.319836] sd 1:0:0:0: Attached scsi generic sg1 type 0
[ 235.833551] sd 1:0:0:0: [sdb] 500118192 512-byte logical blocks: (256
GB/238 GiB)
[ 235.833561] sd 1:0:0:0: [sdb] 4096-byte physical blocks
[ 235.833760] sd 1:0:0:0: [sdb] Write Protect is off
[ 235.833769] sd 1:0:0:0: [sdb] Mode Sense: 53 00 00 08
[ 235.834181] sd 1:0:0:0: [sdb] Write cache: enabled, read cache: enabled,
doesn't support DPO or FUA
[ 235.834862] sd 1:0:0:0: [sdb] Optimal transfer size 33553920 bytes not a
multiple of physical block size (4096 bytes)
[ 235.876806] sda: sda1 sda2
[ 235.879360] sd 1:0:0:0: [sdb] Attached SCSI disk
```

Now mount **/dev/sda1** on the **/mnt** partition. This will be very slow, about one minute, so be patient:

```
$ sudo mount /dev/sda1 /mnt
```

Using sudo edit the **cmdline.txt** file.

```
$ cd /mnt
$ sudo nano cmdline.txt
```

Add the following line to the front of all of the other options and save the file. Use the Vendor and Device ID previously identified for your device.

```
usb-storage.quirks=152d:0579:u
```

The **cmdline.txt** file should look like the following

```
usb-storage.quirks=152d:0579:u console=serial0,115200 console=tty1  
root=PARTUUID=19de2757-02 rootfstype=ext4 elevator=deadline fsck.repair=yes  
rootwait quiet splash plymouth.ignore-serial-consoles  
systemd.run=/boot/firstrun.sh systemd.run_success_action=reboot  
systemd.unit=kernel-command-line.target
```

Now sync the discs.

```
$ sync; sync
```

Log out and power off and remove the SD card. Leave the drive plugged in and got to the next section.

### Boot up the Raspberry Pi from the USB disk drive

Now plug the USB disk drive into one of the USB 3.0 ports and power on. If everything has been correctly done, the Raspberry Pi will boot up from the USB disk drive. If not, check that this procedure has been correctly carried out. This can also be confirmed with the **dmesg** command.

```
$ dmesg | grep -i uas  
[    1.515463] usbcore: registered new interface driver uas  
[    2.161159] usb 2-1: UAS is ignored for this device, using usb-storage instead  
[    2.161289] usb 2-1: UAS is ignored for this device, using usb-storage instead
```

If all is well then continue with installation of the rest of the Operating System as shown **Error! Reference source not found.** on page **Error! Bookmark not defined..**

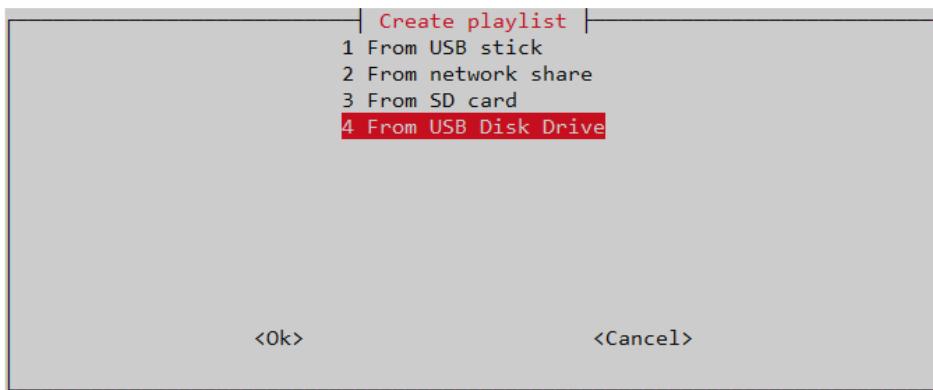
If you wish to boot from an SD card instead of the USB disk drive then remove the USB drive from the USB 3.0 port and insert an SD card configured with the Raspberry Pi OS into the SD card slot and power on. The Raspberry Pi will detect that it does not have a USB drive anymore and will boot from the SD card.

### Playing music from the USB disk drive

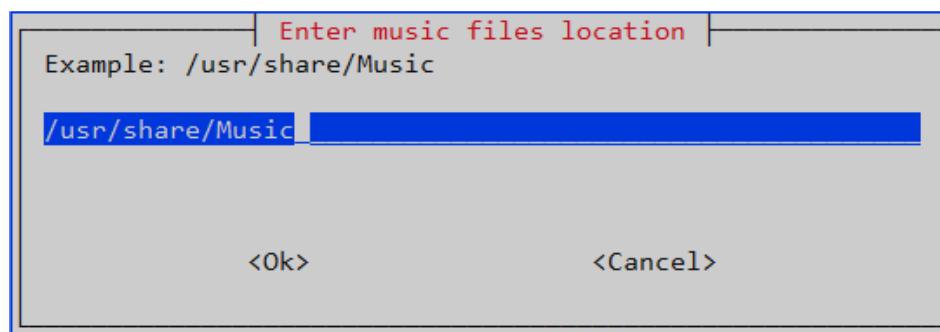
If using a USB disk drive, it makes more sense to play media files from the USB disk drive rather than a separate USB stick.

Using SFTP, copy the music from a PC to the **/usr/share/Music** directory which should already exist and reload the library via the options menu.

Now run the **create\_playlist.sh** program. Select option 3 (SD card).

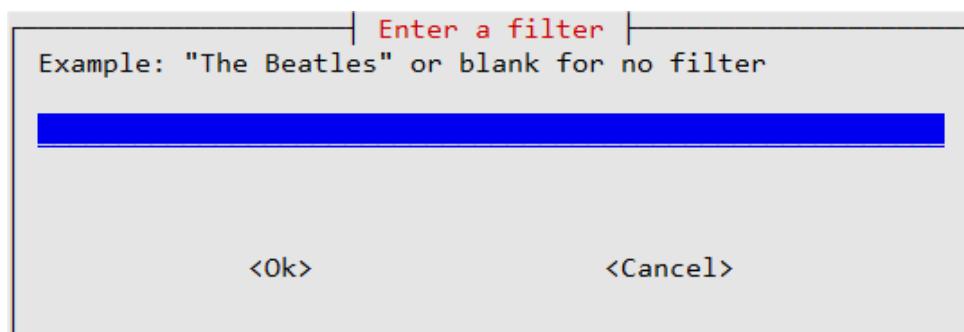


Choose option 4 “From USB Disk Drive”.

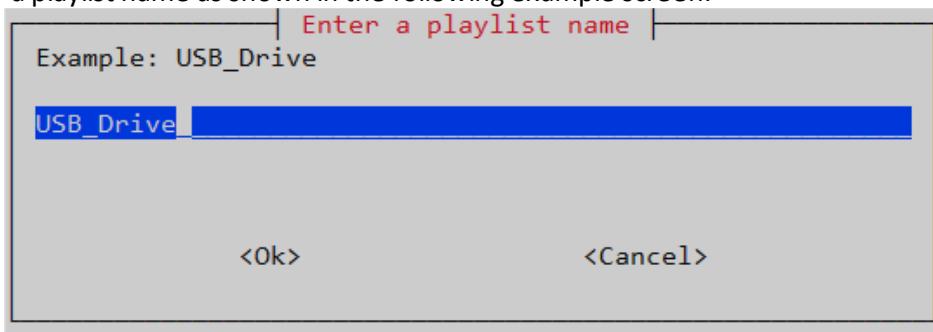


If you already are using Change the default location to **/usr/share/Music2** or any other location preferably in the /usr/share directory. Whatever directory you specify must pre-exist and contain music files.

Confirm your choice. On the next screen press enter for no filter on the media track names.



Now enter a playlist name as shown in the following example screen.



Press OK. The program will now create the Music playlist. Restart the radio and select the source men to select the new Music playlist (USB\_Drive).

## Repairing a non-bootable SD card

One of the most frustrating things is when your radio which has been working fine and suddenly stops booting up. This is extremely frustrating but all may not be lost. It may be possible to repair an SD Card which has become non-bootable. To do this you will need the following:



- A USB SD card reader
- A spare SD card with Raspberry Pi OS

USB SD card readers are very cheap. You should buy one before you need it in an emergency. Also prepare a spare SD card with the Raspberry Pi OS on it.

It is best to do this before you need it in an emergency. See **Error! Reference source not found.** on page **Error! Bookmark not defined.**

Figure 14 A USB SD card reader

1. Boot up the Raspberry Pi with the good SD card previously created.
2. Log into the Raspberry Pi as user **pi**.
3. Plug the faulty SD card into the USB SD card reader and plug the card reader into the Raspberry Pi.
4. Run the **dmesg** command. This should show the device number of the SD card at the end, similar to that shown below.

```
$ dmesg
:
[ 700.247477] sd 0:0:0:1: [sdb] Attached SCSI removable disk
[ 700.252572] sda: sda1 sda2
[ 700.260025] sd 0:0:0:0: [sda] Attached SCSI removable disk
```

Now run the following **fsck** commands to attempt to repair the file systems on **sda1** and **sda2**.

```
$ sudo fsck -y /dev/sda1
fsck from util-linux 2.33.1
fsck.fat 4.1 (2017-01-24)
0x41: Dirty bit is set. Fs was not properly unmounted and some data may be
corrupt.
  Automatically removing dirty bit.
Performing changes.
/dev/sdal: 285 files, 98417/516190 clusters
```

### /dev/sda2

```
$ sudo fsck -y /dev/sda2
fsck from util-linux 2.33.1
e2fsck 1.44.5 (15-Dec-2018)
rootfs: recovering journal
Clearing orphaned inode 914 (uid=111, gid=119, mode=0100600, size=0)
```

```
Setting free inodes count to 765822 (was 765872)
Setting free blocks count to 2122943 (was 2126334)
rootfs: clean, 149826/915648 files, 1607489/3730432 blocks
```

Run the following to synchronise the disk buffers with the disk.

```
$ sync; sync
```

Now power off the Raspberry Pi and plug the repaired SD card into the Raspberry Pi and power on. The above actions should normally be enough to get the RPi to boot. If not carry out the following:

First repeat the above procedure to make sure that the file system on the SD card is OK.

Now mount **/dev/sda1** on the **/mnt** directory. Then change to the **/mnt** directory.

```
$ sudo mount /dev/sda1 /mnt
$ cd /mnt
```

Check the **cmdline.txt** and **config.txt** file sizes look reasonable.

```
$ ls -la cmdline.txt config.txt
-rwxr-xr-x 1 root root 144 Sep 27 11:45 cmdline.txt
-rwxr-xr-x 1 root root 1838 Aug 28 14:00 config.txt
```

Check the **cmdline.txt** file looks something like that shown below.

```
$ cat cmdline.txt
console=tty1 root=PARTUUID=0b18c756-02 rootfstype=ext4 elevator=deadline
fsck.repair=yes rootwait quiet splash plymouth.ignore-serial-consoles
```

Check that the **config.txt** file looks normal. This will vary from that shown below.

```
$ cat config.txt
:
# Enable audio (loads snd_bcm2835)
dtoparam=audio=on

[pi4]
# Enable DRM VC4 V3D driver on top of the dispmanx display stack
dtoverlay=vc4-fkms-v3d
max_framebuffers=2

[all]
dtoverlay=hifiberry-dac
dtoverlay=i2s-mmap
```

If the **config.txt** file looks damaged or is missing, copy it from the good **SD** card. It may be necessary to re-configure the audio system (Run **configure\_audio.sh** once rebooted).

Faulty **cmdline.txt** example. The file size in this example is zero bytes size.

```
$ ls -la cmdline.txt config.txt
```

```
-rwxr-xr-x 1 root root 0 Sep 27 11:45 cmdline.txt
```

First copy a good **cmdline.txt** file from the good SD card to the faulty one.

```
$ sudo cp /boot/cmdline.txt /mnt
```

However, this one will still not work as the root file system UUID is incorrect as it is for the good SD card and not the root UUID of faulty one.

```
$ cat /boot/cmdline.txt
console=tty1 root=PARTUUID=dfde9f91-02 rootfstype=ext4 elevator=deadline
fsck.repair=yes rootwait quiet splash plymouth.ignore-serial-consoles
```

Find out the PARTUUID of **/dev/sda2** with the **blkid** utility.

```
$ sudo blkid /dev/sda2
/dev/sda2: LABEL="rootfs" UUID="3a324232-335f-4617-84c3-d4889840dc93"
TYPE="ext4" PARTUUID="0b18c756-02"
```

Edit the **cmdline.txt** file and insert the correct PARTUUID

```
$ cat /boot/cmdline.txt
console=tty1 root=PARTUUID=0b18c756-02 rootfstype=ext4 elevator=deadline
fsck.repair=yes rootwait quiet splash plymouth.ignore-serial-consoles
```

Run the following to synchronise the disk buffers with the disk.

```
$ sync; sync
```

Try booting using the repaired SD card. If there are still problems, check other files in the boot partition (**/dev/sda1** mounted on **/mnt**) of the faulty SD card. For example, the kernel files:

```
$ ls -la /mnt/kernel*
-rwxr-xr-x 1 root root 6353376 Aug 15 13:46 /mnt/kernel7.img
-rwxr-xr-x 1 root root 6777440 Aug 15 13:46 /mnt/kernel71.img
-rwxr-xr-x 1 root root 7904011 Aug 15 13:47 /mnt/kernel8.img
-rwxr-xr-x 1 root root 6006712 Aug 15 13:47 /mnt/kernel.img
```

On the good SD card. The dates may be different but they otherwise look OK

```
$ ls -la /boot/kernel*
-rwxr-xr-x 1 root root 6353624 Sep 21 14:16 /boot/kernel7.img
-rwxr-xr-x 1 root root 6779080 Sep 21 14:16 /boot/kernel71.img
-rwxr-xr-x 1 root root 7904106 Sep 21 14:16 /boot/kernel8.img
-rwxr-xr-x 1 root root 6006608 Sep 21 14:16 /boot/kernel.img
```

If all of these actions fail to restore the SD card then it is very possible that the SD card is physically damaged, in which case it will be necessary to replace it and install a new OS and Radio software.

## Building your own package

If you do modify the code, it may well be that you wish to create your own Raspbian package or create the **radiod** package directly from GitHub. There are several files and scripts required to build the **radiod** package. You need to first download the following files from GitHub.

See <https://github.com/bobrathbone/piradio6>

- **setup.sh** – From 7.2 onwards, script to set-up the build environment and execute **build.sh**
- **build.sh** – Run this to build a 32-bit package (called from **setup.sh** if a 32-bit system).
- **build64.sh** – Run this to build 64-bit package (called from **setup.sh** if a 64-bit system).
- **piradio** – The 32-bit package definition file which define the package contents.
- **piradio64** – As above but for a 64-bit package
- **piradio.preinst** – This is the script that runs before the package files are installed.
- **piradio.postinst** – This is the script that runs after the package files are installed.
- **piradio.postrm** – This is the script that runs if the package is removed to run clean-up tasks.

```
$ cd  
$ git clone https://github.com/bobrathbone/piradio6
```

From version 7.2 onwards run **setup.sh**. Do not use **sudo**. It only needs to be run once, thereafter it is only necessary to run **build.sh**.

```
$ cd piradio6  
$ ./setup.sh
```

The Web interface also has its own build script namely **buildweb.sh** which is a much simpler example.

Study the **piradio** file in particular to glean how to build your own package. Make copies of the package build files and then modify these with your changes. Do not use the original files used to build the **radiod** package as these will be overwritten if the **radiod** package is updated.

To build the package (Example myradio):

```
$ cp -p build.sh mybuild.sh  
$ cp -p piradio myradio
```

Modify the mybuild.sh file to use your package files

```
PKGDEF=myradio  
PKG=myradiod
```

Modify the package name to match the PKGDEF definition and set the initial version in the **myradio** package definition file.

```
Package : myradiod  
Version : 1.0
```

Run the new build script as user **pi**. Do not use **sudo**.

```
$ ./mybuild.sh
```



Note: Most build warnings can be ignored but you should check if these can be easily corrected. Any errors should be corrected.

Start modifying the code with your changes regularly checking the build still runs OK. Good luck with your build.

### Using PWM GPIOs for sound output on a Pi Zero

Another method of creating audio on a Pi Zero or Pi Zero W, is to use a GPIOs 13 and 18, PWM-0 (Left) and PWM-1 (Right) as shown in the following diagram. Note you will still need an amplifier for speakers.

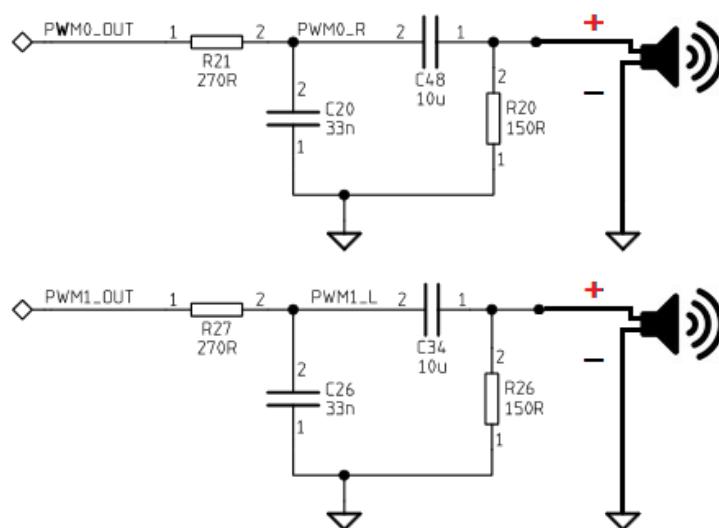


Figure 15 PWM Low band pass filters

This also requires loading the pwm-2chan Device Tree Overlay in **/boot/firmware/config.txt**.

```
dtoverlay=pwm-2chan,pin=18,func=2,pin2=13,func2=4
```

See <https://othermod.com/raspberry-pi-zero-audio-circuit/> for the full article how to do this.

Select the internal audio jack to enable audio output.

### Fitting a Pimoroni On-Off Shim power switch

Unfortunately, the Raspberry Pi has a tendency to corrupt the SD card if the power is suddenly removed. This risk can be lessened by booting from a USB drive or stick. See *Booting from a USB drive or stick* on page 81. Alternatively, you can use one of the several solutions available on-line such as the Pimoroni power on-off shim.



Figure 16 Pimoroni On-Off shim power switch

The Pimoroni On-Off SHIM plugs into the first twelve GPIO pins of the Raspberry Pi GPIO header. There is a small button in the corner of the board next to USB power connector. This button is the on-off button and connects to GPIO17 (Physical pin 11). This button is also connected to the pins marked **BTN** to allow an external button to be fitted as shown in Figure 17 below.

To operate press the button once to switch on power and boot up the Raspberry Pi. When finished with the RPi simply press and hold the button for one second to initiate a clean shutdown and completely cut the power to the Raspberry Pi.

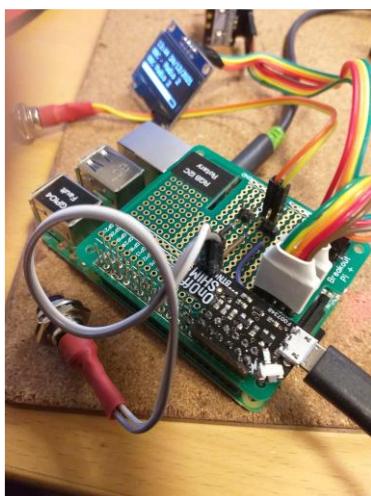


Figure 17 Raspberry Pi Radio with Pimoroni on-off shim

GPIO pin 17 when held low by the button initiates the standard shutdown command. Once this is completed GPIO 4 (Physical pin 7) is lowered by the software to force to completely cut power to the Raspberry Pi.

For more information on the Pimoroni On-Off shim see:  
<https://shop.pimoroni.com/products/onoff-shim>

However, GPIO4 and 17 are used by the radio to connect the mute and menu switch respectively. This means that the conflicting LCD GPIOs need to be reassigned to different GPIOs, for example, GPIO26 and 27.

First edit **/etc/radiod.conf** and change the following definitions from:

```
menu_switch=17  
mute_switch=4
```

to

```
menu_switch=26  
mute_switch=27
```

Physically wire the menu switch to physical pin 37 and the mute switch to physical pin 13.

Pimoroni provide special software on their Web site to support the on-off shim. To install the Pimoroni software run:

```
$ curl https://get.pimoroni.com/onoffshim > onoffshim.sh  
$ sudo ./onoffshim.sh
```

This software must be installed for the Pimoroni on-off shim to work correctly.



Note that the Pimoroni on-off shim cannot be used with the Allo Piano 2.1 DAC with woofer (See **Error! Reference source not found.** on page **Error! Bookmark not defined.**) as the Woofer channel also makes use of GPIO 4 and 17. There is no way around this. Fortunately, most DACs only use a single channel and do not use GPIO 4 and 17.

## Adding a Real Time Clock (RTC)

When the Raspberry Pi is switched off for a while and switched on again it will initially have the system time when the RPi was switched off until the time synchronises with one of the Internet's time servers using the Network Time Protocol (NTP). This is because the RPi does not come with a Real Time Clock (RTC) as standard.



A Real Time Clock (RTC) can be added to a Raspberry Pi. There are several solutions available on-line. The illustration on the left shows the RasClock module available from PiHut and which connects to the RPi GPIO Header. The pins used are:

1. 3.3 Volts +
2. Not used (+5V)
3. SDA GPIO 2
4. Not Used (5V+)
5. SCL GPIO 3
6. GND 0V

Figure 18 RasClock RTC V4.2 module

SDA and SCL are the I2C Data and Clock respectively. The RasClock module needs to be fitted with a CR1220 36mA 3.0 Volt button battery (Bought separately). As the module uses the I2C interface it can be also connected to the above pins using jumper wires. Only pins 1,3,5 and 6 are required.

### Installation

Plug the module into the first six pins of the Raspberry Pi. Power on the Raspberry Pi and add the following line to **/boot/firmware/config.txt** file after the [all] section.

```
dtoverlay=i2c-rtc,pcf2127
```

Reboot the Raspberry Pi. After reboot run the following command:

```
$ sudo hwclock -r  
2022-12-10 12:00:05.664465+00:00
```

This will display the current UTC time. To display local time run **sudo hwclock -l**

## Accessing the Raspberry Pi from a PC or MAC using VNC

If running the graphics versions of the radio (**gradio.py** or **vgradio.py**) or any other graphics programs, it is possible to access the Raspberry Pi desktop using VNC (Virtual Network Computing) using either a PC or Apple Mac computer.

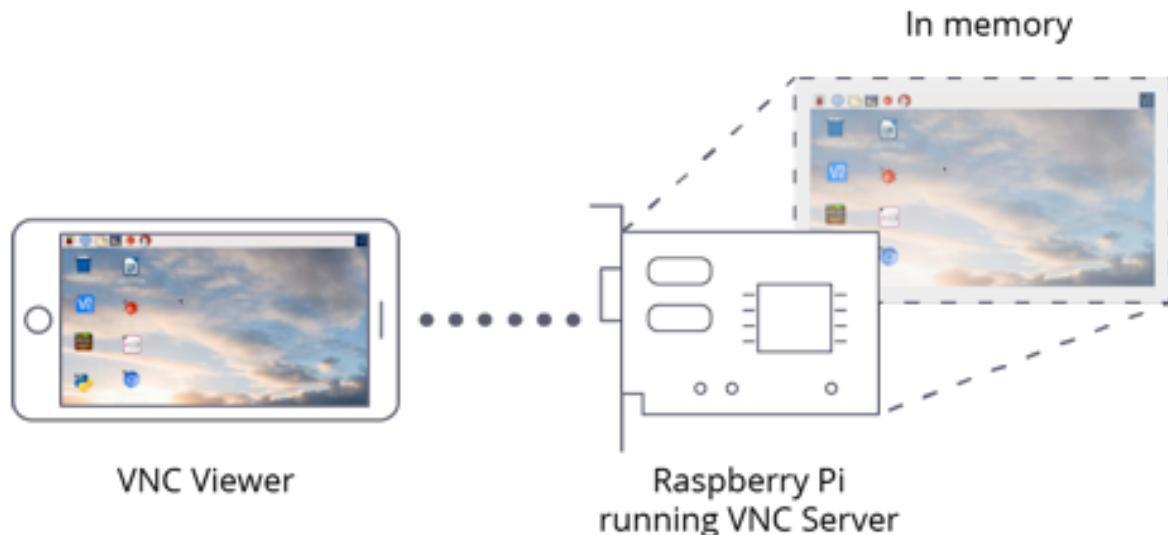


Figure 19 VNC client server architecture

### Enable the VNC server on the Raspberry Pi

The VNC server is normally already installed on the Raspberry Pi. If not skip to the next session. Run **sudo raspi-config** and select **3 Interface Options**. The following screen will be displayed:

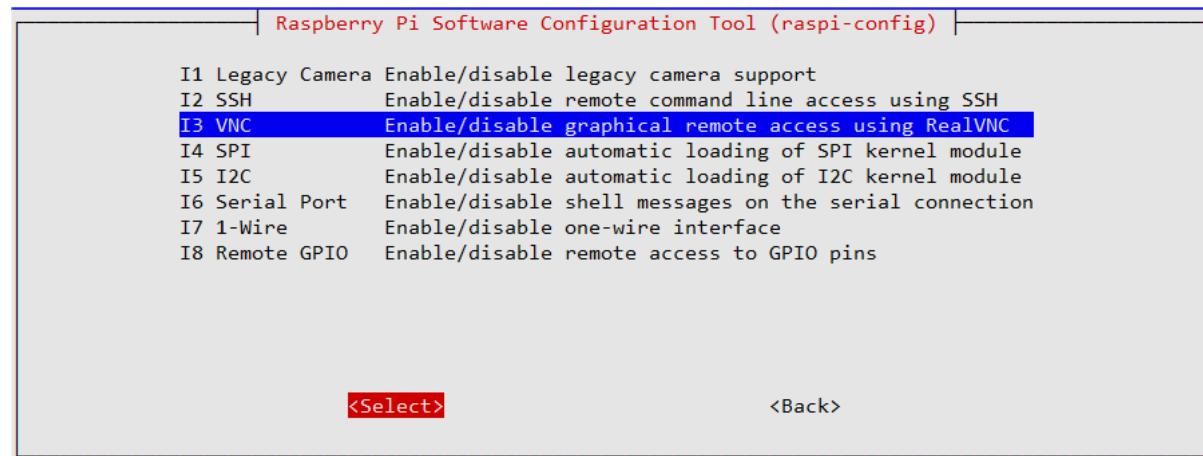
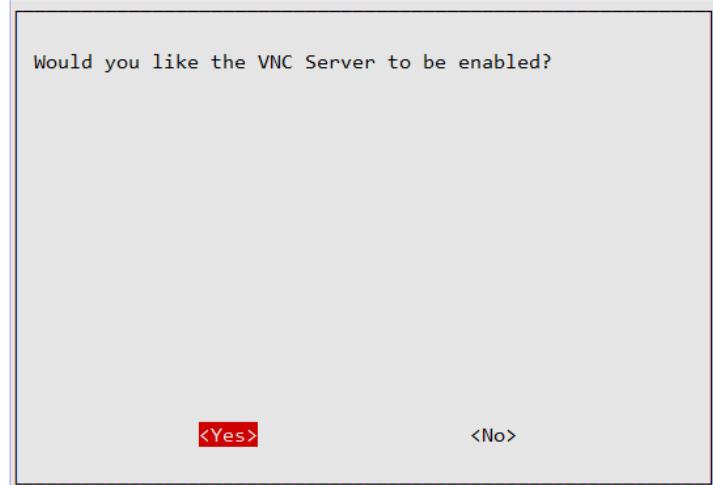


Figure 20 Enabling the VNC server on the Raspberry Pi

Select option **I3 VNC**. The program asks if you wish to enable the VNC server. Answer Yes as shown in the next screen.



There is no need to reboot as raspi-config starts the vnc-server.

### Install Real VNC server on the Raspberry Pi

Skip this section if the above procedure was successful. First make sure that your Raspberry Pi OS is up to date with:

```
$ sudo apt update  
$ sudo apt update -y
```

Next install the VNC server

```
$ sudo apt-get install realvnc-vnc-server
```

Check that it is running

```
$ sudo systemctl status vncserver-x11-serviced.service
```

This should display something similar to the following:

```
● vncserver-x11-serviced.service - VNC Server in Service Mode daemon  
   Loaded: loaded (/lib/systemd/system/vncserver-x11-serviced.service;  
           enabled; vendor preset: enabled)  
   Active: active (running) since Thu 2023-08-31 08:46:49 BST; 1h 19min  
         ago  
     Main PID: 545 (vncserver-x11-s)  
        Tasks: 5 (limit: 1598)  
       CPU: 17.982s  
      CGroup: /system.slice/vncserver-x11-serviced.service  
              └─ 545 /usr/bin/vncserver-x11-serviced -fg  
                  ├ 569 /usr/bin/vncserver-x11-core -service  
                  ├ 1078 /usr/bin/vncagent service 0  
                  ├ 1127 /usr/bin/vncserverui service 0  
                  └─ 1136 /usr/bin/vncserverui -statusicon 0  
  
:
```

## Install the RealVNC viewer on the PC or MAC

There are a number of VNC clients available, both paid for and free. One such client is **RealVNC** and it is free although you will need to create an account using your email address and a password.

The RealVNC viewer is supplied by <https://www.realvnc.com/en/> You can download the viewer from <https://www.realvnc.com/en/connect/download/viewer/>. Do not download **VNC connect** as this is the server version. You need the VNC client viewer. Download the viewer and click on the newly downloaded file and follow the instructions to install it. Once installed open the application.

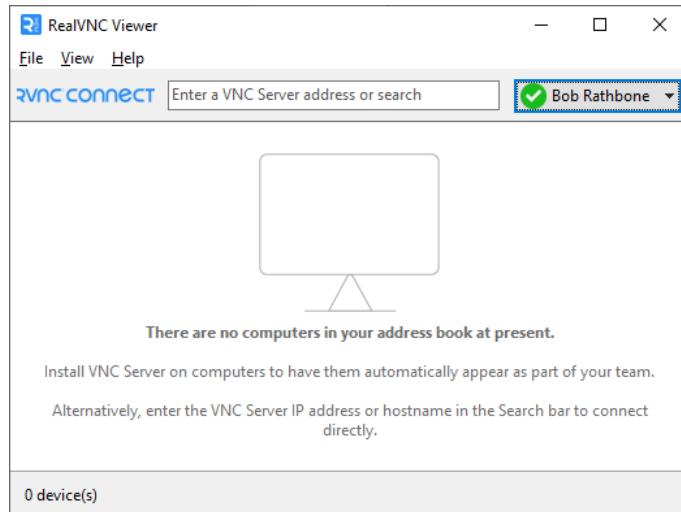


Figure 21 Connecting to the VNC server on the RPi

If you have already created an account this will be displayed (In this case Bob Rathbone) otherwise you will be prompted to create an account when you try to make a connection. First time in you will be prompted for the name of the raspberry pi, thereafter a new connection can be made using the file menu (select new connection).

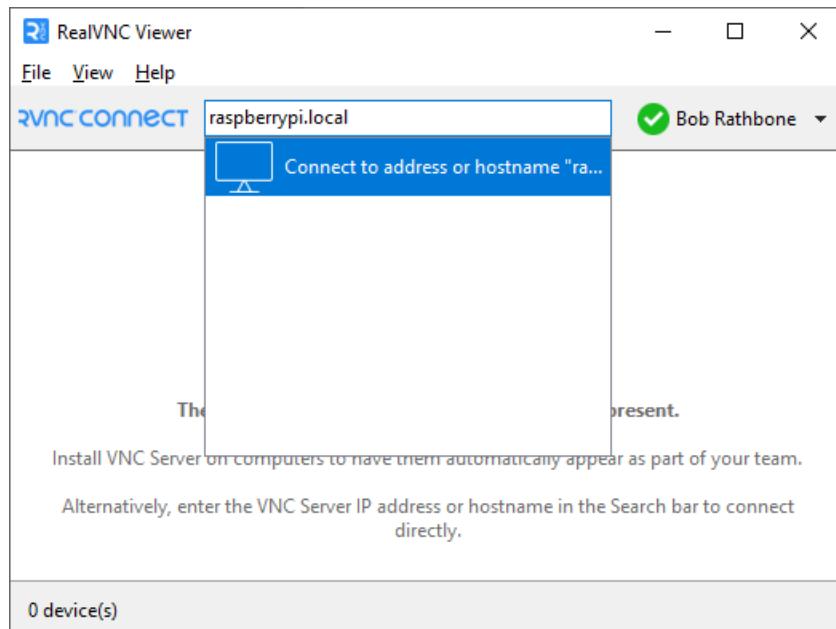
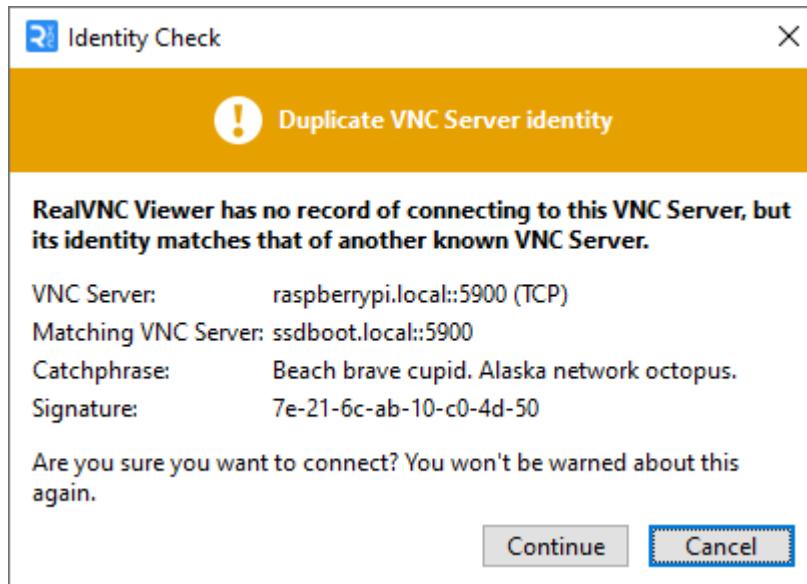


Figure 22 VNC viewer creating a connection

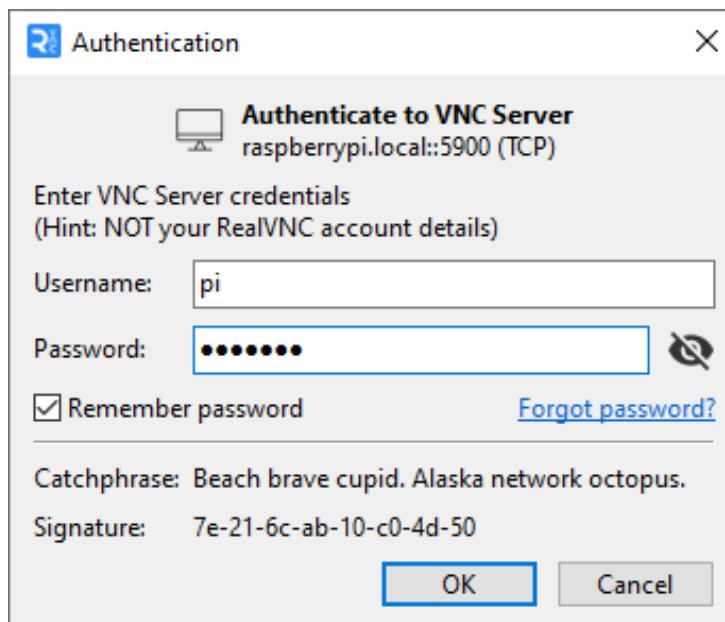
If using the network name of the Raspberry Pi doesn't work then try its IP address (See note on name later in this section).

The following will be displayed. Click on the Continue button:



**Note:** You can also use the IP address of the Raspberry Pi instead of its host name. If you set the SD card up according to the instructions shown earlier. If using the hostname, the hostname was “raspberrypi” then try the name **raspberrypi** first. If that doesn’t work then try **raspberrypi.lan**, **raspberrypi.local** or **raspberrypi.home** in the host field. If the hostname was “piradio” then try the names **piradio**, **piradio.lan**, **piradio.local** or **piradio.home**. If none of these work just use the IP address of the Raspberry Pi.

Now enter the user name (usually pi) and password you are using on the Raspberry Pi.



All being well the Raspberry Pi desktop should be displayed.

The illustration below shows a VNC client running on a PC displaying the graphic version of the radio.

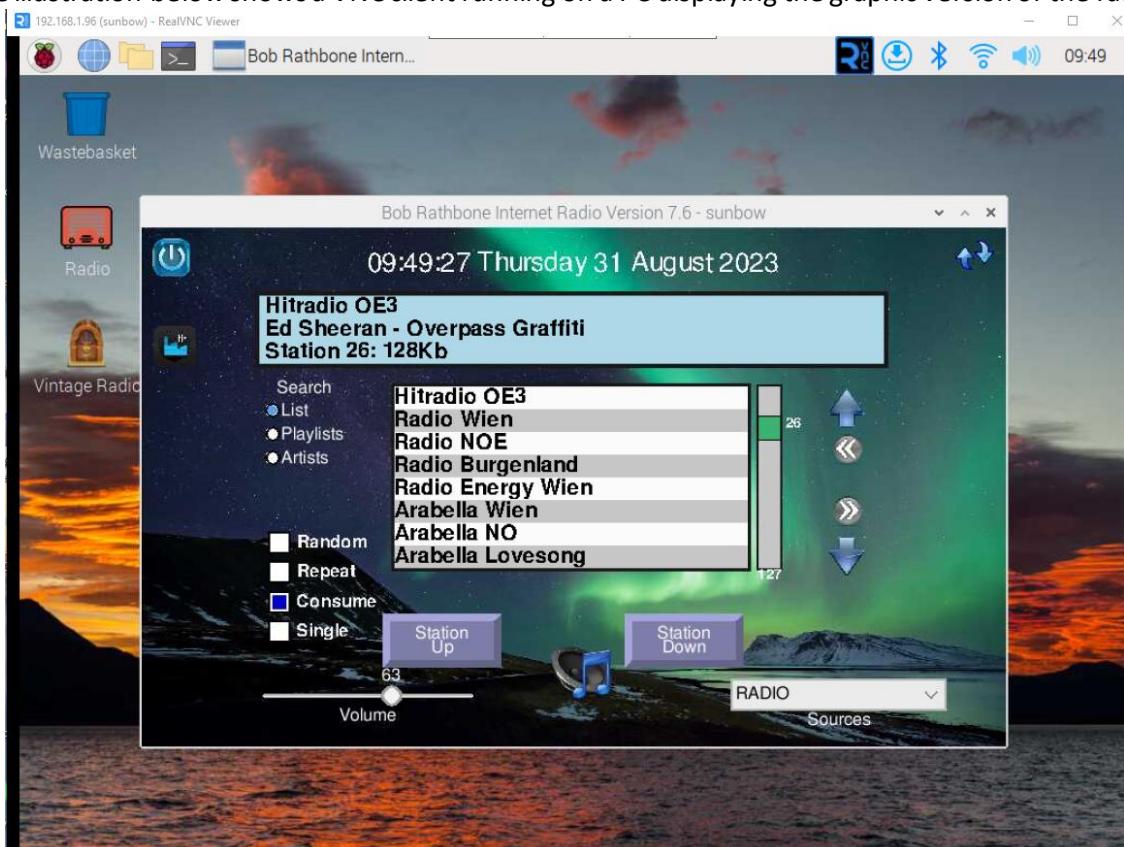


Figure 23 VNC displaying RPi Desktop on a PC



**Note:** *Bob Rathbone Computer Consultancy* does not directly support VNC components. Help for **Real VNC** can be found at <https://help.realvnc.com/hc/en-us>. They also support **realvnc-vnc-server** on the Raspberry Pi.

You may see the following screen if no HDMI screen is connected to the HDMI ports.

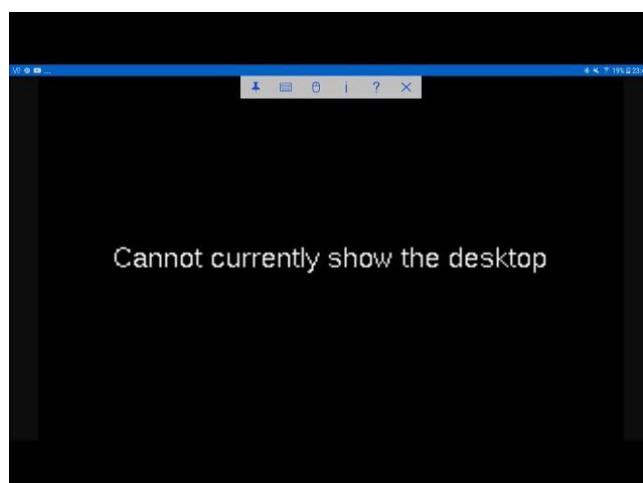


Figure 24 RealVNC display error



Edit the **/boot/firmware/config.txt** file and uncomment the following line.

```
# uncomment if hdmi display is not detected and composite is being output  
hdmi_force_hotplug=1
```

The next time you use VNC viewer you will see that the Raspberry Pi has been added to the connection dialogue. Just click on it to connect.

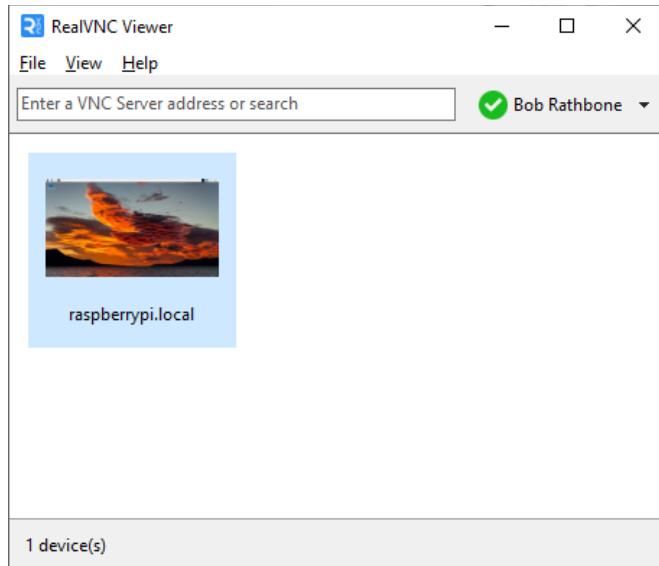
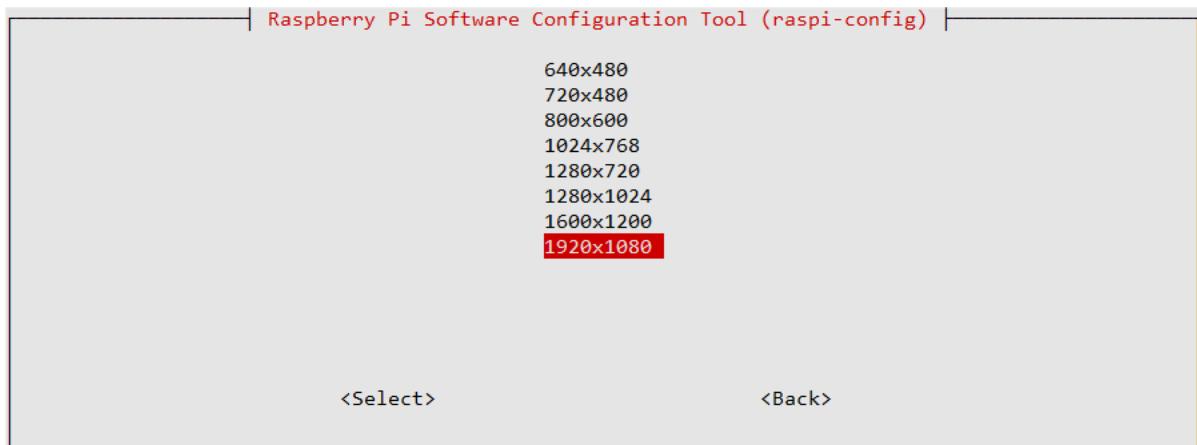


Figure 25 VNC connection dialogue box

It is also possible to alter the VNC screen resolution on the Raspberry Pi.

Run **sudo raspi-config** and select:

**2 Display Options → D5 VNC Resolution**



Select the required resolution (typically the highest 1920 x 1080)

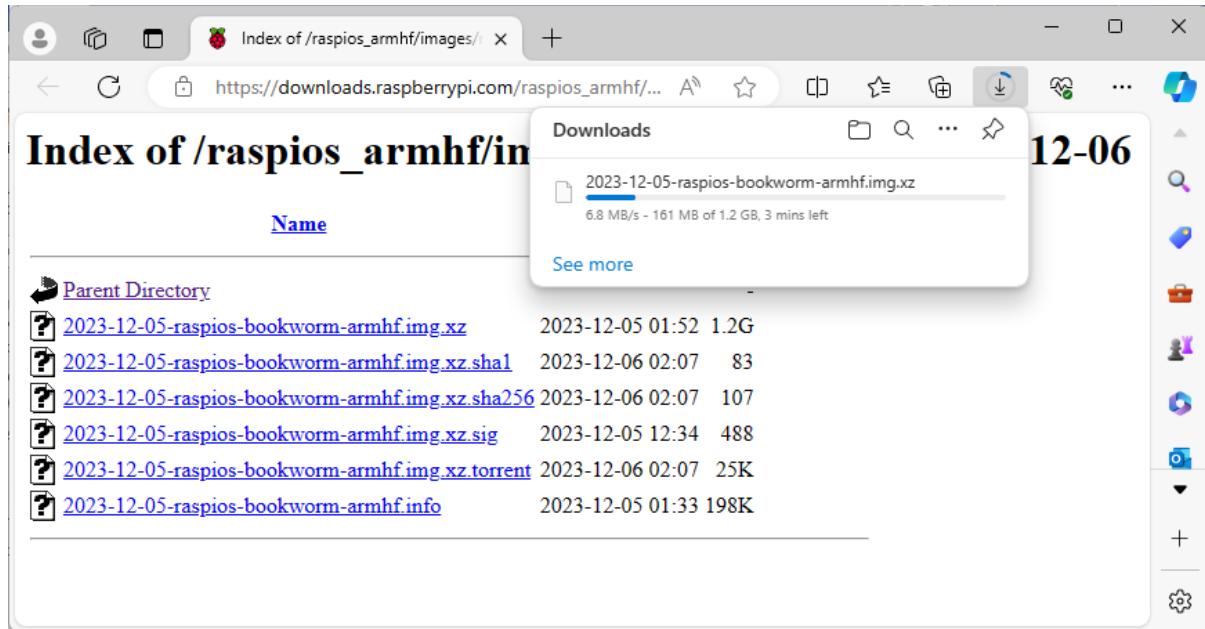
## Installing an earlier release of Bookworm OS

If installing **Bullseye** skip this section as it isn't relevant. If installing **Bookworm** first download the Bookworm OS dated 6<sup>th</sup> of December 2024. Using a Web browser open the following link:

[https://downloads.raspberrypi.com/raspios\\_armhf/images/raspios\\_armhf-2023-12-06/](https://downloads.raspberrypi.com/raspios_armhf/images/raspios_armhf-2023-12-06/)

The following screen will be displayed:

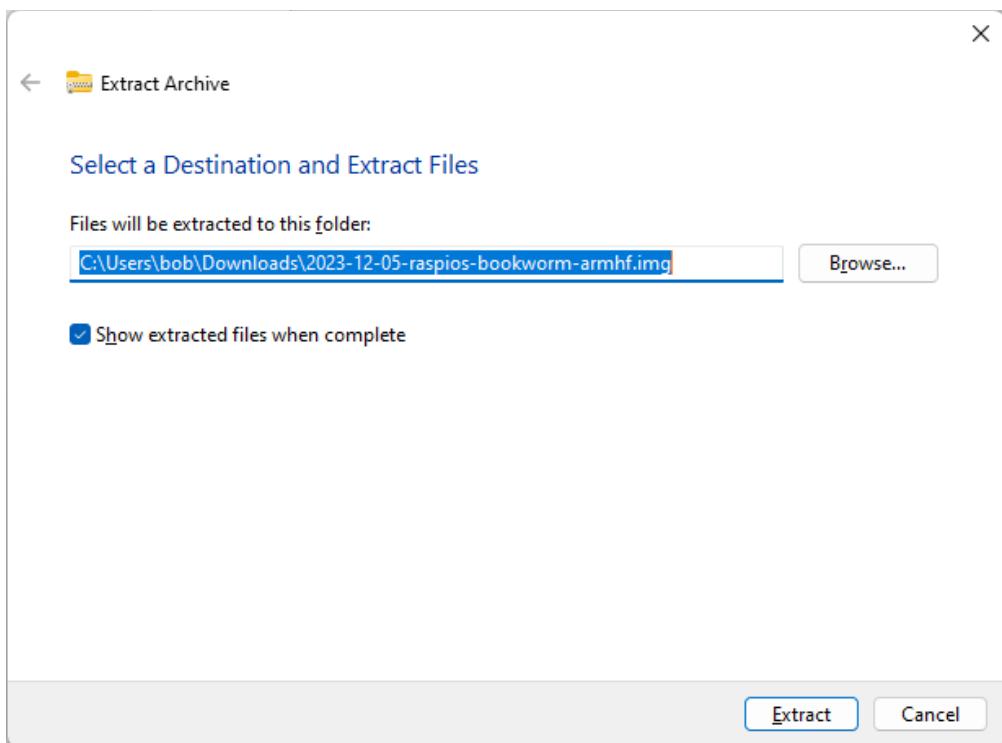
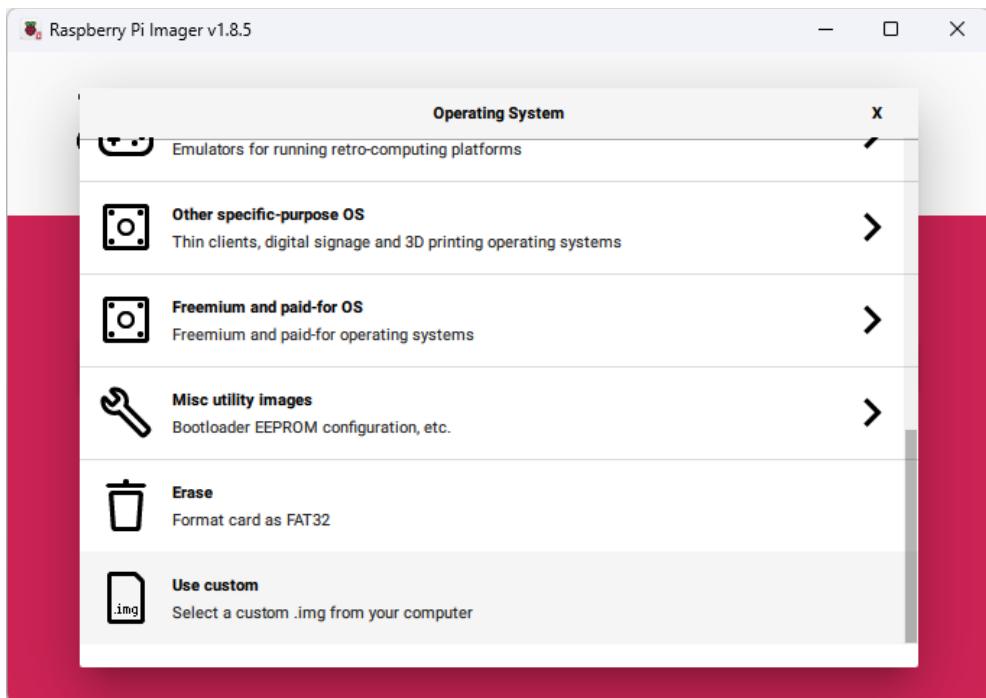
Click on the first file in the list namely **2023-12-05-raspios-bookworm-armhf.img.xz**.



Now open **Windows Explorer** and open the **Downloads** directory. Right click on the **2023-12-05-raspios-bookworm-armhf.img.xz** file and select “Extract all”. This will extract the file **2023-12-05-raspios-bookworm-armhf.img** to a directory of the same name. This is the Bookworm image that will be used later on.

For **Bookworm** scroll down to **Use custom** and then select the image file **2023-12-05-raspios-bookworm-armhf.img** from the Downloads directory.

Now carry out the SD card creation instructions shown **Error! Reference source not found.** on page **Error! Bookmark not defined..**



## Configuring network roaming on a Raspberry Pi

Normally the network is configured by the Rpi-imager software when creating the SD-card. However, you may wish to add a second or third Wi-Fi access point to enable Wi-Fi roaming for example between your home and office.

To see what Wi-Fi access points are available run the following **iwlist** command.

```
$ sudo iwlist scan | grep ESSID
lo      Interface doesn't support scanning.

          ESSID: "EE-K7TZ9R"
          ESSID: "Garden Office"
          ESSID: "WATERSTONES"
          ESSID: "TP-Link_D7D4"
          :
```

This will display all available Wi-Fi access points available in your immediate vicinity. You will of course only be able to connect to those networks that you have a password for.

## Bullseye OS network configuration

Edit the **/etc/wpa\_supplicant/wpa\_supplicant.conf** configuration file and add a second network definition.

```
$ cat /etc/wpa_supplicant/wpa_supplicant.conf
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
country=GB

network={
    ssid=""
    psk=""
    key_mgmt=WPA-PSK
}

network={
    ssid=""
    psk=""
    key_mgmt=WPA-PSK
}
```

## Bookwork OS network configuration using network manager

Bookworm doesn't use **wpa\_supplicant.conf** but now uses Network Manager using the following **nmcli** commands.

```
sudo nmcli radio wifi on
sudo nmcli dev wifi connect <wifi-ssid> password "<network-password>"
```

Each Wi-Fi network has a separate config file in **/etc/NetworkManager/system-connections/** in a well configured system these will have a 256bit WPA PSK rather than a plain text passphrase.

## /etc/NetworkManager/system-connections/preconfigured.nmconnection

```
[connection]
id=preconfigured
uuid=026304ea-6b98-4e52-8765-0e71125962c8
type=wifi
[wifi]
mode=infrastructure
ssid=EE-K7TZ9R
hidden=false
[ipv4]
method=auto
[ipv6]
addr-gen-mode=default
method=auto
[proxy]
[wifi-security]
key-mgmt=wpa-psk
psk=5e3415b308da9e7cb5c633a5dd78d597c77262e73473f63c93ba3552f7cadf21
```

To configure a second Wi-Fi connection use **nmcli**:

**sudo nmcli dev wifi connect <SSID> password <password>**

Where **SSID** is the name of the second Wi-Fi access point (router or repeater) and password is an encrypted key or password. Note that a plain text password must be enclosed in quotes ("").

```
$ sudo nmcli dev wifi connect TP-Link_D7D4 password "89157405"
```

The above command if successful will create a file called in **TP-Link\_D7D4.nmconnection** in the directory **/etc/NetworkManager/system-connections/** as shown in the following example.

```
[connection]
id=TP-Link_D7D4
uuid=4ad31ca8-64ce-41ad-bcb2-34b8b7447117
type=wifi
interface-name=wlan0
timestamp=1725363073

[wifi]
mode=infrastructure
ssid=TP-Link_D7D4

[wifi-security]
auth-alg=open
key-mgmt=wpa-psk
psk=88358475

[ipv4]
method=auto

[ipv6]
addr-gen-mode=default
method=auto

[proxy]
```

## Upgrading the Music Player Daemon

The Raspberry Pi usually is installed with the **Raspberry Pi OS** (Formerly **Raspbian**) operating system. At the time of writing the latest version of **Raspbian** is called **Bookworm** which uses version **5.10.17** of the Linux kernel or later.



**Note:** The MPD version described in this procedure is **0.23.14** whilst the version released with Bookworm is **0.23.12** and is already out-of-date. The latest version of MPD is much more stable. Due to their age, it is impossible to get support on the older versions of MPD.

There are two methods of updating MPD.

1. Download a ready-made Debian package **mpd\_0.23.14\_armhf.deb** (or latest version)
2. Download the latest MPD source, then compile and install it.

### Installing the MPD upgrade from a Debian package

Bookworm currently installs MPD version **0.23.12** and it is questionable if it is worth upgrading to **0.23.14**. However, you can optionally upgrade it.

The **mpd\_0.23.14\_armhf.deb** upgrade package is currently available from the **package's** directory on the Bob Rathbone Web site. This is an upgrade; **mpd** and **mpc** as supplied with the OS must be installed first. There are 32 and 64-bit versions available.

```
$ sudo apt install mpd mpc
```

Answer yes 'y' when asked to continue.

Install the required libraries. Some of these may already be installed if using the desktop version. Execute all three lines below. Do not copy the \$ sign in the commands below.

```
$ sudo apt install libsidutils0 libresid-builder0c2a libcurl3-gnutls  
$ sudo apt install libaudiofile1 libyajl2  
$ sudo apt install libiso9660-11 libzip-0-13 libao4  
$ sudo apt install libadplug-2.3.3-0
```

Download and install the appropriate MPD (32 or 64-bit software) depending on your system's architecture:

#### 32-bit MPD package for Bookworm

```
$ wget https://bobrathbone.com/raspberrypi/packages/mpd_0.23.14_armhf.deb  
$ sudo dpkg -i mpd_0.23.14_armhf.deb
```

#### 64-bit MPD package for Bullseye

```
$ wget https://bobrathbone.com/raspberrypi/packages/mpd_0.23.13_armhf.deb  
$ sudo dpkg -i mpd_0.23.13_armhf.deb
```

#### 32-bit MPD package for Bookworm

```
$ wget https://bobrathbone.com/raspberrypi/packages/mpd_0.23.14_arm64.deb
```

```
$ sudo dpkg -i mpd_0.23.14_arm64.deb
```

Ignore the warnings for existing **/var/log/mpd** and **/var/lib/mpd** directories.

Reboot the Raspberry Pi to enable the new version.

```
$ sudo reboot
```



Note: For some unknown reason when **mpd.socket** is run from this package it changes the ownership of the **/var/run/mpd** directory from **mpd:audio** to **root:root** which causes **mpd.service** to fail with a permissions error. To correct this problem the following line has been added to **mpd.service** just before the **ExecStart** statement.

```
ExecStartPre=-/bin/chown mpd:audio /var/run/mpd
```

There is no need to add this line yourself as it is already added by the installation script. If the Radio/MPD service fails to start see the section called **Music Player Daemon won't start** on page 207.

### Installing the latest MPC client

The **mpc\_0.34\_armhf.deb** upgrade package is currently available from the **packages** directory on the Bob Rathbone Web site. The latest version of **mpd** must be installed first as shown in the previous section.

Download the MPC software upgrade package.

```
$ wget https://bobrathbone.com/raspberrypi/packages/mpc_0.34_armhf.deb
```

Now install with **dpkg**.

```
$ sudo dpkg -i mpc_0.34_armhf.deb
```

Ignore the warnings for existing **/var/log/mpd** and **/var/lib/mpd** directories.

Reboot the Raspberry Pi.

```
$ sudo reboot
```

### Compiling and installing the latest Music Player Daemon

How to download and compile the latest version of Music Player daemon is described in the following document:

<https://bobrathbone.com/raspberrypi/documents/Compiling%20and%20installing%20MPD.pdf>

## X-Windows configuration for the radio software

This section is for information and troubleshooting only and only refer to X-windows configuration to start the Graphics versions of the radio software. There are three possible sets of configuration files for the X-Windows system. These are:

1. The legacy **X11** system
2. **Wayland** using the **wayfire** windows compositor
3. **Wayland** using the **labwc** windows compositor (Currently **Bookworm 32-Bit** only)

Which one is used is selected in the *Advanced configuration* in the **raspi-config** utility. Select option **6 Advanced Options** and then **A6 Wayland**.

All three options for the radio are normally configured using the **configure\_radio.sh** script and by selecting **HDMI or touch screen** as the display.

```
$ cd /usr/share/radio  
$ ./configure_radio.sh
```

### Legacy X11 configuration

**X11** uses the **/home/{USR}/.config/lxsession/LXDE-{USR}/autostart** configuration file to run **vgradio.py** or **gradio.py**.

For example **/home/pi/.config/lxsession/LXDE-pi/autostart**

```
@lxpanel --profile LXDE-pi  
@pcmanfm --desktop --profile LXDE-pi  
@xscreensaver -no-splash  
@point-rpi  
@sudo /usr/share/radio/vgradio.py
```

### Wayland and wayfire configuration

**Wayland** and **wayfire** is configured in the **/home/{USR}/.config/wayfire.ini** file

```
: {output omitted}  
:  
[autostart]  
radiod = sudo /usr/share/radio/vgradio.py
```

### Wayland and labwc configuration

**Wayland** and **labwc** is configured in the **/home/{USR}/.config/labwc/autostart** file which contains an entry for either **gradio.py** or **vgradio.py**.

```
sudo /usr/share/radio/vgradio.py &
```

## Comitup Wi-Fi roaming

Comitup works by configuring a Wi-Fi Hot-Spot on the Raspberry Pi which can be connected to using either a Mobile Phone, tablet or PC via a Web Interface. The web interface then allows selection of any available Wi-Fi network. More information on **comitup** can be found at:

<https://davesteele.github.io/comitup>

There are two ways of installing comitup:

- Download the ready-made image from <https://davesteele.github.io/comitup>
- Install the comitup Rasbian package using `apt install`

### Installation from the Comitup Image

To burn onto an SD card for the Raspberry Pi, first download the image from the above web site. There are currently two versions namely **Bookworm Desktop** and **Bookworm Lite**. Download the zip file to your PC (Not the torrent or Magnet versions). You should find the file in your **Downloads** folder:

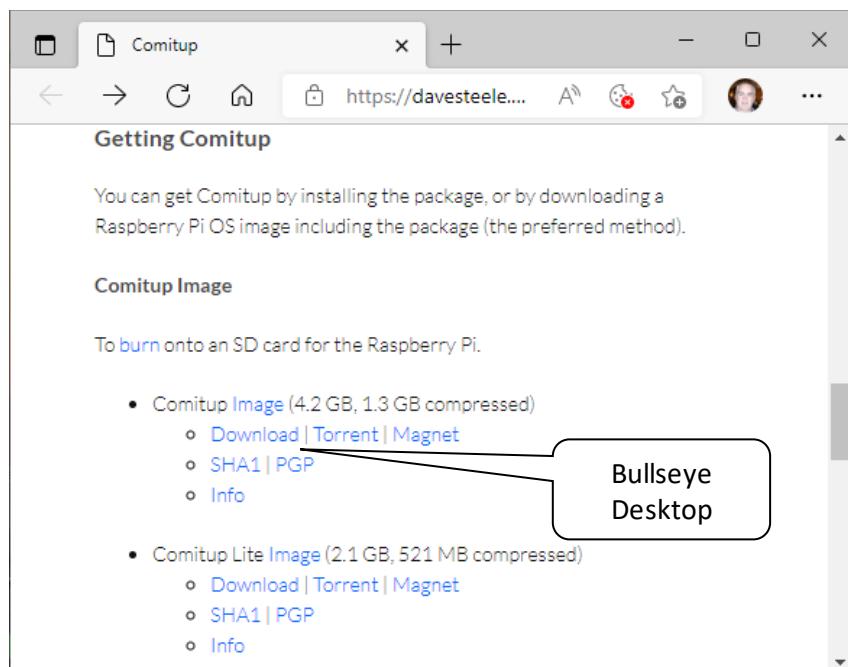
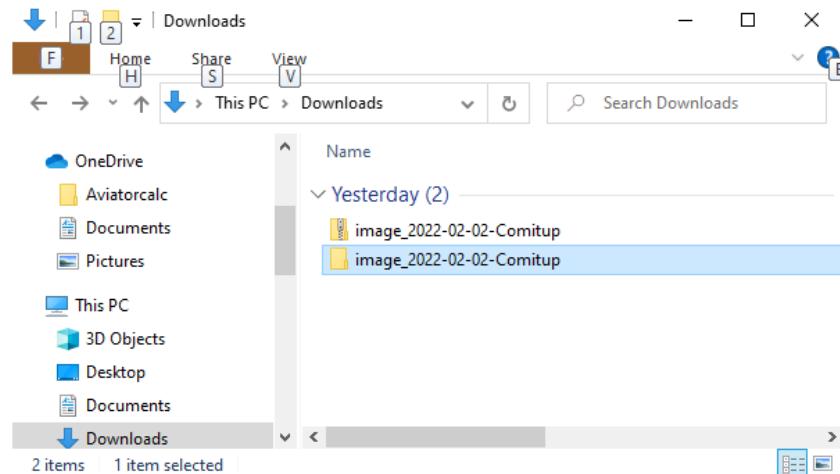
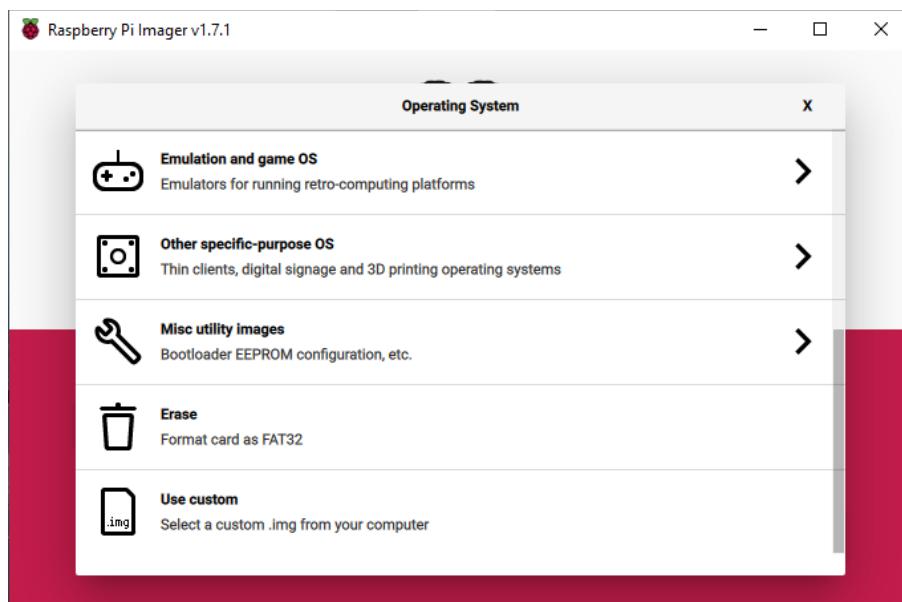


Figure 26 Downloading the comitup image

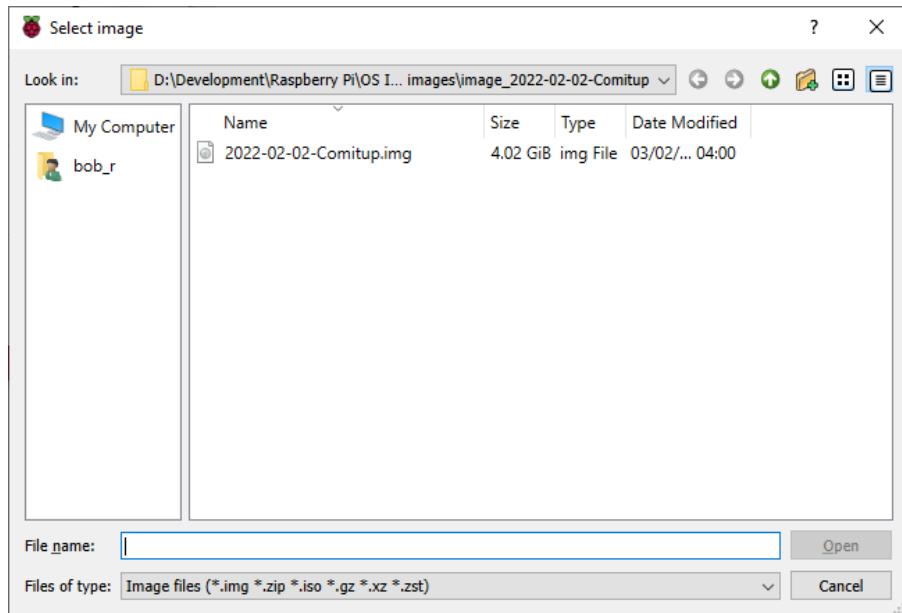
Using **Winzip** or **7Zip** unzip the image file:, Typically this will unzip the compressed image to a directory with the same name as the zip file:



Now open the Raspberry Pi Imager software (See **Error! Reference source not found.** on page **Error! Bookmark not defined.** for more detail). Select “CHOOSE OS” and then scroll down to “Use custom”.



This will open a browser window. Go to your downloads browser and select the img file that was just unzipped.



Click and open the image file. Next select “CHOOSE STORAGE” and select your USB device. Finally select “WRITE”. Don’t change the hostname at this stage. It will have already been set to comitup-xxx where xxx is a three-digit number and will vary for example, comitup-123.

### Installing from a Raspbian package



At the moment this procedure does not seem to produce a working comitup service. It used to work and is being investigated. Please use the procedure described in *Installation from the Comitup Image* on page 120.

Now install required libraries:

```
$ sudo apt install python3-tabulate firewalld libteam-utils
```

Install the Comitup package:

```
$ sudo apt install comitup
```

The current **/etc/wpa\_supplicant/wpa\_supplicant.conf** must be moved out of the way and control handed over to comitup. Rename

```
$ cd /etc/wpa_supplicant  
$ sudo mv wpa_supplicant.conf wpa_supplicant.conf.save
```

Comitup requires access to port TCP 53 however the system-resolved service uses that port so must be disabled.

```
$ sudo systemctl disable systemd-resolved
```

The current Web interface (If installed) must be placed under comitup control. In this case it is the Apache web interface that must be disabled and configure to be controlled by comitup.

```
$ sudo systemctl disable apache2.service
```

Add the following line to the **/etc/comitup.conf** file

```
web_service: apache2.service
```

Enable comitup and associated web service.

```
$ sudo systemctl enable comitup comitup-web
```

Reboot the Raspberry Pi

```
$ sudo reboot
```

After reboot there will no longer be a connection via the normal Wi-Fi Interface. Use a mobile phone or PC and use the Wi-Fi configuration screen to select the comitup

Once logged back in it is possible to check the status of the comitup and comitup-web services.

```
$ sudo systemctl status comitup
● comitup.service - Comitup Wi-Fi Management
  Loaded: loaded (/lib/systemd/system/comitup.service; enabled; vendor
  preset: enabled)
    Active: active (running) since Wed 2020-06-10 19:43:15 BST; 11min ago
      Docs: man:comitup(8)
    Main PID: 493 (comitup)
       Tasks: 2 (limit: 2061)
      Memory: 26.3M
     CGroup: /system.slice/comitup.service
             └─493 /usr/bin/python3 /usr/sbin/comitup
                  ├─731 dnsmasq --conf-file=/usr/share/comitup/dns/dns-hotspot.conf
                  └─731 dnsmasq --conf-file=/usr/share/comitup/dns/dns-hotspot.conf
--interface=wlan0
```

```
$ sudo systemctl status comitup-web
● comitup-web.service - Comitup Web Service
  Loaded: loaded (/lib/systemd/system/comitup-web.service; static; vendor
  preset: enabled)
    Active: active (running) since Wed 2020-06-10 19:54:22 BST; 3s ago
      Docs: man:comitup-web(8)
    Main PID: 2389 (comitup-web)
       Tasks: 1 (limit: 2061)
      Memory: 12.8M
     CGroup: /system.slice/comitup-web.service
             └─2389 /usr/bin/python3 /usr/sbin/comitup-web
```

To use comitup enter <http://10.41.0.1> into a web browser running on a PC, tablet or mobile telephone.

Notes:

The IP address is defined in **/usr/share/comitup/comitup/nm.py**

### *Disabling comitup*

If for any reason comitup is a problem then it can be disabled and the standard Wi-Fi configuration restored with the following instructions.

Disable comitup and associated web service.

```
$ sudo systemctl disable comitup comitup-web
```

Restore **/etc/wpa\_supplicant/wpa\_supplicant.conf** (This assumes that this was previously a working configuration)

```
$ cd /etc/wpa_supplicant  
$ sudo mv wpa_supplicant.conf.save wpa_supplicant.conf
```

Re-enable the **system-resolved** service.

```
$ sudo systemctl enable systemd-resolved
```

Re-enable the **apache2** service.

```
$ sudo systemctl enable apache2.service
```

Reboot the Raspberry Pi

```
$ sudo reboot
```

## Chapter 7 – Configuration

<b>Contents chapter 7</b>	<b>Page</b>
Configuration utility radio-config	126
Configuring the HDMI or Touch Screen	126
Configuring GPIO outputs	127
Configuring the remote control activity LED	128
Specifying the action for exit/shutdown action	129
Changing the date format	129
Configuring the IQaudio Cosmic controller and OLED	130
Configuring the Adafruit LCD backlight colours	131
Configuring startup mode for Radio or Media player	131
Configuring the volume display	132
Configuring the volume range	132
Configuring the MPD client timeout	133
Changing the display language	133
Configuring Music Player Daemon CODECs	134
Configuring an RSS feed	134
Configuration of the mute button action	135
Configuring the Alsa Equalizer	135
Configuration of the FLIRC USB dongle	138
Configuring the display scroll speed	142
Configuring Russian/Cyrillic text	142
Configuring European languages	143
Configuring the Internet Check URL and port	143
Configuring MPD interface	144
Configuring the remote-control parameters	144
Configuring the radio software UDP server listening port	144
Configure the I2C bus number	145
Configuring character LCD lines and width.	145
Configuring TFT display fonts	145
Configuring Rotary Encoders	145
Configuring the Menu rotary switch	146
Configuring soft volume (SoftVol)	147

## Configuration utility radio-config

This section covers additional configuration of the radio. Most items are configured using the **radio-config** utility. To enter the radio configuration utility, first log into the Raspberry Pi and enter the **radio-config** command:

```
$ radio-config
```

This will display the top-level configuration menu.

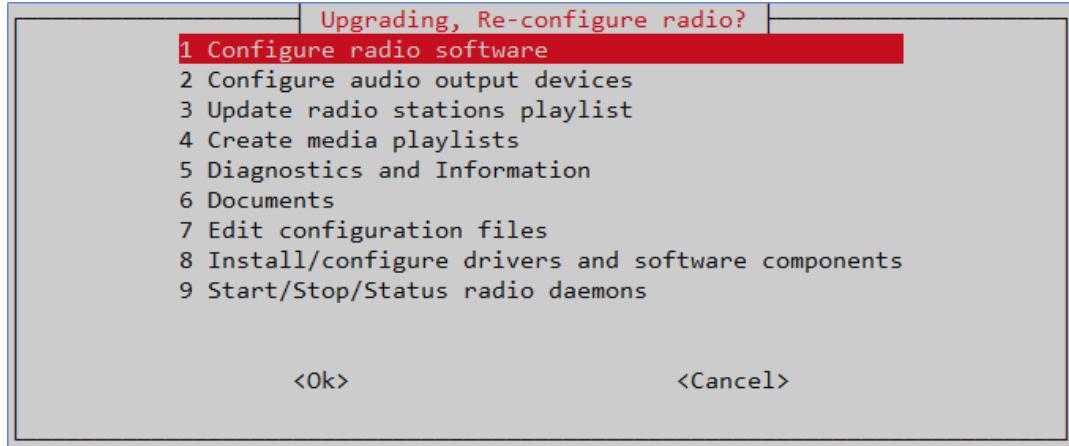


Figure 27 Radio configuration menu

## Configuring the HDMI or Touch Screen

In the **/etc/radiod.conf** file there is a section called [SCREEN] as shown below. This is the HDMI/Touch Screen default configuration.

```
# Graphics (touch screen) screen settings
[SCREEN]
# Size is in pixels. Supported is 800x480 (7" screen) or 720x480(3.5" screen)
# or 480x320 (2.8" or 3.5" screen) or 1024x600 (Maximum)
# Also see framebuffer_width and framebuffer_height parameters in
/boot/firmware/config.txt
screen_size=800x480
fullscreen=yes

# Screen save time in minutes, 0 is no screen saver
screen_saver=0

# Title %V = version %H = hostname
window_title=Bob Rathbone Internet Radio Version %V - %H
window_color=turquoise
banner_color=white
labels_color=white
display_window_color=lightblue
display_window_labels_color=black
slider_color=darkgreen
display_mouse=yes

# Wallpaper backgrounds. See /usr/share/scratch/Media/Backgrounds
wallpaper=/usr/share/scratch/Media/Backgrounds/Nature/beach-malibu.jpg

# Set date format for graphic screen
dateformat=%H:%M:%S %A %e %B %Y

# Allow switching between vgradio and gradio
switch_programs=yes

# The following is specific to the vintage graphical radio
scale_labels_color=white
```

```
stations_per_page=40
display_date=yes
display_title=yes
```

<u>Parameter</u>	<u>Explanation</u>
<b>fullscreen</b>	Set to <b>yes</b> or <b>no</b> . If using a large HDMI monitor or TV set to <b>no</b> .
<b>window_title</b>	Title to display in the desktop window if <b>fullscreen=no</b>
<b>window_color</b>	Window background colour if <b>wallpaper</b> (See below) not specified.
<b>banner_color</b>	This is the colour of the time and date banner.
<b>labels_color</b>	This is the colour of the radio and MPD option labels.
<b>display_window_color</b>	This is the background colour of the station/track display window.
<b>display_window_labels_color</b>	Colour of the display window text.
<b>slider_color</b>	The color of the slider in the slider window next to the search window.
<b>display_mouse</b>	Future use – hide mouse <b>yes/no</b> .
<b>wallpaper</b>	Background wall paper. Any jpeg or gif file can be specified. See directory <code>=/usr/share/scratch/Media/Backgrounds</code> .
<b>dateformat</b>	The format for displaying the time and date banner.

The following settings are specific to the vintage graphical radio:

<b>scale_labels_color</b>	This sets the color of the station names on the tuning scale
<b>stations_per_page</b>	This is the maximum number of stations that will be displayed on each page.
<b>display_date</b>	Display the date at the top of the screen yes/no
<b>display_title</b>	Display the station title at the bottom of the screen yes/no



All parameters use the American spelling for color and not the British spelling.

The **wallpaper** parameter overrides the **window\_color** parameter.

The parameters allow any theme to be configured for the HDMI or Touch Screen window.

## Configuring GPIO outputs

Apart from changing the **down\_switch** GPIO setting to be compatible with the **HiFiBerry DAC** it is not normally necessary to change the GPIO settings for the switches, rotary encoders or LCD display connections. The default settings match the wiring configuration shown in Table 17 on page 314. Unless here is a need to change the GPIO configuration skip this section.

All switches, rotary encoders and LCD display settings are configurable in the **/etc/radiod.conf** file.



If the **GPIO** assignments are changed in the **/etc/radiod.conf** file then these must match the actual physical wiring for your radio project. The configuration in **/etc/radiod.conf** uses **GPIO** numbers and not physical pin numbers. For example, **menu\_switch=17** is using **GPIO 17** (Physical pin 11).

## Switches and rotary encoders GPIO assignments

The default switch settings including the rotary encoders are shown below. Normally there is no need to change these as they are set by the **configure\_radio.sh** program.

```
# Switch settings for 40 pin version (support for IQaudIO)
menu_switch=17
mute_switch=4
up_switch=24
down_switch=23
left_switch=14
```

```
right_switch=15
```

### Disabling button or rotary encoder GPIOs

It is possible to disable a button or rotary encoder GPIO configuration. For example, if you are building a radio with an amplifier with its own volume control you may not need the music player daemon volume control as well. In this case set these to 0 to disable them.

```
menu_switch=17
up_switch=24
down_switch=23
mute_switch=0
left_switch=0
right_switch=0
```

This will free the GPIOs originally configured for the volume control for other uses (4, 14 and 15 in this example).

### LCD display GPIO assignments

The default LCD settings for a 40-pin Raspberry Pi are shown below. Again, there is no need to change these unless your wiring is different. Again, setting these to 0 disables the outputs. Again, the numbers below refer to **GPIO** number assignments and not physical pin numbers.

```
# LCD GPIO connections for 40 pin version of the radio
lcd_select=7
lcd_enable=8
lcd_data4=5
lcd_data5=6
lcd_data6=12
lcd_data7=13
```

### Configuring button interface with pull up resistors

This applies to the radio with push buttons only. The original design of the radio wires the push buttons from low to high (GND 0V to 3.3V). It is now more usual to configure the buttons to operate from high to low (3.3V to GND 0V). This is the case for rotary encoders. It may be easier to wire up the buttons to GND 0V. In such a case it is necessary to configure the **pull\_up\_down** parameter in **/etc/radiod.conf** to 'up' as shown below.

```
# Pull GPIO up/down internal resistors (Applies button interface only).
# Default:down
pull_up_down=up
```

### Configuring the remote control activity LED

It is useful to have an activity LED which flashes every time the remote control is pressed. How to wire the activity LED is shown on page **Error! Bookmark not defined..** Which pins you connect to will depend on the type of radio you are building. **Error! Reference source not found.** on page **Error! Bookmark not defined.** shows the required LED connections. Boards such as the the **AdaFruit RGB plate** will need a 40-pin Raspberry PI as all the first 26-pins are occupied but the plug in card.

Configure the LED in **/etc/radiod.conf** for to pin 11 GPIO 23 for all versions except AdaFruit plate or Vintage radio. For Adafruit RGB plate configure either **remote\_led=0** (No LED) or GPIO 13 (pin 33).

For the vintage radio use GPIO 23 (Pin 16). See the section called **Error! Reference source not found.** on page **Error! Bookmark not defined..**

```
# Output LED for remote control, default GPIO 11 (pin 23) or
# GPIO 13 (pin 33) for AdaFruit plate
# remote_led=0 is no output LED
remote_led=11
```

### Testing the remote control activity LED

It is possible to test the activity LED with the **irreventd.py** program.

```
$ cd /usr/share/radio/
$ ./irreventd.py flash
```

The activity LED should flash about six times. If not then check that the **remote\_led** parameter in the **/etc/radiod.conf** configuration file is correctly set and that the activity LED is correctly wired (See LED wiring on page **Error! Bookmark not defined.**).

### Specifying the action for exit/shutdown action

The radio can be exited by holding the menu button in for three seconds. The action depends upon the **exit\_action** parameter. This is either set to **stop\_radio** or **shutdown**.

```
# Action on exiting radio. Stop radio only or shutdown the system
# exit_action=stop_radio
exit_action=shutdown
```



From version 7.3 onwards the graphic versions of the radio have an exit/shutdown button (as shown on the left). The exit/shutdown button is displayed by default. This can be changed by changing it to no.

```
display_shutdown_button=no
```

If the exit button is enabled and clicked, the radio program will either exit to the operating system (handy during testing) or it will perform a system shutdown.

It is also possible to specify the shutdown command with the **shutdown\_command** parameter in **/etc/radiod.conf**. The default command is shown below.

```
shutdown_command="sudo shutdown -h now"
```

This can be replaced with required command for example "x735off" for a X735 V2.5 shield. Put the command between apostrophes for example "x735off".

### Changing the date format

The date is configured in the **/etc/radiod.conf** file using the **dateformat** parameter:

```
dateformat=%H:%M %d/%m/%Y
```

The default configuration is: **%H:%M %d/%m/%Y**

Where: %H = Hours, %M=Minutes, %d= Day of the month, %m=month, %Y=Year

It is possible to change the date format (for example for the United States) by changing the format. Some valid formats are:

%H:%M %m/%d/%Y	US format
%H:%M %d-%m-%Y	Minus sign as date separator
%d/%m/%Y %H:%M	Reverse date and time
%H:%M %m%d	Short date display for Olimex OLED

Seconds can also be displayed:

%H:%M:%S %d/%m/%Y Display seconds (%S) on 20 character displays only

## Configuring the IQaudIO Cosmic controller and OLED

The **configure\_radio.sh** program can be used to set the configuration to use the 128 by 64 pixel OLED supplied with the Cosmic controller. This sets the **display\_type** parameter to **OLED\_128x64**.

```
display_type=OLED_128x64
```

When running with the Cosmic controller OLED screen there are two relevant settings in the **/etc/radiod.conf** file which are not set by the **configure\_radio.sh** program.

The OLED display can be flipped vertically by setting the **flip\_display\_vertically** parameter to yes.

```
flip_display_vertically=yes
```

Note: If upgrading you will need to add this parameter to the **[RADIO]** section of the **/etc/radiod.conf** file.

The three LEDs on the Cosmic Controller board are driven by the *status\_led\_class.py* program. This class was originally written for the Vintage radio but is now also used with this board. Configure the following parameters in **/etc/radiod.conf** as shown below:

```
rgb_red=14  
rgb_green=15  
rgb_blue=16
```

The left LED means an ERROR, the middle LED means NORMAL operation and the right-most is BUSY.

If you want to switch off the status LEDs then set them to 0. However, GPIO 15 is switched on automatically at boot time. To switch it off add the following two lines to **/etc/rc.local**.

```
gpio -g mode 15 out  
gpio -g write 15 0
```

Set the date format so that it displays fits the display.

```
dateformat=%H:%M %d%m
```

## Configuring the Adafruit LCD backlight colours

Some Adafruit displays such as the **rgb-negative Adafruit LCD** allow changing the colour of the backlight. This is configurable in the **/etc/radiod.conf** file. The colours that can be used are RED, GREEN, BLUE, YELLOW, TEAL, VIOLET and WHITE or OFF (No backlight).

### The colour settings in the /etc/radiod.conf file

```
# Background colours (If supported) See Adafruit RGB plate
# options OFF, RED, GREEN, BLUE, YELLOW, TEAL, VIOLET, WHITE
bg_color=WHITE
mute_color=VIOLET
shutdown_color=TEAL
error_color=RED
search_color=GREEN
info_color=BLUE
menu_color=YELLOW
source_color=TEAL
sleep_color=OFF
```



**Note:** Always use the American spelling ‘color’ in all commands and not the British spelling ‘colour’.

## Configuring startup mode for Radio or Media player

The radio can be configured to start in **RADIO**, **MEDIA**, **LAST** modes or a **playlist** name. The default is **\_Radio**. To change this, edit **/etc/radiod.conf** and change the **startup=\_Radio** parameter to **RADIO**, **MEDIA** or **LAST** to start the radio with the last playlist used in the previous run. For example:

```
# Startup option either RADIO,MEDIA or LAST a playlist name
startup=MEDIA
#startup=Radio
```

Alternatively, the radio can be configured to load a specific playlist. To display the available playlists run the following command:

```
$ mpc lsplaylists
USB_Stick
UK_stations
Beatles
Radio
:
```

To configure the radio to start with a specific playlist change the **startup=** statement.

```
#startup=RADIO
startup=USB_Stick
```

If you configure **startup=RADIO** the program will load the first available Radio playlist. Likewise if you configure **startup=MEDIA** the program will load the first available Media playlist.

## Configuring the volume display

The volume can be displayed as either text or as a series of blocks. This is configured in **/etc/radiod.conf** using the **volume\_display** parameter. The default is text.

```
# Volume display text or blocks  
volume_display=text
```

```
12:01 30/08/2017  
WPJR Country  
Lonestar - Mr. Mom  
Volume 75
```

To display the volume as a series of blocks change this to 'blocks':

```
volume_display=blocks
```

```
12:01 30/08/2017  
WPJR Country  
Lonestar - Mr. Mom  
███████████
```



If the timer or alarm functions are being used then the volume display reverts back to text display so as to allow display of the alarm or timer values.

## Configuring the volume range

This setting affects the volume control sensitivity.

The MPD daemon has a volume range from 0 to 100. The volume is incremented or decremented by one each time the volume button is pressed or rotary encoder is turned a notch. This means a lot of turns of the knob or pushes of the button to change the volume the full range. Also, different devices are more sensitive than others.

For example, the Adafruit plate version allows very rapid change of the volume and the default range of 0 to 100 is not a problem. The rotary encoder version of the radio requires a lot of twisting of the volume knob to get from 0 to 100.

This **volume\_range** parameter allows you to set the volume range to increase the sensitivity of the volume control as shown below. For example, if the volume range is set to 20 you will see the volume displayed from 0 to 20 however the MPD volume is incremented by 5.

Increment = 100 / Volume range. For example,  $100/20 = 5$

So, if the volume displayed on the LCD is 10 and the range is 20, then the MPD volume is  $10 \times 5 = 50\%$ .

The volume range is configured in **/etc/radiod.conf** configuration file using the **volume\_range** parameter:

```
# Volume range 10, 20, 25, 50 or 100
volume_range=20
```

Ideally you should choose a volume range number that divides into 100 equally as shown above however other values will work.

## Configuring the MPD client timeout

When the radio program tries to connect to radio stream it will time out after so many seconds. In all previous versions this timeout was hard set to ten seconds. This is configurable from three to fifteen seconds using the **client\_timeout** parameter in **/etc/radiod.conf**. The default is ten seconds.

```
# MPD client timeout from 2 to 15 seconds default 10
client_timeout=10
```

## Changing the display language

The language file is stored in **/home/pi/radio/language** directory. This contains the text that will be either displayed or spoken. The default language is English. The **language.en** file is copied to **/var/lib/radiod/language**. The language file (if present) file is loaded during start-up of the radio. If not present the default English text is used.

The format of each entry in the language file is:

<label>:<text>

For example:

**select\_source: Select source**

It is possible to display all the labels and text by running **language\_class.py**.

```
$ cd /usr/share/radio
$ ./language_class.py
airplay: Airplay
alarm: Alarm
alarmhours: Alarm hours
alarmminutes: Alarm minutes
colour: Colour
consume: Consume
current_station: Current station
information: Information display
loading: Loading
loading_media: Loading media library
loading_radio: Loading radio stations
main_display: Main
media_library: Media library
menu_find: Find
menu_option: Menu option:
menu_search: Search
:
```

## Creating a new language file

To create a new language file by running the **language\_class.py** program and redirecting the output to a file called **language.<new>** where <new> is the country code. For example, to create a language file in Dutch, the country code is **nl**.

```
$ cd /usr/share/radio  
$ sudo ./language_class.py > language/language.nl
```

Now edit the text (Not the labels) in the language/language.nl file. It isn't necessary to change every message. Lines beginning with # are for any comments.

```
# Nederlands text for uitspraak  
main_display: Hoofd menu  
search_menu: Zoek menu  
select_source: Media selecteren  
options_menu: Opties menu  
rss_display: RSS beeld  
information: Informatie beeld  
the_time: De tijd is  
loading_radio: Radio zenders laden  
loading_media: Media laden  
search: Zoek  
source_radio: Internet Radio  
source_media: Muziek selectie  
sleeping: Slaapen
```

Finally copy the new language file to **/var/lib/radiod/language** (Omit the country code) and restart the radio.

```
$ sudo cp language/language.nl /var/lib/radiod/language  
$ sudo systemctl restart radiod
```

## Configuring Music Player Daemon CODECs

It is possible to configure the Music Player Daemon CODECs list in **/etc/radiod.conf**. There is a parameter called CODECS with the CODECs list between quotes.

```
# Codecs list for media playlist creation (Run 'mpd -V' to display others)  
CODECS="mp3 ogg flac wav"
```

To see what CODECs are available in MPD run the following command.

```
$ mpd -V
```

A CODEC defines the method for encoding and decoding a digital stream. CODEC is a portmanteau of Coder-Decoder. See Wikipedia article <https://en.wikipedia.org/wiki/Codec> for more information on CODECS.

## Configuring an RSS feed

To display an RSS feed it is necessary to create the **/var/lib/radiod/rss** file with a valid RSS URL. For example:

```
http://feeds.bbci.co.uk/news/uk/rss.xml?edition=int
```

The above is the RSS for the BBC news feed however any valid RSS feed may be used. If the **/var/lib/radiod/rss** is missing or contains an invalid RSS URL then this mode is skipped when stepping through the menu. The software comes with a valid BBC RSS feed file in the **/var/lib/radio/rss** file. You can test the feed first by pasting it into your PC's Web browser URL and pressing enter.

If configured, the RSS feed will be automatically displayed by stepping through the menus.

### Configuration of the mute button action

When the mute button is pressed the volume is reduced to zero and the stream is either paused or stopped depending upon the setting of the **mute\_action** parameter in **/etc/radiod.conf**.

```
# Action when muting MPD. Options: pause(Stream continues but not processed)
# or stop(stream is stopped)
# mute_action=stop
mute_action=pause
```

**pause** – The radio stream continues to be downloaded but is not processed (default)

**stop** – The radio stream is stopped altogether.

Both have their own characteristics. When the radio is un-muted using the **stop** option it will play the old remaining stream in its buffer for about 30 seconds before jumping to the new live stream. It does have the advantage that no Internet bandwidth is being consumed.

The **pause** option continues to download the radio stream, but not processing it and consuming Internet bandwidth. When the radio starts playing again MPD simply starts processing the live stream again. There is no buffer to empty so no “jumping” to the new stream. The behaviour of pause and stop is controlled by the Music Player daemon over which the author has no control.

### Configuring the Alsa Equalizer



**Note:** The author of the radio software does not currently have a configuration that is compatible with either **Pulseaudio** or **Bluetooth** audio. Devices. If you wish to use any of these then you cannot currently use the Alsa equalizer. This may change in a future release.

Install the Alsa plugin with **apt**:

```
$ sudo apt install -y libasound2-plugin-equal
```

Amend the “device” parameter in the **audio\_output** block in **/etc/mpd.conf** configuration file.

```
audio_output {
    type          "alsa"
    name          "IQAudio DAC+"
    device        "plug:plugequal"
    mixer_type   "software"
```

```
}
```

In the above example we are using an IQaudIO card but may be any sound card.

Save the existing **asound.conf** file just in case you need to restore the original file

```
$ sudo cp /etc/asound.conf /etc/asound.conf.save
```

Copy the **asound.conf.dist.equalizer** to **/etc/asound.conf**

```
$ cd /usr/share/radio/asound/
$ sudo cp asound.conf.dist.equalizer /etc/asound.conf
```

The new **/etc/asound.conf** file should look as shown below:

```
pcm.!default {
    type plug
    slave.pcm plugequal;
}
ctl.!default {
    type hw card 0
}
ctl.equal {
    type equal;
}
pcm.plugequal {
    type equal;
    slave.pcm "plughw:0,0";
}
pcm.equal {
    type plug;
    slave.pcm plugequal;
}
```

If your sound system is using card 1 (for example a USB audio device) then change the hardware settings in the above configuration to use card 1.

```
:
type hw card 1
:
slave.pcm "plughw:1,0";
```

It is also necessary to amend the card number in the **equalizer.cmd** file in the **/var/lib/radiod** directory

```
#!/bin/bash
# Change -c 0 to your alsamixer device number
lxterminal --command="/bin/bash -c 'sudo -H -u mpd alsamixer -c 1 -D equal'"
```

Reboot the Raspberry Pi.

```
$ sudo reboot
```

After reboot run the Alsa Equalizer as user **mpd**.

```
$ sudo -H -u mpd alsamixer -c 0 -D equal
```

If using card 1 change the **-c 0** parameter above to **-c 1**. The following screen will be displayed:

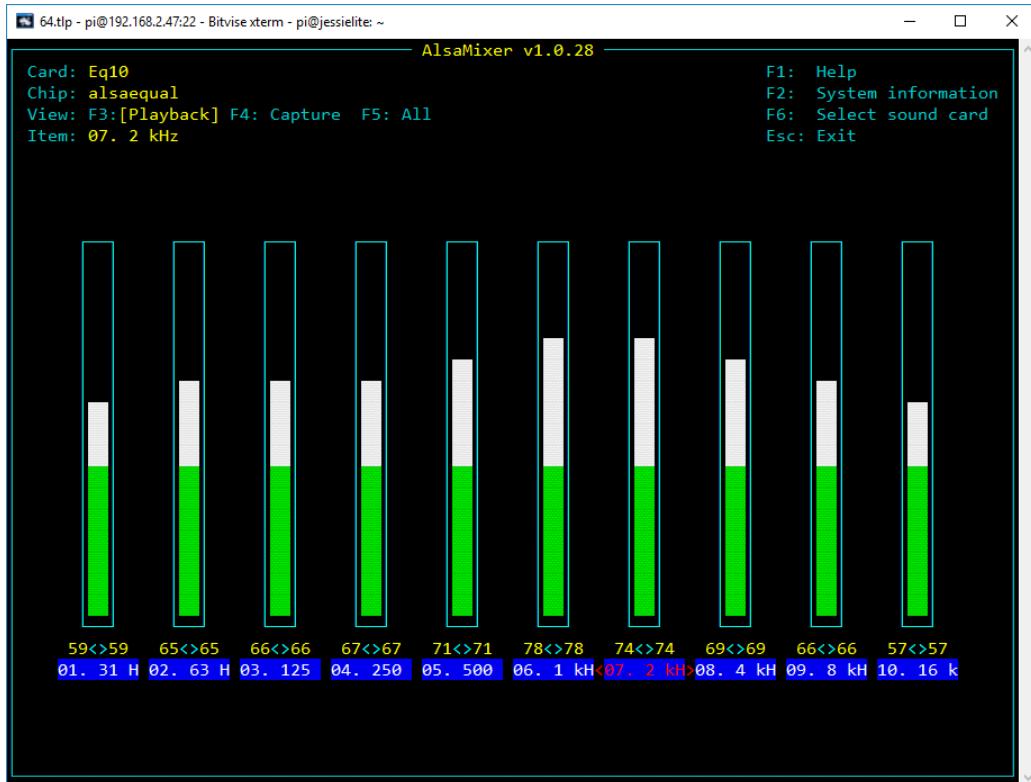


Figure 28 The Alsa

Use the Tab key to move along to the desired frequency to be changed. In this example, it is the <2KHz> block. Use the up and down arrows to adjust the level. The settings are saved in the **/var/lib/mpd/.alsaequal.bin** file. Changes to the sound should be heard.



**Note:** If you set a particular frequency value too high you will cause unpleasant distortion to the sound output.

### Disabling the Alsa equalizer

Restore the original **asound.conf** file:

```
$ cd /usr/share/radio/asound/  
$ sudo asound.conf /etc/asound.conf
```

Restore the original “device” parameter in the **audio\_output** block in **/etc/mpd.conf** configuration file.

```
audio_output {  
    type          "alsa"  
    name          "IQAudio DAC+"  
    device        "hw:0,0"  
    #device       "plug:plugequal"
```

```
        mixer_type      "software"
    }
```

Reboot the Raspberry Pi to restore the original sound configuration.

## Configuration of the FLIRC USB dongle

 **Note:** This configuration procedure is only for the HDMI or Touchscreen display of the radio. If using an LCD display see **Error! Reference source not found.** on page **Error! Bookmark not defined..**

First of all, install the **FLIRC** software as shown in the section called **Error! Reference source not found.** on page **Error! Bookmark not defined..**

Click on the left-hand program icon (A Raspberry) and select Accessories. In Accessories select **Flirc**. The following screen will be displayed. However, on a 7-inch touchscreen you may not be able to see the whole FLIRC window. In this case use the procedure called Configuring FLIRC from the command line on page 139. The first time you run this program it may ask you if you want to upgrade the firmware. Always upgrade the firmware:

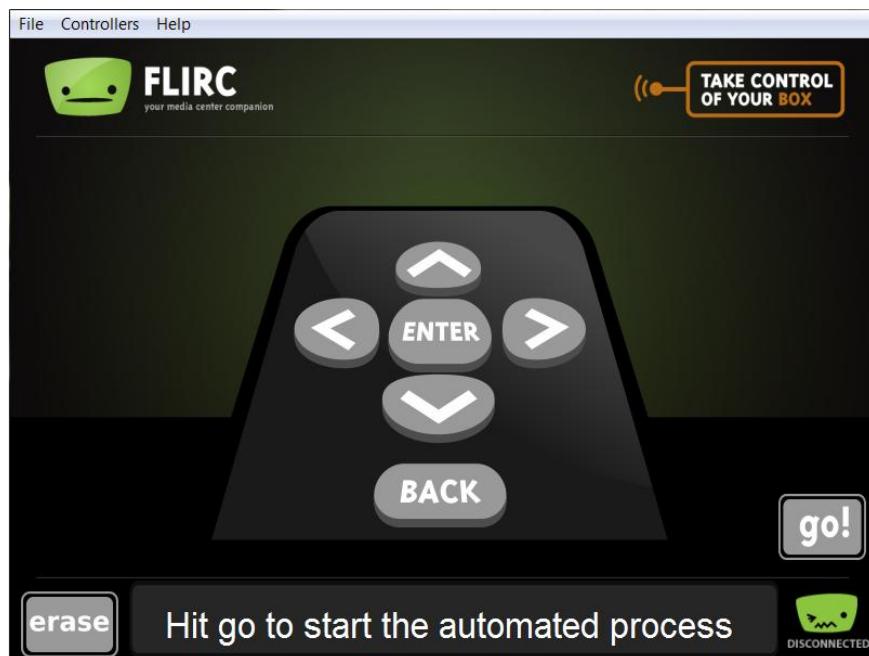


Figure 29 FLIRC setup program

On the **Controllers** drop down menu select the **Full Keyboard** controller.



**Figure 30 FLIRC keyboard controller**

Now map the buttons on the remote control to the keys shown in Table 6 Graphic screen keyboard command on page 161. For example, press the letter **m** on the above keyboard and then press the Mute button on the Remote Control. For volume control up press Shift key followed by the + key on the keyboard, then press the volume up button on the remote control. Do the same with the – key for volume down. Full instructions for configuring FLIRC are to be found at:

<https://flirc.gitbooks.io/flirc-instructions/>

### Configuring FLIRC from the command line

If using a small touchscreen there may not be enough room to see the Flirc screen. If so, do the following:

- 1) Amend the **fullscreen=yes** parameter to **fullscreen=no** in **/etc/radiod.conf**
- 2) Reboot the Raspberry PI
- 3) When rebooted open a terminal session on the desktop (Don't use remote SSH).
- 4) In the terminal window on the command line run the following:

```
$ flirc_util format
```

Now record the buttons:

```
$ flirc_util record up
Press any button on the remote to link it with '+'
```

'up' is the name of the key. Now press the Channel Up key. The following will be displayed:

```
Successfully recorded button.
```

Repeat the command for each key name.

They are:

pageup, pagedown,  
+, -,  
left, right,

```
up, down,  
return,  
l (small letter L), p, a,  
r, t, c, s, m and d.
```

In the case of the + and – keys press shift first, followed by the + or – key.  
Test and if necessary, repeat key-mapping. If configuring on an HDMI Television do not configure volume (+-) or mute (m) keys as the TV will provide these functions.

The configured keys can be displayed with the **flirc\_util keys** command, however this command may be missing from the latest version of **flirc\_util**.

```
$ flirc_util keys

Recorded Keys:
Index hash      key
----  ----
 0  7D14E297   down
 1  ED385097   up
 2  58C86297   right
 3  41787497   left
 4  BF8F6297   return
 5  AB616762   r
 6  2676D097   t
 7  B6536297   c
 8  9206E297   s
 9  590C3E97   l
10  E8E8D097   p
11  C49C5097   a
12  F1EFD097   e
13  B9F03963   escape
14  D1F15097   pageup
15  53DA6297   pagedown
16  9F5BE297   escape
17  A8DDF497   left_ctrl Q
18  66FFBE97   d
```

Saving the configuration:

```
flirc_util saveconfig my_flirc_config

Saving Configuration File 'my_flirc_config.fcfg' to Disk
[=====>] 100%

Configuration File saved
```

There is also a **loadconfig** command.

What if a key does not work after configuring it.

First delete the key by its index. In this example the key d (Display Window) command isn't working.

```
$ flirc_util delete_index 18
```

Re-record the key

```
$ flirc_util record d
Press any button on the remote to link it with '+'
Successfully recorded button.
```

Re-test and repeat until a reliable ‘hash’ is received from the remote control.  
If a key is multiply defined delete the first one you see by its index.

```
13 B9F03963 escape  
14 D1F15097 pageup  
15 53DA6297 pagedown  
16 9F5BE297 escape
```

```
$ flirc_util delete_index 13
```

Re-test and if necessary, re-record.

There is a help facility for the flirc\_utility.

```
$ flirc_util help
```

## Configuring the display scroll speed

This option applies to LCDs only. It does not apply to PiFace CAD, touch screens or OLED devices.

The parameter **scroll\_speed** in **/etc/radiod.conf** can be changed to speed up or slow down the scroll speed of longer display lines. The **scroll\_speed** parameter is actually the inter-character delay in seconds. The smaller the value the faster the display scrolls.

The human eye can only discern 10 to 12 images a second limiting the lower range to approximately 0.08 seconds. The optimum speed is 0.2 to 0.3 seconds.

```
# Display Scroll speed 0.08 to 0.6 seconds  
scroll_speed = 0.2
```

It is only possible to set the scroll speed between 0.08 and 0.6 seconds.  
It may be necessary to adjust the contrast top get a good scrolling display.

## Configuring Russian/Cyrillic text

The radio program can display the Russian language either in **Cyrillic** or **Romanized** (convert to Latin) characters. For example, **Радио Пятница** when Romanized becomes **Radio Pyatnica**.

First purchase a character LCD/OLED with a Russian/Cyrillic character ROM. These devices also will display English characters.

To display Russian/Cyrillic text Romanized it is not necessary to change the configuration as this is the default. To display Russian/Cyrillic change the following parameters in **/etc/radiod.conf**.

Change the language to Russian

```
language=Russian
```

## Switch off Romanization

```
romanize=off
```

Unless using a HD44780U compatible controller leave the controller setting as it is

```
controller=HD44780U
```

If using an older LCD with an HD44780 (No U at the end) controller set it to HD44780

```
controller=HD44780
```

Leave the codepage setting as 0. This will pick up the correct code page from the language translation file in the **/usr/share/radio/codes** directory.

```
codepage=0
```



The **translate\_lcd** parameter must also be set to **on** for Romanization or Cyrillic translation routines to work. Translation is disabled if using an OLED as OLEDs use system fonts.

## Configuring European languages

First purchase a character LCD/OLED with a Western European character ROM. These devices also will display English characters. To display Western European text Romanized it is not necessary to change the configuration as this is the default. Any LCD/character OLED can be used for this.

Change the language to European and carry out the same instructions, except for language, as shown in *Configuring Russian/Cyrillic text* on page 142.

```
language=European
```

There is a detailed explanation of LCD code pages and program settings in Appendix **B.5 Cyrillic/European character LCDs/OLEDS** on page 308.

## Configuring the Internet Check URL and port

Version 7.0 onwards now regularly checks to see if there is an Internet connection when playing radio streams. It is disabled when playing a USB stick or share which do not require an Internet connection.

This feature it is checking to see if there is a connection to a reliable URL and port number across the Internet which can be contacted such as **google.com** on port number 80 (HTTP). There is a distinction between a network connection and an Internet connection. A network connection is local such as Wi-Fi or a wired network switch. An Internet connection is to a network provider via an Internet router which gives access to Internet resources, in this case, radio streams.

Without it, if the radio fails to load a radio stream because it has no network interruption it attempts to switch to another stream but this is pointless if there is no Internet connection.

This is configured in **/etc/radiod.conf**.

```
internet_check_url=google.com
internet_check_port=80
```

If Google ever change the **internet\_check\_url** a new one can be configured.

This feature can be disabled by removing the URL in the configuration

```
internet_check_url=
```

## Configuring MPD interface

The Music Daemon listens on port 6600, and this has not changed since the introduction of the Raspbian OS (Now RaspberryPi OS) in 2013. There is therefore no need change it unless the port setting in **/etc/mpd.conf** changes (port "6600").

```
mpdport=6600
```

## Configuring the remote-control parameters

The remote-control software has three parameters in **/etc/radiod.conf**:

The first parameter is for the remote control UDP server listen host, either 0.0.0.0 (All interfaces) or localhost.

The second parameter is the remote control communication port Default localhost 5100.

```
remote_control_host=localhost
remote_control_port=5100
remote_led=11
```

For information about the remote indicator LED see **Error! Reference source not found.** on page **Error! Bookmark not defined..**

## Configuring the radio software UDP server listening port

This allows another host to send UDP messages to the **radiod** UDP server (not the remote control **irreventd** daemon). It is either 'localhost' or the IP address of the remote **radiod** server.

```
remote_listen_host=localhost
```

You should normally never need to change this from the default **localhost**. This parameter is only intended for any new software used to control the radio from another host using the commands shown in Table 12 UDP messages on page 307. For example:

Remote UDP-Client → UDP network → Radio software UDP-receiver

(IP 192.168.1.66) Commands (192.168.1.203)

```
remote_listen_host=192.168.1.203
```

The remote client would send commands such as 'KEY\_VOLUMEUP', 'KEY\_CHANNELDOWN' etc. as shown in Table 12 to the **radiod** daemon running on host IP 192.168.1.203 (example).

## Configure the I2C bus number

```
i2c_bus=1
```

This only needs to be changed to '0' for the original Model A raspberry Pi's

## Configuring character LCD lines and width.

Display width, 0 use program default. Usual settings 16 or 20. Lines are normally 2 or 4.

```
display_width=20  
display_lines=4
```

These parameters are normally set by the **configure\_radio.sh** program. However, there may be cases where you might want to change these. They do not affect OLED displays.

## Configuring TFT display fonts

From version 7.8 onwards it is possible to configure both the font and its size on TFT displays such as displays using **LUMA**, **SH1106** and **ssd1306** drivers. Both the **radiod** and **weather2** programs can be configured. Both the **/etc/radiod.conf** and **/etc/weather.conf** files contain the following lines.

```
font_size=11  
font_name="/usr/share/fonts/truetype/dejavu/DejaVuSansMono.ttf"
```

Note that the **font\_name** parameter must contain the full path name to the font to be used. The default is **DejaVuSansMono.ttf**. Only TTF (True Type Fonts) have been tested. The default **font\_size** is 11.

TTF Fonts will be found primarily in the following directory **/usr/share/fonts/truetype/** in the following sub-directories: **dejavu**, **freefont**, **liberation2**, **piboto** etc.

## Configuring Rotary Encoders

Rotary encoders are normally configured by the **configure\_radio.sh** script however, you may need to change the configuration to suit your specific rotary encoders.

### Selecting the standard or alternative rotary class

The settings for the **rotary\_class** parameter are **standard**, **alternative**, **rgb\_rotary** or **rgb\_i2c\_rotary**. Some rotary switches do not work well with the standard rotary class so set to "alternative" to use the alternative rotary encoder class.

For rotary encoders with RGB LEDs in the shaft set to **rgb\_rotary**

For rotary I2C encoders with RGB LEDs in the shaft set to **rgb\_i2c\_rotary**

```
rotary_class=standard
```

### Configuring the pull-up resistors

KY-040 encoders etch have their own physical 10K pull-up resistors and do not need the internal GPIO pull-up resistors so the **rotary\_gpio\_pullup** parameter is set to **rotary\_gpio\_pullup=none** otherwise set to "up".

```
rotary_gpio_pullup=up
```

The **rgb\_rotary** or **rgb\_i2c\_rotary** settings use **I2C** and RGB Led settings and should be left to the **configure\_radio.sh** script to configure.

### Configuring step size

The rotary encoder detection step size 'full' step or 'half' step (default) configures the sensitivity of the rotary encoder class by setting the step size. This setting only affects the standard rotary encoder class (**rotary\_class.py**) and NOT the alternative rotary class (**rotary\_class\_alternative.py**)

```
rotary_step_size=half
```

The sensitivity of the rotary encoder can be reduced by setting it to a full step if the half setting gives problems.

**Note:** The **configure\_radio.py** configuration program sets the default step size for the type of encoder namely 'full' for ABC encoders and 'half' for KY040 encoders. It should not normally be necessary to change these settings. However, you may need to experiment with your specific rotary encoders to establish the best setting.

### Configuring the Menu rotary switch

These settings are not used in the standard RPi Internet Radio and can be ignored for this project. They were used in the Vintage Internet radio supplement to select the menu but are more or less redundant with only a few radios using it.

```
# Menu rotary switch (optional) Normal values are 24,8 and 7 respectively.  
Value 0 disables  
menu_switch_value_1=0  
menu_switch_value_2=0  
menu_switch_value_4=0
```

See <http://bobrathbone.com/raspberrypi/documents/Raspberry%20PI%20Vintage%20Radio.pdf> for more information.

## Configuring soft volume (SoftVol)

Some less expensive DACs do not have any controls such as volume. This can be seen by running the **alsamixer** program. It displays the following:

**This sound device does not have any controls**

Being able to control the volume via the mixer controls is needed for volume control of **Airplay** and **Spotify**. However, it is possible to enable a “soft” volume control (SoftVol). First copy the distribution **softvol** **asound** definition file to **/etc/asound.conf**.

```
$ cp /usr/share/radio/asound.conf.dist.softvol /etc/asound.conf
```

The **/etc/asound.conf** configuration file should look like the following:

```
# Software volume control for card 0
pcm.softvol {
    type            softvol
    slave.pcm      "plughw:0,0"
    control {
        name          "PCM"
        card          0
    }
}

pcm.!default {
    type            plug
    slave.pcm      "softvol"
}
```

Edit **/etc/mpd.conf** file to enable the softvol plugin in and DAC.

```
audio_output {
    type          "alsa"
    name          "My DAC"
    device        "plug:softvol"
    mixer_type   "software"
    mixer_control "DAC"
    # mixer_device "default"  # optional
    # mixer_index "0"        # optional
}
```

Note: “Name” can be anything and doesn’t affect the configuration.

The **alsamixer** program should now display the volume control and should allow the volume to be changed via the mixer control.

## Chapter 9 – Operation

<b>Contents chapter 9</b>	<b>Page</b>
Operation of LCD and OLED versions	149
Push buttons or Rotary encoders operations	150
Operation of HDMI and touch screen displays	153
The Vintage Graphic Radio	162
Playing Media	164
MPD Logging	165
Other useful logs	165
Installation and Configuration Logs	166
Configuration and status files	166
Using the Timer and Alarm functions	166
Music Player Clients	168
Creating and Maintaining Playlist files	169
Accessing Shoutcast	177
Overview of media stream URLs	181
Listening to live Air Traffic Control (ATC)	184
Mounting a network drive	187
Controlling the Music Player daemon from Mobile devices	191

## Operation of LCD and OLED versions

This section assumes that the LCD screen is working correctly, the MPD daemon is installed and tested and that there is an Internet connection available. This section is for LCD versions only. For graphical radios see *Operation of HDMI and touch screen displays* on page 153.

### Starting and stopping the LCD and OLED programs

The program must either be run as root user or using sudo.

The basic operation of the program is:

```
$ sudo service radiod start|stop|restart|status|info|version
```

Where start: Start the radio program.

stop: Stop the radio program.

restart: Restart the radio program.

status: Show the status of the radio daemon.

info: Show program information

version: Show the version number of the program

To start the radio:

```
$ sudo systemctl start radiod
```

To stop the radio:

```
$ sudo systemctl stop radiod
```

The following System V commands will also work:

```
$ sudo service radiod start  
$ sudo service radiod stop
```

To display the status either use the program directly or use the **sudo service radiod status** command:

```
$ sudo service radiod status
● radiod.service - Radio daemon
   Loaded: loaded (/lib/systemd/system/radiod.service; enabled; vendor
   preset: enabled)
     Active: active (running) since Wed 2017-11-08 10:06:19 CET; 3h 55min ago
       Main PID: 1619 (python)
          CGroup: /system.slice/radiod.service
                     └─1619 python /usr/share/radio/radiod.py nodaemon
:
{The last relevant log entries will be displayed here}
```

To see what version of the software you are running:

```
$ /usr/share/radio/radiod.py version  
Version 8.0
```

To see what build is running run

```
$ /usr/share/radio/radiod.py build  
Build 8.0.1
```

To display information about the running program:

```
$ systemctl status radiod  
● radiod.service - Radio daemon  
  Loaded: loaded (/lib/systemd/system/radiod.service; enabled; vendor  
  preset: enabled)  
  Active: active (running) since Wed 2020-09-30 12:03:00 BST; 47min ago  
    Main PID: 1560 (python)  
      Tasks: 3 (limit: 2068)  
     CGroup: /system.slice/radiod.service  
           └─1560 python /usr/share/radio/radiod.py nodaemon
```

The above shows the process ID of the radio, the Music Player Daemon version and operating system details.

### Push buttons or Rotary encoders operations

In the following sections there may be an instruction such as “Press left button”. If you are using rotary encoders then the following applies:

Rotary-encoder clockwise = button right  
Rotary-encoder anti-clockwise = button left

Rest of page deliberately left blank

## Radios with push buttons operation

The original radio has five buttons, four function buttons and one menu button. However, the new design can also support a sixth button which is the mute button. The Menu button changes the display mode and the functions of the left and right-hand buttons as shown in the following table. If using rotary encoders please see Table 5 on page 152.

**Table 4 Push Button Operation**

		Volume buttons		Channel buttons	
LCD Display Mode		Right button	Left button	Up button	Down button
<b>Mode = TIME</b> Line 1: Time Line 2: Station or Track		Volume Up	Volume Down	Station/Track up	Station/Track down
<b>Mode = SEARCH</b> If source = RADIO Line 1: Search: Line2: Radio Station		Volume Up	Volume Down	Scroll up radio station	Scroll down radio station
<b>Mode = SEARCH</b> If source = MEDIA Line 1: Search Line2: MusicTrack/Artist		Scroll up through artists	Scroll down through artists	Scroll up through track	Scroll down through track
<b>Mode = SOURCE</b> Line 1: Input Source: Line2: Radio or Media playlist or Airplay		Volume Up Mute	Volume Down Mute	Cycle up through Airplay, Radio and Media playlists	Cycle down through Airplay, Radio and Media playlists
<b>Mode = OPTIONS</b> Line 1: Menu Selection Line 2: <option> Options are Random, Consume, Repeat, Reload Music, Timer, Alarm ,Alarm Time Set (Hours), Alarm Set (Minutes), Streaming:		Toggle selected mode on or off. Set timer and Alarm	Toggle selected mode on or off. Set timer and Alarm	Cycle through Random, Consume, Repeat, Reload Music, Timer, Alarm , Alarm Time Set and Streaming:	Cycle through Random, Consume, Repeat, Reload Music, Timer, Alarm Time Set and Streaming:
<b>Mode = RSS (1)</b> Line 1: Time Line 2: RSS feed		Volume Up Mute	Volume Down Mute	Station/Track up	Station/Track down
<b>MODE = IP address</b> Line 1: IP address Line 2: Station or Track		Volume Up Mute	Volume Down Mute	Scroll up through track or radio station	Scroll down through track or radio station



Note: If the `/var/lib/radiod/rss` file is missing then the RSS mode is skipped.  
If it contains an invalid RSS URL, this will be displayed on the LCD.

## Radios with rotary encoders operation

This option is for a radio with rotary encoders with push buttons. The volume knob when pushed in is the **Mute** sound function. Likewise, the tuner knob when pushed in is the **Menu** switch. The Menu button changes the display mode as shown in the following table.

**Table 5 Rotary Encoder Knob Operation**

		Volume knob		Tuner knob
LCD Display Mode		Clockwise	Anti-clockwise	Clockwise
<b>Mode = TIME</b>				Anti-clockwise
Line 1: Time	Volume Up	Volume Down	Station/Track up	Station/Track down
Line 2: Station or Track				
<b>Mode = SEARCH</b>				
If source = RADIO	Volume Up	Volume Down	Scroll up radio station	Scroll down radio station
Line 1: Search:				
Line2: Radio Station				
<b>Mode = SEARCH</b>				
If source = MUSIC LIBRARY	Scroll up through artists	Scroll down through artists	Scroll up through track	Scroll down through track
Line 1: Search				
Line2: MusicTrack/Artist				
<b>Mode = SOURCE</b>				
Line 1: Input Source:	Volume Up	Volume Down	Cycle up through Airplay, Radio and Media playlists	Cycle down through Airplay, Radio and Media playlists
Line2: Radio, Media playlist, Spotify or Airplay	Mute	Mute		
<b>Mode = OPTIONS</b>				
Line 1: Menu Selection	Toggle selected mode on or off.	Toggle selected mode on or off. Set timer and Alarm	Cycle through Random, Consume, Repeat, Reload Music, Timer, Alarm Time Set, Streaming and Background colour(1)	Cycle through Random, Consume, Repeat, Reload Music, Timer, Alarm , Alarm Time Set, Streaming and Background colour(1)
Line 2: <option>				
Options are Random, Consume, Repeat, Reload Music, Timer, Alarm and Alarm				
Time(Hours), Alarm Time(Minutes) set and Change colour(1), Streaming on/off.				
<b>Mode = RSS (2)</b>				
Line 1: Time	Volume Up	Volume Down	Station/Track up	Station/Track down
Line 2: RSS feed				
<b>MODE = IP address</b>				
Line 1: IP address	Volume Up	Volume Down	Scroll up through track or radio station	Scroll down through track or radio station
Line 2: Station or Track				



Note 1: The colour change option is only available for the AdaFruit RGB plate (ada\_radio.py). Note 2: If the **/var/lib/radiod/rss** file is missing or contains an invalid RSS URL then the RSS mode is skipped.

## Mute function

Pressing both volume buttons together or in the case of a rotary encoder with a push button (Volume) will mute the radio. If voice is enabled then operation is slightly different (See section on espeak). Press either the volume up or down switch to un-mute the radio. If you change channel or use the menu switch the radio will also be un-muted. If the alarm is set then the radio will go into sleep mode.

## Operation of HDMI and touch screen displays

### The graphical screen

The HDMI and Touch Screen versions of the program can be started in three separate ways.

1. Automatically when starting the desktop
2. By clicking on the radio icon on the desktop
3. By manually starting the program from the command line (From a xterminal)

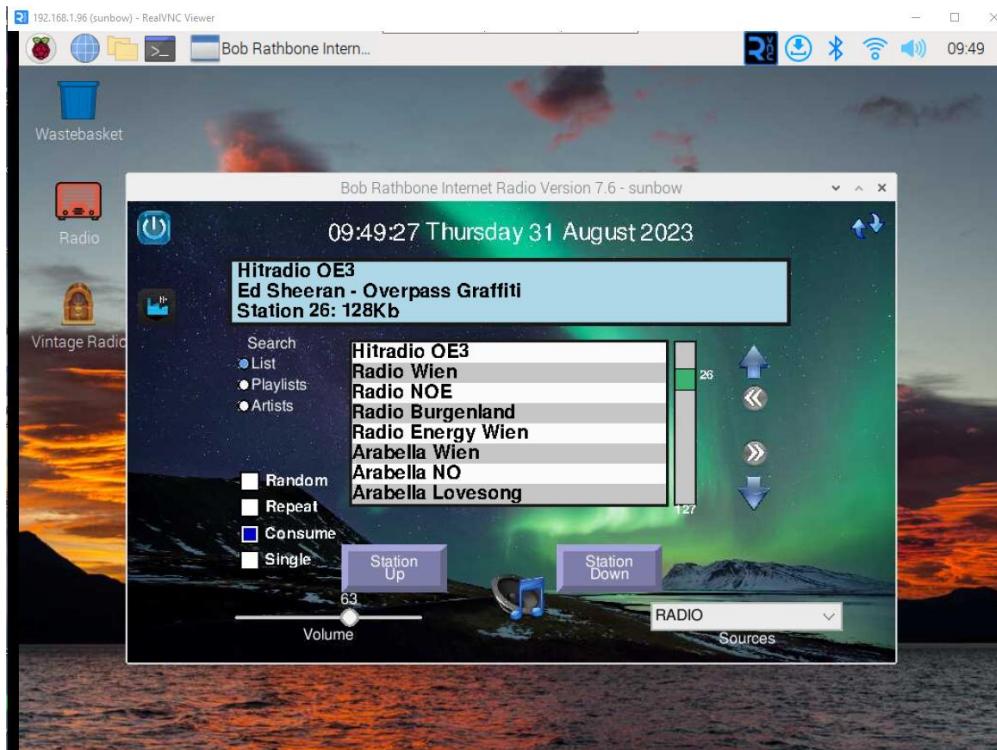
To start the radio from command line run the gradio.py program:

```
$ cd /usr/share/radio  
$ sudo ./gradio.py &
```

Starting the radio from the desktop. Click the icon shown here on the desktop.

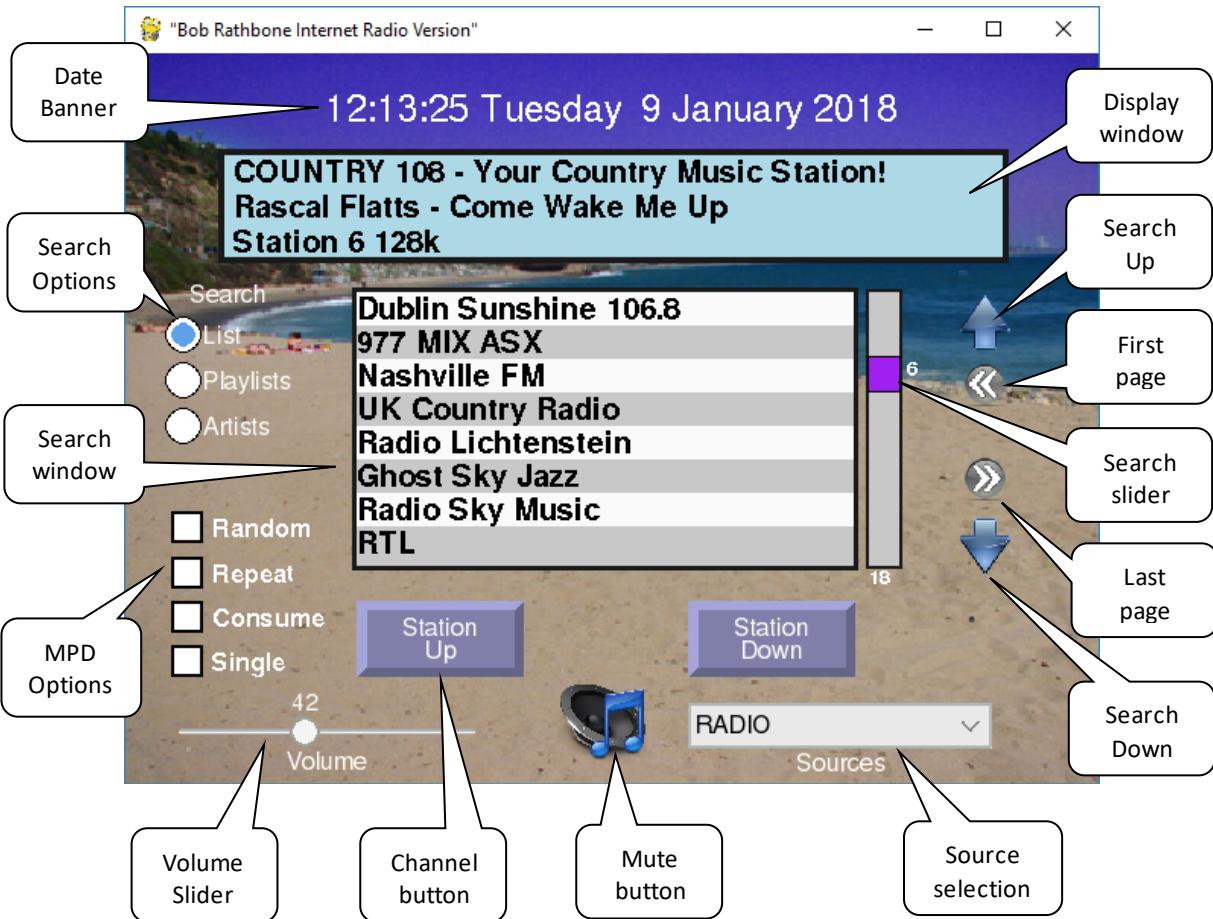


Normally you will need a HDMI monitor or TV with HDMI input and a mouse to operate the graphical radio programs as these programs run on the Raspberry Pi Desktop. However, it is possible to run the RPi desktop on a PC or MAC computer using VNC as shown below:



In all cases a screen similar to the following will be displayed. In this example **fullscreen=no**.

Figure 31 HDMI and Touch Screen Display



Clicking the mouse on a control such as station Up/Down or touching it do the same thing. In the following description we will only refer to “clicking”. By this, also touching a control is also meant.

### The display window

The display window normally displays the Radio station or Media rack that is currently playing. Clicking in the display window changes the third line to display the RSS feed if configured. A second click in the same window displays version details, IP address and hostname.

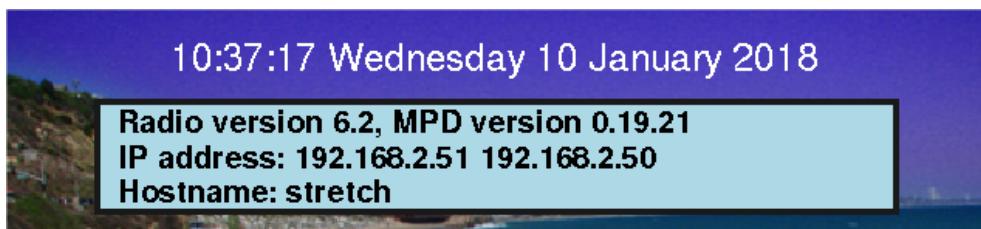


Figure 32 Graphical scree information display

In this example the hostname is ‘stretch’. The version number will be different for later releases. Two IP addresses are displayed (Wireless and Ethernet).

## The search window

The search window normally displays the contents of the currently selected playlist.



Figure 33 Graphical radio search window

Click on a station in the list selects it. Clicking in the slider window or dragging the slider re-positions the list. The current position, 16 in this example, is displayed next to the slider. The length of the current playlist, 28 for this playlist, is displayed at the bottom of the slider window.

Clicking on the Up and Down arrows travels up and down the list. Clicking on the left double arrow goes to the first page in the list. Clicking on the right double arrow goes to the last page in the list.



Figure 34 Graphical radio search functions

Clicking on the Playlists radio button selects the available playlists. This shows the playlists for radio or media such as the USB stick or Network share. It also shows 'airplay' which is not really a playlist but is a source, but can be selected here. Click on the desired playlist in the search window.



Figure 35 Display playlists

In the following example the USB stick playlist was selected. Once a playlist selected the list is displayed.



Figure 36 Display of media tracks

Clicking the Artists radio button displays the list of artists in the search window. Once clicked the search window positions on the first song of that artist's tracks.

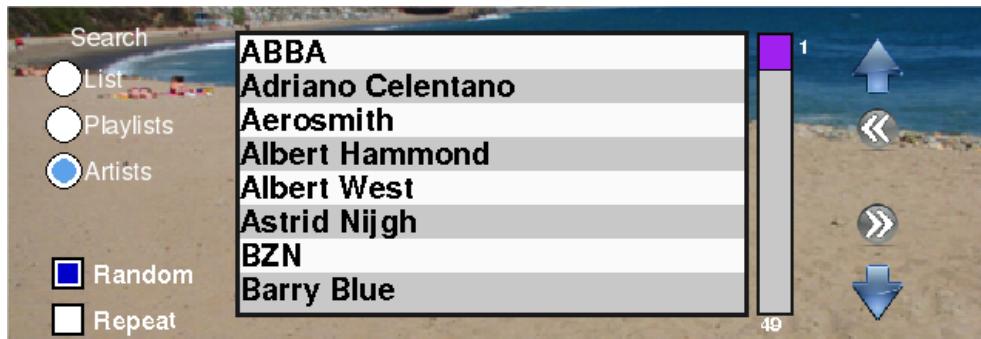


Figure 37 Displaying artists

Note that if you click on the 'Artists' radio button when displaying Radio stations, it will always be forced back to the 'List' display as Artist selection is not relevant for Radio stations.

#### Smaller TFT screens

Screens with a resolution equal to or less than 420 x 320 pixels will display slightly different than previously shown. Only one line will be displayed in the search window. There are no options for Random, Repeat or Consume due to lack of space.



The search list type (Playlist, Station/Track list or Artist) is cycled through by clicking on the Search list type button. All other controls work the same as shown in Figure 31 on page 154.

### Artwork display

If the music track has artwork and the **ffmpeg** (See *Setting up the locale* on page **Error! Bookmark not defined.**) package has been installed then the artwork will be displayed. Clicking on any of the radio search buttons will re-display the search window. The artwork cannot be displayed until the track is re-selected.

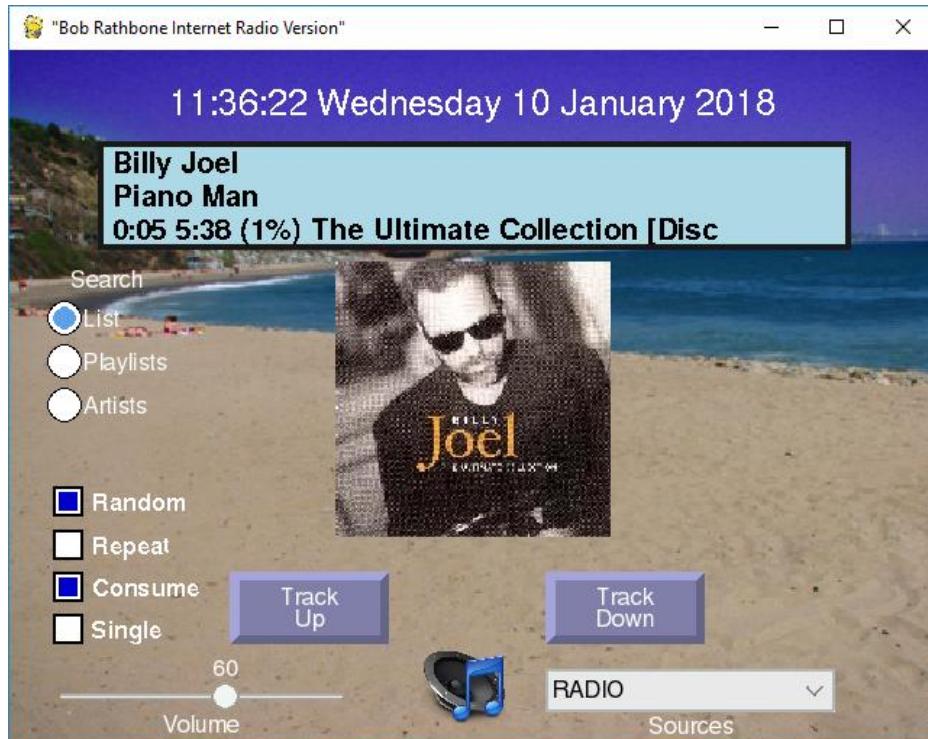
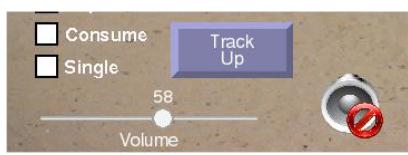


Figure 38 Track artwork display

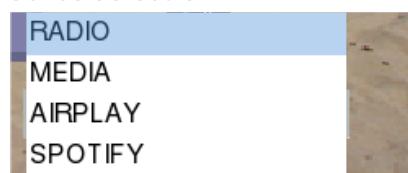
Note that the two grey push buttons now display 'Track Up/Down' instead of 'Station Up/Down'.

### Volume and Mute controls



The volume is controlled by a slider at the bottom left of the window. Clicking on the loud-speaker at the bottom of the screen mutes the sound and displays the mute icon as shown on the left. Any volume control change un-mutes the radio.

### Source selection



Click on the down arrow on the right of the Source selection to select the Source namely Radio, Media, Airplay or Spotify. The radio will select the first playlist in that source. Re-selecting the same source will select the next playlist for that Source.

### Other graphic window controls

Music Player Daemon(MPD) options Random, Repeat, Consume and Single are selected using the square push buttons on the bottom left of the window. Only the Random option is stored for the next time.



## *Running Airplay on the HDMI touchscreen*

Airplay must first of all be installed on the Raspberry Pi. See *Airplay Installation* on page 21 for instructions how to do this. To select Airplay either select it from the Sources drop-down box or from the playlists in the search window.

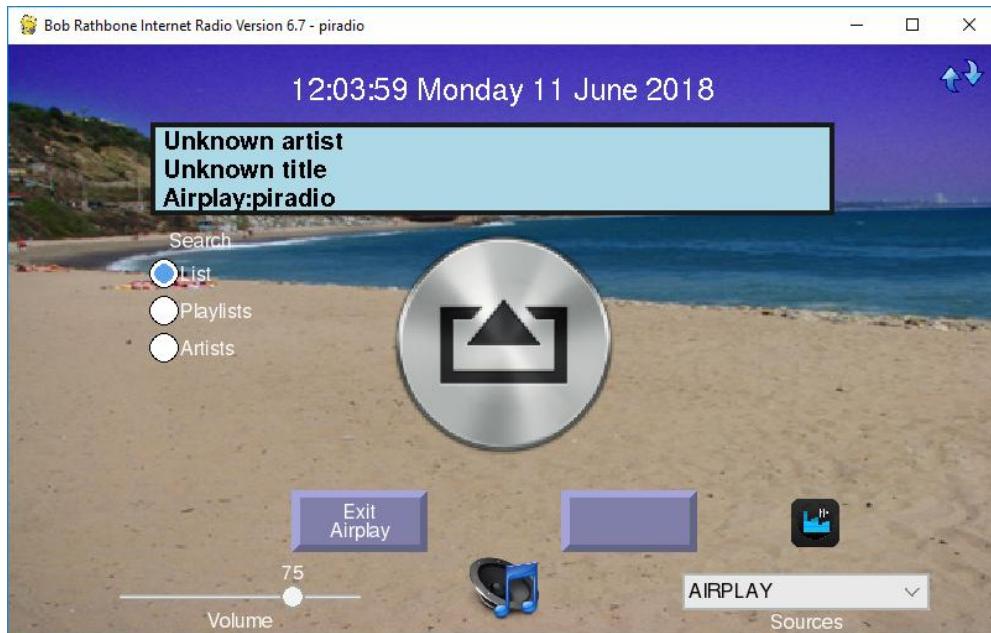


Figure 39 Airplay running on a Graphical screen

Connect to the Raspberry Pi from an Airplay compatible mobile device or run an App such as **CloudBreak**. The hostname to connect to is displayed when Airplay is first opened in the Display Window as shown below:

**Unknown artist**  
**Unknown title**  
**Airplay:piradio**

In this case the hostname is 'piradio'. To exit Airplay, press the left button at the bottom of the screen. The other button on the right has no label and doesn't do anything in Airplay mode.

### *Changing the graphical radio theme*

The colour scheme and background are largely configurable in the [SCREEN] section of the **/etc/radiod.conf** configuration file. Button colours cannot be configured.

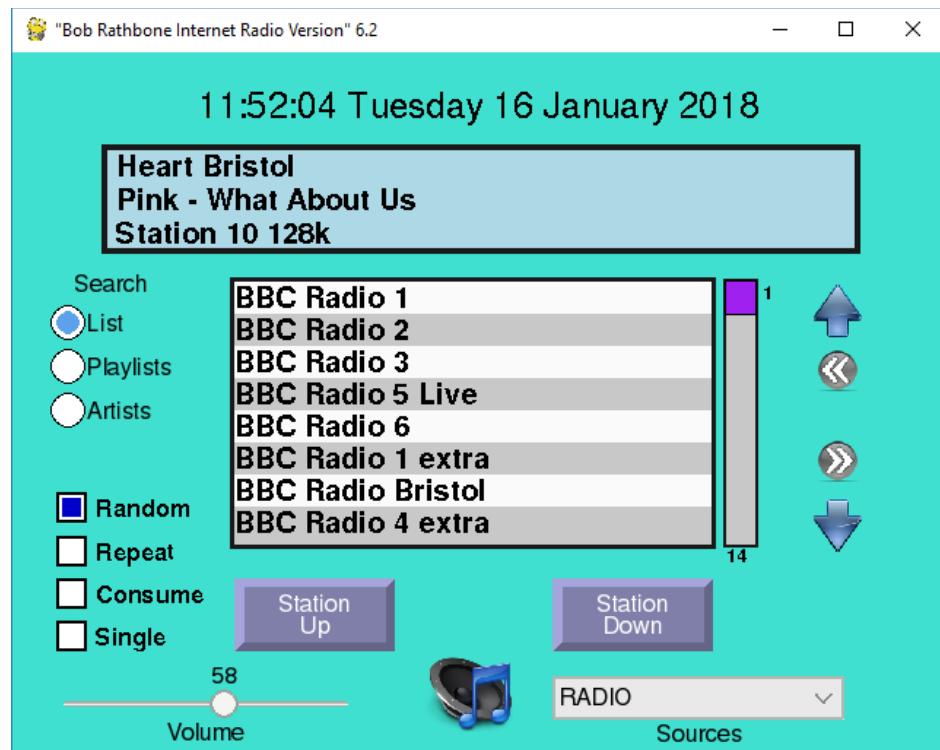


Figure 40 Changing the graphical screen theme

One good personalisation is to use your own favourite holiday picture as the background.

```
wallpaper=<path to your photograph>
```

Window and label colours can be changed to your own preferences. In the above screen the wallpaper option has been disabled, so the **window\_color** option is used.

```
# Graphics (touch screen) screen settings
[SCREEN]
fullscreen=yes
window_title="Bob Rathbone Internet Radio Version"
window_color=turquoise
banner_color=black
labels_color=black
display_window_color=lightblue
display_window_labels_color=black
slider_color=purple
display_mouse=yes
switch_programs=yes
screen_saver=0

# Wallpaper backgrounds. See /usr/share/scratch/Media/Backgrounds
#wallpaper=/usr/share/scratch/Media/Backgrounds/Nature/beach-malibu.jpg
# Set date format for graphic screen
dateformat=%H:%M:%S %A %e %B %Y

# The following is specific to the vintage graphical radio
scale_labels_color=white
stations_per_page=40
display_date=yes
display_title=yes
```

## Python pygame colour constants

See [https://www.pygame.org/docs/ref/color\\_list.html](https://www.pygame.org/docs/ref/color_list.html)

## Graphic screen keyboard controls

The HDMI/Touchscreen version accepts input from the keyboard. It is limited and is only included as a keyboard may be connected to the Raspberry Pi when using an HDMI screen. The normal interface is either touch screen or mouse and not the keyboard.

Table 6 Graphic screen keyboard command

Key	Description	Key	Description
<b>Page Up (PgUp)</b>	Channel/Track Up	<b>Up Arrow</b>	Search Up
<b>Page Down (PgDn)</b>	Channel/Track Down	<b>Down Arrow</b>	Search Down
<b>+ Key</b>	Volume increase	<b>Left arrow</b>	Go to first search page
<b>- Key</b>	Volume decrease	<b>Right arrow</b>	Go to last search page
<b>R</b>	Toggle Random	<b>L</b>	Select Search List
<b>T</b>	Toggle Repeat	<b>P</b>	Select Search Playlists
<b>C</b>	Toggle Consume	<b>A</b>	Select Search Artists
<b>S</b>	Toggle Single	<b>M</b>	Toggle Mute on/off
<b>D</b>	Cycle display window	<b>ESC</b>	Exit program
<b>X</b>	Switch between vgradio and gradio		

## Operating the Alsa Equalizer

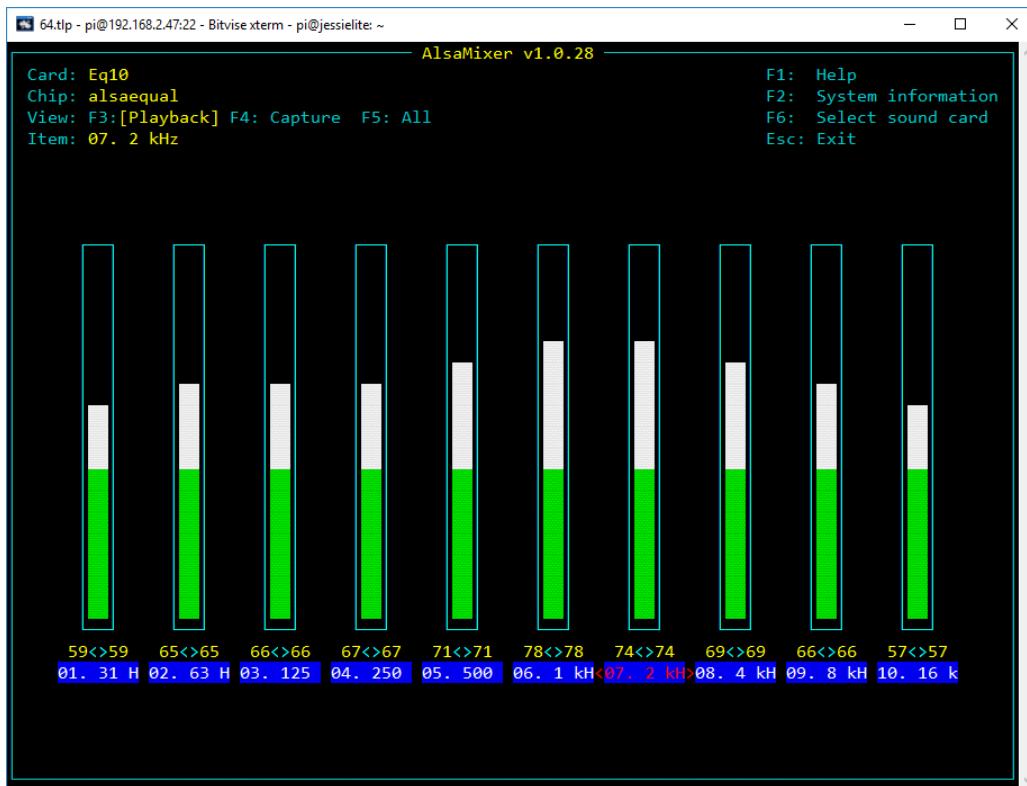


Figure 41 The Alsa

Use the Tab key to move along to the desired frequency to be changed. In this example, it is the <2KHz> block. Use the up and down arrows to adjust the level. The settings are saved in the `/var/lib/mpd/.alsaequal.bin` file. Changes to the sound should be heard.



**Note:** If you set a particular frequency value too high you will cause unpleasant distortion to the sound output.

### The Vintage Graphic Radio

As an alternative to the **gradio.py** program there is a touch-screen version of the radio called **vgradio.py**. This radio program only can play radio stations and not other Media (USB stick for example) or Airplay.



**Note:** This radio program can only play radio stations and not other Media such as a USB stick or Airplay, nor are there currently any plans to change this. If you want to play media you will need the full feature **gradio.py** program previously described.



Figure 42 The vintage graphic radio on a touch-screen

This allows a radio to be constructed to look like a vintage radio with a sliding tuning dial. The pages scroll through the stations so hundreds of stations can be added. When you touch the name of a station on the tuner dial the green slider jumps to that location and plays the selected station.

The double arrows at the top of the screen allow you to page through the stations. At the bottom is the round volume slider. Under that is the title of the currently playing song. The blue arrows are used to step through the stations one at a time. The mute button is on the right-hand side of the screen. This design can also be combined with rotary encoders or switches.

To run the graphics versions of the radio **gradio.py** or **vgradio.py** it is necessary to configure this using the Configure Radio utility **configure\_radio.sh** and select the Graphics display and user interface options. The Raspberry Pi OS supports either the legacy **X11** or the newer **Wayland**

windowing systems. **Wayland** supports either the **wayfire** or **labwc** windows compositors. For more information see the section called **X-Windows configuration for the radio** on page 119.

This radio is designed to work with a single radio playlist. This is normally the **\_Radio** playlist. You should configure the radio to start with this playlist by amending the **startup** parameter in **/etc/radiod.conf**.

```
startup=_Radio
```

However, this does not mean that you cannot have multiple radio playlists. If you have more than one radio playlist then by using the page up button (Double right arrow) it is possible to scroll through to the current playlist to the end and then onto the next playlist. In this case the new playlist name will be displayed in the very top-left of the screen.

You cannot currently scroll back to the previous playlist but must continue scrolling through the pages until you reached the desired playlist.

If using the FLIRC remote control dongle then it is only necessary to program the following keys: **pageup**, **pagedown**, **left**, **right**, **up**, **down**.

#### Switching between graphics programs

It is possible to switch between the full feature graphical radio (**gradio.py**) and the vintage graphical radio (**vgradio.py**). First configure the **switch\_programs** parameter in the [SCREEN] section of **/etc/radiod.conf**.

```
switch_programs=yes
```



Restart the program. The switch icon on the left will appear towards the top of the right-hand side of the screen. By clicking on it the program will switch between the two versions of the desktop radio programs. There will be a very short pause in the music stream whilst it is doing the switch-over.

#### Configuring a screen saver

 **Note:** The **xscreensaver** program described here does not appear to work if the radio program is in full screen mode.

Modern LCD displays are not as susceptible to screen burnout as the old cathode ray tubes of old. However continuous static screen displays will eventually cause shadowing. It is therefore a good idea to install a screen saver. The standard one for **Raspbian** is called **xscreensaver**. To install it run the following:

```
$ sudo apt install xscreensaver
```

After installation of the screen saver it can be configured in the desktop preferences menu. This allows configuration of time, screen saver or a blank screen. Choose a not too busy screen saver or the blank screen option.

There is also a program called **xscreensaver-command** for command line manipulation of the screen saver. However, the advice is not to use it as, at the time of writing, it causes severe problems with both the console and desktop display.

## Playing Media

### Playing MP3 and WMA files

The radio software also allows you to play music from the following sources:

1. From a USB stick
2. From a music directory on the SD card (Create this yourself)
3. From a Network Attached Storage (NAS)

Playlists for all of the above can be created using the **create\_playlist.sh** program.

```
$ cd /usr/share/radio  
$ sudo ./create_playlist.sh
```

See the section on *Creating Media playlists* on page 171 for a detailed description of this program.

### Playing music from a USB stick

Put your music tracks on a USB stick (MP3 and WMA files only) and insert it into the USB port of the Raspberry PI. Run the **create\_playlist.sh** program as shown above. Reboot the PI. Once the Radio program is running again, push the Menu button until "Input source" is displayed. Press either the left or right button to change the source to "Music Library". Now press the Menu button again. The music on the USB stick will now be loaded.

### Playing music from the SD card

With large (32/64GB) SD cards now available music can be stored on the SD card. There is already a directory called **/home/pi/Music** where music can normally be stored, however MPD has problems accessing this directory since the introduction of **Bookworm**. As a result from version 7.7 onwards **/usr/share/Music** is the new default location for music files.

Using FTP or any other file transfer program, copy the music from a PC to the **/usr/share/Music** directory and reload the library via the options menu. Now run the **create\_playlist.sh** program. Select option 3 (SD card). See the section on *Creating Media playlists* on page 171.

### Playing music from a Network Attached Storage (NAS)

This is a bit more involved to set up. See the section called *Mounting a network drive* on page 187. Then go to the section on *Creating Media playlists* on page 171

### Organising the music files

The search (find menu) routines get Artist and Track name directly from MPD which in turn get them from the music media file itself. The files should be placed in the top-level directory of the USB stick or in the **/usr/share/Music** directory if using the SD card. Any directory structure can be used. For example:

## Elvis Presley/The 50 Greatest Hits Disc 1/That's All Right.mp3

The find menu however will not use the directory structure for Artist/Track names so the directory structure and naming is arbitrary but should relate to the Artist and Track names displayed on the radio display. It is however possible sometimes to change the meta-data ((Artist/Track name) in the media file itself. Search on-line for ways of doing this.



Note: It is possible to create playlists for multiple locations. So, for example you can define one or more playlists for each of the USB Stick, NAS Storage and the SD card and select any of them from the radio or Web interface.

## MPD Logging

All logging for the MPD daemon is to the **/var/log/mpd/mpd.log** file by default.

## Radio program logging

The running Radio program logs to a file called **/var/log/radiod/radio.log**. See example log below:

```
$ tail -f /var/log/radiod/radio.log
2024-05-13 11:04:24,873 INFO ===== Starting radio =====
2024-05-13 11:04:24,874 INFO Initialising radio pid 3212
2024-05-13 11:04:24,875 INFO Login name pi
2024-05-13 11:04:24,877 INFO User:pi(1000) Group:pi(1000)
2024-05-13 11:04:25,075 INFO Python version 3
2024-05-13 11:04:25,078 INFO Translation code page in radiod.conf = 0
2024-05-13 11:04:25,513 INFO Display code page 0x1
2024-05-13 11:04:25,514 INFO Loaded 'codes.European'
2024-05-13 11:04:25,514 INFO Loaded 'codes.Russian'
2024-05-13 11:04:25,515 INFO Loaded 'codes.English'
2024-05-13 11:04:25,515 INFO Screen LCD Lines=4 Width=20
2024-05-13 11:04:27,286 INFO Romanize True
2024-05-13 11:04:27,287 INFO IP 192.168.1.251
2024-05-13 11:04:28,690 INFO Board revision 2
2024-05-13 11:04:28,704 INFO OS release: Raspbian GNU/Linux 12 (bookworm)
2024-05-13 11:04:28,708 INFO Linux bookworm32 6.1.0-rpi7-rpi-v8 #1 SMP
PREEMPT Debian 1:6.1.63-1+rpi1 (2023-11-24) aarch64 GNU/Linux
2024-05-13 11:04:29,499 INFO Connected to MPD port 6600
2024-05-13 11:04:29,597 INFO UDP Server listening on localhost port 5100
2024-05-13 11:04:29,598 INFO UDP listen:remote 0.0.0.0 port 5100
2024-05-13 11:04:30,352 INFO Radio ['/usr/share/radio/radiod.py',
'nodaemon'] Version 7.8
2024-05-13 11:04:30,352 INFO Radio running pid 3212
```

There are six levels of logging namely CRITICAL, ERROR, WARNING, INFO, DEBUG or NONE. This is configured in the **/etc/radiod.conf** file. Use DEBUG for more information.

```
# loglevel is CRITICAL,ERROR,WARNING,INFO,DEBUG or NONE
loglevel=INFO
```

## Other useful logs

**daemon.log** A very useful log for various looking at daemon process messages  
For instance, if you want to see what a particular daemon is doing use grep <daemon name>.   
**grep -i <daemon name> /var/log/daemon.log**

For example, to look at Bluetooth processes:

```
$ grep -i bluetoothd /var/log/daemon.log
Jul  8 08:59:55 Bullseye02 bluetoothd[808]: Bluetooth management interface
1.18 initialized
:
```

Daemons of interest are **radiod**, **mpd**, **librespot**(Spotify) and **bluetoothd**.

## Installation and Configuration Logs

Installation and configuration logs are stored in directory **/usr/share/radio/logs**. These are:

1. **install.log** – Output from the **configure\_radio.sh** script
2. **audio.log** - Output from the **configure\_audio.sh** script
3. **stations.log** – Output from **crontab** weekly run of **create\_stations.py**

These logs are overwritten every time the above programs are run.

## Configuration and status files

The main configuration file is **/etc/radiod.conf**. See section **A.1 Files added to the system** on page 294. This file is normally maintained by the **configure\_radio.sh** program. This is run at installation time but can be safely run at any time.

There are some other configuration and status files in the **/var/lib/radiod** directory. These are:

alarm	Alarm setting in t:hh:mm where t is the alarm type (t=0=off)
current_station	The current radio station
current_track	The current music track
language	Espeak language definition file
mixer_volume	Used by Airplay to set mixer volume (See page 22)
mixer_volume_id	Mixer volume ID (Used primarily for Airplay volume control)
rss	RSS feed URL
share	The NAS share instruction
stationlist	The user list of radio station URLs
streaming	Icecast2 streaming on or off
timer	Timer (Snooze) value in minutes
voice	The espeak voice file
volume	The volume setting
equalizer.cmd	The Alsa Equalizer command file called from <b>gradio</b> and <b>vgradio</b>

It isn't normally necessary to change most of these files. However, the **stationlist**, **share**, **language** and **rss** file will need to be edited as required. The other files are maintained by the radio or configuration programs. When the radio program starts up it uses the last settings, for example, the **volume**, **current\_station** and **current\_track** files.

## Using the Timer and Alarm functions



Note: The Raspbian operating system synchronizes time over the Internet. It does this using the **timesync** service. This service is a light-weight, client only, time synchronisation service, using the Network Time Protocol (NTP). In Bookworm this is implemented the **systemd-timesyncd** service.

There is a timer (Snooze) and alarm function (LCD and OLED versions only). The timer and alarm can operate individually or together. The timer when set will put the radio into Sleep Mode when the timer expires. The Alarm can be set to either On, Repeat or "Weekdays only".

### Setting the Timer (Snooze)

Press the Menu button until the “Menu Selection” is displayed. Press either the channel UP or DOWN control until “Timer off” is displayed on line 2 of the LCD screen. Now push the volume UP button to set the timer. Use volume UP and DOWN to adjust the timer which will be displayed as “Timer hh:mm:ss” where hh=hours, mm=minutes and ss=seconds. The Timer can be set up to 24 hours in increments of one minute. Once the timer is set, press the Menu button; the display will return to TIME mode.

On a four-line LCD display the timer will be seen counting down after the Volume display on line 4. On a two-line LCD display the timer count down will be displayed on line 1 after the time display.

When the timer expires (reaches zero) the radio will enter SLEEP mode. Sleep mode can only be exited by pressing the menu button.

To switch the timer off go back to the timer menu as described above and reduce the timer to 0 using the volume DOWN control. This will switch off the timer.

The timer function uses the `/var/lib/radiod/timer` file which will contain the value of the timer in minutes when it was successfully fired. You do not need to change the contents of this file.

### Setting the Alarm

The Alarm menu has three settings:

- The alarm type (On, off, repeat etc)
- The Alarm Hours time (Pressing menu in this mode puts the radio into Sleep mode)
- The Alarm Minutes time (Pressing menu in this mode puts the radio into Sleep mode)

Press the Menu button until the “Menu Selection” is displayed. Press either the channel UP or DOWN (Or rotate rotary encoder) until “Alarm off” is displayed on line 2 of the LCD screen. Using the volume UP control cycle through the options which are

- Alarm off - The Alarm is switched off
- Alarm on – The Alarm is on for one time only. Once the alarm is fired it will return to off.
- Alarm repeat – The Alarm will be repeated every day and not switched off.
- Alarm weekdays only – The Alarm will only fire Monday through Friday. It is not reset.

Now move to “Set alarm time:” using the channel UP control. The current alarm time will be displayed on line 2 of the display. Using the volume UP and DOWN control adjust the alarm time (Hours or Minutes) to the required setting. If you do not wish to put the radio into sleep mode at this stage then use the channel UP/DOWN control to move away from the “Set alarm time:” option and press the Menu button. If you press the Menu button whilst in the “Set alarm time:” option and the Alarm is set to anything except off then the radio will enter Sleep mode and display the alarm on line 2 for a two-line LCD or on line 4 for a four-line LCD.



Note: Sleep mode can only be exited by pressing the Menu button.

The alarm function uses the **/var/lib/radiod/alarm** file which will contain the current alarm type and time. The format is **t:hh:mm** where t is type (0=off, 1=on, 2=repeat, 3=weekdays only) and hh:mm is hours and minutes (24 hour clock). You do not need to change the contents of this file.



PLEASE NOTE THAT THE ALARM RELIES UPON THE SELECTED RADIO STREAM TO BE AVAILABLE WHEN THE ALARM WAKES UP. THIS CANNOT BE GUARANTEED AS THE STATION FEED MAY BE OFF AIR OR THERE IS A PROBLEM WITH THE INTERNET CONNECTION. YOU SHOULD NOT THEREFORE RELY SOLEY ON THIS ALARM FUNCTION IF YOU HAVE AN IMPORTANT APPOINTMENT OR A PLANE OR TRAIN TO CATCH FOR EXAMPLE. ALSO SEE DISCLAIMER ON PAGE **Error! Bookmark not defined..**

### Using the Alarm and Timer functions together

The Alarm and Timer functions can be used together. For example you want to set your radio to a 30 minute snooze time before going to sleep and to sound the alarm in the morning. Simple set the Timer to the required elapse time and then set the alarm as described in the previous section. Press the Menu button and the timer will be seen counting down followed by the alarm time on line 4 or line 1 for the four-line and two-line LCD respectively.

## Music Player Clients

MPD is designed around a client/server architecture, where the clients and server (MPD is the server) interact over a network. A large number of graphical and Web-based clients are available for MPD and are too numerous to mention here. Please see the following link for further information on MPD clients: <http://mpd.wikia.com/wiki/Clients>. The main client used in this project is MPC.

### Using the MPC client

Everything you should normally wish to do can be done using the radio. However, there may be occasions that you wish to test or control music selection, volume etc. using MPC. It is also useful for diagnosing Music Player Daemon problems.

Log into the Raspberry PI using the console or SSH login. To start playing music run:

```
$ mpc play
```

If the following is seen start **mpd** using the **sudo systemctl start mpd** command and retry:

```
mpd error: Connection refused
$ sudo systemctl start mpd
```

To see a list of all available commands, run:

```
$ mpc help
```

Here are some frequently used **mpc** commands:

**Table 7 Common MPC commands**

MPC command	Description
-------------	-------------

<b>mpc</b>	Displays status ( <b>mpc status</b> also does the same)
<b>mpc current</b>	Displays currently playing station or track
<b>mpc next</b>	Play next song
<b>mpc prev</b>	Play previous song
<b>mpc play n</b>	Play station or track where n is the track or station number
<b>mpc volume 75</b>	Set volume to 75%
<b>mpc stop</b>	Stop playing
<b>mpc random &lt;on off&gt;</b>	Toggle shuffling of songs on or off
<b>mpc repeat &lt;on off&gt;</b>	Toggle repeating of the playlist
<b>mpc clear</b>	Clear the playlist
<b>mpc consume &lt;on off&gt;</b>	When playing tracks remove from the playlist once played
<b>mpc playlist</b>	List loaded radio stations or streams
<b>mpc listall</b>	List all songs in the music directory

### Adafruit RGB Plate changing colours

This section is only relevant for the Adafruit RGB plate. When running the radio with an Adafruit RGB plate, it is an option to change the colour of the display. Push the menu button until “Menu selection”. Push the channel button until “Select color” is displayed. Now push the volume button to cycle through the colours. The available colours are red, green, blue, yellow, teal, violet, white or Off (No backlight). Note that the program uses the American spelling ‘color’

### Shutting down the radio

You can simply switch the power off. This doesn’t normally harm the PI at all. However, if you want a more orderly shutdown then press the menu button for at least three seconds. This will stop the MPD daemon and issue a shutdown request to the Raspberry PI. Wait at least another ten seconds and then power off the Radio.

### Creating and Maintaining Playlist files

When the radio software is first installed a Radio playlist is automatically created for you. You can add extra stations to the **/var/lib/radio/stationlist** file.

### Creating and updating radio playlists

There are four ways to create playlists:

1. Create Radio station playlists with the **create\_stations.py** program
2. Create Media playlists from either USB stick or Network share using **create\_playlist.sh**
3. Use the Shoutcast (**get\_shoutcast.py**) program or Web interface

## *The create stations program*

```
$ cd /usr/share/radio  
$ sudo ./create_stations.py
```

### Changing the character set

The default character set (charset) used by the Music Player Daemon is called UTF-8 and is defined by the following parameter in **/etc/mpd.conf**.

```
filesystem_charset      "UTF-8"
```

This can be changed to LATIN-1 for example for Western European languages.

```
filesystem_charset      "LATIN-1"
```

More information can be found in the man page for charsets.

<https://man7.org/linux/man-pages/man7/charsets.7.html>

However, this is a global change which affects all radio stations. To tell MPD to assume a different character set for one or more specific radio stations, specify it in the charset URL fragment parameter, e.g.:

```
http://radio.example.com/stream#charset=latin-2
```

To set this up define the radio station in the **/var/lib/radiod/stationlist** file with a **charset** definition for your region as shown in the following example:

```
[Espace 2] http://stream.srg-ssr.ch/m/espace-2/aacp_96#charset=latin-1
```

When the **create\_stations.py** program is run it will create the following entry in the **/var/lib/mpd/playlists/Radio.m3u** file:

```
#EXTM3U  
#EXTINF:-1,Espace 2  
http://stream.srg-ssr.ch/m/espace-2/aacp_96#charset=latin-1
```

As many different charsets can be defined as required by the individual radio stations. For example, a German radio station would use **latin-1** and a Polish station would use **latin-2**. If most of your radio stations are in a particular region then it is probably better to set this globally in the **/etc/mpd.conf** configuration file by changing the **filesystem\_charset** parameter. For example **LATIN-1**.

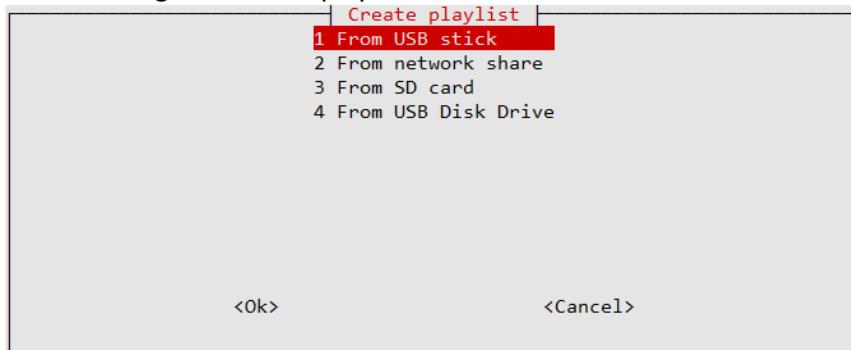
```
filesystem_charset      "LATIN-1"
```

## Creating Media playlists

The radio program comes with a program called **create\_playlist.sh**. This creates a single playlist for a USB stick, SD card location or a network share. It can produce as many playlists as required.

```
$ cd /usr/share/radio  
$ sudo ./create_playlist.sh
```

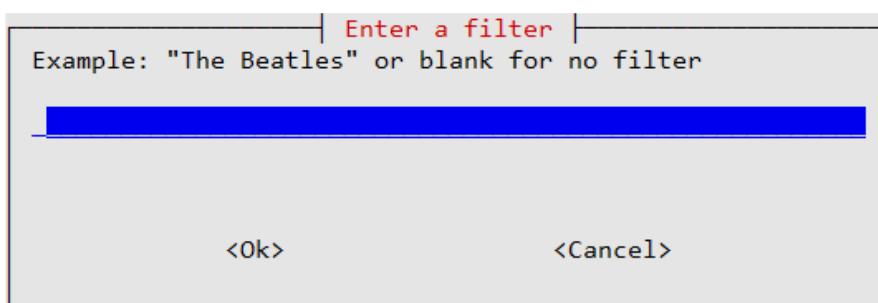
The following screen is displayed.



Select option 1 initially.



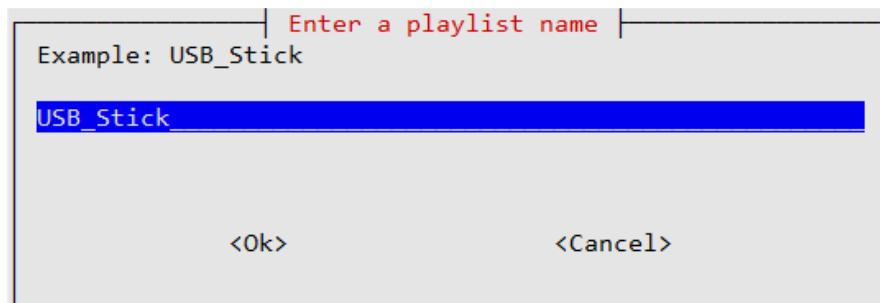
Answer "Yes" to create the playlist from the USB stick.



Enter a filter name or enter for no filter.



Select Yes to continue.

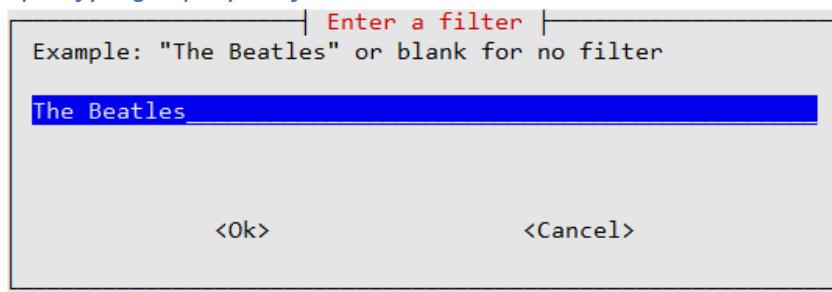


The program will suggest a name for the playlist but you may choose any name (But do not make it too long).

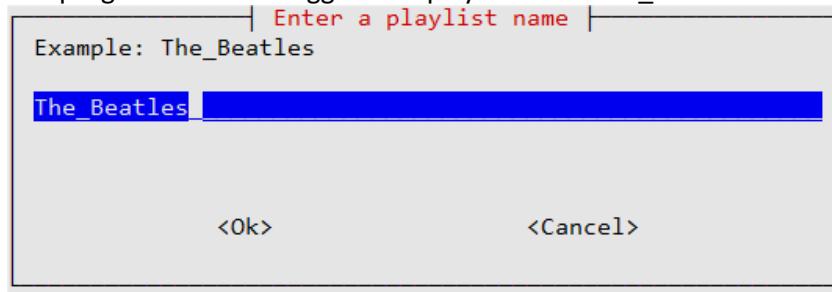
The program continues by creating a playlist called **USB\_stick.m3u** in **/var/lib/mpd/playlists**.

```
sudo service radio stop
sudo service mpd stop
Mounted /dev/sda1 on /media
cd /var/lib/mpd/music/
cd /var/lib/mpd/music
sudo find -L media -type f -name *.mp3 -or -name *.ogg -or -name *.flac >
/tmp/list29073
sudo mv /tmp/list29073 /tmp/USB_Stick
=====
58 tracks found in directory media (No filter)
mv /tmp/USB_Stick.m3u /var/lib/mpd/playlists/USB_Stick.m3u
sudo service mpd start
mpc stop
volume: 58% repeat: off random: off single: off consume: off
mpc update media
Updating DB (#1) ...
volume: 58% repeat: off random: off single: off consume: off
```

#### *Specifying a playlist filter*



The program will then suggest the playlist name **The\_Beatles**.

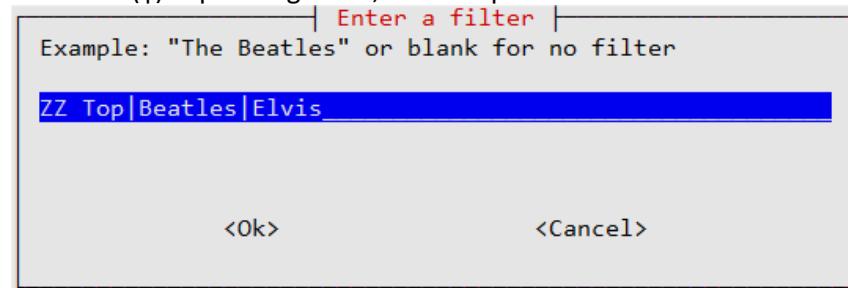


You will note that spaces in the playlist name have been replaced with underscores(\_). This is just for the file name. When the playlist is displayed in the radio program the underscores will be converted back to spaces. The program will now create a playlist with the name **The\_Beatles.m3u** (or whatever name was given).

```
sudo service radio stop
:
29 tracks found in directory share matching "The Beatles"
mv /tmp/The_Beatles.m3u /var/lib/mpd/playlists/The_Beatles.m3u
:
Updating DB (#1) ...
volume: 58%    repeat: off    random: off    single: off    consume: off
```

### Specifying multiple filters

More than one string may be specified in a filter. To do this specify the filter strings with a pipe character (|) separating them, for example:

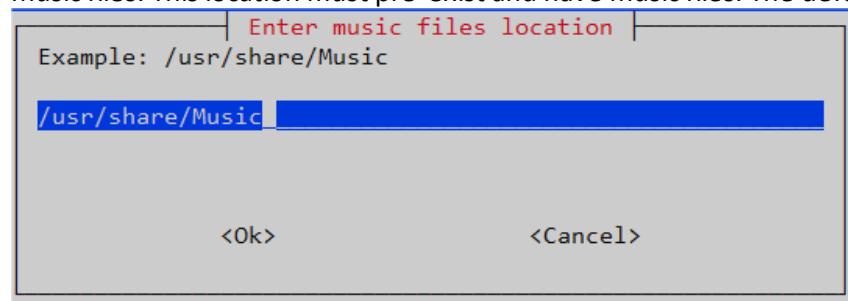


This will filter all songs from ZZ Top, The Beatles and Elvis or any other titles that contain these names. However this may not be what is wanted. Maybe songs by Elvis are wanted and not songs with 'Elvis' in the title. For example *Dire Straits – Calling Elvis*. In such a case use the / character to only look for directory names beginning with 'Elvis'. The above filter becomes:

**ZZ Top|Beatles|/Elvis**

Restart the radio to reload all new playlists. Using the / character gives a more accurate playlist. Please note that filters are not case sensitive. Filter 'Elvis' and 'elvis' will return the same result.

If you selected option 3 (SD card) you will be prompted for the location where you have installed your music files. This location must pre-exist and have music files. The default is **/usr/share/Music**.



### Maintaining playlists using external MPD clients

From version 7.3 onwards it is possible to add, delete and move playlist items using any external MPD clients capable of doing so. For example the **O!MPD** Web client supplied with the **radiod** Web package. See <https://www.musicpd.org/clients> for a list of MPD clients from the Music Player Daemon Web site.

Before it is possible to do this it is necessary to set the **update\_playlists** to **yes** in the **/etc/radiod.conf** configuration file and restart the radio.

```
# Allow updating of playlists by external clients yes/no (Experimental)
update_playlists=yes
```

The current default is set to **no** to maintain backward compatibility with older versions.

The following is an example of adding a new station via the **O!MPD** Web interface.

The recommended format for a new station URL is as follows:

<url>#<name>

For example

**http://relay.181.fm:8098#181.FM Salsa**

In **O!MPD** select the NOW PLAYING tab and at the top of the page click on add. Enter the URL in the format shown above.

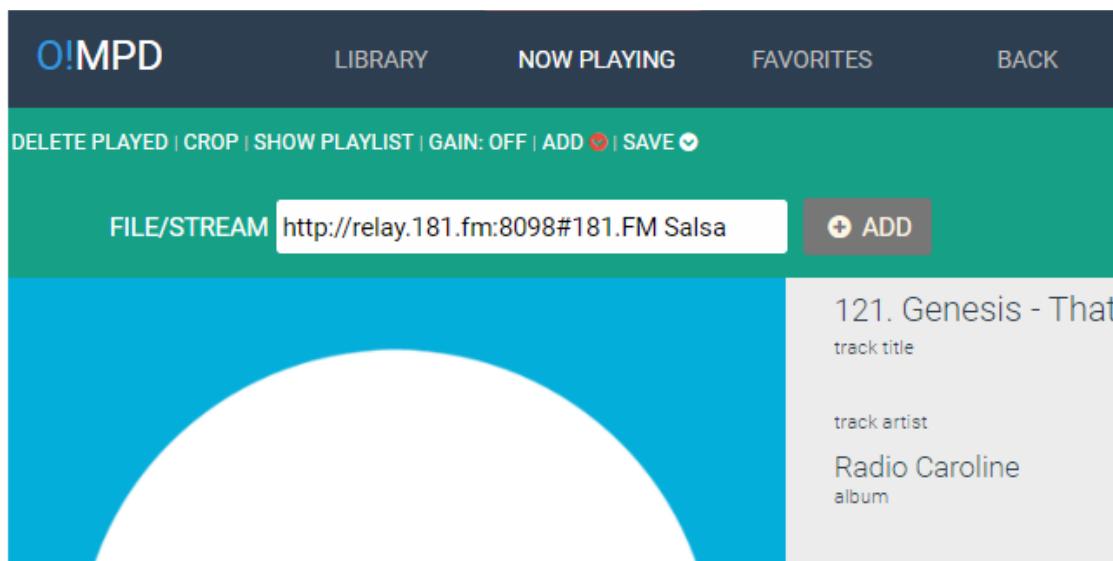


Figure 43 Adding a new station to MPD

Press the ADD button on the right of the URL entry box. The new station will be added to the end of the playlist.

It is possible to add the new station without the **#<name>** part.

For example

**http://relay.181.fm:8098**

In such a case the station will be added with the format New Station <n> where <n> is the position in the playlist. For example **New station 22**.



Figure 44 Adding new station URL in the Web interface

By clicking on the right-hand menu for the new station it is possible to move or delete a URL.

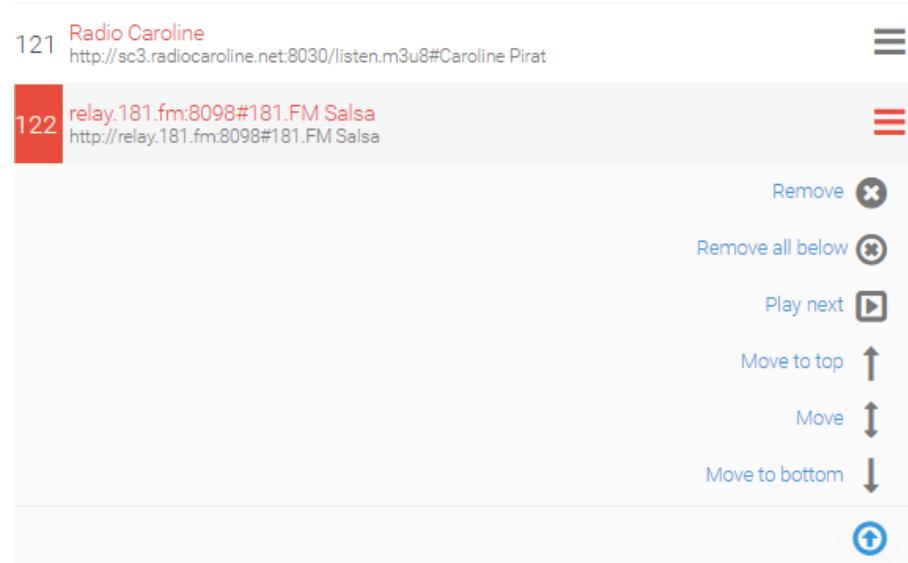


Figure 45 Moving and deleting entries in the Web Interface



**Note:** The author has been unable to make the double arrow **Move** icon work. Any feedback on this would be appreciated.

There is an important setting in **/etc/radiod.conf** which affects the way the station name is displayed namely **station\_names**.

```
# Station names source, list or stream
station_names=list
```

This is normally set to **list** which means that the names displayed by the radio come from the entries contained in the **/var/lib/radiod/stationlist** file as shown below.

```
[181.FM Salsa] http://relay.181.fm:8098
```

In this case station URLs in the playlist use the **<url>#<name>** fomat as previously explained. By setting **station\_names=stream** the station name will come from the radio stream its self.

In the above case new URLs can be added without the **#<name>** part as the name will come from the stream.



## Accessing Shoutcast

It is possible to create playlists from the Shoutcast database. See <http://www.shoutcast.com>. Shoutcast provide what can best be described as “fringe” radio stations. They do have a few stations by country but not many. If you are hoping, for example, to get all the United Kingdom BBC radio stations you will be disappointed. However their support for radio stations by genre is very good, for example: rock, jazz, country or classical. This version of software provides two methods of creating playlists from the Shoutcast database:

1. Using the `get_shoutcast.py` program
2. Using the shoutcast tab in the radio Web interface.

### Using the `get_shoutcast.py` program

Running the program with no parameters will produce the following usage message:

```
$ ./get_shoutcast.py
Shoutcast UDP connect host localhost port 5100
This program must be run with sudo or root permissions!

Usage: sudo ./get_shoutcast.py id=<id> limit=<limit>
search=<string> |genre=<genre> install
    Where: <id> is a valid shoutcast ID.
            <limit> is the maximum stations that will be returned
            (default 100).
            <string> is the string to search the shoutcast database.
            <genre> is the genre search string.
            install - Install playlist to MPD without prompting.

See http://www.shoutcast.com for available genres.
```

If you see the following message then install python-requests as shown below.

```
Traceback (most recent call last):
  File "./get_shoutcast.py", line 19, in <module>
    import requests
ImportError: No module named requests
```

```
$ sudo apt install python-requests
```

The program must be run with sudo. In the following example we want to get fifty jazz stations.

```
$ sudo ./get_shoutcast.py genre="jazz" limit=50
Extracting shoutcast stations: genresearch
Processing URL:
http://api.shoutcast.com/legacy/genresearch?k=anCLSEDQODrElkx1&limit=50&genre=jazz
Abc Lounge
Smoothjazz.com Global
:
:
Created 50 records in /usr/share/radio/playlists/_jazz.m3u
Do you wish to copy this playlist to /var/lib/mpd/playlists [y/n]: y
```

Answer 'y' to install the new playlist.

```
Copied /usr/share/radio/playlists/_jazz.m3u to /var/lib/mpd/playlists
Reload playlists: OK
```

This will copy the new **\_jazz.m3u** playlist to the **/var/lib/mpd/playlists** directory. The program will also signal the radio program to reload its playlists so that the new playlist can be accessed straight away.

If you answer 'n' to the above question your new playlist will be saved in **/usr/share/radio/playlists** repository. You can copy it later, if so wished, to the MPD playlists directory.

```
$ cd /usr/share/radio/playlists
$ sudo cp _jazz.m3u to /var/lib/mpd/playlists
```

You must use **sudo** to do this.

The program requires an authorisation key. This is embedded in the program and it is not necessary to specify it. If it ever changes or expires a new authorisation key must be configured in **/etc/radiod.conf**.

```
shoutcast_key=anCLSEDQODrElkxl
```

You need to get this key directly from <http://www.shoutcast.com>.



Note: Access to the Shoutcast is a free service made available through the goodwill of the folks at Shoutcast. It can be withdrawn at any time if over-used or abused so please do not set up any facility, such as scripting, which will stress their servers.

### Using the Shoutcast Web Interface

The radio Web interface now has a Shoutcast tab. Click on Shoutcast tab to open the interface. Fill in the search form and press the Submit button once and wait until the summary page is displayed.

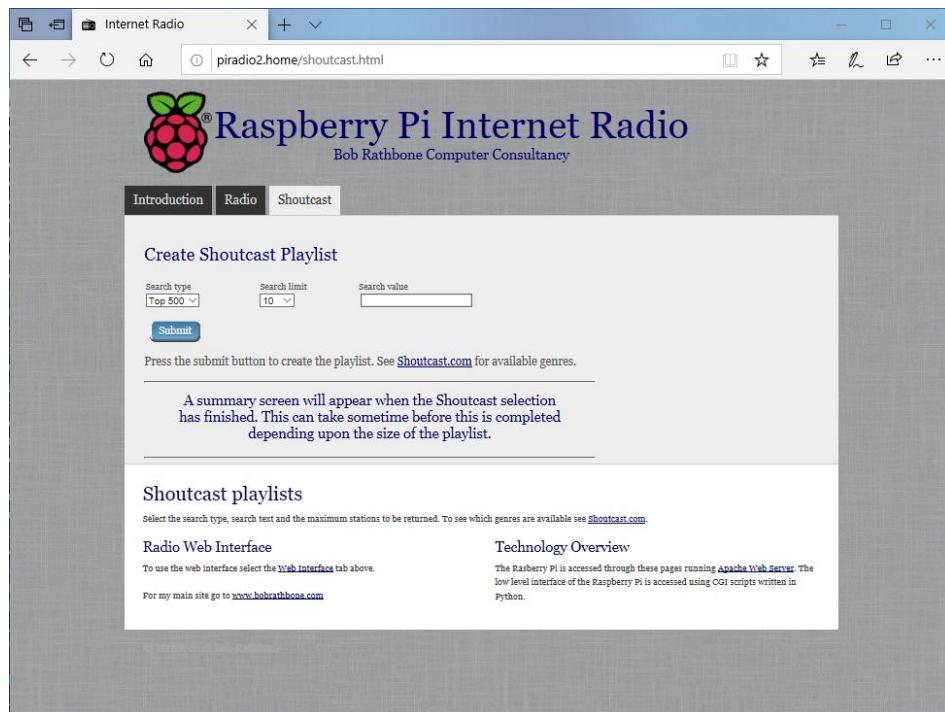


Figure 46 Shoutcast playlist Web page

Select the search type Top 500, Genre or Search from the “search type” drop down box. Select a “limit” for the search then click the “Submit” button. Normally the message “Please wait for selection” is displayed along with a rotating circle wait gif graphic. However, this does not work with **Microsoft Edge** and it is recommended you use the **Firefox** browser.



Figure 47 Shoutcast search selection

The summary page will be displayed. You should see the **Reload playlists: OK** message which means that the new playlist is available in the radio.

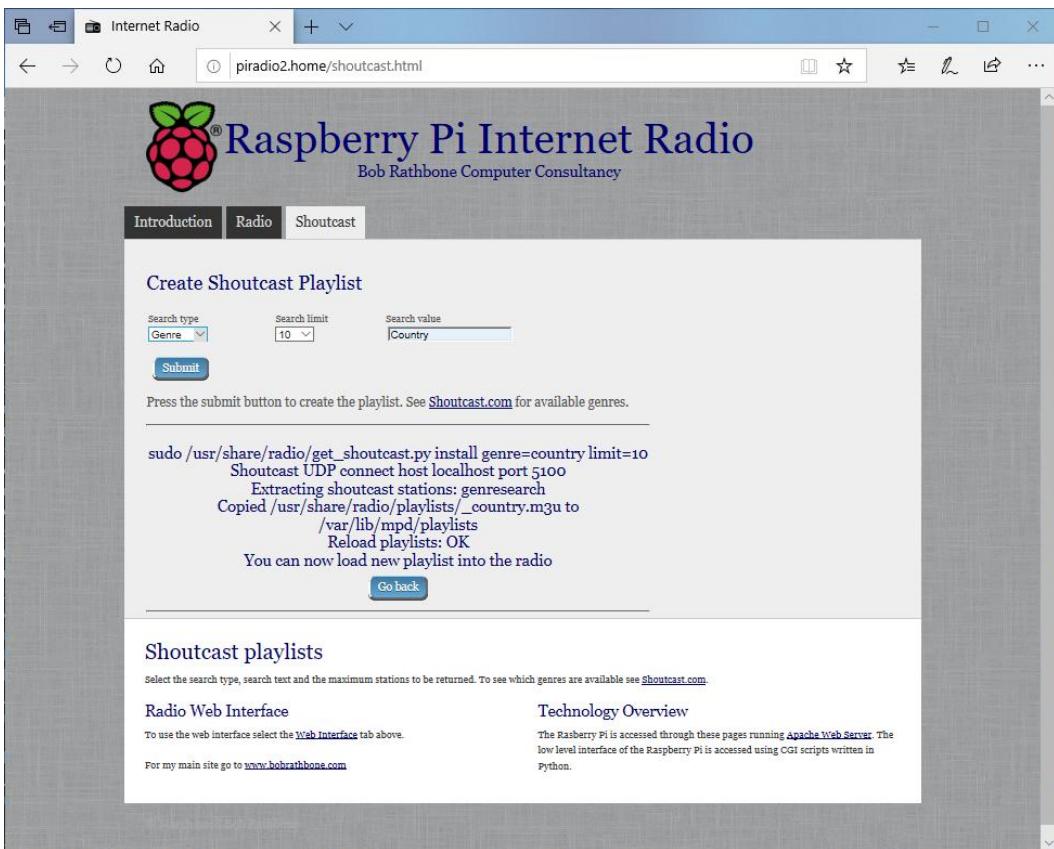


Figure 48 Shoutcast playlist summary

## Using old 5.x Radio playlists

Old 5.x playlists are not compatible with this version of the radio. However, the **/var/lib/radiod/stationlist** file can still be used. Do the following:

1. Stop the radio
2. Copy the old **stationlist** file to **/var/lib/radiod/stationlist**
3. Remove most of the (title) statements from the **/var/lib/radiod/stationlist** file
4. Remove all old playlists from **/var/lib/mpd/playlists** directory

```
$ sudo rm /var/lib/mpd/playlists/*
```

5. Run the **create\_stations.py** program as previously described.

If you find upon running the Radio that you have a lot of radio playlists. Reduce the number by removing title statements in the **stationlist** file as previously mentioned. These are the names in brackets – for example (BBC Radio). Re-run the **create\_stations.py** program.

## Radio stream resources on the Internet

There are a lot of resources on the Internet how to find PLS and M3U files so simply search for “PLS or M3U files” through the search machine of your choice. Below are some good sources of radio streams around the world.

<http://radiomap.eu/>

<http://www.publicradiofan.com>

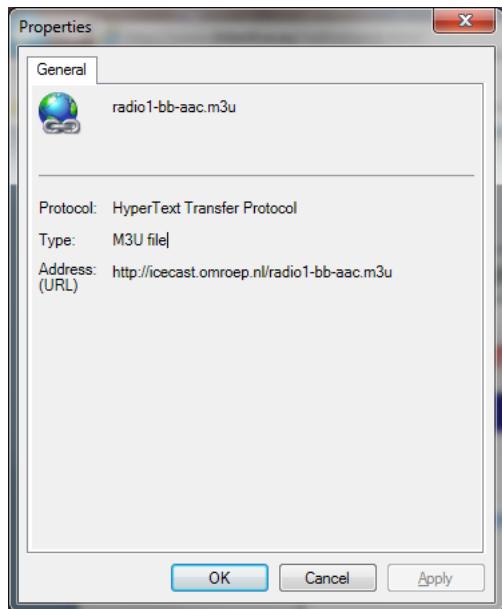
<http://www.radio-locator.com>

<https://www.internet-radio.com/>

<http://bbcstreams.com/>

<http://www.radiofeeds.co.uk/>

## Getting a radio stream from a Web browser



To copy a URL open the Web page in any browser on a PC and right click on the URL. Select properties from the drop-down list. For internet explorer will show a window similar to the illustration on the left will be displayed:

Copy and paste the URL into the **/var/lib/radiod/stationlist** file. Add the title in square brackets as shown in the previous section. Other browsers may provide options such as 'copy link' or 'save link as'. This is browser dependant.

## Overview of media stream URLs

A deep understanding of this section is not necessary but can be useful when creating playlists. This section is provided for background information only. At first the whole business of how music streams are provided can be quite confusing. The URLs on a radio station Web page can be of different types, for example:

1. A URL pointing to a M3U playlist file (MPEG3 URL). This format is used by MPD.
2. A URL pointing to an HLS (HTTP Live Streaming HLS - M3U8) playlist file
3. A URL pointing to a PLS playlist file (Shoutcast Play List)
4. A URL pointing to an ASX playlist file (Advanced Stream Redirector)
5. A URL which is an actual stream such as MP3 (MPEG 3) or AAC (Advanced Audio Coding)

1, 2 and 3 are so called redirector URLs and point to a playlist file containing one or more URLs to the radio stream(s) itself. The *create\_stations.py* program tries to figure out what type of URL that it is and create a playlist from it. This is the facility you should use rather than trying to create your own playlists which can be quite time consuming.

### M3U and M3U8 Files

M3U stands for MPEG3 URL. This is the format that MPD itself uses. The following Wikipedia article explains the M3U and M3U8 file formats:

<http://en.wikipedia.org/wiki/M3U>

The Music Player Daemon uses m3u files. An example [M3U](#) file is shown below:

### radio10.m3u playlist file

```
#EXTM3U
#EXTINF:-1, Radio 10 Gold NL
http://icecast.streaming.castor.nl:80/radio10
```

These playlist files have the **m3u** or **m3u8** (UTF-8 encoding) file extension. i.e. <filename>.m3u. The first line is the header and must be #EXTM3U. The second line is #EXTINF: and is information about the radio stream. The -1 means unlimited play length. This is followed by a comma and then the name of the radio station (*Radio 10 Gold* in this case). The third line is the URL (icecast in this case) for the radio stream. More than one radio stream may be defined in the m3u file. You do not need to create this type of file yourself. Modify the **stationlist** file and run **create\_stations.py**.

M3U files may also contain a simple list of file paths to media files. For example:

```
media/Steve Miller Band/Album onbekend/0726 Steve Miller Band - The
Joker.mp3
media/Stories/Album onbekend/Stories - Brother Louie.mp3
:
```

In this version of the radio the program knows that these are media files as opposed to radio playlists because they do not start with an underscore ‘\_’ which is the convention that the radio program uses for a radio playlist (It is not a general convention).

Note that in the above example **media** is a directory (or a link to it) in the **/var/lib/mpd/music** directory and that the ‘/’ character is omitted.

### [PLS file format](#)

A good place to start is the following Wikipedia article:

[http://en.wikipedia.org/wiki/PLS\\_\(file\\_format\)](http://en.wikipedia.org/wiki/PLS_(file_format))

A PLS playlist file does not contain any music files itself, but rather points to music files stored elsewhere. The PLS file format is often used to play Internet radio streams, for example, if you want to play a radio stream from Shoutcast, you can copy the PLS file URL of the station from the site and play it in a desktop media player like Winamp. A PLS file will be similar to below

```
[playlist]
NumberOfEntries=2
Version=2
File1=http://206.217.213.16:8430
Title1=Blues Radio UK
Length1=-1
File2=http://205.164.62.13:8030
Title2=Absolute Blues Hits
Length2=-1
```

The PLS file always starts with the **[playlist]** statement. The **NumberOfEntries** statement must match the number of streams defined in the PLS file (Two in the above example). Set the **Version** number always to 2. There is a **File<sub>n</sub>**, **Title<sub>n</sub>** and **Length<sub>n</sub>** where *n* is the entry number.

### [ASX file](#)

The Advanced Stream Redirector (ASX) format is a type of XML metafile designed to store a playlist of Windows Media files for a multimedia presentation. An example ASX file is shown below.

```

<ASX version="3.0">

<ABSTRACT>http://www.bbc.co.uk/iplayer/radio/bbc_radio_bristol/</ABSTRACT>
<TITLE>BBC Bristol</TITLE>
<AUTHOR>BBC</AUTHOR>
<COPYRIGHT>(c) British Broadcasting Corporation</COPYRIGHT>
<MOREINFO
HREF="http://www.bbc.co.uk/iplayer/radio/bbc_radio_bristol/" />
<PARAM NAME="HTMLView"
VALUE="http://www.bbc.co.uk/iplayer/radio/bbc_radio_bristol/" />
<Entry>
    <ref href="mms://wmlive-
nonacl.bbc.net.uk/wms/england/lrbristol?BBC-
UID=1523a2f20f858eb0ba0a4385918e6433df886bb650e021b4446ff486f8204e3a&SSO
2-UID=" />
</Entry>
</ASX>

```

## Direct stream URLs

These URLs tend to end with .mp3 or \_SC or AAC etc. However, there are others. For example:

<http://mp3.streampower.be/radio1-high.mp3>  
[http://7639.live.streamtheworld.com:80/977\\_MIX\\_SC](http://7639.live.streamtheworld.com:80/977_MIX_SC)

You can determine if a URL is a direct radio stream by using the **wget** program.

```

# cd /tmp
# wget http://mp3.streampower.be/radio1-high.mp3
--2014-03-14 13:08:10-- http://mp3.streampower.be/radio1-high.mp3
Resolving mp3.streampower.be (mp3.streampower.be)... 80.200.255.61
Connecting to mp3.streampower.be (mp3.streampower.be)|80.200.255.61|:80...
connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [audio/mpeg]
Saving to: `radio1-high.mp3'

[          <=>          ] 365,281      15.8K/s

```

If **wget** doesn't exit and you see the <=> characters moving backwards and forwards then it is a URL to the radio stream itself. You will also see *Length: unspecified* in the output. Press control -C to exit **wget**. Remove the file that **wget** created (/tmp/radio-high.mp3 in this case).

## Listening to live Air Traffic Control (ATC)

For those interested in aviation this is a fascinating use of the radio. Live ATC net provide streaming of live ATC transmissions from airports the world over. Their Web site is <http://www.liveatc.net/>

The screenshot shows a web browser displaying the LiveATC.net search results for the airport code EHAM. The left sidebar contains links for Site-wide search, Google Custom Search, Browse LiveATC Feeds, LiveATC Coverage Map, Top 50 LiveATC Feeds, Bad Weather Areas, LiveATC FAQ, Offer a LiveATC Feed, Contact LiveATC.net, Press Inquiries, LiveATC on iPhone, LiveATC on Android, Windows Phone, Windows 8/10, LiveATC Mobile (Mobile browser), ATC Audio Archives, Interesting Recordings, LiveATC Forums, Twitter | Facebook, and social media icons for LinkedIn and Facebook.

The main content area shows the following information:

- Europe**
- EHAM METAR Weather:** EHAM 130925Z 26014KT 230V290 9999 FEW021 BKN027 19/14 Q1021 NOSIG
- EHAM Flight Activity (FlightAware)**
- EHAM Webcam:** (Airport Webcams)
- EHAA Radar ARTIP**
- Feed Status: UP** **Listeners: 1**
- LISTEN** (in browser, requires Flash)
- LISTEN** (launches your MP3 player)
- LISTEN** (Windows Media Player)
- EHAA Radar ARTIP Audio Archives**

Facility	Frequency
EHAA Radar Sector 1 ARTIP	120.5500

- EHAA Radar East Inbound**
- Feed Status: UP** **Listeners: 1**
- LISTEN** (in browser, requires Flash)
- LISTEN** (launches your MP3 player)
- LISTEN** (Windows Media Player)
- EHAA Radar East Inbound Audio Archives**

Facility	Frequency

Figure 49 Live ATC Web page

Not only do these streams provide the live ATC conversations but also the in the station information ATIS (Aerodrome Terminal Information Service). This consists of coded weather information which all pilots can understand.

One way to add these stations to a radio playlist is to install **WinAmp** on a PC. Enter either the **ICAO** or **IATA** code of the station in the search box on the Live ATC Web site, for example **EHAM** or **AMS** (Schiphol, Amsterdam, the Netherlands) .

Click on the MP3 player **LISTEN** option. The station will be loaded and shortly **WinAmp** will start playing the stream.



Figure 50 WinAmp playing ATC live feed

Right click in the top left box (Display elapsed time of 1:05 in this example). The station information will be displayed.

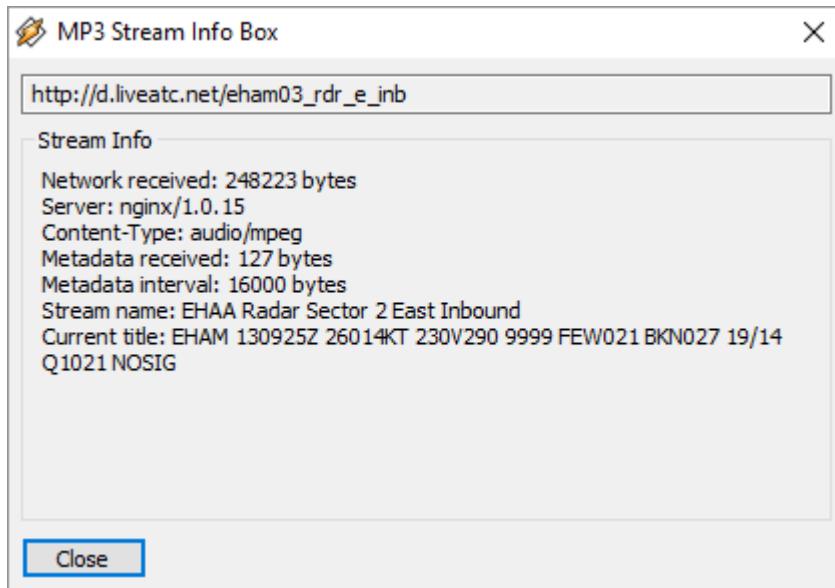


Figure 51 WinAmp station information

The URL for the stream is shown in the top box. [http://d.liveatc.net/eham03\\_rdr\\_e\\_inb](http://d.liveatc.net/eham03_rdr_e_inb)  
The stream name is: EHAA Radar Sector 2 East Inbound  
The station Title shows the ATIS information.  
EHAM 130955Z 25016KT 9999 FEW020 BKN024 19/14 Q1021 NOSIG

Using the URL shown create a playlist to **/var/lib/radiod/stationlist** for the live traffic ATC as shown in the following example.

```
(Air traffic)
[EHAA Approach Departures] http://d.liveatc.net/eham4
[EHAA Radar Sector 2 East Inbound] http://d.liveatc.net/eham03_rdr_e_inb
[EHAA Radar SW] http://d.liveatc.net/eham02_rdr_sw
[EHAA Radar 3 South] http://d.liveatc.net/eham02_rdr_s
[EHEH Approach] http://d.liveatc.net/eveh2_app
[CYYT] http://d.liveatc.net/cyyt
[KJFK Gnd Tower] http://d.liveatc.net/kjfk_gnd_twr
[CYYZ Tower] http://d.liveatc.net/cyyz7
```

Now run the **create\_stations.py** program to create the playlists.

```
$ cd /usr/share/radio
$ sudo ./create_stations.py
```

Now restart the radio or use the menu to reload the radio stations (Select source option):

```
$ sudo service radiod restart
```

Finally select the new ATC station(s).

#### Finding out ICAO and IATA airport codes

Try sites such as [https://en.wikipedia.org/wiki/List\\_of\\_airports\\_by\\_ICAO\\_code](https://en.wikipedia.org/wiki/List_of_airports_by_ICAO_code): A

#### Decoding ATIS information

See site <http://www.dixwx.com/wxdecoding.htm> or search for ATIS/TAF/METAR decode.

In the following example:

EHAM 130955Z 25016KT 9999 FEW020 BKN024 19/14 Q1021 NOSIG

EHAM	Amsterdam Schiphol, the Netherlands
130955Z	13 <sup>th</sup> of the current month. Time 09:55 Zulu (UTC)
25016KT	Wind 250 degrees at 16 Knots
9999	Visibility 10 Kilometres or greater
FEW020	Few clouds at 2000 feet
BKN024	Broken cloud at 2400 feet
19/14	Temperature 19 degrees Celsius. Dew point 14 degrees Celsius
Q1021	Barometric pressure 1021 Millibars (Will be given in Inches Mg in US airports)
NOSIG	No significant weather.

## Mounting a network drive

It is very likely that you may have your music on a shared network drive and want to play the music through the radio. There are two main types of network drive protocols used by Linux on the Raspberry Pi namely:

- CIFS – Common Internet File System
- NFS – Network File System

The protocol used for CIFS is SMB (Server Message Block – Microsoft). Previously connections to SMB were via a product called SAMBA but has been largely replaced by the mount using the CIFS option in the Linux. The steps to mount the network drive are as follows:

1. Find out the IP address or network name of your network drive.
2. Create and test the mount command using either NFS or CIFS.
3. Copy the mount command to **/var/lib/radiod/share** file.
4. In the Radio menu select “Music Library” as the source and press “Menu” again to load
5. Update the playlists to include the files on the new share (Network drive).

This procedure assumes that you already have your Network Drive configured and working with your PC and can play music via the PC. In the examples below a Synology Network Drive was used with a volume called Volume1 with a directory called “music”. The IP address for the Synology Network drive used was 192.168.2.6.

First stop the Radio software when creating and testing the mount command.

Don't configure **/etc/fstab** to do the mount of the network drive. Although this is the usual way of mounting shares however the radio program needs total control of the mount and un-mount process.

The general syntax for the mount command is as follows:

```
mount -t <type> -o option1,option2,... <remote IP address and directory>
<mount point>
```

Where: <type> is either **nfs** or **cifs**.

-o option1,option2 are the mount options.

<remote IP address and directory> Is the IP address and music directory path

<mount point> This will always be /share for this program

## Finding the IP address of the network drive

Only general guidance can be given here. Nearly all network drives have a Web interface. The IP address was almost certainly provided from DHCP in your home router. The IP address will be the IP address of the Web Interface. Look at your network drive documentation for further information.

## The CIFS mount command

The following example mount command assumes that you have a guest user configured with password ‘guest’. Adapt the command as required.

```
mount -t cifs -o username=guest,password=guest,uid=pi,gid=pi,vers=1.0  
//192.168.1.6/music /share
```

The above command is all on one line. The **uid** and **gid** parameters set the ownership of the music files to user **pi**. The **vers** statement is the CIFS version and can be 1.0, 2.0 or 3.0 depending upon the NAS storage. The share directory is created when you first run the Radio program so there is no need to create it. If the command was successful you should be able to display the music from the network drive. Go to section called *Display the share directory* on page 188.

### Older NAS drives sec security option

Older NAS drives may also require the **sec=ntlm** option to the **-o** line. The **sec** option is the authentication protocol and determines how passwords are encrypted between the server and client. Security mode **ntlm** used to be the default authentication method but that is now become **ntlmssp**. If you are accessing a network drive which doesn't support **ntlmssp** you have to add **sec=ntlm** to the options as shown below:

```
-o username=guest,password=guest,uid=pi,gid=pi,sec=ntlm
```

Many NAS devices use older technology so they often only use **ntlm** authentication. There are other authentication methods such as **ntlmv2** but most are not currently supported with the Raspberry Pi OS.

### The NFS mount command

The following NFS mount example assumes the NFS protocol has been configured for the music directory.

```
mount -t nfs -o ro,nolock 192.168.1.6:/volume1/music /share
```

A few things to note here; the NFS mount command uses the volume name (volume1 – can vary), The CIFS mount command doesn't. The second thing is that the IP address and remote directory are separated by a colon (:). If the command was successful, you should be able to display the music from the network drive.

### Display the share directory

If the mount was successful using either CIFS or NFS you should be able to display the **/share** directory with the **ls** command.

```
# ls -la /share  
total 4  
drwxrwxrwx 85 pi pi 0 May 10 14:18 .  
drwxr-xr-x 23 root root 4096 Jul 15 17:57 ..  
drwxrwxrwx 4 pi pi 0 May 10 14:16 Albert Hammond  
drwxrwxrwx 3 pi pi 0 May 10 14:16 Alexander Curly  
drwxrwxrwx 3 pi pi 0 May 10 14:16 Allen Price & Georgie Fame  
drwxrwxrwx 3 pi pi 0 May 10 14:16 Al Martino  
drwxrwxrwx 3 pi pi 0 May 10 14:16 Animals  
drwxrwxrwx 4 pi pi 0 May 10 14:16 Aretha Franklin  
drwxrwxrwx 3 pi pi 0 May 10 14:16 Armand
```

The important thing apart from seeing the files is that you should see that the files are owned by **pi** and group **pi** or whatever login name that you are using.

## Un-mounting the /share directory

To un-mount the share directory use the **umount** command (not **unmount**).

```
# umount /share
```

## Copy the mount command to the configuration

Once the mount command is working copy it to the **/var/lib/radiod/share** file.

For example for the CIFS mount command:

```
# echo "mount -t cifs -o username=guest,password=guest,uid=pi,gid=pi  
//192.168.1.6/music /share" > /var/lib/radiod/share
```

The above command is all on one line.



**Note:** If you decide to directly edit the **/var/lib/radiod/share** file instead of using the above command then do not include quotations marks around the command.

## Loading the media playlists

Now run the radio program. The radio stations will normally be loaded. Cycle through the menu by pressing the menu button until **Input Source:** is displayed. Press the channel up or down buttons or rotate the Channel knob to select one of the media playlists previously created using the **create\_playlist.sh** program as shown in the section *Creating Media playlists* on page 171.

Once the desired playlist has been selected, press the **Menu** button. The radio program loads the selected playlist and starts playing a track from it. The program does not know the last track number that was played previously for the selected playlist and will select the track number that it finds in the **/var/lib/radiod/current\_track** file. If this track number is bigger than the newly selected playlist length it will reset the track number to 1.

## Update the playlists for the new share

Select Music Library Now cycle through the menu until **Menu Selection:** is displayed. Press the channel up or down buttons until the **Update list:No** is displayed. Use the Volume buttons to toggle the display to **Update list:Yes**.

Now press the Menu button. This will cause the MPD database to be cleared and updated from all the files loaded in the **/var/lib/mpd/music** directory including the new share. This can take some time (Several minutes) if the Network Drive contains a large amount of music files. During this process the Radio program will ignore any button depressions and you will see the first **Initialising** (Library) and then **Updating** (Library).

## Create a Playlist for the share

Now create a playlist for the new network share. See *Creating and Maintaining Playlist files* on page 169. DO NOT FORGET TO DO THIS.

## Disabling the share

To disable the share simply put a hash character (#) at the beginning of the line in the **/var/lib/radiod/share** file as shown in the example below. Alternatively remove the share file altogether.

```
# mount -t cifs -o username=guest,password=guest,vers=1.0  
//192.168.1.6/music /share
```

## Further information

### Mount points

For your information if you display the **/var/lib/mpd/music** directory you will see two soft links to the **/share** and **/media** directories for the network drive and USB stick respectively. You may also see a link called **sdcard** to the location of the music library on the SD card.

```
$ ls -la /var/lib/mpd/music/total 8  
drwxr-xr-x 2 root root 4096 Jul 7 12:37 .  
drwxr-xr-x 4 mpd audio 4096 Apr 7 11:02 ..  
lwxrwxrwx 1 root root 6 Jul 7 12:17 media -> /media  
lwxrwxrwx 1 root root 16 Jul 7 12:37 sdcard -> /home/pi/mymusic  
lwxrwxrwx 1 root root 6 Jul 7 12:17 share -> /share
```

These links are created automatically by the Radio program. If these are missing they can be re-created with the **ln -s** command.

```
$ cd /var/lib/mpd/music  
$ sudo ln -s /media  
$ sudo ln -s /share  
$ sudo ln -s /home/pi/mymusic sdcard
```

This shouldn't normally be necessary as the links are created by the program when it creates the media and share mount points.

### Troubleshooting mount problems

See section called **Cannot mount remote network drive** on 216.

## Controlling the Music Player daemon from Mobile devices

### Android devices



Figure 52 M.A.L.P. play screen

There are a number of Android Apps capable controlling the Music Player Daemon from an Android device such as a smart-phone or tablet. One of the most popular seems to be **M.A.L.P** See the following link: <https://m-a-l-p-md-client.en.softonic.com/android> for further information.

Download from the **Android Play Store**. **M.A.L.P** allows you to control a MPD server (Music Player Daemon) and stream from it. The radio daemon is completely integrated with MPD clients such as **mpc** and **M.A.L.P**

### Settings

On the top left tap on the menu 

Go to **Settings → Profiles**.

In the Edit profile page press the **+** icon in the top right. Enter the name you wish for the profile. Enter the hostname or IP address of Raspberry Pi running the radio. Leave the port number at 6600. Leave the password blank (Don't enter your pi user password).

Switch on the remaining three options (*MPD covers*, *Enable streaming* and *Prefer HTTP cover files*).



Finally press the save icon  on the top right to save the profile.



Use the trash icon to delete a profile.



**Note:** M.A.L.P is third party software and no support can be provided by [bobrathbone.com](http://bobrathbone.com).

### Apple devices

There are at least three MPD client Apps for **iOS** mentioned on <https://www.musicpd.org/clients/>  
These are:

**MaximumMPD** - A MPD client for iOS

**Shinobu** - Modern and native client for iOS (iPhone / iPad)

**Stylophone** - A modern, native client for iOS/iPadOS; Also available on Windows!

Unfortunately, the author has no experience of either installing or using these Apps so cannot offer any support on them. You are referred to the above Web site and the Apple Store for further information.

## Chapter 8 -Troubleshooting

<b>Contents chapter 8</b>	<b>Page</b>
General	194
Installation problems	196
Network problems	197
Cannot find my Raspberry Pi	203
HDMI/Touchscreen problems	203
Trouble shooting problems with MPD	207
LCD Problems	210
Pimoroni Pirate Radio problems	212
Playlist problems	212
I2C and SMBUS problems	212
PiFace CAD and SPI problems	213
Olimex OLED problems	213
Pimoroni Pirate Audio crashes	213
Rotary encoder problems	214
Push button radio problems	215
Stream decode problems	216
Cannot mount remote network drive	216
Sound problems	217
Cannot change volume when running Airplay	228
Operational problems	229
Incorrect clock time displayed	231
IR remote control problems	231
Web interface problems	234
Bluetooth device connection problems	237
Using the diagnostic programs	238
Displaying information about the Raspberry Pi	248

## General

Many of the diagnostic messages shown rely on running the radio programs in diagnostic mode. Stop the **radiod** and **mpd** programs first.

### LCD and TFT versions of the programs

The following commands only work for radios using LCDs and TFT displays (**radiod.py**).

```
$ sudo systemctl stop radiod mpd
```

Now run the following commands for all radios using LCDs, TFTs or no display:

```
$ cd /usr/share/radio  
$ sudo ./radiod.py nodaemon
```

The above is particular useful for display diagnostics in the event of a crash.

### Graphic screen program versions

The graphic versions of the programs (**gradio.py** and **vgradiod.py**) must be stopped using the **kill** command. First find the process ID (pid) of the **gradio** or **vgradio** program.

```
$ ps -ef | grep radio  
root      5743     1  0 09:35 ?  00:00:00 sudo /usr/share/radio/gradio.py  
root      5744    5743 62 09:35 ?  00:00:04 python3 /usr/share/radio/gradio.py
```

Now kill the two processes shown:

```
$ sudo kill -9 5743 5744
```

**Important:** The following must be run from a terminal window run from the RPi Windows desktop.

```
$ cd /usr/share/radio  
$ sudo ./gradiod.py nodaemon
```

Or

```
sudo ./gradiod.py nodaemon
```

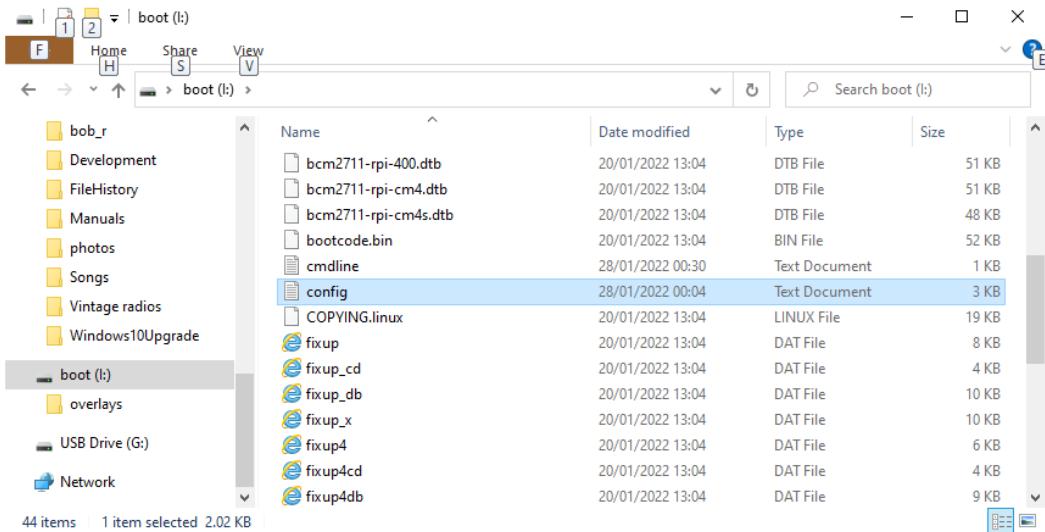
## Boot problems

### Newly created desktop version of Raspberry Pi OS will not boot

This used to happen with on older version of the **Bullseye** in 2022. After creating a new SD card using Raspberry Pi OS (32 bit) dated 1st January 2022 the Raspberry Pi will not boot.

This appears to have started in March 2022. It even seems to be trying to reboot several times. This has been experienced on the Raspberry Pi 4B. Other models may also be affected. This appears to be a problem with new DRM VC4 V3D screen driver (**vc4-kms-v3d**).

Insert the SD card into a PC. The PC will display a boot partition as shown in the example below:



**Figure 53 SD card boot partition**

In the above example this is on drive l: but will most likely be different on your PC. Find the config file (config.txt) on the boot drive and open it with **Notepad** or any other text editor. Find the entry **dtoverlay=vc4-kms-v3d**.

```
# Enable DRM VC4 V3D driver
#dtoverlay=vc4-kms-v3d
max_framebuffers=2
```

Either disable the driver by putting a # character at the beginning of the line or use the older **vc4-fkms-v3d dtoverlay** as shown below

Using the older **vc4-fkms-v3d dtoverlay**:

```
dtoverlay=vc4-fkms-v3d
```

Save the file and try booting off the modified SD card. All being well it should reboot OK. If not look for other causes as shown in the rest of this section.

### The Raspberry Pi will not boot from a previously good SD card

This is always worrying if this happens but doesn't always mean that the situation is irrecoverable. It often indicates a SD card corruption problem. Connect the HDMI output of the Raspberry Pi to the HDMI input of a TV set and attach a USB keyboard. Reboot the Pi. If you see the following:

```
[....] An automatic file system check (fsck) of the root filesystem failed.
A manual fsck must be performed, then the system restarted. The fsck should
be performed in maintenance mode with the root filesystem mounted in read-on
[FAIL] ... failed!
[warn] The root filesystem is currently mounted in read-only mode. A
maintenance shell will now be started. After performing system maintenance,
please CONTROL-D to terminate the maintenance shell and restart the system.
... (warning).
sulogin: root account is locked, starting shell
```

```
root@raspberrypi:~#
```

You are asked to press enter which brings up the dollar \$ prompt

```
Cannot access console, press enter to continue  
$
```

Enter a root password that you can remember.

```
$ sudo passwd root  
Enter new UNIX password:  
Retype new UNIX password:  
Password successfully changed  
$ sudo reboot
```

When reboot has finished you will be asked to enter the root password you just entered which will take you to the # prompt. Then run the following commands:

```
# umount /  
# fsck -y /dev/mmcblk0p2
```

This will, with luck, correct the file system. When **fsck** has finished reboot the system. Once rebooted use **vi** or **nano** to modify **/etc/default/rcS**. Add the following line to **/etc/default/rcS**.

```
# automatically repair filesystems with inconsistencies during boot  
FSCKFIX=yes
```

Finally reboot the Raspberry Pi.

```
$ sudo reboot
```

## Corrupt SD card

If experiencing boot problems, a corrupt SD card cannot be ruled out. See Advanced topics - *Repairing a non-bootable SD card* on page 99. If all of the above efforts fail to resolve the problem try installing the **Raspberry Pi OS** on a different SD card.

## Installation problems

### Missing package dependency problems

If an attempt is made to install the radio software without first installing **mpd** and **python3-mpd** the following maybe be seen:

```
dpkg: dependency problems prevent configuration of radiod:  
radiod depends on python3-mpd; however:  
  Package python3-mpd is not installed.  
radiod depends on mpc; however:  
  Package mpc is not installed.  
radiod depends on mpd; however:
```

```
Package mpc is not installed.
```

To correct both of these problems run:

```
$ sudo apt install python3-mpd
```

The following may also be seen in the case of **Bullseye** or earlier:

```
radiod depends on python-configparser; however:  
  Package python-configparser is not installed.
```

For **Bullseye** only run:

```
$ sudo apt install python-configparser
```

Then run the following to complete the installation:

```
$ sudo apt install --fix-broken
```

### Confused or unsure of wiring

All wiring is configurable in **/etc/radiod.conf**. The physical wiring must match the configuration in the configuration file. Run the **wiring.py** program (See page 242). Adapt either the configuration or wiring to match each other. The configuration in **/etc/radiod.conf** uses GPIO numbers and not physical pin numbers. If you are unsure how you have wired a button or rotary encoder this can be confirmed by running the **test\_gpios.py** program. See *The test\_gpios program* on page 243.

### Unexpected message during an upgrade

It is possible one of the files has been changed in a new package. For example:

```
Configuration file `/etc/logrotate.d/radiod'  
==> Deleted (by you or by a script) since installation.  
==> Package distributor has shipped an updated version.  
What would you like to do about it ? Your options are:  
  Y or I : install the package maintainer's version  
  N or O : keep your currently-installed version  
  D      : show the differences between the versions  
  Z      : start a shell to examine the situation  
The default action is to keep your current version.  
*** radiod (Y/I/N/O/D/Z) [default=N] ? X  
Installing new version of config file /etc/logrotate.d/radiod ...  
:
```

If you see this enter a Y to install the new file unless you have a good reason not to do so.

## Network problems

### Checking the network quality

Connect a screen or a TV and keyboard to the Raspberry Pi. Check the network with the **ip** tool. Run **ip addr** or press the menu button until the information screen is displayed with the IP address(es).

```
$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group
    default qlen 1000
        link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
            valid_lft forever preferred_lft forever
        inet6 ::1/128 scope host
            valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group
    default qlen 1000
        link/ether dc:a6:32:05:36:9b brd ff:ff:ff:ff:ff:ff
        inet 192.168.1.152/24 brd 192.168.1.255 scope global noprefixroute eth0
            valid_lft forever preferred_lft forever
        inet6 fe80::fe6a:9f30:4326:57d1/64 scope link
            valid_lft forever preferred_lft forever
3: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state
    UP group default qlen 1000
        link/ether dc:a6:32:05:36:9c brd ff:ff:ff:ff:ff:ff
        inet 192.168.1.153/24 brd 192.168.1.255 scope global noprefixroute wlan0
            valid_lft forever preferred_lft forever
        inet6 fe80::4c8b:8a00:e1d1:cd27/64 scope link
            valid_lft forever preferred_lft forever
```

The above example shows the Wi-Fi and Ethernet IP addresses.

Display the route to the Router using **ip route**.

```
$ ip route
default via 192.168.1.254 dev eth0 proto dhcp src 192.168.1.152 metric 202
default via 192.168.1.254 dev wlan0 proto dhcp src 192.168.1.153 metric 303
192.168.1.0/24 dev eth0 proto dhcp scope link src 192.168.1.152 metric 202
192.168.1.0/24 dev wlan0 proto dhcp scope link src 192.168.1.153 metric 303
```

The above example shows the route between Wi-Fi and Ethernet interfaces and the IP address of the Router (192.168.1.254).

If the problem is with Wi-Fi check the WiFi network configuration.

### For Bullseye or earlier

Edit the **/etc/wpa\_supplicant/wpa\_supplicant.conf** configuration file.

```
$ cat /etc/wpa_supplicant/wpa_supplicant.conf
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
country=GB

network={
    ssid."<Your-SSID>"
    psk."<Your-Router-Password>"
    key_mgmt=WPA-PSK
}
```

### For Bookworm or later

Bookworm doesn't use **wpa\_supplicant.conf** but now uses network manager using the following **nmcli** commands.

```
sudo nmcli radio wifi on  
sudo nmcli dev wifi connect <wifi-ssid> password "<network-password>"
```

### Example

```
sudo nmcli radio wifi on  
sudo nmcli dev wifi connect EE-J5TZ6R password "AUpU23vtEADt4NuX"  
Device 'wlan0' successfully activated with 'def548a7-2360-4662-a077-  
cdc50ab854f1'.
```

### Checking the Wi-Fi signal strength

If using the Wi-Fi interface, the signal strength received by the Raspberry Pi can be checked either one of the readily available mobile Wi-Fi signal apps or with the **display\_wifi.sh** program supplied with the radio software. Good Wi-Fi signal strength is required for the operation of the radio. Run the following commands:

```
$ cd /usr/share/radio/  
$ ./display_wifi.sh
```

This will display the main information about the Wi-Fi signal.

```
Wi-Fi network information  
-----  
Hostname: pizen  
Wi-FI: wlan0 ESSID:"EE-Hub-4qM4"  
IP address wlan0: 192.168.1.95  
wlan0 Protocol Name:"IEEE 802.11"  
Current Frequency:2.437 GHz (Channel 6)  
Bit Rate=72.2 Mb/s Tx-Power=31 dBm  
Link Quality=70/70 Signal level=-26 dBm  
Power save: on  
0: phy0: Wireless LAN  
    Soft blocked: no  
    Hard blocked: no  
1: hci0: Bluetooth  
    Soft blocked: no  
    Hard blocked: no
```

In the above example the name of the router (ESSID: TP-Link), the Wi-Fi interface (wlan0) IP address, protocol, frequency and channel are displayed. Lastly the really important information is the Tx-Power value in Decibels (dBm). If phy0 is blocked use "rfkill unblock 0" to enable it.

**Table 8 Wi-Fi signal strength**

Signal strength	Signal quality	Notes
-30 dBm	Maximum signal strength, usually attained if the RPi is very close to the Wi-Fi access point	Excellent for all services
-50 dBm	Anything down to this level can be regarded as excellent signal strength	Excellent for all services
-60 dBm	This is still a good, reliable signal strength	Good for all services

<b>-67 dBm</b>	This is the minimum value for all services that require smooth and reliable data traffic.	Minimum required for streaming services including for the radio
<b>-70 dBm</b>	The signal is not very strong, but mostly sufficient.	Only good for Web browsing, email, and the like
<b>-80 dBm</b>	Minimum value required to make a connection. You cannot count on a reliable connection or sufficient signal strength to use services at this level	The radio will not work reliably at this level
<b>-90 dBm</b>	It is very unlikely that you will be able to connect or make use of any services with this signal strength.	The radio cannot operate reliably at this level.

If the signal strength is poor, try re-positioning the radio nearer the router or use a repeater (TP-Link or similar). Position away from radiators or other large metal objects such as fridges and freezers.

### Checking response times with ping

Check the response times using the ping command from a PC or another Raspberry Pi. If possible, connect the PC/RPi via an Ethernet cable to the router rather than via Wi-Fi.

```
C:\Users\bob_r>ping -t 192.168.1.167

Pinging 192.168.1.167 with 32 bytes of data:
Reply from 192.168.1.167: bytes=32 time=5ms TTL=64
Reply from 192.168.1.167: bytes=32 time=7ms TTL=64
Reply from 192.168.1.167: bytes=32 time=5ms TTL=64
Reply from 192.168.1.167: bytes=32 time=9ms TTL=64
Reply from 192.168.1.167: bytes=32 time=8ms TTL=64
Reply from 192.168.1.167: bytes=32 time=140ms TTL=64
Reply from 192.168.1.167: bytes=32 time=7ms TTL=64
Reply from 192.168.1.167: bytes=32 time=17ms TTL=64
Reply from 192.168.1.167: bytes=32 time=4ms TTL=64
Reply from 192.168.1.167: bytes=32 time=161ms TTL=64
Reply from 192.168.1.167: bytes=32 time=4ms TTL=64
Reply from 192.168.1.167: bytes=32 time=6ms TTL=64
Reply from 192.168.1.167: bytes=32 time=75ms TTL=64
Reply from 192.168.1.167: bytes=32 time=5ms TTL=64
Reply from 192.168.1.167: bytes=32 time=5ms TTL=64
Reply from 192.168.1.167: bytes=32 time=7ms TTL=64

Ping statistics for 192.168.1.167:
    Packets: Sent = 16, Received = 16, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 4ms, Maximum = 161ms, Average = 29ms
Control-C
```

Figure 54 The ping command

Typical response times are:

Via Ethernet cable - < 1 ms

Via Wi-Fi - 5 to 12 ms with occasional responses < 300 ms

Below is an example of a **bad** network connection.

```
64 bytes from amradio (192.168.1.167): icmp_seq=2923 ttl=64 time=8.62 ms
64 bytes from amradio (192.168.1.167): icmp_seq=2925 ttl=64 time=1127 ms
64 bytes from amradio (192.168.1.167): icmp_seq=2926 ttl=64 time=6.00 ms
64 bytes from amradio (192.168.1.167): icmp_seq=2927 ttl=64 time=14.2 ms
64 bytes from amradio (192.168.1.167): icmp_seq=2927 ttl=64 time=26.6 ms (DUP!)
64 bytes from amradio (192.168.1.167): icmp_seq=2928 ttl=64 time=7.61 ms
64 bytes from amradio (192.168.1.167): icmp_seq=2929 ttl=64 time=6.12 ms
64 bytes from amradio (192.168.1.167): icmp_seq=2945 ttl=64 time=9.00 ms
64 bytes from amradio (192.168.1.167): icmp_seq=2947 ttl=64 time=138 ms
```

A DUP can be seen (packet resent as no response the first one received). Also, the 2<sup>nd</sup> packet sent took 1.127 seconds to respond.

The above network will give nothing but problems and needs correction. Wired Ethernet connections are very unlikely to give a problem. For a bad Wi-Fi connection consider using a Wi-Fi repeater, such as TP-Link or Netgear, for blind spots in your premises. Put the repeater close to the Raspberry Pi. Search “bad wi-fi” on the Internet for further advice on how to solve this type of problem.

#### [Reboot your router](#)

If you are getting poor ping responses on a Wi-Fi network it is worth re-setting your Wi-Fi router. Switch off your router for at least a minute and then power up the router again. This has been known to cure poor ping responses and problems with the radio.

If you are using Wi-Fi repeaters then also do a power reset on these devices as well.

#### [Checking the Internet connection](#)

The previous section shows if there is a network connection or not. But this only shows connectivity to the Local Area Network (LAN) which is just the Internet Router and local devices and does not mean there is connection across the Internet itself. This can be easily checked with the **host** command as shown below.

```
$ host google.com
google.com has address 216.58.206.142
google.com has IPv6 address 2a00:1450:4009:811::200e
google.com mail is handled by 20 alt1.aspmx.l.google.com.
google.com mail is handled by 10 aspmx.l.google.com.
:
```

The radio software contains a more sophisticated check which is configured in **/etc/radiod.config**.

```
# Internet check URL. This must be a reliable URL and port number
# which can be contacted such as google.com
# The port number is normally 80 (HTTP).
# The internet_timeout in in seconds
# Disable by removing the URL from internet_check_url= parameter
internet_check_url=google.com
internet_check_port=80
internet_timeout=10
```

The radio program regularly checks for a good Internet connection when playing a radio stream across the network. If you find this check is a problem it can be disabled by removing the URL (google.com)

```
internet_check_url=
```

### Wi-Fi is currently blocked by rfkill.

No Wi-Fi network is available and the following message is shown during login:

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

Running the **raspi-config** network set-up doesn't cure the problem. Run the **rfkill** utility to confirm the problem.

```
$ sudo rfkill list all  
0: phy0: Wireless LAN  
    Soft blocked: yes  
    Hard blocked: no  
1: hci0: Bluetooth  
    Soft blocked: no  
    Hard blocked: no
```

The above display confirms that the Wi-Fi network is blocked. Unblock it with **rfkill**:

```
$ sudo rfkill unblock all
```

### Wireless network keeps dropping out

The Raspberry Pi powers off the Wi-Fi adapter if it has been idle for some time. Normally the Wi-Fi adapter is in constant use when streaming a radio station but can become idle when muted.

You may see on-line documentation which asks you to add the following line to **/etc/network/interfaces**. However, this does not work for **Bullseye** or later.

```
wireless-power off
```

To disable Wi-Fi power management, add the following line to the end of **/etc/rc.local** just before the **exit 0** statement.

```
:  
sudo iw dev wlan0 set power_save off  
exit 0
```

After reboot the Wi-Fi power save state can be displayed with the following command.

```
$ iw dev wlan0 get power_save  
Power save: off
```

## Cannot find my Raspberry Pi

A very handy tool for checking devices connected to your network is the **Fing** App. This runs on both Android and Apple devices. See section **Error! Reference source not found.** on page **Error! Bookmark not defined..** More information on Fing can be found at <https://www.fing.com/products/fing-app>.

## HDMI/Touchscreen problems

### 7-Inch Touchscreen is displaying upside down

The screen can be flipped the other way by setting **led\_rotate=2** in **/boot/firmware/config.txt**

```
lcd_rotate=2
```

### Screen size error

The following may be seen in the logging when running the graphics version of the radio (gradio or vgradio).

```
gradio screen size error: No video mode large enough for 1024x600
```

The problem is that the screen size for the display being used has not been correctly set. There are three places this must be set. In the following example the graphics screen has a resolution of 720 x 480 pixels.

Edit the **hdmi\_cvt** parameter in **/boot/firmware/config.txt** (may vary).

```
hdmi_cvt=720 480 60 1 0 0 0
```

Make sure that the resolution specified matches your device. This is 720x480 in the above example.

Set the console to the same size.

```
framebuffer_width=720  
framebuffer_height=480
```

Finally edit **screen\_size** parameter in the [SCREEN] section of **/etc/radiod.conf**.

```
screen_size=720x480
```

### HDMI/Touchscreen radio does not start

Make sure that the LCD/TFT version of the radio isn't running. Stop and disable it with the following commands:

```
$ sudo systemctl stop radiod  
$ sudo systemctl disable radiod  
$ sudo reboot
```

If it is not starting on reboot then re-run the **configure\_radio.sh** program as shown in the section **Error! Reference source not found.** on page **Error! Bookmark not defined..** Select the option to start the program at boot time.

There is a file called **/home/pi/.config/lxsession/LXDE-pi/autostart**. This is where desktop applications are started from.

```
@lxpanel --profile LXDE-pi  
@pcmanfm --desktop --profile LXDE-pi  
@xscreensaver--no-splash@point-rpi  
@sudo /usr/share/radio/gradio.py
```

The last line starts gradiod.py at boot time. It has been known for the above file to disappear. Re-create it as shown above.

### Test the graphic version of the radio

Open a terminal window. The pi user prompt should be displayed. Now run the following

```
$ cd /usr/share/radio  
$ sudo ./gradio.py
```

The graphical version of the radio should start. If it doesn't, it should display the problem.

```
Traceback (most recent call last):  
  File "./gradio.py", line 46, in <module>  
    from gcontrols_class import *  
  File "/usr/share/radio/gcontrols_class.py", line 22, in <module>  
    from sgc.widgets.base_widget import Simple  
  File "/usr/share/radio/sgc/__init__.py", line 19, in <module>  
    import surface  

```

In the above example the **python-pygame** package is missing. The solution is to install it.

```
$ sudo apt install python-pygame
```

Re-test gradio.py

### HDMI/Touchscreen is displaying upside-down

The standard orientation is with the connectors for power and HDMI at the bottom of the screen. However, with the official Raspberry case, it has the cables at the top. If the screen is displaying upside-down then edit the **/boot/firmware/config.txt** configuration file and add the following line.

```
lcd_rotate=2
```

See the following guide:

<https://www.modmypi.com/blog/raspberry-pi-7-touch-screen-assembly-guide>

### The touch screen displays a lightning symbol

This is an under-voltage warning. See:

<https://www.raspberrypi.org/documentation/configuration/warning-icons.md>

Use at least a 1.5 ampere (1500 mA) power supply.

### The touch screen displays a thermometer symbol

The Raspberry Pi is getting too hot. Improve airflow to the Raspberry Pi. Do not ignore this warning.

### Sound is heard but the graphical radio program will not start

This is almost certainly due to the fact that the **radiod** service is running. Either re-run the **configure\_radio.sh** program to reconfigure or run the following commands.

```
$ sudo systemctl stop radiod  
$ sudo systemctl disable radiod
```

### The HDMI/Touchscreen program only displays a blue screen

This is due to missing scratch files in **/usr/share/scratch/Media** directory. To correct this install scratch as shown in the section called **Error! Reference source not found.** on page **Error! Bookmark not defined.** or change the .

### Cannot launch Graphical radio by clicking on the Desktop Icon

This is almost certainly because the **gradio.desktop** **vgradio.desktop** files in the home Desktop folder are not executable. To correct this do the following

```
$ cd /home/pi/Desktop  
$ sudo chmod +x gradio.desktop vgradio.desktop
```

It should now be possible to launch the radio from either desktop icon by double-clicking it.

### Clicking on one of the radio desktop icon asks for an execution option

When attempting to launch the graphical radio programs the following is seen.

```
This text file seems to be an executable script "what do you want to do with it"
```

This is a minor irritation. To run the program just click on Execute.

To disable this behaviour, do the following:

1. On the Windows desktop click on File manager (Folder Icon top left)
2. Move to the home **Desktop** folder by double clicking it
3. Highlight **gradio.desktop** file
4. Select Edit on the top left of the screen then *Preferences → General*
5. Un-tick the option "Don't ask option on launch executable file"
6. Close the *Preferences* window
7. Highlight **vgradio.desktop** file and repeat the above steps 4 through 7

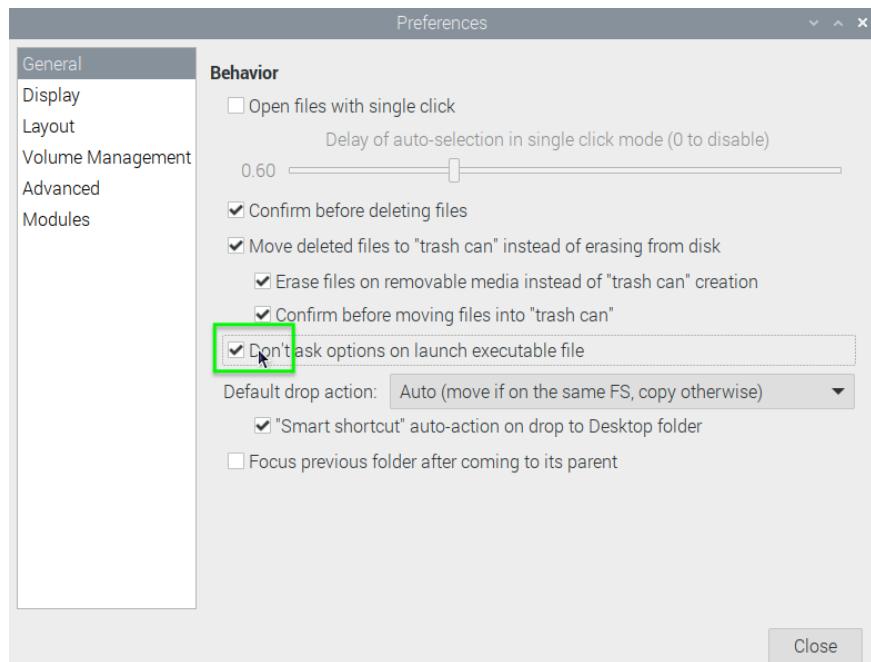


Figure 55 Desktop gradio/vgradio startup options

You may also want to click on “Open files with a single click”. Close the window. Retry clicking on either of the Radio desktop icons. The program should be executed straight away.

## Trouble shooting problems with MPD

Most problems are due to the following:

- Incompatibility with the **pulseaudio** package
- Sound mixer volume set to zero or very low volume
- Incorrect setup of the **/etc/mpd/mpd.conf** file
- Incorrect setup of the **/etc/asound.conf**
- If using a sound card, no driver loaded in **/boot/firmware/config.txt**
- The Raspberry Pi audio is configured to use the **HDMI** output instead of the output jack or sound card.

Check the audio jack cable first before doing anything else.

The **/var/log/mpd/mpd.log** file is the place to look first if all other things seem normal.

Remove the **pulseaudio** package unless required for Bluetooth audio devices.

```
$ sudo apt remove pulseaudio
```

Re-run the **configure\_audio.sh** program as shown in the section **Error! Reference source not found.** on page **Error! Bookmark not defined..** You must do this to configure MPD to work with pulseaudio.

If using the onboard audio output there should not be a problem. The standard **/etc/mpd.conf** settings should be OK. Only if pulse audio is not removed or the Alsa mixer volume is not set can it lead to lack of sound. See *Installing pulseaudio* on page 27 .

If using a USB or HiFiBerry DAC:

1. Check to see if the DAC is visible using **aplay -l** command
2. Check that the **/etc/mpd.conf** is correctly configured.
3. Check that the mixer volume is correctly set.
4. For HiFiBerry DAC ensure **/boot/firmware/config.txt** contains the correct **dtoverlay** statement.

See **Error! Reference source not found.** on page **Error! Bookmark not defined..**

## MPD fails to install

During installation of MPD some files return a 404 error (Not found) the following message is seen.

```
Unable to fetch some archives, maybe run apt update or try with -fix-missing?
```

This is due to that an update was not previously carried out as shown in the section called **Error! Reference source not found.** on page **Error! Bookmark not defined..** Perform the update and upgrade as shown and re-install MPD and MPC.

## Music Player Daemon won't start

The MPD daemon logs to the **/var/log/mpd/mpd.log** file. Examine this file for errors. The MPD daemon is dependent on good M3U files so check that these are correct as described in the section called **Creating and Maintaining Playlist files** on page 169.

### *Missing configuration file in mpd.service file*

This is normally only a problem with the upgraded version (0.22.x onwards) of MPD.

```
$ sudo systemctl start mpd
Job for mpd.service failed because the control process exited with error
code.
See "systemctl status mpd.service" and "journalctl -xe" for details.
```

### Check the status

```
$ systemctl status mpd.service
● mpd.service - Music Player Daemon
  Loaded: loaded (/lib/systemd/system/mpd.service; disabled; preset:
  enabled)
    Active: failed (Result: exit-code) since Sun 2023-11-19 17:21:10 GMT;
  5s ago
      Duration: 4min 18.727s
TriggeredBy: × mpd.socket
    Docs: man:mpd(1)
           man:mpd.conf(5)
           file:///usr/share/doc/mpd/html/user.html
   Process: 2287 ExecStart=/usr/bin/mpd --systemd $MPDCONF (code=exited,
status=1/FAILURE)
 Main PID: 2287 (code=exited, status=1/FAILURE)
    CPU: 556ms

Nov 19 17:21:10 bookworm32 systemd[1]: Starting mpd.service - Music Player
Daemon...
Nov 19 17:21:10 bookworm32 mpd[2287]: exception: No configuration file found
Nov 19 17:21:10 bookworm32 systemd[1]: mpd.service: Main process exited,
code=exited, status=1/FAIL>
Nov 19 17:21:10 bookworm32 systemd[1]: mpd.service: Failed with result
'exit-code'.
```

In this case **/usr/local/bin/mpd** cannot find **/etc/mpd.conf**. Check that the **ExecStart** statement in **/usr/lib/systemd/user/mpd.service** contains the configuration file name.

```
ExecStart=/usr/local/bin/mpd --no-daemon /etc/mpd.conf
```

### *Missing Libraries cause service start to fail*

This is normally only a problem with the upgraded version (0.22.x) of MPD.

A message similar to the following shows that one or more required libraries are missing.

```
$ sudo systemctl status mpd.service
● mpd.service - Music Player Daemon
  :
Apr 12 01:54:59 Bullseye04 mpd[3460]: /usr/local/bin/mpd: error while loading
shared libraries: libao.
Apr 12 01:54:59 Bullseye04 systemd[1]: mpd.service: Main process exited,
code=exited, status=127/n/a
Apr 12 01:54:59 Bullseye04 systemd[1]: mpd.service: Failed with result 'exit-
code'.
Apr 12 01:54:59 Bullseye04 systemd[1]: Failed to start Music Player Daemon.
```

Solution: Install the pre-requisite libraries as shown in the section called *Installing the MPD upgrade from a Debian package* on page 117.

Reset failed units mpd service units. and recheck.

```
$ sudo systemctl reset-failed mpd.socket mpd.service
```

Restart mpd service.

```
$ sudo systemctl start mpd.service
```

Re-check the mpd service.

```
$ sudo systemctl status mpd.service
● mpd.service - Music Player Daemon
   Loaded: loaded (/lib/systemd/system/mpd.service; disabled; preset: enabled)
     Active: active (running) since Mon 2024-03-11 12:47:12 GMT; 1s ago
   TriggeredBy: ● mpd.socket
     Docs: man:mpd(1)
           man:mpd.conf(5)
           file:///usr/share/doc/mpd/html/user.html
    Main PID: 2119 (mpd)
      Tasks: 6 (limit: 1577)
        CPU: 976ms
       CGroup: /system.slice/mpd.service
                 └─2119 /usr/bin/mpd -systemd
```

Finally remove any redundant libraries

```
$ sudo apt autoremove
```

### The MPD program may display a socket error

When starting the MPD daemon the following message is seen:

```
Starting Music Player Daemon: mpdlisten: bind to '[:1]:6600' failed: Failed to create socket: Address family not supported by protocol (continuing anyway, because binding to '127.0.0.1:6600' succeeded)
```

If this message is seen in the MPD log file this is simply because IP version 6 (IPv6) isn't installed so the message doesn't affect operation of the MPD. To prevent it from happening configure the *bind\_to\_address* parameter in the **/etc/mpd.conf** file to "any". The installation procedure should normally set this anyhow.

### The MPD daemon complains about the avahi daemon

The following message is seen in the **/var/log/mpd/mpd.log** file

```
Apr 10 15:37 : avahi: Failed to create client: Daemon not running.
```

It would be normal to change the **zeroconf\_enabled** parameter in the **/etc/mpd.conf** file to "yes" to overcome this. The Avahi daemon is used in small local networks (such as a home LAN). The avahi-

daemon process handles **mDNS** (Multicast DNS), which is used for name resolution and service discovery within the local network. The default for MPD is to disable

Since version 7.1 onwards the following may be seen in /var/log/mpd.log:

```
zeroconf: No global port, disabling zeroconf
```

This is not an error. This is because the radio is now using **mpd.socket** in addition to **mpd.service**. The **mpd.socket** service starts the **mpd.service** as required so that multiple applications can access MPD. In the case of the radio this is either **radiod** or the Web service for the radio.

### Radiod → mpd.socket → mpd.service

Changing **zeroconf\_enabled** to “yes” will have no effect as it will be automatically disabled as **mpd.socket** is being used. By the way zeroconf is the generic term for any service providing mDNS such as Avahi (Linux) or Bonjour (Apple) etc.

## LCD Problems

### LCD screen not showing anything

Check that the wiring conforms to the *wiring list* on page **Error! Bookmark not defined..** Make sure that pin 3 is grounded (0V) to give maximum contrast or if a contrast potentiometer is fitted then make sure it is at the maximum setting. Make sure the correct Radio variant has been selected. Re-run the **configure\_radio.sh** program as shown in the section **Error! Reference source not found.** on page **Error! Bookmark not defined..**

Run the **test\_lcd.py** program to see if the LCD displays anything. This runs independently of any other software and can be used stand alone. Run the **wiring.py** program (See page 242).

### The LCD only displays hieroglyphics

This can be caused by incorrect wiring of the LCD. This problem has also been experienced with faulty LCD hardware particularly when re-booting the Raspberry PI.

Check the wiring conforms to the *wiring list* on page **Error! Bookmark not defined..** In particular check the data lines to pins 11, 12, 13 and 14 (See LCD wiring on page **Error! Bookmark not defined..**). Retest the LCD using the **test\_lcd.py** program.

If the wiring is correct run the **configure\_radio.sh** script to select the correct revision of the board and restart the program. Run the **wiring.py** program (See page 242).

### The LCD displays hieroglyphics or goes blank occasionally

If the LCD is normally working OK but goes wrong when switching on and off lights this is due to Electromagnetic Interference (EMI). See the section **Error! Reference source not found.** on page **Error! Bookmark not defined..**

### LCD backlight not working

Check that pins 15 and 16 of the LCD display have +5V and 0V (GND) respectively.

See section called **Error! Reference source not found.** on page **Error! Bookmark not defined..**

## LCD only displays dark blocks on the first line

This is normal when the raspberry PI starts up. The display should work with the **test\_lcd.py** program. If the **test\_lcd.py** program still doesn't display anything then check that the wiring conforms to the *wiring list* on page 314. If you are using the Adafruit LCD plate the make sure that you are running the **ada\_radio.py** program and not one of the other programs (See **Error! Reference source not found.** on page **Error! Bookmark not defined.**). **configure\_radio.sh** script to select the correct radio daemon. Run the **wiring.py** program (See page 242).

## Constant alternate display of Station Name and Volume

Problem: The LCD screen continually switches between displaying station name and volume. Also the radio log file displays "ERROR radio.\_setVolume error vol=nn" where nn is the volume level. This is due an incompatibility with the **pulseaudio** package.

Solution: Remove the **pulseaudio** package.

```
$ sudo apt remove pulseaudio
```

## Display scrolling too fast or too slow

Adjust the **scroll\_speed** parameter in **/etc/radiod.conf**. Adjust between 0.001 and 0.5 seconds.

```
scroll_speed = 0.2
```

## Adafruit LCD backlight problems

When using an AdaFruit Blue/White 2x16 character display, if stepping through the menus the backlight goes off and on. This is because with this type of display the backlight is only using one of the RGB inputs. To solve this problem, set all backlight colours except sleep\_color to WHITE.

```
bg_color=WHITE
mute_color=WHITE
shutdown_color=WHITE
error_color=WHITE
search_color=WHITE
info_color=WHITE
menu_color=WHITE
source_color=WHITE
sleep_color=OFF
```

## Grove JHD1313 LCD with RGB backlight

When using the Grove JHD1313 LCD the following output is seen from the **systemctl status radiod** command:

```
Sep 18 16:32:57 piradio systemd[1]: Started Radio daemon.
Sep 18 16:32:59 piradio radiod.py[502]: Traceback (most recent call last):
Sep 18 16:32:59 piradio radiod.py[502]:   File "/usr/share/radio/radiod.py",
line 29, in <module>
Sep 18 16:32:59 piradio radiod.py[502]:     from display_class import
Display
Sep 18 16:32:59 piradio radiod.py[502]:   File
"/usr/share/radio/display_class.py", line 22, in <module>
Sep 18 16:32:59 piradio radiod.py[502]:     from PIL import ImageColor
Sep 18 16:32:59 piradio radiod.py[502]: ModuleNotFoundError: No module named
'PIL'
```

```
Sep 18 16:32:59 piradio systemd[1]: radiod.service: Main process exited,  
code=exited, status=1/FAILURE  
Sep 18 16:32:59 piradio systemd[1]: radiod.service: Failed with result  
'exit-code'.
```

Install **PIL** as shown in the section called **Error! Reference source not found.** on page **Error! Bookmark not defined..**

## Pimoroni Pirate Radio problems

### Volume UP button (Y button) not working.

Pimoroni are using either GPIO 20 or 24 for button Y (Volume UP). By default, the setting for **volume\_up** in **/etc/radiod.conf** is GPIO 20. For boards produced before January 2020 this is GPIO 24:

```
right_switch=24
```

See section **Error! Reference source not found.** on page **Error! Bookmark not defined.** for more details. If your card is using GPIO 24 then manually edit the **/etc/radiod.conf** file as shown above.

## Playlist problems

### The display shows the message “No playlists”

Cause: There are no playlists found in **/var/lib/mpd/playlists**.

Check to see if there are any playlists in **/var/lib/mpd/playlists**. You should see files with the **m3u** extension.

Solution: Create playlists by running the **create\_playlists.py** program as shown in the section called Creating and Maintaining Playlist files on page 169.

Restart the radio.

### Cannot play newly mounted network share

Cause: Although the share may have been created, there are no playlists found in **/var/lib/mpd/playlists**.

Solution: Create playlists by running the **create\_playlists.py** program as shown in the section called Creating and Maintaining Playlist files on page 169. Select network share from the menu.

## I2C and SMBUS problems

### Import errors

The following is seen:

```
$ sudo ./radiod.py nodaemon  
Radio running pid 825  
:  
ImportError: No module named PIL
```

The **python3-pil** package has not been installed. Run the following to install it:

```
$ sudo apt-install python3-pil
```

## PiFace CAD and SPI problems

### PiFace CAD not detected

When running the radio configured for PiFace CAD the following is seen when running the radio in **nodaemon** mode.

```
pifacecad.core.NoPiFaceCADDetectedError: No PiFace Control and Display board  
detected (hardware_addr=0, bus=0, chip_select=1).
```

This is due to the Raspberry Pi being unable to operate with the default speed of the SPI bus. Edit **spi.py** program file as shown in the section **Error! Reference source not found.** on page **Error! Bookmark not defined..**

## Olimex OLED problems

### Radio does not start with Olimex screen

Run the program in no daemon mode as shown in the section called *Running the radio program in nodaemon mode* on page 247.

```
Traceback (most recent call last):  
:  
ImportError: No module named PIL
```

If the above is seen then the libraries for the Olimex display have not been installed. Carry out the instructions as shown in the section called *Running the radio program in nodaemon mode* on page 247. If the problem is not missing libraries check wiring and connections to the OLED as these can be easily damaged.

### OLED Screen is displaying upside down

This can be changed by setting the **flip\_vertical** setting in **/etc/radiod.conf** to yes or no.

```
flip_display_vertically=yes
```

Note that this setting only works with OLEDs and not LCDs. Also, it does not work with SSD1306 driver. If you need to flip the screen on an SSD1306 device then use the LUMA.SSD1306 driver.

## Pimoroni Pirate Audio crashes

### Cause 1 SPI not enabled

These devices use an ST7789 OLED display. This requires the SPI interface to be enabled.

A typical crash is seen below when a status check is run.

```
$ sudo systemctl status radiod  
● radiod.service - Radio daemon  
  Loaded: loaded (/lib/systemd/system/radiod.service; enabled; vendor  
  :  
  :  
  "/usr/local/lib/python2.7/dist-packages/ST7789/__init__.py", line 124, in  
  __init__
```

```
Oct 23 18:09:58 raspberrypi radiod.py[356]:     self._spi =
spidev.SpiDev(port, cs)
Oct 23 18:09:58 raspberrypi radiod.py[356]: IOError: [Errno 2] No such file
or directory
Oct 23 18:09:58 raspberrypi systemd[1]: radiod.service: Main process exited,
code=exited, status=1/FAILURE
Oct 23 18:09:58 raspberrypi systemd[1]: radiod.service: Failed with result
'exit-code'.
```

Solution: Using **raspi-config** to enable the SPI interface. See the section called Configure the SPI Kernel Module on page 5.

### Cause 2 Memory pointer error

There is a suspected issue with the **PIL** library (Python Imaging Library) also called Pillow which gets a memory fault (Invalid pointer) after about three to four hours of running. It is hoped to fix this in a later release. In the meantime, edit the **/lib/systemd/system/radiod.service** file and un-comment the Restart options as shown below. This will restart the program on failure although it may skip a station.

```
Restart=on-failure
RestartSec=5s
```

## Rotary encoder problems

### Radio program crashes due to rotary encoder error

During startup a message similar to the following is seen:

```
Rotary Encoder initialise error GPIO 15 Failed to add edge detection
Rotary Encoder initialise error GPIO 24 Failed to add edge detection
```

This will normally only be seen if running on the **Bookworm** Operating OS. This is because the standard **RPi.GPIO** library that has been in use since the introduction of the Raspberry Pi no longer works on **Bookworm**. To correct this problem run the following command:

```
$ touch /usr/share/radio/RPi/__init__.py
```

The above instruction creates an empty file called **\_\_init\_\_.py** which enables the **GPIOconverter** program. All calls to RPi.GPIO will now be diverted to the newer **Igpio** library to control the GPIO pins. See *The GPIOconverter program* on page 64.

### Rotary encoders working in reverse

When a rotary encoder is turned clockwise the encoder operates in an anticlockwise direction and vice versa. To solve this swap the wires to the encoder (A and B or DT and CLK) either at the rotary encoder end or on the Raspberry Pi GPIO header. If this is not possible edit the **/etc/radiod.conf** file and swap the GPIO pin assignments

```
up_switch=24
down_switch=23
left_switch=14
right_switch=15
```

Swap the **up\_switch** and **down\_switch** GPIO settings for the channel rotary encoder and **left\_switch** and **right\_switch** settings for the volume encoder. The above settings are the normal settings.

### Rotary encoder Mute or Menu switch not working

Check that the physical wiring for the problem encoder and that it corresponds to the settings in shown in **/etc/radiod.conf**.

```
menu_switch=17  
mute_switch=4
```

It is very unlikely that the rotary encoder is faulty. It is more likely that the wiring is faulty or does not match the settings in **/etc/radiod.conf**. If the wiring and configuration settings appear to be OK try replacing the encoder. Also see *Push buttons or rotary encoders not working at all* on page 215.

### Testing rotary encoders

Run the test programs shown in the section called *Testing rotary encoders* on page 240. These programs display the configuration found in **/etc/radiod.conf**.

Use the **wiring.py** program to display your configured wiring. Does this match the actual wiring? If not, either correct the wiring or amend the switch settings in **/etc/radiod.conf**.

Check wiring in particular the common pin must be connected to ground (and not 3.3 volts).

Also, you can use the **test\_gpios.py** program to confirm how you have wired a specific button or rotary encoder. See *The GPIO converter program* on page 64 for further explanation.

### RGB I2C rotary encoder - Missing module ioexpander

The following message is displayed when using RGB I2C rotary encoders:  
ImportError: No module named ioexpander.

It is necessary to install **ioe-python**. See section called **Error! Reference source not found.** on page **Error! Bookmark not defined..**

### Push button radio problems

#### Radio program crashes due to push button error

During startup a message similar to the following is seen:

```
Rotary Encoder initialise error GPIO 15 Failed to add edge detection  
Rotary Encoder initialise error GPIO 24 Failed to add edge detection
```

This is the same problem that can occur with rotary encoders. See *Radio program crashes due to rotary encoder error* on page 214 for the solution.

#### Push buttons or rotary encoders not working at all

This is usually one of two problems.

1. Physical wiring problems
2. Incorrect configuration in **/etc/radiod.conf**

Obviously recheck that the wiring buttons encoders connections conform to the wiring shown in **Error! Reference source not found.** on page **Error! Bookmark not defined..** Try running the **test\_gpios.py** program to check how the buttons/rotary encoders are actually wired.

```
$ cd /usr/share/radio/  
$ ./test_gpios.py --pull_up  
GPIO 17 falling  
GPIO 17 rising  
GPIO 4 falling  
GPIO 4 rising
```

Now check the results against the switch configuration in **/etc/radiod.conf**

```
# Switch settings for Rotary encoders or buttons  
menu_switch=17  
mute_switch=4  
up_switch=24  
down_switch=23  
left_switch=14  
right_switch=15
```

If they don't agree correct the problem in either the physical wiring or the configuration.

### Keep getting erroneous button-down events

This can happen with KY-040 rotary encoders which have their own pull-up resistors and the VCC pin has not been connected to +3V.

Without the connection to VCC +3V the A, B and switch are all connected together via the 10K resistors and will interfere with each other (crosstalk). Connect the VCC to 3.3V as shown in See **Error! Reference source not found..**

### Stream decode problems

The radio may display a message similar to the following:

```
ERROR: problems decoding http://173.244.194.212:8078
```

This is due to an invalid URL (In the above example this is <http://173.244.194.212:8078>) in one of the M3U files. Locate the offending URL in the play list file in the **/var/lib/radiod/stationlist** file.

Either correct the radio stream URL or remove it all together. Re-run the **create\_stations.py** program. Also check that the file URL is not the pointer to the playlist file (See section **Creating and Maintaining Playlist files** on page 169).

### Cannot mount remote network drive

There are just too many possibilities to cover all of these here. However, a few common problems are covered here:

**Error:** mount error(112): Host is down

**Cause:** Missing or incorrect **vers** statement.

The vers parameter can be 1.0, 2.0 or 3.0. The mount.cifs man page incorrectly states that version 1.0 is the default. This is correct for the Bullseye OS. Try adding **vers=1.0** to the mount statement in **/var/lib/radiod/share**.

```
...uid=pi,gid=pi,vers=1.0 ...
```

**Error:** mount error(115): Operation now in progress

**Cause:** Most likely an incorrect IP address

**Error:** NFS mount hangs

**Cause:** Most likely an incorrect IP address

**Error:** mount.nfs: access denied by server while mounting <ip address>:/music

**Cause:** The volume name is missing – for example /volume1/music

**Error:** mount error(16): Device or resource busy

**Cause:** The share mount directory is in use because a mount has already been done. Run the umount command.

**Error:** mount error(2): No such file or directory

**Cause:** The path specified in the mount doesn't exist

**Error:**

mount.nfs: rpc.statd is not running but is required for remote locking.

mount.nfs: Either use '-o nolock' to keep locks local, or start statd.

mount.nfs: an incorrect mount option was specified

**Cause:**

You need to include the “–o noclock” option.

If the error isn't in the above list then search the Web for suggestions.

## Sound problems (including Bluetooth)

**Re-run the audio configurator program**

If experiencing any problems with sound, always re-run the audio configuration program and select the correct option for your sound system, for example Headphones, HDMI or DAC etc.

```
$ cd /usr/share/radio  
$ ./configure_audio.sh
```

Select the correct sound configuration as shown the section called **Error! Reference source not found.** on page **Error! Bookmark not defined..** Reboot the Raspberry Pi.

### Check to see if any sound cards are configured

Check first if the card is visible using the **aplay** command. The DAC card should be visible.

```
$ aplay -l  
**** List of PLAYBACK Hardware Devices ****
```

```
card 0: sndrpihifiberry [snd_rpi_hifiberry_dacplus], device 0: HiFiBerry  
DAC+ HiFi pcm512x-hifi-0 []  
Subdevices: 0/1  
Subdevice #0: subdevice #0
```

Re-run the `configure_audio.sh` program as shown in the section **Error! Reference source not found.** on page **Error! Bookmark not defined..**



**Note:** If you have configured Bluetooth speakers then you will not normally see any cards. This is because Bluetooth devices do not use **alsa** but use **pulseaudio-module-bluetooth** instead. Use the **bluetoothctl info** command instead. If using Bluetooth devices see the section called Bluetooth device no sound on page 221.

### Test the sound card using aplay

Run the following command:

```
$ aplay /usr/share/sounds/alsa/Front_Center.wav
```

You should hear a woman's voice saying "Front Centre". If not check the installation of the sound card as shown in **Error! Reference source not found.** on page **Error! Bookmark not defined..** Do not proceed further until the problem is corrected.

Other sounds can also be used from the `/usr/share/sounds/alsa` directory

```
$ ls /usr/share/sounds/alsa/  
Front_Center.wav Front_Right.wav Rear_Center.wav Rear_Right.wav  
Side_Right.wav Front_Left.wav Noise.wav Rear_Left.wav Side_Left.wav
```

### HiFBerry or other types of DAC no sound

If the **DAC** card is still not visible check the following:

Check that the B output of the channel rotary encoder is wired to GPIO 10 (pin 19) and not GPIO 18 (pin 12). GPIO18 is used by the DAC plus. Also check that the **down\_switch** parameter in `/etc/radiod.conf` is set to 10 (Comment out `down_switch=18`) to reflect the actual wiring.

```
#down_switch=18  
down_switch=10
```

Make sure that the `/boot/firmware/config.txt` file contains the correct **dtoverlay** command as shown in *Table 11 Sound card Device Tree overlays* on page 306. The following example is for a HiFiBerry DAC plus.

```
dtoverlay=hifiberry-dacplus
```

Finally modify the **audio\_output** section in `/etc/mpd.conf`. The **Device** parameter should point to the correct card.

```
audio_output {  
    type      "alsa"  
    name      "DAC"  
    device    "hw:0,0"  
    #format   "44100:16:2" # optional  
    mixer_device  "PCM"  
    mixer_control "PCM"
```

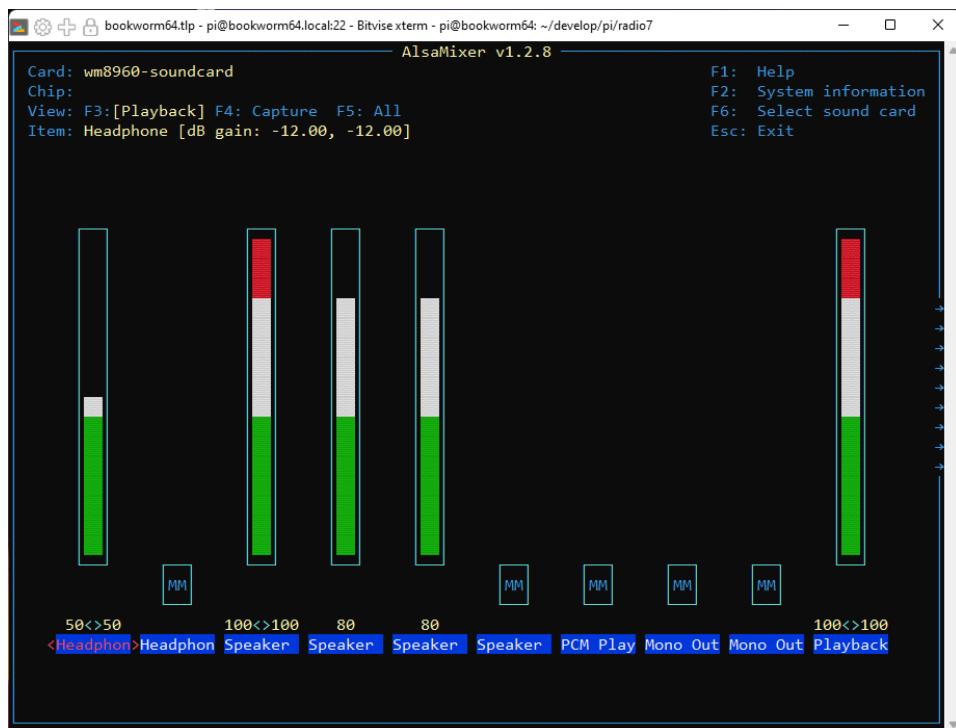
```
    mixer_type    "software"
}
```

Reboot the Raspberry Pi and retest.

## Waveshare WM8960 sound card problems

### Sound very low

Run the **alsamixer** program and if necessary, press F6 to select the **wm8960-soundcard**.



Adjust the volume levels for the Speaker and Headphone levels as required.

Store the settings with the following command.

```
$ sudo alsactl store --file /etc/wm8960-soundcard/wm8960_asound.state
```

### No sound at all

Check the status of the **wm8960-soundcard**.

```
systemctl status wm8960-soundcard.service
× wm8960-soundcard.service - WM8960 soundcard service
   Loaded: loaded (/lib/systemd/system/wm8960-soundcard.service; enabled;
             preset: enabled)
   Active: failed (Result: exit-code) since Tue 2024-09-10 12:39:43 BST;
             1h 23min ago
     Process: 504 ExecStart=/usr/bin/wm8960-soundcard (code=exited, status=99)
      Main PID: 504 (code=exited, status=99)
        CPU: 229ms
:
:
```

If it has failed with status code **99** check that the procedure in the section called **Error! Reference source not found.** on page **Error! Bookmark not defined.** has been properly carried out, especially disabling the **on-board audio** output and the DRM VC4 V3D driver in **/boot/firmware/config.txt**.

Also, in the case of **Bookworm 32-bit** systems it is necessary to add the following parameter to the end of the **/boot/firmware/config.txt** file.

```
arm_64bit=0
```

Reboot and check that the **aplay** command displays only one entry for the **wm8960soundcard** and that it is card 0.

```
$ aplay -l
**** List of PLAYBACK Hardware Devices ****
card 0: wm8960soundcard [wm8960-soundcard], device 0: bcm2835-i2s-wm8960-
hifi wm8960-hifi-0 [bcm2835-i2s-wm8960-hifi wm8960-hifi-0]
Subdevices: 0/1
```

## Bluetooth device no sound

### *Check Bluetooth device connected*

The following commands are useful for checking that the Bluetooth device is connected. The actual address of your Bluetooth device should be used

```
$ bluetoothctl devices
Device 00:75:58:41:B1:25 SP-AD70-B
```

If no device is seen then rescan and pair your device using the instructions in the section *Configure the SPI Kernel Module*

Skip this section unless installing PiFace CAD or devices using the ST7789 OLED display.

If installing the radio on PiFace CAD or the ST7789 OLED display it is necessary to enable the SPI interface. For example, if PiFace CAD was selected the following message will be seen:

```
The chosen display PiFace CAD requires the
SPI kernel module to be loaded at boot time.
The program will call the raspi-config program
Select the following options on the next screens:
  5 Interfacing options
  P4 Enable/Disable automatic loading of SPI kernel module

Press enter to continue:
```

Press enter and select option 1:

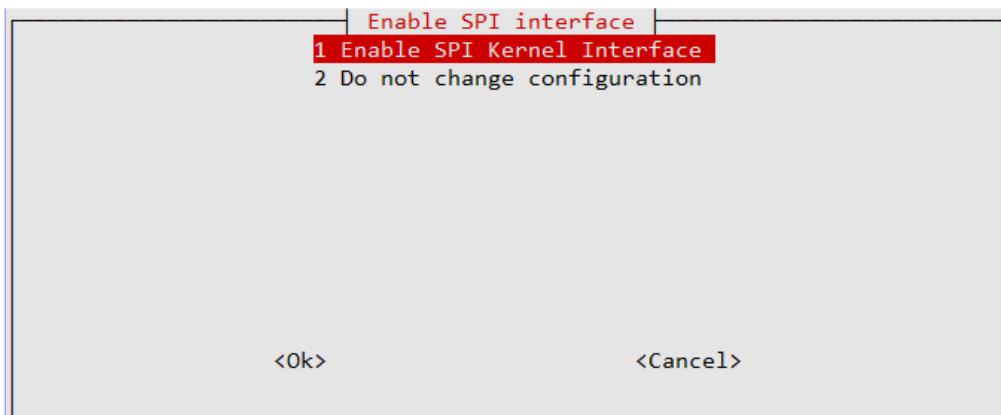


Figure 1 Enable SPI Kernel Module

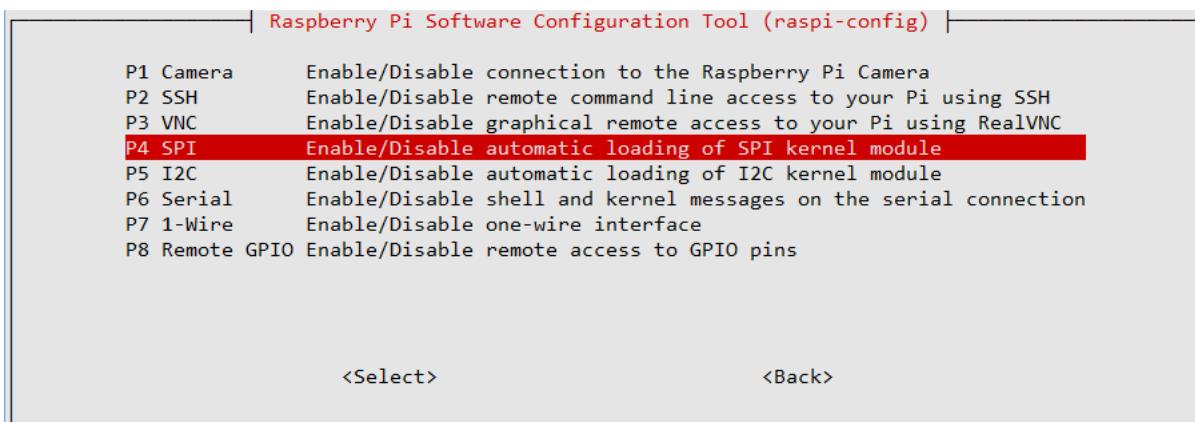
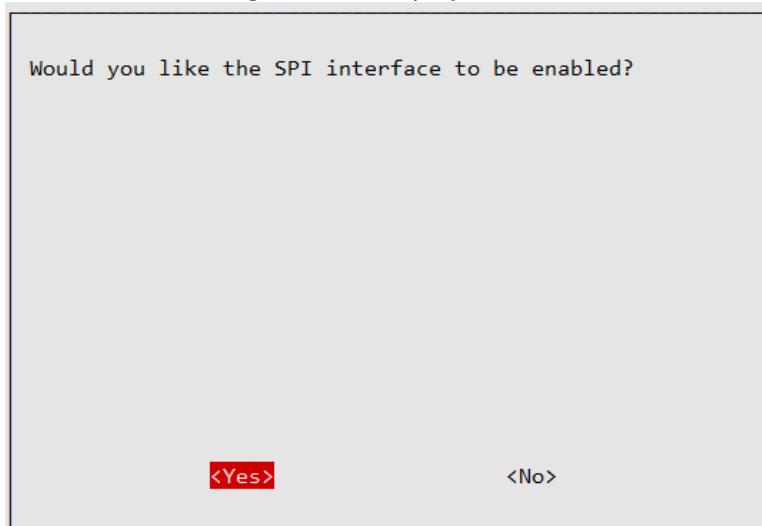


Figure 2 Enable SPI kernel module option

Select the option P4 SPI. The following screen is displayed:



Select Yes to enable the SPI kernel module. Select "Finish" to exit.

```
Would you like the ARM I2C interface to be enabled?
```

<Yes>

<No>

Connecting a Bluetooth audio device on page 5.

Ping the Bluetooth device:

```
$ sudo l2ping 00:75:58:41:B1:25
Ping: 00:75:58:41:B1:25 from DC:A6:32:05:36:9D (data size 44) ...
44 bytes from 00:75:58:41:B1:25 id 0 time 5.72ms
44 bytes from 00:75:58:41:B1:25 id 1 time 4.89ms
44 bytes from 00:75:58:41:B1:25 id 2 time 4.83ms
44 bytes from 00:75:58:41:B1:25 id 3 time 4.91ms
```

Try connecting to it:

```
$ bluetoothctl connect 00:75:58:41:B1:25
Attempting to connect to 00:75:58:41:B1:25
Connection successful
```

Check the HCI UART controller

```
$ dmesg | grep -i Bluetooth
[ 28.274746] Bluetooth: Core ver 2.22
[ 28.274837] Bluetooth: HCI device and connection manager initialized
[ 28.274860] Bluetooth: HCI socket layer initialized
[ 28.274878] Bluetooth: L2CAP socket layer initialized
[ 28.274904] Bluetooth: SCO socket layer initialized
[ 28.285231] Bluetooth: HCI UART driver ver 2.3
[ 28.285249] Bluetooth: HCI UART protocol H4 registered
[ 28.285335] Bluetooth: HCI UART protocol Three-wire (H5) registered
[ 28.285601] Bluetooth: HCI UART protocol Broadcom registered
[ 28.534358] Bluetooth: BNEP (Ethernet Emulation) ver 1.3
[ 28.534367] Bluetooth: BNEP filters: protocol multicast
[ 28.534383] Bluetooth: BNEP socket layer initialized
```

*Check Bluetooth powered on*

Run the following command

```
$ bluetoothctl show | grep -i powered
```

```
Powered: yes
```

If the above check shows “Powered: no” switch power back on with:

```
$ bluetoothctl power on  
Changing power on succeeded
```

### *Check MPD and radio configuration for the Bluetooth device*

TO BE REVISED. The **configure\_audio.sh** program should have amended the **type**, **name**, **server** and **mixer\_type** fields in **/etc/mpd.conf**.

New (not working in Bookworm) configuration

```
audio_output {  
    type      "pulse"  
    name      "SP-AD70-B"  
    server    "127.0.0.1"  
    mixer_type "software"  # optional  
    # mixer_device   "default"  # optional  
    # mixer_control  "PCM"      # optional  
    # mixer_index    "0"        # optional  
}
```

All being well the configuration should match your device if not correct it.

Check that the **bluetooth\_device** parameter in the **[RADIOD]** section of **/etc/radiod.conf** has been configured with the Bluetooth device ID of your device:

```
# Bluetooth device ID - Replace with the ID of your bluetooth  
speakers/headphones  
# Example: bluetooth_device=00:75:58:41:B1:25  
# Use the following command to display paired devices  
# bluetoothctl paired-devices  
bluetooth_device=00:75:58:41:B1:25
```

Amend it if it is not correct.

Restart the radio or reboot. Music should be heard from the Bluetooth device .

The **/etc/asound.conf** file is not used by the **pulseaudio-module-bluetooth** module.

### *Problems pairing the bluetooth device*

TO BE REVISED

Check the hciuart daemon.

```
$ systemctl status hciuart  
● hciuart.service - Configure Bluetooth Modems connected by UART  
   Loaded: loaded (/lib/systemd/system/hciuart.service; enabled; vendor  
   preset: enabled)  
     Active: active (running) since Tue 2019-11-05 16:17:11 GMT; 16h ago  
       Process: 330 ExecStart=/usr/bin/btuart (code=exited, status=0/SUCCESS)  
     Main PID: 818 (hciattach)  
        Tasks: 1 (limit: 1599)  
      Memory: 1000.0K  
     CGroup: /system.slice/hciuart.service
```

```
L-818 /usr/bin/hciattach /dev/serial1 bcm43xx 3000000 flow -
```

### Connection problems

Sometimes a paired device may fail to connect.

```
[bluetooth]# paired-devices
Device 00:75:58:41:B1:25 SP-AD70-B
[bluetooth]# connect 00:75:58:41:B1:25
Attempting to connect to 00:75:58:41:B1:25
Failed to connect: org.bluez.Error.Failed
```

If you have problems connecting to your Bluetooth device, try removing it and re-pairing it.

```
[bluetooth]# remove 00:75:58:41:B1:25
```

Repeat the procedure *Configure the SPI Kernel Module*

Skip this section unless installing PiFace CAD or devices using the ST7789 OLED display.

If installing the radio on PiFace CAD or the ST7789 OLED display it is necessary to enable the SPI interface. For example, if PiFace CAD was selected the following message will be seen:

```
The chosen display PiFace CAD requires the
SPI kernel module to be loaded at boot time.
The program will call the raspi-config program
Select the following options on the next screens:
  5 Interfacing options
  P4 Enable/Disable automatic loading of SPI kernel module

Press enter to continue:
```

Press enter and select option 1:

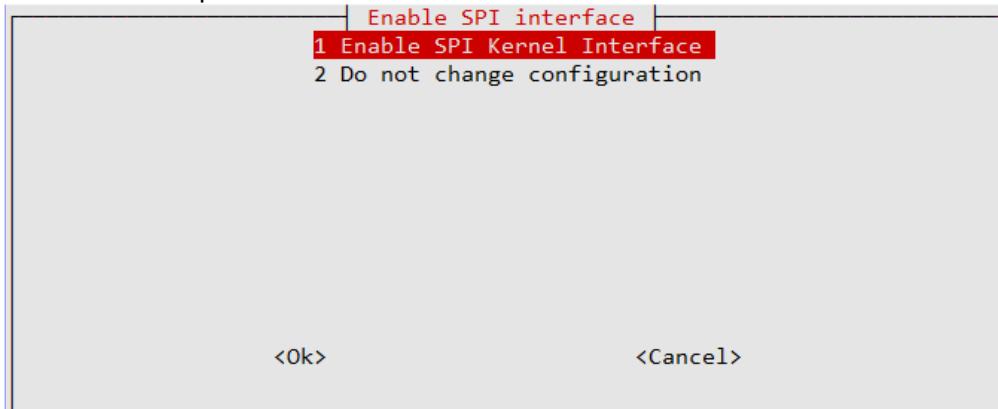


Figure 1 Enable SPI Kernel Module

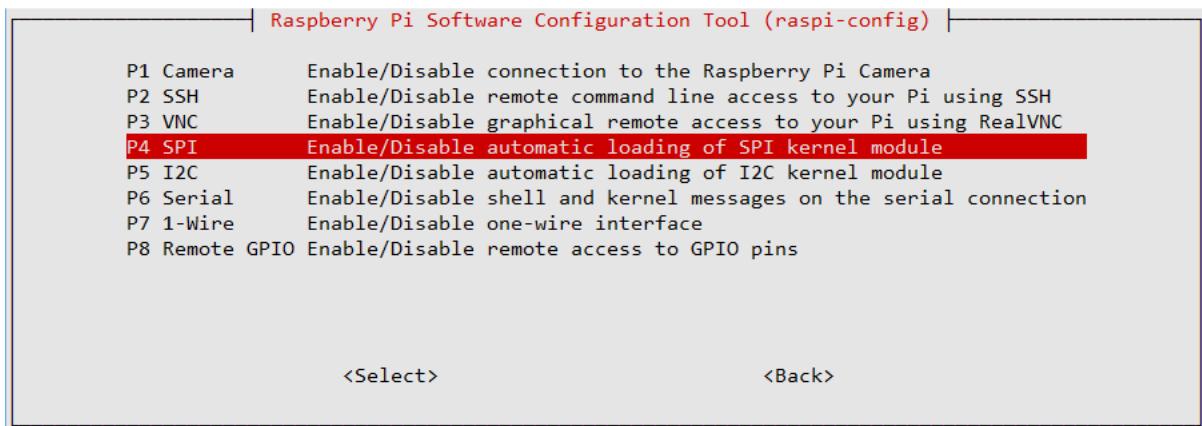
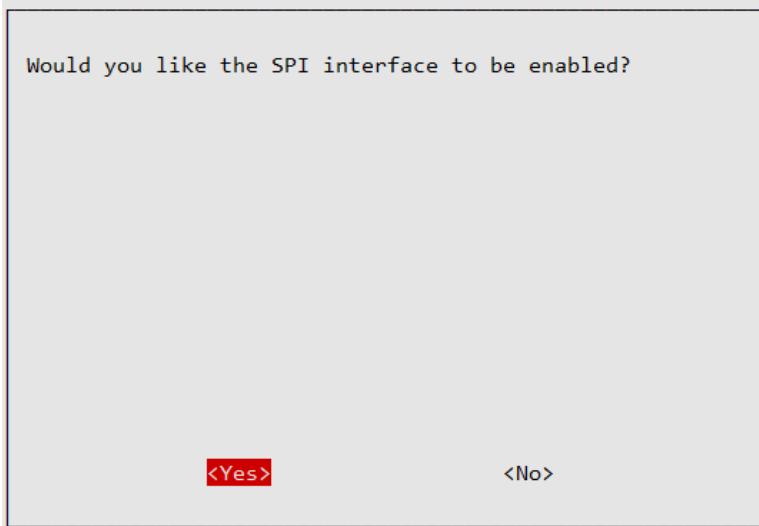
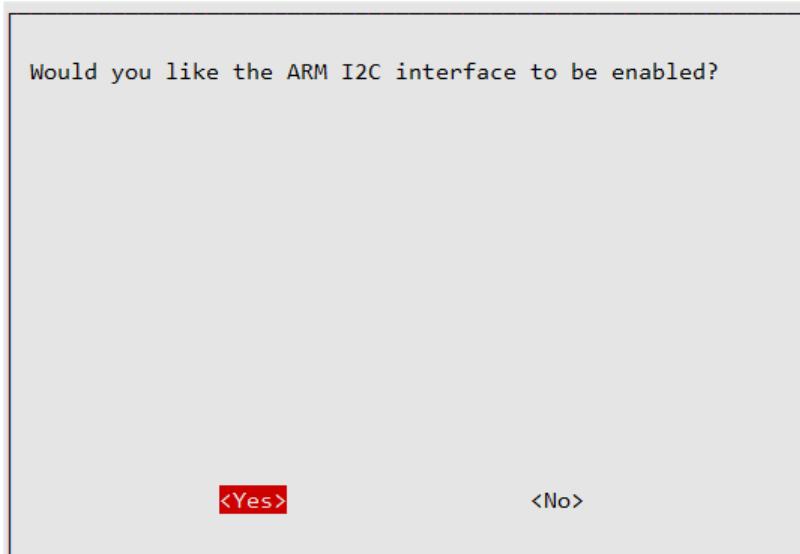


Figure 2 Enable SPI kernel module option

Select the option P4 SPI. The following screen is displayed:



Select Yes to enable the SPI kernel module. Select “Finish” to exit.



Connecting a Bluetooth audio device on page 5.

### No music heard from Bluetooth device

Check that all of the above instructions have been correctly followed.

Check the status of the device using **bluetoothctl info**. Check that it is paired, connected and trusted.

The following is an example using device ID 00:75:58:41:B1:25:. The ‘:’ character means output not shown.

```
$ bluetoothctl info 00:75:58:41:B1:25
Device 00:75:58:41:B1:25 (public)
  Name: SP-AD70-B
  Alias: SP-AD70-B
  Class: 0x00240404
  Icon: audio-card
  Paired: yes
  Trusted: yes
  Blocked: no
  Connected: yes
  LegacyPairing: no
  UUID: Audio Sink           (0000110b-0000-1000-8000-
:
:
```

Try changing channels on radio. This causes MPD to retry connecting. Check the Alsamixer.

Run the following:

```
$ bluetoothctl connect <Your Bluetooth Device>
```

### Noisy interference on the radio

If there is noise interference when playing the radio and this is still present even when the radio is muted this can be for several reasons. This can happen with a wired Ethernet connection and a Wi-Fi dongle are connected to the Raspberry Pi and the Ethernet activity is being picked up by the Wi-Fi dongle. This can be cured by using either a wireless adapter or the Ethernet connection and not both. Another common cause can be an inadequate power supply.

See on page **Error! Bookmark not defined..** Unfortunately, the later 40-pin versions of the Raspberry Pi seem to be more prone to interference. The recommendation is to use a USB DAC or a suitable DAC card.

### Humming sound on the radio

This is usually due to a ground loop somewhere in the design. See the section called **Error! Reference source not found.** on page **Error! Bookmark not defined..**

### Speaker Tests

There are a number of diagnostics available for testing speakers.

#### Simple white noise speaker test

Run speaker-test -c2 to generate white noise out of the speaker, alternating left and right.

If you have a mono output amplifier, the I2S amp merges left and right channels, so you'll hear continuous white noise:

```
$ speaker-test -c2
```

### *Simple WAV speaker test*

Once you've got something coming out, try to play an audio file with **speaker-test** (for WAV files, not MP3). Note that the following does not work with Bluetooth devices.

```
$ speaker-test -c2 --test=wav -w /usr/share/sounds/alsa/Front_Center.wav
```

You'll hear audio coming from left and right alternating speakers

### *Simple MP3 speaker test*

If you want to play a stream of music, you can install the **mpg123** program and use it to test a live stream. To install **mpg123** run:

```
$ sudo apt install -y mpg123
```

To test a stream.

```
$ mpg123 http://ice1.somafm.com/u80s-128-mp3
```

Any online mp3 can be used if the above is not working.

### *Cannot change volume when running Airplay*

This is most likely caused by the wrong mixer volume ID. In normal radio operation is controlled by MPD. As Airplay is nothing to do with MPD, the volume when using Airplay is controlled through the Alsa mixer. Run the following **amixer controls** command to identify the mixer volume ID.

The mixer volume ID can be identified as shown in the following command example.

```
$ amixer controls | grep -i volume
:
numid=4,iface=MIXER,name='Mic Playback Volume'
numid=8,iface=MIXER,name='Mic Capture Volume'
numid=6,iface=MIXER,name='Speaker Playback Volume'
```

Run the **set\_mixer\_id.sh** program to set the mixer volume id in **/var/lib/radiod/mixer\_volume\_id** file.

```
$ cd /usr/share/radio
$ sudo ./set_mixer_id.sh
```

### *Volume control errors*

The following error or similar is seen in the log file:

```
ERROR volume._setVolume error vol=50: [52@0] {setvol} problems setting
volume
```

Set the **mixer\_type** to "software" in **/etc/mpd.conf**. The ":" character means output not shown.

```
audio_output {
```

```
:           mixer_type      "software"
:  
}
```

## Operational problems

### When selecting the source, the USB stick isn't shown

You need to create the playlist for the USB stick first. See the section called **Creating and Maintaining Playlist files** on page 169.

### Radio daemon doesn't start or hangs

This is almost certainly a problem with either the MPD daemon or failed internet connection. Check the network connection and run installation tests on the MPD daemon. Occasionally a bad playlist file can also cause this problem. You can check that your Raspberry PI has an internet connection with the *ip addr* command.

The example below shows interface *eth0* connected as IP 192.168.2.22.

```
# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        brd 00:00:00:00:00:00
        link/ether 00:00:00:00:00:00 brd ff:ff:ff:ff:ff:ff
        inet 127.0.0.1/8 brd 00:00:00:00:00:00 scope host lo
            brd 00:00:00:00:00:00
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether b8:27:eb:fc:46:15 brd ff:ff:ff:ff:ff:ff
    inet 192.168.2.22/24 brd 192.168.2.255 scope global eth0
        brd ff:ff:ff:ff:ff:ff
```

### Volume control not working with DAC or USB speakers

This is hardware dependant. Not all USB hardware and drivers work with mixer type “hardware”. If this problem is being experienced try setting the **mixer\_type** parameter to “software”. Edit the **/etc/mpd.conf** file and change the mixer type to software.

```
mixer_type      "software"
```

Remove the # at the beginning of the line to enable software mixing and save the file. Restart the radio software.

 Note: This solution was provided by one of the constructors and is untested by the author.

### The radio keeps skipping radio stations

**MPD** will automatically skip bad stations. Remove any bad URLs from **/var/lib/radiod/stationlist** and re-run **create\_stations.py**.

### Source selection only shows the radio playlist

When attempting to play music from a USB stick, source selection only shows the radio playlist. This is usually because you have forgotten to run the **create\_playlist.sh** program.

See the section **Creating and Maintaining Playlist files** on page 169.

## Shoutcast playlist not created

Test the **get\_shoutcast.py** program.

```
$ ./get_shoutcast.py
Traceback (most recent call last):
  File "./get_shoutcast.py", line 20, in <module>
    import requests
ImportError: No module named requests
```

If you see the above message, install **python-requests**.

```
$ sudo apt install python-requests
```

## A station plays for a few seconds then skips to the next one

This is a known problem with the Music Player Daemon version 0.19.1 on Raspbian Bookworm.

There is only one solution and that is to create a new SD card with the latest Raspbian operating system (At the time of writing this was Raspbian Bullseye) and install the latest version of the radio. This will install Music Player Daemon version 0.19.21 which will cure this particular problem.

## The volume keeps getting reset to a 100% when the radio is restarted

This is almost certainly that the mixer volume id is incorrectly set. This is used by the Alsa mixer to set the volume. The radio software stores the mixer volume id in **/var/lib/radiod/mixer\_volume\_id**. The **mixer\_volume\_id** is normally set to the correct value by the **configure\_audio.sh** program during installation. Check the setting:

```
$ cat /var/lib/radiod/mixer_volume_id
1
```

Now run the amixer program to determine the numid of the ‘Master Playback Volume’.

```
$ amixer controls
numid=4,iface=MIXER,name='Master Playback Switch'
numid=3,iface=MIXER,name='Master Playback Volume'
numid=2,iface=MIXER,name='Capture Switch'
numid=1,iface=MIXER,name='Capture Volume'
```

Note the **numid** of the mixer volume and update the correct mixer id. In the above example it is 3. Now write the **numid** number to **/var/lib/radiod/mixer\_volume\_id** and check it.

```
$ sudo echo 3 > /var/lib/radiod/mixer_volume_id
$ cat /var/lib/radiod/mixer_volume_id
3
```

Edit the **/etc/radiod.conf** file and set the desired mixer pre-set volume as required.

```
mixer_preset=70
```

Now restart the radio.

## Raspberry Pi shuts down when the Menu button is pressed

The radio by default is designed to shut down the Raspberry if the Menu button is pressed in for more than 3-seconds. If this happens immediately upon pressing the Menu button then this is because the incorrect rotary class driver has been selected. Correct this by re-running the configurator and select the correct rotary class driver.

## Incorrect clock time displayed

If the time date and time displayed is incorrect for a short while after switch on, this is normal. The Raspberry Pi is not the same as a full featured PC motherboard at misses some of the features that are found on such boards. For example, it does not have a Real Time Clock (RTC). A RTC is a chip with battery back-up which maintains the system time when the device is switched off.

As there is no RTC the only time that the RPi has is the last time the device was used. Shortly after the network becomes active the system synchronises its time with one or more NTP time servers on the Internet using the Network Time Protocol (NTP). This used to be done with the NTP daemon **ntpd**. However, in more recent times this is done with a feature of **systemd** called **systemd-timesyncd**. This should synchronise the time within about 10 to 15 seconds although this can take a minute or more.

If the time is synchronising OK but you want to have the correct time immediately upon switch on see *Adding a Real Time Clock (RTC)* on page 105

## IR remote control problems

Is the IR sensor correctly wired up? Run the following:

```
$ cd /usr/share/radio  
$ ./test_events.py --config
```

This shows the kernel events configuration

```
/dev/input/event4 gpio_ir_recv gpio_ir_recv/input0  
Kernel event device rc2  
IR input device found at /dev/input/event4  
Boot configuration in /boot/firmware/config.txt  
dtoverlay= gpio-ir, gpio_pin=25
```

The important information is the Kernel event device (**rc2** in this example) and the GPIO number (**25** in this case). Check that you have correctly wired up the IR sensor to match the configuration.

## Testing the IR sensor with ir-keytable doesn't work.

Running **ir-keytable** doesn't show any events when buttons on the remote control are pressed. This may be because the Kernel device is not **rc0** (The default if no **-s** flag specified).

```
sudo ir-keytable -v -t  
{No events are seen}
```

It is important the **-s** flag is used specifying the correct Kernel events driver (**rc2** in this example).

```
$ sudo ir-keytable -s rc2 -t
```

```
Testing events. Please, press CTRL-C to abort.  
26881.952032: lirc protocol(necxx): scancode = 0xbff05  
26881.952062: event type EV_MSC(0x04): scancode = 0xbff05  
26881.952062: event type EV_KEY(0x01) key_down: KEY_CHANNELUP(0x0192)  
26881.952062: event type EV_SYN(0x00).  
26882.012036: lirc protocol(necxx): scancode = 0xbff05 repeat  
26882.012061: event type EV_MSC(0x04): scancode = 0xbff05  
26882.012061: event type EV_SYN(0x00).  
26882.188029: event type EV_KEY(0x01) key_up: KEY_CHANNELUP(0x0192)  
26882.188029: event type EV_SYN(0x00).  
:
```

Are you using the correct kernel device, rc0, rc1 or rc2?

First make sure that the following statement has been added to **/boot/firmware/config.txt** by the radio configuration program. If not add it to the end of **/boot/firmware/config.txt**.

```
dtoverlay gpio-ir,gpio_pin=9
```

Reboot and retest.

```
$ sudo reboot
```

### Remote control software does not start up

Check that there are no problems with the `remote_control.py` program (Called by service `ireventd`)

```
$ cd /usr/share/radio/
```

```
$ sudo ./ireventd.py nodaemon  
IR Remote control listener running pid 2305  
Using IR kernel events architecture  
Flashing LED on GPIO 16  
Read myremote table  
Old keytable cleared  
Wrote 21 keycode(s) to driver  
Protocols changed to nec  
/dev/input/event2 vc4-hdmi-1 vc4-hdmi-1/input0  
/dev/input/event1 vc4-hdmi-0 vc4-hdmi-0/input0  
/dev/input/event0 gpio_ir_recv gpio_ir_recv/input0  
Listening for IR events:
```

Make a note of the pid (in this example it is 2305). Operate the volume up and down. The following should be displayed:

```
KEY_VOLUMEUP  
KEY_VOLUMEDOWN
```

To stop the program run using the previously noted pid (2305).

```
$ sudo kill -9 2305
```

If the **ireventd.py** program is working OK make sure that the **irevent** service is enabled to start at boot time. Enable using **systemctl**.

```
$ sudo systemctl enable ireventd
```

Reboot the Raspberry Pi

```
$ sudo reboot
```

Check that the UDP server program is listening on port 5100

```
$ netstat -an | grep 5100
udp 0 0 127.0.0.1:5100 0.0.0.0:*
```

If not running stop the radio and check the UDP server software

```
$ cd /usr/share/radio/
$ sudo ./udp_server_class.py
Starting UDP server on port 5100
```

From another terminal run the following command

```
$ sudo ./ireventd.py send TEST
OK
```

On the server side you should see

```
Data = TEST
```

Stop the **udp\_server.py** program with Ctrl-C. Restart the radio and retry.

### Cannot select source through the Web interface

Make sure that the **radiod** is running or the **vgradio** or **gradio** programs on the desktop.

```
$ sudo systemctl status radiod
```

Run the checks or the UDP server as shown in the previous server. Run the UDP server software. Select a source from the Web interface. In the following example the Radio playlist has been selected.

```
$ sudo ./udp_server_class.py
Starting UDP server on port 5100
Data = PLAYLIST:_Radio
```

## Web interface problems

The Web Interface consists of a number of components any, one of which can cause problems.

These components are:

1. **Apache2** Web Server
2. **PHP** (Hypertext Preprocessor) Web page programming language
3. **MySQL** Database
4. **O!MPD** Web Client for accessing O!MPD
5. Communication between the Web Interface and the **radiod** program via **UDP**

### Web page not being displayed

If the initial web page is not being displayed, log into the RPi to check the **apache2.service**. Run the command **systemctl status apache2**. The following display should be displayed. If not, there should be the appropriate error message as to why the service isn't running.

```
$ systemctl status apache2
● apache2.service - The Apache HTTP Server
  Loaded: loaded (/lib/systemd/system/apache2.service; enabled; preset: enabled)
  Active: active (running) since Wed 2023-11-29 07:32:21 GMT; 3h 18min ago
    Docs: https://httpd.apache.org/docs/2.4/
   Process: 5320 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
   Main PID: 5333 (apache2)
     Tasks: 6 (limit: 4384)
       CPU: 506ms
      CGroup: /system.slice/apache2.service
              └─5333 /usr/sbin/apache2 -k start
                  ├─5339 /usr/sbin/apache2 -k start
                  ├─5340 /usr/sbin/apache2 -k start
                  ├─5341 /usr/sbin/apache2 -k start
                  ├─5342 /usr/sbin/apache2 -k start
                  └─5344 /usr/sbin/apache2 -k start

Nov 29 07:32:21 raspberry5 systemd[1]: Starting apache2.service - The Apache HTTP Server...
Nov 29 07:32:21 raspberry5 systemd[1]: Started apache2.service - The Apache HTTP Server.
```

A further check can be done with the following command:

```
$ wget localhost
:
Saving to: 'index.html'

index.html          100%[=====] 3.95K
--.-KB/s   in 0s

2023-11-29 10:17:33 (439 MB/s) - 'index.html' saved [4049/4049]
```

The command should get the first index page from the server.

### Cannot determine hostname

The following error message may appear:

```
apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.1.1. Set the 'ServerName' directive globally to suppress this message
```

The message may be ignored or it can be suppressed by editing the **/etc/apache2/apache2.conf** file and adding a **ServerName** directive.

Edit **/etc/apache2/apache2.conf** file.

Add the following line anywhere in the **apache2.conf** file.

```
ServerName localhost
```

### Cannot access web page from PC

Check the Apache2 access log

**tail -f /var/log/apache2/access.log**

Connect to the Web Server using your PC browser. You should see something like:

```
192.168.1.200 - - [28/Nov/2023:18:55:48 +0000] "GET / HTTP/1.1" 200 2050 "-"  
"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like  
Gecko) Chrome/119.0.0.0 Safari/537.36 Edg/119.0.0.0"
```

If you don't see the above check that the Apache2 is listening on port 80 on all interfaces. Run the following command.

**netstat -tan | grep LISTEN**

tcp	0	0 0.0.0.0:22	0.0.0.0:*	LISTEN
tcp	0	0 127.0.0.1:631	0.0.0.0:*	LISTEN
tcp	0	0 0.0.0.0:5900	0.0.0.0:*	LISTEN
tcp	0	0 127.0.0.1:3306	0.0.0.0:*	LISTEN
tcp6	0	0 :::22	:::*	LISTEN
tcp6	0	0 :::80	:::*	LISTEN
tcp6	0	0 :::6600	:::*	LISTEN
tcp6	0	0 ::1:631	:::*	LISTEN

It may be that your PC isn't configured for IPv6 connectivity.

The interesting ports are ports 22 (SSH) and 80 (HTTP). Port 22 can be seen listening on all IP4 (tcp) interfaces (0.0.0.0). However port 80 is listening is only listening on IPv6 (tcp6 :::80) and not IPv4.

This can be corrected with the following. Edit the **/etc/apache2/ports.conf** file.

```
$ sudo vi /etc/apache2/ports.conf
```

Change the first Listen directive to the following.

```
#Listen 80  
Listen 0.0.0.0:80
```

Now when you check the listening ports you see it now listens to port 80 on all interfaces.

```
netstat -tan | grep LISTEN
tcp        0      0 0.0.0.0:22          0.0.0.0:*          LISTEN
tcp        0      0 0.0.0.0:80          0.0.0.0:*          LISTEN
tcp        0      0 127.0.0.1:631       0.0.0.0:*          LISTEN
tcp        0      0 0.0.0.0:5900       0.0.0.0:*          LISTEN
tcp        0      0 127.0.0.1:3306       0.0.0.0:*          LISTEN
tcp6       0      0 :::22             :::*              LISTEN
tcp6       0      0 :::6600            :::*              LISTEN
tcp6       0      0 :::1:631           :::*              LISTEN
```

Try connecting again.

### O!MPD (Radio page) fails with CURL error

When clicking on the Radio page of the Web interface the following is displayed by O!MPD.

```
-----
ERROR
-----
CURL not loaded
Compile PHP with CURL support.
Or use a loadable module in the php.ini
-----
```

The solution is to install Curl and then refresh the Radio page.

```
$ sudo apt install php8.2-curl
```

### Restart Apache2

```
$ sudo apachectl restart
```

### Failed to connect to the MySql server

The following may be seen:

```
-----
ERROR
-----
Failed to connect to MySQL server on:
127.0.0.1
-----
```

Make sure that you carried out the procedure **Error! Reference source not found.** on page **Error! Bookmark not defined..**

Edit **/var/www/html/ompd/include/config.inc.php** and enter the MySql username and password.

```
// +-----+
// | MySQL configuration
// +-----+-----+
```

```
$cfg['mysqli_host'] = '127.0.0.1';
$cdf['mysqli_db'] = 'ompd';
$cdf['mysqli_user'] = 'pi';
$cdf['mysqli_password'] = 'raspberry';
$cdf['mysqli_port'] = '3306';
```

Restart the system:

```
$ sudo reboot
```

## Bluetooth device connection problems

### Setting “scan on” in bluetoothctl fails 1

The following message is seen when setting “scan on”

```
[bluetooth]# scan on
Failed to start discovery: org.bluez.Error.NotReady
```

To correct problem run “power on”

```
[bluetooth]# power on
changing power on succeeded
```

“scan on” should now work.

### Setting “scan on” in bluetoothctl fails 2

If you see the following message:

```
$ bluetoothctl
[bluetooth]# scan on
No default controller available
```

Run the following:

```
$ sudo hciconfig scan
hci0: Type: Primary Bus: UART
      BD Address: DC:A6:32:E0:50:69 ACL MTU: 1021:8 SCO MTU: 64:1
      UP RUNNING
      RX bytes:77809 acl:38 sco:0 events:9464 errors:0
      TX bytes:11743815 acl:18667 sco:0 commands:122 errors:0
```

If instead of ‘UP RUNNING’ you see ‘DOWN’ check the **hciuart** service.

```
$ systemctl status hciuart.service
● hciuart.service - Configure Bluetooth Modems connected by UART
   Loaded: loaded (/lib/systemd/system/hciuart.service; enabled; vendor
   preset: enabled)
     Active: active (running) since Wed 2021-10-13 22:17:07 BST; 2 days ago
```

```

Main PID: 424 (hciattach)
Tasks: 1 (limit: 2059)
CGroup: /system.slice/hciuart.service
└─424 /usr/bin/hciattach /dev/serial1 bcm43xx 3000000 flow -
b8:27:eb:73:99:d5

Oct 13 22:17:06 pitouch systemd[1]: Starting Configure Bluetooth Modems
connected by UART...
Oct 13 22:17:07 pitouch btuart[341]: Cannot open directory '/etc/firmware':
No such file or director
Oct 13 22:17:07 pitouch btuart[341]: Patch not found, continue anyway
Oct 13 22:17:07 pitouch btuart[341]: bcm43xx_init
Oct 13 22:17:07 pitouch btuart[341]: Set BDADDR UART: b8:27:eb:73:99:d5
Oct 13 22:17:07 pitouch btuart[341]: Set Controller UART speed to 3000000
bits/s
Oct 13 22:17:07 pitouch btuart[341]: Device setup complete
Oct 13 22:17:07 pitouch systemd[1]: Started Configure Bluetooth Modems
connected by UART.
First switch on your Bluetooth device, then switch on scanning. In the
following example this is JVC SP-AD70-B Bluetooth speakers.

```

In the above example it would appear that hciuart cannot find the firmware patches. Check to see if it is present. **Needs mlocate package to be installed.**

```
$ locate BCM4345C0.hcd
/usr/lib/firmware/brcm/BCM4345C0.hcd
```

Now link the above **/usr/lib/firmware** directory to **/etc/firmware**.

```
$ cd /etc/
$ sudo ln -s /usr/lib/firmware
```

Reboot the Raspberry Pi and retry running.

## Using the diagnostic programs

The diagnostic code for testing various components of the radio is contained in the various class code files themselves. For example, **lcd\_class.py** (Test LCD display). First stop the radio and then run the relevant class code.

The classes that contain diagnostic code are as follows:

- **lcd\_class.py** Test the LCD screen (Directly wired to the GPIO)
- **lcd\_adafruit\_class.py** Test the Adafruit LCD plate and buttons
- **lcd\_i2c\_adafruit.py** Test the Adafruit I2C backpack
- **lcd\_i2c\_pcf8574.py** Test LCD with a PCF8574 I2C backpack
- **lcd\_i2c\_jhd1313.py** Test Grove JHD1313 LCD RGB I2C
- **lcd\_piface\_class.py** Test PiFace CAD display and buttons
- **button\_class.py** Test push button switches (Directly wired to the GPIO)
- **rotary\_class.py** Test rotary encoders (Directly wired to the GPIO)
- **rotary\_class\_alternative.py** Test rotary encoders using alternative driver
- **rotary\_class\_rgb\_i2c.py** Test RGB I2C Rotary Encoders

A number of other programs are supplied and can also be used for diagnostics.

- **display\_model.py** Display Raspberry Pi model information
- **display\_current.py** Display current station or track details
- **wiring.py** Display wiring as configured in **/etc/radiod.conf**
- **test\_gpios.py** Test state of all GPIOs and connected buttons or rotary encoders
- **display\_config.sh** Display the current configuration.
- **display\_wifi.sh** Display current Wi-Fi connection details

All diagnostic programs are supplied in the **/usr/share/radio** directory. Change to this directory first!

```
$ cd /usr/share/radio
```

All programs require the **./** characters in front of the name to execute unless called by their full pathname. For example:

```
$ cd /usr/share/radio
$ ./display_wifi.sh
```

This will display a screen similar to the following:

```
Wi-Fi network information
-----
Hostname: piradio
Wi-FI: wlan0 ESSID:"EE-Hub-4qM4"
IP address wlan0: 192.168.1.251
wlan0 Protocol Name:"IEEE 802.11"
Current Frequency:2.462 GHz (Channel 11)
Current Tx-Power=31 dBm (1258 mW)
Bit Rate=72.2 Mb/s Tx-Power=31 dBm
Link Quality=69/70 Signal level=-41 dBm
Power save: on
0: phy0: Wireless LAN
:
:
```

### Running the radio program in diagnostic mode

This is one of the first things you should learn to do. Running the **radiod.py** program in **nodaemon** mode will display any errors it encounters. Use Ctrl-C to exit the program.

```
$ sudo ./radiod.py nodaemon
```

### Using the LCD test code

```
$ ./lcd_class.py
```

The above program will display the following text on the LCD:

**Bob Rathbone**  
**Line 2: abcdefghi**

Line 2 scrolls the alphabet followed by 0 through 9.

The **lcd\_adafruit\_lcd.py** program does the same except it also prints a message on the console screen when a button is pressed.

## Testing push buttons program

Test push buttons directly wired to the GPIO pins.

```
$ ./button_class.py
down_switch
up_switch
right_switch
left_switch
menu_switch
Pull Up/Down resistors DOWN
Button pressed on GPIO 17
:
```

Pressing the switches should show on the screen as shown in the above example.

## Testing rotary encoders

This program does a simple test of the rotary encoders.

```
$ ./rotary_class.py
Test rotary encoder Class
Left switch GPIO 14
Right switch GPIO 15
Up switch GPIO 24
Down switch GPIO 23
Mute switch GPIO 4
Menu switch GPIO 17
Tuner event 1 CLOCKWISE
Tuner event 2 ANTICLOCKWISE
Volume event 1 CLOCKWISE
Volume event 2 ANTICLOCKWISE
Tuner event 4 BUTTON_UP
Tuner event 3 BUTTON_DOWN
Volume event 4 BUTTON_UP
Volume event 3 BUTTON_DOWN
Volume event 1 CLOCKWISE
Volume event 2 ANTICLOCKWISE
```

You can also try

```
$ ./rotary_class_alternative.py
```

## Testing RGB I2C rotarybencoders

Run the `rotary_class_rgb_i2c.py` program

```
$ ./rotary_class_rgb_i2c.py
You must specify at least one encoder to be tested!

Usage: sudo ./rotary_class_rgb_i2c.py --help
        --test_volume
        --test_channel
        --test_both
        --volume_i2c=<volume_i2c_address>
        --channel_i2c=<channel_i2c_address>

Recommended values for volume and channel I2C addresses are 0x0F and 0x1F
(defaults)
Run "i2cdetect -y 1" to check I2C address
```

You can test either one of the rotary encoders individually or both together. For example:

```
$ sudo ./rotary_class_rgb_i2c.py --test_both
RGB I2C Dual Rotary Encoder Test
Volume rotary encoder I2C address=0xf
Mute switch GPIO 27
Channel rotary encoder I2C address=0x1f
Menu switch GPIO 17
Started
Channel rotary encoder event ANTICLOCKWISE
Channel rotary encoder event ANTICLOCKWISE
Channel rotary encoder event CLOCKWISE
Channel rotary encoder event CLOCKWISE
Channel rotary encoder event BUTTONDOWN
Volume rotary encoder event CLOCKWISE
Volume rotary encoder event CLOCKWISE
Volume rotary encoder event CLOCKWISE
Volume rotary encoder event ANTICLOCKWISE
Volume rotary encoder event ANTICLOCKWISE
Volume rotary encoder event BUTTONDOWN
```

The RGB I2C Hex addresses are configured in [/etc/radiod.conf](#).

```
# I2C addresses for RGB I2C Rotary Encoders
volume_rgb_i2c=0x0F
channel_rgb_i2c=0x1F
```

However, the encoders themselves are delivered with the default I2C hex address of **0x0F**. One of the encoders must be set to hex 0x1F (Channel Rotary Encoder) using the [rgb\\_set\\_i2c\\_address.py](#) program.

```
$ ./rgb_set_i2c_address.py --new_i2c_address=0x1F
```

### The [display\\_current](#) program

This is a useful diagnostic that prints out the raw information available from the MPD daemon. The radio daemon uses the same libraries as this test program.

```
$ ./display_current.py

file: http://airspectrum.cdnstream1.com:8116/1649_192#Easy hits Florida HD
title: Lou Rawls - You'll Never Find Another Love Like Mine
name: Easy Hits Florida
time: 0
duration: 0.000
pos: 69
id: 181
current_id 70

Status
volume: 35
repeat: 0
random: 0
single: 0
consume: 0
partition: default
playlist: 20
playlistlength: 111
mixrampdb: 0.000000
```

```

state: play
song: 69
songid: 181
time: 3002:0
elapsed: 3001.688
bitrate: 192
duration: 0.000
audio: 44100:24:2
nextsong: 70
nextsongid: 182

uptime: 3002
playtime: 3002
artists: 0
albums: 0
songs: 0
db_playtime: 0
db_update: 1644564447
Bit rate 192

```

To find out the exact meaning of all these fields please refer to the standard **python-mdp** documentation at <https://pypi.python.org/pypi/python-mdp/> or <http://pythonhosted.org/python-mdp2/topics/getting-started.html>.

### The wiring program

The **wiring.py** program displays a wiring list based upon the configuration that it finds in **/etc/radiod.conf**. In the following example the 40-pin wiring (See **Error! Reference source not found.** on page **Error! Bookmark not defined.**) has been selected during installation. The first column shows the GPIO setting in **/etc/radiod.conf**. The second column (Pin) shows the physical pin for that GPIO. For example, in the **left\_switch** parameter in **/etc/radiod.conf** has been set to GPIO 23 which is physical pin 16 on the 40-pin GPIO header.

The program displays three wiring sections namely:

1. SWITCHES – Rotary encoder and push button wiring
2. LCD – Directly connected LCD wiring
3. OTHER – Remote control and activity LED, I2C backpacks

```

$ cd /usr/share/radio
$ sudo ./wiring.py
Radio wiring scheme based upon configuration in /etc/radiod.conf

----- SWITCHES -----
GPIO  Pin      Switch      Rotary
====  ===  =====
23    16 <----> Left switch   A
24    18 <----> Right switch  B
      6 <----> Gnd 0v       C
4     7 <----> Mute switch   < GND 0V
14   8 <----> Down switch   A
15  10 <----> Up switch    B
      6 <----> Gnd 0v       C
17  11 <----> Menu switch   < GND 0V

Pull Up/Down resistors DOWN

Push button switches must be wired to +3.3V
Rotary push switches must always be wired to GND 0V

----- LCD -----

```

GPIO	Pin	Function	LCD pin
====	==	=====	=====
5	29 <----->	Lcd data4	11
6	31 <----->	Lcd data5	12
12	32 <----->	Lcd data6	13
13	33 <----->	Lcd data7	14
8	24 <----->	Lcd enable	6
7	26 <----->	Lcd select	4
	2 <----->	VCC +5V	2, 15
	6 <----->	GND 0V	1, 16
10K Pot	<----->	Contrast	3
 ----- OTHER -----			
GPIO	Pin	Function	
====	==	=====	
16	36 <----->	Remote led	
3	5 <----->	I2C Data	
2	3 <----->	I2C Clock	
25	22 <----->	IR Remote	(See /boot/firmware/config.txt)

Currently the wiring for the vintage radio RGB Led and Menu switch are not shown.

In the above output the connections are descriptive. The parameters found in **/etc/radiod.conf** can be displayed with the **-p** option:

```
$ ./wiring.py -p
Radio wiring scheme based upon configuration in /etc/radiod.conf

----- SWITCHES -----
GPIO  Pin      Switch      Rotary
====  ==       =====      =====
23    16 <-----> left_switch   A
24    18 <-----> right_switch  B
      6 <-----> GND_0V        C
4     7 <-----> mute_switch    < GND 0V
14   8 <-----> down_switch   A
15  10 <-----> up_switch     B
      6 <-----> GND_0V        C
17  11 <-----> menu_switch   < GND 0V
:
```

In the above display the **left\_switch** label is the parameter found in **/etc/radiod.conf**.

If the wiring shown in the **wiring.py** program then either amend the wiring to match the wiring list shown in the output of the wiring program or amend **/etc/radiod.conf** to match the actual wiring. The program also has a help function (**-h** option).

```
$ ./wiring.py -h
Usage: ./wiring.py -p -h
Where -p print parameters, -h this help text
```

### The **test\_gpios** program

Sometime you may not be sure how you have wired a particular button or rotary encoder or you may have a conflict with another program or a setting in the **/boot/firmware/config.txt** configuration file. The **test\_gpios.py** program allows both situations to be tested. Running the program with no parameters display the help message.

```
$ ./test_gpios.py
Usage: ./test_gpios.py <--help> <--pull_up> <--pull_down> <--none>
Where --help this help text
    --pull_up Set pull-up-down internal resistors high (+3.3V)
    --pull_down Set pull-up-down internal resistors low (0V GND)
    --pull_off No pull-up-down internal resistors (Use external resistors)
```

The other options set the internal pull up/down resistors to UP, DOWN or OFF.

**--pull\_up** Is the normal parameter for most situations. Use this option if unsure.

**--pull\_down** should be used for older radios with push buttons wired to 0V GND.

**--pull\_off** is used mainly for KY-040 rotary encoders which have their own pull-up resistors.

The following example is for a radio with two normal rotary encoders and an IR receiver configured on GPIO 9.

In this case the program reports that it cannot configure GPIO 9 for testing as it is already configured in **/boot/firmware/config.txt** for use by the IR receiver. It attempts to display the conflict.

```
$ ./test_gpios.py --pull_up
GPIO: 2 State:High
GPIO: 3 State:High
GPIO: 4 State:High
GPIO: 5 State:High
GPIO: 6 State:High
GPIO: 7 State:High
GPIO: 8 State:High
GPIO: 9 State:High
Error: GPIO 9 Failed to add edge detection
Check conflict with GPIO 9 in other programs or in /boot/firmware/config.txt
dtoverlay= gpio_ir,gpio_pin=9
GPIO: 10 State:High
GPIO: 11 State:High
GPIO: 12 State:High
GPIO: 13 State:High
GPIO: 14 State:High
GPIO: 15 State:Low
GPIO: 17 State:High
GPIO: 18 State:High
GPIO: 19 State:High
GPIO: 20 State:High
GPIO: 21 State:High
GPIO: 22 State:High
GPIO: 23 State:High
GPIO: 24 State:High
GPIO: 25 State:High
GPIO: 26 State:High
GPIO: 27 State:High
Waiting for input events:
```

Note that GPIO 15 is a rotary encoder input and may be high or low depending upon its position. Same for GPIO 14.

The program waits for events. The Mute button is configured on GPIO 4. The volume rotary encoder A and B inputs are wired to GPIO 14 and 15 respectfully. Pressing the Menu button or turning the volume knob should produce an output similar to the following:

```
GPIO 4 falling
GPIO 4 rising
GPIO 4 falling
GPIO 4 falling
GPIO 11 rising
GPIO 14 falling
GPIO 15 falling
GPIO 14 falling
GPIO 15 rising
```

### The display\_model program

This program displays the Raspberry PI model details.

```
$ ./display_model.py
000e: Model B, Revision 2.0, RAM: 512 MB, Maker: Sony
```

In this example 000e=Manufacturer's revision, B=Model, 2.0=Revision, 512MB RAM, Maker=Sony. If you are unsure of the model or revision of the Raspberry PI use this program to find this out.

### The display configuration program

The **display\_config.sh** provides diagnostic information how the radio, drivers and MPD have been configured. It also produces a compressed tar file called **/usr/share/radio/config.log.tar.gz** with this information. Send this file to [bob@bobrathbone.com](mailto:bob@bobrathbone.com).

```
$ ./display_config.sh
Configuration log for pi Tue Jul  4 20:38:27 BST 2023
Release 24th June 2023 - Build 6
IP address: 192.168.1.251

OS Configuration
-----
PRETTY_NAME="Raspbian GNU/Linux 12 (bookworm)"
NAME="Raspbian GNU/Linux"
VERSION_ID="12"
:
=====
End of run =====
This configuration has been recorded in /usr/share/radio/config.log
A compressed tar file has been saved in /usr/share/radio/config.log.tar.gz
Send /usr/share/radio/config.log.tar.gz to bobrathbone.com if required
```

### The display\_os program

This **display\_os.sh** program display details about the operating system and user details. Although this information is also displayed in the **display\_configuration.sh** program it is a more convenient and compact way to display operating system details. It is purely meant to help diagnosing a problem with the software installation. The following screen is an example run:

```
$ ./display_os.sh
OS configuration log for host bookworm32 Tue Oct  1 10:10:13 BST 2024

OS Configuration
-----
PRETTY_NAME="Raspbian GNU/Linux 12 (bookworm)"
NAME="Raspbian GNU/Linux"
VERSION_ID="12"
VERSION="12 (bookworm)"
```

```
VERSION_CODENAME=bookworm
ID=raspbian
ID_LIKE=debian
HOME_URL="http://www.raspbian.org/"
SUPPORT_URL="http://www.raspbian.org/RaspbianForums"
BUG_REPORT_URL="http://www.raspbian.org/RaspbianBugs"
Debian version 12.7

Boot configuration
-----
/boot/firmware/config.txt
arm_64bit=0
dtoverlay=i2s-mmap
dtoverlay= gpio-ir, gpio_pin=25
dtoverlay=wm8960-soundcard
dtparam=i2c_arm=on
dtparam=i2s=on
dtparam=audio=off

Kernel version
-----
Linux bookworm32 6.6.31+rpt-rpi-v7l #1 SMP Raspbian 1:6.6.31-1+rpt1 (2024-05-29) armv7l GNU/Linux
OS bookworm Architecture 32-bit

Network configuration
-----
IP address: 192.168.1.252
default via 192.168.1.254 dev wlan0 proto dhcp src 192.168.1.252 metric 600
192.168.1.0/24 dev wlan0 proto kernel scope link src 192.168.1.252 metric 600

Wi-Fi network information
-----
Hostname: bookworm32
Wi-FI: wlan0 ESSID:"EE-K7TZ9R"
IP address wlan0: 192.168.1.252
wlan0 Protocol Name:"IEEE 802.11"
Current Frequency:5.5 GHz (Channel 100)
Current Tx-Power=31 dBm (1258 mW)
Bit Rate=433.3 Mb/s Tx-Power=31 dBm
Link Quality=68/70 Signal level=-42 dBm
Power save: on
0: hci0: Bluetooth
    Soft blocked: no
    Hard blocked: no
1: phy0: Wireless LAN
    Soft blocked: no
    Hard blocked: no

User details
-----
User pi, Group: pi
Home directory: /home/pi
uid=1000(pi) gid=1000(pi)
groups=1000(pi),4(adm),20(dialout),24(cdrom),27(sudo),29(audio),44(video),46(plugdev),60(games),100(users),102(input),105(render),106(netdev),995(spi),994(i2c),993(gpio),115(lpadmin)

Desktop installation
-----
X-Windows appears to be installed and is using the Wayland protocol
X-Windows autostart configuration in file /home/pi/.config/wayfire.ini

Radio version
-----
```

```

Version 7.8
Build 7.8.11

Audio Configuration
-----
**** List of PLAYBACK Hardware Devices ****
card 0: wm8960soundcard [wm8960-soundcard], device 0: fe203000.i2s-wm8960-
hifi wm8960-hifi-0 [fe203000.i2s-wm8960-hifi wm8960-hifi-0]
    Subdevices: 0/1
    Subdevice #0: subdevice #0
pulse
    PulseAudio Sound Server

A log of this run will be found in /usr/share/radio/logs/os_config.log
A compressed tar file has been saved in
/usr/share/radio/logs/os_config.log.tar.gz
Send /usr/share/radio/logs/os_config.log.tar.gz to bob@bobrathbone.com if
required

```

### Running the radio program in nodaemon mode

If for some reason the radio program stops or crashes without explanation (particularly if you have modified the code), it can be extremely difficult to see what is happening as the radio software runs as a so-called system daemon.

There is a way to run the software in foreground mode. In this case stop the radio and change to **/usr/share/radio** directory and run the radio program with the **nodaemon** option.

```

$ cd /usr/share/radio
$ sudo ./radiod.py nodaemon

```

If the program crashes it will display a stack trace which will give the file name and line numbers where the program crashed. Use **Control-C** to exit **nodaemon** mode (This will also display a stack trace which is normal and is not an error).

In the case of **gradio.py** and **vgradio.py**, these are not daemons and can be run as shown in the example for gradio.py below:

```

$ sudo ./gradio.py

```

### Creating a log file in DEBUG mode

You may be asked to supply a log file in DEBUG mode for support purposes.

Stop the radio and remote control if running:

```

$ sudo systemctl stop radiod
$ sudo systemctl stop irreventd

```

Edit the **/etc/radiod.conf** file and switch on DEBUG mode as shown below.

```

# loglevel is CRITICAL, ERROR, WARNING, INFO, DEBUG or NONE
loglevel=DEBUG

```

Remove the old log file or in **/etc/radiod.conf** change **log\_creation\_mode=truncate**:

```
$ sudo rm /var/log/radiod/radio.log
```

Start the radio and remote control if required:

```
$ sudo systemctl start radiod  
$ sudo systemctl start irreventd
```

Operate the radio including the operation you are having with.

Send the **/var/log/radiod/radio.log** file to [bob@bobrathbone.com](mailto:bob@bobrathbone.com)

Switch off DEBUG mode by editing the **/etc/radiod.conf** file and as shown below.

```
loglevel=INFO
```

## Display the kernel details

Display the kernel version using **uname**.

```
$ uname -a  
Linux raspberrypi 5.10.63-v7+ #1459 SMP Wed Oct 6 16:41:10 BST 2021 armv7l  
GNU/Linux
```

## Displaying information about the Raspberry Pi

There are a number of standard facilities to provide information about the Raspberry Pi which may be useful when diagnosing a problem.

### Displaying the GPIO information

There are two standard programs for reading from and writing to GPIO Pins.

The **gpio readall** command (**Bullseye** only) can be used to display the GPIO configuration. Stop the radio first!

Model B2															
BCM	wPi	Name	Mode	V	Physical	V	Mode	Name	wPi	BCM	Switch				
		3.3v			1	2		5v							
2	8	SDA.1	IN	1	3	4		5V							
3	9	SCL.1	IN	1	5	6		0v							
4	7	GPIO. 7	IN	1	7	8	0	IN	TxD	15	14	Left			
		0v			9	10	0	IN	RxD	16	15	Right			
17	0	GPIO. 0	IN	0	11	12	0	IN	GPIO. 1	1	18	Up/Down			
27	2	GPIO. 2	OUT	0	13	14		0v							
22	3	GPIO. 3	OUT	0	15	16	0	OUT	GPIO. 4	4	23				
		3.3v			17	18	0	OUT	GPIO. 5	5	24				
10	12	MOSI	IN	0	19	20		0v				Down*			
9	13	MISO	IN	0	21	22	0	IN	GPIO. 6	6	25	Menu			
11	14	SCLK	IN	0	23	24	0	OUT	CEO	10	8				
		0v			25	26	1	OUT	CE1	11	7				
Model B2															

\* Alternative down switch for HiFiBerry DAC+ compatibility.

The physical pins are shown in the center numbered 1 to 26 and 51 through 54 (for a model B2) in this example. The above output is for a radio using a directly wired LCD and push buttons. For example:

Physical pin 8 is BCM GPIO 14 and mode is configured as an input for the left switch and is currently low (Column V is 0).

The second program that can be used is **raspi-gpio**.

```
$ raspi-gpio get
BANK0 (GPIO 0 to 27):
GPIO 0: level=1 fsel=0 func=INPUT
GPIO 1: level=1 fsel=0 func=INPUT
GPIO 2: level=1 fsel=0 func=INPUT
GPIO 3: level=1 fsel=0 func=INPUT
GPIO 4: level=0 fsel=0 func=INPUT
GPIO 5: level=0 fsel=0 func=INPUT
GPIO 6: level=0 fsel=0 func=INPUT
GPIO 7: level=0 fsel=0 func=INPUT
GPIO 8: level=0 fsel=0 func=INPUT
GPIO 9: level=0 fsel=0 func=INPUT
GPIO 10: level=0 fsel=0 func=INPUT
GPIO 11: level=0 fsel=0 func=INPUT
GPIO 12: level=0 fsel=0 func=INPUT
GPIO 13: level=0 fsel=0 func=INPUT
GPIO 14: level=0 fsel=0 func=INPUT
GPIO 15: level=0 fsel=0 func=INPUT
GPIO 16: level=0 fsel=0 func=INPUT
GPIO 17: level=0 fsel=0 func=INPUT
GPIO 18: level=0 fsel=0 func=INPUT
GPIO 19: level=0 fsel=0 func=INPUT
GPIO 20: level=0 fsel=0 func=INPUT
GPIO 21: level=0 fsel=0 func=INPUT
GPIO 22: level=0 fsel=0 func=INPUT
GPIO 23: level=0 fsel=0 func=INPUT
GPIO 24: level=0 fsel=0 func=INPUT
GPIO 25: level=1 fsel=1 func=OUTPUT
GPIO 26: level=0 fsel=0 func=INPUT
GPIO 27: level=0 fsel=0 func=INPUT
BANK1 (GPIO 28 to 45):
GPIO 28: level=1 fsel=0 func=INPUT
GPIO 29: level=1 fsel=0 func=INPUT
GPIO 30: level=0 fsel=7 alt=3 func=CTS0
:
```

Run the following for a full help menu

```
$ raspi-gpio help
```

Examples:

raspi-gpio get	Prints state of all GPIOs one per line
raspi-gpio get 20	Prints state of GPIO20
raspi-gpio get 20,21	Prints state of GPIO20 and GPIO21
raspi-gpio set 20 a5	Set GPIO20 to ALT5 function (GPCLK0)
raspi-gpio set 20 pu	Enable GPIO20 ~50k in-pad pull up
raspi-gpio set 20 pd	Enable GPIO20 ~50k in-pad pull down
raspi-gpio set 20 op	Set GPIO20 to be an output

```

raspi-gpio set 20 dl      Set GPIO20 to output low/zero (must already be
set as an output)
raspi-gpio set 20 ip pd    Set GPIO20 to input with pull down
raspi-gpio set 35 a0 pu    Set GPIO35 to ALT0 function (SPI_CE1_N) with
pull up
raspi-gpio set 20 op pn dh Set GPIO20 to ouput with no pull and driving
high

```

## Displaying information about the Operating system

Display **/etc/os.release** using the **cat** command.

### Bookworm 32-bit

```

$ cat /etc/os-release
PRETTY_NAME="Raspbian GNU/Linux 12 (bookworm)"
NAME="Raspbian GNU/Linux"
VERSION_ID="12"
VERSION="12 (bookworm)"
VERSION_CODENAME=bookworm
ID=raspbian
ID_LIKE=debian
HOME_URL="http://www.raspbian.org/"
SUPPORT_URL="http://www.raspbian.org/RaspbianForums"
BUG_REPORT_URL="http://www.raspbian.org/RaspbianBugs"

```

The results are slightly different between 32 and 64-Bit systems.

### Bookworm 64-bit

```

$ cat /etc/os-release
PRETTY_NAME="Debian GNU/Linux 12 (bookworm)"
NAME="Debian GNU/Linux"
VERSION_ID="12"
VERSION="12 (bookworm)"
VERSION_CODENAME=bookworm
ID=debian
HOME_URL="https://www.debian.org/"
SUPPORT_URL="https://www.debian.org/support"
BUG_REPORT_URL=https://bugs.debian.org/

```

64-Bit systems state that the system is **Debian** instead of **Raspbian**

Likewise, the Debian version can be displayed for both 32 and 64-Bit systems.

```

$ cat /etc/debian_version
12.7

```

The architecture (32 or 64-Bit) can be displayed with the following instruction:

```

$ getconf LONG_BIT
64

```

# Chapter 9 - Source files

## Introduction to the source code

This section is only of interest if you are considering developing your own version of the software or wish to use one of the classes in your own software.

The source consists of several source modules all written in **Python 3** using Object Orientated techniques. The source will be visible in the **/usr/share/radio** directory once the Radio package has been installed. The radio Debian package is available at  
[http://www.bobrathbone.com/pi\\_radio\\_source.htm](http://www.bobrathbone.com/pi_radio_source.htm).

For those who want to develop their own product all source is also available from Github. See [Error! Reference source not found.](#) on [Error! Bookmark not defined..](#)

## The Radio program

The *radiod.py* program is the top-level radio program for the LCD versions of the radio and provides the logic for operating the radio. It is called from the **systemd radiod.service** script in the **/lib/systemd/system** directory.

## The Radio Daemon

The *radio\_daemon.py* code allows the LCD radio program to run as a background daemon. It allows start, stop, restart, version and status commands.

## The Display Class

The *display\_class.py* is the common top-level display used by the radio programs (radiod, gradio and vgradio) and is responsible for displaying messages on the various types of display. It uses the **display\_type** parameter in the **/etc/radiod.conf** configuration file to load the correct software for the display being used. Depending upon the actual device configured it will load one of the following display drivers:

**Table 9** Display classes

Display class file	Description	display_type
<b>lcd_class.py</b>	LCDs with a directly connected HD44780U	LCD
<b>lcd_i2c_adafruit.py</b>	LCDs with an Adafruit I2C backpack	LCD_I2C_ADAFRUIT
<b>lcd_i2c_pcf8574.py</b>	LCDs with a PCF8574 I2C backpack	LCD_I2C_PCF8574
<b>lcd_i2c_jhd1313.py</b>	Grove JHD1313 LCD RGB with I2C interface	LCD_I2C_JHD1313
<b>lcd_adafruit_class.py</b>	LCDs with an Adafruit RGB plate	LCD_I2C_ADAFRUIT
<b>lcd_adafruit_class.py</b>	Adafruit LCD I2C RGB plate with buttons	LCD_ADAFRUIT_RGB
<b>lcd_pifacecad_class.py</b>	PiFace CAD 2x16 LCD with push buttons	PIFACE_CAD
<b>oled_class.py</b>	Solomon Systech SSD1306 OLED Display - IQaudio	OLED_128x64
<b>st7789tft_class.py</b>	Pimoroni Pirate audio with four push buttons	ST7789TFT
<b>ssd1306_class.py</b>	Sitronix SSD1306 128x64 pixel OLED	SSD1306
<b>no_display.py</b>	No display attached (vintage radio/Pirate Radio)	NO_DISPLAY
<b>luma_class.py</b>	LUMA driver for OLEDs with SSD1306, SSD1309, SSD1325, SSD1331, SH1106 and WS0010 chips	LUMA.<DEVICE>

The above settings are either configured by the `configure_radio.sh` program or by modifying the `display_type` parameter directly in `/etc/radiod.conf`. For example, if the PCF8574 I2C backpack is chosen then the following will be configured in `/etc/radiod.conf`:

`display_type=LCD_I2C_PCF8574`

## The Graphical Screen radio programs

The `gradio.py` and `vgradio.py` programs are the radio programs for the HDMI/Touchscreen version of the radio and is launched on the graphical desktop of the Raspberry Pi. It is optionally called from the `Desktop/gradio.desktop` script in the `/home/pi` directory. The `vgradio.py` program gives a vintage radio look and feel to the radio display.

In the case of both programs the `display_type` parameter in `/etc/radiod.conf` is set to GRAPHICAL and is set by the `configure_radio.sh` program.

## The Graphics display class

The `graphic_display.py` class performs auxiliary display functions such as scrolling and screen mapping for the `gradio.py` and `vgradio.py` programs.

## The Graphics controls class

The `gcontrols_class.py` handles the creation of all graphics controls and widgets for the graphic version of the radio. It also uses the SGC widget routines in the `sgc` sub-directory from Sam Bull and Michael Rochester.

## The OLED class

The `cosmic_class.py` is the display interface for the SSD1306 128x64 pixel OLED display supplied with the IQaudIO Cosmic controller. This class is a wrapper for the routines from **Olimex Limited** in the sub-directory `oled`. It drives the OLED screen although not all functions are used by the radio.

## The SH1106 OLED SPI interface (pHat with joystick)

The SH1106 1.3"128x64 pixel SPI interface uses the following three files:

- `sh1106_class.py` Driver for the 1.3" SH1106 OLED (SPI interface)
- `sh1106_config.py` Configuration file for the above driver
- `sh1106_key_test.py` Graphical test program for the pHat with OLED display, joystick and 3-push buttons

It is also necessary to physically re-wire the OLED pHat (with joystick and push buttons) to use the I2C interface. See Appendix **Error! Reference source not found.** on page **Error! Bookmark not defined.** for more information.

## The button class

The `button_class.py` detects all button presses from the push button radios (Not the Adafruit RGB nor the PiFace CAD which have their own buttons using I2C and SPI interfaces respectively). It passes button press events up to the event class described later.

## The rotary class

The `rotary_class.py` and `rotary_class_alternative.py` detect all rotary encoder events from the radios fitted with rotary encoders. It passes rotary encoder events up to the event class described later.

## The Cosmic controller Class

The *cosmic\_class.py* is used as the user interface for the IQaudIO Cosmic controller. This provides the interface for three-button and rotary control interface on the controller board.

## The Event class

All user interfaces in the radio software generate a largely common set of events. These are handled by the *event\_class.py*. The *event\_class.py* program accepts events from the following sources:

- The *gradio.py* and *vgradio.py* graphical radio programs
- The *radiod.py* radio program
- The push button interface user interface (*button\_class.py*)
- The rotary encoder user interface (Either *rotary\_class.py* or *rotary\_class\_alternative.py*)
- The IR remote control user interface (*ireventd.py*)
- The radio Web user interface running on an Apache Web server

## The Menu class

The *menu\_class.py* code provides the logic for stepping through the various menus and their options.

## The Message class

All messages are generated from the *message\_class.py* program. This uses message labels to load the correct text to be displayed or spoken. By using labels and the *language\_class.py* software, the radio can be configured to use any language using a Latin character set. It provides messages to display various menu's, time, station and track information.

## The Language class

The *language\_class.py* provides the text for both the radio display or the **espeak** package. It reads the **/var/lib/radiod/language** file (if present) and passes the text to both the message class and if used. It is used by the message class to deliver messages in the user's own language.

## The Log class

The *log\_class.py* routine provides logging of events to **/var/log/radiod/radio.log** file.

## The Volume class

The *volume\_class.py* program handles all volume and mixer functions for the radio, Spotify and airplay.

## The Configuration Class

The **config\_class.py** reads and stores the radio configuration from the **/etc/radiod.conf** file

## The RSS class

The *rss\_class.py* routines allow sequential gets from an RSS feed. These feeds are provided from news providers such as the BBC. This class gets the RSS feed defined in the **/var/lib/radiod/rss** file.

## The Translate class

The *translate\_class.py* is used to convert special international character sets (particularly from RSS feeds). It does this by first converting them to escape sequences and then to displayable ascii characters (These will show up in DEBUG logging). These *ascii* characters are then passed to the LCD class where they will be converted again to a valid character in the standard LCD character set.

## The `create_stations` program

The `create_stations.py` program creates playlist files in the `/var/lib/mpd/playlists` directory using a list of Web links (URLs) with titles as input. This program creates standard playlists for use with MPD. The operation of the `create_stations.py` program is covered in detail in the section on managing playlist files on page 169.

## The `update_stationlist` program

The `update_stationlist.py` program reads each of the RADIO playlist files in the `/var/lib/mpd/playlists` directory and creates a new `/var/lib/radiod/stationlist` file. This is required to add new station URLs dynamically added by any external MPD clients that are capable of doing so. It is run from the `/etc/cron.daily/radiod` crontab script.

## The `display_current` program

The `display_current.py` program is a small diagnostic program which displays the information for the current radio station or track. It is only used for trouble-shooting or for gaining an insight into the sort of information provided by the MPD daemon.

## The `display_model` script

The `display_model.py` program displays the revision, CPU, memory and maker (If known) of the board. It is only used for trouble-shooting and it will not be used in normal operation.

## The `configure_radio.sh` script

The `configure_radio.sh` script is normally called during installation of the Radio Debian package but may be run by the user at any time. It selects the correct board revision and radio program variant. It configures the display to be used and the user interface.

## The `playlist creation` program

The `create_playlist.sh` script creates playlists from music directories on either a USB stick or a Network drive such as a NAS. It has the ability to accept filters to make a more selective playlist.

## The `configure_audio.sh` script

The `configure_audio.sh` script selects and configures the Audio output. It currently supports selection of the on-board audio jack, HDMI output, USB DAC, HiFiBerry and IQaudIO DACs.

## The `configure_audio_device.sh` script

The `configure_audio_device.sh` script is called by the radio program when it starts up. It is used to select the sound card to be used. This is because Raspbian have recently changed the way audio devices are numbered and these can vary if an HDMI device is attached or not.

It configures `/etc/mpd.conf` and `/etc/asound.conf` with the correct device number depending upon the output of the `aplay -l` command. The script reads the output of the `aplay -l` command and compares each line with the setting of the `audio_out` parameter in `/etc/radiod.conf`.

If the above `audio_out` parameter is missing add it to `/etc/radiod.conf`.

The `configure_audio.sh` script sets the `audio_out` parameter when run during setup. Depending on the sound device selected it sets it to **headphones, HDMI, USB or DAC**

To understand this script run the `aplay` command to display available sound cards (The following is an example only).

```
$ aplay -l | grep -i card
card 0: b1 [bcm2835 HDMI 1], device 0: bcm2835 HDMI 1 [bcm2835 HDMI 1]
card 1: Headphones [bcm2835 Headphones], device 0: bcm2835 Headphones [bcm2835
Headphones]
card 2: IQaudIODAC [IQaudIODAC], device 0: IQaudIO DAC HiFi pcm512x-hifi-0 [IQaudIO
DAC HiFi pcm512x-hifi-0]
```

In the above example, three sound cards(devices) are shown. Edit the **/etc/radiod.conf** file and add the following parameter to the **radiod.conf** file.

#### **audio\_out=<device>"**

Where **<device>** is any unique string (when compared with other lines) from the required card definition.

For example:

```
audio_out="Headphones" {Will configure Card 1 On-board audio output jack}
audio_out="IQaudio DAC" {Will configure Card 2 IQaudio DAC}
audio_out="HDMI 1" {Will configure Card 0 HDMI 1}
audio_out="DAC" {Will configure Card 2 IQaudio DAC}
```

#### **The configure\_ir\_remote script**

The **configure\_ir\_remote.sh** script sets up and partially configures the Kernel Event driver components for an IR remote control. It sets up the **ireventd.service**.

#### **The set mixer id script**

The **set\_mixer\_id.sh** script works out the “Speaker Playback Volume” mixer ID and configures the **mixer\_volume\_id** in **/var/lib/radiod** directory. This information comes from the **amixer controls** command. This mixer ID (integer) is used to set a default mixer volume for MPD and also is used for volume control when using Airplay. The **set\_mixer\_id.sh** script is normally called from the radio program (all versions), usually after a reboot, if the **mixer\_volume\_id** parameter has been removed by the **configure\_audio.sh** program. This script also completes configuration of HDMI audio if selected in the **configure\_audio.sh** program. It is not normally necessary to run this program separately but can be safely run at any time.

#### **The UDP network communications class**

The remote control daemon uses the **udp\_server\_class.py** program which communicates over the local TCP/IP network using UDP port 5100 as the default; however, the port is configurable in **/etc/radiod.conf**.

When a button is pressed on the remote control this program sends the button identity (See Appendix B.4 *UDP messages* on page 307) to a UDP server running in the radio program. It is also used to send commands from the Web Interface to the radio program.

Button press → IR remote control daemon → UDP message over network → Radio program.

#### **The Status LED class**

The **status\_led\_class.py** is called by the vintage radio software. A Red Blue Green LED is driven to indicate status of the radio as there is no LCD screen. See the Raspberry Pi Vintage Radio supplement.

## The Airplay Class

The `airplay_class.py` file contains the routines for stopping and starting the **shairport-sync** daemon and for getting artist, title and album of the playing track. It is used when Airplay is selected as the source.

## The Menu Switch class

The `menu_switch_class.py` code supports an 8-position rotary switch (Not encoder) as an alternative method of operating a simple menu system. It is meant to be used with the vintage radio software but can be used with any variant.

## The init file

The `__init__.py` file contains a couple of global definitions plus the package version number.

## Alsa sound files

The Alsa sound configuration is contained in **/etc/asound.conf** file. You may see mention of the **.asoundrc** file in Alsa documentation but it is not used by the radio project. The **.asoundrc** file does exactly the same as **/etc/asound.conf** file but is specific to one user and is placed in their home directory.

Depending upon the device to be configured a different configuration will be used. These asound files are contained in the **/usr/share/radio/asound** directory. The correct file is copied to the **/etc/asound.conf** file by the **configure\_audio.sh** sound configuration script.

**Table 10 Asound configuration files**

Asound file	Description
<b>asound.conf.dist</b>	Standard asound.conf
<b>asound.conf.dist.bonnet</b>	Adafruit speaker bonnet Alsa configuration file
<b>asound.conf.dist.pivumeter</b>	Pimoroni pHat with VU meter configuration file.
<b>asound.conf.dist.blue</b>	Bluetooth devices Alsa configuration file
<b>asound.conf.dist.equalizer</b>	Configuration for the Alsa graphic equalizer
<b>asound.conf.dist.pipe</b>	Bluetooth devices Alsa configuration file using a pipe (unused)
<b>asound.conf.dist.bonnet</b>	Configuration file for use with the speaker bonnet
<b>asound.conf.dist.softvol</b>	Configuration file for devices that don't have hardware volume controls.
<b>asound.conf.dist.wm8960</b>	Configuration for the Waveshare WM8960 sound card

## The IR remote-control ireventd.service

This IR service runs on Bookworm or later. Configuration for the IR remote control is done by the **ir-keytable** utility. This is the service that all new users should be using. The command to start the IR **ireventd.service** is:

```
$ sudo systemctl start ireventd.service
```

To stop the IR radio service run:

```
$ sudo systemctl stop ireventd.service
```

When the Remote-Control buttons are pressed it passes the relevant commands to the radio program. The **ireventd.py** program provides complete control of the radio and can change menu options, do searches etc. just as the same as the knobs or buttons. It normally communicates with the radio program using UDP port 5100 on the local network interface.

## The GPIOconverter program

The **Raspberry Pi Model 5** was introduced at the end of 2023. It only works with **Raspberry Pi Bookworm OS** or later.

However, the biggest impact for most developers is that the standard **RPi.GPIO** input/output library does not work on the **Raspberry Pi model 5**. This is because the **RPi Model 5** now has a separate chip called **RP1** for controlling I/O including the pins on the GPIO header (**j8**). This means that hundreds of thousands of programs or maybe even millions of programs need to be modified to use one of the newer libraries such as **gpiod** or **Igpio**. The **RP1** chip also controls USB ports, Gigabyte Ethernet, MIPI Camera Controllers and Low Speed Peripherals compatible with earlier versions of the Raspberry Pi.

The **Raspberry Pi Internet Radio** is also such a program and would have meant a lot of work to convert all the GPIO routines to say **GPIOD** which does run on the RPi Model 5. So, it was decided to write a simple interface called **GPIOconverter** which converts **RPi GPIO** calls to one of the newer GPIO interfaces. This is a so-called **software shim**. See the following link for more information: [https://en.wikipedia.org/wiki/Shim\\_\(computing\)](https://en.wikipedia.org/wiki/Shim_(computing)). Although **GPIOD** was advocated as being the best way forward it was poorly documented and there didn't seem to be any examples of how to handle interrupts. The choice was made to use the excellent **python3-Igpio** library for the **GPIOconverter** software. The architecture of the interface is shown below:

**OUTPUT: User Program --> GPIO calls --> GPIOconverter --> LGPIO**  
**INPUT: LGPIO events --> GPIOconverter --> User Program**

The following illustration shows the location of the RPP1 I/O chip.

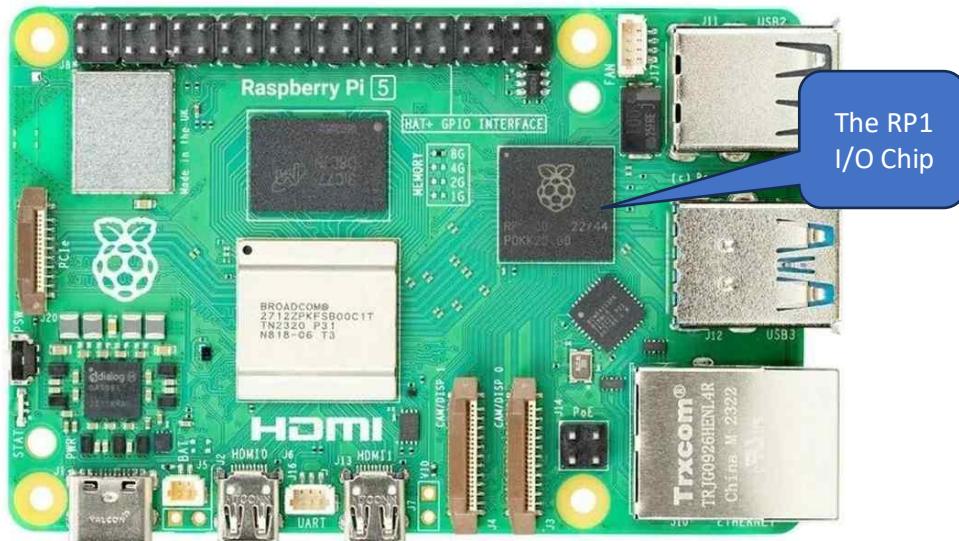


Figure 56 The Raspberry Pi Model 5 RP1 I/O chip

A further problem occurred with Bookworm OS in early 2024 when a new release of the kernel meant that RPi/GPIO no longer works on even earlier models such as the 3B or 4B. So, if installing the radio software on the Bookworm OS the **GPIOconverter** shim is enabled during installation to correct this problem.

The **GPIOconverter** software called GPIO.py is located in the **/usr/share/radio/RPi** directory.

### Enabling GPIO.py

To enable **GPIOconverter** in the case of either a Raspberry Pi model 5 or if running on Bookworm only:

```
touch /usr/share/radio/RPi/__init__.py
```

This creates an empty file called **\_\_init\_\_.py**. The instruction above will cause the code in the **/usr/share/radio/** directory using the GPIO calls to see directory RPi as a package instead of using the system wide GPIO package.

For earlier models such as the 3B or 4 or if running on **Bullseye** disable the package

```
rm /usr/share/radio/RPi/__init__.py
```

### The weather programs

The **weather.py** program has been written for two reasons:

1. To demonstrate how to write your own programs using the display and event classes to use the same hardware as the radio program.
2. To demonstrate how to call your own programs when exiting from the radio program.

**weather.py** - This is the program which displays the current location weather details

**weather2.py** – This program is designed to run on a second screen from the radio screen

**weather\_class.py** – This class actually gets the weather from <https://openweathermap.org>

**/etc/weather.conf** – Configuration details for the weather program and the second screen

**/etc/weather2.conf** - Configuration details for the weather program running standalone

**wxconfig.py** – This is the interface program to the configuration **/etc/weather.conf** file

If running this program on the **Bookworm OS** then run the following instruction:

```
$ sudo mv /usr/lib/python3.11/EXTERNALLY-MANAGED  
/usr/lib/python3.11/EXTERNALLY-MANAGED.old
```

Install the following prerequisite packages

```
$ sudo pip3 install geocoder  
$ sudo pip3 install beautifulsoup4  
$ sudo apt-get install python3-lxml
```

To run the weather program from the radio program replace the **exit\_action** statement in **/etc/radiod.conf** with the following:

```
exit_action=/usr/share/radio/weather.py
```

When the radio program exits, either by pressing the menu button for three seconds or the power button on a remote control, it will call **weather.py** to display the weather details for the location configured in **wxconfig.conf**.

If you want to run the weather2 program on a second screen independent of the radio program then run the following instructions.

```
$ sudo cp weather2.service /usr/lib/systemd/system/.  
$ sudo systemctl enable weather2.service
```

### The `weather.conf` configuration file

The `weather.conf` file contains two sections. The **[WEATHER]** section and the **[DISPLAY]** section.

```
# Location variable to customise weather display  
# Get current weather, For parameters see https://openweathermap.org/current  
# Configuration for weather.py  
  
[WEATHER]  
# Location & language  
city = "Thornbury"  
countrycode = "GB"  
language = "EN"  
  
# Open weathermap Key (Contact OpenWeatherMap.org directly if you need a new  
key)  
api_key="080535da15e1e2a21f933846b0ba824a"  
  
# Temperature units metric(C) or imperial(F)  
# Units standard, metric and imperial units  
units = "metric"  
  
# Atmospheric pressure units millibars(M) or inches mercury(I)  
pressure_units = "M"  
  
# Set date format, US format = %H:%M %m/%d/%Y  
# To display timezone use %Z  dateformat=%H:%M %Z %d/%m/%Y  
date_format = "%H:%M %d/%m/%Y"  
  
# Exit command either "" or the command to execute  
#exit_command = "sudo systemctl restart radiod"  
exit_command = ""  
  
# UDP server for IR remote commands  
udp_host='localhost'  
udp_port=5100  
  
# Display parameters for a secondary display only used by the weather2.py  
program  
# See wxconfig.py for display types  
[DISPLAY]  
display_type=LUMA.SH1106  
i2c_address=0x3c  
display_lines=4  
display_width=16
```

The **API\_KEY** worked at the time of writing and allows access to the interface routines at Open Weather Map. If it doesn't work request a new one from <https://openweathermap.org>

Specify your location using the **CITY**, **COUNTRYCODE** and also the language (**LANG**) you wish to use. **UNITS** are either 'metric' or 'imperial' to display temperature in degrees Centigrade or Farenheight respectively.

Likewise, **PRESSURE\_UNITS** for atmospheric pressure in millibars or inches mercury. The **DATE\_FORMAT** allows the way date and time are displayed.

The above configuration displays the following on a four-line display:

```
10:04 27/08/2023  
London GB  
14.1C 1011.0mb 89%  
Overcast clouds 100%
```

By default, the **exit\_command** restarts the radio program. If not required replace it with:

```
exit_command=""
```

The **[DISPLAY]** section is specifically for the **weather2.py** program and contains the parameters for a secondary screen to display the weather at the same time that the first screen (configured in **/etc/radiod.conf**) displays the radio program.

The next few illustrations show how multiple I2C displays can be connected using a Grove I2C Hub



Figure 57 Radio and weather program running on two separate displays

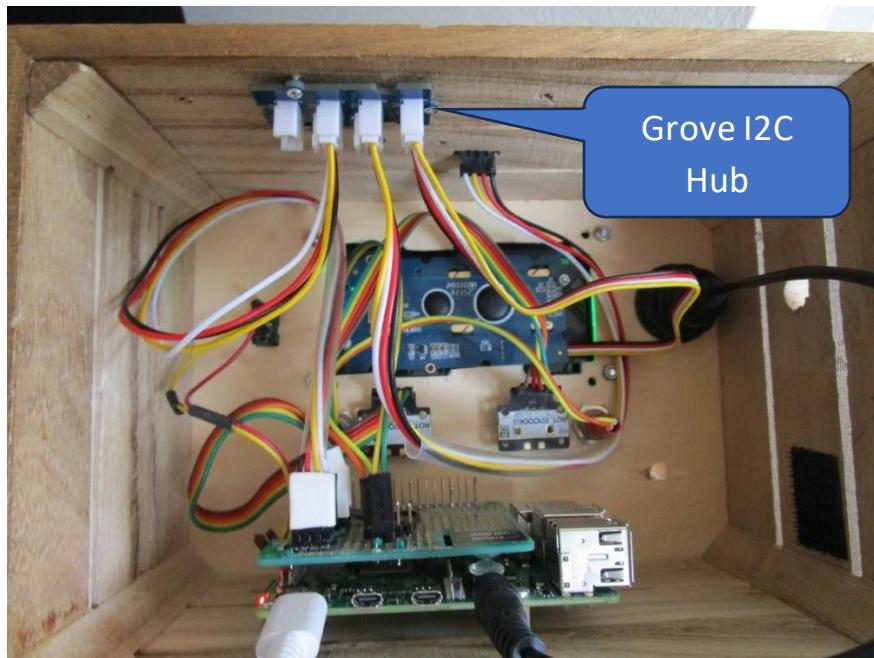
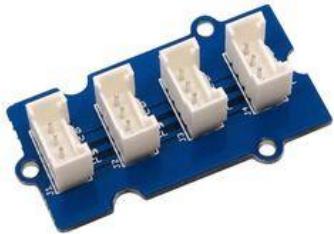


Figure 58 Connecting multiple I2C displays using an I2C hub

Shown below are the **Seeed** (Yes three e's) cables and hub required to connect multiple I2C displays together.



Grove Seeed Hub



Seeed Connector breakout



Seeed connection cables

## Downloading the source from GitHub

This is only of interest if you wish develop your own version of the Raspberry PI radio based upon the mainstream source code. Otherwise simply install the Radio software pre-requisites

### Upgrading from previous versions and other changes

If upgrading from version **6.x** or earlier it is necessary to replace your existing configuration file with the new one supplied in the upgrade. First take a note of any specific configuration items that are not standard or make a copy of your existing **/etc/radiod.conf** configuration file.

**Important:** The name format of Radio playlists has changed from version 7.3 onwards. See Creating and Maintaining Playlist files on page 169. Also, in this release, it is now possible to add, move and delete entries in the play list using any MPD client that is capable of doing so. See Maintaining playlists using external MPD clients on page 173.

**Note:** The location of the log file has changed from **/var/log/radio.log** to **/var/log/radiod/radio.log**. Also, its ownership has been changed from **root** to user **pi**. This has been done to reduce the number of programs that need to run with **sudo**.

**IMPORTANT:** In **Bookworm** the location for the boot configuration file **/boot/config.txt** has been changed to **/boot/firmware/config.txt**. Nearly all instructions in this manual refer to **/boot/firmware/config.txt** however if using **Bullseye** change the instruction to use **/boot/config.txt**. The installation scripts will choose the correct location.

### Installing the Music player daemon

If you haven't already done so upgrade the system packages as shown in **Error! Reference source not found.** on page **Error! Bookmark not defined..**

After reboot install the Music Player Daemon (mpd) and its client (mpc) along with the Python3, MPD and **python3-rpi.gpio** libraries. Note that you must install **python3-mpd** and **not python.mpd**.

```
$ sudo apt install mpd mpc python3-mpd python3-rpi.gpio
```

Answer yes 'y' when asked to continue.

If the installation says it cannot find the above packages, then re-run the update command to update the library list and retry installing the above packages.

```
$ sudo apt update
```

**IMPORTANT:** Unfortunately, the configuration file (**/etc/mpd.conf**) supplied with MPD version 0.22.6 in **Bookworm** and onwards is corrupt. It is necessary to correct this as shown below.

Find the following entries in **/etc/mpd.conf** somewhere around line 232.

#### Corrupt /etc/mpd.conf file

```
#audio_output {
#    type          "alsa"
#    name          "My ALSA device"
#    device        "hw:0,0"
#    mixer_type    "software"
##    mixer_device  "default"      # optional
##    mixer_control "PCM"         # optional
##    mixer_index   "0"           # optional
#}
```

Change the above entries to the following:

#### Corrected /etc/mpd.conf file

```
audio_output {
    type          "alsa"
    name          "My ALSA Device"
    device        "hw:0,0"
    mixer_type    "software"
#    mixer_device  "default"      # optional
#    mixer_control "PCM"         # optional
#    mixer_index   "0"           # optional
}
```

If installing on **Bookworm Lite** this will take quite a long time as there are a lot of software libraries to be installed.

At this stage there are no playlists configured so the music daemon won't play anything. The play lists are created when the **radiod** Debian Package is installed in the next section.

The version of **MPD** released with **Bookworm** is **0.22.6** which is reasonably up-to-date. You should install MPD version **0.22.6** first. When it is all working if you are having problems playing certain stations, it is well worth installing the latest version of MPD as shown in the section called *Compiling and installing the latest Music Player Daemon* on page 118.

#### **Installing pulseaudio**

The **pulseaudio** package may or may not need to be installed. This has been different between various releases of the operating system. Currently the **pulseaudio** package causes a lot of problems with the radio software, in particular **espeak** and the Alsa sound system. If you do not intend to use espeak and need pulseaudio, for example, for blue-tooth devices or pHat BEAT then install it. The following table shows the options:

**Table 2 PulseAudio installation options**

Type of radio installation	Install pulseaudio
Bob Rathbone   Raspberry PI Internet Radio - Chapter 9 - Source files	272

<b>Using espeak</b>	No
<b>Pimoroni Pirate Radio with pHat BEAT</b>	Yes (Installed by phatbeat)
<b>Pimoroni Pirate Audio/mini-speaker</b>	No
<b>Adafruit speaker bonnet</b>	No
<b>LCD display radio</b>	No unless using above DACs
<b>HDMI or touch-screen displays</b>	No unless using above DACs
<b>IQaudIO, HiFiBerry or JustBoom DACs</b>	No
<b>Bluetooth sound devices</b>	Yes

To install **pulseaudio**:

```
$ sudo apt install pulseaudio
```

To remove **pulseaudio**:

```
$ sudo apt remove pulseaudio
```

If installing on Bookworm disable the `pulseaudio-enable-autospawn.service`.

```
$ sudo systemctl stop pulseaudio-enable-autospawn.service
```

At the time of writing there was a fault with the **lxplug-volumepulse** package which stops any of the menus on the Raspberry Pi desktop from being displayed. Remove the **lxplug-volumepulse** package with the following command:

```
$ sudo apt remove lxplug-volumepulse
```

## Installing OLED TFT driver software and libraries

If you are not using any OLED TFT display then skip this section.

**IMPORTANT NOTE:** The following devices and programs use **PIP3** to install the drivers.

- Olimex OLED or SSD1306 OLED TFTs
- Pimoroni Pirate Audio Radio
- SH1106 SPI OLED TFT
- Adafruit 2.5 and 3.5-inch TFT touchscreen
- Monochrome OLEDs which use the LUMA TFT driver
- The weather programs
- The IQaudIO controller

If you attempt to use **PIP3** (The name is a recursive acronym for "Pip Installs Packages") on the **Bookworm OS** it will fail with the following message:

To install Python packages system-wide, try `apt install python3-xyz`, where xyz is the package you are trying to install.

If you wish to install a non-Debian-packaged Python package, create a virtual environment using `python3 -m venv path/to/venv`. Then use `path/to/venv/bin/python` and `path/to/venv/bin/pip`. Make sure you have `python3-full` installed.

For more information visit <http://rptl.io/venv>

To allow **pip3** to be used on **Bookworm OS** carry out the following instruction:

```
$ sudo mv /usr/lib/python3.11/EXTERNALLY-MANAGED  
/usr/lib/python3.11/EXTERNALLY-MANAGED.old
```

Note that the above is all one line. There is no need to carry out the above instruction on system running **Bullseye OS**.

You will now be able to use **pip3** to install the driver software and programs shown in this manual, however you will see the following warning message:

```
WARNING: Running pip as the 'root' user can result in broken permissions and  
conflicting behaviour with the system package manager. It is recommended to  
use a virtual environment instead: https://pip.pypa.io/warnings/venv
```

The above message can be safely ignored for the installation procedures in this manual. However, if installing other author's software with **pip3**, you should seek advice from that author.

If using **Raspbian Lite** also install the **python3-pil** package:

```
$ sudo apt -y install python3-pil
```

### Install display drivers

Note: If you are installing the software for any of the products listed below then first carry out the instructions in the appropriate section. Do this before installing the radio software.

**Table 3 Display driver software installation**

Device	Section/Devices	Page
Pirate Radio	<b>Installing Pimoroni Pirate Radio (pHat BEAT)</b>	34
Pirate Audio	<b>Installing the Pimoroni Pirate Audio</b>	44
SSD1306 0.9-inch OLED	Installing the SSD1306 OLED driver libraries	49
SH1106 SPI (joystick)	Installing the SH1106 SPI OLED driver libraries	53
MHS RPi displays	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
SH1106 1.3-inch OLED	<b>Installing LUMA monochrome OLEDs</b>	59
LUMA OLED devices	SSD1306, SSD1309, SSD1325, SSD1331, SH1106, WS0010	59
Waveshare TFTs	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
Grove LCD RGB	<b>Error! Reference source not found.</b>	<b>Error! Bookmark not defined.</b>
Adafruit TFT	<b>Installing the Adafruit 2.5 and 3.5-inch TFT touchscreen</b>	54

PiFace CAD	Error! Reference source not found.	Error! Bookmark not defined.
Waveshare SH1106 SPI	Installing the SH1106 SPI OLED driver libraries	53

Installing the Radio Daemon the radio software as shown on page 26. You can view the Raspberry PI source at <https://github.com/bobrathbone/piradio6>



Note: This may be out of date compared to the latest version. All source files, including build files, are included in the **radiod** package anyhow so downloading from GitHub isn't necessary.

Before you can download the source from **GitHub** it is necessary to install **git**. For more information on **git** see [http://en.wikipedia.org/wiki/Git\\_\(software\)](http://en.wikipedia.org/wiki/Git_(software))

Install **git** with the following command:

```
$ sudo apt install git
```

Make a development directory and change to it:

```
$ mkdir /home/pi/develop
$ cd /home/pi/develop
```

Now clone the github piradio repository:

```
$ git clone git://github.com/bobrathbone/piradio6
Cloning into 'piradios'...
remote: Counting objects: 71, done.
remote: Compressing objects: 100% (52/52), done.
remote: Total 71 (delta 13), reused 64 (delta 9)
Receiving objects: 100% (71/71), 185.33 KiB | 334 KiB/s, done.
Resolving deltas: 100% (13/13), done.
```

This will create a sub-directory called 'piradio' which will contain the entire source. Also in the **/home/pi/develop/piradio** directory you will also see a directory called **.git** (dot-git). This is the control directory for **git**.

To find out more about **git** and for general support and documentation see:  
<http://git-scm.com>

## Chapter 10 - Streaming to other devices using Icecast2

<b>Contents chapter 10</b>	<b>Page</b>
Inbuilt MPD HTTP streamer	65
Installing Icecast	65
Installing Icecast	65
Overclocking older Raspberry PI's	67
Icecast2 Operation	272
Troubleshooting Icecast2	276

## Inbuilt MPD HTTP streamer

The MPD daemon can be configured to use its own inbuilt streamer. However, this requires a special MPD client such as **gmprc** on the PC. It cannot be easily accessed from a Web browser. If you wish to use the inbuilt streamer see the following URL:

[http://mpd.wikia.com/wiki/Built-in\\_HTTP\\_streaming\\_part\\_2](http://mpd.wikia.com/wiki/Built-in_HTTP_streaming_part_2)

## Introduction to Icecast

You may wish to play the output of the Radio through your PC speakers or a mobile device such as a tablet or telephone. This is possible to do this using **Icecast**. For more information on **Icecast** see the following Wikipedia article: <http://en.wikipedia.org/wiki/Icecast>.

Please also refer to **Error! Reference source not found.** on page **Error! Bookmark not defined..**

## Installing Icecast

Install **icecast2** using the **install\_streaming.sh** script.

```
$ cd /usr/share/radio
$ sudo ./install_streaming.sh
Starting Icecast2 integration with the Music Player Daemon
The Icecast2 installation program will ask if you wish to configure Icecast2.
Answer 'yes' to this. Configure Icecast as follows:
Icecast2 hostname: localhost
Icecast2 source password: mympd
Icecast2 relay password: mympd
Icecast2 administration password: mympd
Continue y/n: y
```

Enter 'y' to continue. The Icecast2 installation program will ask if you wish to configure Icecast2:

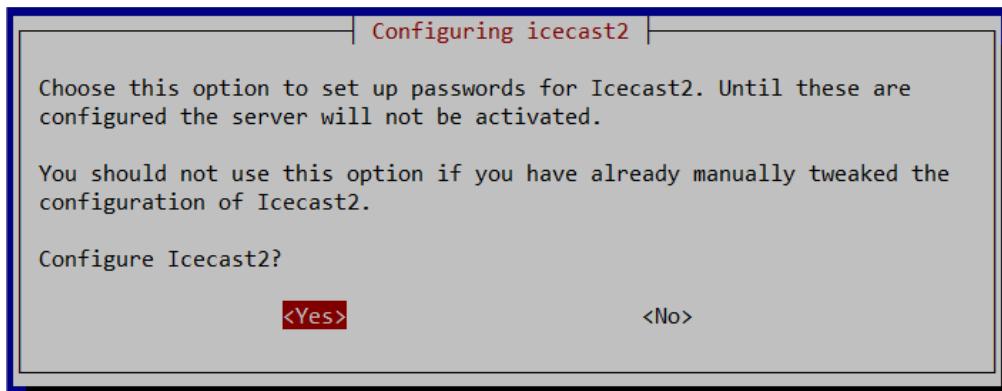


Figure 59 Configuring Icecast2

Answer 'yes' to this. Configure Icecast as follows:

Icecast2 hostname: **localhost** (or the hostname of the Raspberry Pi)  
Icecast2 source password: **mympd**  
Icecast2 relay password: **mympd**  
Icecast2 administration password: **mympd**

It is important that you replace the default password 'hackme' with 'mympd'. The installation program continues configuration. The icecast2 server will be started:

```
Done Configuring icecast2..  
Processing triggers for libc-bin (2.19-18+deb8u6) ...  
Processing triggers for systemd (215-17+deb8u5) ...  
Configuring Icecast2  
Copying /etc/icecast2/icecast.xml to /etc/icecast2/icecast.xml.orig
```

Check that the PI Radio stream (Output 2) is enabled

```
$ mpc outputs  
Output 1 (My ALSA Device) is enabled  
Output 2 (PI Radio MPD Stream) is enabled
```

If not enable it and restart **mpd** and **icecast2**. In this case it is output 2

```
$ mpc enable 2  
$ sudo systemctl restart mpd.service icecast2.service
```

Check that MPD has established a connection with the icecast2 server

```
$ netstat -tn | grep :8000  
tcp        0      0 127.0.0.1:59096      127.0.0.1:8000          ESTABLISHED  
tcp        0      0 127.0.0.1:8000      127.0.0.1:59096          ESTABLISHED
```

The Icecast2 install\_streaming.sh script sets the **streaming\_on** parameter in **/etc/radiod.conf** to yes.

```
streaming_on=yes
```

It also adds the following configuration to **/etc/mpd.conf**.

```
# MPD Radio Stream  
audio_output {  
    type          "shout"  
    name          "PI Radio MPD Stream"  
    description   "MPD stream on Raspberry Pi Radio"  
    host          "localhost"  
    port          "8000"  
    mount         "/mpd"  
    password      "mympd"  
    bitrate       "128"  
    format        "44100:16:2"  
    encoding      "mp3"  
}
```

This completes the installation of Icecast2 however you may need to configure the clock speed if your Raspberry Pi is an earlier version (See *Overclocking older Raspberry PI's* on page 67)

Now go to *Icecast2 Operation* on page 272.

## Overclocking older Raspberry PI's

With older versions of the Raspberry Pi it will almost certainly be necessary to over-clock to handle Icecast2 streaming using the **raspi-config** program. Medium over-clocking seems to be sufficient. Note that the later versions of the Raspberry Pi cannot be overclocked and are fast enough anyhow.

Run **raspi-config**. Select option 'Overclock'. After a warning screen about over-clocking has been displayed, the following screen will be displayed:

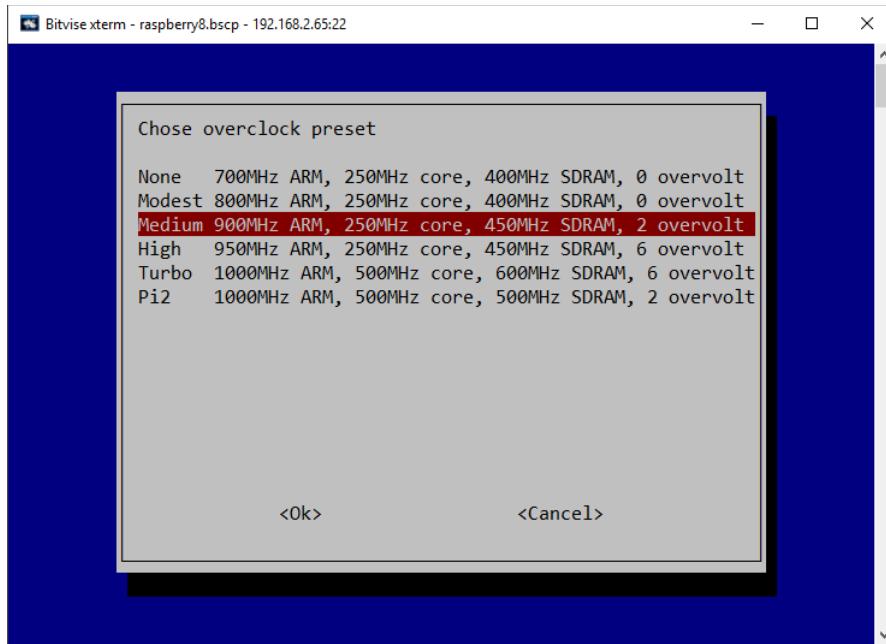


Figure 60 Over-clocking the Raspberry PI

Select 'Medium' to start with. Reboot the Raspberry PI when prompted. Re-test the radio with streaming switched on.

## Icecast2 Operation

The **radiod** daemon has full control over the Icecast2 service and stops and starts it as required. When the radio is first switched on the Icecast2 streaming service will normally be enabled. It can be switched on and off in either LCD/OLED versions of the radio and in the full feature graphical version of the radio.

### Switching on streaming on and off

Use the options menu (Press menu button three times). Step through the menu option using the Channel up/down buttons until "Streaming off" is displayed in the LCD display (assuming Icecast is installed). Press either Volume button and after a short delay the text should change to "Streaming on" in the LCD display. Press the menu button again to exit the options menu.

This starts the Icecast2 service. It also writes the word "on" or "off" to a file called **/var/lib/radiod/streaming**. This file is used to enable or disable the Icecast streaming function at boot time.

## *Checking Icecast2 status*

Use the following command:

```
$ service icecast2 status
● icecast2.service - LSB: Icecast2 streaming media server
  Loaded: loaded (/etc/init.d/icecast2; generated)
  Active: active (running) since Sat 2021-09-04 07:27:34 BST; 21min ago
    Docs: man:systemd-sysv-generator(8)
   Tasks: 5 (limit: 4915)
  CGroup: /system.slice/icecast2.service
          └─632 /usr/bin/icecast2 -b -c /etc/icecast2/icecast.xml
```

## *Enabling Icecast2 at reboot time*

It isn't necessary to enable the Icecast2 service at boot time as the radio program will start it depending on the contents of the **/var/lib/radiod/streaming** file. Should you wish to enable streaming at boot time then enable it with the following command:

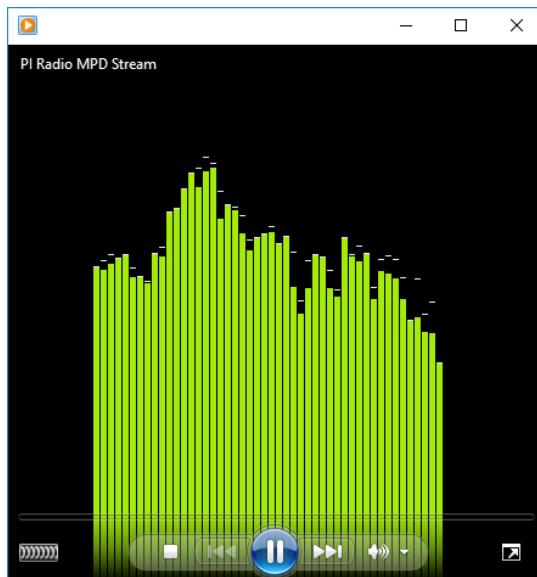
```
$ sudo systemctl enable icecast2
```

## *Playing the Icecast stream on Windows*

To play the Icecast2 radio stream on a PC point your Web browser at the IP address of the radio on port 8000 and the **/mpd** mount point. In the following example the IP address of the radio is 192.168.2.8. So, this would be:

**<http://192.168.2.8:8000/mpd>**

This will normally open the default media player.



**Figure 61 Windows media player**

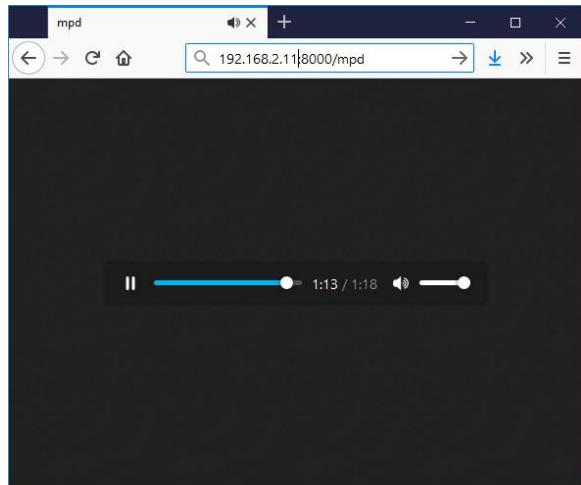


Figure 62 Firefox embedded media player

You will be prompted if you wish to save or open the radio stream. Always select open using the configured media player. The selected radio station or music track should be heard through the PC speakers. At this point you may wish to mute the sound from the radio itself. Simply reduce the volume to almost zero.



Note: It is probably possible to configure Windows 10 Edge or Internet Explorer 11 to use Windows Media player instead of TWINUI. Note: If the mute function is used it will stop the **Icecast** stream.

#### Running the Icecast Administration Web pages

Using the same IP address as shown in the previous example but without the mount point will bring up the administrator window:

<http://192.168.2.8:8000>

The following screen should be displayed. If not continue to the troubleshooting guide at the end of this chapter:

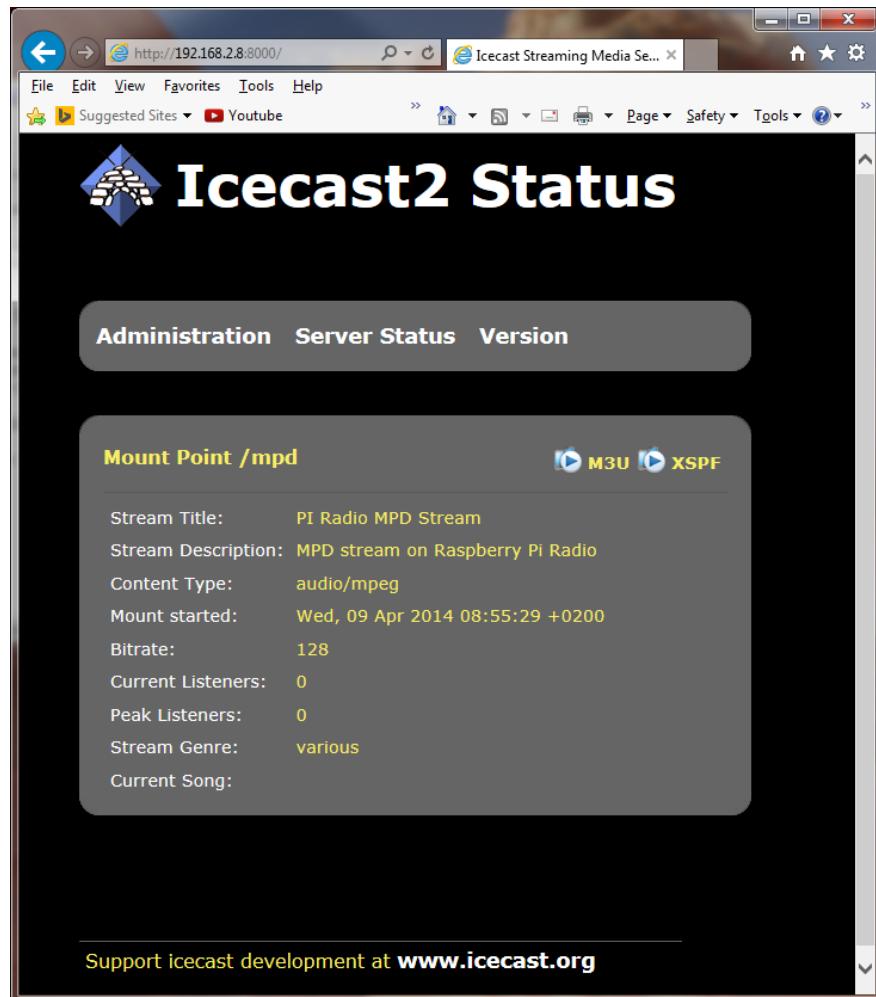


Figure 63 Icecast2 Status

Initially the server status screen is displayed. If you click on the **Administration** tab the you will be prompted for the login credentials.

Log in as admin with user *admin* password *mympd*.



## Playing the Icecast2 stream on an Apple IPad

This is exactly the same as playing the Icecast2 stream on a Windows PC.

1. Open the Safari browser.
2. Type in the Icecast2 URL. For example, <http://192.168.2.11:8000/mpd>
3. Click the M3U button

This should open the iTunes Player and after a short time should start playing the radio stream.

## Playing the Icecast2 stream on an Android device

1. Open your Web browser
2. Type in the Icecast2 URL. For example, <http://192.168.2.11:8000/mpd> (don't include .m3u)
3. When asked to "Complete action with" select your *Android System* then *Music player*

The Icecast stream should start playing. It is important not to key in **mpd.m3u** at the end of the URL. It must be **mpd** only.

## Visual streaming indicator

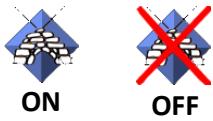
When streaming is switched on an asterix '\*' character is displayed as a visual streaming indicator in the LCD display on the Raspberry PI radio. When the '\*' character is displayed this indicates that the Icecast2 streaming is switched on.

For the four line 20 character display the visual indicator is displayed after the time on the first line.  
09:26 02/05/2014 \*

For the two line by 16 character display there isn't the room to do this so it is displayed after the Volume or Mute message on the second line.

Volume 75 \* or Sound muted \*

## Graphical radio streaming operation



Radio versions prior to version 7.3 could only switch streaming on or off in the LCD/OLED versions of the radio. From 7.3 onwards streaming can be switched on and off in the full featured graphical version of the radio (gradio.py). When Icecast2 is installed the Icecast2 ON icon will appear on the right-hand side of the screen next to the display window. Clicking it once only will switch off streaming and the OFF icon will be shown. It does take a few seconds for this to happen so only click on the icon once.

However, it is necessary to first set the **display\_icecast\_button** in the **[SCREEN]** section of **/etc/radiod.conf** to 'yes'. Add this parameter if it is missing.

```
# Display Icecast streaming button
display_icecast_button=yes
```

## Troubleshooting Icecast2

Help for general problems with icecast2 can be found on the forums at <http://www.icecast.org/>

Icecast2 has two log files in the **/var/log/icecast2** directory namely **access.log** and **error.log**. The error log may give a clue as to the problem.

Below is a simulated error caused by mis-configuring the shoutcast entry in **/etc/mpd.conf** file. Here the hostname 'piradio' has been configured in the **/etc/mpd.conf** shoutcast entry instead of 'localhost'.

```
$ tail -f /var/log/mpd/mpd.log
Apr 07 10:43 : output: Failed to open "PI Radio MPD Stream" [shout]: problem
opening connection to shout server piradio:8000: Couldn't connect
```

### Problem - Icecast streaming page says it can't be displayed.

Possible causes:

- The icecast service is not running on the radio.
  - Start it either from the Radio options menu (Streaming on) or run **sudo service icecast2 start** on the Raspberry PI and retry.
- Incorrect IP address or missing port number in the URL.
  - See Icecast2 Operation on page 272

### Problem – No Mount Point displayed

Possible causes:

This is mostly due to a mis-match in the MPD configuration and the Icecast2 configuration. The icecast configuration is file is **/etc/icecast2/icecast.xml**. Make sure that all of the passwords are set to 'mympd'. The password 'hackme' will not work.

### Problem - Cannot play the stream on my Android device

There are a number of Icecast players which can be downloaded onto Android and play Icecast2 streams across the network without problem. However, the usual Android System Music player should work. The most likely cause of this problem is keying in an incorrect URL (Maybe adding .m3u to the end). See *Playing the Icecast2 stream on an Android device* on page 276.

### Problem – Music keeps stopping or is intermittent

This is difficult to give a definitive answer to this problem. It must be remembered that running MPD and Icecast2 together on a Raspberry PI is pushing the Raspberry PI to its limits. It can also depend on your network or the PC you are using. Personal experience showed no problem playing a stream on PC with a wired network connection however a Laptop connected over a wireless network did not work well. Trying to play two or more devices on the MPD/Icast2 stream is also likely to result in poor results.

With older versions of the Raspberry Pi, try over-clocking the Raspberry PI using the **raspi-config** program. Medium over-clocking seems to be sufficient. See *Overclocking older Raspberry PI* on page 67.



**Note:** The Icecast streaming facility is a fun thing to try out but if it doesn't work properly or is causing you stress; switch the streaming facility off.

## Chapter 11 - Setting up Spotify

<b>Contents chapter 11</b>	<b>Page</b>
Setting up Spotify	279
Spotify hardware requirements	279
Spotify installation	279
Spotify operation	282
Troubleshooting Raspotify	284
Airplay Installation	21

## Setting up Spotify



The radio can also be set up as a Spotify receiver. You will still need a Spotify App on your telephone, PC or Tablet. You will also need a Premium Spotify account and not just a free or trial version.

More information at <https://www.spotify.com>

### Spotify hardware requirements

Spotify works with the on-board audio output or HDMI audio. However, if using a DAC, the card must be capable of hardware volume control. When the radio is running the volume is software controlled by MPD. When Spotify is running, MPD is stopped and the volume is controlled by the standard **amixer** command. The volume control for Spotify is coded in the **volume\_class.py** file. For example, the following command sets the volume to 100%.

```
sudo -u pi amixer cset numid=1 100%
```

The **numid** must match that of the hardware volume control. This is typically numid 1, 2 or 6. The **amixer controls** command will display all the hardware controls for a sound card. To check if your DAC supports Hardware Volume control run the **amixer controls** command.

```
$ amixer controls
:
numid=1,iface=MIXER,name='Digital Playback Volume'
```

DACs using the PCM5122a chip should all support hardware volume control. However, DACs based upon the PCM5102 chip do not have any hardware volume control so the volume cannot be set by **amixer** commands. It is still possible to use these DACs however volume can only be done from the Spotify client App running on a mobile telephone, tablet or PC.

### Spotify installation

The radio software also supports **Raspotify** originally from Dave Cooper which is Spotify for Raspberry Pi OS. First carry out a system update and upgrade as shown in **Error! Reference source not found.** on page **Error! Bookmark not defined..**

Install the **apt-transport-https** package and download and install the **Raspotify** software with the **curl** command into the pi home directory.

```
$ cd
$ sudo apt install apt-transport-https
$ curl -sL https://dtcooper.github.io/raspotify/install.sh | sh
```

This will both download and install **Raspotify**. In the **/etc/default/raspotify** configuration file you will see the **OPTIONS** line for Spotify account details. It is not necessary to configure your account details as these will be picked up from the connecting **Spotify** App on your PC or Mobile.

```
#OPTIONS="--username <USERNAME> --password <PASSWORD>"
```

More information on Raspotify can be found at <https://dtcooper.github.io/raspotify>

Using `sudo edit /lib/systemd/system/raspotify.service` and disable the restart options.

```
#Restart=always  
#RestartSec=10
```

Set the correct device ID to the `ExecStart` in the same `raspotify.service` file. This must match the `device` setting in the `audio_output` definition in `/etc/mpd.conf`

```
ExecStart=/usr/bin/librespot --device=hw:0,0
```

The configuration in `/etc/mpd.conf` is set up by the `configure_audio.sh` script

```
audio_output {  
    type          "alsa"  
    name          "IQAudio DAC/Zero DAC"  
    device        "hw:0,0"
```

Save the file and update `systemd` with the following command:

```
$ sudo systemctl daemon-reload
```

Finally run the `set_mixer_id.sh` script:

```
$ cd /usr/share/radio  
$ sudo ./set_mixer_id.sh  
mixer_volume_id=1
```

The above `mixer_volume_id` output may vary.

The radio will automatically stop and start Raspotify but it may be started and with the following commands:

```
$ sudo systemctl start raspotify  
$ sudo systemctl stop raspotify
```

You can also check the status of `raspotify` with `systemctl`.

```
$ sudo systemctl status raspotify  
● raspotify.service - Raspotify (Spotify Connect Client)  
   Loaded: loaded (/lib/systemd/system/raspotify.service; enabled; vendor  
   preset: enabled)  
     Active: active (running) since Fri 2022-02-18 10:12:25 GMT; 6min ago  
       Docs: https://github.com/dtcooper/librespot  
              https://github.com/librespot-org/librespot  
              https://github.com/dtcooper/raspotify/wiki  
              https://github.com/librespot-org/librespot/wiki/Options  
     Main PID: 13033 (librespot)  
        Tasks: 1 (limit: 2054)  
         CPU: 123ms  
        CGroup: /system.slice/raspotify.service  
                  └─13033 /usr/bin/librespot
```

```
Feb 18 10:12:25 bookworm2 systemd[1]: Started Raspotify (Spotify Connect Client).
```

Disable Raspotify from starting at boot time. Starting and stopping Raspotify is done by the radio.

```
$ sudo systemctl disable raspotify
```

It is necessary to modify **/etc/raspotify/conf** to allow the track title to be displayed.

```
sudo vi /etc/raspotify/conf
```

Comment out the **LIBRESPOT\_QUIET** parameter in **/etc/raspotify/conf**. You will need to use your editor with **sudo**. For example, **sudo nano /etc/raspotify/conf**

```
# Only log warning and error messages.  
#LIBRESPOT_QUIET=
```

Failure to do so will mean that track titles will not be displayed.

Raspotify messages during operation can be observed with the following command:

```
$ journalctl --lines 0 --follow _SYSTEMD_UNIT=raspotify.service
```

Finally stop the Raspotify service and restart radiod.

```
$ sudo systemctl stop raspotify  
$ sudo systemctl restart radiod
```



**Note:** The authors of Raspotify have stated that it is not possible to display the artist's name and never will be. Only the track title can be displayed.

## Spotify operation

To start the Radio as a Spotify receiver either select Spotify from the Playlists (LCD or OLED versions) or from the Sources window in the touchscreen version:

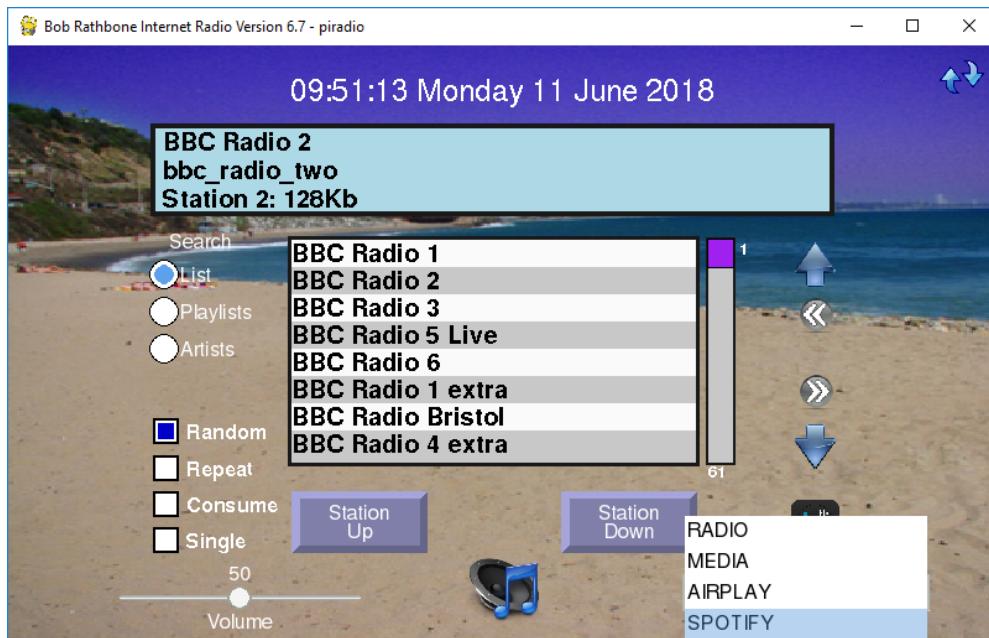


Figure 64 Starting the Spotify Receiver

The following window will appear however a different message may appear on the second line of the display window:



Figure 65 The radio in Spotify mode

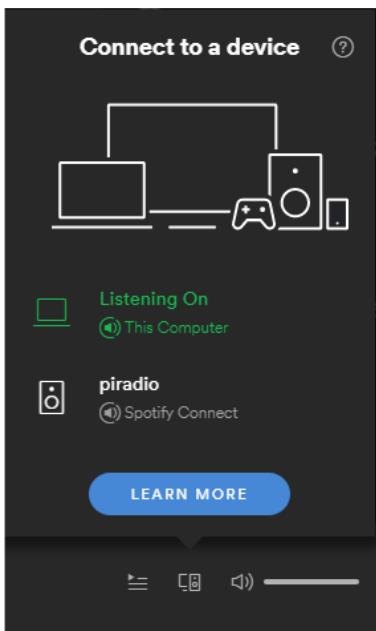


Figure 66 Spotify connecting to the radio

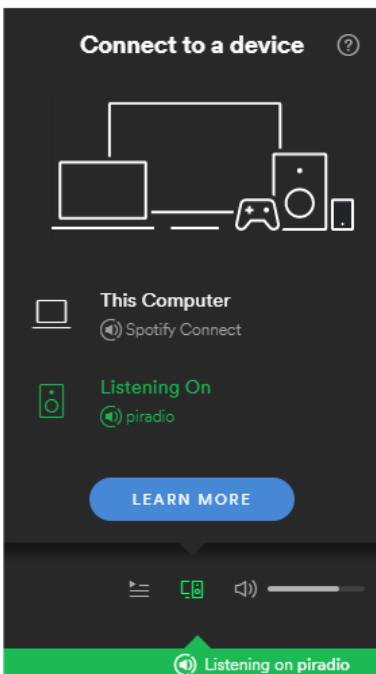


Figure 67 Listening to Spotify on the radio

To use Spotify you will need a Spotify App on your telephone, PC or Tablet.

As previously mentioned you will need a Premium Spotify account and not just a free or trial version.

On the radio, press the Menu button until you come to the sources selection. Press channel Up or Down until you see Spotify displayed.

Press the Menu button again and the radio will stop the MPD player and start raspotify.

Now click on *Connect to a device* in the Spotify application. You should see an entry called Spotify Connect with the hostname of your radio.

Click on the *Spotify Connect* device for the radio (*piradio* in this example).

The Spotify application will switch from the current device (PC, mobile phone or tablet) to the Raspberry Pi radio.

You can control the volume either from the Spotify Application or using the volume control on the radio.

To exit Raspotify on the Raspberry Pi press the Menu button and then select another source such as the radio.

The Spotify application will connect the Raspotify application on the Radio.



**Note:** If you don't hear any sound then turn the volume up to full.

In the case of the touchscreen version of the program the following screen will be displayed:



Figure 68 Spotify playing a music track

In the case of the LCD or OLED version of the radio the title line above will be displayed on the second line of LCD or OLED screen.



**Note:** Raspotify unfortunately does not supply the Artist information. Only the track name is supplied by **librespot** which is used by Raspotify. There is currently no solution for this.

## Exiting Spotify

For the LCD and OLED versions press the Menu button until the Select source: window is displayed. Select any other source (playlist) to exit.

In the case of the touchscreen version of the radio, press the “Exit Spotify”button at the bottom of the screen.

## Troubleshooting Raspotify

### Installation problems

If the curl command to install the Raspotify software fails then carry out a system update and upgrade as shown in **Error! Reference source not found.** on page **Error! Bookmark not defined..** Retry the curl command.

### Raspotify exits with a 101 error code

This is almost certainly an authentication fault. Check your user name and password are correctly set up in **/etc/default/raspotify** and retry. However, it isn't actually necessary to put a username and password in **/etc/default/raspotify** as Raspotify will be using a Raspotify Premium account running on a PC, tablet or mobile phone.

### The client connects to Raspotify but no sound heard

Check that the device ID has been added to the **ExecStart** statement in the same **raspotify.service** file.

```
ExecStart=/usr/bin/librespot . . . . . --device=hw:0,0
```



**Note:** The author does not directly support Raspotify.

Report Raspotify issues to <https://github.com/dtcooper/raspotify/issues>

Raspotify comes with an MIT licence. See <https://opensource.org/licenses/MIT>

### Cannot change Raspotify volume

In some cases, this is normal as many cheaper sound cards (DACs) do not have mixer controls. Mixer controls are required to change the volume on the Raspberry Pi when running Spotify or Airplay. In the case of no mixer controls the sound when running Spotify must be controlled from the App on either your mobile device or from the PC application.

However, it is possible to set up a software mixer (SoftVol).

### Raspotify initial sound too loud

Edit the `/lib/systemd/system/raspotify.service` file. Locate the following line

```
Environment="VOLUME_ARGS=--enable-volume-normalisation --volume-ctrl linear  
--initial-volume 100"
```

Change to the desired volume setting.

```
--initial-volume 75
```

## Acknowledgements

My thanks to [Matt Hawkins](#) for the original LCD screen driver routines. It made the job of writing the `lcd_class.py` much easier.

The original instructions on how to use Rotary Encoders came from an excellent article by [Guy Carpenter](#). See:

<http://guy.carpenter.id.au/gaugette/2013/01/14/rotary-encoder-library-for-the-raspberry-pi/>

To Adafruit Industries for their excellent LCD plate and I2C code. See <http://www.adafruit.com>.

To Steffen Müller for his article on Streaming audio with MPD and Icecast2 on Raspberry Pi.  
See <http://www.t3node.com/blog/streaming-audio-with-mdp-and-icecast2-on-raspberry-pi/>

To contributors such as Alan Broad who supplied photos of the Lego example of the radio plus code contribution.

To Mike Whittaker for his contribution on how to drive the USB speaker set. To other contributors such as Jon Jenkins for his excellent implementation using an old Zenith radio.

Thanks to Michael Uhlmann for the work he did testing various Android Apps for MPD. Also, Simon O'Niel who carried out configuration and testing of Cmedia sound dongle.

To Open Electronics Magazine for their excellent article on the Raspberry PI radio using the Adafruit LCD plate. See <http://www.open-electronics.org/internet-radio-with-raspberry-pi/>

To Joaquin Perez, Broadcast Engineer, Leeds for the backlight dimmer and circuit diagram.

To Luboš Ruckl for his work on the Rotary encoder class (adapted from code by Ben Buxton) and the PCF8574 LCD class (adapted from code by an unknown author but believed to be from the Arduino community).

To Béla Mucs from Hungary for his brilliant idea to support speech for visually impaired and blind persons. This facility uses the **espeak** package.

Gordon Henderson <https://projects.drogon.net/> for the GPIO wiring utility.

To <http://www.allaboutcircuits.com> for **Error! Reference source not found..**

Jim Downey from Mobile Alabama, the USA for his article on the backlight for Chinese 1602 I2C LCDs. See [http://mbvmc.org/LCD\\_Backlight.pdf](http://mbvmc.org/LCD_Backlight.pdf)

Tomás González, Sevilla, Spain for his changes to lcd\_adafruit\_class.py (Previously ada\_lcd\_class.py) to switch on the Chinese 1602 I2C LCD backlight.

To the authors of the SGC Widget routines. Copyright (c) 2010-2012, Sam Bull and Michael Rochester

To ModMyPi (<https://www.modmypi.com>) for their excellent Pi products and setup guides.

Icons used in the graphic versions of the radio. Clipart library <http://clipart-library.com> and IconSeeker <http://www.iconseeker.com>

Thanks to **Olimex Limited** for their SSD1306 OLED routines which were used in the oled\_class.py program and for their excellent technical support. See <http://www.olimex.com>. Original source <https://github.com/SelfDestroyer/pyMOD-OLED.git>

To **Midas Displays**, Great Yarmouth, UK, for their help and sponsorship for the implementation of Russian/Cyrillic OLED character displays. See <https://www.midasdisplays.com>

Thanks to Gordon Garrity at IQaudio <http://iqaudio.co.uk> for his help and sponsorship to develop the radio to support the IQaudio Cosmic Controller and SSD1306 OLED display.

Thanks to **Pimoroni** for their excellent Pirate radio using pHat BEAT software and hardware. See <http://pimoroni.com> for further information.

Thanks to **PiFace UK** for their **PiFace CAD** product. This makes a very easy entry level radio using this software. See <http://www.piface.org.uk/>

Franz-Josef Haffner, from Germany, for his conversion of a Schneider Frères Rondo vintage radio.

To Andrey Gunich (Андрей Гунич) from the Ukraine for his help with the development and testing of Russian/Cyrillic OLED character displays.

To **othermod.com** for their article on using PWM and Low Band Pass filters to produce an audio circuit for the Raspberry Pi Zero (W).

Thanks to **Jim Smith** from Shropshire for his work on optical rotary encoders.

To Syds Post for his work on conversion of the IR Remote Control software from Python2 to Python3 on the Raspberry Pi Bookworm OS. See <https://www.sydspost.nl> (Dutch) in particular his article on the Vintage Internet Radio (Use browser translation facility for English translation).

David Steele for his **comitup** package for configuring Wi-Fi access via a Web Interface. See <https://davesteele.github.io/comitup>

The **ETechnoG Team** <https://www.etechnog.com> for their article on supply voltages.

**Artur Sierzant** for his excellent support and work on converting **O!MPD** (Used in the **radiod** Web interface) to run on **Bookworm OS** and **PHP8.2**.

Website: <http://ompd.pl/>

GitHub <https://github.com/ArturSierzant/OMPD>

Dave Jesse (UK) for his work on the **Grove RGB I2C colour LCD (lcd\_i2c\_jhd1313\_sgm31323.py** driver software) Version 5 uses the SGM31323 controller for the backlight.

To all constructors of this project who have sent in photos of their radio's and their ideas for improvement and the many appreciative e-mails that I have received from them.

## Technical support

Technical support is on a voluntary basis by e-mail only at [bob@boprathbone.com](mailto:bob@boprathbone.com). If there are any problems with this email address then also CC [r.h.rathbone@gmail.com](mailto:r.h.rathbone@gmail.com). Before asking for support, please first consult the troubleshooting section on page 193. I will always respond to e-mails requesting help and will never ignore them. I only ask that you do the same (i.e. Did my suggestions help or not?). Be sure to provide the following information:

- What have you built (Adafruit or normal LCD variants, sound cards etc)?
- Which program and wiring version are you running?
- A clear description of the fault.
- What you have already done to locate the problem?
- Is anything displayed on the LCD or Graphics screen?
- Did you run the test programs and what was the result?
- Switch on DEBUG logging as described on page 165, run the program and include the **/var/log/radiod/radio.log** file.
- Did you vary from the procedure in the manual or add any other software?
- Please do not answer my questions with a question. Please supply the information requested.

Run the configuration display and send the **/usr/share/radio/config.log.tar.gz** that it produces to [bob@boprathbone.com](mailto:bob@boprathbone.com). This will save a lot of questions about your configuration.

```
$ cd /usr/share/radio
$ ./display_config.sh
:
This configuration has been recorded in /usr/share/radio/config.log
A compressed tar file has been saved in /usr/share/radio/config.log.tar.gz
Send /usr/share/radio/config.log.tar.gz to bob@boprathbone.com if required
```



Please note that support for general Raspberry PI problems is not provided. Only issues relating to the Radio software will be investigated.

For general Raspberry PI support see the following site:  
<http://www.raspberrypi.org/forums/>

For support on Music Player Daemon issues see the help pages at the following link:  
<http://www.musicpd.org/>

For issues relating to Icecast2 streaming see:  
<http://www.icecast.org>

For those of you who want to amend the code to suit your own requirements please note: I am very happy to help people with their projects but my time is limited so I ask that you respect that. Please also appreciate that I cannot engage in long email conversations with every constructor to debug their code or to teach Python.

## Glossary

3Vo	In the case of OLEDs this is a +3.3V 100mA auxiliary output (unused in this project)
AC	Alternating Current - In this context 110V or 220V
AP	Application processor (Also see CPU)
API	Application Programming Interface
DC	Direct Current - In this context +5V or +3.3V
A2DP	Advanced Audio Distribution Profile - Bluetooth
AAC	Advanced Audio Coding
ALSA	Advanced Linux Sound Architecture
ARM	Advanced RISC Machine (Type of processor used in Mobile phones and Raspberry Pi)
ASX	Advanced Stream Redirector
ATC	Air Traffic Control
BJT	Bipolar Junction Transistor
BOT	USB Mass Storage Bulk-Only Transport (BOT) – Legacy USB device access (Also see UAS)
CGI	Common Gate Interface – Executable Server-Side scripts
CAD	Control and Display (PiFace) – No longer supported in this version
CD	Compact disc - In this context CD marker pen
CIFS	Common Internet File System
CODECS	Encoder/Decoder for media streams
CPU	Central Processor Unit
DAC	Digital to Analogue Converter (Digital to audio frequency analogue in this case)
DOS	Denial of Service – Attack software aimed at taking down an Internet service (Web etc.)
DOS	Disk Operating System (Microsoft Operating System for PCs)
DDOS	Distributed Denial of Service – DOS attack from hundreds or thousands of computers
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name System. Converts a URL such as google.com to an IP address or addresses
DSP	Digital Signal Processing/processor (In this context it is mixer control)

DT	Device Tree (Overlay). Device (Sound cards) configuration in <b>/boot/firmware/config.txt</b> in Raspbian
EMI	Electromagnetic Interference (For example fluorescent lighting etc.)
EEPROM	Electrically Erasable Programable Read Only Memory (Typically a Boot PROM)
FBCP	Frame Buffer CoPy – HDMI display mirroring driver
FET	Field Effect Transistor
FLIRC	A USB device and software which maps an IR remote control to the keyboard
HCI	Host Controller Interface. Bluetooth standard for operation with three-wire UART
HLS	HTTP Live Streaming. Media streaming protocol for visual and audio media delivery
HDMI	High-Definition Multimedia Interface for audio and video plus Ethernet interface.
GPIO	General Purpose IO (On the Raspberry PI)
I2C	Industry standard serial interface (Philips now NXP) using data and clock signals
I2S	Inter-IC Sound (Used in DAC interface) from Philips (Now NXP)
IATA	International Air Transport Association
ICAO	International Civil Aviation Organization
ID3	Metadata standard for MP3 files which provides title, artist, album, track number, and other information embedded in the currently playing stream
IIC	Alternative name used by some manufacturers for I2C
IOPS	Input/Output Instructions Per Second (pronounced eye-ops)
IP	Internet Protocol
IPS	In-Plane Switching, a type of LED (a form of LCD) display panel technology.
IPv4	Internet Protocol Version 4
IPv6	Internet Protocol Version 6
IR	Infra-Red (sensor) for use with infra-red devices such as remote controls
LCD	Liquid Crystal Display, also see OLED
LE	Low Energy – In this context Bluetooth LE (Bluetooth 4.0 core specification)
LIRC	Linux Remote Control software
M3U	MPEG3 URL

MAC Media Access Control (address)

MEMS Micro Electro Mechanical Systems (Manufacturing technique, Microphones in this context).

Micro HDMI Miniaturized version of the High-Definition Multimedia Interface specification

MIPI Mobile Industry Processor Interface (Used for camera and display interfaces)

MMS Microsoft Media Server Internet protocol

MOSFET Metal Oxide Semiconductor Field Effect Transistor

MPC Command line client for MPD

MPD Music Player Daemon

MPEG Moving Picture Experts Group

MPEG3 Music encoding standard from MPEG

NAS Network Attached Storage

NFS Network File System

NTP Network Time Protocol

OLED Organic Light Emitting diode. Also, OLED character displays gradually replacing LCDs

OS Operating system (Raspbian Bullseye in this case)

PA Personal Amplifier (Input to audio stage of a vintage radio)

PC Personal Computer

PCM Pulse-Code Modulation is a method used to digitally represent sampled analogue signals

PDF Portable Document Format

PIL Python Image Library. Used by Python to manipulate images and colours

PWM Pulse Width Modulation, is an alternative method for audio circuit design.

PHP A server-side scripting language designed primarily for Web development

PID Process ID

PIL Python Imaging Library – Linux graphics routines – **Pillow** is a branch of PIL

PLS MPEG Playlist File (as used by Winamp)

RISC Reduced Instruction Set Computer – Also see ARM

RPi Raspberry Pi

RSS	Really Simple Syndication – Web feed usually containing news items
RTC	Real Time Clock – Usually a battery backed up clock chip updating system time
SCO	Synchronous Connection Oriented link – Bluetooth
SD	(Secure Digital) Memory Card commonly found in cameras and Smartphone's
SDHC	Secure Digital High Capacity – SD Memory card up to 32 Gigabyte capacity
SDK	Software development kit
SDXC	Secure Digital Extended Capacity - SD Memory card up to 2 Terabyte capacity
SPI	Serial Peripheral Interface (Motorola) used by PiFace CAD
S/P-DIF	Sony/Philips Digital Interface for short distance audio transmissions.
SSH	Secure Shell – Encrypted terminal
SSID	Service Set Identifier. An SSID is the public name of a wireless network
SSD	Solid State Disk Drive
SVI	Standard Volume Indicator. Another name for VU meter/indicator
System V	Particular version of UNIX, many features of which have found their way into Linux
TCP/IP	The common name for network protocols used by the Internet and computer networks.
TFT	Thin Film Transistor – Used in display technology and touch-screens
TTS	Text-To-Speech (eSpeak in this case)
TV	Television (In this case, with one or more HDMI inputs)
UART	Universal Asynchronous Receiver-transmitter for asynchronous serial communication
UAS	USB Attached SCSI protocol – Newer faster USB drive access protocol (Also see BOT)
UDP	Universal Datagram Protocol. A connectionless network protocol over IP
UHS	Ultra High Speed (SDHC and SDXC memory cards)
URL	Universal Resource Locator (A link to a Web page for example)
USB	Universal Serial Bus
USB 2.0	USB with a maximum signalling rate of 480 Mbit/s (60 MB/s)
USB 3.x	USB using full duplex communication at speeds up to 5 Gbit/s (625 MByte/s)

**USB-C** 24-pin USB connector system which can be inserted either way

**USB OTG** USB On-The-Go, software to support USB devices (Not supported with this radio)

**VCC** Common Collector Voltage; the positive supply voltage to a device

**VDD** Voltage Drain Drain Supply; the positive supply voltage to a device

**VEE** Voltage Emitter. the negative power supply voltage to a device

**VIN** Voltage In – It is equivalent to VCC

**VLC** Media player used by Pimoroni software (Not used by this radio software)

**VNC** Virtual Network Computing – Used to display a remote screen on a local desktop

**VSS** Voltage Source Supply; the negative supply voltage to a device

**VU** Volume Unit – Volume meter/indicator also known as SVI (Standard Volume Indicator)

**WEP** Wired Equivalent Privacy (WEP) is a security algorithm considered less secure than WPA

**WI-FI** Wireless Network typically using the 802.11 Wireless Network protocol

**WPA** Wi-Fi Protected Access (WPA) – Also see WPA2

**WPA2** Wi-Fi Protected Access version II, an enhanced, more secure version of WPA.

**XML** Extensible Mark-up Language. A Web technology used for transmitting data structures

## Appendix A - System Files used by the Radio Program

### A.1 Files added to the system

#### /etc/radiod.conf

This is the main configuration file for the radio program. It is mainly configured by the **configure\_radio.sh** program.

```
# Raspberry Pi Internet Radio Configuration File
# $Id: radiod.conf,v 1.64 2024/09/03 07:55:14 bob Exp $

# Configuration file for version 7.0 onwards
# Used for both 40 and 26-pin Raspberry Pi versions
#
# Please Note: Configuration of this file, for the most part, is done by
# running the configure_radio.sh and configure_audio.sh programs.
# NOTE: The configuration in this file uses GPIO numbers and not physical
pin numbers.

[RADIOD]

# loglevel is CRITICAL,ERROR,WARNING,INFO,DEBUG or NONE
loglevel=INFO

# Logfile creation mode, either truncate or tail
log_creation_mode=truncate

# Startup option either RADIO,MEDIA or LAST a playlist name
#startup=RADIO
startup=Radio

# MPD client timeout from 2 to 15 seconds default 10
client_timeout=10

# Codecs list for media playlist creation (Run 'mpd -V' to display others)
CODECS="mp3 ogg flac wav wma"

# Set date format, US format = %H:%M %m/%d/%Y
# To display timezone use %Z  dateformat=%H:%M %Z %d/%m/%Y
dateformat=%H:%M %d/%m/%Y

# Volume range 10, 20, 25, 50 or 100
volume_range=100

# Volume display text or blocks
volume_display=blocks

# MPD port number (Do not change unless MPD reconfigured)
mpdport=6600

# Remote control UDP server listen host either 0.0.0.0 (All interfaces) or
localhost
remote_control_host=localhost

# Remote control communication host and port Default localhost 5100
remote_control_port=5100

# This allows another host to send UDP messages to the UDP server
# It is either localhost or the IP address of the remote server
remote_listen_host=localhost

# Audio output device - Must match an output using the "aplay -l" command
```

```

# The configure_audio.sh program will set this to headphones(default), HDMI,
# DAC or USB
# depending upon the audio device/card selection. You can override this
# setting
# with your own unique string from the aplay command, for example
"HiFiBerry"
audio_out="headphones"

# Audio config lock stops audio configuration from being dynamically
# changed if HDMI plugged in or out. Default no
audio_config_locked=no

# Output LED for remote control, default GPIO 11 (pin 23) or
# GPIO 13 (pin 33) for AdaFruit plate or PiFace CAD (40 pin RPi needed)
# Use GPIO 16 (pin 36) for designs using IQAUDIO DAC cards etc.
# Use GPIO 14 (pin 8) for designs using IQAudIO Cosmic controller
# remote_led=0 is no output LED
remote_led=0

# Display playlist number in brackets yes or no
display_playlist_number=no

# Background colours (If supported) See Adafruit RGB plate
# Adafruit RGB colors: OFF, RED, GREEN, BLUE, YELLOW, TEAL, VIOLET, WHITE
# Pimoroni displays RGB colors: Refer to PIL library color chart
bg_color=WHITE
mute_color=PURPLE
shutdown_color=TEAL
error_color=RED
search_color=GREEN
news_color=CYAN
info_color=BLUE
menu_color=YELLOW
source_color=TEAL
sleep_color=OFF

# Status LED (Typically for vintage radio) Normally 27,22,23 respectively
rgb_red=0
rgb_green=0
rgb_blue=0

# Menu rotary switch (optional) Normal values are 24,8 and 7 respectively.
Value 0 disables
menu_switch_value_1=0
menu_switch_value_2=0
menu_switch_value_4=0

# The i2c_address overrides the default i2c address. 0x00 = use default
# Some backpacks use other addresses such as 0x2F, then set i2c_address=0x2F
i2c_address=0x00

# Some i2c displays such as Grove I2C RGB have a seperate interface for
colour
# The Grove RGB jhd1313_sgm31323 LCD uses address 0x30.
i2c_rgb_address=0x30

# I2C normaly uses bus 1 on the I2C interface. However the very first
Raspberry
# used bus 0. If you are using a very old Pi then set i2c_bus=0
# Run ./display_model.py to see what model Pi you are running
i2c_bus=1

# Set LCD character translation on or off. Graphic and OLED versions
unaffected
# as they do not need language translation and use system fonts
translate_lcd=on

```

```

# Language font translation table to be used.
# Current choices are English(Default), European(Western) and Russian
# Translation tables are contained in the /usr/share/radio/codes directory
# Add other translation tables to the above directory
language=English

# Set LCD/OLED controller being used. HD44780U (default) or HD44780 (Older
LCDs)
controller=HD44780U

# Select LCD code page table 0,1,2 or 3. Default 0
# 0 = Use codepage parameter specified in primary font file (Selected by
language)
# 1, 2 or 3 Override codepage setting in the primary font file
codepage=0

# Romanize characters (eg convert Cyrillic to Latin characters),
# Set to on or off. Default is on
romanize=on

# Speech for visually impaired or blind listeners, yes or no
# Needs espeak package - sudo apt-get install espeak
speech=no
# Speech volume as a percentage of the normal MPD volume
speech_volume=75
# Verbose - yes = each station change is spoken
verbose=no
# Speak hostname and IP address
speak_info=no

# Set the user interface to 'buttons' or 'rotary_encoder' or 'graphical'
# These can also be used in conjunction with a graphical/touchscreen display
user_interface=rotary_encoder

# Switch settings for Rotary encoders or buttons
menu_switch=17
mute_switch=4
up_switch=24
down_switch=23
left_switch=14
right_switch=15

# Pull GPIO up/down internal resistors (Applies to button interface only).
# Default:down
pull_up_down=down

# Display types
# NO_DISPLAY = No display connected
# LCD = directly connected LCD via GPIO pins
# LCD_I2C_PCF8574 = Arduino (PCF8574) I2C backpack
# LCD_I2C_ADAFRUIT = Adafruit I2C backpack
# LCD_ADAFRUIT_RGB = LCD I2C RGB plate with buttons
# GRAPHICAL = Graphical or touch screen display
# OLED_128x64 = 128x64 pixel OLED
# PIFACE_CAD = PiFace CAD with six push buttons using the SPI interface
# ST7789TFT = Pimoroni Pirate audio with four push buttons using the SPI
interface
# SSD1306 = Sitronix SSD1306 controller for the 128x64 pixel OLED
# SH1106_SPI = Sino Wealth SH1106 controller for the Waveshare 128x64 pixel
OLED (SPI)
# LUMA = Luma driver for SSD1306, SSD1309, SSD1325, SSD1331, SH1106,
SH1106_128x32, WS0010
#           For example LUMA.SH1106 for OLEDs using the sh1106 chip
# LCD_I2C_JHD1313 = Grove RGB 2x16 LCD (AIP31068L controller)

```

```

# LCD_I2C_JHD1313_SGM31323 = Grove RGB 2x16 LCD (SGM31323 controller for
backlight control)
display_type=LCD

# Display width, 0 use program default. Usual settings 16 or 20
display_width=20
display_lines=4

# Font size and name (TFT displays only)
font_size=11
font_name="/usr/share/fonts/truetype/dejavu/DejaVuSansMono.ttf"

# LCD GPIO connections for 40 pin version of the radio
lcd_select=7
lcd_enable=8
lcd_data4=5
lcd_data5=6
lcd_data6=12
lcd_data7=13

# Display Scroll speed 0.01 to 0.6 seconds
scroll_speed=0.05

# Settings are standard, alternative, rgb_rotary rgb_i2c_rotary
# Some rotary switches do not work well with the standard rotary class
# Set to "alternative" to use the alternative rotary encoder class
# For rotary encoders with RGB LEDs in the shaft set to rgb_rotary
# For rotary I2C encoders with RGB LEDs in the shaft set to rgb_i2c_rotary
rotary_class=standard

# Rotary encoder detection step size 'full' step or 'half' step (default)
# Only used in the standard rotary encoder class and NOT the alternative
# rotary class
rotary_step_size=half

# KY-040 encoders etc have their own physical 10K pull-up resistors and do
not
# need the internal gpio pull-up resistors.
# In that case set rotary_gpio_pullup=none otherwise set it to "up"
rotary_gpio_pullup=up

# Station names source, list or stream
station_names=list

# Action on exiting radio. Stop radio only, shutdown the system or execute
program
# 1) Shutdown system: exit_action=stop_radio
# 2) Exit to operating system: exit_action=stop_radio
# 3) Execute a program: exit_action=<programm name>
#   For example: exit_action=/usr/share/radio/weather.py
exit_action=shutdown

# Bluetooth device ID - Replace with the ID of your bluetooth
speakers/headphones
# Example: bluetooth_device=00:75:58:41:B1:25
# Use the following command to display paired devices
# bluetoothctl paired-devices
bluetooth_device=00:00:00:00:00:00

# Action when muting MPD. Options: pause(Stream continues but not processed)
# or stop(stream is stopped)
# mute_action=stop
mute_action=pause

# Shoutcast ID
shoutcast_key=anCLSEDQODrElkxl

```

```

# Pimoroni PHat Beat (pivumeter)
pivumeter=no

# Internet check URL. This must be a reliable URL and port number
# which can be contacted such as google.com
# The port number is normally 80 (HTTP).
# The internet_timeout in in seconds
# Disable by removing the URL from internet_check_url= parameter
internet_check_url=google.com
internet_check_port=80
internet_timeout=10

# ireventd daemon keytable name
keytable=myremote.toml
# Event device name. Usually rc0, rc1 or rc2
event_device=rc0

# OLED parameters
# Flip display vertically (yes or no) OLED only at present
flip_display_vertically=no

# Splash screen
splash=bitmaps/raspberry-pi-logo.bmp

# Allow updating of playlists by external clients yes/no (Experimental)
update_playlists=no

# I2C addresses and interrupt pins for RGB I2C Rotary Encoders
volume_rgb_i2c=0x0F
channel_rgb_i2c=0x1F
volume_interrupt_pin=22
channel_interrupt_pin=23

# Shutdown command (Default "sudo shutdown -h now")
# Replace with required command for example "x735off" for a X735 v2.5 shield
shutdown_command="sudo shutdown -h now"

# Graphics (touch screen) screen settings
[SCREEN]
# Size is in pixels. Supported is 800x480 (7" screen) or 720x480(3.5"
screen)
# or 480x320 (2.8" or 3.5" screen) or 1024x600 (Maximum)
# Also see framebuffer_width and framebuffer_height parameters in
/boot/config.txt
screen_size=800x480
fullscreen=yes

# Screen save time in minutes, 0 is no screen saver
screen_saver=0

# Title %V = version %H = hostname
window_title=Bob Rathbone Internet Radio Version %V - %H

# Colours - See https://htmlcolorcodes.com/color-names/
window_color=turquoise
banner_color=white
labels_color=white
display_window_color=lightblue
display_window_labels_color=black
slider_color=mediumseagreen

# Display or hide mouse
display_mouse=yes

# Display Icecast streaming button

```

```

display_icecast_button=no

# Wallpaper backgrounds. See /usr/share/rpd-wallpaper/
# Run: sudo apt install rpd-wallpaper
wallpaper=/usr/share/rpd-wallpaper/aurora.jpg
# More backgrounds in /usr/share/scratch/Media/Backgrounds (Install scratch)
# Run: sudo apt install scratch

# Set date format for graphic screen
dateformat=%H:%M:%S %A %e %B %Y

# Allow switching between vgradio and gradio
switch_programs=yes

# Display shutdown button. The action of this button is dependent
# upon the exit_action parameter setting (shutdown or stop radio
display_shutdown_button=yes

# The following is specific to the vintage graphical radio
scale_labels_color=white
stations_per_page=40
display_date=yes
display_title=yes

[AIRPLAY]

# Airplay activation yes or no
airplay=no

# Mixer preset volume for radio and media player if using sound card
# Set to 0 if using onboard audio or USB sound dongle.
# If using a sound card set to 100% initially and adjust as necessary
# Old name was mixer_volume
mixer_preset=100

```

### [/etc/logrotate.d/radiod](#)

This file causes the **/var/log/radiod/radio.log** to be rotated so that it doesn't continue to grow and fill the disk.

```

/var/log/radiod/radio.log {
    weekly
    missingok
    rotate 4
    compress
    notifempty
    maxsize 150000
    copytruncate
    create 600
}

```

Old log files are compressed and renamed, for example **/var/log/radiod/radio.log.1.gz**.

### [/lib/systemd/system/radiod.service](#)

This file is part of the new **systemd** startup services. This file defines the so-called radiod service. It defines how the service will start-up and what services it needs.

```

# Radio systemd script
:
[Unit]
Description=Radio daemon
After=network.target

```

```
[Service]
Type=simple
ExecStart=/usr/share/radio/radiod.py nodaemon

[Install]
WantedBy=multi-user.target
```

### [/usr/lib/systemd/system/ireventd.service](#)

The **ireventd.service** service definition file is part of the new **systemd** startup services. This file defines the so-called **ireventd** service which starts and stops the IR Remote Control software.

```
# Radio remote control systemd script
# $Id: ireventd.service,v 1.3 2023/07/22 15:58:19 bob Exp

# It uses ireventd.py and ir_daemon.py (Daemon)
# This file is copied to /lib/systemd/system/ireventd.service

[Unit]
Description=Radio remote control daemon
After=network.target

[Service]
Type=simple
ExecStart=/usr/share/radio/ireventd.py nodaemon
ExecStop=/usr/share/radio/ireventd.py stop

[Install]
WantedBy=multi-user.target
```

### [/etc/asound.conf](#)

The source of this file is one of the files found in the **/usr/share/radio/asound** directory. Which file will be copied depends upon the required configuration for the sound device being used

```
.# Set default mixer controls
ctl.!default {
    type hw
    card 0
}

# Set default PCM device
pcm.!default {
    type plug
    slave {
        pcm "plughw:0,0"
        format S32_LE
    }
}
```

The above example is for the on-board audio jack.

## Cronjob scripts

### *The /etc/cron.weekly/radiod script*

This script runs once each week and creates playlist files and copies them to the **/var/lib/mpd/playlists** from the stations defined in the **/var/lib/radiod/stationlist** file.

```
#!/bin/sh
:
PATH=/sbin:/usr/sbin:/usr/local/sbin:/bin:/usr/local/bin:/usr/bin

DIR=/usr/share/radio
LOGDIR=${DIR}/logs
LOG=${LOGDIR}/stations.log

mkdir -p ${LOGDIR}
chown pi:pi ${LOGDIR}
sudo ${DIR}/create_stations.py --no_delete 2>&1 >${LOG}
chown pi:pi ${LOG}
```

### *The /etc/cron.daily/radiod script*

Since version 7.3 MPD playlists can be amended by any external MPD client capable of doing so. However, this makes the the **/var/lib/radiod/stationlist** file out-of-date. This daily script updates the **stationlist** file with all the current entries.

```
PATH=/sbin:/usr/sbin:/usr/local/sbin:/bin:/usr/local/bin:/usr/bin

DIR=/usr/share/radio
LOGDIR=${DIR}/logs
LOG=${LOGDIR}/update_stationlist.log
CONFIG=/etc/radiod.conf

mkdir -p ${LOGDIR}
chown pi:pi ${LOGDIR}
grep "update_playlists=yes" ${CONFIG} 2>&1 >${LOG}
if [ $? -eq 0 ]; then
    sudo ${DIR}/update_stationlist.py --update 2>&1 >${LOG}
    sudo ${DIR}/create_stations.py --no_delete --force 2>&1 >${LOG}
fi
chown pi:pi ${LOG}
```



**Note:** These files require the **anacron** package to be installed to run the above scripts regularly.

## A.2 System files modified by the installation

All files to be modified by the installation process are first copied to <filename>.orig.

### /etc/modules

If the i2C interface is installed then the i2c-dev module definition is added to this file. A reboot is required to load the module.

```
 snd-bcm2835  
 i2c-bcm2708  
 i2c-dev  
 # Original file stored as /etc/modules.orig
```

### /boot/firmware/config.txt

This is amended if installing the IR software by adding the **gpio-ir** dtoverlay. For example:

```
dtoverlay= gpio-ir, gpio_pin=25
```

It may also be modified to support **HiFiBerry DAC** and **DAC+**. For example:

```
dtoverlay=hifiberry-dacplus
```

For **IQaudIO** devices the relevant overlay will be specified.

```
dtoverlay=iqaudio-dacplus, unmute_amp
```

The **configure\_audio.sh** script configures the audio device to be used in **/boot/firmware/config.txt** and sets up the correct **/etc/asound** audio configuration. In the case of DAC configuration, it disables the on-board by changing the **dtparam=audio** parameter from:

```
dtparam=audio=on
```

To:

```
dtparam=audio=off
```

## A.3 X-Windows radio desktop files

### The lxsession autostart file for the desktop/touchscreen radio

The **/home/pi/.config/lxsession/LXDE-pi/autostart** file is modified to start either **gradio.py** or **vgradio.py** during the desktop start-up if so configured.

```
@lxpanel --profile LXDE-pi  
@pcmanfm --desktop --profile LXDE-pi  
@xscreensaver -no-splash  
@point-rpi  
@sudo /usr/share/radio/gradio.py
```

### Desktop radio icon files

The configuration files for the two desktop Icons for graphic versions of the radio are copied into the **/home/pi/Desktop** directory. This displays two radio icons on the X-windows desktop. Clicking either of these starts the appropriate desktop version of the radio.

#### The Desktop file **gradio.desktop**

```
[Desktop Entry]  
Name=Radio  
Comment=Internet radio  
Icon=/usr/share/radio/images/radio.png  
Exec=sudo /usr/share/radio/gradio.py  
Type=Application  
Encoding=UTF-8  
Terminal=false  
Categories=None;
```

#### The Desktop file **vgradio.desktop**

```
[Desktop Entry]  
Name=Vintage Radio  
Comment=Vintage Internet radio  
Icon=/usr/share/radio/images/Vintage.png  
Exec=sudo /usr/share/radio/vgradio.py  
Type=Application  
Encoding=UTF-8  
Terminal=false  
Categories=None;
```

## Appendix B – Technical specification and other notes

### B.1 – Technical specification

The specification of the Raspberry PI internet radio is:

- The software runs on **Raspbian Bookworm** (Debian 11) on all Raspberry Pi's except version 1
- 32-bit and 64-bit software available
- Any sudo enable user can install the software and not just user 'pi'
- Uses the standard Music Player Daemon (MPD)
- The following displays are supported:
  - Raspberry Pi 7-inch touch screen or HDMI screen
  - Most 2.8 and 3.5-inch touch screens
  - 2x16, 2x8, 4x16 or 4x20-character LCD
  - Adafruit LCD plate with 5 push buttons (I2C interface)
  - Adafruit 3.5-inch TFT touch-screen
  - A 128 by 64-pixel OLED display
  - PiFace CAD with 2x16 LCD (Version 1.12 only)
  - Pimoroni Pirate Radio with six buttons and a 3W speaker
  - Pimoroni Pirate Audio radio with a 1W mini-speaker and four push buttons.
  - OLEDs using the Sitronix SSD1306 I2C controller
  - OLEDs supported by the LUMA driver (SH1106, SSD1306 etc)
  - Grove 2x16 LCD with RGB backlight (AIP31068L LCD controller)
  - 1.3" 128x64 pixel with SH1106 SPI interface

The LCD can be directly interfaced via GPIO pins or using an I2C backpack interface

- Optional IR sensor and remote-control using IR-KEYTABLE or FLIRC
- Clock display or IP address display
- Five configurable user interfaces are available:
  - Five push button operation (Menu, Volume Up, Down, Channel Up, Down)
  - As alternative to the above, rotary encoders may be used
  - Touchscreen with Mouse/Keyboard interface
  - IQaudIO Cosmic controller with three push buttons and rotary encoder
  - Pimoroni Pirate radio using pHat BEAT sound card and VU indicator
  - PiFace CAD with six push-buttons (Only five are used). Needs version 6.5.
- Support for Russian/Cyrillic, West European, English (Depending upon LCD capabilities)
- Support for Digital sound cards such as **HiFiBerry**, **IQaudIO**, **JustBoom**, **Waveshare WM8960** or **Pimoroni pHat/BEAT**
- Support for Bluetooth speaker or headphones using **pulseaudio-module-bluetooth** software
- Vintage radio conversion to Internet radio supported
- Timer (Snooze) and Alarm functions (Not touch screen version)
- Artist and track scrolling search function
- Plays music from a USB stick, SD card or from a Network drive (NAS)
- Menu option to display a single RSS news feed
- Web interface using Snoopy
- Control the radio from either an Android device or **iPhone** and **iPad**
- Plays Radio streams or MP3 and WMA tracks
- Can function as a **Spotify** receiver (Needs a premium Spotify account)

- Optional support for **Apple Airplay** speaker using **shairport-sync**.
- Play output on PC or on a mobile device using ICECAST streaming
- Playlist creation program using a list of URLs (M3U file)
- Playlist creation from the Shoutcast database via command line or Web interface
- Add and delete stations/tracks via any external MPD client capable of doing so
- Fully integrated with mobile apps such as Android **MPDdroid** or Apple **mPod**
- Speech for visually impaired and blind persons using **espeak**
- Support for Russian/Cyrillic and European character sets (Depending upon LCD capabilities)
- Music Player Daemon supports the **ID3** metadata standard. See  
<https://en.wikipedia.org/wiki/ID3> This provides meta-data such as the title, artist, album, track number, bit-rate, and other information.



Please note that this is not a consumer product. No claims are made to suitability for all users. It is solely intended as a fun construction project. Please also see **Disclaimer** on page **Error! Bookmark not defined..**

## B.2 -Elecrow 7-inch touch-screen notes

Below are some notes on how to set up the Elecrow 7-inch TFT Capacitive touch screen display, with 1024x600 Resolution.



Please note, much of this information was supplied by a third-party and has not been tested by the author. Therefore, any support by the author is limited.

Add the following to **/boot/firmware/config.txt** file.

```
hdmi_force_hotplug=1
max_usb_current=1
hdmi_group=2
hdmi_mode=1
hdmi_mode=87
hdmi_cvt 1024 600 60 0 0 0
hdmi_drive=1
```

If when running in full screen mode (**fullscreen=yes** in **/etc/radiod.conf**) you see a small window surrounded by a thick black border then in **/boot/firmware/config.txt** file amend the following:

```
framebuffer_width=1280
framebuffer_height=720
```

To

```
framebuffer_width=800
framebuffer_height=480
```

Also set the **screen\_size** parameter in **/etc/radiod.conf**.

```
screen_size=800x480
```

Both files must be edited using sudo. For example:

```
$ sudo nano /boot/firmware/config.txt
```

Further information on the Elecrow touch-screen can be found at:

[https://www.elecrow.com/wiki/index.php?title=7\\_Inch\\_1024\\*600\\_HDMI\\_LCD\\_Display\\_with\\_Touch\\_Screen](https://www.elecrow.com/wiki/index.php?title=7_Inch_1024*600_HDMI_LCD_Display_with_Touch_Screen)

### B.3 Sound card DT Overlays

The following table contains the known Device Tree (DT) overlays for various sound cards. The third column contains the DT overlay statement that needs to be added to the **/boot/firmware/config.txt** configuration file. This is either done by running the **configure\_audio.sh** program or by directly editing **/boot/firmware/config.txt**. Some of the DACs require the **pulseaudio** package to be installed.

**Table 11 Sound card Device Tree overlays**

Manufacturer	Sound Card	DT Overlay and dtparam	Pulse
Adafruit	3W Stereo Amplifier Bonnet	dtoverlay=hifiberry-dac	Yes
Allo BOSS	384 kHz/32bit DAC PCM5122	dtoverlay=allo-boss-dac-pcm512x-audio	No
HiFiBerry	DAC+ Light/DAC Zero/MiniAmp	dtoverlay=hifiberry-dac dtparam=i2s:on	No
HiFiBerry	DAC+ standard/pro	dtoverlay=hifiberry-dacplus	No
HiFiBerry	Digi/Digi+ all models	dtoverlay=hifiberry-digi	No
HiFiBerry	Amp+	dtoverlay=hifiberry-amp	No
IQaudIO	Pi-DAC+	dtoverlay=iqaudio-dacplus	
IQaudIO	Pi-DigiAMP+	dtoverlay=iqaudio-dacplus,unmute_amp dtoverlay=iqaudio-dacplus,auto_mute_amp	No
IQaudIO	Pi-DACZero	dtoverlay=iqaudio-dacplus	No
IQaudIO	Pi-DAC PRO	dtoverlay=iqaudio-dacplus	No
IQaudIO	Pi-Digi+	dtoverlay=iqaudio-digi-wm8804-audio	No
JustBoom	Amp, Amp Zero, DAC and DAC Zero	dtoverlay=justboom-dac	No

<b>JustBoom</b>	Digi and Digi Zero	dtoverlay=justboom-digi	No
<b>Pimoroni</b>	pHAT Beat or Pirate Audio	dtoverlay=hifiberry-dac dtparam=i2s=on	Yes
<b>Waveshare</b>	DACs using WM8960 DAC chip	dtoverlay=wm8960-soundcard	No
<b>Various</b>	PCM5102A or PCM5100A based	dtoverlay=hifiberry-dac dtparam=i2s=on	No

In all cases, disable the on-board sound system by modifying the **dtparam=audio=on** parameter in the **/boot/firmware/config.txt** configuration file to off, or by commenting it out.

```
dtparam=audio=off
```

Or

```
#dtparam=audio=on
```

### Configuring other audio devices

For other audio devices, the DT overlays can be found in the **/boot/overlays** directory. See the **/boot/overlays/README** file. For example, to enable the Cirrus WM5102 you would add the following line to the end of the **/boot/firmware/config.txt** configuration file:

```
dtoverlay=rpi-cirrus-wm5102
```

### B.4 UDP messages

This section is only of interest to developers wishing to interface with the radio program. These are messages (events) sent to the UDP listener in the `radio_class.py` program. These are sent from the IR remote control program and from the Shoutcast program & Web interface.

**Table 12** UDP messages

Message	Source	Description
<b>MUTE_BUTTON_DOWN</b>	Remote control	Mute button held down
<b>KEY_VOLUMEUP</b>	Remote control	Not used, same as KEY_RIGHT
<b>KEY_RIGHT</b>	Remote control	Volume up or menu up function
<b>KEY_VOLUMEDOWN</b>	Remote control	Not used, same as KEY_LEFT
<b>KEY_LEFT</b>	Remote control	Volume down or menu down function
<b>LEFT_SWITCH</b>	Radio class	Left switch pressed
<b>RIGHT_SWITCH</b>	Radio class	Right switch pressed
<b>KEY_CHANNELUP</b>	Remote control	Not used, same as KEY_UP
<b>KEY_UP</b>	Remote control	Channel up or menu up function
<b>KEY_CHANNELDOWN</b>	Remote control	Not used, same as KEY_DOWN
<b>KEY_DOWN</b>	Remote control	Channel down or menu down function
<b>KEY_MENU</b>	Remote control	Menu function on remote control pressed

<b>KEY_OK</b>	Remote control	OK key on remote pressed
<b>KEY_LANGUAGE</b>	Remote control	Toggle speech facility on/off
<b>KEY_INFO</b>	Remote control	Toggle info on/off
<b>PLAY_&lt;n&gt;</b>	Remote control	Play station or track <n> e.g. PLAY_23
<b>MEDIA</b>	select_source.cgi script	Cycle through Media playlists
<b>RADIO</b>	select_source.cgi script	Cycle through Radio playlists
<b>AIRPLAY</b>	select_source.cgi script	Select Airplay as source
<b>SPOTIFY</b>	select_source.cgi script	Select Spotify as source
<b>INTERRUPT</b>	select_source.cgi script	Not used (Test only)
<b>RELOAD_PLAYLISTS</b>	shoutcast.cgi script	Reload (new) playlists

## B.5 Cyrillic/European character LCDs/OLEDS

It is possible to purchase LCDs with a Russian/Cyrillic or other languages including Western European character ROMs.



Figure 69 Russian/Cyrillic character LCD

The radio program can display the Russian language either in **Cyrillic** or **Romanized** (convert to Latin) characters. For example, **Радио Пятница** when Romanized becomes **Radio Pyatnica**.

### Romanization of Russian characters

For devices that do not currently support Russian/Cyrillic characters, it is possible to Romanize (convert to Latin characters) Russian text as shown in the following example.

This Romanization is achieved by setting the **romanize** parameter in **/etc/radiod.conf** to **on** which is the default.

```
romanize=on
```

For example, the following Russian text:

```
Низкая цена на нефть все больше давит на рубль
```

Converts to:

```
Nizkaja cena na neft' vse bol'she davit na rubl'
```

Translation (just out of interest):

Low oil price puts more and more pressure on the ruble

For more information on Romanization of Russian characters see:

[https://en.wikipedia.org/wiki/Romanization\\_of\\_Russian](https://en.wikipedia.org/wiki/Romanization_of_Russian)

### Displaying Russian/Cyrillic or European characters

To display the Russian language in native Cyrillic alphabet the following are required:

1. An LCD capable of displaying Russian Cyrillic or Western European characters
2. The **romanize** parameter must be set to **off**
3. The correct code page **number** in the LCD controller must be selected
4. The correct LCD **controller** must be selected.
5. **The translate\_lcd** parameter must be set on unless using OLEDs.

The language in this example is selected by setting the language parameter in **/etc/radiod.conf** to the required language. There are currently only three choices. English (Default), European (Western European) and Russian. In this example Russian has been chosen

```
# Language font translation table to be used.  
# Current choices are English(Default), European(Western) and Russian  
# Translation tables are contained in the /usr/share/radio/codes directory  
# Add other translation tables to the above directory  
language=Russian
```

Font code page selection is achieved by setting the **codepage** parameter in **/etc/radiod.conf**

```
# Select LCD code page table 0,1,2 or 3. Default 0  
# 0 = Use codepage parameter specified in primary font file  
# 1, 2 or 3 Override setting in font file  
codepage=0
```

Setting codepage to 0 tells the radio program to lookup the default code page setting found in the selected code page translation file in the codes sub-directory. Otherwise, this can be overridden by setting the codepage specifically as shown in the following table.

codepage	LCD code page	Language characters
0		Select code page from translation file
1	0x0	Japanese & English
2	0x1	Western European & English
3	0x2	Russian & English

Table 13 Character font table selection

All LCDs have one or more character tables in ROM which provide selection of the correct character font for the character to be displayed. There are usually two code/font ROM pages often known as A0 and A2. However, MC0100 controller devices, for example, may have three. Each table can support only 256 characters, so to support say different language character sets a table must be provided for each one. For example, Midas supply an LCD which supports three languages plus English in each case. These code charts will be found in the controller specification (available from the manufacturer) for the character LCD/OLED device.

In the above table, another manufacturer might use table 0x0 for Russian. This can only be established by looking at the specification for your device.

The final setting for the correct language display is the **controller** parameter in `/etc/radiod.conf`.

```
# Set LCD/OLED controller being used. HD44780U (default) or HD44780 (Older
LCDs)
controller=HD44780U
```

Along with the language parameter the controller parameter selects the code translation files to be used.

The codes sub-directory contains the character to code page PROM translation tables. There are two code for each file. These are called Romanized (Convert to Latin characters) and codes (Native characters).

The actual tables to be selected are dependent upon the **language** and **controller** parameters.

**Table 14 Code page translation files**

Controller	English	Russian	European
<b>HD44780U</b>	European.py Russian.py English.py	Russian.py (codes) European.py (romanized) English.py	European.py (codes) Russian.py (romanized) English.py
<b>HD44780</b>	European_HD44780.py Russian_HD44780.py English_HD44780.py	Russian_HD44780.py (native) European_HD44780.py (romanized) English_HD44780.py	European_HD44780.py (native) Russian_HD44780s.py (romanized) English_HD44780.py

You will notice that **English.py** is always loaded and is last. This means that any codes missed in the other translation files will be caught in **English.py**. The file which relates to the language selection is always the first file and its native codes will be used to display text. The above table can be extended in the future for other languages. The exception is English as all codes will be Romanized.

The `HTMLcodes.py` file found in the **codes** sub-directory is used by the RSS news feeds to translate/strip HTML tags and entities from the RSS feed (in XML). There is also a `README` file in the **codes** sub-directory.



The **translate\_lcd** parameter must also be set to on for Romanization or Cyrillic translation routines to work unless using an OLED display.



The author is not a Russian speaker so testing Russian/Cyrillic character LCDs has relied heavily on Russian and Baltic region constructors to test this and is only able to provide limited support on these types of devices.

Below are the settings for the type of display being used:

**Table 15 Russian Cyrillic and Romanization display configurations**

Type of Display	translate_lcd	romanize_russian
<b>Non-Russian/Cyrillic Latin LCD</b>	on	on
<b>Russian/Cyrillic LCD</b>	off	off

<b>OLED and HDMI displays</b>	n/a	n/a
<b>Raspberry Pi Touch screen</b>	off	off
<b>HDMI Display</b>	off	off

### Character Translation routines

The following is only of interest if you wish to modify the existing translation tables or create your own. The original ASCII code chart for English user only had 256 characters using a single byte. All computers could only use the limited character set at the time and Romanization of character sets such as Cyrillic was the only option. As time went on it was decided to encode these additional characters using two or three bytes. A universal encoding system was invented. The translation routines in the radio software (**translate\_class.py**) convert these characters to a specific character font position in the LCD ROM. For example:

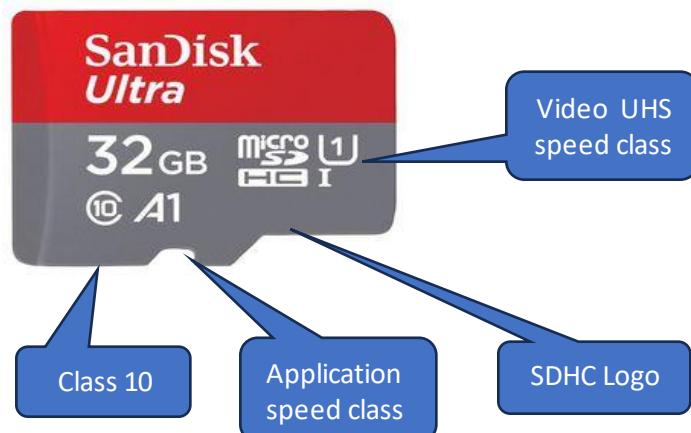
Character	Encoding	Character ROM
Δ	\xd0\xb4	0xe3 (227)

The actual translation tables are contained in the **/usr/share/radio/codes** directory.

## Appendix C - SD Cards for Raspberry Pi

Always use at least a **Class 10 micro SDHC** (Secure Digital High Capacity) card. A **Class 10 SDHC** card is a memory card that can transfer data at a minimum rate of 10 MBytes per second (MB/s) with a capacity of 2GByte maximum. The class of card can be identified from the broken circle with '10' inside it as shown in the figure on the right. You can also use SDXC cards (Secure Digital eXtended Capacity) with capacity up to 2 Terabyte.

Figure 70 Micro Class 10 SDHC card



The "A1" or "A2" rating on an SD card refers to the Application Performance Class 1 or 2 respectively. It is a performance standard that was introduced by the SD Association to indicate that the card meets specific requirements for running applications smoothly and efficiently. In other words, an A1 or A2 rated SD card is optimized for better app performance, making it an ideal choice for devices that rely heavily on running apps, such as smartphones and tablets and in this case the Raspberry Pi. A1 cards are rated at **1500 IOPS** (Input/Output Instructions Per Second) whilst A2 cards are rated at **2000 IOPS**. Both types have a write performance of 10 Mbytes per second.

Each type of card has a memory capacity range, and these are:

- **SD** standard – Up to 2GB SD memory card using FAT 12 and 16 file systems (too small)
- **SDHC** standard – over 2GB-32GB SDHC memory card using FAT32 file system
- **SDXC** standard – over 32GB-2TB SDXC memory card using exFAT file system

So, to sum up choose a **Class 10 micro SDHC card**, **A1** Application speed class and Video Speed Class **UHS 1**.

Instead of the class rating of 1 to 10 you will also find cards rate **V30** to **V90** which are specifically for high-speed video recording. These can also be used with the Raspberry Pi as they support both the **UHS-I** and **UHS-II** SD card interface. However, the current Raspberry Pi models only support the **UHS-I** interface.

The following SD cards are suitable for use with Raspberry Pi:

- SanDisk Extreme Pro
- Samsung Micro SD EVO Plus Series
- Silicon Power 3D Nand Pro Micro SDXC Memory Card
- Raspberry Pi 32GB Preloaded (Noobs) SD Card
- Transcend 400X 64GB MicroSDXC Class10 UHS-1 Memory Card
- SanDisk Ultra 32 GB Micro SDHC Memory Card
- LoveRpi Micro SD Card
- Silicon Power 3D NAND
- Kingston Canvas React
- Samsung Pro Endurance

## Appendix D – Explanation of Vcc, Vdd, Vss and Vee

Throughout this guide the terms **Vcc**, **Vss**, **Vdd** and **Vee** may be used. They all apply to the DC voltage applied to a circuit, component or integrated circuit.

**Vcc** = Voltage Collector Collector

**Vdd** = Voltage Drain Drain

**Vss** = Voltage Source Source

**Vee** = Voltage Emitter Emitter

**GND** = Ground (0 Volts)

**Vcc** and **Vdd** are the positive supply voltage to an IC or circuit.

**Vss** and **Vee** are the negative supply voltage to an IC or electronic circuit.

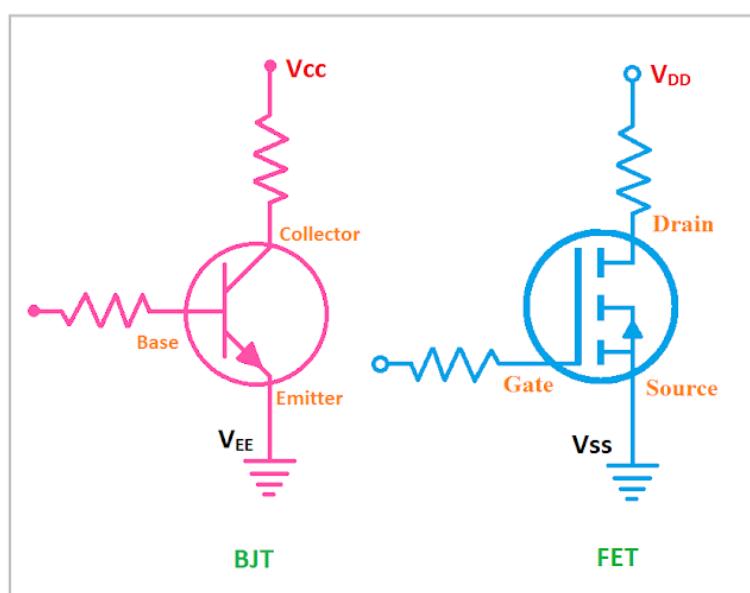


Figure 71 BJT and FET transistors

Which terms are used depends upon the type of circuit being connected to as shown in Figure 71 *BJT and FET transistors*. The terms **Vcc** and **Vee** are invariably used the most. **Vee** and **Vss** are normally tied to the 0v rail but as in the case of some LCDs can actually be a negative voltage.

Table 16 Supply voltage definitions summary

Supply Voltage	BJT (Bipolar Junction Transistor)	FET (Field Effect Transistor)
Positive Supply Voltage	Vcc	Vdd
Negative Supply Voltage	Vee	Vss

## Appendix E Raspberry Pi Model 1 wiring

### Raspberry Pi wiring

Table 17 and **Error! Reference source not found.** on the following pages 314 and **Error! Bookmark not defined.** respectively show the interface wiring for both the push button and rotary encoder versions of the radio. There are two versions of the wiring, 26 and 40-pin versions (Table 17 and **Error! Reference source not found.** respectively). The connections used by the radio are highlighted in yellow. The **IQaudIO** or **JustBoom** and newer **HiFiBerry** DACs require 40-pin versions of the Raspberry Pi. The 40-pin version of the wiring is recommended for all new projects.



If using DAC products do not use the 26-pin version of the wiring but use the wiring shown in **Error! Reference source not found.** on page **Error! Bookmark not defined..**

The table below is now redundant except for very early versions of the radio.

**Table 17 Controls and LCD wiring 26 pin version**

Pin	Description	Radio Function	Name	LCD pin	Push Buttons	Encoder (Tuner)	Encoder (Volume)
1	3V3	+3.3V supply			COMMON		
2	5V	5V for LCD		2,15			
3	GPIO2	I2C Data*	I2C Data				
4	5V						
5	GPIO3	I2C Clock*	I2C Clock				
6	GND	Zero volts		1,3*,5,16		Common	Common
7	GPIO 4	Mute volume			MUTE		Knob Switch
8	GPIO 14	Volume down	UART TX		LEFT		Output A
9	GND	Zero Volts					
10	GPIO 15	Volume up	UART RX		RIGHT		Output B
11	GPIO 17	Channel Up			UP	Output B	
12	GPIO 18**	Channel Down			DOWN	Output A	
13	GPIO 27	LCD Data 4		11			
14	GND	Zero Volts					
15	GPIO 22	LCD Data 5		12			
16	GPIO 23	LCD Data 6		13			
17	3V3	+3.3V supply					
18	GPIO 24	LCD Data 7		14			
19	GPIO 10**	Channel Down	SPI-MOSI		DOWN	Output A	
20	GND	Zero Volts					
21	GPIO 9	IR Sensor in (1)	SPI-MOSO				
22	GPIO 25	Menu Switch			MENU	Knob Switch	
23	GPIO 11	IR LED out (1)	SPI-SCLK				
24	GPIO 8	LCD E	SPI-CEO	6			
25	GND	Zero Volts					
26	GPIO 7	LCD RS	SPI-CE1	4			

Colour Legend Radio I2C (shared) SPI Interface

\* These pins are used for the I2C LCD backpack if used instead of the directly wired LCD to GPIO pins.

\*\* Pin 18 is used by some DACs so pin 10 should be used for the down switch.



Note: Make sure you are using the correct columns in the above wiring tables. Use column 6 (Push Buttons) for the push button version and the last two columns (Encoder Tuner/Volume) for the rotary encoder version. Also note that the configuration in `/etc/radiod.conf` uses **GPIO** numbers and not physical pin numbers. For example, `menu_switch=17` is using **GPIO 17** (Physical pin 11).

## LCD Module Wiring

The following shows the wiring for a directly wired HD44780U LCD controller. It has 16 or 18 pins. There are two ways of wiring the LCD data lines using either the 26 pin or 40-pin wiring schemes (Table 17 and **Error! Reference source not found.**, respectively). For all new 40-pin Raspberry Pi's the 40-pin wiring is strongly recommended. The 26-pin version of the wiring can be used on both 26 and 40-pin Raspberry Pi's.

**Table 18 LCD module wiring for 26 and 40-pin Raspberry Pi's**

LCD Pin	GPIO	Pin	GPIO	Pin	Description
	26-pin	26 #	40-pin	40 #	
1	n/a	6	n/a	6	Ground (0V) – Wire this directly to LCD pin 5
2	n/a	2	n/a	2	VCC +5V
3	n/a	Note1	n/a	Note1	Contrast adjustment (0V gives maximum contrast)
4	GPIO7	26	GPIO7	26	Register Select (RS). RS=0: Command, RS=1: Data
5	n/a	6 or 9	n/a	6 or 9	Read/Write (RW). Very important this pin must be grounded! R/W=0 (GND): Write, R/W=1 (+5V): Read. Will damage the PI if not grounded (0V). Wire LCD pin 5 and 1 together
6	GPIO8	24	GPIO8	24	Enable (EN)
7					Data Bit 0 (Not required in 4-bit operation)
8					Data Bit 1 (Not required in 4-bit operation)
9					Data Bit 2 (Not required in 4-bit operation)
10					Data Bit 3 (Not required in 4-bit operation)
11	GPIO27	13	GPIO5	29	Data Bit 4 (D4)
12	GPIO22	15	GPIO6	31	Data Bit 5 (D5) Note if using IQaudIO products GPIO22 conflicts !!
13	GPIO23	16	GPIO12	32	Data Bit 6 (D6)
14	GPIO24	18	GPIO13	33	Data Bit 7 (D7)
15	n/a	2	n/a	2	LED Backlight Anode (+5V) or Red LED (Adafruit RGB plate) [2] or VEE negative voltage on Midas HD44780
16	n/a	6 or 9	n/a	6 or 9	LED Backlight Cathode (GND) or Red LED [2]
17					Optional Green LED (Adafruit RGB plate) [2]
18					Optional Blue LED (Adafruit RGB plate) [2]

## Version 1 boards push-button wiring

Version 1 boards only had 26 pins and did not have internal pull-up resistors on the GPIO inputs. It has become increasingly difficult to support version 1.0 boards and you are advised to purchase a newer Raspberry Pi board for this project. Tips for using version 1 boards will be retained in this manual; however, if there is a problem, regrettably no support can be provided. As shown in the

diagram below, wire one side of the switches to the 3.3V pin. Wire the other side of each switch to the GPIO pin as shown in the last column of the above table. Also wire this same side of the switch to the GND(0V) pin via a 10KΩ resistor. See Figure 72 below.

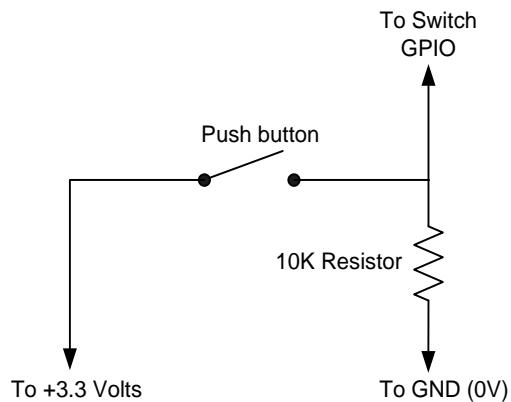


Figure 72 Push-button Wiring version 1 boards

# Index

AAC, 235  
activity LED, 127, 128  
Adafruit, 124, 127, 130, 131, 165, 192, 232, 234, 250  
AdaFruit, 127, 128  
AdaFruit RGB plate, 127  
airplay, 21, 208, 211  
Airplay, 21, 22, 23, 211, 251  
Alarm, 250  
Allo BOSS DAC, 252  
**alsamixer**, 136, 146  
**anacron**, 247  
**aplay**, 31, 32, 161, 171, 246  
**asound.conf**, 32, 135, 136, 246  
ASX, 235  
backup, 68  
**Bitvise**, 72  
Bluetooth device, 7, 178, 181  
Bluetooth speakers, 7  
bluetoothctl, 7, 190  
Bookworm, 26, 35, 39, 46, 50, 56, 60, 71, 83, 87, 219, 237  
**Bookworm Lite**, 26, 35, 39, 46, 50, 56, 60, 83, 87, 219  
CAD, 127, 235  
CGI, 235  
CIFS, 235  
CODEC, 133  
colours, 130  
Comitup, 119  
*configure\_radio.sh*, 126, 129, 157, 159, 164, 165, 207, 209, 240  
Cosmic controller, 129, 207, 208, 250  
Cyrillic, 124, 141, 142, 250, 254, 255, 256, 257  
Cyrillic character LCD, 141, 142, 254  
**DAC**, 32, 126, 161, 172, 181, 183, 202, 209, 236, 248, 260  
daemon, 26, 35, 39, 46, 50, 56, 60, 64, 75, 83, 87, 131, 161, 163, 164, 165, 183, 195, 200, 206, 209, 210, 219, 224, 226  
date format, 74, 76, 128, 129  
**DDOS**, 70  
Device Tree, 252  
DHCP, 235  
DNS, 235  
**dpkg**, 29, 42  
DSP, 235  
DT. See Device Tree  
Elecrow, 251, 252  
Electromagnetic Interference, 164, 236  
electronic ink displays, 77  
EMI, 164, 236  
equalizer, 134, 136  
espeak, 30, 31, 32, 208, 232, 251  
eSpeak, 238  
**fail2ban**, 71, 72  
FBCP, 53  
**Fing**, 157  
**firewall**, 71  
firmware, 21  
Fonts configuring, 144  
**fsck**, 149, 150  
**FTP**, 70  
GPIO, 127, 128, 192, 193, 236, 250, 260, 261  
GPIO pins, 193, 250, 261  
GPIOconverter, 24, 63, 168, 169, 205, 213  
hciuart, 191  
HD44870, 206  
HDMI, 31, 149, 161, 209  
HiFiBerry, 32, 126, 161, 172, 202, 209, 248, 250, 260  
hostname, 231  
I2C, 232, 236, 250, 260, 261  
I2C interface, 250  
Icecast2, 65, 66, 224, 225, 226, 227, 229, 230, 231, 232, 234  
*install\_airplay*, 21  
Internet Security, 70  
*ioe-python*, 169  
*ioexpander*, 169  
iPad, 250  
iPhone, 250  
**iptables**, 71, 72  
**iptables-persistent**, 73  
*iptables-save*, 73  
IPv4, 236  
IPv6, 163, 236  
IQAudio, 27, 32, 36, 40, 47, 51, 57, 61, 84, 88, 124, 126, 129, 134, 135, 136, 207, 208, 209, 220, 248, 250  
IR, 210, 236, 248, 250, 260  
IR Sensor, 260  
**ireventd.service**, 19, 211  
**ir-keytable**, 14, 15, 16, 17, 18, 211  
**irreventd.service**, 211, 246  
JustBoom, 250, 260

language file, 32, 132, 133  
LCD, 74, 76, 124, 130, 131, 164, 165, 166, 192, 193, 208, 226, 230, 232, 234, 236, 250, 251, 260, 261  
LIRC, 236  
M3U, 170, 230, 236, 251  
MAC, 237  
metadata, 251  
MP3, 250  
MPC, 161, 237  
**mpd**, 25, 34, 38, 45, 49, 55, 59, 60, 64, 76, 82, 86, 116, 161, 163, 166, 170, 183, 195, 209, 218, 224, 230, 231, 232  
MPD, 25, 34, 38, 45, 49, 55, 59, 64, 66, 75, 76, 82, 86, 131, 161, 163, 183, 195, 209, 218, 223, 224, 225, 231, 232, 237, 250  
MPD Upgrading, 116  
**MPDroid**, 251  
MPEG, 237  
MPEG3, 236, 237  
NAS, 237, 250  
Network Time Protocol, 237  
news feed, 134, 250  
NFS, 171, 237  
NTP, 237  
OLED, 129, 207, 233, 237, 250  
Olimex Limited, 207, 233  
OS, 237  
Parameters  
  \*\_color, 130  
  client\_timeout, 132  
  CODECS, 133  
  codepage, 142  
  controller, 142  
  dateformat, 128  
  display\_lines, 144  
  display\_type, 206  
  display\_width, 144  
  down\_switch, 126  
  exit\_action, 128  
  flip\_display\_vertically, 129  
  font\_name, 144  
  font\_size, 144  
  i2c\_bus, 144  
  internet\_check\_port, 143  
  internet\_check\_url, 143  
  language, 141, 142  
  lcd\_data4, 127  
  lcd\_data5, 127  
  lcd\_data6, 127  
  lcd\_data7, 127  
  lcd\_enable, 127  
  lcd\_select, 127  
  left\_switch, 126  
  menu\_switch, 126  
  menu\_switch\_value\_\*, 145  
  mpdport, 143  
  mute\_action, 134  
  mute\_switch, 126  
  pull\_up\_down, 127  
  remote\_control\_host, 143  
  remote\_control\_port, 143  
  remote\_led, 127  
  remote\_listen\_host, 143  
  rgb\_green, 129  
  rgb\_red, 129  
  right\_switch, 127  
  romanize, 142, 254  
  rotary\_class, 145  
  rotary\_gpio\_pullup, 145  
  scroll\_speed, 141  
  shutdown\_command, 128  
  speak\_info, 31  
  speech, 30  
  speech\_volume, 30  
  startup, 130  
  streaming\_on, 66, 225  
  translate\_lcd, 256  
  up\_switch, 126  
  verbose, 30  
  volume\_display, 131  
  volume\_range, 131, 132  
Parameters [AIRPLAY]  
  airplay, 22  
  mixer\_preset, 21, 184  
Parameters [SCREEN]  
  banner\_color, 126  
  dateformat, 126  
  display\_date, 126  
  display\_icecast\_button, 230  
  display\_mouse, 126  
  display\_title, 126  
  display\_window\_color, 126  
  display\_window\_labels\_color, 126  
  fullscreen, 126  
  labels\_color, 126  
  scale\_labels\_color, 126  
  slider\_color, 126  
  stations\_per\_page, 126  
  wallpaper, 126  
  window\_color, 126  
  window\_title, 126

PC, 29, 42, 64, 65, 96, 134, 224, 227, 228, 230, 231, 237, 251  
PC speakers., 228  
PCF8574, 232  
PCM, 32, 237  
pHat BEAT, 26, 28, 33, 35, 37, 39, 41, 46, 48, 50, 52, 56, 58, 61, 62, 83, 84, 85, 87, 88, 89, 219, 220, 221  
**Phishing**, 70  
PID, 237  
PiFace, 235  
**PiFace CAD**, 5, 175, 179, 238, 250  
**PIL**  
    See Pillow, 168  
Pillow, 168  
**Pimoroni pHat**, 253  
Pimoroni Pirate radio, 33  
Pirate Audio, 43, 44  
PLS, 161, 237  
potentiometer, 164  
pulseaudio, 26, 27, 35, 36, 39, 40, 46, 50, 56, 61, 83, 84, 87, 88, 161, 165, 168, 219, 220  
**Putty**, 72  
PWM, 102  
**radiod** package, 101  
**radiod.conf**, 30, 77, 127, 128, 130, 131, 201, 208, 210, 240  
Random, 76  
Rasbian package, 101  
Raspberry PI, 1, 3, 29, 42, 66, 67, 127, 164, 183, 198, 218, 226, 230, 231, 232, 234, 236, 250  
Real Time Clock, 79, 104, 185, 238  
RealVNC, 107  
Red Blue Green LED, 210  
remote control, 32, 124, 127, 128, 201, 210, 250  
**Rexec**, 70  
rfkill, 156  
Romanize, 254  
Romanized, 141, 142, 254, 256  
rotary encoder, 77, 131, 192, 250, 260, 261  
RSS, 124, 133, 134, 208, 238, 250  
RTC. *See* Real Time Clock  
Russian, 124, 141, 142, 250, 254, 255, 256  
Russian, 141  
Russian, 254  
SD, 238  
SD card, 68  
Secure Shell. *See* SSH  
**service radiod**, 133, 201  
SH1106, 28, 37, 41, 48, 52, 58, 59, 63, 85, 89, 206, 221, 250  
**SH1106 SPI**, 52  
shairport-sync, 21, 22, 211, 251  
softvol, 146  
SoftVol, 124, 146  
Solid State Drives, 81  
**sourceforge**, 31, 32  
speech, 30  
**speech\_volume**, 30  
SPI interface, 5, 175, 179  
Spotify, 250  
SSD, 81, 95  
SSD1306, 28, 37, 41, 43, 48, 52, 58, 62, 63, 85, 89, 206, 207, 221, 233, 250  
**SSH**, 70, 238  
SSID, 238  
systemd, 245, 246  
TCP/IP, 210, 238  
**Telnet**, 70  
TFT, 53, 54, 238, 250  
timeout, 132  
touch screen, 250  
UDP, 210, 212, 238  
URL, 64, 75, 133, 134, 170, 224, 230, 231, 236, 238  
USB, 181, 183, 232, 238, 250  
USB disk drive, 80, 81, 93, 96  
USB stick, 250  
Vcc, 259  
Vdd, 259  
Vee, 259  
version 1.0 boards, 261  
vintage radio, 128, 210, 211  
Vintage radio, 127, 250  
VNC, 107  
Vss, 259  
Waveshare, 77  
**weather.py**, 214  
Web interface, 74, 75, 76, 250  
WEP, 239  
**wget**, 29, 42  
WIFI, 239  
Wi-Fi signal strength, 153  
Win32DiskImager, 68  
wiring, 128, 164, 165, 169, 260  
WM8960, 253  
WMA, 250  
WPA, 239  
WPA2, 239  
XML, 239

**xscreensaver**, 118