Raspberry Pi Internet Radio

Constructors Manual

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

Bob Rathbone Computer Consultancy

www.bobrathbone.com

24th of December 2024

Version 8.0

Contents

About this manual

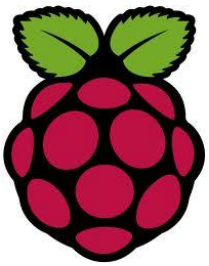
To do

Add reference to the Raspberry Pi Internet Radio Technical Manual

Quick Links

Please note that there is also a full document index on page 233 at the end of this document.

Chapter 1 - Introduction



This manual describes how to create one of the most popular Internet Radios using the Raspberry PI educational computer. This manual provides a detailed overview of construction and software installation.

The source and basic construction details are available from the following Web site:
https://bobrathbone.com/raspberrypi/pi_internet_radio.html

The main features of the Raspberry PI internet radio are:

Raspberry Pi Internet Radio

Turn your Raspberry Pi into an Internet Radio using a variety of designs as shown in this manual. It runs on both 32 and 64-bit systems. It also runs on an X-Windows Desktop.

Media Player

Play your favourite MP3 tracks from a USB stick, SD card or from a NAS (Network Attached Storage).

Airplay Receiver

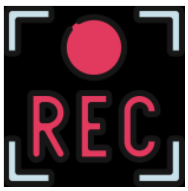
This design allows the Raspberry Pi to act as an Airplay receiver. Music tracks can be played from your Apple or Android mobile phone or tablet.

Spotify Receiver

Turn your Raspberry Pi into a Spotify Receiver. This requires a Spotify Premium account.

RSS Feed Reader

This software also allows you to read any configured RSS feed. For example, your favourite news feed.



Record Radio Streams to disc

Record any Radio streams using the **streamripper** utility to record any streams that are using the **Shoutcast** or **Icecast** formats whilst also listening to a different radio station.

Bluetooth speaker/headphone support

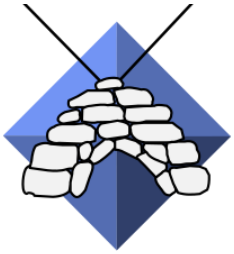
This manual contains instructions how to run the radio software with Bluetooth speakers or headphones. Bluetooth versions 1.x through 5.x supported depending upon the Raspberry Pi model used.

Digital Clock

The Internet Radio software displays as standard a digital clock and date with alarm and snooze functions.

Shoutcast

Shoutcast radio is a streaming audio which is used by some 50,000 Internet Radio stations across the internet. This radio software, using a Web interface, allows multiple playlist creation from Shoutcast radio stations by genre or country which can then be selected through the radio or Web interface menus.



Icecast

Icecast is free streaming software which supports a variety of streams such as MP3 and OGG. Icecast can optionally be installed on the Raspberry Pi and allows the currently playing station or track to be streamed around the local network or out to the Internet (Legal and Copyright issues apply).

User Interface and displays

This design caters for a number of user interfaces such as push-buttons, rotary encoders or touch screens. Also, a number of displays such as 2 and 4-line LCDs, OLED displays, TFT or full graphical displays such as HDMI and touch-screen are supported. Examples are shown later on in this manual.

Web interface

The radio software includes an optional Web interface (radioweb) powered by **Apache 2**, **MySql** and **O!MPD**. This allows stations and playlists to be selected via Web pages on your PC, Mobile Telephone or tablet.

The **eSpeak** engine is a small, lightweight text-to-speech (TTS) program that supports a large number of languages. It is used with the radio software to assist blind or visually impaired users by “speaking” the menus out loud. It can also be used with radios without a screen to navigate the menus.

Language files enable the messages to be displayed in the user's own language, such as French, Dutch or Spanish. The user can easily create their own language file if required.

Highly configurable allowing a broad range of displays such as two or four-line LCDs, TFTs, graphic screens or HDMI monitors to be selected at installation time. Likewise, the user input devices such as rotary encoders or buttons etc. An audio configuration script selects a range of audio devices such as DACs or Bluetooth speakers.

Comprehensive menu driven installation and maintenance. All main software components and hardware options are all menu selectable. Underlying installation and configuration scripts carry out all the usual installation and maintenance tasks.

Support for English, Russian/Cyrillic and Western European character set, dependent upon LCD capabilities. Both native and Romanized (Conversion to Latin) characters supported.

A full specification can be found in the **Raspberry PI Internet Radio Supplement**

REMEMBER TO HAVE FUN DOING THIS PROJECT!

Figure 1 Fun radio using an old toaster (Courtesy Robert Knight)

This manual is continually being updated. Please check to see if there is a more up to date manual before commencing any work.

Is the size of this manual worrying you then try starting with *Raspberry Pi Internet Radio, A Beginners Guide* at:
https://bobrathbone.com/raspberrypi/documents/RPi%20Radio%20beginners%20guide.pdf

Examples

This design caters for both the complete novice and more advanced constructors. Do not be put off by the size of this manual as it shows a lot of different designs. Simply read through the following examples and decide which one is the best for you. Some examples are shown in the following pages. This manual is designed to provide inspiration for your own ideas and unique solution to building an Internet Radio using the Raspberry Pi.

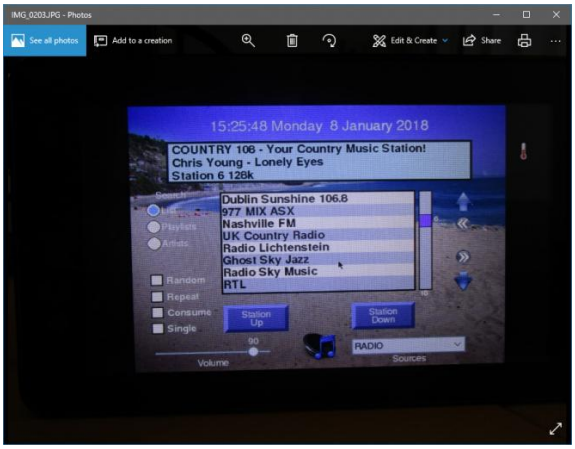


Figure 2 Raspberry pi 7-inch touchscreen radio

The Radio software supports the Raspberry Pi 7-inch touch screen. Using the graphic version of this software, the radio can be operated using the touch screen or a mouse and HDMI screen or TV with HDMI. A keyboard can also be attached and used to operate the radio. The touch screen version supports the same functionality as the LCD versions of the radio except for timer and alarm functions. The touch screen can also be configured to use either rotary encoders or buttons.

Figure 3 HDMI Television running the radio

The HDMI/Touch screen version of the radio can also be configured to run in a window on the Raspberry Pi desktop. Here it is running on the HDMI input of a typical flat-screen television. It can also be configured to use an IR remote control using a FLIRC USB IR detector.

Figure 4 Vintage tuning touch-screen radio

As an alternative to the above design this touch-screen radio is made to look like a vintage radio with a tuning dial. The green slider marks the currently playing station. When a station name is touched on the screen then the slider jumps to that position and plays the selected radio station. The design supports multiple pages of radio stations which can be scrolled left or right. The volume control slider is at the bottom of the screen. This version currently only plays radio stations and not media or Airplay. The touchscreen can also be configured to use either rotary encoders or buttons.

Figure 5 Adafruit 3.5-inch TFT

This example shows an Adafruit 3.5-inch TFT (Thin Film Transistor) touch-screen running the graphical version of the software (Version 6.7 onwards). This is the smallest screen that is currently supported.

Installation of the Adafruit TFT touchscreen is found in the section called **Error! Reference s ource not found.** on page **Error! Bookmark not defined.**.

Figure 6 Radio using the Adafruit LCD plate

Example of the PI internet radio using an Adafruit RGB-backlit LCD plate for Raspberry PI from AdaFruit industries. It has five push buttons and is one of the easiest options to construct. If you want to build this into a case then don't use the buttons supplied with the kit but use external buttons.

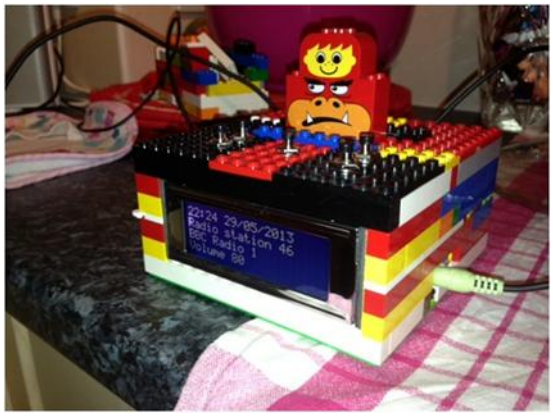


Figure 7 Lego Internet Radio

Example of a fun radio built using this design and Lego from Alan Broad (United Kingdom). This really puts the fun back into computing.

Figure 8 Pi radio using rotary encoders

The rotary encoder switch version of the radio consists of a Raspberry PI connected to an Adafruit 20-character x 4-line RGB LCD display housed. It is all housed in a LogiLink PC speaker set with two rotary encoders. The rotary encoders also have push buttons (Push the knob in). The left one is the *Mute* switch and the right one is the *Menu* switch. The blue glow in the sub-woofer opening comes from a bright blue LED.

Figure 9 Old Zenith radio using rotary encoders

Example of the PI radio from Jon Jenkins built into an old Zenith valve radio case. The two original controls have been replaced by two rotary encoders. The old valve radio inside has been completely removed and replaced with the Raspberry PI and radio components. The LCD display has been built into the top so as not to spoil the original face of the radio. This is a fine example of what can be done with this project.

Figure 10 Transparent Perspex Radio

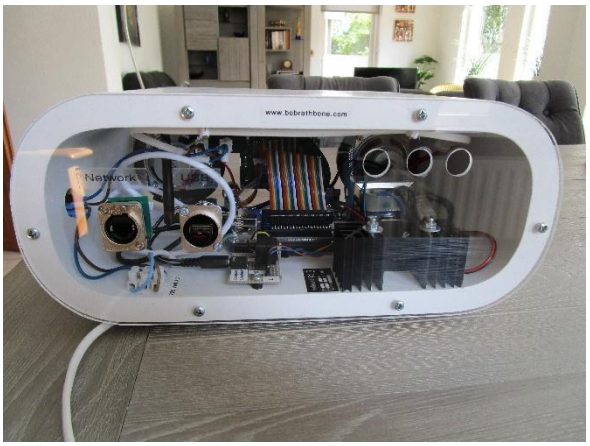


Figure 11 Perspex radio rear view

The above example built by the author has a transparent perspex front and back panel. It uses a Raspberry Pi with a HiFiBerry sound card and a Velleman 30 watt amplifier.

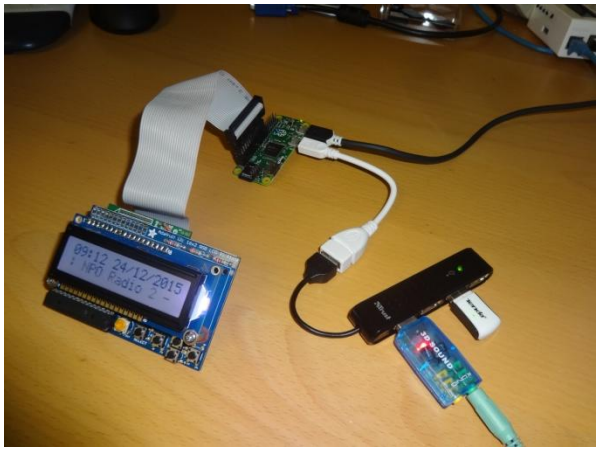


Figure 12 The Radio running on a Pi Zero

This is an example of the radio running on a Raspberry Pi Zero. In this example it uses a micro to standard USB adaptor to connect a simple USB hub. A USB sound dongle and Tenda wireless adapter are plugged into the USB hub. A USB to Ethernet adapter can also be used in place of the wireless adapter. The display used is the Adafruit LCD plate. Also note that the Pi Zero comes with an unpopulated 40 pin GPIO interface. You need to either directly connect directly wires to the GPIO interface (Not advised as they keep falling off) or solder onto a 40-pin male extender header (Advised).

This beautiful radio is a fine example built by the author. It is using a Raspberry PI model 2B and rotary encoders with inbuilt push button. The display is a 4 x 20 LCD. The sound system is a Velleman 30-Watt amplifier (bottom right) and two 5 ¼ inch 50-watt speakers. It has an IR sensor (Left speaker on the right side) and an activity LED (between the two knobs).

Figure 13 Boom Box radio front view

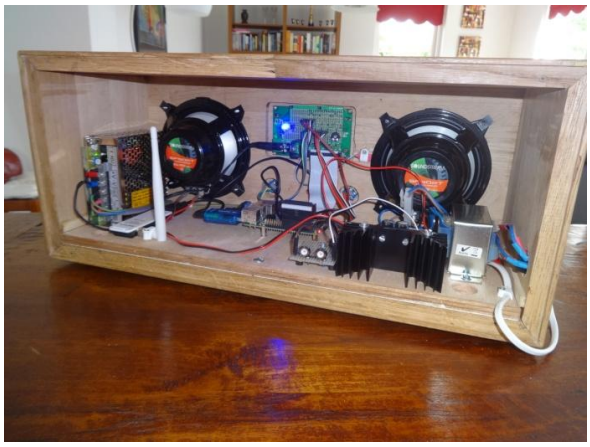


Figure 14 Boom Box Radio rear view

Below is a Raspberry Pi radio built into a old wine box. It uses a 2x8 character LCD and rotary encoders. The amplifier and loud speakers are from a set of old PC speakers.

Figure 15 Raspberry Pi Wine Box radio

Figure 16 Wine box internet radio internal view

Figure 17 Very small radio using the Cosmic Controller

Here is a really cute radio made using the IQaudIO Cosmic Controller and Olimex 128x64 pixel OLED display. The Cosmic Controller provides an excellent solution where space is limited or you simply want a very small radio. The audio output is on the rear of the case.

Note: The Cosmic Controller has been discontinued and is difficult to find but you can still build the radio with separate components.

Below is a fascinating use of both modern and bygone era technology. The radio shown below was created by Broesel from Austria. In this design a Raspberry Pi has been used with the software described in this manual. However, the audio amplifier has been constructed with an ECL84 vacuum valve. The ECL84 valve provides a two-stage mono audio amplifier driving an elliptical wide frequency response loud speaker. Broesel has very kindly provided the full construction details at the Radio Board Forum. See: http://theradioboard.com/rb/viewtopic.php?t=6314

Figure 18 RPI radio with two-stage valve amplifier

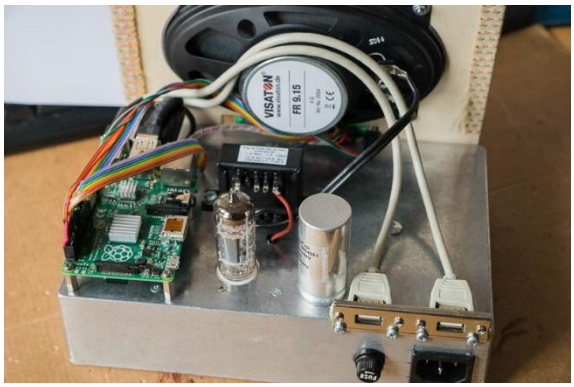


Figure 19 The RPi valve radio chassis view

Figure 20 Pimoroni Pirate Radio

The radio software supports the Pimoroni Pirate radio with pHat BEAT.

The pHAT BEAT gives high-quality, digital, amplified, stereo or mono audio and 16 RGB LEDs, in two rows of 8, which are ideal as a VU (Volume Unit) indicator, and 6 buttons to control the radio (Five on the left and one at the top). See: https://shop.pimoroni.com/products/pirate-radio-pi-zero-w-project-kit
It can also be ordered with an additional speaker.

Figure 21 PiFace CAD Radio with IR Remote Control

The radio supports the PiFace Control and Display (CAD) board. This is a good choice for complete beginners but is quite slow. See: http://www.piface.org.uk/products/piface_control_and_display/

The PiFace CAD has 5 push buttons, a reset button (not currently used) and an Infra Red (IR) sensor for a remote control. It has one drawback in that the push buttons are on the bottom of the unit. The PiFace CAD uses the Serial Peripheral Bus interface (SPI) on the Raspberry Pi.

Below is an example of a Raspberry Pi Internet radio, made by the author, running the graphic version of the radio which gives a vintage look-and-feel to the final radio. Two vintage Bakelite knobs help complete the vintage appearance.

Figure 22 Vintage look-and-feel Internet radio

The Raspberry Pi model 3B outputs to a Pimoroni 3W stereo D-Class amplifier driving two 3-inch loudspeakers. It is housed in a wooden case painted black with two vintage radio Bakelite knobs.

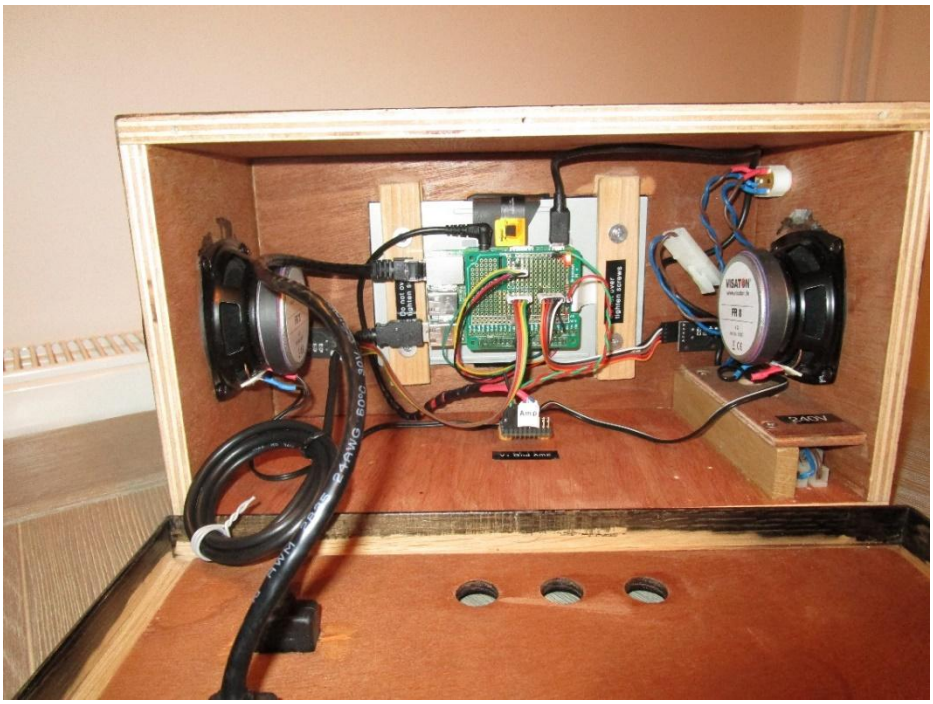


Figure 23 Vintage look-and-feel - rear view

Vintage Radio Conversion

This version of the software allows for the program to be configured without a screen for use with a vintage radio as shown below:

Figure 24 Philips BX490A (1949) Vintage Internet Radio

The radio is a Philips BX490A manufactured in the Netherlands in 1949. The purpose of this design is retain as much of the original look and feel of a vintage radio which has been converted to run as an Internet radio. It does not have any LCD display. In the above example the following controls are used:

- Far left switch - Simple tone control
- Middle left switch - Volume and mute switch
- Middle right switch – Radio channel (Tuner) or media track selection
- Far right switch – Menu switch (8 positions)
- Push button on right side (Not shown) - Standard menu switch

At the top left the so-called magic eye tuning indicator has been replaced with a Red,Green,Blue status LED. In the above picture the LED is glowing green (Normal operation). This window also contains the IR sensor and activity LED for a remote control. If the radio is busy (loading stations for example) it glows blue. For an error or shutdown the LED glows RED. The IR remote control also flashes red to indicate IR remote control activity.

The software allows espeak to be configured to 'speak' station and search information etc.
The details on how to construct a similar project is contained in the following documents:
http://www.bobrathbone.com/raspberrypi/documents/Raspberry%20PI%20Vintage%20Radio.pdf

http://www.bobrathbone.com/raspberrypi/documents/Raspberry%20PI%20Vintage%20Radio%20Operating%20Instructions.pdf

The illustration below shows a French Radio Schneider Frères Rondo from the 1950's which has been converted to an internet radio by Franz-Josef Haffner, from Germany.

Figure 25 Vintage radio using a touch screen

What makes this project also very interesting is that he has removed all of the RF section of the valve radio leaving only the audio amplifier and power supply. This is an excellent example of combining old and new technology to extend the life of these increasingly rare radios.
See the Vintage Radio supplent for further details using the following link:
https://bobrathbone.com/raspberrypi/documents/Raspberry%20PI%20Vintage%20Radio.pdf

Figure 26 Vintage Radio with AM transmitter

Figure 27 Lusya DIY 3-channel AM transmitter

Another way to connect to a vintage radio (if it is working) is to use a Raspberry Pi and a low power AM transmitter. In the above illustration, the box on the right contains a Pi Zero W running the radio software with a tuning knob. The audio comes out of a USB DAC which connects to the Lusya 3-channel AM transmitter which transmits on the 300 Meter wavelength (AM waveband) to a vintage radio. In this example the radio is the British made **Bush DAC 90A** medium/longwave receiver.
Details on construction can be found in the following document:
https://bobrathbone.com/raspberrypi/documents/Raspberry%20PI%20Vintage%20Radio.pdf

Chapter 2 – Hardware components overview

Principle components

The principal hardware required to build the radio consists of the following components:

- Current versions of the Raspberry PI computer (Version 1 boards no longer supported)
- Push buttons or rotary encoders for the user interface
- A display such as a HD44780U LCD, OLED or a Raspberry PI touch-screen display
- Optional Digital to Analogue Converter (DAC) Sound card
- Optional InfraRed (IR remote control)

Raspberry Pi computer

The **Raspberry Pi** is a credit-card-sized single-board computer developed in the United Kingdom by the Raspberry Pi Foundation originally with the intention of promoting the teaching of basic computer science in schools. It has however become immensly popular with hobbyists and engineers. The official site for the Raspberry Pi Web site is https://www.raspberrypi.com

More information on the Raspberry PI computer may be found here:
http://en.wikipedia.org/wiki/Raspberry_Pi
If you are new to the Raspberry PI try the following “Getting started” guide at:
https://www.raspberrypi.com/documentation
You will also find the Raspberry Pi Help Forums and other useful documentation on the above page.

Raspberry Pi model 4B

The Raspberry Pi model 4B was released in June 2019.

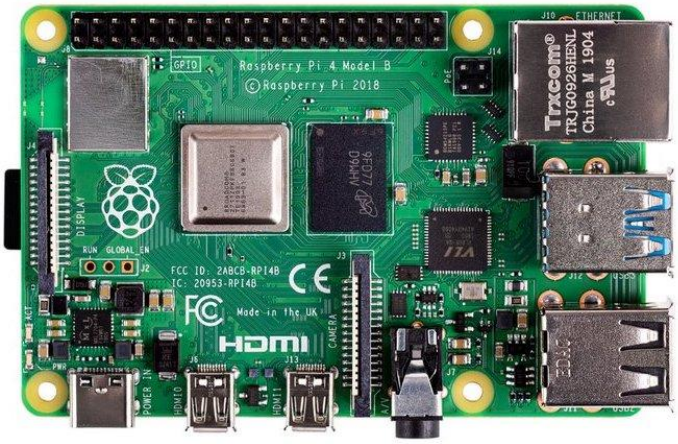


Figure 28 Raspberry Pi Model 4B Computer

Note: The Raspberry Pi 4B requires **Raspberry Pi Bullseye** or **Bookworm** SD card. It will not work with earlier OS versions. However, the recommended OS for the Radio software is **Bookworm**.

Figure 29 USB-C plug

The power supply on the model 4B uses a USB-C connector. The usual micro-USB power supply will not fit used for other Raspberry Pi models will not fit. The USB-C specification allows the cable to be inserted either way around. When purchasing the model 4B also purchase the official model 4B power supply which is 5 Volt 3 Amps.

Figure 30 Rasberry Pi MicroHDMI cable

The model 4B also requires an official Raspberry Pi Micro-HDMI cable for each of the micro-HDMI ports (This is not the same as the Pi Zero HDMI adapter). Order one or two of these adaptors if using an HDMI or TV screen.

Audio and video output jack

Earlier versions of the Raspberry Pi have a separate audio output jack and composite video output. Later versions (3 and 4) of the Raspberry Pi have a new AV (Audio/Video) port which combines the audio and video signals in a single jack. Instead of using a standard composite cable, this new connector requires a 4 pole 3.55mm AV cable. To complicate matters: not all of these cables are the same! However, existing audio jack plugs are compatible with the new AV connector.

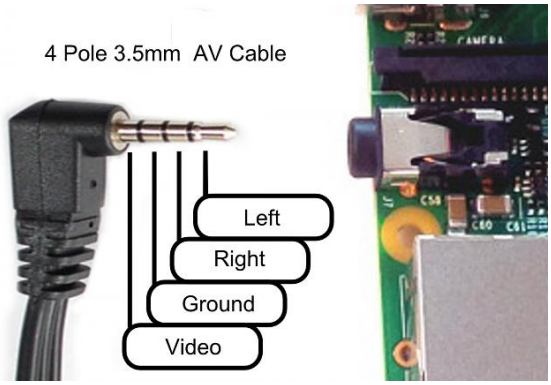


Figure 31 Raspberry PI B+ AV cable

When choosing a cable, seek an **iPod 4 pole AV** cable. This will however result in the left and right audio channels being reversed but otherwise provides the proper connections. Using other cables, such as a camcorder cable will be hit or miss. Typically, camcorder cables have the wrong pin connections for Video and Ground. This change also can cause some issues with shared grounding with audio speakers. If separate audio and composite AV connector is required, these can be split apart using the same jack inputs as for the model A and B.

Raspberry Pi Model 5

The Raspberry Pi model 5 was introduced in September 2023 and is approximately 2.5 times faster than its predecessor; the model 4B.

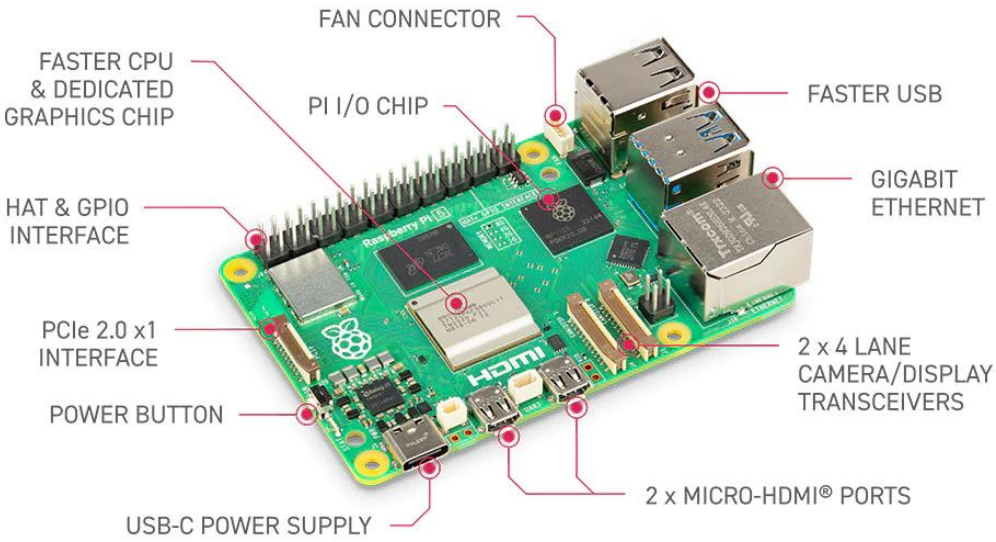


Figure 32 Raspberry Pi Model 5

Note: There is no audio or composite video output jack! See *Audio options for Raspberry Pi 5* on page 33.

Raspberry Pi Zero and Pi Zero W

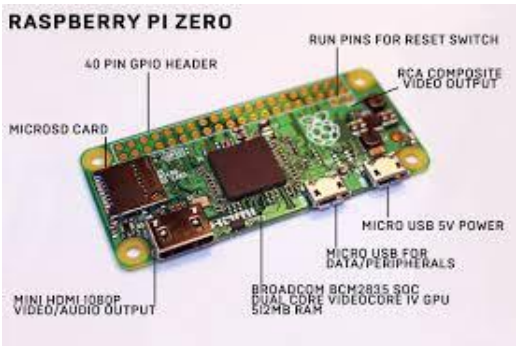


Figure 33 Raspberry Pi Zero

On the original Pi Zero, network connection is only possible with either a USB to Ethernet adapter or a Wi-Fi Dongle. Note that the USB is a Micro USB and will need a micro-USB to standard USB adapter. The Pi Zero W has onboard Wi-Fi and Bluetooth and is a better choice for the radio.

Figure 34 USB Ethernet adapter

In October 2021 the 1GHz quad core Pi Zero 2 was added to the list of Raspberry Pi's.

The Raspberry PI Zero does not have an onboard audio output jack. Sound must be played through either the HDMI port or a USB sound dongle (See Figure 12) or one of the Pi Zero DAC boards available from manufacturers such as **IQaudIO**, **HiFiBerry**, **Pimoroni** or **JustBoom**. Alternatively, use Bluetooth speakers.

One useful and simplest audio device is the **Pimoroni Audio DAC Shim** for the Pi Zero/W. Because the Audio DAC SHIM adds no extra bulk to the Raspberry Pi Zero it allows the GPIO header to be

accessed as normal, for example other HATs can be fitted on top of it. It pushes over the GPIO header and doesn't require any soldering. It is a very simple way to provide audio output to the Raspberry Pi Zero/W. See *Pimoroni Audio DAC Shim for Pi Zero* on page 29

Raspberry Pi 400

In November 2020 the Raspberry Pi foundation released the model 400. This is housed in a keyboard case with all connectors including a 40-pin GPIO header at the back of the unit. Most notably it does not have a 3.5mm audio Jack. If connected to a HDMI monitor or TV then sound can be played through the HDMI interfaced speakers on the monitor/TV. Alternatively, a USB DAC or Bluetooth speakers can be used. As the RPi 400 has a 40-pin GPIO header it can, in theory, be connected to a suitable DAC via a ribbon cable.

Note: At the time of writing no testing with a DAC other than a USB DAC has been done by the author.

Figure 35 Raspberry Pi 400 running the graphical radio

Displays

The HD44780U LCD display

Figure 36 The HD44780U LCD display

The HD44780U LCD interface is an industry standard interface for a variety of LCD displays. These can come in various sizes but the two lines by 2x16 or 4x20 character displays are the most popular. The software for this Internet radio supports either display. Most of these modules compatible with the Hitachi HD44780U LCD

Figure 37 OLED 4 x20 LCD display

controller so there is a wide choice of these displays.

The latest displays use **OLED character** displays (Organic Light Emitting Diode) and give very good results and are becoming more popular when compared to traditional LCD displays.

See https://en.wikipedia.org/wiki/OLED

For pin-out details see LCD pin outs on page 45.

Also see *Configuring Russian/Cyrillic text* on page 165.

Midas LCD displays with VEE

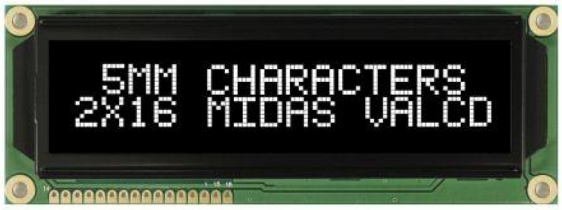


Figure 38 Midas LCD display with VEE

Some LCDs from Midas are compatible with the HD44780U except for pin 15 (VEE) which outputs a negative voltage. For pin-out details see LCD pin outs on page 45.

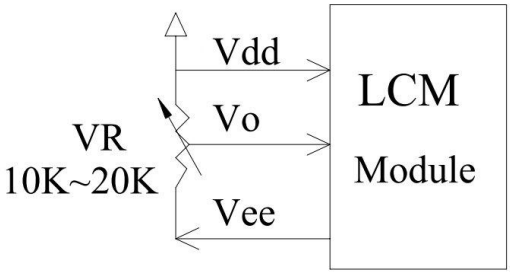


Figure 39 HD44780 potentiometer wiring

The diagram on the left shows the wiring for the 10K contrast potentiometer.

Pin	Name	Description
2	VDD	+5 Volt
3	VO	Contrast adjustment
15	VEE	Negative voltage output

Do not connect pin 15(VEE) to the +5V supply. It will damage the LCD and possibly the Raspberry Pi.

WARNING: DO NOT CONNECT AN I2C BACKPACK TO THIS TYPE OF DEVICE AS BACKPACKS CONNECT +5V ONTO PIN 15 AND WILL DAMAGE BOTH THE BACKPACK AND THE LCD.

The term **LCD** is used throughout this manual to mean both traditional **LCDs** and **OLED character** displays which are gradually replacing **LCDs**.

Midas Character OLED with MC0100 Controller

So-called Character OLEDs are gradually replacing LCDs. Midas market a wide range of such displays.

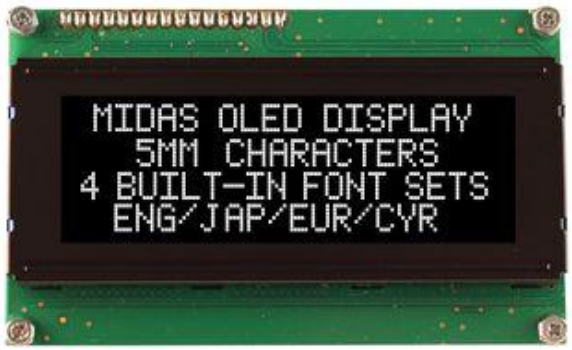


Figure 40 Midas character OLED

These displays use the **MC0100** controller which is largely compatible with the Hitachi **HD44780U** controller.

This controller can support various built-in font sets such as English, Western European, Japanese and Russian.

Since OLEDs generate their own illumination, they do not need pins 15 and 16 connected for backlighting.

Olimex OLED 128x64 pixel screen

The Olimex 128x64 pixel OLED is a low cost, low power, high contrast LCD display with a UEXT connector. It is controlled via the I2C or SPI interface. The power supply required is only in the range of 1 uA in sleep mode, 200 uA in operating mode and 7mA in display ON mode. View area is 21 x 11 mm. It is particularly useful where space is very limited.
See:
https://www.olimex.com/Products/Modules/LCD/MOD-OLED-128x64/open-source-hardware

Note: The Olimex OLED screen is not a particularly fast device when compared with say an LCD or graphics screen. However, its biggest advantage is its size.

Sitronix SSD1306 128x64 pixel OLED

Figure 41 Sitronix SSD1306 128x64 pixel monochrome OLED

The Sitronix SSD1306 128x64 pixel 0.96-inch OLED is display marketed under various names (Such as Makerhawk) has just four connections for the I2C interface.

The wiring from left to right is:

1. VCC +5 volts – GPIO header pin 2
2. GND (0 volts) – GPIO header pin 6
3. SCL I2C Clock – GPIO 3 (pin 5)
4. SDA I2C Data – GPIO 2 (pin 3)

The I2C interface hex address for this device **0x3C**. It uses the **ssd1306_class.py** driver.

Note: It is not currently possible to flip the display up-side down due to limitations of the SSD1306 driver software. Use the LUMA.SSD1306 driver instead which can be flipped.

SSH1106 1.3-inch I2C monochrome OLED

Figure 42 1.3-inch SH1106 I2C OLED

This 1.3-inch monochrome 132x64-pixel display connects via the I2C interface. It uses the SH1106 display chip.

The wiring from left to right is:

1. VCC +5 volts – GPIO header pin 2
2. GND (0 volts) – GPIO header pin 6
3. SCL I2C Clock – GPIO 3 (pin 5)
4. SDA I2C Data – GPIO 2 (pin 3)

The I2C interface hex address for this device **0x3C**. This OLED uses the LUMA driver. See **Error! Reference source not found.** on page **Error! Bookmark not defined.**.

Figure 43 SH1106 0.91" 128x32-pixel TFT

These OLEDs also come in in 128 x 32-pixel sizes from various manufacturers and can also be used with the Luma driver. The wiring from left to right in the example on the left is:

1. SDA I2C Data – GPIO 2 (pin 3)
2. SCL I2C Clock – GPIO 3 (pin 5)
3. Reset pin – Not used
4. GND (0 volts) – GPIO header pin 6
5. 3Vo – +3.3V output. Not used
6. VIN +5 volts – GPIO header pin 2

These OLEDs can also use the Luma driver and usually have an I2C address of 0x3C.

Note: These displays can be difficult to read due to their very small size.

SH1106 128x32 pixel OLED monochrome OLED

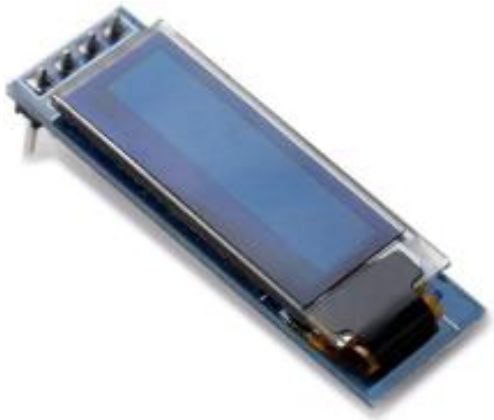


Figure 44 - 0.91-Inch 128x32 OLED

These 0.91 x 0.38-inch 128x32 pixel monochrome OLED display modules connect via the I2C Serial Interface. They are marketed under various names such as WayinTop or AzDelivery. This is the smallest display supported by this project.

Because of its size, the text can be hard to read. Also, one peculiarity is that the PIL driver software recognises the display as being 128x64 pixels and not 128x32. It is most useful where there is little space available for a display.

Note: OLED devices are very slow when compared to character LCDs or Graphics (touch) screens. Their main advantage is their size.

Grove LCD RGB Backlight

Figure 45 Grove I2C LCD RGB Backlight

The **Grove JHD1313 LCD RGB** backlight is a 2x16 character LCD with an RGB backlight. Version 4 and lower of the Grove LCD RGB LCD uses two controllers.
AIP31068L – I2C controller for the LCD
PCA9632DP1 – I2C RGB backlight driver

Version 5 onwards uses the **SGM31323** controller for the backlight.
The interface is the 2-wire I2C interface plus +5V power supply. The backlight can be set to any colour and is configurable in the radio software.

RPi 1.3" 128x64 OLED (SH1106) with joystick

Figure 46 RPi 1.3" 128x64 OLED with joystick

This device uses a 1.3-inch 128x64 pixel OLED display driven by an SH1106 chip and the SPI interface.

It comes with a 5-button joystick (on the left) and three push buttons (on the right). It is designed to fit on a 40-pi Raspberry Pi Zero-W but can also fit on a standard Raspberry Pi such as a 3B or 4B. If fitted to a Pi Zero Audio, audio output can be provided using a USB to jack-plug audio adaptor or Bluetooth speakers.

Note: This device comes wired to use an SPI interface. It can be modified to use the I2C interface using the I2C address **0x03**. In such a case use the LUMA-SH1106 I2C driver (See *Converting the 1.3" OLED pHat to use the I2C interface* on page 229).

Please note that the Pimoroni Audio DAC shim is not compatible with devices using SPI. This is because the OLED driver uses GPIO 25 which is also the MUTE pin on the Audio DAC shim. It Audio shim can only be used if the OLED with joystick is re-wired to use the I2C interface.

The following table shows the functions of the various switches and the GPIO pins that they are using. The Joystick and button wiring cannot be physically rewired however the Radio software needs to know what the wiring of each switch is and this is configured using the **configure_radio.sh** program.

Table 1 - 1.3" OLED with joystick button designations

Joystick	Buttons	Radio Function	GPIO Pin
Up		Channel Up	6
Down		Channel Down	19
Left		Volume Down	5
Right		Volume Up	26
Push		Menu switch	13
	KEY1	Mute Switch	21
	KEY2	Not used	n/a
	KEY3	Not used	n/a

Waveshare 2.42 and 1.5-Inch SPI interface OLEDs

Figure 47 Waveshare 1.42-inch SPI OLED Interface

Some OLEDS are rather on the small side and can be difficult to read at times. Waveshare make 1.42-inch OLED display that solves this problem. It has 5-wire SPI interface and the 2.42-inch interface gives good results.

Wiring details are described in Appendix *A.8 Waveshare 1.42" and 1.5" SPI OLED display wiring* on page 231.

Using DAC Sound Cards

The sound output of the on-board audio jack on the Raspberry Pi is known to be limited. Using a DAC (Digital Audio Converter) will give much better quality and output. Several types are available. The one you choose depends upon your requirements. These DAC cards use PCM (Pulse Code Modulation) technology and the Raspberry Pi i2s interface.

The following sound cards and adapters are supported in this project:

Manufacturer	Section	Page
HiFiBerry	HiFiBerry DAC	25
IQaudIO	IQaudIO DAC sound products	26
Justboom	JustBoom DAC products	27
Various	PCM5102A DAC Sound Card	28
Pimoroni	Pimoroni audio DACs	28
Adafruit	Adafruit speaker bonnet	30
Allo	Allo DAC products	30
Waveshare	Waveshare WM8960 Audio Hi-Fi Sound Card	31
Rpi DAC	Raspberry Pi badged sound cards	31

If you are going to use an external amplifier then almost any DAC will do. If you want a complete solution, a DAC card with an in-built amplifier (typically 3W up to 25W) is a good choice. The output from these in-built amplifiers is usually so-called class D-type amplification. Another consideration is the required connection to the amplifier copper, S/PDIF or optical audio connection (Toslink).

If price is a big consideration there are a number of very reasonably priced DACs which emulate some of the better-known ones and use the same device driver software as the one, they are emulating. See **Error! Reference source not found.** on page **Error! Bookmark not defined.**.

For more information on DACs see https://en.wikipedia.org/wiki/Digital-to-analog_converter

HiFiBerry DAC

This version supports the HiFiBerry DAC from HiFiBerry. See https://www.hifiberry.com. There is a comprehensive range of DACs available from this manufacturer. A few are shown below:

1. HiFiBerry DAC+ Light/Light – Entry level and standard solution
2. HiFiBerry Digi+ Light – Optical (TOSLink) and RCA connectors
3. HiFiBerry AMP+ –Standard DAC with a 25W D-Class amplifier
4. HiFiBerry DAC+ Zero Form Factor
5. HiFiBerry 3W Miniamp Zero Form Factor

The HifBerry DAC Plus

The HiFiBerry DAC PLUS uses the 40-pin connector and has an unpopulated 40-pin header to extend the GPIO pins on the HiFiBerry DAC to use with other cards.

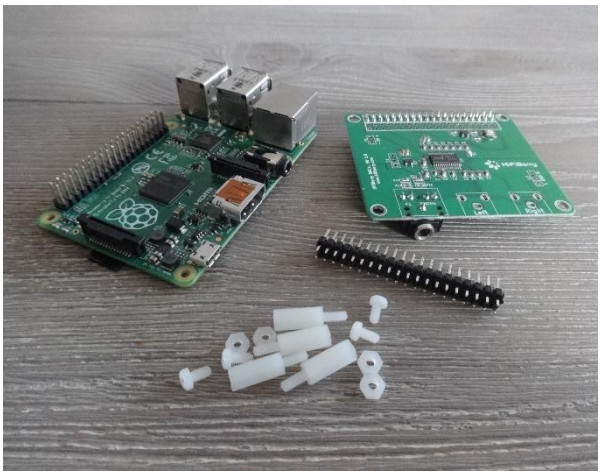


Figure 48 HiFiBerry DAC Plus

The DAC plus uses the 40-pin connector on new Raspberry PIs. A 40-pin dual in line male header is required (purchase separately).

Solder the 40-pin male header into the component side of the HiFiBerry DAC as shown Figure 49 below. The Radio controls and LCD screen for example are then connected on this this header.

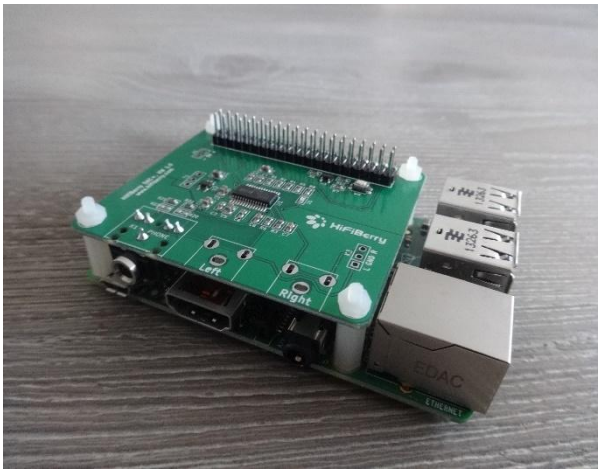


Figure 49 HiFiBerry mounted on the Rasberry Pi

The A+/B+/Pi2 uses the following pins supporting PCM.
Pin 12 GPIO18 PCM_CLK (Conflict!)
Pin 35 GPIO19 PCM_FS
Pin 38 GPIO20 PCM_DIN
Pin 40 GPIO21 PCM_DOUT
(All set to mode ALT0)

Pin 12 (GPIO) conflicts with the down switch on the radio. Wire the down switch to GPIO 10 (Pin 19) and configure the **down_switch=10** parameter in **/etc/radiod.conf** or by running the **configure_radio.py** program.

Note: All settings in the /etc/radiod.conf file use GPIO numbers and NOT physical pin numbers. So, in the above example down_switch is GPIO 10 (Physical pin 19).

The Pimoroni pHat is compatible with HiFiBerry DAC (Not DAC+) and uses the same Device Tree (DT) overlay.

IQaudIO DAC sound products

IQaudIO DAC also have a comprehensive range of products. Again, these provide excellent results. These cards fit within the Pi's form factor and provide additional full access to the Pi's 40way I/O signals allowing easy addition of IR sensors, Rotary Encoder or i2c devices (such as OLED screens) etc. See http://iqaudio.co.uk

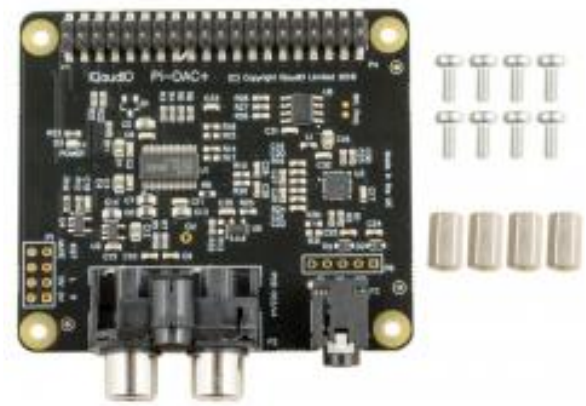


Figure 50 IQaudIO DAC plus

DAC for the Raspberry Pi. This latest revision of the IQaudIO Pi-DAC PRO and is pre-programmed for auto detection. Line out: 2x Phono/RCA
Headphone: 3.5mm socket.

Figure 51 IQaudIO Pi-DigiAMP+

This provides the DAC+ along with a 35W amplifier which fits the. Raspberry Pi A+/B+/RPi2/3/3B+. This card requires supports up to 24v power supply and delivers the full 2.5amp to the Pi.

JustBoom DAC products

The construction using **JustBoom** products is similar to other sound cards. The radio must be wired as shown in *Table 4 Radio and DAC* devices 40-pin wiring. The **/etc/radiod.conf** configuration file must also be configured to support these devices by running the audio configuration program.

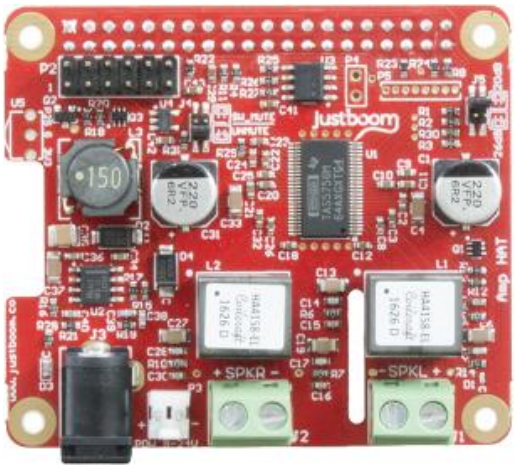


Figure 52 JustBoom Amp HAT

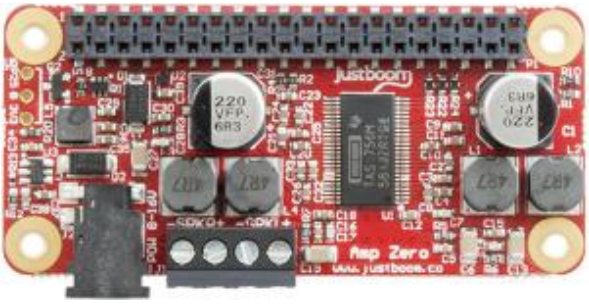


Figure 53 JustBoom Amp Zero pHAT

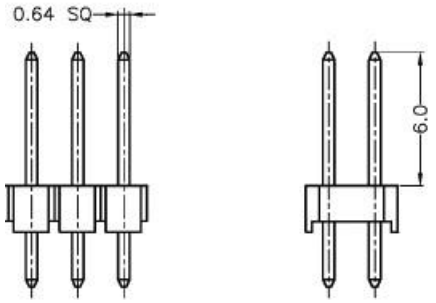


Figure 54 JustBoom Zero stacker requirements

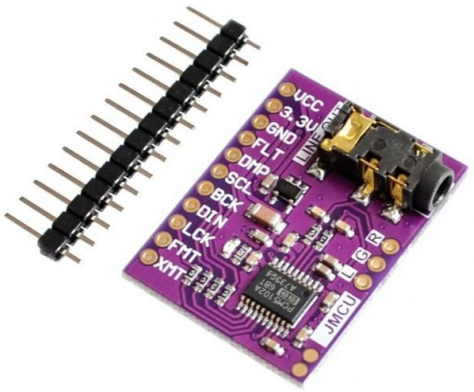
The **JustBoom** Zero boards are used with stackers or installed directly on the Raspberry Pi Zero. Some stackers and some 2x20 male headers on the market though are too thin or too short to provide good contact with the board. Use stackers that the pins are squared and are at least 0.6mm in width. If you are soldering the 2x20 male header on the Raspberry Pi Zero make sure that the pins are 0.6mm in width and 6mm in usable height.

Figure 55 Using the 40-pin stacker

Plug a suitable stacker onto the Raspberry Pi Zero. Plug the JustBoom Zero board on top of the stacker so that the pins protrude through the JustBoom Zero board.

Plug the radio interface card or ribbon cable (not shown) on top of these protruding pins.

PCM5102A DAC Sound Card



There are a number of inexpensive DACs which use the PCM5102A chip and have a 3.5mm Stereo Jack 24. These give a very reasonable 24-bit sound. Despite the fact that they claim to be a pHat they do not plug directly into the Raspberry Pi. Use either jumper wires or build your own interface board. These boards use the HiFiberry DAC device driver and the I2S dtoverlay. Wiring as below:

DAC	Function
FLT	Filter Select. Normal latency (Low) / Low latency (High)
DMP	De-emphasis control for 44.1kHz sampling rate: Off (Low) / On (High)
FMT	Audio format selection: I2S (Low) / Left justified (High)

DAC	RPi	Pin
VCC	5V	2
3.3V	N.C.	-
GND	GND	6
FLT	N.C.	-
DMP	N.C.	-
SCL	N.C.	-
BCK	GPIO18	12
DIN	GPIO21	40
LCK	GPIO19	35
FMT	N.C.	-
XMT (1)	3.3V	1

N.C. Not Connected. (1) Connect via 10K resistor.

Also edit **/boot/firmware/config.txt** and enable the I2S dtoverlay.

```
dtparam=i2s=on
```

Pimoroni audio DACs

Pimoroni pHat DAC

The Pimoroni pHAT DAC provides an affordable high-quality DAC for the Raspberry Pi. The 3.5mm stereo jack comes soldered onto the board already. Though designed to match the format of the Raspberry Pi Zero it is compatible with all 40-pin GPIO Raspberry Pi variants.

Features:

- 24-bit audio at 192KHz
- Line out stereo jack
- pHAT format board
- Uses the PCM5102A DAC to work with the Raspberry Pi I2S interface

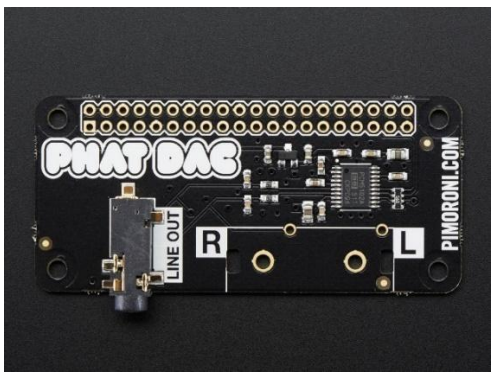


Figure 56 Pimoroni pHat DAC

Note 1: Do not use the 40 female header that comes with the board but use a 40-pin extender so that other cards can be used on top of it.

Note 2: The Pimoroni pHat is not completely compatible with the HiFiberry DAC although it uses the same software driver. In particular to use the Alsa sound mixer a package called **pulseaudio** is required. However, **pulseaudio** is not compatible with several of the features of this package such as the **espeak** speech package. Normally the **pulseaudio** package must be removed as shown in the section called **Error! Reference source not found.** on page **Error! Bookmark not defined.**. The Pimoroni pHat DAC will run fine without any mixer controls.

Pimoroni Audio DAC Shim for Pi Zero

Please note that the Pimoroni Audio DAC shim is not compatible with *RPi 1.3" 128x64 OLED (SH1106) with joystick* (See page 23). This is because the OLED driver uses GPIO 25 which is also the MUTE pin on the Audio DAC shim. It Audio shim can only be used if the OLED with joystick is re-wired to use the I2C interface (See *Converting the 1.3" OLED pHat to use the I2C interface* on page 229).

Pimoroni now supply a DAC which uses PCM5100A DAC chip. It has a friction fit header which slips pushes down over the Pi Zero 40-pin GPIO header and doesn't require any soldering and is removable.

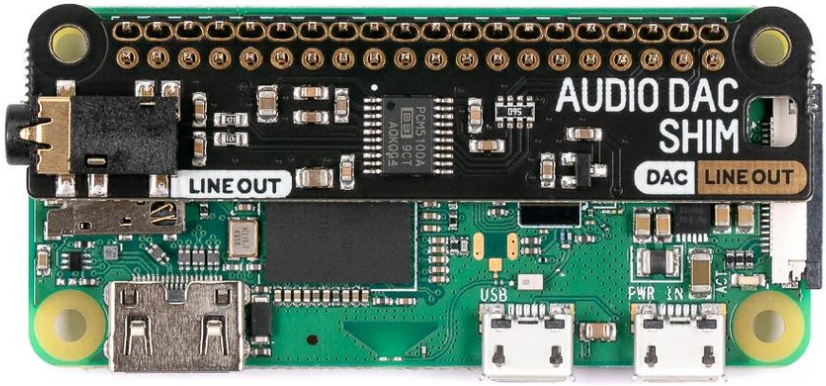


Figure 57 Pimoroni Audio DAC Shim for Pi Zero/W

The above figure shows a Pimoroni Audio DAC shim fitted to a Pi Zero W. The PCM5100A DAC chip takes high quality digital audio from the Pi Zero and pipes out crisp, line-level 24-bit / 192KHz stereo audio through the 3.5mm jack.

Because the Audio DAC SHIM adds no extra bulk to the Raspberry Pi Zero it allows the GPIO header to be accessed as normal, for example other HATs can be fitted on top of it. It should also fit easily inside any standard case. It's a simple way to add an audio output to the Pi Zero. Pimoroni even claim that it can also be fitted to a Raspberry Pi 400.

It uses the same driver as HiFiBerry DAC (Not DAC Plus or Digi). This can be configured during installation of the radio software.

Alternatively, it can be manually configured using by adding the following lines to the [All] section of **/boot/firmware/config.txt** (for **Bookworm**) **or /boot/config.tx** (for **Bullseye**) file:

```
dtoverlay=hifiberry-dac
```

```
gpio=25=op,dh
```

If you're using a Pi that has an audio jack you might also need to disable onboard audio by either adding a # to the beginning of the following line or setting **on** to **off**.

```
#dtparam=audio=on
```

Adafruit speaker bonnet

Figure 58 Adafruit Speaker Bonnet

The Adafruit speaker bonnet is primarily designed for the Raspberry Pi Zero format but can be used on any Raspberry Pi.

It consists of a stereo 3W amplifier connected to the I2S interface of the Raspberry Pi via a 40-pin DIL female header.

It can be purchased with two miniature speakers or it may be used with any small 4 or 8-Ohm speakers.

Allo DAC products

Allo is a manufacturer of very high-quality audiophile sound products. A good example is the **Allo Piano 2.1 HiFi DAC (EU)** with **woofer output** as shown below.

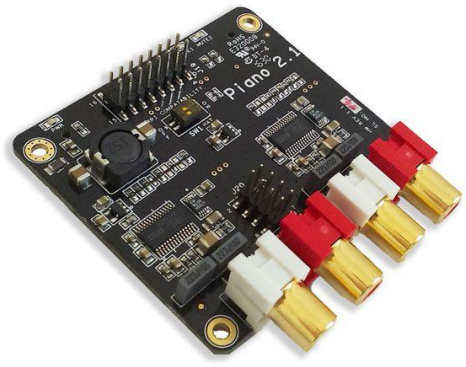


Figure 59 Allo Piano 2.1 HiFi DAC with woofer

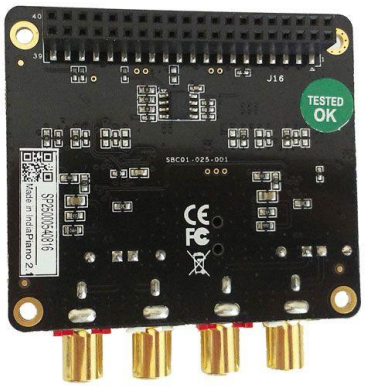


Figure 60 Allo Piano 2.1 HiFi DAC (Rear)

The **Allo Piano** has two outputs, one for normal sound ranges and one for a sound woofer. The board comes pre-programed as 2.1 for the Raspberry Pi. The subwoofer has 2 outputs (Stereo left

and right) but is mono only. However, it uses a second I2S channel on GPIO5 and mute signal on GPIO6 both used to control the woofer amplifier. This conflicts with the LCD signals **lcd_data4** an **lcd_data5** pins.

Allo products currently need to be specially configured to correct this conflict. See *Configuring Allo Sound Cards* on page 130.

Further information about Allo products can be found at https://www.allo.com/sparky/index-dac.html

Waveshare WM8960 Audio Hi-Fi Sound Card

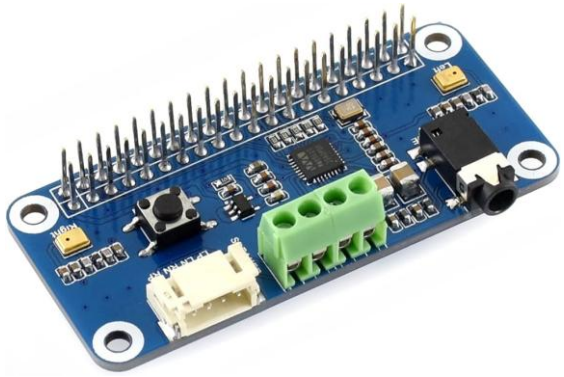


Figure 61 Waveshare WM8960 Sound Card

The Waveshare WM8960 Audio Hi-Fi Sound Card integrates WM8960 low power stereo CODEC, which communicates via I2S interface. It also has two high-quality MEMS silicon Microphones (Not used in this project) mounted on the left and right of the board.
Onboard standard 3.5mm earphone jack, play music via external earphones.
Onboard dual-channel speaker interface, directly drives speakers. **Note:** Waveshare have dropped support for this card on Bullseye so you can only run it on **Bookworm OS**.

These cards are often sold with a pair of speakers which plug directly into the card. The particular speakers shown are 5W enclosed speakers.

For more information about this sound card see https://www.waveshare.com/wm8960-audio-hat.htm

This card has special installation and uninstallation scripts. See *Installing the Waveshare WM8960 DAC* on page 131.

Raspberry Pi Audio Cards

Figure 62 Raspberry Pi DAC Pro

Raspberry Pi DAC Pro

The **Raspberry Pi DAC Pro** is the highest-fidelity audio HAT. With the Texas Instruments PCM5242, the DAC Pro provides an outstanding signal-to-noise ratio (SNR) and supports balanced/differential output in parallel to phono/RCA line-level output. It also includes a dedicated headphone amplifier. This card is a rebadged **IQaudIO DAC Pro**.

Figure 63 Raspberry Pi DigiAMP+

Raspberry Pi DigiAMP+

The **Raspberry Pi DigiAMP+** is a high-performance audio HAT. With its on-board Texas Instruments TAS5756M stereo amplifier, it delivers a direct connection to passive stereo speakers at up to 35 Watts per channel with variable output, and is ideal for use in Raspberry Pi-based Hi-Fi systems. Powered by an external 12–24V DC power supply, it connects directly to Raspberry Pi's GPIO header, providing power to the Raspberry Pi itself. This card is a re-badged **IQaudIO DigiAMP+.**

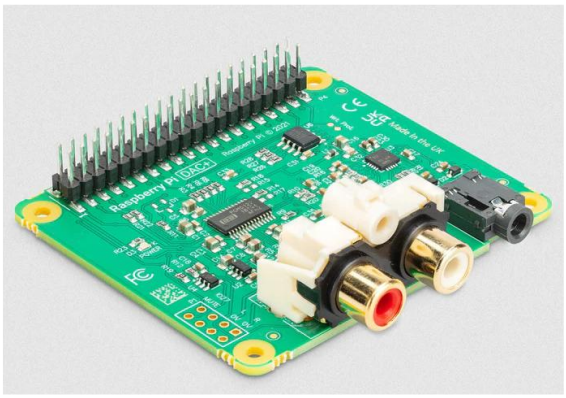


Figure 64 Raspberry Pi DAC+

Raspberry Pi DAC+

Rebadged from the **IQaudIO DAC+,** the Raspberry Pi DAC+ a low-cost audio output HAT for the Raspberry Pi (any 40-pin model), supporting 24-bit 192kHz high-resolution digital audio. It uses a Texas Instruments PCM5122 DAC to deliver stereo analogue audio to a pair of phono connectors, and also provides a dedicated headphone amplifier. The DAC+ is compatible with any 40-pin Raspberry Pi and requires no soldering. It uses a Texas Instruments PCM5242 DAC for even higher signal-to-noise ratio.

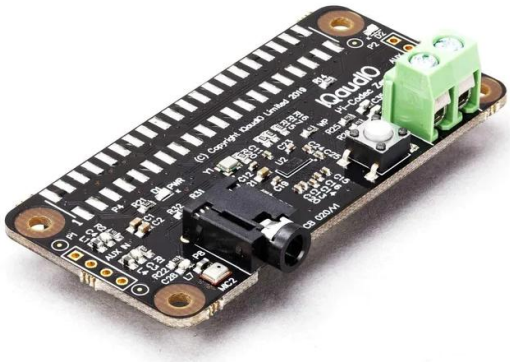


Figure 65 Raspberry Pi Codec Zero

Rasperry Pi Codec Zero (Mono)

The Codec Zero is a Raspberry Pi Zero-sized audio HAT that delivers bi-directional digital audio signals (I2S) between a Raspberry Pi and the Codec Zero's on-board Dialog Semiconductor DA7212 codec, and allows you to use a variety of input and output devices. With features such as programmable green and red LEDs, plus a push button for user input, Codec Zero is suitable for projects requiring a microphone input. It supports a 1.2W 8Ω mono speaker.

Pimoroni Products

Pimoroni are a UK based company who produce electronic products for both Raspberry Pi and Arduino. They make a range of all-in-one audio boards for Raspberry Pi, with high-quality digital audio. Their Website is at https://shop.pimoroni.com/

Pimoroni Pirate Radio

Figure 66 Pimoroni Pirate Radio - Rear view

The illustration on the left shows the rear of Pimoroni Pirate radio. The amplifier consists of dual I2S DAC/amplifiers for stereo audio (MAX98357A) at 3 Watts per channel.

The Pirate radio comes as a kit (Soldering skills required). The Pimoroni software is disabled and the software from this project used instead. See **Error! Reference source not found.** on page **Error! Bookmark not defined.**. Note: It does not have a screen.

Pimoroni Pirate Audio

Figure 67 Pimoroni Pirate Audio

No soldering skills are required to construct this project when using a Pimoroni Pirate Audio range of products and a Raspberry Pi Zero with a pre-soldered 40-pin header.

It comes with a 240x240 pixel colour 1.3-inch IPS (In-plane switching) display which gives a very good viewing angle. The display is driven by an ST7789 controller.

There are four variants of the Pimoroni Audio but they all use the same DAC and display software:

1. Pirate Audio Speaker - MAX98357A DAC with mini 1W / 8Ω speaker
2. Pirate Audio Line-out - PCM5100A DAC chip with 3.5mm output stereo jack
3. Pirate 3W Stereo Amp - MAX98357A DAC with mini 3W amplifier output
4. Pirate 3W Headphone Amp - PCM5100A DAC driving a PAM8908 headphone amplifier

Note: The software for Pimoroni Pirate Audio is very basic at the moment in particular the menu selection function. It is hoped to improve this at a later date.

Audio options for Raspberry Pi 5

The **Raspberry Pi 5** does not have an on-board audio jack. This means that it requires one of the following three options for audio output:

1. A half size audio DAC such as an Adafruit I2S DAC bonnet
2. A USB Bluetooth audio dongle
3. Bluetooth speakers or headset

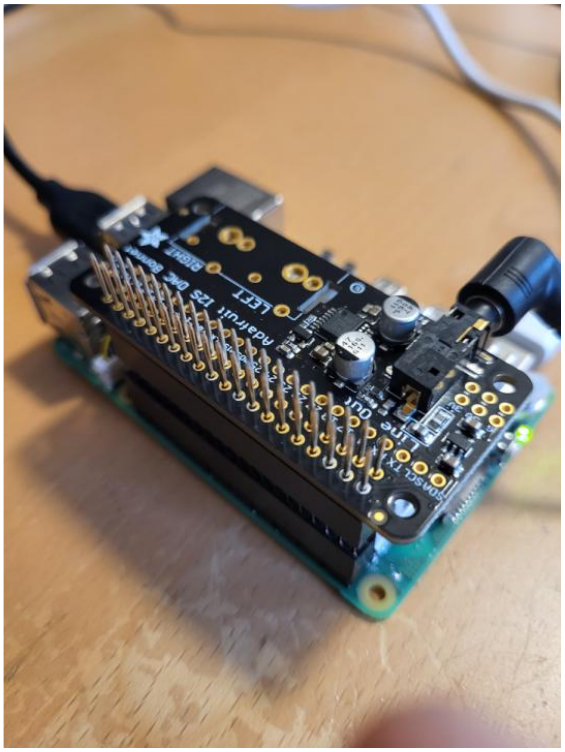


Figure 68 Raspberry Pi with a Adafruit DAC

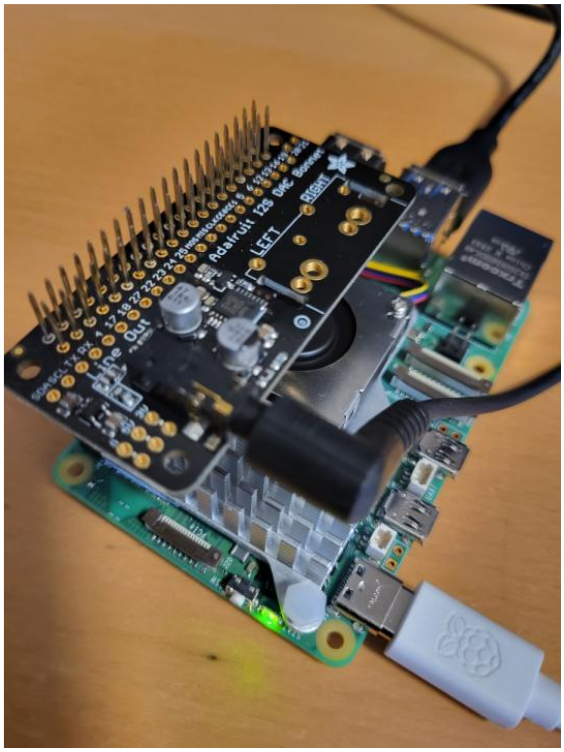


Figure 69 Raspberry Pi 5 DAC cooling fan clearance

Figure 68 above shows an Adafruit I2S DAC bonnet mounted on a 40-pin header extender. This is needed to give adequate clearance for the cooling fan (if fitted) as shown in the second Figure 69.

Note: It is probably advisable to fit a half-size DAC (Pi Zero format) rather than a full-size DAC to improve cooling fan airflow. A full-size DAC Hat may still allow for adequate cooling but no testing has been done on this.

Touch-screens

Adafruit 2.8 and 3.5-inch TFT touch-screens

The radio software supports the Adafruit 2.8 and 3.5-inch TFT touch screen (480x320 and 720x480 pixels respectively). The small size can make the controls difficult to use but it will still work. It is best to use a touch-screen stylus.

Figure 70 Adafruit 3.5-inch TFT touchscreen

For **Bullseye** only: If the screen is displaying upside-down then edit **/boot/config.txt** (for **Bullseye**) and add the following line.

```
lcd_rotate=2
```

Uncomment to force a console size of 1280x720 pixels.

```
framebuffer_width=1280
framebuffer_height=720
```

In the same file disable the vc4-kms-driver

```
#dtoverlay=vc4-kms-v3d
```

For **Bookworm**: To rotate screen if no desktop, add this to the end of cmline.txt in **/boot/firmware/cmdline.txt** separating it with a space from the rest of the line

```
video=DSI-1:800x480@60,rotate=180
```

Insert the following two lines into **/boot/firmware/config.txt** after the block called *# Enable DRM VC4 V3D driver*. If the file contains

```
dtoverlay=vc4-kms-dsi-7inch,invx,invy
#display_auto_detect=1
```

Reboot the Raspberry Pi.

Waveshare 2.8 and 3.5-inch TFT touch-screens

Waveshare provide a 2.8 and 3.5-inch TFT touch screen (480x320 and 720x480 pixels respectively). These have 26-pin DIL connector which plugs in directly to the GPIO header.

Figure 71 Waveshare 2.8-inch TFT touch screen

The above screen shows a 2.8-inch Waveshare touch screen (Courtesy - Recep A. Güleç). Note the revised layout due to its size when compared to the 3.5 and 7-inch touch screens.

Note: This software has only been tested with the Raspberry Pi 7-inch touch screen and the Adafruit 3.5-inch TFT touch-screen . Smaller than 7-inch screens may prove difficult to operate. The following resolutions are supported: 800x480, 720x320, 480x320 or 1024x600 pixels.

Raspberry Pi 7-inch touch screen

As an alternative to building a radio using limited LCD screens it is possible to build a radio using the Raspberry Pi 7-inch or 3.5-inch touch screen or any other HDMI screen (touch-screen or otherwise). If the screen does not have touch capability, then it is possible to use it with a mouse or keyboard or both. Also, the touch screen can be used in conjunction with rotary encoders or push buttons.

Figure 72 Raspberry Pi 3 with 7-inch touch screen

There is a very good setup guide for the Raspberry Pi touch-screen at:
https://www.modmypi.com/blog/raspberry-pi-7-touch-screen-assembly-guide

Connection via ribbon cable

Smaller touch screens usually plug directly into the Raspberry Pi GPIO header, however that may not be convenient especially if you want to build it all into a case. Fortunately, it is possible to purchase a 40-way DIL male to female ribbon cable which allows the display to be panel mounted.

The following illustration shows TFT screen on the connectors side. If it is required the display can be put on the GPIO side but will be displayed upside down. Luckily it is easy to flip the display upside down as shown in the installation instructions.

Figure 73 TFT connected by a 40-pin male/female ribbon cable

MHS 3.5-inch RPi Display

The MHS 3.5-inch RPi Display appears to be a badged version of the same hardware as the Waveshare 3.5" TFT. The installation software is also very similar.

Figure 74 MHS-3.5-inch RPi Display (Courtesy Brent Fraser)

More information the MHS 3.5-inch RPi Display will be found at:
http://www.lcdwiki.com/MHS-3.5inch_RPi_Display

Radio variants

Before starting you need to make a choice which type of radio you are going to build. There are several combinations of user interface and display type which can be constructed as shown in the following tables.

Table 2 Display Type options

	Display Type
1	Two-line 8-character LCD
2	Two-line 16-character LCD
3	Four-line 16-character LCD
4	Four-line 20-character LCD
5	Àdafruit 2x16 RGB Plate (I2C)
6	Raspberry Pi 7-inch touch screen
7	Olimex 128 by 64-pixel OLED
8	Adafruit 3.5-inch TFT touch screen
9	No display (Vintage radio design)
10	Pirate Radio (No display)
11	Waveshare 1.8/2.3" touchscreen
12	Grove I2C 2-line 16-char. LCD
13	OLED displays supported by LUMA (SH1106, SSD1306 etc.)
14	SH1106 128x64 1.3" OLED (SPI interface)

Table 3 User interface options

	User interface
1	Five or six push buttons
2	Two rotary encoders with push buttons
3	Adafruit RGB plate with push buttons
4	Raspberry Pi 7-inch touch-screen
5	Adafruit 2.8" or 3.5" TFT
6	Mouse (HDMI/Touchscreen only)
7	Keyboard (HDMI/Touchscreen only)
8	IQaudIO Cosmic controller
9	Pirate Radio – 6 push-buttons
10	IR remote control – all versions
11	Waveshare 1.8/2.3" touchscreen
12	pHat with 5-positon joystick an 3-buttons

Any type of **HD44780U** LCD display can be used with any user interface. The HD44780U can either be connected directly to the GPIO pins or via a so-called I2C (also known as IIC) backpack.

The Adafruit RGB plate has a two-line 16-character display and comes with five inbuilt pushbuttons. It also has its own I2C interface using the MCP23017 chip so it does not require a separate I2C backpack.

The **PiFace CAD** comes with a two-line 16-character display and comes with six inbuilt pushbuttons. It uses the SPI interface from Motorola. The touch screens can be used with or without rotary encoders or push buttons. The touch screen variants can also use a mouse and keyboard.

It is a simple choice of which display (two or four lines, 8,16 or 20-characters LCDs or a touch screen or HDMI screen, OLED display or Pirate radio) and whether to use rotary encoders or push-button switches as the user interface. The rotary encoder options give the most natural feel for the radio as most conventional radios use knobs to control the volume and station tuning. The keyboard interface, whilst supported on the touch-screen versions, is a very limited option.

There is a configuration program called **configure_radio.sh** which configures the choice of display and user interface required. It can be safely re-run at any time.

The vintage radio software (Display option 9) specifically intended for converting an old radio to an Internet radio whilst retaining the original look and feel of the radio. It has no LCD display.
The four lines LCD can display more information than two-line versions.

Note: The touch screen software (**gradio.py** or **vgradio.py**) cannot be run at the same time as the LCD version of the radio software (**radiod.py**). It is a case of using one or the other. It is however it is possible to switch between **gradio.py** or **vgradio.py** programs during operation.

Connecting the LCD display

There are two ways of wiring up the display:

- Directly connect the LCD to the GPIO pins. This uses six GPIO pins.
- Connection via an I2C backpack or inbuilt I2C interface. These uses the two-pin I2C interface

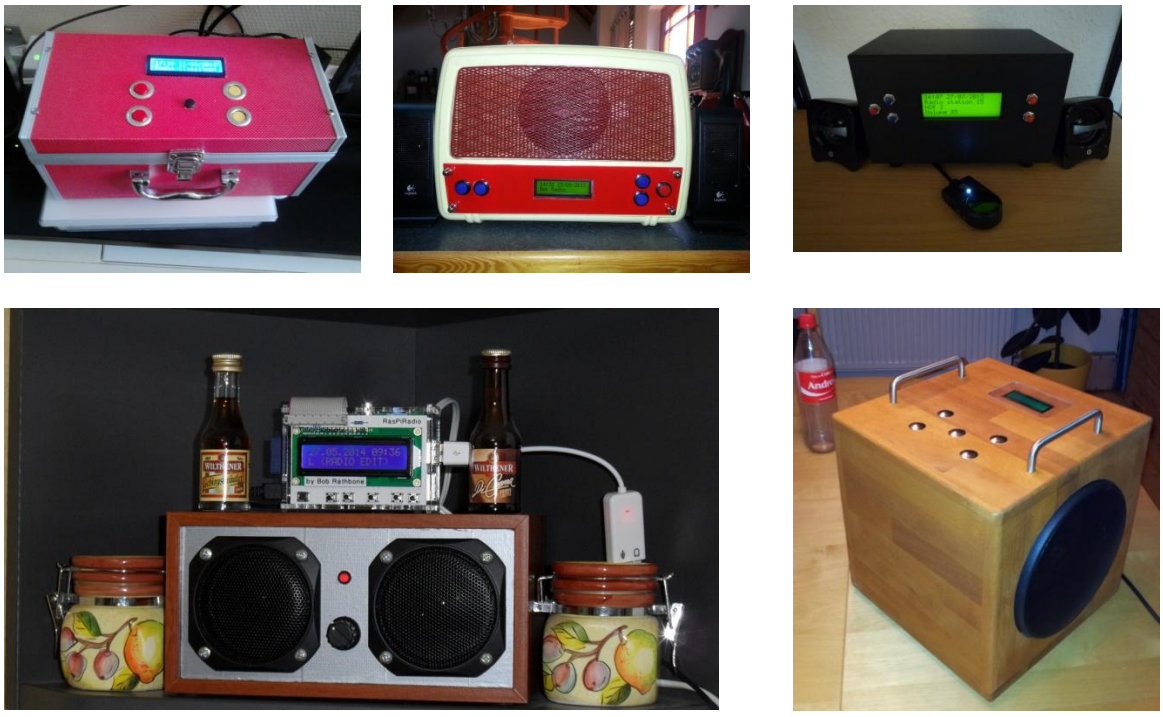
The first choice uses more wiring but is the cheapest option. The second choice uses an I2C backpack which is an extra component to be purchased. However, I2C backpacks are reasonably cheap.

Housing the radio

This manual describes a couple of ways of housing the radio. A few ideas are below:

- A custom-built case as shown in this manual
- Old plastic boxes or food containers
- Construct a case using Lego
- Use a pair of speaker housings that have enough room
- Install in an old vintage radio (really cool)
- Use an old wooden wine box
- Use an old video recorder, CD player or desktop set
- Buy a PC speaker set with enough room to build in the radio.

Figure 75 Some examples of radio cases



Take a look at the constructor's gallery at https://bobrathbone.com/raspberrypi/pi_internet_radio.html to get some ideas that other constructors have used.

Note: Don't forget to make sure that there is adequate airflow through the radio housing to allow cooling of the Raspberry PI and other components. If necessary, drill at least five or six holes at the top and bottom of the housing.

If you decide to use a metal case (not advised) you will need a Wi-Fi dongle with an aerial mounted externally to the case. Also, the case must be earthed at the main supply both for safety reasons and to prevent interference with sound and/or the LCD screen

Building in an IR sensor and remote control

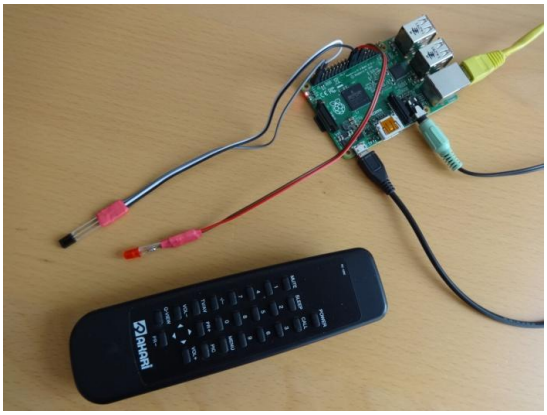


Figure 76 IR Sensor and Remote control

The radio can be built with an IR Sensor and remote control. Also included is an activity LED which flashes when the remote control is used.

A **TSOP382xx** series IR Sensor is used in conjunction with almost any remote control. An activity LED can also be added which flashes every time remote control signal is detected. The remote control provides the same functionality as the buttons or rotary encoders.

Although nearly any remote control that you happen to have lying around can be used, it is recommended to use the **Raspberry Pi Mini IR Remote Control** available from suppliers such as **PiHut**. Apart from its size and price advantage you will find a definition for this device called **mini.toml** in the **/usr/share/radio/remotes** directory which will save the effort of setting up your own.

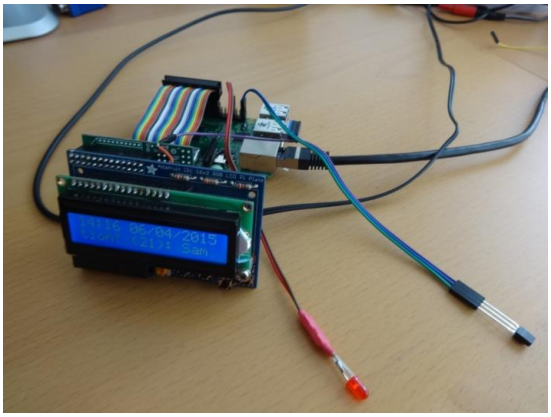


Figure 77 Adafruit and IR sensor and activity LED

The **AdaFruit** RGB plate can also be fitted with an IR sensor and activity LED but needs a model B+, 2B or 3B (40 GPIO pins) and 26 pin extender as shown in Figure 95 on page 55.

Note that a 40 pin Raspberry PI is needed as the Adafruit Plate occupies the first 26 pins of the Raspberry Pi GPIO header.

Chapter 3 – Wiring and Hardware

All Raspberry Pi models using 40-pin GPIO headers

Table 4 Radio and DAC devices 40-pin wiring

Pin	Description	Radio Function	Name	Audio DAC Function	LCD Pin	Push Button	Encoder (Tuner)	Encoder (Volume)
1	3V3	+3.3V supply	+3.3V	+3.3V		+3.3V		
2	5V	5V for LCD	+5V	+5V	2,15			
3	GPIO2	I2C Data	I2C Data	I2C Data				
4	5V			+5V				
5	GPIO3	I2C Clock	I2C Clock	I2C Clock				
6	GND	Zero volts	0V	0V	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		0V				
10	GPIO 15	Volume up	UART RX			RIGHT		Output B
11	GPIO 17	Menu switch				MENU	Knob Switch	
12	GPIO 18			I2S CLK				
13	GPIO 27	Record button						
14	GND	Zero Volts		0V				
15	GPIO 22			Mute				
16	GPIO 23	Channel down		Rotary enc A		DOWN	Output A	
17	3V3	+3.3V supply		0V				
18	GPIO 24	Channel up		Rotary Enc B		UP	Output B	
19	GPIO 10		SPI-MOSI					
20	GND	Zero Volts						
21	GPIO9		SPI-MISO					
22	GPIO 25	IR Sensor		IR sensor				
23	GPIO 11		SPI-SCLK					
24	GPIO 8	LCD E	SPI-CE0		6			
25	GND	Zero Volts		0V				
26	GPIO 7	LCD RS	SPI-CE1		4			
27	DNC			PiDac+ Eprom				
28	DNC			PiDac+ Eprom				
29	GPIO5	LCD Data 4			11			
30	GND	Zero Volts						
31	GPIO6	LCD Data 5			12			
32	GPIO12	LCD Data 6			13			
33	GPIO 13	LCD Data 7			14			
34	GND	Zero Volts						
35	GPIO 19	IQaudIO DAC+	I2S	I2S				
36	GPIO 16	IR LED out						
37	GPIO 26							
38	GPIO 20	IQaudIO DAC+	I2S DIN	I2S DIN				
39	GND	Zero Volts						
40	GPIO 21	IQaudIO DAC+	I2S DOUT	I2S DOUT				

Colour Legend	Radio	IQaudIO	I2C (shared)	SPI Interface

Wiring Push Button radios and the record button

Wire one side of the push-buttons the GPIO pin as shown in the last column of Table 4 on page 43. Wire the other side of the switches to either +3.3V (Old wiring scheme) or to GND (0V) (Preferred wiring scheme recommended for new projects). Whichever wiring you use; the radio configuration program will ask which wiring scheme is being used. Version 2 onwards boards have internal pull up/down resistors and don't require external resistors. In fact, including these can cause problems.

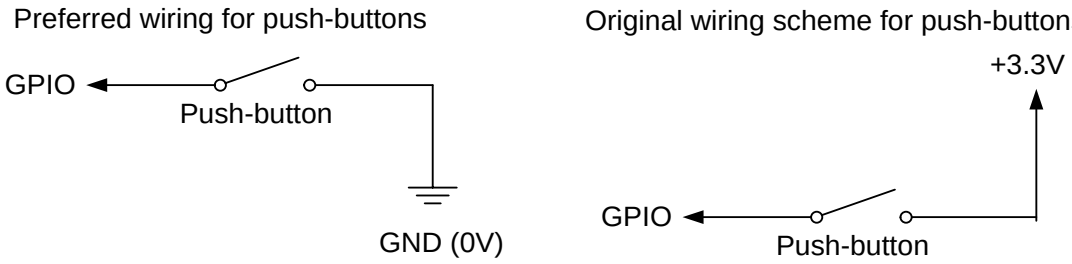


Figure 78 Push-button wiring version 2 onwards boards

The scheme chosen must be configured using the **pull_up_down** parameter in **/etc/radiod.conf** to 'up' or 'down. See the section called **Error! Reference source not found.** on page **Error! Bookmark not defined.**.

Radios using rotary encoders

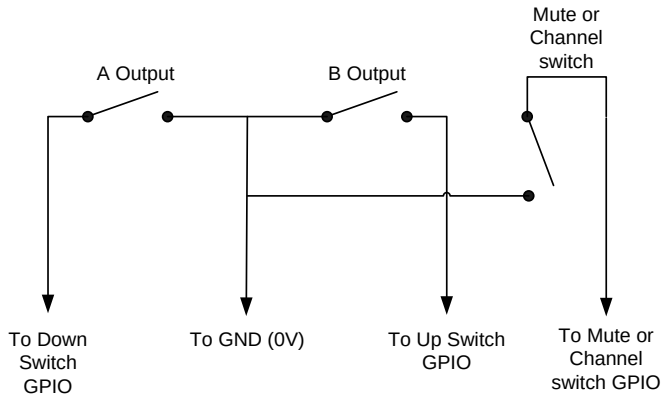


Figure 79 Rotary Encoder Diagram

Rotary encoders have three inputs namely Ground, Pin A and B as shown in the diagram on the left. Wire the encoders according that shown in **Error! R eference source not found.** on page **Error! Bookmark not defined.**. If the encoder also has a push button knob, then wire one side to ground and the other to the GPIO pin. In the case of the mute switch this will be pin 7 (GPIO 4). Version 1 boards are not supported but will probably work.

Warning: The push switches (if fitted) on the rotary encoder are wired differently from the push buttons in the earlier push button versions of the radio. For these encoders one side of the push button is wired to GND (not 3.3V) and the other to the relevant GPIO.

If using a Revision 1 board it is necessary to use 10K pull up resistors connected between the GPIO inputs of the rotary encoder outputs and the 3.3-volt line. Do not add resistors if using revision 2 boards and onwards.

Note: Although version 1 boards with 26-pin GPIO headers are no longer supported, you can still find wiring details for these in the appendices for the *Raspberry Pi Internet Radio – Technical Reference* manual.

Figure 80 Rotary encoder with push switch

This project originally uses a COM-09117 12-step rotary encoder or PEC11R series encoders. It also has a select switch (by pushing in on the knob). These are "Incremental Rotary Encoders". An incremental rotary encoder provides cyclical outputs (only) when the encoder is rotated. The other type is an absolute rotary encoder which maintains position information even when switched off (See Wikipedia article on rotary encoders).

The rotary encoders used in this project are wired with the COMMON or GND pin in the middle and the A and B outputs either side. However, some rotary encoders are wired with A and B as the first two pins and GND (COM) as the third pin. Note that not all encoders come with a switch, so separate switches for the Menu and Mute button will need to be installed. Check the specification first.

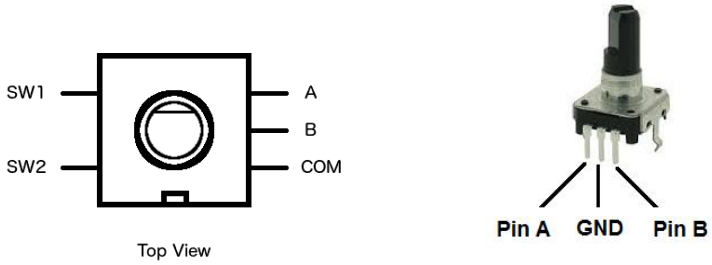


Figure 81 Rotary encoder pin-outs

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 encoders.

Using KY-040 Rotary encoders

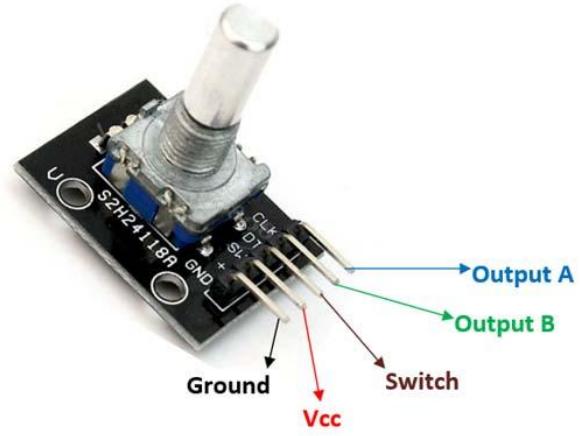


Figure 82 KY-040 Rotary Encoder

These cost-effective Rotary Encoders from Handson Technology originally designed for use with Arduino are now being used more and more by constructors. The KY-040 Rotary Encoder specification shows that these are powered by +5V to the **VCC** pin however VCC must be connected to **+3.3V** and not +5V otherwise you will damage the Raspberry Pi

Figure 83 KY-040 with three 10K resistors

Originally these rotary encoders had three 10K pull-up resistors connected between outputs CLK (R2), DT(R3) and SW(R1) to VCC supply. However, manufacturers discovered that developers did not necessarily want to have the pull-up resistor between the switch connection and VCC as they might want to connect the switch to GND (0V) when operated.

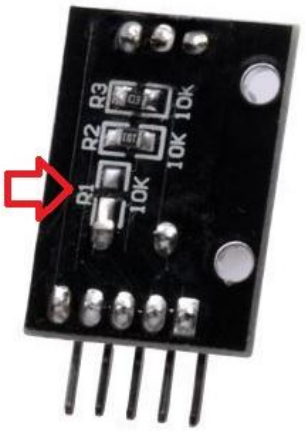


Figure 84 KY-040 with resistor R1 omitted

As a result, manufacturers now tend to supply these rotary encoders with the R1 10K resistor omitted leaving the decision on how to wire up the switch (SW) to the developer. From version 8.0 onwards an internal 10K resistor will be added by the radio configuration software.

During set-up if using KY040 encoders, you will be asked to select the option for either thetwo or three resistor version.

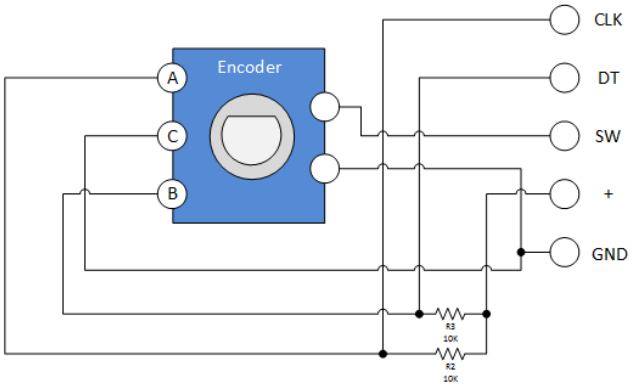


Figure 85 KY-040 Circuit Diagram

The specification shows the rotary encoders are labelled CLK(Clock), DT(Data) and + (VCC) however it is more usual to label these A, B and C.

Connect **+** to the +3.3V supply.
Do **not** connect to +5V.

Note: Some early KY-040 encoders have a 10K resistor (R1) between SW and VCC (+3.3V).

From version 7.2 onwards the internal pull-up resistors can be disabled with the radio configuration program as they are not required for the KY-040 encoder as it has its own pull-up resistors. See Figure 165 *Rotary encoder* on page 110.

Waveshare Rotation Sensor

Just another name for Rotary Encoder. Waveshare make a very similar encoder to the KY-040, however the **GND** and **VCC** connections are reversed.

Figure 86 Waveshare Rotation Sensor

The wiring is as follows:
SIA – Serial Input A
SIB - Serial Input B
SW - Knob Switch
GND – GND (0V) – Warning: Opposite to the KY-040
VCC - +3.3V " " " "

Make sure that VCC and GND are connected the correct way around otherwise you will damage the Raspberry Pi.
See *Table 8 Waveshare rotation sensor wiring* on page 50 for wiring information.

Rotary encoders with RGB LEDs

Figure 87 Sparkfun RGB rotary encoder

There are also rotary encoders available with RGB (Red, Green, Blue) LEDs which light up the transparent shaft. These also have a press switch. There are also versions which just have Red and Green LEDs. These encoders are also supported by the radio software from version 7.4 onwards.

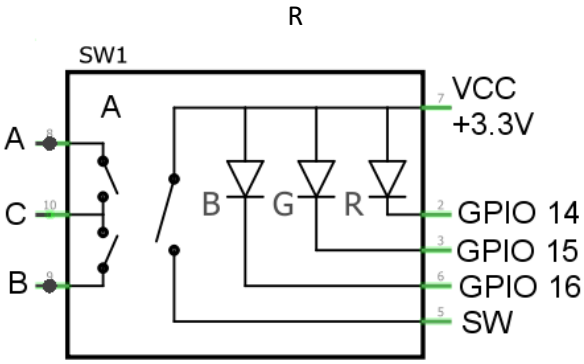


Figure 88 RGB Rotary Encoder circuit

For this type of rotary encoder, the cathodes of the LEDs are connected to VCC (3.3V). Unlike the standard encoders one side of the switch is also connected to VCC which means that the push-switch works the opposite way around to the on the standard rotary encoder. In this case the **rgb_rotary** device driver has to be selected during configuration.

Figure 89 Transparent knob

There are transparent plastic knobs available which on the standard 6mm shaft. These simply press-fit onto the shaft of the rotary encoder.

They are clear so that they can be used with illuminated rotary encoders such as the Sparkfun RGB encoder.

The LEDs can be driven by configuring the following parameters in **/etc/radiod.conf** as shown below:

```
rgb_red=14
rgb_green=15
rgb_blue=16
```

The LEDs are driven by the **status_led_class.py** driver software. The LEDs connect directly to the above GPIO pins 14, 15 and 16 and are driven by 3.3 volts so do not particularly need resistors but 100 Ohm resistors can be used to reduce the brightness.

Warning: Do not, under any circumstances connect these rotary encoders to +5 Volts, even if the specification says so, otherwise the Raspberry Pi will be irreparably damaged!

RGB I2C rotary encoders

These Rotary Encoders use three Red, Green and Blue (RGB) LEDs in the Rotary Encoder Shaft. This type of rotary encoder is entirely driven by the I2C interface. One drawback of these encoders is that they do not have a push button meaning that these must installed extra to the encoders.

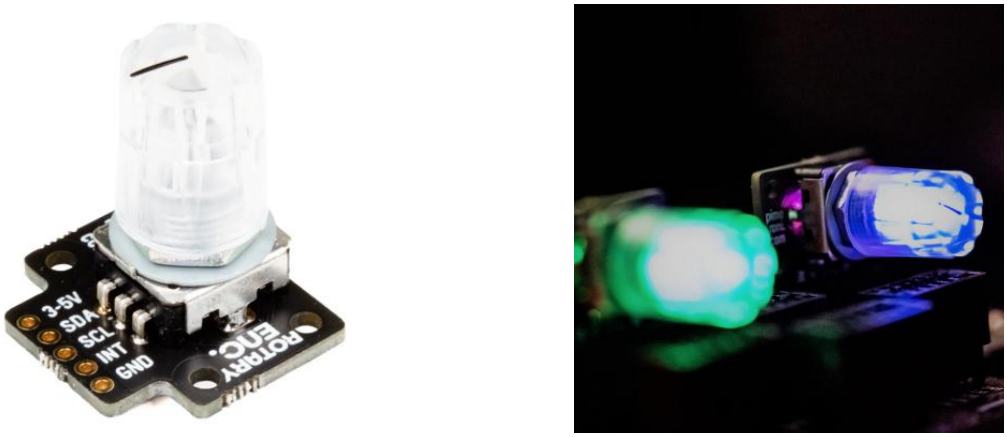


Figure 90 Pimoroni RGB I2C Rotary Encoders

These encoders contain Nuvoton MS51XB9AE microcontroller which let you directly control the RGB LEDs inside the encoder. The default I2C address is 0x0F but it is possible to configure a second device for the Channel encoder with a separate I2C address (0x1F for the radio project). The encoders also use an interrupt pin (INT). The default interrupts are GPIO22 for the Volume control and GPIO23 for the Channel interrupt. See *Table 9 RGB I2C Rotary Encoders wiring* on page 50.

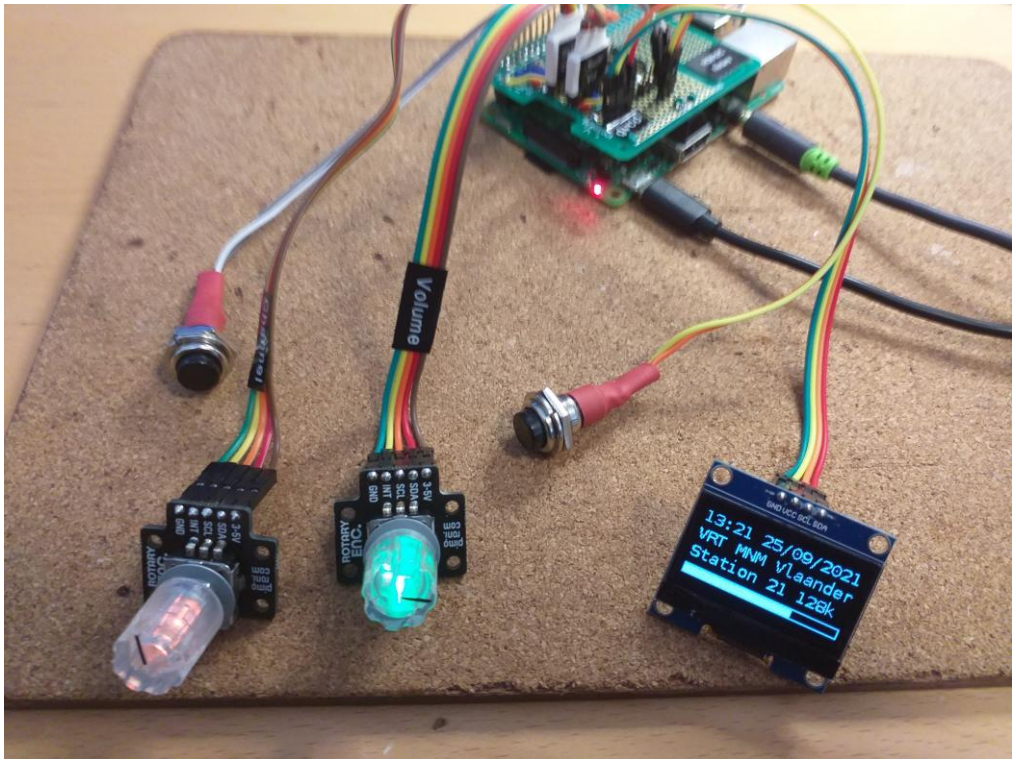


Figure 91 Radio project using RGB I2C rotary encoders

The above photo shows two RGB I2C encoders (Volume 0x0F and Channel 0x1F) running the radio software. The Mute and Menu buttons are wired the same as the standard encoders. Also shown is 128x64 pixel OLED also connected by I2C (In this case with 0x3C as the address).

Rotary encoder wiring

The following tables show how to wire up the four types of rotary encoders used in this project. The first table shows how to wire up conventional rotary encoders (without pull-up resistors). These have five connections A, B, C and two connections to the switch. See *Figure 81* on page 45.

Table 5 Radio A, B, C Rotary Encoder Wiring

GPIO Pin	Description	Radio Function	Volume Rotary Encoder	Channel Rotary Encoder
6	GND	Zero volts	Common C	
8	GPIO 14	Volume up/down	Output A	
10	GPIO 15	Volume up/down	Output B	
7	GPIO 4	Mute volume	Knob Switch 1 *	
GND	GND 0V	Mute volume	Knob Switch 2 *	
9	GND	Zero volts		Common C
16	GPIO 23	Channel up/down		Output A
18	GPIO 24	Channel up/down		Output B
11	GPIO 17	Menu switch		Knob Switch 1 *
GND	GND 0V	Menu switch GND		Knob Switch 2 *

GND is found on physical pins 6, 9, 14, 20, 25, 30, 34 and 39.

* Note: In the case of RGB LED Rotary Encoders one side of the push switch is wired to the VCC (3.3V) supply internally in the switch. The GPIO goes from low to high when the button is pushed.

Table 6 RGB LED Rotary Encoder switch wiring

7	GPIO 4	Mute volume	Knob Switch 1	
+3.3V	VCC	" "	Internal connection	
11	GPIO 17	Menu switch		Knob Switch 1
+3.3V	VCC	" "		Internal connection

The second table shows how to wire up KY-040 rotary encoders (fitted with pull-up resistors). These have five connections namely CLK, DT, SW, VCC and GND. See *Figure 85 KY-040 Circuit Diagram* on page 46.

Table 7 KY-040 Rotary Encoder Wiring

GPIO Pin	Description	Radio Function	Volume Rotary Encoder	Channel Rotary Encoder
1	3V3	+3.3V supply	+3.3V (VCC)	
6	GND	Zero volts	Common (GND)	
7	GPIO 4	Mute volume	Knob Switch (SW)	
8	GPIO 14 (TX)	Volume up/down	Output A (CLK)	
10	GPIO 15 (RX)	Volume up/down	Output B (DT)	
17	3V3	+3.3V supply		+3.3V (VCC 3.3V)
9	GND	Zero volts		Common C (GND)
11	GPIO 17	Menu switch		Knob Switch (SW)
16	GPIO 23	Channel up/down		Output A (CLK)
18	GPIO 24	Channel up/down		Output B (DT)

Table 8 Waveshare rotation sensor wiring

GPIO Pin	Description	Radio Function	Volume Rotary Encoder	Channel Rotary Encoder
1	3V3	+3.3V supply	+3.3V (VCC)	
6	GND	Zero volts	Common (GND)	
7	GPIO 4	Mute volume	Knob Switch (SW)	
8	GPIO 14 (TX)	Volume up/down	Input B (SIB)	
10	GPIO 15 (RX)	Volume up/down	Input A (SIA)	
17	3V3	+3.3V supply		+3.3V (VCC 3.3V)
9	GND	Zero volts		Common C (GND)
11	GPIO 17	Menu switch		Knob Switch (SW)
16	GPIO 23	Channel up/down		Input B (SIB)
18	GPIO 24	Channel up/down		Input A (SIA)

Warning: GND and VCC are the opposite way around to a KY-040 Rotary Encoder.

Table 9 RGB I2C Rotary Encoders wiring

GPIO Pin	Description	Radio Function	Volume RGB I2C Rotary Encoder	Channel RGB I2C Rotary Encoder	Optional I2C display
1	3V3	+3.3V supply	VCC +3.3V	VCC +3.3V	VCC +3.3V
3	GPIO2	SDA	SDA	SDA	SDA
5	GPIO3	SCL	SCL	SCL	SCL

22	Volume Interrupt	n/a	INT		
23	Channel Interrupt	n/a		INT	
6 or 9	GND	Zero volts	GND	GND	GND
3,5	I2C convection	I2C address	Hex 0x0F	Hex 0x1F	Hex 0x3C*
7	GPIO4	Mute	Mute Button		
11	GPIO17	Menu		Menu Button	

* Example OLED I2C address which will differ depending upon the attached device.
The above table shows two RGB I2C encoders. The I2C address for the Channel Rotary encoder is configured with the **gb_set_i2c_address.py** utility.

```
$ ./rgb_set_i2c_address.py
RGB I2C Rotary Encoder set new I2C address 0xf

Usage: sudo ./rgb_set_i2c_address.py --help
               --i2c_address=<i2c_address>
               --new_i2c_address=<new_i2c_address>

Recommended values for current and new addresses are 0x0F and 0x1F
Run "i2cdetect -y 1" to check I2C address before and after
```

First only plug in one RGB I2C Rotary encoder with hex address **0x0F**.

```
$ cd /usr/share/radiod
$ sudo ./rgb_set_i2c_address.py --i2c_address=0x0f -new_i2c_address=0x1F
```

Below is the output of the **i2cdetect -y 1** utility with two RGB I2C rotary encoders (**0x0F** and **0x1F**) and an OLED display (0x3C)

```
$ i2cdetect -y 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- 0f
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- 1f
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- 3c -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

Optical Rotary encoders

Optical rotary encoders are much more expensive than the mechanical (conductive) rotary encoders previously shown. They offer the higher resolution compared to mechanical ones. They are usually used for scientific and industrial applications. They are overkill for this project but may be used. This software has been successfully tested with an HRPG-ASCA #16F optical encoder from Avago. See https://media.digikey.com/pdf/Data%20Sheets/Avago%20PDFs/HRPG_Series.pdf

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 (**Error! Reference source not found.** and Table 4 respectively). For all new 40-pin Raspberry Pi's the 4

0-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 10 LCD module wiring fo 40-pin Raspberry Pi's

LCD Pin	GPIO 40-pin	Pin 40 #	Description
1	n/a	6	Ground (0V) – Wire this directly to LCD pin 5
2	n/a	2	VCC +5V
3	n/a	Note1	Contrast adjustment (0V gives maximum contrast)
4	GPIO7	26	Register Select (RS). RS=0: Command, RS=1: Data
5	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	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	GPIO5	29	Data Bit 4 (D4)
12	GPIO6	31	Data Bit 5 (D5) Note if using IQaudIO products GPIO22 conflicts !!
13	GPIO12	32	Data Bit 6 (D6)
14	GPIO13	33	Data Bit 7 (D7)
15	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	LED Backlight Cathode (GND) or Red LED [2]
17			Optional Green LED (Adafruit RGB plate) [2]
18			Optional Blue LED (Adafruit RGB plate) [2]

Note 1: If using the Midas display with VEE on pin 15 do not connect this pin to the +5V supply or you will damage the display. Connect pin 15 as shown in the section called *Midas LCD display* on page 20.

Note 2: The contrast pin 3 (VE) should be connected to the center pin of a 10K potentiometer. Connect the other two pins of the potentiometer to 5v (VDD) and 0v (VSS) respectively. Adjust the preset potentiometer for the best contrast.

Note 3: The standard LCD comes with 16 pins. Adafruit supply an RGB backlit LCD with two extra pins namely pins 17 and 18. These are non-standard. These must be wired to ground (0 Volts) to work. For more information see the section *Using the Adafruit* backlit RGB LCD display on page 76.

The following diagram (Courtesy protostack.com) shows the electrical connections for the standard 16 pin LCD. Do not use this diagram for Midas displays. See instead *Midas LCD display* on page 20.

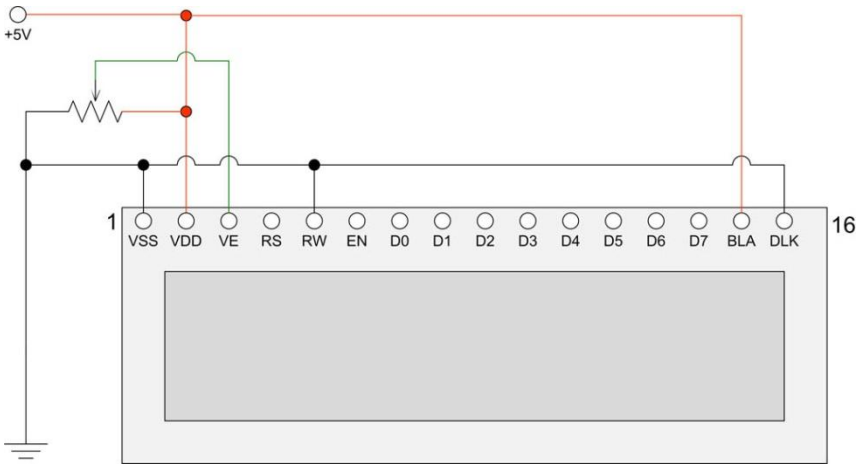


Figure 92 HD44780U LCD electrical circuit

The standard LCD comes with 16 pins. Adafruit supply an RGB backlit LCD with two extra pins namely pins 17 and 18. These are non-standard. These must be wired to ground (GND 0 Volts) to work. For more information see the section *Using the Adafruit* backlit RGB LCD display on page 76.

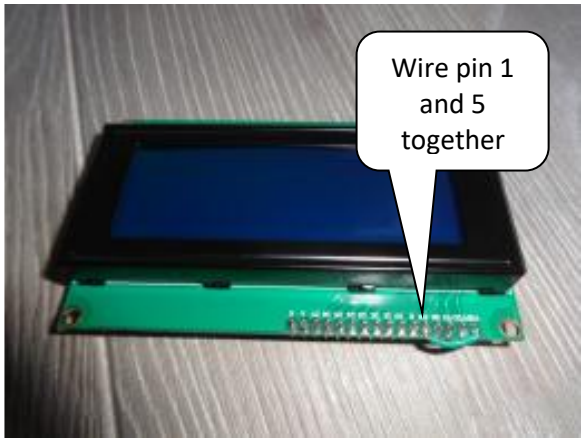


Figure 93 Wire LCD pin 1 (GND) and 5 (RW) together

The Read/Write (RW) pin 5 must be connected to pin 1 (0V). It is very important that this pin is grounded! If pin 5 is not grounded it will damage the Raspberry PI. Always wire LCD pin 5 and 1 directly together. Do not rely on grounding pin 5 with a GND wire on the connector. If this wire drops off then the LCD data lines will be put into write mode putting +5V on the GPIO pins which will probably cause irreparable damage to the Raspberry Pi. If using an I2C backpack this step is not necessary as it is already done in the backpack.

Warning – Some LCD displays such as the Midas with VEE have a different voltage arrangement for Pin 15 and Pin 5 (Contrast). Pin 15 is an output which provides a negative voltage (VEE) which connects to one end of the 10K contrast potentiometer and the other end to +5V (VDD). Connecting +5 Volts to pin 15 will destroy the LCD device. See section called *Midas LCD* display on page 20 for further information.

Note: All settings in the **/etc/radiod.conf** file use GPIO numbers and NOT physical pin numbers. So, in the above example **lcd_width** is GPIO 16 (Physical pin 36).

There is a useful program called **wiring.py** which will display physical the wiring required for the settings found in **/etc/radiod.conf**. See page **Error! Bookmark not defined.**.

Also, there is another program called **test_gpios.py** which will test for button or rotary encoder events. For example, you are not sure how you have wired a button or rotary encoder this program will confirm this for you. See page **Error! Bookmark not defined.**.

Power supply considerations

The Raspberry Pi except for the model 4B uses a standard Micro USB (type B) power connector, which runs at +5V. The model 4B uses a 5 Volt 3 Ampere power supply with a USB-C adaptor. In general, the Raspberry PI can draw up to 2.5. Many telephone adapters are not suitable. You also need to consider the LCD screen which can also need up to 20mA but depends on the type of backlight.

Try to find a power adapter that delivers at least 1.5 Ampere. As mentioned above a 2.5A supply will be required to run a Raspberry Pi model 5.

http://elinux.org/RPi_VerifiedPeripherals#Power_adapters

The Raspberry PI can be powered either the USB port or via the GPIO header (Pin 2 or 4). Some prototyping boards used to provide power in this way.

If using an adaptor or separate 5-volt Power Supply try to use a switched-mode power supply adaptor. This takes less current and generate less heat that a power dissipation device. If a power supply is designed to be earthed then use a 3-core cable with live, neutral and earth wires.

Things not to do:

- Do not try to tap off power from the Power supply or transformer used by the speaker's amplifier. This won't work (earth loops) and can cause damage to the PI and peripherals.
- Do not tap off (cascade) from the amplifier DC supply (12 volts for example) with another 5V voltage regulator. This will most likely cause interference.
- Do not feed power to the PI from two sources (USB hub and Power adapter). Try to use USB hubs that don't feed 5 volts back through the USB ports of the Raspberry PI
- Do not connect an untested power supply to the Raspberry PI without checking the voltage first.

Things to do:

- Use double pole mains switches for isolating the mains supply when switched off. A lot of European plugs can be reversed leaving the live wire un-switched if using a single pole switch.
- If using a metal case always earth it and use a three-pin plug with earth pin.
- In general feed the 5-volt supply via the Raspberry Pi rather than via the GPIO header. This is because the Raspberry Pi is fitted with a so-called poly-fuse for protection.

You should try to use a single power supply switch for the radio. Connect the AC power supply of the adaptor to the mains switch. This switch can also provide the mains supply to the speaker amplifier. Also see the section on *preventing electrical interference* on page 73

Always consider safety first and make sure that no-one including yourself can receive an electric shock from your project including when the case is open. Also see disclaimer on page 224.

Also see *Appendix B – Using a battery pack* on page 232

GPIO Hardware Notes

The following shows the pin outs for the GPIO pins on revision 1 and 2 boards. For more information see: http://elinux.org/RPi_Low-level_peripherals.

GPIO Numbers

Raspberry Pi B
Rev 1 P1 GPIO Header

	Pin No.		
3.3V	1	2	5V
GPIO0	3	4	5V
GPIO1	5	6	GND
GPIO4	7	8	GPIO14
GND	9	10	GPIO15
GPIO17	11	12	GPIO18
GPIO21	13	14	GND
GPIO22	15	16	GPIO23
3.3V	17	18	GPIO24
GPIO10	19	20	GND
GPIO9	21	22	GPIO25
GPIO11	23	24	GPIO8
GND	25	26	GPIO7

Raspberry Pi A/B
Rev 2 P1 GPIO Header

	Pin No.		
3.3V	1	2	5V
GPIO2	3	4	5V
GPIO3	5	6	GND
GPIO4	7	8	GPIO14
GND	9	10	GPIO15
GPIO17	11	12	GPIO18
GPIO27	13	14	GND
GPIO22	15	16	GPIO23
3.3V	17	18	GPIO24
GPIO10	19	20	GND
GPIO9	21	22	GPIO25
GPIO11	23	24	GPIO8
GND	25	26	GPIO7

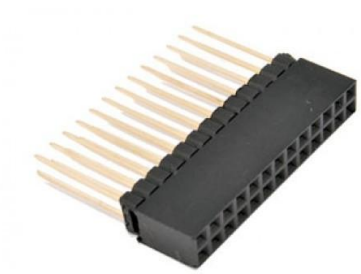
Raspberry Pi B+
B+ J8 GPIO Header

	Pin No.		
3.3V	1	2	5V
GPIO2	3	4	5V
GPIO3	5	6	GND
GPIO4	7	8	GPIO14
GND	9	10	GPIO15
GPIO17	11	12	GPIO18
GPIO27	13	14	GND
GPIO22	15	16	GPIO23
3.3V	17	18	GPIO24
GPIO10	19	20	GND
GPIO9	21	22	GPIO25
GPIO11	23	24	GPIO8
GND	25	26	GPIO7
DNC	27	28	DNC
GPIO5	29	30	GND
GPIO6	31	32	GPIO12
GPIO13	33	34	GND
GPIO19	35	36	GPIO16
GPIO26	37	38	GPIO20
GND	39	40	GPIO21

Key	
Power +	UART
GND	SPI
I²C	GPIO

Figure 94 GPIO Numbers

Note: The B+, 2B, 3B, 4B and P400 have the same pin-outs.



If connecting any 40-pin interface board via a 26-way ribbon cable it will be necessary to fit a 26-pin header extender and plug the ribbon cable into it.

Figure 95 26-pin header extender

Chapter 4 - Construction details

Contents chapter 4 **Page**

Construction using an interface board

It isn't necessary to construct the radio connect via an interface board but it does make maintenance easier and is much more reliable and will be needed if you wish to connect to the Raspberry Pi using a ribbon cable. Always bring the ribbon cable into the top of the board as shown in Figure 96 below. If the ribbon cable is connected to the back (underside) of the board the two rows of the 40-pin connector will be swapped over. In the next section how to use breakout boards is covered which may be an easier alternative for many constructors rather than building your own.

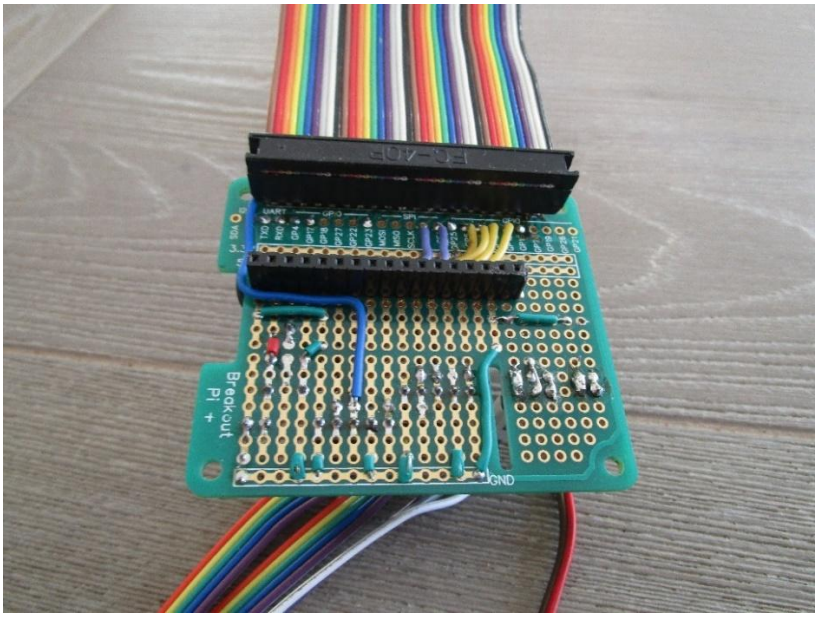


Figure 96 40-pin Interface board with ribbon cable

The figure below shows a 4x20 HD44780U LCD plugged into the interface board.

Figure 97 Interface board with LCD screen attached

Below is the other side of the interface board showing the connections to the rotary encoders, an IR sensor the IR activity LED.

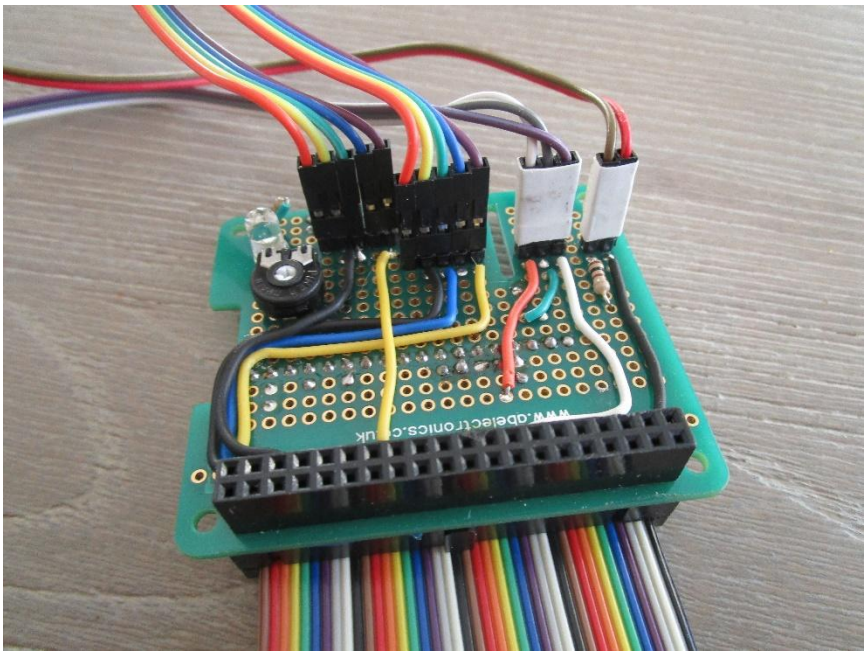


Figure 98 Radio controls connections

Below is the complete overview of the interface card with some test rotary encoders, an IR sensor the IR activity LED.

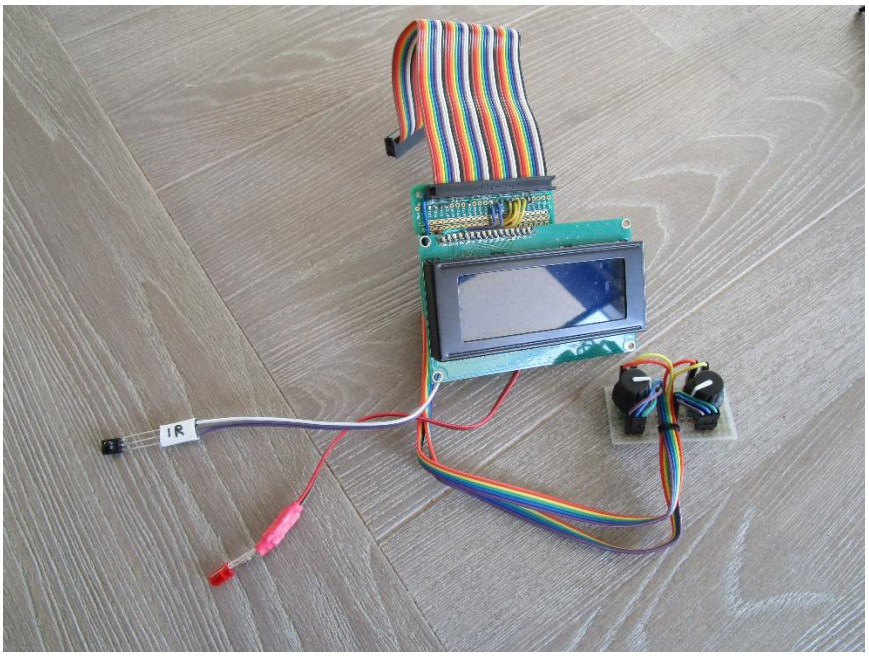


Figure 99 Interface board overview

There are various interface boards available on the market for both 26-pin and 40-pin Raspberry PIs.

Construction using breakout boards

When this project was begun in the early days of Raspberry Pi there was very little in the way of breakout boards. It was necessary to make connections to button, LCDs and Rotary encoders either direct on the GPIO header or via a specially constructed breakout board. Things became more complex when digital sound cards (DAC) were introduced as these occupied the GPIO header and either did not extend the header pins, or if they did so, only extended a few of them.

This has now changed and there is a wide variety of breakout boards available.

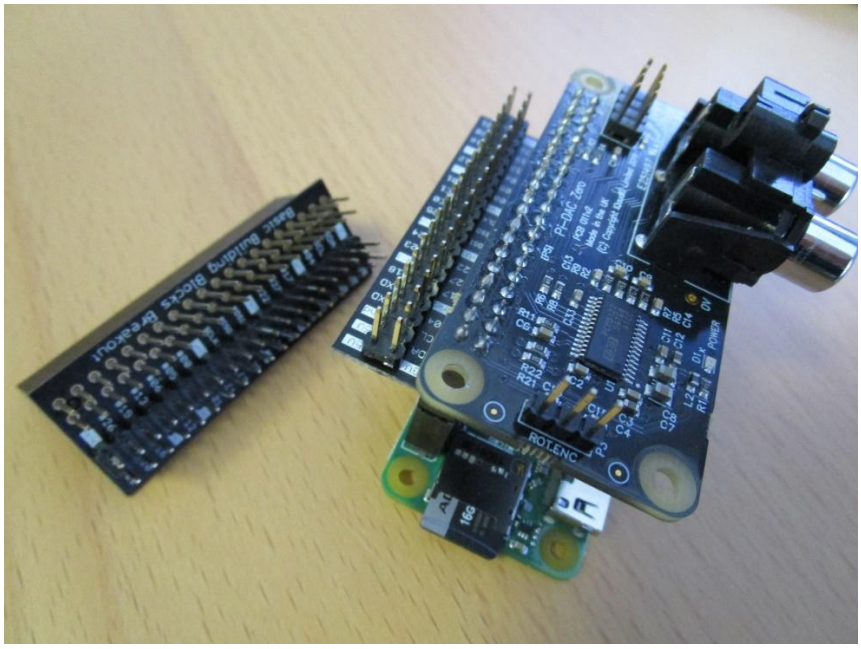


Figure 100 GPIO header breakout board

On the left of the above photo is an example of the 4Tronix GPIO breakout and extender board. On the right of the same photo is the break-out board used with a Raspberry Pi Zero W and an IQaudIO DAC plus. All 40-pins are now made available, except for those used by the DAC, to attach buttons and the like to the GPIO pins.

These boards are available from http://4tronix.co.uk

See:
https://shop.4tronix.co.uk/products/gpio-interceptor-gpio-breakout-for-40-pin-raspberry-pi

Note: Soldering skills are required to solder the 40-pin header to the breakout board. The above breakout board is only shown as an example and many more are available on the Internet.

Construction using an Adafruit LCD plate

Introduction

This section is for the radio using an Adafruit RGB-backlit LCD plate for Raspberry PI. The complete instructions for both ordering and building this product are available from the following Web site: http://www.adafruit.com/products/1110 (See tutorials)

Note: Don't confuse this product (which has an I2C interface chip) with the two-line and four-line RGB LCDs which Adafruit also sell.

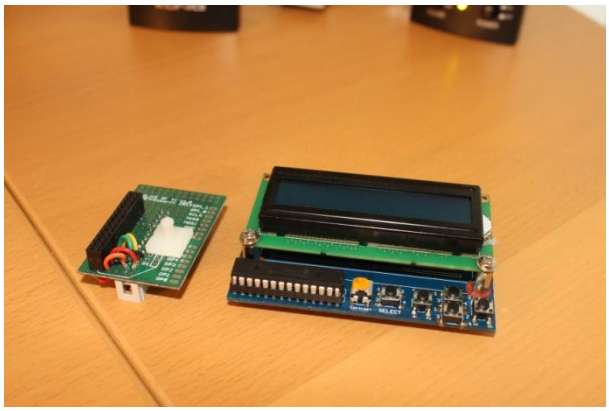


Figure 101 Adafruit LCD plate

The Adafruit LCD plate is designed to directly into the GPIO header on the Raspberry PI. These fit into a female 26 way header. If you want to connect the Adafruit LCD via a ribbon cable you will need to mount a 26 way male header instead of the female header and you will also need to construct a reversing board (Shown on the left of the picture on the left). Because ribbon cable swaps the two rows of pins over the reversing card is required to swap the two rows of pins back to their correct orientation.

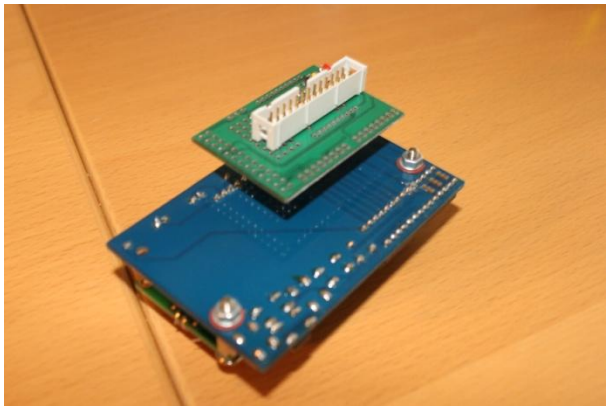


Figure 102 Adafruit LCD plate with ribbon cable adapter

Back view of the reversing board plugged into the Adafruit LCD plate.

The GPIO pins used are:

1. 3.3 Volts
2. 5.0 volts
3. SDA0
4. -
5. SCL0
6. Ground

Note 1: If you are going to plug the Adafruit LCD plate directly into the GPIO header on the Raspberry PI then you don't need the above reversing plate. Just follow the construction instructions on the tutorial on the Adafruit site.

Note 2: The "Select" button on the Adafruit plate is the "Menu" button for the radio.

Using other switches

The Adafruit Plate comes with five 4-pin switches which are mounted on the interface board. You will almost certainly want to use other switches say mounted on a front panel. It doesn't matter which type of switch you use as long as it is a push to make type. The only reason that a four-connector switch is used is for mechanical strength. If you look closely, you will see push button symbol between pins 2 and 4 and 1 or 3 on the component side for four of the switches. Either 2 and 4 and 1 or 3 should be connected to the switches.

It is advisable to solder two posts (male pins) for each switch on the reverse side of the board (The non-component side). Don't solder wires directly into the board. It is better to use push-on jumper wires connected to the switches to connect to the posts on the card.

Note: Rotary encoders cannot be used with the Adafruit Plate as these require three connections and the Adafruit routines to utilise them are not supplied by Adafruit.

Using the Adafruit LCD plate with the model B+, 2B and 3B

The plate is designed for revisions of the Raspberry Pi. It uses the I2C (SDA/SCL) pins. Adafruit supply a special extra tall 26-pin header so the plate sits above the USB and Ethernet jacks. For Pi Model B+, the resistors sit right above the new set of USB ports. To keep them from shorting against the metal, a piece of electrical tape must be placed on the top of the USB ports.

Construction using an I2C LCD backpack

Skip this section if you are not using an I2C backpack. There are two versions of the backpack supported:

1. Adafruit I2C backpack using an MCP23017 port expander – Hex address 0x20
2. Arduino I2C backpack using a PCF8574 port expander – Hex address 0x27 or 0x37

The I2C interface only requires two signals namely the I2C Data and Clock. This saves six GPIO pins when compared with the directly wired LCD interface. See https://www.adafruit.com/product/292.

The radio software also supports the more common PCF8574 chip-based backpack popular with the Arduino hobby computer may also be used. See http://www.play-zone.ch/en/i2c-backpack-pcf8574t-fur-1602-lcds-5v.html for example.

This is configurable in the **/etc/radiod.conf** file.

```
# The i2cbackpack is either ADAFRUIT or PCF8574
# i2c_backpack=PCF8574
i2c_backpack=ADAFRUIT
```

Note: In previous versions the **i2c_backpack** parameter was incorrectly shown as PCF8475 instead of PCF8574. Check the **/etc/radiod.conf** file.

Adafruit I2C Backpack

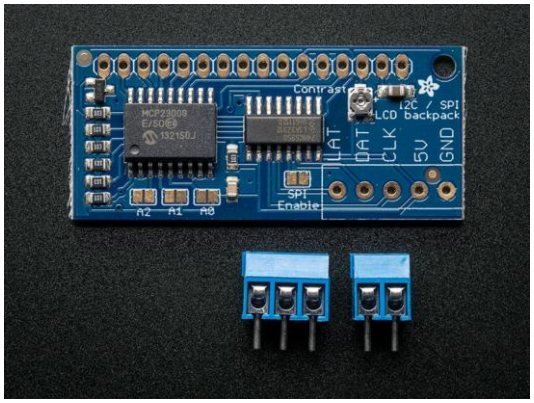


Figure 103 Adafruit I2C Backpack

The Adafruit I2C/SPI backpack interface is shipped as shown in the diagram opposite. There are no connectors shipped to connect to the LCD itself to this interface. These must be ordered separately.

Order a 16 in-line connector.
The Adafruit backpack also supports the SPI interface but is not used in this project

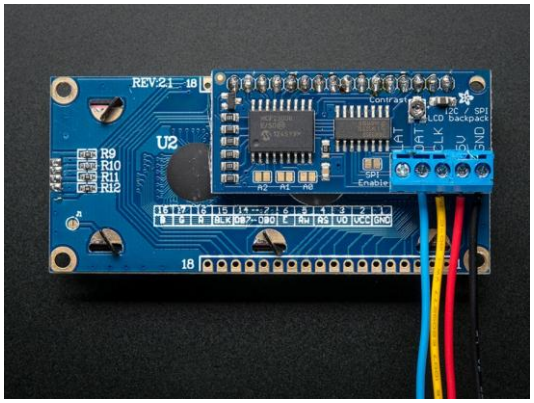


Figure 104 LCD connected to an Adafruit I2C backpack

The diagram shown on the left shows a 2x16 character LCD connected to the I2C backpack. The wiring right to left is:

1. LAT (Unused)
2. Blue: I2C Data – GPIO 2 (pin 3)
3. Yellow: I2C Clock – GPIO 3 (pin 5)
4. Red: +5 volts – GPIO header pin 2
5. Black: GND (0v) – GND GPIO header pin 6

The I2C Data (DAT) connects to pin 3 on the Raspberry Pi GPIO header and the I2C Clock (CLK) to pin 5 on the GPIO header.

Arduino PCF8574 I2C backpacks

These types of backpack are popular with Arduino users. The device address is usually hex 0x27. Another manufacture may use hex 0x37. This is configurable in the radio configuration program.

Figure 105 Arduino I2C backpack

The wiring From top to bottom is:

7. GND (0 volts) – GPIO header pin 6
8. VCC +5 volts – GPIO header pin 2
9. SDA I2C Data – GPIO 2 (pin 3)
10. SCL I2C Clock – GPIO 3 (pin 5)

The blue potentiometer on the right is the contrast adjustment.

```
i2c_backpack=PCF8574
#i2c_backpack=ADAFRUIT
```

To use this device either amend the **i2c_backpack** parameter in **/etc/radiod.conf** (Comment out the ADAFRUIT line) or run the **configure_radio.sh** program.

Table 11 I2C Backpack connections

Backpack	Label	Description	GPIO	Physical pin
1	GND	Zero volts	-	14
2	VCC	+5 Volts	-	4
3	SDA	I2C Data	2	3
4	SCL	I2C Clock	3	5
5	LAT	Latch – Not used	-	-

The LAT pin is for the Adafruit backpack only and is for the SPI interface but isn't used by I2C.

Creating the interface board for the I2C back pack

An interface board is recommended to connect the I2C backpack and rotary encoders etc. to the GPIO interface. Any number of Raspberry Pi prototyping boards are available for all versions of the Raspberry Pi. The Ciseco Humble Pi prototype board shown in Figure 106 has been discontinued.

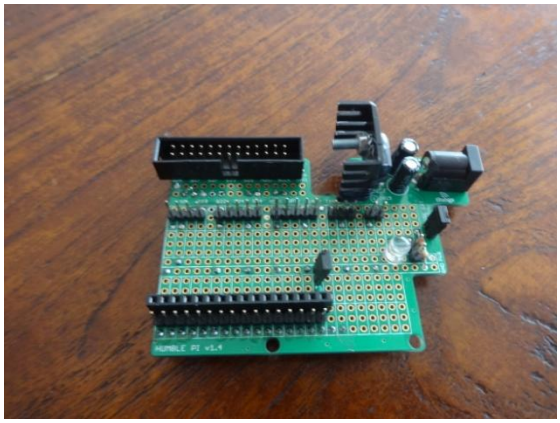


Figure 106 Ciseco Humble PI I2C interface board

Figure 107 The I2C backpack interface board

The above figure shows the I2C interface board using the Ciseco Humble PI (Discontinued). The header pins in the centre from left to right are, I2C interface connector (4 pins), Volume rotary encoder (5 pins), Channel rotary encoder (5 pins), IR sensor (3 pins) and front panel LED (2

The above diagram shows the Adafruit I2C backpack connected to the interface board along with the rotary encoders. The 26-pin male header connects to the GPIO ribbon cable on the Raspberry PI. On the left is a 6V to 9V power input feeding a 5 Volt regulator.

pins). In this version there are two rows of 18 pins (male and female) to allow different I2C backpack to be connected. You will normally only need one or the other.

Fitting a wake-up button

One of the features of this radio's design is that the menu button (LCD versions) or a special key (Touchscreen version) can do an orderly system shutdown. This is more desirable, and certainly safer and more convenient than pulling the power plug out. The system when properly shutdown goes into a so-called halt state. If the power is left connected the Raspberry Pi, it can be woken up by a button connected between pins 5(GPIO3) and 6(GND). GPIO3 can still be used as normal for example for I2C connections.

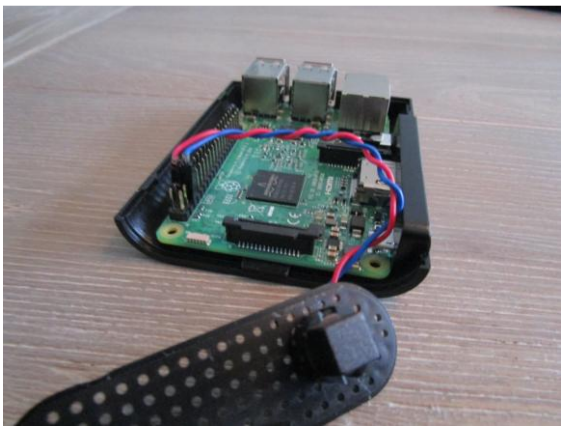


Figure 108 Wake-up button

On the left is the Raspberry Pi unit which is running the radio on a TV with HDMI inputs as shown in Figure 3 on page 7. A small black wake-up button is fitted to the case and connects to physical pins 5 and 6. When pressed with the Raspberry Pi in a halt state but power still applied it will start its boot-up sequence. It should be noted that pin 5 (GPIO3) is also used as the I2C data line. Although the button could still be fitted it is probably not a good idea as it will disrupt the I2C signal if the wake-up button is pressed.

Installing an IR sensor and remote control

Note: This installation procedure is only for LCD versions of the radio. If using a HDMI or Touchscreen display see *Installing the FLIRC USB remote* control on page 69.

IR Sensor

If you wish to use an IR remote control with other variants of the radio then purchase an IR sensor TSOP38238 or similar. The output pin connectivity depends on the exact hardware being used. See *Table 16 IR Remote Control Sensor* Pin outs on page 140 for the GPIO pin connection.

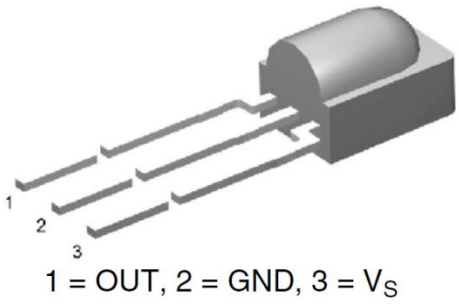


Figure 109 TSOP38238 IR sensor

The TSOP38xxx series works from 2.5 to 5.5 volts and is ideal for use the Raspberry PI.

IR sensor	Description	RPi
Pin 1	Signal Out	GPIO in *
Pin 2	Ground	Pin 6
Pin 3 **	Vs 3.3 Volts	Pin 1

* See on *Table 16 IR Remote Control Sensor* Pin outs page 140.

** Caution; Do not accidently connect to 5 volts

There are equivalent devices on the market such as the TSOP4838 which operate on 3.3 volts only.

See http://www.vishay.com/docs/82491/tsop382.pdf for more information on these IR sensors.

Tip: These IR sensors are very prone to damage by heat when soldering them. It is a good idea to use a 3-pin female connector and push the legs of the IR detector into them. If you solder the IR detector directly into a circuit then take precautions by connecting a crocodile clip across each pin in turn whilst soldering it. See figure below:

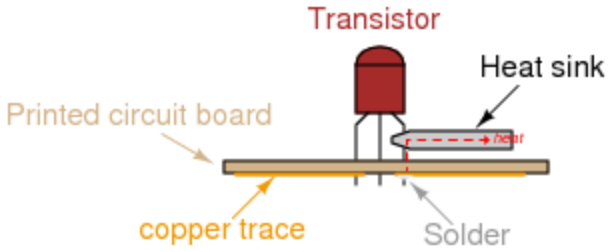


Figure 110 Soldering precautions

Remote control

Almost any surplus IR remote control can be used with this project although it is recommended to use the Raspberry Pi Mini IR remote control. Later on, it is explained how to set up the remote control with the radio software. You will need to install the software for IR sensor. See the section called ***Error! R eference source not found.****Installing the IR remote control software*140 on page 140.

Remote Control Activity LED

If wanted an activity LED can be connected to GPIO 11 or 13 depending on the type of radio. This flashes every remote control activity is detected. It is a good idea to include this.

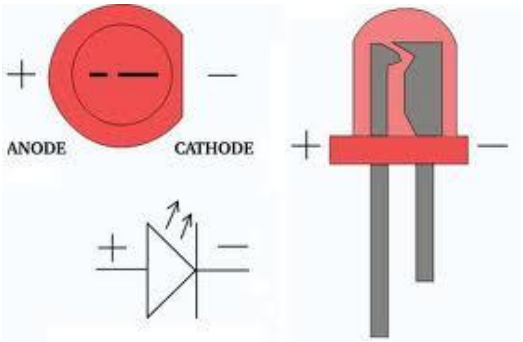


Figure 111 LED polarity

LEDs have polarity and must be wired correctly to work. The diagram shows the polarity of a typical LED. The longer lead is the positive (+) connection and connects to the Anode (The smaller terminal inside the LED)
Also, the LED must be wired in series with a resistor to limit the current, typically 100 Ohms is OK. Failure to do this may cause the LED to burn brightly for a while then burn out. Connect the cathode to GND (RPi Pin 6) and the Anode (+) to the GPIO pin shown in the following table via a 100 Ohm resistor.

The following table shows the GPIO pin used for the LED connections.

Table 12 Remote Control Activity LED

Radio Type	Pin	GPIO	Type of Raspberry PI
Activity LED not fitted	none	n/a	Not applicable
Two- or Four-line LCD with Push Buttons	23	**11**	26 or 40-pin version
Two- or Four-line LCD with Rotary encoders	23	**11**	26 or 40-pin version
Two- or Four-line LCD with I2C backpack	23	**11**	26 or 40-pin version
Adafruit RGB plate with push buttons	33	**13**	40-pin version only
Vintage radio with no LCD display	16	**23**	26 or 40-pin version
Designs using IQaudIO etc. sound boards	36	**16**	40-pin version only
PiFace CAD with IR sensor	16	**23**	26 or 40-pin version

How to configure the LED is shown on in the section called **Error! Reference source not found.** on page **Error! Bookmark not defined.**.

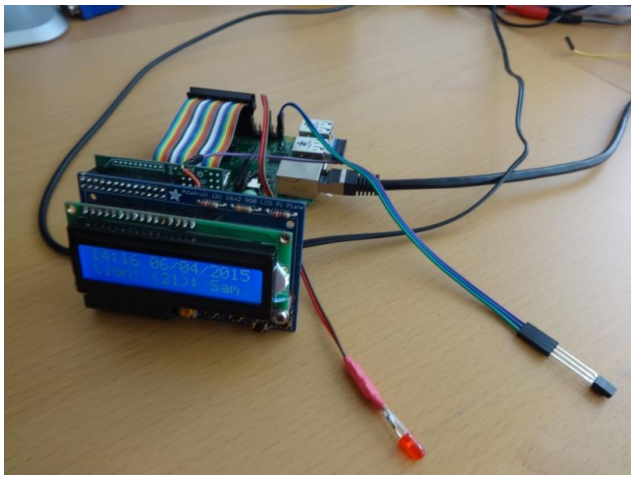


Figure 112 Adafruit plate with IR sensor and activity LED.

The illustration on the left shows an Adafruit RGB plate with IR sensor and activity LED.

The IR sensor picks up 3.3 volts from the reversing plate and connects the signal output to Pin 40 (GPIO 21) and GND (pin 39).

The LED connects to pin 33 (GPIO 13) and ground (pin 34).

Construction using an IQaudIO Cosmic Controller

IQaudIO manufacture a comprehensive range of sound devices and controller boards. See their Web site at: http://www.iqaudio.co.uk/

The radio software provides support for the IQaudIO Cosmic controller.
The IQaudIO Cosmic controller consists of the following:

- A three push-button interface (Channel UP/DOWN and Menu)
- A rotary encoder (Used as volume control)
- Three status LEDs (Normal, Busy and Error)
- A 128 by 64 pixel OLED display (I2C interface)
- Optional IR detector (Ordered separately)

Figure 113 IQaudIO Cosmic controller and OLED display

The main advantage of this hardware, is that it contains everything required by the radio software.

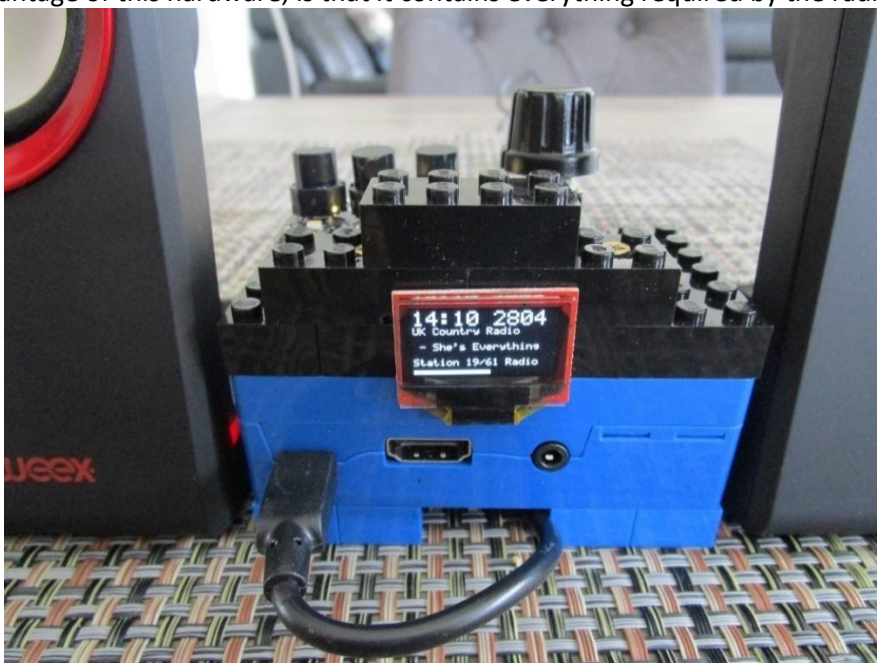


Figure 114 Lego radio with IQaudIO Cosmic controller and OLED

If using the IQaudIO Cosmic controller it is necessary to configure it as shown in the section **Error! Reference source not found.** on page **Error! Bookmark not defined.**.

Construction using the Pimoroni Pirate radio

A full set of instruction for building the Pimoroni Pirate radio with pHat BEAT can be found here: https://learn.pimoroni.com/tutorial/sandyj/assembling-pirate-radio

Soldering skills are required.

Construction using the PiFace CAD

Fortunately, no soldering or construction is required with the PiFace CAD. Just plug it in and install and run the software. The PiFace CAD also has an IR sensor which means that it can be used with a remote control. It is however more sluggish in its operation when compared to other variants of the radio as the SPI interface on the Raspberry Pi is fairly slow. It also has the disadvantage that the push buttons are on the bottom of the unit.

Figure 115 PiFace CAD and Raspberry PI

Figure 116 PiFace CAD in a case

Various ready-made cases are available from various suppliers. Warning: not all fit properly and might require some modification.

The PiFace CAD uses the SPI interface (from Motorola)
See http://en.wikipedia.org/wiki/Serial_Peripheral_Interface_Bus and the section called *Installing PiFace CAD software* on page 124 for further information on SPI.

Installing the FLIRC USB remote control

Note: This installation procedure is only for the HDMI or Touchscreen display of the radio. If using an LCD display see *Installing an IR sensor and remote control* on page 65.

FLIRC USB Remote Control dongle

The FLIRC USB dongle allows the use of any remote control with your Raspberry Pi. In this design it is intended for use with the graphical version of the radio (Touch-screen or HDMI displays). It allows button presses on a remote control to be mapped to the keyboard input of the Raspberry Pi. For example, pressing the volume up button on the remote control will act just like pressing the + key on keyboard (If so mapped). The graphical version of the radio accepts key presses. FLIRC will not work with the LCD versions of the program. This may however change in a later version.

More can be found at the FLIRC Web site: https://flirc.tv
Installation documentation can be found at: https://flirc.tv/ubuntu-software-installation-guide

Note: This installation 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.**.

It is first necessary to install FLIRC. First install the necessary libraries.

```
$ sudo apt install libhidapi-hidraw0 libqt5xmlpatterns5
```

Install the FLIRC software by running the following:

```
$ curl apt.flirc.tv/install.sh | sudo bash
```

The following will be displayed:

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time
Current
                                 Dload  Upload   Total   Spent    Left
Speed
100  8266  100  8266    0     0  17185      0 --:--:-- --:--:-- --:--:--
17149
:
:
Distribution: debian
Checking for curl...
Detected curl...
Installing flirc deb-repo...
Running apt update... done.
Installing apt-transport-https... done.
Installing /etc/apt/sources.list.d/flirc_fury.list...done.
Running apt update... done.
```

Answer Y to the following question:

```
Do you want to install the flirc utilities? [Y/n]
```

Reboot the Raspberry pi in Desktop mode (Important).

On the desktop open **Programs → Accessories → Flirc**

Note: At this stage you may see a pop-up window offering a Firmware Upgrade. Answer Yes to upgrade the firmware in the FLIRC dongle. Failure to do so may result in the Dongle not connecting.

Now see *Configuration of the FLIRC USB dongle* on page 161.

Construction Tips and Tricks

This section contains some construction tips which may be useful. It goes without saying that having the correct tools such as a good fine tipped soldering iron, wire strippers and the like will greatly help constructing the radio.

Wiring up rotary encoders and switches

Figure 117 Rotary encoder wiring components

Purchase prototype board jumper wires with at least one end fitted with female connectors. The best type is the ribbon type particularly in the case of rotary encoders. Strip off five wires for the rotary encoder or two for push buttons. Also, it is better to use shrink wrap to cover the wires after they have been soldered onto the switch or rotary encoder. This improves both insulation and strength.

Figure 118 Using wire strippers

Cut the plugs off the end that is to be soldered to the rotary encoder or switch. Leave the other end with *female* connectors. Using good wire strippers, strip a few millimetres off the wires. Separate the wires for about 30 millimetres.

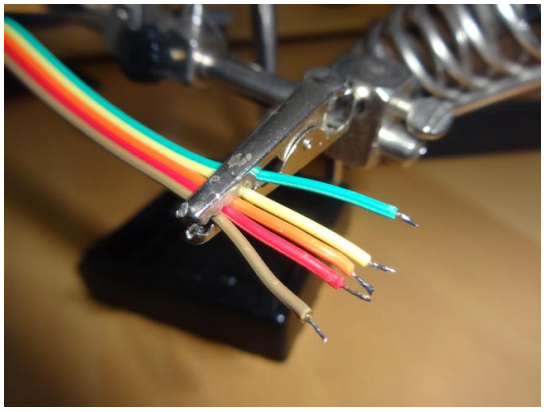


Figure 119 Tinning the wires with solder

Twist the copper strands together as tightly as possible and tin the wires with a little solder. A so called “Extra pair of hands” is very useful for gripping the wires using crocodile clips.

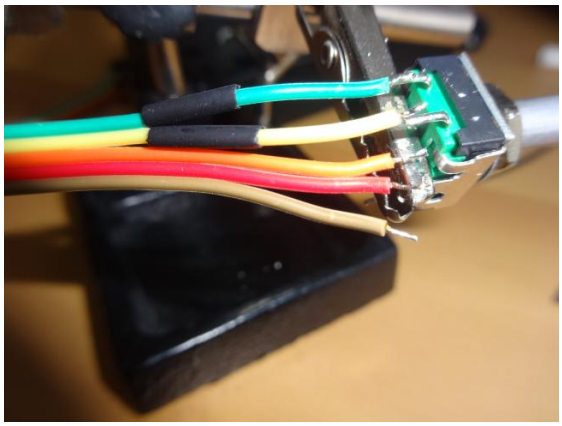


Figure 120 Soldering up the switch

Tin the switch connections with solder. Cut a few millimetres of shrink wrap and slide onto the wires to be soldered. Make sure that the shrink wrap sleeves are well away from the heat of soldering iron as these will shrink easily with the slightest bit of heat. Tack the wire onto the top of the switch connector. Don’t attempt to twist the wire around the connector. Just tack it on top with a little bit of heat from the soldering iron.

Figure 121 Shrink shrink-wrap with a hair dryer

After soldering on all wires push the shrink wrap over the newly soldered wires. Using a hair dryer heat the shrink wrap until it has completely shrunk tightly over the wires.

The finished switch can either be connected directly to the Raspberry Pi or to an interface board.

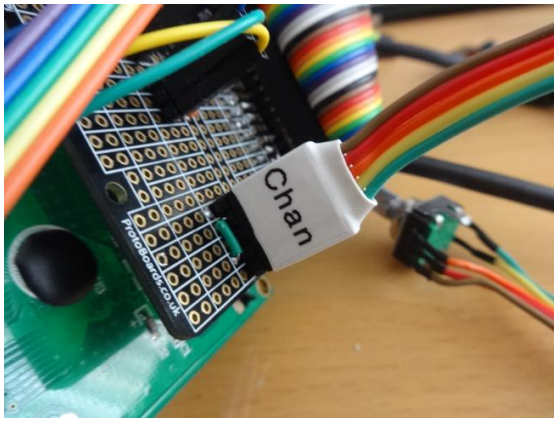


Figure 122 Connecting the rotary encoder an interface board

If using an interface board, the female connectors can be bound together to form a plug using a larger piece of shrink-wrap. Slip the shrink wrap over the female pins and heat with a hair dryer. This can then be labelled up using Dymo tape machine or a CD marker pen. Push the newly created plug onto the male pins on the interface board.

Selecting an audio amplifier

There is a wide range of amplifiers that can be used with the radio which fall into five main categories:

1. A set of PC speakers with amplifier (Logitech or similar) – the simplest option
2. A dedicated AB or Class D stereo amplifier (Velleman or similar)
3. A combined DAC (I2S) and amplifier from manufacturers such as **IQaudIO** or **HiFiBerry**
4. The audio stage of an existing (vintage) radio. Use the PA or record player input (Usually mono only)
5. Bluetooth speakers or headset

Figure 123 PC speakers

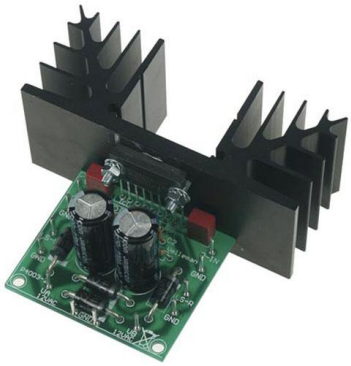


Figure 124 Velleman 30W stereo amplifier

Figure 125 IQaudIO DAC and 20W Amplifier

Figure 126 Vintage radio PA input

Figure 127 JVC Bluetooth Speakers

Figure 128 Audio output jack amplifier

The above manufacturer's products are examples only and do not imply any specific recommendations. There are an enormous number of solutions available. The choice is determined by, amongst other things, price, mono/stereo, power output and quality.

Preventing electrical interference

One of the most irritating faults that one can have is the LCD screen occasionally either going blank or displaying hieroglyphics especially when switching on and off other apparatus or lights on the same circuit. This is due to Electromagnetic Interference (EMI).
See https://en.wikipedia.org/wiki/Electromagnetic_interference for more information.

EMI can be caused by any number of sources such as fluorescent lighting, switching on and off equipment on the same circuit as the radio or even electrical storms. If you are using a standard Raspberry PI USB power supply then you will probably not experience this problem as nearly all are fitted with a ferrite core (This is the big lump in the cable or may be built into the adapter itself). If you do experience this problem then try the following solutions one at a time in the order shown below. They can all be used together if required.

Using a clip-on ferrite core on the +5 volt cable

Figure 129 Clip on ferrite core

One of the most effective solutions is to put a clip-on ferrite core on the +5V cable going to both the Raspberry Pi and USB hub. Loop the wire through at least once. Even a single loop seems to be enough. Try this first!

Figure 130 Loop +5V supply around the core

Fit a mains filter

Figure 131 Various mains filters

Try using a mains filter. This has the advantage that it can prevent spikes coming in from the mains and protect against electrical storms. The picture on the right shows an integrated filter and panel mount mains socket.

Figure 132 Integrated mains socket and filter

Use an I2C LCD backback

If all else fails replace the directly wired LCD wiring with an I2C backpack. See the section called *Construction using an I2C LCD backpack* on page 61 for further information.

Preventing ground loops

Figure 133 3.5mm Jack Ground Loop Isolator

Avoid creating ground loops in the first place during construction. Ground loop issues usually cause a humming or electronic noise. Trying to tap off the Raspberry power supply from the power supply for the amplifier (if used) is one sure way to create a ground loop. If you experience such a problem then a 3.5mm Jack Ground Loop Isolator available from suppliers such as Kenable (http://www.kenable.co.uk) can prevent unwanted hum on the audio system. Place the isolator between the Audio output of the Rasberry Pi or sound card and the amplifier input.

For further information on ground loops:
https://en.wikipedia.org/wiki/Ground_loop_(electricity)

Connecting up a USB power adapter

Figure 134 Connecting up a USB power adapter

It is convenient to connect all power supply components for the Raspberry Pi, amplifier and USB hub etc via a single mains switch. USB 240V AC to +5V power adapters are designed to connect directly to the mains and not via a mains switch. One idea is to purchase a European round pin adapter and use standard electrical connector blocks to connect the incoming AC power cable to the two pins of the power adapter. AC cables to other components such as the amplifier can also be connected to the connector blocks. Use electrical tape or shrink-wrap to isolate the connector blocks.

Cooling the Raspberry Pi

The Raspberry Pi is built from commercial chips which are qualified to different temperature ranges; the LAN9512 is specified by the manufacturers being qualified from 0°C to 70°C, while the Application Processor (AP) is qualified from -40°C to 85°C. Operation outside these temperatures is not guaranteed. The temperature of the CPU should not really go above 80°C. If it does the CPU will throttle back its processor clock speed to reduce the temperature. The official line from the Raspberry Pi is that does not require cooling even though you can get temperature warnings when using the Raspberry Pi 7-inch touch screen.

The **vcgencmd** command can be used to check the CPU temperature.

```
$ vcgencmd measure_temp
temp=67.1'C
```

For most of the radio designs in this document no specific cooling is required. If you need to cool the Raspberry Pi (Those with touchscreens in particular) then a variety of heat sinks and cooling fans are available as shown in Figure 135 below. Even with a heat sink good ventilation is necessary.

Figure 135 Heat sink kit

A variety of heat sink and cooling fan kits are available for the Raspberry Pi. Fans are a less good idea as they will produce a background hum. Also, as the CPU gets hot and the RPi is mounted vertically the adhesive softens and the fan and its heat sink tend to fall off unless secured in some way. Try to using a heat sink first.

Figure 136 Cooling fans

Miscellaneous

Simple tone regulator

It may be that you wish to fit a tone regulator to the radio. Below is one option.
The following diagram and modified text came from Jack Orman at:
http://www.muzique.com/lab/swtc.htm

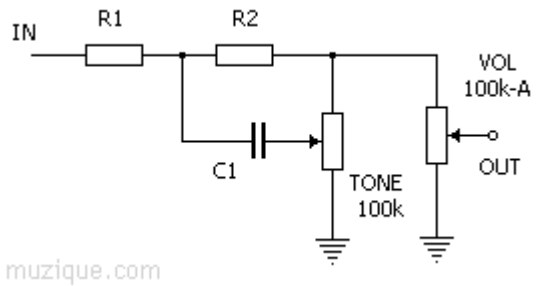


Figure 137 Simple tone control circuit

This tone control circuit that has a response that can be altered from high cut to high boost as the knob is turned. The output resistance is constant so the volume does not vary as the tone control is adjusted.

Suggested values for beginning experimentation with, are R1=10k, R2=47k, C1=0.022uF and 100k for the tone and volume pots.

Note that the above circuit has a lot of attenuation of the audio output so using the onboard audio output of the Raspberry Pi might result in a disappointing level of volume. It is recommended to use a sound output DAC or USB sound dongle.

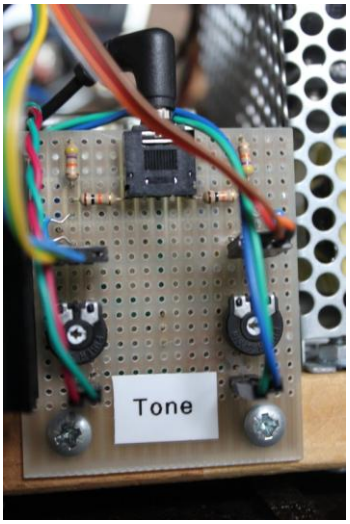


Figure 138 Tone control board

The illustration on the left shows a simple passive tone regulator board using the above circuit.

The audio output from the Raspberry Pi or DAC is fed into the board via a standard Audio socket.

Below the input are the connections to the tone regulator potentiometer mounted on the front panel of the radio.
Below the potentiometer connections are the two 100K presets for adjusting the output level to the Audio Amplifier.

Below these the Left and Right audio outputs connect to the Amplifier.

Using the Adafruit backlit RGB LCD display

The Adafruit backlit RGB LCD has three LED backlights (Red, Blue and Green) which can either be switched on individually or in various combinations together as shown in the table below:

Table 13 Adafruit backlit RGB display wiring

Switch pin	Red (Pin 16)	Green (Pin 17)	Blue (Pin 18)	Colour	Diodes required
1	0	0	0	Off	0
2	0	0	1	Blue	0
3	0	1	0	Green	0
4	0	1	1	Light Blue	2
5	1	0	0	Red	0
6	1	0	1	Purple	2
7	1	1	0	Yellow	2
8	1	1	1	White	3
Common	GND			Total diodes	9

Figure 139 IN4148 diode

The diodes used are any low voltage low current diodes such as the IN4148. So to use all of the above combinations would require a single pole 8 way rotary switch and logic and nine diodes. The first switch position is off.

- Do not wire anything to position 1.
- Wire pin 16 (Blue) of the LCD backlight to switch position 2.
- Wire pin 17 (Green) of the LCD to switch position 3
- Wire pin 18 (Red) of the LCD to switch position 5
- Wire pin 17 and 18 via two diodes to pin 4 to give the colour light blue
- Do the same for the other two-colour combinations
- Wire pin 16, 17 and 18 to pin 8 via three diodes to give the colour white
- Wire the centre pin of the switch to 0v (GND)

Chapter 5 – System Software Installation

Conventions used in this tutorial

Installation of the radio program requires you to enter lines at the command line prompt. This requires you to log into the Raspberry PI as user '**pi**'. The default password is **raspberry**.

Note: Don't carry out any of the following commands just yet. They are just examples.

```
Raspberrypi login: pi
Password: raspberry
pi@raspberrypi:~$ Last login: Sun Apr  6 10:18:18 2014 from 192.168.2.100
pi@raspberrypi:~$
```

The prompt line is displayed ending with a **$** sign. The **pi@raspberrypi:~** string means user 'pi' on host machine called 'raspberrypi'. The ~ character means the user 'pi' home directory **/home/pi**. In this tutorial if you are required to do something as user **pi** then only the $ sign will be shown followed by the command as shown in the example below:

```
$ mpc status
```

Some commands produce output which does not need to be shown. In such a case a ':' is used to indicate that some output has been omitted.

```
$ aplay -l
**** List of PLAYBACK Hardware Devices ****
: {Omitted output}
card 0: ALSA [bcm2835 ALSA], device 1: bcm2835 ALSA [bcm2835 IEC958/HDMI]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
card 1: Device [USB PnP Sound Device], device 0: USB Audio [USB Audio]
  Subdevices: 0/1
  Subdevice #0: subdevice #0
```

END OF EXAMPLE COMMANDS.

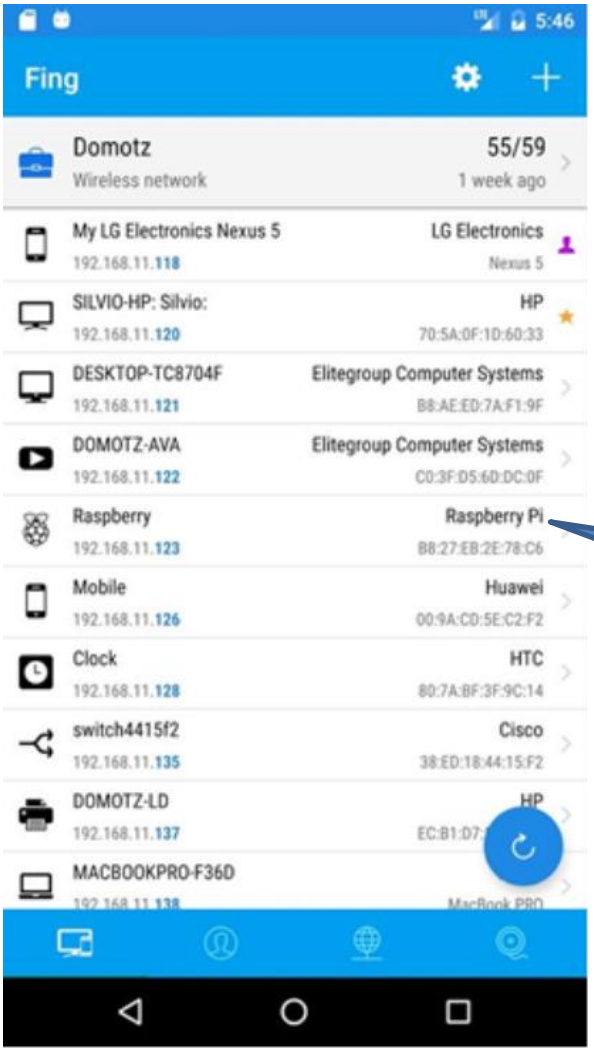
What version of the OS works with my Raspberry Pi model

The following article contains a table showing which model Raspberry Pi's work with which Raspberry Pi OS:
https://en.wikipedia.org/wiki/Raspberry_Pi_OS

Only Bookworm and Bullseye OS have been fully tested with a from-scratch installation.

Useful installation tools

Finding the Raspberry Pi on a network using Fing



Raspberry Pi detected

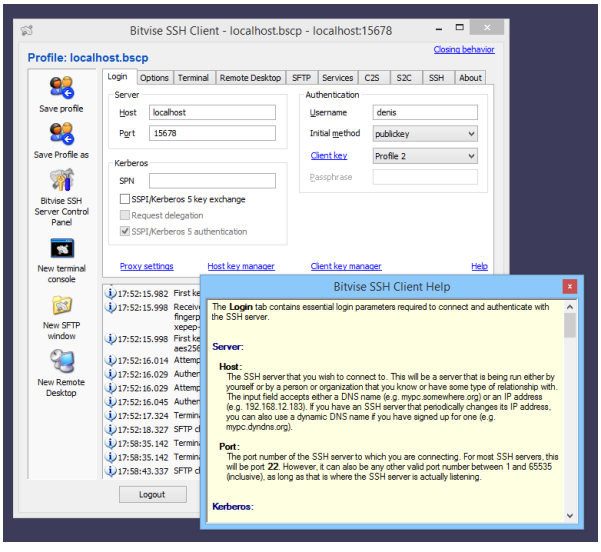
The **Fing** App is a free network toolkit and scanner for iOS (Apple) and Android. It will discover all the devices on your network, identify intruders and can also run Internet speed tests.

In this context it is very useful for finding out the IP address of any Raspberry Pi connected to the network. It will only see it of course once the Raspberry Pi has been set up first on the Wireless Network or hard-wired in. It scans both Wireless and hard-wired Ethernet devices.

Once detected it is the possible to the SSH to the Raspberry Pi using **Bitvise** or Putty.

See https://www.fing.com/products/fing-app for further information.

Bitvise and Putty



Bitvise is a free SSH client available for Windows or Mac to connect SSH server enabled Unix or Linux operating systems. It is a graphical based SSH client just like Putty with more features. It supports the File Transfer using SFTP Secure File Transfer Protocol). Once installed you can easily make a terminal connection to the Raspberry Pi using the IP address discovered by Fing above as well as easily transferring files.

See https://www.bitvise.com/ssh-client for more information.

Entering system commands

If you are new to Linux there are a couple of things that may cause confusion.

1. Entering program names on the command line
2. File path names
3. File permissions

Take the following examples:

```
$ raspi-config
```

The following command fails

```
$ configure_radio.sh
$ -bash: configure_radio.sh: command not found
```

The following two commands both work.

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

```
$ /usr/share/radio/configure_radio.sh
```

The third command has a **./** in front of it, the first one doesn't. Why?
The reason is that the **raspi-config** program is in the **/usr/bin** directory which is in the **PATH** environment directive. This can be seen with the following command.

```
$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/local/game
s:/usr/games
```

The second program is located in the **/usr/share/radio** directory which is not in the PATH directive. The **./** in front of the command means that the program or script will be found the current directory.

So, this means for programs not in directories specified in the **PATH** environment directive, you must either specify the full path name to the command or change to its directory and then enter the command with a **./** in front of it.

In the system prompt for user pi you will see a **~** character. The **~** character means the home directory for the current user, this case pi. So **~** is the same as **/home/pi**.

```
pi@raspberry3:~ $
```

For information on file permissions see the following link:
https://wiki.archlinux.org/index.php/File_permissions_and_attributes

Editing configuration files

At various points during the installation procedures in this manual you will be asked to edit certain configuration files such as **/etc/radiod.conf** (The radio configuration file) or **/boot/firmware/config.txt** (The boot configuration file). There are various text editors that can be used but the main ones in the case of the Raspberry Pi are:

1. Nano - **nano** is a small, free and friendly editor particularly suited for use by beginners.
2. Vi – **vi** is usually the professional user's choice of editor. It is very powerful but a lot harder to use for someone unfamiliar with it.

Usage:

nano <filename>
or
vi <filename>
Where *<filename>* is the name of the file to be edited.

See https://www.raspberrypi.org/documentation/linux/usage/text-editors.md for an overview of available text editors.

It is important to know that most configuration files are owned by root so you may be able to read them but not write them. For example:

```
$ nano /etc/radiod.conf
```

This will allow you to read the file but not change it. To give user **pi** temporary root user permissions so that you can save changes to the file use the **sudo** command in front of the editor command:

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

Or

```
$ sudo vi /etc/radiod.conf
```

Note: Make sure that you edit the file **/etc/radiod.conf** and not **/usr/share/radio/radiod.conf** which is the distribution file. The latter is copied to the **/etc** directory during installation and configuration.

Using the Vi editor

This is too big a subject to cover here. Type "Using vi" into a search engine such as Google or Bing to display various tutorials on how to use **vi**.

Using Nano

When **nano** is started it will display the contents of the file being edited. For example, **/etc/radiod.conf**. In this example the following screen will be displayed.

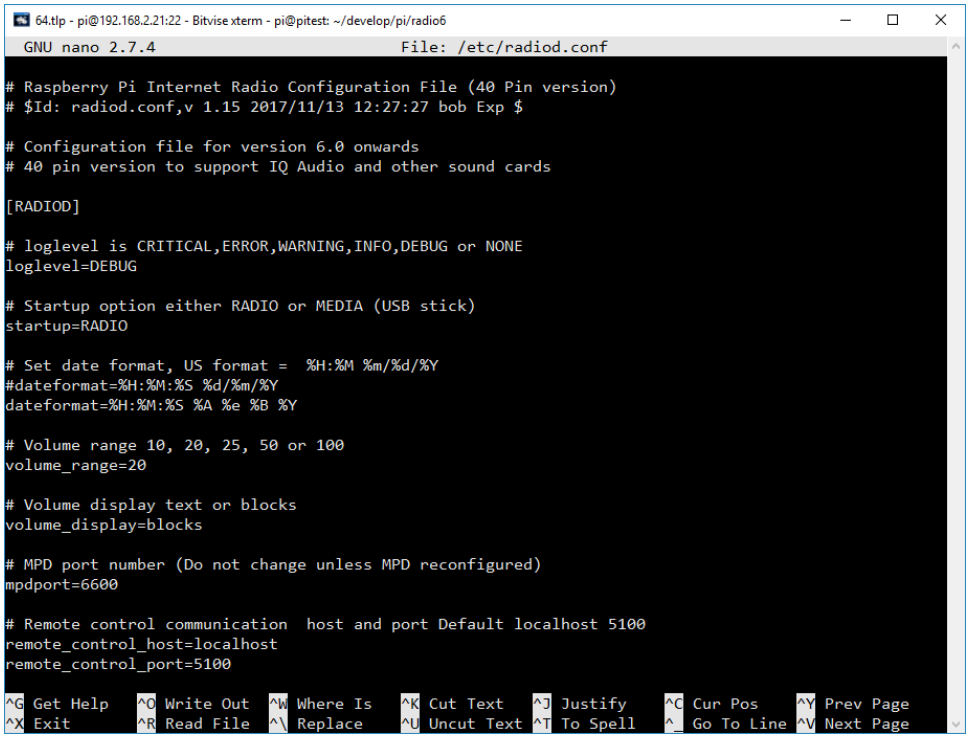


Figure 140 The nano file editor

Hold down the Ctrl key and press the letter G on the keyboard to display the help text. The following screen will be displayed:

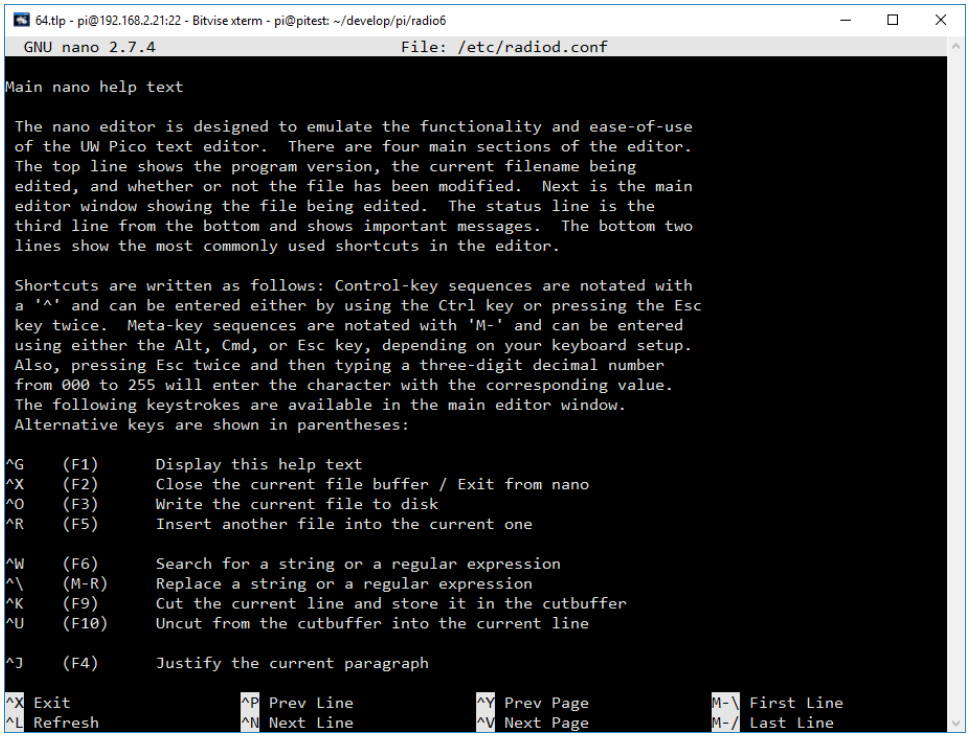


Figure 141 The nano editor help screen

The ^ character means the Control-key (Ctrl). So, for example ^O above is Ctrl + O. For more information on **nano** see https://www.nano-editor.org/dist/v2.0/nano.html

System Software installation

Raspberry Pi OS (previously called **Raspbian**) is the Foundation's official supported operating system. The latest version of **Raspberry Pi OS** is called **Bookworm**. Create a new SD card with **Bookworm** or **Bookworm Lite**. There is also a "Full" version of **Bookworm** however this is unnecessary for this project.

A lot of very useful Raspberry Pi documentation will be found at:
https://www.raspberrypi.org/documentation

Note: The touch-screen or HDMI TV version of the software requires a desktop version of the operating system so use **Raspberry Pi Bookworm** and not the **Lite** version. Only use **Lite** for LCD versions of the radio.

Warning: Version 8.0 is designed to work on primarily on **Raspberry Pi Bookworm** but can run with restrictions on **Bullseye**. **Bookworm** uses Music Player Daemon (MPD) version 0.23.12.

SD card creation

Use at least a 16 Gigabyte Card (Class 10) for **Bookworm Lite** or 32 Gigabyte for **Bookworm Desktop/Full**. Create an SD card running the latest version of **Raspberry Pi Bookworm** or **Bookworm Lite**. A Class 10 SD card is a memory card that can transfer data at a minimum rate of 10 Megabytes per second (MB/s).

Note: This version now works on either **32-bit** or **64-bit** architecture of the Raspberry Pi Operating System. However, it is important to choose the correct archictecture for your Raspberry Pi.

The following Raspberry Pi models have a 64-bit architecture and run Linux:

- Raspberry Pi 4
- Raspberry Pi 400
- Raspberry Pi 3B
- Raspberry Pi 3B+
- Raspberry Pi 3A+
- Raspberry Pi 5
- Raspberry Pi Zero 2 W

The Raspberry Pi 4 and Raspberry **Pi 400** models will appear as **32-bit** when using the default Raspberry Pi OS, which is 32-bit. However, a 64-bit version of the operating system is available as an option when using **Raspberry Pi Imager**.

The Raspberry Pi 1, 2, and Zero models are not 64-bit because their processors only have a **32-bit** architecture width. Bookworm or Bullseye 32-bit works on all Raspberry Pi models.

There are a couple of ways of creating the SD card but in this tutorial, we are using the **Raspberry Pi Imager** software to create the SD card. You will need a Windows PC or Laptop with a SD Card Reader as shown in Figure 142 below.

Figure 142 Windows Laptop with SD card reader

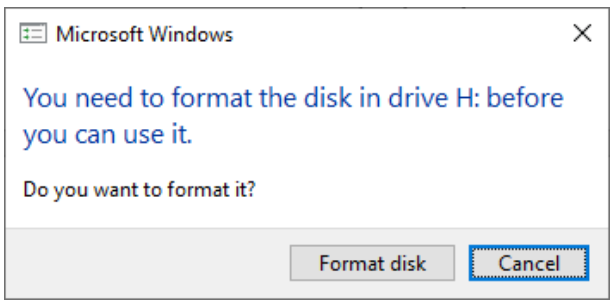
Figure 143 USB SD Card reader

If your PC does not have an SD Card Reader then you can use a USB SD Card reader as shown on the left. The Raspberry Pi Imager software can then be used to write the Raspberry Pi OS to SD Card.

Use a **Class 10** SD card such as **San Disk Ultra**. See

on page 232 for further information.

An Apple Mac PC can also be used but will require that you download the Mac OS version of the software. First insert the SD card into the SD card reader on your PC.
When you insert an SD Card that already has an OS other than Windows you may see the following:



Ignore this message and close the above dialogue box. The above may occur a number of times during this process.

There are two architectures supported in this version of the radio software:

- **32-Bit** for either Bookworm or Bullseye Operating Systems
- **64-Bit** for Bookworm only (1)

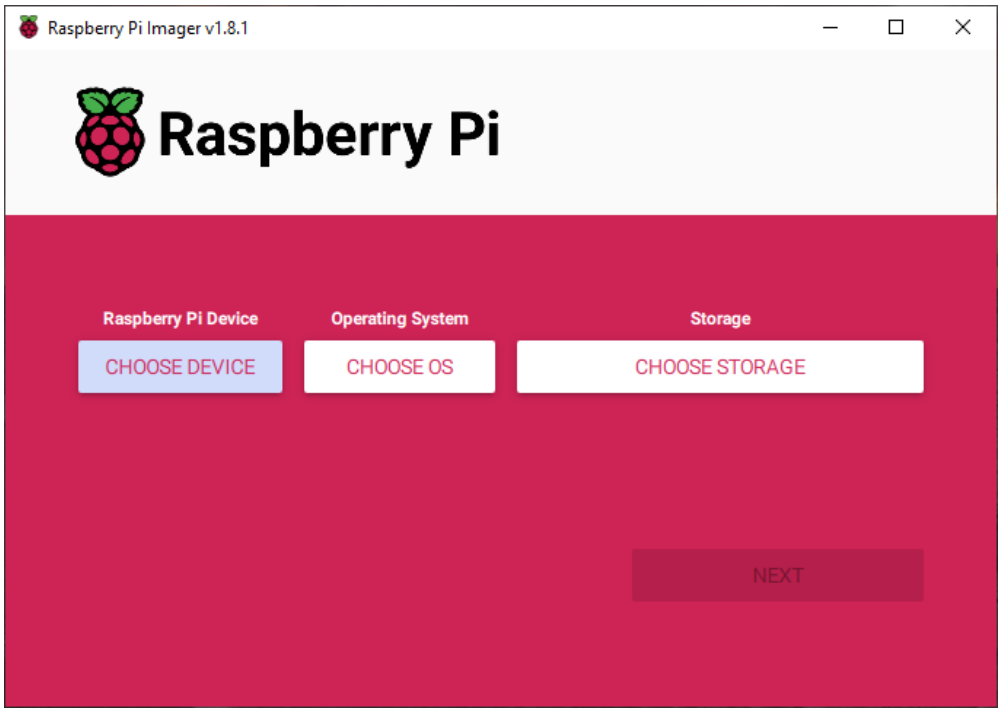
Note 1: You should now use the **64-bit** software as it is noticeably faster than the 32-bit OS. Use **Bookworm** for **64-bit** systems. **Bullseye** should not be used for fresh installations. Use Bookworm 32 or 64-bit Operating Systems for fresh installations.

Note 2: It is possible to boot from a USB 3.0 disk drive or stick instead of from an SD card. This is described TO BE DONE

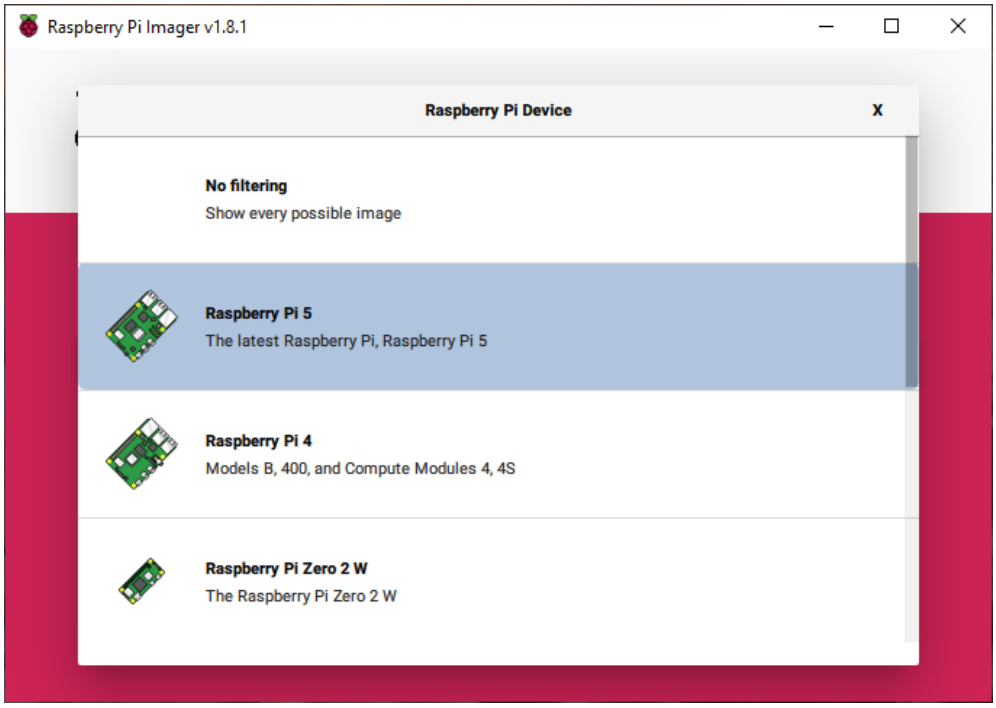
Using a Web browser, go to https://www.raspberrypi.org/software/ Download and install the **Raspberry Pi Imager** software for your PC Operating System (Normally Microsoft Windows or Mac OS). Once installed you will see the Raspberry Pi Imager Icon on your desk-top. Click on the desktop Imager icon.

The latest version of the software now re-directs all **RPi.GPIO** calls to **lgpio** in the **/usr/share/radio/RPi** directory when running on **Bookworm OS.** This mechanism is enabled by the Radio Configuration installation program.

Installing the OS using Raspberry Pi Imager



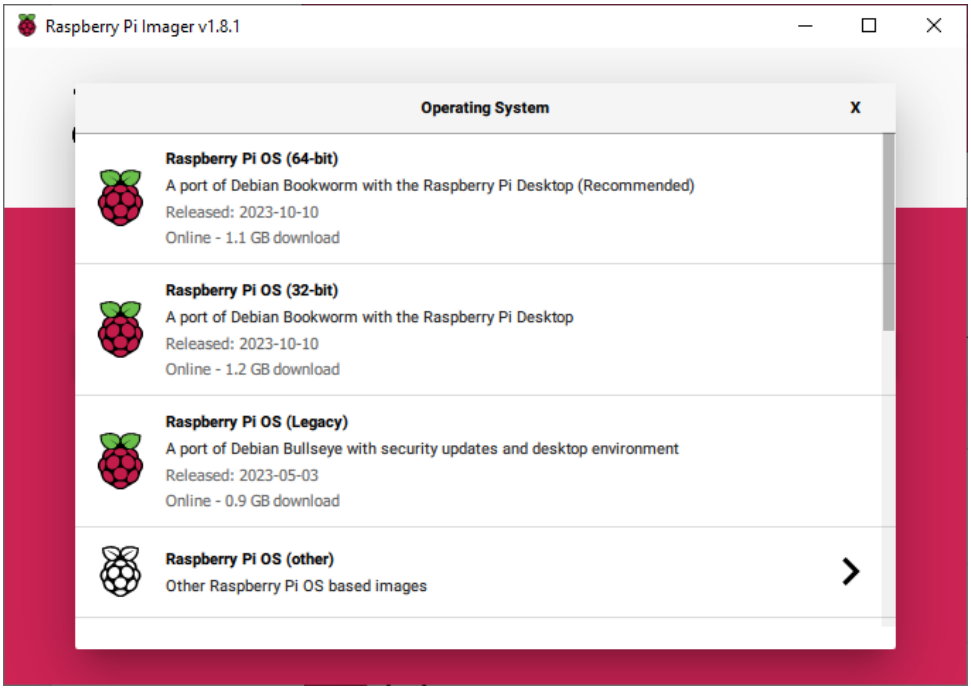
Insert the SD card into your card reader. Ignore any Windows messages to format the card as it isn't necessary. Click on "Choose Device"



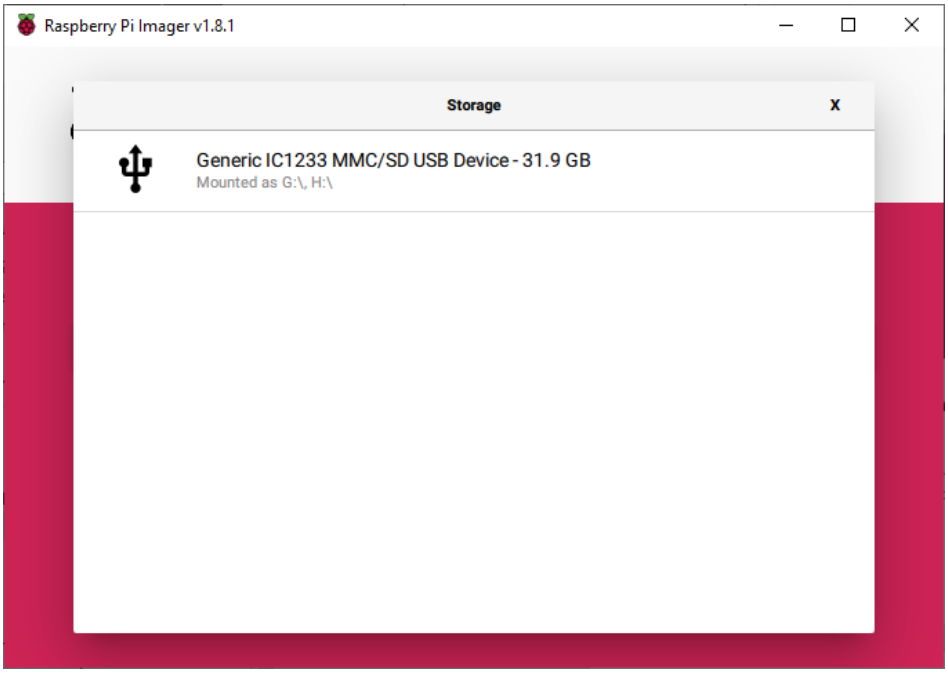
Select your device from the dropdown list. The program will then show the correct OS choices for your device.

Creating the SD card

The OS versions for your RPi will be displayed. For **Bullseye** select the third entry **Raspberry Pi OS (Legacy).**



The program will return to the first screen. Select CHOOSE STORAGE. This will display the available USB devices. The actual drive letter shown will have been assigned by your PC. You should see something like the following. In this example it is drive **H:**
Select your SD card. The imager software should only display USB devices.

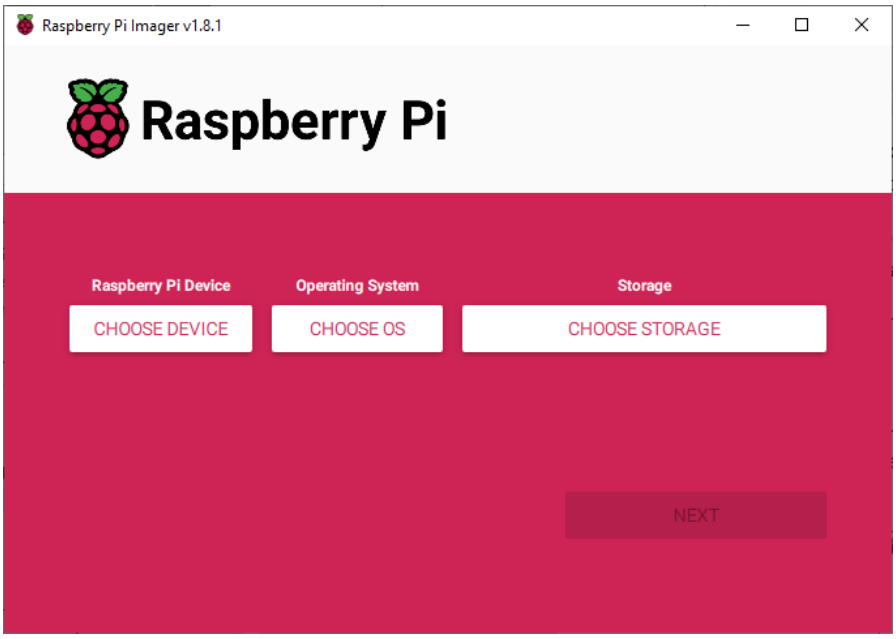


Note: If there are multiple choices of USB drives make sure that you select the correct one. Do not proceed until you are absolutely sure you are selecting the correct drive and not for example a USB backup drive.

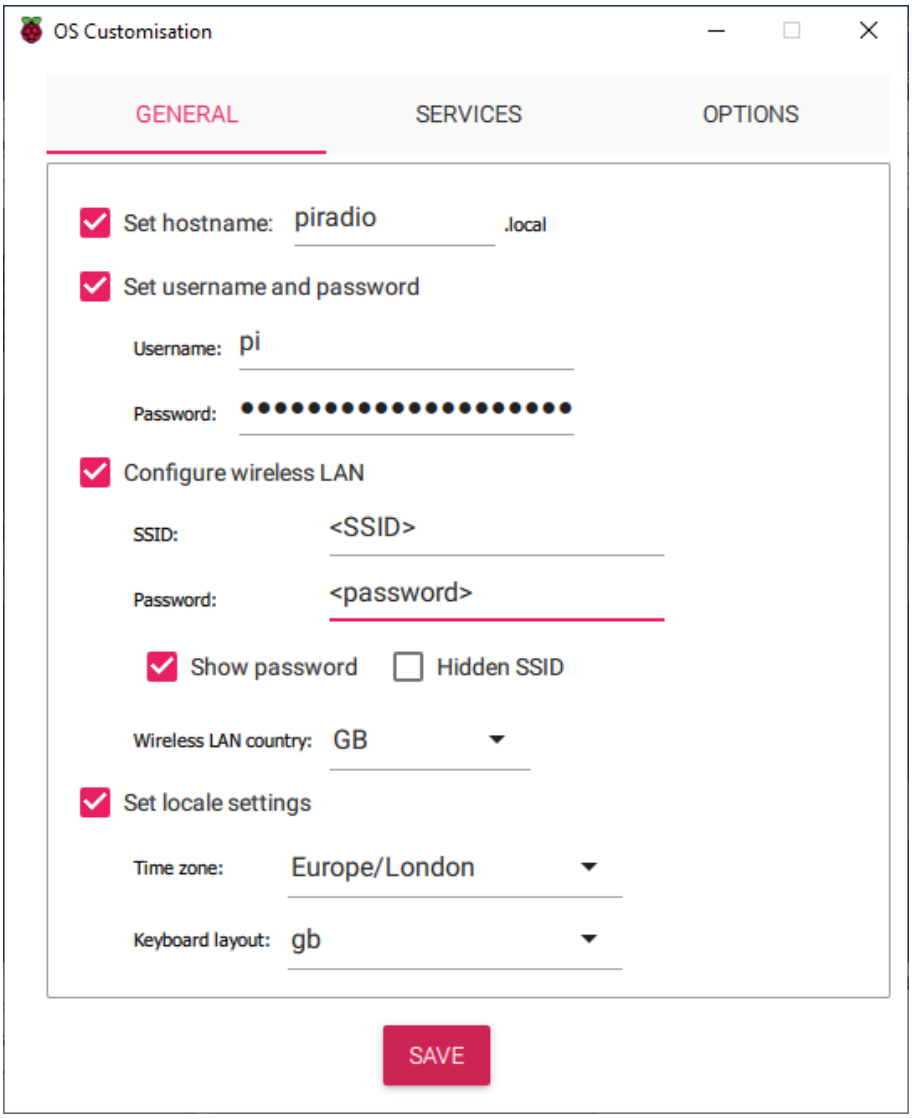
Note: Do **not** press write at this stage as you need to set up SSH (Secure Shell login) and your Wi-Fi identifier (SSID) and password and other optional changes.

Note: The default user name is **pi**. It is possible to use another username other than **pi** when creating the SD card. However, only limited testing has been carried out.

Once both the OS and USB drive have been selected the following screen is displayed:



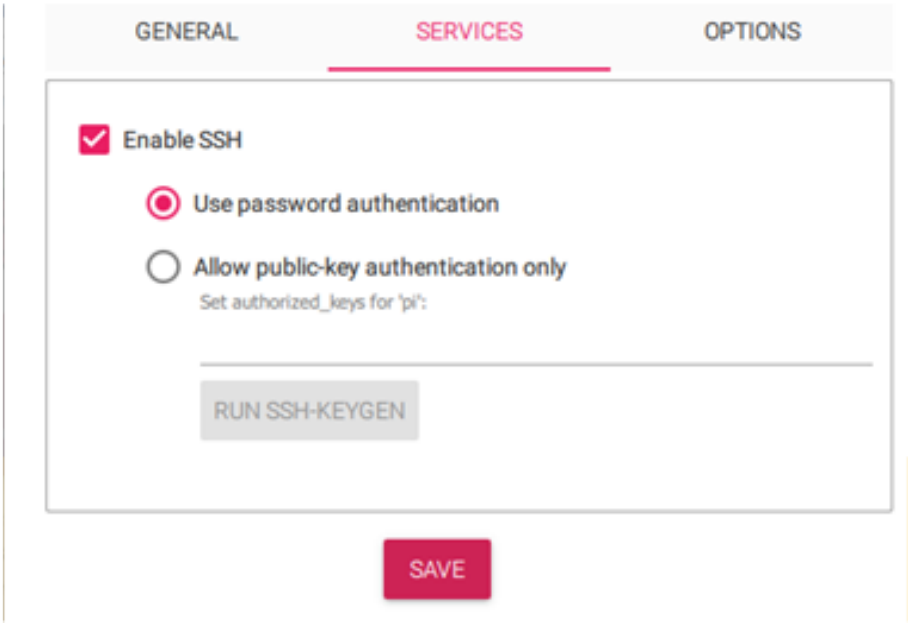
To select the customisation menu, press either the configuration icon (Gear wheel) in the bottom right or **CTRL**, **Shift** and **X** keys together. Clicking on WRITE will ask if you want to edit the parameters. The following Advanced Options screen will be displayed:



Initially the screen will show the GENERAL tab. Click the “Set hostname” and “Enable SSH” tick boxes. Enter a hostname (such as **piradio**) and a password for user **pi**. You can change the username from **pi** but it is recommended to use user **pi**. Make a note of the username and password.

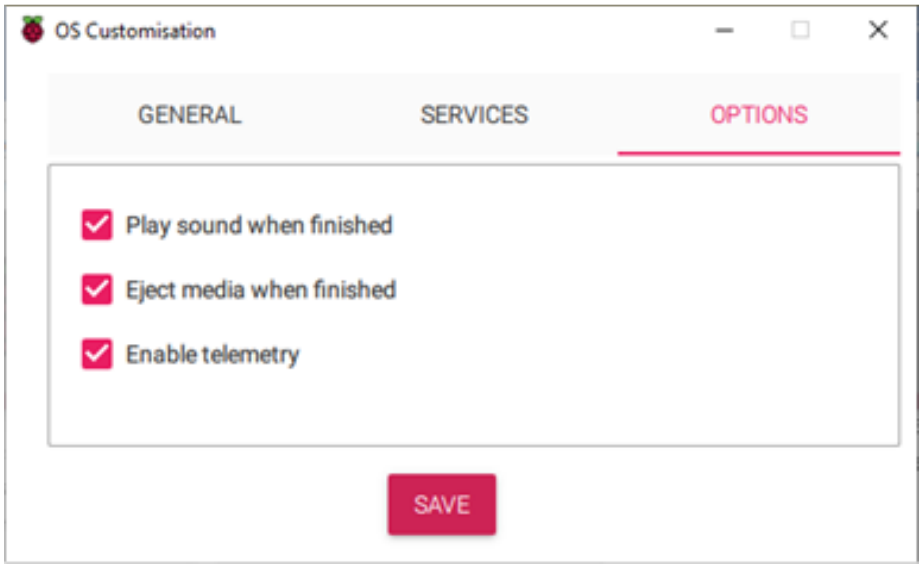
You must also set the locale using the two-letter country code for your location, for example GB (Great Britain), DE (Germany) or NL(Netherlands). It is required for correct set-up of Wi-Fi.

Next click on the SERVICES tab. “Enable SSH” should be ticked and “Use password authentication” should be selected.

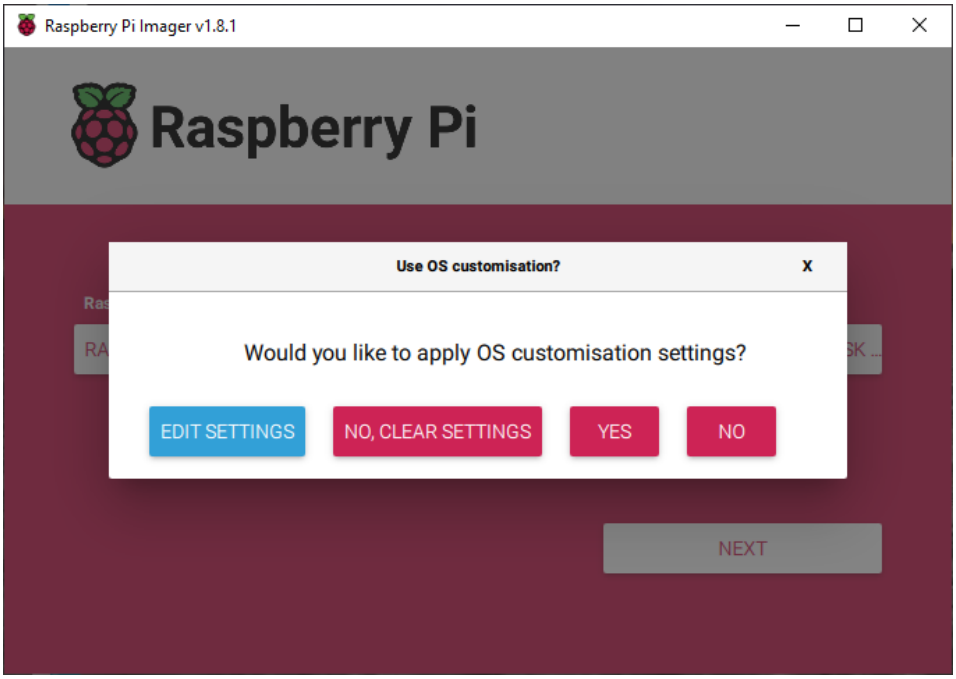


Important: Enable the SSH option “Use password authentication”. If you select “Allow public-key authentication only” you will have to set up SSH keys between the PC and the RPi which is more complicated and not covered yet in this manual.

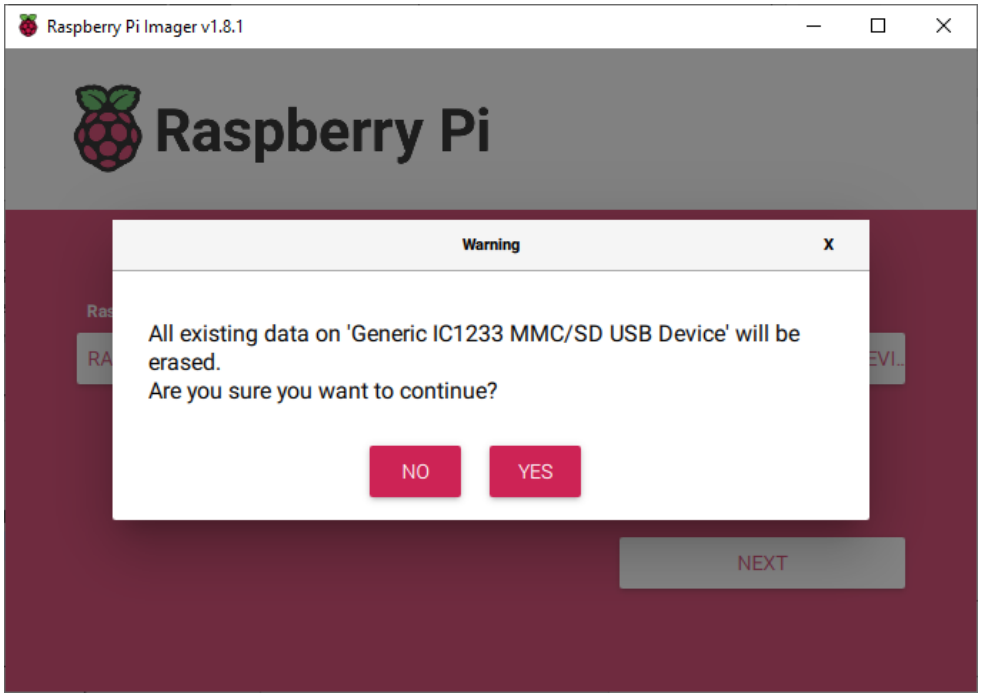
Finally select the OPTIONS tab:



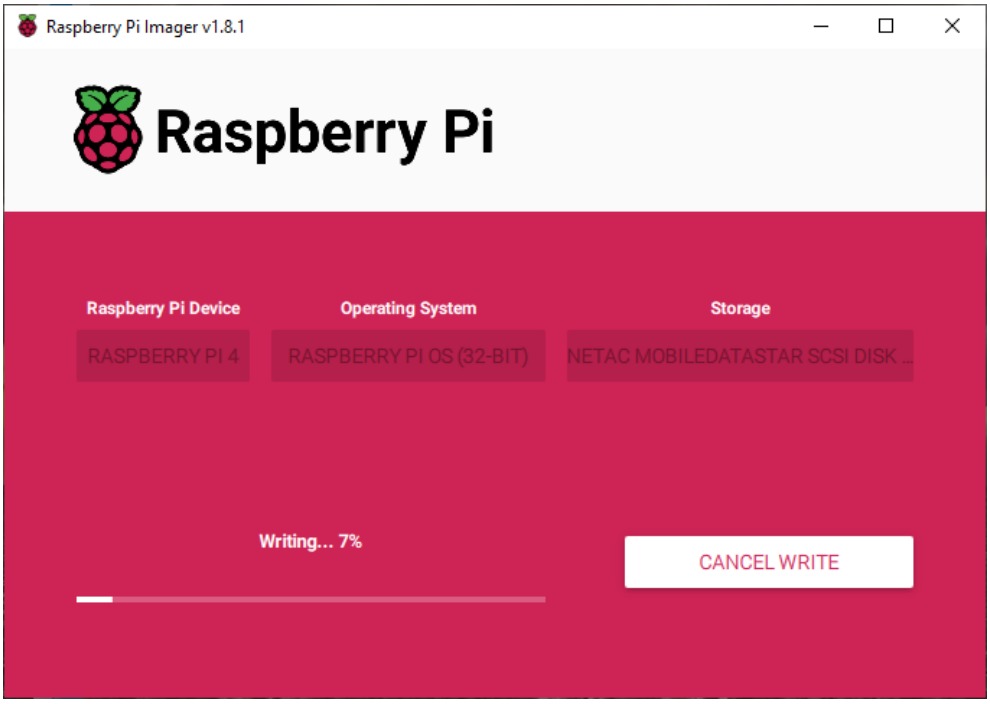
Click on “Save” followed by “Write” to continue.



Click “YES” to continue with writing to the new SD card.

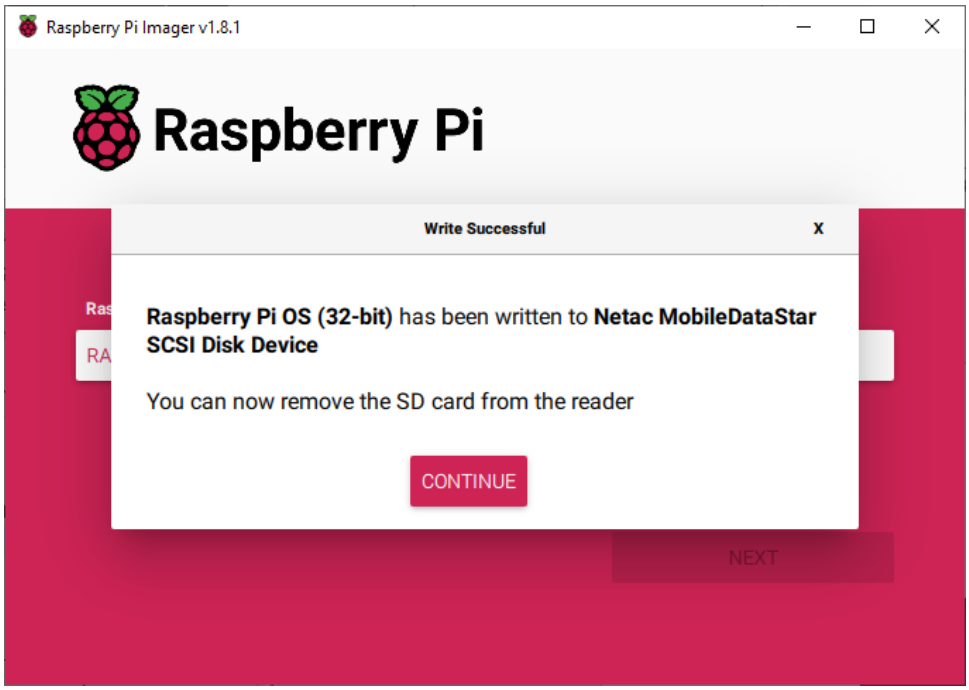


This is the point of no return. Pressing “YES” will continue to overwrite all existing on your USB device and replace it with the Operating System you selected.



This will take about 5 minutes to complete including the verification stage.

When the write operation is complete you will see the following:



Press Remove the SD card from your computer and insert it into the Raspberry Pi (Remove power from the Raspberry Pi first). Press CONTINUE. The program will return to the first screen. Close the program by clicking X in the top right-hand corner.

Booting the Raspberry Pi for the first time

Power up the Raspberry Pi. The first boot will take a little longer as there are a few jobs that the OS must carry out, such as re-sizing the file system on the SD card.

Note: If you are using a USB SSD device with say 250GB, the Raspberry Pi will take a few minutes to resize the file system to the capacity of the disk. Be patient whilst it is doing this.

Log into the Raspberry Pi for the first time

There are two ways of logging into the Raspberry Pi:

1. Using the graphical Linux Windows desktop
2. Using SSH to log into the RPi over the network

Logging in with the Raspberry Pi desktop

If you installed the OS with the Linux desktop you will need an HDMI monitor, a USB keyboard and mouse. Boot up the Raspberry Pi with the new SD card with the **Raspberry Pi Bookworm** Operating System (OS). If you have used **Bullseye** or **Bookworm** desktop then a graphical desktop will be displayed. Start a terminal session by clicking on the black terminal icon >_ on the top left of the screen to the left of the "Welcome" message. With **Raspberry Pi Lite** only a log in prompt will be displayed. In such a case log into the Raspberry Pi as user **pi** and using the password **raspberry**. Alternatively log in using SSH (See following section).

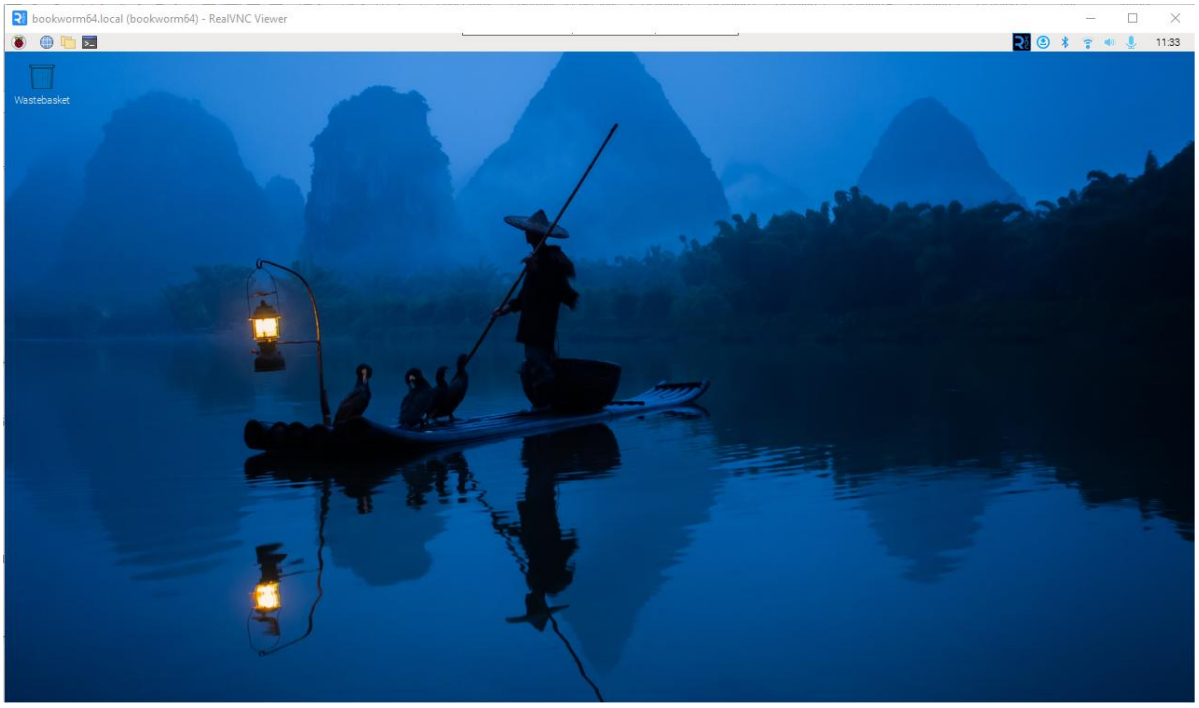


Figure 144 Raspberry Pi Bookworm desktop

If you created the SD card with the Lite version of the OS then you will not see a desktop. Instead, you will see a terminal session which will allow you to carry out the next steps of the installation.

Logging in with SSH (Bitvise)

If you don't have a USB keyboard, mouse and screen then you can log into the RPi using the Bitvise or Putty utility running on a Personal Computer. Run the **Bitvise** interface and using **Fing** to find the IP address of the Raspberry Pi as shown in the section *Finding the Raspberry Pi on a network using Fing* on page 80. Enter the IP address from Fing into the Host field (192.168.1.36 in this example). Also enter the user's name **pi** and the password you set up.

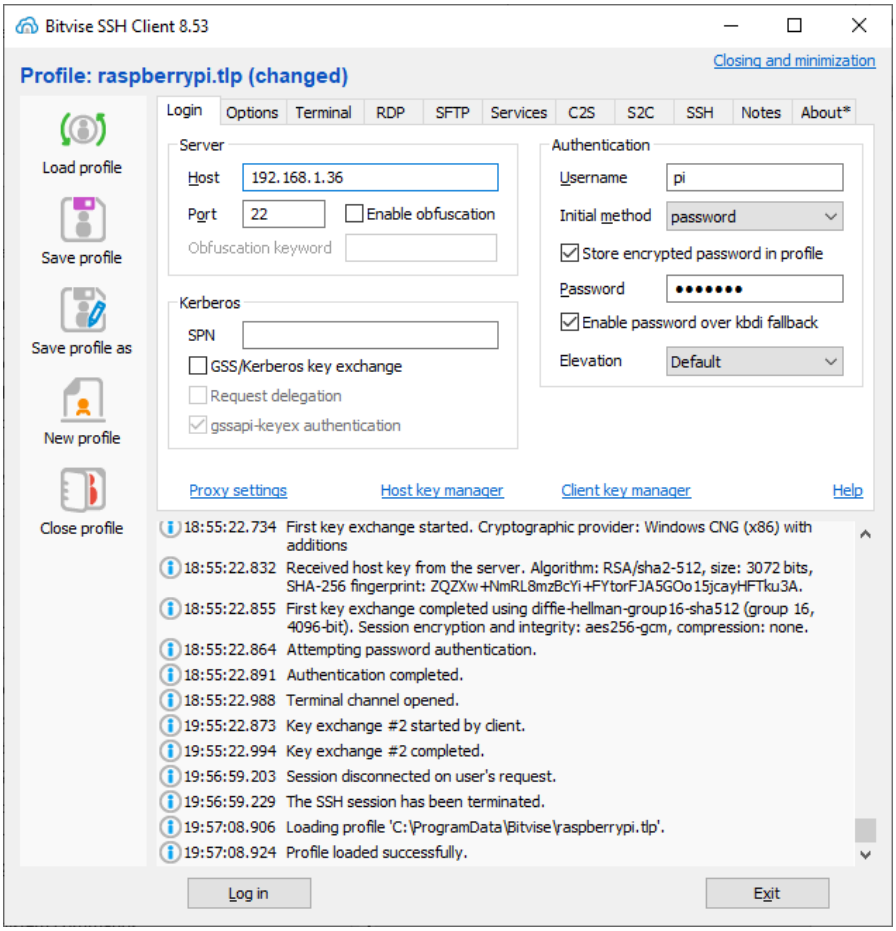


Figure 145 Bitvise connection dialogue

Note: You can also use the host name of the Raspberry Pi instead of its IP address. If you set the SD card up according to the instructions shown earlier. For example, if 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.

Enabling SSH using raspi-config

If SSH was enabled when creating the SD card this step should not be necessary and you can skip it. You will need a USB keyboard, mouse and HDMI display screen. Once logged in run **raspi-config**.

```
$ sudo raspi-config
```

Select option 3 – Interface options. The following screen will be displayed

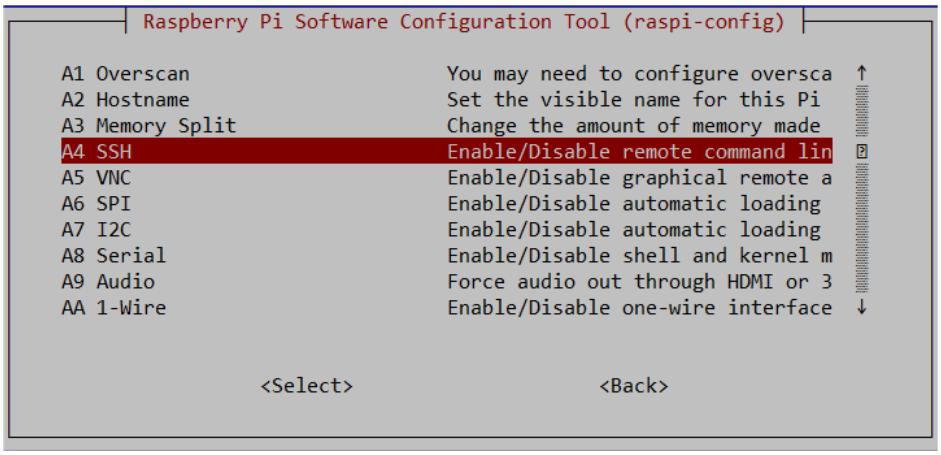


Figure 146 Enabling SSH

Reboot the Raspberry Pi after which it will be possible to log into the Raspberry Pi using SSH.

```
$ sudo reboot
```

Now log into the Raspberry Pi using SSH.

```
$ sudo reboot
```

After logging back in the following message may be displayed if a new password wasn't set.

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

Note: Security is becoming more and more of an issue for devices connected to the internet. If SSH has been enabled then please change the user password at the first opportunity. See the section called **Error! Reference source not found.** page **Error! Bookmark not defined.** for further information on security issues.

Preparing the Operating System for software installation

Update to the latest the system packages

Run the following command to update the library list.

```
$ sudo apt update
```

The above command will take some time! If you see a message similar to the following.

```
E: Repository 'http://raspbian.raspberrypi.org/raspbian Bookworm InRelease'
changed its 'Suite' value from 'testing' to 'stable'
```

Run the following command

```
$ sudo apt update --allow-releaseinfo-change
```

Now re-run the update.

```
$ sudo apt update
```

Run the following command to upgrade to the latest packages for this release.

```
$ sudo apt upgrade
```

IMPORTANT: Do not, under any circumstances, run the apt f**ull-upgrade** or **rpi-update** command especially if installing on **Bookworm** as the kernel in these updates isn't stable yet. Older Raspbian versions such as Stretch required a separate command called **rpi-update** to update the firmware. Do not use **rpi-update** with **Bookworm** or **Bullseye** unless specifically advised to do so by an official source.

Reboot the Raspberry Pi.

```
$ sudo reboot
```

Important: After upgrading the system the repository locations may no longer be valid. Re-run **apt update** to refresh the package list. Failing to do this may result in packages failing to install.

Re-run the update command to update the library list.

```
$ sudo apt update
```

Once you have finished updating the operating system login to the system and run **raspi-config**.

```
$ sudo raspi-config
```

Warning: If you are intending to run the touch-screen/HDMI version of the radio, do not be tempted to start removing components of the **pygame** software such as games as this may unfortunately remove graphic libraries used by the radio software.

Configuring the Operating System

Disable booting to the desktop environment

If you are planning to use a touch-screen or HDMI display skip this section.
The desktop environment is not required for the LCD, TFT or OLED versions of the Radio and takes a lot of processing power. It is enabled by default in **Bookworm** but is not installed with **Bookworm Lite**. If you are not planning to use the touch-screen or HDMI version of the radio or you don't otherwise plan to use it then disable it.

Run raspi-config:

```
$ sudo raspi-config
```

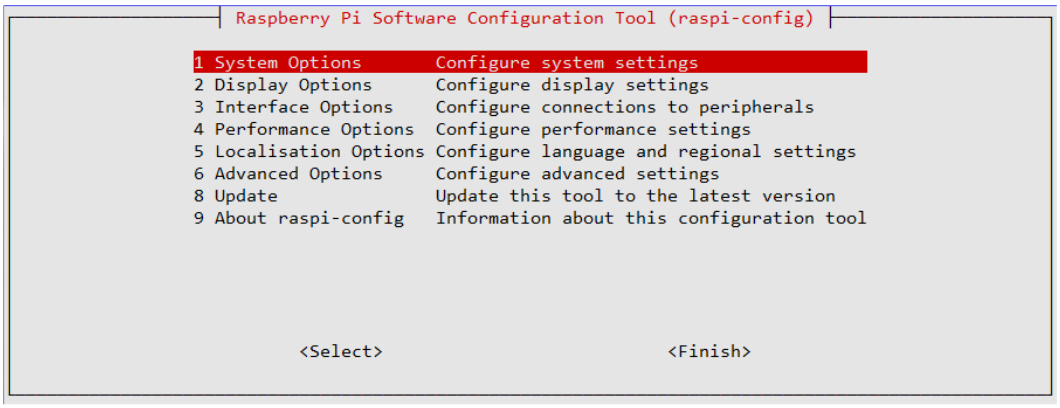


Figure 147 raspi-config main screen

Select option **1 System options**

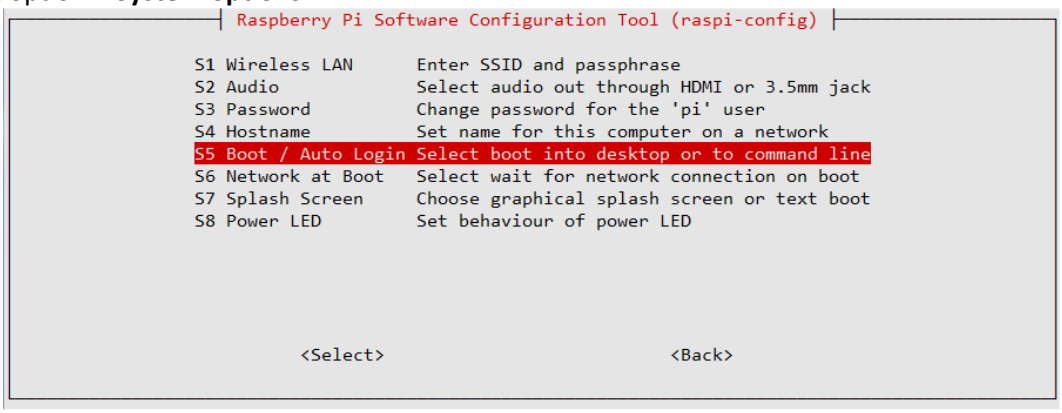


Figure 148 Disabling the graphical desktop

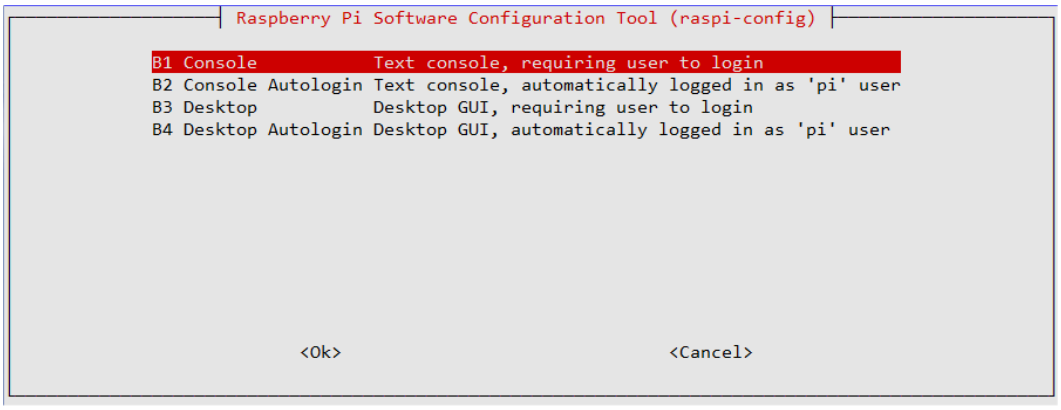


Figure 149 Desktop enable/disable selection

Select option **B1**(secure) or **B2**(insecure) to disable the desktop and select OK

Setting the time zone

The **Raspberry Pi OS** operating system is usually set to UK time or the time zone that you set with the **Raspberry Pi OS imager** settings. If you still need to set the time zone then the easiest way to set the time zone for your country if you are in a different time zone is to use the **raspi-config** program and select option 5 “Localisation Options”:

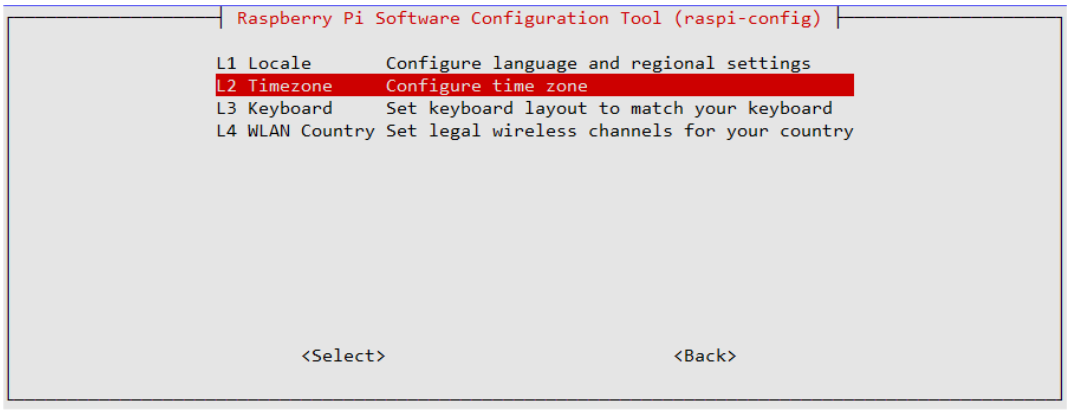


Figure 150 Setting the time zone

Select option L2 “Timezone”:

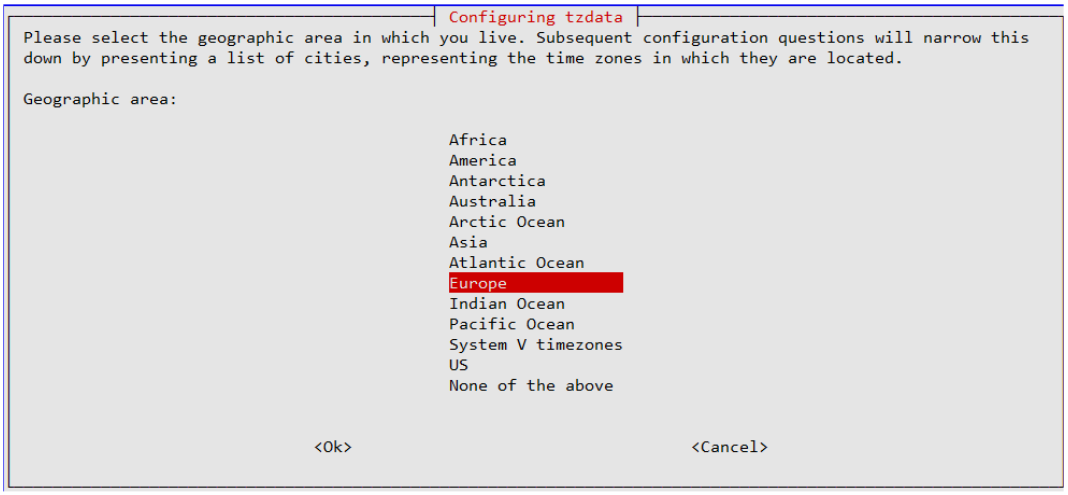


Figure 151 Saving the time zone

Select the region your country is in, Europe for example. Use the tab key to move to <OK> and press enter. The program will then display a list of time zones for the selected region.

Figure 152 Time zone country selection

Select the correct one and save it by tabbing to <ok> and pressing the enter key. The time zone will be updated. Exit the program once finished.

Changing the system hostname and password

If the **Raspberry Pi** Imager was used to create the SD card it may be that the hostname and password may have already been set, in which case this option can be skipped.

It is a good idea to change the system password for security reasons especially if your raspberry PI is connected to a network with a wireless (Wi-Fi) network. Changing the hostname is also a good idea as it makes identifying your radio on the network much easier. If you wish to do this, change the default hostname from 'raspberrypi' to something like 'piradio' or 'myradio'.

Both the password and hostname can be changed using the **raspi-config** program. Select option **1 System options**

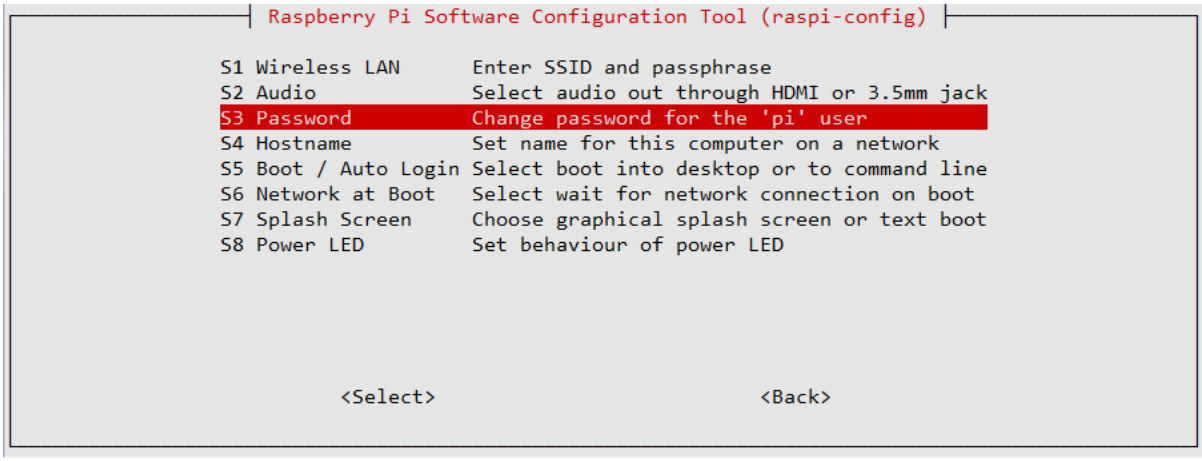


Figure 153 Changing the Raspberry PI password

Option **S3** is used to change the password. Make sure you record your new password somewhere safe (It is easy to forget it).

The hostname is changed in option S4 Hostname:

```
Enter new UNIX password:
Retype new UNIX password:
```

As the password is entered nothing is displayed. This is normal. You will be asked if you wish to reboot the system. After you reboot the system you will see the new hostname at the login prompt.

```
pi@piradio:~$
```

In the above example the new hostname is **piradio**. Once the hostname has been changed the program will ask if you wish to reboot. Answer "yes" to reboot.

Configuring the Wi-fi Connection via raspi-config

If the **RaspberryPi** Imager was used to create the SD card it may be that the Wi-Fi interface has already been configured, in which case this option can be skipped.

Select the country you are in for legal Wi-Fi channel selection. From the first **raspi-config** screen select **1 System Options** followed by option S1 Wireless LAN:

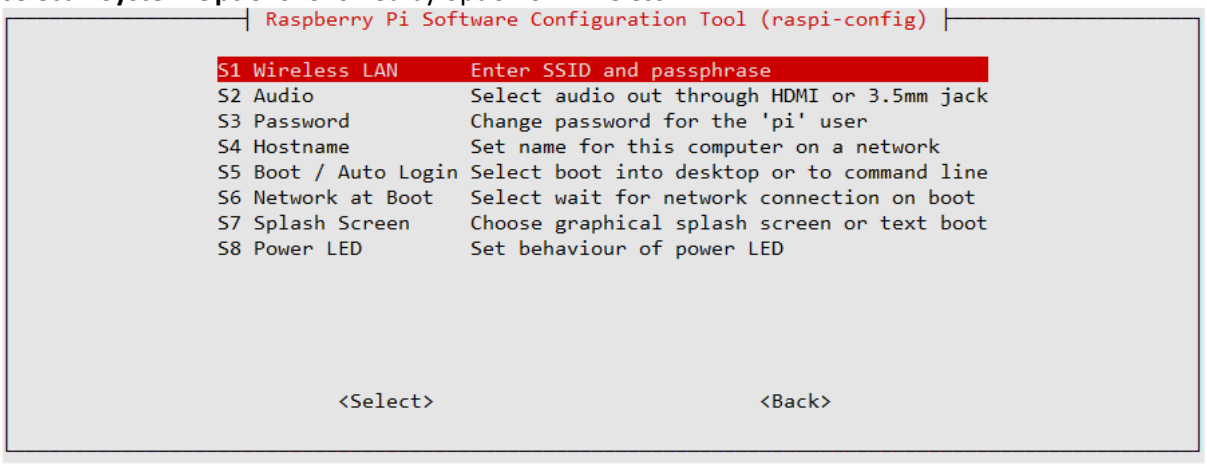


Figure 154 Setting up the Wi-Fi in raspi-config

Now enter the SSID and passphrase for your network. Save the settings and reboot the Raspberry PI. If this option is not available then use the procedure

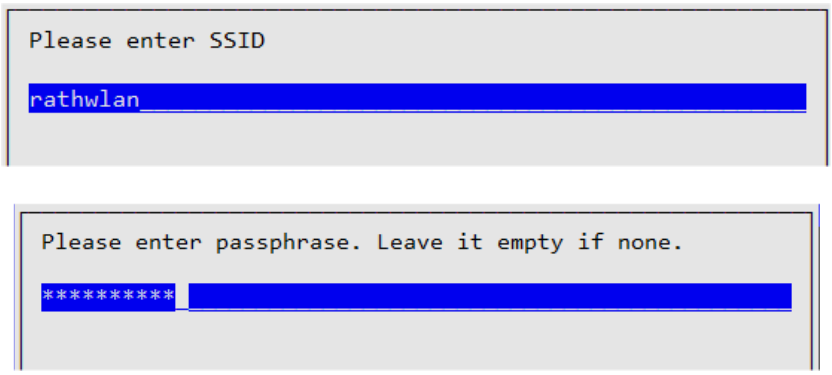


Figure 155 Entering Wi-Fi credentials

Reboot the system. After reboot it is possible that you may see the following message:

```
Wi-Fi is disabled because the country is not set.
Use raspi-config to set the country before use.
```

In such a case re-run **raspi-config**. Select localisation options and select I4 to set Wi-Fi country:

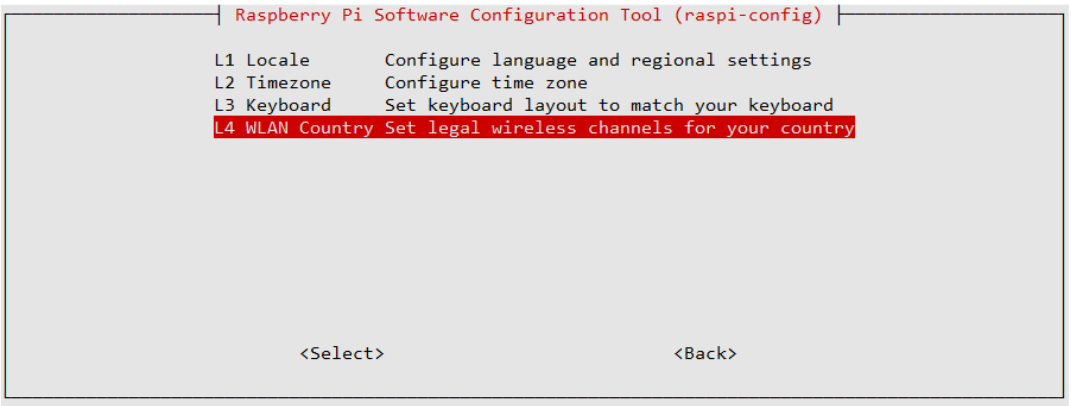


Figure 156 Setting up the Wi-fi country

Select your country from the dropdown menu and exit the **raspi-config** to save the setting.

Setting up the locale

As default the language used by Raspbian is English. To change this in Advanced options select option I1 Change Locale.

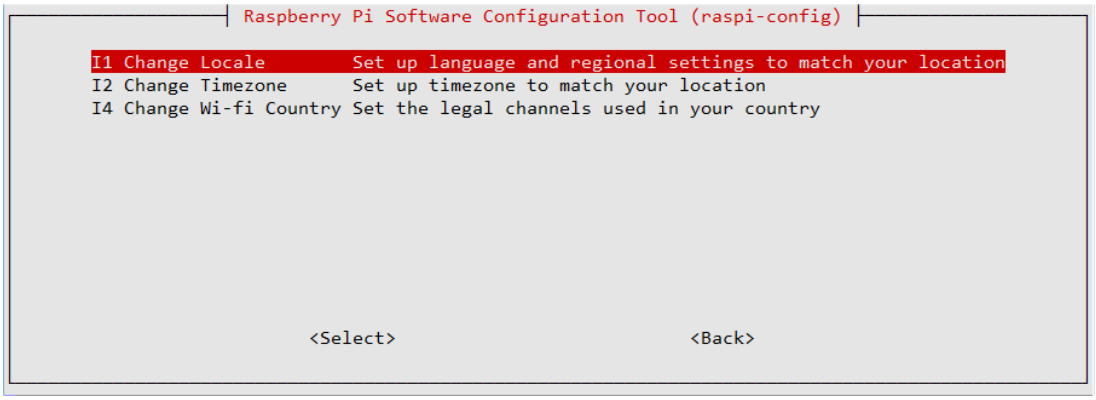


Figure 157 Selecting change locale

Select a locale beginning with the two-letter international code for your country and language.

Usually this will be a locale containing the string ISO or UTF-8.
fi_FI.ISO-8859-1 (Try this one first)
fi_FI.ISO-8859-15@euro
fi_FI.UTF-8

In the following example for the Netherlands the locale is this **nl_AW UTF-8** for the Netherlands (nl).

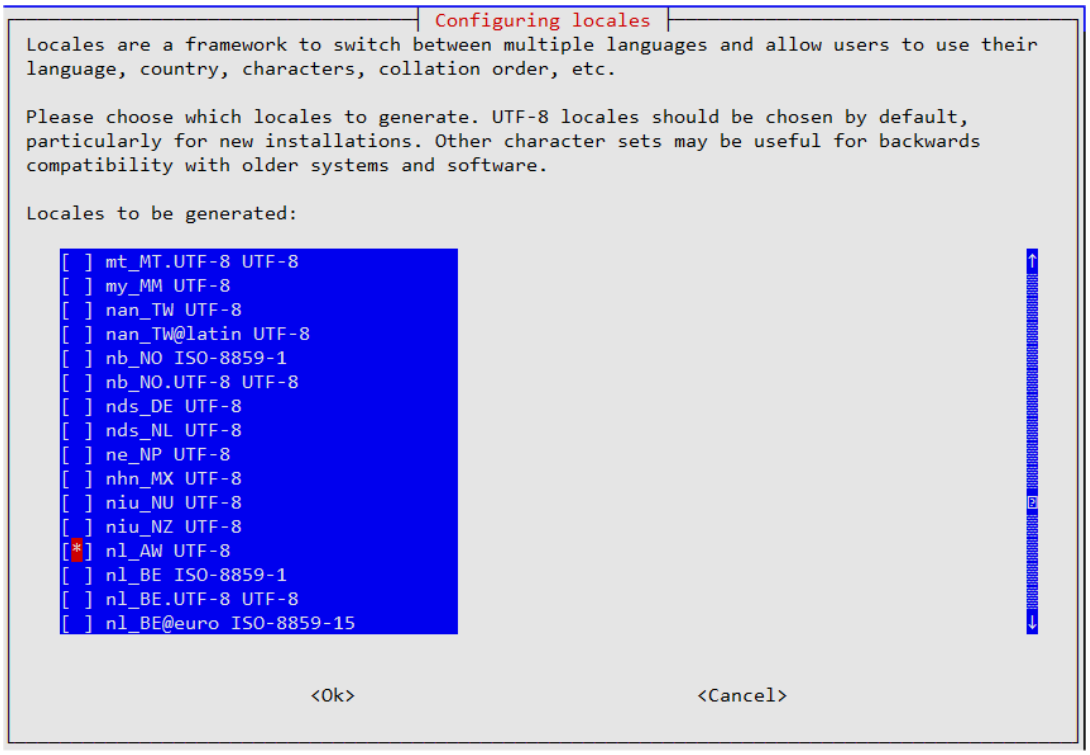


Figure 158 Generating the locale.

Warning: Do not be tempted to generate “All locales” as this is not only unnecessary and futile and will take a very long time.

Press OK to continue. The program will now generate the selected locales.

```
Generating locales (this might take a while)...
  en_GB.UTF-8... done
  nl_AW.UTF-8... done
Generation complete.
```

The following screen is displayed:

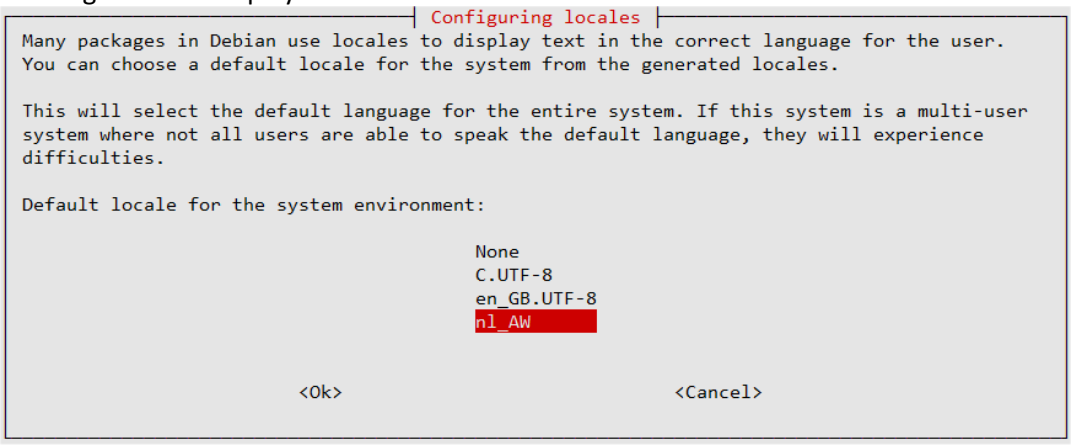


Figure 159 Selecting the locale

Select the required locale **nl AW** in this example and press OK to save. The new locale will become active after the next reboot.

Install other required packages

The table below shows additional packages which are required depending upon the design.

Table 14 Additional system packages

Package	Grove	HDMI/ Touchscreen	Olimex	Shoutcast	RGB I2C Rotary Encoder	Page
ffmpeg	No	Yes	n/a	n/a	No	102
libffi-dev **build-essential** **libi2c-dev** **i2c-tools** **python3-dev**	No	No	Yes	n/a	No	
python3-pil	Yes	No	Yes	n/a	No	103
scratch	No	Optional	No	No	No	103
rpd-wallpaper	No	yes	No	No	No	103
ioe-python	No	No	No	No	Yes	104

Install the ffmpeg video converter

This section is only applicable if the HDMI/Touchscreen version of the radio is to be used. If using an LCD version of the radio then skip this section. The **ffmpeg** video converter is required to extract artwork (if included) from music files mpeg files. Install **ffmpeg** video converter with the following

command (Note that it may already be installed but may not be in the case of the Lite version of the OS):

```
$ sudo apt install ffmpeg
```

```
Reading package lists... Done
:

The following packages have unmet dependencies:
 ffmpeg : Depends: libavdevice57 (>= 7:3.2.10) but it is not going to be
installed
          Depends: libsdl2-2.0-0 (>= 2.0.4) but it is not installable
E: Unable to correct problems, you have held broken packages.
```

If the installation fails with the above message re-run the **apt update** command to update the library list and retry installing **ffmpeg**.

```
$ sudo apt update
```

Run the following for further information about **ffmpeg**.

```
$ man ffmpeg
```

Install the anacron package

The **anacron** package is required to run the **/etc/cron.weekly/radiod** and **/etc/cron.daily/radiod** scripts. These scripts are required to run the **create_stations.py** and **/etc/cron.daily/radiod** playlist maintenance programs.

Install it with the following command.

```
$ sudo apt -y install anacron
```

Install the Scratch and rpd-wallpaper packages

This section is only applicable if the HDMI/Touchscreen version of the radio is to be used. There is an option to provide extra backgrounds. If using an LCD or OLED versions of the radio then skip this section. Scratch and rpd-wallpaper used to be installed by default, however, in the latest versions of **Raspberry Pi OS Desktop** they may not be. Extra backgrounds can be used by installing **Scratch**. Scratch is a visual programming tool for children.

Scratch is not actually used by the radio software but the graphic files in the **/usr/share/scratch/Media/Backgrounds** directory provide additional background wallpaper for the full feature graphical radio (gradiod.py).

The background wallpaper is set by the following parameter in **/etc/radiod.conf** and can be amended to use a different background. This can only be done after installing the radio software in the next section.

The default wallpapers are available in the **/usr/share/rpd-wallpaper** directory. Amend the wallpaper command once you have installed the radio software.

To install **rpd-wallpaper** run the following:

```
$ sudo apt install rpd-wallpaper
```

The default wallpaper is aurora.jpg

```
wallpaper=/usr/share/rpd-wallpaper/aurora.jpg
```

If **scratch** is installed other additional wallpapers can be used:

```
wallpaper=/usr/share/scratch/Media/Backgrounds/Nature/beach-malibu.jpg
```

To install scratch run the following:

```
$ sudo apt install scratch
```

Install Pimoroni ioe-python

The **ioe-python** python package is required for RGB I2C Rotary Encoders.

```
$ cd
$ git clone https://github.com/pimoroni/ioe-python
$ cd ioe-python/
$ sudo ./install.sh
```

The following message will be displayed.

```
This script should be run in a virtual Python environment.
Would you like us to create and/or use a default one? [y/N] y
```

Answer Y to the above question. The following will be displayed.

```
Creating a new virtual Python environment in /home/pi/.virtualenvs/pimoroni,
please wait...
Checking for /home/pi/Pimoroni/auto_venv.sh

Creating /home/pi/Pimoroni/auto_venv.sh
```

For both **Bookworm** and Bullseye **OS** you will see the following message.

```
Would you like to copy examples to /home/pi/Pimoroni/pimoroni-ioexpander?
[y/N] N
```

Unless you particularly want the example code answer N.

```
Would you like to generate documentation? [y/N] N
```

Again, answer no unless you particularly want the documentation.

It is now necessary to link the **ioexpande**r package installed in the virtual directory to the site If running **Bullseye OS** run the following command (Note: all one line):

```
$ sudo ln -s ~/.virtualenvs/pimoroni/lib/python3.9/site-packages/ioexpander
/usr/lib/python3.9/dist-packages/ioexpander
```

If running **Bookworm OS** run the following commands:

```
$ sudo ln -s ~/.virtualenvs/pimoroni/lib/python3.11/site-packages/ioexpander
/usr/lib/python3.11/dist-packages/ioexpander
```

Reboot the RPi and test the Rotary encoders.

```
$ sudo reboot
```

Extra packages required for Raspberry OS Lite

If you are running **Raspberry OS Lite then** not all required packages are installed. Run the following commands.

For **Bullseye**. Skip this step if running **Bookworm**.

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

For both **Bullseye** and **Bookworm** or later:

```
$ sudo apt install pythonwm8960
3-pip git
```

The following are required for any displays that are using **I2C**.

```
$ sudo apt install -y i2c-tools
```

Chapter 6 - Installing the radio Software

Installing the Radio Daemon

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
Either download it to your PC or Macintosh and copy it to the **/home/pi** directory or get it directly using the **curl** facility. To use the **curl** 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

```
$ curl -L -O http://bobrathbone.com/raspberrypi/packages/radiod_8.0_armhf.deb
```

Run **dpkg** to install the package.

```
$ sudo dpkg -i radiod_8.0_armhf.deb
```

For 64-bit systems:

```
$ curl -L -O http://bobrathbone.com/raspberrypi/packages/radiod_8.0_arm64.deb
```

Run **dpkg** to install the package.

```
$ sudo dpkg -i radiod_8.0_arm64.deb
```

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

Configuring the radio

Once the basic radio package installation has been completed the installation will automatically run the **configure_radio.sh** script.

This updates the configuration settings in the radio configuration file **/etc/radiod.conf**.

When run the installation script detects if this is a software upgrade and displays the following screen. Normally select option 2 if to install the software. Option 1 allows you to upgrade the software without doing any configuration. You can run the configurator at a later time.

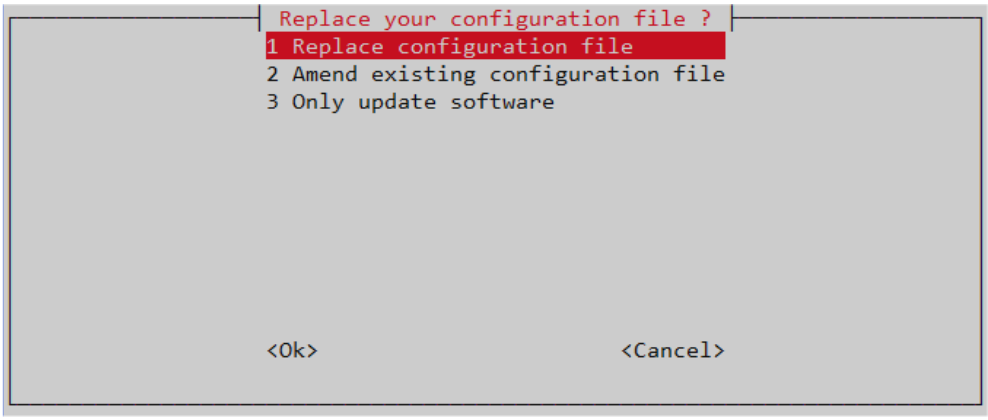


Figure 160 Configure radio

If you selected option **1 Replace configuration file** above and you are upgrading the software from a previous version, the program will ask if you wish to overwrite the existing configuration. Unless you have a heavily modified configuration you may safely overwrite the configuration file. You should select this option when installing the radio software the first time.

Option **2 Amend existing configuration file** is used in the case of re-installs and allows you to keep your existing configuration file. This is useful to keep any audio device settings that you have already made.

Option **3 Only update software** installs the software only. The existing configuration is left untouched.

Note: This configuration program can be re-run at any time in the future. Change directory to **/usr/share/radio** and run **configure_radio.sh**. To do this run the following:

If this is a new installation select option 1, the existing configuration file will be replaced. A backup copy of the original configuration is written to **/etc/radiod.conf.save**. If upgrading select option 2 to keep your existing configuration. The following screen will then be displayed:

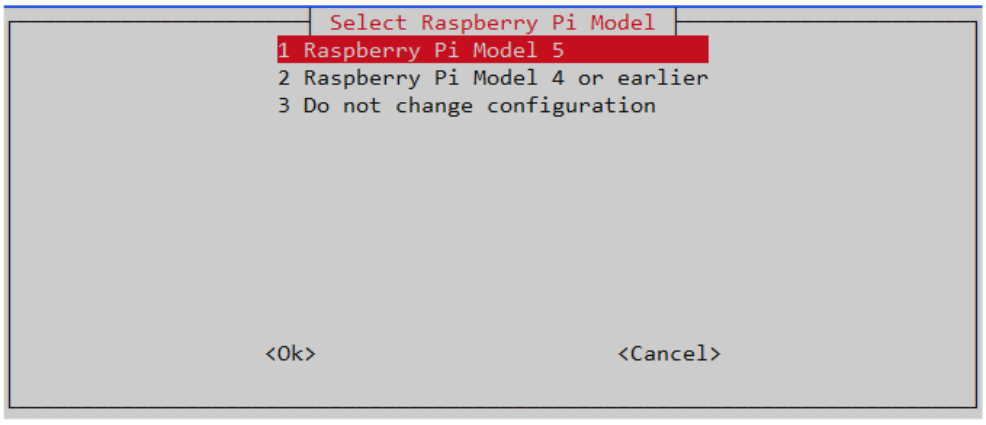


Figure 161 Select Rasberry Pi Model

Select your model of Raspberry Pi, either the Model 5 or the model 4 or earlier.

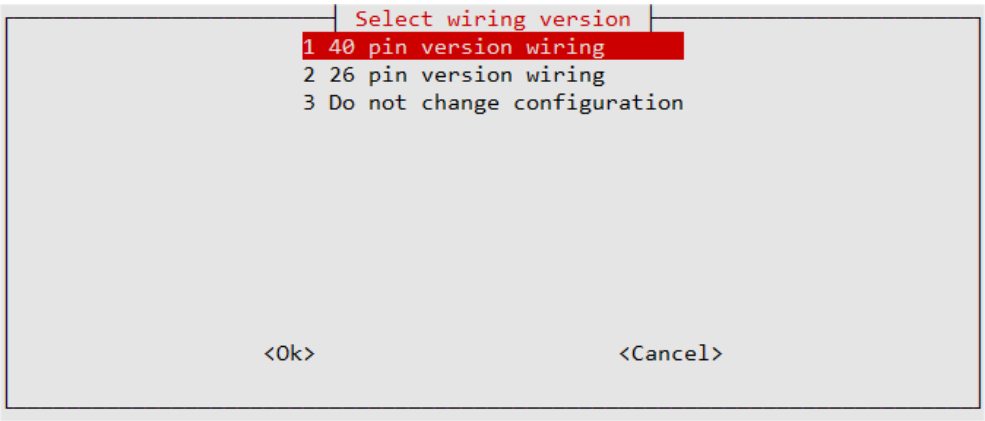
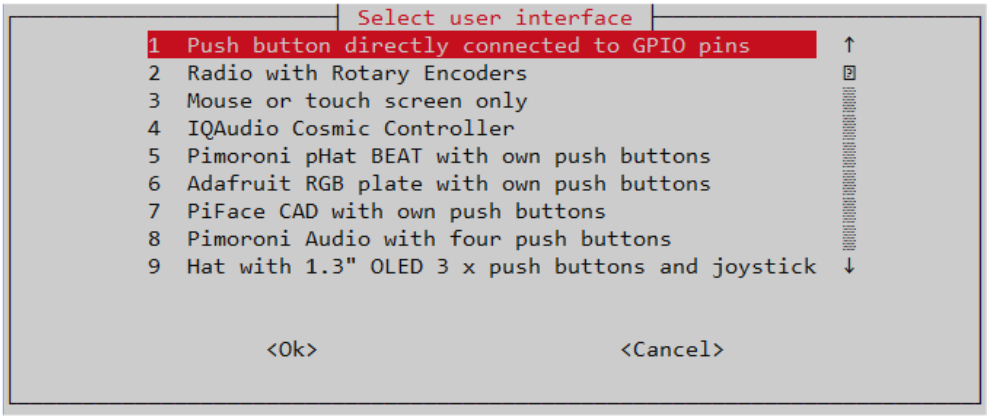


Figure 162 Configure radio - wiring selection

Normally select the 40-pin option unless you have used the 26-pin wiring scheme. See **Error! R eference source not found.** and *Table 4 Radio and DAC devices 40-pin wiring.*
Confirm selection and continue to the next screen:



Select **option 1** if push buttons are being used or **option 2** if using rotary encoders. Otherwise select the type of user interface being used in **option 3** onwards. If you scroll down, you will be given the option not to change the display configuration.
This screen and all following screens have the option to not change the configuration. The program will always ask you if your choice is correct and give you the opportunity to change the selection.

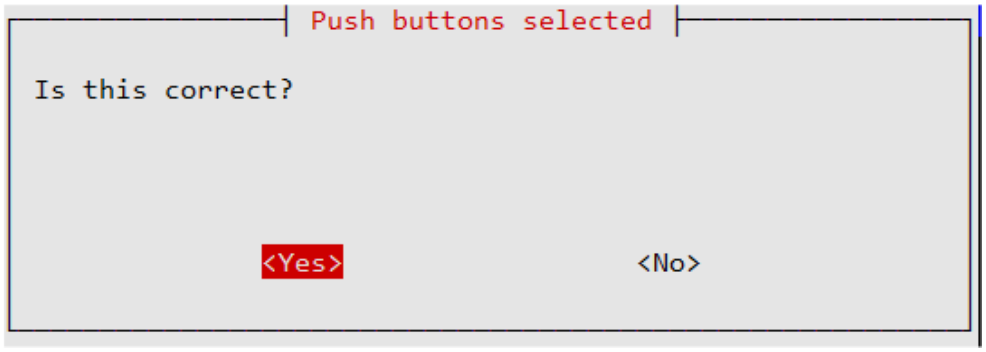


Figure 163 Configure radio - Confirmation screen

If the push-button interface is selected, the program will ask how they have been wired. The push-buttons can be wired to either +3.3V (The original scheme) or GND(0V).

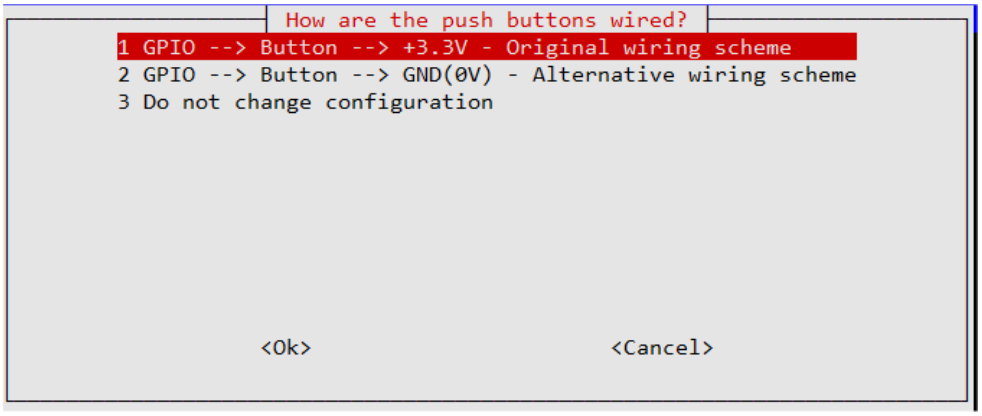


Figure 164 Push-button voltage selection

If option **2 Rotary Encoders** was selected then the following screen will be seen:

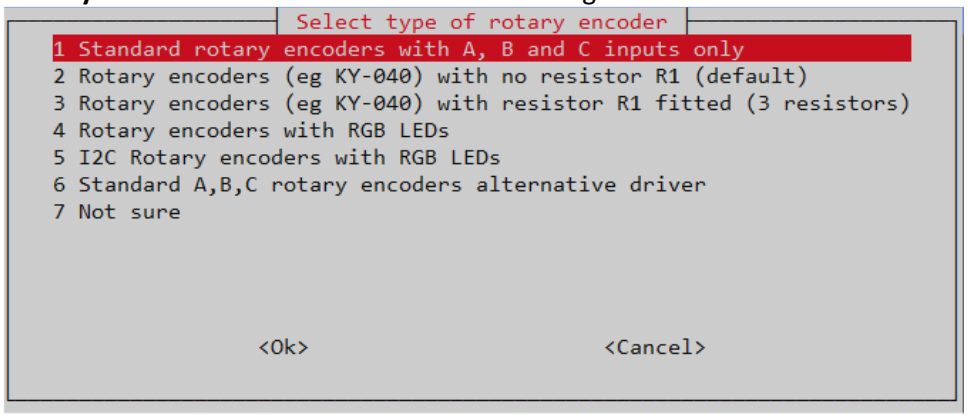


Figure 165 Rotary encoder type selection

Traditional rotary encoders have three connections called A, B and C. They usually also have two connections to a switch (See Radios using rotary encoder on page 44). If these rotary encoders are being used the select option 1 as shown above.

If rotary encoders such as the KY-040 (See *Using KY-040 Rotary encoders* on page 45) are being used, these have their own 10K pull-up resistors and so do not need the internal GPIO pull-up resistors as well. Select option 2 KY-040 encoders to disable the internal GPIO pull-up resistors.

If you are missing the push-button 10K resistor as shown in *Figure 83 KY-040 with three 10K* resistor on page 46 then select option 2 (KY040 with resistor R1 fitted). If resistor R1 is missing select option 2 (KY040 with no resistor R1).

If using rotary encoders with RGB LEDs select option 4.
If there are problems with the standard A, B, C rotary encoders try option 4 - Alternative driver.

Note: Option 2 will not work with traditional A, B and C encoders. If unsure select option 3 and the internal pull-up resistors will be enabled and will work with both types of encoders.

Next select the GPIO header wiring. This is either 26-pin (Older RPi models) or 40-pin wiring.
Note that the 26-pin wiring scheme can also be used on 40-pin headers if so wished.

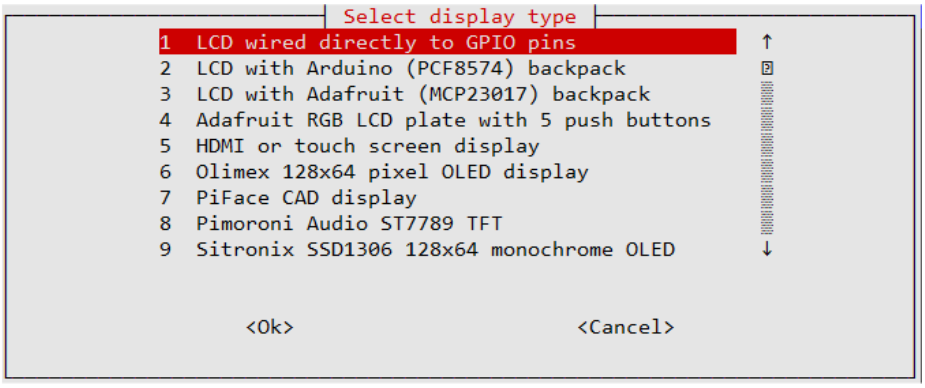


Figure 166 Configure radio - Display interface selection 1

Notice that there is a scroll bar to the right of the options. Use the Up/Down keys to scroll down to the remaining options. Select the option which matches the display you are using.

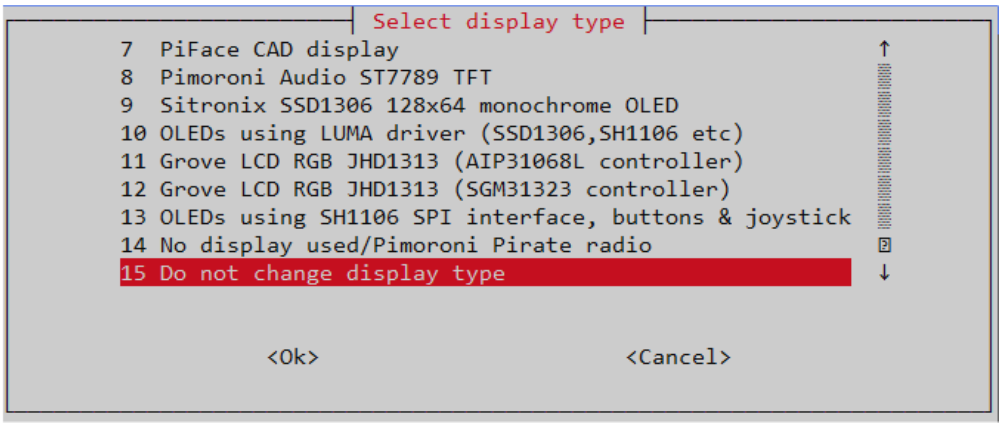


Figure 167 Configure radio - Display selection 2

Select the correct option for the display interface and confirm selection. Again, it is possible leave the configuration unchanged. There is also an option 10 (Scroll down) called “Do not change display type”.
If option 5 – ‘HDMI or touch screen display’ was selected the go to the section called *Configuring HDMI monitor and* Touchscreen on page 121.
If options 6, 8, 9 or 10 were selected go to *Configuring OLEDs* on page 113.

If option 2, 3, 4 or 6 was selected then this will require the hex address to be configured. Otherwise, the program will skip the screens in the next section and go to the section called *Select the type of LCD* display on page 112.

If option 6 (Olimex OLED) was selected then the following will be displayed:

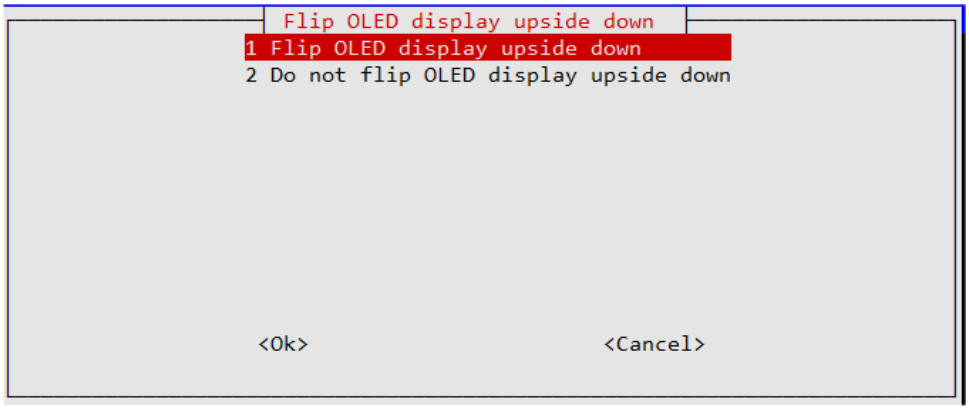


Figure 168 Olimex OLED flip display

This option sets the **flip_display_vertically** parameter in **/etc/radiod.conf** to **yes** or **no** and allows the Olimex OLED display to be flipped vertically. This option doesn't work with the ST7789TFT.

Select the type of LCD display

Skip this section if you are not using an LCD display directly connected to the GPIO pins. Confirm the selection and continue to the next screen to select the type of LCD display. This section is not relevant for a HDMI or touch screen.

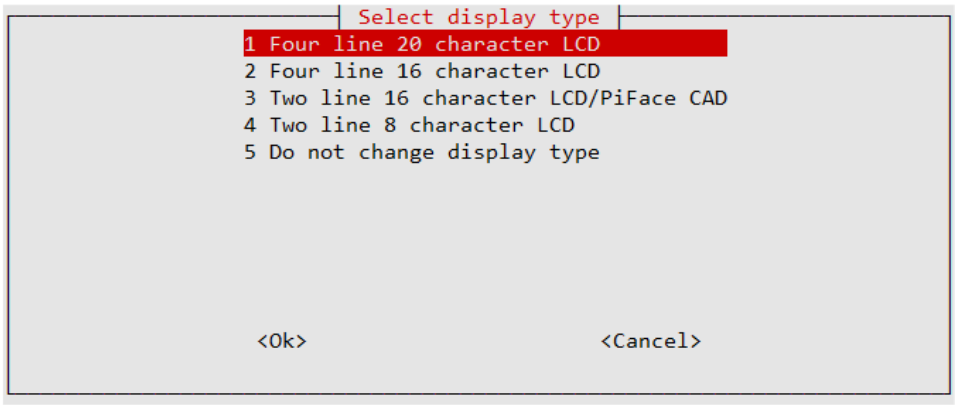


Figure 169 Configure radio - Display type selection

Select the type of display to be used and confirm the selection. The installation script asks if you wish to configure the audio device:

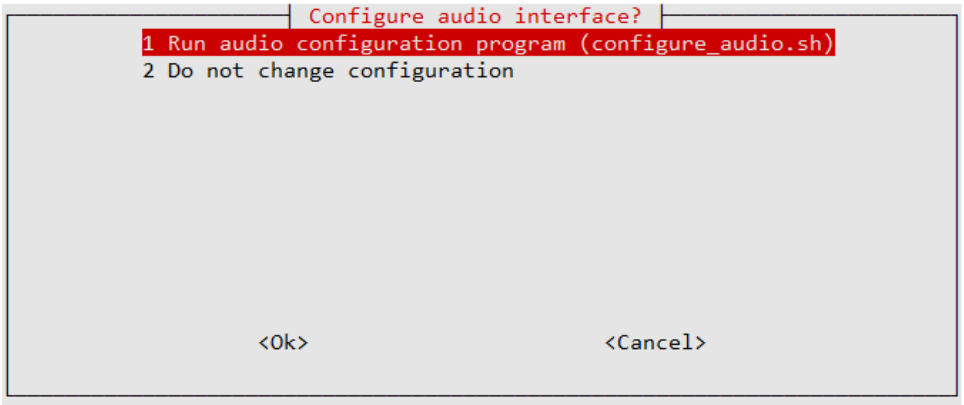


Figure 170 Configure radio audio output option

You should select option 1 to run the audio device configuration program.
If you selected option 1 then go to the section *Configuring the audio output* on page 117.

Configuring OLEDs

If not using an OLED display, then go to the section *Configuring the audio output* on page 117.

There are five drivers available for OLED displays. These can have either the 2-wire **I2C** interface (+5v, GND, Data and Clock) or the 4-wire **SPI** interface. They are selected by the **display_type** parameter in **/etc/radiod.conf**. This is setup by the **configure_radio.sh** configuration program.

Table 15 OLED rivers

display_type	Interface	Lines	Driver	Option	Notes
ST7789TFT	SPI	5	st7789tft_class.py	8	Used by the Pimoroni Pirate Audio hat
OLED_128x64	I2C/SPI	5	oled_class.py	6	The driver automatically detects I2C or SPI
SSD1306	I2C	4	ssd1306_class.py	9	From version 7.2 onwards
LUMA	I2C	4	luma_class.py	10	From version 7.3 onwards
SH1106_SPI	SPI	4	sh1106_class.py	11	From version 7.6 onwards

The Luma device supports multiple devices namely devices using SSD1306, SSD1309, SSD1325, SSD1331, SH1106 or WS0010 hardware driver chips. Only SSD1306 and SH1106 devices have been tested.

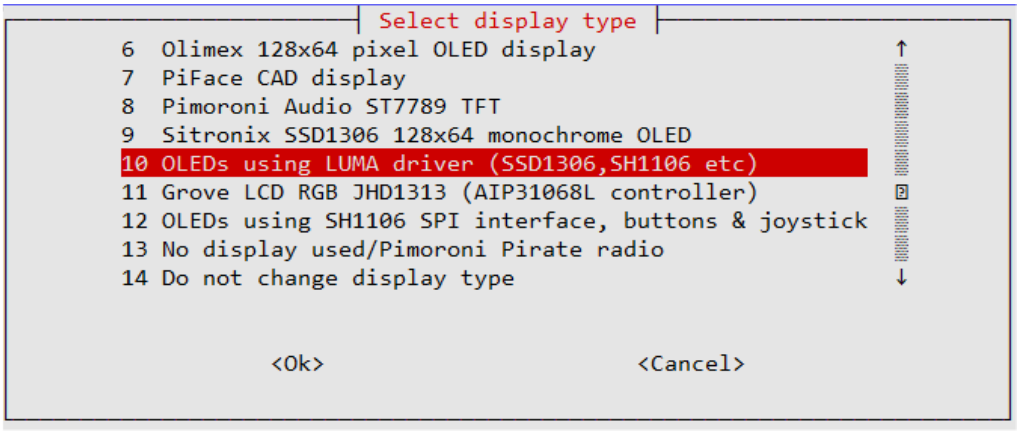


Figure 171 Selecting an OLED display

Select option 6, 8, 9, 10 or 11 depending upon the type of OLED being configured.

If option 10 (Luma device driver) was selected the following screen will be displayed:

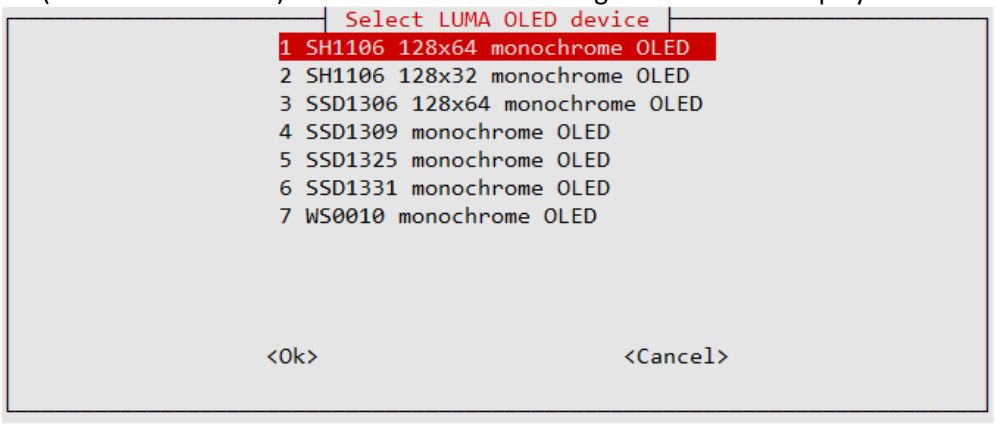


Figure 172 Luma devices

Select the device that your OLED is using.

If using devices with an I2C interface then select the appropriate I2C device address. OLED devices usually use address **0x3C** but check your device specification or run **i2c_detect.py -y 1** to display it.

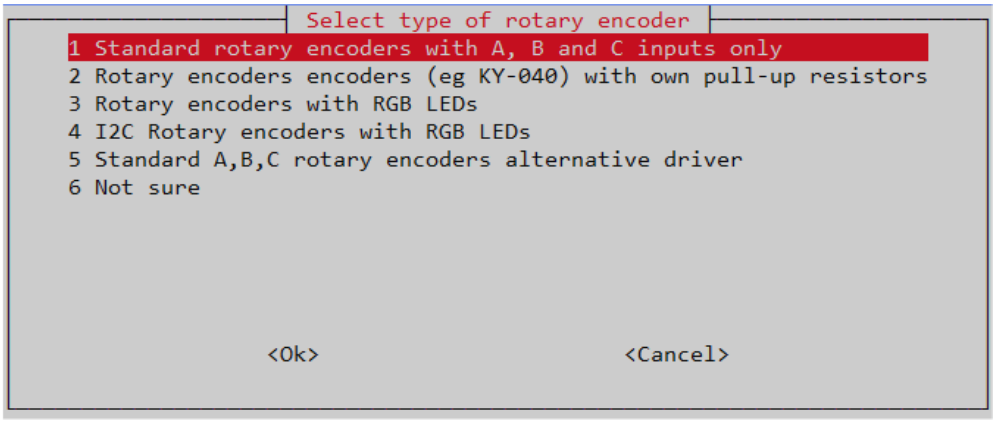


Figure 173 Selecting the OLED I2C address

After configuring I2C address go to the section *Configuring the audio output* on page 117.

Installing the HDMI or touch screen software

This section is only relevant if configuring an HDMI or touchscreen interface. If using an LCD or OLED display then skip this section.

Note: The **X-Windows** system for graphical programs including **gradio** and **vgradio** has changed in **Bookworm**. The traditional X11 windows protocol is being replaced with the new **Wayland** Windows protocol and new Window Manager called **Wayfire**. Bookworm now comes with **Wayland/Wayfire** configured as the default Windowing system.

The following screen is displayed:

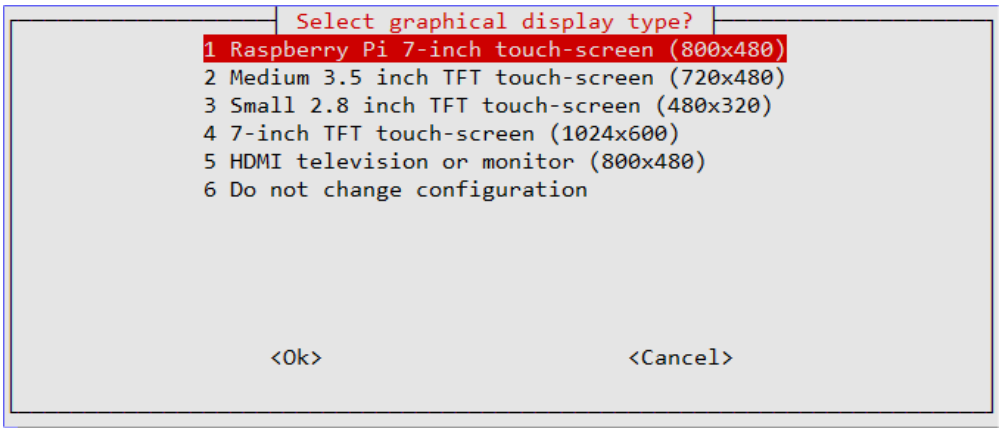


Figure 174 Touchscreen selection

Select the type of screen that is connected to the Raspberry Pi.

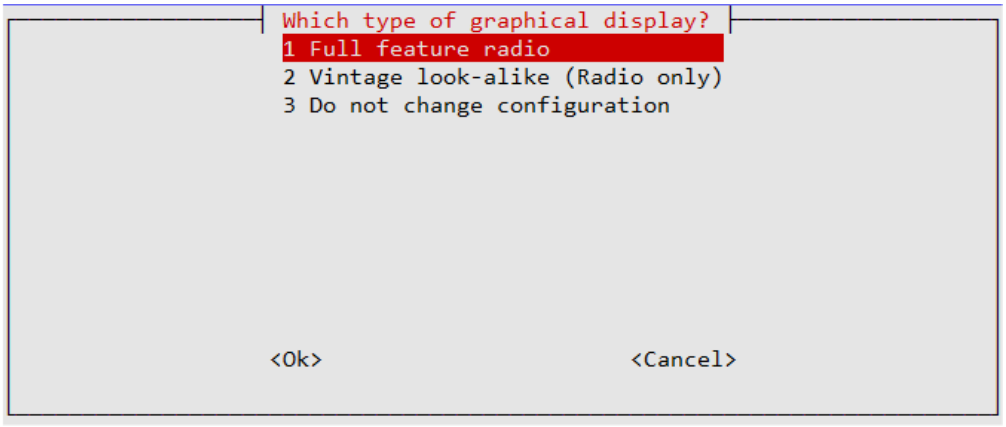


Figure 175 Selecting the type of radio display

Select which radio program to start-up. See *Figure 2 Raspberry pi 7-inch touchscreen radio* and *Figure 4 Vintage tuning touch-screen radio* on page 7.

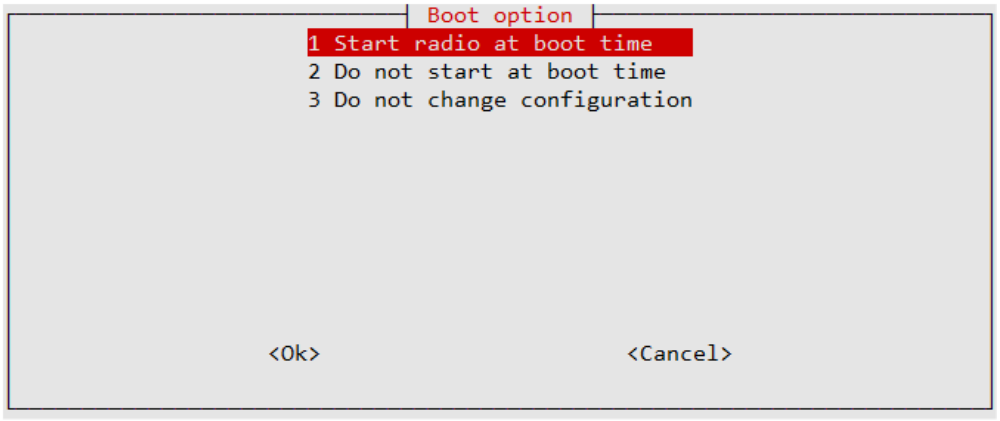


Figure 176 Configuring the HDMI or touch screen display startup

Normally select option 1 to automatically start the gradio.py program when the Graphical Desktop is loaded. This copies a desktop configuration to the file **/home/pi/Desktop/gradio.desktop** file. There is also a similar file created called **vgradio.desktop** for the vintage graphical radio. For more information see the Appendix **Error! Reference source not found.** on page **Error! Bookmark not defined.Error! Bookmark not defined.**.

The installation script also copies the graphic screen configuration to **/etc/radiod.conf**. It also disables start-up of the **radiod** service which is only used for the LCD versions of the radio.
Now select the option to display the radio full screen (7-inch touchscreen) or in a desktop window (Large HDMI monitor or TV).

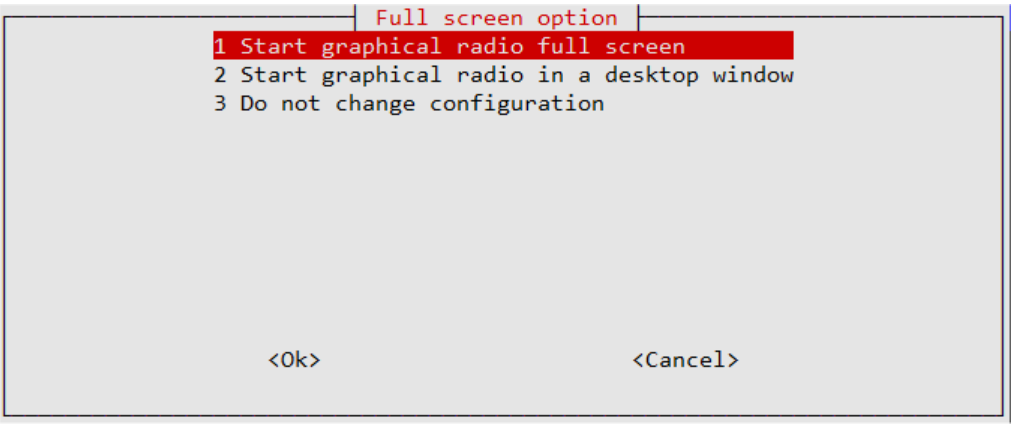


Figure 177 Configuring the graphical radio full screen

Configuration of the HDMI/Touch screen is shown in the section **Error! Reference source not found.** on page **Error! Bookmark not defined.**. Operation of the HDMI/Touch screen is shown in the section called *Operation of HDMI and touch screen displays* on page 172.

Configure the graphical radio startup programs for Wayland/Wildfire

As mentioned previously Bookwork now comes with **Wayland X-protocol** and **Wayfire Window manager** as default in **Bookworm**.

You can check which X-Window system is running with the following command:

```
$ echo $XDG_SESSION_TYPE
wayland
```

In the above example the X-windowing system is running Wayland.

Wayland is configured differently from the traditional **X-11** Windowing System.
It is configured in a file called **wayfire.ini** which you will find in the **.config** directory in your user home directory (**~/.config/wayfire.ini**). Edit this file.

```
$ sudo vi ~/.config/wayfire.ini
```

Add the following two lines to the end of **wayfire.ini**.

```
[autostart]
radio = sudo /usr/share/radio/vgradio.py
```

The first part (radio) is just a name and can be anything. The second part starts **vgradio** when the Desktop becomes active (or configure gradio.py for the full feature version of the radio. It must be started using **sudo**.

Reboot the Raspberry Pi to activate.

```
$ sudo reboot
```

Configuring the audio output

At the end of the radio configuration process the radio installation calls the **configure_audio.sh** script which then displays the screen shown below in Figure 178.

Currently it is not possible to configure anything but the onboard audio jack or HDMI when the **configure_audio.sh** script is called directly from the initial installation script.

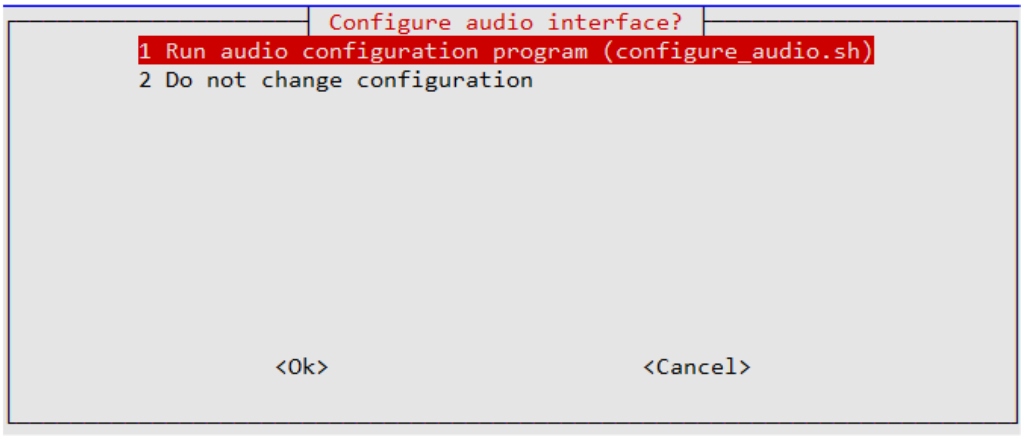


Figure 178 Audio configuration selection

If you wish to configure the on-board audio select option 1. If you wish to configure DAC, USB or Bluetooth devices then select option 2 to exit the program, then run the following command:

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

If building a new radio start by using the Raspberry Pi on-board audio output jack. Leave configuring a digital audio card such as HiFiBerry or IQaudIO until later unless you are using a Raspberry Pi Zero which doesn't have an on-board audio jack. In that case there is no choice but to configure the sound card. The package installation **configure_radio.sh** script will automatically run the **configure_audio.sh** script.

The following screen will be displayed:

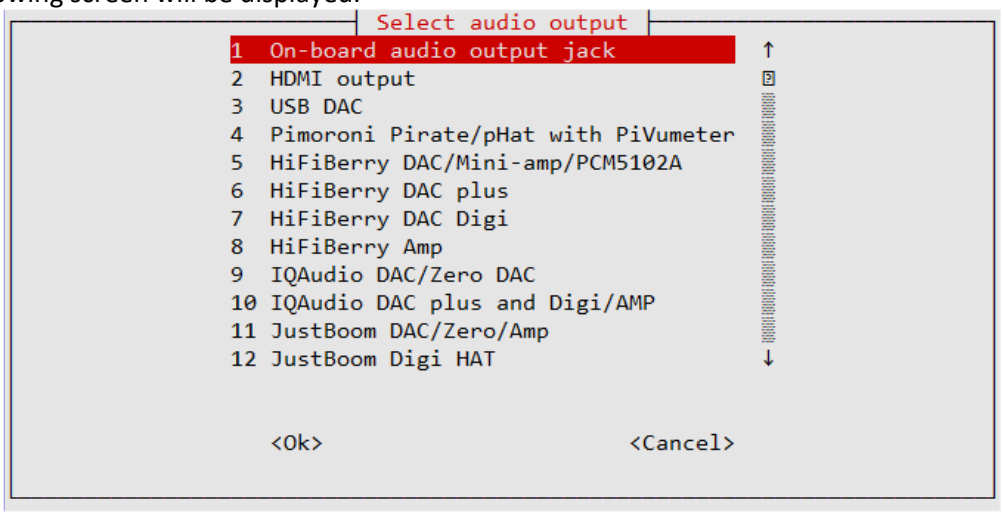
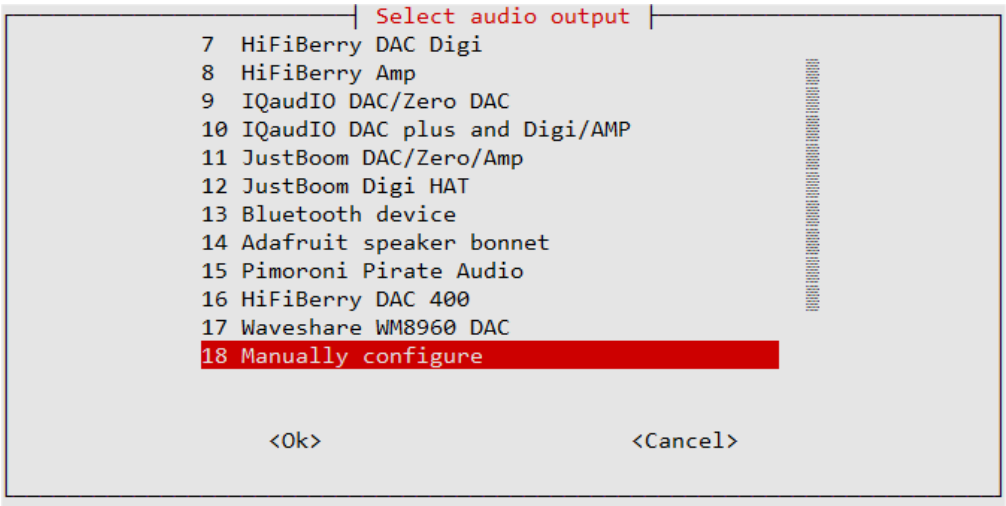


Figure 179 Selecting the audio output device

Use the Up/Down keys to scroll down to the remaining options:



Please note the scroll bar on the right side of the above screen. There are more options after option 12 (Bluetooth device). Use the Up/Down keys to scroll up and down.

When the installation run is completed, the following text will be displayed.

```
:
PI Radio software successfully installed
See /usr/share/doc/radiod/README for release information
Installation complete
It is necessary to reboot the system to start the radio
Run the radio-config program to further configure the radio software
```

The configuration program can be safely re-run at any time in the future. To do this run the **radio-config** utility from the command line. See the /usr/share/doc/radiod/README file for release notes.

Installation logs

A log of the changes made by the radio configuration program will be written to the **/usr/share/radio/logs/install.log** file. For the audio configuration program changes will be written to the **/usr/share/radio/logs/audio.log** file.

Reboot to enable the software

The software is installed in the **/usr/share/radio** directory. Now reboot the Raspberry PI.

```
$ sudo reboot
```

Once rebooted the software should run and display the time and station details on the screen. However, sound might not be heard. This is because final audio configuration has to be done. Restart the radio with the following command.:

```
$ sudo systemctl start radiod
```

Once restarted music should be heard out of the on-board audio jack. If not go to the section called *Chapter* 10 -Troubleshooting on page 209.

The radio daemon (LCD versions only) can be started and stopped with the **systemctl** command:

```
$ sudo systemctl start radiod
$ sudo systemctl stop radiod
```

This will also stop and start the MPD daemon.

To prevent automatic start-up of the radio at boot time run the following command:

```
$ sudo systemctl disable radiod
```

To re-enable it:

```
$ sudo systemctl enable radiod
```

Testing the I2C interface.

Version 6.13 onwards comes with its own **SMbus** library (smbus2) in the smbus2 sub-directory of the radio package.

The I2C interface should have already been enabled by the installation program. To test the I2C interface, carry out the following:

If you are using a revision 2 Raspberry Pi (Newer boards) carry out the following:

```
$ sudo i2cdetect -y 1
```

If you are using a revision 1 Raspberry Pi (Very old V1 boards) carry out the following:

```
$ sudo i2cdetect -y 0
```

This will search **/dev/i2c-0** (Very old v1 RPIs) or **/dev/i2c-1** (Later RPi versions) for all address, and if correctly connected, it should show up at **0x20** for the Adafruit LCD Plate or normally **0x27** for the Arduino PCF8574 backpack but might be another address such as **0x3F.** The OLED 128x64 pixel display uses address **0x3C**. See Figure 180 *The I2C bus display using the i2cdetect program*.

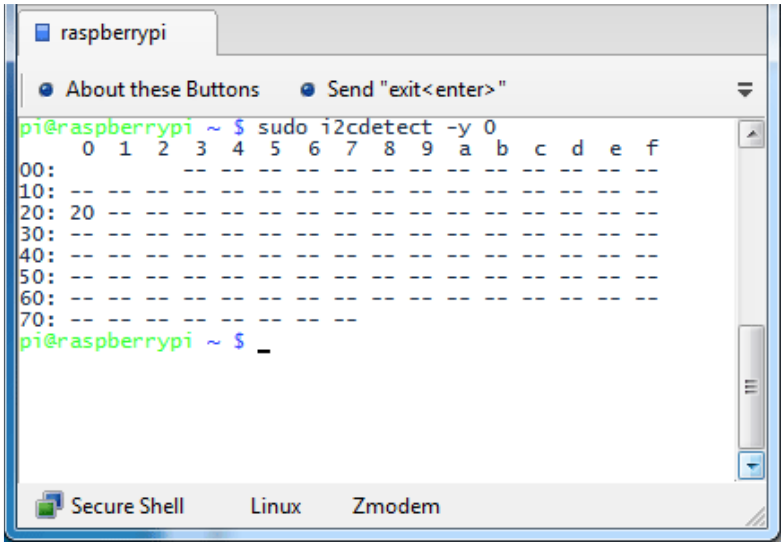


Figure 180 The I2C bus display using the i2cdetect program

If the following is seen instead then it is necessary to run enable the I2C module at boot time using **raspi-config**.

```
Error: Could not open file `/dev/i2c-1' or `/dev/i2c/1': No such file or
directory
```

If problems with **i2cdetect** are still encountered, then edit the **/boot/firmware/config.txt** (for **Bookworm**) **or /boot/config.tx** (for **Bullseye**) file using **sudo nano** and change the following line:

```
#dtparam=i2c_arm=on
```

Change to:

```
dtparam=i2c_arm=on
```

Also, the **i2c-dev** module must be added to the **/etc/modules** file.

```
i2c-dev
```

Reboot and retry the i2cdetect program.

Note: If the Arduino **PCF8574** backpack is using another address other than **0x27**, **0x37** or **0x3F** then you must modify the **i2c_address** parameter in **/etc/radiod.conf**. For example, if the backpack is using the address **0x2F** then modify the **i2c_address** parameter to match this as shown in the example below:

```
# The i2c_address parameter overides the default i2c address. 0x00 = use default
# Some backpacks use other addresses such as 0x3F, then set i2c_address=0x3F
i2c_address=0x2F
```

Once both of these packages have been installed, you have everything you need to get started accessing I2C and SMBus devices in Python. Now reboot the Raspberry PI.

```
$ sudo reboot
```

The Radio should start automatically. If not then go to the section called *Chapter* 10 - Troubleshooting on page 209.

Configuring HDMI monitor and Touchscreens

Configuring an HDMI monitor or TV screen

If using a touch-screen or HDM TV/Monitor add the following lines to **/boot/firmware/config.txt**.

```
hdmi_group=2
hdmi_mode=4
hdmi_cvt 800 480 60 6 0 0 0
max_usb_current=1
```

If the screen upside-down then add the following line.

```
# Rotate screen 180
lcd_rotate=2
```

Now carry out the instructions shown in section *Installing* the Radio Daemon on page 107.

Waveshare TFT device installation

If not using Waveshare devices skip this section.

Full instructions for installing and configuring the Waveshare TFTs will be found at:
https://www.waveshare.com/wiki/RPi_LCD_User_Guides

Below are the basic installation instructions extracted from the above site. First clone the Waveshare installation software.

```
$ cd
$ git clone https://github.com/waveshare/LCD-show.git
```

It is now necessary to change to the LCD-show directory and run the correct installer for the type of Waveshare screen being installed. For example:

LCD28-show Waveshare TFT 2.8“
LCD35-show Waveshare TFT 3.5“
LCD35B-show Waveshare TFT 3.5“ type B
LCD35C-show Waveshare TFT 3.5“ type C

There are other installation scripts for different Waveshare LCD models in the

Below is an example of the installation for the Waveshare TFT 3.5“ type C.

```
$ cd LCD-show/
```

```
$ ./LCD35C-show
```

This command to rotate the screen 180 degrees and can be run at any time (reboot required)

```
$ ./LCD35C-show 180
```

However, do refer to the Waveshare documents as the above are only very basic instructions.

Edit **/boot/firmware/config.txt** (for **Bookworm**) **or /boot/config.txt** (for **Bullseye**) and force the console to 720 x 480.

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

Warning: The Waveshare installation script currently copies its own **config.txt** to **/boot/config.txt** and will overwrite any previous settings however this is for Bullseye. For Bookworm the configuration file is **/boot/firmware/config.txt**. For example, Bluetooth settings. The file it copies to config.txt (config-32c.txt-retropie) is also in **DOS** format. Use **dos2unix** to convert it back to Unix format. See below

```
$ sudo apt install dos2unix
$ sudo dos2unix /boot/config.txt
```

Now copy any changes in **/boot/config.txt** to **/boot/firmware.txt** if using Bookworm.

If you are using Bluetooth then after running the Waveshare configurator and correcting the DOS format add the following two lines to the **/boot/firmware/config.txt** configuration file.

```
dtoverlay=pi3-miniuart-bt
core_freq=250
```

Below is an example of Waveshare driver code added to the **/boot/firmware/config.txt** configuration file.

```
dtoverlay=waveshare35b:rotate=270
hdmi_force_hotplug=1
hdmi_group=2
hdmi_mode=1
hdmi_mode=87
hdmi_cvt 480 320 60 6 0 0 0
hdmi_drive=2
display_rotate=0
```

Now carry out the instructions shown in section *Installing* the Radio Daemon on page 107.

MHS 3.5-inch RPi display installation

If not using MHS devices skip this section.

Full instructions for installing and configuring the MHS devices will be found at:
http://www.lcdwiki.com/MHS-3.5inch_RPi_Display

Below are the basic installation instructions extracted from the above site. First clone the Waveshare installation software.

```
$ cd
$ git clone https://github.com/goodtft/LCD-show.git
```

It is now necessary to change to the **LCD-show** directory and run the correct installer for the type of MHS screen being installed. For example:

MHS35-show MHS 3.5-inch RPi display
MHS55-show MHS 5.5-inch RPi display
MHS35B-show MHS 3.5-inch RPi display
LCD7B-show MHS 7-inch RPi display
There are other installation scripts for different MHS LCD models in the

Below is an example of the installation for the MHS TFT 3.5“ RPi display

```
$ cd LCD-show/
$ ./MHS35-show
```

This command to rotate the screen 180 degrees and can be run at any time (reboot required)

```
$ ./MHS35-show 180
```

However, do refer to the MHS documents as the above are only very basic instructions.

Edit **/boot/firmware/config.txt** and force the console to 720 x 480.

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

Now carry out the instructions shown in section *Installing* the Radio Daemon on page 107.

Using other touch screens

There are various other touch screens on the market but this version of the software does not support screen sizes of less the 480 x 320 pixels. There is a **screen_size** parameter in **/etc/radiod.conf** configuration file.

```
# Size is in pixels. Supported is 800x480 (7" screen) or 720x480(3.5"
screen)
# or 480x320 (2.8" or 3.5" screen)
screen_size=800x480
```

Another important aspect of screen size are the following parameters in **/boot/firmware/config.txt**.

```
framebuffer_width=1280
framebuffer_height=720
```

Changing the above can force a console size. By default, it will be display's size minus overscan settings in **/boot/firmware/config.txt.**

If using the Elecrow 7-inch TFT Capacitive touch screen display then see **Error! Reference source not found.** on page **Error! Bookmark not defined.**.
Now carry out the instructions shown in section *Installing* the Radio Daemon on page 107.

Installing the Grove LCD RGB

The **Grove LCD** with RGB backlight is capable of displaying any colour wanted. For instance, the colour AQUA would be RGB (5,195,221). To turn the word "AQUA" into its RGB value the PIL (Python Image Library) package is required. Install it with the following command.

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

Now carry out the instructions shown in section *Installing* the Radio Daemon on page 107.

Installing PiFace CAD software

Version 7.4 onwards re-introduces **PiFace CAD**. The PiFace CAD software is no longer supplied as a package but as source which has to be compiled into a library (egg). It also requires the **SPI** interface to be enabled and also needs the **lirc** package to be installed.

First install git which is required to download the software from https://github.com/piface

```
$ sudo apt install git
```

Install **pifacecommon**

```
$ cd
$ git clone https://github.com/piface/pifacecommon.git
$ cd pifacecommon/
$ sudo python setup.py install
$ sudo python3 setup.py install
```

Install **pifacecad**:

```
$ cd
$ git clone https://github.com/piface/pifacecad.git
$ cd pifacecad/
$ sudo python setup.py install
$ sudo python3 setup.py install
```

Now install the IR remote control software as shown in the section called *Install the IR remote control software* on page 141. Put the settings for the IR Sensor to use GPIO25 and 0 for the IR remote Led (No LED available).

If you configured the IR sensor to use LIRC for **Bullseye** also carry out the instructions below.
Amend the LIRC import in the pifacecad **ir.py** Python script

```
$ cd /usr/local/lib/python3.9/dist-packages/pifacecad/
$ cp ir.py ir.py.lirc
```

Edit **ir.py** using sudo nano or sudo vi and change:

```
import lirc
```

to

```
import pylirc
```

Run the Hello World demo using Python3:

```
$ python3
>>> import pifacecad
>>> cad = pifacecad.PiFaceCAD()    # create PiFace Control and Display
object
>>> cad.lcd.backlight_on()         # turns the backlight on
>>> cad.lcd.write("Hello, world!") # writes hello world on to the LCD
```

A full test can be run with the following:

```
$ cd /usr/share/radio
$ ./lcd_piface_class.py
```

Configuring sound devices

Other sound devices can be used with the radio. Currently supported are the following devices:

- CMedia USB speakers or devices (See page 126)
- Sound cards such as **HiFiBerry**, **IQaudIO**, **JustBoom, RPi audio devices** and **Pimoroni pHat** DAC and DAC+ products (See page 127)
- Bluetooth speakers or headphones (See page 153).

To check if the audio device is present run the **aplay** command.

```
$ 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 [USB PnP Sound Device], device 0: USB Audio [USB Audio]
  Subdevices: 0/1
  Subdevice #0: subdevice #0
```

In the above example **Card 0** is the on-board devices namely the audio output jack and HDMI. **Card 1** is a USB PnP sound device.

To configure other sound devices run the **configure_audio.sh** utility.

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

Configuring a USB sound device

To configure a USB DAC sound devices such as CMedia speakers or sound dongles run the **configure_audio.sh** utility.

To configure USB audio devices run the Run the **configure_audio.sh** utility.

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

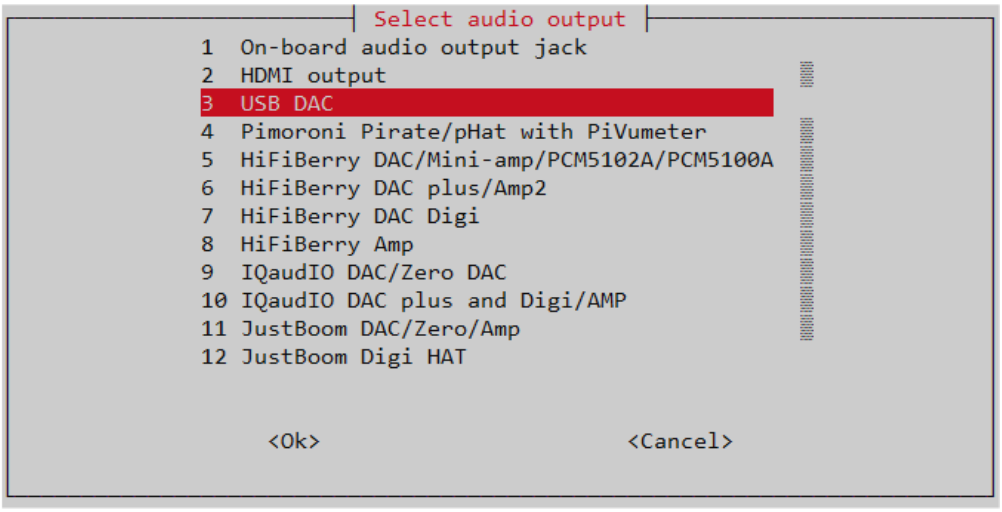


Figure 181 Configure USB DAC

Reboot when prompted. After rebooting the Raspberry Pi run the **alsamixer** program.

```
$ alsamixer
```

The following screen is displayed:

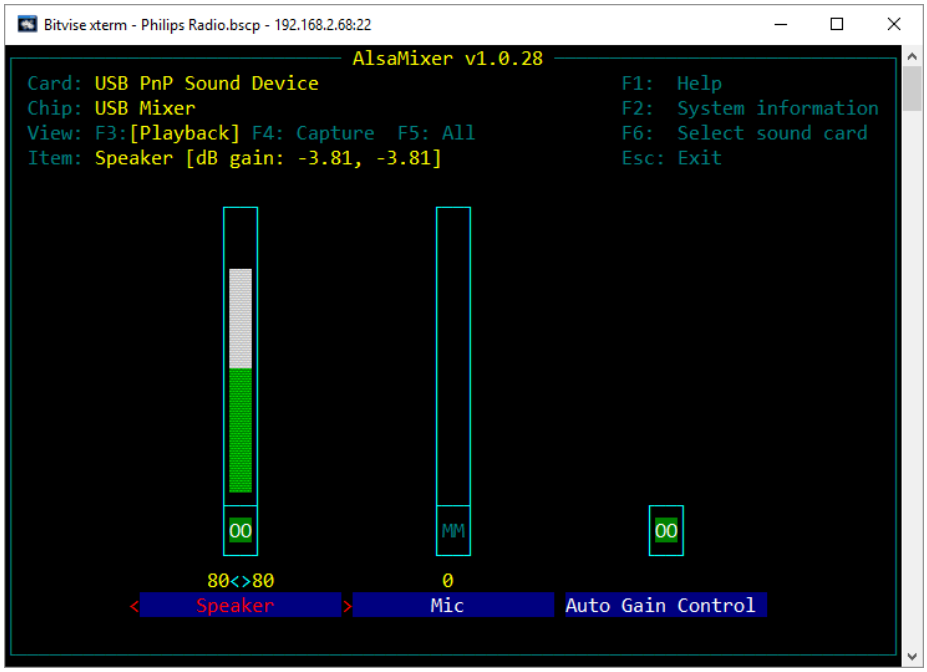


Figure 182 The USB PnP Alsa Mixer

Use the Left and Right keys to position on the 'Speaker field'. Adjust the sound level using the Up and Down keys (80% in the above example). Pres **Esc key** or **Ctl Z** key to exit.

Configuring a Sound Card

This section covers configuration of add on DAC boards such as **HiFiBerry**, **IQaudIO** and **JustBoom DAC**, **DAC+** and Amplifier products. Older versions of the **HiFiBerry DAC** that used the 26-pin GPIO header are not supported.

To configure add on audio cards run the Run the **configure_audio.sh** utility:

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

The following screen will be displayed.

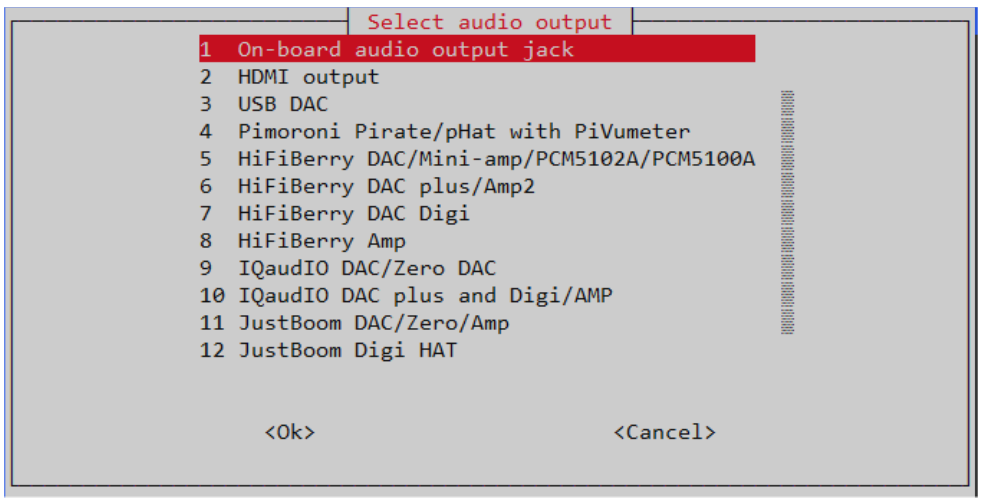


Figure 183 Configuring the onboard audio jack output

More options are available by scrolling down with the down arrow key:

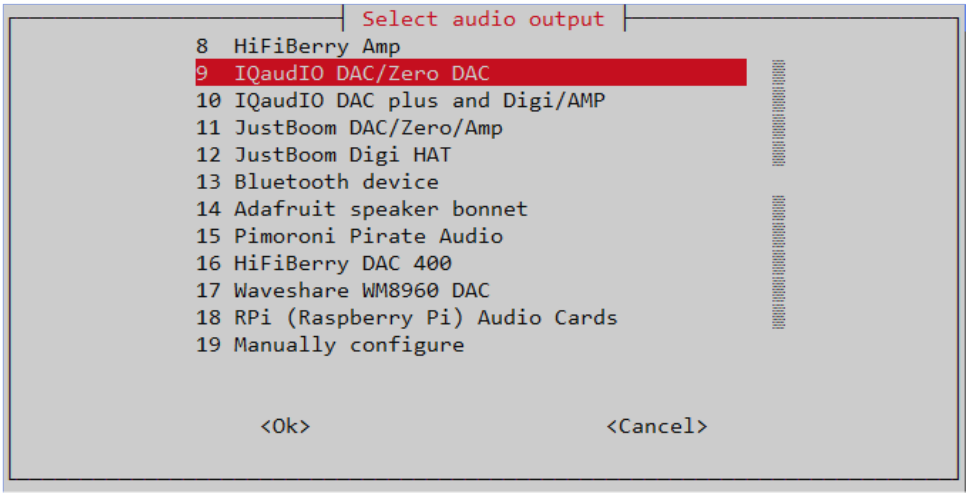


Figure 184 Configuring add-on DAC sound cards

Select option for the DAC being used and press OK. Reboot when prompted by the next screen.
If using Bluetooth devices such as speakers or headphones then select option 13 Bluetooth device.

The Pimoroni pHat is compatable with HiFiBerry DAC (Not DAC+) and uses the same Device Tree (DT) overlay so select HiFiBerry DAC if using the pHat.

To configure Raspberry Pi Sound Cards select option 18 RPi (Raspberrry Pi) Audio Cards.

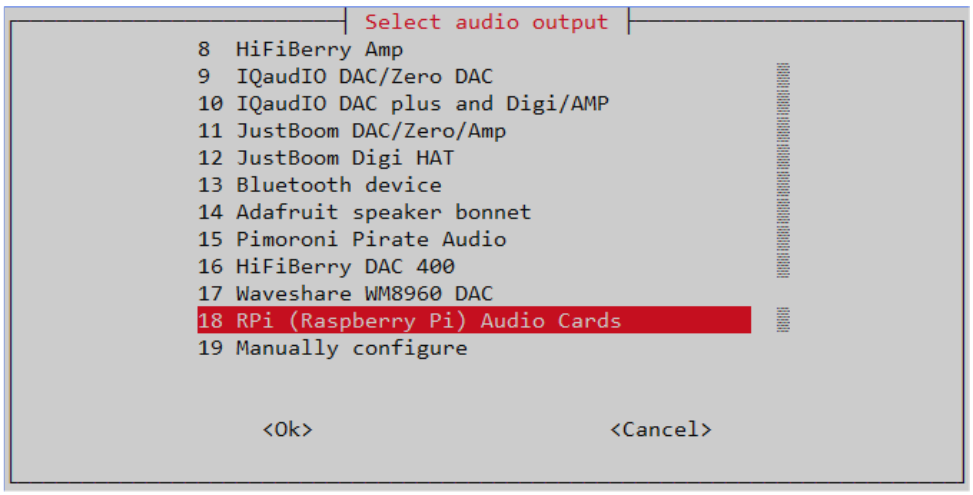


Figure 185 Selecting Raspberry Pi Audio cards

The following screen will be displayed:

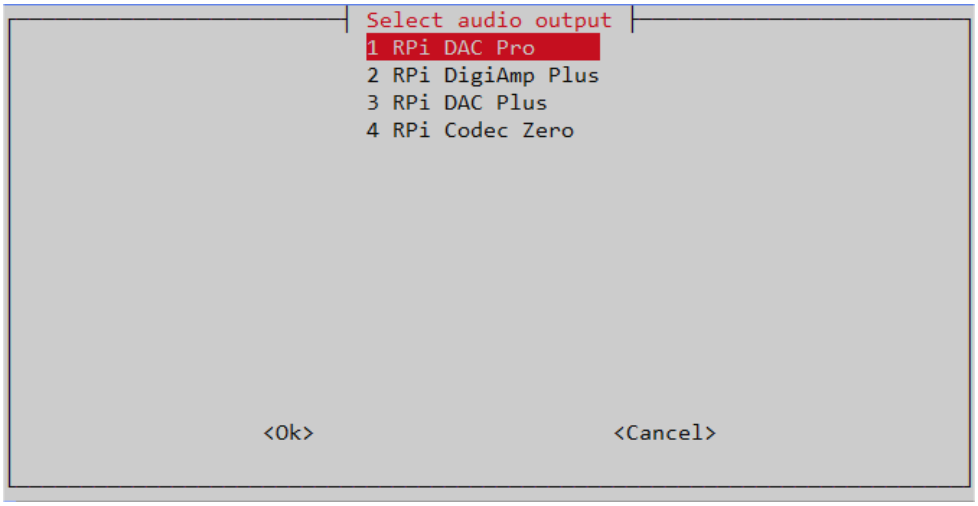


Figure 186 Raspberry Pi DAC options

The program displays the configuration changes it has made. Reboot after it has completed.

```
$ reboot
```

After rebooting run the **alsamixer** program.

```
$ alsamixer
```

Use the left and right keys to select the mixer control (Analogue) and use the up down keys to change the volume to 100%.

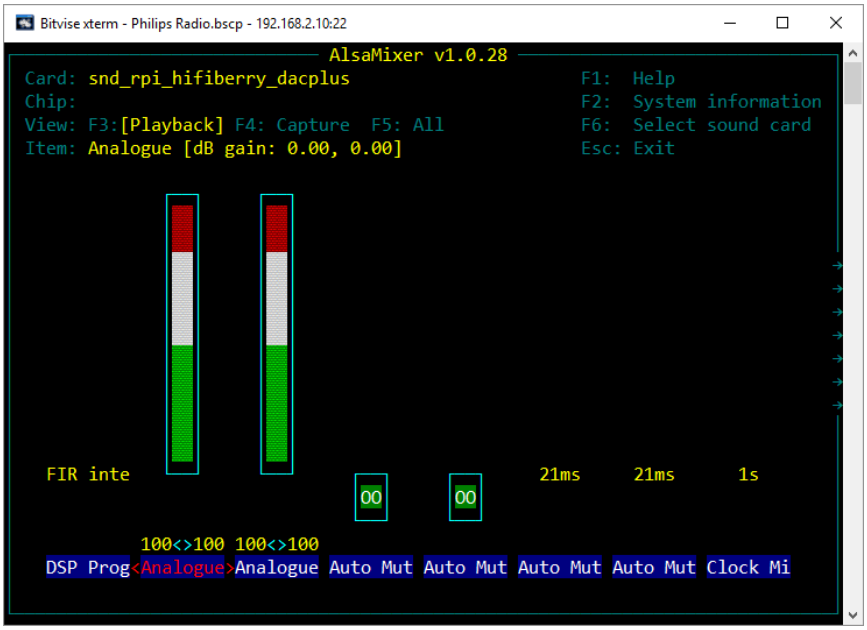


Figure 187 Set mixer analogue volume

Next use the right key to position on the “Digital” mixer control and use the up down keys to change the mixer volume:

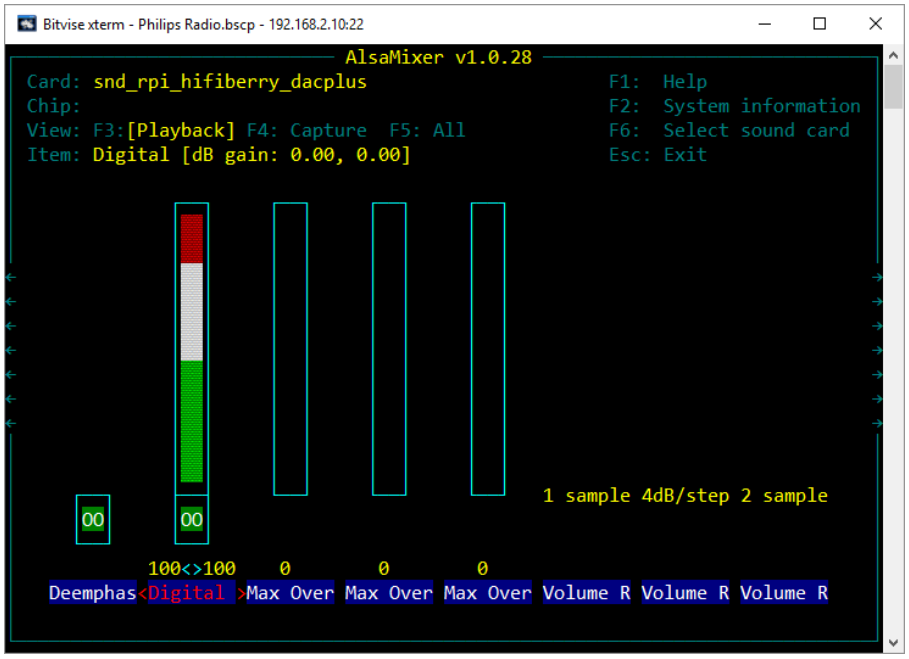


Figure 188 Set mixer digital volume

Configuring Allo Sound Cards

The Allo Piano, in particular, has two outputs, one for normal sound ranges and one for a sound woofer. The board comes pre-programed as 2.1 for the Raspberry Pi. The subwoofer has a right and left output but is mono only.

However, it uses a second I2S channel on GPIO5 and mute signal on GPIO6 both used to control the woofer amplifier. This conflicts with the LCD signals **lcd_data4** an **lcd_data5** pins. Other Allo products are not affected as they only use one I2S channel. There are two ways to correct this.

Method 1

Edit **/etc/radiod.conf** and locate the LCD GPIO connection definitions.

```
# 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
```

Change the **lcd_data4/5** assignments in **/etc/radiod.conf** to:

```
lcd_data4=26
lcd_data5=27
```

Wire LCD **data4/5** to GPIO 26 (physical pin 37) and GPIO 27 (physical pin 13).

Method 2

Configure the LCD to use an I2C backpack. This frees up 6 GPIOs including GPIO5 and 6. See *Construction using an I2C LCD backpack* on page 61.

Configuring Allo driver dtoverlay

Allo manufacture several audio products as well as the Piano 2.1 with woofer. Currently these cannot be configured via the radio installation program. It is necessary to load the correct **device tree overlay** (**dtoverlay**) by adding the following line for the corresponding **Allo** audio card the end of the **/boot/firmware/config.txt** file.

Allo Piano HIFI DAC

```
dtoverlay=allo-piano-dac-pcm512x-audio
```

Allo Piano 2.1 HIFI DAC with woofer

```
dtoverlay=allo-piano-dac-plus-pcm512x-audio
```

Allo Boss HIFI DAC / MINI BOSS HIFI DAC

```
dtoverlay=allo-boss-dac-pcm512x-audio
```

Allo DIGIONE

```
dtoverlay=allo-digione
```

Also disable the on-board output jack by amending it in the **/boot/firmware/config.txt** file.

```
dtparam=audio=off
```

Reboot the Raspberry Pi to load the Allo dtoverlay.

```
$ sudo reboot
```

Verification

Check, if the sound card is enabled with “aplay”. The output below will vary depending upon which Allo DAC has been selected:

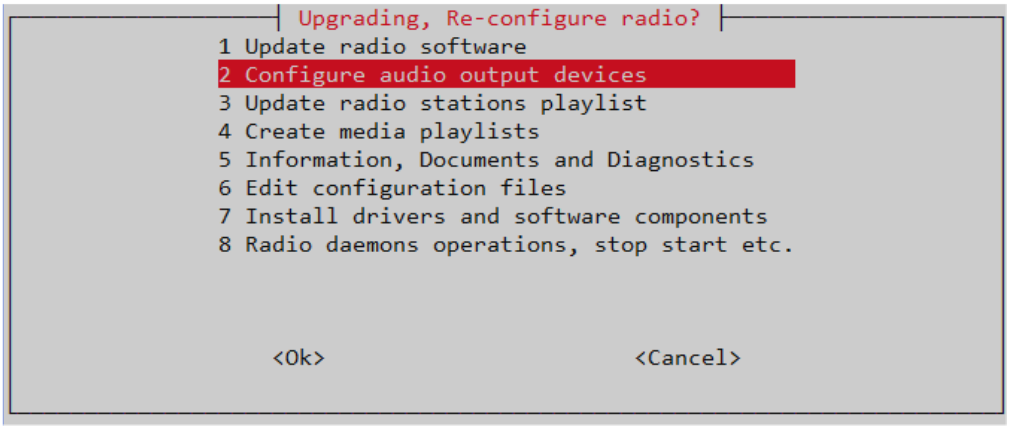
```
$ aplay -l
**** List of PLAYBACK Hardware Devices ****
card 0: PianoDAC [PianoDAC], device 0: PianoDAC multicodec-0 []
 Subdevices: 0/1
 Subdevice #0: subdevice #0
```

Installing the Waveshare WM8960 DAC

Note: Waveshare have dropped support for the WM8960 sound card on **Bullseye**. This means that this card can only be installed on the **Bookworm 32** or **64-bit** OS.

Run the radio-config utility and select option “2 Configure audio output devices”

```
$ radio-config
```



Scroll down to Select option 17 and select it.

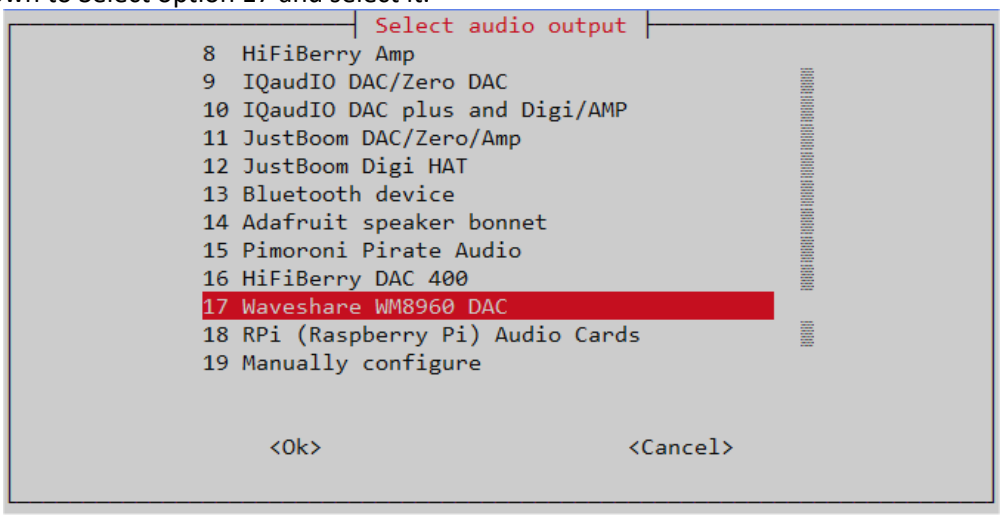


Figure 189 Waveshare WM8960 DAC selection

Setting the Headphone and Speakers Volume

Using **aplay -l** to display WM8960 sound card. The **wm8960soundcard** should be visible and should be **card 0**. If not check that all the steps in this procedure have been carried out.

```
$ aplay -l
**** 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
```

The WM8960 sound card has separate controls for the speaker volume and the headphone volume. However, the headphone volume is set to 0db. To set the headphone volume, run **alsamixer**:

```
$ alsamixer
```

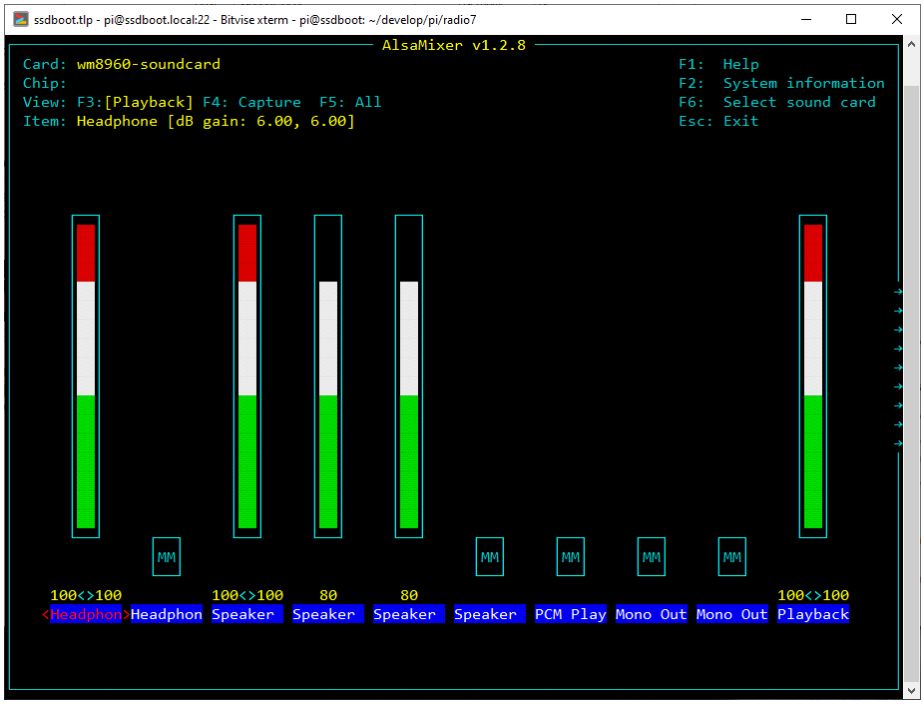


Figure 190 Alsamixer setting the Headphone Volume

Set the headphone volume to maximum, press Esc to exit, then store the settings with the following command.

```
$ sudo alsactl store --file  /etc/wm8960-soundcard/wm8960_asound.state
```

Reboot the Raspberry Pi to test.

```
$ sudo reboot
```

Uninstalling the Waveshare WM8960 sound card

```
$ cd /usr/share/radio/WM8960-Audio-HAT
$ sudo ./uninstall.sh
```

Now run the **configure_audio.sh** configuration script as shown in *Configuring the audio output* on page 117.

Installing a Bluetooth audio device

Install the Bluetooth software

If you haven't already done so update the operating system first as shown in the section Update to the latest the system packages on page 95. To install the Bluetooth software do the following

- Power on your Bluetooth device
- Run the **radio-config** utility.
- Select option **8 Install/configure drivers and software components** from the **radio-config** menu shown in *Figure 193 The radio-config menu* on page 140.
- Select option **7 Install a Bluetooth audio device**

The following menu will be displayed:

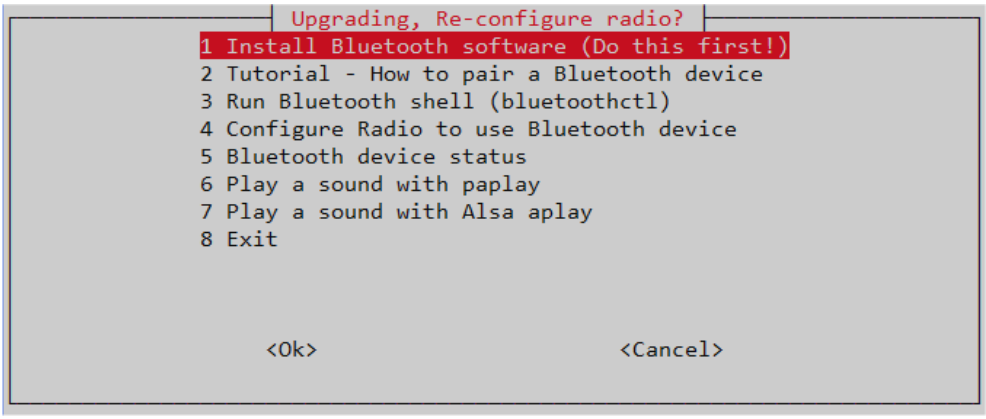


Figure 191 The Bluetooth Configuration Menu

The menu is arranged in the order that you need to do things. Start by selecting option **1 Install Bluetooth software**. As it says do this first do this first.

The installation software will display the following:

```
/usr/share/radio/configure_bluetooth.sh Bluetooth configuration log, Wed 13
Nov 09:47:45 GMT 2024
Boot configuration in /boot/firmware/config.txt
Removing pipewire package
sudo apt -y remove pipewire
:
sudo usermod -G bluetooth -a pi
Bluetooth software installation complete
```

Pairing your Bluetooth device

Read **option 2 Tutorial - How to pair a Bluetooth device**. The operations to pair a device are summarised below:
Select option or run **bluetoothctl** from the command line. In the **bluetoothctl** prompt run the following instructions shown in bold:

```
[bluetooth]# power on
[bluetooth]# agent on
Agent registered
[CHG] Controller B8:27:EB:A3:E4:52 Pairable: yes
[bluetooth]# default-agent
Default agent request successful
[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
[bluetooth]# scan off
:
[CHG] Controller DC:A6:32:05:36:9D Discovering: no
Discovery stopped
[bluetooth]# pair 00:75:58:41:B1:25
Attempting to pair with 00:75:58:41:B1:25
:
Pairing successful
[bluetooth]# connect 00:75:58:41:B1:25
Connection successful
[SP-AD70-B]#trust 00:75:58:41:B1:25
```

Note: The example device In the above output 00:75:58:41:B1:25 will be different for your device.

Now display the info for your 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
:
[SP-AD70-B]# exit
```

You should see the highlighted options set as above.

Configure radio software for Bluetooth

Select option **4 Configure Radio to use Bluetooth device** to configure the Radio (radiod) and Music Player Daemon (MPD) configuration files to use your Bluetooth device.

```
 /usr/share/radio/configure_bluetooth.sh Bluetooth configuration log, Wed 13
Nov 10:52:28 GMT 2024
Device 00:75:58:41:B1:25 SP-AD70-B
Paired 00:75:58:41:B1:25 found
:
Copying /usr/share/radio/asound/asound.conf.dist.blue to  /etc/asound.conf
Configuring audio_out parameter in /etc/radiod.conf
audio_out="bluetooth"
A log of this run will be found in /usr/share/radio/logs/bluetooth.log
```

This completes the installation.

Testing the Bluetooth device

Reboot the Raspberry Pi. If everything is OK you should hear radio on your Bluetooth device. If not try rebooting again. If that doesn't help, stop the radio.

```
$ sudo systemctl stop radiod
```

Re-run radio-config and select the Bluetooth configuration menu as previously shown. There are two tests available from the menu.

- **6 Play a sound with paplay**
- **7 Play a sound with Alsa aplay**

The **paplay** option is the most basic sound test. You should hear a piano playing. If no sound heard, first check that your device is connected and normal by display the status with option **5 Bluetooth device status**.

If it is not connected, power on Bluetooth adapter and connect it with option **3 Run Bluetooth shell**

```
[SP-AD70-B]# power on
[SP-AD70-B]# connect 00:75:58:41:B1:25
```

The **aplay** command relies on the Alsa and MPD configuration being correct. In particular **/etc/asound.conf** and **/etc/mpd.conf** configuration files.

The **/etc/asound.conf** should be a copy of **/usr/share/radio/asound/asound.conf.dist.blue** file configured with your Bluetooth device.

The MPD configuration in **/etc/mpd.conf** should look like the following.

```
audio_output {
    type        "alsa"
    name        "SP-AD70-B"
    device      "bluealsa"
    mixer_type  "software"
}
```

Manually configuring sound cards

Unless you have a need to manually configure some other sound card or need to troubleshoot a non-working card you can skip this section. Configuring a HiFiBerry DAC is shown in this example

Edit the **/boot/firmware/config.txt** and add the following line to the end of the file depending upon the version you are using.

```
dtoverlay=hifiberry-dacplus
```

See https://www.hifiberry.com/guides/configuring-linux-3-18-x/ for other devices.

Modify the **audio_output** section in **/etc/mpd.conf** to support the HiFiBerry DAC and software mixer.

```
audio_output {
    type        "alsa"
    name        "HiFiBerry DAC"
    device      "hw:0,0"
#   mixer_type    "hardware"
    mixer_type    "software"
}
```

Reboot the Raspberry PI.

```
$ sudo reboot
```

If no music is heard run the **alsamixer** program and set the volume to at least 80% as shown in the previous section on **HiFiBerry** devices.

Setting the mixer volume

All sound output goes through a mixer. After rebooting the Raspberry Pi, for the on-board output jack, run the **alsamixer** program:

```
$ alsamixer
```

The following screen is displayed:

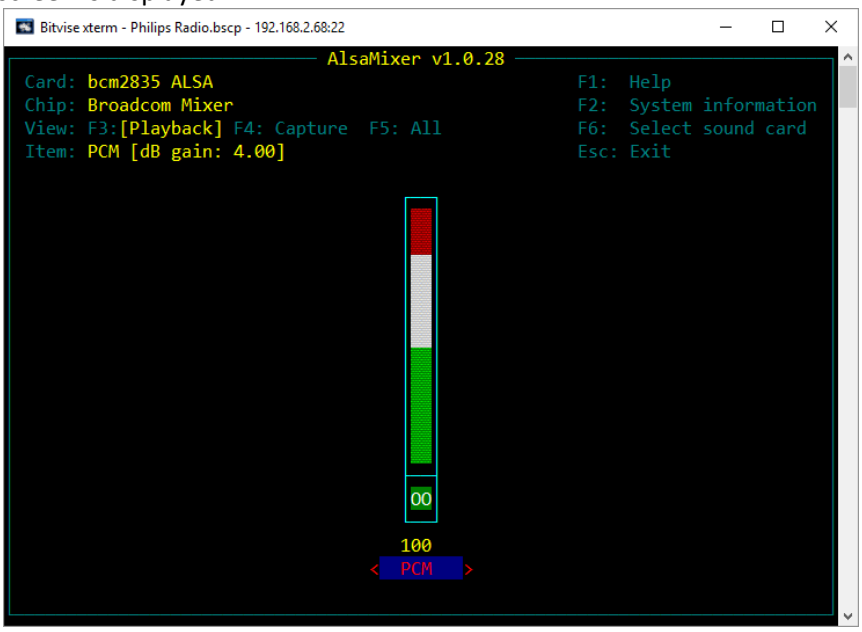


Figure 192 Basic Alsa sound mixer

The above illustration shows the **bcm2835** Alsa Mixer. There is only one mixer control called PCM (Pulse Code Modulated). Adjust the volume to 100% if not already set by using the Up and Down keys on the keyboard. Press the **Esc** key or **Ctl-Z** to exit the program.

It is also possible to set the volume for the on-board mixer volume with the **amixer** program.

```
$ amixer cset numid=1 100%
numid=1,iface=MIXER,name='PCM Playback Volume'
  ; type=INTEGER,access=rw---R--,values=1,min=-10239,max=400,step=0
  : values=400
  | dBscale-min=-102.39dB,step=0.01dB,mute=1
.toml
```

Chapter 7 Configuring additional radio components

The configure radio menu radio-config

Run **radio-config**. The following menu will be displayed:

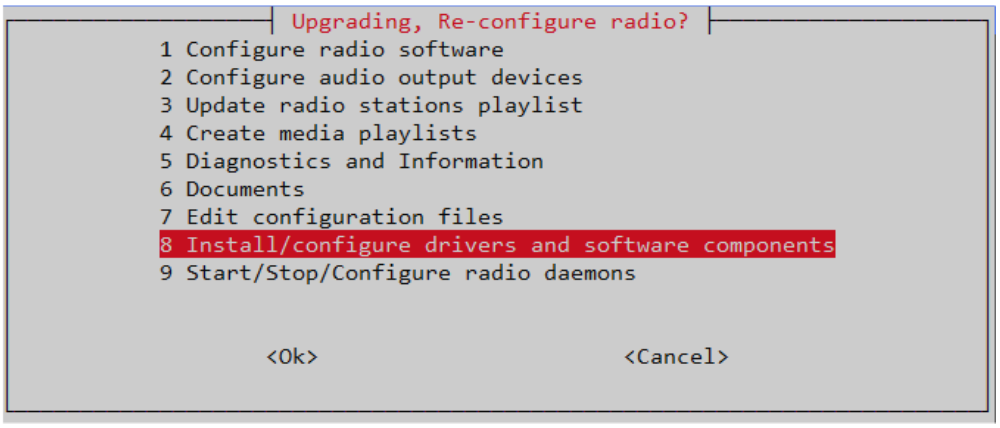


Figure 193 The radio-config menu

Select **8 Install/configure drivers and software components** from the radio-config menu

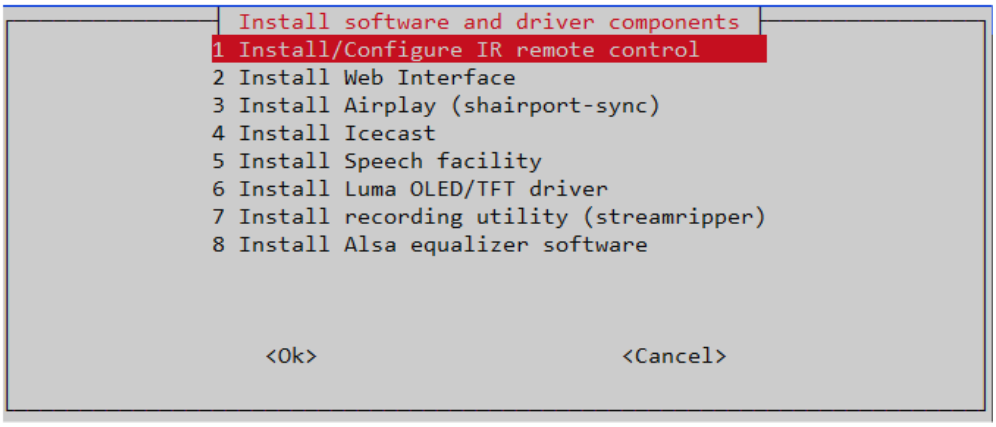


Figure 194 The additional radio software components menu

This displays additional components such as the Web interface and hardware driver software.

Installing the IR remote control software

Before starting, the **TSOP382xx** series 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 16 IR Remote Control Sensor Pin outs

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

Note: If you have wired your IR sensor to a different GPIO number 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** (Bookworm) or **/boot/ /config.txt** (Bullseye) where **x** is the GPIO number you have used.

Install the IR remote control software

The service that handles the IR sensor and its associated IR activity LED is called **ireventd**. It is installed with the radio software but needs to be configured as shown in the following pages.

If you haven't already done so update the operating system first as shown in the section Update to the latest the system packages on page 95. To configure the IR remote control software run:

- Run the **radio-config** utility.
- Select option **8 Install/configure drivers and software components** from the **radio-config** menu shown in *Figure 193 The radio-config menu* on page 140.
- Select option **1 Install IR remote control** from the **radio-config** menu shown in *Figure 193 The radio-config menu* on page 140.

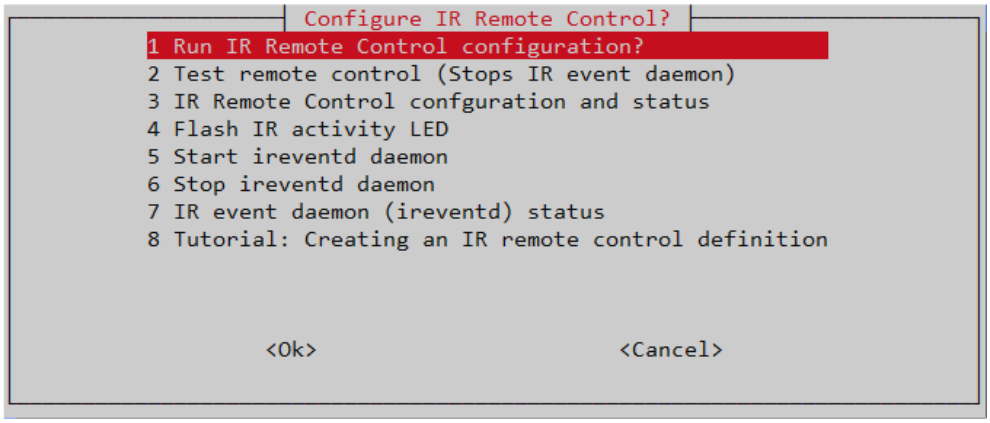


Figure 195 IR Remote Control Installation program

It is now necessary to select which GPIO is to be used for the IR sensor. This is either 9, 16 or 25 as shown in *Table 16 IR Remote Control Sensor Pin outs* above.

Select option **1 Run IR Remote Control configuration** from the menu. Select the correct GPIO pin configuration. This must match how you have physically wired up the IR sensor.

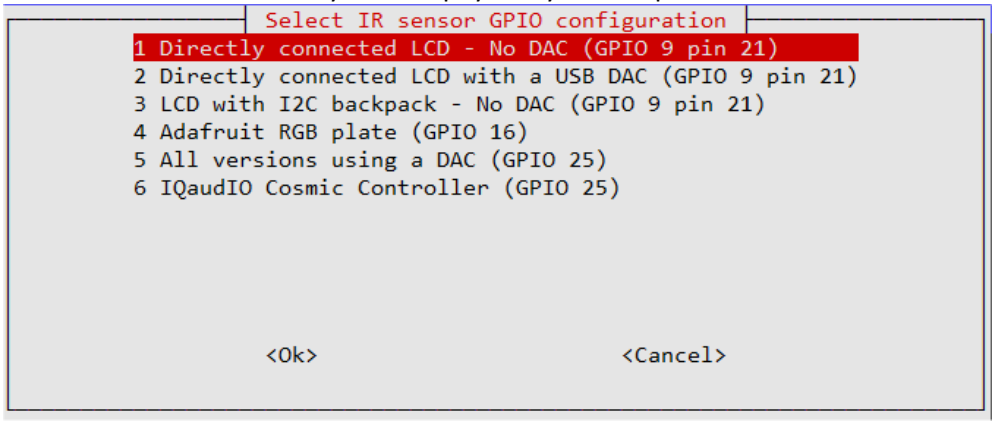


Figure 196 IR configuration IR sensor GPIO selection

Now select the Remote Activity LED GPIO. This is either GPIO 11, 13, 14 or 16. Again this must match up how you have physically wired the IR activity LED.

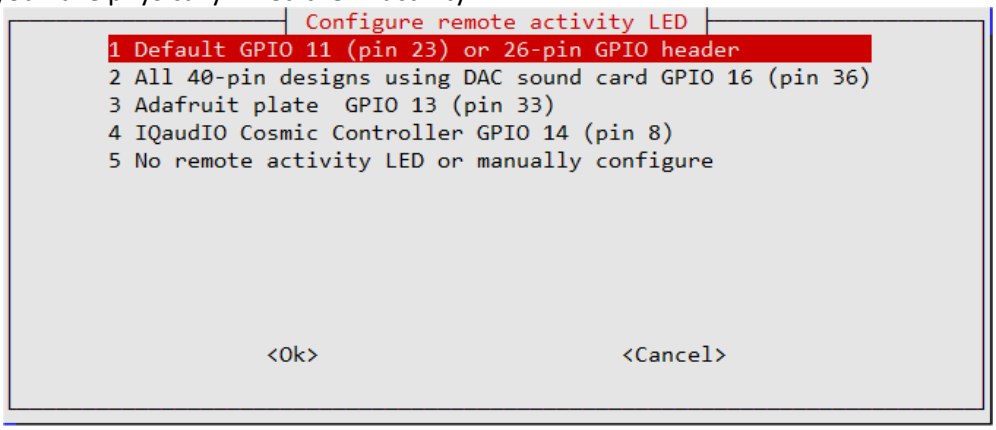


Figure 197 IR configuration Activity LED GPIO selection

Once configuration selection has been confirmed the program displays the IR remote control definition file selection. It lists the file that it finds in the **/usr/share/radio/remotes directory**.

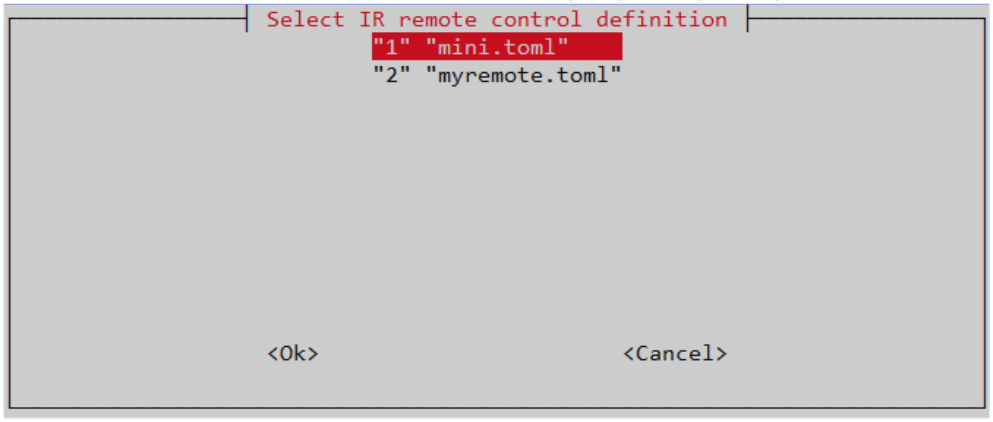


Figure 198 IR remote control definition file selection

If you are using the Raspberry Pi mini IR remote control select “mini.toml”.

Once the file selection has been confirmed 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.

```
:
A log of this run will be found in /usr/share/radio/logs/install_ir.log
Reboot the Raspberry Pi or run the following command:
sudo systemctl restart ireventd radiod
```

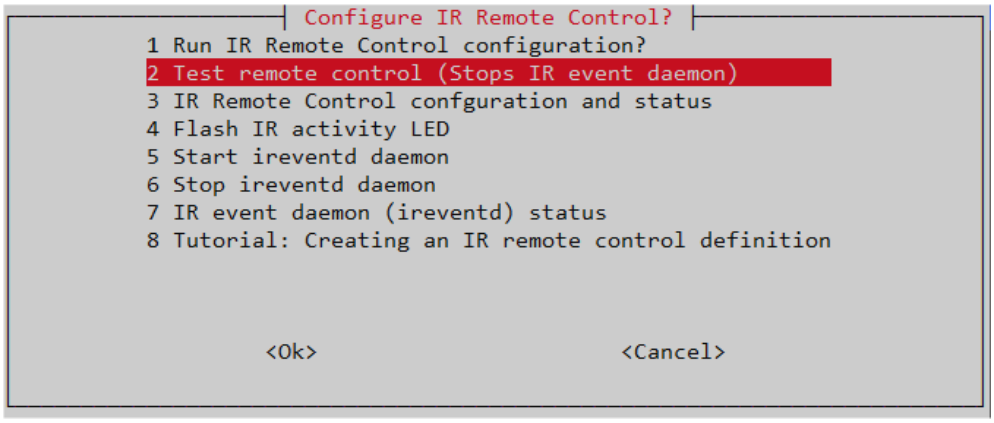
The program adds the **gpio-ir** dtoverlay to the **/boot/firmware/config.txt** file. The **gpio_pin** varies with the selection you made. For example:

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

Reboot the radio:

```
$ sudo reboot
```

After reboot check that the correct IR daemon is running. Re-run radio-config and select the IR Remote Control configuration menu as previously shown. Now select option **2 Test remote control**. Note that this will stops the IR event daemon (ireventd)



This runs the test_events.py program which will display the following

```
Press Ctl-C to end test
/dev/input/event0 vc4-hdmi-0 vc4-hdmi-0/input0
/dev/input/event1 vc4-hdmi-0 HDMI Jack ALSA
/dev/input/event2 vc4-hdmi-1 vc4-hdmi-1/input0
/dev/input/event3 vc4-hdmi-1 HDMI Jack ALSA
/dev/input/event4 gpio_ir_recv gpio_ir_recv/input0
IR input device found at /dev/input/event4
Boot configuration in /boot/firmware/config.txt
dtoverlay=gpio-ir,gpio_pin=25
Press Ctl-C to end events test
Waiting for IR events
```

Now press the buttons on the Remote Control that you configured. Output similar to the following should be displayed.

```
KEY_CHANNELUP 0x192 1
KEY_CHANNELDOWN 0x193 1
KEY_RIGHT 0x6a 1
KEY_LEFT 0x69 1
KEY_VOLUMEUP 0x73 1
KEY_VOLUMEDOWN 0x72 1
KEY_MUTE 0x71 1
```

If you are not using the Raspberry Pi mini remote control the you will have to create an IR remote control definition using **ir-keytable** program to handle IR remote control events. See the tutorial in the IR Remote Control configuration menu.

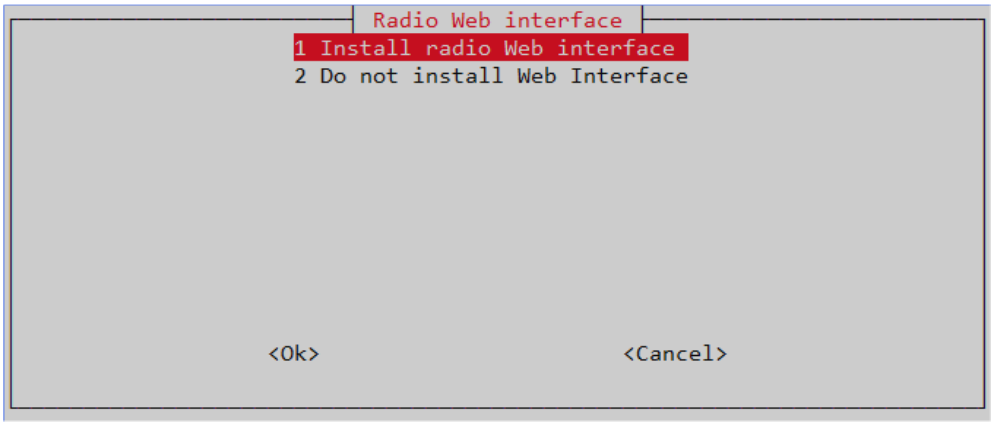
The configuration files for **ir-keytable** have the **toml** extension. These can be listed with the following command:

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

This will list over 430 different remote controls and if you are lucky, it may find one that matches the remote control you are using. If not, create one using the following method.

Installing the Web Interface

- Run the **radio-config** utility.
- Select option **8 Install/configure drivers and software components** from the **radio-config** menu shown in *Figure 193 The radio-config menu* on page 140.
- Select option **2 Install Web Interface** from the **radio-config** menu shown in *Figure 193 The radio-config menu* on page 140.



Select option **1 Install radio Web interface** or option 2 to exit. The following output will be displayed as Apache Web Server and components are installed.

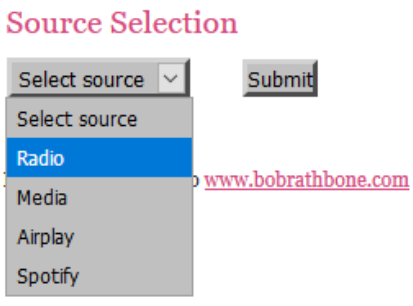
```
/usr/share/radio/install_web_interface.sh configuration log, Sun 10 Nov
08:03:25 GMT 2024
Installing radio Web interface for bookworm OS
Installing Apache Web server
Reading package lists...
Building dependency tree...
:
: {Output ommited}
:
Reading package lists...
Building dependency tree...
Reading state information...
0 upgraded, 0 newly installed, 0 to remove and 107 not upgraded.
Setting up MariaDB database
PASSWORD('raspberry')
*1844F2B11CCAEF3B31F573A1384F608BB6DE3DF9
A log of these changes has been written to
/usr/share/radio/logs/install_web.log
```

Starting the radio Web interface

Version 3.x is used in this example. Point your web browser at the IP address of the Raspberry PI. For example: http://192.168.1.251 you should see the following display :

Figure 199 Version 3.x Radio Web interface

Use the drop down box under “Select source” as shown in the following section:



The example on the left shows four music sources that can be selected namely Radio, Media, Airplay and Spotify. If installed it will also display Shoutcast.

The desired source can be selected from the source drop-down selection box. Click on the required source then click on ‘Submit’ button to load the selected source in the radio. If you have more than one Media or Radio playlist, then repeatedly clicking on the appropriate source and Submit button will cycle through the playlists for that source. The name of the new playlist, however, is not displayed.

The Shoutcast tab is explained in *Using the Shoutcast Web Interface* on page 195.

Now select the **Radio** tab. This will display **O!MPD** login screen Loging in as user **admin**, password **admin**.

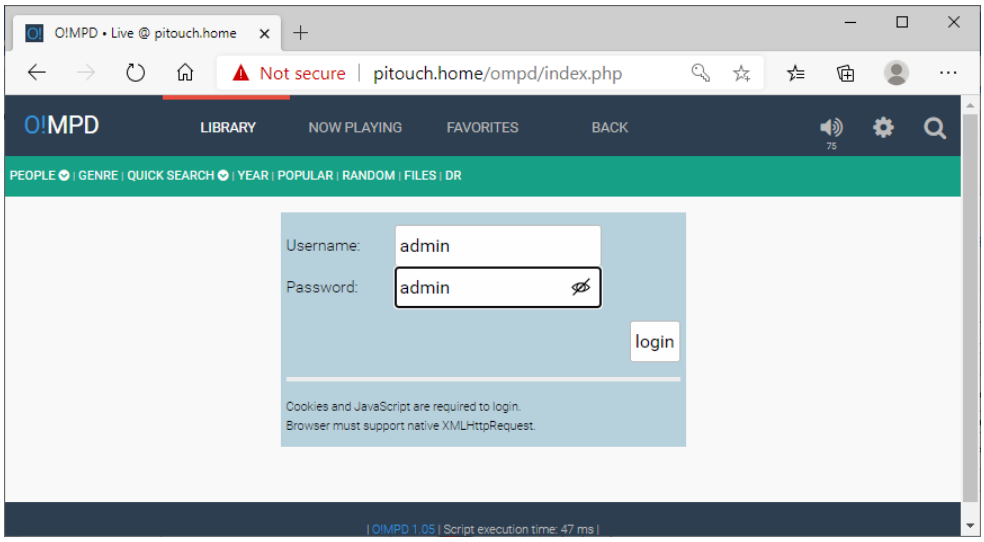
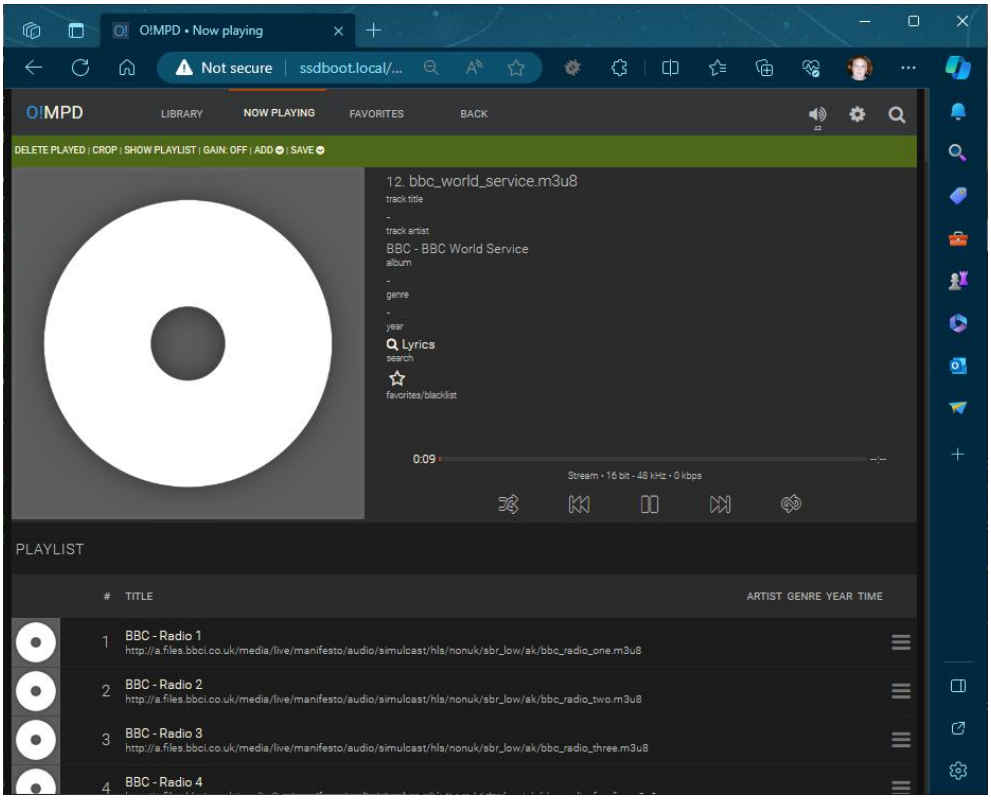
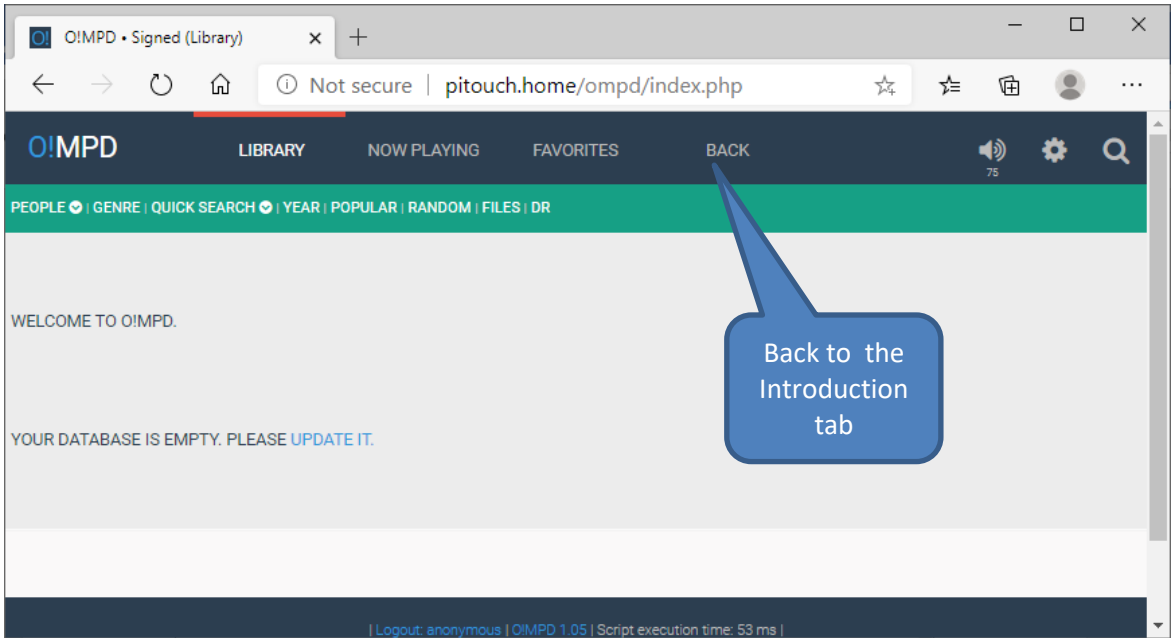


Figure 200 O!MPD Login screen

Now select the **NOW PLAYING** tab.



Note: The radio tab only displays radio stations or media tracks from MPD. It is not currently capable of displaying **Spotify** or **Airplay** details which can only be seen on the radio itself.



Note that there is an extra tab called **O!MPD** when compared to the **Snoopy** interface.
If you have already created a music library by running the **create_playlist.sh** program then it is necessary to update the **O!MPD MySql** database. Otherwise skip these instructions.

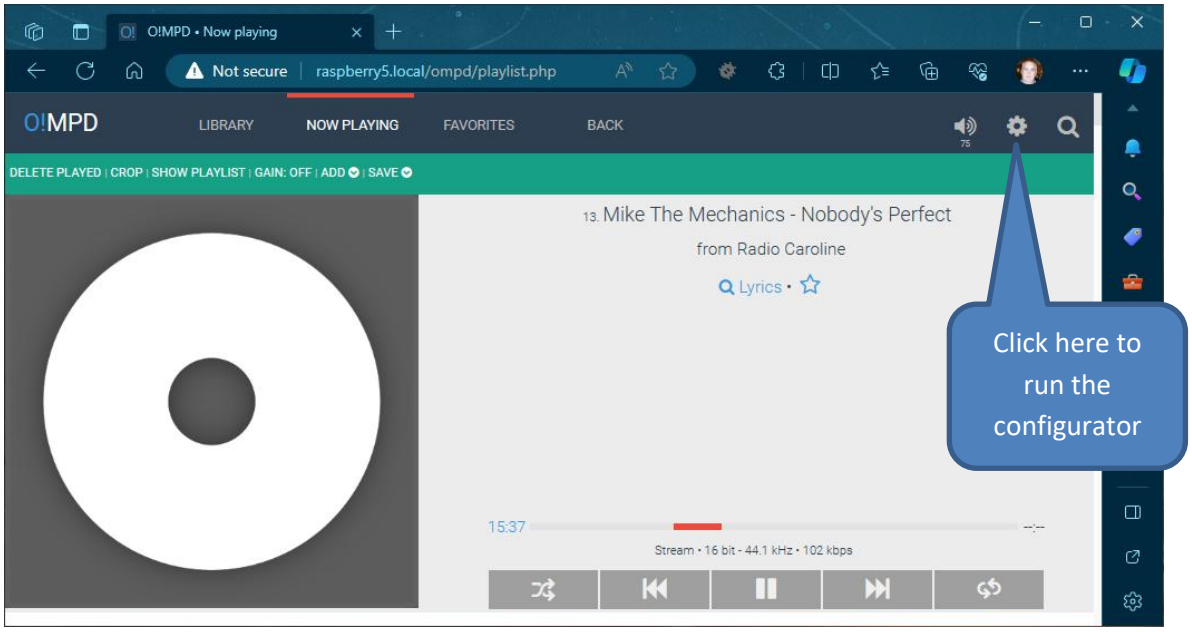
Note: The "**(i) Not secure**" message in the above screens can be ignored. It is because the web pages do not have a security certificate to verify that the pages come from a legitimate source and also are not using the HTTPS protocol. Since these pages are not coming from a source on the Internet and are locally installed, they do not pose a security risk and come from a reliable source namely the **Bob Rathbone radioweb** package.

Setting up the Music directory

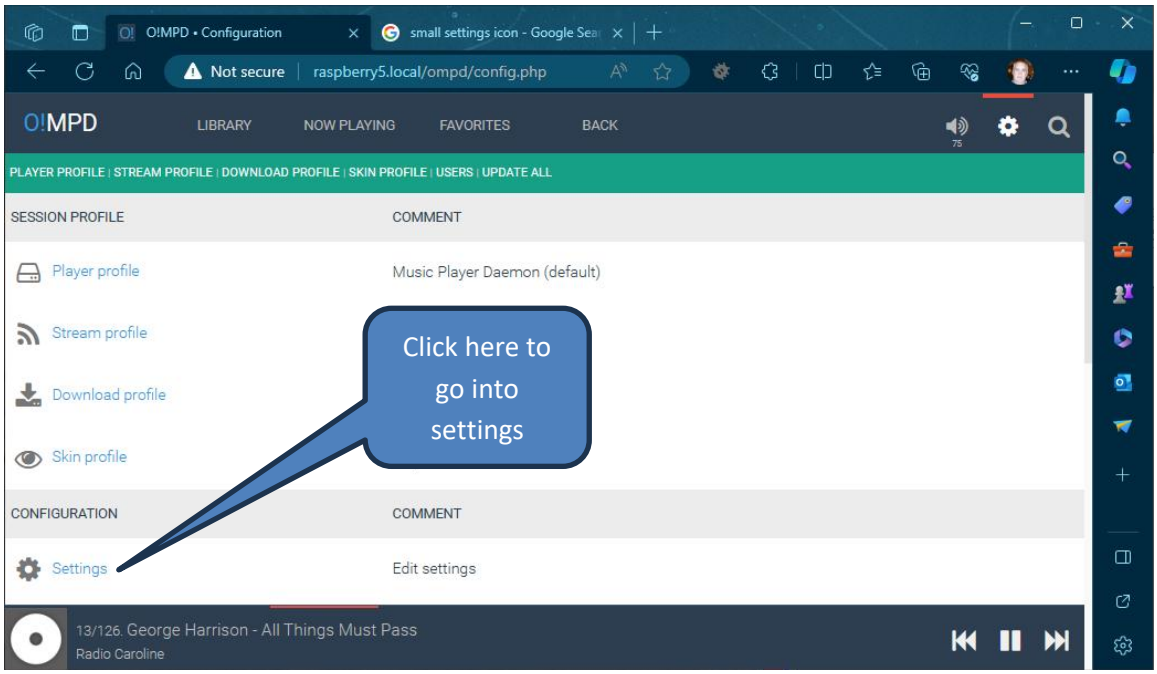
Before you can access music in the media libraries (US Stick, SDcard and remote shares) you need to tell **O!MPD** where the music is stored (or at least check it). MPD by default stores its music in **/var/lib/mpd/music**. The radio software adds soft links to the actual locations of the media files as shown in the example below.

```
$ ls -la /var/lib/mpd/music
total 8
drwxr-xr-x 2 root root  4096 Mar 15 07:32 .
drwxr-xr-x 5 mpd  audio 4096 Mar 15 07:42 ..
lrwxrwxrwx 1 root root     9 Mar 15 07:32 media -> /media/pi
lrwxrwxrwx 1 root root    16 Mar 15 07:20 sdcard -> /usr/share/Music
lrwxrwxrwx 1 root root     6 Mar 15 07:32 share -> /share
```

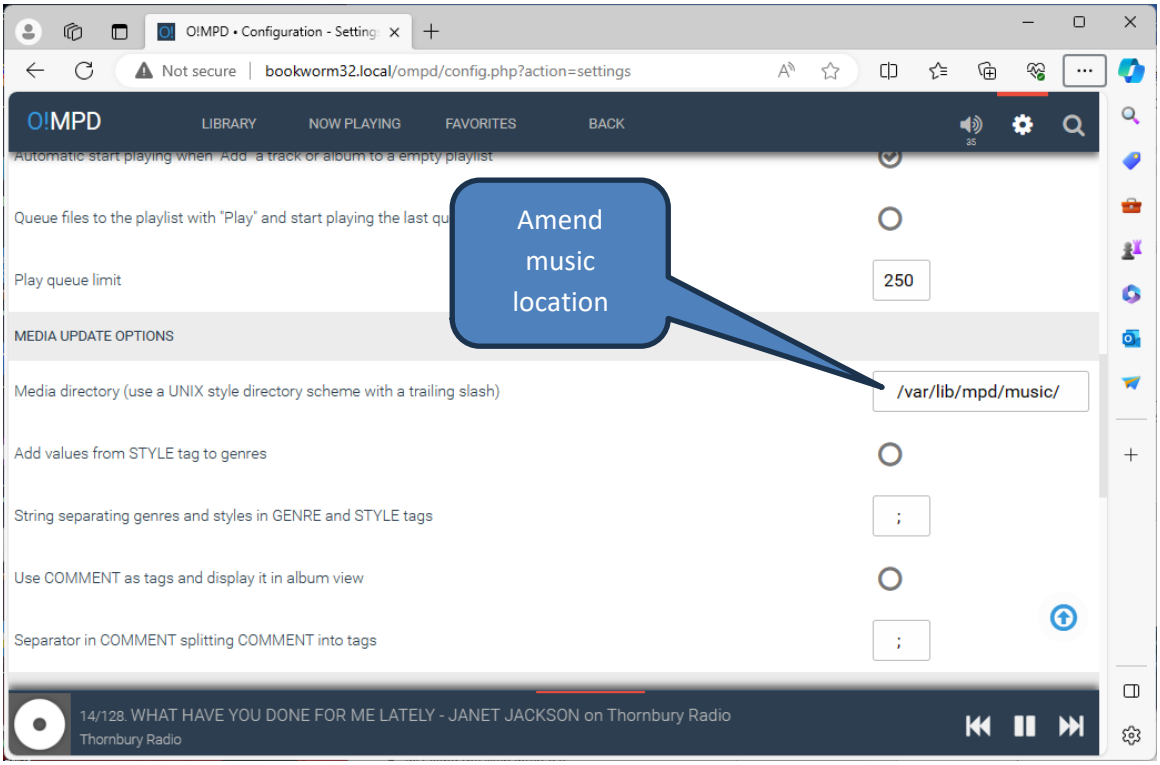
O!MPD needs to be told where the music media is stored. This is now pre-configured but you should check it. Click on the settings icon as shown in the next screen.



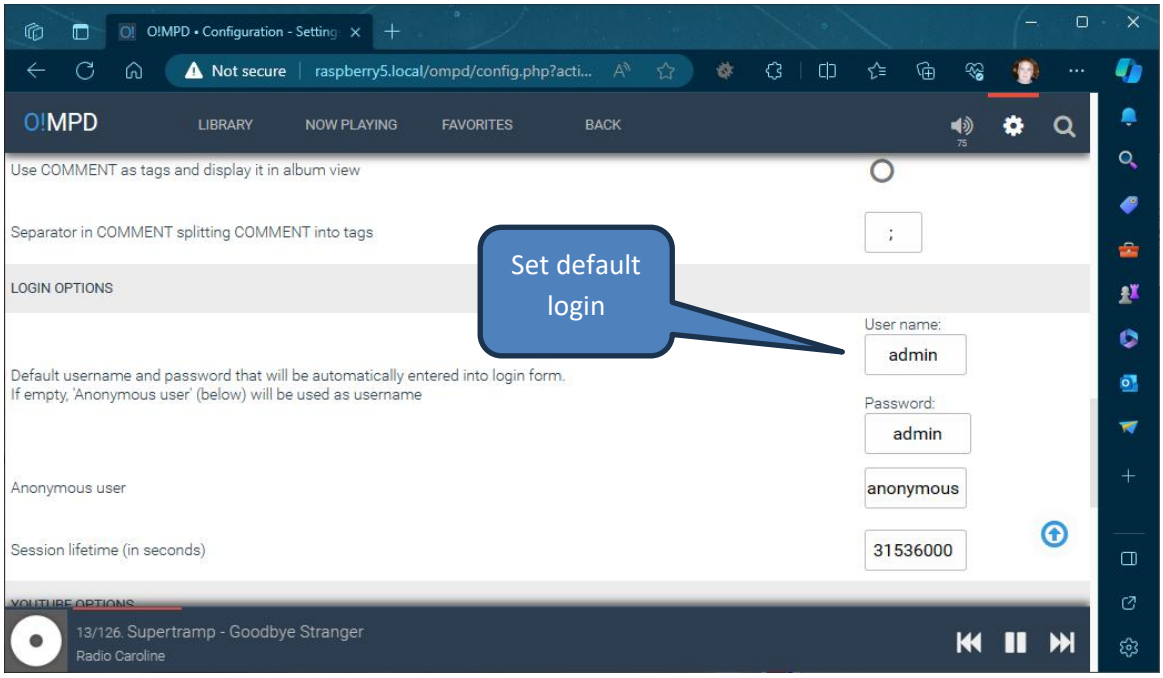
Now click on the CONFIGURATION → Settings Icon.



Once the configuration screen opens, scroll down to MEDIA UPDATE OPTIONS as shown in the next screen.



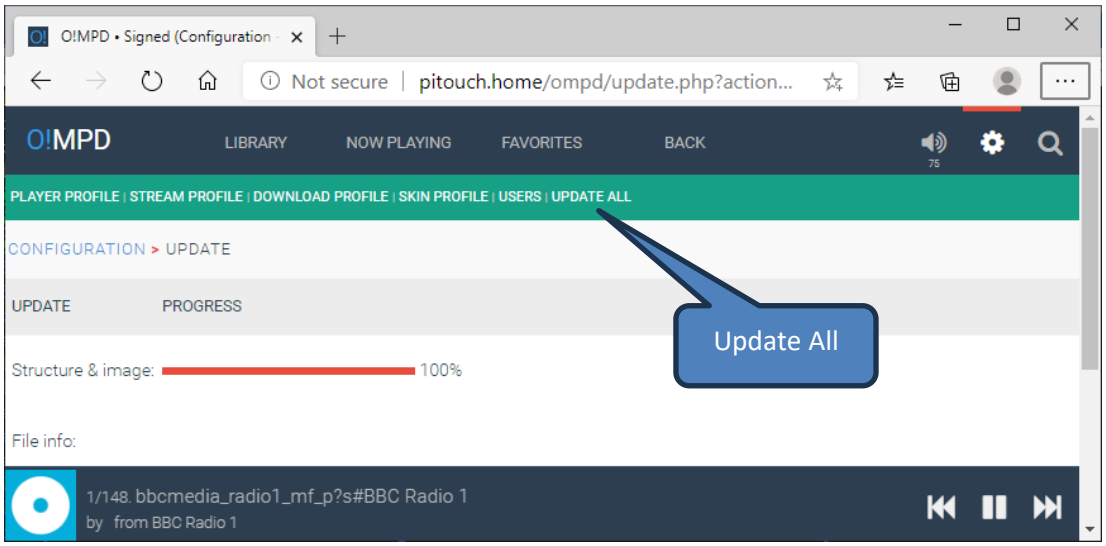
If missing, enter **/var/lib/mpd/music/** into the location box as shown above. Do not forget to add the trailing **/** character to the end of the name.



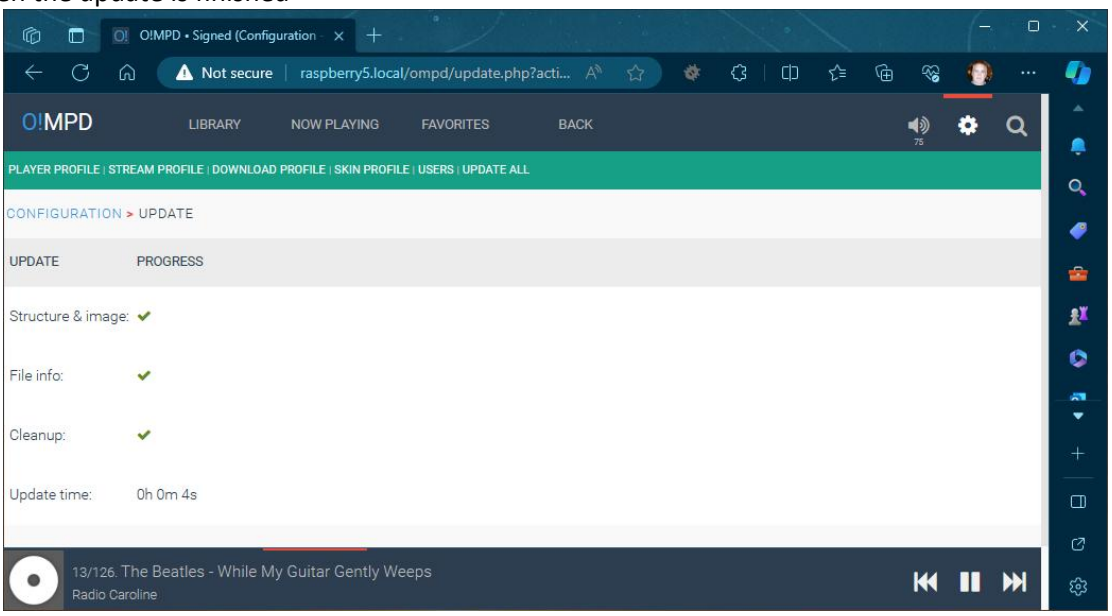
You can also set the default LOGIN OPTIONS on the same screen. Once done you will automatically be logged in as **admin** password **admin**.

Now scroll down to the to the SAVE SETTINGS button at the bottom of the page and click on it.

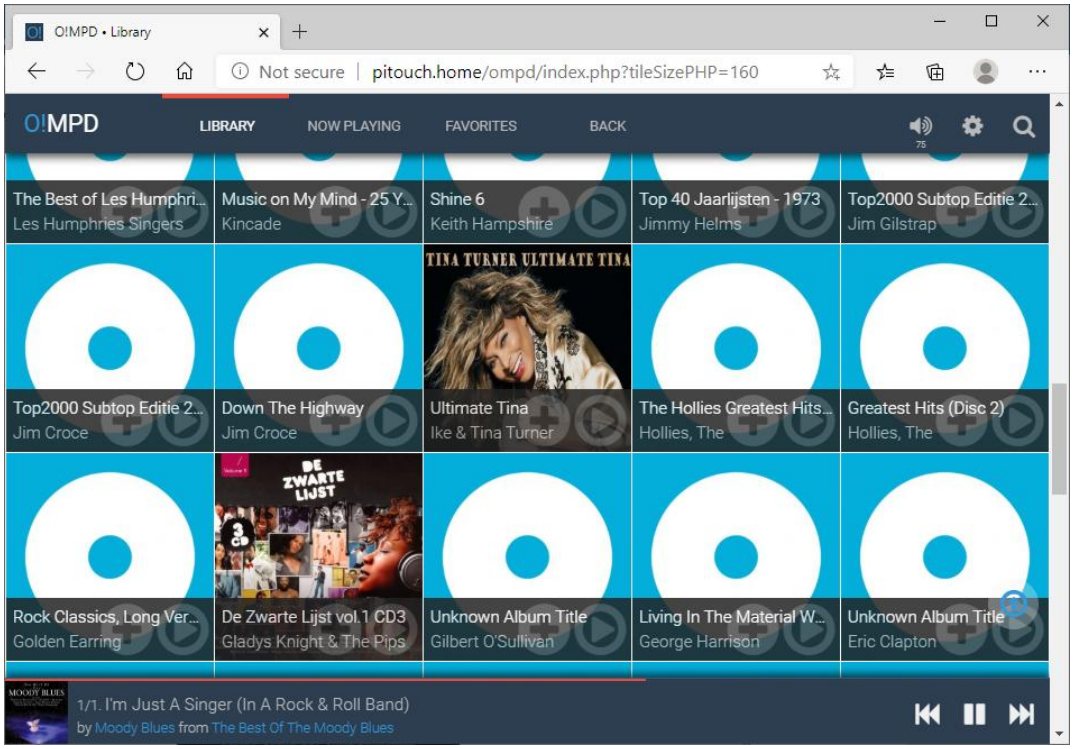
Finally scroll back up to the top of the screen and click on the UPDATE ALL menu item. The media library will now be updated:



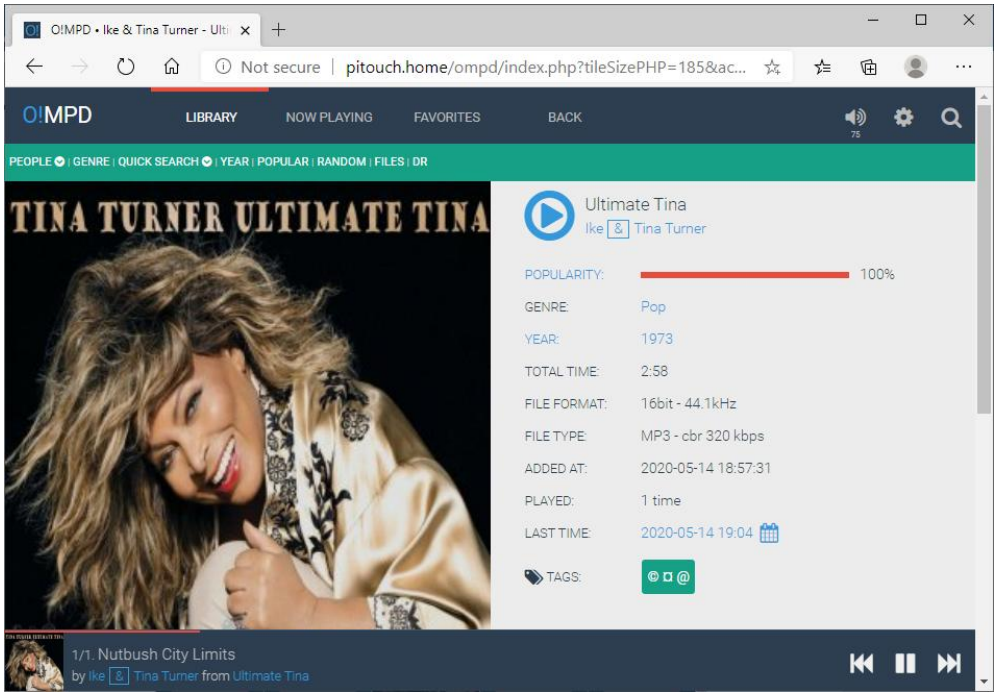
When the update is finished



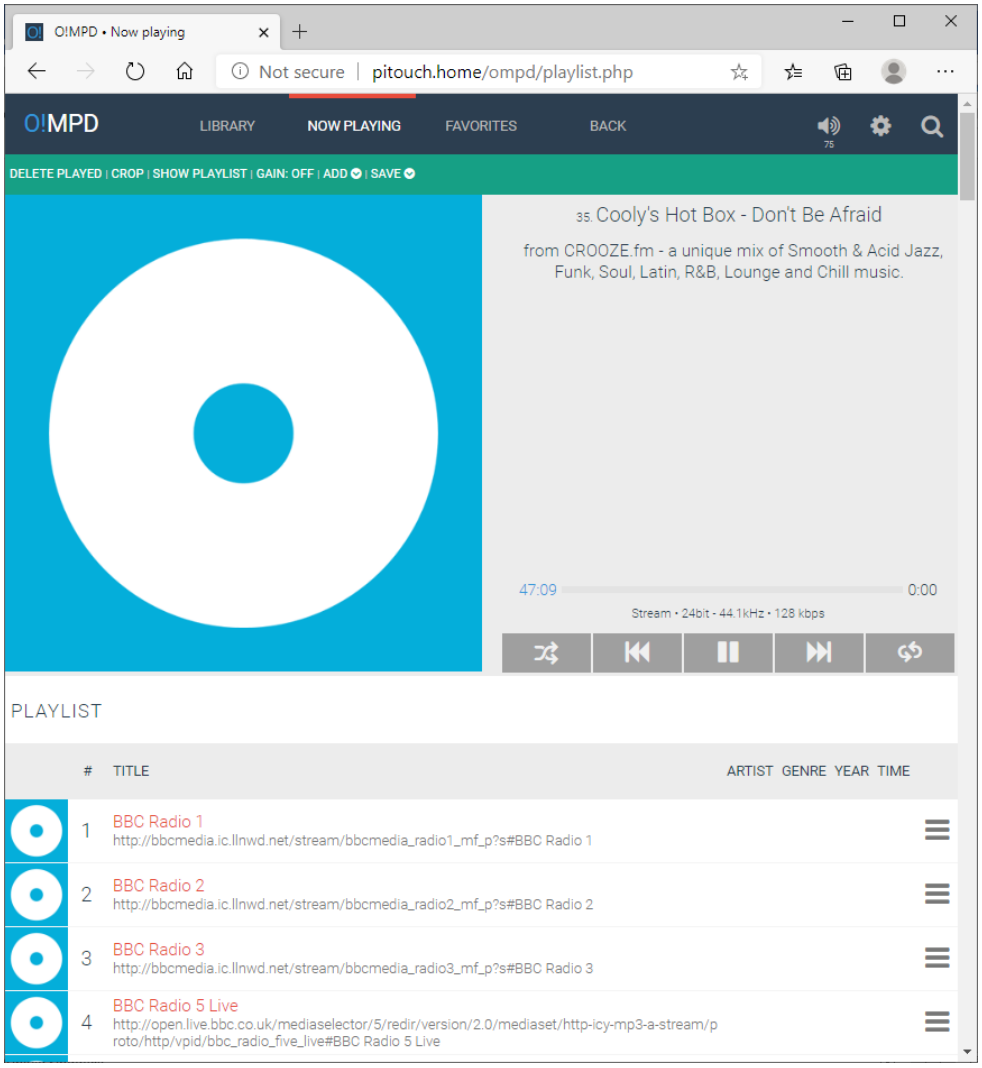
The Web interface will now display the contents of the media library.



Clicking on one of the tracks will display its details along with the album artwork.



Clicking on the blue play icon will play the track. To play a radio station click on the “NOW PLAYING" top menu item. Select the required radio station to start playing the stream.



Changing the Web Interface Radio photo

If you want to change the photo displayed by the Web interface on the first page, then replace the **jpeg** photo file at **/var/www/html/images/radio.jpg**. Try to adjust the size on disk to about 50K using a suitable photo editor such as Photo Shop. Copy the new jpeg photo to the pi home directory with any ftp program.

Now copy it to the Web pages image directory using **sudo**.

```
$ sudo cp radio.jpg /var/www/html/images/.
```

If the new image looks stretched then it may also be necessary to change image proportions in the <img..> statement in **/var/www/html/index.html** (v2.3) or **/var/www/html/index.html** (v3.x). Find the following line in the index file and adjust the width/height values to display the photo with the correct proportions.

```
<img border="0" src="images/radio.jpg" width="780" height="580"></td>
```

Airplay (shairport-sync) Installation

If you have not already done so, carry out a system and firmware upgrade, as shown in *Preparing the Operating System* on page 95. To install Airplay which uses the **shairport-sync** package carry out the following.

- Run the **radio-config** utility.
- Select option **8 Install/configure drivers and software components** from the **radio-config** menu shown in *Figure 193 The radio-config menu* on page 140.
- Select option **8 Install Airplay (shaireport-sync)** from the **radio-config** menu shown in *Figure 193 The radio-config menu* on page 140. The following menu will be displayed.

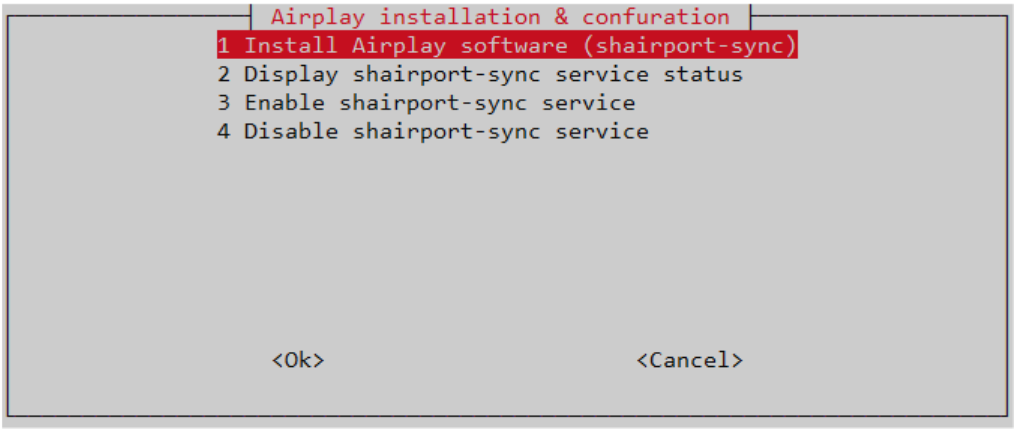


Figure 201 Airplay (shairport-sync) installation menu

Select option 1 Install Airplay software (shairport-sync) to install the Airplay software.

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 177. LCD versions of the radio are described here.

Figure 202 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 203 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”.

Install Icecast streaming

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 *Intellectual Property, Copyright, and Streaming Media* on page 224.

To install Icecast streaming:

- Run the **radio-config** utility.
- Select option **8 Install/configure drivers and software components** from the **radio-config** menu shown in *Figure 193 The radio-config menu* on page 140.
- Select option **4 Install Icecast** from the **radio-config** menu shown in *Figure 193 The radio-config menu* on page 140.

```
Starting Icecast2 integration with the Music Player Daemon
The Icecast2 installation progra 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:

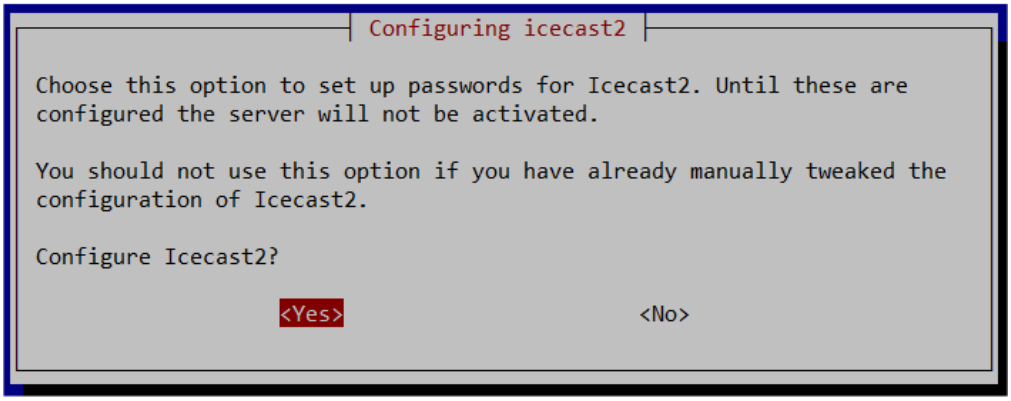


Figure 204 Configuring Icecast2

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
```

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

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.

- To install and configure **espeak** run **radio-config** from the command line
- Select option **8 Install/configure drivers and software components**
- Then select option **5 Install Speech facility**

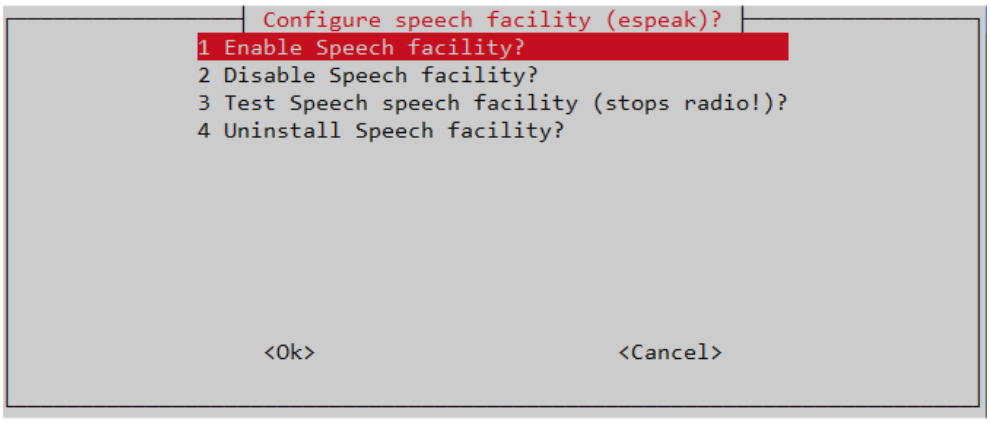


Figure 205: Enabling the speech facility

Select option **1 Enable Speech facility?** to install and enable the **espeak** software.
Once the **espeak** software has been installed it can be tested using option **3 Test Speech speech facility**. When option 3 is selected you should hear a voice saying 'Speech facility enabled'.

You can disable the speech facility without uninstalling the software by selecting option **2 Disable Speech facility**. For further information on **espeak** see https://espeak.sourceforge.net/

Install Luma OLED/TFT driver:

To install the Luma TFT drivers:

- Run the **radio-config** utility.
- Select option **8 Install/configure drivers and software components** from the **radio-config** menu shown in *Figure 193 The radio-config menu* on page 140.
- Select option **7 Install Luma OLED/TFT driver** from the **radio-config** menu shown in *Figure 193 The radio-config menu* on page 140.

On confirmation the following will be displayed

```
/usr/share/radio/scripts/install_luma.sh configuration log, Thu Nov 28
12:06:35 GMT 2024
Using /usr/share/radio
Reading package lists...
Building dependency tree...
Reading state information...
:
A log of these changes has been written to
/usr/share/radio/logs/install_luma.log
```

This completes the installation of the Luma drivers.

Install recording utility (streamripper)

From version 8.0 it is possible to record Radio stations which are transmitted using Shout cast or Icecast format. To do this it is necessary to install the **streamripper** package and to select the **GPIO** setting for the **Record** button.

To install the Radio station recording utility (streamripper):

- Run the **radio-config** utility.
- Select option **8 Install/configure drivers and software components** from the **radio-config** menu shown in *Figure 193 The radio-config menu* on page 140.
- Select option **7 Install recording utility (streamripper)** from the **radio-config** menu shown in *Figure 193 The radio-config menu* on page 140.

After confirmation the following screen will be displayed:

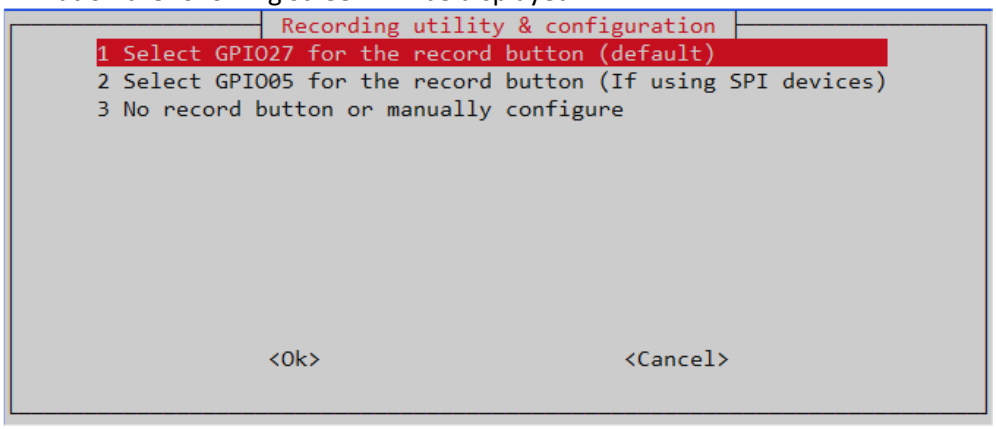


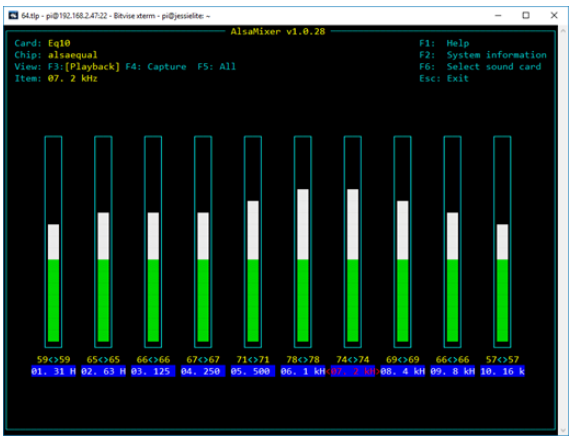
Figure 206 Installing the station recording utility (streamripper)

The radio design now includes an additional 6th button for recording. Normally this uses the **GPIO27** pin (physical pin 13). Select this when using LEDs, OLEDs or I2C displays. If using any displays that use the SPI interface such as the **Waveshare 2.4"** or **1.5" TFTs** using the **SPI** interface select option 2 **GPIO5** (physical pin 29). If you do not want to configure a Record button or wish to configure this manually later, select option 3. Confirm your selection. The installation program will then install **streamripper** and configure the **record_switch** parameter to use the selected GPIO setting.

```
Installing streamripper recording software
sudo apt-get install streamripper
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
:
0 upgraded, 1 newly installed, 0 to remove and 56 not upgraded.
Configured record_switch in /etc/radiod.conf
record_switch=27
End of installation. Press enter to continue:
```

Restart the radio to implement the changes. When either the Record button on the radio or IR remote control are pressed, these call the **record.py** program which starts the recording process. At the end of recording the **record.py** program creates the **Recordings** playlist which can be accessed via the Radio menu.

Install the Alsa Mixer software



The Alsa mixer (**alsamixer**) only runs in an **X-Windows** desktop. It is normally launched from the **gradio** or **vgradio** radio programs. It allows individual adjustment of the audio frequencies.

It cannot currently be used with Bluetooth audio devices nor with the Waveshare WM8960 audio DAC as this requires its own special **/etc/asound.conf** file.

To install the Alsa mixer software (alsamixer):

- Run the **radio-config** utility.
- Select option **8 Install/configure drivers and software components** from the **radio-config** menu shown in *Figure 193 The radio-config menu* on page 140.
- Select option **8 Install Alsa equalizer software** from the **radio-config** menu shown in *Figure 193 The radio-config menu* on page 140.

Chapter 8 – Configuration

Configuration utility radio-config

This section covers manual configuration of the radio. 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.

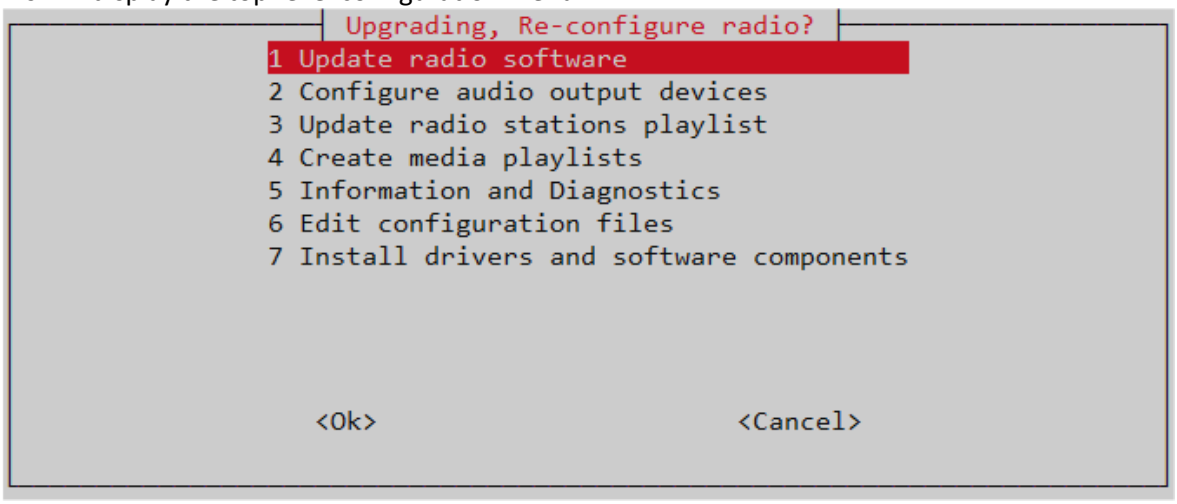
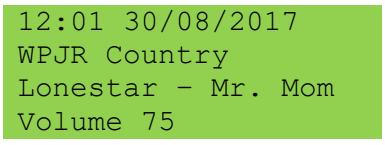


Figure 207 Radio configuration menu

Configuring the volume display

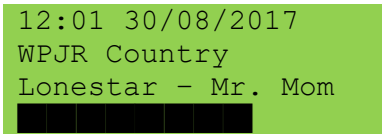
The volume can be displayed as either text or as a series of blocks. This is configured in **/etc/radiod.conf** using the **volume_display** parameter. The default is text.

```
# Volume display text or blocks
volume_display=text
```



To display the volume as a series of blocks change this to 'blocks':

```
volume_display=blocks
```



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.

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 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 FLIRC USB dongle

Note: This configuration procedure is only for the HDMI or Touchscreen display of the radio. If using an LCD display see the Technical Reference manual..

First of all, install the **FLIRC** software as shown in the section called *Installing the FLIRC USB remote control* on page 69.

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 163. The first time you run this program it may ask you if you want to upgrade the firmware. Always upgrade the firmware:

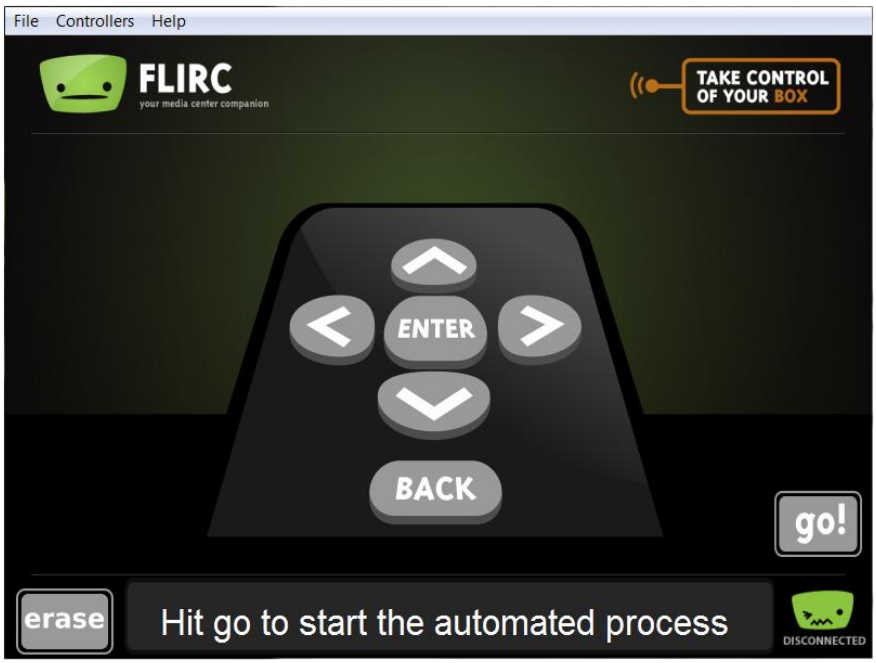


Figure 208 FLIRC setup program

On the **Controllers** drop down menu select the **Full Keyboard** controller.

Figure 209 FLIRC keyboard controller

Now map the buttons on the remote control to the keys shown in Table 19 Graphic screen keyboard command on page 179. 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 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 165.

```
language=European
```

There is a detailed explanation of LCD code pages and program settings in the Technical Reference manual.

Chapter 9 – Operation

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 172.

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 start 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 displyed here}
```

To see what version of the software you are running:

```
$ /usr/share/radio/radiod.py version
Version 8.0
```

Top 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 18 on page 171.

Table 17 Push Button Operation

	Volume buttons		Channel buttons	
LCD Display Mode	Right button	Left button	Up button	Down button
Mode = TIME **Line 1: Time** **Line 2: Station or Track**	Volume Up	Volume Down	Station/Track up	Station/Track down
Mode = SEARCH **If source = RADIO** **Line 1: Search:** **Line2: Radio Station**	Volume Up	Volume Down	Scroll up radio station	Scroll down radio station
Mode = SEARCH **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
Mode = SOURCE **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
Mode = OPTIONS **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:
Mode = RSS (1) **Line 1: Time** **Line 2: RSS feed**	Volume Up Mute	Volume Down Mute	Station/Track up	Station/Track down
MODE = IP address **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 18 Rotary Encoder Knob Operation

	Volume knob		Tuner knob	
LCD Display Mode	Clockwise	Anti-clockwise	Clockwise	Anti-clockwise
Mode = TIME **Line 1: Time** **Line 2: Station or Track**	Volume Up	Volume Down	Station/Track up	Station/Track down
Mode = SEARCH **If source = RADIO** **Line 1: Search:** **Line2: Radio Station**	Volume Up	Volume Down	Scroll up radio station	Scroll down radio station
Mode = SEARCH **If source = MUSIC LIBRARY** **Line 1: Search** **Line2: MusicTrack/Artist**	Scroll up through artists	Scroll down through artists	Scroll up through track	Scroll down through track
Mode = SOURCE **Line 1: Input Source:** **Line2: Radio, Media playlist, Spotify or Airplay**	Volume Up Mute	Volume Down Mute	Cycle up through Airplay, Radio and Media playlists	Cycle down through Airplay, Radio and Media playlists
Mode = OPTIONS **Line 1: Menu Selection** **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.**	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 Time Set, Streaming and Background colour(1)	Cycle through Random, Consume, Repeat, Reload Music, Timer, Alarm , Alarm Time Set, Streaming and Background colour(1)
Mode = RSS (2) **Line 1: Time** **Line 2: RSS feed**	Volume Up	Volume Down	Station/Track up	Station/Track down
MODE = IP address **Line 1: IP address** **Line 2: Station or Track**	Volume Up	Volume Down	Scroll up through track or radio station	Scroll down through track or radio station

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 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.

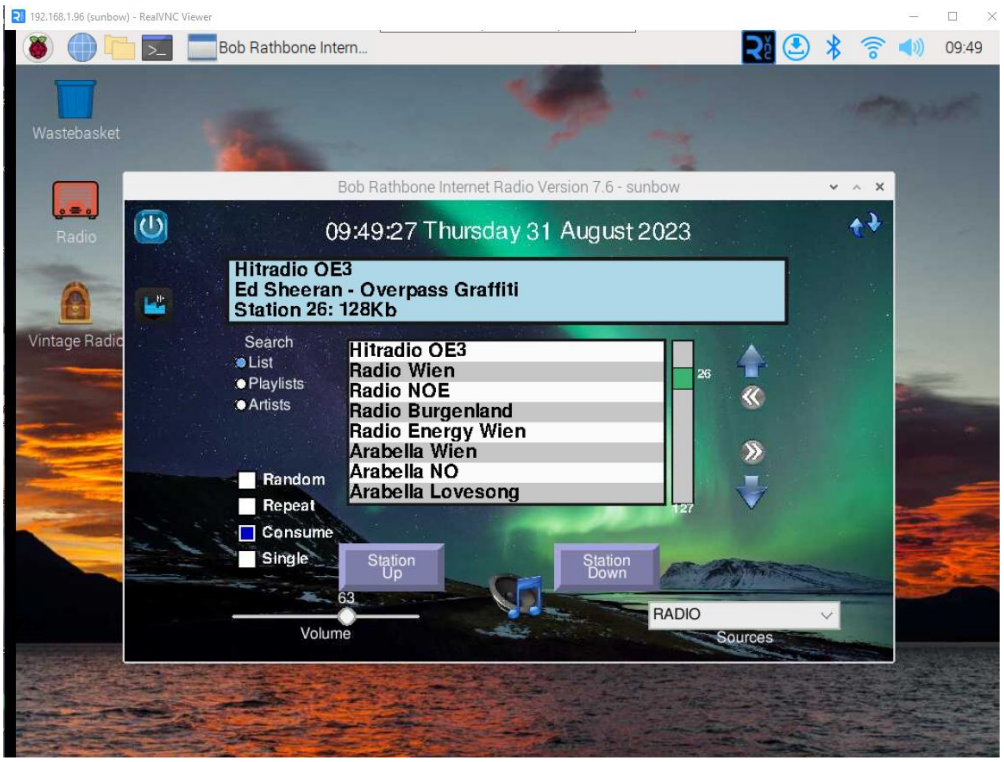
1. Automatically when starting the desktop
2. By clicking on the radio icon on the desktop
3. By manually starting the program from the command line (From a xterminal)

To start the radio from command line run the gradio.py program:

```
$ cd  /usr/share/radio
$ sudo ./gradio.py &
```

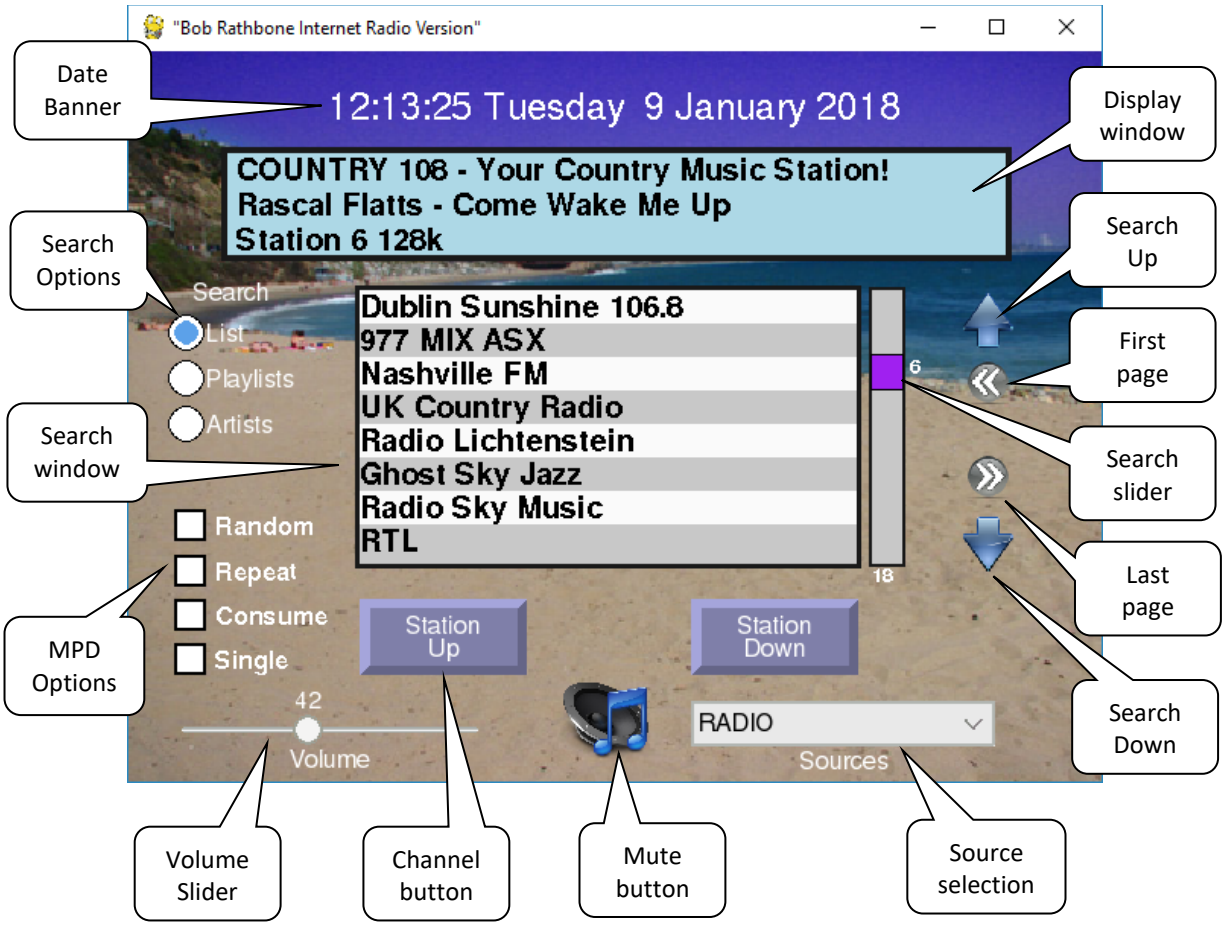
Starting the radio from the desktop. Click the icon shown here on the desktop. 

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 210 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.

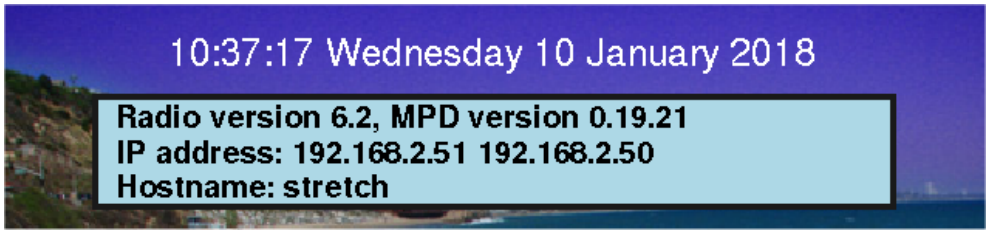


Figure 211 Graphical scree information display

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 212 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 213 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 214 Display playlists

In the following example the USB stick playlist was selected. Once a playlist selected the list is displayed.

Figure 215 Display of media tracks

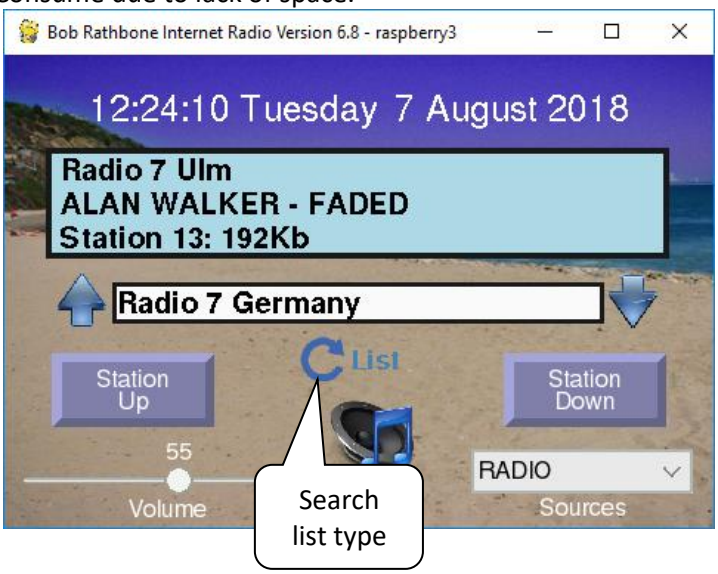
Clicking the Artists radio button displays the list of artists in the search window. Once clicked the search window positions on the first song of that artist's tracks.

Figure 216 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 210 on page 173.

Artwork display

If the music track has artwork and the **ffmpeg** (See *Setting up the locale* on page 100) 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.

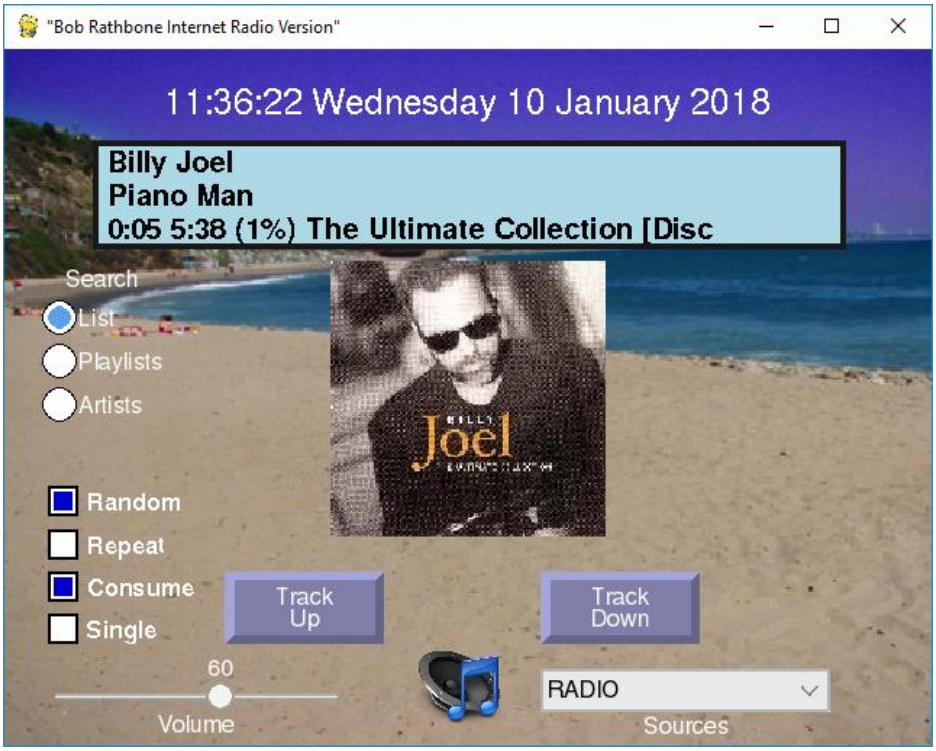
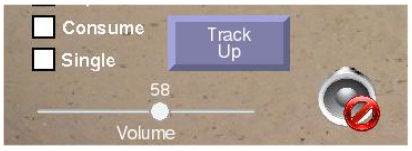


Figure 217 Track artwork display

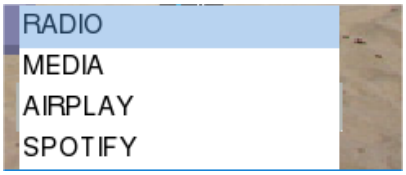
Note that the two grey push buttons now display 'Track Up/Down' instead of 'Station Up/Down'.

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 (*shairport-sync) Installation on page 153 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.

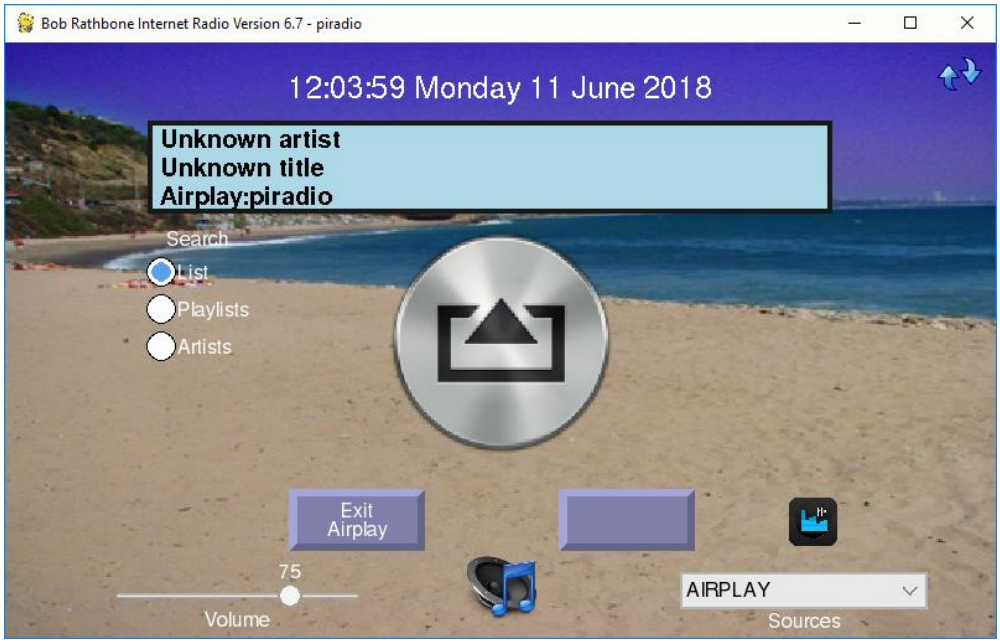


Figure 218 Airplay running on a Graphical screen

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:

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.

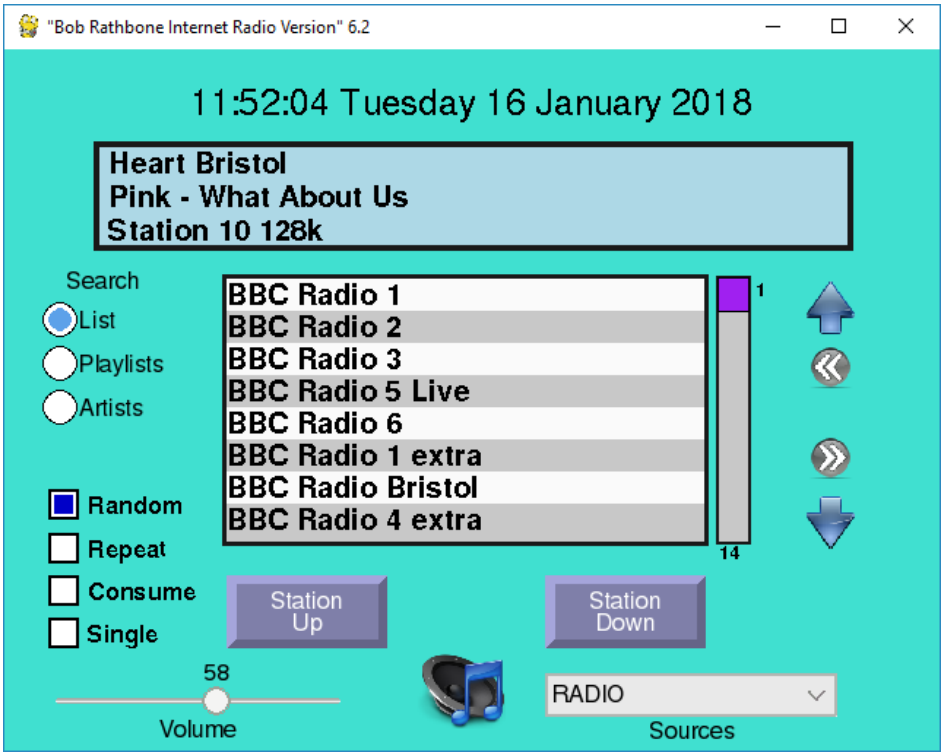


Figure 219 Changing the graphical screen theme

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

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 19 Graphic screen keyboard command

Key	Description	Key	Description
Page Up (PgUp)	Channel/Track Up	**Up Arrow**	Search Up
Page Down (PgDn)	Channel/Track Down	**Down Arrow**	Search Down
+ Key	Volume increase	**Left arrow**	Go to first search page
- Key	Volume decrease	**Right arrow**	Go to last search page
R	Toggle Random	**L**	Select Search List
T	Toggle Repeat	**P**	Select Search Playlists
C	Toggle Consume	**A**	Select Search Artists
S	Toggle Single	**M**	Toggle Mute on/off
D	Cycle display window	**ESC**	Exit program
X	Switch between **vgradio** and **gradio**		

Operating the Alsa Equalizer

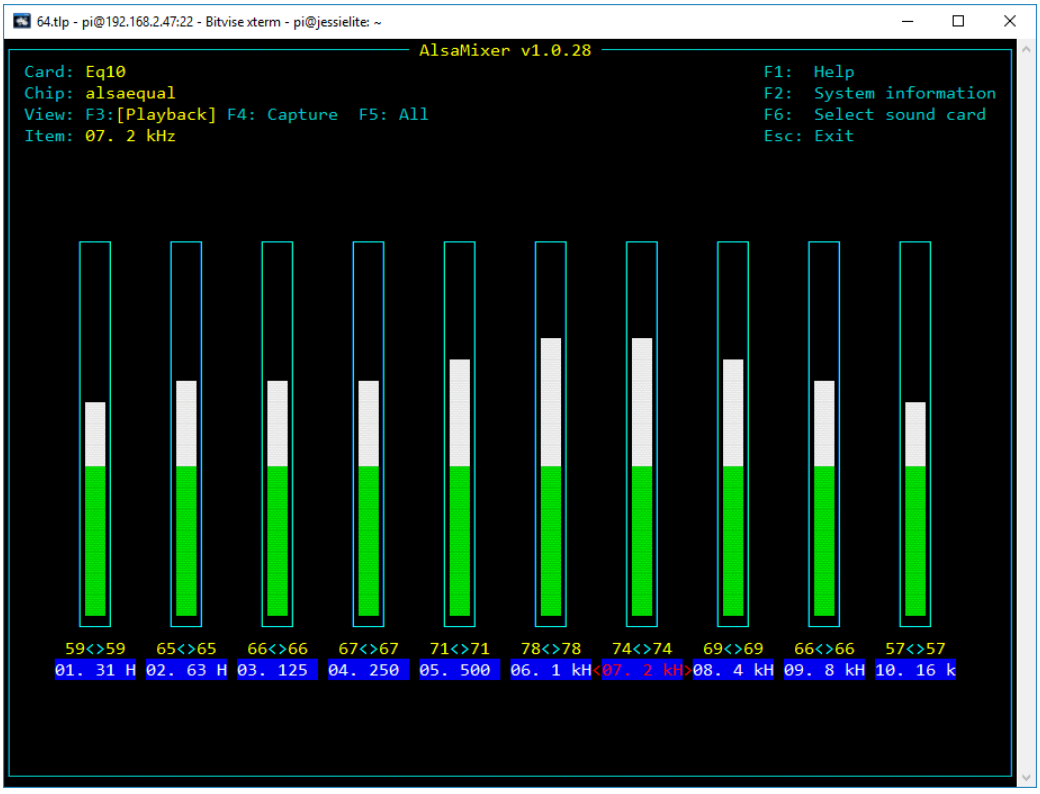


Figure 220 The Alsa

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.

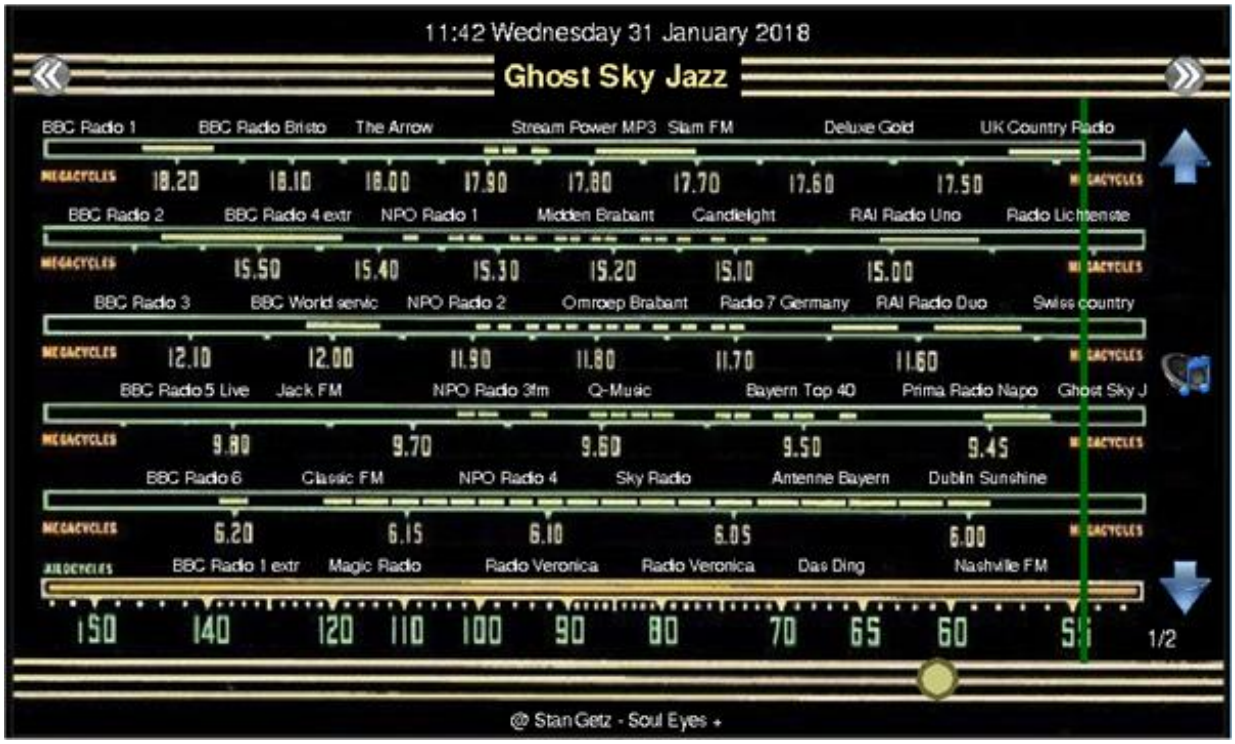


Figure 221 The vintage graphic radio on a touch-screen

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 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 ***Error! Reference source not found.*** on page **Error! Bookmark not defined.**.

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 188 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 188.

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 *204*. Then go to the section on *Creating Media playlists* on page 188

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+rpt1 (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 **Error! Reference source not found.** on p age **Error! Bookmark not defined.**. 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 **Error! Bookmark not defined.**)
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 **gradio** and **vgradio**

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 the 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 THERFORE 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 224.

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 to 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 **systemctl** 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 20 Common MPC commands

MPC command	Description
mpc	Displays status (**mpc status** also does the same)
mpc current	Displays currently playing station or track
mpc next	Play next song
mpc prev	Play previous song
mpc play n	Play station or track where n is the track or station number
mpc volume 75	Set volume to 75%
mpc stop	Stop playing
mpc random <on|off>	Toggle shuffling of songs on or off
mpc repeat <on|off>	Toggle repeating of the playlist

mpc clear	Clear the playlist
mpc consume <on|off>	When playing tracks remove from the playlist once played
mpc playlist	List loaded radio stations or streams
mpc listall	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 Amircan 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, the 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 a sshown 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.

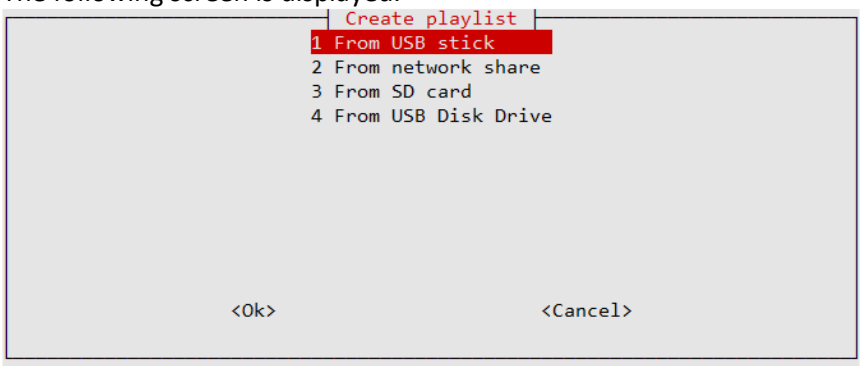
```
filesystem_charset      "LATIN-1"
```

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.

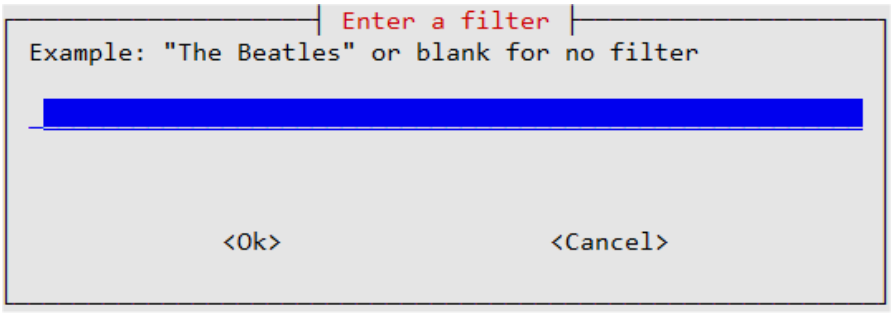
```
$ cd /usr/share/radio
$ sudo ./create_playlist.sh
```

The following screen is displayed.

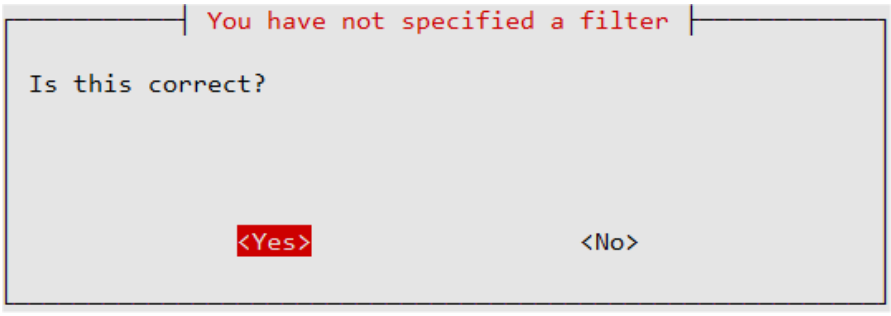


Select option 1 initially.

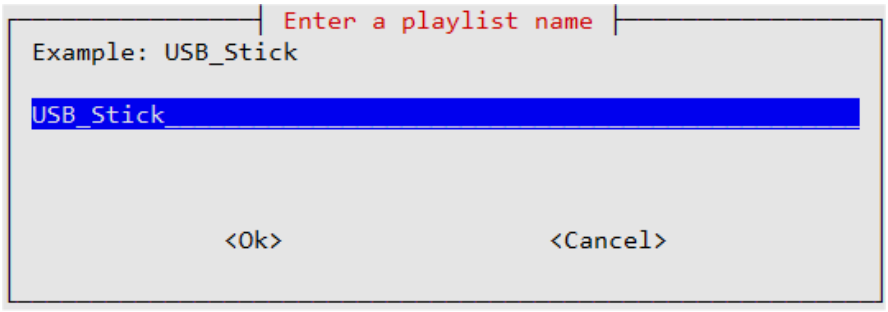
Answer “Yes” to create the playlist from the USB stick.



Enter a filter name or enter for no filter.



Select Yes to continue.



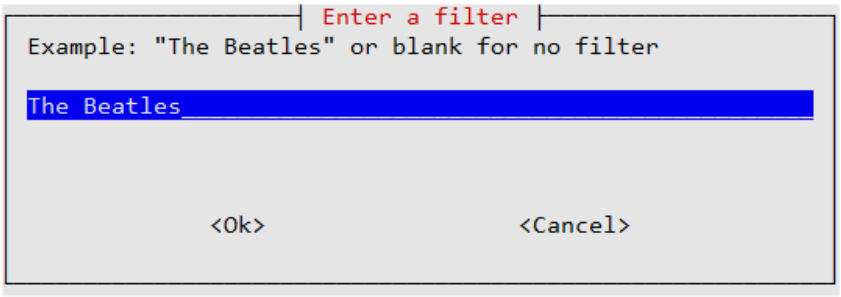
The program will suggest a name for the playlist but you may choose any name (But do not make it too long).

The program continues by creating a playlist called **USB_stick.m3u** in **/var/lib/mpd/playlists**.

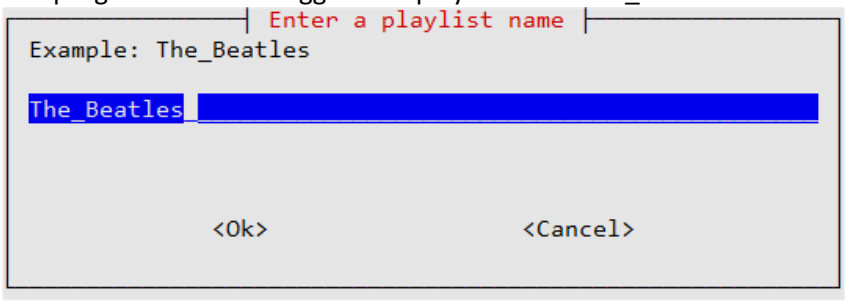
```
sudo service radio stop
sudo service mpd stop
Mounted  /dev/sda1 on /media
cd /var/lib/mpd/music/
/var/lib/mpd/music
sudo find -L media -type f -name *.mp3 -or -name *.ogg -or -name *.flac  >
/tmp/list29073
sudo mv /tmp/list29073 /tmp/USB_Stick
```

```
============================================================
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.

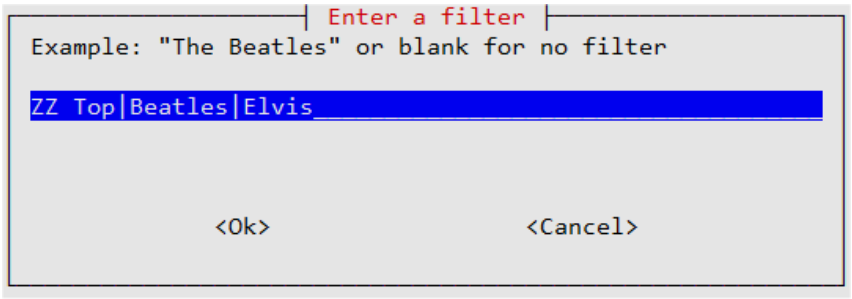


You will note that spaces in the playlist name have been replaced with underscores(_).
This is just for the file name. When the playlist is displayed in the radio program the underscores will be converted back to spaces. The program will now create a playlist with the name **The_Beatles.m3u** (or whatever 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 (|) seperating them,for example:

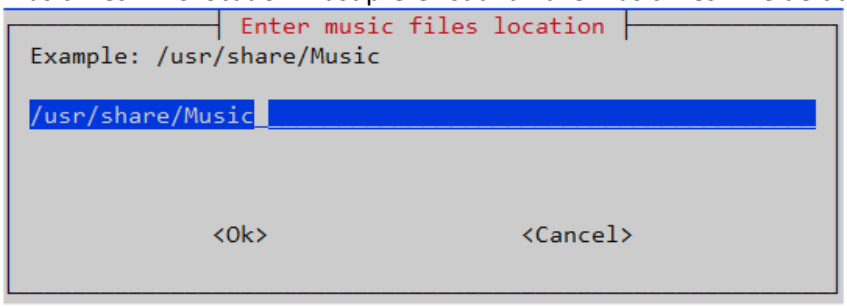


This will filter all songs from ZZ Top, The Beatles and Elvis or any other titles that contain these names. However this may not be want is wanted. Maybe songs by Elvis are wanted and not songs with 'Elvis' in the title. For example *Dire Straits – Calling Elvis*. In such a case use the **/** character 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 you 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 comapatability 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.

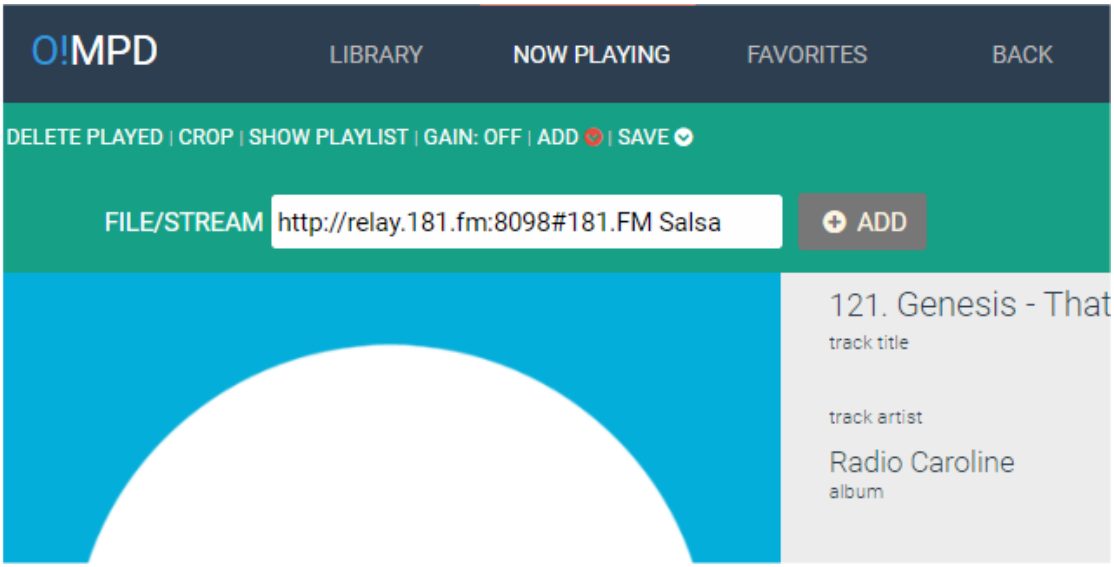


Figure 222 Adding a new station to MPD

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 223 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.

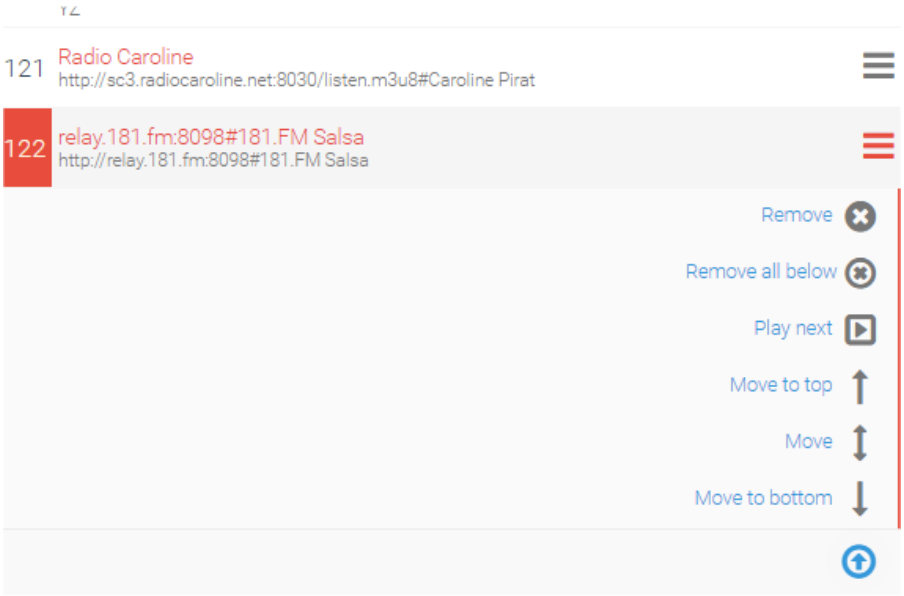


Figure 224 Moving and deleting entries in the Web Interface

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 poreviously 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 disapointed. 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 availble 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=anCLSEDQODrElkxl&limit=50&genr
e=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 225 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 226 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 227 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

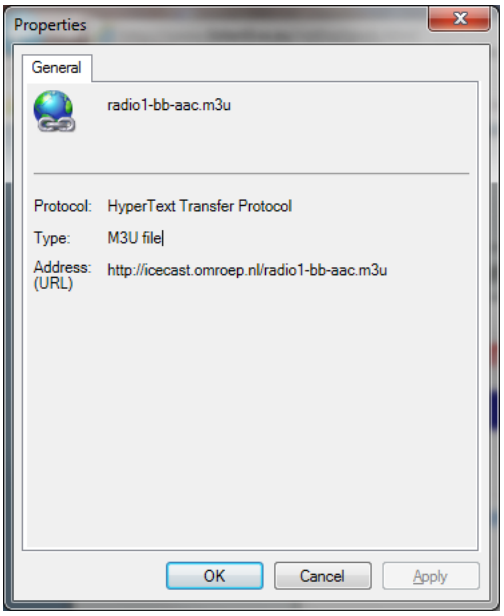
There are a lot of resources on the Internet how to find PLS and M3U files so simply search for "PLS or M3U files" through the search machine of your choice. Below are some good sources of radio streams around the world.

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)

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*n*, Title*n* and Length*n* 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&amp;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

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/

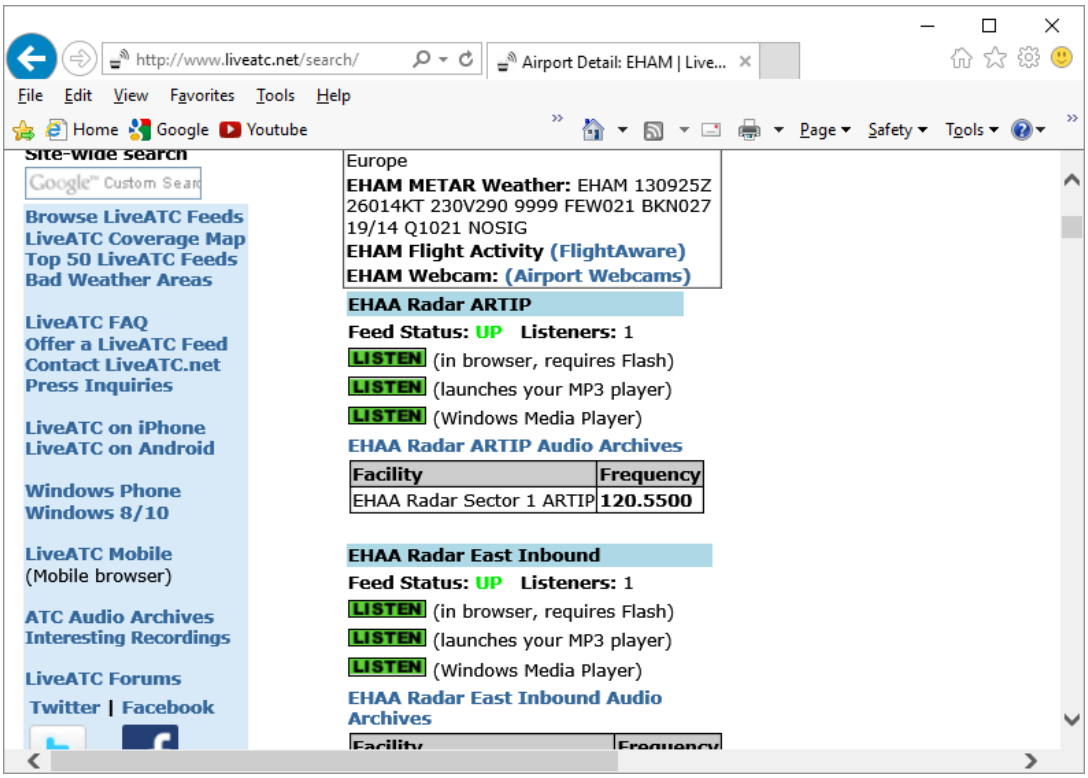


Figure 228 Live ATC Web page

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 229 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.

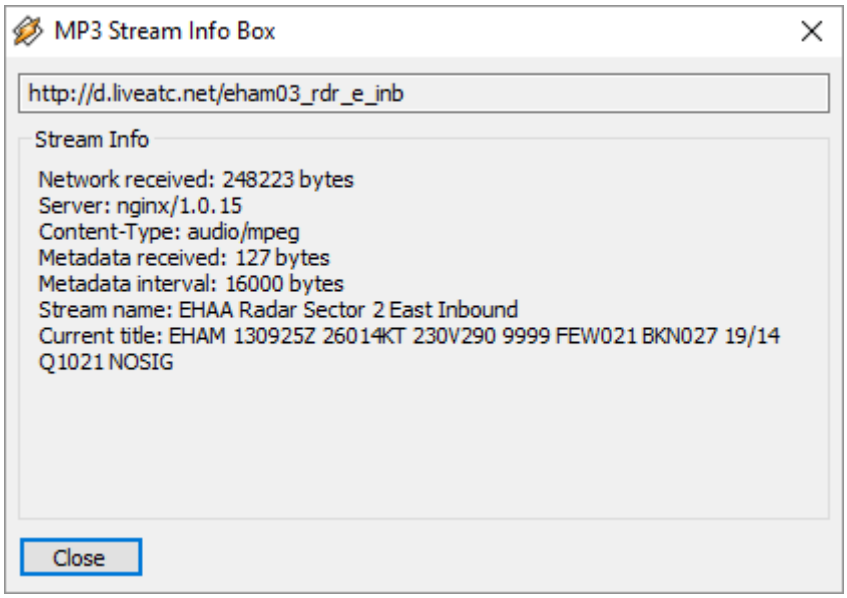


Figure 230 WinAmp station information

The URL for the stream is shown in the top box. **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/eheh2_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:_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	13th 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 205.

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 188.

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 187. 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 ..
lrwxrwxrwx 1 root root     6 Jul  7 12:17 media -> /media
lrwxrwxrwx 1 root root    16 Jul  7 12:37 sdcard -> /home/pi/mymusic
lrwxrwxrwx 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 **Error! Reference source not found.** on **Error! Bookmark not defined.**.

Controlling the Music Player daemon from Mobile devices

Android devices

Figure 231 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-mpd-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.

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 10 -Troubleshooting

General troubleshooting

Most of the diagnostics can be run from the **radio-config** menu.

Note: When selecting diagnostics from the following menu by necessity need to stop the Radio and IR remote control daemons. Make sure that you are not doing anything important such as recording a radio station before you select any diagnostic programs.

Log into the Raspberry Pi and enter **radio-config** on the command line:

```
$ radio-config
```

Most diagnostics will be found in option **5 Diagnostics and Information** of the **radio-config** menu as shown below.

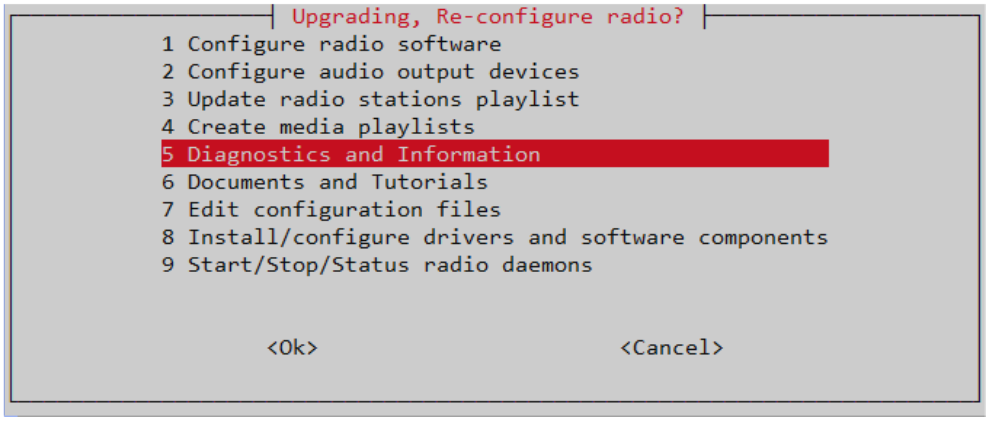


Figure 232 Diagnostics and Information menu

Selecting option 5 will show the following menu:

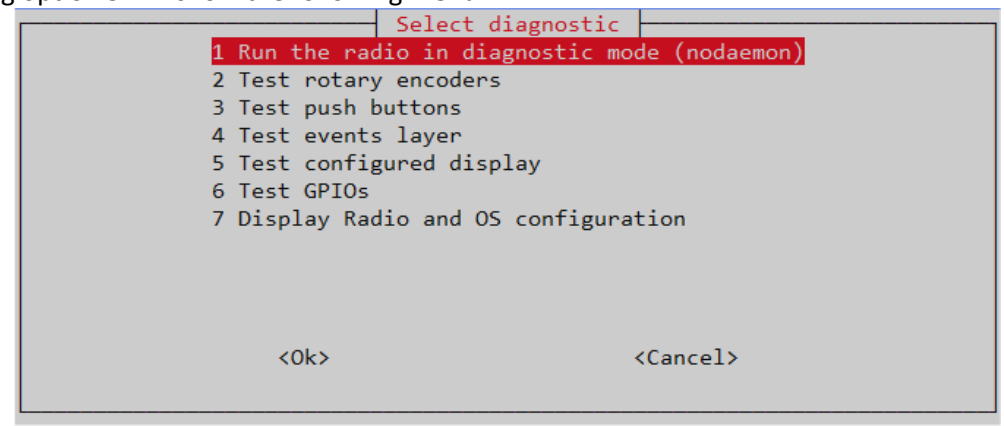


Figure 233 Diagnostics selection menu

Running the radio software in diagnostic mode

The **radiod** program normally runs as a background service (a so-called **daemon**) and is controlled with the **systemctl stop/start** commands. As previously mentioned, the diagnostics, when required, will stop the **radiod daemon** and start these in foreground mode so that you can see what is happening.

To run the radio software in diagnostic mode.

- Select option **1 Run the radio in diagnostic mode (nodaemon)** shown in *Figure 232 Diagnostics and Information menu* on page 210.

The radio program will be started in diagnostic mode.

```
Press Ctrl-C to exit diagnostic mode
radiod.py nodaemon diagnostic log, Fri 13 Dec 12:39:12 GMT 2024
```

If there are any errors they will be displayed immediately under the above message.

To exit and return to the menu press **Ctrl-C**

```
^C
A log of this run has been recorded in
/usr/share/radio/logs/radiod_nodaemon.log
A compressed tar file has been saved in
/usr/share/radio/logs/radiod_nodaemon.log.tar.gz
Send /usr/share/radio/logs/radiod_nodaemon.log.tar.gz to bob@bobrathbone.com
if required
Press enter to continue:
```

Either copy and paste any errors into an email or send the log the above log and send this send to support.

Testing Rotary encoders

To test the rotary encoders

- Select option **2 Test rotary encoders** from the diagnostics menu. shown in *Figure 232 Diagnostics and Information menu* on page 210.

```
Press Ctrl-C to exit diagnostic mode
Test standard 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
Rotary encoder step size = full
KY040 encoder R1 resistor fitted = no
Waiting for events
```

Now operate each rotary encoder in turn including the push-button on each one. Volume and Tuner events should be seen. If not check your wiring against the configured GPIO settings shown above. Press **Ctrl-C** to end test.

```
Tuner event  2 ANTICLOCKWISE
Tuner event  2 ANTICLOCKWISE
Tuner event  2 ANTICLOCKWISE
Tuner event  1 CLOCKWISE
Tuner event  1 CLOCKWISE
Tuner event  1 CLOCKWISE
Tuner event  1 CLOCKWISE
Tuner event  3 BUTTON DOWN
Volume event  1 CLOCKWISE
Volume event  1 CLOCKWISE
Volume event  1 CLOCKWISE
Volume event  2 ANTICLOCKWISE
```

```
Volume event  2 ANTICLOCKWISE
Volume event  3 BUTTON DOWN
```

Test radios which are using buttons

To test the rotary encoders

- Select option **2 Test rotary encoders** from the diagnostics menu shown in *Figure 232 Diagnostics and Information menu* on page 210.

```
Test Button Class or Ctl-C to exit:
Left switch GPIO 14
Right switch GPIO 15
Mute switch GPIO 4
Up switch GPIO 24
Down switch GPIO 23
Menu switch GPIO 17
Pull Up/Down resistors UP
Record switch GPIO 27 Pull Up
```

Press each button in turn including the Record button.

```
Button pressed on GPIO 4
Button pressed on GPIO 4
Button pressed on GPIO 4
Button pressed on GPIO 17
Button pressed on GPIO 17
Button pressed on GPIO 27
Button pressed on GPIO 27
Button pressed on GPIO 14
Button pressed on GPIO 14
Button pressed on GPIO 15
Button pressed on GPIO 15
Button pressed on GPIO 24
Button pressed on GPIO 23
Button pressed on GPIO 23
```

Test events layer

All events from lower-level drivers such as the Rotary encoder and Button classes are sent up to the event class. This formats all events into a common set of events which are then passed up to various top level radio programs (radiod, gradio and vgradio).

- Select option **5 Test configured display** from the diagnostics menu shown in *Figure 232 Diagnostics and Information menu* on page 210. This displays the following output

```
Press Ctrl-C to exit diagnostic mode
Waiting for events:
```

Note: If the lower-level Rotary encoder or button tests are not working then there are no events to be seen in the Event layer. Also note that the numbers displayed are Event numbers and not GPIO numbers.

The following output is typical from running this test whilst operating Rotary encoders and the Record switch.

```
Event 6 DOWN_SWITCH
Event 5 UP_SWITCH
Event 2 LEFT_SWITCH
Event 1 RIGHT_SWITCH
Event 3 MUTE_BUTTON_DOWN
Event 7 MENU_BUTTON_DOWN
Event 22 RECORD_BUTTON
```

Note: Also note that events from the IR Remote Control do not use the **event_class** so will not be seen by the event layer. IR events are sent over the network directly to the Radio Class.

Test configured display

To test the configured display:

- Select option **5 Test configured display** from the diagnostics menu shown in *Figure 232 Diagnostics and Information menu* on page 210.

Note: That the success of this test relies upon the display being correctly configured. It is only used for testing **LCDs**, **TFTs** and **OLED** display screens. This test is not relevant for Graphical displays which run in a Windows Desktop.

```
Test display_class.py ESC to exit:
Screen LCD Lines=4 Character width=20
Display type 1 LCD
bg_color 7 White

Enter to continue or ESC to exit:
```

Testing GPIO inputs

The Rotary encoder in particular may not be working but it may not be obvious what the problem is. The GPIOs test program may be useful to locate the problem.

- Select option **6 Test GPIOs** from the diagnostics menu shown in *Figure 232 Diagnostics and Information menu* on page 210.

```
Press Ctrl-C to exit diagnostic mode
Test Rotary encoders and buttons
GPIO: 2 State:High
GPIO: 3 State:High
GPIO: 4 State:High
GPIO: 5 State:High
GPIO 5 rising
GPIO: 6 State:High
GPIO 6 rising
GPIO: 7 State:High
GPIO: 8 State:High
GPIO 8 rising
GPIO: 9 State:High
GPIO: 10 State:High
GPIO: 11 State:High
GPIO: 12 State:High
GPIO: 13 State:High
GPIO 13 rising
```

```
GPIO: 14 State:High
GPIO: 15 State:High
Error: GPIO 16 'GPIO busy'
Check conflict with GPIO 16 in other programs or in /boot/config.txt
GPIO: 17 State:High
GPIO: 18 State:High
GPIO: 19 State:High
GPIO 19 rising
GPIO: 20 State:Low
GPIO: 21 State:High
GPIO 21 rising
GPIO: 22 State:High
GPIO 22 rising
GPIO: 23 State:High
GPIO: 24 State:High
Error: GPIO 25 'GPIO busy'
Check conflict with GPIO 25 in other programs or in /boot/config.txt
GPIO: 26 State:High
GPIO 26 rising
GPIO: 27 State:High
Waiting for input events:
```

At first it looks like GPIO 16 and 25 have a problem but this is simply because these two GPIOs are being used by the IR Remote Control program (ireventd.py) so can be ignored. Now test the problem Rotary encoder or push button that you are having problems with. In this example when you turn the volume Rotary Encoder you see that GPIO14 and GPIO18 are active.

```
Waiting for input events:

GPIO 15 falling
GPIO 18 falling
GPIO 15 rising
GPIO 15 rising
GPIO 15 rising
GPIO 18 rising
```

However, when you check against the results from the Rotary encoder test in *Testing Rotary encoders* on page 211 (See below) you see that it is configured for GPIO14 and GPIO15 and not GPIO15 and GPIO18 and so is incorrectly wired. Correct the wiring to clear the fault.

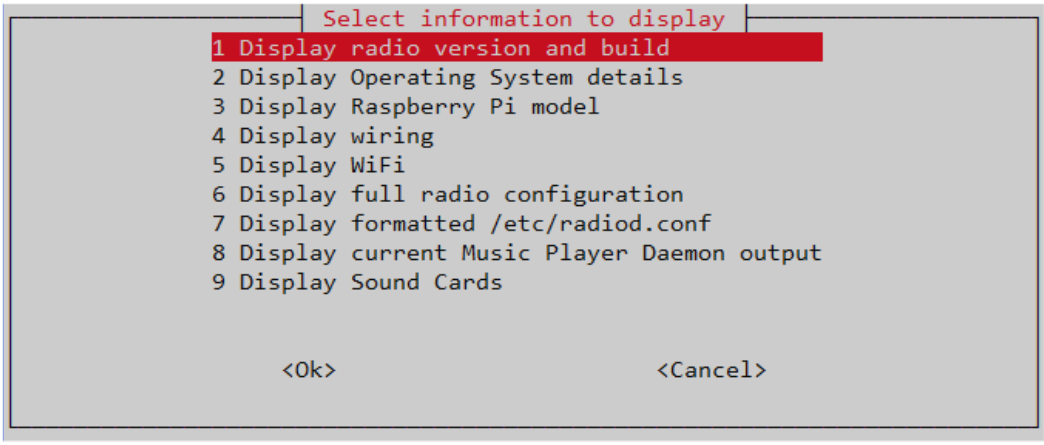
```
Press Ctrl-C to exit diagnostic mode
Test standard rotary encoder Class
Left switch GPIO 14
Right switch GPIO 15
```

The switch settings for Rotary encoders and Push buttons are configured in the **/etc/radiod.conf** configuration file.

```
# 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
```

Displaying the Radio and Operating System configuration

- Select option 7 Display Radio and OS configuration from the diagnostics menu shown in *Figure 232 Diagnostics and Information menu* on page 210.



The following table shows the details of each available report.

	Option and description
1.	**Display radio version and build** Version number and build, for example Version 8.0 Build 8.0.1
2.	**Display Operating System details** Displays OS (Bookworm or Bullseye) details
3.	**Display Raspberry Pi model** Raspberry Pi model details eg. b03115: Model 4B, Revision 1.5, RAM: 2GB MB etc.
4.	**Display wiring** This displays how the wiring for displays, buttons etc. have been configured
5.	**Display Wi-Fi** Wi-Fi network information such as SSID, channel and signal strength
6.	**Display full radio configuration** Detailed display of all configuration details (OS and Radio) This creates the file **/usr/share/radio/logs/config.log.tar.gz** to be sent to support
7.	**Display formatted /etc/radiod.conf** Easy to read details of the Radio configuration file
8.	**Display current Music Player Daemon output** This displays the details of the currently playing radio station or track from MPD
9.	**Display Sound Cards** Displays all configured sound cards (DACs, onboard jack etc.)

Chapter 11 - Setting up Spotify

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 *Update to the latest the system* packages on page 95.
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:

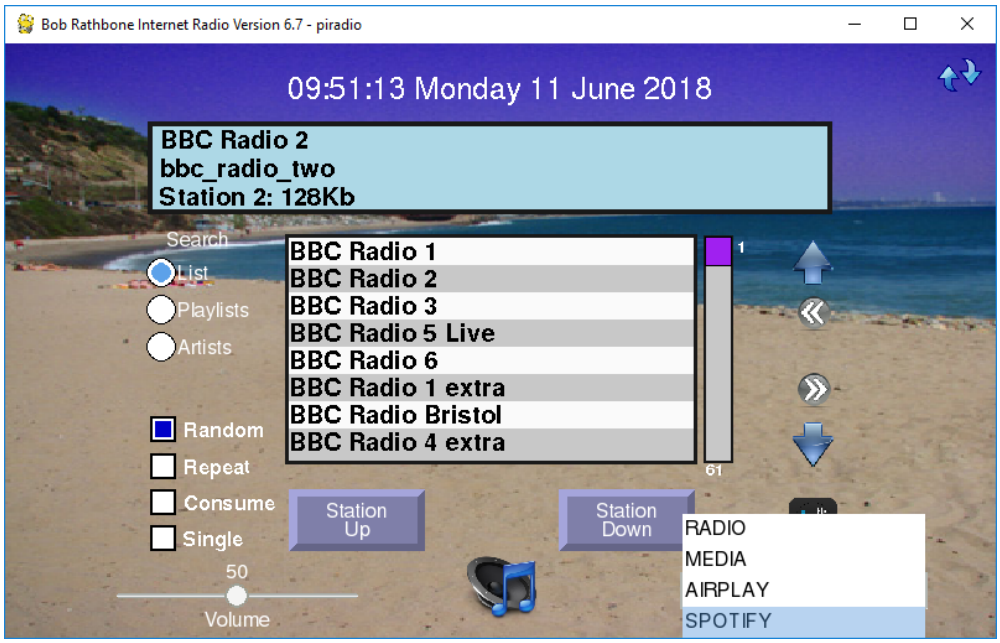


Figure 234 Starting the Spotify Receiver

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

Figure 235 The radio in Spotify mode

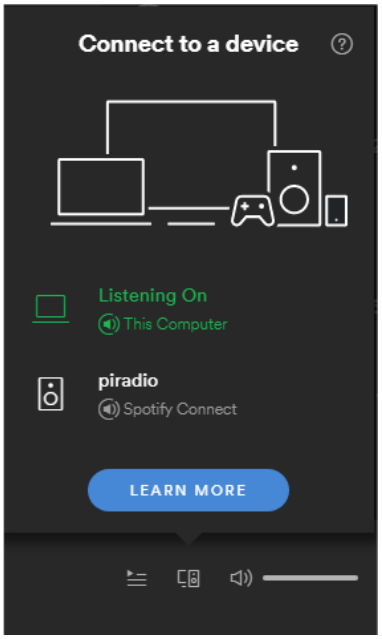


Figure 236 Spotify connecting to the radio

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.

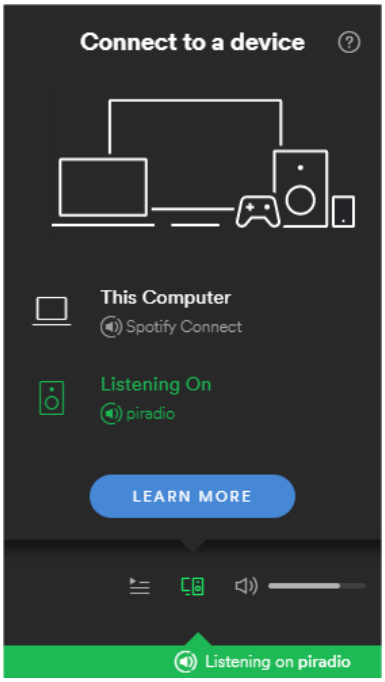


Figure 237 Listening to Spotify on the radio

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 238 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 *Update to the latest the system* packages on page 95. 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).

See section ***Error! Reference source not found.*** on page **Error! Bookmark not defined.** for further information.

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
```

Licences and disclaimer

Licences

The software and documentation for this project is released under the GNU General Public Licence.

The GNU General Public License (GNU GPL or GPL) is the most widely used free software license, which guarantees end users (individuals, organizations, companies) the freedoms to use, study, share (copy), and modify the software. Software that ensures that these rights are retained is called free software. The license was originally written by Richard Stallman of the Free Software Foundation (FSF) for the GNU project.

The GPL grants the recipients of a computer program the rights of the Free Software Definition and uses *copyleft* to ensure the freedoms are preserved whenever the work is distributed, even when the work is changed or added to. The GPL is a *copyleft* license, which means that derived works can only be distributed under the same license terms. This is in distinction to permissive free software licenses, of which the BSD licenses are the standard examples. GPL was the first *copyleft* license for general use. This means that you may modify and distribute the software and documentation subject to the conditions of the licences.

See http://www.gnu.org/licenses for further information on the GNU General Public License.

The licences for the source and documentation for this project are:

GNU General Public License.	See http://www.gnu.org/licenses/gpl.html
GNU AFFERO General Public License.	See http://www.gnu.org/licenses/agpl.html
GNU Free Documentation License.	See http://www.gnu.org/licenses/fdl.html

Intellectual Property, Copyright, and Streaming Media

This is an unbelievably complex subject. The author is not a lawyer and cannot offer any legal advice on this subject. If you decide to stream your music content or relay a radio station stream back out to the internet or within a public building or space then you should seek legal advice.
See also: http://en.wikipedia.org/wiki/Copyright_aspects_of_downloading_and_streaming

In general Radio stations are providing a stream to promote their radio station. As media providers they should have arrangements in place to make the content that they provide is legally streamed across the Internet but not all do. The question is it legal to listen (or view) such content is a complex one and subject to local and international laws and which vary considerably.

If you implement **Icecast** or any other streaming technology to re-stream content within your own home then provided that this is not streamed back out to the Internet or a public location then one would think that you will not encounter any problems (but you never know).

If you stream music tracks or relay radio stations back out onto the internet or public space then almost certainly you will be infringing a copyright law or intellectual property rights somewhere. The penalties for such an infringement can be severe.

WARNING: YOU USE THE ICECAST STREAMING IN THIS PROJECT AT YOUR OWN RISK ESPECIALLY IF YOU MAKE THE STREAM CONTENT AVAILABLE ACROSS THE INTERNET OR PUBLIC SPACE, EVEN IF YOU ARE JUST RELAYING AN EXISTING MEDIA STREAM, LEGAL OR OTHERWISE.

Also see the Disclaimer on page 224.

Disclaimer

THIS SOFTWARE AND DOCUMENTATION IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS 'AS IS' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BELIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE OR DOCUMENTATION, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Glossary

ALSA Advanced Linux Sound Architecture

CAD Control and Display (PiFace)

DAC Digital to Analogue Converter (Digital to audio frequency analogue in this case)

DT Device Tree (Overlay). Device (Sound cards) configuration in **/boot/firmware/config.txt** in Raspbian

EMI Electromagnetic Interference (For example fluorescent lighting etc.)

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)

LCD Liquid Crystal Display, also see OLED

LIRC Linux Remote Control software

M3U MPEG3 URL

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

OLED Organic Light Emitting diode. Also, OLED character displays gradually replacing LCDs

OS Operating system (Raspbian Bullseye in this case)

RSS Really Simple Syndication – Web feed usually containing news items

Appendix A – Wiring diagrams and lists

The following tables shows the wiring for the various versions of the radio. These configurations are normally set up by the **configure_radio.sh** program with the exception of the Vintage radio.
See *Configuring the radio* on page 107. It is also necessary to set the **pull_up_down** parameter in **/etc/radiod.conf** depending on wiring.

A1 Push Button and Rotary Encoder 40-pin wiring

The following table shows the wiring for the 40-pin push-buttons or rotary encoders.

Table 21 40-PinPush-buttons/Rotary encoder Wiring

Buttons/Encoders	Pin	GPIO	Configuration	Radio function
Menu button	11	17	menu_switch	Menu
Channel down button/Rotary switch A	16	23	down_switch	Channel down
Channel up button/Rotary switch B	18	24	up_switch	Channel up
Mute button	7	4	mute_switch	Mute sound
Volume down button/Rotary switch A	8	14	left_switch	Volume Down
Volume up button/Rotary switch B	10	15	right _switch	Volume Up

A.2 Push Button and Rotary Encoder 26-pin wiring

The following table shows the wiring for the 26-pin push-buttons or rotary encoders.

Table 22 26-PinPush-buttons/Rotary encoder Wiring

Buttons/Encoders	Pin	GPIO	Configuration	Radio function
Menu button	22	25	menu_switch	Menu
Channel down button/Rotary switch A	19	10	down_switch	Channel down
Channel up button/Rotary switch B	11	17	up_switch	Channel up
Mute button	7	4	mute_switch	Mute sound
Volume down button/Rotary switch A	8	14	left_switch	Volume Down
Volume up button/Rotary switch B	10	15	right_switch	Volume Up

Set **pull_up_down=up** in **/etc/radiod.conf** depending on wiring.

A.3 IQaudIO Cosmic Controller wiring

The following table shows the wiring for the IQaudIO Cosmic controller.

Table 23 IQaudIO Cosmic Controller Wiring

Physical control	Pin	GPIO	Configuration	Radio function
Left hand push button	7	6	down_switch	Channel down
Middle push button	29	5	menu_switch	Menu
Right hand push button	31	4	up_switch	Channel up
Rotary encoder A input	16	23	left_switch	Volume control
Rotary encoder B input	18	24	right_switch	" "
Rotary encoder Switch	13	27	mute_switch	Mute switch
Left status LED	8	14	rgb_red	Error status
Middle status LED	10	15	rgb_blue	Busy status
Right status LED	36	16	rgb_green	Normal status
IR sensor	22	25	In /boot/firmware/config.txt	Remote control

A.4 Pimoroni Pirate Radio wiring

The following table shows the wiring for the Pimoroni Pirate radio (pHat BEAT). Orientation is with the basic push buttons on the left-hand side. The menu button is on the top left-hand side.

Table 24 Pimoroni Pirate radio (pHat BEAT) Wiring

Pimoroni buttons	Pin	GPIO	Configuration	Radio function
On Off button	32	12	menu_switch	Menu
Channel Up button	29	5	up_switch	Channel up
Mute button	31	6	mute_switch	Mute sound
Channel Down button	16	13	down_switch	Channel Up
Volume Up button	36	16	right_switch	Volume Up
Volume Down Button	37	26	left_switch	Volume Down

The On/Off button used in the Pimoroni Radio software re-assigned as the menu switch with the Rathbone radio software. All LCD outputs are set to zero (No display).
Set **pull_up_down=up** in **/etc/radiod.conf**

A.5 Pimoroni Pirate Audio wiring

The following table shows the wiring for the Pimoroni Pirate radio (pHat BEAT).

Table 25 Pimoroni Pirate radio Audio Wiring

Radio buttons	Pimoroni	Pin	GPIO	Configuration	Radio function
Channel Up button	X	36	16	up_switch	Channel up
Channel Down button	A	29	5	down_switch	Channel Down
Volume Up button	Y	18	20/24 (1)	right_switch	Volume Up
Volume Down Button	B	31	6	left_switch	Volume Down

Note 1: Button Y (Volume up) can be GPIO 24 on earlier versions of the Pirate Audio card.
The new settings in **/etc/radiod.conf** are normally set to the following:

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

For earlier versions of the Pirate Audio this is.

```
right_switch=24
```

The Pimoroni Pirate Audio only has four buttons which means there are no buttons available for **Menu** or **Mute**.
For **Mute** press **Volume UP** and **Volume DOWN** together.
For **Menu** press **Channel UP** and **Channel Down** together.

The menu function currently doesn't work that well at the moment but is good enough to get to the Search menu. There is no shutdown function using menu button long press as with other designs.

Set **pull_up_down=up** in **/etc/radiod.conf**

A.6 Vintage Radio Push-button/Rotary Encoder 40-pin wiring

The following table shows the wiring for the 40-pin push-buttons or rotary encoders. This set-up must be manually configured in **/etc/radiod.conf**. It cannot currently be configured by the **configure_radio.sh script**.

Table 26 40-PinPush-buttons/Rotary encoder Wiring

Buttons/Encoders	Pin	GPIO	Configuration	Radio function
Menu button	22	25	menu_switch	Menu
Channel down button/Rotary switch A	19	10	down_switch	Channel down
Channel up button/Rotary switch B	11	17	up_switch	Channel up
Mute button	7	4	mute_switch	Mute sound
Volume down button/Rotary switch A	8	14	left_switch	Volume Down
Volume up button/Rotary switch B	15	10	right _switch	Volume Up

Table 27 Status LED indications

GPIO	Pin	LED	Function
23	16	Red	Error condition, shutdown in progress, IR activity (If configured)
22	15	Blue	Busy condition such as start-up, loading or changing radio stations or tracks.
27	13	Green	Normal operation such as playing stations or tracks.

Table 28 Rotary menu switch

GPIO	Pin	Switch value
24	18	1
8	24	2
7	26	4

Combining the above switch values gives a composite switch value of 0 through 7.
0=Idle, 1=Speak current station/track, 3=Search mode, 4=Source menu, 5=Options Menu 6,7 unused

A.7 SH1106 SPI 1.3” OLED with joystick and 3-buttons

Table 29 SH1106 1.3" OLED with joystick – GPIO pin assignments

Joystick	Buttons	SPI Interface	Radio Function	GPIO Pin	Configuration
Up			Channel Up	6	/etc/radiod.conf
Down			Channel Down	19	“ “ “
Left			Volume Down	5	“ “ “
Right			Volume Up	26	“ “ “
Push			Menu switch	13	“ “ “
	KEY1		Mute Switch	21	/etc/radiod.conf
	KEY2		Not used	n/a	“ “ “
	KEY3		Not used	n/a	“ “ “
		RST_PIN		25	sh1106_config.py
		DC_PIN		24	“ “
		CS_PIN		8	“ “
		BL_PIN		18	“ “

Please note that none of these pin assignments for the SH1106 1.3” OLED display with joystick can be changed by the user and are documented purely for reference.

The GPIO pin assignments for the joystick and button interface are defined in the **/etc/radiod.conf** configuration file and are configured by the **configure_radio.sh** program. The display settings are also configured in **radiod.conf**.

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

display_type=SH1106_SPI

display_width=16
display_lines=4

lcd_select=0
lcd_enable=0
lcd_data4=0
lcd_data5=0
lcd_data6=0
lcd_data7=0
```

The GPIO pin assignments for the SPI configuration are defined in the **sh1106_config.py** configuration file in the **/usr/share/radio program** directory.

```
# Pin definition
RST_PIN         = 25
DC_PIN          = 24
CS_PIN          = 8
BL_PIN          = 18
```

Converting the 1.3” OLED pHat to use the I2C interface

Warning: Do not attempt this procedure unless you possess considerable soldering skills. The risk of damaging both the card and the Raspberry Pi is great if you make a mistake.

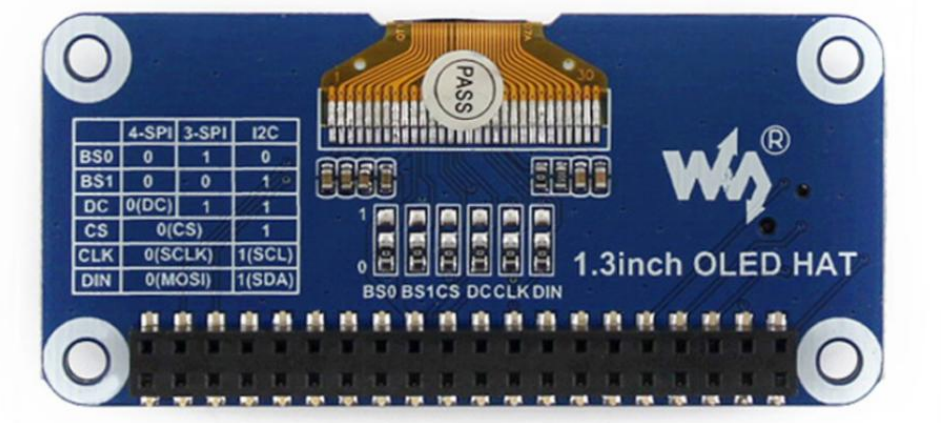


Figure 239 1.3" OLED pHat reverse side

The module uses a 4-wire SPI communication mode by default, that is, BS0, BS1, DC, CS, CLK, and DIN are connected to 0 by default (1 and 0 do not represent the level, but the welding method of connecting or connecting the resistance, the specific hardware link is as follows surface):

Note: The above picture is the welding on the hardware, the following table is the actual hardware connection:

Communication method	BS1/BS0	CS	DC	DIN	CLK
3-wire SPI	0/1	CS	1	MOSI	SCLK
4-wire SPI	0/0	CS	DC	MOSI	SCLK
I2C	1/0	0	1	SDA	SCL

The specific hardware re-wiring for I2C is as follows:
Connect BS0 to 0 to GND, BS1 to 1 to VCC (3.3V), CS to 1 to connect to GND, D/C to 1 to connect to GND, DIN to 1 to connect to Raspberry Pi SDA, CLK to 1 to connect to the tree Raspberry Pi SCL; when using I2C: the high and low states of DC can control the address of the slave device, here is connected to GND, then the 7-bit address of I2C is: 0x3C.

To switch to I2C using the I2C address 0x03 it is necessary to amend the **sh1106_config.py** configuration file. Change the **Device_SPI** and **Device_I2C** definitions in the configuration file to enable the I2C interface instead of I2C.

```
Device_SPI = 1
Device_I2C = 0
```

To

```
Device_SPI = 0
Device_I2C = 1
```

See https://www.waveshare.com/wiki/1.3inch_OLED_HAT for further information.

The author has not tested the above procedure so if any technical support is required, please contact Waveshare directly at https://support.waveshare.com

A.8 Waveshare 1.42” and 1.5” SPI OLED display wiring

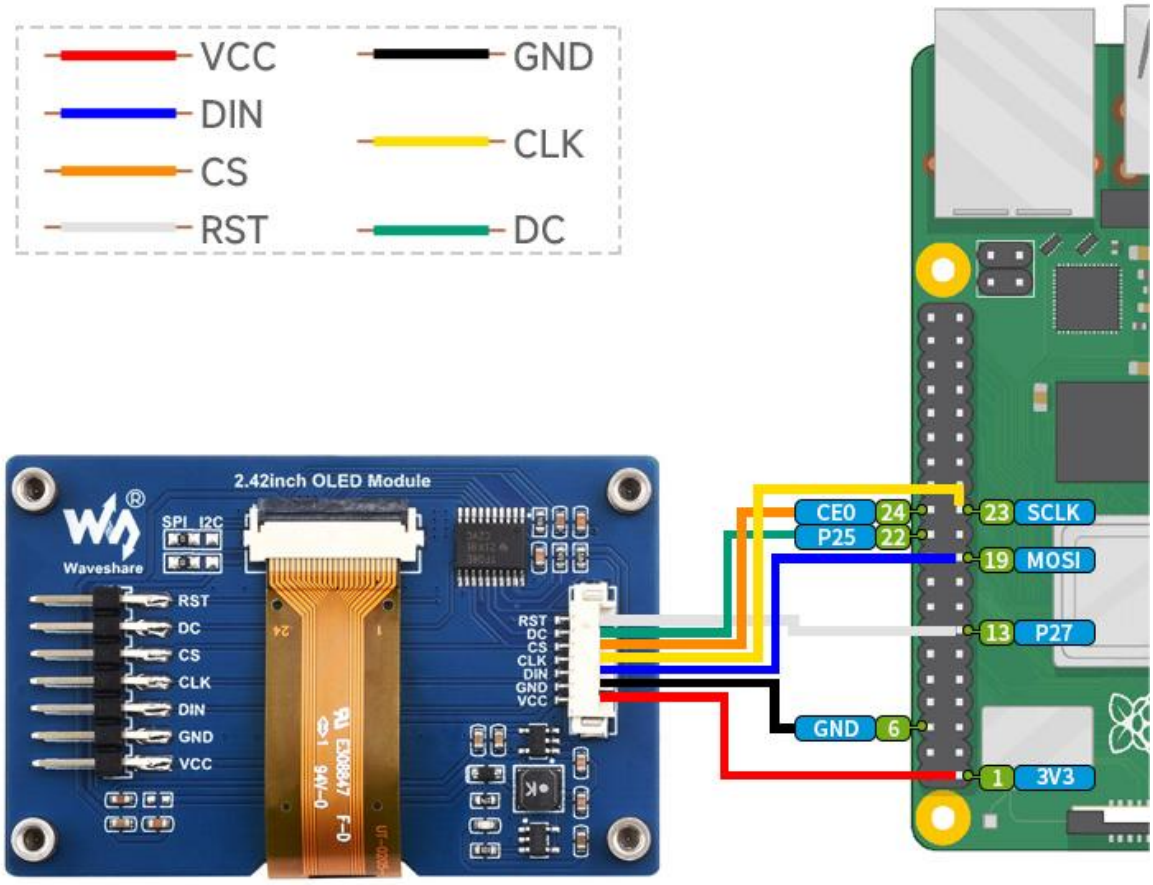


Figure 240 Waveshare 1.42" SPI OLED wiring

Table 30 SPI wiring scheme for OLEDs using the SPI interface

Waveshare	Alt. Name	Description	Colour	RPi Pin	RPi Name
RST	RES	Reset	White	13	GPIO27
DC	D/C	Data/Command	Green	22	GPIO25
CS	CS	Chip Select	Orange	24	CE0
CLK	SCLK	Clock	Yellow	23	SCLK
DIN	SDA	Data In	Blue	19	MOSI
GND	VDD	GND 0V	Black	6	GND
VCC	VSS	+3.3V	Red	1	3V3

RPi Pin numbers in column five are the physical GPIO header pin numbers and not GPIO numbers
The second column are alternative names used by other manufacturers of SPI products

Appendix B – Using a battery pack

It may be that you wish to make your radio portable. This can be done by using a 5V battery pack.

Figure 241 Raspberry Pi 5 Volt battery pack

There are various manufacturers who supply a range of battery 5-Volt packs specifically for the Raspberry Pi. These are charged from any normal 5V charger.

Try to use a battery pack that directly connects to the GPIO header rather than connecting via the micro-USB or USB-C port.

The reason is that if connected by the USB port the internal regulator it will drop the voltage to 4.8 Volts which may give problems with the Wi-Fi connection.

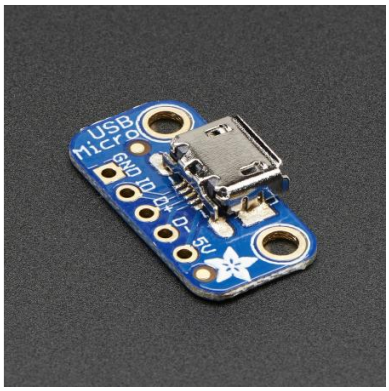


Figure 242 Micro USB breakout board

If the battery pack chosen doesn't connect directly to the GPIO header then purchase a micro-USB breakout board. This allows the 5 volt and GND connections to be connected directly to pins 2 and 6 respectively on the GPIO header.

Index