

Raspberry PI Internet Radio

Constructors Manual



Bob Rathbone Computer Consultancy

www.bobrathbone.com

20th of August 2013

Contents

Introduction	5
Hardware	7
Raspberry PI computer	7
The HD44780 LCD display	7
Wiring.....	8
LCD Module Wiring.....	9
GPIO Hardware Notes.....	10
Parts List.....	11
Construction HD44780 LCD.....	12
Building the LCD and pushbuttons interface board.....	12
Construction using an Adafruit LCD plate.....	14
Software installation.....	15
Software download.....	15
Update the GPIO libraries	15
Install the I2C libraries	16
Enable Network Time Daemon	17
Source files.....	18
The LCD Class	18
The Radio Daemon.....	18
The Adafruit Radio daemon.....	18
The Daemon Class.....	18
The Radio Class	18
The Log class	18
The RSS class	19
The Translate class.....	19
LCD test programs.....	19
Installing and Testing the Music Player Daemon.....	20
Install the Music Player Daemon	20
Configure the MPD daemon	20

Test the Music Player Daemon	20
Configure the Radio daemon	21
Disable serial interface.....	21
Configure radio program log rotation.....	21
Configure the Radio program to start automatically at boot time.....	21
Operation	22
Starting the program.....	22
Buttons.....	23
Mute function	24
Playing MP3 and WMA files.....	24
Organising the music files	24
MPD Logging	24
Radio program logging.....	24
Configuration files.....	25
Displaying an RSS feed	25
Music Player Clients	25
Shutting down the radio	25
Understanding PLS files	26
Installing the Web interface.....	28
Install Apache.....	28
Test the Apache web browser	28
Install the Web Browser server pages	28
Start the radio web interface.....	28
Mounting a network drive	30
Finding the IP address of the network drive.....	30
The CIFS mount command.....	30
The NFS mount command	31
Display the share directory	31
Un-mounting the /share directory.....	31
Copy the mount command to the configuration.....	31
Load the music library.....	31
Update the playlists for the new share.....	31
Disabling the share.....	32
Further information	32

Troubleshooting.....	33
LCD screen not working	33
The LCD displays hieroglyphics	33
LCD backlight not working	33
LCD only displays dark blocks on the first line	33
Music Player Daemon won't start	33
The MPD may display a socket error	33
The MPD daemon complains about the avahi daemon	33
Radio daemon doesn't start or hangs.....	34
Stream decode problems.....	34
Cannot mount remote network drive.....	34
Configuring a wireless adaptor	36
Install the wireless adapter	36
Configure the adaptor.....	36
Explanation of the network fields	37
Operating the wireless interface	37
Troubleshooting the wireless adapter	38
Licences.....	39
Technical support.....	39
Acknowledgements.....	39
Glossary.....	40

Introduction

This manual describes how to create an Internet Radio using the Raspberry PI educational computer. The source and basic construction details are available from the following web site:

http://www.bobrathbone.com/raspberrypi_radio.htm

This manual provides a more detailed overview of construction and software installation than provided by the web site. It contains instructions for building the radio using either the HDD44780 LCD directly wired to the Raspberry PI GPIO pins or using an Adafruit RGB-backlit LCD plate for Raspberry PI.

The features of the radio are:

- Raspberry PI running standard Music Player Daemon (MPD)
- Three different LCDs supported
 - 2 x 16 character LCD with HD44780 controller
 - 4 x 20 character LCD with HD44780 controller
 - Adafruit LCD plate with 5 push buttons (I2C interface)
- Clock display or IP address display (for web interface)
- Five push button operation (Menu, Volume Up, Down, Channel Up, Down)
- Artist and track scrolling search function
- Plays music from a USB stick or from a Network drive (NAS)
- Menu option to display a single RSS news feed
- Web interface using snooppy and others
- Plays Radio streams or MP3 and WMA tracks
- Controlled by an Object Orientated Python application
- Support for European character sets (Limited by LCD capabilities)

Various examples of the Raspberry PI internet radio were built using this design.



This classic style Internet Radio is built into a wooden case. This is using two four inch speakers and audio amplifier stripped out from an old pair of PC speakers. It has five buttons in all. The centre square button is the menu selection.

For alternative ideas see both the next pages and the constructors page at http://www.bobrathbone.com/pi_radio_constructors.htm



Example of the Internet Radio with a four line by twenty character compatible HD4478 LCD display. The transparent case was an old cream cracker box.



Example of the PI internet radio using an Adafruit RGB-backlit LCD plate for Raspberry PI from AdaFruit industries.



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



The prototype system. This consists of a Raspberry PI connected to an LCD display mounted in a black plastic case and two rocker switches and a centre push button for the menu. Using rocker switches however means that the mute function won't work with this design as it isn't possible to press the volume up and down buttons simultaneously.

Hardware

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

- A Raspberry Pi computer
- An HD44780 LCD display or an Adafruit RGB-backlit LCD plate for the Raspberry Pi
- LCD and switches interface board

Raspberry Pi computer

The **Raspberry Pi** is a credit-card-sized single-board computer developed in the United Kingdom by the [Raspberry Pi Foundation](http://www.raspberrypi.org/) with the intention of promoting the teaching of basic computer science in schools.

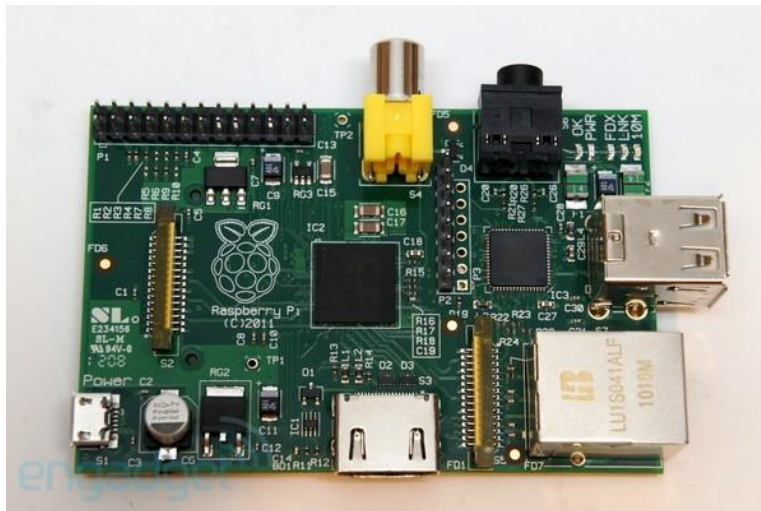


Figure 1 Raspberry Pi Computer

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 beginners guide. http://elinux.org/RPi_Beginners

The HD44780 LCD display



The HD44780 LCD interface is an industry standard interface for a variety of LCD displays. These can come in various sizes but the two lines by 16 character display is the most popular. The software for this Internet radio also supports the four lines by twenty character display. Most of the 16x2 modules available are compatible with the Hitachi HD44780 LCD controller so there is a wide choice of these displays. For pin-out details see *LCD Module Wiring* on page 9.

Wiring

The following table shows the GPIO LCD interface wiring. If using the Adafruit LCD plate then skip this section (See *Construction using an Adafruit LCD plate* on page 14). This wiring works for both revisions 1 and 2 boards of the Raspberry PI however revision 2 boards require a small code change in the `lcd_class.py` code. Refer to RPI Low level Peripherals page at elinux.org for more information on Raspberry PI low level peripheral wiring.

Pin	Description	ModMyPi Board	LCD	Function	Switches
1	3V3	3V3			COMMON
2	5V	5V	2,15	5V	
3	GPIO 0	SDA0			
4	Reserved	Reserved			
5	GPIO1	SCL0			
6	GND	0V	1,3,5,16	GND	
7	GPIO 4	GPIO 7			
8	GPIO 14	UART TX			LEFT
9	Reserved	Reserved			
10	GPIO 15	UART RX			RIGHT
11	GPIO 17	GPIO 0			UP
12	GPIO 18	GPIO 1			DOWN
13	GPIO 27(21)	GPIO 2	11	Data 4	
14	Reserved	Reserved			
15	GPIO 22	GPIO 3	12	Data 5	
16	GPIO 23	GPIO 4	13	Data 6	
17	Reserved	Reserved			
18	GPIO 24	GPIO 5	14	Data 7	
19	GPIO 10	SPI MOSI			
20	Reserved	Reserved			
21	GPIO 9	SPI MOSO			
22	GPIO 25	GPIO 6			MENU
23	GPIO 11	SPI SCLK			
24	GPIO 8	SPIO CE0	6	E	
25	Reserved	Reserved			
26	GPIO 7	SPIO CE1	4	RS	

Pin 13 is GPIO27 on Rev 2 boards and GPIO21 on Rev 1 boards

Wire one side of the switches to the 3.3V pin. Wire the other side of each switch to the GPIO pin as shown in the last column of the above table via a 1K Ω resistor. Also wire this same side of the switch to the 0V pin via a 10K Ω resistor. See Figure 2 on page 9.

For more information about wiring GPIO interface lines see the following link for some excellent GPIO wiring examples from Circuit Lab:

<https://www.circuitlab.com/browse/by-tag/gpio>

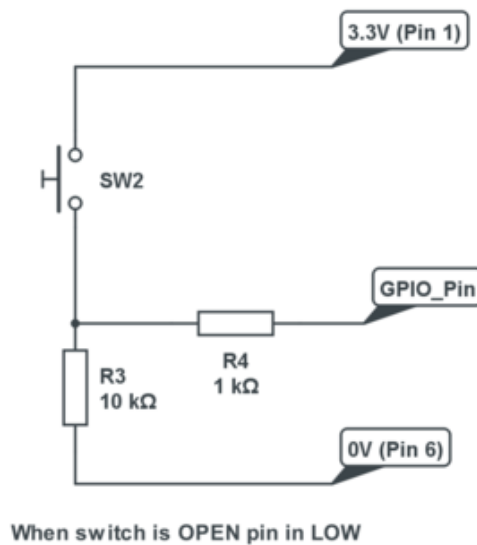


Figure 2 Switch Wiring

Note: The circuit will work without the 1KΩ resistor but is advised for extra protection for the GPIO input.

LCD Module Wiring

The following shows the wiring for the HD44780 LCD controller. It has 16 pins.

1. Ground (0V)
2. VCC (Usually +5V)
3. Contrast adjustment (0V gives maximum contrast)
4. Register Select (RS).
RS=0: Command, RS=1: Data
5. Read/Write (R/W). Very important this pin must be grounded!
R/W=0 (GND): Write, R/W=1 (+5V): Read
6. Enable
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. Data Bit 4
12. Data Bit 5
13. Data Bit 6
14. Data Bit 7
15. LED Backlight Anode (+5V)
16. LED Backlight Cathode (GND)

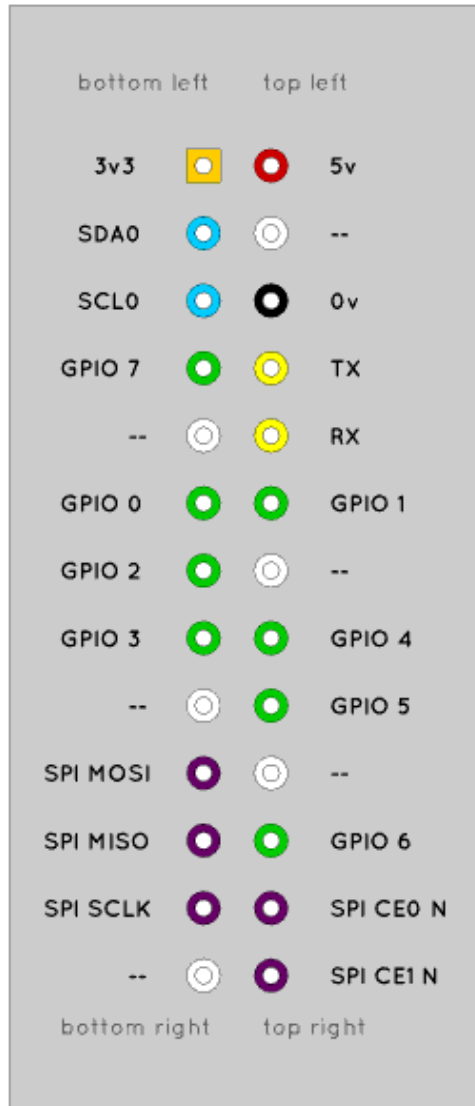
Usually the device requires 8 data lines to provide data to Bits 0-7. However the device can be set to a “4 bit” mode which allows you to send data in two chunks (or nibbles) of 4 bits. This is great as it reduces the number of GPIO connections you require when interfacing with your Raspberry Pi.

GPIO Hardware Notes

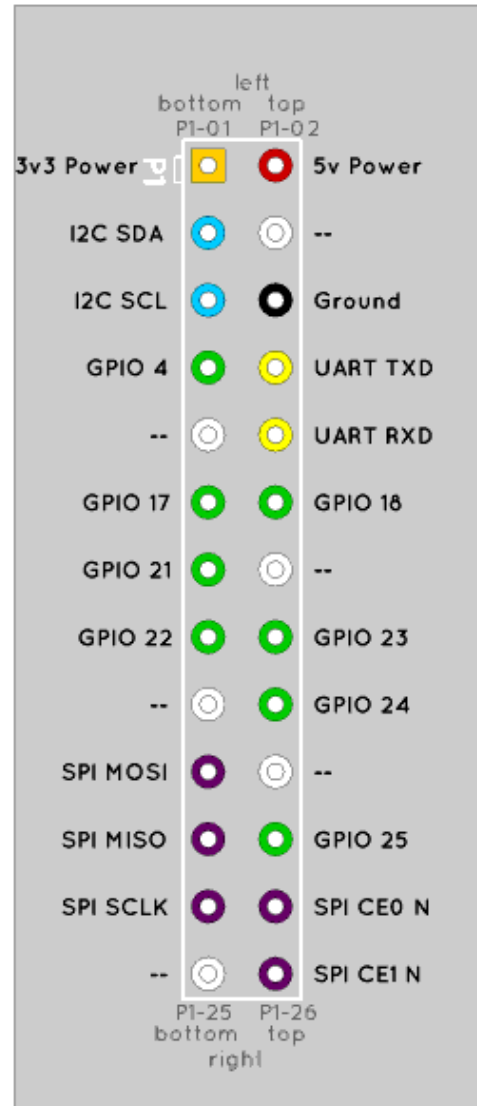
The following shows the pin outs for the GPIO pins. For more information see:

http://elinux.org/RPi_Low-level_peripherals

Raspberry PI GPIO numbering



Raspberry GPIO Broadcom numbering



Note: On rev 2 boards GPIO21 is now GPIO27

Parts List

The following table shows the parts list for the Raspberry Pi Internet Radio. This list is for the version using the HD44780 LCD directly connected to the GPIO pins. If using the Adafruit five button LCD Plate then don't order the parts marked with an asterisk (*)

Qty	Part	Supplier
1	Raspberry Pi Computer	Farnell Element 14
1	Clear Raspberry Case	RS Components
1	4GByte SD Card	Any PC or Photographic supplier
1	Wooden Radio Case	A good friend of mine
1	Raspbian Wheezy OS	Raspberry Pi foundation downloads
2	Four inch loudspeakers	From set of old PC speakers
2	Four inch loudspeaker grills	Any electronics shop
1	Stereo Amplifier (3 to 5 watt)	From set of old PC speakers
1	Transformer for amplifier	From set of old PC speakers
1	LCD HD44780 2 x 16 Display *	Farnell Element 14
1	ModMyPi Slice of Pi *	Ciseco PLC
4	Round push buttons *	Any electronics shop
1	Square push button *	Any electronics shop
1	26 way ribbon cable	Tandy or Farnell Element 14
5	10KΩ resistors *	Tandy or Farnell Element 14
5	1K resistors *	Tandy or Farnell Element 14
1	Four port USB hub	Any PC supplier
1	External power supply for USB hub (1200 mA)	Any PC supplier
1	26 way PCB mount male connector	Any electronics shop
1	Mains cable	Hardware shop
1	Double pole mains switch with neon	Farnell Element 14
5	Male 2 pin PCB mount connectors	Any electronics shop
2	Female 4 pin PCB connectors	Any electronics shop
1	Female 2 pin PCB connectors	Any electronics shop
1	16 pin male in-line PCB mount connector	Any electronics shop
1	Stereo jack plug socket	Any electronics shop
1	Wall mount Ethernet socket	Any do-it-self shop
	Shrink wrap	Any electronics shop
	Thin wire for PCB wiring	Any electronics shop

* These components are not required if using the Adafruit LCD plate.

Construction HD44780 LCD

The following is for an HD44780 LCD display directly wired to the GPIO pins. If using the Adafruit LCD plate see section called *Construction using an Adafruit LCD* plate on page 14.

The following illustration shows the parts for the classic style radio.

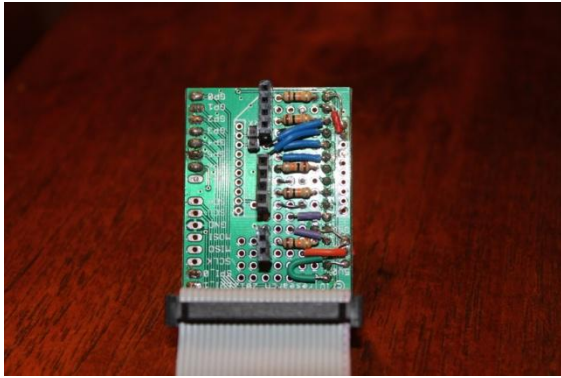


The above photo shows the components before assembly. See from back and left to right. A wooden case, mains cable, Raspberry Pi in a transparent plastic case, speaker grills, 5v power supply, 4 inch speakers, 2 x 16 LCD display, four port USB hub, stereo amplifier, five push button switches, 11.5v transformer for the stereo amplifier, ribbon cable and the LCD and push buttons interface board. Not shown is the (optional) USB wireless dongle.

Building the LCD and pushbuttons interface board

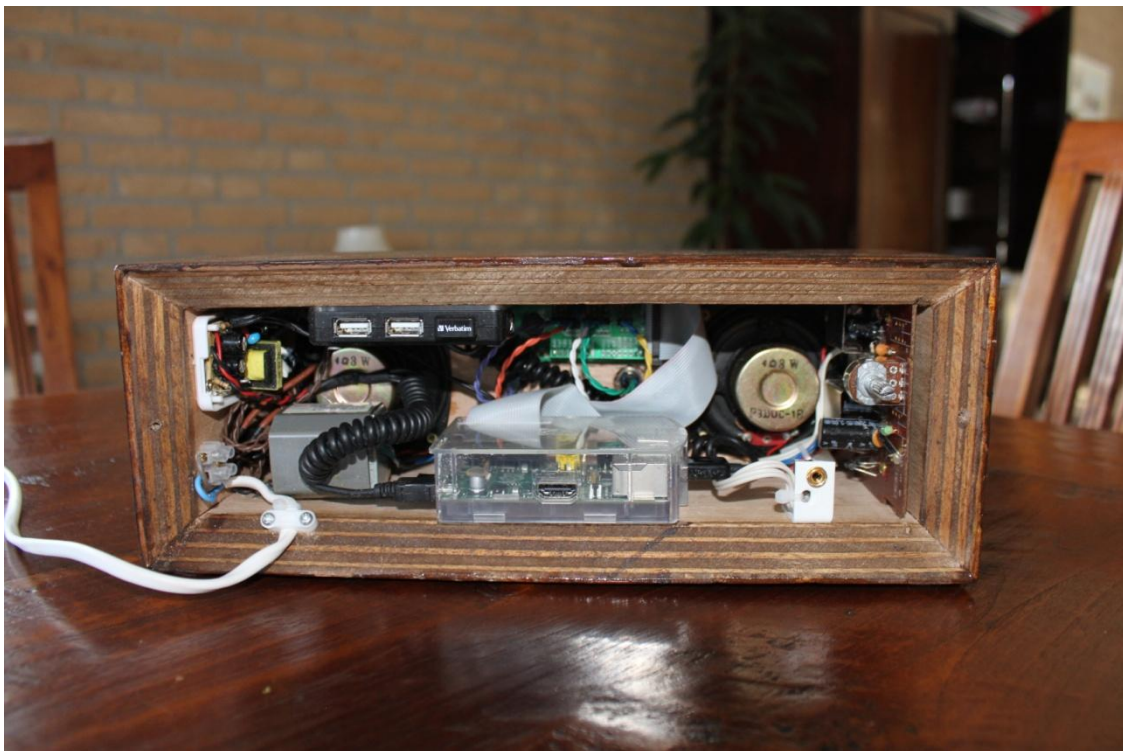


The LCD and push button interface was built using a ModMyPi Slice of PI from [Ciseco PLC](#) but a suitable prototype board will do. The board was fitted with 1 female 16 pin in-line connector for LCD and a male 26 pin connector for the 26 way ribbon cable. If no ribbon cable is to be used then use a female 26 way connector to plug into the Raspberry PI GPIO interface



The component side of the LCD and push button shows the female connectors for five push button switches and the five 10KΩ pull down resistors. In this construction the 1KΩ resistors shown in Figure 2 Switch Wiring on page 7 weren't used but do provide some extra protection for GPIO inputs.

The following picture shows the components mounted in the wooden case.



Shown from top left to bottom right: 5V power supply (from a standard phone charger), four port USB with a memory stick for music files, LCD and Switch interface board (You can just see one of the switches connected with a twisted green wire), stereo amplifier (volume control now just a preset) and 4 inch speakers from a set of old PC speakers, mains input (mains switch behind this), mains transformer for the amplifier, Raspberry PI in a clear plastic case and finally the headphones socket.

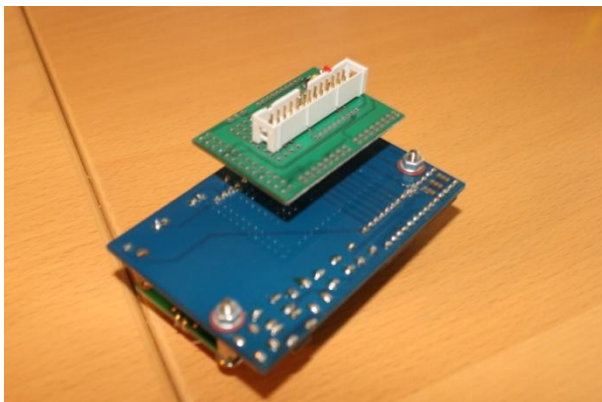
Construction using an Adafruit LCD plate

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)



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



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.

Software installation

This procedure assumes that the Raspberry PI is installed with Debian Wheezy and with a working Internet Connection. There are five steps to carry out to install the software.

- Download and un-tar the software from the Bob Rathbone web site
- Update the GPIO libraries to at least version 0.5.2 or install I2C library (Adafruit plate)
- Install the Music Player Daemon (MPD) and Client (MPC)
- Create play lists for the MPD and test
- Configure and run the radio software

Software download

The software is contained in a compressed tar file called *pi_radio.tar.gz*. This can be downloaded from the following location.

http://www.bobrathbone.com/pi_radio_source.htm

Either download it to the PC and copy it across to the Raspberry PI or use the *wget* facility if there is an Internet connection on the Raspberry PI.

Create a directory called **/home/pi/radio**. Copy the *pi_radio.tar.gz* to the **/home/pi/radio** directory or use **wget** to download it.

```
# wget http://www.bobrathbone.com/raspberrypi/source/pi_radio.tar.gz
```

Un-tar the file with the following command:

```
# tar -xvf pi_radio.tar.gz
```

This will unzip the following files and directory:

lcd_class.py, radio4.py, radio_class.py, radio_daemon.py, radiod.py, test_lcd.py, test_ada_lcd.py, ada_radiopy, i2c_class.py, ada_lcd_class.py, playlists/ python.deb rss/**

Make sure all of the *.py files in this directory are executable with the following command:

```
# chmod +x *.py.
```

Don't try to run the software just yet! It has to be configured first.

Update the GPIO libraries

If using the Adafruit LCD plate you can skip this section.

The last tested version of Debian wheezy (2013-05-25-wheezy-raspian) installs the correct libraries by default so if using this version or above then you may also skip this section.

Update the GPIO libraries to at least version 0.5.2a (Included in the *pi_radio.tar.gz* file). You can check the current version with the command "`dpkg -l | grep gpio`".

```
# dpkg -i python-rpi.gpio_0.5.2a-1_armhf.deb
# dpkg -i python3-rpi.gpio_0.5.2a-1_armhf.deb
```

You may delete the above deb files once the latest libraries have been installed. **Note:** The libraries are for Debian wheezy only. If using another OS download the correct GPIO libraries.

Install the I2C libraries

If you are using the Adafruit plate it is necessary to install the I2C libraries. If using the LCD directly wired to the GPIO pins then skip this section.

For a more basic introduction to setting up I2C on your Pi then you may wish to take a look at this Adafruit tutorial: <http://learn.adafruit.com/adafruits-raspberry-pi-lesson-4-gpio-setup/configuring-i2c>

Edit **/etc/modules** file and add the following lines to the end of the file. Then save and reboot to enable the hardware I2C driver.

```
i2c-bcm2708
i2c-dev
```

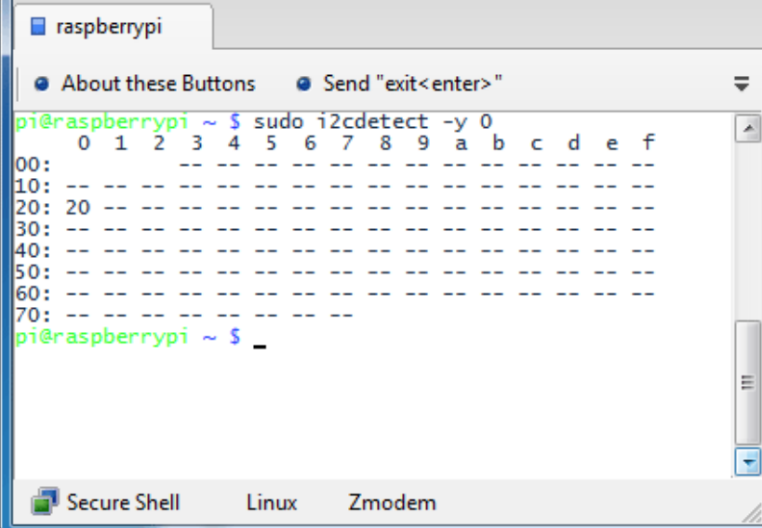
Enter the following commands to add SMBus support (which includes I2C) to Python:

```
sudo apt-get install python-smbus
sudo apt-get install i2c-tools
```

The i2c-tools isn't strictly required, but it's a useful package since you can use it to scan for any I2C or SMBus devices connected to the Raspberry. If you know something is connected, but you don't know it's 7-bit I2C address, this library has a great little tool to help you find it:

```
sudo i2cdetect -y 0 (if you are using a version 1 Raspberry Pi)
sudo i2cdetect -y 1 (if you are using a version 2 Raspberry Pi)
```

This will search /dev/i2c-0 or /dev/i2c-1 for all address, and if the Adafruit LCD Plate is correctly connected, it should show up at **0x20**.



The screenshot shows a terminal window titled 'raspberrypi'. The user has entered the command 'sudo i2cdetect -y 0'. The output is a grid showing I2C addresses from 00 to 70. Address 20 is detected, indicated by the number '20' in the column for address 20. The terminal window also shows buttons for 'About these Buttons' and 'Send "exit<enter>"' at the top, and 'Secure Shell', 'Linux', and 'Zmodem' at the bottom.

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

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

Enable Network Time Daemon

Since the radio displays the time it is necessary to sync the time with a Network Time Protocol (NTP) server. If you don't do this your clock will not be accurate. As root carry out the following command:

```
# service ntp start
```

Test that the time is synchronising OK with the **ntpq peers** command

```
# ntpq
ntpq> peers
      remote               refid          st t when poll reach   delay   offset  jitter
=====
+ns0.solcon.nl      193.79.237.14      2 u  438  512  377   20.337   -1.030   5.302
-mu.monshouwer.e    193.79.237.14      2 u  352  512  377   20.197   -1.858   0.349
*ev001.tilaa.nl     193.190.230.65     2 u  415  512  377   19.891   -0.899   0.379
+ntp.raqxs.nl       212.45.32.36       3 u  388  512  377   20.354   -1.127   0.391
ntpq> quit
```

However to make the NTP daemon start at boot time it needs to be enabled first. Use the **update-rc.d** command to enable it.

```
# update-rc.d ntp enable 2 3 5
```

This enables the NTP daemon (ntpd) for run levels 2 3 and 5.

Source files

The source consists of several source modules all written in Python using Object Orientated techniques. The source can be downloaded from

http://www.bobrathbone.com/pi_radio_source.htm

The LCD Class

The LCD *lcd_class.py* class handles all of the LCD display routines. It contains simple commands to display and scroll lines of text on the HDD44780 2 x 16 LCD or 4 x 20 characters LCD. It is a useful standalone class that can be used in other projects. It is based on the routines from Matt Hawkins. If you are using a revision 1 board you will need to make a small change to the following line:

```
LCD_D4 = 21 # Rev 1 Board
#LCD_D4 = 27 # Rev 2 Board
```

If using the Adafruit LCD plate you can skip the above section.
This class can be used for other projects using the HD44780 LCD display.

The Radio Daemon

There are two versions of this program for the HDD44780 LCD (directly wired to the GPIO pins).

The *radiod.py* source provides the logic for operating the radio. It reads the push button switches and configuration files, loads the music files and radio stations. It is used with a 16 character 2 line LCD.

The *radio4.py* source provides the same logic for operating the radio but for a 20 character 4 line LCD.

The Adafruit Radio daemon

If you are an Adafruit RGB-backlit LCD plate for Raspberry Pi then the following programs are used:

<i>ada_radio.py</i>	The radio daemon for the Adafruit LCD plate.
<i>ada_lcd_class.py</i>	The LCD class using an I2C interface (Also interfaces the switches).
<i>i2c_class.py</i>	The IC2 class courtesy of Adafruit Industries (renamed).
<i>test_ada_lcd.py</i>	Test Adafruit LCD and switches.

The Daemon Class

The *radio_daemon.py* code allows the radio program to run as a background daemon. It allows start, stop, restart, version and status commands.

The Radio Class

The *radio_class.py* contains the actual commands that interface to the Music Player Daemon (MPD).

The Log class

The *log_class.py* routine provides logging of events to **/var/log/radio.log** file.

The RSS class

The *rss_class.py* routines allow sequential gets from an RSS feed. These feeds are provided from news providers such as the BBC. This class gets the RSS feed defined in the ***/var/lib/radiod/rss*** file.

The Translate class

The *translate_class.py* is used to convert special international character sets (particularly from RSS feeds). It does this by first converting them to escape sequences and then to displayable *ascii* characters (These will show up in DEBUG logging). These *ascii* characters are then passed to the LCD class where they may be converted again to a valid character in the standard LCD character set.

LCD test programs

The *lcd_test.py* program provides some simple code to test the LCD. The *test_ada_lcd.py* program is used to test the LCD if you are using an Adafruit RGB-backlit LCD plate for the Raspberry PI.

Installing and Testing the Music Player Daemon

Install the Music Player Daemon

Install the [Music Player Daemon](#) (mpd) and its client (mpc)

```
# apt-get install mpd mpc
```

Copy an example PLS file provided with the software from the `/home/pi/radio/playlists` directory for example the `ukblues.pls` PLS file to the `/var/lib/mpd/playlists` directory.

Configure the MPD daemon

The configuration file for the MPD daemon is `/etc/mpd.conf`. See the **mpd** man page for further information. It is only necessary to amend a couple of parameters in this file to prevent annoying start-up messages. The MPD program will run without modifying the configuration file.

Edit the `/etc/mpd.conf` file:

Change the `bind_to_address` parameter from “localhost” to “any”.

```
#bind_to_address      "localhost"
bind_to_address       "any"
```

This will prevent IPv6 bind errors on start-up.

Change the `zeroconf_enabled` parameter to “no”

```
#zeroconf_enabled     "yes"
zeroconf_enabled       "no"
```

This prevents *avahi* daemon error messages on start-up.
Save the `/etc/mpd.conf` file when finished.

Test the Music Player Daemon

Connect a set of speakers or headphones to the Raspberry Pi and run the following commands to test the MPD daemon:

```
# service mpd start
# mpc load ukblues.pls
# mpc play 1
```

Music should now be heard from Raspberry Pi. If not troubleshoot the problem before continuing.
To stop the MPD daemon enter:

```
# service mpd stop
```

If these tests succeed you are ready to run the Radio software.

Configure the Radio daemon

Disable serial interface

If you are using the Adafruit LCD plate – skip this section.

Two of the pins used by this design namely pin 8(GPIO8) and 10 (GPIO 15) are configured for the serial interface (UART). This must be disabled for reliable operation by removing all references to `ttyAMA0` in the `/boot/cmdline.txt` and `/etc/inittab` files

Add a hash character at the beginning of lines containing `ttyAMA0`.

`/boot/cmdline.txt`

```
#dwc_otg.lpm_enable=0 console=ttyAMA0,115200 kgdboc=ttyAMA0,115200  
console=tty1 root=/dev/mmcblk0p2 rootfstype=ext4 elevator=deadline rootwait
```

`/etc/inittab`

```
#Spawn a getty on Raspberry Pi serial line  
#T0:23:respawn:/sbin/getty -L ttyAMA0 115200 vt100
```

Configure radio program log rotation

The Radio program logs to a file called `/var/log/radio`. This can eventually fill the SD card. Create a file called `/etc/logrotate.d/radiod` with the following lines:

```
/var/log/radio.log {  
    weekly  
    missingok  
    rotate 7  
    compress  
    notifempty  
    copytruncate  
    create 600  
}
```

This will rotate the log files every week so prevent the SD card from eventually filling up.

Configure the Radio program to start automatically at boot time

To automatically start the program at boot time edit the `/etc/rc.local` file and add the following line to the end of it. If using the four line LCD then use `radio4.py` instead of `radiod.py`.

```
/home/pi/radio/radiod.py start
```

The Radio daemon will start automatically at boot time. The above command is for the directly wired LCD. If using the Adafruit LCD plate use the following command instead to start the radio.

```
/home/pi/radio/ada_radio.py start
```

Operation

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. If you are using a 4x20 LCD then substitute *radiod.py* for *radio4.py* in all of the following commands. If using the Adafruit LCD plate then substitute *ada_radio.py* for all of the following commands.

Starting the program

The program must either be run as root user or using sudo.
The basic operation of the program is:

```
radiod.py start|stop|restart|status|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.
version: Show the version number of the program

If you do not see all of the above options you are running old software and should update to the latest release.

Change to the /home/pi/radio directory and run the following command:

```
pi@raspberrypi:~$ cd /home/pi/radio/  
pi@raspberrypi:~/radio$ sudo ./radiod start
```

Alternatively run as root. All commands as root user don't need sudo.

```
# pi@raspberrypi:~$ sudo bash  
# cd /home/pi/radio  
# ./radiod.py start
```

To stop the radio

```
# ./radiod.py stop
```

For the 4 x 20 character LED run:

```
# ./radiod.py start
```

To display the status run:

```
# ./radiod.py status  
radiod running pid 2098
```

The above pid (Process ID) number will be different each time the program is run.

To see what version of the software you are running:

```
# ./radiod.py version  
Version 1.16
```

Buttons

There are five buttons, four function buttons and one menu button. The Menu button changes the display mode and the functions of the left and right hand buttons as shown in the following table.

Table 1 Push Button Operation

LCD Display Mode	Left hand buttons		Right hand buttons	
	Left button	Right button	Left button	Right 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 = 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: Internet Radio or Music Library	Volume Up Mute	Volume Down Mute	Toggle mode between Radio and Music Library	Toggle mode between Radio and Music Library
Mode = OPTIONS Line 1: Menu Selection Line 2: <option> Options are Random, Consume, Repeat or Reload Music:	Toggle selected mode on or off	Toggle selected mode on or off	Cycle through Random, Consume, and Repeat	Cycle through Random, Consume, and Repeat
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 1: If the `/var/lib/radiod/rss` file is missing or contains an invalid RSS URL then this mode is skipped.

Mute function

Pressing both volume buttons together will mute the radio. 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.

Playing MP3 and WMA files

The software also allows you to play music from a USB memory 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. Reboot the PI. Once the Radio program is running, 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.

Organising the music files

For the search routines to work properly the music must be organised in a certain way. The files must be placed in the top level directory of USB stick. The search routines use the first directory as the artist name and the music files themselves as the track name. For example:

Elvis Presley/The 50 Greatest Hits Disc 1/01 That's All Right.mp3

In the above example the first directory is set up with the artist name and this will appear in the search. The subsequent directory "The 50 Greatest Hits Disc 1" is not used by the search routines. The file name "That's All Right.mp3" without the mp3 extension becomes the track name in this example.

MPD Logging

All logging for the MPD daemon is to the **/var/log/mpd/mpd.log** file by default.

Radio program logging

The Radio program logs to a file called **/var/log/radio.log**. See example log below

```
2013-07-02 16:14:49,230 INFO Radio running pid 31168
2013-07-02 16:14:49,268 INFO Radio daemon version 1.15
2013-07-02 16:14:50,168 INFO Linux rathpil 3.2.27+ #250 PREEMPT Thu Oct 18
19:03:02 BST 2012 armv6l GNU/Linux
2013-07-02 16:14:50,225 INFO GPIO version 0.5.2a
2013-07-02 16:14:53,248 DEBUG radio.loadStations
2013-07-02 16:14:54,619 INFO MPD started
2013-07-02 16:14:55,156 INFO mpd version: 0.16.0
```

There are four levels of logging namely DEBUG, INFO, WARNING and ERROR. The log level is configured in the **/var/lib/radiod/loglevel** file. The default is INFO. If you want to increase the logging say to DEBUG carry out the following command as root user and restart the program.

```
# echo DEBUG > /var/lib/radiod/loglevel
# ./radiod.py restart
```

Configuration files

There are some other configuration files in the **/var/lib/radiod** directory. These are:

current_station	The current radio station
current_track	The current music track
volume	The volume setting

You don't normally need to change these files. They are maintained by the program so that when it starts up the program uses the last setting, for example, the volume setting.

Displaying 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.bbc.co.uk/news/uk/rss.xml?edition=int
```

The above is the RSS for the BBC news 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 is provided with a valid BBC RSS feed file in the **rss** directory. You can test the feed first by pasting it into your PC's web browser URL and pressing enter.

Music Player Clients

MPD is designed around a client/server architecture, where the clients and server (MPD is the server) interact over a network. A large number of graphical and web based clients are available for MPD and are too numerous to mention here. Please see the following link for further information on MPD clients.

<http://mpd.wikia.com/wiki/Clients>

Shutting down the radio

You can simply switch the power off. This doesn't appear to 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.

Understanding PLS files

A good place to start is the following Wikipedia article:

[http://en.wikipedia.org/wiki/PLS_\(file_format\)](http://en.wikipedia.org/wiki/PLS_(file_format))

A 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.

Download or create a [PLS](#) file for the radio stations you wish to listen to and copy it to the **/var/lib/mpd/playlists** directory. These files must end in the extension **.pls** for example **mystations.pls** (See 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 must always start with the **[playlist]** statement. The **NumberOfEntries** statement must match the number of streams you have defined in the PLS file (Two in the above example). Set the **Version** number always to 2.

There must be a **File n** , **Title n** and **Length n** where n is the entry number.

The **File n** statement is the pointer to the Radio stream URL. The **Title n** statement may be any useful title you wish. The **Length n** statement is always -1 (Unlimited) for a Radio station. You can create as many playlist files as you wish. The Radio program will attempt to load all of the PLS files it finds in **/var/lib/mpd/playlists** directory.

Please note that a bad PLS file can stop the Radio from working (MPD limitation not the Radio program). Not all radio streams are supported by MPD.

If you suspect a bad playlist stop the Radio and tail the **/var/log/mpd/mpd.log** file and restart the Radio. Below is a typical error:

```
# tail -f /var/log/mpd/mpd.log
May 01 08:22 : player thread: played "http://ics2bss.omroep.nl:80/radio2-bb-aac?q=/npo/aac/radio2-bb.pls&stream=ok"
```

The reason for the above error is that the above URL points to the PLS file. It is not the PLS file entry itself. Use **wget** to get the correct PLS file. Most radio stations provide a URL which points to a URL file.

For example:


```
# cd /tmp/
# wget http://ics2bss.omroep.nl:80/radio2-bb-aac?q=/npo/aac/radio2-
bb.pls&stream=ok
--2013-05-01 09:52:16--
http://network.absoluteradio.co.uk/core/audio/mp3/live.pls?service=vrbb
Resolving network.absoluteradio.co.uk (network.absoluteradio.co.uk)...
31.186.234.209
Connecting to network.absoluteradio.co.uk
(network.absoluteradio.co.uk)|31.186.234.209|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 171 [audio/x-scpls]
Saving to: `live.pls?service=vrbb.1'

100%[=====
=====>] 171          --.-K/s   in 0s

2013-05-01 09:52:16 (2.33 MB/s) - `live.pls?service=vrbb.1' saved [171/171]
```

Now display the file that **wget** just downloaded:

```
# cat live.pls\?service\=vrbb
[playlist]
NumberOfEntries=1
File1=http://stream.timlradio.co.uk/ABSOLUTERADIOIRMP3
Title1=Absolute Radio 1215AM, Discover Real Music (High Quality)
Length1=-1
Version=2
root@raspberrypi:/tmp#
```

Now copy this file to the **/var/lib/mpd/playlist** directory (with a better name).

```
# cp /tmp/live.pls\?service\=vrbb /var/lib/mpd/playlists/absolute_radio.pls
```

Alternatively add the contents to an existing PLS file.

There are a lot of resources on the Internet how to find PLS files so simply search for “PLS files” through the search machine of your choice.

Installing the Web interface

MPD has several web clients. See the following link: <http://mpd.wikia.com/wiki/Clients>. The one used in this example is called snoopy is released with version 1.16 onwards of the radio software.

Install Apache

Install Apache the web server. Make sure that the system is up to date with the following command.

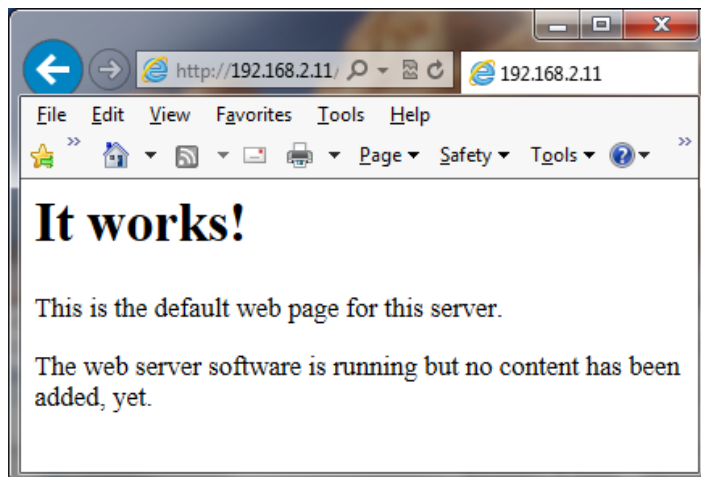
```
# apt-get update
```

Now install Apache and the PHP libraries for Apache.

```
# apt-get install apache2 php5 libapache2-mod-php5
```

Test the Apache web browser

Point your web browser at the IP address of the Raspberry PI. For example: <http://192.168.2.11>. You should see the following display.



Install the Web Browser server pages

It is now necessary to install the web pages for the Radio. Download the radio web pages from http://www.bobrathbone.com/raspberrypi/source/pi_radio_web.tar.gz

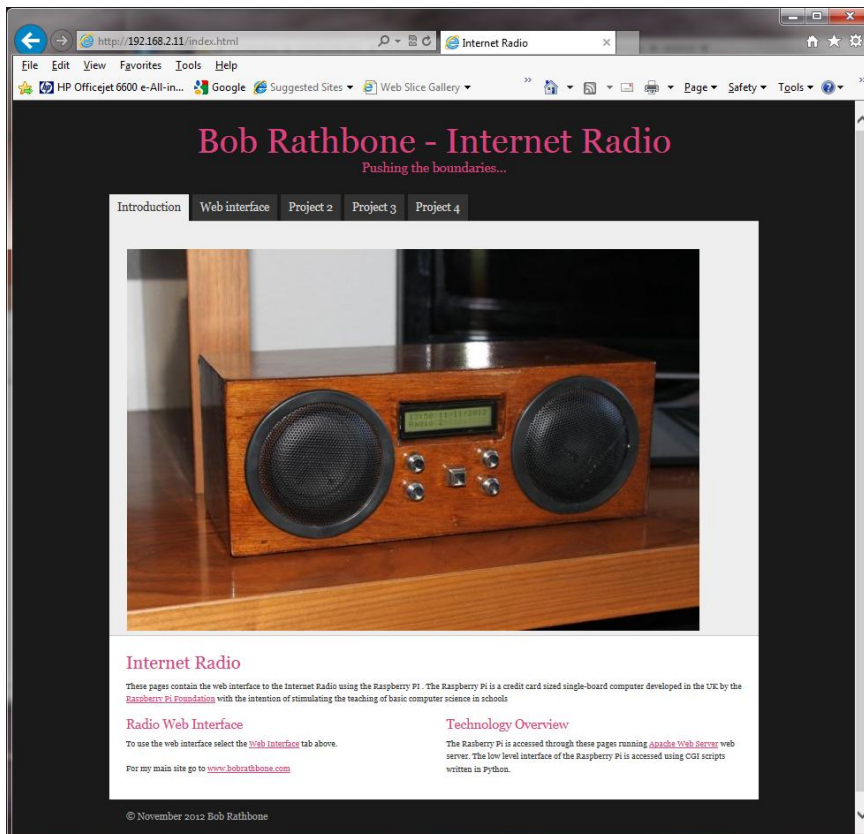
Copy these to the **/home/pi/radio** directory. Now install the pages with the following commands:

```
# cd /  
# tar -xvf /home/pi/radio/pi_radio_web.tar.gz
```

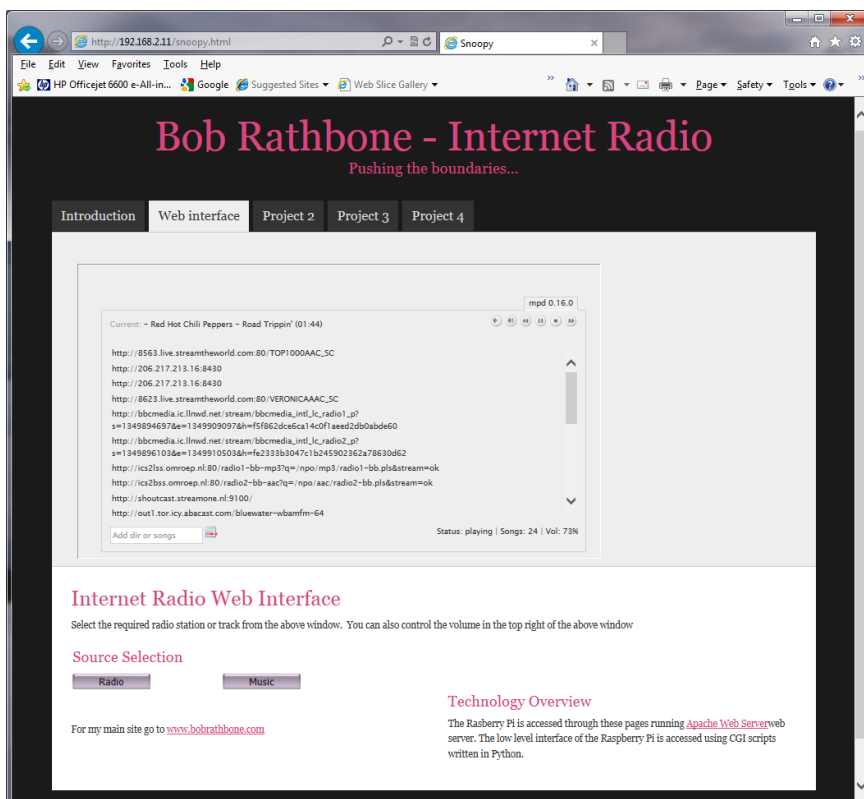
This will install the radio pages in the **/var/www** directory and the CGI scripts in **/usr/lib/cgi-bin** directory.

Start the radio web interface

Point your web browser at the IP address of the Raspberry PI. For example: <http://192.168.2.11>. You should see the following display:



Now click on the 'Web interface tab'. If the radio software is running you will see the following:



Click on any station on the list to select a station. The Radio and Music buttons select the source.

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. This is possible from version 1.6 onwards of the radio software. There are two main types of network drive protocols used by Raspian Wheezy on the Raspberry Pi namely:

- CIFS – Common Internet File System
- NFS – Network File System

There is a third type of network file protocol called SMB (Server Message Block – Microsoft) but is replaced by CIFS in the Raspberry PI. Your PC will be using SMB most probably. The steps to mount the network drive are as follows:

1. Find out the IP address 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 //192.168.2.6/music /share
```

The share directory is created when you first run the Radio program (v1.6 onwards) 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 31.

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 192.168.2.6:/volume1/music /share
```

A few things to note here; the NFS mount command uses the volume name (Volume1), 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 `ls`.

```
# ls -la /share
total 576
drwxrwxrwx 144 1024 users 4096 Feb  1 12:07 .
drwxr-xr-x  23 root  root  4096 May 15 10:48 ..
drwxrwxrwx   3 1024 users 4096 Feb  1 12:05 Adriano Celentano
drwxrwxrwx   3 1024 users 4096 Feb  1 12:07 Afric Simone
drwxrwxrwx   3 1024 users 4096 Feb  1 12:07 Al Martino
drwxrwxrwx   4 1024 users 4096 Feb  1 12:05 America
drwxrwxrwx   3 1024 users 4096 Feb  1 12:06 Aphrodite's Child
```

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 //192.168.2.6/music /share" > /var/lib/radiod/share
```

Load the music library

Now run the radio program. The radio stations will be loaded. Cycle through the menu until **Input Source:** is displayed. Press the channel up or down buttons to select **Music Library**. Now press the **Menu** button. The program loads whatever playlists it has in its database, and will most likely be only those from the USB stick if installed. However the *playlist* for the new share files are not yet in the MPD database. The playlist needs to be updated in the following section.

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).

Disabling the share

To disable the share simply put a hash character (#) at the beginning of the line in the **/var/lib/mpd/share** file as shown in the example below. Alternatively remove the share file altogether.

```
#mount -t cifs -o username=guest,password=guest //192.168.2.6/music /share
```

Further information

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.

```
# ls -la /var/lib/mpd/music/
total 8
drwxr-xr-x 2 root root 4096 May 19 11:17 .
drwxr-xr-x 4 mpd audio 4096 May 16 19:02 ..
lrwxrwxrwx 1 root root    6 May 19 11:17 media -> /media
lrwxrwxrwx 1 root root    6 May 19 11:17 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
# ln -s /media
# ln -s /share
```

This shouldn't normally be necessary as the links are created by the program when it creates the media and share mount points.

Troubleshooting

LCD screen not working

Check that the wiring conforms to the *wiring list* on page 8.

Make sure that pin 3 is grounded (0V) to give maximum contrast.

Run the *lcd_test.py* program to see if the LCD displays anything. This runs independently of any other software and can be used stand alone.

The LCD displays hieroglyphics

If you have a revision 1 Raspberry PI board you need to amend the *lcd_class.py* code as shown in the section called *The LCD Class* on page 18. If you are using a revision 2 board do not amend the *lcd_class.py* code as it is designed to work with this board.

Check the wiring conforms to the *wiring list* on page 8. In particular check the data lines to pins 11, 12, 13 and 14 (See *LCD Module Wiring* on page 9). Retest the LCD using the *lcd_test.py* program.

LCD backlight not working

Check that pins 15 and 16 of the LCD display have +5V and 0V(GND) respectively. See *LCD Module Wiring* on page 9.

LCD only displays dark blocks on the first line

This is normal when the raspberry PI starts up. The display should work with the *lcd_test.py* program. If the *lcd_test.py* program still doesn't display anything then check that the wiring conforms to the *wiring list* on page 8.

Music Player Daemon won't start

The MPD daemon logs to the */var/log/mpd/mpd.log* file. Examine this file for errors. The MPD daemon is dependant on good PLS files so check that these are correct as described in the section called *Understanding PLS files* on page 26.

The MPD may display a socket error

When starting the MPD daemon the following message is seen:

```
Starting Music Player Daemon: mpdlisten: bind to '[:,:1]:6600' failed: Failed
to create socket: Address family not supported by protocol (continuing
anyway, because binding to '127.0.0.1:6600' succeeded)
```

If this message is seen in the MPD log file this is simply because IP version 6 (IPv6) isn't installed so the message doesn't affect operation of the MPD.

To prevent it from happening configure the *bind_to_address* parameter in the */etc/mpd.conf* file to "any" as explained in the section called *Configure the MPD daemon* on page 20.

The MPD daemon complains about the avahi daemon

The following message is seen in the */var/log/mpd/mpd.log* file

```
Apr 10 15:37 : avahi: Failed to create client: Daemon not running.
```

Change the `zeroconf_enabled` parameter in the `/etc/mpd.conf` file to “no” as explained in the section called *Configure the MPD daemon* on page 20. The *avahi* daemon is used to configure systems without a network connection but is not enabled by default. It is not required for this design.

Radio daemon doesn't start or hangs

This is almost certainly a problem with either the MPD daemon or failed internet connection.”Check the network connection and run installation tests on the MPD daemon. Occasionally a bad PLS file can cause this problem. You can check that your Raspberry PI has an internet connection with the `ip addr` command. The example below shows interface `eth0` connected as IP 192.168.2.22.

```
# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state
UP qlen 1000
    link/ether b8:27:eb:fc:46:15 brd ff:ff:ff:ff:ff:ff
    inet 192.168.2.22/24 brd 192.168.2.255 scope global eth0
```

Also check that the GPIO libraries are at least version 0.5.2a as shown in the section called *Update the GPIO libraries* on page 15.

Stream decode problems

The radio may display a message similar to the following:

```
ERROR: problems decoding http://173.244.194.212:8078
```

This is due to an invalid URL (In the above example this is <http://173.244.194.212:8078>) in one of the PLS files.

Locate the offending URL in the play list file in the `/var/lib/mpd/playlists` directory. Either correct the radio stream URL or remove it all together.

Also check that the file URL is not the pointer to the PLS file (See section Understanding PLS files on page 26.

Cannot mount remote network drive

There are just too many possibilities to cover all of these here. However a few common problems are covered here:

Error: mount error(115): Operation now in progress

Cause: Most likely an incorrect IP address

Error: NFS mount hangs

Cause: Most likely an incorrect IP address

Error: mount.nfs: access denied by server while mounting <ip address>:/music

Cause: The volume name is missing – for example /volume1/music

Error: mount error(16): Device or resource busy

Cause: The share mount directory is in use because a mount has already been done. Run the umount command.

Error: mount error(2): No such file or directory

Cause: The path specified in the mount doesn't exist

Error:

mount.nfs: rpc.statd is not running but is required for remote locking.

mount.nfs: Either use '-o nolock' to keep locks local, or start statd.

mount.nfs: an incorrect mount option was specified

Cause:

You need to include the “-o noclock” option.

If the error isn't in the above list then search the web for suggestions.

Configuring a wireless adaptor

You will almost certainly want to configure a wireless adaptor for the radio instead of a wired network connection. Choose a wireless adapter that has been approved for the Raspberry PI. See the following link for approved Raspberry PI peripherals:

http://elinux.org/RPi_VerifiedPeripherals

Install the wireless adapter

Switch off the Raspberry PI and plug in the adaptor into one of the USB ports. Power the PI back on and log in. Check to see if your Wireless Adapter has been recognised by running the **lsusb** command.

```
pi@raspberrypi:~$ lsusb
Bus 001 Device 002: ID 0424:9512 Standard Microsystems Corp.
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 001 Device 003: ID 0424:ec00 Standard Microsystems Corp.
Bus 001 Device 004: ID 148f:5370 Ralink Technology, Corp. RT5370 Wireless Adapter
```

The above shows a Ralink (Tenda) wireless adaptor but this will vary depending on the adapter that has been installed.

Configure the adaptor

The configuration is contained in the **/etc/network/interfaces** file as shown below

```
pi@raspberrypi:~$ cat /etc/network/interfaces
auto lo

iface lo inet loopback
iface eth0 inet dhcp

allow-hotplug wlan0
iface wlan0 inet manual
wpa-roam /etc/wpa_supplicant/wpa_supplicant.conf
iface default inet dhcp
```

You should not need to change this file. The file to be amended is shown on the line beginning with wpa-roam and is **/etc/wpa_supplicant/wpa_supplicant.conf**. Edit this file. It will only contain a couple of lines.

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
```

Add the following after the above lines:

```
network={
    ssid="YOUR_SSID"
    scan_ssid=1
    psk="YOUR_KEY"
    proto=RSN
    key_mgmt=WPA-PSK
    pairwise=CCMP
    auth_alg=OPEN
}
```

Substitute YOUR_SSID and YOUR_KEY with the actual SSID and key for your wireless router. The above configuration is for a router using WPA encryption. If your router is using the older WEP encryption then you will need to adapt the configuration to use WEP. See next section.

Explanation of the network fields

Field	Description
ssid	your wifi (SSID) name
scan_ssid	A value of 1 means broadcast and value of 2 means a hidden SSID (Normally enter a value of 1)
psk	Your WIFI password
proto	Your choice of RSN or WPA. RSN is WP2 and WPA is WPA1. (most configurations are RSN)
key_mgmt	Either WPA-PSK or WPA-EAP (pre-shared or enterprise respectively)
pairwise	Either CCMP or TKIP (WPA2 or WPA1 respectively)
auth_alg	OPEN option is required for WPA and WPA2 (other option, SHARED & LEAP)

Operating the wireless interface

If configured correctly the wireless adapter will start up when the Raspberry PI is rebooted.

The adaptor can be started and stopped with the following commands:

```
root@raspberrypi:/home/pi# ifup wlan0
```

and

```
root@raspberrypi:/home/pi# ifdown wlan0
```

To see what SSIDs are available run the iwlist command as shown in the following example:

```
root@raspberrypi:/home/pi# iwlist wlan0 scanning | grep ESSID
ESSID:"mywlan"
ESSID:"VGV751926F4B9"
ESSID:"prime"
ESSID:"Sitecom6A212C"
```

To display the IP address of the Wireless Adapter run the **ip addr** command:

```
root@raspberrypi:/home/pi# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether b8:27:eb:fc:46:15 brd ff:ff:ff:ff:ff:ff
    inet 192.168.2.11/24 brd 192.168.2.255 scope global eth0
3: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP qlen 1000
    link/ether c8:3a:35:c8:64:cd brd ff:ff:ff:ff:ff:ff
    inet 192.168.2.13/24 brd 192.168.2.255 scope global wlan0
```

Troubleshooting the wireless adapter

Problem – Starting the wireless adapter gives the following message:

```
root@raspberrypi:/home/pi# ifup wlan0
wpa_supplicant: /sbin/wpa_supplicant daemon failed to start
run-parts: /etc/network/if-pre-up.d/wpa_supplicant exited with return code 1
Failed to connect to wpa_supplicant - wpa_ctrl_open: No such file or
directory
wpa_supplicant: /sbin/wpa_cli daemon failed to start
run-parts: /etc/network/if-up.d/wpa_supplicant exited with return code 1
```

This is due to an incorrect `/etc/wpa_supplicant/wpa_supplicant.conf` file. The problem is due to an incorrect configuration. For example a space after the `ssid=` directive as shown below.

```
network={
    ssid= "homelan"
    scan_ssid=1
    psk="d762c954df"
    proto=RSN
    key_mgmt=WPA-PSK
    pairwise=CCMP
    auth_alg=OPEN
}
```

Solution: Correct the error and run the `ifup wlan0` command.

Problem: The following is seen:

```
root@raspberrypi:/home/pi# ifup wlan0
ifup: interface wlan0 already configured
```

Solution: This isn't actually an error. Just run the `ifdown wlan0` command and retry the `ifup wlan0` command. It should then work.

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.

See <http://www.gnu.org/licenses/#GPL> for further information on the GNU General Public License.

Technical support

Technical support is on a voluntary basis by e-mail only at bob@bobrathbone.com. Before asking for support, please first consult the troubleshooting section on page 33.

Be sure to provide the following information:

- What have you built (Adafruit or normal LCD) and which program you are running
- A clear description of the fault
- Is anything displayed on the LCD
- Switch on DEBUG logging as described on page 24, run the program and include the log file

Acknowledgements

My thanks to [Matt Hawkins](#) for the original LCD screen driver routines. It made the job of writing the *lcd_class.py* much easier.

Also Adafruit Industries for their excellent LCD plate and I2C code. See <http://www.adafruit.com>.

To contributors such as Alan Broad who supplied photos of the Lego example of the radio plus code contribution.

Glossary

I2C	Industry standard serial interface (Philips) using data and clock signals
CGI	Common Gate Interface – Executable Server Side scripts
CIFS	Common Internet File System
DHCP	Dynamic Host Configuration Protocol
GPIO	General Purpose IO (On the Raspberry PI)
LCD	Liquid Crystal Display
MPC	Command line client for MPD
NFS	Network File System
MPD	Music Player Daemon
PID	Process ID
PLS	MPEG Playlist File (as used by Winamp)
RSS	Really Simple Syndication – Web feed usually containing news items
SSID	An SSID is the public name of a wireless network.
URL	Universal Resource Locator (A link to a Web page for example)
USB	Universal Serial Bus
WEP	Wired Equivalent Privacy (WEP) is a security algorithm considered less secure than WPA
WPA	Wi-Fi Protected Access (WPA) and Wi-Fi Protected Access II (WPA2)