Using a USB Audio Device With a Raspberry Pi

computers.tutsplus.com/articles/using-a-usb-audio-device-with-a-raspberry-pi--mac-55876

Jordan Merrick

My previous tutorial on the Raspberry Pi, Using a Raspberry Pi as an AirPlay Receiver, made use of the device's built-in 3.5mm jack for audio output. This, however, produces sound that is mediocre at best and if you're wanting to use the Raspberry Pi for any amount of audio listening, it's going to have to produce a better quality of audio. We're not going to squeeze much more out of the built-in components so, in this tutorial, I'll explain how to set up and configure a cheap USB sound card as the Raspberry Pi's default audio output device, providing a better quality, and volume, audio.

Before You Begin

This tutorial will follow on from my previous tutorial, Using a Raspberry Pi as an AirPlay Receiver. It will be expected that you have completed this tutorial and your Raspberry Pi is currently able to act as an AirPlay receiver, albeit with its built-in audio output.

USB Audio Device

Back in the good old days, most audio devices needed their own set of drivers and software to work. This often proved very frustrating for end-users and developers alike.

The reason for this is that many devices from different manufacturers actually use the same *chipset*, the physical components that make up the device. This is what the drivers and software are for and you'll often find that a number of different audio devices from many manufacturers will use the exact same chipset, the only difference between them will usually just be the branding.

Nowadays, most modern operating systems and audio devices have no such requirements as most chipsets work to an accepted standard, and since most manufacturers will make use of the same small number of chipsets, providing widespread support is far easier. There are certainly a small number of more specialised audio devices that require additional software but any generic audio output device is usually just plug-and-play

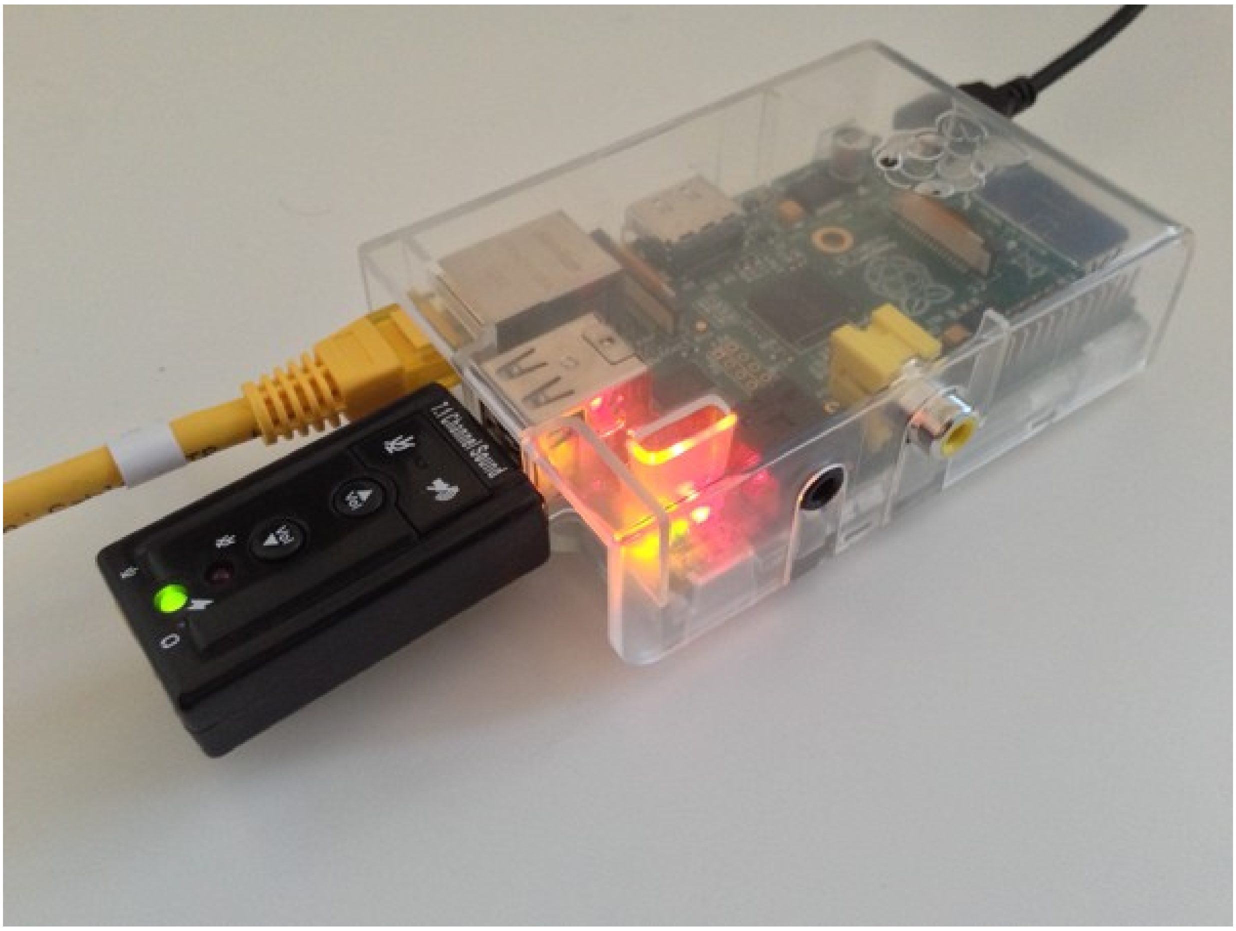
Compatibility

This is applicable not only to Mac OS X and Windows, but also Linux as well, with Raspbian being able to support devices via the AlsaProject, a set of widely compatible audio drivers for Linux.

Tip: Not all USB audio output devices are supported, though you can check the AlsaProject Compatibility List to ensure the device you're planning to use should be compatible.

What I Recommend

The USB audio device I will be using with my Raspberry Pi is a generic one I found on Amazon, costing just $2. It's called, quite vaguely, the 7.1 Channel USB External Sound Card Audio Adapter. The reason for using this particular USB audio device are as follows:



I've previously used this USB audio device with both Mac and PC. Not bad for something that costs less than a cup of coffee!

- It's cheap
- It doesn't need much power
- It's small enough to be easily used
- It's supported

This particular audio device uses the C-Media audio chipset, something that is supported by AlsaProject in Raspian.

When using any USB audio device, you'll need to make sure that your Raspberry Pi is being powered enough that it is able to support a USB-powered device. If you're running your Raspberry Pi from a mains adapter then you should be all set.

Tip: USB audio devices are usually a little larger than a memory stick and will probably take up more physical space, blocking access to the second USB port. If you're needing to use both USB ports at the same time, I would recommend using a USB extension cable and connecting your USB audio device that way, instead.

Setting Up

Now that I've covered the requirements for sourcing a suitable USB audio device, I will show you how to configure it.

The following steps will be performed within the command line on the Raspberry Pi. If you're familiar with SSH and know how to access your Raspberry Pi using it, you can perform the following steps without needing to be physically at your Raspberry Pi.

Step 1

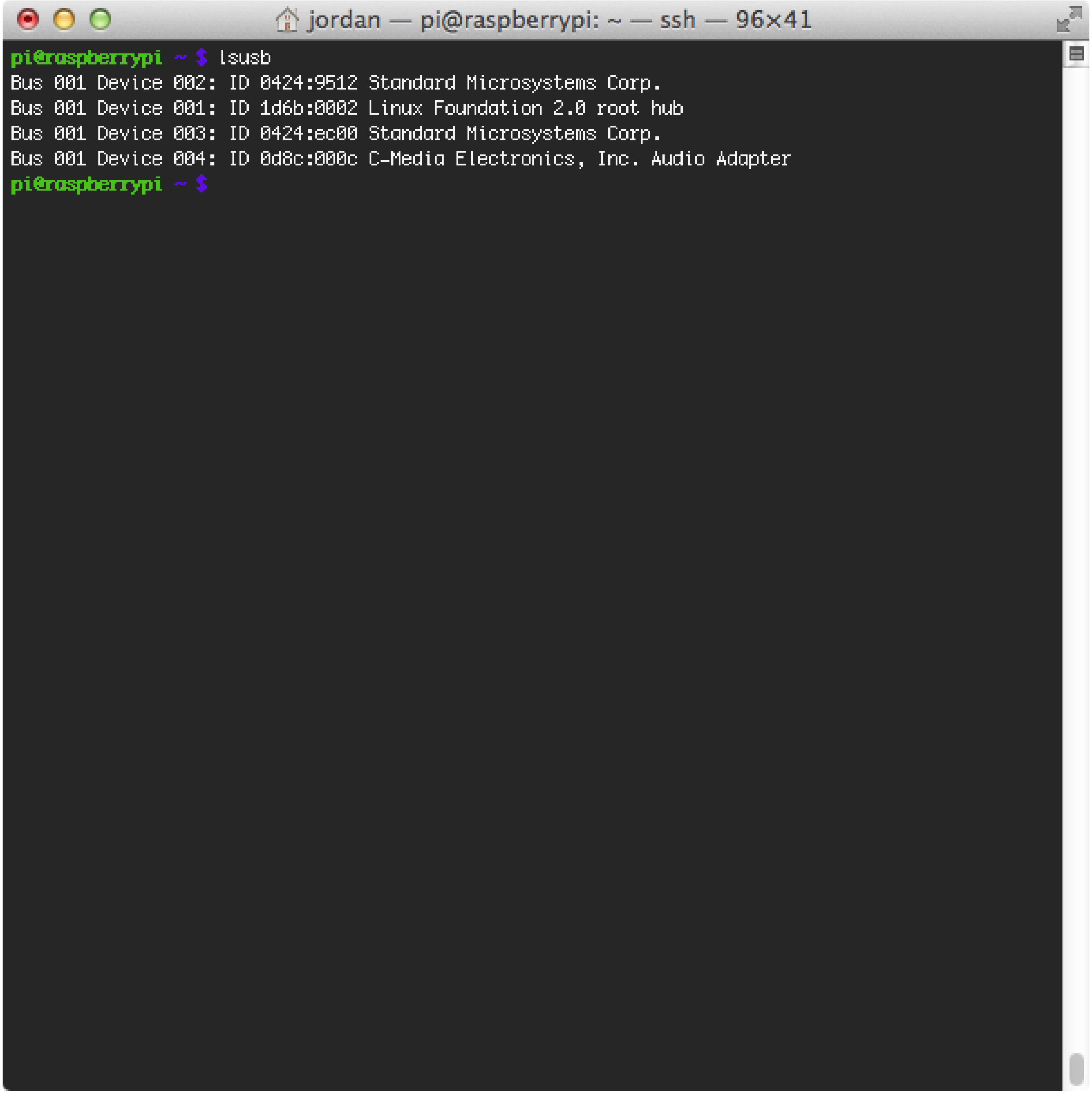
Before continuing, make sure your Raspberry Pi is switched off and then connect the USB audio device to its USB port. Remember, if you need access to both USB ports then use the USB audio device in conjunction with a USB extension cable, preventing the second port from being blocked.

Step 2

Power up your Raspberry Pi and, once it has booted, open the**LXTerminal** app. (Again, if you're familiar with SSH then you can simply connect and perform the remaining commands remotely).

First, I'll make sure that the USB audio device is being detected by both the hardware and software. Enter the following command and press enter:

```
$ lsusb
```



```
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 0d8c:000c C-Media Electronics, Inc. Audio Adapter
pi@raspberrypi ~ $
```

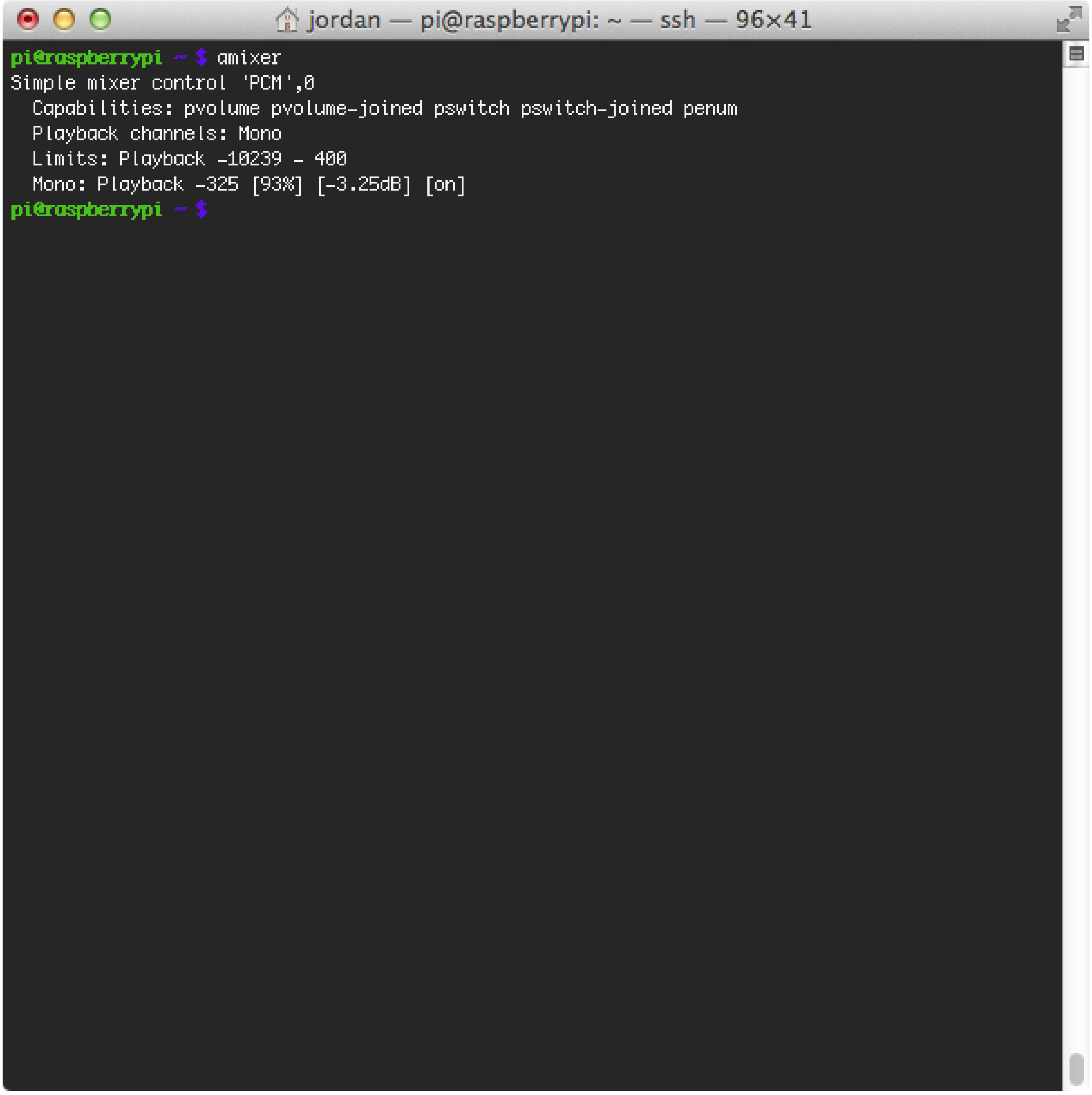
We can see what USB devices are connected and, hopefully, see the USB audio device.

This will display information regarding attached USB devices. As you can see, the last device listed in the screenshot above is the USB audio device labelled as **C-Media Electronics, Inc. Audio Adapter**. So far, so good.

Step 3

I will use a command to display the currently set audio device, which will still be the built-in audio. This command displays various information regarding the current audio device.

```
$ amixer
```



```
pi@raspberrypi ~ $ amixer
Simple mixer control 'PCM',0
  Capabilities: pvolume pvolume-joined pswitch pswitch-joined penum
  Playback channels: Mono
  Limits: Playback -10239 - 400
  Mono: Playback -325 [93%] [-3.25dB] [on]
pi@raspberrypi ~ $
```

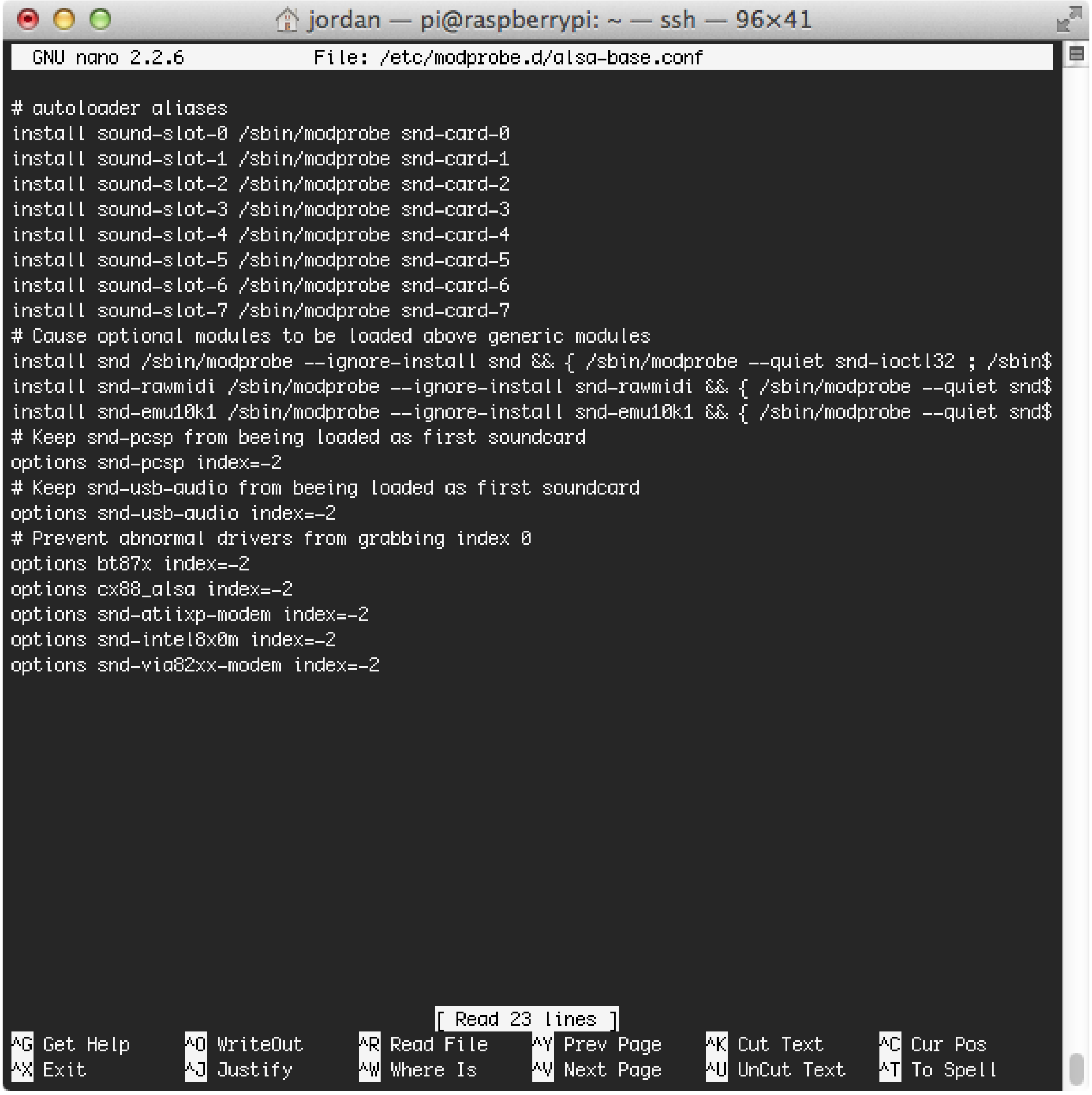
This command provides information about the designated audio device. By default, this will be the built-in audio output.

Step 4

I don't want to use the built-in audio device, instead I'd like to use the USB audio device I've connected. I need to edit a configuration file so that whenever the Raspberry Pi boots, it will default to using the USB audio device as the preferred method of audio output.

The configuration file that I need to edit is called **alsa-base.conf**. This controls how any attached audio devices are dealt with. Using this configuration file, I can set the order at which to use an attached audio device, making the USB audio device the first one to use and then fall back to the others if it isn't detected. To do this, enter:

```
$ sudo nano /etc/modprobe.d/alsa-base.conf
```



```
  GNU nano 2.2.6             File: /etc/modprobe.d/alsa-base.conf

# autoloader aliases
install sound-slot-0 /sbin/modprobe snd-card-0
install sound-slot-1 /sbin/modprobe snd-card-1
install sound-slot-2 /sbin/modprobe snd-card-2
install sound-slot-3 /sbin/modprobe snd-card-3
install sound-slot-4 /sbin/modprobe snd-card-4
install sound-slot-5 /sbin/modprobe snd-card-5
install sound-slot-6 /sbin/modprobe snd-card-6
install sound-slot-7 /sbin/modprobe snd-card-7
# Cause optional modules to be loaded above generic modules
install snd /sbin/modprobe --ignore-install snd && { /sbin/modprobe --quiet snd-ioctl32 ; /sbin$
install snd-rawmidi /sbin/modprobe --ignore-install snd-rawmidi && { /sbin/modprobe --quiet snd$
install snd-emu10k1 /sbin/modprobe --ignore-install snd-emu10k1 && { /sbin/modprobe --quiet snd$
# Keep snd-pcsp from beeing loaded as first soundcard
options snd-pcsp index=-2
# Keep snd-usb-audio from beeing loaded as first soundcard
options snd-usb-audio index=-2
# Prevent abnormal drivers from grabbing index 0
options bt87x index=-2
options cx88_alsa index=-2
options snd-atiixp-modem index=-2
options snd-intel8x0m index=-2
options snd-via82xx-modem index=-2

                                        [ Read 23 lines ]
^G Get Help     ^O WriteOut     ^R Read File    ^Y Prev Page    ^K Cut Text     ^C Cur Pos
^X Exit         ^J Justify      ^W Where Is     ^V Next Page    ^U UnCut Text   ^T To Spell
```

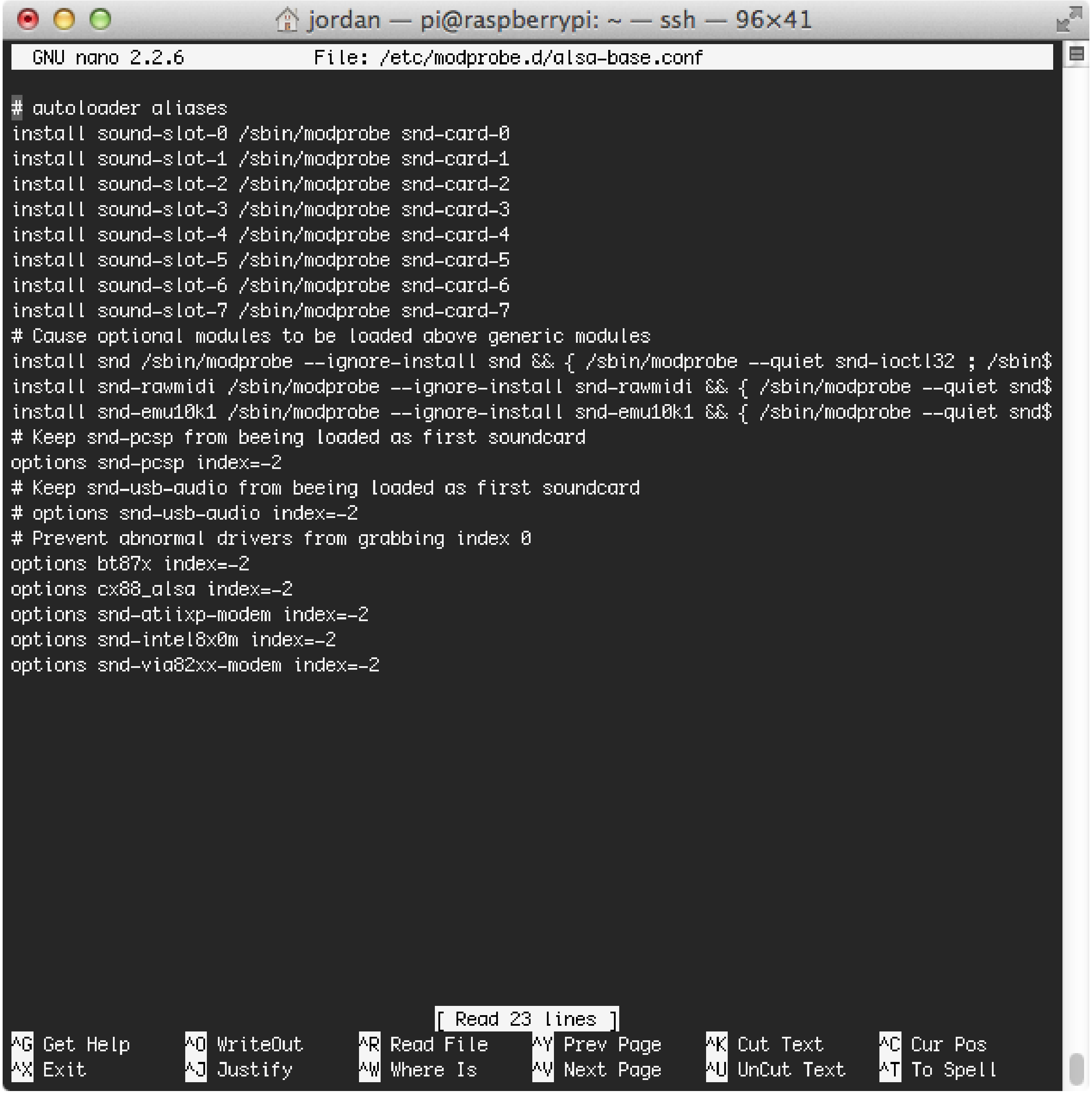
The configuration file denotes all the various settings for audio output. It is here we need to make a single change.

This will launch the text editor **nano** with the configuration file so I can edit it accordingly. It might look a little complicated or hard to understand, but all I need to do is add a single character. Look for the following line:

```
options snd-usb-audio index=-2
```

At the moment, this line causes any USB audio devices that are detected to be pushed down the order list and not be used by default. To override this, simply add a pound/hash symbol in front of the line so it reads like so:

```
# options snd-usb-audio index=-2
```



```
  GNU nano 2.2.6            File: /etc/modprobe.d/alsa-base.conf

# autoloader aliases
install sound-slot-0 /sbin/modprobe snd-card-0
install sound-slot-1 /sbin/modprobe snd-card-1
install sound-slot-2 /sbin/modprobe snd-card-2
install sound-slot-3 /sbin/modprobe snd-card-3
install sound-slot-4 /sbin/modprobe snd-card-4
install sound-slot-5 /sbin/modprobe snd-card-5
install sound-slot-6 /sbin/modprobe snd-card-6
install sound-slot-7 /sbin/modprobe snd-card-7
# Cause optional modules to be loaded above generic modules
install snd /sbin/modprobe --ignore-install snd && { /sbin/modprobe --quiet snd-ioctl32 ; /sbin$
install snd-rawmidi /sbin/modprobe --ignore-install snd-rawmidi && { /sbin/modprobe --quiet snd$
install snd-emu10k1 /sbin/modprobe --ignore-install snd-emu10k1 && { /sbin/modprobe --quiet snd$
# Keep snd-pcsp from beeing loaded as first soundcard
options snd-pcsp index=-2
# Keep snd-usb-audio from beeing loaded as first soundcard
# options snd-usb-audio index=-2
# Prevent abnormal drivers from grabbing index 0
options bt87x index=-2
options cx88_alsa index=-2
options snd-atiixp-modem index=-2
options snd-intel8x0m index=-2
options snd-via82xx-modem index=-2




                                        [ Read 23 lines ]
^G Get Help     ^O WriteOut     ^R Read File    ^Y Prev Page    ^K Cut Text     ^C Cur Pos
^X Exit         ^J Justify      ^W Where Is     ^V Next Page    ^U UnCut Text   ^T To Spell
```

Simply commenting out the line will allow the USB audio device to be the primary method of output.

Step 5

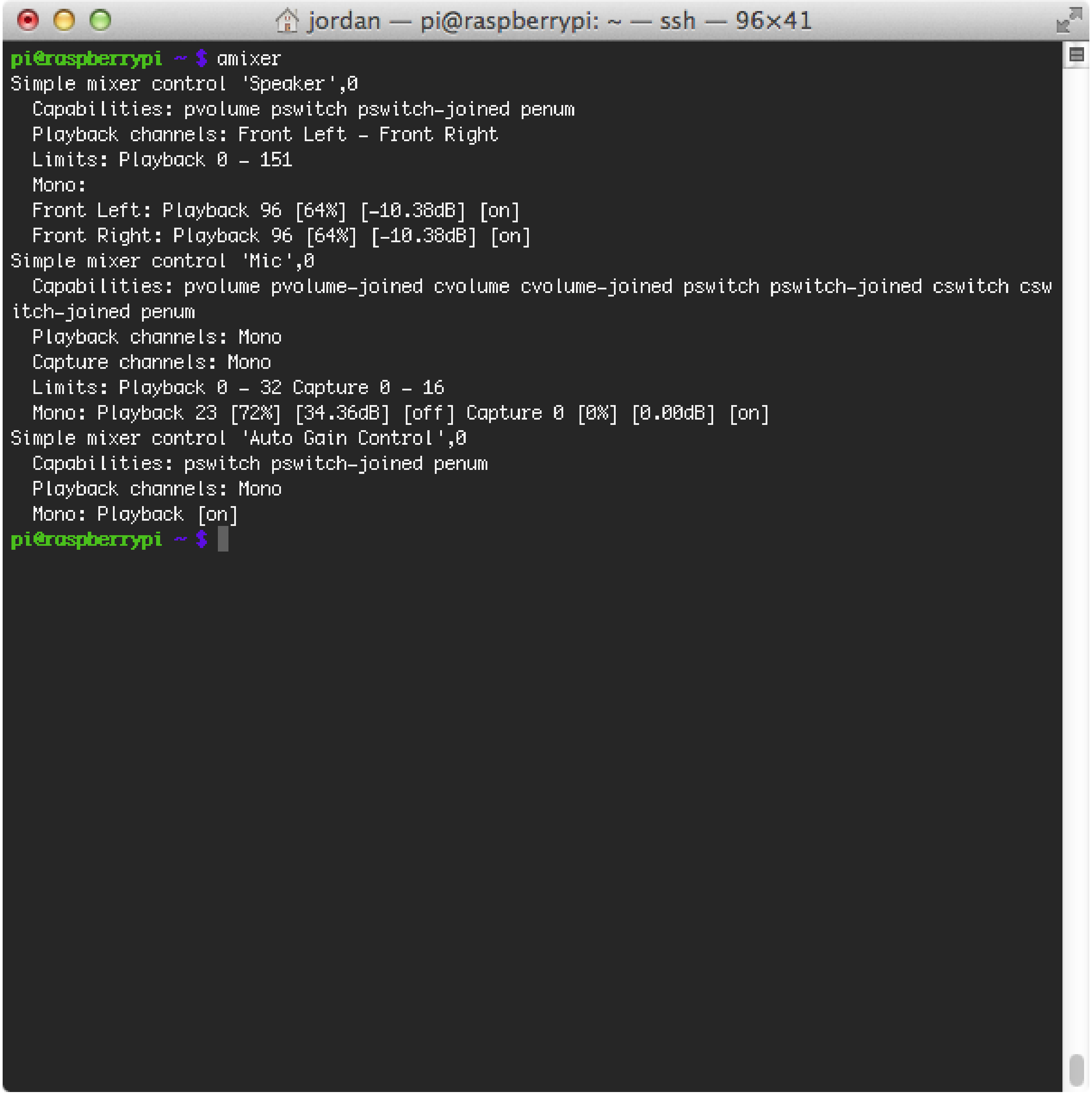
Press **Ctrl Z** and save the changes, then restart your Raspberry Pi by entering:

```
$ sudo reboot
```

Step 6

With the Raspberry Pi restarted, re-run the following command:

```
$ amixer
```



```
pi@raspberrypi ~ $ amixer
Simple mixer control 'Speaker',0
  Capabilities: pvolume pswitch pswitch-joined penum
  Playback channels: Front Left - Front Right
  Limits: Playback 0 - 151
  Mono:
  Front Left: Playback 96 [64%] [-10.38dB] [on]
  Front Right: Playback 96 [64%] [-10.38dB] [on]
Simple mixer control 'Mic',0
  Capabilities: pvolume pvolume-joined cvolume cvolume-joined pswitch pswitch-joined cswitch csw
itch-joined penum
  Playback channels: Mono
  Capture channels: Mono
  Limits: Playback 0 - 32 Capture 0 - 16
  Mono: Playback 23 [72%] [34.36dB] [off] Capture 0 [0%] [0.00dB] [on]
Simple mixer control 'Auto Gain Control',0
  Capabilities: pswitch pswitch-joined penum
  Playback channels: Mono
  Mono: Playback [on]
pi@raspberrypi ~ $
```

If all goes well, this will now display more information as it gathers it from the USB audio device.

Instead of the same information I was given earlier, I should now have a completely different set. You'll notice that I also have information regarding audio capture, further confirming I'm using the USB sound device as my default audio output option.

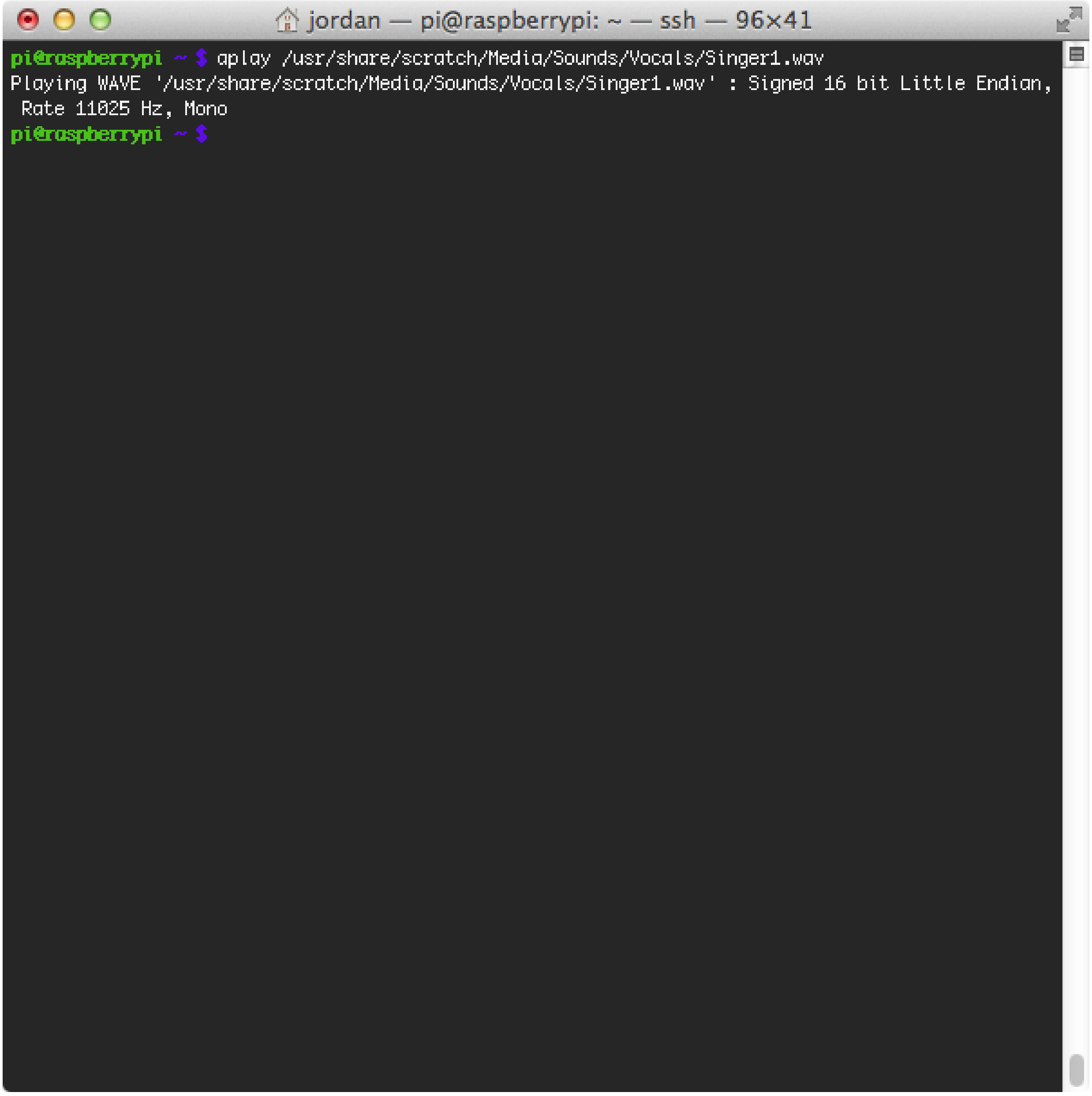
Oddly, it is still stated that the output is in mono. This can be ignored as it's actually not correct, the output will indeed be stereo, as you'll see (or hear) shortly.

Testing Audio

Now that I've set the USB audio device as the default audio output for the Raspberry Pi, it's time to test it to ensure it is working. Connect a set of speakers to the USB audio device and then start playing audio through them.

Raspian includes a number of test sound clips I can use to check that everything is working properly. Enter the following command:

```
$ aplay /usr/share/scratch/Media/Sounds/Vocals/Singer1.wav
```



```
pi@raspberrypi ~ $ aplay /usr/share/scratch/Media/Sounds/Vocals/Singer1.wav
Playing WAVE '/usr/share/scratch/Media/Sounds/Vocals/Singer1.wav' : Signed 16 bit Little Endian,
 Rate 11025 Hz, Mono
pi@raspberrypi ~ $
```

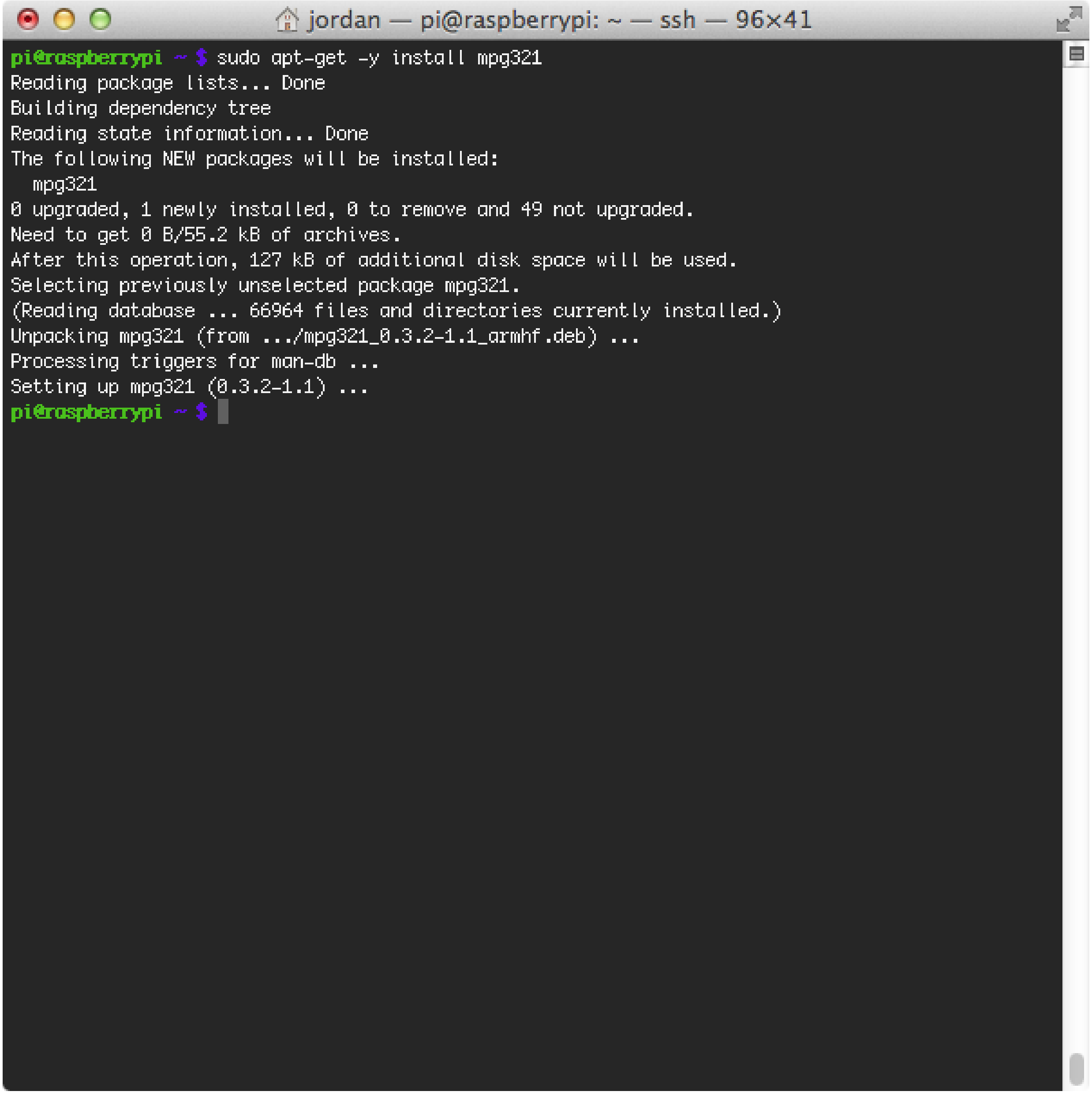
There are a number of included sound clips for testing purposes.

If everything went well, you should have heard a quick clip of an opera singer's voice. The clip is incredibly low quality and mono, so I'll doubly make sure everything is working and use a clip that is in stereo.

Step 1

I'm going to test this further by using an MP3 file, but I first need to install some software that's able to do this. I can use the utility **mpg321** to do this. Installing it is very straightforward, simply use:

```
$ sudo apt-get -y install mp321
```



```
pi@raspberrypi ~ $ sudo apt-get -y install mpg321
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  mpg321
0 upgraded, 1 newly installed, 0 to remove and 49 not upgraded.
Need to get 0 B/55.2 kB of archives.
After this operation, 127 kB of additional disk space will be used.
Selecting previously unselected package mpg321.
(Reading database ... 66964 files and directories currently installed.)
Unpacking mpg321 (from .../mpg321_0.3.2-1.1_armhf.deb) ...
Processing triggers for man-db ...
Setting up mpg321 (0.3.2-1.1) ...
pi@raspberrypi ~ $
```

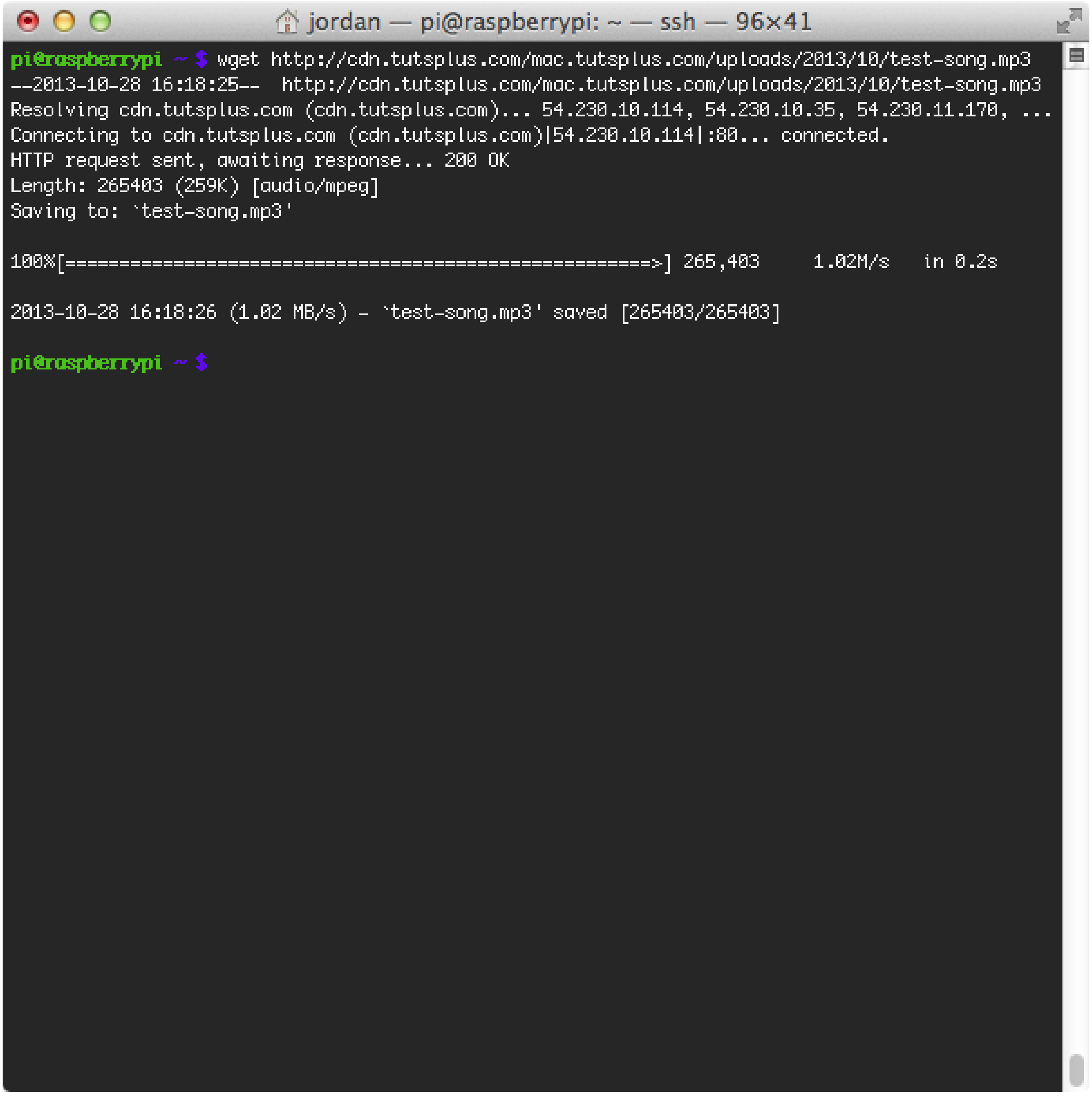
Additional software is required to play back MP3 files.

Raspian will automatically install the utility and any required additional bits and pieces it might need to work.

Step 2

I've created a small MP3 clip that pans from left to right, ensuring that any stereo audio will work correctly. To download it, enter:

```
$ wget https://cdn.tutsplus.com/mac.tutsplus.com/uploads/2013/10/test-song.mp3
```



```
pi@raspberrypi ~ $ wget http://cdn.tutsplus.com/mac.tutsplus.com/uploads/2013/10/test-song.mp3
--2013-10-28 16:18:25--  http://cdn.tutsplus.com/mac.tutsplus.com/uploads/2013/10/test-song.mp3
Resolving cdn.tutsplus.com (cdn.tutsplus.com)... 54.230.10.114, 54.230.10.35, 54.230.11.170, ...
Connecting to cdn.tutsplus.com (cdn.tutsplus.com)|54.230.10.114|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 265403 (259K) [audio/mpeg]
Saving to: `test-song.mp3'

100%[======================================================>] 265,403     1.02M/s   in 0.2s

2013-10-28 16:18:26 (1.02 MB/s) - `test-song.mp3' saved [265403/265403]

pi@raspberrypi ~ $
```

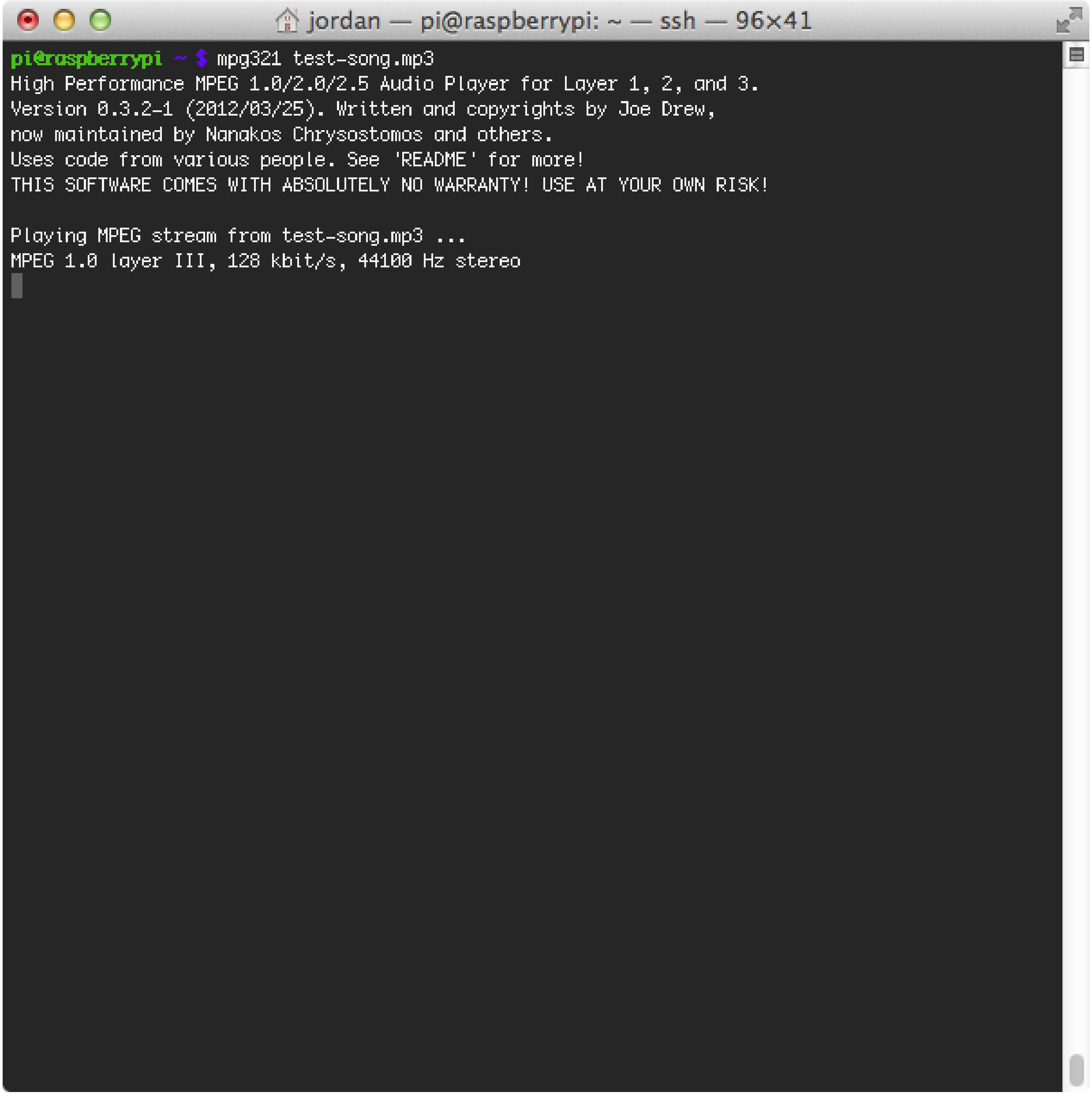
'wget' downloads files to your Raspberry Pi, so make sure an internet connection is available.

This is what it should sound like:

Step 3

After the song is downloaded, enter:

```
$ mpg321 test-song.mp3
```



```
pi@raspberrypi ~ $ mpg321 test-song.mp3
High Performance MPEG 1.0/2.0/2.5 Audio Player for Layer 1, 2, and 3.
Version 0.3.2-1 (2012/03/25). Written and copyrights by Joe Drew,
now maintained by Nanakos Chrysostomos and others.
Uses code from various people. See 'README' for more!
THIS SOFTWARE COMES WITH ABSOLUTELY NO WARRANTY! USE AT YOUR OWN RISK!

Playing MPEG stream from test-song.mp3 ...
MPEG 1.0 layer III, 128 kbit/s, 44100 Hz stereo
```

Some information is displayed during playback.

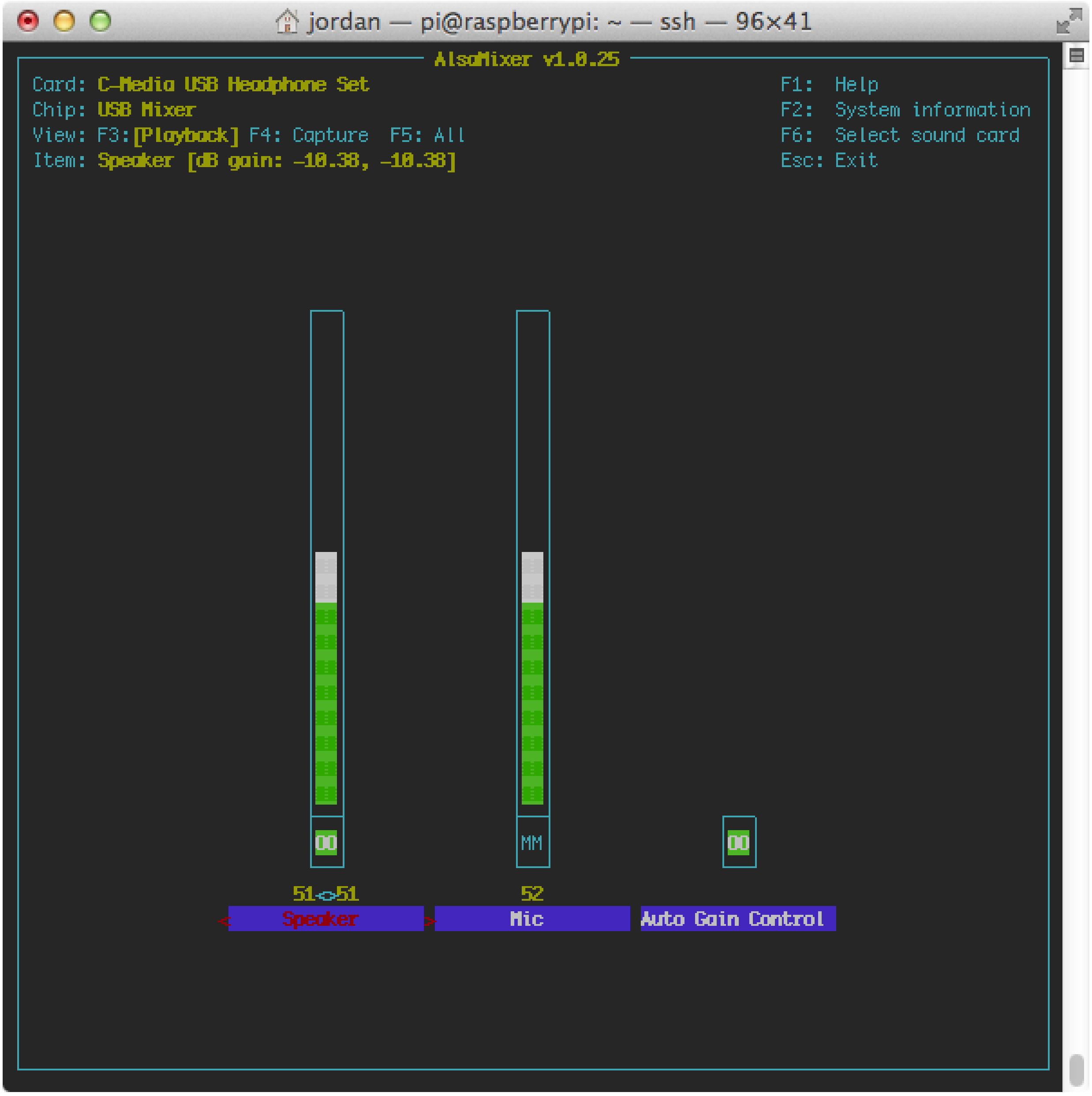
You should hear a short music clip that pans audio from left to right.

Adjusting Volume

Should the audio output be too loud or quiet, we can adjust this using another utility, called **alsamixer**.

To start it, simply enter the name in the command line, like so:

```
$ alsamixer
```



```
──────────────────────────── AlsaMixer v1.0.25 ────────────────────────────
 Card: C-Media USB Headphone Set                          F1:  Help
 Chip: USB Mixer                                          F2:  System information
 View: F3:[Playback] F4: Capture  F5: All                 F6:  Select sound card
 Item: Speaker [dB gain: -10.38, -10.38]                  Esc: Exit




                    OO              MM              OO

                  51<>51            52
             <    Speaker    >      Mic         Auto Gain Control
```

'alsamixer' controls some functions of the audio output, mainly volume.

This presents a more graphical view of the volume and information regarding the USB audio device. Using the arrow keys on your keyboard, select the volume column and adjust the volume higher or lower, dependent on your needs. Where possible, keep the volume level below 80–90% to avoid any distortion.

Wrapping Up

Using a dedicated USB audio device will mean better audio quality and more control, as well as better volume as many devices are able to output far better than the Raspberry Pi's built-in audio.

By the end of this tutorial, you should now be able to configure a new USB audio device and set it as the Raspberry Pi's default audio output option when using Raspian. If you were using your Raspberry Pi as an AirPlay receiver, you don't need to reconfigure any settings, simply start the AirPlay service back up and audio will be outputted to the USB audio device automatically.