

L



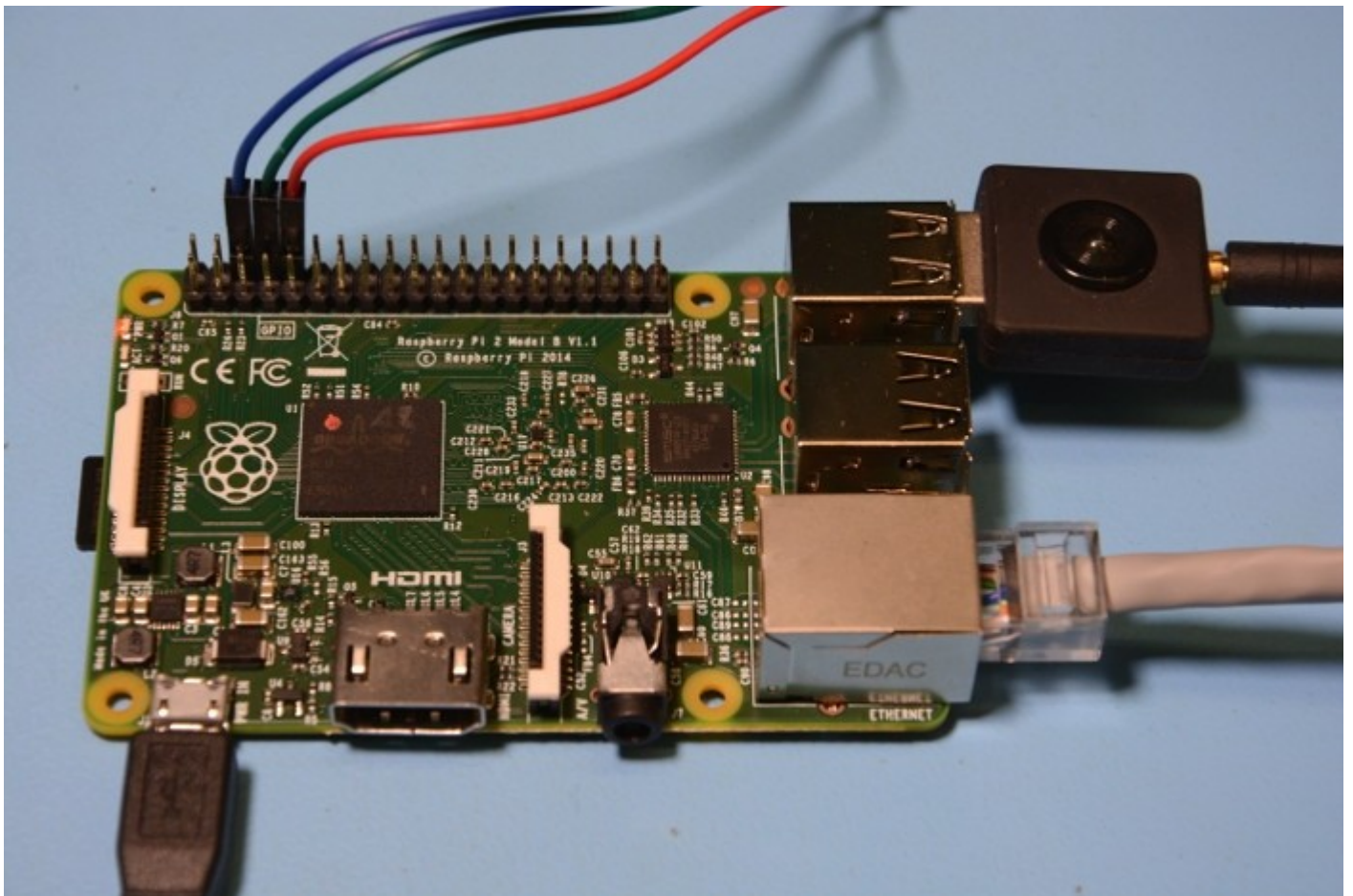
Taking the Raspberry Pi 2 for a Test Drive with GNU Radio



Posted by [Andrew Back](#) on Mon, Feb 02 2015 09:27:00

[Follow](#)

51150 views



Installing GNU Radio and receiving aircraft radar with a USB TV tuner

The Raspberry Pi has been put to countless creative uses, with it's low cost and easy to use GPIO, coupled with a passionate and inventive community, giving birth to applications that have ranged from simple fun, to inspired and even profound.

However, the single-core Raspberry Pi couldn't be everything to everyone and a common wish was for just a little more horsepower, for those applications that really need it. A great example being software-defined radio (SDR), where components usually implemented in hardware are instead implemented in software. Resulting in flexibility, but at the cost of being computationally intensive.

The [Raspberry Pi 2](#) Model B, with it's quad-core processor and 1GB RAM, weighs in at around six times the performance of its predecessor and should *far* better accommodate SDR applications.

```
pi@raspberrypi:~$ cat /proc/cpuinfo
processor       : 0
model name     : ARMv7 Processor rev 5 (v7l)
Features      : half thumb fastmult vfp edsp vfpv3 tls vfpv4 idiva idivt vfpd32 lpae evtstrm
CPU implementer : 0x41
CPU architecture: 7
CPU variant    : 0x0
CPU part      : 0xc07
CPU revision   : 5

processor       : 1
model name     : ARMv7 Processor rev 5 (v7l)
Features      : half thumb fastmult vfp edsp vfpv3 tls vfpv4 idiva idivt vfpd32 lpae evtstrm
CPU implementer : 0x41
CPU architecture: 7
CPU variant    : 0x0
CPU part      : 0xc07
CPU revision   : 5

processor       : 2
model name     : ARMv7 Processor rev 5 (v7l)
Features      : half thumb fastmult vfp edsp vfpv3 tls vfpv4 idiva idivt vfpd32 lpae evtstrm
CPU implementer : 0x41
CPU architecture: 7
CPU variant    : 0x0
CPU part      : 0xc07
CPU revision   : 5

processor       : 3
model name     : ARMv7 Processor rev 5 (v7l)
Features      : half thumb fastmult vfp edsp vfpv3 tls vfpv4 idiva idivt vfpd32 lpae evtstrm
CPU implementer : 0x41
CPU architecture: 7
CPU variant    : 0x0
CPU part      : 0xc07
CPU revision   : 5

Hardware      : BCM2708
Revision     : 0010
Serial       : 00000000b3251f45
pi@raspberrypi:~$
```

CTRL-A Z for help	115200 8N1	APP	Minicom 2.6.1	VT102	Offline
-------------------	------------	-----	---------------	-------	---------

Installing GNU Radio

The [GNU Radio](#) SDR toolkit is a fairly substantial codebase and with some equally heavyweight dependencies. Thankfully, Raspbian packages are available — not in the current “wheezy” release , but in the “jessie” testing release. The MicroSD card that came supplied with the Pi 2 was based on wheezy, but adding a line to the Apt configuration was all that was needed to get jessie packages.

Edit /etc/apt/sources.list and add the line:

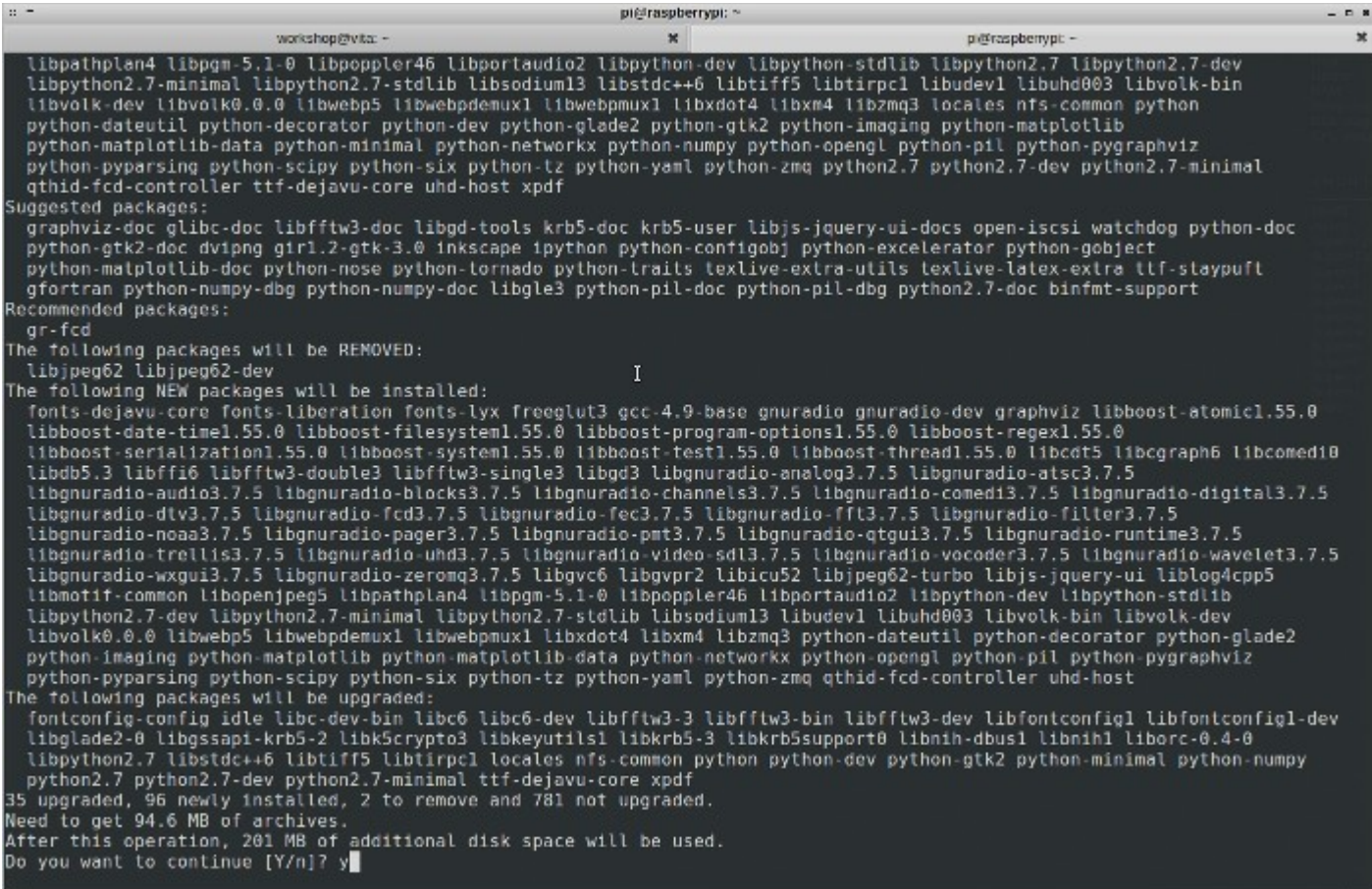
```
deb http://archive.raspbian.org/raspbian jessie main
```

Update the Apt cache:

```
$ sudo apt-get update
```

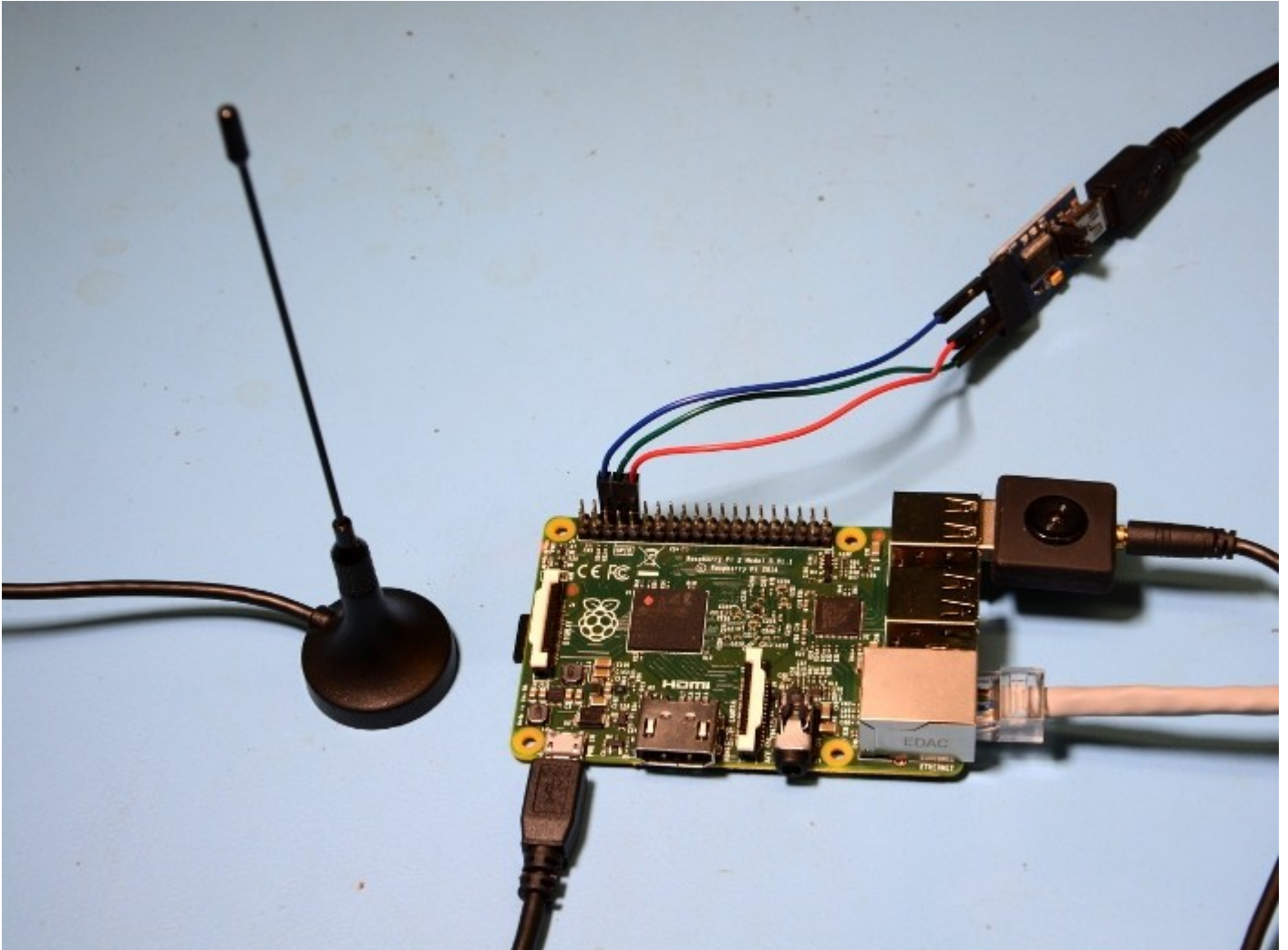
Install GNU Radio runtime and development files:

```
$ sudo apt-get install gnuradio gnuradio-dev
```



```
pi@raspberrypi: ~  
libpathplan4 libpgm-5.1-0 libpoppler46 libportaudio2 libpython-dev libpython-stdlib libpython2.7 libpython2.7-dev  
libpython2.7-minimal libpython2.7-stdlib libsodium13 libstdc++6 libtiff5 libtirpc1 libudev1 libuhd003 libvolk-bin  
libvolk-dev libvolk0.0 libwebp5 libwebpdemux1 libwebpmux1 libxdot4 libxm4 libzmq3 locales nfs-common python  
python-dateutil python-decorator python-dev python-glade2 python-gtk2 python-imagimg python-matplotlib  
python-matplotlib-data python-minimal python-networkx python-numpy python-opengl python-pil python-pygraphviz  
python-pyparsing python-scipy python-six python-tz python-yaml python-zmq python2.7 python2.7-dev python2.7-minimal  
qthid-fcd-controller ttf-dejavu-core uhd-host xpdf  
Suggested packages:  
graphviz-doc glibc-doc libfftw3-doc libgd-tools krb5-doc krb5-user libjs-jquery-ui-docs open-iscsi watchdog python-doc  
python-gtk2-doc dwipng glib1.2-gtk-3.0 inkscape ipython python-configobj python-excelerator python-gobject  
python-matplotlib-doc python-nose python-tornado python-traits texlive-extra-utils texlive-latex-extra ttf-staypuft  
gfortran python-numpy-dbg python-numpy-doc libgle3 python-pil-doc python-pil-dbg python2.7-doc binfmt-support  
Recommended packages:  
gr-fcd  
The following packages will be REMOVED:  
libjpeg62 libjpeg62-dev  
The following NEW packages will be installed:  
fonts-dejavu-core fonts-liberation fonts-lyx freetype3 gcc-4.9-base gnuradio gnuradio-dev graphviz libboost-atomic1.55.0  
libboost-date-time1.55.0 libboost-filesystem1.55.0 libboost-program-options1.55.0 libboost-regex1.55.0  
libboost-serialization1.55.0 libboost-system1.55.0 libboost-test1.55.0 libboost-thread1.55.0 libcdt5 libcgraph6 libcomedi0  
libdb5.3 libffi6 libfftw3-double3 libfftw3-single3 libgd3 libgnuradio-analog3.7.5 libgnuradio-atsc3.7.5  
libgnuradio-audio3.7.5 libgnuradio-blocks3.7.5 libgnuradio-channels3.7.5 libgnuradio-comedi3.7.5 libgnuradio-digital3.7.5  
libgnuradio-dtv3.7.5 libgnuradio-fcd3.7.5 libgnuradio-fec3.7.5 libgnuradio-fft3.7.5 libgnuradio-filter3.7.5  
libgnuradio-noaa3.7.5 libgnuradio-pager3.7.5 libgnuradio-pmt3.7.5 libgnuradio-gtgui3.7.5 libgnuradio-runtime3.7.5  
libgnuradio-trellis3.7.5 libgnuradio-uhd3.7.5 libgnuradio-video-sdl3.7.5 libgnuradio-vocoder3.7.5 libgnuradio-wavelet3.7.5  
libgnuradio-wxgui3.7.5 libgnuradio-zeroMQ3.7.5 libgvc6 libgvpr2 libicu52 libjpeg62-turbo libjs-jquery-ui liblog4cpp5  
libmotif-common libopenjpeg5 libpathplan4 libpgm-5.1-0 libpoppler46 libportaudio2 libpython-dev libpython-stdlib  
libpython2.7-dev libpython2.7-minimal libpython2.7-stdlib libsodium13 libudev1 libuhd003 libvolk-bin libvolk-dev  
libvolk0.0 libwebp5 libwebpdemux1 libwebpmux1 libxdot4 libxm4 libzmq3 python-dateutil python-decorator python-glade2  
python-imagimg python-matplotlib python-matplotlib-data python-networkx python-opengl python-pil python-pygraphviz  
python-pyparsing python-scipy python-six python-tz python-yaml python-zmq qthid-fcd-controller uhd-host  
The following packages will be upgraded:  
fontconfig-config idle libc-dev-bin libc6 libc6-dev libfftw3-3 libfftw3-bin libfftw3-dev libfontconfig1 libfontconfig1-dev  
libglade2-0 libgssapi-krb5-2 libk5crypto3 libkeyutils1 libkrb5-3 libkrb5support0 libnib-dbus1 libnib1 liborc-0.4-0  
libpython2.7 libstdc++6 libtiff5 libtirpc1 locales nfs-common python python-dev python-gtk2 python-minimal python-numpy  
python2.7 python2.7-dev python2.7-minimal ttf-dejavu-core xpdf  
35 upgraded, 96 newly installed, 2 to remove and 781 not upgraded.  
Need to get 94.6 MB of archives.  
After this operation, 201 MB of additional disk space will be used.  
Do you want to continue [Y/n]? y
```

Setting up RTL-SDR



It still never ceases to amaze me what can be done with a humble USB TV tuner dongle that can be picked up for around £10 plus open source SDR software. Above can be seen the Pi 2 with such a tuner plugged in to one of the USB ports, and the supplied antenna attached. For more information on the wonder that is [rtl-sdr](#), see the [post I wrote about this back in 2012](#).

Since we're re-purposing a TV tuner that is supported by the Linux kernel and which would otherwise be claimed by it and for TV reception, we need to first stop the kernel from doing so.

Edit the file `/etc/modprobe.d/raspi-blacklist.conf` and add the line:

```
blacklist dvb_usb_rtl28xxu
```

Install the rtl-sdr software and GNU Radio support:

```
$ sudo apt-get install rtl-sdr gr-osmosdr
```

In order to access the device as a non-root user we need to set up a new udev rule, but first we need to

ascertain the USB ID. Ensure that the tuner is plugged in and type:

```
$ lsusb
```

This gave me:

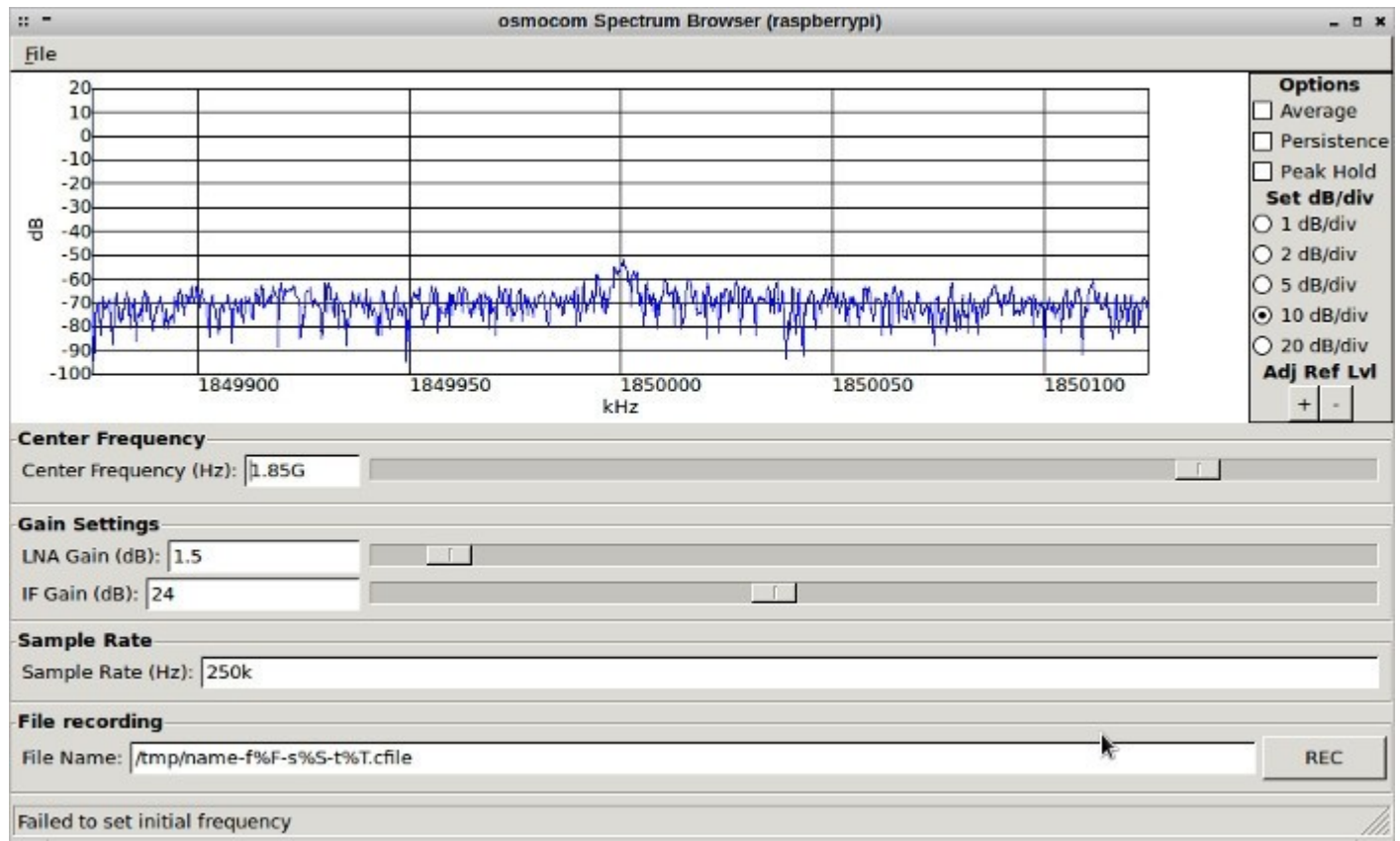
```
Bus 001 Device 004: ID 0bda:2832 Realtek Semiconductor Corp. RTL2832U DVB-T
```

Next we create the file `/etc/udev/rules.d/20.rtlsdr.rules`, with the line:

```
SUBSYSTEM=="usb", ATTRS{idVendor}=="0bda", ATTRS{idProduct}=="2832", GROUP="adm",  
MODE="0666", SYMLINK+="rtl_sdr"
```

We could restart udev at this point, but since we also blacklisted a kernel module it's probably just easiest to reboot.

A simple test



To get a simple spectrum display we can run the FFT application which is provided as part of the `osmocom` software.

```
$ osmocom_fft
```

If we then check the CPU load we can see that we have plenty of capacity to spare, with just one core at around 70% utilisation.

```
top - 14:21:22 up 14 min, 3 users, load average: 1.08, 0.69, 0.40
Tasks: 83 total, 2 running, 81 sleeping, 0 stopped, 0 zombie
%Cpu(s): 17.2 us, 1.3 sy, 0.0 ni, 81.3 id, 0.0 wa, 0.0 hi, 0.2 si, 0.0 st
KiB Mem: 764060 total, 203720 used, 560340 free, 17100 buffers
KiB Swap: 102396 total, 0 used, 102396 free, 102792 cached
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
2189	pi	20	0	286m	99m	53m	R	70.4	13.4	3:54.14	osmocom_fft
2103	pi	20	0	10176	3488	2856	S	3.3	0.5	0:22.10	sshd
2185	pi	20	0	4948	2464	2100	R	0.7	0.3	0:03.64	top
43	root	20	0	0	0	0	S	0.3	0.0	0:00.54	kworker/0:1
1	root	20	0	2248	1332	1232	S	0.0	0.2	0:01.69	init

gr-air-modes

Around 2 ½ years ago I [wrote about](#) how you could use rtl-sdr hardware together with the GNU Radio-based gr-air-modes software, to receive position and heading information from aircraft Mode-S transponders. At the time I used a laptop, and did try using a Raspberry Pi Model B also, but this didn't have quite enough processing power and resulted in buffer underruns.

In order to build gr-air-modes a few additional dependencies are required.

```
$ sudo apt-get install sqlite pyqt4-dev-tools liblog4cpp5-dev swig
```

With these installed the sources can be cloned from GitHub:

```
$ git clone https://github.com/bistromath/gr-air-modes.git
```

To then build and install:

```
$ cd gr-air-modes
```

```
$ mkdir build
```

```
$ cd build
```

```
$ cmake ../
```

```
$ make
```

```
$ sudo make install
```

```
$ sudo ldconfig
```

We can then run the application with:

```
$ modes_rx -s osmocom
```

And with only a tiny antenna and a good number of miles from the nearest airport, I still managed to get no shortage of output!

```
pi@raspberrypi ~ $
pi@raspberrypi ~ $ modes_rx -s osmocomb
linux; GNU C++ version 4.9.1; Boost_105500; UHD_003.007.003-0-unknown

gr-osmosdr 0.1.3 (0.1.3) gnuradio 3.7.5
built-in source types: file osmosdr fcd rtl rtl_tcp uhd miri hackrf bladerf rfspace airsipy
Using device #0 Realtek RTL2832U SN: 00000991
Found Elonics E4000 tuner
Invalid sample rate: 4000000 Hz
Gain is 34
Rate is 4000000
Using Volk machine: generic_orc
(-40 0.00000000) No handler for message type 24 from 5b44c1
(-36 0.00000000) Type 4 (short surveillance altitude reply) from 5125f at 87700ft (GROUND ALERT)
(-36 0.00000000) Type 0 (short A-A surveillance) from 648693 at 200ft (speed <75kt)
(-37 0.00000000) Type 4 (short surveillance altitude reply) from 4ee766 at 38300ft (AIRBORNE ALERT)
(-39 0.00000000) Type 4 (short surveillance altitude reply) from 4d3bf at 125500ft (AIRBORNE ALERT)
(-41 0.00000000) No handler for message type 24 from 4f5d0d
(-36 0.00000000) Type 0 (short A-A surveillance) from 4d6e4a at 12400ft (No TCAS)
(-38 0.00000000) Type 5 (short surveillance ident reply) from 51ebf with ident 242 (AIRBORNE ALERT)
(-35 0.00000000) Type 5 (short surveillance ident reply) from albd2c with ident 134 (aircraft is on the ground)
(-35 0.00000000) Type 4 (short surveillance altitude reply) from 330fce at 3425ft (SPI ALERT)
(-38 0.00000000) Type 0 (short A-A surveillance) from a54ae6 at 31500ft (speed 75-150kt)
(-37 0.00000000) Type 5 (short surveillance ident reply) from 9fcfc1 with ident 1300 (AIRBORNE ALERT)
(-40 0.00000000) Type 4 (short surveillance altitude reply) from c966c8 at 1700ft (SPI)
(-40 0.00000000) Type 0 (short A-A surveillance) from 43c86d at 53400ft (No TCAS)
(-41 0.00000000) No handler for message type 24 from c02d66
(-40 0.00000000) Type 5 (short surveillance ident reply) from a79fe0 with ident 6108 (SPI)
(-35 0.00000000) Type 0 (short A-A surveillance) from ae43d7 at 98800ft (TCAS resolution inhibited)
```

Not to mention, once again with plenty of headroom to spare.

```
top - 09:28:22 up 18:23, 3 users, load average: 0.50, 0.37, 0.19
Tasks: 83 total, 1 running, 82 sleeping, 0 stopped, 0 zombie
%Cpu(s): 5.4 us, 1.2 sy, 0.0 ni, 92.4 id, 0.9 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem: 764060 total, 318912 used, 445148 free, 27420 buffers
KiB Swap: 102396 total, 0 used, 102396 free, 207648 cached
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
9067	pi	20	0	282m	80m	40m	S	25.2	10.7	0:08.57	modes_rx
9086	pi	20	0	4948	2524	2156	R	0.7	0.3	0:00.14	top
57	root	20	0	0	0	0	S	0.3	0.0	0:00.25	kworker/2:1
1	root	20	0	2556	1364	1256	S	0.0	0.2	0:02.40	init
2	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kthreadd
3	root	20	0	0	0	0	S	0.0	0.0	0:00.09	ksoftirqd/0
5	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	kworker/0:0H

Conclusion

The [Pi 2](#) provides a marked improvement on the first generation hardware, with four cores instead of one and each of these being a more recent and more powerful version of the ARM architecture. A performance improvement that will be welcomed by many and not least of all those with SDR applications in mind.

— [Andrew Back](#)

[BUY A Pi2](#)

[Follow](#)

13 comments

[Like this](#)

- [Raspberry Pi](#)
- [Development Kits - Electronics](#)
- [Communication Applications](#)

Comments

- [andrew back](#)

Posted by [andrew back](#) at 12:13 on 23/04/2015

Hi Peter,

Well, as I think you know there are performance issues with the GNU Radio GUI applications built on WX, and I get frozen controls when I run these up on the Novena also. However, I do wonder whether, in addition to which there are some specific performance issues with GNU Radio on ARM (and which cannot be explained by simply the relatively limited resources compared with, say, an Intel/AMD desktop class machine).

Regards,

Andrew

- [pebol](#)

Posted by [pebol](#) at 21:11 on 03/03/2015

Thanks for this how-to. Both osmocom_fft and gr-air-modes work fine. A small grc test circuit consisting of an Audio Source, a Audio Sink, a FFT Sink and a slider to control the frequency also does what it should.

With a similar circuit consisting of a RTL SDR Source, a FFT Sink and a slider the slider is shown but can't be moved. I also can't type in the frequency and none of the FFT sink's buttons can be pressed. If I minimize the top block window and resize it afterwards the slider and all the buttons are gone. CPU load is approx. 35 %, 300 MB RAM is used.

A RTL SDR based WBFM receiver works, again with inactive buttons and sliders.

What could be wrong?

Peter

- [fotografale](#)

Posted by [fotografale](#) at 08:22 on 25/02/2015

That's very interesting!

And please, does anyone has already tested the behaviour of the GPIO4 pin, with the new Raspberry Pi 2 version?

I'm a supporter of the following ham radio applications.

<http://youtu.be/-ONIrqeE63Y>



- [andrew back](#)

Posted by [andrew back](#) at 11:04 on 18/02/2015

@pirat3uk GPIO connection not required — I just prefer to use a serial console with such boards, rather than a keyboard, mouse and monitor. Of course, most of the time you don't need either, provided you can ascertain the DHCP address and SSH in.

- [pirat3uk](#)

Posted by [pirat3uk](#) at 09:35 on 18/02/2015

really interested by all this since having play using an old laptop a while ago, cant seem to find any reference to the GPIO connection in the picture? (what is it?)

Thanks

- [ptamike](#)

Posted by [ptamike](#) at 17:12 on 17/02/2015

Apologies - brain and mouth not in gear today!

Adding the following to /boot/config.txt is the fix for the X-Windows Badmatch problem.

Thanks to G0UKB :-)

```
framebuffer_depth=32
```

```
framebuffer_ignore_alpha=1
```

- [ptamike](#)

Posted by [ptamike](#) at 16:09 on 17/02/2015

Has anyone cracked the X-Windows Badmatch problem yet?

- [g0ukb](#)

Posted by [g0ukb](#) at 10:19 on 13/02/2015

FIXED! Needed to add :

```
framebuffer_depth=32  
framebuffer_ignore_alpha=1
```

to /boot/config.txt

- [g0ukb](#)

Posted by [g0ukb](#) at 21:07 on 12/02/2015

Well, doing a full release upgrade to jessie hasn't solved the X Badmatch. On the plus side everything else is working and getting plenty of output from gr-air-modes. For anyone else going down this route I had a pretty vanilla wheezy Raspian to start with and the only things I needed to do were to add cmake and libboost-dev to the list of dependencies to build gr-air-modes.

- [g0ukb](#)

Posted by [g0ukb](#) at 18:10 on 12/02/2015

Great instructions - alas for me osmocom-fft is failing with an X-Windows Badmatch error. The FFT screen appears briefly so looks like all is OK with the RTL_SDR setup.

More stuff to go investigate - thanks for the instructions and hopefully when I crack this I'll be up and running.



- [andrew back](#)

Posted by [andrew back](#) at 20:23 on 02/02/2015

Thanks, Peter!

Frequency range depends on which particular tuner IC your dongle uses. The rtl-sdr wiki lists a few:

<http://sdr.osmocom.org/trac/wiki/rtl-sdr>

It's also possible to bypass the tuner (hardware mod) and run the RTL2832U IC in direct sampling mode, so that you can receive HF. However, I gather you really need to apply ESD protection and filtering. All doable given a bit of time and patience.

- [peterjfrancis](#)

Posted by [peterjfrancis](#) at 15:24 on 02/02/2015

Great Post Andrew !

Can't wait to get a Raspberry Pi 2 setup in my new 'Man Cave' when it is completed

What is the lowest practical frequency the USB TV dongle will receive ?

- [siddly](#)

Posted by [siddly](#) at 13:19 on 02/02/2015

Another application is ghpsdr3-alex at napan.ca/ghpsdr3/index.php/Main_Page. Building is quite involved but the wiki steps if followed bullet by bullet anyone can build it.

It supports many SDR's including RTL-SDR and can be made accessible on the internet to its QtRadio application running on a PC or ARM board or glSDR from Google Play store running on an Android phone or tablet.

I had a dongle up on the internet for a number of weeks with a 2m/70cms co-linear antenna above the roof.

I built an HF upconverter to allow it to receive the HF bands, but have not yet got around to testing HF - too many other more urgent SDR projects on the go.