
HackRF One

新手使用手册

简介版

开源 sdr

淘宝店铺名称：开源 SDR 实验室

Content

1. HackRF 软件环境搭建	2
1.1 源码安装 HackRF 的 host	2
1.1.1 安装依赖	2
1.1.2 下载 host 源码	2
1.1.3 编译安装	2
1.1.4 测试 host 安装是否成功	3
1.2 源码安装 Gnuradio	4
1.2.1 安装依赖	4
1.2.2 修改 Python PATH	6
1.2.3 下载 GNU Radio 源码	6
1.2.4 编译安装	6
1.2.5 测试 GNU Radio 安装是否成功	10
1.3 源码安装 gr-osmosdr	10
1.3.1 下载 gr-osmosdr 源码	11
1.3.2 编译安装	11
1.4 apt-get 安装 Gqrx	12
1.4.1 下载安装	12
1.4.2 测试 Gqrx 安装是否成功	12
2. HackRF 固件烧写(新手可暂时不看)	14
2.1 下载固件	14
2.2 编译安装 dfu-util	14
2.2.1 安装依赖	14
2.2.2 下载 dfu-util 源码	15
2.2.3 编译安装	15
2.3 烧写固件	15
2.3.1 进入 DFU 模式	15
2.3.2 烧写 Boot(hackrf_one_usb_ram.dfu)	16
2.3.3 烧写 Flash 固件	16
2.3.4 烧写 CPLD 固件	16
2.4 HackRF 固件升级到 2017.02.1 版本	17
2.4.1 HackRF host 软件更新	17
2.4.2 HackRF 固件更新	17
3. 参考文献:	17

【注意】：请在 Ubuntu64 位机器上来使用本教程，因为有用户反映 32 位机器上可能存在一些问题！具体问题，作者目前没有亲自验证，在此谢谢回馈的用户！

1. HackRF 软件环境搭建

1.1 源码安装 HackRF 的 host

1.1.1 安装依赖

目的是为安装 host 做准备，主要是安装一些编译或 USB 驱动相关工具。

```
$ sudo apt-get install build-essential cmake libusb-1.0-0-dev  
pkg-config libfftw3-dev
```

1.1.2 下载 host 源码

```
$ git clone --progress http://github.com/mossmann/hackrf.git
```

注：如果您对 github 的使用还不太熟悉，或者由于下载慢，或者在运行上述命令时碰到不能解决的问题，别担心，随本教程赠送的资料中《before build》文件夹下的《hackrf.tar.gz》压缩包即为作者执行上述语句后下载得到的文件经压缩后的压缩包，您只需解压该压缩包到您指定的路径下即可，如解压到<your user name>/Downloads 文件夹下，然后在进行后续操作即可。

1.1.3 编译安装

Step1: 进入 clone 得到的 hackrf 源代码文件夹的 host 文件夹中，

```
$ cd host
```

Step2: 当前路径下，创建一个 build 文件夹，

```
$ mkdir build
```

Step3: 进入到上述新建的 build 文件夹中，

```
$ cd build
```

Step4：使用 `cmake` 创建 `makefile` 文件，注：以下语句中的【`-DINSTALL_UDEV_RULES=ON`】意思是开启 `hackrf` 识别权限

```
$ cmake ../ -DINSTALL_UDEV_RULES=ON
```

Step5：编译 `make`，

```
$ sudo make
```

Step5：安装，

```
$ sudo make install
```

Step6：运行 `ldconfig` 命令，使得新安装的库可以用。

```
$ sudo ldconfig
```

至此，HackRF 的电脑中的 `host` 安装完毕。

1.1.4 测试 `host` 安装是否成功

将 HackRF 用 USB 线接入电脑，在 `shell` 终端中先输入

```
$ hackrf_info
```

会显示文字如下：

```
$ hackrf_info

Found HackRF board.

Board ID Number: XXXXXX

Firmware Version: XXXXXX

Part ID Number: XXXXXX

Serial Number: 0xXXXXXX
```

这也就证明 **HackRF** 已经能成功被电脑所识别。

1.2 源码安装 Gnuradio

1.2.1 安装依赖

按照 Gnuradio 官网上对于 Ubuntu 源码安装的教程【<http://gnuradio.org/redmine/projects/gnuradio/wiki/UbuntuInstall>】，得知在 Ubuntu 14.04 64 位机器上安装依赖的命令应该为：

```
$ sudo apt-get -y install git-core cmake g++ python-dev swig
pkg-config libfftw3-dev libboost1.55-all-dev libcppunit-dev
libgsl0-dev libusb-dev libsdl1.2-dev python-wxgtk2.8 python-
numpy python-cheetah python-lxml doxygen libxi-dev python-sip
libqt4-opengl-dev libqwt-dev libfontconfig1-dev libxrender-dev
python-sip python-sip-dev
```

但是作者执行上述命令后，出现了如下错误，通过查看发现错误提示为：

libboost1.55-all-dev: Depends: libboost1.55-dev but it is not going to be installed, 这说明要想安装 libboost1.55-all-dev，那么必须先安装 libboost1.55-dev。

不要担心，请看下边的解决办法。

```
lsc@lsc-Desktop: ~
lsc@lsc-Desktop:~$ sudo apt-get -y install git-core cmake g++ python-dev swig p
kg-config libfftw3-dev libboost1.55-all-dev libcppunit-dev libgsl0-dev libusb-de
v libsdl1.2-dev python-wxgtk2.8 python-numpy python-cheetah python-lxml doxygen
libxi-dev python-sip libqt4-opengl-dev libqwt-dev libfontconfig1-dev libxrender-
dev python-sip python-sip-dev
[sudo] password for lsc:
Reading package lists... Done
Building dependency tree
Reading state information... Done
cmake is already the newest version.
doxygen is already the newest version.
g++ is already the newest version.
g++ set to manually installed.
libusb-dev is already the newest version.
pkg-config is already the newest version.
python-cheetah is already the newest version.
python-sip is already the newest version.
libqt4-opengl-dev is already the newest version.
libqt4-opengl-dev set to manually installed.
python-lxml is already the newest version.
Some packages could not be installed. This may mean that you have
requested an impossible situation or if you are using the unstable
distribution that some required packages have not yet been created
or been moved out of Incoming.
The following information may help to resolve the situation:

The following packages have unmet dependencies:
 libboost1.55-all-dev : Depends: libboost1.55-dev but it is not going to be inst
alled
                          Depends: libboost1.55-tools-dev
                          Depends: libboost-atomic1.55-dev but it is not going to
be installed
```

上述问题的解决办法是执行以下命令：

```
$ sudo apt-get install libboost1.55-dev
```

上述命令执行结束后，shell 终端显示如下：

```
lsc@lsc-Desktop: ~  
Removing libboost-coroutine-dev (1.54.0.1ubuntu1) ...  
Removing libboost-coroutine1.54-dev (1.54.0-4ubuntu3.1) ...  
Removing libboost-context1.54-dev:amd64 (1.54.0-4ubuntu3.1) ...  
Removing libboost-date-time-dev:amd64 (1.54.0.1ubuntu1) ...  
Removing libboost-date-time1.54-dev:amd64 (1.54.0-4ubuntu3.1) ...  
Removing libboost-dev (1.54.0.1ubuntu1) ...  
Removing libboost-exception-dev:amd64 (1.54.0.1ubuntu1) ...  
Removing libboost-exception1.54-dev:amd64 (1.54.0-4ubuntu3.1) ...  
Removing libboost-filesystem-dev:amd64 (1.54.0.1ubuntu1) ...  
Removing libboost-wave-dev:amd64 (1.54.0.1ubuntu1) ...  
Removing libboost-wave1.54-dev:amd64 (1.54.0-4ubuntu3.1) ...  
Removing libboost-filesystem1.54-dev:amd64 (1.54.0-4ubuntu3.1) ...  
Removing libboost-graph-dev:amd64 (1.54.0.1ubuntu1) ...  
Removing libboost-graph-parallel-dev (1.54.0.1ubuntu1) ...  
Removing libboost-graph-parallel1.54-dev (1.54.0-4ubuntu3.1) ...  
Removing libboost-graph1.54-dev:amd64 (1.54.0-4ubuntu3.1) ...  
Removing libboost-iostreams-dev:amd64 (1.54.0.1ubuntu1) ...  
Removing libboost-iostreams1.54-dev:amd64 (1.54.0-4ubuntu3.1) ...  
Removing libboost-locale-dev:amd64 (1.54.0.1ubuntu1) ...  
Removing libboost-locale1.54-dev:amd64 (1.54.0-4ubuntu3.1) ...  
Removing libboost-math-dev:amd64 (1.54.0.1ubuntu1) ...  
Removing libboost-math1.54-dev:amd64 (1.54.0-4ubuntu3.1) ...  
Removing libboost-mpi-dev (1.54.0.1ubuntu1) ...  
Removing libboost-mpi-python-dev (1.54.0.1ubuntu1) ...  
Removing libboost-mpi-python1.54-dev (1.54.0-4ubuntu3.1) ...  
Removing libboost-mpi1.54-dev (1.54.0-4ubuntu3.1) ...  
Removing libboost-program-options-dev:amd64 (1.54.0.1ubuntu1) ...  
Removing libboost-program-options1.54-dev:amd64 (1.54.0-4ubuntu3.1) ...  
Removing libboost-python-dev (1.54.0.1ubuntu1) ...  
Removing libboost-python1.54-dev:amd64 (1.54.0-4ubuntu3.1) ...  
Removing libboost-random-dev:amd64 (1.54.0.1ubuntu1) ...  
Removing libboost-random1.54-dev:amd64 (1.54.0-4ubuntu3.1) ...  
Removing libboost-regex-dev:amd64 (1.54.0.1ubuntu1) ...  
Removing libboost-regex1.54-dev:amd64 (1.54.0-4ubuntu3.1) ...  
Removing libboost-serialization-dev:amd64 (1.54.0.1ubuntu1) ...  
Removing libboost-serialization1.54-dev:amd64 (1.54.0-4ubuntu3.1) ...  
Removing libboost-signals-dev:amd64 (1.54.0.1ubuntu1) ...  
Removing libboost-signals1.54-dev:amd64 (1.54.0-4ubuntu3.1) ...  
Removing libboost-system-dev:amd64 (1.54.0.1ubuntu1) ...  
Removing libboost-system1.54-dev:amd64 (1.54.0-4ubuntu3.1) ...  
Removing libboost-test-dev:amd64 (1.54.0.1ubuntu1) ...  
Removing libboost-test1.54-dev:amd64 (1.54.0-4ubuntu3.1) ...  
Removing libboost-timer-dev:amd64 (1.54.0.1ubuntu1) ...  
Removing libboost-timer1.54-dev:amd64 (1.54.0-4ubuntu3.1) ...  
Removing libboost1.54-dev (1.54.0-4ubuntu3.1) ...  
Processing triggers for man-db (2.6.7.1-1) ...  
Selecting previously unselected package libboost1.55-dev.  
(Reading database ... 174870 files and directories currently installed.)  
Preparing to unpack .../libboost1.55-dev_1.55.0-1_amd64.deb ...  
Unpacking libboost1.55-dev (1.55.0-1) ...  
Setting up libboost1.55-dev (1.55.0-1) ...
```

【请注意】，解决完 libboost1.55-all-dev: Depends: libboost1.55-dev but it is not going to be installed 问题后，您仍需继续重新执行一次以下安装依赖的命令才能完成所有依赖的安装，

即：

```
$ sudo apt-get -y install git-core cmake g++ python-dev swig  
pkg-config libfftw3-dev libboost1.55-all-dev libcppunit-dev  
libgsl0-dev libusb-dev libsdl1.2-dev python-wxgtk2.8 python-  
numpy python-cheetah python-lxml doxygen libxi-dev python-sip  
libqt4-opengl-dev libqwt-dev libfontconfig1-dev libxrender-dev  
python-sip python-sip-dev
```

1.2.2 修改 Python PATH

安装完 GNU Radio 的依赖之后 GNU Radio 后需要配置环境参数 PYTHONPATH, shell 终端中输入以下命令即可：

```
$ export PYTHONPATH=/usr/local/lib/python2.7/dist-packages
```

1.2.3 下载 GNU Radio 源码

打开一个 shell 终端，切换到你想存放 GNU Radio 源代码的文件夹下并执行以下命令，例如我的文件夹是/home/lsc/Downloads，其中“lsc”是我的电脑用户名。

```
$ git clone --recursive http://git.gnuradio.org/git/gnuradio.git
```

注：如果您对 github 的使用还不太熟悉，或者由于下载慢，或者在运行上述命令时碰到不能解决的问题，别担心，随本教程赠送的资料中《before build》文件夹下的《gnuradio.tar.gz》压缩包即为执行上述语句后下载得到的文件经压缩后的压缩包，您只需解压该压缩包到您指定的路径下即可，如解压到<your user name>/Downloads 文件夹下。

1.2.4 编译安装

Step1: 进入到你 clone 得到的 GNU Radio 文件夹，

```
$ cd gnuradio
```

Step2: 创建一个 build 文件夹

```
$ mkdir build
```

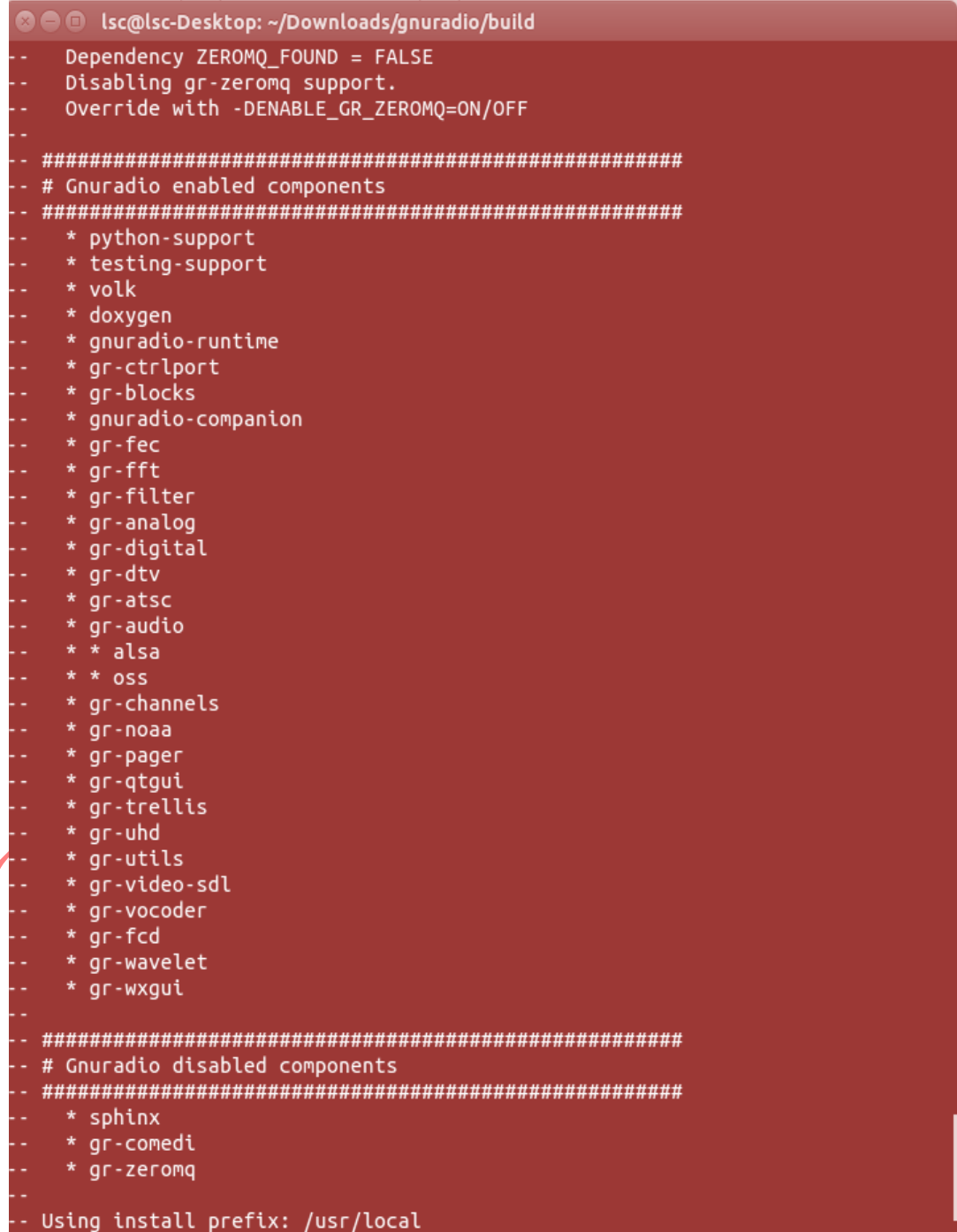
Step3: 进入 build 文件夹

```
$ cd build
```

Step4: cmake 创建 makefile 文件

```
$ cmake ../
```

cmake 后的 shell 终端截图如下所示,




```
lsc@lsc-Desktop: ~/Downloads/gnuradio/build
-- Dependency ZEROMQ_FOUND = FALSE
-- Disabling gr-zeromq support.
-- Override with -DENABLE_GR_ZEROMQ=ON/OFF
--
-- #####
-- # Gnuradio enabled components
-- #####
-- * python-support
-- * testing-support
-- * volk
-- * doxygen
-- * gnuradio-runtime
-- * gr-ctrlport
-- * gr-blocks
-- * gnuradio-companion
-- * gr-fec
-- * gr-fft
-- * gr-filter
-- * gr-analog
-- * gr-digital
-- * gr-dtv
-- * gr-atsc
-- * gr-audio
-- * * alsa
-- * * oss
-- * gr-channels
-- * gr-noaa
-- * gr-pager
-- * gr-qtdgui
-- * gr-trellis
-- * gr-uhd
-- * gr-utils
-- * gr-video-sdl
-- * gr-vocoder
-- * gr-fcd
-- * gr-wavelet
-- * gr-wxgui
--
-- #####
-- # Gnuradio disabled components
-- #####
-- * sphinx
-- * gr-comedi
-- * gr-zeromq
--
-- Using install prefix: /usr/local
```



```
--
-- Using install prefix: /usr/local
-- Building for version: 3.7.10git-0-gcf674211 / 3.7.10git
-- Configuring done
-- Generating done
-- Build files have been written to: /home/lsc/Downloads/gnuradio/build
lsc@lsc-Desktop:~/Downloads/gnuradio/build$
```

Step5: make 编译

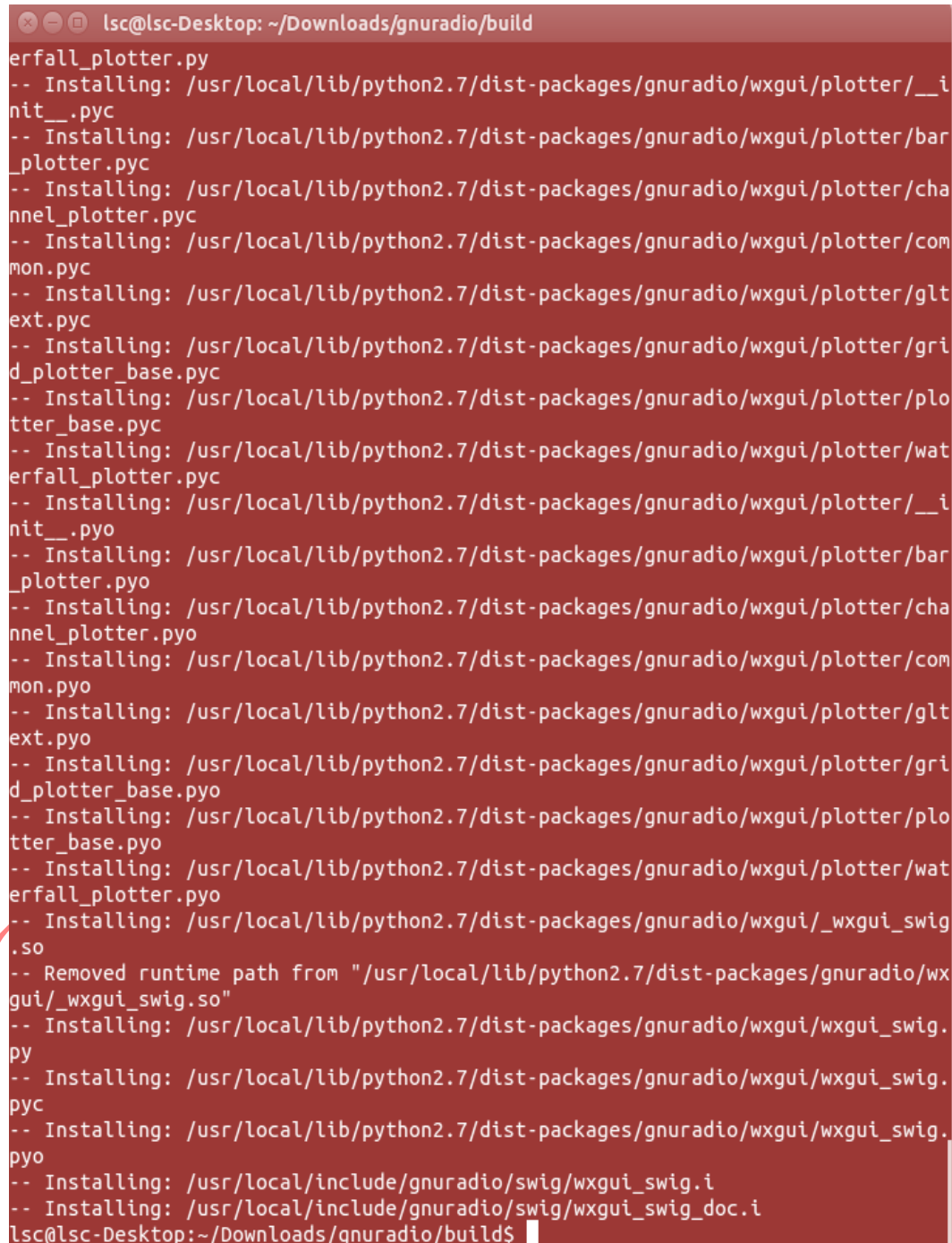
```
$ make
```



```
lsc@lsc-Desktop: ~/Downloads/gnuradio/build
pyc, scopesink_gl.pyc, scope_window.pyc, termsink.pyc, waterfallsink2.pyc, waterfallsink_nongl.pyc, waterfallsink_gl.pyc, waterfall_window.pyc, slider.pyc, stdgui2.pyc
[ 98%] Generating __init__.pyo, common.pyo, constants.pyo, constsink_gl.pyo, const_window.pyo, form.pyo, fftsink2.pyo, fftsink_nongl.pyo, fftsink_gl.pyo, fft_window.pyo, gui.pyo, histosink_gl.pyo, histo_window.pyo, numbersink2.pyo, number_window.pyo, plot.pyo, powermate.pyo, pubsub.pyo, scopesink2.pyo, scopesink_nongl.pyo, scopesink_gl.pyo, scope_window.pyo, termsink.pyo, waterfallsink2.pyo, waterfallsink_nongl.pyo, waterfallsink_gl.pyo, waterfall_window.pyo, slider.pyo, stdgui2.pyo
[ 98%] Built target pygen_gr_wxgui_python_wxgui_91d94
Scanning dependencies of target pygen_gr_wxgui_python_wxgui_93e78
[ 98%] Generating forms/__init__.pyc, forms/forms.pyc, forms/converters.pyc
[ 98%] Generating forms/__init__.pyo, forms/forms.pyo, forms/converters.pyo
[ 98%] Built target pygen_gr_wxgui_python_wxgui_93e78
Scanning dependencies of target _wxgui_swig_doc_tag
[ 98%] Building CXX object gr-wxgui/swig/CMakeFiles/_wxgui_swig_doc_tag.dir/_wxgui_swig_doc_tag.cpp.o
Linking CXX executable _wxgui_swig_doc_tag
[ 98%] Built target _wxgui_swig_doc_tag
Scanning dependencies of target wxgui_swig_swig_doc
[ 99%] Generating doxygen xml for wxgui_swig_doc docs
[ 99%] Generating python docstrings for wxgui_swig_doc
[ 99%] Built target wxgui_swig_swig_doc
Scanning dependencies of target _wxgui_swig_swig_tag
[ 99%] Building CXX object gr-wxgui/swig/CMakeFiles/_wxgui_swig_swig_tag.dir/_wxgui_swig_swig_tag.cpp.o
Linking CXX executable _wxgui_swig_swig_tag
[ 99%] Built target _wxgui_swig_swig_tag
[ 99%] Generating wxgui_swig.tag
Scanning dependencies of target wxgui_swig_gr_wxgui_swig_bfff2
[ 99%] Building CXX object gr-wxgui/swig/CMakeFiles/wxgui_swig_gr_wxgui_swig_bfff2.dir/wxgui_swig_gr_wxgui_swig_bfff2.cpp.o
Linking CXX executable wxgui_swig_gr_wxgui_swig_bfff2
Swig source
[ 99%] Built target wxgui_swig_gr_wxgui_swig_bfff2
Scanning dependencies of target _wxgui_swig
[ 99%] Building CXX object gr-wxgui/swig/CMakeFiles/_wxgui_swig.dir/wxgui_swigPYTHON_wrap.cxx.o
Linking CXX shared module _wxgui_swig.so
[ 99%] Built target _wxgui_swig
Scanning dependencies of target pygen_gr_wxgui_swig_a87ba
[100%] Generating wxgui_swig.pyc
[100%] Generating wxgui_swig.pyo
[100%] Built target pygen_gr_wxgui_swig_a87ba
lsc@lsc-Desktop:~/Downloads/gnuradio/build$
```

Step6: make install 安装

```
$ sudo make install
```



```
lsc@lsc-Desktop: ~/Downloads/gnuradio/build
erfall_plotter.py
-- Installing: /usr/local/lib/python2.7/dist-packages/gnuradio/wxgui/plotter/__i
nit__.pyc
-- Installing: /usr/local/lib/python2.7/dist-packages/gnuradio/wxgui/plotter/bar
_plotter.pyc
-- Installing: /usr/local/lib/python2.7/dist-packages/gnuradio/wxgui/plotter/cha
nnel_plotter.pyc
-- Installing: /usr/local/lib/python2.7/dist-packages/gnuradio/wxgui/plotter/com
mon.pyc
-- Installing: /usr/local/lib/python2.7/dist-packages/gnuradio/wxgui/plotter/glt
ext.pyc
-- Installing: /usr/local/lib/python2.7/dist-packages/gnuradio/wxgui/plotter/gri
d_plotter_base.pyc
-- Installing: /usr/local/lib/python2.7/dist-packages/gnuradio/wxgui/plotter/plo
tter_base.pyc
-- Installing: /usr/local/lib/python2.7/dist-packages/gnuradio/wxgui/plotter/wat
erfall_plotter.pyc
-- Installing: /usr/local/lib/python2.7/dist-packages/gnuradio/wxgui/plotter/__i
nit__.pyo
-- Installing: /usr/local/lib/python2.7/dist-packages/gnuradio/wxgui/plotter/bar
_plotter.pyo
-- Installing: /usr/local/lib/python2.7/dist-packages/gnuradio/wxgui/plotter/cha
nnel_plotter.pyo
-- Installing: /usr/local/lib/python2.7/dist-packages/gnuradio/wxgui/plotter/com
mon.pyo
-- Installing: /usr/local/lib/python2.7/dist-packages/gnuradio/wxgui/plotter/glt
ext.pyo
-- Installing: /usr/local/lib/python2.7/dist-packages/gnuradio/wxgui/plotter/gri
d_plotter_base.pyo
-- Installing: /usr/local/lib/python2.7/dist-packages/gnuradio/wxgui/plotter/plo
tter_base.pyo
-- Installing: /usr/local/lib/python2.7/dist-packages/gnuradio/wxgui/plotter/wat
erfall_plotter.pyo
-- Installing: /usr/local/lib/python2.7/dist-packages/gnuradio/wxgui/_wxgui_swig
.so
-- Removed runtime path from "/usr/local/lib/python2.7/dist-packages/gnuradio/wx
gui/_wxgui_swig.so"
-- Installing: /usr/local/lib/python2.7/dist-packages/gnuradio/wxgui/wxgui_swig.
py
-- Installing: /usr/local/lib/python2.7/dist-packages/gnuradio/wxgui/wxgui_swig.
pyc
-- Installing: /usr/local/lib/python2.7/dist-packages/gnuradio/wxgui/wxgui_swig.
pyo
-- Installing: /usr/local/include/gnuradio/swig/wxgui_swig.i
-- Installing: /usr/local/include/gnuradio/swig/wxgui_swig_doc.i
lsc@lsc-Desktop:~/Downloads/gnuradio/build$
```

Step7: 运行 ldconfig 命令，使得新安装的库可以用。

```
$ sudo ldconfig
```

1.2.5 测试 GNU Radio 安装是否成功

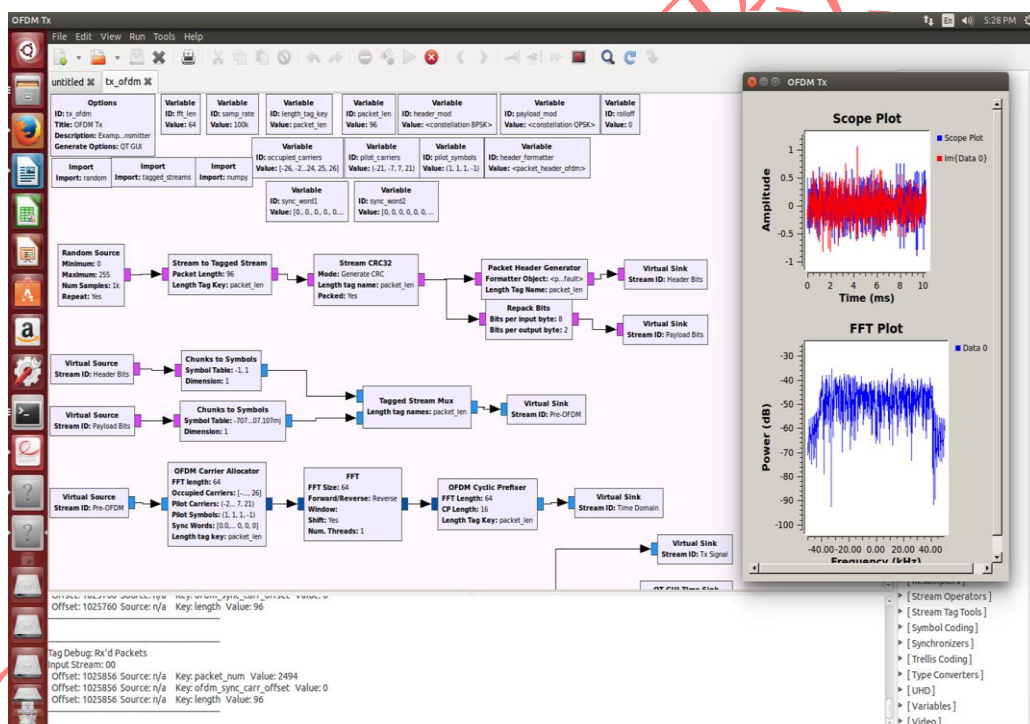
安装结束之后，测试 gnuradio 是否成功。

命令行输入：gnuradio-companion

打开 GRC 界面，

点击 file—open，打开/usr/local/share/gnuradio/examples/digital/ofdm 文件夹下【只要你按照本教程来做，那么这些文件路径应该就与作者一致】的 tx_ofdm.grc 文件，界面如下，依

次点击生成按钮和运行按钮结果如下：



至此，说明 GNU Radio 的安装已成功。

1.3 源码安装 gr-osmosdr

GrOsmoSDR 是一个必须的中间件，它用于支持 GNU Radio 与 HackRF 软件之间进行通信，以此来控制 HackRF 板子。

1.3.1 下载 gr-osmosdr 源码

```
$ git clone git://git.osmocom.org/gr-osmosdr
```

1.3.2 编译安装

Step1: 进入到你 clone 得到的 gr-osmosdr 文件夹,

```
$ cd gr-osmosdr
```

Step2: 创建一个 build 文件夹

```
$ mkdir build
```

Step3: 进入 build 文件夹

```
$ cd build
```

Step4: cmake 创建 makefile 文件

```
$ cmake ../
```

Step5: make 编译

```
$ make
```

Step6: make install 安装

```
$ sudo make install
```

Step7: 运行 ldconfig 命令, 使得新安装的库可以用。

```
$ sudo ldconfig
```

1.4 apt-get 安装 Gqrx

1.4.1 下载安装

依据 Gqrx 官网【<http://gqrx.dk/download/install-ubuntu>】，依次进行如下命令即可。

```
$ sudo apt-get purge --auto-remove gqrx
$ sudo apt-get purge --auto-remove gqrx-sdr
$ sudo add-apt-repository -y ppa:bladerf/bladerf
$ sudo add-apt-repository -y ppa:ettusresearch/uhd
$ sudo add-apt-repository -y ppa:myriadrf/drivers
$ sudo add-apt-repository -y ppa:myriadrf/gnuradio
$ sudo add-apt-repository -y ppa:gqrx/gqrx-sdr
$ sudo apt-get update
$ sudo apt-get install gqrx-sdr
```

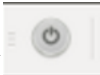
1.4.2 测试 Gqrx 安装是否成功

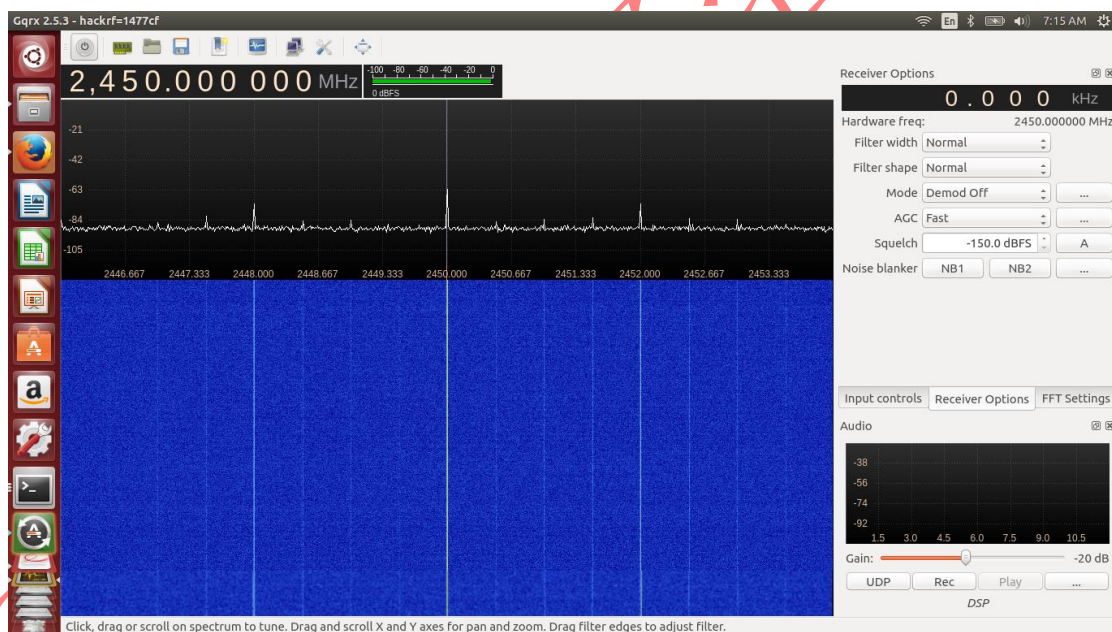
当 HackRF 用 USB 线接入电脑后，shell 终端输入，

```
$ gqrx
```

`gqrx` 会运行起来，当你第一次运行 `gqrx` 的时候，会弹出一个设备配置的界面，如下图所示：



点击开始按钮 ，如果你能看到下边的图形，那么祝贺你已经完成 gqrx 的安装！



如果你想源码安装 Gqrx，则可按照以下命令来完成。

源码安装 Gqrx

qt

下载 qt 安装器

下载网址：http://mirror.bit.edu.cn/qtproject/archive/online_installers/2.0/qt-unified-linux-x64-2.0.2-2-online.run

修改文件权限：

```
$ sudo chmod +x qt-unified-linux-x64-2.0.2-2-online.run
```

```
$ sudo ./qt-unified-linux-x64-2.0.2-2-online.run
```



```
git clone https://github.com/csete/gqrx.git
cd gqrx
mkdir build
cd build
qmake ../gqrx.pro
make
sudo make install
sudo ldconfig
```

2. HackRF 固件烧写(新手可暂时不看)

关于 HackRF 固件烧写内容,新手可根据自己实际情况来决定是否要看,如果对于 linux 不是很熟悉的话,可以暂时先不用看。因为我们每一块板子发货之前都会进行烧写固件及测试,保证测试通过后才会发货。

2.1 下载固件

从 <https://github.com/mossmann/hackrf/releases> 上下载固件压缩包,或者查看随本教程赠送的资料中的《hackrf-2015.07.2.zip》。解压压缩包后得到以下文件:

名称	修改日期	类型
doc	2016/4/8 10:53	文件夹
firmware	2016/4/8 10:53	文件夹
firmware-bin	2016/4/8 10:53	文件夹
hardware	2016/4/8 10:53	文件夹
host	2016/4/8 10:53	文件夹
COPYING	2015/7/24 10:21	文件
Readme.md	2015/7/24 10:21	MD 文件
RELEASENOTES	2015/7/24 10:23	文件
TRADEMARK	2015/7/24 10:21	文件

2.2 编译安装 dfu-util

当 HackRF 的固件损坏或者你拿到一个新板子时,需要在 DFU 引导模式(Development FirmwareUpgrade Boot Mode)下来往板子中烧写固件。

2.2.1 安装依赖

```
$ sudo apt-get build-dep dfu-util
```

```
$ sudo apt-get install libusb-1.0-0-dev
```

2.2.2 下载 dfu-util 源码

```
$ git clone git://git.code.sf.net/p/dfu-util/dfu-util
```

2.2.3 编译安装

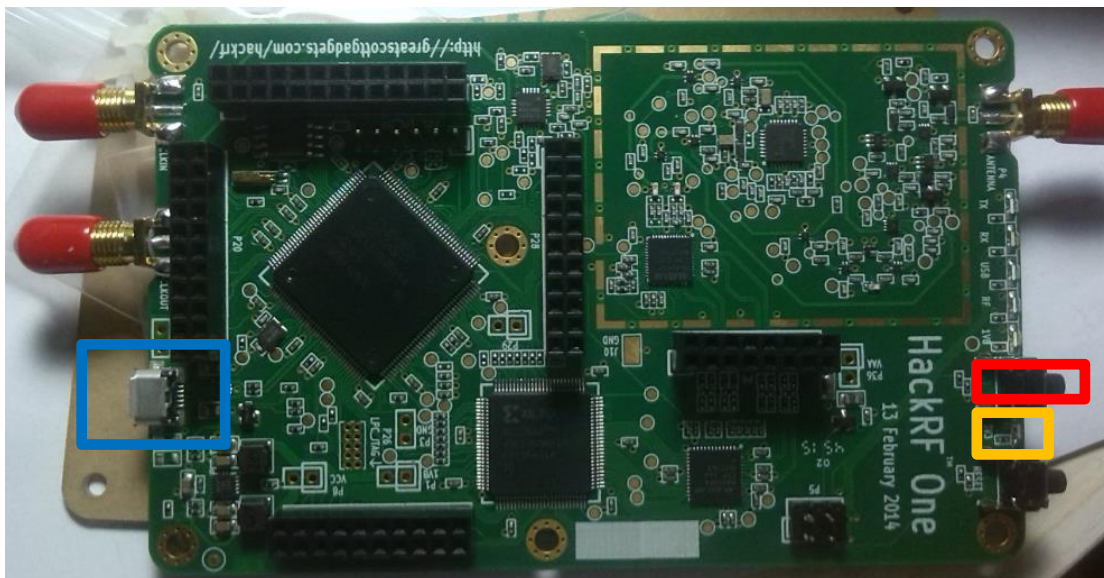
```
$ cd dfu-util
$ git checkout v0.7
$ ./autogen.sh
$ ./configure
$ make
$ sudo make install
```

注意：其中的 `git checkout v0.7` 命令是指定 `dfu-util` 的版本，本教程使用的是 0.7 版本。

2.3 烧写固件

2.3.1 进入 DFU 模式

如下图标红框区域所示，该按键就是 DFU 按键，按住 DFU 按键不送手，然后将 USB 线接入标蓝框区域接口，此时下图标黄框区域的灯(标有 3V3 的 LED 灯)会亮起，则说明板子已进入 DFU 模式。



2.3.2 烧写 Boot(hackrf_one_usb_ram.dfu)

首先进入 2.1 节中你解压后的文件夹中的固件所在的文件夹中。

```
$ cd 2.1 节中你解压后的文件夹/firmware-bin/
```

```
$ dfu-util --device 1fc9:000c --alt 0 --download hackrf_one_usb_ram.dfu
```

2.3.3 烧写 Flash 固件

```
$ hackrf_spiflash -w hackrf_one_usb_rom_to_ram.bin
```

2.3.4 烧写 CPLD 固件

```
$ hackrf_cpldjtag -x hackrf_cpld_default.xsvf
```

2.4 HackRF 固件升级到 2017.02.1 版本

在使用 HackRF 的过程中碰到如下所示的问题,可通过升级 HackRF 的固件版本来解决。

```
call hackrf_set_sample_rate(2600000 Hz/2.600 MHz)
call hackrf_set_hw_sync_mode(0)
hackrf_set_hw_sync_mode() failed: feature not supported by installed firmware (-1005)
```

下载 2017.02.1 版本固件压缩包,解压缩后依次执行以下操作:

2.4.1 HackRF host 软件更新

```
cd host
mkdir build
cmake ../ -DINSTALL_UDEV_RULES=ON
make
sudo make install
sudo ldconfig
```

2.4.2 HackRF 固件更新

进入 firmware-bin 目录 (cd firmware-bin), 执行以下操作

1) 更新 Flash

```
hackrf_spiflash -Rw hackrf_one_usb.bin
```

2) 更新 CPLD

```
hackrf_cpldjtag -x hackrf_cpld_default.xsvf
```

3. 参考文献:

<http://gnuradio.org/redmine/projects/gnuradio/wiki/UbuntuInstall>

Building GNU Radio on Ubuntu Linux

<https://mborgerson.com/getting-started-with-the-hackrf-one-on-ubuntu-14-04>

Getting Started with the HackRF One on Ubuntu 14.04

<http://gqrx.dk/download/install-ubuntu>

Install Gqrx SDR on Ubuntu Linux

<https://github.com/mossmann/hackrf/wiki/Updating-Firmware>

Updating Firmware