Notes on SDR

J.D. McCormack

2015-07-01

1 Introduction

The following document will contain information related to my efforts in software defined radio. From time to time, the document may include information on other topics as well. It will initially serve as a day to day journal of my activites. After a certain discovery phase, this document will be updated to create a manual or tutorial for future students to learn from. It is not recommended that the document be used for self study until a newer version is created. Many of the steps that are outlined will be dead ends or poor practices while the disocvery process takes place.

2 7-1-2015

Currently working on using the RTL SDR.

What I have learned so far:

- This SDR is different than a FUNCUBE dongle
- It will work from around 500 Hz to 1700 Hz
- It can only RECIEVE

The fact that this can only recieve is by far the largest setback. This will ultimately not be what we can use for cognitive radio.

However, the tutorials still seem more widely available than those found for the BladeRF which is bi-directional. Therefore I will continue for a bit. It may serve a purpose as part of the larger sensor network at some point. We could still use it to relay information over a large distance.

There are "Cubelites" being sent out that are basically funcube dongle based satellites. These sound pretty cool.

I want to start taking notes in Latex so they will be more readable than a plain .txt file and still useable with version control.

I need to install texlive-full and texmaker.

Currently I am set back by needing to update ubuntu. I can't install anything else while the updates are installing.

Success So far I have made decent progress. The RTL is working with GNU SDR. I had trouble at first when the kernel was loading a separate driver that bogs it down.

I used this command to fix that problem:

```
# sudo rmmod dvb_usb_rtl28xxu rtl2823
```

There are more permanent ways of fixing the problem but I wanted to start with this as it is non-permanent and will default back to normal on a restart.

The new file in the RTL-SDR will allow you to hear FM stations. But they are extremely faint. I'm going to continue to play with the file settings to see if I can get a clearer transmission.

3 7-2-2015

I began the day with a quick interlude into learning LaTex. After about an hour and a half of studying I was able to produce the document you are currently reading. I'm not making use of many libraries but I believe it already seems more organized and official than my traditional notes. Also, it is much more portable than a normal word document and less prone to formating failures seen in google docs.

Yesterday I got the SDR to recieve FM stations. However, they were staticy and sounded slowed down. I'm not sure if this is an error in my demodulation values or just that the computer that I'm using is too slow. I tried using a different one but that ended up being slower than I remembered. I will try to get linux installed on my laptop later tonight when I'm done working, but I'm not confident it will work as I had trouble in the past.

Today I will try to replicate the results I had yesterday, but through the use of the python libraries instead of relying on the GNURadio GUI interface. A small note about the gui interface. I did learn that its possible to make variables, attach them to GUI widgets, and then alter then live. It appears that having more than 3 of these can severely impact performace. To use the GUI widgets, simple drag one onto your workspace (I used the Wx widgets) and then give it a unique name, and appropriate value range (if its a slider). Then in the blocks for different components, you can use these variable names instead of hardcoded values. This is useful for gains and frequencies.

/paragraphGNURadio in Python

I will be following the tutorial found here. I will begin with the dial tone generator.

Listing 1: dial_tone.py

```
#!/usr/bin/env python

# Gr is the basic gnu radio module
# gr is always required
from gnuradio import gr
# audio and analog are libraries of blocks
from gnuradio import audio, analog

# Here we define a top block that inherits from
# the gr class top block.
class my_top_block(gr.top_block):
  def __init__(self):
    gr.top_block.__init__(self)

    # standard quality sample rate and amplitude value
    sample_rate = 32000
    ampl = 0.1

    # We create two sine waves with the same amplitude and sameple rate
    # from above.

    src0 = analog.sig_source_f(sample_rate, analog.GR_SIN_WAVE, 350, ampl)
    src1 = analog.sig_source_f(sample_rate, analog.GR_SIN_WAVE, 440, ampl)

    # dst is the "destination" or audio sink block
    dst = audio.sink(sample_rate, "")

    # We then connect the two sine waves to the destination block
    self.connect(src0, (dst, 0))
    self.connect(src1, (dst, 1))

if __name__ == '__main__':
  try:
    my_top_block().run()
  except [[KeyboardInterrupt]]:
    pass
```

Static I'm still only getting static when I run the RTL-SDR. I was able to use the GNURadio toolchain from just python no problem, however I still need to implent the RTL-SDR from the python toolchain. I used an additional tool put out by osmocom but that did not product better results. Automatic gain was on, so its possible I need to manually set the gain,

but I am not sure. It could also be the poor quality of the antenna. the command was:

```
# rtl_fm -f 96.3e6 -M wbfm -s 200000 -r 48000 - | aplay -r 48k -f S16_LE
```

aplay is a command line utility for playing audio. I had to run that separately as piping the data did not seem to work. I got this command from osmocom's website

4 07-03-2015

I will not have much time to work today due to the holiday weekend. I will try to get Linux installed on my laptop in my free time. Either tomorrow or next week I will try to find a python example using the RTL-SDR driver. After that, I may switch gears back to the BladeRF.

5 07-04-2015

Happy 4th of July! No progress will be made today as I'll be busy all day.

6 07-05-2015

Didn't get a chance to work today either. Family event. Will continue progress tomorrow.

7 07-06-2015

Back to work today. I was attempting to install a program called Gqrx. Using the instructions from their github I added a repository and installed just the gqrx program. However, this created a massive conflict in my repositories. I uninstalled and figured I would just go back to working on the FM model to see if I could be sure its working right. However, installing that package messed up my installation of gnu radio. So now I must uninstall and reinstall evertyhing. At the time, I was following this tutorial On Gqrx. I found a person with a similar problem to mine here. There are instructions below on how to reinstall everything. I will be trying that next and will hopefully get back to where I was. It may also be time to start learning more about virtuan environments or just how to reimage a computer quickly.

Currently trying to use the following command to reset the broken packages. This is taking a long time so I will have to wait for it to finish in order to continue reinstalling gnuradio. The command is:

```
# sudo apt-get dist-upgrade
```

This did not end up working. Next I tried using aptitude, which as I have read, will work to fix these broken dependencies and packages. The command for that is:

```
# sudo aptitude install <packagename>
```

Still trying to get everything installed and working. The repository made GNURadio work again, but I lost the Osmocom packages. Still trying. I have been following the instructions direct from osmocom but I'm getting errors related to "Gruel" when I try to use cmake.

Success Finally remembered that the instructables I was using previously had the commands listed (although for arch). The three necessary components can be installed by using:

```
# sudo apt-get install gnuradio


# sudo apt-get install rtl-sdr


# sudo apt-get install gr-osmosdr
```

As a note, the instructable used arch linux and the final package was installed as gr-osmosdr-git. Debian/Fedora/Arch users may need to use this form of the package instead of gr-osmosdr. Another github was found here with plenty of examples. They may be out of date as the AM example did not compile correctly. Another useful site was found here this site had a single example, but it seemed to be working. I'm going to try to test to see if I can use my CB radio with this. I think it will be too low of a frequency, but this would make testing the blade very easy as I can control the transmission with just the hand held radio.

R820T So after trying to find a software solution for determining which tuner the device was using, I decided that the smarter solution was to just open the SDR up. The chip inside is the Rafael Micro R820T. This is GOOD. Next to the elusive E4000 this is the best possible tuner to have. Its range is 24-1766 MHz. Not quite as high as the E4000 but much lower than that one and much higher than any other tuner that it could have had. I'm pretty happy. The device came apart with no tools or fuss, everything was just snapped together. The device even went back together when it was done!

CB Radio After a bit of struggling it looks like I'm able to pick up CB signals. The main issue I have now is that without better knowledge of signal processing I can't properly adjust the FIR filter. So I am able to recieve CB signals but they are not specific to any channel. The antenna that comes with this RTL-SDR is also awful. If I transmit over wireless I recieve really garbled sounds. If I unscrew both antennas and connect by contact it is crystal clear but still not on any specific channel. I don't think I'll spend much more time on this however. The file is on github under the name airband_4.grc and airband_5.grc. The 5 was an attempt to fix some of the issues with the downloaded file. It was called airband_4 on the website because it was able to pickup 4 different AM broadcasts used by airplanes to communicate.

I found two resources to keep track of for general SDR and DSP information.

- http://complextoreal.com/tutorials/#.VZr6nKH7s_t
- http://greatscottgadgets.com/sdr/

8 07-07-2015

Today I will be taking a look at the bladeRF SDR. The first problem that comes up is it appears I forgot to bring a compatible antenna with me. I have to double check all my stuff to be sure, but due to the typically large size of these antennas I think it is likely that they got left behind. Anyway, despite not having an antenna I'm going to push on anyway and then see what I have lying around. I should be able to order something on Amazon fairly quickly if needed anyway. Nuand is nice enough to have a fairly detailed set of documents on their github account. It also looks like these may have been updated since the last time I went through this process. The github account is found here. I also just saw on this webpage that there is now matlab support for the bladeRF on windows. I'm not sure if this was always a feature or if this is indeed new to the program. Eitherway it is something to be aware of. It has support for Simulink too.

Following the tutorial, the first step is to add the appriproate repository so we can download the files. There is a snapshot section that allows you to get the most up to date releases but these are usually far from stable. The commands used are.

8.1 Install BladeRF

The commands are:

```
# sudo apt-get-repository ppa:bladerf/bladerf

# sudo apt-get update

# sudo apt-get install bladerf
```

Of course I immediately get an error saying "you have held broken packages." I should probably do more research into how to avoid this scenario, as it seems to come up quite a lot in working with GNURadio. This could be related to using the other PPA from before. I'm going to try to use aptitude to install this and see if that helps. Aptitude seems to be able to solve the problem, however it will be removing:

- gr-osmosdr (which I know I need)
- libbladerf0
- libgnuradio-osmosdr0.0

So I know that all three of these libraries are needed to run the blade and all but the libbladerf0 are needed to also run the RTL-SDR. I'll give it a shot and see what happens, I can always reinstall from the repository (I Hope). After running

```
# sudo aptitude install bladerf
```

and accepting their suggestions it seems to have installed properly. I also ran:

```
# sudo apt-get install libbladerf-dev
```

but it seemed to have already been installed. Now we can install the updated FPGA drivers.

8.2 Download new firmware

The commands are:

```
# sudo apt-get install bladerf-firmware-fx3

# sudo apt-get install bladerf-fpga-hostedx40

# sudo apt-get install bladerf-fpga-hostedx115
```

It's worth noting that we have the x40 not the x115 so we only need to install the x40.

8.3 Reinstall GNURadio

The next step seems to indicate that I should build the gnuradio from sources and not use the version found in the repository. Luckily, BladeRF links to a download script that pretty much puts cruise control on. Hopefully you have luck too. Here are the commands.

```
# mkdir -p  /software/gnuradio-build

# cd  /software/gnuradio-build

# wget http://www.sbrac.org/files/build-gnuradio

# chmod +x ./build-gnuradio
```

```
# ./build-gnuradio -m prereqs gitfetch
```

It took about 15-20 minutes to finish but everything downloaded successfully. Afterwards its necessary to actually build the downloaded programs. I followed the steps outlined to build GNURadio and the program halts halfway through with a cryptic error message. Hopefully I can diagnose the problem because it took roughly an hour to get that far.

Success! I ended up needing to use the Pybombs tool. It worked like a charm. This is linked to in the github wiki mentioned above and also found on the GNUiRadio webpage.

9 07-08-15

Today I will be traveling and not have much time to work. I spoke too soon on the success of pybombs. It looked like it worked and it did install the BladeRF software properly, however the GNURadio software did not. I tried several times and at one point got an error about the jdk which wasn't listed as a prereq. I tried the plain build again and got much further (89% vs 56% originally). I'll try running it again and see if it goes any further.

10 07-09-15

Today's focus switched to some open tasks I had with Dr. Integlia. The first is a comparison of various FPGA platforms available. After looking at everything, I am most excited by the offerings from digilent, especially the Zynq based boards. These have an onboard FPGA and a microprocessor onboard. I also prepared a comparison of SDR platforms. One board even had a Zynq SoC on board that was programmable. This could lead to interesting research down the line as students become better at using the FPGAs. The FPGA comparison is available in this repository but for now the SDR is not. I will try to migrate it here soon too.

11 07-10-15

Today I started the Literature review. There is a lot of ground to cover. I got through 14 sources. So far the general gist of what I've read is that the USRP is the go to SDR for research, but that people are excited by the RTL-SDR. GNU Radio seems to be very well recieved for both simulation and real world testing. I'm curious to see if some of the other software tools identified have a following in the academic world too.

I also learned how to use the IEEEtrans format for Latex and how to use Bibtex.

12 07-11-15

I won't have as much time to work today but I will try to get some amount done. I need to port the FPGA comparison to excel and continue the Lit Review. I will be trying to spend most of the afternoon working through Node.js and research graduate schools. I've been keeping up on Node.js fairly well but have been falling behind on Grad school searches.

13 07-12-15

Today I will be converting the Latex notes into an excel spreadsheet for sharing. I may not have much time to work today or tomorrow but I will put full days on Tuesday and Wednesday.

14 07-13-15

I'm going to try to make some more progress on the SDR today. I realized I may be more limited by this computer than I thought. It only has USB 2.0's at best and that could be throttling it too much as this is a USB3.0 boad. Regardless, I'll keep trying to build GNURadio for it to at least get the correct way to set that up.

First run:

```
# git clone https://github.com/Nuand/bladeRF.git
```

From there I followed the directions found on their github account here. In step four I switched to the directions here and was able to build the software without errors. You can check to make sure it worked by running

```
# bladeRF-cli -e info -e version
```

or

```
# bladeRF-cli -p
```

So now again I was back to the point where I needed to build GNURadio. I also need to load the drivers to allow for the expansion board, but one step at a time.

Here I would like to point out that it is possible to just use a live CD to run all of this without having to configure anything. However, the live CD does not come with the ability to install itself. I'm not sure what their thought process was for that, maybe to keep everything fresh and up to date, but it is a huge inconvience. If all else fails, the live CD combined with github would allow you to rebuild the environment pretty quickly each time. There is also direct support for bladeRF in Pentoo, a penetration testing varient of Gentoo. I can try this route if all else fails.

I am currently Running PyBombs again to try to get GNURadio to install. Hopefully it works this time, but it is unlikely that it will.

15 07-14-15

I don't want to speak too soon but it looks like GNURadio may have installed. I need to make a note of the following:

```
#  $prefix =  /software/BladeRF/bladeRF/build/target
```

I had been running the following without the sudo command, adding sudo seemed to work. I'm not sure why I didn't see a clearer error before:

```
#  sudo ./pybombs install gnuradio
```

For reference. Pybombs is located here:

```
#  /software/BladeRF/bladeRF/build/pybombs
```

The steps to open GNURadio were:

```
# : /software/BladeRF/bladeRF/build/pybombs$ ./pybombs env


#  /software/BladeRF/bladeRF/build/target$ source setup_env.sh


#  /software/BladeRF/bladeRF/build/target$ gnuradio-companion
```

After installing a few other modules and running an old test program, I can confirm that it is back up and running with the RTL-SDR. Now I am beginning the test with the BladeRF. One problem with the BladeRF is in order to easily test a signal I will have to use the transverter board. This is mostly because FM/CB Radio channels are below the normal operating range of the board. To activate the transverter board use the command:

```
# bladerf-cli -i


# xb 200 enable 200
```

Note the spaces in the command. The first command is just used to bring up the interactive inteface. This command was listed wrong on the wiki page, so I submitted a request to edit it accordingly.

A book to be aware of can be found here. This book's link was also incorrect so I fixed that on the wiki as well.

The biggest issue now is trying to remove the DC offset. Everytime I open the radio in the GNU Radio Companion there is a large spike in the center, and no audio is heard. Currently, my computer keeping freezing whenever I try to run anything using the BladeRF, I believe this may have to do with the limitations of this computer. I will begin trying to get Ubuntu running on my laptop tonight. First I have to clear off a lot of space on the harddrive. It may also have to do with the fact that I am using USB 2.0 ports on this computer. If thats the case, chaning to my laptop will not fix the problem.

16 07-15-15

Spent today trying to install Linux on my laptop. It still isnt cooperating. The problem is the laptop already has 4 partitions on it and HP did not make any of them extended by default as it should be with that many. I am currently trying to copy everything from a recovery partition, onto the main hard drive. I'll then wipe that partition and reformat it. Hopefully it doesn't corrupt my computer. After correspondance with Dr. Integlia, I am now aware of some other ongoing goals. These include:

- Preparing for next semester's courses
- Creating a short document
- Creating a longer more comprehensive document
- Identifying more conferences

I will continue spending today trying to install ubuntu and begin the transition tomorrow.

17 07-16-15

I ended up breaking my computer today while installing Ubuntu and corrupting the windows portion. I unloaded the documents I needed and then starting over with Windows 10 and Ubuntu. Everything works now and I can access google drive again.

18 07-17-15

Worked on the document for Harris corporation. The document can be found in this repository and on google drive.

19 07-18-15

I will not have much time to work today, and probably not much tomorrow. I should be able to make a bigger effort this upcoming week however.

20 07-19-15

Spent the day going over advanced MongoDB work in Node.js. I also had time to begin creating the example problems for the Circuits I class. I will be traveling tomorrow.

21 07-20-15

I will be traveling today with limited access to a computer until tomorrow.

22 07-21-15 to 07-24-15

I am traveling with limited access to SDR equipment. I have been continuing with the literature review list this week and hope to have 20 sources idenfitied by Monday

23 07-25-15

Today I will be working remotely from a friend's office for a few hours. Going to try either getting the SDR back online or to try to finsih up the literature review.

23.1 PyBombs

I still havent had luck with pybombs on the HP Laptop I am working from. It worked on the older computer no problem once I used sudo. I am currently reinstalling it. I have noticed that if I sudo whilte running PyBombs for the first time then the program is not able to be installed. The only chnanges I made this time were selecting "Force install" for everything I knew I would need ahead of time, and setting the number of threads that make could use to 8. I'm not sure if this will prove to be helpful at all. The 8 threads may have been too many as now I'm having trouble running other programs at the same time (music slows down and skips and even vim is noticeably less responsive).
It looks like everything is now installing correctly. The only changes were made were the ones specifed. After install gnuradio, I'm not installing RTL-SDR, BladeRF, and some other packages that sound useful. Once they are done installing its important to remember that you actually need to activate them within the environment you are using. They will not stay permanately active in linux. To do this we navigate to the "source" destination. For me, I just had to go one folder up, and then it was listed under target. If you cant find where it is, just install something else and pay attention to where it says it is writing the files . Then go to the appropriate folder. Once in there, run:

```
# source setup_env.sh
```

After testing it seems everything is back to working, at least with the RTL-SDR. Now I can begin testing with the BladeRF upon my return home. An important note is that I had a slowed down audio effect. It turned out I had miss matched sampling rates. One was being set manually, while another was being set by a variable. I hadn't noticed that there was a discrpency and now the audio sounds normal.

24 07-29-15

I was able to do a quick test once I got home to try out the BladeRF. It is working with GNU Radio now but still needs to get the DC offset fixed.

25 08-04-15

I'm back to working on the BladeRF after a brief hiatus. Currently, I am seeing if I had to reload the FPGA manually. To do this I use the command:

```
# bladeRF-cli -l <path/to/fpga/file>
```

In my case, the fpga file was stored in downloads. The FPGA file can be found on the Nuand website. I also learned that there are string arugments that allows GNURadio to add the xb200 as well as automatically load the FPGA firmware. More information can be found here.

25.1 09-14-15

Finally back at it. Today I will be focusing on the Ettus Research USRP B200 board. I'm going to try to follow the tutorial found at Ettus Research. I was able to use pybombs to install the first set of software. This was simply called "UHD" under the hardware tag in pybombs. As last time, make sure you start pybombs with sudo. There was another set of drivers called "Ettus" that would not install. I later saw this comand

```
# sudo apt-get install libuhd-dev libuhd003 uhd-host
```

It said the last two were already installed, but then installed libuhd-dev. I'm still getting errors that the device is not found in GNURadio. Typing in "lsusb" shows that Linux doesnt seem to recognize the device. I'm going to reboot and see if that helps.

That did not fix the issue. Next, I double checked that "apt-get" covered everything. As some part of the install may be from pybombs, which creates a virtual environment, I'm hoping that this does not cause a mismatch. Ok, so that had no effect. I'm going to google a bit more and see if I can figure out what I'm missing.

```
# uhd_find_devices
```

Can be used to locate the device. Originally, it was throwing an error message. Similar to the below:

```
# /usr/bin/uhd_find_devices:  symbol lookup error:


# /usr/bin/uhd_find_devices:  undefined symbol:


# _ZN3uhd6device4findERKNS_13device_addr_tENS0_15device_filter_tE
```

Which brought me here. Eventually, I discovered that the main issue was I had used the repositores to download the 3 files shown above (libuhd-dev, libuhd003, uhd-host) and then also done the commands shown below:

```
# sudo bash -c 'echo "deb http://files.ettus.com/binaries/uhd/repo/uhd/ubuntu/`lsb_release -cs` `lsb_release -cs` main" > /etc/apt


# sudo apt-get update


# sudo apt-get install -t `lsb_release -cs` uhd
```

When I went back and used

```
# sudo apt-get remove libuhd-dev libuhd003 uhd-host
```

and then ran

```
# sudo uhd_find_devices
```

I got the below message showing that it was now registerd:

– Loading firmware image: /usr/share/uhd/images/usrp_b200_fw.hex... done ——————————————————————————– UHD Device 0 ——————————————————————————– Device Address: type: b200 name: serial: 3087692 product: B200

However I'm still getting an "Empty device address" error in GNURadio. I did recently read that with out using USB 3.0 (my computer only has 2.0) It may be necessary to use either an external power adapter rated at 6 V, 3A or to use one of the 2 A to 1 B type connectors. I saw a brief document saying it may be an issue with not having root access, but running in a shell with root access via

```
# sudo -i
```

did not change anything.

Eventually I Came across the command

```
# uhd_usrp_probe
```

. This loaded the firmware and FPGA. After running this and returning to GNU Radio, I still am getting the errors as before. It looks like the problem may be in switching into the virtual envinronment created with pybombs. I ran the uhd_usrp_probe command again and it mentioned that there were no images installed. The error message provides a command to run which will download the images. The command for me was

```
#  /software/target/lib/uhd/utils/uhd_images_downloader.py
```

But make sure whatever you run is whats given to you in the error output. I then ran the uhd_usrp_probe command again but got errors about the compatibility error:

```
# Error:  RuntimeError:  Expected firmware compatibility number 8.0, but got 7.0:
```

With information saying to run the same image downloader python script.

At this point I realized I messed up more than I can probably recover from easily. I tried to delete the images but wasn't double checking the folder I was in and removed the entire target directory that pybombs outputs to. So I'm going to cut my losses and go back to the Ettus page and install GNU radio per their instructions instead of fighting pybombs to get everything to work.

While going through the documentation trying to find the GNU Radio section, I realized there is a post-download set of instructions found here. I'm going to go through those now which include

```
# uhd_images_downloader
```

.

Also changing a configuration so non-root users can access the radio:

```
# cd <install-path>/lib/uhd/utils

# sudo cp uhd-usrp.rules /etc/udev/rules.d/

# sudo udevadm control --reload-rules

# sudo udevadm trigger
```

For my case, ¡install-path¿ was simply "usr".

I was hoping to just use apt-get to install gnuradio but that installs otherfiles not compatible with the UHD drivers installed earlier. Using the same remove command we used the last time we got the error message worked (apt-get remove gnuradio was all that was really needed though). Back to square one.

26 09-15-15

Last night before class I ran the GNURadio build script found here. After it ran I was still having issues. I tried re-running it to just install the UHD Drivers, but I had to go to class. When I got back it was waiting for a user acknolowedgement (press the Y key). So I don't believe it actually ran. When I got in today I ran

```
# sudo apt-get remove gnuradio
```

again to see if that would remove the build script install. It clearly stated it was removing something, but when I got back to the terminal and ran the USRP Probe command and then gnuradio suddenly everything was working. I'm assuming the one from sources never uninstalled and was actually the one that ended up running. Its likely I forgot to uninstall the one from the repositories and that that was the one that was causing issues. Either way, there is now "appropriate sounding static" on the audio from the FM block. I'll have to run further testing once I finish with TA stuff for the day to make sure its working. I should be able to broadcast FM and pick it up on the spectrum analyzers. I assume I will be unable to get FM in the VTC due to the nature of the building. I could always bring it outside to test though.

I am currently following this tutorial to try to get a transmitter setup: HERE

27 09-16-15

Today I succesfully managed to send FM radio broadcasts out of the SDR. The blocks can be found in this github repository. I followed the tutorials from Ettus Research.

28 09-18-15

I found this helpful presentation from WPI.

There's also the main webpage found Here

They have a textbook available as well Here

I have success creating a python program that slowly ramps up the frequency it is outputting. Using my trusty Tivoli Audio PAL radio I was able to keep turning the dial to keep the signal in tune. The key was to put a separate thread to handle this task. I realized this when I saw that the gui function used a callback. This prevents the function that changes the frequency from blocking the flow of data through the other filter blocks. This file is saved as FMThread.py and will be shown below.

Listing 2: dial_tone.py

```
#!/usr/bin/env python2
##################################################
# GNU Radio Python Flow Graph
# Title: Top Block
# Generated: Fri Sep 18 13:28:09 2015
```

```
#
# The goal of this python program is to force
# the frequency to slowly increase over time
# to prep for cognitive radio work
##################################################

if __name__ == '__main__':
    import ctypes
    import sys
    if sys.platform.startswith('linux'):
        try:
            x11 = ctypes.cdll.LoadLibrary('libX11.so')
            x11.XInitThreads()
        except:
            print "Warning: failed to XInitThreads()"

from gnuradio import analog
from gnuradio import blocks
from gnuradio import eng_notation
from gnuradio import filter
from gnuradio import gr
from gnuradio import uhd
from gnuradio.eng_option import eng_option
from gnuradio.filter import firdes
from gnuradio.wxgui import forms
from grc_gnuradio import wxgui as grc_wxgui
from optparse import OptionParser
import time
import wx
import threading

class top_block(grc_wxgui.top_block_gui):

    def __init__(self):
        grc_wxgui.top_block_gui.__init__(self, title="Top Block")
        _icon_path = "/usr/share/icons/hicolor/32x32/apps/gnuradio-grc.png"
        self.SetIcon(wx.Icon(_icon_path, wx.BITMAP_TYPE_ANY))

        ##################################################
        # Variables
        ##################################################
        self.samp_rate = samp_rate = 250e3
        self.freq = freq = 88.1e6
        self.audio_rate = audio_rate = 44100
        self.audio_interp = audio_interp = 4

        ##################################################
        # Blocks
        ##################################################
        _freq_sizer = wx.BoxSizer(wx.VERTICAL)
        self._freq_text_box = forms.text_box(
          parent=self.GetWin(),
          sizer=_freq_sizer,
          value=self.freq,
          callback=self.set_freq,
          label='freq',
          converter=forms.float_converter(),
          proportion=0,
        )
        self._freq_slider = forms.slider(
```

```
            parent=self.GetWin(),
            sizer=_freq_sizer,
            value=self.freq,
            callback=self.set_freq,
            minimum=88.0e6,
            maximum=107.9e6,
            num_steps=1000,
            style=wx.SL_HORIZONTAL,
            cast=float,
            proportion=1,
        )
        self.Add(_freq_sizer)
        self.uhd_usrp_sink_0 = uhd.usrp_sink(
            ",".join(("", "")),
            uhd.stream_args(
                cpu_format="fc32",
                channels=range(1),
            ),
        )
        self.uhd_usrp_sink_0.set_samp_rate(samp_rate)
        self.uhd_usrp_sink_0.set_center_freq(freq, 0)
        self.uhd_usrp_sink_0.set_normalized_gain(0.5, 0)
        self.uhd_usrp_sink_0.set_antenna("TX/RX", 0)
        self.rational_resampler_xxx_0 = filter.rational_resampler_ccc(
                interpolation=int(samp_rate * 1.0),
                decimation=audio_rate * audio_interp,
                taps=None,
                fractional_bw=None,
        )
        self.blocks_wavfile_source_0 = blocks.wavfile_source("/home/john/Downloads/documents-exp
        self.blocks_multiply_const_vxx_0 = blocks.multiply_const_vff((25, ))
        self.analog_wfm_tx_0 = analog.wfm_tx(
            audio_rate=audio_rate,
            quad_rate=audio_rate * audio_interp,
            tau=75e-6,
            max_dev=5e3,
        )

    thread = threading.Thread(target=self.sweep, args=())
    thread.daemon = True
    thread.start()

        ##################################################
        # Connections
        ##################################################
        self.connect((self.analog_wfm_tx_0, 0), (self.rational_resampler_xxx_0, 0))
        self.connect((self.blocks_multiply_const_vxx_0, 0), (self.analog_wfm_tx_0, 0))
        self.connect((self.blocks_wavfile_source_0, 0), (self.blocks_multiply_const_vxx_0, 0))
        self.connect((self.rational_resampler_xxx_0, 0), (self.uhd_usrp_sink_0, 0))


    def get_samp_rate(self):
        return self.samp_rate

    def set_samp_rate(self, samp_rate):
        self.samp_rate = samp_rate
        self.uhd_usrp_sink_0.set_samp_rate(self.samp_rate)

    def get_freq(self):
        return self.freq
```

```
    def set_freq(self, freq):
        self.freq = freq
        self._freq_slider.set_value(self.freq)
        self._freq_text_box.set_value(self.freq)
        self.uhd_usrp_sink_0.set_center_freq(self.freq, 0)

    def get_audio_rate(self):
        return self.audio_rate

    def set_audio_rate(self, audio_rate):
        self.audio_rate = audio_rate

    def get_audio_interp(self):
        return self.audio_interp

    def set_audio_interp(self, audio_interp):
        self.audio_interp = audio_interp

    def sweep(self):
    while True:
            start_freq = self.get_freq()
      self.set_freq(start_freq + 100000)
      print start_freq
      time.sleep(1)

if __name__ == '__main__':
    parser = OptionParser(option_class=eng_option, usage="%prog: [options]")
    (options, args) = parser.parse_args()
    tb = top_block()
    tb.Start(True)
    tb.Wait()
```