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.