Project Report

NSF REU Supplement Project: Testbed Development for AC-MWN: A Novel Architecture for Application-Aware Cognitive Multihop Wireless Networks

Undergraduate Students: Kossivi Tossou, Trenton Evans
Graduate Students: Feng Ye, Zhihui Shu, Mehrnaz Sharifian Esfahani
PIs: Dr. Yi Qian, Dr. Yaoqing Yang, Dr. Hamid Sharif

Communication Networks & Security Lab
University of Nebraska-Lincoln
Febrary 11, 2014

Abstract

Given the recent upward trend in wireless traffic, capacity demand increases faster than spectral efficiency and availability. However, it has been well-known that in wireless communications, some of the spectra are significantly under-utilized. This fact has attracted researchers from academia and industry who are interested in the next generation of cognitive radio communication systems. Despite the tremendous ongoing efforts on this research topic, there is a huge gap between the research on cognitive radios and the network applications. This project exploits a novel wireless networking paradigm called Application-Aware Cognitive Multihop Wireless Networks (AC-MWN). Besides theoretical analysis, a testbed has been developed in this project to demonstrate some of the theoretical results. The testbed applications are established using USRP smart radios to apply cognitive radio technique in various applications, such as file transferring, voice chat and video conference.

Keywords: cognitive radio, multi-hop, testbed

CONTENTS

CHAPTER 1 INTRODUCTION

1.1 Background

Cognitive radio network (CRN) is a promising technology based on spectrum sensing and opportunistic spectrum access which can improve the spectrum efficiency of wireless networks [1]. The secondary users would sense the activities of primary users periodically according to Federal Communications Commission (FCC) [2]. When a channel is not occupied by a primary user, a secondary user can use the channel opportunistically after sensing. The major characteristic of cognitive radios is that the activities of primary users change dynamically and channel availability changes from time to time. Thus, the secondary users have to efficiently sense the activities of the primary users and make decisions on which channel to use in the dynamic changing environment. In order to study the performance of the cognitive radios, we investigate how to develop a testbed and implement algorithms on the testbed.

Multi-hop cognitive radio networks bring many challenges in testbed design.
First, transmissions on different hops might interfere with each other. Therefore, we investigate how to implement multi-hop transmission on different channels using USRP2 and USRP N210. Second, secondary users have to sense the activities of the primary users instantly to get the availability information of the channels. Thus, we need to develop a practical sensing scheme to find whether the primary users are working or not. After one available channel is found, the secondary users have to switch to this channel as soon as possible. Third, the routing and channel allocation schemes have to be embedded on testbed. We need to implement our proposed routing and allocation schemes on testbed to show the effectiveness of our proposed algorithms.

To verify the theoretical performance in multi-hop cognitive radio networks, we develop a testbed in this project for proof-concept. First, we set up the system so that the USRP can be interfaced with GNU radio. Second, we study several different transmission schemes including single-hop and multi-hop transmissions based on USRP and smart radios. Then, we investigate how to perform spectrum sensing using cognitive radios. Finally, we implement our routing and channel allocation schemes on our testbed.

1.2 Equipment

The testbed consists of three laptops, two USRP2-N210 SDRs, and one USRP2 SDR. Each laptop is running Ubuntu version 13.10 64-bit operating system with gnuradio software package. The radios were equipped with firmware and WBX-120 RF daughterboards.

1.3 Goals

The main tasks of this REU project include 1) Computer programming and network configuration using Universal Software Radio Peripheral (USRP) hardware; and 2) Performance evaluation through experiments and measurements. We will complete the followings:

1. Basic functions
    a. Na ïve one-way transmission
    b. Na ïve half-duplex transmission
    c. Multi-channel one-way transmission
    d. Na ïve full-duplex transmission
    e. Adaptive one-way transmission
    f. Adaptive half-duplex transmission
2. Sensing
    a. Sensing if primary user has activity at a given channel
    b. Determine a threshold so that false alarm shall be the minimum
3. Cognitive radio technique application
    a. Real-time adaptive channel switching
4. Multi-hop applications
    a. File transferring, voice chat, and video conference applications using cognitive radio technique

CHAPTER 2 BASIC FUNCTIONS

2.1 Equipment Setup

There are three main things that must be done to setup the system so that the USRP can be interfaced with GNU radio. These include:

- Download GNU radio software and packages, and once installation is complete use "gnuradio-companion" to run the graphical programming interface of GNU radio.
- Download required packages for building the UHD driver.
- Configure the network interface by assigning a static IP address to the host computer Ethernet interface card connected to the USRP2 with *$ ifconfig eth0 192.168.10.1* and finally to test connection with *$ ping 192.168.10.2*.

A test example in GNU Radio Companion interface "FM Receiver with USRP2" is attached below. The example shows how to use the Universal Software Radio Peripheral (USRP2) device with GNU radio to build an FM receiver. One radio with one antenna is set up as shown in Fig. 2-1.

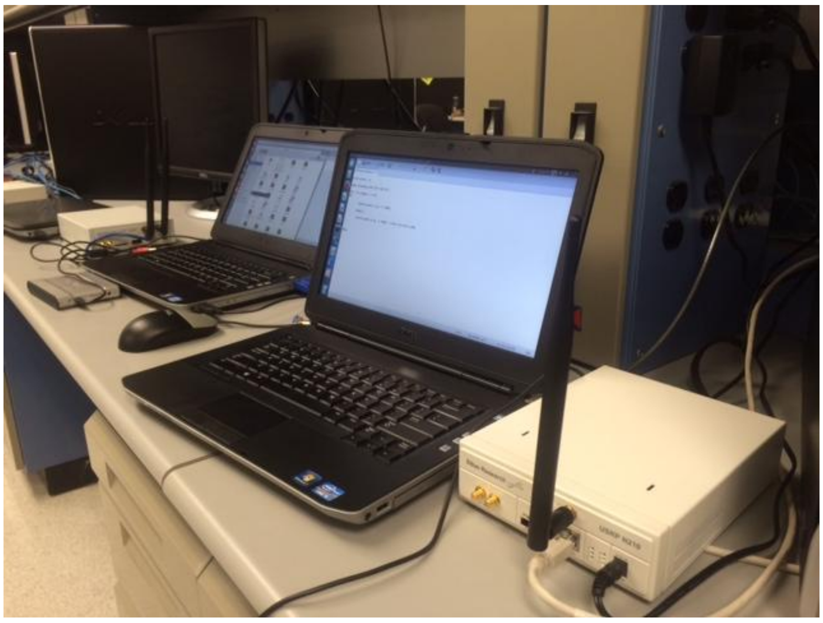
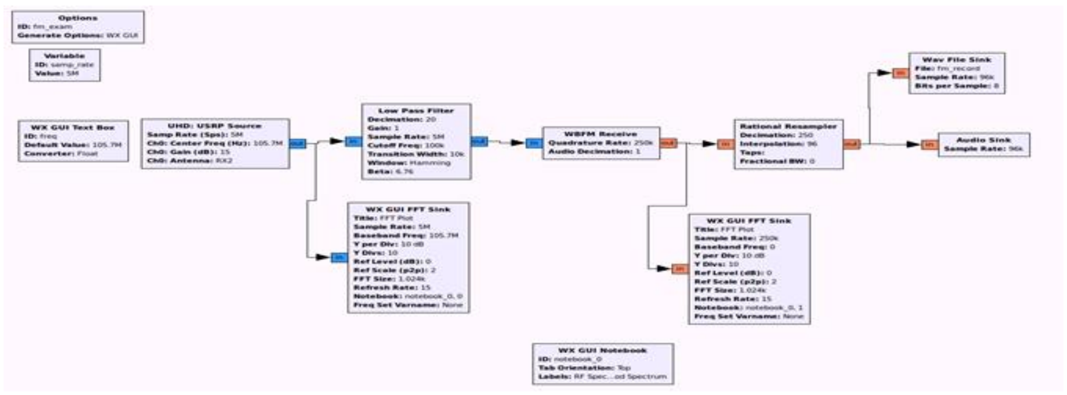


Figure 2-1



Options
ID: fm_exam
Generate Options: WX GUI

Variable
ID: samp_rate
Value: 5M

WX GUI Text Box
ID: freq
Default Value: 105.7M
Converter: Float

UHD: USRP Source
Samp Rate (Sps): 5M
Ch0: Center Freq (Hz): 105.7M
Ch0: Gain (dB): 15
Ch0: Antenna: RX2

Low Pass Filter
Decimation: 20
Gain: 1
Sample Rate: 5M
Cutoff Freq: 100k
Transition Width: 10k
Window: Hamming
Beta: 6.76

WBFM Receive
Quadrature Rate: 250k
Audio Decimation: 1

Rational Resampler
Decimation: 250
Interpolation: 96
Taps:
Fractional BW: 0

Wav File Sink
File: fm_record
Sample Rate: 96k
Bits per Sample: 8

Audio Sink
Sample Rate: 96k

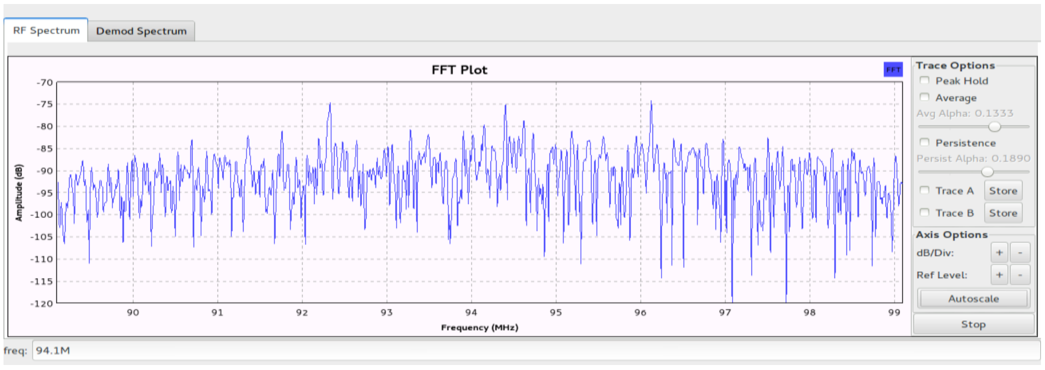
WX GUI FFT Sink
Title: FFT Plot
Sample Rate: 5M
Baseband Freq: 105.7M
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 0
Ref Scale (p2p): 2
FFT Size: 1.024k
Refresh Rate: 15
Notebook: notebook_0, 0
Freq Set Varname: None

WX GUI FFT Sink
Title: FFT Plot
Sample Rate: 250k
Baseband Freq: 0
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 0
Ref Scale (p2p): 2
FFT Size: 1.024k
Refresh Rate: 15
Notebook: notebook_0, 1
Freq Set Varname: None

WX GUI Notebook
ID: notebook_0
Tab Orientation: Top
Labels: RF Spec....od Spectrum

Figure 2-2

Output



freq: 94.1M

Figure 2-3

2.2 Single-Hop Transmission

2.2.1 Half-duplex one-way transmission

In this part, we are going to set up one radio as the transmitter and one for receiving, and successfully send and receive verifiable data. The data is verified at the receiving end when benchmark_rx.py prints the message and how many successful packets were received.

a. Two smart radios, each equips one antenna. One smart radio is set up for transmitting, the other for receiving, single dedicated channel (e.g., FM radio)

We successfully achieved this milestone by transmitting and receiving based on frequencies, by packages transfer and data transfer. We used benchmark code under our gnuradio folder to accomplish these tasks.

- First we composed a .odb file with the sentence "*It is a beautiful day") .* The .odb is associated with 'OpenOffice/StarOffice in Ubuntu (It is like notepad in Windows system) which we saved in the same folder of our benchmark code

  (*usr/local/share/gnuradio/examples/digital/ofdm*).

Then we modified the benchmark_tx.py and benchmark_rx.py code to conform as transmitting and receiving a data file. And finally we ran the scripts below from the transmission terminal and reception terminal:

./benchmark_tx.py -f 900M - - from-file=nice.odb

./benchmark_rx.py -f 900M

The settings and results are shown below:

Setting: two radios with one antenna each

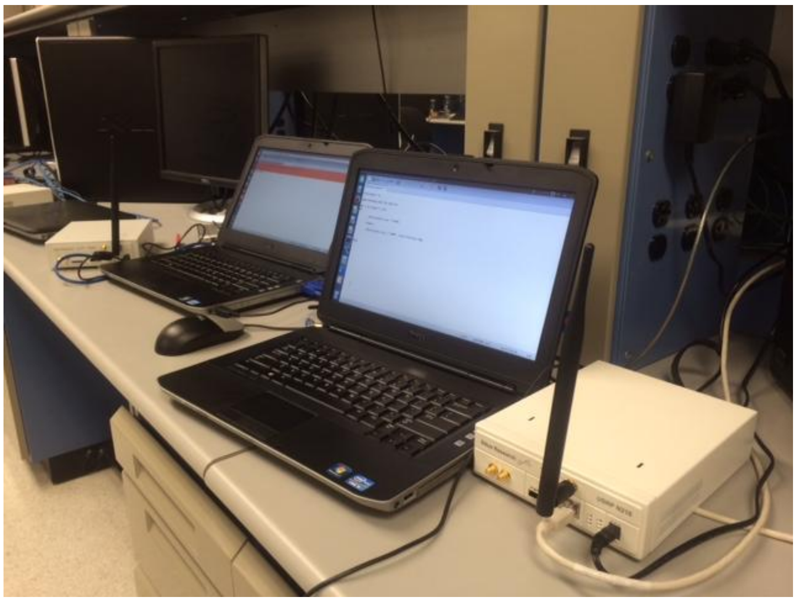
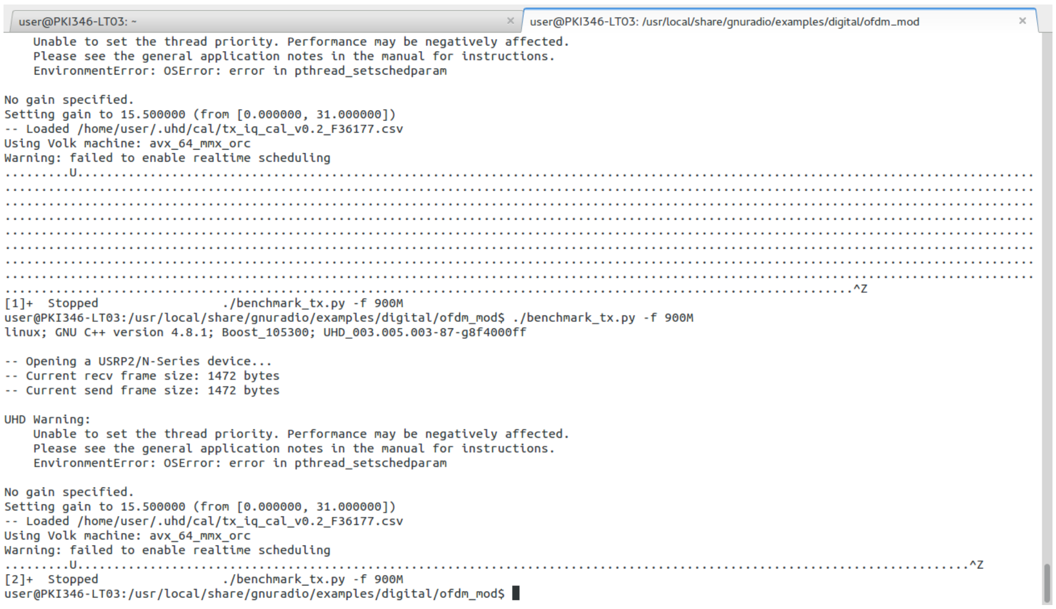


Figure 2-4

Output at the transmitter radio without sending file



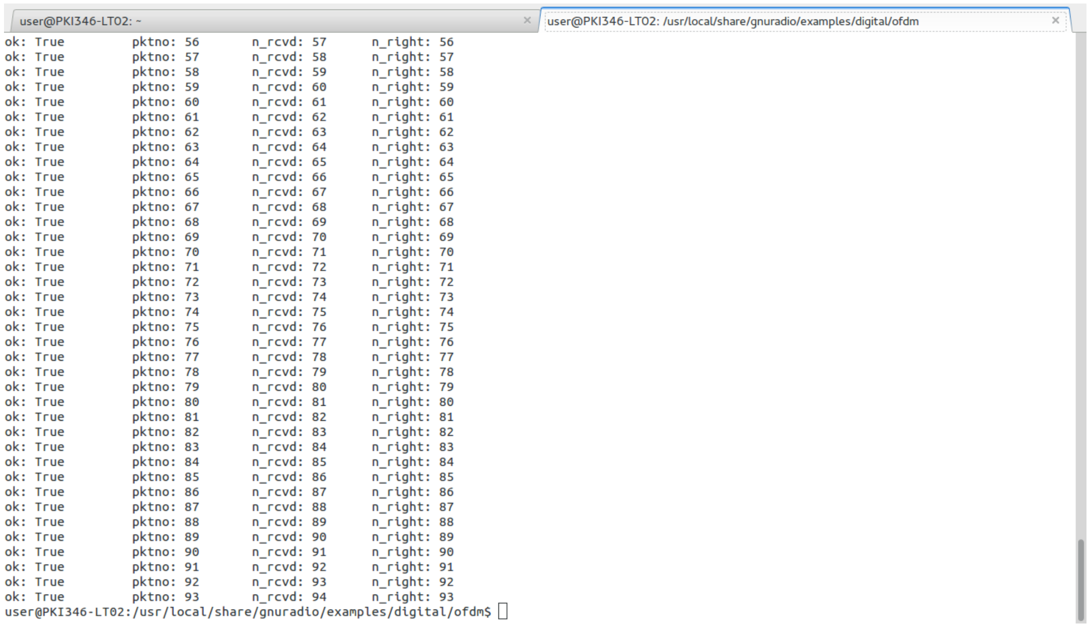
user@PKI346-LT03: ~

user@PKI346-LT03: /usr/local/share/gnuradio/examples/digital/ofdm_mod

```
    Unable to set the thread priority. Performance may be negatively affected.
    Please see the general application notes in the manual for instructions.
    EnvironmentError: OSError: error in pthread_setschedparam

No gain specified.
Setting gain to 15.500000 (from [0.000000, 31.000000])
-- Loaded /home/user/.uhd/cal/tx_iq_cal_v0.2_F36177.csv
Using Volk machine: avx_64_mmx_orc
Warning: failed to enable realtime scheduling
.........U....................................................................................................................................
..............................................................................................................................................
..............................................................................................................................................
..............................................................................................................................................
..............................................................................................................................................
..............................................................................................................................................
..............................................................................................................................................
..............................................................................................................................................
.....................................................................................................................^Z
[1]+  Stopped                 ./benchmark_tx.py -f 900M
user@PKI346-LT03:/usr/local/share/gnuradio/examples/digital/ofdm_mod$ ./benchmark_tx.py -f 900M
linux; GNU C++ version 4.8.1; Boost_105300; UHD_003.005.003-87-g8f4000ff

-- Opening a USRP2/N-Series device...
-- Current recv frame size: 1472 bytes
-- Current send frame size: 1472 bytes

UHD Warning:
    Unable to set the thread priority. Performance may be negatively affected.
    Please see the general application notes in the manual for instructions.
    EnvironmentError: OSError: error in pthread_setschedparam

No gain specified.
Setting gain to 15.500000 (from [0.000000, 31.000000])
-- Loaded /home/user/.uhd/cal/tx_iq_cal_v0.2_F36177.csv
Using Volk machine: avx_64_mmx_orc
Warning: failed to enable realtime scheduling
.........U...........................................................................................................................^Z
[2]+  Stopped                 ./benchmark_tx.py -f 900M
user@PKI346-LT03:/usr/local/share/gnuradio/examples/digital/ofdm_mod$ 
```

Figure 2-5

Output at the receiver radio without sending file

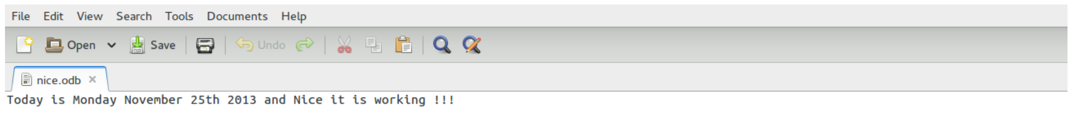


user@PKI346-LT02: ~ | user@PKI346-LT02: /usr/local/share/gnuradio/examples/digital/ofdm

```
ok: True        pktno: 56       n_rcvd: 57      n_right: 56
ok: True        pktno: 57       n_rcvd: 58      n_right: 57
ok: True        pktno: 58       n_rcvd: 59      n_right: 58
ok: True        pktno: 59       n_rcvd: 60      n_right: 59
ok: True        pktno: 60       n_rcvd: 61      n_right: 60
ok: True        pktno: 61       n_rcvd: 62      n_right: 61
ok: True        pktno: 62       n_rcvd: 63      n_right: 62
ok: True        pktno: 63       n_rcvd: 64      n_right: 63
ok: True        pktno: 64       n_rcvd: 65      n_right: 64
ok: True        pktno: 65       n_rcvd: 66      n_right: 65
ok: True        pktno: 66       n_rcvd: 67      n_right: 66
ok: True        pktno: 67       n_rcvd: 68      n_right: 67
ok: True        pktno: 68       n_rcvd: 69      n_right: 68
ok: True        pktno: 69       n_rcvd: 70      n_right: 69
ok: True        pktno: 70       n_rcvd: 71      n_right: 70
ok: True        pktno: 71       n_rcvd: 72      n_right: 71
ok: True        pktno: 72       n_rcvd: 73      n_right: 72
ok: True        pktno: 73       n_rcvd: 74      n_right: 73
ok: True        pktno: 74       n_rcvd: 75      n_right: 74
ok: True        pktno: 75       n_rcvd: 76      n_right: 75
ok: True        pktno: 76       n_rcvd: 77      n_right: 76
ok: True        pktno: 77       n_rcvd: 78      n_right: 77
ok: True        pktno: 78       n_rcvd: 79      n_right: 78
ok: True        pktno: 79       n_rcvd: 80      n_right: 79
ok: True        pktno: 80       n_rcvd: 81      n_right: 80
ok: True        pktno: 81       n_rcvd: 82      n_right: 81
ok: True        pktno: 82       n_rcvd: 83      n_right: 82
ok: True        pktno: 83       n_rcvd: 84      n_right: 83
ok: True        pktno: 84       n_rcvd: 85      n_right: 84
ok: True        pktno: 85       n_rcvd: 86      n_right: 85
ok: True        pktno: 86       n_rcvd: 87      n_right: 86
ok: True        pktno: 87       n_rcvd: 88      n_right: 87
ok: True        pktno: 88       n_rcvd: 89      n_right: 88
ok: True        pktno: 89       n_rcvd: 90      n_right: 89
ok: True        pktno: 90       n_rcvd: 91      n_right: 90
ok: True        pktno: 91       n_rcvd: 92      n_right: 91
ok: True        pktno: 92       n_rcvd: 93      n_right: 92
ok: True        pktno: 93       n_rcvd: 94      n_right: 93
user@PKI346-LT02:/usr/local/share/gnuradio/examples/digital/ofdm$ 
```

Figure 2-6

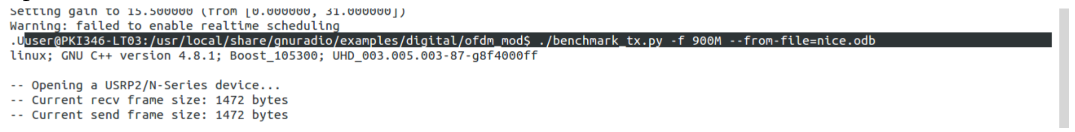
Odb file



```
Today is Monday November 25th 2013 and Nice it is working !!!
```

Figure 2-7

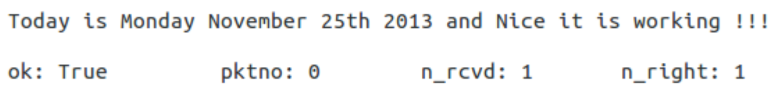
Output at the transmitter radio with Nice.odb file



```
Setting gain to 15.500000 (from [0.000000, 31.000000])
Warning: failed to enable realtime scheduling
.Uuser@PKI346-LT03:/usr/local/share/gnuradio/examples/digital/ofdm_mod$ ./benchmark_tx.py -f 900M --from-file=nice.odb
linux; GNU C++ version 4.8.1; Boost_105300; UHD_003.005.003-87-g8f4000ff

-- Opening a USRP2/N-Series device...
-- Current recv frame size: 1472 bytes
-- Current send frame size: 1472 bytes
```

Figure 2-8

Output at the receiver radio with nice.odb file



```
Today is Monday November 25th 2013 and Nice it is working !!!

ok: True        pktno: 0        n_rcvd: 1       n_right: 1
```

Figure 2-9

2.2.1.1 Adaptive One-Way Transmission

In this part, we are going to have two smart radios, each equipped with one antenna. One smart radio is set up for transmitting, the other for receiving, single dedicated channel, but the channel can be switched according to some algorithm automatically (e.g., FM radio, but will change channel once every 10 seconds)

Setting: two radios with two antennas

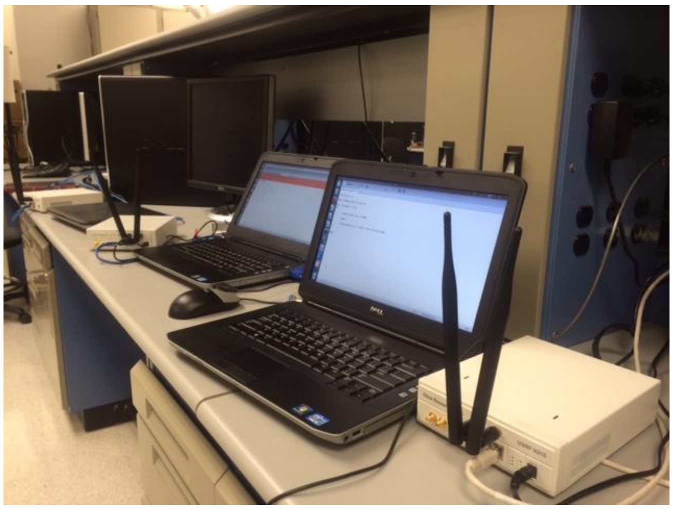
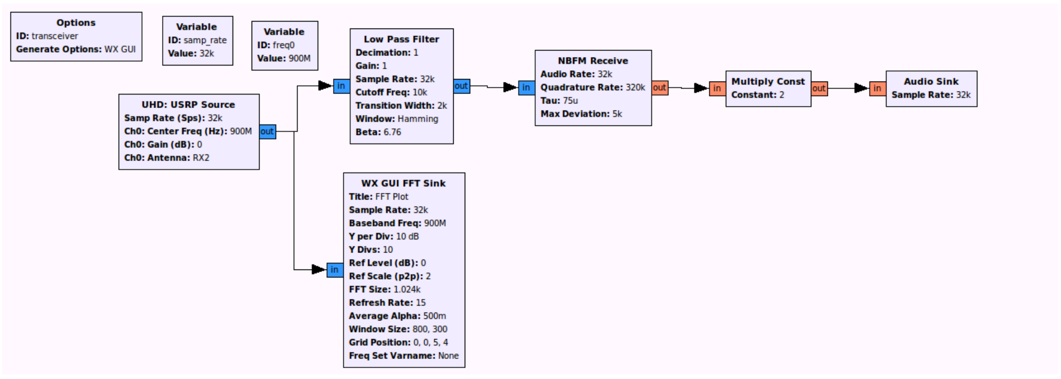


Figure 2-10

GRC companion for Radio 1: Receiving blocks



Options
ID: transceiver
Generate Options: WX GUI

Variable
ID: samp_rate
Value: 32k

Variable
ID: freq0
Value: 900M

UHD: USRP Source
Samp Rate (Sps): 32k
Ch0: Center Freq (Hz): 900M
Ch0: Gain (dB): 0
Ch0: Antenna: RX2

Low Pass Filter
Decimation: 1
Gain: 1
Sample Rate: 32k
Cutoff Freq: 10k
Transition Width: 2k
Window: Hamming
Beta: 6.76

NBFM Receive
Audio Rate: 32k
Quadrature Rate: 320k
Tau: 75u
Max Deviation: 5k

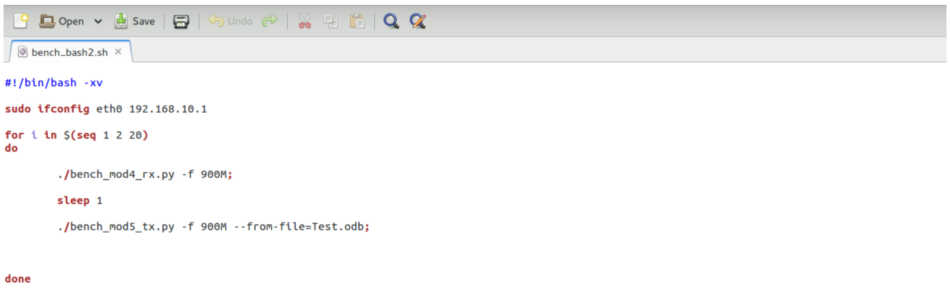
Multiply Const
Constant: 2

Audio Sink
Sample Rate: 32k

WX GUI FFT Sink
Title: FFT Plot
Sample Rate: 32k
Baseband Freq: 900M
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 0
Ref Scale (p2p): 2
FFT Size: 1.024k
Refresh Rate: 15
Average Alpha: 500m
Window Size: 800, 300
Grid Position: 0, 0, 5, 4
Freq Set Varname: None

Figure 2-11

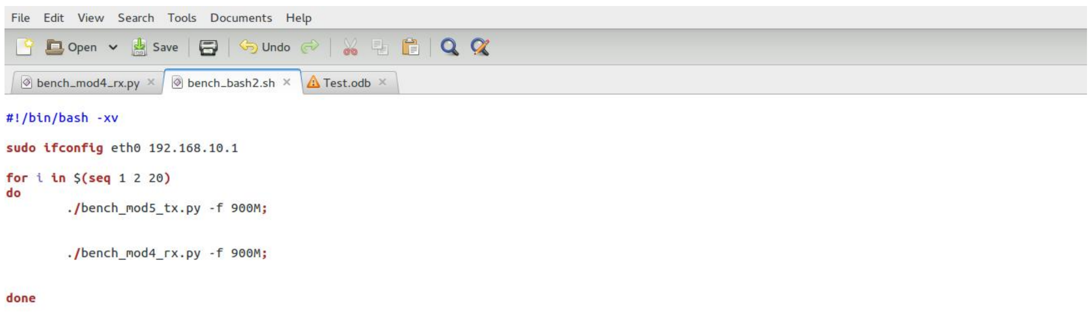
We ran the bash script and we can see that each radio was taking turn to transmit and receive automatically. We are working on the correct synchronization and time delay to perfect this half duplex transmission. Bash script at the transmitter



```
#!/bin/bash -xv

sudo ifconfig eth0 192.168.10.1

for i in $(seq 1 2 20)
do

        ./bench_mod4_rx.py -f 900M;

        sleep 1

        ./bench_mod5_tx.py -f 900M --from-file=Test.odb;


done
```

Figure 2-13

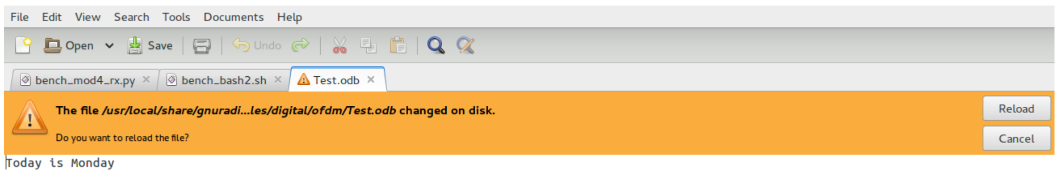
Bash script at the receiver



```
#!/bin/bash -xv

sudo ifconfig eth0 192.168.10.1

for i in $(seq 1 2 20)
do
        ./bench_mod5_tx.py -f 900M;


        ./bench_mod4_rx.py -f 900M;


done
```

Figure 2-14

Output



The file /usr/local/share/gnuradi...les/digital/ofdm/Test.odb changed on disk.

Do you want to reload the file?

```
Today is Monday
```

Figure 2-15

Setting: two radios with one antenna each

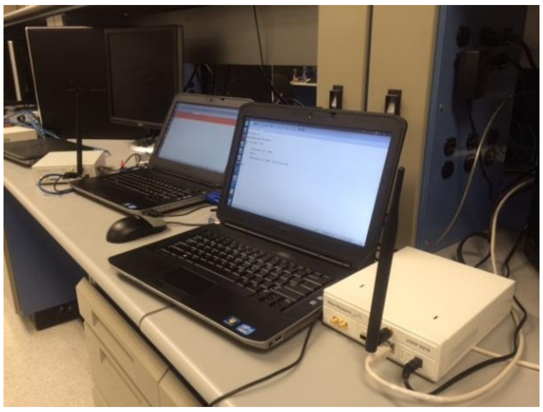


Figure 2-16

Then, we are going to have two radios with two antennas each. Radio 1 will transmit using the TX/RX antenna while Radio 2 will receive on the same channel using the RX2 antenna. Next, Radio 2 will be set up to transmit on a different channel using its TX/RX antenna while Radio 1 receives on the same channel using the RX2 antenna.

First, we test one-way multi-channel transmission. The settings include two smart radios, each equips two antennas. One smart radio is set up for transmitting; the other for receiving, transmission is set up using two channels through two antennas simultaneously.

Setting: two radios with two antennas

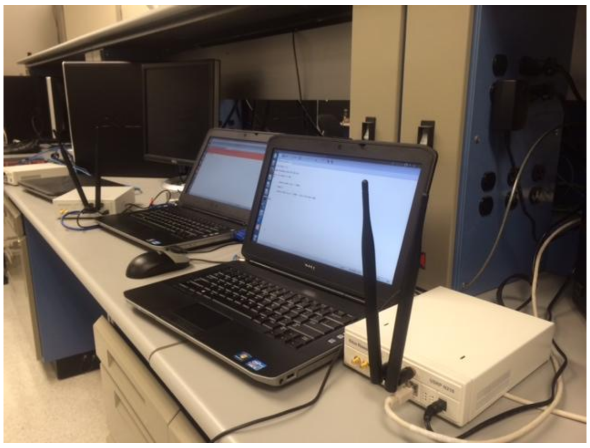


Figure 2-17

We repeated the same design and procedure with the Naïve one-way transmission but with two antennas.

2.2.3 Full-Duplex Transmission

In this part, we are going to achieve full-duplex transmission with two smart radios, each equips two antennas. One antenna of a smart radio is set up for transmitting; the other antenna of this smart radio is set up for receiving. Two-way transmission should be achieved.

We also worked on IO file management between the transmission and receiving radios, and then added an acknowledgement time when either receiver or transmitter received or transmitted in the proper folder designed in advance by using the benchmark scripts
The objective was to save the data in text file after receiving in from the transmitting radio; then to resend the same data in the text file to the first radio with acknowledgement time and date of reception.

We accomplished this by adding in our original script:

- Transmission script: (needed to be added to the benchmark_tx script)

  def *read_dir():*
  If os.path.exist("")
  file = open('', '') - Opens a file on the specified path with read/write access.
  ts = str(datetime.datetime.now()) – print the actual time
  file.write(ts)
  global ack
  file.close()

- Reception script: (needed to be added in the benchmark_rx script)

  def *ensure_dir():*
  If not os.path.exist("")
  Os.mkdir("")
  file = open('', '') - Opens a file on the specified path with read/write access.
  gobal x

x = payload -- - data to be received

file.write(x)-- - save data in the specified path with read/write access

file.close()

Setting: two radios with two antennas

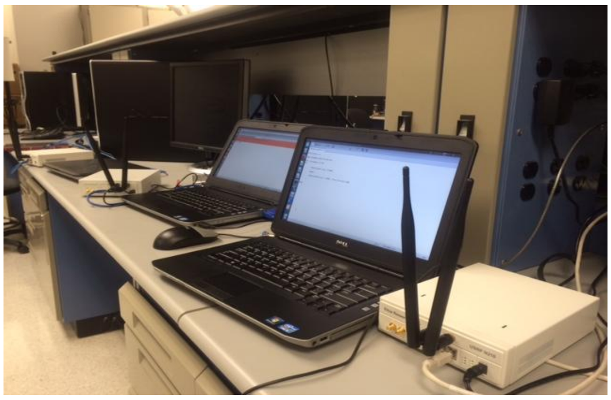
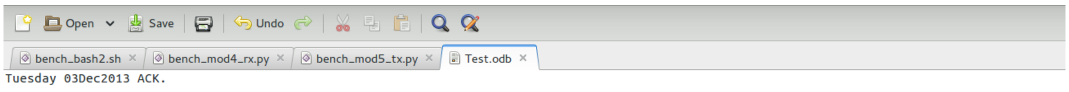


Figure 2-18

Odb file on the transmitter side.



```
Tuesday 03Dec2013 ACK.
```

Figure 2-19

Empty folder to send the odb file to. Cd /usr/local/share/gnuradio/examples/digital/ofdm_mod

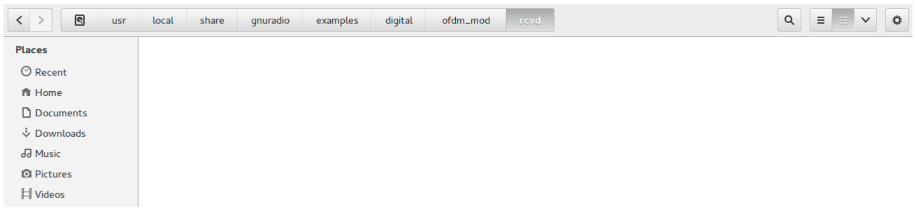


Figure 2-20

Output of command prompt on the receiver side

```
^Cuser@PKI346-LT03:/usr/local/share/gnuradio/examples/digital/ofdm_mod$ ./bench_mod4_rx.py -f 900M
linux; GNU C++ version 4.8.1; Boost_105300; UHD_003.005.003-87-g8f4000ff

-- Opening a USRP2/N-Series device...
-- Current recv frame size: 1472 bytes
-- Current send frame size: 1472 bytes

UHD Warning:
    Unable to set the thread priority. Performance may be negatively affected.
    Please see the general application notes in the manual for instructions.
    EnvironmentError: OSError: error in pthread_setschedparam

No gain specified.
Setting gain to 19.000000 (from [0.000000, 38.000000])
Using Volk machine: avx_64_mmx_orc
>>> gr_fir_ccf: using SSE
>>> gr_fir_fff: using SSE
Warning: failed to enable realtime scheduling

ok: True         pktno: 0         n_rcvd: 1         n_right: 100
```

Output after creating the file to. Cd /usr/local/share/gnuradio/examples/digital/ofdm_mod

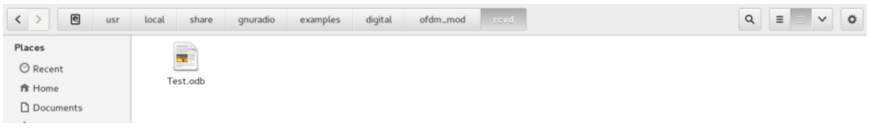
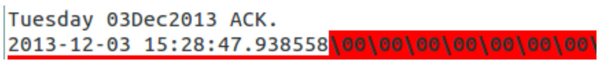


Figure 2-21

File with datetime acknowledgement.



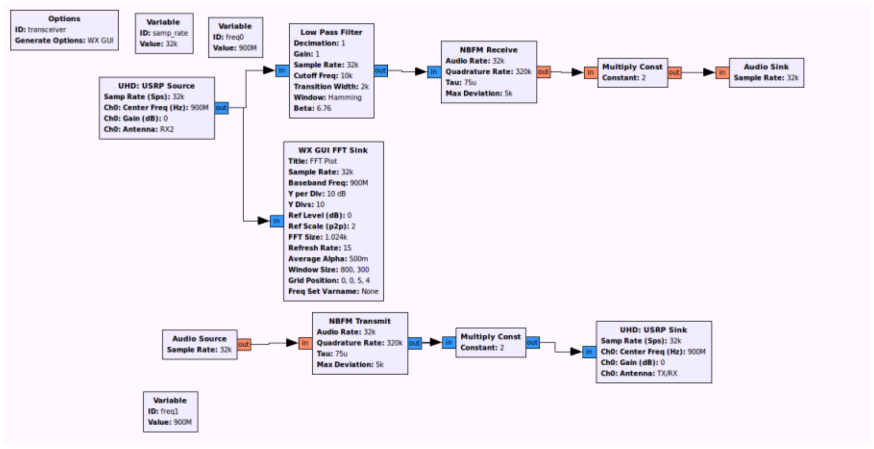
```
Tuesday 03Dec2013 ACK.
2013-12-03 15:28:47.938558\00\00\00\00\00\00\00\
```

Figure 2-22

2.2.3.1 Adaptive Full-Duplex Transmission

In this part, we are going to set up hardware is Similar to 5, but we use gnuradio-companion flow chart software to utilize the built in threading capabilities.

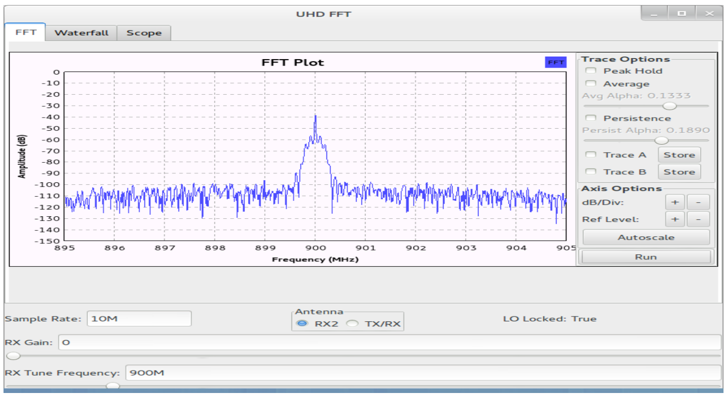
The settings are similar to 5 but the flow charts shown in Figures 2-11 & 2-12 are combined to make one flow chart to create a transceiver.



```
Options
ID: transceiver
Generate Options: WX GUI

Variable
ID: samp_rate
Value: 32k

Variable
ID: freq0
Value: 900M

UHD: USRP Source
Samp Rate (Sps): 32k
Ch0: Center Freq (Hz): 900M
Ch0: Gain (dB): 0
Ch0: Antenna: RX2

Low Pass Filter
Decimation: 1
Gain: 1
Sample Rate: 32k
Cutoff Freq: 10k
Transition Width: 2k
Window: Hamming
Beta: 6.76

NBFM Receive
Audio Rate: 32k
Quadrature Rate: 320k
Tau: 75u
Max Deviation: 5k

Multiply Const
Constant: 2

Audio Sink
Sample Rate: 32k

WX GUI FFT Sink
Title: FFT Plot
Sample Rate: 32k
Baseband Freq: 900M
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 0
Ref Scale (p2p): 2
FFT Size: 1.024k
Refresh Rate: 15
Average Alpha: 500m
Window Size: 800, 300
Grid Position: 0, 0, 5, 4
Freq Set Varname: None

Audio Source
Sample Rate: 32k

NBFM Transmit
Audio Rate: 32k
Quadrature Rate: 320k
Tau: 75u
Max Deviation: 5k

Multiply Const
Constant: 2

UHD: USRP Sink
Samp Rate (Sps): 32k
Ch0: Center Freq (Hz): 900M
Ch0: Gain (dB): 0
Ch0: Antenna: TX/RX

Variable
ID: freq1
Value: 900M
```

Figure 2-23

Output is



Sample Rate: 10M
Antenna [✓ RX2] TX/RX
LO Locked: True
RX Gain: 0
RX Tune Frequency: 900M

Figure 2-24

2.3 Multi-Hop Transmission

2.3.1 Simple Transceiver Transmission with Two Smart Radios

In this part, we are going to use two radios setup with the grc file shown in Figure 2-26 and both channels are observed with the FFT GUI as shown in Figure 2-27.

The settings are shown below:
The block below was designed in gnuradio companion. It consisted of two separated blocks, one for the transmission and the other for the receiving. The grc components used for this design are:

One radio set up with one laptop

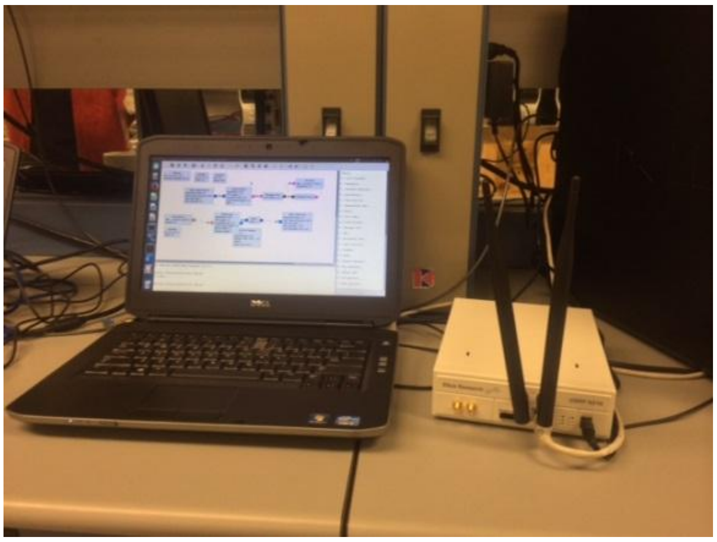
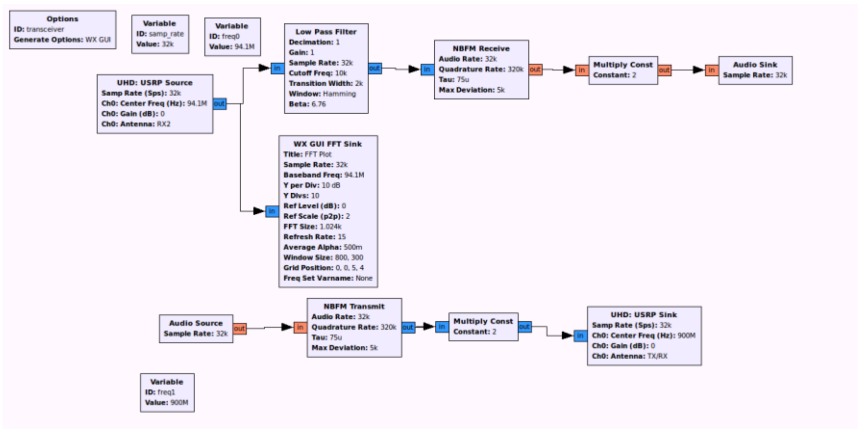


Figure 2-25

Transmission blocks:
Audio Source, NBFM Transmit, Multiply Const, UHD: USRP Sink
Receiving blocks:
UHD: USRP Source, Low Pass filter, NBFM Receive, Audio Sink and WX GUI FFT Sink



```
Options
ID: transceiver
Generate Options: WX GUI

Variable
ID: samp_rate
Value: 32k

Variable
ID: freq0
Value: 94.1M

UHD: USRP Source
Samp Rate (Sps): 32k
Ch0: Center Freq (Hz): 94.1M
Ch0: Gain (dB): 0
Ch0: Antenna: RX2

Low Pass Filter
Decimation: 1
Gain: 1
Sample Rate: 32k
Cutoff Freq: 10k
Transition Width: 2k
Window: Hamming
Beta: 6.76

NBFM Receive
Audio Rate: 32k
Quadrature Rate: 320k
Tau: 75u
Max Deviation: 5k

Multiply Const
Constant: 2

Audio Sink
Sample Rate: 32k

WX GUI FFT Sink
Title: FFT Plot
Sample Rate: 32k
Baseband Freq: 94.1M
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 0
Ref Scale (p2p): 2
FFT Size: 1.024k
Refresh Rate: 15
Average Alpha: 500m
Window Size: 800, 300
Grid Position: 0, 0, 5, 4
Freq Set Varname: None

Audio Source
Sample Rate: 32k

NBFM Transmit
Audio Rate: 32k
Quadrature Rate: 320k
Tau: 75u
Max Deviation: 5k

Multiply Const
Constant: 2

UHD: USRP Sink
Samp Rate (Sps): 32k
Ch0: Center Freq (Hz): 900M
Ch0: Gain (dB): 0
Ch0: Antenna: TX/RX

Variable
ID: freq1
Value: 900M
```

Figure 2-26

Finally we generated flow graph and then executed, and we had the output FFT plot of Transceiver block design with 1 MHz bandwidth as shown in Fig. 2-27.

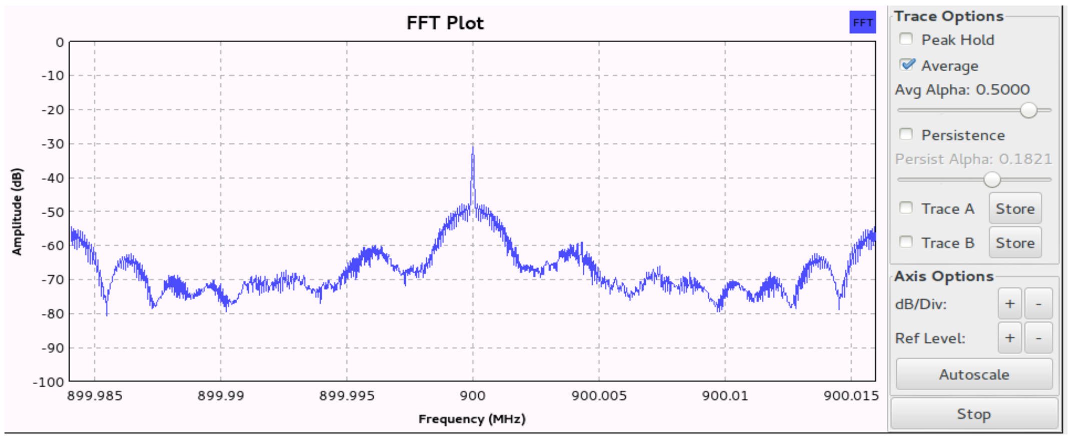


Figure 2-27

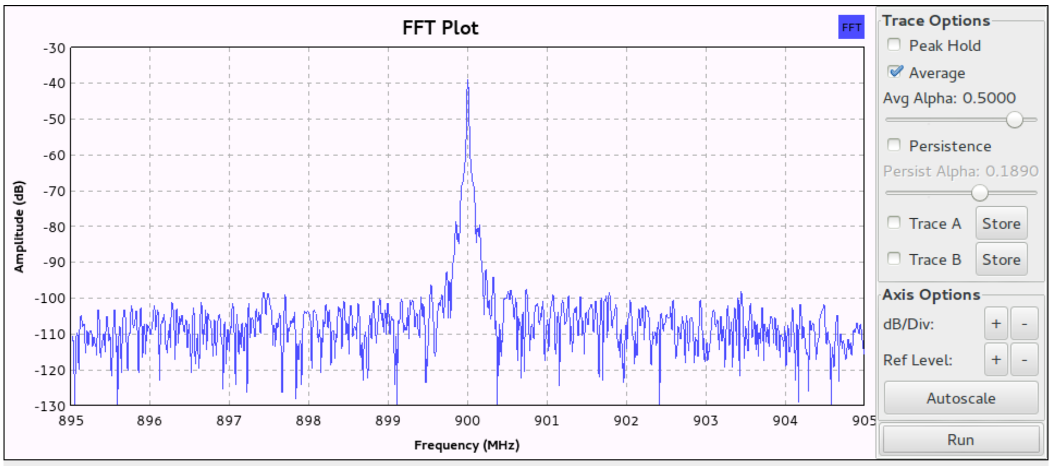


Figure 2-28

The output FFT plot of Transceiver block design with 10 MHz bandwidth at 900 MHz is shown in Fig. 2-28. The output FFT plot of Transceiver block design with 10 MHz bandwidth at 920 MHz is shown in Fig. 2-29.

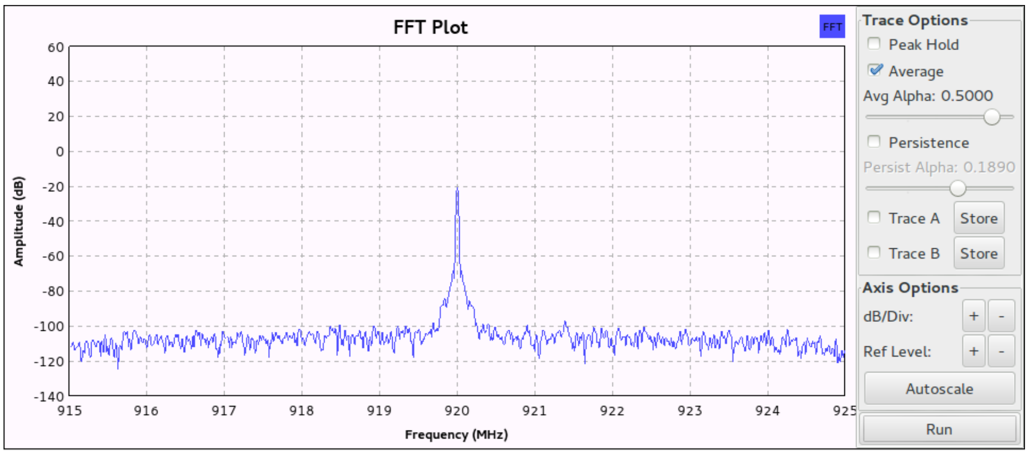


Figure 2-29

2.3.2 Two Radios Multi-Hop (Round-Trip) Transmission

In this part, we are going to use two radios, one with our regular transceiver grc file and another with a relay file. The relay is set up to receive on one frequency and immediately transmit the data on another frequency. Therefore, the data is transmitted from Radio 1 and channel A, received by Radio 2 on channel A then transmitted by Radio 2 on channel B, and finally, received by Radio 1 on channel B.

In this section we designed a full duplex transmission with data exchange between two radios. We used specific blocks as OFDM, File Source and File sink. The full duplex was tested and it worked perfectly by exchanging data from the receiving radio to the transmission radio in a folder we pre-created for the circumstance.

Setting:
Two radios set up with one radio

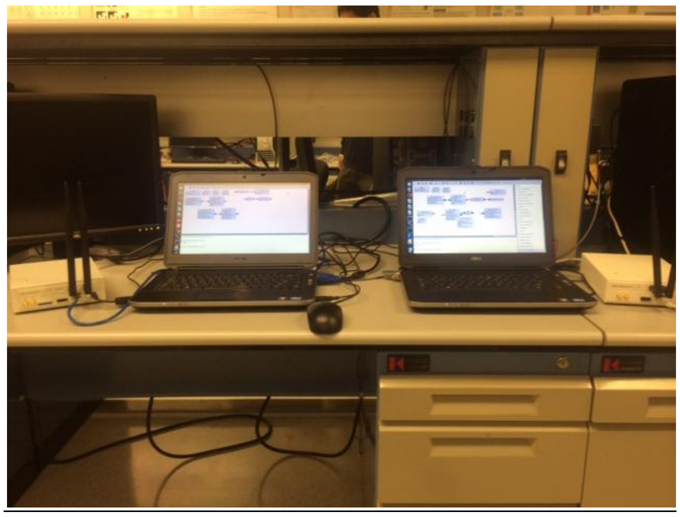


Figure 2-30

- The first radio sent a data information with two sets of blocks in GRC companion (a receiver and transmitter blocs).
- The second received the data sent, saved it and resent it to the first radio and this is loop.

Radio 1:
Transmission frequency: 950MHz
Receiving frequency: 900MHz

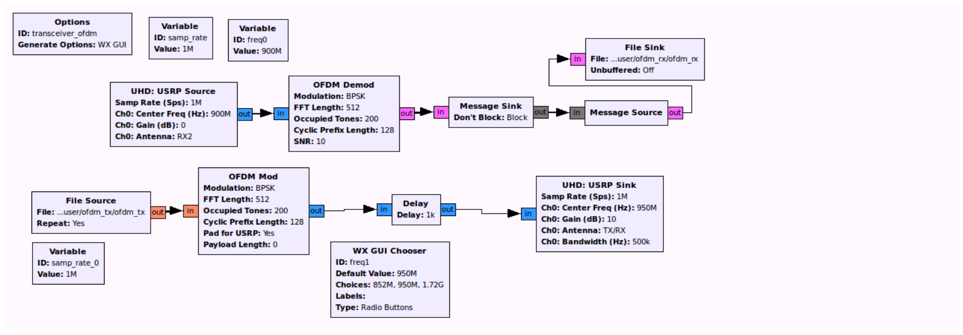
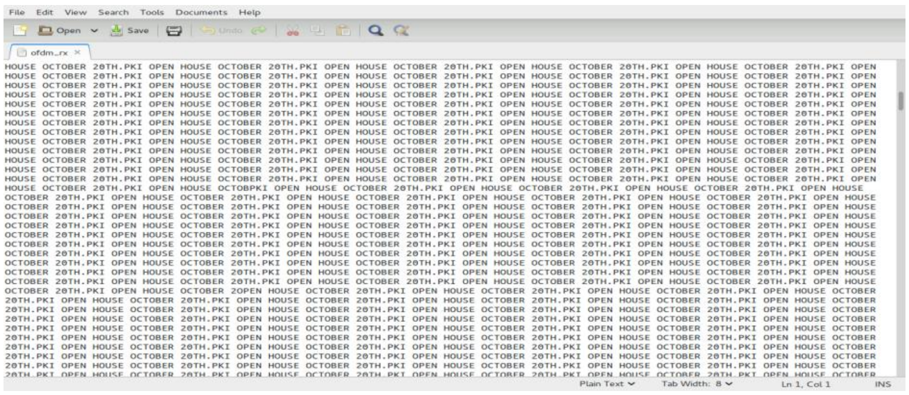


Figure 2-31

File source:

We created an ofdm_tx folder with the sentence “PKI OPEN HOUSE…” on directory /home/user/ofdm_tx. File sink (where the file is received after the second radio resent what it received.) /home/user/ofdm_rx.

Output:



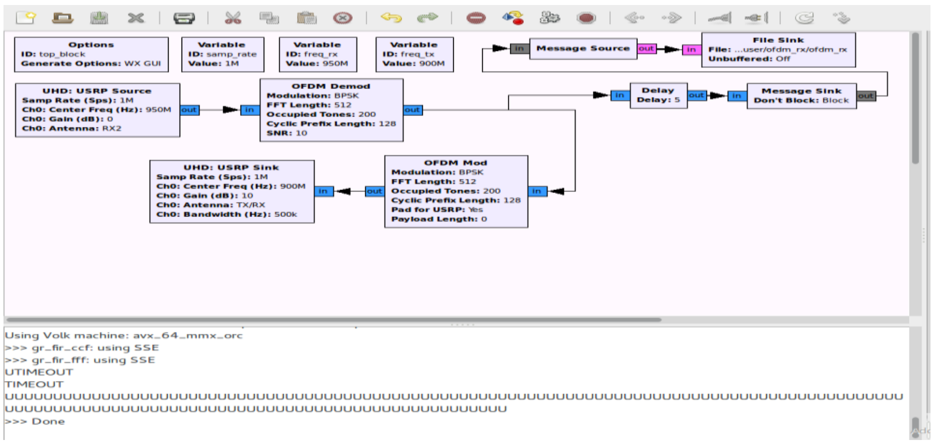
```
HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN
HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN
HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN
HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN
HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN
HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN
HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN
HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN
HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN
HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN
HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN
HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN
HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN
HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBPKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 2OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER
20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER
20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER
20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER
20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER
20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER
20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER
20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER
20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER
20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER
```

Figure 2-32

Radio 2:

Transmission frequency: 900 MHz

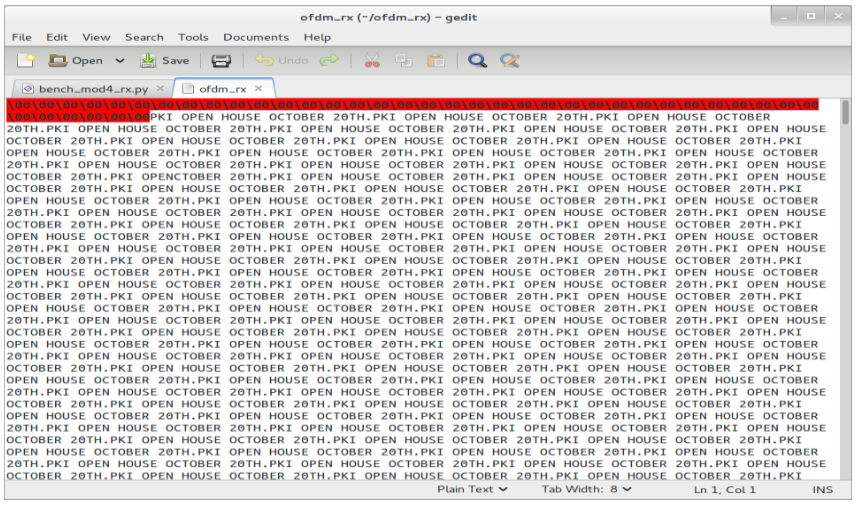
Receiving frequency: 950MHz



```
Using Volk machine: avx_64_mmx_orc
>>> gr_fir_ccf: using SSE
>>> gr_fir_fff: using SSE
UTIMEOUT
TIMEOUT
UUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUU
UUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUU
>>> Done
```

Figure 2-33

Output:



```
\00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\00
\00\00\00\00\00\00PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER
20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI
OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER
20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPENCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI
OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER
20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI
OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER
20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI
OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER
20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI
OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER
20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI
OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER
20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI
OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER
20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI
OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER
20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI
OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER
20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI
```

Figure 2-34

2.3.3 Three Radios in Multi-Hop (Circular) Transmission

In this part, we are going to use the same design for the simple loop transmission between 2 radios. The first radio was set on the frequency of 950 MHz to transmit and 900 MHz to receive. The second radio receives at 950 MHz but retransmits the same information at 830 MHz and the third radio receives at the frequency of 830 MHz but retransmits the same data to the first radio on the frequency of 900MHz. The design was tested and it worked successfully.

Setting:

Three radios set up with one radio

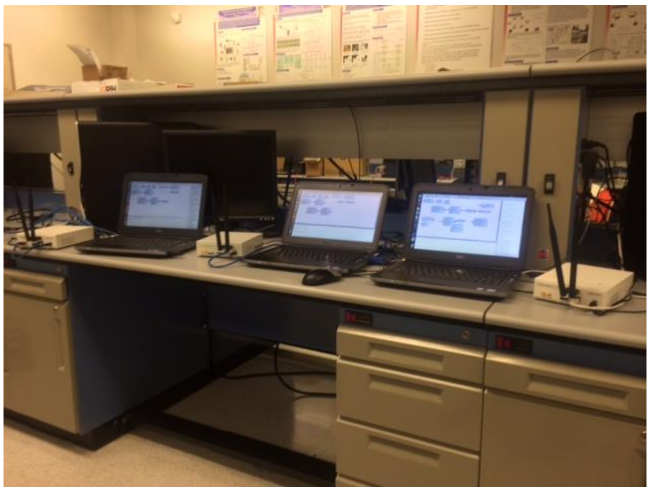


Figure 2-35

Radio 1:

Transmission frequency: 950MHz

Receiving frequency: 900MHz

Radio 2:

Transmission frequency: 830MHz

Receiving frequency: 950MHz

Radio3:

Transmission frequency: 900 MHz

Receiving frequency: 830MHz

Radio 3: Design Template and output:

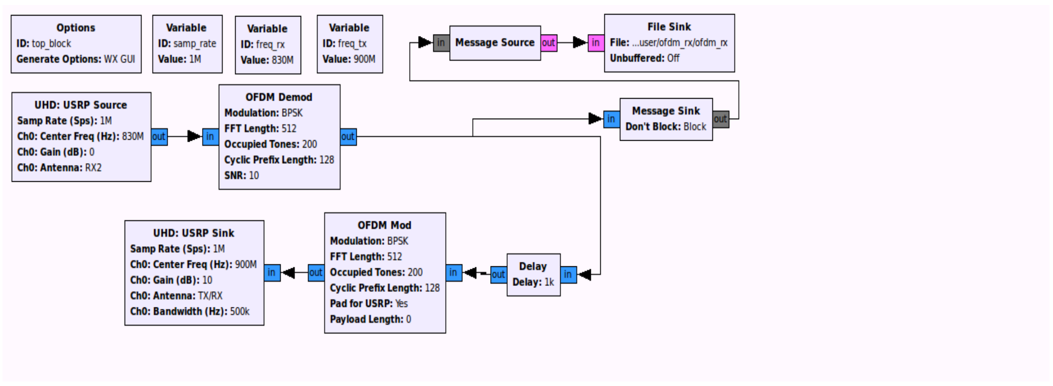
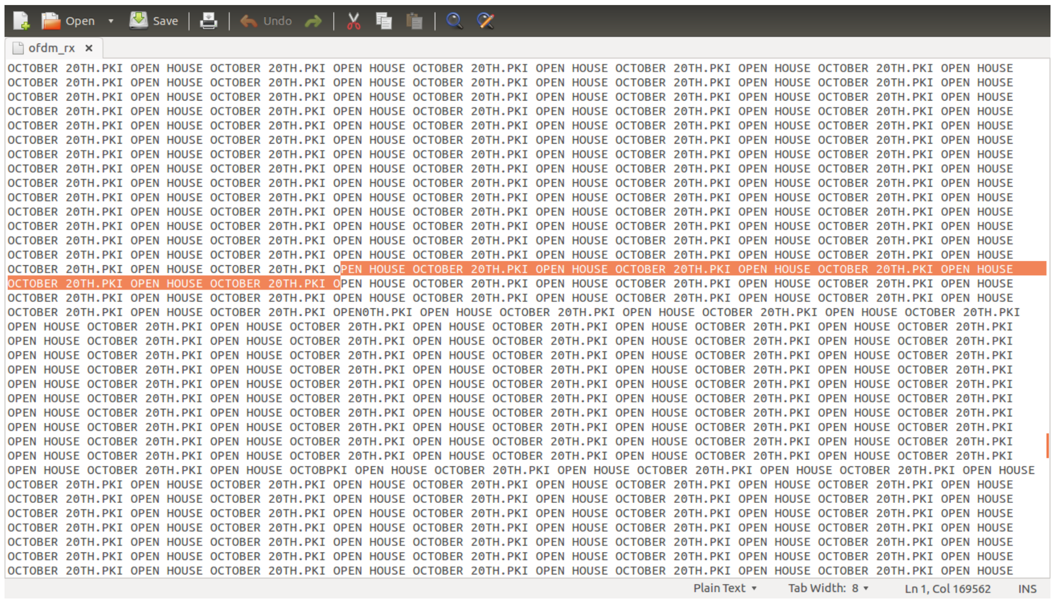


Figure 2-36

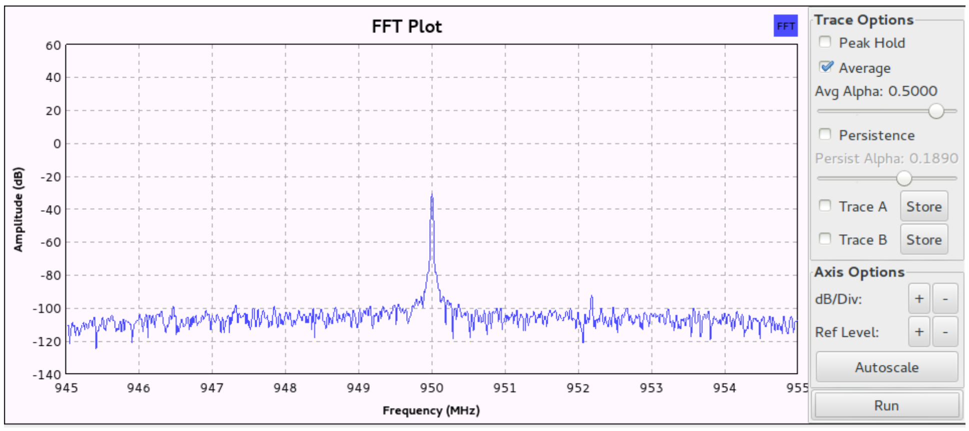
Output:



```
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN0TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI
OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI
OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI
OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI
OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI
OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI
OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI
OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI
OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI
OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI
OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI
OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBPKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE OCTOBER 20TH.PKI OPEN HOUSE
```

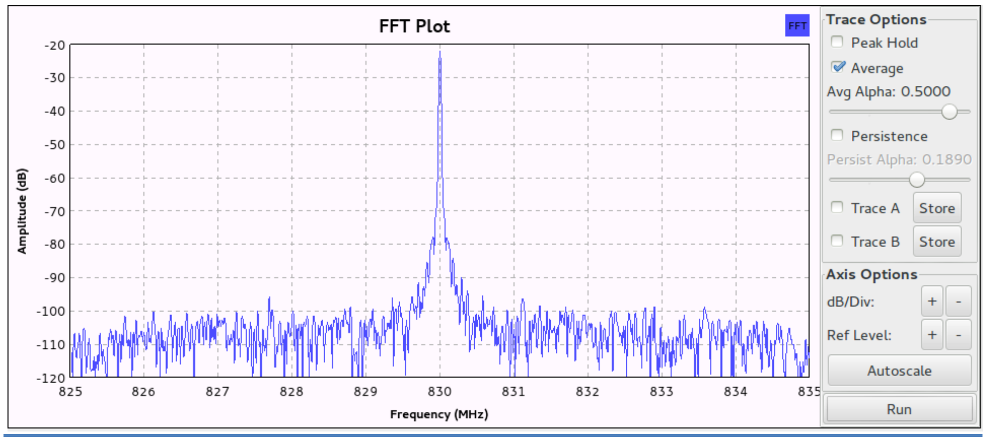
Figure 2-37

FFT plot of Transceiver block design with 10MHz bandwidth at 950MHz center freq is shown in Fig. 2-38. The FFT plot of Transceiver block design with 10MHz bandwidth at 830MHz center freq is shown in Fig. 2-39.



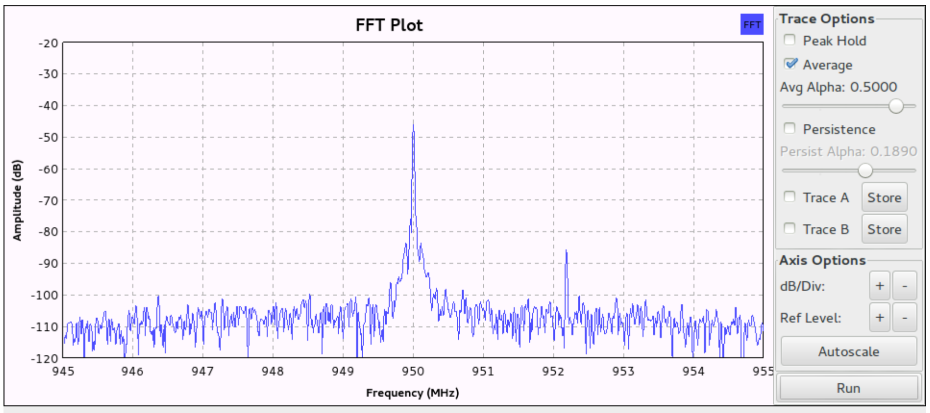
Trace Options
Peak Hold
✓ Average
Avg Alpha: 0.5000
Persistence
Persist Alpha: 0.1890
Trace A Store
Trace B Store
Axis Options
dB/Div: + -
Ref Level: + -
Autoscale
Run

Figure 2-3



Trace Options
Peak Hold
✓ Average
Avg Alpha: 0.5000
Persistence
Persist Alpha: 0.1890
Trace A Store
Trace B Store
Axis Options
dB/Div: + -
Ref Level: + -
Autoscale
Run

Figure 2-39



Trace Options
Peak Hold
✓ Average
Avg Alpha: 0.5000
Persistence
Persist Alpha: 0.1890
Trace A Store
Trace B Store
Axis Options
dB/Div: + -
Ref Level: + -
Autoscale
Run

Figure 2-40

CHAPTER 3 SENSING

3.1 Graphic targeted experiments

3.1.1 Sensing without Primary Users

In this part, we are going to use programs provided with the gnuradio package such as uhd_fft.grc to observe spectrum activity. Specifically, we redesigned a block (RX transmission) in gnuradio companion to output amplitude and frequency (uhd_fft.grc). The block is attached below with the graph. On the graph we could see the average noise amplitude was around -120dB and close to 900M it was little bit over -100dB.

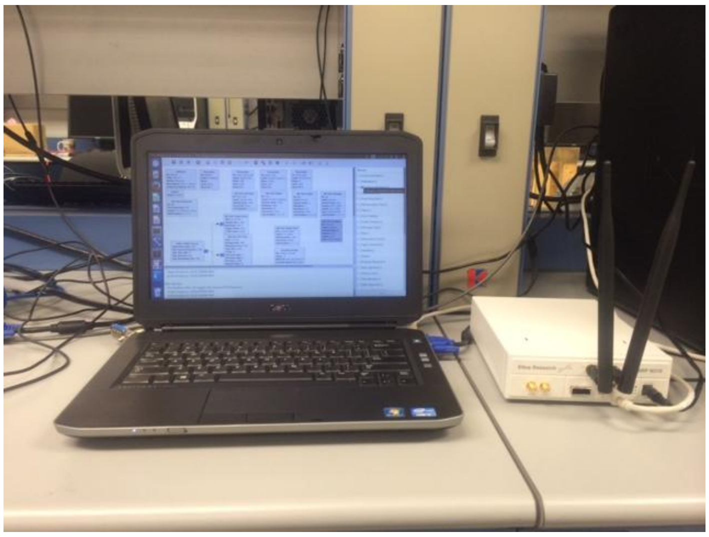
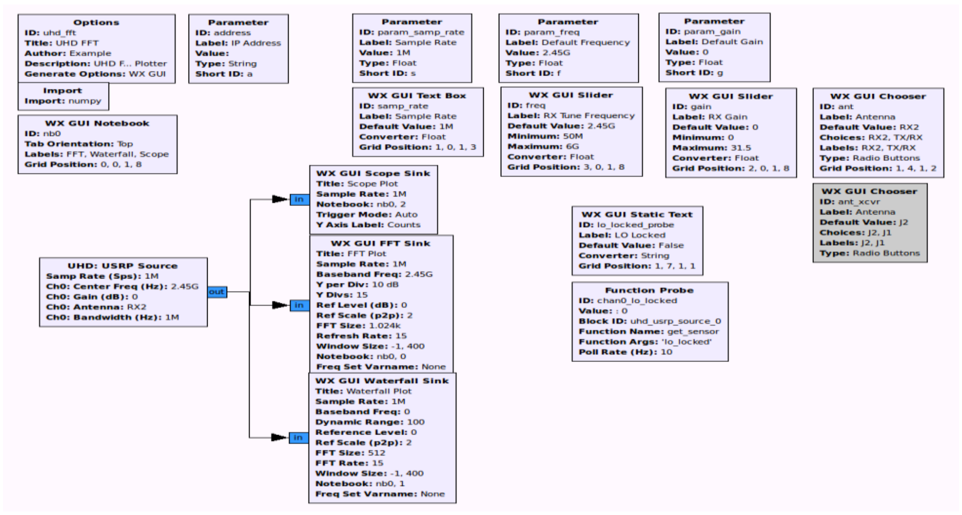


Figure 3-1

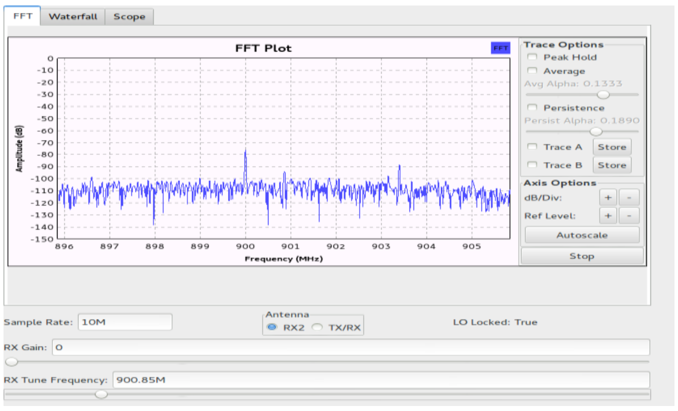
Uhd_fft.grc block



```
Options
ID: uhd_fft
Title: UHD FFT
Author: Example
Description: UHD F... Plotter
Generate Options: WX GUI

Import
Import: numpy

WX GUI Notebook
ID: nb0
Tab Orientation: Top
Labels: FFT, Waterfall, Scope
Grid Position: 0, 0, 1, 8

Parameter
ID: address
Label: IP Address
Value:
Type: String
Short ID: a

Parameter
ID: param_samp_rate
Label: Sample Rate
Value: 1M
Type: Float
Short ID: s

WX GUI Text Box
ID: samp_rate
Label: Sample Rate
Default Value: 1M
Converter: Float
Grid Position: 1, 0, 1, 3

Parameter
ID: param_freq
Label: Default Frequency
Value: 2.45G
Type: Float
Short ID: f

WX GUI Slider
ID: freq
Label: RX Tune Frequency
Default Value: 2.45G
Minimum: 50M
Maximum: 6G
Converter: Float
Grid Position: 3, 0, 1, 8

Parameter
ID: param_gain
Label: Default Gain
Value: 0
Type: Float
Short ID: g

WX GUI Slider
ID: gain
Label: RX Gain
Default Value: 0
Minimum: 0
Maximum: 31.5
Converter: Float
Grid Position: 2, 0, 1, 8

WX GUI Chooser
ID: ant
Label: Antenna
Default Value: RX2
Choices: RX2, TX/RX
Labels: RX2, TX/RX
Type: Radio Buttons
Grid Position: 1, 4, 1, 2

WX GUI Chooser
ID: ant_xcvr
Label: Antenna
Default Value: J2
Choices: J2, J1
Labels: J2, J1
Type: Radio Buttons

UHD: USRP Source
Samp Rate (Sps): 1M
Ch0: Center Freq (Hz): 2.45G
Ch0: Gain (dB): 0
Ch0: Antenna: RX2
Ch0: Bandwidth (Hz): 1M

WX GUI Scope Sink
Title: Scope Plot
Sample Rate: 1M
Notebook: nb0, 2
Trigger Mode: Auto
Y Axis Label: Counts

WX GUI FFT Sink
Title: FFT Plot
Sample Rate: 1M
Baseband Freq: 2.45G
Y per Div: 10 dB
Y Divs: 15
Ref Level (dB): 0
Ref Scale (p2p): 2
FFT Size: 1.024k
Refresh Rate: 15
Window Size: -1, 400
Notebook: nb0, 0
Freq Set Varname: None

WX GUI Waterfall Sink
Title: Waterfall Plot
Sample Rate: 1M
Baseband Freq: 0
Dynamic Range: 100
Reference Level: 0
Ref Scale (p2p): 2
FFT Size: 512
FFT Rate: 15
Window Size: -1, 400
Notebook: nb0, 1
Freq Set Varname: None

WX GUI Static Text
ID: lo_locked_probe
Label: LO Locked
Default Value: False
Converter: String
Grid Position: 1, 7, 1, 1

Function Probe
ID: chan0_lo_locked
Value: : 0
Block ID: uhd_usrp_source_0
Function Name: get_sensor
Function Args: 'lo_locked'
Poll Rate (Hz): 10
```

Figure 3-2

Output at 900MHz with 10MHz bandwidth



Sample Rate: 10M

Antenna: [✓ RX2] TX/RX

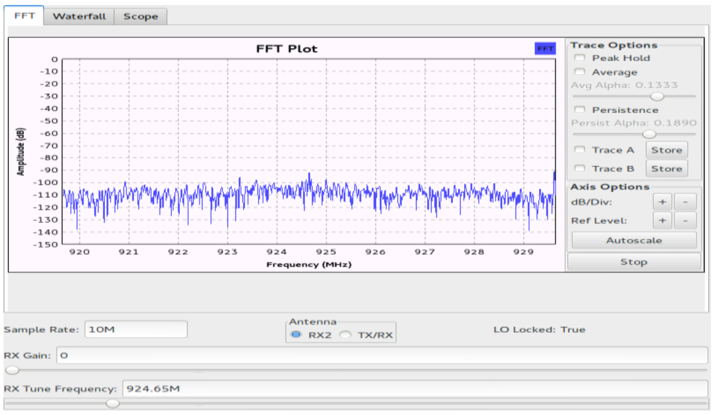
LO Locked: True

RX Gain: 0

RX Tune Frequency: 900.85M

Figure 3-3

Output at 925MHz with 10MHz bandwidth



Sample Rate: 10M

Antenna: [✓ RX2] TX/RX

LO Locked: True

RX Gain: 0

RX Tune Frequency: 924.65M

Figure 3-4

3.1.2 Sensing with Primary Users

In this part, we are going to include another radio to transmit at different frequencies to observe changes in noise floor power received. We continue to work on the frequency sensing milestone but this time we used a transmission block design to send a signal at 900M. (Pictures below, since we could not take a screen shot with the cursor on the maximum amplitude. The picture is when no signal was sent to the uhd_fft design on the radio 1. The 2nd and third pictures are when the transmission signal is run on the second radio at 900M. The amplitude when no signal is available was at -93.75 dBm, and it was -67 dBm when a signal was detected with -34 dBm as pick amplitude.

The output (attached) below shows that our noise_floor_db average was -102 dBm around 900M the center frequency at which the transmission radio is sending signal. At 899 M the average noise_floor was -112 dBm and at 901M the noise_floor was at -114 dBm.

Setting:

Set two radios with GRC companion, one to transmit at 900M the other to run the uhd_fft.grc to sense

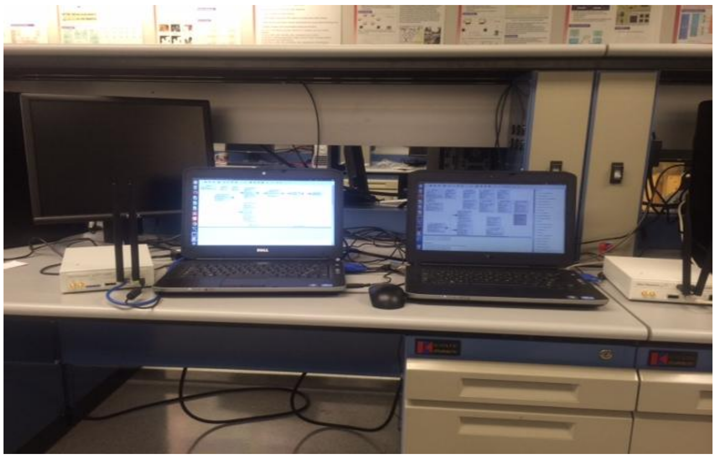
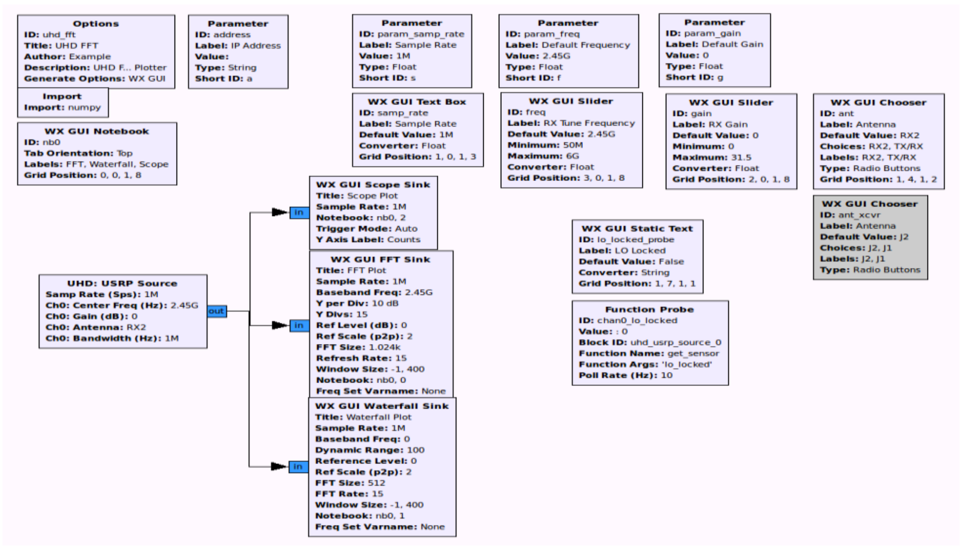


Figure 3-6

Uhd_fft block



Options
ID: uhd_fft
Title: UHD FFT
Author: Example
Description: UHD F... Plotter
Generate Options: WX GUI

Import
Import: numpy

WX GUI Notebook
ID: nb0
Tab Orientation: Top
Labels: FFT, Waterfall, Scope
Grid Position: 0, 0, 1, 8

Parameter
ID: address
Label: IP Address
Value:
Type: String
Short ID: a

Parameter
ID: param_samp_rate
Label: Sample Rate
Value: 1M
Type: Float
Short ID: s

Parameter
ID: param_freq
Label: Default Frequency
Value: 2.45G
Type: Float
Short ID: f

Parameter
ID: param_gain
Label: Default Gain
Value: 0
Type: Float
Short ID: g

WX GUI Text Box
ID: samp_rate
Label: Sample Rate
Default Value: 1M
Converter: Float
Grid Position: 1, 0, 1, 3

WX GUI Slider
ID: freq
Label: RX Tune Frequency
Default Value: 2.45G
Minimum: 50M
Maximum: 6G
Converter: Float
Grid Position: 3, 0, 1, 8

WX GUI Slider
ID: gain
Label: RX Gain
Default Value: 0
Minimum: 0
Maximum: 31.5
Converter: Float
Grid Position: 2, 0, 1, 8

WX GUI Chooser
ID: ant
Label: Antenna
Default Value: RX2
Choices: RX2, TX/RX
Labels: RX2, TX/RX
Type: Radio Buttons
Grid Position: 1, 4, 1, 2

WX GUI Chooser
ID: ant_xcvr
Label: Antenna
Default Value: J2
Choices: J2, J1
Labels: J2, J1
Type: Radio Buttons

WX GUI Static Text
ID: lo_locked_probe
Label: LO Locked
Default Value: False
Converter: String
Grid Position: 1, 7, 1, 1

Function Probe
ID: chan0_lo_locked
Value: : 0
Block ID: uhd_usrp_source_0
Function Name: get_sensor
Function Args: 'lo_locked'
Poll Rate (Hz): 10

UHD: USRP Source
Samp Rate (Sps): 1M
Ch0: Center Freq (Hz): 2.45G
Ch0: Gain (dB): 0
Ch0: Antenna: RX2
Ch0: Bandwidth (Hz): 1M

WX GUI Scope Sink
Title: Scope Plot
Sample Rate: 1M
Notebook: nb0, 2
Trigger Mode: Auto
Y Axis Label: Counts

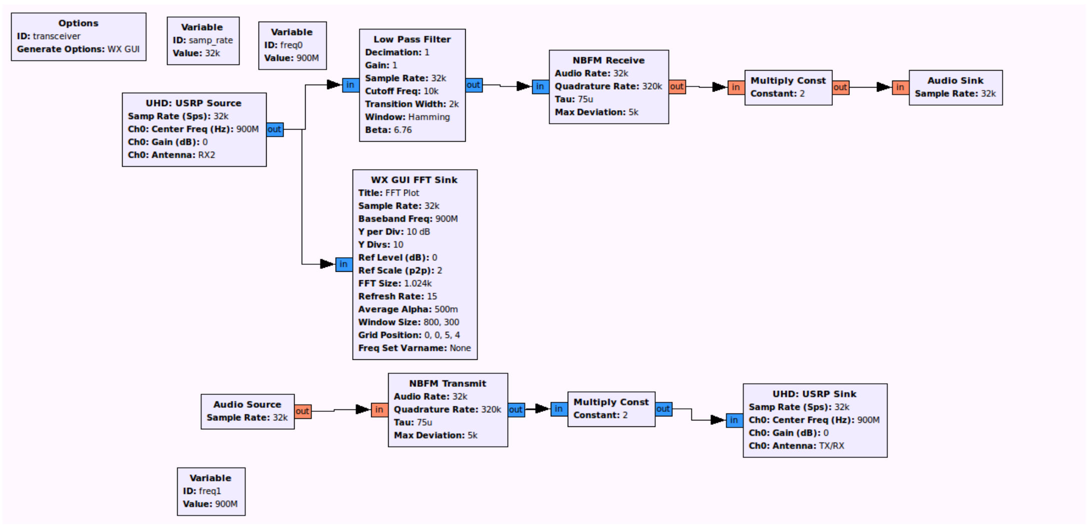
WX GUI FFT Sink
Title: FFT Plot
Sample Rate: 1M
Baseband Freq: 2.45G
Y per Div: 10 dB
Y Divs: 15
Ref Level (dB): 0
Ref Scale (p2p): 2
FFT Size: 1.024k
Refresh Rate: 15
Window Size: -1, 400
Notebook: nb0, 0
Freq Set Varname: None

WX GUI Waterfall Sink
Title: Waterfall Plot
Sample Rate: 1M
Baseband Freq: 0
Dynamic Range: 100
Reference Level: 0
Ref Scale (p2p): 2
FFT Size: 512
FFT Rate: 15
Window Size: -1, 400
Notebook: nb0, 1
Freq Set Varname: None

.

Figure 3-7

Transceiver block to transmit at 900M



Options
ID: transceiver
Generate Options: WX GUI

Variable
ID: samp_rate
Value: 32k

Variable
ID: freq0
Value: 900M

Low Pass Filter
Decimation: 1
Gain: 1
Sample Rate: 32k
Cutoff Freq: 10k
Transition Width: 2k
Window: Hamming
Beta: 6.76

NBFM Receive
Audio Rate: 32k
Quadrature Rate: 320k
Tau: 75u
Max Deviation: 5k

Multiply Const
Constant: 2

Audio Sink
Sample Rate: 32k

UHD: USRP Source
Samp Rate (Sps): 32k
Ch0: Center Freq (Hz): 900M
Ch0: Gain (dB): 0
Ch0: Antenna: RX2

WX GUI FFT Sink
Title: FFT Plot
Sample Rate: 32k
Baseband Freq: 900M
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 0
Ref Scale (p2p): 2
FFT Size: 1.024k
Refresh Rate: 15
Average Alpha: 500m
Window Size: 800, 300
Grid Position: 0, 0, 5, 4
Freq Set Varname: None

Audio Source
Sample Rate: 32k

NBFM Transmit
Audio Rate: 32k
Quadrature Rate: 320k
Tau: 75u
Max Deviation: 5k

Multiply Const
Constant: 2

UHD: USRP Sink
Samp Rate (Sps): 32k
Ch0: Center Freq (Hz): 900M
Ch0: Gain (dB): 0
Ch0: Antenna: TX/RX

Variable
ID: freq1
Value: 900M

Figure 3-8

Output:

Before transmission

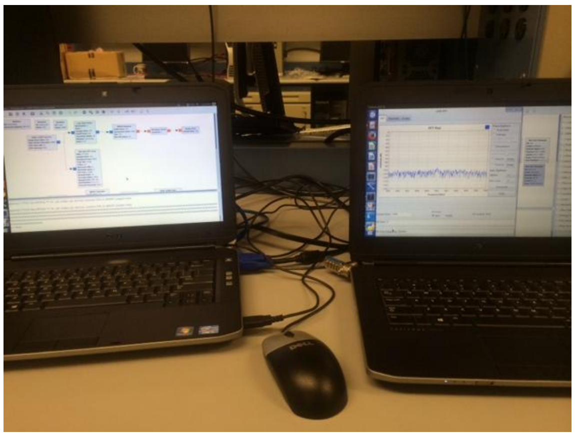


Figure 3-9

After

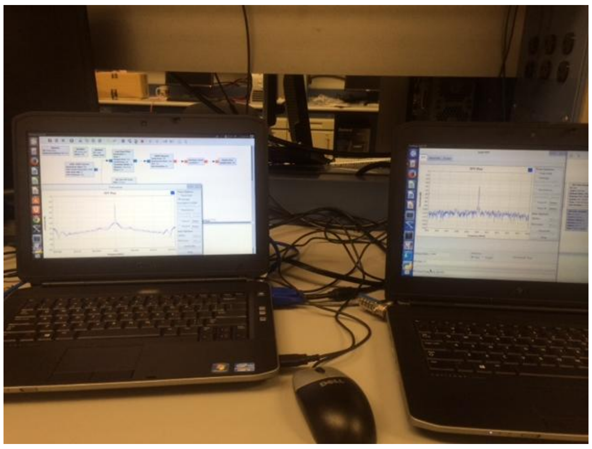
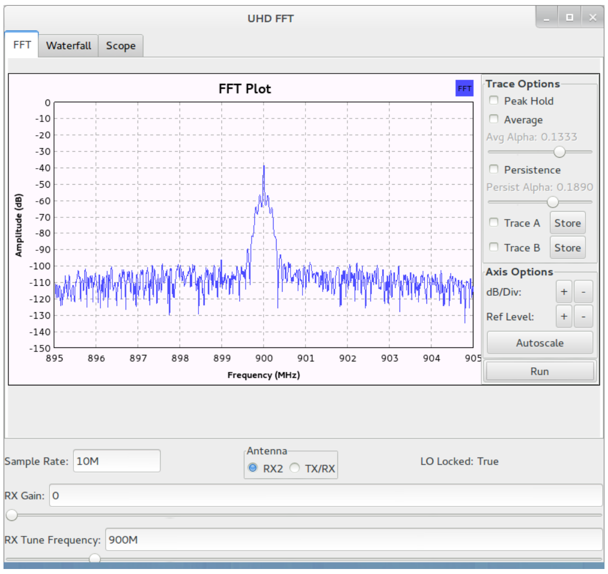


Figure 3-10

A closer look is shown in Fig. 3-11



Avg Alpha: 0.1333

Persist Alpha: 0.1890

Sample Rate: 10M

Antenna [✓ RX2] TX/RX

LO Locked: True

RX Gain: 0

RX Tune Frequency: 900M

Figure 3-11

3.2 Intensive Measurements

From the previous experiments we see that with or without primary user cause different results of sensing. However, we need to quantify those sensing values and finally come up with a threshold value so that we can determine if the primary user is active in real time. Instead of using a UI interface to show the results, we apply python code for faster and automated executions. The measurements are taken 1000 times for each frequency in two cases, 1) without primary user activity, 2) with primary user activity. The frequencies measured are from 830 MHz-960 MHz in 5 MHz step, and 1720 MHz-1980 MHz in 10 MHz step.

One example of measuring f=900 MHz is shown in Fig. 3-12. As we can see, there is a gap for us to choose the threshold.

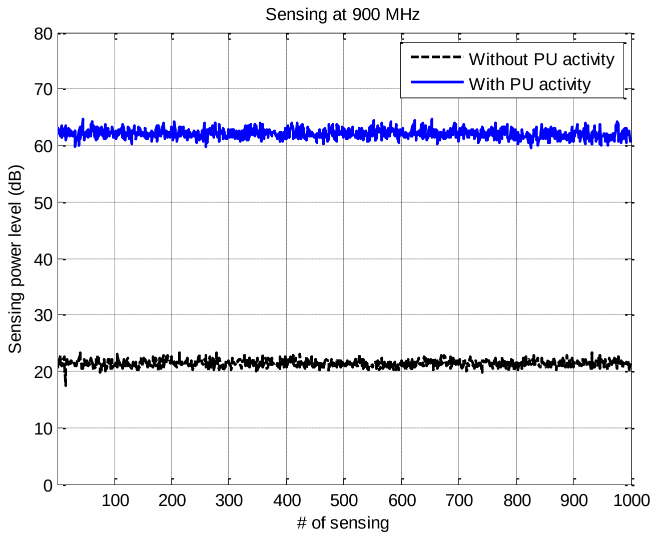


Figure 3-12

In Fig. 3-13 and Fig. 3-14 we show the average value of the 1000 measurements of each frequency for the whole testing frequency range.

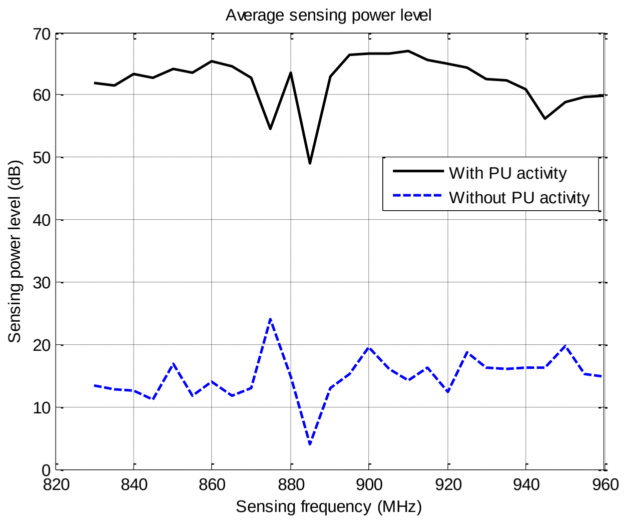


Figure 3-13

It is shown in Fig. 3-14 that at f=1960 MHz there is a steep down, we ran the same testing for another 4000 times and its maximum value was still below 10 dB (above the noise level). We will do more evaluations about this frequency however we conclude it as a hardware issue by now.

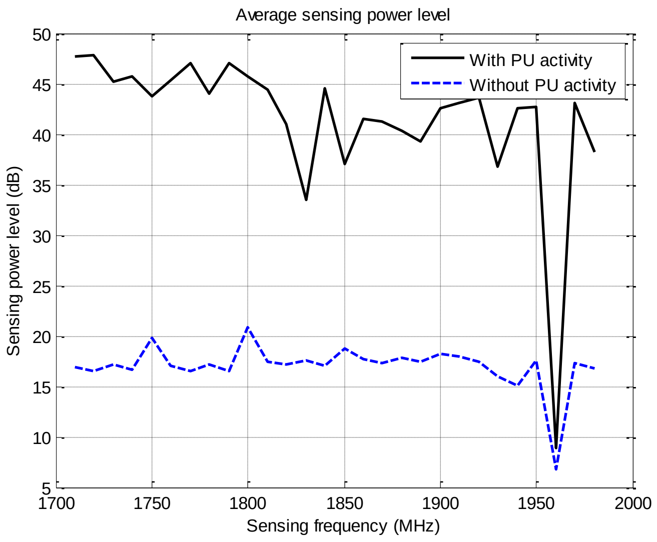


Figure 3-14

CHAPTER 4 MWCRN APPLICATIONS

4.1 TBD

Under Construction

CHAPTER 5 CONCLUSION

Under construction

References

[1] J. Mitola. III, “Cognitive Radio for Flexible Mobile Multimedia communications,” IEEE 1999 Mobile Multimedia Conference (MoMuC), pp. 3-10, Nov. 1999.

[2] FCC, ET Docket No. 03-222 Notice of proposed rule making and order, Dec. 2003.