

HackRF One

Disclaimer:

This document contains information based on research that has been gathered by employee(s) of The Senator Patrick Leahy Center for Digital Investigation (LCDI). The data contained in this project is submitted voluntarily and is unaudited. Every effort has been made by LCDI to assure the accuracy and reliability of the data contained in this report. However, LCDI nor any of our employees make no representation, warranty or guarantee in connection with this report and hereby expressly disclaims any liability or responsibility for loss or damage resulting from use of this data. Information in this report can be downloaded and redistributed by any person or persons. Any redistribution must maintain the LCDI logo and any references from this report must be properly annotated.

12/01/2017

<http://computerforensicsblog.champlain.edu/>

175 Lakeside Ave, Room 300A

Phone: (802) 865-5744

Fax: (802) 865-6446 12/01/2017

Contents

Introduction	3
Background	3
Purpose and Scope	3
Research Questions	3
Terminology	3
Getting Started	5
Radio Basics	5
Hardware Introduction	6
Software Introduction	9
HackRF One Basics	9
Setting up the HackRF One Device	9
Setting up the work environment	9
Installing GNU Radio Companion	10
Receiving FM Radio Signals	12
Customization of Receiving FM Radio Signals	16
Transmitting FM Radio Signals	18
Retransmitting FM Radio Signals	20
Live Broadcasting FM Radio Signals	21
Further Work	22
Appendix	23
Commonly Used Functions on GNU Radio Companion	23
FCC ID	26
References	33

Introduction

This document will serve as a tutorial for the use of the HackRF One (HRF1) device, which is used to receive and transmit signals between the frequencies of 1 MHz and 6 GHz (Ossmann, "HackRF One," n.d.). The basic functions of the HackRF One device and the GNU Radio Companion software will be explained throughout this tutorial.

Background

The HackRF One is a Software Defined Radio (SDR) device with the ability to digitize radio signals that are received or transmitted by the device. The HRF1 works with frequencies from 1MHz to 6 GHz, which includes most devices operating with Bluetooth, FM radio, near-field communication (NFC), and cellular technology. The device is operated in conjunction with a computer and software that can process SDR, such as GNU Radio Companion (GRC) ("HackRF One," n.d.).

The HRF1 device operates in half duplex mode, meaning that it is only able to either receive transmissions or transmit signals one at a time, rather than receiving and transmitting at the same time (Ossmann, "HackRF One," n.d.).

Purpose and Scope

The purpose of this tutorial is to provide a basic understanding of the HackRF One device and the GNU Radio Companion software through written instructions, pictures, and video tutorials in order to demonstrate the process of receiving, broadcasting, retransmitting, and live transmitting signals.

Research Questions

1. What is the HackRF One?
2. How can the HackRF One be used?

Terminology

Amplitude - The strength of the radio signal ("Amplitude Modulation," 2017).

Digital Signal Processing (DSP) - Signals processed via analysis, modification, and synthesis by a sequence of numbers that represent samples of a continuous variable in a domain such as time, space, or frequency (Rouse, "DSP," n.d.).

GNU Radio Companion - A graphic user interface (GUI) software that allows users to create a flowgraph that visualizes and processes signals ("GNU Radio Companion," n.d.).

Filter - Helps clean up received signals in order to limit unnecessary noise and interference, and to clean up signals when transmitting to cause less radio interference ("Low-pass Filter," 2017).

Hertz (Hz) - Derived unit of frequency. Defined as one cycle per second.

Radio Frequency - Number of cycles per second of radio waves; measured in hertz (Hz) ("Radio Frequency," n.d.).

Receive - The act of obtaining and interpreting radio waves from other devices ("Radio Receiver," 2017).

Functions in the form of software controlled modules while using SDR (Rouse, "SDR," n.d.).

Sample Rate - Rate at which data is taken digitally over a period of time. Measured in Hz. (Christensson, 2015).

Software Defined Radio (SDR) - Used to transmit radio waves via hardware. SDR emulates radio hardware

Transmit - The act of sending out a signal via a wire or radio waves ("Transmit," 2014).

Getting Started

Radio Basics

1. **Radio Waves** - Part of a larger group of waves called *electromagnetic radiation*. Radio waves can be found on the Electromagnetic Spectrum with X-rays, Infrared, Microwaves, etc.
2. **Data on Radio Waves** - There are two major ways that audio data is encoded onto a radio wave, A.M. and F.M. These two types usually have a consistent pattern throughout the entire wavelength. (Northwestern University, "How is data put on radio waves?") (Northwestern University, "What are radio waves?," n.d.).
 - a. A.M. – Amplitude Modulation
 - i. Information is put in a radio wave by changing the height of the wave.
 - b. F.M. – Frequency Modulation
 - i. Information is put in a radio wave by changing how close the waves are to each other.

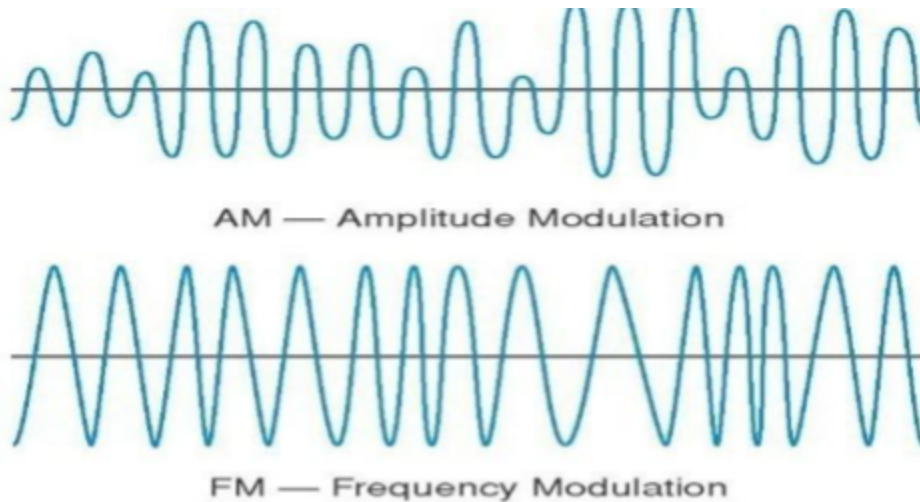


Figure 1: (AM/FM Modulation)

Hardware Introduction

1. **Computer** – Needed to run software to analyze what was captured.
2. **Radio device** – Any device that is capable of receiving transmissions from the HackRF One.
3. **HackRF One** – The device used to receive and transmit signals. The packaging includes:
 - a. The HackRF One device.
 - b. Micro- USB to USB cable used to connect the HackRF One device to a computer.
 - c. ANT500 antenna. Note that using the HRF1 without an antenna or a dummy plug could cause damage to the device (Ossmann, 2014). Additionally, the ANT500 only supports frequencies from 75 MHz to 1 GHz (“ANT500," n.d.).

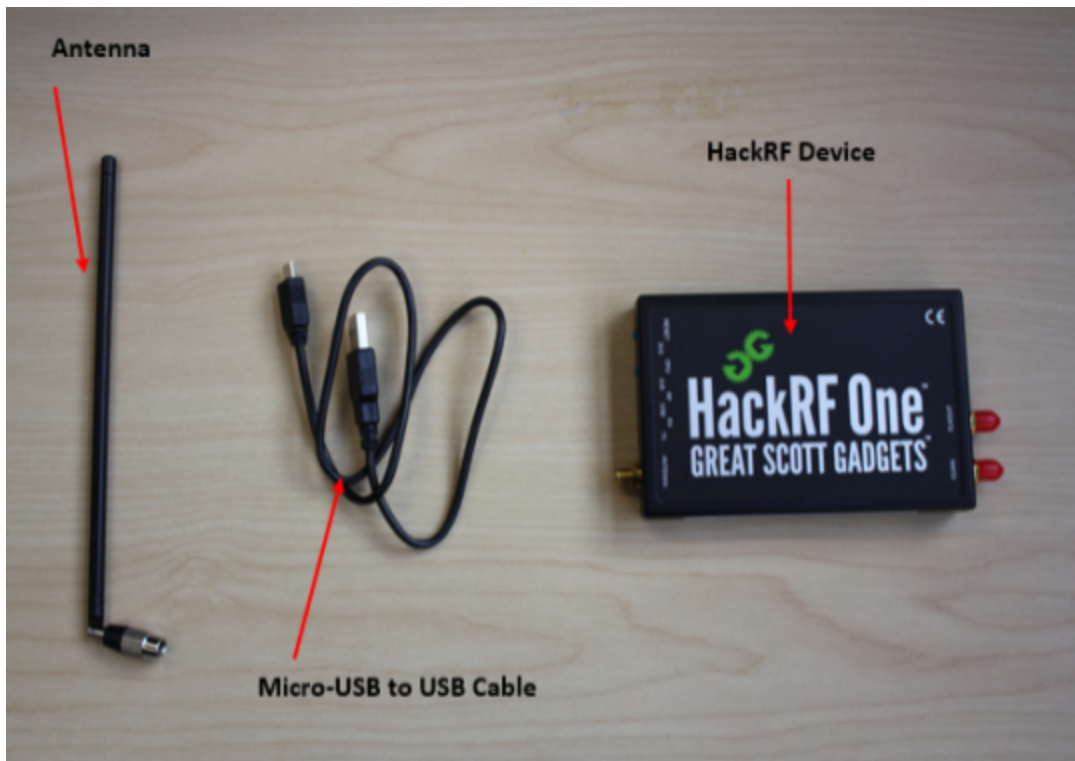


Figure 2: (Hardware and Peripherals)



Figure 3: (HackRF One Top View)



Figure 4: (HackRF One Left View)

Button/Light	Function
Reset Button	Used to reboot the HackRF One, equivalent to unplugging the device and plugging it back in ("HackRF One One," n.d.).
3v3 LED	All three of these LEDs are used to indicate power and should be lit when the HackRF One is plugged in. The various colors are used to distinguish between the multiple LEDs on the side of the HackRF One ("FAQ," n.d.).
1V8 LED	
RF LED	
USB LED	Indicates that the HackRF One is communicating over USB ("FAQ," n.d.).
DFU Button	Used to install or update the firmware if it is not working properly or has never been installed ("HackRF One," n.d.).
RX LED	An orange light that indicates that the device is receiving information ("FAQ," n.d.).
TX LED	A red light that indicates that the device is transmitting information ("FAQ," n.d.).

Table 1 – HackRF One LED Lights

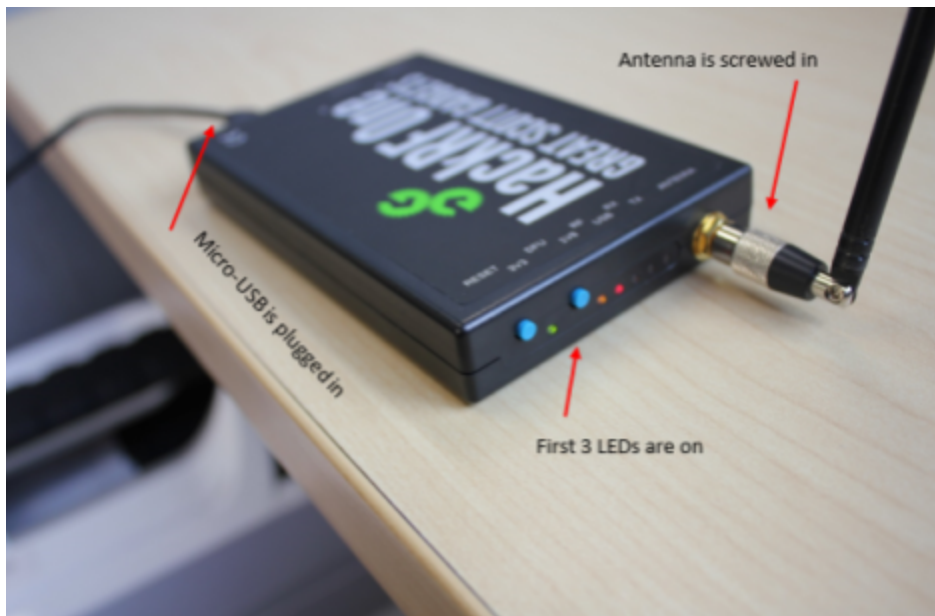


Figure 5: (HackRF One Ready)

Software Introduction

The HackRF One is primarily operated using the **Ubuntu** Operating System, which the **GNU Radio Companion** SDR system uses as a means to capture, analyze, and transmit radio frequencies generated from another hardware device.

HackRF One Basics

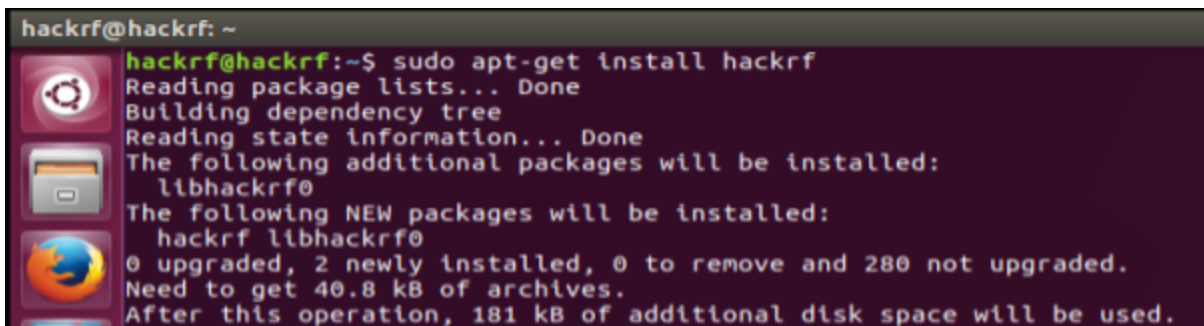
Setting up the HackRF One Device

1. Attach the antenna to the antenna port after removing the red safety cap. **Attaching the antenna first is vital because running the HackRF One without an antenna can damage the device’s hardware.**
2. Ubuntu Linux is recommended as the operating system alongside GNU Radio. Ubuntu can be downloaded from this link: <https://www.ubuntu.com/download>
3. Plug the HackRF One into the computer with the Micro-USB to USB cable. Confirm that the first three LED lights are illuminated to ensure the device is working.

Video 1 – [Device Overview](#):

Setting up the work environment

1. Before getting started, the HackRF One libraries must be installed on the computer to start using the device.
 - a. The libraries can be installed with the Terminal command “*sudo apt-get install hackrf*”



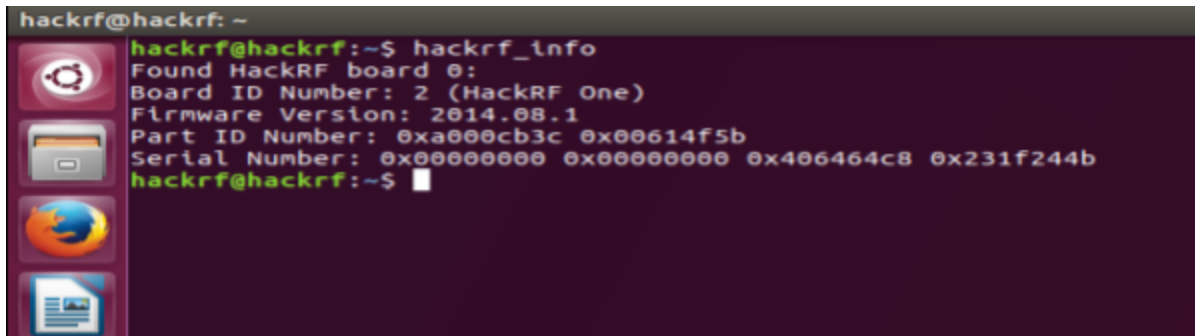
```

hackrf@hackrf: ~
hackrf@hackrf:~$ sudo apt-get install hackrf
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
 libhackrf0
The following NEW packages will be installed:
 hackrf libhackrf0
0 upgraded, 2 newly installed, 0 to remove and 280 not upgraded.
Need to get 40.8 kB of archives.
After this operation, 181 kB of additional disk space will be used.
    
```

Figure 6: (Install HackRF One)

2. After installation, type the command “*hackrf_info*” to verify that the HRF1 device is connected.
 - a. After a successful installation, the terminal should respond with “*Found HackRF board*”.

- b. If the terminal shows the HRF1 as not connected, troubleshoot by verifying wires are plugged in and power is being supplied to the device.



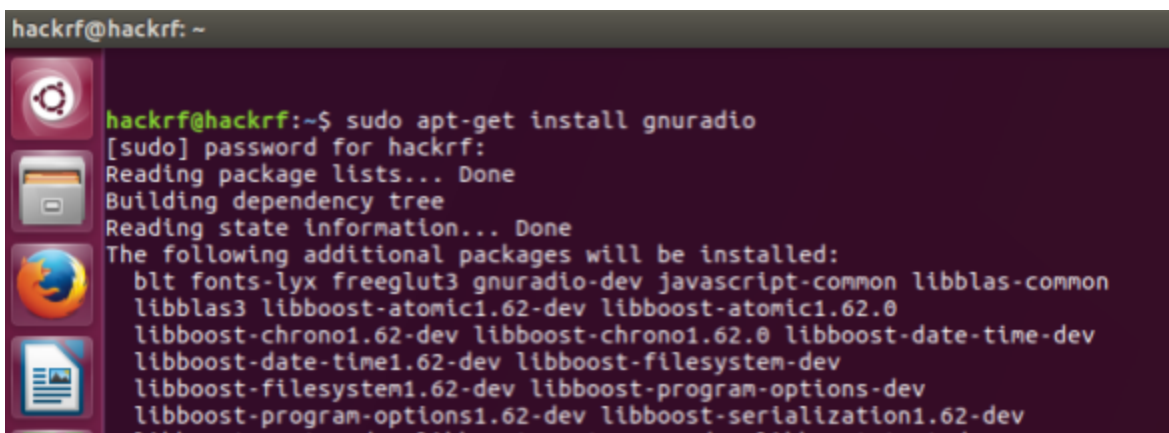
```

hackrf@hackrf: ~
hackrf@hackrf:~$ hackrf_info
Found HackRF board 0:
Board ID Number: 2 (HackRF One)
Firmware Version: 2014.08.1
Part ID Number: 0xa000cb3c 0x00614f5b
Serial Number: 0x00000000 0x00000000 0x406464c8 0x231f244b
hackrf@hackrf:~$
  
```

Figure 7: (HackRF One_Info)

Installing GNU Radio Companion

1. In order to install the GNU Radio Companion software, open a terminal window.
2. Type in the command: “*apt-get install gnuradio*”.
 - a. If the root user is not logged in, type “*sudo*” in front of this command.



```

hackrf@hackrf: ~
hackrf@hackrf:~$ sudo apt-get install gnuradio
[sudo] password for hackrf:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  blt fonts-lyx freeglut3 gnuradio-dev javascript-common libblas-common
  libblas3 libboost-atomic1.62-dev libboost-atomic1.62.0
  libboost-chrono1.62-dev libboost-chrono1.62.0 libboost-date-time-dev
  libboost-date-time1.62-dev libboost-filesystem-dev
  libboost-filesystem1.62-dev libboost-program-options-dev
  libboost-program-options1.62-dev libboost-serialization1.62-dev
  
```

Figure 8: (Install GNU Radio)

3. Now, open GNU Radio Companion using the command: “*gnuradio-companion*”.
 - a. Once initialized, GNU Radio Companion displays its user interface, which automatically opens an **untitled project**. This will show an empty white space at the center, which will be used to build a **flowgraph**. The default *options* and *variable* blocks will already be in the flowgraph (See Figure 9 below). The sidebar to the right holds other blocks to add to the flowgraph.

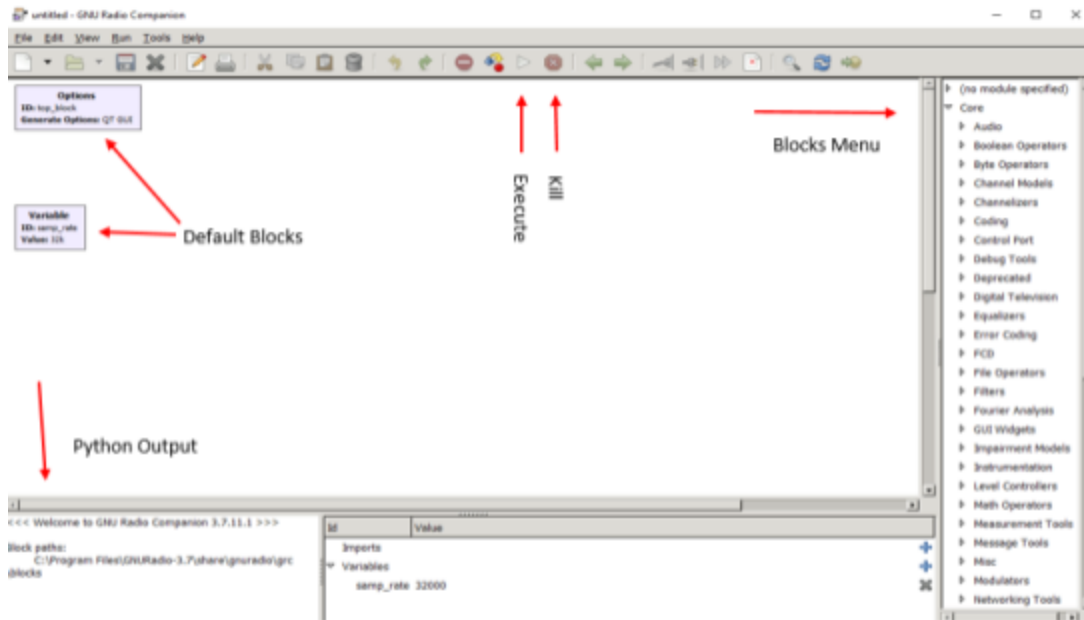


Figure 9: (Untitled Project)

- b. The play button on the top toolbar is used to execute the flowgraph. The red 'X' next to it can be used to kill the program, but it is recommended to use the red 'X' in the popup window when the program is running. Using the kill button in the top toolbar will cause issues with file sinks.
- c. The flowgraph itself is divided into 4 components:
 - i. Options (available by default upon launching GRC) - generate options should be set to QT GUI. Given the capabilities of the current technology, it has been recommended that **QT functions are used instead of WX on all functions**, as WX may not be supported in the near future. WX and QT are Python Frameworks.
 - ii. Variable (available by default upon launching GRC) - used to set the sample rate for the HRF1. Double click this block to set the sample rate value.
 1. It is recommended to use this block to change the sample rate, as it will change all of the sample rates to match when the flowgraph is more complex, instead of changing it individually in each block.
 - iii. Functions - the wide array of data blocks, which are listed in the menu to the right of the screen, are used to build the flowgraph. The function blocks encompass the most essential parts of the software because they provide different commands to accomplish

specific tasks. Selecting and dragging a function block to the flowgraph provides the ability to initialize a set of commands, and double clicking on a block inside the flowgraph opens its properties.

- iv. Arrows - connect function blocks in order to create the flowgraph.

Receiving FM Radio Signals

This section outlines how to receive FM radio signals from the HackRF One using GNU Radio Companion in order to visualize the signal in a flowgraph. This section also contains a video tutorial.

1. Upon starting the GNU Radio Companion software, there will be two boxes already present on the flowgraph area: *Variable* and *Options*. These default to a set value, which can be altered, as mentioned above. In the *Options* block, the *Generate Options* should be set to *QT GUI*.
2. In the *Variable* block, change the sample rate (*samp_rate*) to the correct value. The lowest recommended sample rate is 2e6 (2 million) Hz, and the highest sample rate used should be 20e6 (20 million) Hz. It is recommended to begin at 4e6 Hz, and decrease as needed. A computer with less processing speed should not have a high sample rate frequency.
3. To begin creating the flowgraph to receive FM radio signals, add an *osmocom source* function via (no module specified) > Source > *osmocom Source* (or search for *osmocom source* via the magnifying glass, which is found on the right side of the top menu). This enables the ability to receive data captured by the HackRF One device.
 - a. The sample rate should already be set to *samp_rate*, which was assigned in the *Variable* box. Changing the *samp_rate* in the *Variable* box will reflect throughout the rest of the flowgraph.
 - b. Change the *Ch0 Frequency (Hz)* to the radio station that is strongest in your area. For example, this should be formatted as “98.9e6”. Setting the *osmocom source* > *Ch0 frequency* tells the HackRF One device which radio channel to tune to.
 - c. It is recommended to turn off the RF amplifier by setting the *Ch0: RF Gain (dB)* value to 0. The *RF Gain* amplifies the frequency source at its default setting of 10 dB, but it is not necessary or recommended to have this setting turned on.
4. Now, add the *DC Blocker*, which filters out the direct current from the power supply to the HRF1, which can interfere with the device. If there is no *DC Blocker* present in the flowgraph, there will be a peak in the center of the flowgraph when being executed, which can be misinterpreted as a signal.

- a. Set the length value to 1024.
 - b. Connect the *osmocom Source* block to the *DC Blocker* with an arrow.
5. In order to visualize the signals being received by the HackRF One device, the frequency sink function needs to be added to the flowgraph. This function can be found via Instrumentation > QT > *QT GUI Sink* (or by searching for *QT GUI Sink*).
- a. Set the *Center Frequency (Hz)* to the frequency from *Ch0 Frequency (Hz)* in the *osmocom Source* function. For example, 98.9e6.
 - b. In order to visualize the whole range of FM radio signals, change the *bandwidth (Hz)* to 20e6. This is normally set to your sample rate by default, but changing the value will not affect your *samp_rate* in the *Variable* function. Also, the focus of the flowgraph will still be on the frequency set as the *Ch0 Frequency* while also showing the full range of radio channels in the area.
 - c. The *Update Rate* can be changed from 10 to 5. This indicates the amount of times per second the graph updates. 5 will cause less strain on your system if you do not have a high end computer.
 - d. Change *Show RF Frequency* to “yes”. This will display the frequencies on the x-axis of the frequency sink correctly.
 - e. Lastly, connect the *QT GUI Sink* (in) function to the *osmocom Source* (out) function with an arrow. The flowgraph should look like this:

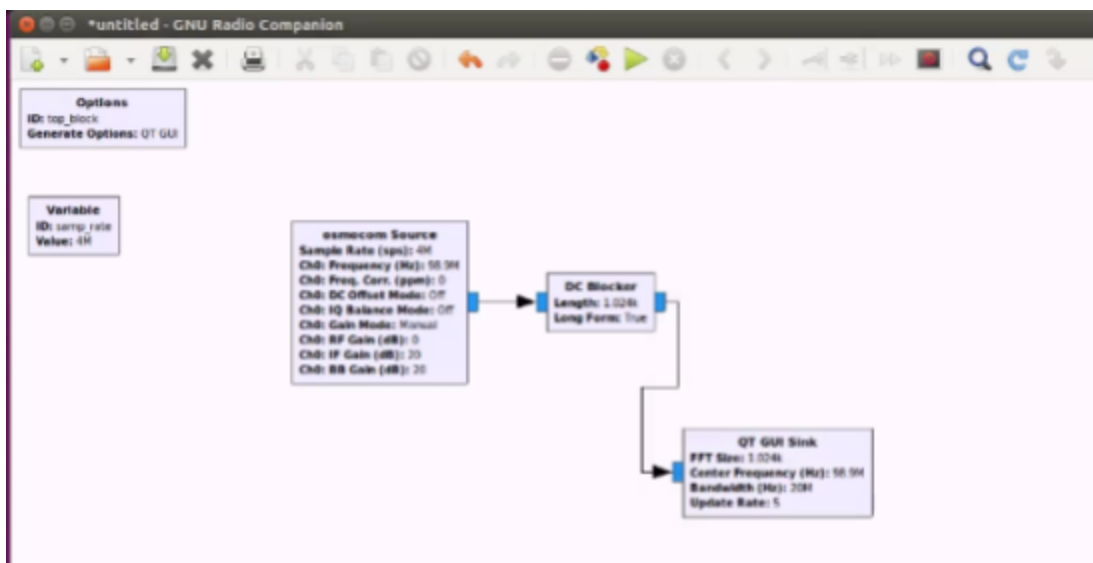


Figure 10: (Receive)

6. Press the green play button at the top in order to run the flowgraph. Upon pressing it, a prompt to save the flowgraph will appear in a new window. The file will autosave every time afterwards. After the file saves, the python script will be generated in the bottom left of the screen and the visualization will show on the main area of the screen.

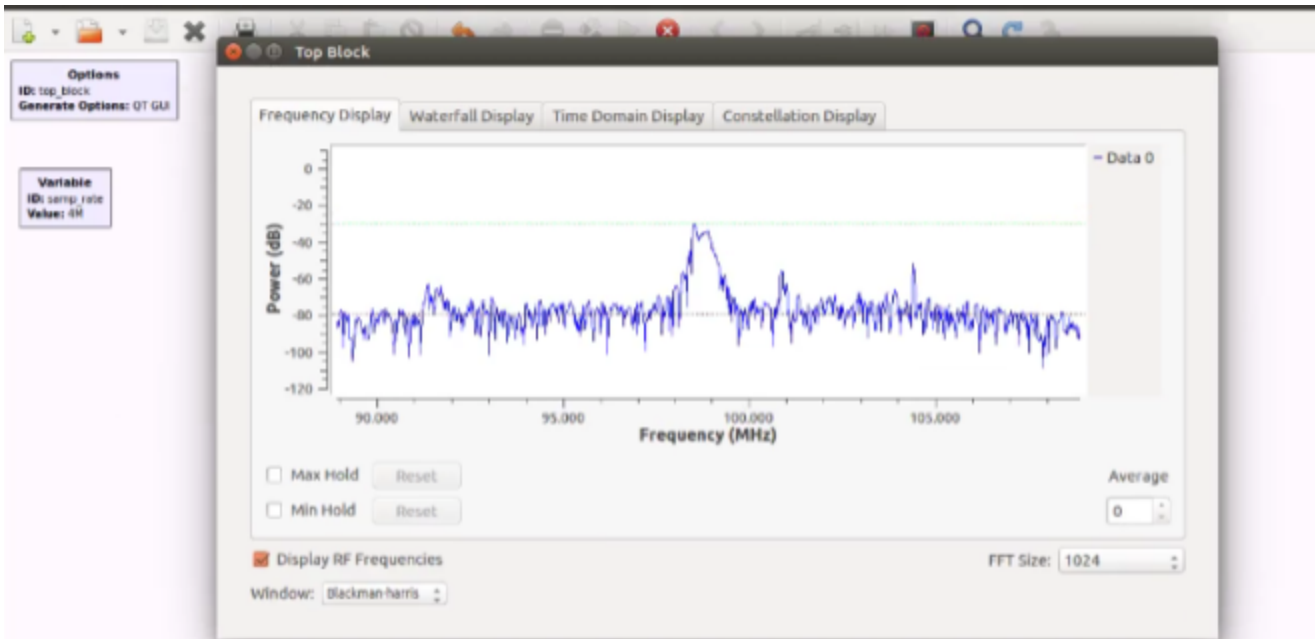


Figure 11: (Flowgraph)

- a. The first tab in the Top Block window will show the Frequency Display. There should be obvious differences between the strong signals and the rest of the frequencies. For example, channels 98.9, 91.3, and 102.9 are the strongest in the Burlington, VT area.
7. To stop the flowgraph, click the red X in the top left of the pop up window. Sometimes, the HackRF One may need to be reset (via the reset button on the device) because it will continue receiving signals even after the flowgraph is closed.
8. To listen to the signals, a few things need to be done in addition to the flowgraph that has already been built. The *Low Pass Filter* function (found via searching for *Low Pass Filter*) will focus on the center frequency and reduce noise from the surrounding stations.
 - a. The *Decimation* value should be set to 20, which will divide the sample rate by 20. This will get the sample rate to the same rate as the audio.

- b. The *Cutoff Frequency* should be $75e3$. This is how wide the signal is.
 - c. The *Transition Width* should be $25e3$. This is how sharply the unneeded signals are filtered out, and applies a more focused center frequency. The lower the transition width, the sharper the transition width. $25e3$ is a good width value for FM radio.
 - d. Connect the *Low Pass Filter* function to the *osmocom Source* function.
9. A *Rational Resampler* should now be added as a way to modify the sample rate by a ratio, as GNU radio companion will only allow the sample rate to be changed by integers. By setting this rate as 12 to 5, this will help move the sample rate closer to the targeted audio frequency. If the sample rate is faster than the actual audio rate, then it will affect the speed of the audio and distort the sound, so this can fix that problem.
- a. Connect this function to the *Low Pass Filter* block.
10. In addition, the Wide Band Frequency function is needed. This is found by searching for *WBFM Receive*. This block will have one blue input (like the rest of the inputs/outputs on the flowgraph) and one orange output. Blue outputs are complex numbers so there is a real and imaginary number being received. The *WBFM Receive* function is combining both complex numbers and creating one output via the orange real number output.
- a. Set the *Quadrature Rate* to $480e3$, which is recommended for the FM radio audio receive task.
 - b. Set the *Audio Decimation* to 10, which will reduce the audio rate to 48 KHz since you're dividing it by 10. This is the rate at which the audio will be heard.
 - c. Connect this function to the *Rational Resampler*.
11. The last function needed is the *Audio Sink* (found via searching for *Audio Sink*). This will allow you to actually hear the audio from the radio station set as the center frequency. This block has an orange input, which will be connected to the orange *WBFM Receive* output.
- a. Change the sample rate to 48 KHz from the drop down menu.
12. Now, execute the flowgraph again by clicking the green play button in the menu. The audio should immediately begin playing and the radio signals for the center frequency should be visualized in the flowgraph. The audio should be clear, like listening to a normal radio, but it is dependent on the processing speed of the computer.

Customization of Receiving FM Radio Signals

This section outlines some of the additional functionality that GRC presents, like changing the FM radio station in real time and changing volume from the flowgraph view. This section also contains a video tutorial.

1. From the GRC flowgraph already created from the first part of this tutorial, further customizable options are available in order to listen to a wider range of frequencies and change the volume.
2. First, copy the *Variable* function block and paste it twice so there are three *Variable* blocks present in the flowgraph. These will aid in changing the radio station being listened to, and with updating the sample rate. The ID tags in each of the three blocks will, by default, be named *samp_rate*, *samp_rate0*, and *samp_rate1*.
 - a. In the second *Variable* block, change the ID of *samp_rate0* to *channel_width*, with the value of $200e3$.
 - b. These are variables and can be set in any blocks to universally change the variables in the rest of the flowgraph. After setting the channel width to $200e3$, change the decimation in the *Low Pass Filter* function to $int(samp_rate/channel_width)$. Since radio waves are transmitted at 480 KHz, this should be the final value of the flowgraph. By dividing the sample rate by the channel width in the Low Pass filter, you will not have to change the decimation values if you change the sample rate.
 - c. In the third *Variable* block, change the ID of *samp_rate1* to *center_freq*, with the value of the target radio station. For example, $98.9e6$ in the Burlington, VT area.
3. In order to be able to change radio stations within the flowgraph without reentering values throughout the flowgraph, add a *QT GUI Range*. This block has no input or output values.
 - a. Change the *ID* of the block to *channel_freq*.
 - b. Change the *Default Value* to the targeted radio station. For example, $98.9e6$.
 - c. Change the *Start* value to $87.9e5$ and the *Stop* value to $107.9e6$. These values represent the entire range at which FM radio signals broadcast.
 - d. Change the *Step* to $200e3$. This represents the distance between each tick on the slider being added. Because FM radio stations are 200 MHz wide, set this to $200e3$ so each tick on the slider represents a different radio station.

4. In order to control volume within the flowgraph via a slider, add a second *QT GUI Range* to the flowgraph. This block has no input or output values.
 - a. Change the *ID* of the block to *audio_gain*.
 - b. Change the *Default Value* to 1.
 - c. Change the *Start* value to 0 and the *Stop* value to 10. These values represent the highest and lowest audio volume.
 - d. Change the *Step* to 0.1.
5. Now that all of the values needed are within the *Variable* and *QT GUI Range* blocks, they need to be implemented into the flowgraph. In order to change the channel frequency, start by adding a *Signal Source* block. On its own, this would just generate the one targeted frequency, with a spike at the default 1000 Hz frequency. The *Signal Source* function can aid in changing the radio station by creating the variables in this block and multiplying those signals by the values in the *osmocom Source*.
 - a. Change the *Frequency* to “*center_freq - channel_freq*” to subtract the channel frequency from the center frequency.
 - b. Do not connect this block to any other function yet.
6. Now, add a *Multiply* function. This block has two input sources and one output source.
 - a. Delete the arrow between *osmocom Source* and the *Low Pass Filter* blocks. Do this by clicking on the connecting arrow and dragging away.
 - b. Position the *Multiply* function and connect its output to the *Low Pass Filter* input. Connect the *Signal Source* output to one of the *Multiply* inputs. Connect the *osmocom Source* output to the *Multiply* function input.
 - i. Notice that the *Signal Source* block has a grey input area. This is because it is not necessary to have an input connection with the *Signal Source* function.
7. After adding these functions, execute the flowgraph. The radio station can be changed by moving the *channel_freq* slider.
 - a. If no peak shows up initially, start slowly moving the slider to search for the signal. Once found, set the center frequency to this value so it doesn't have to be searched for again.
8. To add an additional slider to control volume, delete the arrow connecting the *WBFM Receive* function to the *Audio Sink* function.

9. Add a *Multiply Const* block, which, instead of multiplying two values from two different blocks, it will multiply the input value by the value set within this block.
 - a. Notice that the *Multiply Const* block has blue inputs and outputs, while the *WBFM Receive* block has an orange output. The easiest way to change from *Complex to Float* inputs and outputs is to click on the block and press the keyboard's down arrow. The input will change from blue to orange to green to yellow. Another way to change the input and output in order to match is to alter the properties within the *Multiply Const* block. Double click the block and change the *ID type* from *Complex* to *Float* from the drop down menu.
 - b. Connect the *WBFM Receive* output to the *Multiply Const* input, and the *Multiply Const* output to the *Audio Sink* input.
 - c. Set the Constant in the *Multiply Const* block to *audio_gain*.
 - d. Now, while executing the flowgraph, there will be a slider at the top of the screen labeled *audio_gain* to control the volume, along with the *channel_freq* slider to change the radio station.

Video 3 – [Adding Functionality to the Receive File](#)

Transmitting FM Radio Signals

This section outlines how to transmit radio signals to an open radio station through the use of the HackRF One device and the GNU Radio Companion software. This section also contains a video tutorial.

1. Upon starting the GNU Radio Companion software, there will be two boxes already present on the flowgraph area: *Variable* and *Options*. These default to a set rate and value, which can be altered, as mentioned above. In the *Options* block, the *Generate Options* should be set to *QT GUI*.
2. In the *Variable* block, change the sample rate (*samp_rate*) to the desired value. The lowest recommended sample rate is 2e6 (2 million), and the highest recommended sample rate is 20e6 (20 million). A good starting sample rate for transmitting radio signals is 2e6 (2 million), and increase from there. A computer with less processing speed should not have a high sample rate frequency.
3. To even begin the process of transmitting audio to an FM radio station via the HRF1 device, obtain a file source to transmit, such as any available '.wav' audio file. Please note that every '.wav' file may have its own audio rate (for example, 22050 Hz). This can be confirmed by checking the file's properties (by right-clicking the file in the folder and checking its Properties > Audio > Sample rate).
4. In GRC, add the *Wav File Source* function and place it into the flowgraph.

- a. From the *file* option within the block, add the desired ‘.wav’ file.
5. Add a *WBFM Transmit* function to the flowgraph.
 - a. Set both the *Audio Rate* and *Quadrature Rate* to an integer four times the file’s audio rate (for example, 88200, which needs to be an integer, not in scientific notation). This will generate a clearer sound from the audio file transmitted from the HRF1 without lowering the volume.
 - b. Set *Tau* to 25e-6. This will increase volume.
 - c. Connect the WBFM Transmit input to the *Wav File Source* output with an arrow.
6. Next, add a *Rational Resampler* block to the flowgraph.
 - a. Set the *Interpolation* value to 90 in order to decimate the sample rate to the correct value. If the audio rate of the .wav file is 44100 Hz, set the interpolation to 45.
7. Obtain a *QT GUI Range* to determine the broadcasting station.
 - a. Rename the ID as *freq*.
 - b. Set the *Default Value* and *Start* value inputs of the desired radio station to broadcast (for example, 87.7e6). Set the *Stop* frequency to 107.7e6, which is the highest FM radio value. Additionally, the *Step* value represents the width of each FM Radio channel in *Step*. Set this to 200e3.
8. Add an *Osmocom Sink* function, which will allow the Hack RF to transmit the signals.
 - a. Set the *Ch0: Frequency* to *freq*, which will enable the utilization of a slider within the pop up window in order to change the broadcast radio station.
9. Click the play button and save the flowgraph as a file. Upon executing the flowgraph, a new window will appear with just the slider. The .wav file uploaded to the *Wav File Source* will begin playing. Depending on the quality of the antennae used, the sound may be slightly distorted.

Video 4 – [Transmitting Signals from the Device](#)

Retransmitting FM Radio Signals

This section outlines the procedure to record an audio sample from a radio station and retransmit it to a different radio station. This section also contains a video tutorial.

1. First, follow the tutorials above or open the previously saved files for receiving (which will be referred to as *tutorial 1.grc*) and transmitting (which will be referred to as *tutorial 2.grc*) FM radio signals. Open these files in separate tabs of the same window.
2. In the *tutorial 1* tab, add a *Wav File Sink* function to allow the signals received to be saved as a '.wav' file.
 - a. Change the file path in the *File* prompt to somewhere on the desktop (for example, an empty folder on the desktop).
 - b. Change the number of channels (*N Channels*) to 2. This value needs to be 2 in order to double the number of samples to successfully save the file. This will enable the *Wav File Sink* function to have 2 inputs.
 - c. Set the *Sample Rate* to 48000.
 - d. Connect one of the inputs of the *Wav File Sink* to the *Multiply Const.*
3. Obtain a *Constant Source* function, and set the value to 0. This will keep the values within the .wav as real numbers.
 - a. Connect the second *Wav File Sink* input to the *Constant Source* output.
4. Execute the flowgraph for about 10 seconds. GRC will immediately begin recording the audio sample. It is recommended to only play the flowgraph for a few seconds because .wav files can become very large, very fast.
 - a. Again, a reminder to close the flowgraph via the red x in the upper left-hand corner of the flowgraph window, not the red x by the play button. Otherwise, the .wav file will not save properly.
5. Now, in the tab containing *tutorial 2*, from the *Wav File Source* function, select the recorded filename from *Tutorial 1* as the desired file to transmit.

6. Change the *Interpolation* value in the *Rational Resampler* function to 45. This is half of the previous *Interpolation* value of 90. This will prevent the file from sounding sped up or slowed down, since the sample rate of the recorded .wav file is different from the *WBFM Transmit Audio Rate*.
7. Now, by executing the flowgraph, the received file recorded from *tutorial 1* should play to any radio near the HRF1 antenna. That concludes the tutorial to retransmit FM radio files via the HRF1.

Video 5 – [Retransmitting Signals from the Device](#)

Live Broadcasting FM Radio Signals

1. In order to live broadcast, just delete the *Wav File Source* function. Replace it with the *Audio Source* function. This enables the computer's sound card, so instead of transmitting a recorded file, the audio played will be any sound picked up by the computer's microphone.
 - a. In the *Audio Source* block, change the *Sample Rate* to 44.1 KHz. This value aligns better with the other values in the flowgraph.
 - b. Connect the *Audio Source* output to the *WBFM Transmit* input.
2. By changing the *Audio Source* > *Sample Rate*, the *Variable* > *Sample Rate* should also be changed to 44100.
3. Set the *osmocom Sink* > *Sample Rate* back to 2000000 (2M). Doing this will unlink the default *samp_rate* within the function to the *Variable* block. These two values need to be different because the signal will not transmit correctly should the sample rate still be linked.
4. Execute the flowgraph and speak into the computer's microphone. Your voice should be heard by receiving radios near the HRF1 antenna.

Video 6 – <http://bit.ly/2jsLXTn> - Retransmitting and Broadcasting

Further Work

After documenting its setup, use, and functionality, our team proceeded to further research practical applications of the HackRF One. We found our research can be applied to a number of different radio spectrums and devices. For example, we explored modern technologies and consumer products that utilize radio frequency such as: Key Fobs from automobiles, NFC, and Bluetooth CR/LE.

In terms of Bluetooth, we learned about the processing of signals like frequency hopping. This was then studied in conjunction with the Ubertooth One, a device which specializes in the analysis of Bluetooth signals, in order to determine how to receive information into the Hack RF One. As a result, we were able to record and analyze data from devices that use this wireless technology. For example, MAC Address and RSSI information can be used to determine the distance of devices within a vicinity. These discovered devices can then be exploited by transmitting foreign data.

With the rise of consumer products that utilize wireless technologies, the HackRF One is a handy tool that allows the examination of communicating devices. We hope that our research will help others further the practical application of the HackRF One in future projects.

Appendix

Commonly Used Functions on GNU Radio Companion

1. **Osmocom Source** [No module specified → Sources] – Allows the HackRF One to enter ‘receive mode’ and communicate with another device.
 - a. **Sample rate** – Sets the number of samples per second the HackRF One receives or transmits, by default it is connected to variable *samp_rate*.
 - b. **Ch0 Frequency** – The center frequency that the Osmocom is taking samples of.
 - c. **Ch0 RF Gain (dB)** – Amplifies the strength of the signal, usually can be set to 0 without any issues.

2. **Osmocom Sink** [No module specified → Sinks] – Allows the HackRF One to enter ‘transmit mode’ in order to send out signals to other devices.
 - a. **Sample rate** – Connected to *samp_rate*, the number of samples per second the HackRF One is receiving or transmitting
 - b. **Ch0 Frequency** – The frequency that the HackRF One is transmitting to.

3. **QT GUI Sink** [Instrumentation → QT] – Visualizes the signals in four different graphs: A frequency sink, a time sink, a waterfall sink, and a constellation sink. These can be used individually on their own. Each sink is explained below.
 - a. **Center Frequency** – The frequency that the flowgraph is centered at on the x-axis.
 - b. **Refresh Rate** – How many times per second the graph updates. It can be lowered to increase performance on slower computers.

4. **QT GUI Frequency Sink (WX FFT Sink)** [Instrumentation → QT] – Visualizes signals as a waveform. The x-axis is the frequency of the signal (measured in Hz) and the y-axis is the power of the signal (measured in dB).
 - a. **Center Frequency** – Sets what frequency is at the center of the graph.
 - b. **Bandwidth** – Sets the width of the graph. For example, a bandwidth of 1,000 with a center frequency of 0 Hz would show the range of signals from -500 Hz to 500 Hz.

5. **QT Time Sink** (WX GUI Scope Sink) [Instrumentation → QT] – Visualizes signals over an x-axis of time and a y-axis of count (from 1 to -1). Useful for measuring the power of signals and seeing how long a transmission of a signal is.
6. **QT Constellation Sink** [Instrumentation → QT] – Shows signals represented by points on a complex number scale, with the real number component as the x-axis and the imaginary number component as the y-axis.
 - a. **Number of points** – Sets how many points are on the screen at a given time.
7. **QT Waterfall Sink** [Instrumentation → QT] – Visualizes signals over a waterfall, which is a three variable graph. The x-axis is the frequency, the y-axis is time, and the color of the waterfall is the amplitude of the signal (white being the highest amplitude, dark blue being the lowest).
 - a. **Center Frequency** – The frequency set to be at the center of the waterfall.
 - b. **Bandwidth** – Sets the width of the graph. For example, a bandwidth of 1,000 with a center frequency of 0 Hz would show the range of signals from -500 Hz to 500 Hz.
8. **QT GUI Range** (WX Slider) [GUI Widgets → QT] – Provides the ability to make real time adjustments to a variable using a slider.
 - a. **ID** – A unique identifier for the block. Set any value in another block to this ID instead of a number to utilize the slider. The name of the ID will also be how it appears in the GUI when the program is running.
 - b. **Default Value** – The value that will be used when the program is first started.
 - c. **Start/Stop** – The minimum and maximum values, respectively, that the slider can go to.
 - d. **Step** – The amount that the slider changes by with each tick. For example, if the step was 5, then the slider would only move in increments of 5.
9. **Signal Source** [Waveform Generators] – Provides a constant signal at a specified frequency.
 - a. **Waveform** – Sets the type of signal being created. The default waveform is cosine..
 - b. **Frequency** – Indicates what frequency the signal will be transmitting at.
 - c. **Amplitude** – Sets the strength of the generated signal.
10. **Low Pass Filter** [Filters] – Helps to clear out noise from the instrumentation sinks. It helps peaks stand out and gets rid of noise that could interfere with signals.
 - a. **Cutoff Frequency** – Sets how wide the non-filtered section is. A higher value means a wider band of signals not filtered out.

- b. **Transition Width** – Controls how steep the transition from filtered to not filtered signals is. A low value means a quicker and steeper transition.
- 11. **Quadrature Demod** [Modulators] – A complex to float converter that combines both the real and imaginary component of a complex number into a single float.
 - a. **Gain** – Set to 1, and by default is a variable with definitions typically not present in the flowgraph.
- 12. **Throttle** [Misc.] – Limits the data throughput to the specified sampling rate. This prevents GNU Radio from consuming all CPU resources.
- 13. **Float to Complex** [Type Converters] – Converts a complex number into its real and imaginary parts, providing a separate output for each part. Only one of the outputs needs to be connected for the block work, but both can be used.
- 14. **DC Blocker** [Filters] – Filters out the direct current voltage that is going into the HackRF One. If there is not a DC Blocker, there will be a spike in the middle of the graph that could be confused with an actual radio signal. Always set the value of the DC blocker to 1024.
- 15. **(WAV) File Sink** [File Operators] – Saves raw digital waveform information coming from the Hack RF (connect to Osmocom Source) into a file (specific path, set within block) without processing it further. It can also be saved as an audio file (in .wav format). ****Caution!** The size of the file can become big very quickly, only use the HackRF One in conjunction with a file sink for a few seconds.
- 16. **(WAV) File Source** [File Operators] – Runs the file from File Sink in a new flowgraph. Specify whether the process is repeated, over a period of time, or not. Can also be a .wav file.
- 17. **Multiply** [Math Operators] – Multiplies two signals by one another.
- 18. **Multiply Const.** [Math Operators] – Multiplies input by a constant into an output, set the constant within the block itself.
- 19. **WBFM Receive** [Modulators] – FM demodulator, turns FM radio signals into audio signals. Note that this is a Complex to Float converter with one output.
 - a. **Quadrature Rate** – The rate that signals are coming into the block, it should be sent to whatever the sample rate is at the time of coming into the block.
 - b. **Decimation** – Divides the sample rate by the specified integer.
- 20. **WBFM Transmit** [Modulators] – FM modulator, turns an audio source into an FM signal that can be broadcasted to an FM radio station.

- a. **Audio Rate** - The rate that samples are sent to the sound card in the computer.
 - b. **Quadrature Rate** – The rate that samples are coming into the block. In a WBFM Transmit block it must have a number that is equal to the Audio Rate divided by an integer.
21. **Audio Sink** [Audio] – Allows user to listen to signals within GRC. The sample rate should be set to the audio rate of the source that is being listened to.
22. **Rational Resampler** [Resamplers] – Changes the sample rate by a ratio. The sample rate is multiplied by the interpolation and then divided by the decimation in the same block. This is helpful because GRC only allows for interpolating/decimating the sample rate by integers.

FCC ID

Although the above demonstrated the use of the HRF1 device with FM Radio Signals, there are also capabilities to receive and transmit along other signals. It is important to note that other devices, other than the FM Radio Tuner, will more than likely operate along these other signals of varying frequencies.

The **FCC ID** of the other device will be needed to obtain information about the device such as its radio frequencies and documents available to the public about the operation of the device. Most devices have been reviewed and tested by the Federal Communications Commission, and this information will be available via “fcc.gov”. Assuming that the chosen device has been reviewed and tested by the Federal Communications Commission, the FCC ID will give information on the frequency that the device operates on, pictures of the interior of the device, and more useful information.

1. A good test device is something simple that has a clear on and off key that transmits signals. For example, car keys, a garage door opener, a TV remote, etc.
2. Proceed through the following options:
 - a. Assuming that the chosen device has been reviewed and tested by the **Federal Communications Commission**, enter ‘**fcc.gov**’ into the URL bar of the browser to access the database.
 - i. Proceed through the following options: **Licensing and Databases** → **Databases** → **Equipment Authorization Search**

Home / Licensing & Databases /

Search FCC Databases

Licensing & Databases

Overview

About Licensing

Databases

Fees

Forms

FCC Registration System (CORES)

Explore granular search interfaces into more than 40 specialized FCC databases such as radio call signs and equipment authorization.

- AM Radio Station Search
- Antenna Structure Registration – Application Search
- Antenna Structure Registration – Registration Search
- Cable Search
- Call Sign Query
- Children's Programming Report Search
- Consolidated Public Database System – Antenna Search
- Consolidated Public Database System – Application Search
- Consolidated Public Database System – EEO Filing Search
- Consolidated Public Database System – Ownership Report Search
- Consolidated Public Database System – Station Search
- Consumer Complaints Search
- Earth Station Location Search
- EAS Test Reporting System
- ECFS Search – Filings
- ECFS Search - Proceedings
- Electronic ARMIS Filing System
- Electronic Tariff Filing System Search
- **Equipment Authorization Search**
- Equipment Authorization System Grantee Search
- Equipment Authorization System Pending Application Search
- Equipment Authorization System Test Firm Search
- Experimental Licensing System – Call Sign Search
- Experimental Licensing System – Generic Search

Figure 12: (Equip. Auth. Search)

Federal Communications Commission
 Browse by CATEGORY BUREAUS & OFFICES Search
 About the FCC Proceedings & Actions Licensing & Databases Reports & Research News & Events For Consumers

Overview	ASR	EA	ETRS	MyIBFS	TCB
About Licensing	CDBS	ECFS	GenMen	NORS	TCNS
Databases	COALS	EDOCS	HAM	PIF	ULS
Fees	CORES	ELS	KDB	RLD	VPD
Forms	CSRS	ETFS	KIDVID	SADCS	
FCC Registration System (CORES)	DIRS		LMS		

Figure 13: (Licensing and Database)

- b. Input the FCC ID into the search box.
 - i. Depending on the manufacturer, this will differ on location, but will generally be clearly defined and found on the outer layer of the physical device.

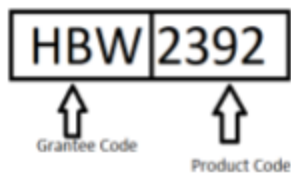


Figure 14: (FCC ID)



Figure 15: (FCC Example)

Equipment Authorization Search

Filing Options

[Grantee Registration](#)

[Modify Grantee Information](#)

[Submit Correspondence](#)

[Renew Test Firm/Act Exhibit](#)

[Test Firm Accredited Body Logic](#)

[Return to 158 Form](#)

[Add/Modify Grant Deferral Data](#)

[Change Short-Term Confidential Date](#)

Reports

[Pending Application Status](#)

[Authorization Search](#)

[Grantee Search](#)

[Pending Grantee Search](#)

[TCB Search](#)

[Test Firm](#)

[Test Firm Accredited Bodies](#)

[Equipment Class/Rule Part List](#)

Miscellaneous

[Get FRN!](#)

[Knowledge Databases](#)

[Hearing Aid Compatibility Status Reporting](#)

[Measurement Procedures](#)

Application Information:

Grantee Code: (First three or five characters of FCCID)

Product Code: Exact Match (Remaining characters of FCCID)

Applicant Name:

Final Action Date Range (mm/dd/yyyy): to

Grant Comments:

Application Purpose:

Software Defined Radios:

FCC Approved Applications Only:

TCB Approved Applications Only:

Composite Applications Only:

Grant Note: & & [View Grant Note Descriptions](#)

Test Firm:

Application Status:

Equipment Information:

Equipment Class:

Frequency Range in MHz: to Exact Match

Necessary Bandwidth:

Emission Designator:

Frequency Tolerance: to Exact Match

Power Output (in Watts): to Exact Match

Rule Parts (up to three): & & Exact Match

Product Description:

Modular Type:

OR show all modular OR show all non-modular

TCB Information:

TCB Name:

TCB Scope:

Formatting Options:

Show results in format

Show Records at a Time (HTML output only)

Figure 16: (EAS Form)

- ii. Enter the **Grantee Code** and **Product Code**, click on ‘Start Search’, and proceed to the product's ‘detail summary’ and view the ‘User Manual’.
 1. If there is more than one option for the device, check the external photographs to see which one matches the physical device. The operating frequencies will be listed to the right of each device listing.

6 results were found that match the search criteria:
 Grantee Code: **HBW** Product Code: **2392**

Displaying records 1 through 6 of 6.

View Form	Display Exhibits	Display Grant	Display Correspondence	Applicant Name	Address	City	State	Country	Zip Code	FCC ID	Application Purpose	Final Action Date	Lower Frequency In MHz	Upper Frequency In MHz
	Detail Summary			Chamberlain Group Inc,	The 300 Windsor Dr	Oak Brook	IL	United States	60523	HBW2392	Original Equipment	10/29/2007	300.0	300.0
	Detail Summary			Chamberlain Group Inc,	The 300 Windsor Dr	Oak Brook	IL	United States	60523	HBW2392	Original Equipment	10/29/2007	310.0	310.0
	Detail Summary			Chamberlain Group Inc,	The 300 Windsor Dr	Oak Brook	IL	United States	60523	HBW2392	Original Equipment	10/29/2007	315.0	315.0
	Detail Summary			Chamberlain Group Inc,	The 300 Windsor Dr	Oak Brook	IL	United States	60523	HBW2392	Original Equipment	10/29/2007	318.0	318.0
	Detail Summary			Chamberlain Group Inc,	The 300 Windsor Dr	Oak Brook	IL	United States	60523	HBW2392	Original Equipment	10/29/2007	372.5	372.5
	Detail Summary			Chamberlain Group Inc,	The 300 Windsor Dr	Oak Brook	IL	United States	60523	HBW2392	Original Equipment	10/29/2007	390.0	390.0

Figure 17: (Equip. Frequency)

OET Exhibits List

8 Matches found for FCC ID **HBW2392**

View Attachment	Exhibit Type	Date Submitted to FCC	Display Type	Date Available
Cover Letter Requesting Confidentiality	Cover Letter(s)	10/29/2007	pdf	10/29/2007
Cover Letter Requesting Confidentiality	Cover Letter(s)	10/29/2007	pdf	10/29/2007
Exhibit C External Photographs per 2 1033 b7	External Photos	10/29/2007	pdf	10/29/2007
Exhibit A ID Label and Location Info per 2 1033 b7	ID Label/Location Info	10/29/2007	pdf	10/29/2007
Exhibit C Internal Photographs per 2 1033 b7	Internal Photos	10/29/2007	pdf	10/29/2007
Exhibit E Test Measurement Report per 2 1033 b6	Test Report	10/29/2007	pdf	10/29/2007
Exhibit C RE Test Setup photographs	Test Setup Photos	10/29/2007	pdf	10/29/2007
Exhibit D Users Manual per 2 1033 b3	Users Manual	10/29/2007	pdf	10/29/2007

Figure 18: (Device Photos)

FCC > FCC E-File > EAS > Grantee Search

Enter in as much info as you can about the device.

Grantee Information:

Grantee Name:

Grantee Code:

EBN:

Address:

P.O. Box:

City:

State:

Zip/Postal Code:

Country:

Grantee Contact Information:

First Name:

Middle Name:

Last Name:

Title:

Telephone Number:

Extension:

Fax Number:

E-mail Address:

Mail Stop:

Figure 19: (Grantee Search)

Home / Licensing & Databases /

Search FCC Databases

Licensing & Databases

- Overview
- About Licensing
- Databases**
- Fees
- Forms
- FCC Registration System (CORES)

Explore granular search interfaces into more than 40 specialized FCC databases such as radio call signs and equipment authorization.

- AM Radio Station Search
- Antenna Structure Registration - Application Search
- Antenna Structure Registration - Registration Search
- Cable Search
- Call Sign Query
- Children's Programming Report Search
- Consolidated Public Database System - Antenna Search
- Consolidated Public Database System - Application Search
- Consolidated Public Database System - EEO Filing Search
- Consolidated Public Database System - Ownership Report Search
- Consolidated Public Database System - Station Search
- Consumer Complaints Search
- Earth Station Location Search
- EAS Test Reporting System
- ECFS Search - Filings
- ECFS Search - Proceedings
- Electronic ARMS Filing System
- Electronic Tariff Filing System Search
- Equipment Authorization Search
- Equipment Authorization System Grantee Search**
- Equipment Authorization System Pending Application Search
- Equipment Authorization System Test Firm Search
- Experimental Licensing System - Call Sign Search

Figure 20: (EASG Search)

View Form	Display Exhibits	Display Grant	Display Correspondence	Applicant Name	Address	City	State	Country	Zip Code	FCC ID
	Detail Summary			Logitech Far East Ltd	#2 Creation Rd. 4, Science-Based Ind. Park	Hsinchu	N/A	Taiwan	300	JNZMR0061

Figure 21: (Grantee Code Details)

- c. If the FCC ID cannot be found on the device, it can be searched by the manufacturer. Underneath Equipment Authorization Search, there is Equipment Authorization System Grantee Search.
 - i. Click on detail. There should be a list of documents relating to that device. The user manual is the document most likely to list the frequencies which the device functions on.

OET Exhibits List

10 Matches found for FCC ID JNZMR0061

View Attachment	Exhibit Type	Date Submitted to FCC	Display Type	Date Available
Attestation - Label Location Declaration.pdf	Attestation Statements	12/07/2015	pdf	12/07/2015
Confidentiality Request.pdf	Cover Letter(s)	12/07/2015	pdf	12/07/2015
Cover Letter - Agent Authorization.pdf	Cover Letter(s)	12/07/2015	pdf	12/07/2015
External Photos.pdf	External Photos	12/07/2015	pdf	05/10/2016
ID Label Location Information.pdf	ID Label/Location Info	12/07/2015	pdf	12/07/2015
Internal Photos.pdf	Internal Photos	12/07/2015	pdf	05/10/2016
Test Report.pdf	Test Report	12/07/2015	pdf	12/07/2015
Test Setup Photos.pdf	Test Setup Photos	12/07/2015	pdf	05/10/2016
User Manual 1 of 2.pdf	Users Manual	12/07/2015	pdf	05/10/2016
User Manual 2 of 2 - Compliance Statements.pdf	Users Manual	12/07/2015	pdf	05/10/2016

Figure 22: (Device User Manual)

References

- Amplitude modulation. (2017, November 05). Retrieved from
https://en.wikipedia.org/wiki/Amplitude_modulation
- ANT500. (n.d.) Retrieved from
<https://greatscottgadgets.com/ant500/>
- Christensson, P. (2015, August 22). Sample Rate. Retrieved from
https://techterms.com/definition/sample_rate
- FAQ. (n.d.) Retrieved from the HackRF Wiki:
<https://github.com/mossmann/hackrf/wiki/FAQ>
- GNU Radio Companion. (n.d.). Retrieved from the GNU Radio Wiki:
<https://wiki.gnuradio.org/index.php/GNURadioCompanion>
- Georgia State University. (n.d.). [AM/FM Radio Waves Visual]. Retrieved from
<http://springfield.gsu.edu/omeka/files/fullsize/6f4be87348f12211687c2529d8be5387.jpg>
- Getting Started with the HackRF and GNU Radio. (n.d.). Retrieved from the HackRF Wiki:
<https://github.com/mossmann/hackrf/wiki/Getting-Started-with-HackRF-and-GNU-Radio>
- Guided Tutorial GRC. (n.d.). Retrieved from the GNU Radio Wiki:
https://wiki.gnuradio.org/index.php/Guided_Tutorial_GRC
- Guided Tutorial Introduction. (n.d.). Retrieved from the GNU Radio Wiki:
https://wiki.gnuradio.org/index.php/Guided_Tutorial_Introduction
- HackRF One. (n.d.). Retrieved from the HackRF Wiki:
<https://github.com/mossmann/hackrf/wiki/HackRF-One>
- Low-pass filter. (2017, November 07). Retrieved from https://en.wikipedia.org/wiki/Low-pass_filter
- Northwestern University. (n.d.). How is data put on radio waves? Retrieved from
<http://www.qrg.northwestern.edu/projects/vss/docs/communications/1-how-is-data-put-on-radio-waves.html>
- Northwestern University. (n.d.). What are radio waves? Retrieved from
<http://www.qrg.northwestern.edu/projects/vss/docs/communications/1-what-are-radio-waves.html>
- Ossmann, M. (2014). *Software Defined Radio with HackRF* Lessons 1-11 [Videos]. Retrieved from
<https://greatscottgadgets.com/sdr/>

Ossmann, M. (n.d.). *HackRF One*. Retrieved from

<https://greatscottgadgets.com/hackrf/>

Radio frequency. (n.d.). In *Merriam-Webster*. Retrieved from

<https://www.merriam-webster.com/dictionary/radio%20frequency>

Radio receiver. (2017, November 09). Retrieved from https://en.wikipedia.org/wiki/Radio_receiver

Rouse, M. (n.d.). What is digital signal processing (DSP)? - Definition from WhatIs.com. Retrieved from

<http://whatis.techtarget.com/definition/digital-signal-processing-DSP>

Rouse, M. (n.d.). What is software-defined radio (SDR)? - Definition from WhatIs.com. Retrieved from

<http://searchnetworking.techtarget.com/definition/software-defined-radio>

Transmit. (2014). In *Collins English Dictionary – Complete and Unabridged, 12th Edition 2014*. Retrieved from <https://www.thefreedictionary.com/transmit>