NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

SUPPLEMENTAL TO THESIS
(14 of 14: GNU RADIO QUICK START GUIDE)

A NOISE-MASKING PHYSICAL LAYER SECURITY METHOD FOR MULTIPLE-INPUT, MULTIPLE-OUTPUT DIGITAL COMMUNICATION SYSTEMS

by

Robert C. Sellin

March 2018

Thesis Advisor:	Frank Kragh
Second Reader:	Ric Romero

Approved for public release. Distribution is unlimited.

GNU RADIO Quick Start Guide

R.C. Sellin **Dec 2017**

This guide is meant for an MSEE student whose research dictates use of SDR, but who has no experience with gnu radio. Gnu radio is an open source software platform that allows communication hobbyists and engineers develop and share pieces of software defined communication and signal processing systems. Though all aspects *can* be done through Python coding and command line interaction, the real power for students is in the gnu radio companion (GRC). Basically, the blocks are written in C/C++ and connected by Python, but really executed via GRC. Plan to avoid coding until you run into a block/connection need that is not already met by GRC.

1. Download and install correct version for your machine from http://www.gcndevelopment.com/gnuradio/downloads.htm.

Follow instructions for your hardware and software setup. You may be tempted to partition a hard drive like you would for a vm, but that is not necessary for even advanced gnu functionality.

2. Watch Intro to Gnu radio Part 1 (1 hr): https://www.youtube.com/watch?v=_hGNT1w-jig . Mostly install and very basic communications and DSP concepts. Alternatively you can jump straight into the tutorials at **https://wiki.gnuradio.org/index.php/Guided_Tutorial_GRC.**

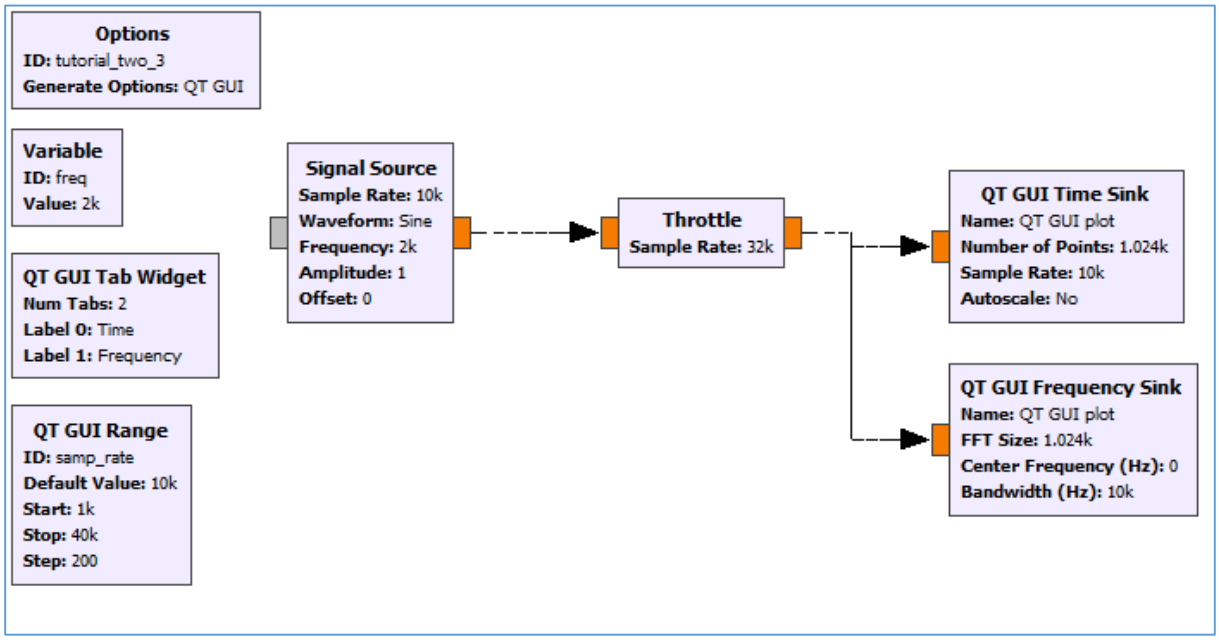
The linked tutorials are relatively straightforward and should be done in order.

Important Basics (READ AFTER YOU HAVE INSTALLED BUT BEFORE TUTORIALS):

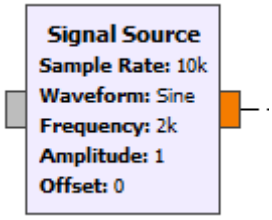
a. Use blocks from the right side column list to build your flow graph, editing the specific properties of each along the way. Then connect up your components (blocks), generate your flow graph (think compile), and execute. Inputs can be real (through an antenna or microphone) or generated by the software. Output can be real (through a soundcard or external device) or can be displayed in many fashions.

b. When a flow graph is displayed in the main space of GRC, each parameter has a value displayed. This may or may not be the actual assignment for that parameter; therefore, a screen capture of a flow graph is NOT SUFFICIENT to reproduce it. See example from tutorial 2.2.3 below.

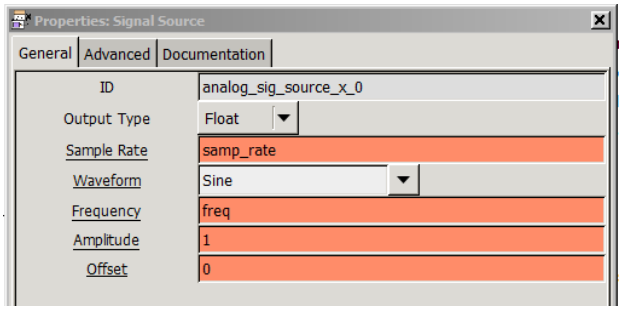
Here is the flow graph:



Instead of trying to digest the whole thing, let's get some basics down. Look at the source block:

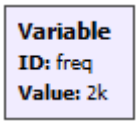


This represents a simple wave generator. This signal source block shows its sample rate as 10k, the waveform as Sine, frequency as 2k, and amplitude as 1. What you **cannot** tell from the block is whether those are static values or the default of a variable value. Double clicking on the block (or right clicking and selecting "Properties"), will bring up the following window:

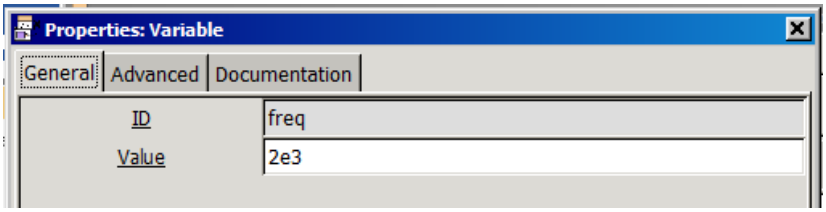


This window reveals that the amplitude is in fact a static value of 1, but the sample rate and frequency are controlled by the variables *samp_rate* (the default variable for every flow graph), and *freq*. (Note separately that the output type 'float' is represented as the color orange on the

output of the block. See Help-Types for more.) These variables (*samp_rate* and *freq*) must be located on the flow graph so that the signal source block can call these values. Find the block that contains the variable *freq*:

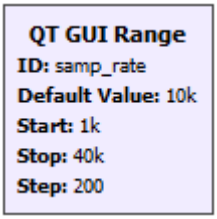


This block tells you that *freq* value is 2k, but again, that may or may not be a static value, check out the properties.



This is in fact a static value. So why would I ever set up a flow graph like this? In general, we build simple things, test them, and then add complexity and repeat. This approach will easily allow us to make the value of *freq* variable in later instances and apply that variability to the output.

Check out the block containing the other variable, *samp_rate* (remember we are still figuring out our source):



This one is clearly not a constant, but can take on a range of values that will be controlled by the GUI your flow graph is going to output. Notice that the default value is the value that is displayed on your source block. The tutorial does a pretty good job of explaining the GUI control (see 2 below for more there). If you get stuck at any point, switch to YouTube and watch:

1. Intro to Gnu radio Part 2 (1 hr) : https://www.youtube.com/watch?v=cg3TA3EDx78 . Covered basic filter construction and gui interaction. Built and see basic noise filter with real-time bandwidth slider control. Throttle block introduced. –OR-

2. The following is a particularly good resource is you are stuck, or if you have SDR experience but need a refresher. (also very helpful if you get sidetracked from your research for a few weeks and need to get back up to speed): https://hackaday.com/2015/11/11/getting-started-with-gnu-radio/. This source explains the display concepts, particularly the “GUI hint” range.

If you are interested in the depths of software design and filtering, see:.

1. Intro to Gnu radio Part 3 (1 hr): https://www.youtube.com/watch?v=nemfS9QAYHc. More on throttle and sample rate. Overall the first 30 mins are good and get the user past the point of the first tutorial but with much more background.

2. Reading for the python tutorials https://wiki.gnuradio.org/index.php/TutorialsWritePythonApplications

3. Intro to Gnu Radio Part 4 (1 hr): https://www.youtube.com/watch?v=94R2qE7mEc4 .

4. And for a much more advanced project, try: https://hackaday.com/2015/11/12/your-first-gnu-radio-receiver-with-sdrplay/