

Introduction to GNU Radio Companion

Linux and GRC Basics

Dr Heather Lomond CEng MIET M0HMO



Introduction to GNU Radio Companion

Linux and GRC Basics

Dr Heather Lomond CEng MIET M0HMO

This presentation is released under a Creative Commons licence.

Details of how and where you may re use or modify this this can
be found at

<http://creativecommons.org/licenses/by-sa/4.0/>







Let's get going!

If you haven't already done so, plug in your USB stick and power it all up!

Linux (Kernel and Distros)

- Linux, on its own, doesn't do anything. It is just a set of functions and drivers etc. (called the Linux Kernel) that allow other programs to do things.
- Many people have written front ends that humans can interact with – these are called distributions. Ubuntu, Raspbian, Mint, Debian are all distributions. They all look and feel different but underneath is the same, good old Linux.
- Apple OS and Android are similar in that they give the user a front end (GUI) but underneath is a Linux kernel

Linux (Ubuntu in 30 seconds)

- Double Left Click on an icon to run it. E.g. to run the GNU Radio Companion click on this icon: 
- To shutdown the computer, Right Click on the cog icon (top right) 
-  Navigate your files/directories in the GUI using the file manager.
- To open a “Terminal” window (similar to “Command” or “DOS” in Windows), 

Linux (command line)

- Linux uses ‘/’ to denote levels in directories (DOS uses ‘\’)
e.g.

```
cd heather/temp/new_files
```

- Linux is case sensitive so

```
cd Heather      and
```

```
cd heather      are not the same thing
```

- You list the files using `ls` (equivalent to “dir” in DOS),
for more detail use `ls -al`

Linux (File structure)

- In windows lots of things are at the top level of the file structure – network drives, hard disks, USB drives etc. are all up at the top and labelled as C: or X:\\myNAS\ and so on. In Linux you have what is called the root. Get to this by typing `cd /`
- All drives, USB sticks etc. live below this level. Mostly in the /mnt directory but sometimes elsewhere.
- At a simplistic level, everything in Linux is a file. So a serial device would appear in Device Manager in Windows but it will appear as a file (/dev/Ser0 perhaps) in Linux.

Linux (More commands)

- `cp` to copy files
- `rm` to remove (delete) files

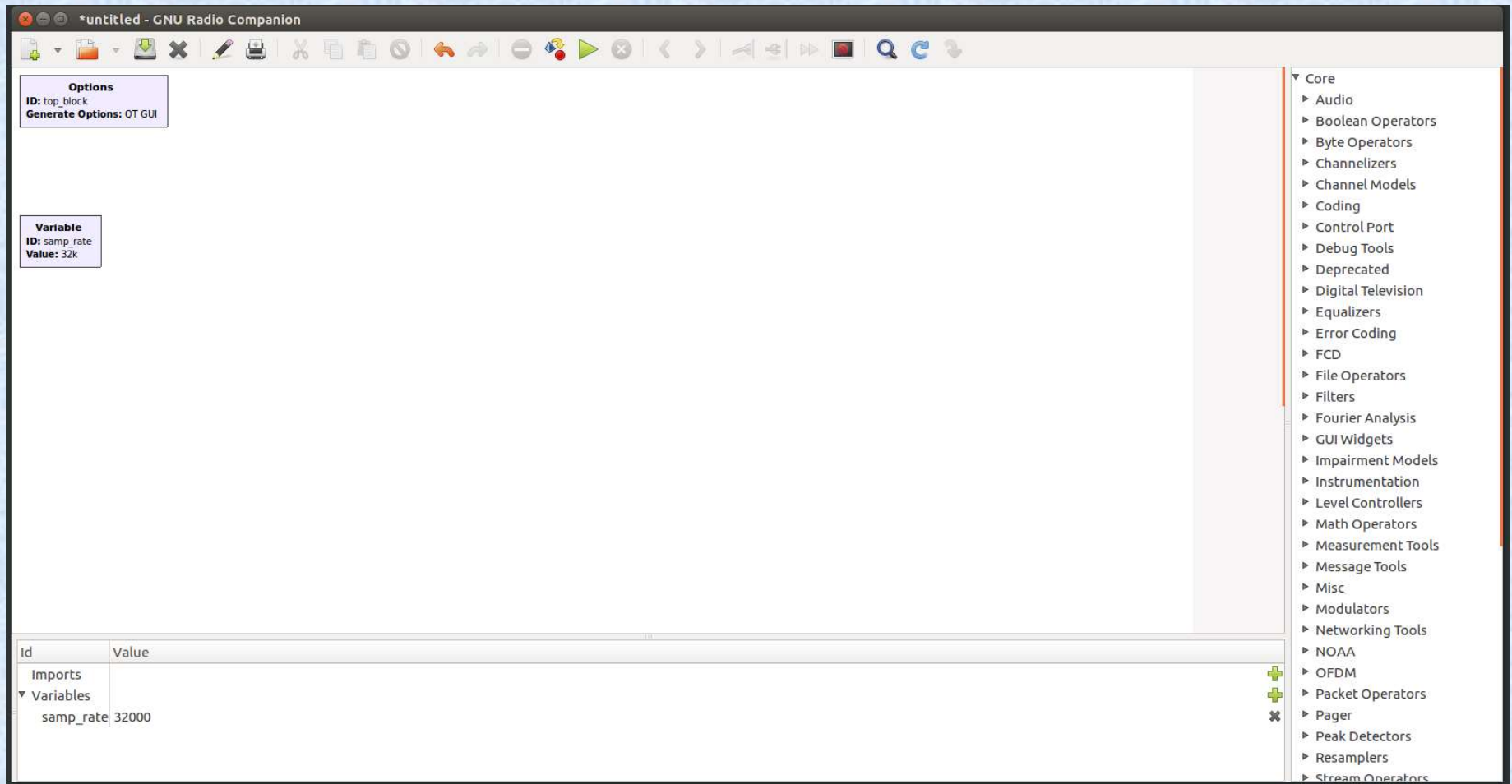
GNU Radio

- GNU Radio is a set of software blocks that you can combine to make all sorts of software defined radio and signal processing applications
- Designed for simulation as well as real world applications
- Free & Open Source
- Program in C++ or Python

Gnu Radio Companion

- A graphical interface for easing the use of the GNU Radio Blocks

The GRC Window



The screenshot displays the GNU Radio Companion (GRC) interface. The window title is "*untitled - GNU Radio Companion". The top toolbar contains various icons for file operations, editing, and execution. The main workspace is currently empty, showing two panels on the left:

- Options**
ID: top_block
Generate Options: QT GUI
- Variable**
ID: samp_rate
Value: 32k

At the bottom left, a table displays the current configuration:

Id	Value
Imports	
Variables	
samp_rate	32000

On the right side, a component library is visible, listing various blocks categorized by function:

- Core
 - Audio
 - Boolean Operators
 - Byte Operators
 - Channelizers
 - Channel Models
 - Coding
 - Control Port
 - Debug Tools
 - Deprecated
 - Digital Television
 - Equalizers
 - Error Coding
 - FCD
 - File Operators
 - Filters
 - Fourier Analysis
 - GUI Widgets
 - Impairment Models
 - Instrumentation
 - Level Controllers
 - Math Operators
 - Measurement Tools
 - Message Tools
 - Misc
 - Modulators
 - Networking Tools
 - NOAA
 - OFDM
 - Packet Operators
 - Pager
 - Peak Detectors
 - Resamplers
 - Stream Operators

The Menu Strip



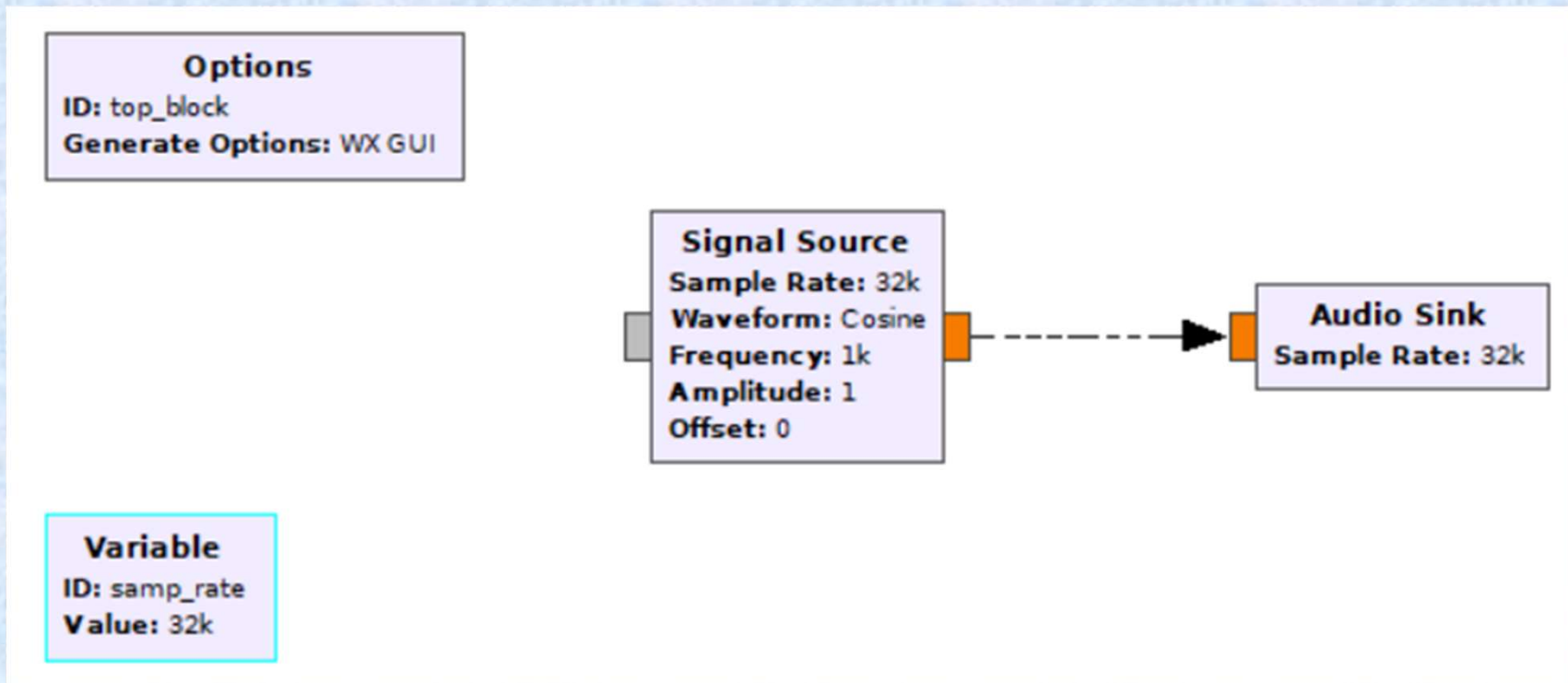
Things to know

- There are some graphical display (GUI) blocks that were written in the QT IDE and some developed in WX. They are not interchangeable so you need to decide which set you are going to use at the start.
 - WX widgets are more intuitive and have more GUI features.
 - QT is the preferred option going forward.
- Sources give out data (e.g. a microphone).
- Sinks take in data (e.g. speakers or a waveform display widget).

Useful Things To Do At The Start

- **View -> Find Blocks** This lets you search the blocks for keywords
- Toggle the variables view to see the errors and status text

Build our first flowgraph



Web Links

- **The Windows installer is here:**
<https://wiki.gnuradio.org/index.php/InstallingGR>
- **The Instruction Manual for all the underlying GNU radio modules**
<https://www.gnuradio.org/doc/doxygen/index.html>
- **The top level gnuradio website**
<https://www.gnuradio.org/>
- **All the tutorials**
<https://wiki.gnuradio.org/index.php/Tutorials>

The End of Linux and GRC Basics

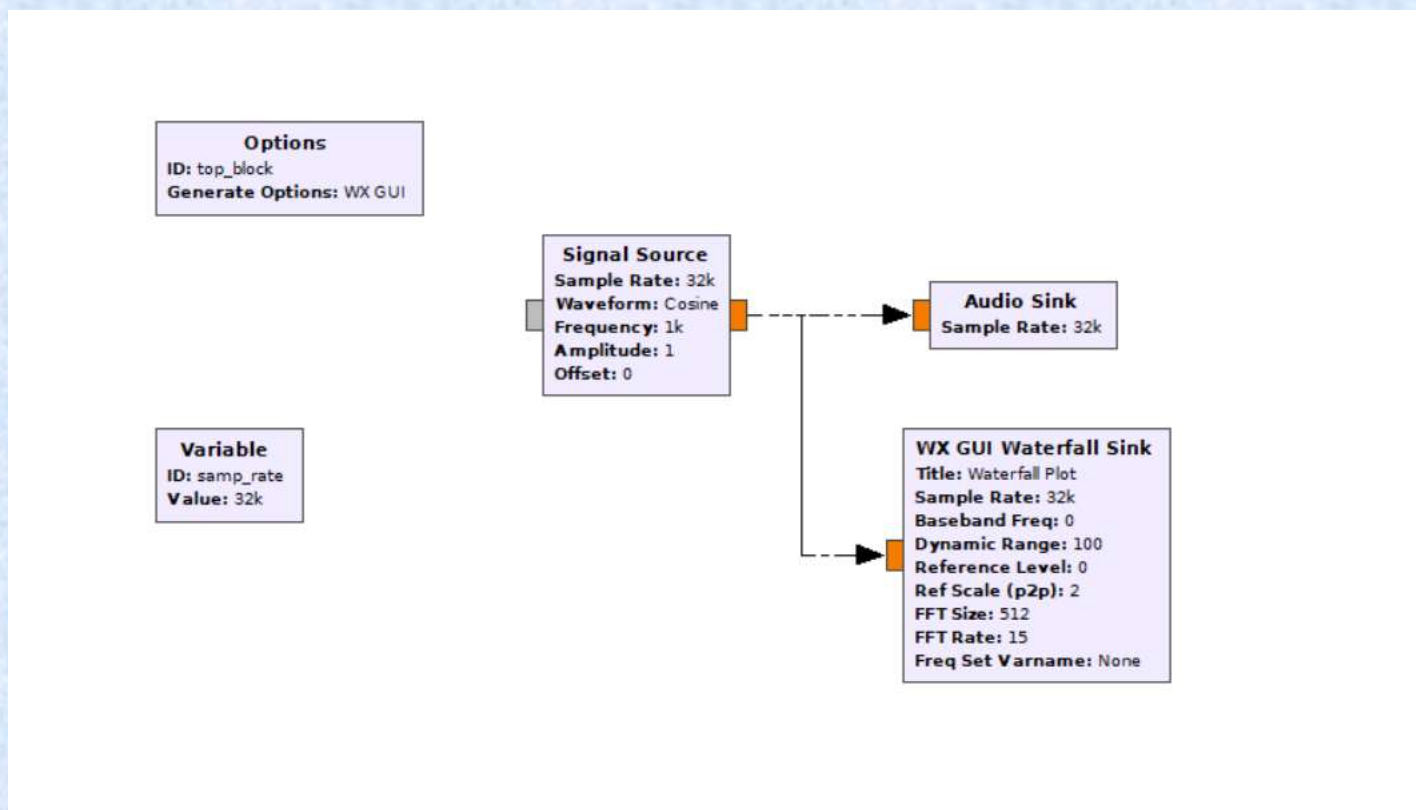
Introduction to GNU Radio Companion

Building Our first Receiver

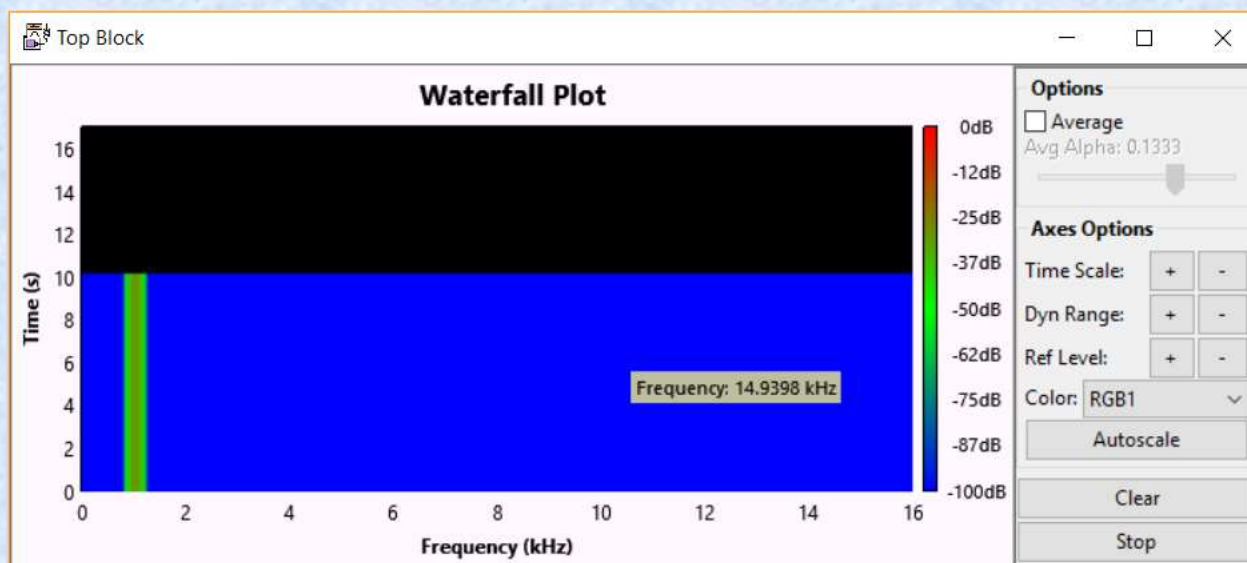
Dr Heather Lomond CEng MIET M0HMO



Instrumentation



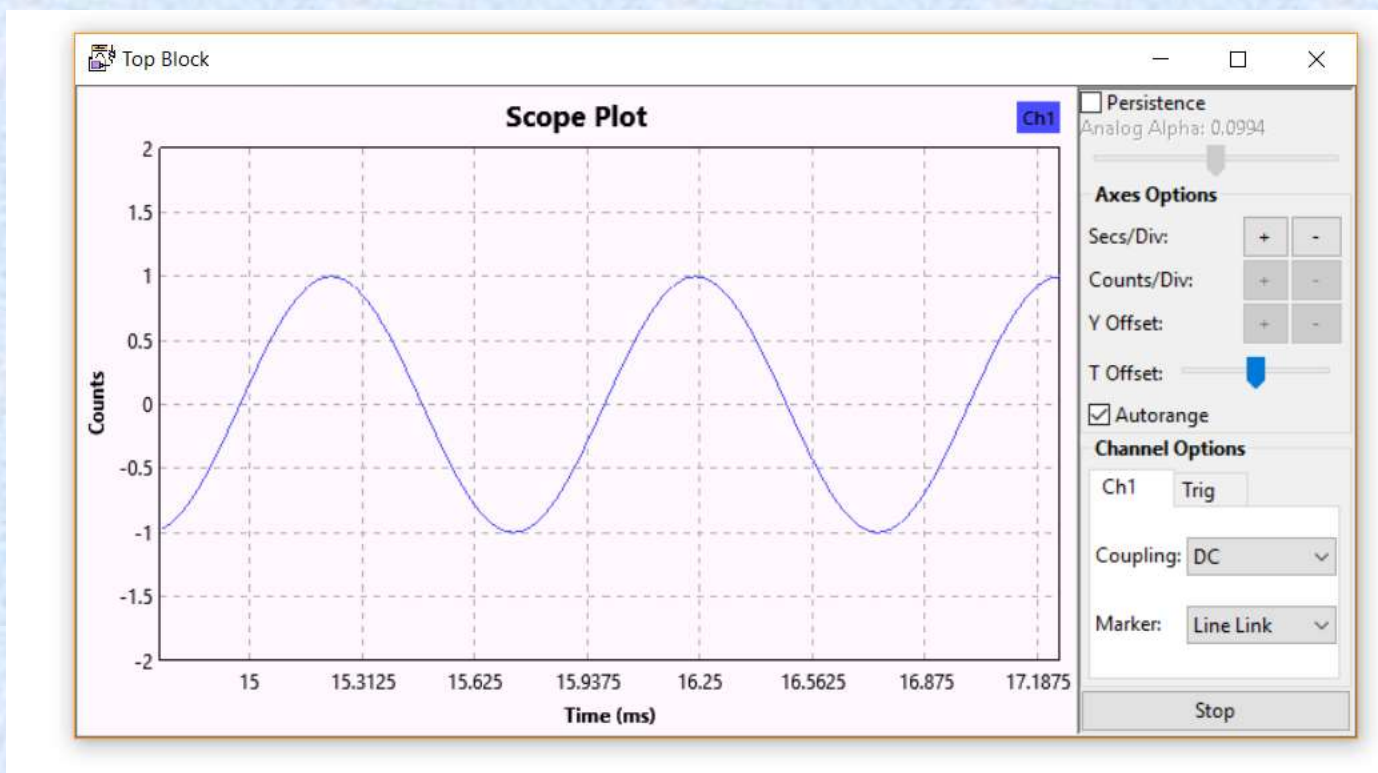
The Output



Exercise 1

- Add a scope instrument to the flowgraph and connect it to the output of the signal source.
- Set up the scope block to autorange for Voltage and Time scales.
- Disable the waterfall.
- Execute it to get ...

Result for Exercise 1



Simulation vs. real world

- What we have right now is a “real” flowgraph. It does stuff in the physical world i.e it makes the speaker sound.
- In this case, the output to the speakers is setting the real world time for the sample rate.
- If we delete the Audio Sink block, then the flowgraph has nothing to physically limit how fast it does everything, so it will go flat out processing the signals.
- To stop loading up the CPU we can put in a Throttle block which just makes sure that things happen in real time (it acts as if there was a real world element to the flowgraph).

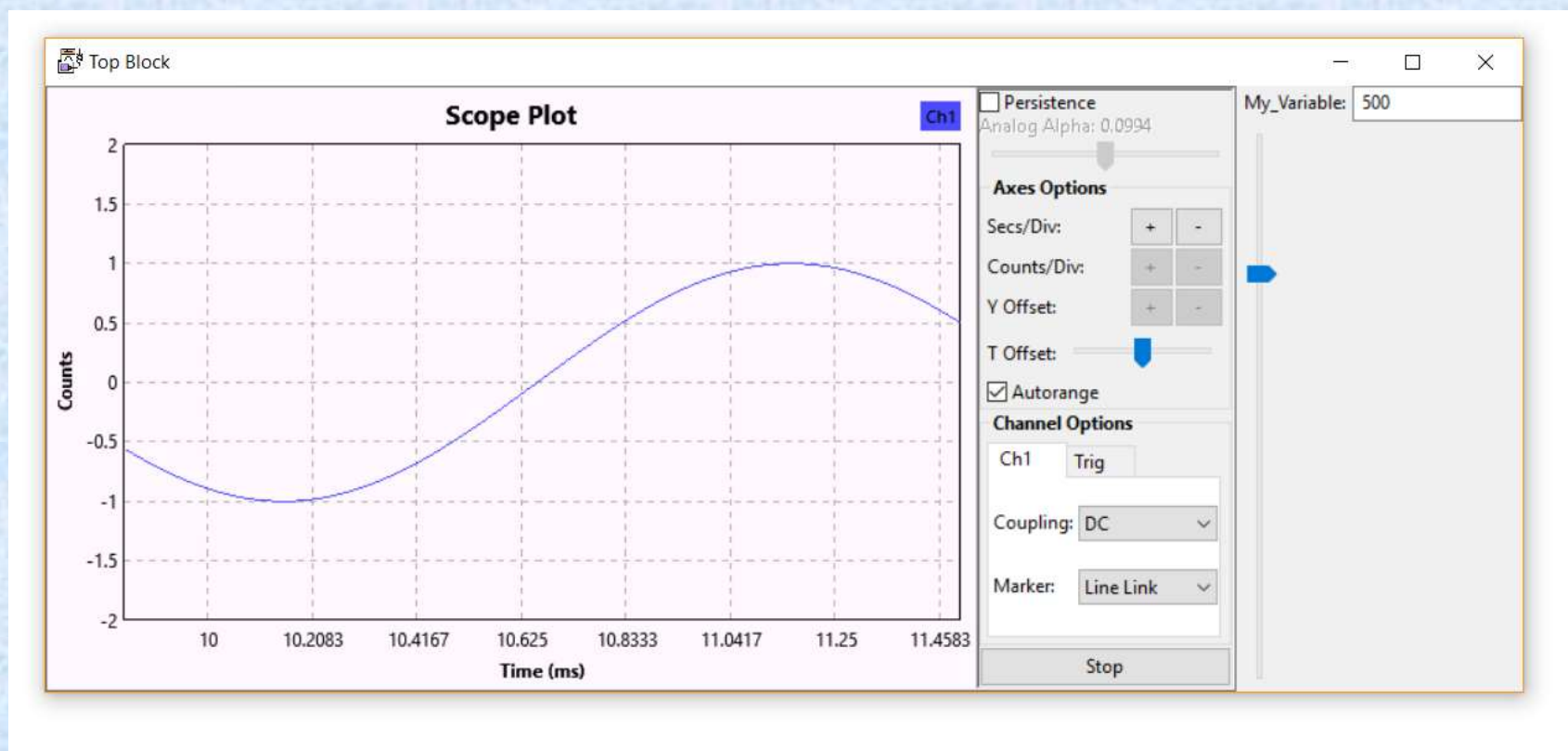
Exercise 2

- Disable the Audio Sink
- Execute the flowgraph and watch the CPU usage
- Stop it, add a Throttle between the Signal Source and the Scope.
- Note on the way by that the throttle can act on all the different data types
- Run again and see the CPU usage go down.

Some GUI elements

- Take a look at GUI Widgets / WX. This contains a selection of possible GUI elements.
- Re-enable the Audio Sink, remove the throttle.
- Add a Text Box, change ID to “My_Variable”, also note the “Grid Position” box for later.
- In the Signal Source change the frequency to “My_Variable” as well.
- Run it.

Result for Exercise 3



Some Useful Blocks

- Let's take a look at some real radio stuff:

WX GUI FFT Sink

RTL-SDR Source

Rational Resampler

Frequency Xlating FIR Filter

WAV File Source

WX GUI FFT Sink

Properties: WX GUI FFT Sink

General | Advanced | Documentation

ID	wxgui_fftsink2_0
Type	Complex ▾
Title	FFT Plot
Sample Rate	samp_rate/64
Baseband Freq	0
Y per Div	10 dB ▾
Y Divs	10
Ref Level (dB)	0
Ref Scale (p2p)	2.0
FFT Size	1024
Refresh Rate	15
Peak Hold	Off ▾
Average	Off ▾
Window	Automatic ▾
Window Size	
Grid Position	
Notebook	
Freq Set Varname	None

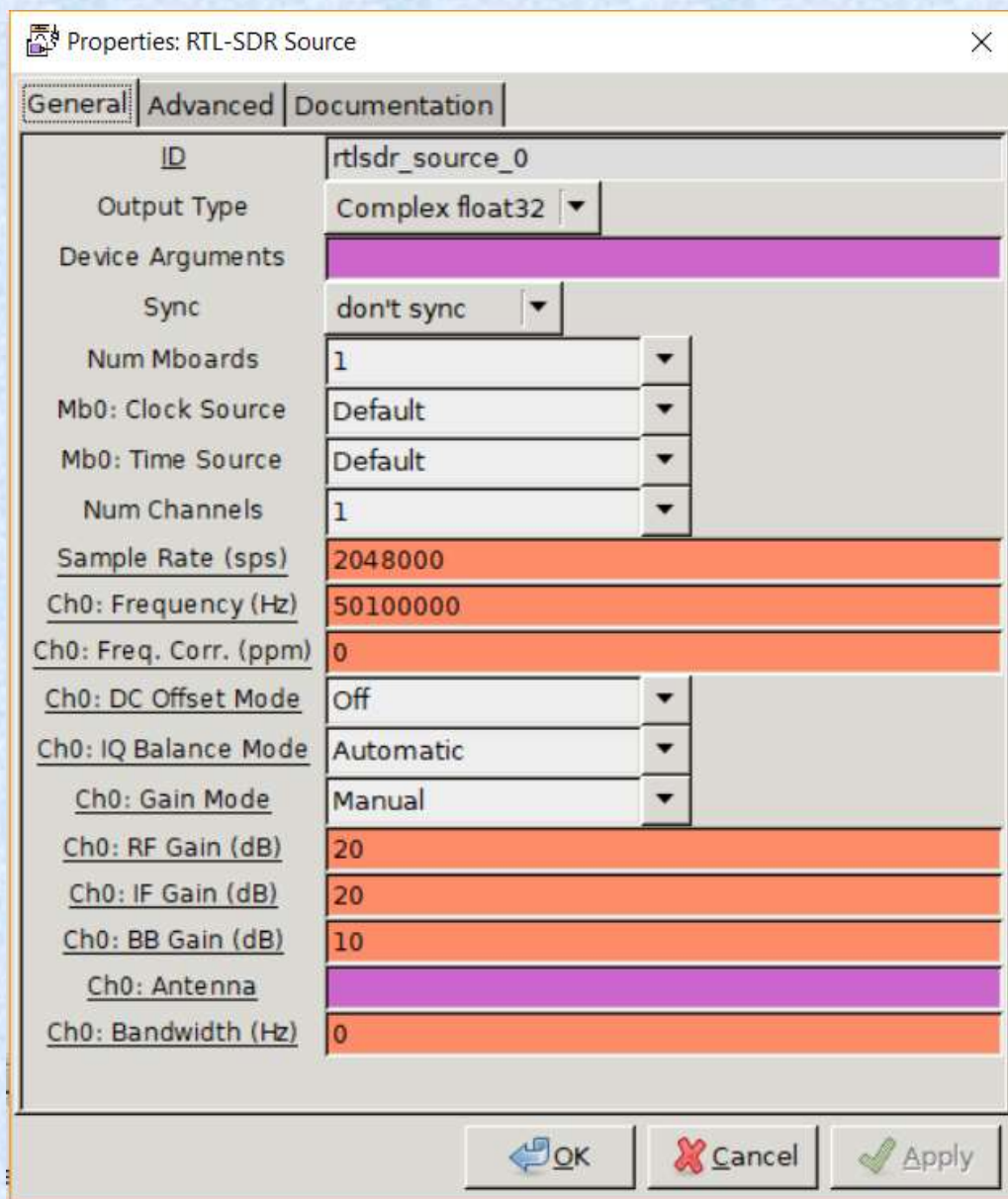
OK Cancel Apply

RTL-SDR Source

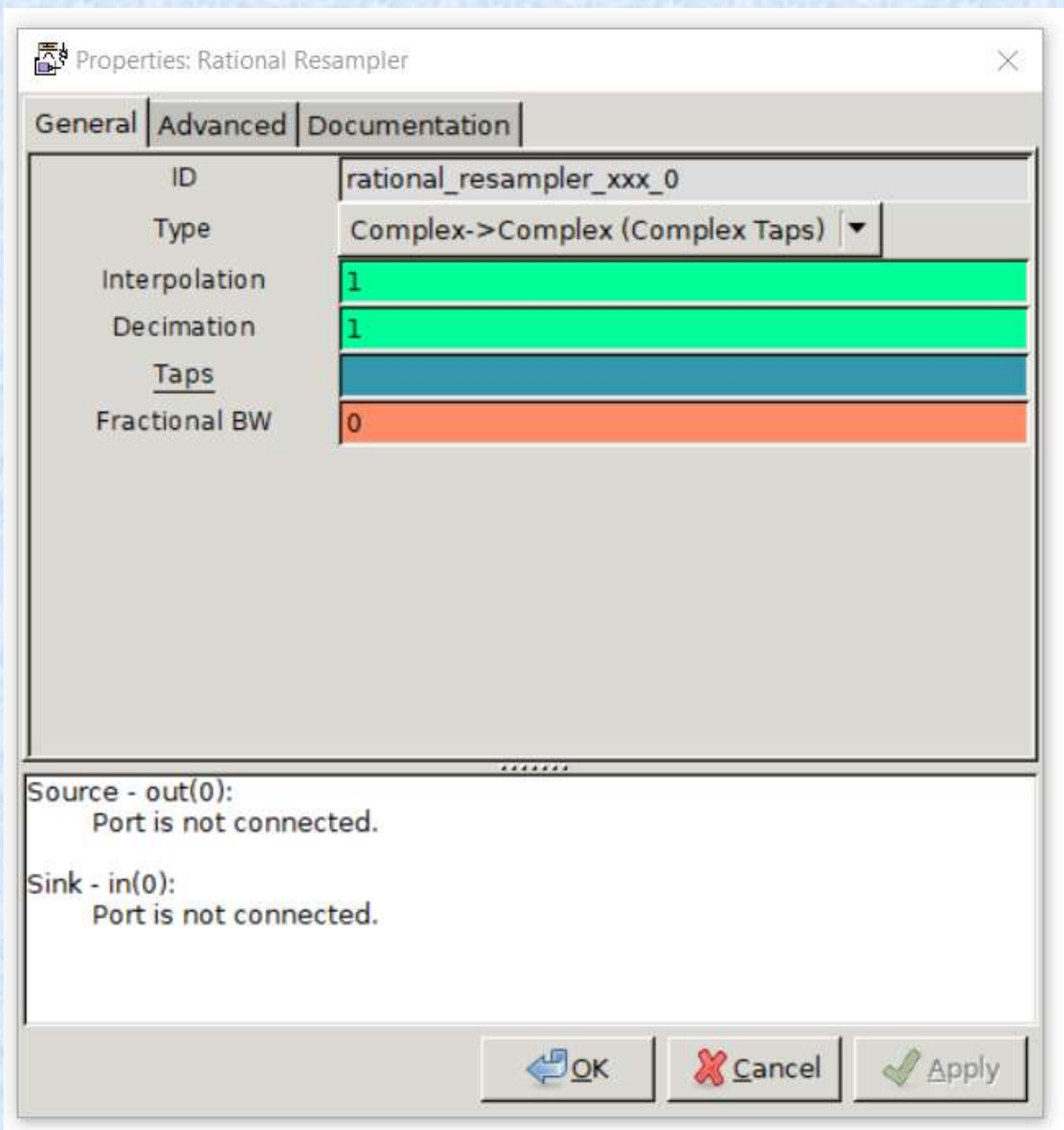
- Find this under:

```
[no module specified] / Sources /RTL-SDR Source
```

- Now we need to really study the documentation section To get a handle on what this is doing



RTL-SDR Source



Rational Resampler

Frequency Xlating FIR Filter

- Find this under:

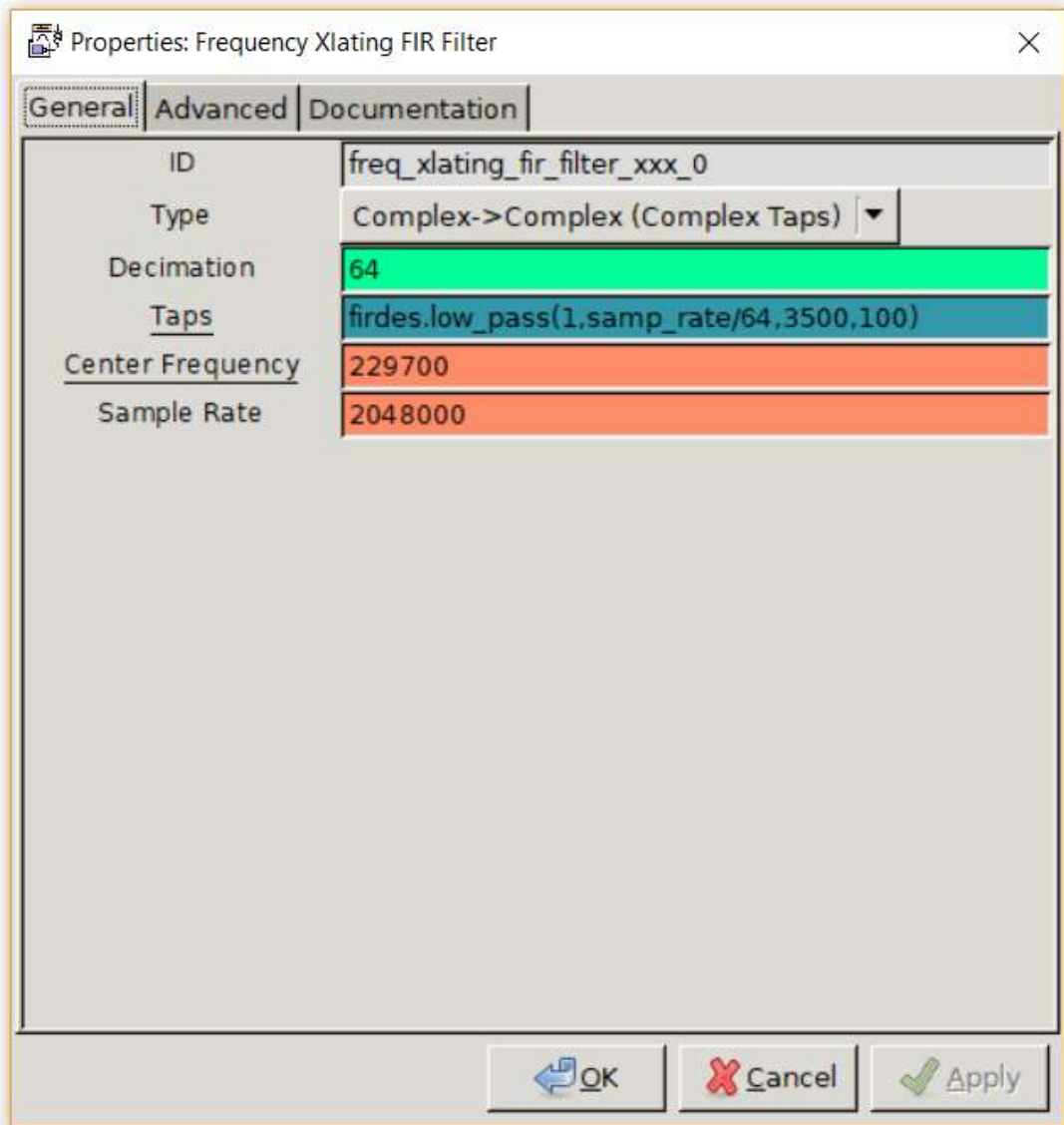
Core / Channelizers /

- This does 3 things:

- Frequency shifts an input signal
- Decimates it to a slower sample rate
- FIR Filters the signal

- The Filtering is done via a bit of python code

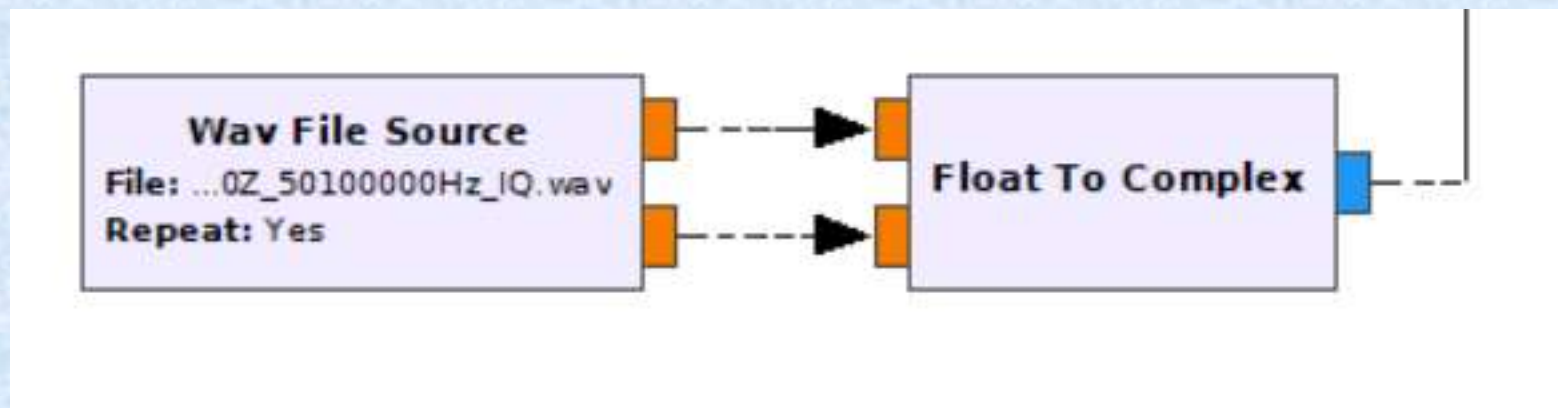
```
firdes.low_pass(gain,output_samp_rate,  
                filter_cut_off,transition_width)
```

Frequency Xlating FIR Filter

Wav File Source

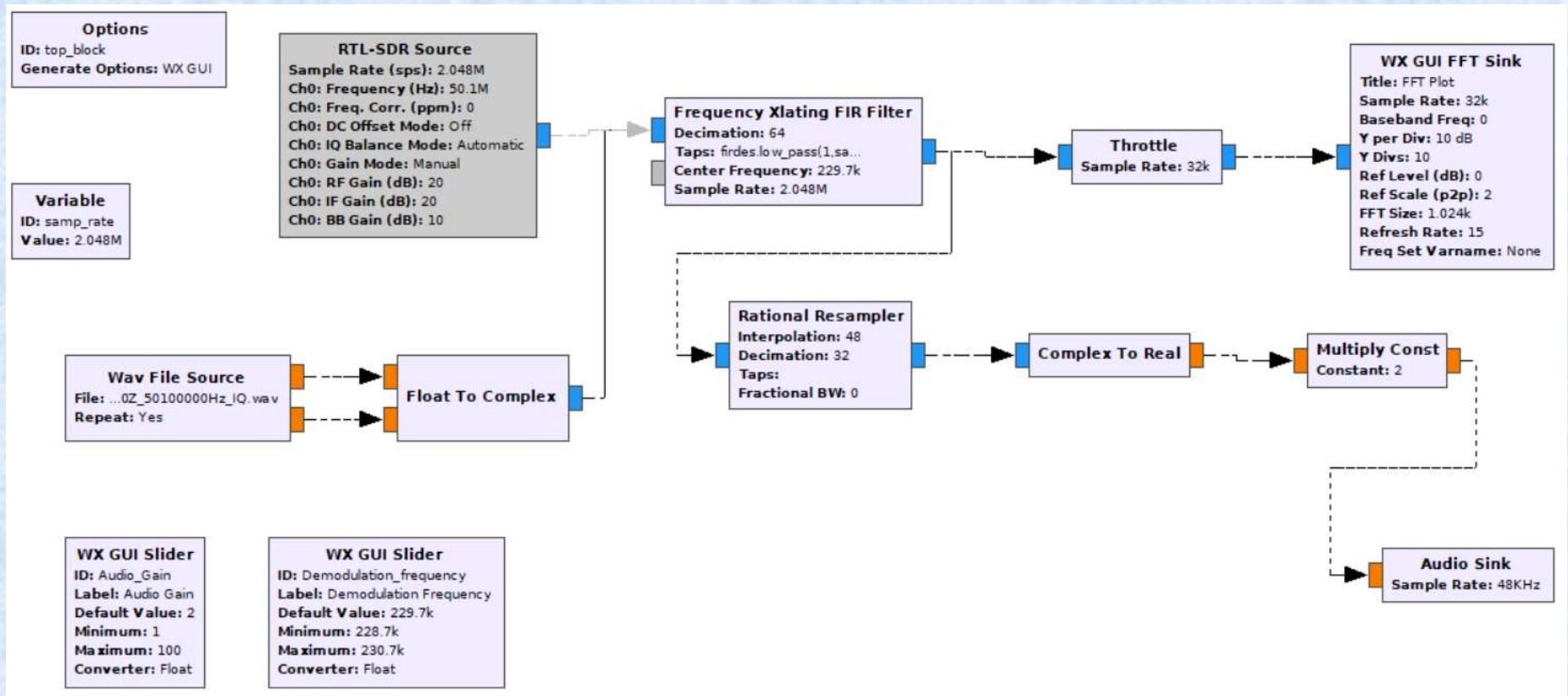
- If you use SDRSharp to save a data as a WAV file, this is a very useful block for testing.
- Note: you will probably need to update to 2 channels (I and Q) and combine them into a complex signal.



Exercise 4

- Build a “Messy” USB SSB Receiver for 50.330 MHz
- Hints:
 - **RTL-SDR Source:** Sample rate=2.048M, RF gain = 20dB, IF Gain = 20dB Frequency 50.1MHz
 - **Frequency Xlating FIR Filter:** Decimation set to give 32 kHz output, Centre Frequency set via a GUI slider between 228700 and 230700 Hz
 - **Rational Resampler** (if your sound card doesn't support 32KHz)
 - Convert 32KHz to 48KHz or something your sound card is happy with
 - **Complex To Real** will be needed
 - **Multiply Const:** for audio gain control, set via slider between 1.0 and 100.0 in 0.1 steps
 - It is a good idea to add a FFT Sink in as well to see what is present!

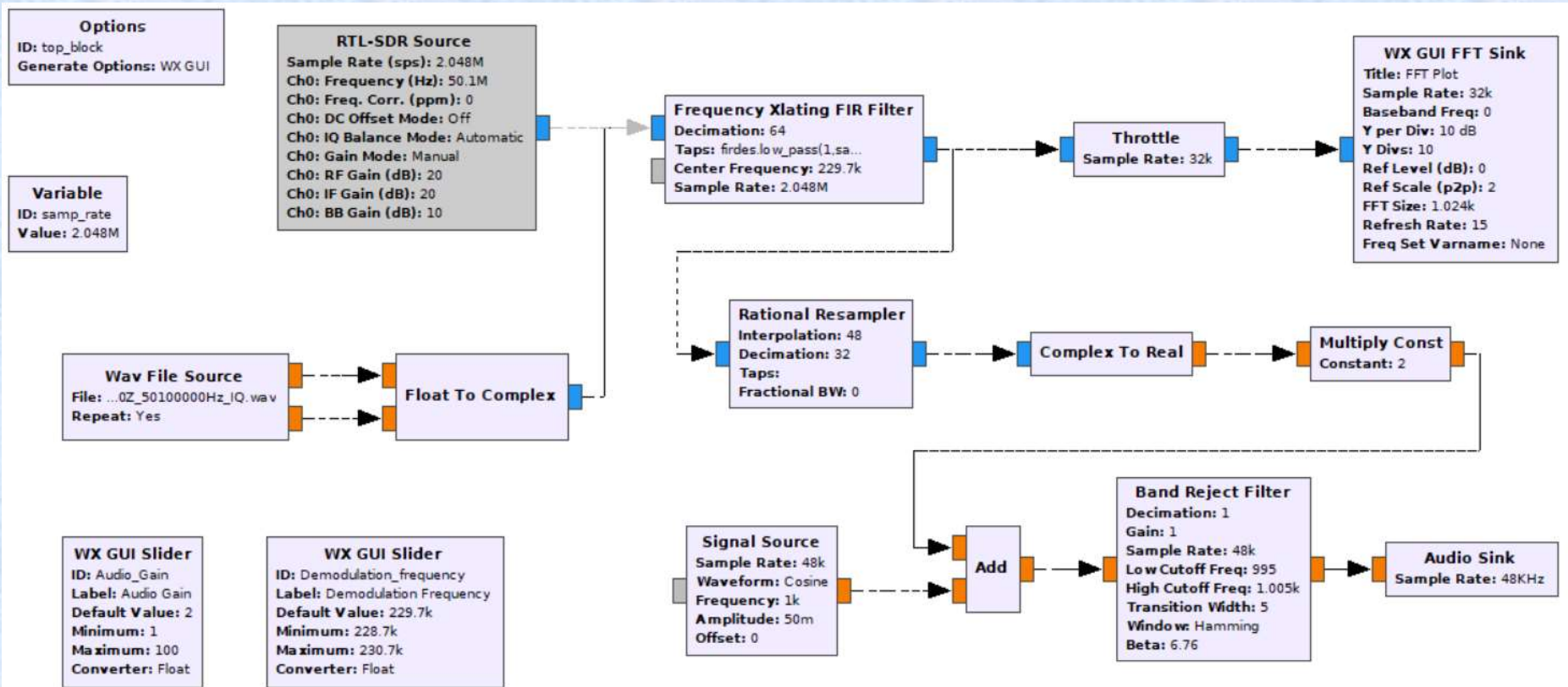
One Possible Result for Exercise 4



Exercise 5

- Add in an “interfering” source
 - just add a Signal Source with freq=1k, amplitude=0.05. Use an Add block to combine this with the output of the Multiply Const block you used to set the gain.
- Now add in a notch filter to get rid of it.
 - Hint: low cutoff=995, high cutoff=1005, transition width=5 will work!

One Possible Result for Exercise 5



The End of Building Our First Receiver