TP CELL
Travail sur GNU Radio

GNU Radio is a free and open source software development toolkit that provides signal processing blocks to implement software-defined radio systems. It can be used without hardware in a simulation-like environment, or with readily-available low-cost external RF hardware to create software-defined radios. GNU Radio supports several kinds of hardware radio front-ends. The most commonly used equipment are the USRP (Universal Software Radio Peripheral) devices by Ettus Research, LLC.. The USRP platform is designed for RF applications from DC to 6 GHz. Example application areas include white space radios, mobile phones, public safety radio, land mobiles, broadcast TV, FM radio, satellite navigation and amateur radio bands. In this lab, we use GNU Radio to simulate a simple radio system, where we generate a transmitting signal & receive a RF signal using a USRP B200 device.

GNU radio will be used in Ubuntu, a Linux based operating system. There are also a number of resources available online which can be utilized if there are any remaining questions.

Steps to download GnuRadio + UHD

- GnuRadio is already downloaded in your machines! This is for your personal research.

For most users, the recommended way of installation is using the build-gnuradio script. The following command first downloads the installer (build-gnuradio) and makes it executable. And then downloads and installs all dependencies, downloads both UHD and GNU Radio from Git (which means it will automatically install the latest version from the ‘master’ branch), runs the make process, and installs it on your system (a lot of this is done silently, so if there's a lot to do for the script, don't be surprised if it doesn't say anything for a while). This process may take 1-2 hours if you have a slower system and/or internet connection.

1) Open a terminal window.
2) Create a new folder where the software will be installed: *mkdir gnuradio*
3) Move to the directory: *cd gnuradio*
4) Run this command. The --verbose option at the end of this command allows you to see a percentage complete indicator.

   ```
   wget http://www.sbrac.org/files/build-gnuradio && chmod a+x ./build- gnuradio && ./build-gnuradio --verbose
   ```

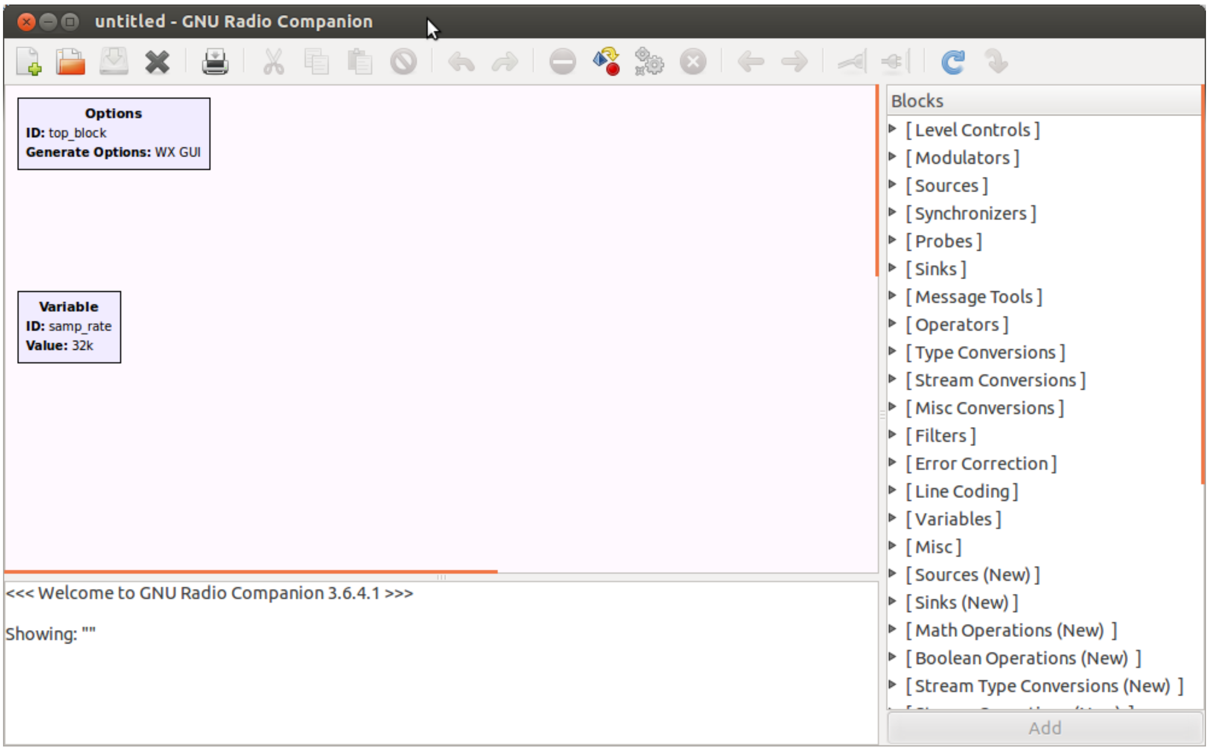
5) If you get error like 'the following packages have unmet dependencies: libboost-filesystem 1.48-dev'. Please install the missing package.

   ```
   sudo apt-get install libboost1.46-all-dev
   ```

6) Proceed? *Y*
7) Sudo privelidges? *Y*

Lab 1: Modeling and Simulation of a Waveform Signal Using GNU Radio Companion

1). Open a terminal window. At the prompt type: ***gnuradio-companion.*** An untitled GRC window similar to the one below should open.



Options
ID: top_block
Generate Options: WX GUI

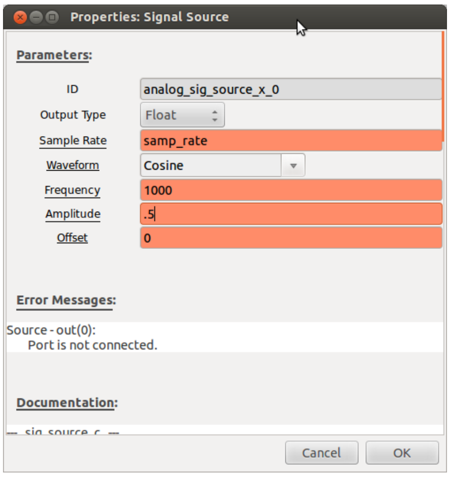
Variable
ID: samp_rate
Value: 32k

```
<<< Welcome to GNU Radio Companion 3.6.4.1 >>>

Showing: ""
```

2).Double click on the Options block. This block sets some general parameters for the flow graph. You can rename the ID and type in a project title, author and description. Leave the other settings as default. Another block that is present is the Variable block. It is used to set the sample rate.

3). On the right side of the window is a list of the blocks that are available. By expanding any of the categories you can see the blocks available. Explore each of the categories so that you have an idea of what is available.

4). Open the Sources category and double click on the Signal Source. Note that a Signal Source block will now appear in the main window. Double click on the block and the properties window will open. Adjust the settings to match those as shown in the figure below.



Properties: Signal Source

Parameters:

ID: analog_sig_source_x_0
Output Type: Float
Sample Rate: samp_rate
Waveform: Cosine
Frequency: 1000
Amplitude: .5
Offset: 0

Error Messages:

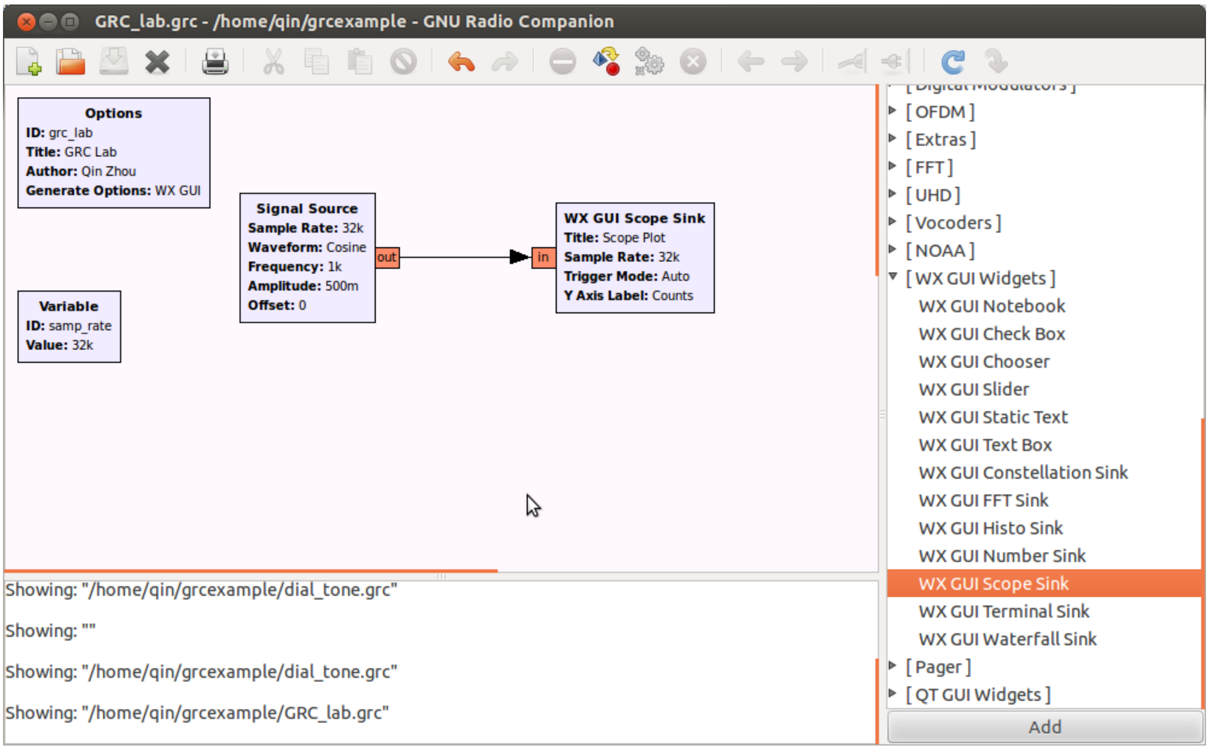
Source - out(0):
    Port is not connected.

Documentation:

— sig_source_c —

5). In order to view this wave we need one of the graphical sinks. Add a WX GUI Scope Sink to the flow graph. In the properties, change the Type to Float.

6). Connect these two blocks by clicking once on the "out" port of the Signal Source, and then once on the "in" port of the Scope Sink. The following flow graph should be displayed.



GRC_lab.grc - /home/qin/grcexample - GNU Radio Companion

Options
ID: grc_lab
Title: GRC Lab
Author: Qin Zhou
Generate Options: WX GUI

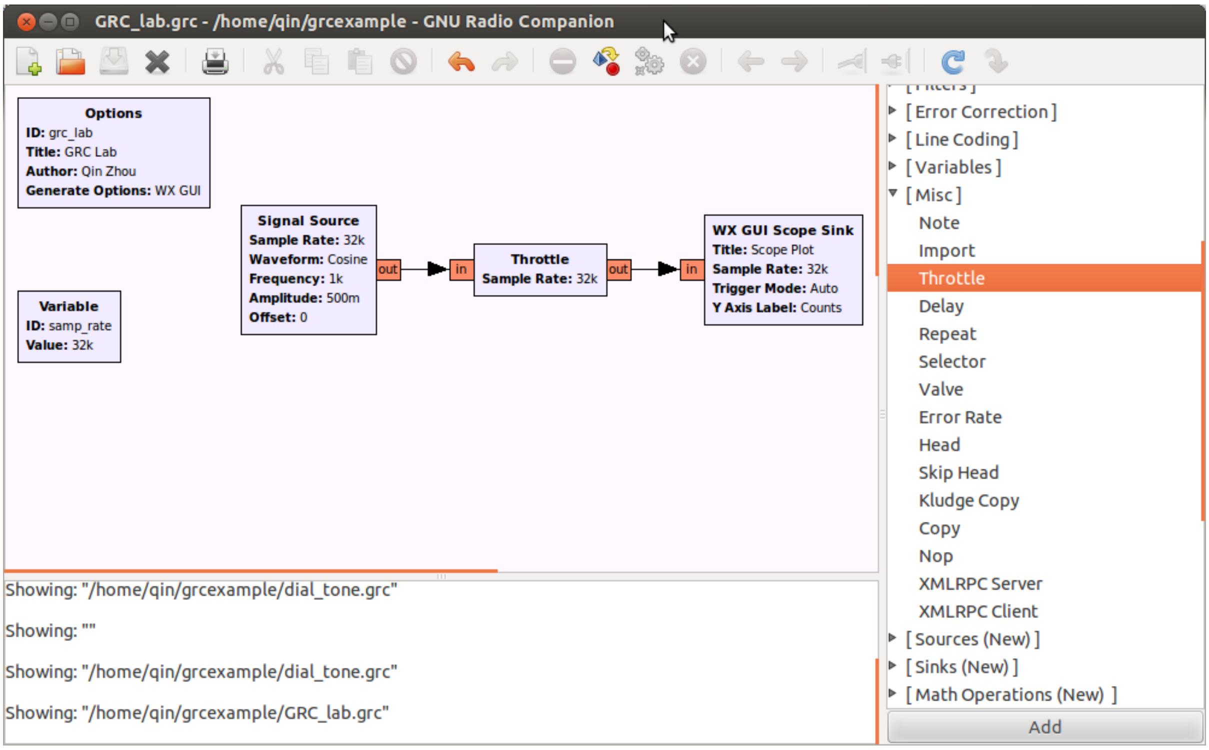
Variable
ID: samp_rate
Value: 32k

Signal Source
Sample Rate: 32k
Waveform: Cosine
Frequency: 1k
Amplitude: 500m
Offset: 0

WX GUI Scope Sink
Title: Scope Plot
Sample Rate: 32k
Trigger Mode: Auto
Y Axis Label: Counts

```
Showing: "/home/qin/grcexample/dial_tone.grc"

Showing: ""

Showing: "/home/qin/grcexample/dial_tone.grc"

Showing: "/home/qin/grcexample/GRC_lab.grc"
```

7). Now if you click on the “Generate the flow graph” icon, you may have noticed a warning in the message window indicating that the flow graph may not have flow control. Add a Throttle block between the source and the sink as shown in the figure below. Remember to change the Throttle Type to Float.



GRC_lab.grc - /home/qin/grcexample - GNU Radio Companion

Options
ID: grc_lab
Title: GRC Lab
Author: Qin Zhou
Generate Options: WX GUI

Variable
ID: samp_rate
Value: 32k

Signal Source
Sample Rate: 32k
Waveform: Cosine
Frequency: 1k
Amplitude: 500m
Offset: 0

Throttle
Sample Rate: 32k

WX GUI Scope Sink
Title: Scope Plot
Sample Rate: 32k
Trigger Mode: Auto
Y Axis Label: Counts

Showing: "/home/qin/grcexample/dial_tone.grc"

Showing: ""

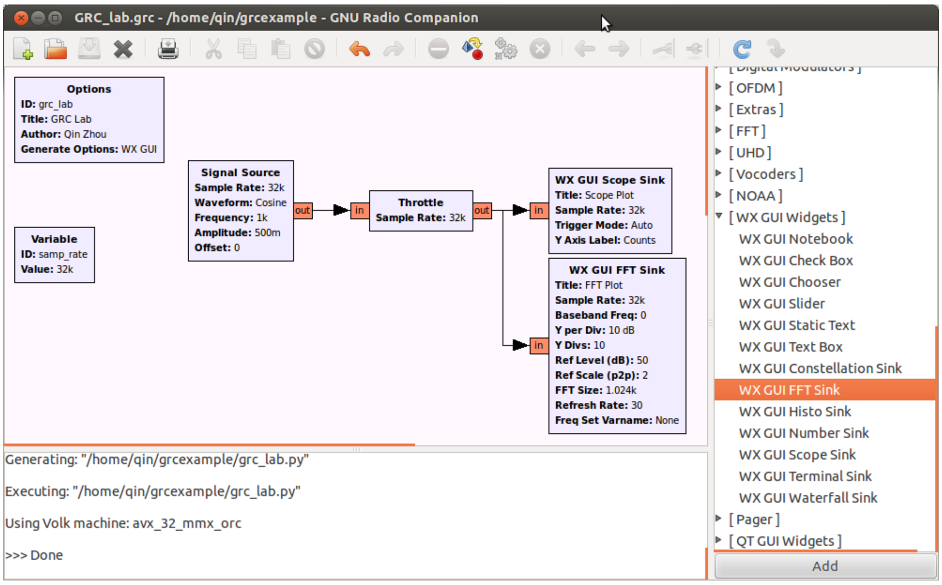
Showing: "/home/qin/grcexample/dial_tone.grc"

Showing: "/home/qin/grcexample/GRC_lab.grc"

8). Save the flow graph and click the “Execute the flow graph” icon. A scope plot should open displaying several cycles of the sinusoid. Confirm that the frequency and amplitude match the value that you expect. Change the Channel Options, Marker setting to Dot Large. Now you can see the actual sample values. Recall that the Variable block set the sampling rate to 32000 samples/second. Note that there are in fact 32 samples within one cycle of the wave.

9). Close the Scope Plot and reduce the sample rate to 10000. Re-generate and execute the flow graph. Note that there are now fewer points per cycle. How low can you drop the sample rate? Experiment with this and see how the output changes as you drop the sample rate.

10). Now change the sample rate back to 32000. Add a WX GUI FFT Sink to the main window to create the flow graph shown below. Change the Type to Float and leave the remaining parameters at their default values. Execute the flow graph. You should observe the scope as before along with an FFT plot correctly showing the frequency of the input at 1 KHz.



GRC_lab.grc - /home/qin/grcexample - GNU Radio Companion

Options
ID: grc_lab
Title: GRC Lab
Author: Qin Zhou
Generate Options: WX GUI

Variable
ID: samp_rate
Value: 32k

Signal Source
Sample Rate: 32k
Waveform: Cosine
Frequency: 1k
Amplitude: 500m
Offset: 0

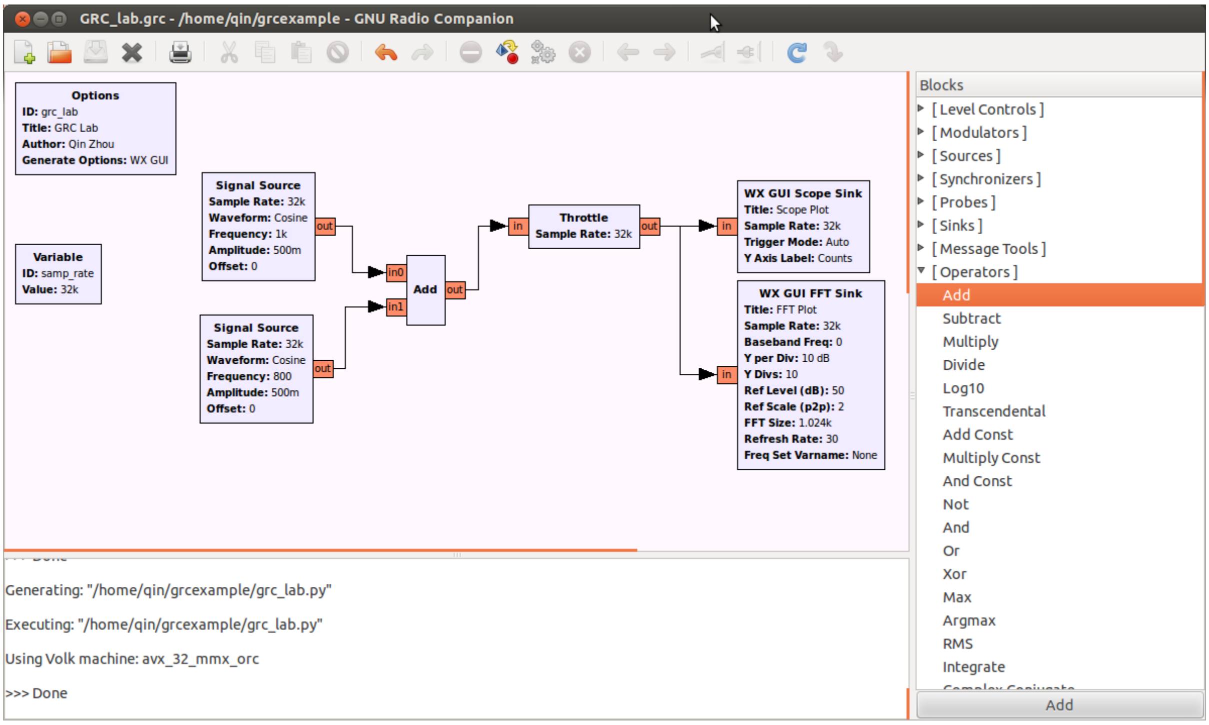
Throttle
Sample Rate: 32k

WX GUI Scope Sink
Title: Scope Plot
Sample Rate: 32k
Trigger Mode: Auto
Y Axis Label: Counts

WX GUI FFT Sink
Title: FFT Plot
Sample Rate: 32k
Baseband Freq: 0
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 50
Ref Scale (p2p): 2
FFT Size: 1.024k
Refresh Rate: 30
Freq Set Varname: None

```
Generating: "/home/qin/grcexample/grc_lab.py"
Executing: "/home/qin/grcexample/grc_lab.py"
Using Volk machine: avx_32_mmx_orc
>>> Done
```

11). Construct the flow graph shown below. Set the sample rate to 32000. The two Signal Sources should have frequencies of 1000 and 800, respectively.



GRC_lab.grc - /home/qin/grcexample - GNU Radio Companion

Options
ID: grc_lab
Title: GRC Lab
Author: Qin Zhou
Generate Options: WX GUI

Variable
ID: samp_rate
Value: 32k

Signal Source
Sample Rate: 32k
Waveform: Cosine
Frequency: 1k
Amplitude: 500m
Offset: 0

Signal Source
Sample Rate: 32k
Waveform: Cosine
Frequency: 800
Amplitude: 500m
Offset: 0

Add

Throttle
Sample Rate: 32k

WX GUI Scope Sink
Title: Scope Plot
Sample Rate: 32k
Trigger Mode: Auto
Y Axis Label: Counts

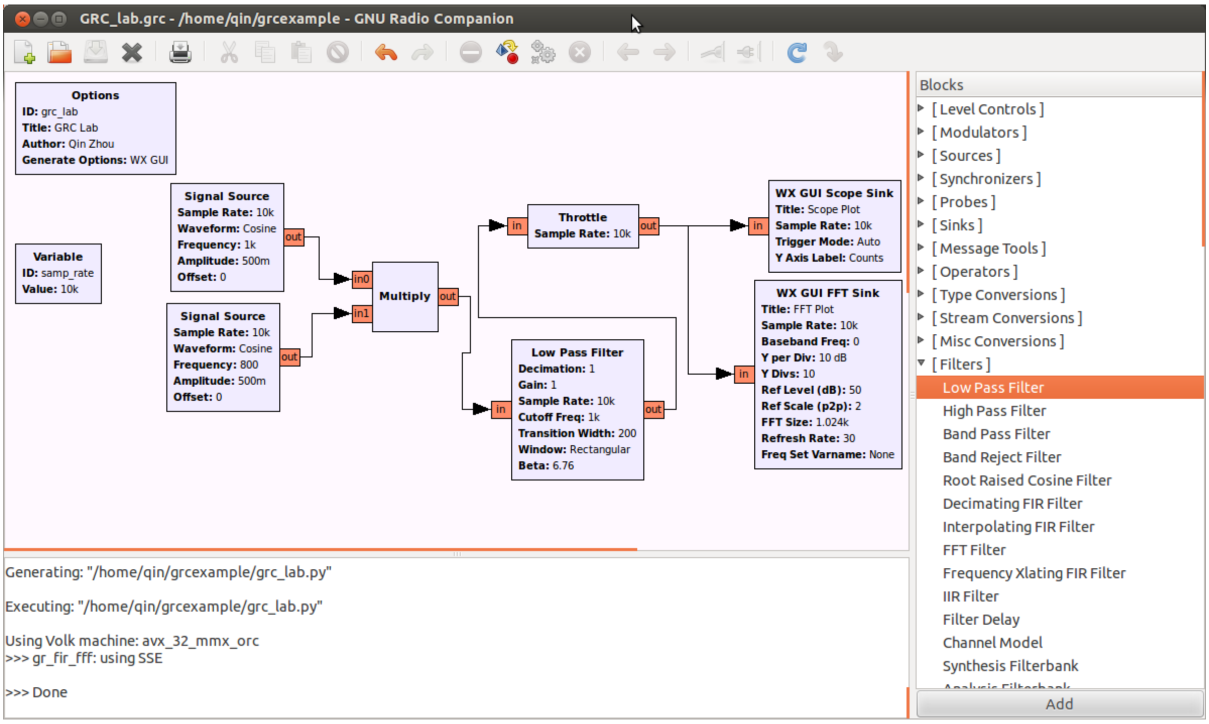
WX GUI FFT Sink
Title: FFT Plot
Sample Rate: 32k
Baseband Freq: 0
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 50
Ref Scale (p2p): 2
FFT Size: 1.024k
Refresh Rate: 30
Freq Set Varname: None

```
Generating: "/home/qin/grcexample/grc_lab.py"
Executing: "/home/qin/grcexample/grc_lab.py"
Using Volk machine: avx_32_mmx_orc
>>> Done
```

12). Generate and execute the flow graph. Observe the result on the Scope plot as well as the FFT plot. In order to obtain better resolution of the FFT plot, we can lower the sample rate. Try lowering the sample rate to 10 KHz.

13). Replace the Add block with a Multiply block. What output do you expect from the product of two sinusoids? Confirm your result on the Scope and FFT displays.

14). Modify the flow graph to include a Low Pass Filter block as shown in the figure below. Set the low pass filter to have a cutoff frequency of 1 KHz and a transition width of 200 Hz. Use a Rectangular Window. Execute the flow graph. What result do you observe? Experiment with the High Pass Filter. What do you observe?



Options
ID: grc_lab
Title: GRC Lab
Author: Qin Zhou
Generate Options: WX GUI

Variable
ID: samp_rate
Value: 10k

Signal Source
Sample Rate: 10k
Waveform: Cosine
Frequency: 1k
Amplitude: 500m
Offset: 0

Signal Source
Sample Rate: 10k
Waveform: Cosine
Frequency: 800
Amplitude: 500m
Offset: 0

Multiply

Throttle
Sample Rate: 10k

Low Pass Filter
Decimation: 1
Gain: 1
Sample Rate: 10k
Cutoff Freq: 1k
Transition Width: 200
Window: Rectangular
Beta: 6.76

WX GUI Scope Sink
Title: Scope Plot
Sample Rate: 10k
Trigger Mode: Auto
Y Axis Label: Counts

WX GUI FFT Sink
Title: FFT Plot
Sample Rate: 10k
Baseband Freq: 0
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 50
Ref Scale (p2p): 2
FFT Size: 1.024k
Refresh Rate: 30
Freq Set Varname: None

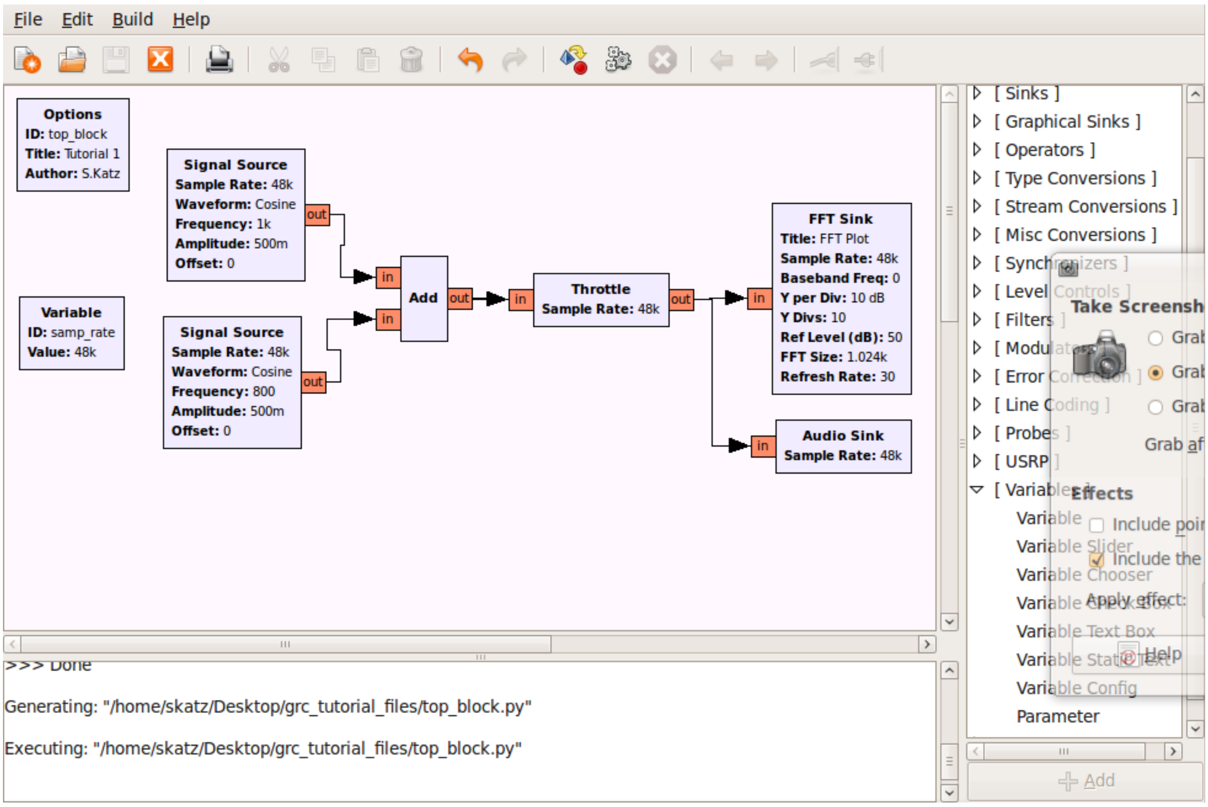
```
Generating: "/home/qin/grcexample/grc_lab.py"

Executing: "/home/qin/grcexample/grc_lab.py"

Using Volk machine: avx_32_mmx_orc
>>> gr_fir_fff: using SSE

>>> Done
```

15). Using the same flow graph, change the sample rate variable to 20000. Change the Decimation in the Low Pass Filter to 2. Decimation decreases the number of samples that are processed. A decimation factor of two means that the output of the filter will have a sample rate equal to one-half of the input sample rate, or in this case only 10000 samples/sec. This is a sufficient sample rate for the frequencies that we are dealing with. Execute the flow graph. What frequency do you observe on the FFT plot? Why is this error occurring? Double click on the FFT Sink block and change the sample rate to samp_rate/2. How is the result now?

16). Open a file browser in Ubuntu. Go to the directory where you have saved the GRC file. In addition to a “.grc” file, note that there is also a “.py” file sharing the same file name. Double click on this block. You will be given the option to Run or Display this file. Select Display. This is the Python file that is generated by GRC. It is this file being run when you execute the flow graph. You can modify this file and run it from the terminal window.

Lab 2: Sliders

1). Construct the flow graph shown below. Note that the sample rate is set to 48000 in this example.



Options
ID: top_block
Title: Tutorial 1
Author: S.Katz

Variable
ID: samp_rate
Value: 48k

Signal Source
Sample Rate: 48k
Waveform: Cosine
Frequency: 1k
Amplitude: 500m
Offset: 0

Signal Source
Sample Rate: 48k
Waveform: Cosine
Frequency: 800
Amplitude: 500m
Offset: 0

Add

Throttle
Sample Rate: 48k

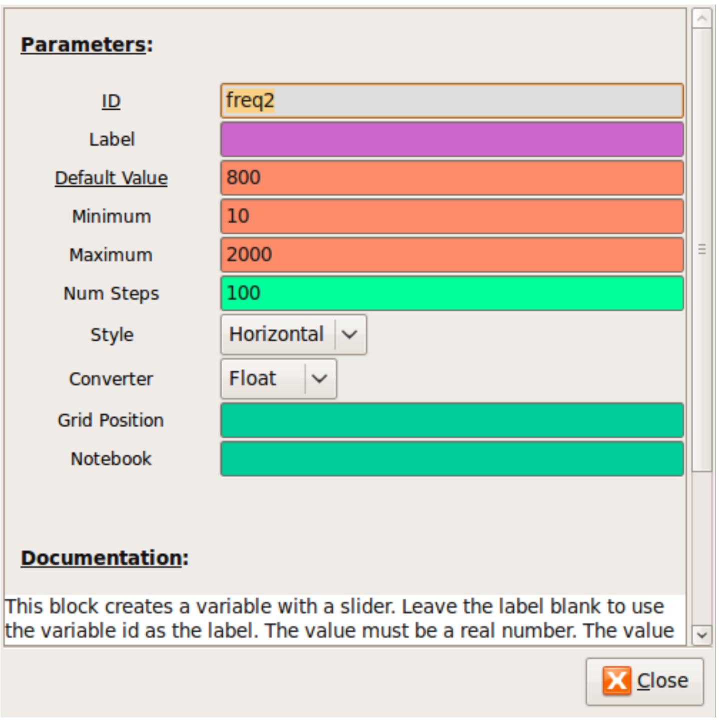
FFT Sink
Title: FFT Plot
Sample Rate: 48k
Baseband Freq: 0
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 50
FFT Size: 1.024k
Refresh Rate: 30

Audio Sink
Sample Rate: 48k

```
>>> Done

Generating: "/home/skatz/Desktop/grc_tutorial_files/top_block.py"

Executing: "/home/skatz/Desktop/grc_tutorial_files/top_block.py"
```

2). Execute the flow graph. You should hear the composite tone and see the FFT sink display of the spectrum. Experiment with the FFT size in the FFT sink. It must always be a multiple of two. Note that as you increase the FFT size the resolution of the display increases. Reset the FFT size to 1024 when you are done.

3). Add a Variable Slider (from the Variables category) to the flow graph. Double click on the block and set the parameters as shown below.



Parameters:

ID	freq2
Label	
Default Value	800
Minimum	10
Maximum	2000
Num Steps	100
Style	Horizontal
Converter	Float
Grid Position	
Notebook	

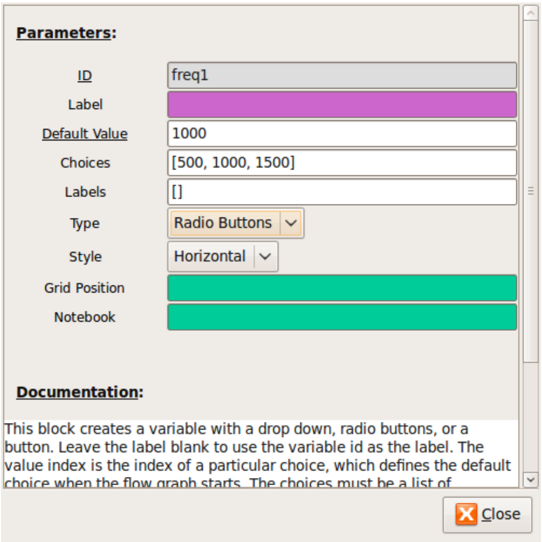
Documentation:

This block creates a variable with a slider. Leave the label blank to use the variable id as the label. The value must be a real number. The value

4). Execute the flow graph. This time when the FFT sink display opens you will see a horizontal slider at the top. Move the slider back and forth to change the value of freq2 between 10 and 2000. You should observe that this does not change the spectrum or the sound. This is because freq2 has not been assigned to control anything.

5). Double click on the bottom Signal Source (set to 800). Replace the frequency (800) with the variable freq2. Execute the flow graph. You should now observe that both the spectrum and the sound change as you vary the frequency of the Signal Source using the slider.

6). Add a Variable Text Box to the flow graph. Set the parameters as shown in the figure 1.



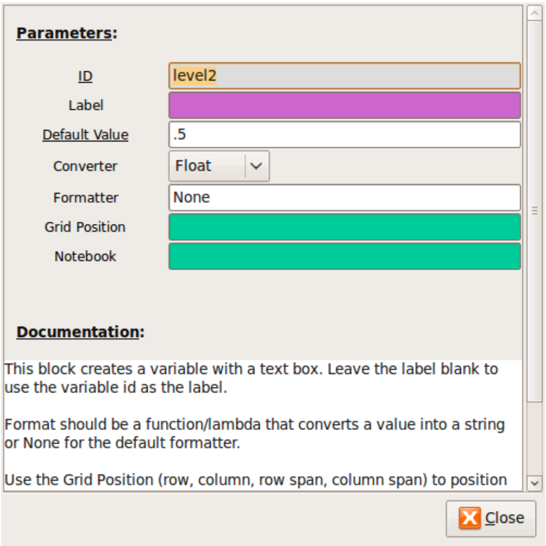
Parameters:

ID: freq1
Label:
Default Value: 1000
Choices: [500, 1000, 1500]
Labels: []
Type: Radio Buttons
Style: Horizontal
Grid Position:
Notebook:

Documentation:

This block creates a variable with a drop down, radio buttons, or a button. Leave the label blank to use the variable id as the label. The value index is the index of a particular choice, which defines the default choice when the flow graph starts. The choices must be a list of

fig 1



Parameters:

ID: level2
Label:
Default Value: .5
Converter: Float
Formatter: None
Grid Position:
Notebook:

Documentation:

This block creates a variable with a text box. Leave the label blank to use the variable id as the label.

Format should be a function/lambda that converts a value into a string or None for the default formatter.

Use the Grid Position (row, column, row span, column span) to position

fig2

7). Double click on the bottom Signal Source and replace the Amplitude (.5) with the variable, level2. Execute the flow graph. Note that a text box will not appear in the display. The default value of 0.5 will be entered. Change the value to 0.1 followed by Enter. The volume of the 800 Hz tone will decrease and this will be reflected on the spectrum plot. Do not vary the level above one.

8). Add a Variable Chooser block to the flow graph. This block will add either a drop down menu, radio buttons or a button. Input the parameters as shown in the figure 2.

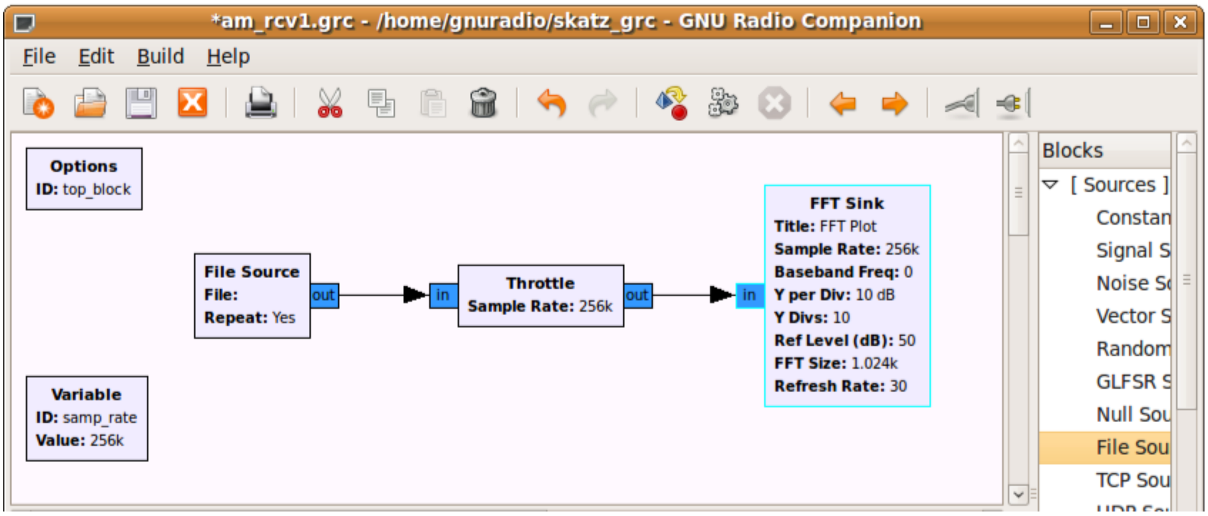
9). Change the Frequency of the top Signal Source (1KHz) to freq1. Execute the flow graph and change the frequency of the top Signal Source using the radio buttons. Experiment with the Drop Down menu and the Button to see how they work.

Lab 3: Receiving AM signals

In this lab, we are going to work with a data file containing several seconds of recorded signals from the AM broadcast band. This data file was obtained from the USRP B200. The sampling rate was 256k/s.

1). Download the file: am_usrp710.dat from http://www.csun.edu/~skatz/katzpage/sdr_project/sdr/am_usrp710.dat.zip .

2). Construct the flow graph shown below consisting of a File Sink, Throttle, and FFT Sink. Set the Sample Rate in the variable block to 256000. This is the rate at which the saved data was sampled.



Options
ID: top_block

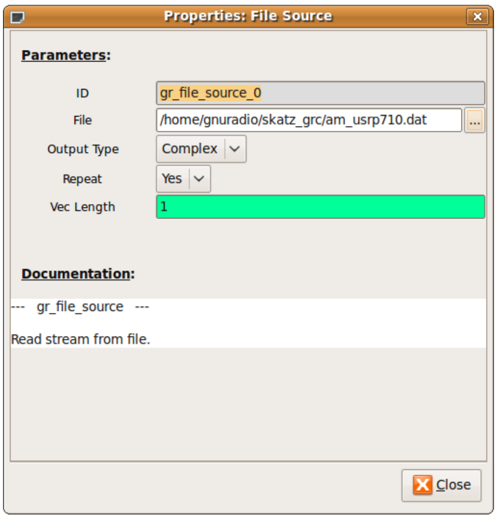
File Source
File:
Repeat: Yes

Throttle
Sample Rate: 256k

FFT Sink
Title: FFT Plot
Sample Rate: 256k
Baseband Freq: 0
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 50
FFT Size: 1.024k
Refresh Rate: 30

Variable
ID: samp_rate
Value: 256k

3). Double click on the File Source block. Click on the ellipsis (...) next to the File parameter. Locate the am_usrp710.dat file that you saved in step 1 (unzipped). The path to your file will appear as shown in the figure below. Set the Output Type to Complex. The use of complex data to describe and process waveforms in SDR will be discussed next week. Set Repeat to Yes. This will cause the data to repeat so that you have a continuously playing signal.



Parameters:

ID: gr_file_source_0

File: /home/gnuradio/skatz_grc/am_usrp710.dat

Output Type: Complex

Repeat: Yes

Vec Length: 1

Documentation:

--- gr_file_source ---

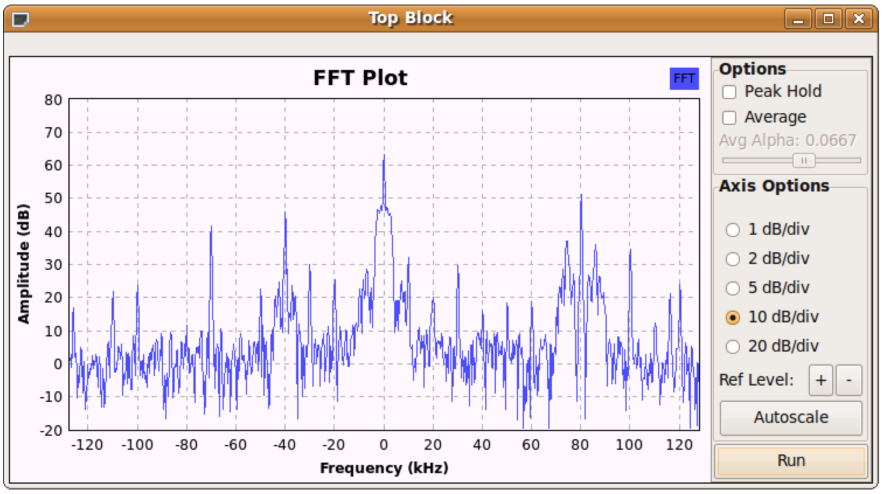
Read stream from file.

4). Save and execute the flow graph. You should observe an FFT display similar to the one shown below. You may need to click on Autoscale button to scale the data as shown. Note the following:

- This data was recorded with a USRP set to 710KHz. Thus, the signal you see at the center (indicated as 0 KHz) is actually at 710 KHz. Similarly, the signal at 80 KHz is actually at 710KHz + 80KHz = 790KHz.

- The display spans a frequency range from just below -120KHz to just above 120KHz. This exact span is 256KHz, which corresponds to the Sample Rate that the data was recorded at.

- The peaks that you observe on this display correspond to the carriers for AM broadcast signals. You should also be able to observe the sidebands for the stronger waveforms.



Options

□ Peak Hold

□ Average

Avg Alpha: 0.0667

Axis Options

○ 1 dB/div

○ 2 dB/div

○ 5 dB/div

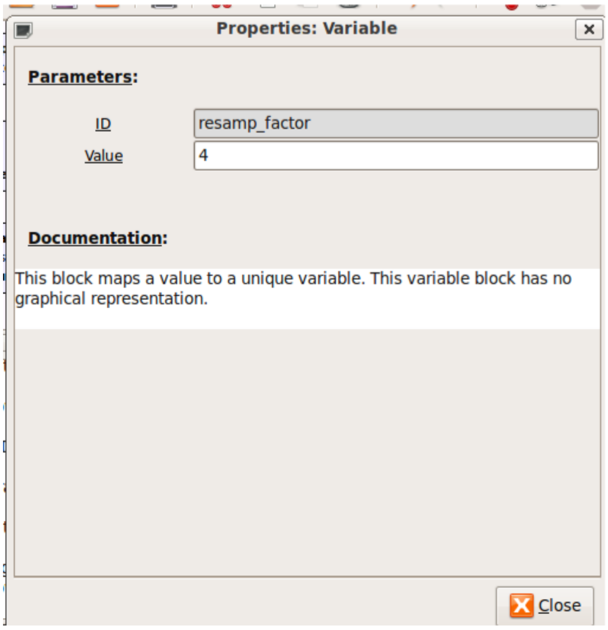
● 10 dB/div

○ 20 dB/div

Ref Level: + -

5). In this step we will expand the frequency scale on the FFT display so that you can view the signals with greater resolution. Recall from step 4 above that the span of the frequency axis is determined by the sample rate (256K for this file). While we cannot change the original data, we can resample it to either increase or decrease the sample rate. We will decrease the sample rate by using decimation. Modify the flow graph as follows:

- Add a Variable block (under Variables menu). Set the ID to *resamp_factor* and the Value to 4 as shown below.



Properties: Variable

Parameters:

ID: resamp_factor

Value: 4

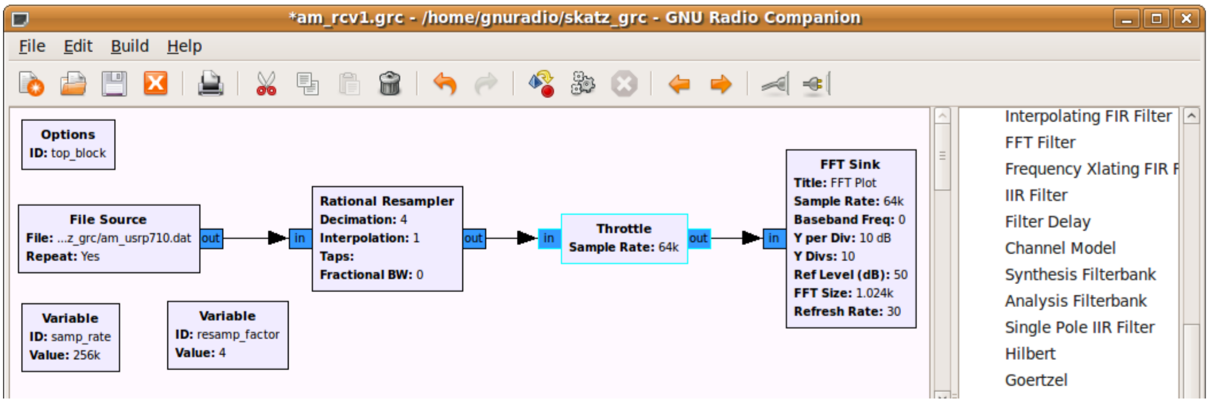
Documentation:

This block maps a value to a unique variable. This variable block has no graphical representation.

- Add the Rational Resampler (NOT Rational Resampler Base) from the Filters menu as shown below. Set its decimation factor to resamp_factor. Thus, it will use the value of the variable set in the previous step (3) to decimate the incoming data. That means that it will divide the incoming data rate by the decimation factor. In this example, the incoming 256K samp/sec data will be converted down to 256K/4 = 64K samp/sec.

- Note that the Throttle and FFT Sink now need their Sample Rates changed to correspond to this new rate. Change the Sample Rate in both of these blocks to: samp_rate/resamp_factor . Now we can change the decimation factor in the Variable block and it will be reflected in each of the other blocks automatically.

- Your flow graph should now appear as shown below.



Options
ID: top_block

File Source
File: ...z_grc/am_usrp710.dat
Repeat: Yes

Variable
ID: samp_rate
Value: 256k

Variable
ID: resamp_factor
Value: 4

Rational Resampler
Decimation: 4
Interpolation: 1
Taps:
Fractional BW: 0

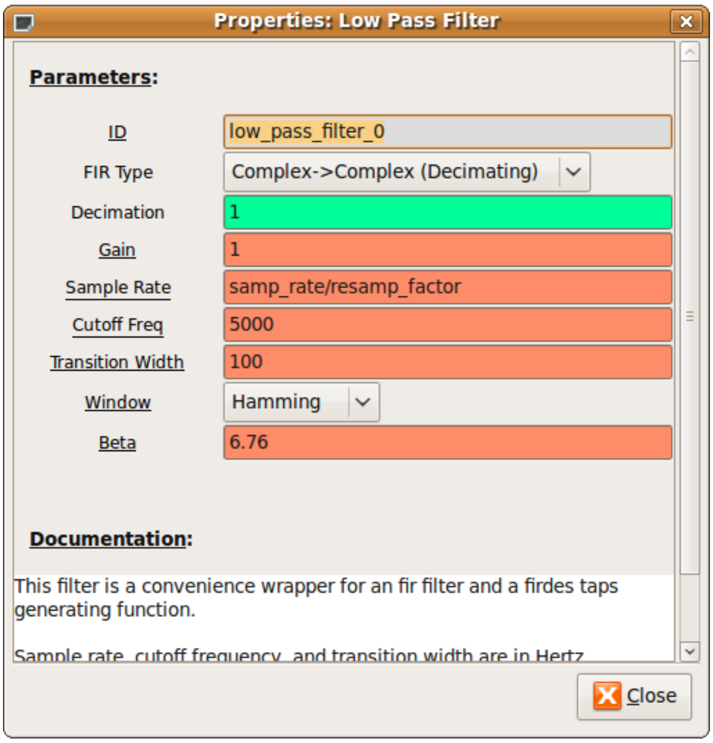
Throttle
Sample Rate: 64k

FFT Sink
Title: FFT Plot
Sample Rate: 64k
Baseband Freq: 0
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 50
FFT Size: 1.024k
Refresh Rate: 30

6). Execute the new flow graph. You should now observe a frequency span of only 64KHz (-32KHz to +32KHz). What actual frequency range does this correspond to?

7). The bandwidth of an AM broadcast signal is 10KHz (+/- 5KHz from the carrier frequency). You may find it useful to click the “Stop” button on the FFT plot to see this more clearly. Also, note that many stations also include additional information outside of the 10KHz bandwidth. This will be discussed later.

8). In order to select the station at 710 KHz (or 0KHz) we need to insert a filter to eliminate all but the one station that we want to receive. This is often referred to as a channel filter. Since the station at 710 KHz has been moved to 0 KHz (in the USRP) we will use a low pass filter. The station bandwidth is 10 KHz, so we will use a low pass filter that cuts off at 5 KHz. Insert the Low Pass Filter (from the Filters menu) between the Rational Resampler and the Throttle. Set the parameters as shown below.



Parameters:

ID	low_pass_filter_0
FIR Type	Complex->Complex (Decimating)
Decimation	1
Gain	1
Sample Rate	samp_rate/resamp_factor
Cutoff Freq	5000
Transition Width	100
Window	Hamming
Beta	6.76

Documentation:

This filter is a convenience wrapper for an fir filter and a firdes taps generating function.

Sample rate, cutoff frequency, and transition width are in Hertz

9). Execute the flow graph. You should see that only the station between +/- 5KHz remains.

10). The next step is to demodulate the signal. In the case of AM, the baseband signal is the
envelope or the magnitude of the modulated waveform. GNU Radio contains a Complex to Mag block (in the Type Conversions menu) that can be used for this purpose. Again, the use of complex signal representation will be dealt with in depth in the future. Insert this block between the Low Pass Filter and the Throttle.

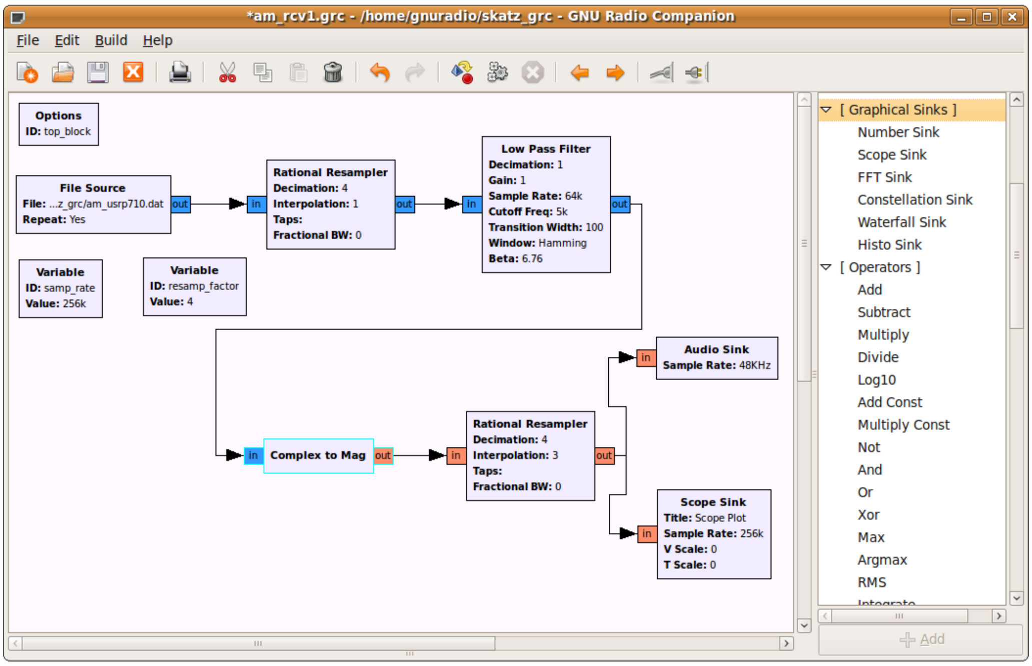
11). Note that the titles of some of the blocks are now red and the Execute Flow Graph icon is dimmed. This indicates an error. Prior to adding this block, all of the block inputs and outputs were complex values. However, the output of the Complex to Mag block is Float (a real number). Thus, any blocks following this block should be Type: Float. Modify the Throttle and FFT Sink accordingly.

12). Execute the flow graph. You should now observe the spectrum of the baseband signal in the FFT Sink. Note that since the input data type to the FFT Sink is Float, only the positive frequency spectrum is displayed.

13). The next step is to listen to this demodulated waveform to confirm that it is in fact receiving the baseband signal. Remove the Throttle and the FFT Blocks. Add an Audio Sink block to the output of the Complex to Mag block. Note that the default sample rate for the Audio Sink is 32KHz. Most current audio cards require a sample rate of at least 44.1KHz. Change the sample rate to 48 KHz.

14). Note that the sample rate out of the Complex to Mag block is 64K and the input to the Audio Sink is 48K. In order to convert 64K to 48K we need to divide (decimate) by 4 and multiply (interpolate) by 3. Insert a Rational Resampler between the Complex to Mag and Audio Sink blocks and set the decimation and interpolation as noted above. Also set its Type to Float->Float.

15). Place a Scope Sink at the output of the Rational Resampler (in addition to the Audio Sink). Change its Type to Float. The flow graph should be similar to the one shown below.



Options
ID: top_block

File Source
File: ...z_grc/am_usrp710.dat
Repeat: Yes

Rational Resampler
Decimation: 4
Interpolation: 1
Taps:
Fractional BW: 0

Low Pass Filter
Decimation: 1
Gain: 1
Sample Rate: 64k
Cutoff Freq: 5k
Transition Width: 100
Window: Hamming
Beta: 6.76

Variable
ID: samp_rate
Value: 256k

Variable
ID: resamp_factor
Value: 4

Audio Sink
Sample Rate: 48KHz

Complex to Mag

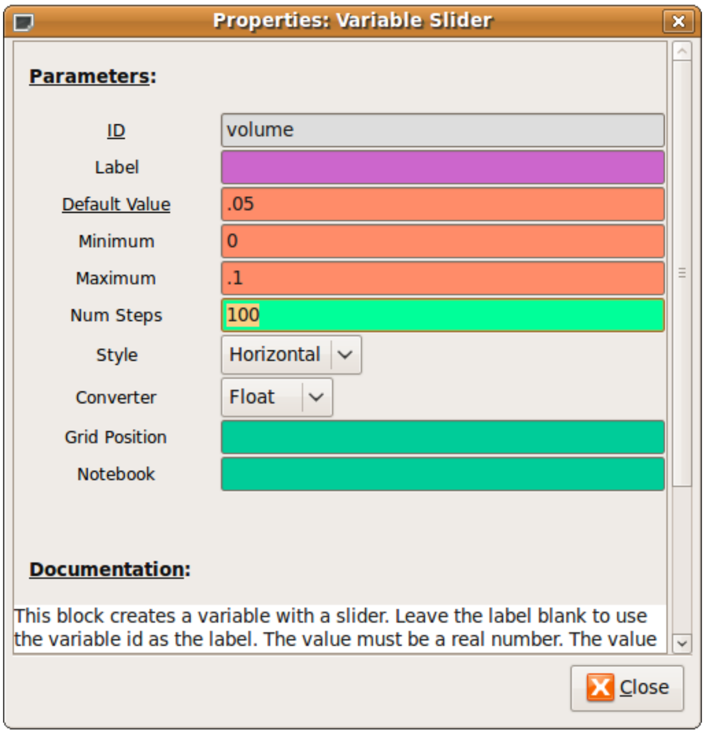
Rational Resampler
Decimation: 4
Interpolation: 3
Taps:
Fractional BW: 0

Scope Sink
Title: Scope Plot
Sample Rate: 256k
V Scale: 0
T Scale: 0

16). Execute the flow graph. The Scope Sink should open and display the output waveform. However, you may not yet hear the audio from your speaker. This is due to the fact that the values of the samples going in to the Audio Sink block are too large for the speaker. We need to apply an attenuator to reduce the size of the samples.

17). Insert a Multiply Const block from the Operators category between the Complex to Mag block and the Rational Resampler. Set the IO Type of the block to Float.

18). Add a Variable Slider block. Set the parameters as shown below.



Parameters:

ID volume

Label

Default Value .05

Minimum 0

Maximum .1

Num Steps 100

Style Horizontal

Converter Float

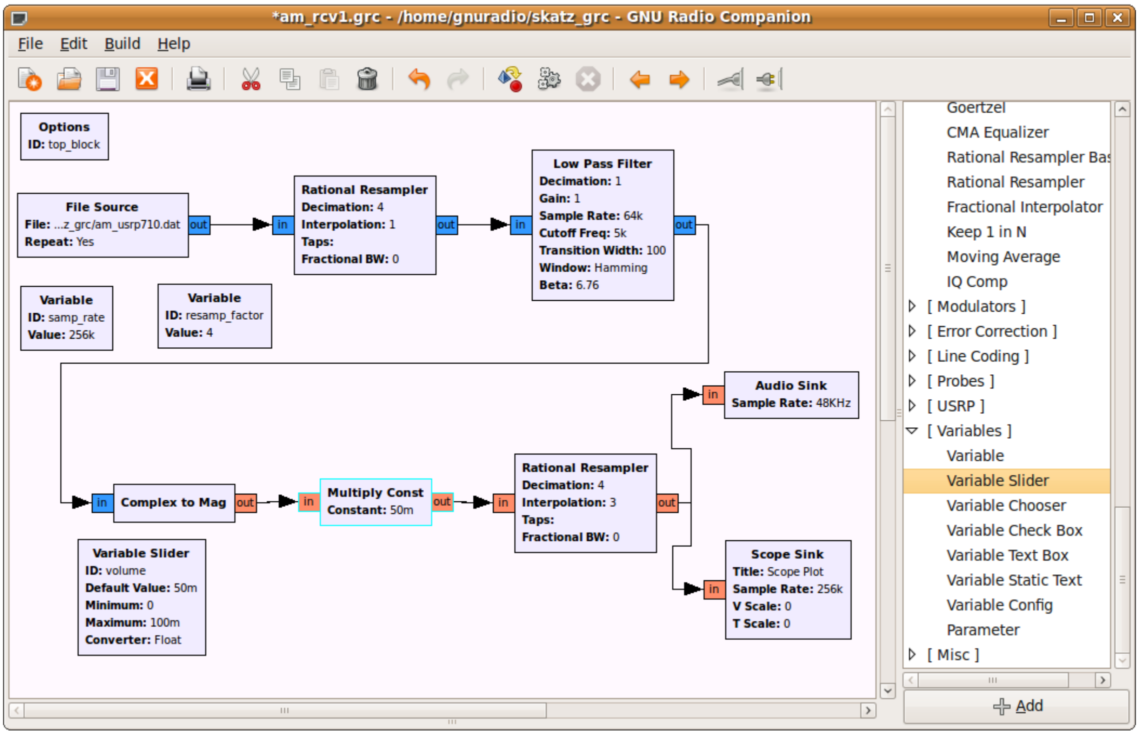
Grid Position

Notebook

Documentation:

This block creates a variable with a slider. Leave the label blank to use the variable id as the label. The value must be a real number. The value

19). Set the Constant in the Multiply Const block to “volume” so that the slider controls it. The final flow graph is shown below.



Options
ID: top_block

File Source
File: ...z_grc/am_usrp710.dat
Repeat: Yes

Rational Resampler
Decimation: 4
Interpolation: 1
Taps:
Fractional BW: 0

Low Pass Filter
Decimation: 1
Gain: 1
Sample Rate: 64k
Cutoff Freq: 5k
Transition Width: 100
Window: Hamming
Beta: 6.76

Variable
ID: samp_rate
Value: 256k

Variable
ID: resamp_factor
Value: 4

Audio Sink
Sample Rate: 48KHz

Complex to Mag

Multiply Const
Constant: 50m

Rational Resampler
Decimation: 4
Interpolation: 3
Taps:
Fractional BW: 0

Variable Slider
ID: volume
Default Value: 50m
Minimum: 0
Maximum: 100m
Converter: Float

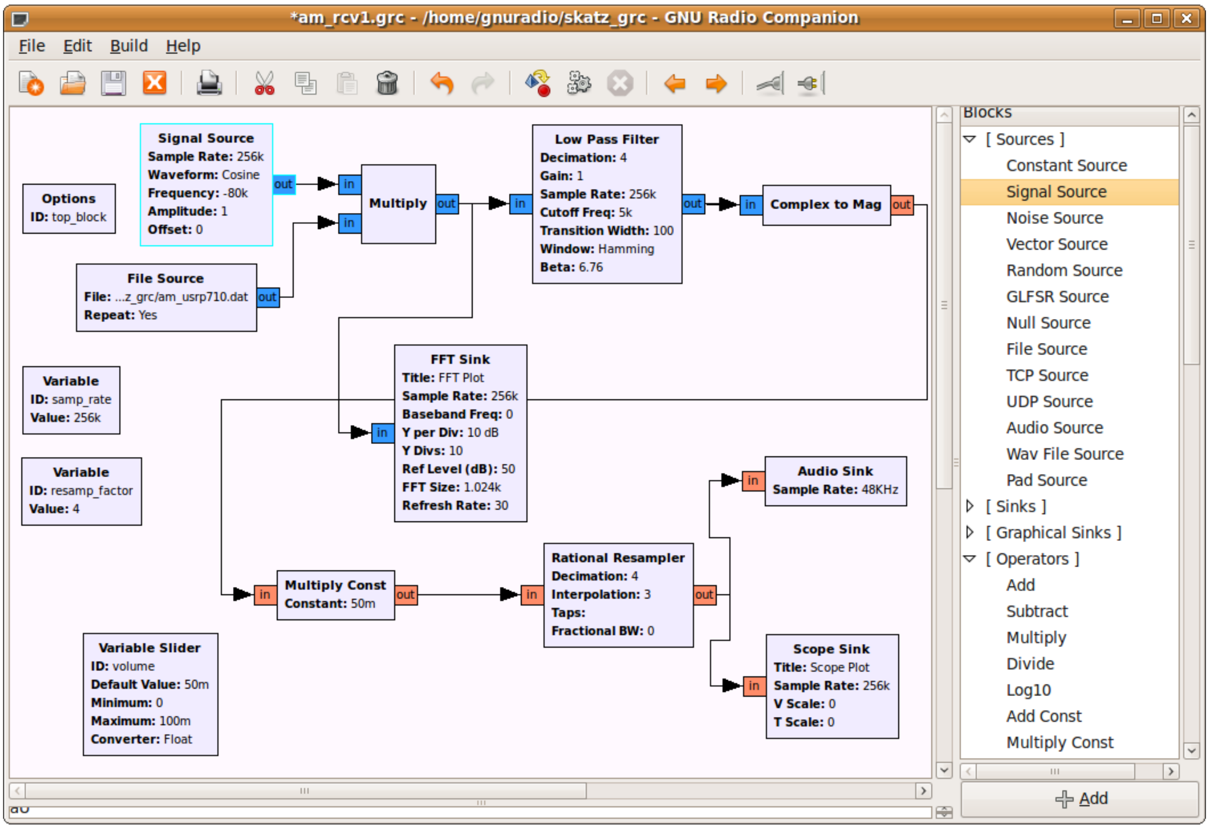
Scope Sink
Title: Scope Plot
Sample Rate: 256k
V Scale: 0
T Scale: 0

20). Execute the flow graph. You will probably hear a distorted signal coming out of the speaker. Reduce the slider until the audio is easy to understand. This will probably be around 3m (.003). You now have a functioning AM receiver.

21). Double click on the Low Pass Filter. Note that it can also decimate. Change the decimation in the filter block to resamp_factor and remove the Rational Resampler. The filter now handles both operations. Test the receiver again.

22). Place an FFT sink at the output of the file source, leaving the rest of the flow graph unchanged. Execute the flow graph and observe the location of the other stations in the spectrum. Note that there is a fairly strong signal at 80 KHz (really 710 + 80 = 790 KHz).

23). In order to receive this signal we need to shift it down to zero frequency so that it will pass through the low pass filter. One way to accomplish this is to multiply it by a sinusoid. Modify the flow graph as shown below. Add a Signal Source and set its parameters to output a Cosine at a frequency of -80000. This negative frequency will shift the entire spectrum to the left by 80KHz. Use a Multiply block and move the FFT sink to observe its output. Test this receiver.



Options
ID: top_block

Signal Source
Sample Rate: 256k
Waveform: Cosine
Frequency: -80k
Amplitude: 1
Offset: 0

File Source
File: ...z_grc/am_usrp710.dat
Repeat: Yes

Multiply

Low Pass Filter
Decimation: 4
Gain: 1
Sample Rate: 256k
Cutoff Freq: 5k
Transition Width: 100
Window: Hamming
Beta: 6.76

Complex to Mag

Variable
ID: samp_rate
Value: 256k

Variable
ID: resamp_factor
Value: 4

FFT Sink
Title: FFT Plot
Sample Rate: 256k
Baseband Freq: 0
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 50
FFT Size: 1.024k
Refresh Rate: 30

Audio Sink
Sample Rate: 48KHz

Multiply Const
Constant: 50m

Rational Resampler
Decimation: 4
Interpolation: 3
Taps:
Fractional BW: 0

Variable Slider
ID: volume
Default Value: 50m
Minimum: 0
Maximum: 100m
Converter: Float

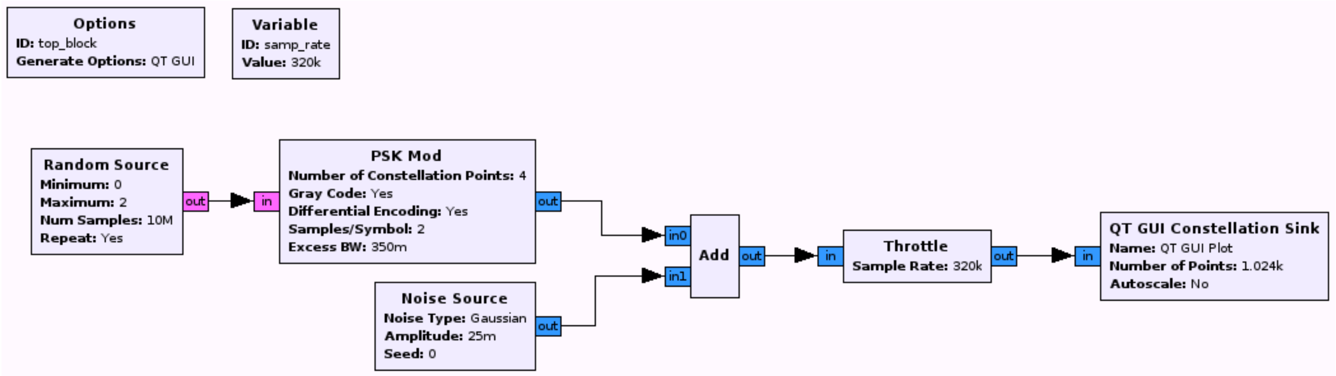
Scope Sink
Title: Scope Plot
Sample Rate: 256k
V Scale: 0
T Scale: 0

24). Add another Variable slider so that you can adjust the frequency with a slider. Test your flow graph and demonstrate that it works. You may need to adjust your volume slider for each station. This is because the stations are at varying distances away from the receiver and have different transmitted power. (Remember the link equation?)

25). The volume adjustment can be automated with an Automatic Gain Control (AGC) block. This block works by sampling its own output and adjusting its gain to keep the average output at a particular level. Insert the AGC2 block (found under Level Controls) between the Low Pass Filter and the Complex to Mag block. The audio card can only accept values between -1.0 and 1.0. We want the AGC to adjust the gain so that the sample values are always in this range. The Threshold parameter controls this. Set Threshold to 0.2. This means the AGC block will adjust its gain so that the average output value is 0.2. We will discuss the other parameters later. Set the Attack to 6.25E-4 and the Decay to 1E-5. Leave the other parameters at their default values. Set the range of the Volume variable from 0 to 1. Test the receiver again. Adjust the volume to a comfortable level on the first station you hear and tune up and down the band. Notice you no longer need to adjust the volume. The radio functions the same as a hardware radio.

Lab 4: BER Simulator

1). Build the following PSK modulator and transmitter:



Options
ID: top_block
Generate Options: QT GUI

Variable
ID: samp_rate
Value: 320k

Random Source
Minimum: 0
Maximum: 2
Num Samples: 10M
Repeat: Yes

PSK Mod
Number of Constellation Points: 4
Gray Code: Yes
Differential Encoding: Yes
Samples/Symbol: 2
Excess BW: 350m

Noise Source
Noise Type: Gaussian
Amplitude: 25m
Seed: 0

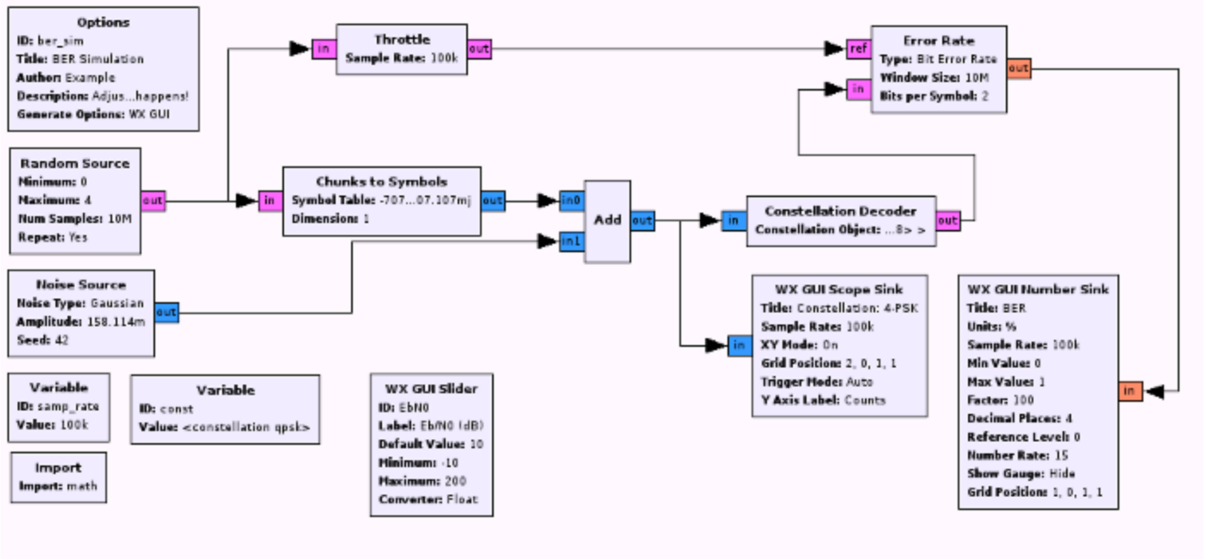
Throttle
Sample Rate: 320k

QT GUI Constellation Sink
Name: QT GUI Plot
Number of Points: 1.024k
Autoscale: No

2). Generate the flow graph and run it. What do you observe? Add a Time Sink and a Frequency Sink. What do you observe?

3). Add a Channel Model. Compare the figures in two Sinks as outputs of the PSK Modulator and the Channel Model. Increase the value of the noise amplitude. What do you observe?

4.) Build the following bER simulator: Move the Eb/N0 slider and see how it will increase or decrease the bit error rate.



Options
ID: ber_sim
Title: BER Simulation
Author: Example
Description: Adjus...happens!
Generate Options: WX GUI

Random Source
Minimum: 0
Maximum: 4
Num Samples: 10M
Repeat: Yes

Noise Source
Noise Type: Gaussian
Amplitude: 158.114m
Seed: 42

Variable
ID: samp_rate
Value: 100k

Import
Import: math

Variable
ID: const
Value: <constellation qpsk>

WX GUI Slider
ID: EbN0
Label: Eb/N0 (dB)
Default Value: 10
Minimum: -10
Maximum: 200
Converter: Float

Throttle
Sample Rate: 100k

Chunks to Symbols
Symbol Table: -707...07.107mj
Dimension: 1

Error Rate
Type: Bit Error Rate
Window Size: 10M
Bits per Symbol: 2

Constellation Decoder
Constellation Object: ...8> >

WX GUI Scope Sink
Title: Constellation: 4-PSK
Sample Rate: 100k
XY Mode: On
Grid Position: 2, 0, 1, 1
Trigger Mode: Auto
Y Axis Label: Counts

WX GUI Number Sink
Title: BER
Units: %
Sample Rate: 100k
Min Value: 0
Max Value: 1
Factor: 100
Decimal Places: 4
Reference Level: 0
Number Rate: 15
Show Gauge: Hide
Grid Position: 1, 0, 1, 1