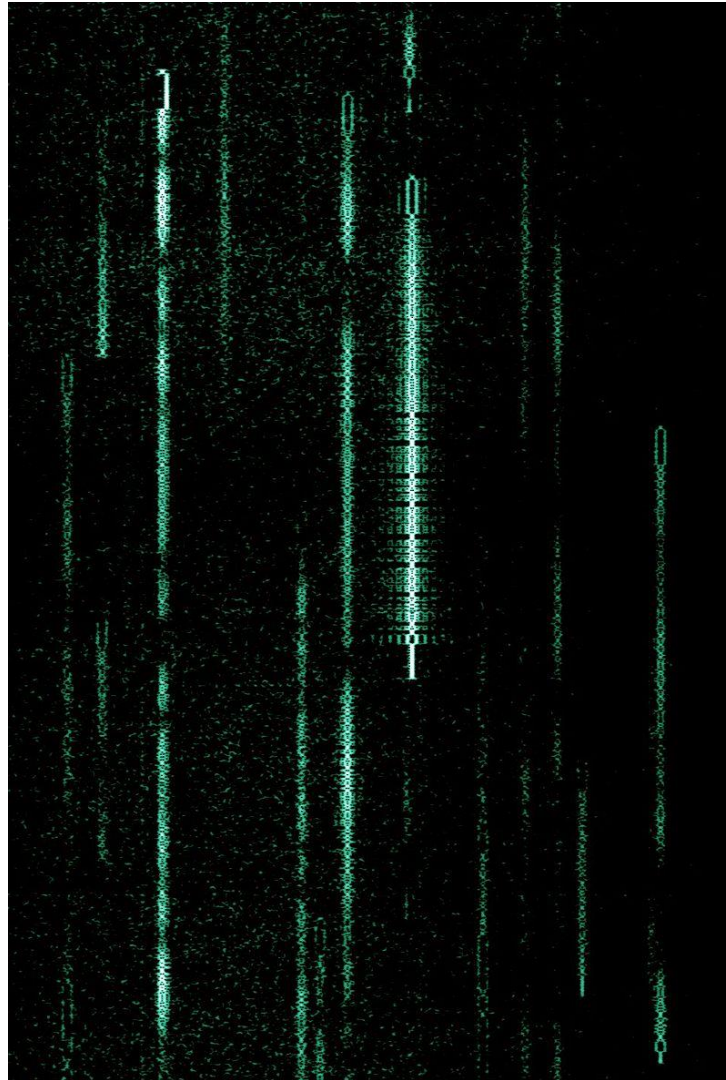


PSK31

Pamela O'Shea - @0xsh_ - Platypuscon - 24th Sept 2016



PSK31 looks like...



PSK31 sounds like...



PSK31_sample.ogg

Amateur Radio Frequencies (RTLSDR Ranges)

- RTLSDR dongle frequency range of 24 – 1766 MHz (frequencies above 1.2 GHz may require cooling/heatsinking)
- PSK31 typical transmit frequencies (in range of RTLSDR dongle):
 - 24.920 MHz 12 meters
 - 28.120 MHz 10 meters
 - 50.290 MHz 6 meters

Some terms

- **Bit rate (bps):** Speed of data in bits per second. With PSK31 where 1 bit per symbol, baud & bit rates are the same.
- **Baud rate (Bd):** number of symbols transmitted over a line per second.
- **Bandwidth (Hz):** difference between upper and lower frequencies of a given spectrum e.g. can have multiple channels within a bandwidth.
- **Baseband:** signal transmitted without modulation i.e. no shift in the range of frequencies of the signal, and is a low frequency - contained within the band of frequencies from close to 0 hertz up to a higher cut-off frequency or maximum bandwidth.

What is PSK31?

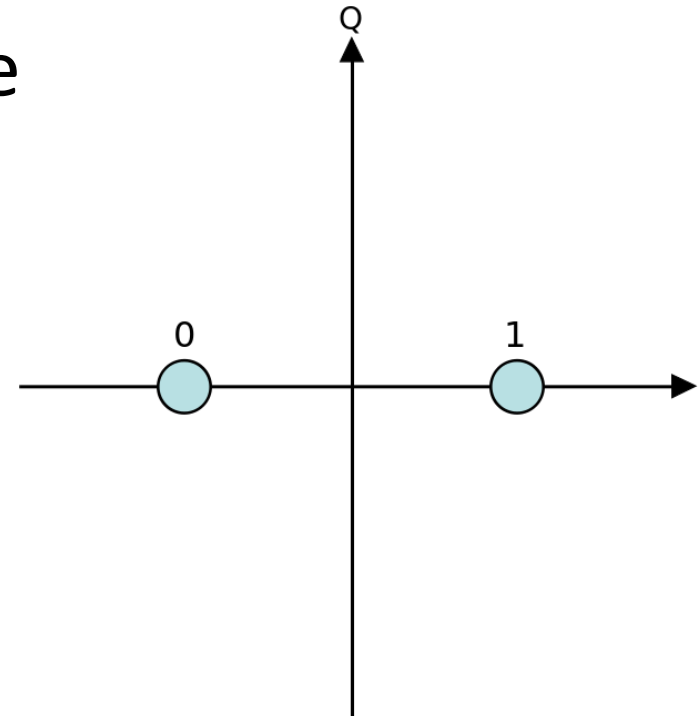
- Created by Peter Martinez (G3PLX)
- Released in 1998
- Used by amateur radio operators for real time chat
- Phase Shift Keying
- 31.25 baud rate & bits per second
- Baud rate matches a typical typing speed of 50 wpm approx.

What is PSK31?

- Phase modulated
- Remember the three main types of modulation are:
 - Amplitude-shift keying (ASK)
 - Frequency-shift keying (FSK)
 - Phase-shift keying (PSK)

What is PSK31?

- Uses binary phase-shift keying (BPSK/2-PSK)
- Two phases
- 180 degrees apart (anywhere on plane)
- BPSK is most tolerant to noise



What is PSK31?

- Not packet based
- Modulates in phase an audio signal
- Audio signal then modulates in amplitude a carrier sent over the air
- Can be used by equipment designed for audio

What is PSK31?

- 0 = phase shift of π radians
- 1 = no phase shift
- Characters = varicode
- Start of character = 00
- Data rate = 31.25 baud

Varicode

- Includes most of 7-bit ASCII characters
- Start with: 1
- End with: 1
- Never: have 00
- Break between characters: 00
- Prefix with long string of 000s

Varicode

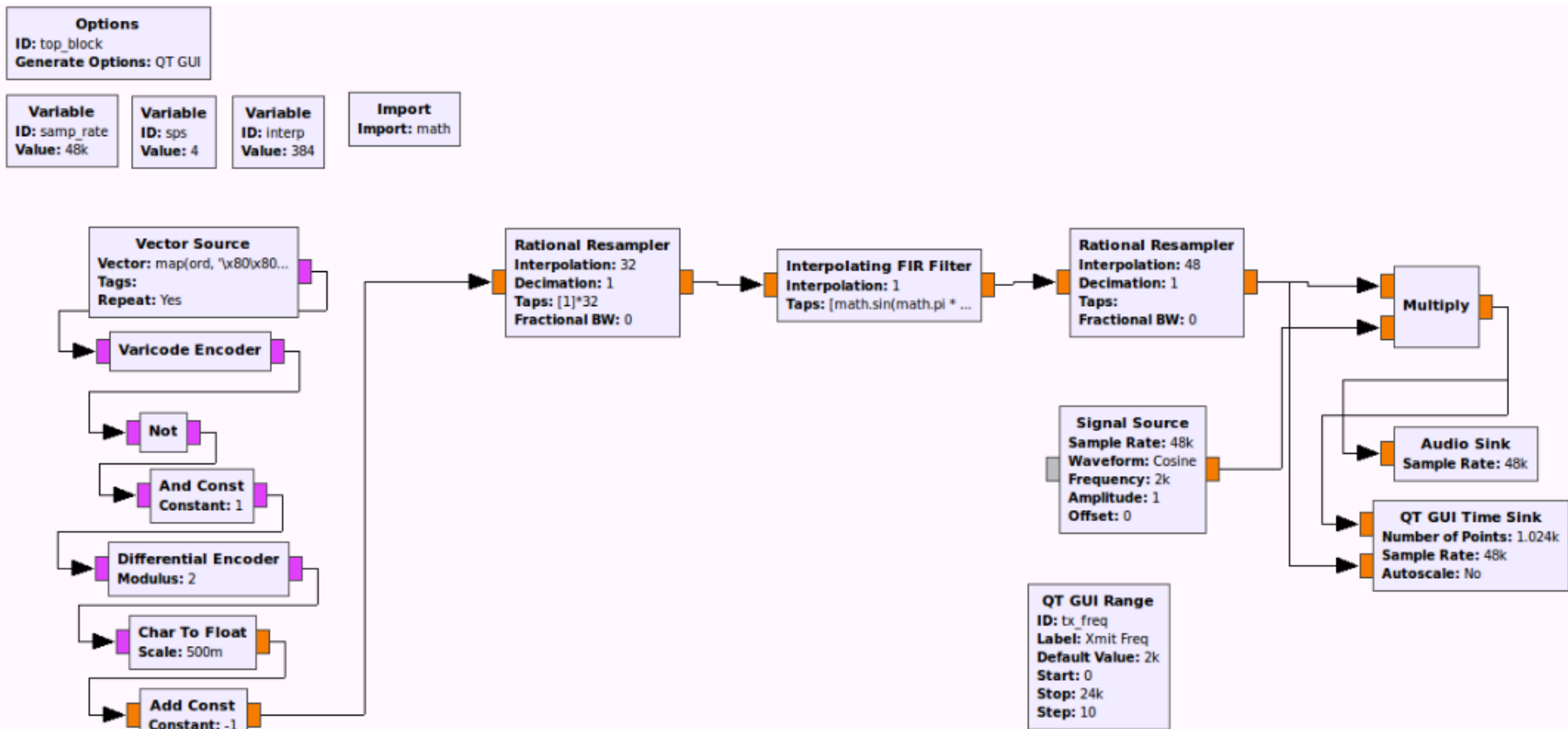
- platypus:
- 111111 00 11011 00 1011 00 101 00
1011101 00 111111 00 110111 00
10111 00

```
'a': '1011',  
'b': '1011111',  
'c': '101111',  
'd': '101101',  
'e': '11',  
'f': '111101',  
'g': '1011011',  
'h': '101011',  
'i': '1101',  
'j': '111101011',  
'k': '10111111',  
'l': '11011',  
'm': '111011',  
'n': '1111',  
'o': '111',  
'p': '111111',  
'q': '110111111',  
'r': '10101',  
's': '10111',  
't': '101',  
'u': '110111',  
'v': '1111011',  
'w': '1101011',  
'x': '11011111',  
'y': '1011101',  
'z': '111010101',
```

Generating a message

- Sample rate = 48,000Hz (audio rate)
- 1 channel
- 2 bytes per sample

Transmit PSK31 with an audio card



Materials

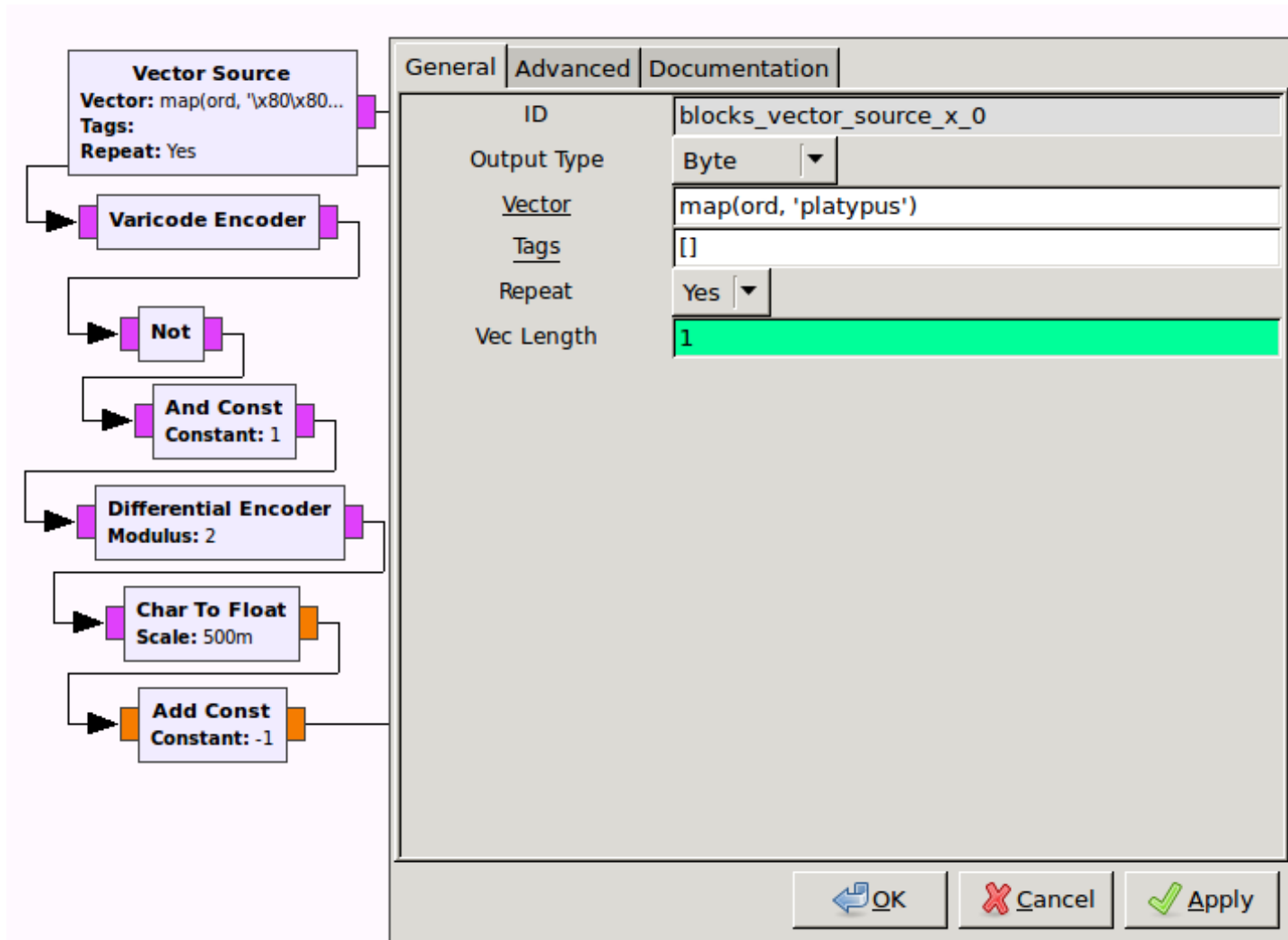
- Using this PSK31 example as a base:
<https://github.com/tkuester/gr-psk31/>
- All files, install instructions and modifications are described here: <https://github.com/0xsh>

Transmit Graph: Vector Source

Vector source

- A vector is a dynamically sized sequence of objects (compared to a fixed array)
- Interpreted as list of integers
- Truncated to bytes when output
- Vector: “map(ord, ‘platypus’)”
- Repeat: Yes

Transmit Graph: Vector Source

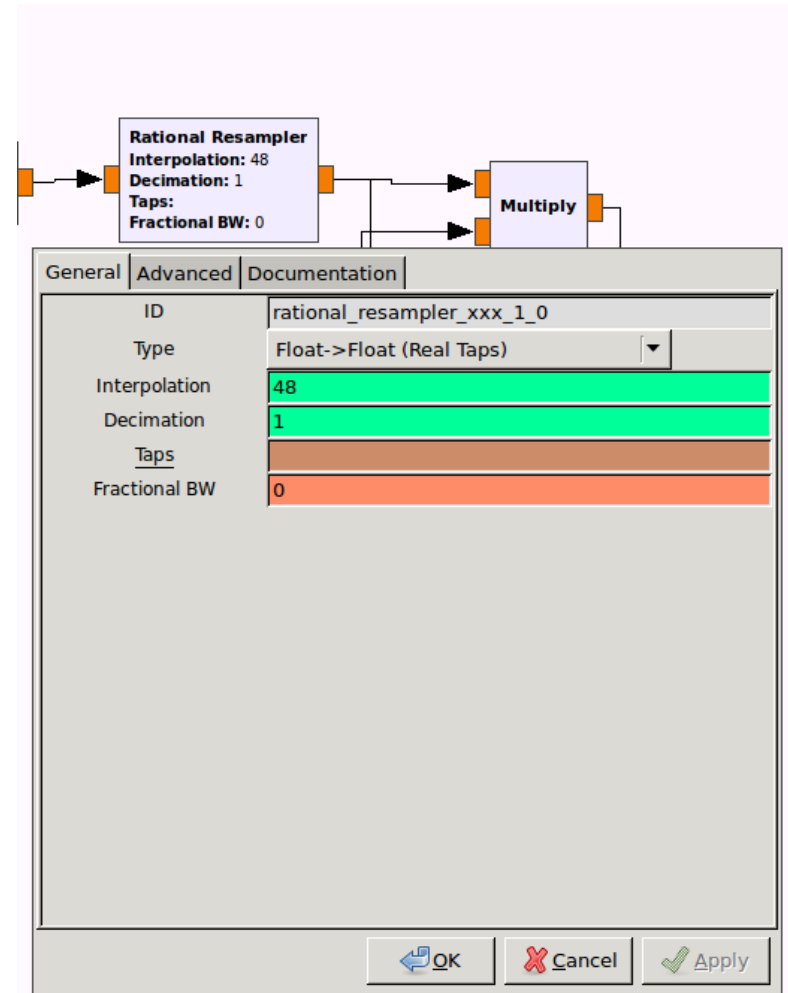
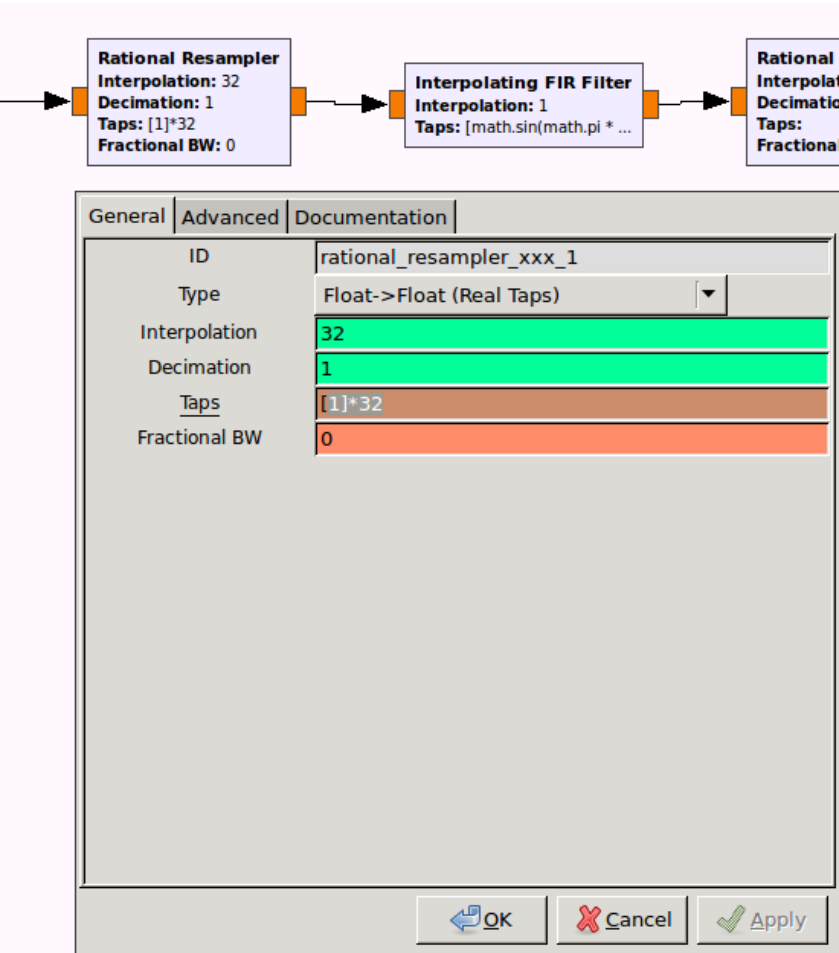


Transmit Graph: Rational Resampler

Rational Resampler

- Convert from one sample rate to another
- Combined interpolator (multiply by) & decimator (divide by)
- All following blocks should use newly set sample rate

Transmit Graph: Rational Resampler

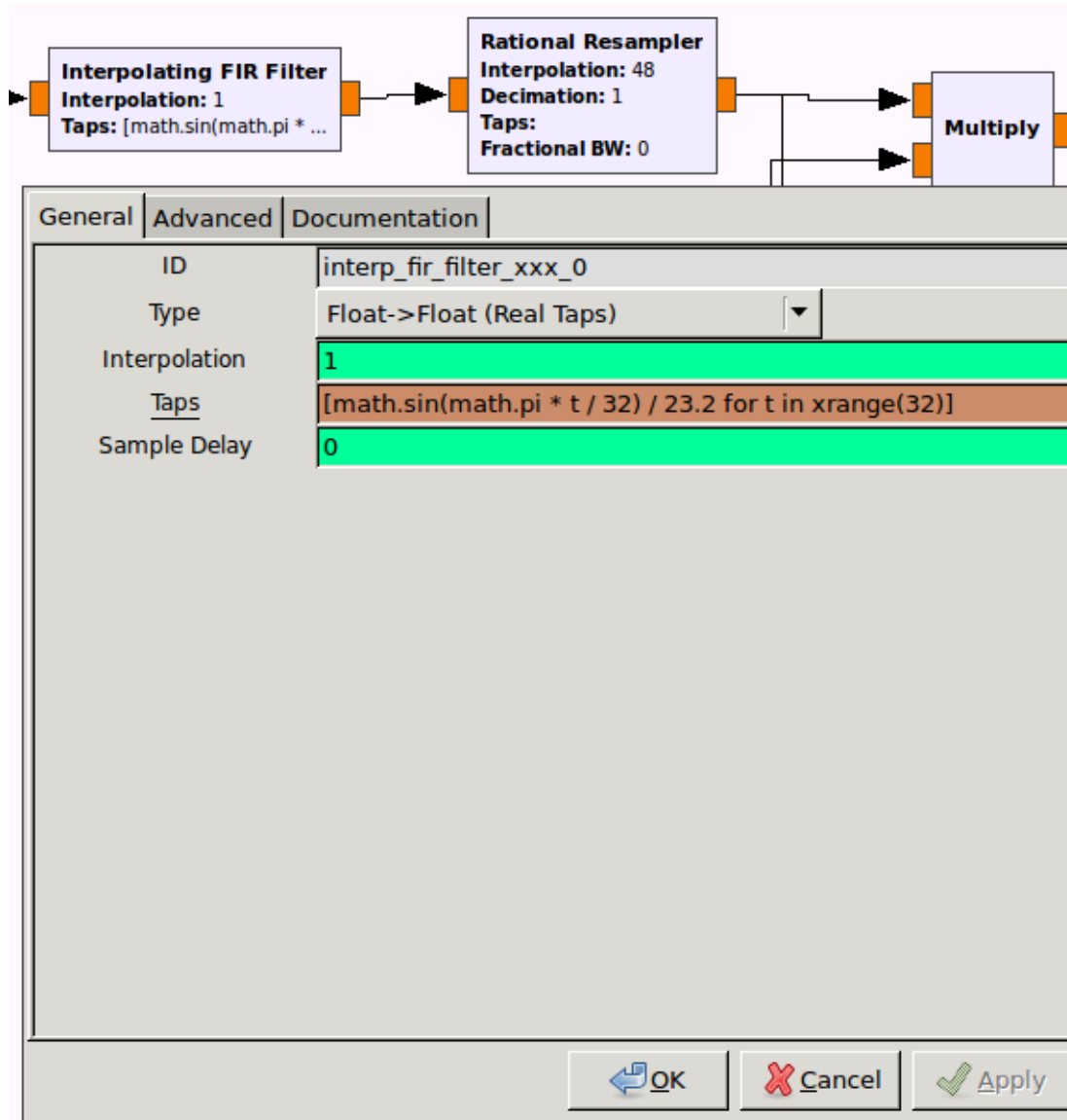


Transmit Graph: Interpolating FIR Filter

Interpolating FIR Filter

- FIR = Finite Impulse Response filter
- Settles to 0 in finite time
- Tap = a delay
- More taps = more stopband attenuation, less ripple, narrower filters

Transmit Graph: Interpolating FIR Filter



Transmit Graph: Audio Sink

Signal Source
Sample Rate: 48k
Waveform: Cosine
Frequency: 2k

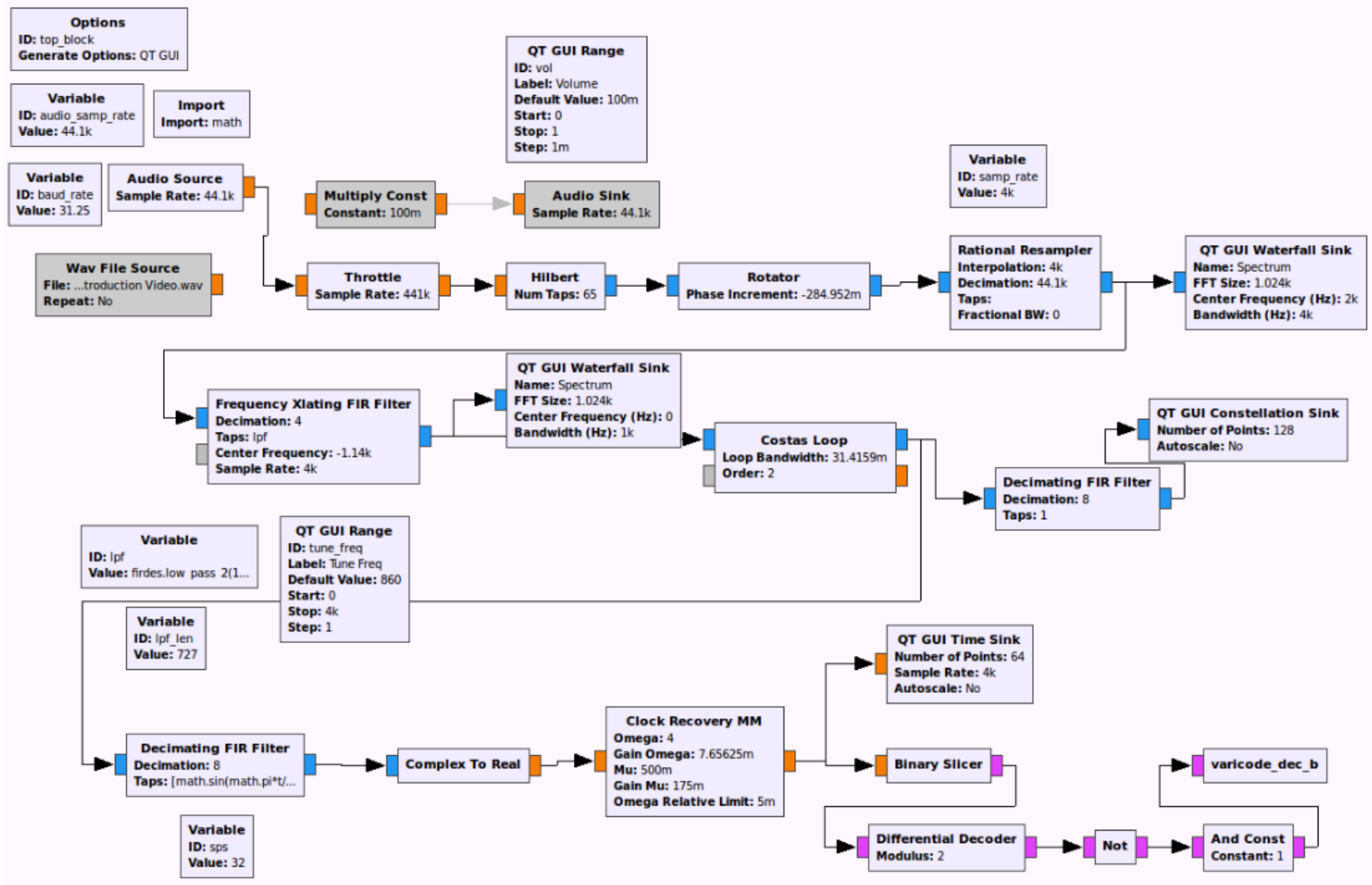
Audio Sink
Sample Rate: 48k

General | Advanced | Documentation

ID	audio_sink_0_0
Sample Rate	samp_rate
Device Name	
OK to Block	Yes
Num Inputs	1

OK Cancel Apply

Receive PSK31 with an audio card



Receive Graph: Audio Source

Audio Source
Sample Rate: 44.1k

General | Advanced | Documentation

ID	audio_source_0
Sample Rate	audio_samp_rate
Device Name	
OK to Block	Yes
Num Outputs	1

OK Cancel Apply

Receive Graph: Throttle

Throttle

- Gnuradio operates at full speed (when no real hardware in way to slow down)
- Need throttle to control rate
- Don't use with real hardware (throttle is a bad clock and will end up with timing issues)

Receive Graph: Hilbert Transform

Hilbert Transform

- Truncates the filter to the number of taps
- Introduces a delay into the signal

Receive Graph: Hilbert Transform

The image shows a block diagram and a configuration window for a Hilbert Transform. The block diagram at the top consists of three blocks connected in series: a Hilbert block (Num Taps: 65), a Rotator block (Phase Increment: -284.952m), and a Rational Resamp block (Interpolation: 4k, Decimation: 44.1k, Taps: , Fractional BW: 0). Below the diagram is a configuration window for the Hilbert block, titled 'Hilbert'.

The configuration window has three tabs: General, Advanced, and Documentation. The General tab is selected. It contains a table with the following parameters:

ID	Value
Num Taps	65
Window	Hamming
Beta	6.76

At the bottom of the window are three buttons: OK, Cancel, and Apply.

Receive Graph: Rotator

Rotator

- Frequency shifting
- Shift specified as a complex vector (amount of rotation per sample)

Receive Graph: Rotator

The image shows a software interface for configuring a 'Rotator' block. At the top, a signal flow graph displays the 'Rotator' block with a 'Phase Increment' of -284.952m, followed by a 'Rational Resampler' block with 'Interpolation: 4k', 'Decimation: 44.1k', 'Taps', and 'Fractional BW: 0'.

Below the graph is a configuration dialog box with three tabs: 'General', 'Advanced', and 'Documentation'. The 'General' tab is active, showing a table with the following data:

ID	
blocks_rotator_cc_0	
<u>Phase Increment</u>	$2 * \text{math.pi} * -2000 / \text{audio_samp_rate}$

At the bottom of the dialog are three buttons: 'OK' (with a blue arrow icon), 'Cancel' (with a red X icon), and 'Apply' (with a green checkmark icon).

Receive Graph: Rational Resampler

The image shows a configuration window for a 'Rational Resampler' block. At the top, a block diagram shows the 'Rational Resampler' block connected to a 'QT GUI Waterfall Sink' block. The 'Rational Resampler' block has the following parameters: Interpolation: 4k, Decimation: 44.1k, Taps, and Fractional BW: 0. The 'QT GUI Waterfall Sink' block has the following parameters: Name: Spectrum 1, FFT Size: 1.024k, Center Frequency (Hz): 2k, and Bandwidth (Hz): 4k.

The configuration window has three tabs: 'General', 'Advanced', and 'Documentation'. The 'General' tab is selected. It contains a table with the following parameters:

ID	Value
rational_resampler_xxx_2	
Type	Complex->Complex (Complex Taps) ▼
Interpolation	4000
Decimation	44100
Taps	
Fractional BW	0

At the bottom of the window are three buttons: 'OK' (with a blue arrow icon), 'Cancel' (with a red X icon), and 'Apply' (with a green checkmark icon).

Receive Graph: Frequency Xlating FIR Filter

Frequency Xlating FIR Filter

- Frequency-translating FIR filter
- Often used for channel selection block
- Performs frequency translation, channel selection and decimation in one step

Receive Graph: Frequency Xlating FIR Filter

The screenshot displays a software interface for configuring a **Frequency Xlating FIR Filter**. The top section shows a block diagram with the filter block connected to a **Spectrum 2** block. The filter block's parameters are:

- Decimation: 4
- Taps: lpf
- Center Frequency: 3
- Sample Rate: 4k

The **Spectrum 2** block's parameters are:

- Name: Spectrum 2
- FFT Size: 1.024k
- Center Frequency (Hz): 0
- Bandwidth (Hz): 1k

Below the block diagram is a configuration window with three tabs: **General**, **Advanced**, and **Documentation**. The **General** tab is active, showing the following settings:

Parameter	Value
ID	freq_xlating_fir_filter_xxx_0
Type	Complex->Complex (Complex Taps)
Decimation	4
Taps	lpf
Center Frequency	tune_freq-2000
Sample Rate	samp_rate

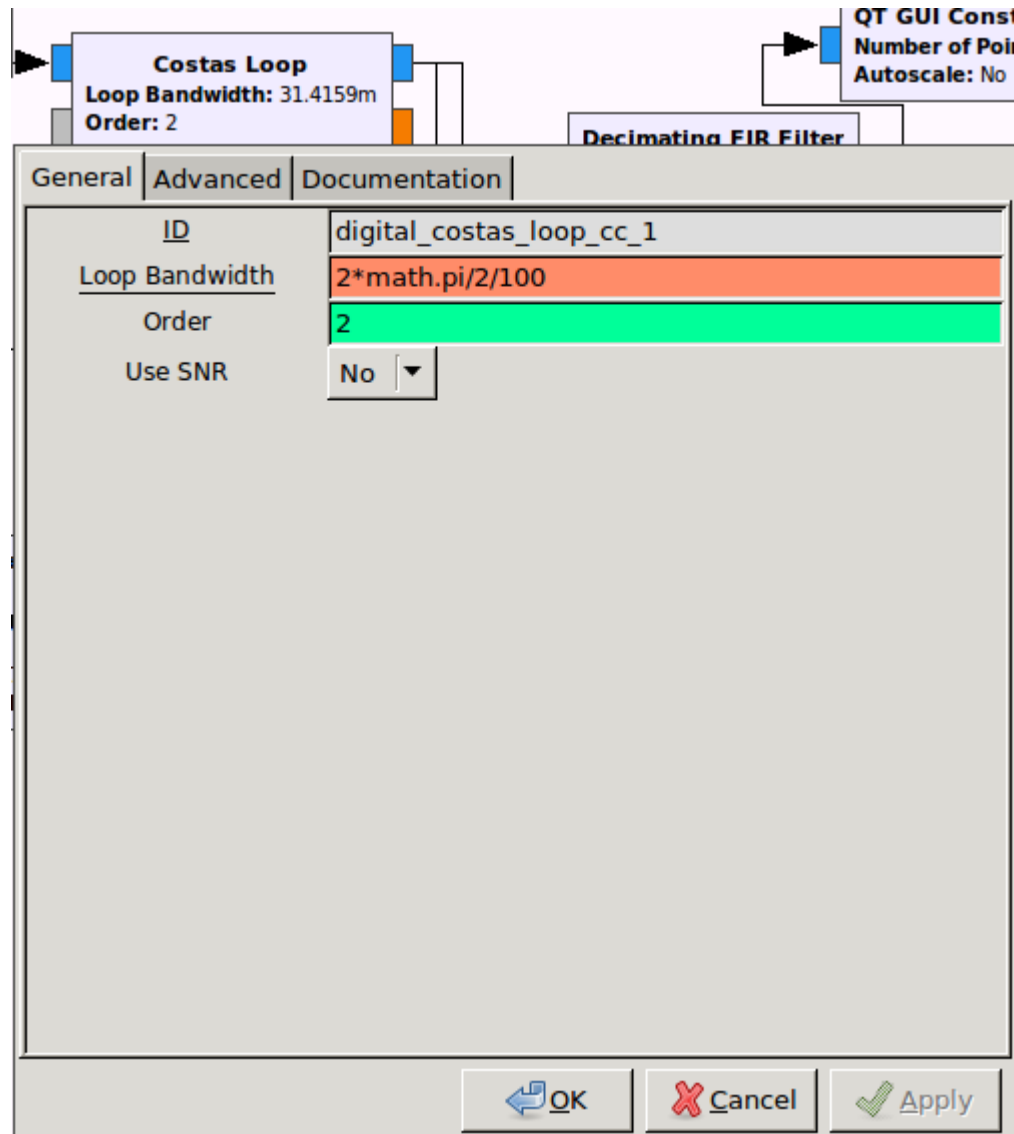
At the bottom of the configuration window are three buttons: **OK**, **Cancel**, and **Apply**.

Receive Graph: Costas Loop

Costas Loop

- Locks to the centre frequency of a signal and downconverts it to baseband

Receive Graph: Costas Loop

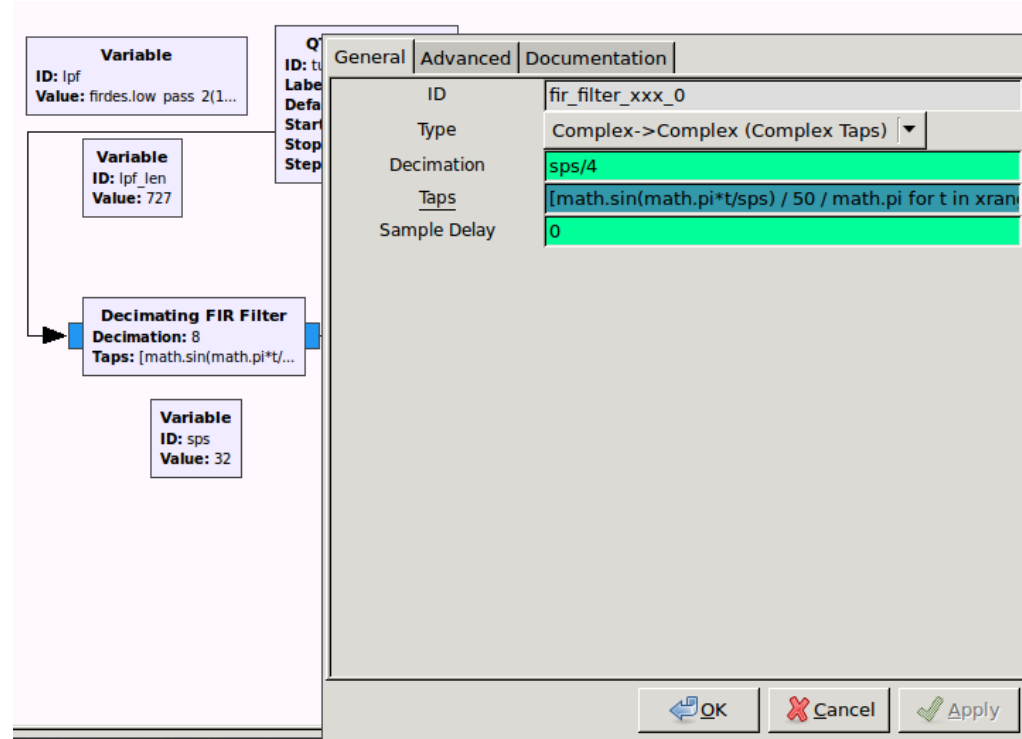
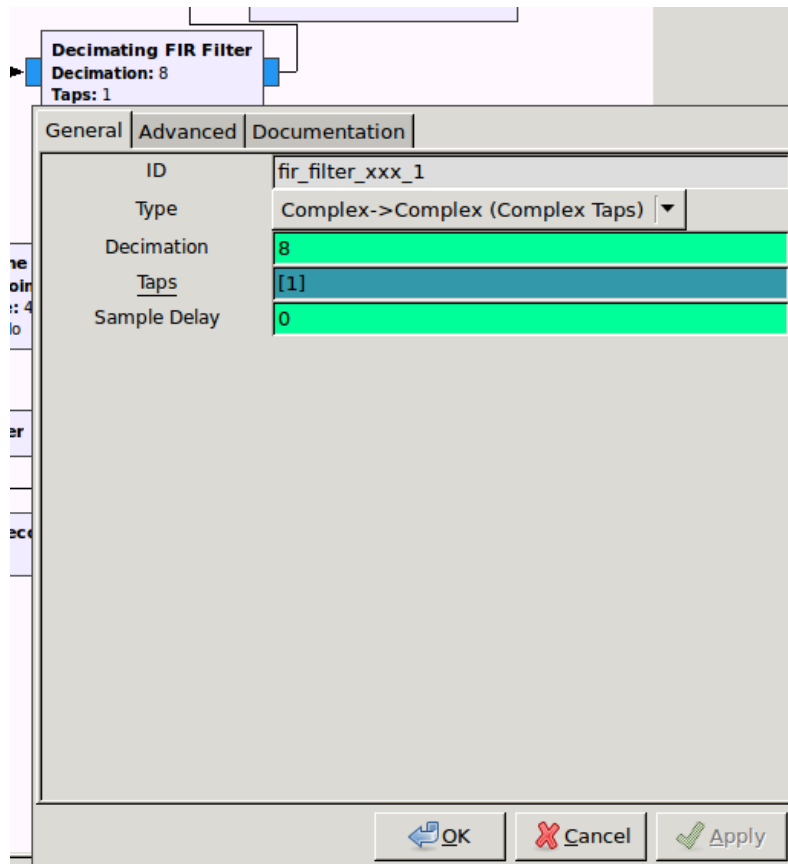


Receive Graph: Decimating FIR Filters

Decimating FIR Filters

- Decimation is reducing the output sampling rate by ignoring all but every N th sample

Receive Graph: Decimating FIR Filters



Receive Graph: Clock Recovery

The image shows a configuration window for a 'Clock Recovery MM' block. The window has three tabs: 'General', 'Advanced', and 'Documentation'. The 'Advanced' tab is selected, displaying a table of parameters. Below the table are 'OK', 'Cancel', and 'Apply' buttons. At the bottom, a block diagram shows the 'Clock Recovery MM' block connected between a 'Complex To Real' block and a 'Binary Slicer' block. The block diagram also displays the configured parameter values.

Parameter	Value
ID	digital_clock_recovery_mm_xx_0
Type	Float
<u>Omega</u>	$4 \cdot (1 + 0.0)$
<u>Gain Omega</u>	$0.25 \cdot 0.175 \cdot 0.175$
<u>Mu</u>	0.5
<u>Gain Mu</u>	0.175
Omega Relative Limit	0.005

Clock Recovery MM
Omega: 4
Gain Omega: 7.65625m
Mu: 500m
Gain Mu: 175m
Omega Relative Limit: 5m

Materials

- Next, follow the Step1, Step2 & Step 3 instructions here: <https://github.com/0xsh>

Other work you can play with...

- **GoodPSK** “is a tool for generating PSK31 audio recordings, sometimes with strange or clever attributes” & check out the lectures!: <https://github.com/travisgoodspeed/goodpsk>
- <https://sdradventure.wordpress.com/2011/10/15/gnuradio-psk31-decoder-part-1>
- <https://sdradventure.wordpress.com/2011/10/15/gnuradio-psk31-decoder-part-2/>
- <https://github.com/JasonBens/PSK31-transceiver>
- <https://github.com/tkuester/gr-psk31>
- <https://github.com/christophL/gr-digimodes>
- <https://github.com/argilo/sdr-examples>

References

- <http://aintel.bi.ehu.es/psk31theory.html>
- <http://www.arrl.org/digital-data-modes>
- https://en.wikipedia.org/wiki/Phase-shift_keying
- <https://en.wikipedia.org/wiki/PSK31>
- <https://en.wikipedia.org/wiki/Baud>
- <http://people.scs.carleton.ca/~barbeau/SDRCRBook/Content/chapter09.pdf>
- <https://github.com/travisgoodspeed/goodpsk>
- <https://github.com/tkuester/gr-psk31>
- <http://www.w1hkj.com/FldigiHelp/index.html>

Glossary

- FIR Filter: Finite Impulse Response Filter
- PSK: phase shift keying
- SPS: samples per second

Questions...

IT'S A
WOMAN'S WAR
TOO!

JOIN THE
WAVES

YOUR COUNTRY NEEDS YOU NOW

APPLY TO YOUR NEAREST NAVY RECRUITING STATION
OR OFFICE OF NAVAL OFFICER PROCUREMENT

JOHN FASTER USNR

Cyberspectrum Melbourne

Twitter

[@sdr_melbourne](https://twitter.com/sdr_melbourne)

Email

sdr.melbourne@gmail.com

Slack

sdr-melbourne.slack.com (email for an invite)

Meetup

<https://www.meetup.com/Cyberspectrum-Melbourne>

Blog

<http://randomkeystrokes.com/category/sdr/>

Github

<https://github.com/sdr-melbourne>

YouTube

<https://www.youtube.com/channel/UCLBloqxOXEj4fH7s>
ULA (SDR Melbourne Channel)

