

GNU Radio ile Uygulamalı Haberleşme Sistemleri

Linux Kış Kampı
Eskişehir, 10-13 Şubat 2025

Outline

Getting started

Survey

Who am I?

[Schedule](#)

[Installation](#)

Getting Started Survey

- Think before answering



About me

BSc, METU, 1998



MSc, AYBU, 2015



PhD, TOBB ETÜ, ...



TOBB ETÜ
Economic and Technical University

Certificates

**Certified Instructor and University
Ambassador at NVIDIA**



GitHub Teacher



Work Experience

- ASELSAN
 - Türkiye's leading defence company
 - Aselsan 47. Rank in "Defense News' Top 100 list"
 - ~15 years
 - March '23



About Me

- 25 years in software development
- 15+ years in telecom field
- PhD student @ TOBB ETÜ
- Lecturer @ TOBB ETÜ
 - 2021-2022 Summer ELE361L course (Telecom Lab)
 - 2022-2023 Fall ELE361L
 - 2023-2024 Fall ELE361L
 - 2023-2024 Summer ELE361L
 - 2024-2025 Fall ELE361L
 - ELE495 Senior Project

Embedded Experience - Monitoring Receivers

TI 8-core DSP/SysBIOS

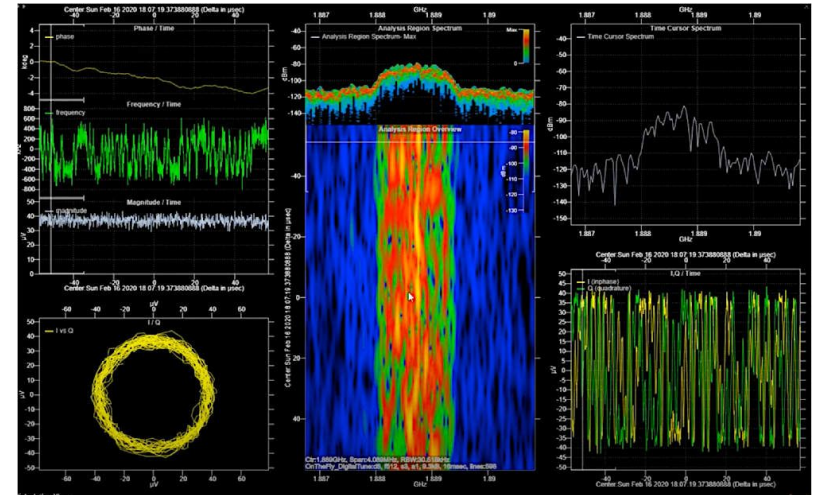


Intel i7/VxWorks



SIGINT: Signal Analysis Project

- Offline/Online Analysis
- Demodulation/Decoding
- Parameters
 - Center Freq
 - Modulation Type
 - Baud Rate



ELE361L

ELE361L

Communication Systems Laboratory

Course Info - 5min 

<https://ele361l.github.io/>



Open System

Course was designed from the ground up to have open system components.



Modular

Course has modular architecture. It can be customized or extended by adding new modules.



Real-World Signals

We are dealing with real-world signals like AM in airband, broadcast FM signals, or ADS-B signals emitted from aircrafts.



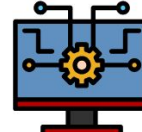
Professional Development

We support professional development of students in software. They learn industry-standard tools and improve their programming skills.



Mobility/Remote Opportunity

Course provides students mobility. They can complete their tasks wherever they feel comfortable. And it is also remote-ready.



Software-based

Since we are using Software Defined Radios, all modules are software-based.

Awards: 9. Başakşehir Innovation Contest



Awards: 3. AOSB R&D and Innovation Contest



3. Ulusal
Sanayi Odaklı
AR-GE ve İnovasyon
Proje Yarışması

50
YIL
AOSB

ÖDÜLÜ KAZANANLAR
Akademisyen/Lisansüstü Mezun

BİRİNCİ
Mücahid KUTLU
Tematik Alan: Bilgisayar
100.000 TL

İKİNCİ
Mutlu KURBAN
Tematik Alan: Tekstil
80.000 TL

ÜÇÜNCÜ
Murat SEVER
Tematik Alan: Elektrik/Elektronik
60.000 TL

500.000 TL
Ödül Havuzu

Events: SDR Academy Friedrichshafen, Germany



Events: GNU Radio Conference 2023

5-9 September 2023

Talk & Workshop



Survey Results

- Let's have a quick look at the survey results!

It's your turn

- Briefly introduce yourself
 - Your name
 - Where you're coming from
 - What you study/do
 - Your interests
 - Your expectations
- Also pin your location (university/work address) on the map
 - https://www.google.com/maps/d/edit?mid=1nKMwIWh8m1cMWTThHj1pDI5m3rw4rWfw&usp=s_haring

Schedule

- **First Day: GNU Radio Introduction, DSP, GR Simulation Mode**
- Second Day: SDR Introduction, RTL-SDR, GR Real-Time Mode
- Third Day: Analog Communications
- Fourth Day: Digital Communications

Course materials

- Check your e-mail for the repo address

GNU Radio Installation

- GR can run on all platforms
 - Linux
 - MacOS
 - Windows
- `sudo apt-get install gnuradio`

GNU Radio is...

- A signal processing library
- Designed for real-time
- The software part of an SDR
- Not a radio application
- The tool to **build your own** transceivers
- **FOSS**: Free and Open Source Software

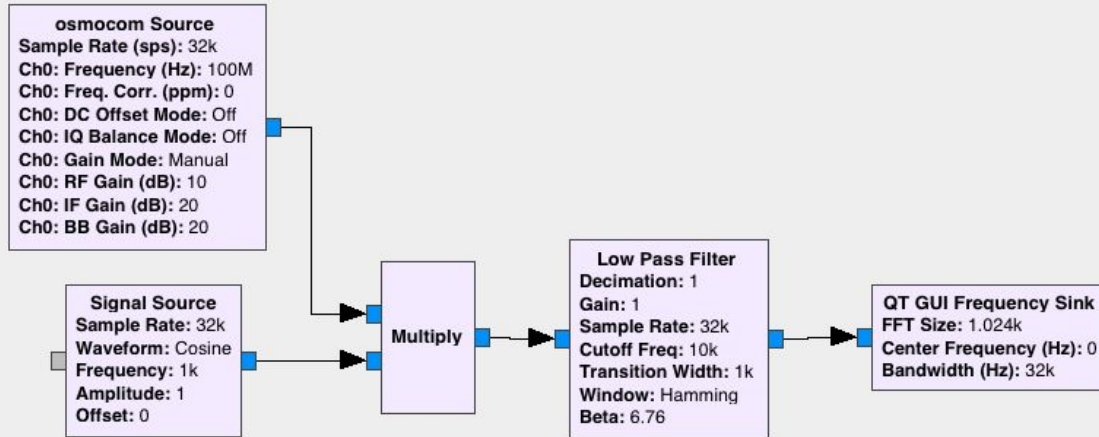


GNU Radio

- Open-source framework for SDR and signal processing
 - Founded by Eric Blossom in 2001
 - Block-based dataflow architecture
 - Each block runs in its own thread
 - Data flows through a graph called a Flowgraph
 - Blocks are nodes in a Flowgraph, and perform operations and signal processing
 - Signals normalized between -1.0 and +1.0
 - Similar in concept to MathWorks Simulink™
 - Running C++ and Python under-the-hood
 - Can write code directly, or use the GNU Radio Companion (GRC) graphical tool
-

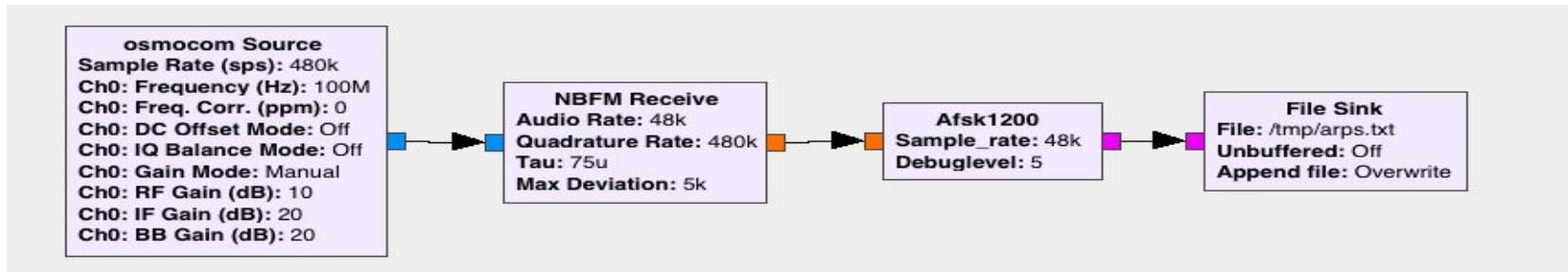
Basic Concept: Flow Graph

- Transceivers are implemented as *flow graphs*
- Similar to Simulink / schematics
- Define structure and parameters of *blocks*



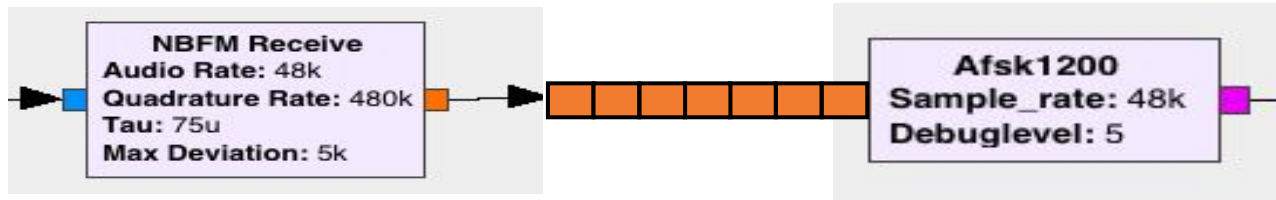
Basic Concept: Block

- Written in C++ or Python
- Implement one logical step
- Each block run in separate thread

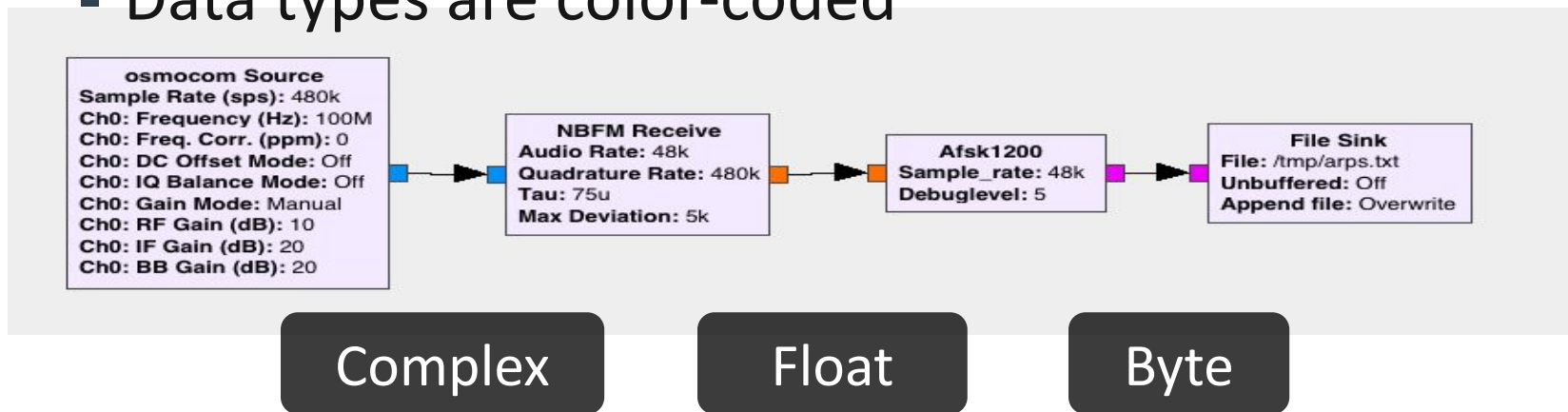


Data Streams

- Samples are buffered

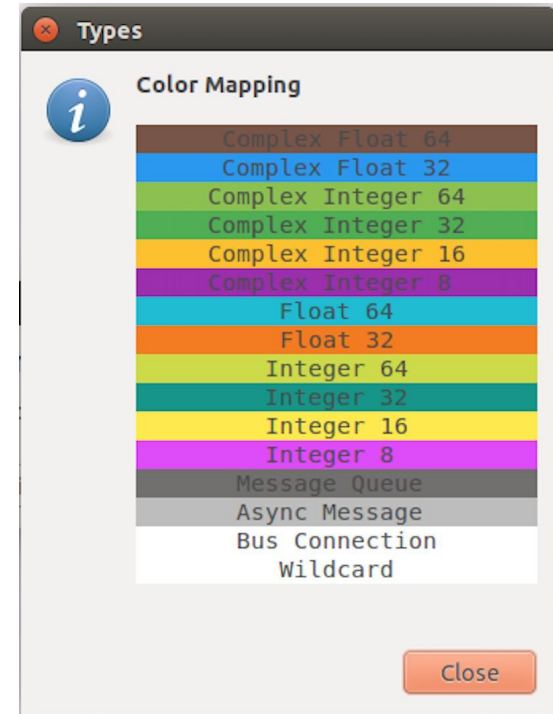


- Data types are color-coded



Color Types

Click on menu item Help->Types

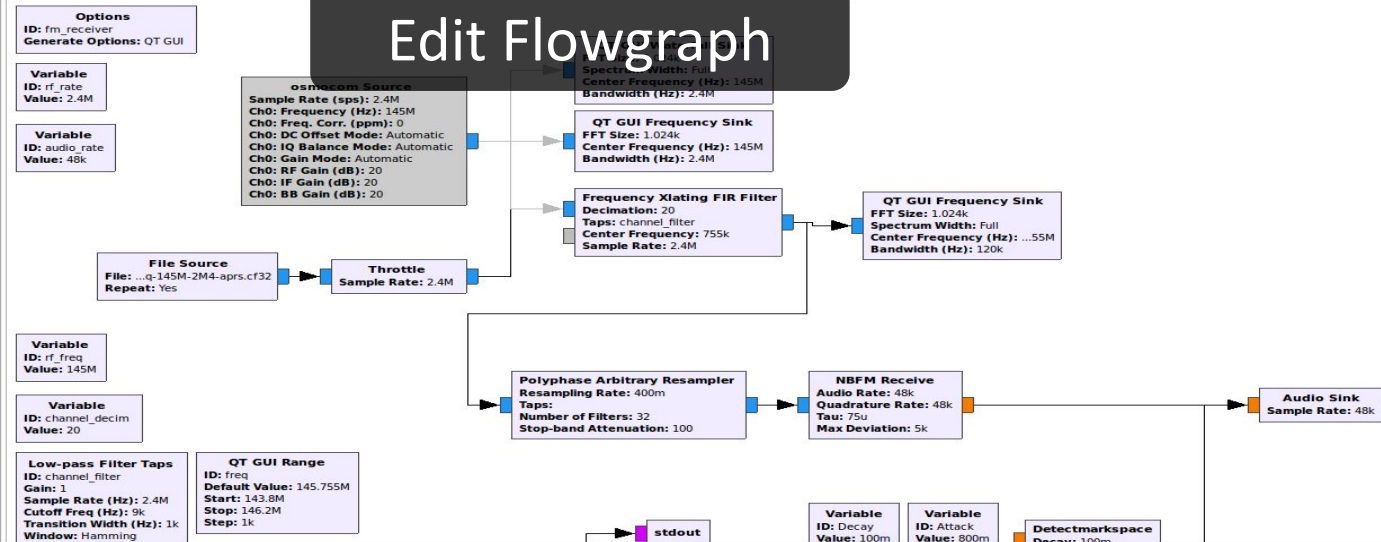


GNU Radio Companion

File Edit View Run Tools Help



fm_receiver x rds_rx x



<<< Welcome to GNU Radio Companion 3.7.12git-1109-gcbf30e9c >>>

Block paths:

/home/basti/.gnc_gnuradio
/home/basti/.gnuradio-next/share/gnuradio/gc/blocks

Loading: "/home/basti/.gnc-workshop/fm_r...
>>> Done
Loading: "/home/basti/.gnc-rds/apps/rds_rx.grc"
>>> Done

Console

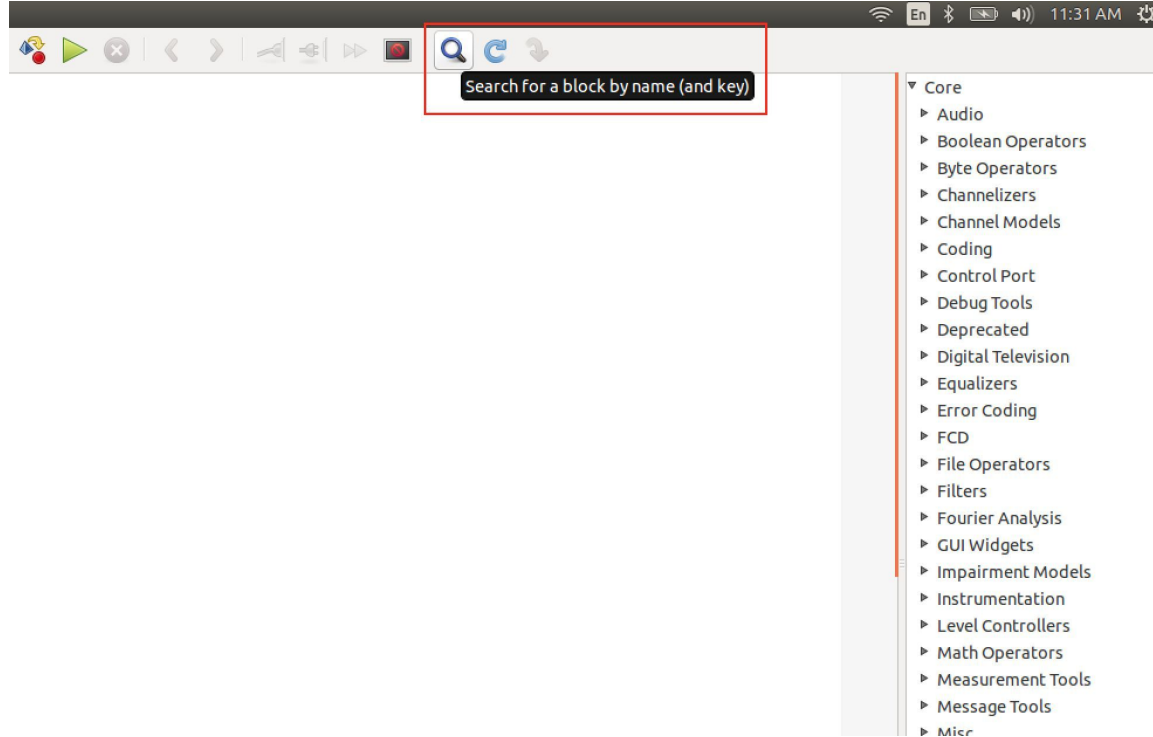
Id		Value
Imports		
Variables		
Attack	0.8	
audio_rate	48000	
channelLc	20	
channelLc	<Open Properties>	
Decay	0.1	
freq	<Open Properties>	

Variables

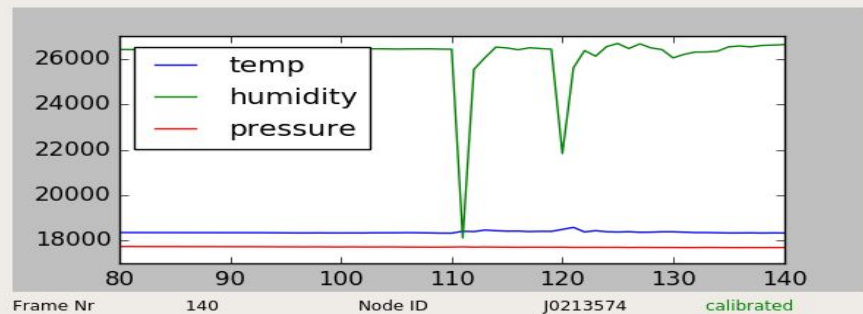
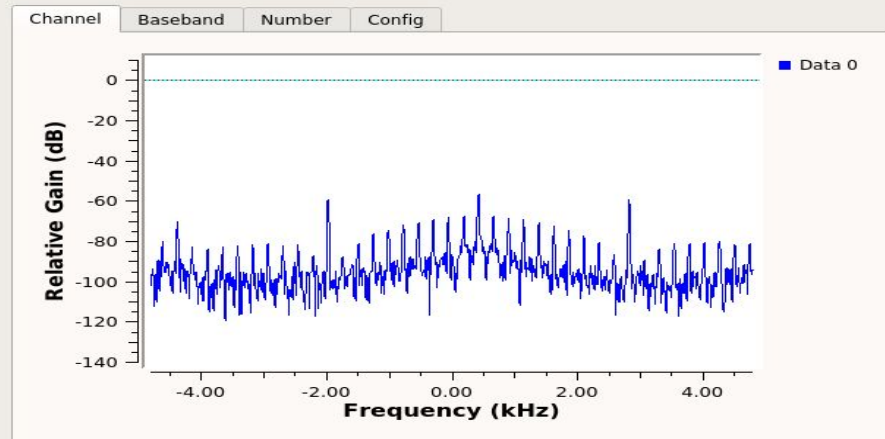
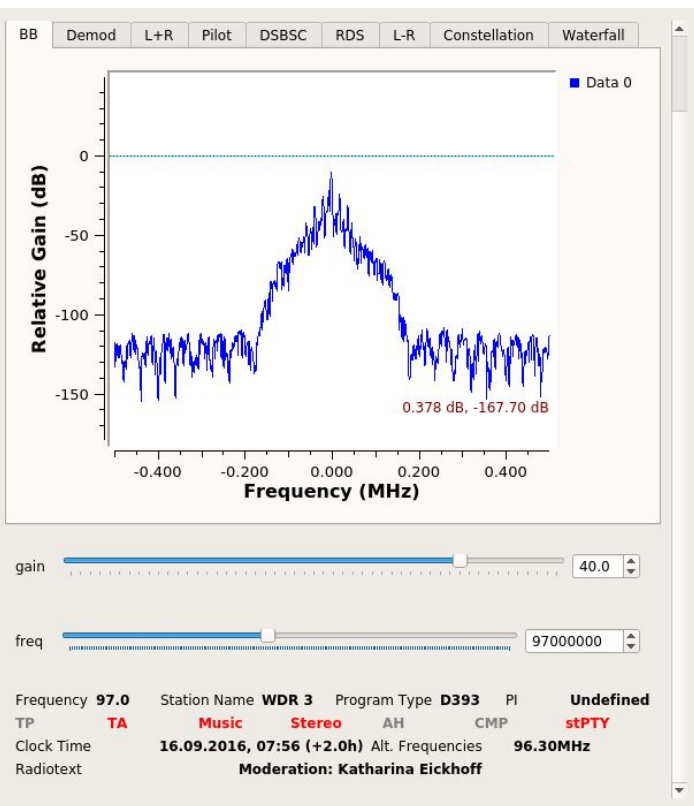
Block Library

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

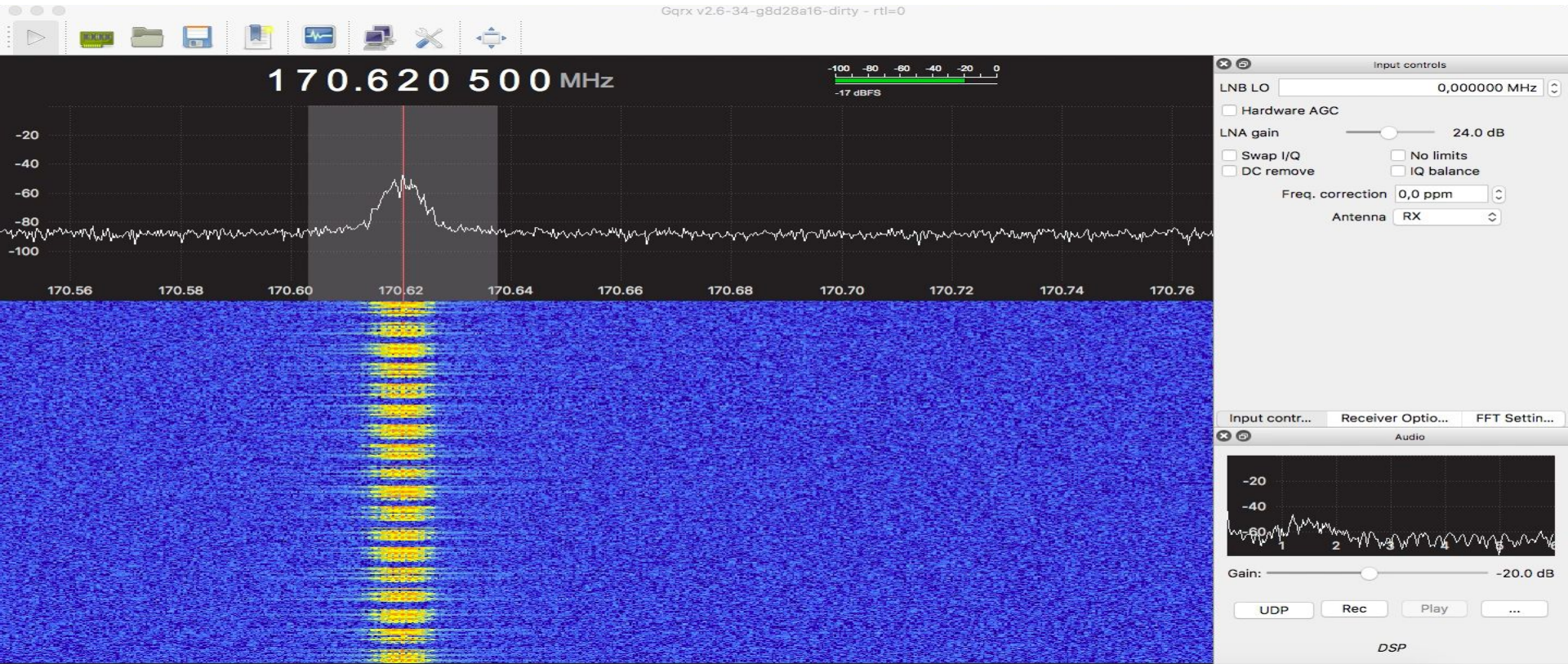
Search Blocks



GUI Output and Instrumentation




GQRX - a GNU Radio Application



Out Of Tree Modules

- GNU Radio can be extended with OOTs
- OOTs cover more specific functionality
- There is a large number available
- CGRAN is our central database




[CGRAN](#) [Projects](#) [Documentation](#) [GNU Radio](#) [VOLK](#)



The Comprehensive GNU Radio Archive Network

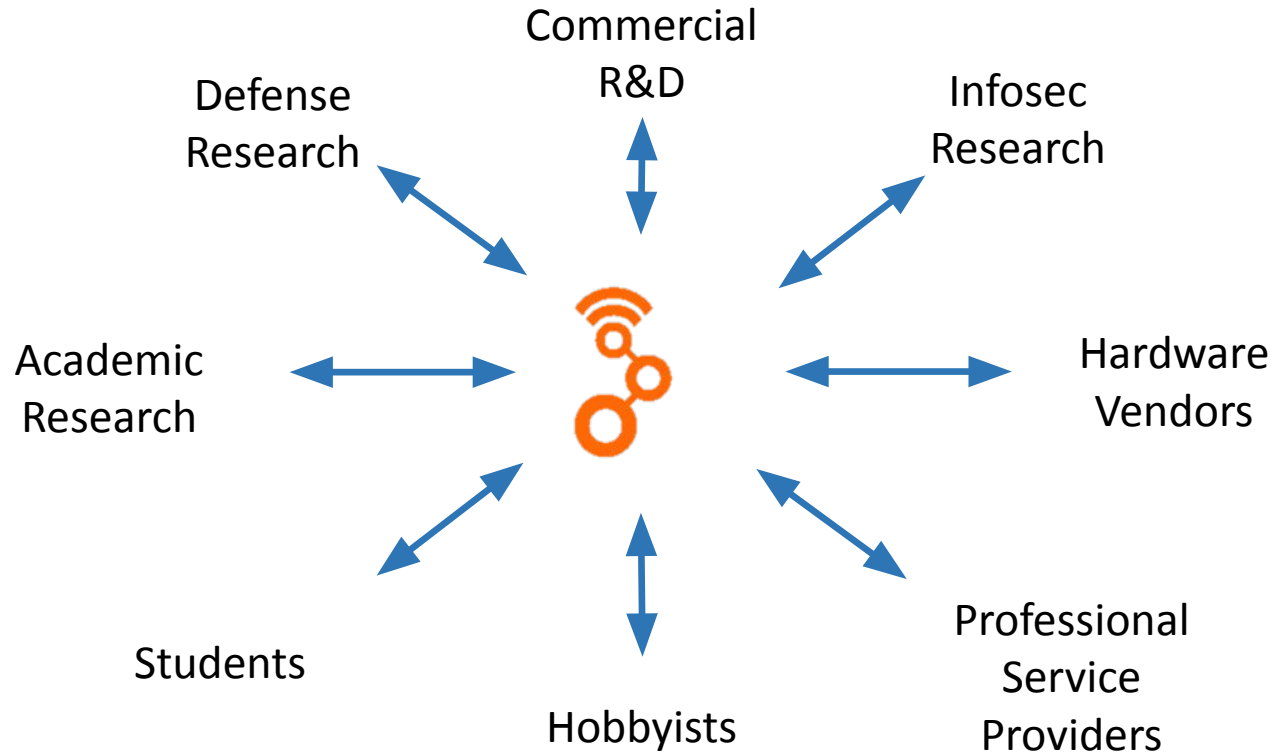
The Comprehensive GNU Radio Archive Network (CGRAN) is a free open source repository for 3rd party GNU Radio applications a.k.a Out Of Tree Modules that are not officially supported by the GNU Radio project.

[Browse~Checkout~Hack](#)

Name	Tags	Description	Repository
gr-eventstream	scheduler, streams, bursty	The event stream scheduler	Github

GNU Radio is used by



GNU Radio is an Ecosystem

- Active Open Source community since 2001
- PyBombs, OOTs
- GRCon since 2011
- GNU Radio Foundation
- FOSDEM SDR DevRoom
- GSoC, SoCIS, R&S Competition, SDR Academy
- GNU Radio Europe



What is a signal?

A signal is any measurable quantity that varies with time

It carries or conveys information

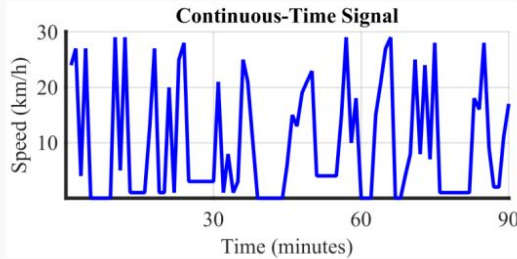
- Speech
- GPS
- ECG
- Stock prices
- Earthquake



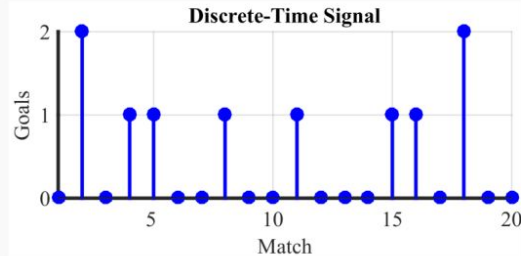
Continuous-Time vs Discrete-Time

- Continuous
 - Defined at every point
- Discrete
 - Only defined at discrete points in time

Player speed

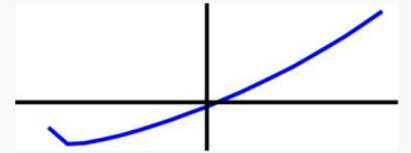
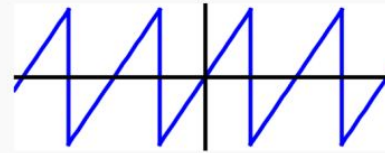
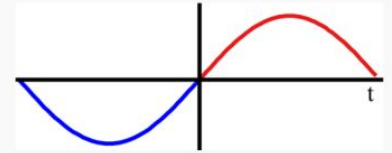
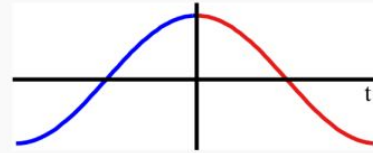
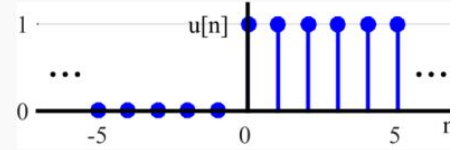
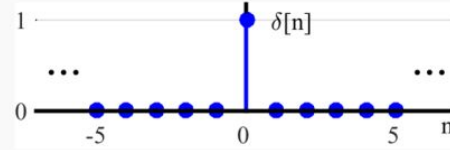


Player goals

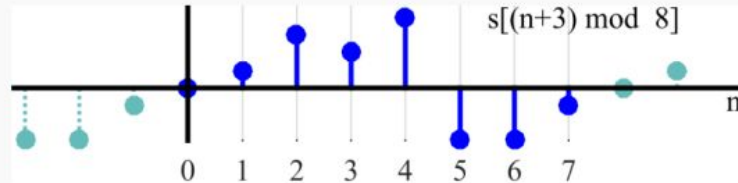
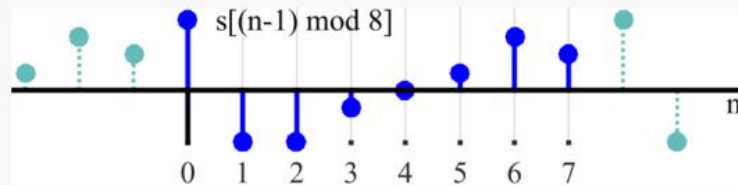
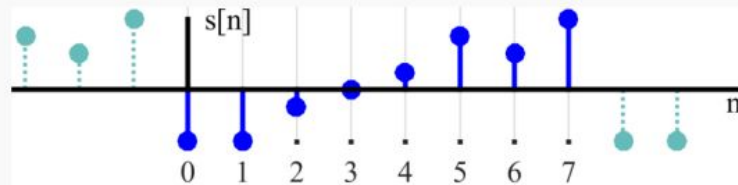
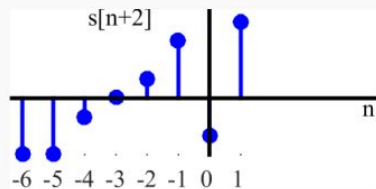
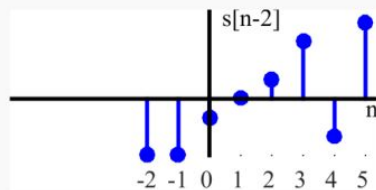
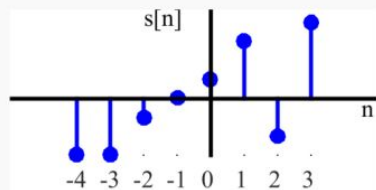


Basic Signals

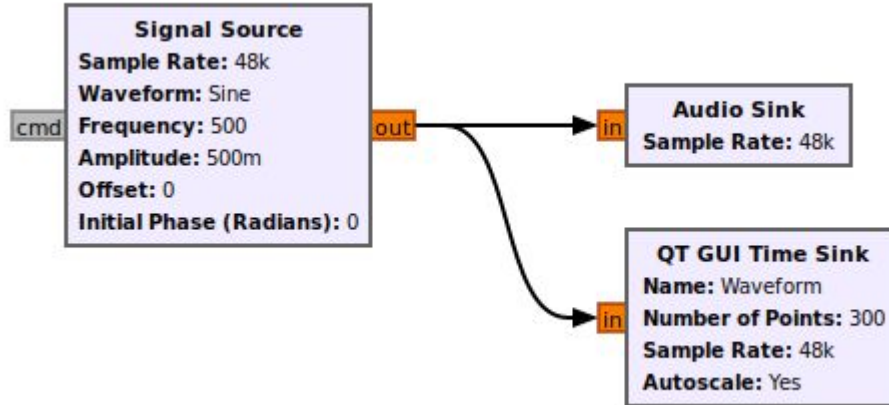
- Unit impulse
- Unit step
- Even/Odd
- Periodic/Nonperiodic



Shift in Time

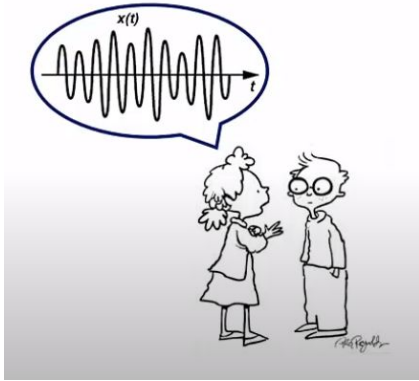


Signals in Time Domain

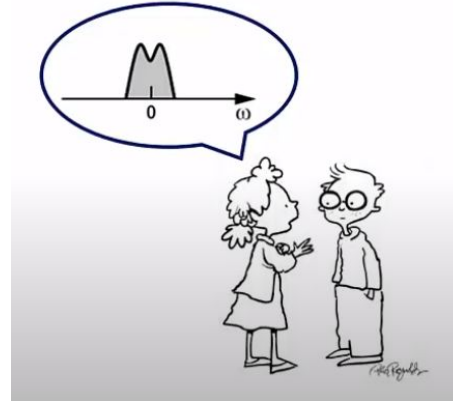


Time vs Frequency Domain

- The real world happens in the time domain

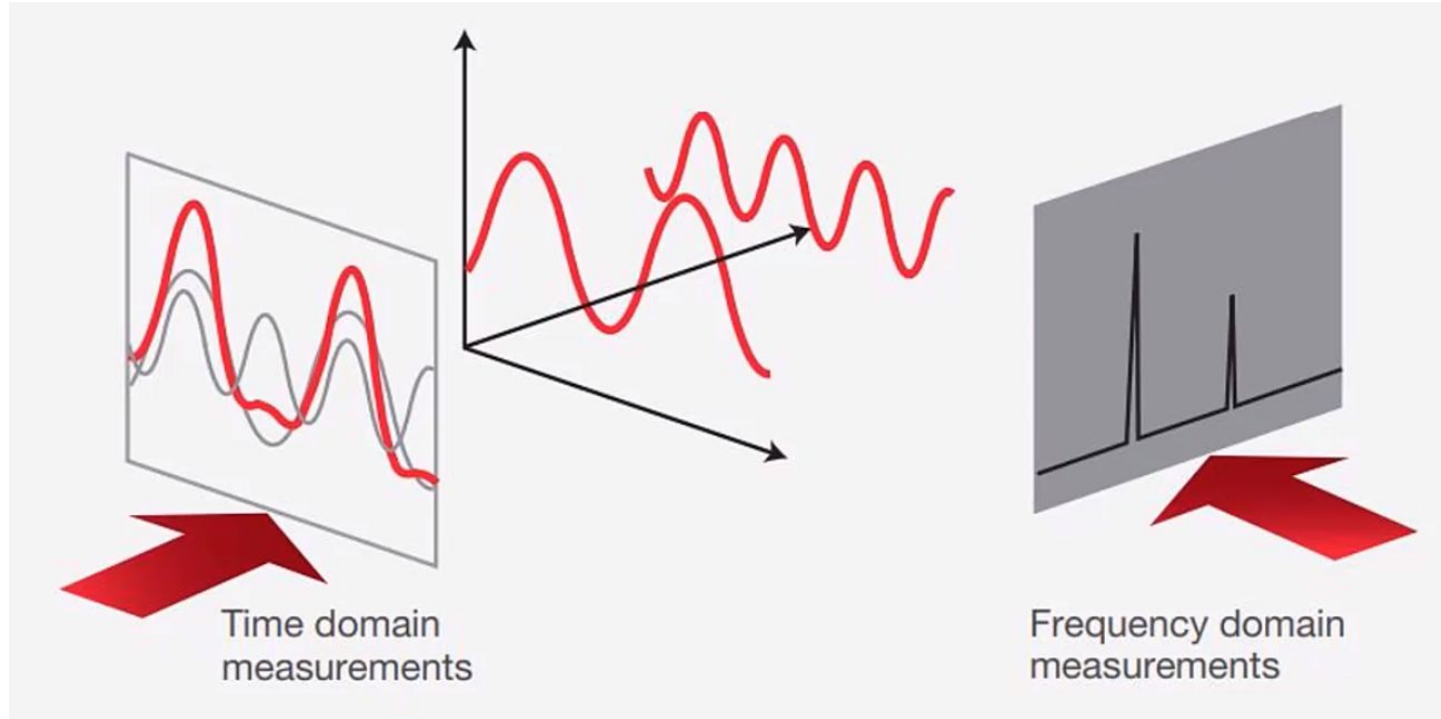


- Signals can be represented by frequency components



$$x(t) \longleftrightarrow \text{Fourier} \longleftrightarrow X(\omega)$$

Time vs Frequency Measurements



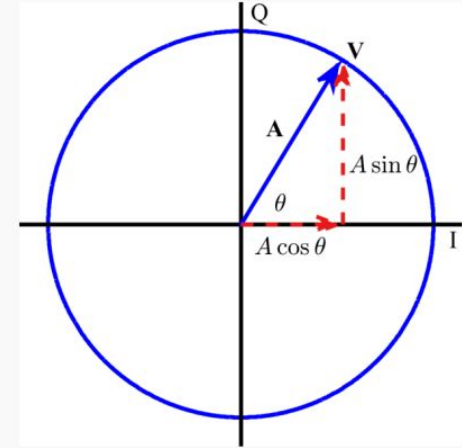
Complex Numbers

- Pair of real numbers
- I and Q parts
- Magnitude
- Phase

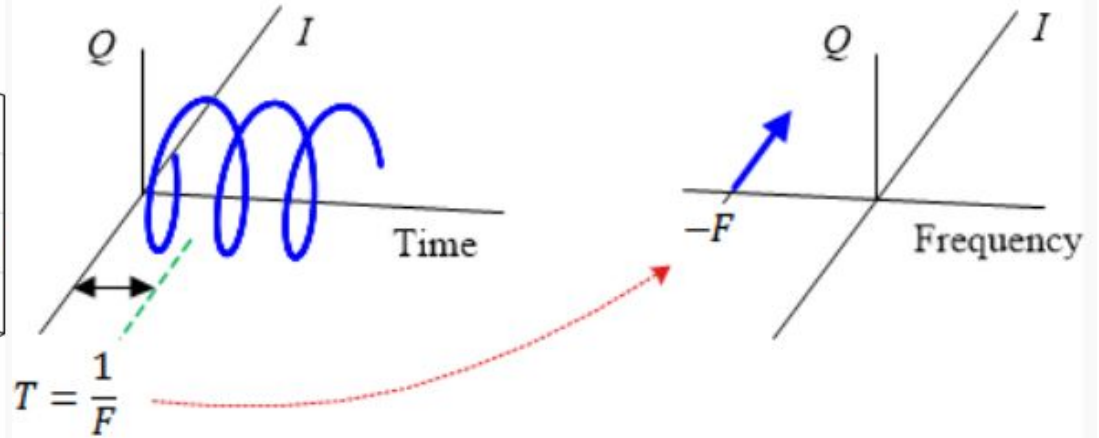
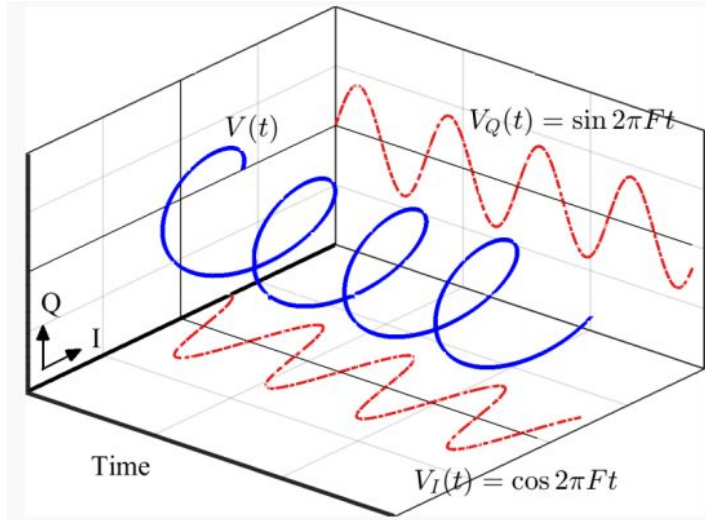
$$V_I = |V| \cos \angle V$$

$$V_Q = |V| \sin \angle V$$

$$|V| = \sqrt{V_I^2 + V_Q^2}$$



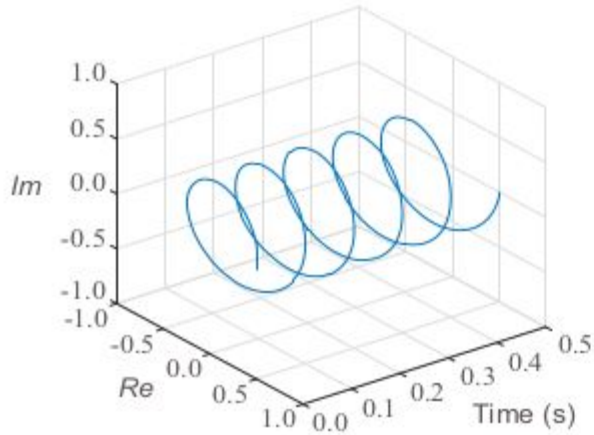
Complex Sinusoid



Positive/Negative Frequencies

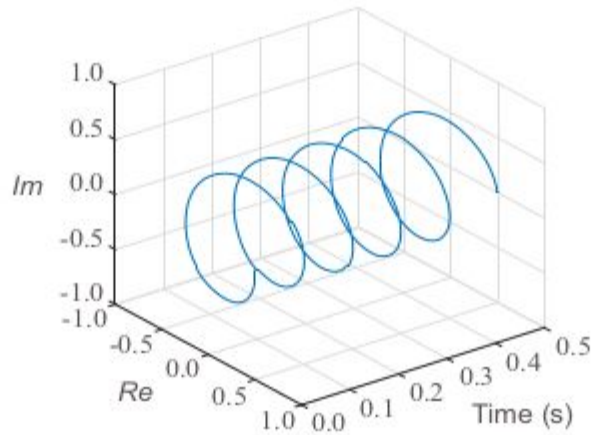
A

Positive-Frequency
Complex Exponential



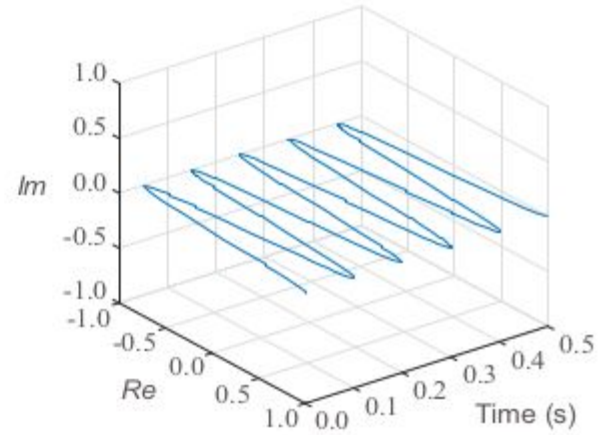
B

Negative-Frequency
Complex Exponential



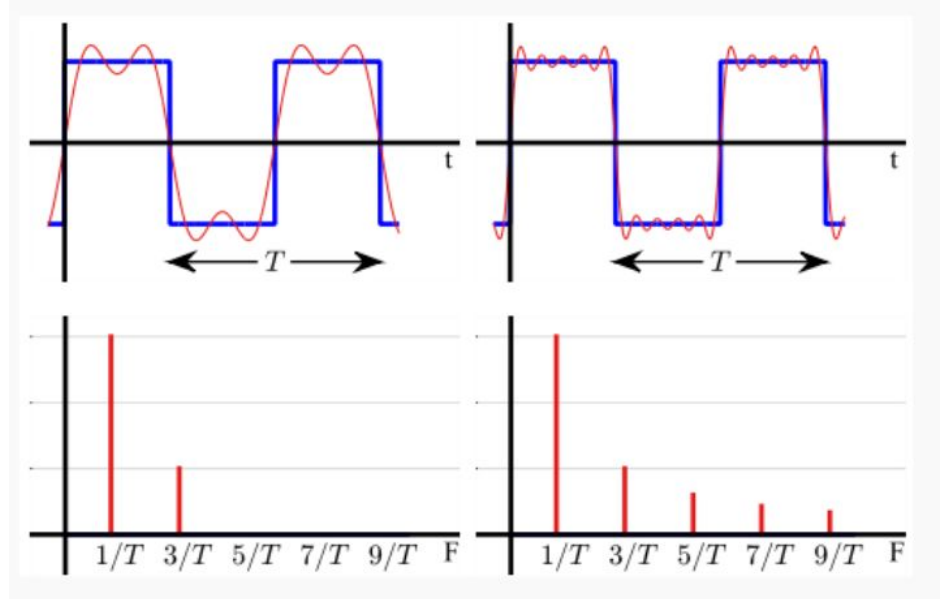
C

Sum of Complex
Exponentials (Cosine Wave)



Making Up a Signal

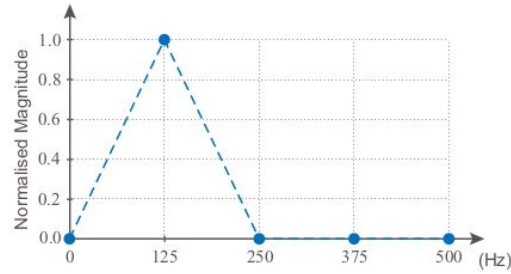
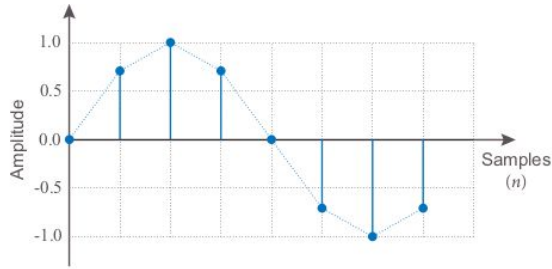
- Every signal is composed of sinusoids with different frequencies
- A better approximation is achieved with more sinusoids



Spectral Analysis

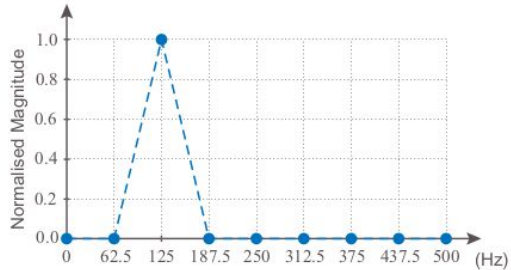
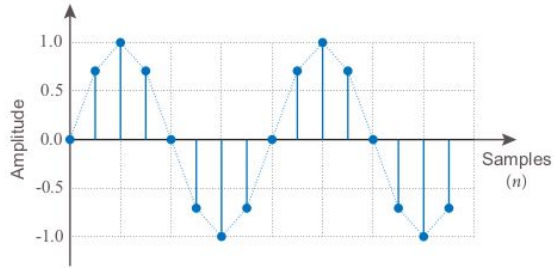
- **DFT** can be used to obtain the frequency representation of discrete-time waveforms
- **FFT** is not an approximation of the DFT; rather, it is the DFT and is effective when reducing computational complexity. We established that the FFT technique could only be used with DFT sizes that are a power of two.

125Hz sine wave sampled at 1kHz



$$N = 8$$

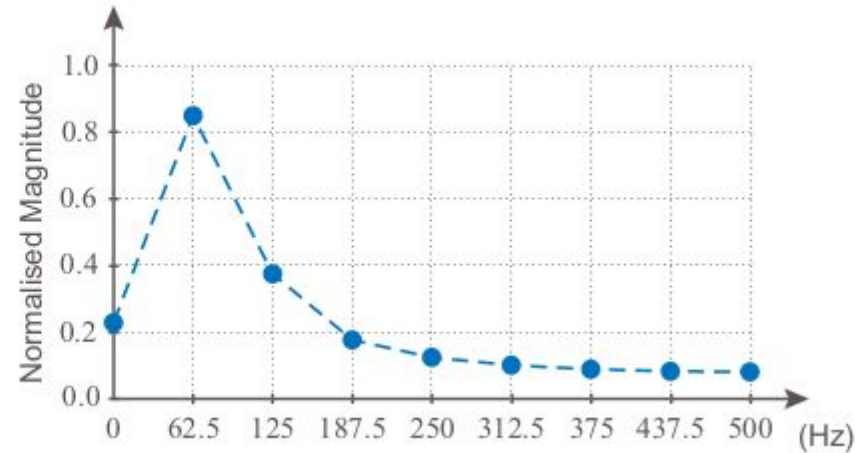
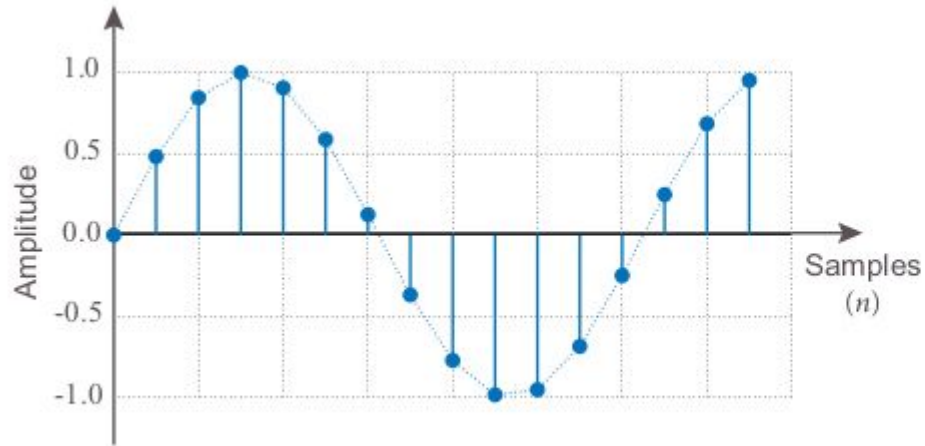
$$\Delta f = \frac{f_s}{N}$$



$$N = 16$$

Spectral Leakage

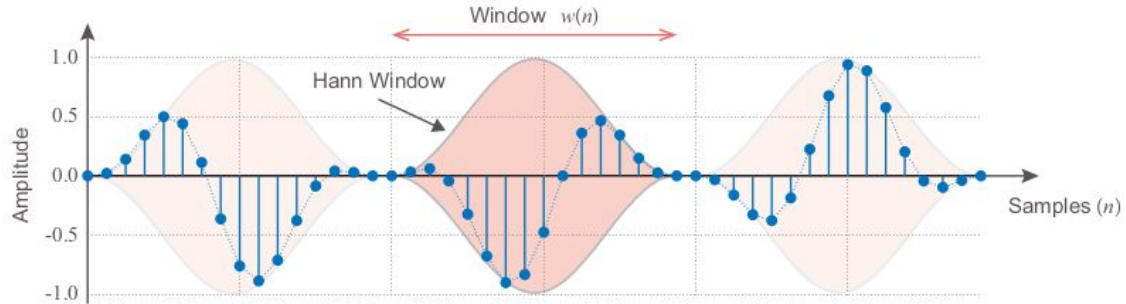
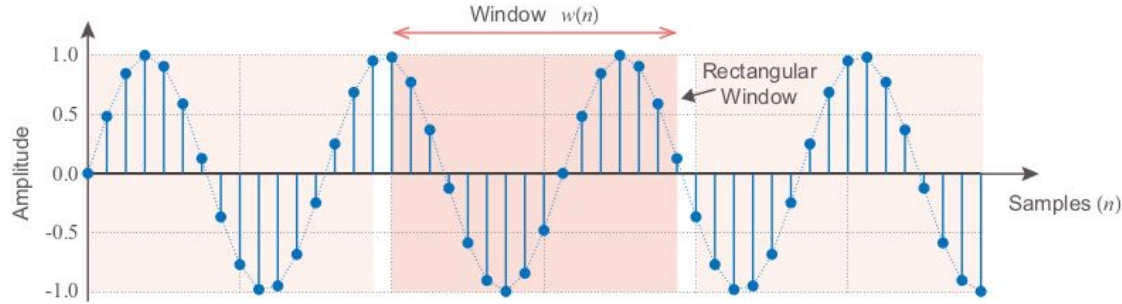
Now have a look at a discrete sine wave with a frequency of 80Hz sampled at 1kHz.



Windowing

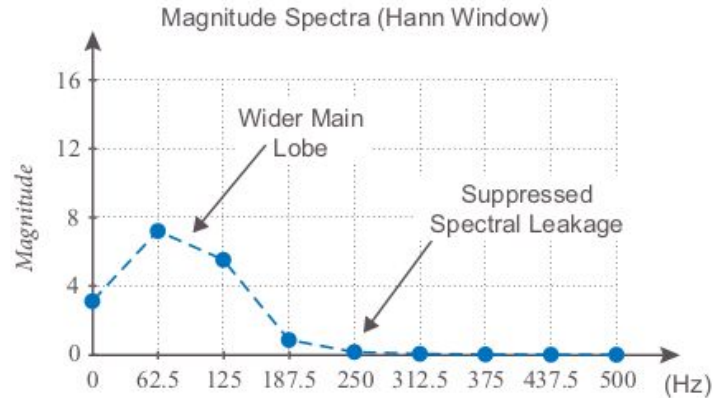
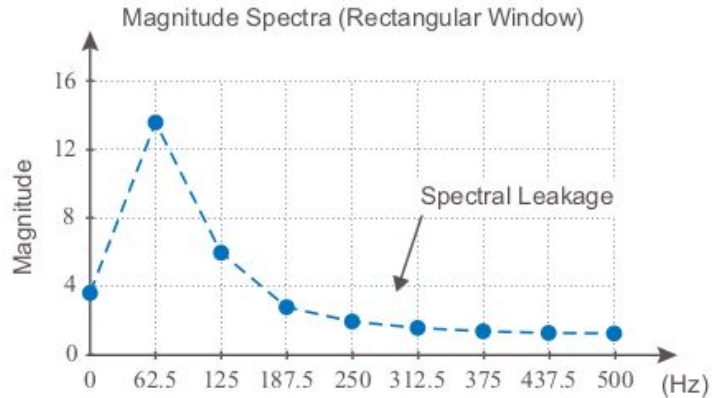
- We can reduce the effect of spectral leakage by applying particular windows to a discrete waveform before using the DFT
 - Hamming,
 - Hann,
 - Blackman-Harris and
 - Bartlett.

A Hann window applied to a discrete sine wave of 80Hz



A Hann window applied to a discrete sine wave of 80Hz

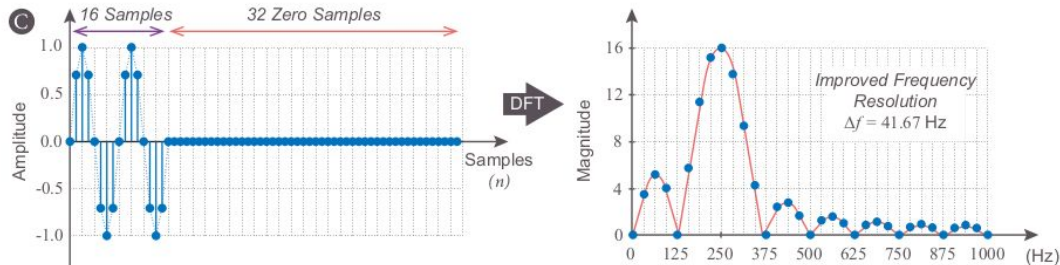
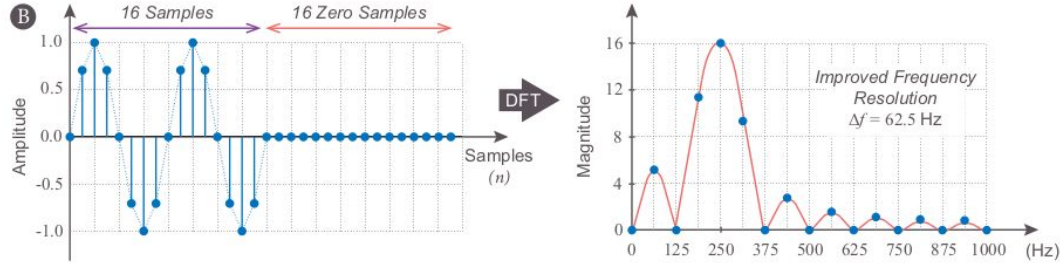
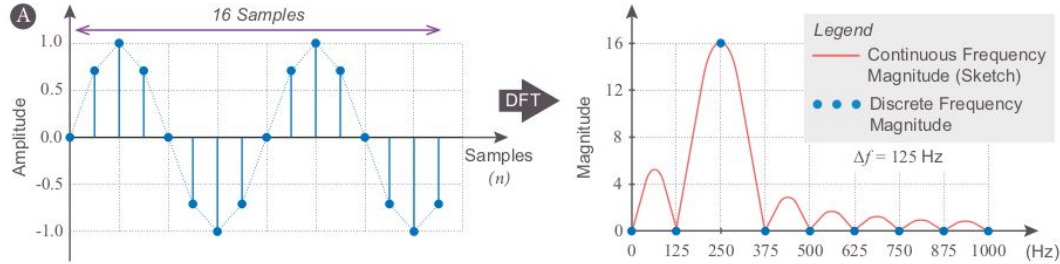
- Tapered windows can reduce spectral leakage in the DFT.
- However, there are some caveats.
 - Windowing has the effect of widening the main lobe of the peak frequency.
 - However, the side lobes that cause spectral leakage are reduced.



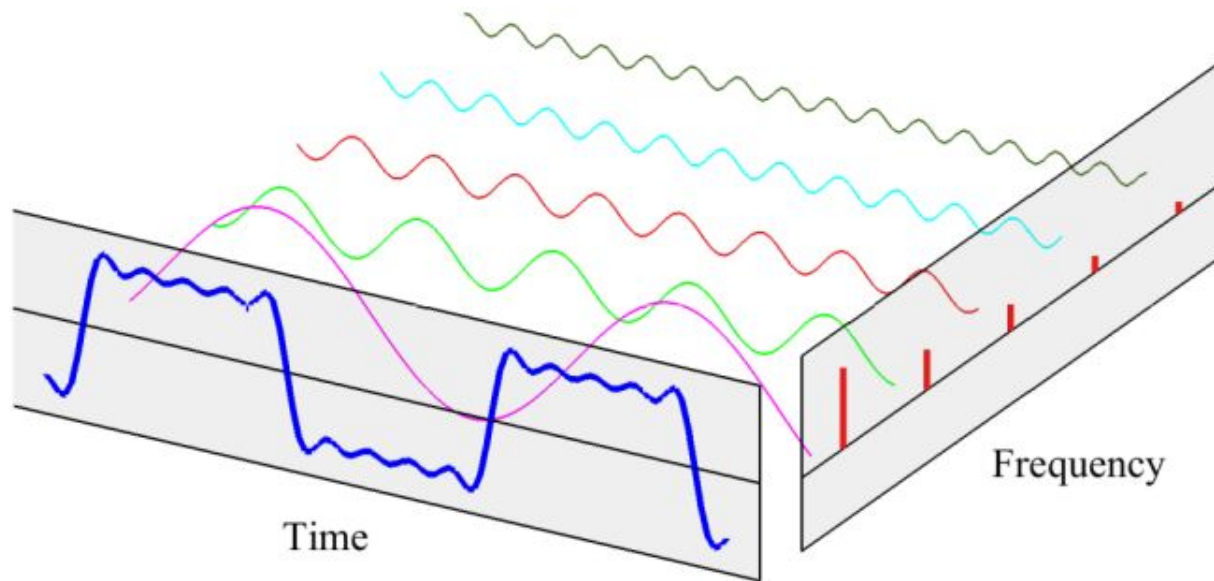
Zero Padding

- Zero padding is a technique that involves inserting zero-valued samples at the end of a discrete waveform to improve the frequency resolution of the DFT plot.
- The effect of zero padding is essentially an interpolation of the frequency sample points in the DFT and as such no extra 'information' is created on the signal

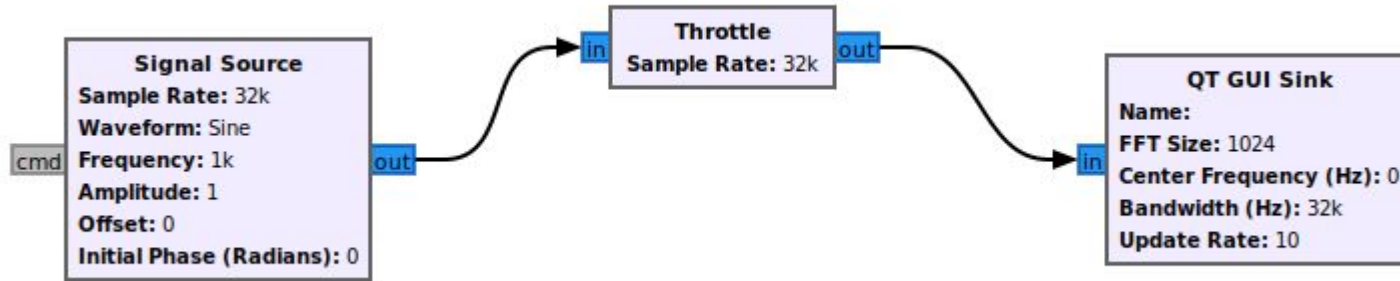
250 Hz Sine wave sampled at 2k Hz



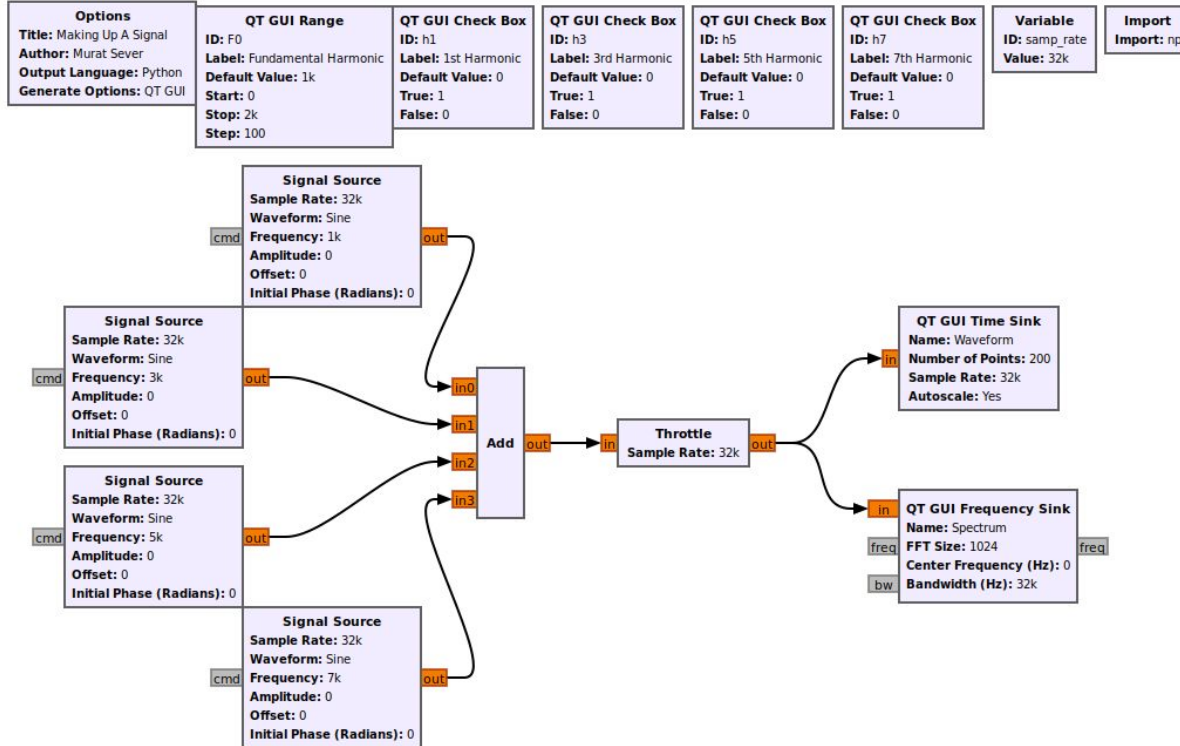
Squarewave



Exploration of Signals in Frequency Domain



Making Up a Signal

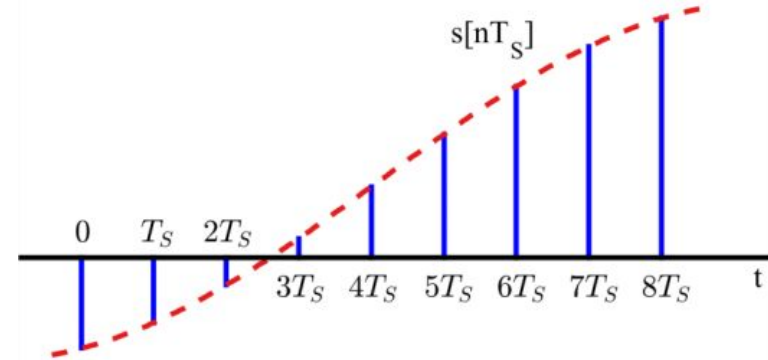


Using GNU Radio from Python

- Generate Python from GRC Flow graph
- Invoke directly from the Linux command line:
 - `$ python3 makingupasignal.py`

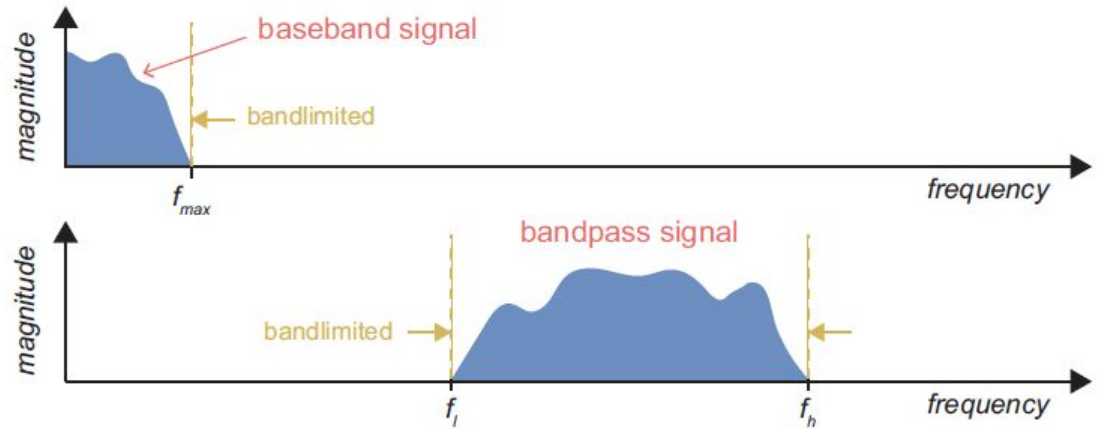
Sampling

- Communication signals are continuous-time
- We (ADCs) take samples at regular times
- T_s is sampling period
- F_s is sampling frequency



Baseband & Bandpass

- Baseband: Information signal
- Bandpass: Communication signal

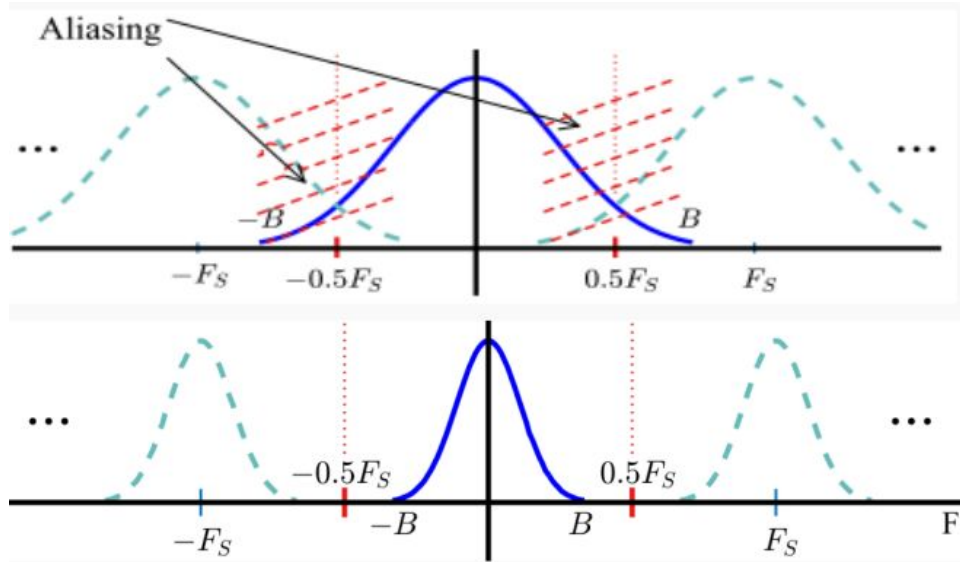


Nyquist Sampling Theorem

- The **Nyquist Sampling Theorem** states that a baseband, bandlimited signal must be sampled at **greater than twice the bandwidth** present in the signal, i.e.
 - $f_s > 2 * f_{max}$
 - $f_s > 2 * (f_{high} - f_{low})$

Aliasing

- Sampling produces aliases (spectral replicas)
- To prevent aliasing F_s must satisfy $F_s > 2 * BW$

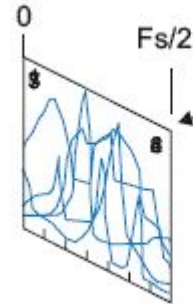
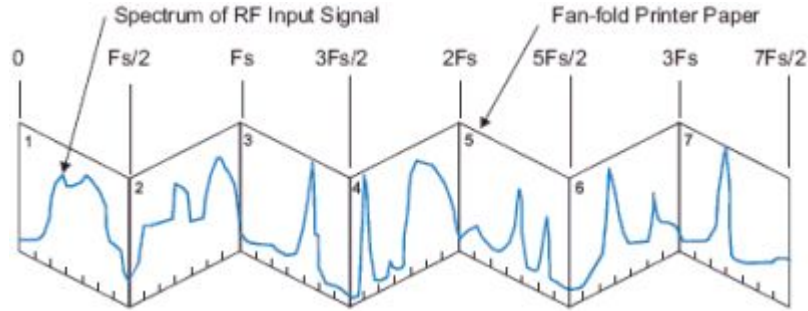


Nyquist Zones

- Partitions of bandwidth $0.5f_s$ in the frequency domain
- Any signal components present in higher Nyquist Zones are 'folded' down into the 1st Nyquist Zone as a result of aliasing



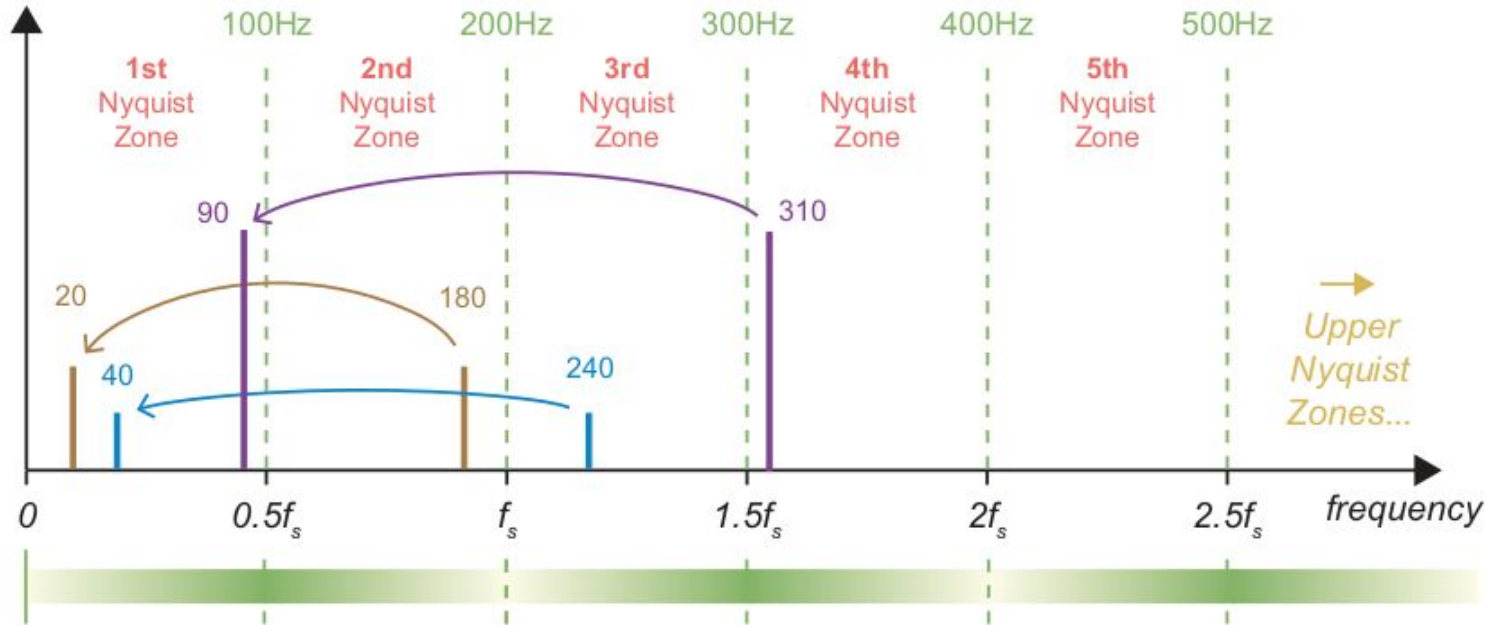
Folded Spectrum View



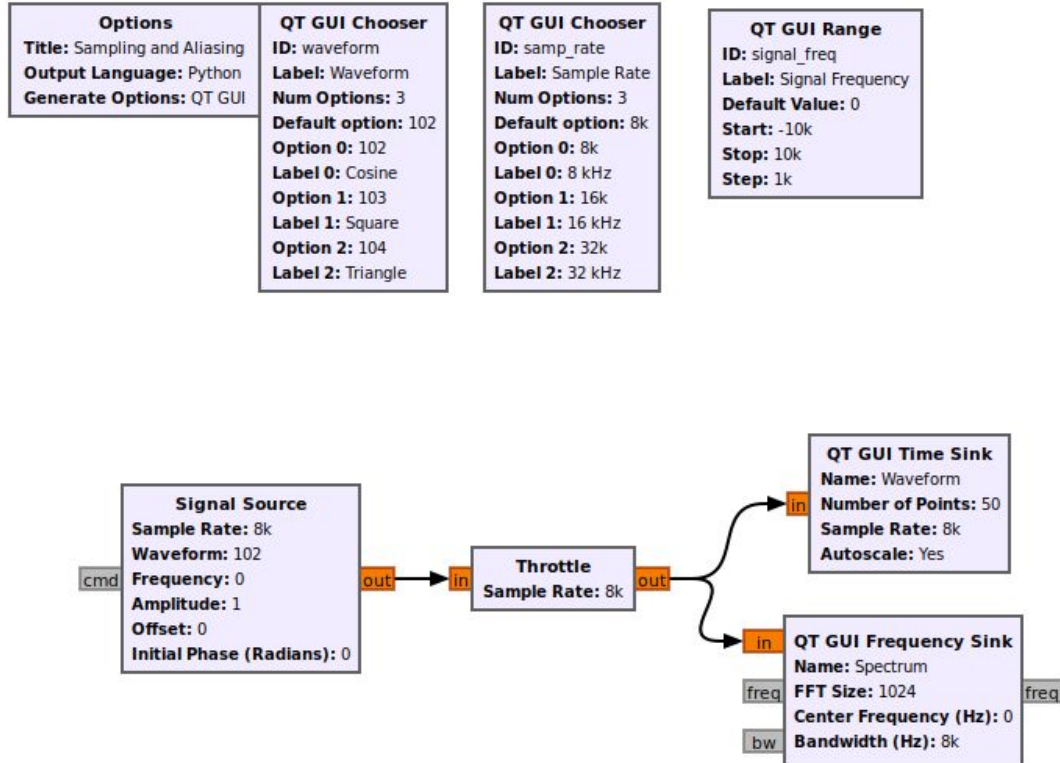
After sampling, all out of band signals and noise are folded into the band between 0 and $F_s/2$



Examples of aliasing with reference to Nyquist Zones

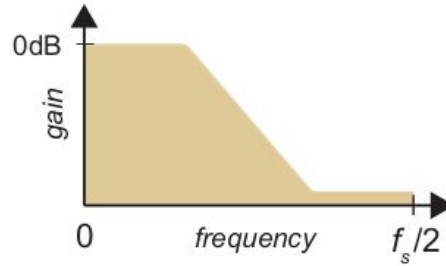


Sampling and Aliasing

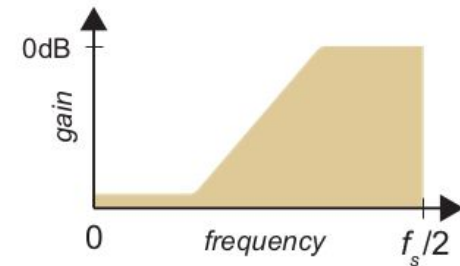


Digital Filters

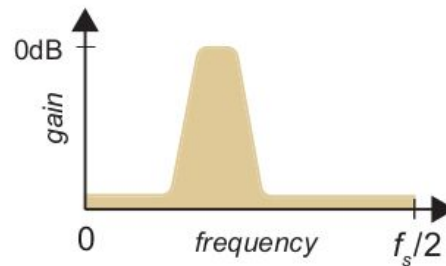
- A filter modifies the frequency contents of an input signal
- Types
 - LPF
 - HPF
 - BPF
 - Notch



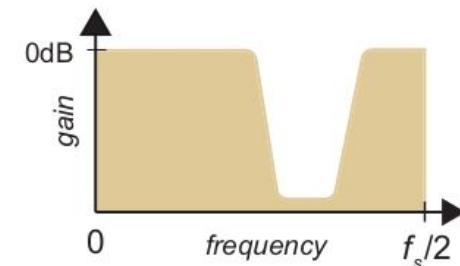
(a) lowpass



(b) highpass

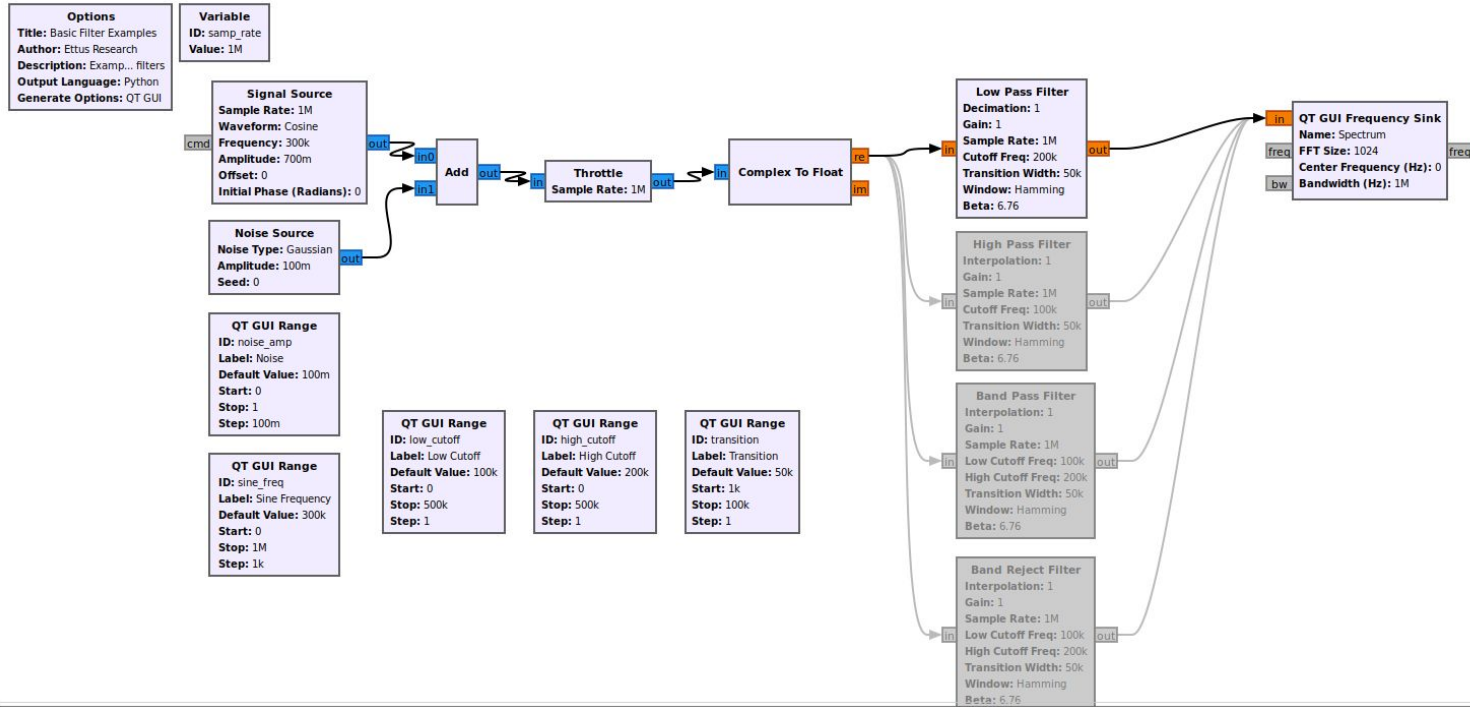


(c) bandpass



(d) bandstop

Filters Using GNU Radio

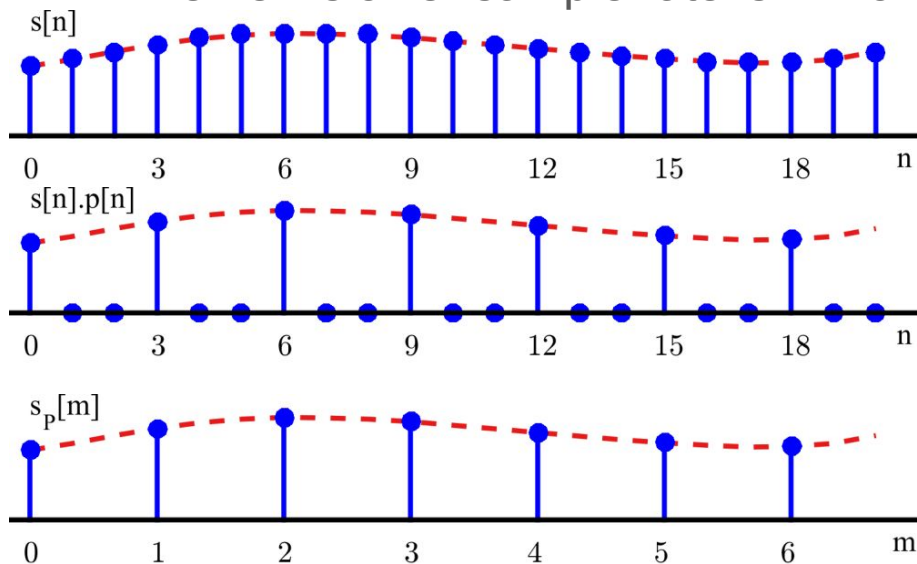


Multirate Signal Processing

- Multirate operations are required to change the sampling rate in a DSP system to optimise computational efficiency
- Some example scenarios
 - To match the sampling rates of two signal paths that will be combined
 - To adjust the sampling rate closer to Nyquist when the signal bandwidth changes
 - To match the sampling rate of an external interface, such as a DAC
 - To ease analogue anti-alias or image-rejection filter requirements

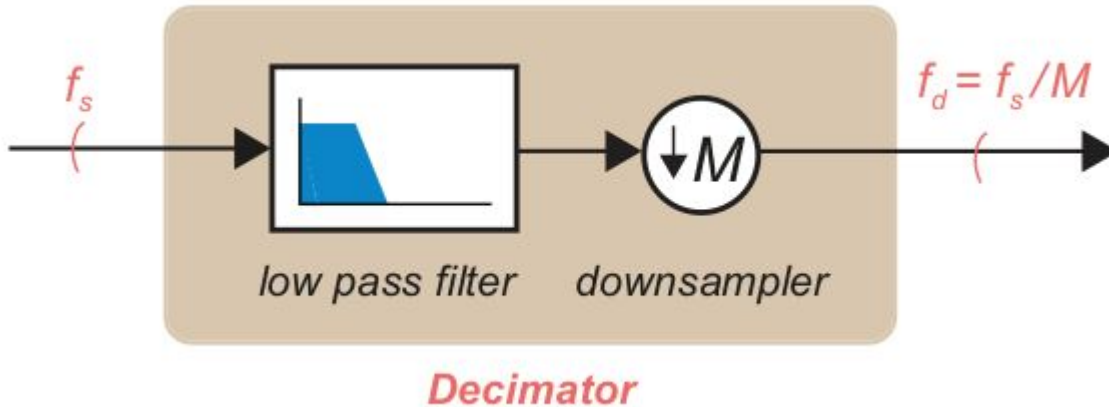
Decimation

- Reducing the sample rate by an integer factor
- Retain every P th sample and discard the remaining samples
- The new slower sample rate is $1/P$ of the original faster sample rate



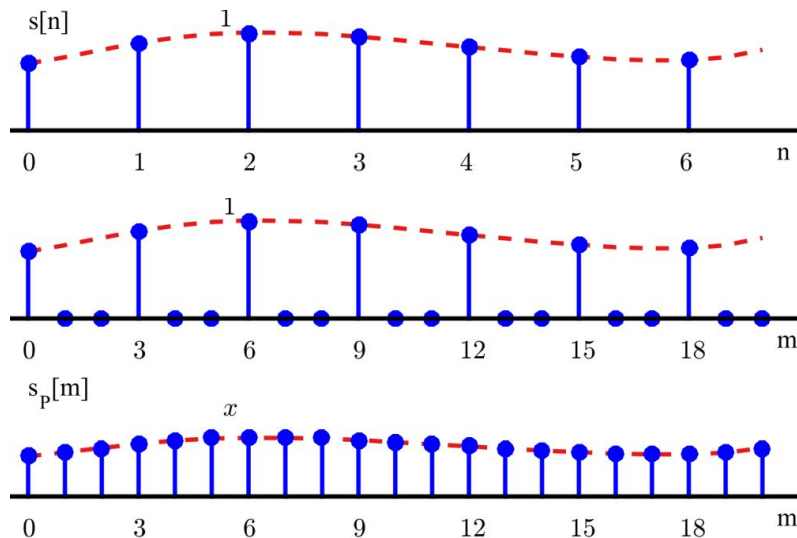
Decimation

- Decimation involves two processes:
 - anti-alias low pass filtering, followed by
 - downsampling



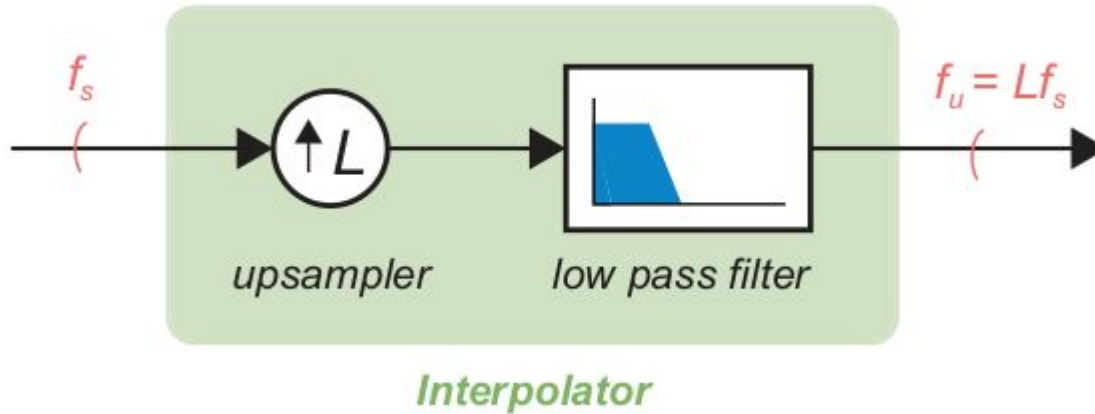
Interpolation

- Increasing the sample rate by an integer factor
- Insert $P - 1$ zeros between the original input samples and interpolate
- The new faster sample rate is P times the original slower sample rate



Interpolation

- An interpolator is composed of
 - an upsampling operation, followed by
 - a low pass image rejection filter

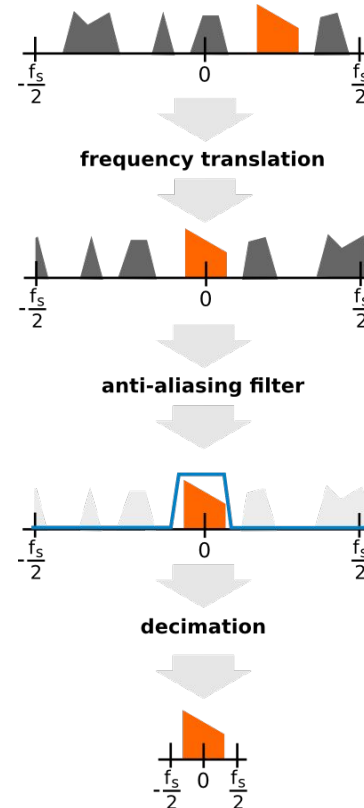


Other Multirate Operations

- There are other types of operation to be aware of, beyond simple decimation and interpolation by integer factors
- Resampling a signal by a **rational fraction**
 - If the sampling rate is to be changed by the ratio of two integers, e.g. a rate change from 100 MHz to 150 MHz could be expressed as $R = 3 / 2$. Rational fractional rate changes can be achieved using a **cascade** of an interpolator and decimator, e.g. $L = 3$ and $M = 2$ in this example. The resulting structure can be optimised using polyphase methods.
- Resampling a signal by an **irrational fraction**, or by a factor that changes over time
 - Where there is no convenient integer-based expression for the resampling ratio, or where it is dynamic, a different type of approach is required. Popular methods include highly oversampled polyphase filters, and Farrow structures.

Frequency Xlating FIR Filter

- Frequency Xlating FIR Filter is a block that:
 - performs frequency translation on the signal,
 - downsamples the signal by running a decimating FIR filter on it.
- It can be used as a channelizer:
 - it can select a narrow bandwidth channel from the wideband receiver input.



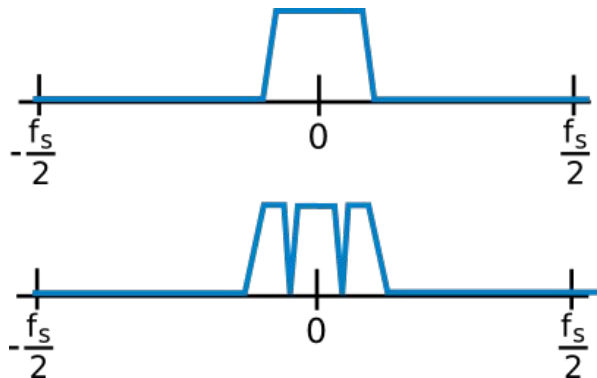
Suppose this is the stations in FM radio example!

Our aim is to select only one channel

Frequency Xlating FIR Filter

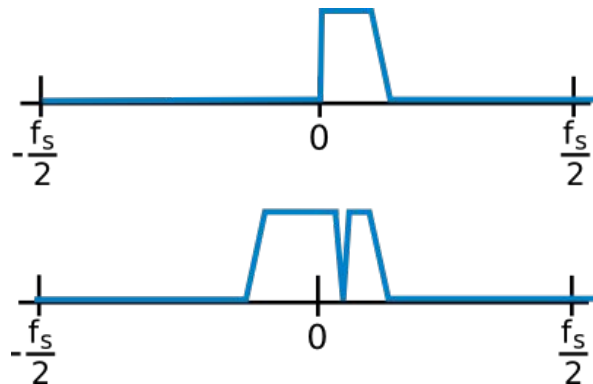
- If you have Real taps, then your FIR filter will be symmetric in the frequency domain.

```
firdes.low_pass(1,samp_rate,samp_rate/(2*decimation), transition_bw)
```



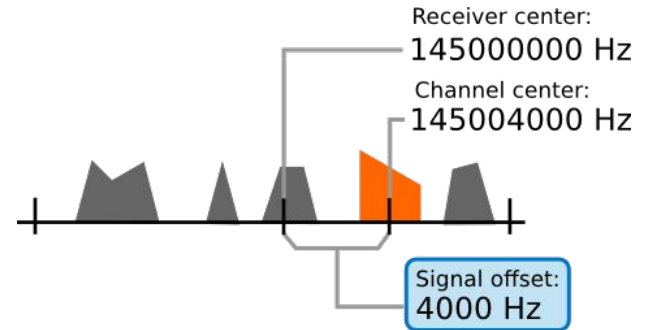
- If you have Complex taps, then your FIR filter will not have to be symmetric in the frequency domain.

```
firdes.complex_band_pass(1, samp_rate, -samp_rate/(2*decimation), samp_rate/(2*decimation), transition_bw)
```



Frequency Xlating FIR Filter

- **Decimation**: the integer ratio between the input and the output signal's sampling rate.
- Example:
 - Input sample rate = 240000
 - Decimation factor = 5
 - Output sample rate = $240000 \div 5 = 48000$
- **Center frequency**: the frequency translation offset frequency.
- In practice, it is the frequency offset of the signal of interest to be selected from the input.



Frequency Xlating FIR Filter

