

# GNU-Radio

**Presentasjon/demo for LA7M, Modum-gruppen**

# Hva og Hvordan



**GNU Radio** (<https://www.gnuradio.org>) er et gratis og åpen kildekode-programvarerammeverk for programvare definert radio (SDR) og digital signalbehandling.

Det lar deg bygge radiosystemer ved å koble sammen ferdige "blokker" i en grafisk eller programmatisk arbeidsflyt (flytgraf), slik at du raskt kan prototype, eksperimentere og til og med utvikle systemer som kan kjøres i produksjon.

## Installere

Instruksjoner for installasjon av GNU-Radio:

<https://wiki.gnuradio.org/index.php/InstallingGR>

Bruk RadioConda for Mac OS og Windows:

<https://github.com/radioconda/radioconda-installer>

## Kjøre (Mac OS)

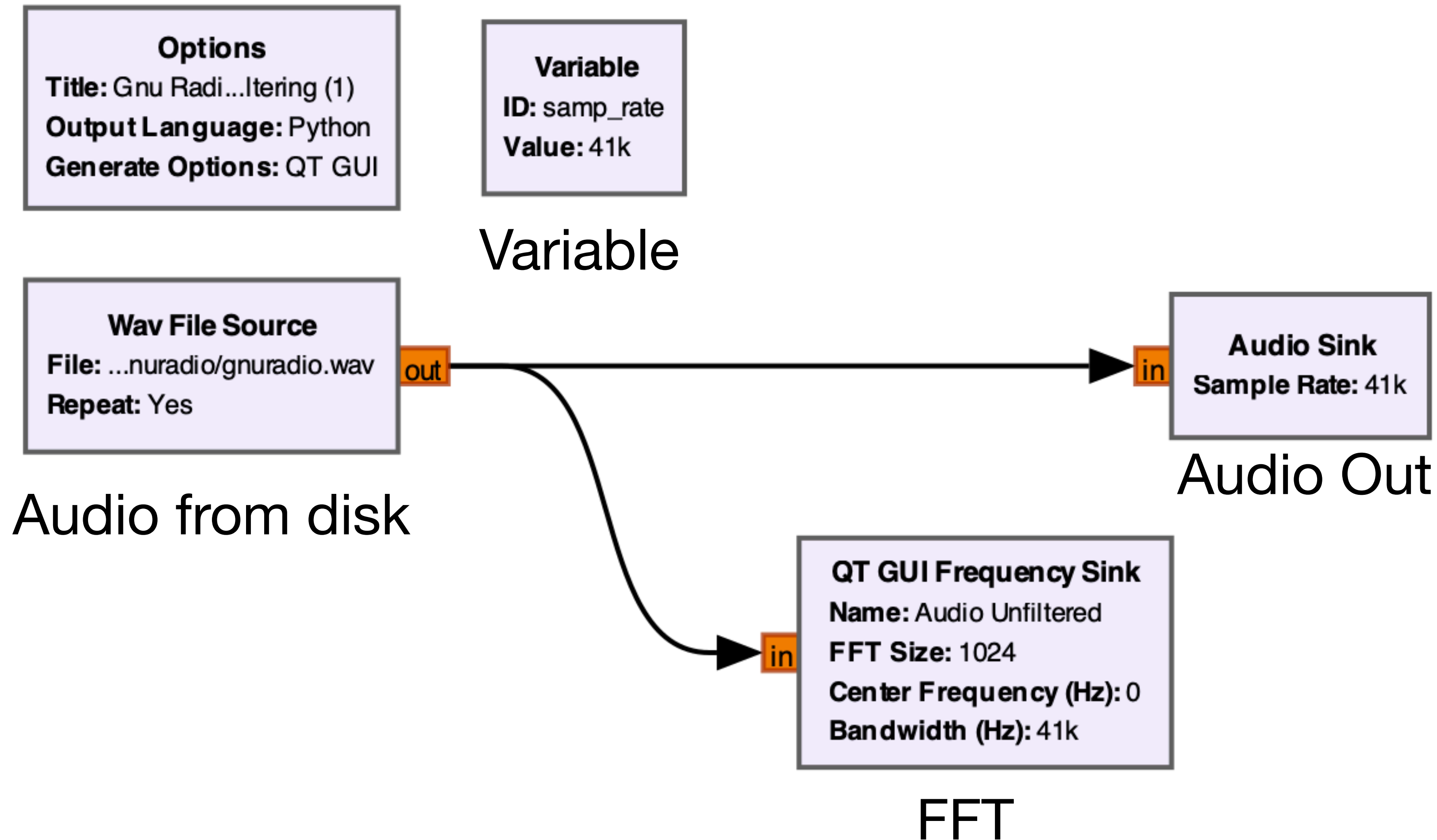
A screenshot of a macOS terminal window. The title bar shows the window name "paul — gnuradio-companion — 120x40". The terminal content shows the following commands and output:

```
[(base) paul@m1-air ~ % conda activate gnuradio
[(gnuradio) paul@m1-air ~ % gnuradio-companion
<<< Welcome to GNU Radio Companion 3.10.11.0 >>>
```

Merk: Nyeste versjon er GR4 (desember 2025). Jeg kommer fortsatt til å bruke versjon 3.10 i dag.

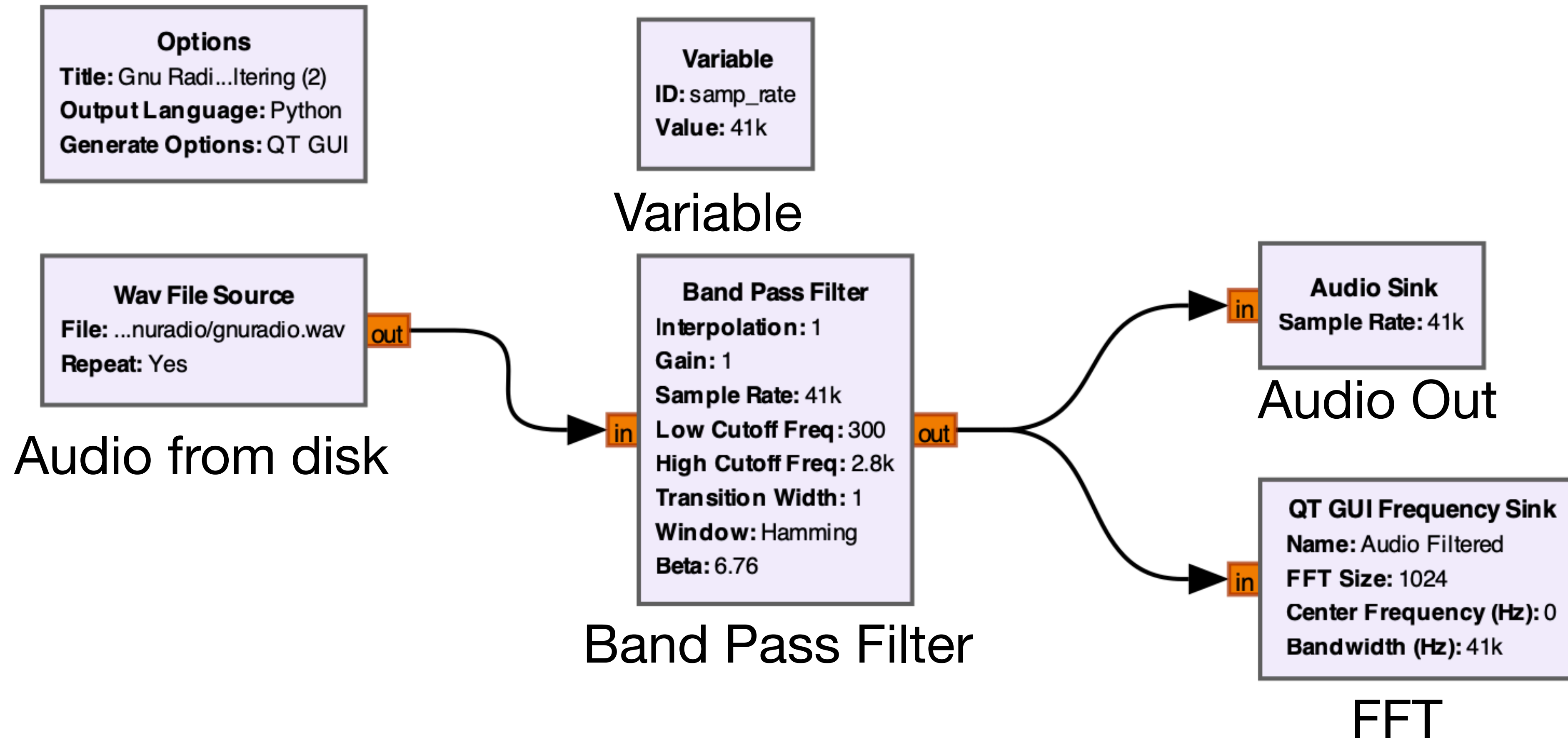
# Audio (1)

## Simple Audio player with Frequency Diagram (FFT)



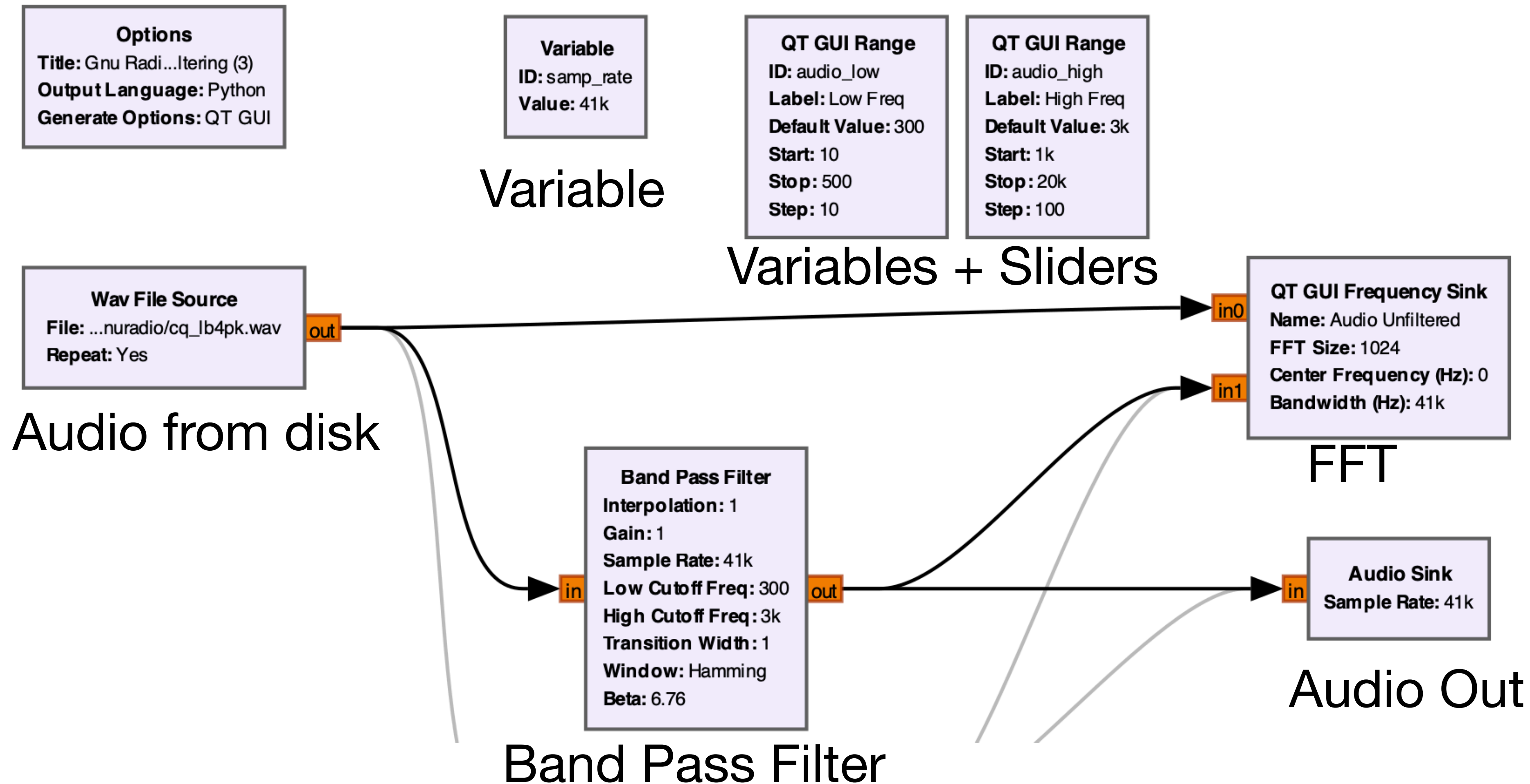
# Audio (2)

## Simple Audio player adding a Filter



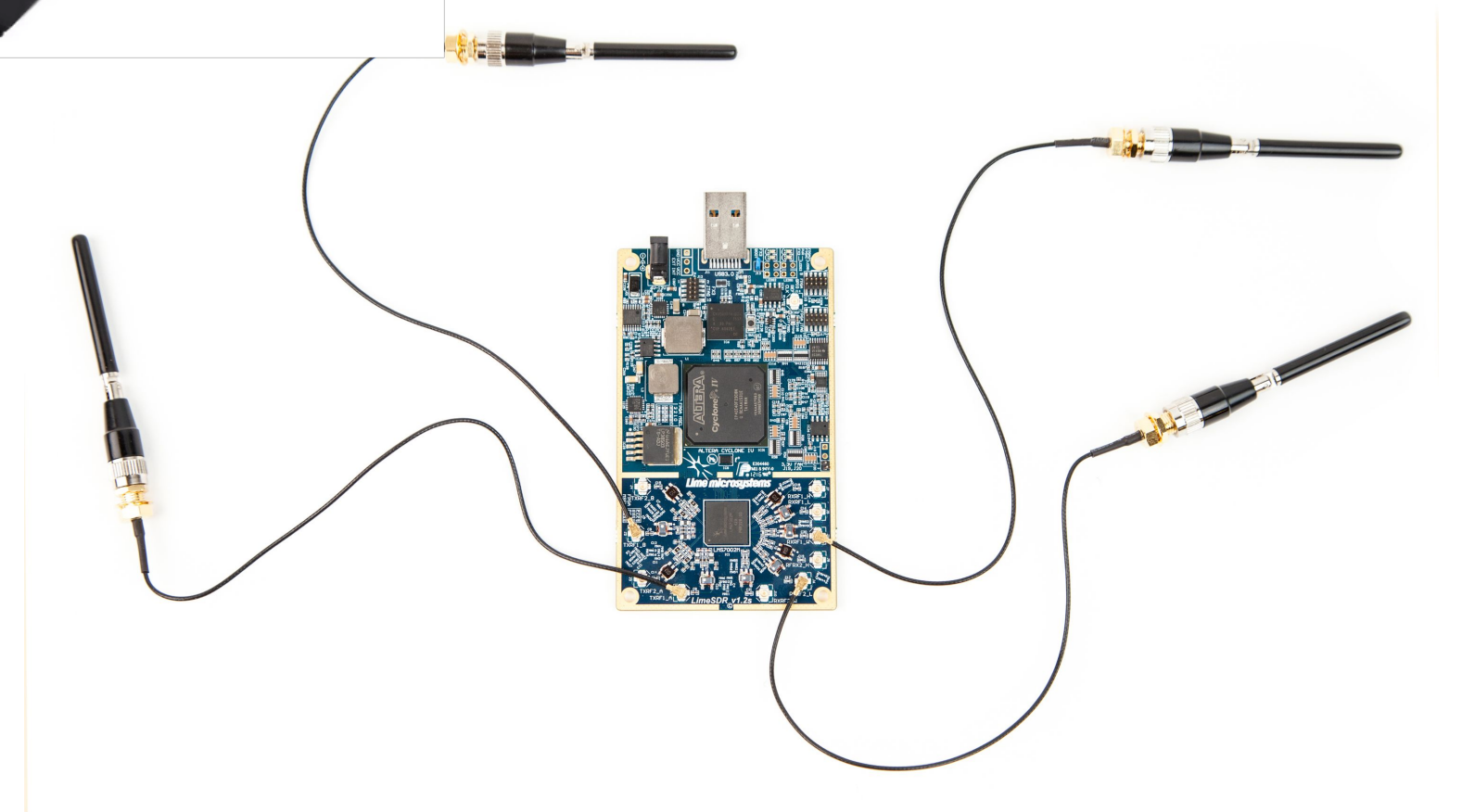
# Audio (3)

## Simple Audio player with Adjustable Filter



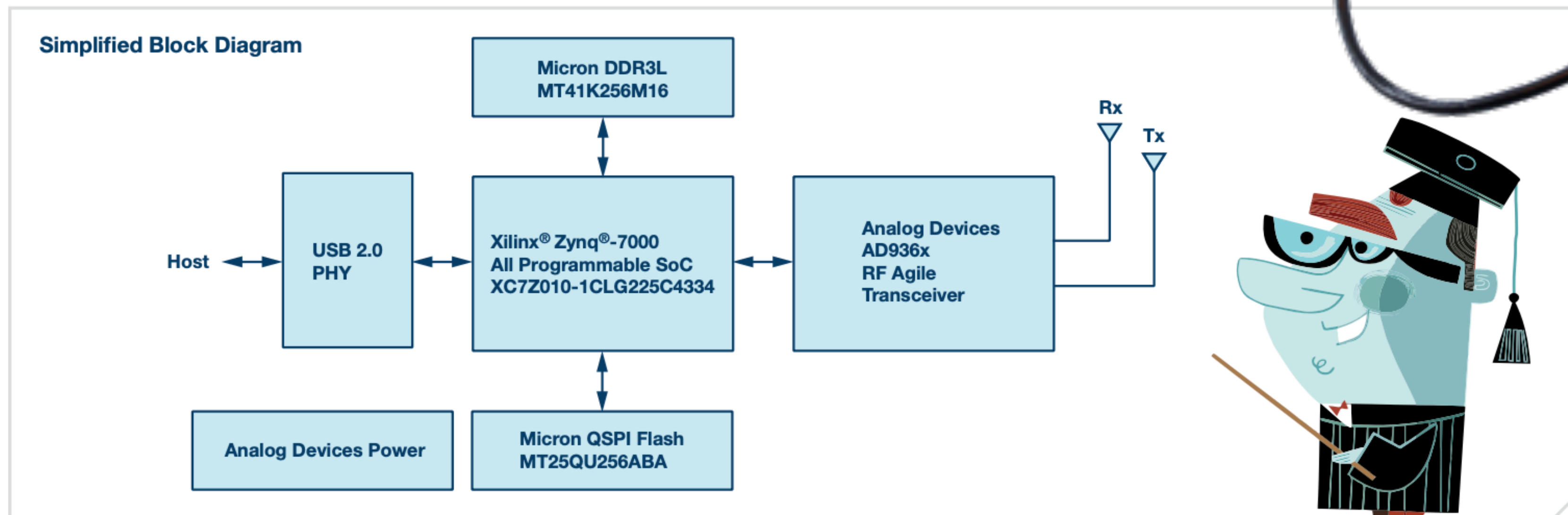
# Støtter radioer i GNU-Radio

- Analog Device ADALM-PLUTO
- Ettus Research USRP™ Devices
- Fairwaves XTRX
- Fairwaves UmTRX
- Funcube Pro+ Dongle
- KerberosSDR
- Great Scott Gadgets HackRF
- Lime SDR
- Microtelecom Perseus
- New Horizons NH7020
- Novena + Myriad-RF module
- Nuand BladeRF
- Nuand BladeRF 2.0
- rtl-sdr TV tuners
- SDRplay RSP family of SDR receivers
- Softrock-like Radio frequency interfaces



# ADALM-PLUTO

The **ADALM-PLUTO Active Learning Module (PlutoSDR)** is an easy to use tool available from Analog Devices Inc. ([ADI](#)) that can be used to introduce fundamentals of **Software Defined Radio (SDR)**



325 MHz to 3.8 GHz  
12-bit ADC and DAC

Modified :

- 70M to 6GHz
- 2Rx, 2Tx

# ADALM-PLUTO

## AD9361 Block Diagram

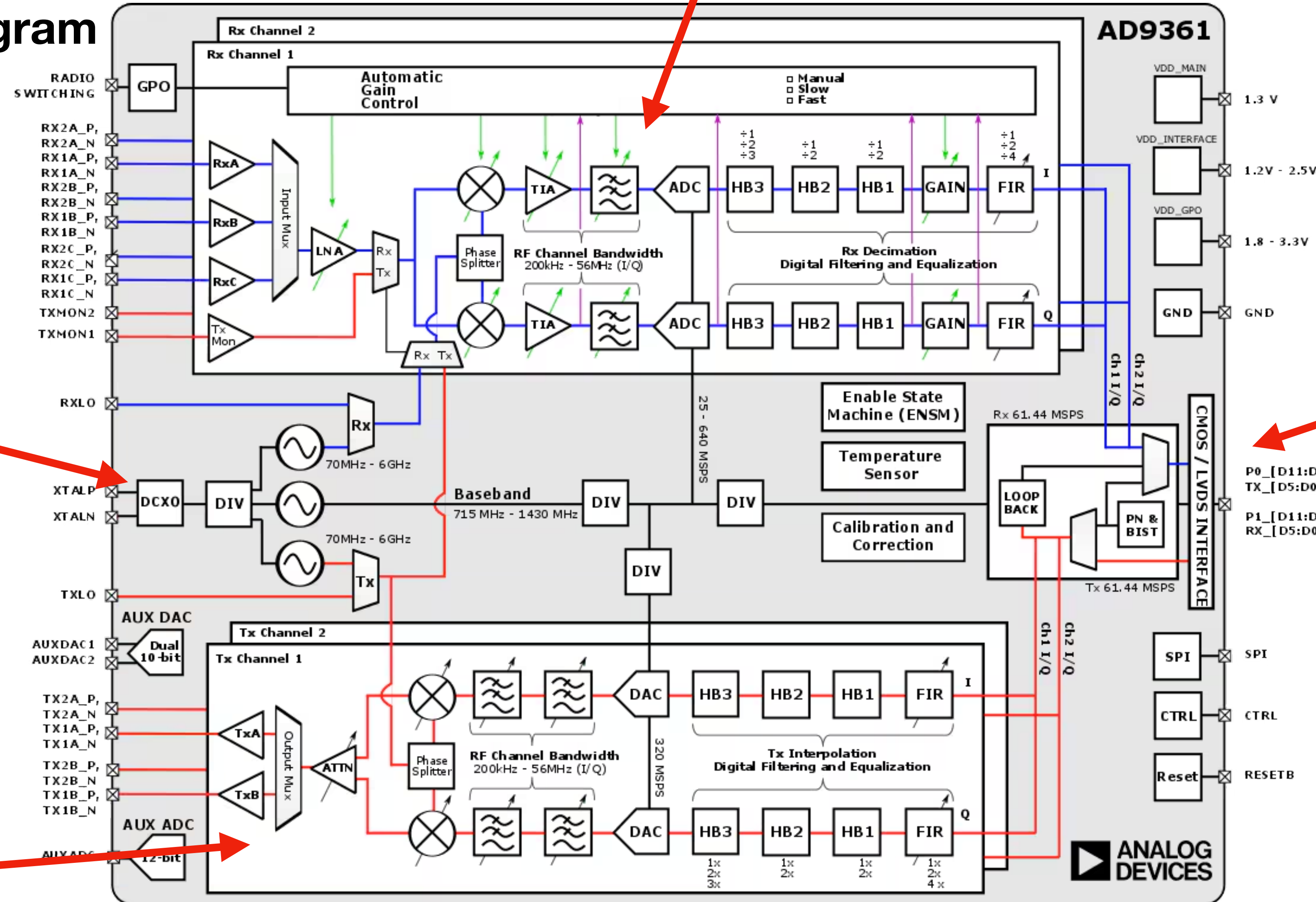


Mottaker

Oscillator

Sender

Digital interface



# ADALM-PLUTO in GnuRadio

## Enkel å bruke

Properties: PlutoSDR Source

General Advanced Filter Documentation

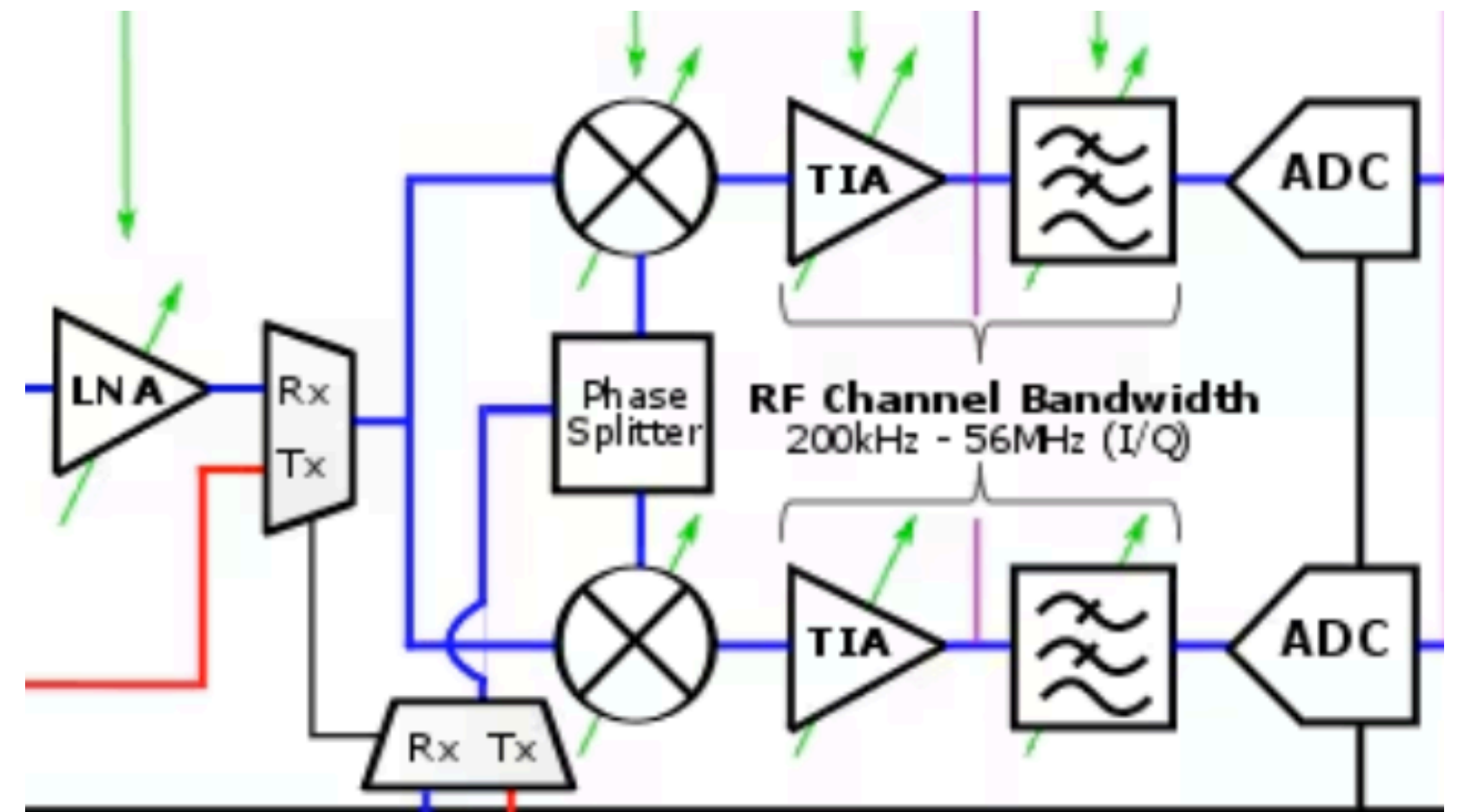
Output Type	Complex float32 ▾	
IIO context URI	<input type="text"/>	[string]
LO Frequency	LO_Freq	[int]
Sample Rate	int(samp_rate)	[int]
Buffer size	2*32768	[int]
Quadrature	True ▾	[bool]
RF DC Correction	True ▾	[bool]
BB DC Correction	True ▾	[bool]
Gain Mode (RX1)	Hybrid ▾	
Packet Length Tag	packet_len	[string]

OK Cancel Apply

# Quadrature Receiver

## What is a Quadrature Receiver?

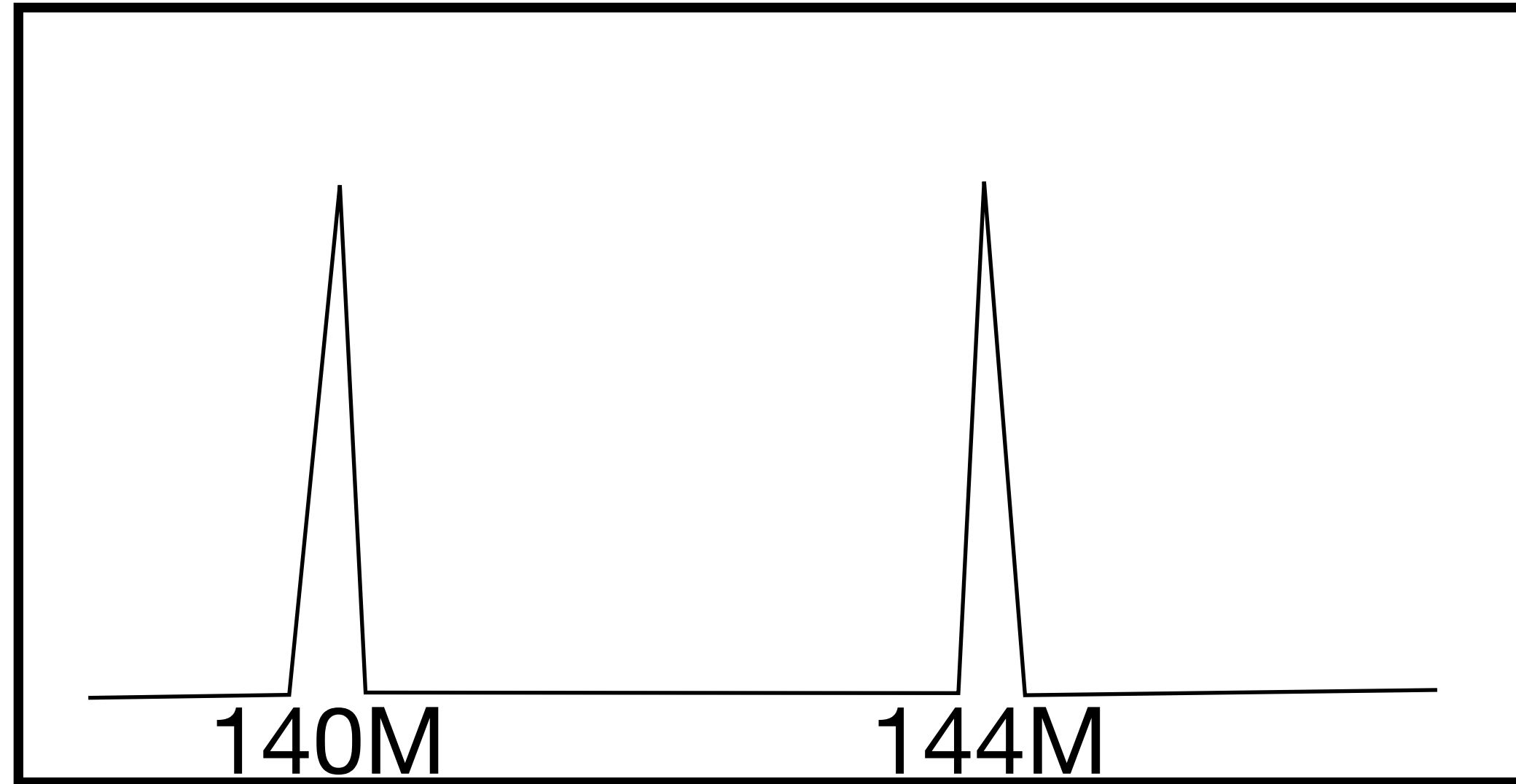
- 1x RF input
- 2x mixers and 2x ADC
  - 90 degrees phase shift
- 2x data out
  - I and Q
  - Magnitude and Phase
  - Complex number



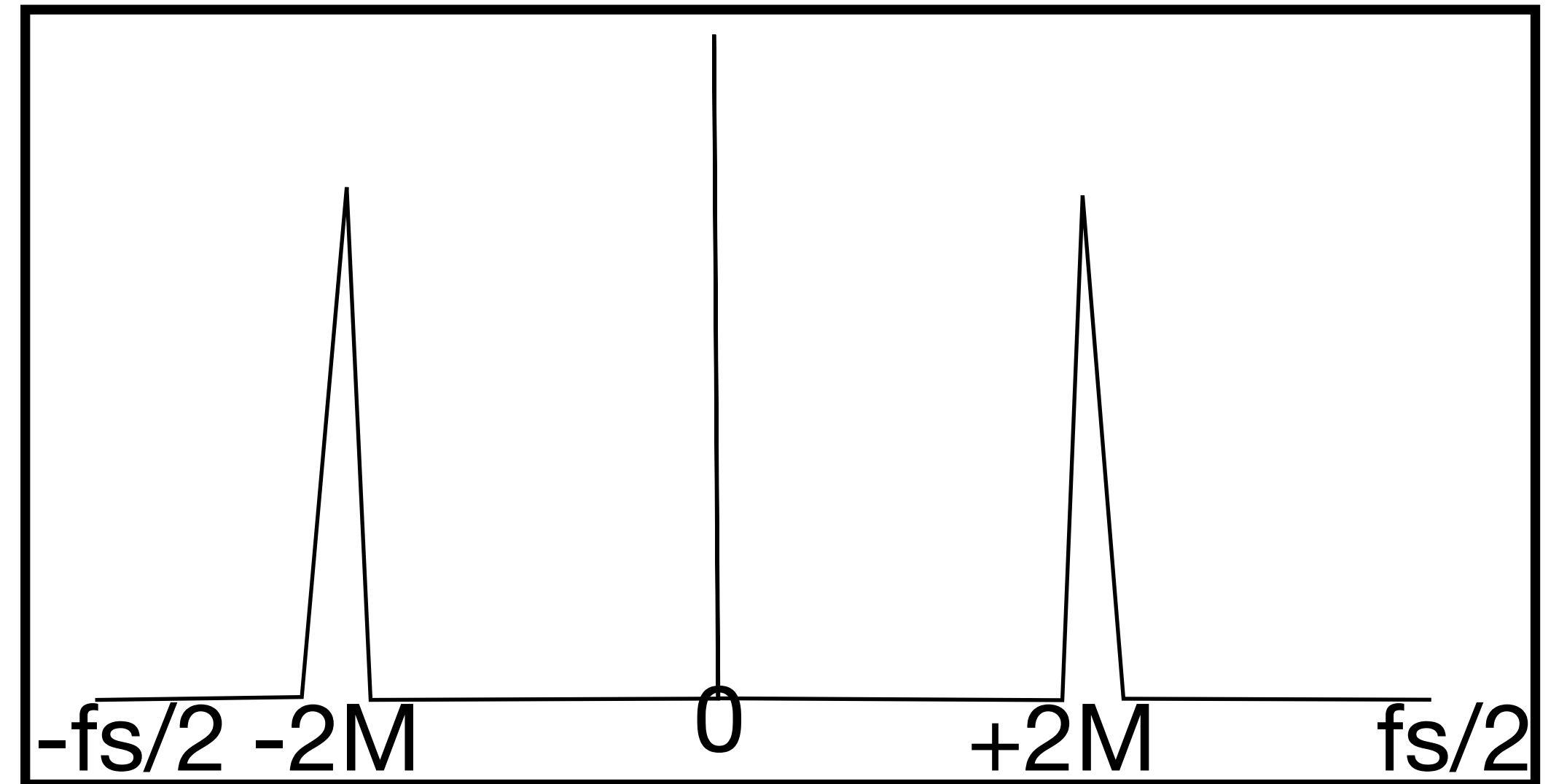
- 8 hours of video course: <https://www.greatscottgadgets.com/sdr/>
- Important: Frequency can be positive and negative! Because we have the phase information

# Example negative frequencies

RF in



Pluto out (complex)

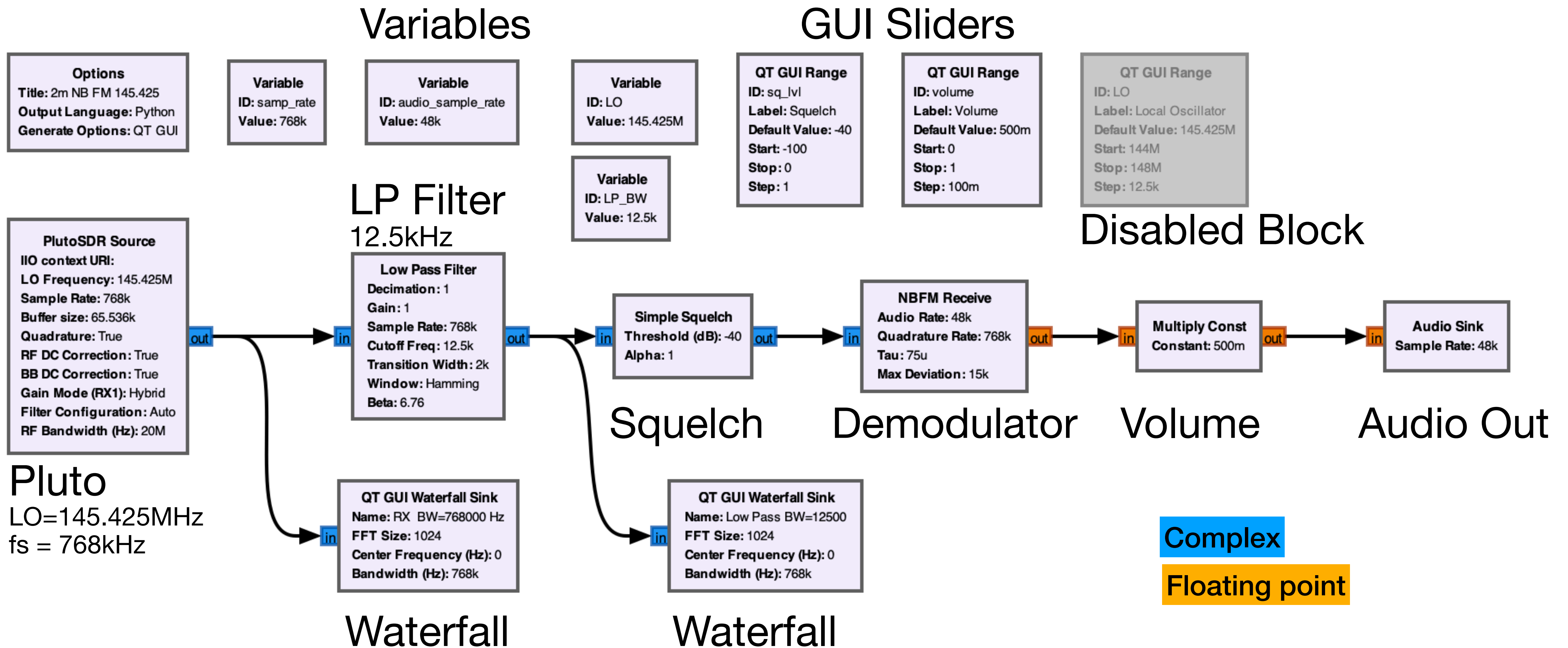


$f_s = \text{sample frequency (6M)}$

LO = 142 MHz

# FM Receiver

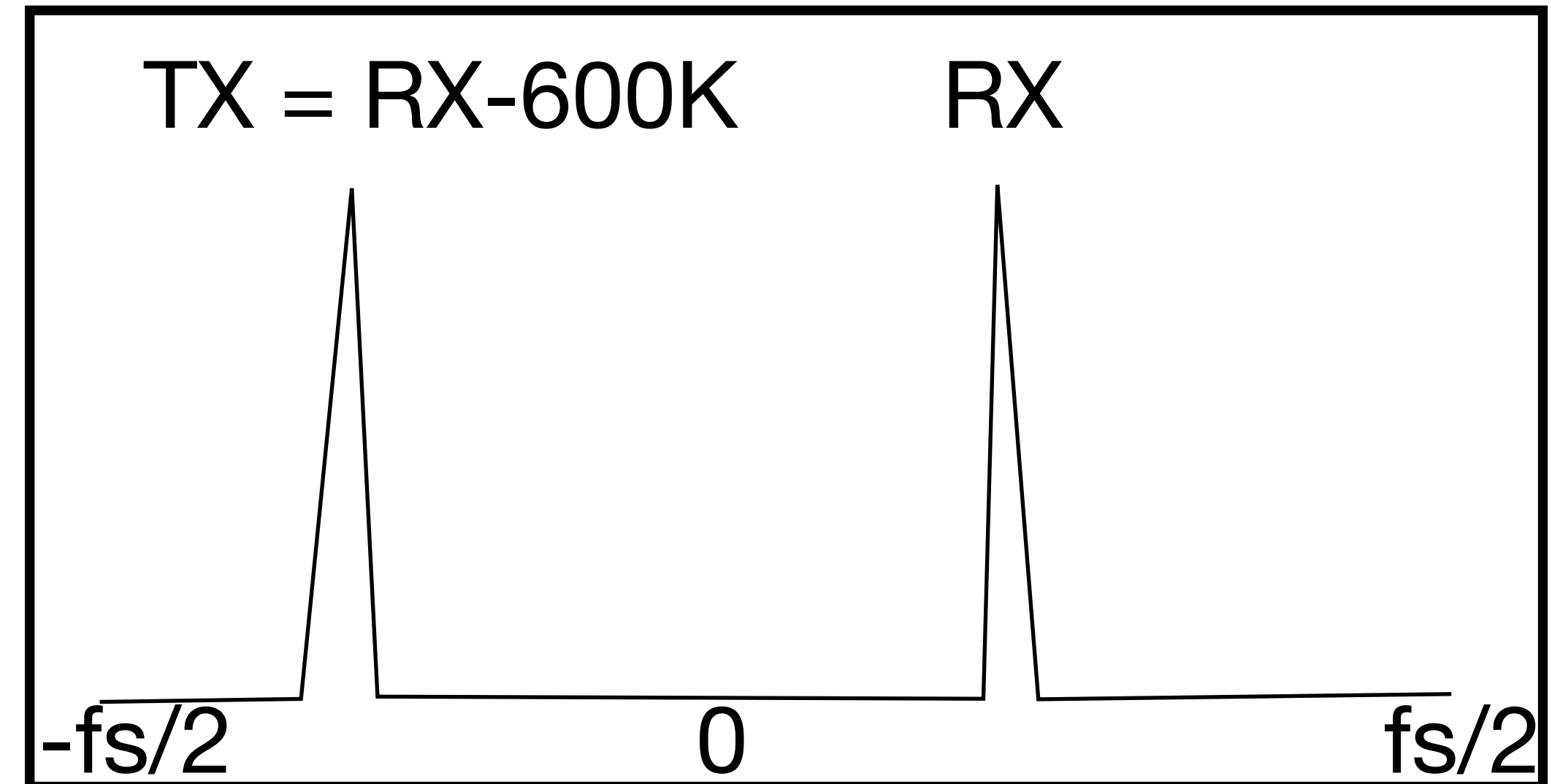
## 12.5kHz BW, Local Oscillator (LO) = 145.425 MHz



# Frequency Offset

## Repeater example

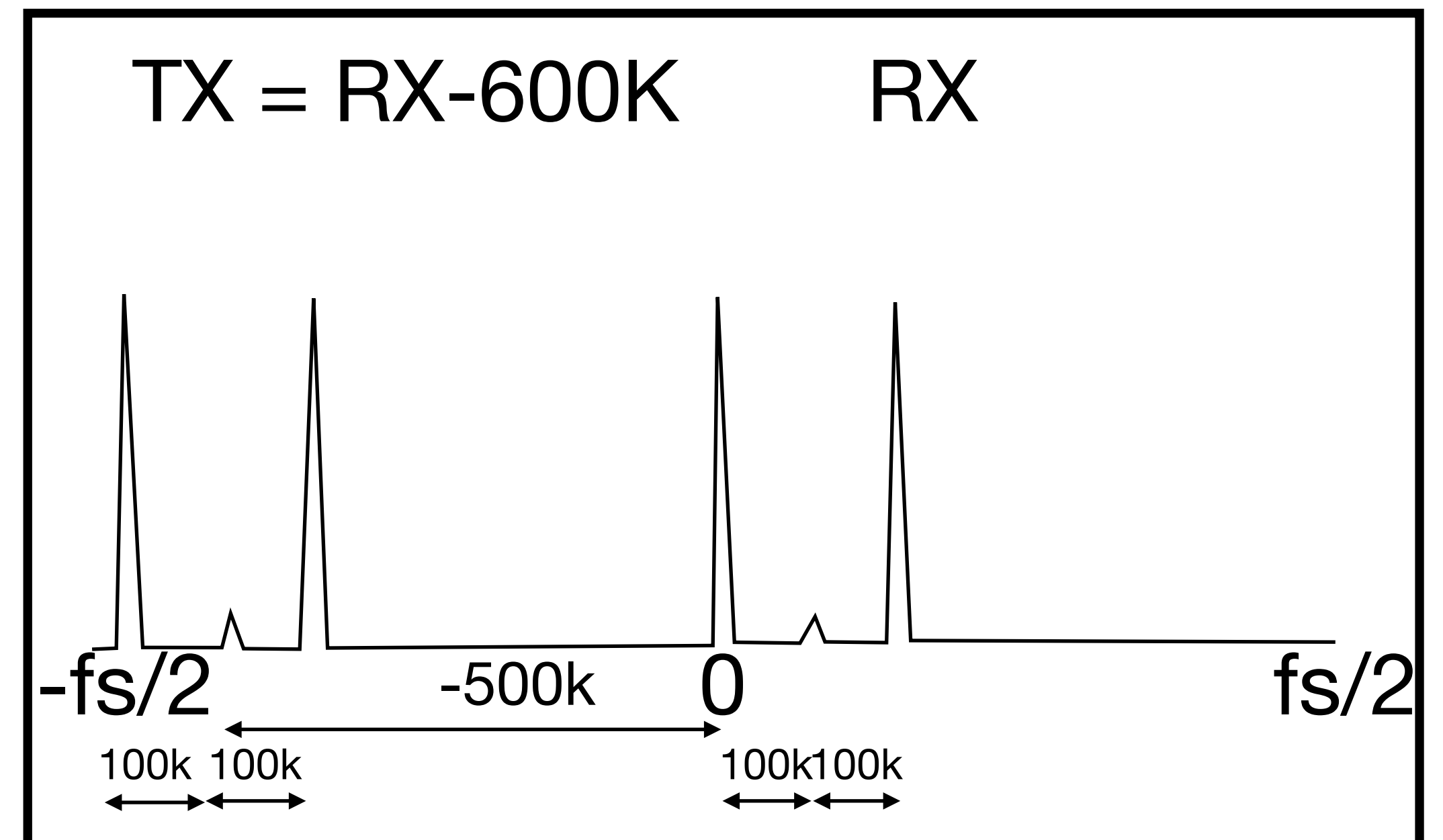
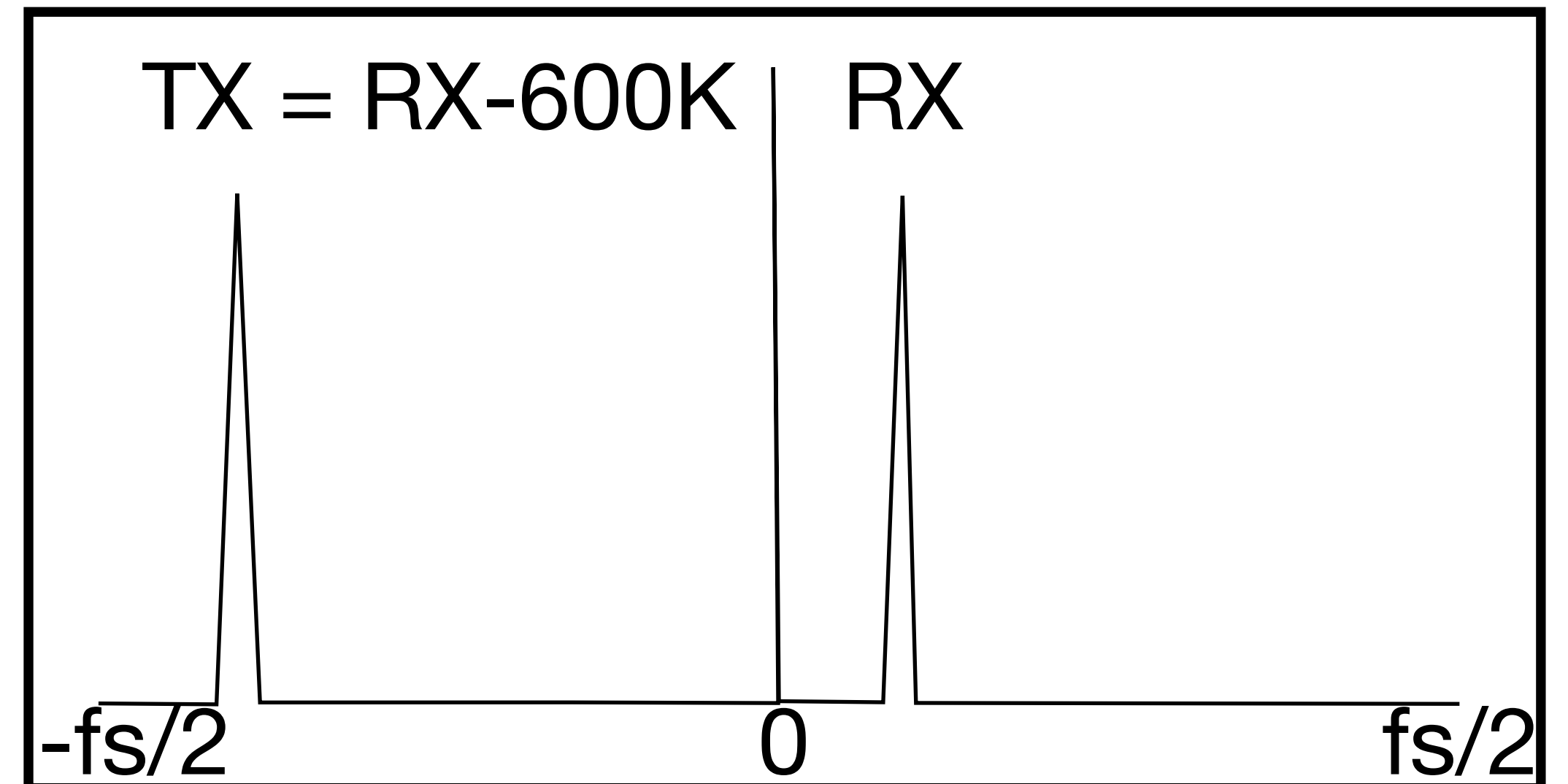
- We do not have to put LO of SDR at exact frequency!  
There can be a shift or offset!
- Set LO=145.4875 MHz (100k below RX)
- RX = 145.5875 MHz
- TX = RX - 600k = 144.9875 Hz
- Now need 100kHz down conversion to get RX
- We use a mixer and 100kHz oscillator to do that.



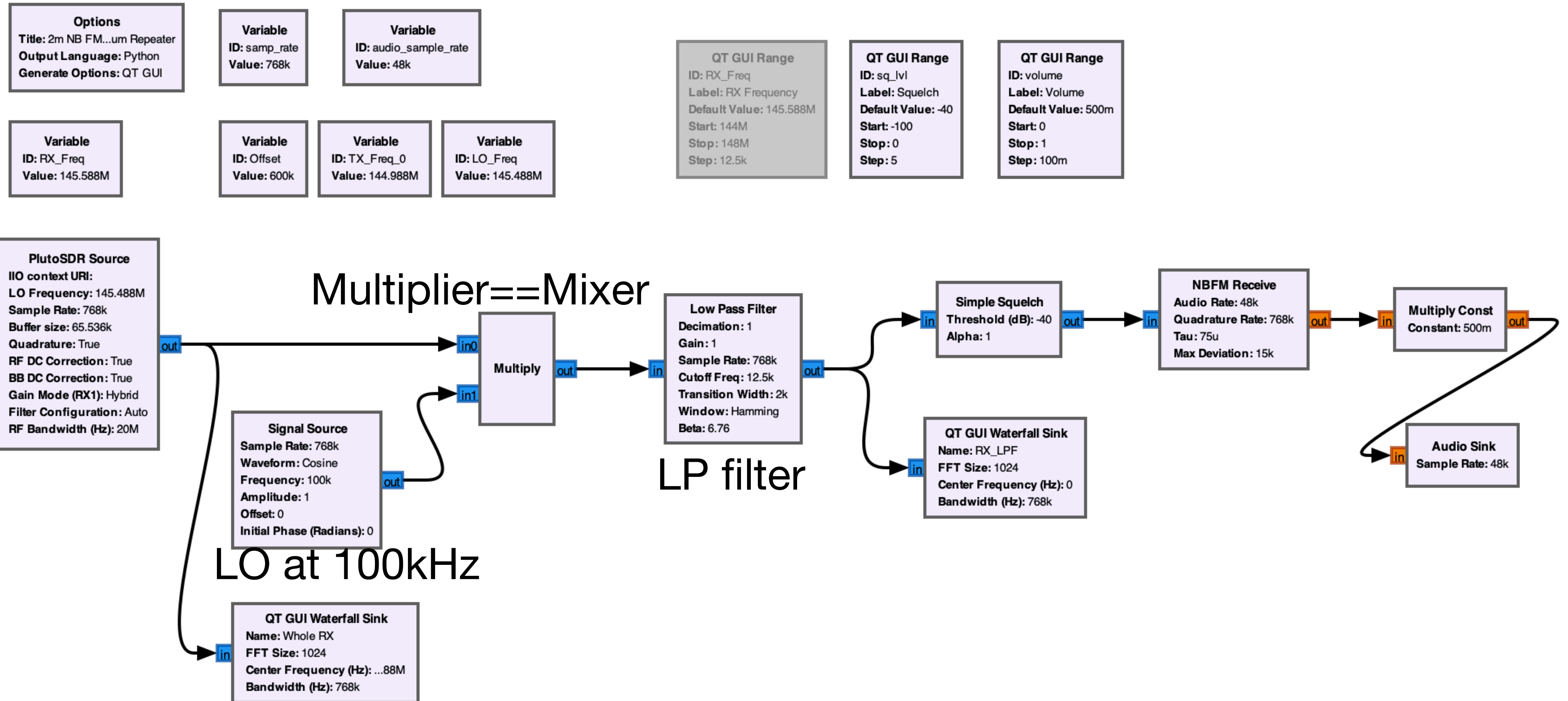
# Repeater

**$RX - 600\text{kHz} = TX$**

- For example set Oscillator for mixer at 100kHz
  - $Rx - 100k = 0 \text{ Hz}$
  - $Rx + 100k = 200 \text{ kHz}$
  - $TX - 100k = -500k - 100k = -600 \text{ kHz}$
  - $TX + 100k = -500k + 100k = -400 \text{ kHz}$
- After LowPass -> only RX
- Low pass filters between +12.5k & -12.5kHz!

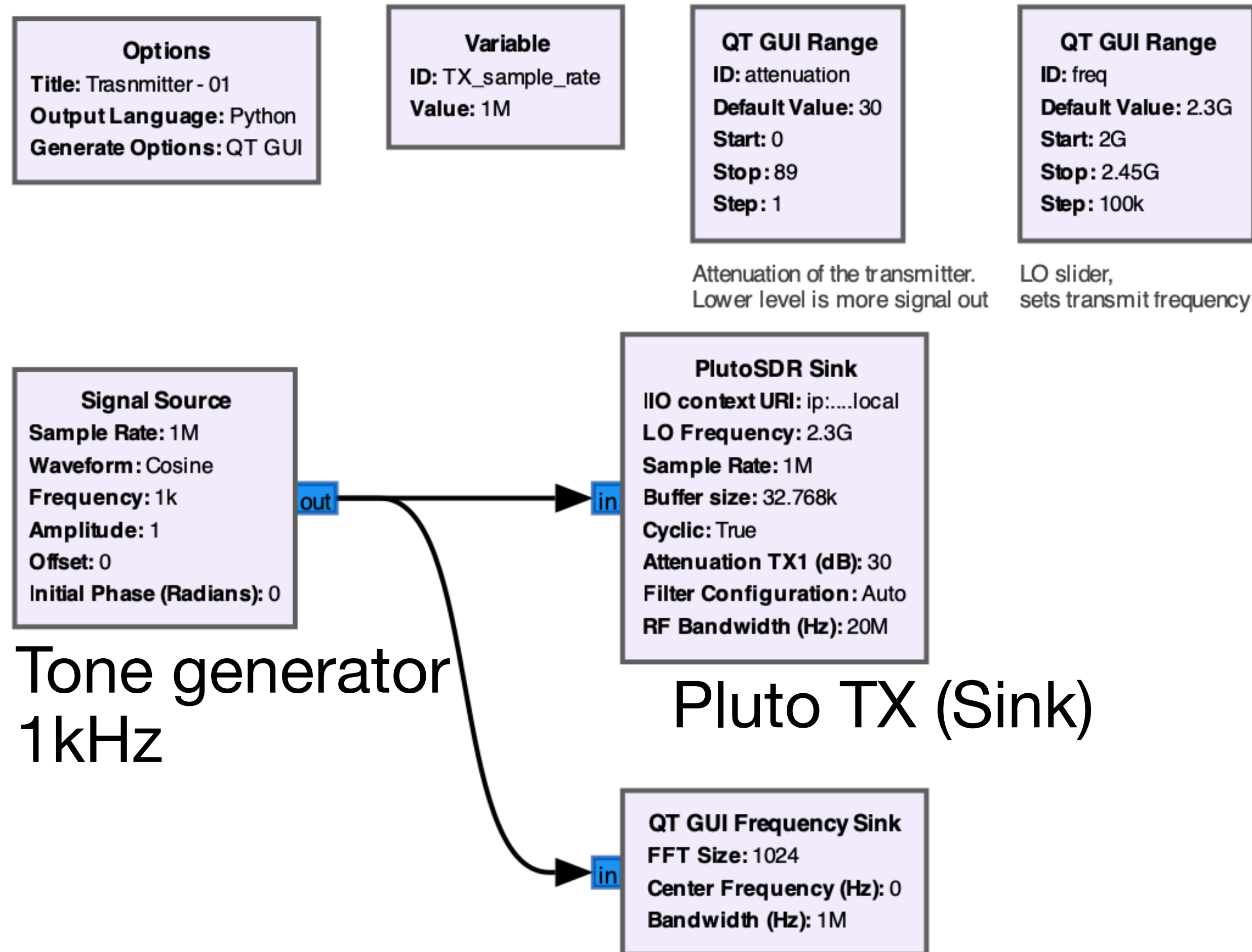


# RX with offset



# Sende på 13cm bandet

## Ekte CW :-) eller 'Single Tone'



# Motta på 13cm bandet

**Options**  
**Title:** Beamform...ADALM-Pluto  
**Author:** Paul H.S. de Vries  
**Description:** Using...strength  
**Output Language:** Python  
**Generate Options:** QT GUI

**Variable**  
**ID:** RX\_sample\_rate  
**Value:** 1M

The sample rate of the receiver

**Variable**  
**ID:** frequency  
**Value:** 2.3G

The frequency of the LO of the receiver

**QT GUI Range**  
**ID:** rf\_gain  
**Label:** RF Gain  
**Default Value:** 40  
**Start:** 0  
**Stop:** 71  
**Step:** 1

The gain of the receiver

**FComms2/3/4 Source**  
**IIO context URI:** ip:...local  
**LO Frequency:** 2.3G  
**Sample Rate:** 1M  
**Buffer size:** 32.768k  
**RX1 Enabled:** True  
**RX2 Enabled:** False  
**Quadrature:** True  
**RF DC Correction:** True  
**BB DC Correction:** True  
**Gain Mode (RX1):** Manual  
**Manual Gain (RX1)(dB):** 40  
**RF Port Select:** A\_BALANCED  
**Filter Configuration:** Auto  
**RF Bandwidth (Hz):** 20M

Using the FComms2/3/4 block instead of the PlutoSDR to be able to access both RX channels of the modified Pluto

**QT GUI Frequency Sink**  
**Name:** RX0  
**FFT Size:** 512  
**Center Frequency (Hz):** 2.3G  
**Bandwidth (Hz):** 1M

**QT GUI Constellation Sink**  
**Name:** RX0 - (R...Phase shift)  
**Number of Points:** 1.024k  
**Autoscale:** No

Constellation graph for the difference of both channels

# Dual Receiver 13cm

## For beam forming experiments

**Options**  
**Title:** Beamform...ADALM-Pluto  
**Author:** Paul H.S. de Vries  
**Description:** Using...strength  
**Output Language:** Python  
**Generate Options:** QT GUI

**Variable**  
**ID:** RX\_sample\_rate  
**Value:** 1M

The sample rate of the receiver

**Variable**  
**ID:** frequency  
**Value:** 2.3G

The frequency of the receiver

**QT GUI Range**  
**ID:** rf\_gain  
**Label:** RF Gain  
**Default Value:** 60  
**Start:** 0  
**Stop:** 71  
**Step:** 1

The gain of the receiver

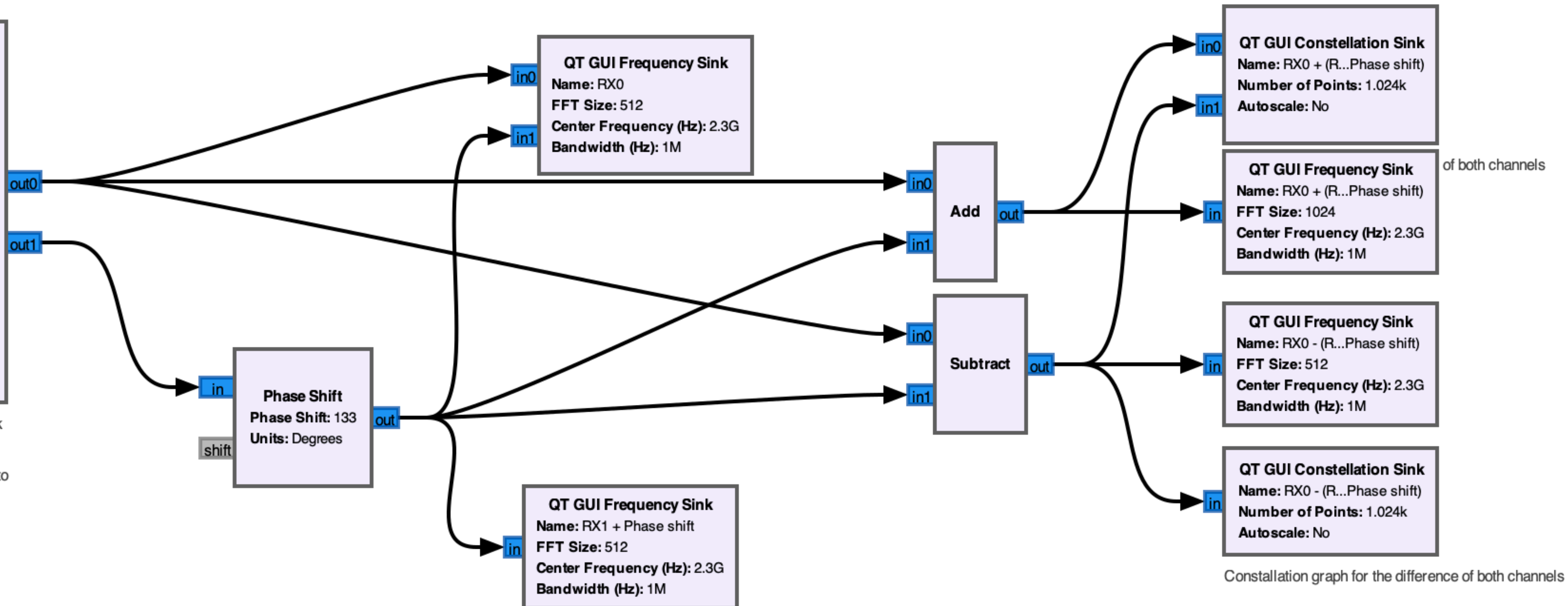
**QT GUI Range**  
**ID:** phase\_delay  
**Label:** Phase delay  
**Default Value:** 0  
**Start:** -180  
**Stop:** 180  
**Step:** 1

**Variable**  
**ID:** phase\_delay\_calibration  
**Value:** 133

As the wires/tracks for both RX channels do not have equal length, there will always be a delay between the two channels. With the phase\_delay\_calibration you can compensate

**FMComms2/3/4 Source**  
**IIO context URI:** ip:...local  
**LO Frequency:** 2.3G  
**Sample Rate:** 1M  
**Buffer size:** 32.768k  
**RX1 Enabled:** True  
**RX2 Enabled:** True  
**Quadrature:** True  
**RF DC Correction:** True  
**BB DC Correction:** True  
**Gain Mode (RX1):** Manual  
**Manual Gain (RX1)(dB):** 60  
**Gain Mode (RX2):** Manual  
**Manual Gain (RX2)(dB):** 60  
**RF Port Select:** A\_BALANCED  
**Filter Configuration:** Auto  
**RF Bandwidth (Hz):** 20M

Using the FMComms2/3/4 block instead of the PlutoSDR to be able to access both RX channels of the modified Pluto



**Questions?**