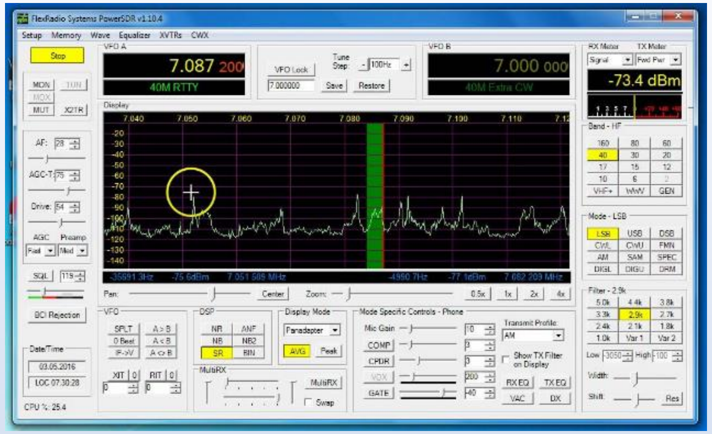
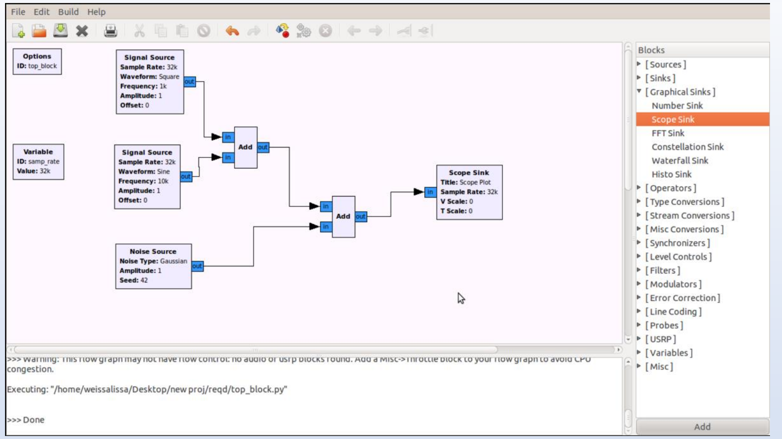


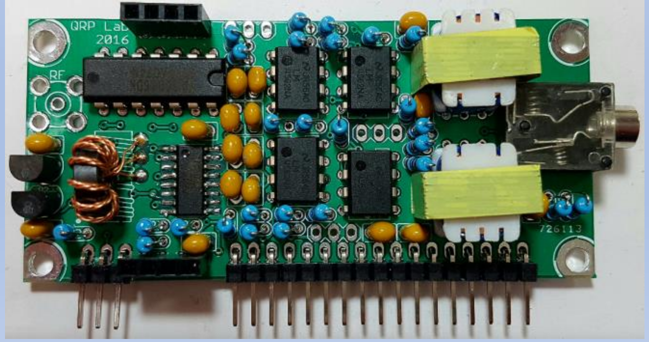


```
>>> Warning: This flow graph may not have flow control: no audio or usrp blocks found. Add a Misc->Throttle block to your flow graph to avoid CPU congestion.

Executing: "/home/weissalissa/Desktop/new proj/reqd/top_block.py"

>>> Done
```

World of SDR

Tino Zottola, VE2GCE

April 19, 2021



Agenda

- Introduction
- HDR Architecture
- SDR Architecture
- Commercial SDR: Icom 7300
- SDR Kits
  - SoftRock Lite II Receiver
  - Ensemble RXTX Transceiver
  - uSDX Transceiver
- Homebrew SDR
- Hack RF One and GNU Radio
- Conclusion

SDR Introduction

What is SDR (Software Defined Radio) ?

1970: US DoD lab researcher coined the term ‘Digital radio’ which operated on Midas SW platform.

1984:“Software radio" term for digital baseband receiver coined by E-Systems team (Raytheon)

1991: Joe Mitola championed SDR idea (using Texas Instruments DSP chips) to US Air Force

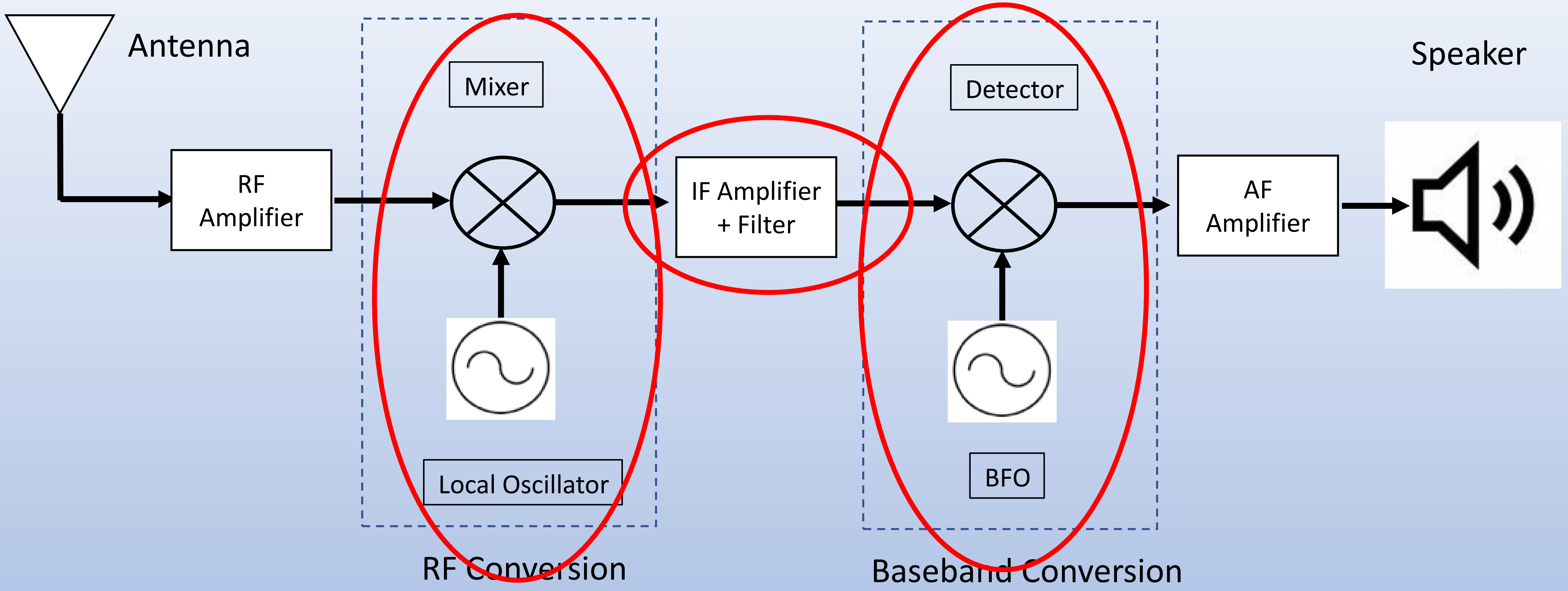
USAAF SPEAKeasy radio program had the following requirements:

- Physical layer components implemented in software
- Single radio supports ten different military radio protocols
- 2 MHz through 2 GHz coverage
- Future-proof radio hardware, not possible with HDR (Hardware Defined Radio)

2003: Gerald Youngblood (K5SDR) wrote pioneering SDR articles for ARRL in early 2000’s.

- He is the founder of Flex-Radio Systems.
- One of 1st SDR radios (e.g. FlexRadio’s SDR-1000) for ham radio was designed by Youngblood

HDR Architecture - Receiver

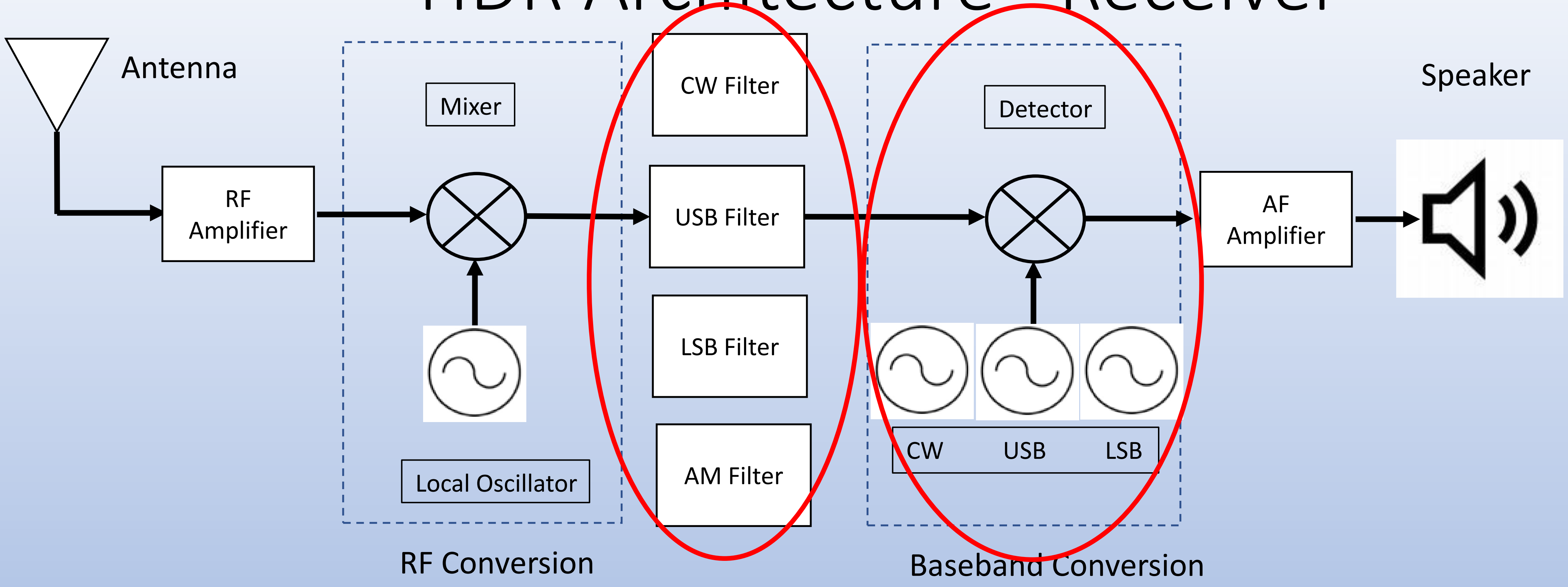


Superheterodyne (invented by Major Edwin Armstrong) has been the standard for 100+ years, but has the following deficiencies:

1. Each heterodyne stage incurs -6 dB loss during analog conversion (for passive mixers)
   - -12 dB loss ➔ single conversion (heterodyne stage + detector)
   - -18 dB loss ➔ double conversion (2 x heterodyne stages + detector)
2. Crystal filter adds another -6 dB loss

18/24 dB signal path loss

HDR Architecture - Receiver



4. Require expensive *add-on crystal filters, one per mode (i.e. LSB, USB, CW, etc.) for selectivity.
5. Filter bandwidth of crystal filter is fixed (i.e. CW 800 Hz , SSB 2.1 KHz , AM 3.0 KHz)
6. Topology is rigid: Demodulation schemes are fixed options
7. Many stages (heterodyne and filters) required to obtain reasonable selectivity and image rejection

* Practice of requiring purchase of separate crystal filters started by Collins Radio bean counters in 1960s to increase revenue.

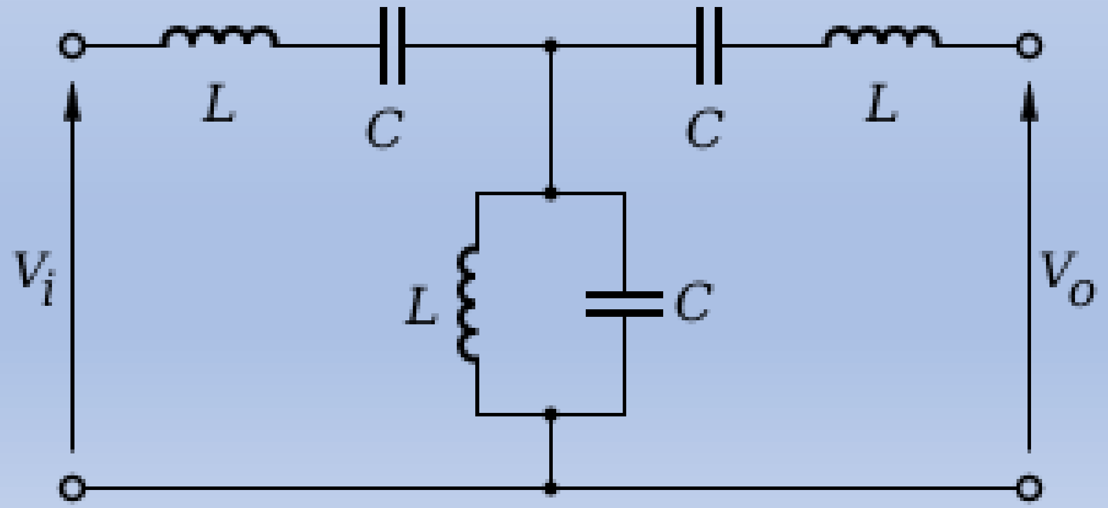
SDR Introduction

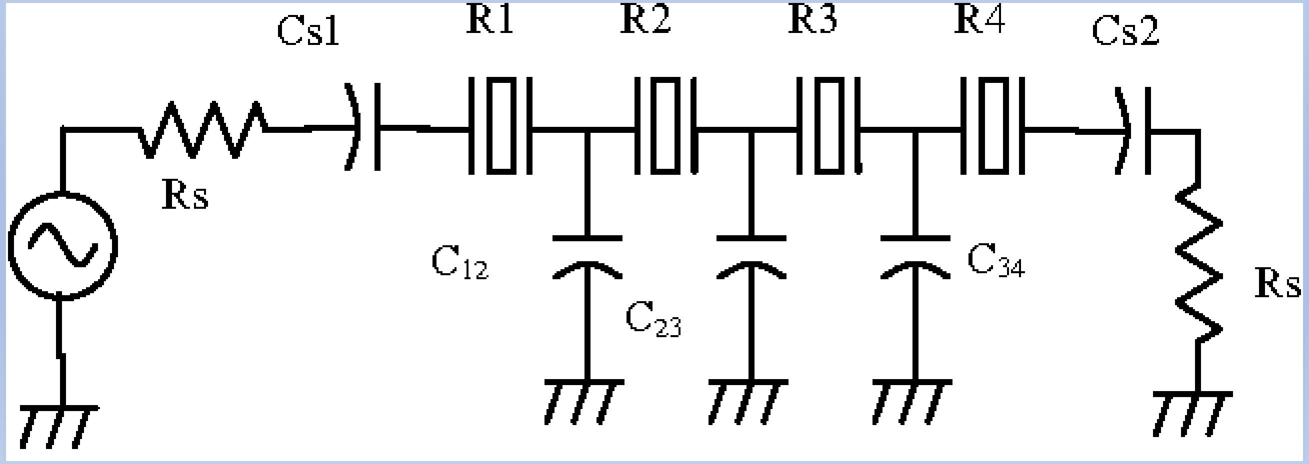
Conventional HDR band pass filter can take one of the following forms:

- LC filter
- Crystal filter

Altering filter response is complicated:

- Change in physical components (e.g. capacitors, inductors, resistors) is needed
- Changes are much more complicated (if not impossible) with crystal filters.



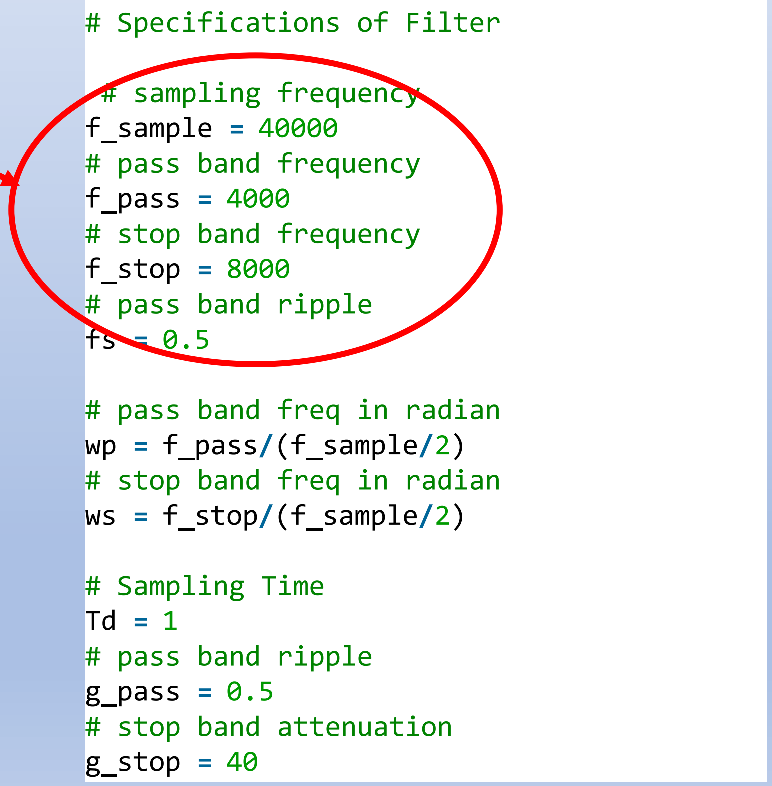


SDR Introduction

SDR uses software to perform radio functions (i.e. filtering, demod, phase delay, etc) in signal path.

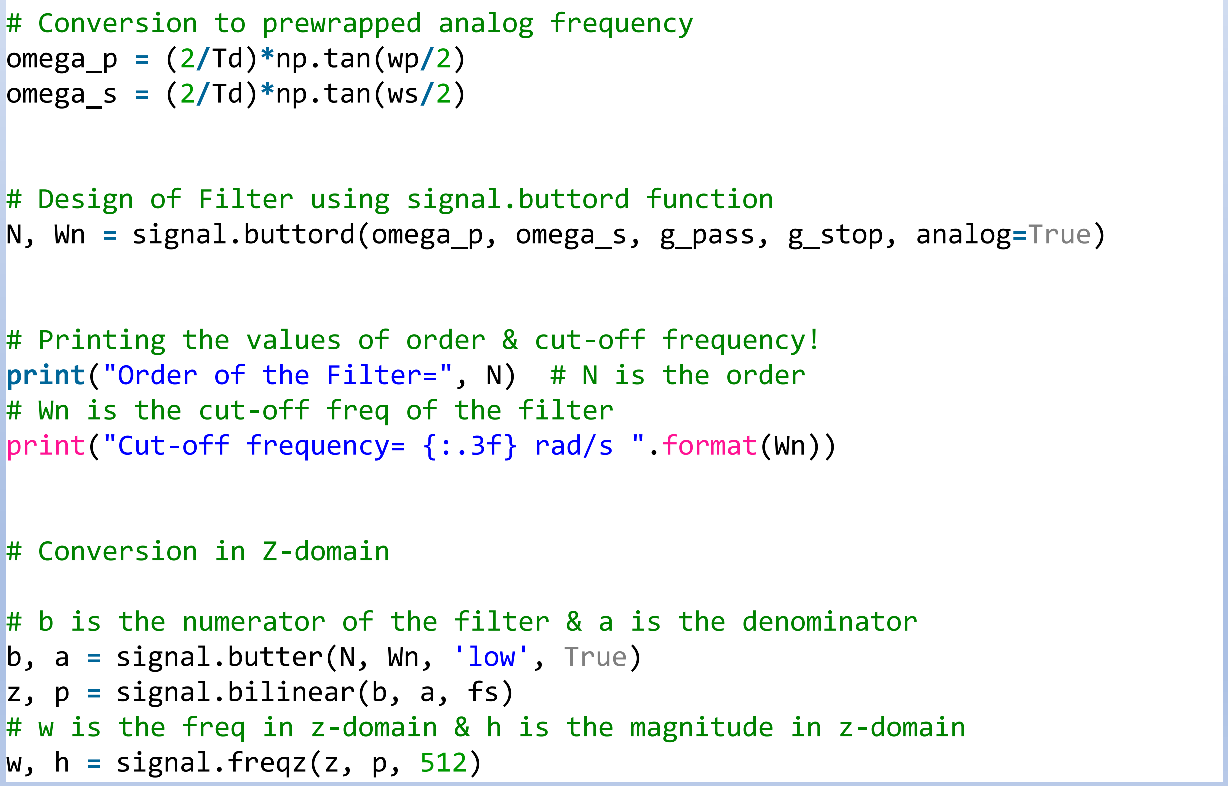
For example, mathematical representation of Butterworth BPF in software could look like:

- Changes in filter response involve only changing variable(s) vs physical parts in HDR
- Changes are instantaneous and can be done on-the-fly via the user interface



Filter parameters

```
# Specifications of Filter

 # sampling frequency
f_sample = 40000
# pass band frequency
f_pass = 4000
# stop band frequency
f_stop = 8000
# pass band ripple
fs = 0.5

# pass band freq in radian
wp = f_pass/(f_sample/2)
# stop band freq in radian
ws = f_stop/(f_sample/2)

# Sampling Time
Td = 1
# pass band ripple
g_pass = 0.5
# stop band attenuation
g_stop = 40
```

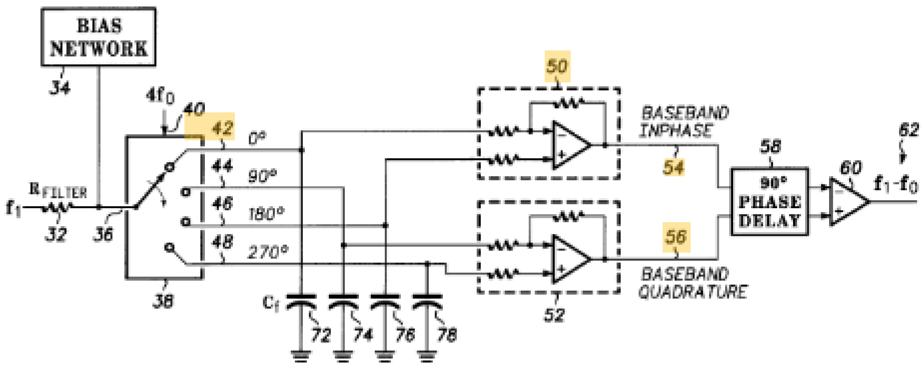


```
# Conversion to prewrapped analog frequency
omega_p = (2/Td)*np.tan(wp/2)
omega_s = (2/Td)*np.tan(ws/2)


# Design of Filter using signal.buttord function
N, Wn = signal.buttord(omega_p, omega_s, g_pass, g_stop, analog=True)


# Printing the values of order & cut-off frequency!
print("Order of the Filter=", N)  # N is the order
# Wn is the cut-off freq of the filter
print("Cut-off frequency= {:.3f} rad/s ".format(Wn))


# Conversion in Z-domain

# b is the numerator of the filter & a is the denominator
b, a = signal.butter(N, Wn, 'low', True)
z, p = signal.bilinear(b, a, fs)
# w is the freq in z-domain & h is the magnitude in z-domain
w, h = signal.freqz(z, p, 512)
```

SDR Architecture – Front End

Many SDR radios use QSD (Tayloe*) for front end conversion.

- No RF amp needed, +100 dB stable baseband gain, almost unity gain conversion gain.
- Built in front end selectivity ➔ e.g. 10Mhz input @ filter knee = 2 x 1KHz ➔ Q = 10Mhz/2KHz ➔ 5000

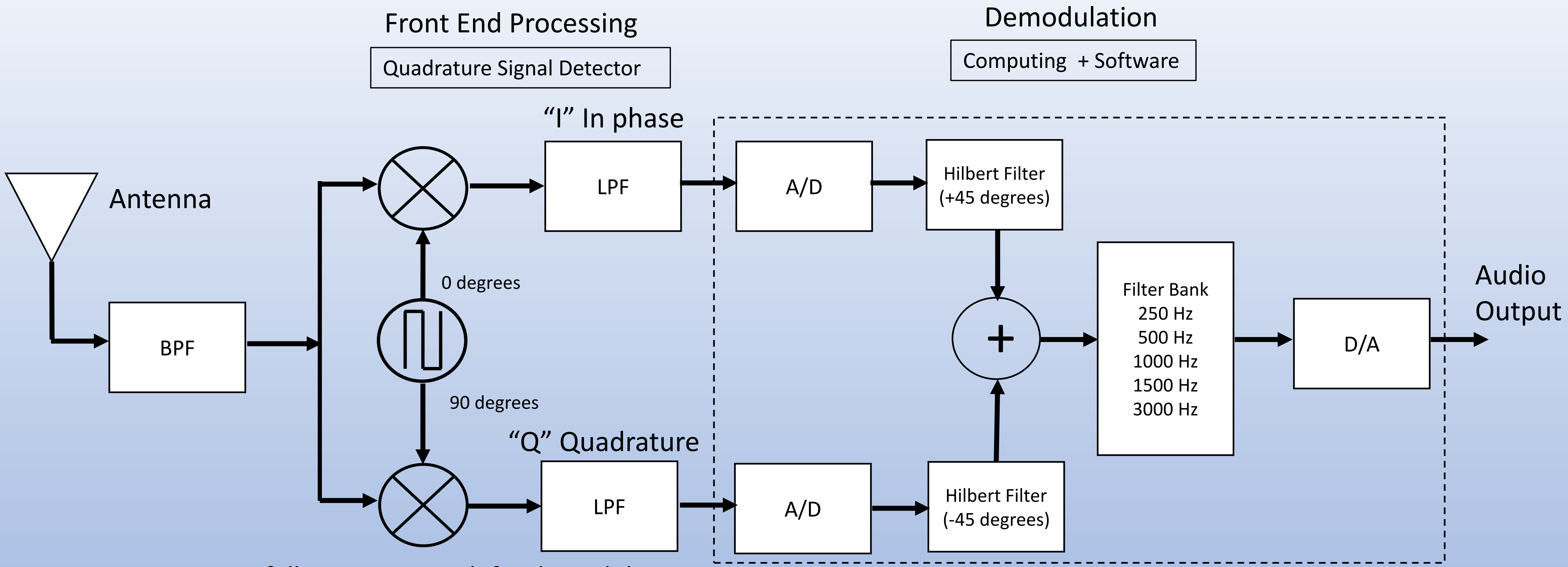


30 *FIG. 3*

- Quadrature Signal Demodulation: Daniel Tayloe (N7VE), Motorola, patented in 2001 (Quasi-Digital)
- Quadrature mod-demod (phasing method) by Hartley patented in 1928. (Analog)
- What is the difference ? ➔ Hartley demod uses heterodyne stages (0°, 90°) to create I-Q + analog delay stage (90 °)

  ➔ Tayloe demod uses sampler stage (0°, 90°, 180°, 270°) to create I-Q + Hilbert delay stage (90 °)
- It is very difficult to achieve an exact 90° phase shift at the RF and AF level using analog technology.
- This is why most commercial radios between 1955-2015 used the filter method over the phasing method for SSB.

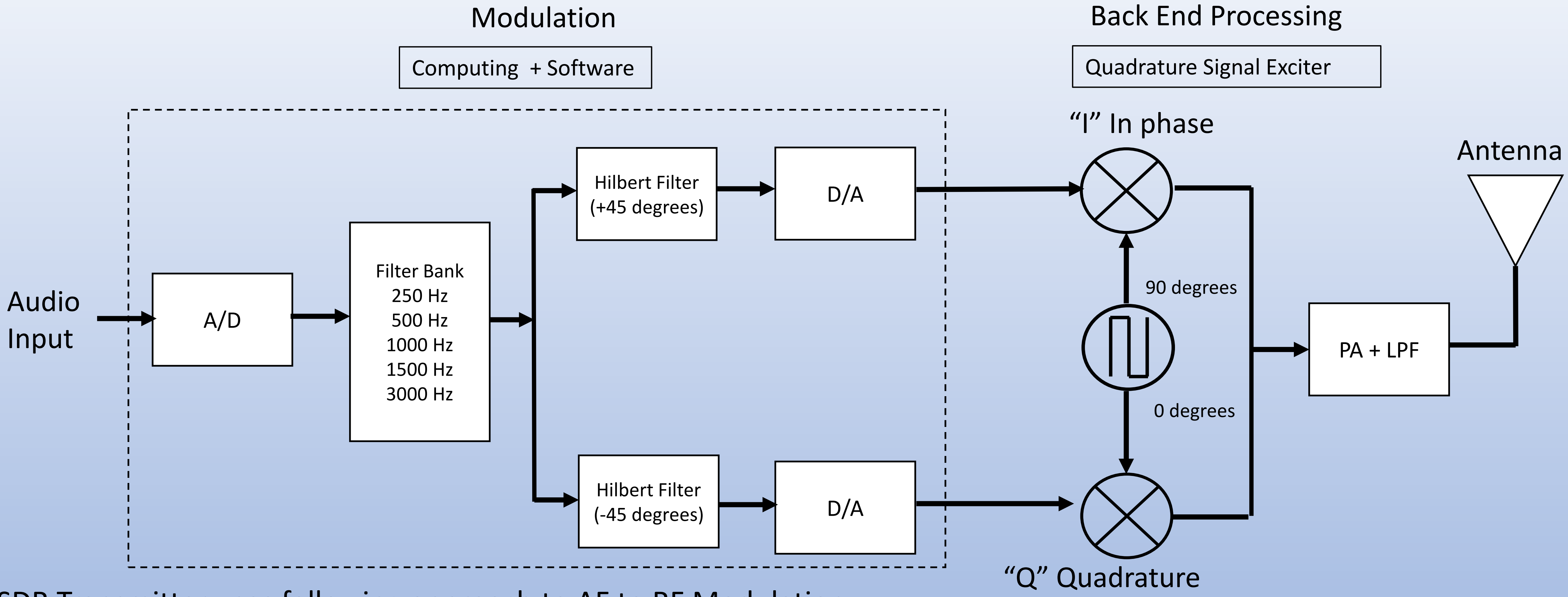
SDR Architecture - Receiver



SDR Receiver uses following approach for demodulation.

- Front End: RF to baseband conversion via sampling mixer: Tayloe QSD
- Baseband to AF conversion (demodulation and filtering) done by software
- Hilbert Filters used as wideband phase shifter
- To go from LSB to USB reception, we simply invert the "I" signal.

SDR Architecture - Transmitter



SDR Transmitter uses following approach to AF to RF Modulation.

- DSP: AF to baseband (modulation and filtering) done by software
- Back End: Baseband to RF conversion done by digital sampling (Tayloe or QSE (Quadrature Signal Exciter)
- Followed by conventional power amplifier followed by LPF for emissions compliance.

Commercial SDR Radio

Is a radio with an onboard computer (i.e. microcontroller) an SDR radio ?

Answer:

- Yes, if part or all of the physical layer is implemented with software and/or programable HW
- No, if microcontroller is only used for ancillary functions, e.g. VFO control + frequency display

Example 1:

Icom IC-730 uses a microprocessor for split VFO, frequency display and memory.

- Radio signal path is implemented in fixed hardware ➔ Not SDR

Generally speaking, most pre-2015 radios are not SDR.

Commercial SDR Radio

Example 2:

Icom IC-7300 has a microprocessor and significant SDR hardware (i.e. FPGA, DSP chips)

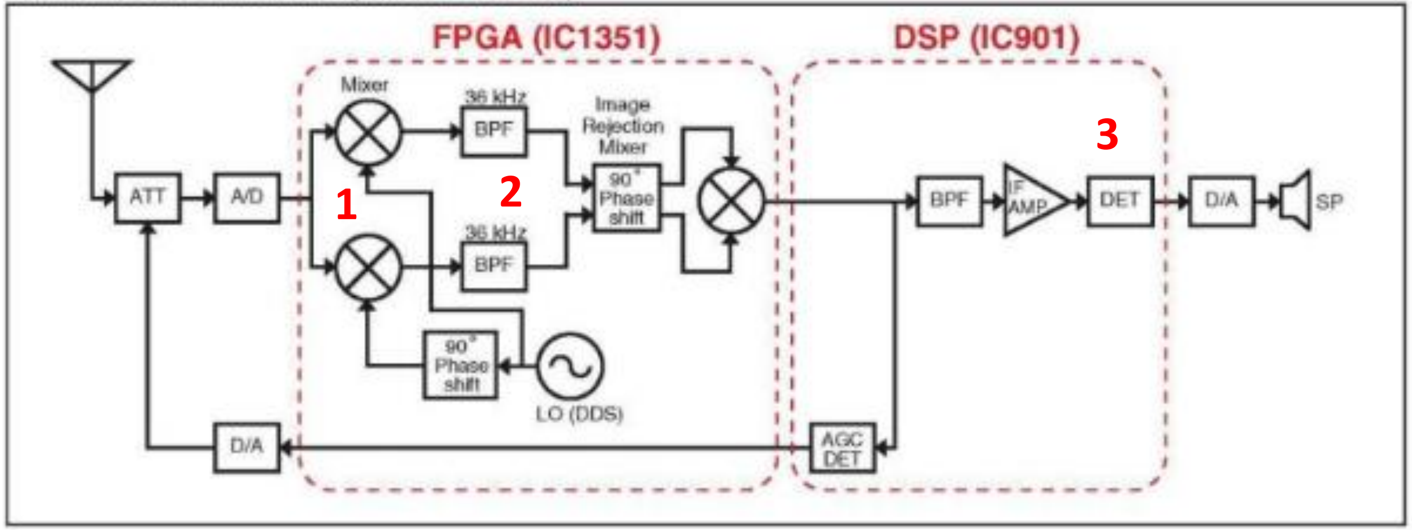
Significant part of physical layer is software configurable and HW programable ➔ Definitely SDR

SDR layer is implemented with the following components:

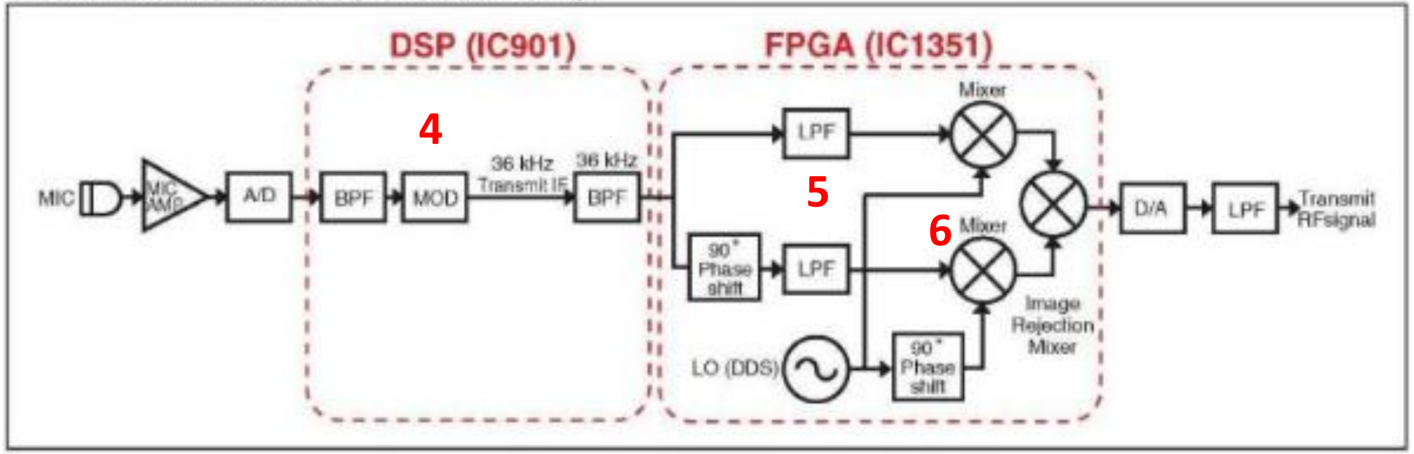
- CPU (Central Processing Unit) ➔ general purpose computer
- FPGA (Field Programmable Gate Array) ➔ programmable HW
- DSP (Digital Signal Processor) ➔ mathematical function processor

Icom 7300 Overview

- FPGA BLOCK DIAGRAM (Receive circuits)



- FPGA BLOCK DIAGRAM (Transmit circuits)



RX Side

1) Taylor Sampler (QSD)
2) IQ split
3) 36 KHz IF + Demodulation

TX Side

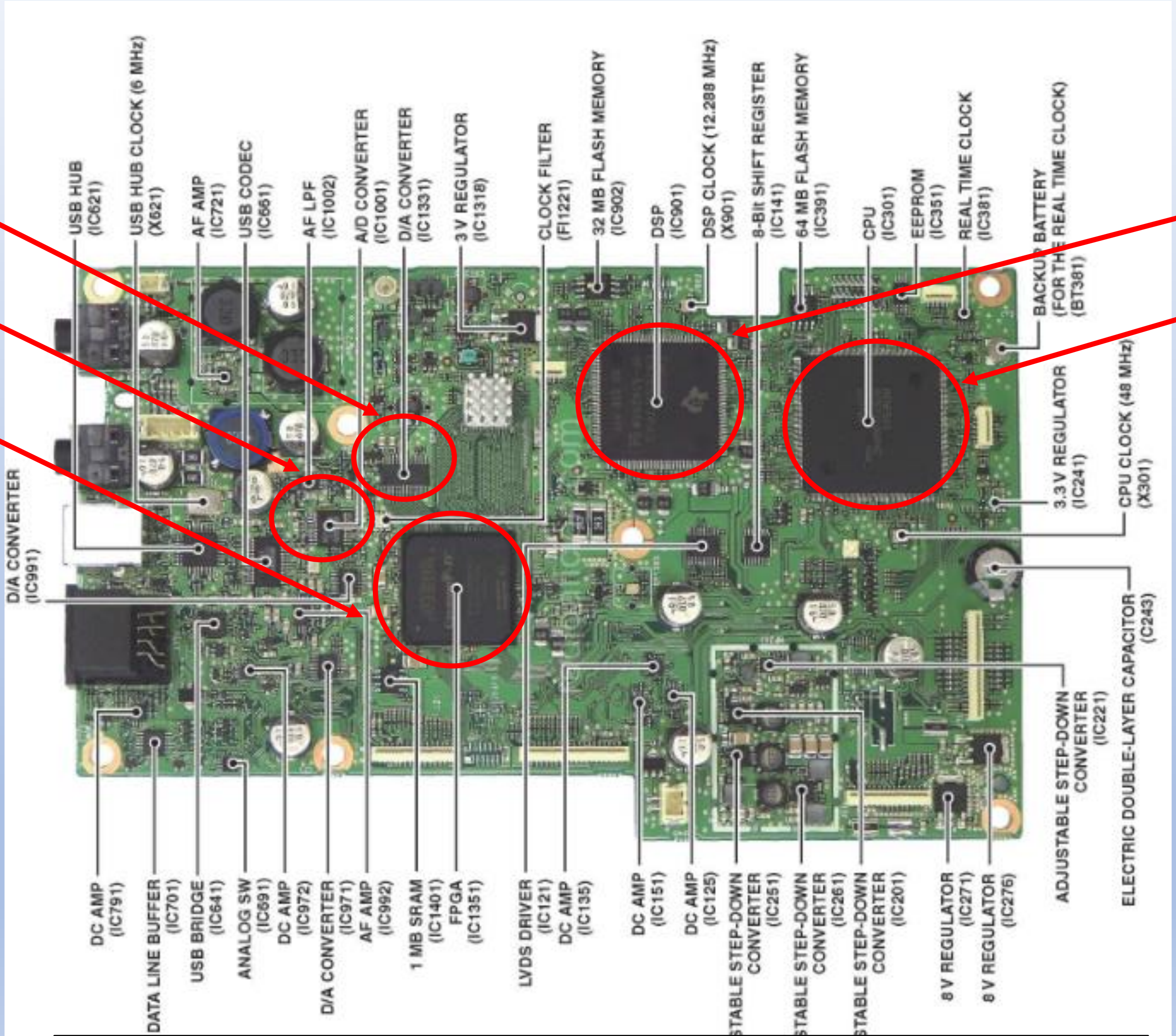
4) Modulated signal onto 36 KHz IF
5) IQ creation
6) Taylor Sampler (QSE)

Notes:

a) Complete digital processing end-to-end
- Examples shown previously had analog front-half and digital back-half

b) Normally QSE and QSD uses zero IF
- DSP chip used here requires 36 KHz IF
- Lower phase noise than zero IF

Icom 7300 Overview



D/A (Digital to Analog)

A/D (Analog to Digital)

FPGA (Programmable HW)

DSP (Digital Signal Processor)

CPU (Central Processing Unit)

Note the absence of the following in the RF core:

- No band crystals
- No crystal filters
- No IF transformers
- No heterodyne stages

RF Core looks like a computer board, rather than a RF assembly

SoftRock Lite II Receiver

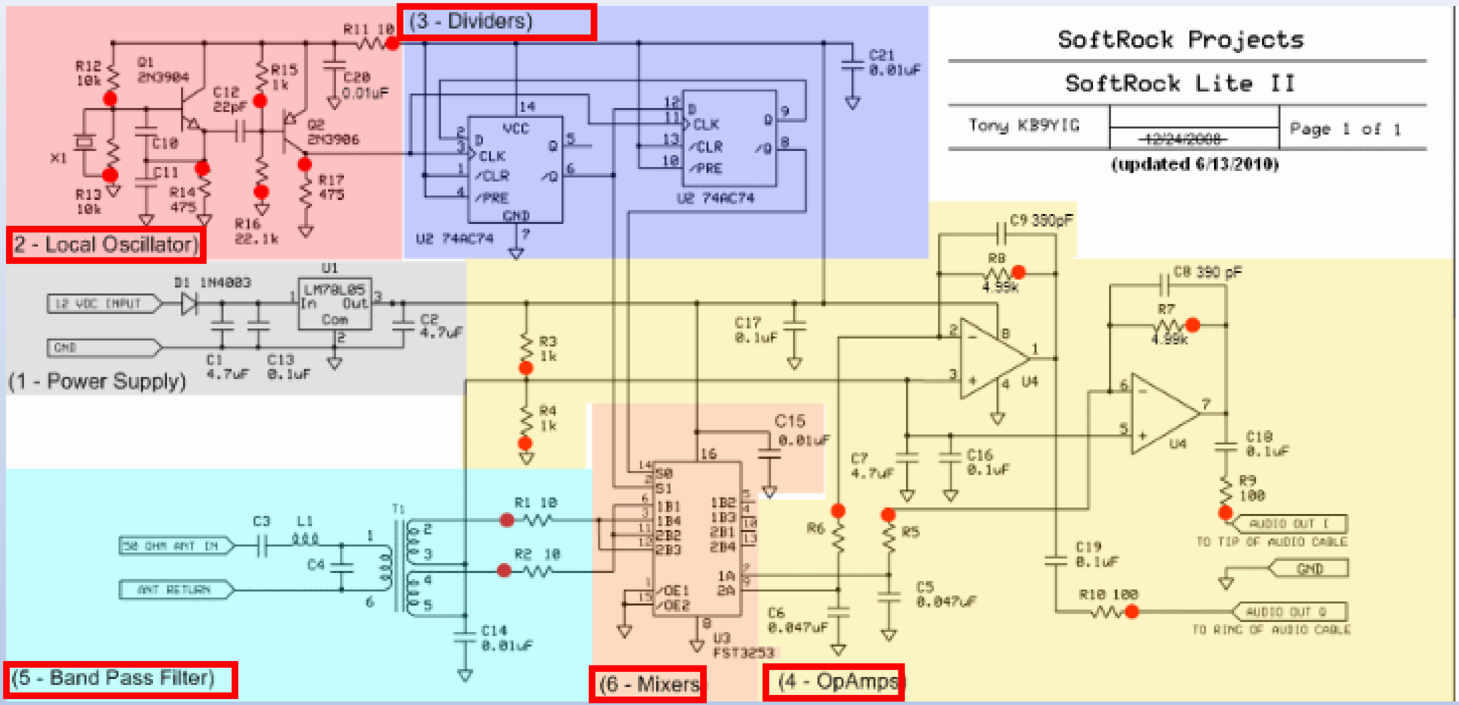
Simplest SDR receiver kit features:

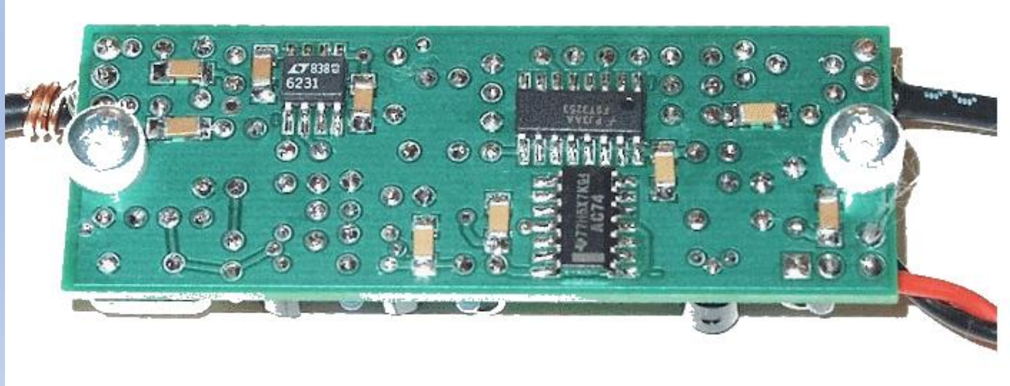
- Single band (160 - 40 meter) SDR receiver kit
- Available for $20 USD from http://fivedash.com/
- Connects to external computer via stereo cable

Main circuit blocks:

- (2) Local Oscillator @ 4 x fo
- (3) Johnson counter ➔ 0 & 90 degree clock
- (5) Bandpass filter
- (6) FST3253 analog switch as Tayloe (QSD) sampler
- (4) LT6231 Opamp: I and Q lowpass filters

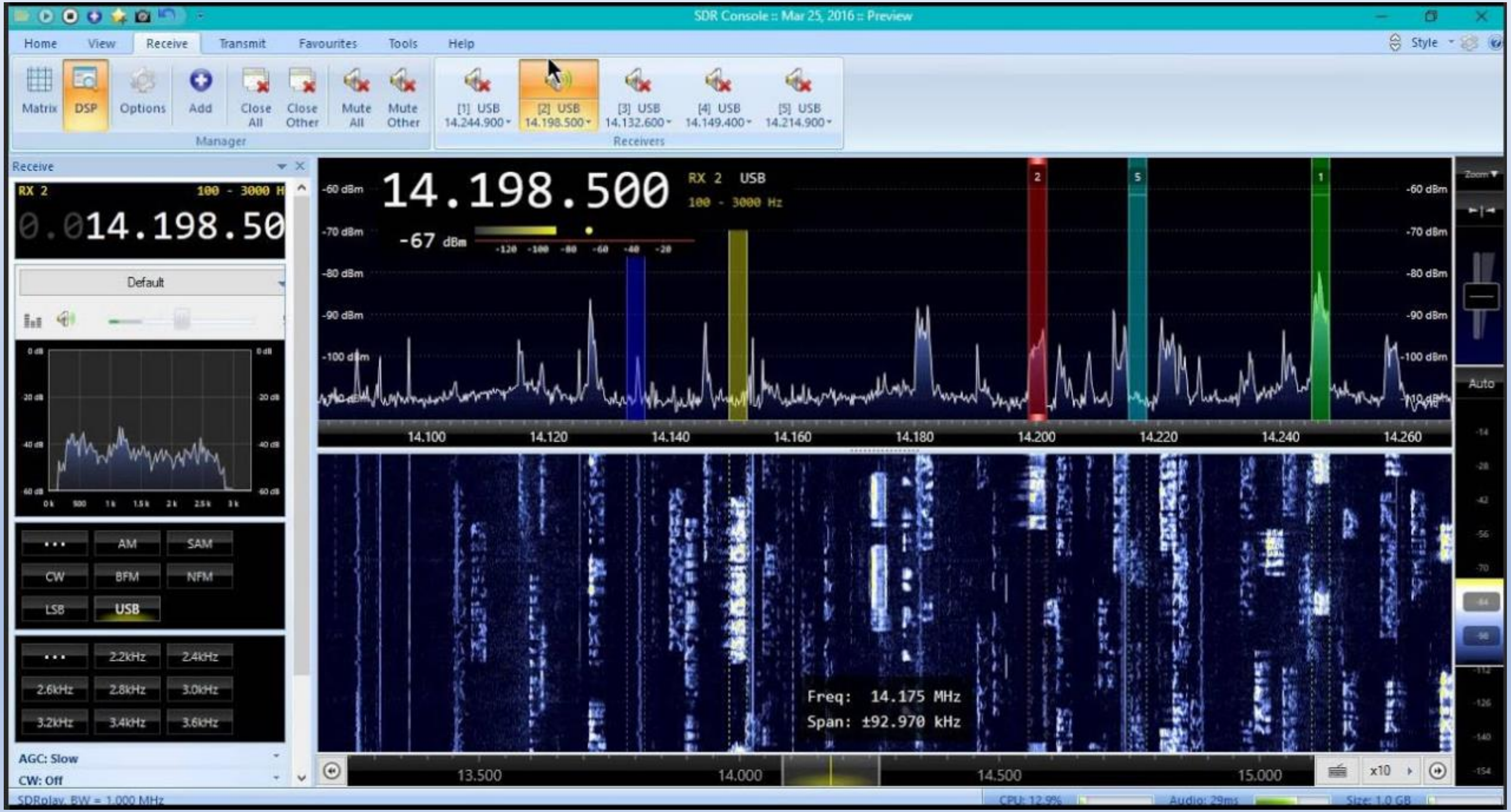
➔ Computer + SDR SW completes demodulation process



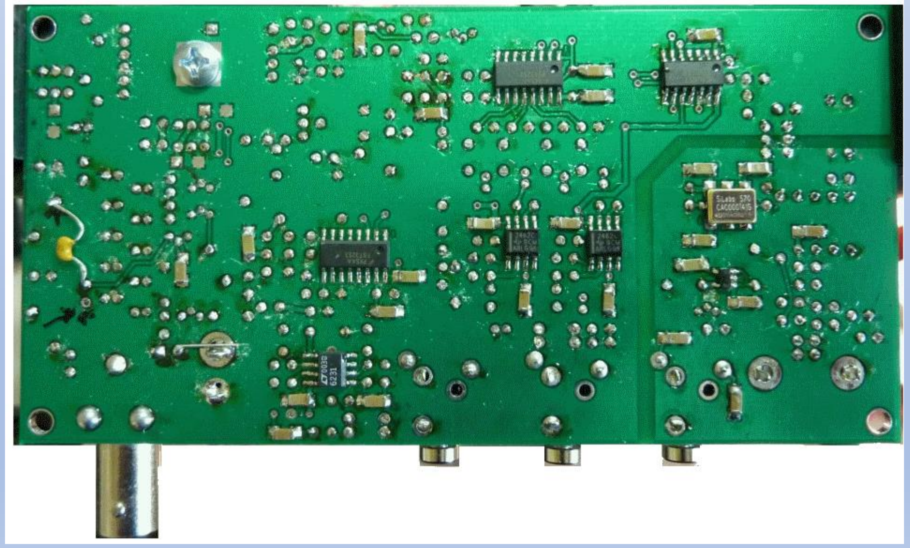


SDR Console

Simple SDR kits consist only of the front end for RX and back end for TX.
They require a computer running SDR software to complete I and Q demodulation and modulation process



Ensemble RxTx SDR Transceiver



SDR transceiver kit features the following:

- Single band (160 -17 meter) SDR transceiver kit
- Available for $89 USD from http://fivedash.com/
- Attiny85 uC for keyer and PTT control (not used for SDR)

SDR Receiver

- Tayloe QSD architecture
- Same Rx as SoftRock Lite II Receiver

SDR Transmitter

- Tayloe QSE architecture (essentially QSD in reverse)
- Power amplifier for 1 watt RF output
- Solid state TR circuitry

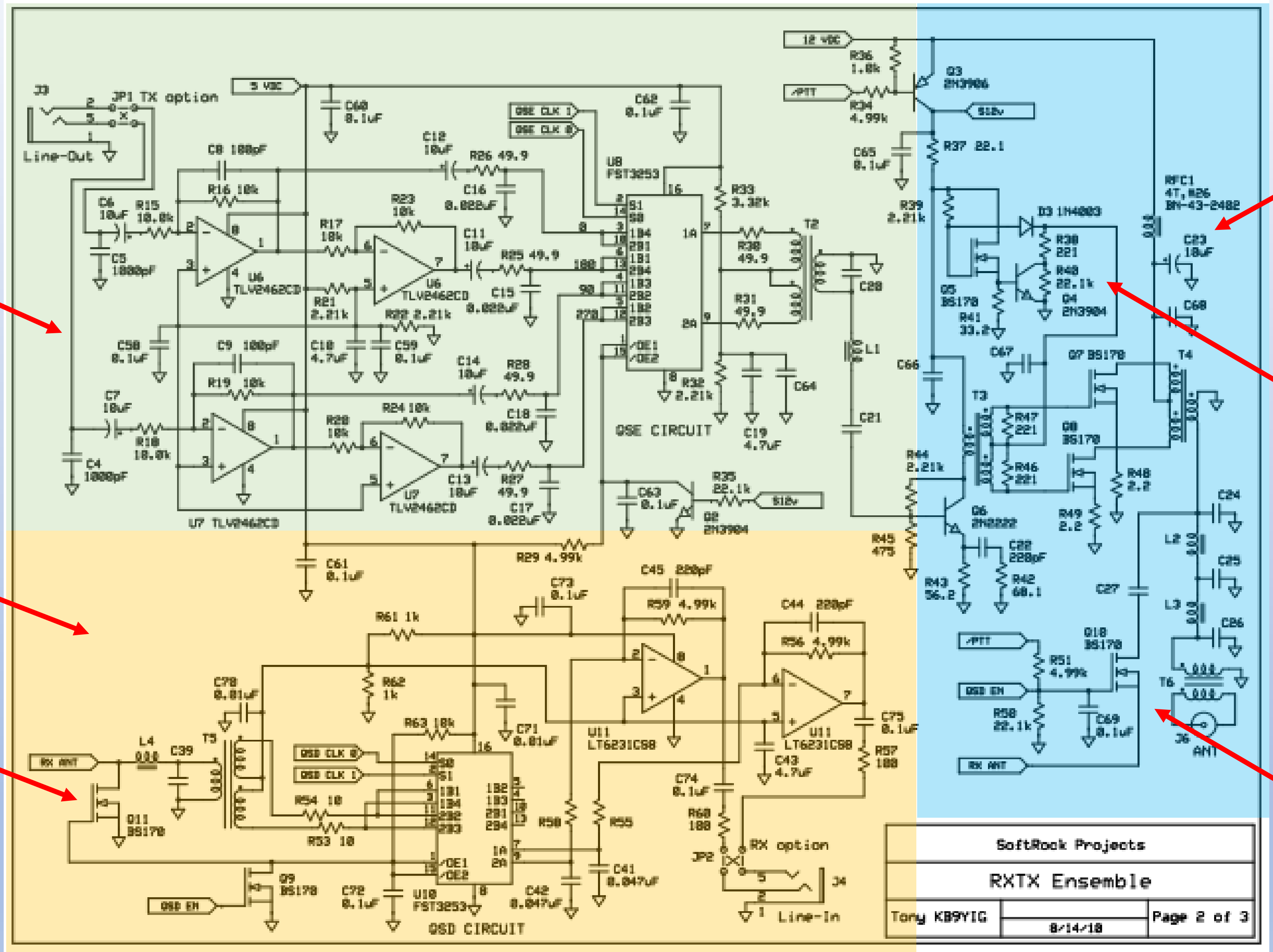
Host computer + SDR SW complete SDR demod-mod process

Ensemble RXTX

QSE Transmitter

QSD Receiver

Ant input blocked during Tx



TX Power Amplifier

Tx PA enabled during Tx

Ant path connected during Rx

Note: Shared two phase clock, CW keyer and power supply not shown.

uSDX SDR Transceiver

PE1NNZ hacked QSX transceiver (QRP labs) into SDR based transceiver

- About 70 components removed and SDR implemented with $3.00 Arduino uC
- WB2CBA took PE1NNZ hacked QSX design and created custom kit
- Single band. Requires separate board for each band (80 to 17 meters)
- Complete SDR transceiver, no external computer needed
- Features VFO and user interface

This kit is available for $55 USD from
https://shop.offline.systems/collections/frontpage/products/wb2cba-v1-02-kit-group-buy

SDR Receiver

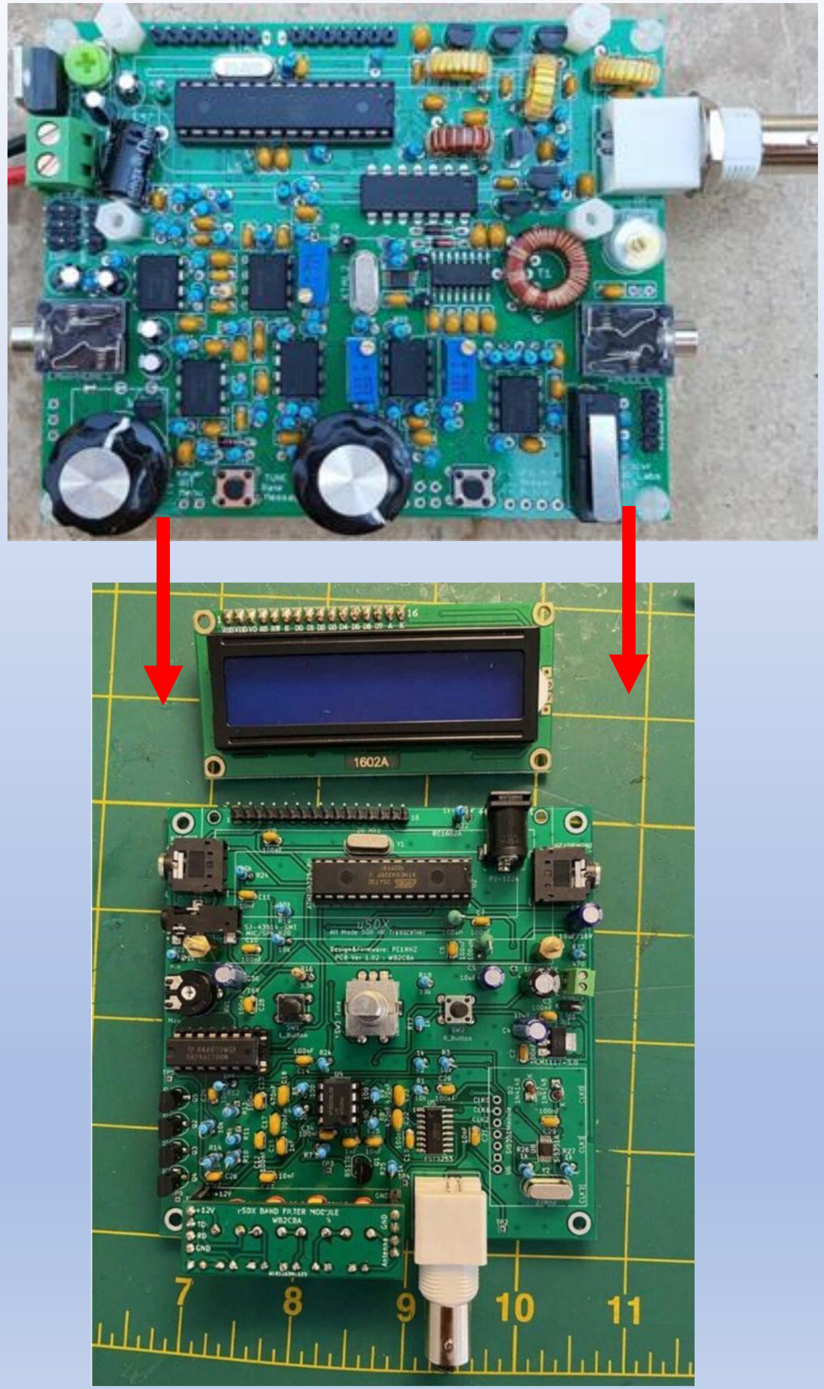
- Tayloe QSD front end similar to SoftRock Lite II Receiver
- I and Q signals processed onboard by Arduino processor to create AF output

SDR Transmitter

- Arduino takes microphone input directly and create I and Q signals internally
- Arduino processor outputs class E RF drive into 3 x BS170 (3 watts out)

Arduino processor connected to 16 x 2 alpha-numeric display

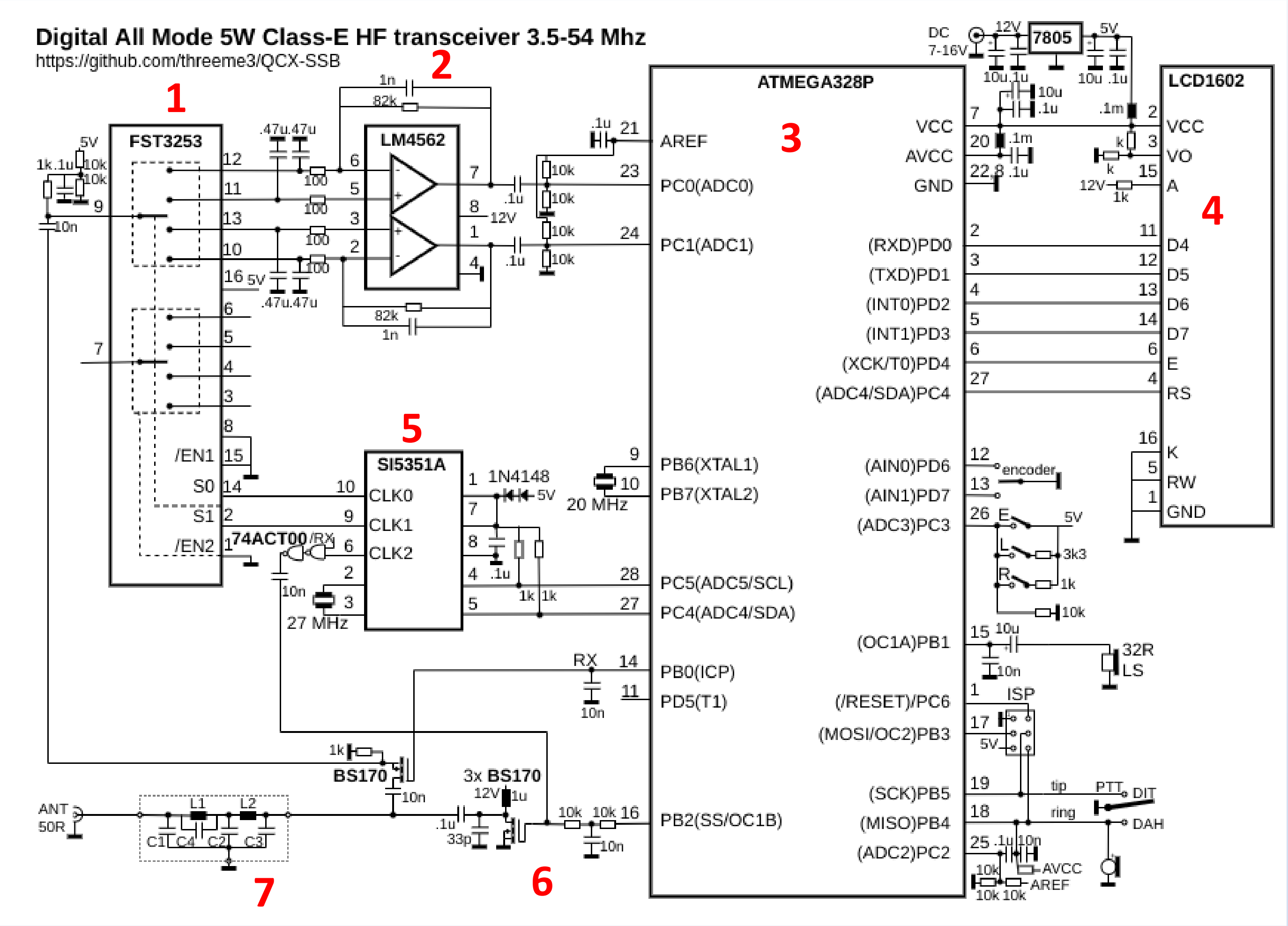
- Provides band select, frequency tune, mode (AM/USB/LSB) and other functions
- CW Decoder



uSDX SDR Transceiver

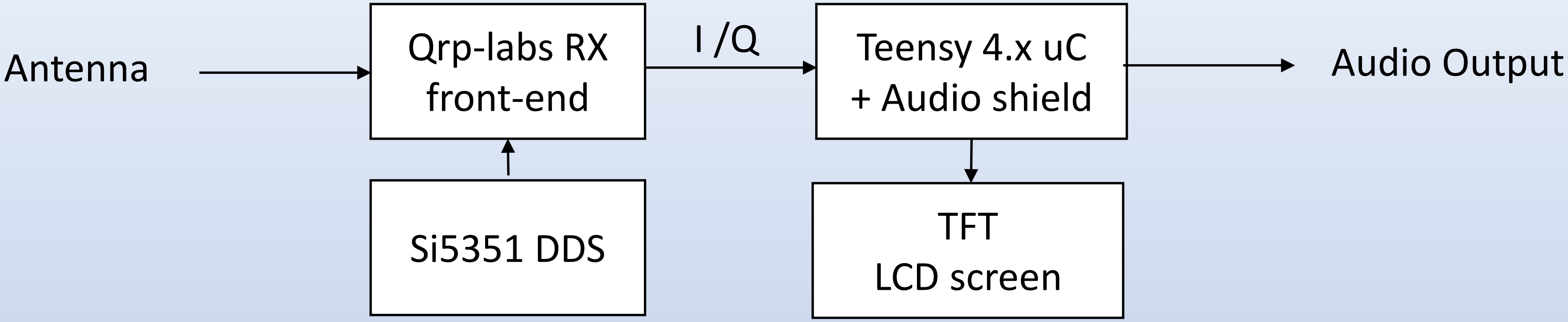
SDR radio consists of 7 principal components:

1) Tayloe Rx sampler
2) Sample and hold LPF
3) Arduino uC (SDR radio and user control)
4) 16 x 2 LCD (displays frequency, mode, etc)
5) S5351A master oscillator
6) 3 x BS170 Output stage
7) Bandpass filter



Homebrew SDR

One approach for homebrew is the integration of pre-built modules into a SDR Rx or transceiver. There is a *Facebook group dedicated to this.



Antenna

Qrp-labs RX front-end

I /Q

Teensy 4.x uC + Audio shield

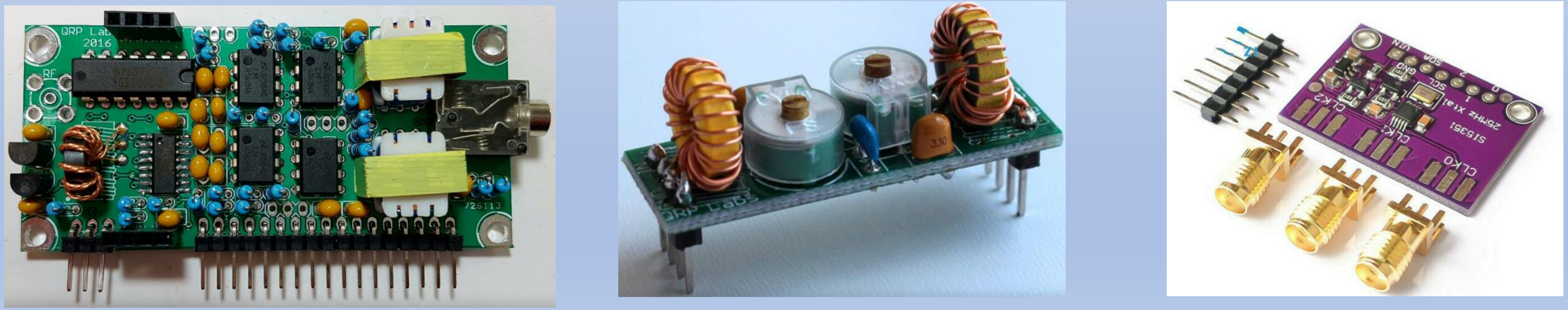
Audio Output

Si5351 DDS

TFT LCD screen

For example, the following subassemblies are relatively cheap:

- Tayloe QSD receiver front end ➔ $25 from www.qrp-labs.com
- One bandpass filter is included with RX kit. Additional filters ➔ $5 each from qrp-labs
- Si5351 clock generator (8 KHz – 160 MHz) ➔ $5 from eBay



- Examples here taken from Keiths_SDR on facebook https://groups.io/g/keithsdr

Homebrew SDR

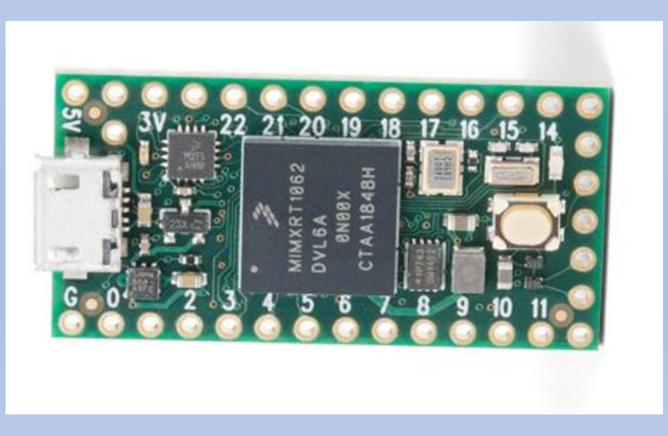
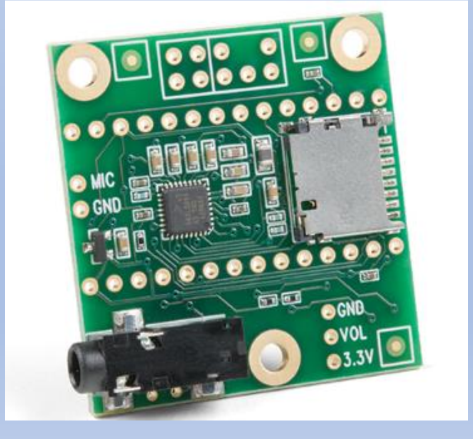
At this point two options are available for processing the I and Q signals.

- Connect audio to computer with SDR software
- Use dedicated SDR processor

Teensy uC offers more power and memory than Arduino controller

- Teensy 4.0 ➔$30 Cdn
- Teensy Audio shield ➔$30 Cdn
- TFT Display (allows for spectral or waterfall displays)

SDR SW is available on site of SDR group. Must register to access it.

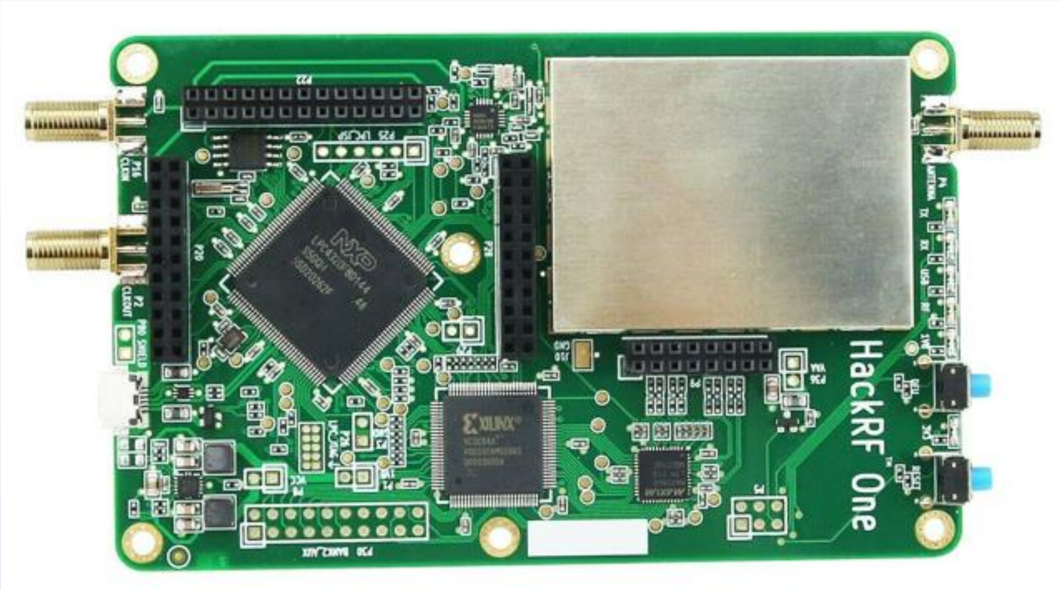
Hack RF One

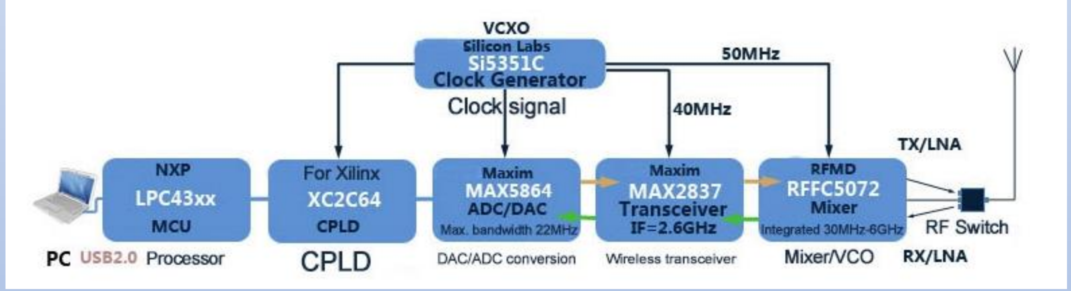
*Hack RF One dev board by Great Scott Gadgets ($100 USD)
- Half-duplex transceiver
- 1 MHz to 6 GHz operating frequency

RX direction
- Wideband mixer 1 MHz - 6 GHz
- 2.3 – 2.7 GHz IF amplifier
- CPLD (Complex programmable Logic Device)
  Baseband filtering and IQ creation via programmable HW
- IQ processing done with external computer

TX direction
- Same flow in reverse
- Transmitter is 30 mW maximum output

Needs power amplifier and bandpass filter to be useable for amateur radio

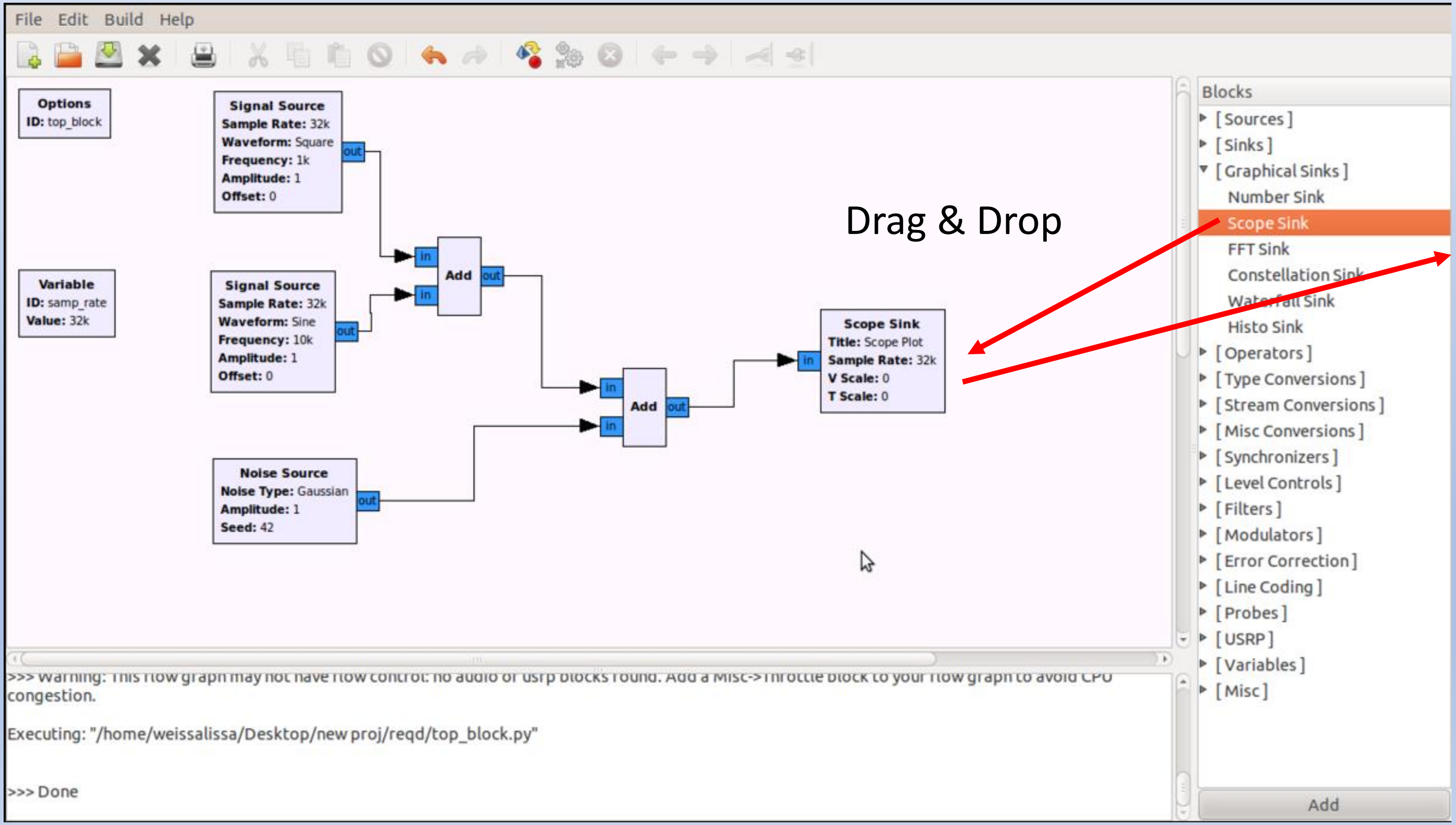




* Very popular with hackers for spoofing GPS, hacking military radio systems, cloning car key FOBs, etc.

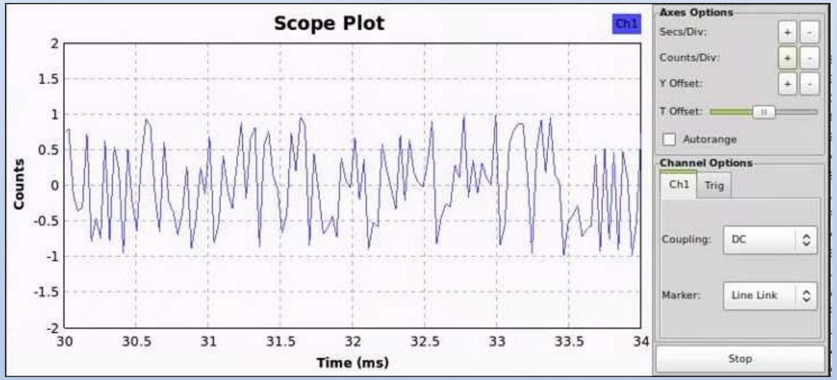
GNU Radio

- GNU Radio: Free open-source software development toolkit
- Provides predefined signal processing and test blocks to implement and test SDR.
- Connects to HW platform via source (output) and sink (input) blocks
- GNU Radio allows you to draw a radio topology and export it to the Hack RF One board.
- Once block diagram is drawn, it creates python code and is executed on Hack RF One.



```
>>> Warning: This flow graph may not have flow control: no audio or usrp blocks found. Add a Misc->Throttle block to your flow graph to avoid CPU
congestion.

Executing: "/home/weissalissa/Desktop/new proj/reqd/top_block.py"

>>> Done
```

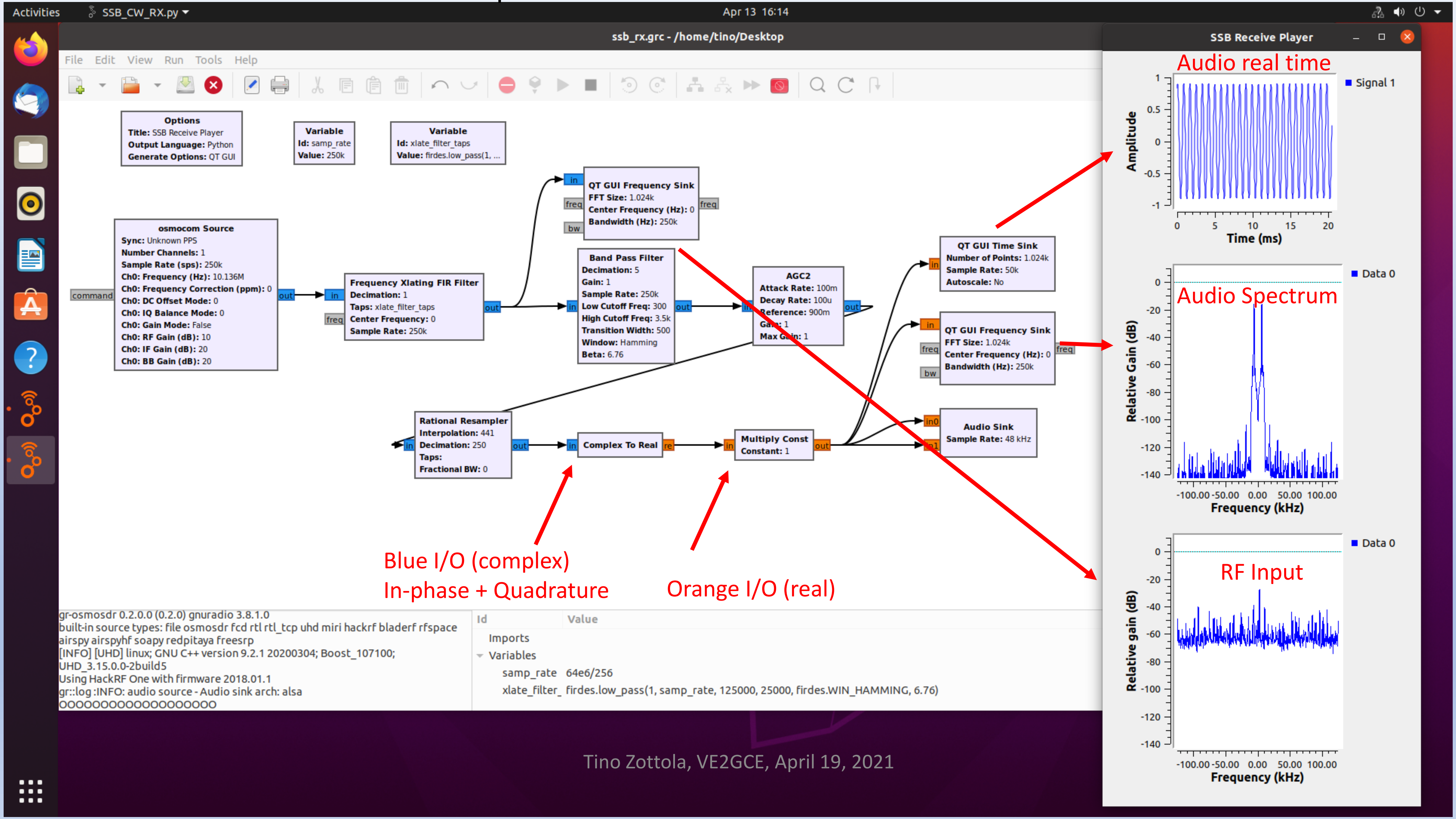
Drag & Drop



Test UI popup

GNU Radio

SSB/CW receiver derived from OZ9AEC.net example.



Options
Title: SSB Receive Player
Output Language: Python
Generate Options: QT GUI

Variable
Id: samp_rate
Value: 250k

Variable
Id: xlate_filter_taps
Value: firdes.low_pass(1, ...

osmocom Source
Sync: Unknown PPS
Number Channels: 1
Sample Rate (sps): 250k
Ch0: Frequency (Hz): 10.136M
Ch0: Frequency Correction (ppm): 0
Ch0: DC Offset Mode: 0
Ch0: IQ Balance Mode: 0
Ch0: Gain Mode: False
Ch0: RF Gain (dB): 10
Ch0: IF Gain (dB): 20
Ch0: BB Gain (dB): 20

Frequency Xlating FIR Filter
Decimation: 1
Taps: xlate_filter_taps
Center Frequency: 0
Sample Rate: 250k

QT GUI Frequency Sink
FFT Size: 1.024k
Center Frequency (Hz): 0
Bandwidth (Hz): 250k

Band Pass Filter
Decimation: 5
Gain: 1
Sample Rate: 250k
Low Cutoff Freq: 300
High Cutoff Freq: 3.5k
Transition Width: 500
Window: Hamming
Beta: 6.76

AGC2
Attack Rate: 100m
Decay Rate: 100u
Reference: 900m
Gain: 1
Max Gain: 1

QT GUI Time Sink
Number of Points: 1.024k
Sample Rate: 50k
Autoscale: No

QT GUI Frequency Sink
FFT Size: 1.024k
Center Frequency (Hz): 0
Bandwidth (Hz): 250k

Rational Resampler
Interpolation: 441
Decimation: 250
Taps:
Fractional BW: 0

Complex To Real

Multiply Const
Constant: 1

Audio Sink
Sample Rate: 48 kHz

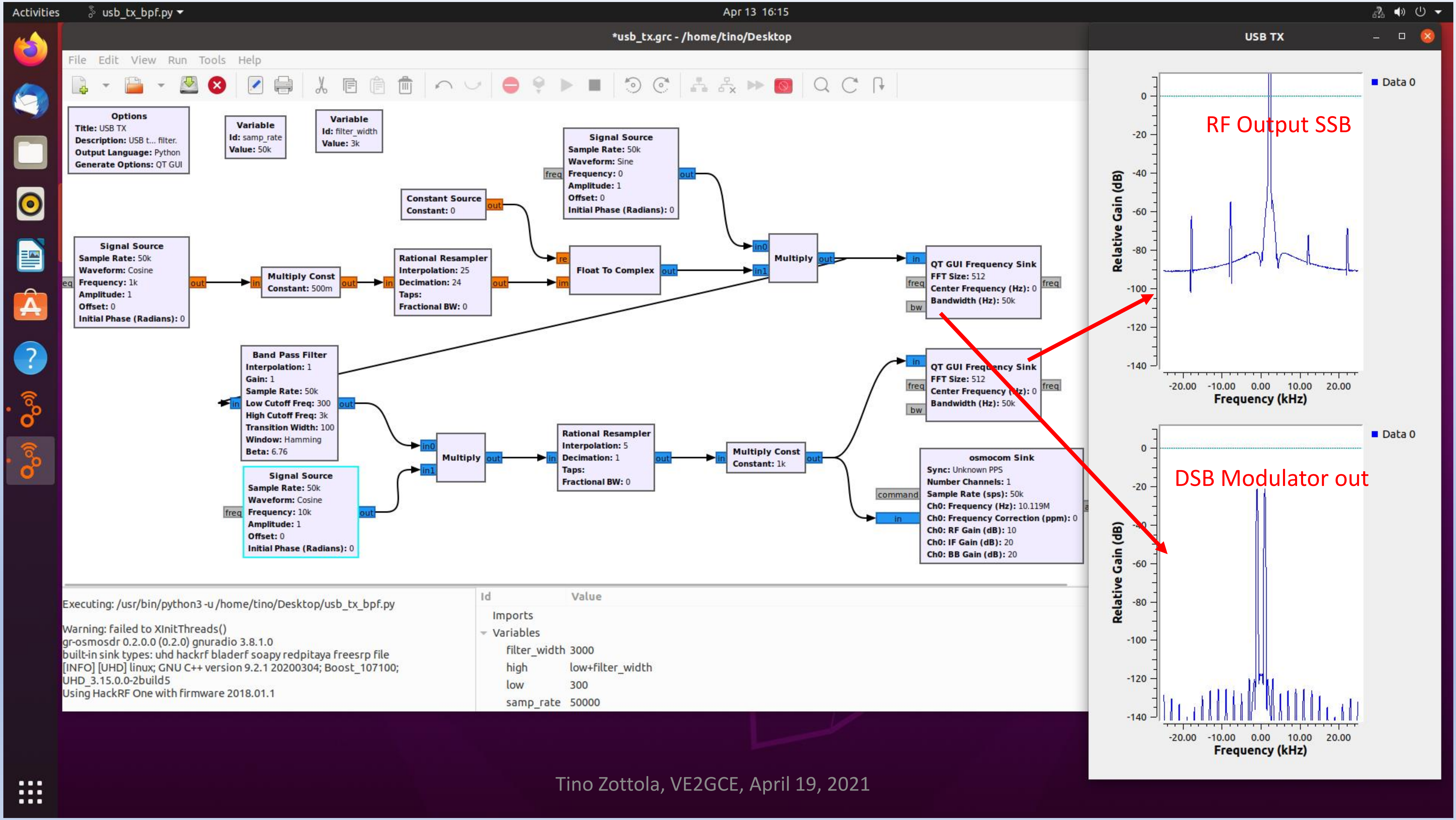
```
gr-osmosdr 0.2.0.0 (0.2.0) gnuradio 3.8.1.0
built-in source types: file osmosdr fcd rtl rtl_tcp uhd miri hackrf bladerf rfspace
airspy airspyhf soapy redpitaya freesrp
[INFO] [UHD] linux; GNU C++ version 9.2.1 20200304; Boost_107100;
UHD_3.15.0.0-2build5
Using HackRF One with firmware 2018.01.1
gr::log :INFO: audio source - Audio sink arch: alsa
OOOOOOOOOOOOOOOOOOOO
```

Id	Value
Imports	
Variables	
samp_rate	64e6/256
xlate_filter_	firdes.low_pass(1, samp_rate, 125000, 25000, firdes.WIN_HAMMING, 6.76)

Tino Zottola, VE2GCE, April 19, 2021

GNU Radio

SSB transmitter derived from OZ9AEC.net example.



Options
Title: USB TX
Description: USB t... filter.
Output Language: Python
Generate Options: QT GUI

Variable
Id: samp_rate
Value: 50k

Variable
Id: filter_width
Value: 3k

Signal Source
Sample Rate: 50k
Waveform: Sine
Frequency: 0
Amplitude: 1
Offset: 0
Initial Phase (Radians): 0

Constant Source
Constant: 0

Signal Source
Sample Rate: 50k
Waveform: Cosine
Frequency: 1k
Amplitude: 1
Offset: 0
Initial Phase (Radians): 0

Multiply Const
Constant: 500m

Rational Resampler
Interpolation: 25
Decimation: 24
Taps:
Fractional BW: 0

Float To Complex

Multiply

QT GUI Frequency Sink
FFT Size: 512
Center Frequency (Hz): 0
Bandwidth (Hz): 50k

Band Pass Filter
Interpolation: 1
Gain: 1
Sample Rate: 50k
Low Cutoff Freq: 300
High Cutoff Freq: 3k
Transition Width: 100
Window: Hamming
Beta: 6.76

Signal Source
Sample Rate: 50k
Waveform: Cosine
Frequency: 10k
Amplitude: 1
Offset: 0
Initial Phase (Radians): 0

Multiply

Rational Resampler
Interpolation: 5
Decimation: 1
Taps:
Fractional BW: 0

Multiply Const
Constant: 1k

QT GUI Frequency Sink
FFT Size: 512
Center Frequency (Hz): 0
Bandwidth (Hz): 50k

osmocom Sink
Sync: Unknown PPS
Number Channels: 1
Sample Rate (sps): 50k
Ch0: Frequency (Hz): 10.119M
Ch0: Frequency Correction (ppm): 0
Ch0: RF Gain (dB): 10
Ch0: IF Gain (dB): 20
Ch0: BB Gain (dB): 20

```
Executing: /usr/bin/python3 -u /home/tino/Desktop/usb_tx_bpf.py

Warning: failed to XInitThreads()
gr-osmosdr 0.2.0.0 (0.2.0) gnuradio 3.8.1.0
built-in sink types: uhd hackrf bladerf soapy redpitaya freesrp file
[INFO] [UHD] linux; GNU C++ version 9.2.1 20200304; Boost_107100;
UHD_3.15.0.0-2build5
Using HackRF One with firmware 2018.01.1
```

Id	Value
Imports	
Variables	
filter_width	3000
high	low+filter_width
low	300
samp_rate	50000

RF Output SSB

DSB Modulator out

Tino Zottola, VE2GCE, April 19, 2021

Conclusion / Recommendations

SDR offer many benefits:

- Simpler radios requiring fewer physical components
- Better performance using QSD and QSE front/back ends; more efficient conversion and higher Q
- Obsolescence is delayed, since new features (or protocols) can be added via software upgrade

Several options for Homebrew SDR.

- SDR kits are available as low as $20 USD (receiver) or $55 USD (transceiver)
- Homebrew: Use pre-built modules + Teensy processors for a simple but powerful SDR radio
- GNU Radio option uses HW platform “Hack RF One” with GNU Radio editor
    - No need to build any custom hardware
    - No knowledge of software programming
- Commercial programs (like SDR Radio) exist to interface I and Q, if you prefer external processing + interface

Resources

SDR Theory:

http://norcalqrp.org/files/Tayloe_mixer_x3a.pdf

http://www.arrl.org/files/file/Technology/tis/info/pdf/020708qex013.pdf

http://www.arrl.org/files/file/Technology/tis/info/pdf/020910qex010.pdf

http://www.arrl.org/files/file/Technology/tis/info/pdf/021112qex027.pdf

http://www.arrl.org/files/file/Technology/tis/info/pdf/030304qex020.pdf

http://pe1nnz.nl.eu.org/2013/05/direct-ssb-generation-on-pll.html

User Groups

https://groups.io/g/keithsdr Canadian SDR builder group. Homebrew SDR around QRP Labs Rx module

https://groups.io/g/rpitx/topics RTL-SDR group

https://groups.io/g/ucx/topics uSDX group

https://wiki.gnuradio.org/index.php/Main_Page

Kits and Build descriptions

https://antrak.org.tr/blog/projeler/usdx-an-arduino-based-sdr-all-mode-hf-transceiver-pcb-iteration-v1-02/

https://github.com/threeme3/QCX-SSB

http://www.wb5rvz.org/

http://www.fivedash.com/

https://greatscottgadgets.com/hackrf/one/

https://wiki.gnuradio.org/index.php/Tutorials

https://github.com/mossmann/hackrf/wiki/Getting-Started-with-HackRF-and-GNU-Radio

Questions ?