Introduction to Software Defined Radio

Fabian Kaup (DK2FK)

DARC Ortsverband F03, Darmstadt

07.10.2016

What is Software Defined Radio?

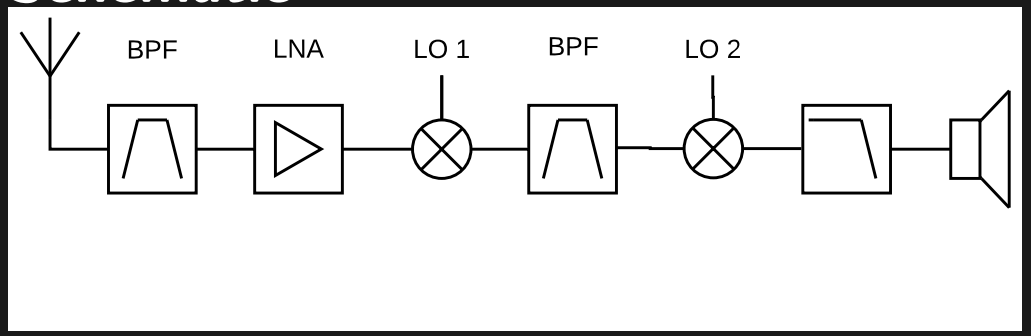
Definition

Software-defined radio (SDR) is a radio communication system where components that have been typically implemented in hardware (e.g. mixers, filters, amplifiers, modulators/demodulators, detectors, etc.) are instead implemented by means of software on a personal computer or embedded system.[a]

[a] Software Defined Radio: Architectures, Systems and Functions (Markus Dillinger, Kambiz Madani, Nancy Alonistioti) Page xxxiii (Wiley & Sons, 2003, ISBN 0-470-85164-3)

Conventional Receivers

Schematic

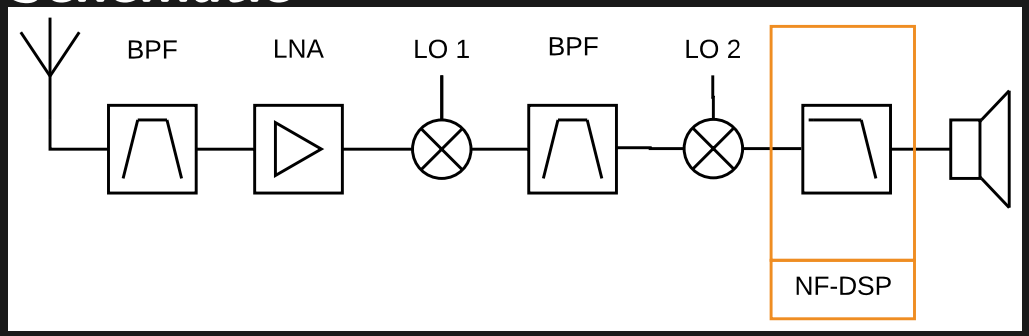


- Superheterodyne receiver
- Completely analog design
- High complexity
- High accuracy of components required

Example: FT 817

Superheterodyne with DSP

Schematic



- DSP functionality in AF
- High complexity
- High accuracy of components required
- Improved NF quality

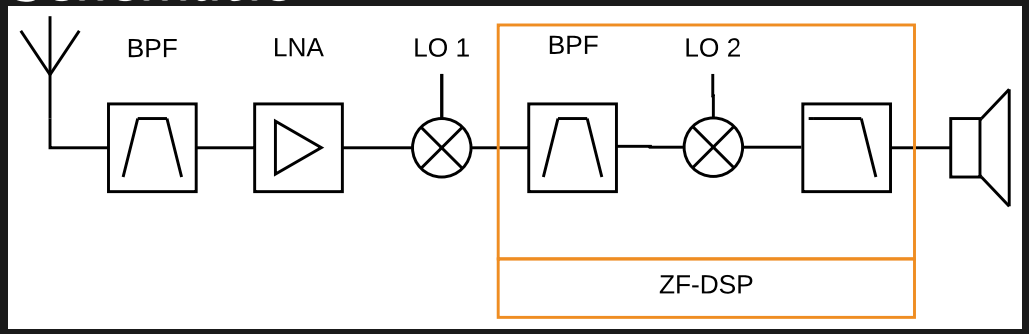
Example: IC 7600

Photo by Kjell, SM0FOB, src:

`http://picasaweb.google.com/sm0fob/SomeHamRadioActivity#5342440689000150978`

TRX with IF-DSP

Schematic



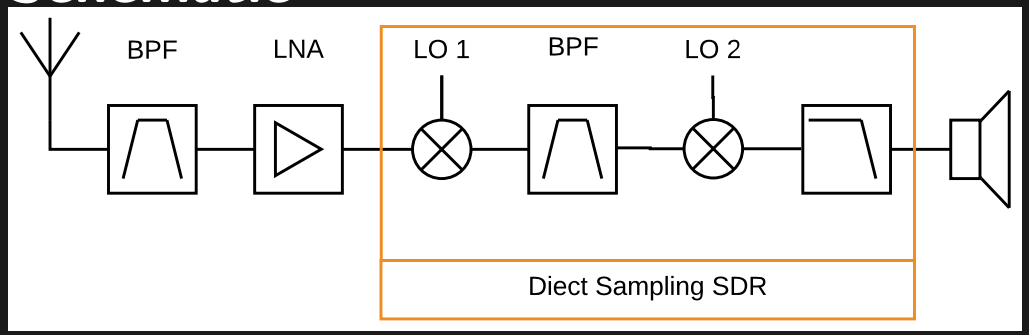
- Only one mixer/LO
- Reduced number of analog components
- Increased flexibility of filters and processing

Example: KX3

Photo by JR1CHU, `http://photozou.jp/photo/show/216071/224352880`

Direct Sampling TRX

Schematic



- No mixer/LO required
- Higher bandwidth
- Filter bank required (due to aliasing)

Example: Hamlab

Image removed due to unclear copyright

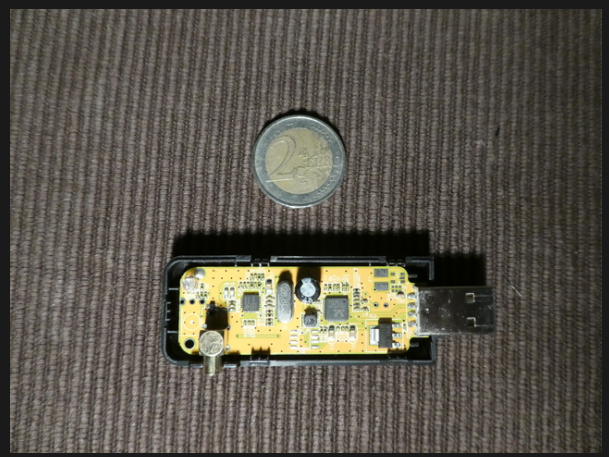
Why is SDR interesting?

- Higher bandwidth for digital applications
  $\rightarrow$ can monitor one, sometimes multiple bands
- Higher flexibility
  $\rightarrow$ can work on different modes
- Fast reconfigurability
  $\rightarrow$ but antenna may not be resonant...

RTL-SDR

Various models

- **Cost:** $\sim$ 10 €
- **Frequency range:** 50 - 1800 MHz
- **Bandwidth:** $<$ 2.4 MHz
- **Resolution:** 7-8 bit
- **Analog filters:** none
- **RX/TX:** RX only

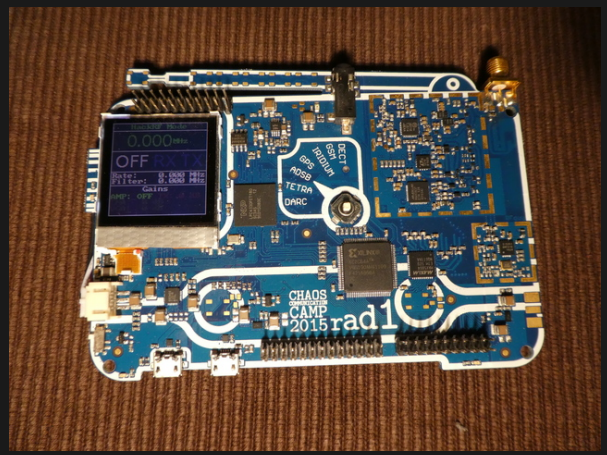


DVB-T USB dongle using the RTL 2832U controller and R820T tuner chip

Medium-level Boards

HackRF One/rad1o

- **Cost:** $\sim$ 300 €
- **Frequency range:** 50 MHz - 6 GHz
- **Bandwidth:** $\leq$ 20 MSps
- **Resolution:** 8 bit
- **Analog filters:** none
- **RX/TX:** Half duplex



rad1o

High-end Boards

USRP B200/B210

- **Cost:** $\sim$ 700 €
- **Frequency range:** 50 MHz - 6 GHz
- **Bandwidth:** $\leq$ 56 MSps
- **Resolution:** 12 bit
- **Analog filters:** none
- **RX/TX:** Full duplex

USRP B200

High-end Boards

LimeSDR

- **Cost:** $\sim$ 260 €
- **Frequency range:** 0.1 MHz to 3800 MHz
- **Bandwidth:** 61.44 MHz
- **Resolution:** 12 bit
- **Analog filters:** none
- **RX/TX:** 2RX/TX, full duplex

Image from https://www.crowdsupply.com/lime-micro/limesdr

Image removed due to copyright reasons.

LimeSDR

Other Boards

FlexRadio/ApacheLabs
for the ham radio operator

- **Cost:** $>$ 1000 €
- **Frequency range:** 30 kHz - 54 MHz
- **Bandwidth:** $\leq$ 96 kHz/ 7 MHz
- **Resolution:** 16 bit
- **Analog filters:** Available
- **TX Power:** $\leq$ 100 W

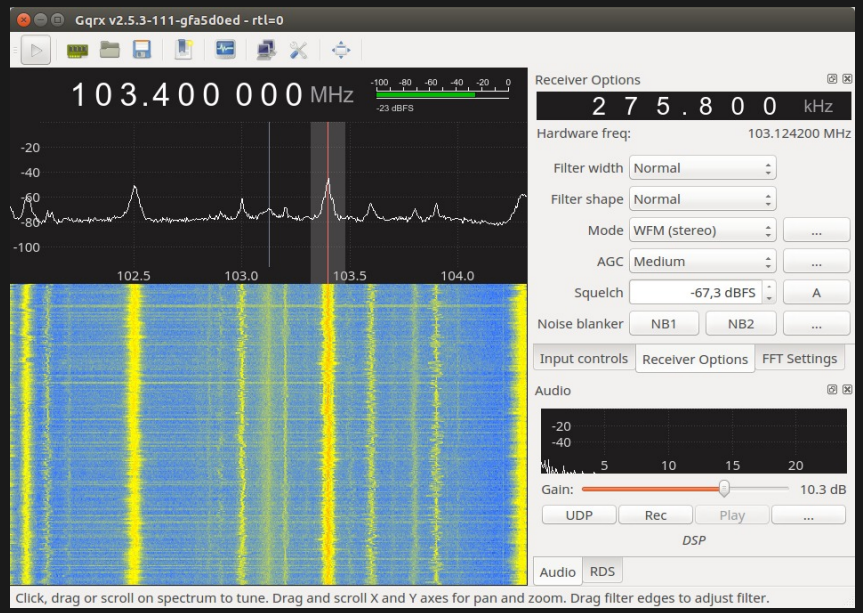
USRP Networked Series
for research and development

- **Cost:** $>$ 1500 €
- **Frequency range:** DC - 6 GHz
- **Bandwidth:** $\geq$ 100 MHz
- **Resolution:** 16 bit
- **Analog filters:** none
- **RX/TX:** Full duplex

HDSDR/SDR#/GQRX

General "software radios"

- Similar functionality to hardware radio
- Common demodulators included
- Can be "wired" to other tools (e.g. fldigi) using virtual audio cables/pipes
- Windows: HDSDR/SDR#/Gqrx
- Linux/OSX: GQRX

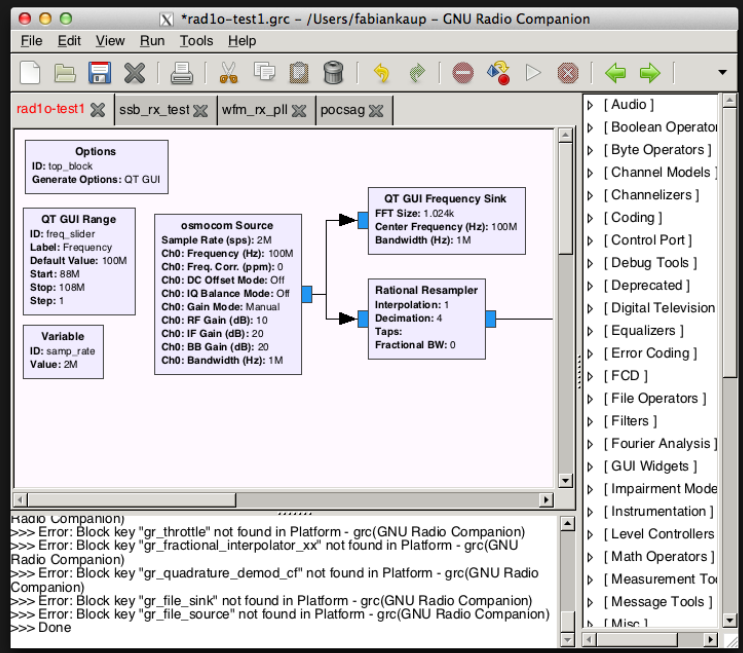


GQRX

gnuradio

Environment to connect functional blocks

- Blocks written in Python
- Well defined interfaces
- GUI: Gnuradio-companion
- GUI applications possible
- Wide variety of functions/ blocks available
- Works on real-time data and recorded samples

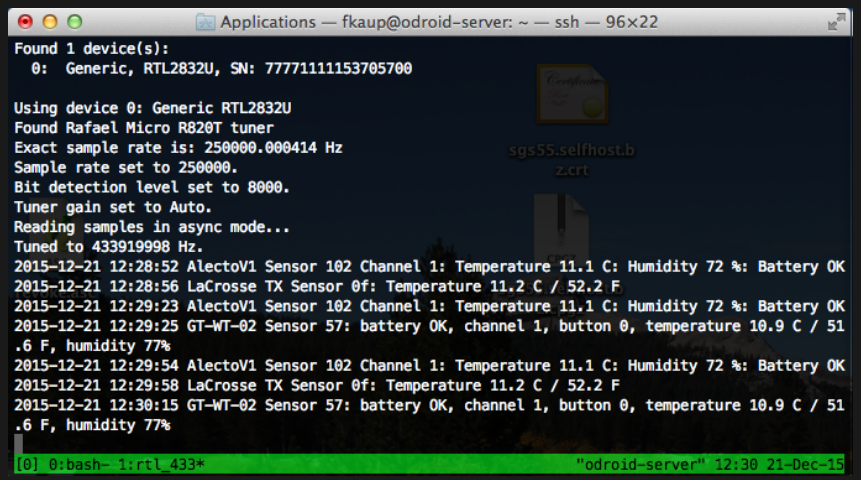


Gnuradio-companion

rtl_*

CLI tools compiled for DVB-T dongles with RTL chip

- **FM radio:** rtl_fm
- **RF power:** rtl_power
- **TCP server:** rtl_tcp
- **433 MHz ISM:** rtl_433
- **ADSB monitoring:** rtl_adsb, dump1090

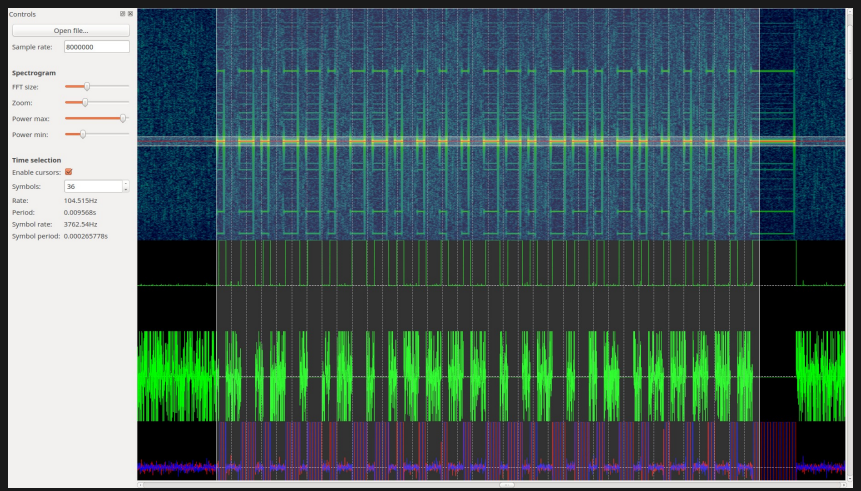


Weather monitoring using rtl_433

inspectrum

"tool for analysing captured signals, [...] from software-defined radio receivers." [1)]

- Works on recorded data
- Time and frequency domain analysis
- Tools to simplify sample identification
- Sampling of data
- Export of sampled data



Screenshot of inspectrum

1) `https://github.com/miek/inspectrum`

Conclusions

- Lots of projects under heavy development
- Hardware about to get affordable
- Combination of hardware and software engineering
- Unprecedented flexibility

Additional Resources

Tutorial:

- SDR tutorial: `https://greatscottgadgets.com/sdr/`
- Recordings from the Software-defined Radio Academy: `https://www.youtube.com/channel/UC1GAlgAQrkjeeLmIkCB8pgQ`

Hardware:

- RTL-SDR: `http://rtlsdr.org/`
- rad1o: `http://rad1o.de/`
- HackRF One: `https://greatscottgadgets.com/hackrf/`
- USRP B200/B210: `http://www.ettus.com/product/details/UB200-KIT`

Projects:

- RTL-SDR: `http://www.rtl-sdr.com/`
- Gnuradio: `http://gnuradio.org/`
- Inspectrum: `https://github.com/miek/inspectrum`
- . . . and many more

License