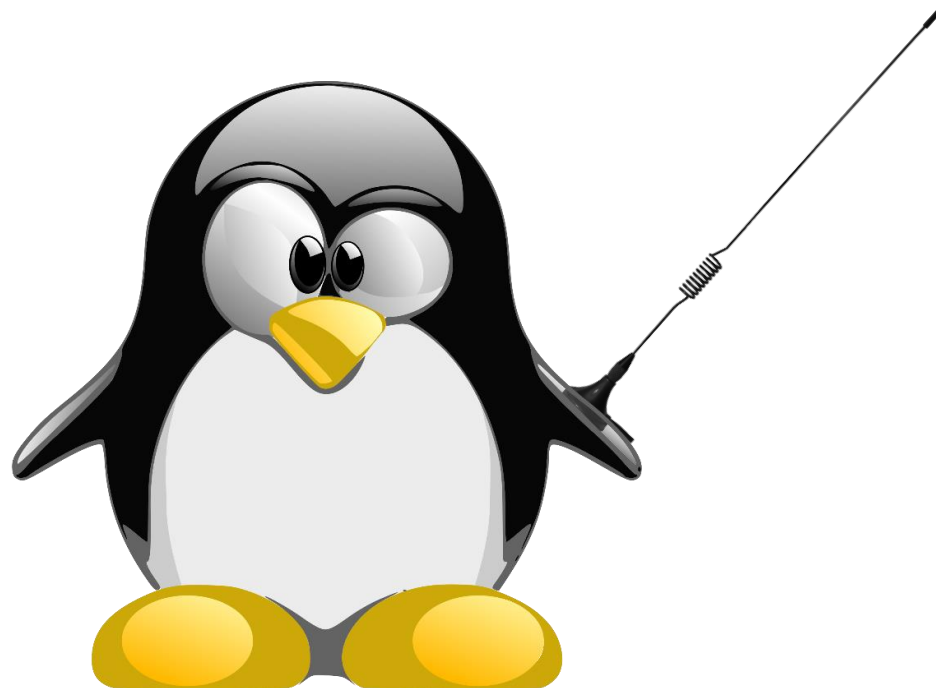


# Software Defined Radio

## Open Source Wireless Hacking



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



**15. Augsburger Linux-Infotag 2016**



**Jiska Classen**

Technische Universität Darmstadt  
Secure Mobile Networking Lab - SEEMOO  
Department of Computer Science  
Center for Advanced Security Research Darmstadt - CASED

Mornewegstr. 32  
D-64293 Darmstadt, Germany  
jclassen@seemoo.tu-darmstadt.de  
Tel.+49 6151 16-70924, Fax. +49 6151 16-70921  
<https://seemoo.de/jclassen>



---

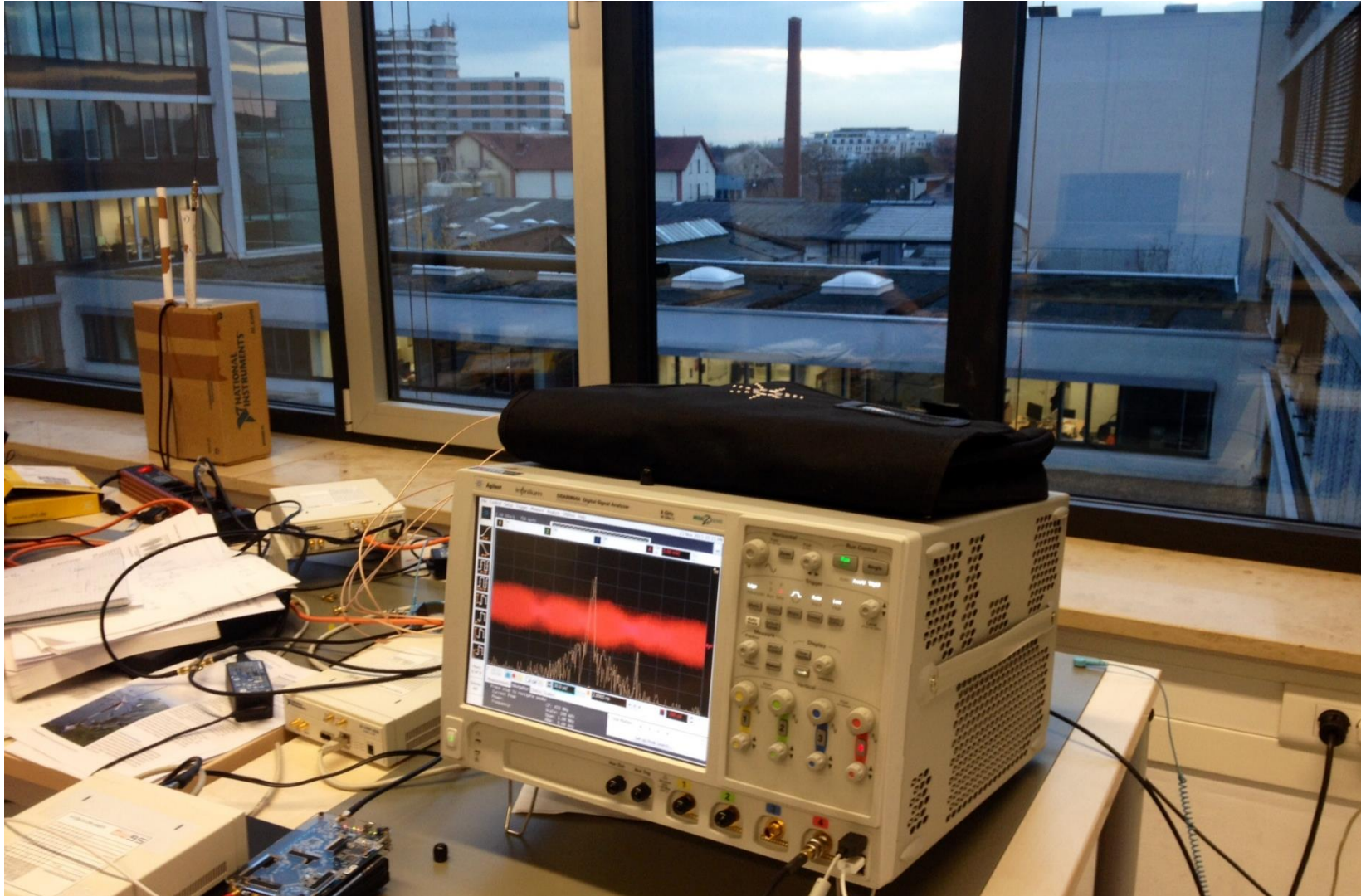
# Overview

- (1) Problem Statement
- (2) Hardware Overview
- (3) Interesting Frequencies
- (4) gqrx Demo
- (5) gnuradio Demo
- (6) Getting Started
- (7) Q&A

---

# **Problem Statement**

# Spectrum Analyzer or Oscilloscope



---

# Remaining Problems

Great hardware, goes up to 28GHz, 160MHz bandwidth 😊

In CROSSING, we do mobile device pairing and trust models, but...

- **Mobile experiments** – hard to move
- **Distributed experiments** – only one device

Working with students...

- Teaching with **30+ students** – only one device...
- Students should be able to do some wireless hacking with **hardware they can afford** after the course

---

# Hardware Overview

# USRP

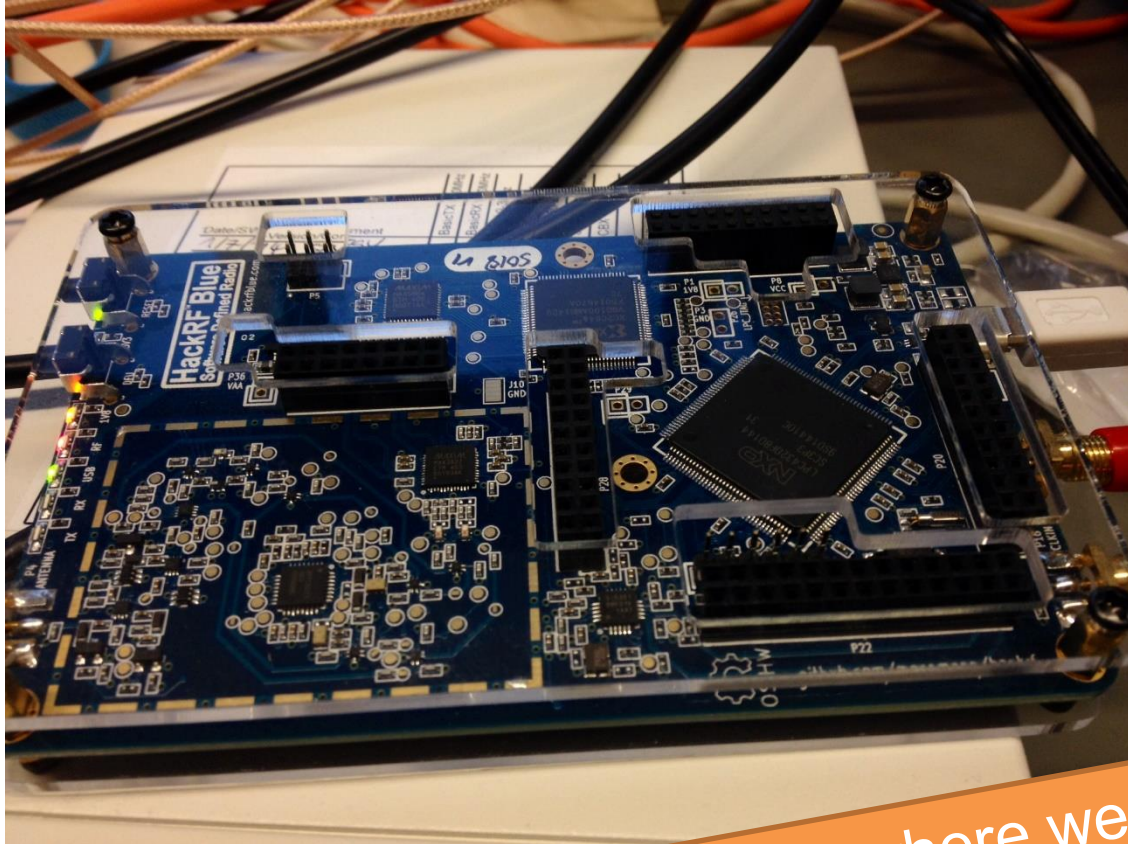
- Simultaneous transmission and reception
- Many different models available, from 700€
- Even within one model, different daughterboards are available
- Most popular for **research projects**
- Requires flashing a **Linux compatible** image  
(works with `uhd-host 3.9.3-1` in Debian testing)

For PhD projects

- What I brought today:
  - USRP **N210** – 1810€
  - **SBX** daughterboard – 495€  
400MHz-4.4GHz frequency range,  
40MHz bandwidth (=2 WiFi channels)
  - Also see: `uhd_usrp_probe -args addr=192.168.10.2`



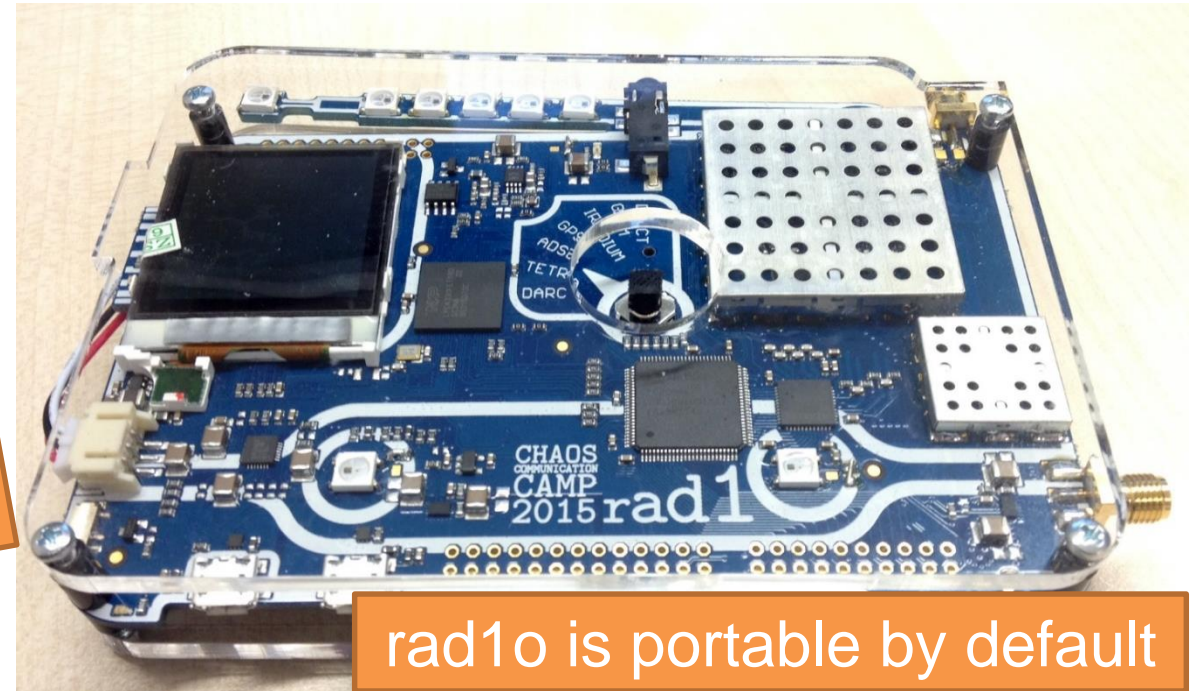
# rad1o badge / HackRF



In a price range where we can borrow it to students 😊

## HackRF Blue

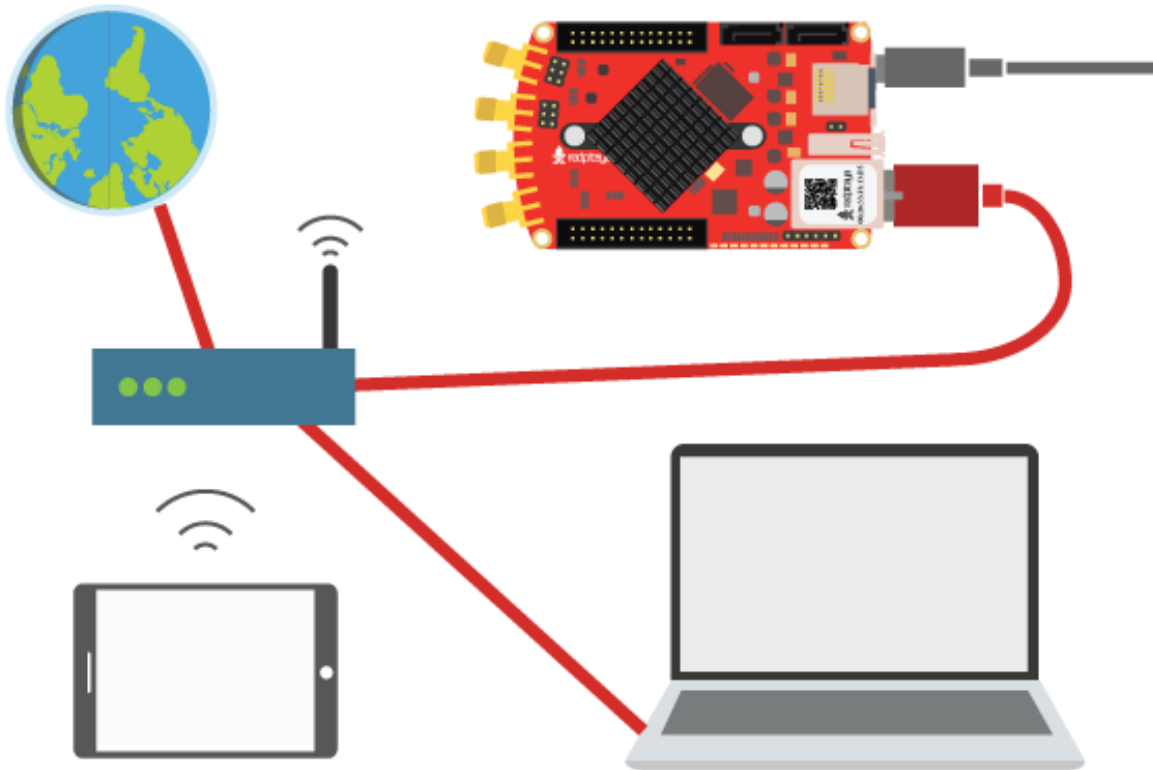
- Open source hardware
- Receiver or transmitter
- 1MHz-6GHz, 20Msps (**rad1o**: 1MHz-4GHz)
- 200€



rad1o is portable by default



# Red Pitaya



- Provides **open source applications** that run on the board:
  - Oscilloscope
  - Spectrum Analyzer
  - ...
- Close to typical software defined radio features, but more powerful
- **Low frequency range: 0-50MHz**
- 234€ on reichelt

In a price range where we can borrow it to students 😊

# DVB-T Sticks



- Receiver for 22MHz-2.2GHz, frequencies vary depending on the actual model,  $\sim 2$ Msp/s
- From 7€

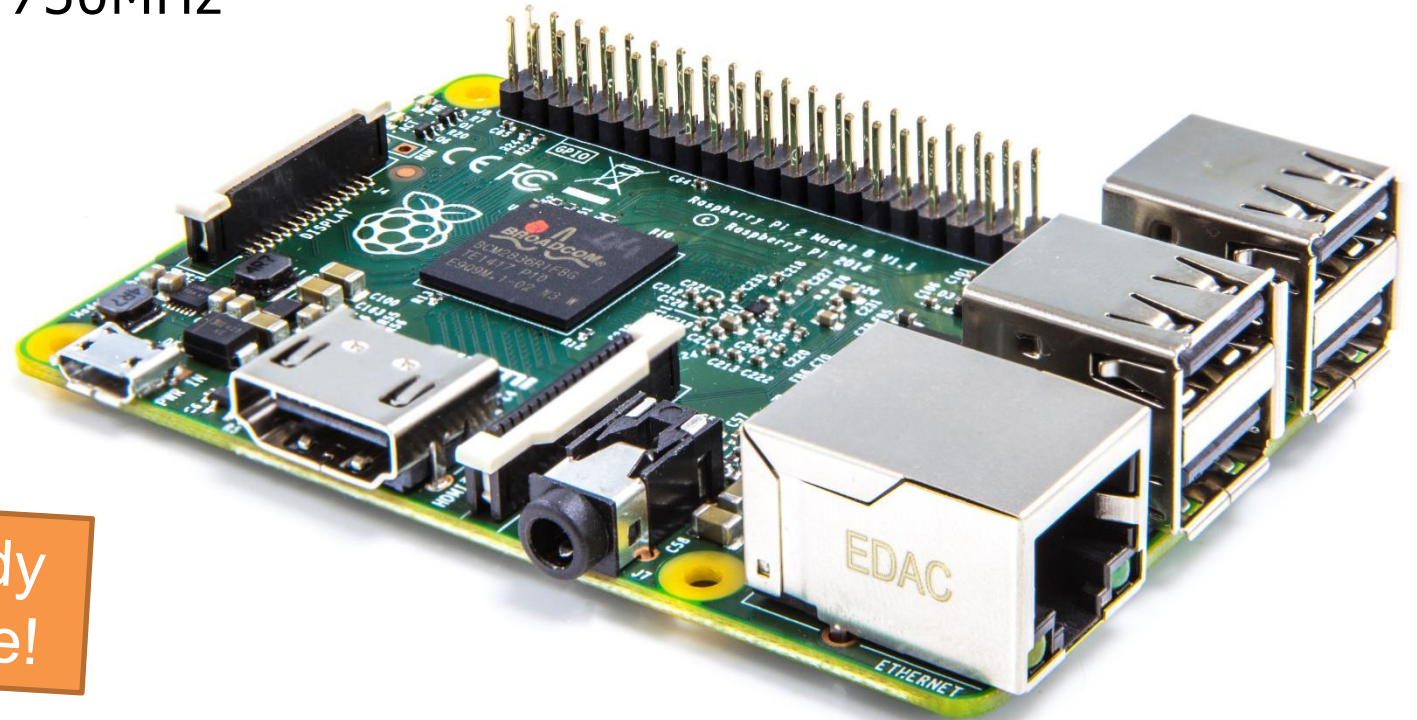
Tuner	Frequency range
Elonics E4000	52 – 1100 Mhz, 1250 – 2200 MHz
Rafael Micro R820T	24 – 1766 MHz
Rafael Micro R828D	24 – 1766 MHz
Fitipower FC0013	22 – 1100 MHz
Fitipower FC0012	22 – 948.6 MHz
FCI FC2580	146 – 308 MHz, 438 – 924 MHz

We can borrow these to a whole course, or they buy these themselves...

<http://sdr.osmocom.org/trac/wiki/rtl-sdr>

# rpitx

- Cheap transmitter for Raspberry Pi (B, B+ and PI2)
- Use GPIO pins + long wire as antenna
- Low frequency signals: 130kHz-750MHz
- 35€

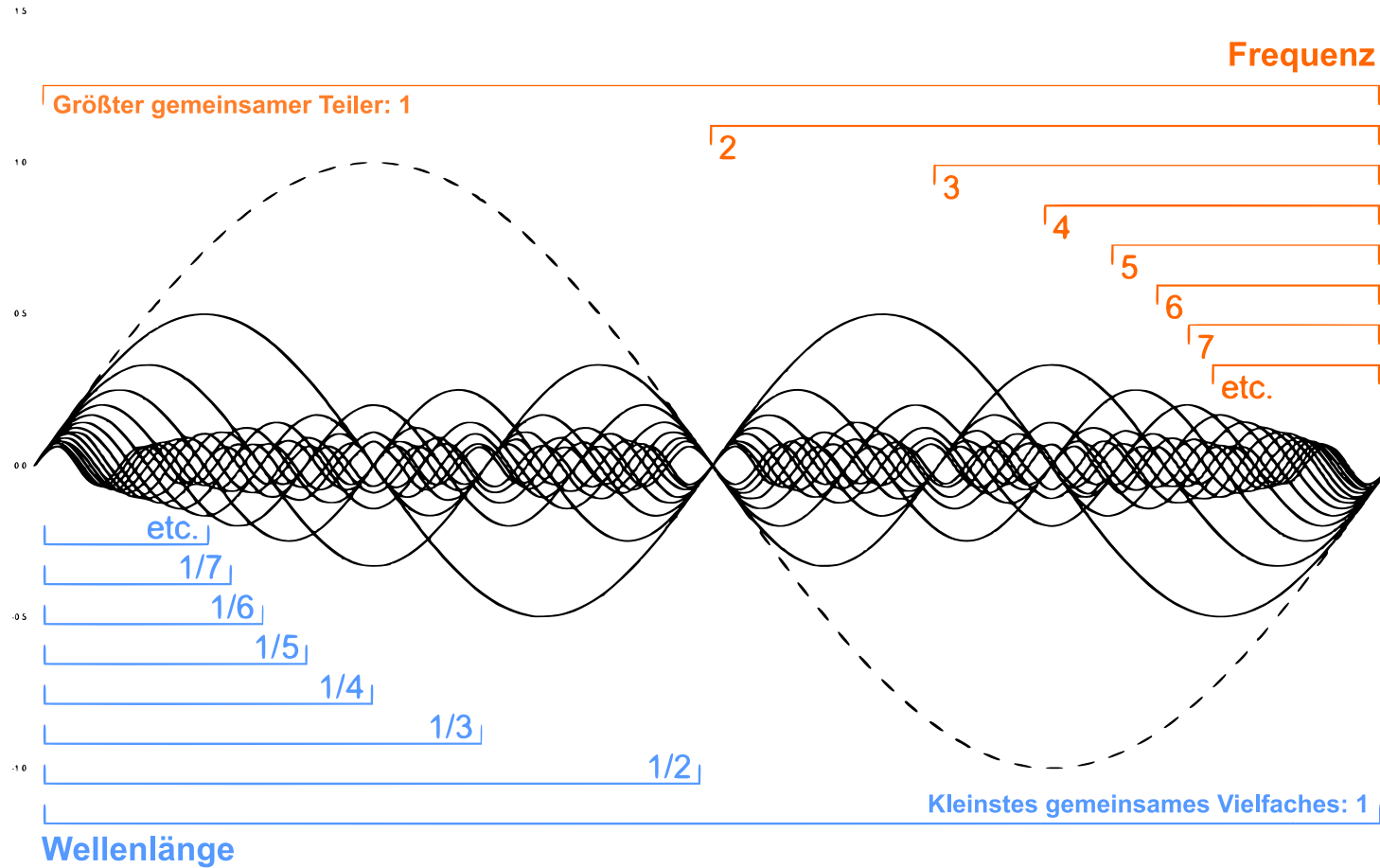


Many nerds already  
have this hardware!

---

# Interesting Frequencies

# Wavelength vs. Frequency



$$f \sim \frac{1}{\lambda}$$

[http://upload.wikimedia.org/wikipedia/commons/7/71/Missing\\_fundamental\\_Fourier\\_series.png](http://upload.wikimedia.org/wikipedia/commons/7/71/Missing_fundamental_Fourier_series.png)

# Low Frequency (2200m)

- Long wavelength requires huge antennas
- Transmitter for „Deutschlandfunk“: 153 kHz (1960m wavelength) is 363m high



LF

MF

HF

VHF

UHF

SHF

EHF

---

# Medium Frequency (160m)



---

# High Frequency

## (80m, 40m, 30m, 20m, 17m, 15m, 12m, 10m)

- 80m, 40m, 20m used for long distances in ham radio (DX)
- Transmissions from Europe to USA or even Japan possible



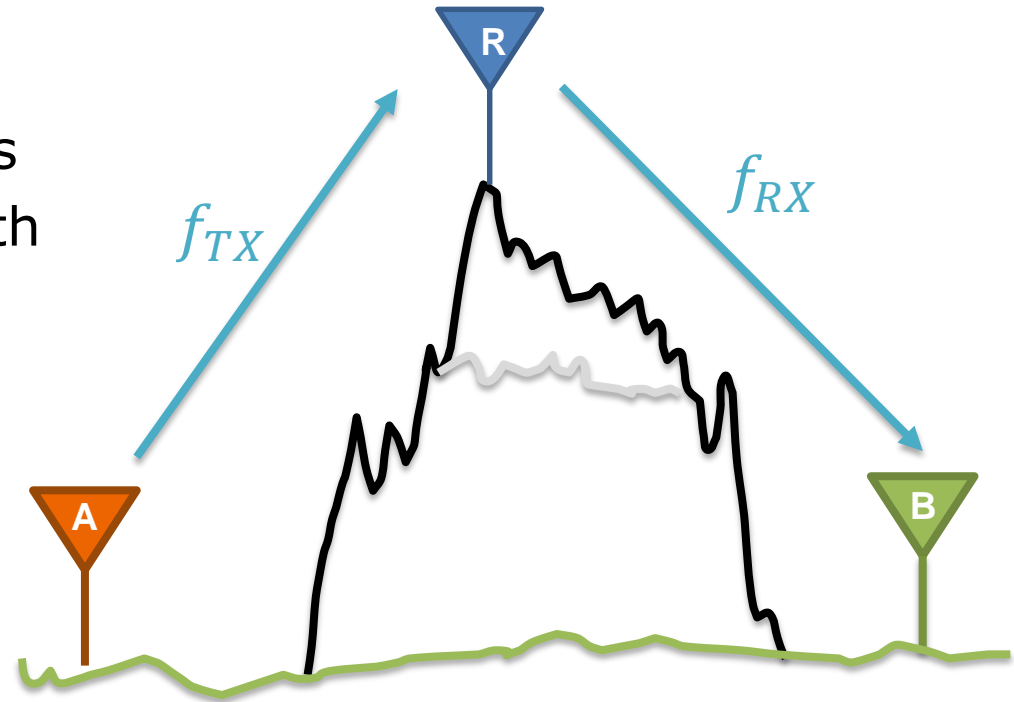


# Very High Frequency (6m, 2m)

- 2m and 70cm used for handheld receivers
- Small sizes possible
- **Relays** required for longer distances
- FM radio stations: ca. 3m wavelength



<https://www.flickr.com/photos/alexkerhead/3608747482>



---

# Ultra High Frequency (70cm, 23cm, 13cm)

- 12.5cm: 2.4GHz **WLAN**
- 900MHz and 1.8GHz **GSM**



---

# Super High Frequency (9cm, 6cm, 3cm, 1.2cm)

- 6cm: 5GHz WLAN



---

# Millimeter Wave (6mm, 4mm, 2.5mm, 2mm, 1.2mm)

- mmWave/60GHz WLAN
- Only a few meters range
- Walls etc. completely block the signal
- Typical application scenarios are indoor, e.g. wireless docking stations



# Hardware Capabilities

TX



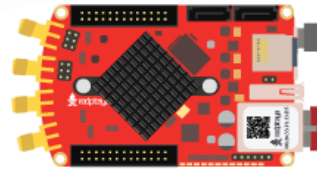
RX



RX|TX



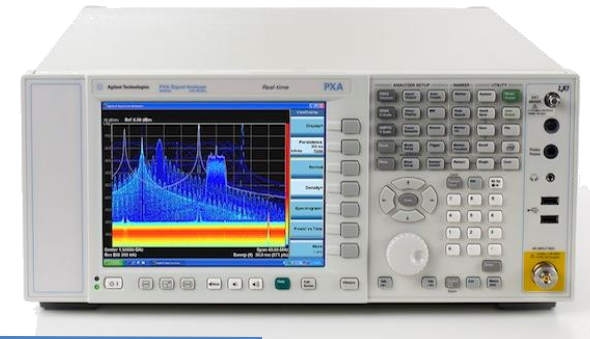
RX&TX



RX&TX



RX



LF

MF

HF

VHF

UHF

SHF

EHF

---

# Frequenznutzungsplan

Details, **wer** welche **Frequenz** mit welcher **Betriebsart** und mit welcher **Leistung** nutzen darf, sind dem Frequenznutzungsplan der Bundesnetzagentur zu entnehmen.

[http://www.bundesnetzagentur.de/DE/Sachgebiete/Telekommunikation/Unternehmen\\_Institutionen/Frequenzen/Grundlagen/Frequenzplan/frequenzplan-node.html](http://www.bundesnetzagentur.de/DE/Sachgebiete/Telekommunikation/Unternehmen_Institutionen/Frequenzen/Grundlagen/Frequenzplan/frequenzplan-node.html)

---

**Demo**  
**gqrx**

---

# Features

- Signal reception and **capture**
- Basic **demodulation** schemes, e.g. AM, FM, SSB
- Compatible to **HackRF, rad1o, Red Pitaya, DVB-T sticks** and more
- Typical application: check if signal reception is working, signal processing in external software



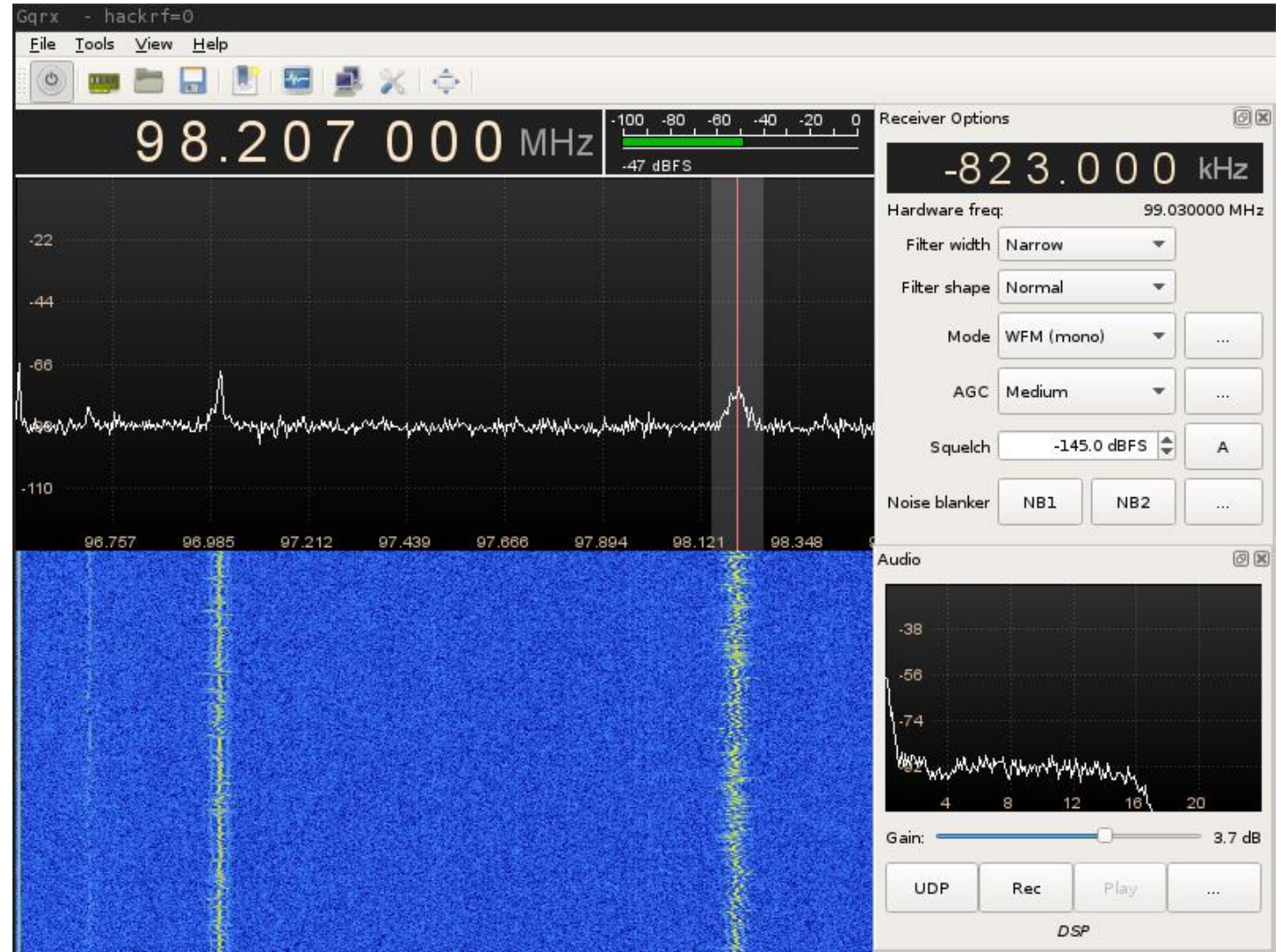
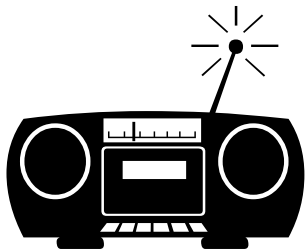
---

# Demo

- Receive nearby FM radio stations (DVB-T, rad1o)
- Check frequencies of GSM stations (DVB-T, USRP, rad1o)
- Check frequencies of WiFi access points (rad1o, USRP)

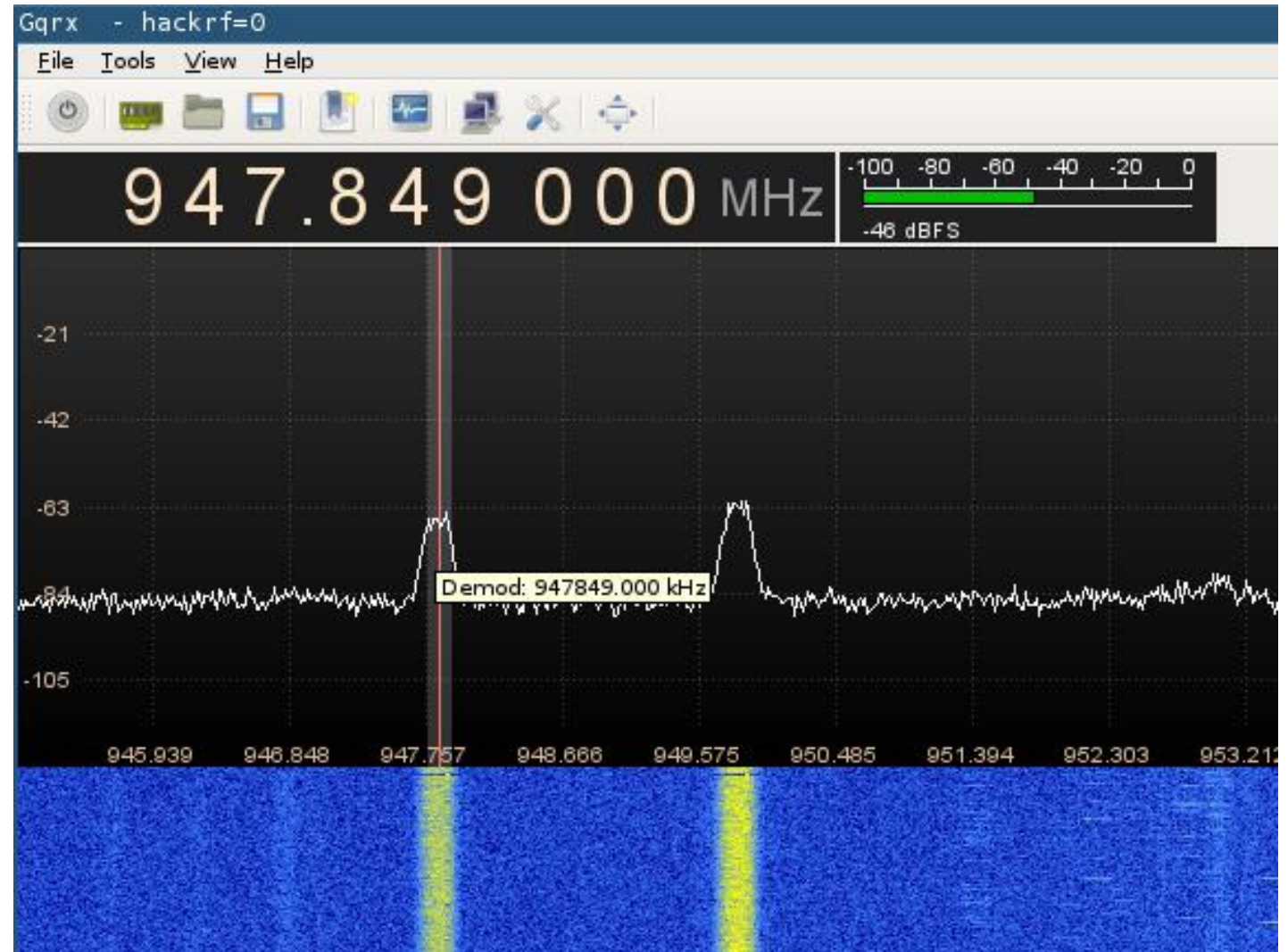
# Listen to the radio

- German FM stations are located between 87.5MHz and 108MHz
- Set demodulation to „WFM (stereo)“
- For a noisy signal: update Squelch setting
- Adjust volume by setting the audio gain



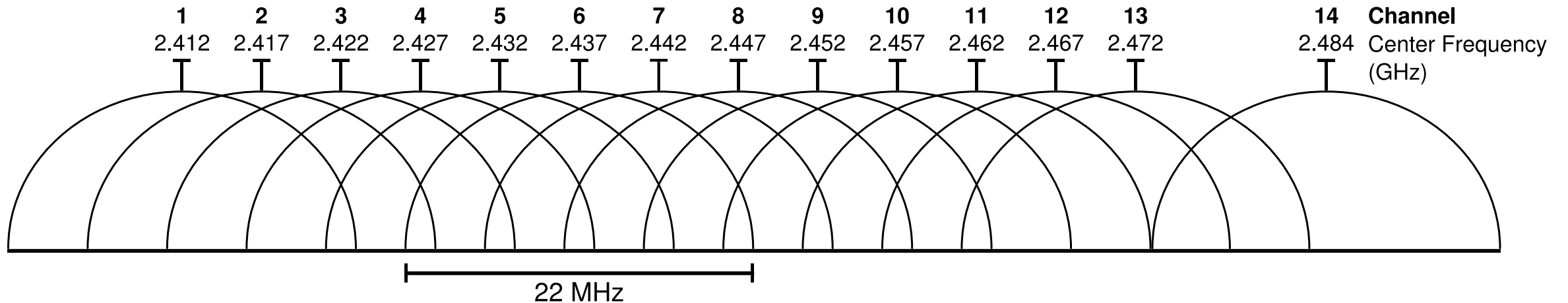
# GSM

- GSM downlink is located between 925MHz and 960MHz (Germany)
- Set maximum sampling rate + bandwidth to find ARFCNs in use

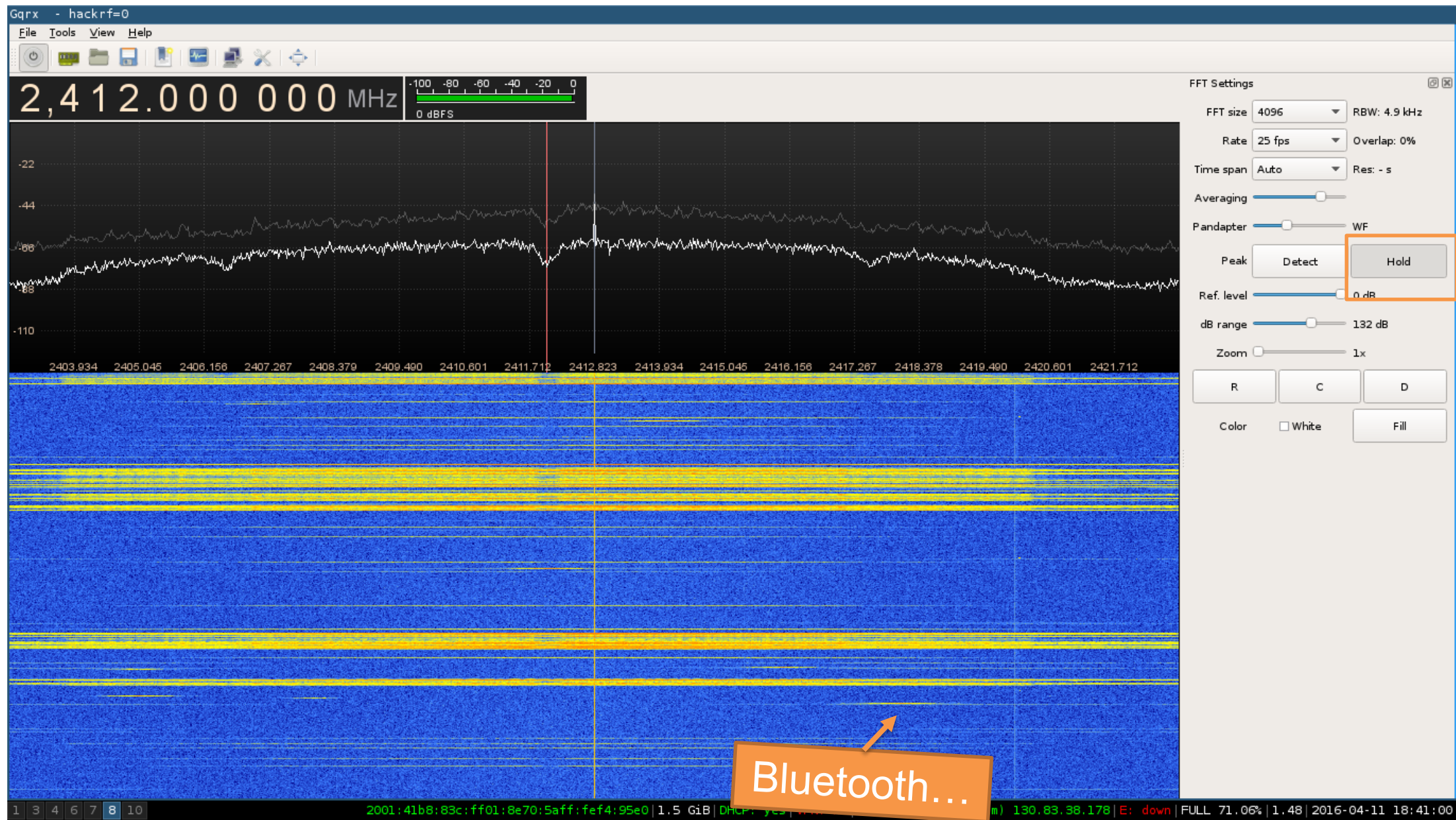


# WiFi

- A bandwidth of 20MHz is required – does not work with DVB-T sticks!
- Also, DVB-T sticks only go up to 2.2GHz...
- We need to select a channel center frequency for WiFi sniffing:



[https://en.wikipedia.org/wiki/IEEE\\_802.11#/media/File:2.4\\_GHz\\_Wi-Fi\\_channels\\_\(802.11b,g\\_WLAN\).svg](https://en.wikipedia.org/wiki/IEEE_802.11#/media/File:2.4_GHz_Wi-Fi_channels_(802.11b,g_WLAN).svg)



---

# Demo gnuradio

---

# Features

- Open source **signal processing**
- Many interesting projects available, e.g. GSM, Bluetooth, WiFi, TETRA
- Supports HackRF, rad1o, USRP, ...
  
- Demo projects:
  - `gr-ieee80211`
  - `gr-gsm`

# gr-ieee802-11

The image displays the GNU Radio Companion (GRC) interface for a WiFi receiver project. The main window shows a flow graph with the following components:

- UHD: USRP Source**: Samp Rate (Sps): 20M, Ch0: Center Freq (Hz): 2.4120, Ch0: Gain Value: 20.
- Complex to Mag^2**: A block that converts complex samples to their magnitude squared.
- Delay**: Delay: 16.
- Complex Conjugate**: A block that conjugates the complex samples.
- WX GUI Chooser** and **WX GUI Notebook**: Control panels for various parameters like LO Offset, channel estimates, and notebook settings.
- Variable**: window\_size (Value: 48) and sync\_length (Value: 320).

The **Scope Plot** window shows a constellation plot with the following settings:

- Sample Rate: 5 MHz, 10 MHz, 20 MHz (selected).
- constellation | autocorrelation
- Scope Plot: XY view, Persistence off, Analog Alpha: 0.0994.
- Channel Options: Ch1, Ch2, Trig, XY (selected).
- Channel X: Ch 1, Channel Y: Ch 2, Marker: Dot Med.
- LO Offset: 0 MHz.

The **ofdm.pcap** packet capture window shows the following data:

Source	Destination	Protocol	Info
IntelCor_f4:95:e0	CiscoInc_43:94:c0	802.11	QoS Null function (No data), SN=0, FH=0, Flags=.....T
IntelCor_f4:95:e0	CiscoInc_43:94:c0	802.11	QoS Null function (No data), SN=0, FH=0, Flags=.....T
IntelCor_f4:95:e0	CiscoInc_43:94:c0	802.11	QoS Null function (No data), SN=0, FH=0, Flags=.....T
IntelCor_f4:95:e0	CiscoInc_43:84:c0 (00:21:d8:83:ff01:8e70:5aff:fe)	802.11	Acknowledgement, Flags=.....T
IntelCor_f4:95:e0	CiscoInc_43:94:c0	802.11	QoS Null function (No data), SN=0, FH=0, Flags=.....T
IntelCor_f4:95:e0	CiscoInc_43:94:c0 (00:21:d8:83:ff01:8e70:5aff:fe)	802.11	Acknowledgement, Flags=.....T
IntelCor_f4:95:e0	CiscoInc_43:94:c0 (00:21:d8:83:ff01:8e70:5aff:fe)	802.11	Acknowledgement, Flags=.....T

Frame 4: 27 bytes on wire (216 bits), 27 bytes captured (216 bits)

- ▶ Radiotap Header v0, Length 17
- ▶ 802.11 radio information
- ▶ IEEE 802.11 Acknowledgement, Flags=.....T
  - Type/Subtype: Acknowledgement (0x001d)
  - Frame Control Field: 0xd400
  - .000 0000 0000 0000 = Duration: 0 microseconds
  - Receiver address: CiscoInc\_43:94:c0 (00:21:d8:83:ff01:8e70:5aff:fe)



# gr-gsm

ARFCN from gqrx:

**947.8MHz**

$$= 890\text{MHz} + 0.2\text{MHz} * 64 + 45\text{MHz}$$

```
grgsm_capture -a 64 -c output.gsm
```

```
wireshark -k -f udp -Y gsmtap -i  
lo
```

```
grgsm_decode -a 64 -c output.gsm
```

The image shows a Wireshark capture of a GSM CCCH packet. The packet list pane shows a single packet from 127.0.0.1 to 127.0.0.1, identified as GSMTAP (CCCH) (RR) System Information Type 4. The packet details pane shows the following structure:

- Frame 1: 81 bytes on wire (648 bits), 81 bytes captured (648 bits) on interface 0
- Ethernet II, Src: 00:00:00\_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00\_00:00:00 (00:00:00:00:00:00)
- Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
- User Datagram Protocol, Src Port: 45571 (45571), Dst Port: 4729 (4729)
- GSM TAP Header, ARFCN: 0 (Downlink), TS: 0, Channel: BCCH (0)
- GSM CCCH - System Information Type 4
  - L2 Pseudo Length
    - ... 0110 = Protocol discriminator: Radio Resources Management messages (0x06)
    - Message Type: System Information Type 4
  - Location Area Identification (LAI)
    - Location Area Identification (LAI) - 262/02/664
      - Mobile Country Code (MCC): Germany (262)
      - Mobile Network Code (MNC): Vodafone D2 GmbH (02)
      - Location Area Code (LAC): 0x0298 (664)
    - Cell Selection Parameters
    - RACH Control Parameters
    - SI 4 Rest Octets

The packet bytes pane shows the following hex and ASCII data:

```
0000 00 00 00 00 00 00 00 00 00 00 08 00 00 45 00 .....E.  
0010 00 43 71 ae 40 00 40 11 ca f9 7f 00 00 01 7f 00 .Cq.@.@.....  
0020 00 01 b2 03 12 79 00 2f fe 42 02 04 01 00 00 00 .....y/.B.....  
0030 a2 00 00 1f 82 93 01 00 00 00 31 06 1c 52 f2 20 .....1..b.  
0040 02 98 65 04 b9 00 00 80 00 43 2b 2b 2b 2b 2b 2b ..e.....C+++++  
0050 2b +
```

---

# Where to start?

# Getting Started

- Get a **rtl-sdr** compatible DVB-T stick
  - Connecting software defined radios to **virtual machines can cause data loss!**
  - Some software might also run under Windows, but even harder to install...
  - Use a **Live CD**, e.g. Kali Linux 😊
- 
- Demo today used:
    - Debian testing packets with `gnuradio 3.7.9.1-2+b1`
    - `gr-ieee802-11` and `gr-gsm` built from github sources on April 11 2016



---

# Q&A