Free and Open Source Software in SDR

Harald Welte <hwelte@sysmocom.de>

osmocom.org
sysmocom GmbH

June 29, SDR'12 - WInnForum Europe

Outline

About the speaker

- Linux Kernel / bootloader / driver / firmware developmer since 1999
- IT security expert, focus on network protocol security
- Former core developer of Linux packet filter netfilter/iptables
- Board-level Electrical Engineering
- Always looking for interesting protocols (RFID, DECT, GSM)
- OpenPCD, Openmoko, deDECTed.org, OpenBSC, OsmocomBB, OsmoSGSN

About sysmocom GmbH

systems for mobile communications

- small company, started by two Osmocom developers in Berlin
- provides commercial R&d and support for professional users of Osmocom software
- develops its own products like sysmoBTS (inexpensive, small-form-factor, OpenBSC compatible BTS)
- runs a small webshop for Osmocom related hardware like OsmocomBB compatible phones, SIMtrace, etc.

- Free and Open Source Software (FOSS) is everywhere
- Particularly Servers and all areas of Embedded
- FOSS has fundamentally changed the software industry
- Systems architecture of products becomes more complex
- Nobody can afford to build complex products from scratch
- Everyone builds products on existing FOSS components, particularly the Linux kernel and other OS-level components

Linux and Free Software (FOSS) everywhere

- FOSS is not a technology
- FOSS is not a product
- FOSS is not a company
- FOSS is a development methodology and culture
- Only companies with sufficient FOSS experience understand the value of how to interact with the wider FOSS communities

- FOSS enables participation
- you don't have to work for a specific company in order to do OS development
- nobody has to have any formal relationship with their collaborators, suppliers.
- any *nobody* can contribute, even so-called amateurs, hobbyists, students
- it doesn't matter how deep your pockets are
- meritocracy (the better your merits, the more you have a say in the development process)

FOSS: Democracy / Equal Access

- The means of productions (Computers, OS, Compilers) are abundant and inexpensive (for the first world)
- Anyone can create and produce software, all you need is your brain
- No membership required in exclusive forums, industry clubs, consortia

Traditional Radio Engineering

- Traditional radio development required electrical engineering in hardware. You have to
  - know analog / RF electronics
  - spin board revisions / prototypes
  - actually physically build something
- Aside from the skills, there is a significant non-HR cost involved for actually doing this development

SDR and FOSS

- SDR transforms radio engineering into the software domain
- In Software, all you need to do R&D is a bit of general-purpose hardware and your brains
- With inexpensive general-purpose SDR hardware, the same conditions apply to development of radio software!
- Participatory, collaborative, community driven R&D

- When you (the audience) thinks of SDR, it's probably mostly bleeding-edge high-end and high-cost
- At the same time, if you don't have the same high-end requirements, SDR receiver hardware is available cheap
- commoditization effect

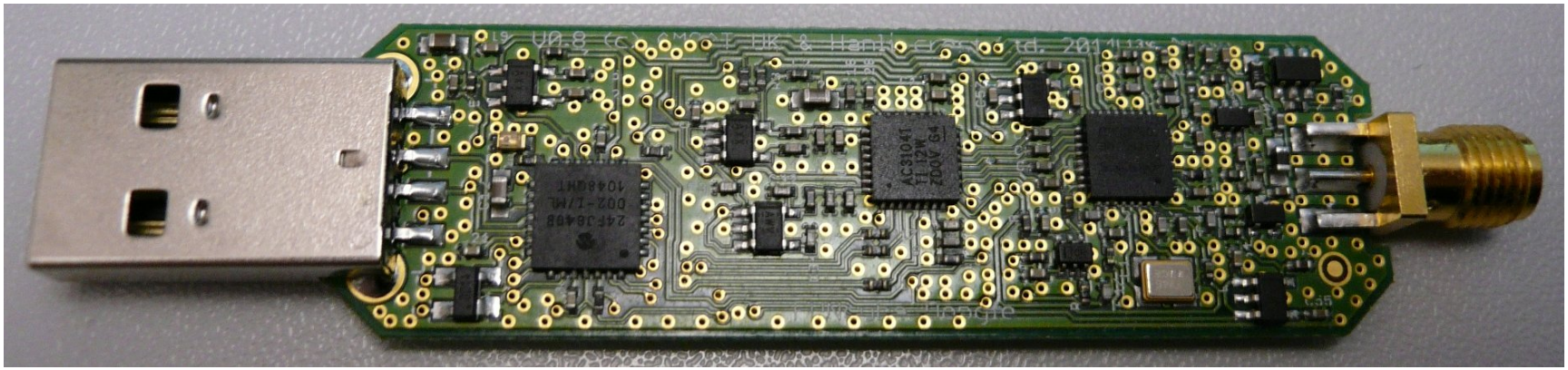
The USRP family

- probably the most-used SDR hardware in the FOSS world
- still the primarily radio used with gnuradio today
- at the low end of the 'professional sdr' price segment
- still, typical configuration costs > 1000 USD
- not everyone is able to spend that (students, hobbyists, especially outside first world countries)

Fun Cube Dongle Pro (2010)

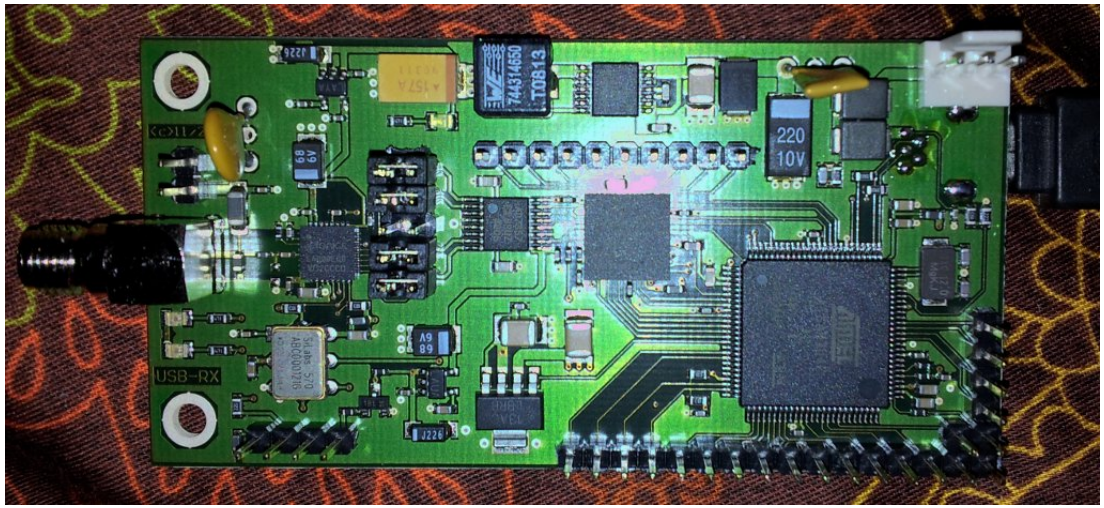
- 64 MHz to 1700 Mhz USB SDR receiver (193 USD)
- limited to 96 kHz I/Q baseband sampling
- great for amateur radio and TETRA, but most other communications systems (like GSM introduced in 1992) use wider band-widths
- great progress in terms of size and cost, but much more limited than USRP
- Hardware design and firmware sadly are proprietary

Fun Cube Dongle Pro (2010)



OsmoSDR (2012)

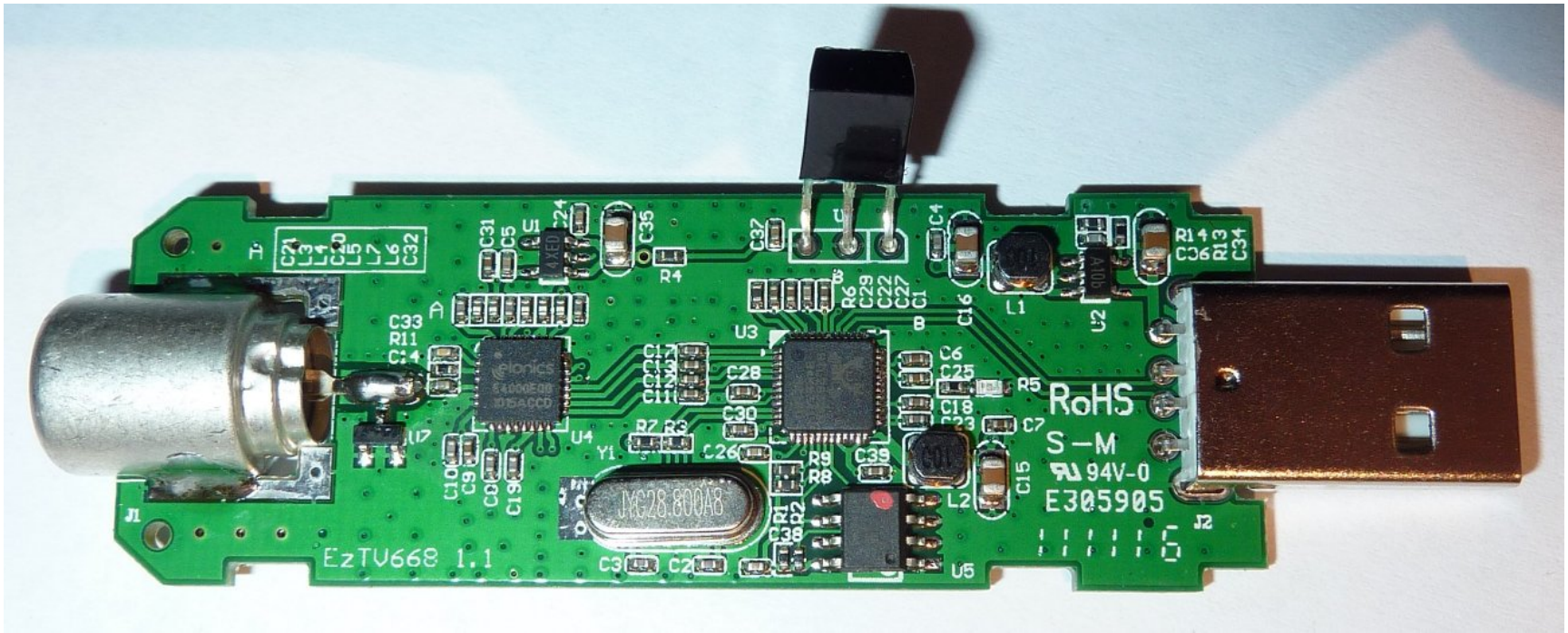
- small, low-power / low-cost USB SDR hardware (225 USD)
- higher bandwidth than FunCubeDonglePro (1.2 Ms/s / 14bit)
- much lower cost than USRP, but more expensive than FCDP
- Open Hardware (schematics), software (FPGA, firmware)
- Undergoing another re-spin for 4.2 Ms/s @ 14bit



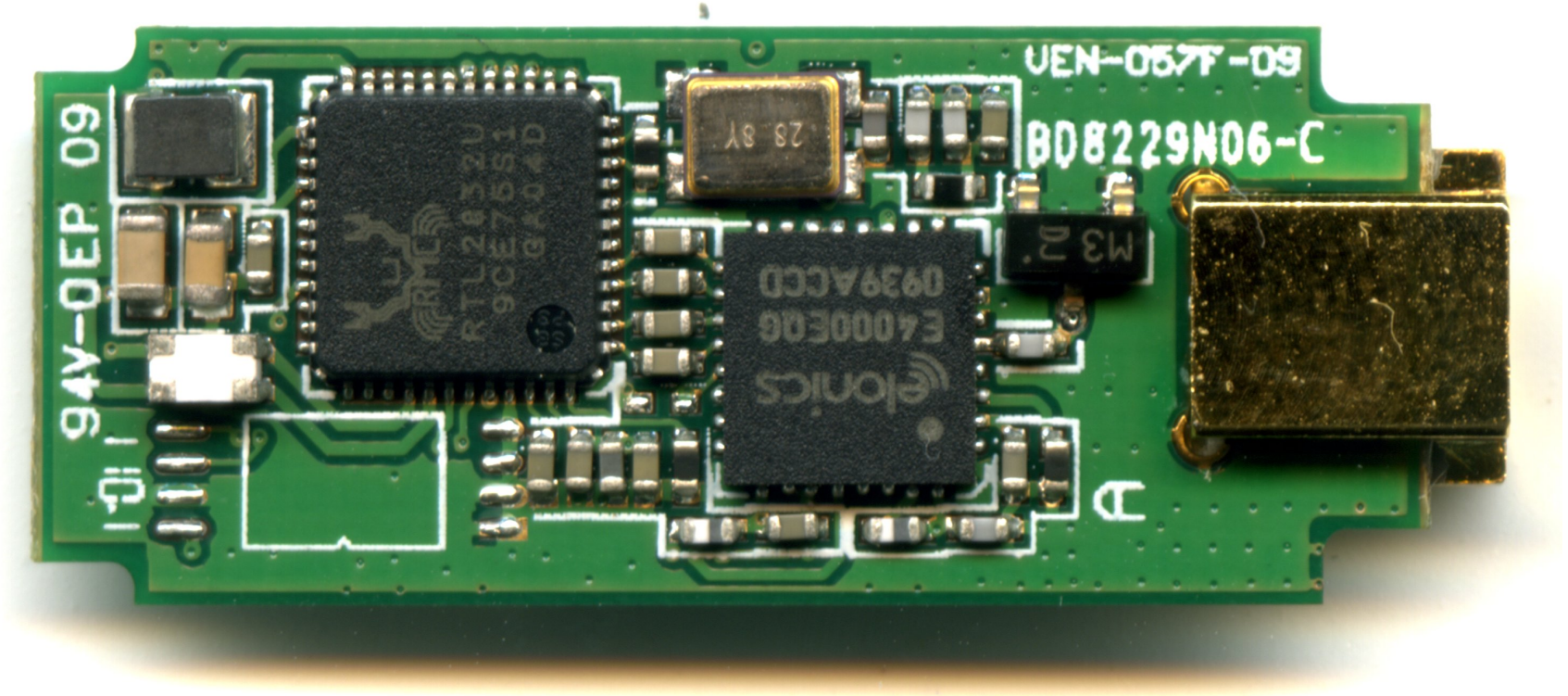
Realtek RTL2832U based DVB-T receivers

- Realtek RTL2832U based DVB-T receivers are cheaply available on the market (USD 20)
- RTL2832U implements ADC, DVB-T demodulator and high-speed USB device
- Normal mode of operation includes full DVB-T receiver inside RTL2832U hardware and only sends MPEG2-TS via USB
- Reverse engineering the USB protocol and replaying certain commands from custom libusb based code was able to trigger the raw sample transmission to the host PC

RTL2832U based devices: EzTV 668



RTL2832U based devices: Hama nano1



Gnuradio

- Philosophy: Implement SDR not as hand-crafted special-case hand-optimized assembly code in some obscure DSP, but on a general purpose PC
  - with modern x86 systems at multi-GHz clock speeds and with many cores this becomes feasible
  - of course way too expensive for a mass-produced product, but very suitable for research, teaching and rapid prototyping
- Implement various signal processing elements in C++
  - assembly optimized libraries for low-level operations
  - provide python bindings for all blocks
- Python script to define interaction, relation, signal routing between blocks

gnuradio based waveform implementations

- Of course plenty of gr-based implementations for the various analog modulation schemes
- Check out CGRAN (comprehensive gnuradio archive network): Includes 802.11, Zigbee, RDS, DECT, AIS, UHF RFID, ADS-B
- Many other projects out of academia and community, such as OpenLTE (early stage of downlink Rx/Tx)

Osmocom / osmocom.org

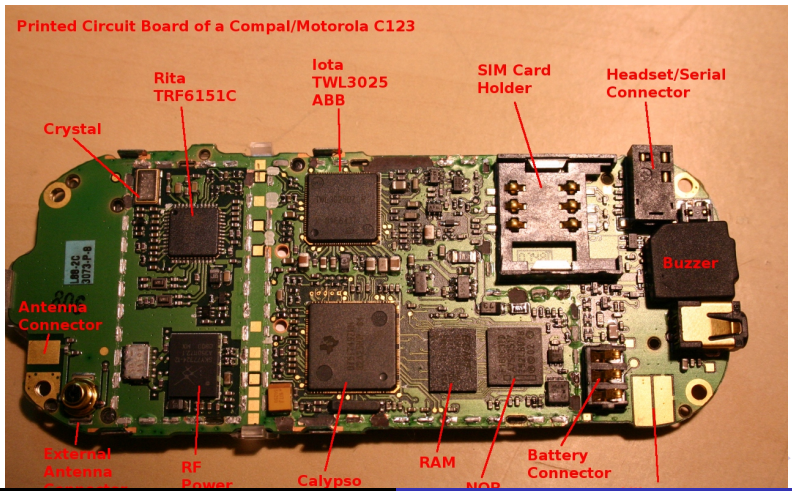
- Osmocom == Open Source Mobile Communications
- Classic collaborative, community-driven FOSS project
- Gathers creative people who want to explore this industry-dominated closed mobile communications world
- http://osmocom.org/
  - non-sdr sub-projects like L2/L3 protocol stacks
  - sdr sub-projects for mostly Rx side

OpenBSC

- first Osmocom project
- Implements GSM A-bis interface towards BTS
- Supports Siemens, ip.access, Ericsson and Nokia BTS
- can implement only BSC function (osmo-bsc) or a fully autonomous self-contained GSM network (osmo-nitb) that requires no external MSC/VLR/AUC/HLR/EIR
- deployed in > 200 installations world-wide, commercial and research

OsmocomBB

- Full baseband processor firmware implementation of a mobile phone (MS)
- We re-use existing phone hardware and re-wrote the L1, L2, L3 and higher level logic
- Higher layers reuse code from OpenBSC wherever possible
- Used in a number of universities and other research contexts (including Ericsson Research)



OsmocomTETRA

- SDR implementation of a TETRA radio-modem (PHY/MAC)
- Rx is fully implemented, Tx only partial
- Can be used for air interface interception
- Accompanied by wireshark dissectors for the TETRA protocol stack

OsmocomGMR

- ETSI GMR (Geo Mobile Radio) is "GSM for satellites"
- GMR-1 used by Thuraya satellite network
- OsmocomGMR implements SDR based radiomodem + PHY/MAC (Rx)
- Partial wireshark dissectors for the protocol stack
- Reverse engineered implementation of GMR-A5 crypto
- Speech codec is proprietary, still needs reverse engineering

OsmocomOP25

- APCO25 is Professional PMR system used in the US
- Can be compared to TETRA in Europe
- OsmocomOP25 is again SDR receiver + protocol analyzer

The OpenBTS Um - SIP bridge

- OpenBTS is a SDR implementation of GSM Um radio interface
- directly bridges to SIP/RTP, no A-bis/BSC/A/MSC
- suitable for research on air interface, but very different from traditional GSM networks

airprobe.org

- SDR implementation of Um sniffer
- suitable for receiving GSM Um downlink and uplink
- predates all of the other projects
- more or less abandoned at this point

Thanks

Thanks for your attention. I hope we have time for Q&A.