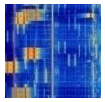
Amateurfunktagung 2014, München

Amateurfunk-Anwendungen mit dem Raspberry Pi

8./9. März 2014 | Stefan Hüpper, DH5FFL

Agenda

- **Überblick: Was ist der Raspberry Pi?**
  -  **Blick auf die Hardware**
  -  **Zubehör**
  -  **Grundinstallation**
  -  **Blick auf die Platine**
- **Anwendungsbeispiele**
  -  **1. Rx-iGate für APRS mit TNC2C**
  -  **2. Echolink-Gateway**
  -  **3. Server für ein LAN-SDR**
  -  **4. Mit dem Clockgenerator via GPIO-Port auf Sendung**
- **Fazit**

Liebe auf den ersten Blick!

Deutscher Amateur-Radio-Club e.V.
Bundesverband für Amateurfunk in Deutschland

… mit dieser Meldung fing es an:

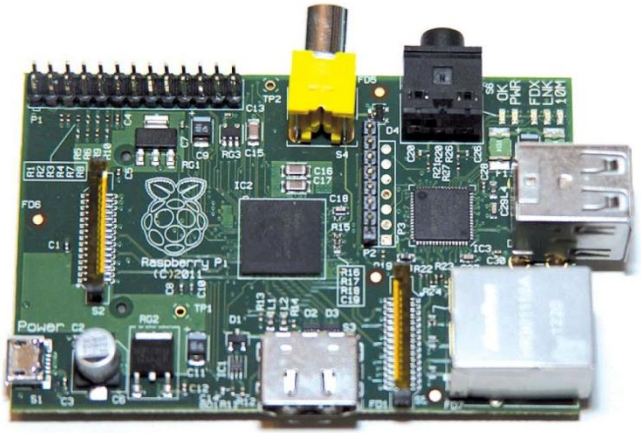
„<u>25-Dollar-Minicomputer geht in die Vorserie</u>
30 Jahre nach dem sehr erfolgreichen BBC Micro soll bald ein neuer britischer Mini-Computer wieder auf sich aufmerksam machen, der 25-Dollar-Minicomputer der gemeinnützigen Raspberry Pi Foundation, der nur 1 Watt verbrauchen soll. Gedacht ist der preiswerte Kleinrechner vor allem zur Ausbildung von Kindern (…)

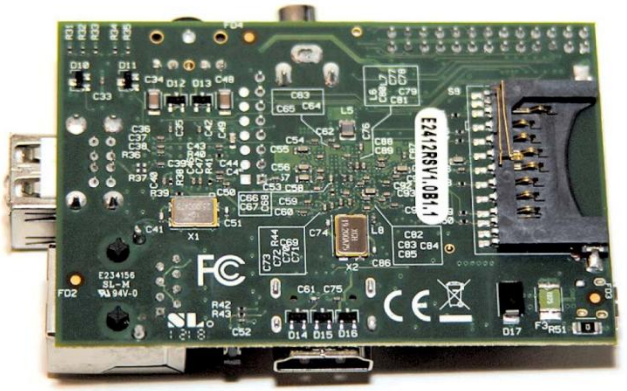
(Quelle: Heise.de, 31. Juli 2011)

In Kreditkartengröße - der Raspberry Pi

Deutscher Amateur-Radio-Club e.V.
Bundesverband für Amateurfunk in Deutschland

- **Platinengröße: 85,6 mm × 53,98 mm × 17 mm**
- **Prozessor (CPU): ARM1176JZF-S (700 MHz, übertaktbar)**
- **Speicher (RAM): 512 MB, nicht erweiterbar**
- **Grafikprozessor (GPU): Broadcom VideoCore IV**
- **System-on-Chip (SoC): Broadcom BCM2835**
- **Peripherie: 1 × Ethernet 10/100 MBit, 2 × USB, Audio-Out, HDMI, Video-out, GPIO-Port (u.a. mit UART)**
- **Strom: über Micro-USB, ca. 300…700 mA (mit USB-Geräten ca. 3,2 W vor Schaltnetzteil gemessen)**
- **Betriebssysteme: Linux (Debian, Fedora, Arch Linux, FreeBSD, RiscOS)**

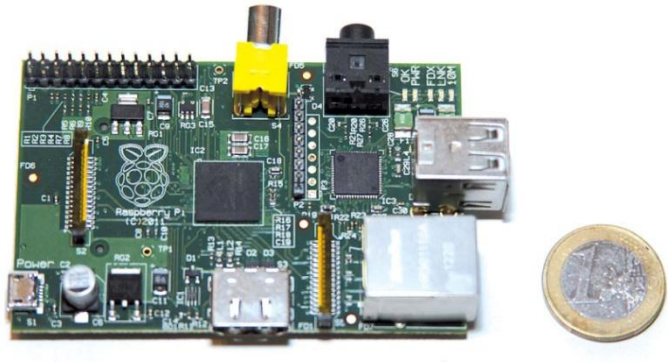




Bezug, Typen, Erstausstattung

- **Bezug über RS, Element14 (Farnell), Allied Electronics. Alternativ auch: Amazon, DARC Verlag**
- **Es gibt zwei Versionen:**

Modell A	und	Modell B
- 256 MB RAM		- 512 MB RAM
- 1x USB 2.0		- 2x USB 2.0
- kein Ethernet		- Ethernet 10/100 Mbit
- geringfügig stromsparsamer		
- Preis: 24,25 €		- Preis: 32,88 €



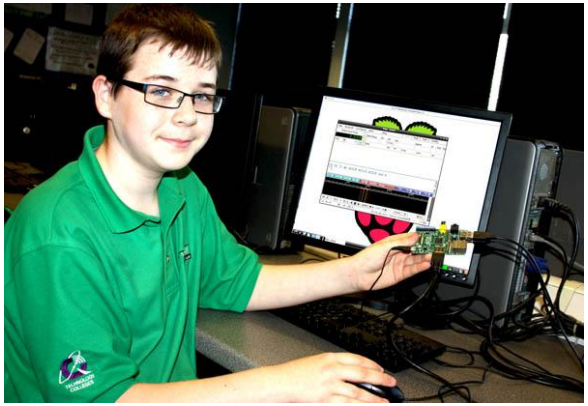
- **Erstausstattung u.U. nicht kostenlos: Kauft man alles neu kommen ca. 45 € hinzu:**

Ziele und Marktstart des Raspberry Pi‘s

Deutscher Amateur-Radio-Club e.V.
Bundesverband für Amateurfunk in Deutschland

- **Entstanden innerhalb der „Raspberry Pi Foundation“, einer Wohltätigkeitsorganisation in Großbritannien**

- **Zielgruppe zunächst Kinder und Studenten. Vor allem aber Erwachsene zeigten Interesse**



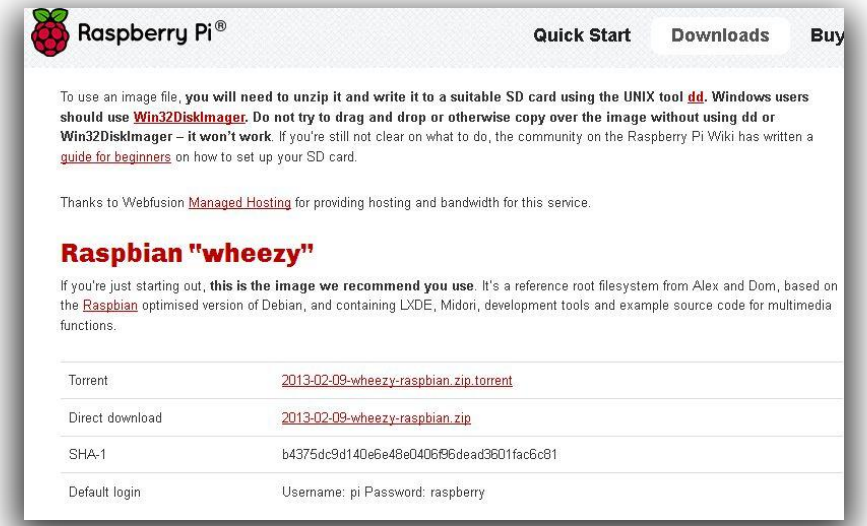
(Foto: © Newton News, UK)

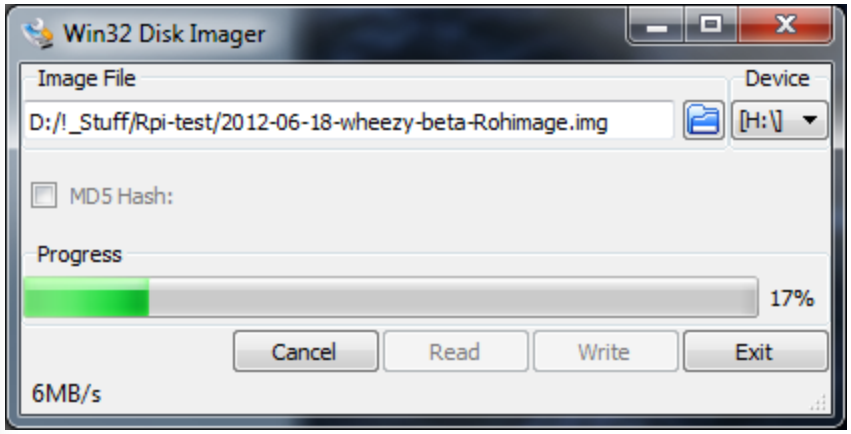
- **Vorbestellungen waren ab April 2012 möglich**

- **350 000 Vorbestellungen resultierten Anfangs in einer Wartezeit von ca. 12 Wochen. Mit Stand Juli 2012 wurden damals erst 10 000 Stück ausgeliefert. Mit Stand Oktober 2013: 1,75 Mio Exemplare**

- **Der „C64“ von heute (geschätzte Verkaufszahl damals: 12,5 bis 30 Millionen Exemplare)**

Softwarebezug & Erstinstallation (1/3)





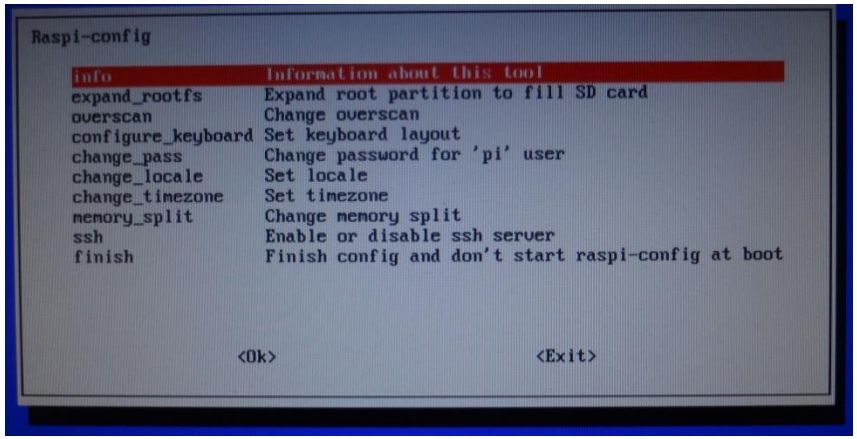
1. Von Webseite www.raspberrypi.org Betriebssystem-Image herunterladen: Raspbian „wheezy“.

Außerdem gibt es: Arch Linux ARM, RISC OS und ein XBMC-System (http://www.raspbmc.com)

2. Z.B. mit dem „Win32 Disk Imager“ das Iso auf SD-Karte übrtragen. (Bezug z.B. via www.chip.de)

Softwarebezug & Erstinstallation (2/3)

3. Netzteil via Mini-USB-Buchse anschließen. Hier werden sparsame 2,3 W Leistungsaufnahme gemessen. Mit Tastatur/Maus bzw. USB-Soundkarte etwa 3,5 W



4. Beim Erststart fährt „raspi-config“ hoch. Einstellung von:

- Zeitzone
- Keyboardlayout
- Expand RootFS
- Overclocking
- Memory Split für die GPU
- SSH-Server ein/ausschalten
- Bootverhalten (X gleich starten)

Softwarebezug & Erstinstallation (3/3)

5. Willkommen im Linux-System!
Es empfiehlt sich ein System-Update mit folgenden zwei Schritten:

```
sudo apt-get update
sudo apt-get upgrade
```

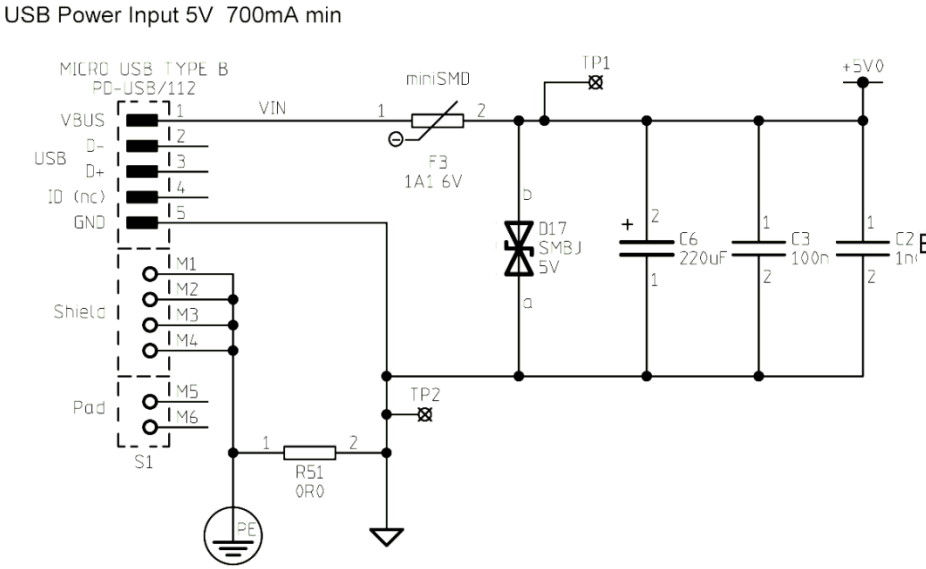
…und dem „Rpi-Updater“ von Hexxeh. Damit wird die Firmware, d.h. der Kernel und die Systemdateien auf den aktuellsten Stand gebracht.

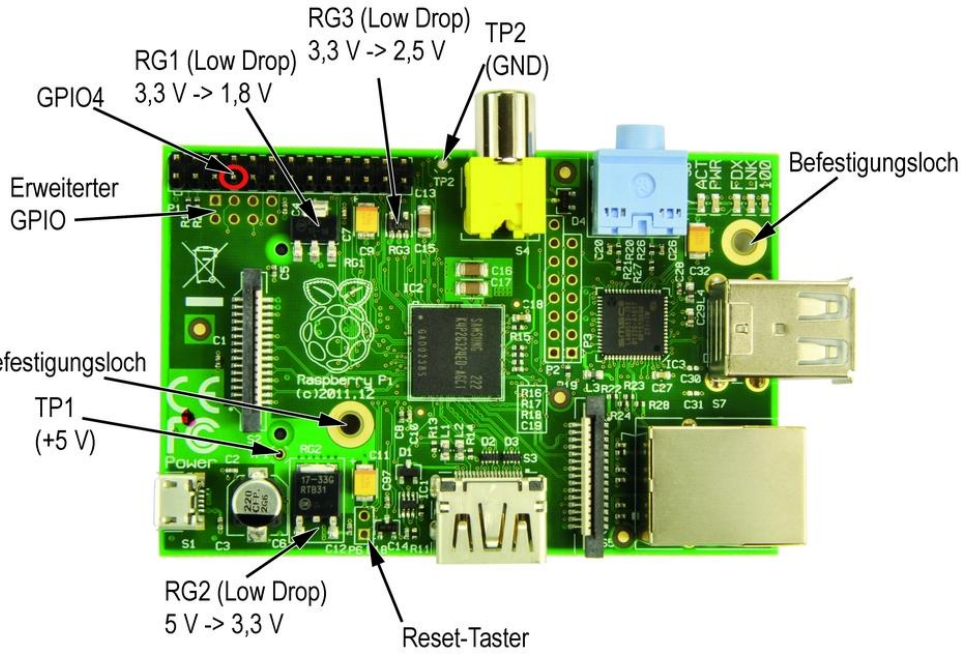
```
sudo apt-get install ca-certificates git-core
sudo wget http://goo.gl/1BOfJ -O /usr/bin/rpi-update
sudo chmod +x /usr/bin/rpi-update
sudo rpi-update
```

Nun ist das System einsatzbereit!

Blick auf die Platine (1/2)

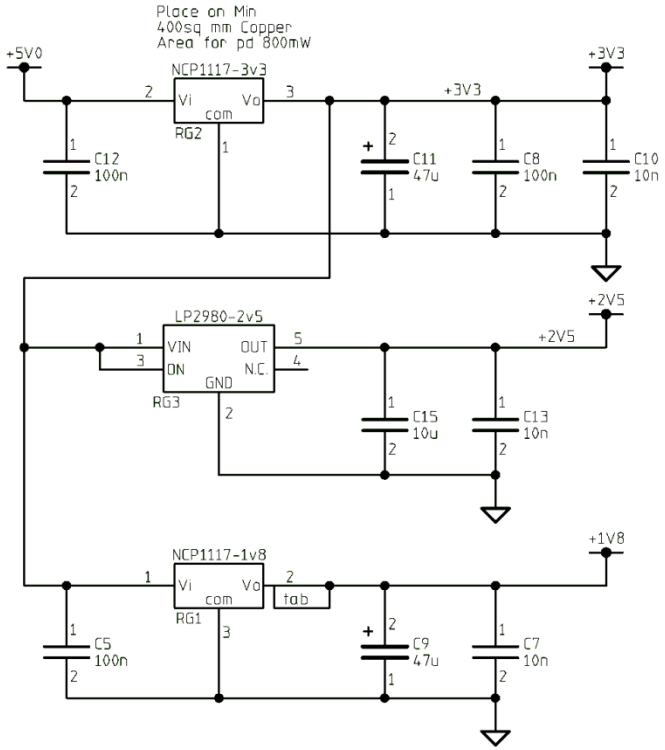
- **Problem: Mikro-USB-Buchse zur Versorgung „sehr wackelig“.**
- **Lösung: Messpunkte TP1/2 nutzen! Direkteinspeisung der Versorgung darüber möglich; alternativ über GPIO**

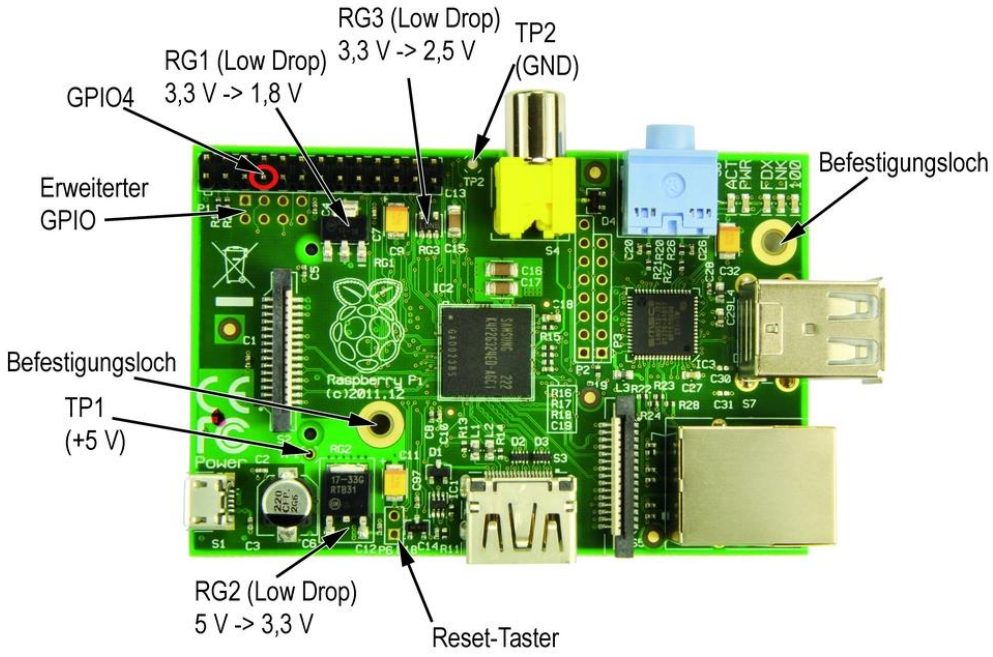




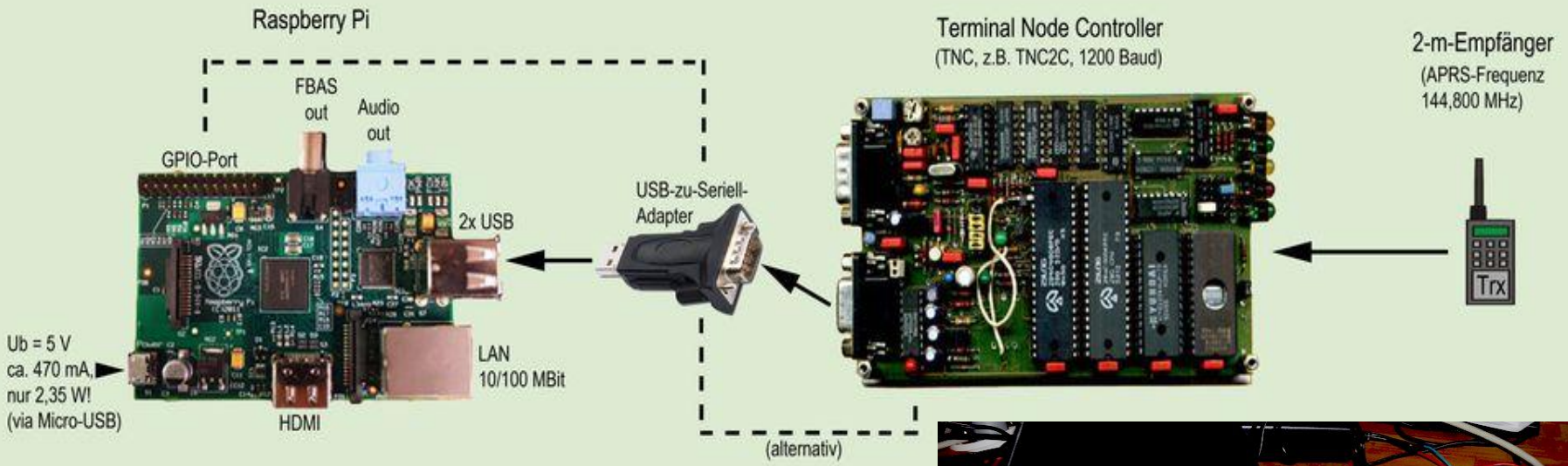
Blick auf die Platine (2/2)

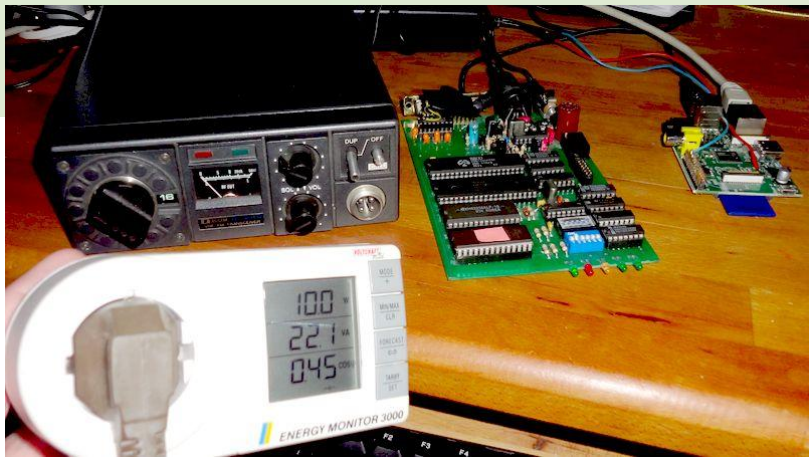
- **Stromsparen? RG2 (& RG1) durch Schaltwandler ersetzen**
- **Raspberry Pi benötigt 5 V nur für HDMI und USB. Betrieb wäre an 3,3 V möglich.**
- **Batteriebetrieb: Idee von Dave Akerman, M6RPI. 4x AA-Zellen plus Low-Drop-Regler MCP1825S -> 3,3 V out**
  Zeit = Batteriekapazität / Entladestrom
  Beispiel: 3000 mAh/200 mA = 15 Stunden





APRS-Rx-iGate mit TNC2c (1/8)





Mobil-APRS macht nur Spaß, wenn es genug Empfangsstellen auf der 144,800 MHz gibt! Diese sollten wegen des Dauerbetriebs stromsparsam sein. Hier punktet z.B. der Raspberry Pi.

(Laborversuch)

APRS-Rx-iGate mit TNC2c (2/8)

Deutscher Amateur-Radio-Club e.V.
Bundesverband für Amateurfunk in Deutschland

Zunächst machen wir die Installation grundlegend AX.25-tauglich. Als Serveranwendung kommt „APRX“ von Matti Aarnio, OH2MQK, zum Einsatz. Zum Testen der seriellen Verbindung laden wir zusätzlich das Terminalprogramm Minicom.

```
sudo apt-get install libax25 libax25-dev ax25-apps ax25-tools minicom
sudo wget http://ham.zmailer.org/oh2mqk/aprx/aprx-2.05.svn485.tar.gz
```

Auspacken & Compilieren der Sources mit

```
sudo tar xvf aprx-2.05.svn485.tar.gz
cd aprx-2.05.svn485
sudo ./configure && make && make install
```

Testen der seriellen Verbindung mit dem Terminalprogramm Minicom. Hier sollte die Einschaltmeldung des TNCs auf dem Bildschirm erscheinen (Hostmode!, /dev/ttyUSB0, z.B. 19200 Bd, 8N1). TNC-Kommunikation läuft später jedoch mit KISS-Mode.

Nächster Schritt: APRX konfigurieren. Datei: /etc/ax25/axports

```
# /etc/ax25/axports
# The format of this file is:
# name callsign speed paclen window description
Packet DH5FFL-10 19200 128 2 144.800MHz APRS 1200bps
```

APRS-Rx-iGate mit TNC2c (3/8)

Deutscher Amateur-Radio-Club e.V.
Bundesverband für Amateurfunk in Deutschland

Datei: /etc/aprx.conf

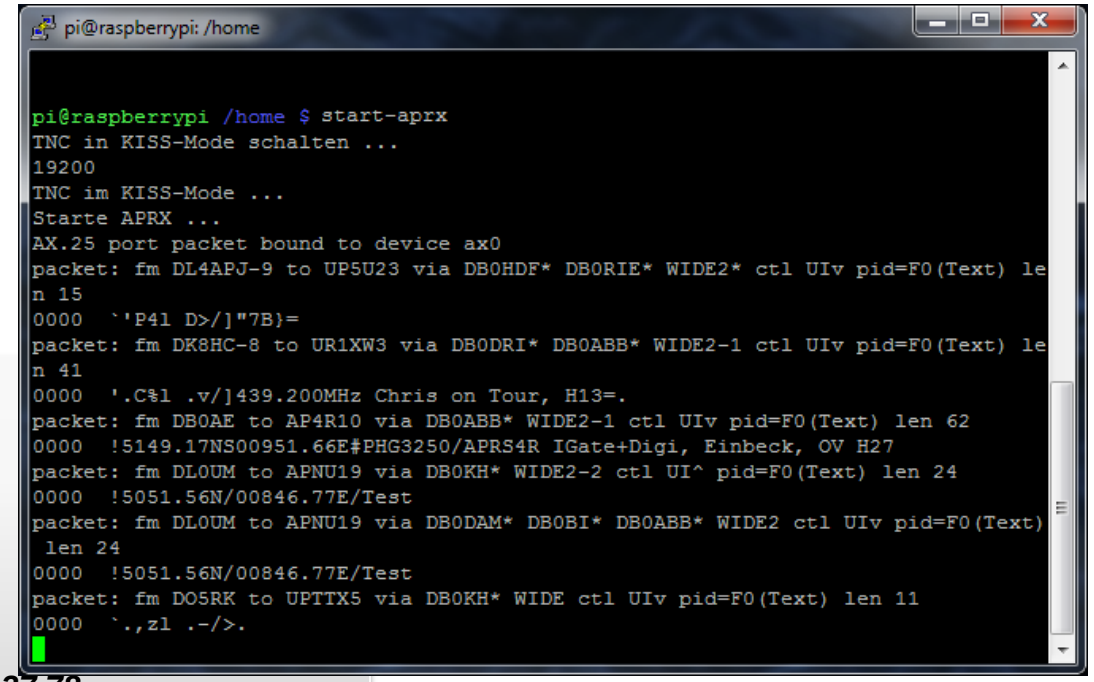
```
mycall DH5FFL-10

<aprsis>
   server euro.aprs2.net 14580
</aprsis>

<logging>
   pidfile /var/run/aprx.pid
   rflog /var/log/aprx/aprx-rf.log
   aprxlog /var/log/aprx/aprx.log
   erlangfile /var/run/aprx.state
</logging>

<interface>
   ax25-device $mycall
   tx-ok false # transmitter enable defaults to false
</interface>

<beacon>
   beaconmode aprsis
   cycle-size 5m
   beacon symbol „R&“ lat „5125.50N“ lon „00920.50E“ comment „Rx-iGate auf Raspberry Pi“
</beacon>
```

APRS-Rx-iGate mit TNC2c (4/8)

Deutscher Amateur-Radio-Club e.V.
Bundesverband für Amateurfunk in Deutschland

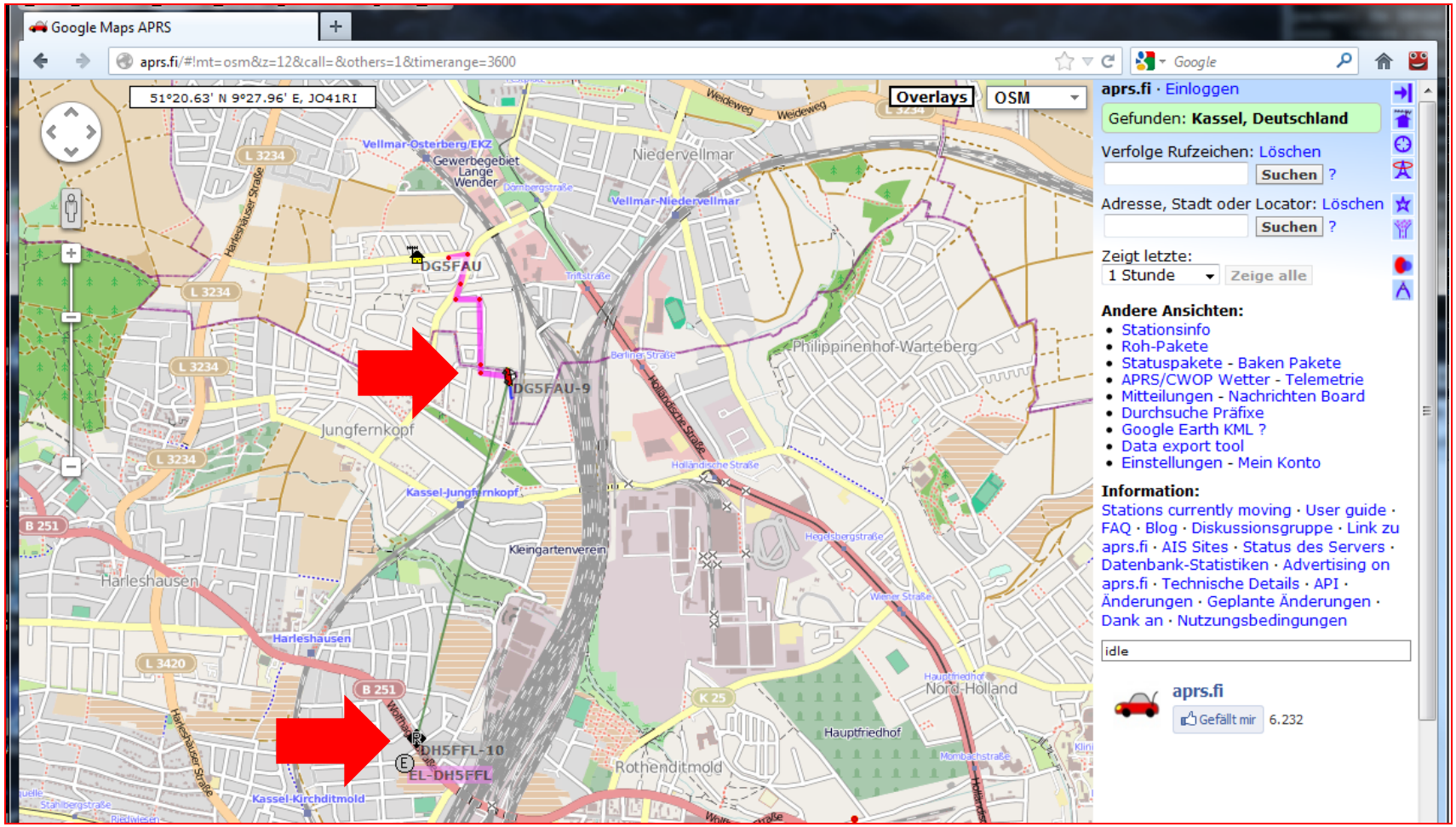
Ein (Labor-)Startskript …
Datei: /usr/local/bin/start-aprx

```
#!/bin/bash
#
echo „TNC in KISS-Mode schalten ...“
stty -F /dev/ttyUSB0 speed 19200
sleep 1
echo -ne „\r\033@K\r“ > /dev/ttyUSB0
echo „TNC im KISS-Mode ...“
sleep 2
echo „Starte APRX ...“
kissattach /dev/ttyUSB0 packet 44.130.27.78
#für 6pack anstelle TF-Firmware lautet der Kissattach-Befehl:
#kissattach -6 /dev/ttyUSB0 packet 44.130.27.78
kissparms -p packet -t 700 -s 200 -r 32 -l 100 -f n
aprx
axlisten -p packet
```



APRS-Rx-iGate mit TNC2c (5/8)

Deutscher Amateur-Radio-Club e.V.
Bundesverband für Amateurfunk in Deutschland



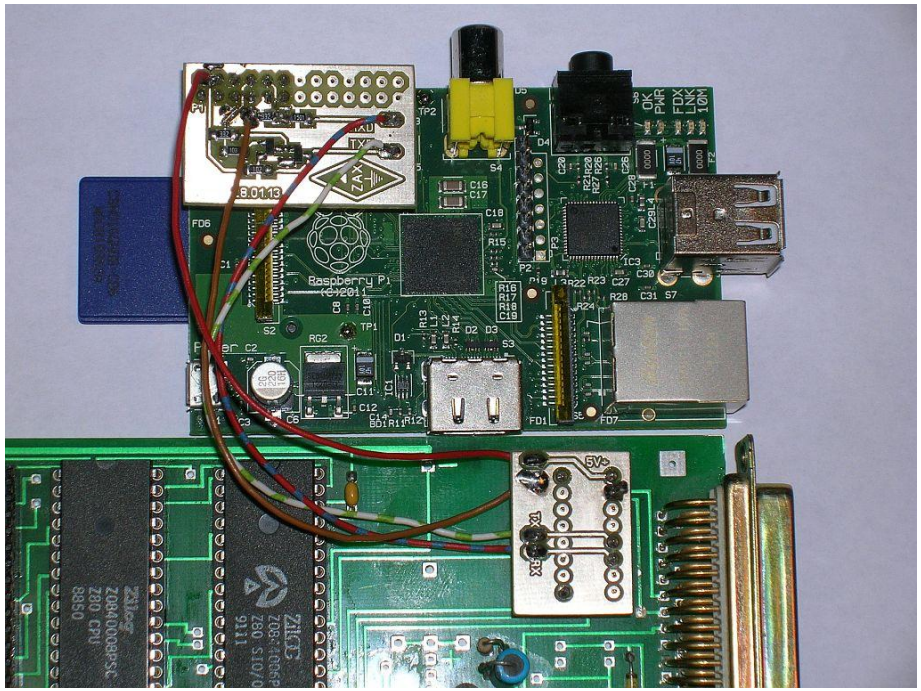
APRS-Rx-iGate mit TNC2c (6/8)

Deutscher Amateur-Radio-Club e.V.
Bundesverband für Amateurfunk in Deutschland

- **Alternativ zu einem USB-zu-Seriell-Wandler lässt sich die UART auf der GPIO-Pinleiste nutzen**

- **Idee: - „Sandwich“ aus TNC plus Raspberry Pi in einem Gehäuse und**
  - anstelle des MAX-232 auf dem TNC eine direkte Verbindung zum Raspberry Pi

- **Problem: Der Rpi arbeitet mit 3,3 V TTL und verträgt keine 5 V Signale!**

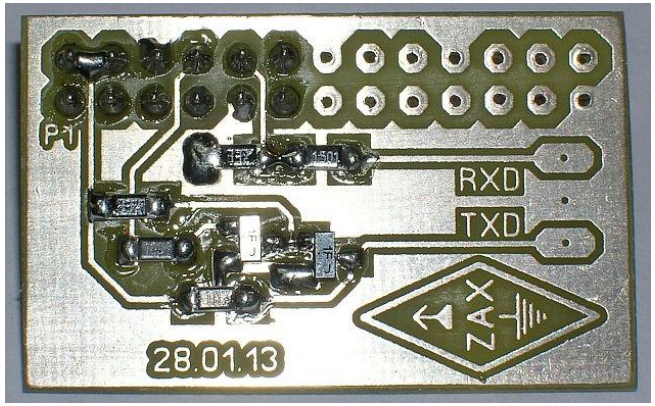


3.3V	1	2	5V
I2C0 SDA	3	4	DNC
I2C0 SCL	5	6	GROUND
GPIO4	7	8	UART TXD
DNC	9	10	UART RXD
GPIO 17	11	12	GPIO 18
GPIO 21	13	14	DNC
GPIO 22	15	16	GPIO 23
DNC	17	18	GPIO 24
SP10 MOSI	19	20	DNC
SP10 MISO	21	22	GPIO 25
SP10 SCLK	23	24	SP10 CE0 N
DNC	25	26	SP10 CE1 N

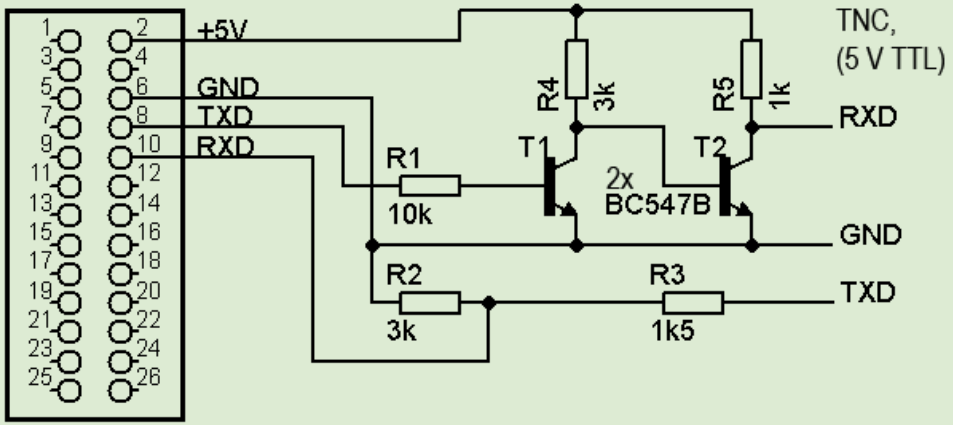
APRS-Rx-iGate mit TNC2c (7/8)

Deutscher Amateur-Radio-Club e.V.
Bundesverband für Amateurfunk in Deutschland

- **Lösung: Pegelwandler in SMD-Form!**



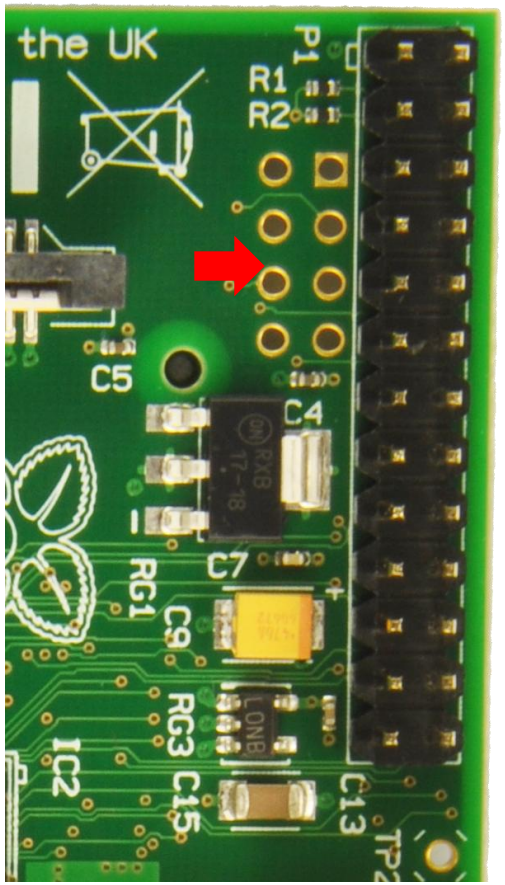
GPIO-Schnittstelle
Raspberry Pi, TTL 3,3 V



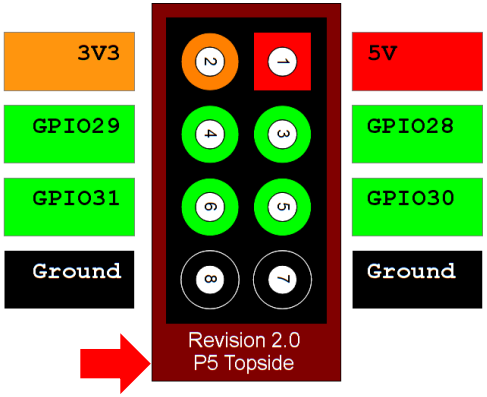
- **Auf dem Weg zur „APRX-Box“:**

… gebaut von DL1ZAX

GPIO-Pinleiste im Detail: RTS/CTS aktivieren



Model B, Version 2/512 MB



3.3V	1	2	5V
I2C0 SDA	3	4	DNC
I2C0 SCL	5	6	GROUND
GPIO4	7	8	UART TXD
DNC	9	10	UART RXD
GPIO 17	11	12	GPIO 18
GPIO 21	13	14	DNC
GPIO 22	15	16	GPIO 23
DNC	17	18	GPIO 24
SP10 MOSI	19	20	DNC
SP10 MISO	21	22	GPIO 25
SP10 SCLK	23	24	SP10 CE0 N
DNC	25	26	SP10 CE1 N

Config-Schritte vorherige Folie plus:

```
cd /home/pi
mkdir rpi-tools
cd rpi-tools/
git clone https://github.com/rewolff/
          bw_rpi_tools.git
cd bw_rpi_tools/gpio
sudo make install
```

Autostart via /etc/rc.local

```
# CTS+RTS auf P5-Header aktivieren
/usr/local/bin/gpio_setfunc 31 ALT3
/usr/local/bin/gpio_setfunc 30 ALT3
```

APRS-Rx-iGate mit TNC2c (8/8)

Deutscher Amateur-Radio-Club e.V.
Bundesverband für Amateurfunk in Deutschland

Um diese Errungenschaft zu nutzen ist folgendes nötig:

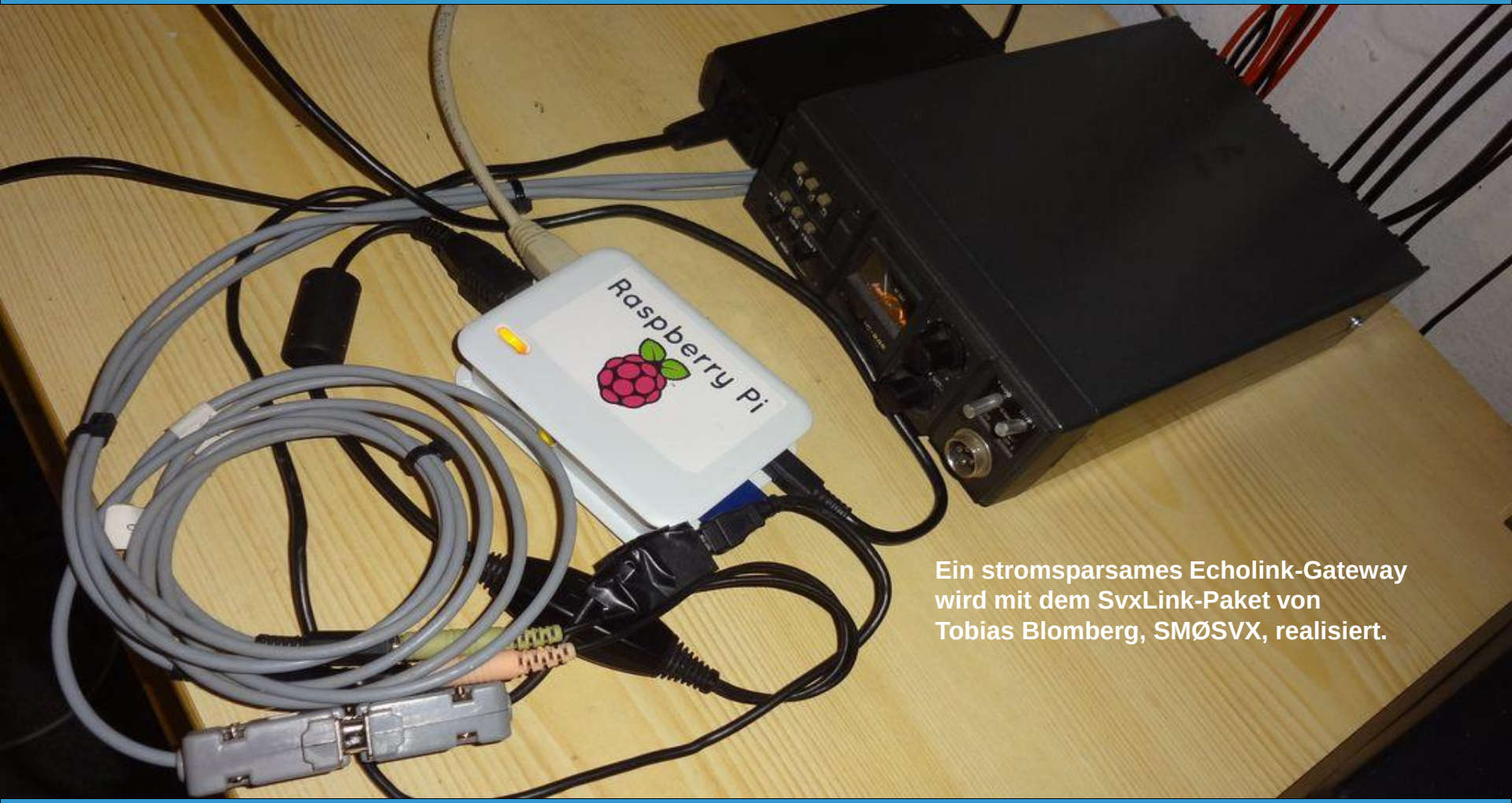
• Anstelle /dev/ttyUSB0 muss es nun überall /dev/ttyAMA0 lauten

• Bordeigene UART freimachen (zuvor für Konsolen-Login in Verwendung):
In /etc/inittab Zeile
T0:23:respawn:/sbin/getty -L ttyAMA0 115200 vt100
Durch „#“ auskommentieren

• In /boot/cmdline.txt den roten Teil entfernen:
dwc_otg.lpm_enable=0 *console=ttyAMA0,115200 kgdboc=ttyAMA0,115200* console=tty1
root=/dev/mmcblk0p2 rootfstype=ext4 elevator=deadline rootwait

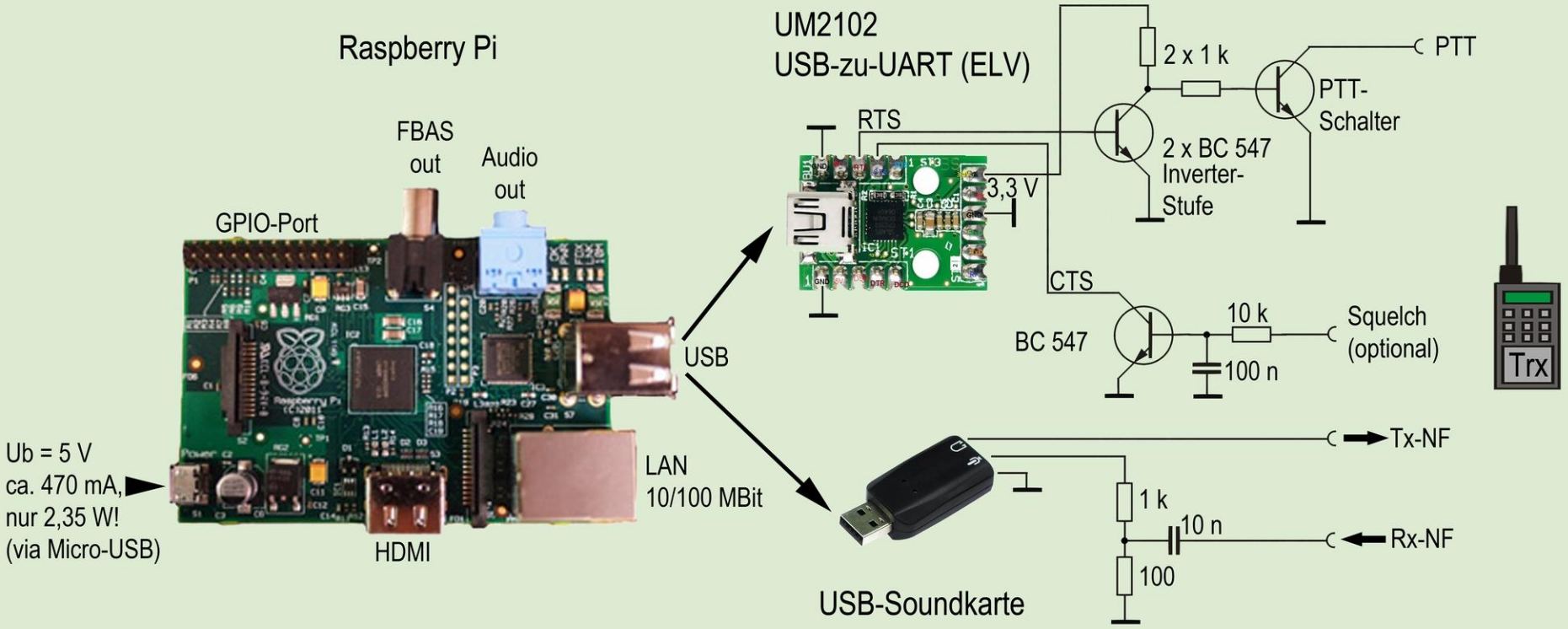
• Neustart

Svxlink-/Echolink-Gateway (1/5)



Ein stromsparsames Echolink-Gateway wird mit dem SvxLink-Paket von Tobias Blomberg, SMØSVX, realisiert.

Svxlink-/Echolink-Gateway (2/5)



… noch mit externer USB-Seriell-Platine …

Svxlink-/Echolink-Gateway (3/5)

- **Binarys gibt es nur für X86er Systeme, der Raspberry Pi hat aber eine ARM-CPU, daher muss SvxLink für diese Plattform compiliert werden. Um Compilen zu können, ist Software nötig:**

```
sudo apt-get install g++ make libsigc++-1.2-dev libgsm1-dev libpopt-dev tcl8.5-dev libgcrypt-dev
    libspeex-dev libasound2-dev alsa-utils
```

- **SvxLink herunterladen, auspacken und Compilieren (dauert ca. 10-15 Minuten):**

```
wget http://sourceforge.net/projects/svxlink/files/svxlink/11.11/svxlink-11.11.1.tar.gz
tar xvzf svxlink-11.11.1.tar.gz
cd svxlink-11.11.1.tar.gz
sudo make
sudo install
```

Svxlink-/Echolink-Gateway (4/5)

/etc/svxlink/svxlink.d/ ModuleEchoLink.conf

```
CARD_SAMPLE-RATE 44100
…
[ModuleEchoLink]
NAME=EchoLink
ID=2
TIMEOUT=60
#ALLOW_IP=192.168.1.0/24
#DROP_INCOMING=^()$
#REJECT_INCOMING=^()$
#ACCEPT_INCOMING=^(.*)$
#REJECT_OUTGOING=^()$
#ACCEPT_OUTGOING=^(.*)$
SERVER=europe.echolink.org
CALLSIGN=IhrCall
PASSWORD=IhrPasswort
SYSOPNAME=IhrOpName
LOCATION=[Svx] Ihr QTH & QRG
MAX_QSOS=3
MAX_CONNECTIONS=3
LINK_IDLE_TIMEOUT=300
DESCRIPTION="SvxLink Node\n"
```

/etc/svxlink/svxlink.conf

```
…
[SimplexLogic]
TYPE=Simplex
RX=Rx1
TX=Tx1
MODULES=ModuleEchoLink
CALLSIGN=IhrCall
…
[Rx1]
…
AUDIO_DEV=alsa:default
…
SERIAL_PORT=/dev/ttyUSB0
…
[Tx1]
…
AUDIO_DEV=alsa:hw:0
…
PTT_PORT=/dev/ttyUSB0
```

/etc/asound.conf

```
pcm.!default {
    type plug
    slave {
      channels 1
        pcm "hw:0,0"
    }
}
pcm.low {
    type plug
    slave {
        pcm "hw:0,0"
    }
}
```

/etc/modules

```
snd-usb-audio
#snd-bcm2835
```

/etc/modprobe.d/alsa-base.conf

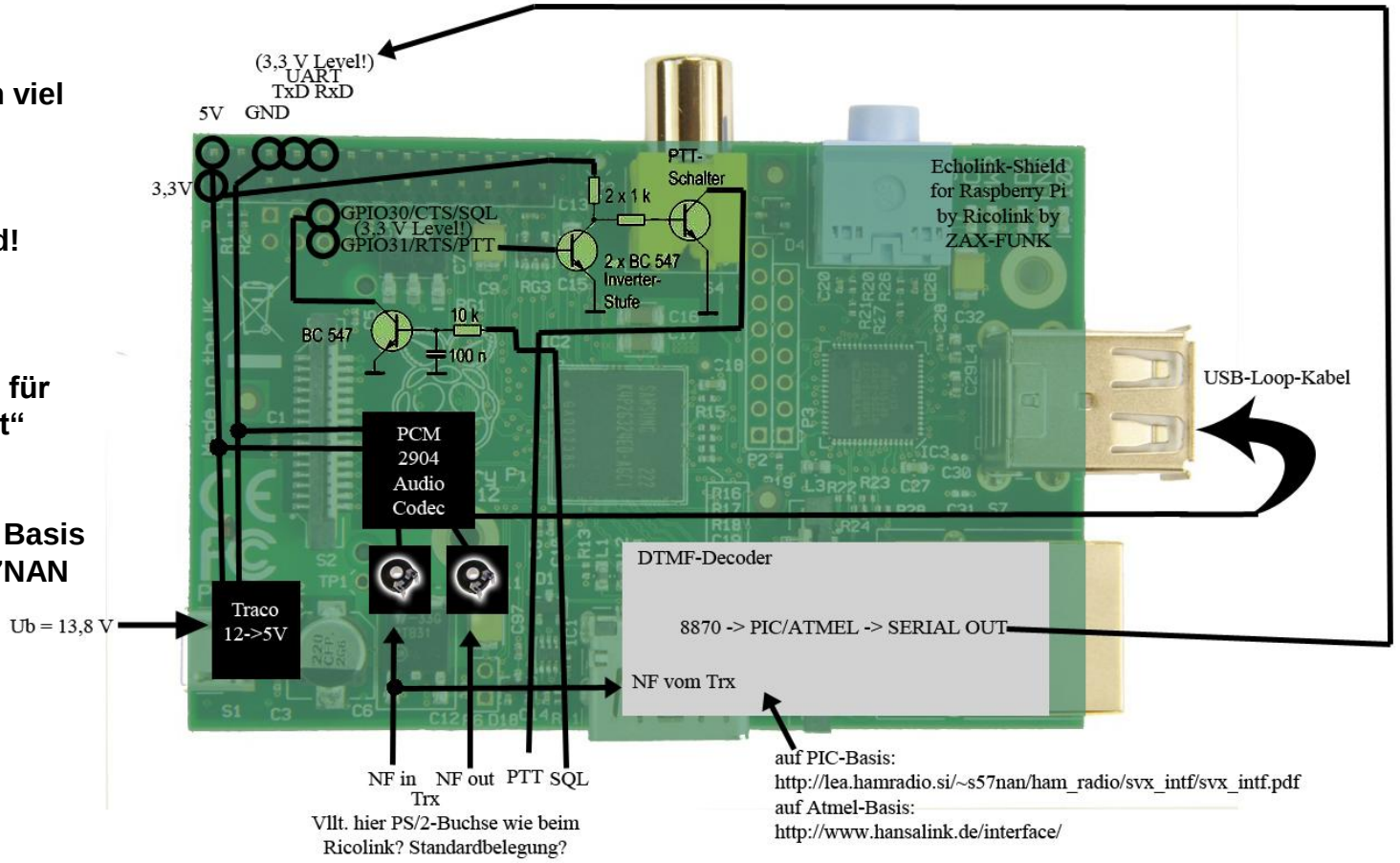
```
Options snd-usb-audio index=0
```

Svxlink-/Echolink-Gateway (5/5)

Problem:
- **Trotz kompakter Hardware dennoch viel „Drahtverhau“**

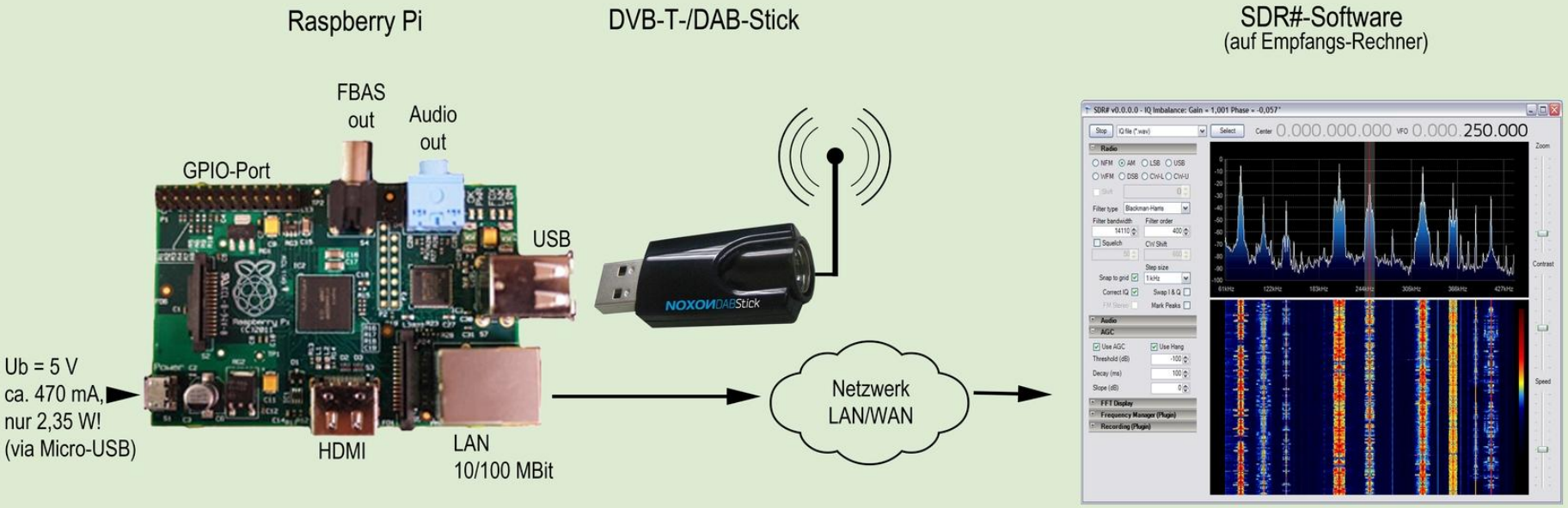
Lösung:
- **Ein Echolink-Shield!**

Pflichtenheft:
- **Betrieb an 13,8 V**
- **Eigene Soundkarte für Audio „in“ und „out“**
- **NF-Pegelregelung**
- **PTT via GPIO**
- **DTMF-Decoder auf Basis Hansalink oder S57NAN**



LAN-SDR mit DAB-/DVB-T-Stick (1/4)

Deutscher Amateur-Radio-Club e.V.
Bundesverband für Amateurfunk in Deutschland



LAN-SDR mit DAB-/DVB-T-Stick (2/4)

Deutscher Amateur-Radio-Club e.V.
Bundesverband für Amateurfunk in Deutschland

Für die Umsetzung zunächst ein paar Pakete zur Installation:

```
sudo apt-get install git cmake libusb-1.0-0.dev
  build-essential
```

Ausgehend vom Home-Verzeichnis /home/pi führt man die Installationsschritte aus:

```
git clone git://git.osmocom.org/rtl-sdr.git
cd rtl-sdr/
mkdir build
cd build
cmake ../
make
sudo make install
sudo ldconfig
```

Damit das System den USB-Stick ansprechen kann:

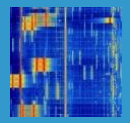
```
cp /home/pi/rtl-sdr.rules /etc/udev/rules.d
```

Tipps:

Die Anwendung benötigt nur 10 % CPU-Last.

Bei Versuchen mit einem Terratec Noxon-Stick blockierte das Modul „dvb_usb_rtl28xxu“ den Stick, weshalb es mit folgendem Befehl aus dem Speicher temporär entfernt wurde:

```
sudo rmmod dvb_usb_rtl28xxu
```



LAN-SDR mit DAB-/DVB-T-Stick (3/4)

Deutscher Amateur-Radio-Club e.V.
Bundesverband für Amateurfunk in Deutschland

Mit „*Rtl_test*“ kann man feststellen, ob der Raspberry Pi den SDR-Stick „sieht“:

```
pi@raspberrypi ~ $ rtl_test -t
Found 1 device(s):
  0:  Terratec Cinergy T Stick RC (Rev.3)

Using device 0: Terratec Cinergy T Stick RC (Rev.3)
Found Elonics E4000 tuner
Supported gain values (14): -1.0 1.5 4.0 6.5 9.0 11.5 14.0 16.5 19.0 21.5 24.0 29.0 34.0 42.0
Benchmarking E4000 PLL...
[E4K] PLL not locked for 51000000 Hz!
[E4K] PLL not locked for 2185000000 Hz!
[E4K] PLL not locked for 1093000000 Hz!
[E4K] PLL not locked for 1236000000 Hz!
E4K range: 52 to 2184 MHz
E4K L-band gap: 1093 to 1236 MHz
```

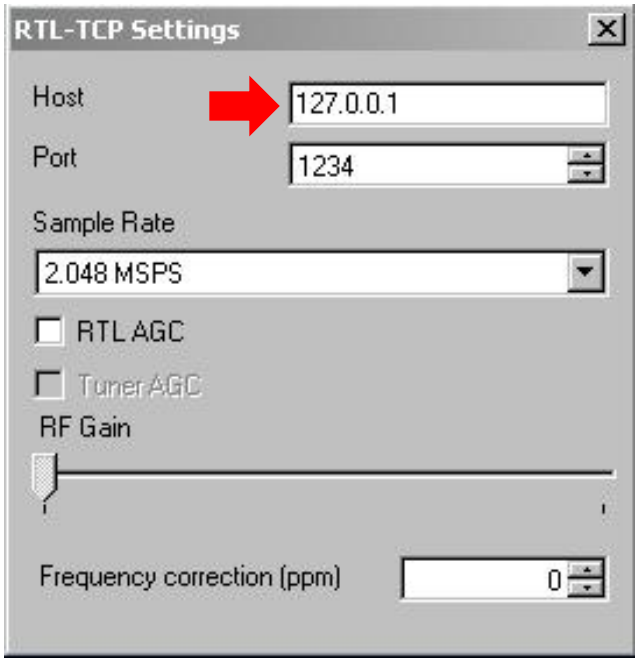
Server wird mit „Rtl_tcp“ gestartet (IP diejenige des Raspberry Pi‘s):

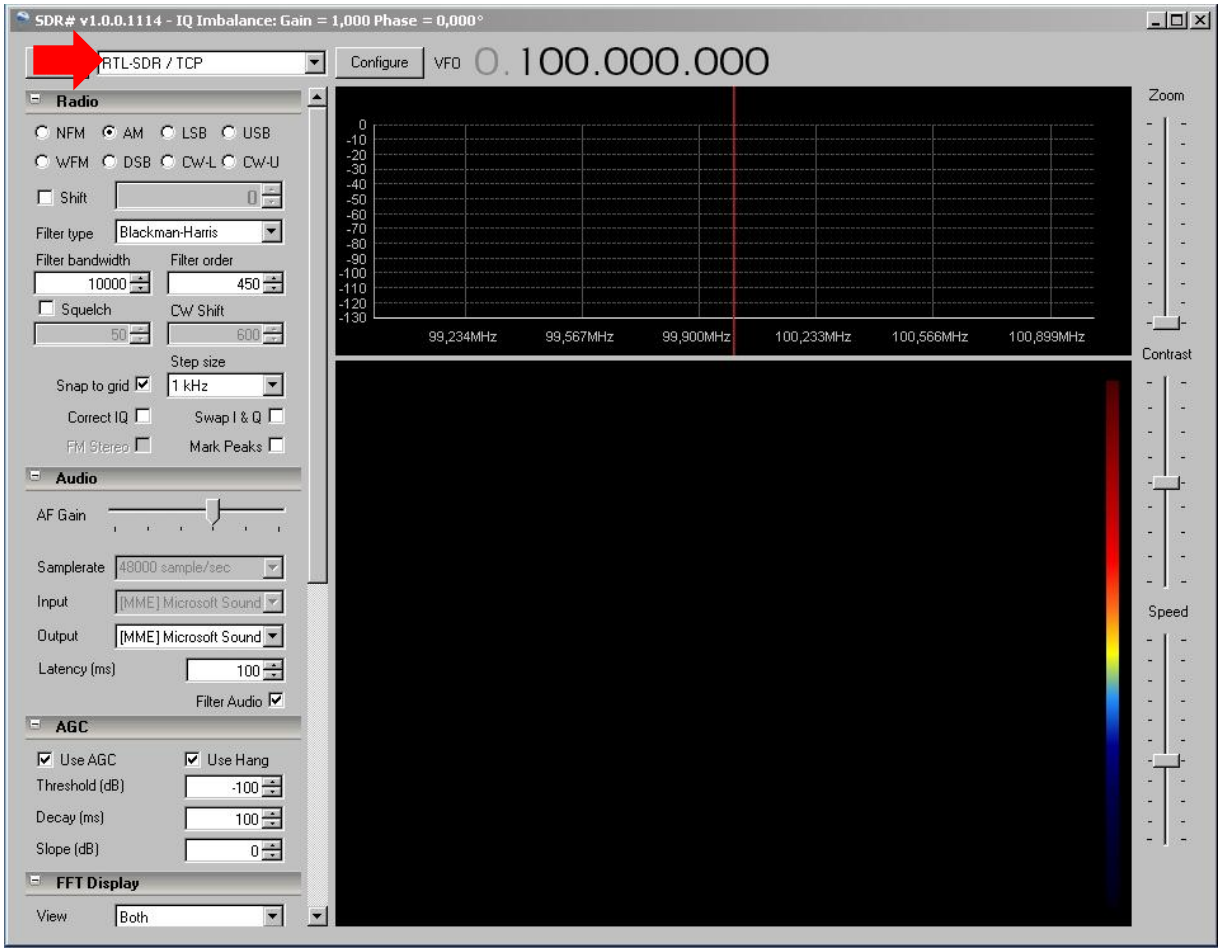
```
pi@raspberrypi ~ $ rtl_tcp -a 192.168.6.100
Found 1 device(s). Found Elonics E4000 tuner. Using Terratec Cinergy T Stick RC (Rev.3)
Tuned to 100000000 Hz.
listening...
Use the device argument 'rtl_tcp=192.168.6.100:1234' in OsmoSDR (gr-osmosdr) source
to receive samples in GRC and control rtl_tcp parameters (frequency, gain, ...).
```

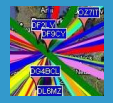
LAN-SDR mit DAB-/DVB-T-Stick (4/4)

Deutscher Amateur-Radio-Club e.V.
Bundesverband für Amateurfunk in Deutschland

Empfangsseitig wählt man im SDR# als Empfänger „RTL-SDR / TCP“:

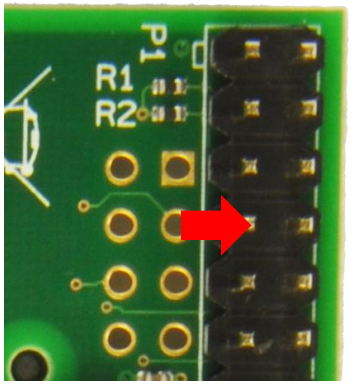




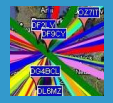


Mit dem Clockgenerator auf Sendung (1/2)

Deutscher Amateur-Radio-Club e.V.
Bundesverband für Amateurfunk in Deutschland



- **Idee: Raspberry Pi über den bordeigenen Clockgenerator an GPIO-Pin 4 direkt als Sender nehmen!**
- **Problem: Rechtecksignal ungeeignet zur Aussendung (stark ausgeprägtes Oberwellenspektrum). (Tiefpass-)Filter erforderlich!**
- **Die Software "PiFm" kann z.B. im UKW-Radio-Bereich Wave-Dateien aussenden http://www.icrobotics.co.uk/wiki/index.php/Turning_the_Raspberry_Pi_Into_an_FM_Transmitter**
- **Guido, PE1NNZ: Erzeugung eines SSB-Signals durch Modulation des PLL-Trägers. NF-Input über USB-Soundkarte, 1 W HF mit 3× BS170 MOSFETs. QSOs auf 40/20 m in Europa, Rx via WebSDR. http://pe1nnz.nl.eu.org/**
- **PE1NNZ‘s zweiter Streich: WSPR (gesprochen „Whisper“) mit dem Raspberry Pi**
  - **Entwickelt vom Physik-Nobelpreisträger Joe Taylor, K1JT zur Erforschung der Ausbreitungs-Bedingungen**
  - **Frequenzumtastung von vier Symbolfrequenzen (4-FSK, Abstand 1,46 Hz), Bandbreite 5,9 Hz**
  - **Vorwärts-Fehlerkorrektur**
  - **Genaue Systemzeit erforderlich, z.B. via NTP-Protokoll (ptbtime1.ptb.de)**



Mit dem Clockgenerator auf Sendung (2/2)

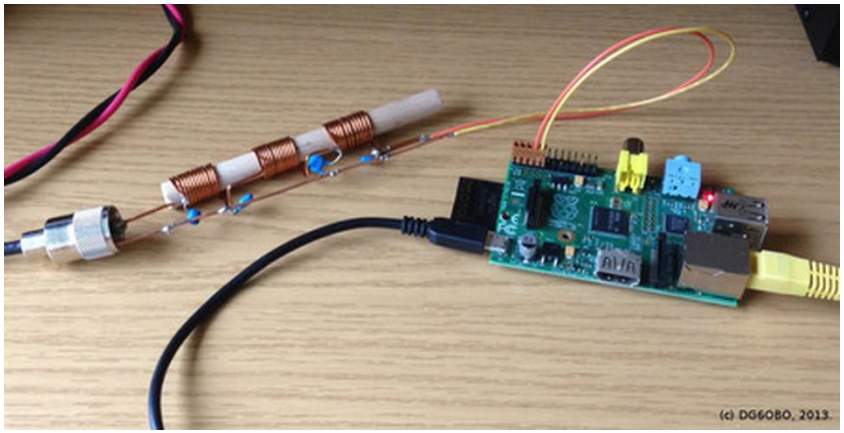
Deutscher Amateur-Radio-Club e.V.
Bundesverband für Amateurfunk in Deutschland

- **Clockgenerator jedoch nicht recht ungenau und Frequenz gar temperaturabhängig. Software mit Stand Juli 2013 kann anhand eines Zeitservers kalibriert werden**

- **Installation: GCC-Compiler laden, Software aus GIT-Repository beziehen, Kompilieren und aufrufen**

```
sudo apt-get install gcc git
git clone https://github.com/threeme3/WsprryPi.git
gcc -lm -owspr wspr.c
sudo ./wspr <Callsign> <Locator> <Power in dBm> <Frequenz in Hz>
```

- **Erfolge/Fotos von Jan, DG6OBO: mit 10 mW HF ODX 2020 km!**

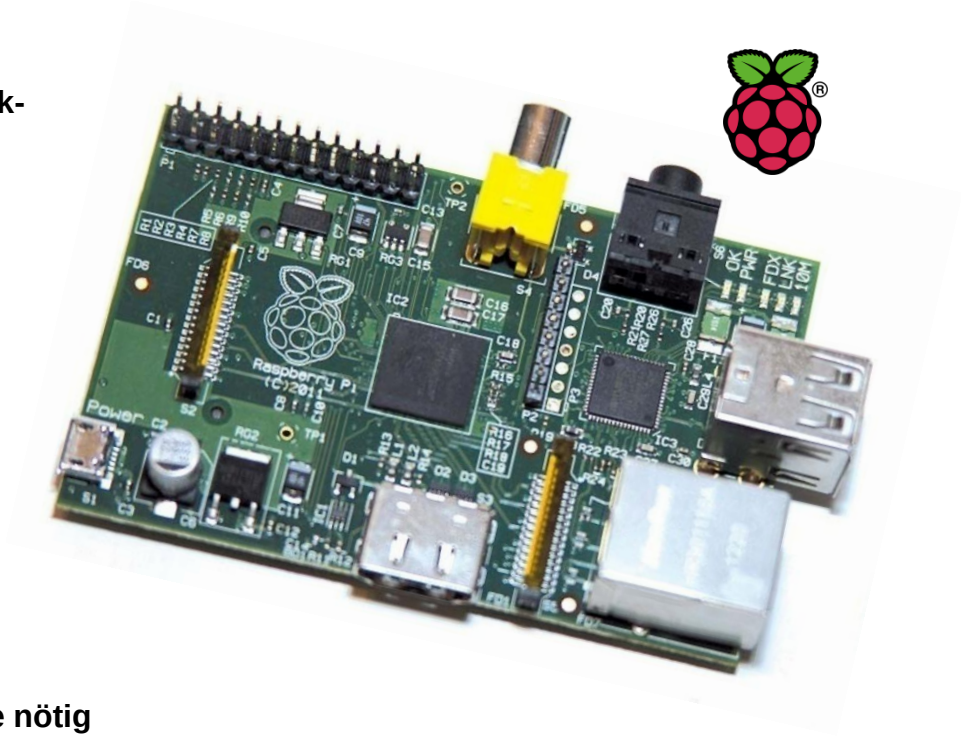


Fazit

Deutscher Amateur-Radio-Club e.V.
Bundesverband für Amateurfunk in Deutschland

Vorteile

- **Stromsparsamer Server-Rechner für Amateurfunk-anwendungen und Vieles mehr …**
- **Läuft mit USB-Ladegeräten oder gar mit Akkus**
- **Externe Steuermöglichkeiten via GPIO-Pinleiste**
- **Sehr klein**
- **Sehr preiswert**
- **Gute Unterstützung in Punkto Software durch große Community – vgl. andere auch modernere Kleincomputer**
- **Dank GPU FullHD-Video-Wiedergabe bei Ausbau eines Media-Centers (XBMC)**
- **HF direkt aus dem GPIO4-Pin**



Nachteile

- **Rechenleistung begrenzt, speziell beim Desktop-Einsatz**
- **Kein Soundeingang vorhanden, USB-Soundkarte nötig**
- **Stromzufuhr über Mikro-USB-Buchse in mind. einem Fall sehr „wackelig“ :-/**
- **Kein Hardware-Handshake (RTS, CTS, DSR, DTR, RI fehlt) auf Standard-GPIO-Leiste (@Model B, V2 nachbestückbar!)**
- **HF aus Clockgenerator/GPIO4 muss natürlich zwingend ein Filter nachgeschaltet werden**

Dank/Literatur

Vielen Dank für die Aufmerksamkeit!

```
pi@raspberrypi ~ $ uptime
 10:53:09 up 107 days,  4:36,  1 user,  load average: 0.50, 0.33, 0.33
```

73 de Stefan, DH5FFL@darc.de

Inhalte des Vortrags zum Nachlesen:

[1] Vortrag mit Stand 6/13 online via: darc.de -> DARC-Info -> Geschäftsstelle -> Verbandsbetreuung -> HamRadio-Vorträge -> HamRadio 2013
[2] Svxlink-Anwendung: http://svxlink.de/?page_id=1606
[3] Stefan Hüpper, DH5FFL: „Amateurfunkanwendungen mit dem Raspberry Pi“, Tagungsskript UKW-Tagung 2013, Bezug: www.box73.de
[4] Stefan Hüpper, DH5FFL: „Tipps und Tricks zum Raspberry Pi“, CQ DL 11/13, S. 774ff.
[5] Stefan Hüpper, DH5FFL: „Auf dem Weg zum LAN-SDR“, CQ DL 4/13, S. 239
[6] Stefan Hüpper, DH5FFL: „APRS-iGate mit dem Raspberry Pi“, CQ DL 4/13, S. 240ff.
[7] Stefan Hüpper, DH5FFL: „Ein Echolink-Rechner mit <3 W Leistungsaufnehme“, CQ DL 10/12, S. 695ff.