

Introduction to the RTL-SDR

Dayton Software Defined Radio Meetup
March 7, 2016

Welcome to Our First Meetup!



- Goal: Discuss interesting topics related to software defined radios and signal processing.
- Plan on hosting monthly meetings.
 - Possibly also host a special meeting during Hamvention week in May
- Looking for feedback:
 - What kinds of topics would you like to see presented?
 - What meeting time works best?
- Introductions
- Have an interesting topic or demo that you'd like to share? Please e-mail bhart@pretalen.com .

Meetup Sponsor -- PreTalen



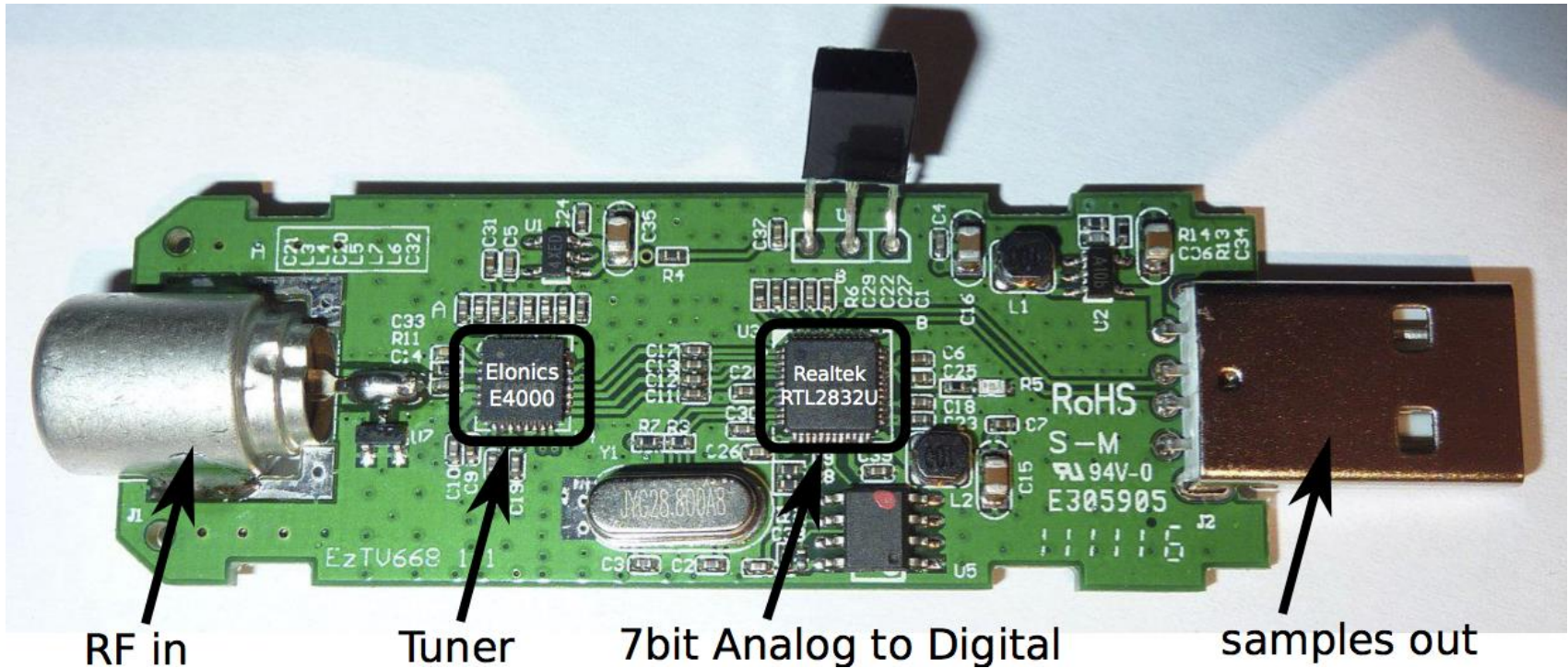
- Thank you to our sponsor PreTalen for paying our group hosting fees, providing food and allowing us to use the office!
- If you are interested in signal processing and software defined radio PreTalen is currently looking to hire engineers in our Dayton office.
- Great opportunity to join a fast growing company with amazing benefits.
- Visit <http://www.pretalen.com/careers> for more information or see me after the meeting if you are interested in learning more.

What is an RTL-SDR anyway?

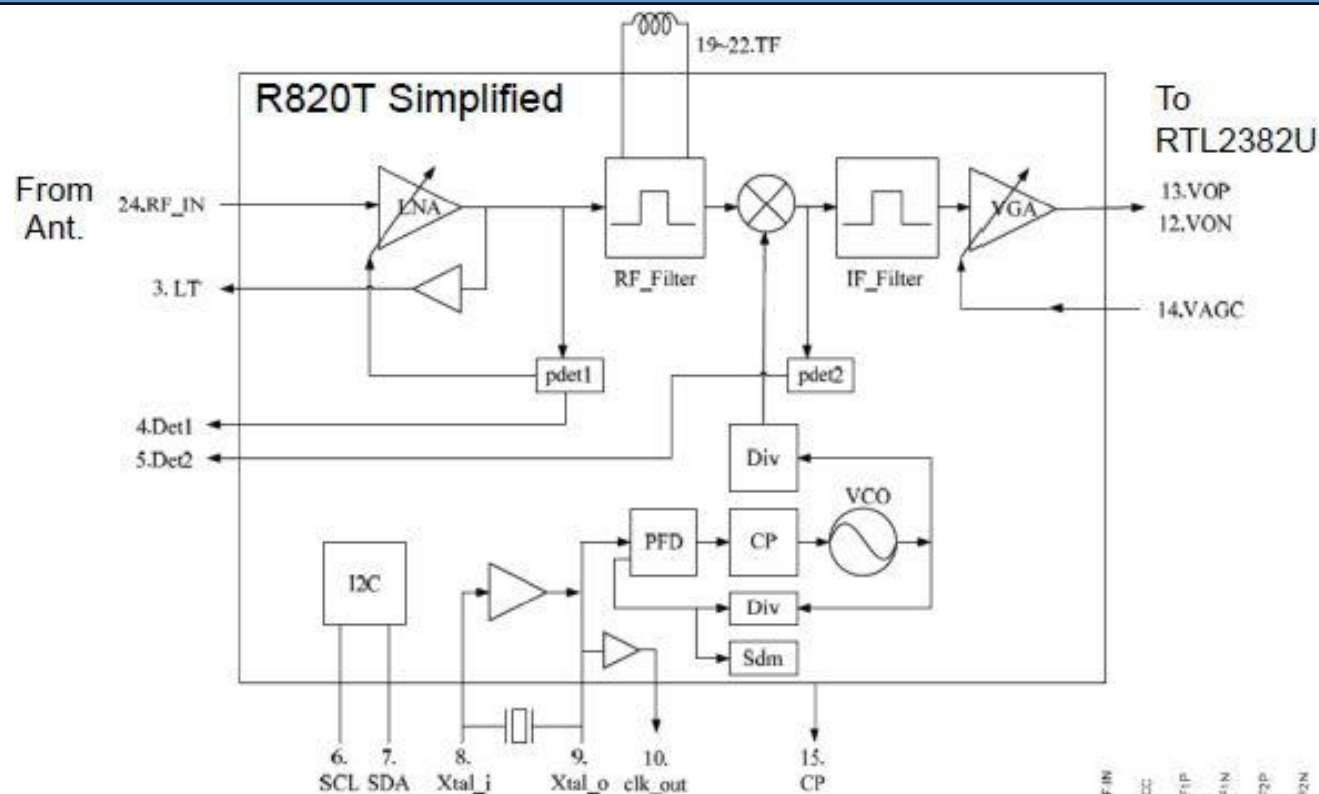
- Originally created as a low cost DVB-T receiver in a USB device
 - DVB-T is the TV transmission standard for Europe, Asia (except China), Australia and Africa
- Hobbyists discovered general purpose capabilities
- Using custom software, device can tune to other frequencies.
- Chipset is readily available to overseas manufacturers



Circuit Board



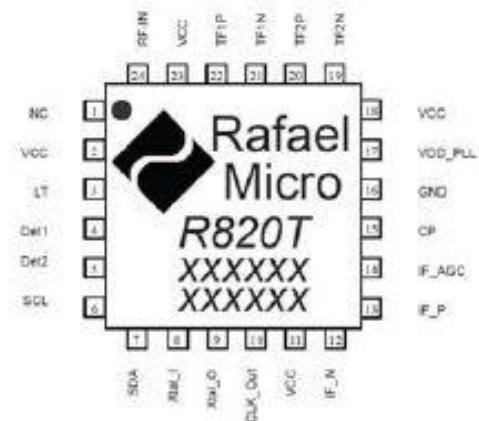
Tuner Component



Typical figures

- Frequency range: 42 to 1002 MHz
- Noise figure : 3.5 dB @ RF_IN
- Phase noise: -98 dBc/Hz @ 10 kHz
- Current consumption: < 178 mA @ 3.3V power supply
- Max input power: +10 dBm
- Image rejection: 65 dBc

note: [dBm]=[dBuV on 75Ω] -108.75dB



Features of the RTL-SDR

- Low cost – less than \$20 on Amazon
- Wide tuning range – 24 MHz to 1.7 GHz
- Instantaneous bandwidth of up to 3.2 MHz
- Works with almost any computer
 - Windows and Linux compatible
- Open source software
- Small and easy to use -- plugs into any USB 2.0 port on your computer

Limitations of the Hardware

- Cheap oscillator
 - Frequencies are inaccurate (up to +/- 70 ppm error)
 - Drifts with temperature
 - Phase noise
- Limited dynamic range
 - Only an 8 bit ADC
 - Roughly 45 dB of SFDR
- Cheap antenna
- No shielding

Setting Up the RTL-SDR on Linux

- Install GNURadio (recommended):

```
sudo apt-get install gnuradio-dev
```

- Install RTL-SDR library:

```
git clone https://github.com/balint256/gr-baz
cd gr-baz
sh bootstrap
./configure
make
sudo make install
sudo ldconfig
```

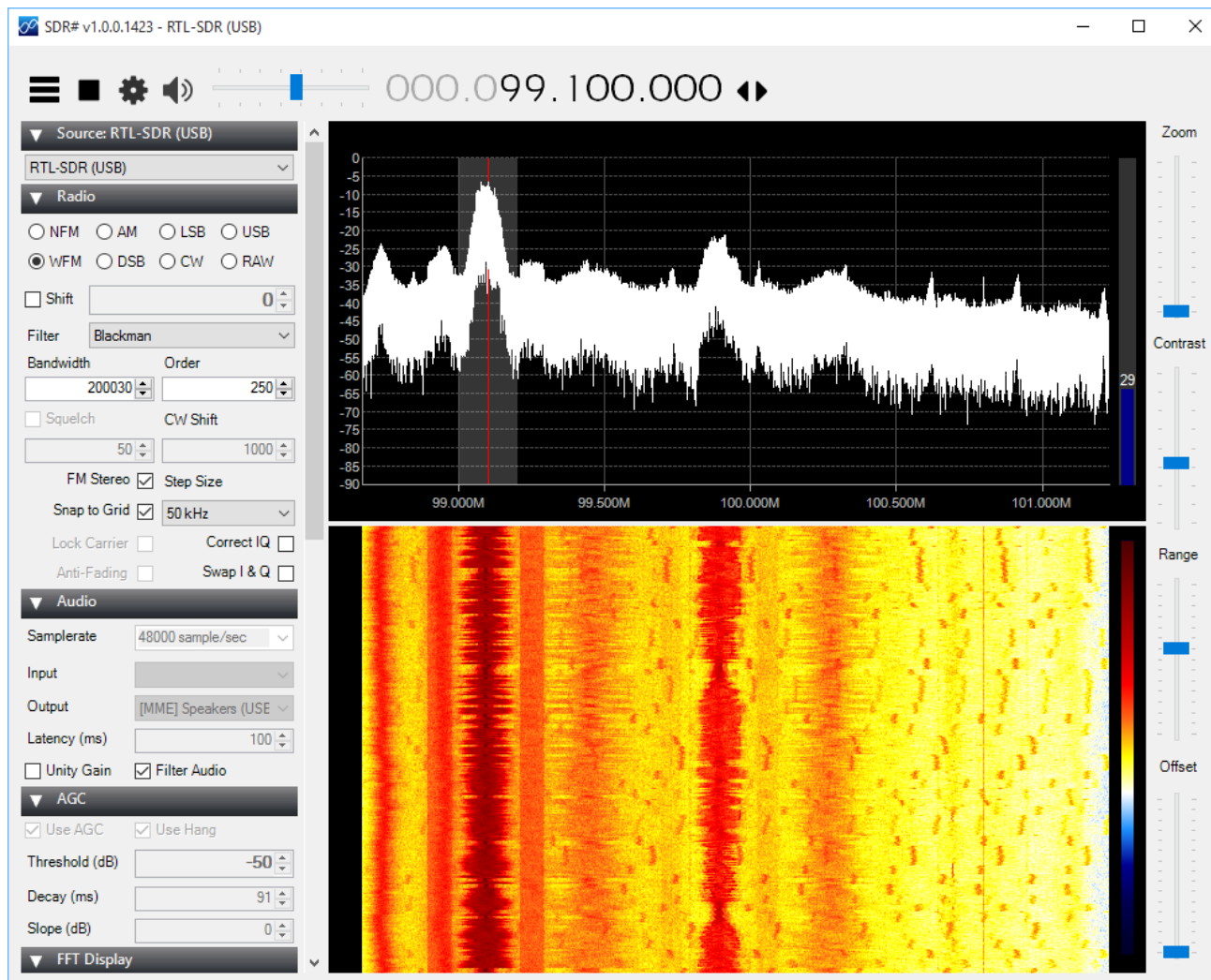
- Install GQRX:

```
sudo apt-get install gqrx-sdr
```

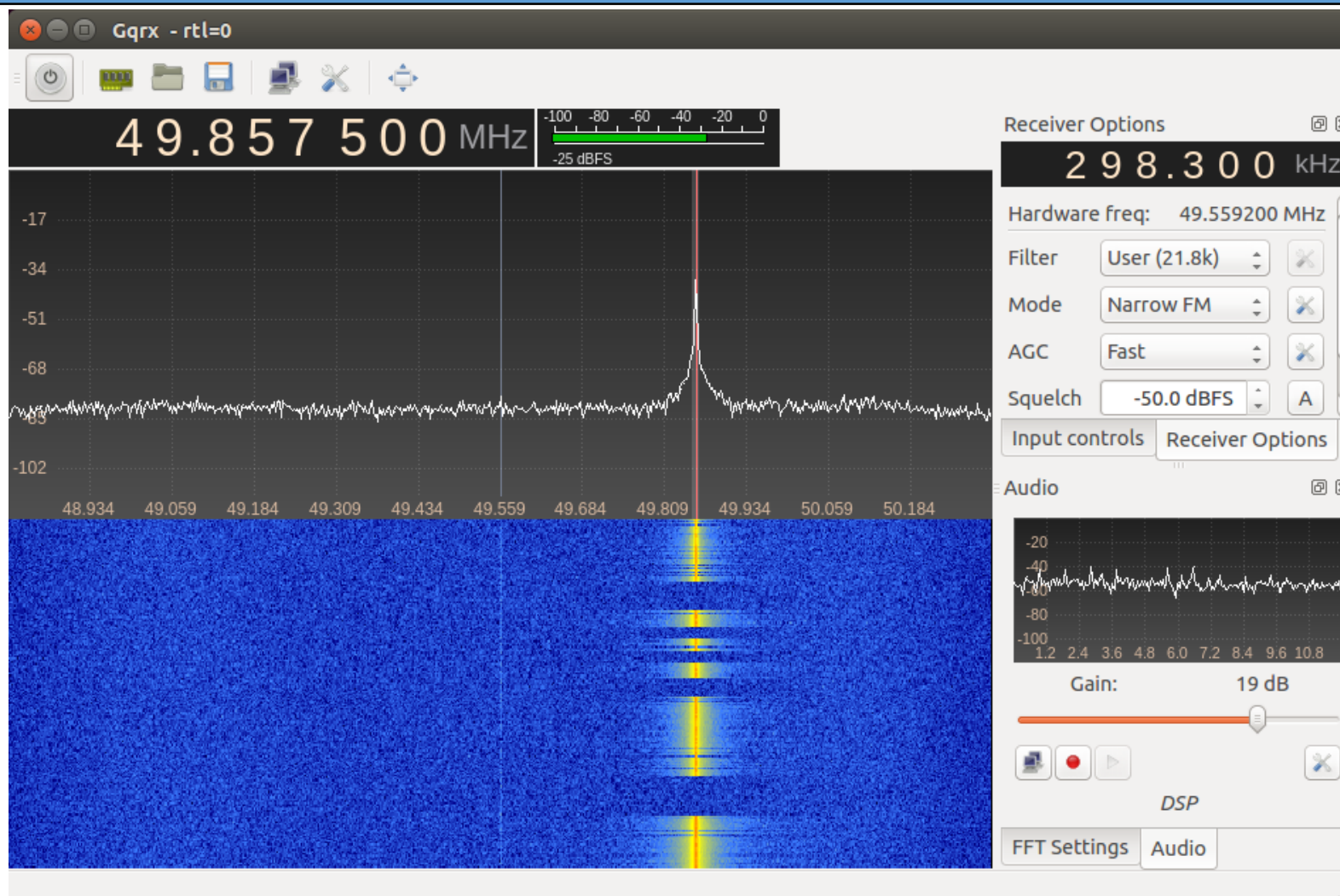
Blacklist Incompatible Driver

- `dvb_usb_rtl28xxu` module is incompatible with driver for RTL-SDR
- Need to add to blacklist
- Edit `/etc/modprobe.d/rtlsdr.conf`
- Add the following line and reboot:
 - `blacklist dvb_usb_rtl28xxu`
- Can also run (if module is not locked):
 - `rmmod dvb_usb_rtl28xxu`

SDRSharp for Windows



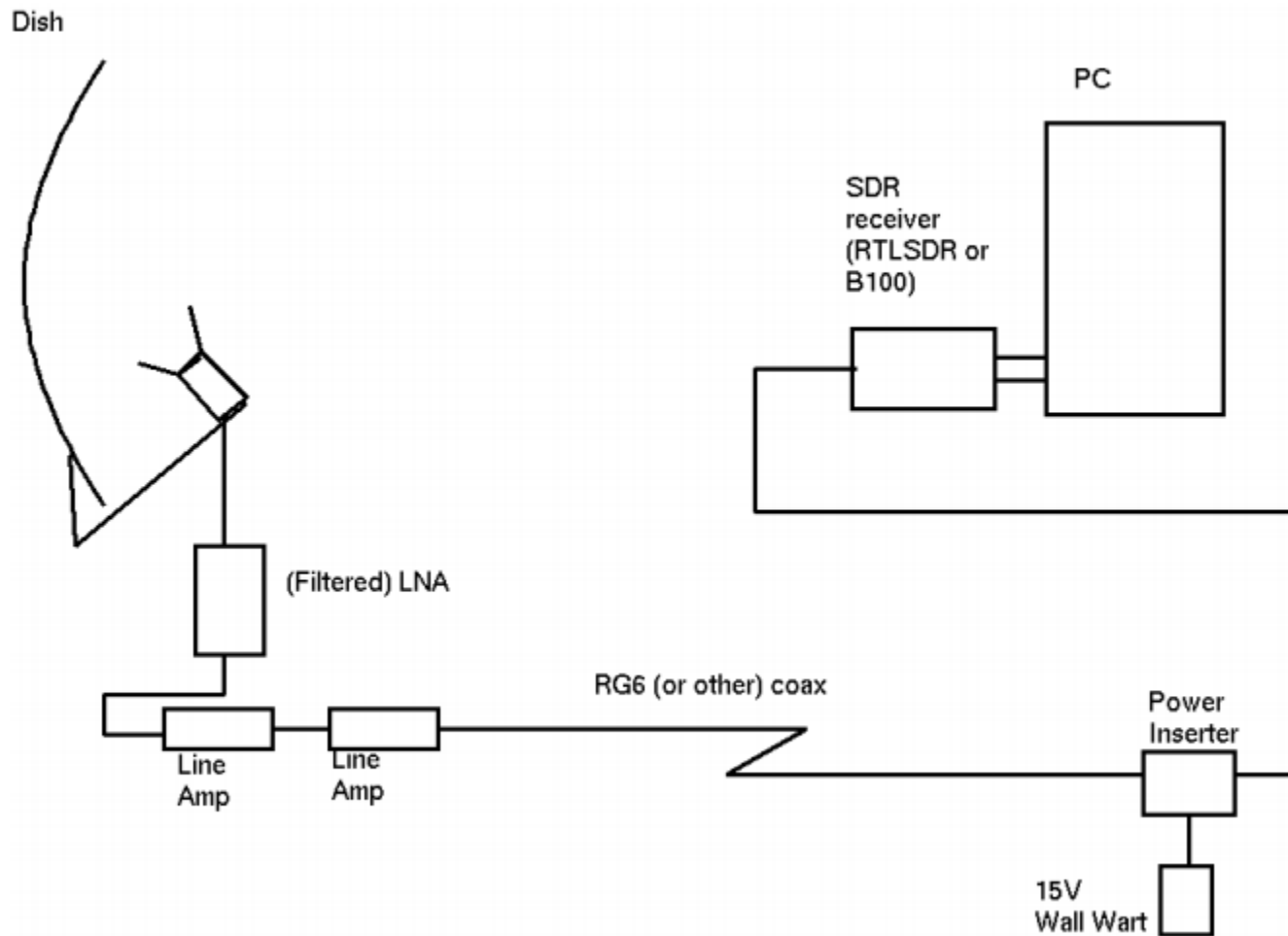
GQRX for Linux



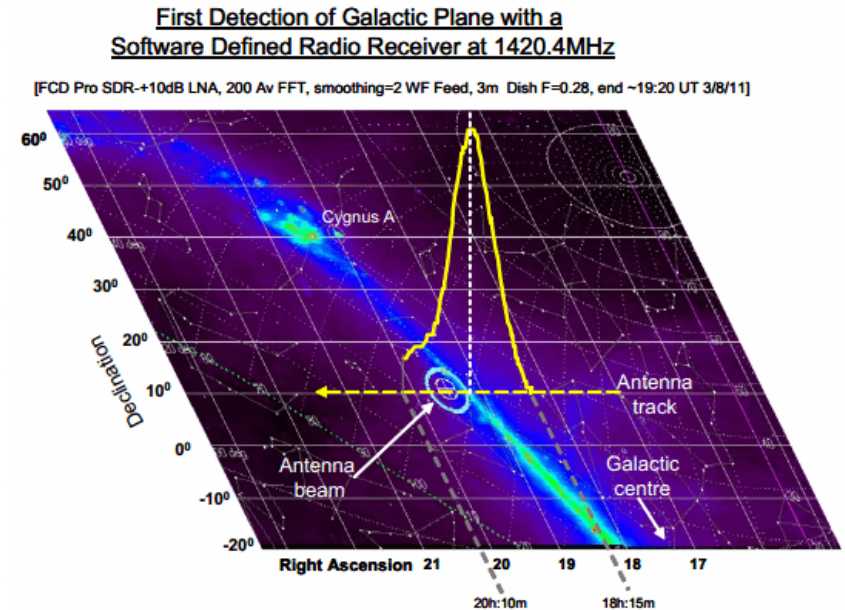
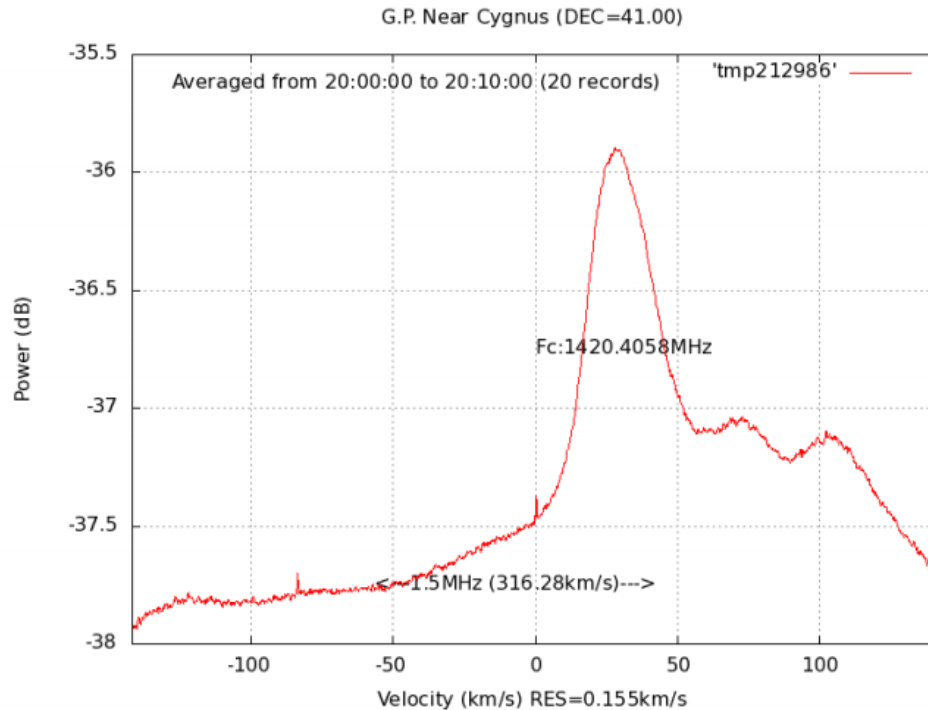
GQRX Demo

- This will show a demo of the GQRX software in action.
- FM radio stations:
 - 107.7 MHz
 - 103.9 MHz
 - 104.7 MHz
 - 88.1 MHz

Cool Application – Radio Telescope



Radio Telescope Results

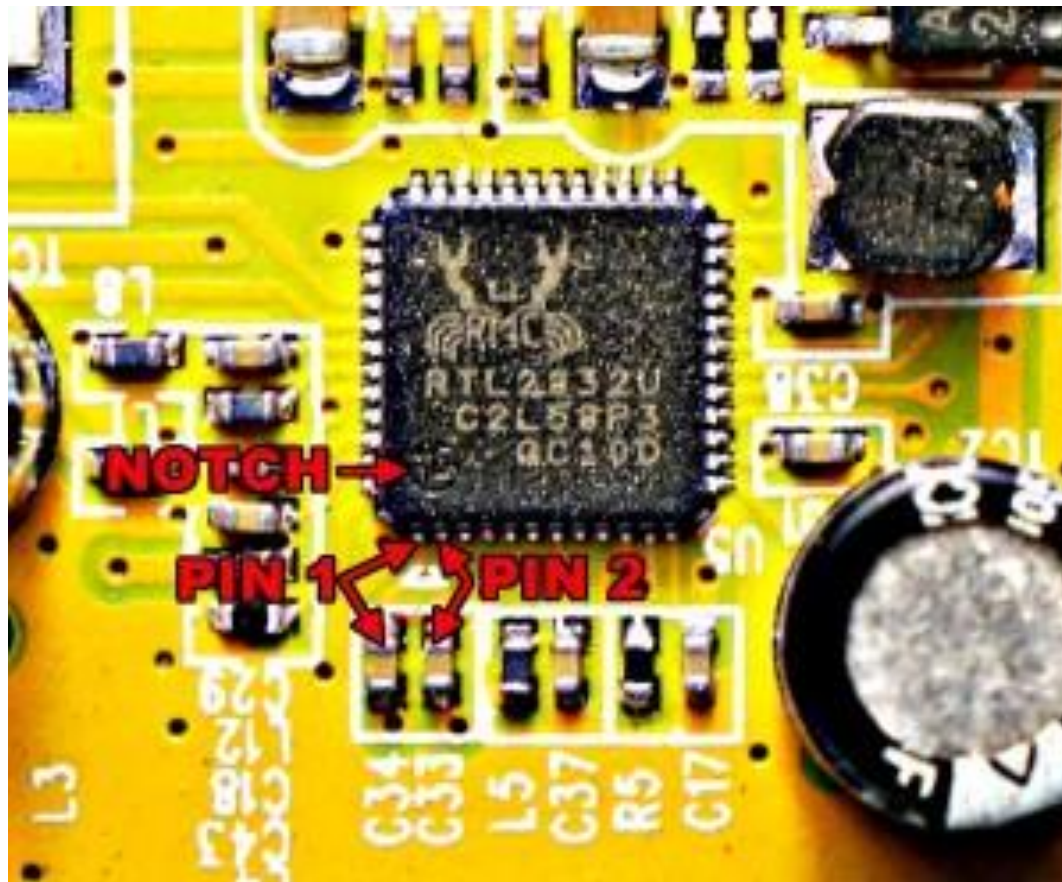


- More details:

http://www.sbrac.org/files/budget_radio_telescope.pdf

Hardware Modification – Lower Tuning Range

- Solder a wire to pin 1 or pin 2 of the RTL chip

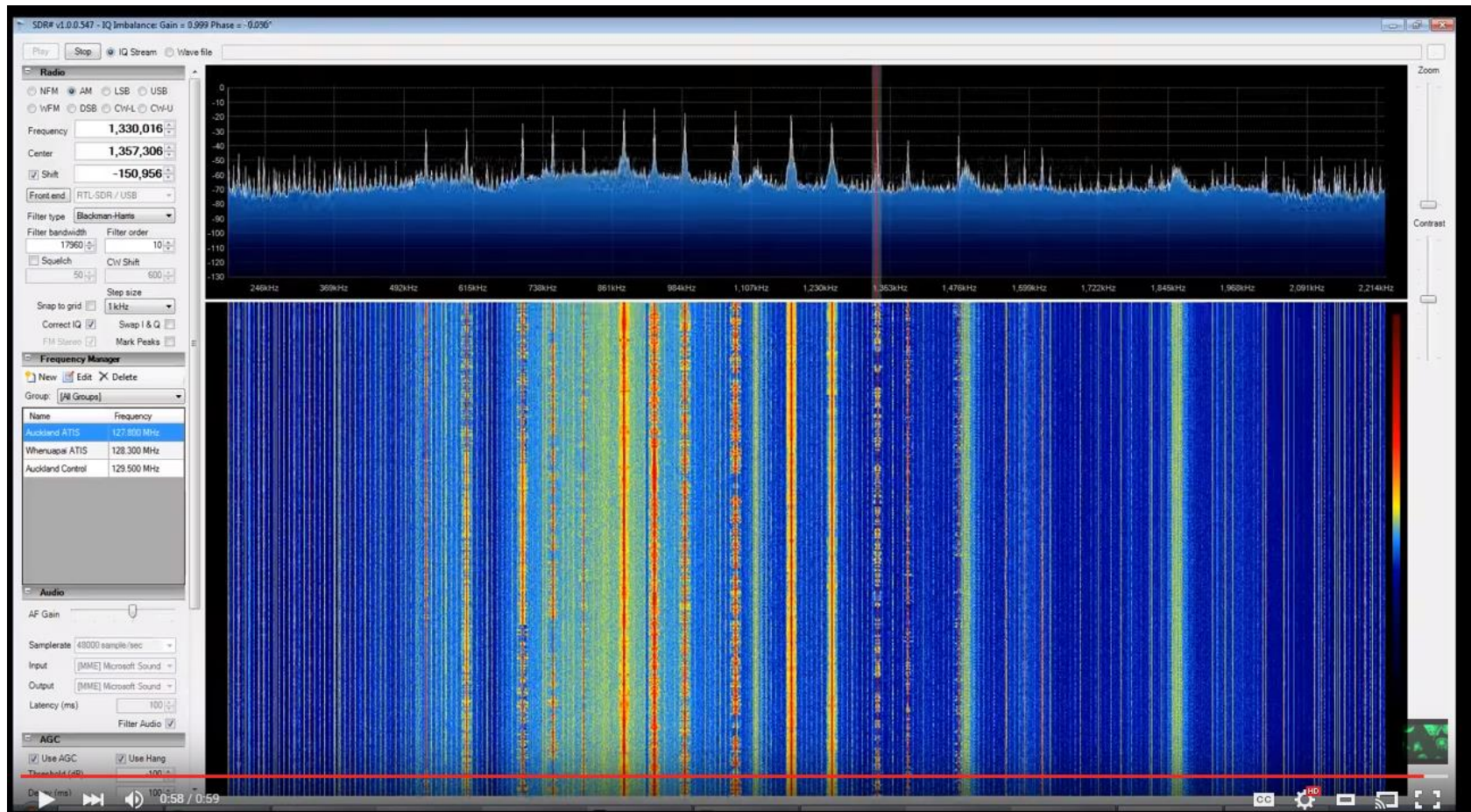


Hardware Modification – Lower Tuning Range

- How does the modification work?
- Wire acts as an antenna – bypasses tuner and goes directly into ADC
- Receives signals from 0-14.4 MHz
- Can also receive image from 14.4-28 MHz if band pass filter is used
- FM interference can be a problem – filtering is recommended
- Matching transformer for differential input can also improve the reception at low frequencies
- More information: <http://www.rtl-sdr.com/rtl-sdr-direct-sampling-mode/>

AM Radio Demo with Low Frequency Mod

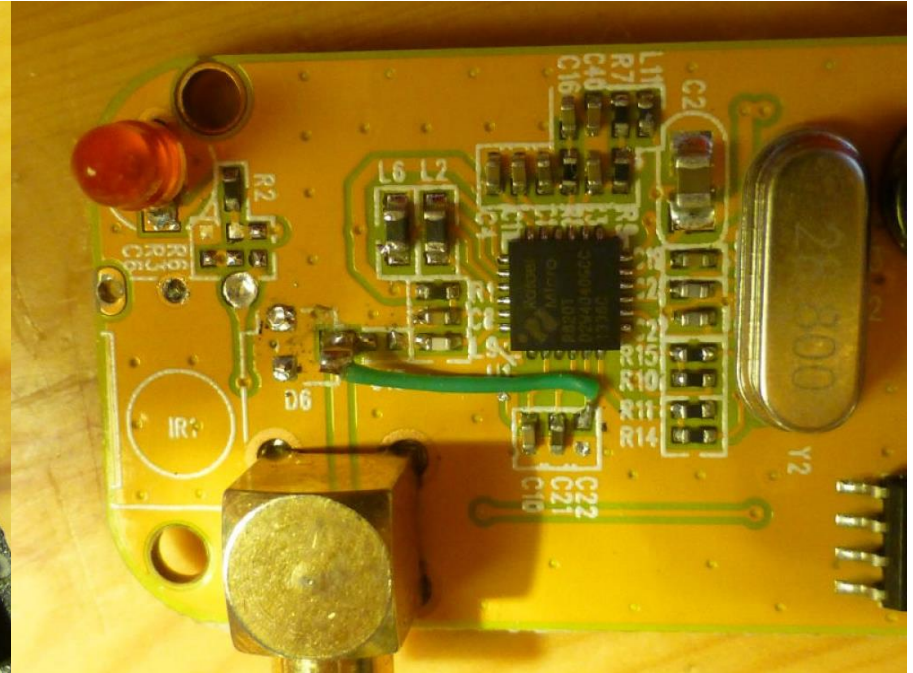
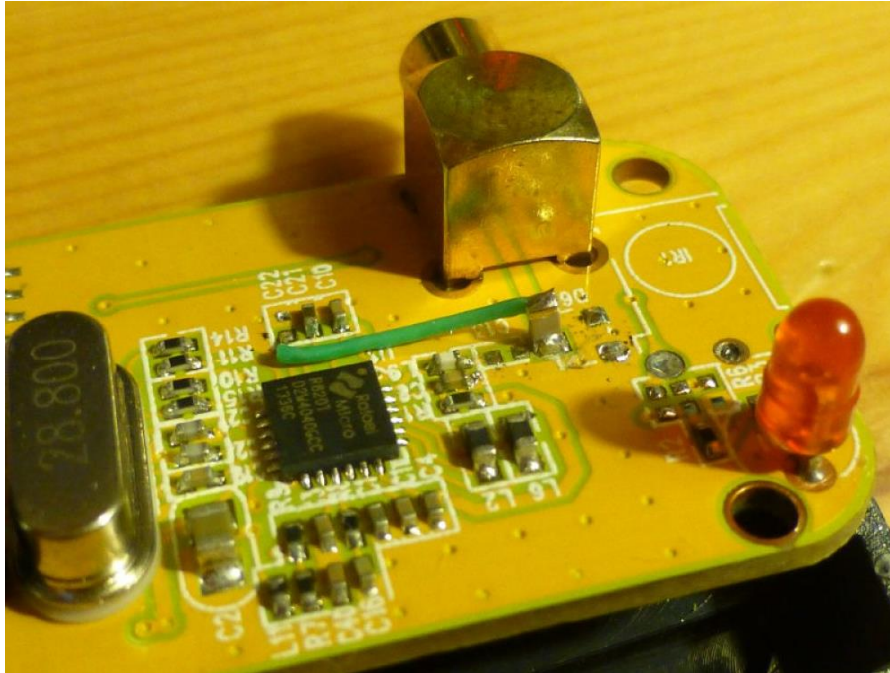
- <https://www.youtube.com/watch?v=JCE9SeQ3dJQ>



Hardware Modification – Tone Transmission

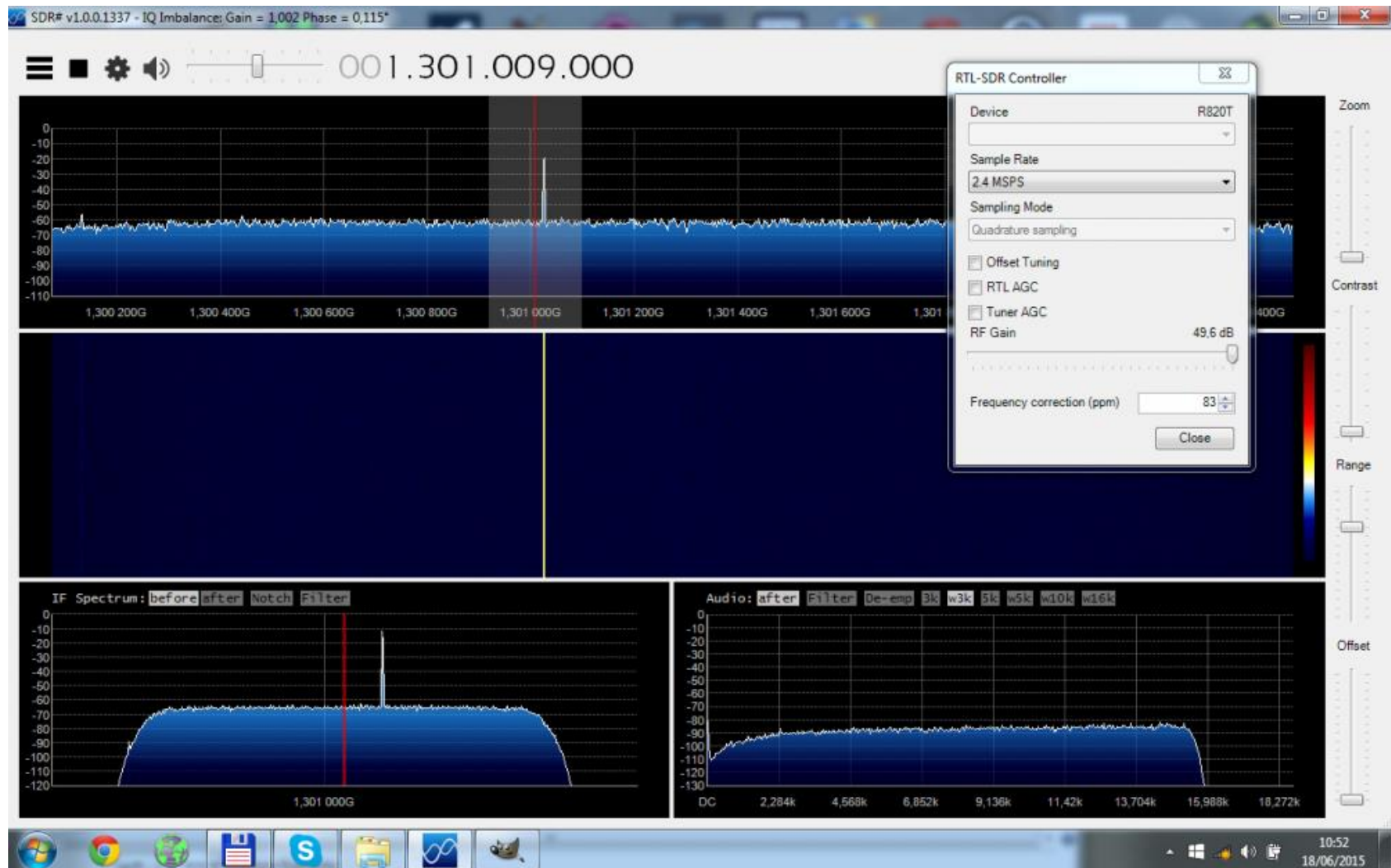
- Possible to use the hardware to transmit tones.
- Be careful! Could violate FCC regulations if transmitting in regulated bands.
- Hardware mod steps:
 - Remove the input capacitor C13 ;
 - Remove the bypass capacitor C22 ;
 - Remove the protection diode D6 ;
 - Wire a capacitor 100pF connecting the Antenna input (D6 pad) to the pin 5 of R820T (ex C22 pad).

Hardware Modification – Tone Transmission



- More info:
http://www.steila.com/blog/index.php?controller=post&action=view&id_post=8

Tone Transmission



RTL-SDR in GNURadio

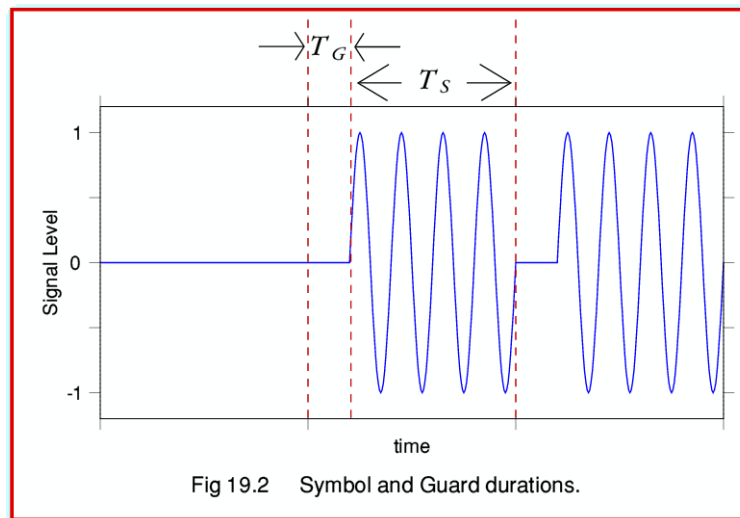
- RTL-SDR can act as a source directly in GNURadio
- Makes it easy to tune and acquire samples
- Simple to do things like:
 - Record data to a file
 - Demodulate signals to audio
 - Filtering of signal data
 - Plot signal waterfall or frequency spectrum
- GNURadio companion exposes GNURadio components as drag and drop blocks

RC Car Controller Demonstration

- Goal: Demonstrate decoding an RF protocol using the RTL-SDR
- Something different than online tutorials
- More interesting than broadcast AM/FM station
- Selected an RC car controller:
 - 49 MHz center frequency
 - RF protocol is unknown – need to use GNU radio to analyze and build decoder
 - 8 different commands : up, down, left, right, up+left, up+right, down+left, down+right

RC Signal Analysis

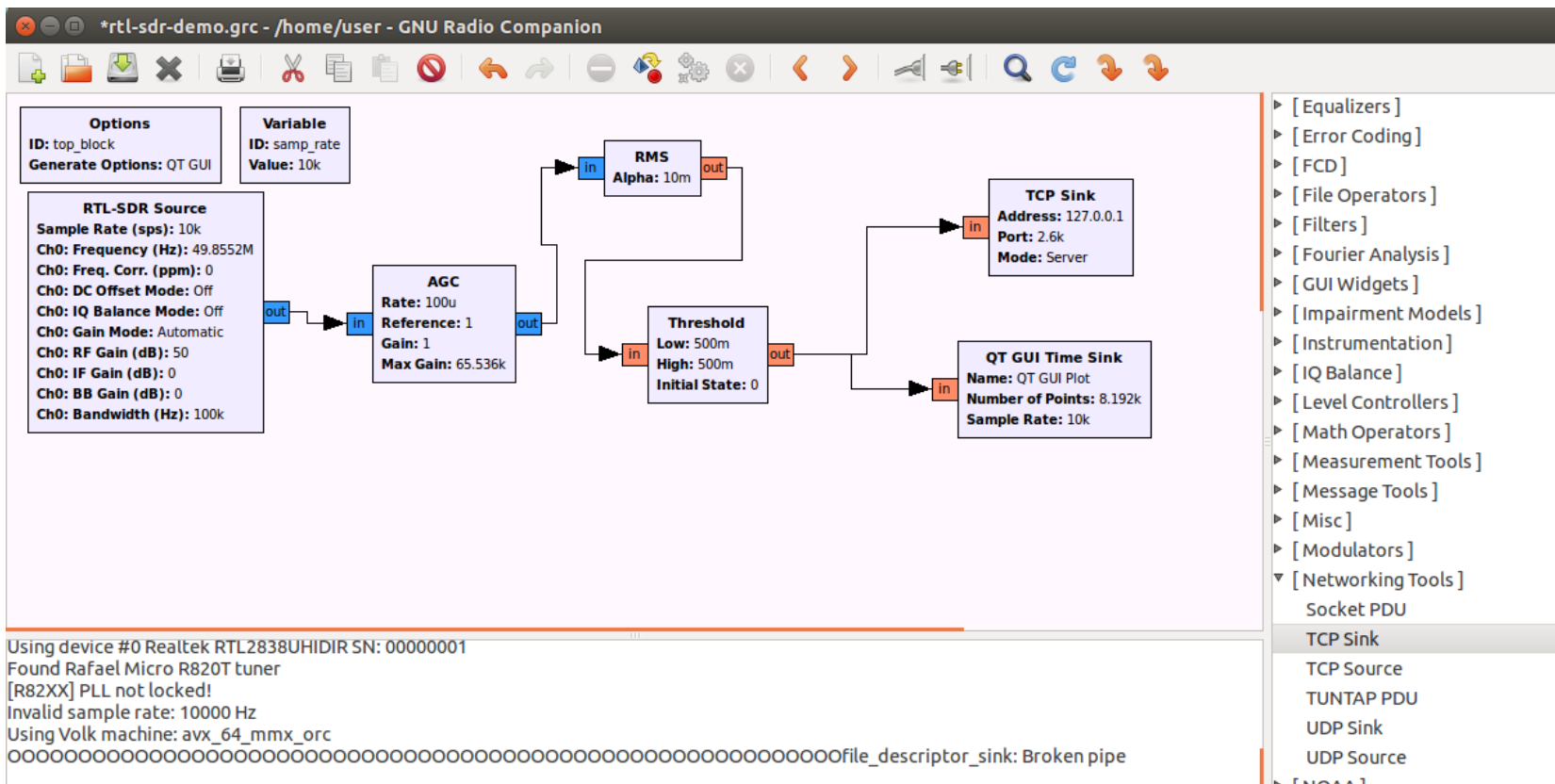
- Plotted data -- hard to see in real time
- Used GNURadio to write samples to a file
- Discovered the signal is pulsed CW
 - Also called on-off keying (OOK) or binary keying
- Simple signal – easy to generate with cheap electronics
- Single frequency – avoids intermod products on cheap amplifier



Signal Decoder

- Demodulation is pretty simple: the goal is to get the envelope of the function and then decode the bit sequence
- Process: filter on the center frequency, take the root mean squared of the signal
- Signal can only take on two values: on or off
- Use thresholding to convert to on-off values
- How to decode the bit sequence?
 - Captured all commands to data files
 - Wrote a C++ program to analyze the data

Block Diagram of Receiver



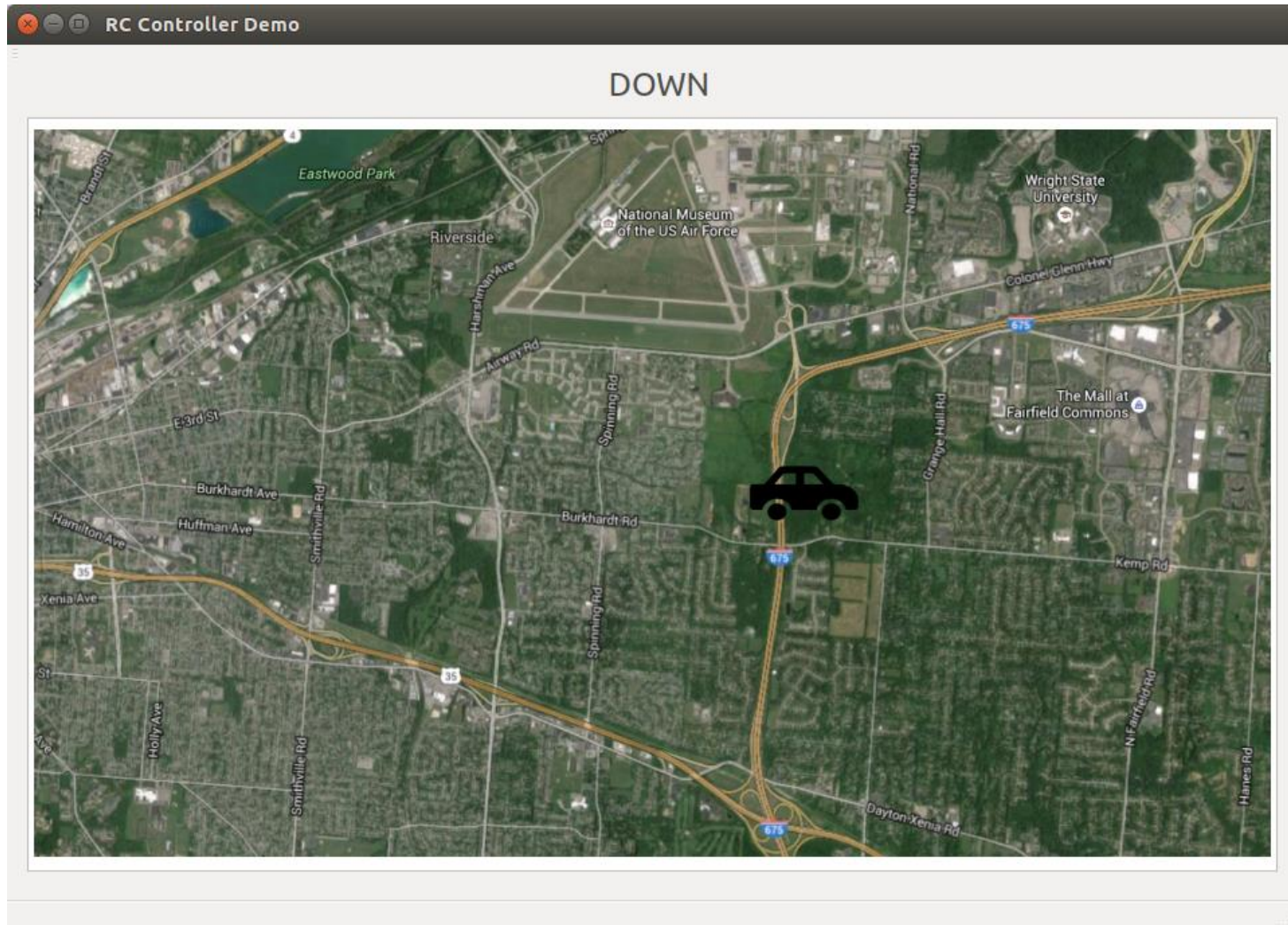
Bit Sequence

- Controller sends two types of pulses: short pulse (3 ms) and long pulse (8.25 ms)
- Used a 5 ms cutoff to separate short and long pulses
- Command always starts with four long pulses followed by a series of short pulses:
 - UP 10 pulses DOWN 40 pulses
 - LEFT 58 pulses RIGHT 64 pulses
 - DOWN + LEFT 52 pulses UP + LEFT 28 pulses
 - DOWN + RIGHT 46 pulses UP + RIGHT 34 pulses

Bit Decoder

- GNURadio feeds output over a local TCP socket to a custom Qt application
- Application uses a simple state machine to determine command:
 - Short pulses increment a counter
 - Long pulses reset the counter, execute commands
- When command is decoded, car icon is moved around on the screen
- Qt signals and slots approach makes it easy to buffer up TCP data and process periodically

Qt Software Demonstration



Thank You!

- Questions?
- Next Meetup: April 4th
 - An Introduction to GNURadio
- Feedback on what you'd like to see from the group:
 - Email bhart@pretalen.com
- Invite your friends. We are trying to grow this group over time.
- Any interest in online streaming of meetings?
- Presentations are available on the Meetup site:
 - <http://www.meetup.com/Dayton-Area-Software-Defined-Radio-Meetup/files/>
- Software demonstration files are available on GitHub:
 - <https://github.com/brucehart/dayton-sdr>