SigDigger

Blind signal analysis made easy

Introduction, examples, design details and seeking collaboration.

Gonzalo J. Carracedo

% whoami

- Gonzalo José Carracedo Carballal

BatchDrake at gmail.com

twitter.com/BatchDrake

github.com/BatchDrake

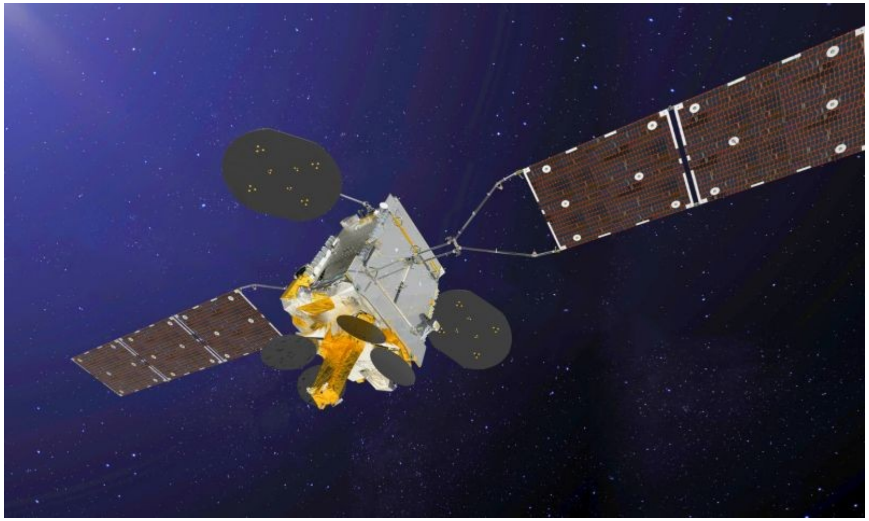
https://actinid.org

But what is SigDigger exactly?

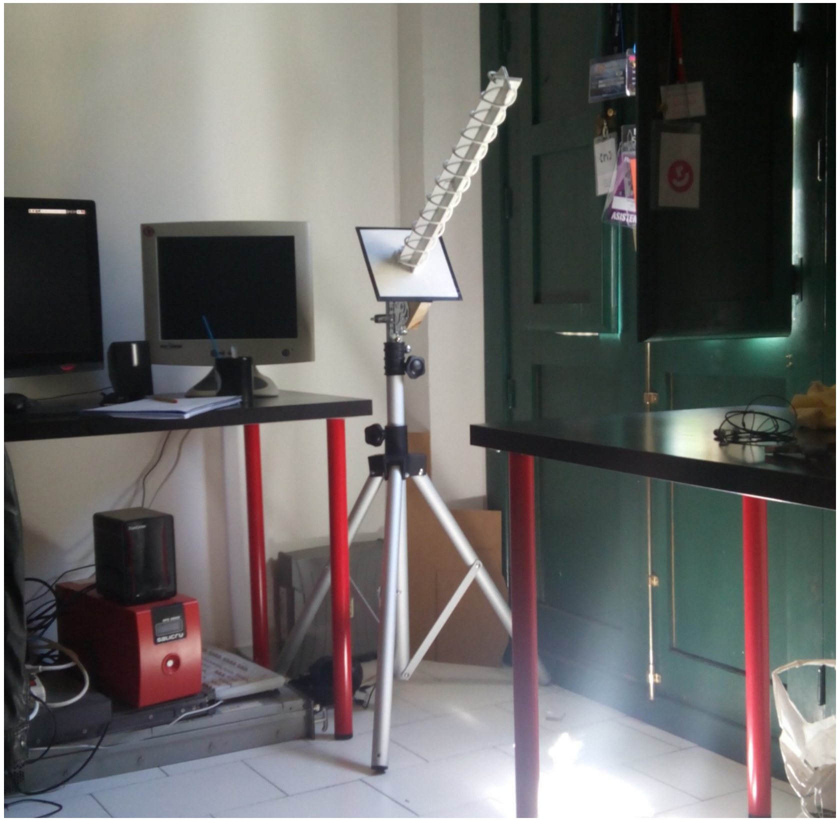
- SigDigger is a free (as in freedom) and graphical signal **analyzer**.
- You mean, like, another one? Gqrx, CubicSDR, URH, SDR#, baudline, HDSDR...
  - Well, **yes**, but simpler.
  - Main use case: reverse engineering of radio signals.
  - Continuous evolution from a pet project of mine 6 years ago.
  - A bit of history is necessary

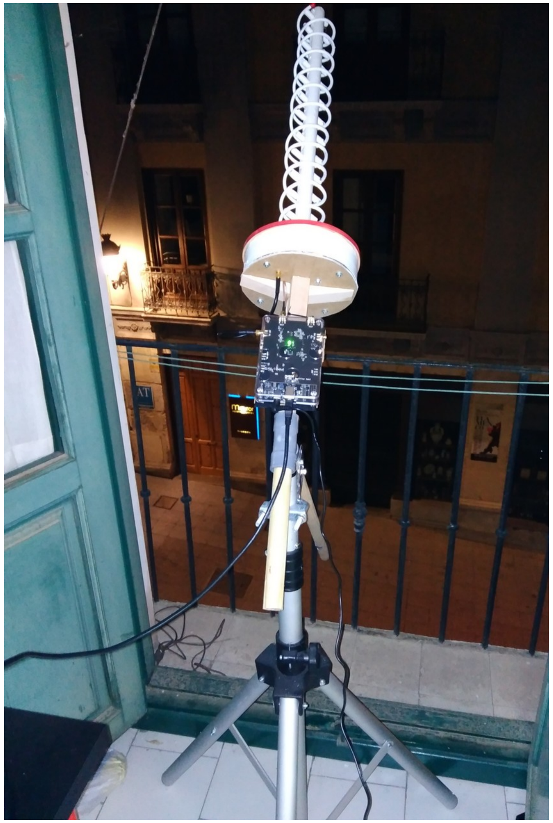
The boring summer of 2016

- Very basic knowledge about radio propagation and data acquisition.
- I have a BladeRF and some spare time
- How about receiving satellite signals for fun?
  - Inmarsat satellites in L-Band (around 1500 MHz, RHCP)
  - Classic Aero ACARS messages using JAERO (https://jontio.zapto.org/hda1/jaero.html)

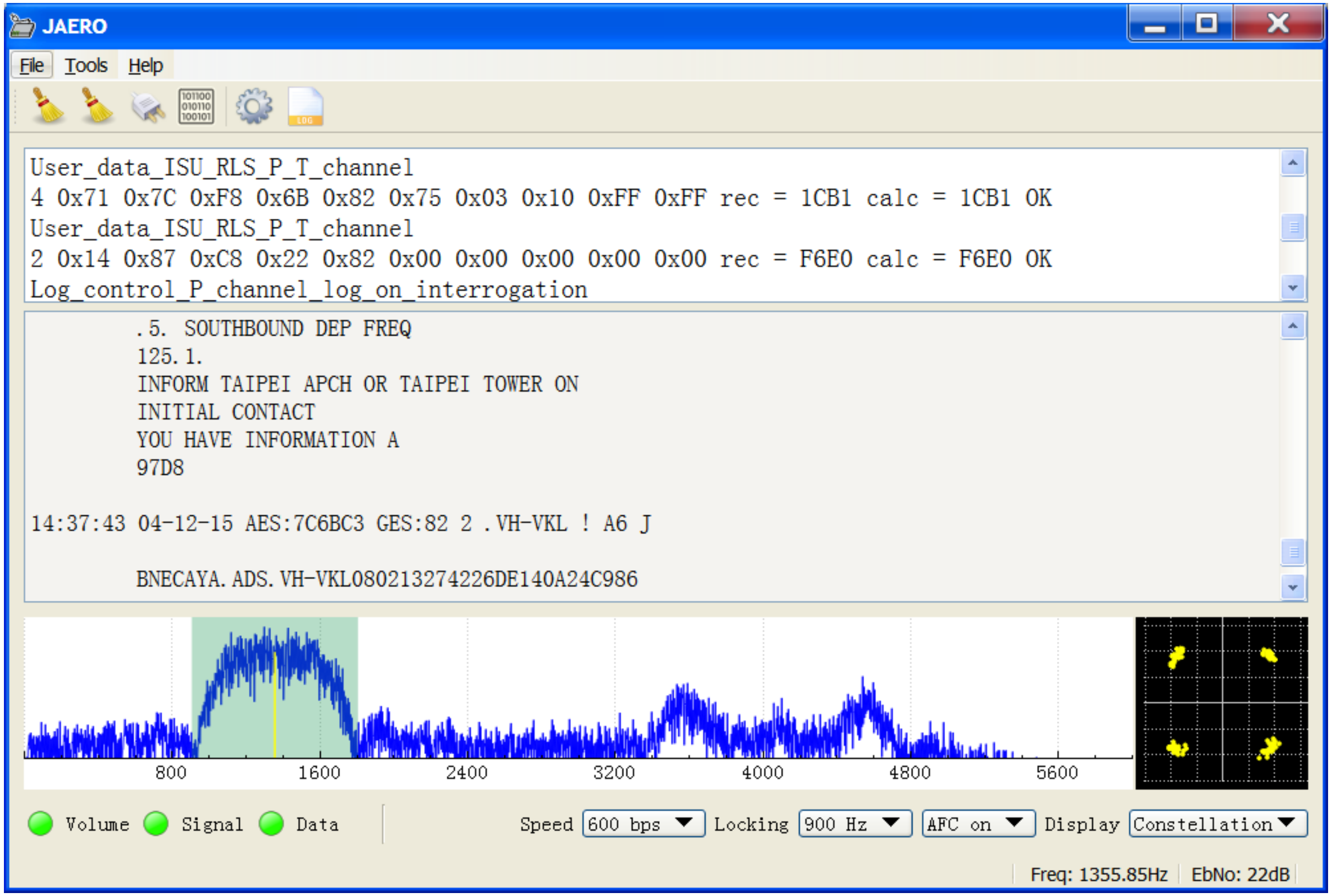


DIY antennas!





JAERO



```
User_data_ISU_RLS_P_T_channel
4 0x71 0x7C 0xF8 0x6B 0x82 0x75 0x03 0x10 0xFF 0xFF rec = 1CB1 calc = 1CB1 OK
User_data_ISU_RLS_P_T_channel
2 0x14 0x87 0xC8 0x22 0x82 0x00 0x00 0x00 0x00 0x00 rec = F6E0 calc = F6E0 OK
Log_control_P_channel_log_on_interrogation
```

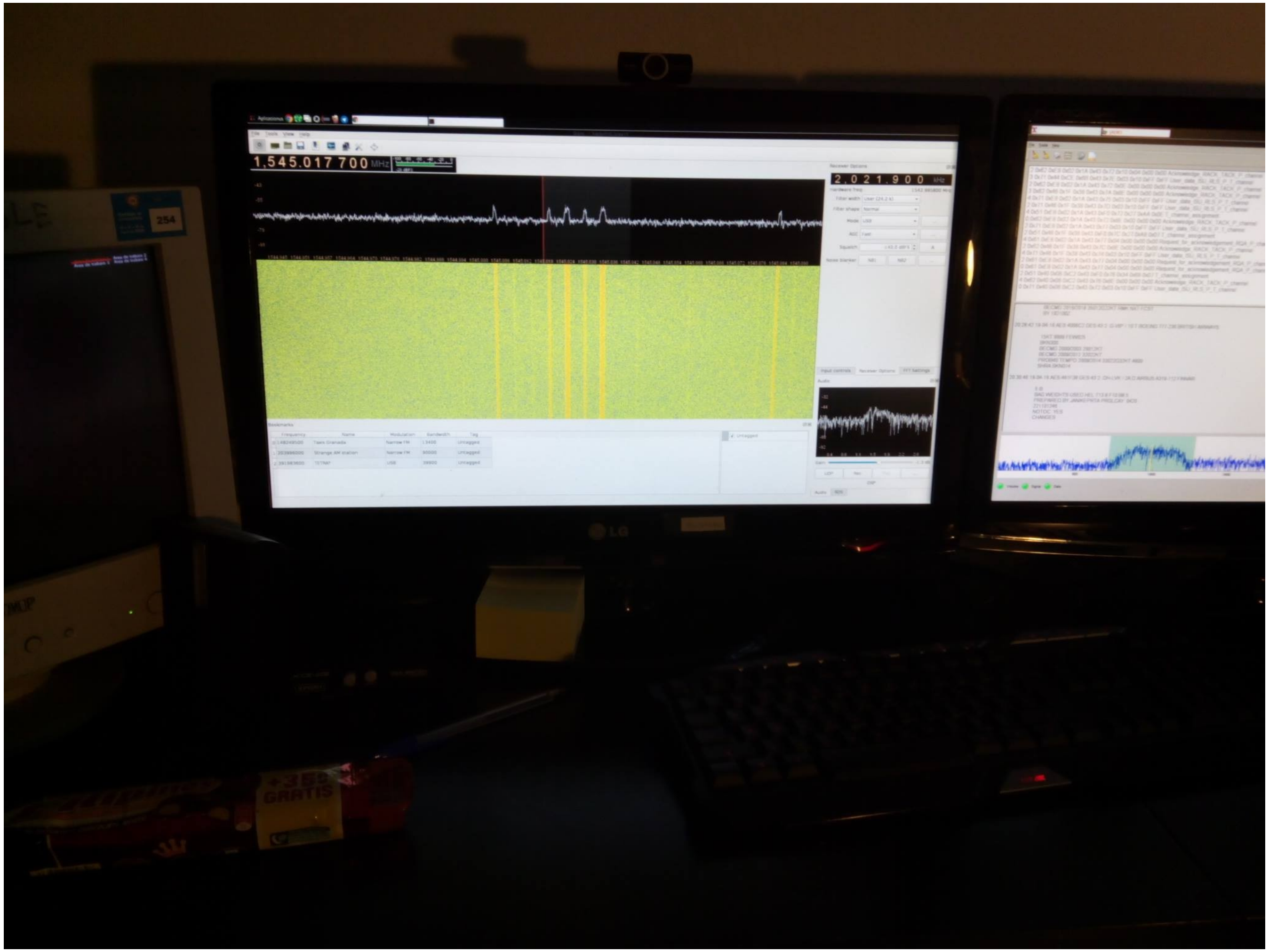
```
        .5. SOUTHBOUND DEP FREQ
        125.1.
        INFORM TAIPEI APCH OR TAIPEI TOWER ON
        INITIAL CONTACT
        YOU HAVE INFORMATION A
        97D8

14:37:43 04-12-15 AES:7C6BC3 GES:82 2 .VH-VKL ! A6 J

        BNECAYA.ADS.VH-VKL080213274226DE140A24C986
```

● Volume ● Signal ● Data

Speed 600 bps ▼ Locking 900 Hz ▼ AFC on ▼ Display Constellation ▼

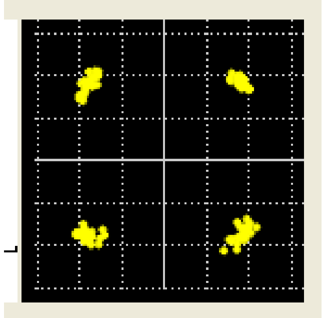
Freq: 1355.85Hz EbNo: 22dB

Pluggin everything: Gqrx



And now what?

- That was fun, I was able to demodulate it and receive signals. Yoohoo I“m a hacker
- Okay, that was it?
  - Many other signals in adjacent frequencies with different frequency envelopes
  - Coming from different satellites (pointing-dependant)
  - What is **this**?



The challenge: blind demodulation

- What if I knew **nothing** about the signal? Would I be able to demodulate it?
- And even if I could demodulate it, would I be able to decode it?
- And even if I could decode it, could I extract data from the decoded bits?
- Welcome to the fantastic world of **AMC**!
  - References: **Balint Seeber**, **Daniel Estévez (EA4GPZ)**
  - Rigurous moment-based automatic modulation classification (**Darek Kawamoto**): https://www.youtube.com/watch?v=lqXSxhn_A2o

The goals

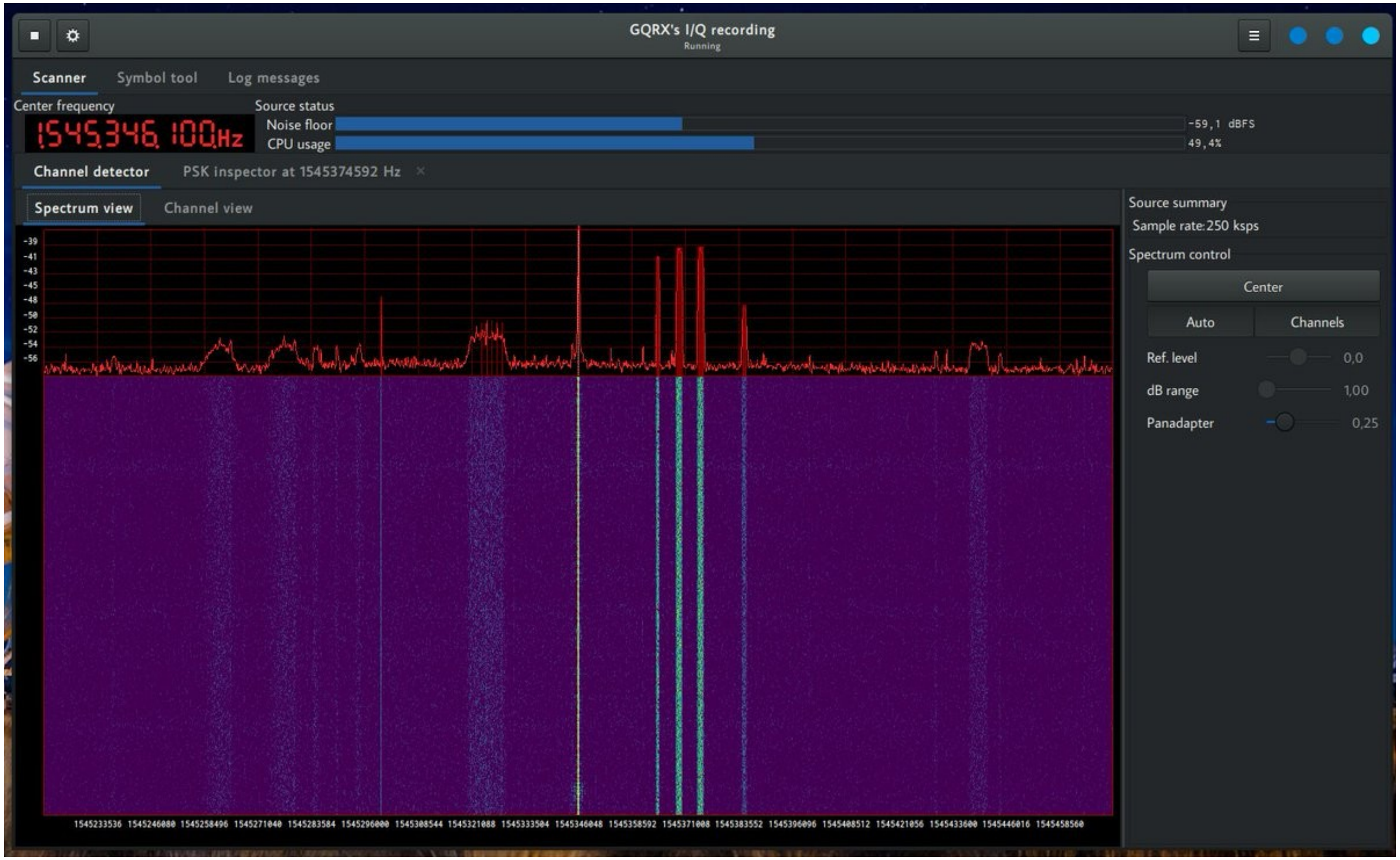
- Extremely basic knowledge of DSP in general. Need to acquire skills.
  - Way to go: code your own DSP library in C and learn the hard way. **Sigutils.**
- Small application: **suscan** (from **Sigutils Scanner**):
  - Curses (this was a mistake)
  - Minimal human intervention
  - Automatic channel detection
  - Pseudocontinuous-based SNR detection
  - AMC strategies (2^n-th power, cyclostationary analysis)
  - Integrated PSK demodulator
  - Direct interaction with libbladerf, libhackrf, librtlsdr…

Suscan in action

Reconsidering the design

- Ncurses was a **mistake**.
  - Pre-SOLID era library. Unmaintainable.
  - Add a more practical GTK+3 interface.
- Suscan internal API was still a hack, needed to redesign it.
  - GR-like pluggable blocks
  - Message passing for thread communication
  - Client-server model
- Add support for raw I/Q captures

The new Suscan



GQRX's I/Q recording
Running

Center frequency: 1545346 100Hz

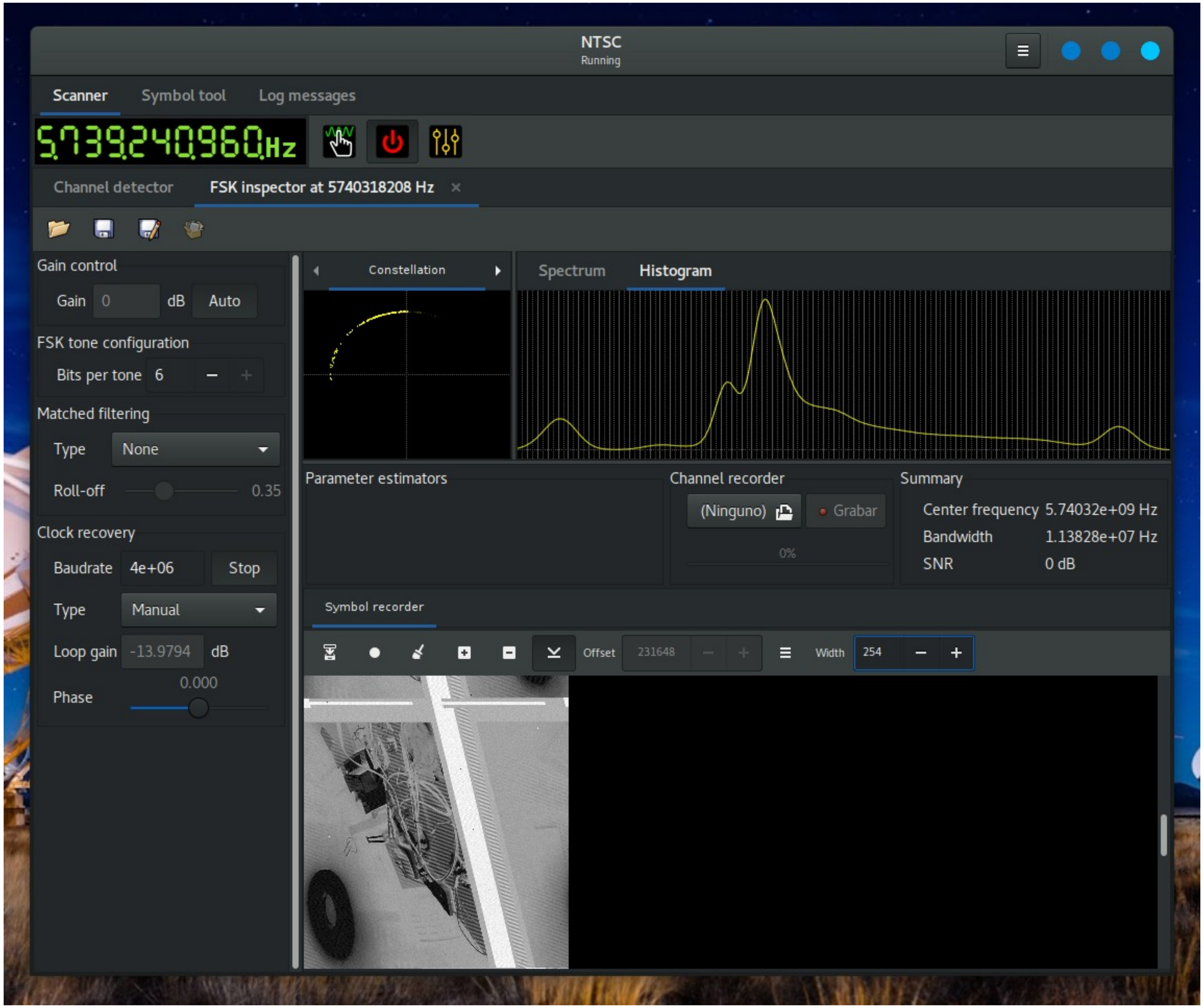
Source status
Noise floor -59,1 dBFS
CPU usage 49,4%

PSK inspector at 1545374592 Hz

Source summary
Sample rate: 250 ksps

Spectrum control
Ref. level 0,0
dB range 1,00
Panadapter 0,25

The new Suscan



5739240960Hz

Gain control

Gain 0 dB Auto

FSK tone configuration

Bits per tone 6

Matched filtering

Type None

Roll-off 0.35

Clock recovery

Baudrate 4e+06 Stop

Type Manual

Loop gain -13.9794 dB

Phase 0.000

Parameter estimators

Channel recorder

(Ninguno) Grabar

0%

Summary

Center frequency	5.74032e+09 Hz
Bandwidth	1.13828e+07 Hz
SNR	0 dB

Offset 231648 Width 254

This CPU is on fire! 🔥🔥🔥

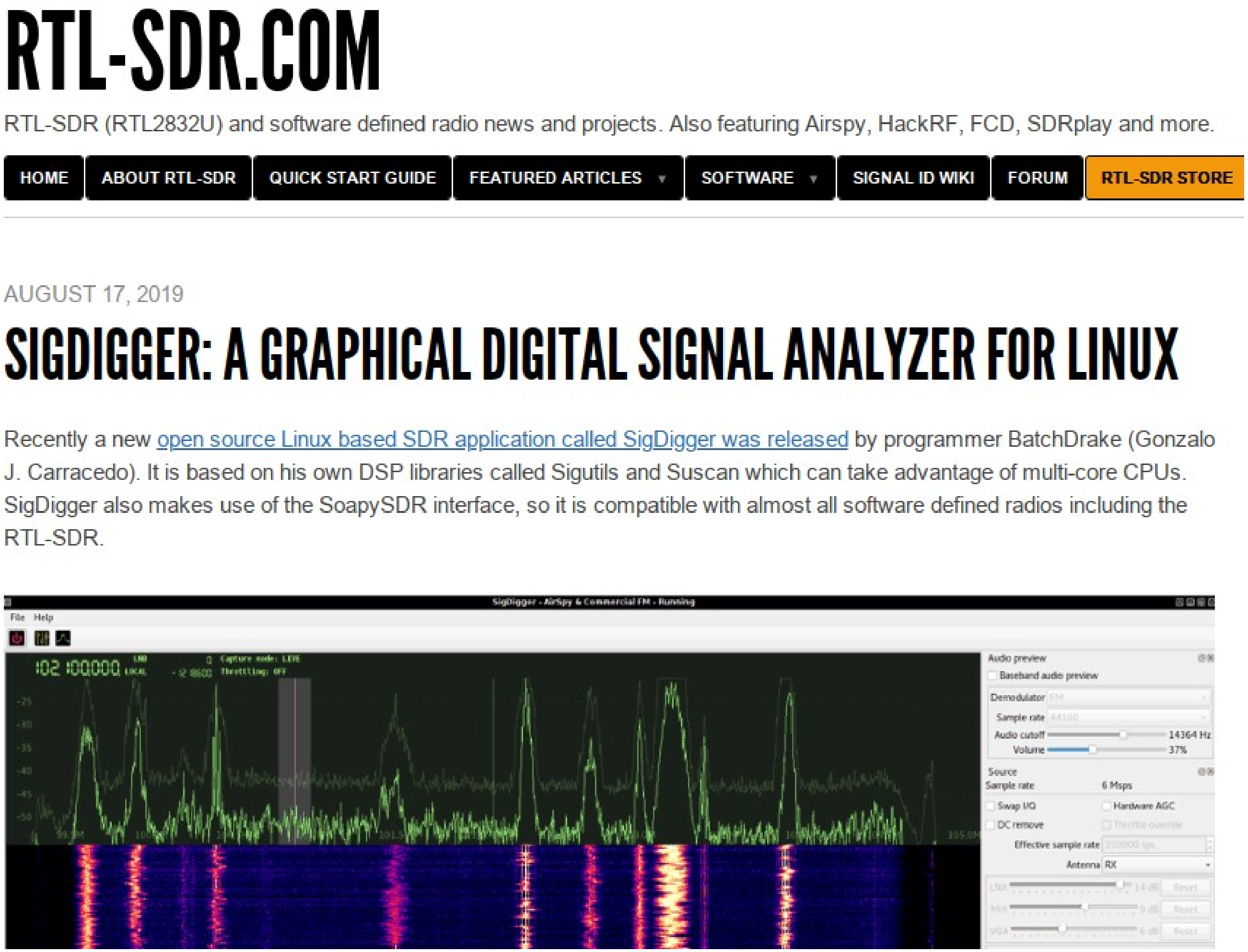
- The block-based flowgraph was poorly implemented and it was also a mess
  - Concurrency overhead
  - Replaced by the worker approach (more on this later)
- FIR-based channelizer!
  - Real-time filtering at device rate! Ouch!
  - Use FFT channelizer.
- **GTK+3 is another mistake**
  - Used to like it because of its native C interface.
  - Otherwise slooow. Cairo is one of the slowest graphical APIs I ever dealt with.
  - Extremely difficult to bypass it and barely maintainable.
  - Too much boilerplate, even with GtkBuilder.
- Most of the Suscan"s core functionality can be detached from the GUI at this point.

https://commons.wikimedia.org/wiki/File:Texture_Fire.jpg

The great refactor

- Ad-hoc SDR compatibility code replaced by **SoapySDR**.
  - Automatic compatibility with most SDRs in the market.
- **Removed GTK+3** support and all references to GUI.
  - Now suscan is actually a real-time signal analyzer library (**libsuscan**), providing a big server class called `suscan_analyzer_t`
  - Client-independent API (6 dec 2018)
- **Start to work on the Qt5 frontend: 5 jul 2019**
  - C++. Yikes. But damn, Qt5 is so fast
  - Based on Gqrx" spectrum widget directly.
- **First beta release of SigDigger in 16 aug 2019**

SigDigger in rtl-sdr.com!



RTL-SDR.COM

RTL-SDR (RTL2832U) and software defined radio news and projects. Also featuring Airspy, HackRF, FCD, SDRplay and more.

AUGUST 17, 2019

SIGDIGGER: A GRAPHICAL DIGITAL SIGNAL ANALYZER FOR LINUX

Recently a new open source Linux based SDR application called SigDigger was released by programmer BatchDrake (Gonzalo J. Carracedo). It is based on his own DSP libraries called Sigutils and Suscan which can take advantage of multi-core CPUs. SigDigger also makes use of the SoapySDR interface, so it is compatible with almost all software defined radios including the RTL-SDR.

SigDigger - AirSpy & Commercial FM - Running

Audio preview

□ Baseband audio preview

Demodulator FM

Sample rate 44100

Audio cutoff 14364 Hz

Volume 37%

Source

Sample rate 6 Msps

□ Swap I/Q □ Hardware AGC

□ DC remove □ Throttle override

Effective sample rate 250000 sps

Antenna RX

LNA 14 dB Reset

MIX 0 dB Reset

VGA 6 dB Reset

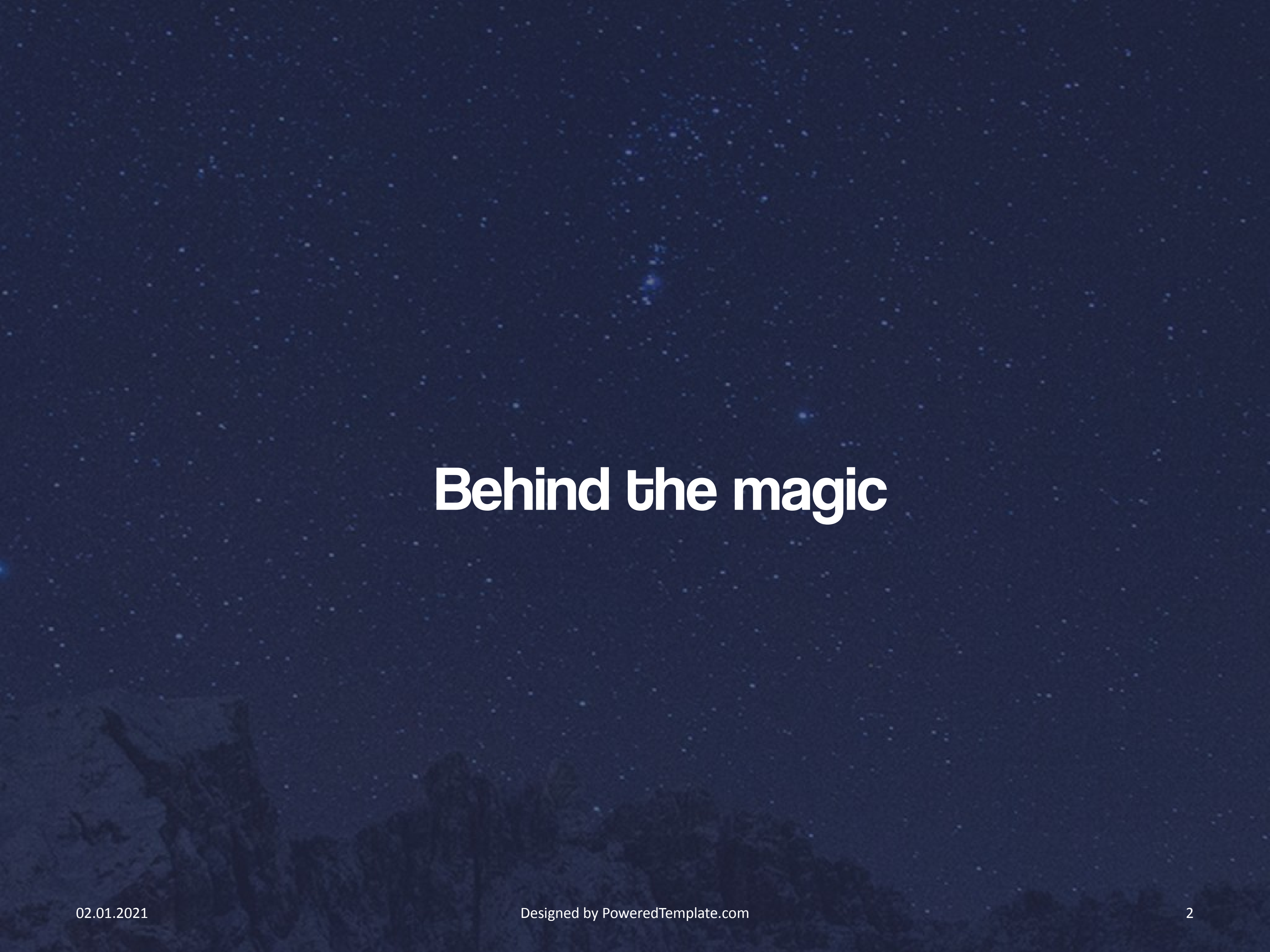
What is SigDigger now?

- SigDigger is a free (as in freedom) and graphical signal **analyzer**.
- It is an **analyzer** because it is supposed to let you **analyze** individual frequency-multiplexed signals.
  - Capture small bursts and inspect the wave
  - Demodulate signals in real time (PSK / ASK / FSK)
  - Watch generic analog TV (presets for PAL and NTSC)
  - Previous AMC features (cyclostationary...)
  - Listen to AM / FM / SSB signals
  - Bookmarks & bandplans
  - Panoramic spectrum

Some performance figures

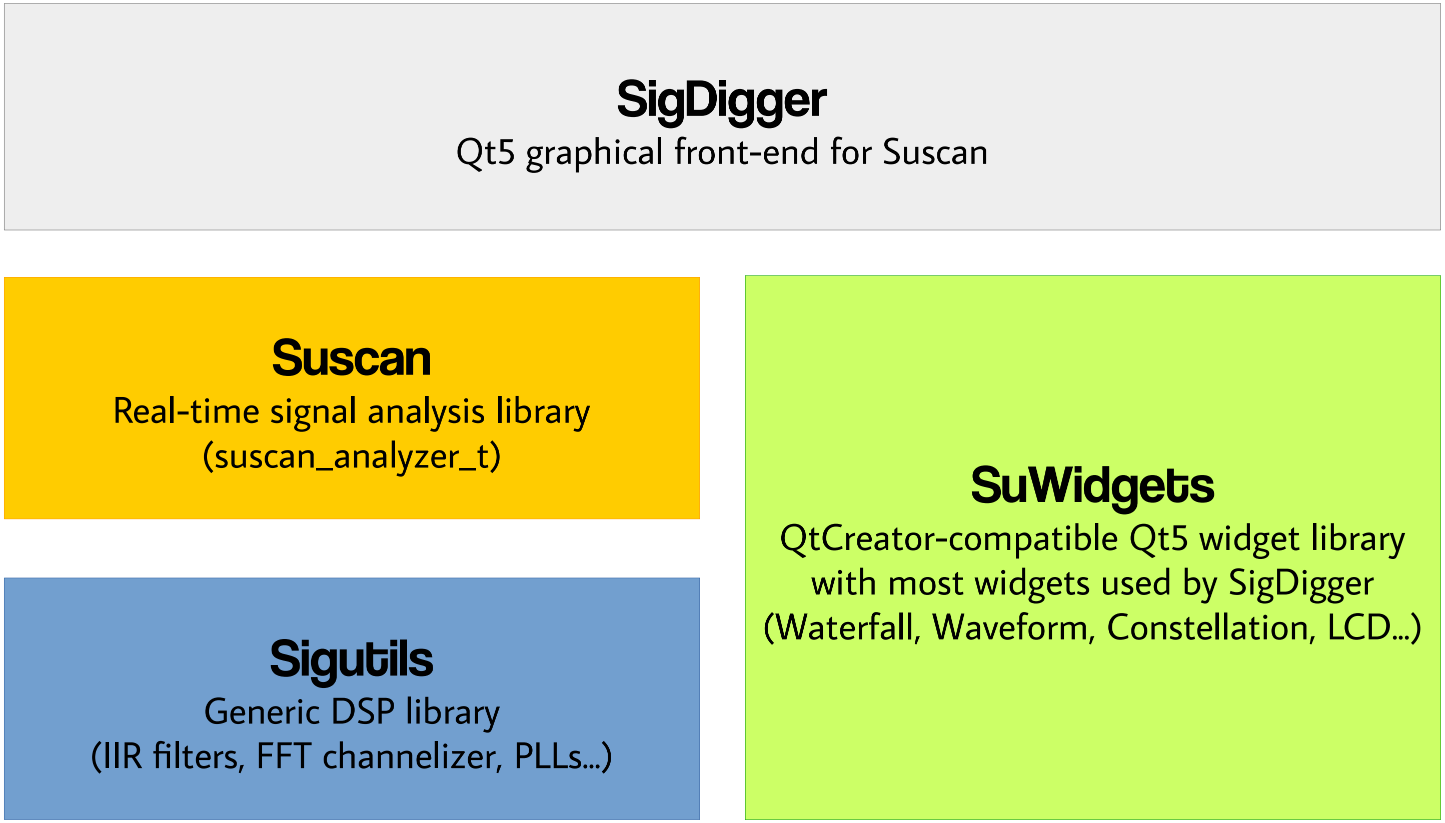
- Test computer:
  - Intel(R) Core(TM) i5-6200U CPU @ 2.30GHz
  - 2 cores, 4 threads
- CPU usage w.r.t. Gqrx, same signal source
  - Around 20% less, equivalent configurations
  - CubicSDR is still less CPU intensive
- Processing speeds:
  - Spectrum only, 16K FFT bins, 60 fps: **108 Msps**
  - Spectrum only, 64K FFT bins, 60 fps: **97±5 Msps (fluctuating)**
  - FM demodulator, 333 kHz BW: **17 Msps**
  - Analog TV demod: **5.6 Msps**

Demo time



Behind the magic

The architecture



SigDigger
Qt5 graphical front-end for Suscan

Suscan
Real-time signal analysis library
(suscan_analyzer_t)

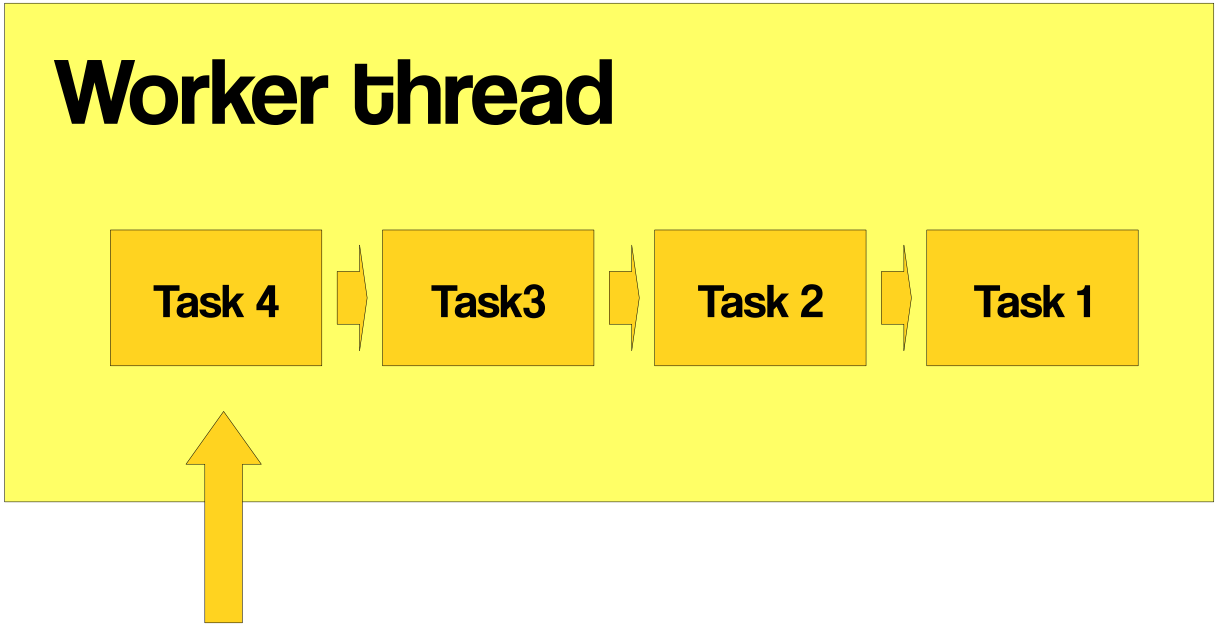
Sigutils
Generic DSP library
(IIR filters, FFT channelizer, PLLs…)

SuWidgets
QtCreator-compatible Qt5 widget library
with most widgets used by SigDigger
(Waterfall, Waveform, Constellation, LCD…)

How come it is so fast?

- Three keys:
  - FFT channelization via FFTW3
  - Worker thread approach distributed in different cores
  - No blocks, just a barrier after all inspector workers have finished with their batches
- Other important aspects:
  - Qt5 is incredibly fast at drawing things!
  - Important fraction of the analyzer API async and message-based.

Workers are just callback queues



Worker thread

Task 4

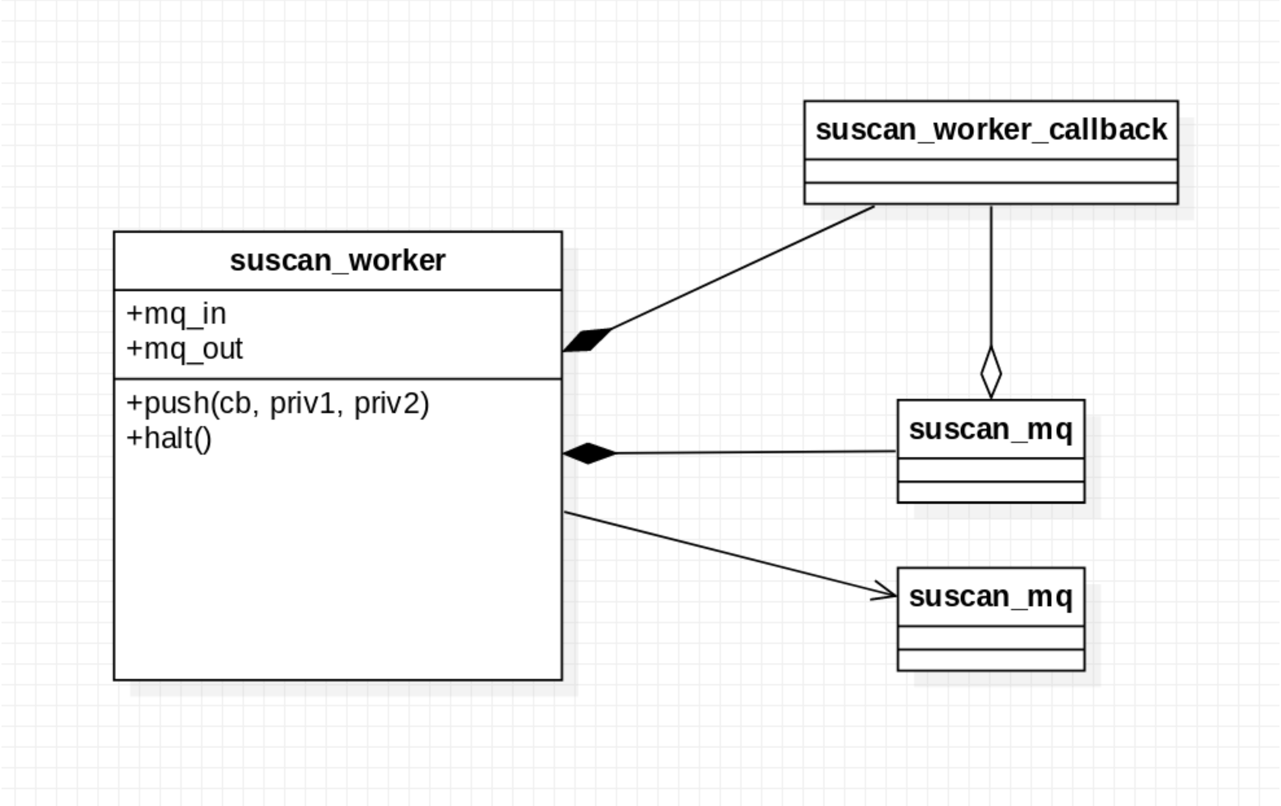
Task3

Task 2

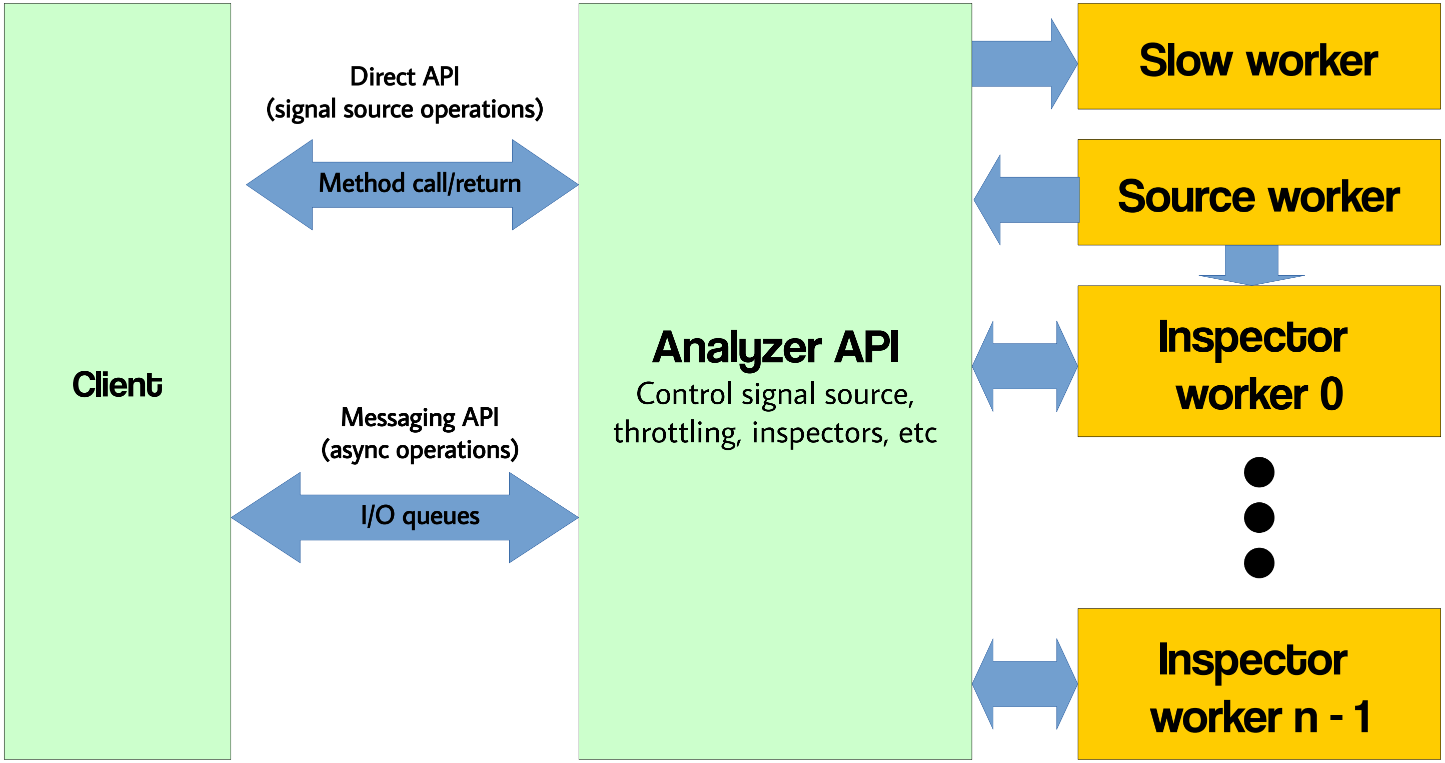
Task 1

push()

Workers are just callback queues



Suscan"s Analyzer architecture



Client

Direct API
(signal source operations)

Method call/return

Messaging API
(async operations)

I/O queues

Analyzer API
Control signal source,
throttling, inspectors, etc

Slow worker

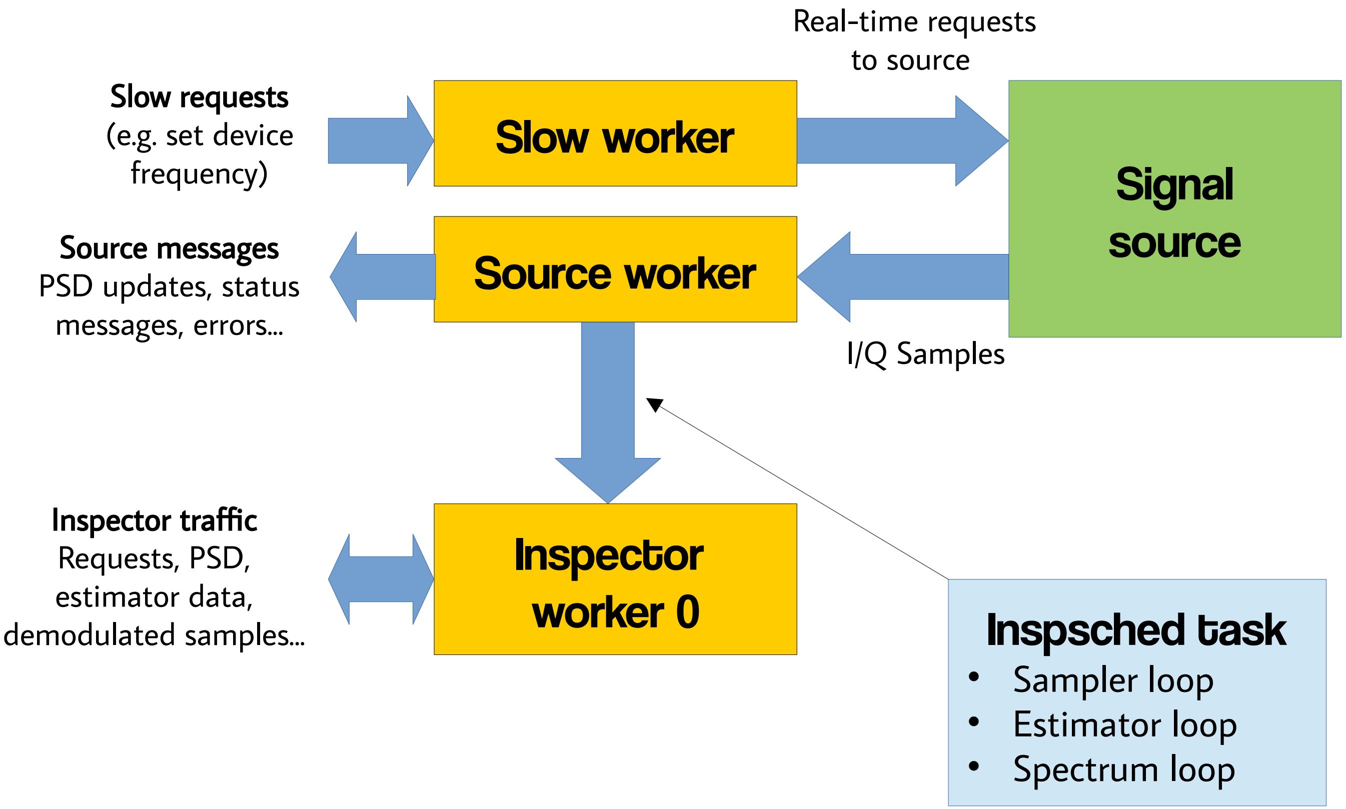
Source worker

Inspector worker 0

●
●
●

Inspector worker n - 1

Workers in detail



Slow requests
(e.g. set device
frequency)

Slow worker

Real-time requests
to source

Signal
source

Source messages
PSD updates, status
messages, errors…

Source worker

I/Q Samples

Inspector traffic
Requests, PSD,
estimator data,
demodulated samples…

Inspector
worker 0

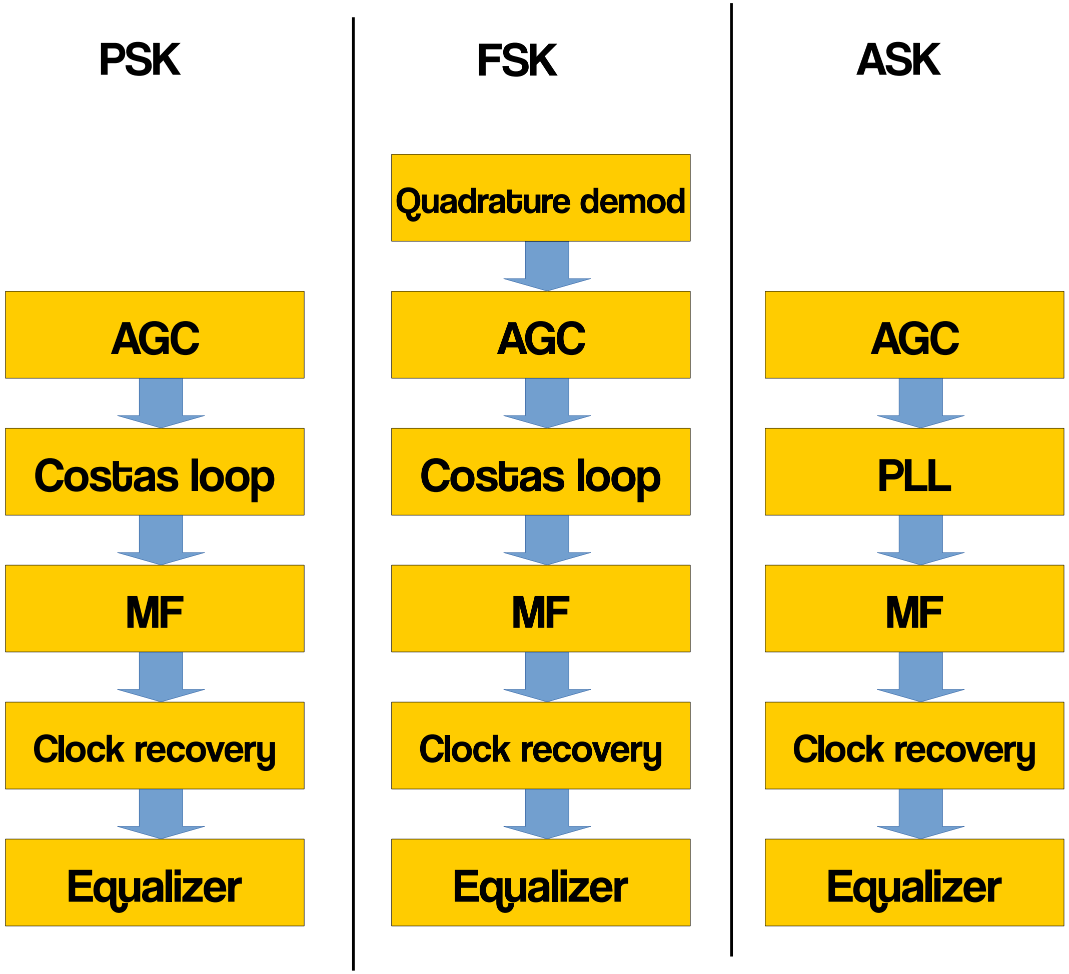
Inspsched task

- Sampler loop
- Estimator loop
- Spectrum loop

The channel inspector

- Representation of a channel being analyzed in real-time
- Actually, it is a real-time configurable demodulator
- Several specializations
  - The PSK inspector
  - The FSK inspector
  - The ASK inspector
  - The RAW inspector
  - The audio inspector
- Processes batches of samples produced by the source worker"s FFT channelizer by its **loops**

Sampler loops



PSK

AGC → Costas loop → MF → Clock recovery → Equalizer

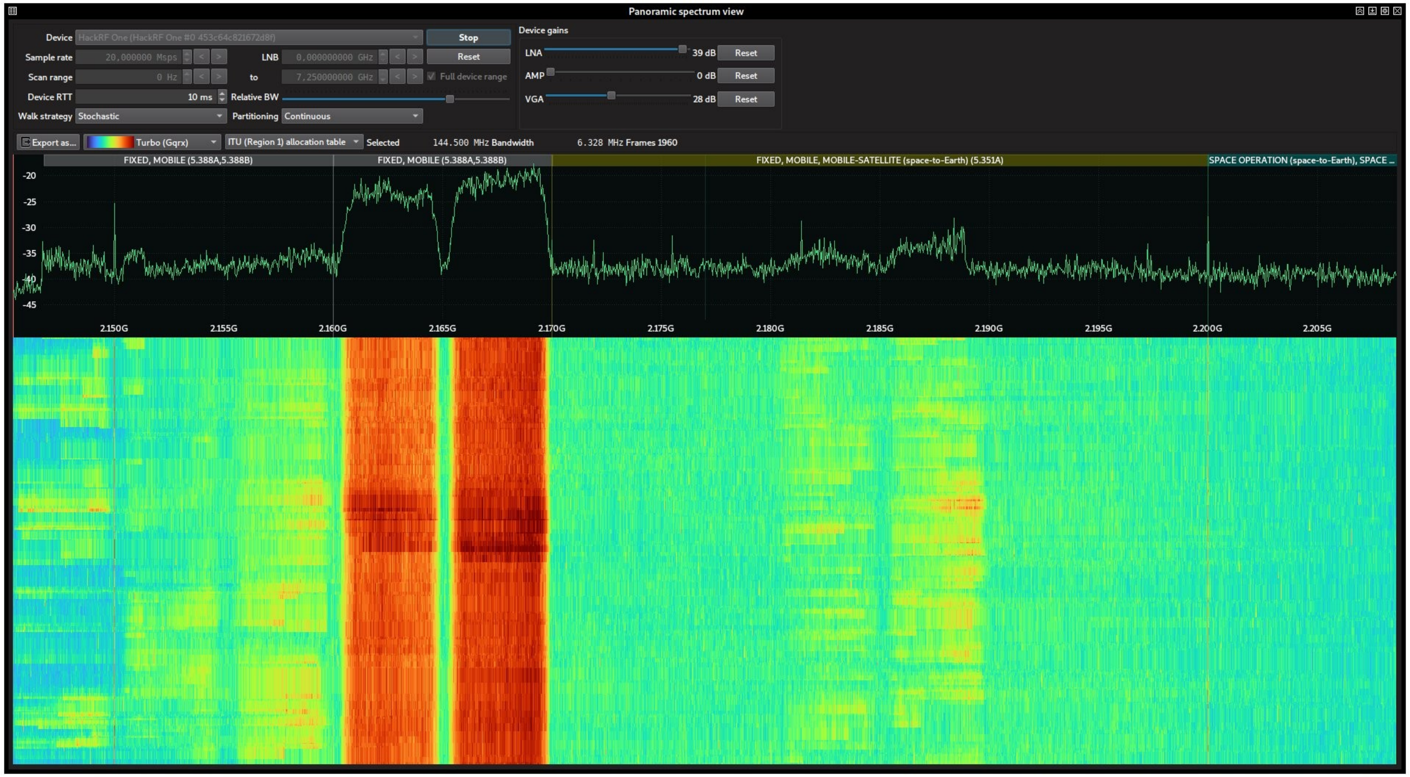
FSK

Quadrature demod → AGC → Costas loop → MF → Clock recovery → Equalizer

ASK

AGC → PLL → MF → Clock recovery → Equalizer

Panoramic spectrum



Panoramic spectrum view

Device HackRF One (HackRF One #0 453c64c821672d8f)

Sample rate 20,000000 Msps

LNB 0,000000000 GHz

Scan range 0 Hz to 7,250000000 GHz

☑ Full device range

Device RTT 10 ms

Relative BW

Walk strategy Stochastic

Partitioning Continuous

Device gains

LNA 39 dB

AMP 0 dB

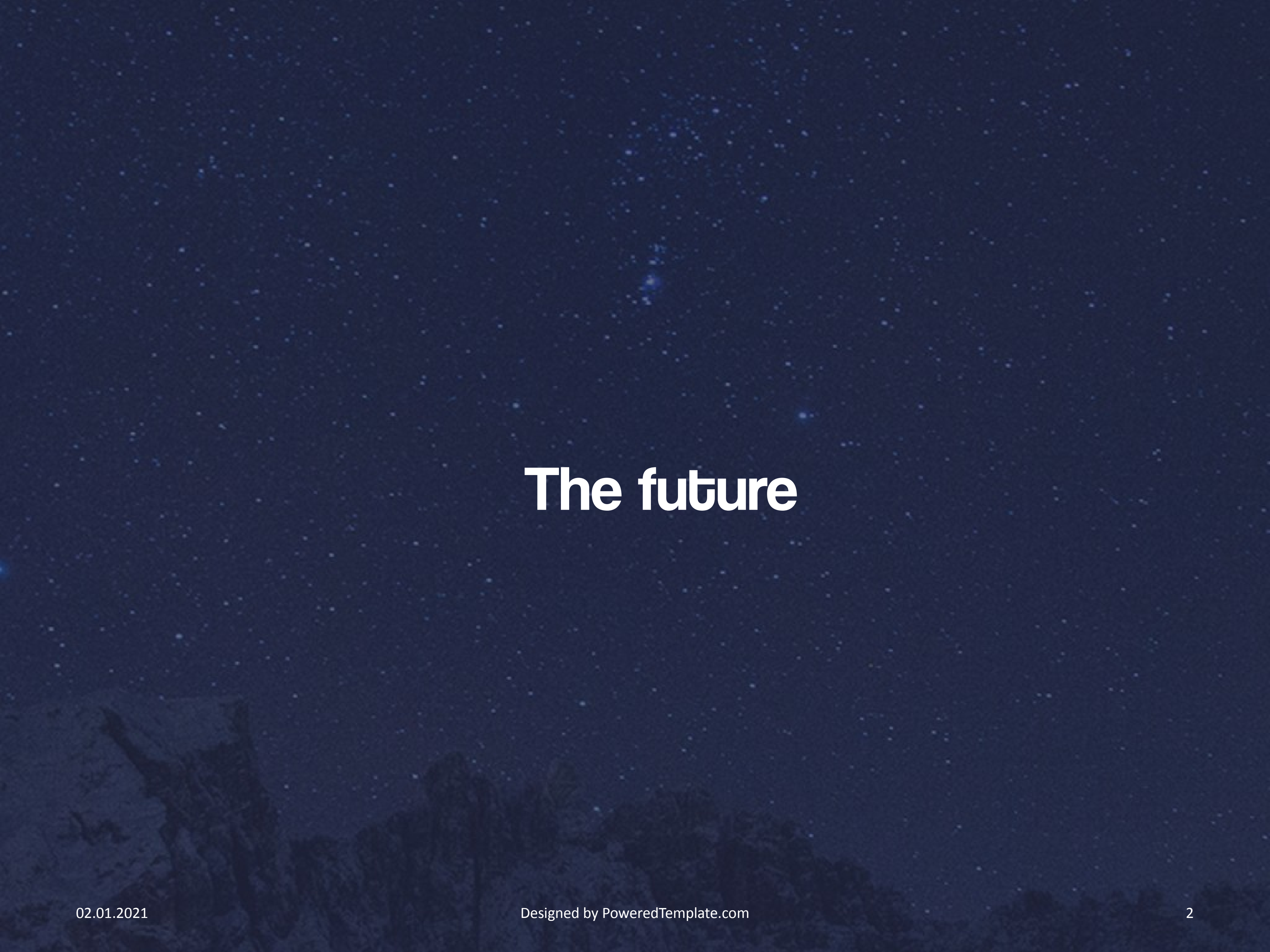
VGA 28 dB

Turbo (Gqrx)

ITU (Region 1) allocation table

Selected 144.500 MHz Bandwidth 6.328 MHz Frames 1960

Stop
Reset
Reset
Reset
Reset
Export as...
FIXED, MOBILE (5.388A,5.388B)
FIXED, MOBILE (5.388A,5.388B)
FIXED, MOBILE, MOBILE-SATELLITE (space-to-Earth) (5.351A)
SPACE OPERATION (space-to-Earth), SPACE ...
-20
-25
-30
-35
-40
-45
2.150G
2.155G
2.160G
2.165G
2.170G
2.175G
2.180G
2.185G
2.190G
2.195G
2.200G
2.205G



The future

List of open fronts

- RPC-like remote analyzers (CBOR based)
- **Remove barriers.** Use buffer pools instead.
- Embed SoapySDR modules in the macOS bundle.
- Deeper refactor of the analyzer
- Alternative interfaces (web interface, mobile?)
- TLE-based Doppler correction for satellites / spacecrafts
- Digital decoders (Blind viterbi decoder, symbol tagger, differential decoder, etc). **Hobbits** integration?
  - https://github.com/Mahlet-Inc/hobbits
- Pluggable inspectors (APT requires this, also for FM. SDR#-like slicing?)
- Device-specific settings and hacks (Bias Tee)
- PlutoSDR off-loading (spectrum, channelization…)

Want to help? :)

Thanks!

Especially to Jeff Sipek, Aaron Foster, Mehdi Asgari, Shiki Owo, Andrés Perez
and all the people that helped me out with SigDigger one way or another

Remember you have to attribute!*

- Creating content takes a lot of time and effort, but all we need from you is only an attribution link.

- In order to use the content or a part of it, you must attribute it to PoweredTemplate.com, so we will be able to continue creating new graphic resources every day.

- Insert the attribution line in the credits section of your presentation. If it’s not possible, place it wherever it's visible on a web page, close to where you’re using the resource.

- For example: This presentation has been designed using resources from PoweredTemplate.com

* This only applies if you downloaded this content as an unsubscribed (free) user.