UNIVERSITÄT BONN

Reverse Engineering von Funkprotokollen

Oder: Wie der Nachbar seine Steaks brät

Ben Swierzy

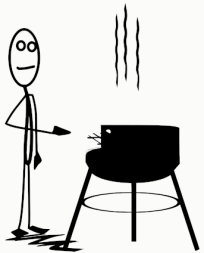
swierzy@uni-bonn.de

Universität Bonn | Institut für Informatik 4

CooleLeute.Live | Bonn | 6. Juli 2020

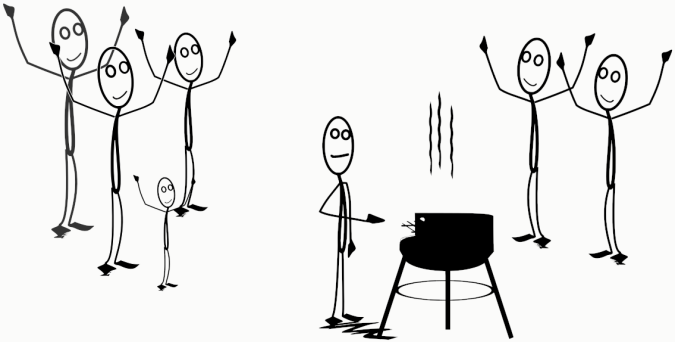
UNIVERSITÄT BONN

Motivation



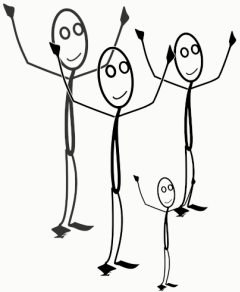
UNIVERSITÄT BONN

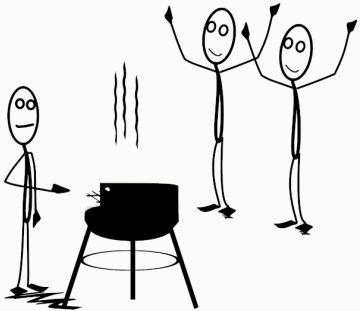
Motivation



UNIVERSITÄT BONN

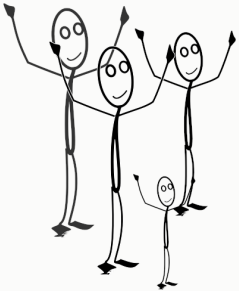
Motivation

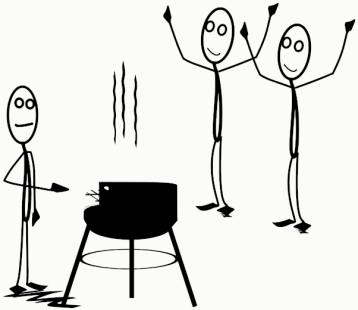


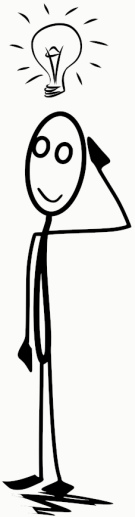


UNIVERSITÄT BONN

Motivation







UNIVERSITÄT BONN

Disclaimer

Es ist leicht die Inhalte dieses Vortrags für Straftaten zu verwenden.
Es liegt in eurer eigenen Verantwortung, das Wissen innerhalb der geltenden Gesetze einzusetzen.

Warnung:
Technischer Vortrag

UNIVERSITÄT BONN

Inhalte

GRUNDLAGEN

ANWENDUNG

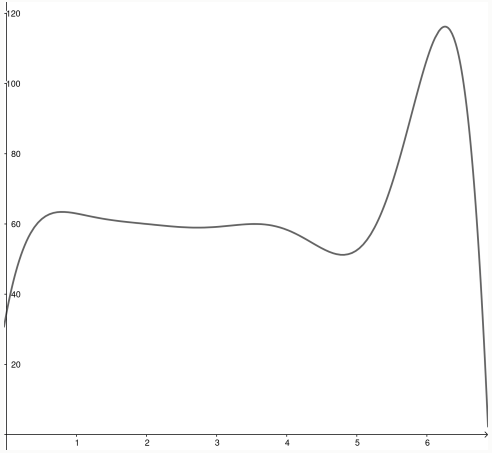
Grundlagen

Teil 1: Signale

UNIVERSITÄT BONN

Was ist ein Signal?

$$f : \mathbb{R} \to \mathbb{R}$$

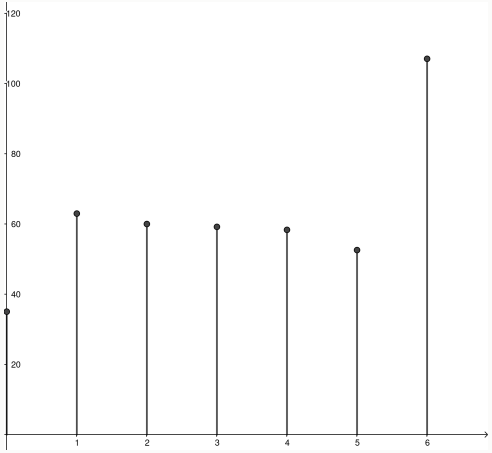


UNIVERSITÄT BONN

Analog oder Digital?

$f : \mathbb{R} \to \mathbb{R}$

$g : \mathbb{Z} \to \mathbb{R}$



UNIVERSITÄT BONN

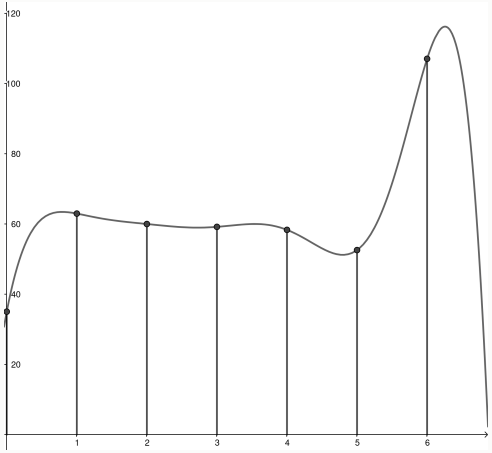
Analog und Digital

$f : \mathbb{R} \to \mathbb{R}$

$g : \mathbb{Z} \to \mathbb{R}$

Sample Rate $s = 1$ Hz

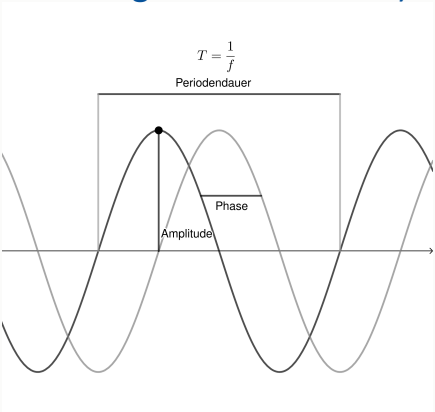
$g(k) = f(\frac{k}{s})$



UNIVERSITÄT BONN

Eigenschaften von Signalen

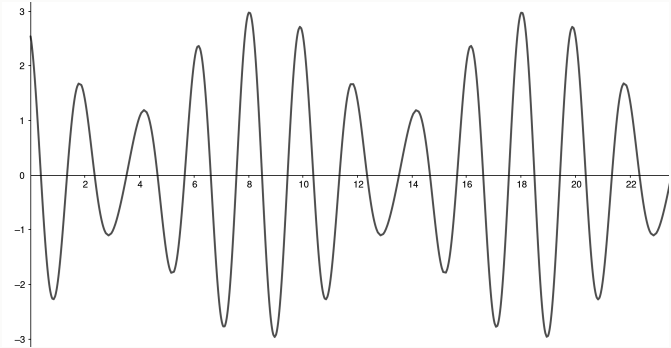
- Wellen (z.B. Schallwellen, elektromagnetische Wellen)
- $f(t) = A \cdot \cos(2\pi f t + \varphi)$
- Frequenz f
- Amplitude A
- Phase φ



UNIVERSITÄT BONN

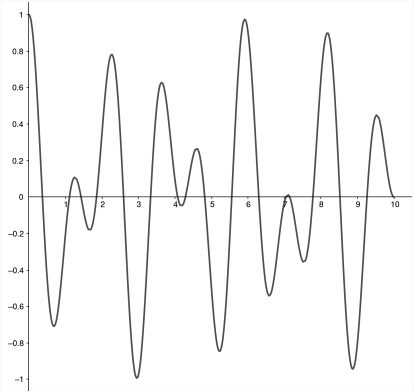
Eigenschaften von Signalen

- Überlagerung durch Addition
- Bandbreite eines Signals: $BW = f_{\text{max}} - f_{\text{min}}$

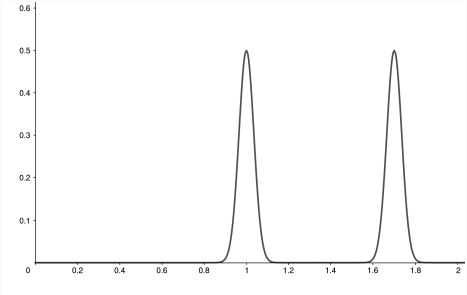


UNIVERSITÄT BONN

Fouriertransformation



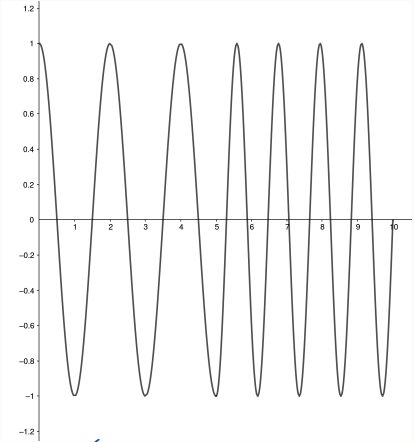
$$f(t) = \frac{\cos(2\pi t)+\cos(\frac{17}{10}\cdot 2\pi t)}{2}$$



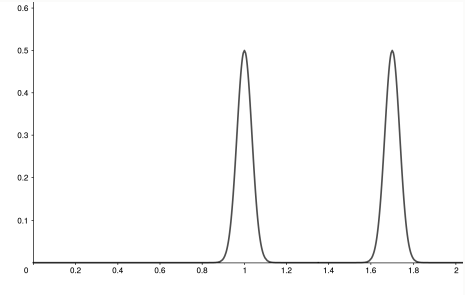
$\mathrm{FFT}(f)$

UNIVERSITÄT BONN

Fouriertransformation

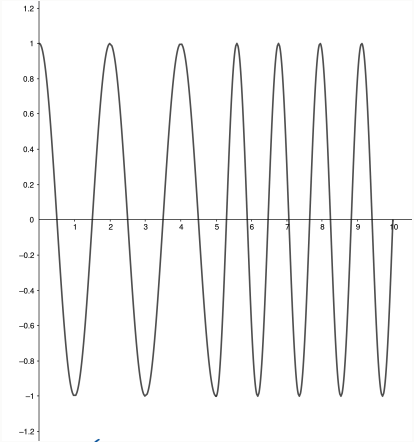


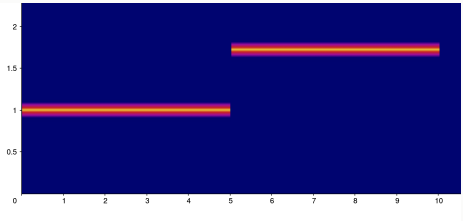
$$g(t) = \begin{cases} \cos(2\pi t) & \text{if } 0 \leq t < 5 \\ \cos(\frac{17}{10} \cdot 2\pi t + \pi) & \text{if } 5 \leq t \leq 10 \end{cases}$$



$\mathsf{FFT}(g)$

Auch mit Fenster





$$g(t) = \begin{cases} \cos(\pi t) & \text{if } 0 \leq t < 5 \\ \cos(\frac{17}{10} \cdot 2\pi t + \frac{2\pi}{2}) & \text{if } 5 \leq t \leq 10 \end{cases}$$

$\mathrm{WFT}(g)$

UNIVERSITÄT BONN

Digital und Analog

Analoges Signal: $f : \mathbb{R} \to \mathbb{R}$

Sampling mit Rate s: $f_s : \mathbb{Z} \to \mathbb{R}$

UNIVERSITÄT BONN

Digital und Analog

Analoges Signal: $f : \mathbb{R} \to \mathbb{R}$

Sampling mit Rate s: $f_s : \mathbb{Z} \to \mathbb{R}$

Samplingtheorem (Shannon):
Enthält f keine Frequenz über B Hz und ist $s \geq 2B$, so kann f perfekt aus f_s rekonstruiert werden.

UNIVERSITÄT BONN

Digital und Analog

Analoges Signal: $f : \mathbb{R} \to \mathbb{R}$

Sampling mit Rate s: $f_s : \mathbb{Z} \to \mathbb{R}$

Samplingtheorem (Shannon):
Enthält f keine Frequenz über B Hz und ist $s \geq 2B$, so kann f perfekt aus f_s rekonstruiert werden.

Erweiterung:
Ist die Bandbreite von f durch BW_f beschränkt, ist zur perfekten Rekonstruktion $s \geq BW_f$ erforderlich.

Signale sind komplex

Signale sind komplex

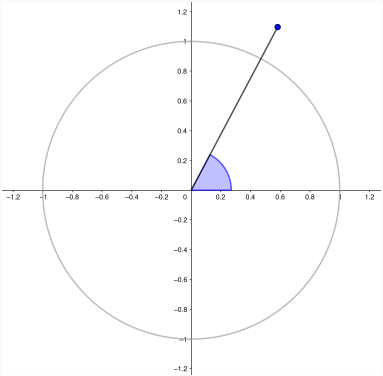
Aber nicht kompliziert

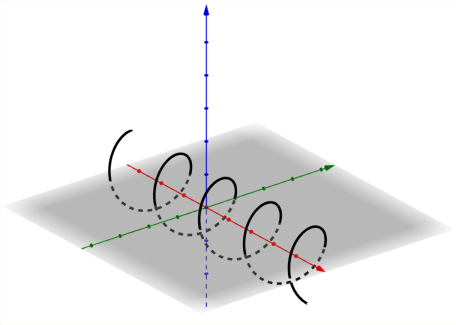
UNIVERSITÄT BONN

Komplexe Signale

$$f : \mathbb{R} \to \mathbb{C}$$

$$e^{2\pi i f t} = \cos(2\pi f t) + i \cdot \sin(2\pi f t)$$

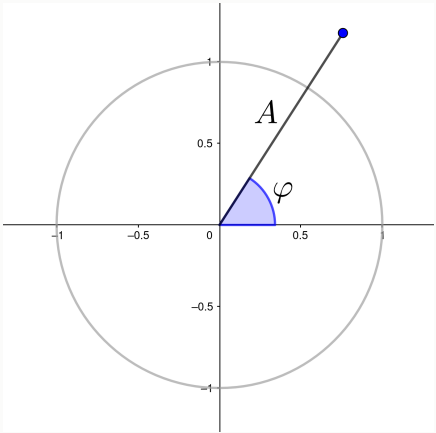




UNIVERSITÄT BONN

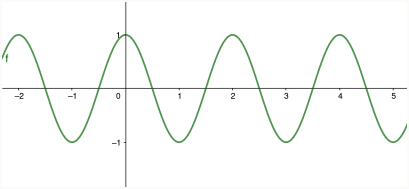
Eigenschaften bei komplexen Signalen

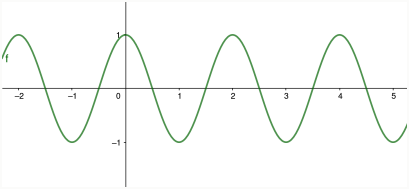
- Amplitude A
- Frequenz f
- Phase φ



UNIVERSITÄT BONN

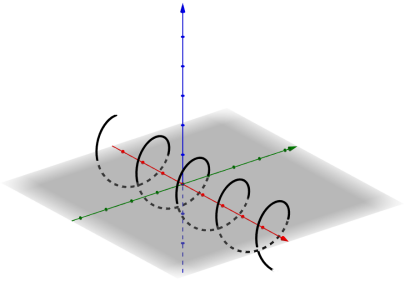
Wer erkennt den Unterschied?

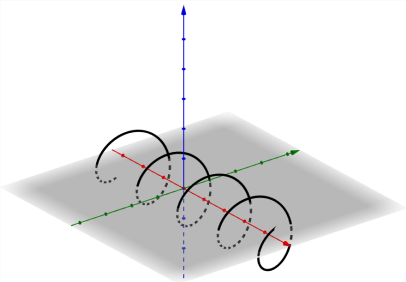




UNIVERSITÄT BONN

Wer erkennt den Unterschied?





UNIVERSITÄT BONN

Drahtlose Signale

UNIVERSITÄT BONN

Drahtlose Signale

- Geteiltes Medium

UNIVERSITÄT BONN

Drahtlose Signale

- Geteiltes Medium
- Reguliertes Medium

Drahtlose Signale

- Geteiltes Medium
- Reguliertes Medium

Frequenzteilplan:	**191**	**Eintrag:**	191003	**Stand:**	OKTOBER 2019
Frequenzbereich:	39,00 - 39,5 MHz				
Nutzungsbestimmung(en):	5 31				
Funkdienst:	Nichtnavigatorischer Ortungsfunkdienst D132A				
Nutzung:	ziv				
Frequenznutzung:	Ozeanographische Radare				
Frequenzteilbereich(e):	39,00 - 39,50 MHz				
Frequenznutzungs-bedingungen:					

Quelle: Bundesnetzagentur, Frequenzplan (Stand 02.12.2019)

UNIVERSITÄT BONN

Drahtlose Signale

- Geteiltes Medium
- Reguliertes Medium
- ISM-Bänder

UNIVERSITÄT BONN

Modulation

Signal $s : \mathbb{R} \to \mathbb{C}$

Trägersignal $c(t) = e^{2\pi i f t}$

Moduliertes Signal $m = s \cdot c$

UNIVERSITÄT BONN

Beispiel: Modulation von Audio

Signal $s : \mathbb{R} \to \mathbb{C}$

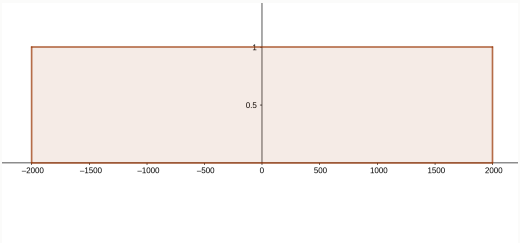
$$\forall f \geq 2 \text{ kHz} : |\text{FFT}(s)(f)| = 0$$

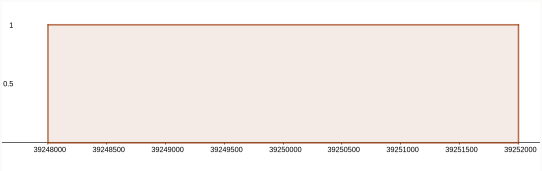
Trägersignal ($f = 39{,}25$ MHz)

$$c(t) = e^{2\pi i f t}$$

Moduliertes Signal

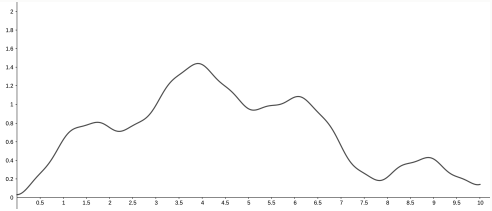
$$m(t) = s(t) \cdot c(t)$$

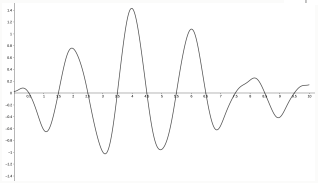




UNIVERSITÄT BONN

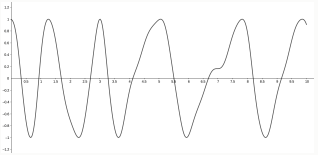
Modulationsverfahren





AM

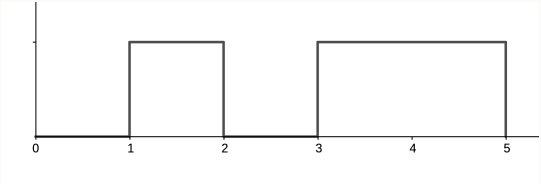
FM

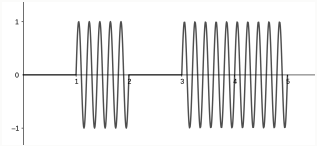


PM

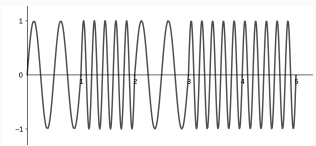
UNIVERSITÄT BONN

Modulationsverfahren

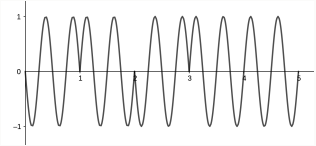




ASK



FSK



PSK

Grundlagen

Teil 2: SDR, DSP und Co.

Software **D**efined **R**adio

Idee: universelle Hardware, Spezialisierung durch Software

Vorteile:

- Flexibilität
- Kosten

Nachteile:

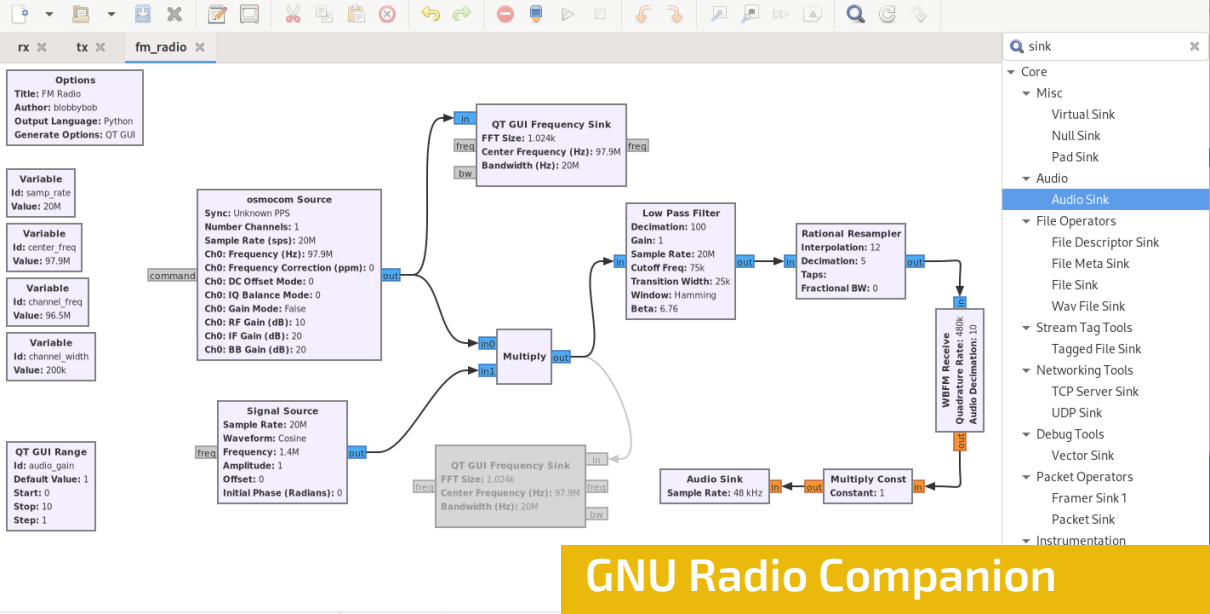
- Ineffizient
- Kosten

UNIVERSITÄT BONN

HackRF One

- Open-Source SDR Hardware
- 20 Msps ($\rightarrow$ 20 MHz Bandbreite)
- 16 Bit IQ Samples
- 1 MHz bis 6 GHz
- halbduplex

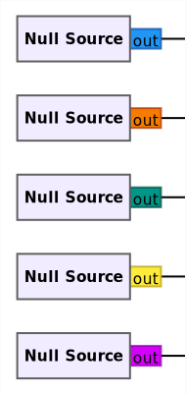
GNU Radio


rx
tx
fm_radio
Options
Title: FM Radio
Author: blobbybob
Output Language: Python
Generate Options: QT GUI
Variable
Id: samp_rate
Value: 20M
Variable
Id: center_freq
Value: 97.9M
Variable
Id: channel_freq
Value: 96.5M
Variable
Id: channel_width
Value: 200k
QT GUI Range
Id: audio_gain
Default Value: 1
Start: 0
Stop: 10
Step: 1
osmocom Source
Sync: Unknown PPS
Number Channels: 1
Sample Rate (sps): 20M
Ch0: Frequency (Hz): 97.9M
Ch0: Frequency Correction (ppm): 0
Ch0: DC Offset Mode: 0
Ch0: IQ Balance Mode: 0
Ch0: Gain Mode: False
Ch0: RF Gain (dB): 10
Ch0: IF Gain (dB): 20
Ch0: BB Gain (dB): 20
command
QT GUI Frequency Sink
FFT Size: 1.024k
Center Frequency (Hz): 97.9M
Bandwidth (Hz): 20M
Multiply
Signal Source
Sample Rate: 20M
Waveform: Cosine
Frequency: 1.4M
Amplitude: 1
Offset: 0
Initial Phase (Radians): 0
QT GUI Frequency Sink
FFT Size: 1.024k
Center Frequency (Hz): 97.9M
Bandwidth (Hz): 20M
Low Pass Filter
Decimation: 100
Gain: 1
Sample Rate: 20M
Cutoff Freq: 75k
Transition Width: 25k
Window: Hamming
Beta: 6.76
Rational Resampler
Interpolation: 12
Decimation: 5
Taps:
Fractional BW: 0
WBFM Receive
Quadrature Rate: 480k
Audio Decimation: 10
Multiply Const
Constant: 1
Audio Sink
Sample Rate: 48 kHz
sink
Core
Misc
Virtual Sink
Null Sink
Pad Sink
Audio
Audio Sink
File Operators
File Descriptor Sink
File Meta Sink
File Sink
Wav File Sink
Stream Tag Tools
Tagged File Sink
Networking Tools
TCP Server Sink
UDP Sink
Debug Tools
Vector Sink
Packet Operators
Framer Sink 1
Packet Sink
Instrumentation
QT GUI Frequency Sink
QT GUI Histogram Sink
QT GUI Number Sink
<<< Welcome to GNU Radio Companion 3.8.1.0 >>>
Block paths:
/usr/share/gnuradio/grc/blocks
Id
Value
Variables
audio_gain
<Open Properties>
GNU Radio Companion

UNIVERSITÄT BONN

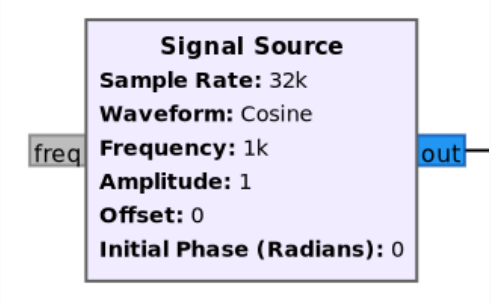
GNU Radio: Datentypen

- complex
- float
- int
- short
- byte



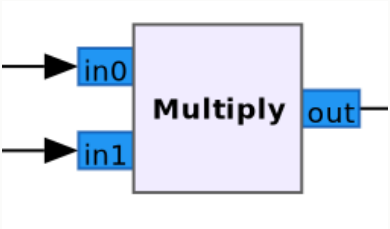
UNIVERSITÄT BONN

GNU Radio: Blöcke



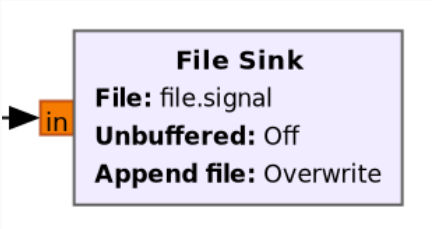
UNIVERSITÄT BONN

GNU Radio: Blöcke



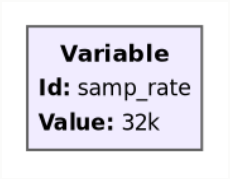
UNIVERSITÄT BONN

GNU Radio: Blöcke



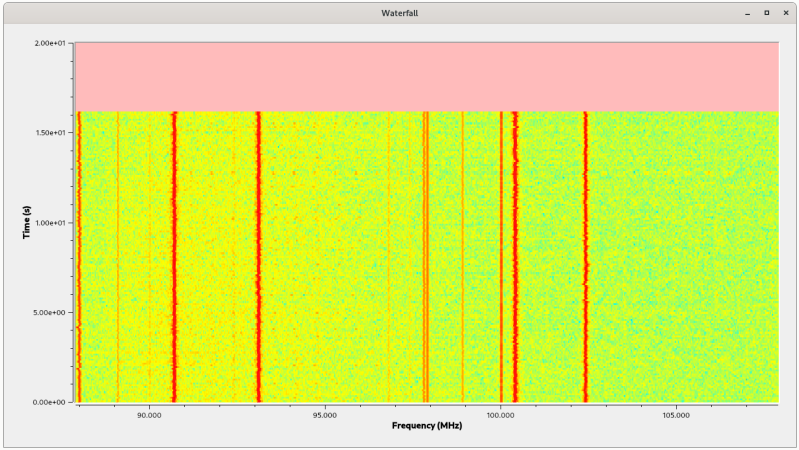
UNIVERSITÄT BONN

GNU Radio: Blöcke



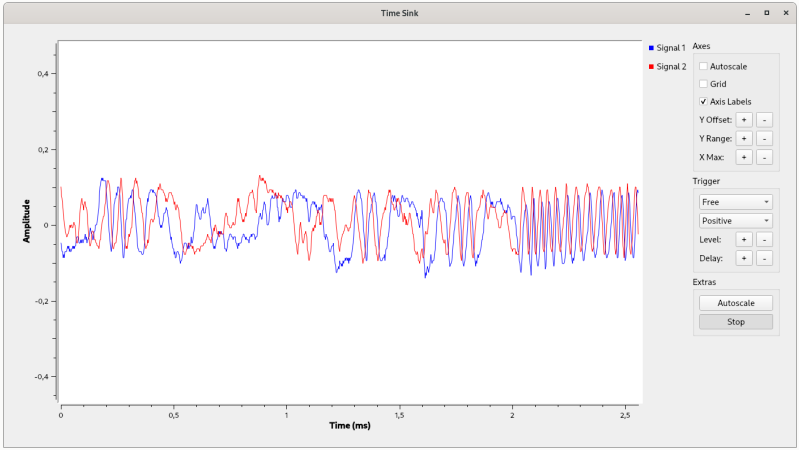
UNIVERSITÄT BONN

Wasserfall



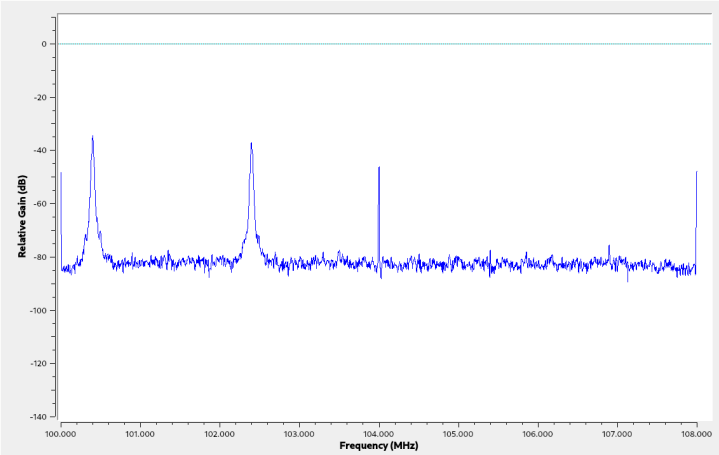
UNIVERSITÄT BONN

Zeitspüle



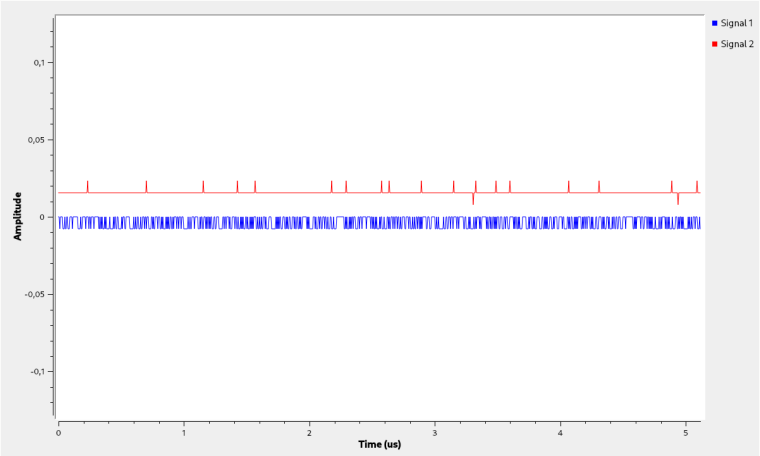
UNIVERSITÄT BONN

DC Bias



UNIVERSITÄT BONN

IQ Offset



Und Action

Teil 1: Die Stühle brennen...

UNIVERSITÄT BONN

Objekt der Begierde

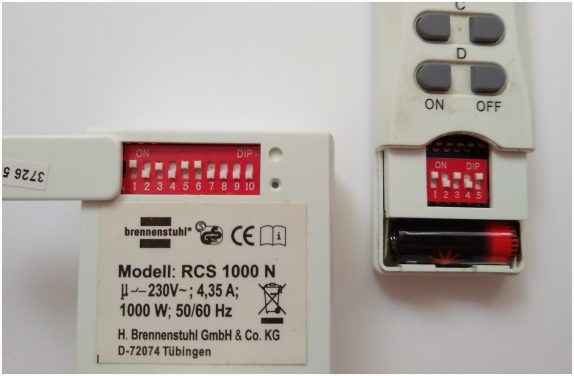
- Objekt: Funksteckdose
- Hersteller: Brennenstuhl
- Modell: RCS 1000 N/SN
- 3 Funksteckdosen
- 4-Kanal Fernbedienung
- „Ideal für zahlreiche DIY-Projekte”
- A23 (12 V) Batterie

UNIVERSITÄT BONN

Objekt der Begierde

UNIVERSITÄT BONN

Objekt der Begierde



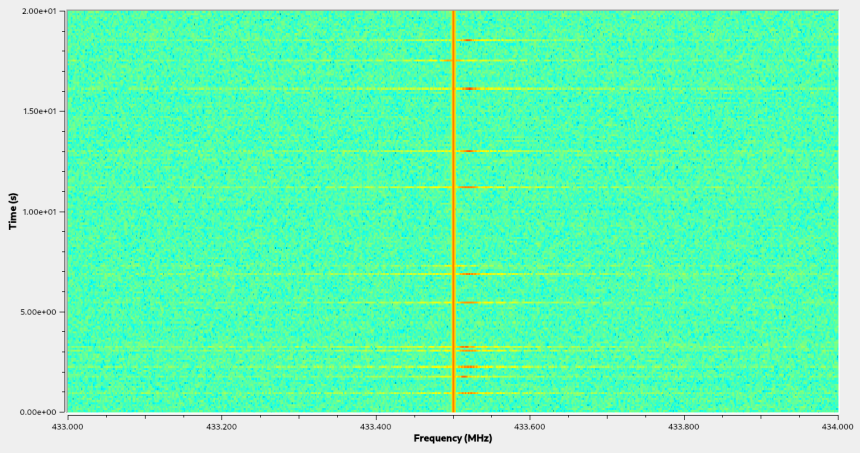
UNIVERSITÄT BONN

Ort der Kundgebung

Angegebene Frequenz: 433,92 MHz

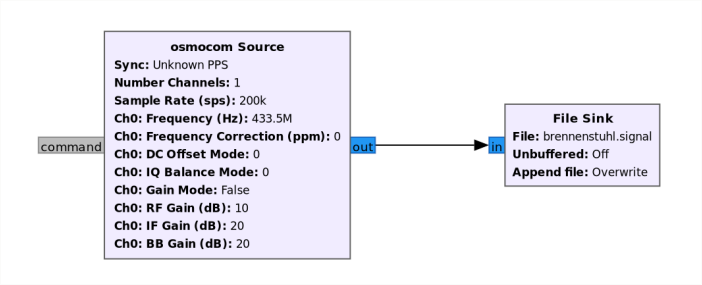
UNIVERSITÄT BONN

Ort der Kundgebung



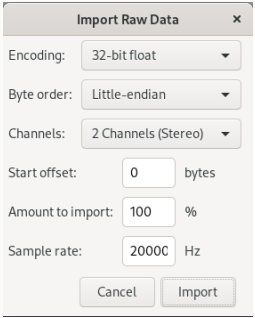
UNIVERSITÄT BONN

Signal im Detail



Signal im Detail

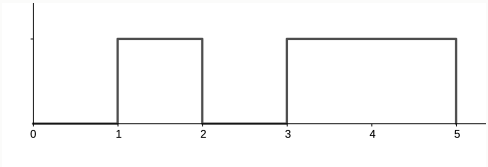
Audacity (File > Import > Raw Data)



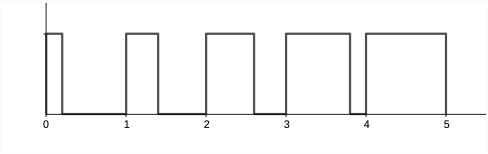
Live Analyse

Modulationsverfahren

OOK
(On-Off Keying)



PWM
(Pulsweitenmodulation)

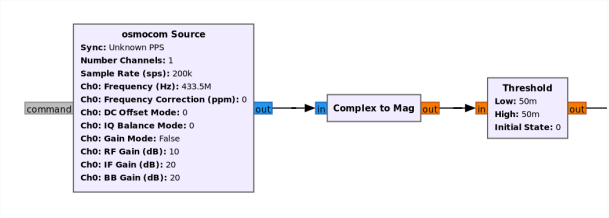


Und jetzt?

UNIVERSITÄT BONN

Demodulator: Preprocessing

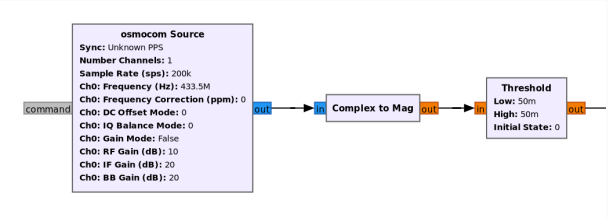
- Osmocom Source
- Complex to Mag
- Threshold



UNIVERSITÄT BONN

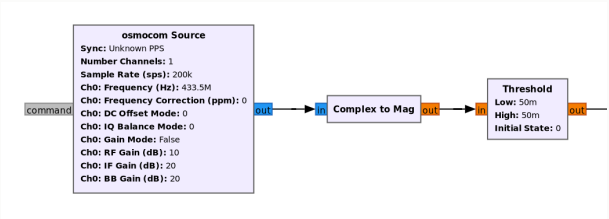
Demodulator: Preprocessing

- Osmocom Source
- Complex to Mag
- Threshold
- ...



Demodulator: Preprocessing

- Osmocom Source
- Complex to Mag
- Threshold
- ...
- Eigenen Block schreiben



UNIVERSITÄT BONN

Modul erstellen

```
1 $ gr_modtool newmod
2 Name of the new module: brennenstuhl
3 Creating out-of-tree module in ./gr-brennenstuhl...
4 Done.
5 Use 'gr_modtool add' to add a new block to this currently empty
    module.
```

UNIVERSITÄT BONN

Block erstellen

```
$ gr_modtool add
GNU Radio module name identified: brennenstuhl
('sink', 'source', 'sync', 'decimator', 'interpolator', 'general',
    'tagged_stream', 'hier', 'noblock')
Enter block type: general
Language (python/cpp): cpp
Language: C++
Enter name of block/code (without module name prefix): demodulator
Block/code identifier: demodulator
Enter valid argument list, including default arguments:
bool debug
...
```

UNIVERSITÄT BONN

Block erstellen

```
$ gr_modtool add
...
...
Add Python QA code? [Y/n] n
Add C++ QA code? [Y/n] n
Adding file 'lib/demodulator_impl.h'...
Adding file 'lib/demodulator_impl.cc'...
Adding file 'include/brennenstuhl/demodulator.h'...
Editing swig/brennenstuhl_swig.i...
Adding file 'grc/brennenstuhl_demodulator.block.yml'...
Editing grc/CMakeLists.txt...
```

UNIVERSITÄT BONN

Modulstruktur

```
|- apps
|- cmake
|- docs
|- examples
|- grc
|   |- brennenstuhl_demodulator.block.yml
|- include
|   |- brennenstuhl
|       |- api.h
|       |- modulator.h
|- lib
|-  |- modulator_impl.cc
|-  |- modulator_impl.h
|- python
|- swig
```

UNIVERSITÄT BONN

block.yml

```yaml
label: Brennenstuhl Demodulator
category: '[Brennenstuhl]'

templates:
  imports: import brennenstuhl
  make: brennenstuhl.demodulator(${debug})

parameters:
  - id: debug
    label: Debug
    dtype: bool

inputs:
  - label: in
    dtype: byte
```

UNIVERSITÄT BONN

demodulator_impl.cc

```
/*
 * The private constructor
 */
demodulator_impl::demodulator_impl(bool debug)
        : gr::block( name: "demodulator",
                     input_signature: gr::io_signature::make( min_streams: 1,  max_streams: 1, sizeof(unsigned char)),
                     output_signature: gr::io_signature::make( min_streams: 0,  max_streams: 0,  sizeof_stream_item: 0)),
          d_debug(debug),
          d_packet() {...}

int demodulator_impl::general_work(int noutput_items,
                                   gr_vector_int &ninput_items,
                                   gr_vector_const_void_star &input_items,
                                   gr_vector_void_star &output_items) {...}
```

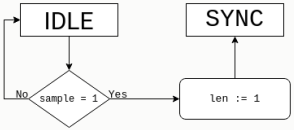
UNIVERSITÄT BONN

demodulator_impl.cc

```
int demodulator_impl::general_work(int noutput_items,
                                   gr_vector_int &ninput_items,
                                   gr_vector_const_void_star &input_items,
                                   gr_vector_void_star &output_items) {
    const auto *in = (const unsigned char *) input_items[0];

    int i;
    for (i = 0; i < ninput_items[0]; i++) {
        if (d_idle) {
            if (d_sync) {
                if (in[i] == 1) {
                    if (d_debug) {
                        std::cerr << "Info: Found possible preamble." << std::endl;
                    }
                    d_sync = true;
                    d_symb_len = 1;
                }
            } else {
                if (in[i] == 1 && d_last_sample == 0) {
                    if (d_symb_len >= d_min_symbol_len) {
                        // Use fixed symbol length if set
                        if (d_fixed_symbol_len > 0) d_symb_len = d_fixed_symbol_len;

                        // Transition to receiving state
                        if (d_debug) {
                            std::cerr << "Info: Started Packet receiving. Symbol length: " << d_symb_len
                                      << std::endl;
                        }
                        d_sync = false;
                        d_idle = false;
                        d_symb_pos = 0;
                        d_symb_zeros = 0;
                        d_symb_ones = 0;
                        d_packet.clear();
                    } else {
                        d_sync = false;
                        if (d_debug) {
                            std::cerr << "Info: Discarding preamble (symbol length too small)" << std::endl;
                        }
                    }
                } else {
                    d_symb_len++;
                }
                if (d_symb_len > d_max_symbol_len) {
                    d_sync = false;
                    std::cerr << "Info: Discarding preamble (symbol length too large)" << std::endl;
                }
            }
        }

        if (d_idle) {
            switch (in[i]) {
                case 0:
                    d_symb_zeros++;
                    break;
                case 1:
                    d_symb_ones++;
                    break;
                default:
                    if (d_debug) {
                        std::cerr << "Warning: received invalid sample with value " << +in[i] << std::endl;
                    }
```

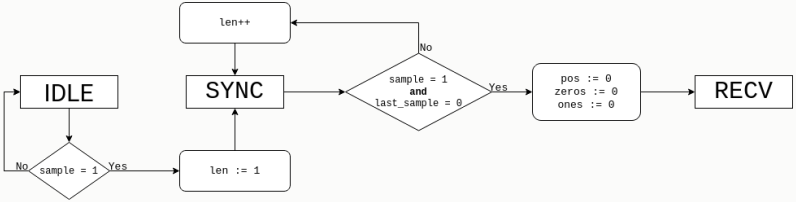
UNIVERSITÄT BONN

Demodulator: Funktionsweise



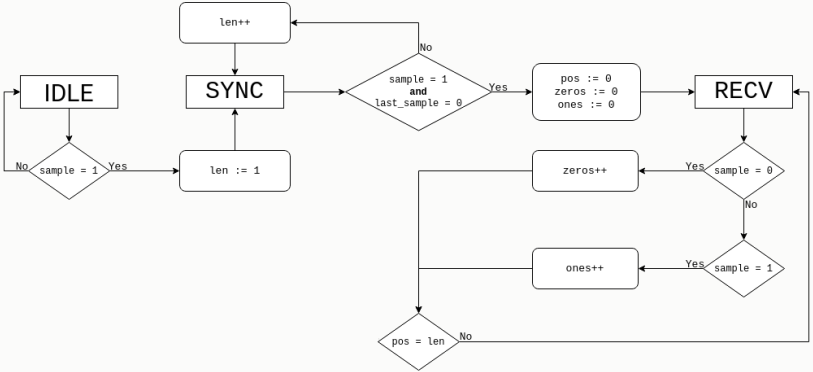
UNIVERSITÄT BONN

Demodulator: Funktionsweise



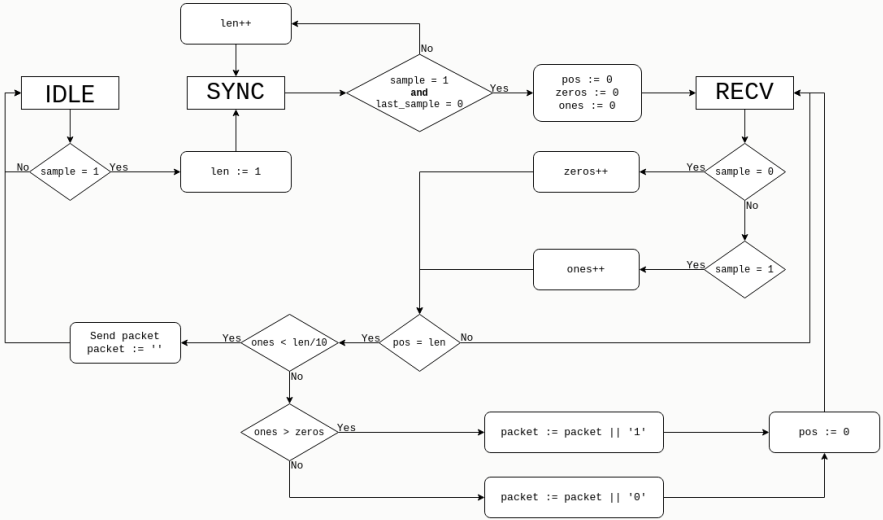
UNIVERSITÄT BONN

Demodulator: Funktionsweise



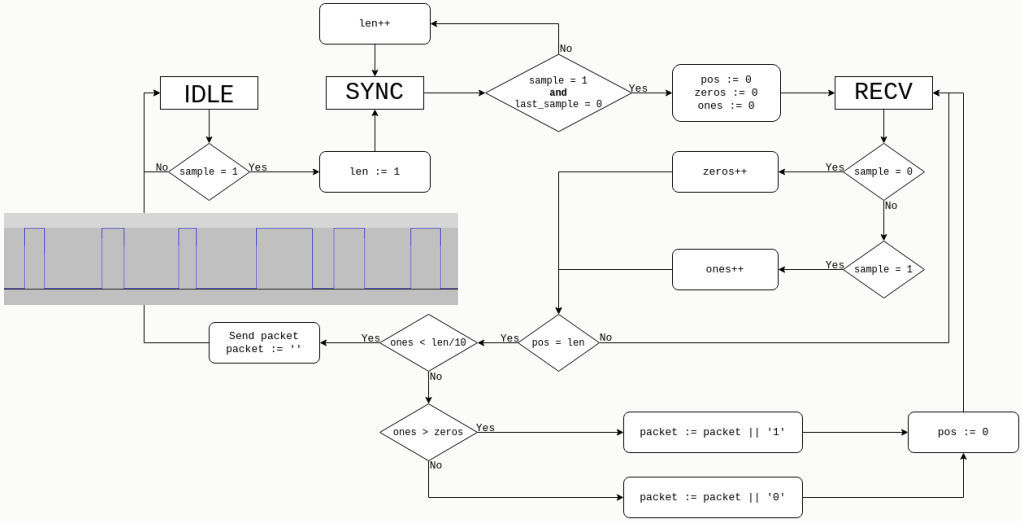
UNIVERSITÄT BONN

Demodulator: Funktionsweise



UNIVERSITÄT BONN

Demodulator: Funktionsweise



UNIVERSITÄT BONN

Demodulator: Ausgabe

```
Info: Found possible preamble.
Info: Discarding preamble (symbol length too large)
Info: Found possible preamble.
Info: Started Packet receiving. Symbol length: 263
Info: Received packet: 00100010000101010100010
Info: Found possible preamble.
Info: Started Packet receiving. Symbol length: 264
Info: Received packet: 00101011000010101010110
Info: Found possible preamble.
Info: Started Packet receiving. Symbol length: 262
Info: Received packet: 00100010110010111010010
Info: Found possible preamble.
Info: Started Packet receiving. Symbol length: 261
Info: Received packet: 00100010000101010110010
```

UNIVERSITÄT BONN

Daten sammeln

```
## A on
manual: 00101010000010101010010   00101010000010101010010   00101010000010101010110   00101010001010101011~
demod:  00101010000010101010001   00101010000010101010010   00101010000010101010111   00101010001010101011~
symlen: 268                       262                       267                       262
errors: too short                 none                      too short                 none

## B on
manual: 00101010001000101010010   00101010001000101010010   00101010001000101010010   00101010001000101010~
demod:  00101010001000101010010   00101010001000101010010   00101010001000101010010   00101011001000101111~
symlen: 261                       264                       262                       256
errors: none                      none                      none                      bit error(s)
```

UNIVERSITÄT BONN

Demodulator: Verbesserungen

- Probleme
  - Sensitive Synchronisation
  - Absoluter Threshold

UNIVERSITÄT BONN

Demodulator: Verbesserungen

- Probleme
  - Sensitive Synchronisation
  - Absoluter Threshold
- Lösungen

UNIVERSITÄT BONN

Demodulator: Verbesserungen

- Probleme
  - Sensitive Synchronisation
  - Absoluter Threshold
- Lösungen
  - Averaging Filter

UNIVERSITÄT BONN

Demodulator: Verbesserungen

- Probleme
  - Sensitive Synchronisation
  - Absoluter Threshold
- Lösungen
  - Averaging Filter
  - Fehlertoleranz

UNIVERSITÄT BONN

Demodulator: Verbesserungen

- Probleme
  - Sensitive Synchronisation
  - Absoluter Threshold
- Lösungen
  - Averaging Filter
  - Fehlertoleranz
  - Kontinuierliche Synchronisation

UNIVERSITÄT BONN

Demodulator: Verbesserungen

- Probleme
  - Sensitive Synchronisation
  - Absoluter Threshold
- Lösungen
  - Averaging Filter
  - Fehlertoleranz
  - Kontinuierliche Synchronisation
  - Symbollänge festsetzen

UNIVERSITÄT BONN

Feste Symbollänge

```
1x255: 01
1x256: 01
1x257: 01
1x258: 01
1x259: 00
4x260: 00 01 00 00
4x261: 00 00 00 00
5x262: 00 00 00 00 00
4x263: 00 00 00 00
4x264: 00 00 00 00
3x265: 00 00 10
3x266: 00 10 10
2x267: 10 10
1x268: 10
1x269: 11
```

UNIVERSITÄT BONN

Daten sammeln

```
A on:  00101010000010101010010   00101010000010101010010   00101010000010101010111
B on:  00101010001000101010010   00101010001000101010010   00101010001000101010010
C on:  00101010001010001010010   00101010001010001010010   00101010001010001011111
D on:  00101010001010100010010   00101010001010100010010   00101010001010100010010
A off: 00101010000010101010100   00101010000010101010100   00101010000010101010101
B off: 00101010001000101010100   00101010001000101010101   00101010001010101011111
C off: 00101010001010001010100   00101010001010001010100   00101010001010001010101
D off: 00101010001010100010100   00101010001010100010100   00101010001010100010100
```

UNIVERSITÄT BONN

Daten auswerten

```
A on:   00101010000010101010100010
B on:   00101010001000101010100010
C on:   00101010001010001010100010
D on:   00101010001010100010100010
```

UNIVERSITÄT BONN

Daten auswerten

```
A on:   00101010000010101010100010
B on:   00101010001000101010100010
C on:   00101010001010001010100010
D on:   00101010001010100010100010
A off:  00101010000010101010101000
B off:  00101010001000101010101000
C off:  00101010001010001010101000
D off:  00101010001010100010101000
```

UNIVERSITÄT BONN

Packetformat

UNIVERSITÄT BONN

Dekodierer: Kommunikation

- Problem: Zusätzliche Information über Paketstart nötig

UNIVERSITÄT BONN

Dekodierer: Kommunikation

- Problem: Zusätzliche Information über Paketstart nötig
- Lösungen:
  - Messages: Asynchrone Nachrichten
  - Tags: Zusätzliche Informationen

UNIVERSITÄT BONN

Dekodierer: Kommunikation

- Problem: Zusätzliche Information über Paketstart nötig
- Lösungen:
  - Messages: Asynchrone Nachrichten
  - Tags: Zusätzliche Informationen
  - In jedem Fall: PMT (Permutable Types)

UNIVERSITÄT BONN

Dekodierer: Code

```
def __init__(self, debug=False):
    gr.basic_block.__init__(self,
                            name="brennenstuhl_parser",
                            in_sig=None,
                            out_sig=None)

    self.__debug = debug

    self.message_port_register_in(pmt.intern("packet"))
    self.set_msg_handler(pmt.intern("packet"), self.parse_packet)
```

UNIVERSITÄT BONN

Dekodierer: Code

```
def parse_packet(self, packet):

    if not pmt.is_symbol(packet):
        if self.__debug:
            print("PMT is no symbol (string)")
        return

    packet = pmt.to_python(packet)

    if len(packet) != 24:
        if self.__debug:
            print("Message length invalid:", len(packet))
        return

    if '11' in packet:
        if self.__debug:
            print("Discarding packet, because of repetitive 1's")
        return

    if packet[20] == '1' and packet[22] == '1':
        if self.__debug:
            print("Discarding packet, because on and off is both set")
        return
```

UNIVERSITÄT BONN

Dekodierer: Code

```
info = "DIP "
if packet[0] == '0':
    info += "1"
if packet[2] == '0':
    info += "2"
if packet[4] == '0':
    info += "3"
if packet[6] == '0':
    info += "4"
if packet[8] == '0':
    info += "5"
```

```
info += " Key "
if packet[10] == '0':
    info += "A"
if packet[12] == '0':
    info += "B"
if packet[14] == '0':
    info += "C"
if packet[16] == '0':
    info += "D"
if packet[18] == '0':
    info += "E"
```

```
if packet[20] == '1':
    info += " off"
if packet[22] == '1':
    info += " on"

print(info)
```

Live Demo

Und Action

Teil 2: Der Grillmeister

UNIVERSITÄT BONN

Objekt

- Objekt: Funk-Grillthermometer
- Hersteller: Globaltronics
- Modell: GT-TMBBQ-05s, GT-TMBBQ-05e
- Basisstation mit Fühler
- Empfänger mit verschiedenen Einstellungen

UNIVERSITÄT BONN

Objekt

UNIVERSITÄT BONN

Frequenz

- Keine Angabe auf dem Gerät
- Keine Angabe in der Anleitung

UNIVERSITÄT BONN

Frequenz

- Keine Angabe auf dem Gerät
- Keine Angabe in der Anleitung
- Nach längerer Recherche:

TECHNISCHE SPECIFICATIES

Model:	GT-TMBBQ-05s (zender) GT-TMBBQ-05e (ontv.)
Overdracht zonder kabel: 10 - 12 m	
Frequentieband:	433,92 MHz ± 330 kHz

Live Analyse

UNIVERSITÄT BONN

Datenanalyse

Datenanalyse

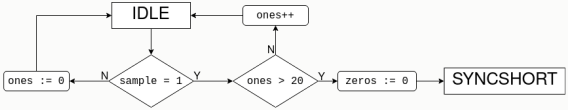
```
10000100001011011011100001110001 1--
10000100001011011011100001110001 1--
10000100001011011011100001110001 1--
10000100001011011011100001110001 1--
10000100001011011011100001110001 1--
10000100001011011011100001110001 1--
10000100001011011011100001110001 1
```

Datenanalyse

```
10000100001011011011100001110001 1
10000100001011001011100001111001 0
10000100001010111011100001110111 1
10000100001010110011100001111111 1
```

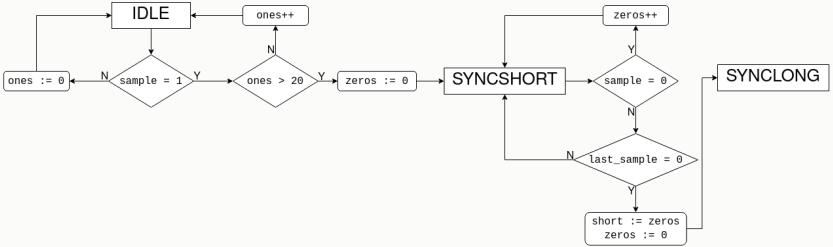
UNIVERSITÄT BONN

Demodulator: Funktionsweise



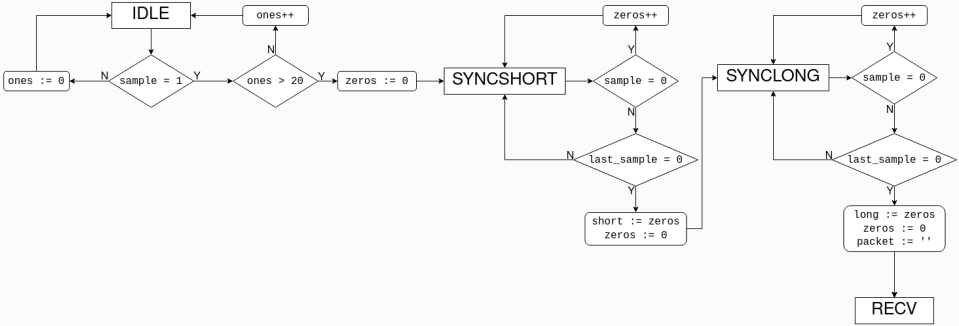
UNIVERSITÄT BONN

Demodulator: Funktionsweise



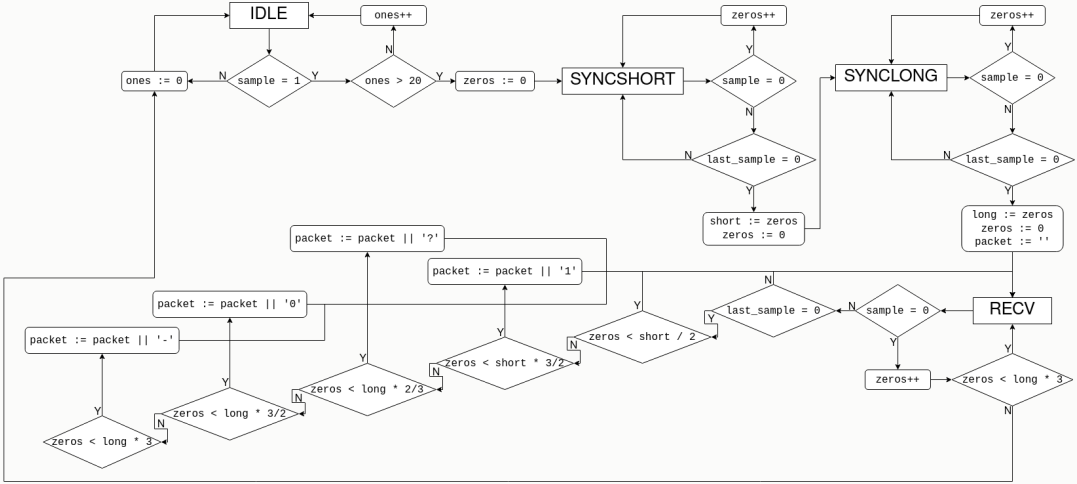
UNIVERSITÄT BONN

Demodulator: Funktionsweise

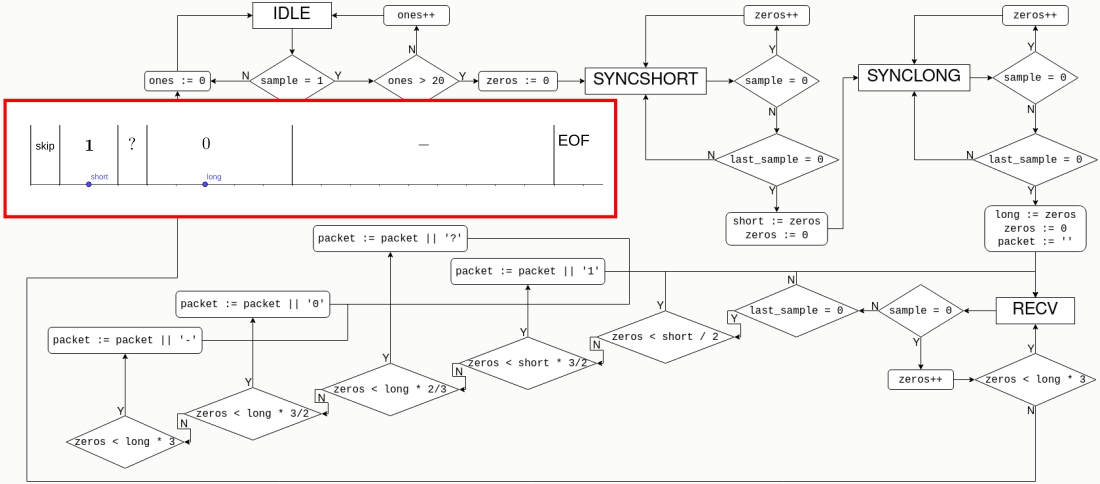


UNIVERSITÄT BONN

Demodulator: Funktionsweise

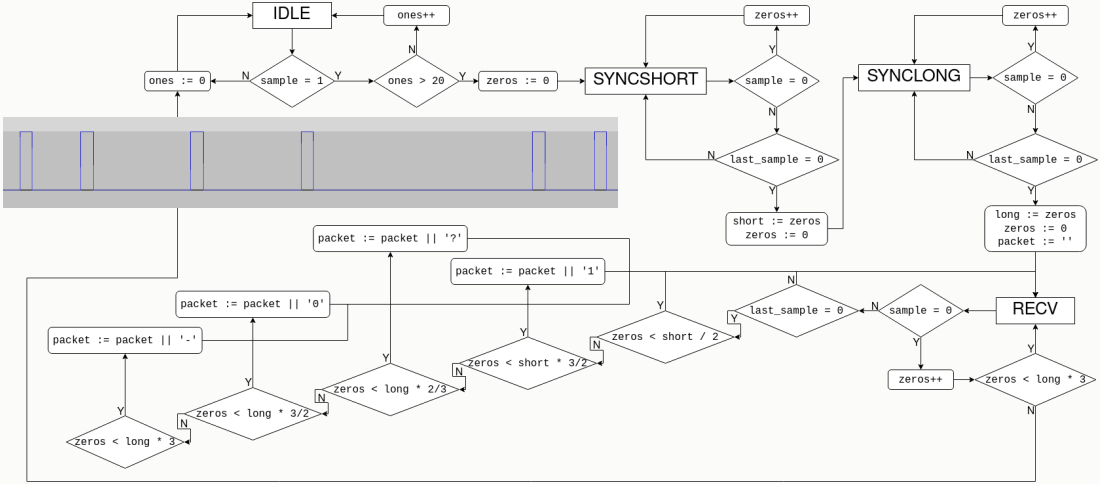


Demodulator: Funktionsweise



UNIVERSITÄT BONN

Demodulator: Funktionsweise

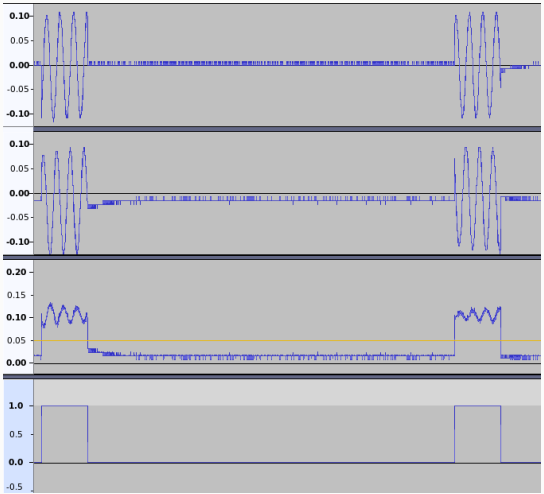


Fehlerhaft

```
11111111111

1101
-1
??1110
1-11
1
-1??-11??0?10
??0?110
111110111117011
-1
1
1


111?1110
11

1?11?????11??11???-?????111?111111?111???-?11?1?111?1?11?111111111?10??????111??1111????011??1??11111???11107
```

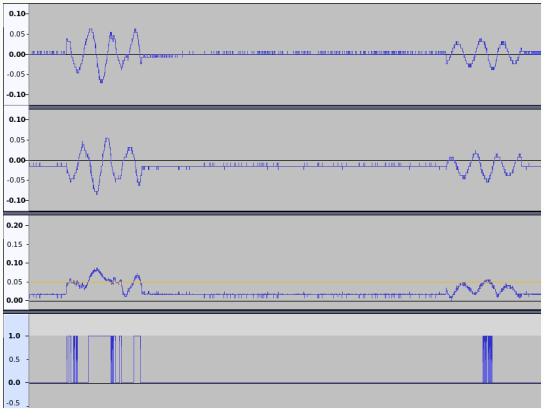
UNIVERSITÄT BONN

Aufbereitung



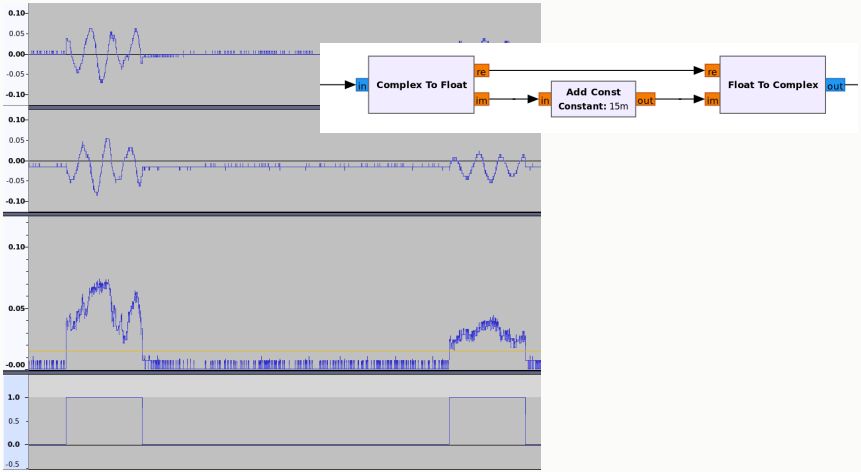
UNIVERSITÄT BONN

Aufbereitung



UNIVERSITÄT BONN

Aufbereitung



UNIVERSITÄT BONN

Fehlerhaft war gestern

```
00010000101101101110000111000011-1-10000100001011011011100001110000 11-1-1000010000101101
00010000101101101110000111000011-1-10000100001011011011100001110000 11-1-1000010000101101
00010000101101101110000111000011-1-10000100001011011011100001110000 11-1-1000010000101101
00010000101101101110000111000011-1-10000100001011011011100001110000 11-1-1000010000101101
00010000101101101110000111000011-1-10000100001011011011100001110000 11-1-1000010000101101
00010000101101101110000111000011-1-10000100001011011011100001110000 11-1-1000010000101101
00010000101100101110000111110010-1-10000100001011001011100001111001 0-1-1000010000101100
00010000101011101110000111101111-1-10000100001010111011100001110111 1-1-1000010000101011
00010000101011101110000111101111-1-10000100001010111011100001110111 1-1-1000010000101011
00010000101011101110000111101111-1-10000100001010111011100001110111 1-1-1000010000101011
00010000101011001110000111111111-1-10000100001010110011100001111111 1-1-1000010000101011
```

UNIVERSITÄT BONN

Das Wasser

- Idee: Sensor in abkühlendes Wasser zur Informationsgewinnung
- Empfänger zur Kontrolle aktiviert
- Ziel: Monoton fallende Temperaturen in den Paketen

UNIVERSITÄT BONN

Das Wasser

- Idee: Sensor in abkühlendes Wasser zur Informationsgewinnung
- Empfänger zur Kontrolle aktiviert
- Ziel: Monoton fallende Temperaturen in den Paketen
- Vorsicht: Thermometer ist laut

UNIVERSITÄT BONN

Finde die Bits

```
1  10000100001011011011100001110011
2  10000100001011000011100001110000
3  10000100011100101011100001011011
4  10000100011100000011100001011110
5  10000100011101010011100001011100
6  10000100011100111011100001010010
7  10000100011100101011100001011011
8  10000100011101000011100001010010
9  10000100011100100011100001010000
10 10000100011101000011100001010010
```

UNIVERSITÄT BONN

Hälfte weg

```
01011011        11100011
01011000        11100000
11100101        10110011
11100000        10111110
11101010        10111000
11100111        10100100
11100101        10110011
11101000        10100101
11100100        10100001
11101000        10100101
```

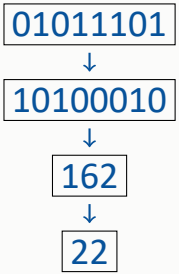
UNIVERSITÄT BONN

Zahlen und ???

```
 1          00000000        11110100
 2          00000001        11100100
 3          00000010        11100101
 4          00000011        11110111
 5          00000100        11100111
 6          00000101        11111001
 7          00000110        11111010
 8          00000111        11101010
 9          00000110        11111010
10          00000111        11101010
11          00001000        11101011
12          00000111        11101010
13          00001000        11101011
14          00001001        11111101
15          00001010        11111110
16          00001011        11101110
17          00001100        11110000
18          00001101        11100000
19          00001100        11110000
20          00001101        11100000
21          00001110        11100001
22          00001111        11110011
```

Skalenangleichung

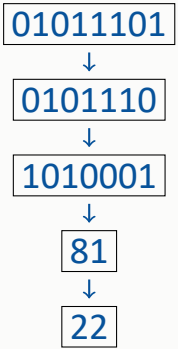
1. Invertieren
2. In Dezimal konvertieren
3. Angleichung



UNIVERSITÄT BONN

Skalenangleichung

0. Shiften
1. Invertieren
2. In Dezimal konvertieren
3. Angleichung



Live Demo

Und was könnt ihr machen?

UNIVERSITÄT BONN

Mit Geld

- SDR kaufen
  - HackRF One (20 Msps, 379€)

UNIVERSITÄT BONN

Mit Geld

- SDR kaufen
  - HackRF One (20 Msps, 379€)
  - USRP B210 (56 Msps, FPGA, 1290€)

UNIVERSITÄT BONN

Mit Geld

- SDR kaufen
  - HackRF One (20 Msps, 379€)
  - USRP B210 (56 Msps, FPGA, 1290€)
  - FLEX-6700 (245 Msps, FPGA, 100 W Tx, 7699€)

UNIVERSITÄT BONN

Mit Geld

- SDR kaufen
  - HackRF One (20 Msps, 379€)
  - USRP B210 (56 Msps, FPGA, 1290€)
  - FLEX-6700 (245 Msps, FPGA, 100 W Tx, 7699€)
- Funkende Gegenstände kaufen

UNIVERSITÄT BONN

Mit Geld

- SDR kaufen
  - HackRF One (20 Msps, 379€)
  - USRP B210 (56 Msps, FPGA, 1290€)
  - FLEX-6700 (245 Msps, FPGA, 100 W Tx, 7699€)
- Funkende Gegenstände kaufen

- Empfehlung: Finger weg von WiFi, Bluetooth und Co.

UNIVERSITÄT BONN

Ohne Geld

- HackRF SDR Tutorial

  https://greatscottgadgets.com/sdr/

- GNU Radio ausprobieren

UNIVERSITÄT BONN

Ohne Geld

- HackRF SDR Tutorial

  https://greatscottgadgets.com/sdr/

- GNU Radio ausprobieren
- Funksteckdosen ausprobieren
- Thermometer ausprobieren
- Infos auf GitHub

  https://github.com/BlobbyBob/rf-reversing

UNIVERSITÄT BONN

Universität Bonn

- Kommunikation in verteilten Systemen (Bachelor)
- Mobile Communications (Master)
- Foundations of Audio Signal Processing (Master)

UNIVERSITÄT BONN

Universität Bonn

- Kommunikation in verteilten Systemen (Bachelor)
- Mobile Communications (Master)
- Foundations of Audio Signal Processing (Master)
- PG, Seminar, Lab, Abschlussarbeiten (Daniel Vogel)

Ben Swierzy
Universität Bonn | Institut für Informatik 4
swierzy@uni-bonn.de

https://github.com/BlobbyBob/rf-reversing

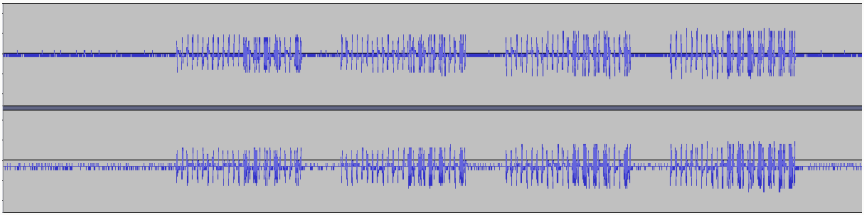
UNIVERSITÄT BONN

Weitere Links

- https://greatscottgadgets.com/hackrf/one/
- https://www.gnuradio.org/
- https://wiki.gnuradio.org/index.php/Tutorials
- https://www.audacityteam.org/

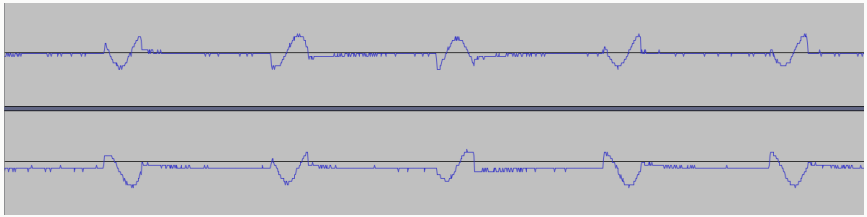
UNIVERSITÄT BONN

Live Analyse 1



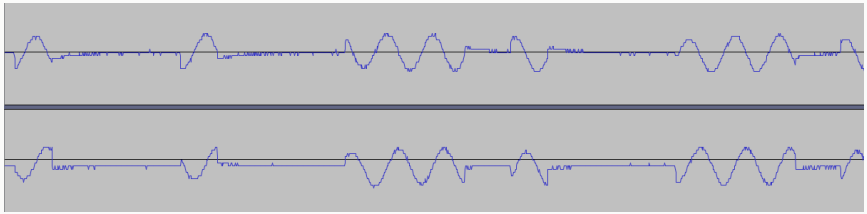
UNIVERSITÄT BONN

Live Analyse 1



UNIVERSITÄT BONN

Live Analyse 1

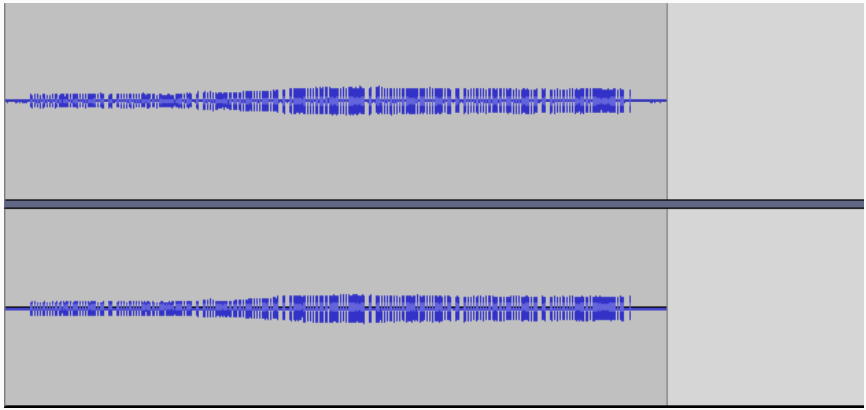


UNIVERSITÄT BONN

Live Analyse 1

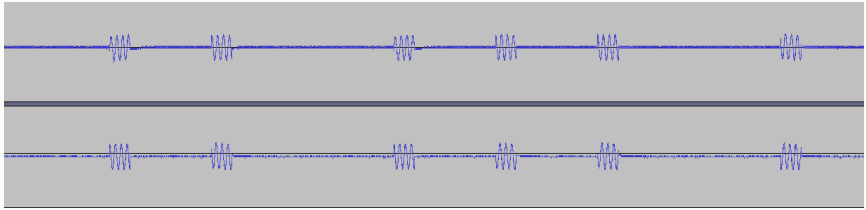
UNIVERSITÄT BONN

Live Analyse 2



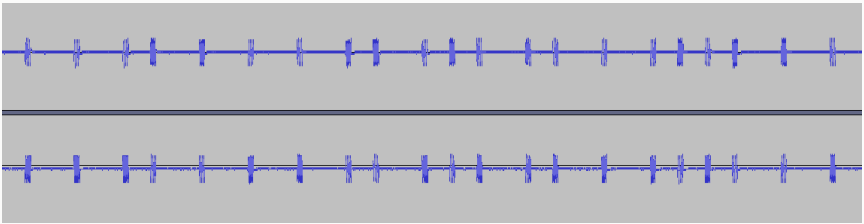
UNIVERSITÄT BONN

Live Analyse 2



UNIVERSITÄT BONN

Live Analyse 2



UNIVERSITÄT BONN

Live Analyse 2