Demo-Projekte

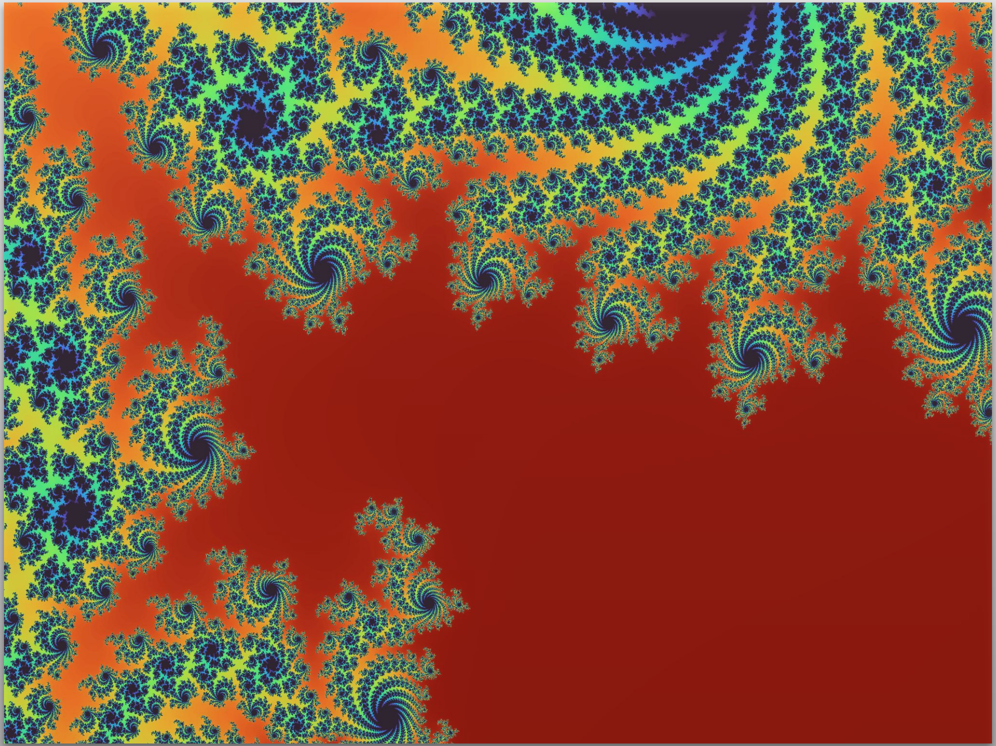
„Ich höre und ich vergesse. Ich sehe und ich erinnere mich. Ich tue und ich verstehe.“ – Konfuzius

Rafael Bachmann
Software-Entwickler

Speisekarte

- Multithreaded Renderer für Julia-Fraktale [rayon, clap, klask]
- Interpreter für minimales LISP: C Parser Library und Rust Applikation in “friedlicher” Koexistenz [C-FFI, thiserror, pest, pils, WASM]
- Async IO chat server [tokio, tokio-console, anyhow, clap, serde-json]
- Raspberry Pi LoRa transmitter/receiver mit CLI/GUI [snafu, clap, klask, embedded-hal]
- Email parser in WebAssembly [WASM, cross compilation, serde, mail-parser]
- Treiber für SDP8xx Differential Pressure Sensor [embedded-hal, i2c, mocking]
- Minimale Quadkopter-Firmware [stm32f1, mpu6050, nrf24, embedded-hal]

Multithreaded Renderer für Julia-Fraktale

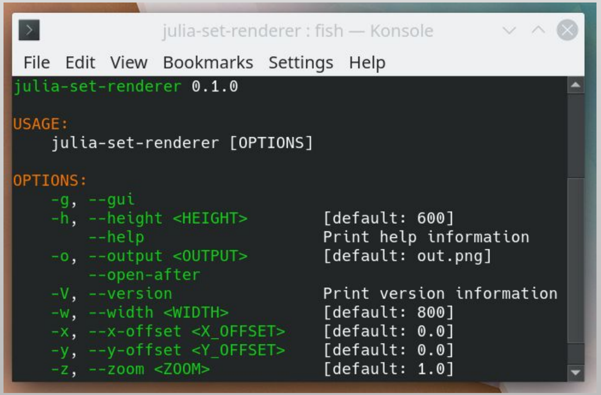


Multithreaded Renderer für Julia-Fraktale

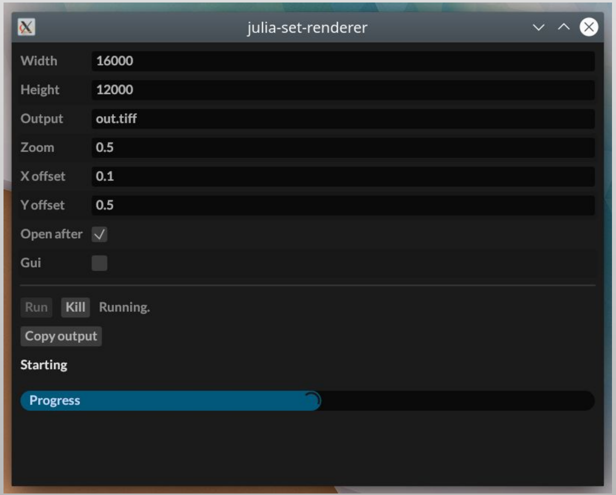
- Wie Mandelbrot-Fraktal: Iteriere Punkte in der Komplexen Ebene.

  Konvergierende Punkte sind Teil der Julia-Menge.
- "Embarassingly Parallelizable Problem"
- (Parallel != Nebenläufig)
- Mutierbare Iteration über alle Pixel (iter_mut)
- Parallelität automatisch durch `rayon` (par_iter_mut)

Multithreaded Renderer für Julia-Fraktale

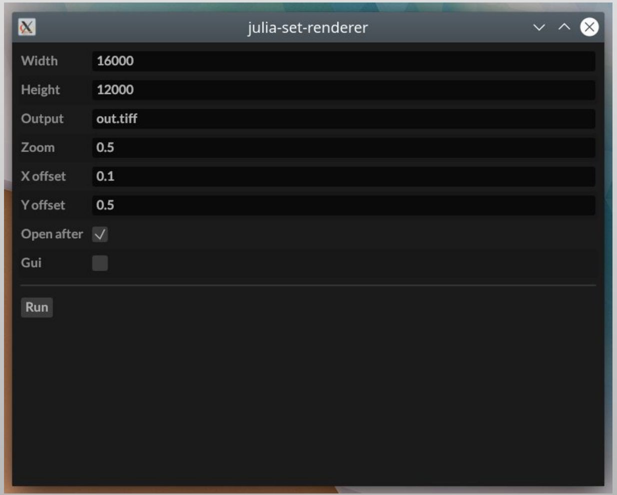

julia-set-renderer : fish — Konsole
File Edit View Bookmarks Settings Help

```
julia-set-renderer 0.1.0

USAGE:
    julia-set-renderer [OPTIONS]

OPTIONS:
    -g, --gui
    -h, --height <HEIGHT>         [default: 600]
        --help                    Print help information
    -o, --output <OUTPUT>         [default: out.png]
        --open-after
    -V, --version                 Print version information
    -w, --width <WIDTH>           [default: 800]
    -x, --x-offset <X_OFFSET>     [default: 0.0]
    -y, --y-offset <Y_OFFSET>     [default: 0.0]
    -z, --zoom <ZOOM>             [default: 1.0]
```



Width 16000
Height 12000
Output out.tiff
Zoom 0.5
X offset 0.1
Y offset 0.5
Open after ✓
Gui

Running.
Starting



Width 16000
Height 12000
Output out.tiff
Zoom 0.5
X offset 0.1
Y offset 0.5
Open after ✓
Gui

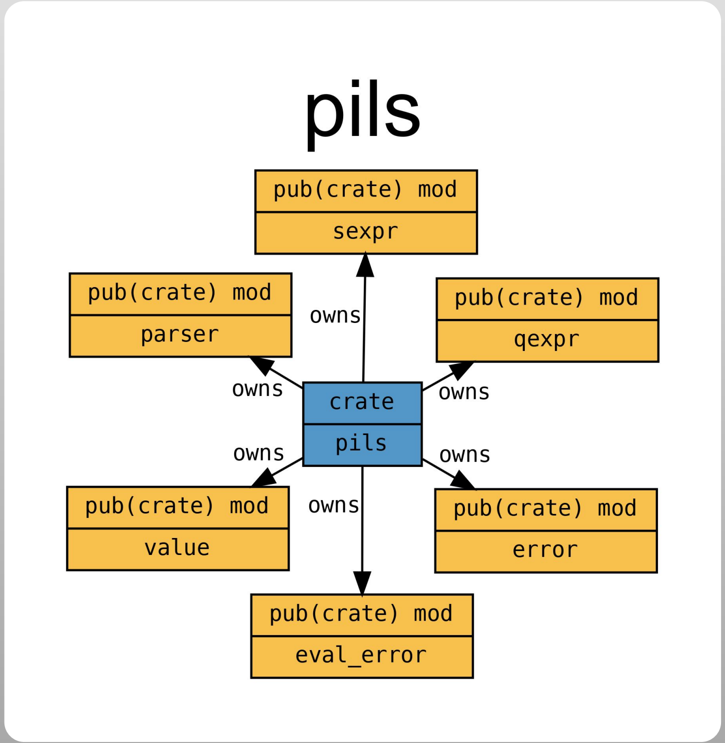
github.com/cocomundo/julia-set-renderer

Pils: Interpreter für minimales LISP

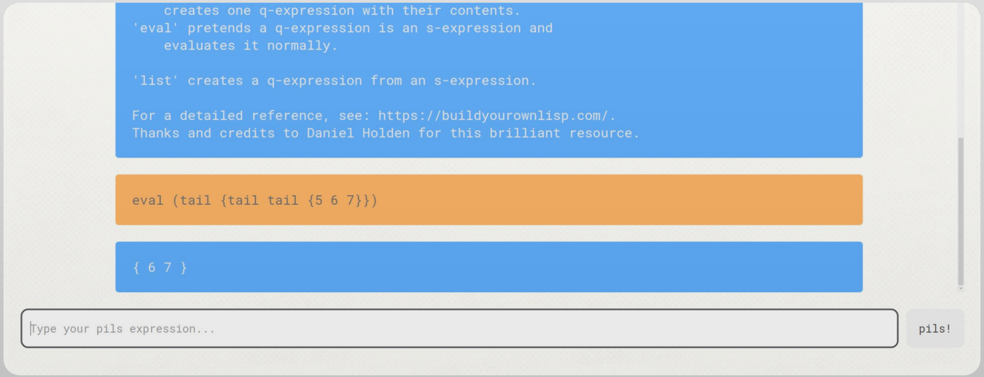
- Inspiriert von: buildyourownlisp.com
- Mini-Lisp: S-Expressions, Q-Expressions, Values, Operators, …
- Implementierung 1 imitiert exakt C Originalvariante
    - Implizite Annahmen aus C verunstalten Rust Code
    - Siehe c2rust online Demo
- Implementierung 2: idiomatisches Rust
    - Als web-REPL via WASM verfügbar

Pils: Interpreter für minimales LISP

```
cargo modules generate graph --lib > mods.dot
```



Pils: Interpreter für minimales LISP



```
    creates one q-expression with their contents.
'eval' pretends a q-expression is an s-expression and
    evaluates it normally.

'list' creates a q-expression from an s-expression.

For a detailed reference, see: https://buildyourownlisp.com/.
Thanks and credits to Daniel Holden for this brilliant resource.
```

```
eval (tail {tail tail {5 6 7}})
```

```
{ 6 7 }
```

barafael.github.io/pils/

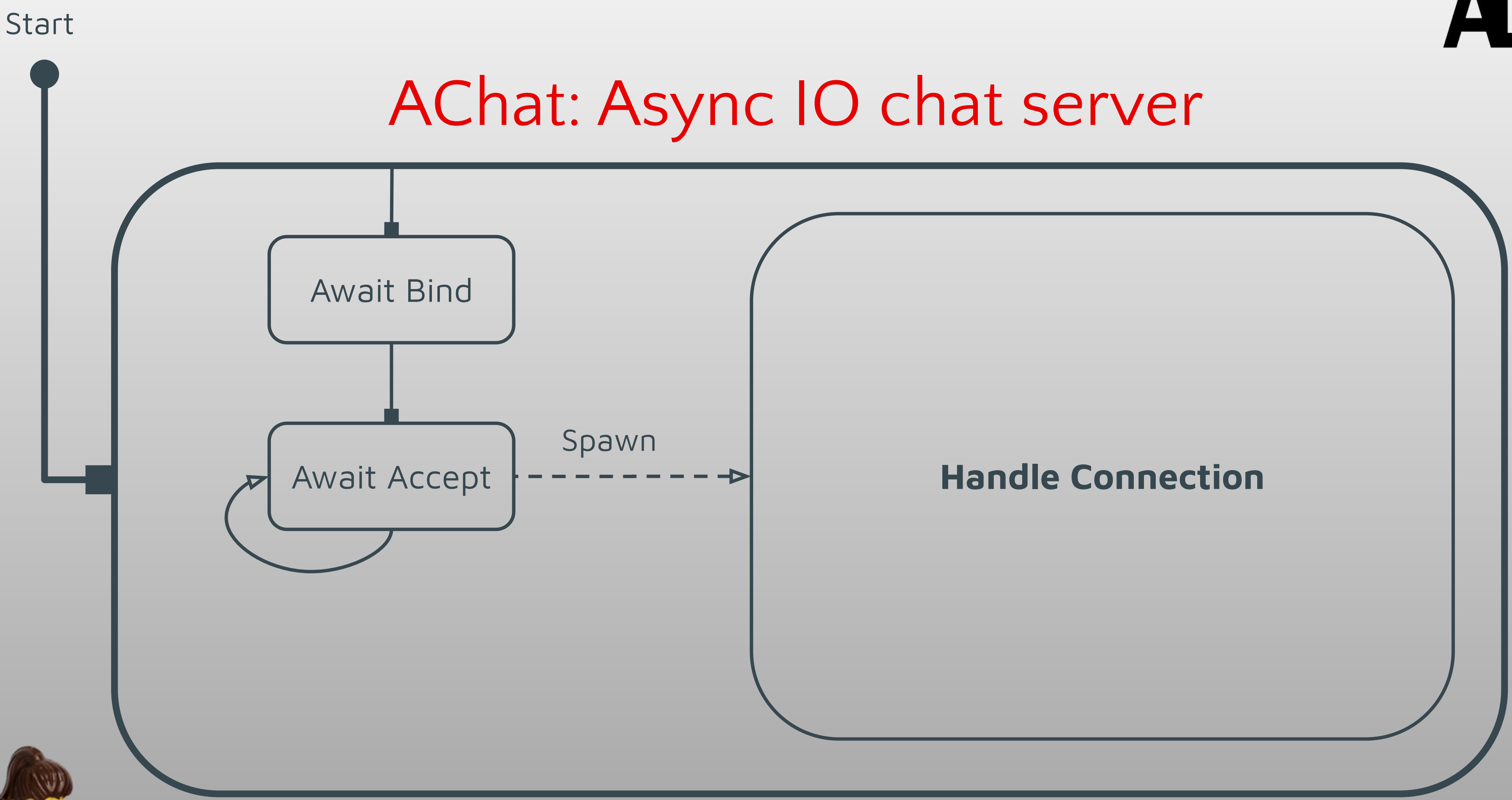
github.com/barafael/crisp

github.com/barafael/pils

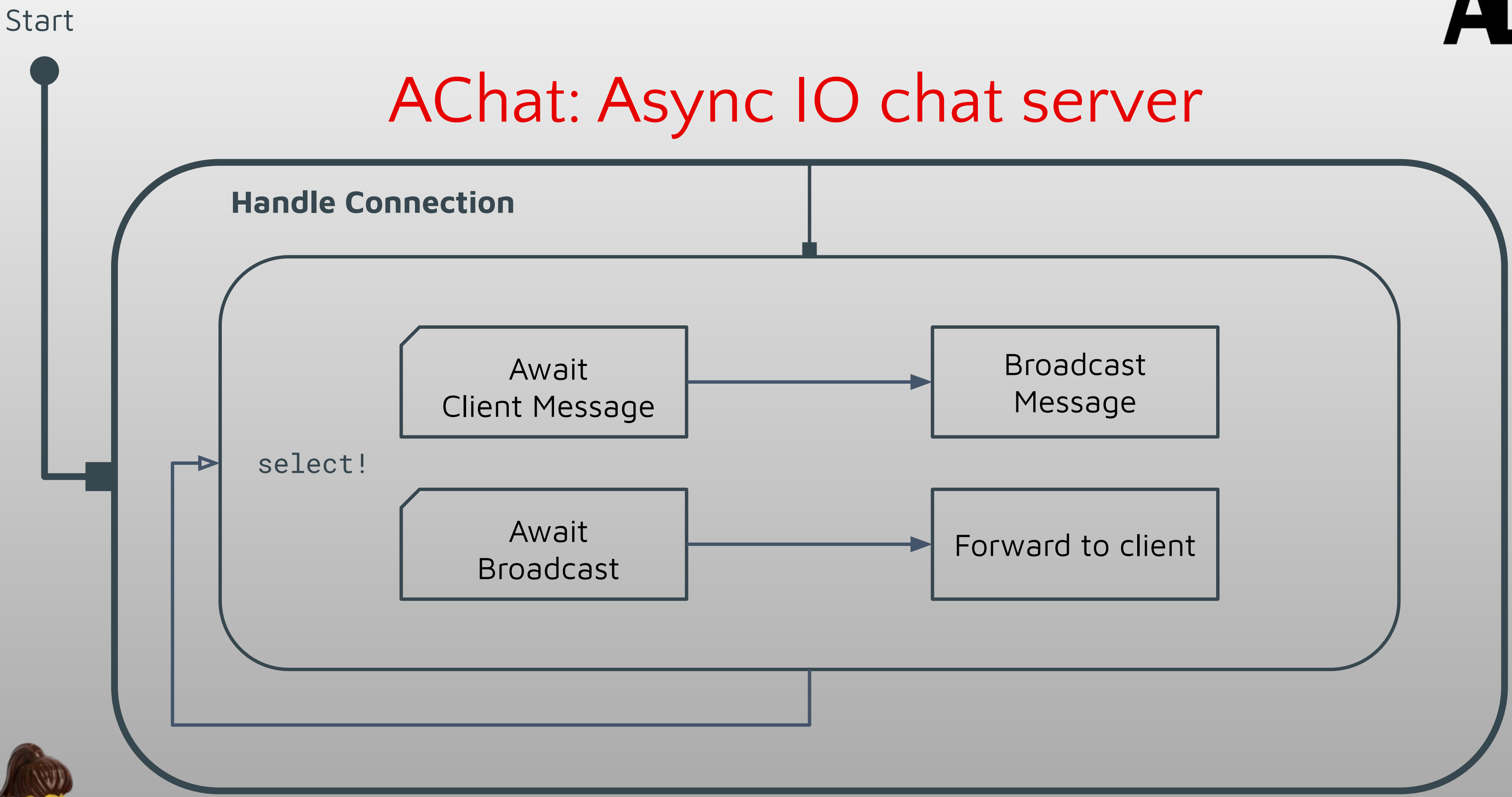
AChat: Async IO chat server

- Chat über TCP so einfach wie möglich
- `async`/`.await` mit `tokio`
- Futures, tasks, channels, `select!`/`join!`, `tokio-console`
- Weitere Beispielprogramme
  - Chat with announce
  - Collector
  - Echo

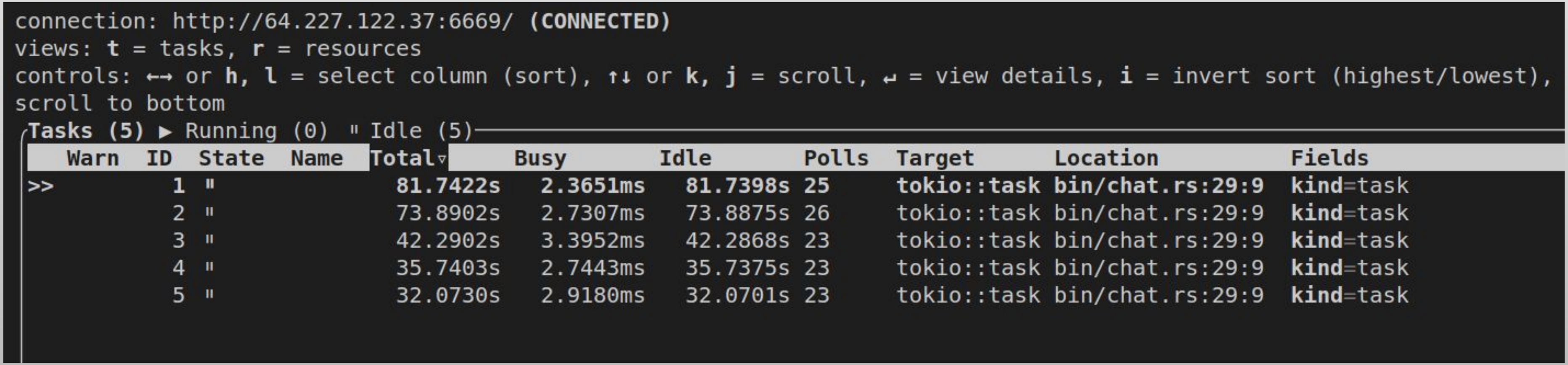
AChat: Async IO chat server



AChat: Async IO chat server

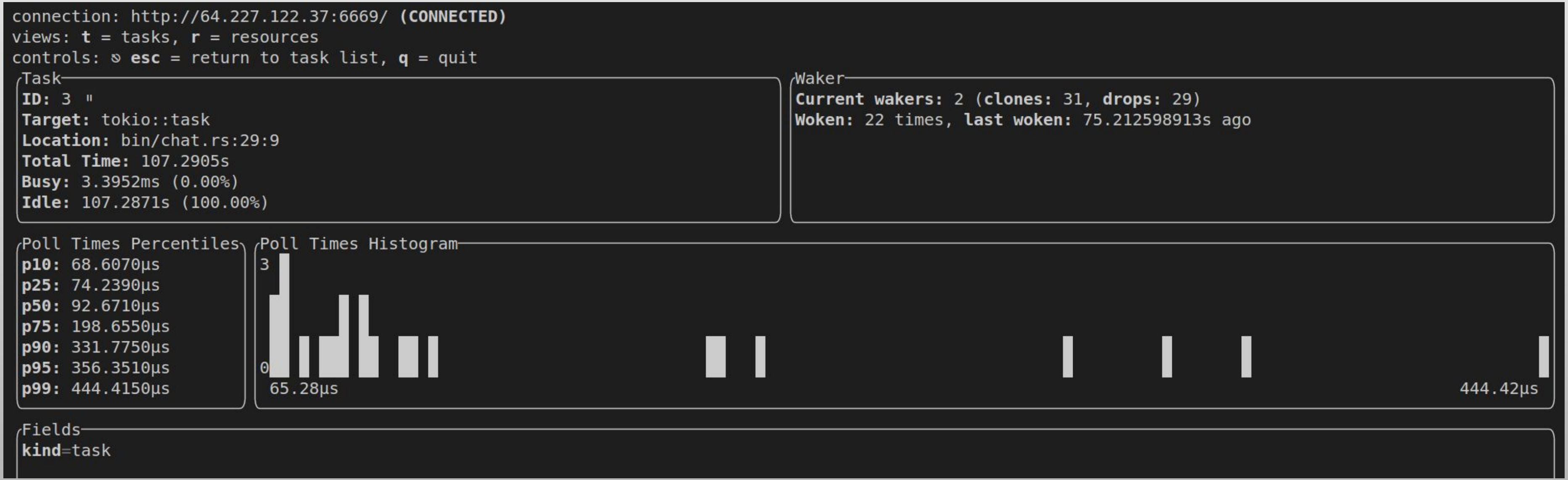


AChat: Async IO chat server



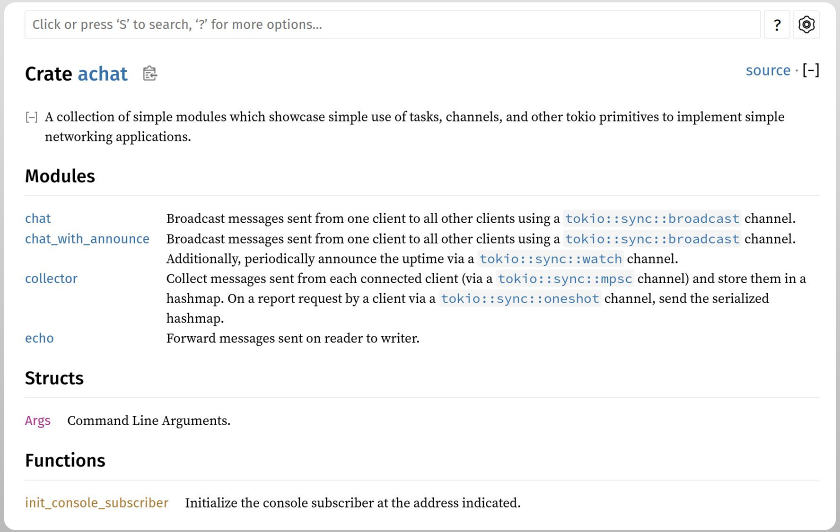
```
connection: http://64.227.122.37:6669/ (CONNECTED)
views: t = tasks, r = resources
controls: ←→ or h, l = select column (sort), ↑↓ or k, j = scroll, ↵ = view details, i = invert sort (highest/lowest),
scroll to bottom
╭Tasks (5) ▶ Running (0) ॥ Idle (5)──────────────────────────────────────────────────────────────────────────
│ Warn  ID  State  Name  Total▿     Busy        Idle      Polls  Target      Location            Fields
│>>      1  ॥            81.7422s   2.3651ms    81.7398s  25     tokio::task bin/chat.rs:29:9    kind=task
│        2  ॥            73.8902s   2.7307ms    73.8875s  26     tokio::task bin/chat.rs:29:9    kind=task
│        3  ॥            42.2902s   3.3952ms    42.2868s  23     tokio::task bin/chat.rs:29:9    kind=task
│        4  ॥            35.7403s   2.7443ms    35.7375s  23     tokio::task bin/chat.rs:29:9    kind=task
│        5  ॥            32.0730s   2.9180ms    32.0701s  23     tokio::task bin/chat.rs:29:9    kind=task
```

AChat: Async IO chat server



```
connection: http://64.227.122.37:6669/ (CONNECTED)
views: t = tasks, r = resources
controls: ⎋ esc = return to task list, q = quit
╭Task─────────────────────────────────────╮ ╭Waker──────────────────────────────────────────────╮
ID: 3 ⏸                                     Current wakers: 2 (clones: 31, drops: 29)
Target: tokio::task                         Woken: 22 times, last woken: 75.212598913s ago
Location: bin/chat.rs:29:9
Total Time: 107.2905s
Busy: 3.3952ms (0.00%)
Idle: 107.2871s (100.00%)

╭Poll Times Percentiles╮ ╭Poll Times Histogram──────────────────────────────────╮
p10: 68.6070µs           3
p25: 74.2390µs
p50: 92.6710µs
p75: 198.6550µs
p90: 331.7750µs
p95: 356.3510µs          0
p99: 444.4150µs           65.28µs                                        444.42µs

╭Fields─────────────────────────────────────────────────────────────────────────╮
kind=task
```

AChat: Async IO chat server



Click or press 'S' to search, '?' for more options...

Crate achat

source · [−]

[−] A collection of simple modules which showcase simple use of tasks, channels, and other tokio primitives to implement simple networking applications.

Modules

chat	Broadcast messages sent from one client to all other clients using a `tokio::sync::broadcast` channel.
chat_with_announce	Broadcast messages sent from one client to all other clients using a `tokio::sync::broadcast` channel. Additionally, periodically announce the uptime via a `tokio::sync::watch` channel.
collector	Collect messages sent from each connected client (via a `tokio::sync::mpsc` channel) and store them in a hashmap. On a report request by a client via a `tokio::sync::oneshot` channel, send the serialized hashmap.
echo	Forward messages sent on reader to writer.

Structs

Args	Command Line Arguments.

Functions

init_console_subscriber	Initialize the console subscriber at the address indicated.

github.com/barafael/achat

Documentation

Raspberry Pi LoRa transmitter/receiver CLI und GUI

- `#[no_std]` Treiber für Ebyte E32 LoRa Module
  - Embedded-Hal
  - Mocking + Property-Based Testing
- Deklarative CLI Definition via `clap`
- Generierte GUI mit `klask` (nutzt `clap`)
- Cross-Compilation für Raspberry Pi mit `cross` (Docker)

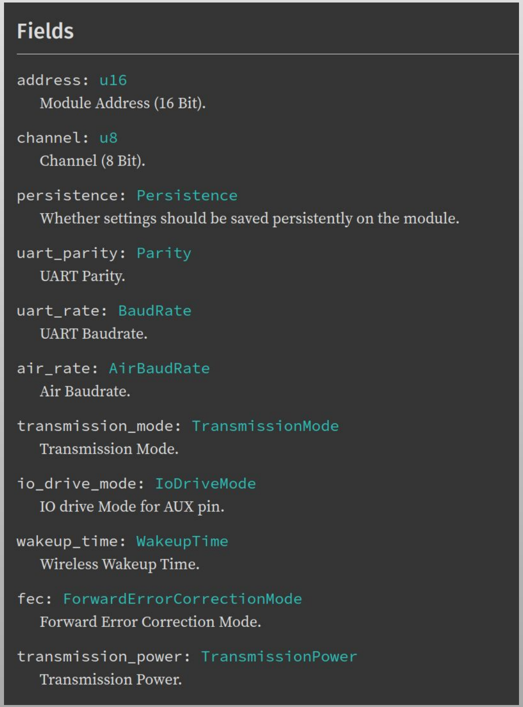
```
cross build --target armv7-unknown-linux-musleabihf
```

ADC++

Raspberry Pi LoRa transmitter/receiver CLI

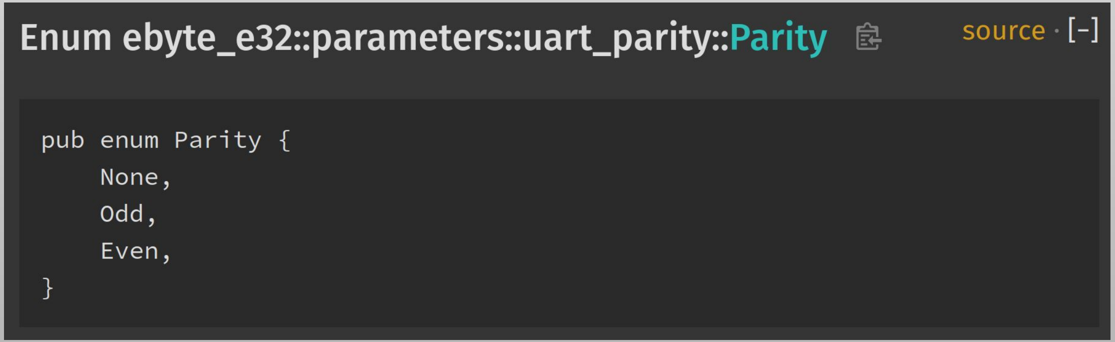
```rust
#[derive(Clone, Debug, PartialEq, Eq, Parser)]
#[clap(author, version, about, long_about = None)]
pub struct App {
    /// Module Address (16 Bit).
    #[clap(short, long, required = true)]
    pub address: u16,

    /// Whether settings should be saved persistently on the module.
    #[clap(arg_enum, long, required = false, ignore_case(true), default_value_t)]
    pub persistence: Persistence,
}
```

Raspberry Pi LoRa transmitter/receiver CLI



Fields

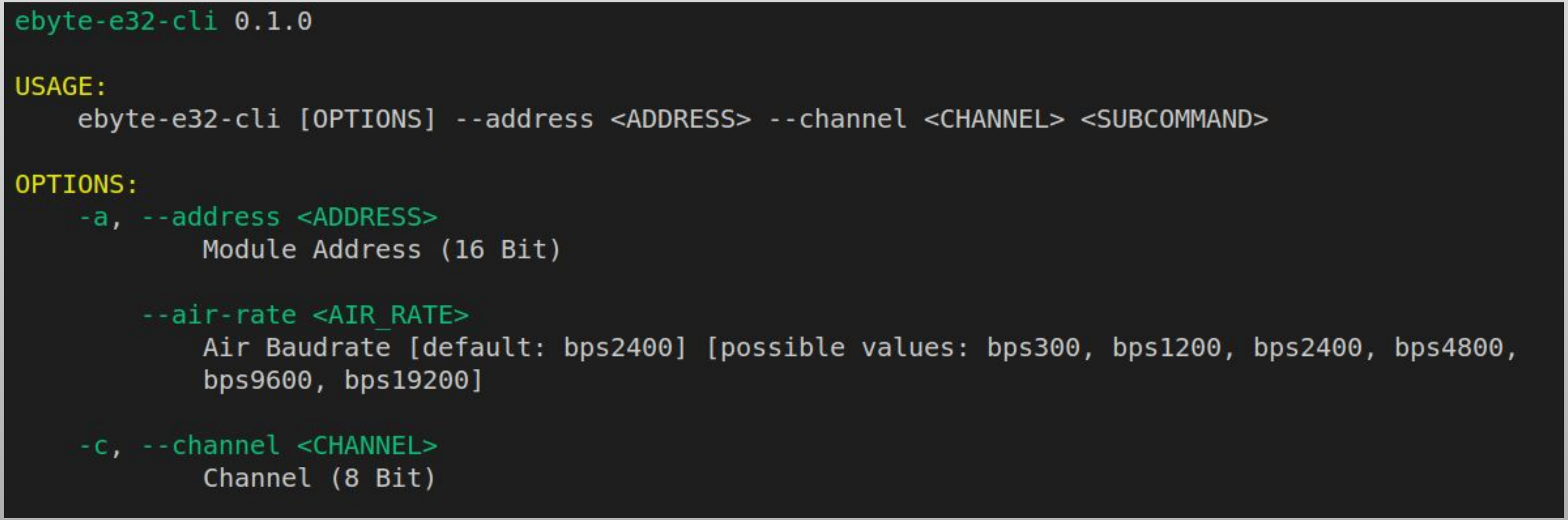
```
address: u16
    Module Address (16 Bit).

channel: u8
    Channel (8 Bit).

persistence: Persistence
    Whether settings should be saved persistently on the module.

uart_parity: Parity
    UART Parity.

uart_rate: BaudRate
    UART Baudrate.

air_rate: AirBaudRate
    Air Baudrate.

transmission_mode: TransmissionMode
    Transmission Mode.

io_drive_mode: IoDriveMode
    IO drive Mode for AUX pin.

wakeup_time: WakeupTime
    Wireless Wakeup Time.

fec: ForwardErrorCorrectionMode
    Forward Error Correction Mode.

transmission_power: TransmissionPower
    Transmission Power.
```



Enum ebyte_e32::parameters::uart_parity::Parity source · [−]

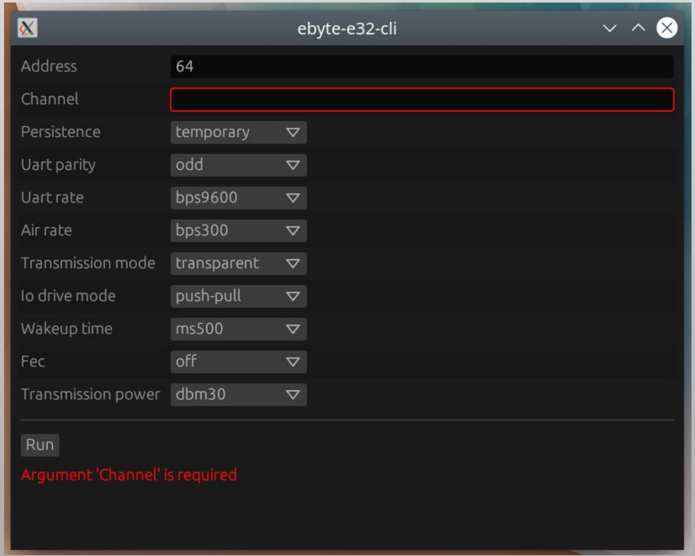
```
pub enum Parity {
    None,
    Odd,
    Even,
}
```

Raspberry Pi LoRa transmitter/receiver CLI



```
ebyte-e32-cli 0.1.0

USAGE:
    ebyte-e32-cli [OPTIONS] --address <ADDRESS> --channel <CHANNEL> <SUBCOMMAND>

OPTIONS:
    -a, --address <ADDRESS>
            Module Address (16 Bit)

        --air-rate <AIR_RATE>
            Air Baudrate [default: bps2400] [possible values: bps300, bps1200, bps2400, bps4800,
            bps9600, bps19200]

    -c, --channel <CHANNEL>
            Channel (8 Bit)
```

Raspberry Pi LoRa transmitter/receiver GUI

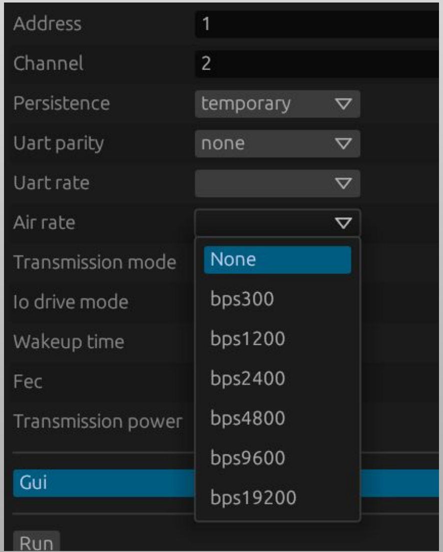


ebyte-e32-cli

Address	64
Channel	
Persistence	temporary
Uart parity	odd
Uart rate	bps9600
Air rate	bps300
Transmission mode	transparent
Io drive mode	push-pull
Wakeup time	ms500
Fec	off
Transmission power	dbm30

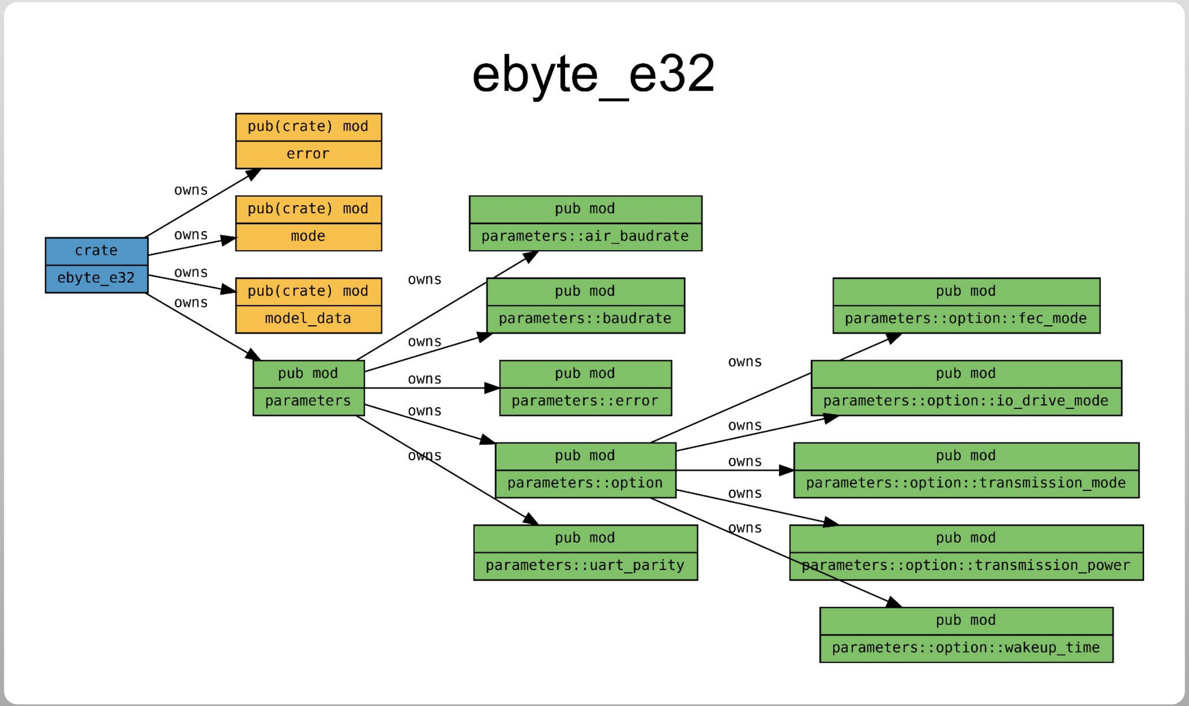
Run

Argument 'Channel' is required

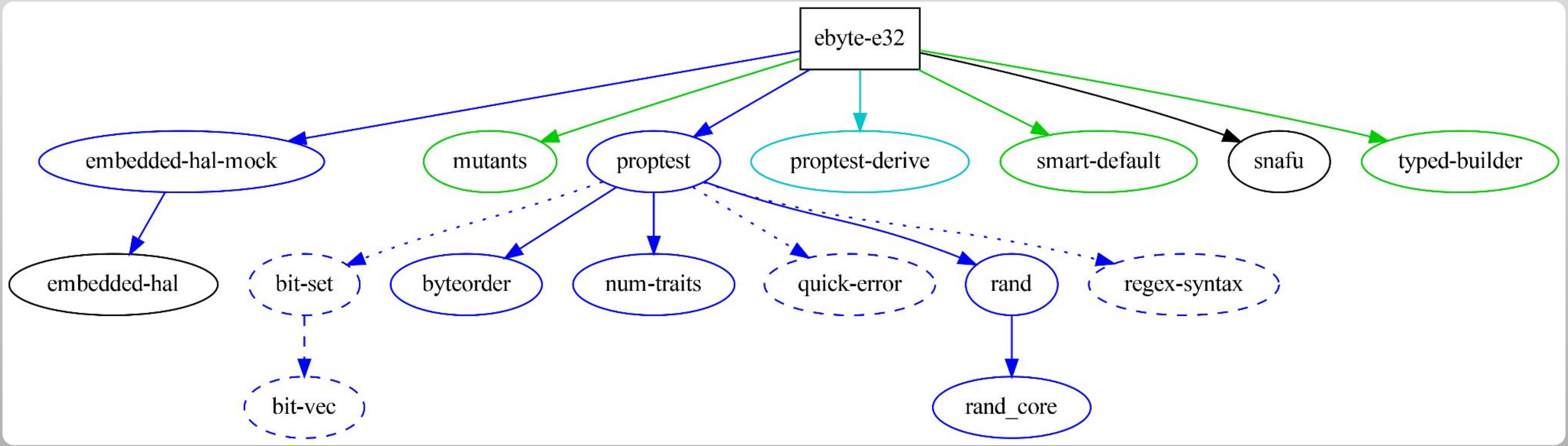


Address	1
Channel	2
Persistence	temporary
Uart parity	none
Uart rate	
Air rate	
Transmission mode	
Io drive mode	
Wakeup time	
Fec	
Transmission power	

Raspberry Pi LoRa transmitter/receiver



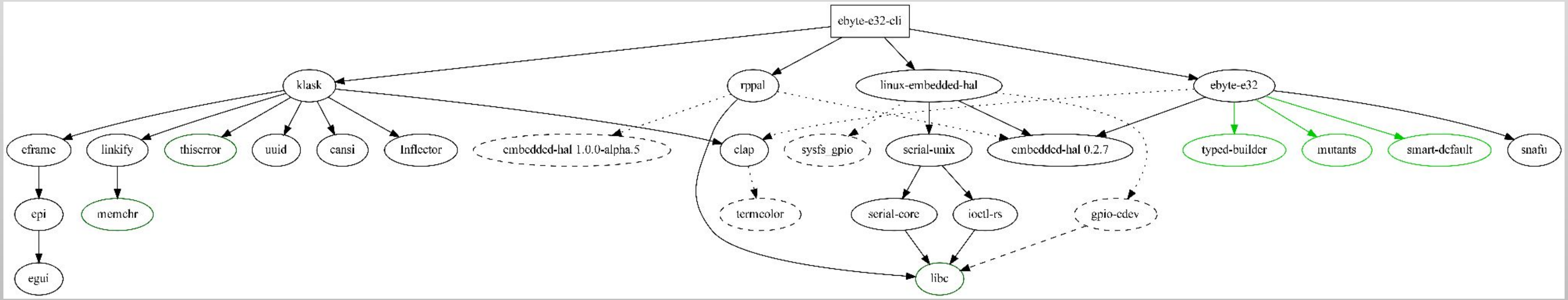
Raspberry Pi LoRa transmitter/receiver



Raspberry Pi LoRa transmitter/receiver

```
macro_rules! impl_mode {
    ($($type: ty)*, $id: literal, $m0_state: path, $m1_state: path) => {
        $(
            impl Mode for $type {
                fn id(&self) -> u8 { $id }
                fn set_pins<Aux, M0, M1, D>(...) { ... }
        )*
    };
}

impl_mode!(Normal, 0, Low, Low);
...
impl_mode!(Program, 3, High, High);
```

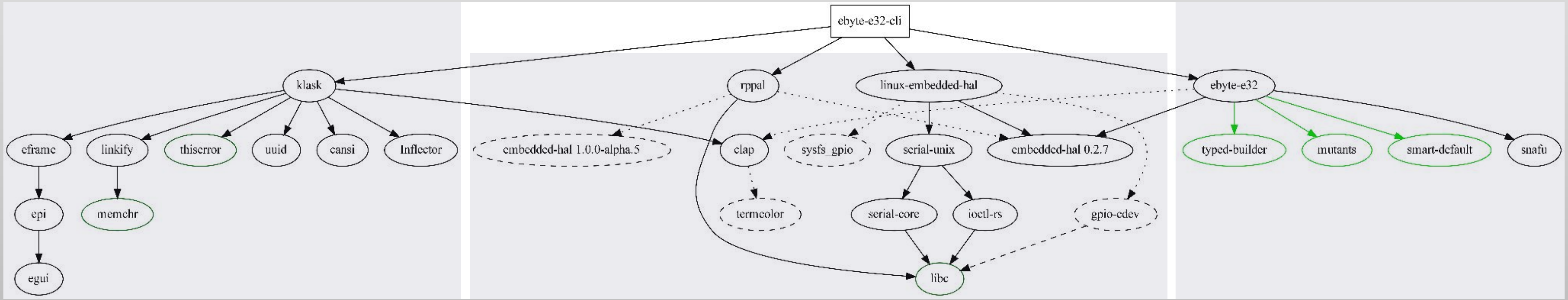
Raspberry Pi LoRa transmitter/receiver



github.com/barafael/ebyte-e32-rs

github.com/barafael/ebyte-e32-ui

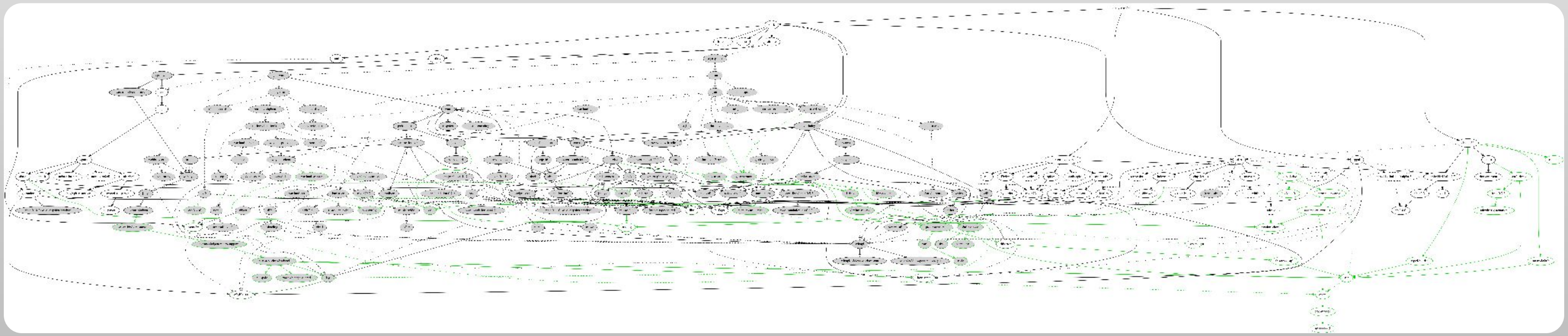
Raspberry Pi LoRa transmitter/receiver



github.com/barafael/ebyte-e32-rs

github.com/barafael/ebyte-e32-ui

Raspberry Pi LoRa transmitter/receiver



github.com/barafael/ebyte-e32-rs

github.com/barafael/ebyte-e32-ui

Email parser in WebAssembly

- Basierend auf stalwartlabs/mail-parser

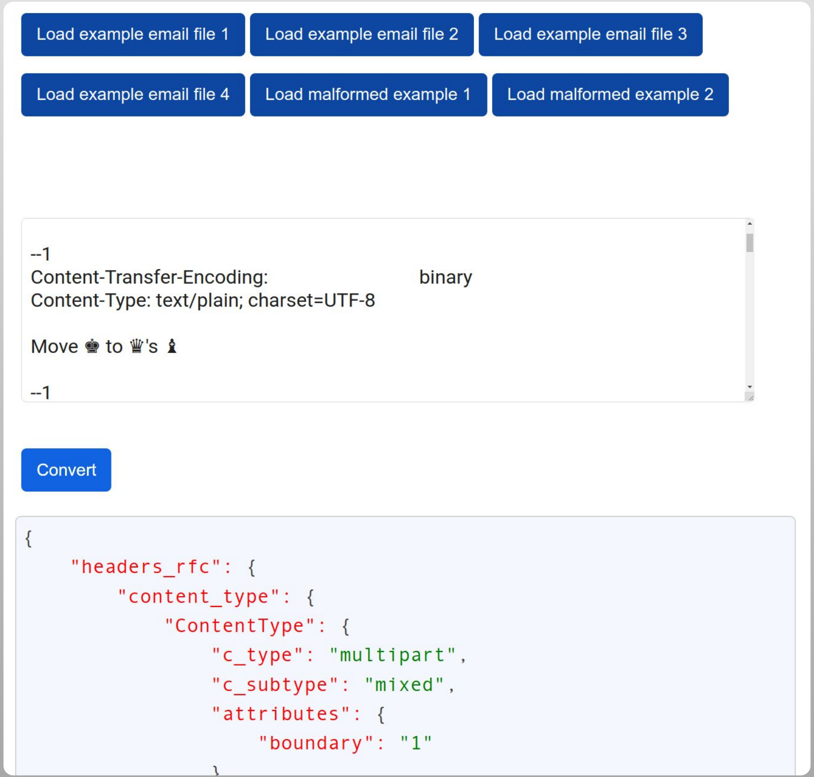
  (Email Parsing ist ein chaotisches Schlamassel)
- CLI Interface on top: barafael/mail2json
- WASM cross compilation: barafael/mail2json-web

  Web Demo hier: barafael.github.io/mail2json-web

Email parser in WebAssembly

```rust
#[wasm_bindgen]
pub fn convert(input: &str) -> String {
    let message = Message::parse(input.as_bytes());
    serde_json::to_string_pretty(&message).unwrap_or_default()
}
```

Email parser in WebAssembly


Load example email file 1
Load example email file 2
Load example email file 3
Load example email file 4
Load malformed example 1
Load malformed example 2
Convert

```
--1
Content-Transfer-Encoding:                    binary
Content-Type: text/plain; charset=UTF-8

Move ♚ to ♛'s ♝

--1
```

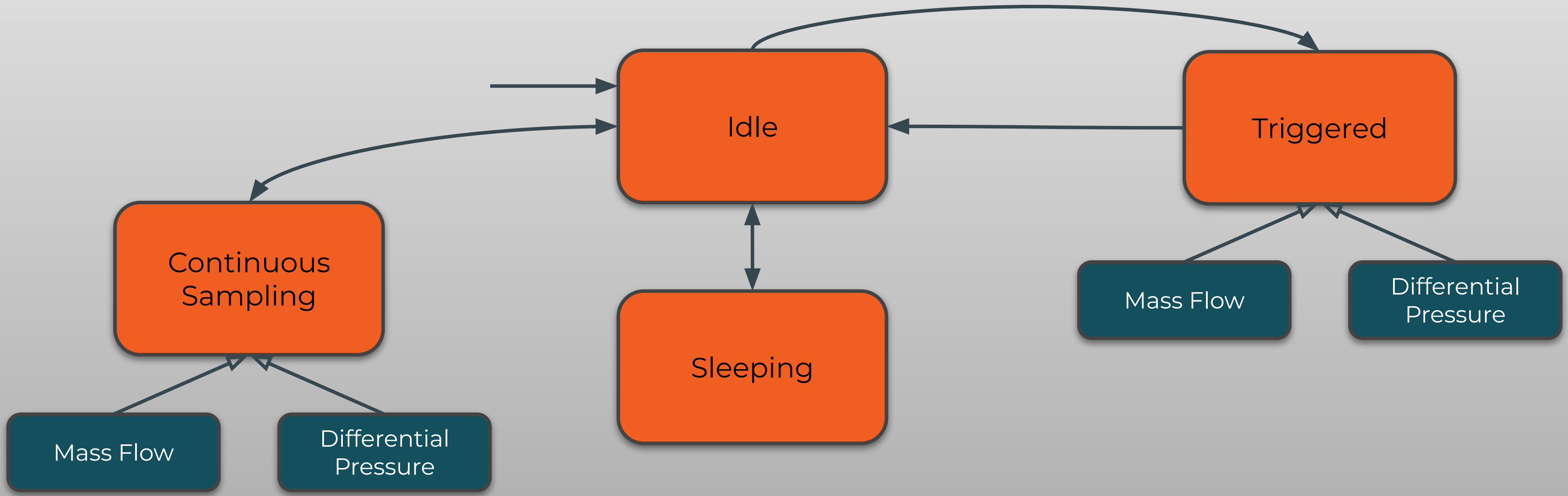
```
{
    "headers_rfc": {
        "content_type": {
            "ContentType": {
                "c_type": "multipart",
                "c_subtype": "mixed",
                "attributes": {
                    "boundary": "1"
```

Treiber für SDP8xx Differential Pressure Sensor

- Niedlicher kleiner I2C-basierter Drucksensor
- Treiber basiert auf embedded-hal
- Type-State Style für Zustandsautomaten
- Property-Based Testing im Treiber Code
- Mocking mit embedded-hal-mock

Beispiel hier

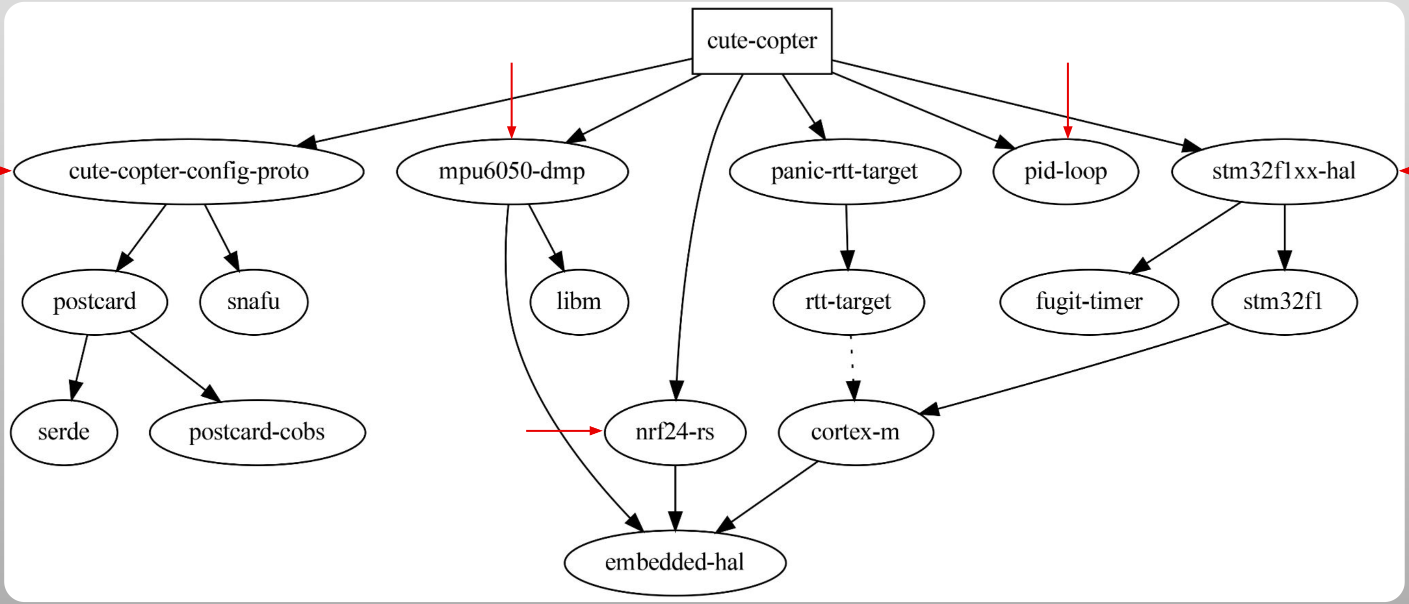
Treiber für SDP8xx Differential Pressure Sensor



CuteCopter: Minimale Quadkopter-Firmware

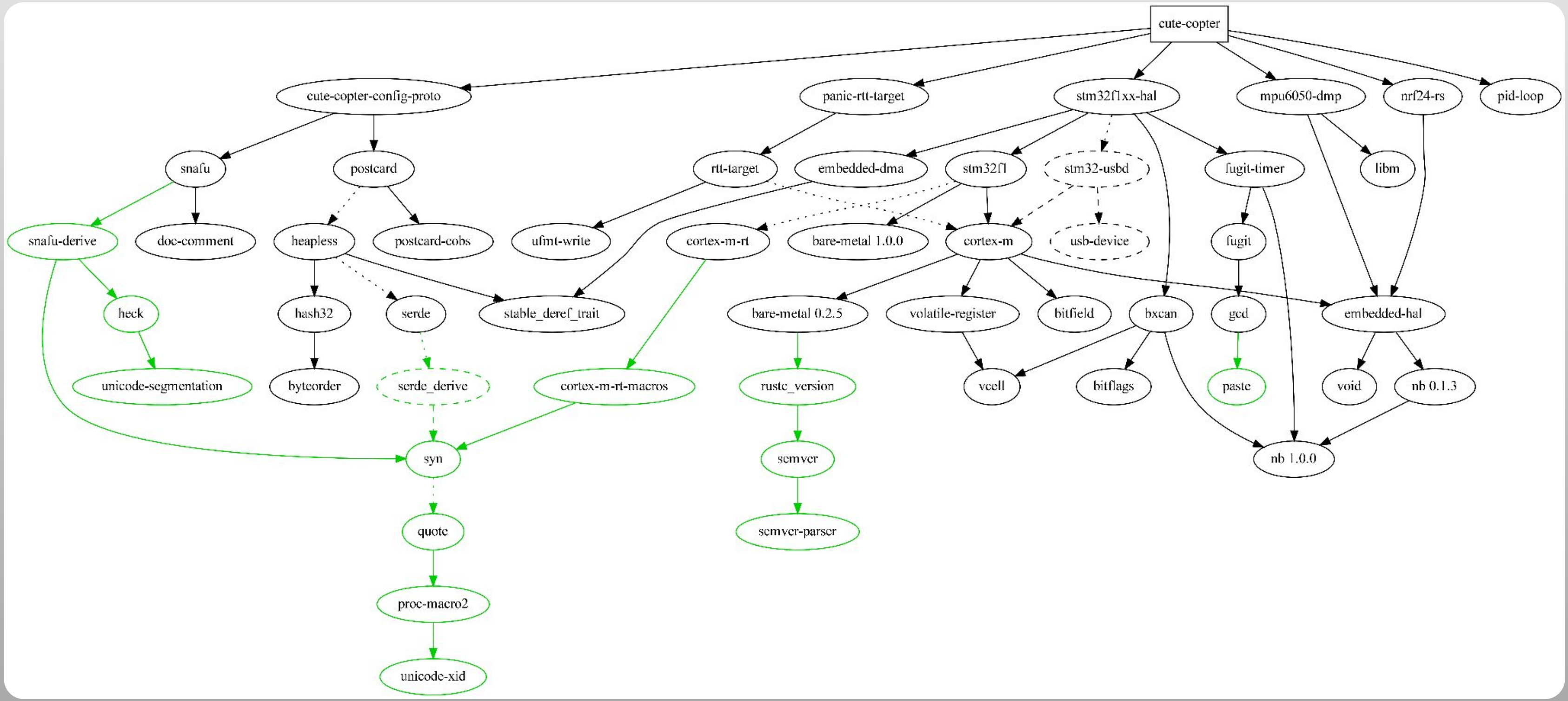
- Günstiger PCB Copter Frame: AliExpress
  - Samt PCB Controller + Transmitter
- Definitiv keine Premium-Komponenten:
  - STM32F103, MPU6050, NRF24
- Keine Dokumentation, chinesische PCB Markings
- Reverse Engineering mit Oszi und Multimeter
- Rust Firmware für Copter und Sender

CuteCopter: Minimale Quadkopter-Firmware



(Vereinfacht)

CuteCopter: Minimale Quadkopter-Firmware



Let's Vote!