Realzeitsystem: Robocar

Joshua Rutschmann AIN

Gliederung

- Hardware
- Systemsoftware
- Rust
- Systementwurf
- Realzeitnachweis (hinreichend)
- Realzeitnachweis (notwendig)
- Fazit

Hardware

- Selbe Hardware wie alle:
  - Raspberry Pi 3B v1.2
  - H-Brücke
  - HC-SR04 Ultraschall Module
  - Infrarotsensoren
  - ...
- Eigene Änderungen:
  - Zwei Leitungen pro Ultraschall Echo Pin (wegen Interrupts)
  - RC522 RFID Lesegerät mit **originalem** MRFC522 Chip

Die Systemsoftware

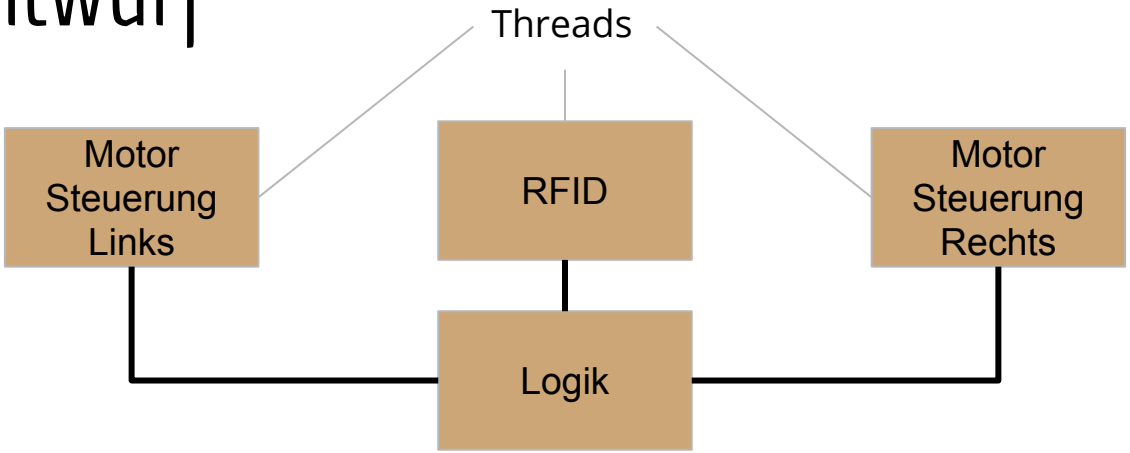
- DietPi mit 4.14 Realtime Kernel
  - Performance Governor
  - Keine dynamischen idle ticks
- Rust + Cross-Compiling
- Pullups und PWM Aktivierung über Device Tree
- So viel wie möglich in Kernelmodulen
  - So viele Interrupts wie möglich
  - Kein sysfs Zugriff

Rust

Abhängigkeiten

- rust_gpiozero
  - Um Infrarotsensoren auszulesen
- nix
  - Um Signale zu behandeln
- mfrc522 & linux-embedded-hal
  - RFIDs lesen
- libc
  - Zeiten messen, CPU pinning & scheduler setzen

Systementwurf



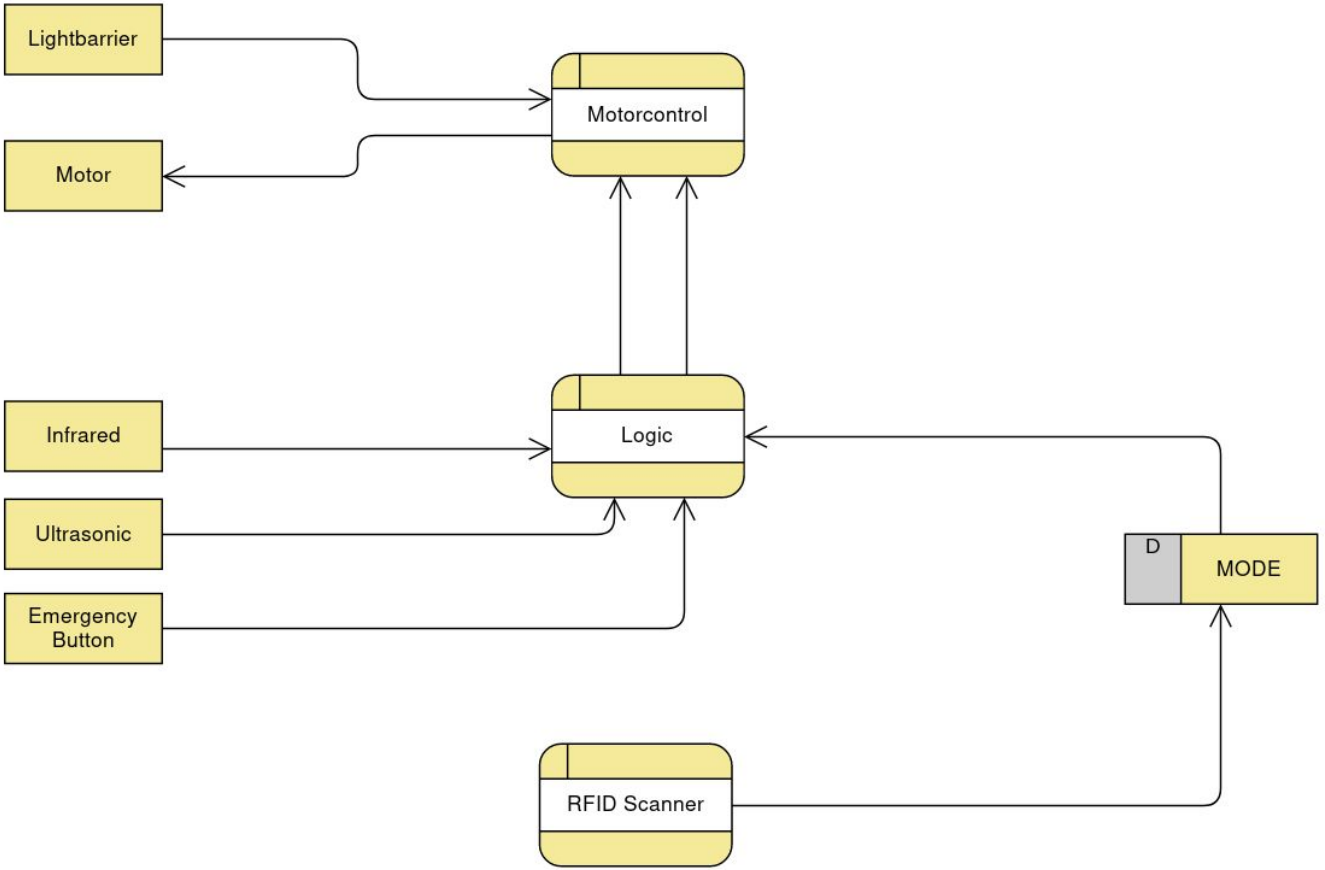
Rust

C

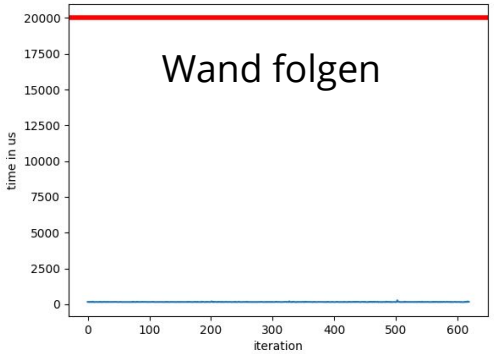
Kernelmodule

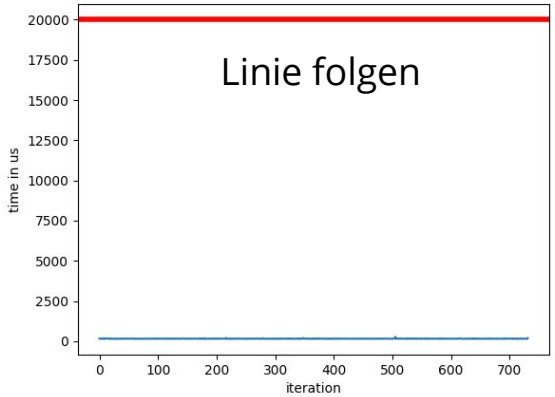
Motor	Ultraschall	Lichtschranke	Not-Aus
Hardware PWM	Interrupts	Interrupts	Interrupts

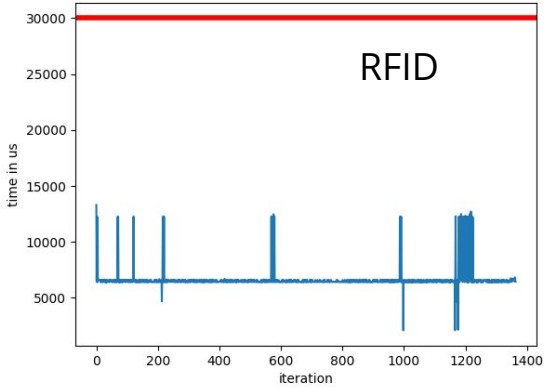
Datenfluss

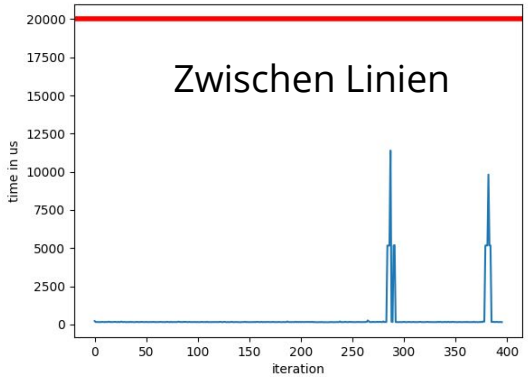

Lightbarrier
Motor
Motorcontrol
Infrared
Ultrasonic
Emergency Button
Logic
D
MODE
RFID Scanner

Ohne Last - nohup & ~~ssh~~

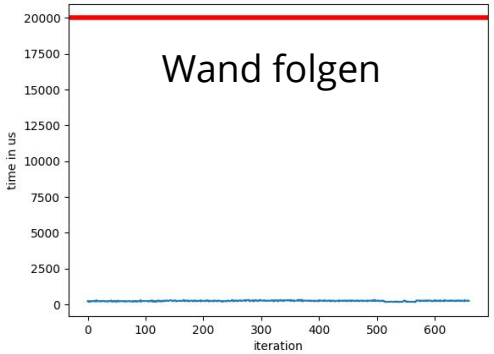


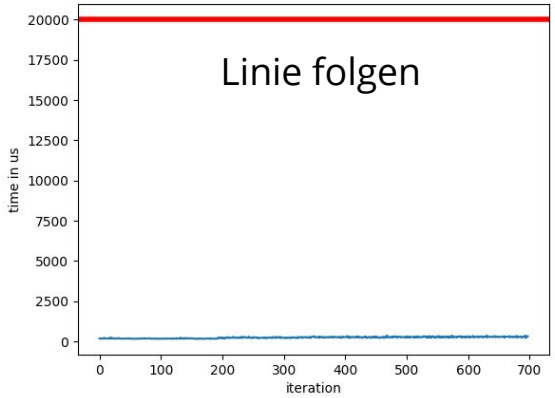


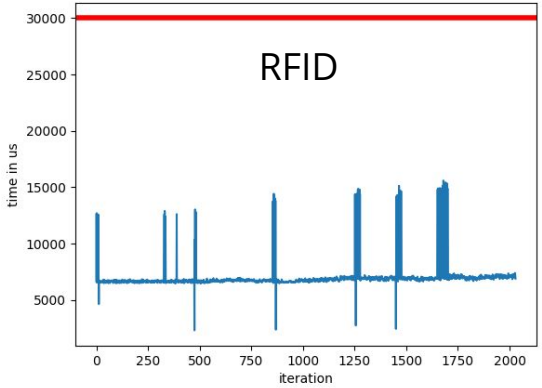


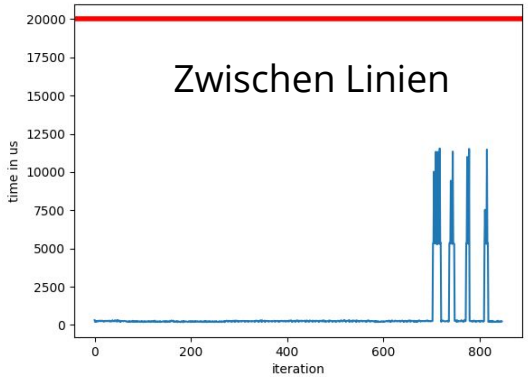


Mit Last - scp copy & cpu burn & io burn

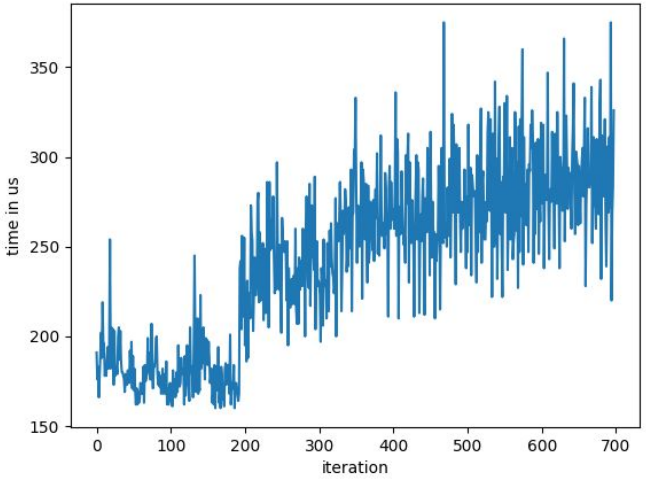




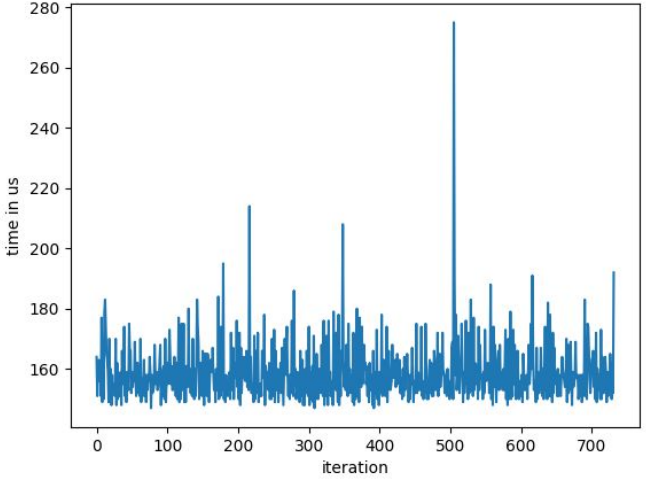




Beispiel: Linie folgen



Mit Last



Ohne Last

Realzeitnachweis (hinreichend)

Task	**t_{Pmin} / t_{Dmax}**	**t_{Emax}**	**t_{Emin}**	**Auslastung**
Motor	100ms	0,295ms	0,091ms	0,295%
RFID	30ms	15,771ms	2,295ms	52,57%
Linie folgen	20ms	0,381ms	0,262ms	1,905%
Wand folgen	20ms	0,544ms	0,152ms	2,72%
Zwischen Linien	20ms	5,523ms	5,177ms	27,615%

Für 1 Prozessorsystem:

0,295% + 52,57% + 1,905% + 2,72% + 27,615% = 85,105%
85,105% $\lessdot$ 73,5%

Realzeitnachweis (notwendig)

Taskset	Maximale Reaktionszeit	Maximal zulässig sind
RFID, Follow Line, Motorcontrol	16,447ms	20ms
RFID, Follow Wall, Motorcontrol	16,610ms	20ms
RFID, Between Lines, Motorcontrol	27,112ms	20ms

Fazit

Pro

- Geringe Auslastung
- Bis auf Infrarotsensoren alles Interrupts
- Logik + MODE Variable = flexibel
- Rust 👍

Verbesserungen

- Infrarot in KernelModule
- P- oder PID-Regler implementieren
- Regler in Kernelmodul (kein HR-Timer)
- Relatives Schlafen