


# Introduction of IoT lab

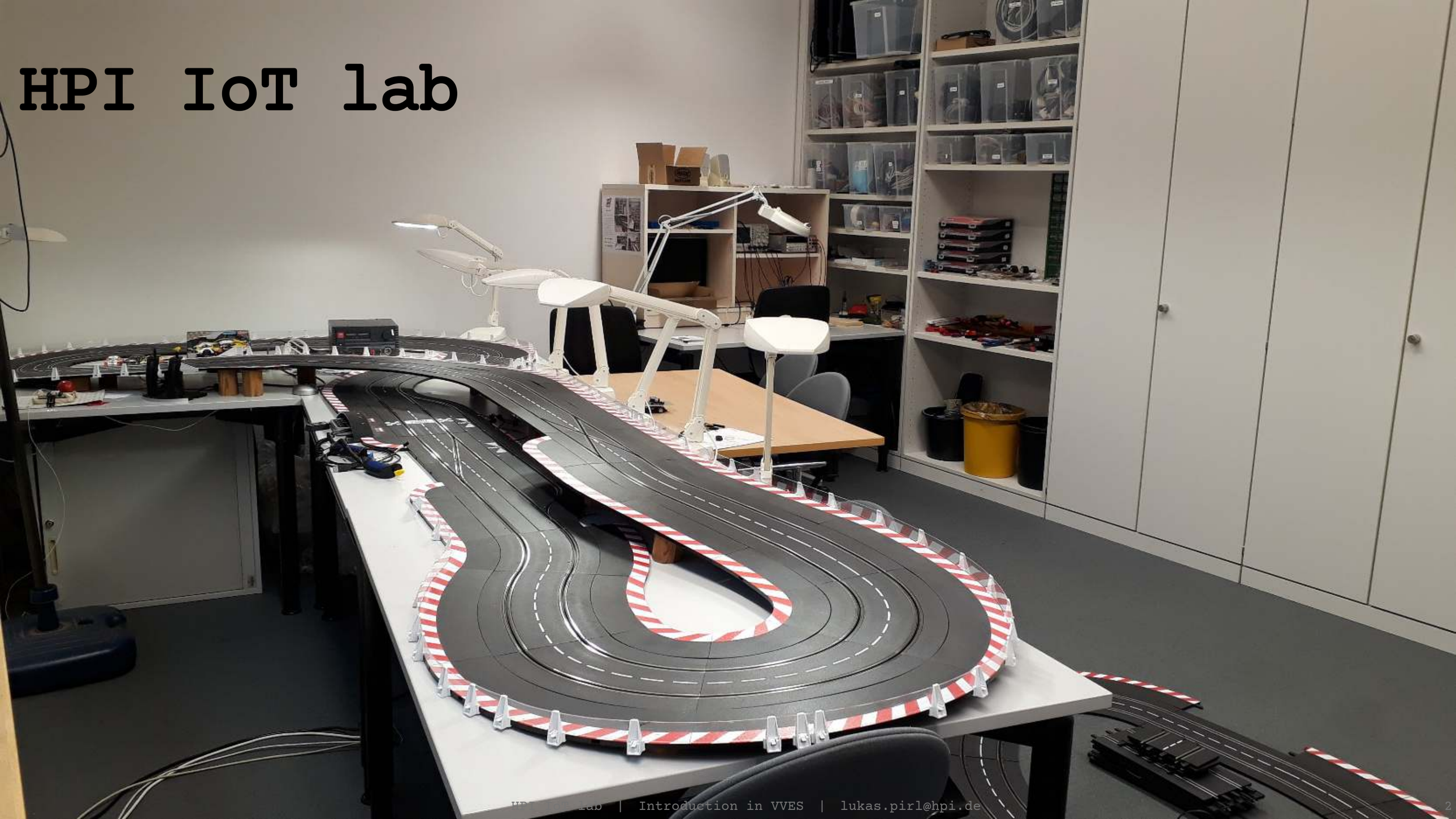
Lecture on Distributed Dependable Embedded Systems

Professorship for Operating Systems and Middleware of Prof. Andreas Polze

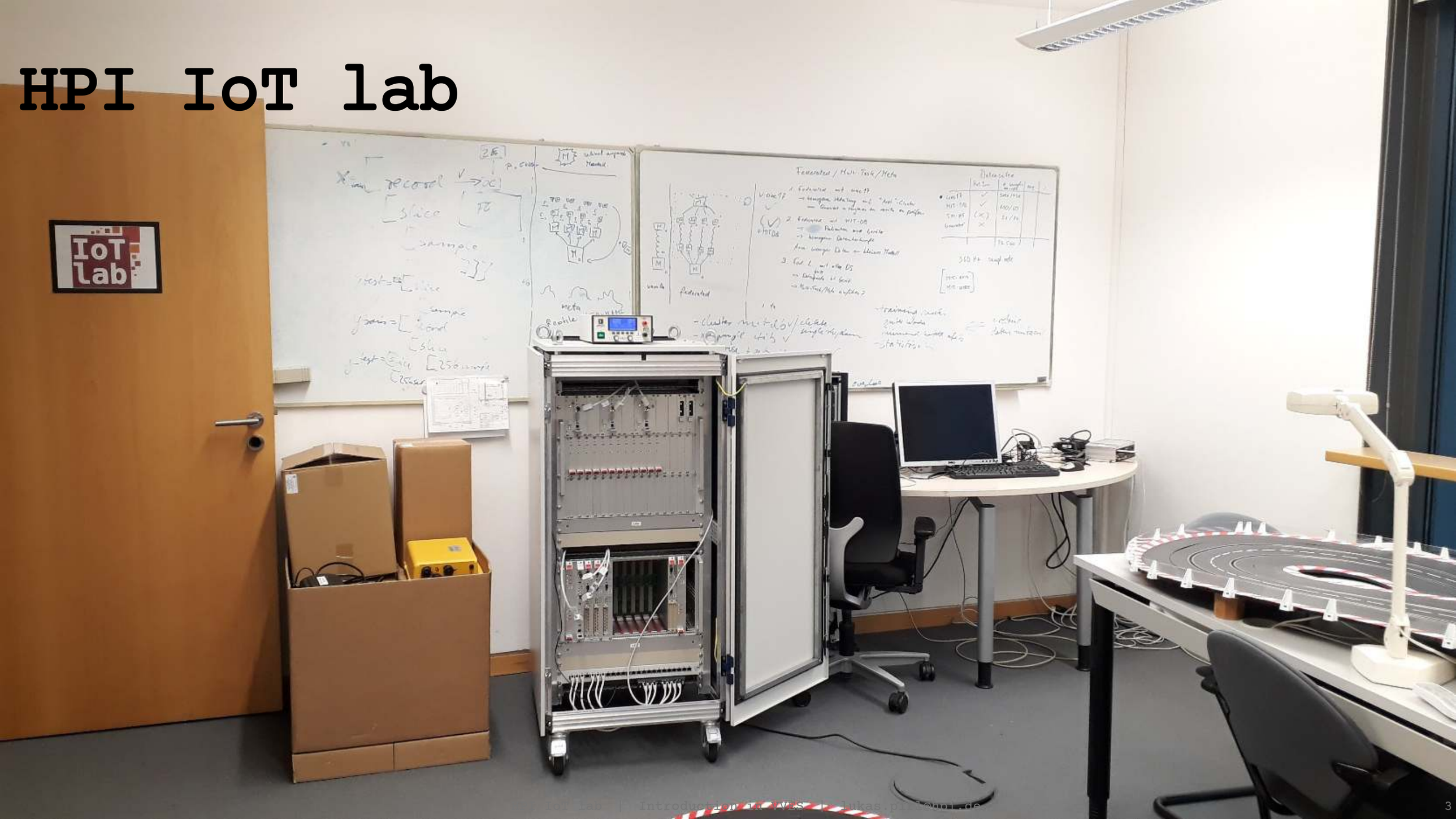
Hasso Plattner Institute, University of Potsdam

Lukas Pirl (he/him)

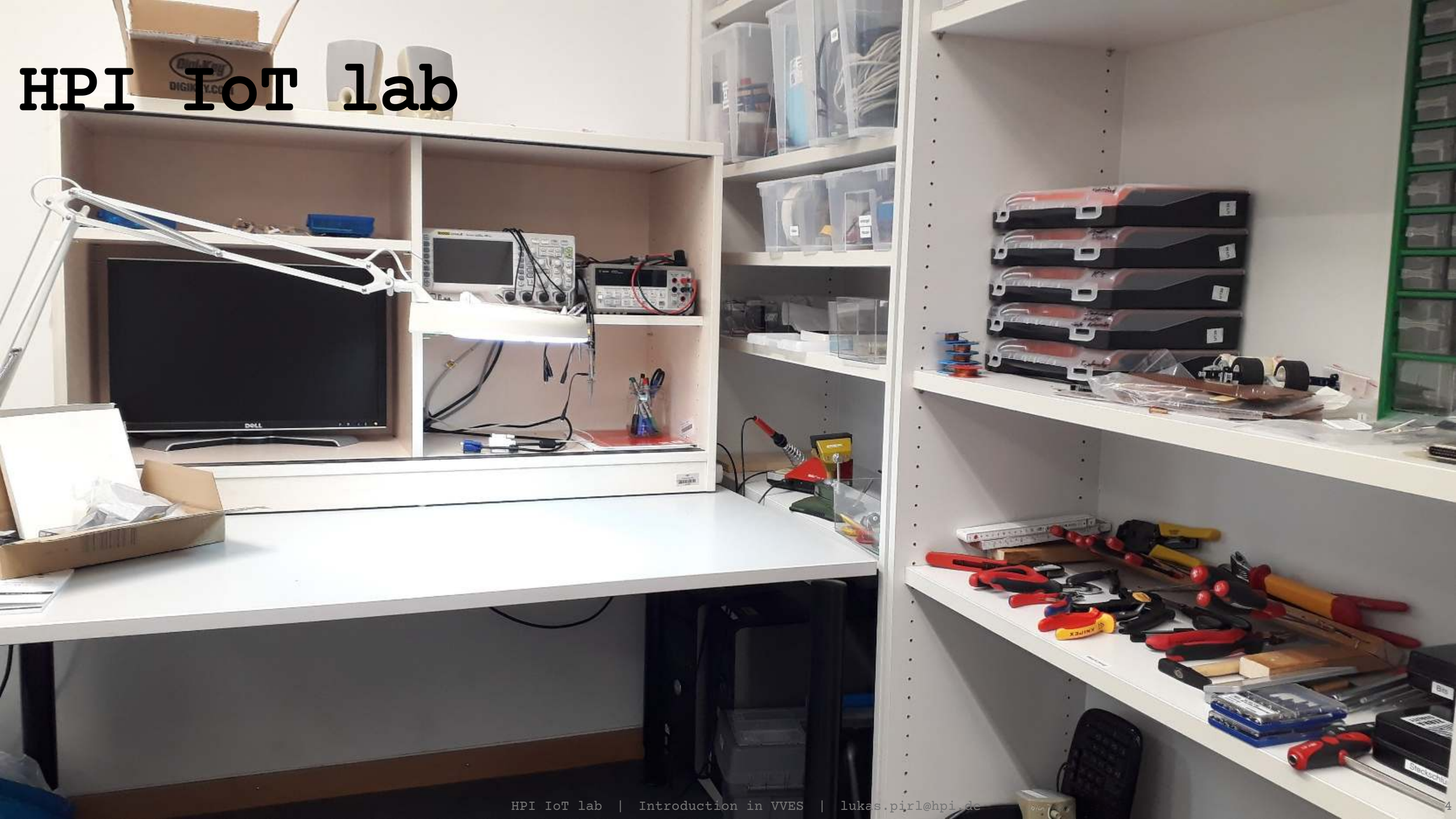
# HPI IoT lab



# HPI IoT lab



# HPI IoT lab



# workshop




general tools

soldering

electronics components

bulk & DIY cables, connectors, etc.

power supply

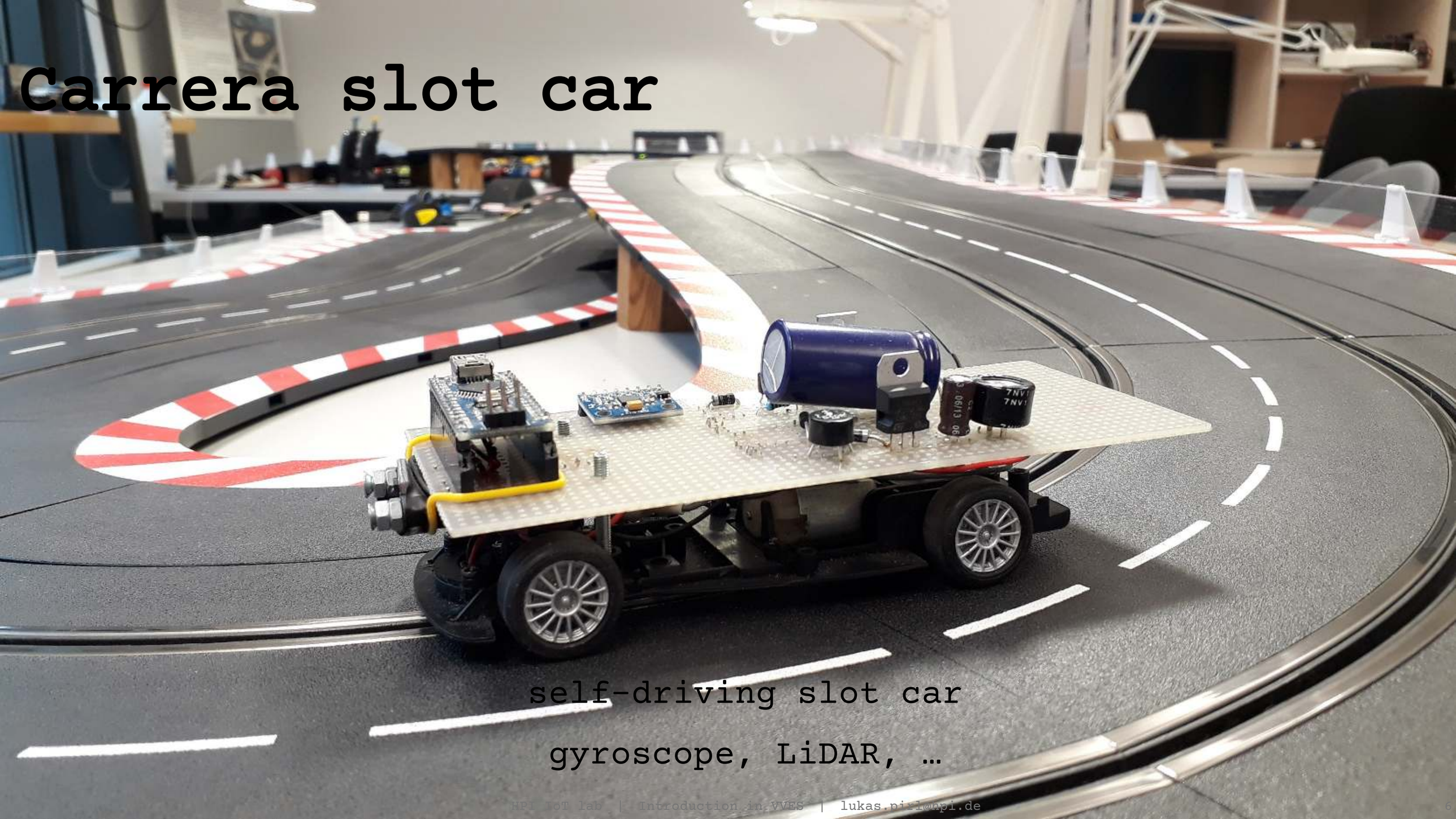
measurement

…






# Carrera slot car

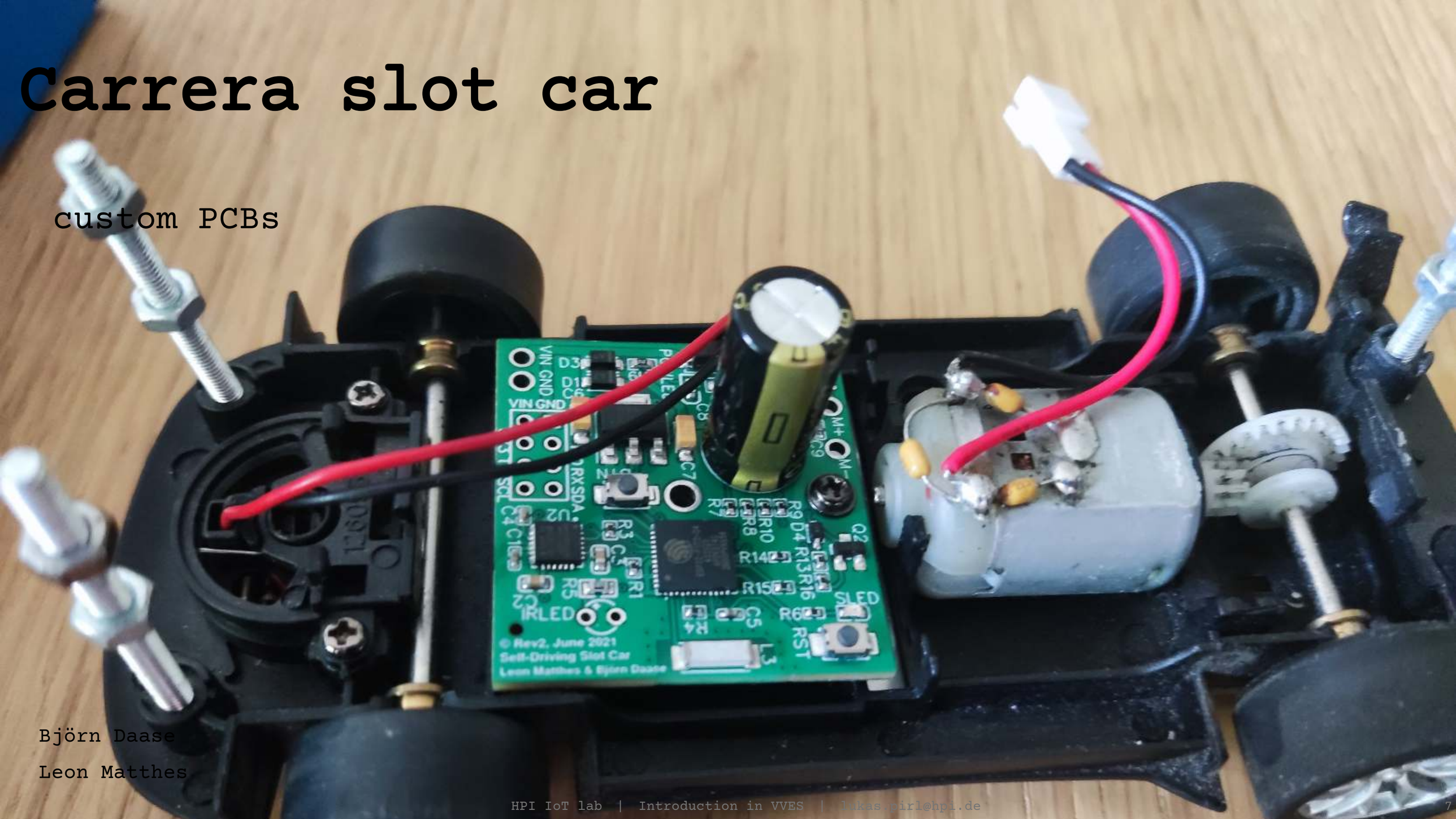


self-driving slot car

gyroscope, LiDAR, …

# Carrera slot car

custom PCBs




Björn Daase

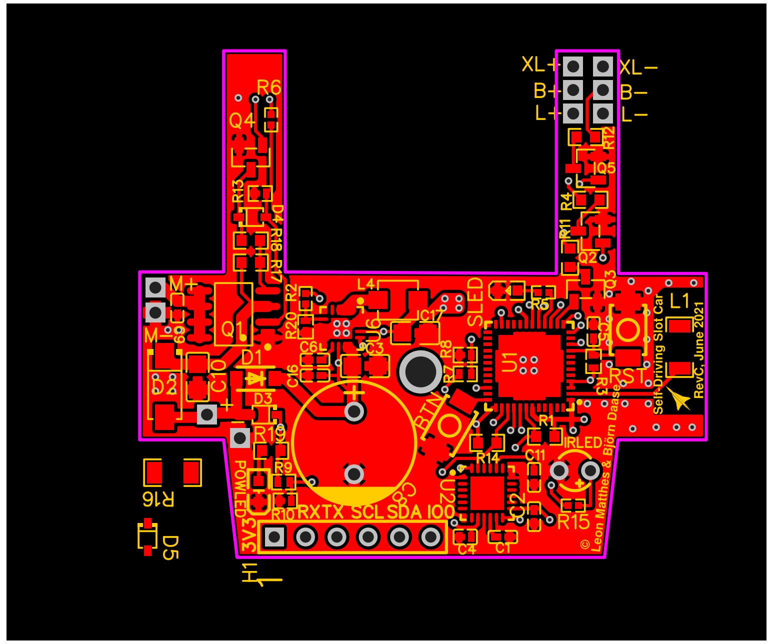
Leon Matthes

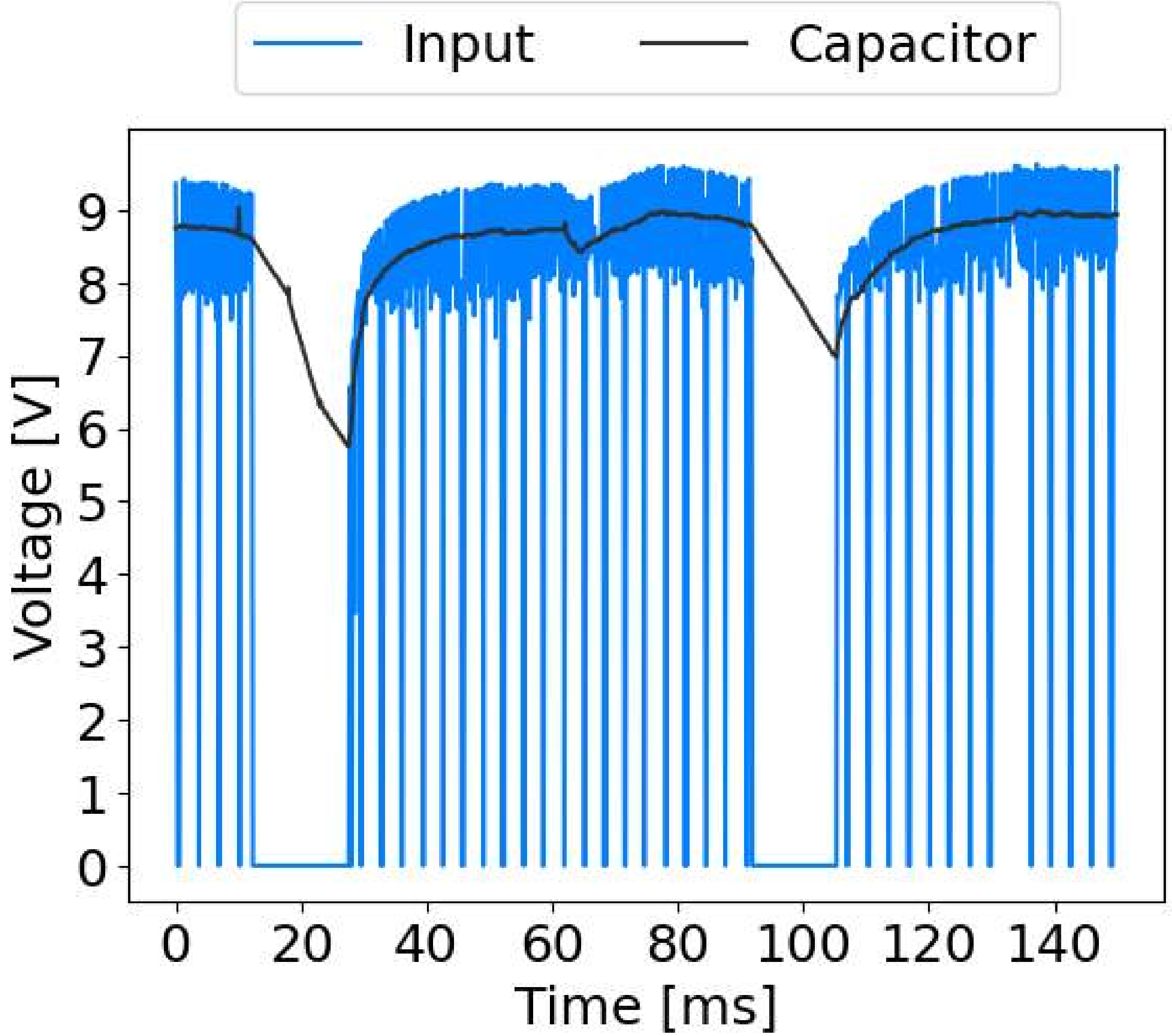
# Carrera slot car

challenges

- unusual power supply characteristics
- remote/mobile debugging

PCB design



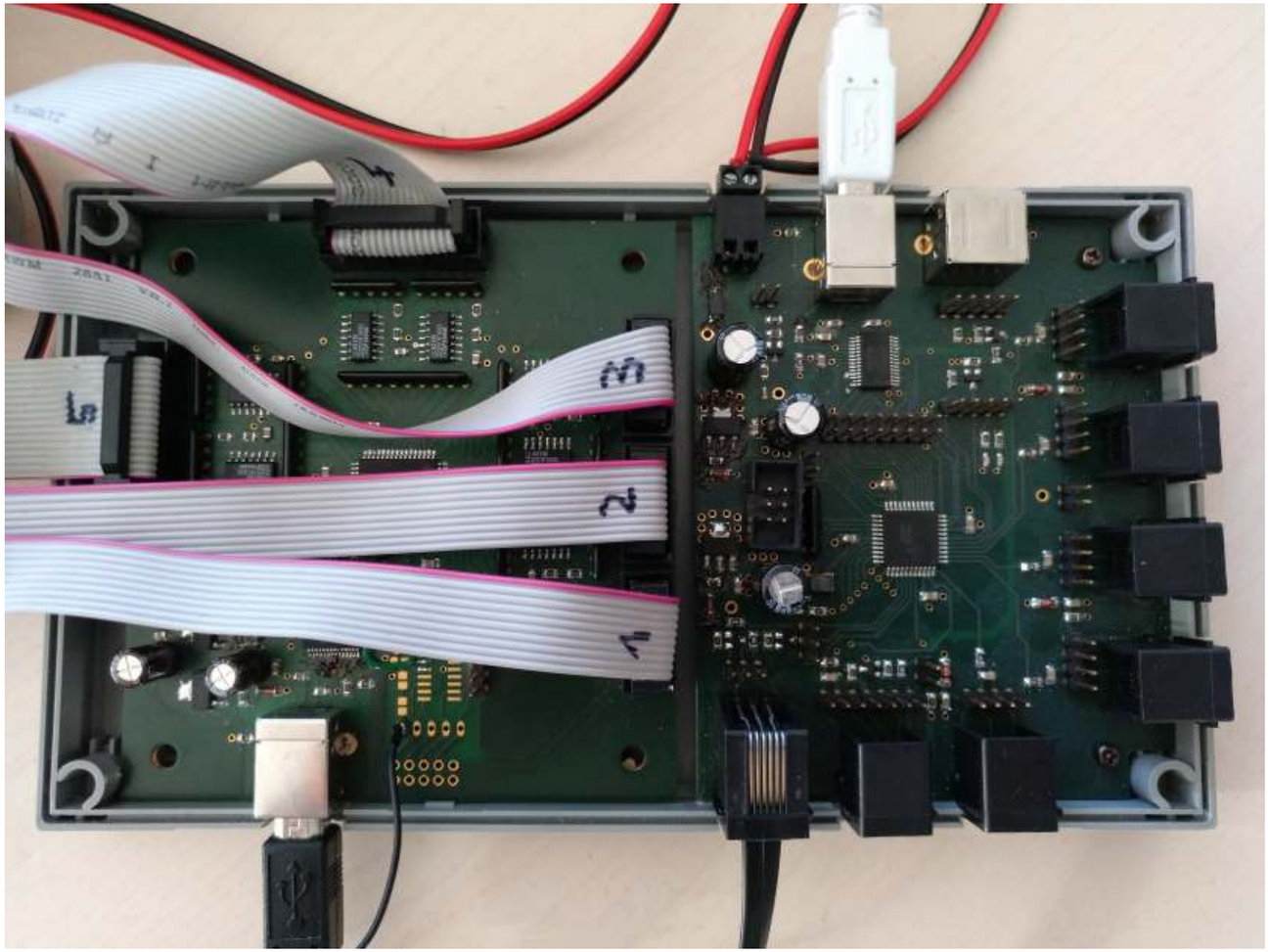

Daase, B., Matthes, L., Pirl, L., & Wenzel, L. Observing a Moving Target – Reliable Transmission of Debug Logs from Embedded Mobile Devices.

# Carrera sensor & actuator board

working with the
digital protocol

firmware level

e.g., prevent overtaking
the safety car



# single-board computers

distributed computing

energy-aware computing

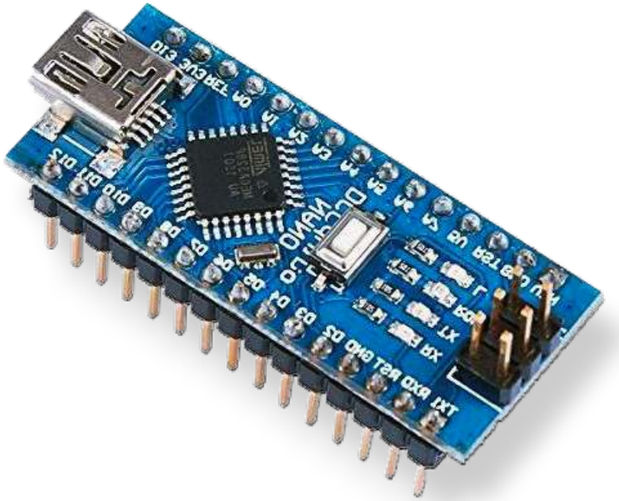
heterogeneous computing

(GP)GPU, big.LITTLE, …

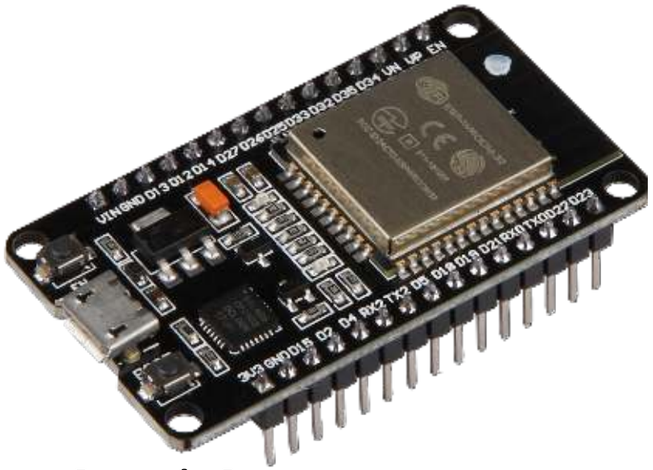


list of experiment hardware
https://osm.hpi.de/iot-lab/docs/hardware/generated-overview-of-experiment-hardware-list.html

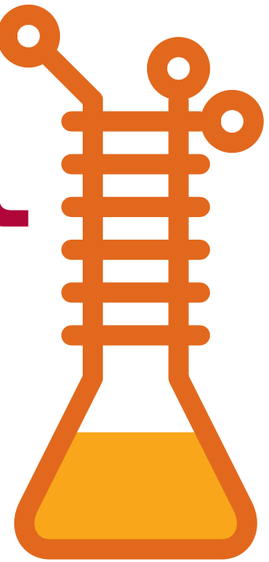
# embedded devices & OS



- devices
  - Arduino
  - ESP32
  - Beckhoff SPS
  - Lego
  - Fischertechnik
  - … *open to suggestions!* :)
    - incl. designing own PCBs
- operating systems
  - RIOT, *RTOS, OpenWRT, embedded Linux, Contiki, Android, …



# Digital Rail Lab



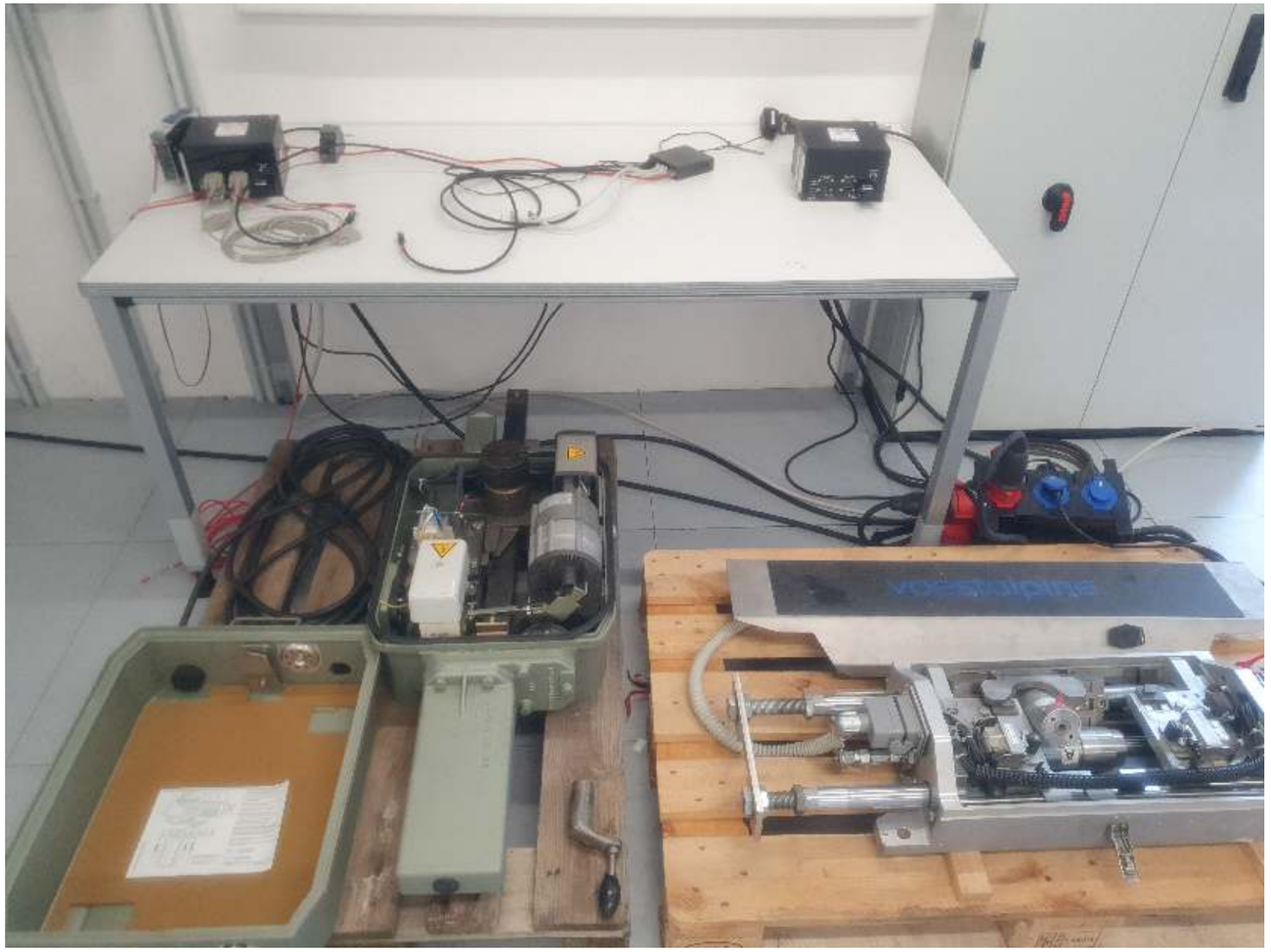
# Digital Rail Lab




level crossing, color light signal

# Digital Rail Lab

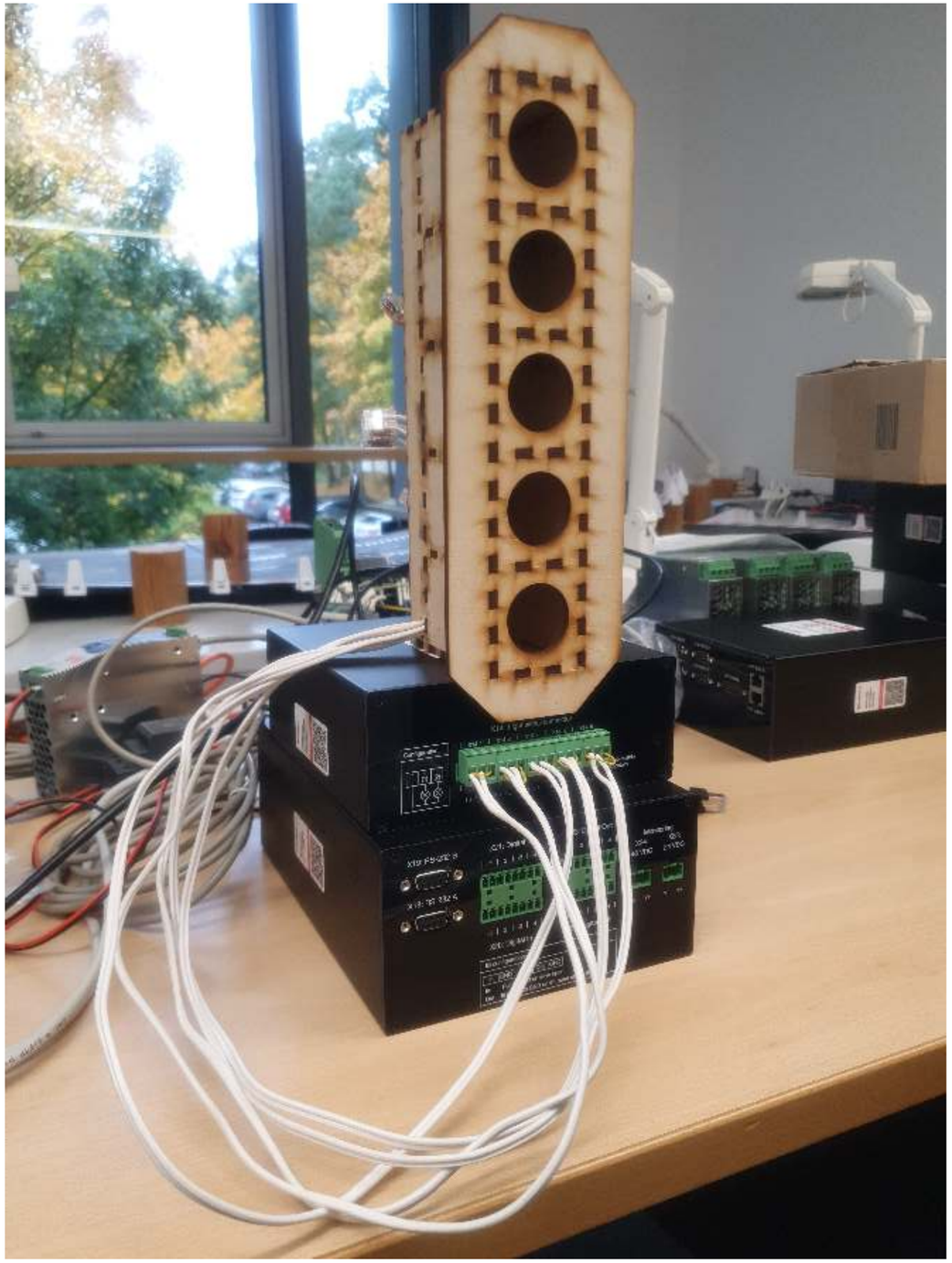
point machines





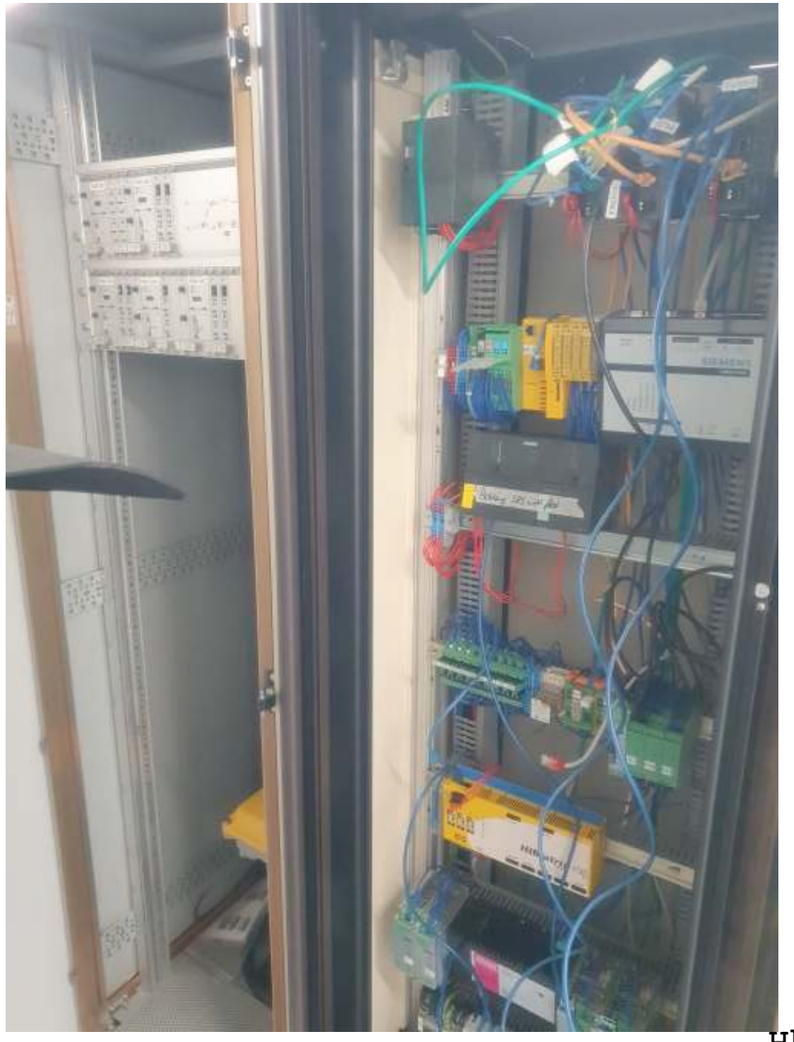
# Digital Rail Lab

object controllers



# Digital Rail Lab

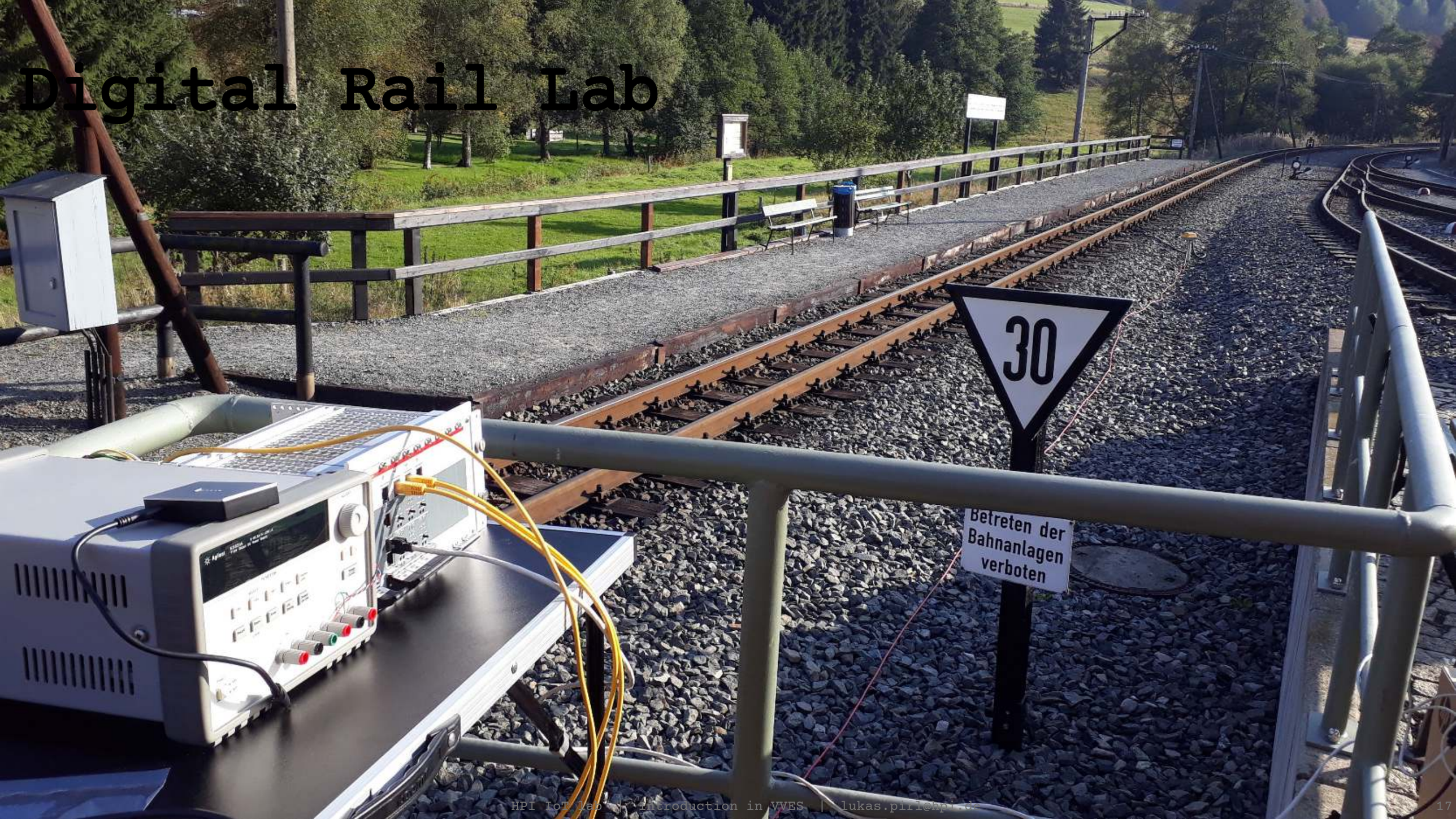




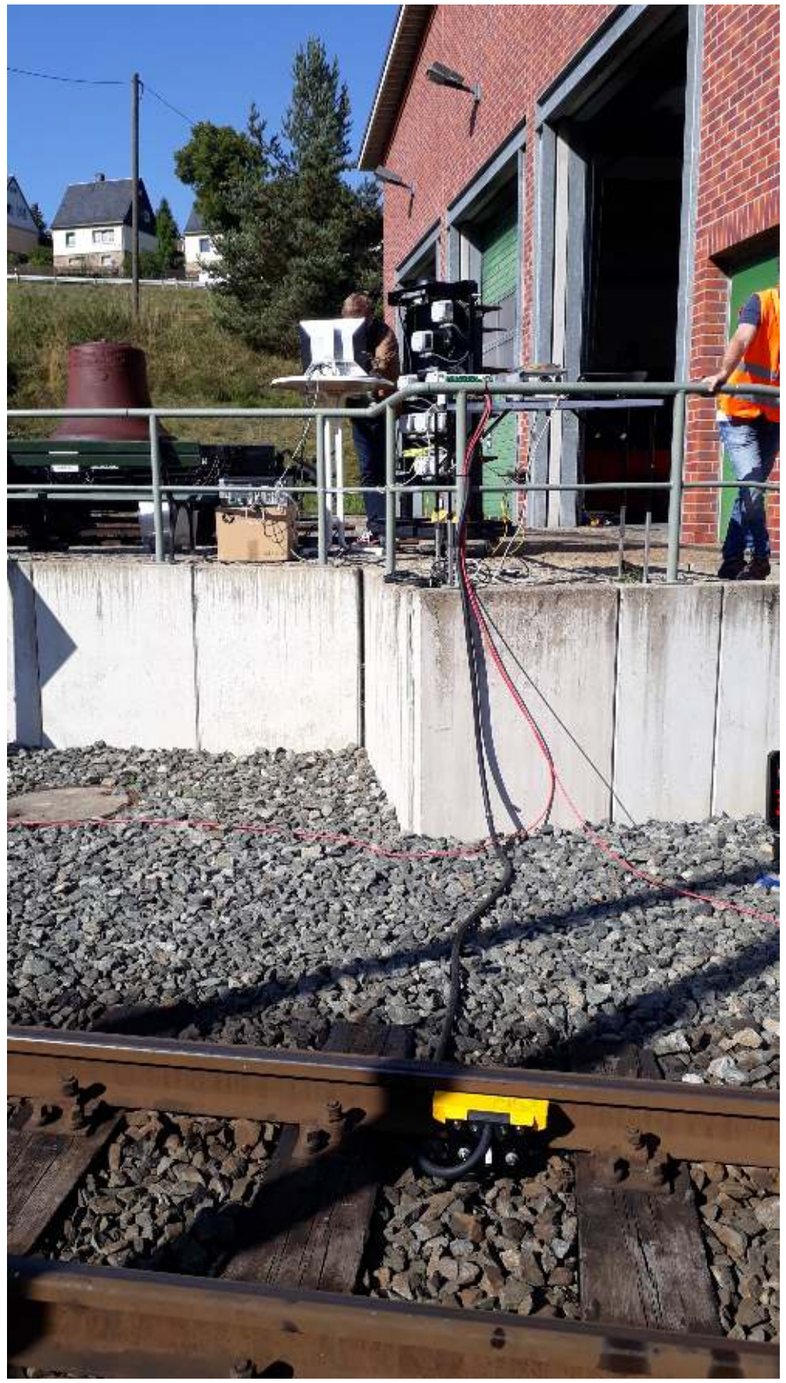


Hbst-Stw (Hilfsbetriebsstellenstellwerk)

# Digital Rail Lab



# Digital Rail Lab



# Digital Rail Lab




model-generated code counts axles and controls signal



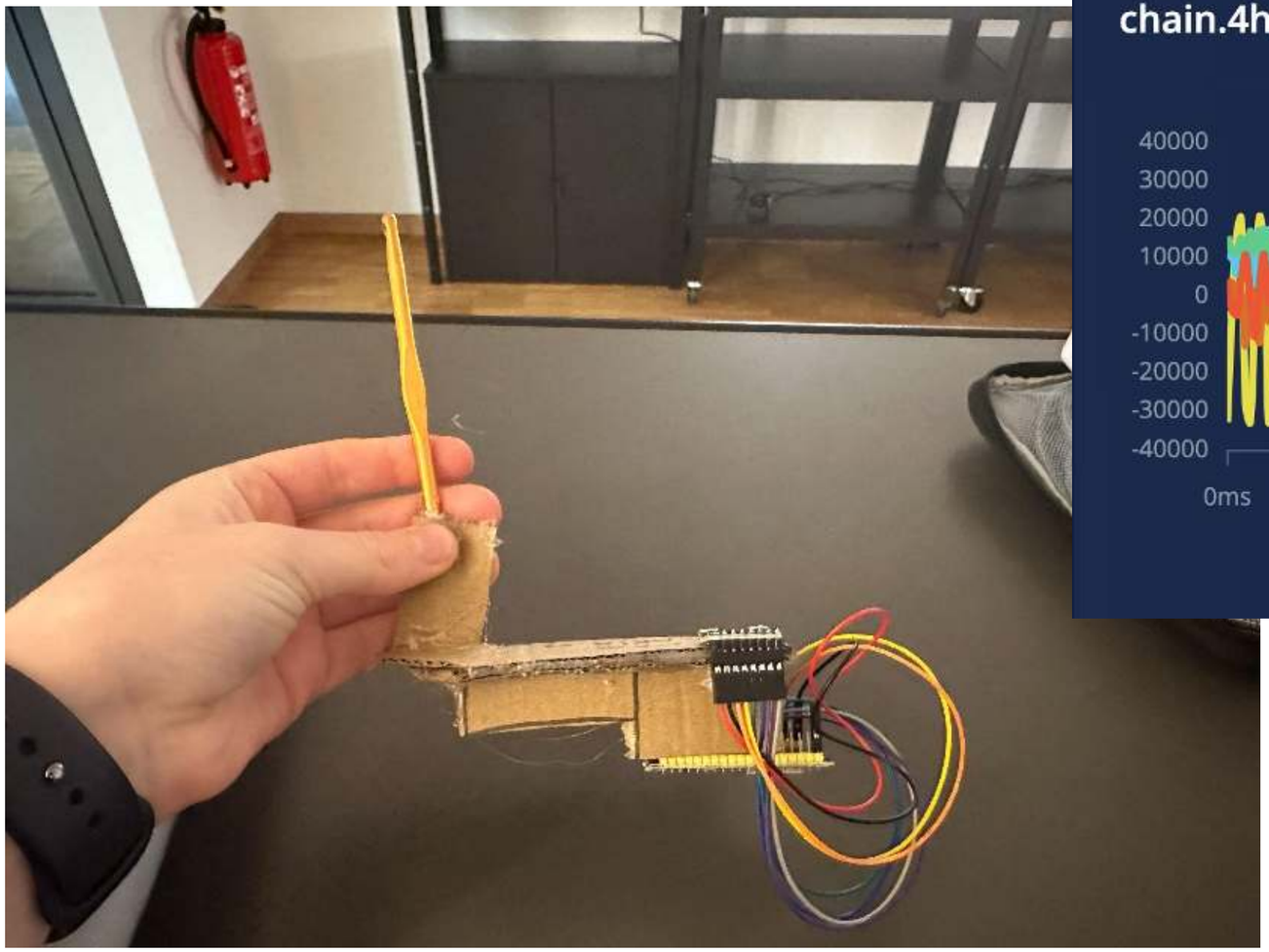
# DRSS

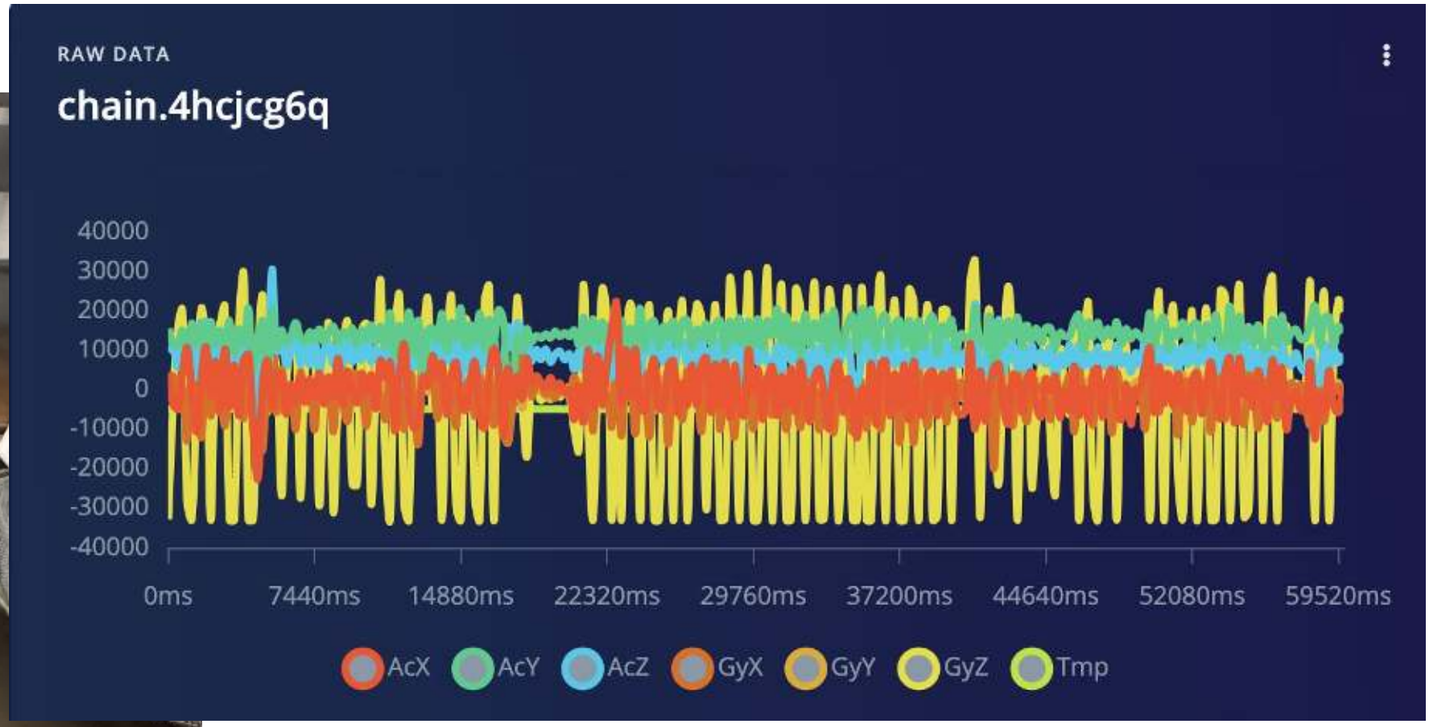
## Digital Rail Summer School

Join next year! :)

https://hpi.de/drss

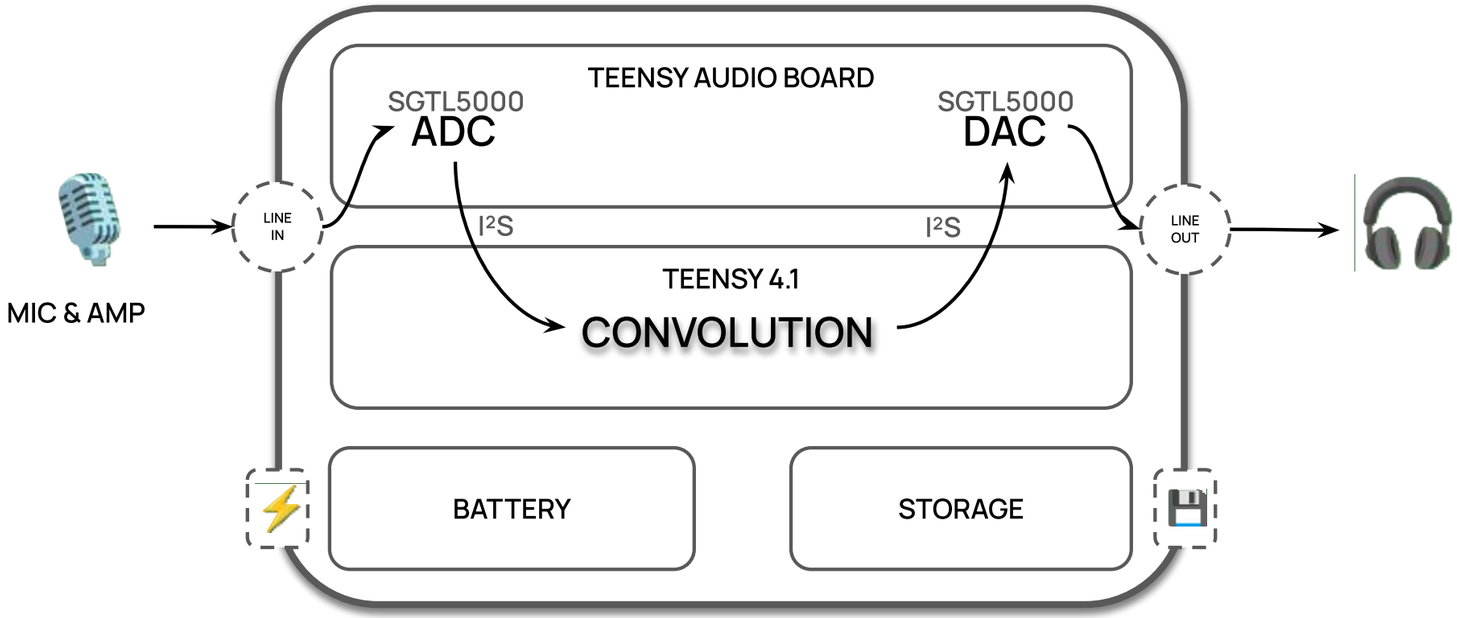
# crochet buddy

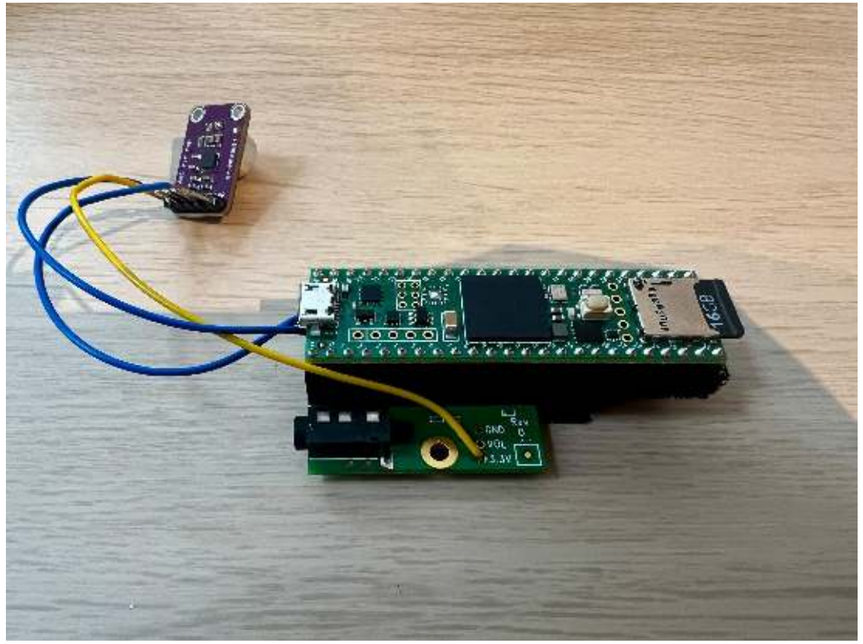





Franziska Lauterbach
Benjamin Daniel

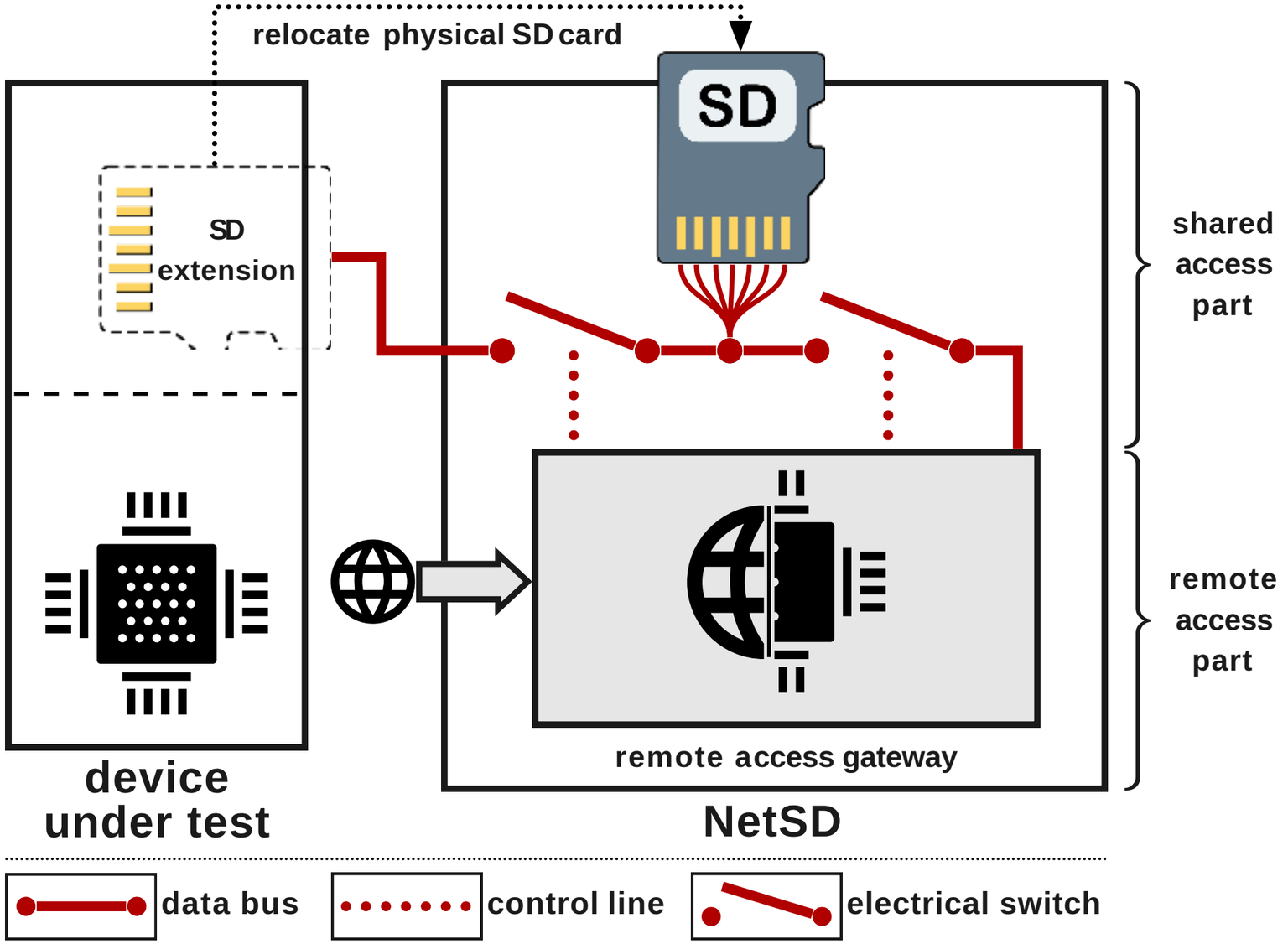
# mobile convolution reverb



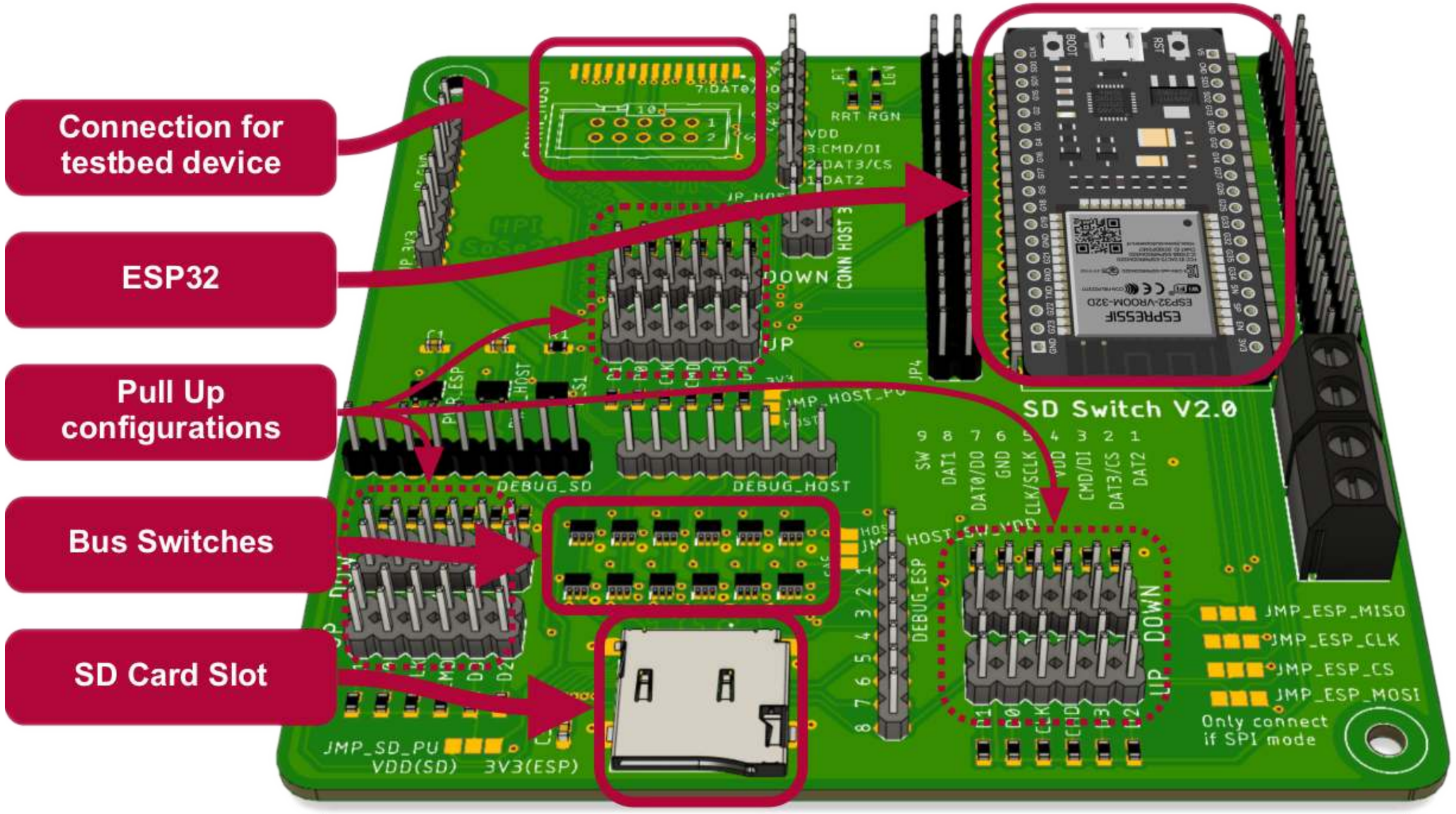
Ennio Strohauer

# testbed automation




Schröter, V., Boockmeyer, A., & Pirl, L.
NetSD: Remote Access to Integrated
SD Cards of Embedded Devices.

# testbed automation

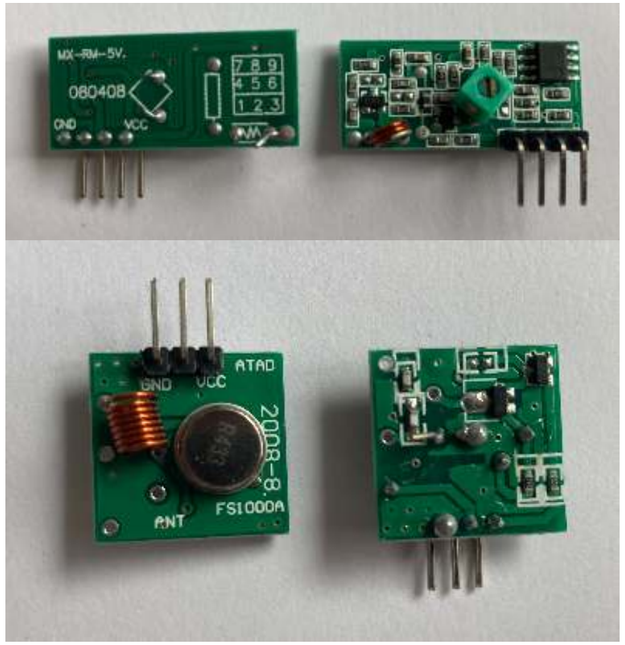


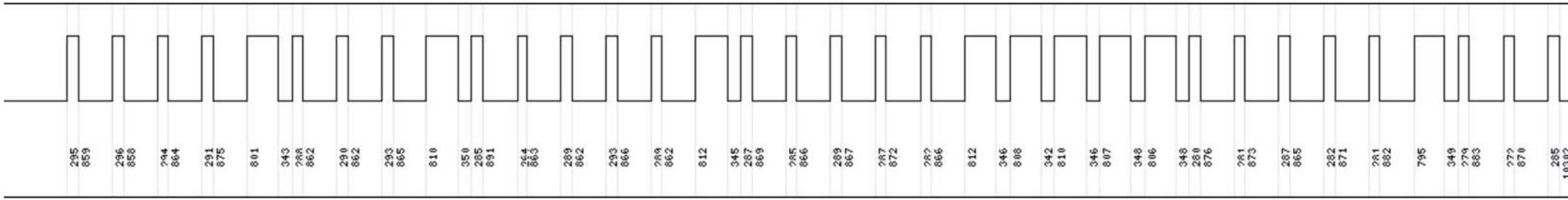

Valentin Schröter
Tobias Zagorni

# testbed automation

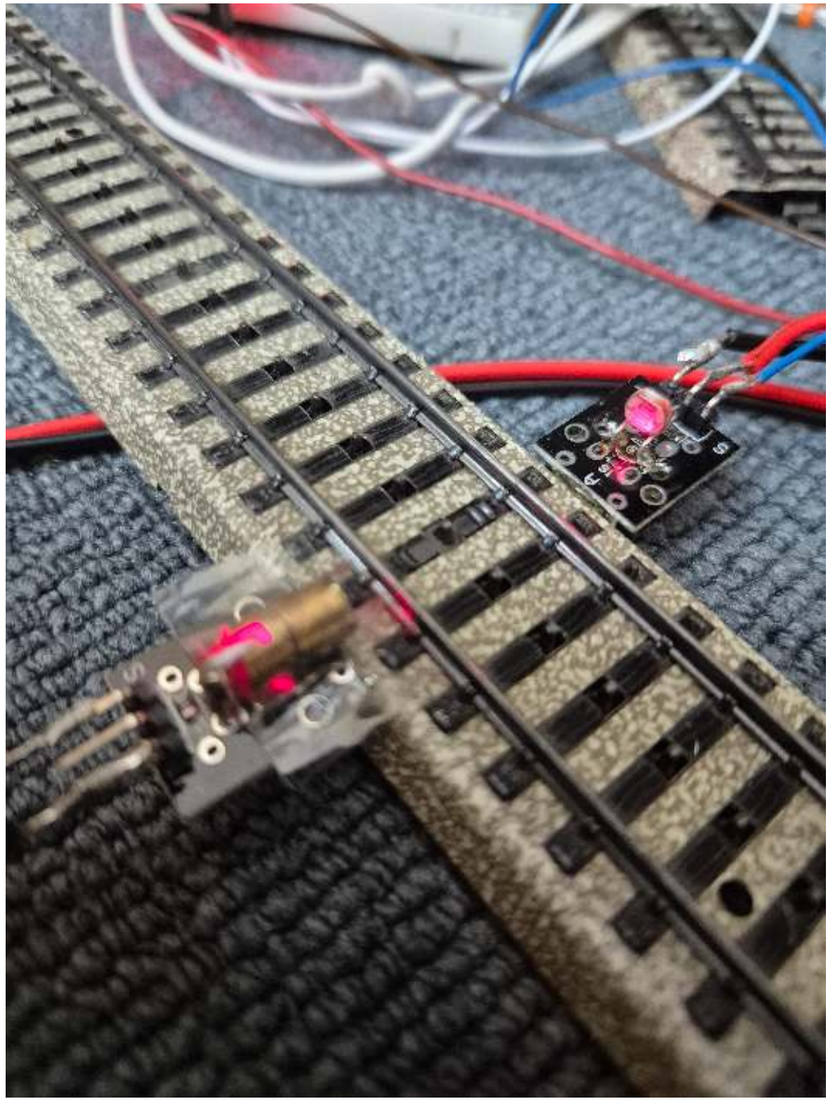
reverse engineering of switchable power sockets

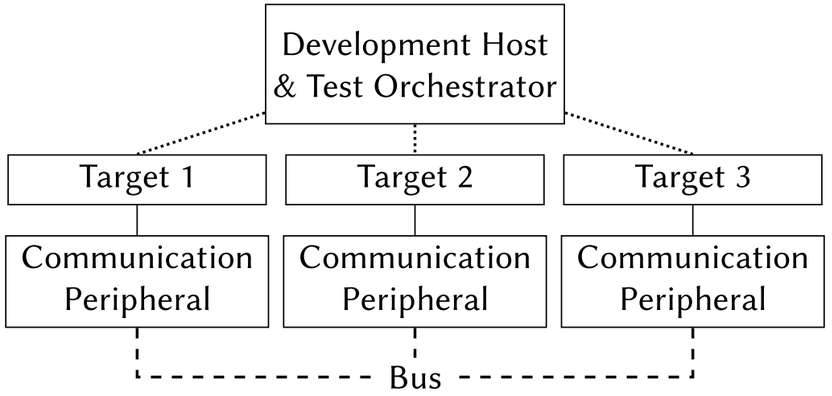


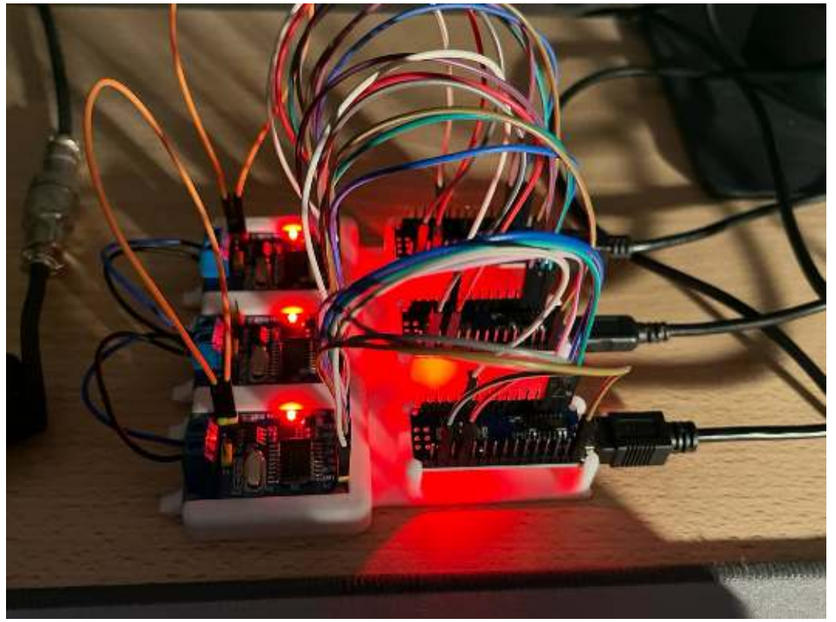


# model railway axle counter





## Minimalist Integration Testing Framework for Early-Stage Developments of Heterogeneous Distributed Embedded Systems

Luca Kleinschmidt
Luca.Kleinschmidt@student.hpi.de
Hasso Plattner Institute, University of Potsdam
Germany

Lukas Pirl
Andreas Polze
{first}.{last}@hpi.de
Hasso Plattner Institute, University of Potsdam
Germany

**Abstract**

For developing with commercial off-the-shelf microcontrollers, there are established tools. However, there are shortcomings regarding the integration testing of early-stage distributed embedded systems. Particularly, manual efforts for test execution and configuration management are repetitive, time-consuming, and error-prone. Existing frameworks lack low entry barriers, support for target platforms, heterogeneous setups, assertions on serial output, or device resets.

We present the *pytest* plugin *pytest-pio*. It reuses conventions (e.g., file system structure), configurations (e.g., device port mappings), and capabilities (e.g., to compile and upload) from *pytest* and *PlatformIO*. Test executions are orchestrated using a superordinate declarative configuration per test.

For an early-stage development of a heterogeneous embedded track vacancy detection system for model railways, *pytest-pio* served us as a minimalist integration testing framework. With its contribution, we hope to facilitate automation, timely fault detection, repeatability, representativeness, and overall efficient test-driven development cycles.

***CCS Concepts:*** • **Computer systems organization → Embedded software**; *Sensors and actuators*; Distributed architectures; • **Software and its engineering → Software testing and debugging**; • **Networks →** *Netw. experimentation.*

**1 Introduction**

In early phases of the implementation of an embedded system, testing the functionality of a single device can usually be done straightforwardly. For example, unit test tools enable automated testing on the actual microcontroller (*target*). In such a scenario, peripheral devices can be included natively. Assertions are either checked on the target itself or through reading and processing its serial output on the developer's computer (*host*). The situation complicates when developing distributed embedded systems. For distributed systems, not only the state and the behavior of single targets is of interest. Instead, possibly rapid chronological sequences of their interactions and the emergent behavior of the whole system are of central concern. However, low-barrier automation of testing on multiple targets simultaneously cannot be taken for granted, not to mention testing on heterogeneous targets. Missing automation fosters time-consuming, repetitive, and error-prone manual routines, e.g., for compiling, uploading, executing, measuring, and checking. Missing support for heterogeneous targets impedes integration testing, e.g., for early identification of design or implementation deficiencies.

Apart from testing on physical hardware, testing using simulations might appear appealing instead. However, simulations are less representative because they mimic hardware

Tilman Ripke
Sandro Steeger
Luca Kleinschmidt

# radio technology




LoRa-WAN

# radio technology

security analysis of railway radios

- Global System for Mobile Communications for Railway (GSM-R)

collision warning for vessels

- Global Satellite Navigation System (GNSS)
- Automatic Identification System (AIS)

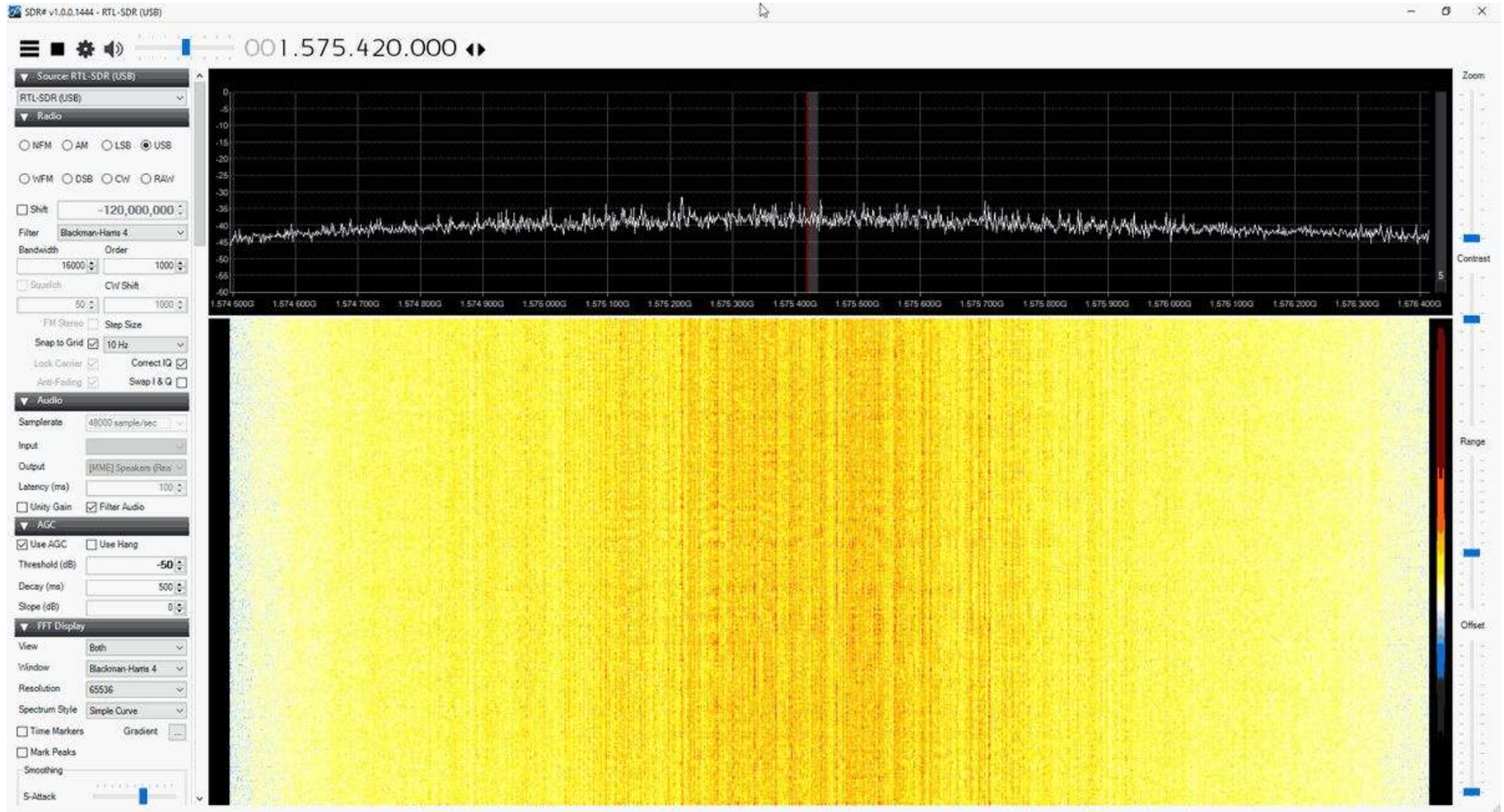



SDR

# radio technology

## lab GNSS





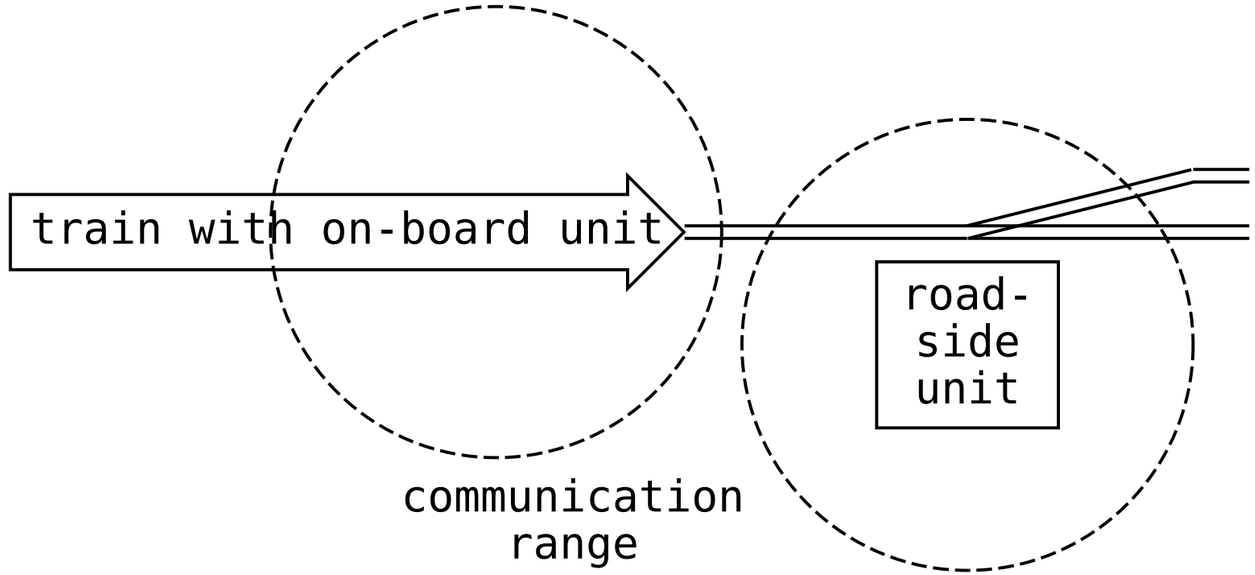

Jan Reppien
Ansgar Richter
Florian Papsdorf

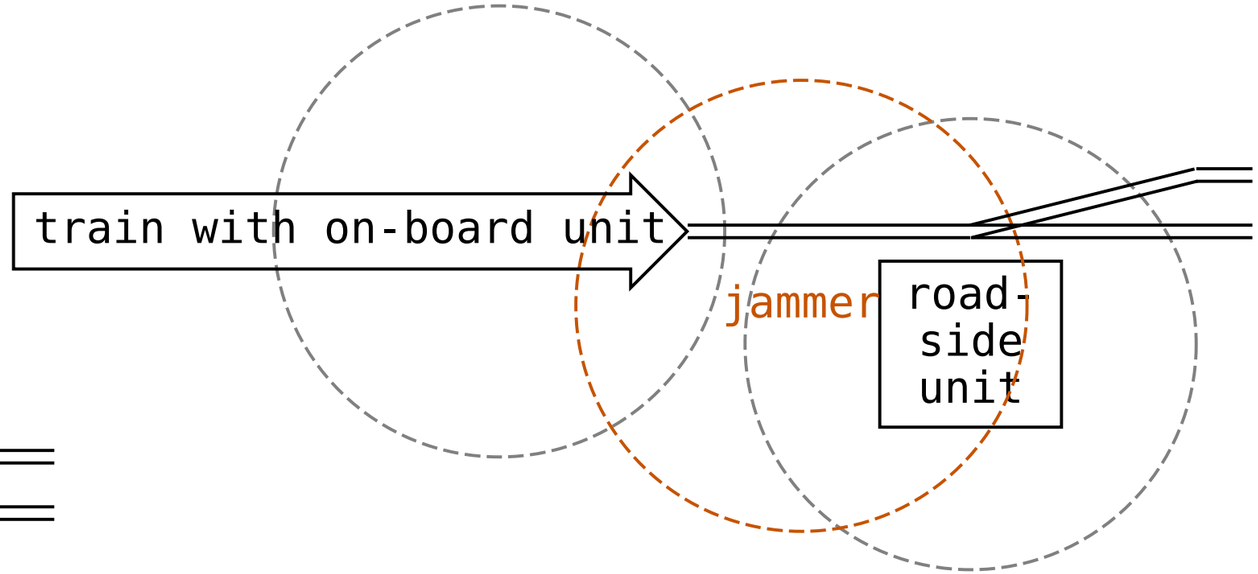
# radio technology



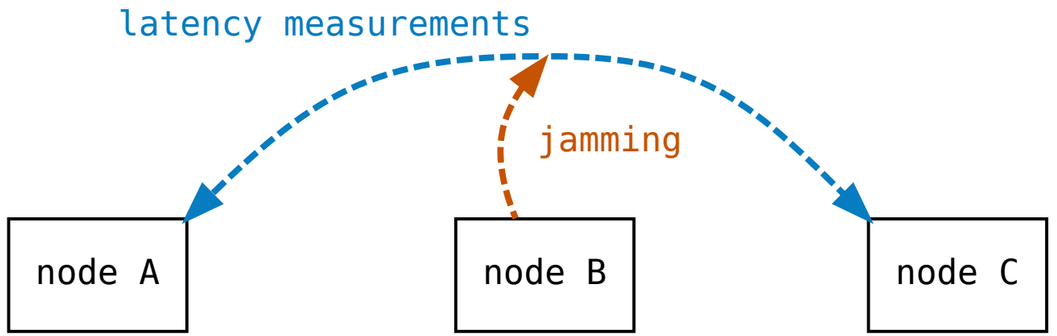

IEEE 802.11p

# fault injection

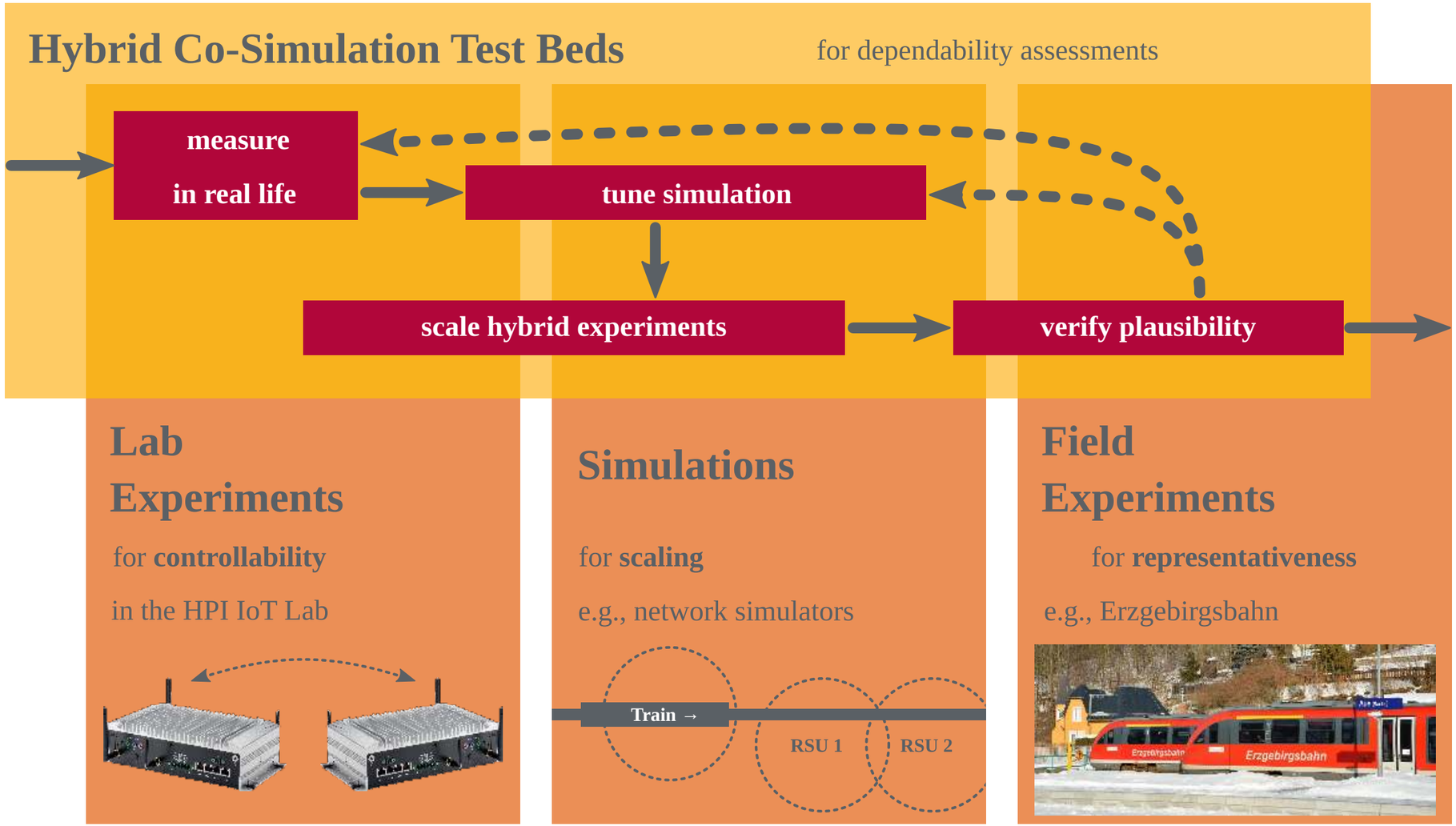


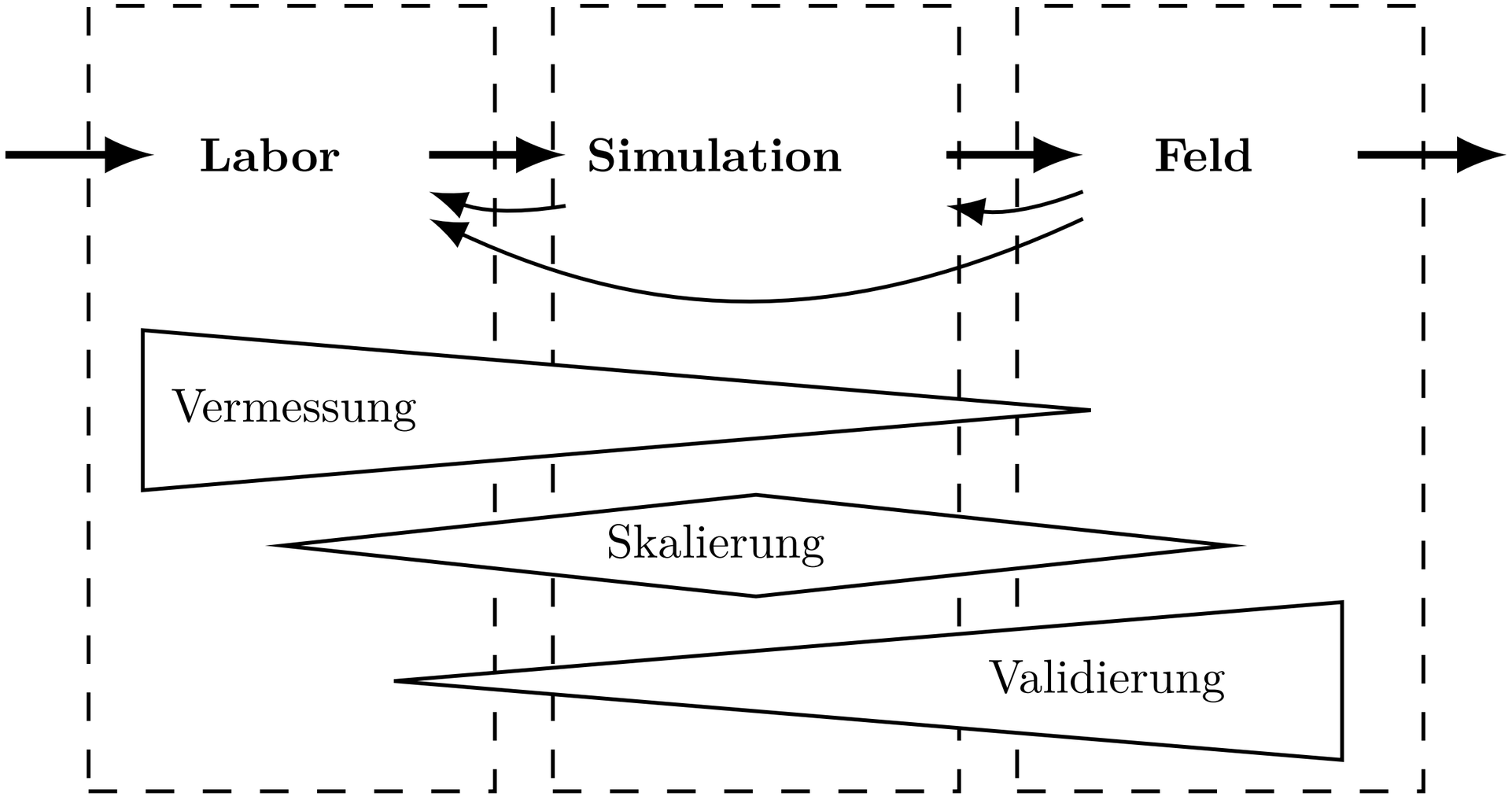


# development of methodologies



# development of methodologies



# hybrid testbeds & co-simulation

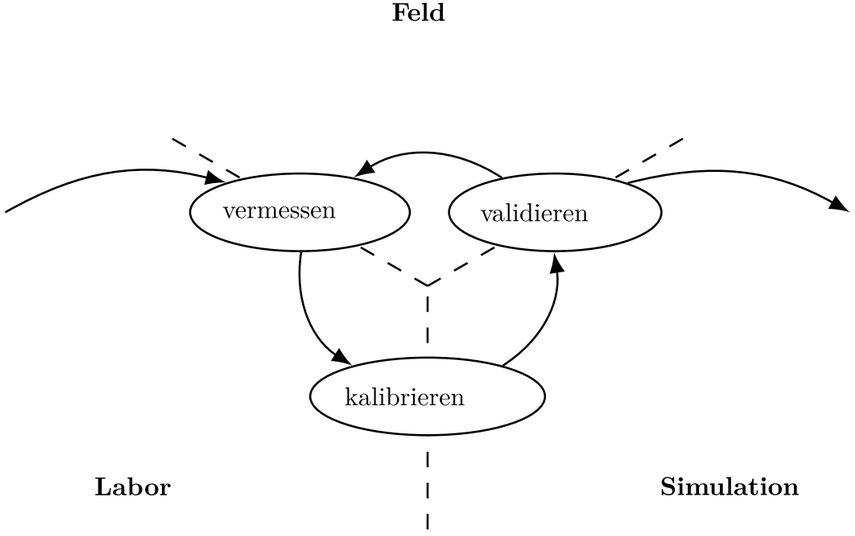
hybrid: software, hardware, and models “in-the-loop”

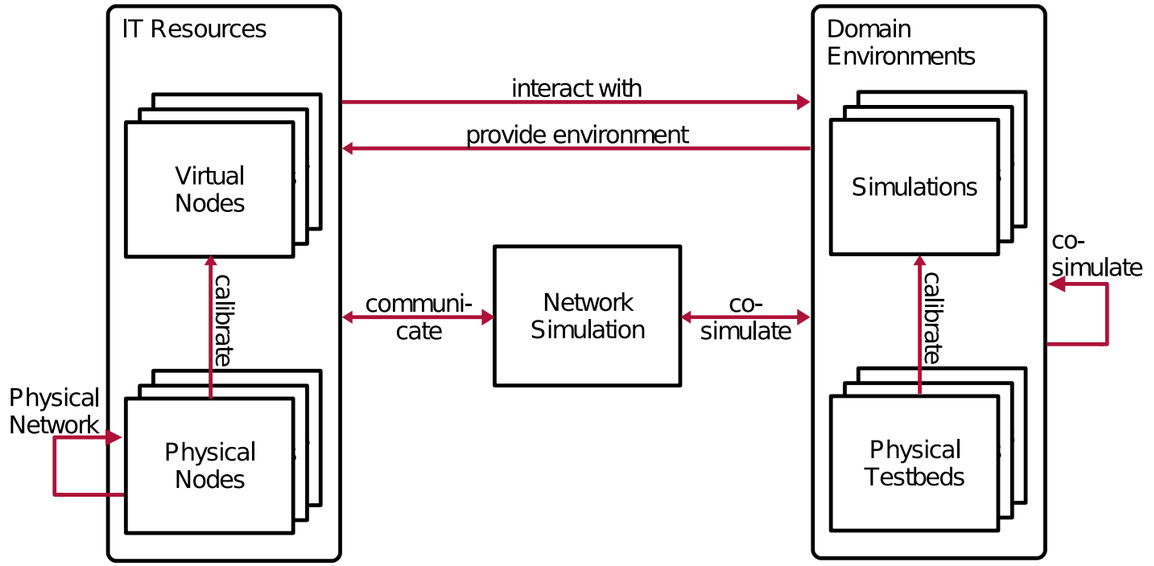
e.g., simulated wireless & physical wireless network

co-simulation: coupling of multiple domain-specific simulations

e.g., *SUMO* for traffic & *ns-3* for networking

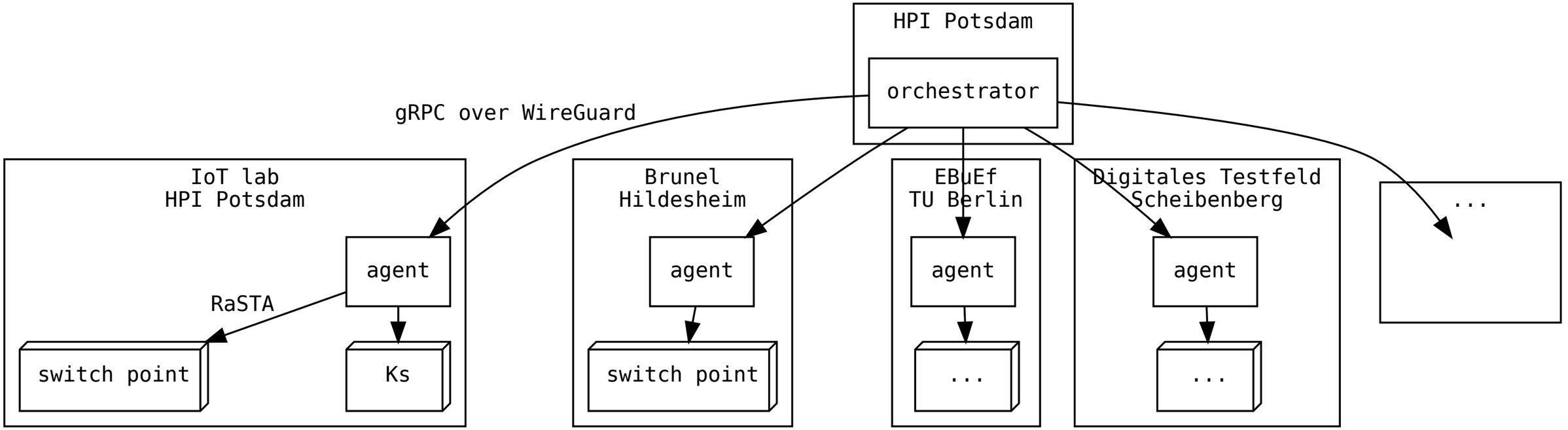
https://github.com/diselab/marvis



# distributed testbeds

EULYNX live lab: distributed test environment



# HPI IoT lab

*… an environment for prototyping and assessments*

building, DIY, testing, …

hybrid, co-simulated, and distributed setups

model-driven and -supported approaches

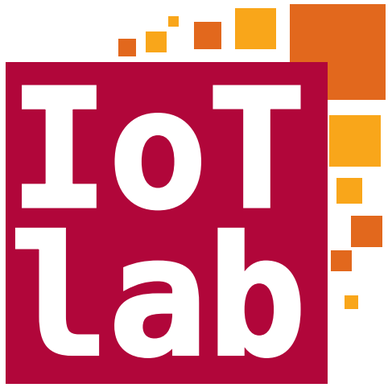
alternatives to specialized hardware

hardware, software and model “in the loop”

simulated environments

integrating domains

dependability




Introduction of IoT lab

Lecture on Distributed Dependable Embedded Systems

Professorship for Operating Systems and Middleware of Prof. Andreas Polze

Hasso Plattner Institute, University of Potsdam

Lukas Pirl