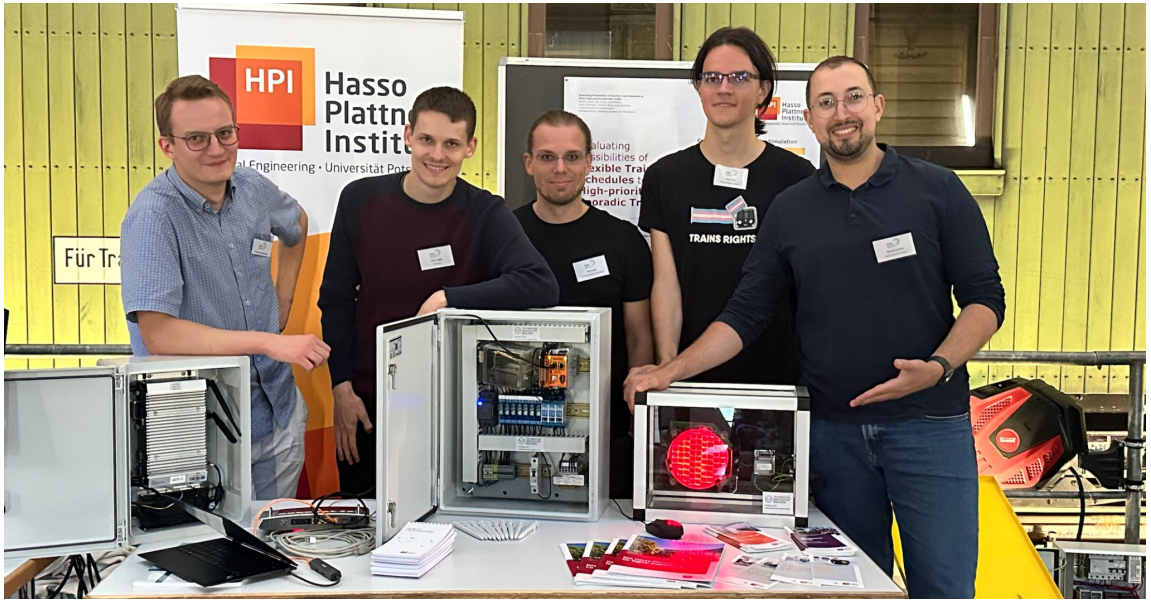
EULYNX Live

Entwicklung eines EULYNX-basierten Object Controllers für ein Ks-Signal

Das Team



TUC	TUD	TUB	HPI
Merlin Stollenwerk	Mark Hendrik Tegge Justus Woldt	Niels Geist	Ahmed Azzouz Lukas Rost

Motivation & Problemstellung

Motivation & Problemstellung

EULYNX

- **EULYNX:** Initiative 14 europäischer Infrastrukturbetreiber zur Standardisierung von Stellwerkskomponenten und Schnittstellen zwischen diesen
  - verbesserte Austauschbarkeit und Kombinierbarkeit von Komponenten
- **SCI-LS:** EULYNX-Standard für Object Controller (OC) von Lichtsignalen (definiert Verhalten sowie Schnittstelle zum Stellwerk)
- **Unser Ziel:** Entwicklung eines Proof of Concept OC nach SCI-LS
  - standardisierte Kommunikation mit dem Stellwerk
  - Ansteuerung der Signallampen über einen Feldelementeanschlusskasten (FeAk)
  - zunächst Fokus auf Implementierung des Verhaltens im Erfolgsfall, nicht im Fehlerfall
  - Nutzung der speichersicheren und hardwarenahen Programmiersprache Rust

Unsere Lösung

Unser Lösungsansatz

Stellwerk

gRPC

RaSTA Bridge

RaSTA

RaSTA Bridge

gRPC

Object Controller

Protokolle: gRPC, RaSTA, SCI-LS - was ist was?

Protokolle: RaSTA, gRPC, SCI-LS - was ist was?

- **Rail Safe Transport Application (RaSTA):** Netzwerkprotokoll für spezielle Bedürfnisse der digitalen LST
  - Sicherheits- und Sendewiederholungsschicht: verlässliche Übertragung ohne unbemerkten Paketverlust, Überwachung der Verbindungsqualität durch Heartbeats, garantierte Zustellung innerhalb eines Zeitfensters
  - Redundanzschicht: Nutzung mehrerer Transportkanäle für erhöhte Ausfallsicherheit

Protokolle: RaSTA, gRPC, SCI-LS - was ist was?

- **gRPC Remote Procedure Calls:** Protokoll zum Aufruf von Funktionen in verteilten Systemen
  - Kapselung der RaSTA-Implementierung durch RPCs
  - Ermöglicht Senden und Empfangen von Datenstreams

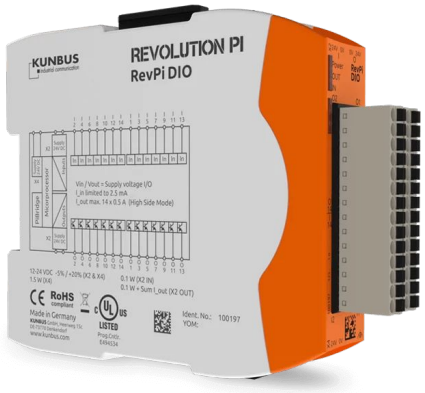
Protokolle: RaSTA, gRPC, SCI-LS - was ist was?

- **Standard Communication Interface - Light Signal (SCI-LS):** standardisierte Schnittstelle nach EULYNX zur Kommunikation mit Lichtsignalen
  - Nutzung von RaSTA zur Übertragung
  - Handshake: Protokollversions-Abgleich, Synchronisierung des initialen Signalzustandes
  - Stellbefehle und Statusnachrichten, Befehle zur Tag-Nacht-Umschaltung

Soft- & Hardware: Wie kommt das Signal zum Signal?

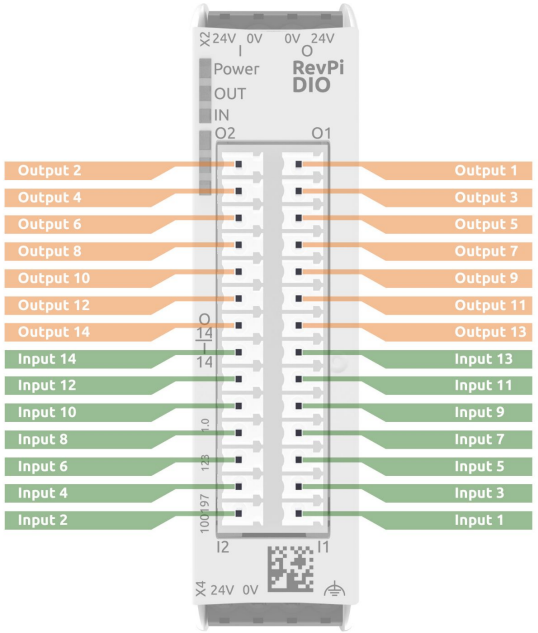
Soft- & Hardware: Wie kommt das Signal zum Signal?

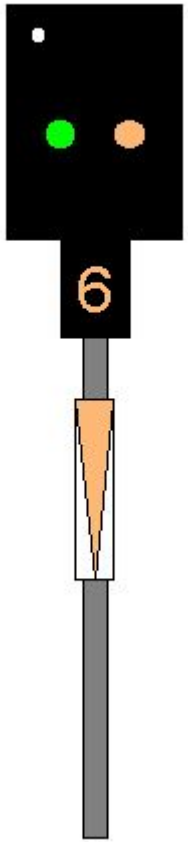
- Verwendung eines Revolution Pi zur Ausführung der OC-Software
  - industrietauglicher Computer auf Basis des Einplatinenrechners Raspberry Pi
  - Hutschienengehäuse, Stromversorgung gemäß Industriestandard (24 V)
  - I/O-Modul für digitale Ein- und Ausgänge
- PiControl-Bibliothek für den Revolution Pi
  - ermöglicht, auf den digitalen I/O-Ports eingehende Signale zu lesen und ausgehende Signale zu erzeugen



Soft- & Hardware: Wie kommt das Signal zum Signal?

- Pin Belegung je nach SCI-Signal
- Generische Implementierung



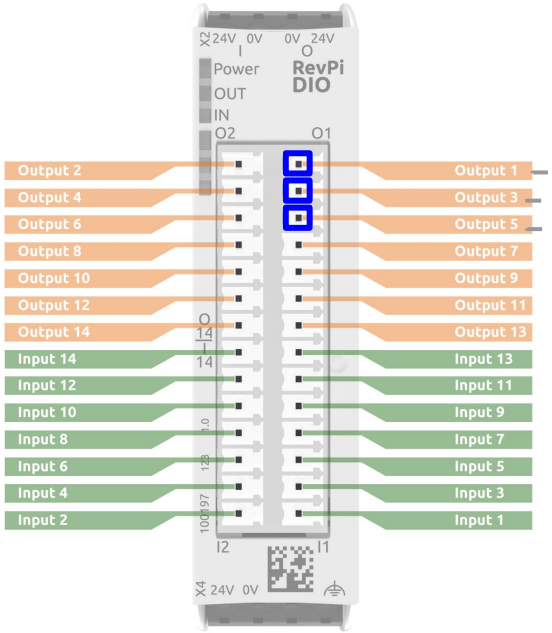


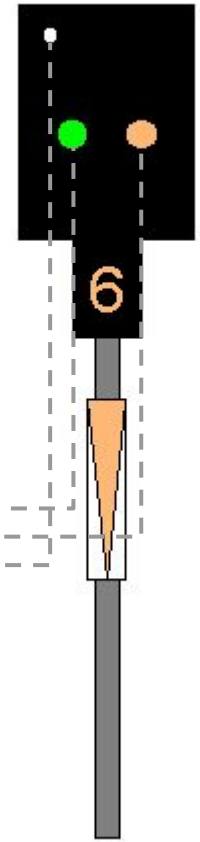
Soft- & Hardware: Wie kommt das Signal zum Signal?

- Pin Belegung je nach SCI-Signal
- Generische Implementierung

Beispiel:

```
pins_output=["O_1", "O_3", "O_5"]
```



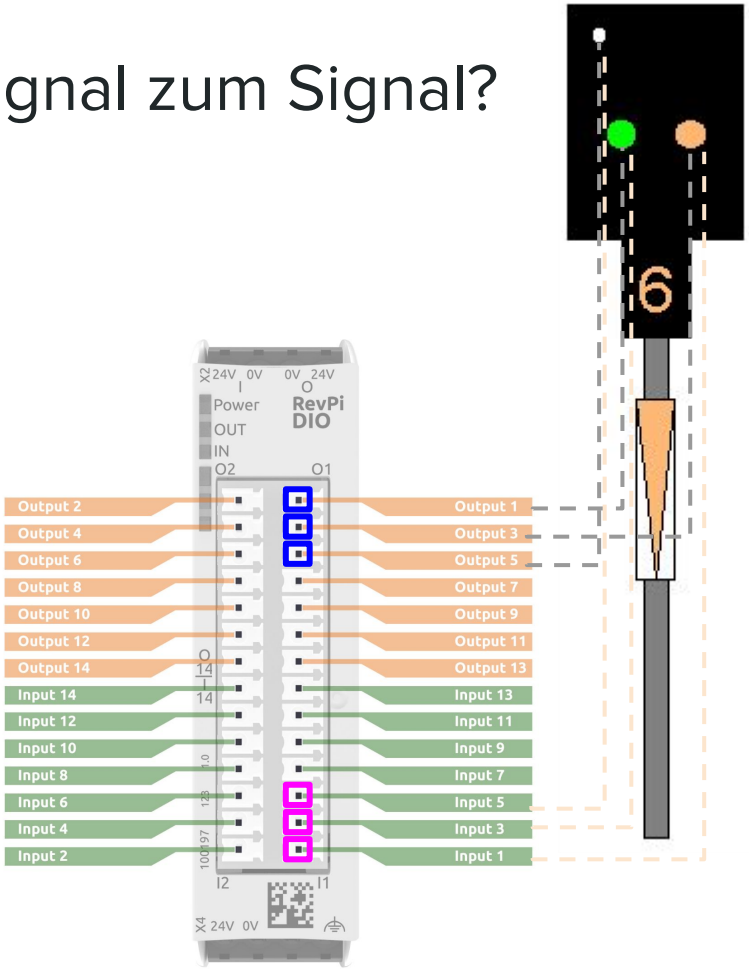


Soft- & Hardware: Wie kommt das Signal zum Signal?

- Pin Belegung je nach SCI-Signal
- Generische Implementierung

Beispiel:

```
pins_output=["O_1", "O_3", "O_5"]

pins_inputs=["I_1", "I_3", "I_5"]
```

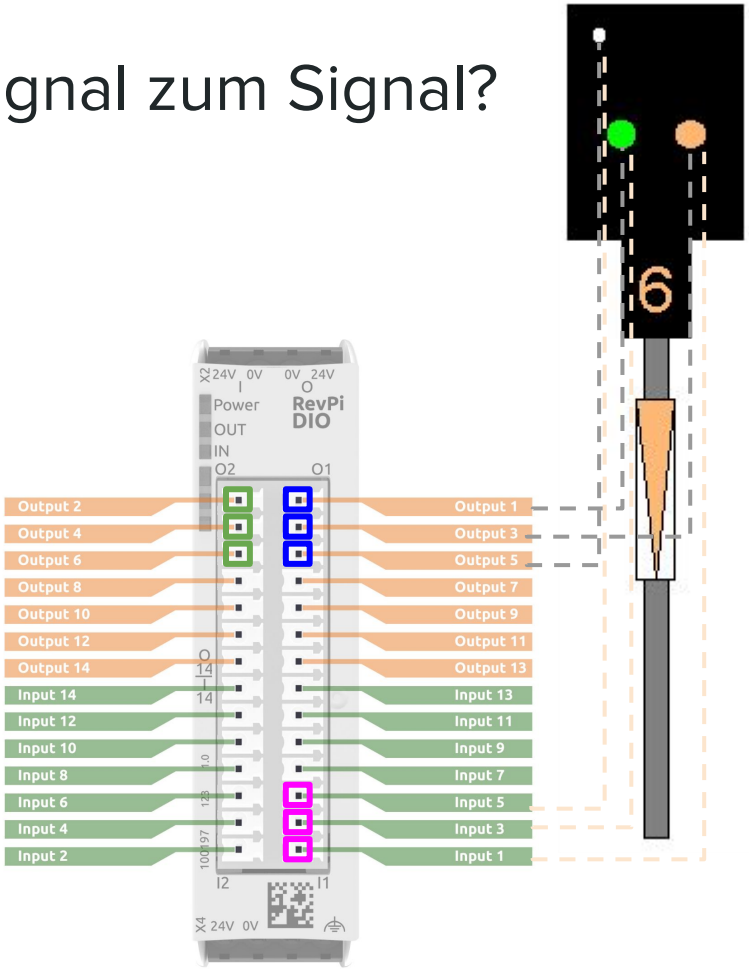


Soft- & Hardware: Wie kommt das Signal zum Signal?

- Pin Belegung je nach SCI-Signal
- Generische Implementierung

Beispiel:

```
pins_output=["O_1", "O_3", "O_5"]

pins_inputs=["I_1", "I_3", "I_5"]

pins_output_backup=["O_2", "O_4", "O_6"]
```

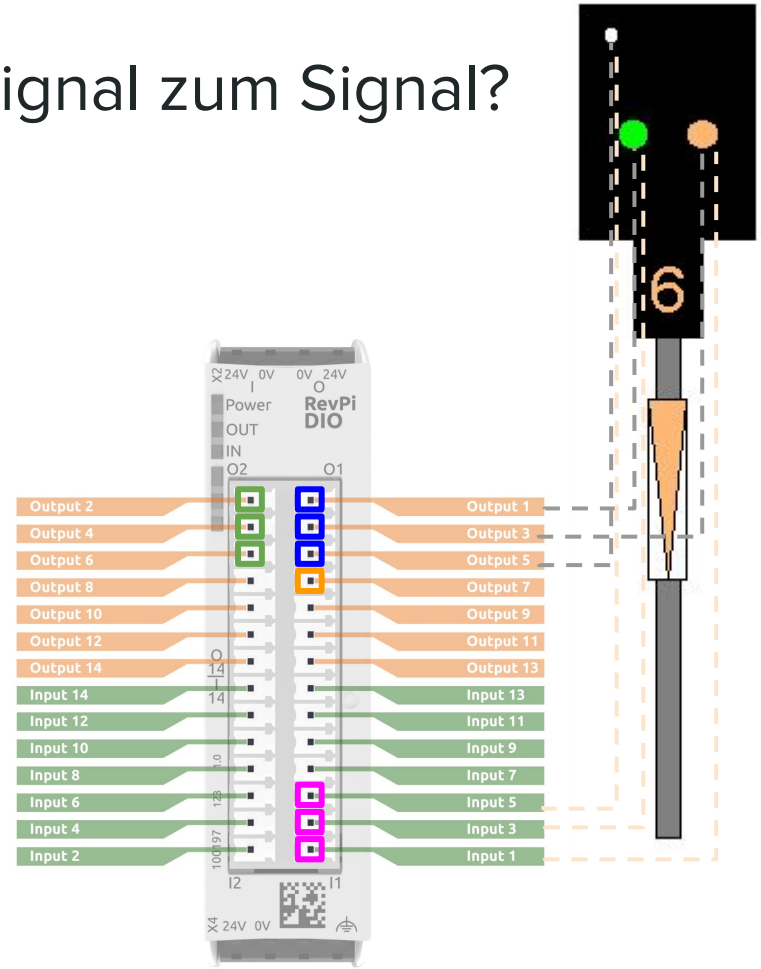


Soft- & Hardware: Wie kommt das Signal zum Signal?

- Pin Belegung je nach SCI-Signal
- Generische Implementierung

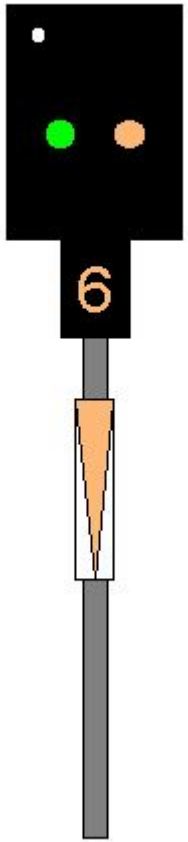
Beispiel:

```
pins_output=["O_1", "O_3", "O_5"]

pins_inputs=["I_1", "I_3", "I_5"]

pins_output_backup=["O_2", "O_4", "O_6"]

day_night_pin="O_7"
```



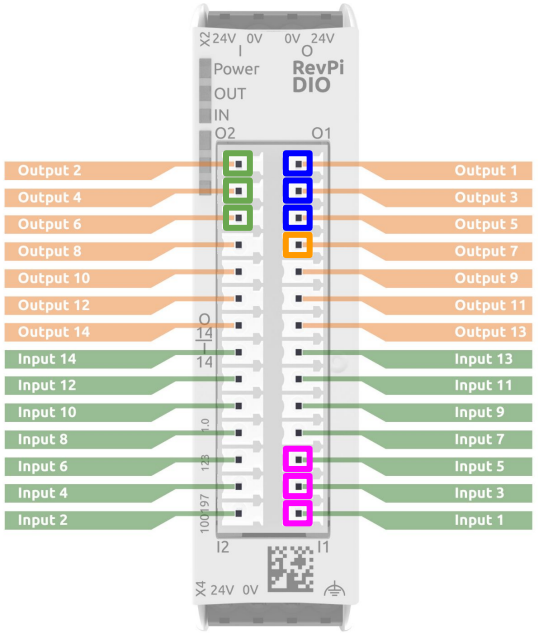
Soft- & Hardware: Wie kommt das Signal zum Signal?

- Pin Belegung je nach SCI-Signal
- Generische Implementierung



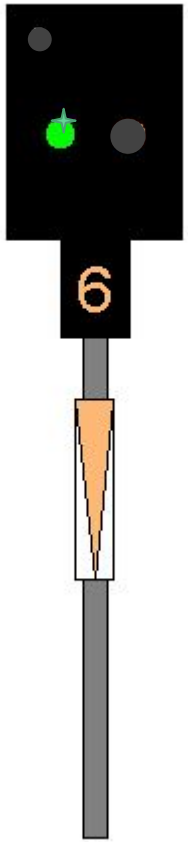
Beispiel:

```
pins_output=["O_1", "O_3", "O_5"]

pins_inputs=["I_1", "I_3", "I_5"]

pins_output_backup=["O_2", "O_4", "O_6"]

day_night_pin="O_7"

signals={"Off"=[0,0,0],"KS1"=[1,0,0],"KS2"=[0,1,0]}
```



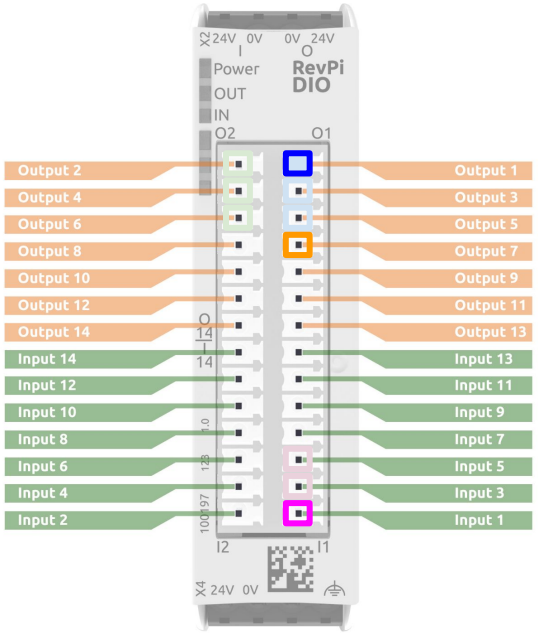
Soft- & Hardware: Wie kommt das Signal zum Signal?

- Pin Belegung je nach SCI-Signal
- Generische Implementierung



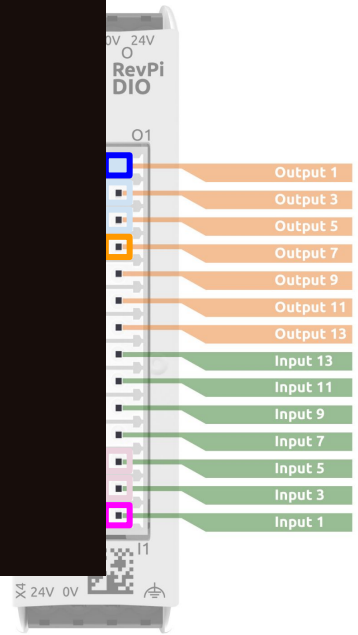
Beispiel:

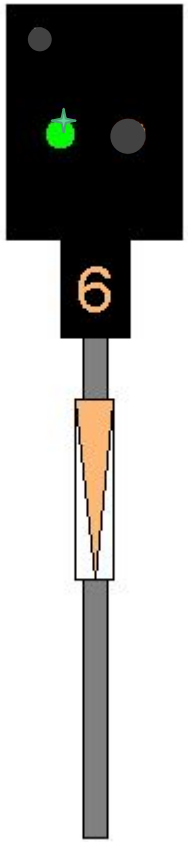
```
pins_output=["O_1", "O_3", "O_5"]

pins_inputs=["I_1", "I_3", "I_5"]

pins_output_backup=["O_2", "O_4", "O_6"]

day_night_pin="O_7"

signals={"Off"=[0,0,0],"KS1"=[1,0,0],"KS2"=[0,1,0]}
```



Soft- & Hardware: Wie kommt das Signal zum Signal?

```
Signal brightness is now Day
Received version request - sending version response telegram -> version check successful
Forwarding gRPC message...
Received status request - sending status telegrams
Forwarding gRPC message...
Forwarding gRPC message...
Forwarding gRPC message...
Forwarding gRPC message...

--------------------------------------------------
Check signal Ks2
17-09-2023 12:34:07 ERROR: NO INPUT SIGNAL FOUND AT I_3, TRY TO USE THE BACKUP LINE!
PIN: O_13, VALUE: 1

--------------------------------------------------
Received show signal aspect telegram: changing main to Ks1
Signal shows Ks1
PIN: O_1, VALUE: 0
PIN: O_2, VALUE: 0
PIN: O_3, VALUE: 0
PIN: O_14, VALUE: 1
Forwarding gRPC message...

--------------------------------------------------
Check signal Ks1
17-09-2023 12:34:12 Signal OK! No errors found.

--------------------------------------------------
```

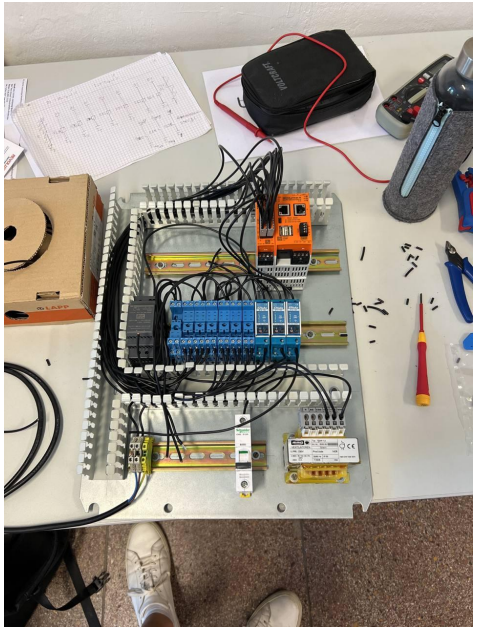




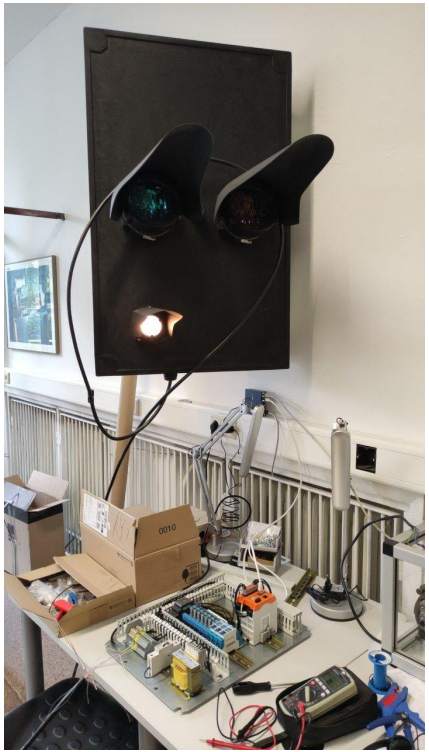
Der Feldelementeanschlusskasten (FeAk)

Der Feldelementeanschlusskasten (FeAk)

- Signaloptiken mit Haupt- und Nebenfaden
  - Umschaltung bei Ausfall des Hauptfadens möglich
- Ansteuerung von Haupt- und Nebenfäden mittels Relais
  - zusätzlich: Überwacherrelais zur Detektierung von Fadenbruch für die Hauptfäden
- Transformatoren zur Erzeugung von für die Signaloptiken (170 V - Tag / 130 V - Nacht) und den Revolution Pi (24 V) geeigneten Versorgungsspannungen



Demonstration: Signale auf Fahrt erwarten stellen!



Zusammenfassung und Ausblick

Zusammenfassung und Ausblick

- **Ergebnis:** Entwicklung eines funktionstüchtigen Proof of Concept für einen Lichtsignal-Object Controller nach EULYNX
  - zunächst noch mit starkem Fokus auf Verhalten im Erfolgsfall (“Happy Path”)
- **Erweiterungsmöglichkeiten:**
  - vollständige Implementierung des SCI-LS Standards, insbesondere bezüglich Verhalten in Fehlerfällen
  - Weiterleitung der Information über Fadenbruch an das Stellwerk mittels Diagnoseprotokoll SDI
  - ggf. Anpassung für andere Signalsysteme außer Ks (z.B. Hl)
  - Redundante Netzwerkverbindungen

Danke!

...an alle Teammitglieder, an unsere Projektbetreuer und an Sie als Zuhörer!

Bildquellen

- EULYNX-Logo: Eulynx Consortium, https://eulynx.eu/images/eulinx/eulynx-logo.png
- ESTW Hagen Hbf: Christian Liebscher (CC BY-SA 3.0), https://commons.wikimedia.org/wiki/File:Bahnhof_Hagen_Hbf_09_Stellwerk_Hpf.jpg
- Zwischensignal: Falk2 (CC BY-SA 4.0), https://commons.wikimedia.org/wiki/File:I20_062_Zwsig_Q15,_Fahrtstellung_Ri_Dre.jpg
- Router Icon Transparent: https://icon-library.com/icon/router-icon-transparent-16.html
- Revolution Pi: https://revolutionpi.de/wp-content/uploads/manuell/RevPi-Connect-4-thumbnail.png, https://revolutionpi.de/wp-content/uploads/manuell/IO_Modul_RevPi.png