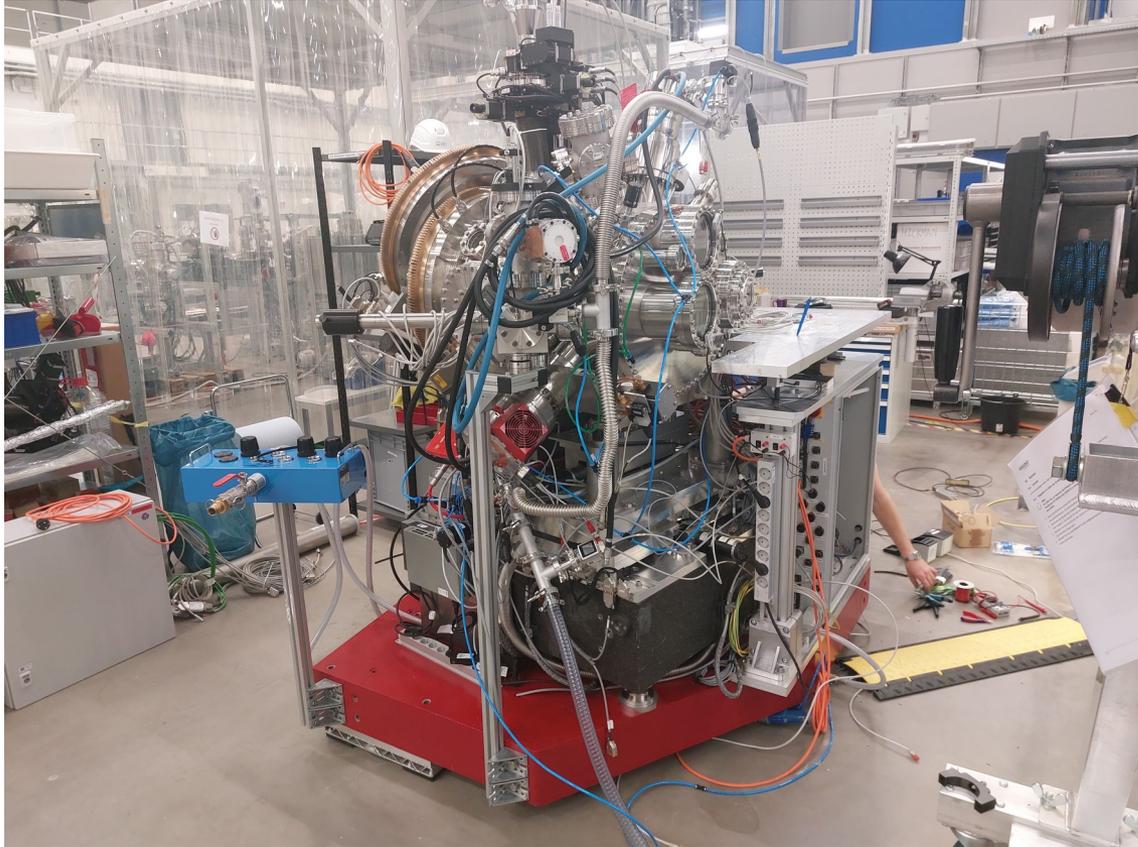

EBPFCat: Beckhoff ohne Beckhoff - Individuelle Lösungen mit Python programmiert

Martin Teichmann
SCS Instrument @ XFEL

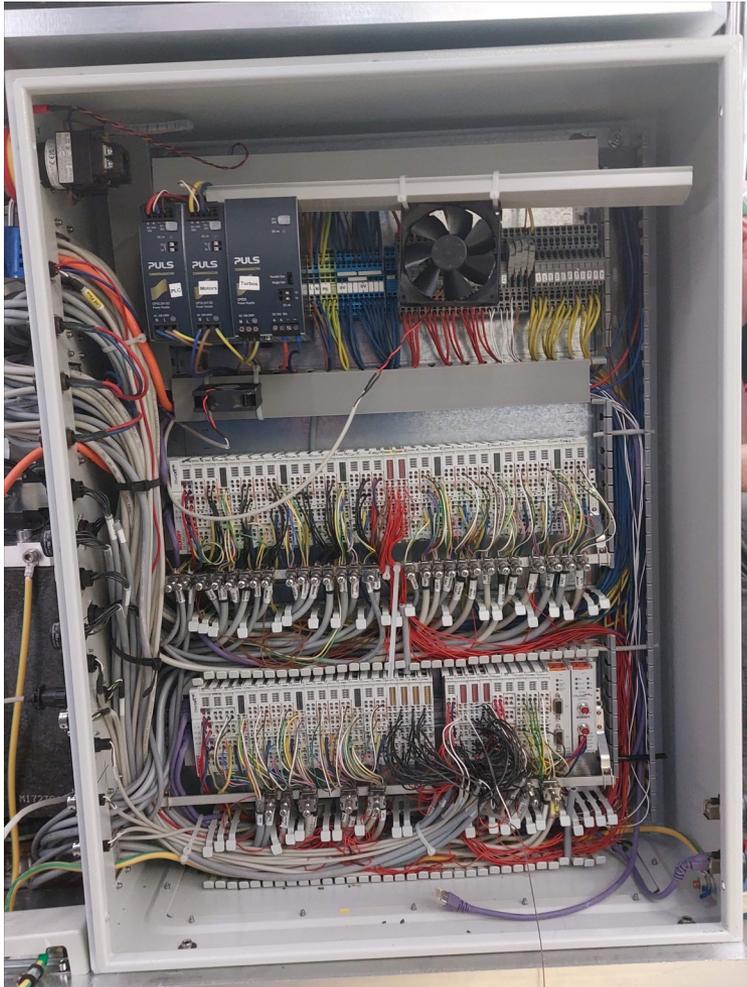


Motivation



Ausserhalb von Strahlzeiten hat der experimentelle Aufbau keine Verbindung zu den XFEL SPS CPUs

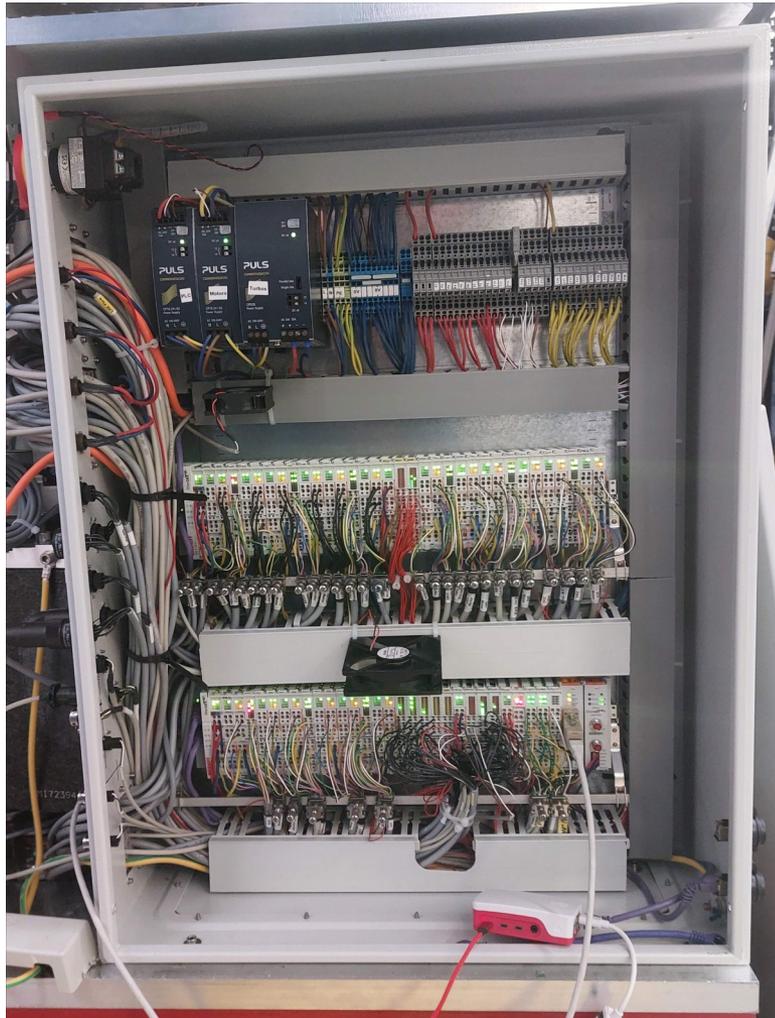
Problemstellung



Beim European XFEL verwenden wir EtherCAT-Klemmen von Beckhoff zur Experimentsteuerung.

Können wir die Elektronik ohne eine SPS CPU ansteuern?

Problemstellung

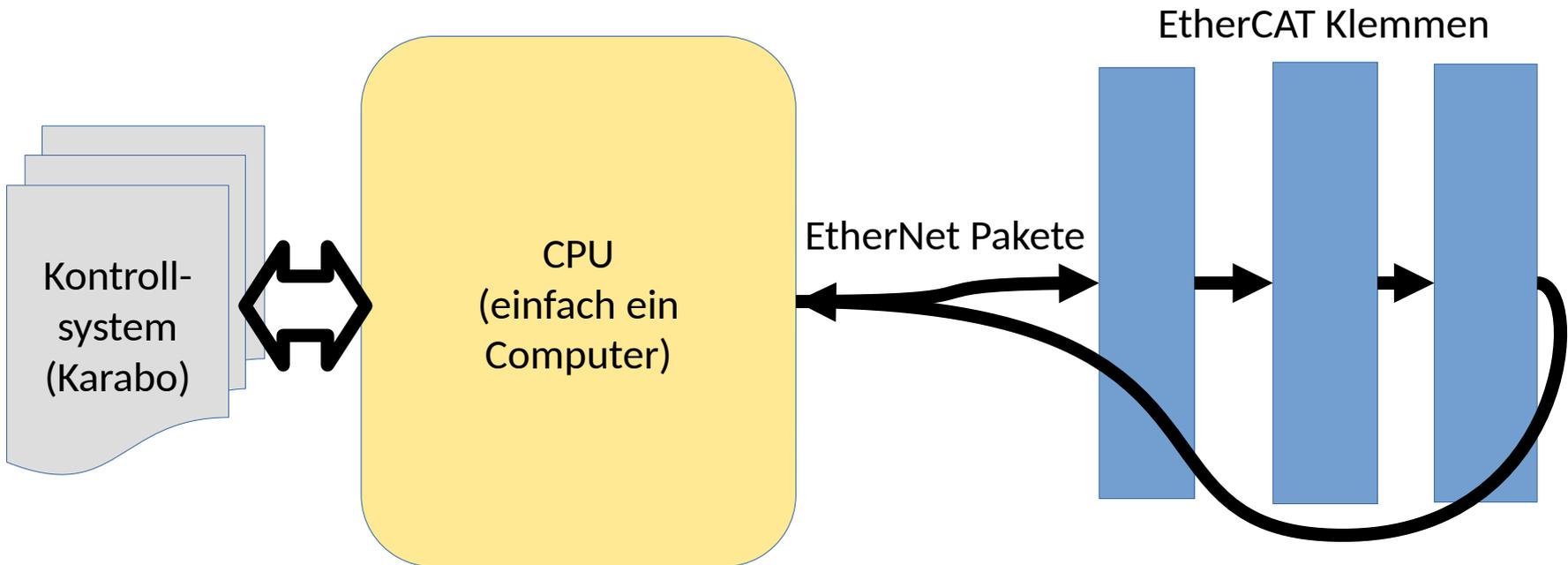


Können wir die Elektronik ohne eine SPS CPU ansteuern?

Ja!

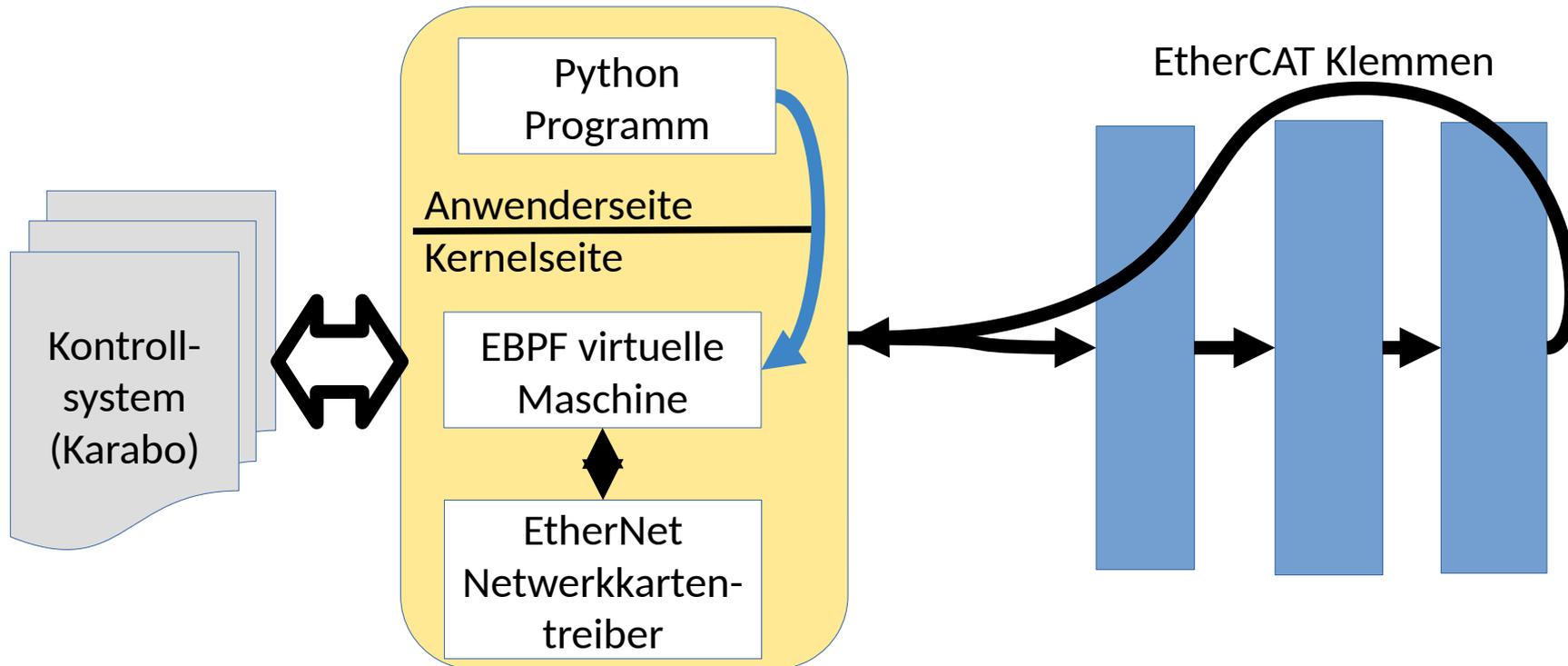
Dazu benötigen wir keinerlei Software von Beckhoff oder anderer Fremdfirmen – wir können alles mit unserem Projekt „**EBPFCat**“ machen

Wie funktioniert EtherCAT?



EtherCAT verwendet normale EtherNet Pakete, die von einer zur nächsten Klemme weitergereicht werden, um am Ende wieder am Anfang anzukommen.

Regelung mit EBPF



- EBPF (Extended Berkley Packet Filter) ist eine virtuelle Maschine im Linux-Kernel die direkt mit der Netzwerkkarte interagieren kann
- Wir können den Regelkreis für die SPS direkt in diese virtuelle Maschine injizieren

Schnelle und Langsame Regelung

Langsame Regelung direkt in Python

Regelschleifen mit etwa 100 Hz

Ventile
Druckmessröhren
Lasershutter
Pumpen

Auch alle Geräte, die per RS232 oder RS485 angesteuert werden

Schnelle Regelung mit EPBF

Regelschleifen bis etwa 10 kHz

Motoren (insbesondere Regelschleife zwischen Encoder und Motor)

Gasdruckregelung

Laserstabilisierung

EBPF-Programme werden vom Linux-Kernel auf Korrektheit verifiziert

Adressierung via fester Nummer



- Normalerweise bestimmt allein die Position der Klemme ihre Funktion
- Dies macht Umbauten schwer: Software und Hardware müssen immer nachgeführt werden
- **Eindeutige Nummer im EEPROM der Klemme:** Software spricht immer mit derselben Klemme, unabhängig von ihrem Ort

Benutzung im Kontrollsystem

The screenshot displays a control system interface. On the left, a 'Projects' tree shows a hierarchy under 'VACUUM' > 'Device Servers' > 'playground/raspi-beckhoff'. The selected item is 'SCS_XRD_AA/MOTOR/SXTZ'. On the right, a 'Property' table shows the current values for various motor parameters. Below the table are control buttons for 'Move', 'Stop', 'Off', 'Step Up', and 'Step Down'. At the bottom, there are buttons for 'Shutdown instance', 'Apply all', and 'Decline all'.

Property	Current value on device	Value
<input type="checkbox"/> motor	PM3	
<input type="checkbox"/> encoder	PM2.channel2	
<input type="checkbox"/> proportional	1.0	
<input type="checkbox"/> encodeStep	5e-05 mm	
<input type="checkbox"/> offset	-9167.095 mm	
<input type="checkbox"/> deadband	0.001 mm	0.001 mm
<input type="checkbox"/> actualPosition	-0.07160814 mm	
<input type="checkbox"/> actualVelocity	0	
<input checked="" type="checkbox"/> isCWLimit	True	
<input checked="" type="checkbox"/> isCCWLimit	True	
<input type="checkbox"/> qstatus	38913	
<input type="checkbox"/> targetVelocity	1000	1000
<input type="checkbox"/> targetPosition		
<input type="checkbox"/> stepSize		

Alle Funktionen können **vom Benutzer** des XFEL Kontrollsystems „Karabo“ eingerichtet werden.

Benutzung im Kontrollsystem

The screenshot displays the Beckhoff control system interface. On the left, a tree view under 'Projects' shows a 'VACUUM' device with a 'playground/raspi-beckhoff' server. A red bracket highlights a group of terminal devices, with the word 'Klemmen' (Terminals) written next to it. The 'SCS_XRD_AA/TERM/PM3' device is selected. On the right, the 'Property' window shows the 'Current value on device' and 'Value' for various parameters. The 'State' is 'ON' and 'Status' is 'ready'. A 'Clear Lock' button is visible. At the bottom, there are buttons for 'Shutdown instance', 'Apply all', and 'Decline all'.

Property	Current value on device	Value
Process ID	708	
State	ON	
Status	ready	
Locked By		
Clear Lock		
Last command		
Logger		
master	SCS_XRD_AA/EC/MAS...	
vendorId	2	
productCode	461451346	
revisionNo	1638400	
serialNumber	50910	
coe		Table Element
max_current	2500	2500
max_voltage	24000	24000
coil_resistance	100	100
motor_emf	0	0
invLogicLim1	True	True
invLogicLim2	True	True

Für jede Klemme gibt es einen Eintrag im Kontrollsystem, sowie für jedes mit einer Klemme verbundene Gerät.

Programmierung eines Motors

```
class Motor(Device):
```

Programmierung eines Motors

```
class Motor(Device):  
    velocity = TerminalVar()  
    encoder = TerminalVar()  
  
    target = DeviceVar()  
    proportional = DeviceVar()
```

Programmierung eines Motors

```
class Motor(Device):
    velocity = TerminalVar()
    encoder = TerminalVar()

    target = DeviceVar()
    proportional = DeviceVar()

    def program(self):
        self.velocity = (self.target - self.encoder) \
            * self.proportional
```

Programmierung eines Motors

```
class Motor(Device):
    velocity = TerminalVar()
    encoder = TerminalVar()

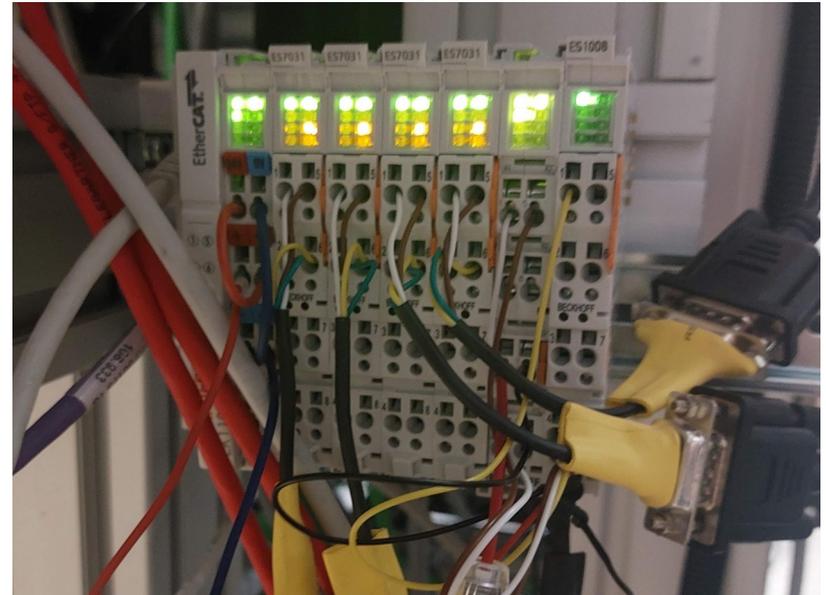
    target = DeviceVar()
    proportional = DeviceVar()
    max_velocity = DeviceVar()

    def program(self):
        self.velocity = (self.target - self.encoder) \
            * self.proportional

        with self.velocity > self.max_velocity:
            self.velocity = self.max_velocity
        with self.velocity < -self.max_velocity:
            self.velocity = -self.max_velocity
```

Voraussetzungen

- Ein Rechner auf dem Linux läuft
 - z.B. irgendein Laptop
 - Oder Raspberry Pi 4 - Erfahrungsgemäss das beste
- Administrator / „root“ - Rechte
- Ein Kontrollsystem obendrüber
 - Das XFEL Kontrollsystem „Karabo“ wird bereits unterstützt
 - Andere Kontrollsysteme (Tango, EPICS, DOOCS...) denkbar
- Auch sehr kleine Projekte sind einfach umsetzbar



Zusammenfassung

- Wir können Beckhoff EtherCAT Steuerungen ohne eine Beckhoff CPU steuern
- Langsame Regelkreise werden direkt in Python geschrieben, schnelle können direkt in den Linux Kernel injiziert werden
- Klemmen haben feste Nummern – das vereinfacht Änderungen

Alles ist offen verfügbar unter <https://github.com/tecki/ebpfcats>

Dank an meine Gruppe – SCS beim European XFEL, insbesondere Carsten Broers für die Hardware, und Jan Tolkiehn von EEE - XFEL