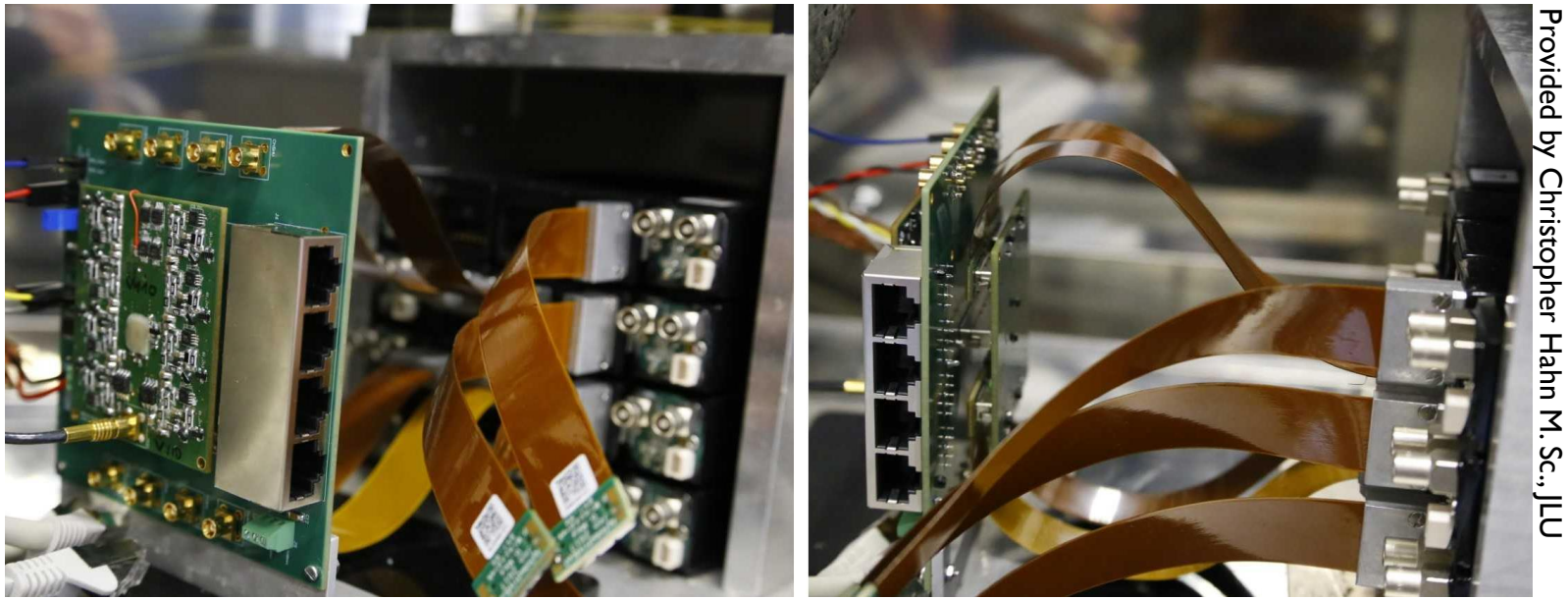
IMPLEMENTATION OF THE BARREL FRONT END SLOW CONTROL INTO EPICS

BY NICLAS FIEDLER

JUSTUS-LIEBIG-UNIVERSITÄT GIESSEN

II. Physikalisches Institut – Experimental Physics II



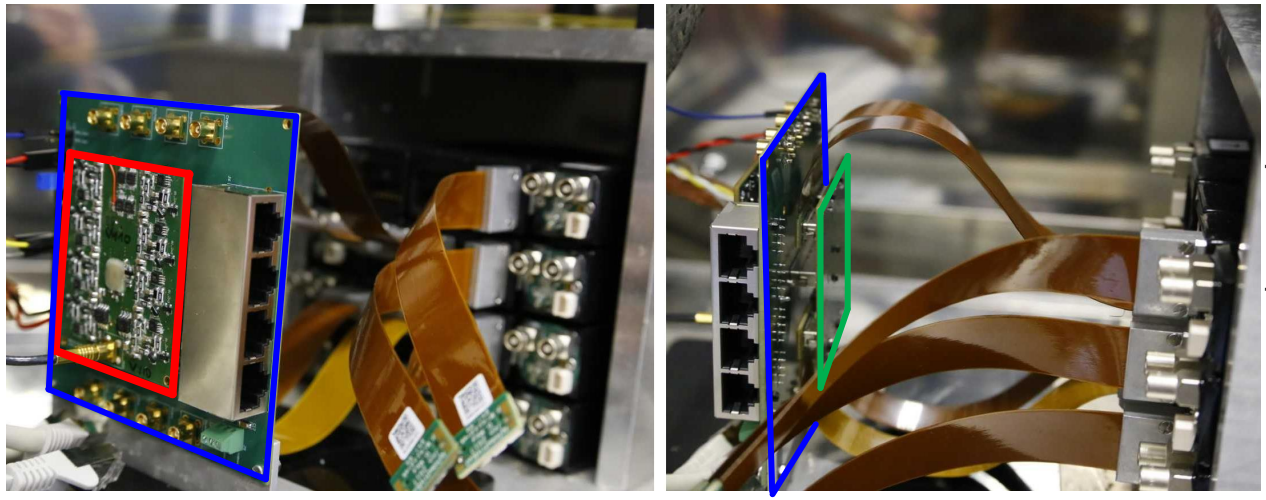
Provided by Christopher Hahn M. Sc., JLU

Photographs of a test setup for the Front-End electronics

PANDA

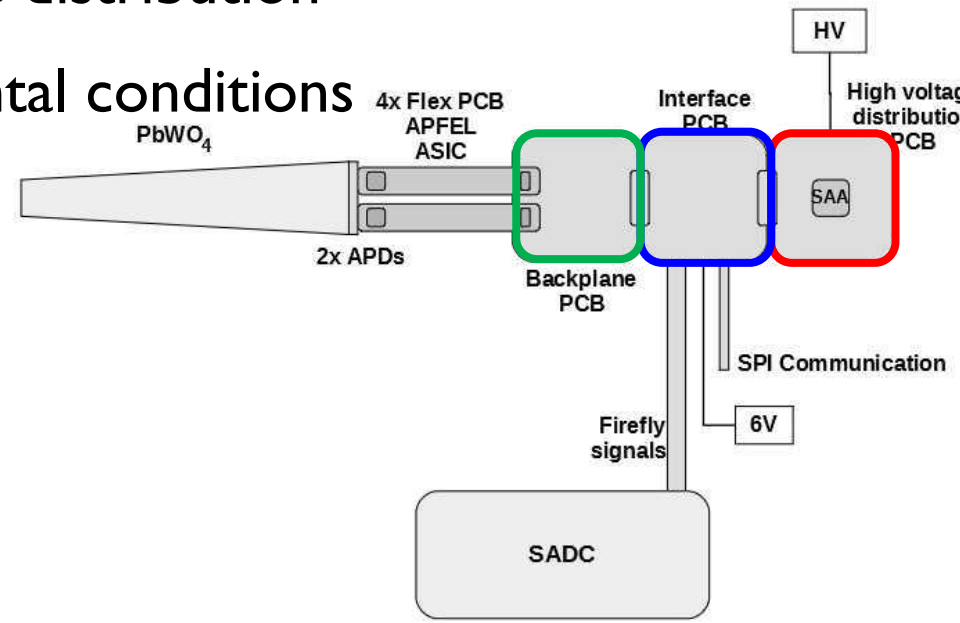
FRONT-END ELECTRONICS

- High Voltage Distribution Board: Power supply and adjustment for APDs
- Interface Plate: SPI communication and low voltage distribution
- Backplane: APFEL communication and environmental conditions



Provided by Christopher Hahn M. Sc., JLU

Photographs of a test setup for the Front-End electronics

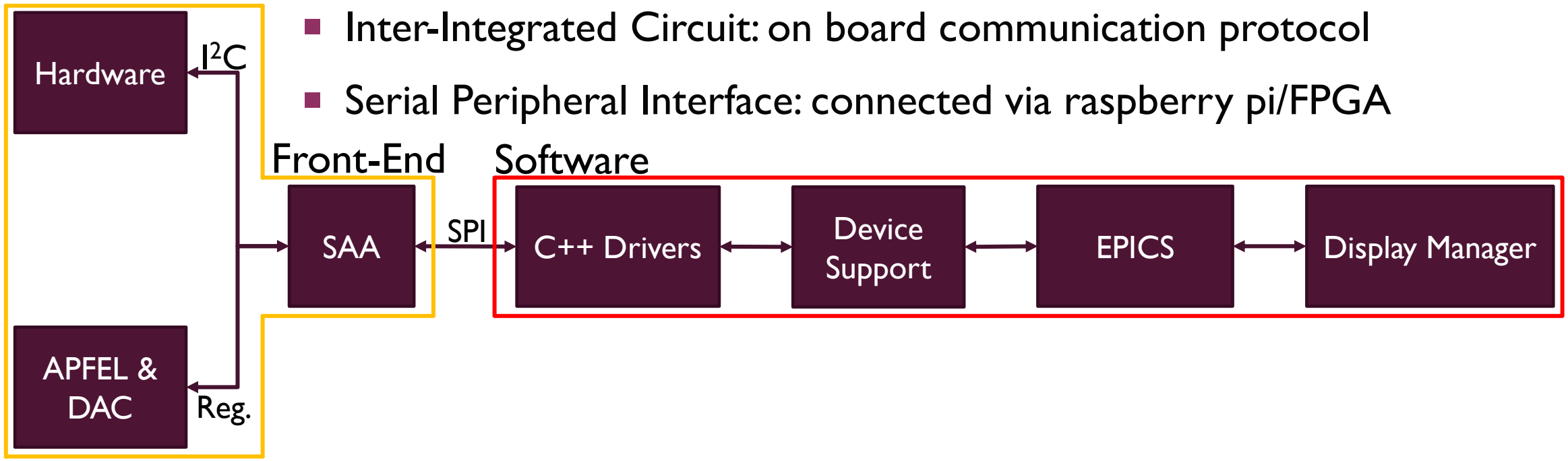


Provided by Christopher Hahn M. Sc., JLU

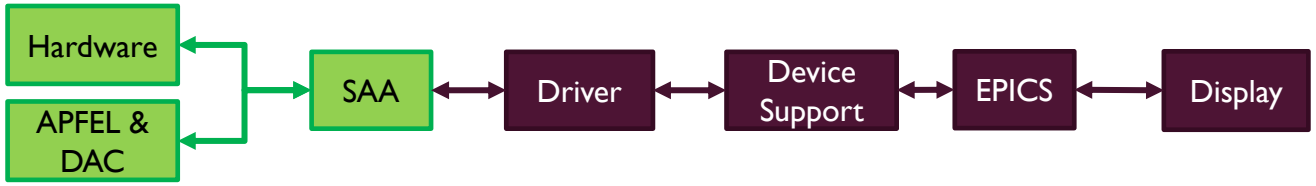
Schematic structure of the Front-End electronics

COMMUNICATION OVERVIEW

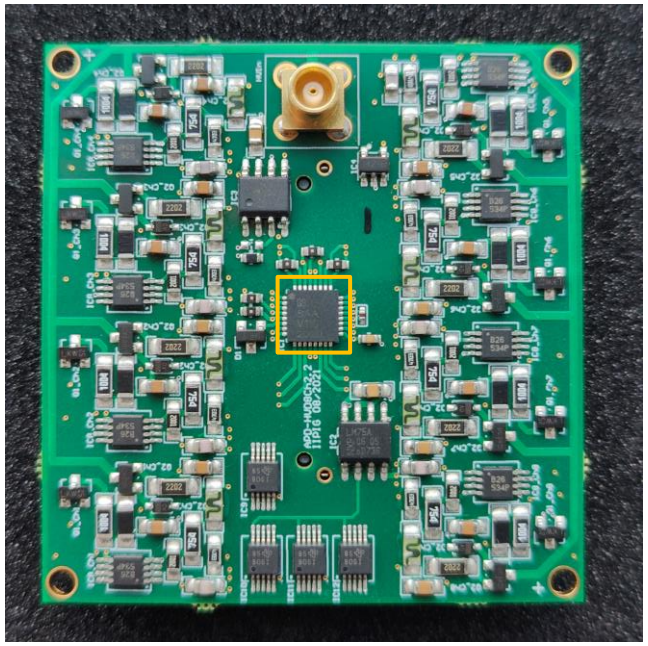
- Inter-Integrated Circuit: on board communication protocol
- Serial Peripheral Interface: connected via raspberry pi/FPGA



SERIALADAPTER ASIC & HARDWARE

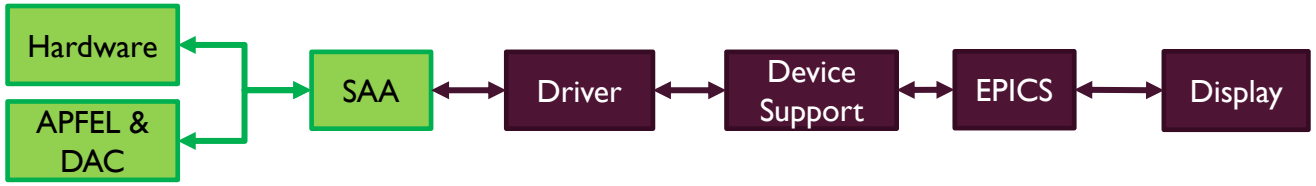


- SAA controls the Front-End electronics
- Hardware: SHT21, LM75, ADS1115, EEPROM, PCA9554
  - Environmental condition, APFEL calibration, ID, I/O expander
  - I2C-accessible devices
- APFELs & DACs are controlled via register values



Photograph of the high voltage distribution board

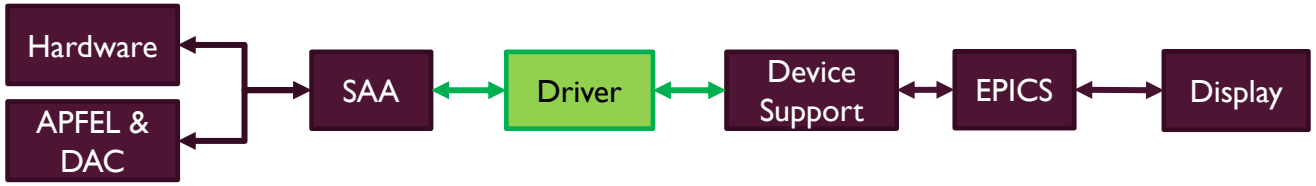
SERIALADAPTER ASIC & HARDWARE



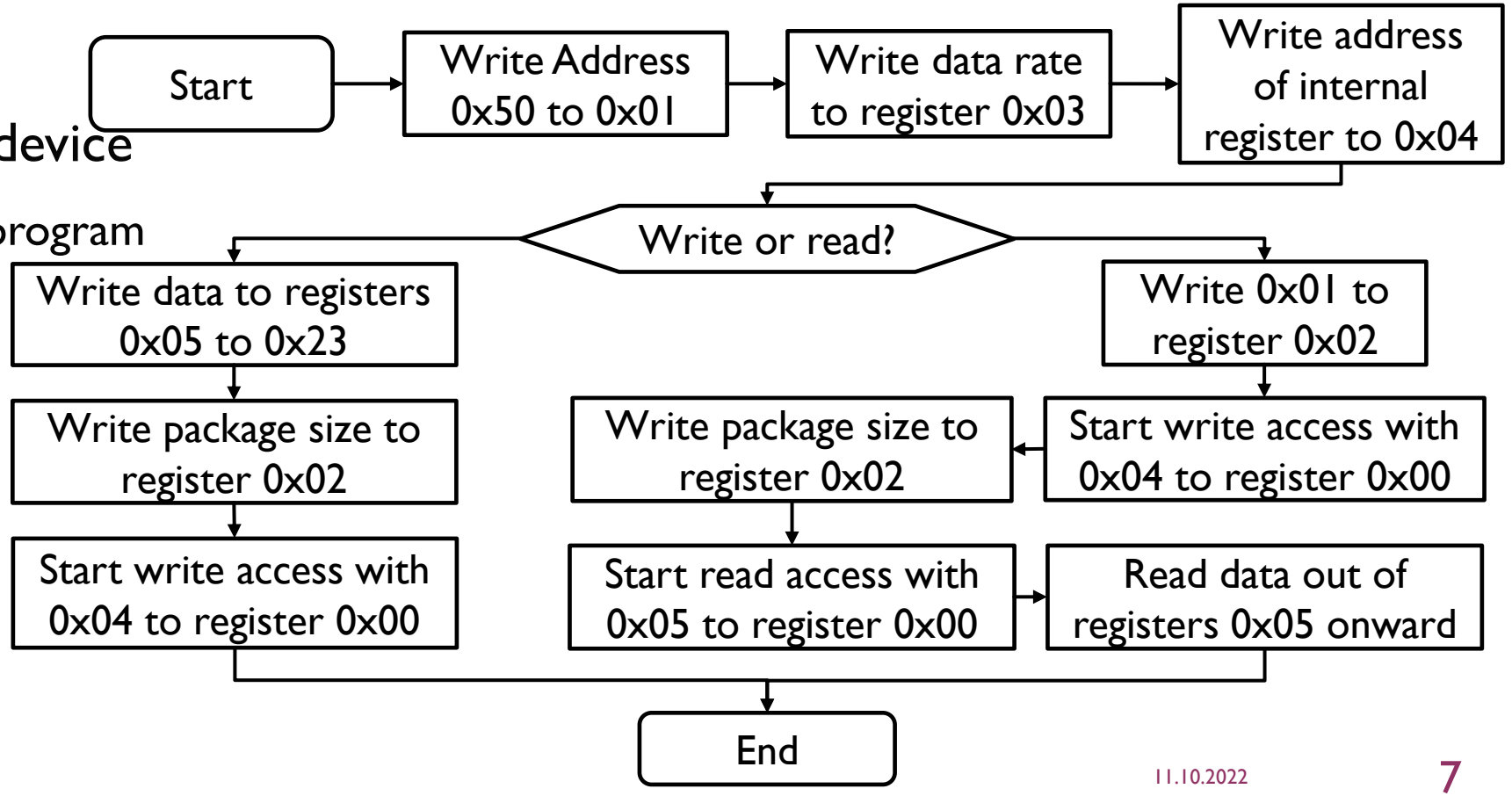
- SPI access writes SAA's registers
- I2C usage
  - Configuration registers 0x01-0x03
  - Transmission data 0x04-0x23
  - Start access through register 0x00
- DAC register values adjust the high voltage
- Control APFELs via registers 0x48 and 0x49

Register	Description
0x00	General configuration
0x01	I2C address
0x02	I2C package size
0x03	I2C bit rate
0x04-0x23	I2C data
0x40-0x47	DAC control
0x48-0x49	APFEL
0x4a	Hamming error
0x4b-0x4c	MPIO access
0xff	Version

DRIVERS

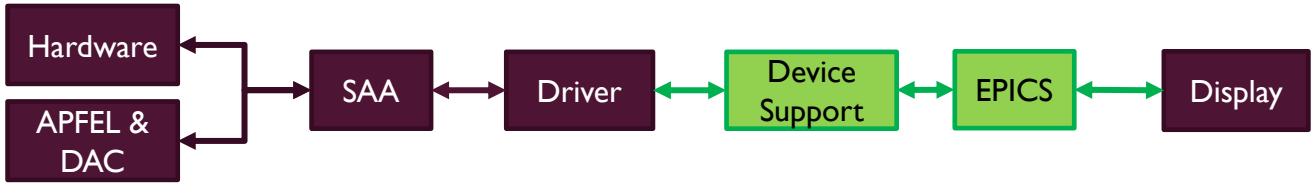


- C++ implementation
- Individual driver per device
  - Included in one main program
- E.g. EEPROM

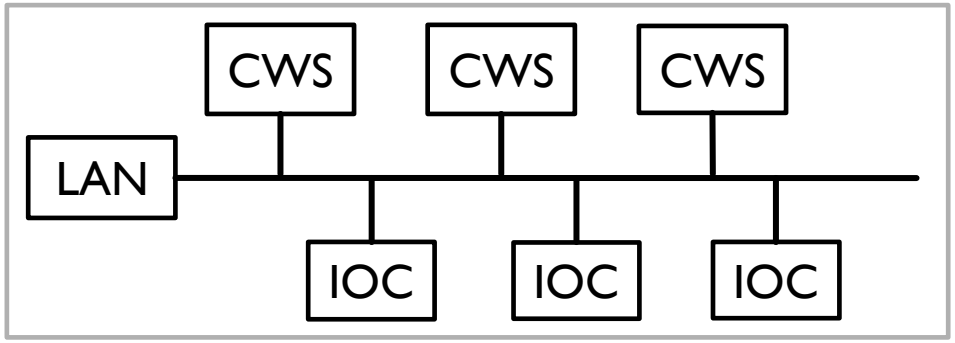


Experimental Physics and Industrial Control System

EPICS OVERVIEW

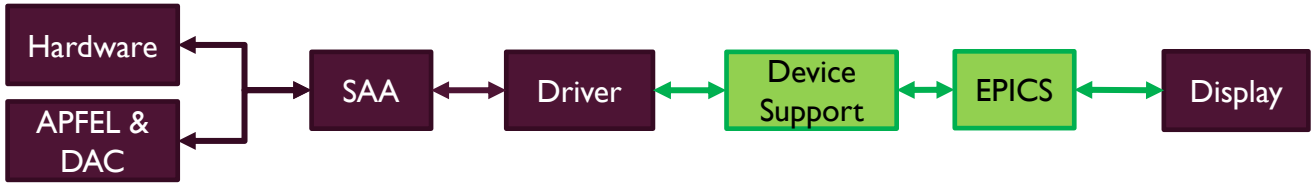


- Experimental Physics and Industrial Control System
- Distributed control system
  - Designed for large-scale experiments
- Interface between the control system and client workstation
- Uses Input/Output Controllers as servers
  - Collect/provide experimental data

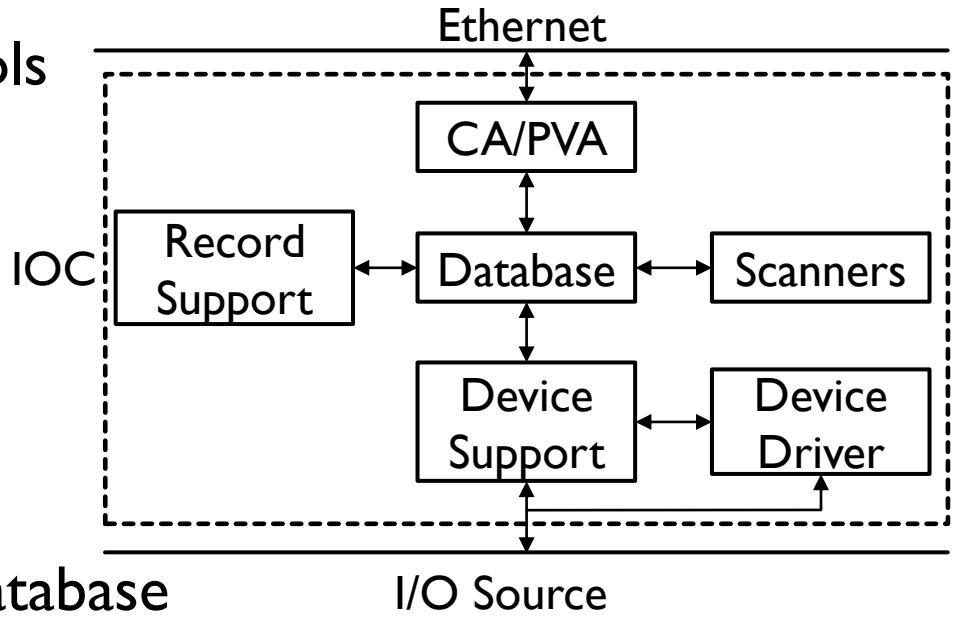


Schematic overview of a simple EPICS control system structure

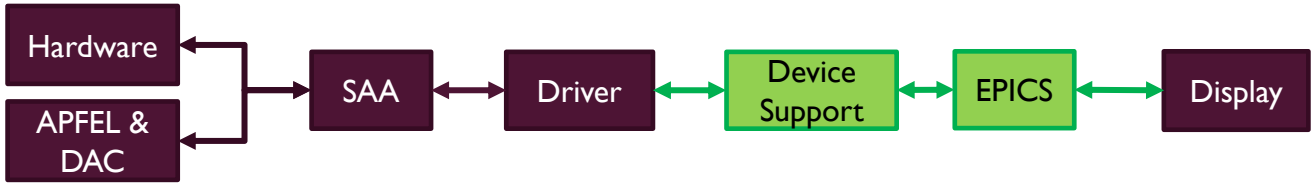
EPICS IOC



- Interface between an I/O sources and ethernet
- Channel access and process variable access protocols
  - Links ethernet to database
- Database consists of records holding data
- Scanners process records
- Record support defines record attributes
- Device support interfaces the I/O sources to the database
  - Split into direct device support and indirect using drivers



EPICS IMPLEMENTATION

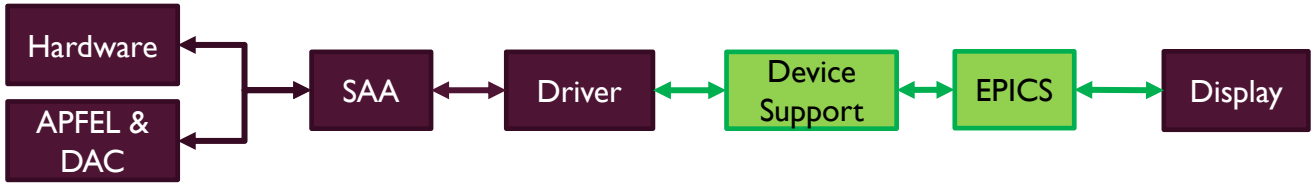


- Implementation using EPICS Base 7
- Records holding attributes in fields
- Substitution file holding macros
  - Easily scalable without record knowledge
- Record type defines behaviour and properties
- Individual analogue and binary I/O records per device as needed

```
record(ai, SAA$(boardid):SHT21:TEMP){
        field(DESC, "Read temperature from SHT21.")
        field(DTYP, "DevSHT21")       # Device type
        field(EGU, "°C")              # Engineering units
        field(SCAN, "5 second")       # Scan properties
        field(INP, "@$(bitrate).$(boardid)") # Input Link
}
```

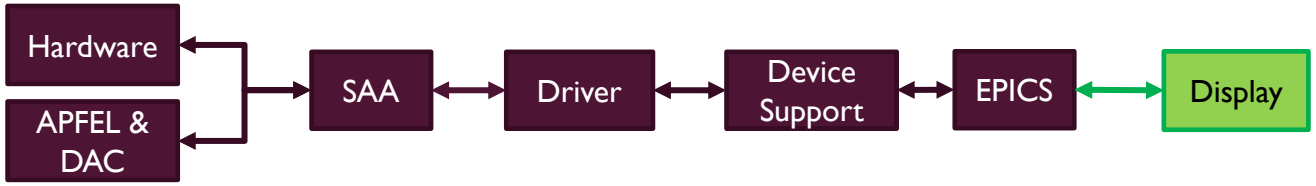
Simple example record for SHT21 temperature measurements

EPICS DEVICE SUPPORT

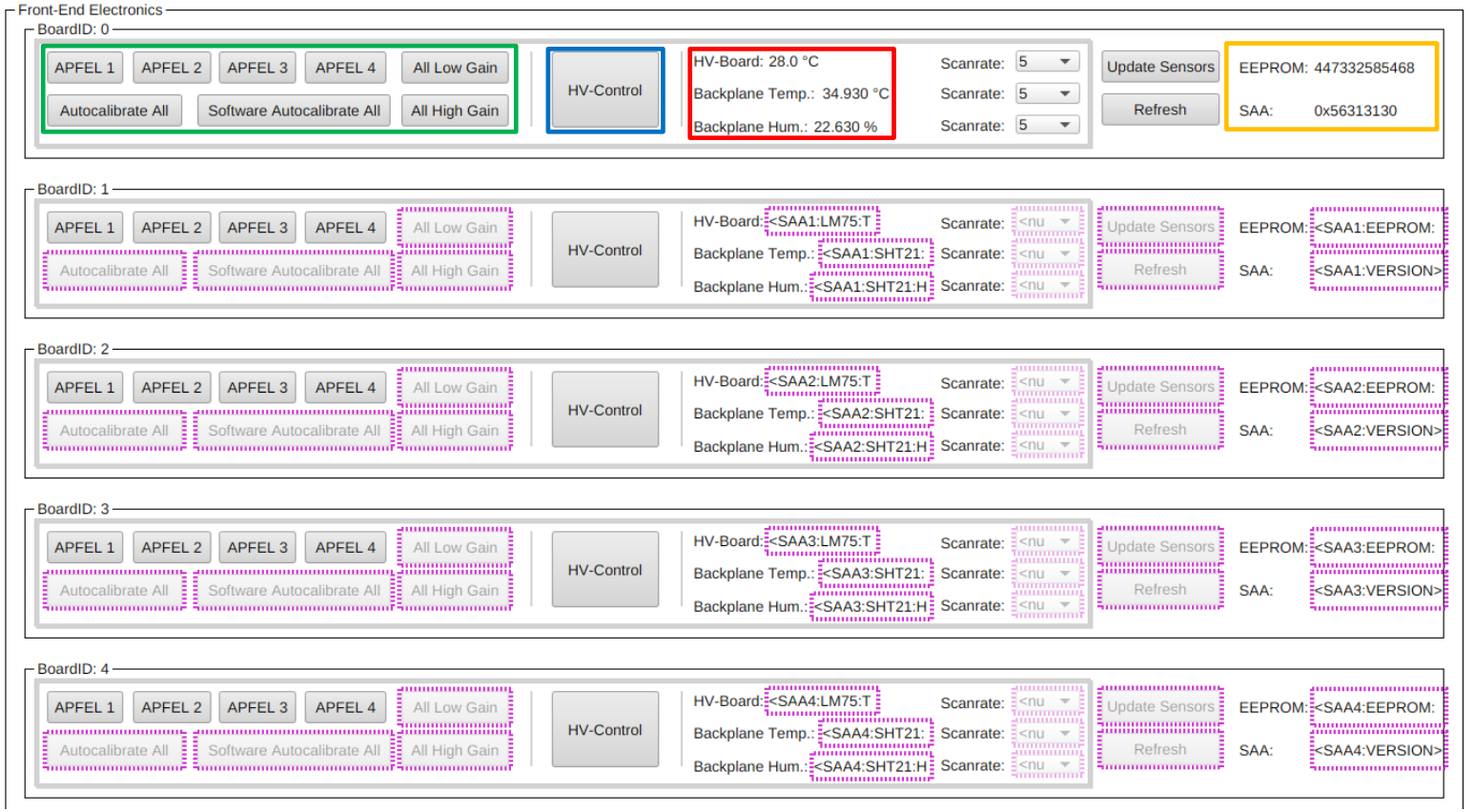


- Device support implemented in C++
- Links device drivers to IOC's database
- Database definition connects records with C++ implemented device support
- Consists of initialization and read/write routines
  - Others are not used (e.g. report)
- Initialization routine loads parameters
- Read/write routines access devices through drivers at runtime

DISPLAY MANAGER

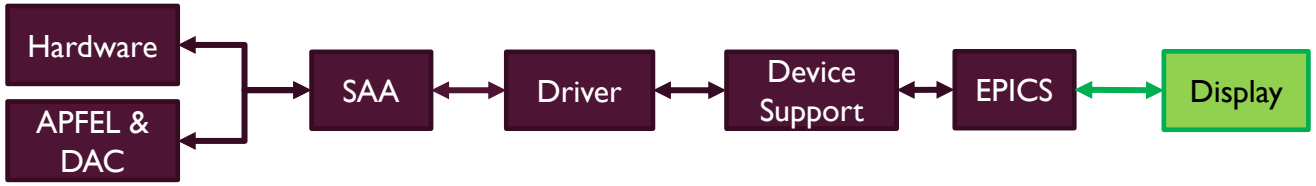


- CSS Phoebus
  - PANDA DCS based EPICS and CSS
- Individual Front-End units
- Individual APFELs
- HV-Control
- Environmental conditions
- Distinguishability



Display in test run of one Front-End unit

DISPLAY MANAGER



- High voltage control for each channel
- APFELs with two APDs
- Display current and voltage
  - Variable scan rate
- Individually settable DAC values

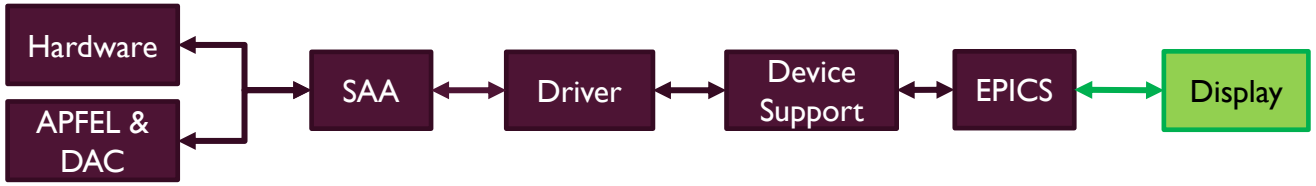
HV-Control

Back | BoardID: 0 | Set Input Voltage: 400

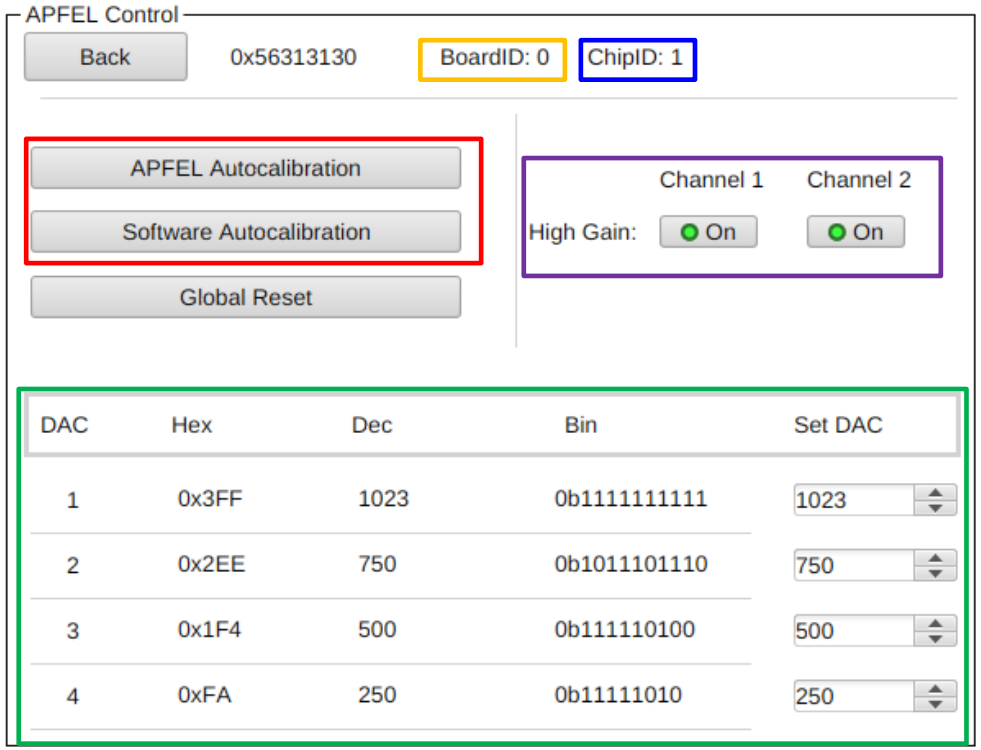
	ChipID:APD	Current	Scanrate	Voltage	Scanrate	DAC	Set DAC
1	01:01	0.0510 nA	P	397.4489 V	P	1023	1023
2	01:02	0.0509 nA	P	401.2101 V	P	900	900
3	02:01	0.0779 nA	P	398.7754 V	P	800	800
4	02:02	0.0820 nA	P	391.5552 V	P	700	700
5	03:01	0.0479 nA	P	389.9936 V	P	500	500
6	03:02	0.0507 nA	P	390.3126 V	P	400	400
7	04:01	0.0514 nA	P	386.8872 V	P	300	300
8	04:02	0.0494 nA	P	385.3928 V	P	150	150

Control All | Read Current | Read Voltage | All Passive | All 0.2 Hz | 0x56313130

DISPLAY MANAGER



- Each APFEL per Front-End unit
- Start software or hardware autocalibration
- Change between low and high gain per channel
- Set individual DAC values



OUTLOOK

- EPICS implementation working in a test setup
- Further tests with an FPGA and the preseries slice
- Complete Daisy Chain needs to be implemented
  - Currently, only single access of individual chain members is possible
- EPICS operating ranges and alarms for sensors need to be added
- Unification of EPICS record notations