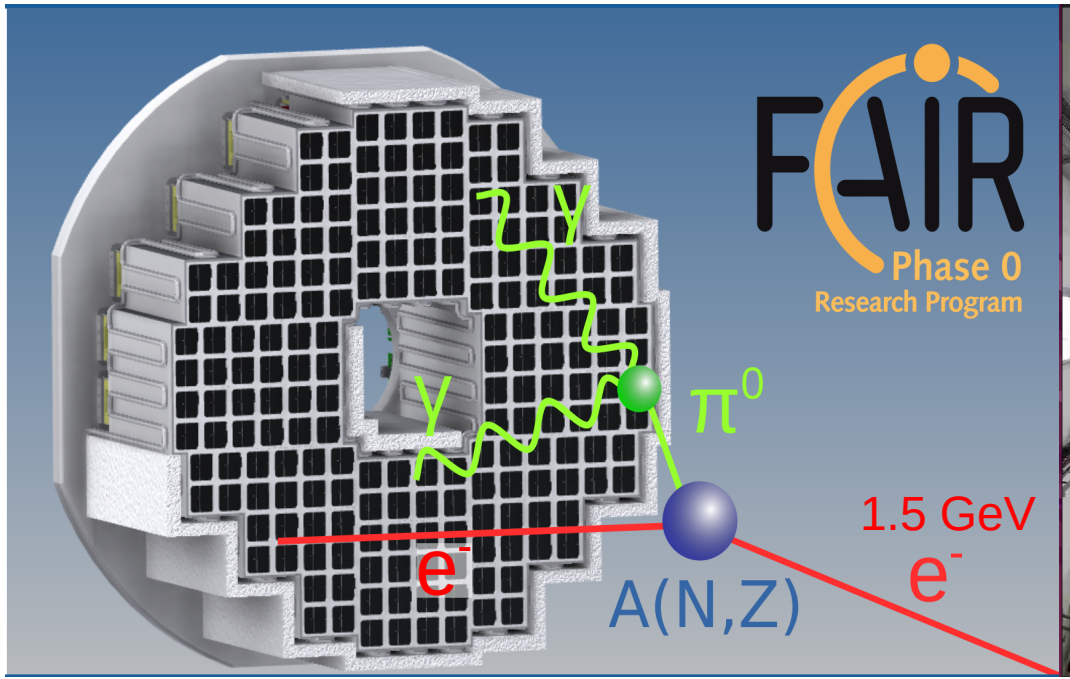
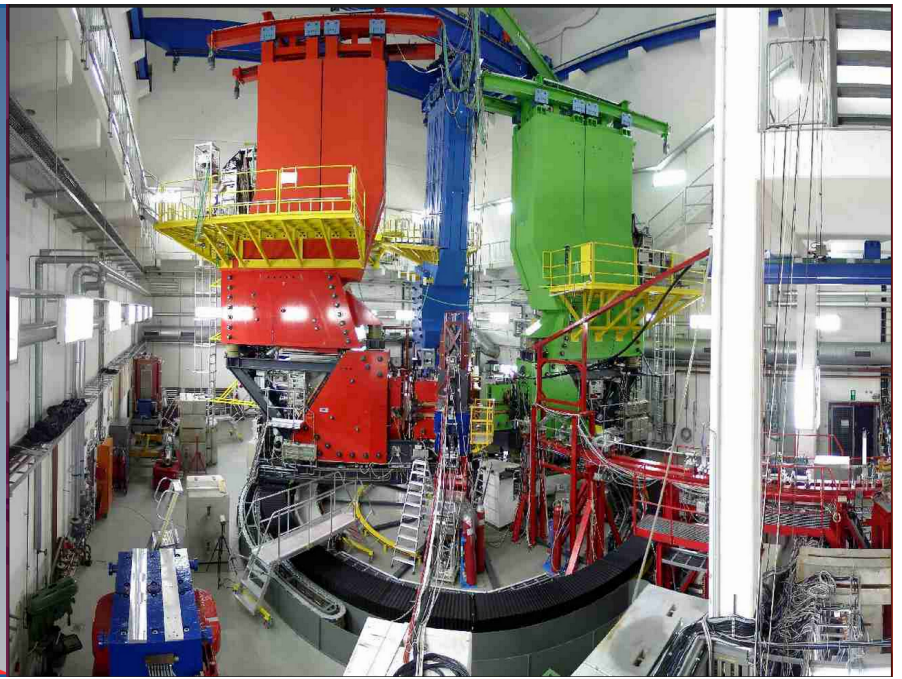
Development of a Detector Control System for PRIMA-exp at MAMI

13 June 2023
PANDA Collaboration Meeting 23/2
Ravi Gowdru Manjunata

PRIMA EXPERIMENT

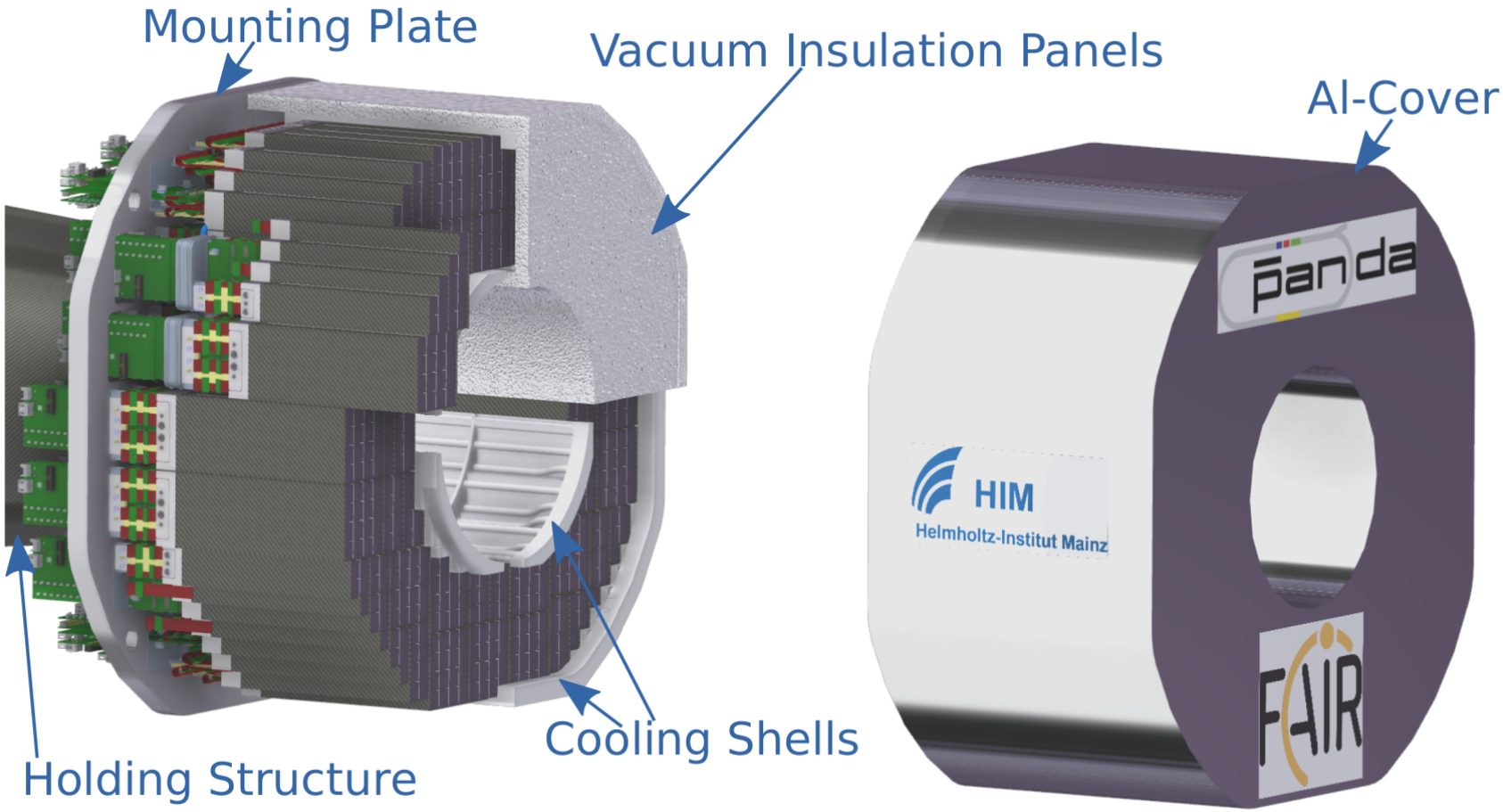




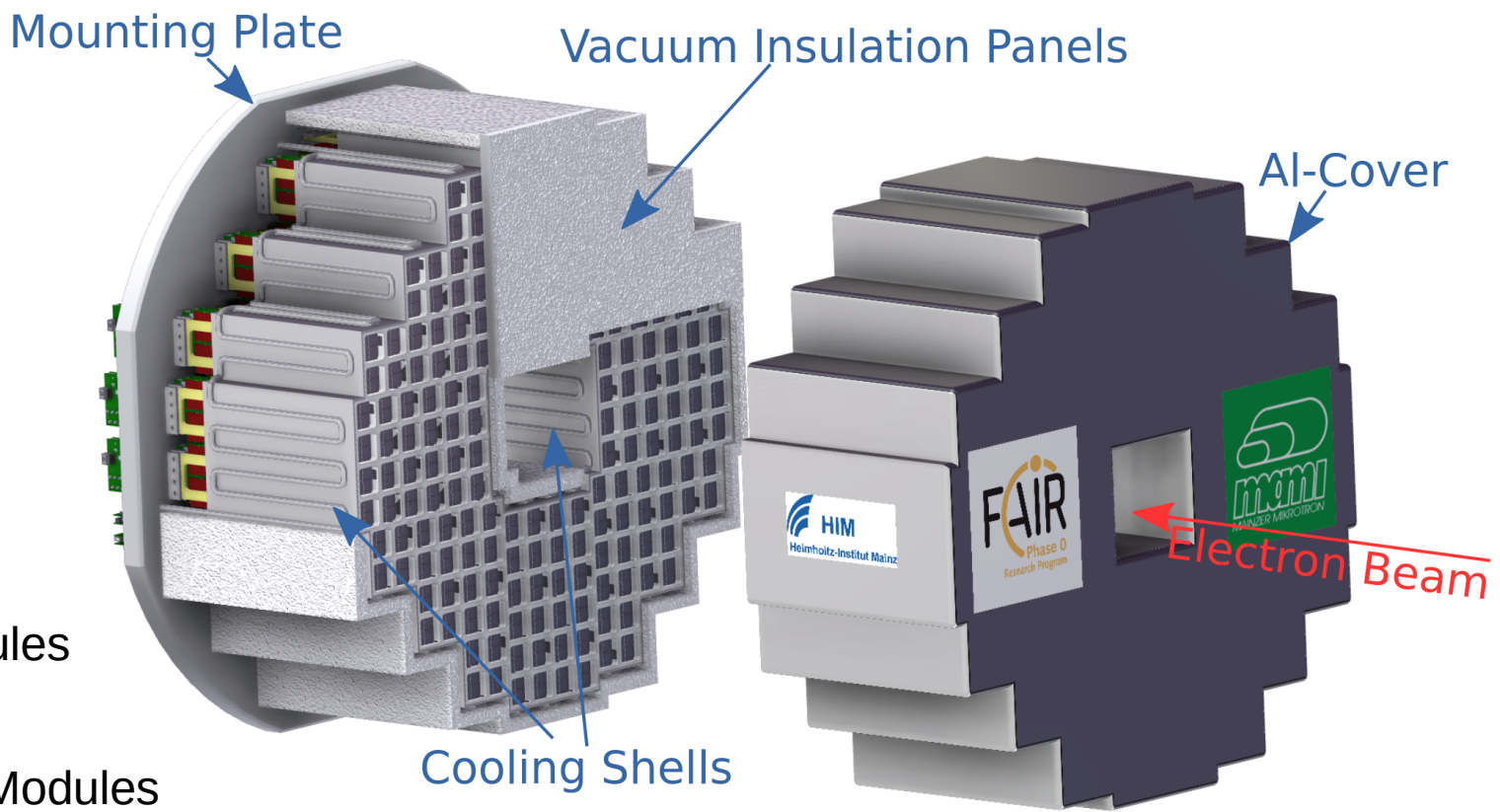
EMC-Detector

A1-Hall at MAMI

P̄ANDA backward EMC



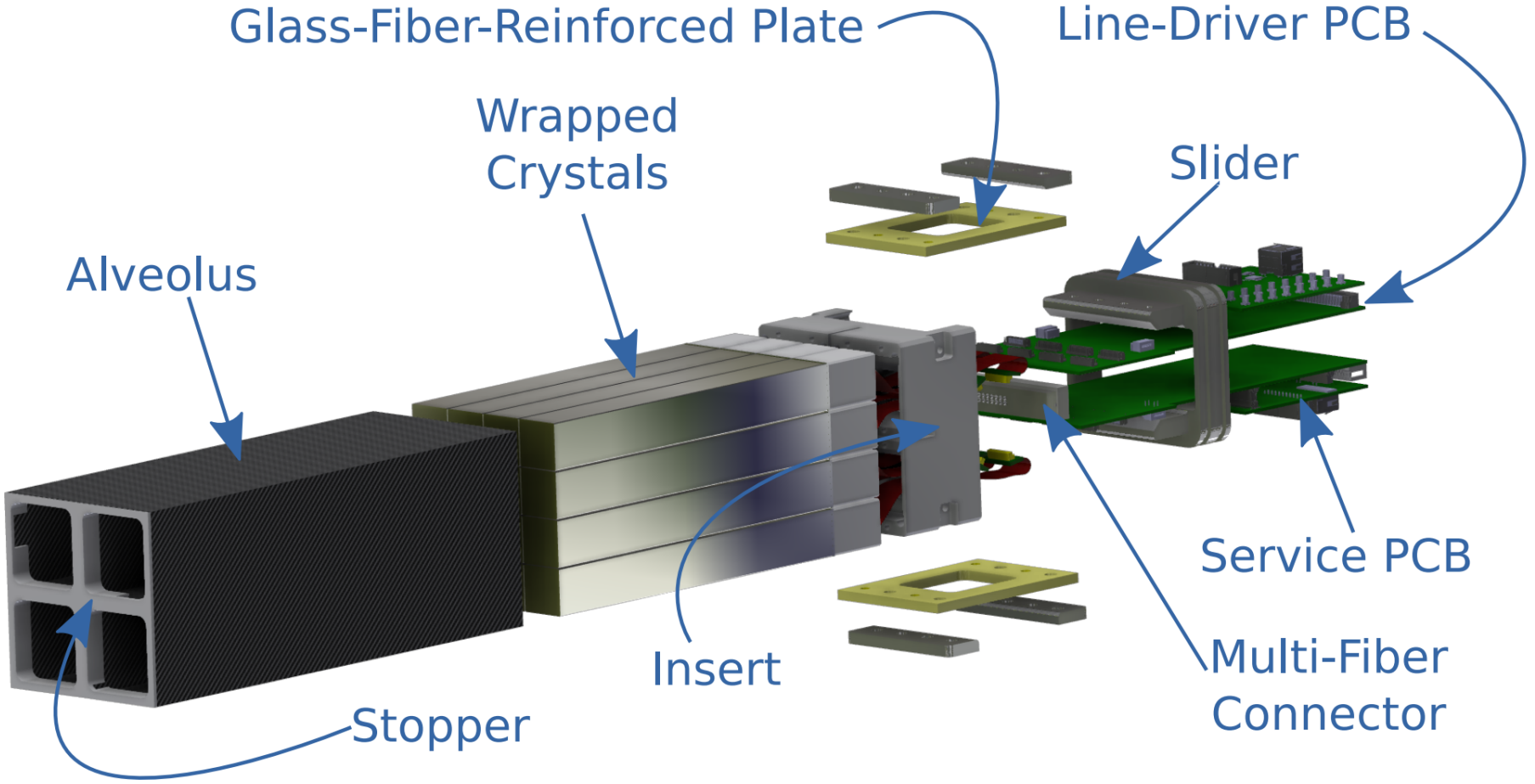
Electromagnetic Calorimeter



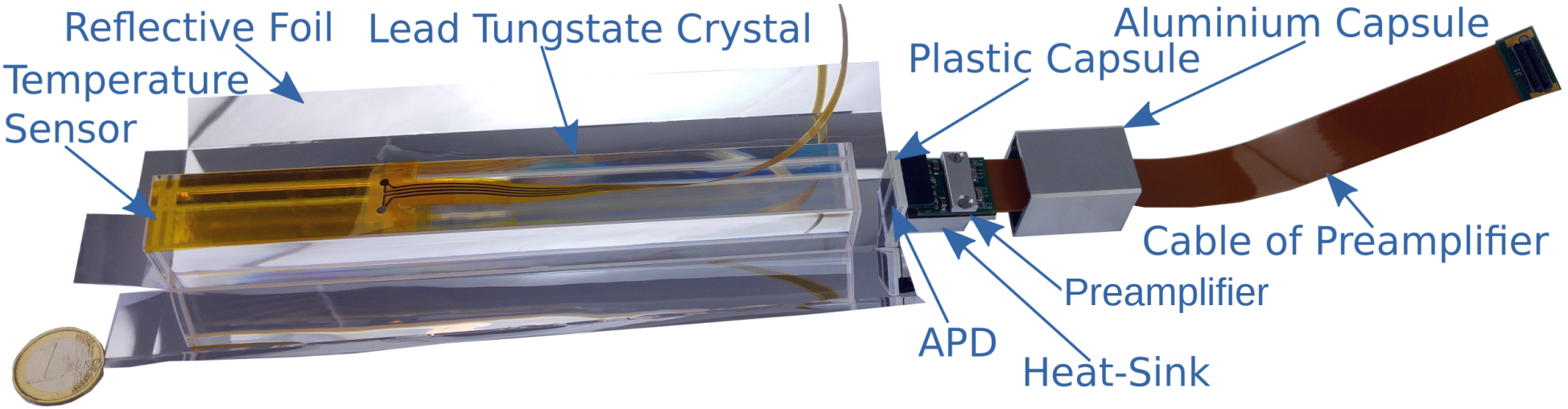
48 Sub-Modules
768 Crystals

32 Full Sub-Modules
16 half Sub-Modules
640 Crystals

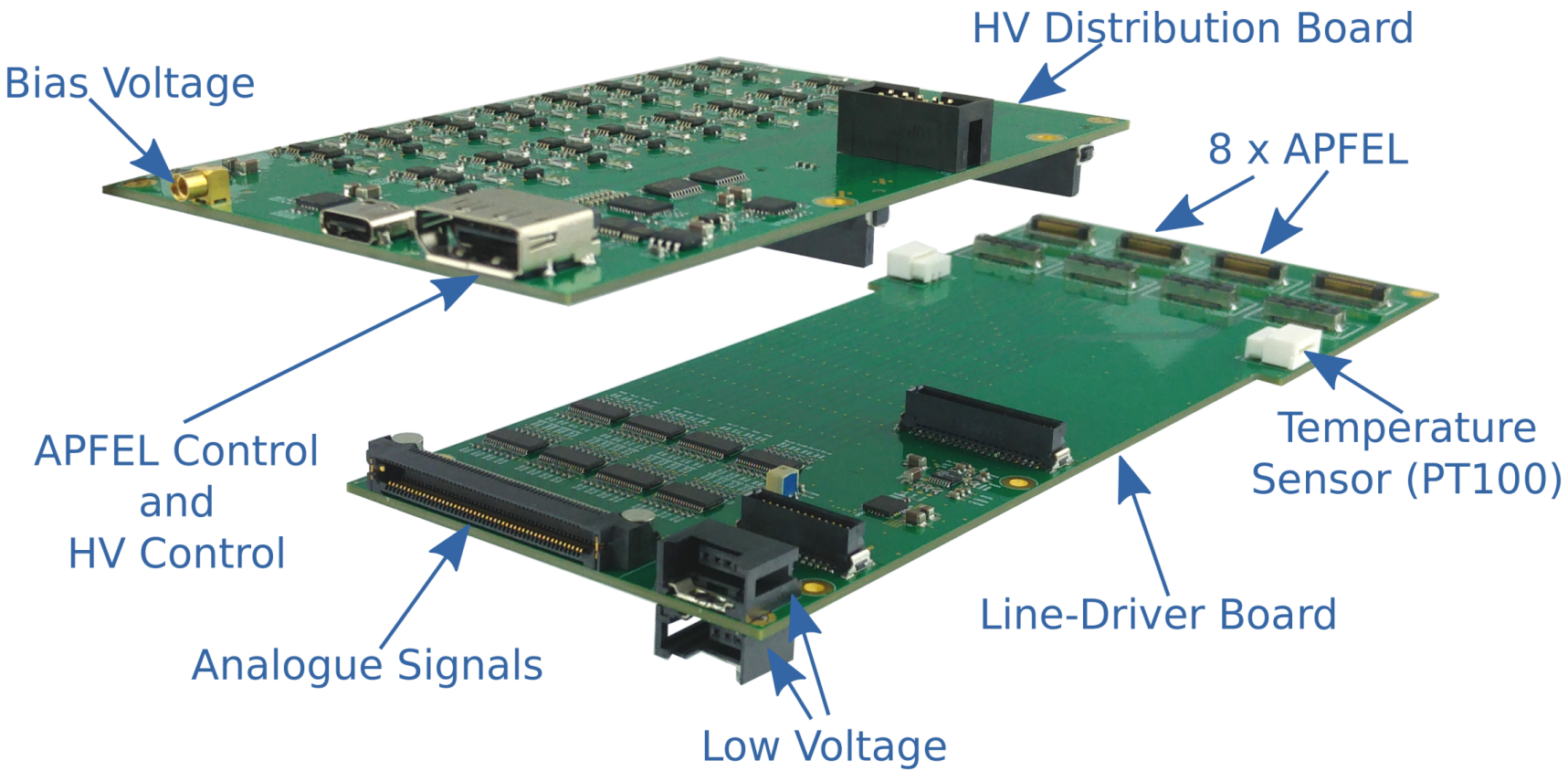
The Sub-Module



Single Unit


Reflective Foil
Lead Tungstate Crystal
Aluminium Capsule
Temperature Sensor
Plastic Capsule
Cable of Preamplifier
Preamplifier
APD
Heat-Sink

Front-End Boards



Need for a Control System

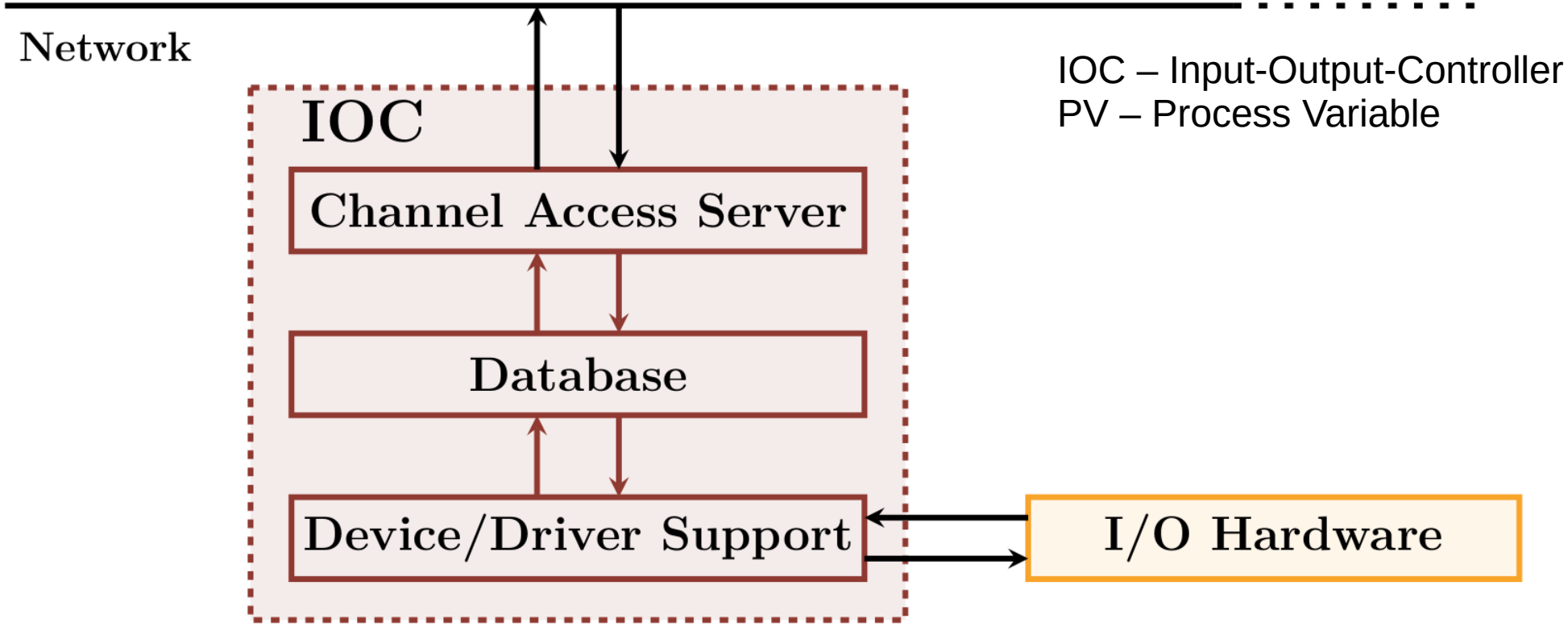
Component	Parameters	parameter-count per sub-module
Crystal	1. Temperature sensors 2. Light pulse (16x bias, 16x enable, frequency, width, output)	4 35
Pre-amps	1. Amplification 2. 4 x DAC-values 3. Auto-Calibration 4. Test pulse	16 64 16 16
Front-End Boards	1.DAC-voltages 2.ADC-voltages/currents 3.Temperature sensors	32 64 4
Voltage supply	1. High Voltage 2. Low Voltage	2 1
		Total = 254

About 10 K Parameters for the whole detector accounted so far.

EPICS

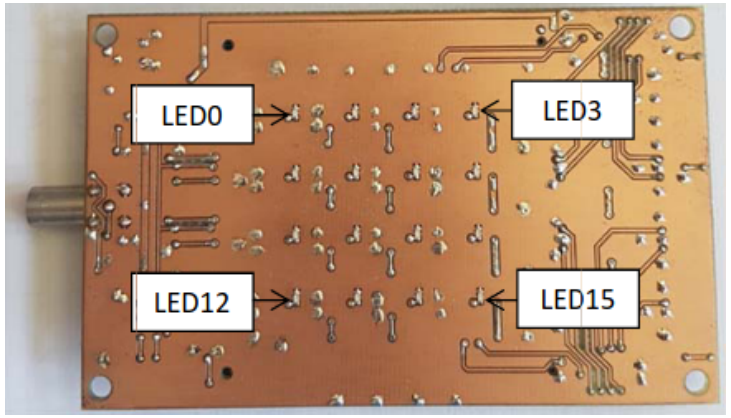
- Experimental Physics and Industrial Control System
- EPICS Collaboration
  - Argonne National Laboratories, Los Alamos National Laboratory, etc.
- Distributed Control System
- Scalable system
- Support wide range of components
- Archiving through caMonitor.

EPICS-IOC



IOC – Input-Output-Controller
PV – Process Variable

IOC for LED Matrix




LEDMatrix.proto


LEDMatrix.db


st.cmd

- Install EPICS-BASE
- $ ***makeBaseApp.pl***
- .db – configure database
- .proto – between IOC and Device
  - Stream Device Protocol
  - Could change based on the device
- .cmd – startup script.
- $ ***docker run ***
  ***-v ${PWD}/volume:/config ***
  ***paluma.ruhr-uni-bochum.de/epics/ioc ***
  st.cmd

IOC for LEDMatrix

```
readFrequency{
   out "frequency ?";
   in "%u";
}

setFrequency{
   out "frequency %u";
}
```

LEDMatrix.proto

```
record (ai, "PRIMA:CAL:LEDMatrix:readFrequency") {
  field (DTYP, "stream")
  field (SCAN, "1 second")
  field (EGU, "Hz")
  field (INP, "@LEDMatrix.proto readFrequency $(BUS)")
}

record (ao, "PRIMA:CAL:LEDMatrix:setFrequency"){
  field (DTYP, "stream")
  field (EGU, "Hz")
  field (OUT, "@ledMatrix.proto setFrequency $(BUS)")
}
```

LEDMatrix.db

ai – analog input
ao – analog output
DYTP – datatype

IOC for LEDMatrix

```
epicsEnvSet ("STREAM_PROTOCOL_PATH","/config")
dbLoadDatabase( "/epics/ioc/dbd/epicsIoc.dbd", 0, 0 )
epicsIoc_registerRecordDeviceDriver( pdbbase )




# Loads records
dbLoadRecords ("/config/ledMatrix.db"," BUS=PS1 ")


# connect to the device ... IP-Address ! Port 7 used by
FPGA, see manual
drvAsynIPPortConfigure( "PS1", "192.168.1.23:7")


iocInit()
```

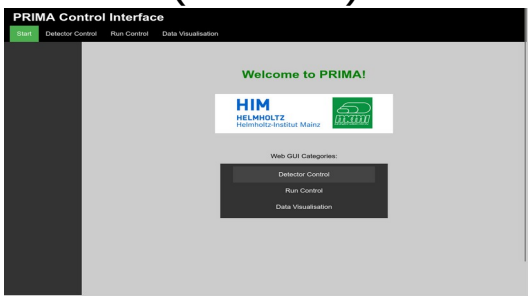
st.cmd

PyEpics

```
from epics import PV

class LEDMatrix:
  def __init__(self):
      self.readFrequency = PV(PRIMA:CAL:LEDMatrix:readFrequency)
      self.setFrequency = PV(PRIMA:CAL:LEDMatrix:setFrequency)



ledMatrix = LEDMatrix()
ledMatrix.readFrequency.get()
ledMatrix.setFrequency.put(1000)
```

Web-Interface

web interface
(html/JS)



python™

web server

django

Detector interface scripts

...

Network/EPICS

Devices

HV

LV

L. Pulser

...

The View Function

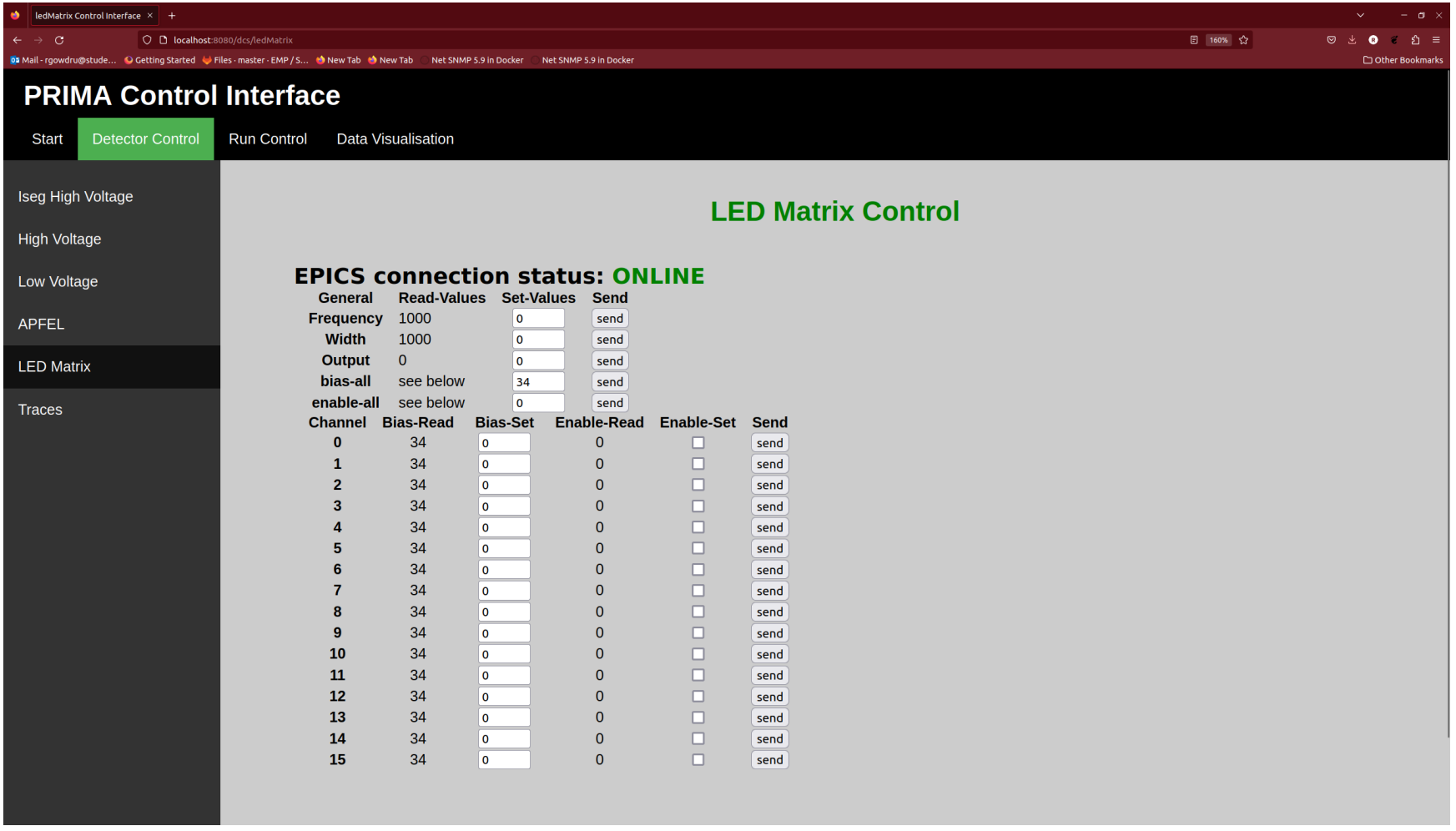
```
def ledMatrixView(request):
    if request.method == 'POST':
        print('******* LEDMATRIX POST ******* ')
        req = request.POST['request_type']
        print('request type:',req)
        data = {'reply':'none'}

        if req == 'refresh':
            print("refresh request ledMatrix")
            data = ledMatrix.readData()
            print(data)

        if req == 'change':
            print("change request ledMatrix")
            reqdata = json.loads(request.POST['data'])
            ledMatrix.putData(reqdata)

        response =JsonResponse(json.dumps(data),safe=False)
        return response
    else:
        print('******* LEDMATRIX GET ******* ')
        data = ledMatrix.readData()
        html = render(request, 'templates/ledMatrix.html',
                {'dcs': 'active','ledMatrix':'active-app', 'data': data})
        return html
```

Views.py

```
from epics import PV

class LEDMatrix:
  def __init__(self):
      self.readFrequency = PV(PRIMA:CAL:LEDMatrix:readFrequency)
      self.setFrequency = PV(PRIMA:CAL:LEDMatrix:setFrequency)

  def readData(self):
      data['frequency'] = self.readFrequency.get()
      return data

  def putData(self, data):
       parameter = data['parameter']
       value = data['value']
       if parameter == "frequency":
          self.setFrequency.put(int(value))
```

PyEpics file.

PRIMA Web-Interface



Current Status And Next Steps

- IOCs : Voltage Supply, APFEL, Front-End Boards, LED-Matrix.
  - Pending: Temperature sensors and Cooling system
- Web-Interface: Improvements with visual representations
- Data Archiving and Interface to browse archived data
- Alarm System

Summary

- Detector Components of EMC
- Parameters to be monitored
- Brief Introduction to EPICS
- Input-Output-Controller configuration
- PyEpics and Web-Interface (Django frame-work)