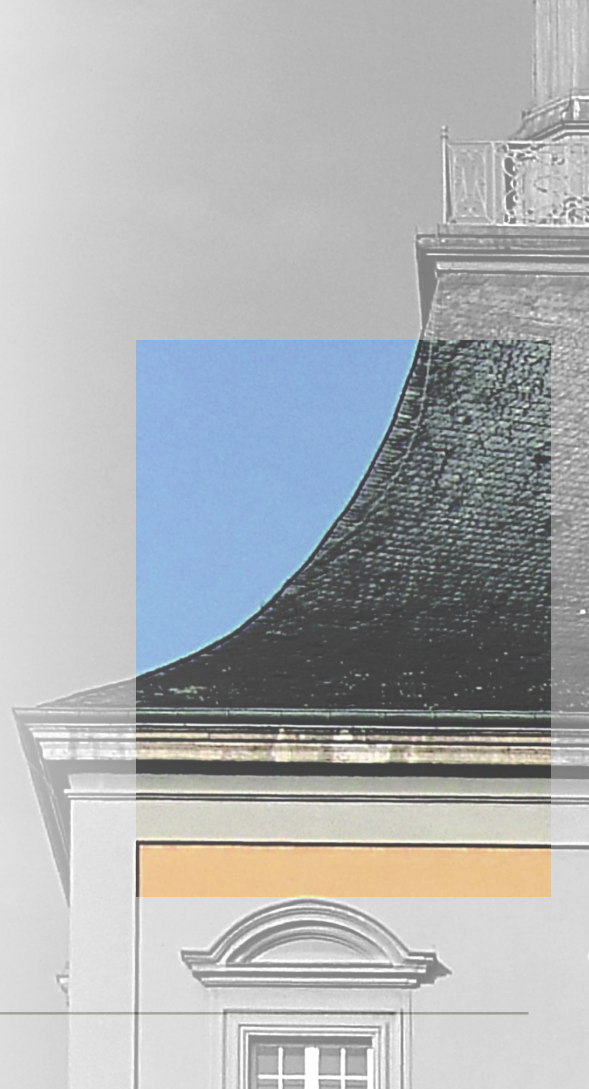
UNIVERSITÄT BONN

SI LAB
Silizium Labor Bonn

EDDA MEETING

EXPERIENCES OF BEAM LINE FOR SCHOOLS AT ELSA USING CONSTELLATION

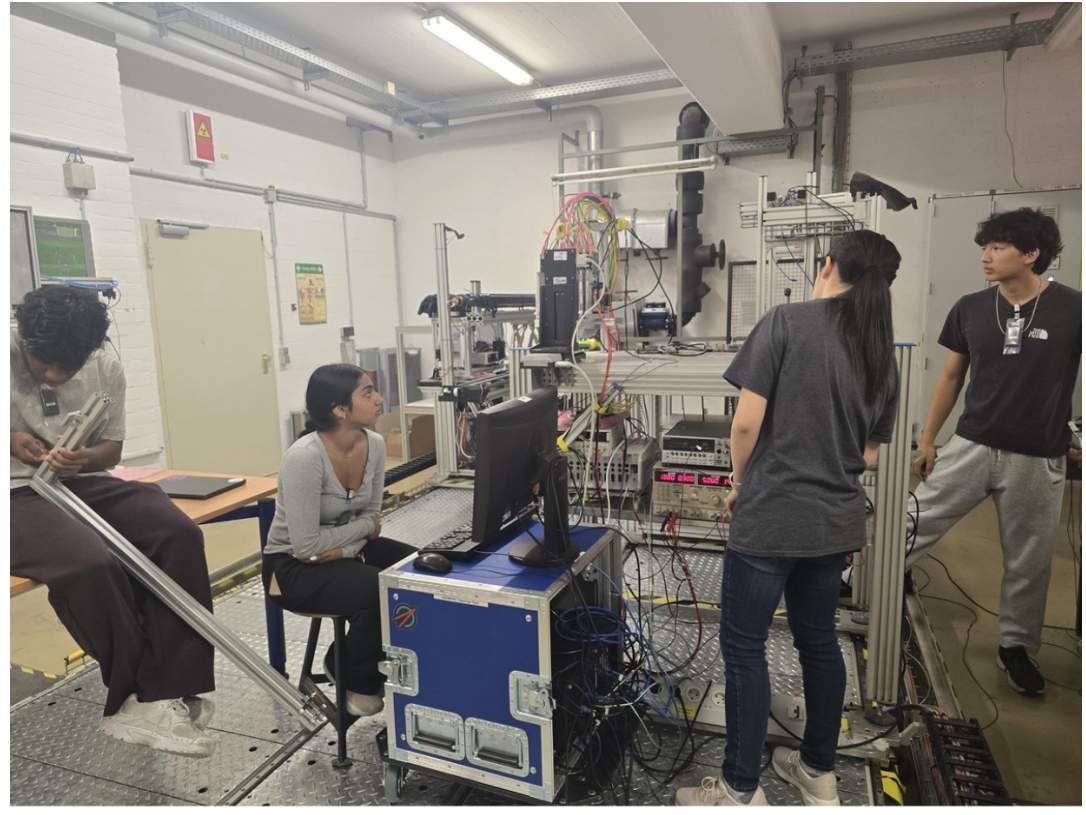
Rasmus Partzsch



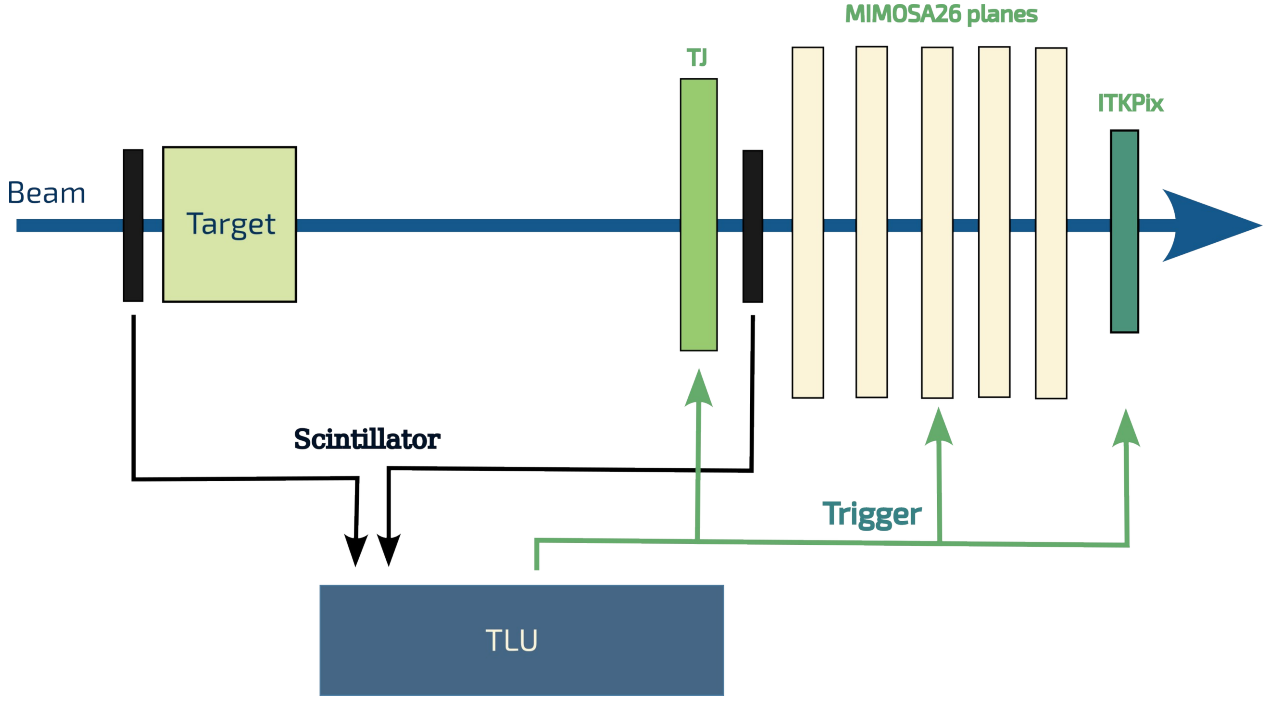
Rasmus Partzsch – rasmus.partzsch@uni-bonn.de

Beam Line for Schools

- Worldwide competition for students to carry out small test beam experiments
- 5 winning teams: 2 at CERN, 2 at DESY, 1 at Bonn ELSA
- 8 full days of test beam

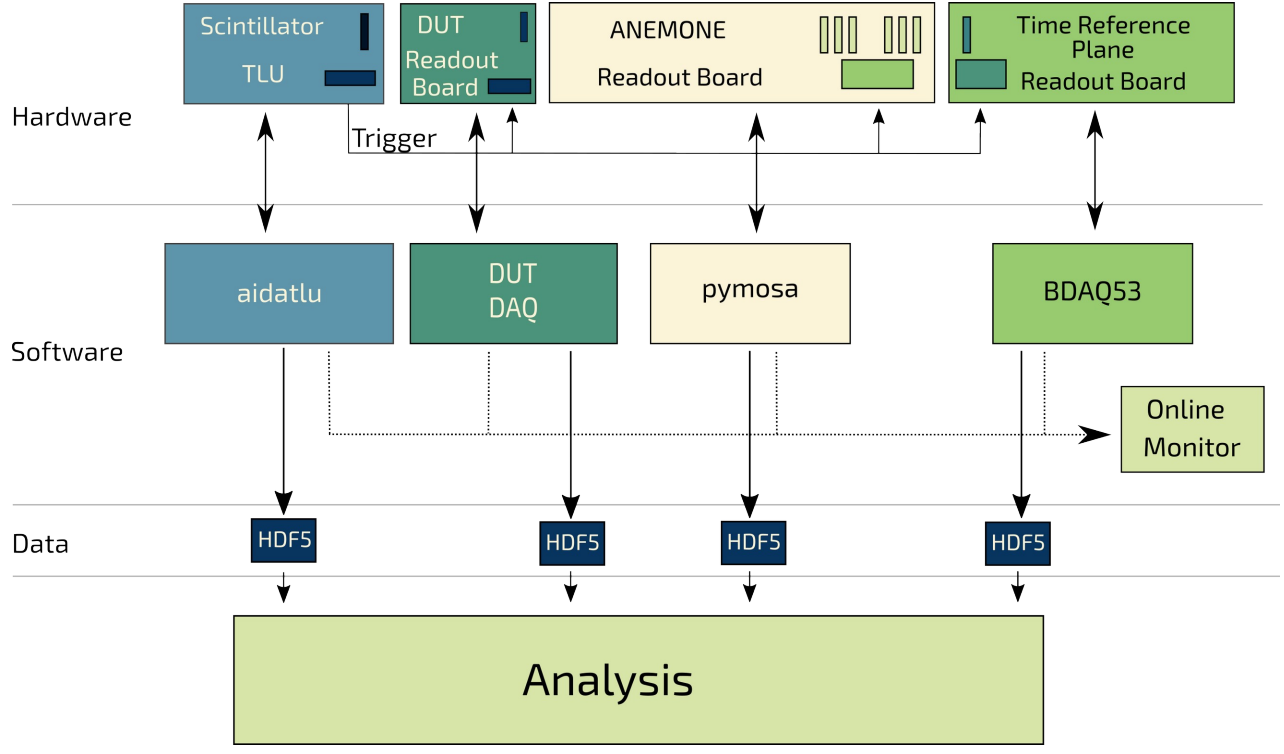


Test Beam Setup



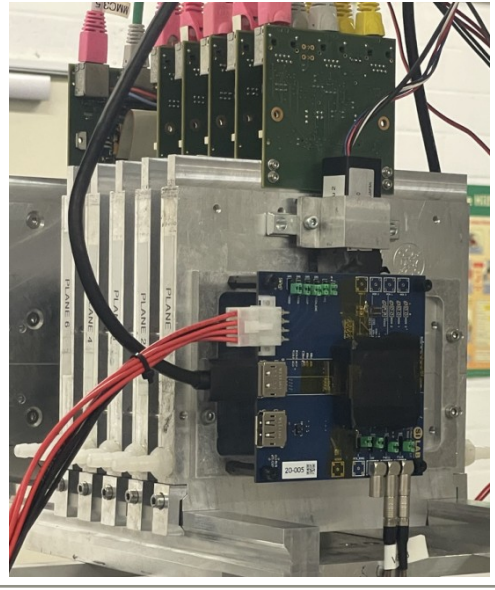
Readout of the Bonn Test Beam Telescope

- Modular software framework
- Each device is connected to a readout board and controlled using a Python based control software



Test Beam Setup Integration

- **Pymosa (EUDET-Telescope with MMC3)**
  - Start and stop of scans
  - Logging and Metrics (Trigger number)



- **Bdaq (ITkPix with BDAQ53)**
  - Start and stop external trigger scans
  - Basic configurations
  - Logging and Metrics (Trigger number)
- **TJ (TJ-Monopix2 with BDAQ53)**
  - Start and stop external trigger scans
  - Basic configurations
    - Tuning files
  - Logging and Metrics (Trigger number)
- **AidaTLU (AIDA 2020 TLU)**
  - Fully integrated

UNIVERSITÄT BONN

Configuration

- Configuration saved in TOML file format
- General configuration for all satellites
  - Max trigger number
  - Output data folder
- Satellite specific configuration
  - Activated pixels
  - Tuning files
  - Trigger configurations

```toml
[satellites]

max_triggers = 10000
scan_timeout = 0
output_folder = "/media/data/testbeam/"

[satellites.AidaTLU]

internal_trigger_rate = 0
dut_interfaces = ['eudet', 'eudet', 'aida', 'off']
trigger_threshold = [-0.1, -0.1, -0.1, -0.1, -0.1, -0.1]
trigger_inputs_logic = 'CH1 and CH2'
trigger_polarity = 'falling'
trigger_signal_stretch = [2, 2, 2, 2, 2, 2]
trigger_signal_delay = [0, 0, 0, 0, 0, 0]
enable_clock_lemo_output = false
pmt_power = [0.8, 0.8, 0.0, 0.0]

[satellites.Pymosa]

run_number = "None"
m26_configuration_file = "None"
m26_jtag_configuration = true
no_data_timeout = 30
scan_timeout = 0
enabled_m26_channels = "None"

[satellites.Bdaq]

start_column = 0
stop_column= 400
start_row = 0
stop_row = 384

[satellites.TJ]
start_column = 0
stop_column = 224
start_row = 0
stop_row = 512
tot_calib_file = "None"
chip_config_file = "/path/to/tuning_file.h5"
```

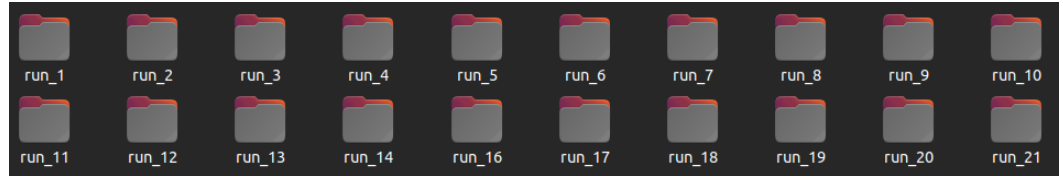
Configuration YAML

- BDAQ uses two configuration yaml, not possible to access yaml after configuration
- Testbench YAML
  - These include also a lot of unnecessary configurations (e.g. specific data formats)
  - But also output folder, chip tuning configuration…
- Bdaq53.yaml
  - Includes IP address of DAQ board
- In current implementation:
  - Testbench yaml is read in in initializing step, some parameters are overwritten by configuration TOML
  - YAML is read by DAQ during launching step

```
with open(TESTBENCH_DEFAULT_FILE, 'r') as f:
    conf = yaml.full_load(f)
    conf['general']['output_directory'] = configuration['output_folder']
    conf['modules']['module_0']['chip_0']['chip_config_file'] = configuration['chip_config_file']
```

Data Handling

- Data is saved to common output folder
  - Each Satellite creates its own subdirectory and saves all the data to this folder
  - Data is saved with a timestamp for identification
- After some runs, data of different detectors can be collected in run folders for analysis (this is done by hand)



Conclusion

- Problems:
  - Data interpretation, satellites needed to be landed after each run (This can easily be fixed on my end).
  - Sometimes a readout board lost connection when being in launched state for too long. Running state was fine (need to investigate).
  - Formatting bug with one of the trigger number metrics of pymosa, could not reproduce this bug for now…
- Independent operation of a complete test beam setup by high school students using Constellation (v0.5.1).