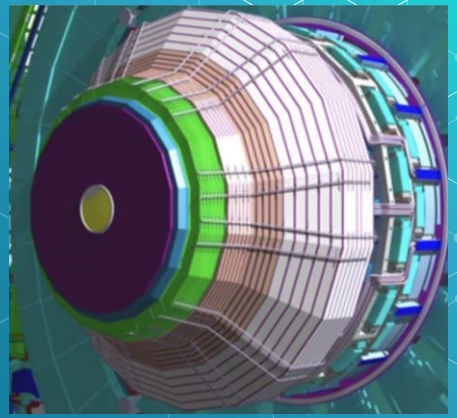
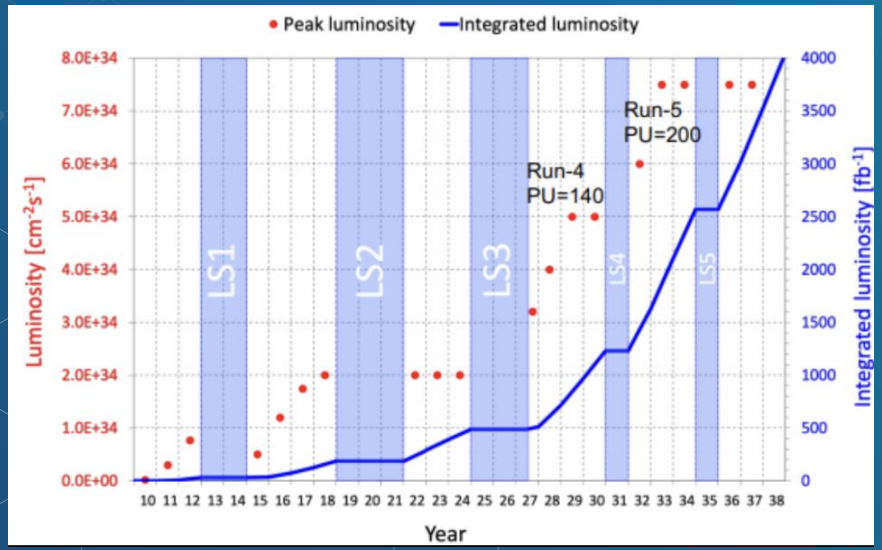
Electronics data acquisition systems for the future HGCal of CMS

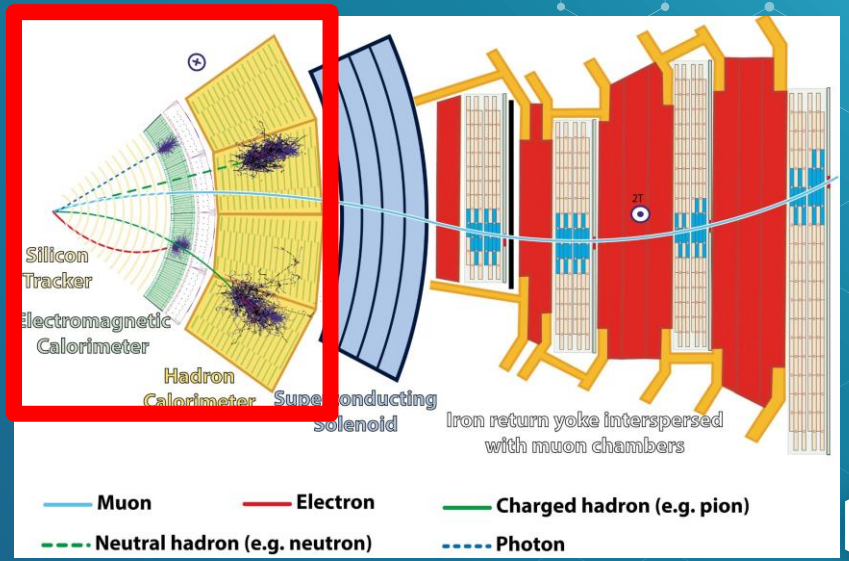
Marija Žukovski
Arsen Miković
Andreja Maksimović
Supervisor: Miloš Vojinović

Introduction

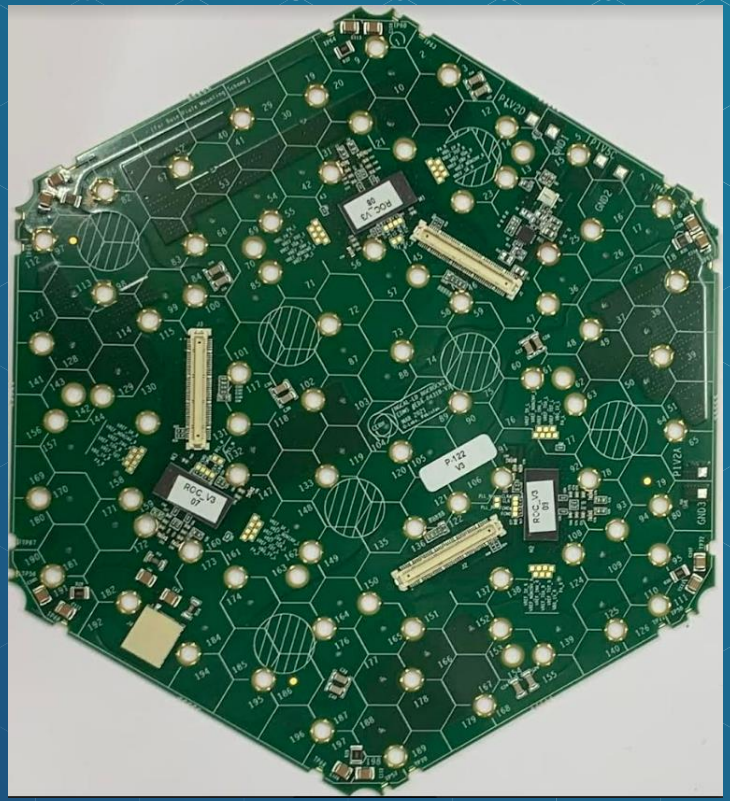
- What does a calorimeter do?
- Why do we need a new one?



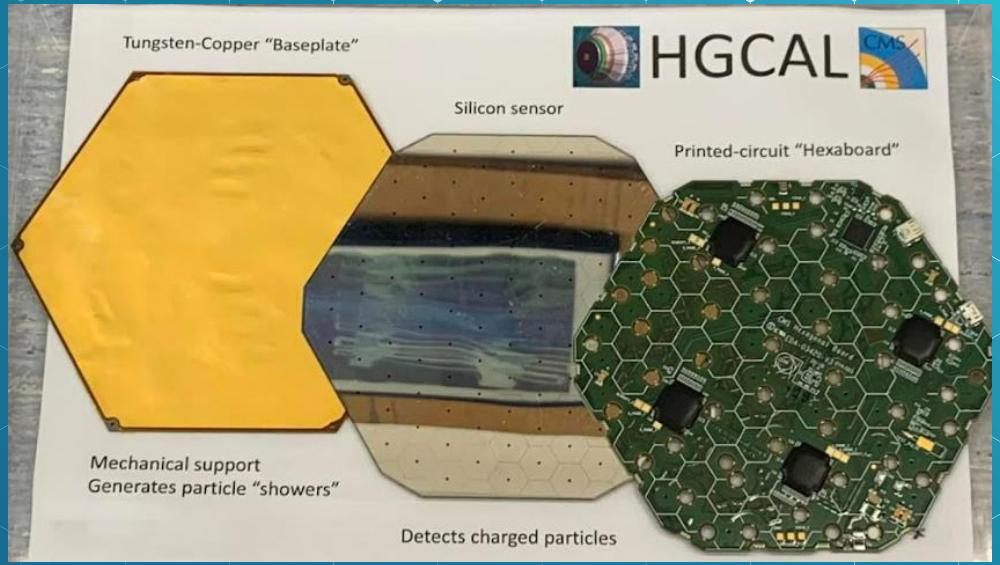




- The calorimeter will detect particles with custom made PCBs – hexa-boards
- Two alternating layers – silicon and absorber
- Particles interact with the absorber and create electromagnetic showers.
- The showers traversing through the hexa-boards leave traces of deposited energy and position

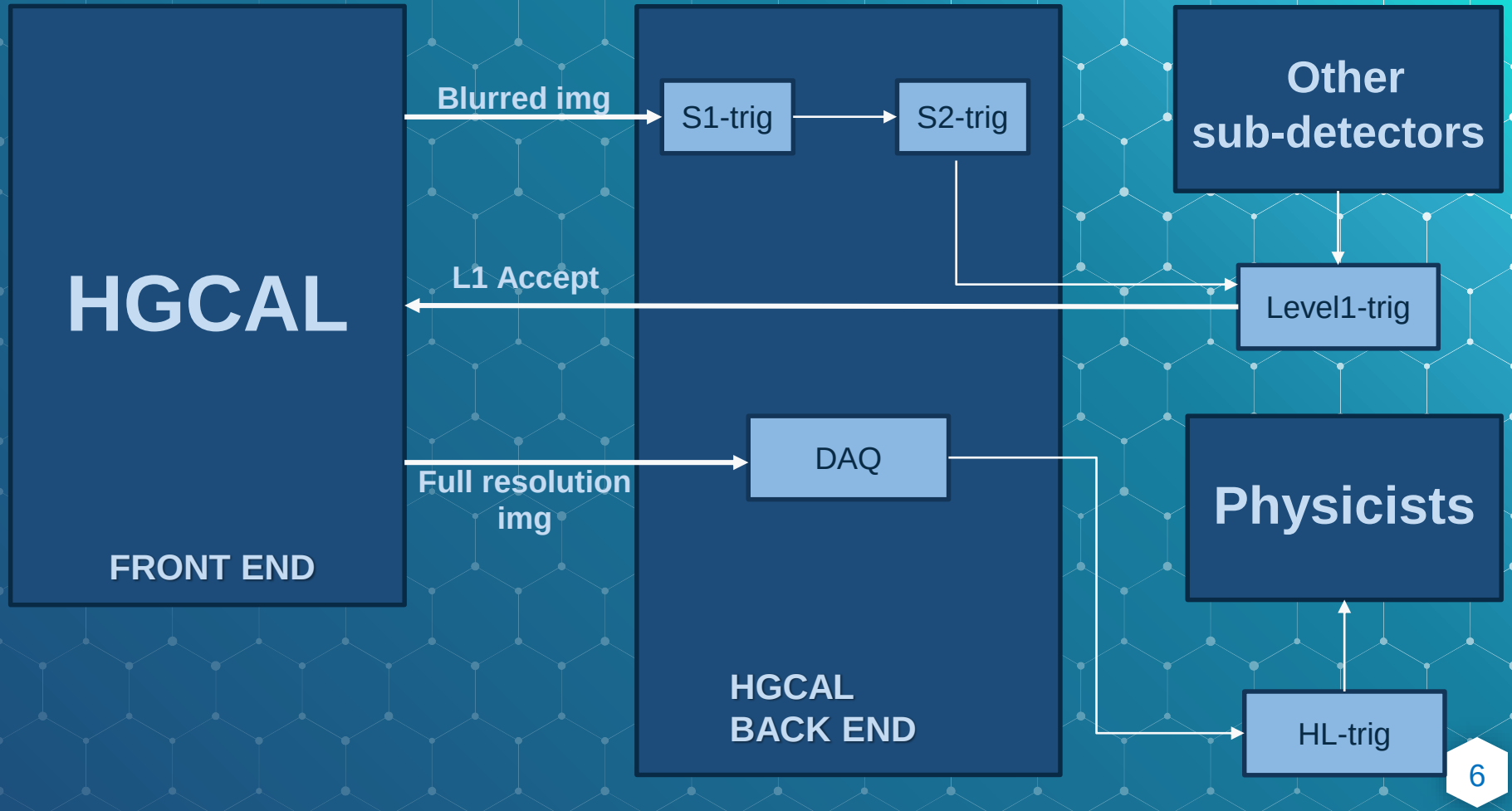


Hexa-board

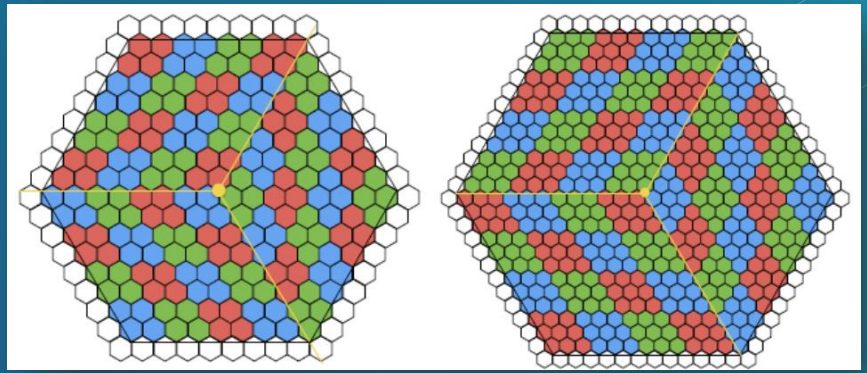


Silicon and absorber

Trigger and DAQ


HGCAL
FRONT END
Blurred img
S1-trig
S2-trig
Other sub-detectors
L1 Accept
Level1-trig
DAQ
Full resolution img
Physicists
HGCAL BACK END
HL-trig

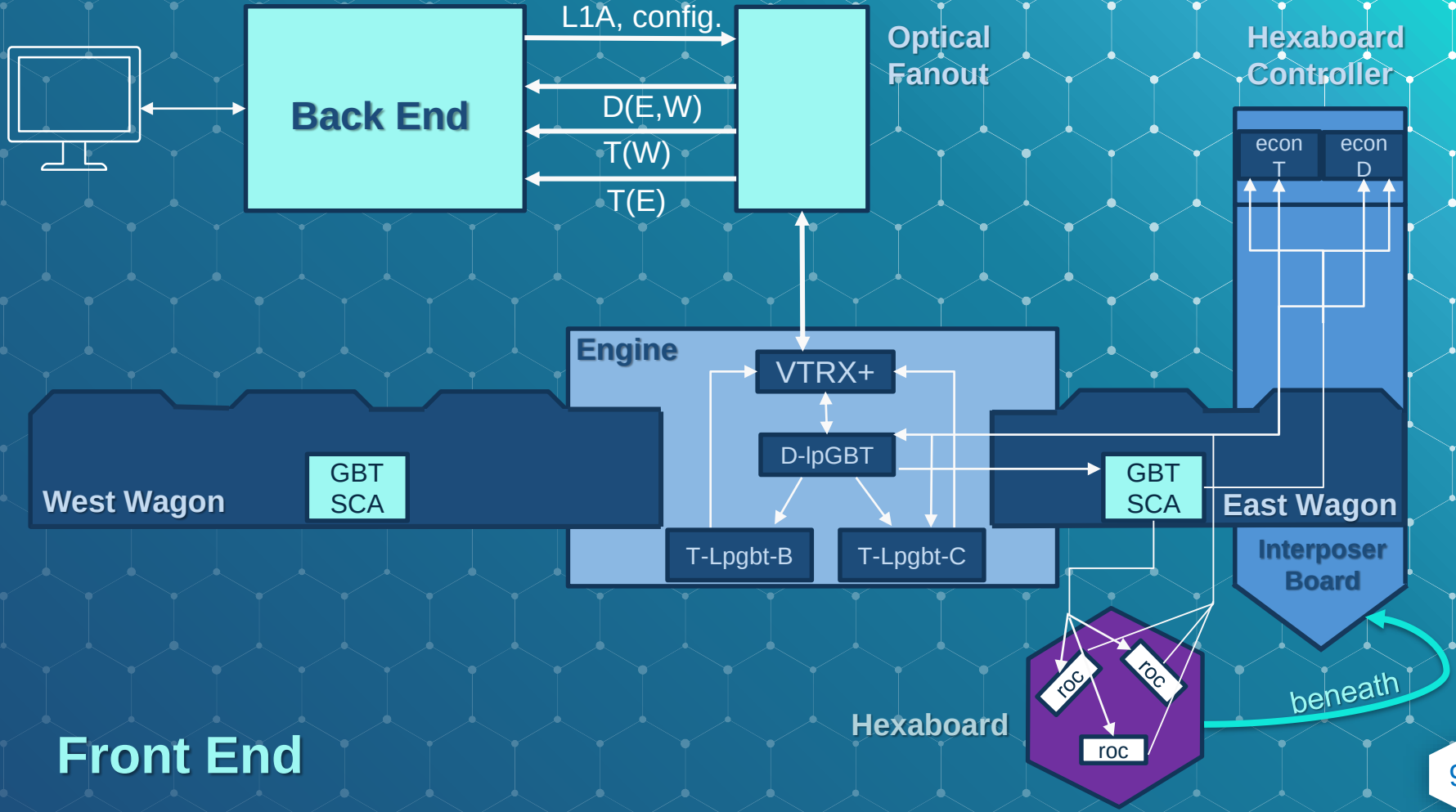
- Blurred image – set of 4 or 9 hexa-cells per pixel



- After L1 trigger – 2% pass the selection
- After High Level trigger – 0,02%
- The images are sent to physicists and super-computers for analysis

The Train Test System

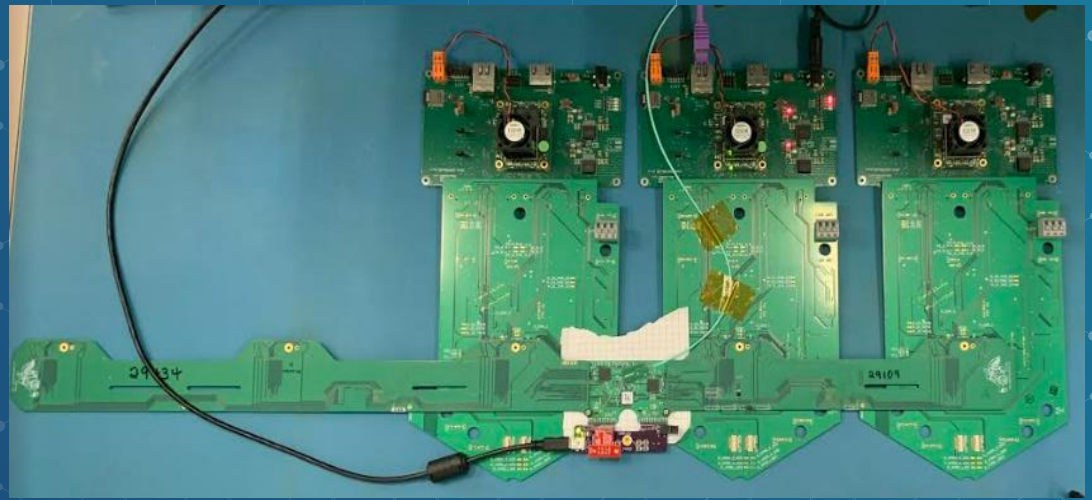
A prototype of HGCAL's FE-BE system

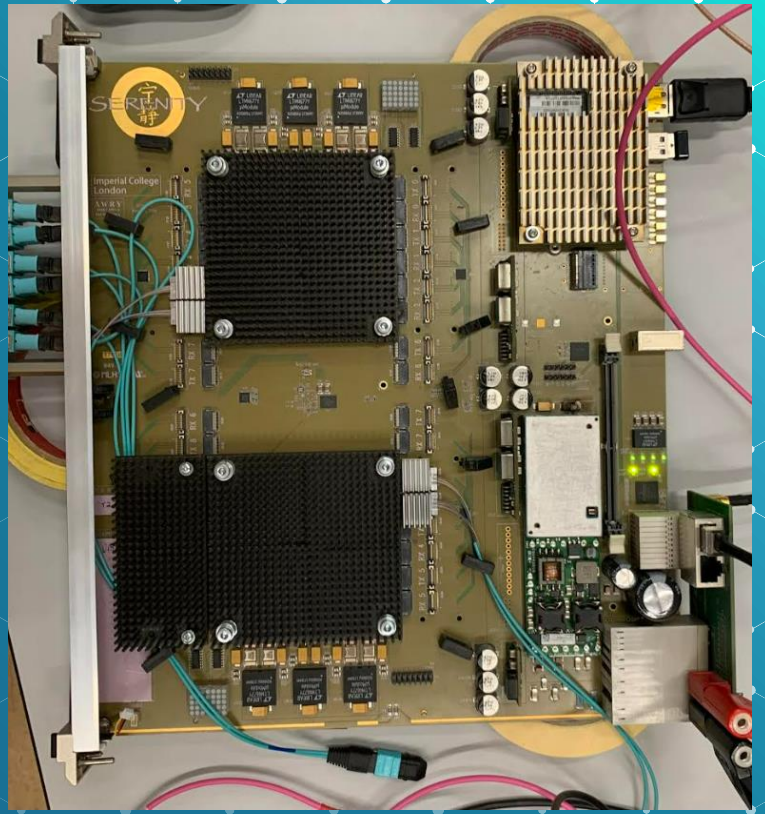

Back End
L1A, config.
D(E,W)
T(W)
T(E)
Optical Fanout
Hexaboard Controller
econ T
econ D
Engine
VTRX+
D-lpGBT
T-Lpgbt-B
T-Lpgbt-C
West Wagon
GBT SCA
GBT SCA
East Wagon
Interposer Board
roc
roc
roc
Hexaboard
beneath

Front End

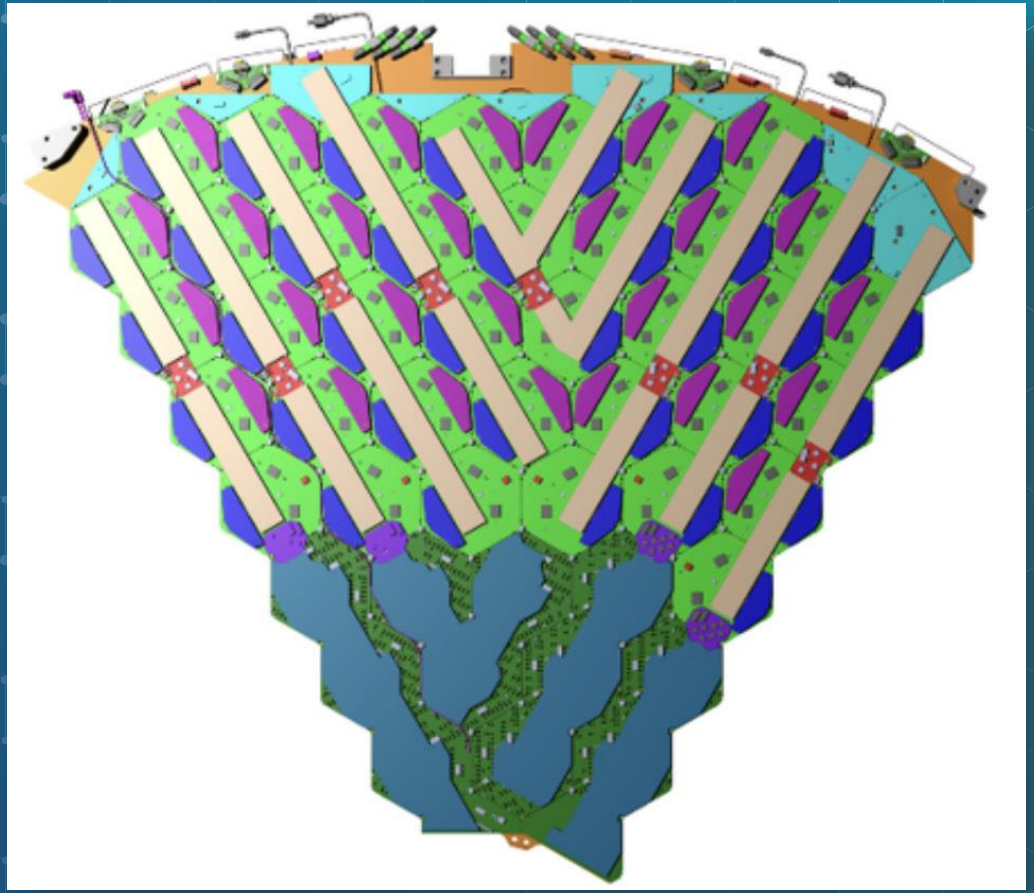
Optical Fanout

Train



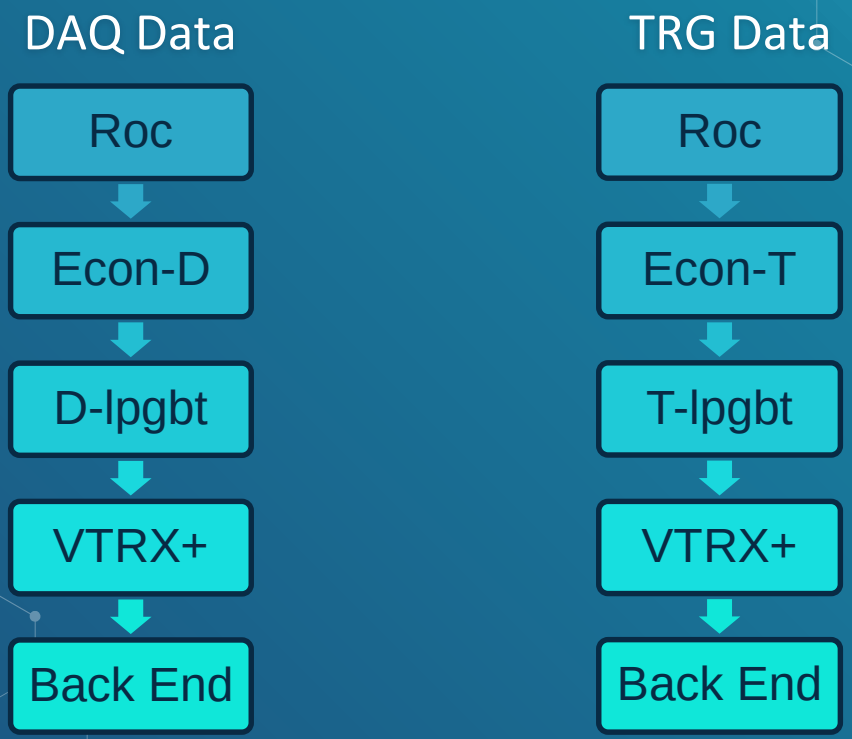


Serenity (BackEnd)



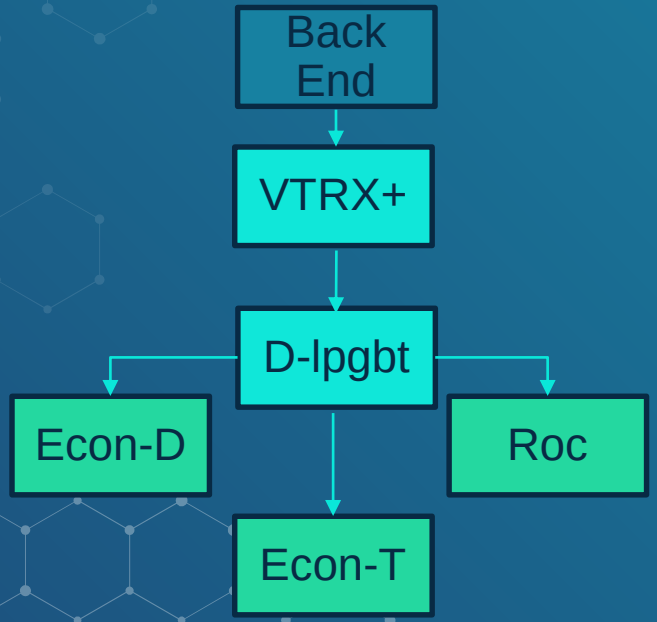
Train Arrangement

Front End → Back End

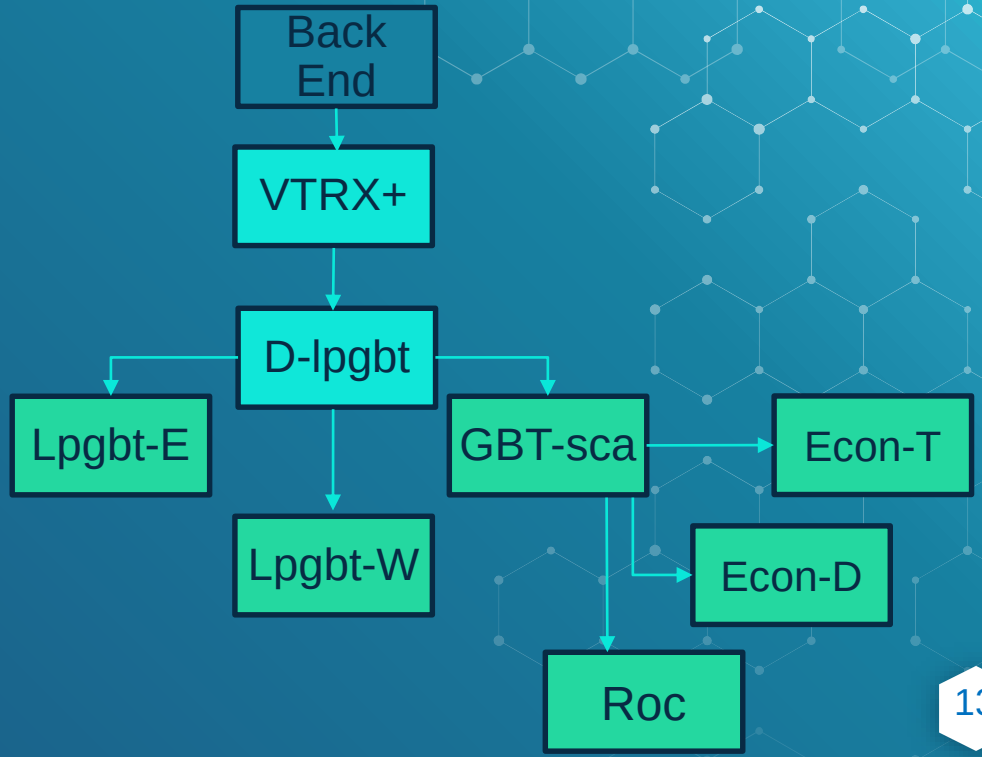


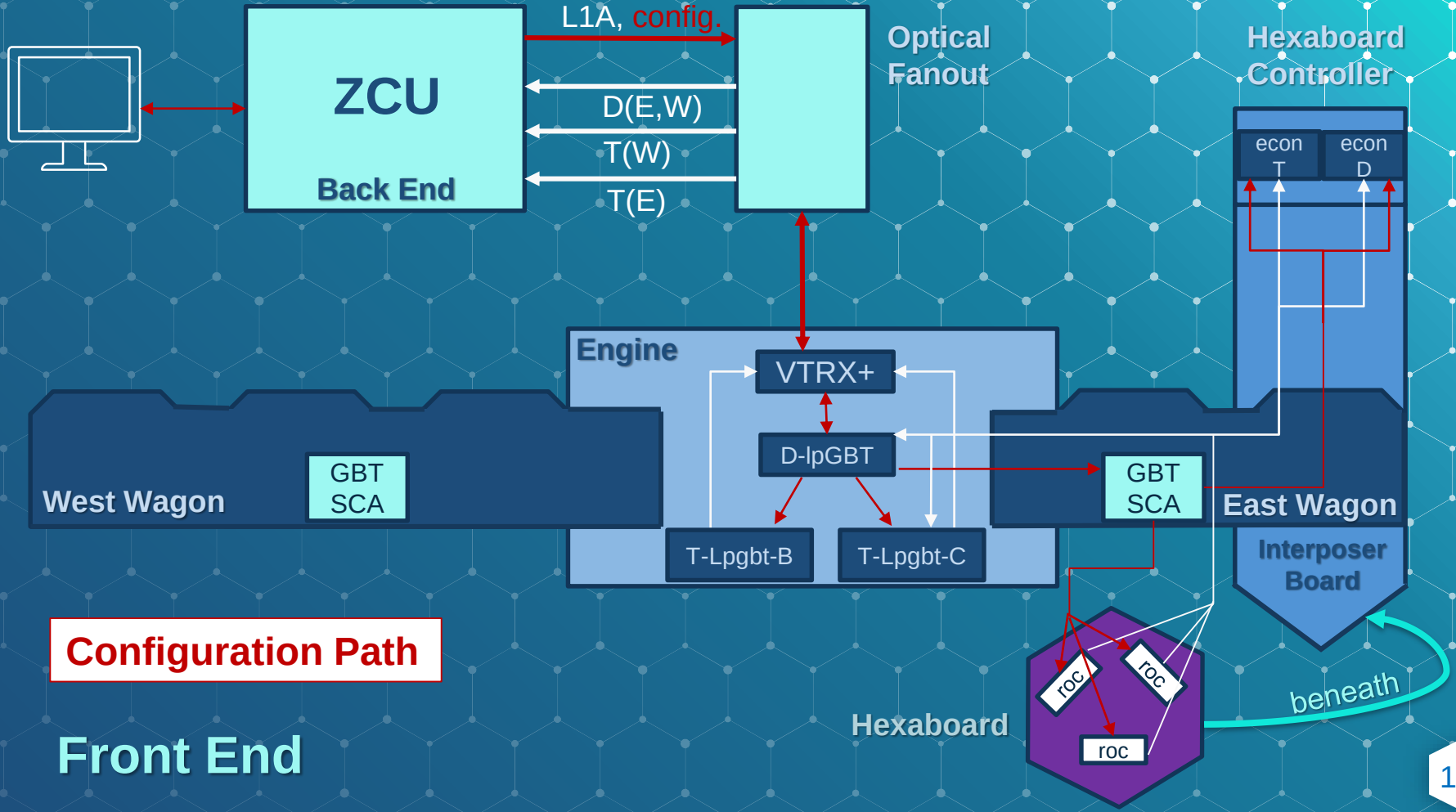
Back End → Front End

L1A



Configuration




ZCU
Back End
L1A, config.
D(E,W)
T(W)
T(E)
Optical Fanout
Hexaboard Controller
econ T
econ D
Engine
VTRX+
D-lpGBT
T-Lpgbt-B
T-Lpgbt-C
West Wagon
GBT SCA
GBT SCA
East Wagon
Interposer Board
roc
roc
roc
Hexaboard
beneath
Configuration Path
Front End

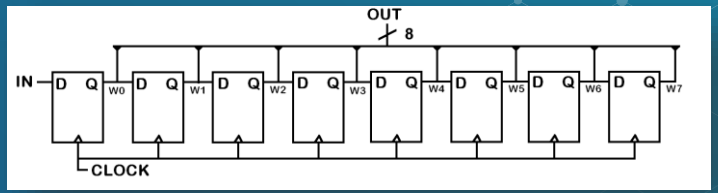
A Deeper look into HGCal Front End Chips

What are registers?

- Binary and hexadecimal representations of numbers 0-15.

Decimal (Base 10)	Binary (Base 2)	Hexadecimal (Base 16)
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

- Flip-flops store 1 bit of information
- A Register is made of N flip-flops, storing an N-bit word of info (e.g. N=4)
- Every register in an lpGBT has 8 bits (1 byte)



Hands On Task

Configuring the LpGBT to send constant patterns and capturing them in the backend

Steps Breakdown

1. Read the lpGBT manual to understand Constant Pattern Generation possibilities
2. Chose one of the above
3. Figure out what binary values need to be written in what registers to achieve the wanted behaviour of the lpGBT
4. Transforming the required binary values to hexadecimal
5. Writing the appropriate values in the appropriate registers using a python script (setup_lpgbt_Serbia.py)

Steps 1,2

- Find the right path using the manual

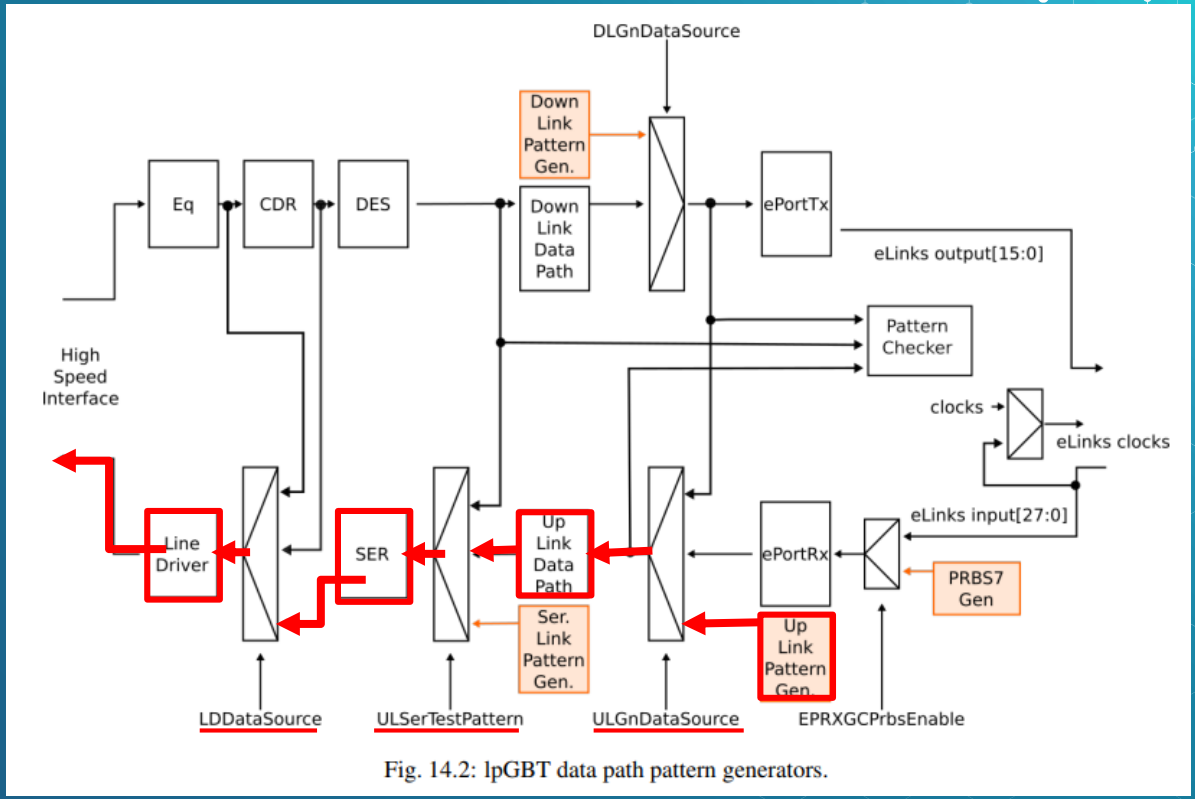


Fig. 14.2: lpGBT data path pattern generators.

Steps 3,4 (e.g.)

- Register ULDataSource (0x119)
- d00100100=0x24

[0x119] ULDataSource1

Uplink data path test patterns.

- **Bit 7:6 - LDDataSource[1:0]** - Data source for the line driver.

LDDataSource[1:0]	Description
2'd0	Data from serializer (normal mode of operation)
2'd1	Data resampled by CDR loopback
2'd2	Equalizer output data loopback
2'd3	reserved

- **Bit 2:0 - ULG0DataSource[2:0]** - Data source for uplink data group 0

ULG0DataSource[2:0]	Name	Description
3'd0	EPORTRX_DATA	Normal mode of operation, data from ePortRx
3'd1	PRBS7	PRBS7 test pattern
3'd2	BIN_CNTR_UP	Binary counter counting up
3'd3	BIN_CNTR_DOWN	Binary counter counting down
3'd4	CONST_PATTERN	Constant pattern (`DPDataPattern[31:0]`)
3'd5	CONST_PATTERN_INV	Constant pattern inverted (`~DPDataPattern[31:0]`)
3'd6	DLDATA_LOOPBACK	Loop back, downlink frame data
3'd7	Reserved	Reserved

- **Bit 5:3 - ULG1DataSource[2:0]** - Data source for uplink data group 1

ULG1DataSource[2:0]	Name	Description
3'd0	EPORTRX_DATA	Normal mode of operation, data from ePortRx
3'd1	PRBS7	PRBS7 test pattern
3'd2	BIN_CNTR_UP	Binary counter counting up
3'd3	BIN_CNTR_DOWN	Binary counter counting down
3'd4	CONST_PATTERN	Constant pattern (`DPDataPattern[31:0]`)
3'd5	CONST_PATTERN_INV	Constant pattern inverted (`~DPDataPattern[31:0]`)
3'd6	DLDATA_LOOPBACK	Loop back, downlink frame data
3'd7	Reserved	Reserved

Step 5

```
self.iic.write_lpgbt_trig(Upgbt_id, 0x118, 0x00)
self.iic.write_lpgbt_trig(lpgbt_id, 0x119, 0x24)
self.iic.write_lpgbt_trig(lpgbt_id, 0x11A, 0x24)
self.iic.write_lpgbt_trig(lpgbt_id, 0x11B, 0x24)
self.iic.write_lpgbt_trig(Upgbt_id, 0x11C, 0x04)

self.iic.write_lpgbt_trig(Lpgbt_id, 0x11E, 0xDE)
self.iic.write_lpgbt_trig(lpgbt_id, 0x11F, 0xAD)
self.iic.write_lpgbt_trig(lpgbt_id, 0x120, 0xBE)
self.iic.write_lpgbt_trig(lpgbt_id, 0x121, 0xEF)
```

Results

```
0xdeadbeef 0xdeadbeef
0xdeadbeef 0xdeadbeef
0xdeadbeef 0xdeadbeef
0xdeadbeef 0xdeadbeef
0xdeadbeef 0xdeadbeef
0xdeadbeef 0xdeadbeef
0xdeadbeef 0xdeadbeef
0xdeadbeef 0xdeadbeef
0xdeadbeef 0xdeadbeef
0xdeadbeef 0xdeadbeef
0xdeadbeef 0xdeadbeef
0xdeadbeef 0xdeadbeef
0xdeadbeef 0xdeadbeef
0xdeadbeef 0xdeadbeef
0xdeadbeef 0xdeadbeef
0xdeadbeef 0xdeadbeef
0xdeadbeef 0xdeadbeef
0xdeadbeef 0xdeadbeef
0xdeadbeef 0xdeadbeef
0xdeadbeef 0xdeadbeef
0xdeadbeef 0xdeadbeef
0xdeadbeef 0xdeadbeef
0xdeadbeef 0xdeadbeef
0xdeadbeef 0xdeadbeef
0xdeadbeef 0xdeadbeef
0xdeadbeef 0xdeadbeef
```

- Lpgbt-C will return a pattern of Cs
- Lpgbt-B will return a pattern of Bs

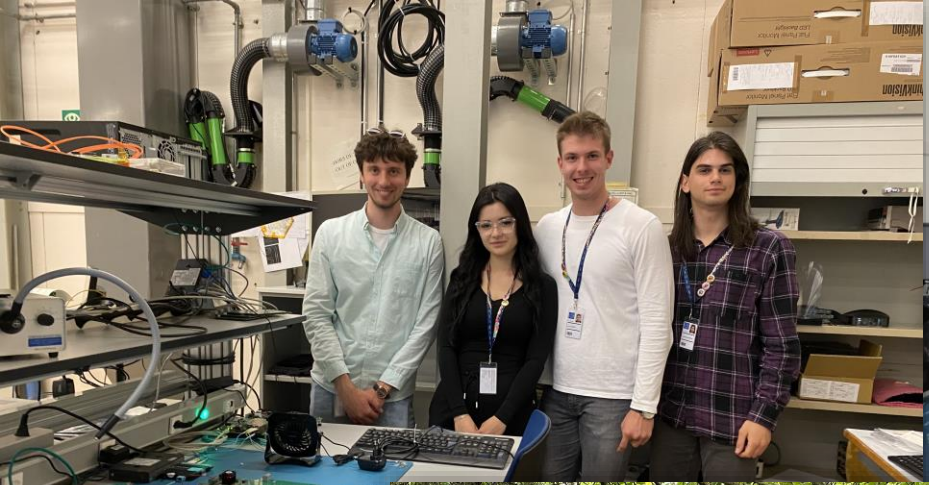
```
self.iic.write_lpgbt_trig(Upgbt_id, 0x118, 0x00)
self.iic.write_lpgbt_trig(lpgbt_id, 0x119, 0x24)
self.iic.write_lpgbt_trig(lpgbt_id, 0x11A, 0x24)
self.iic.write_lpgbt_trig(lpgbt_id, 0x11B, 0x24)
self.iic.write_lpgbt_trig(Upgbt_id, 0x11C, 0x04)

if lpgbt_id=="C":

    self.iic.write_lpgbt_trig(Lpgbt_id, 0x11E, 0xCC)
    self.iic.write_lpgbt_trig(lpgbt_id, 0x11F, 0xCC)
    self.iic.write_lpgbt_trig(lpgbt_id, 0x120, 0xCC)
    self.iic.write_lpgbt_trig(lpgbt_id, 0x121, 0xCC)

else:

    self.iic.write_lpgbt_trig(Lpgbt_id, 0x11E, 0xBB)
    self.iic.write_lpgbt_trig(lpgbt_id, 0x11F, 0xBB)
    self.iic.write_lpgbt_trig(lpgbt_id, 0x120, 0xBB)
    self.iic.write_lpgbt_trig(lpgbt_id, 0x121, 0xBB)
```

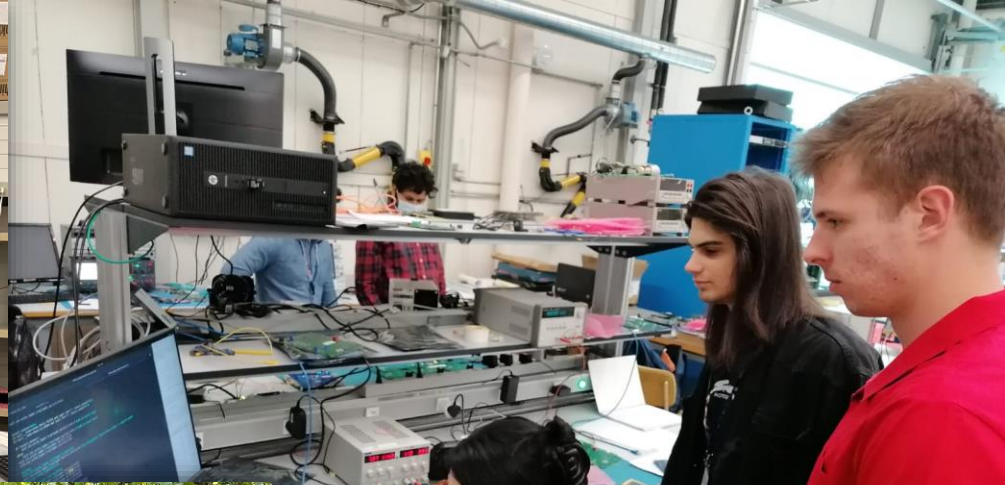
Results

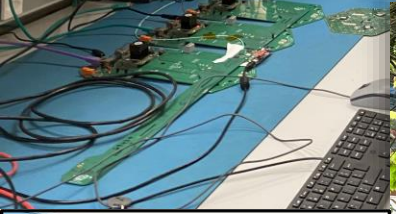
lpGbt-C

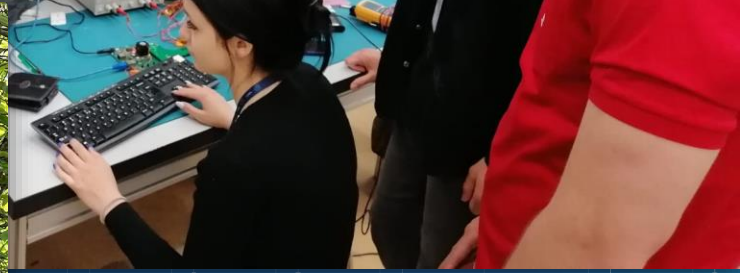
lpGbt-B

```
0xcccccccc 0xbbbbbbbb
0xcccccccc 0xbbbbbbbb
0xcccccccc 0xbbbbbbbb
0xcccccccc 0xbbbbbbbb
0xcccccccc 0xbbbbbbbb
0xcccccccc 0xbbbbbbbb
0xcccccccc 0xbbbbbbbb
0xcccccccc 0xbbbbbbbb
0xcccccccc 0xbbbbbbbb
0xcccccccc 0xbbbbbbbb
0xcccccccc 0xbbbbbbbb
0xcccccccc 0xbbbbbbbb
0xcccccccc 0xbbbbbbbb
0xcccccccc 0xbbbbbbbb
0xcccccccc 0xbbbbbbbb
0xcccccccc 0xbbbbbbbb
0xcccccccc 0xbbbbbbbb
0xcccccccc 0xbbbbbbbb
0xcccccccc 0xbbbbbbbb
0xcccccccc 0xbbbbbbbb
0xcccccccc 0xbbbbbbbb
0xcccccccc 0xbbbbbbbb
0xcccccccc 0xbbbbbbbb
0xcccccccc 0xbbbbbbbb
0xcccccccc 0xbbbbbbbb
```









CMS

Compact Muon Solenoid

Marija Žukovski
Arsen Miković
Andreja Maksimović
Supervisor: Miloš Vojinović