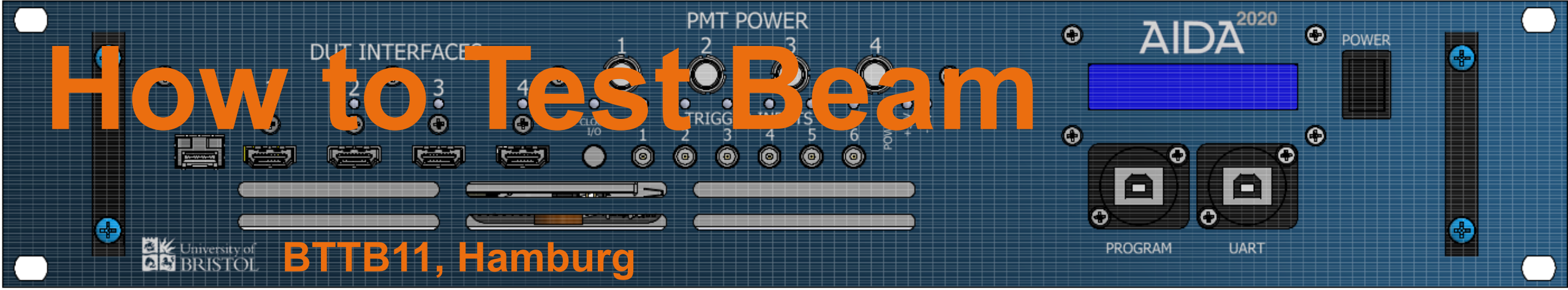
How to Test Beam



BTTB11, Hamburg

Adrian Herkert & Lennart Huth

HELMHOLTZ

DESY.

What did you sign up for?

- Short Introduction
- Getting started with the software
- Getting started with the hardware
- How to set up a trigger
- Communicating with the TLU
- Evaluating the different trigger modes
- Any issues we might discover
- Real particles and beam :)

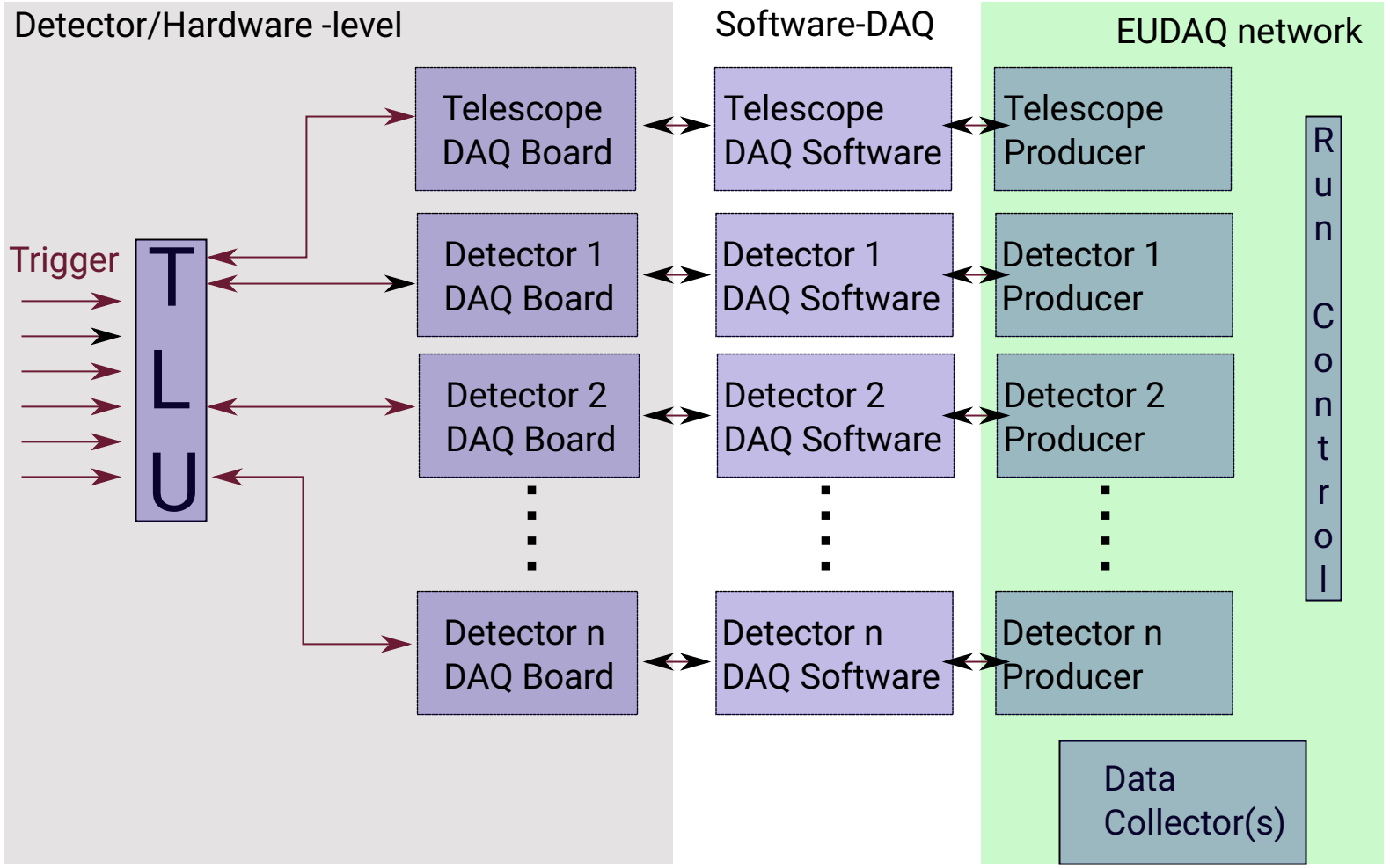
This tutorial is meant to be interactive - it requires input from everyone: Whenever you have a question/ comment/… just interrupt me.

Introduction

A typical test beam setup

Multiple layers of hard- and software

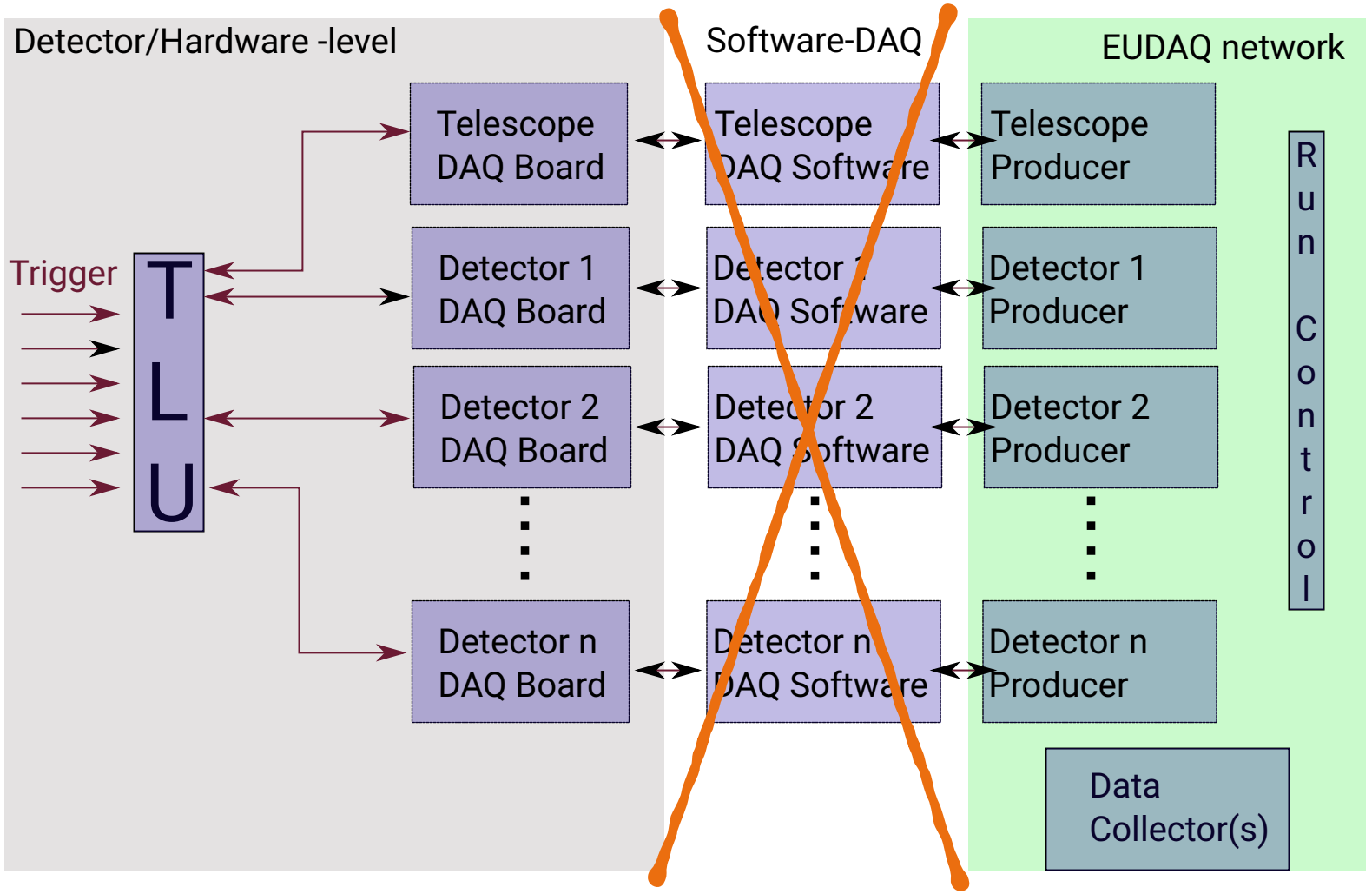
- Reference Telescope for tracking (Provided by facility)
- Your own detectors(s)
- Trigger Logic unit to synchronise detectors on HW level
- EUDAQ(2) as software framework to steer the readout of all connected detectors
- Assumption: You have a hardware interface to the TLU available
-



A typical test beam setup

Multiple layers of hard- and software

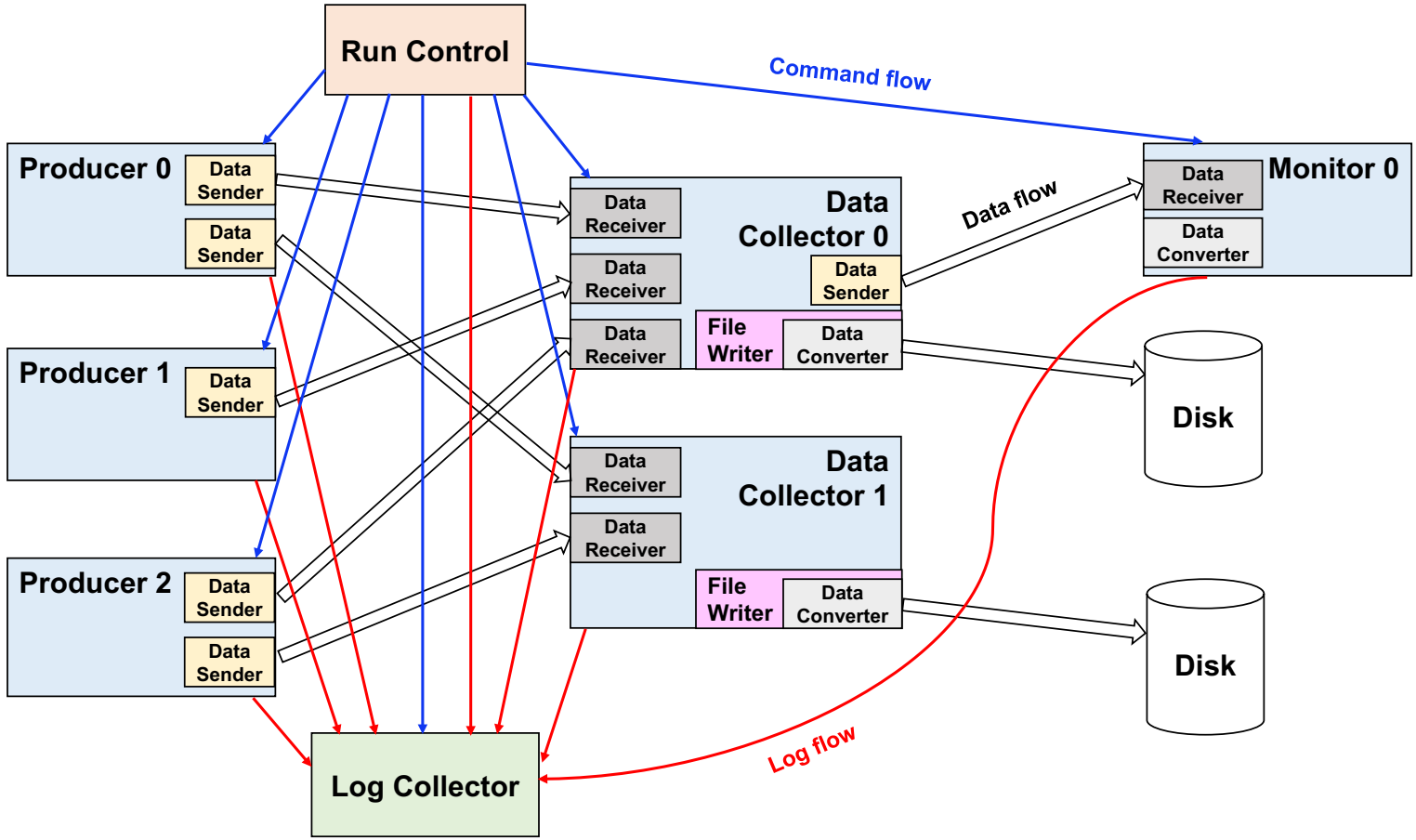
- Reference Telescope for tracking (Provided by facility)
- Your own detectors(s)
- Trigger Logic unit to synchronise detectors on HW level
- EUDAQ(2) as software framework to steer the readout of all connected detectors
- Assumption: You have a hardware interface to the TLU available
- Individual DAQ softwares - not covered today



EUDAQ

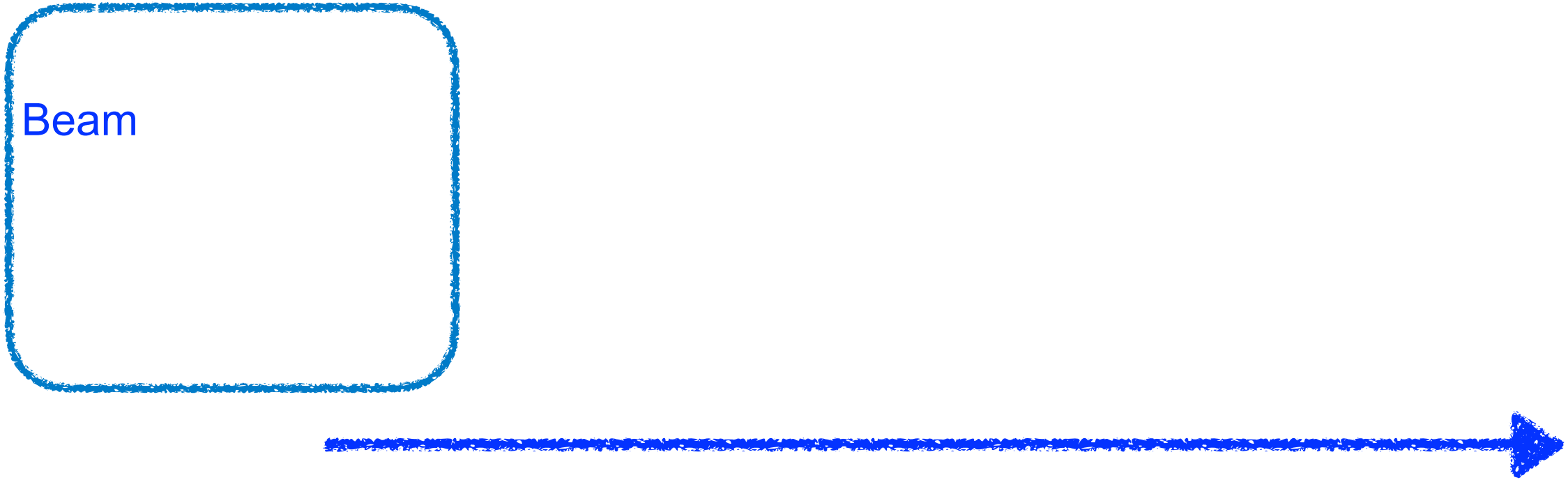
A network based multi platform DAQ system

- EUDAQ is developed as common DAQ software for test beams
- Currently *EUDAQ2* is state of the art
- One *RunControl* instance
- Multiple *Producer* that stream data from DAQ system
- Multiple *DataCollector* that receive and store data
- Central *LogCollector* to gather status information
- *Monitors* to keep an eye on data while recording
- Plain text files to configure and initialise EUDAQ
- **Users need to implement their Producers and (if they want to use the EUDAQ monitors a data converter)**



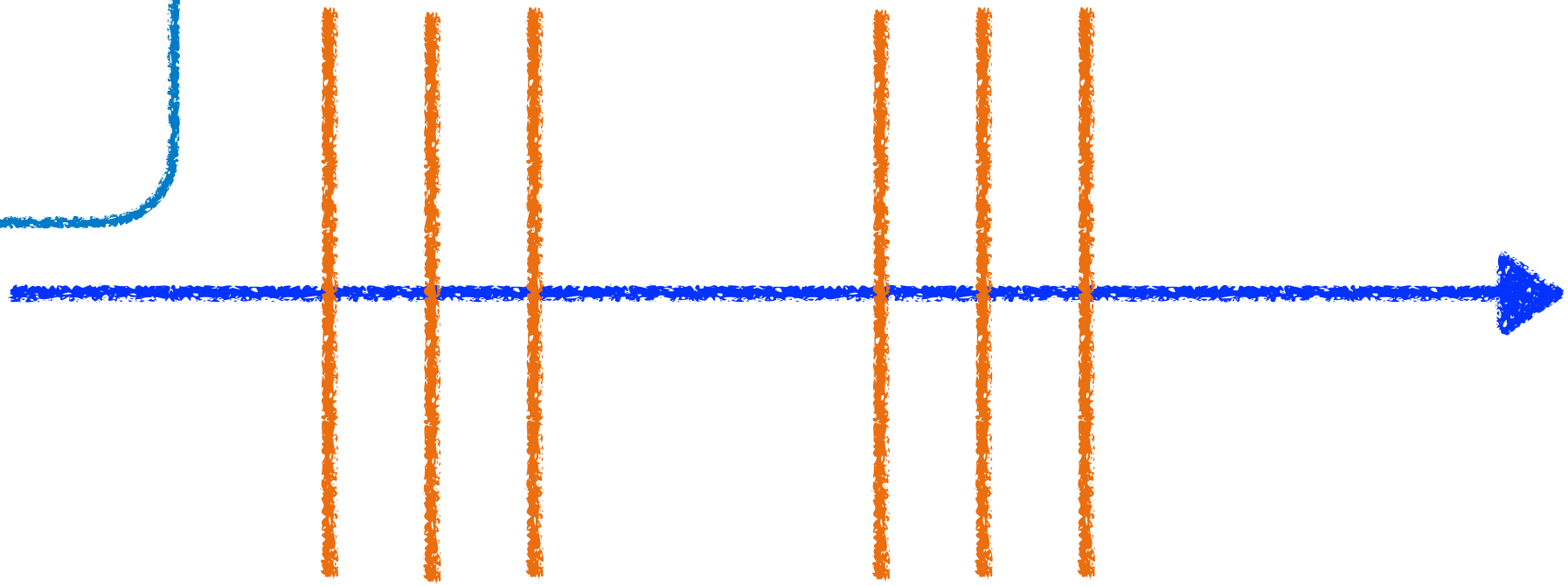
The setup at the DESY II beam lines

Typical Test Beam Setup



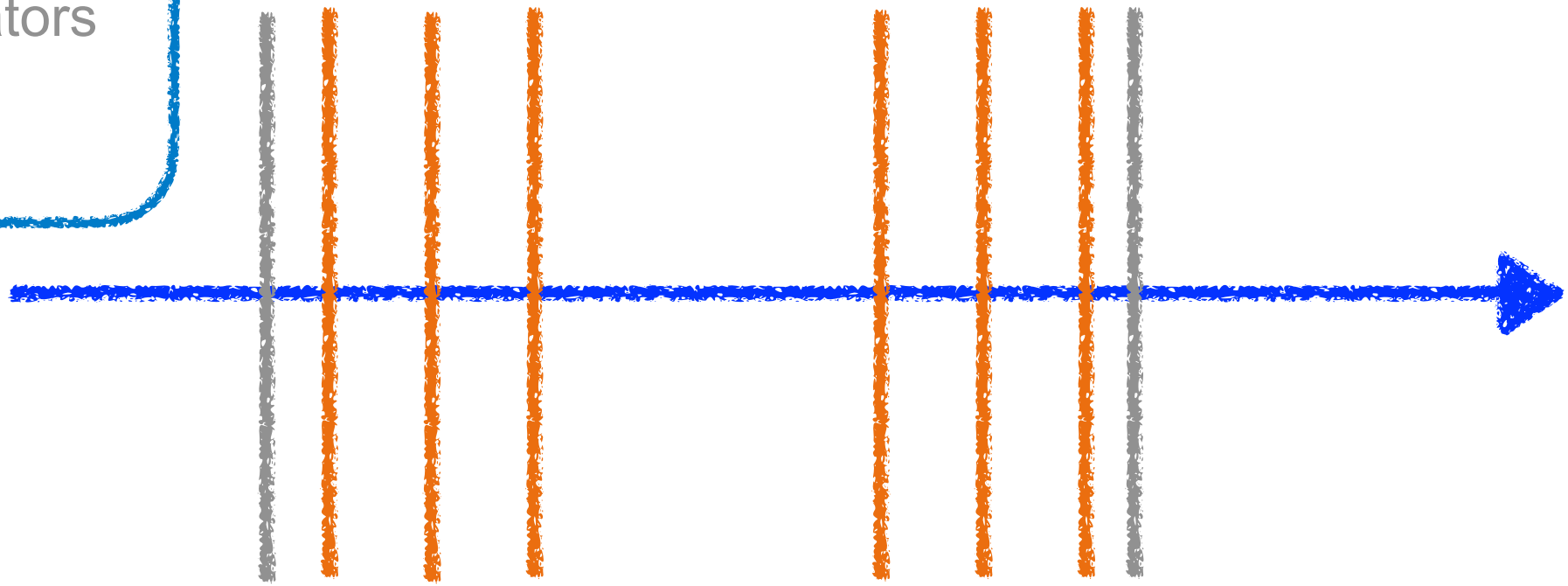
Typical Test Beam Setup

Beam
6 layer telescope



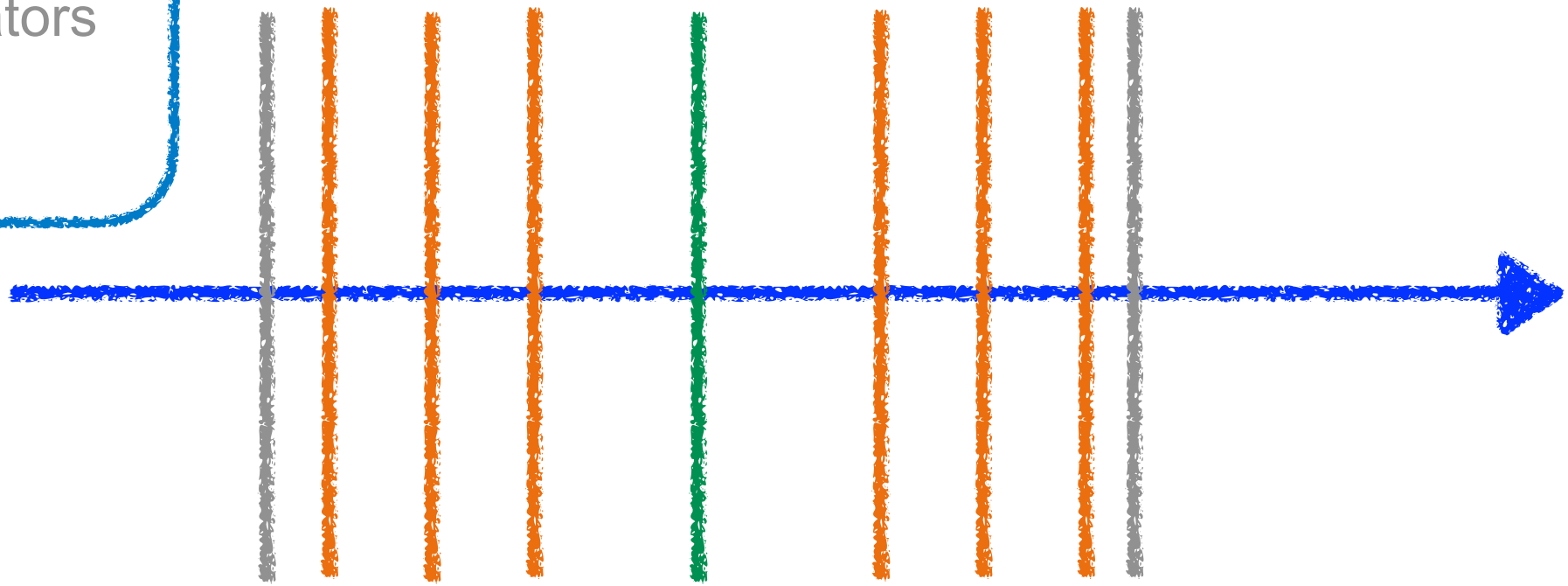
Typical Test Beam Setup

Beam
6 layer telescope
Trigger scintillators



Typical Test Beam Setup

Beam
6 layer telescope
Trigger scintillators
DUT



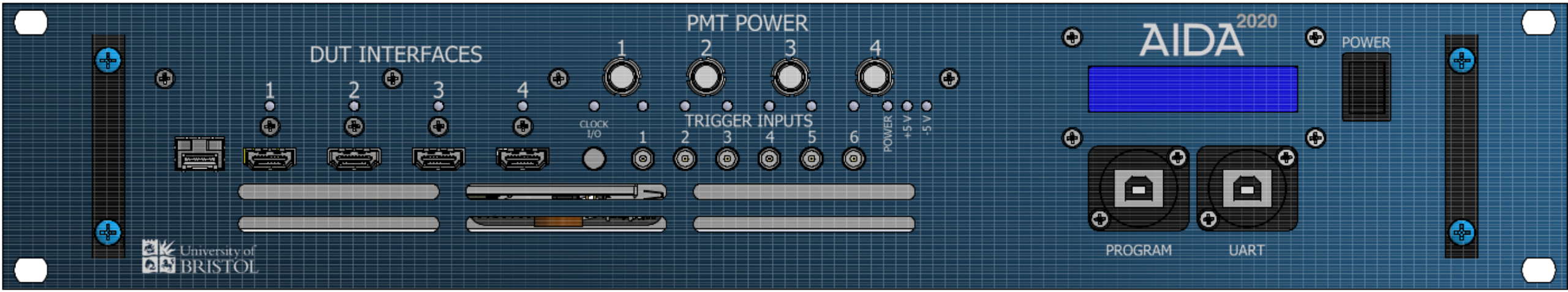
The AIDA-TLU

The AIDA-2020 Trigger Logic Unit

Flexible and Versatile Tool for DAQ System synchronisation

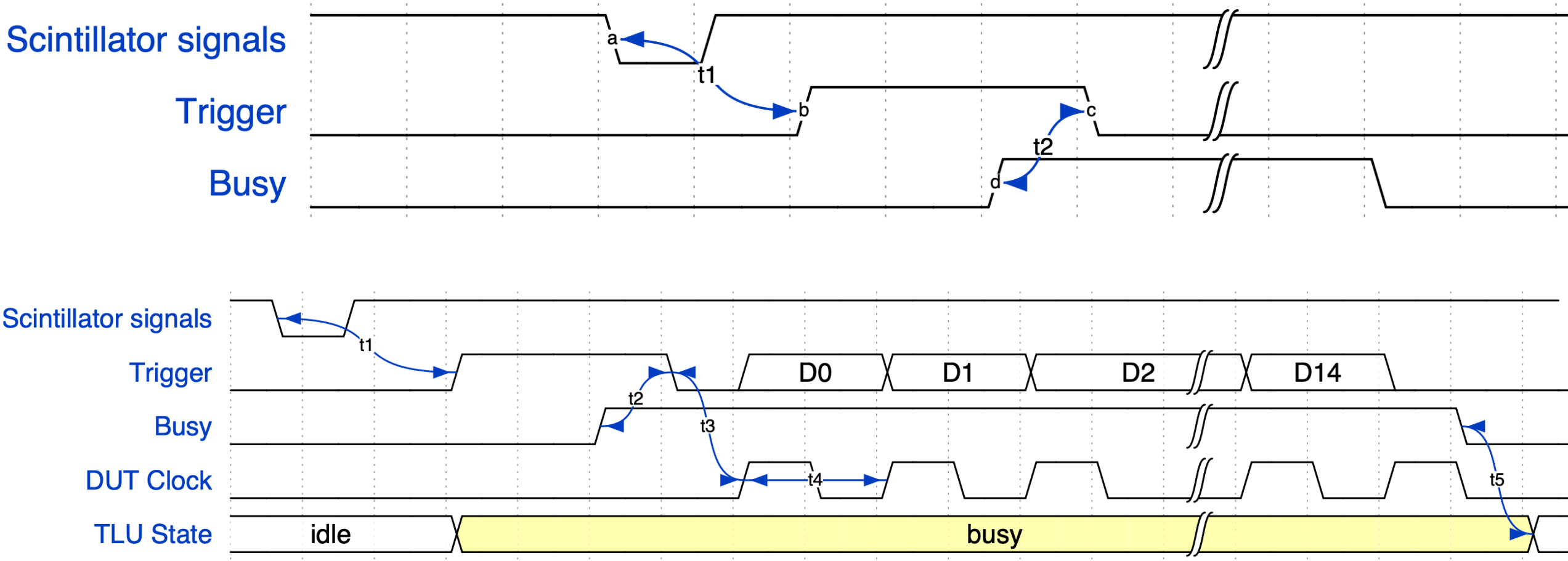
- 4 HDMI based DUT interfaces
- 6 trigger inputs (+/-1.2V range, DAC controlled)
- 4 PMT power outputs via 4pin LEMO
- Optical port for low jitter clock distribution
- Network connection and power on backside

- Development within the AIDA 2020 framework
- Designed at Bristol by D. Cussans
- 19 inch rack format
- Mass production at DESY (30 pieces, distributed all over the globe



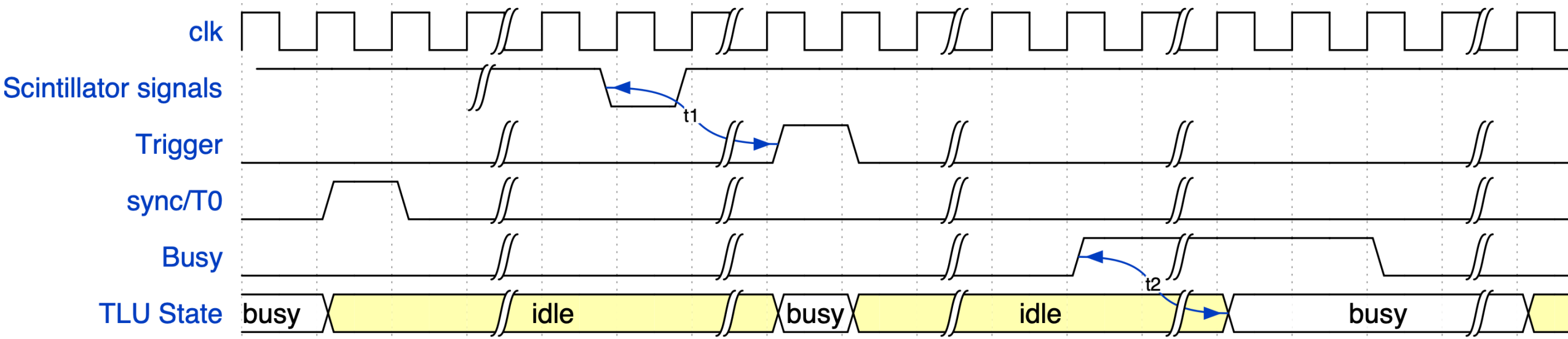
TLU Interfaces I

The EUDET Mode



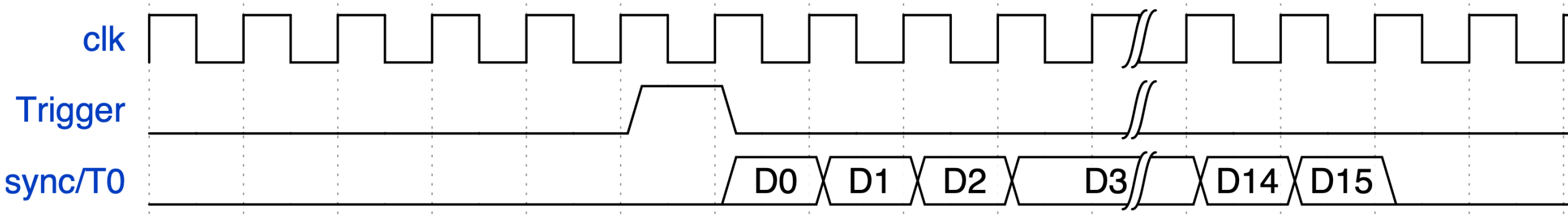
TLU Interfaces II

The AIDA Mode



TLU Interfaces III

The AIDA Mode with trigger ID



Parameters to configure the TLU - Initialisation

```
[Producer.aida_tlu]
# you can use this to track your changes, e.g. using the date
initid = 20180925
TLUmod= "1e"

# Path on the PC with TLU Producer and relative path is starting path euRun!
ConnectionFile = "file:///opt/eudaq2/user/eudet/misc/hw_conf/aida_tlu/aida_tlu_connection.xml"
# relative path from execution directory
DeviceName = "aida_tlu.controlhub"
# Set CONFCLOCK to 1 to configure clock, which is necessary after a power cycle
CONFCLOCK = 1
# Path to clock file
CLOCK_CFG_FILE = "/opt/eudaq2/user/eudet/misc/hw_conf/aida_tlu/aida_tlu_clk_config.txt"
# Set skipini to 1, if you want to skip the init-step
skipini = 0
```

```
# further expert setting, do not change
nDUTs = 4               # number of HDMI inputs, leave 4 even if you only use fewer inputs
nTrgIn = 6
intRefOn = 0            # 0 = False (Internal Reference OFF), 1 = True
VRefInt = 2.5
VRefExt = 1.3
I2C_COREEXP_Addr = 0x21 # I2C address of the bus expander on Enclustra FPGA
I2C_CLK_Addr = 0x68     # I2C address of the Si5345
I2C_DAC1_Addr = 0x13    # I2C address of 1st AD5665R
I2C_DAC2_Addr = 0x1F    # I2C address of 2nd AD5665R
I2C_ID_Addr = 0x50      # I2C address of unique Id number EEPROM
I2C_EXP1_Addr = 0x74    # I2C address of 1st expander PCA9539PW
I2C_EXP2_Addr = 0x75    # I2C address of 2nd expander PCA9539PW
```

These paths have to be adjusted to match the locations on the PC with the Producer

Parameters to configure the TLU - Configuration

```
[Producer.aida_tlu]
verbose = 0
confid = 20180910
skipconf = 0

##########################################################
# DUT IN/OUTPUT
# Mask: 0 CONT, 1 SPARE, 2 TRIG, 3 BUSY (1 = driven by TLU, 0 = driven by DUT)
# EUDET mode: 7
HDMI1_set = 0x7
HDMI2_set = 0x7
HDMI3_set = 0x7
HDMI4_set = 0x7

# same as above for the clock line, 1 = AIDA mode, 2 = FPGA
HDMI1_clk = 0
HDMI2_clk = 0
HDMI3_clk = 0
HDMI4_clk = 0
LEMOclk = 1   # if input, then also adjust clk.txt

# DUTs
DUTMask=0x1
# Define mode: 2 bits per channel
DUTMaskMode = 0xFC # 1st is reading out

#Bitmask to ignore busy
DUTIgnoreBusy = 0x0

##########################################
# AUTOTRIGGER
InternalTriggerFreq = 0
```

```
#  2 words 32bit: Hi + Lo
# combinations of coincidence are now possible!
trigMaskHi = 0x00000000
trigMaskLo = 0x00000000

# Define the data collector to be used by the producer
EUDAQ_DC = tlu_dc
```

```
# AUTOTRIGGER
InternalTriggerFreq = 0

# EXTERNAL TRIGGER INPUTs
# Stretch, delay in 6.25ns ticks
in0_STR = 1  # factor to stretch
in0_DEL = 0  # factor to delay,
in1_STR = 1
in1_DEL = 0
in2_STR = 1
in2_DEL = 0
in3_STR = 1
in3_DEL = 0
in4_STR = 0
in4_DEL = 0
in5_STR = 0
in5_DEL = 0

# DAC INPUT THRESHOLD
DACThreshold0 = -0.04
DACThreshold1 = -0.04
DACThreshold2 = -0.04
DACThreshold3 = -0.04
DACThreshold4 = -0.20
DACThreshold5 = -0.20

# PMT Power
PMT1_V = 0.80
PMT2_V = 0.80
PMT3_V = 0.00
PMT4_V = 0.00
```

.de <today>

Defining the active input channels to trigger on

Trigger on coincidence of I0 & I1

I0+I1 == line 3
0b1000 ➔ 0x8
trigger_mask = 0x8

First little hands on :)

Define the trigger mask for the (non exclusive) or on channel 2 and 3

word 0x80000000 in the TriggerPattern_lowW register.

DEC	I5	I4	I3	I2	I1	I0	PATTERN	CONFIG. WORD		2^n
0	0	0	0	0	0	0	0			1
1	0	0	0	0	0	1	0	0		2
2	0	0	0	0	1	0	0			4
3	0	0	0	0	1	1	0			8
4	0	0	0	1	0	0	0			16
5	0	0	0	1	0	1	0	0		32
6	0	0	0	1	1	0	0			64
7	0	0	0	1	1	1	0			128
8	0	0	1	0	0	0	0			256
9	0	0	1	0	0	1	0	0		512
10	0	0	1	0	1	0	0			1024
11	0	0	1	0	1	1	0			2048
12	0	0	1	1	0	0	0			4096
13	0	0	1	1	0	1	0	0		8192
14	0	0	1	1	1	0	0			16384
15	0	0	1	1	1	1	0			32768
16	0	1	0	0	0	0	0			65536
17	0	1	0	0	0	1	0	0		131072
18	0	1	0	0	1	0	0			262144
19	0	1	0	0	1	1	0			524288
20	0	1	0	1	0	0	0			1048576
21	0	1	0	1	0	1	0	0		2097152
22	0	1	0	1	1	0	0			4194304
23	0	1	0	1	1	1	0			8388608
24	0	1	1	0	0	0	0			16777216
25	0	1	1	0	0	1	0	0		33554432
26	0	1	1	0	1	0	0			67108864
27	0	1	1	0	1	1	0			134217728
28	0	1	1	1	0	0	0			268435456
29	0	1	1	1	0	1	0	8		536870912
30	0	1	1	1	1	0	0			1073741824
31	0	1	1	1	1	1	1			2147483648

LOWEST 32-bits

Defining the active input channels to trigger on

Trigger on coincidence of I0 & I1

Valid combinations:
I2 & !I3
!I2 & !I3
I2 & I3
➔ 0b 1 0001 0001 0000 ➔ 0x1110

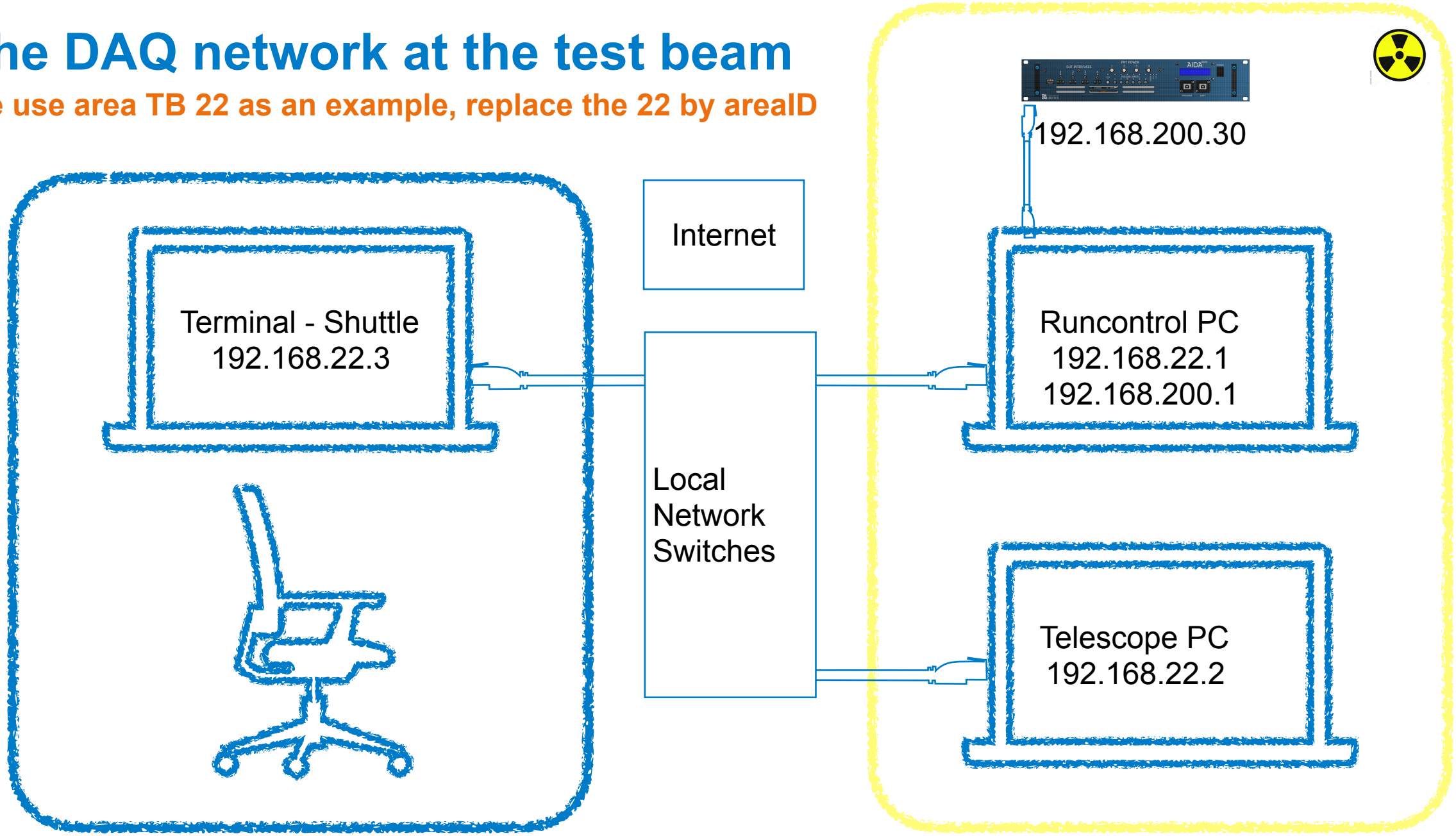
Easy, right :P

word 0x80000000 in the TriggerPattern_lowW register.

DEC	I5	I4	I3	I2	I1	I0	PATTERN	CONFIG. WORD		2^n
0	0	0	0	0	0	0	0 0 0 0	0	LOWEST 32-bits	1
1	0	0	0	0	0	1	0 0 0 0			2
2	0	0	0	0	1	0	0 0 0 0			4
3	0	0	0	0	1	1	0 0 0 0			8
4	0	0	0	1	0	0	0 0 0 0	0		16
5	0	0	0	1	0	1	0 0 0 0			32
6	0	0	0	1	1	0	0 0 0 0			64
7	0	0	0	1	1	1	0 0 0 0			128
8	0	0	1	0	0	0	0 0 0 0	0		256
9	0	0	1	0	0	1	0 0 0 0			512
10	0	0	1	0	1	0	0 0 0 0			1024
11	0	0	1	0	1	1	0 0 0 0			2048
12	0	0	1	1	0	0	0 0 0 0	0		4096
13	0	0	1	1	0	1	0 0 0 0			8192
14	0	0	1	1	1	0	0 0 0 0			16384
15	0	0	1	1	1	1	0 0 0 0			32768
16	0	1	0	0	0	0	0 0 0 0	0		65536
17	0	1	0	0	0	1	0 0 0 0			131072
18	0	1	0	0	1	0	0 0 0 0			262144
19	0	1	0	0	1	1	0 0 0 0			524288
20	0	1	0	1	0	0	0 0 0 0	0		1048576
21	0	1	0	1	0	1	0 0 0 0			2097152
22	0	1	0	1	1	0	0 0 0 0			4194304
23	0	1	0	1	1	1	0 0 0 0			8388608
24	0	1	1	0	0	0	0 0 0 0	0		16777216
25	0	1	1	0	0	1	0 0 0 0			33554432
26	0	1	1	0	1	0	0 0 0 0			67108864
27	0	1	1	0	1	1	0 0 0 0			134217728
28	0	1	1	1	0	0	0 0 0 0	8		268435456
29	0	1	1	1	0	1	1 0 0 0			536870912
30	0	1	1	1	1	0	1 0 0 0			1073741824
31	0	1	1	1	1	1	1 0 0 0			2147483648

The DAQ network at the test beam

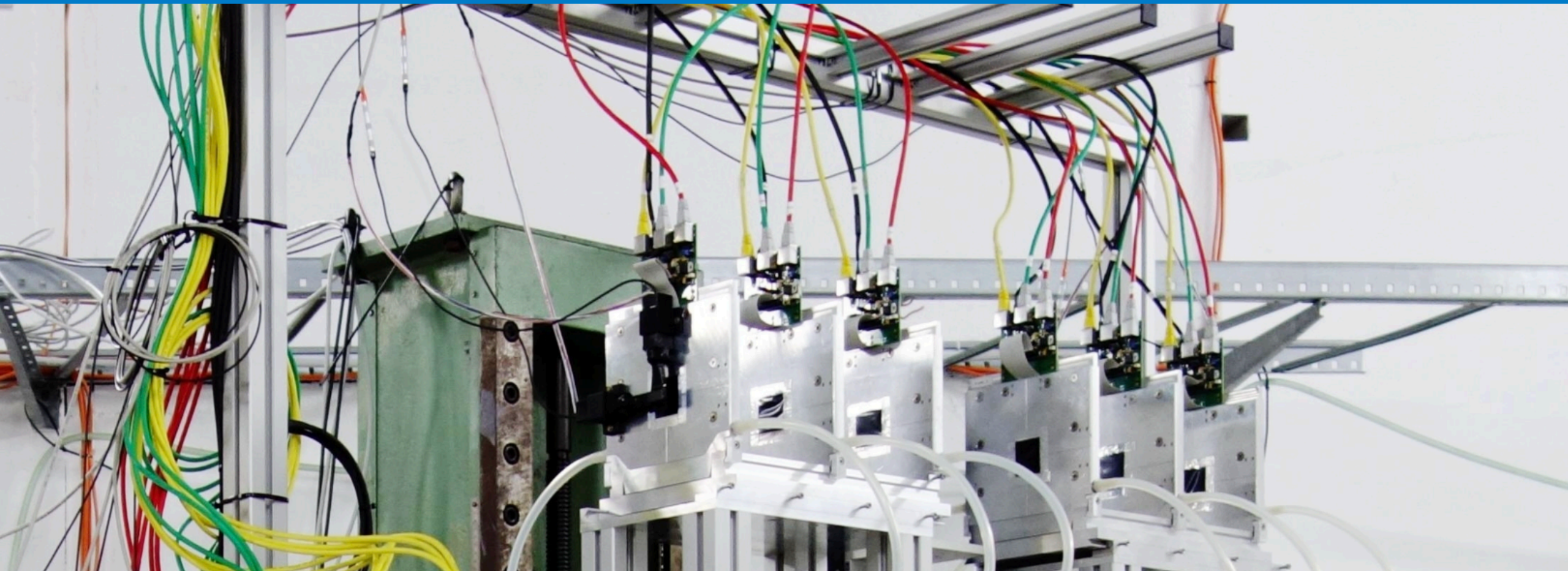
We use area TB 22 as an example, replace the 22 by arealD



A little walk to the test beam…

… to start with the real hands on

Starting up the MIMOSA Telescopes



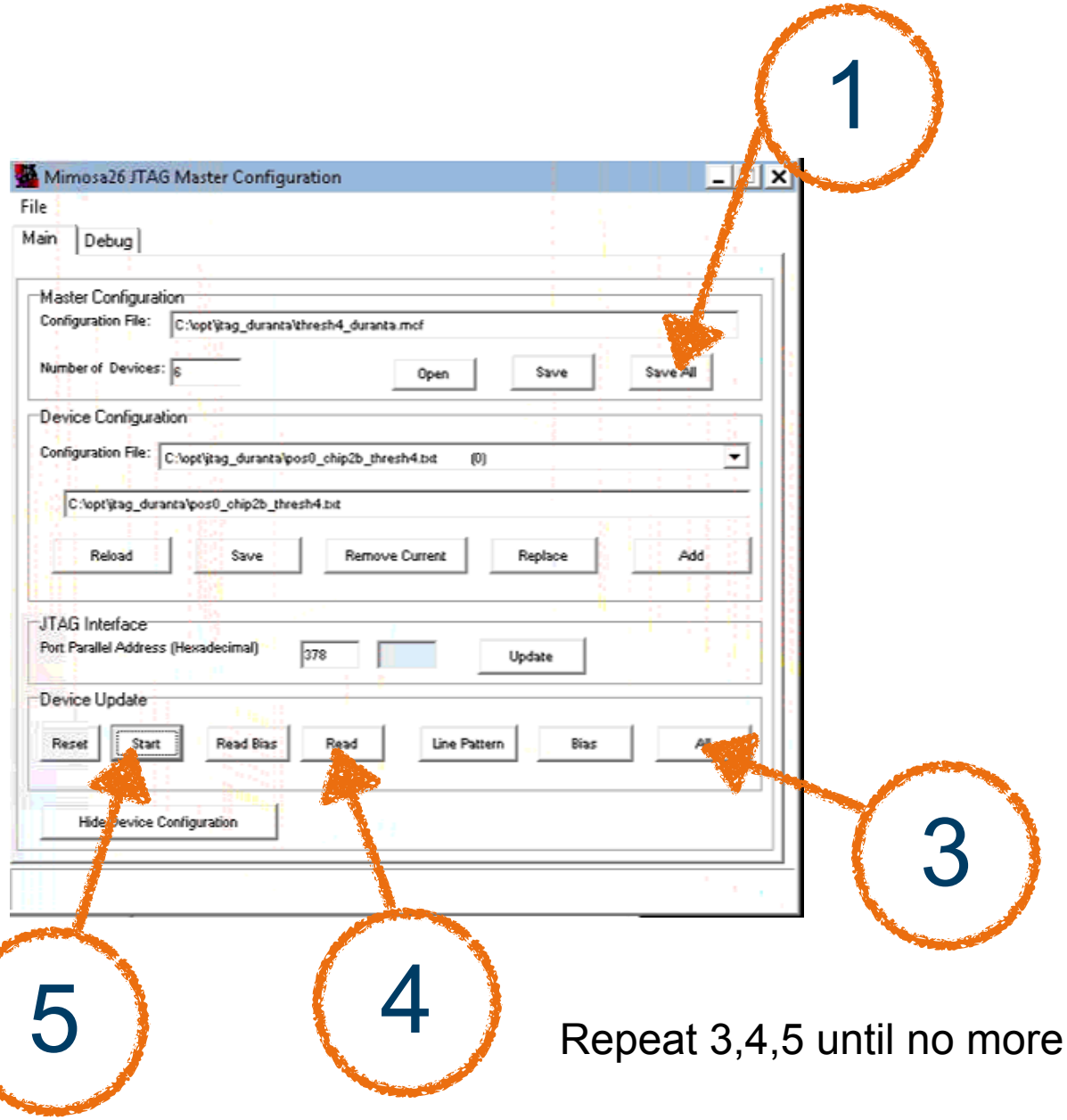
Powering up and starting chiller

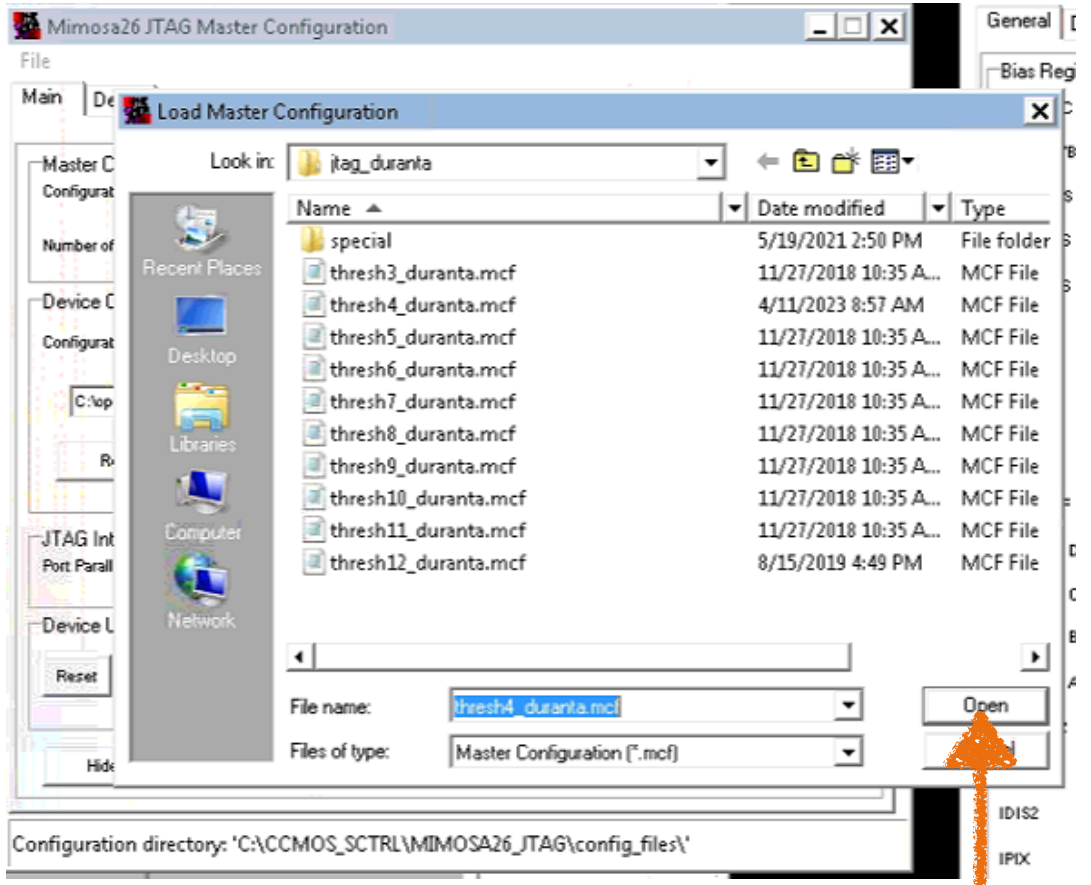
2

1

3

Start up and configure MIMOSAS



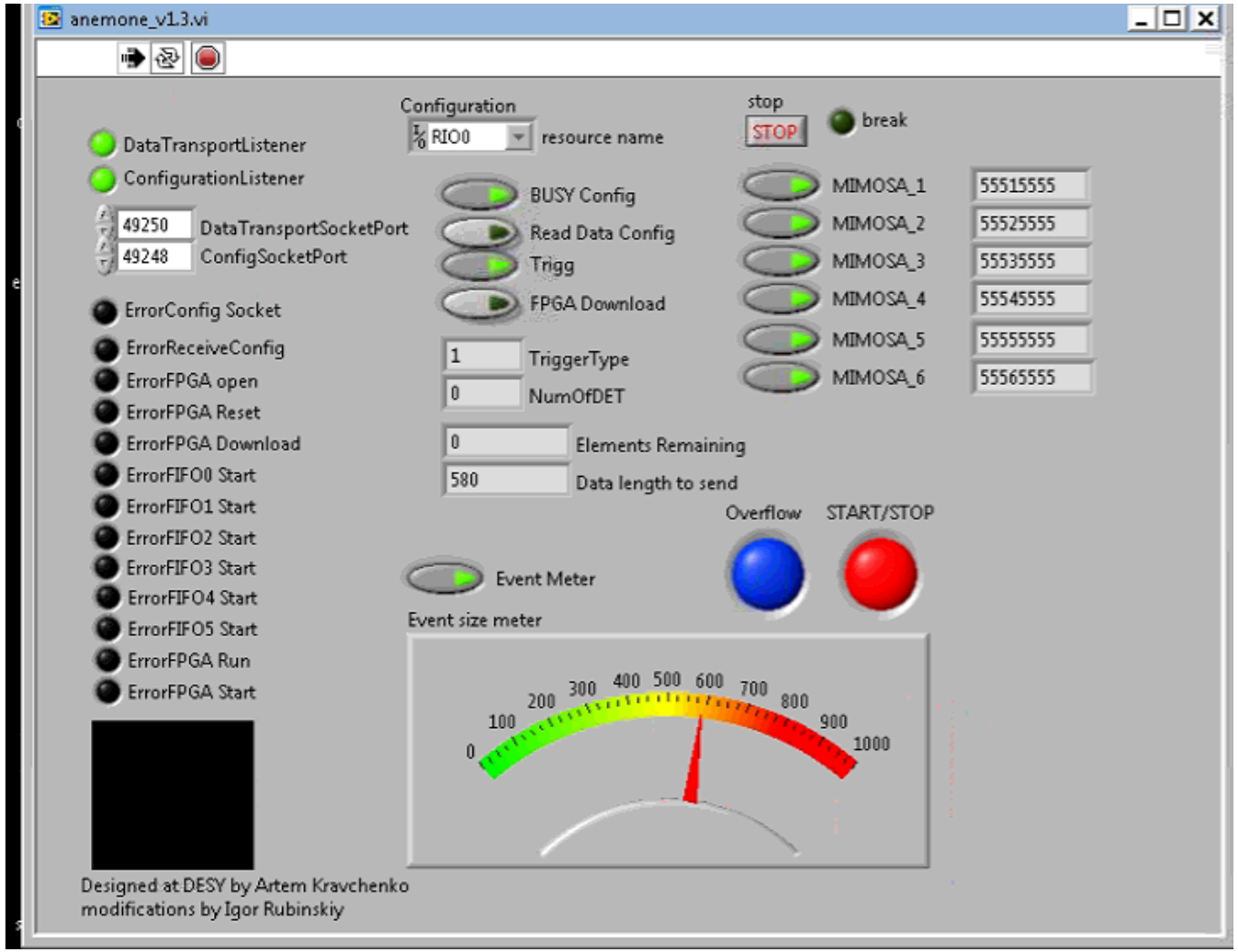


Higher threshold
➔ less noise + reduced efficiency

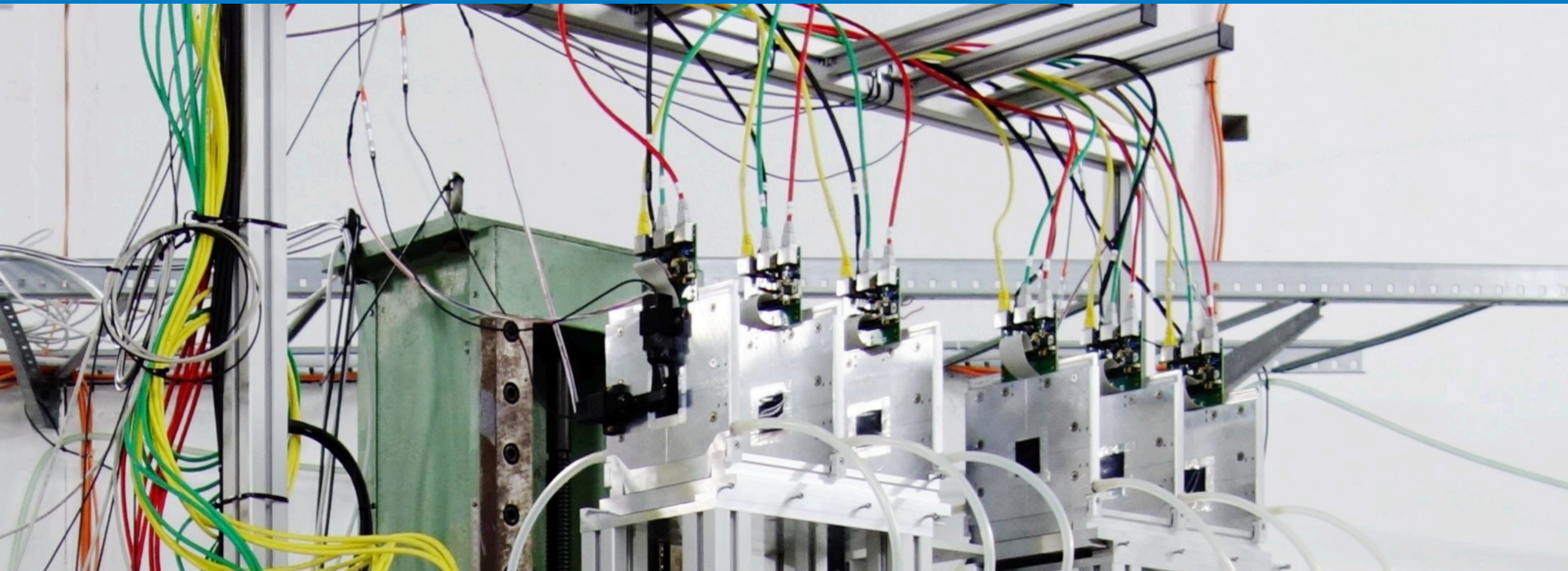
2

Repeat 3,4,5 until no more errors printed

The NI DAQ GUI for the MIMOSAS

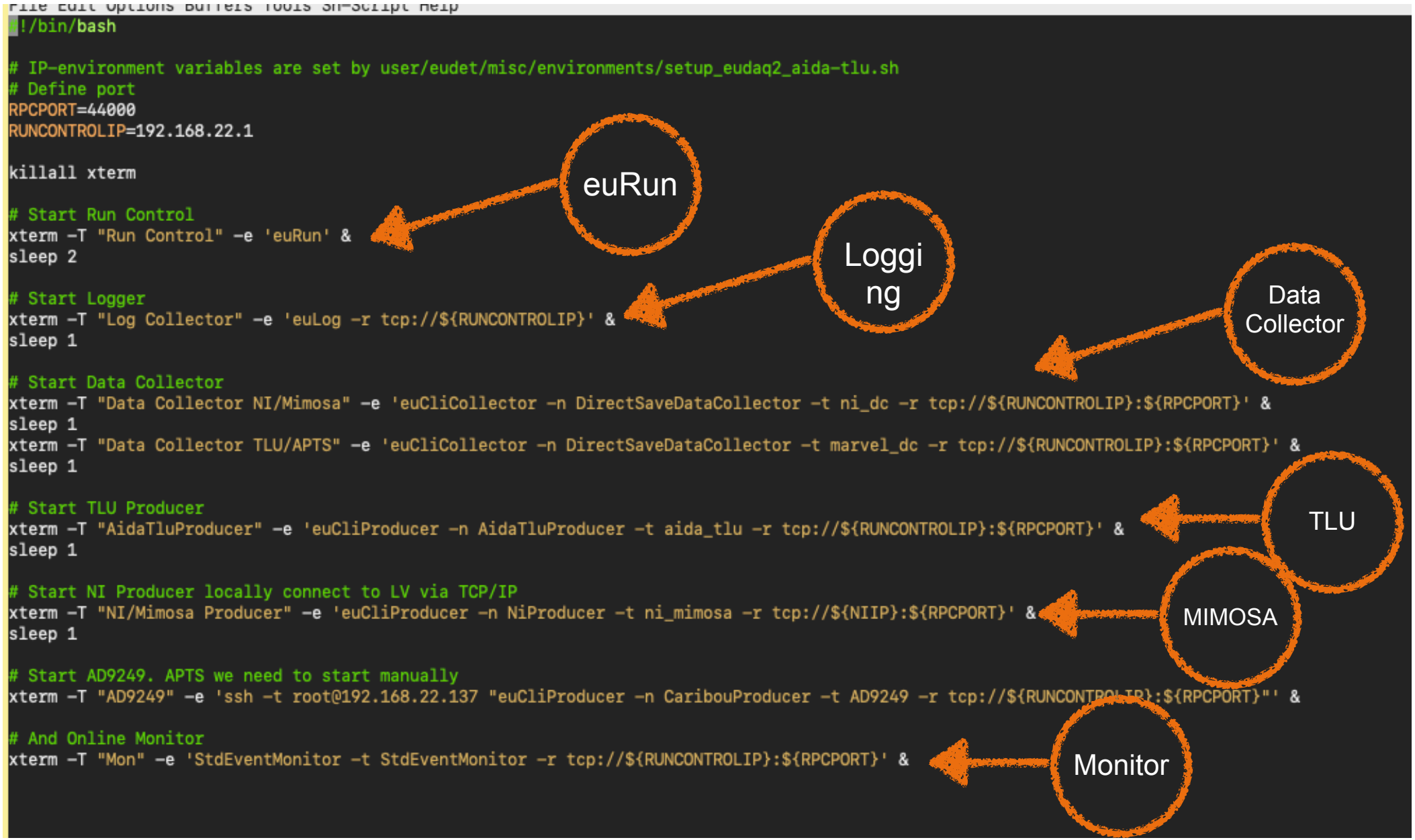


Starting EUDAQ/Connecting all required components

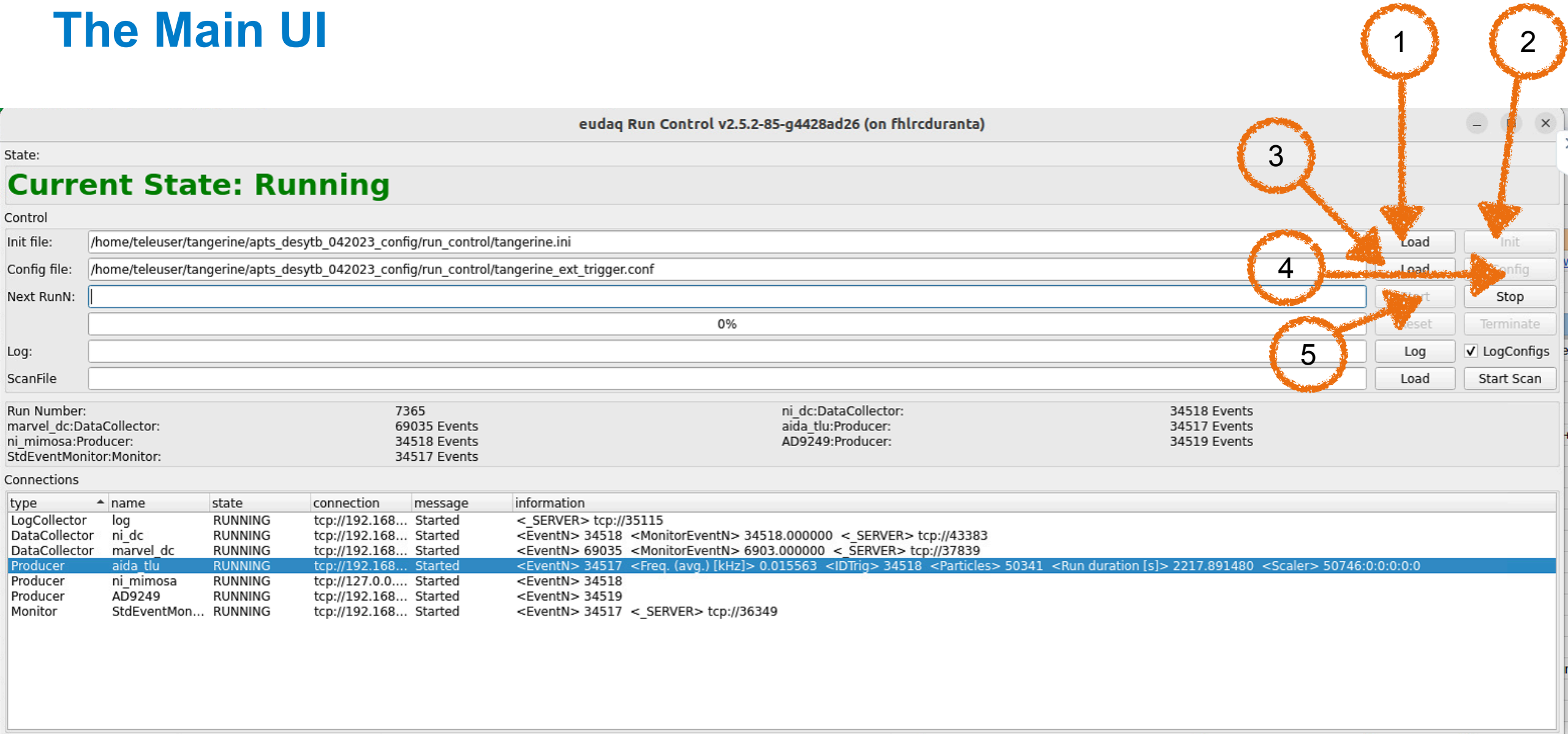


Starting all components

```
File Edit Options Buffers Tools Sh-Script Help
#!/bin/bash

# IP-environment variables are set by user/eudet/misc/environments/setup_eudaq2_aida-tlu.sh
# Define port
RPCPORT=44000
RUNCONTROLIP=192.168.22.1

killall xterm

# Start Run Control
xterm -T "Run Control" -e 'euRun' &
sleep 2

# Start Logger
xterm -T "Log Collector" -e 'euLog -r tcp://${RUNCONTROLIP}' &
sleep 1

# Start Data Collector
xterm -T "Data Collector NI/Mimosa" -e 'euCliCollector -n DirectSaveDataCollector -t ni_dc -r tcp://${RUNCONTROLIP}:${RPCPORT}' &
sleep 1
xterm -T "Data Collector TLU/APTS" -e 'euCliCollector -n DirectSaveDataCollector -t marvel_dc -r tcp://${RUNCONTROLIP}:${RPCPORT}' &
sleep 1

# Start TLU Producer
xterm -T "AidaTluProducer" -e 'euCliProducer -n AidaTluProducer -t aida_tlu -r tcp://${RUNCONTROLIP}:${RPCPORT}' &
sleep 1

# Start NI Producer locally connect to LV via TCP/IP
xterm -T "NI/Mimosa Producer" -e 'euCliProducer -n NiProducer -t ni_mimosa -r tcp://${NIIP}:${RPCPORT}' &
sleep 1

# Start AD9249. APTS we need to start manually
xterm -T "AD9249" -e 'ssh -t root@192.168.22.137 "euCliProducer -n CaribouProducer -t AD9249 -r tcp://${RUNCONTROLIP}:${RPCPORT}"' &

# And Online Monitor
xterm -T "Mon" -e 'StdEventMonitor -t StdEventMonitor -r tcp://${RUNCONTROLIP}:${RPCPORT}' &
```

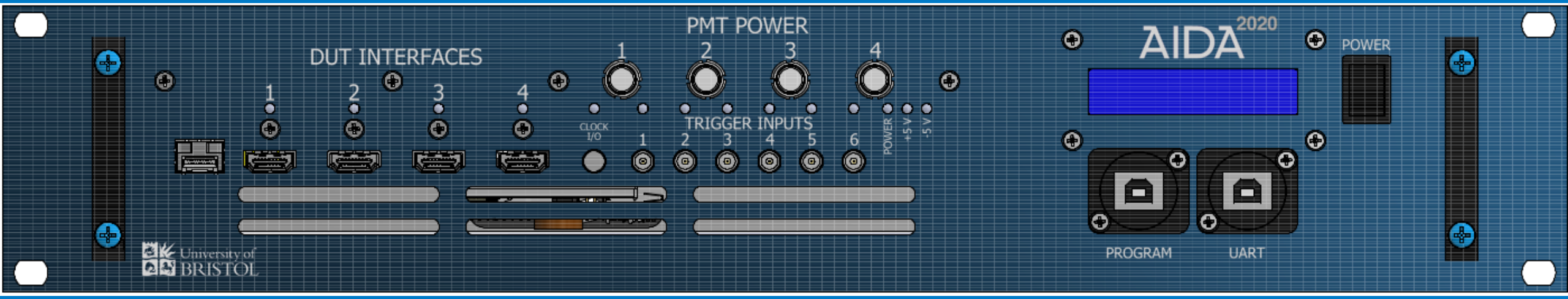


The Main UI



The basic example config/init files can be found in `/home/teleuser/bttb11/*

The trigger modes of the AIDA TLU



Trigger Modes - Recap

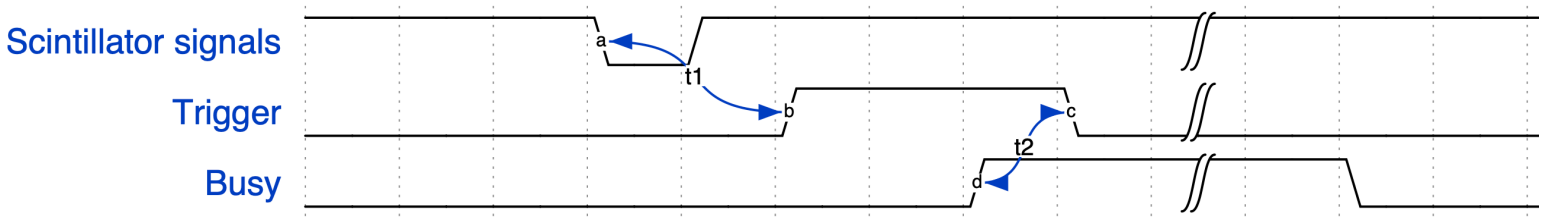
Use all of the modes :)

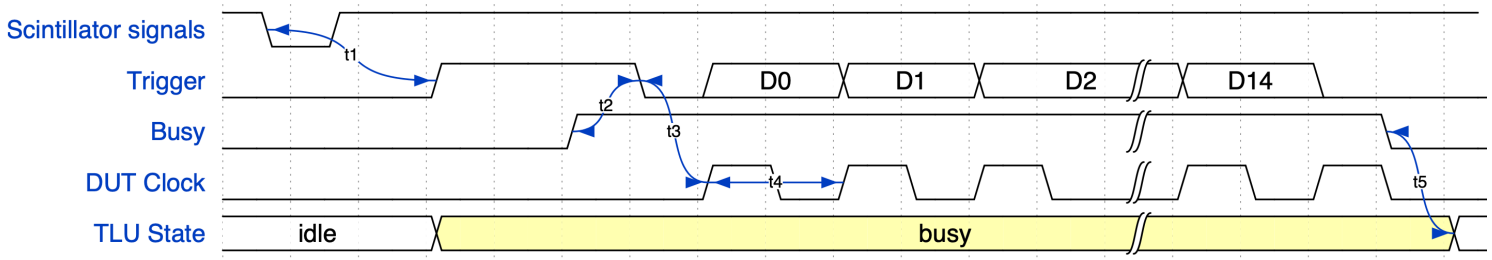
“DUTMaskMode”

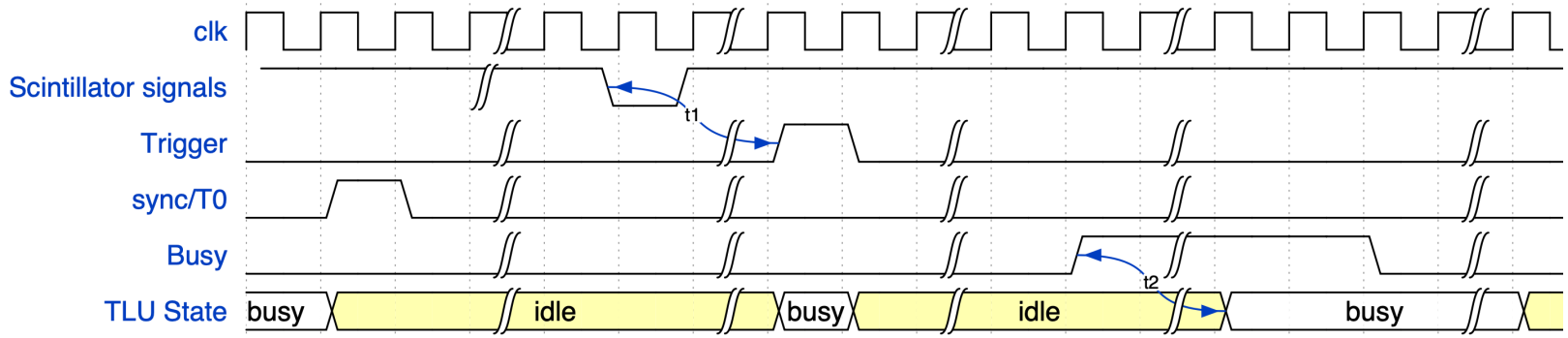
0x00

0x01

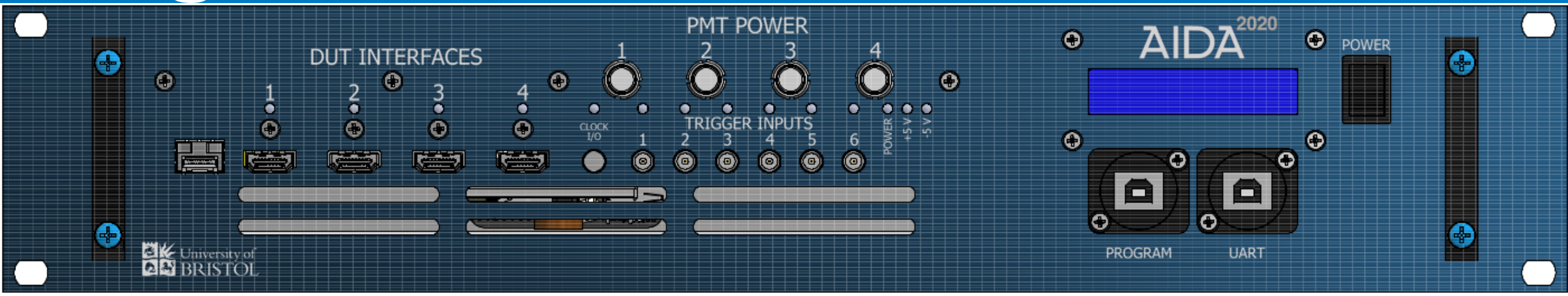
0x11







Optimising the data taking & alignment



1. Start the telescope & connect a HDMI-LEMO board and check all modes
2. Mount the trigger scintillators + adjust the config files to turn the PMT power on, check threshold and see when noise appears
3. Set interlock and turn on beam
4. Observe the trigger/pmt rates ➔ How can you make sure there is no noise
5. Scan the beam momentum and plot the rate as function of momentum, discuss what the optimal conditions for your setup are
6. Go to high rate settings ➔ what effect is the ignore busy having
7. Align the telescope
8. Time delay to compensate for different cable length, scope test first
9. Any other points?

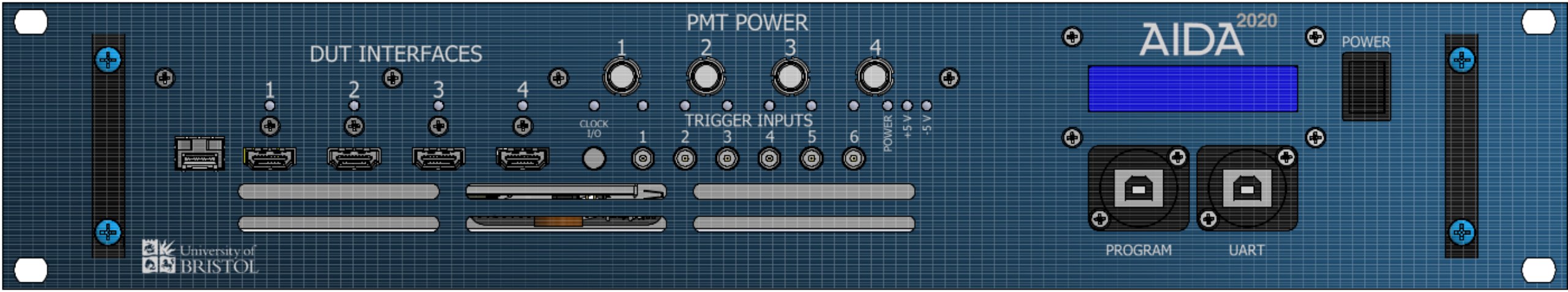
This will be moved to proper slides later on

Summary

TLU Features and interplay with EUDAQ

- The TLU provides several interface modes and flexible triggering
- Depending on your device different modes are most efficient
- EUDAQ2 is capable of steering the TLU
- MIMOSA telescopes are fully integrated ➔ easy starting point for your own test beam DAQ integrations

Contact:
lennart.huth@desy.de
adrian.herkert@desy.de



Any other topics, questions?