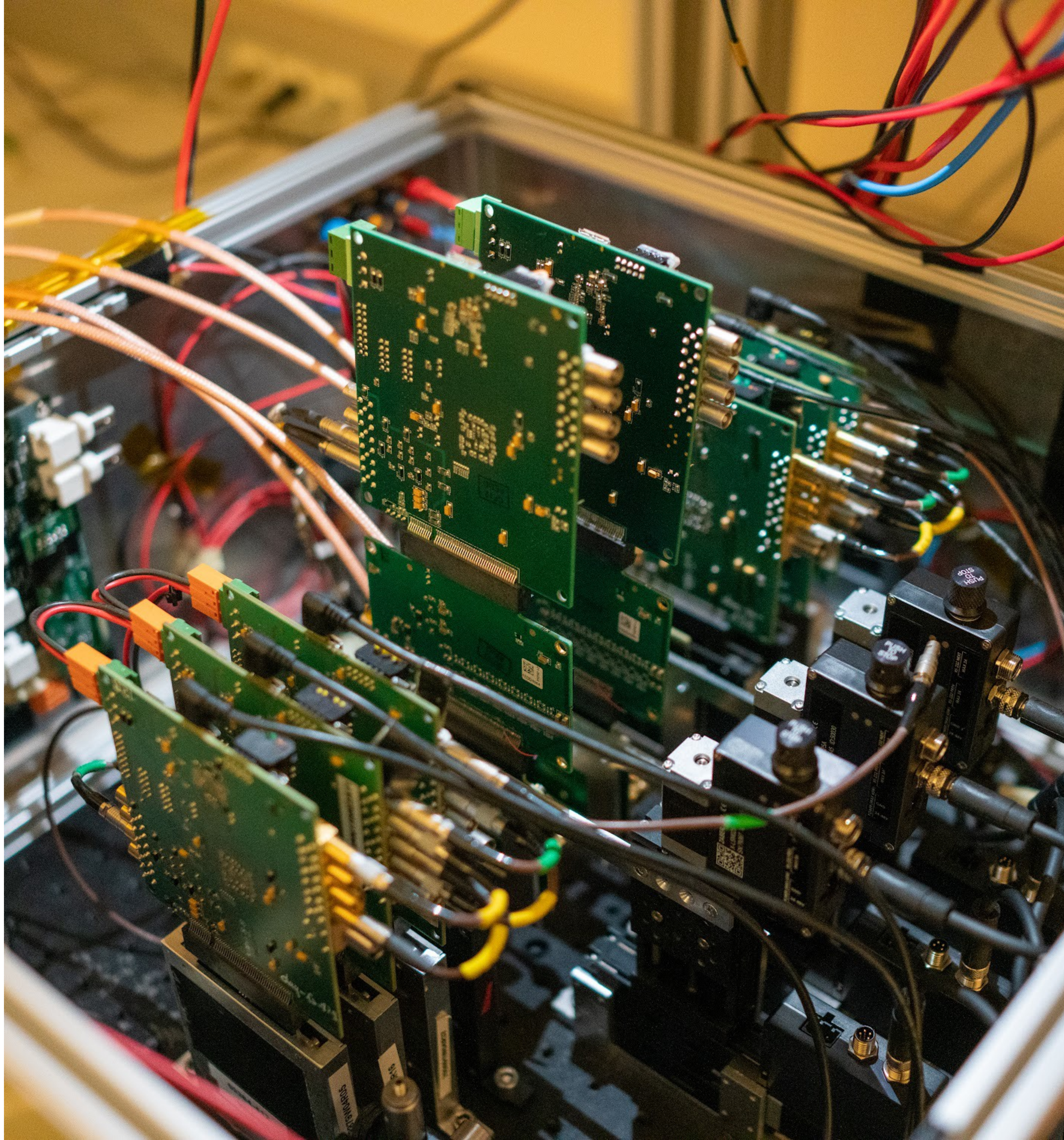
APTS-OA
TB for the Timing resolution

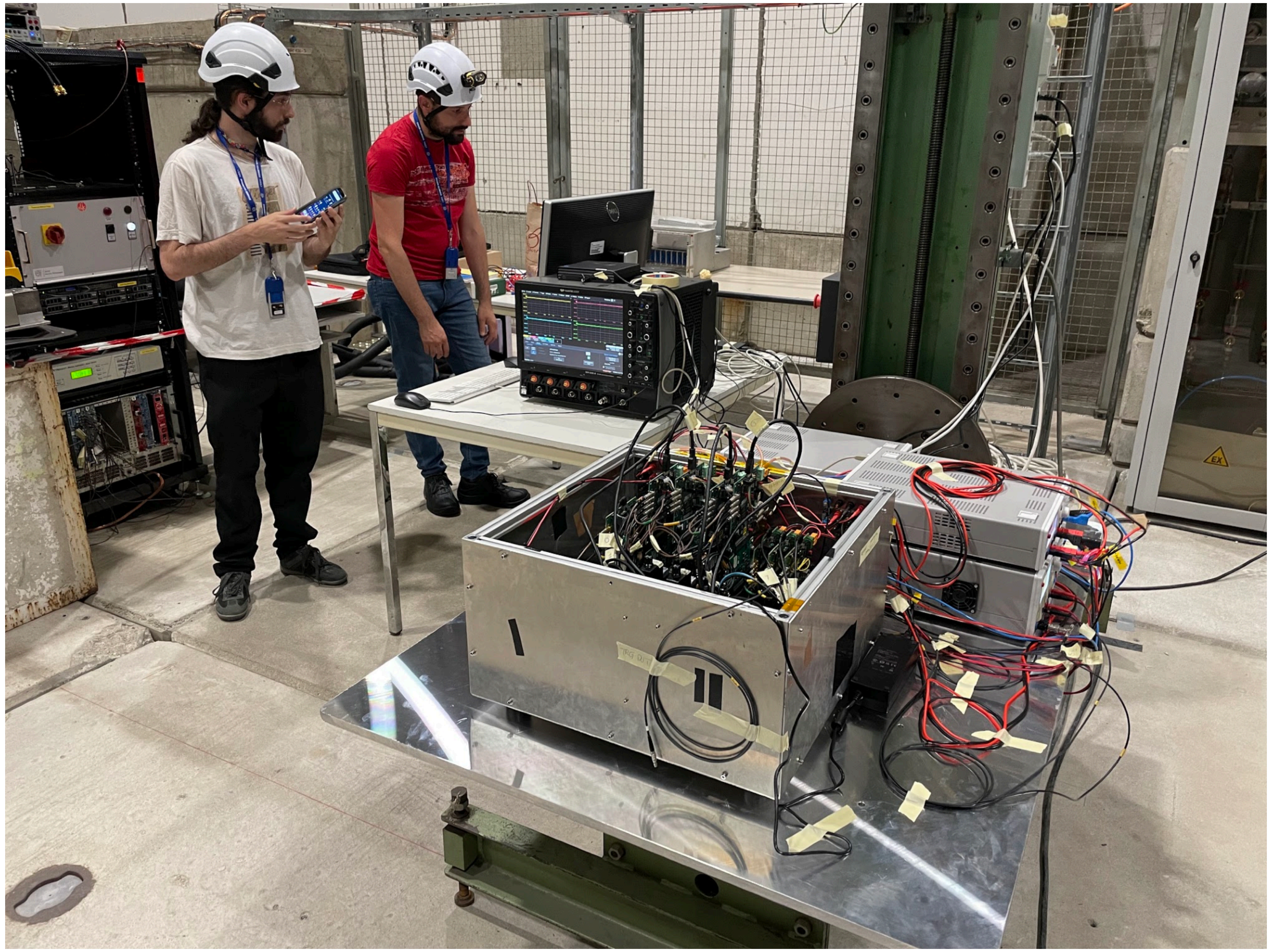
Bong-Hwi Lim
on be half of Università di Torino & INFN



General information on TB

Target and Schedule

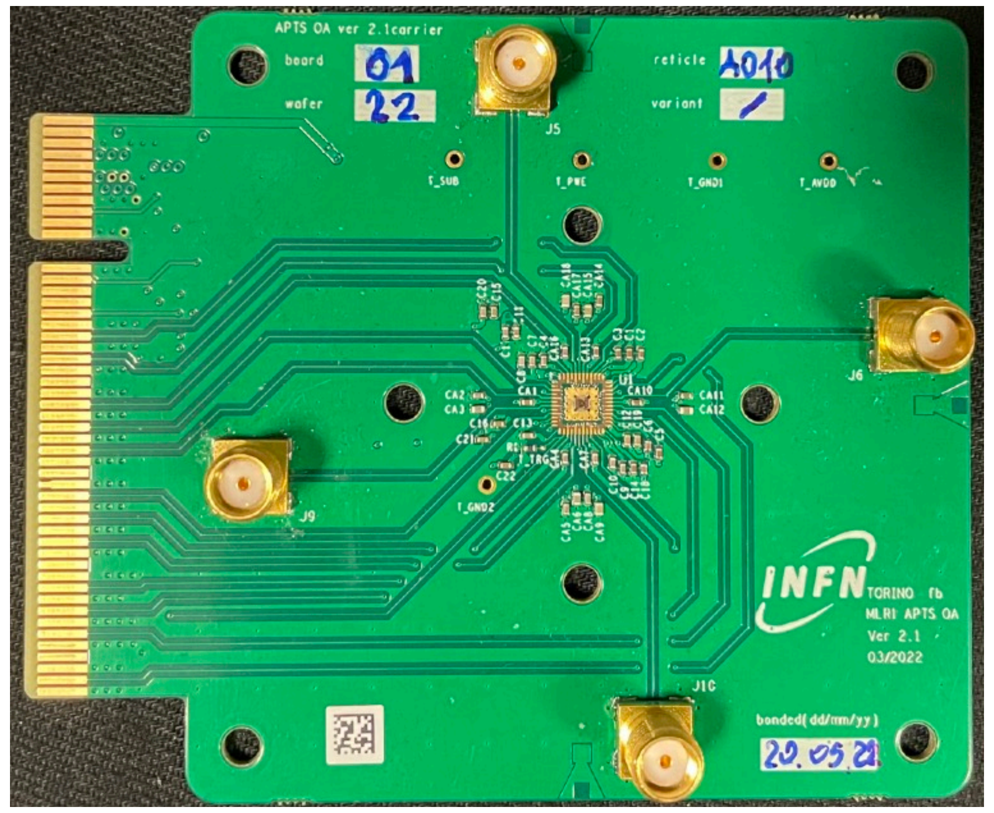
- **Beam:**
  - CERN-SPS, T4-H6, 120 GeV pion
- **Schedule:** 15/06/2022 ~ 07/06/2022 (~ 3 weeks)
  - ~ a week of data taking
- **Target**: Testing the chip performance of APTS-OA
  - **Time resolution**
  - Cluster size, Efficiency
- **Specification:**
  - Use an additional oscilloscope for the fine time resolution.
  - Use APTS-SF for triggering the pixel-level alignment.
  - Use the scope as a trigger (2 pixels trigger)
- **Miscellaneous:**
  - Twiki: https://twiki.cern.ch/twiki/bin/view/ALICE/ITS3WP3SPS2022June
  - Utilities: https://gitlab.cern.ch/alice-its3-wp3/opamp-utils



Setup at SPS, T4-H6

Beam test setup

- **3 REF** / **APTS-SF** / **2 APTS-OA** / **APTSF-SF** / **3 REF**
- **Detector Under Test (DUT)**
  - AO10_06_W22_P: V_bb = **2.4 V**
  - AO10_09_W22_P; V_bb = **2.4 V**
  - bonded on the **APTS carrier V2** Twiki
    - 2 SMA outputs for the oscilloscope
    - 2 outputs connected to the edge connector
- **Oscilloscope**
  - LeCroy WaveMaster 820Zi-B
  - **20** GHz, 4 x **80** GS/s, 25 ps interval
    - 2 ch for AO10_06_W22_P
    - 2 ch for AO10_09_W22_P
- **3 Moving stages**
  - 2 for APTS-OA
  - 1 for APTS-SF



APTS-OA carrier V2

LeCroy WaveMaster 820Zi-B

Moving stage

125 mm — Ref.

100 mm

75 mm

50 mm — APTS-SF

0 mm — APTS-OA

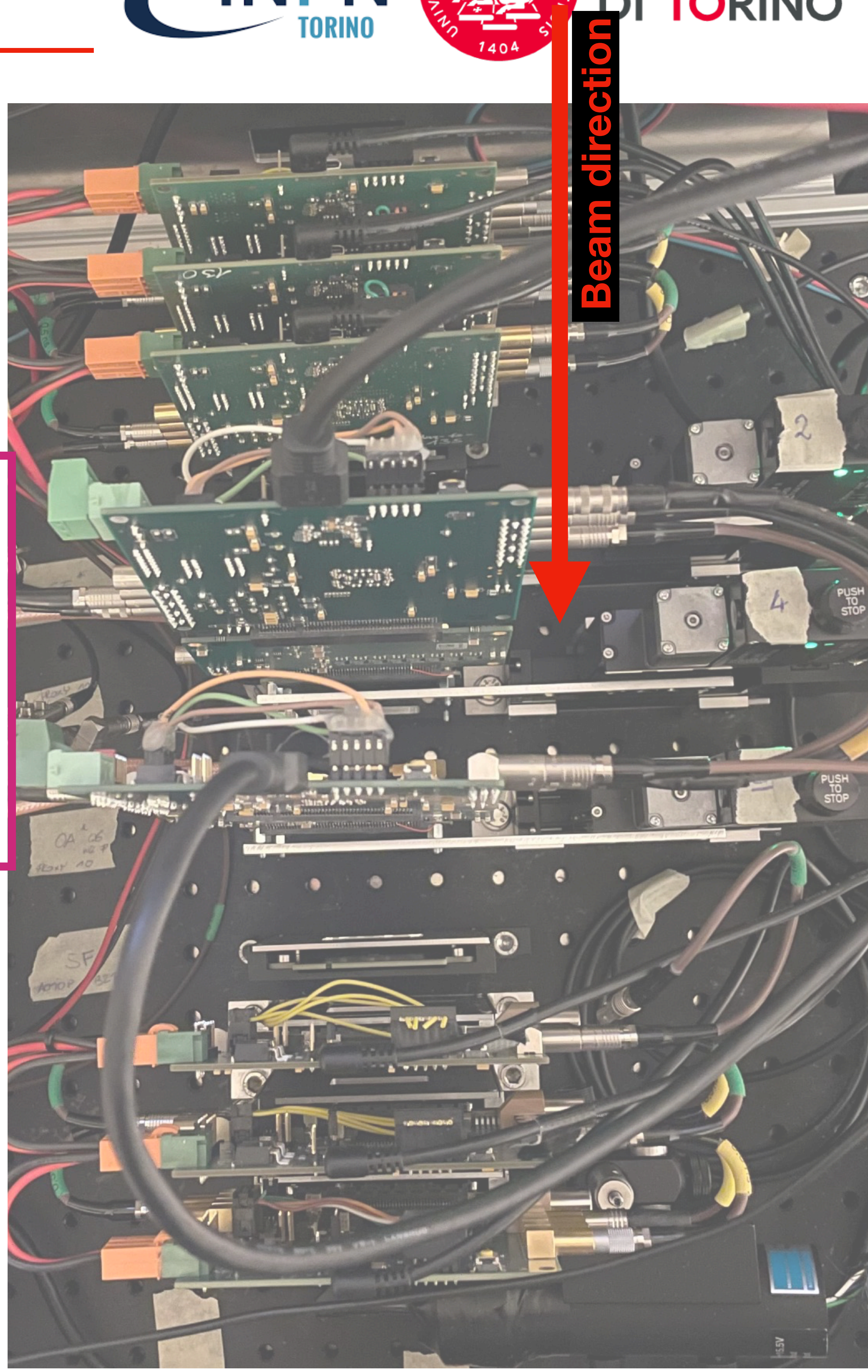
-50 mm — APTS-OA

-100 mm — Fixed APTS-SF

-125 mm

-150 mm

-175 mm

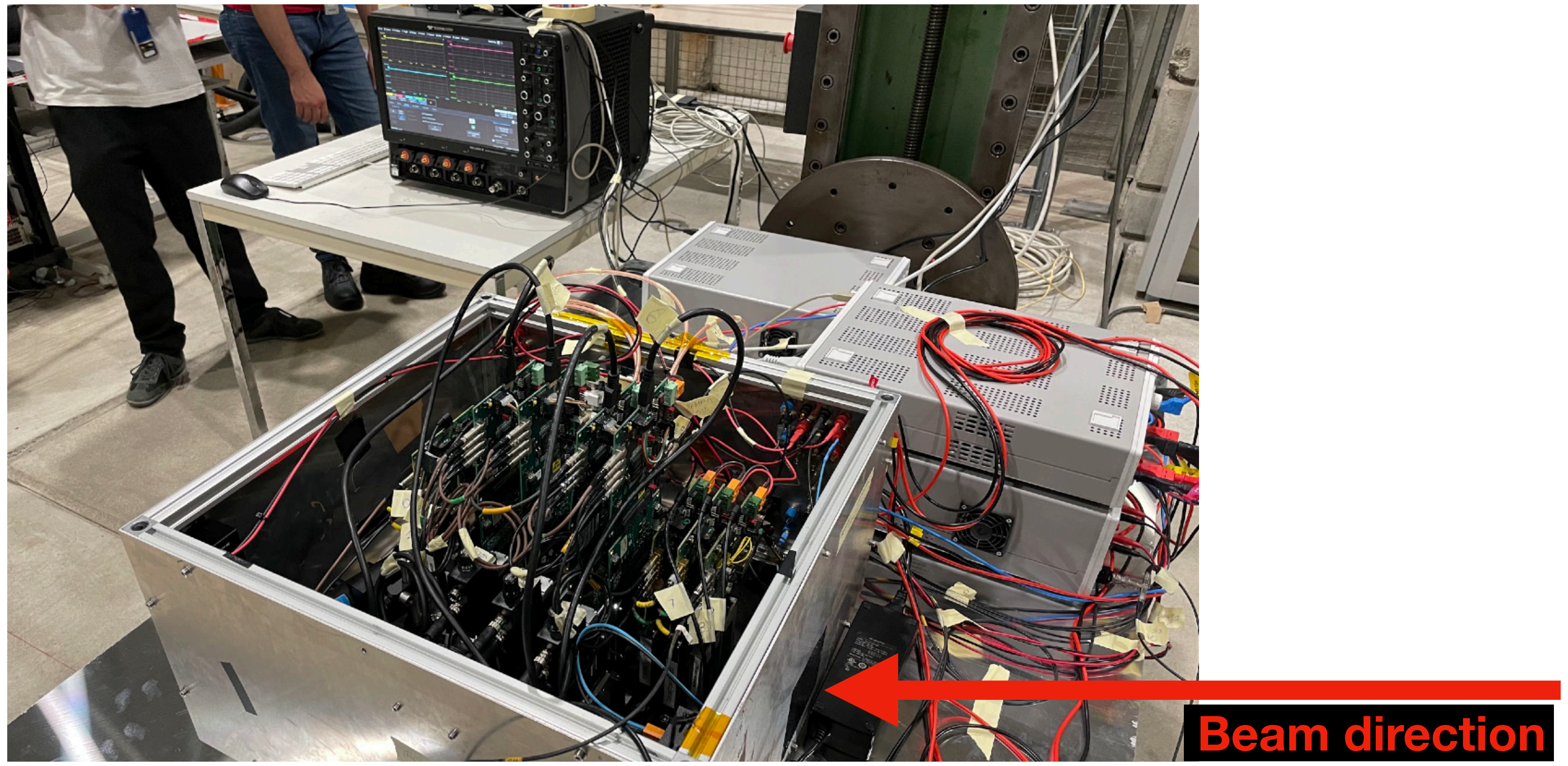


Telescope setup

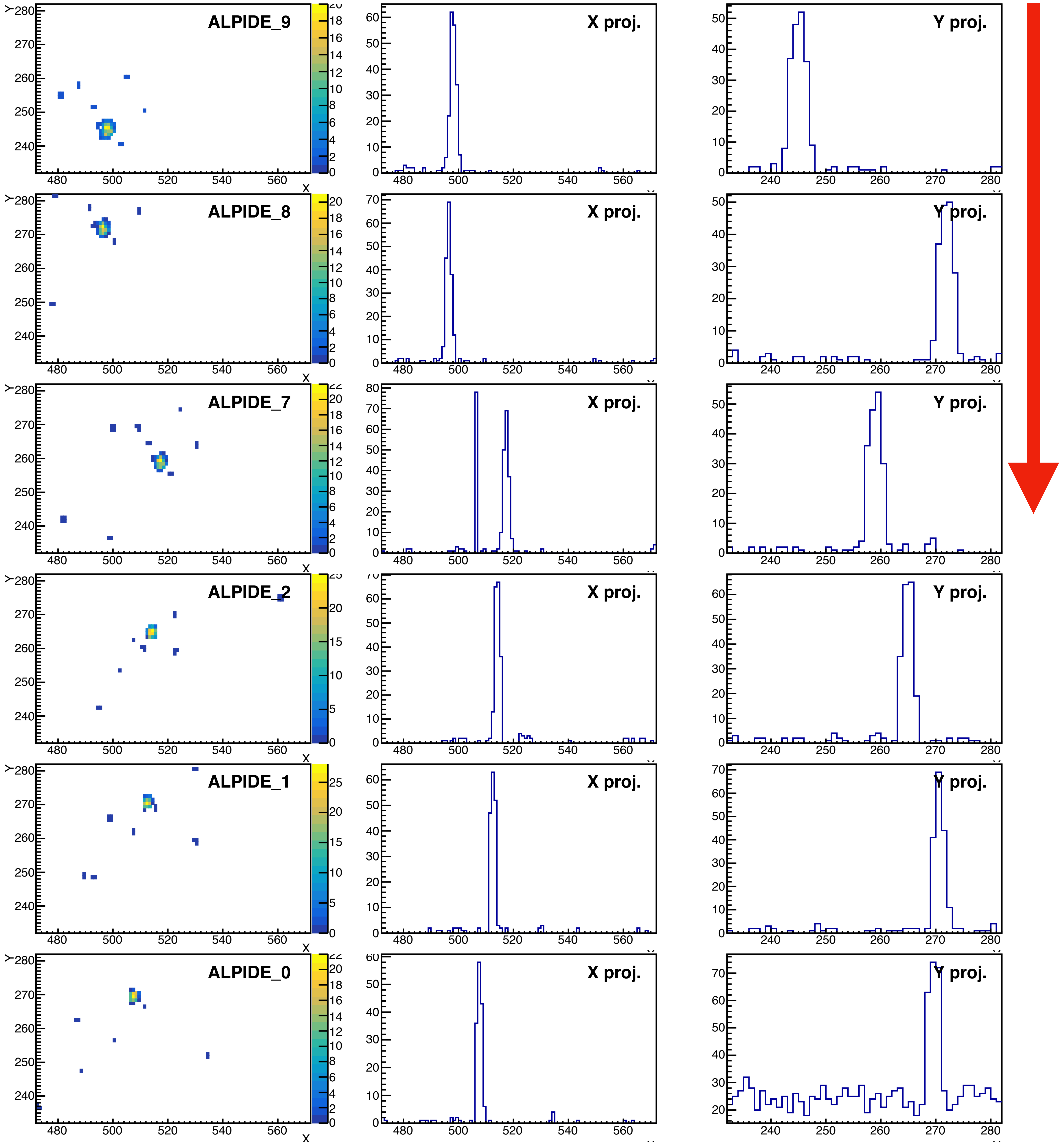
Installation and Alignment

Umberto, Andrea, Paolo, Luciano

- **Telescope setup is installed in <u>SPS-T4-H6</u>**
  - Setup is on the moving table.
  - Scope and DAQ PC is on the another table.
- **Alignment** of the telescope setup
  - First alignment - using **ALPIDE**s with PMT trigger DONE
    - High precision w.r.t. the beam direction
  - Second alignment - using (fixed) **APTS-SF** On going
    - To align APTS-SF - APTS-OA- APTS-OA - APTS-SF in um level accuracy.



Telescope setup and Oscilloscope



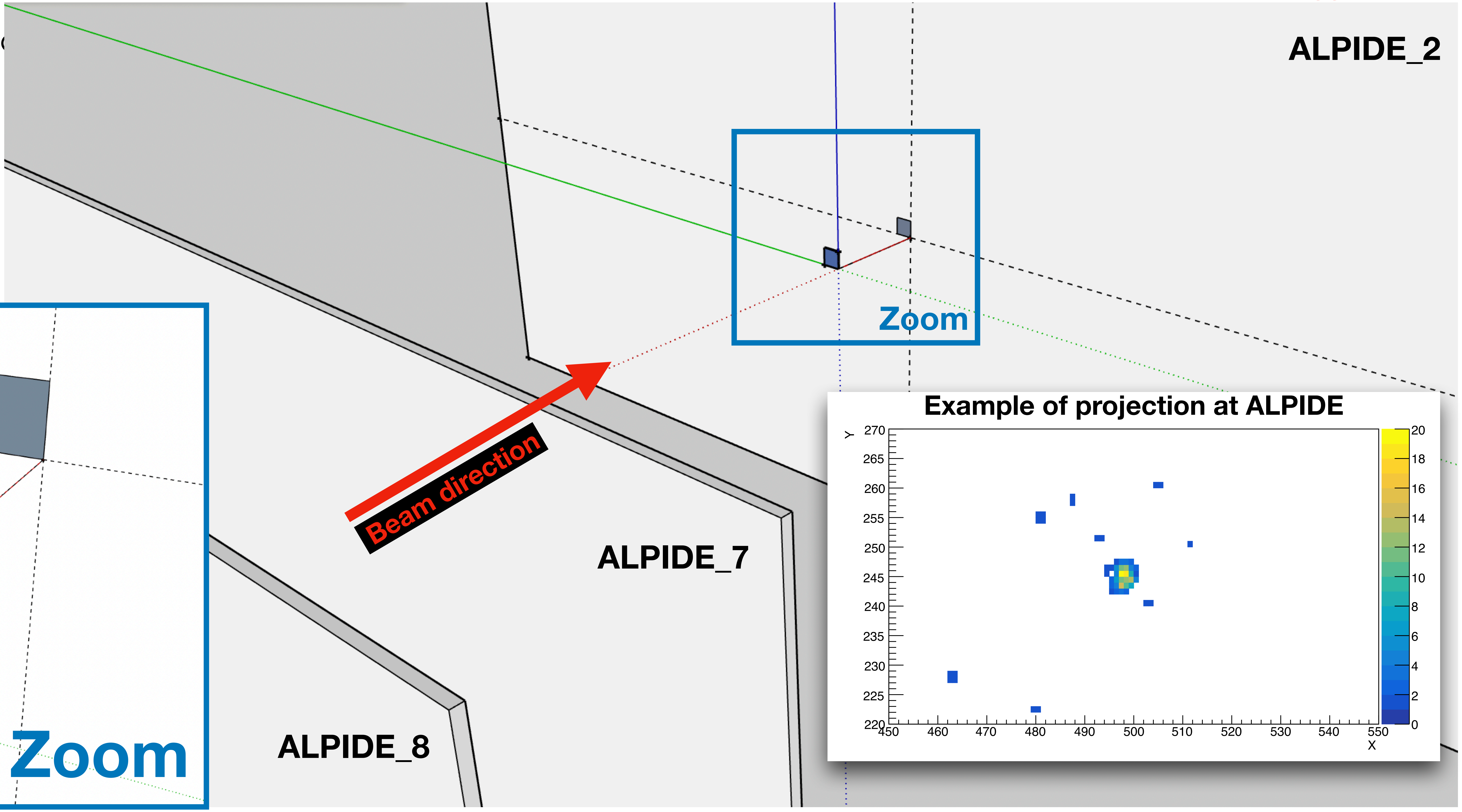
First look of APTS-SF triggered hit map of ALPIDEs

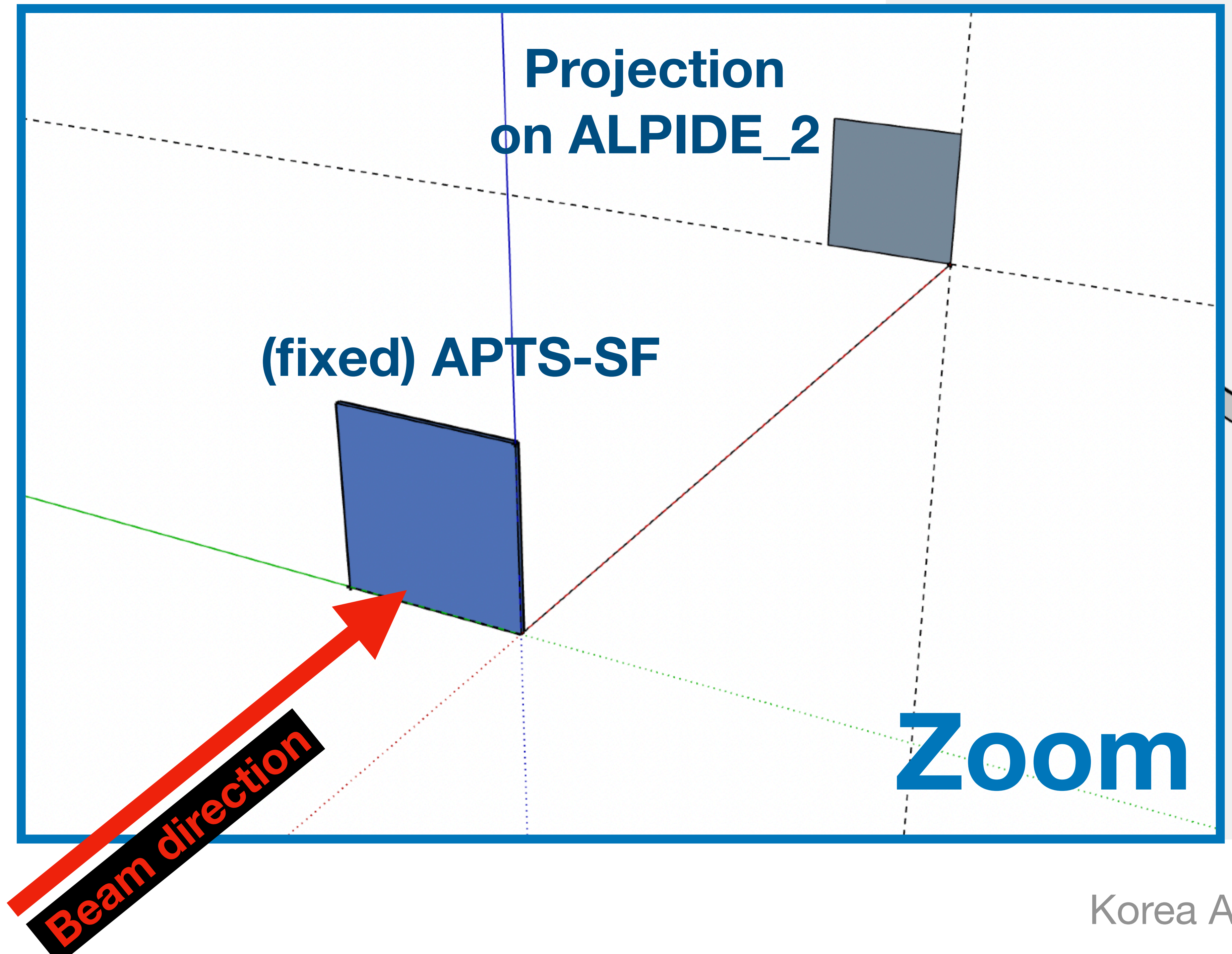
Detailed alignment procedure

#1 Find the position of the shadow(projection) of APTS-SF

- **Use a trigger from the (fixed) APTS-SF**
  - It will make a projection to all of ALPIDEs
  - this will be the referen

Note that all the sizes and distances are exaggerated ! !

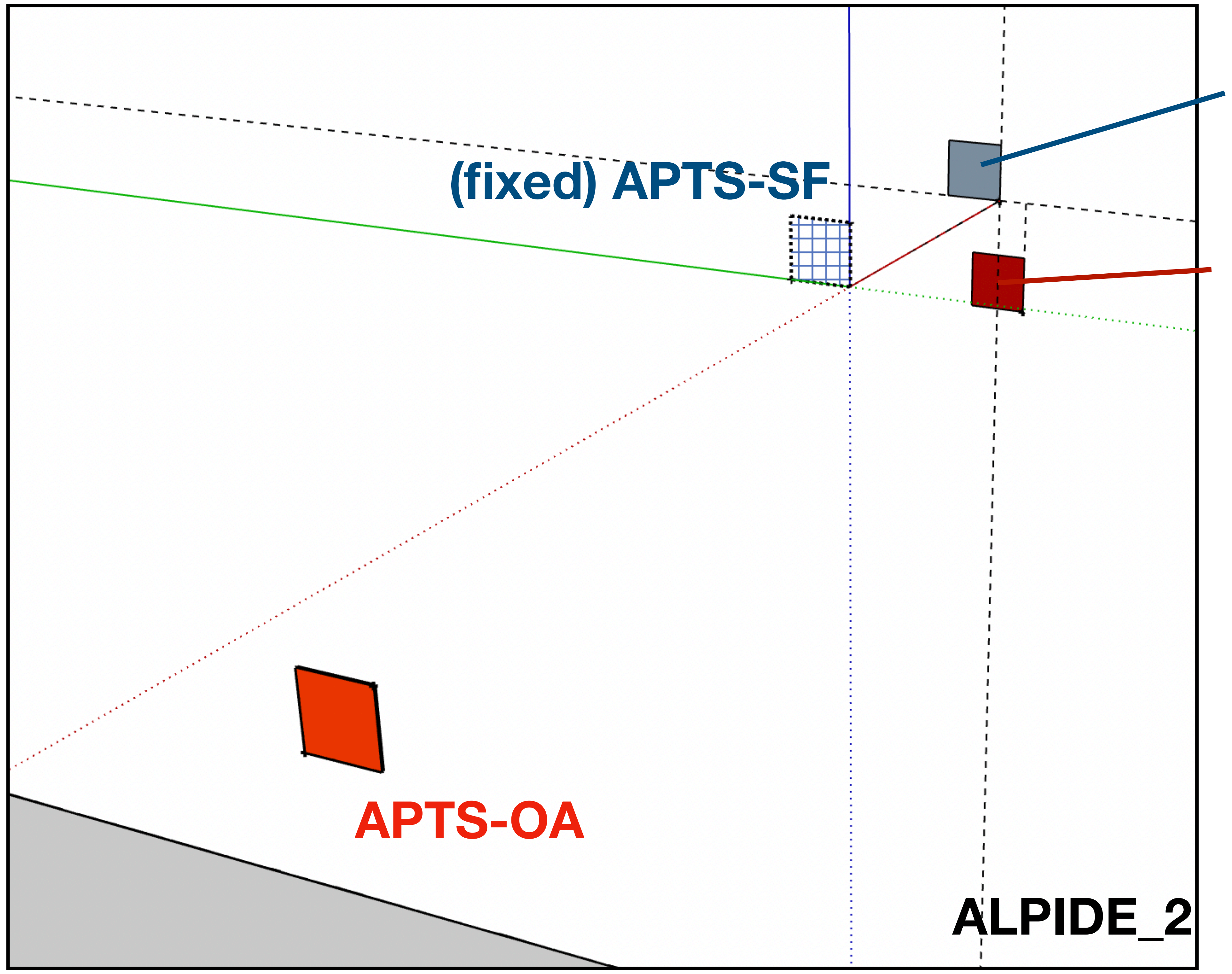


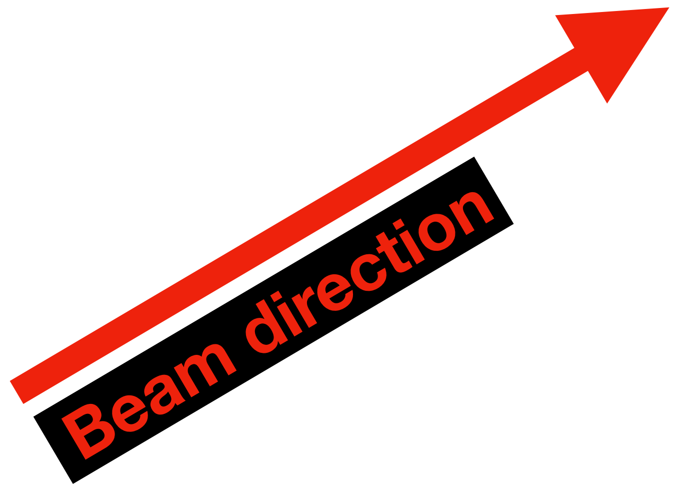


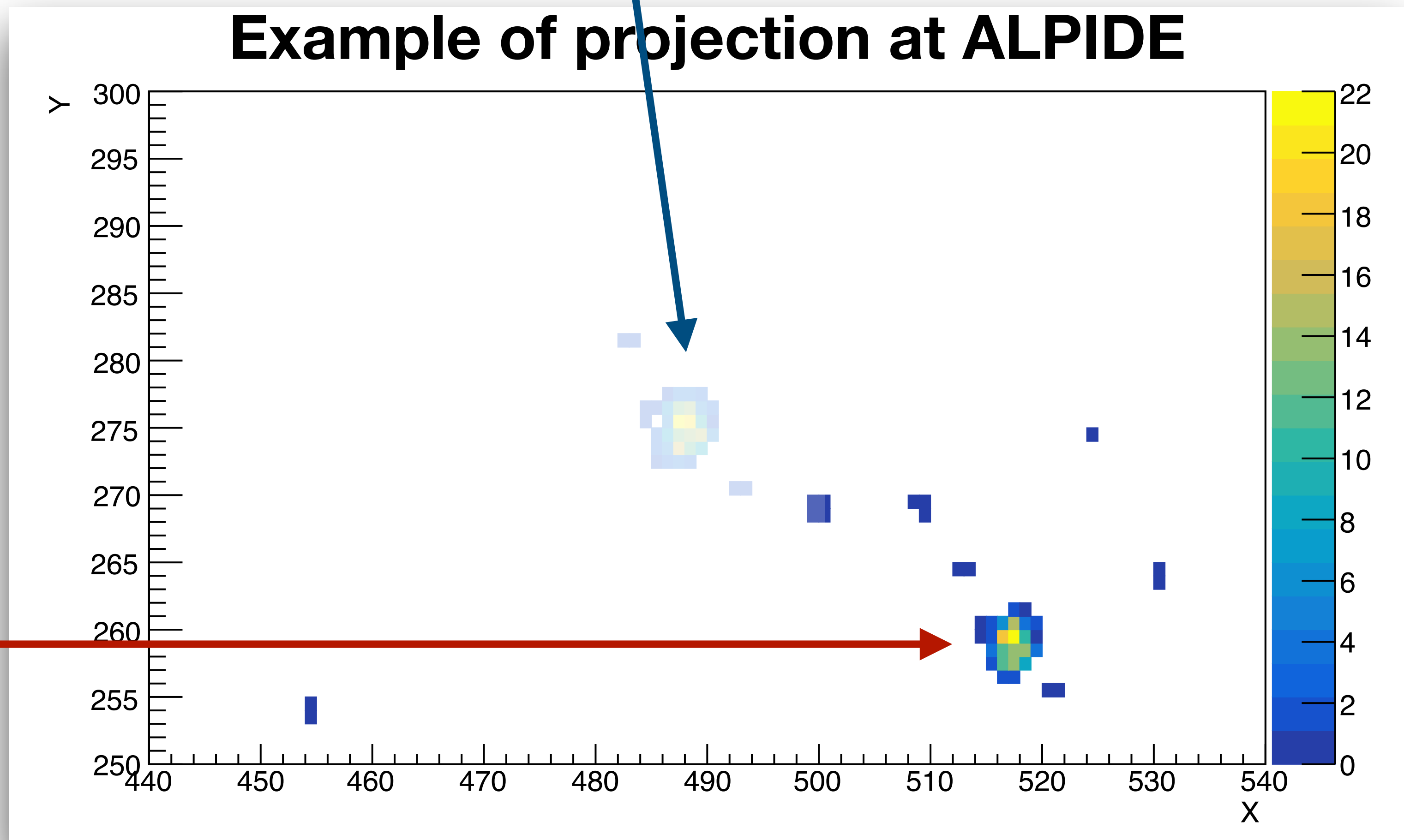
Detailed alignment procedure

#2 Find the position of the shadow(projection) of the other APTSs

- **Use a trigger from the other APTS-SFs**
  - It will make a projection to all of ALPIDEs (Figure, APTS-OA is trigger)
  - The new projection **should be in the same position of the reference.**





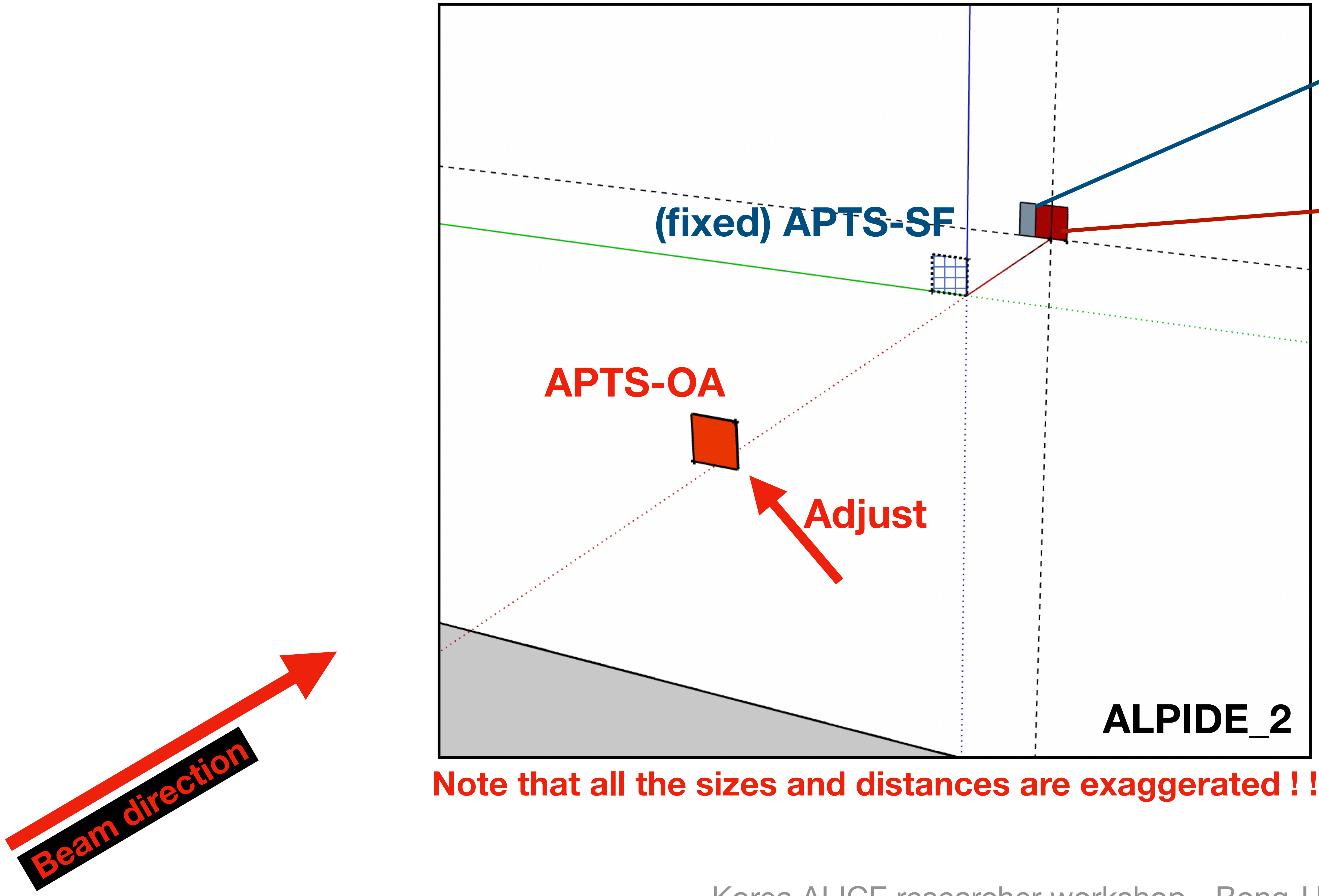


Note that all the sizes and distances are exaggerated ! !

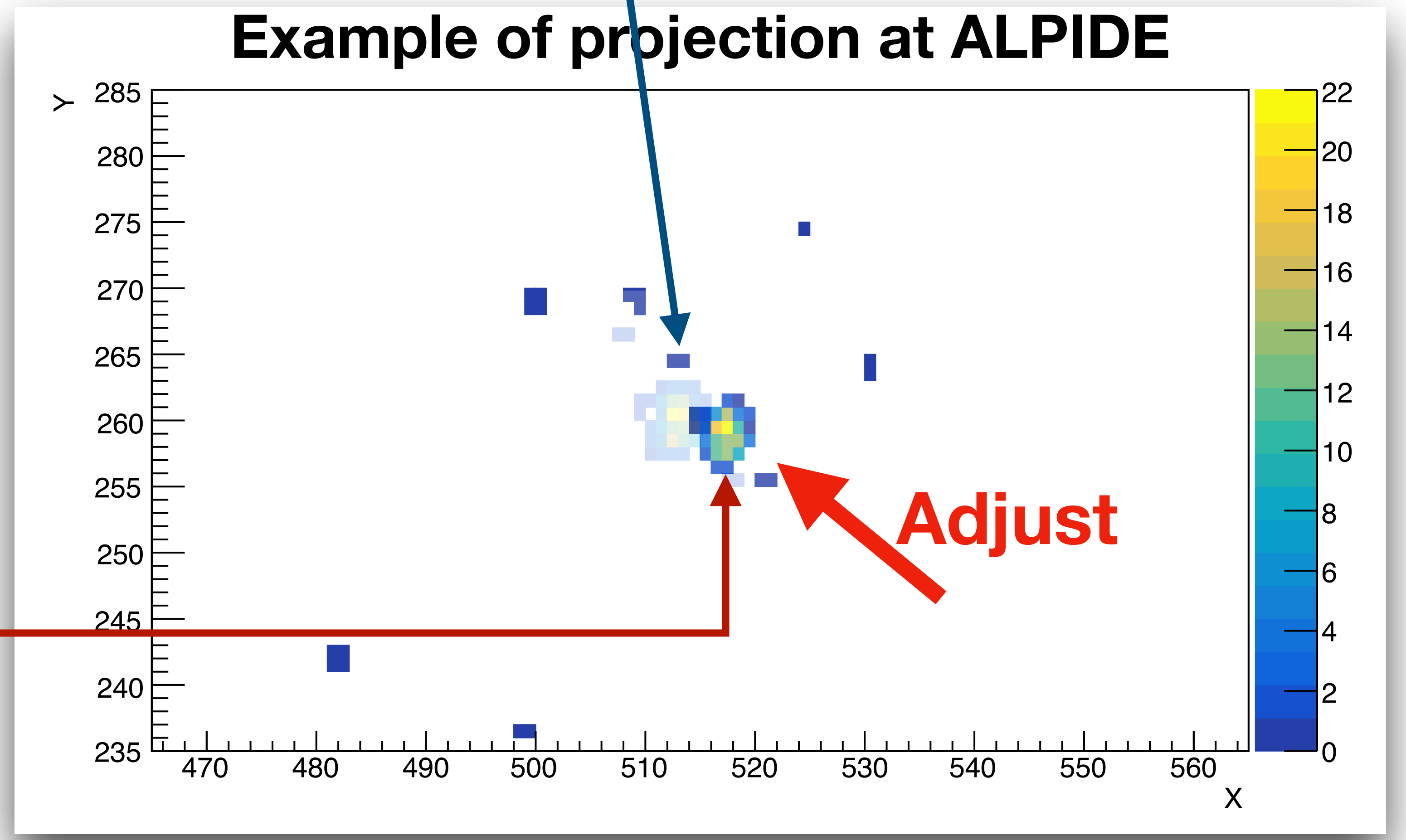
Detailed alignment procedure

#3 Adjust the position of the APTS

- **Use ZABER (moving stage) to adjust the position of the new projection**
  - Once it is well-matched, **iterate this procedure to all APTSs (OPAMP_0, OPAMP_1, APTS_1)**



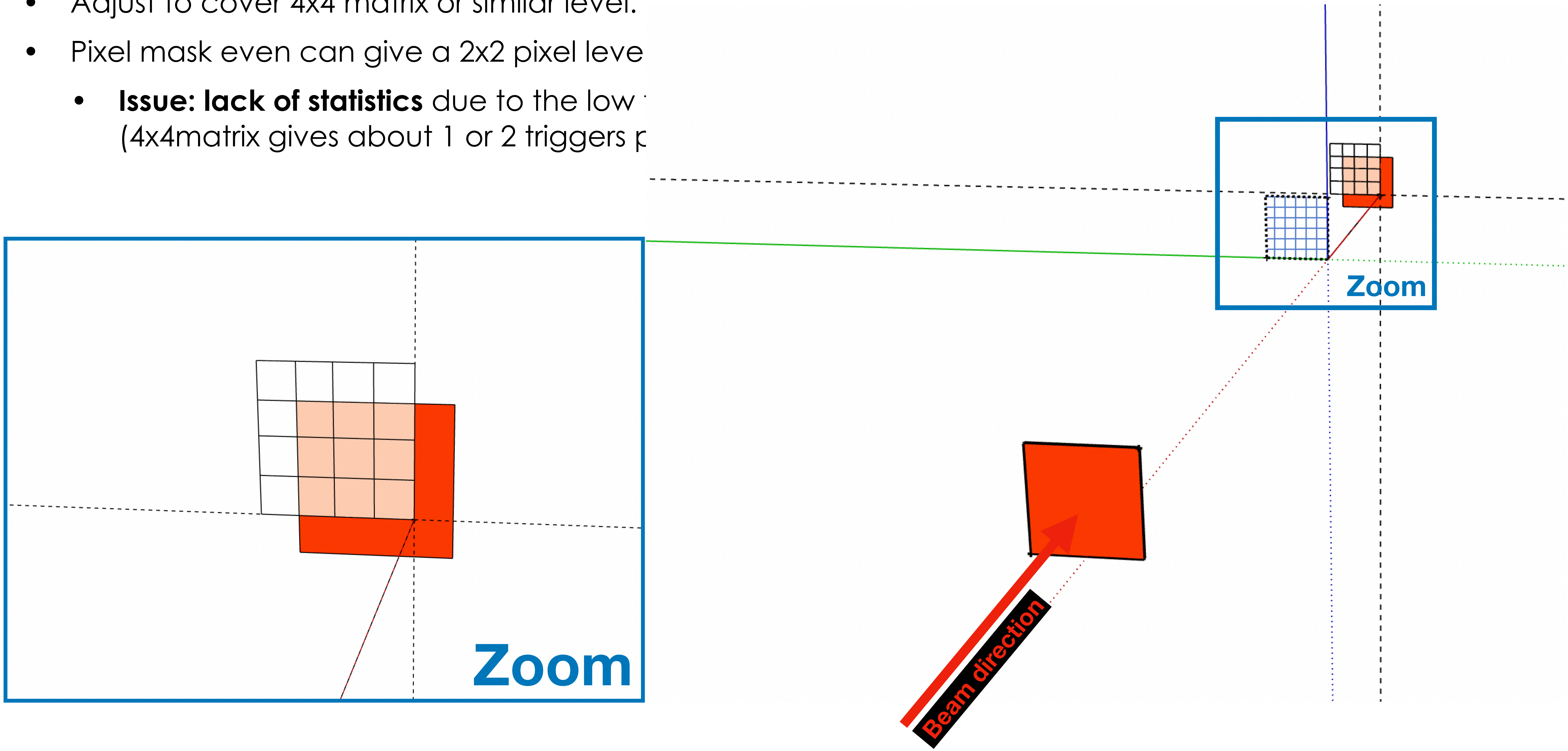
Note that all the sizes and distances are exaggerated ! !



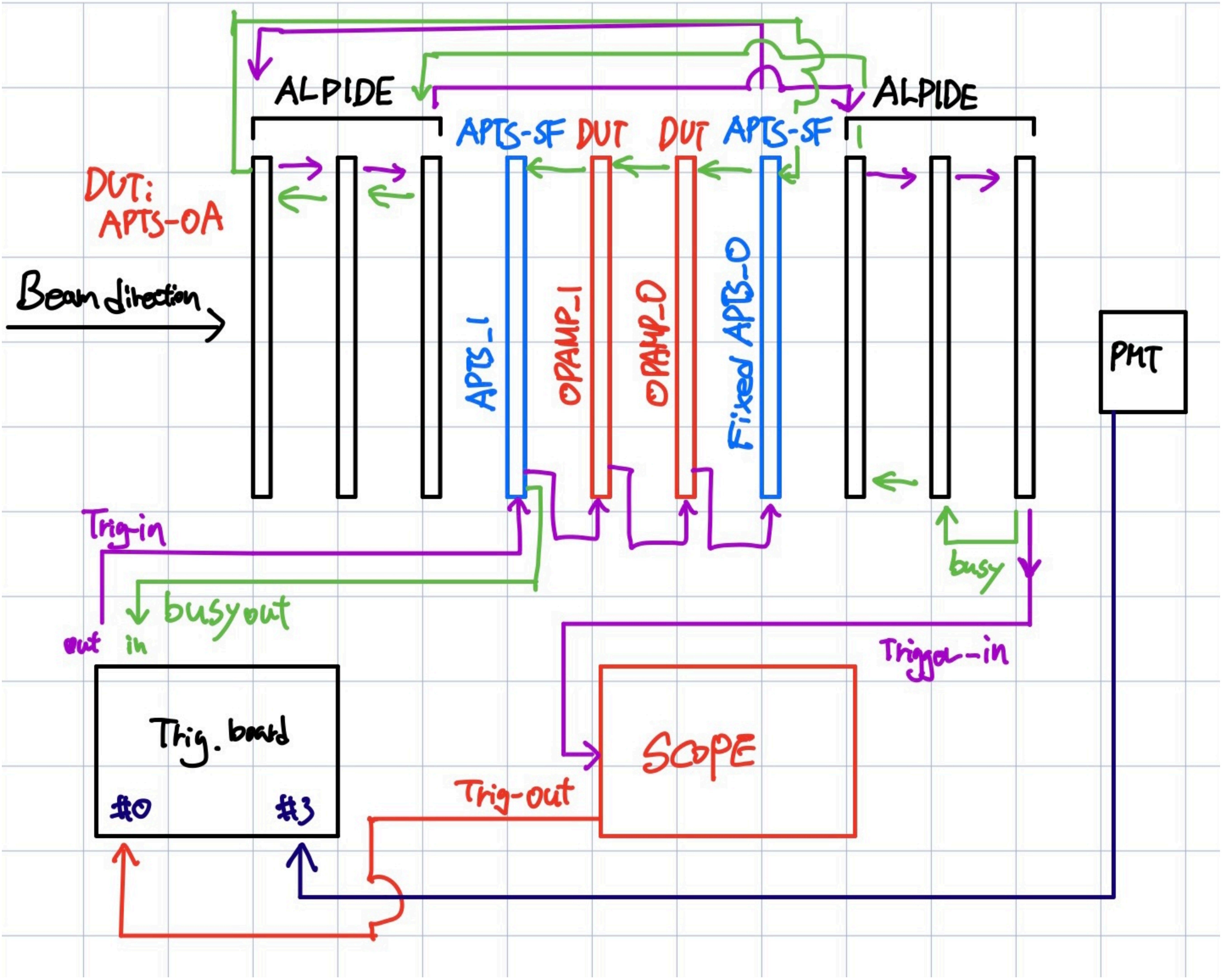
Detailed alignment procedure

#4 Fine tuning using APTS hitmap

- It is also be **possible(?)** to have a **pixel level alignment based on APTS hit map.**
  - Adjust to cover 4x4 matrix or similar level.
  - Pixel mask even can give a 2x2 pixel leve
    - **Issue: lack of statistics** due to the low
      (4x4matrix gives about 1 or 2 triggers p



Trigger/Busy connection scheme



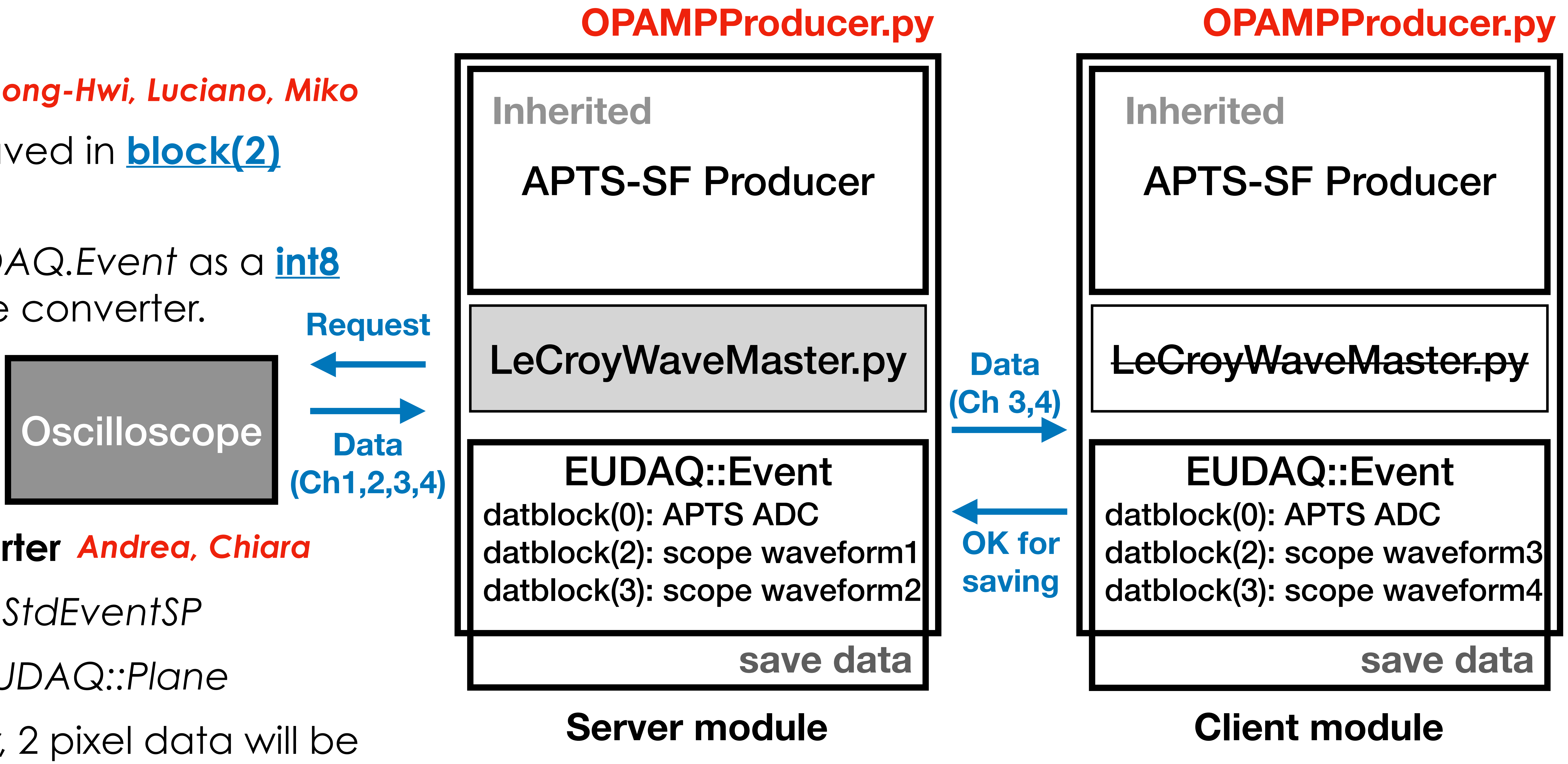
Connection scheme

- Trigger starts from **APTS_1**
  - Self-trigger (*APTS trigger*)
  - External trigger: Oscilloscope
- **Busy** came from the **last ALPIDE**
  - No busy connection from the scope

- **Adopted for the alignment and data taking**
  - No needs for additional modification.
  - Eg. OPAMP_0 trigger for alignment.
    - Run APTS_1 and OPAMP_1 as **standalone data taking** with the highest threshold **to prevent any busy signal**.
    - Taking the data using EUDAQ only including OPAMP_0 and APTS_0 and ALPIDEs.

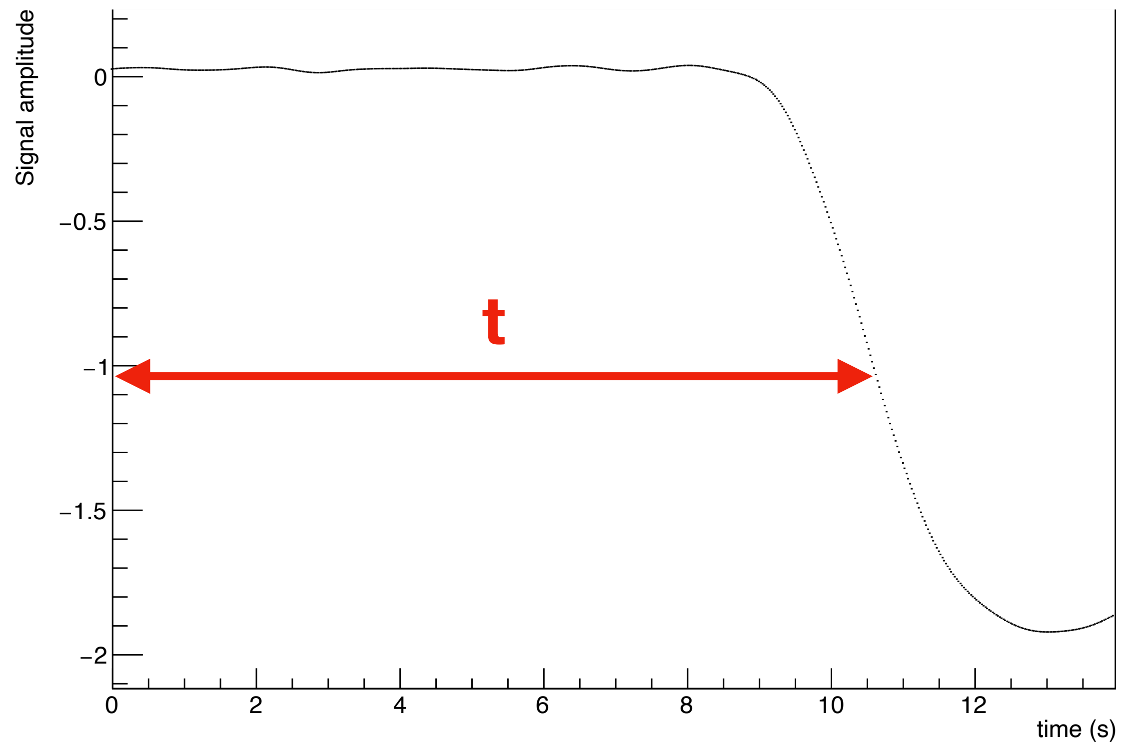
Data taking and Converting

- **OPAMP Producer: READY** *Andrea, Bong-Hwi, Luciano, Miko*
  - Waveforms from the scope is saved in **block(2)** and **block(3)**
  - Waveforms will be stored in *EUDAQ.Event* as a **int8 array** and will be handled in the converter.
- **OPAMPDump.py**
  - Basically **READY TO USE** (first QA of the scope data)
- **OPAMPRawEvent2StdEvent converter** *Andrea, Chiara*
  - *EUDAQ::Event* (raw) to *EUDAQ::StdEventSP*
    - Converted data saved as *EUDAQ::Plane*
  - Based on the APTS-SF converter, 2 pixel data will be overwrited based on the scope information.

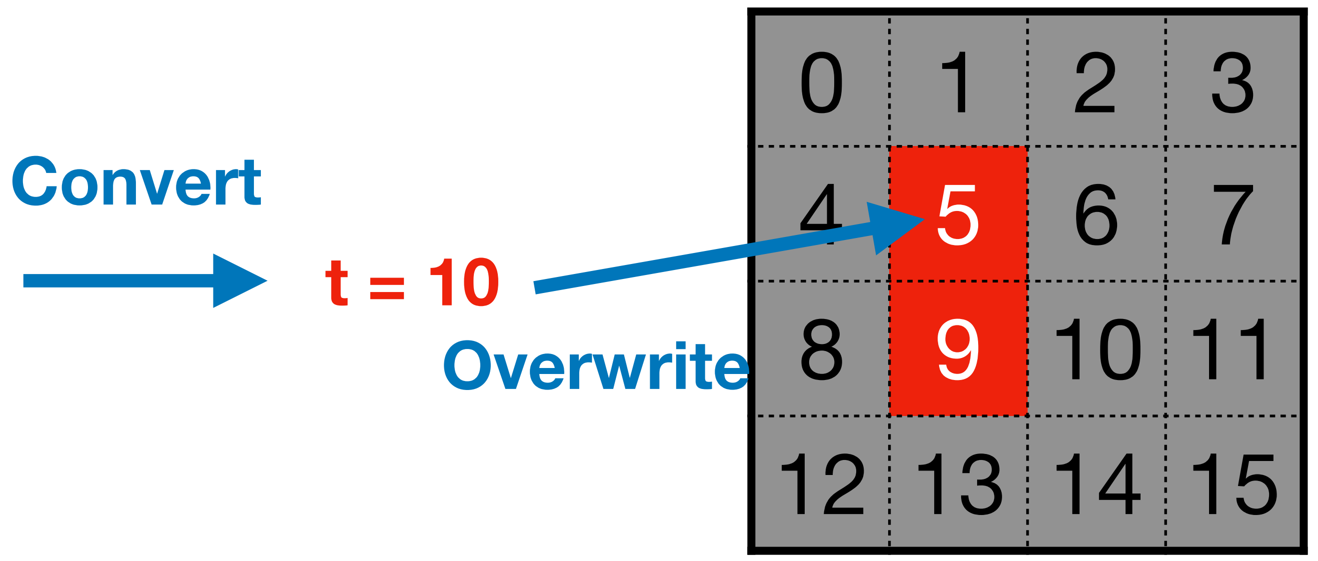



```
array([67, 49, 58, 87, 70, 32, 68,
65, 84, 49, 44, 35, 57, 48, 48, 48,
48, 48, 48, 56, 48, 50, 80, 78, 77,
77, 77, 77, 79, 78, 82, 81, 81, 81,
81, 81, 79, 80, 79, 80, 80, 82, 78,
80, 79, 79, 82, 81, 80, 80, 80, 80,
79, 77, 79, 80, 82, ... ],
dtype=np.float32)
```

Expected waveform (array)



Expected waveform (graph)



EUDAQ::StandardPlane

Studying the unscramble function in APTS-SF raw event converter

MLR1 DAQ Board data

UNIVERSITA DEGLI STUDI DI TORINO

ALICE

Process to understand

- **APTS "Data"**
  - Coming from 16 ch output from DAQ board
  - Composed with **40** block of uint8_t (**8 bit**) numbers

8

40

data[0]	0	0	0	0	0	0	0	0
data[1]	0	0	0	0	0	0	0	0
data[2]	0	0	0	0	0	0	0	0
...								
data[39]	0	0	0	0	0	0	0	0

MLR1 DAQ Board data

Data taking from DAQ board #1

- When taking the data, output from DAQ boards are transferred in a raw..

data[0]	0	0	0	0	0	0	0	0
data[1]	0	0	0	0	0	0	0	0
data[2]	0	0	0	0	0	0	0	0
...								
data[39]	0	0	0	0	0	0	0	0

MLR1 DAQ Board data

Data taking from DAQ board #2

- They are saved in a row from the lowest bit(0) to the higher bit(7)

MLR1 DAQ board

data flow

last data

1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 1

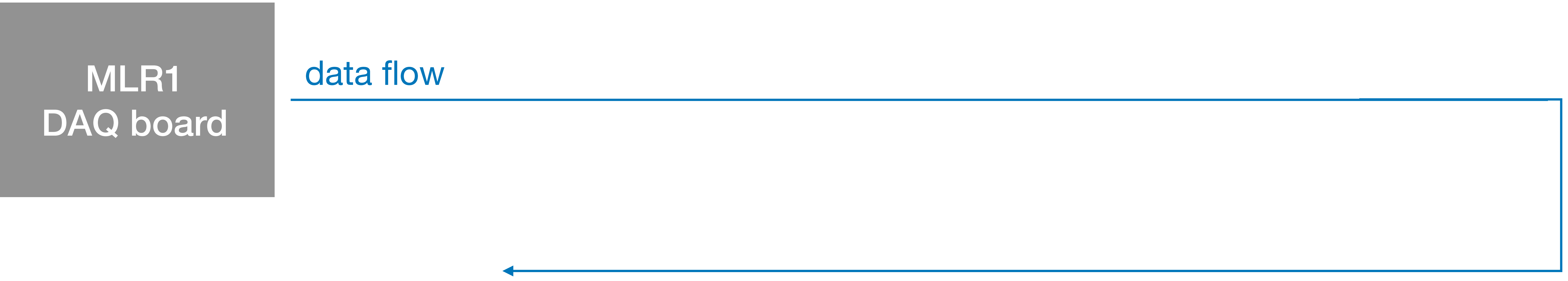
data[0]	0	0	0	0	1	0	1	0
data[1]	0	0	0	0	0	0	0	0
data[2]	0	0	0	0	0	0	0	0
...								
data[39]	0	0	0	0	0	0	0	0

first data

MLR1 DAQ Board data

Data taking from DAQ board #3

- In the end, 2 row of 'data' is filled and it continues until it fills data[39]



data[0]	1	0	0	1	1	0	1	0	first data
data[1]	1 last data	0	1	1	0	1	0	1	
data[2]	0	0	0	0	0	0	0	0	
...									
data[39]	0	0	0	0	0	0	0	0	

MLR1 DAQ Board data

Decoding data #1

- When decoding data, it is divided into a unit of 40 items.
  - Let's focus on the first block.

First block

data[0]	1	0	0	1	1	0	1	0
data[1]	1	0	1	1	0	1	0	1
data[2]	1	0	1	0	1	0	0	1
...								
data[39]	0	0	0	0	0	0	0	0

second block

data[0]	1	0	0	1	1	0	1	0
data[1]	1	0	1	1	0	1	0	1
data[2]	1	0	1	0	1	0	0	1
...								
data[39]	0	0	0	0	0	0	0	0

MLR1 DAQ Board data

Decoding data #2

- In **`unscramble`** function in the **`APTSRawEvent2StdEventConverter`**,
  - it tries to decode the data by dividing it into 2 sectors (data[2*ib] and data[2*ib+1])

data[0]	1	0	0	1	1	0	1	0
data[2]	1	0	1	1	0	1	0	1
data[4]	1	0	1	0	1	0	0	1
...								
data[30]	0	0	0	0	0	0	0	0

data[1]	1	0	0	1	1	0	1	0
data[3]	1	0	1	1	0	1	0	1
data[5]	1	0	1	0	1	0	0	1
...								
data[31]	0	0	0	0	0	0	0	0

data[2*ib] for iw < 8
data[2*ib + 1] for iw >= 8

```
for(int iw=0;iw!=16;++iw) {
      out[chmap[iw]]=0;
      for(int ib=0;ib!=16;++ib)
        out[chmap[iw]]|=(in[ib<<1|((iw>>3)&1)]>>(iw&0x7)&1)<<(15-ib);
    }
```

MLR1 DAQ Board data

Decoding data #3

- In **`unscramble`** function in the **`APTSRawEvent2StdEventConverter`**,
  - iw stands for each channel, ib stands for each bit

	iw: 7							iw: 0
data[0]	1	0	0	1	1	0	1	0
data[2]	1	0	1	1	0	1	0	1
data[4]	1	0	1	0	1	0	0	1
...								
data[30]	0	0	0	0	0	0	0	0

	iw: 15				iw: 11			
data[1]	1	0	0	1	1	0	1	0
data[3]	1	0	1	1	0	1	0	1
data[5]	1	0	1	0	1	0	0	1
...								
data[31]	0	0	0	0	0	0	0	0

Select 'iw'th item

```
for(int iw=0;iw!=16;++iw) {
      out[chmap[iw]]=0;
      for(int ib=0;ib!=16;++ib)
        out[chmap[iw]]|=(in[ib<<1|((iw>>3)&1)]>>(iw&0x7)&1)<<(15-ib);
    }
```

MLR1 DAQ Board data

UNIVERSITA DEGLI STUDI DI TORINO

ALICE

Decoding data #4

- In **`unscramble`** function in the **`APTSRawEvent2StdEventConverter`**,
  - Each bits are saved to uint16_t out[] from first to last(16th) bit in order

iw: 7 … iw: 0 (total 16 bits)

data[0]	1	0	0	1	1	0	1	0
data[2]	1	0	1	1	0	1	0	1
data[4]	1	0	1	0	1	0	0	1
...								
data[30]	0	0	0	0	0	0	0	0

iw: 15 … iw: 11

data[1]	1	0	0	1	1	0	1	0
data[3]	1	0	1	1	0	1	0	1
data[5]	1	0	1	0	1	0	0	1
...								
data[31]	0	0	0	0	0	0	0	0

Save 'iw'th bit into uint16_t value from top to bottom

```
for(int iw=0;iw!=16;++iw) {
      out[chmap[iw]]=0;
      for(int ib=0;ib!=16;++ib)
        out[chmap[iw]]|=(in[ib<<1|((iw>>3)&1)]>>(iw&0x7)&1)<<(15-ib);
    }
```

MLR1 DAQ Board data

Decoding data #5

- In **`unscramble`** function in the **`APTSRawEvent2StdEventConverter`**,
  - Collect all bits in uint16_t out[] array

```
for(int iw=0;iw!=16;++iw) {
      out[chmap[iw]]=0;
      for(int ib=0;ib!=16;++ib)
        out[chmap[iw]]|=(in[ib<<1|((iw>>3)&1)]>>(iw&0x7)&1)<<(15-ib);
   }
```

iw: 0	0	1	1	...	0
iw: 1	1	0	0	...	0
...					
iw: 11	1	0	1	...	0
...					
iw: 16	1	1	1	...	0

unit16_t array

iw: 0

data[0]	1	0	0	1	1	0	1	0
data[2]	1	0	1	1	0	1	0	1
data[4]	1	0	1	0	1	0	0	1
...								
data[30]	0	0	0	0	0	0	0	0

total 16 bits

iw: 11

data[1]	1	0	0	1	1	0	1	0
data[3]	1	0	1	1	0	1	0	1
data[5]	1	0	1	0	1	0	0	1
...								
data[31]	0	0	0	0	0	0	0	0

MLR1 DAQ Board data

UNIVERSITA DEGLI STUDI DI TORINO
ALICE

Decoding data #6

- In **`unscramble`** function in the **`APTSRawEvent2StdEventConverter`**,
  - Re-order it using the channel map (chmap)

iw: 0	0	1	1	...	0
iw: 1	1	0	0	...	0
...					
iw: 11	1	0	1	...	0
...					
iw: 16	1	1	1	...	0

ch0	1	0	1	...	0
ch1	1	0	0	...	0
ch2	0	1	1	...	0
...					
ch15	1	0	1	...	0

unit16_t adcs[16]

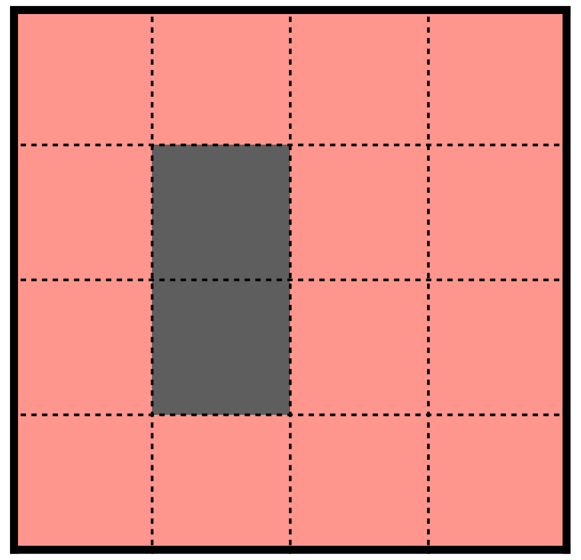
```
const int chmap[]={2,1,5,0,4,8,9,12,13,14,10,15,11,7,6,3};
```

Data analysis

Measurements and Datasets

Run number	# of events	Remarks
run264020927_220630021003.raw	9643	*APTS trigger*
run264214855_220630214931.raw	13605	*APTS trigger*
run265113818_220701113854.raw	23072	*APTS trigger*
run266124048_220702124125.raw	2506	*APTS trigger*
run266143343_220702143419.raw	22056	*APTS trigger*
run267142125_220703142202.raw	11893	*APTS trigger*
run271223026_220704223103.raw	535	**OPAMP pixel trigger (scope)**
run272093909_220705093946.raw	191	**OPAMP pixel trigger (scope)**
run272111314_220705111352.raw	273	**OPAMP pixel trigger (scope)** / modified APTS conf
run272151341_220705151418.raw	190	**OPAMP pixel trigger (scope)** / modified APTS conf
run272174811_220705174848.raw	1083	**OPAMP pixel trigger (scope)** / modified APTS conf
run273075645_220706075721.raw	12	**OPAMP pixel trigger (scope)** / modified APTS conf

- **# of events of the *APTS triggered* runs** are including all events that track passing the other pixels.
  - **OPAMP pixel trigger**: using the scope trigger connected to the 2 pixels on OPAMP.
    - Pros: concentrated data taking only for the **interesting region**.
- **modified APTS conf:** n_frames_before/after from 20/50 to 100/100
- Full run list and raw files: https://cernbox.cern.ch/index.php/apps/files/?dir=/__myprojects/aliceits3/ITS3-WP3/Testbeams/2022-06_SPS&



APTS trigger
vs OPAMP pixel trigger

Analysis Overview

Applied in this analysis

Data Conversion → Data Extraction → Tree Analysis → Final plots

Data Conversion

- **.Raw to .ROOT**
  - Save only 4 waveforms from the scope
- **Use *ROOT::TGraph***
  - Easy to handle and fit

Data Extraction

- **.ROOT to .ROOT**
  - Extract value from fit function
  - Save as *ROOT::TTree*
- **Basic Selections**
  - Coincidence
  - Signal size
- **Extract time parameters**
  - CFD method
  - Fitting method
- **Draw QA plots**

Tree Analysis

- **From *ROOT::TTree* to *pandas::dataframe***
  - Easy to handle the datasets
- **Draw QA histograms**
  - All extracted values with run/channel
- **Selections**
  - Based on QA plots
- **Draw the basic outputs**

Final plots

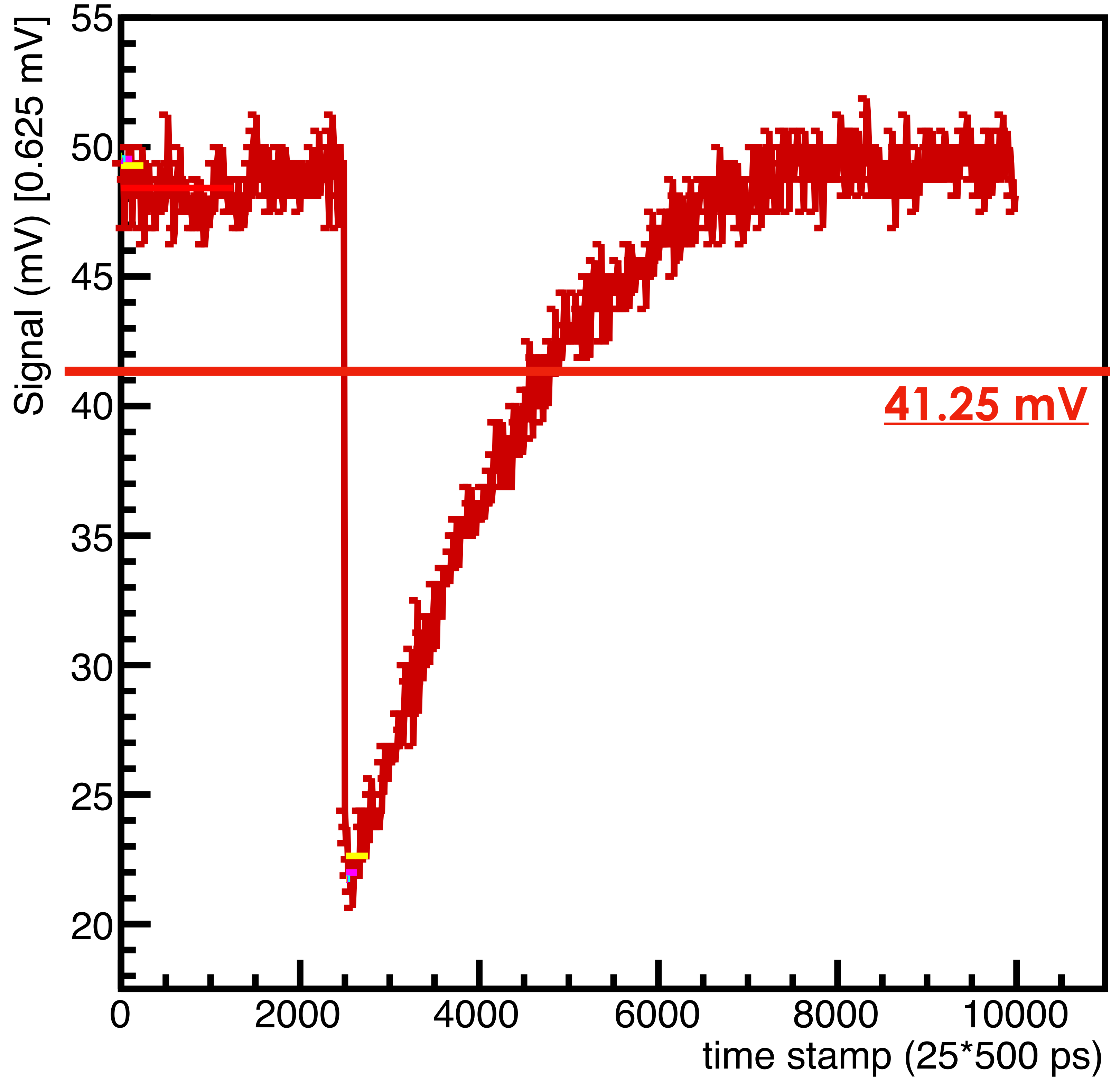
- **Time difference**
  - vs Amplitude fraction
  - vs different methods

- **Size of each waveform:** ~0.4 MB
  - Integer (8 bits, 1 Byte) with 40,000 points ~ 40 kB
- **Size of a event:** > 1.6 MB
- **Raw file size with 1000 events**: 1.6 GB

Covert from Raw to ROOT::TGraph

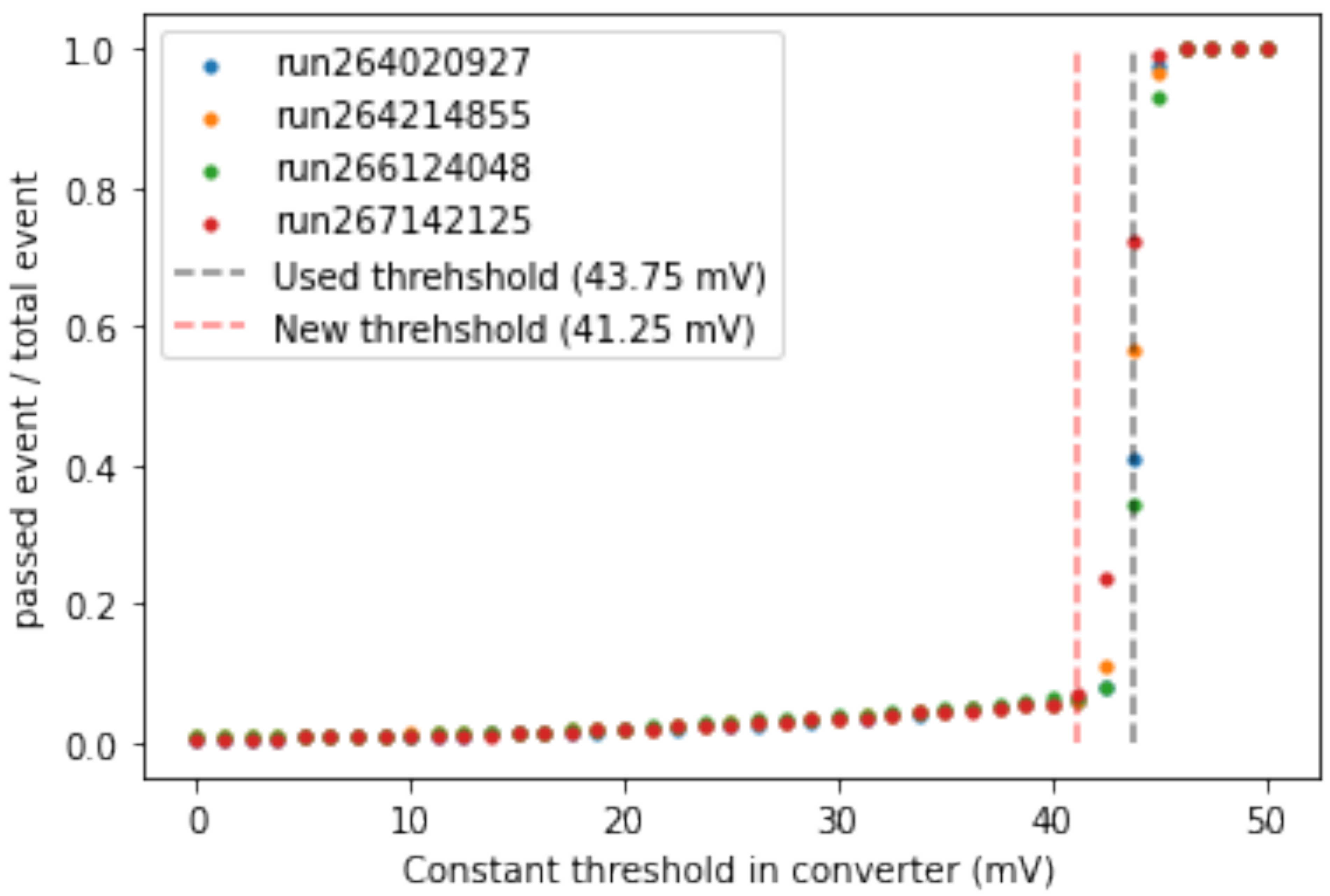
Data Conversion

- Import the ***binary*** and Convert the int8 array to ***ROOT::TGraph***
  - Constant filtering threshold applied: **41.25 mV**
    - To minimise computation.
    - Threshold was selected based on the study



Example of *ROOT::TGraph* with threshold

Passed event vs Constant threshold

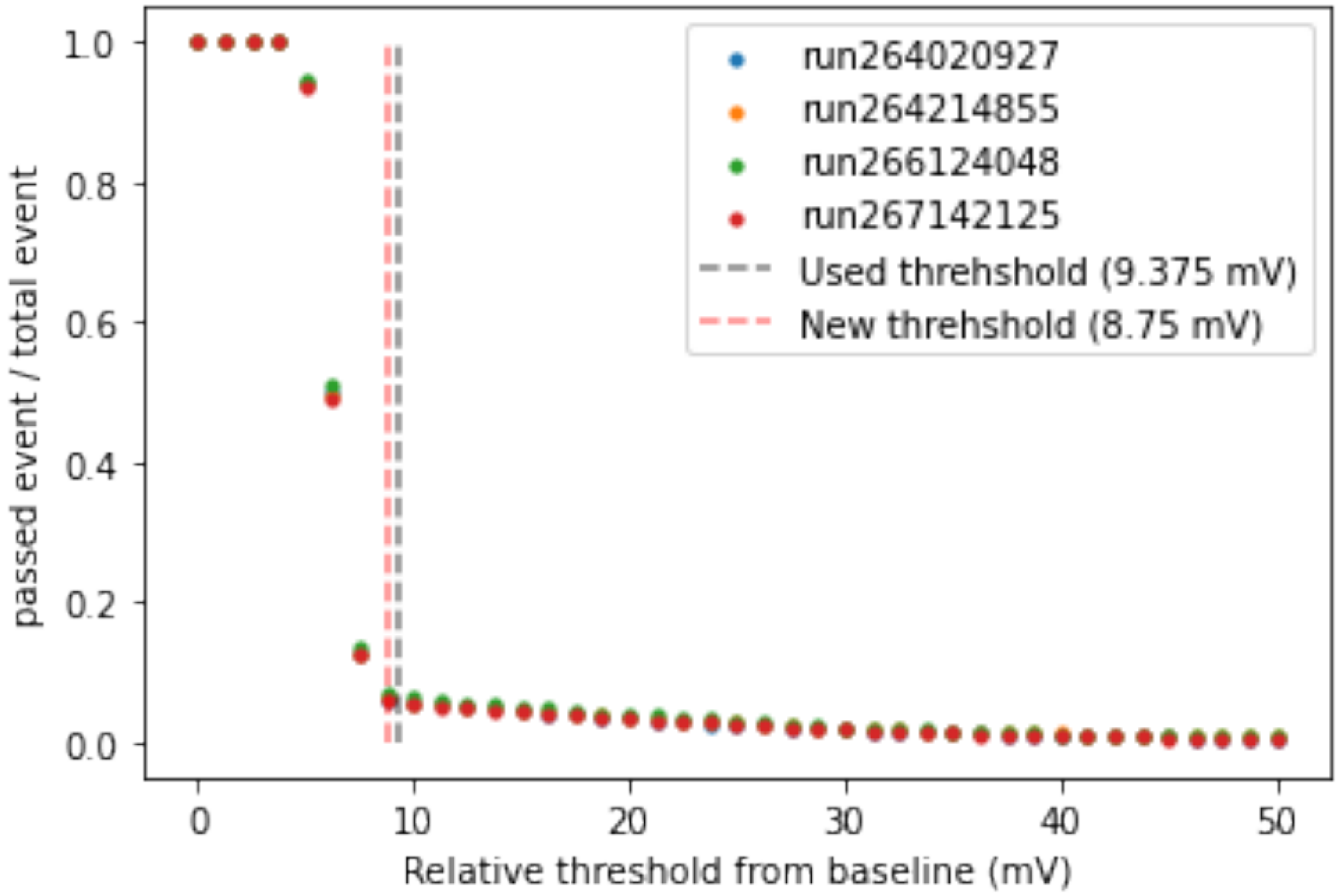


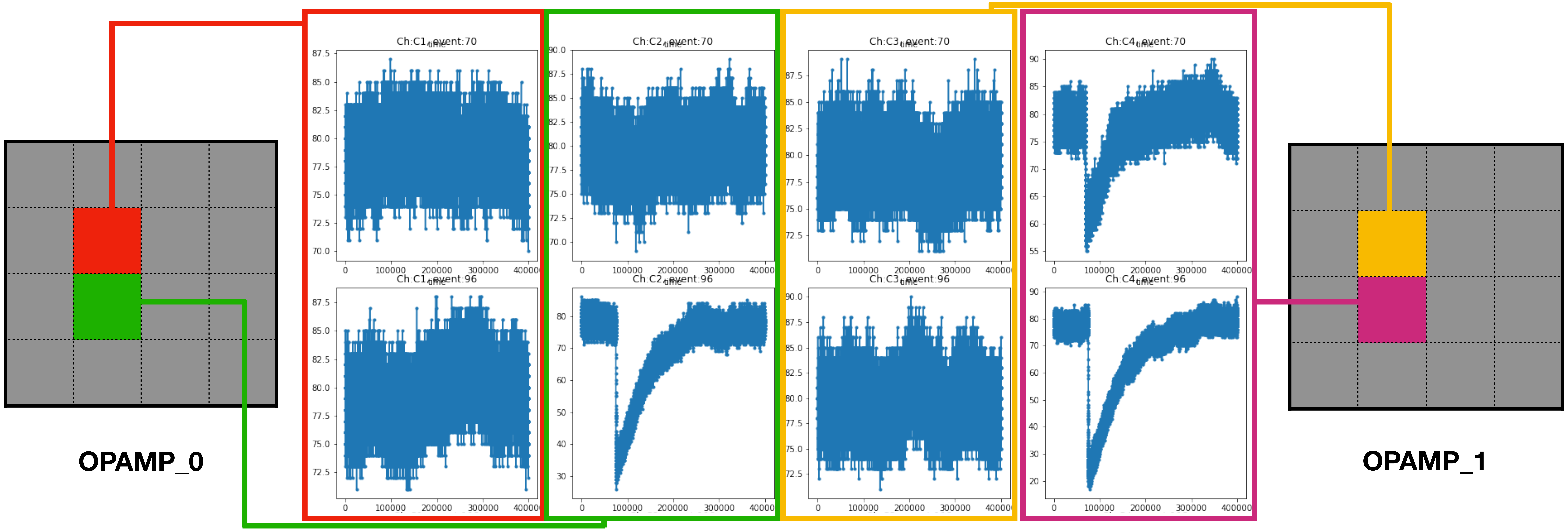
- **The fraction of passed event to total events**
  - vs constant thresholds
  - APTS triggered run: most events don't have signals in the pixels of interest.
  - Choose the first point in the stable trend.

Extraction#1 - Find Coincidence Events

Data Extraction

1. Import all ***ROOT::TGraph*** from the input file.
2. Apply the threshold of the signal
   - Calculated the signal size from the baseline.
   - Applied threshold: **8.75 mV** (Right figure)
3. Find the coincidence event
   - Simply check: (Ch1||Ch2)&&(Ch3||Ch4)



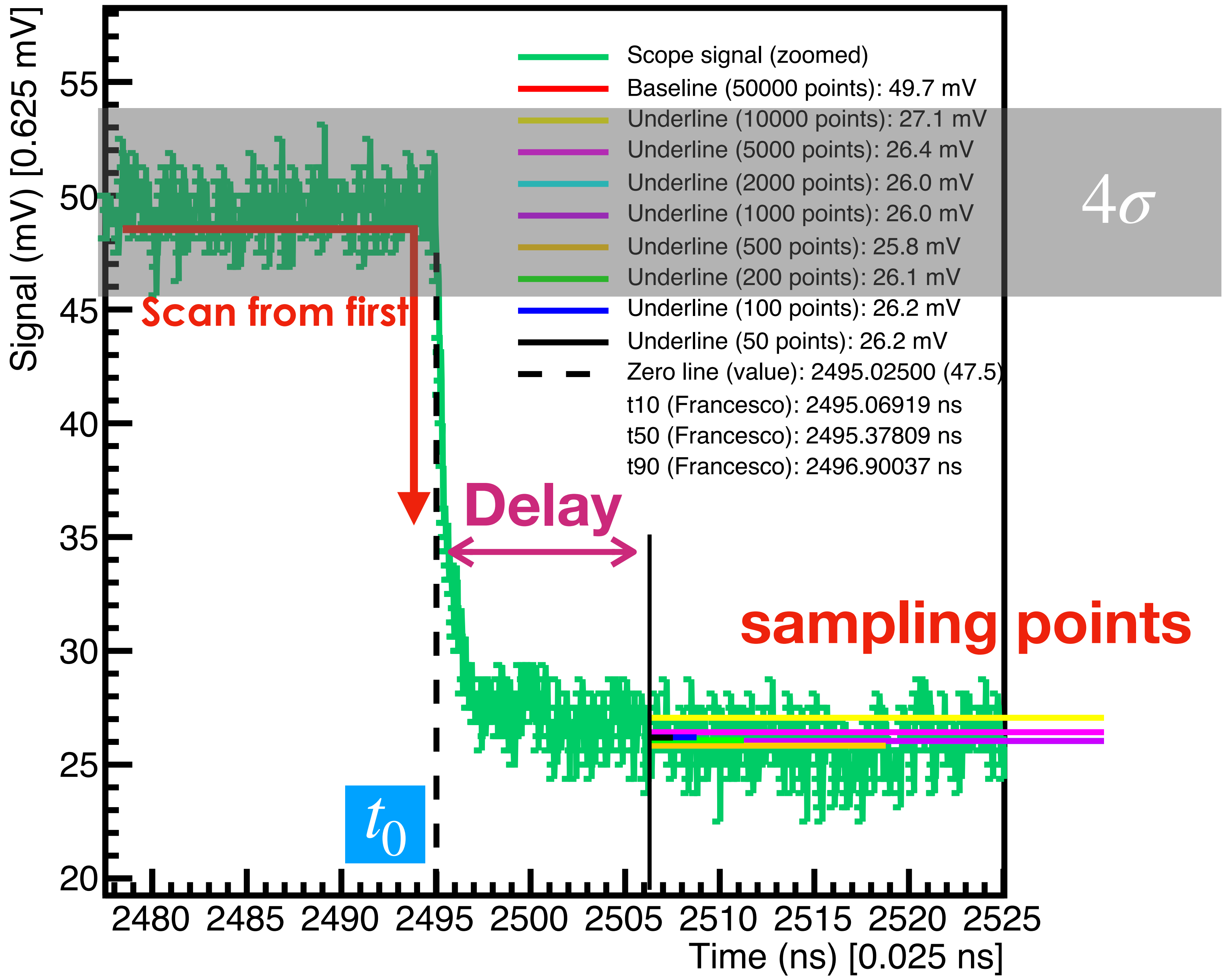


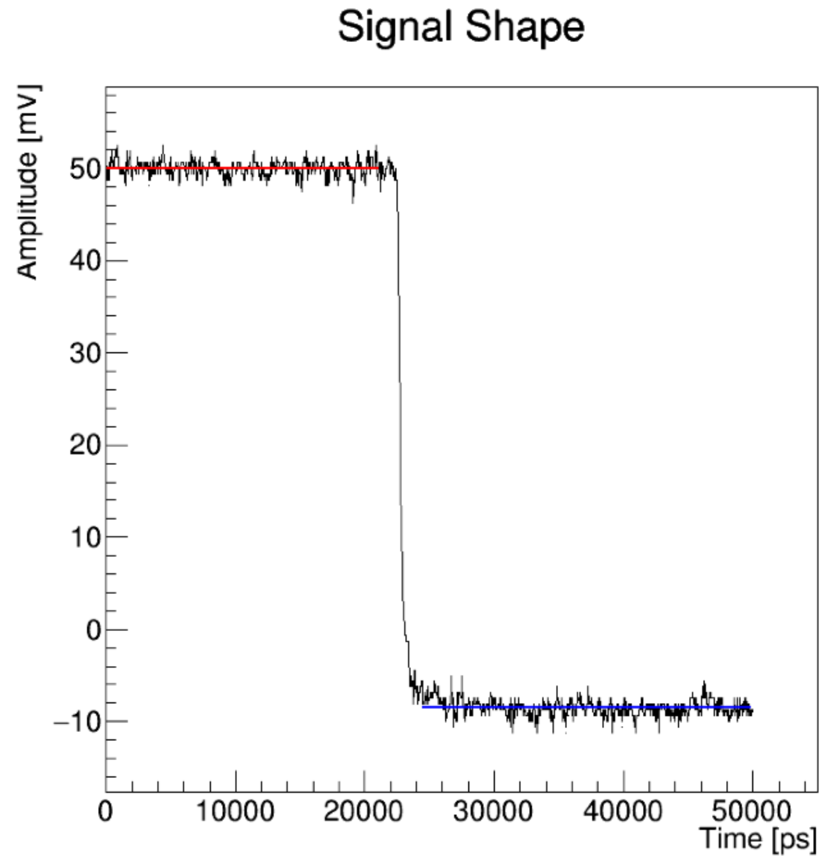
Waveforms (Top: no coincidence / Bottom: coincidence)

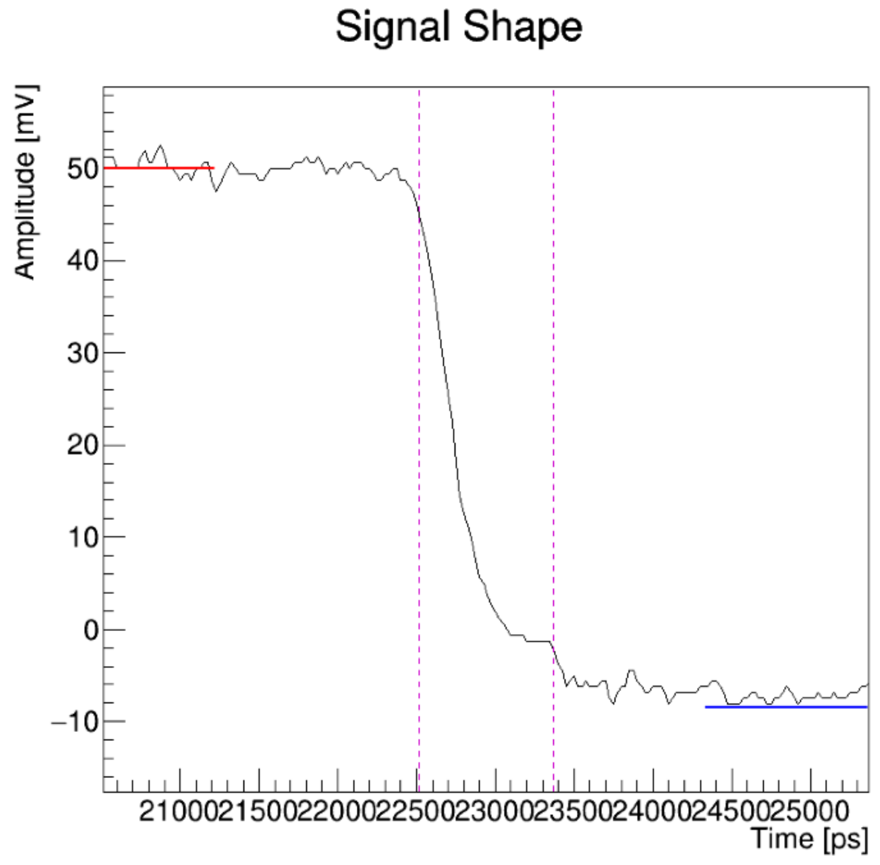
*Y axis calibration factor: 20*4/128 = 0.625 mV/point

Extraction#2: Determine t_0

Ch:2, event:272175495



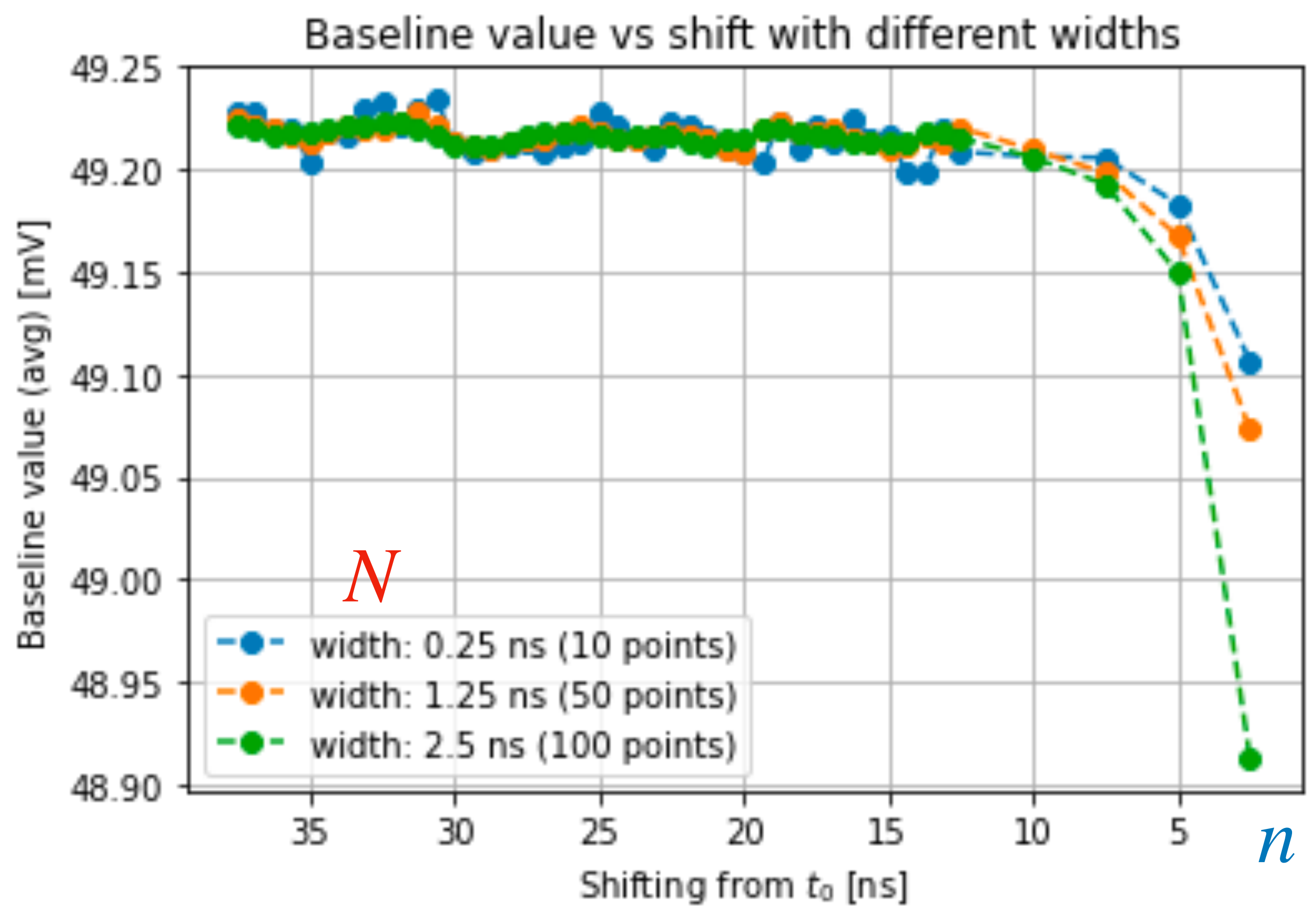




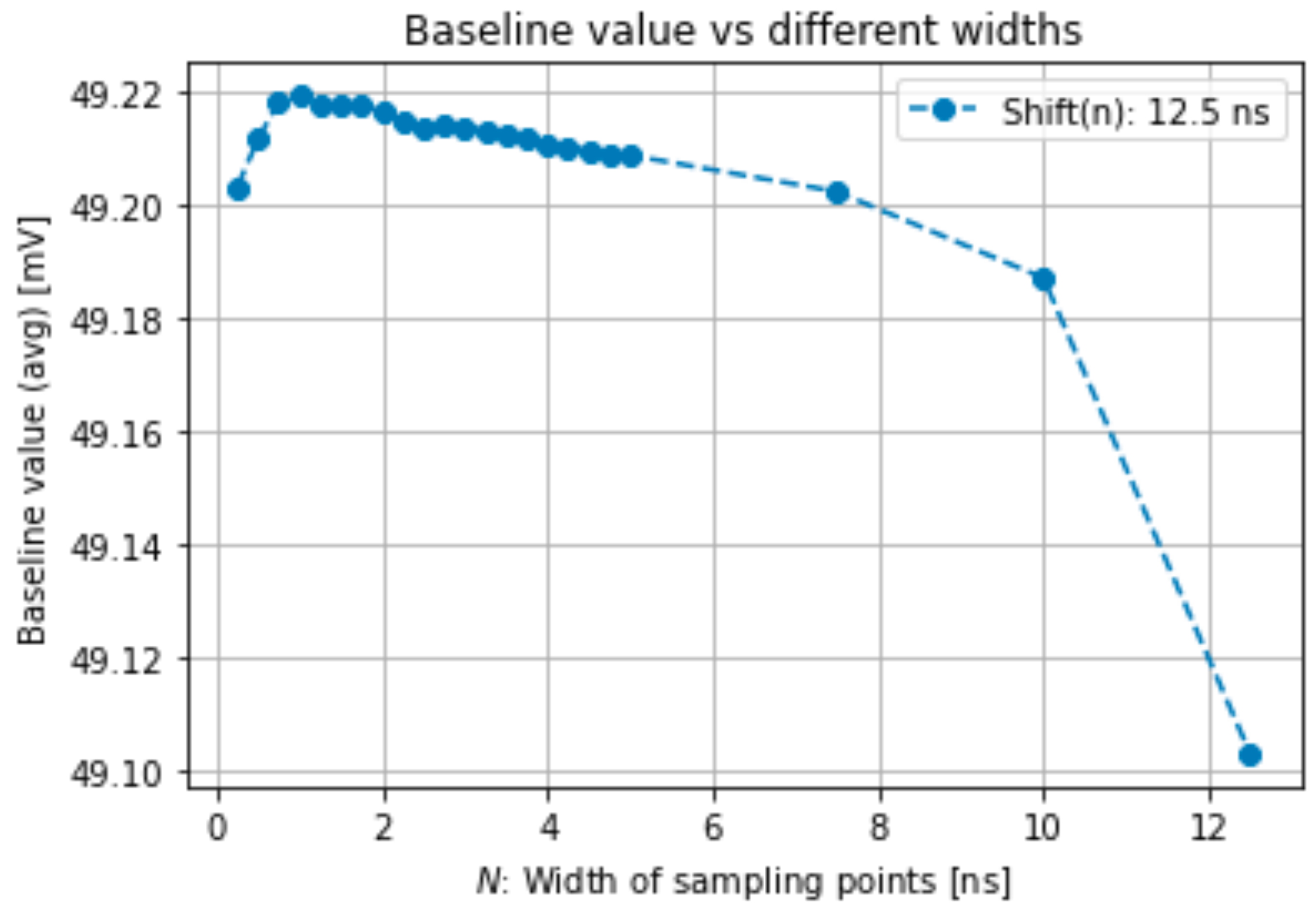
- **Determination of**
  - Important for the stabilisation of the underline value
  - Procedure
    - Compare the signal with the baseline, starting from the left.
    - Count the number of points **~4 sigmas (~4.3 mV)** smaller than the **baseline**.
      - Count **10 points in a row**
        250 ps
      - If there is any point within ~ 4 sigmas during the check, remove the current count stack and scan next point.
    - If the above test is passed, use the point placed **10 points** before as a t_0
      250 ps
      - t_0 is not exactly pointing the starting point of signal. but this value is used as a standard point to obtain a relative position eg. $t_0 + 5ns$
- **Underline**
  - Underline values are determined based on **the relative starting point from t_0**.
  - Similar to the determination of baseline
    - But the falling signal has a **tail** - need to put a **delay**
  - Dependency on the delay and **number of sampling points** studied.

Extraction#3 - Determine baseline

Data Extraction

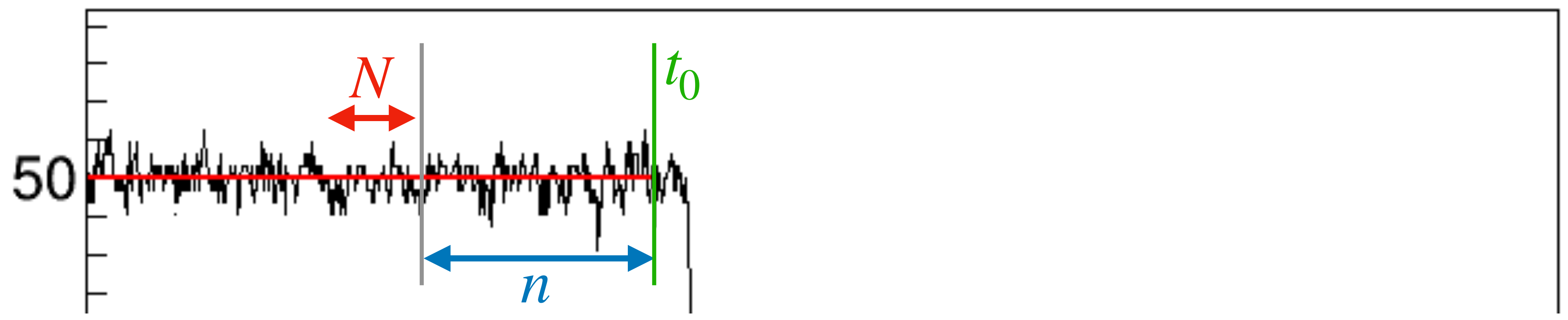


Baseline with different shift/sampling points



Baseline with different sampling points (*N*)

- **Baseline calculation:**
  - Average value of N sampling points(width) counting from n ns shifting to t_0



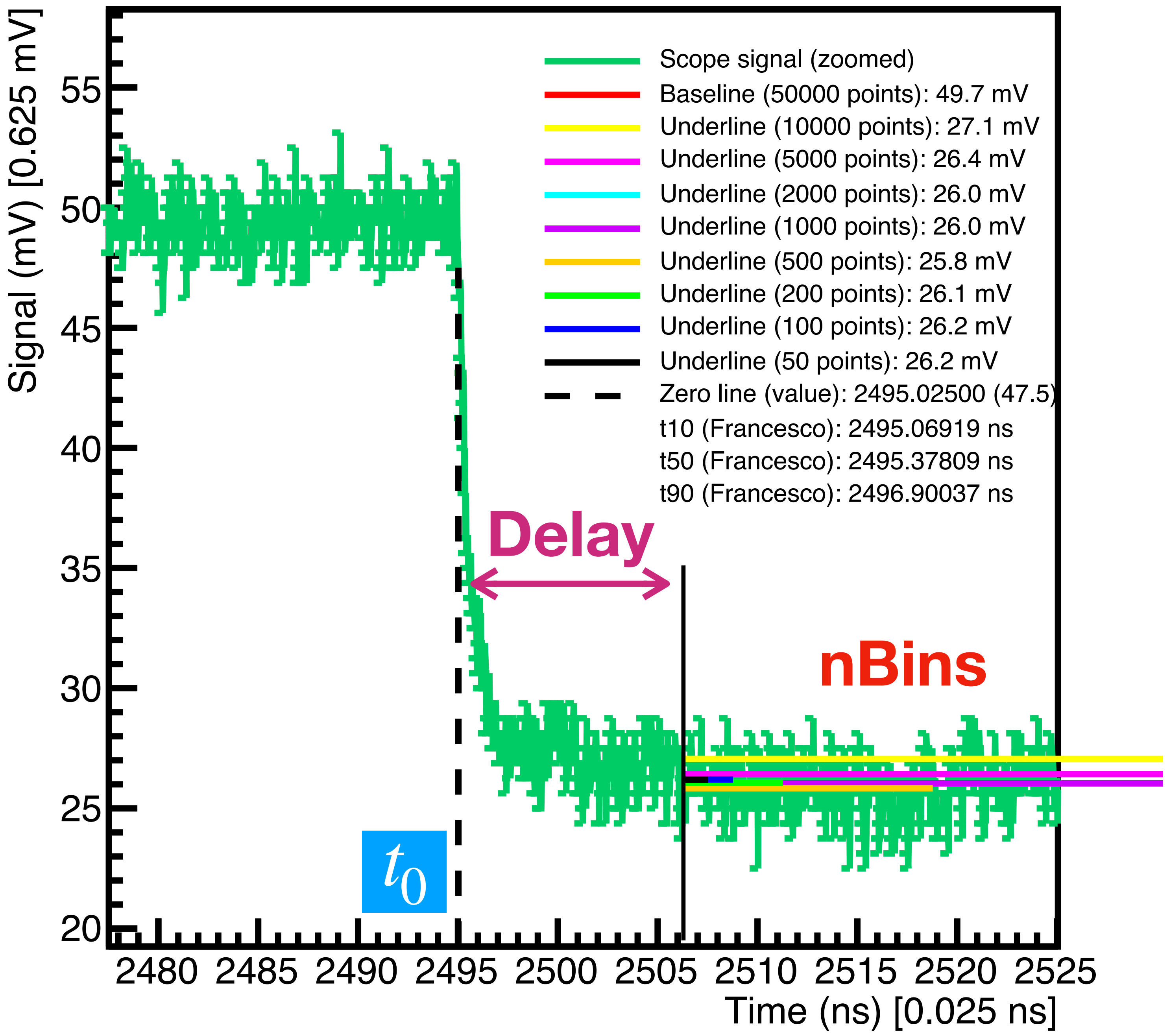
Baseline calculation example with t_0, n shifting, N sampling

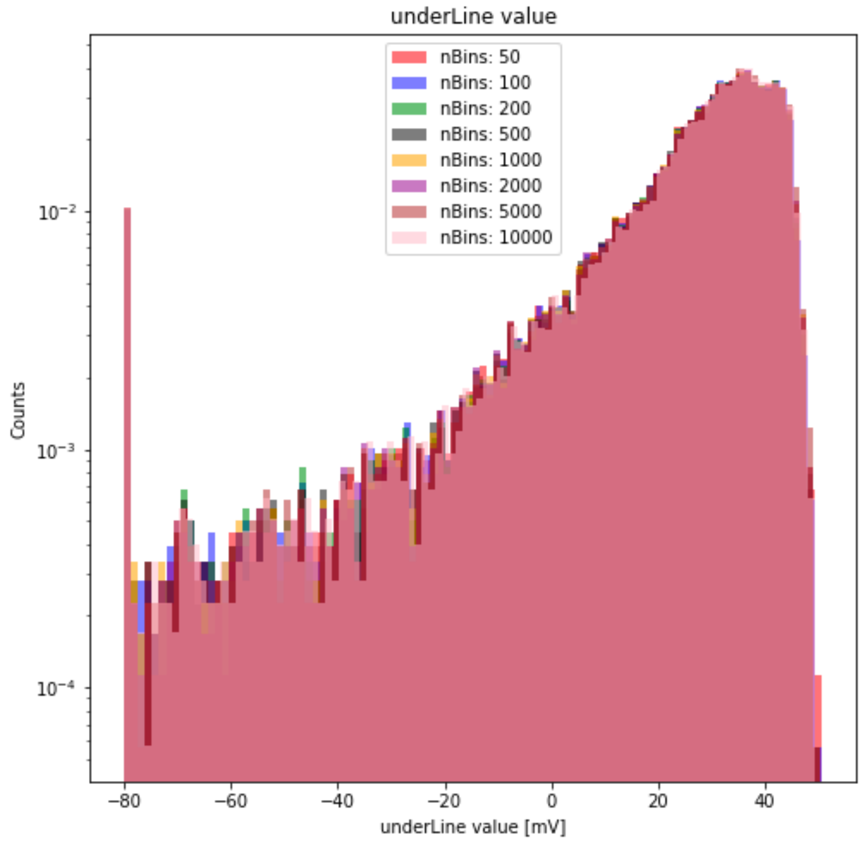
  - Average value converges around 49.215 mV after **12.5 ns shift**
    - Standard deviation of sampling points: **~1.6 mV**
  - Average baseline vs various sampling points(*N, width*)
    - The value has the **maximum at 1 ns width.**
      - Relatively stable until 5 ns
      - **Between 0.75 and 2 ns** will be suitable.

Extraction#4 - Determine underline

Data Extraction

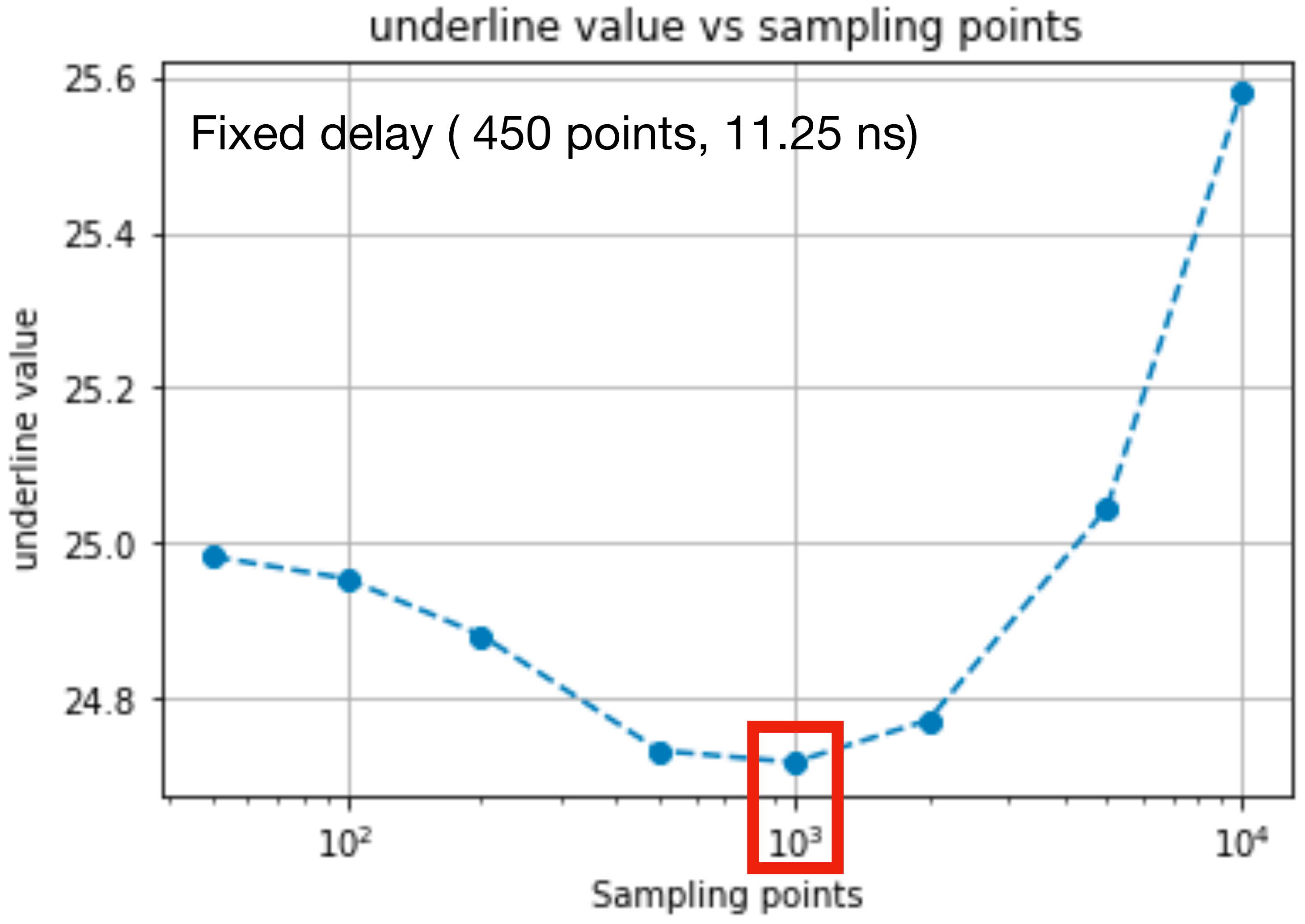
Ch:2, event:272175495



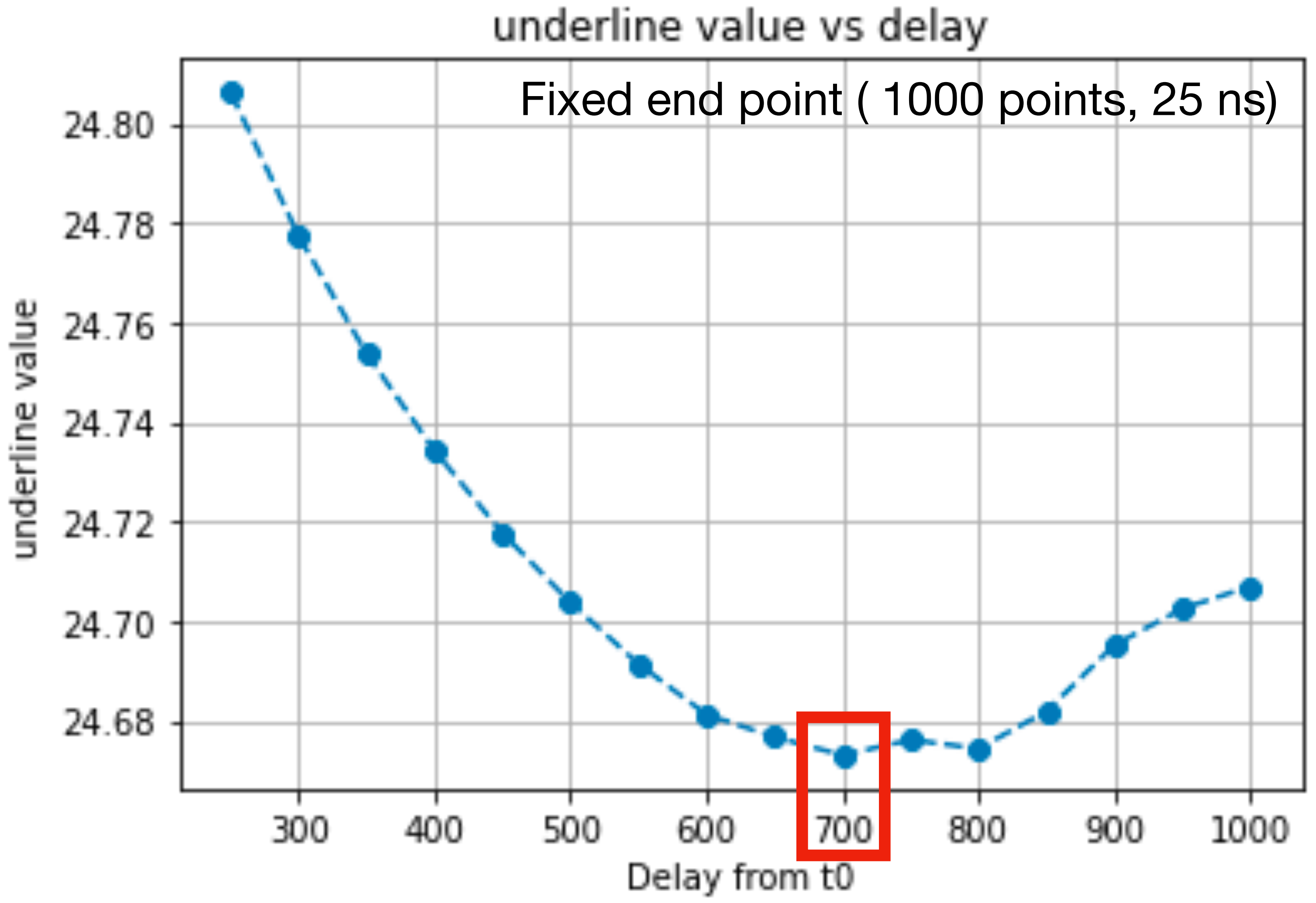


Histograms with nBins

- **Underline**
  - Underline values are determined based on **the relative starting point from t_0**.
  - Similar to the determination of baseline
    - But the falling signal has a **tail** - need to put a **delay**
  - Dependency on the delay and nBins studied (below 2 plots)



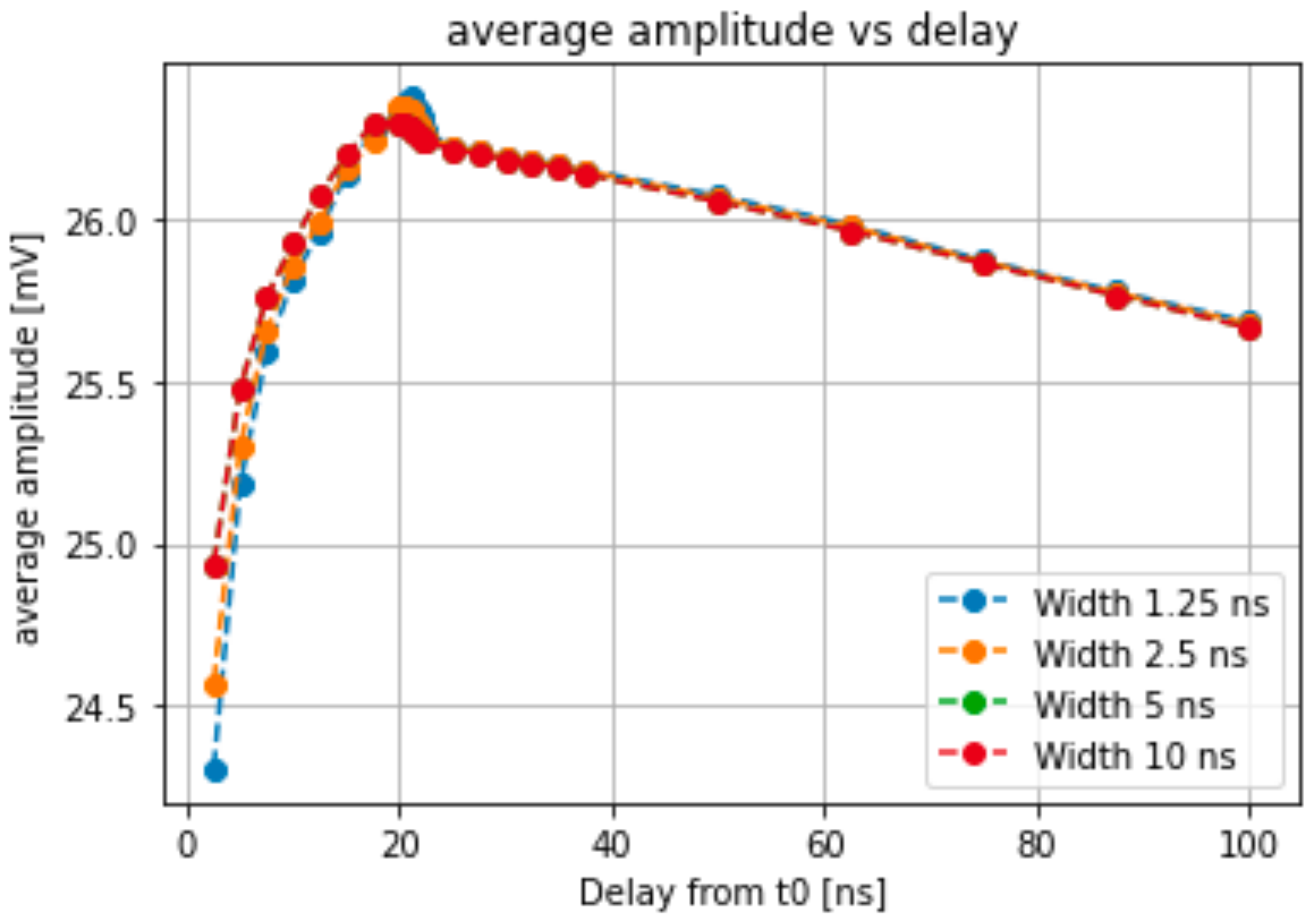
Mean vs N sampling point



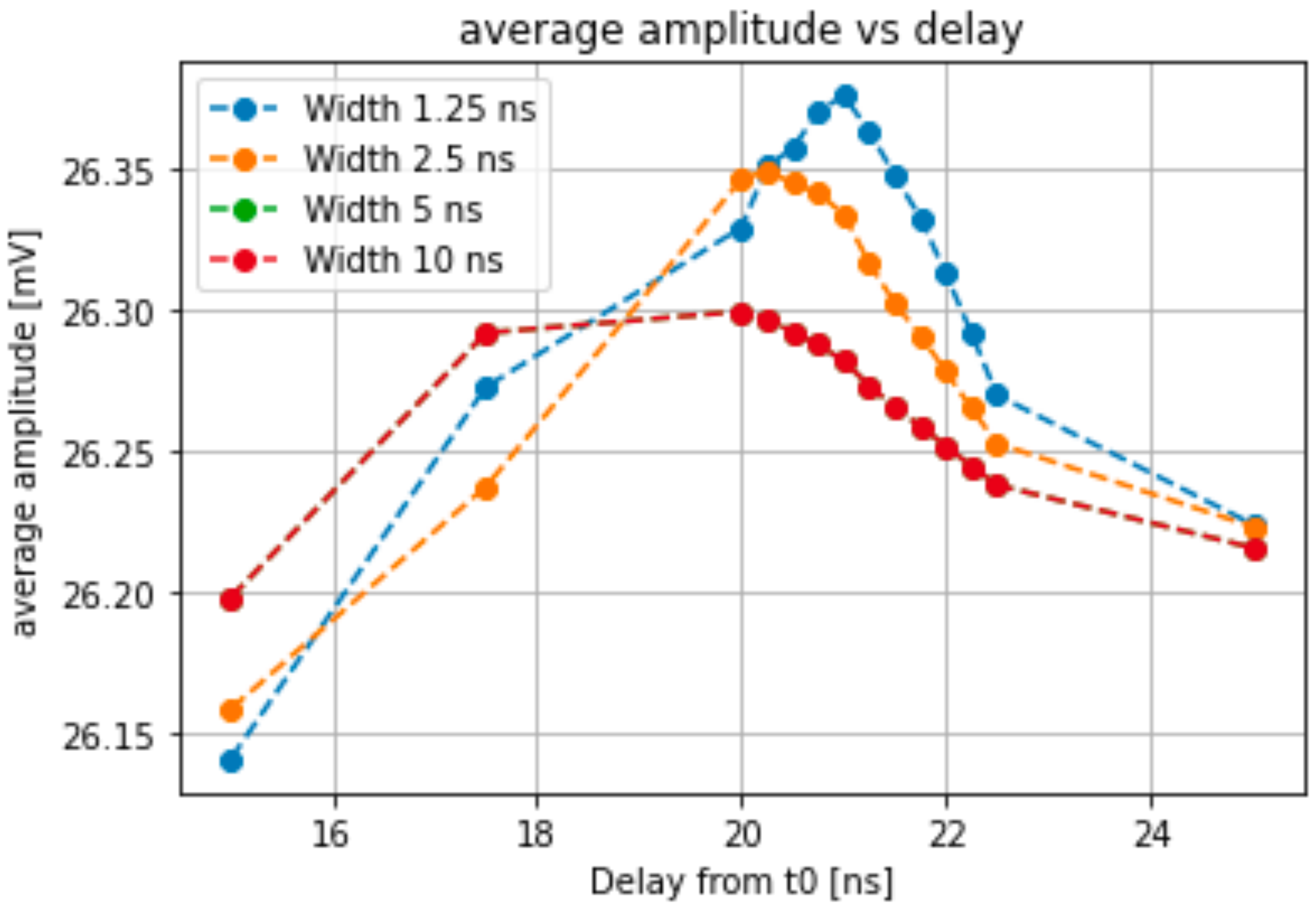
Mean vs delay

Average amplitudes with delay

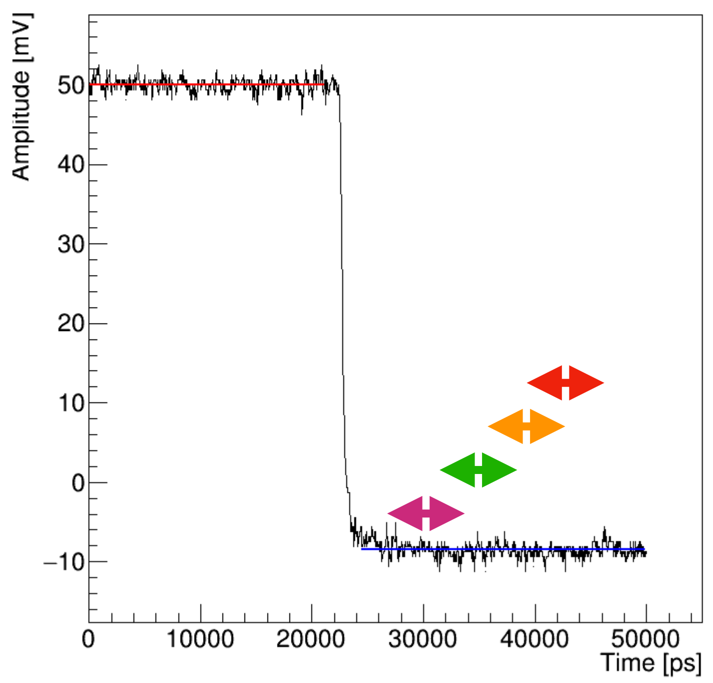
Baseline: 5 ns from t_0 + 5 ns sampling



Cumulated signal



Zoom up around 21 ns delay



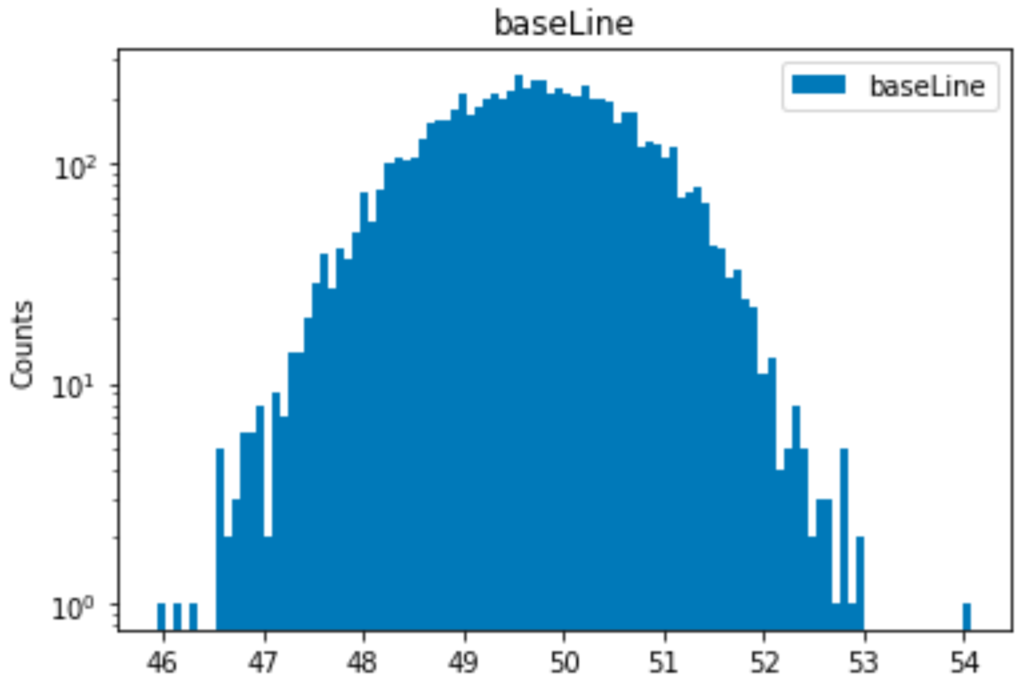
Delay variation of underline (fixed width as 1.25 ns)

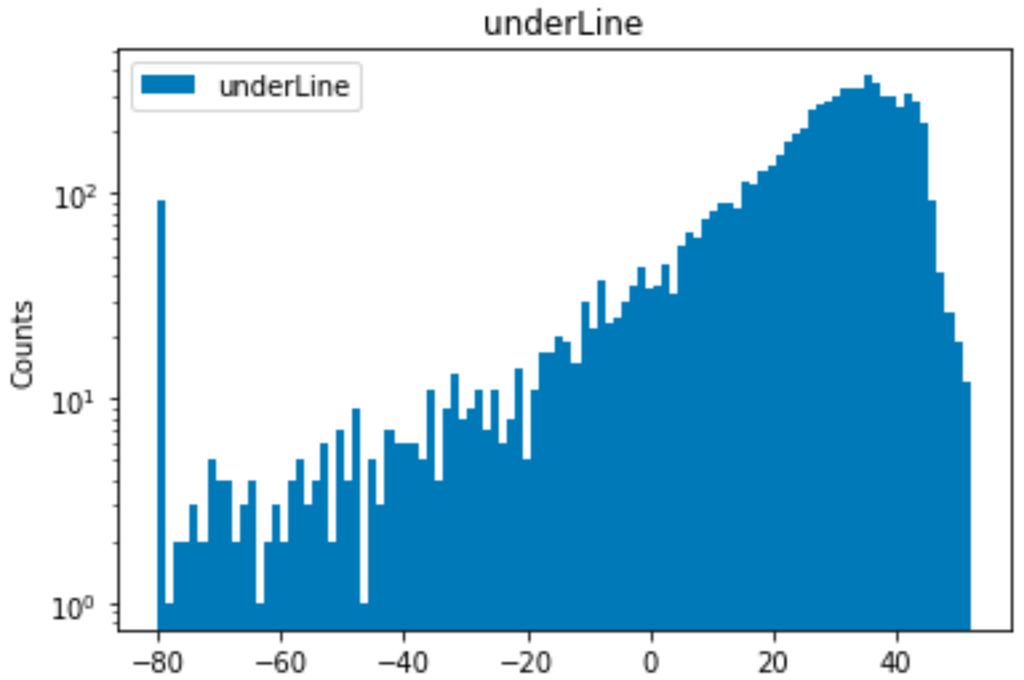
- **Maximum amplitude point scan**
  - The maximum amplitude can be obtained when the underline value is the minimum.
    - The signal decreases continuously even after rapid fall.
  - **The maximum point can be found around 21 ns delay (1.25 ns width)**
    - If we choose around 200 sampling points (10 ns), the between 17.5 to 20 ns will be stable.

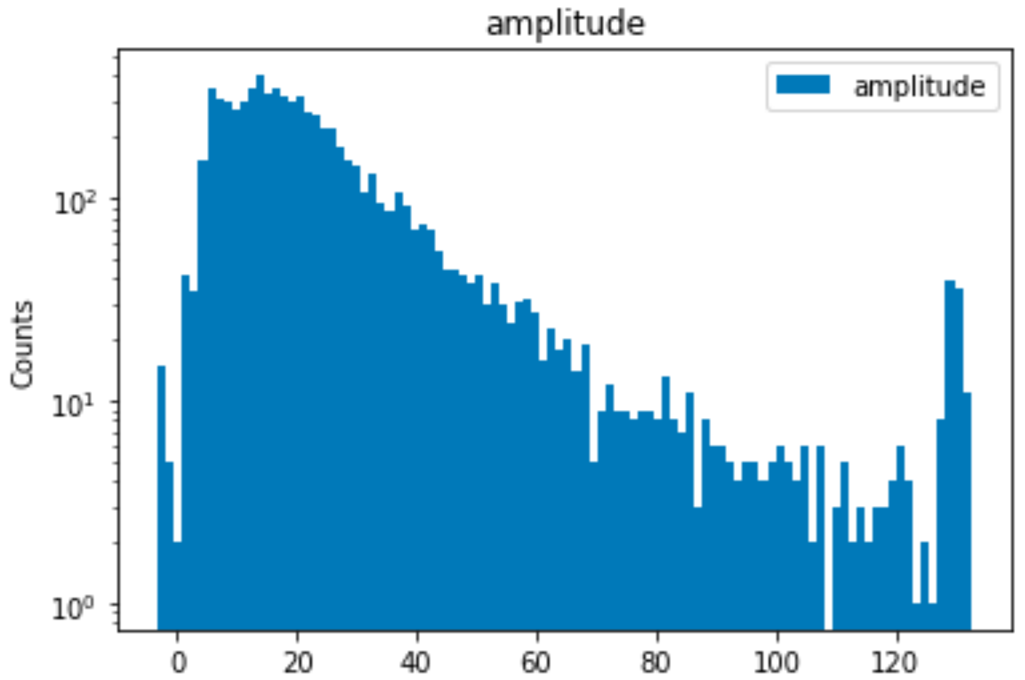
Extraction#5: Basic selections

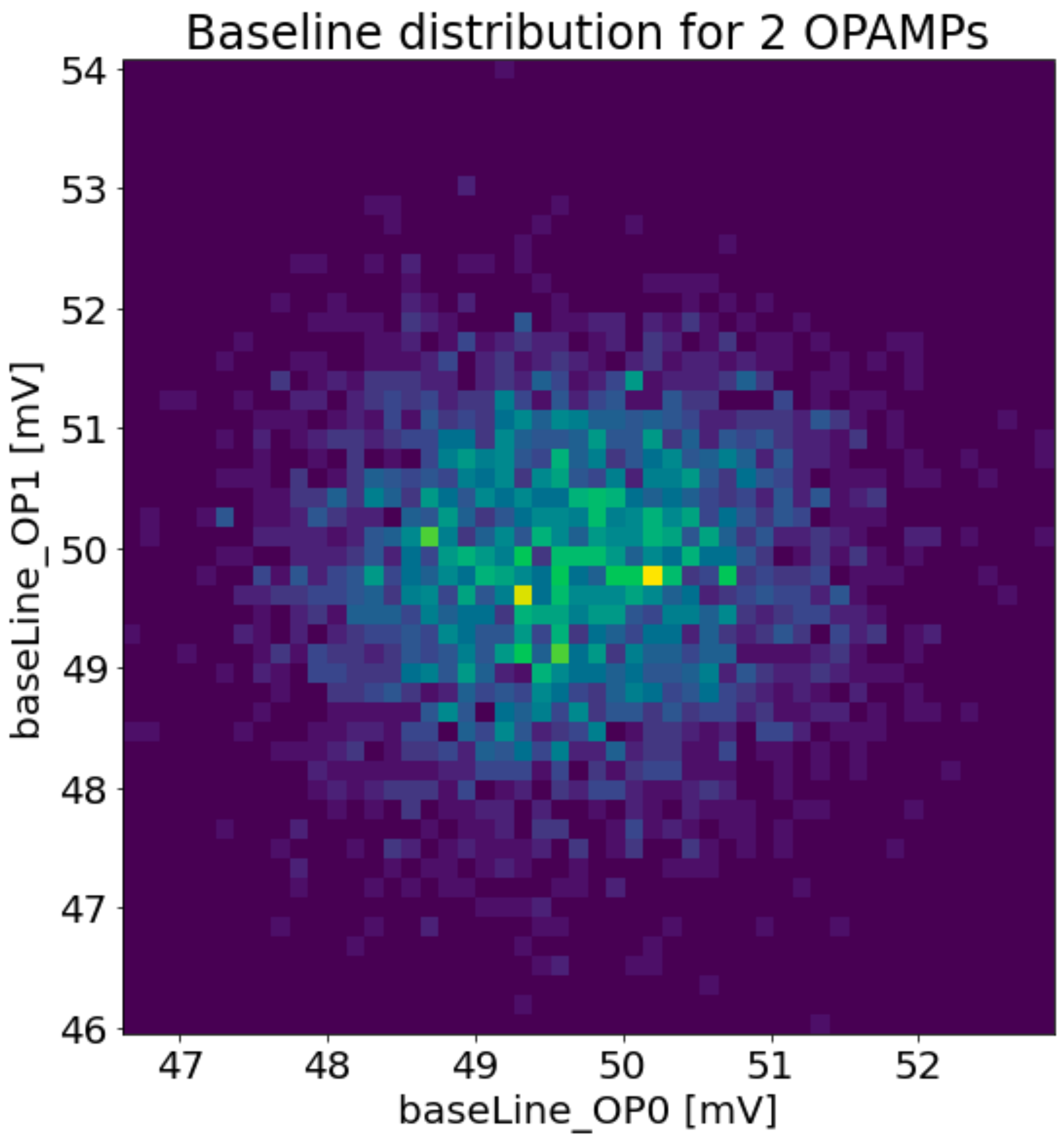
Data Extraction

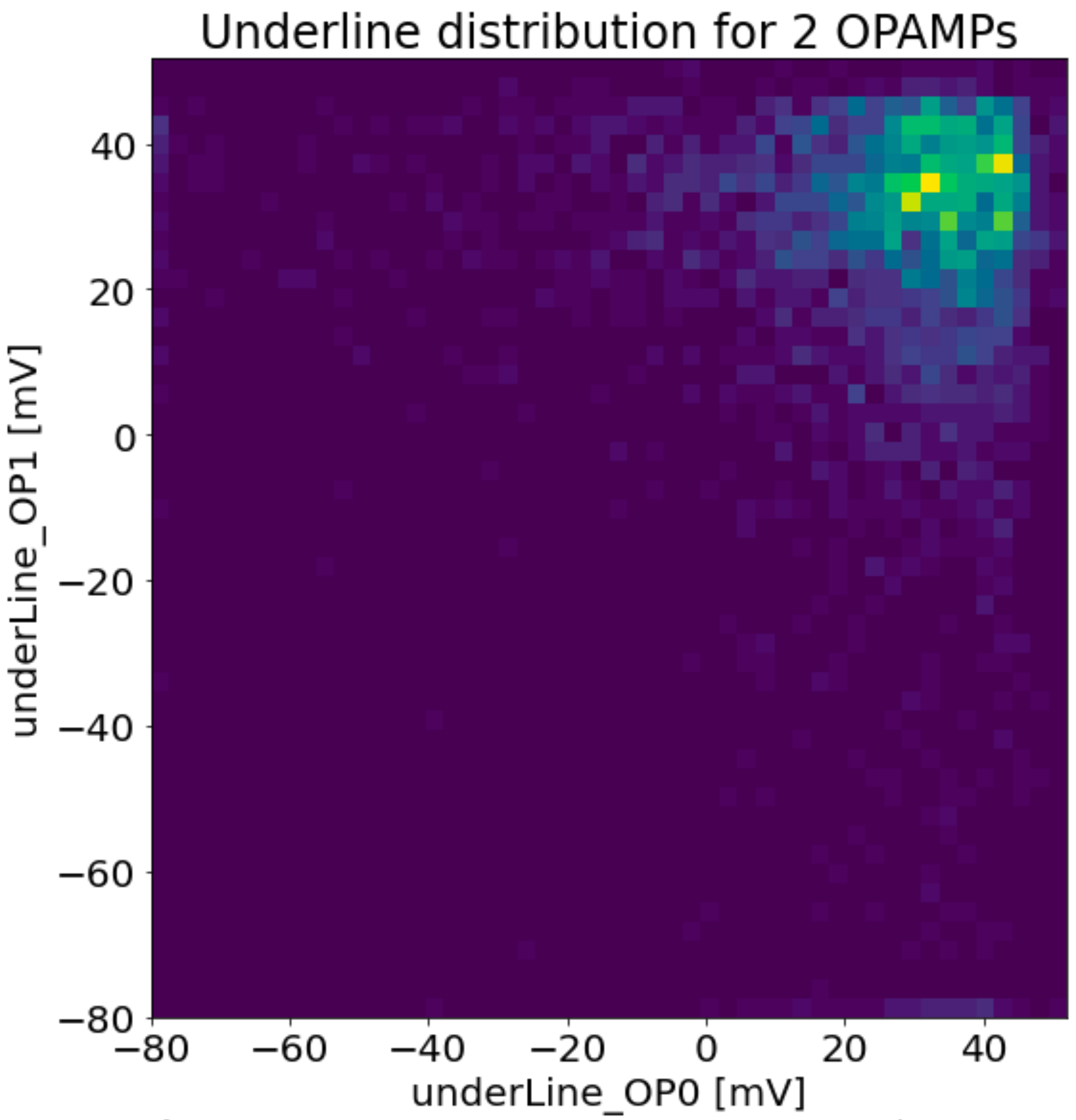
- **Basic selections (applied both OPAMPs):**
  - 45 mV < baseline < 55 mV (not affecting)
  - Underline > -79 mV (overflow)
  - Amplitude > 8.75 mV
- **Entries**: Original(3345) → underline selection (3278) → amplitude selection (2287)

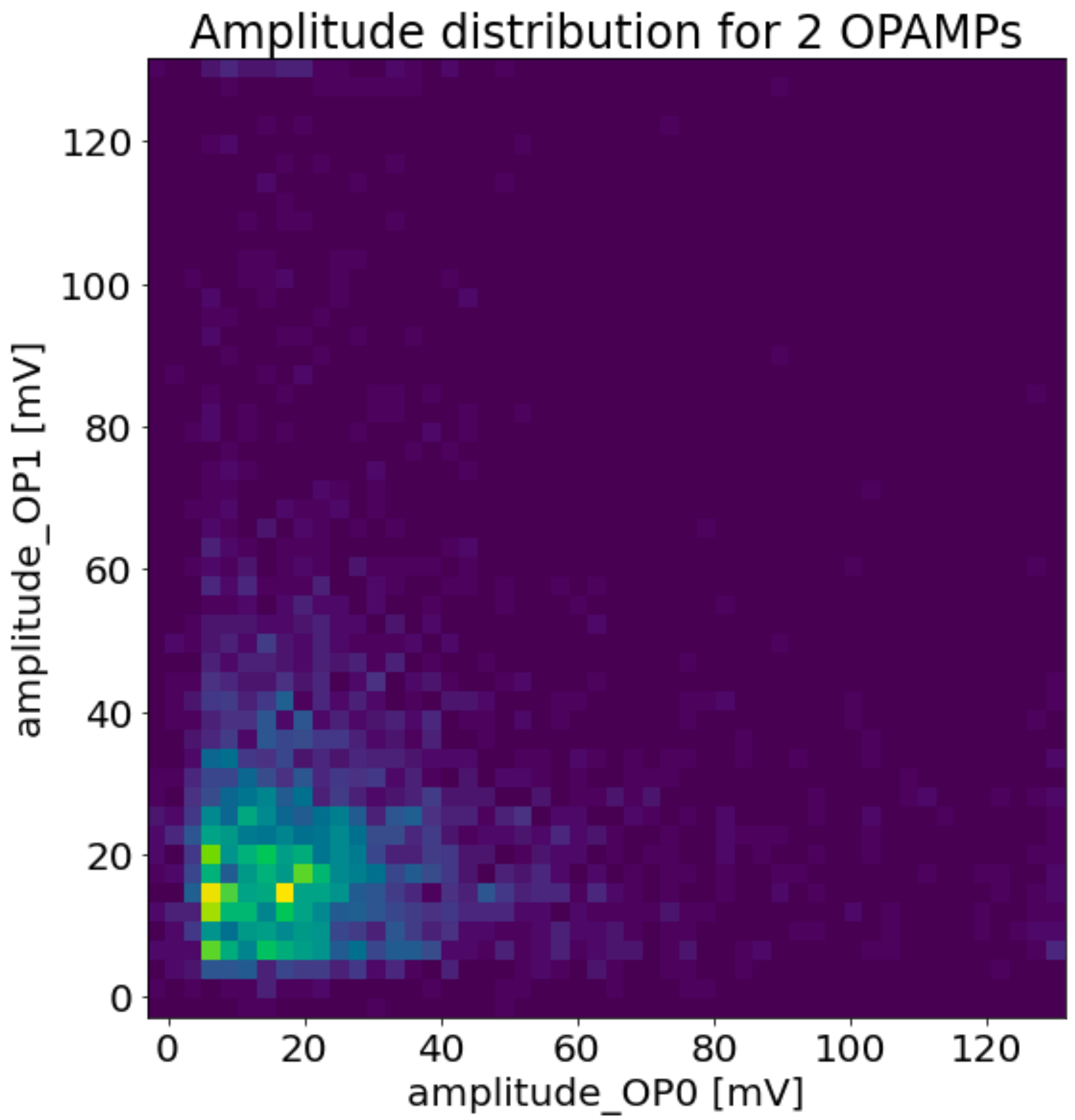






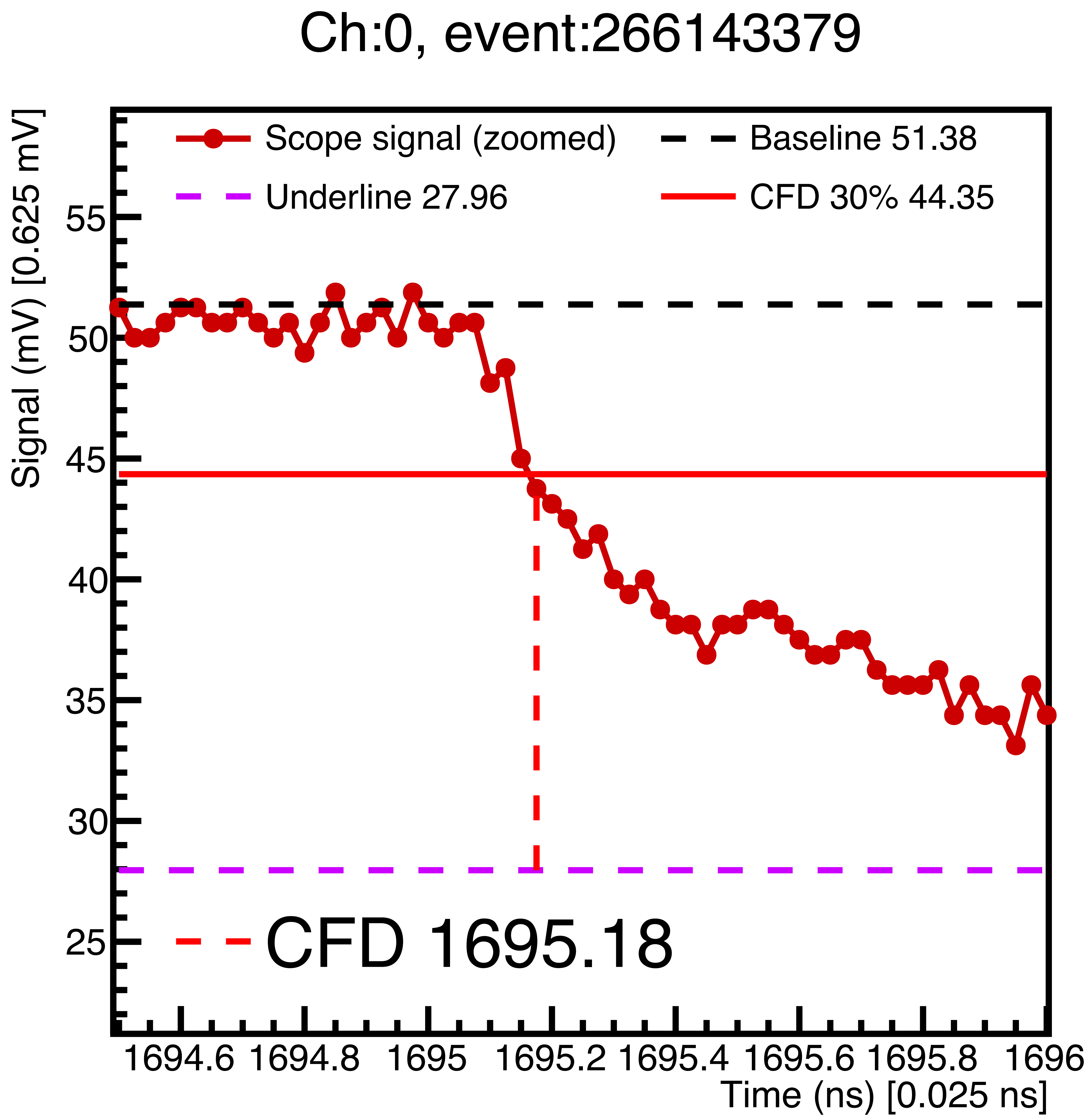






Extraction#6: Simple CFD method

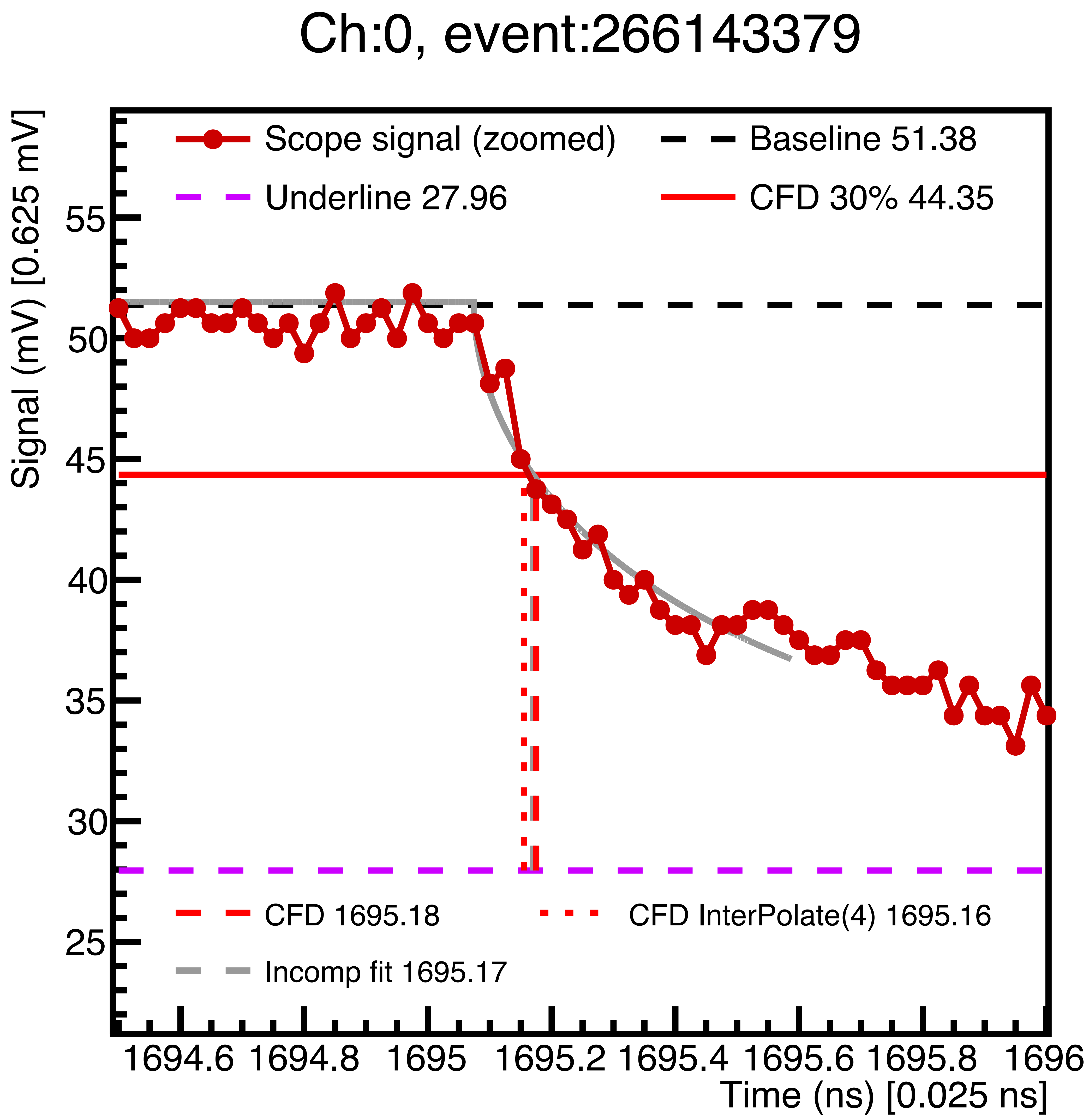
Data Extraction **Numerical computation with most simplest approach**



- **Constant Fraction Discriminator (CFD)**
- **Process:**
  - Scan the data point from left(or right)
  - Compare the point with a reference value (eg. CFD 30%)
    - **If the value is lower (higher) than the reference, use the time stamp as a CFD time stamp.**
    - If we have more than one point, **use the centre of these points.**
- Used scan signal amplitude fractions for this study: **10, 20, 30, 40, 50, 60%**
- As a systematic trial, also try to use the average x position between 2 points crossing the reference value.

Extraction#7: CFD method with interpolation

Data Extraction



- Similar to the simple CFD approach but considers **neighbourhoods.**
  - default: 4 points before/after the point
  - Systematic checks: 1,2,3,4(default) variation + backwards(4)
- Interpolation method: **Linear Regression**
  - Assume: the slope will have a linear trend in local region

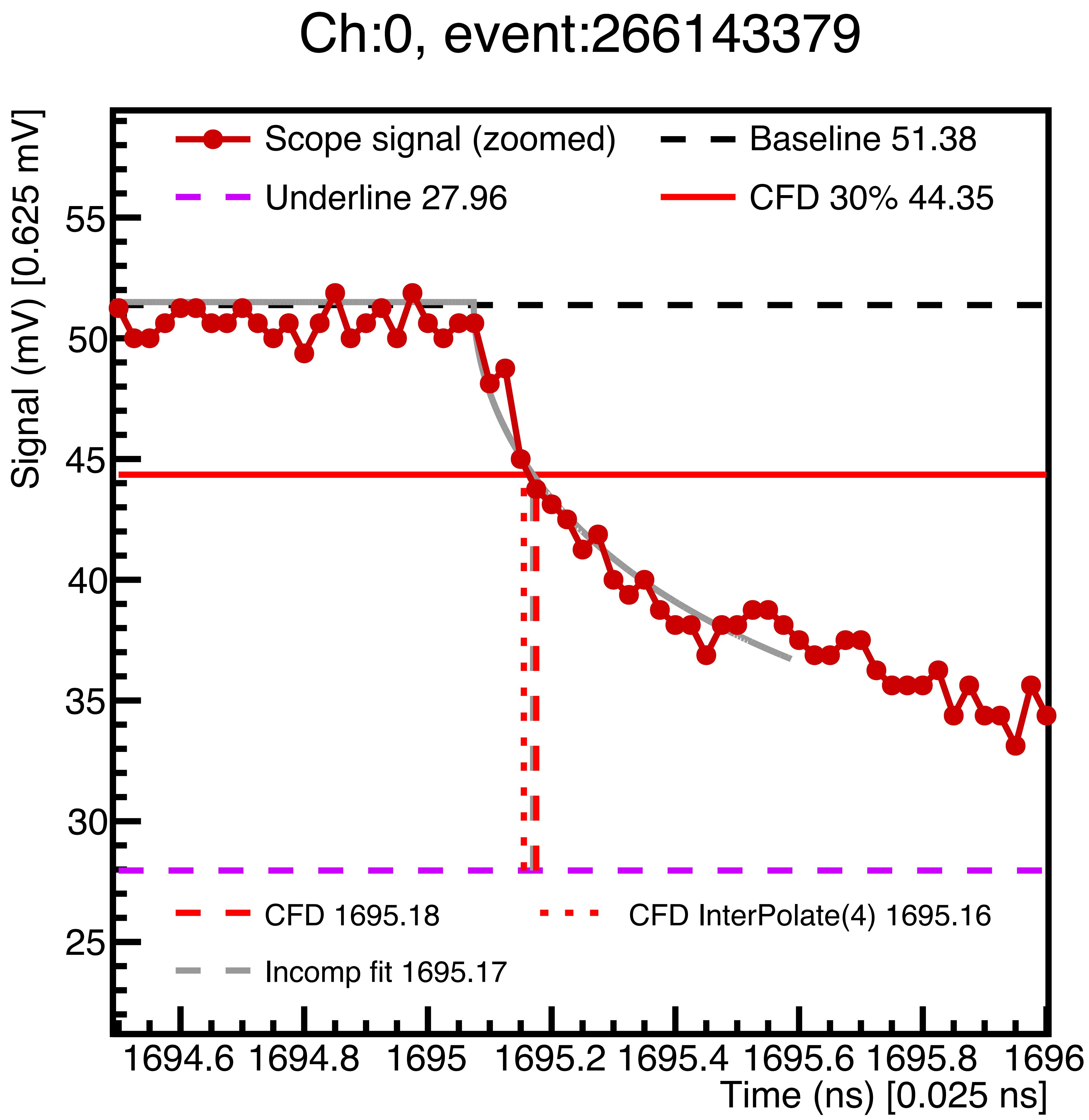
$$y = ax + b$$

  - $b = \dfrac{(\Sigma y)(\Sigma x^2) - (\Sigma x)(\Sigma xy)}{n(\Sigma s^2) - (\Sigma x)^2}$
  - $a = \dfrac{n(\Sigma xy) - (\Sigma x)(\Sigma y)}{n(\Sigma x^2) - (\Sigma x)^2}$
  - $x = (y - b)/a$

- Used scan signal amplitude fractions for this study: **10, 20, 30, 40, 50, 60%**

Extraction#8: Fit method

Data Extraction



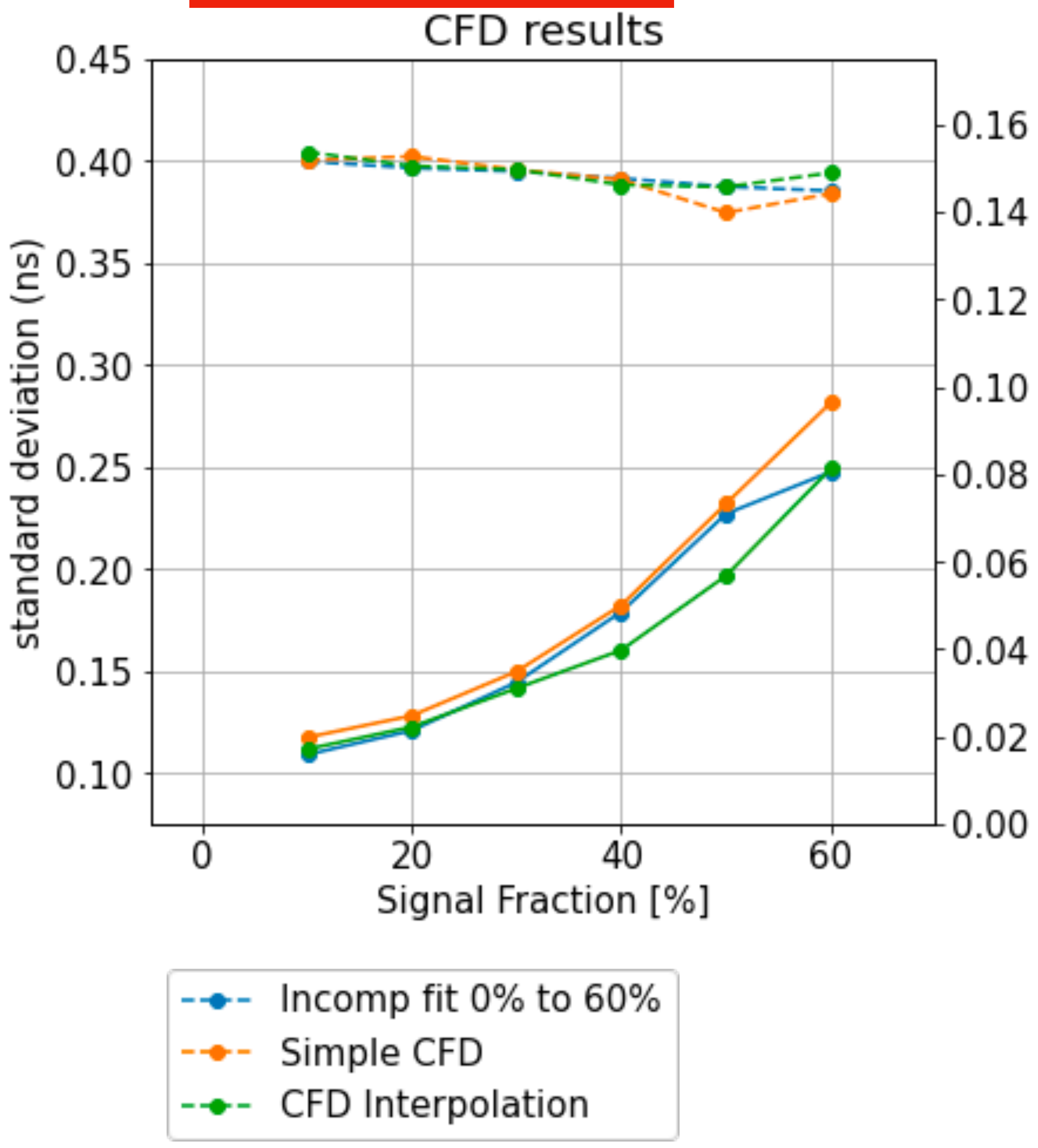
- **Incomplete gamma fit:**
  - Fit the total signal using the

$$F(t) = \begin{cases} c & t \leq t_0 \\ c - \Delta \cdot \gamma(\alpha, \frac{t - t_0}{\beta}) & t > t_0 \end{cases}$$

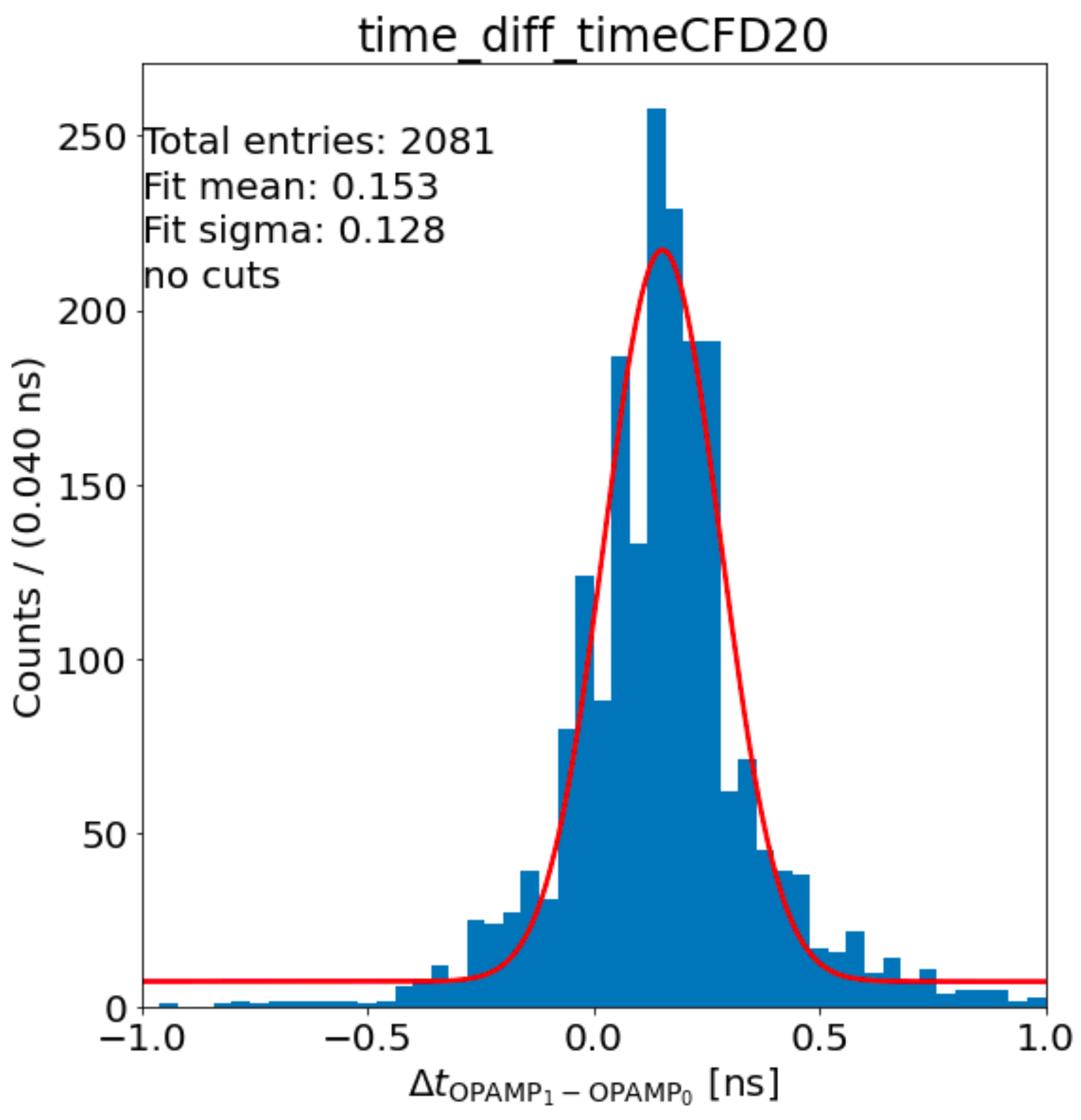
    - 5 parameters (c, t_0, Δ, α, β)
    - Reference: Appendix.B, CERN-THESIS-2017-304
    - Extract the time stamp position based on fit function
  - fit range variation
    - Full size
    - up to CFD 60% positions.
  - Tested for comparison.

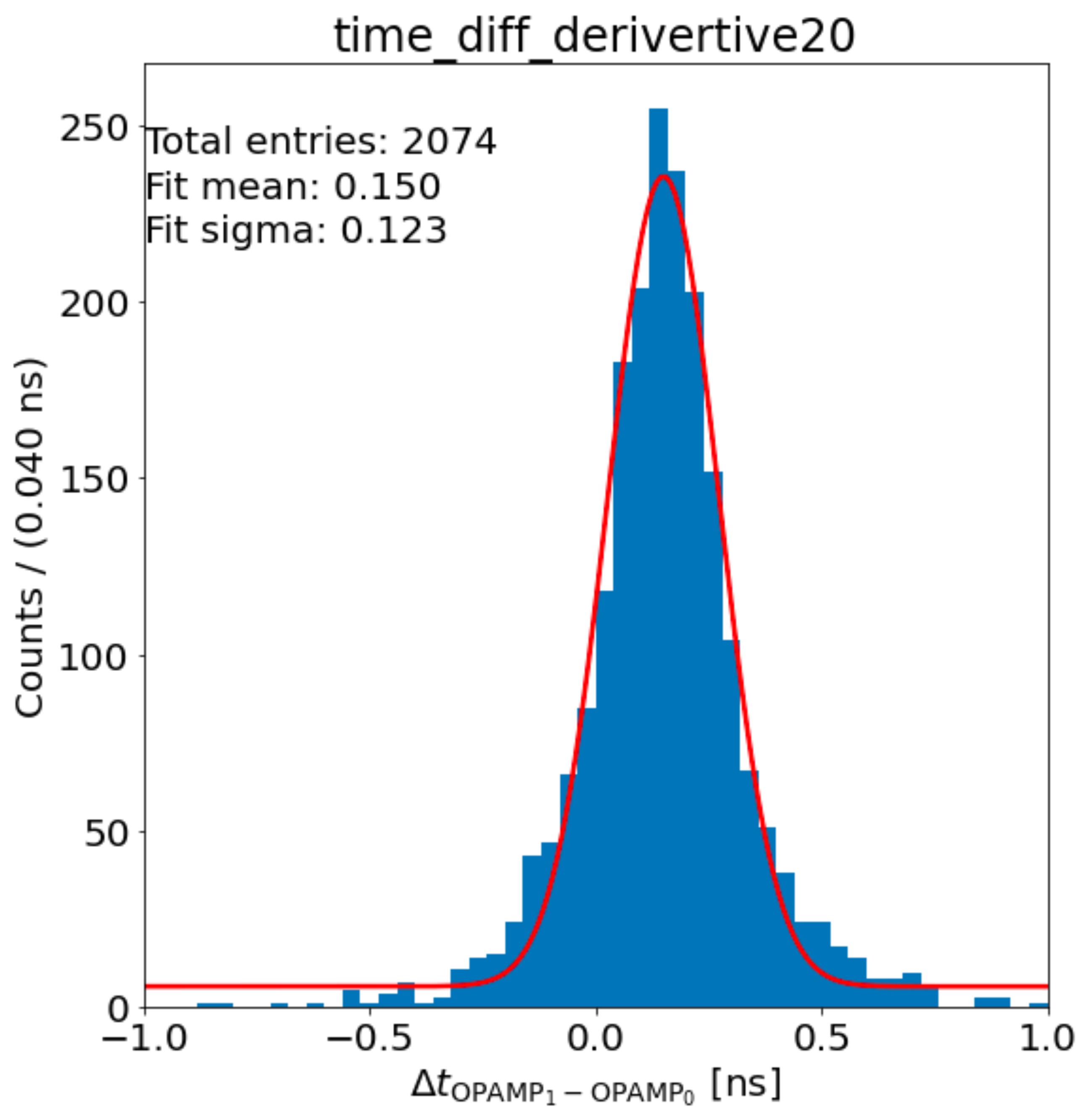
Final plots#1: CFD methods

Final plots



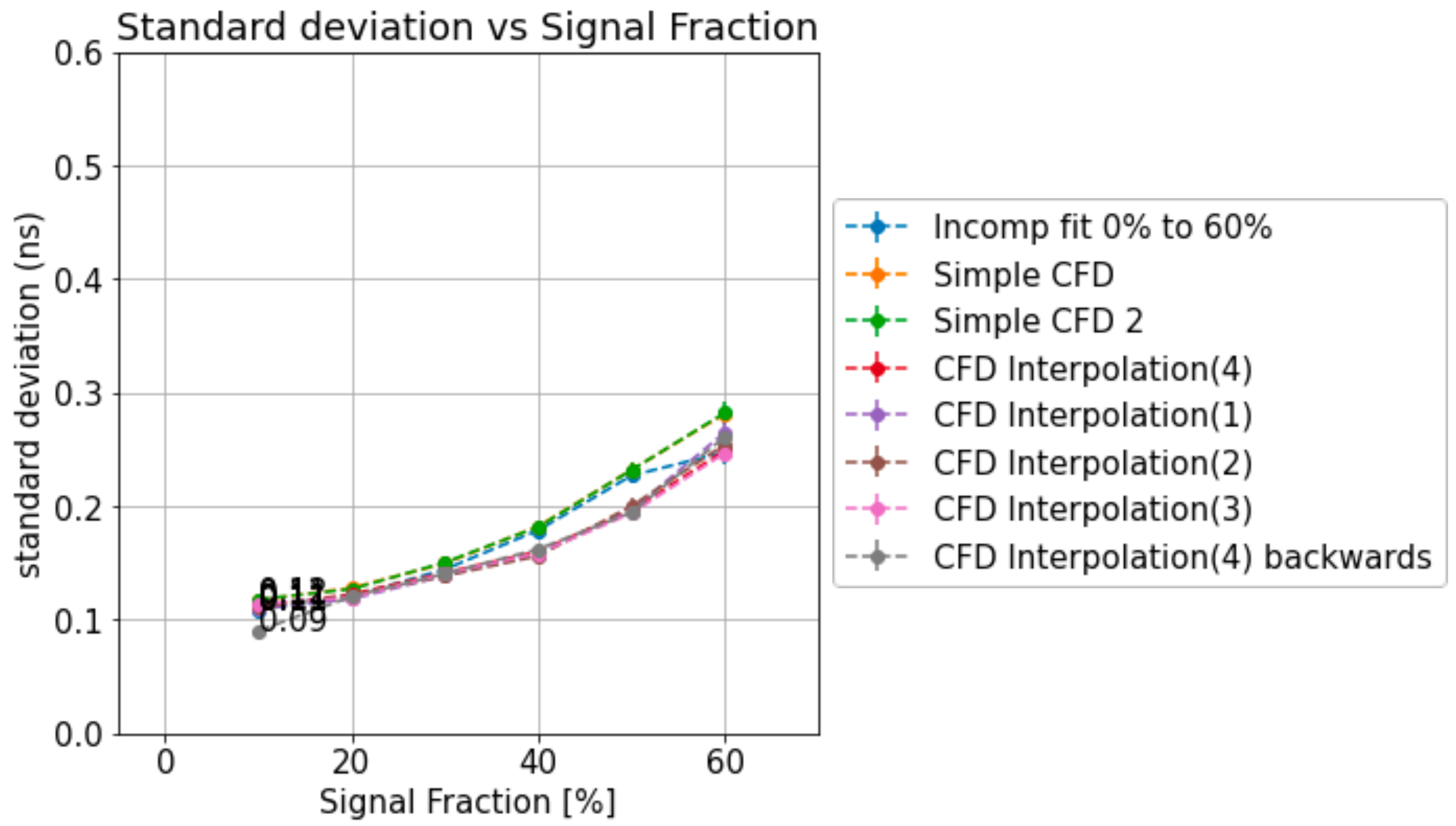
- **Time difference was calculated and fitted with gauss function.**
  - eg. $\Delta t_{30\%} = t_{30\%|\mathrm{OPAMP}\,1} - t_{30\%|\mathrm{OPAMP}\,0}$
  - Expected gauss mean: ~0.167 ns (~5cm interval with speed of light)
- The result extracted from the mean is roughly compatible with expected value
  - Gauss mean shows a small fluctuation.
  - Standard deviation from Gauss fit shows smallest value at 10% signal fraction.

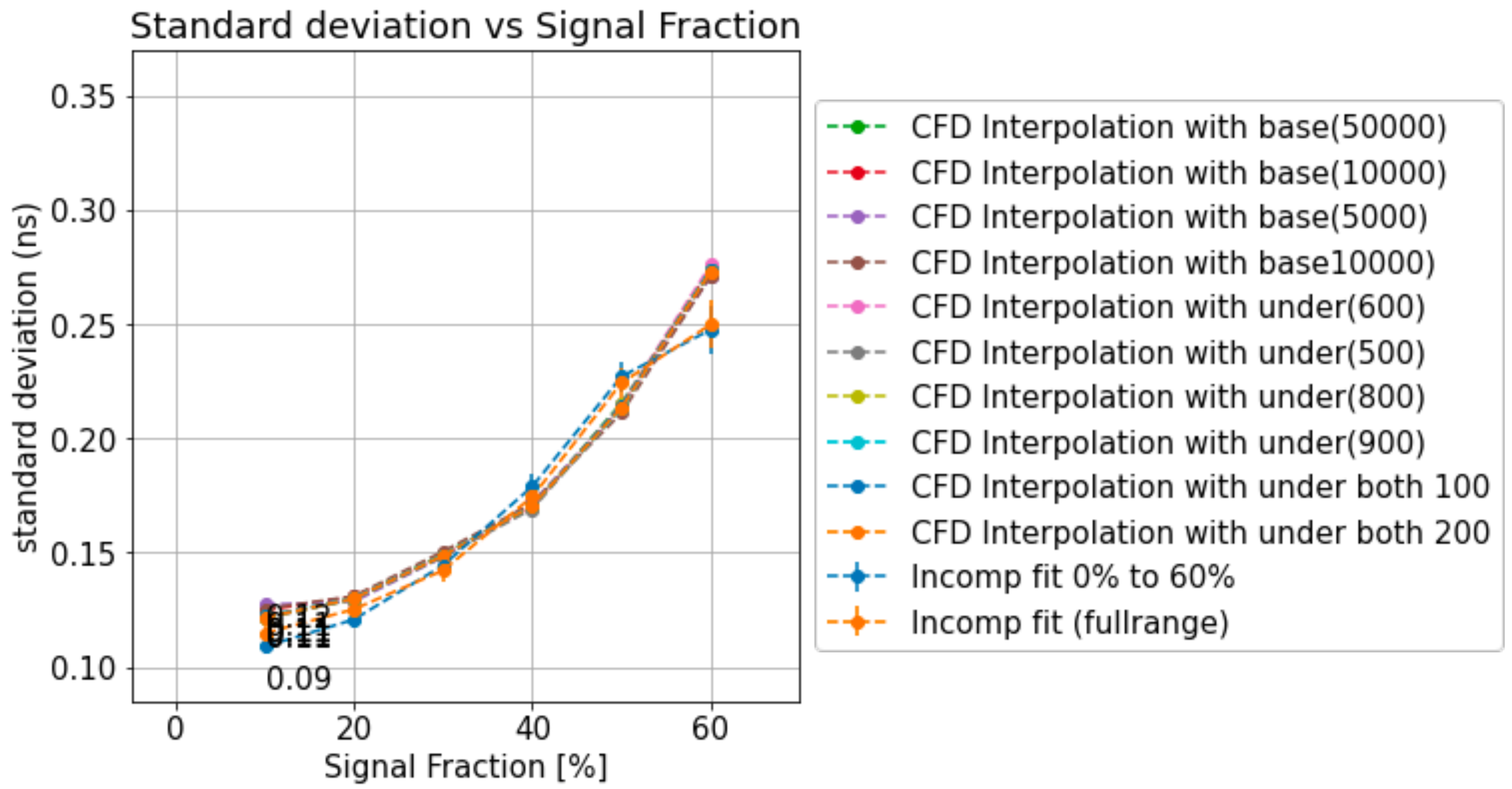




Final plots#2: CFD methods variations

Final plots

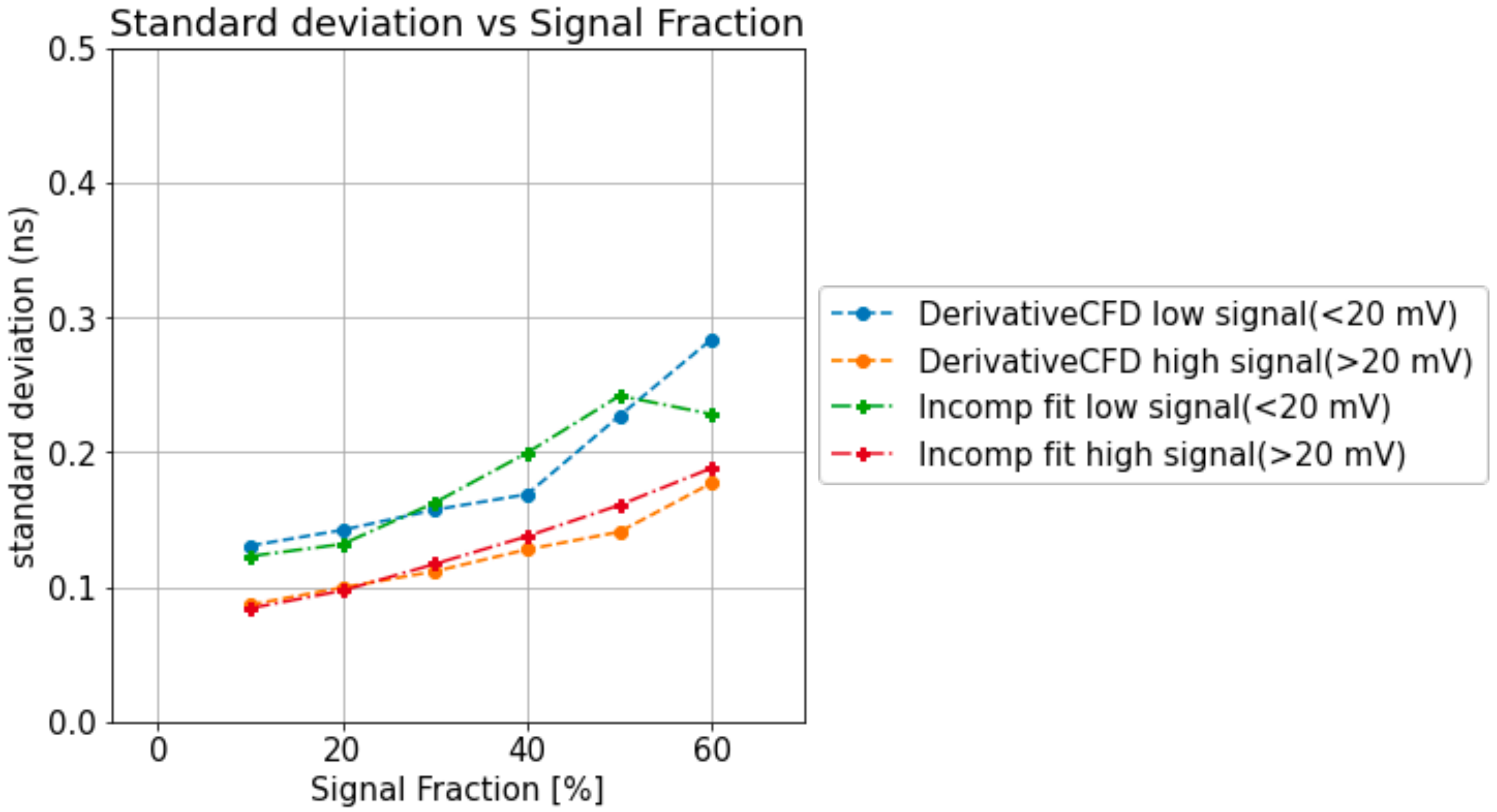


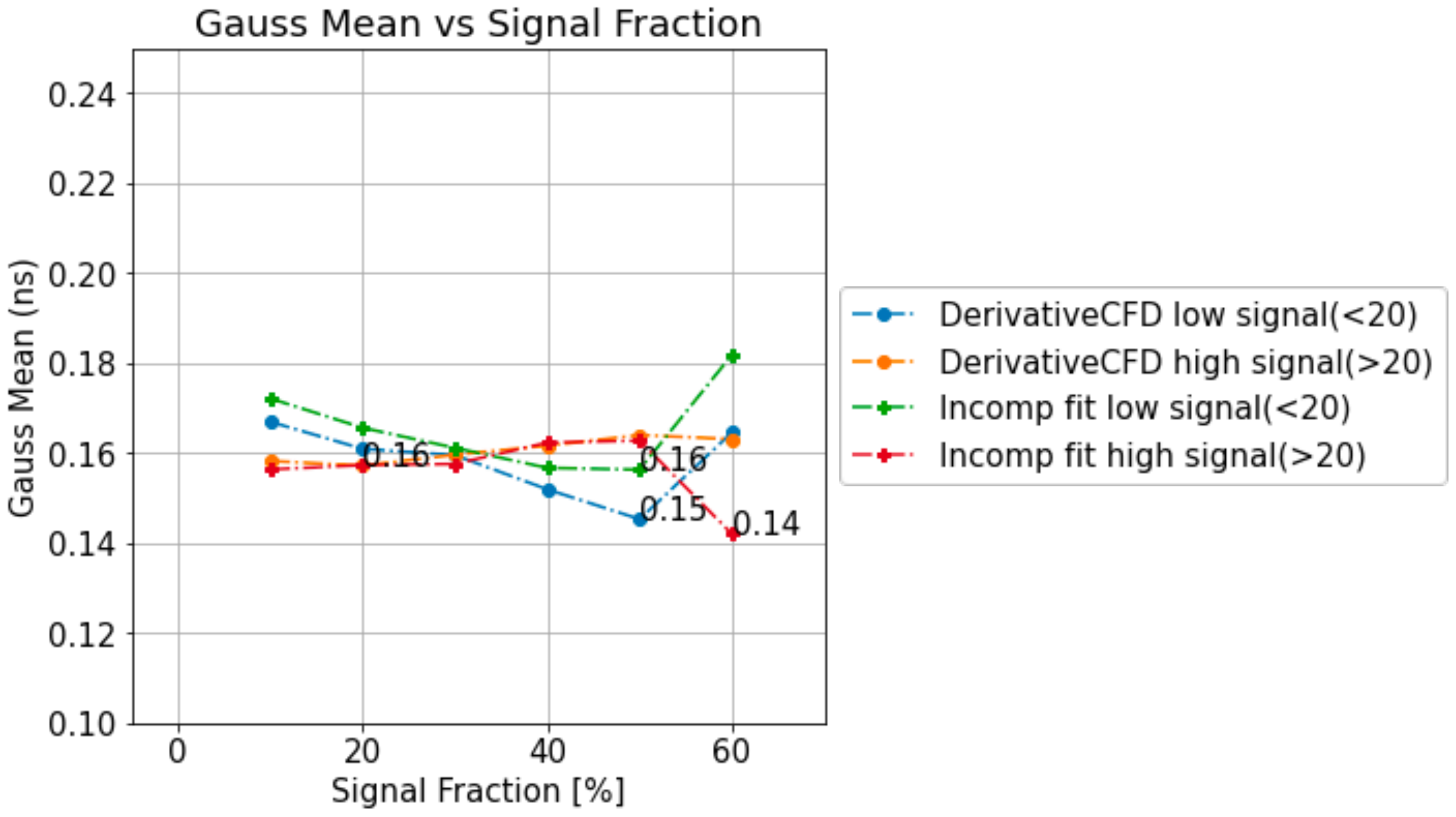


- **Systematic studies**
  - Based on the previous analysis methods
  - Incomplete gamma fit with full signal range
  - Simple CFD2 - Using average between CFD value and before.
  - CFD Interpolations - Using various intervals for the linear regression 1,2,3,4 with counting from backward
  - CFD interpolation with various baseline and underline determination trials.
    - Baseline: 50000/10000/5000/1000 bins
    - Underline: try to move the window to left/right/both directions.
- They can be used as a systematic uncertainties.
- 30% case looks stable.

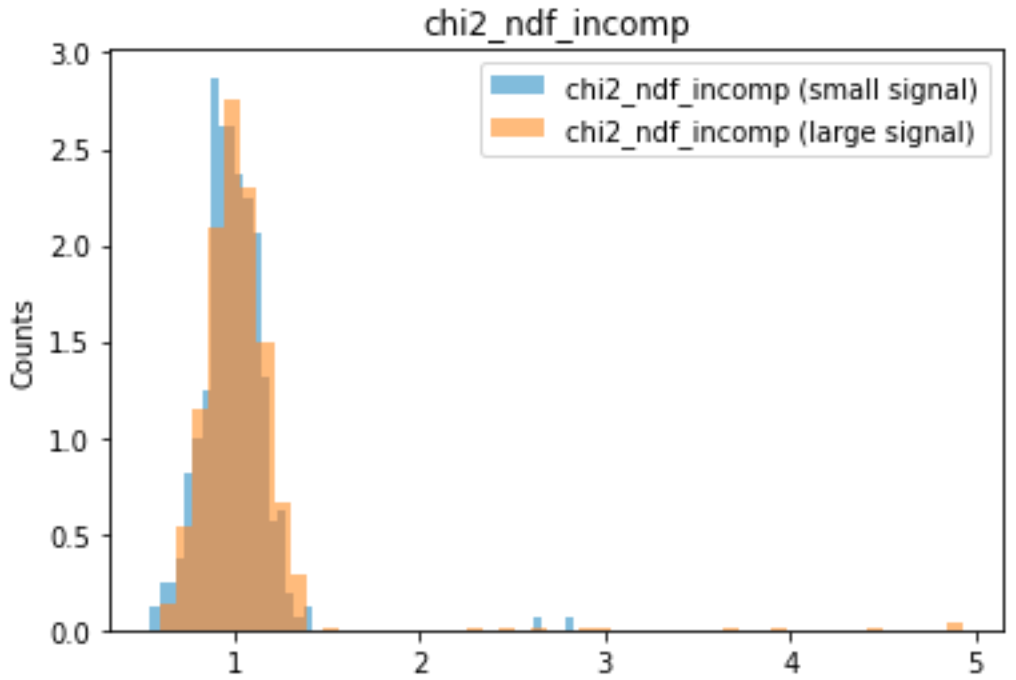
Final plots#3: CFD Result vs Signal Amplitude

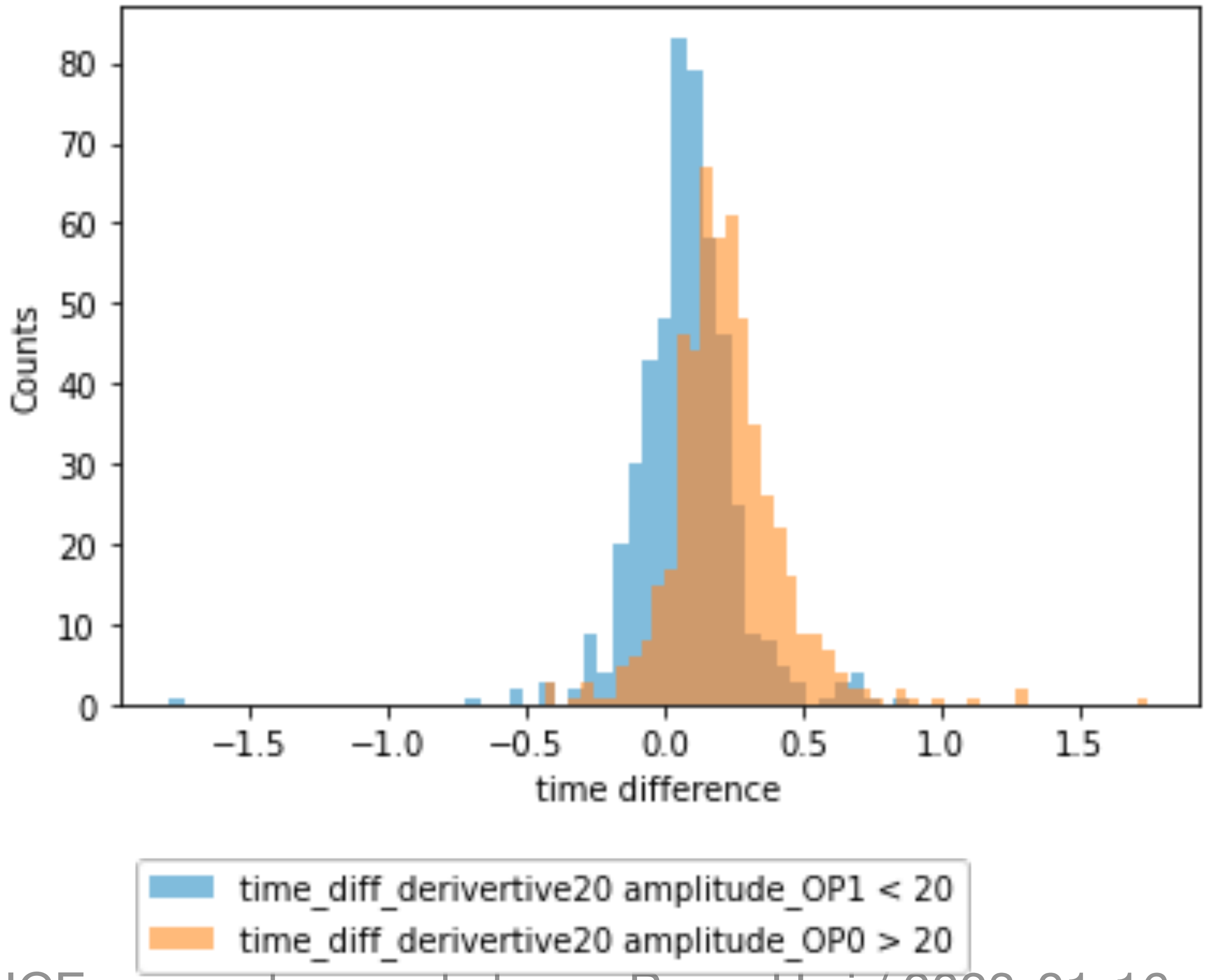
Final plots

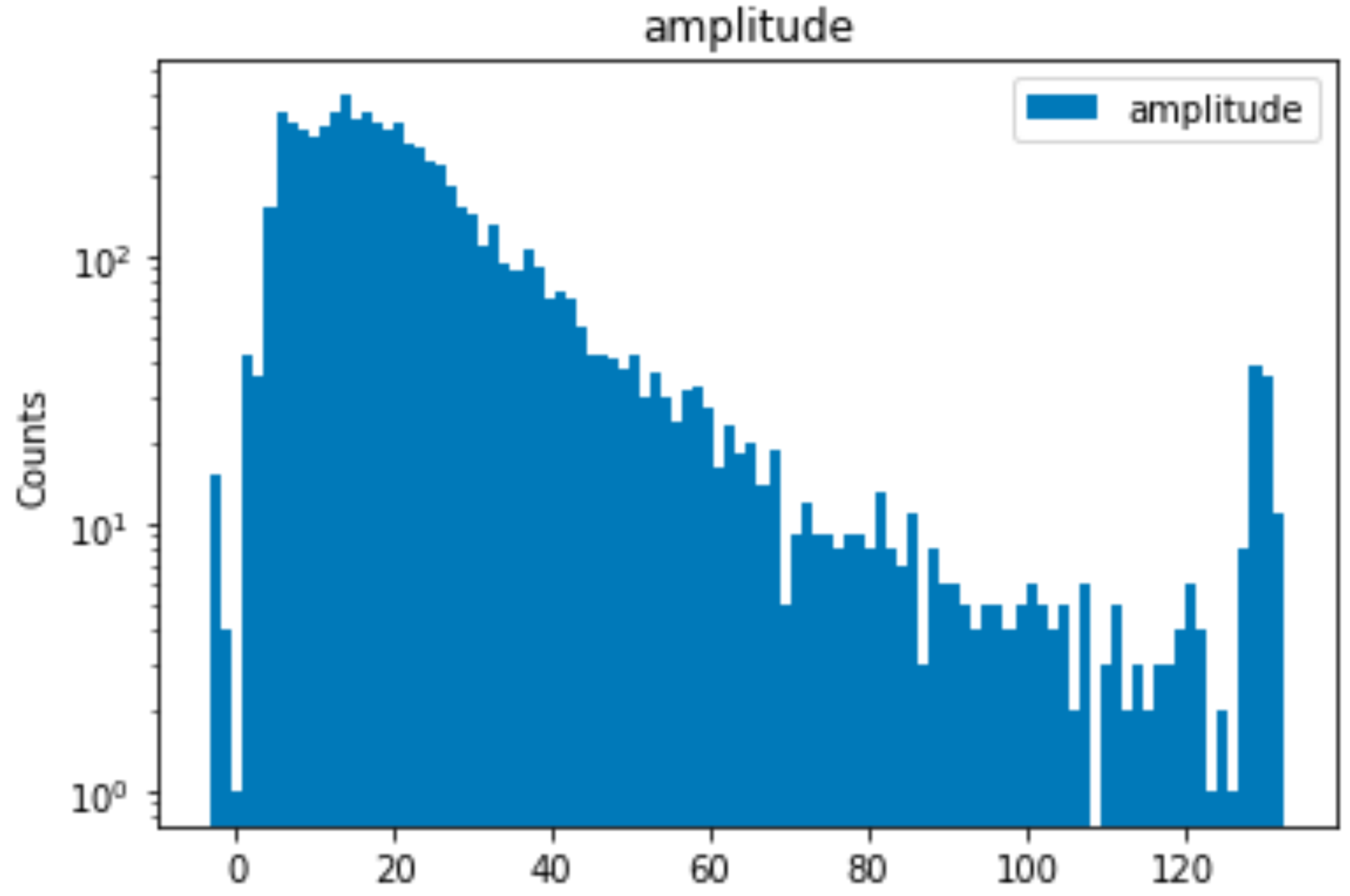




- **Signal amplitude dependence**
  - Larger signal (> 20 mV) has systematically lower standard deviations.
    - Amplitude median: ~19 mV
  - Their mean values are compatible.
- Bad fit results due to the different signal shape? -> Fit χ^2 looks similar for these 2 types
  - This is not the case.
- Time difference shows clear difference.







Summary and Outlook

- **TB analysis of APTS-OA time resolution has been performed.**
  - Conversion -> Extraction -> Analysis process
  - Determination of Baseline and Underline (Amplitude)
  - Time difference based on various methods are shown:
    - Simple CFD method
    - CFD method with interpolation
    - Fitting method

- **Outlook**
  - Detailed study of the dependence on the signal amplitude
  - Compute proper error estimation
    - Stat: ROOT Fit parameter uncertainty
    - Syst: Update based on the systematics studies

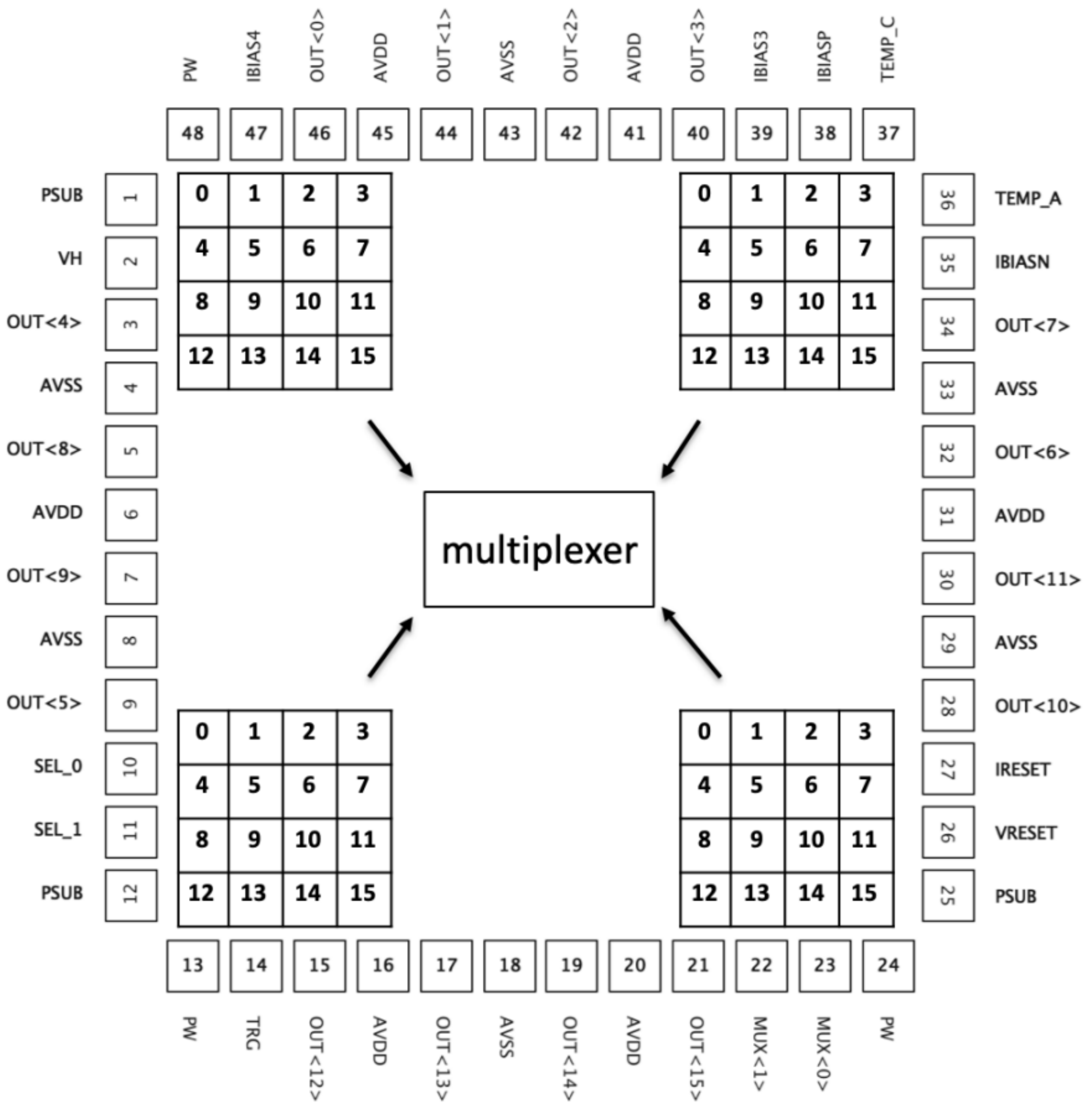
Back up

Comparison of chip to carrier board connection between APTS-SF V1.1 and APST-OA V2

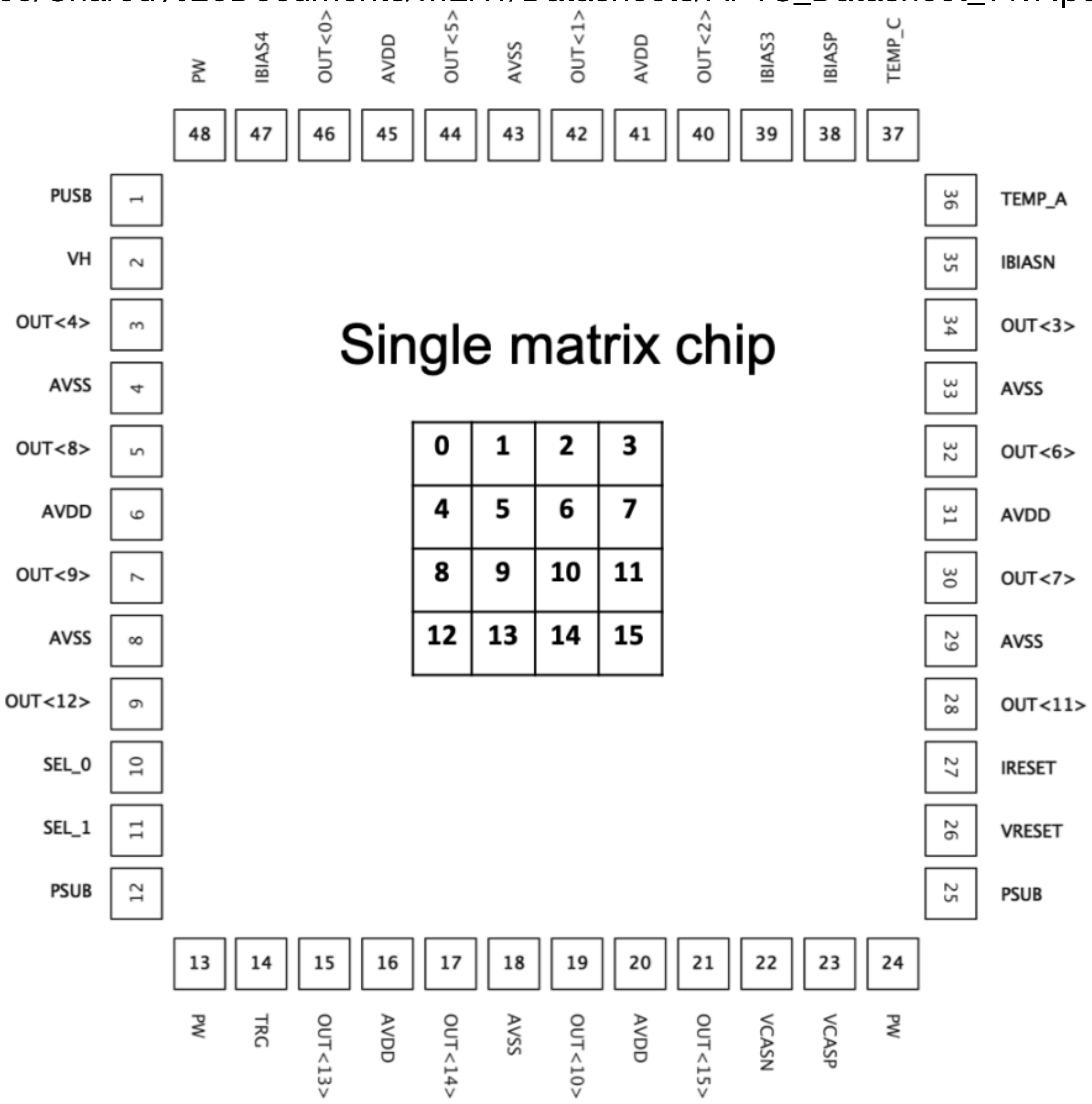
APTS-SF

Materials

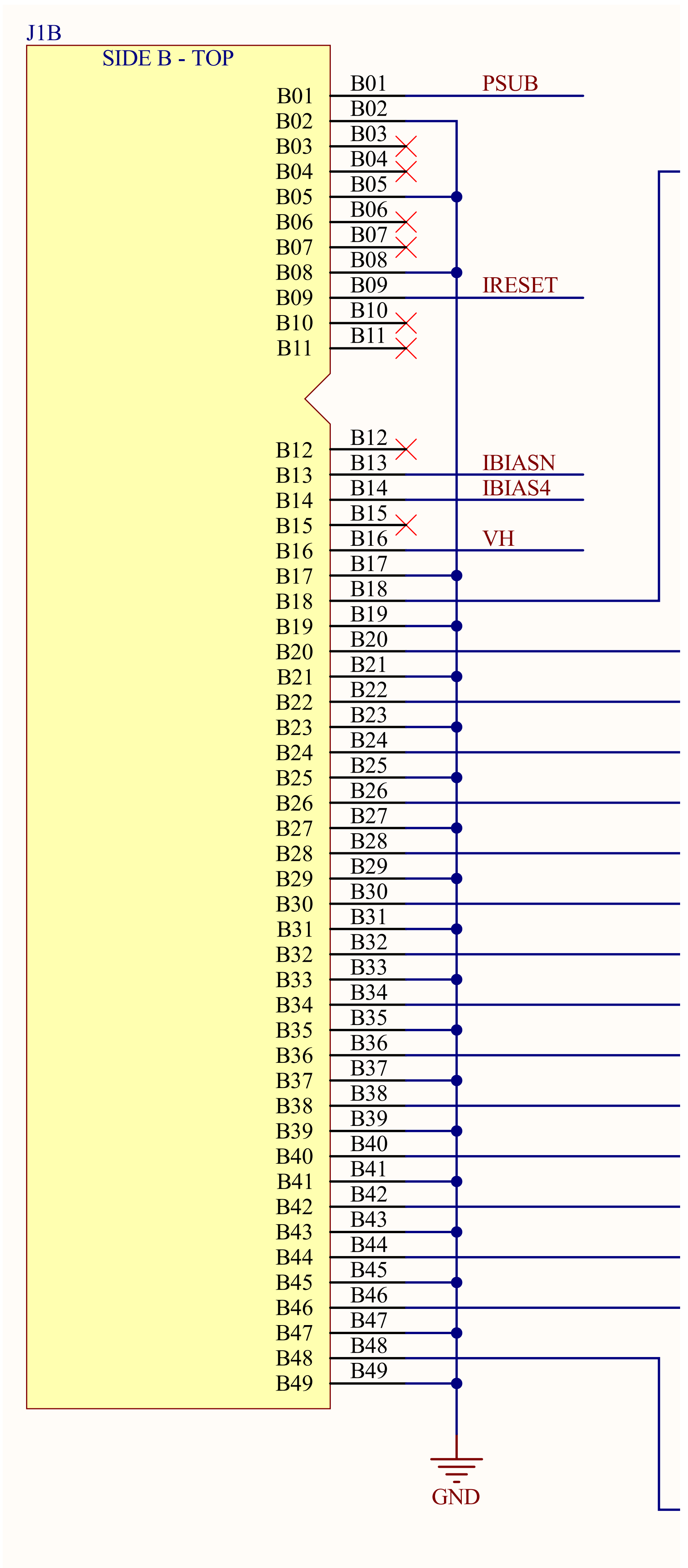
Reference: https://espace.cern.ch/ep-project-rnd-tpsco65/Shared%20Documents/MLR1/Datasheets/APTS_Datasheet_v1.7.pdf



APTS-SF (MUX)



APTS-SF

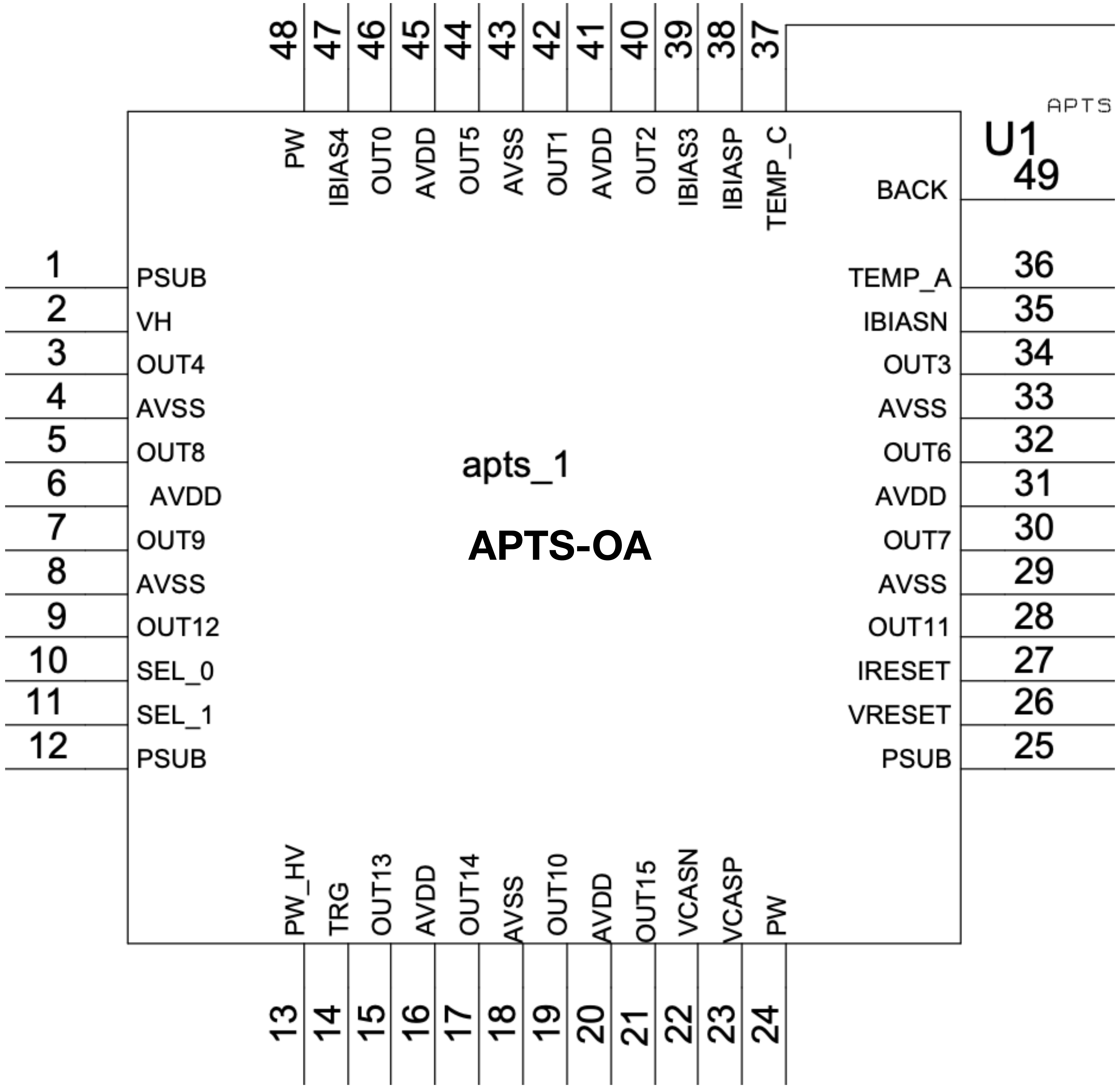


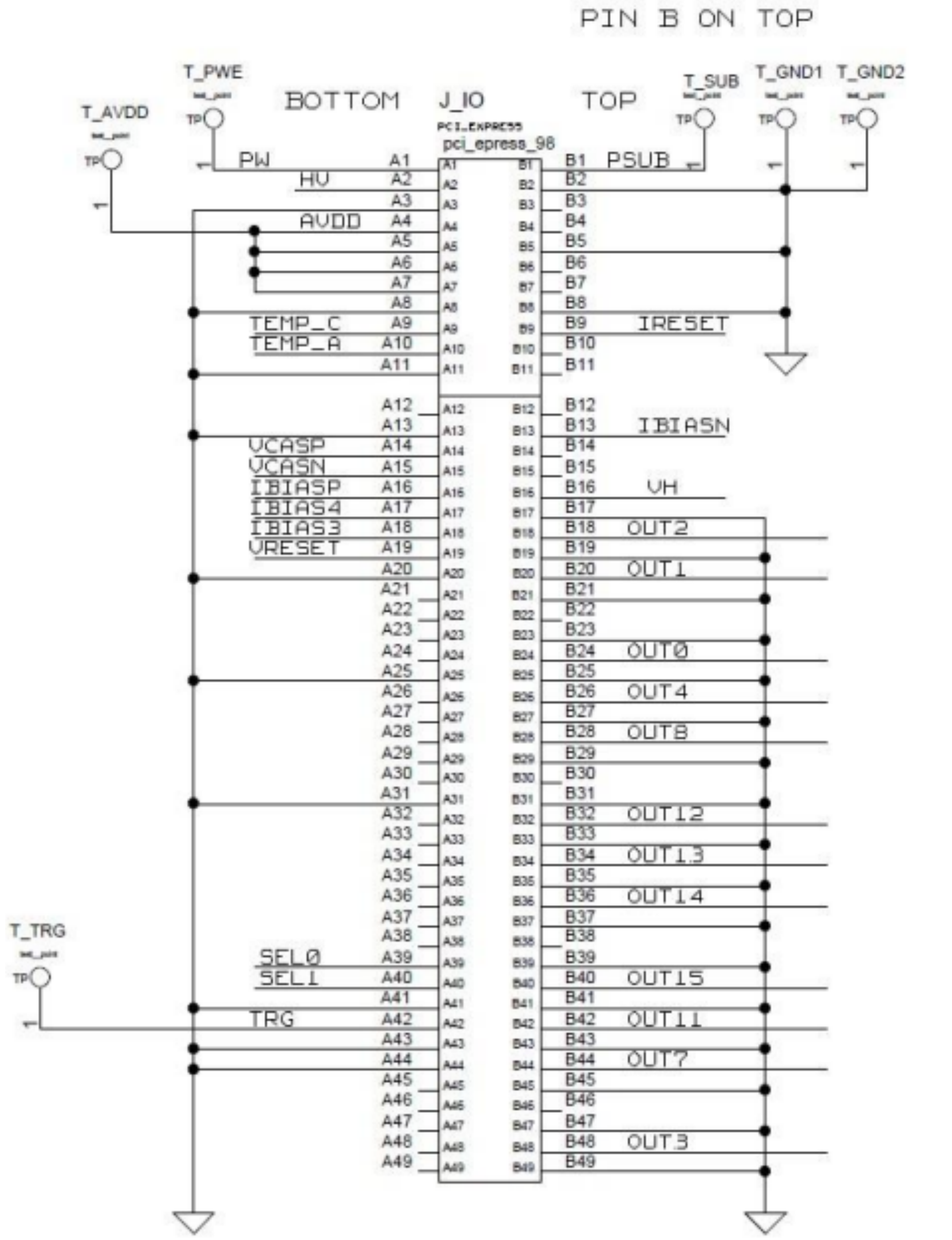
APTS-SF V1.1
Edge connector

APTS-OA (OPAMP)

Materials

Reference: https://twiki.cern.ch/twiki/pub/ALICE/ITS3WP3MLR1TestSystem/APTS_OA_ver1_schematic.pdf





APTS-OA V2 Edge connector

APTS: Chip output to Edge connector

Details

Chip to Edge	APTS-SF (MUX)	APTS-SF	APTS-OA
OUT0	46	46	46
OUT1	44	42	42
OUT2	42	40	40
OUT3	40	34	34
OUT4	3	3	3
OUT5	9	44	44
OUT6	32	32	32
OUT7	34	30	30
OUT8	5	5	5
OUT9	7	7	7
OUT10	28	19	19
OUT11	30	28	28
OUT12	15	9	9
OUT13	17	15	15
OUT14	19	17	17
OUT15	21	21	21

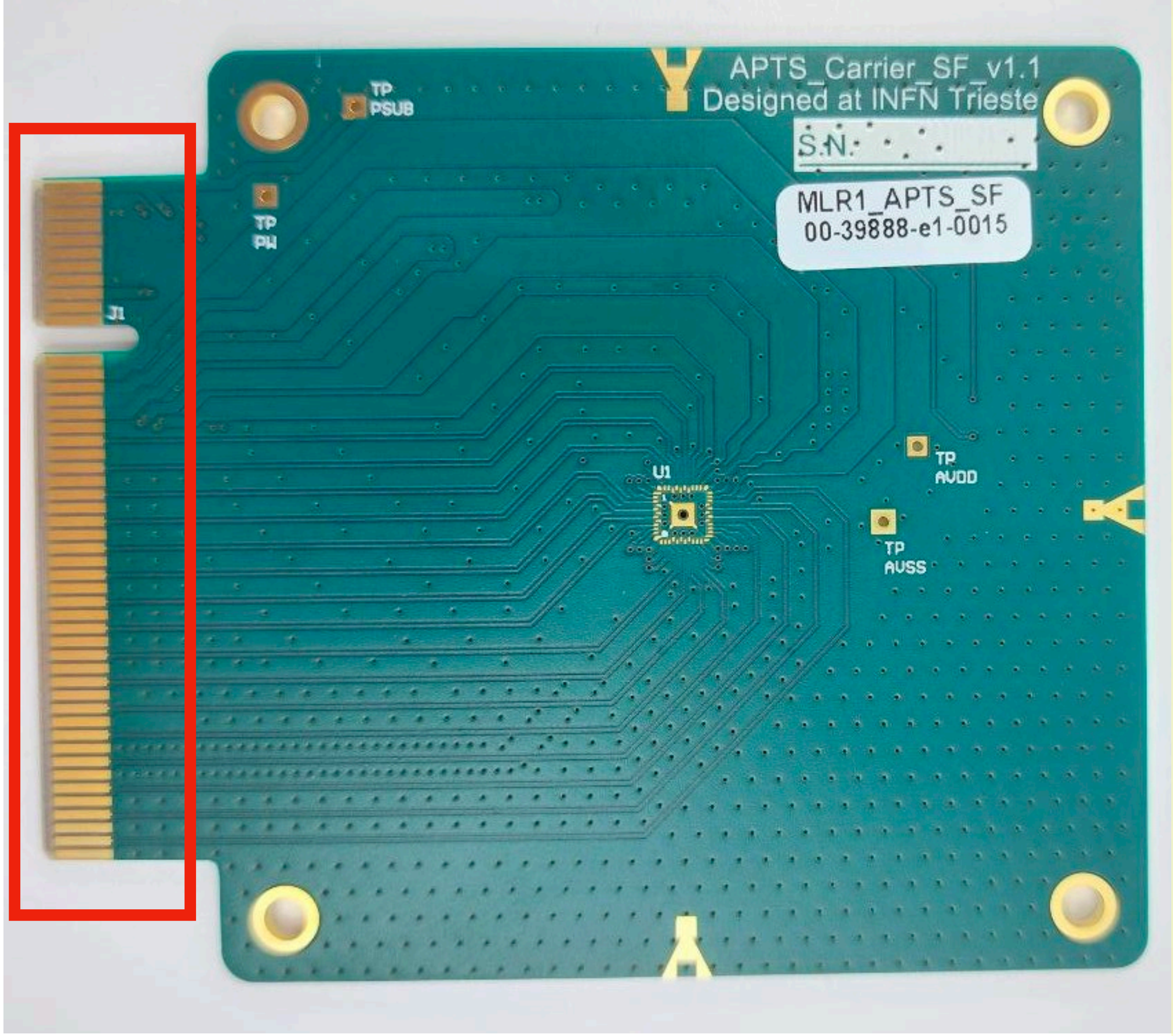
- ATPS-SF and APTS-OA have the same connection to the Edge.
- APTS-SF (MUX) has slightly different connections.

Real connection: APTS-SF v1.1

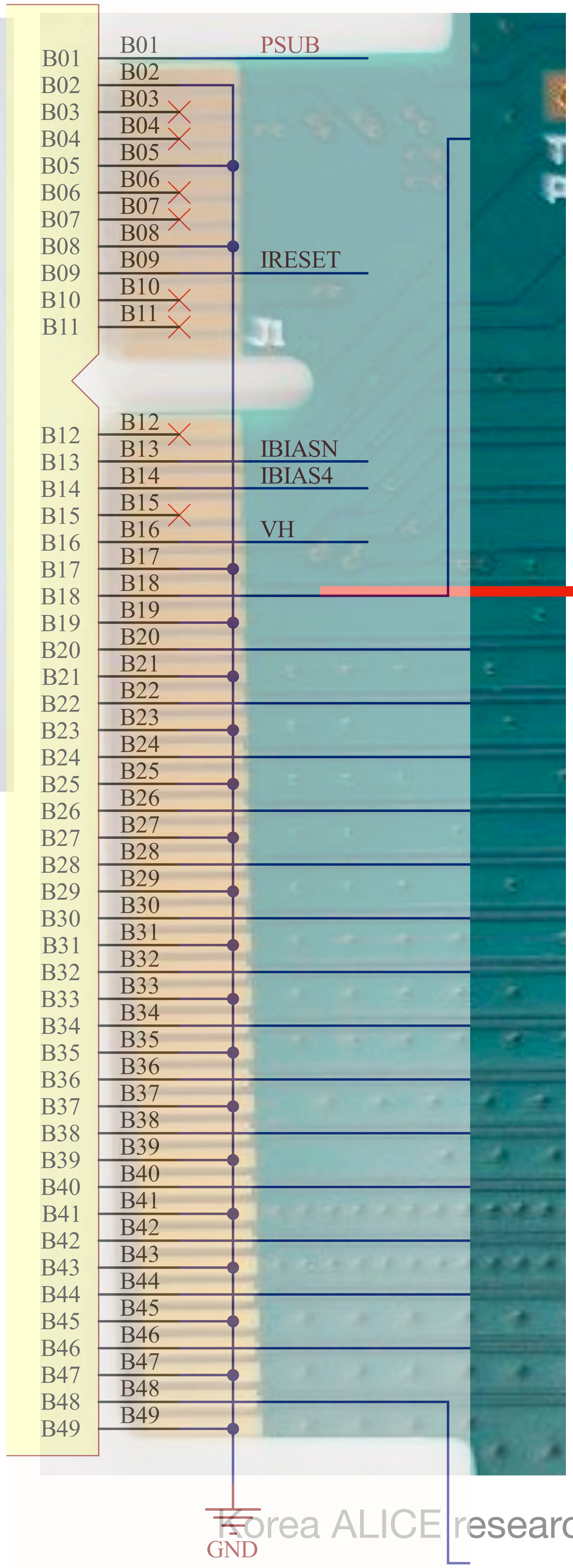
UNIVERSITA DEGLI STUDI DI TORINO

ALICE

Reference: https://twiki.cern.ch/twiki/pub/ALICE/ITS3WP3MLR1TestSystem/APTS_SF_Carrier_v2_top.jpeg



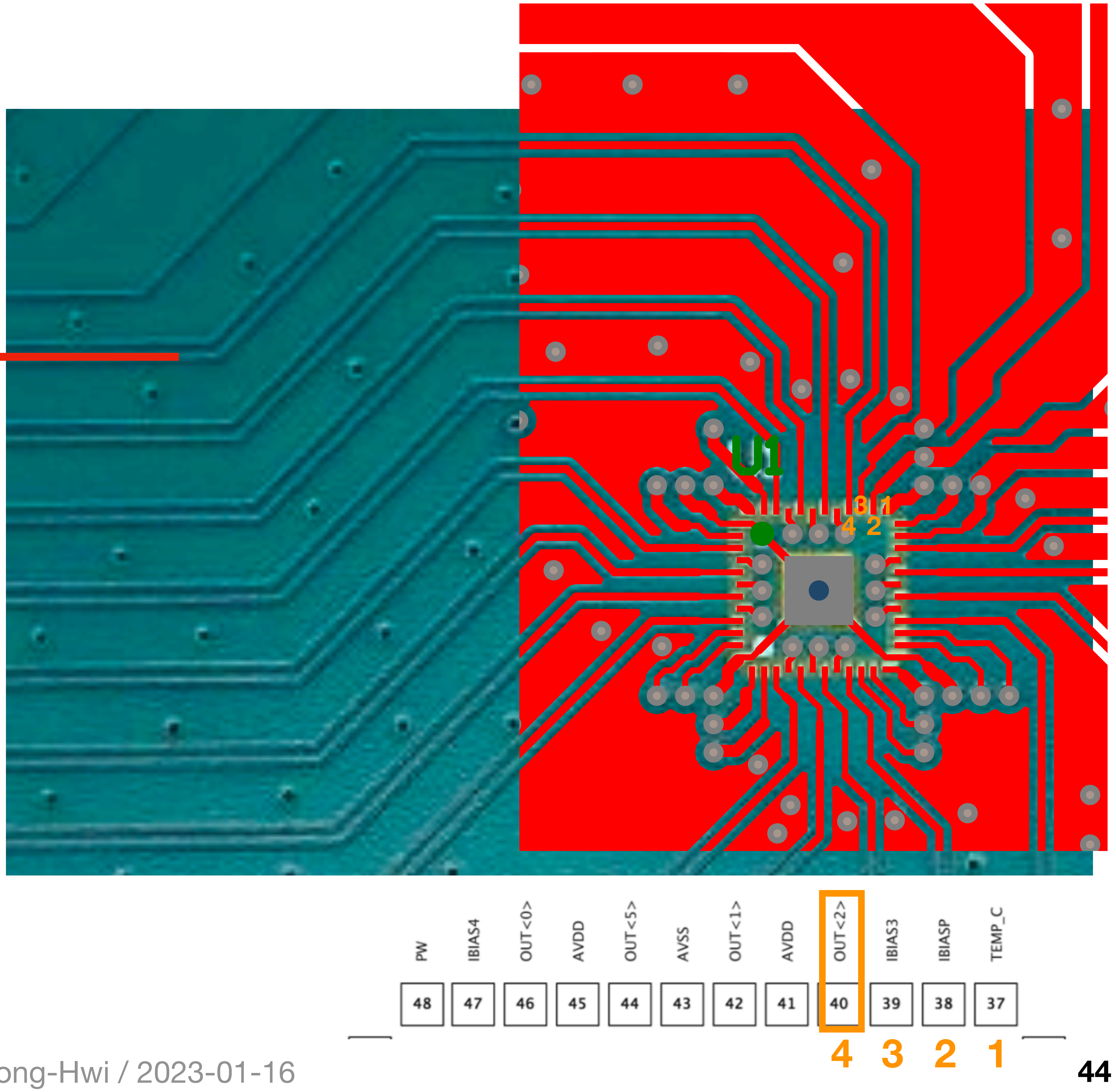
APTS-SF v1.1



APTS-SF V1.1
Edge connector
(overlay)

OUT2

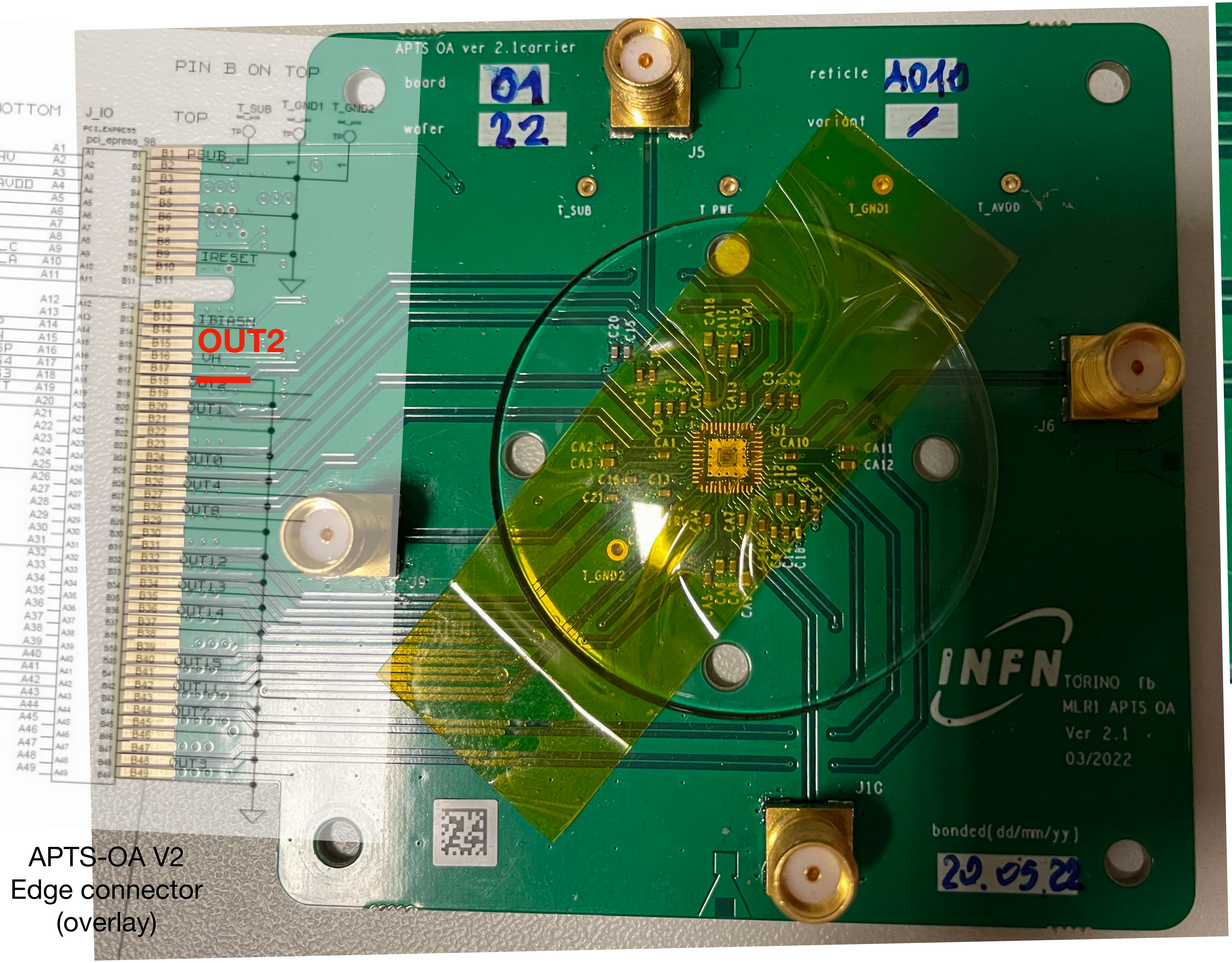
from: APTS_SF_Carrier_Board_PCB.pdf



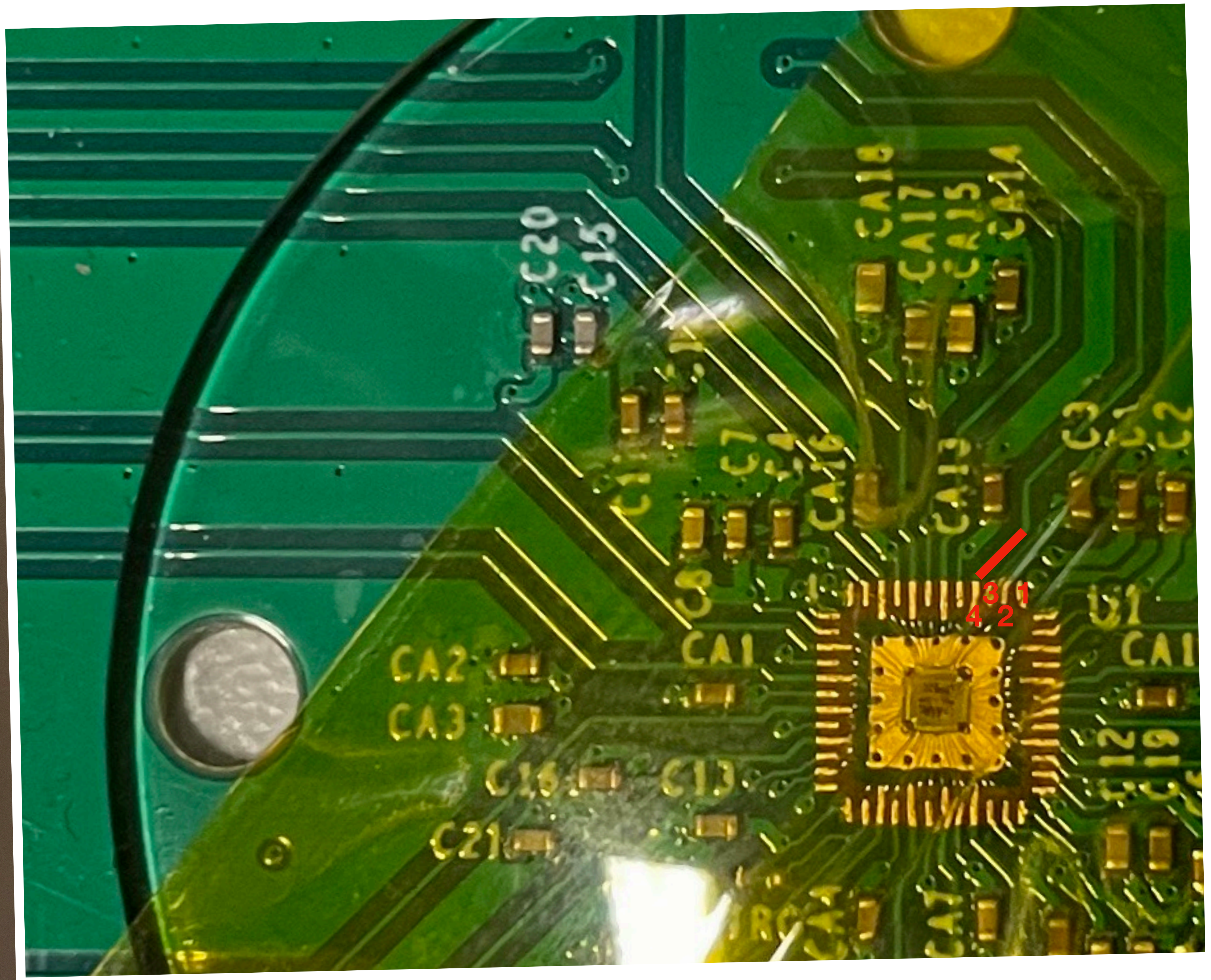
Real connection: APTS-OA v2.1

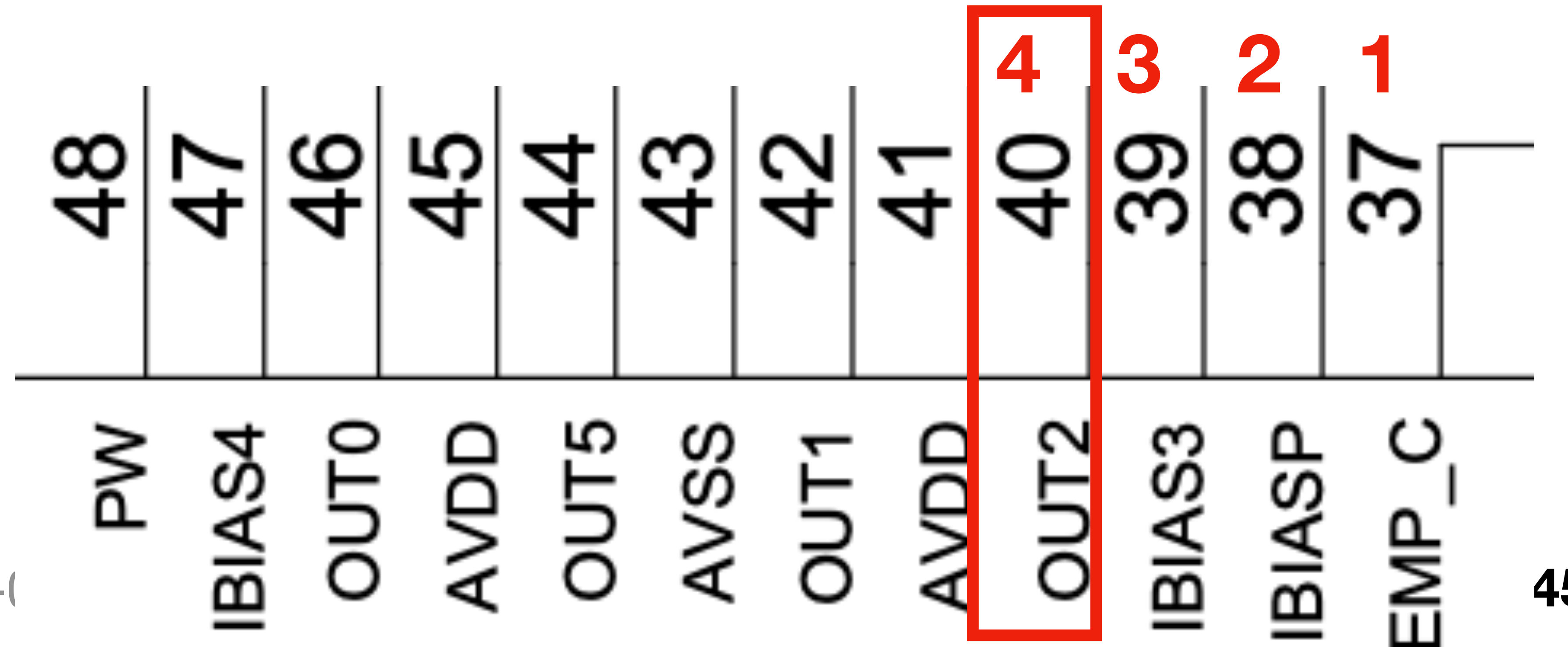
UNIVERSITA DEGLI STUDI DI TORINO

ALICE



APTS-OA V2
Edge connector
(overlay)



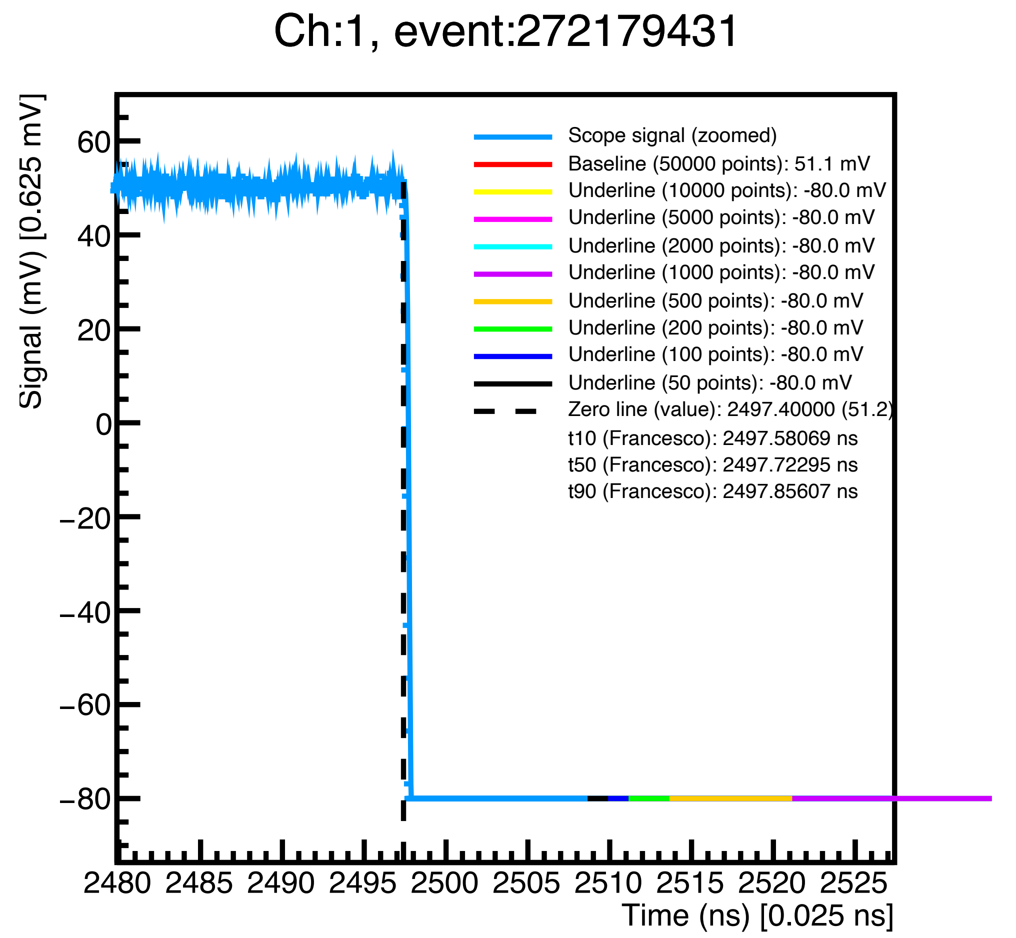


Intrinsic uncertainty

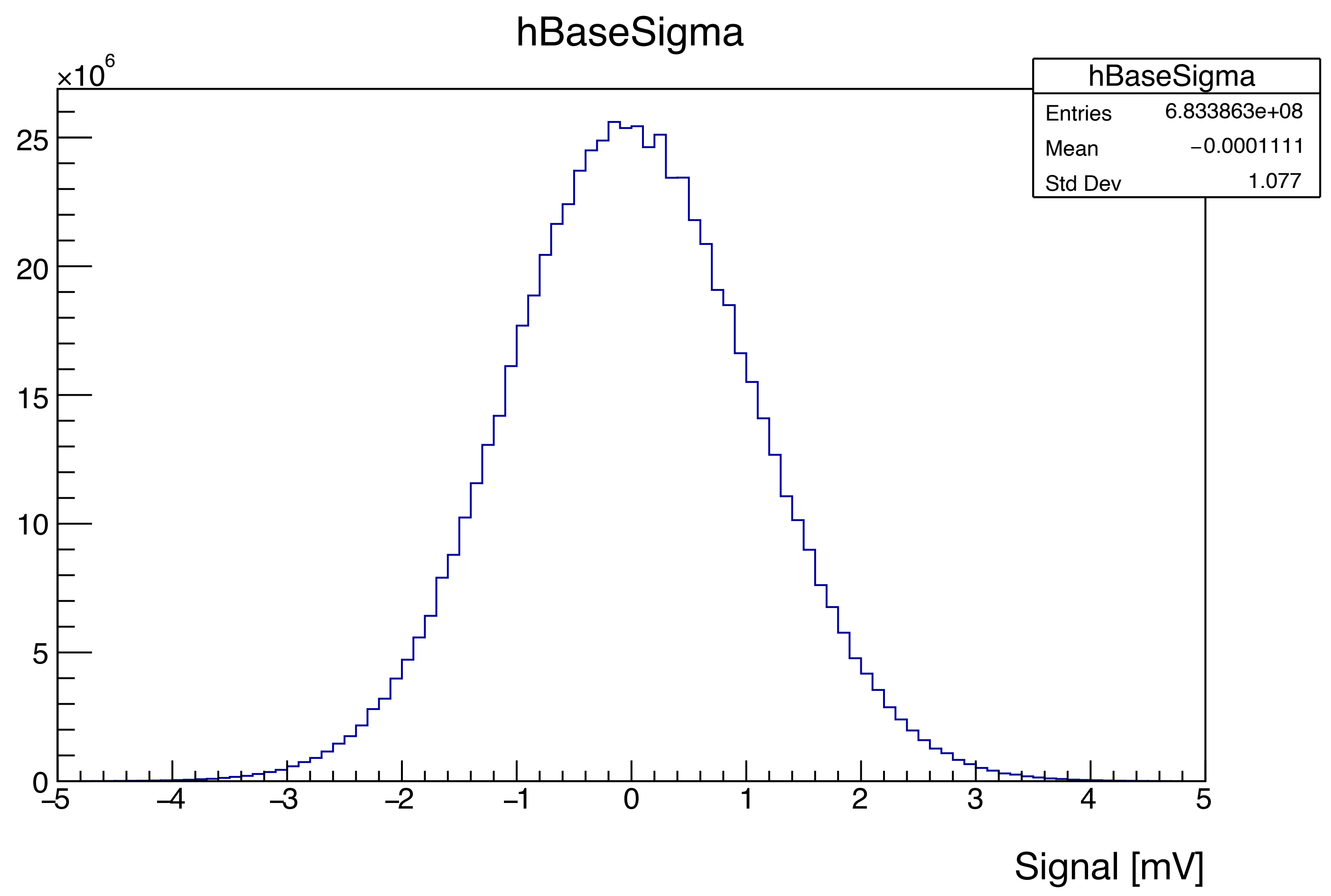
Baseline & Underline

- **Additional check:**
  - Intrinsic signal fluctuation - Make the histogram from each values
    - **Method:** Get differences from the each mean values, fill the histogram from all events.
    - **Baseline sigma:** $\sigma_{\text{baseline}} \approx 1\text{mV}$
    - **Underline:** $\sigma_{\text{underline}} \approx 1\text{mV}$
  - **Uncertainty of signal amplitude:**
    - Signal amplitude is calculated based on the baseline and underline.
    - Uncertainty will be quadrature sum of uncertainties:

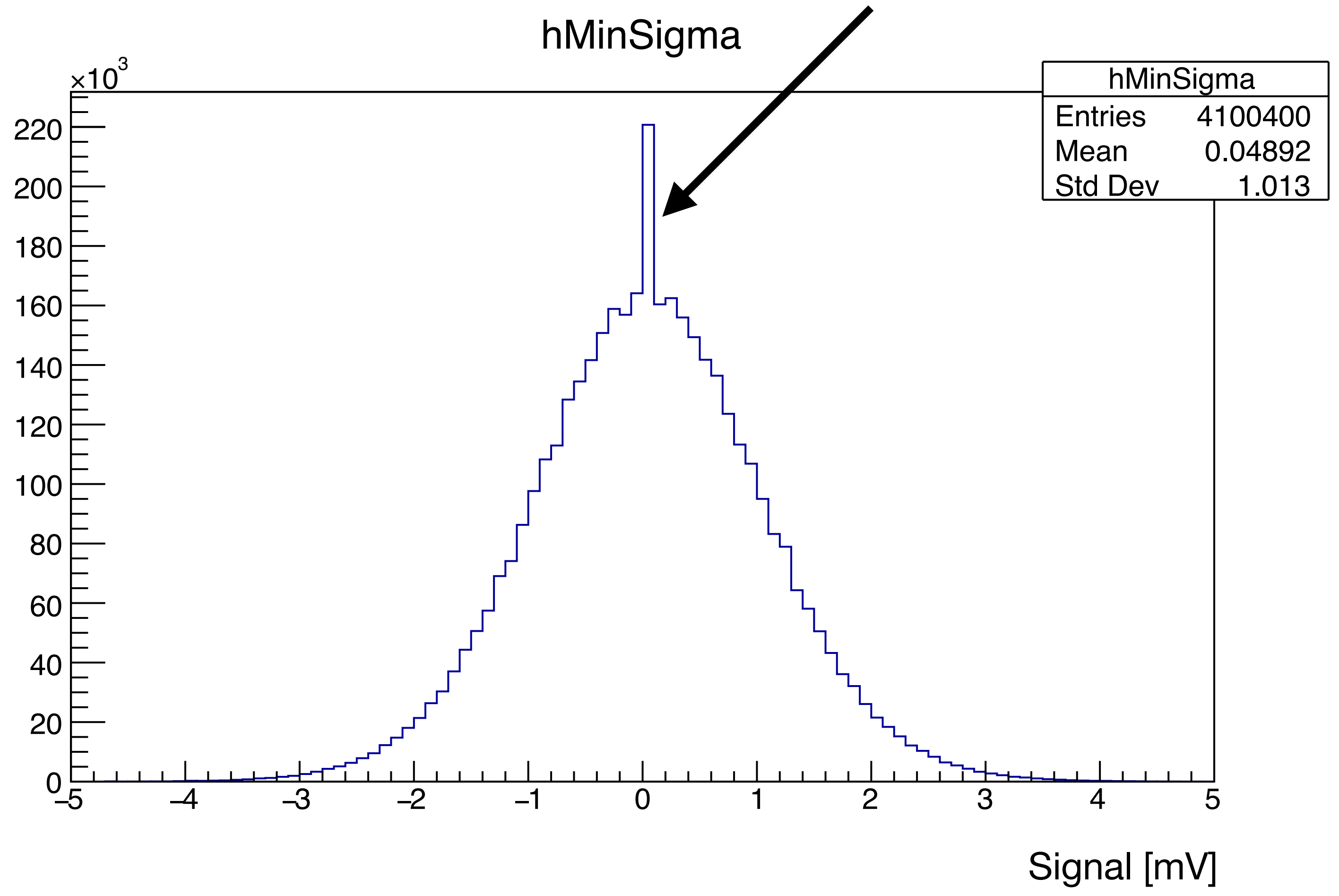
$$\sigma_{\text{Amplitude}} = \sqrt{\sigma^2_{\text{baseline}} + \sigma^2_{\text{underline}}} \approx 1.41\text{mV}$$



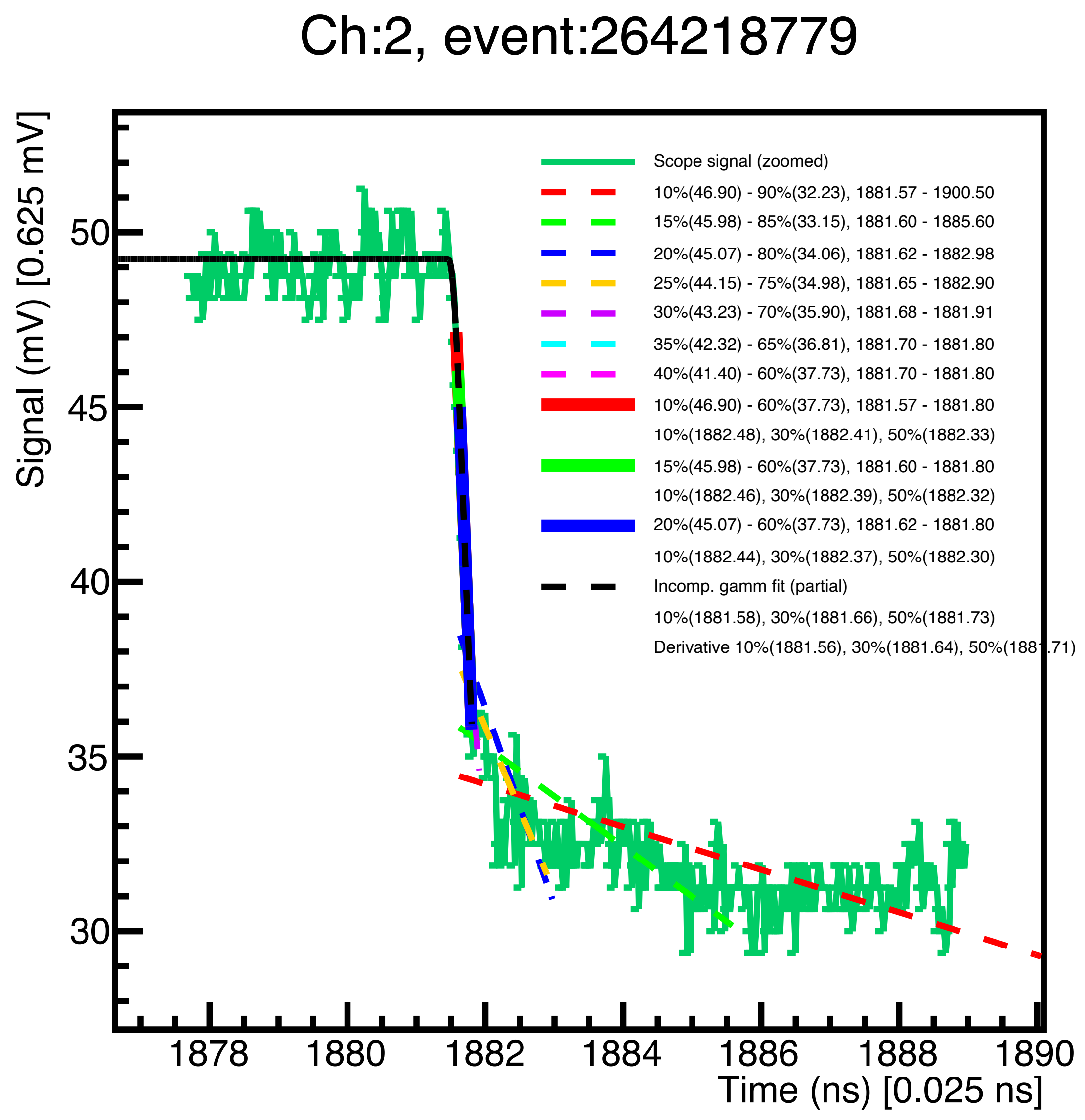
Example of overflow

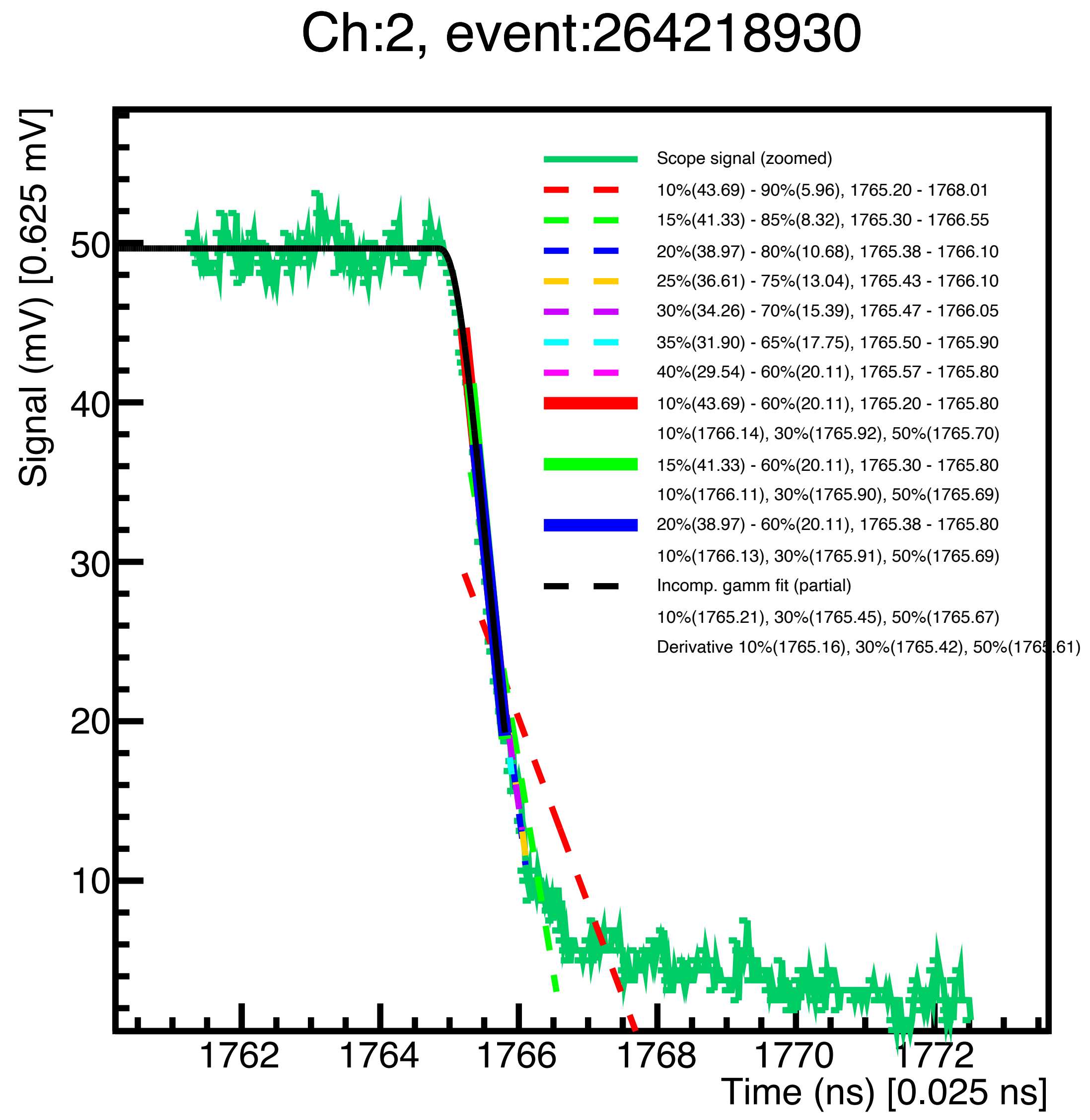


Strange peak around 0 comes from overflow



Signal Examples





Difference from the adaptive amplitude computing