BNL report

Kanato Matsutani

WPI-SKCM2, Hiroshima Univ.

Introduction

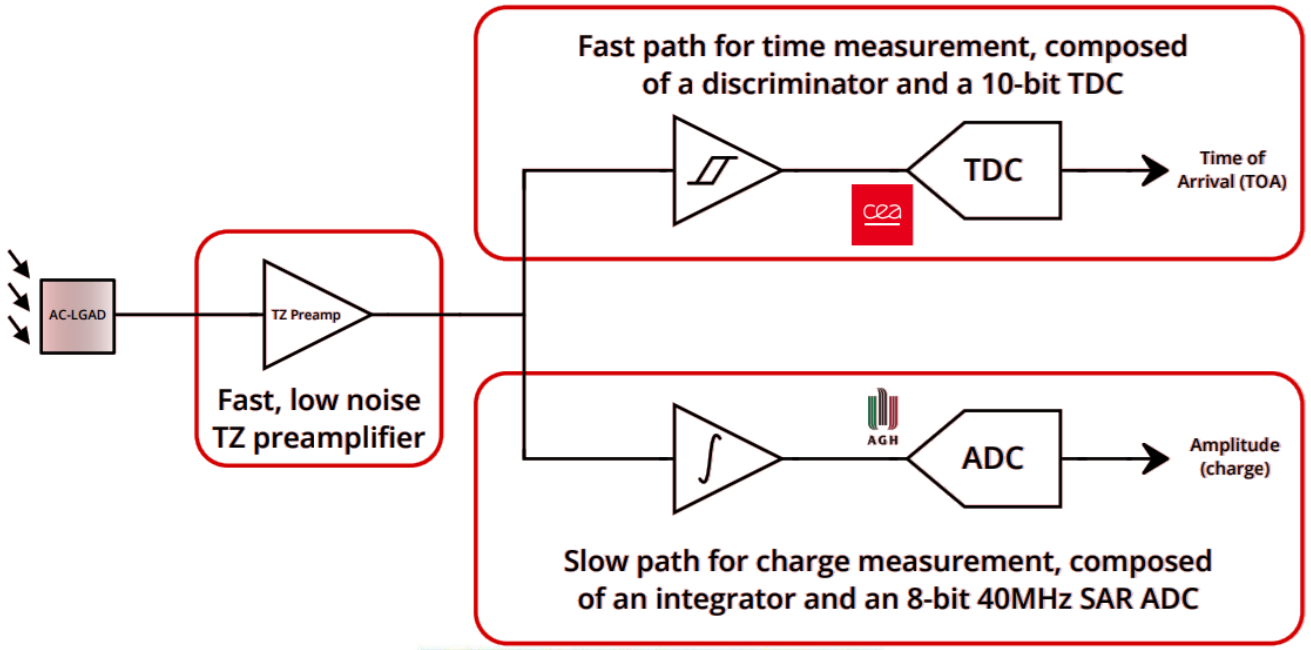
Initial Research (Training, Preparation) for developing an ASIC for Strip-type AC-LGAD (BTOF AC-LGAD)

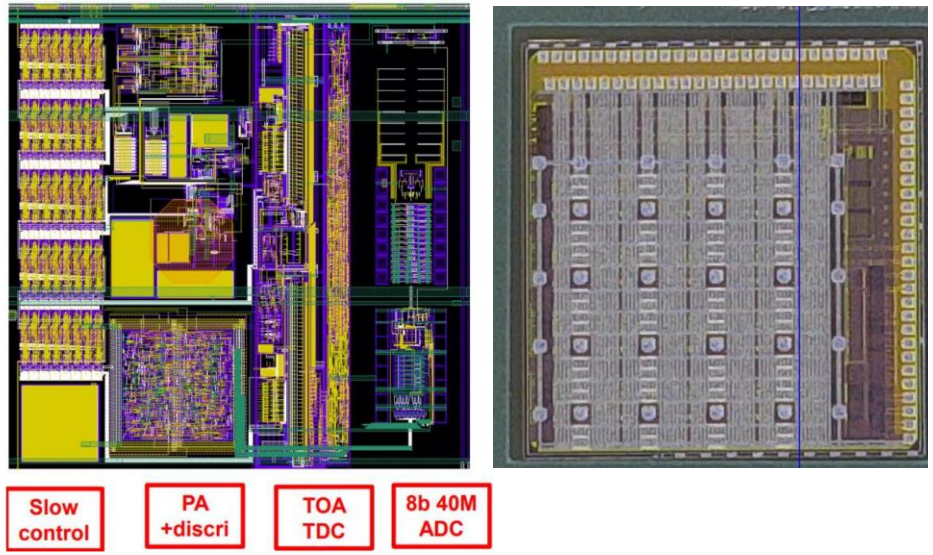
- ◆Motivation
  - ➢**Development study of Barrel TOF in EIC accelerator experiments**
- ◆Purpose
  - ➢Evaluate the performance together with Alessandro, Prithwish, who is developing ASIC+AC-LGAD on site at BNL.
  - ➢Reproduce BNL test environment at Hiroshima University after returning to Japan
- ◆Goal
  - ➢Learn how to evaluate ASIC performance. (Also how to evaluate performance with lasers)
  - ➢Perform and report the performance evaluation of EICROC0 by myself

EICROC0 (ASIC)

- EIC Reed Out Chip ver.0
- 4x4 channel AC-LGAD readout ASIC designed for precise **charge and timing information**
- ~2mW/channel + 4x20mW « analog probe preamp »

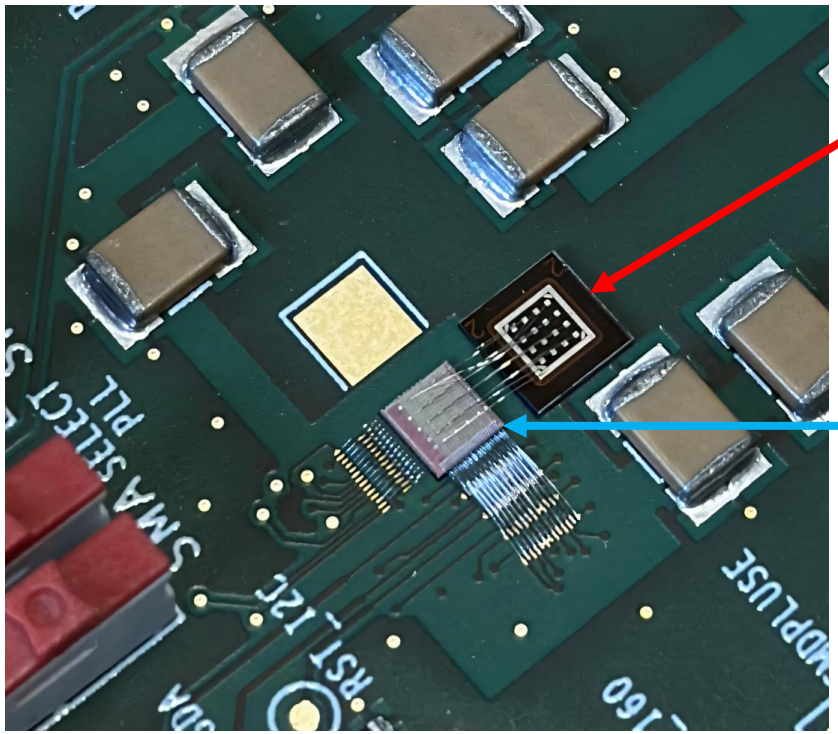
Final EICROC specifications :
- 32x32 pixels / ASIC
- ~1 mW/channel
- Charge detection range 1-50 fC
- Timing jitter ~15-20 ps
- Input capacitance : cD = 1-5 pF



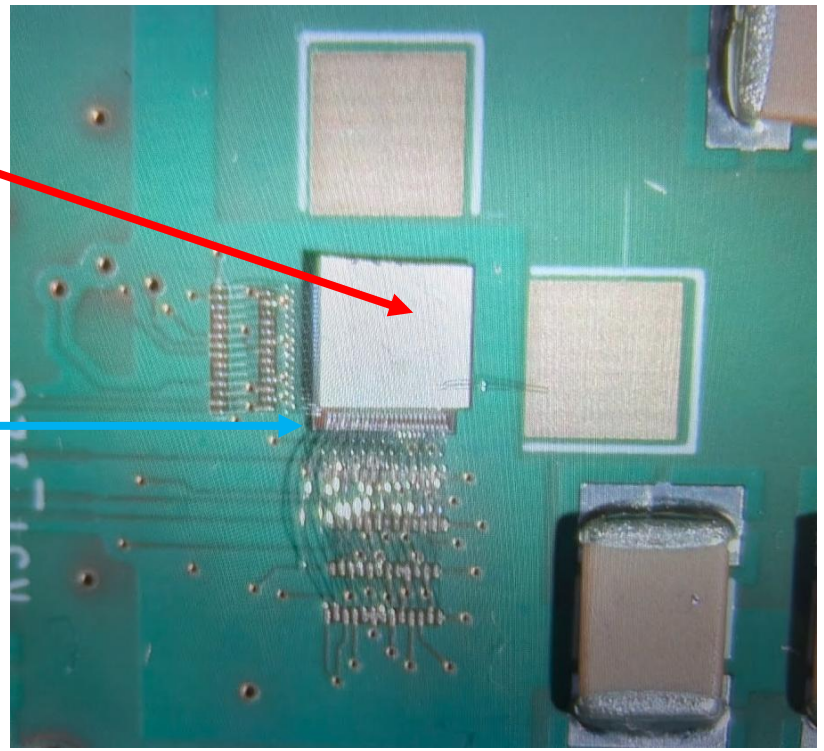


Wire-bonded and Bump-bonded AC-LGAD+EICROC0

◆Wire - bonded (A1 board)



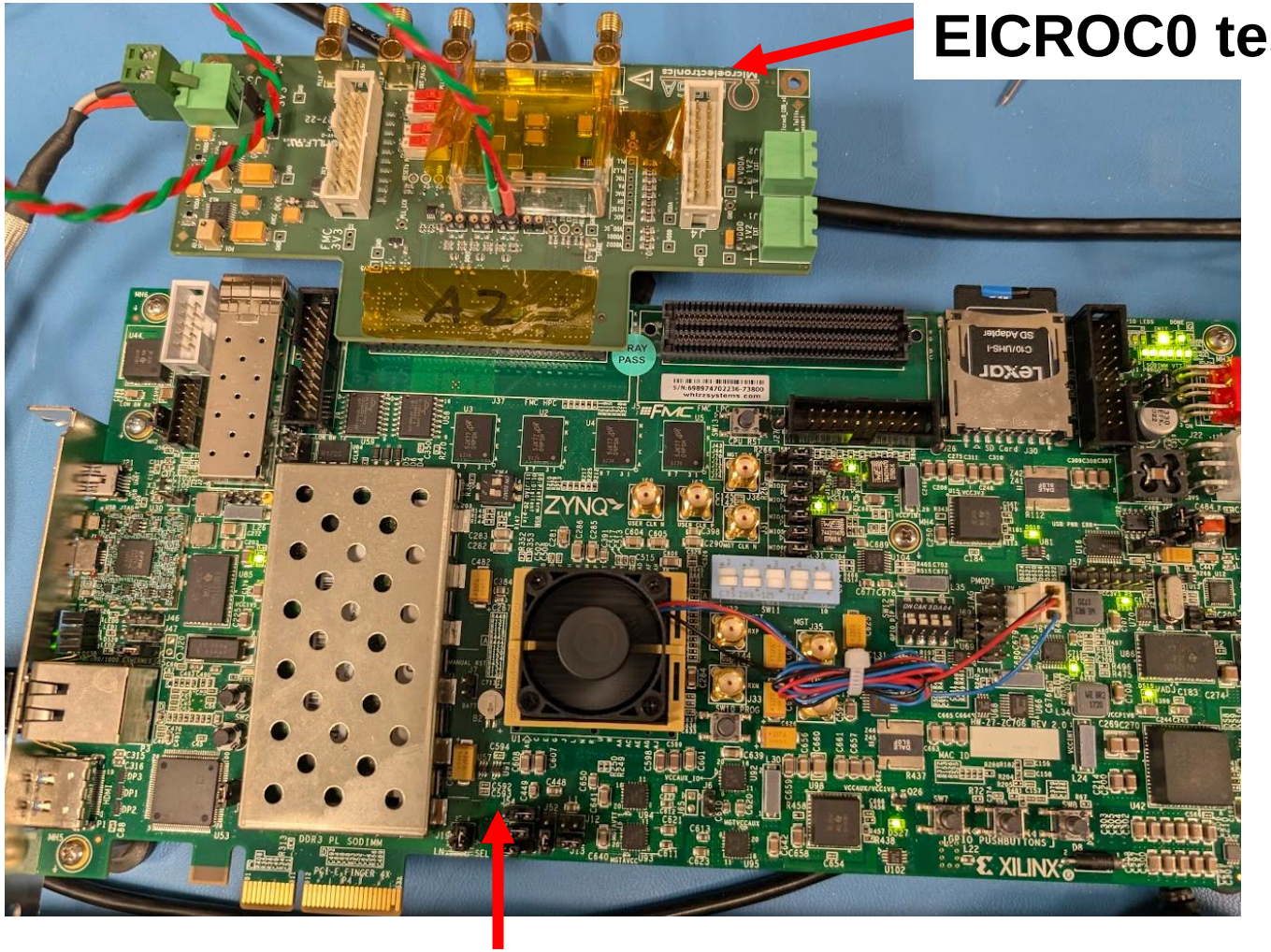
■ Bump – bonded (B1 board)

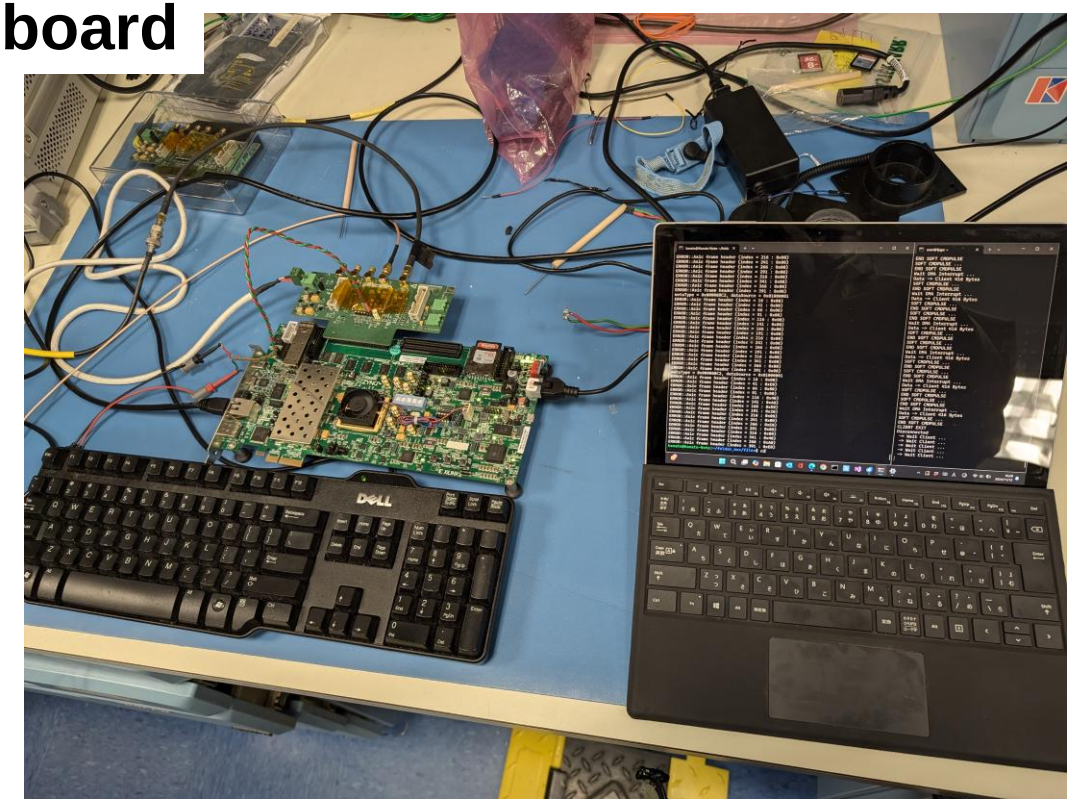


■ No AC-LGAD (A2 board)

The performance difference between the two is not yet known.
This is one of the tests we will be looking into.

Test Bench





- input_data(charge, threshold, injected pixel,etc) and output_data(TDC, ADC, Hitbit) are taken into My_PC using the scp command.
- Analog data is analyzed through an oscilloscope.

Flow chart

◆Data acquisition (on FPGA)
- ➢Select injected channel, charge value
- ➢Threshold scan

↓ Plotting.py (convert .dat file → .csv file)

◆Data check (on PC)
- ➢referenceValue.csv
- ➢threshold sweep(wide range)

↓ Find an optimal threshold for each charge

◆Data acquisition (on FPGA)
- ➢For analysis, fixed threshold, 10000events

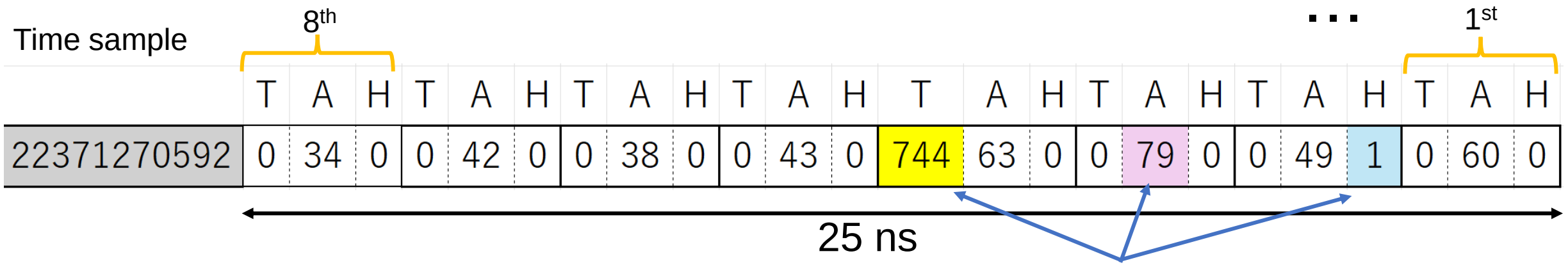
↓ Once you have the data for analysis…

◆Data Analysis (on PC)
- ➢See TDC and ADC distribution
- ➢Calculate jitter
- ➢Data construction

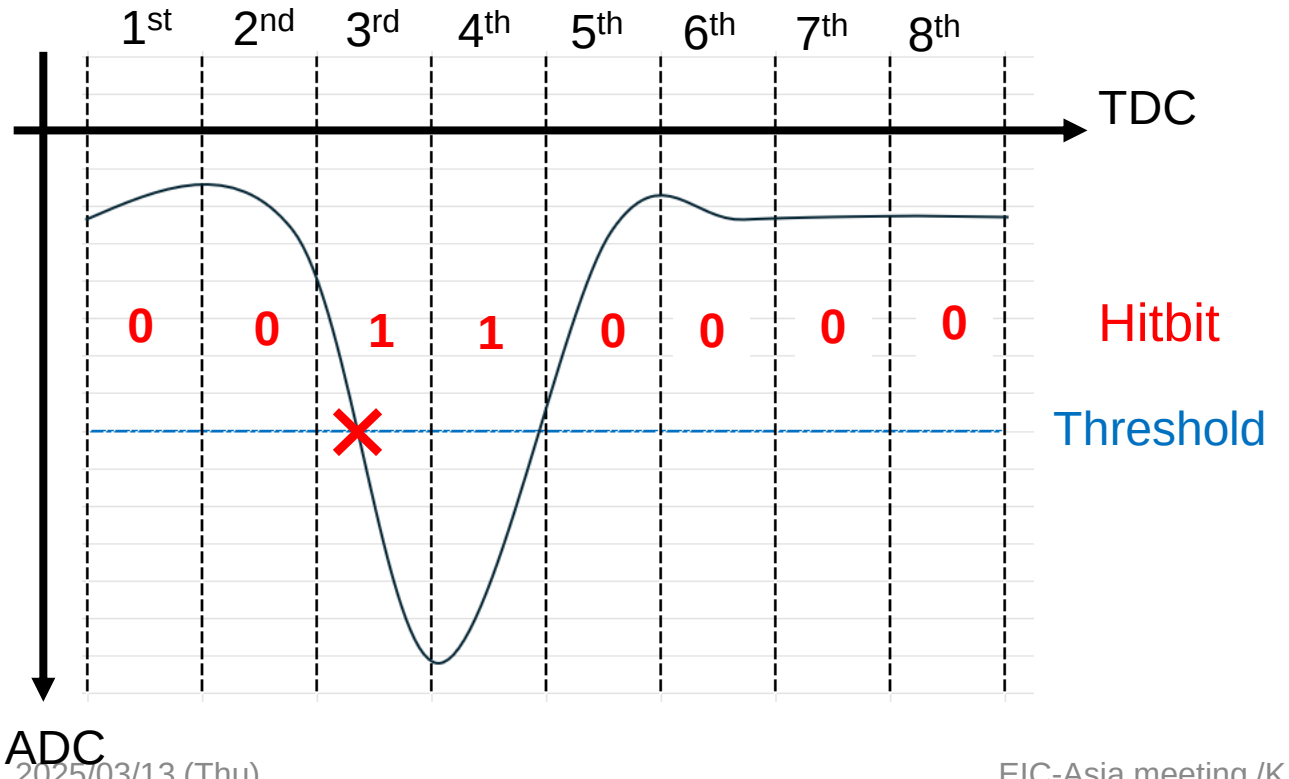
My Codes

We use a new farmwear made by Dominiqu, Adrien & team (Thanks a lot)

Data Construction



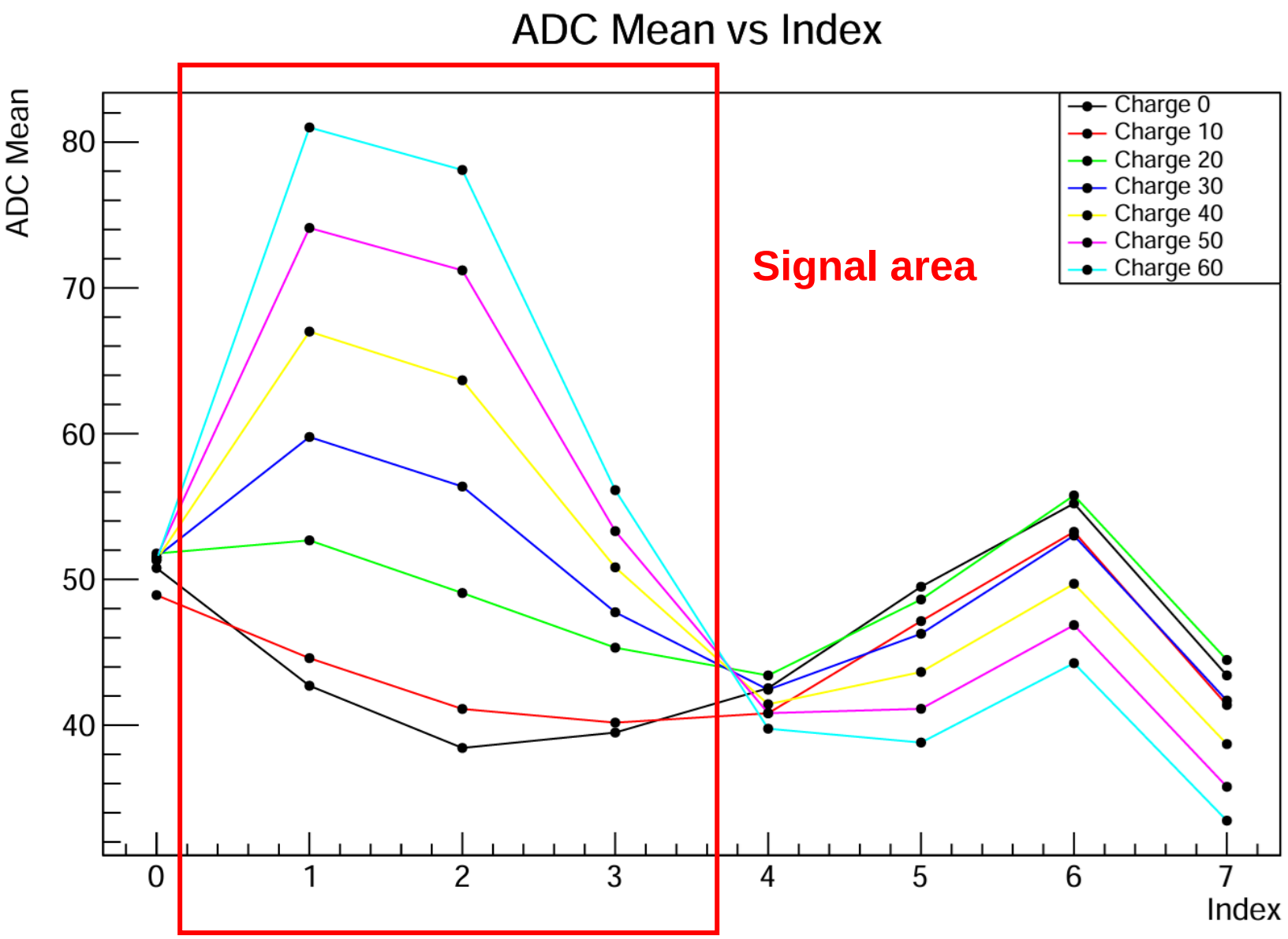
Max TDC, Max ADC, Hitbit
Output in different time sample



TDC = 10 bit (0 – 1023) , 1 unit = 25 / 1024 ns
ADC = 8 bit (0 – 255) , 40MHz, Max ADC for each time sample,

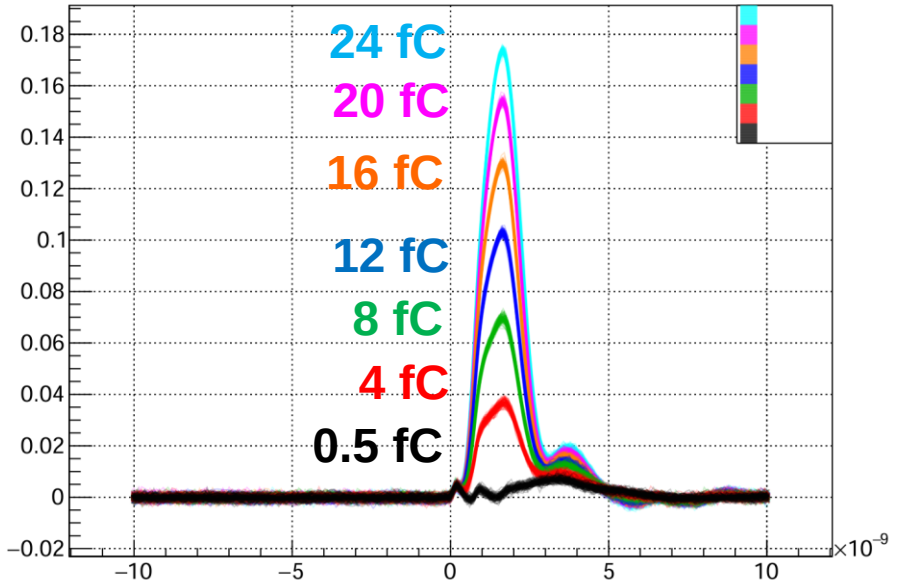
1 event = 8 time-samples
= 25 ns

ADC -> Waveform



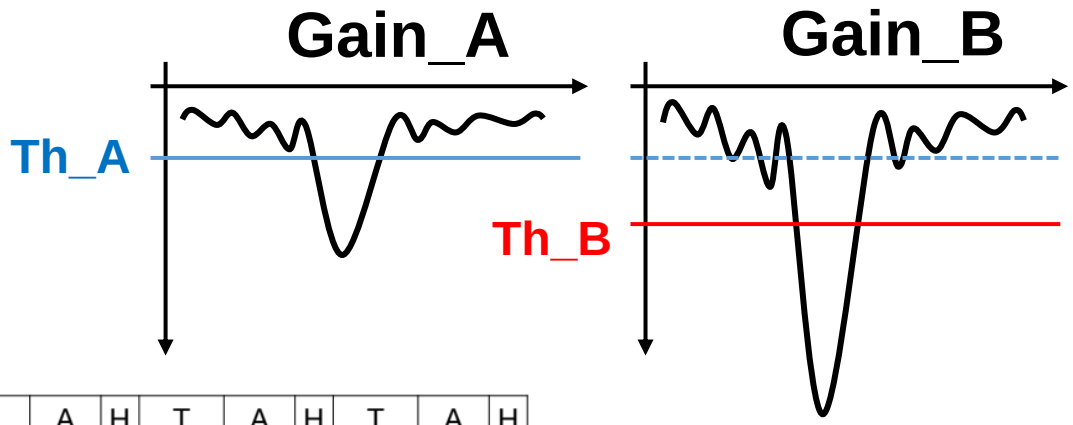
A2 (No sensor) board
(0,0) pixel
Injected charge

Analog waveforms (not same test)



TDC ADC Hitbit

◆Every channel has a unique gain
→ Unique threshold for every channel



Data Format: {TDC, ADC, HB} x 8

Th=70 DAC **Low threshold**

Event #	T	A	H	T	A	H	T	A	H	T	A	H	T	A	H	T	A	H	T	A	H	T	A	H
43125864692	525	88	1	469	88	1	526	88	1	0	88	0	516	88	1	0	88	0	0	88	1	1024	88	0

Th=160 DAC **Optimal threshold**

Event #	T	A	H	T	A	H	T	A	H	T	A	H	T	A	H	T	A	H	T	A	H	T	A	H
52838449618	0	88	0	0	88	0	0	88	0	157	88	0	0	88	0	0	88	1	0	88	0	0	88	0

Th=220 DAC **High threshold**

Event #	T	A	H	T	A	H	T	A	H	T	A	H	T	A	H	T	A	H	T	A	H	T	A	H
62320292269	0	88	0	0	88	0	0	88	0	0	88	0	0	88	0	0	88	0	0	88	0	0	88	0

Analysis Code

```
while (std::getline(infile, line)) {
    std::istringstream iss(line);
    std::vector<long long> values;
    std::string value;

    pixel_number = event_counter % 16;      //Define a pixel number from data structure (16 lines per event)

    try {
        for (int i = 0; i < 25; i++) {
            if (std::getline(iss, value, ';')) {
                values.push_back(std::stoll(value));    //The data includes {event, (TDC, ADC, HitBit) x8} (total 25 cells in 1 line)
            }
        }
    } catch (const std::exception& e) {
        std::cerr << "Error: Failed to parse line: " << line << std::endl;
        continue;
    }

    if (values.size() >= 24) {
        std::vector<long long> tdc, adc, hitbit;

        for (int i = 0; i < 8; i++) {
            tdc.push_back(values[3 * i + 1]);
            adc.push_back(values[3 * i + 2]);
            hitbit.push_back(values[3 * i + 3]);
        }

        long long hitbit_sum = std::accumulate(hitbit.begin(), hitbit.end(), 0LL);
        const long long tdc_min = 10;
        const long long tdc_max = 1000;

        if (pixel_number == PN && !hitbit.empty() && hitbit_sum >= 1 && hitbit_sum <= 3 && !tdc.empty() && !adc.empty()) {
            //pixel_number(0 - 15) , hitbit_sum = 1 (very clear signal), hitbit >= 1 (without No signal)

            //long long max_tdc_value = *std::max_element(tdc.begin(), tdc.end());
            long long max_tdc_value = tdc[5];

            //long long max_adc_value = *std::max_element(adc.begin(), adc.end());
            long long max_adc_value = adc[6];                                   // You should select this index after checking the distribution for each time sample !!

            if (max_tdc_value >= tdc_min && max_tdc_value < tdc_max) {
                results.push_back({max_tdc_value, max_adc_value});
            }
```

Data import part

Separating data into TDC, ADC, and Hitbit part

Data selecting part

Data analysis

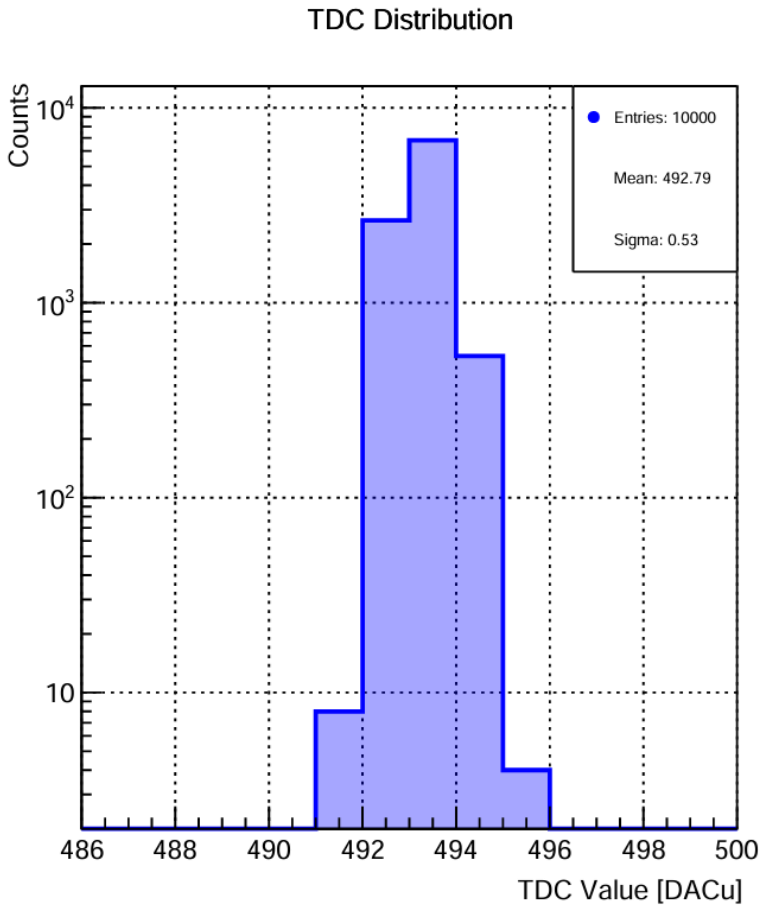
1. TDC distribution and ADC distribution

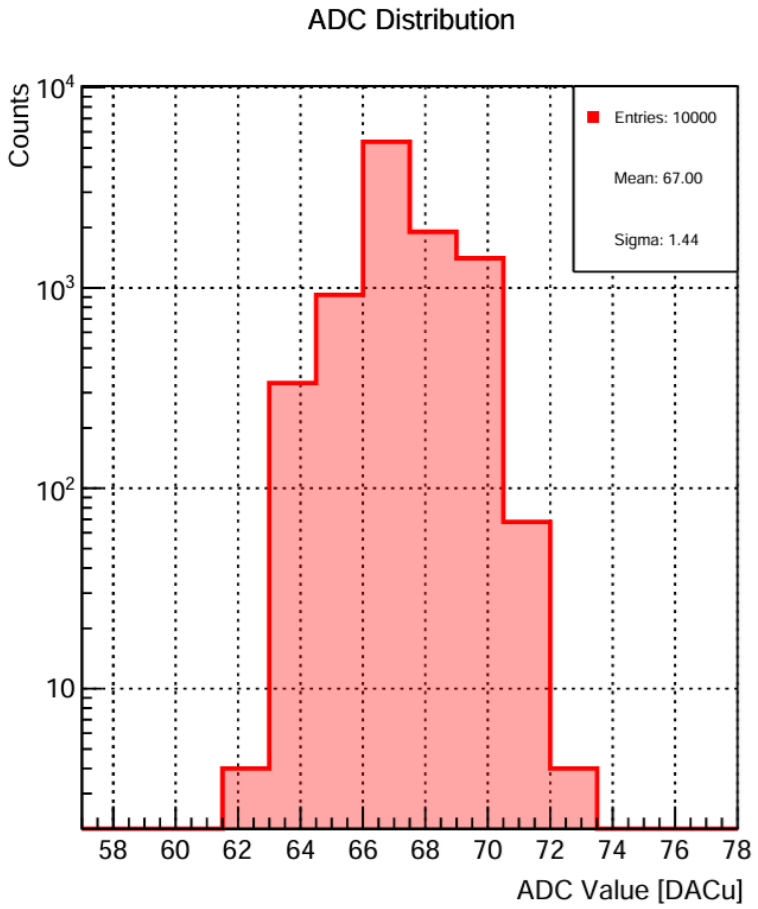
- ✓ charge DAC = [0,10, 20, 30, 40, 50, 60] (0.5 fC – 24 fC)
  charge value= [0.5, 4, 8, 12, 16, 20, 24] fC
- ✓ threshold DAC= [250, 280, 310, 340, 370, 400, 430]
  (I decided the values to look at threshold_sweep)
- ✓ TDC range = 1 – 1023
- ✓ 10000 events for each charge
- ✓ A2 board (No sensor) and B1 board (Bump-bonded)

2. TDC_sigma and ADC_mean trend with error

(0,0) pixel Analysis
charge is injected to a pixel (0,0)

TDC, ADC distributions (for example)





Event data: input charge = 16 fC
threshold = 370 DACu
N_ event = 10000
Injected pixel = [0,0]

Results

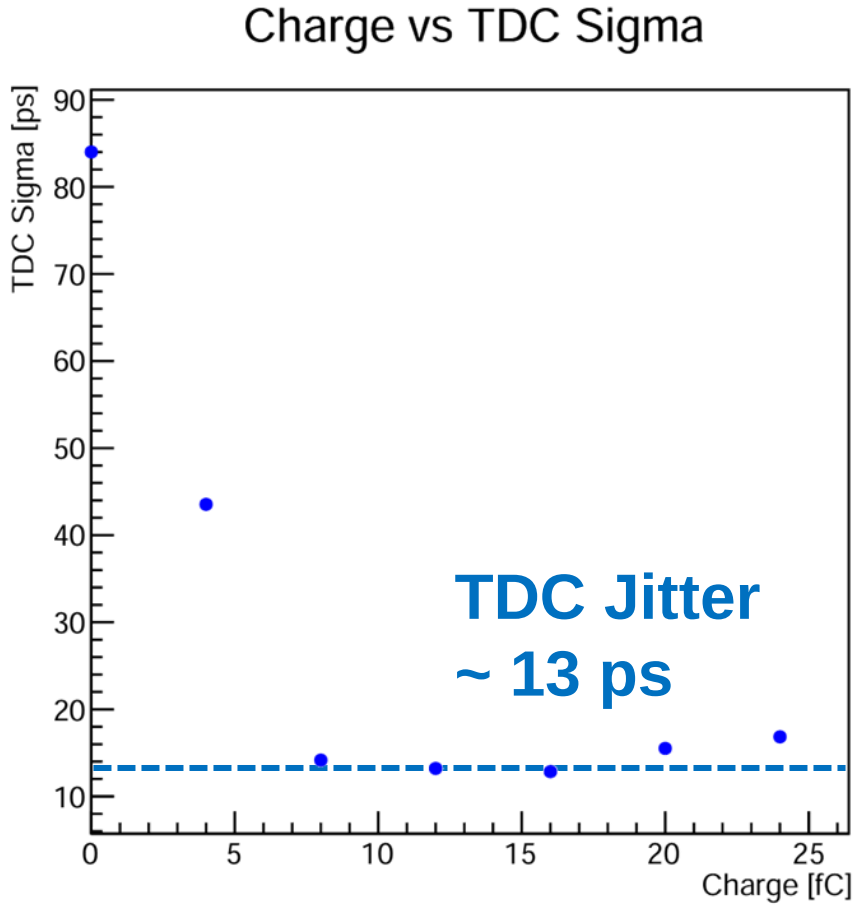
Entries = 10000

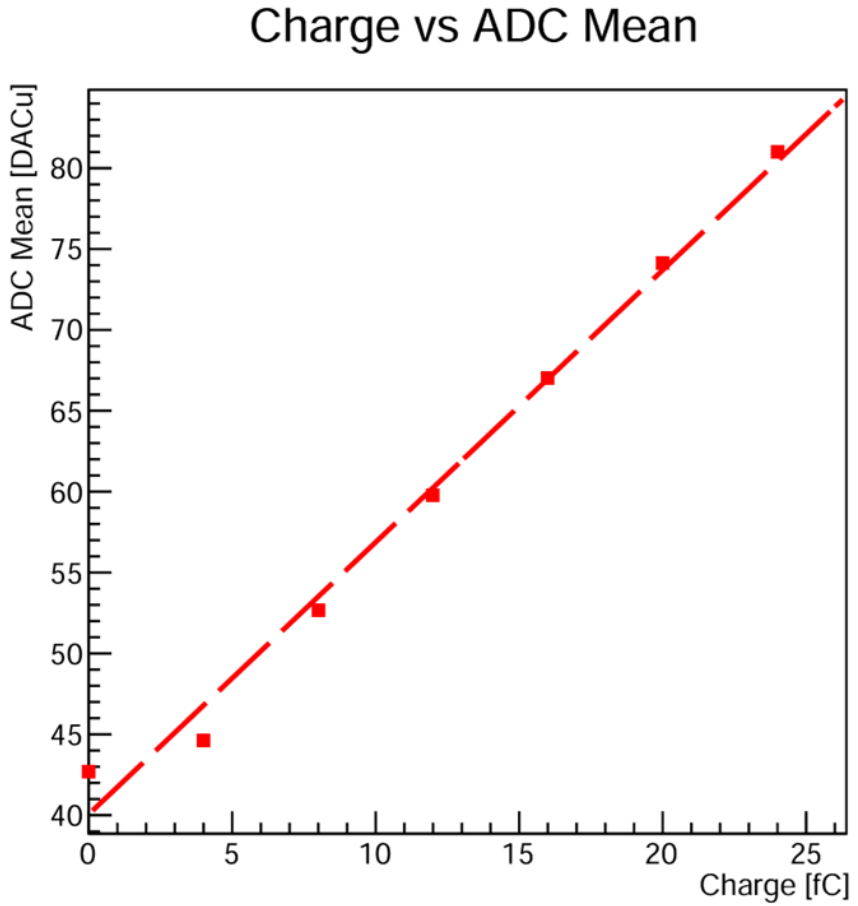
TDC Sigma = 0.53 [DACu]

~ 13 [ps]

ADC Mean = 67.00

TDC_sigma and ADC_mean trends ([0,0] pixel, without overflow)



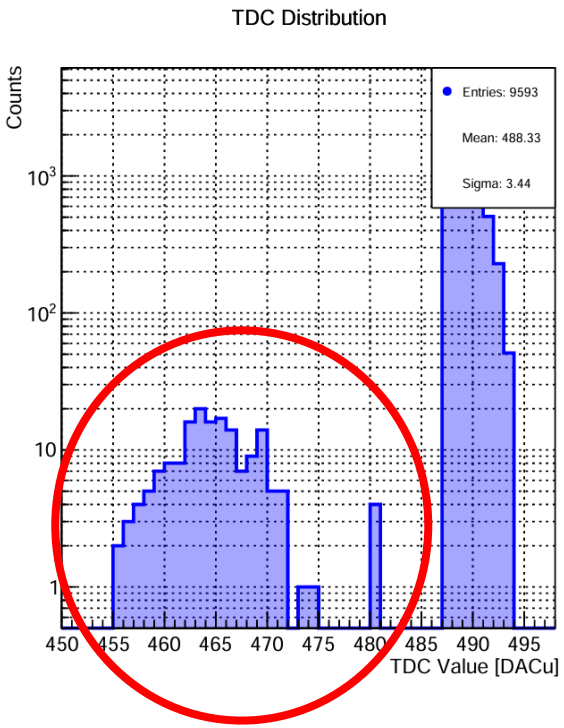


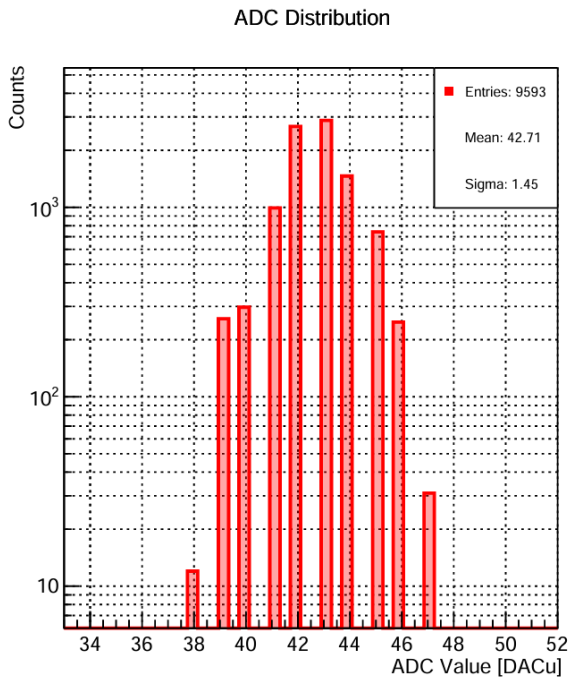
Sigma tended to take a stable value with a smaller sigma for larger charge value.

Positive proportional relationship between the amount of injected charge and the digital signal.

TDC ADC distributions

Q = 0

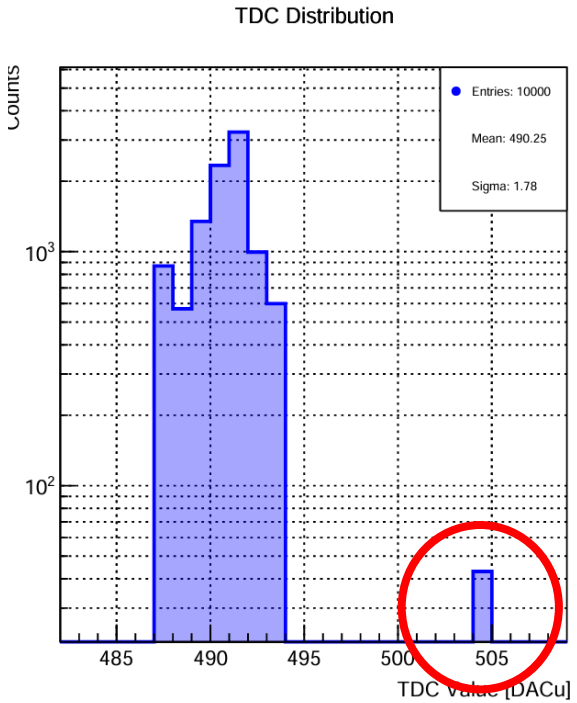


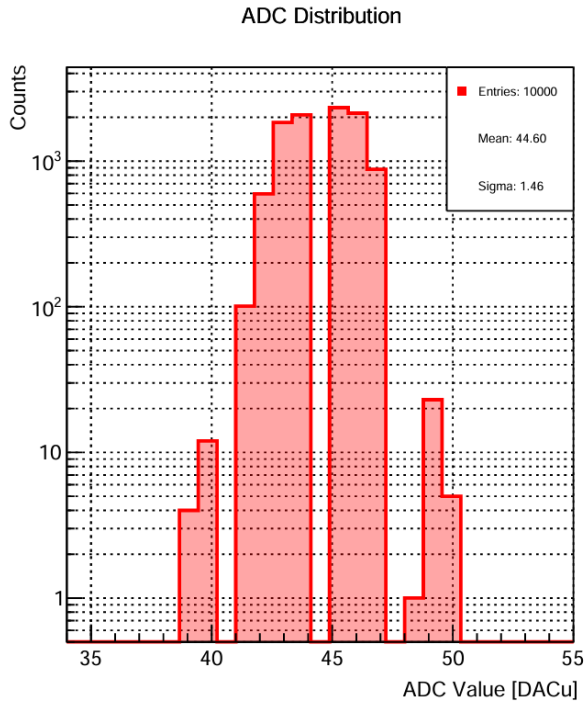


N_event: 10000
Entries: 9593
Mean: 488.33
Sigma(StdDev) : 3.44

N_event: 10000
Entries: 9593
Mean: 42.71
Sigma(StdDev) : 1.45

Q = 10



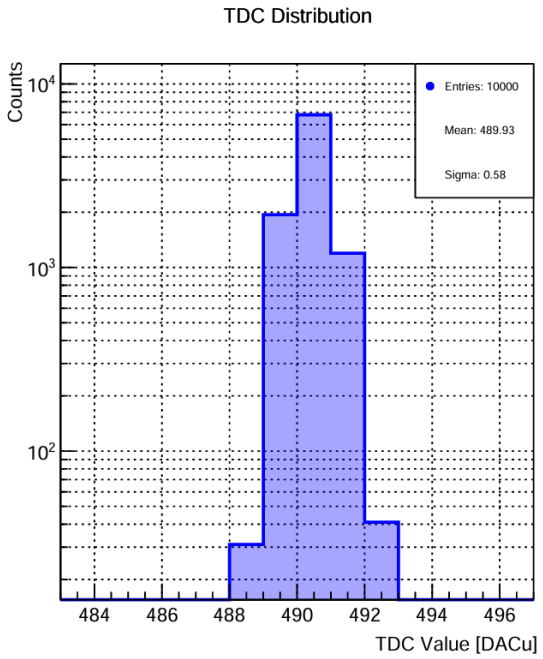


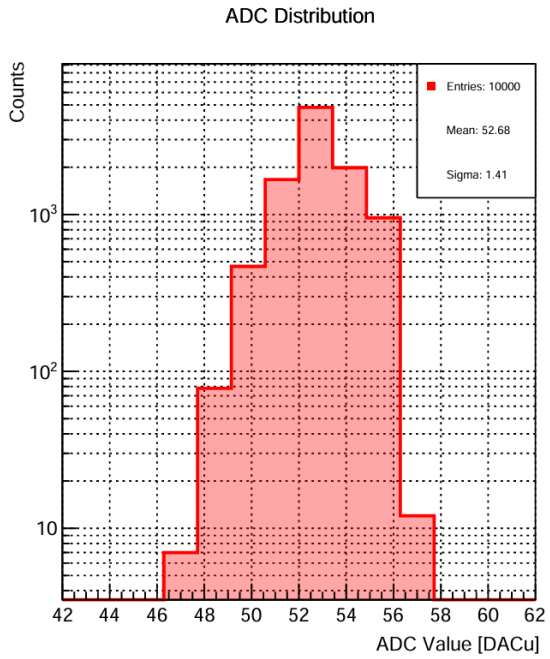
N_event: 10000
Entries: 10000
Mean: 490.25
Sigma(StdDev) : 1.78

N_event: 10000
Entries: 10000
Mean: 44.60
Sigma(StdDev) : 1.46

TDC ADC distributions

Q = 20

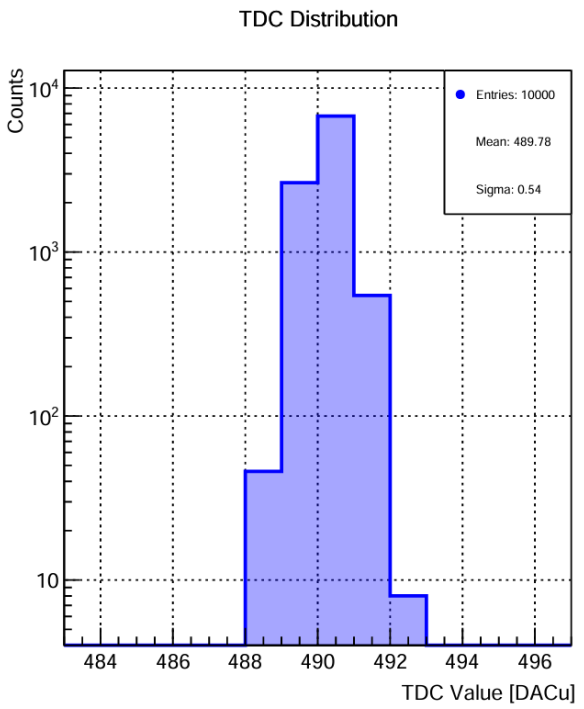


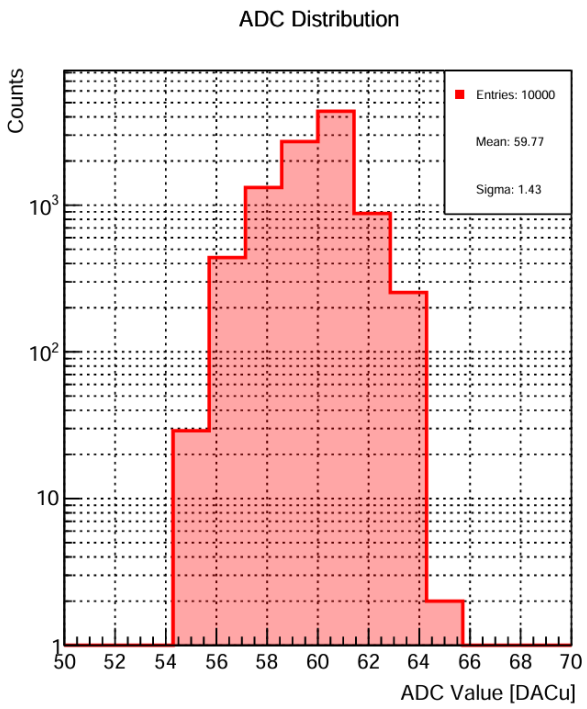


N_event: 10000
Entries: 10000
Mean: 489.93
Sigma(StdDev) : 0.58

N_event: 10000
Entries: 10000
Mean: 52.68
Sigma(StdDev) : 1.45

Q = 30



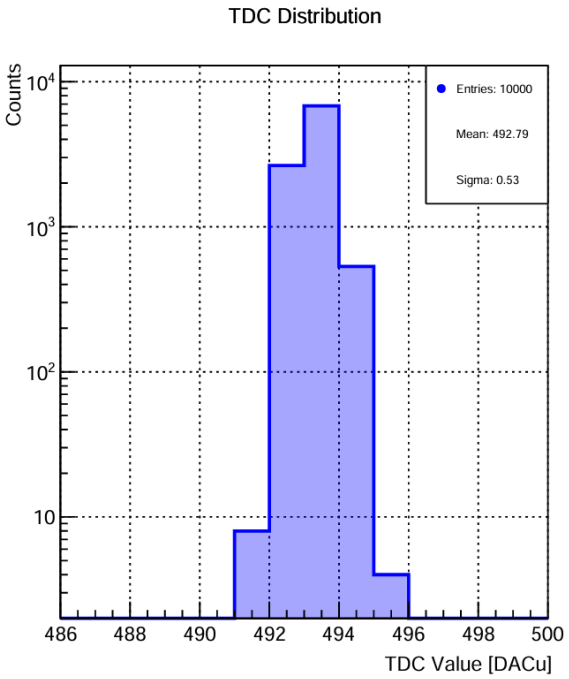


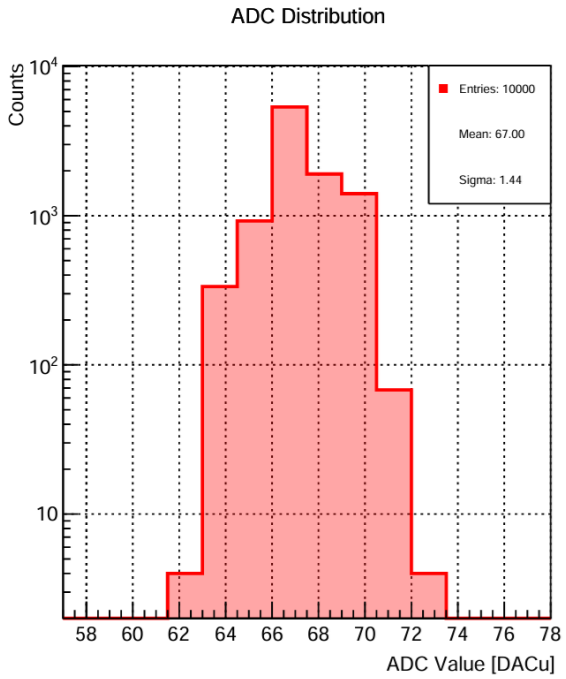
N_event: 10000
Entries: 10000
Mean: 489.78
Sigma(StdDev) : 0.54

N_event: 10000
Entries: 10000
Mean: 59.77
Sigma(StdDev) : 1.43

TDC ADC distributions

Q = 40

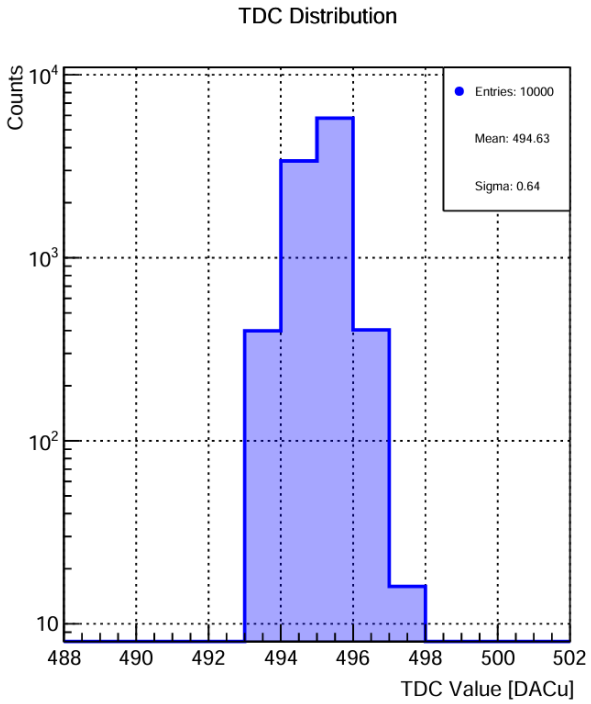


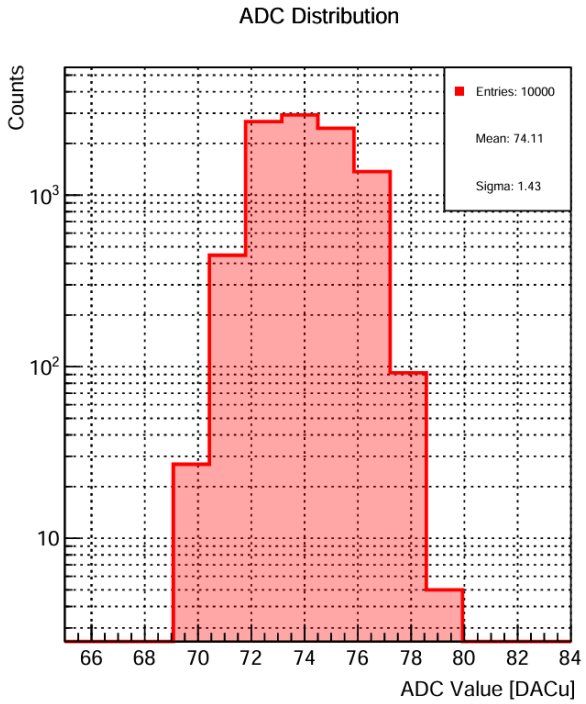


N_event: 10000
Entries: 10000
Mean: 492.79
Sigma(StdDev) : 0.53

N_event: 10000
Entries: 10000
Mean: 67.00
Sigma(StdDev) : 1.44

Q = 50



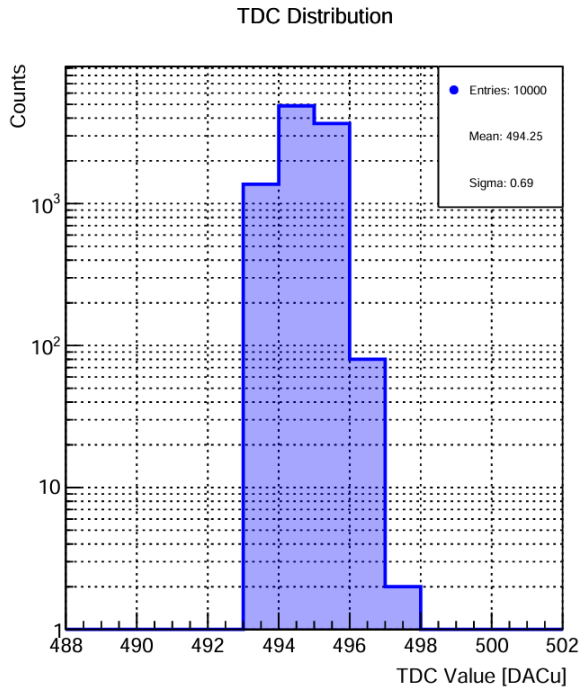


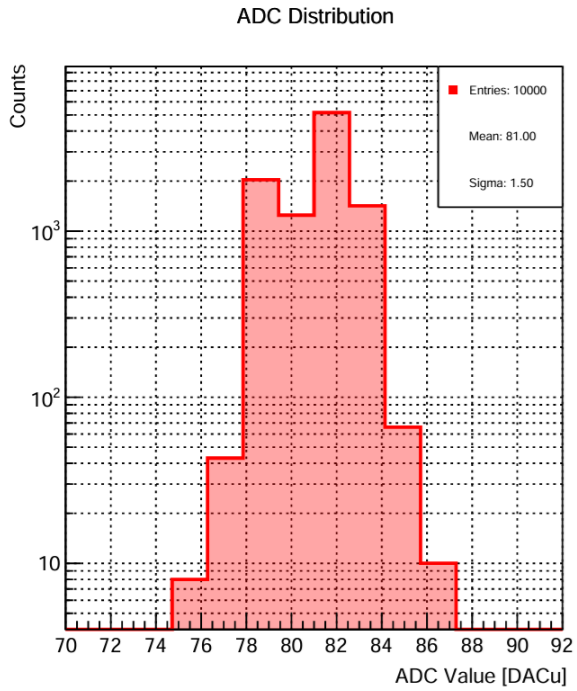
N_event: 10000
Entries: 10000
Mean: 494.63
Sigma(StdDev) : 0.64

N_event: 10000
Entries: 10000
Mean: 74.11
Sigma(StdDev) : 1.43

TDC ADC distributions of EICROC0 (ASIC)

Q = 60

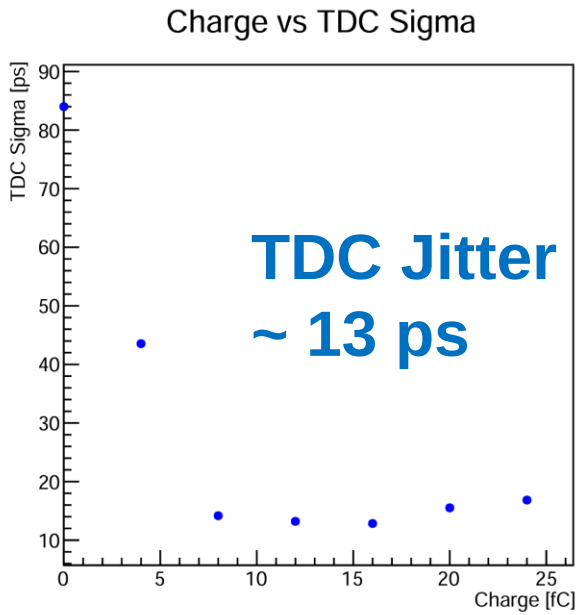




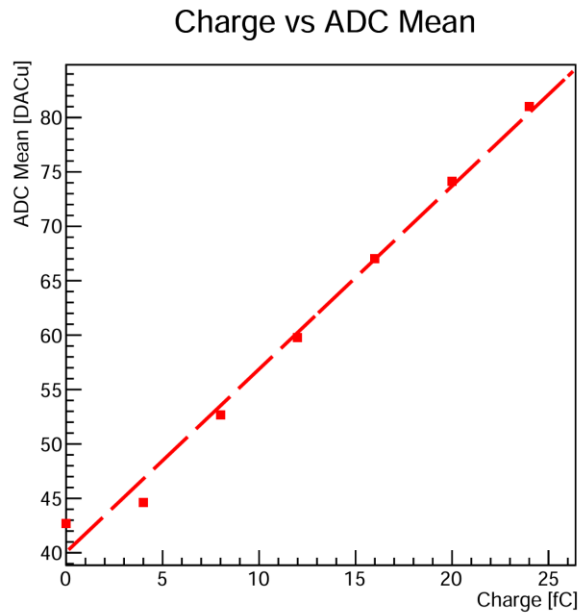
N_event: 10000
Entries: 10000
Mean: 494.25
Sigma(StdDev) : 0.69

N_event: 10000
Entries: 10000
Mean: 81.00
Sigma(StdDev) : 1.50

Charge vs TDC_sigma and ADC_mean



TDC Jitter
~ 13 ps



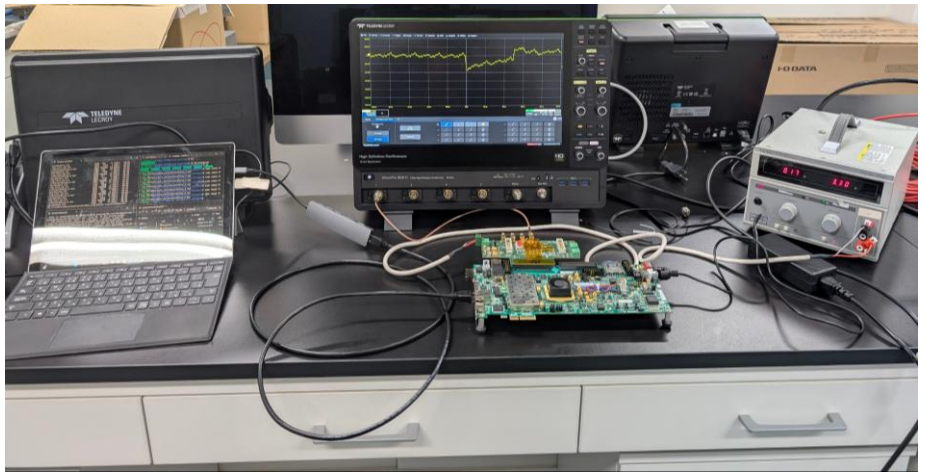
TDC info.
~ 24.4 ps/ 1 unit

ADC info.
? mV / 1 unit
~ (3.75 x DACu – 130) mV

Charge injection info.
~ 0.4 fC / 1DACu, Q = 0.5 – 25 fC

Summary

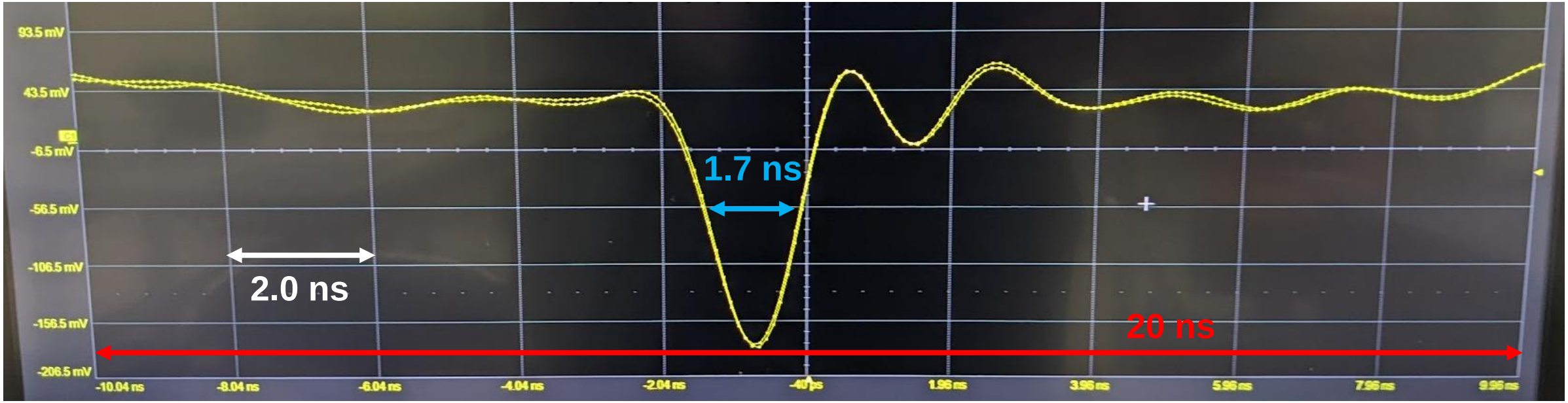
- **I analyzed the digital signal of EICROC0 by charge injection.**
  - ✓ TDC sigma tended to take a stable value with a smaller sigma(~13 ps) for higher charge value.
  - ✓ ADC distribution appeared to have a positive proportional relationship between the amount of injected charge and the digital signal.
- **Next to do in HU**
  - ✓ Replicate what we did at BNL.
  - ✓ AC-LGAD + EICROC0(wire-bonded) analysis with radiation source
  - ✓ Install the laser system in HU this year.



Back up

Continuity of Time sample

◆Analog Signal of EICROC0



Hitbit can only be 1 for at most 2 consecutive time samples.

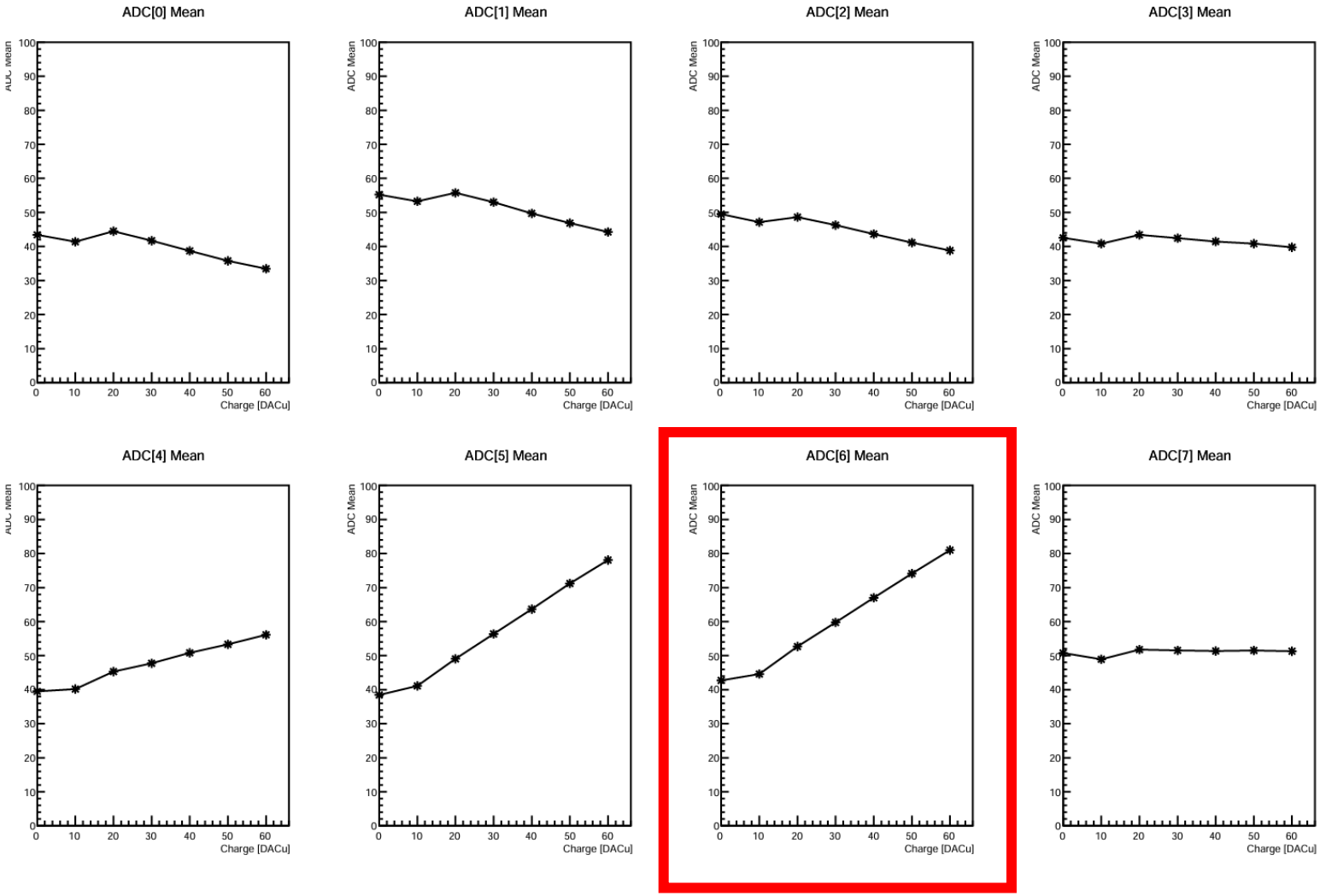
(I will check the digital data of this analog signal)

ADC_mean trend for each time samples

	ADC[0]			ADC[1]			ADC[2]			ADC[3]			ADC[4]			ADC[5]			ADC[6]			ADC[7]		
Q = 0																								
15909746629	0	44	0	0	56	0	0	51	0	0	44	0	0	39	0	487	39	0	0	44	0	0	51	1
Q = 10																								
25131773979	0	42	0	0	54	0	0	47	0	0	39	0	0	39	0	490	41	0	0	45	0	0	50	1
Q = 20																								
53036210291	0	42	0	0	55	0	0	49	0	0	44	0	0	45	0	490	48	0	0	53	0	0	52	1
⋮																								
Q = 60																								
79678721975	0	34	0	0	44	0	0	39	0	0	42	0	0	59	0	494	79	0	0	80	0	0	50	1

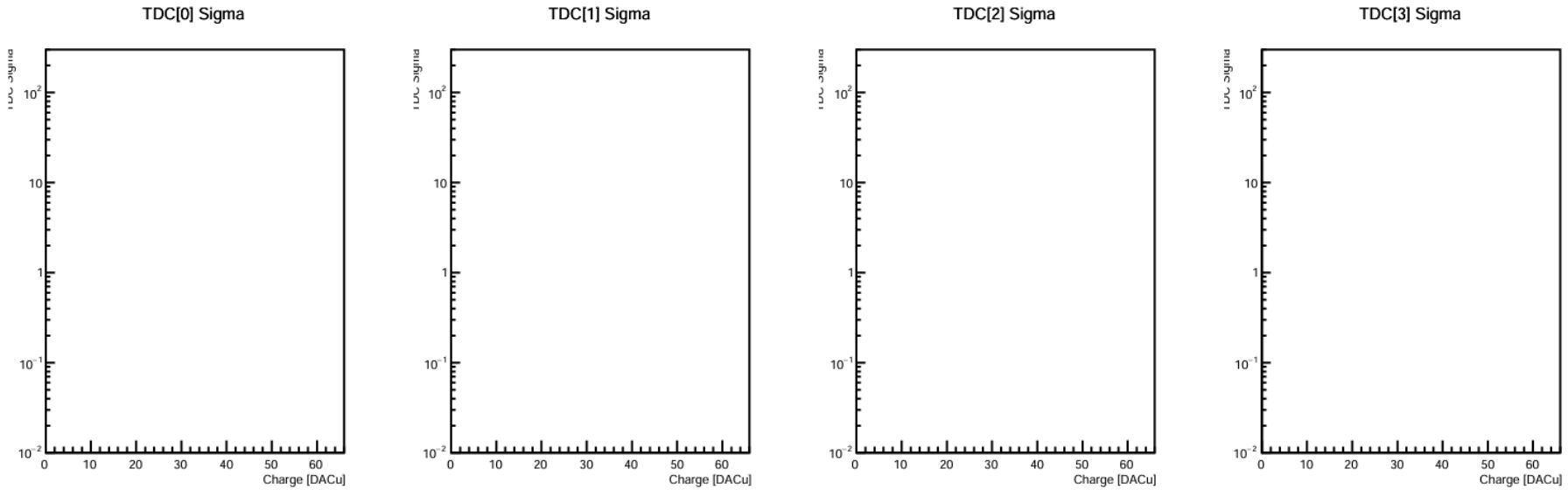
TDC has only one value, so we can extract it. However, ADC has a value in every time sample, so it is necessary to discuss which value should be obtained.

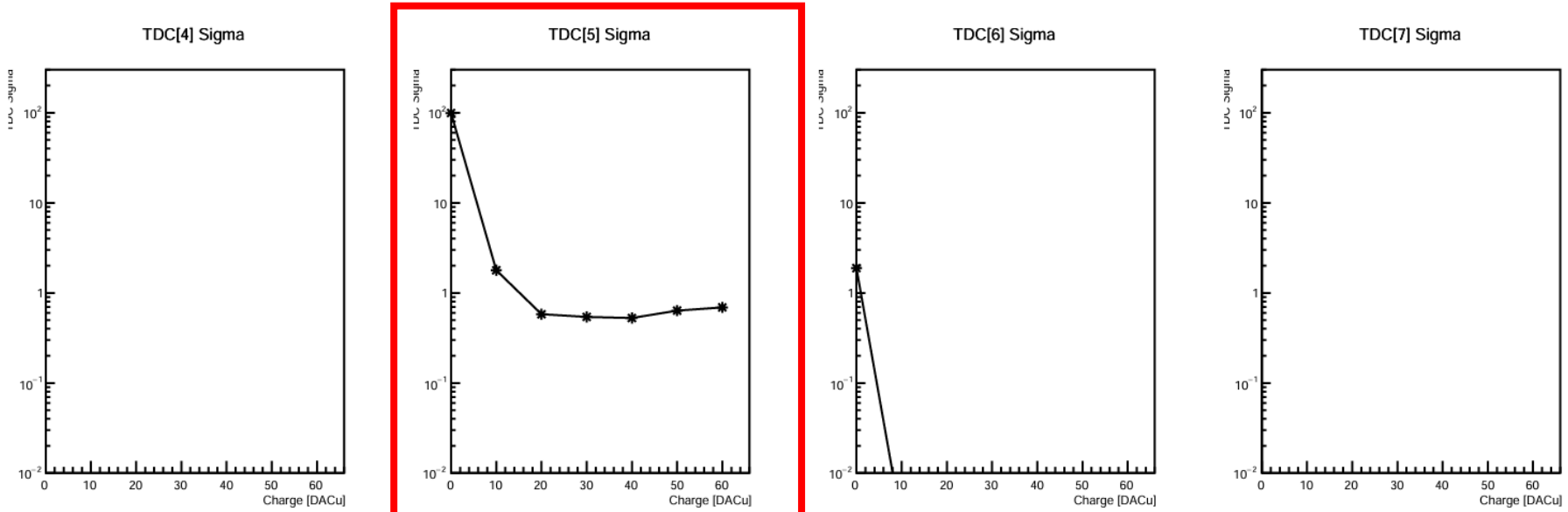
ADC_mean trend for each time samples

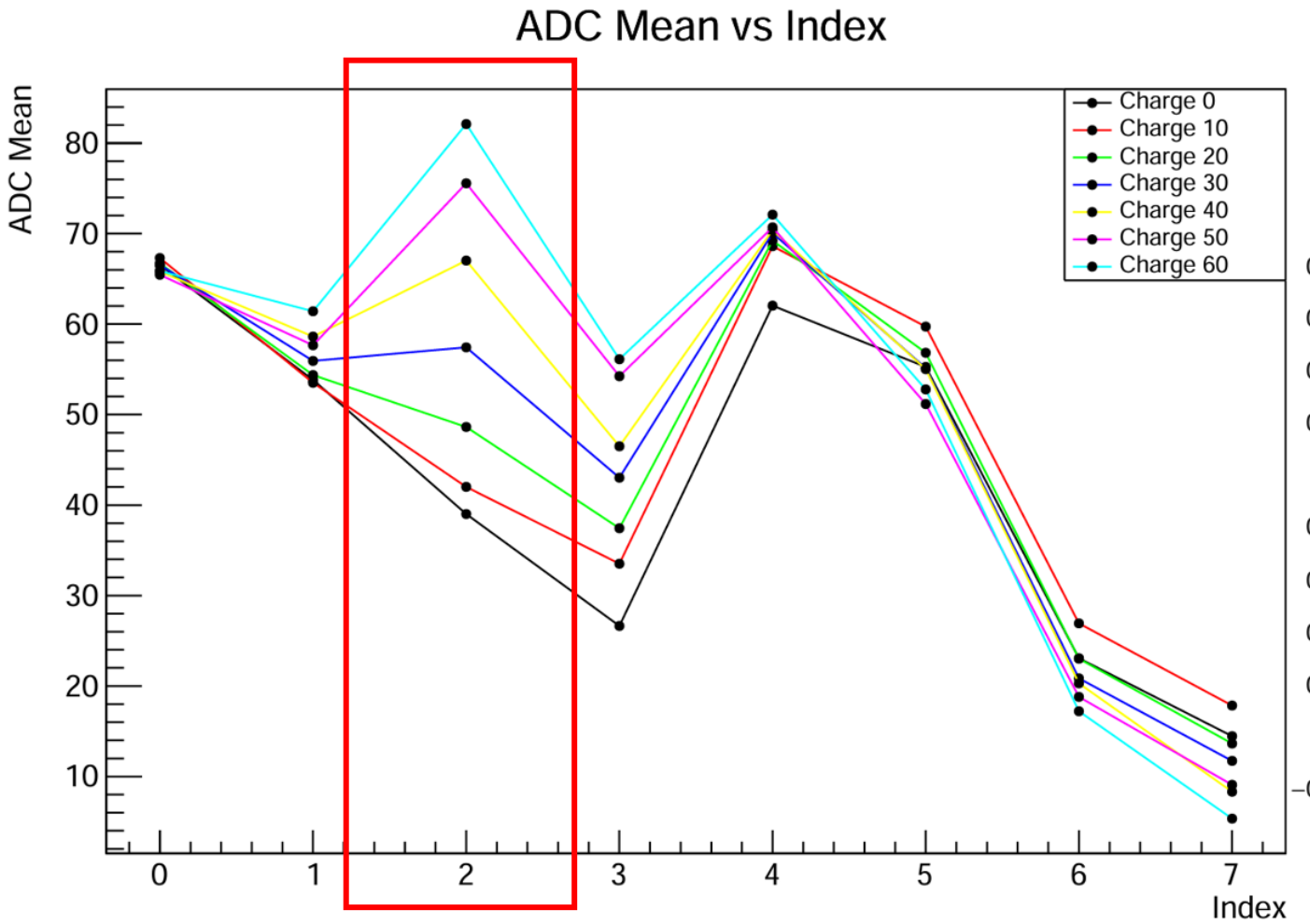


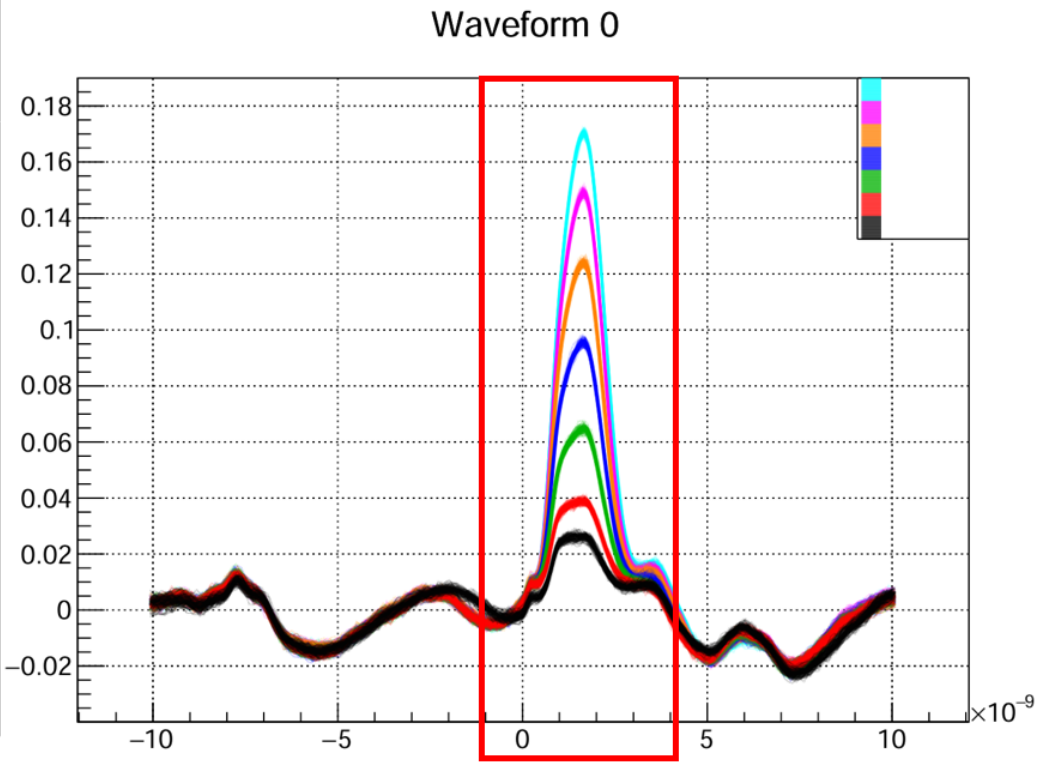
The larger the injected charge, the larger the ADC value.

TDC sigma trend for each time sample



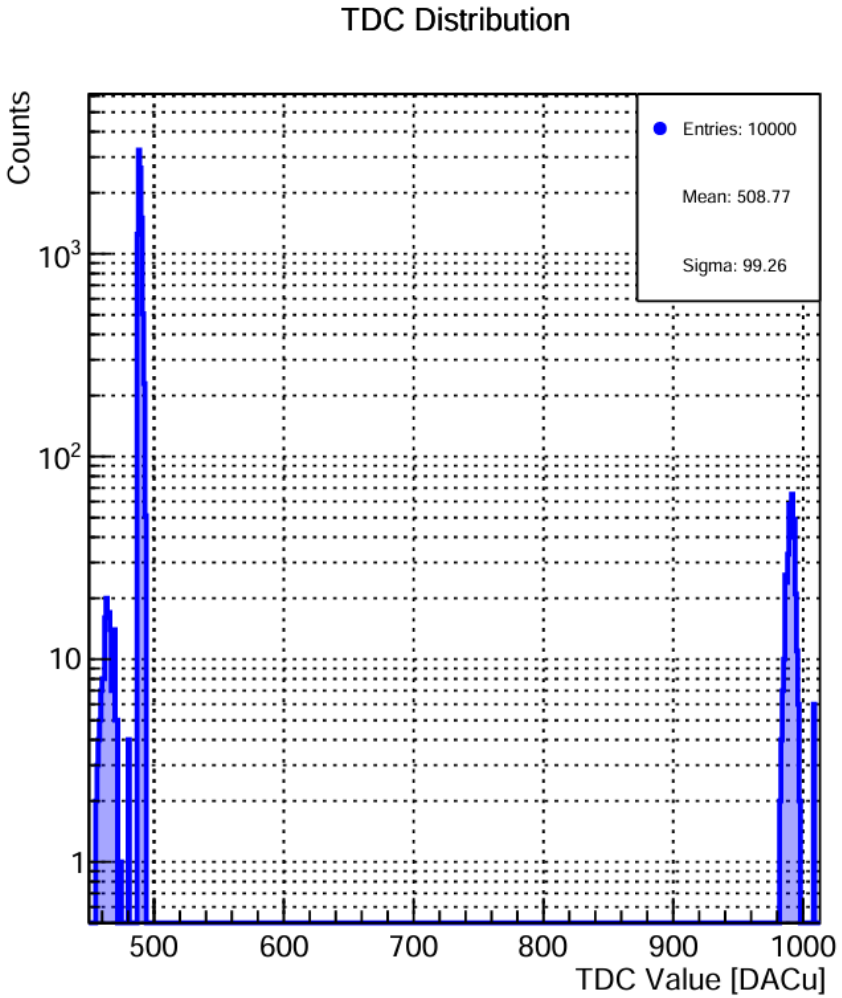



ADC Mean vs Index
ADC Mean
Index
Charge 0
Charge 10
Charge 20
Charge 30
Charge 40
Charge 50
Charge 60

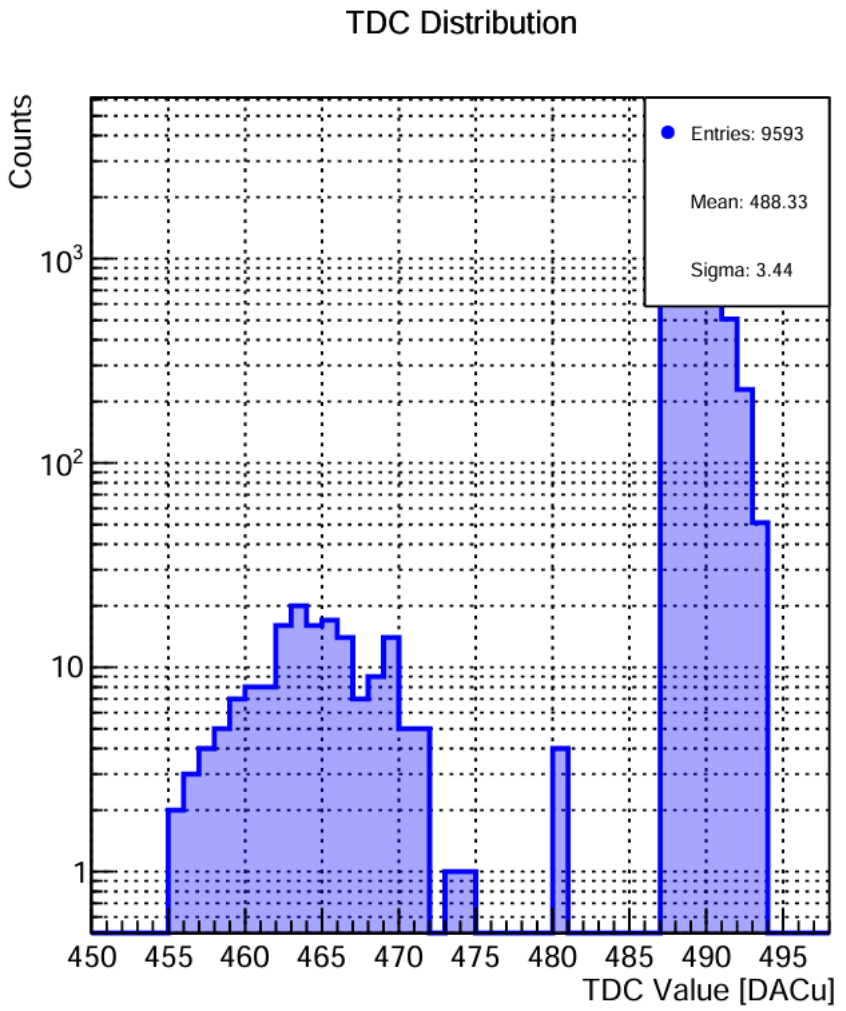

Waveform 0
×10⁻⁹

Except Overflow(?) for Q = 0 DACu

- Threshold = 1 - 1023



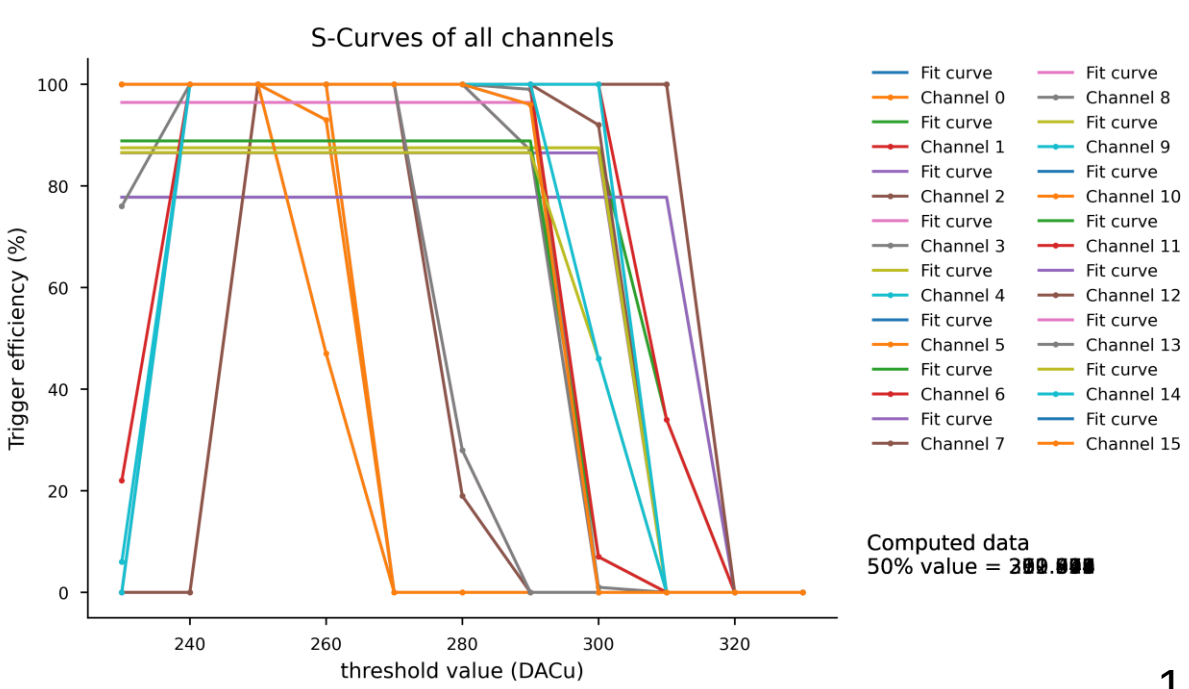
- Threshold = 1 - 990



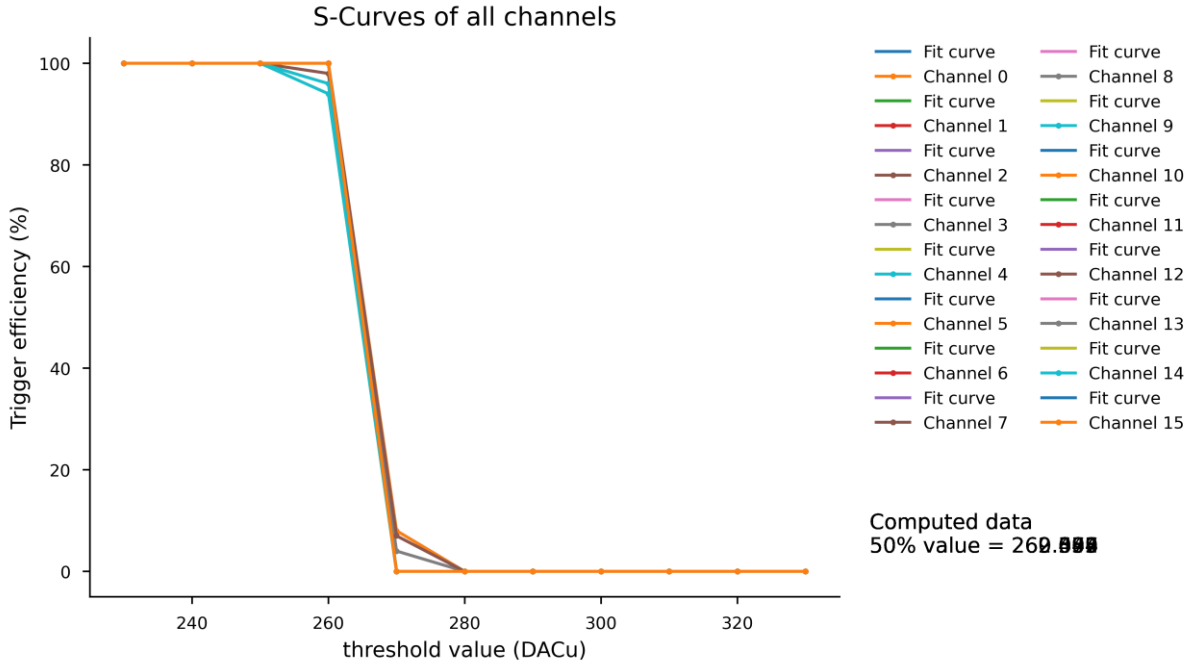
All pixel analysis
charge is injected to all pixels

Threshold & Charge Correction

Before Correction



After Correction

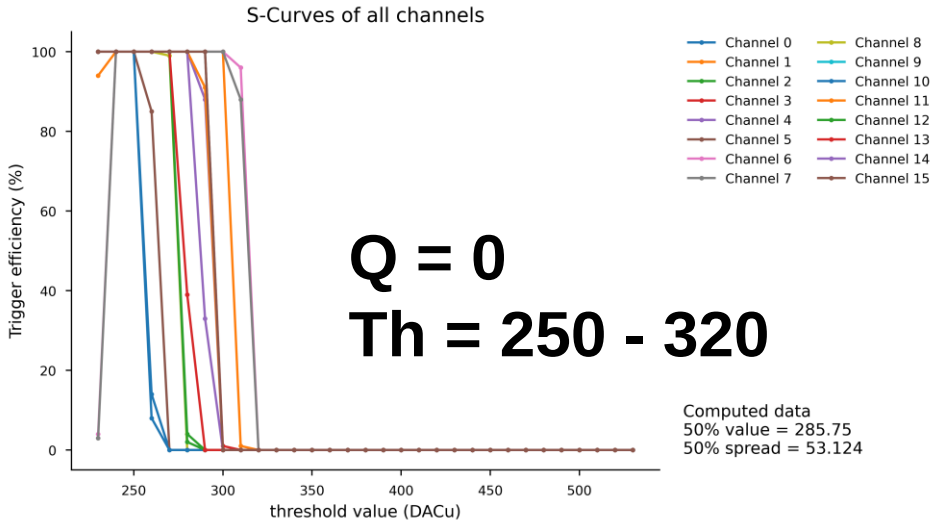


Testbench name : threshold_sweep
Pixels in which we inject signal : All pixels
Charge values tested : [12]
Threshold values tested : [230 - 330]
Number of cmd_pulse per event : 100

1. Correction Threshold for charge value = 12 DAC
2. Correction ADC for charge value = 12 DAC
3. Adjust “vth_cor_opt” by myself while watching Threshold_sweep.

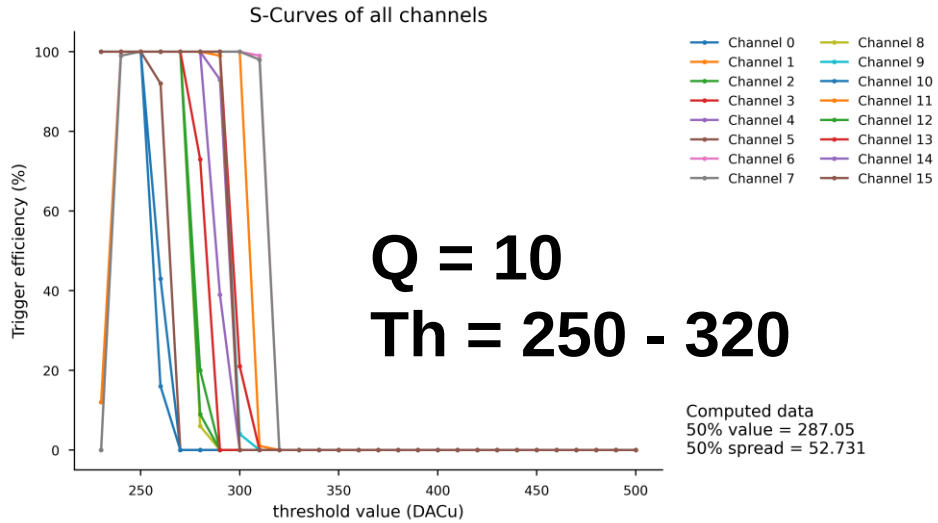
All channels now fall at 270 DACu when charge value is [12]

No correction threshold sweep (A2 board)



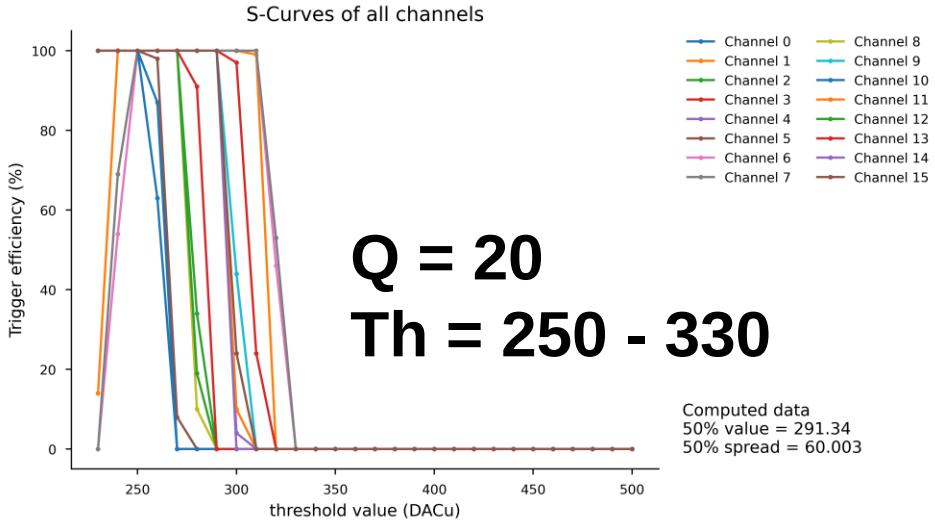
Q = 0
Th = 250 - 320

Computed data
50% value = 285.75
50% spread = 53.124



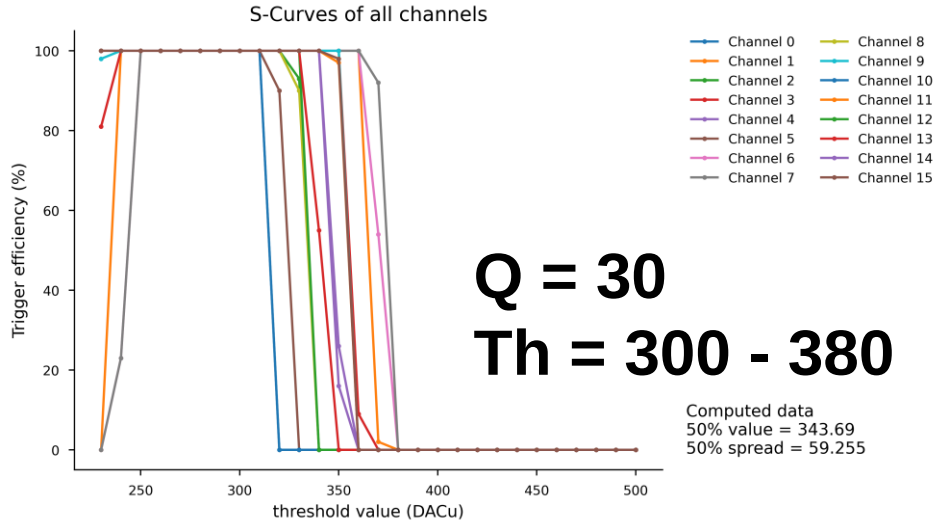
Q = 10
Th = 250 - 320

Computed data
50% value = 287.05
50% spread = 52.731



Q = 20
Th = 250 - 330

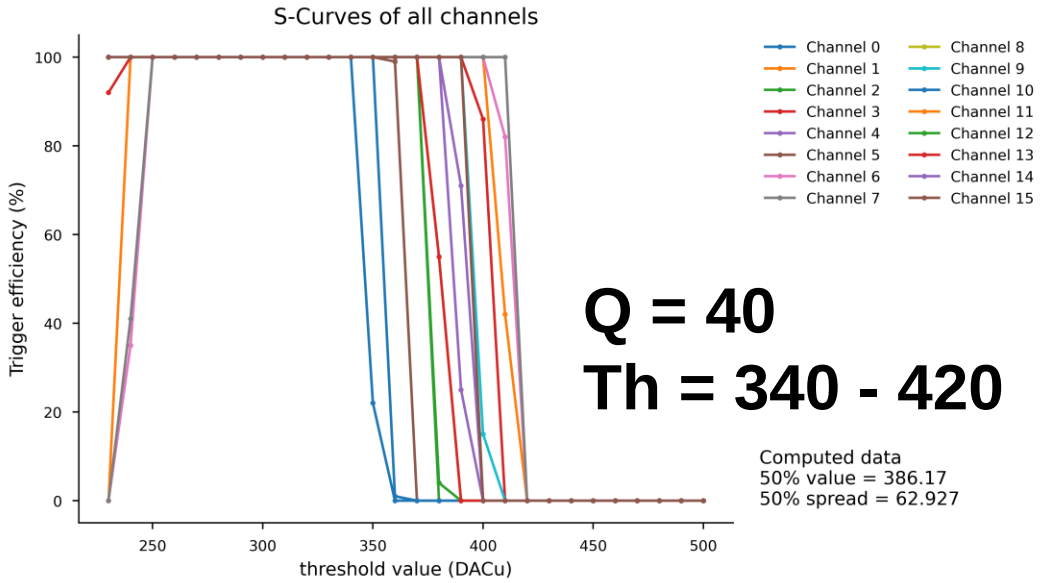
Computed data
50% value = 291.34
50% spread = 60.003



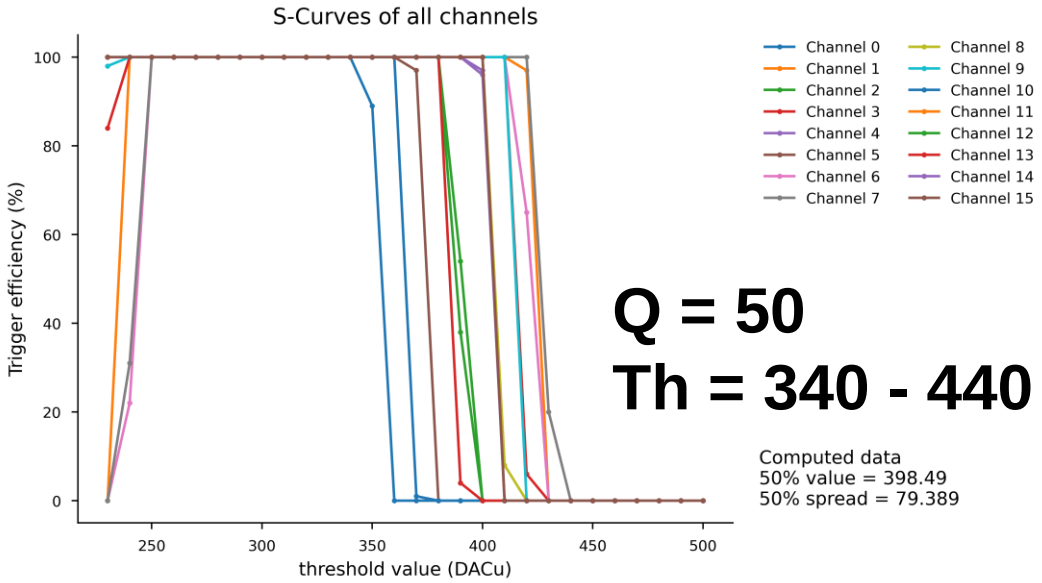
Q = 30
Th = 300 - 380

Computed data
50% value = 343.69
50% spread = 59.255

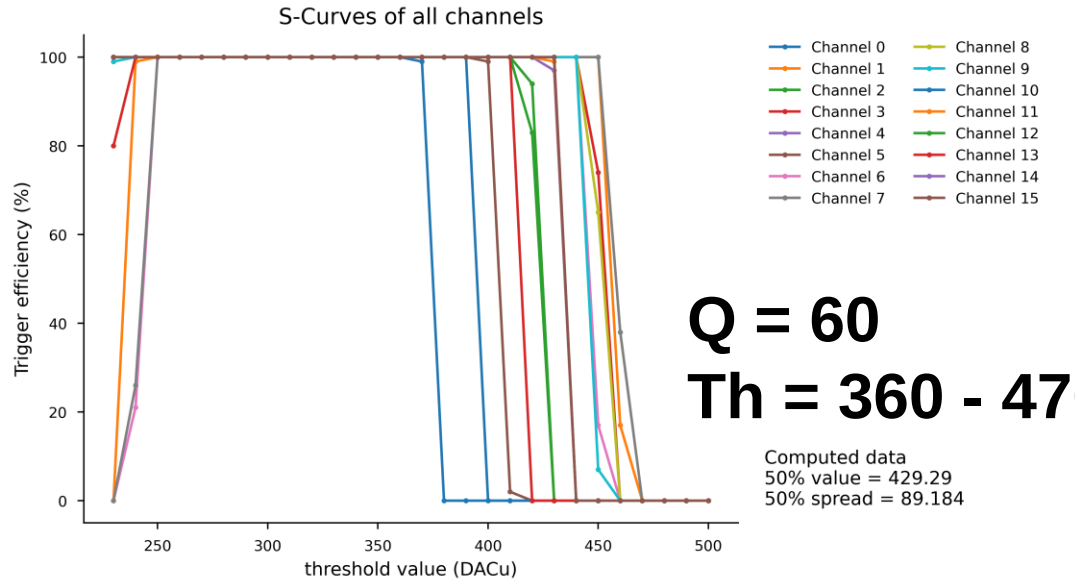
No correction threshold sweep



Q = 40
Th = 340 - 420



Q = 50
Th = 340 - 440



Q = 60
Th = 360 - 470

- Threshold sweep range is wide if no correction is made
- An error occurred after Q = 20, so the FPGA was rebooted. This may have caused the value to increase significantly (if so, this is critical info.)

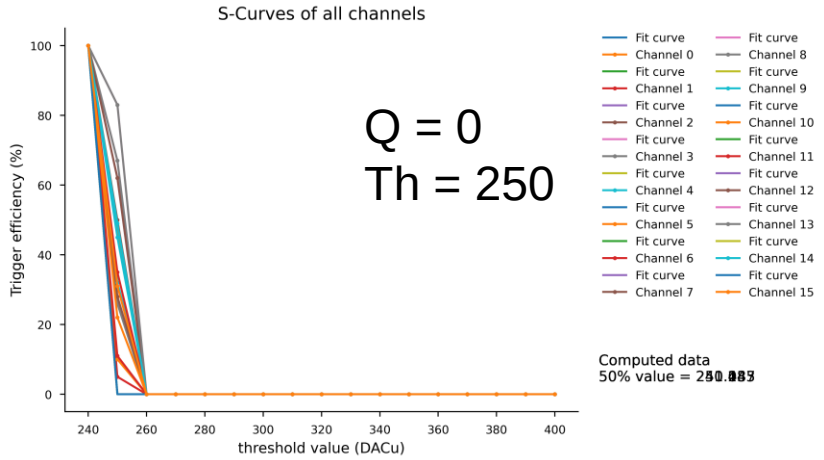
Threshold and ADC charge correction

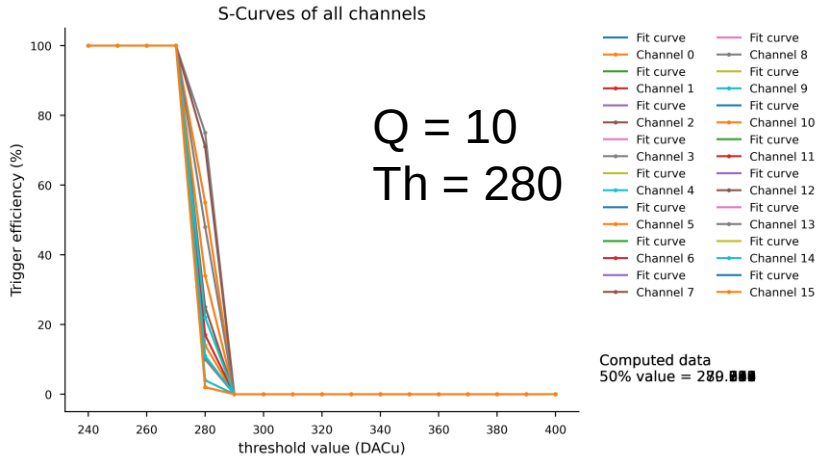

```
#Q = 0 DAC #th = 250
#vth_cor_list  = [62, 46, 47, 54, 46, 61, 40, 39, 51, 44, 62, 48, 56, 57, 47, 50]
#vref_cor_list = [64, 70, 79, 79, 66, 78, 79, 79, 70, 82, 76, 85, 78, 84, 77, 78]

#Q = 10 DAC #th = 280
#vth_cor_list  = [75, 60, 62, 68, 60, 76, 54, 54, 66, 59, 78, 62, 72, 72, 63, 66]
#vref_cor_list = [64, 70, 79, 79, 66, 77, 79, 79, 70, 82, 76, 85, 78, 84, 77, 78]

#Q = 20 DAC #th = 310
#vth_cor_list  = [79, 54, 56, 62, 55, 71, 50, 48, 58, 54, 83, 56, 65, 65, 56, 58]
#vref_cor_list = [64, 70, 79, 79, 66, 77, 78, 79, 70, 82, 76, 85, 77, 83, 77, 78]

#Q = 30 DAC #th = 340
vth_cor_list  = [74, 52, 54, 59, 54, 68, 49, 46, 54, 51, 70, 54, 62, 64, 54, 57]
vref_cor_list = [64, 70, 80, 79, 66, 77, 79, 79, 70, 81, 76, 86, 78, 83, 77, 78]

#Q = 40 DAC #th = 370
#vth_cor_list  = [88, 57, 62, 63, 62, 76, 58, 53, 55, 57, 81, 65, 72, 75, 62, 67]
#vref_cor_list = [64, 70, 79, 79, 66, 77, 78, 79, 70, 81, 75, 85, 77, 83, 77, 78]

#Q = 50 DAC #th = 400
#vth_cor_list  = [92, 53, 60, 60, 59, 74, 56, 48, 43, 53, 80, 62, 69, 75, 60, 66]
#vref_cor_list = [64, 70, 80, 79, 66, 77, 79, 79, 70, 81, 76, 86, 78, 83, 77, 78]

#Q = 60 DAC #th = 420
#vth_cor_list  = [93, 44, 53, 51, 54, 69, 52, 41, 23, 47, 79, 58, 64, 72, 53, 62]
#vref_cor_list = [64, 70, 80, 79, 66, 77, 79, 79, 71, 81, 75, 86, 78, 84, 77, 79]

#Default
#vth_cor_list  = [64, 64, 64, 64, 64, 64, 64, 64, 64, 64, 64, 64, 64, 64, 64, 64]
#vref_cor_list = [64, 64, 64, 64, 64, 64, 64, 64, 64, 64, 64, 64, 64, 64, 64, 64]
```

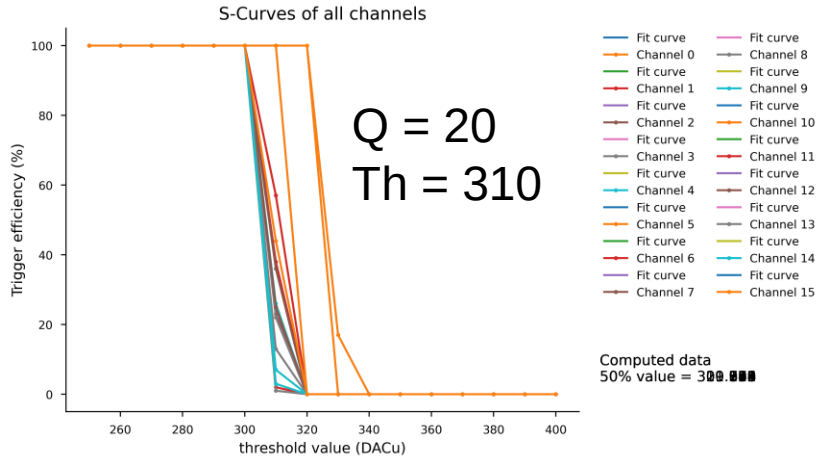
New firmware

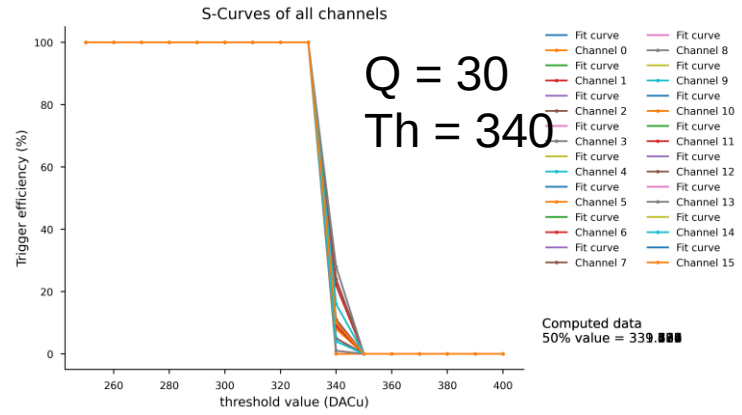
A2 Board EICROC0
(No AC-LGAD)

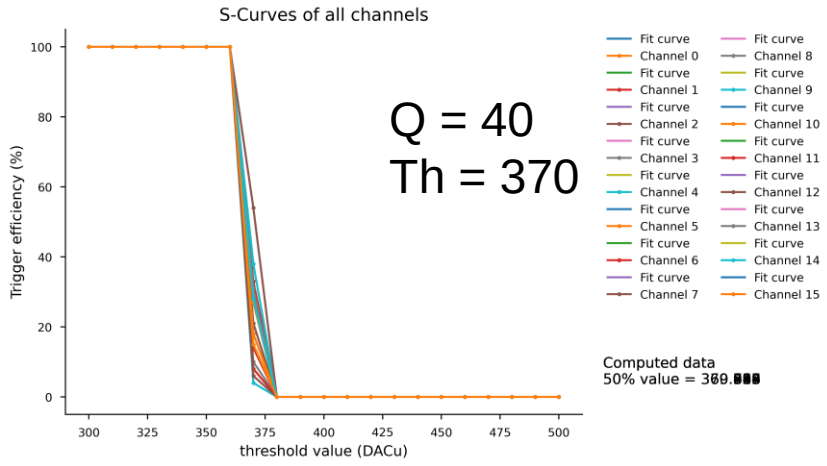
A2 Board (only EICROC0)



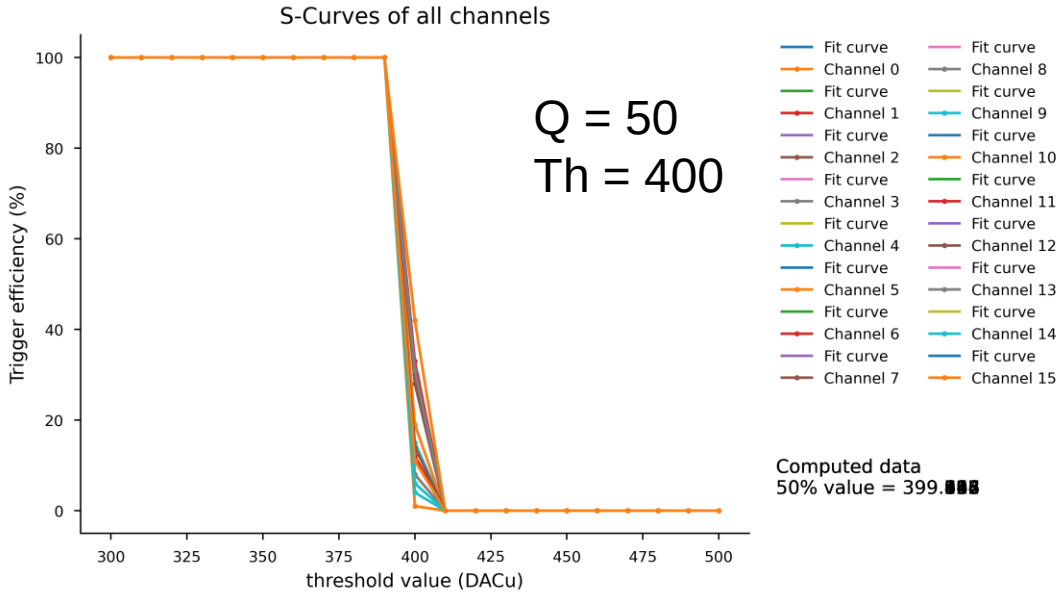


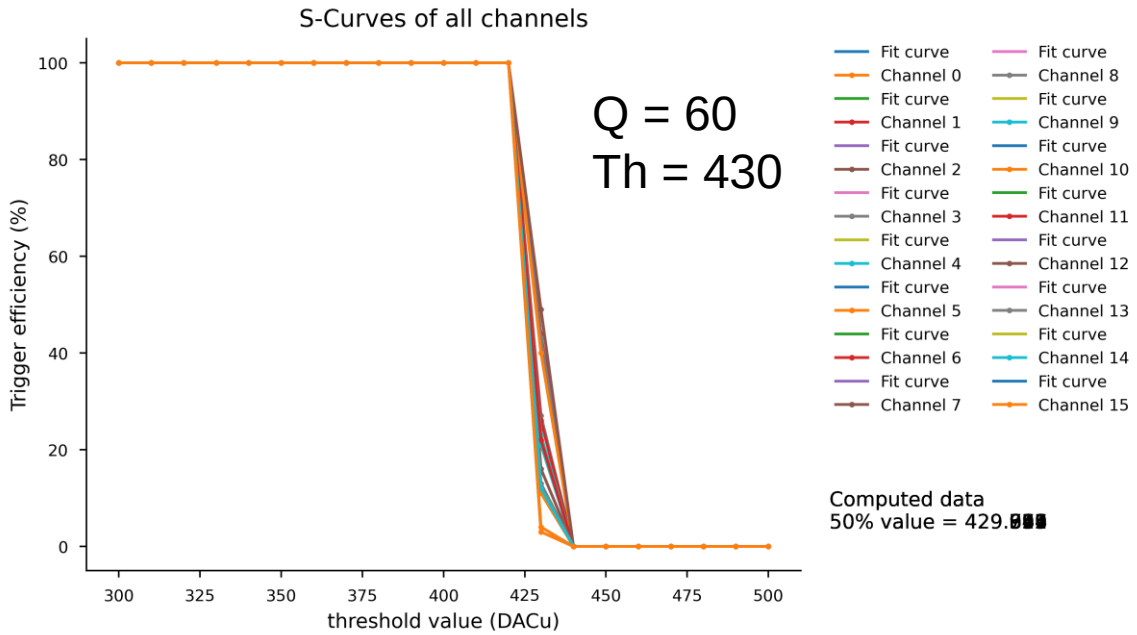






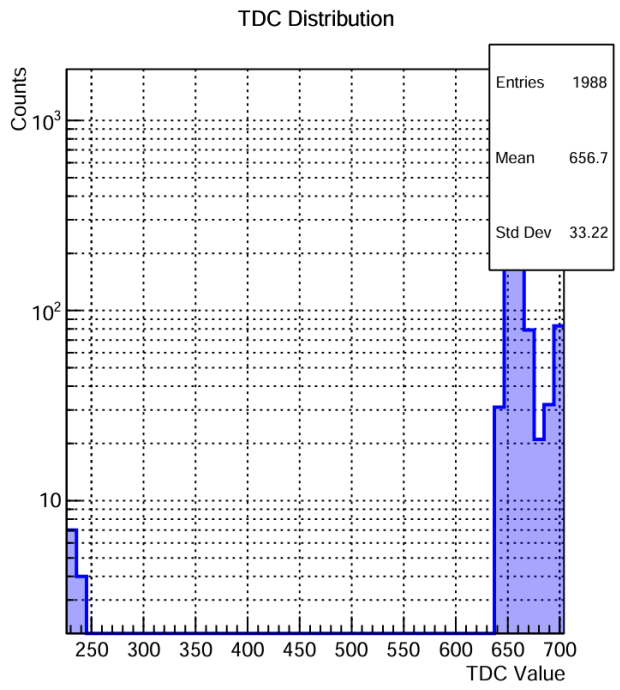
A2 Board (only EICROC0)

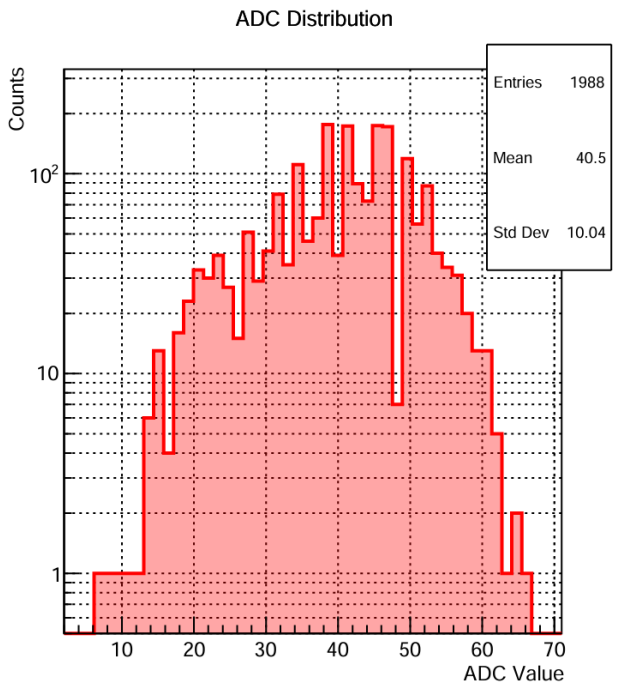




- I can correct the threshold sweep and charge for Q = 0, 10, 20, …, 60
- Q = 20 is a bit strange because ch 0, 5, 10 is a bit larger value than my setting value.
- The other charges have roughly the same value but are less efficient, making all-pixel analysis possible.

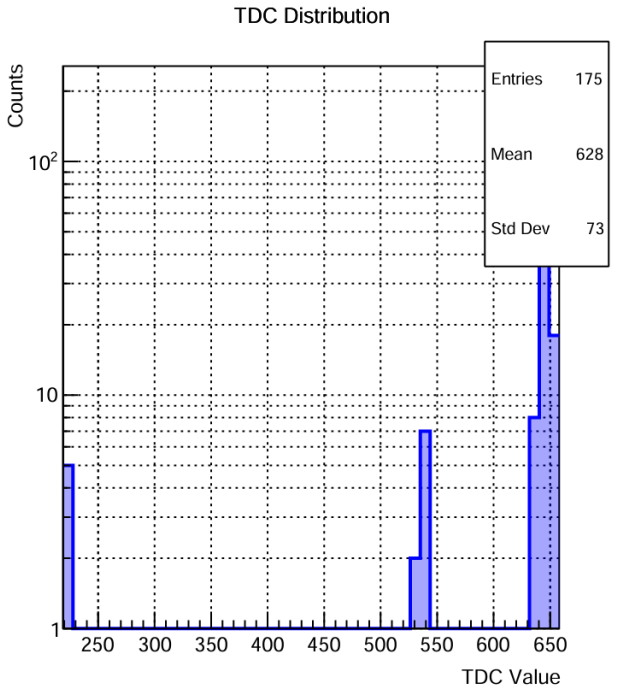
A2 board TDC ADC distribution (10000 events)

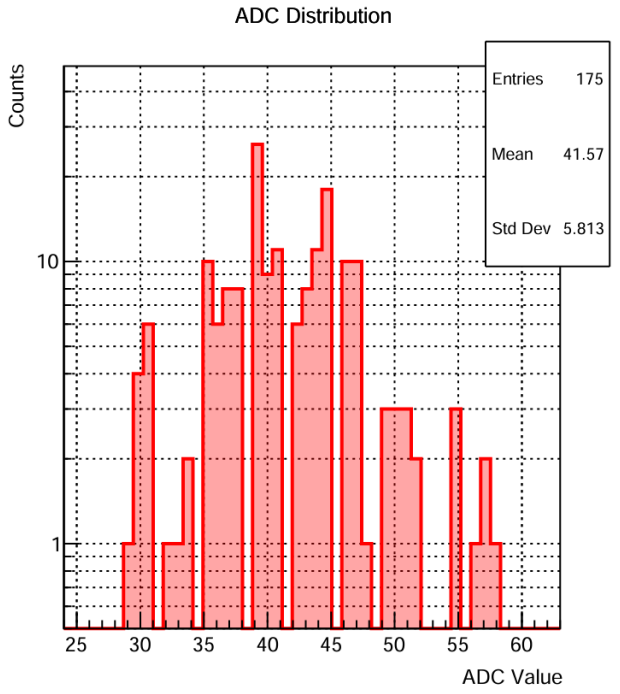




Q = 10

Mean = 40.5



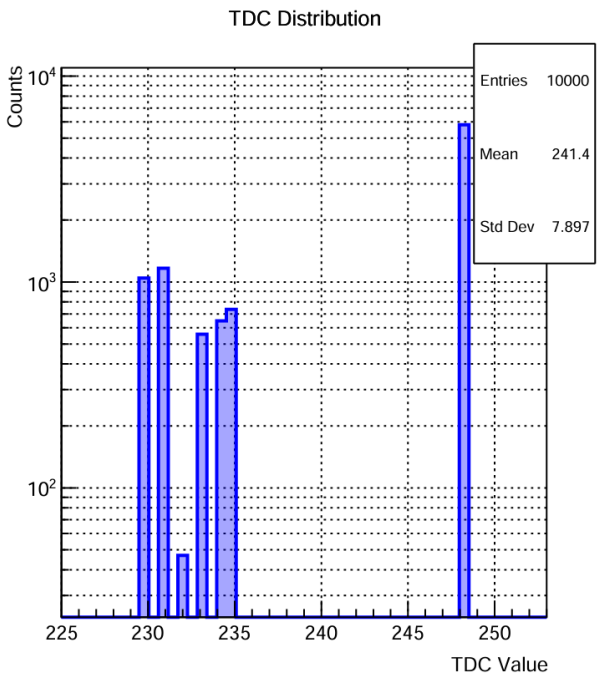


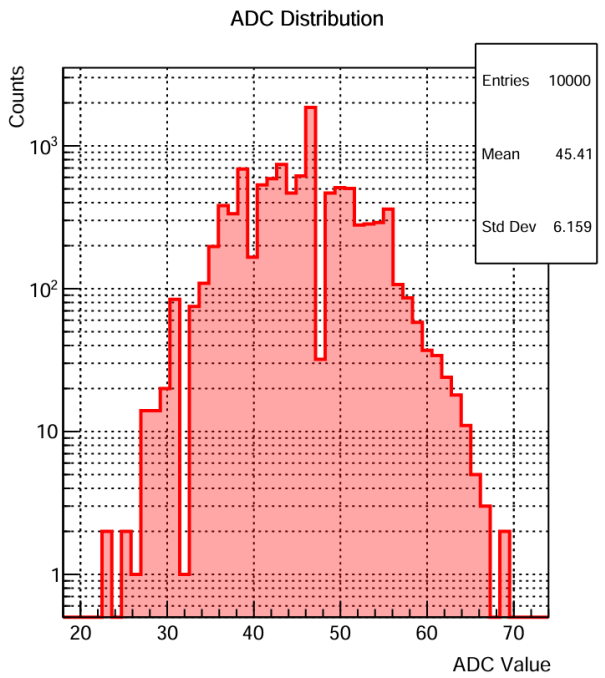
Q = 20

Mean = 41.6

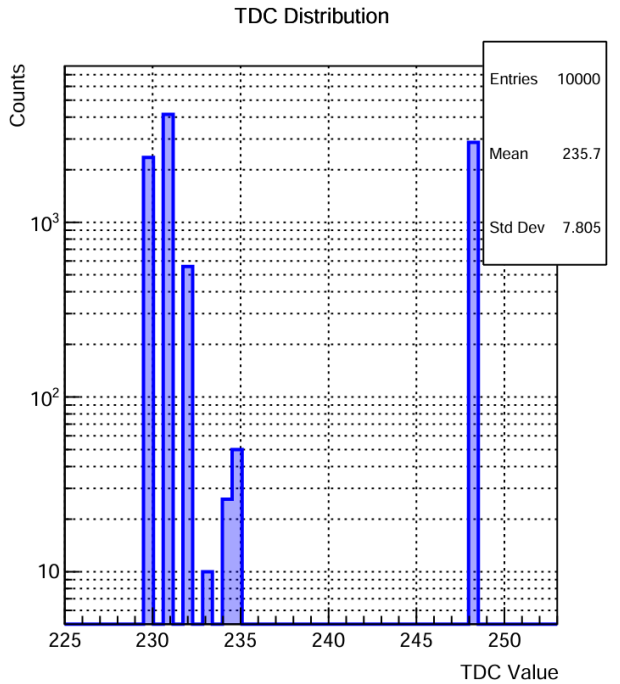
- Q = 0 is no entry (all events are not only one hitbit cell = 1)
- Q = 10 and 20 are very small entries because they are lost in the noise (?)
- ADC distribution looks better

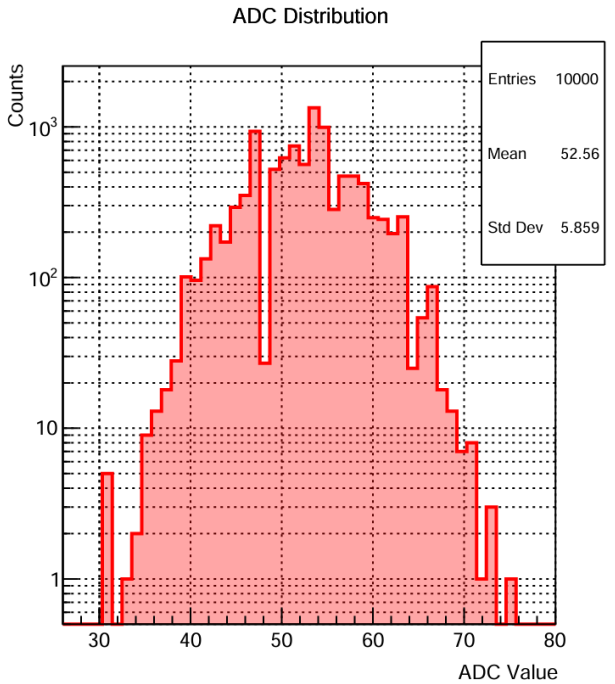
A2 board TDC ADC distribution





Q = 30

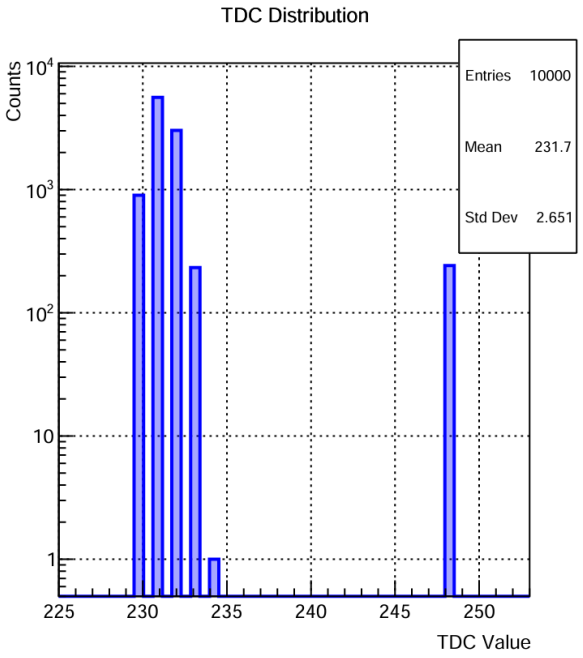


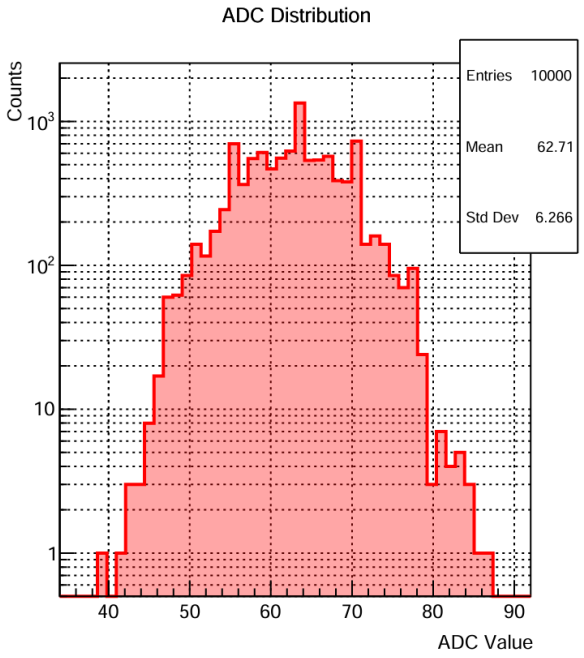


Q = 40

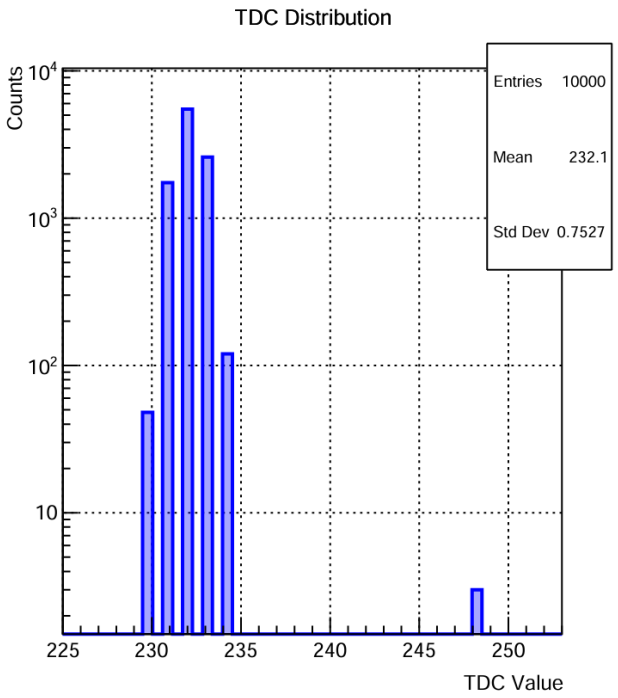
Q = 30 and 40 are all entries and small TDC sigma
But, in both cases, there are many entries in the 248 DACu bin. The reason is unknown.

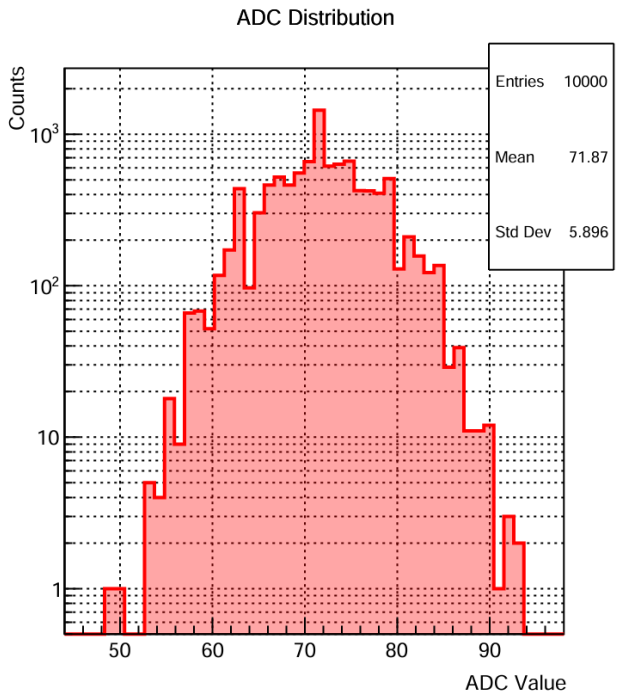
A2 board TDC ADC distribution





Q = 50





Q = 60

Q = 50 and 60 are all entries and small TDC sigma
But, in both cases, there are many entries in the 248 DACu bin. The reason is unknown. (same for Q = 30, 40)

B1 board (No HV of AC-LGAD)

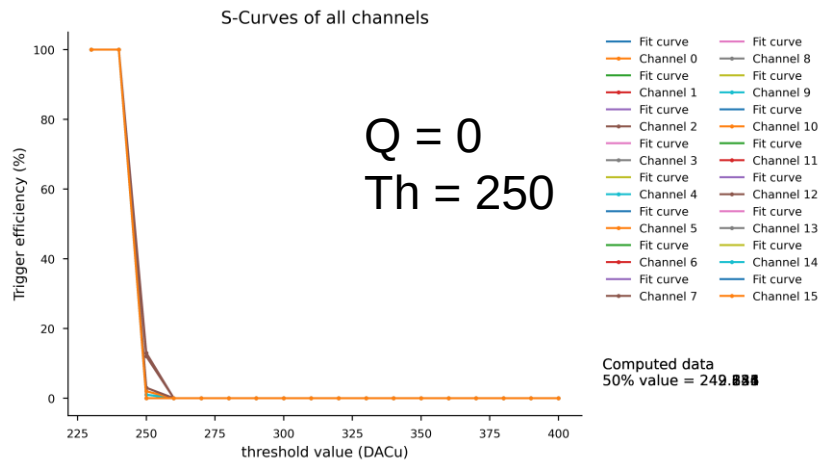
Correction Values

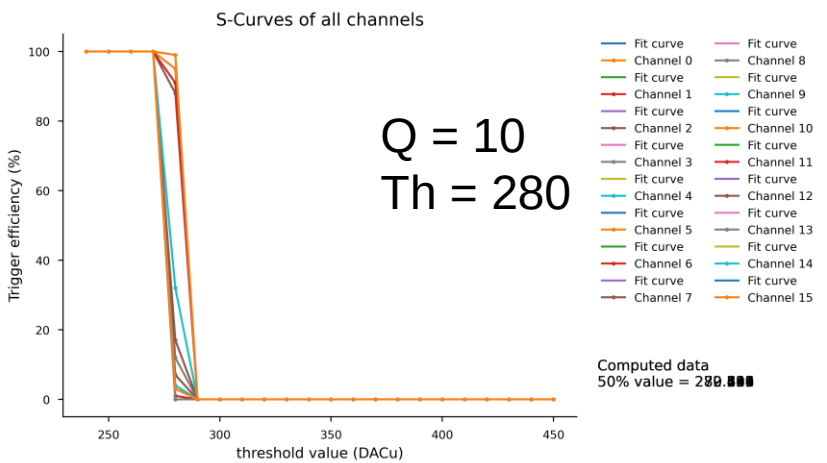
```
#==========B1 Board============#

#Q = 0 DAC #th = 250
#vth_cor_list  = [30, 31, 14, 29, 28, 26, 44, 25, 39, 26, 45, 32, 31, 35, 32, 34]
#vref_cor_list = [64, 71, 64, 75, 66, 61, 69, 66, 67, 72, 74, 83, 76, 81, 72, 83]

#Q = 10 DAC #th = 280
#vth_cor_list  = [44, 45, 28, 44, 42, 34, 52, 35, 53, 34, 54, 46, 46, 50, 46, 50]
#vref_cor_list = [64, 71, 64, 75, 66, 61, 69, 66, 67, 72, 74, 83, 76, 81, 72, 83]

#Q = 20 DAC # th = 310
#vth_cor_list  = [55, 54, 31, 55, 50, 31, 53, 41, 60, 35, 53, 54, 59, 59, 53, 63]
#vref_cor_list = [64, 71, 64, 75, 66, 61, 69, 66, 67, 72, 74, 83, 76, 81, 72, 83]

#Q = 30 DAC #th = 340
#vth_cor_list  = [68, 58, 31, 63, 54, 30, 56, 44, 64, 37, 54, 60, 72, 63, 56, 73]
#vref_cor_list = [64, 71, 64, 75, 66, 61, 69, 66, 67, 72, 74, 83, 76, 81, 72, 83]

#Q = 40 DAC #th = 370
#vth_cor_list  = [73, 64, 30, 68, 58, 31, 62, 48, 70, 42, 58, 67, 80, 67, 60, 80]
#vref_cor_list = [64, 71, 64, 75, 66, 61, 69, 66, 67, 72, 74, 83, 76, 81, 72, 83]

#Q = 50 DAC #th = 400
#vth_cor_list  = [77, 70, 28, 73, 62, 30, 67, 54, 75, 46, 62, 74, 87, 74, 67, 88]
#vref_cor_list = [64, 71, 64, 75, 66, 61, 69, 66, 67, 72, 74, 83, 76, 81, 72, 83]

#Q = 60 DAC #th = 430
vth_cor_list  = [82, 77, 26, 79, 68, 31, 74, 59, 83, 51, 68, 83, 94, 80, 72, 96]
vref_cor_list = [64, 71, 64, 75, 66, 61, 69, 66, 67, 72, 74, 83, 76, 81, 72, 83]
```

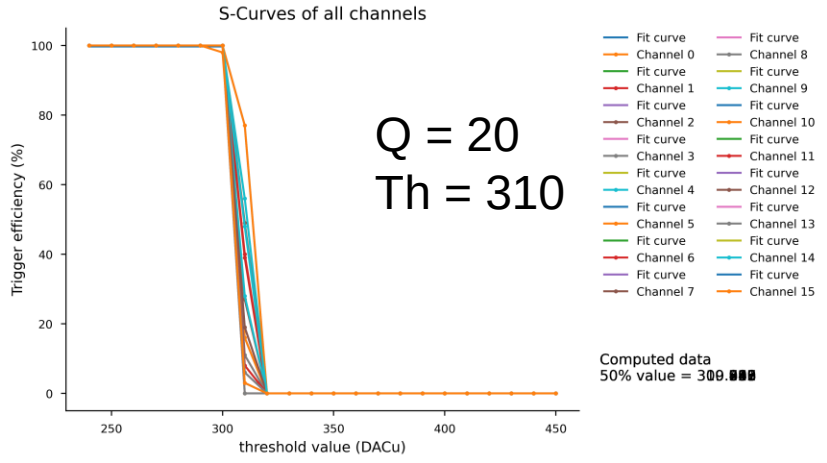
New firmware

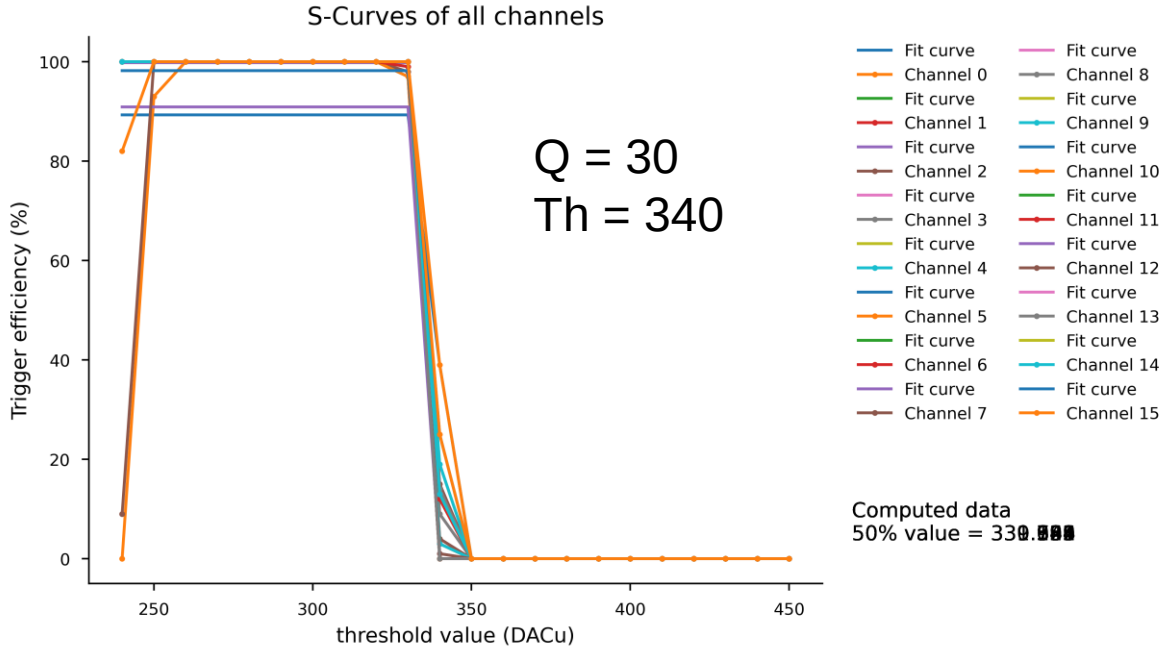
B1 Board EICROC0 (Bump bonded)

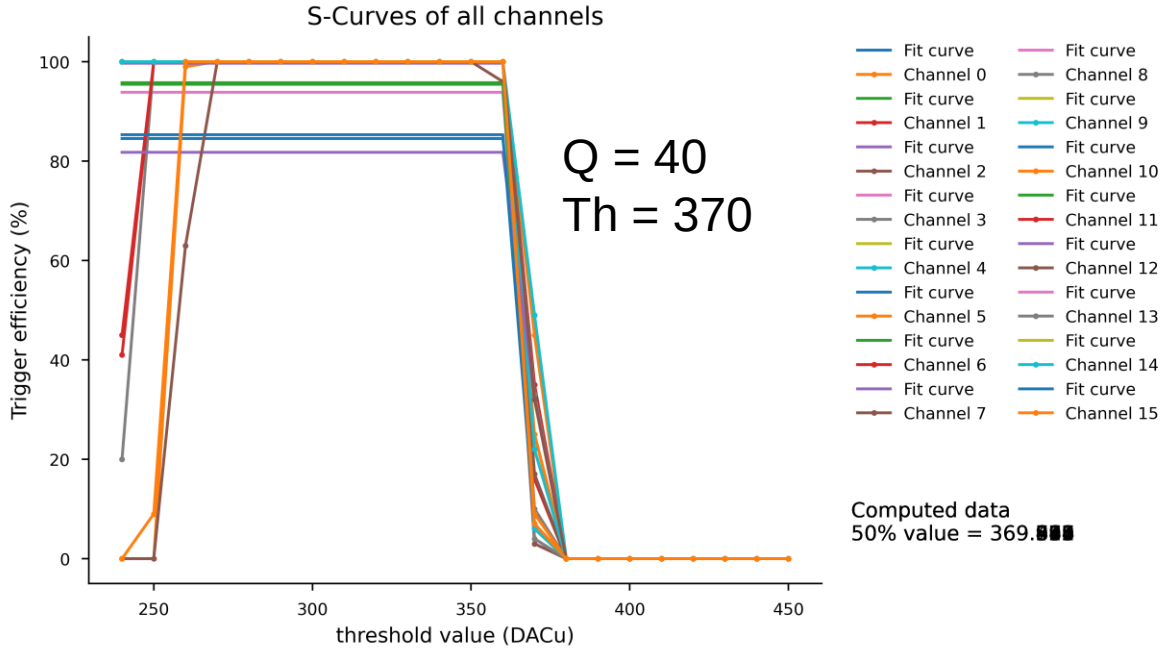
B1 board (No HV of AC-LGAD)



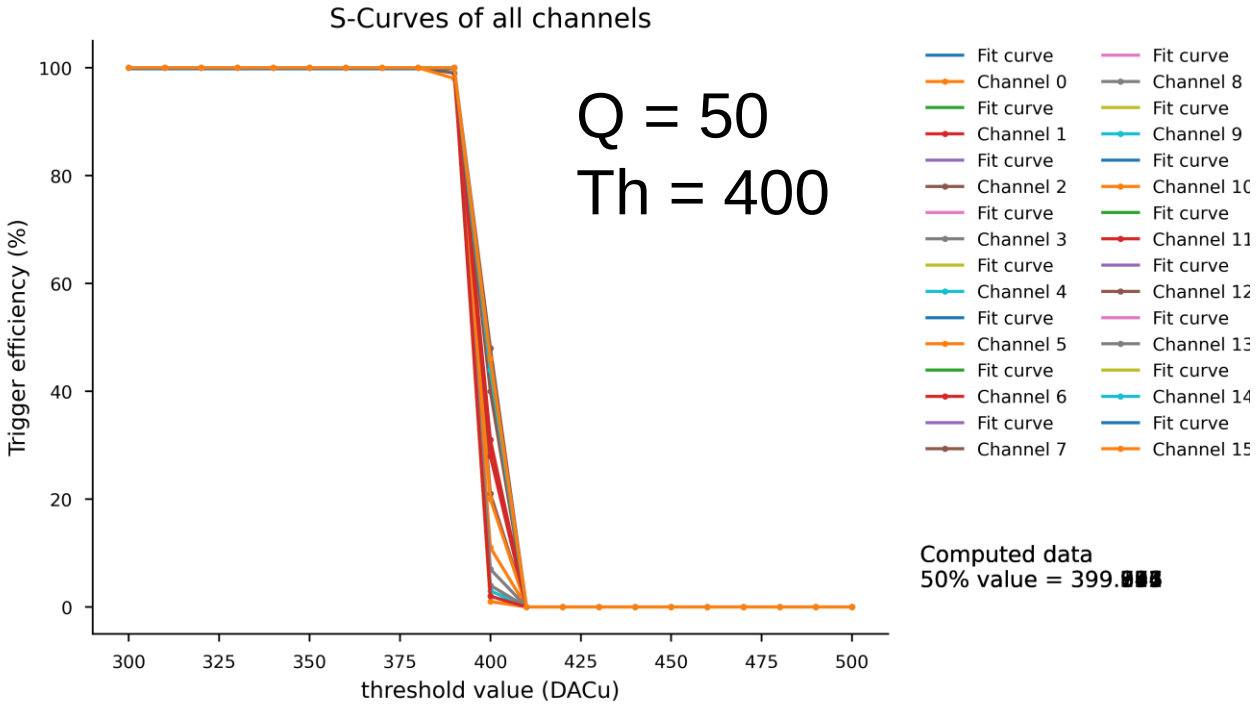


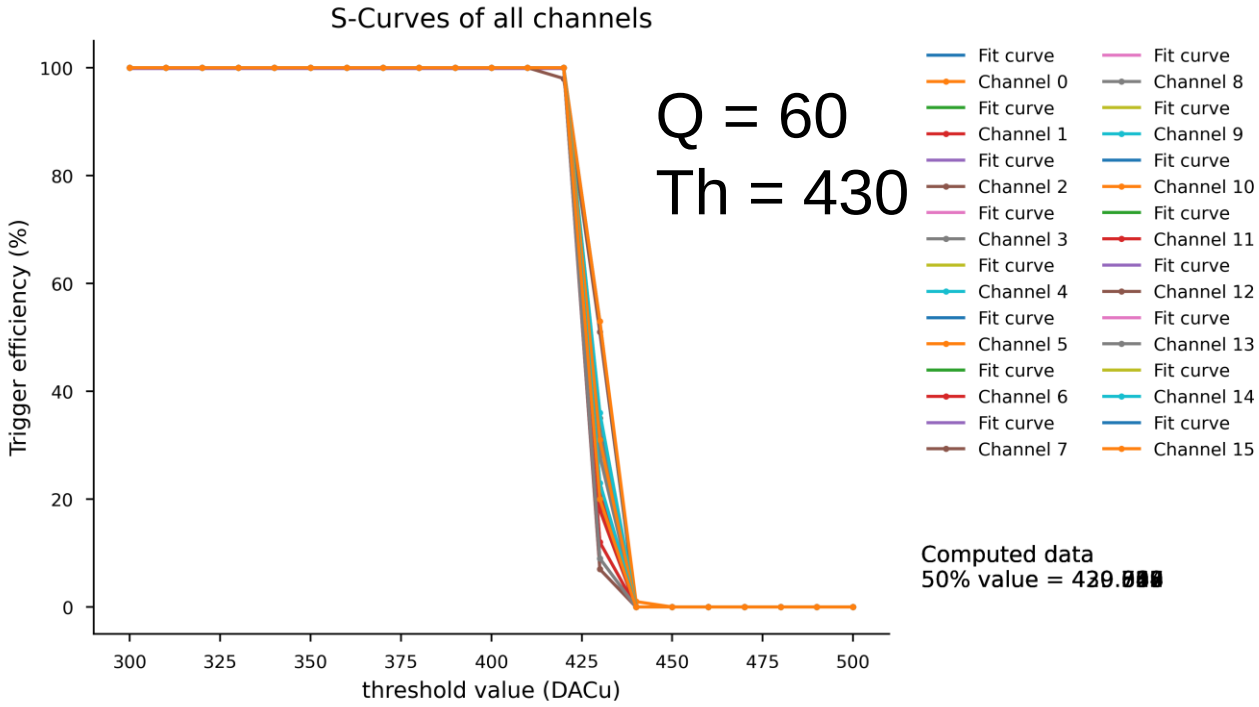






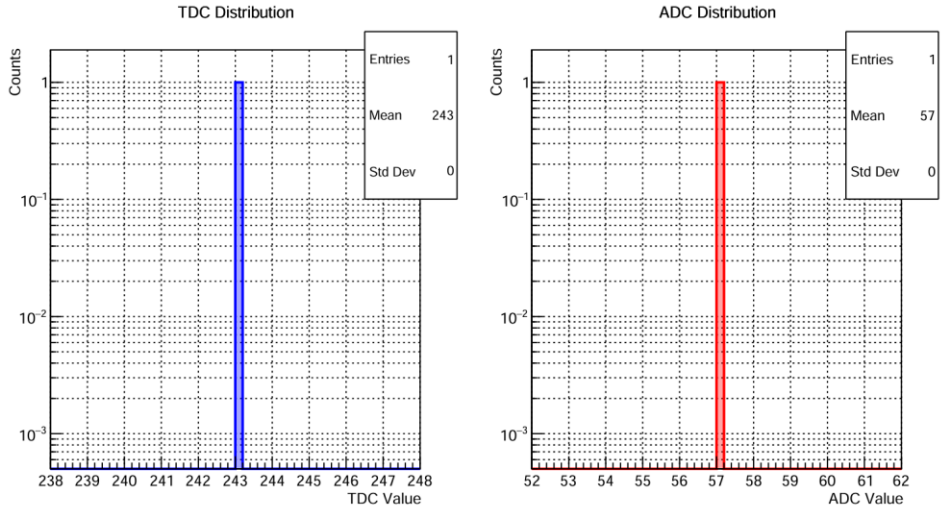
B1 board (No HV of AC-LGAD)



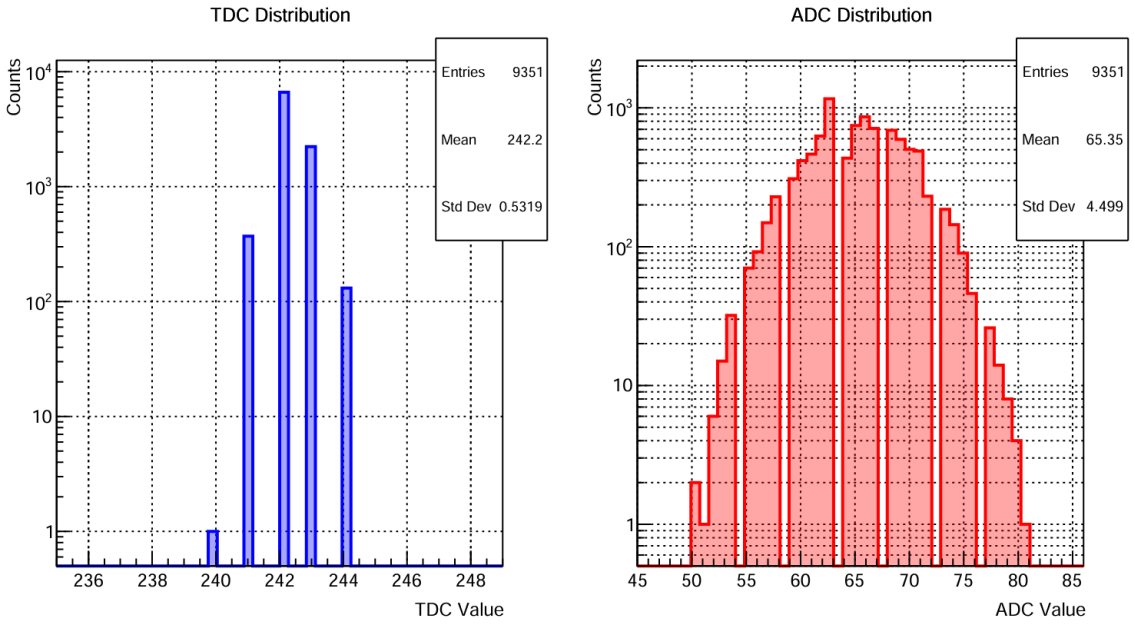


B1 board TDC and ADC distributions

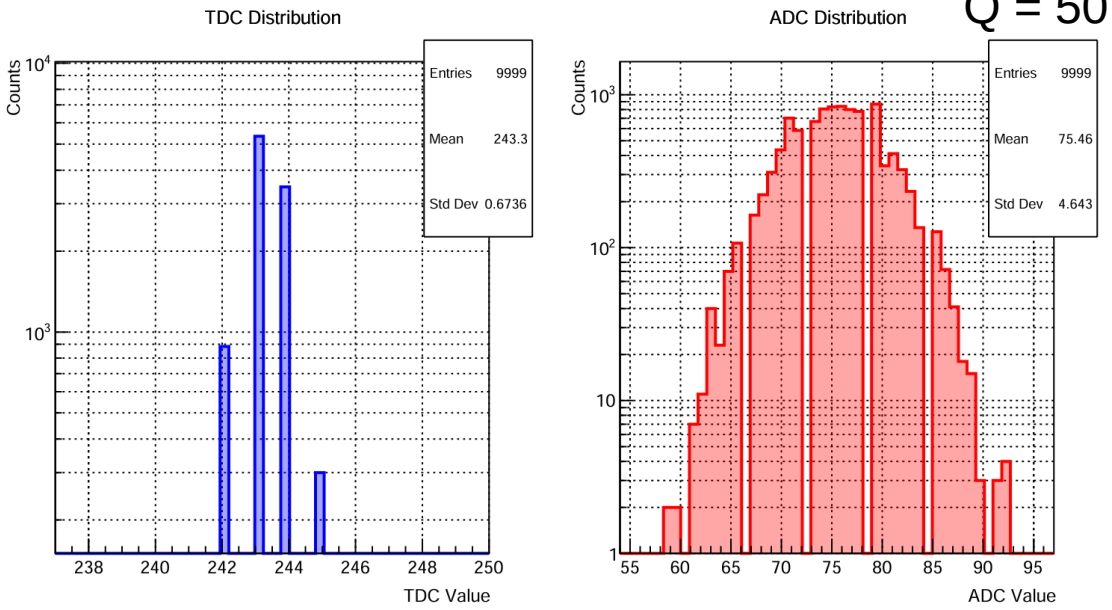
Q = 30



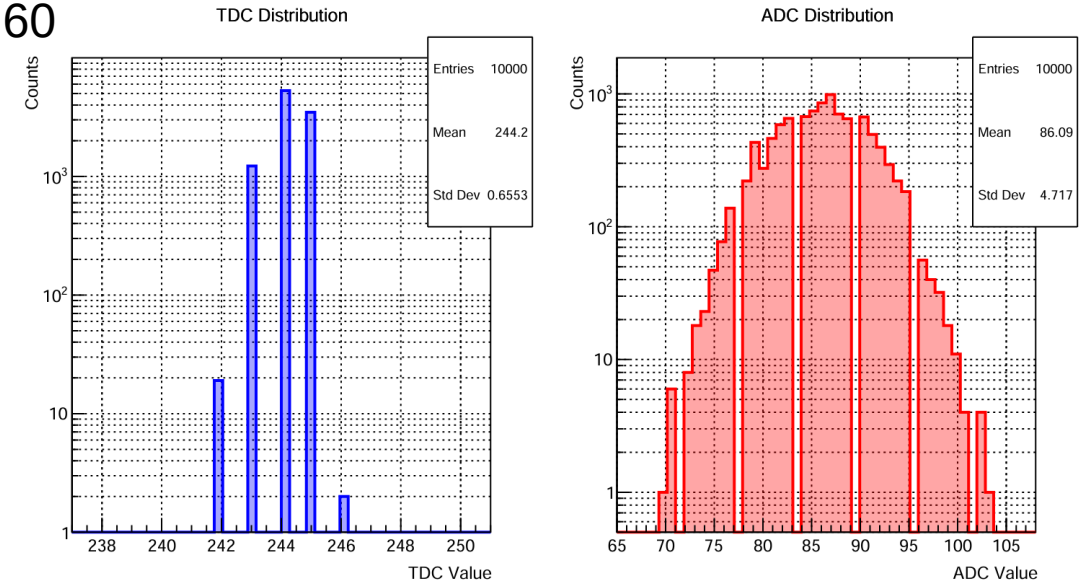
Q = 40



Q = 50



Q = 60



AC-LGAD Analysis @HU

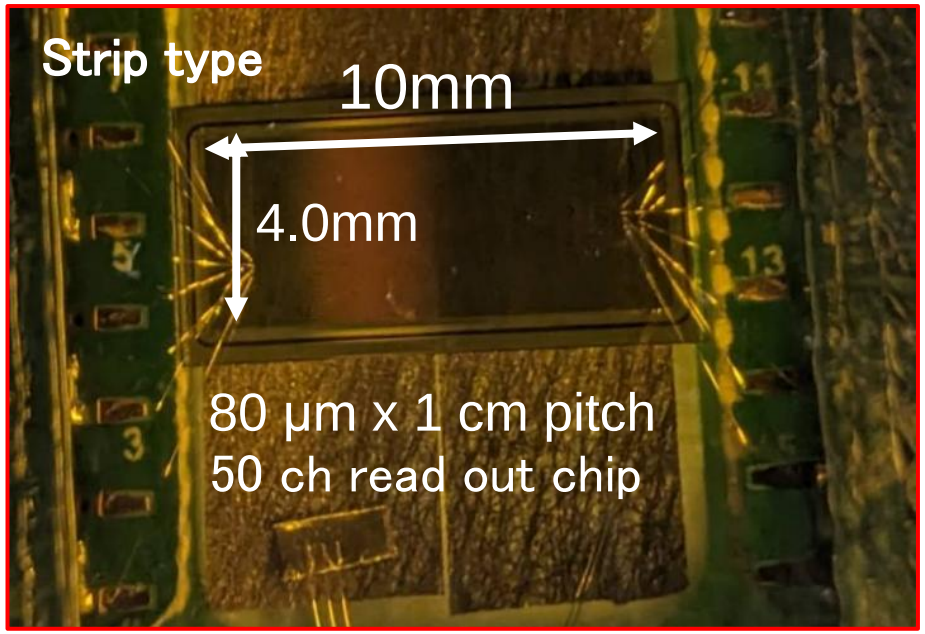
AC-coupled Low Gain Avalanche Detector (AC-LGAD)

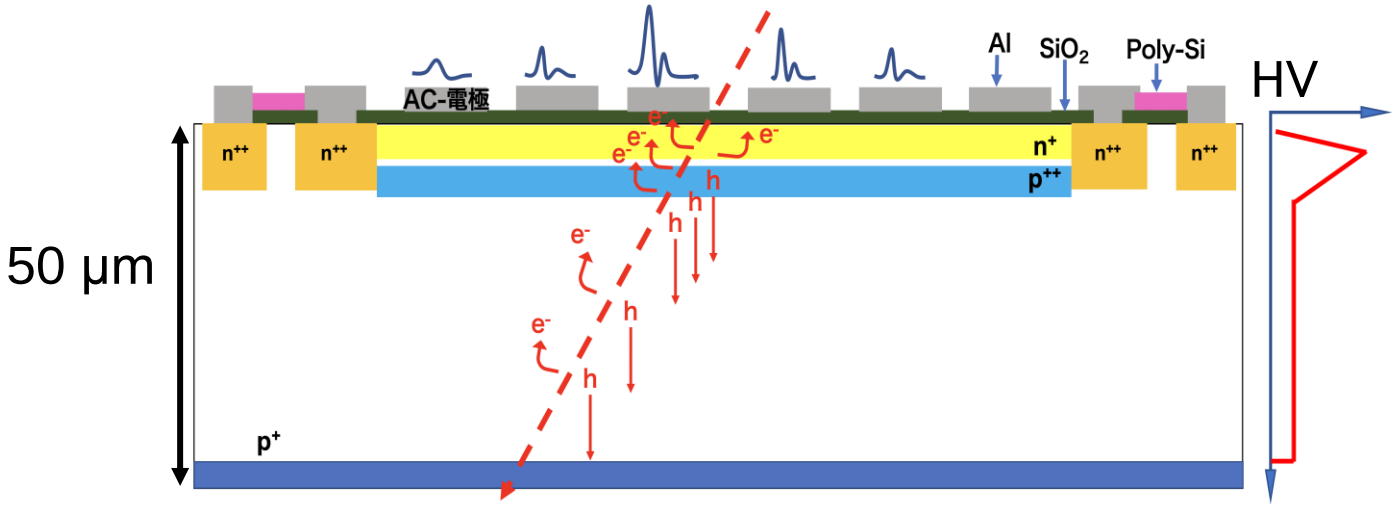
Silicon semiconductor detectors capable of achieving the required performance of TOF[1]

AC-LGAD used for Barrel TOF

- Strip type
- 0.05 x 1 cm^2 pitch, 64 x 4 ch read out sensor
- 3.2 x 4.0 cm^2

Prototype AC-LGAD



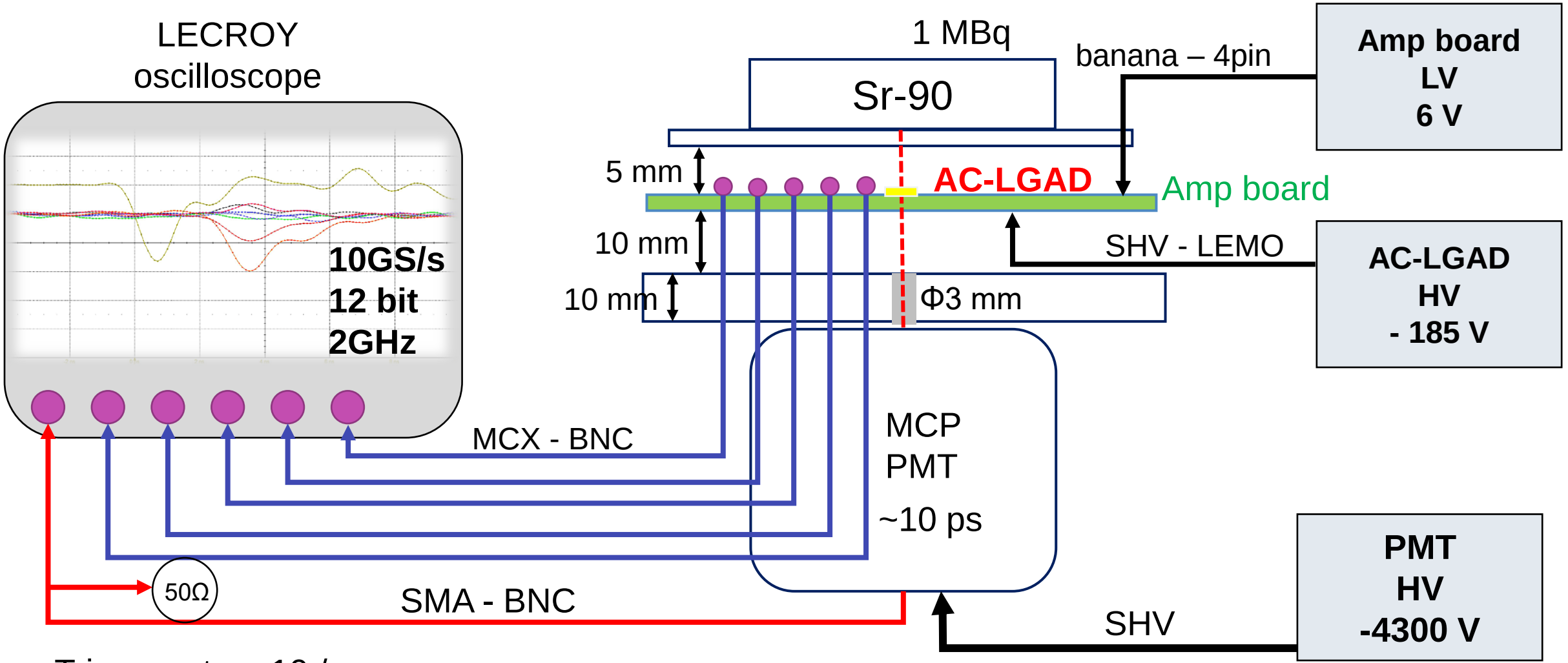


Time resolution → 30 ps
Spatial resolution → 30 μm

[1] https://arxiv.org/pdf/2201.07772

Experimental Environment

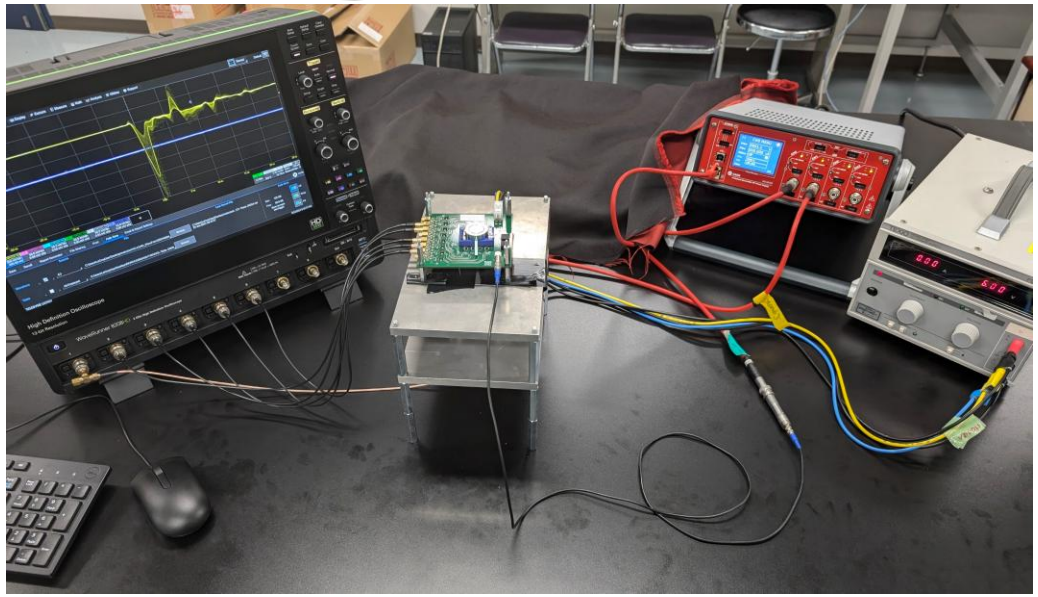
Temperature：21℃, light cut

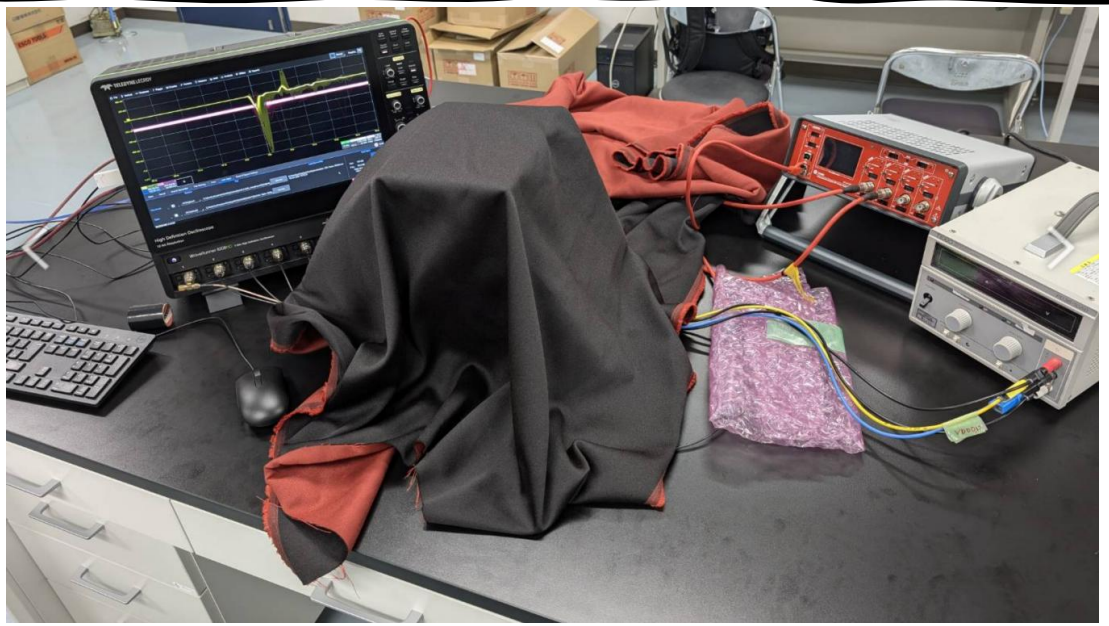


Trigger rate: ~10 /s

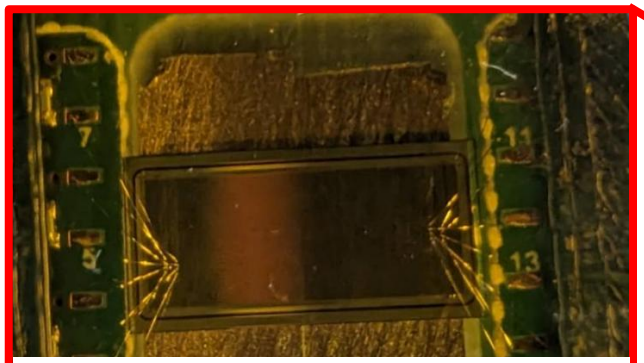
AC-LGAD signal rate:~ 1 /s

Experiment photos

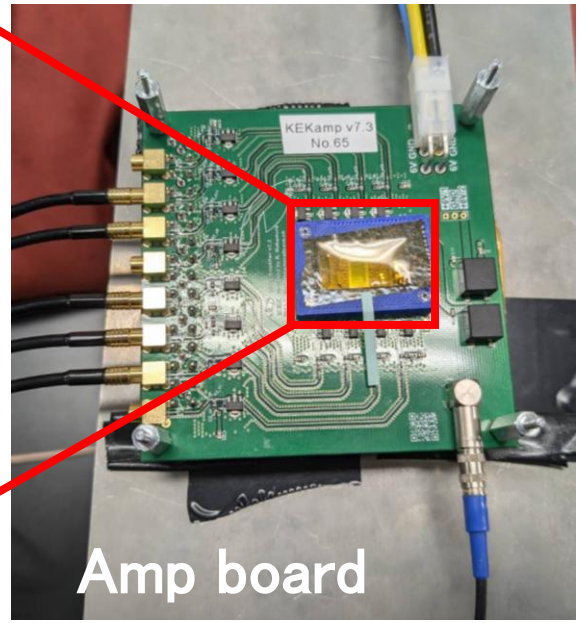


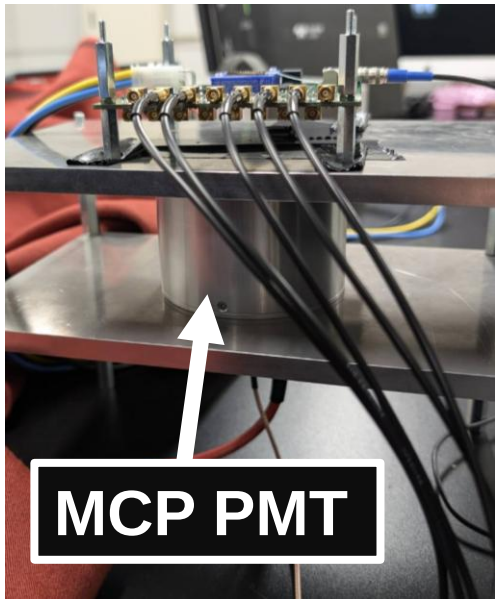


- 80 μm x 1 cm pitch
- 4 mm x 10 mm size
- 50 ch read out sensor
- 16 ch wire-bonded

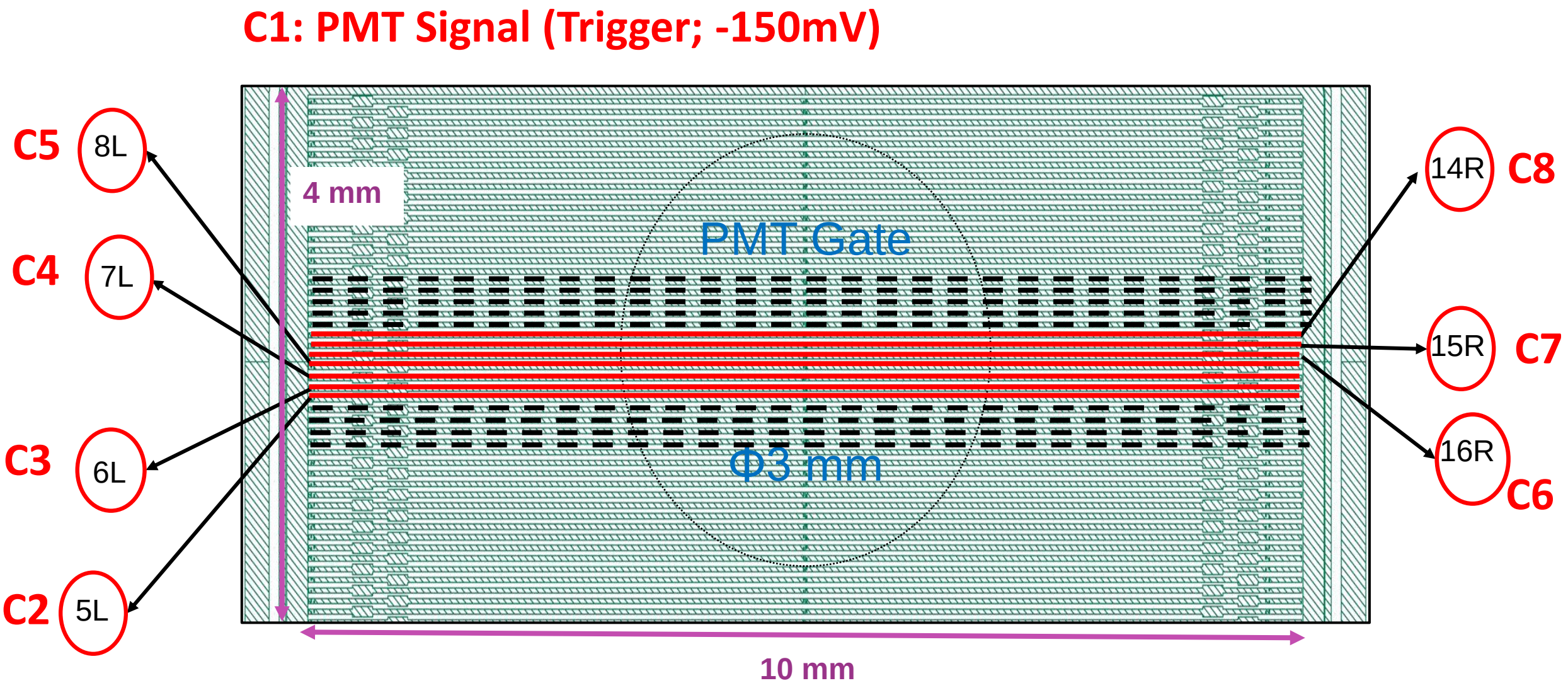


Strip type AC-LGAD
(made by HPK)

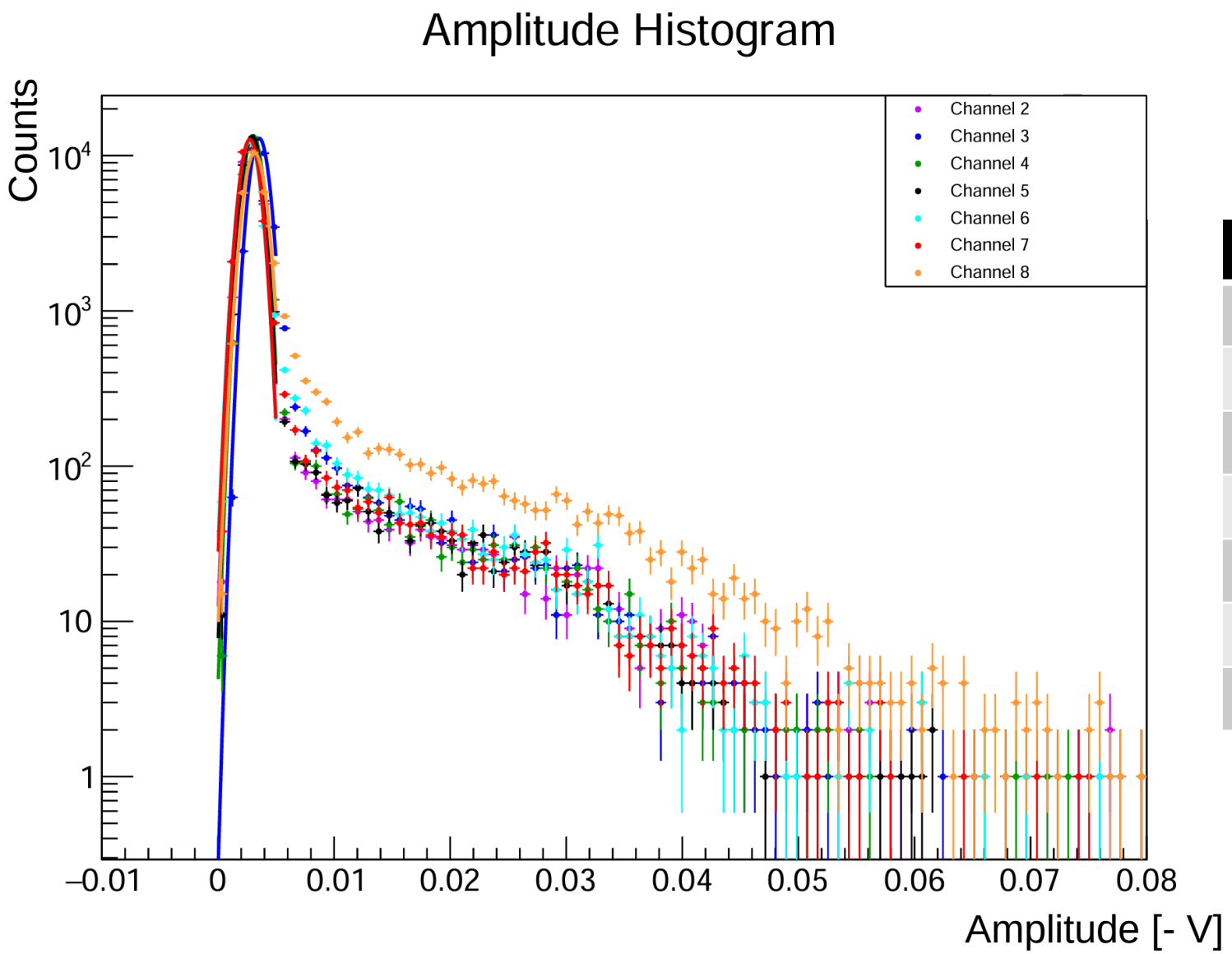




Strip type AC-LGAD Rayout



Max Amplitude Histogram

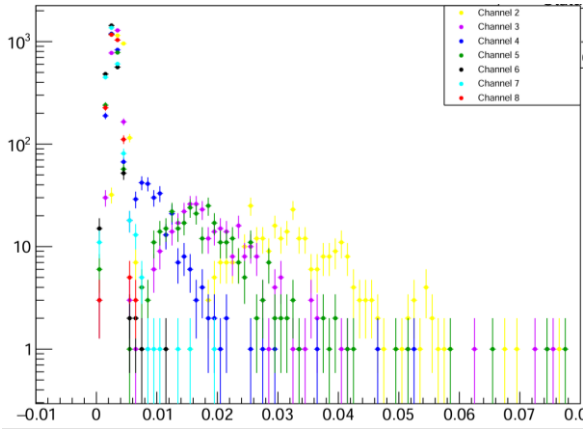


Background Fitting
Range: 0 - 5 mV
Fitting Function: Gaussian

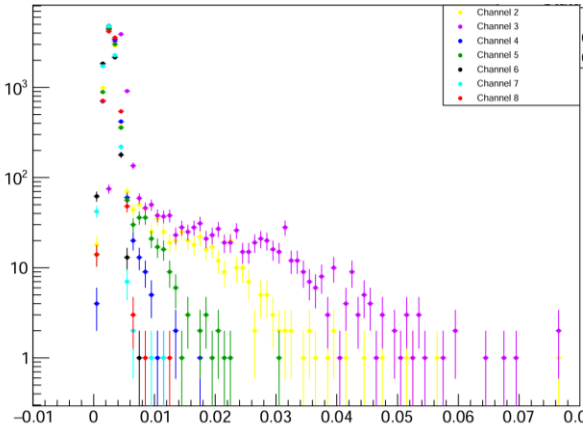
Channel	Mean [mV]	Sigma
2	2.893±0.005	0.769±0.003
3	3.555±0.005	0.761±0.003
4	3.027±0.005	0.748±0.003
5	2.934±0.005	0.748±0.003
6	2.717±0.005	0.786±0.004
7	2.74±0.005	0.777±0.003
8	3.156±0.006	0.839±0.004

Filtered Amplitude Histogram

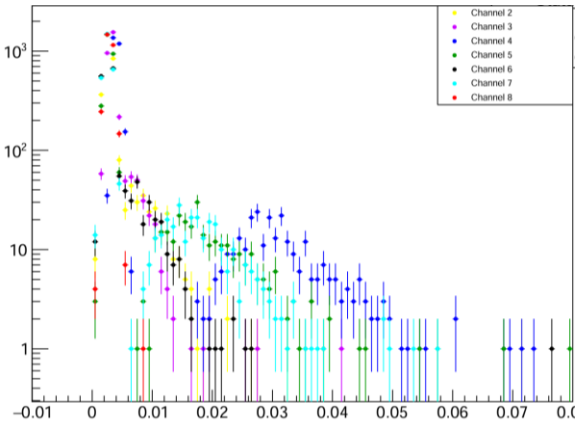
- Total events = 30000
- Filtered by the channel that is max amplitude in each event



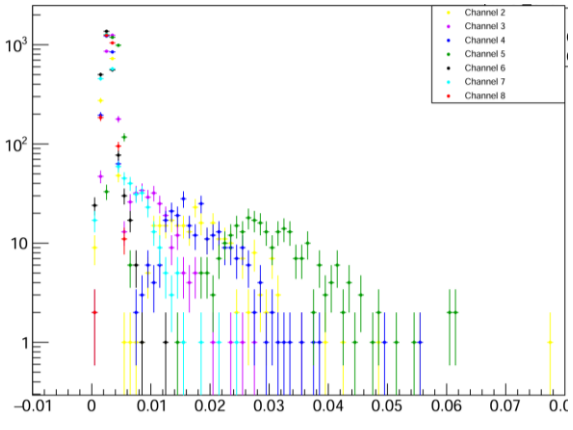
Filtered by ch2



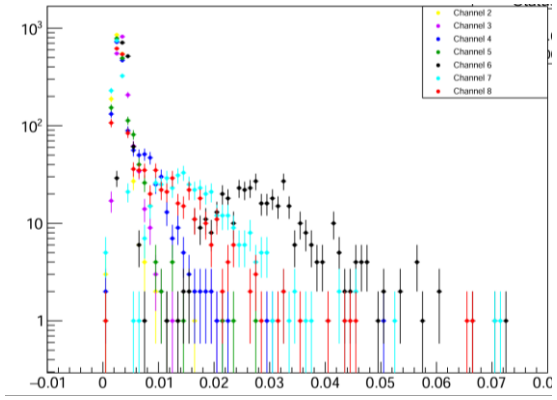
Filtered by ch3



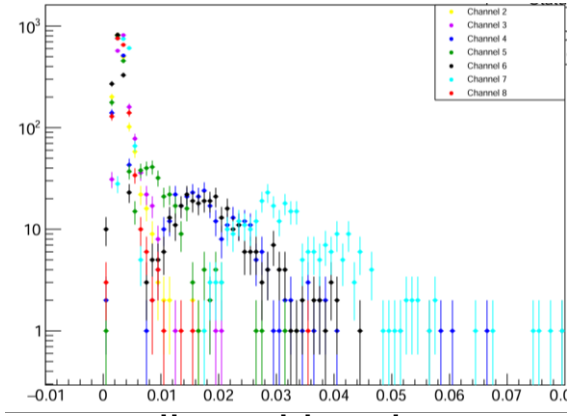
Filtered by ch4



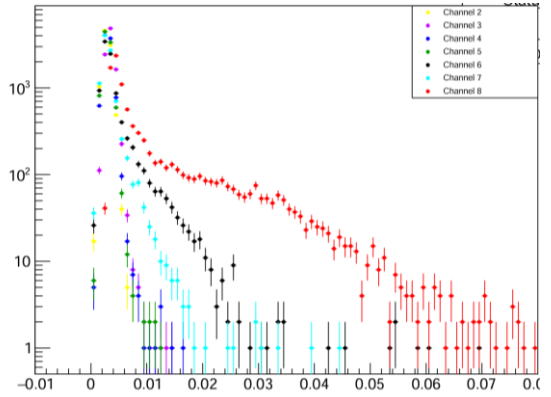
Filtered by ch5



Filtered by ch6

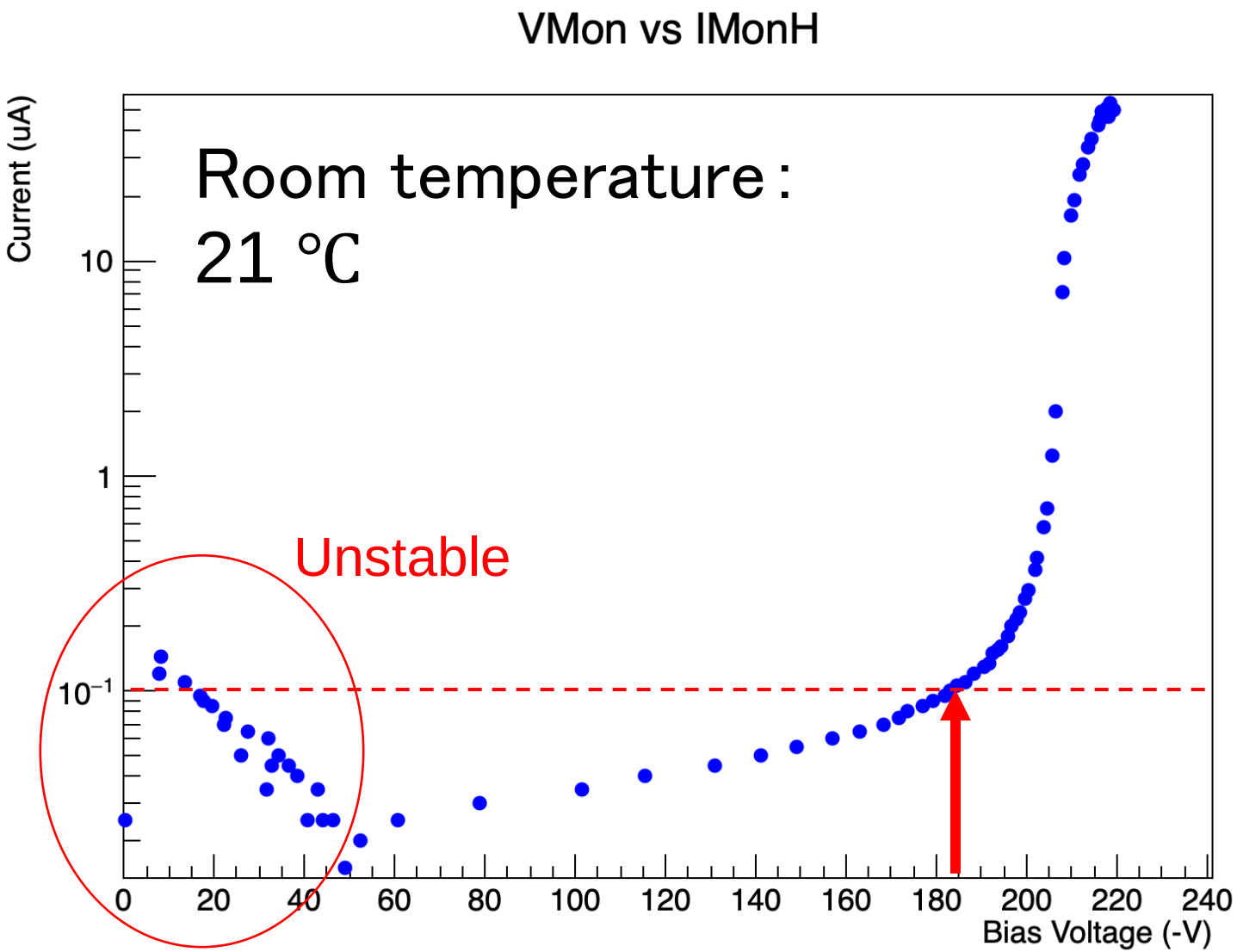


Filtered by ch7



Filtered by ch8

AC-LGAD: Bias Voltage (Temperature dependence)



Purpose

- Large current and voltage applied to AC-LGAD → large load and noise
- Find the optimum value where the signal becomes large without load
- Bias voltage applied to AC-LGAD depends on temperature[1]

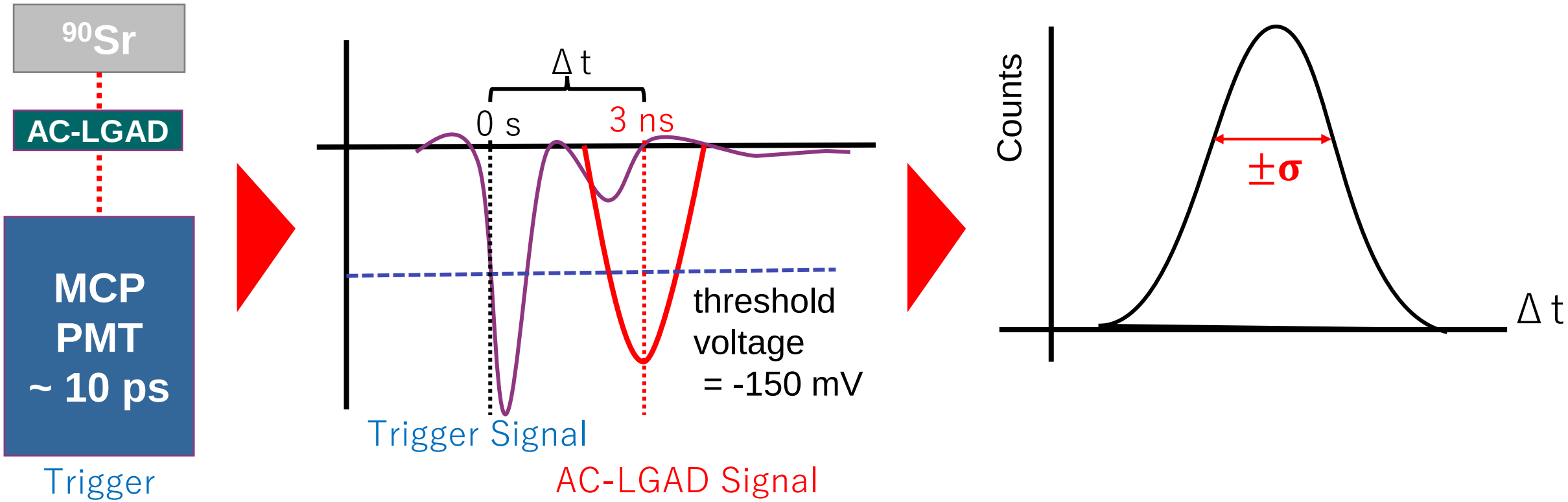
Result

- Current value surges from around 190 V
- High signal & low current → Determined to be **-185 V**
- Unstable behavior at low voltages → depletion layer peculiar to semiconductors is in the process of forming

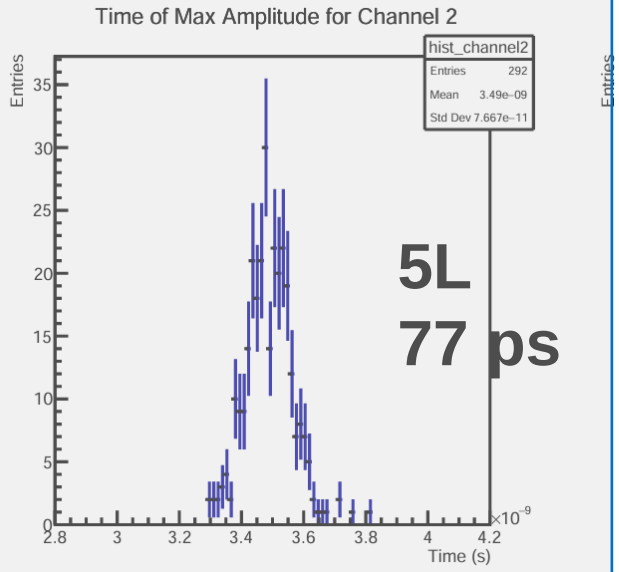
[1] S. Kita et al, Nuclear Inst. and Methods in Physics Research, A 1048 (2023) 168009

How to evaluate time resolution of AC-LGAD

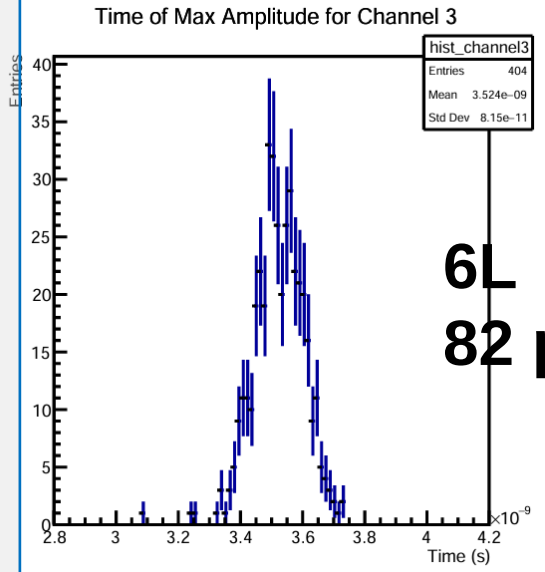
1. β-rays from 90Sr radiation source irradiated to AC-LGAD and trigger (MCP PMT: ~10 ps)
2. Measure the arrival time difference Δt between the trigger signal and the AC-LGAD signal
3. Fit the time distribution of Δt with a Gaussian and calculate the time resolution σ



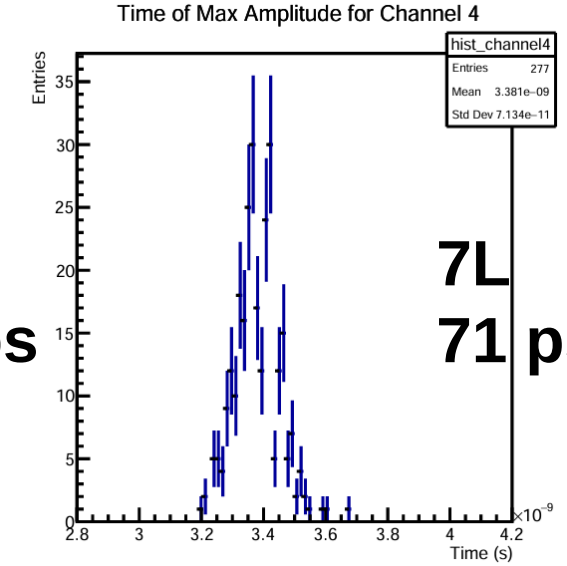
AC-LGAD time resolution (2nd test)



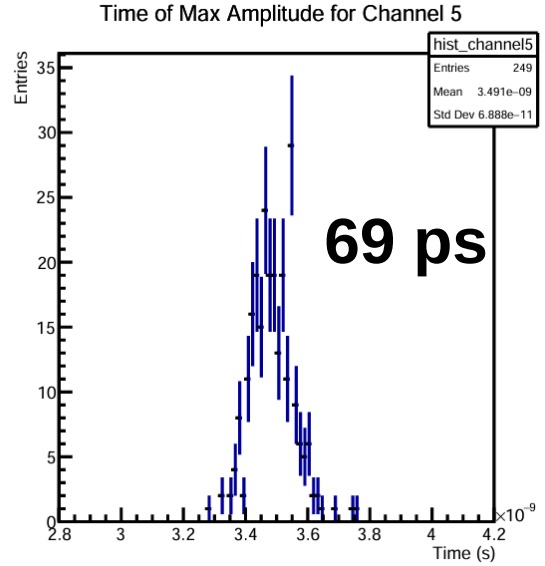
5L
77 ps



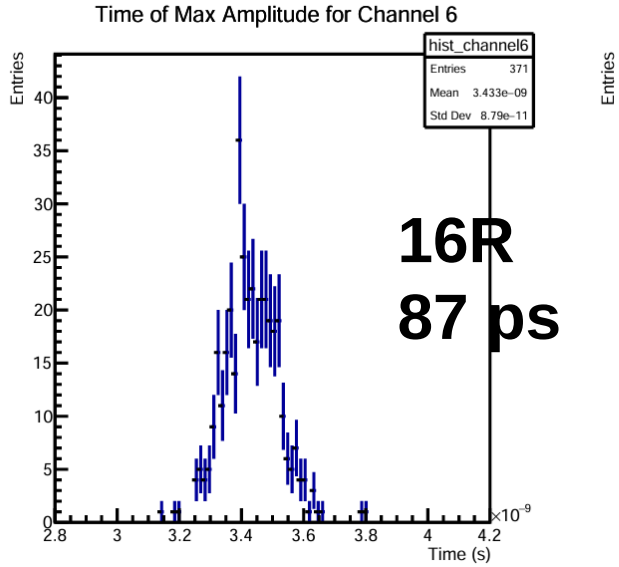
6L
82 ps



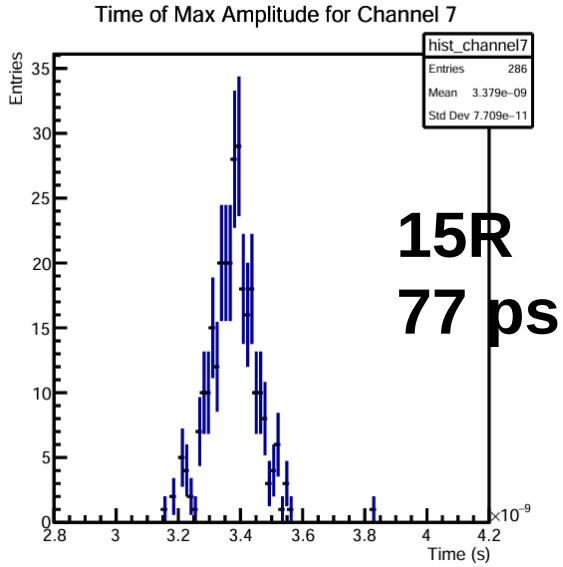
7L
71 ps



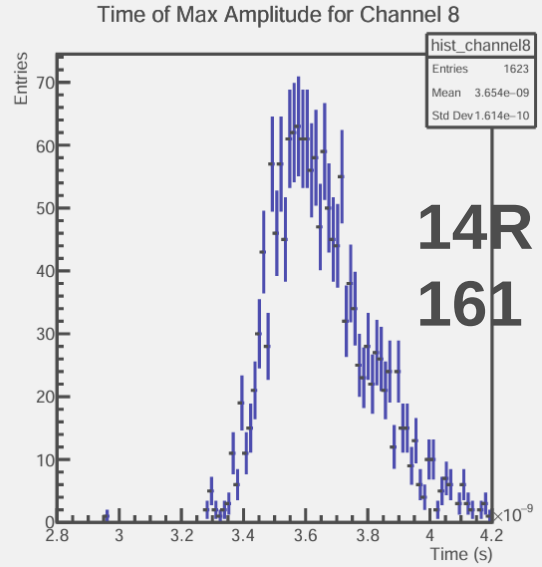
69 ps



16R
87 ps



15R
77 ps



14R
161 ps

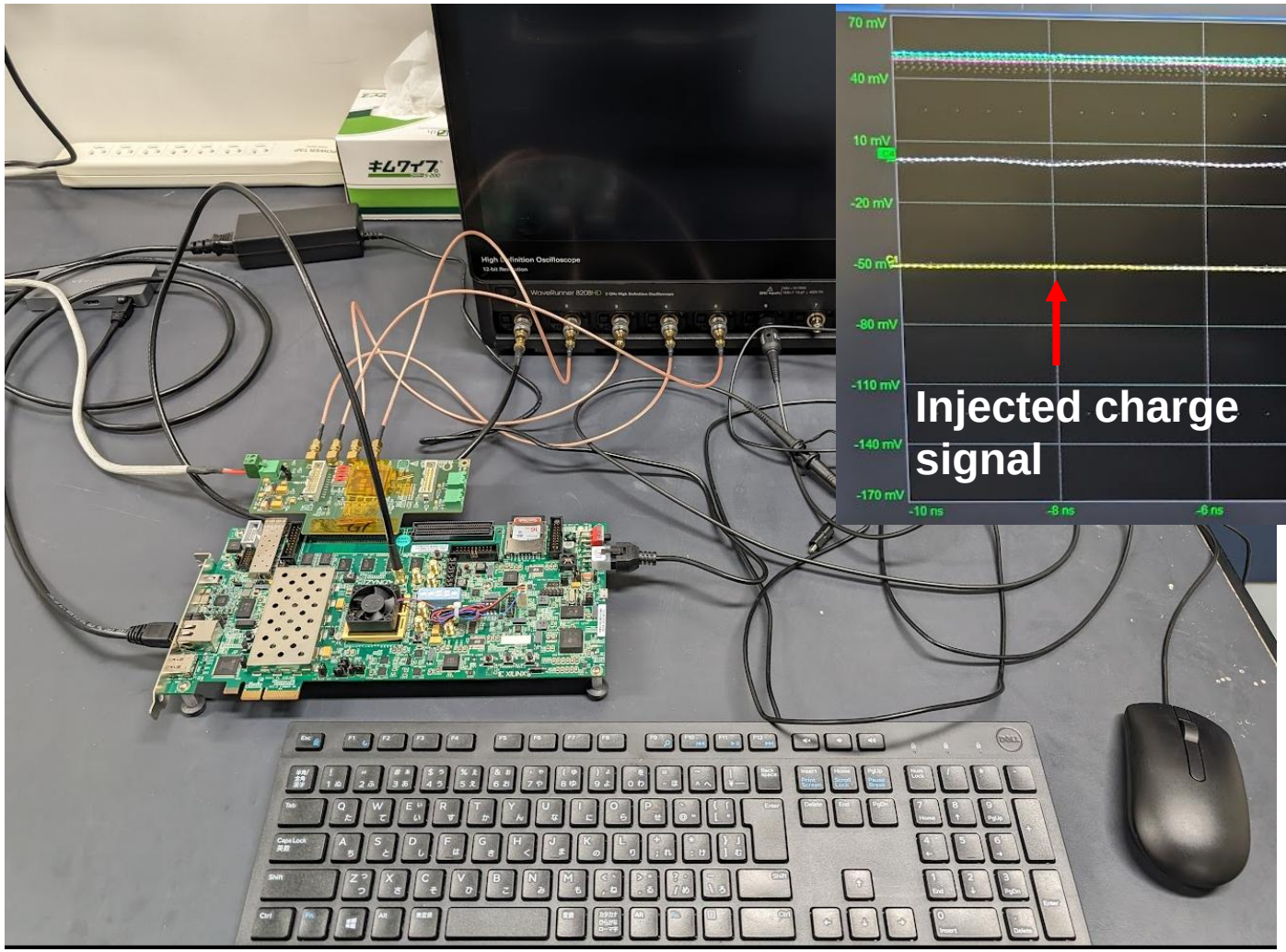
Statistics

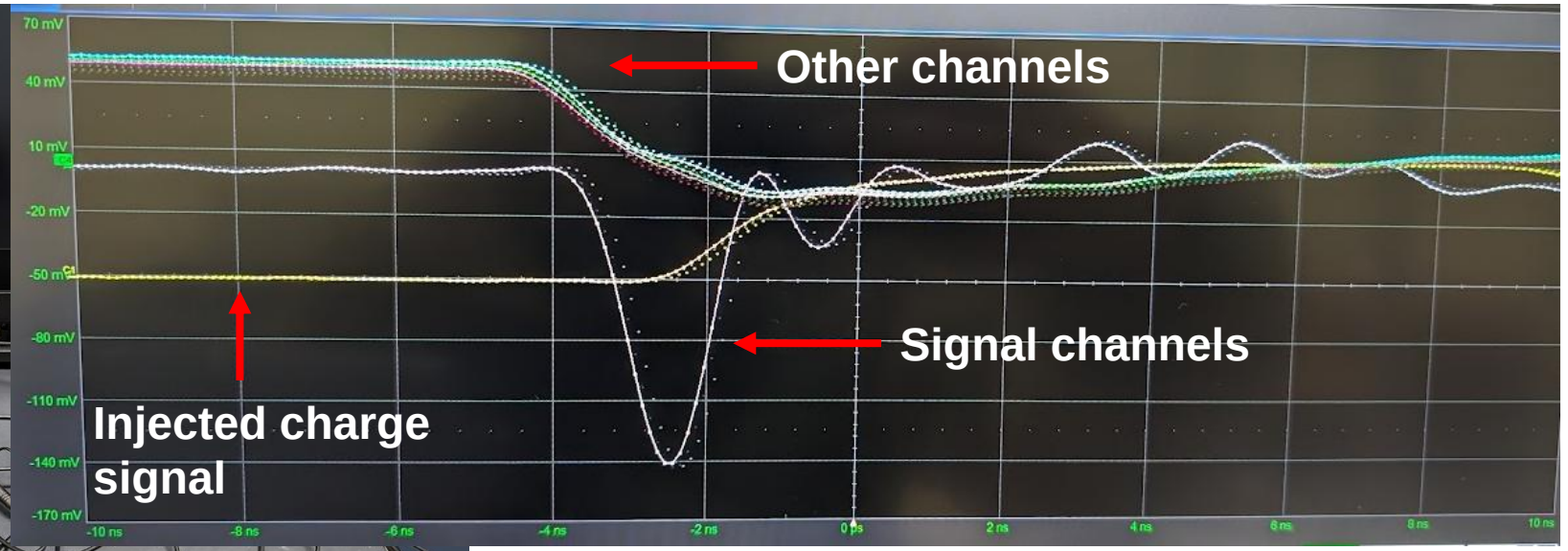
- 100000 events
- Filtered with the maximum ampl. For each events
- Threshold < −10 mV

σ = 70 –90 ps

EICROC0 Analysis @ HU

EICROC0 charge injection Setup



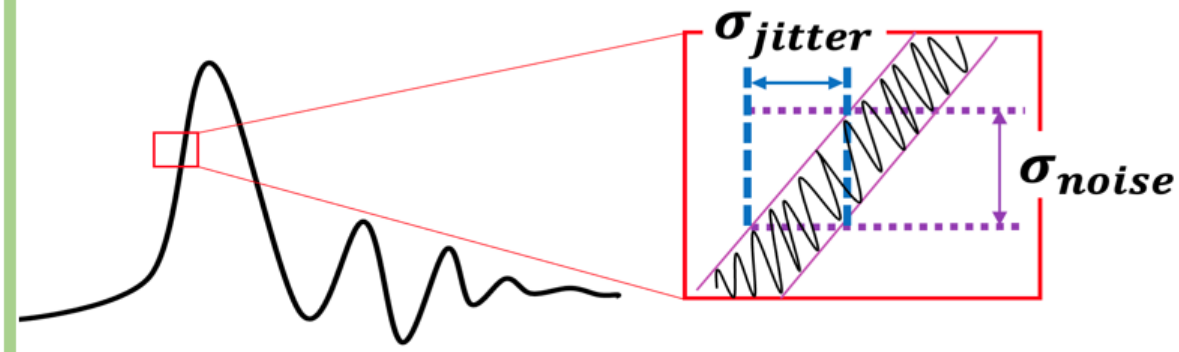


Time jitter Evaluation

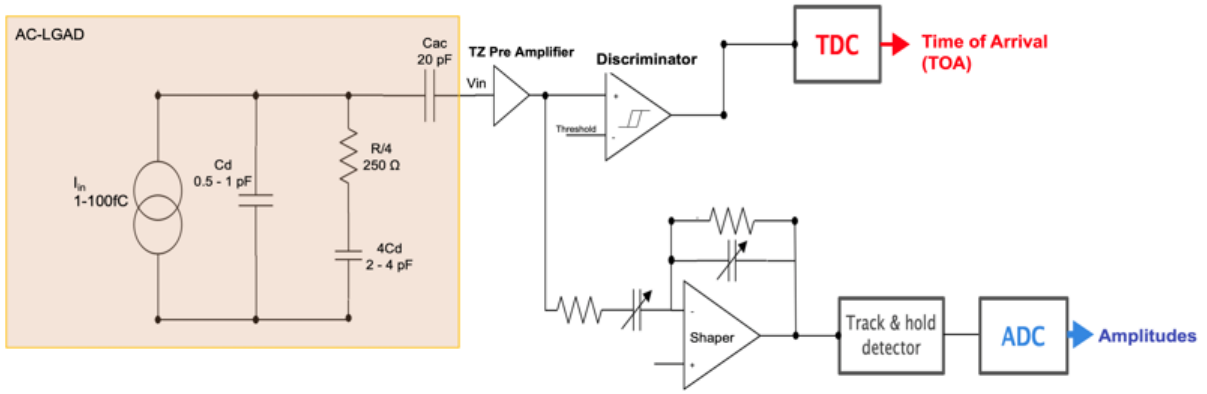
Time jitter

$$\sigma_{jitter} = \frac{\sigma_{noise}}{\left| Signal / t_{rise} \right|}$$

σ_{noise}: noise width, t_{rise}: rise time,
Signal: Max amplitude



Used ASIC… **EICROC0 (16ch)**



AC-LGAD + EICROC0 回路図/ch [4]

- 入力信号から**到達時間と振幅**を出力する集積回路
- Performance：**low jitter < 20 ps**
- AC-LGAD用に開発されたテスト版
  - ➢ 改良版及び新ASICの開発予定あり

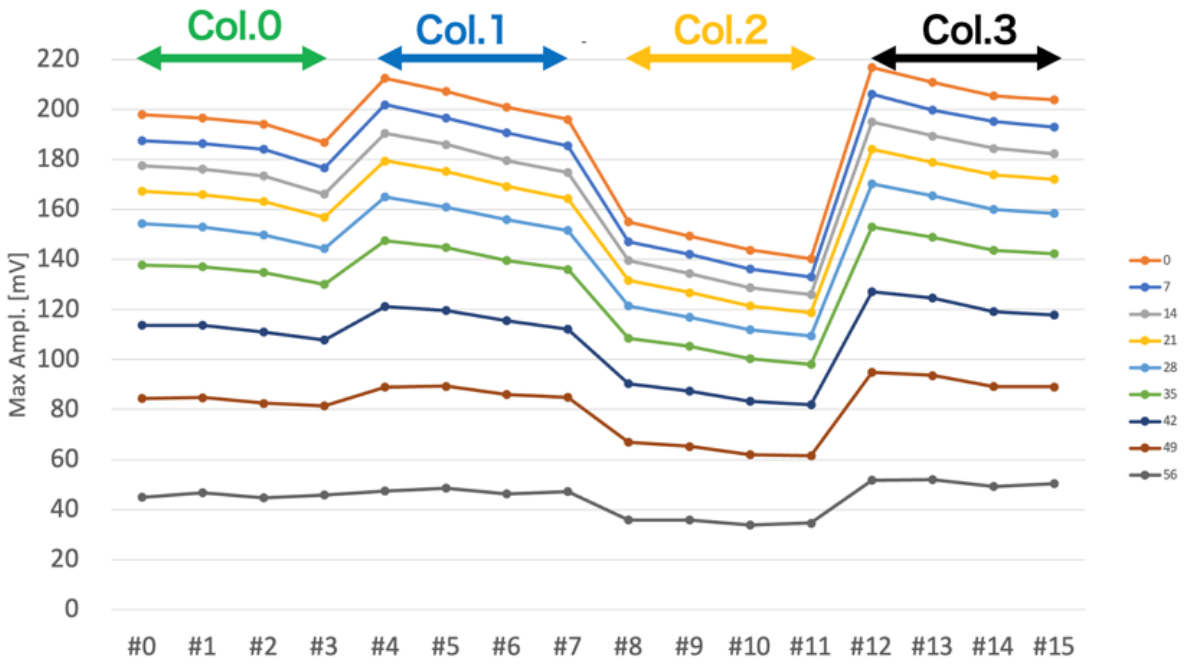
I measured signal, noise, rise time.
Then, I calculated Time jitter by this formula

Time jitter evaluation

Results

Evaluate jitter all of 16ch

■ Max Amplitudes

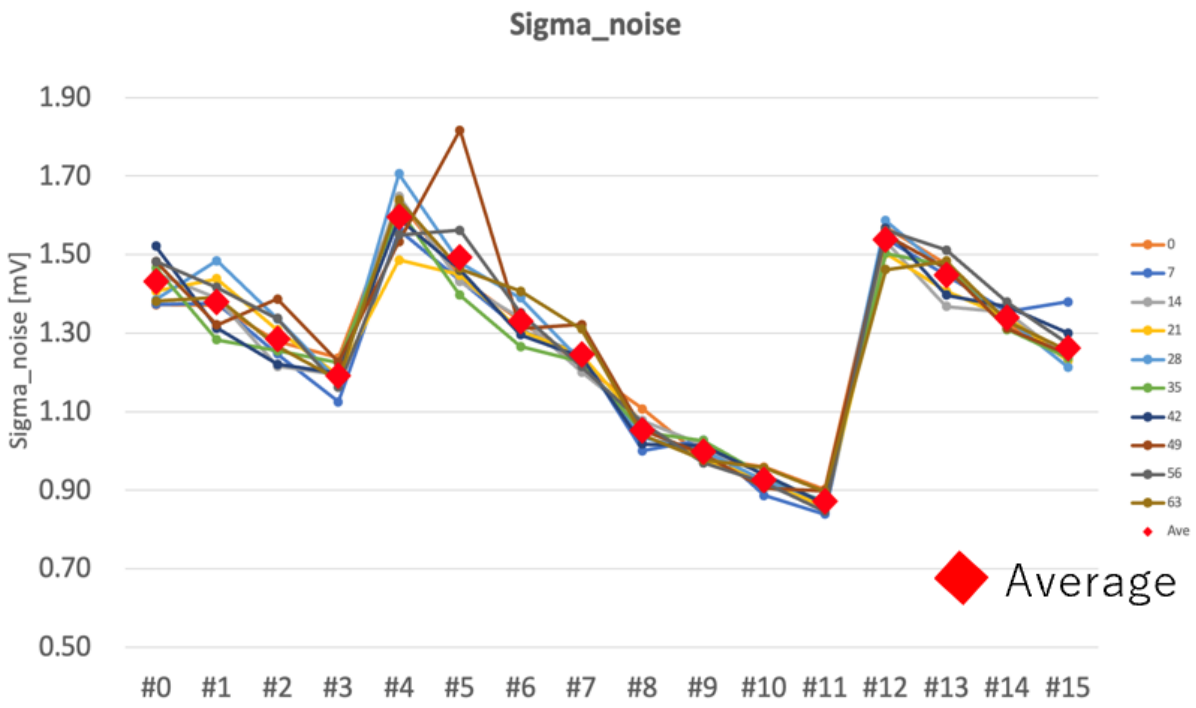


Pixel / Channel Mapping	Column 0	Column 1	Column 2	Column 3
Line 0	Pixel (0,0) #00	Pixel (1,0) #04	Pixel (2,0) #08	Pixel (3,0) #12
Line 1	Pixel (0,1) #01	Pixel (1,1) #05	Pixel (2,1) #09	Pixel (3,1) #13
Line 2	Pixel (0,2) #02	Pixel (1,2) #06	Pixel (2,2) #10	Pixel (3,2) #14
Line 3	Pixel (0,3) #03	Pixel (1,3) #07	Pixel (2,3) #11	Pixel (3,3) #15

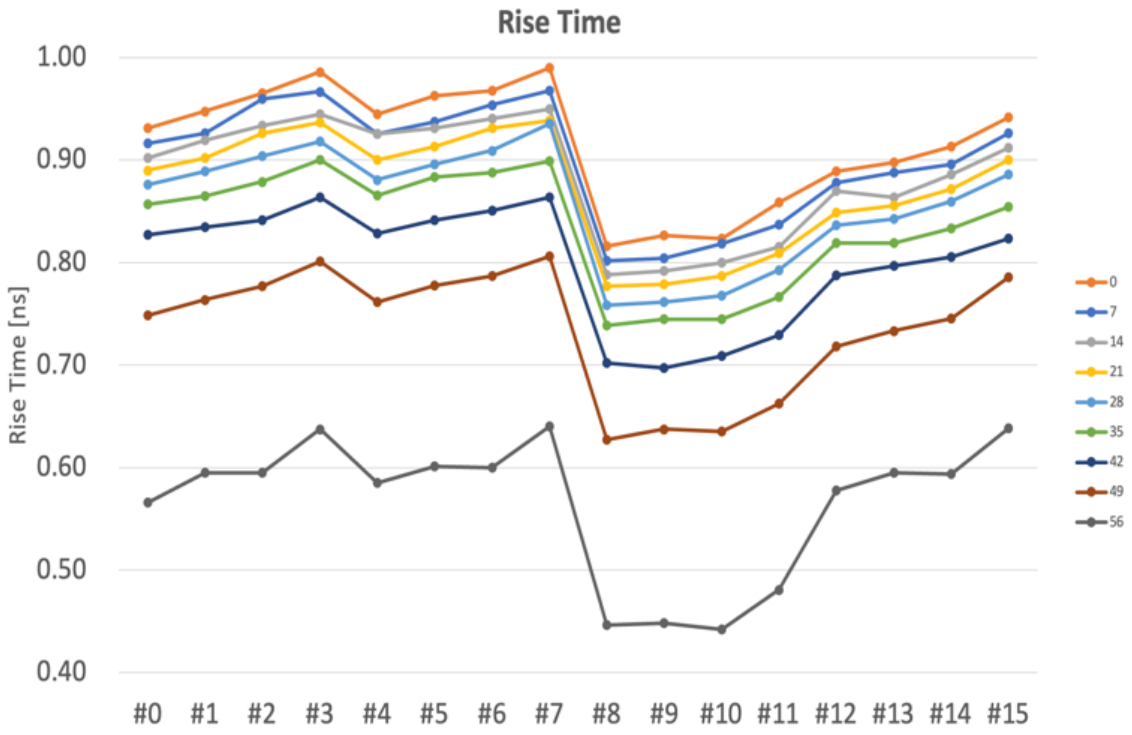
↑ Each channel of EICROC0

Time jitter evaluation

■ Noise sigma

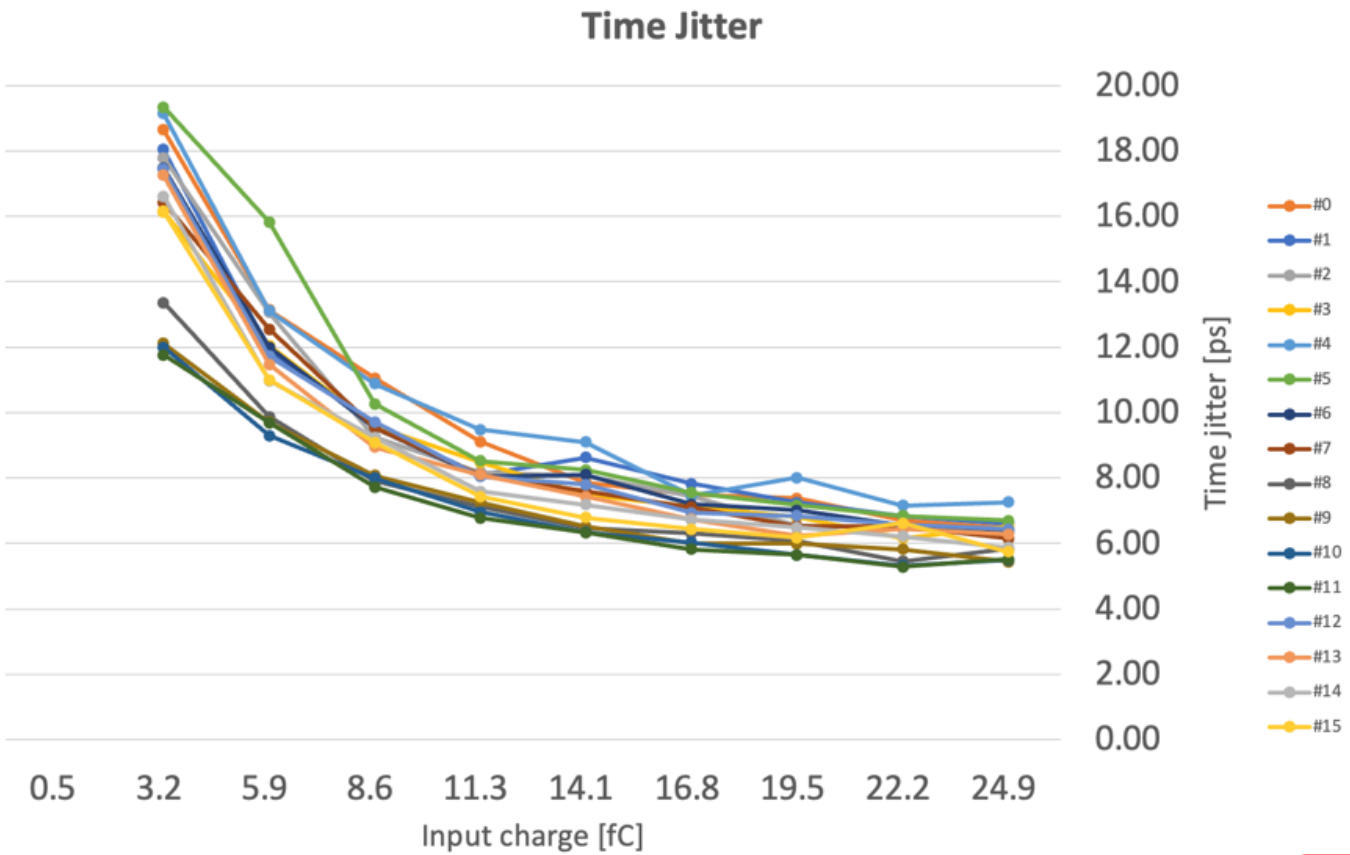


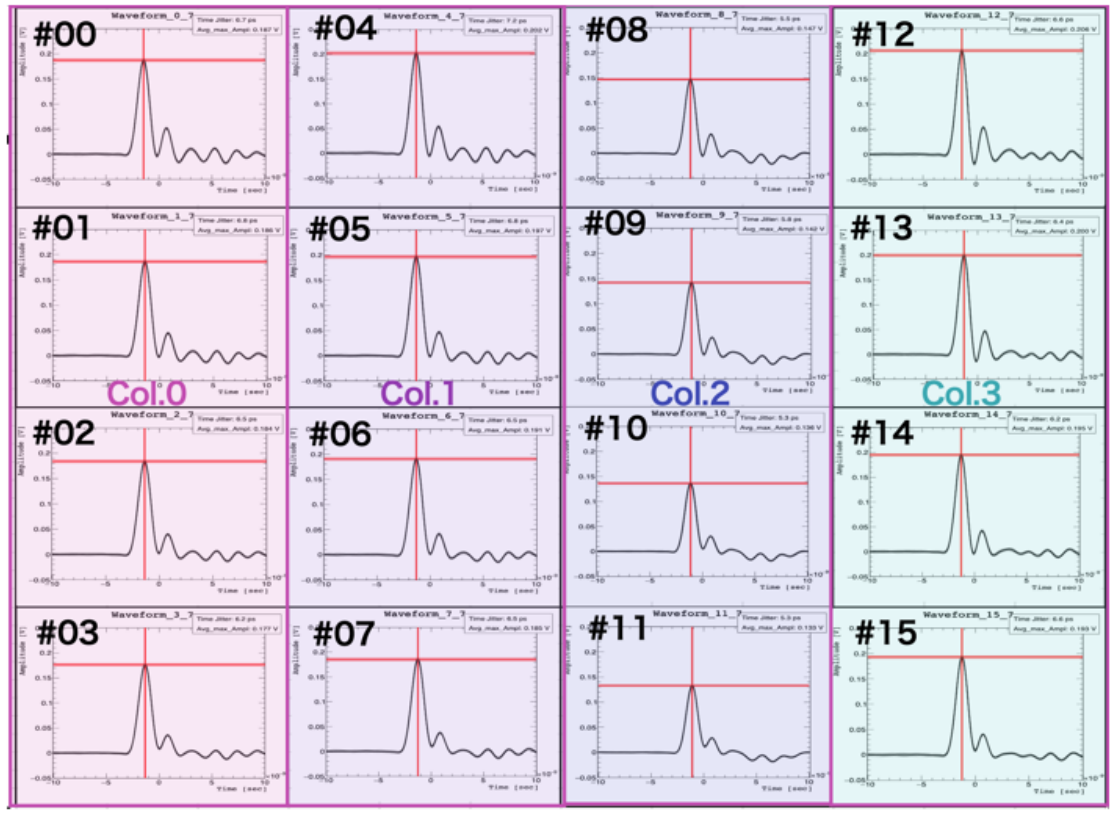
■ Rise time



Time jitter evaluation

■ Time Jitter





↑ Waveform when imputed 22.2 fC

Charge injection 15 fC ↑ ➡ 5 ~ 10 ps

Charge injection6 fC ↓ ➡ 10 ~ 20 ps

(low jitter < 20 ps) is satisfied

Backup