

H2GCROC

2025 - 2026

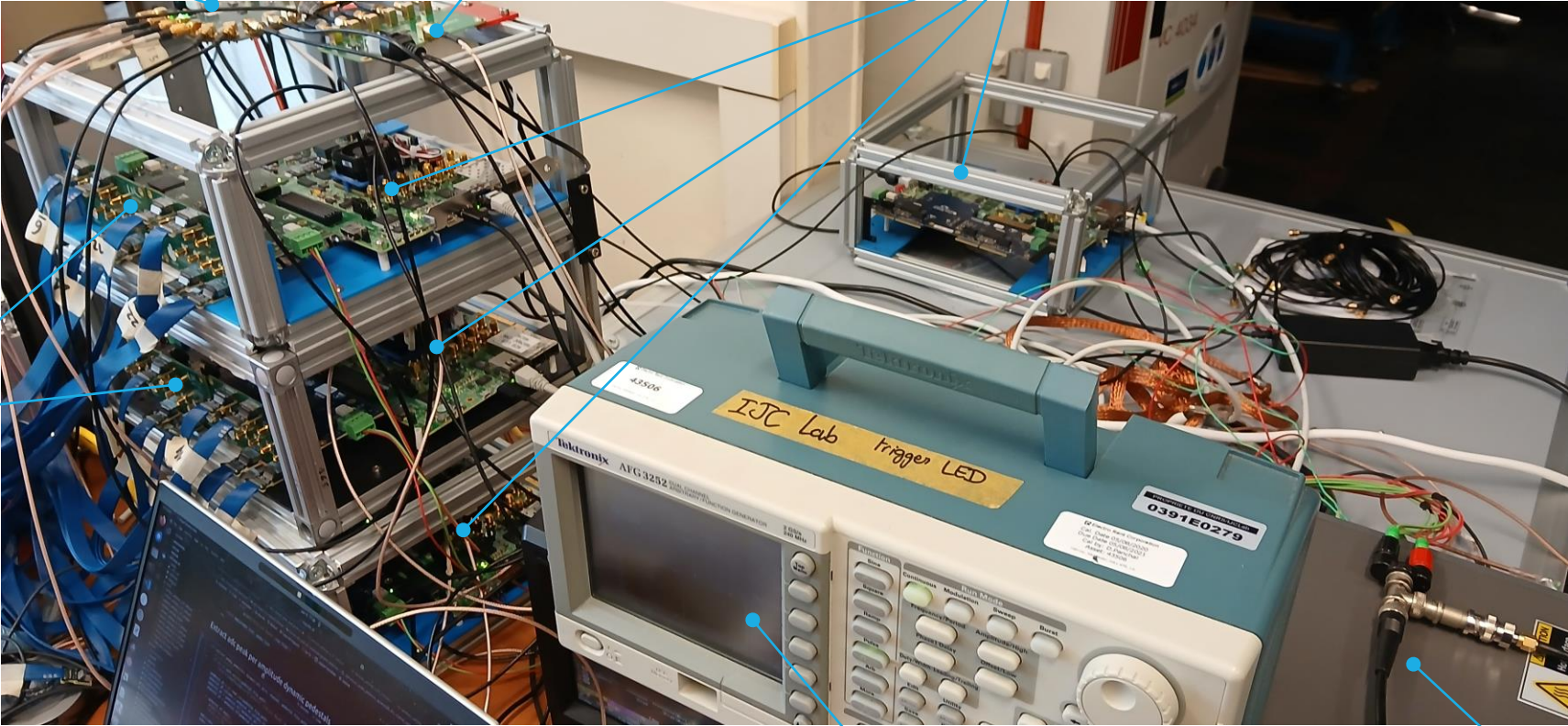
Hardware

Common clock board

Trigger fan out board

KCU105 & protoboard x4

Crystal



Adapter board

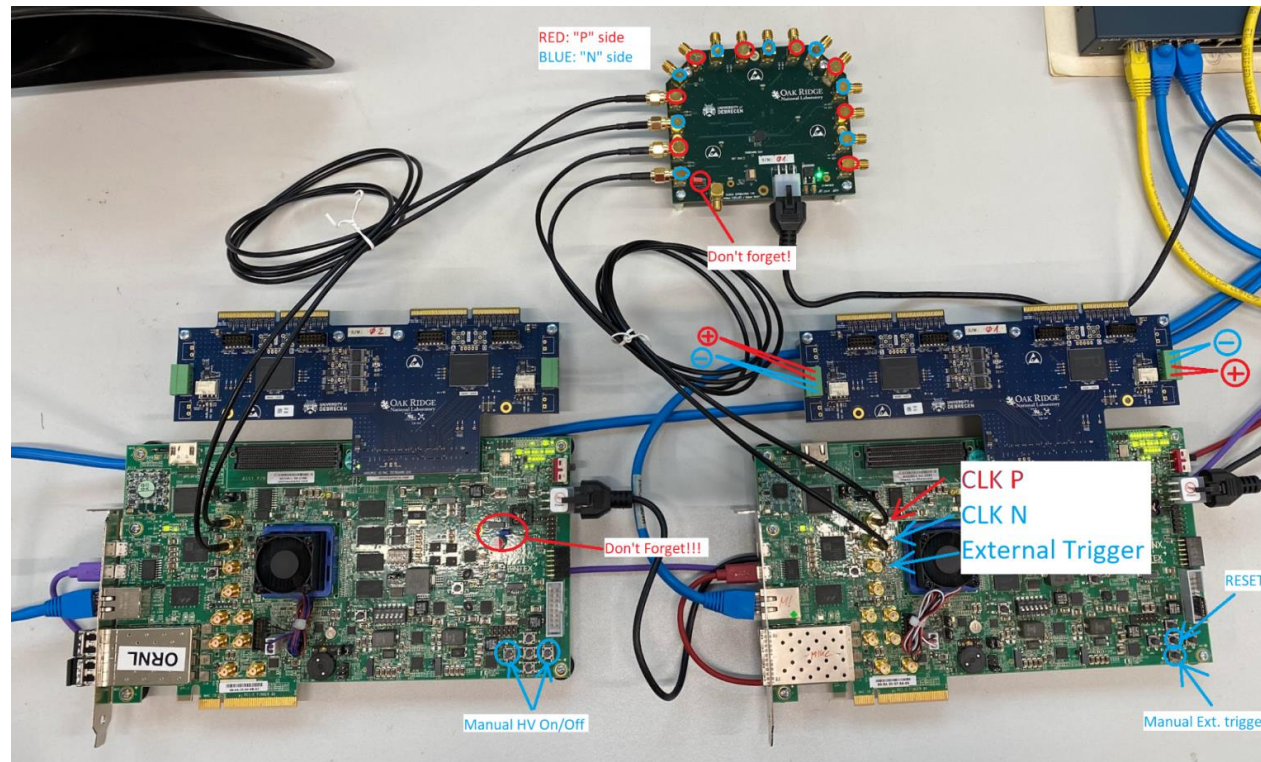
◇ Clock & Trigger & Power cable xN

Signal generator x1

High Voltage Supply

FAQ

- ◇ Must insert jumper J14 on the FPGA board
- ◇ If you have multiple protoboard and FPGA board, you need a common clock input



Programming FPGA

- ◇ Install & Launch Vivado (no need license key)
- ◇ Open Hardware Manager
- ◇ Connect JTAG to PC and power on FPGA board
 - ◇ Click Open target on the left Panel
 - ◇ Open the device
 - ◇ Right click on it and Click “Add Configuration Memory Device”
 - ◇ Select device “mt25qu256-spi-x1_x2_x4” and continue
 - ◇ Select file “frame_protoboard_v2.mcs” (version v4.11)
 - ◇ Program it
- ◇ Firmware: v4.11
 - ◇ <https://web.tresorit.com/l/86zmA#BE8fq4of3N8pFNz-cXpgg&ZZ8blfjDltJSr5qvDdoZYuGsgz6LDm0Q>
- ◇ Programming guide: KCU_Flashing.pdf
 - ◇ <https://web.tresorit.com/l/86zmA#BE8fq4of3N8pFNz-cXpgg&GYtZ9jQtioHk1QwqwK5nizBt2rLRRpSt>

Quick Start

Create Project >

Open Project >

Open Example Project >

Tasks

Manage IP >

Open Hardware Manager >

Vivado Store >

Learning Center

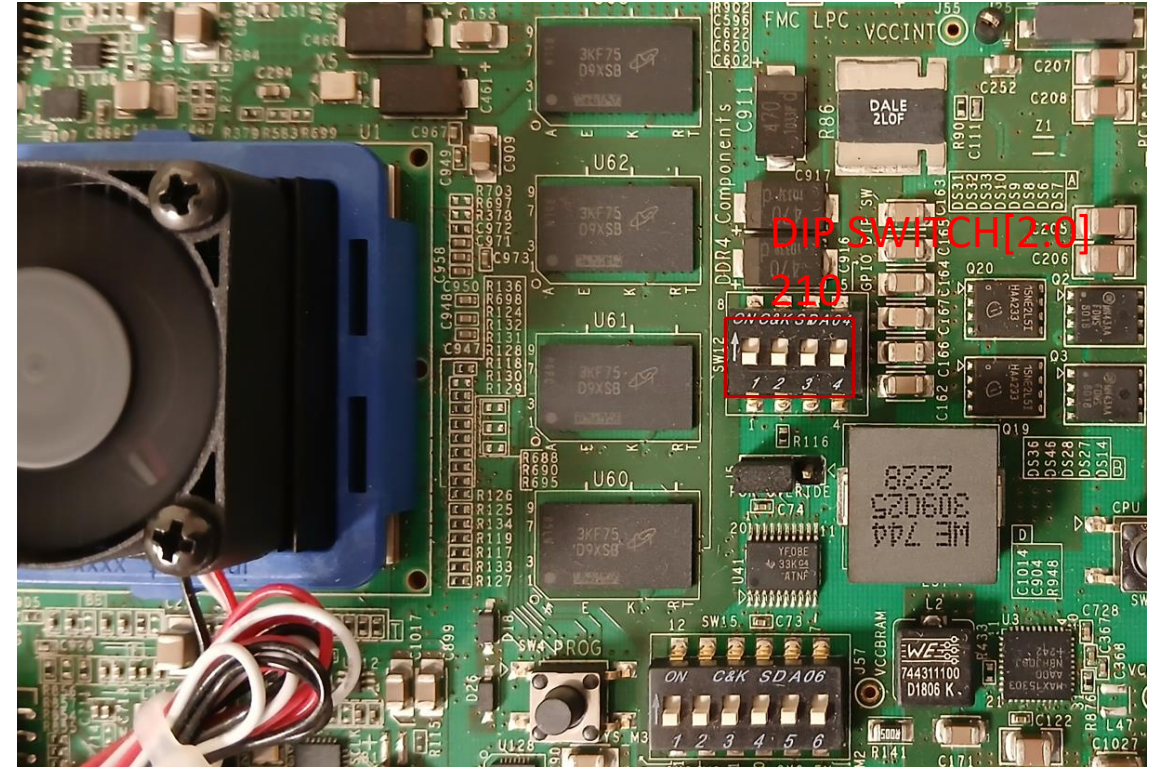
Documentation and Tutorials >

Quick Take Videos >

What's New in 2022.2 >

FPGA IP address

- ◇ DIP Switch on KCU105
- ◇ FPGA IP address = 10.1.2.208 + DIP SWITCH[2:0]



Software environ

- ◇ AlmaLinux 9.5, Use AnaConda
 - ◇ `conda create -n hgc`
 - ◇ `conda activate hgc`
 - ◇ `conda install -c conda-forge pygame pyside6 numpy matplotlib loguru pandas tqdm`
- ◇ Ubuntu 24.04, Use python venv
 - ◇ `sudo apt-get install python3-pip python3-venv libxcb-cursor-dev net-tools`
 - ◇ `mkdir myenv`
 - ◇ `python3 -m venv myenv`
 - ◇ `source myenv/bin/activate`
 - ◇ `pip install pyside6 numpy matplotlib loguru pygame pandas tqdm`
- ◇ Windows 11, Use AnaConda
 - ◇ the same as AlmaLinux 9.5, Use AnaConda, you can install git with conda
- ◇ baremetal, other ...

Software H2GConfig

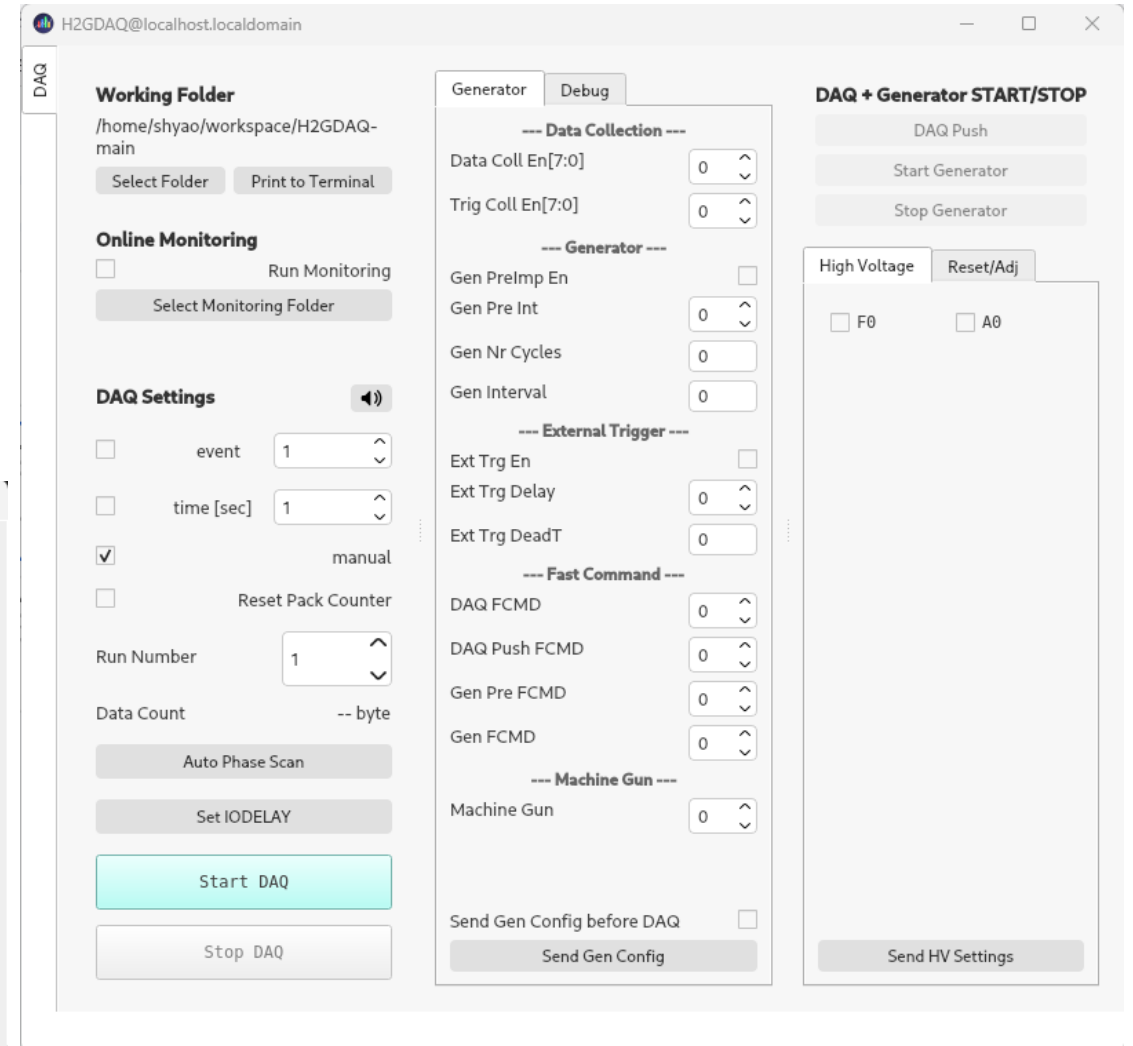
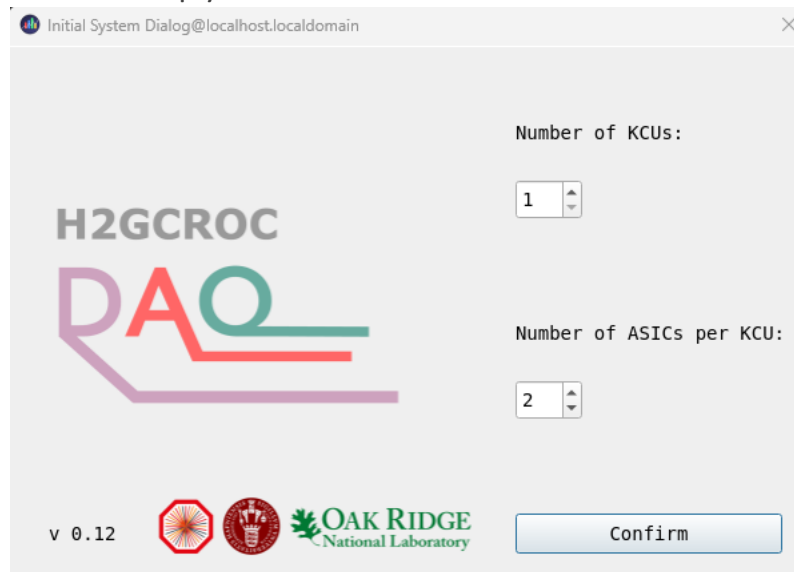
- ◇ H2GConfig
 - ◇ conda activate hgc
 - ◇ cd workspace
 - ◇ git clone <https://gitlab.cern.ch/sjia/H2GConfig.git>
 - ◇ cd ~/workspace/H2GConfig
 - ◇ python H2GConfig.py

5e478442b 2025/2/26

The image shows two windows from the H2GConfig application. The left window, titled 'Initial System Dialog@localhost.localdomain', is a configuration dialog for H2GCROC. It features a large 'H2GCROC Config' logo, a 'Number of KCUs:' field with a spinner set to 1, and a 'Number of ASICs per KCU:' field with a spinner set to 2. At the bottom, there are logos for Oak Ridge National Laboratory and a 'Confirm' button. The right window, titled 'H2GConfig@localhost.localdomain', is the main configuration interface. It has a sidebar with tabs for 'FPGA 0', 'ASIC 0', 'Robot Mode', 'Human Mode', and 'Calibration'. The 'Robot Mode' tab is active, showing a list of registers: 'Top Register' (Top), 'Global Analog Registers' (Global_Analog_1, Global_Analog_0), 'Reference Voltage Registers' (Reference_Voltage_1, Reference_Voltage_0), 'Master TDC Registers' (Master_TDC_1, Master_TDC_0), 'Digital Half Registers' (Digital_Half_1, Digital_Half_0), and 'Channel Wise Registers' (CM_2, CM_3, Channel_L36, Channel_L37, Channel_L38, Channel_L39, Channel_L40, Channel_L41). A central area says 'Select a register to view its content'. On the right, there are controls for 'Current Config', 'FPGA selection: FPGA 0', 'ASIC selection: ASIC 0', 'Register Selection' (checkboxes for 'Use half-wise registers' and checked 'Use channel-wise registers'), 'Config File Info' (File Not Set), and buttons for 'Load', 'Save', 'Save As', 'Readback', 'Ping FPGA', 'Send Current ASIC Config', and 'Send All Configs'. The bottom right corner shows 'Link Info' with IP Address: 10.1.2.208 and Port: 11000.

Software H2GDAQ

- ◇ The same virtual environ work for H2GDAQ
 - ◇ conda activate hgc
 - ◇ cd workspace
 - ◇ git clone <https://gitlab.cern.ch/sjia/H2GDAQ.git> 8a17963 2025/4/25
 - ◇ cd ~/workspace/H2GDAQ
 - ◇ git checkout dev-0v12
 - ◇ python H2GDAQ.py



Software H2GCalib

- ◇ Create a virtual environ for H2GCalib or add package to hgc environ
 - ◇ conda activate hgc
 - ◇ conda install -c conda-forge pygame pyside6 numpy matplotlib loguru tqdm pandas
 - ◇ cd workspace
 - ◇ git clone https://gitlab.cern.ch/sjia/H2GCalib_3B 8dce2723fb 2025/5/9
 - ◇ cd ~/workspace/H2GCalib_3B
- ◇ Usage 1: GUI mode
 - ◇ python 100_UI.py
- ◇ Usage 2: console mode
 - ◇ Python 000_SocketPool.py
 - ◇ ...
 - ◇ Python 010_ToTCalib.py



Ethernet communication

- ◇ Connect ethernet cable between PC & FPGA board
- ◇ Set PC IP address to 10.1.2.207
 - \$ IF=enp4s2
 - \$ sudo ifconfig \${IF} 10.1.2.207 netmask 255.255.255.0
 - \$ sudo ifconfig \${IF} mtu 9000 up
- ◇ make sure your switch, cable and network card support 1 Gbps speed, 100 Mbps will not work
 - \$ sudo ethtool \${IF} | grep Speed
 - Speed: 1000Mb/s
- ◇ H2GConfig has a button “Ping FPGA” can test the connection, it don't support under windows
- ◇ Ping
 - ◇ On Linux: ping 10.1.2.208 -s 32
 - ◇ On Windows: ping 10.1.2.208 -l 32

The screenshot displays the H2GConfig application window titled "H2GConfig@localhost.localdomain". The interface is divided into several sections:

- Left Panel:** A vertical sidebar with buttons for "FPGA 0", "ASIC 0", "Robot Mode", "Human Mode", and "Calibration".
- Main Panel:** A list of registers under various categories:
 - Top Register --: Top
 - Global Analog Registers --: Global_Analog_1, Global_Analog_0
 - Reference Voltage Registers --: Reference_Voltage_1, Reference_Voltage_0
 - Master TDC Registers --: Master_TDC_1, Master_TDC_0
 - Digital Half Registers --: Digital_Half_1, Digital_Half_0
 - Channel Wise Registers --: CM_2, CM_3, Channel_36, Channel_37, Channel_38, Channel_39, Channel_40, Channel_41
- Right Panel:** Configuration options:
 - Current Config --: FPGA selection: FPGA 0, ASIC selection: ASIC 0
 - Register Selection --: Use half-wise registers, Use channel-wise registers
 - Config File Info --: not set, File Not Set
 - Link Info --: IP Address: 10.1.2.208, Port: 11000

A modal dialog box titled "Ping Result@localhost.localdomain" is overlaid on the main panel, displaying the message "Successfully pinged FPGA 5 times" and an "OK" button. The "Ping FPGA" button in the right panel is highlighted with a red border.

Ethernet communication

- ◇ Network Optimization (optional)
- ◇ Create config file 50-h2gcroc.conf and add the following lines:
 - ◇ net.core.rmem_max=8388608
 - ◇ net.core.wmem_max=8388608
 - ◇ net.core.rmem_default=8388608
 - ◇ net.core.wmem_default=8388608
 - ◇ net.core.netdev_max_backlog=250000
 - ◇ net.ipv4.udp_mem=8388608 8388608 8388608
 - ◇ net.ipv4.udp_rmem_min=16384
 - ◇ net.ipv4.udp_wmem_min=16384
- ◇ Then run the following command to apply the changes:
 - ◇ sudo cp 50-h2gcroc.conf /etc/sysctl.d/
 - ◇ sudo sysctl -p

Ethernet Firewall

◇ AlmaLinux 9.5

◇ DAQ and Calibration programs use UDP port 11000 ~

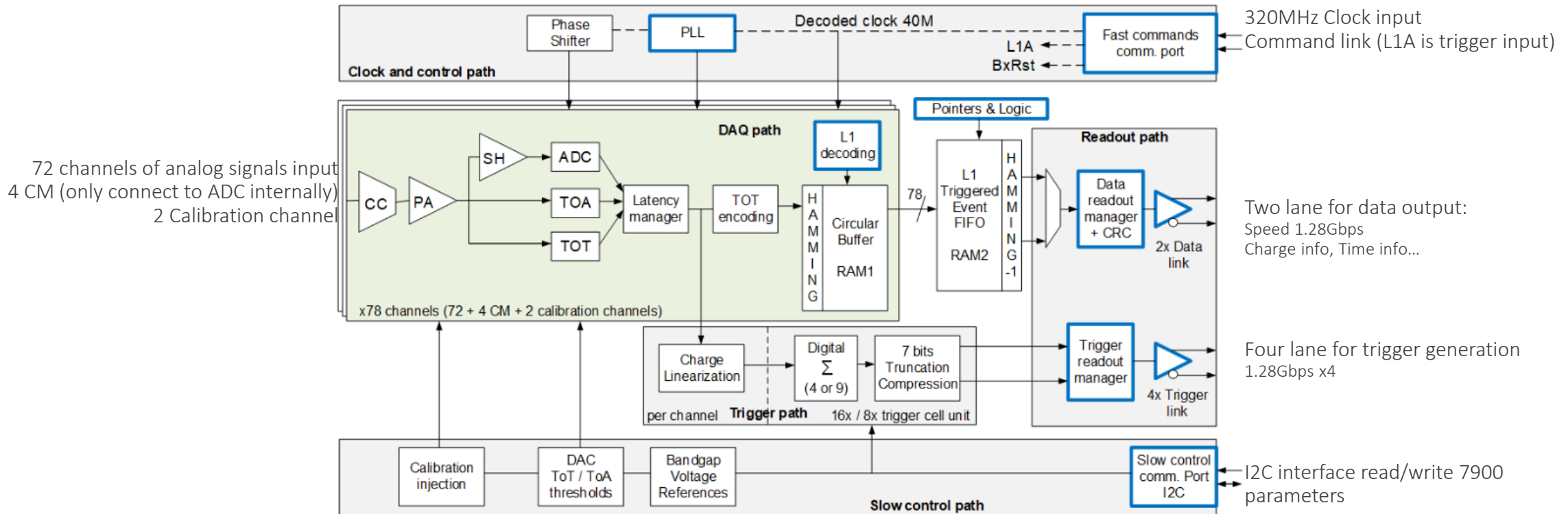
```
[root@localhost ~]# firewall-cmd --zone=public --add-interface=enp87s0 --permanent
success
[root@localhost ~]# firewall-cmd --zone=public --add-port=11000-11100/udp --permanent
success
[root@localhost ~]# firewall-cmd --reload
success [root@localhost ~]# firewall-cmd --zone=public --list-all
public (active)
  target: default
  icmp-block-inversion: no
  interfaces: enp86s0 enp87s0
  sources:
  services: cockpit dhcpv6-client ssh
  ports: 11000-11100/udp
  protocols:
  forward: yes
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
```

Note interface enp86s0 connect to office network
interface enp87s0 is for h2gc data transfer

Architectural

1 H2GCROC3b: architectural overview

The overall block diagram is described as per below.



CC: Current conveyor (VFE), calibration SiPM gain variation
 PA: Pre-amplifier
 TOT: Time Over Threshold, dedicated discriminator and TDC (50ps), 12 bit
 TOA: Time Of Arrival, dedicated discriminator and TDC (24ps), 12 bit
 SH: Shaper
 ADC: 10bit 40MHz SARADC

Default register value

The default config value of H2GCROC after power up

```
"Channel_0-71          ": "00 00 00 00 00 00 00 00 00 00 00 00 00 00 00",
"CM_0-3               ": "00 00 00 00 00 00 00 00 00 00 00 00 00 00 00",
"CALIB_0-1            ": "00 00 00 00 00 00 00 00 00 00 00 00 00 00 00",

"Global_Analog_0-1    ": "cf d3 83 28 28 00 9a 9a a2 8a 50 4b 4b 68 6f",
"Reference_Voltage_0-1": "dc 00 3c 1c 58 d0 00 00 00 00 00 00 00 00",
"Master_TDC_0-1       ": "37 d4 40 80 08 b0 02 00 00 80 08 b0 02 00 00 00",
"Digital_Half_0-1     ": "00 00 00 00 80 00 00 00 00 00 00 00 00 19 00 08 cc cc cc 0c cc cc cc cc 0f 01 00",
"HalfWise_0           ": "00 00 00 00 00 00 00 00 00 00 00 00 00 00 00",

"Top                  ": "08 0f 40 31 00 03 85 00 00 00 00 00 00 00 00 00 00 00 00 00"
```

Total 1363 Bytes

Operation Procedure

- ◇ H2GCROC can generate a pulse signal internally and inject it to input channel, with adjustable amplitude
- ◇ H2GCalib_3B
 - ◇ Set IODELAY – adjust the data/trigger line delay between FPGA and H2GCROC, this parameter is provided for FPGA, not for H2GCROC.
 - ◇ ADC pedestal
 - ◇ TOA calibration – adjust the channel threshold to make the output value close to the user's target value
 - ◇ TOT calibration – adjust the channel threshold to make the output value close to the user's target value
- ◇ Repeat this procedure for all FPGA and H2GCROC combination and generate configuration files

Operation Procedure

- ◇ H2GDAQ
 - ◇ Set IODELAY – this step is the same as calibration
- ◇ H2GConfig
 - ◇ Send All configs – upload all calibration values to all H2GCROC
- ◇ H2GDAQ
 - ◇ Start DAQ
- ◇ Decode data or Online Monitoring

H2GCalib_3B

- ◇ Modify file H2GCalib_3B/config/common_settings_3B.json

```
{
  "udp": {
    "h2gcroc_ip": "10.1.2.208",
    "pc_ip": "10.1.2.207",
    "h2gcroc_port": 11000,
    "pc_cmd_port": 11000,
    "pc_data_port": 11001
  }
}
```

H2GCalib_3B – Socket Pool

◇ python H2GCalib_3B/ 000_SocketPool.py

[Pool] Control → 127.0.0.1:6002

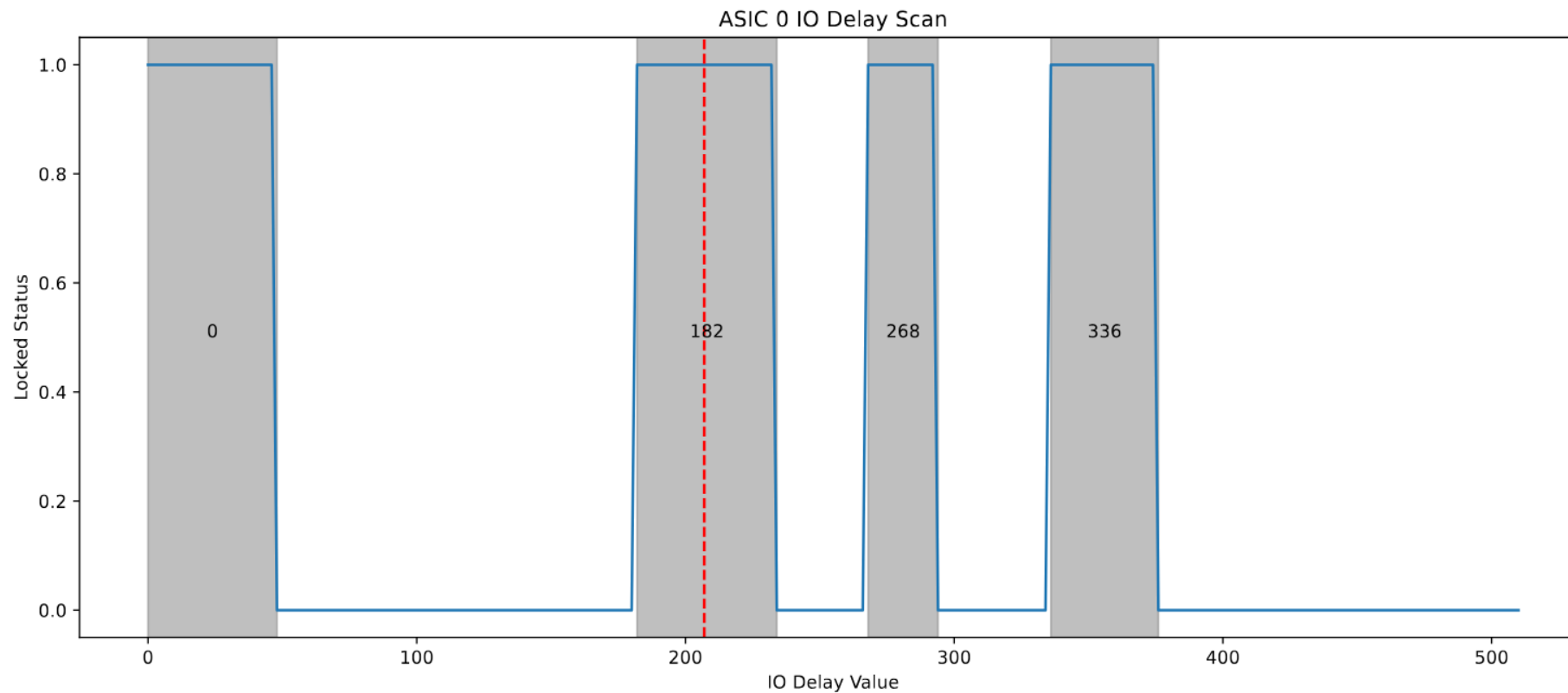
[Pool] Data → 127.0.0.1:6001

...

◇ This program maintains multiple network connections, and other calibration programs use this as a relay, keep this program running in the background

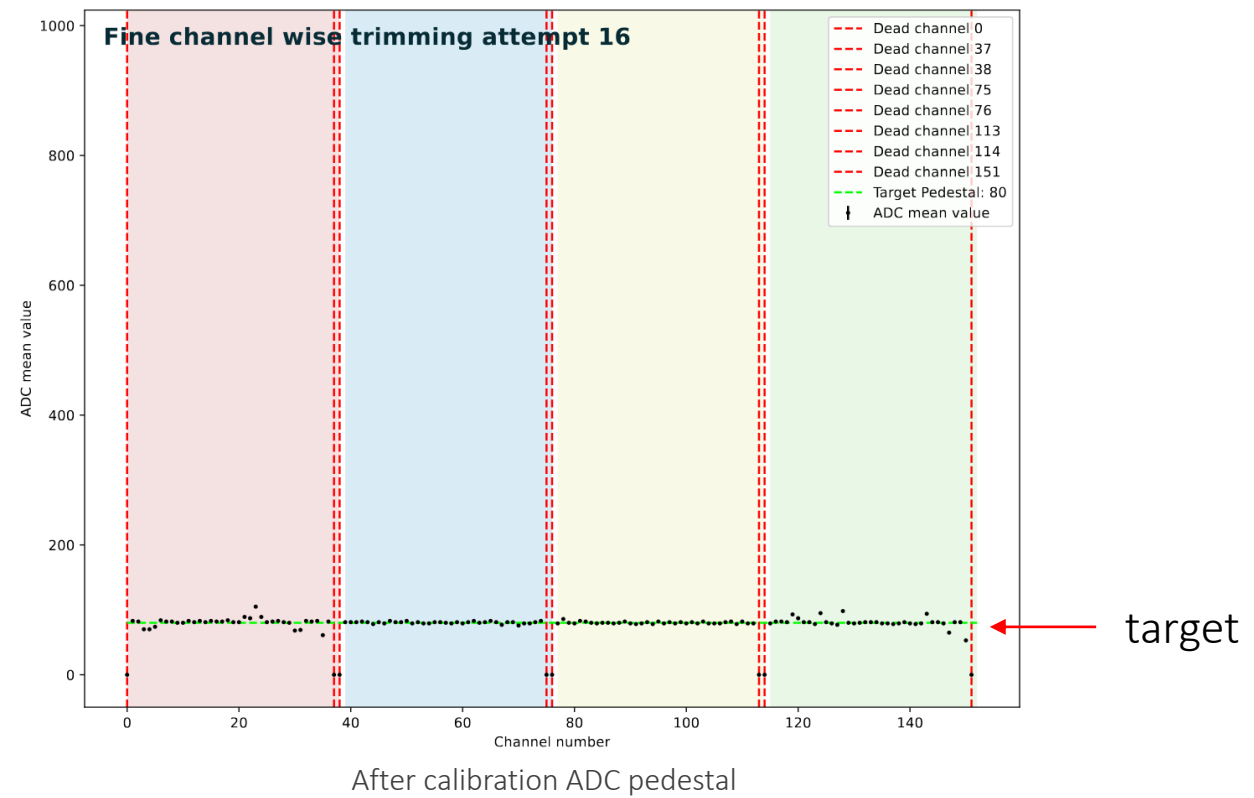
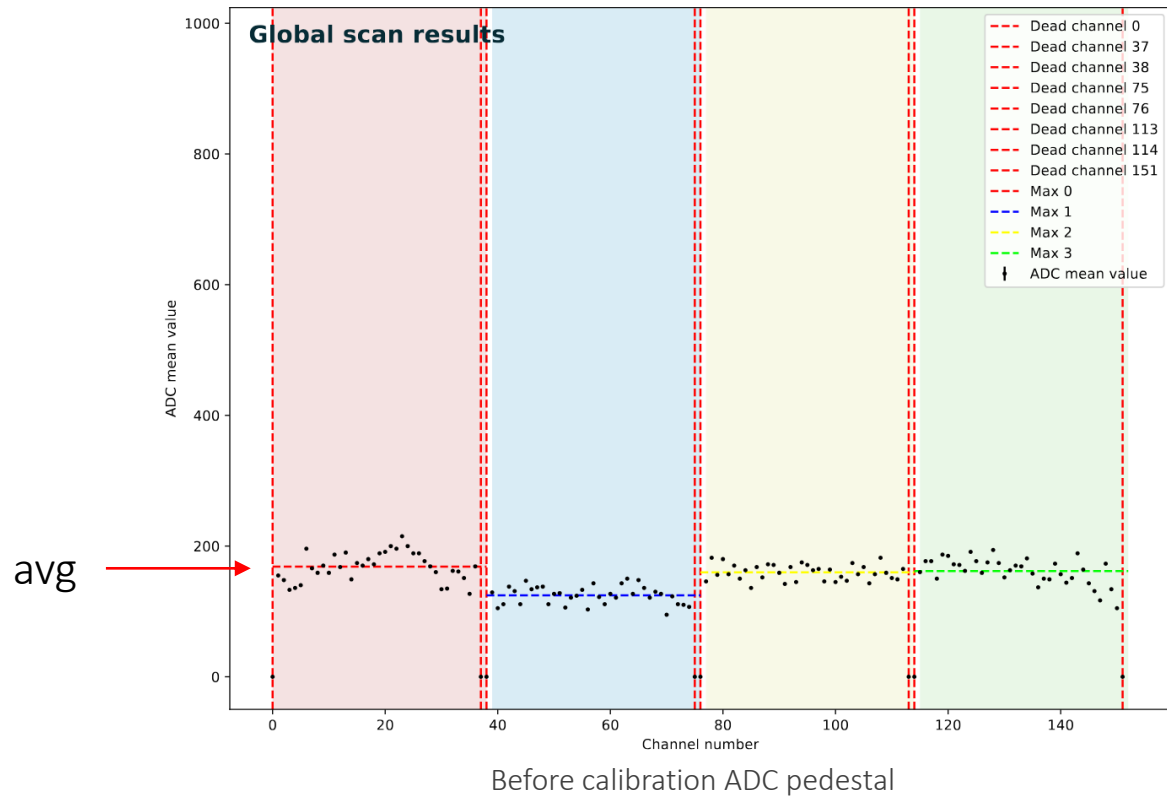
H2GCalib_3B – IO delay

- ◇ python 007_IODelay.py
- ◇ You can also use H2GDAQ's IO Delay button to do this, but it don't generate report



H2GCalib_3B – ADC pedestal calibration

- ◇ This program try to get the best pedestal value, it start from config “default_2024Aug_config.json”
- ◇ python 005_PedestalCalib.py



H2GCalib_3B – ADC pedestal calibration

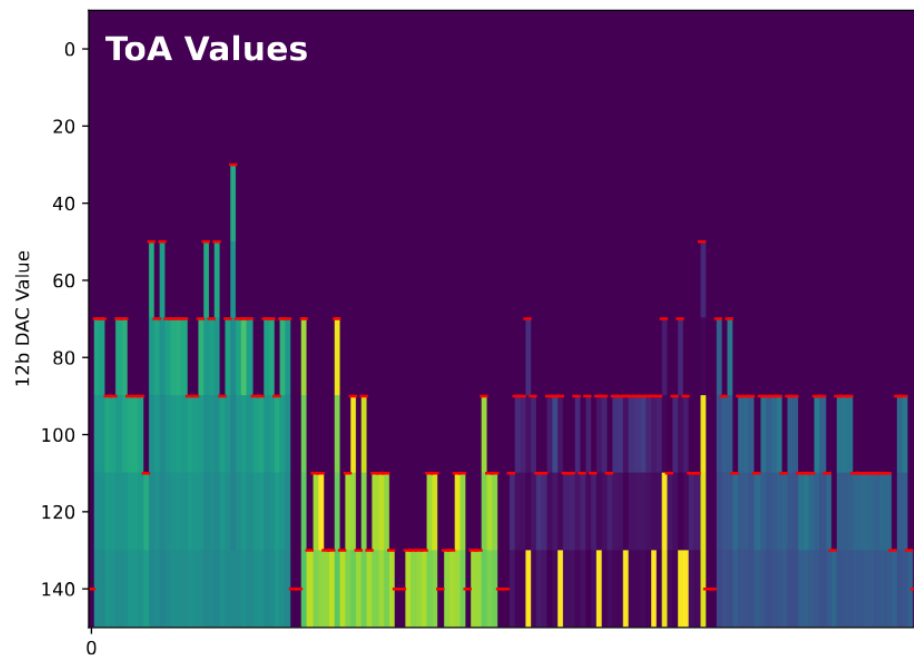
- ◇ Run calibration again but base on the previous output **but it don't work, TODO**
- ◇ `python 005_PedestalCalib.py -i python 005_PedestalCalib.py -i dump/005_PedestalCalib_data_20250625_135655/config_pede_a0_20250625_135817.json, dump/005_PedestalCalib_data_20250625_135655/config_pede_a1_20250625_135817.json`

previous ADC pedestal calibration result

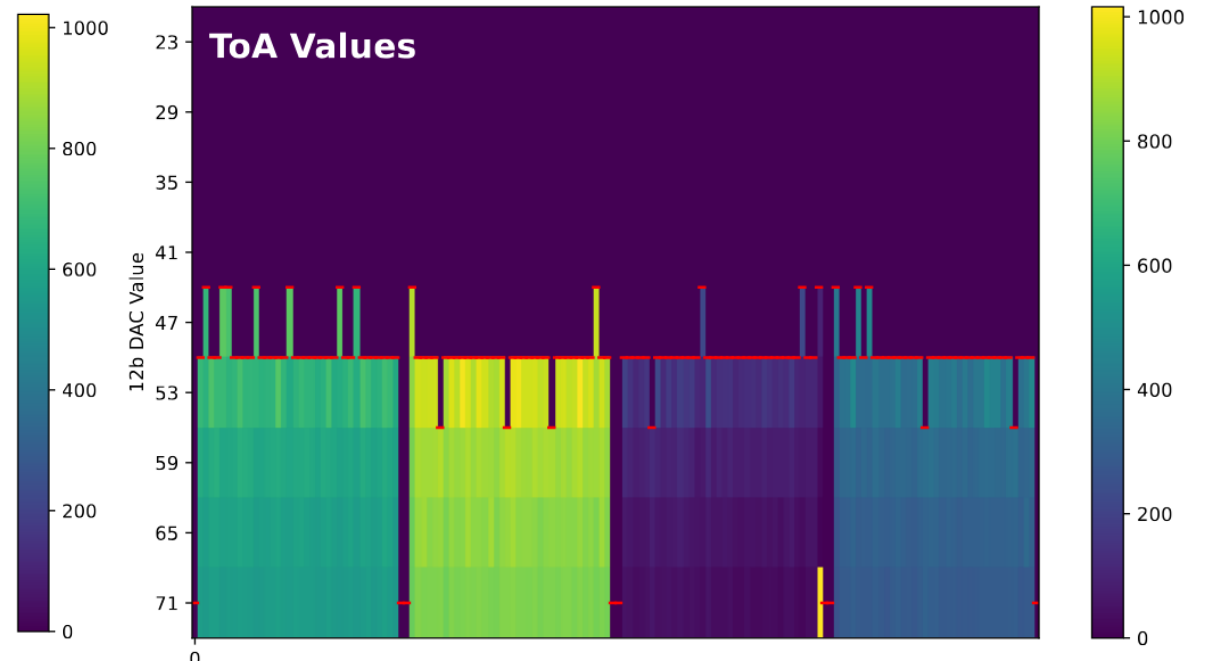
final ADC pedestal calibration result

H2GCalib_3B – ToA Calibration

◇ python 006_ToACalib.py -t 50 -i
dump/005_PedestalCalib_data_20250625_135655/config_pede_a0_20250625_135817.json,dum
p/005_PedestalCalib_data_20250625_135655/config_pede_a1_20250625_135817.json



Before calibration ToA



After calibration ToA

H2GCalib_3B – ToA Calibration

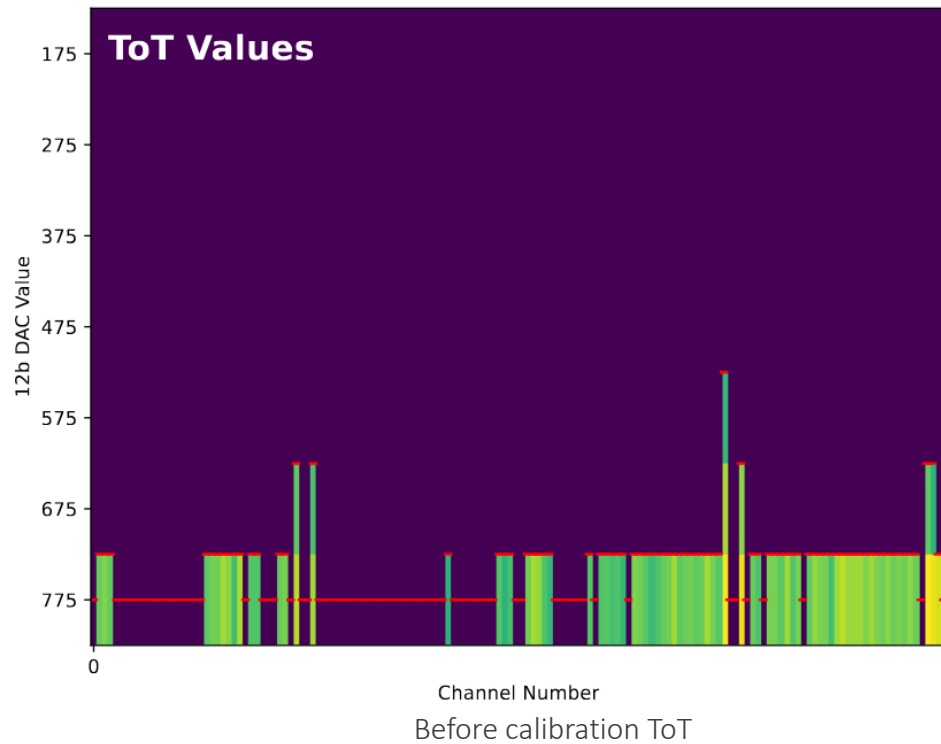
- ◇ Run calibration again but base on the previous output **TODO**
- ◇ `python 006_ToACalib.py -t 50 -i dump/006_ToACalib_data_20250625_143708/config_toa_a0_20250625_151028.json,dump/006_ToACalib_data_20250625_143708/config_toa_a1_20250625_151028.json`

previous ToA calibration result

final ToA calibration result

H2GCalib_3B – ToT Calibration

◇ python 010_ToTCalib.py -t 300 -i
dump/006_ToACalib_data_20250625_143708/config_toa_a0_20250625_151028.json,dump/006_
ToACalib_data_20250625_143708/config_toa_a1_20250625_151028.json



H2GCalib_3B – ToT Calibration

- ◇ Run calibration again but base on the previous output **TODO**
- ◇ `python 010_ToTCalib.py -t 300 -i dump/006_ToACalib_data_20250625_143708/config_toa_a0_20250625_151028.json,dump/006_ToACalib_data_20250625_143708/config_toa_a1_20250625_151028.json`

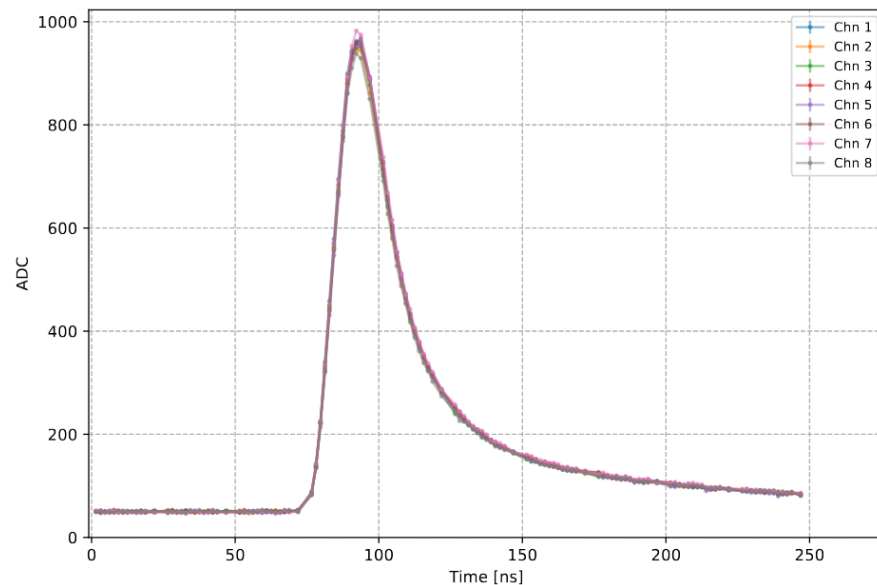
previous ToT calibration result

final ToT calibration result

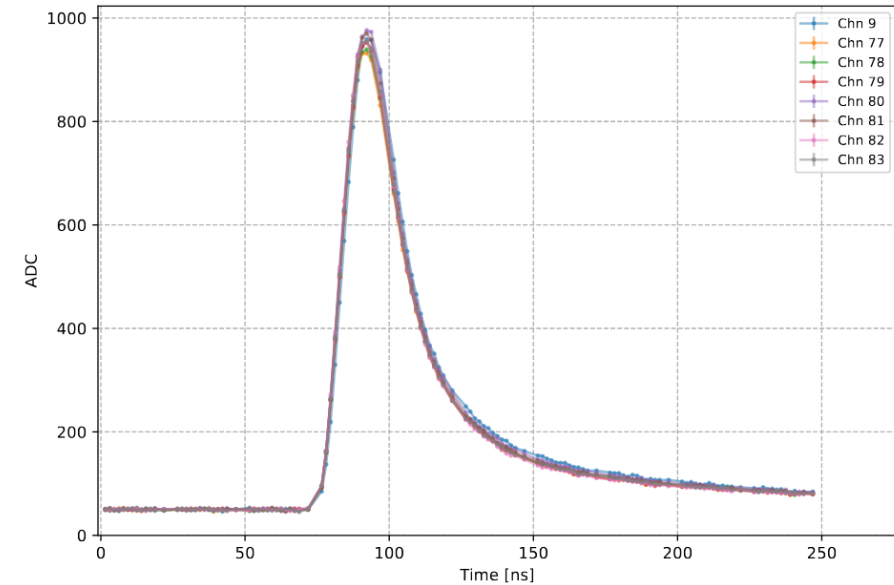
H2GCalib_3B – inject 2V5 signal

◇ modify file , line 282 `phase_settings = [0, 1, 2, 3, 5, 8, 9, 10, 11, 12, 13, 14, 15]`

◇ `python 009_IntInjection_2V5.py -i dump/010_ToTCalib_data_20250625_152005/config_tot_a0_20250625_155055.json,dump/010_ToTCalib_data_20250625_152005/config_tot_a1_20250625_155055.json`



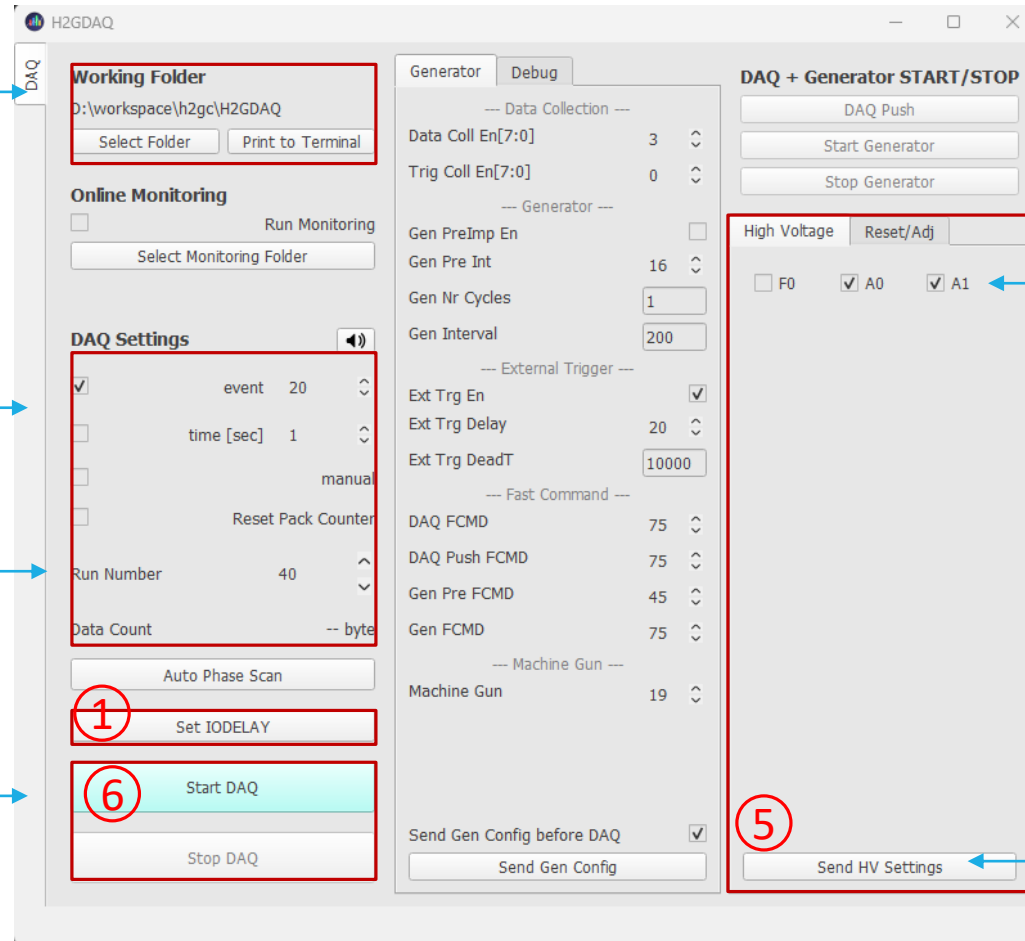
ASIC 0 (ch 1 ~ 9)



ASIC 1 is fine (ch 77 ~ 85)

Operation Procedure - DAQ

Directory to store data



DAQ stop condition: By event count, by time or manual

Output Run040.h2g

Start/Stop DAQ button

high voltage relay setting
A0 = ASIC 0 only
A1 = ASIC1 only
F0 = ASIC0 and ASIC1 both

Send command to turn On/Off the relay base on the settings above. You also need to turn on your power supply

Operation Procedure - Config

- ◇ H2GConfig
 - ◇ The GUI can read the configuration file and display it graphically. It allow you to modify register bit by bit
 - ◇ It can upload configuration to the ASIC
 - ◇ There is also a Ping FPGA button for you to test the ethernet connection

The screenshot displays the H2GConfig application window. On the left, there is an 'Initial System Dialog' with 'Number of KCUs' set to 1 and 'Number of ASICs per KCU' set to 2. The main window has a sidebar with 'Robot Mode', 'Human Mode', and 'Calibration' tabs. The 'Calibration' tab is active, showing a list of registers: Top Register (Top), Global Analog Registers (Global_Analog_1, Global_Analog_0), Reference Voltage Registers (Reference_Voltage_1, Reference_Voltage_0), Master TDC Registers (Master_TDC_1, Master_TDC_0), Digital Half Registers (Digital_Half_1, Digital_Half_0), and Channel Wise Registers (CM_2, CM_3, Channel_36, Channel_37, Channel_38, Channel_39). A central area says 'Select a register to view its content'. On the right, there are sections for 'Current Config', 'Register Selection' (with 'Use channel-wise registers' checked), 'Config File Info' (showing a file path and a 'Synced to File' status), 'Link Info' (IP Address: 10.1.2.208, Port: 11000), and a 'Readback' section containing 'Ping FPGA', 'Send Current ASIC Config', and 'Send All Configs' buttons. Red circles 2, 3, and 4 highlight the FPGA/ASIC selection, the Load button, and the Send All Configs button respectively. Blue arrows point from the text 'Load configuration' to the Load button and from 'Test network connection' to the Ping FPGA button.

2

3

4

Load configuration

Test network connection

Upload to ASIC

Operation Procedure - online_monitoring

- ◇ https://github.com/tlprotzman/h2g_online_monitoring
 - ◇ dependence: ROOT
 - ◇ developed by Tristan Protzman from Lehigh University

```
$ cat eeemcal_20sample.cfg
MAX_SAMPLES=20
NUM_FPGA=4
DETECTOR_ID=2
```

```
$ git clone https://github.com/tlprotzman/h2g_online_monitoring.git
$ cd ~/workspace/h2g_online_monitoring
$ make
$ export DATA_DIRECTORY=~/workspace/H2GDAQ
$ export MONITORING_PORT=12345
$ root -q -b -x -l RunMonitoring.c\(RunNumber)
```

```
Processing RunMonitoring.c(9)...
Parsing config file eeemcal_20sample.cfg
Key: MAX_SAMPLES, Value: 20
Key: NUM_FPGA, Value: 4
Key: DETECTOR_ID, Value: 2
```

```
Configuring for detector ID 2: EEEMCal
Configuration values:
NUM_LINES: 5
NUM_FPGA: 4
NUM_CHANNELS: 72
```

```
NUM_LINES: 5
MAX_SAMPLES: 20
MACHINE_GUN_MAX_TIME: 600
PACKET_SIZE: 1452
```

```
Configuring for detector ID 2: EEEMCal
Configuration values:
```

```
NUM_LINES: 5
NUM_FPGA: 4
NUM_CHANNELS: 72
NUM_LINES: 5
MAX_SAMPLES: 20
MACHINE_GUN_MAX_TIME: 600
PACKET_SIZE: 1452
```

```
huh??
```

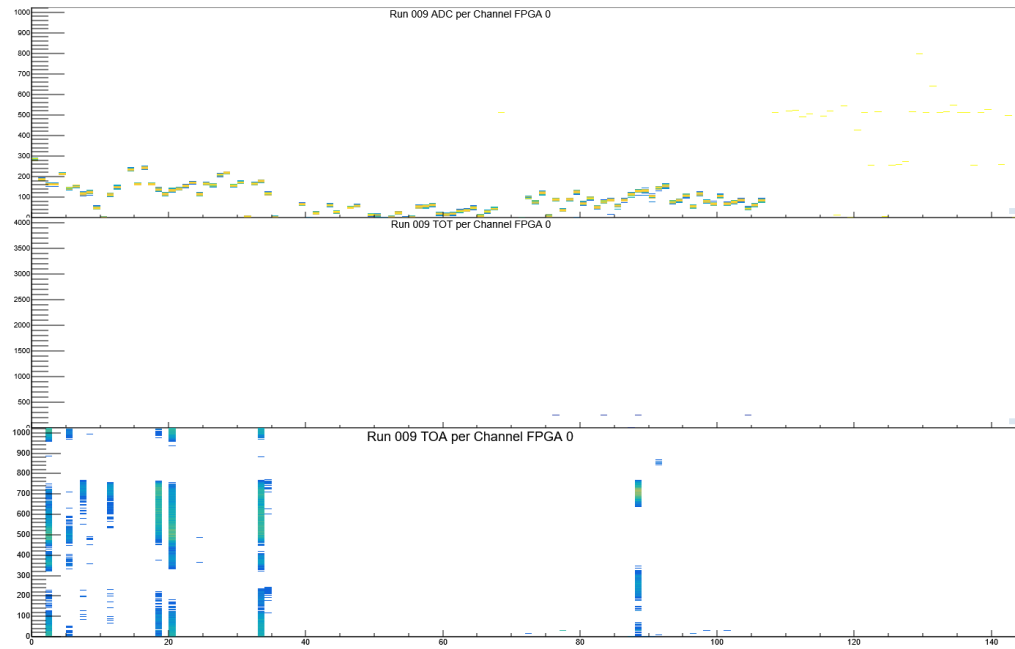
```
Info in <THttpEngine::Create>: Starting HTTP server on port 12345
Attempting to open file /home/shyao/workspace/H2GDAQ/Run009.h2g
Starting at byte 828
Building events... done!
Updating canvases... done!
```

```
.....
```

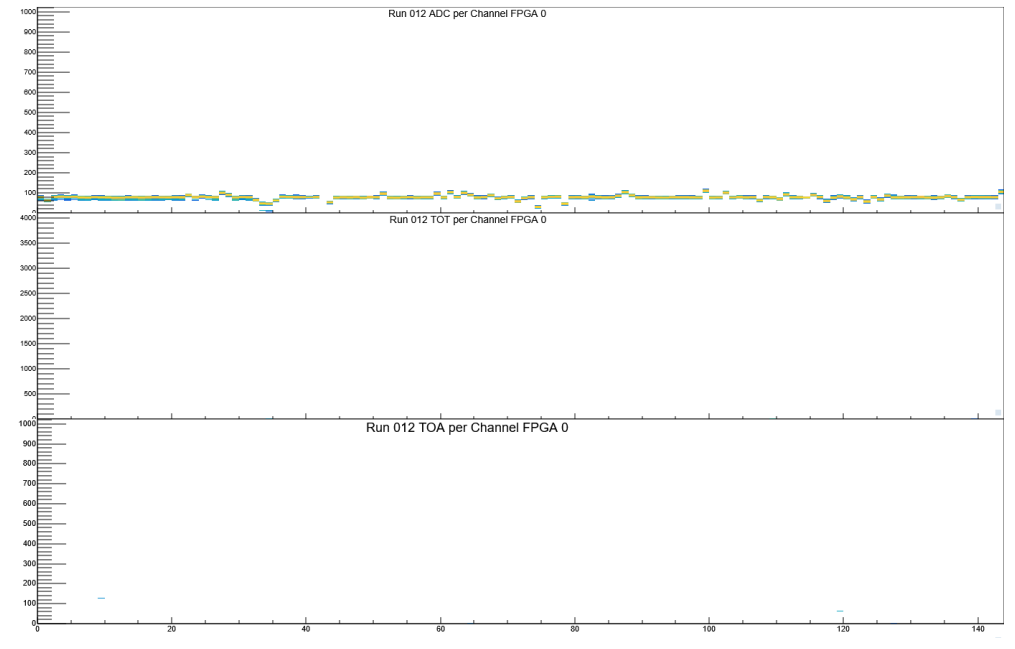
Operation Procedure - online_monitoring

- ◇ Launch web browser and connect to localhost:12345

ROOT online server
JSROOT version 7.8.1 22/01/2025
Hierarchy in json and xml format
Monitoring simple
expand all collapse all reload clear
ROOT
Spectra
QA Plots
FPGA Summaries
FPGA_0
FPGA_1
FPGA_2
FPGA_3
Waveform
DAQ Performance
EEMCal
16 Individual
Waveforms_eemcal_individ
SIPM 0
SIPM 1
SIPM 2
SIPM 3
SIPM 4
SIPM 5
SIPM 6
SIPM 7
SIPM 8
SIPM 9
SIPM 10
SIPM 11
SIPM 12
SIPM 13
SIPM 14
SIPM 15
16 Parallel
4x4 Readout
realistic_waveforms_eemca
useful_waveforms_eemcal_4
SIPM 0
SIPM 1
SIPM 2
SIPM 3
Clear_Histograms



before calibration



After calibration

Operation Procedure - online_monitoring

- ◇ Inject light to crystal, each crystal use 4x4 SiPM. The light comes from an LED with a pulse width of 100ns

ROOT online server

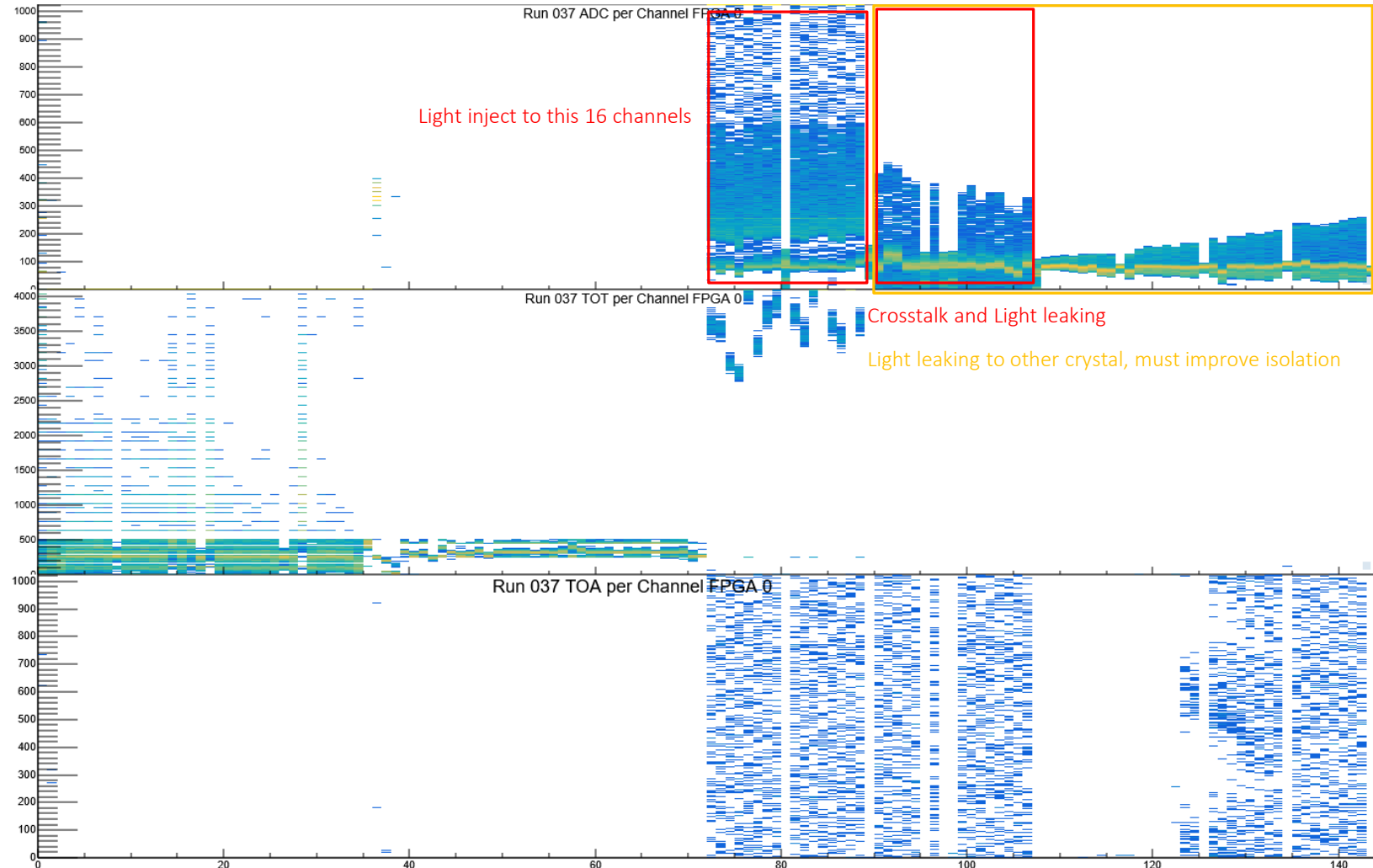
JSROOT version 7.8.1 22/01/2025

Hierarchy in [json](#) and [xml](#) format

Monitoring **simple**

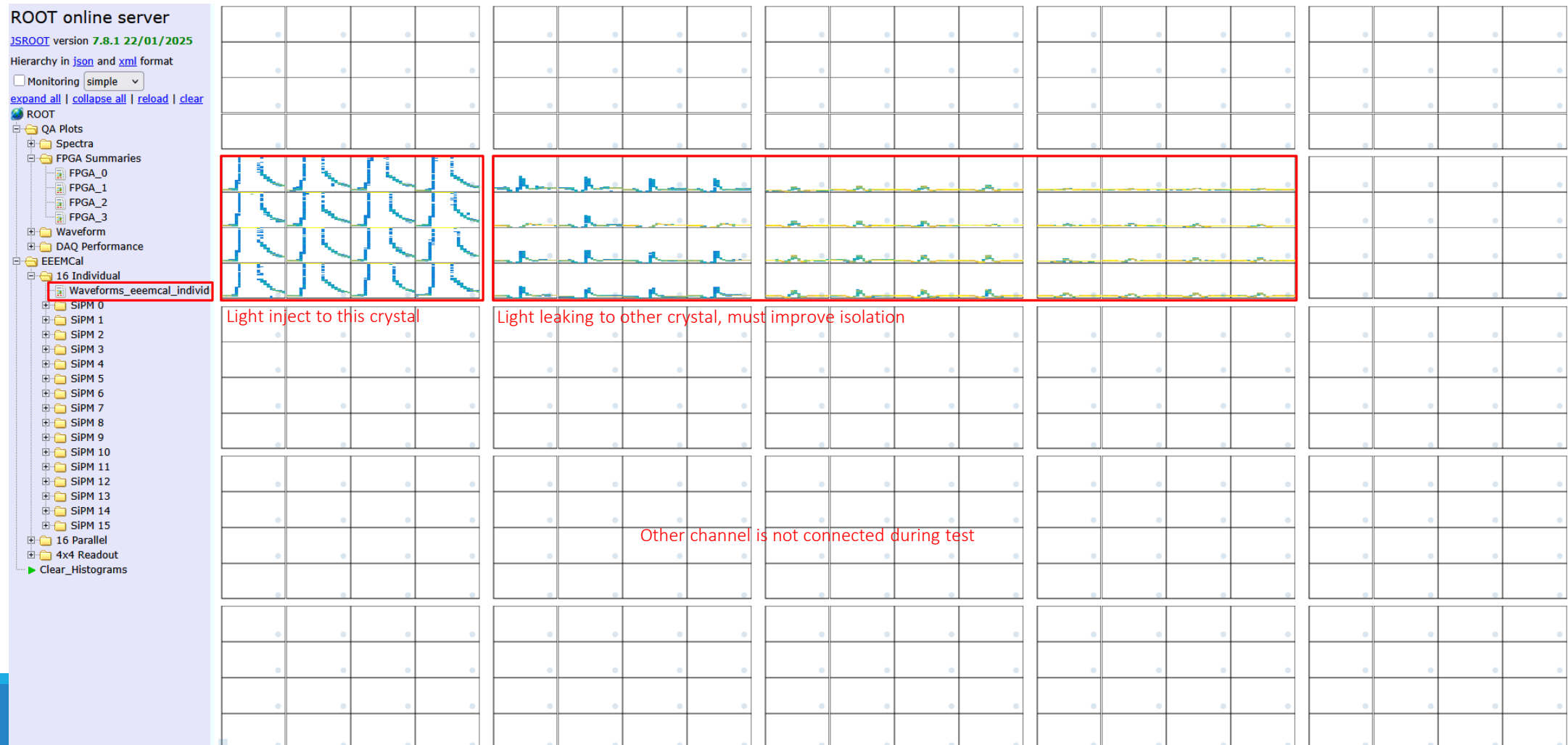
[expand all](#) | [collapse all](#) | [reload](#) | [clear](#)

- ROOT
 - QA Plots
 - Spectra
 - FPGA Summaries
 - FPGA_0**
 - FPGA_1
 - FPGA_2
 - FPGA_3
 - Waveform
 - DAQ Performance
 - EEEMCa
 - 16 Individual
 - Waveforms_eemcal_individ
 - SIPM 0
 - SIPM 1
 - SIPM 2
 - SIPM 3
 - SIPM 4
 - SIPM 5
 - SIPM 6
 - SIPM 7
 - SIPM 8
 - SIPM 9
 - SIPM 10
 - SIPM 11
 - SIPM 12
 - SIPM 13
 - SIPM 14
 - SIPM 15
 - 16 Parallel
 - 4x4 Readout
 - Clear_Histograms



Operation Procedure - online_monitoring

◇ Draw in crystal Array, it depend on your channel mapping

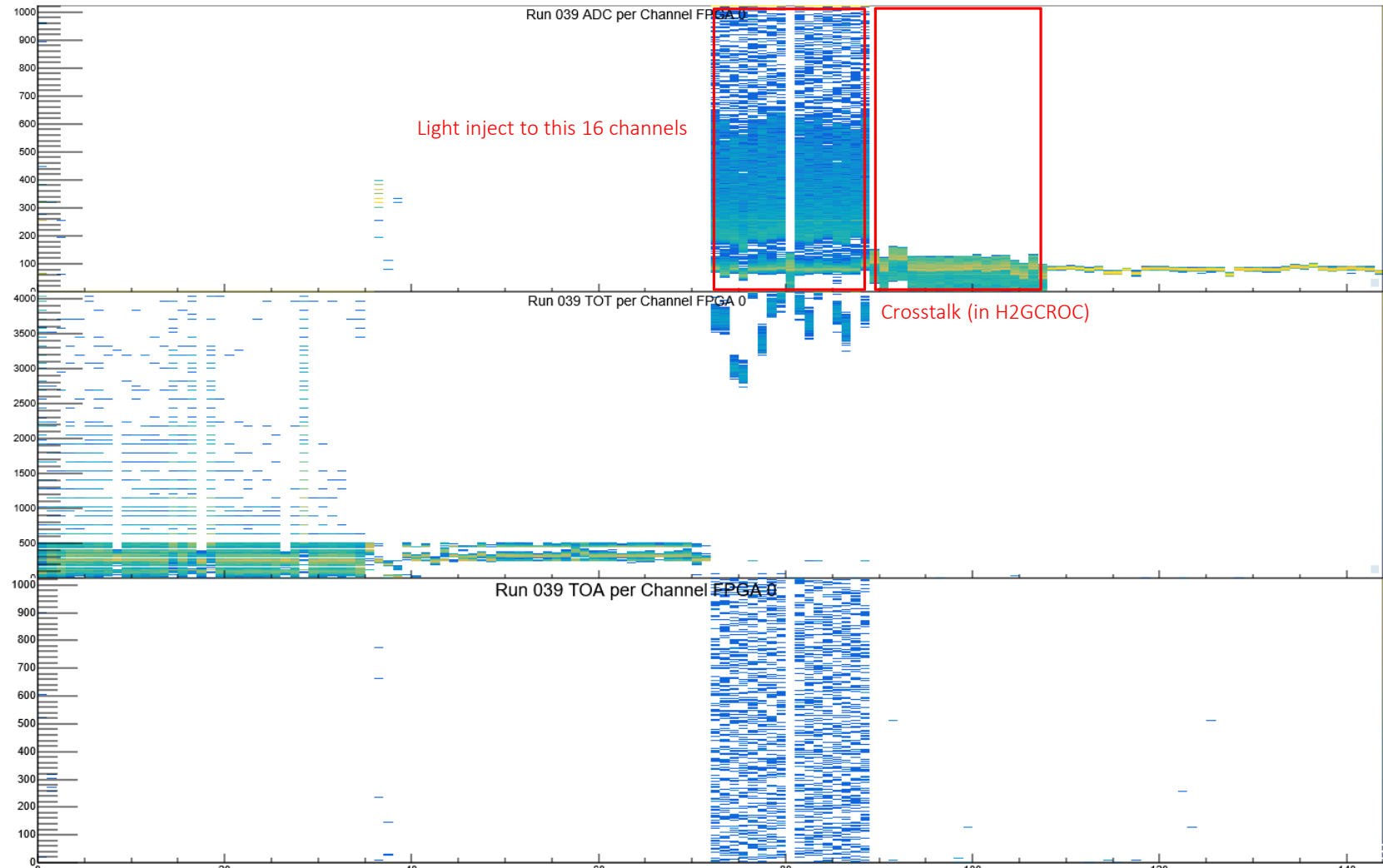


Operation Procedure - online_monitoring

◇ Inject light to crystal, without leaking

ROOT online server
JSROOT version 7.8.1 22/01/2025
Hierarchy in [json](#) and [xml](#) format
 Monitoring [simple](#) [expand all](#) | [collapse all](#) | [reload](#) | [clear](#)

- ROOT
 - QA Plots
 - Spectra
 - FPGA Summaries**
 - FPGA_0
 - FPGA_1
 - FPGA_2
 - FPGA_3
 - Waveform
 - DAQ Performance
 - EEEMCal
 - 16 Individual
 - Waveforms_eeemcal_individ
 - SIPM 0
 - SIPM 1
 - SIPM 2
 - SIPM 3
 - SIPM 4
 - SIPM 5
 - SIPM 6
 - SIPM 7
 - SIPM 8
 - SIPM 9
 - SIPM 10
 - SIPM 11
 - SIPM 12
 - SIPM 13
 - SIPM 14
 - SIPM 15
 - 16 Parallel
 - 4x4 Readout
 - Clear_Histograms



Operation Procedure - decode

- ◇ https://github.com/tlprotzman/h2g_decode
 - ◇ dependent ROOT , developed by Tristan Protzman from Lehigh University
 - ◇ It provide a library h2g_decode and a executable h2g_run

```
$ cd ~/workspace
$ git clone https://github.com/tlprotzman/h2g_decode.git
$ cd ~/workspace/h2g_decode
$ mkdir data build; cd build; cmake ..; make
$ export DATA_DIRECTORY=~/workspace/H2GDAQ
$ export OUTPUT_DIRECTORY=~/workspace/h2g_decode
```

```
$ ./build/h2g_run
```

```
Usage: h2g_decode -r <run_number> [-d <detector_id>] [-n <num_kcu>] [-g] [-G LEVEL] [-T]
```

```
-r, --run      Run number (required)
```

```
-d, --detector Detector ID (default: 0, LFHCAL: 1, EEEMCAL: 2)
```

```
-n, --num-kcu  Number of KCUs (default: 4)
```

```
-g, --debug    Enable debug output with INFO level
```

```
-G, --debug-level Set debug level explicitly:
```

```
0: OFF, 1: ERROR, 2: WARNING, 3: INFO, 4: DEBUG, 5: TRACE
```

```
-T, --truncate Enable ADC truncation
```

```
-h, --help     Show this help message
```

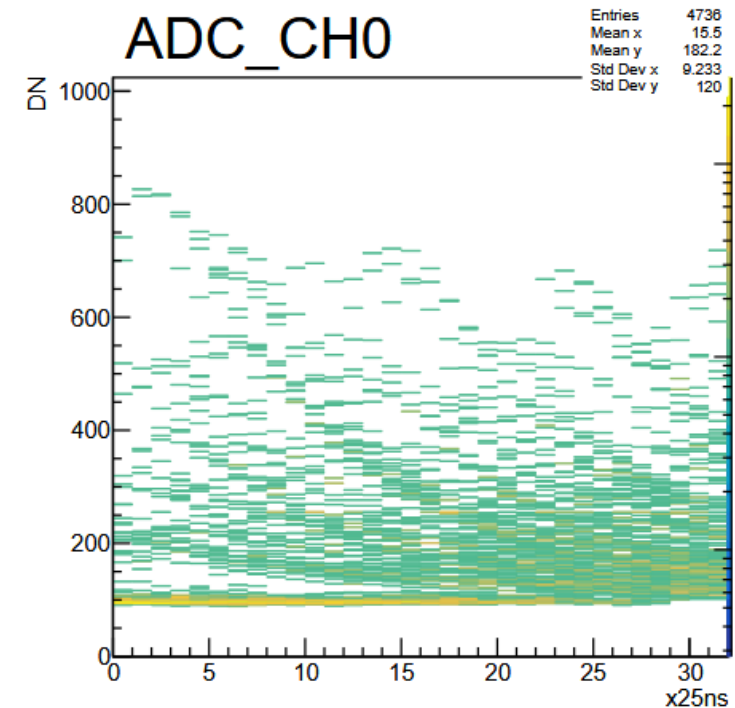
** Data file is named with "Run%03d.h2g"

Operation Procedure - decode

- ◇ Convert rawdata to root file for future analysis.

```
$ ./build/h2g_run -r 14 -d 0 -n 1
$ root
root [1] TFile f("Run014.root")
root [2] events->Print()
*****
*Tree   :events   : Events
*Entries :    98258 : Total =    7360953537 bytes File Size = 492028194 *
*       :         : Tree compression factor = 14.96
*****
*Br    0 :event_number : event_number/i
*Entries :    98258 : Total Size=    395183 bytes File Size =   138994 *
*Baskets :     18 : Basket Size=    51200 bytes Compression=   2.84 *
*.....*
*Br    1 :timestamps : timestamps[1]/i
*Entries :    98258 : Total Size=    395145 bytes File Size =   114179 *
*Baskets :     18 : Basket Size=    51200 bytes Compression=   3.45 *
*.....*
*Br    2 :num_samples : num_samples/i
*Entries :    98258 : Total Size=    395161 bytes File Size =     4019 *
*Baskets :     18 : Basket Size=    51200 bytes Compression=  98.15 *
*.....*
*Br    3 :adc         : adc[144][32]/i
*Entries :    98258 : Total Size= 1811620311 bytes File Size = 408563585 *
*Baskets :    5745 : Basket Size=   7088128 bytes Compression=   4.43 *
*.....*
*Br    4 :toa         : toa[144][32]/i
*Entries :    98258 : Total Size= 1811620311 bytes File Size =  22647543 *
*Baskets :    5745 : Basket Size=   7088128 bytes Compression=  79.99 *
*.....*
*Br    5 :tot         : tot[144][32]/i
*Entries :    98258 : Total Size= 1811620311 bytes File Size =   8514786 *
*Baskets :    5745 : Basket Size=   7088128 bytes Compression= 212.75 *
*.....*
*Br    6 :hamming    : hamming[144][32]/i
*Entries :    98258 : Total Size= 1811643307 bytes File Size =   8529147 *
*Baskets :    5745 : Basket Size=   7089664 bytes Compression= 212.39 *
*.....*
*Br    7 :hit_max    : hit_max[144]/i
*Entries :    98258 : Total Size=   56630819 bytes File Size =  21343750 *
*Baskets :     353 : Basket Size=    221184 bytes Compression=   2.65 *
*.....*
*Br    8 :hit_pedestal : hit_pedestal[144]/i
*Entries :    98258 : Total Size=   56632604 bytes File Size =  22016214 *
*Baskets :     353 : Basket Size=    221696 bytes Compression=   2.57 *
*.....*
* 2026/4/1
```

ADC waveform
x-axis: time
y-axis: amplitude



Q/A

◇ WWW

Operation Procedure - online_monitoring

◇ To modify channel mapping: online_monitor.cxx

```
// EEEMCal mapping - instead of "layers", we have a single plane, where each crystal
is one connector
// FPGA IP | ID
// 208     | 0
// 209     | 1
// 210     | 2
// 211     | 3
int eeemcal_fpga_map[25] = {0, 0, 0, 0, 2,
                          0, 0, 0, 0, 2,
                          1, 1, 1, 1, 2,
                          1, 1, 1, 1, 2,
                          2, 2, 2, 2, 3};

// ASIC | ID
// 0    | 0
// 1    | 1
int eeemcal_asic_map[25] = { 0, 0, 0, 0, 1,
                          1, 1, 1, 1, 1,
                          0, 0, 0, 0, 1,
                          1, 1, 1, 1, 1,
                          0, 0, 0, 0, 0};

// Connector | ID
// A         | 0
// B         | 1
// C         | 2
// D         | 3
int eeemcal_connector_map[25] = { 0, 1, 2, 3, 0,
                                0, 1, 2, 3, 1,
                                0, 1, 2, 3, 2,
                                0, 1, 2, 3, 3,
                                0, 1, 2, 3, 0};

int eeemcal_16i_channel_a_map[16] = { 0, 1, 2, 3, 4, 5, 6, 7,
                                       9, 10, 11, 12, 13, 14, 15, 16};

int eeemcal_16i_channel_b_map[16] = {19, 20, 21, 22, 23, 24, 25, 26,
                                       27, 28, 29, 30, 31, 32, 33, 34};

int eeemcal_16i_channel_c_map[16] = {55, 56, 57, 58, 59, 60, 61, 62,
                                       63, 64, 65, 66, 67, 68, 69, 70};

int eeemcal_16i_channel_d_map[16] = {36, 37, 38, 39, 40, 41, 42, 43,
                                       45, 46, 47, 48, 49, 50, 51, 52};

/*
int eeemcal_16i_channel_a_map[16] = { 2, 6, 11, 15, 0, 4, 9, 13,
                                       1, 5, 10, 14, 3, 7, 12, 16};

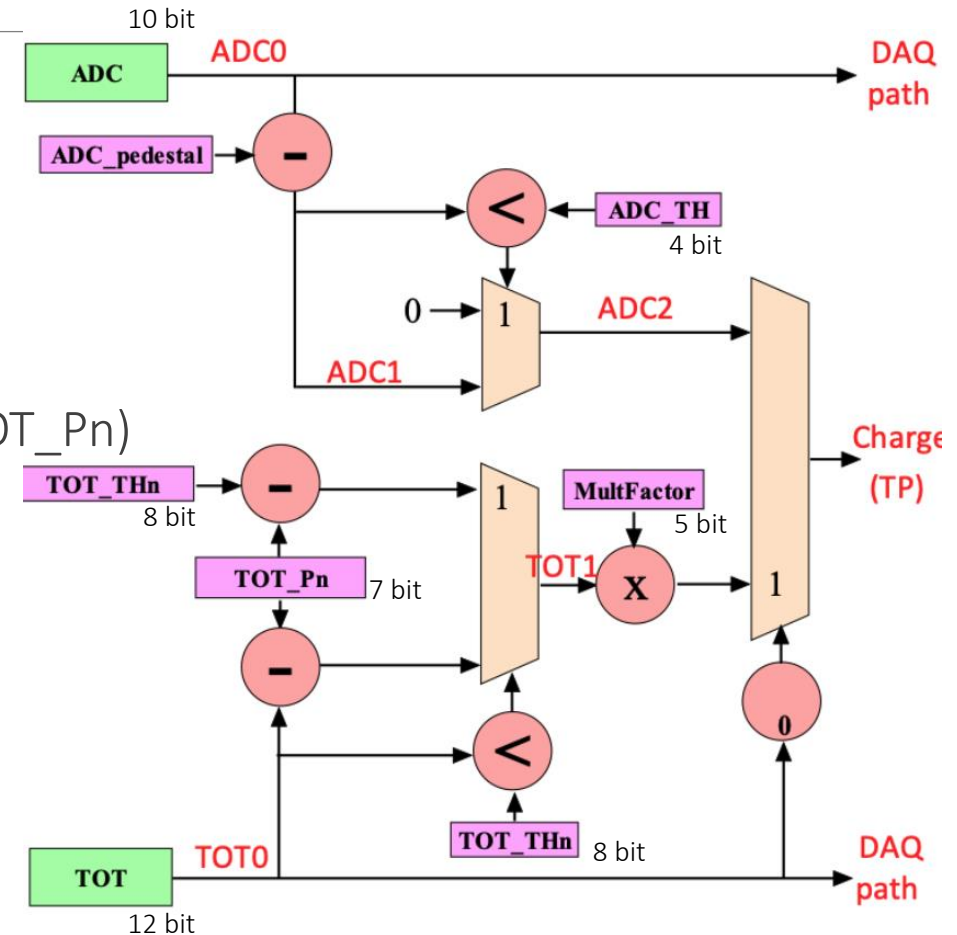
int eeemcal_16i_channel_b_map[16] = {20, 24, 29, 33, 18, 22, 27, 31,
                                       19, 23, 28, 32, 21, 25, 30, 34};

int eeemcal_16i_channel_c_map[16] = {67, 63, 59, 55, 69, 65, 61, 57,
                                       70, 66, 60, 56, 68, 64, 58, 54};

int eeemcal_16i_channel_d_map[16] = {50, 46, 40, 36, 52, 48, 42, 38,
                                       51, 47, 43, 39, 49, 45, 41, 37};
*/
```

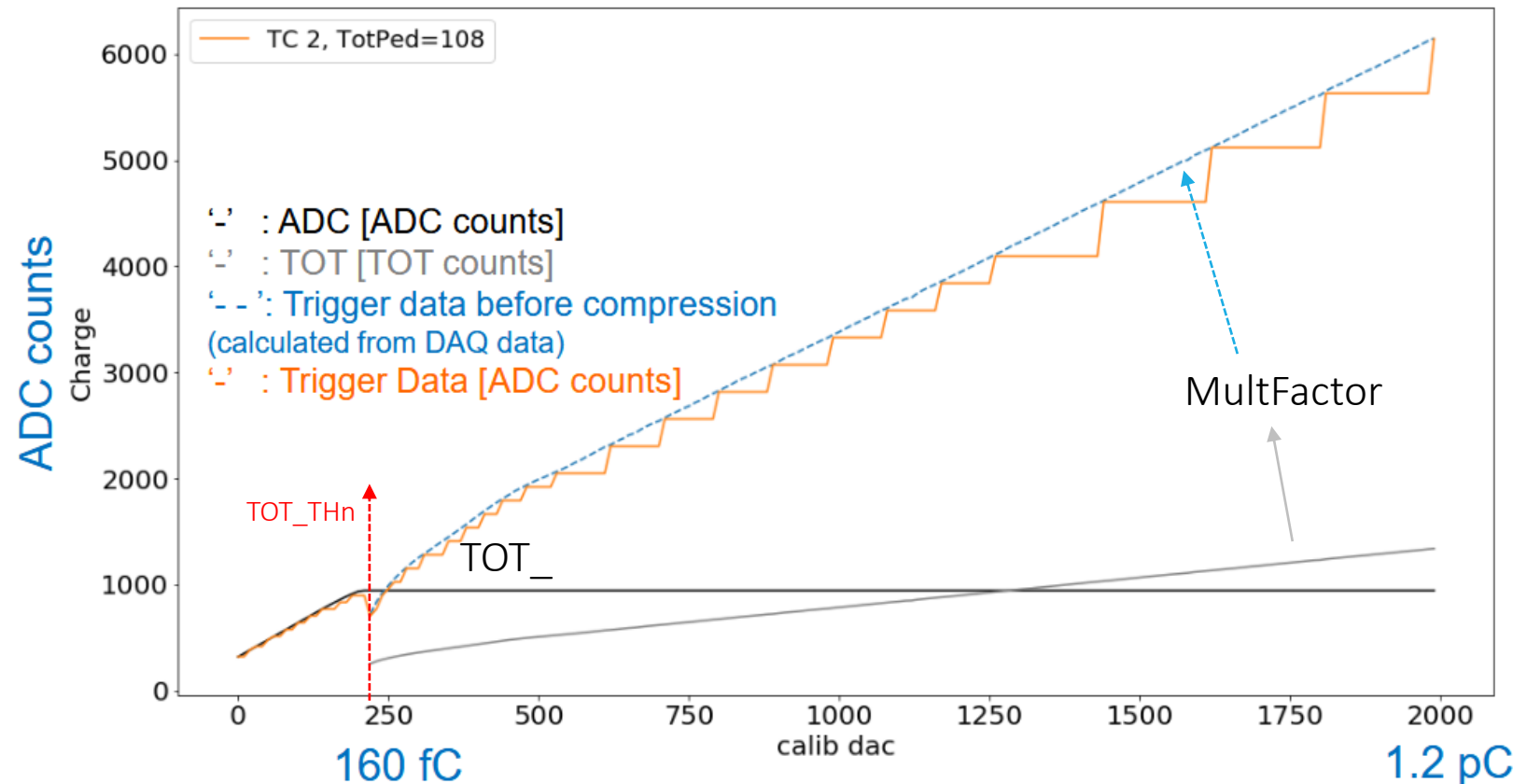
Trigger path

- ◇ $ADC1 = ADC0 - ADC_pedestal$
- ◇ $ADC2 = (ADC1 < ADC_TH) ? 0 : ADC1$
 - ◇ ***if set $ADC_pedestal = ADC_TH = 0$
 - ◇ *** $ADC2 = ADC0$
- ◇ $TOT1 = (TOT0 < TOT_THn) ? (TOT_THn - TOT_Pn) : (TOT0 - TOT_Pn)$
 - ◇ *** if set $TOT_THn = TOT_Pn = 0$
 - ◇ *** $TOT1 = TOT0$
- ◇ $Charge = (TOT0 \neq 0) ? TOT1 * MultFactor : ADC2$



Trigger path

- ◇ ADC [ADC counts] is 10 bit
- ◇ TOT [TOT counts] is 12 bit
- ◇ linearization

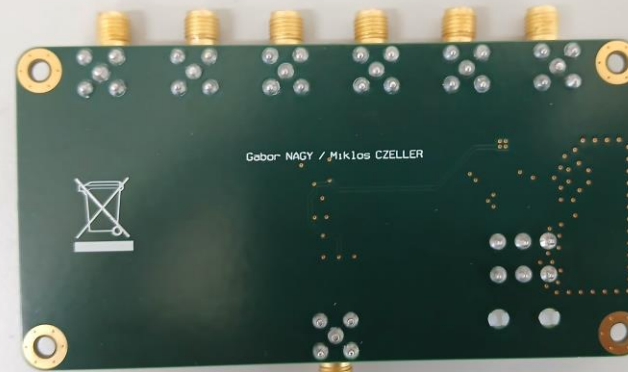


Local R&D

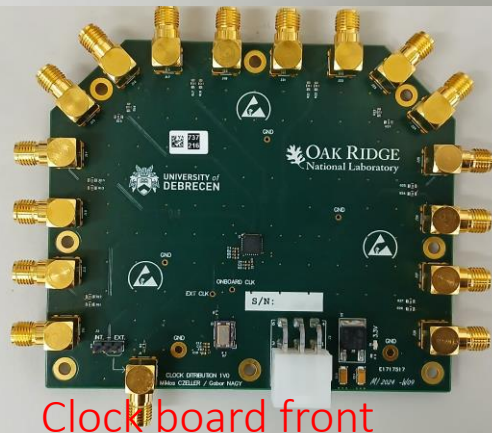
◇ WWW



Trigger board front



Trigger board back



Clock board front

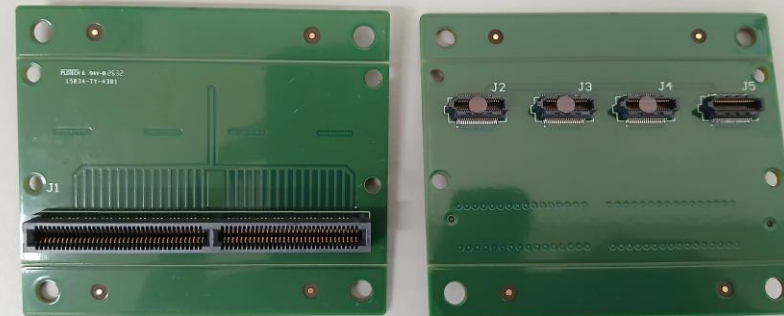


Clock board back



Clock board power

Trigger board power



Adapter board

Hardware @ NCU

◇ Device

- ◇ PbWO/LYSO Crystal 1/1
- ◇ SiPM board 1/1
- ◇ Adapter board 2/4
- ◇ protoboard 3/4
- ◇ KCU105 1/2
- ◇ Clock board 1/1
- ◇ Trigger board 1/1

◇ Equip

- ◇ PC 1/1
- ◇ Power supply (for Clock distribution board and Trigger board) 1/1
- ◇ Signal generator 1/1
- ◇ Network Switch 1GbE 1/1

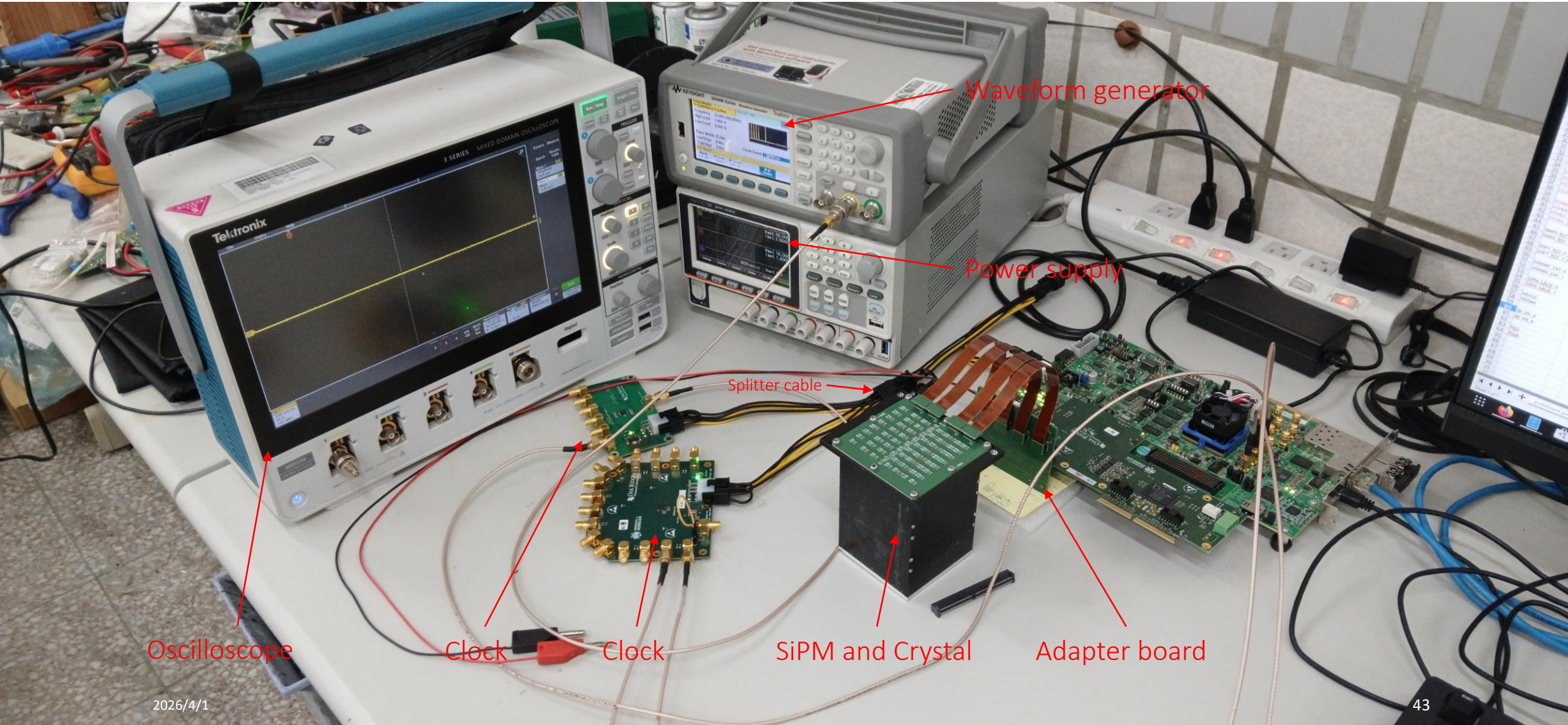
◇ Cable

- ◇ SMA male to male cable 3/8
- ◇ SMA to BNC connector 2/6
- ◇ SMA to <unknown> cable for clock board connect to <unknown input> 0/1
- ◇ SMA to <unknown> cable for trigger board connect to signal generator 0/1
- ◇ Power cable for Clock distribution board 1/1
- ◇ Power cable for Trigger board 1/1
- ◇ SiPM board to Adapter board ?/? not sure
- ◇ Network Cable 1GbE 1/2

Assume 2 KCU105 and 4 protoboard setup



Current environment



About 12V Power

- ◇ Shared FPGA evaluation board power supply via splitter

KCU 105



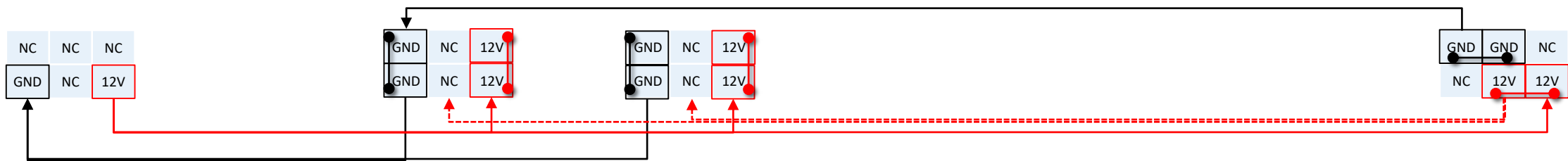
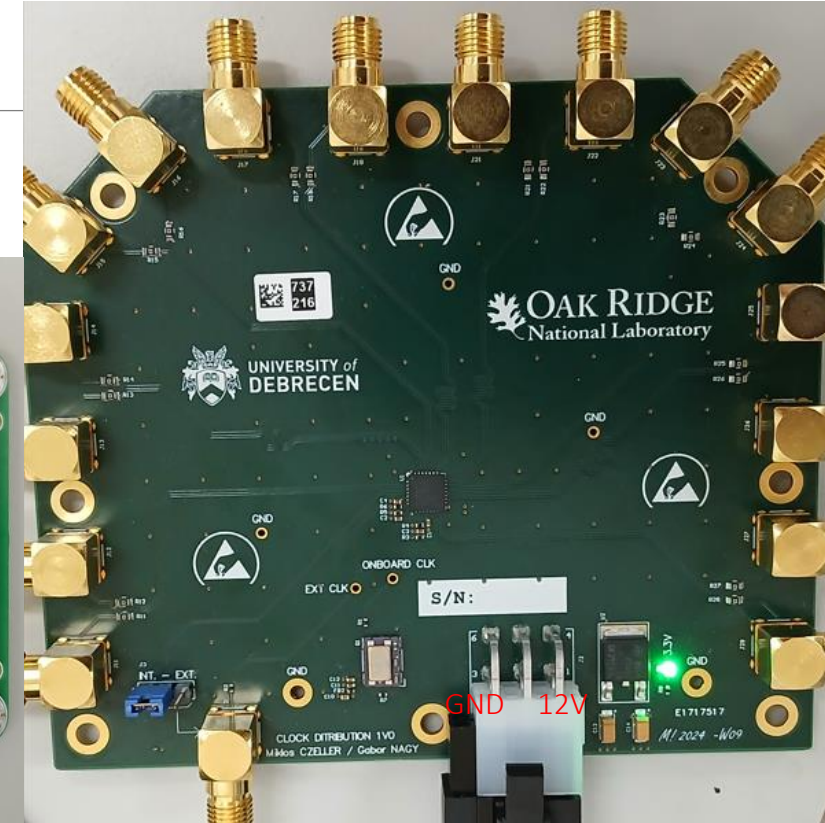
KCU105 Power supply



Trigger board



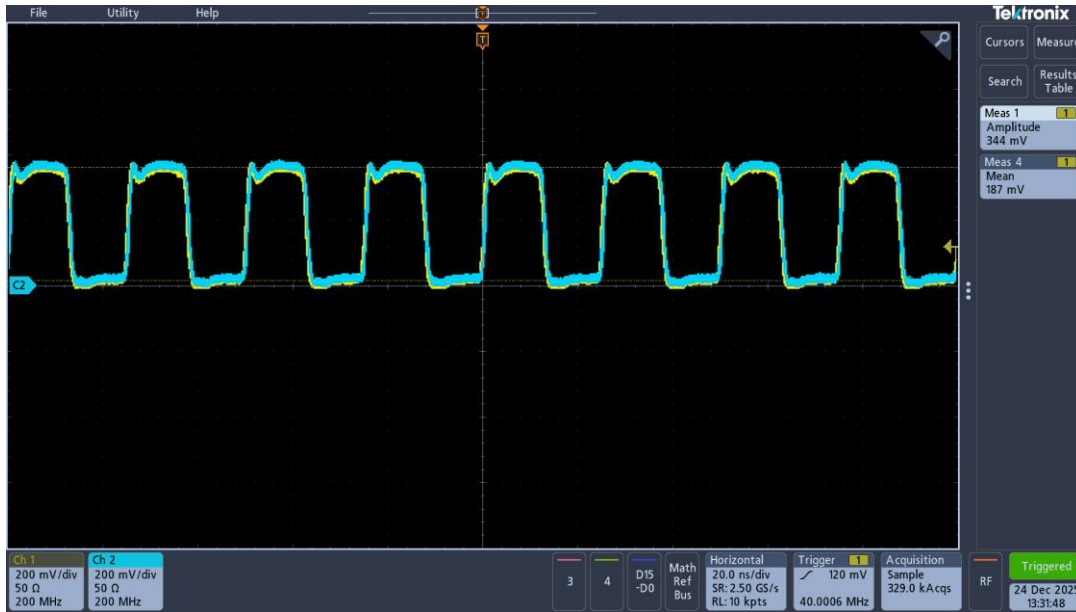
Clock board



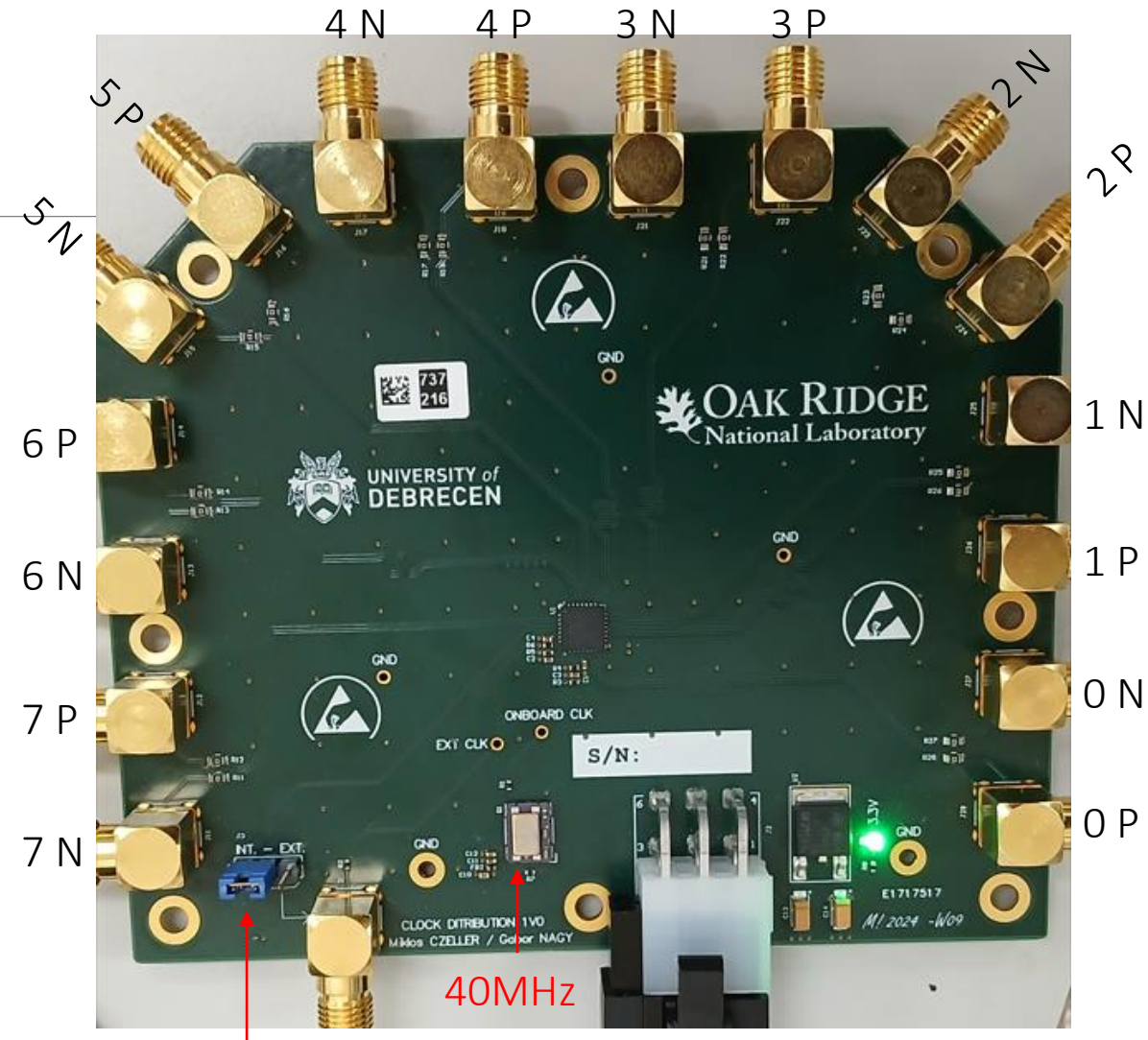
GND	GND	12V
GND	12V	12V

← Connect all them together, your cable will be

Clock board



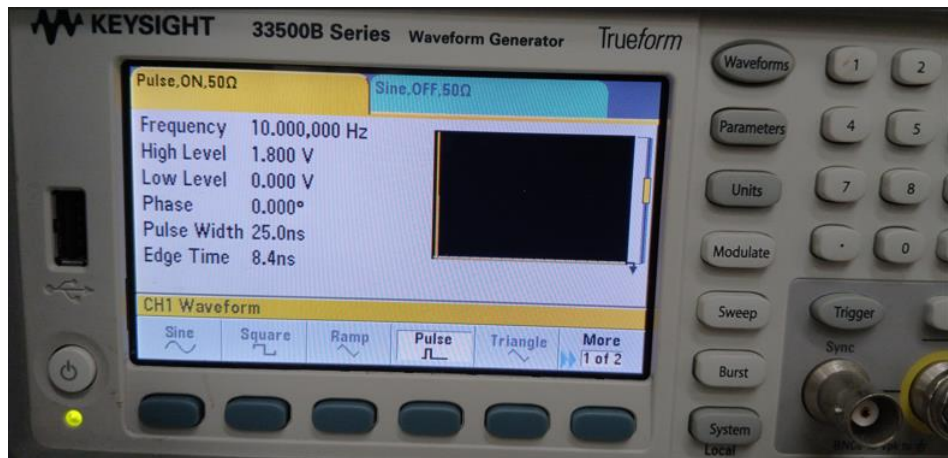
The output of 7P and 6P



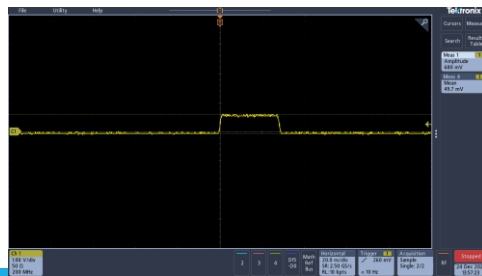
Jumper select:
Int: On board 40MHz 0.5ppm clock
Ext: External input

external trigger signal

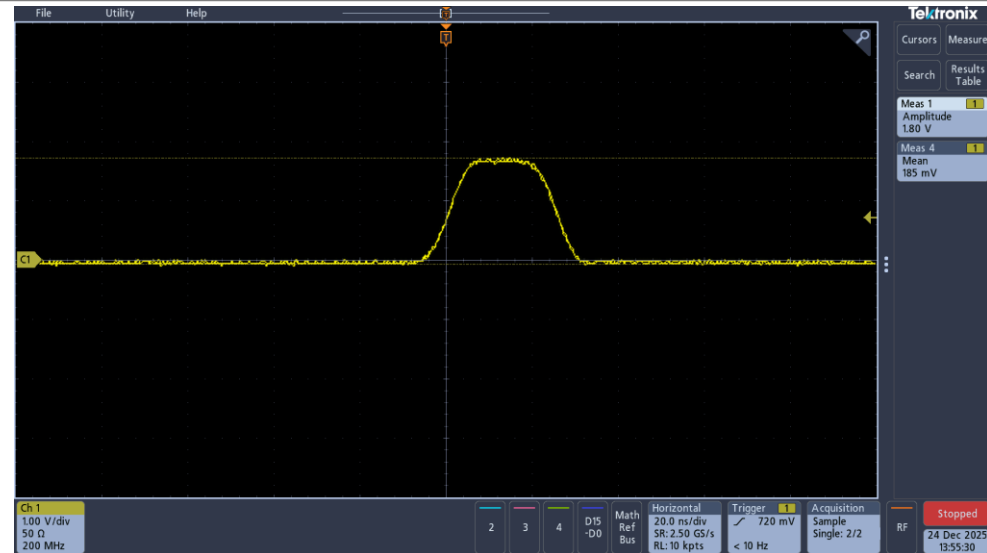
- ◇ 1.8V output, 10Hz, Pulse width 25ns
- ◇ Send to Trigger input



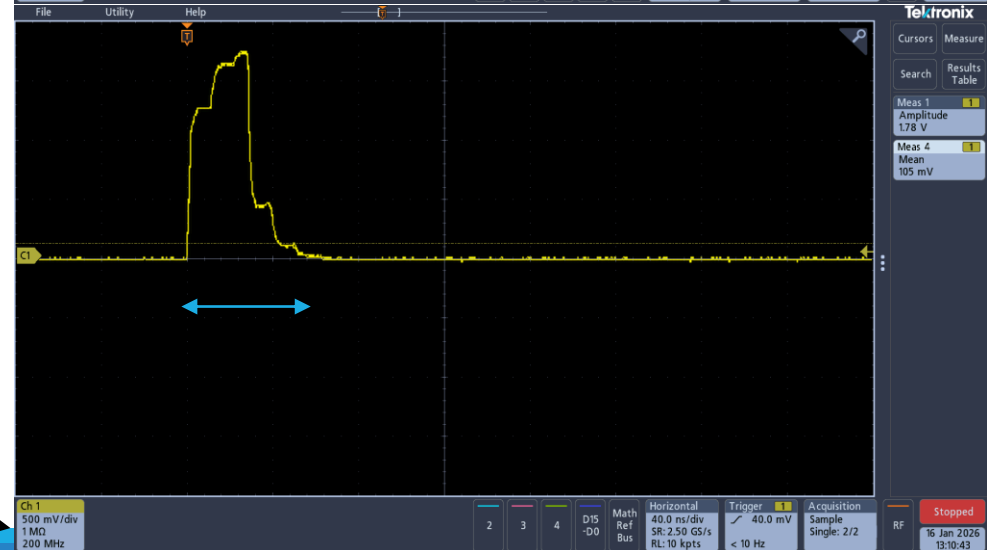
If the input signal voltage decreases, the output width will decrease. Smaller than 0.8V output disappears



1MΩ



Input



Output

UDP packet format (FPGA -> PC)

◇ a UDP packet

1452 or 8972 Bytes

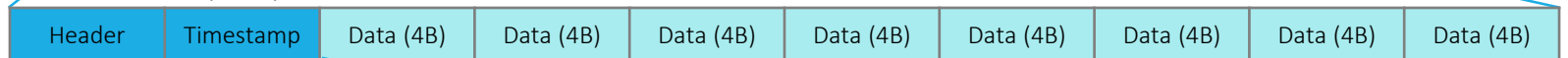
Normal frame N = 0 ~ 36
JUMBO frame N = 0 ~ 224



UDP header:

2B	Byte Counter	(0 ~ 36) * 40
2B	Packet Counter	0 ~ 65535
8B	String ("0")	

Data (40B):



Data format header:

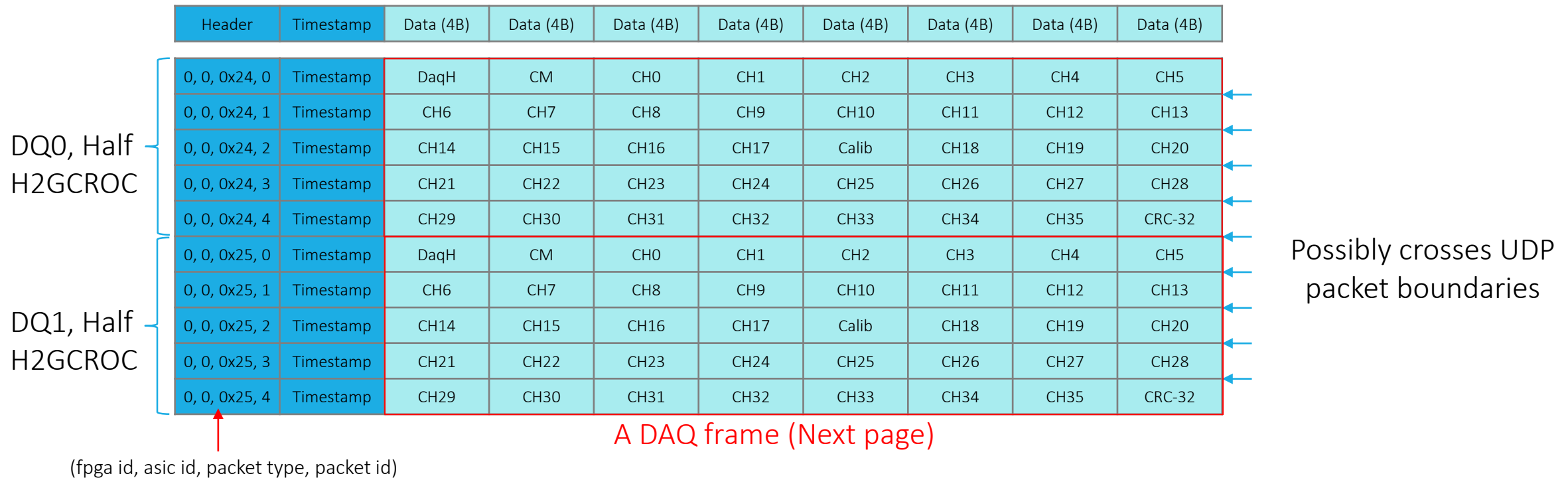
1B	Header	b0-3: Asic address. b4-7: 0xA0
1B	Fpga board address	0x00 ~ 0x07
1B	packet type	TR0: 0x20, TR1: 0x21, TR2: 0x22, TR3: 0x23, DQ0: 0x24, DQ1: 0x25
1B	packet id	0 ~ 4

Timestamp:

32B	Timestamp	Guess: Unit 25ns, for time sync between FPGAs
-----	-----------	---

UDP packet format (FPGA -> PC)

- ◇ One read command generate ten of 40B packet (400Bytes), Two ASIC is 800 Bytes
- ◇ Timestamp field is duplicated



DAQ frame (H2GCROC->FPGA)

◇ A read command, send a DAQ frame (From datasheet)

◇ $40 * 32\text{bit} = 1280\text{ bits}$

41 x 32bits



DaqH:

4b	hd	0b1111
12b	BxCounter	Value of the Bunch Crossing Counter (BCD) on 12 bits. (40MHz). Range: 1 - 3564. Reset value: 1
6b	Event	Value of the Event Counter (EC) , to detect if a L1 trigger has been sent but related data wasn't saved (RAM2 full). Range: 0 - 63. Reset value: 1
3b	Orbit	Value of the Orbit Counter (OC). Range: 0 - 7. Reset value: 0. Increase 1 when BCD wrapping
1b	H1	Error during hamming decoding in Header
1b	H2	Error during hamming decoding in channels from CM to CH17 (1st Quarter)
1b	H3	Error during hamming decoding in channels from Calib to CH35 (2nd Quarter)
4b	tr	0b0101: no first event flag 0b0010: first event flag

CM:

10b	CM1 ADC
10b	CM0 ADC
12b	0b000

~ 89us BxCounter wrapping

~ 712us OC wrapping

CH#, Calib:

	CASE1, CASE2	CASE3	CASE4
10b	TOA	TOA	TOA
10b	ADC	TOT	TOT
10b	ADC-1	ADC-1	ADC
1b	Tp	Tp	Tp
1b	0	1	Tc

CASE1: Charge $Q < \text{TOA Threshold}$, $\text{TOA} = 0$

CASE2: Charge $Q < \text{TOT Threshold}$, $\text{TOA} > 0$

CASE3: Charge $Q > \text{TOT Threshold}$

CASE4: Debug mode

Tp: TOT-In-Progress

Tc: TOT Complete

-1: Previous BX ?

Decode

```

UDP_TX_Packet_Counter: 57 ByteCounter: 1440
hd: 15 H1: 0 H2: 0 H3: 0 tr: 5 BxCounter: 2827 Event: 13 Orbit: 0
FPGA: 0 ASIC: 0 packet_type: 25 packet_id: 00 timestamp: 0ad3daca Data: fb0b3405 00017000 06800000 09600000 08b00000 06600000 07800000 07800000
FPGA: 0 ASIC: 0 packet_type: 25 packet_id: 01 timestamp: 0ad3daca Data: 06500000 07100000 06200000 06800000 07900000 05f00000 06200000 07900000
FPGA: 0 ASIC: 0 packet_type: 25 packet_id: 02 timestamp: 0ad3daca Data: 08000000 05a00000 04600000 07a00000 05800000 06f00000 07a00000 07b00000
FPGA: 0 ASIC: 0 packet_type: 25 packet_id: 03 timestamp: 0ad3daca Data: 09e00000 06e00000 06700000 09200000 06f00000 05500000 09300000 07200000
FPGA: 0 ASIC: 0 packet_type: 25 packet_id: 04 timestamp: 0ad3daca Data: 07000000 08500000 05b00000 07d00000 06e00000 08b00000 05100000 ec0d330c
timestamp differ 181656266
hd: 15 H1: 0 H2: 0 H3: 0 tr: 5 BxCounter: 3130 Event: 13 Orbit: 7
FPGA: 0 ASIC: 1 packet_type: 24 packet_id: 00 timestamp: 0ad3daca Data: fc3a3785 0001c400 09200000 03b00000 07200000 05b00000 08100000 05c00000
FPGA: 0 ASIC: 1 packet_type: 24 packet_id: 01 timestamp: 0ad3daca Data: 06900000 08500000 08e00000 08400000 08200000 08600000 08000000 0830019b
FPGA: 0 ASIC: 1 packet_type: 24 packet_id: 02 timestamp: 0ad3daca Data: 09800000 08100000 09600000 08b00000 05900000 09000000 07800000 07d00000
FPGA: 0 ASIC: 1 packet_type: 24 packet_id: 03 timestamp: 0ad3daca Data: 05f00000 08500000 08600000 08c00000 06b00000 07800000 07400000 07900000
FPGA: 0 ASIC: 1 packet_type: 24 packet_id: 04 timestamp: 0ad3daca Data: 08900000 07200000 07b00000 08100000 07100000 07100000 06200000 5133d9f4
hd: 15 H1: 0 H2: 0 H3: 0 tr: 5 BxCounter: 2827 Event: 13 Orbit: 0
FPGA: 0 ASIC: 0 packet_type: 24 packet_id: 00 timestamp: 0ad3daca Data: fb0b3405 00013000 06100000 06400000 08d00000 07d00000 0a500000 06000000
FPGA: 0 ASIC: 0 packet_type: 24 packet_id: 01 timestamp: 0ad3daca Data: 06400000 07900000 05900000 07800000 03f00000 07c00000 04900000 07f00000
FPGA: 0 ASIC: 0 packet_type: 24 packet_id: 02 timestamp: 0ad3daca Data: 04e00000 06600000 05700000 05700000 05600000 03100000 04a00000 03200000
FPGA: 0 ASIC: 0 packet_type: 24 packet_id: 03 timestamp: 0ad3daca Data: 03100000 04700000 05200000 06300000 05800000 03e00000 04e00000 05a00000
FPGA: 0 ASIC: 0 packet_type: 24 packet_id: 04 timestamp: 0ad3daca Data: 06000000 07600000 3ff00000 06b00000 05e00000 05800000 04400000 ec98fb1f
hd: 15 H1: 0 H2: 0 H3: 0 tr: 5 BxCounter: 3130 Event: 13 Orbit: 7
FPGA: 0 ASIC: 1 packet_type: 25 packet_id: 00 timestamp: 0ad3daca Data: fc3a3785 00017800 06a00000 05500000 03600000 05f00000 06100000 07000000
FPGA: 0 ASIC: 1 packet_type: 25 packet_id: 01 timestamp: 0ad3daca Data: 05d00000 05a00000 06d00000 05300000 06d00000 06000000 05600000 04900000
FPGA: 0 ASIC: 1 packet_type: 25 packet_id: 02 timestamp: 0ad3daca Data: 04f00000 08000000 06900000 06400000 07000000 04600000 07500000 05600000
FPGA: 0 ASIC: 1 packet_type: 25 packet_id: 03 timestamp: 0ad3daca Data: 07600000 06300000 03f00000 05600000 05c00000 05c00000 04200000 03a00000
FPGA: 0 ASIC: 1 packet_type: 25 packet_id: 04 timestamp: 0ad3daca Data: 05d00000 08e00000 07a00000 01900000 06a00000 02b00000 05a00000 dedc3d7c
...

```

Raw Data

- ◇ A custom program can send start_daq and store UDP packet
- ◇ red boxed fields require byte swapping, i.e. Big-endian

For firmware V4.11, the last UDP packet is truncated. And it also send a full zero packet when stopping DAQ.

```
[shyao@cats myprj]$ od -Ax -t x4 -N 12 data/rawdata.10.bin
```

```
000000 5300a005 00000000 00000000      UDP Header
```

```
00000c
```

```
[shyao@cats myprj]$ od -Ax -t x4 -j 12 -w40 -N 1452 data/rawdata.10.bin
```

```
00000c 002400a0 9c626808 85ad3cf5 00280100 00009005 0000d005 00006008 00001001 0000d009 0000b005
000034 012400a0 9c626808 0000a005 0000a00f 0000b005 00004007 0000b003 00005007 0000f003 00008007
00005c 022400a0 9c626808 00006004 00002006 0000f03f 0000f03f 00004005 0000f03f 00002004 0000e002
000084 032400a0 9c626808 00008002 00005004 0000a004 0000f005 00002005 00009003 0000c004 00005005
0000ac 042400a0 9c626808 0000b005 0000f03f 0000f03f 00005006 00006005 0000f03f 0000d003 cb3a5347
0000d4 002500a1 9c626808 05ad6bf6 00780100 0000c006 00005005 00008003 0000e005 0000c005 00002007
0000fc 012500a1 9c626808 0000f005 0000a005 0000d006 00004005 0000e006 00001006 00009005 00009004
000124 022500a1 9c626808 0000f004 00004008 0000a006 00002006 00003007 00005004 00002007 00006005
00014c 032500a1 9c626808 00008007 00006006 00002004 0000a005 0000d005 0000d005 00001004 0000d003
000174 042500a1 9c626808 0000d005 00002009 0000a007 00009001 0000a006 0000d002 0000e005 da35dee7
00019c 002500a0 9c626808 85ad3cf5 41010400 00001010 00000000 00009000 00001000 00000010 00001000
0001c4 012500a0 9c626808 00001000 00000000 00001000 00000000 00001000 00009017 00001000 00006015
0001ec 022500a0 9c626808 00009018 00001015 00000000 00001000 00001010 00001000 00001010 0000f017
000214 032500a0 9c626808 00000000 00003018 00001000 00001001 00000000 00000000 00000019 00001010
00023c 042500a0 9c626808 0000d016 00001010 00000010 00008019 fc010000 00001019 00002010 625b12cb
000264 002400a1 9c626808 05ad6bf6 00c80100 00000009 0000d003 00002007 0000a005 00002008 00009005
00028c 012400a1 9c626808 0000a006 00004008 0000f008 00006008 00001008 00008008 0000f007 00004008
0002b4 022400a1 9c626808 00006009 00000008 00003009 0000a008 00007005 0000f008 00008007 0000b007
0002dc 032400a1 9c626808 00000006 00003008 00006008 00008008 0000a006 00007007 00005007 00009007
000304 042400a1 9c626808 0000a008 00001007 0000d007 0000f007 0000f006 0000f006 0000f005 ce6c3496
00032c 002400a0 c5626808 85b13df5 00300100 0000a005 0000c005 00008008 00001001 0000c009 0000b005
000354 012400a0 c5626808 0000c005 0000600f 0000a005 00004007 0000b003 00005007 00000004 00006007
00037c 022400a0 c5626808 00008004 00002006 0000f03f 0000f03f 00002005 0000f03f 00006004 0000f002
0003a4 032400a0 c5626808 0000b002 00004004 0000c004 0000e005 00003005 0000a003 0000f004 00008005
0003cc 042400a0 c5626808 0000e005 0000f03f 0000f03f 00005006 00006005 0000f03f 0000f003 808acca5
0003f4 002500a1 c5626808 05b16cf6 00780100 0000d006 00004005 00009003 0000e005 0000e005 00001007
00041c 012500a1 c5626808 00000006 00009005 0000d006 00004005 0000f006 00001006 0000a005 0000b004
000444 022500a1 c5626808 00000005 00004008 0000a006 00005006 00002007 00004004 00005007 00006005
00046c 032500a1 c5626808 00008007 00008006 0000f003 0000a005 0000f005 0000e005 0000f003 0000d003
000494 042500a1 c5626808 0000d005 00000009 00009007 00008001 0000c006 0000d002 0000d005 f1be7c7b
0004bc 002500a0 c5626808 85b13df5 41010400 00001010 00000000 00009000 00001000 00000010 00001000
0004e4 012500a0 c5626808 00001000 00000000 00001000 00000000 00001000 00009017 00001000 00006015
00050c 022500a0 c5626808 00009018 00001015 00000000 00001000 00001010 00001000 00001010 0000f017
000534 032500a0 c5626808 00000000 00003018 00001000 00001001 00000000 00000000 00000019 00001010
00055c 042500a0 c5626808 0000d016 00001010 00000010 00008019 00000000 00001019 00002010 70dc1c4d
000584 002400a1 c5626808 05b16cf6 00c40100 00001009 0000d003 00001007 0000b005 00002008 0000c005
0005ac 5400a005 00000000 00000000      Next UDP Header
0005b8
----- 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
```

DAQ mode

- ◇ In practice, the external trigger signal is sent to the FPGA.
- ◇ Four possible mode
 - ◇ DAQ Push
 - ◇ Generator
 - ◇ Internal trigger
 - ◇ External trigger
- ◇ Common parameter - **Machine Gun**
 - ◇ The FPGA firmware generates a series of read commands and sends them to H2GCROC.
 - ◇ The real number of fast read commands sent will be the setting value plus 1.
 - ◇ This field has 8 digits, but the reasonable value range is 1 to 31.

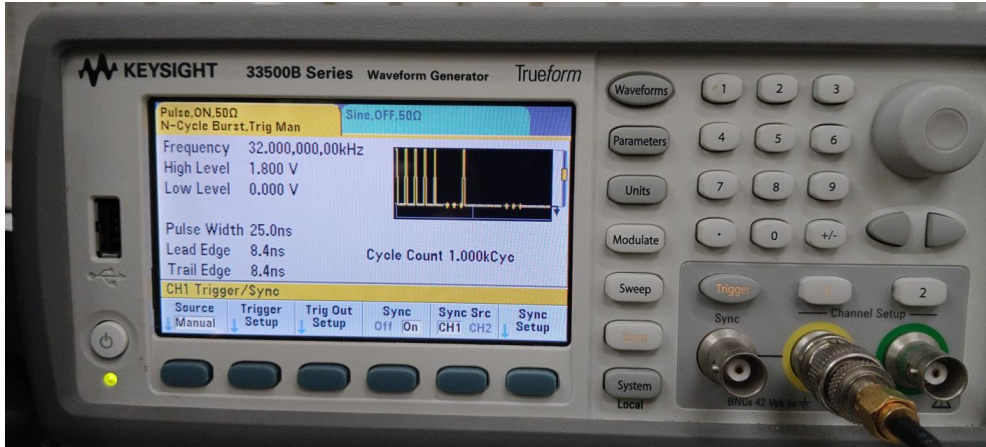
DAQ Push

- ◇ Software trigger
- ◇ Send this trigger by user interface

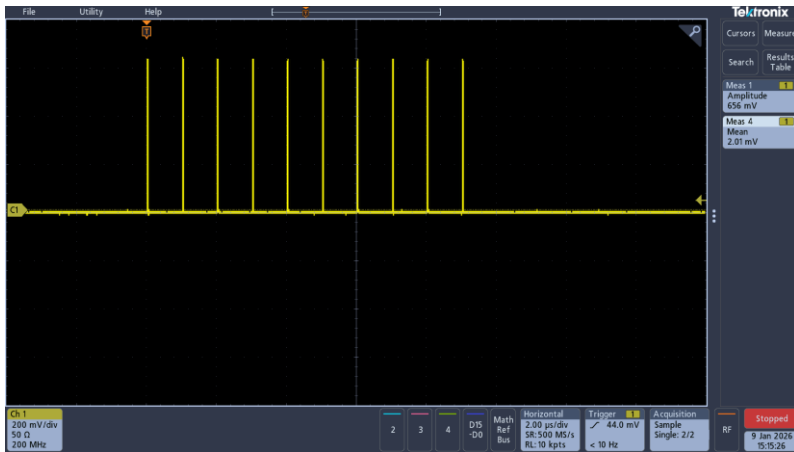
The screenshot displays the H2GDAQ@localhost.localdomain software interface, which is divided into several functional panels:

- Working Folder:** Shows the current path as /home/shyao/workspace/H2GDAQ-main, with buttons for 'Select Folder' and 'Print to Terminal'.
- Online Monitoring:** Includes a 'Run Monitoring' checkbox and a 'Select Monitoring Folder' button.
- DAQ Settings:** Features checkboxes for 'event', 'time [sec]', 'manual', and 'Reset Pack Counter', along with spinners for 'event' (set to 1) and 'time [sec]' (set to 1). A 'Run Number' spinner is also present, set to 1. A 'Data Count' label is followed by '-- byte'. Buttons for 'Auto Phase Scan', 'Set IODELAY', 'Start DAQ', and 'Stop DAQ' are provided.
- Generator / Debug Panel:** Contains multiple sections:
 - Data Collection:** Spinners for 'Data Coll En[7:0]' and 'Trig Coll En[7:0]', both set to 0.
 - Generator:** Includes checkboxes for 'Gen PreImp En' and 'Gen Pre Int', and spinners for 'Gen Nr Cycles' (set to 0) and 'Gen Interval' (set to 0).
 - External Trigger:** Includes checkboxes for 'Ext Trg En' and 'Ext Trg Delay', and spinners for 'Ext Trg Delay' (set to 0) and 'Ext Trg DeadT' (set to 0).
 - Fast Command:** Spinners for 'DAQ FCMD', 'DAQ Push FCMD', 'Gen Pre FCMD', and 'Gen FCMD', all set to 0.
 - Machine Gun:** A spinner for 'Machine Gun' is set to 0.
 - A checkbox for 'Send Gen Config before DAQ' is present, with a 'Send Gen Config' button below it.
- DAQ + Generator START/STOP:** A 'DAQ Push' button is highlighted with a red box. Below it are 'Start Generator' and 'Stop Generator' buttons.
- High Voltage / Reset/Adj:** Includes checkboxes for 'F0' and 'A0', and a 'Send HV Settings' button.

External trigger



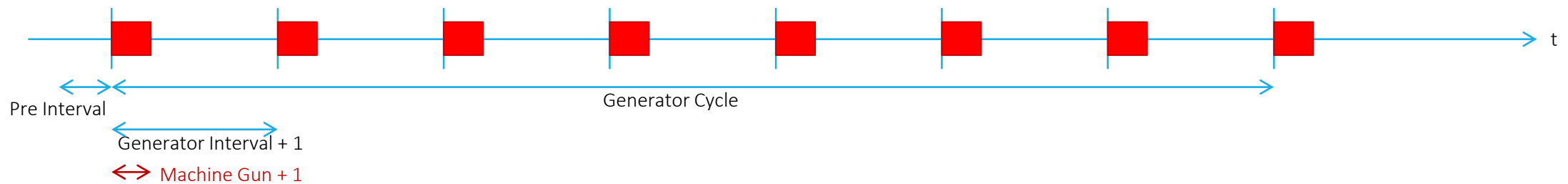
test signal : 32KHz, 1000 pulse



test signal : 1MHz, 10 pulse

Generator

- ◇ The generator function is enabled by user commands and is executed once after being enabled.
- ◇ Three parameters
 - ◇ Pre Interval – delay before the first fire, 16 bit, Unit 25ns
 - ◇ Generator Cycle - fire how many time, 32 bit
 - ◇ Generator Interval – Time between two fires, 32 bit, Unit 25ns
- ◇ Number of fast read command send = (Generator Cycle) * (Machine Gun + 1)



Generator

- ◇ Send parameter by user interface

The screenshot shows the H2GDAQ@localhost.localdomain interface. The 'Generator' tab is active, and the 'Debug' sub-tab is selected. The interface is organized into several panels:

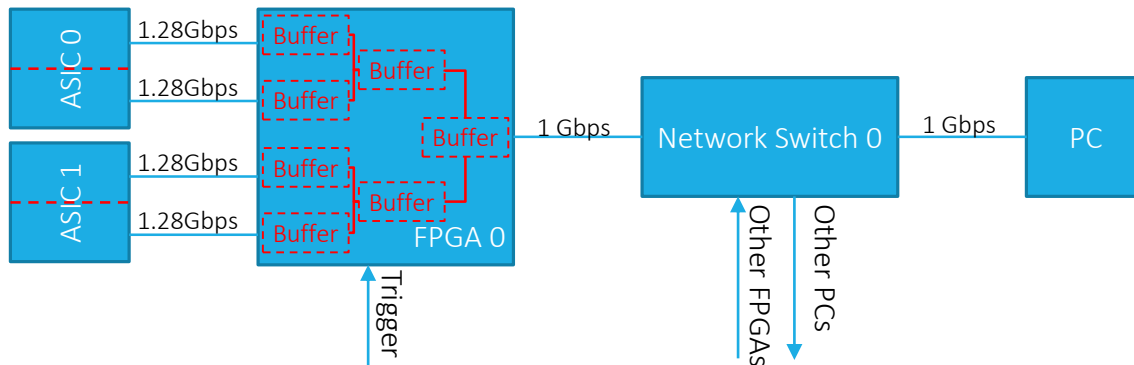
- Working Folder:** /home/shyao/workspace/H2GDAQ-main. Includes 'Select Folder' and 'Print to Terminal' buttons.
- Online Monitoring:** Includes a 'Run Monitoring' checkbox and a 'Select Monitoring Folder' button.
- DAQ Settings:** Includes checkboxes for 'event', 'time [sec]', 'manual', and 'Reset Pack Counter'. It also features a 'Run Number' spinner set to 1 and a 'Data Count' spinner set to 1. Buttons for 'Auto Phase Scan', 'Set IODELAY', 'Start DAQ', and 'Stop DAQ' are present.
- Generator (Debug):**
 - Data Collection:** 'Data Coll En[7:0]' and 'Trig Coll En[7:0]' spinners, both set to 0.
 - Generator:** A red box highlights 'Gen Pre Imp En' (checkbox), 'Gen Pre Int' (spinner, 0), 'Gen Nr Cycles' (spinner, 0), and 'Gen Interval' (spinner, 0).
 - External Trigger:** 'Ext Trg En' (checkbox), 'Ext Trg Delay' (spinner, 0), and 'Ext Trg DeadT' (spinner, 0).
 - Fast Command:** 'DAQ FCMD', 'DAQ Push FCMD', 'Gen Pre FCMD', and 'Gen FCMD' spinners, all set to 0.
 - Machine Gun:** A red box highlights the 'Machine Gun' spinner, set to 0.
 - A red box highlights the 'Send Gen Config' button at the bottom of this section.
- DAQ + Generator START/STOP:** Includes a 'DAQ Push' button and a red box highlighting 'Start Generator' and 'Stop Generator' buttons. Below this are 'High Voltage' and 'Reset/Adj' sections with 'F0' and 'A0' checkboxes.

Internal trigger

- ◇ No way was found to enable this feature from the user interface. Try using custom code.
- ◇ FPGA firmware manual say it only support ASIC 0

Data rate / trigger rate

- ◇ Each channel generates 4 bytes of data (include TOA, TOT, ADC) every 25 nanoseconds.
- ◇ A DAQ Frame is 1280+32 bits, it takes 1.025 microseconds to transmit data to FPGA.
- ◇ If maximum consecutively read (Machine Gun = 31)
 - ◇ It takes 32.8 microseconds to transmit data to FPGA.
 - ◇ Maximum trigger rate is about 6kHz

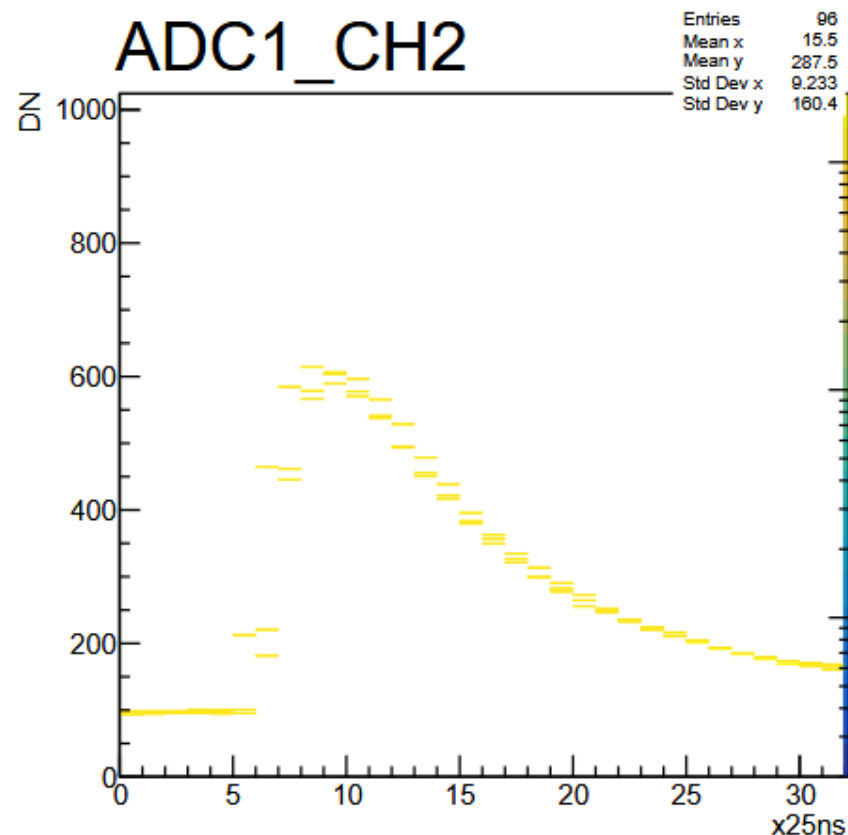
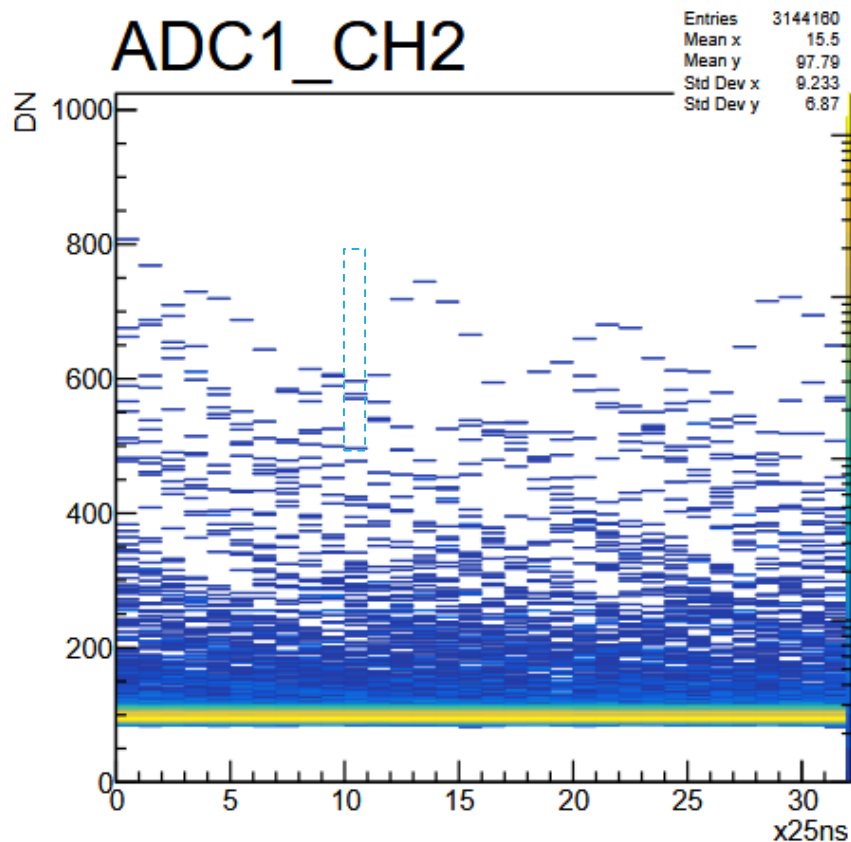


Data transfer in parallel, Each lane handle half channels. ~5 Gbps

1312 bits = header + half channel + crc + idle

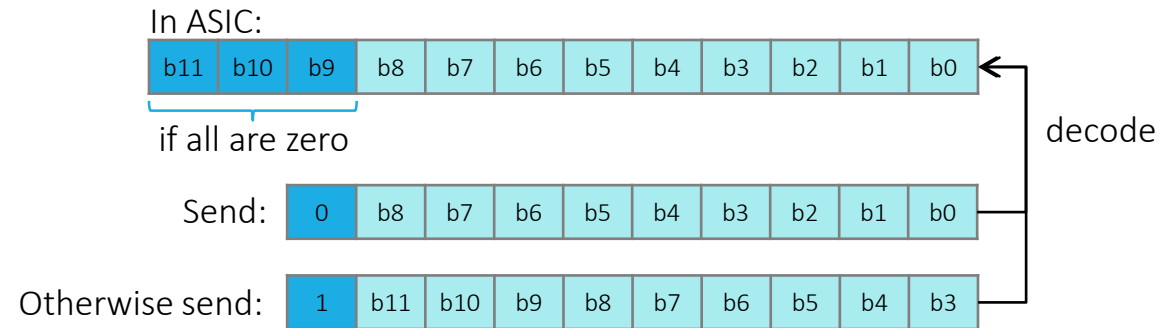
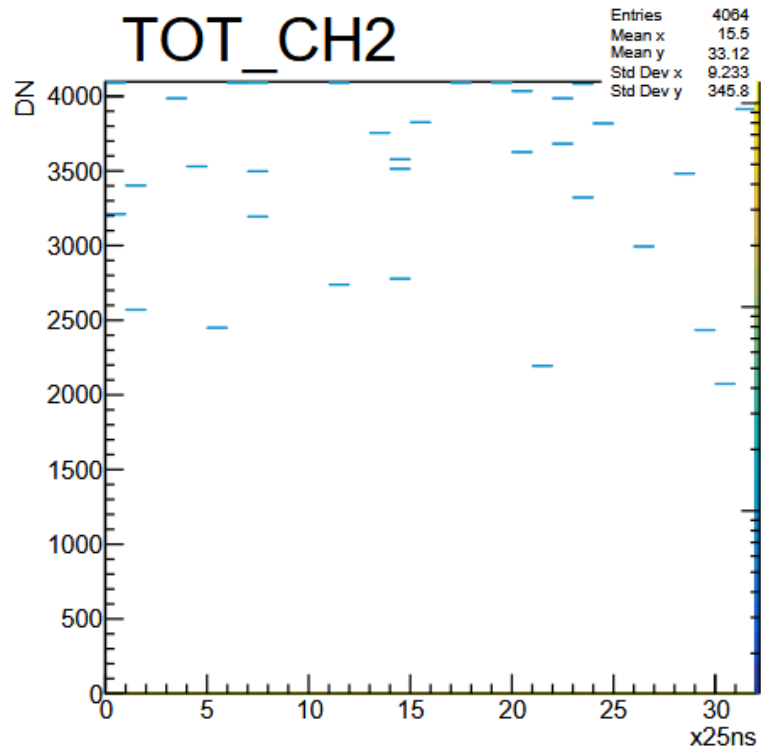
DAQ test

- ◇ Set generator parameter to 1KHz rate, 100k cycles. Set machine gun to 32 read
- ◇ Set High voltage to 28V.



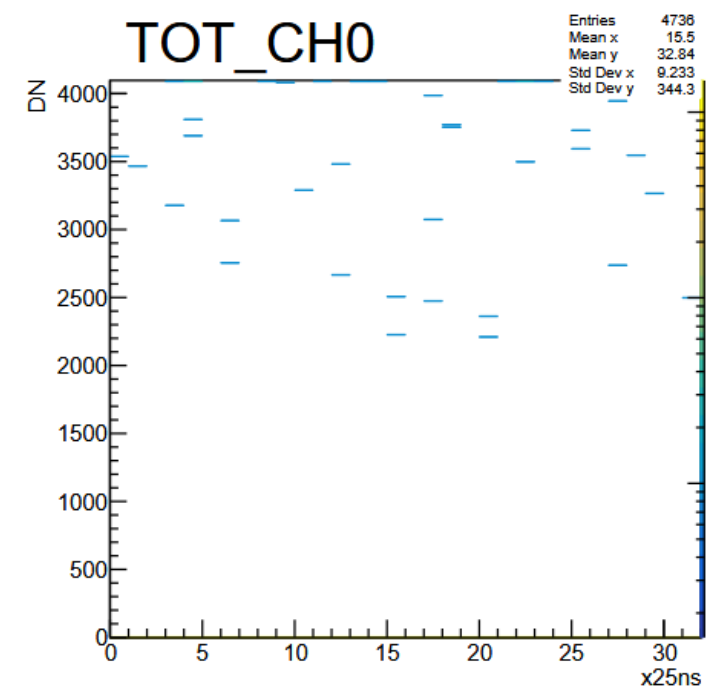
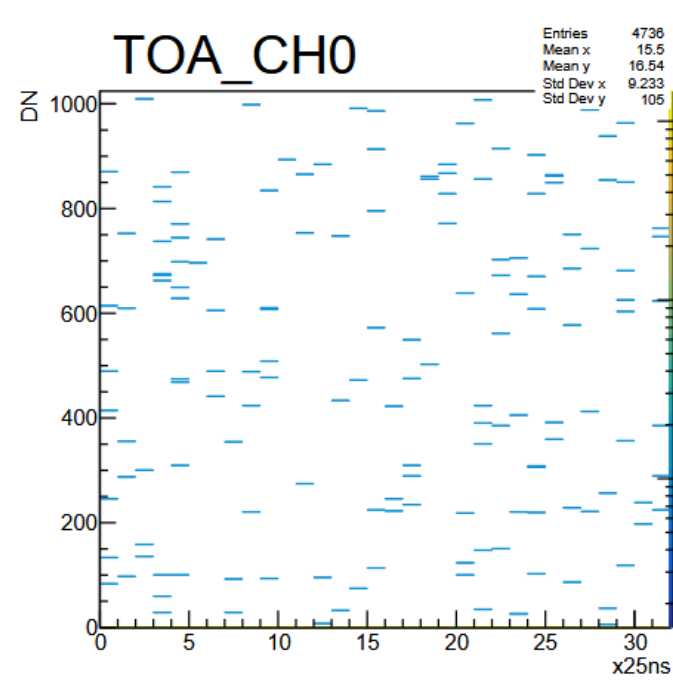
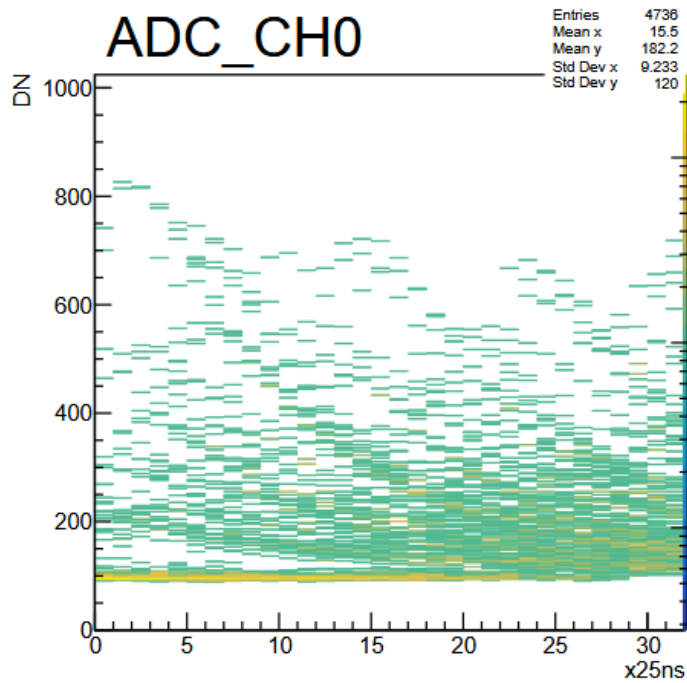
TOT

- ◇ TOT is 12 bit in ASIC, but send 10 bit only



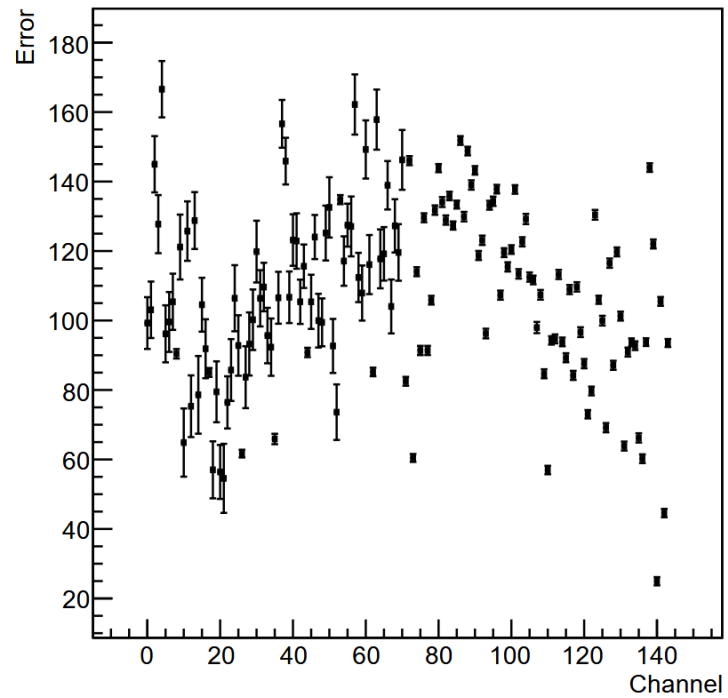
TOA

- ◇ TOA output is not zero when signal over the threshold
 - ◇ I use it as the filter
- ◇ 28V, Efficiency: $4736/32/98258 = \sim 0.1\% \sim 0.3\%$



Noise

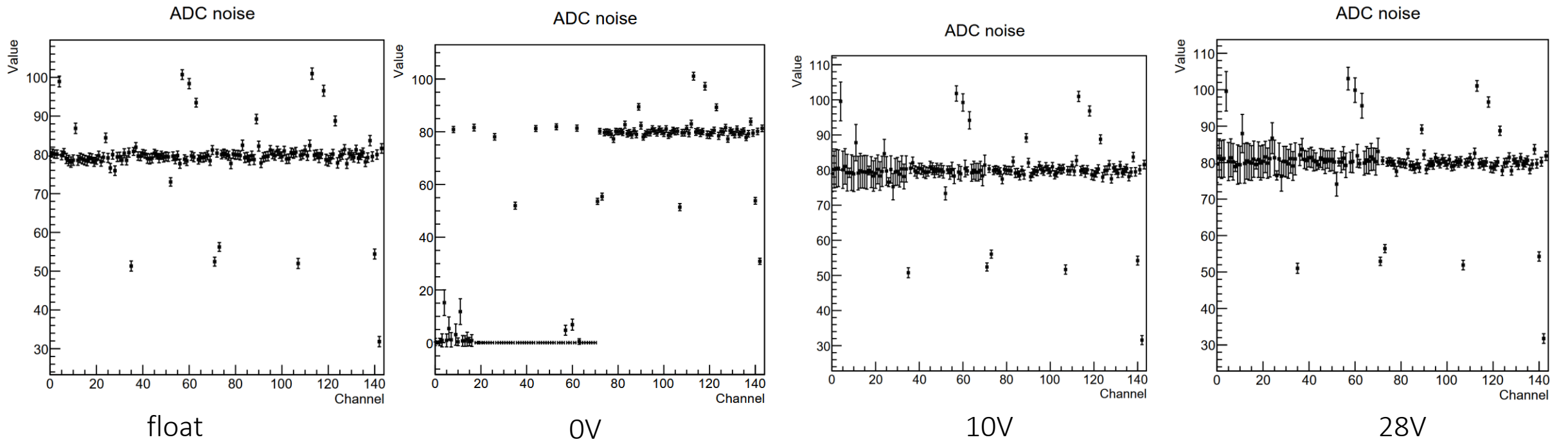
- ◇ 28V. Event Filter: (TOA == 0)
- ◇ pedestal calibration is wrong



28V Noise

Noise

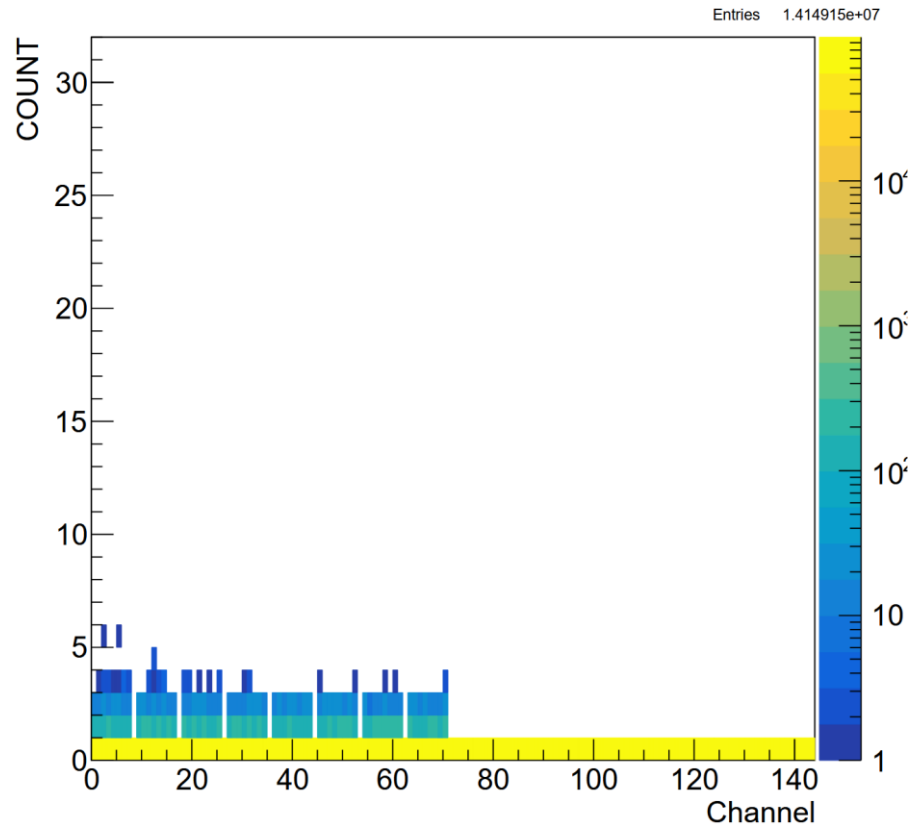
◇ Event Filter: (TOA == 0)



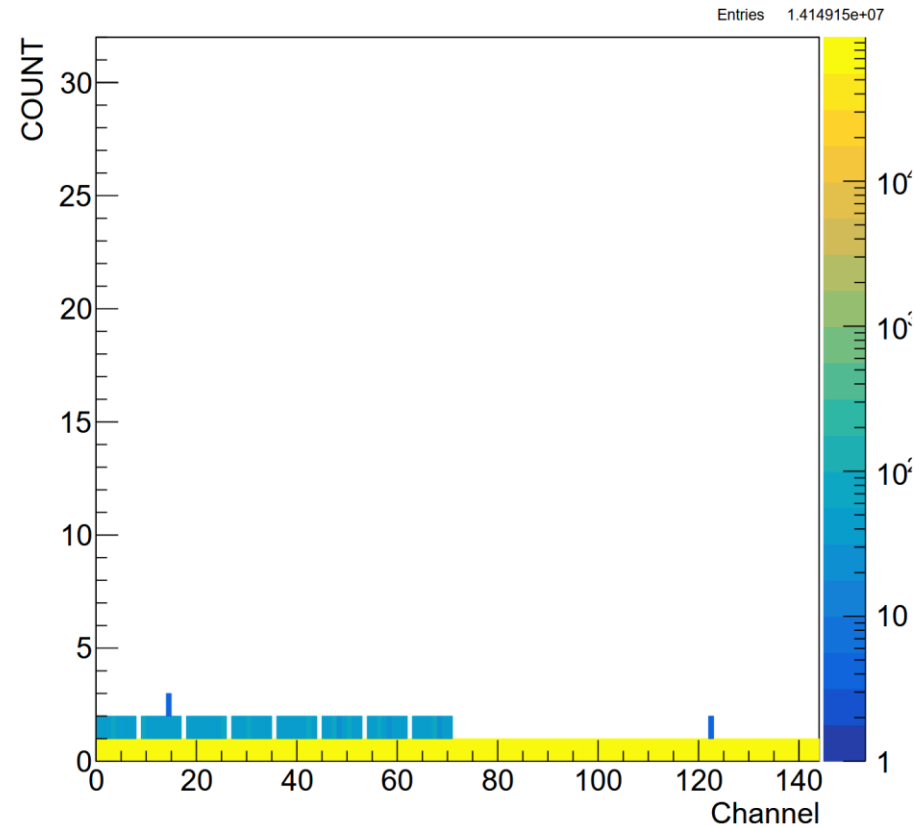
Count

◇ 28V

TOA COUNT

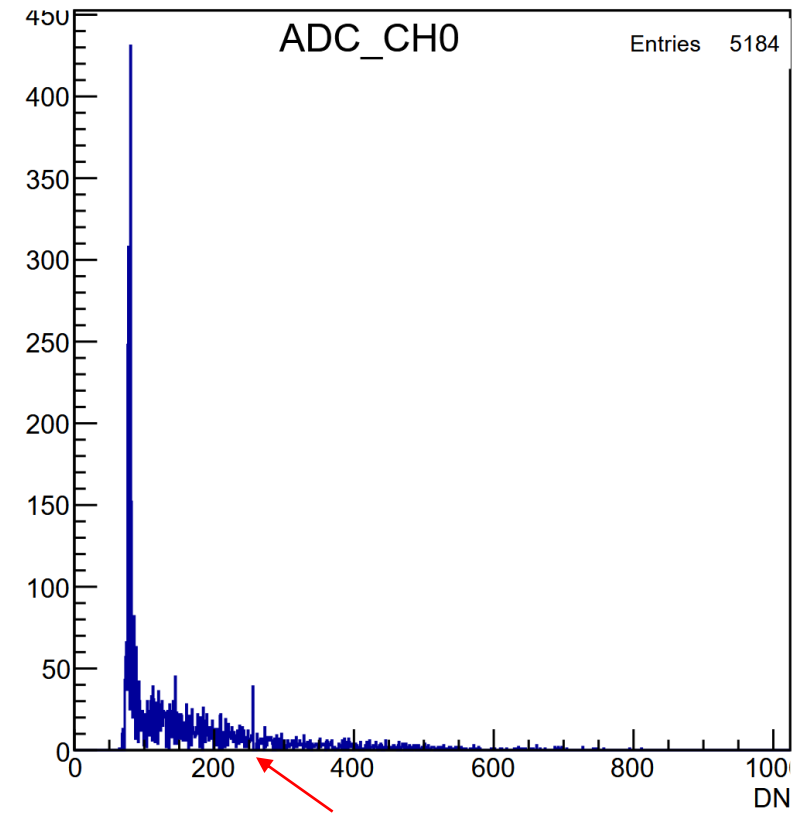
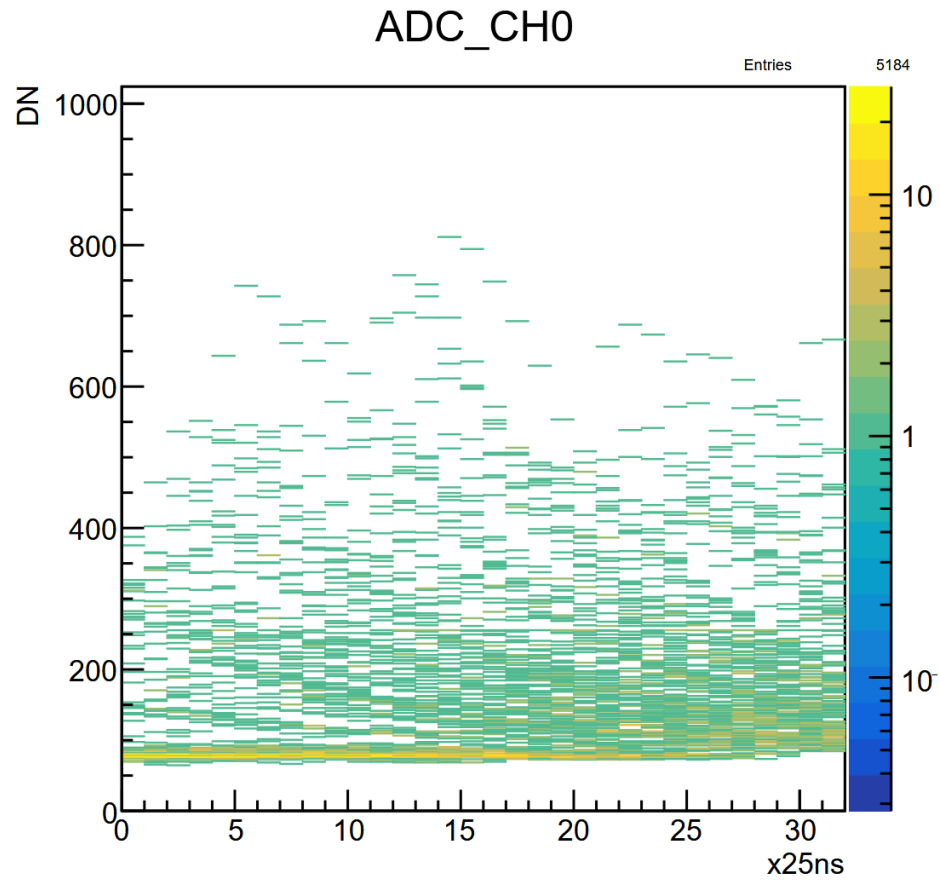


TOT COUNT



ProjectionY

- ◇ 28V. Event Filter: (TOA > 0)



All channel there is a peak.

HGCROC „ProtoBoard” – 5.069 V2 (Blue) V3 (Green) Setup for KCU105

In case of KCU105 and ProtoBoard
- 1 ProtoBoard: use the right side (LPC) slot
- 2 ProtoBoard: use both

FW version example: 5.64
„5” - Main version / „64” - Sub version
Subversion Start From:
0.. - 10GbE LPC / 64.. - 10GbE LPC+HPC
128.. - 1GbE LPC / 192.. - 1GbE LPC+HPC

FW handles I2C address shifting. Set the ASIC addresses to 0 and 1.
No resistor soldering / changing is necessary!

Ext. CLK (40MHz) In P - LVDS
Ext. CLK (40MHz) In N - LVDS
Ext. Trigger In - CMOS 1.8V
Ext. Sync In - CMOS 1.8V

Not Used
For Board Programming
1GbE Connection

Packet Size:
- Control - 46 Byte (Complete: 74+14)
- Data Long - 1358 Byte (Complete: 1386+14)
- Data Jumbo - 8846 Byte (Complete: 8874+14)

10GbE - ! Use the upper slot !
Tested with: TP-Link TL-SM5310-T
No any special requirement

High Voltage

High Voltage

High Voltage A and High Voltage B is not connected to each other

LED7 - PM BUS / Board Config Ready (Left)
LED6 - External Clock
LED5 - 160MHz Clock ok, ≈1Hz Blinking
LED4 - COMM_ETH_STATUS(0) and COMM_ETH_READY and COMM_MAC_RECEIVED
LED3 - COMM TX ACTIVE
LED2 - COMM RX ACTIVE
LED1 - DAQ Started
LED0 - Internal generator is running (Right)

Ext. Sync Out - CMOS 3.3V - Not Used
PM I2C SDA - DBG - CMOS 3.3V - Not Used
PM I2C SCL - DBG - CMOS 3.3V - Not Used

SiPM Calib1 - CMOS 3.3V - Not Used
SiPM Calib0 - CMOS 3.3V - Not Used

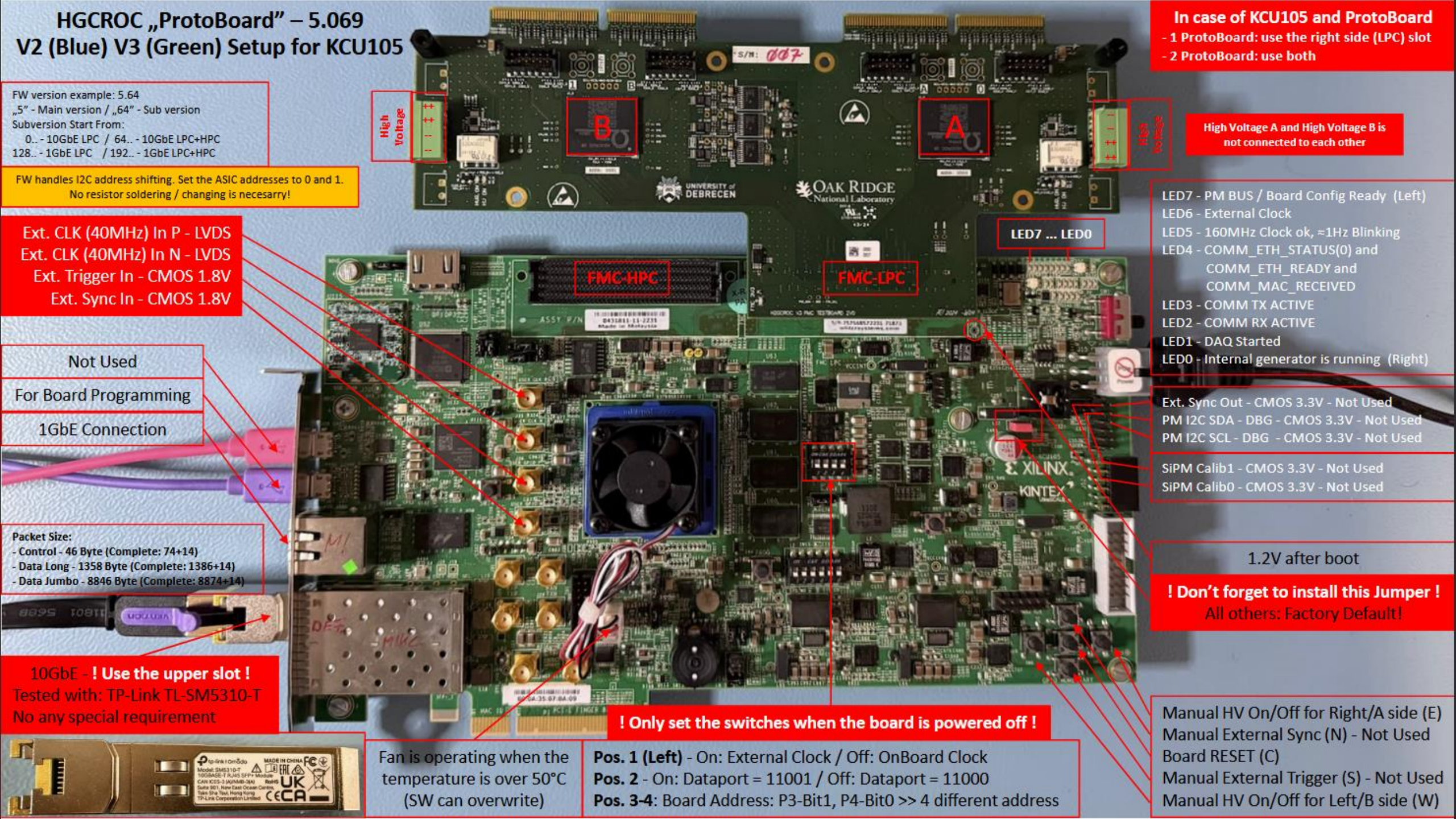
1.2V after boot
! Don't forget to install this Jumper !
All others: Factory Default!

Manual HV On/Off for Right/A side (E)
Manual External Sync (N) - Not Used
Board RESET (C)
Manual External Trigger (S) - Not Used
Manual HV On/Off for Left/B side (W)

Fan is operating when the temperature is over 50°C
(SW can overwrite)

! Only set the switches when the board is powered off !

Pos. 1 (Left) - On: External Clock / Off: OnBoard Clock
Pos. 2 - On: Dataport = 11001 / Off: Dataport = 11000
Pos. 3-4: Board Address: P3-Bit1, P4-Bit0 >> 4 different address



10G HW

- ◇ FMC Cable: HDR-169468-01 or HDR-169468-02 (4 rows)

- ◇ <https://www.digikey.tw/zh/products/detail/samtec-inc/HDR-169468-01/6678214>

- ◇ 10GbE, SFP to RJ45: TP-Link TL-SM5310-T

- ◇ For PC with Thunderbolt 4: QNAP qna-t310g1t

- ◇ <https://www.qnap.com/zh-tw/product/qna-t310g1t>

- ◇ 10Gbps Switch – Netgear XS512EMv2 NT\$39000

- ◇ CAT6a/CAT.7 Cable, SSD

- ◇ todo



NT\$ 1900 x4



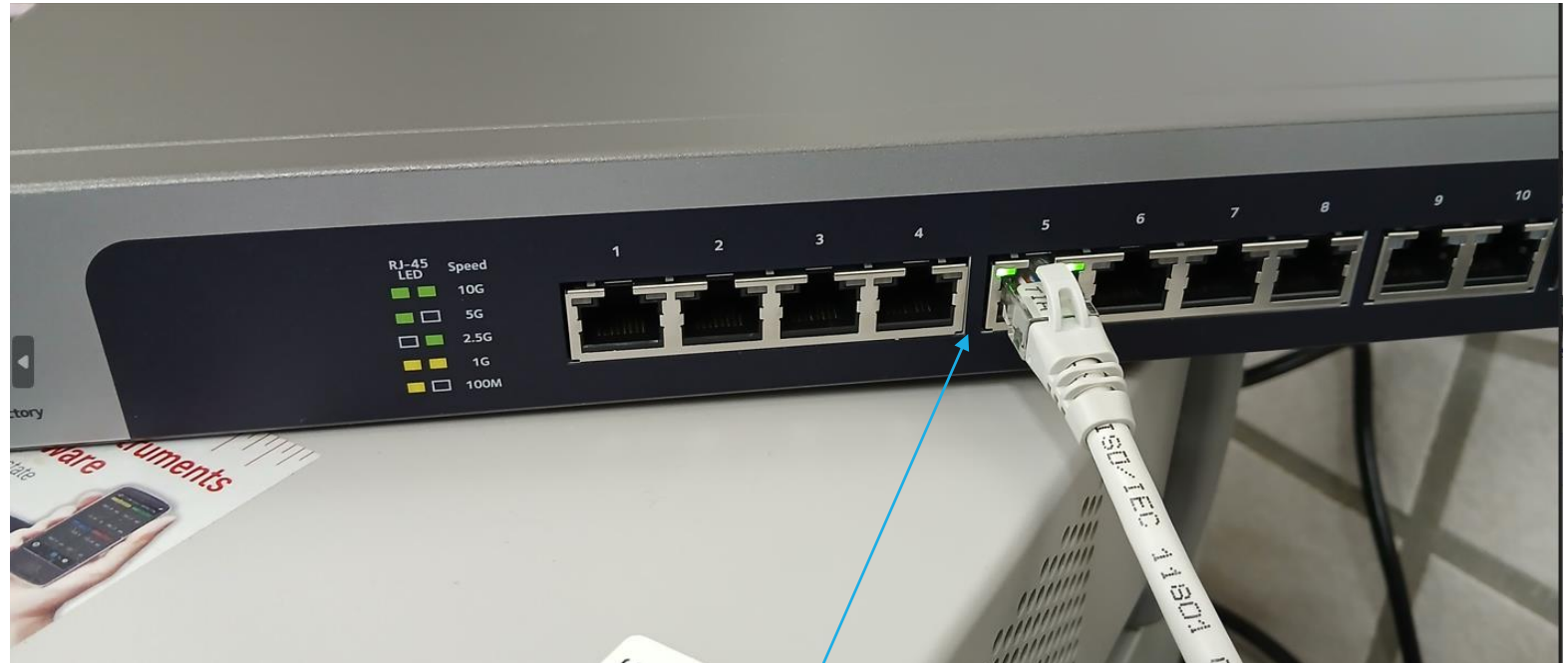
NT\$ 5084 x4



NT\$ 6990 x1

10G Hardware is ready

◇ 2026/3/4

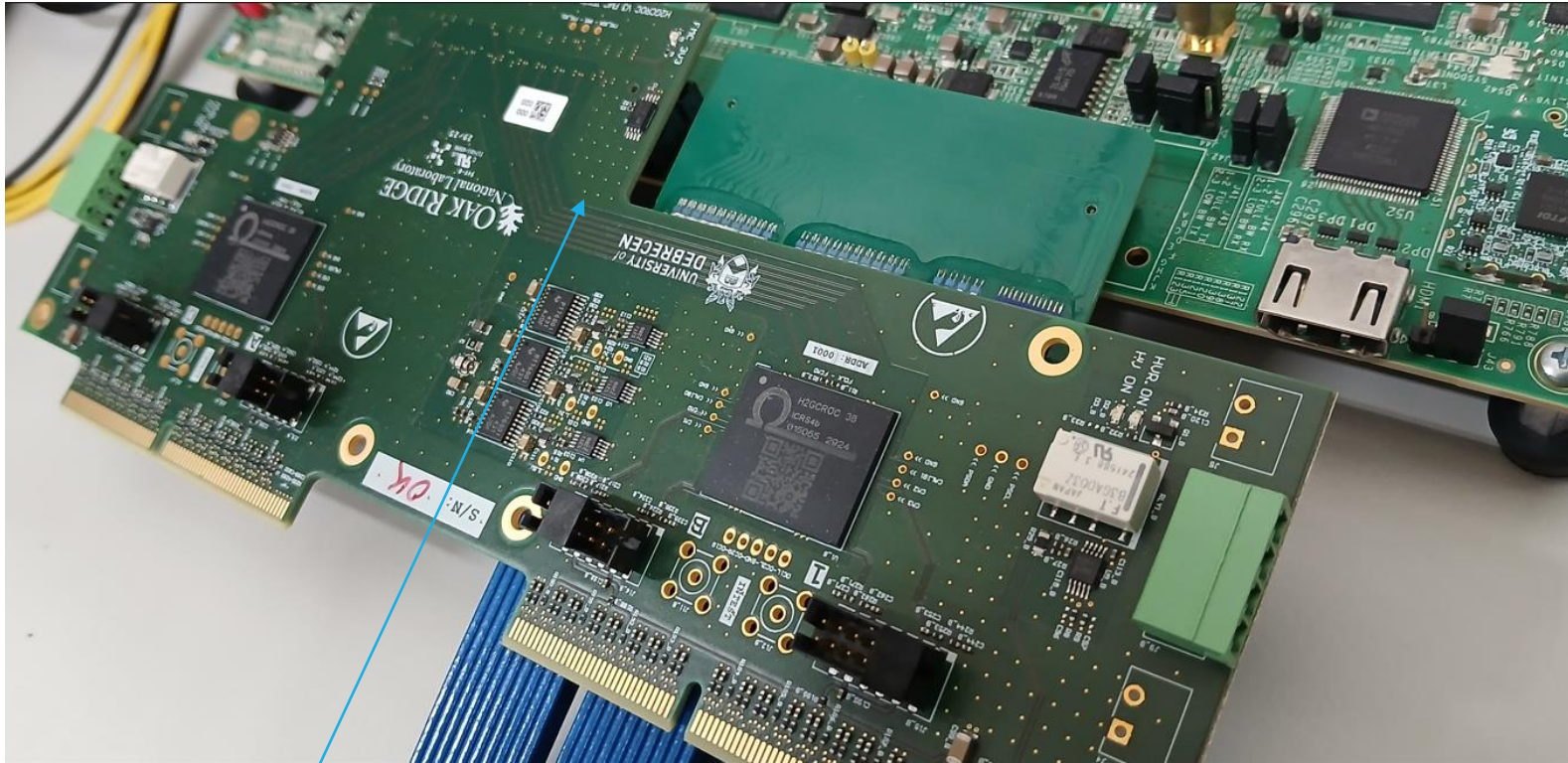


10G is work



Test FMC Cable

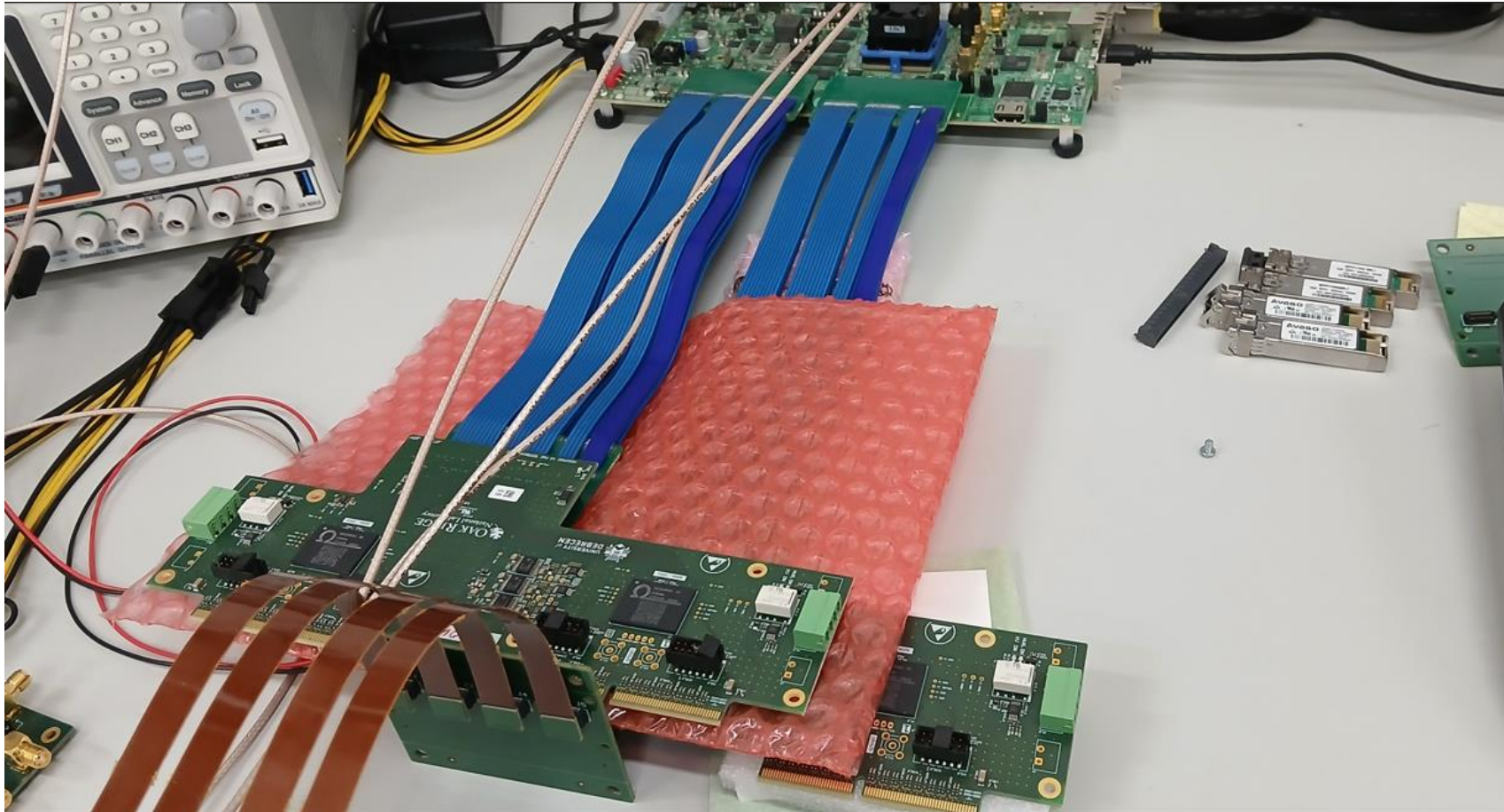
- ◇ The cable is very stiff, unable to insert



The LPC connector damaged and frequent failures in I/O delay adjustment.

Test FMC Cable

◇ Currently



Test IO delay with FMC cable

- ◇ The I/O alignment script provided by the new firmware seems to work correctly.
- ◇ The LPC connector damaged and frequent failures in I/O delay adjustment. This is an old problem, but the advent of FMC cables has made it even more serious.

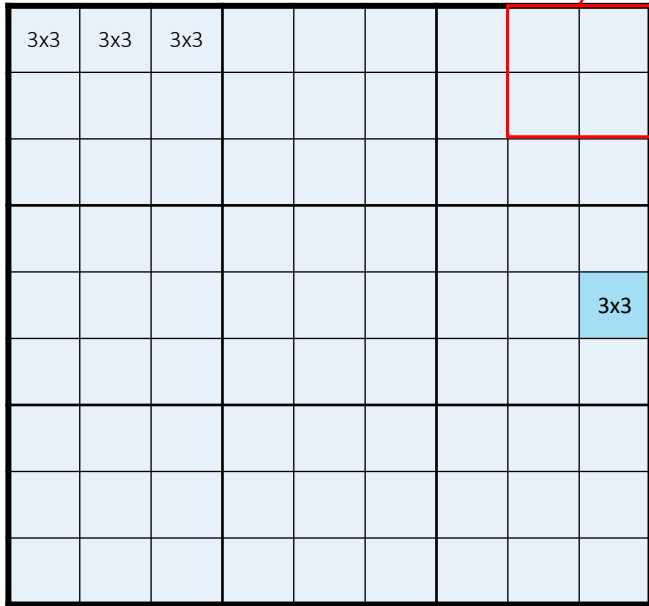
```
$ python 003_10G_Test_Align.py
Test ASIC 0...
Delay: 46
```

	ASIC	CMD	DLY10	AR	Data1	Data0	Trig3	Trig2	Trig1	Trig0
Iteration #000:	0xa0	0x0c	2e/2e	0x1d	06666666	accccccc	2cc444c4	accccccc	06666666	accccccc
Iteration #001:	0xa0	0x0c	2e/2e	0x1d	06666666	accccccc	accccccc	accccccc	06466666	accccccc
Iteration #002:	0xa0	0x0c	2e/2e	0x1d	06666666	accccccc	24444444	accccccc	06666666	2ec44646
Iteration #003:	0xa0	0x0c	2e/2e	0x1d	06666666	accccccc	acccc4c4	accccccc	06466666	accccccc
Iteration #004:	0xa0	0x0c	2e/2e	0x1d	06666666	accccccc	accccccc	accccccc	06464666	accccccc
Iteration #005:	0xa0	0x0c	2e/2e	0x1d	06666666	accccccc	2ccc4444	accccccc	06666666	accccc4
Iteration #006:	0xa0	0x0c	2e/2e	0x1d	06666666	accccccc	accccccc	accccccc	06444446	accccccc
Iteration #007:	0xa0	0x0c	2e/2e	0x1d	06666666	accccccc	accccccc	accccccc	06464666	accccccc
Iteration #008:	0xa0	0x0c	2e/2e	0x1d	06666666	accccccc	accccccc	accccccc	06466666	accccccc
Iteration #009:	0xa0	0x0c	2e/2e	0x1d	06666666	accccccc	accccccc	accccccc	04444444	accccccc
Iteration #010:	0xa0	0x0c	2e/2e	0x1d	06464666	accccccc	accccccc	accccccc	04444444	accccccc
Iteration #011:	0xa0	0x0c	2e/2e	0x1d	06666666	accccccc	accccccc	accccccc	04444444	accccccc
Iteration #012:	0xa0	0x0c	2e/2e	0x1d	06666666	accccccc	2cc44444	accccccc	06666666	accccc4
Iteration #013:	0xa0	0x0c	2e/2e	0x1d	06666666	accccccc	accccccc	accccccc	04444444	accccccc
Iteration #014:	0xa0	0x0c	2e/2e	0x1d	06666666	accccccc	acc4c444	accccccc	06666666	accccc4
Iteration #015:	0xa0	0x0c	2e/2e	0x1d	06666666	accccccc	accccc4	accccccc	06466666	accccccc
Iteration #016:	0xa0	0x0c	2e/2e	0x1d	06666666	accccccc	accccccc	accccccc	06466666	accccccc
Iteration #017:	0xa0	0x0c	2e/2e	0x1d	06666666	accccccc	accccc4	accccccc	06466666	accccccc
Iteration #018:	0xa0	0x0c	2e/2e	0x1d	04464666	accccccc	accccccc	accccccc	24444444	accccccc
Iteration #019:	0xa0	0x0c	2e/2e	0x1d	06666666	accccccc	2cc44444	accccccc	06666666	acccc4c4

Channel

2026/8 beam test size, need 8 H2GCROC

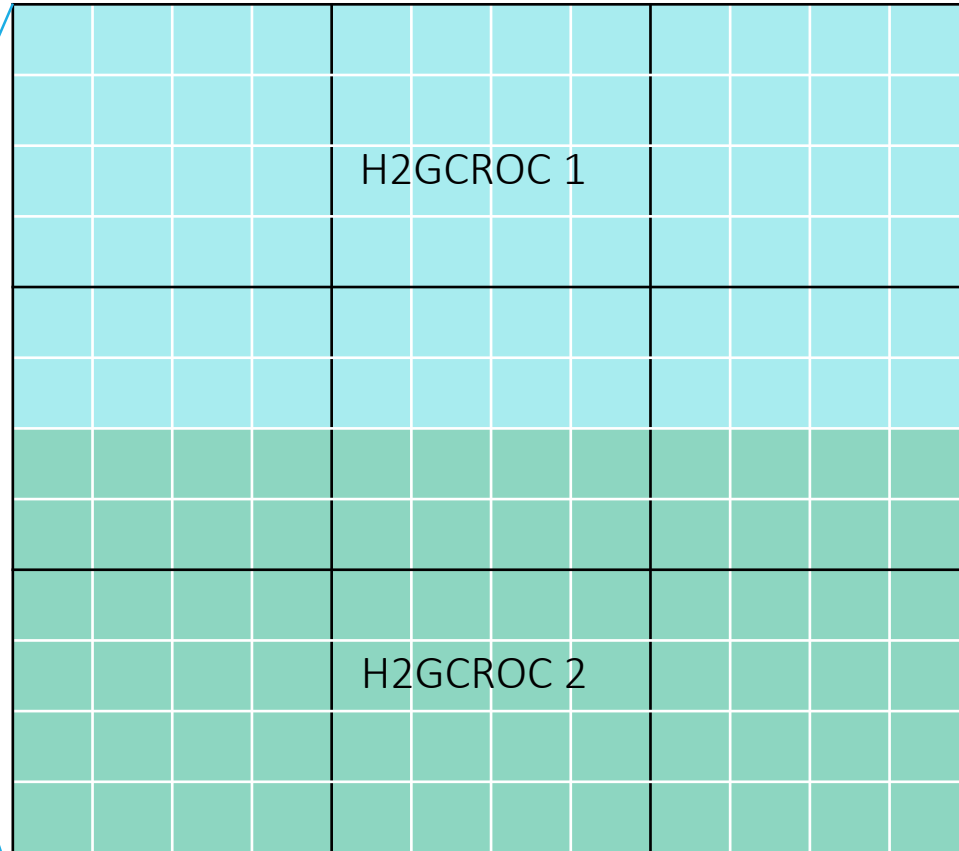
◇ d



1 EMCAL = 9 Super module
 1 Super module = 9x 3x3 Crystal
 3x3Crystal = 2 H2GCROC

=>

1 EMCAL = 162 H2GCROC = 20.16 FPGA



$3 \times 3 \times 16 = 144$ SiPM













$144 = 2 \times \text{H2GCROC}$

How to connect

Newer FPGA firmware

- ◇ Firmware 5.069 => main version 5, subversion 69
 - ◇ subversion 0 - 63: 10GbE + LPC
 - ◇ subversion 64 - 127: 10GbE + LPC + HPC
 - ◇ subversion 128 - 191: 1GbE + LPC
 - ◇ subversion 192 - ????: 1GbE + LPC + HPC
- ◇ Since main version 5, the communication format has changed, need newer software.
- ◇ The newer calibration software develop by Shihai Jia, it combine H2GConfig & H2GCalib
 - ◇ <https://gitlab.cern.ch/sjia/h2gcalibx>
- ◇ Need a new DAQ software to replace H2GDAQ, online monitoring and decode
 - ◇ ... ask them

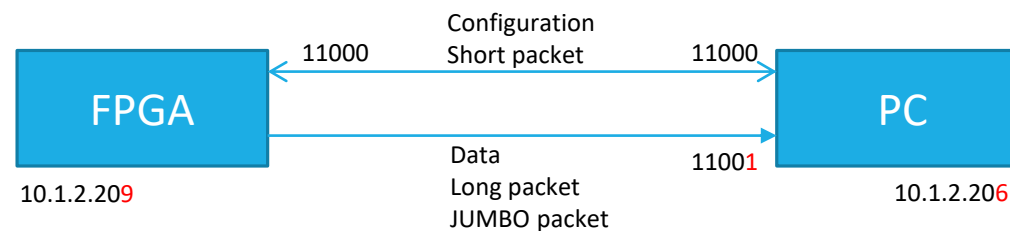
GGGGGGGGGG
Alexander_Norbert / GGGGGGGGGG

	5.000
	5.001
	5.066
	5.067
	5.068
	5.069
	5.070
	5.197
	5.199
	200_10G_Test_001.py
	release.txt
	toplevel_test_teng.bit

Multiple FPGA Setting

- ◇ DIP1 Select Clock: OFF = on board clock, ON = External clock.
- ◇ DIP2 Data Port: OFF = 11000, ON = 11001
- ◇ DIP3 & DIP4: Select IP address

DIP3, DIP4	FPGA IP	Config. Port	PC IP	PC Config Port	PC Data port
OFF,OFF	10.1.2.208	11000	10.1.2.207	11000	11000 + DIP2
OFF,ON	10.1.2.209	11000	10.1.2.206	11000	11000 + DIP2
ON,OFF	10.1.2.210	11000	10.1.2.205	11000	11000 + DIP2
ON,ON	10.1.2.211	11000	10.1.2.204	11000	11000 + DIP2



Ethernet firewall

- ◇ Use external and internal firewall zone
- ◇ Add 10G interface enp46s0

```
$ firewall-cmd --zone=public --remove-interface=enp46s0
$ firewall-cmd --zone=internal --add-interface=enp46s0
$ firewall-cmd --zone=internal --add-port=11000-11100/udp
$ firewall-cmd --runtime-to-permanent
$ firewall-cmd --zone=internal --list-all
internal (active)
  target: default
  icmp-block-inversion: no
  interfaces: enp46s0
  sources:
  services: cockpit dhcpv6-client mdns mountd nfs nfs3 rpc-bind samba samba-client ssh
  ports: 11000-11100/udp
  protocols:
  forward: yes
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
```

multiple IP

- ◇ Setting PC IP address for 10G device

```
$ nmcli con down "enp46s0"  
$ nmcli con modify "enp46s0" ipv4.addresses "10.1.2.207/24" ethernet.mtu 9000 ipv4.method manual  
$ nmcli con up "enp46s0"
```

- ◇ For testing, assign multiple IP addresses to a single interface

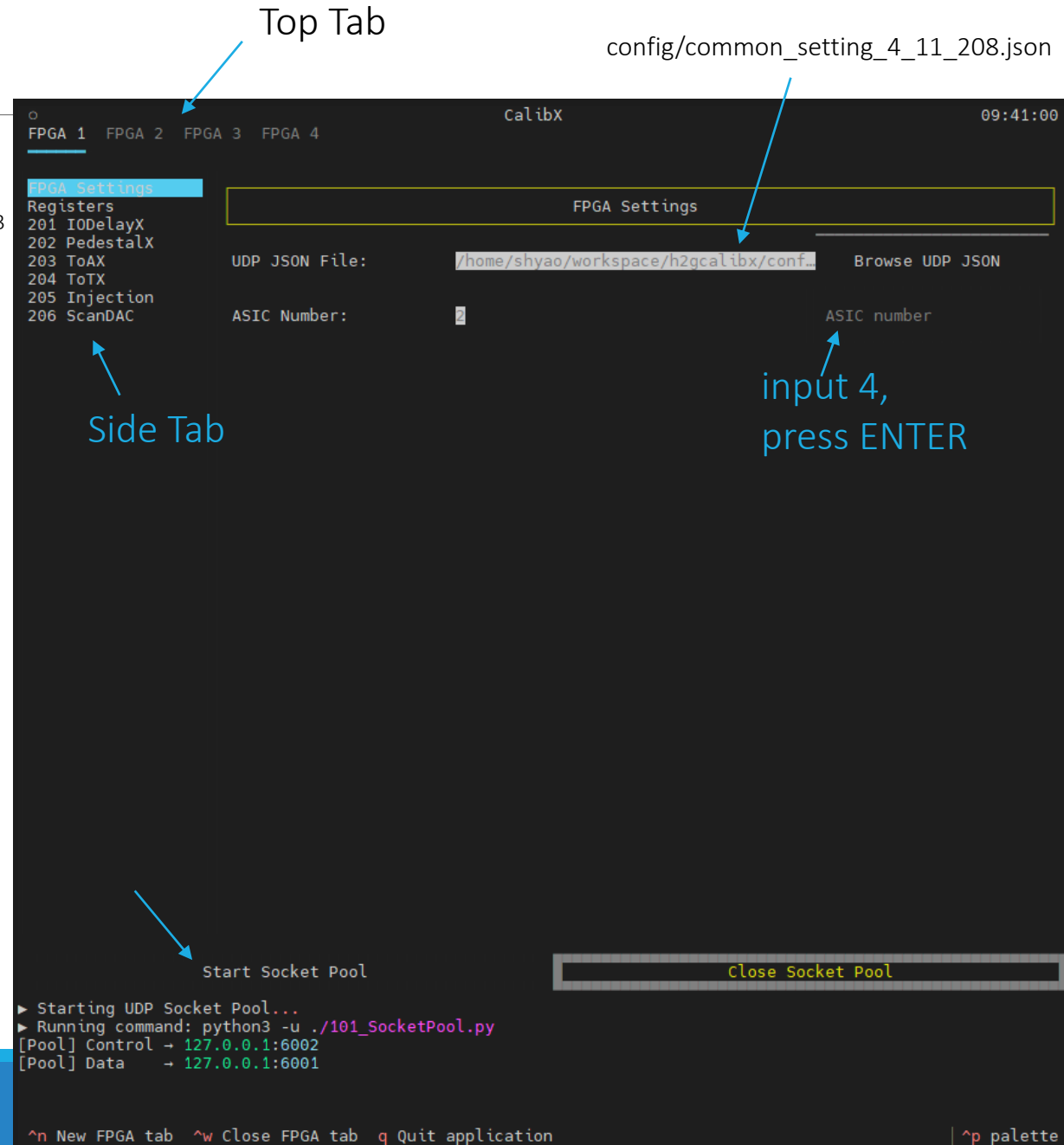
```
$ nmcli con down "enp46s0"  
$ nmcli con modify "enp46s0" ipv4.addresses  
"10.1.2.207/24,10.1.2.206/24,10.1.2.205/24,10.1.2.204/24" ethernet.mtu 9000 ipv4.method manual  
$ nmcli con up "enp46s0"
```

```
$ nmcli dev status  
$ nmcli con show
```

Software h2gcalibx

- ◇ git clone https://gitlab.cern.ch/sjia/h2gcalibx
58e4fd28c9f44 2026/2/23
- ◇ conda create -n h2gc -c conda-forge
"python=3.12.3" loguru matplotlib numpy pandas
textual tqdm pytest textual-image pymupdf pillow
gnuplot
- ◇ conda activate h2gc
- ◇ python 200_UI.py

This user interface renders all content in the terminal and is responsive to mouse input.



Side Tab Registers

- ◇ After calibration, you can use this UI to upload parameter to ASIC

The screenshot displays the CalibX configuration tool interface. The top bar shows 'calibX' and the time '09:45:32'. Below the top bar, there are tabs for 'FPGA 1', 'FPGA 2', 'FPGA 3', and 'FPGA 4'. The left sidebar contains 'FPGA Settings' with a sub-menu 'Registers' highlighted in blue. A red arrow points from this menu to the main content area. The main content area is titled 'Register Settings' and is divided into two columns: 'ASIC 0' and 'ASIC 1'. The 'ASIC 0' column lists registers from 'Channel_36' to 'Channel_62'. The 'ASIC 1' column shows 'register_0' through 'register_3'. The 'register_0' section is expanded, showing 'Conveyor Gain: <0> = 0.025, <1> = 0.05' and 'Input DAC value'. Below this, there is a section for 'External trigger selection for TOA' with a table of settings for '0' and '1'. At the bottom of the interface, there are buttons for 'Start Socket Pool' and 'Close Socket Pool', and a terminal window showing the status of the socket pool.

```
Starting UDP Socket Pool...
Running command: python3 -u ./101_SocketPool.py
[Pool] Control → 127.0.0.1:6002
[Pool] Data → 127.0.0.1:6001
```


202 PedestalX

- ◇ Pre-amp reset default setting
 - ◇ $R_f = 8 \Rightarrow 80K \text{ Ohm}$
 - ◇ $C_f = 10 \Rightarrow 500 \text{ fF}$
 - ◇ $C_{f_Comp} = 10 \Rightarrow 500 \text{ fF}$
 - ◇ $\tau = R_f * (C_f + C_{f_Comp}) = 80\text{ns} = 3 \text{ samples}$
- ◇ Conveyor Gain default setting
 - ◇ $CC = 8 \Rightarrow \text{Gain} = 0.2$

CC	0.025, 0.05, 0.1, 0.2	
C_d (pF)	5, 10, 20	At the conveyor output and at the preamplifier input. To ensure the preamp stability
R_f (Ohm)	10K, 20K, 40K, 80K	In parallel, these resistors provide 15 values to be adjusted with the C_f and C_{f_comp} values
C_f (fF)	50, 100, 200, 400	Added with the C_{f_comp} capacitors, provide the preamplifier constant time with R_f
C_{f_comp} (fF)	50, 100, 200, 400	Same purpose than C_f capacitors but connected differently to improve the preamplifier stability.

The screenshot shows the CalibX configuration tool for the 202 PedestalX. Key settings include:

- Target pedestal: 100
- Template JSON File: /home/shyao/workspace/h2gcalibx/config/default_2024Aug_config.json
- Feedback Resistor (RF): 8 (Unit: 10K (0-15))
- Feedback Capacitor (CF): 10 (Unit: 15fF (0-15))
- Current Conveyor Gain (CC): 8 (Unit: 0.025 (0-15))
- CF Compensation (CFComp): 10 (Unit: 15fF (0-15))

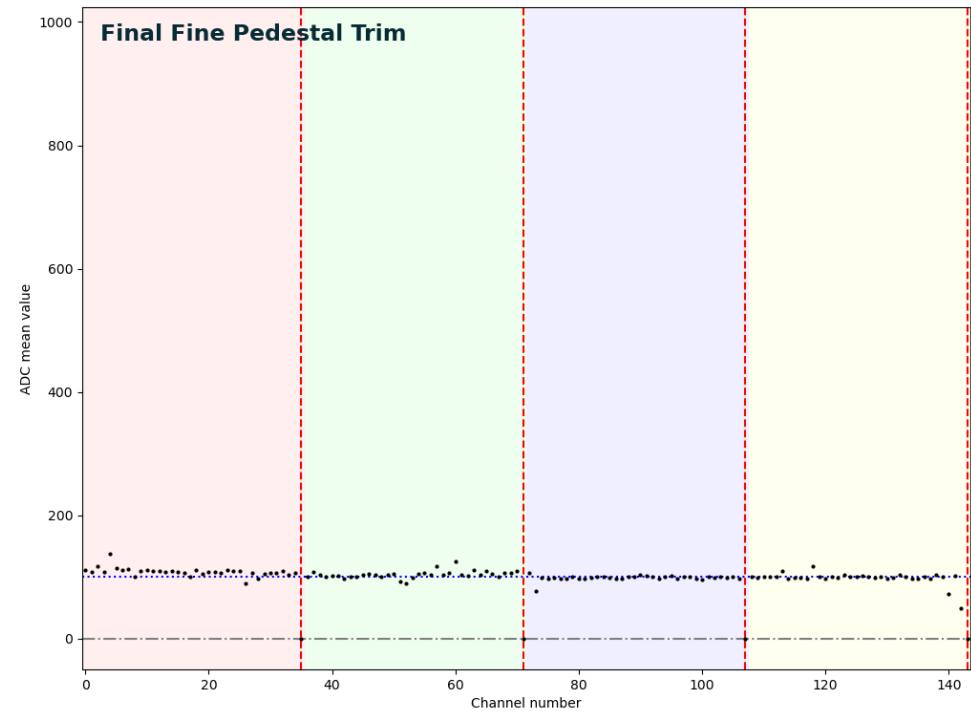
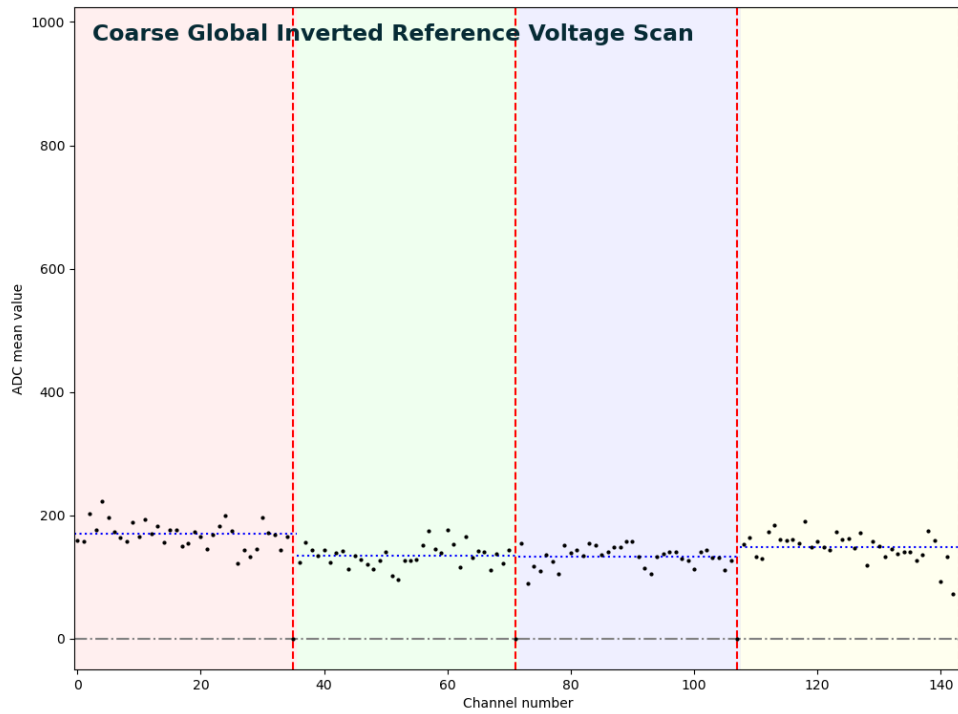
The 'Final Tune' log shows the following data for 15 attempts:

```

-- Final Tune Attempt 4 / 17:
--A0: Ch03 115 Ch04 150 Ch05 125 || Ch39 102 Ch40 99 Ch41 97 ||
--A1: Ch03 98 Ch04 100 Ch05 101 || Ch39 101 Ch40 105 Ch41 115 ||
-- Final Tune Attempt 5 / 17:
--A0: Ch03 91 Ch04 124 Ch05 98 || Ch39 98 Ch40 99 Ch41 98 ||
--A1: Ch03 103 Ch04 100 Ch05 101 || Ch39 101 Ch40 104 Ch41 113 ||
-- Final Tune Attempt 6 / 17:
--A0: Ch03 95 Ch04 132 Ch05 105 || Ch39 100 Ch40 95 Ch41 102 ||
--A1: Ch03 99 Ch04 100 Ch05 100 || Ch39 100 Ch40 103 Ch41 111 ||
-- Final Tune Attempt 7 / 17:
--A0: Ch03 112 Ch04 141 Ch05 114 || Ch39 99 Ch40 100 Ch41 101 ||
--A1: Ch03 99 Ch04 101 Ch05 100 || Ch39 100 Ch40 101 Ch41 109 ||
-- Final Tune Attempt 8 / 17:
--A0: Ch03 99 Ch04 129 Ch05 104 || Ch39 98 Ch40 101 Ch41 101 ||
--A1: Ch03 99 Ch04 98 Ch05 100 || Ch39 102 Ch40 104 Ch41 111 ||
-- Final Tune Attempt 9 / 17:
--A0: Ch03 90 Ch04 121 Ch05 93 || Ch39 102 Ch40 101 Ch41 101 ||
--A1: Ch03 101 Ch04 103 Ch05 99 || Ch39 101 Ch40 102 Ch41 112 ||
-- Final Tune Attempt 10 / 17:
--A0: Ch03 92 Ch04 122 Ch05 98 || Ch39 102 Ch40 101 Ch41 100 ||
--A1: Ch03 101 Ch04 100 Ch05 99 || Ch39 99 Ch40 103 Ch41 111 ||
-- Final Tune Attempt 11 / 17:
--A0: Ch03 101 Ch04 127 Ch05 108 || Ch39 99 Ch40 99 Ch41 101 ||
--A1: Ch03 101 Ch04 101 Ch05 102 || Ch39 98 Ch40 102 Ch41 111 ||
-- Final Tune Attempt 12 / 17:
--A0: Ch03 101 Ch04 120 Ch05 100 || Ch39 101 Ch40 102 Ch41 100 ||
--A1: Ch03 98 Ch04 100 Ch05 97 || Ch39 100 Ch40 103 Ch41 111 ||
-- Final Tune Attempt 13 / 17:
--A0: Ch03 101 Ch04 128 Ch05 103 || Ch39 98 Ch40 101 Ch41 102 ||
--A1: Ch03 102 Ch04 103 Ch05 100 || Ch39 100 Ch40 103 Ch41 113 ||
-- Final Tune Attempt 14 / 17:
--A0: Ch03 108 Ch04 133 Ch05 108 || Ch39 100 Ch40 100 Ch41 101 ||
--A1: Ch03 99 Ch04 95 Ch05 98 || Ch39 98 Ch40 104 Ch41 109 ||
-- Final Tune Attempt 15 / 17:
    
```

202 PedestalX

◇ The same as before



203 ToAX

- ◇ By inject known signal, calibration ToA threshold
- ◇ Target TOA = 50

meaning?

Base on ADC output

203 ToAX

Target ToA: 50

Template JSONs:

- /home/shyao/workspace/h2gcalibx/. /dump/202_PedestalCalibX_20260323_181539/asic_0_final_i2c_settings.json
- /home/shyao/workspace/h2gcalibx/. /dump/202_PedestalCalibX_20260323_181539/asic_1_final_i2c_settings.json

Channels injected in parallel: 8

Channels per ASIC to scan: 76

ASIC 0 ToA (0-90):

ASIC 1 ToA (35-90):

```
-- Inj_LowRange: DAC 60, Channels: 16, 52, 17, 53, 18, 54, 19, 55
-- Inj_LowRange: DAC 60, Channels: 20, 56, 21, 57, 22, 58, 23, 59
-- Inj_LowRange: DAC 60, Channels: 24, 60, 25, 61, 26, 62, 27, 63
-- Inj_LowRange: DAC 60, Channels: 28, 64, 29, 65, 30, 66, 31, 67
-- Inj_LowRange: DAC 60, Channels: 32, 68, 33, 69, 34, 70, 35, 71
-- Inj_LowRange: DAC 70, Channels: 00, 36, 01, 37, 02, 38, 03, 39
-- Inj_LowRange: DAC 70, Channels: 04, 40, 05, 41, 06, 42, 07, 43
-- Inj_LowRange: DAC 70, Channels: 08, 44, 09, 45, 10, 46, 11, 47
-- Inj_LowRange: DAC 70, Channels: 12, 48, 13, 49, 14, 50, 15, 51
-- Inj_LowRange: DAC 70, Channels: 16, 52, 17, 53, 18, 54, 19, 55
-- Inj_LowRange: DAC 70, Channels: 20, 56, 21, 57, 22, 58, 23, 59
-- Inj_LowRange: DAC 70, Channels: 24, 60, 25, 61, 26, 62, 27, 63
-- Inj_LowRange: DAC 70, Channels: 28, 64, 29, 65, 30, 66, 31, 67
-- Inj_LowRange: DAC 70, Channels: 32, 68, 33, 69, 34, 70, 35, 71
-- Inj_LowRange: DAC 80, Channels: 00, 36, 01, 37, 02, 38, 03, 39
-- Inj_LowRange: DAC 80, Channels: 04, 40, 05, 41, 06, 42, 07, 43
-- Inj_LowRange: DAC 80, Channels: 08, 44, 09, 45, 10, 46, 11, 47
-- Inj_LowRange: DAC 80, Channels: 12, 48, 13, 49, 14, 50, 15, 51
-- Inj_LowRange: DAC 80, Channels: 16, 52, 17, 53, 18, 54, 19, 55
-- Inj_LowRange: DAC 80, Channels: 20, 56, 21, 57, 22, 58, 23, 59
-- Inj_LowRange: DAC 80, Channels: 24, 60, 25, 61, 26, 62, 27, 63
-- Inj_LowRange: DAC 80, Channels: 28, 64, 29, 65, 30, 66, 31, 67
-- Inj_LowRange: DAC 80, Channels: 32, 68, 33, 69, 34, 70, 35, 71
-- Inj_LowRange: DAC 90, Channels: 00, 36, 01, 37, 02, 38, 03, 39
-- Inj_LowRange: DAC 90, Channels: 04, 40, 05, 41, 06, 42, 07, 43
-- Inj_LowRange: DAC 90, Channels: 08, 44, 09, 45, 10, 46, 11, 47
-- Inj_LowRange: DAC 90, Channels: 12, 48, 13, 49, 14, 50, 15, 51
-- Inj_LowRange: DAC 90, Channels: 16, 52, 17, 53, 18, 54, 19, 55
-- Inj_LowRange: DAC 90, Channels: 20, 56, 21, 57, 22, 58, 23, 59
-- Inj_LowRange: DAC 90, Channels: 24, 60, 25, 61, 26, 62, 27, 63
-- Inj_LowRange: DAC 90, Channels: 28, 64, 29, 65, 30, 66, 31, 67
-- Inj_LowRange: DAC 90, Channels: 32, 68, 33, 69, 34, 70, 35, 71
[ ] TOA Scan completed with exit code 0.
```

Start ToA Scan

Start Socket Pool

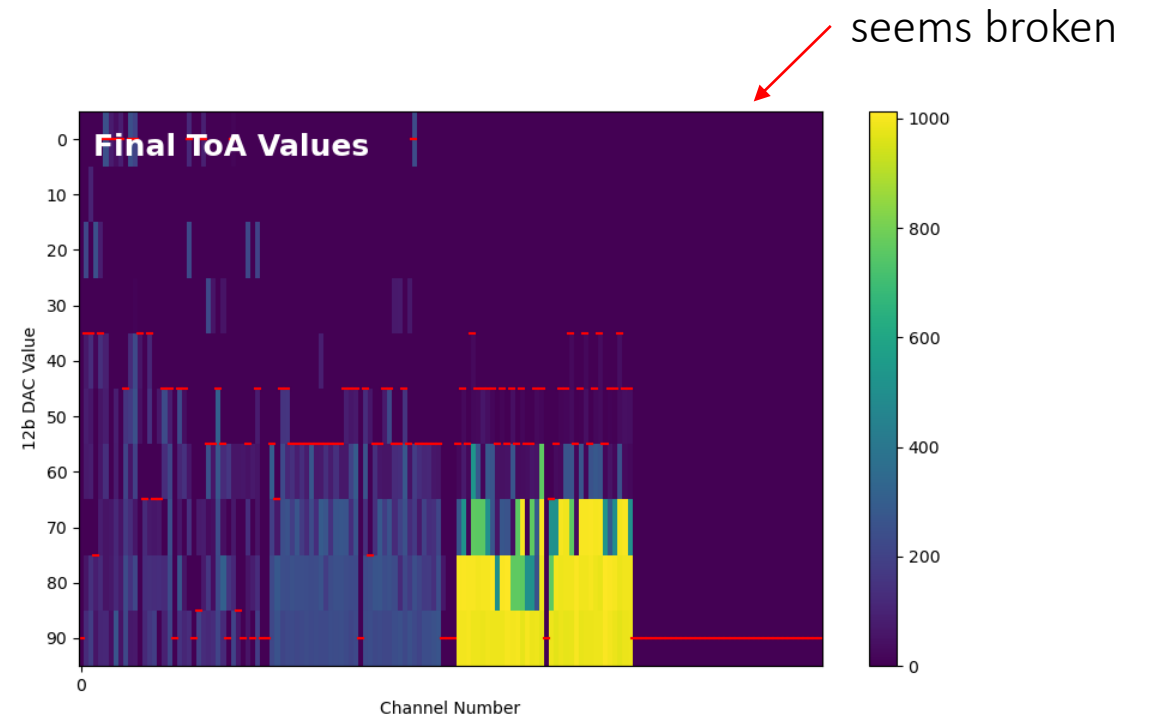
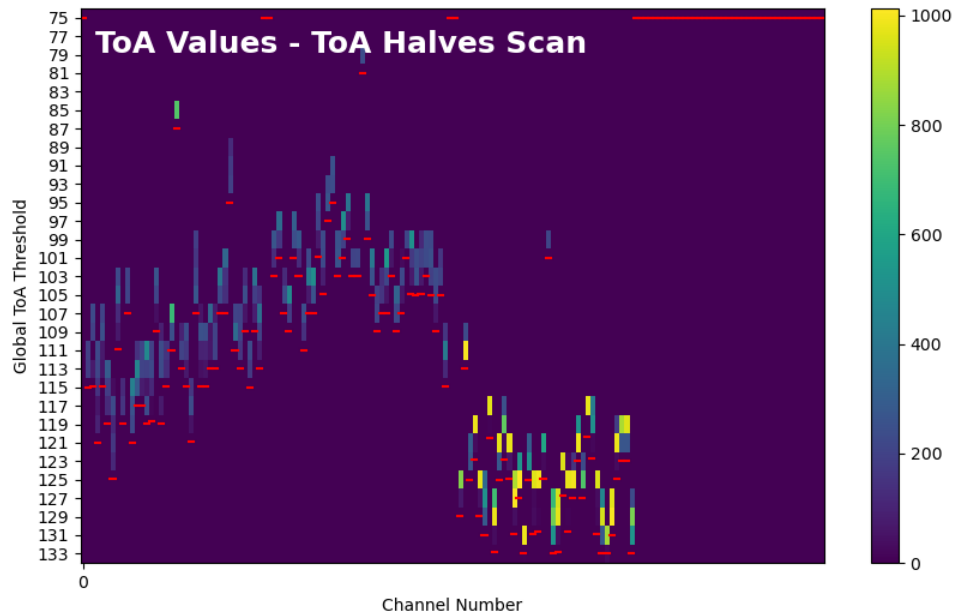
Close Socket Pool

```
[Pool] Bound UDP port 11001
[Pool] REGISTER bel13a22f-2a4d-42f6-bad5-dcdef3d7c880 -> ('data', 11001, '10.1.2.208')
[Pool] Closed UDP port 11000
[Pool] UNREGISTER bel13a22f-2a4d-42f6-bad5-dcdef3d7c880 -> ('cmd', 11000, '10.1.2.208')
[Pool] Closed UDP port 11001
[Pool] UNREGISTER bel13a22f-2a4d-42f6-bad5-dcdef3d7c880 -> ('data', 11001, '10.1.2.208')
```

0% --:--:--

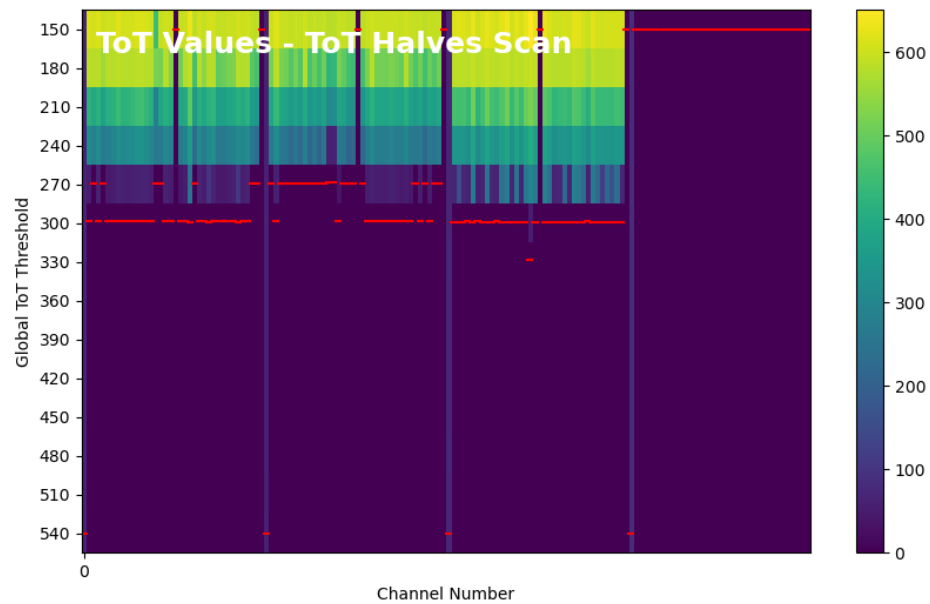
^n New FPGA tab ^w Close FPGA tab q Quit application ^p palette

203 ToAX

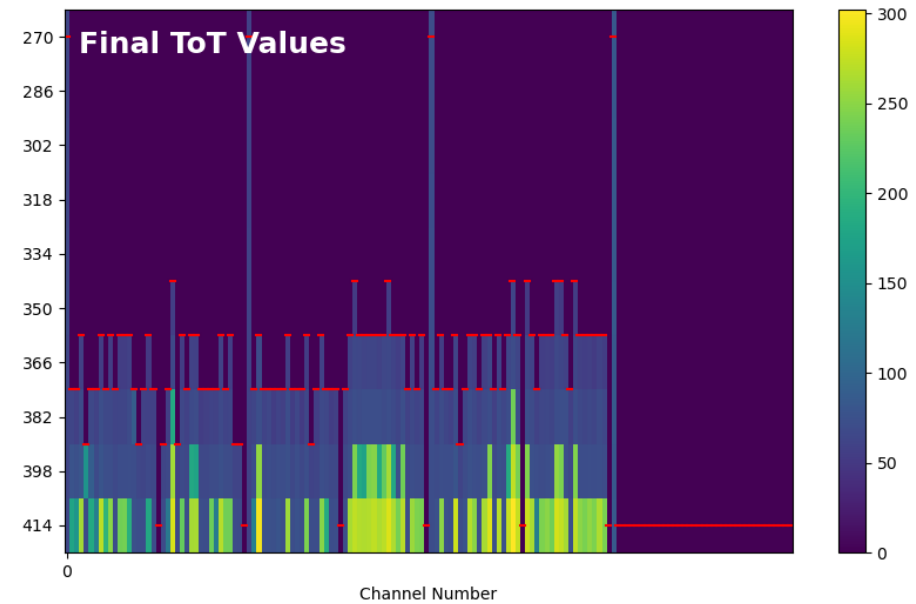


204 TOTX (302 TOT Omega)

◇ oops



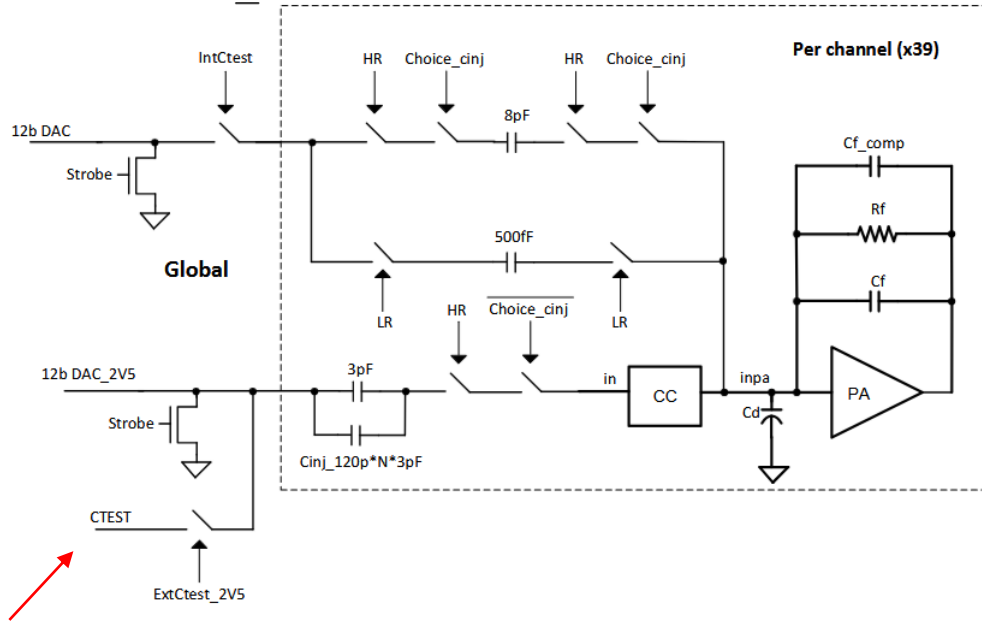
something wrong ?



?

205 Injection

- ◇ Two 12bit DAC for calibration
 - ◇ 12b DAC: 0 - around 1 V
 - ◇ DAC_2V5: 0 - 2.5V
- ◇ There is also a 12bit DAC output to PCB
 - ◇ DAC_ALDO: 0 - 2V



Use external signal to calibration all ASICs?

FPGA 1
CalibX
20:59:42

FPGA Settings
Registers
201 IODelayX
202 PedestalX
203 ToAX
204 ToTX
205 Injection
206 ScanDAC

Base on TOT
output

205 Injection

Injection DAC: Enter target injection DA

Enable 2.5V Injection Enable High Range Injection

Machine Gun: (0-19) Pre-Inj: (0-15)

Template JSONs:
/home/shyao/workspace/h2gcalibx/./dump/302_ToTCalib_Omega_20260323_193518/asic0_final_calib_i2c.json
/home/shyao/workspace/h2gcalibx/./dump/302_ToTCalib_Omega_20260323_193518/asic1_final_calib_i2c.json
Read from 204 Output

Channels injected in parallel: Number of channels

Channels per ASIC to scan: Number of channels

ASIC 0 Chn 39 (0-363):

ASIC 1 Chn 39 (0-361):

```

-- Inj_LowRange: DAC 2048, Channels: 00, 36, 01, 37, 02, 38, 03, 39
-- Inj_LowRange: DAC 2048, Channels: 04, 40, 05, 41, 06, 42, 07, 43
-- Inj_LowRange: DAC 2048, Channels: 08, 44, 09, 45, 10, 46, 11, 47
-- Inj_LowRange: DAC 2048, Channels: 12, 48, 13, 49, 14, 50, 15, 51
-- Inj_LowRange: DAC 2048, Channels: 16, 52, 17, 53, 18, 54, 19, 55
-- Inj_LowRange: DAC 2048, Channels: 20, 56, 21, 57, 22, 58, 23, 59
-- Inj_LowRange: DAC 2048, Channels: 24, 60, 25, 61, 26, 62, 27, 63
-- Inj_LowRange: DAC 2048, Channels: 28, 64, 29, 65, 30, 66, 31, 67
-- Inj_LowRange: DAC 2048, Channels: 32, 68, 33, 69, 34, 70, 35, 71
- Phase 13: Starting injection DAC 2048 scan...
-- Inj_LowRange: DAC 2048, Channels: 00, 36, 01, 37, 02, 38, 03, 39
-- Inj_LowRange: DAC 2048, Channels: 04, 40, 05, 41, 06, 42, 07, 43
-- Inj_LowRange: DAC 2048, Channels: 08, 44, 09, 45, 10, 46, 11, 47
-- Inj_LowRange: DAC 2048, Channels: 12, 48, 13, 49, 14, 50, 15, 51
-- Inj_LowRange: DAC 2048, Channels: 16, 52, 17, 53, 18, 54, 19, 55
-- Inj_LowRange: DAC 2048, Channels: 20, 56, 21, 57, 22, 58, 23, 59
-- Inj_LowRange: DAC 2048, Channels: 24, 60, 25, 61, 26, 62, 27, 63
-- Inj_LowRange: DAC 2048, Channels: 28, 64, 29, 65, 30, 66, 31, 67
-- Inj_LowRange: DAC 2048, Channels: 32, 68, 33, 69, 34, 70, 35, 71
- Phase 14: Starting injection DAC 2048 scan...
-- Inj_LowRange: DAC 2048, Channels: 00, 36, 01, 37, 02, 38, 03, 39
-- Inj_LowRange: DAC 2048, Channels: 04, 40, 05, 41, 06, 42, 07, 43
-- Inj_LowRange: DAC 2048, Channels: 08, 44, 09, 45, 10, 46, 11, 47
-- Inj_LowRange: DAC 2048, Channels: 12, 48, 13, 49, 14, 50, 15, 51
-- Inj_LowRange: DAC 2048, Channels: 16, 52, 17, 53, 18, 54, 19, 55
-- Inj_LowRange: DAC 2048, Channels: 20, 56, 21, 57, 22, 58, 23, 59
-- Inj_LowRange: DAC 2048, Channels: 24, 60, 25, 61, 26, 62, 27, 63
-- Inj_LowRange: DAC 2048, Channels: 28, 64, 29, 65, 30, 66, 31, 67
-- Inj_LowRange: DAC 2048, Channels: 32, 68, 33, 69, 34, 70, 35, 71
- Saved output files to /home/shyao/workspace/h2gcalibx/./dump/205_Injection_20260323_203005:
- Generating plots to
/home/shyao/workspace/h2gcalibx/./dump/205_Injection_20260323_203005/205_Injection_asic2_injdac2048_mg7_pack8_chn76_plots.pdf...
- Plot generation complete:
/home/shyao/workspace/h2gcalibx/./dump/205_Injection_20260323_203005/205_Injection_asic2_injdac2048_mg7_pack8_chn76_plots.pdf
 Injection Scan completed successfully.
                    
```

Start Injection Scan

Open Result PDF

Start Socket Pool

Close Socket Pool

[Pool] Bound UDP port 11001

[Pool] REGISTER f79472bd-2eb2-4cf8-8bf0-9a296097f906 -> ('data', 11001, '10.1.2.208')

[Pool] Closed UDP port 11000

[Pool] UNREGISTER f79472bd-2eb2-4cf8-8bf0-9a296097f906 -> ('cmd', 11000, '10.1.2.208')

[Pool] Closed UDP port 11001

[Pool] UNREGISTER f79472bd-2eb2-4cf8-8bf0-9a296097f906 -> ('data', 11001, '10.1.2.208')

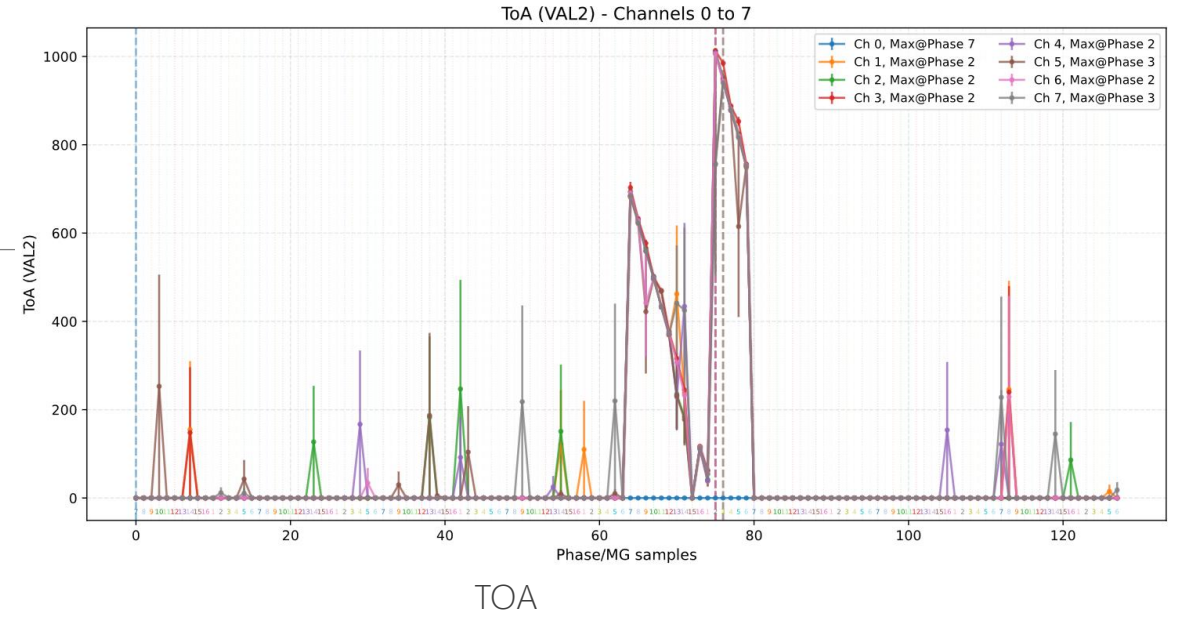
^n New FPGA tab ^w Close FPGA tab ^q Quit application

^p palette

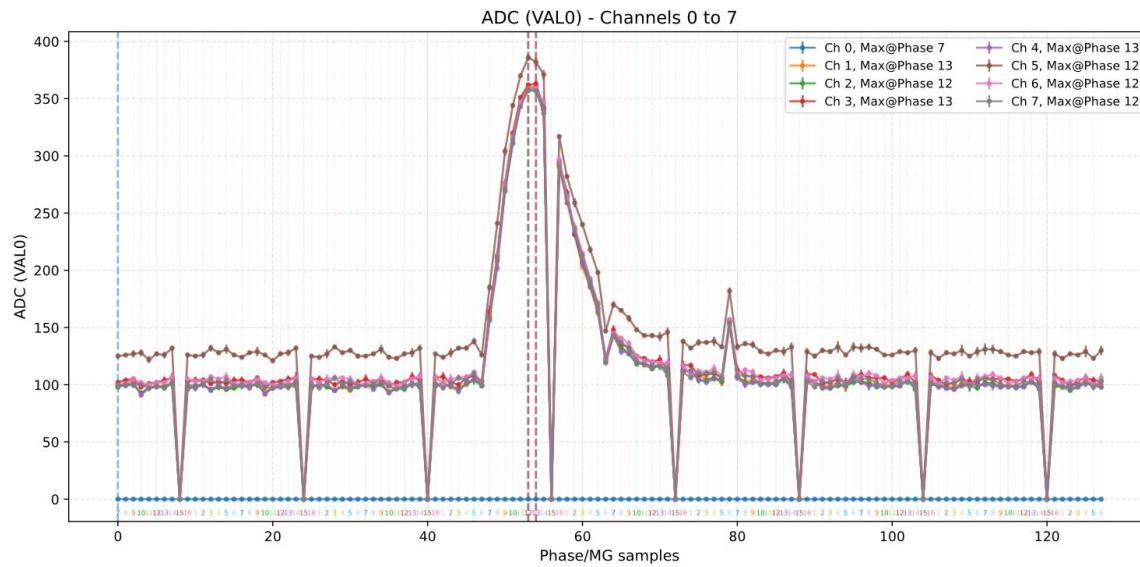
205 Injection

◇ Change phase

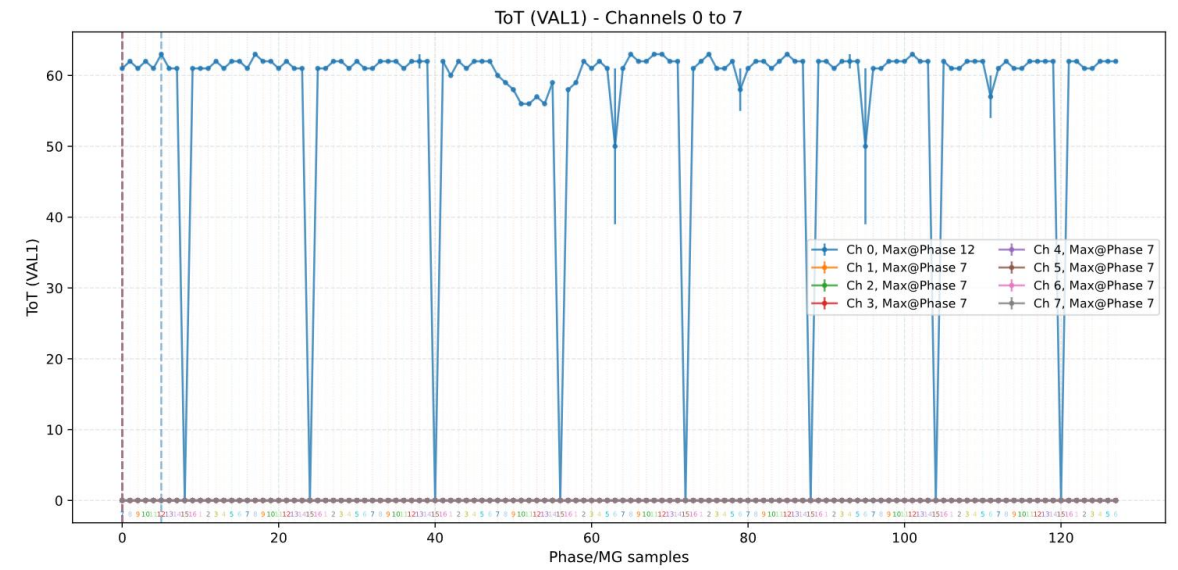
◇ Seem to broken



TOA



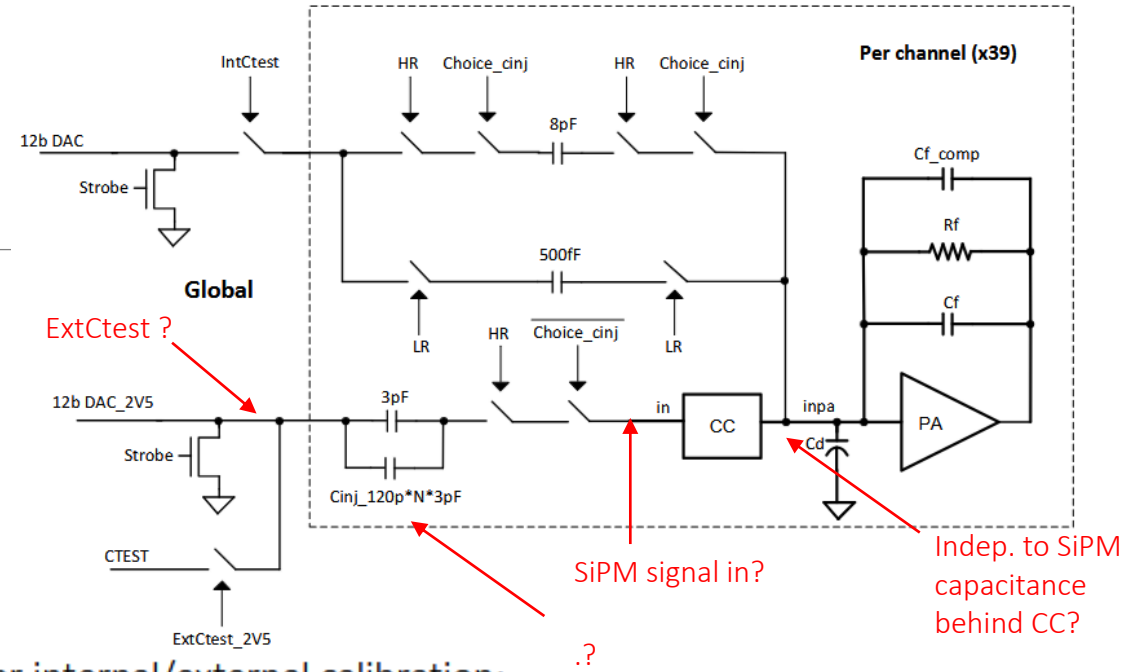
ADC



TOT

Register

- ◇ Channel wised register:
 - ◇ HR, LR, Cf, Cd, Rf, Cf_comp
- ◇ ½ H2GC register:
 - ◇ IntCtest, ExtCtest, ExtCtest_2V5, choice_cinj



The following tables show how to select the different configurations for internal/external calibration:

	DAC LR	DAC HR	CINJ 3pF	CINJ 3pF*N	CTEST 3pF	CTEST 3pF*N
HighRange	0	1	1	1	0	0
LowRange	1	0	0	0	0	0
Choice_cinj	X	1	0	0	0	0
Cinj_120p	0	0	0	1	0	1
IntCtest	1	1	0	0	0	0
ExtCtest_2V5	0	0	0	0	1	1
Calib INPA	0 - 500fC	0 - 8pC	-	-	-	-
Calib IN	-	-	0 - 3pC	0 - 3pC*N	-	-
Calib Ext Inj	-	-	-	-	CTEST * (0 - 3pC)	CTEST * (0 - 3pC*N)
Normal ?	0	0	0	0	0	0

206 ScanDAC

- ◇ Take long time, many hours
 - ◇ DAC 0 – 4095, Step = 1, phase = 0 -15
- ◇ not complete

The screenshot shows the CalibX FPGA 1 interface. The left sidebar lists 'FPGA Settings' with '206 ScanDAC' selected. The main panel is titled '206 DAC Scan' and contains the following configuration fields:

- DAC Min: 0 (0-4095)
- DAC Max: 4095 (0-4095)
- Step Size: 1 (1-4095)
- Phase: 0 (0-15)
- Machine Gun: 7 (0-19)
- Pre-Inj: 15 (0-15)
- Enable 2.5V Injection:
- Enable High Range Injection:

Below the configuration, there are fields for 'Channels injected in parallel' (set to 8) and 'Channels per ASIC to scan' (set to 76). There are also input fields for 'ASIC 0:' and 'ASIC 1:'. A 'Read from 204 Output' button is visible on the right.

The execution log shows the following output:

```
▶ Starting Injection Scan ...
▶ Running command: python3 -u ./206_InjectionDAC.py --ui --plot -a 2 -c
/home/shyao/workspace/h2gcalibx/config/common_settings_4_11_208.json --inj-dac-min 0 --inj-dac-max 4095 --inj-dac-step 1 --phase 0
--inj-mg 7 --inj-pre-int 15 -i
/home/shyao/workspace/h2gcalibx/./dump/302_ToTcalib_Omega_20260323_193518/asic0_final_calib_i2c.json,/home/shyao/workspace/h2gcalibx
/./dump/302_ToTcalib_Omega_20260323_193518/asic1_final_calib_i2c.json
-- 206_InjectionDAC (v1.0)-----
-----
- UDP from /home/shyao/workspace/h2gcalibx/config/common_settings_4_11_208.json:
-- PC IP: 10.1.2.207, Port: 11000/11001
-- Board IP: 10.1.2.208, Port: 11000
[clx_udp] INFO: Worker: 2b705b58-e18a-44d6-9eca-070309df79ed, Port: 41002
[clx_udp] INFO: DataCMD Port: 33428, DataDATA Port: 33438
- Loaded I2C settings from /home/shyao/workspace/h2gcalibx/./dump/302_ToTcalib_Omega_20260323_193518/asic0_final_calib_i2c.json for
ASIC 0.
- Loaded I2C settings from /home/shyao/workspace/h2gcalibx/./dump/302_ToTcalib_Omega_20260323_193518/asic1_final_calib_i2c.json for
ASIC 1.
- Setting I2C for ASIC 0...
- Setting I2C for ASIC 1...
-- Warning: Failed to set DAQ/Gen parameters.
-- Inj_LowRange: DAC 0, Channels: 00, 36, 01, 37, 02, 38, 03, 39
-- Inj_LowRange: DAC 0, Channels: 04, 40, 05, 41, 06, 42, 07, 43
-- Inj_LowRange: DAC 0, Channels: 08, 44, 09, 45, 10, 46, 11, 47
-- Inj_LowRange: DAC 0, Channels: 12, 48, 13, 49, 14, 50, 15, 51
-- Inj_LowRange: DAC 0, Channels: 16, 52, 17, 53, 18, 54, 19, 55
```

At the bottom, there are buttons for 'Stop Injection Scan', 'Start Socket Pool', and 'Close Socket Pool'. The status bar at the bottom shows '[Pool] DATA socket connected: 2b705b58-e18a-44d6-9eca-070309df79ed' and '[Pool] Control socket connected: 127.0.0.1:41002'.

software update?

- ◇ Asking the update of software about H2GDAQ, online monitor and decode

UDP packet

- ◇ Firmware V4.12

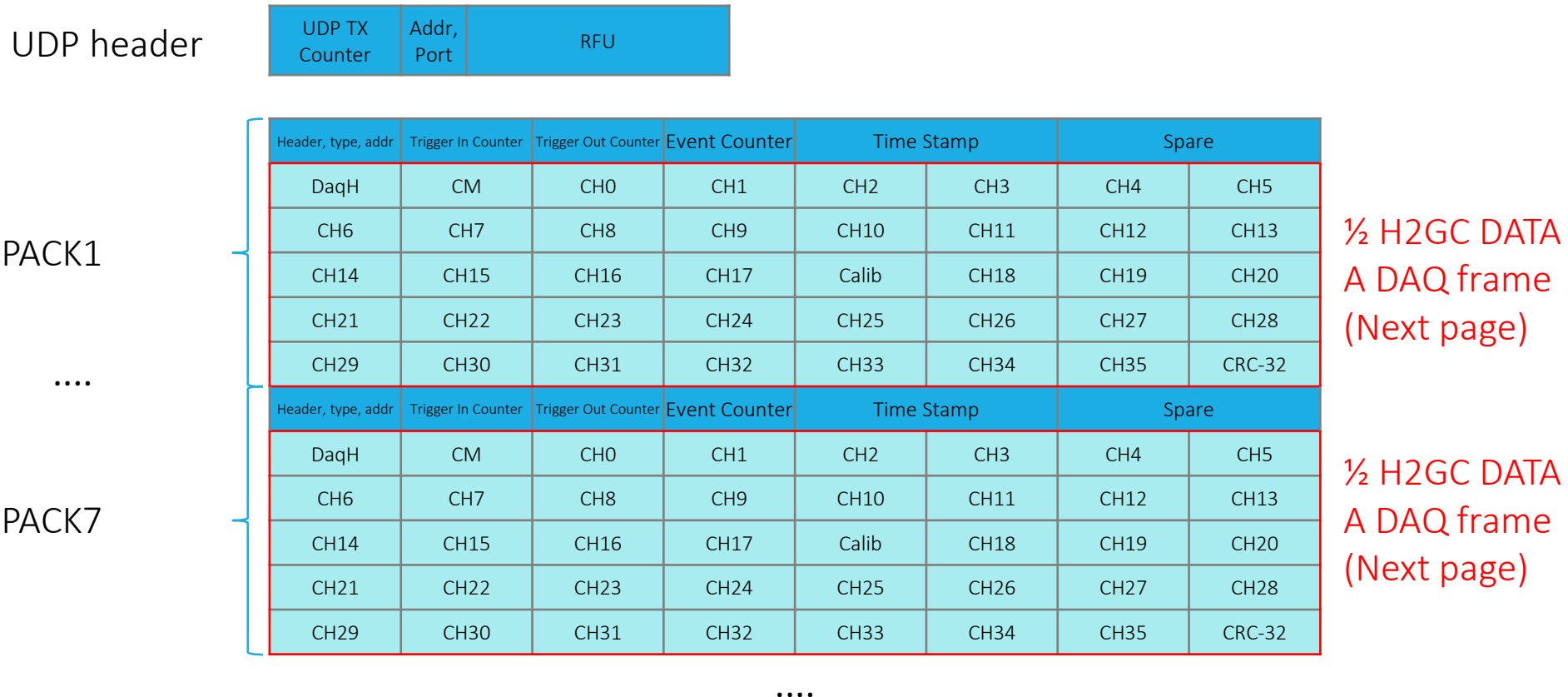
- ◇ Short packet = 40B = 0B (UDP header) + 40B (DATA)
- ◇ Long packet = 1452B = 12B (UDP header) + 40B (DATA) * 36
- ◇ Jumbo packet = 8972B = 12B (UDP header) + 40B (DATA) * 224

- ◇ Firmware V5.069

- ◇ Short packet = 46B = 6B (UDP header) + 40B (DATA)
- ◇ Long packet = 1358B = 6B (UDP header) + 8B (RFU) + 192B (DATA) * 7
- ◇ Jumbo packet = 8846B = 6B (UDP header) + 8B (RFU) + 192B (DATA) * 46

UDP Long packet format V5 (FPGA -> PC)

- ◇ One read command generate two ½ H2GC Data per ASIC, each H2GC Data is 160 Bytes
- ◇ A UDP Long packet contain:



Internal trigger V5

◇ No way was found to enable this feature from the user interface. Try using custom code.

◇ ADC

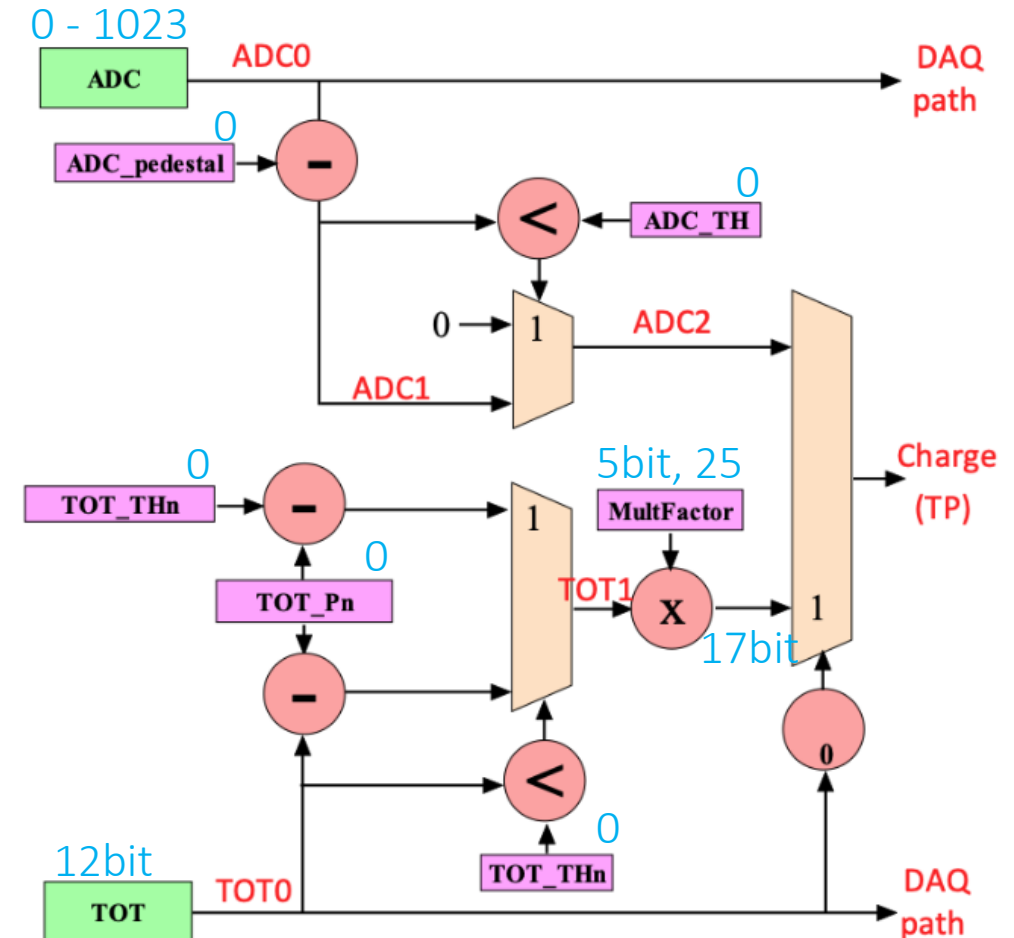
- ◇ ADC_pedestal: One per channel, 8bit, default = 0
- ◇ ADC_TH: One per ½ H2GC, 5bit, default = 0
 - ◇ $ADC2 = (ADC0 - ADC_pedestal > ADC_TH) ? ADC0 - ADC_pedestal : 0;$

◇ TOT

- ◇ TOT_THn: One per Group, 8bit, default = 0
- ◇ TOT_Pn: One per Group, 8bit, default = 0
- ◇ MultiFactor: One per ½ H2GC, 5bit, default = 25

◇ Two kind of Group

- ◇ TC4: 8 Groups each 4 channels, use TR0, TR1, TR2, TR3
- ◇ TC9: 4 Groups each 9 channels, use TR0, TR2



Internal trigger V5

◇ Huh

Bits	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
											ADCO	0	0	0	0	0	0	0	0	0	0	0	
											ADC Pedestal			0	0	0	0	0	0	0	0	0	
											ADC Threshold						0	0	0	0	0	0	
											TOTO	0	0	0	0	0	0	0	0	0	0	0	0
											TOT Pedestal			0	0	0	0	0	0	0	0	0	
											MultiFactor						1	1	0	0	1		



TC4:

ADC2	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOT2	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

TC9:

TOT2	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
------	----	----	----	----	----	----	----	----	----	----	----	---	---	---	---	---	---	---	---	---	---

Raw Data V5

- ◇ A custom program can send start_daq and store UDP packet
- ◇ red boxed fields require byte swapping, i.e. Big-endian

```
[shyao@cats myprj]$ od -Ax -t x1 -j 0 -N14 data/run1.bin
000000 00 1c 49 af d0 f9 ee 00 00 00 00 00 00 00 00
```

UDP Header

```
[shyao@cats myprj]$ od -Ax -t x4 -j 14 -w32 -N1358 data/run1.bin
```

```
00000e 24015aaa 01000000 04000000 00000000 00000000 40000000 00000000 00000000
00002e 050681f1 00c40100 00004009 0000d003 00004007 0000b005 00001008 0000a005
00004e 00009006 00001008 0000e008 00005008 00000008 00005008 0000c007 00004008
00006e 00005009 0000f007 00001009 00006008 00008005 0000f008 00005007 0000b007
00008e 00001006 00002008 00004008 00009008 0000a006 00009007 00003007 00009007
0000ae 00009008 00002007 0000d007 00001008 00001007 00003007 00003006 d4c6bef7
0000ce 24005aaa 01000000 04000000 00000000 00000000 40000000 00000000 00000000
0000ee 8505a9f9 00300100 00001006 00003006 0000e008 0000d007 0000400a 0000c005
00010e 00001006 00005006 00009005 00008007 0000c003 0000e007 00006004 00009007
00012e 00009004 00004006 00004005 00006005 00003005 00003003 00008004 00002003
00014e 00001003 00005004 00002005 00001006 00005005 0000d003 0000e004 0000b005
00016e 00002006 00004007 00003006 0000a006 0000c005 00009005 00004004 3f196ed5
00018e 25015aaa 01000000 04000000 00000000 00000000 40000000 00000000 00000000
0001ae 050681f1 00740100 0000c006 00005005 00008003 0000e005 0000c005 00000007
0001ce 0000a005 00006005 0000c006 00002005 0000a006 0000f005 00005005 00004004
0001ee 0000c004 00001008 00006006 00002006 0000e006 00004004 00004007 00009005
00020e 00009007 00006006 00002004 0000d005 0000f005 0000e005 00004004 0000c003
00022e 0000e005 00001009 0000e007 0000d001 0000d006 00002003 00001006 81bd3bf4
00024e 25005aaa 01000000 04000000 00000000 00000000 40000000 00000000 00000000
00026e 8505a9f9 00700100 0000b006 0000c009 00001009 00008006 00007007 0000a007
00028e 00008006 00002007 0000d005 00009006 0000b007 00001006 00001006 0000a007
0002ae 00001008 0000d005 00009004 00009008 00007005 00005007 0000e007 0000c007
0002ce 0000000a 0000f006 00008006 00000009 0000f006 00006005 0000a009 00004007
0002ee 00006007 00008008 00001006 0000f007 00004007 0000f008 00005005 0984db82
00030e 24005aaa 01000000 04000000 01000000 00000000 e4000000 00000000 00000000
00032e 8509aaf9 002c0100 00001006 00001006 00000009 0000a007 0000500a 0000c005
00034e 00001006 00006006 0000a005 00007007 0000c003 0000d007 00008004 0000a007
00036e 0000d004 00004006 00006005 00004005 00005005 00003003 00007004 00002003
00038e 00002003 00005004 00001005 00002006 00005005 0000d003 0000f004 0000a005
0003ae 00002006 00005007 00004006 0000c006 0000d005 0000a005 00005004 15627d3d
0003ce 24015aaa 01000000 04000000 01000000 00000000 e4000000 00000000 00000000
0003ee 050a82f1 00c40100 00004009 00000004 00002007 00009005 00002008 0000c005
00040e 0000a006 00004008 0000e008 00005008 0000f007 00005008 0000d007 00002008
00042e 00005009 00000008 00002009 00008008 00008005 0000e008 00004007 0000c007
00044e 00000006 00001008 00005008 00009008 0000a006 0000a007 00001007 00009007
00046e 00009008 00001007 0000c007 00002008 00002007 00002007 00005006 9f4976e0
00048e 25005aaa 01000000 04000000 01000000 00000000 e4000000 00000000 00000000
0004ae 8509aaf9 006c0100 00009006 0000b009 00000009 00009006 0000c007 00009007
0004ce 00008006 00002007 0000d005 00009006 0000e007 00001006 00001006 0000c007
0004ee 00005008 0000d005 00009004 00009008 00009005 00003007 0000f007 0000c007
00050e 0000e009 00000007 00008006 00001009 00002007 00006005 00008009 00002007
```

Decode V5

```
[shyao@cats myprj]$ ./build/debug/ana/ana decode data/run1.bin
Open rawdata file data/run1.bin
udp_tx_counter: 1853871 fpga_ip_address: 208 fpga_send_port: 249 RFU: ee 00 00 00 00 00 00
Address: 01 header: aa5a packet_type: 24 Trigger_IN_Counter: 1 Trigger_OUT_Counter: 4 Event_Counter: 0 timestamp: 64
hd: 15 H1: 0 H2: 0 H3: 0 tr: 5 BxCounter: 385 Event: 1 Orbit: 4 BX_NEW timestamp diff: 64
f1810605 0001c400 09400000 03d00000 07400000 05b00000 08100000 05a00000
06900000 08100000 08e00000 08500000 08000000 08500000 07c00000 08400000
09500000 07f00000 09100000 08600000 05800000 08f00000 07500000 07b00000
06100000 08200000 08400000 08900000 06a00000 07900000 07300000 07900000
08900000 07200000 07d00000 08100000 07100000 07300000 06300000 f7bec6d4
Address: 00 header: aa5a packet_type: 24 Trigger_IN_Counter: 1 Trigger_OUT_Counter: 4 Event_Counter: 0 timestamp: 64
hd: 15 H1: 0 H2: 0 H3: 0 tr: 5 BxCounter: 2473 Event: 1 Orbit: 3 BX_NEW timestamp diff: 64
f9a90585 00013000 06100000 06300000 08e00000 07d00000 0a400000 05c00000
06100000 06500000 05900000 07800000 03c00000 07e00000 04600000 07900000
04900000 06400000 05400000 05600000 05300000 03300000 04800000 03200000
03100000 04500000 05200000 06100000 05500000 03d00000 04e00000 05b00000
06200000 07400000 06300000 06a00000 05c00000 05900000 04400000 d56e193f
Address: 01 header: aa5a packet_type: 25 Trigger_IN_Counter: 1 Trigger_OUT_Counter: 4 Event_Counter: 0 timestamp: 64
hd: 15 H1: 0 H2: 0 H3: 0 tr: 5 BxCounter: 385 Event: 1 Orbit: 4 BX_NEW
f1810605 00017400 06c00000 05500000 03800000 05e00000 05c00000 07000000
05a00000 05600000 06c00000 05200000 06a00000 05f00000 05500000 04400000
04c00000 08100000 06600000 06200000 06e00000 04400000 07400000 05900000
07900000 06600000 04200000 05d00000 05f00000 05e00000 04400000 03c00000
05e00000 09100000 07e00000 01d00000 06d00000 03200000 06100000 f43bbd81
Address: 00 header: aa5a packet_type: 25 Trigger_IN_Counter: 1 Trigger_OUT_Counter: 4 Event_Counter: 0 timestamp: 64
hd: 15 H1: 0 H2: 0 H3: 0 tr: 5 BxCounter: 2473 Event: 1 Orbit: 3 BX_NEW
f9a90585 00017000 06b00000 09c00000 09100000 06800000 07700000 07a00000
06800000 07200000 05d00000 06900000 07b00000 06100000 06100000 07a00000
08100000 05d00000 04900000 08900000 05700000 07500000 07e00000 07c00000
0a000000 06f00000 06800000 09000000 06f00000 05600000 09a00000 07400000
07600000 08800000 06100000 07f00000 07400000 08f00000 05500000 82db8409
```

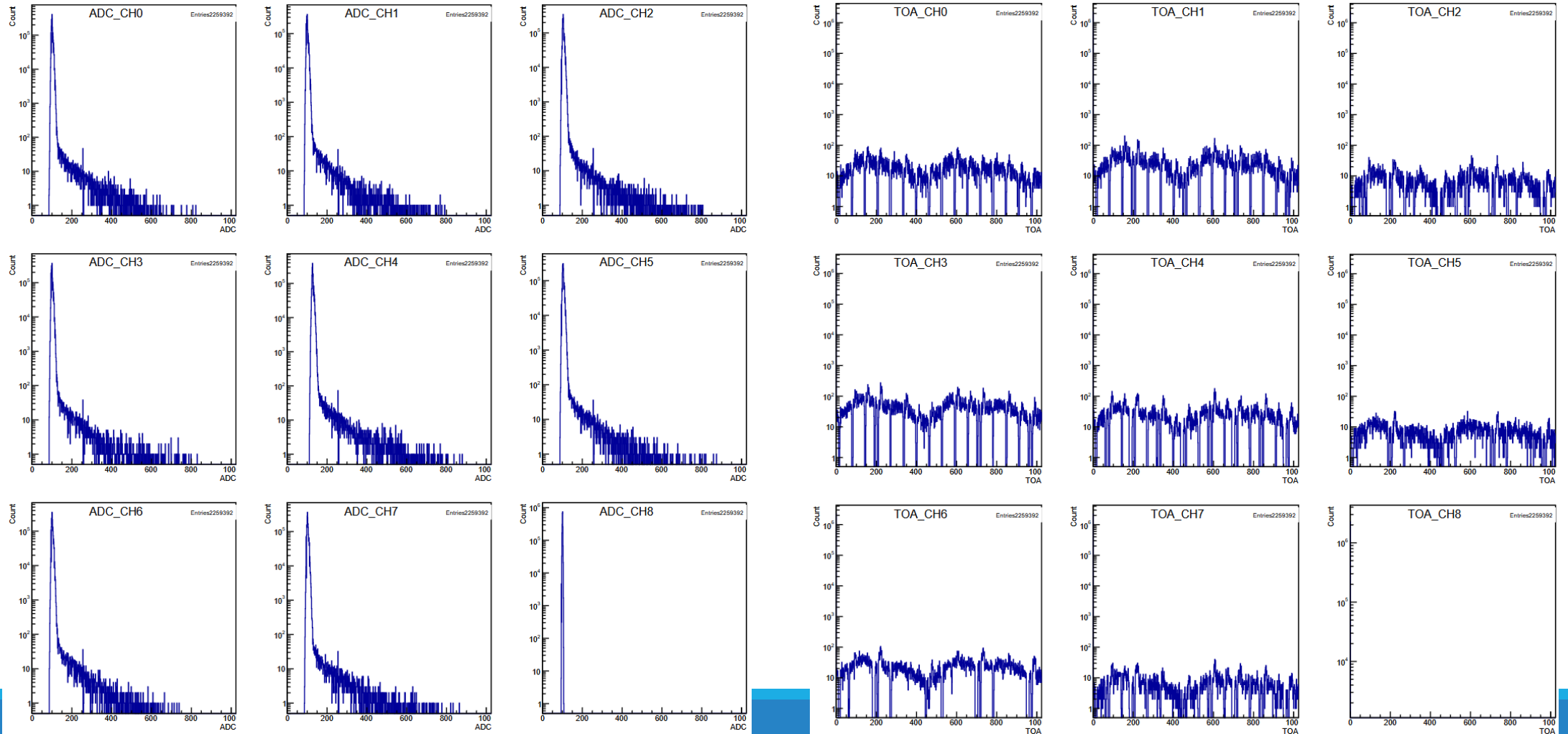
DAQ

- ◇ Custom...
- ◇ Generator = 100000, period = 40000, Machine Gun = 31, Voltage = 28V

```
*****
*Tree   :events   : A Tree with Events                                     *
*Entries :   90432 : Total =      2501195088 bytes File Size =  273825839 *
*       :         : Tree compression factor =    9.13                       *
*****
*Br    0 :toa     : toa[144][32]/s                                         *
*Entries :   90432 : Total Size=  833731575 bytes File Size =   14359615 *
*Baskets :    3227 : Basket Size=   3200512 bytes Compression=   58.05     *
*.....*
*Br    1 :tot     : tot[144][32]/s                                         *
*Entries :   90432 : Total Size=  833731575 bytes File Size =    3972140 *
*Baskets :    3227 : Basket Size=   3200512 bytes Compression=  209.86     *
*.....*
*Br    2 :adc     : adc[144][32]/s                                         *
*Entries :   90432 : Total Size=  833731575 bytes File Size =  255423704 *
*Baskets :    3227 : Basket Size=   3200512 bytes Compression=    3.26     *
*.....*
total 90432 entries
```

Analysis

◇ After convert to root file, analysis is the same



To be continue...
