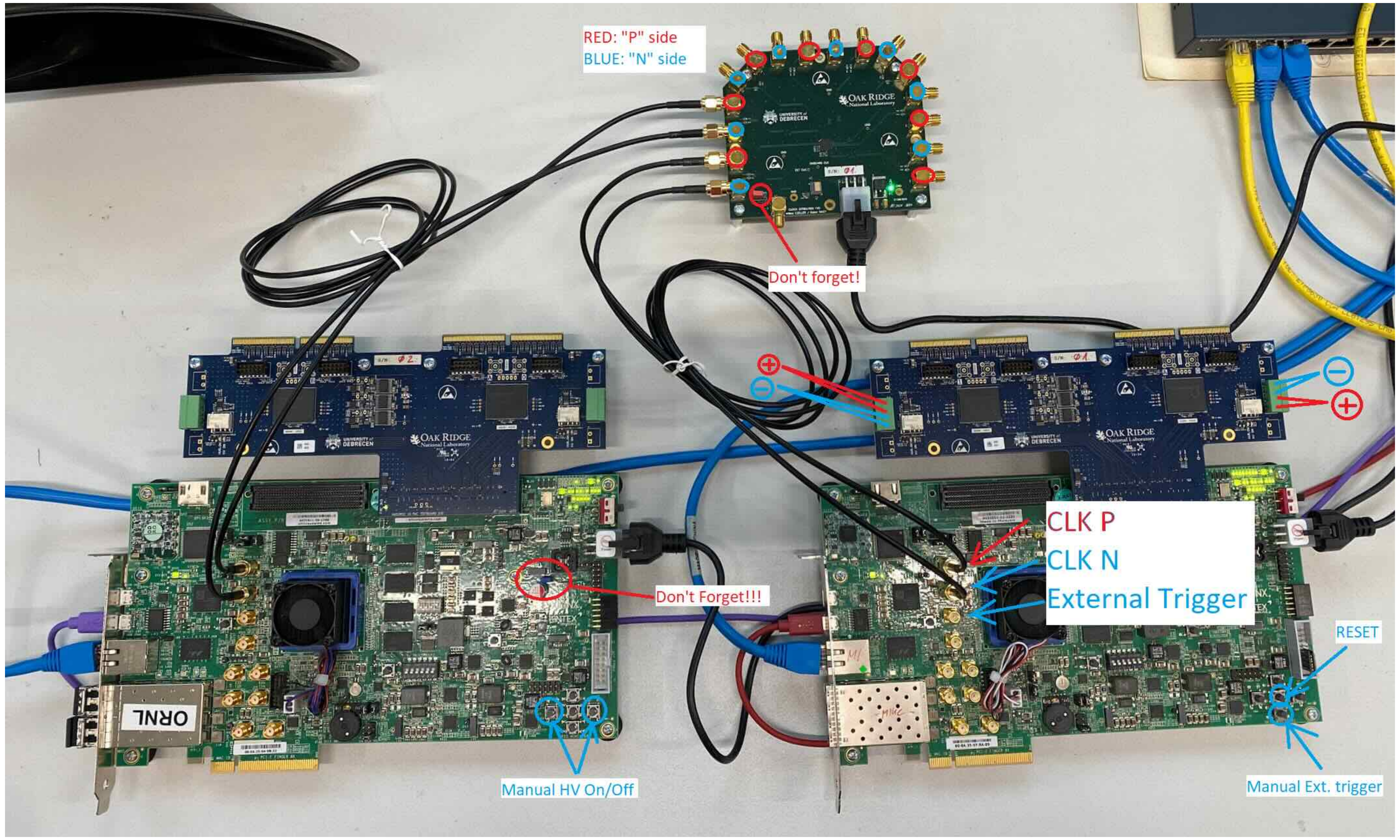
Quick recap on H2GCROC

ORNL is managed by UT-Battelle LLC for the US Department of Energy

ProtoBoard2.0 – second iteration with the H2GCROC

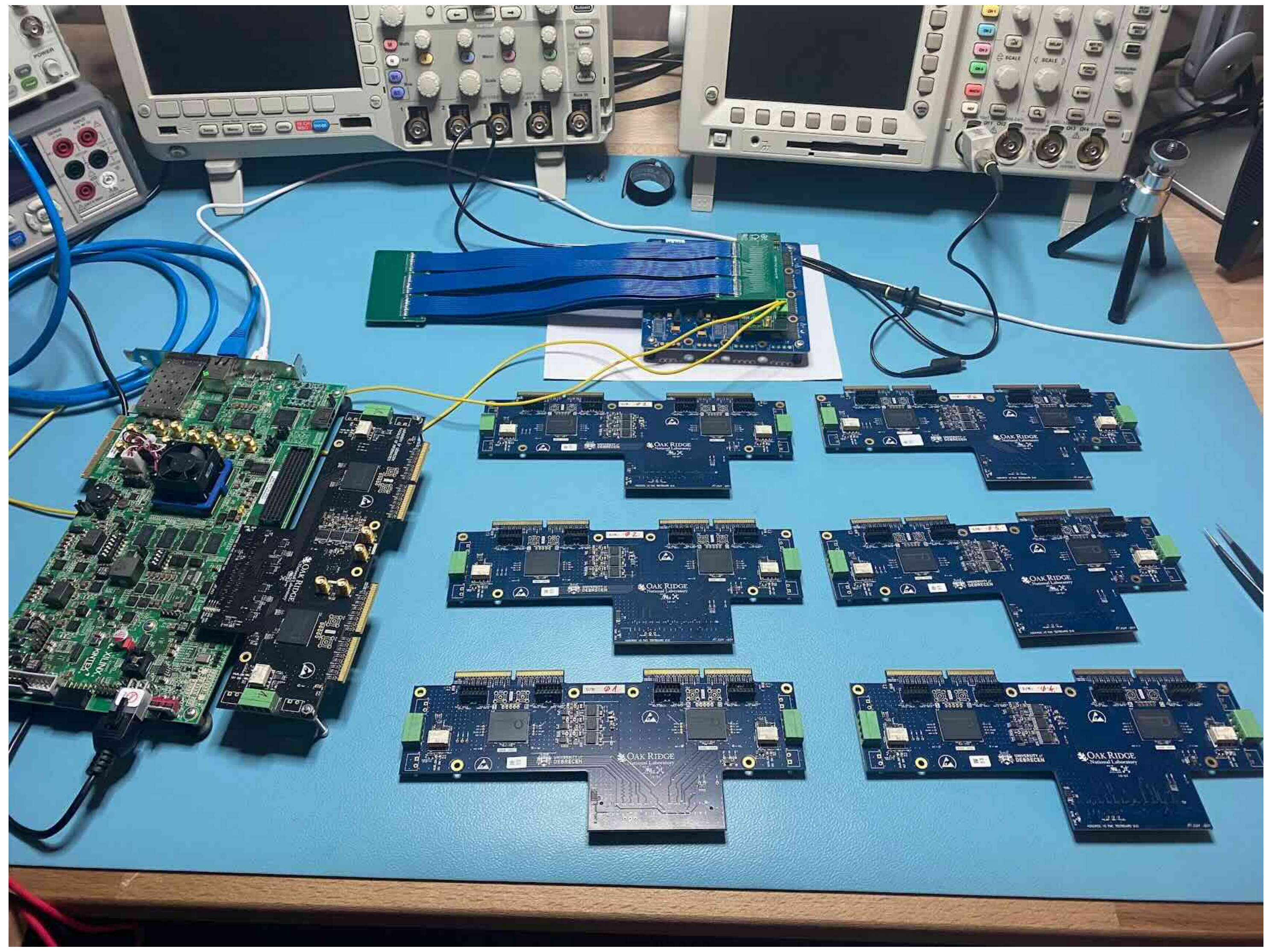


Compatible design with the commercial CAEN unit:

- Ease of testing with many collaborators:
  - CAEN has slow readout, capable of 10kHz only in reality

Started with the commercial KCU105 FPGA evaluation board:

- Multiple KCU's are possible to combine with single clock and trigger distribution boards

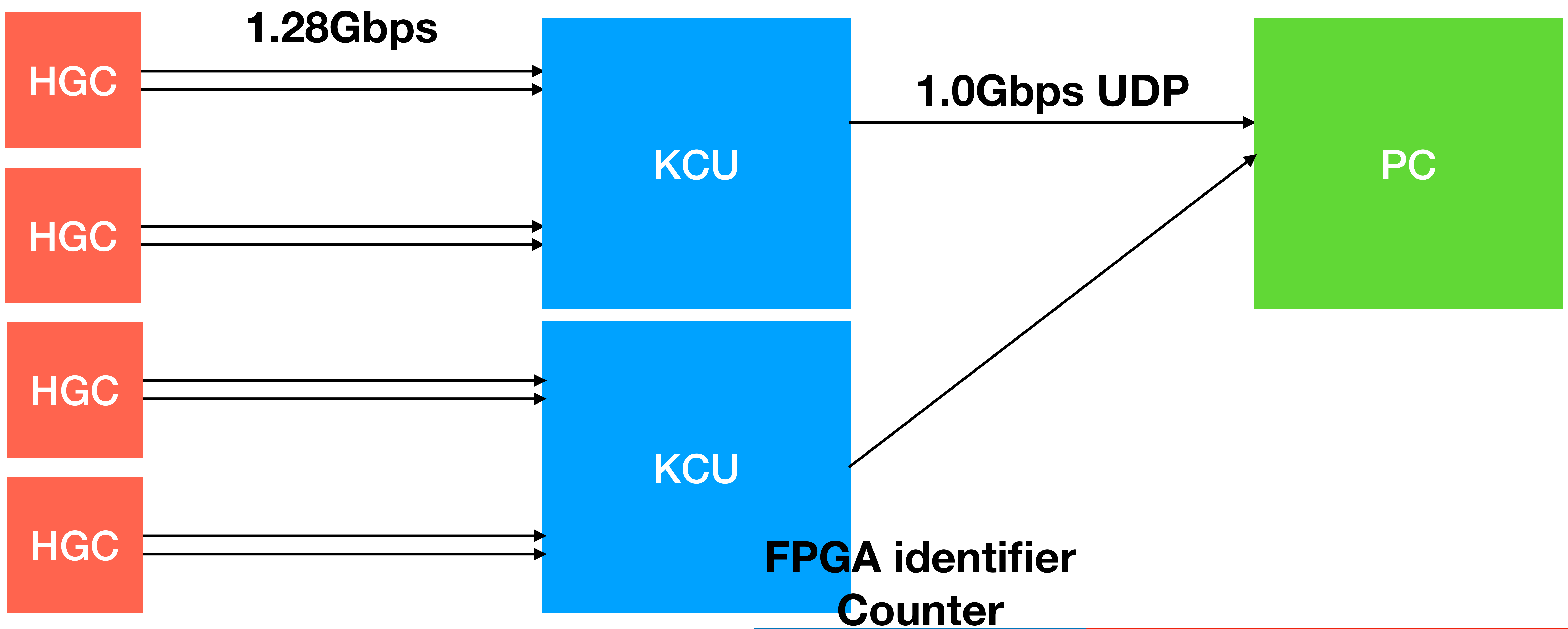


Produced multiple boards total 864 channels

Synergy also with other detectors using HXGCROC (same firmware)

Oak Ridge National Laboratory

Bottle neck - for now



Buffering in:

- HGCROC (32 deep)
  - Too many samples would create Hamming errors
- FPGA (few samples only)

Strategy:

It takes ~1µs to readout one sample from HGCROC

Not to overwhelm the FPGA buffer (implement dead time)

FPGA identifier

Counter

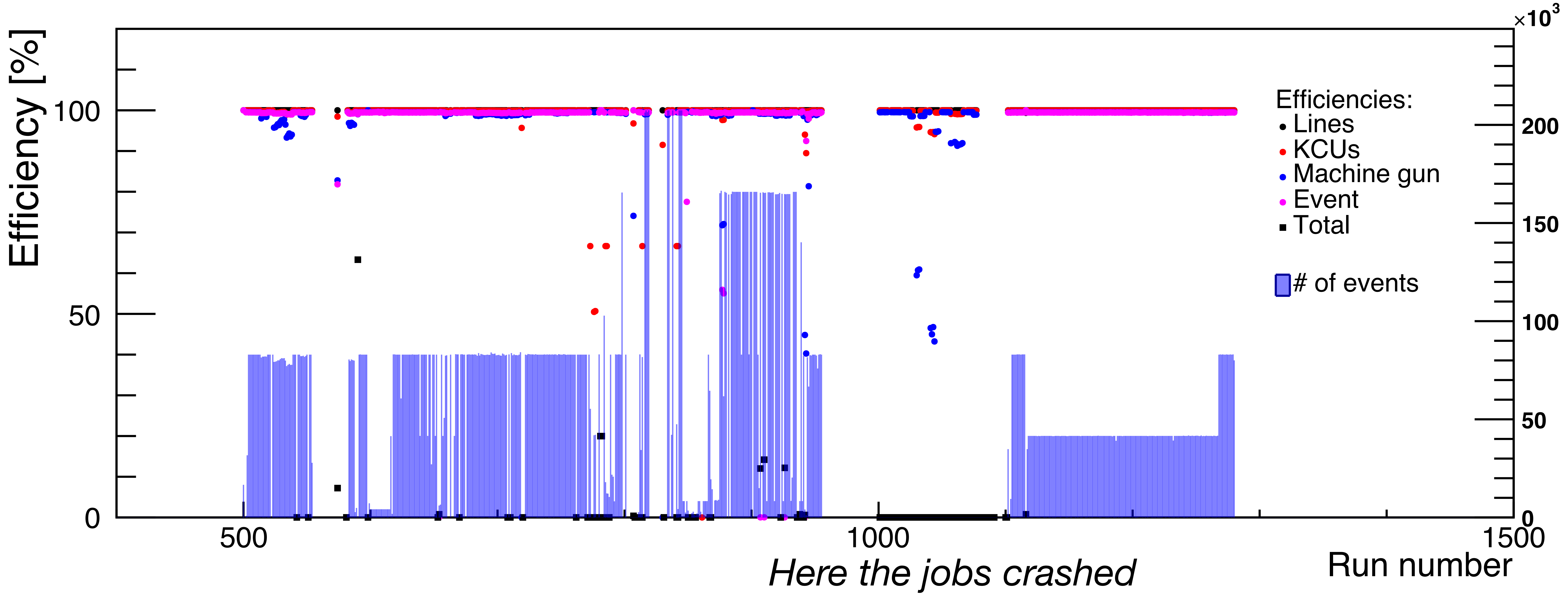
HGCROC data

```
a0 01 24 00 0c 71 20 ec 59 67 a6 85 00 00 00 00 04 e0 00 00 05 00 00 00 05 50 00 00 05 40 00 00 03 c0 00 00 05 20 00 00
a0 01 24 01 0c 71 20 ec 05 00 02 05 05 30 00 00 05 80 00 00 04 70 00 00 04 70 00 00 04 e0 00 00 05 30 00 00 05 10 03 c7
a0 01 24 02 0c 71 20 ec 05 40 00 00 05 00 03 b6 05 50 00 00 05 30 03 da 03 10 00 00 05 30 00 00 05 50 00 00 04 e0 00 00
a0 01 24 03 0c 71 20 ec 04 f0 00 00 05 20 00 00 05 20 00 00 05 00 00 00 05 10 03 78 05 50 00 00 05 10 00 00 05 10 02 17
a0 01 24 04 0c 71 20 ec 05 60 00 00 05 30 00 00 05 00 00 00 05 30 00 00 05 10 00 00 05 20 00 00 01 f0 00 00 d3 a8 e8 eb
a1 01 24 00 0c 71 20 ec 59 96 a7 05 00 00 00 00 05 50 00 00 05 20 00 00 05 50 00 00 04 80 00 00 05 40 00 00 05 10 00 00
a1 01 24 01 0c 71 20 ec 05 20 00 00 05 00 00 00 05 30 00 00 05 20 00 00 05 10 00 00 05 10 00 00 05 40 00 00 05 10 00 00
a1 01 24 02 0c 71 20 ec 05 30 00 00 05 00 00 00 05 40 00 00 05 00 00 00 03 70 00 00 05 10 00 00 05 20 00 00 05 10 00 00
a1 01 24 03 0c 71 20 ec 04 d0 00 00 05 50 00 00 05 40 00 00 05 30 00 00 05 60 00 00 05 40 00 00 05 20 00 00 05 00 00 00
a1 01 24 04 0c 71 20 ec 05 20 00 00 05 30 00 00 05 30 00 00 05 60 00 00 05 00 00 00 05 50 00 00 03 f0 00 00 73 e3 39 57
a0 01 25 00 0c 71 20 ec 59 67 a6 85 00 00 00 00 05 50 00 00 04 f0 00 00 05 80 00 00 05 60 00 00 05 30 00 00 05 40 01 6d
a0 01 25 01 0c 71 20 ec 05 30 00 00 05 80 00 00 05 60 00 00 05 30 00 00 05 40 00 00 05 10 00 00 05 40 00 00 05 50 00 00
a0 01 25 02 0c 71 20 ec 05 10 00 00 05 30 00 00 05 20 00 00 05 40 00 00 02 60 00 00 05 30 00 00 05 70 00 00 05 10 01 fb
a0 01 25 03 0c 71 20 ec 05 00 00 00 05 50 00 00 04 10 00 00 05 60 00 00 05 30 00 00 05 30 00 00 05 40 00 00 05 50 00 00
a0 01 25 04 0c 71 20 ec 05 20 00 00 05 70 00 00 05 40 00 00 05 50 00 00 05 70 00 00 05 60 00 00 02 20 00 00 bc 34 76 f6
a1 01 25 00 0c 71 20 ec 59 96 a7 05 00 00 00 00 05 20 00 00 04 80 00 00 05 60 00 00 05 40 00 00 05 20 00 00 05 20 00 00
a1 01 25 01 0c 71 20 ec 05 40 00 68 04 60 00 00 05 00 00 00 05 00 00 00 05 30 00 00 05 00 00 00 05 10 00 00 05 40 00 00
a1 01 25 02 0c 71 20 ec 05 50 00 00 04 e0 00 00 05 30 00 00 04 f0 00 00 03 70 00 00 05 10 00 00 05 80 00 00 04 f0 00 00
a1 01 25 03 0c 71 20 ec 04 f0 00 00 05 60 00 00 05 50 00 00 05 20 00 00 05 30 00 00 04 f0 00 00 05 40 00 00 05 40 00 00
a1 01 25 04 0c 71 20 ec 00 10 00 00 05 d0 00 00 03 50 00 00 05 60 00 00 05 70 01 d9 05 30 00 00 02 10 00 00 f9 e5 e5 d4
```

The FPGA adds 2x32 bit words:

- ID to check in which line the data was coming in, plus line number (32-bit)
- Counter added when the data is received (32-bit counter)

Stitching together events



Line reconstruction
- find 5 lines per KCU

KCU:
- Reconstruct the full KCU input (20 lines)

Machine gun:
- Collect all machine guns (41-counter difference)

Event:
- Stitch together the two (or more) KCU's

Total:
- From number of lines in file to reconstructed events

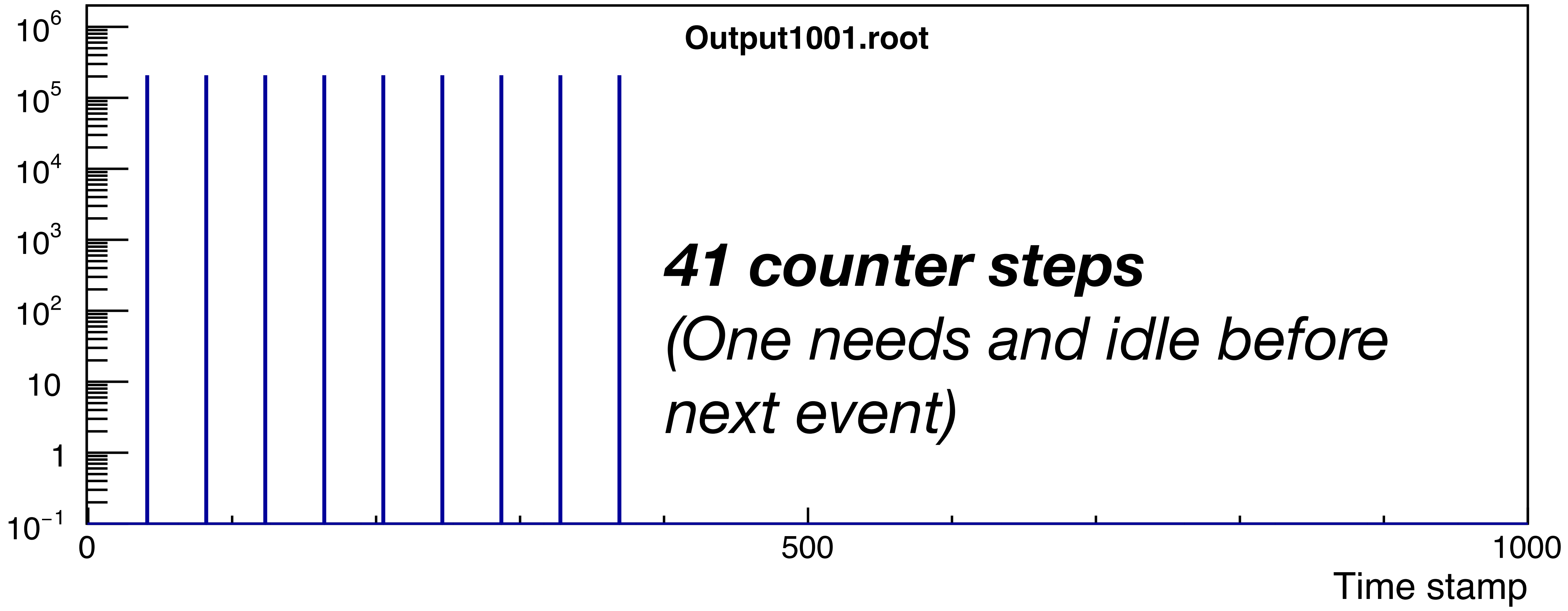
Collected 1.1TB of data in total, 700 runs in total now:
- Total of 150M events
  - Does not include the calibration runs
- Different runs with different gains (changing the gain conveyer settings)
  - Here calibration has to be made more quick as every settings change requires new calibration run

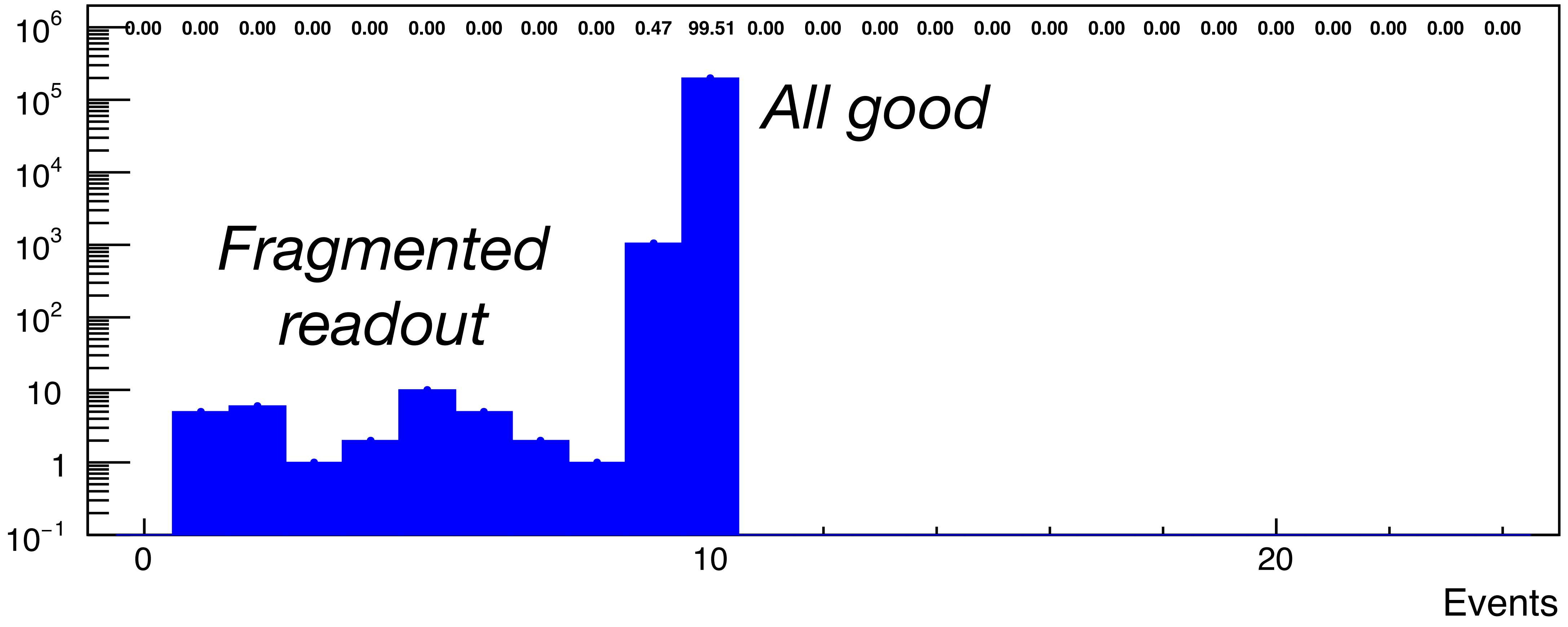
Total loss now (excluding the crashed jobs), defined when total efficiency is <90%, is 1.35% of the events

Oak Ridge National Laboratory

Some more details

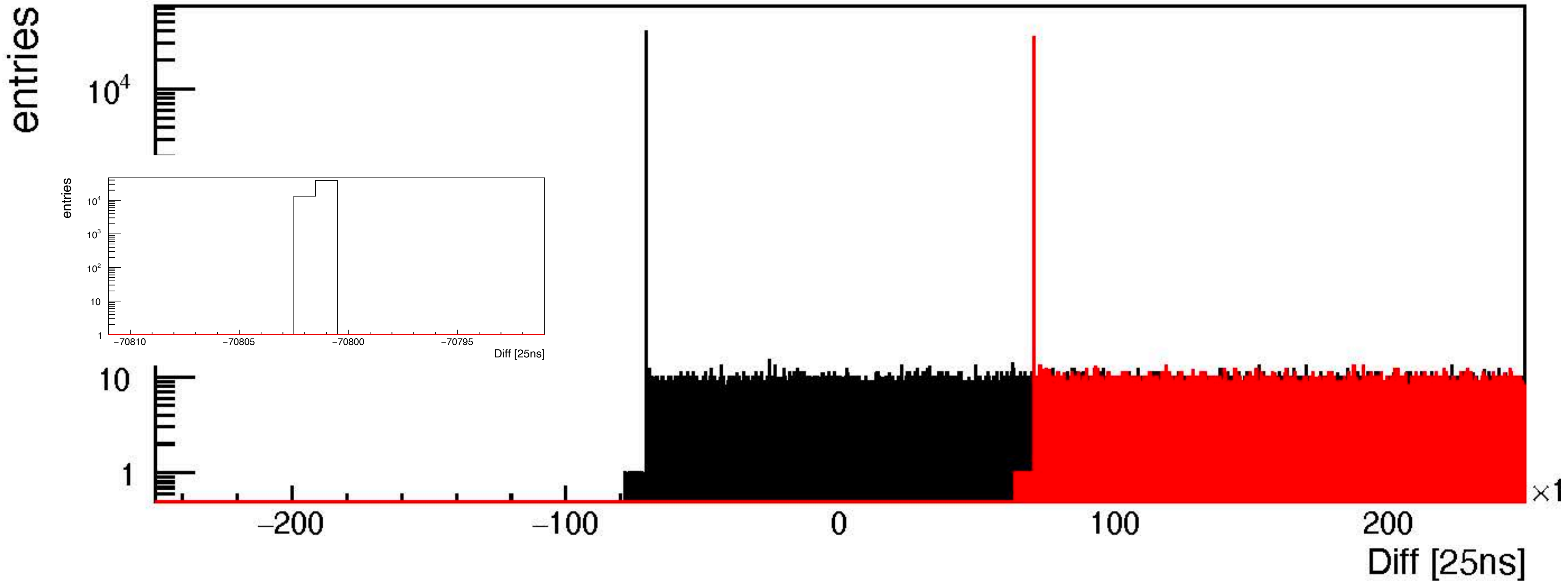
Putting one KCU together - 10 consecutive counters are needed

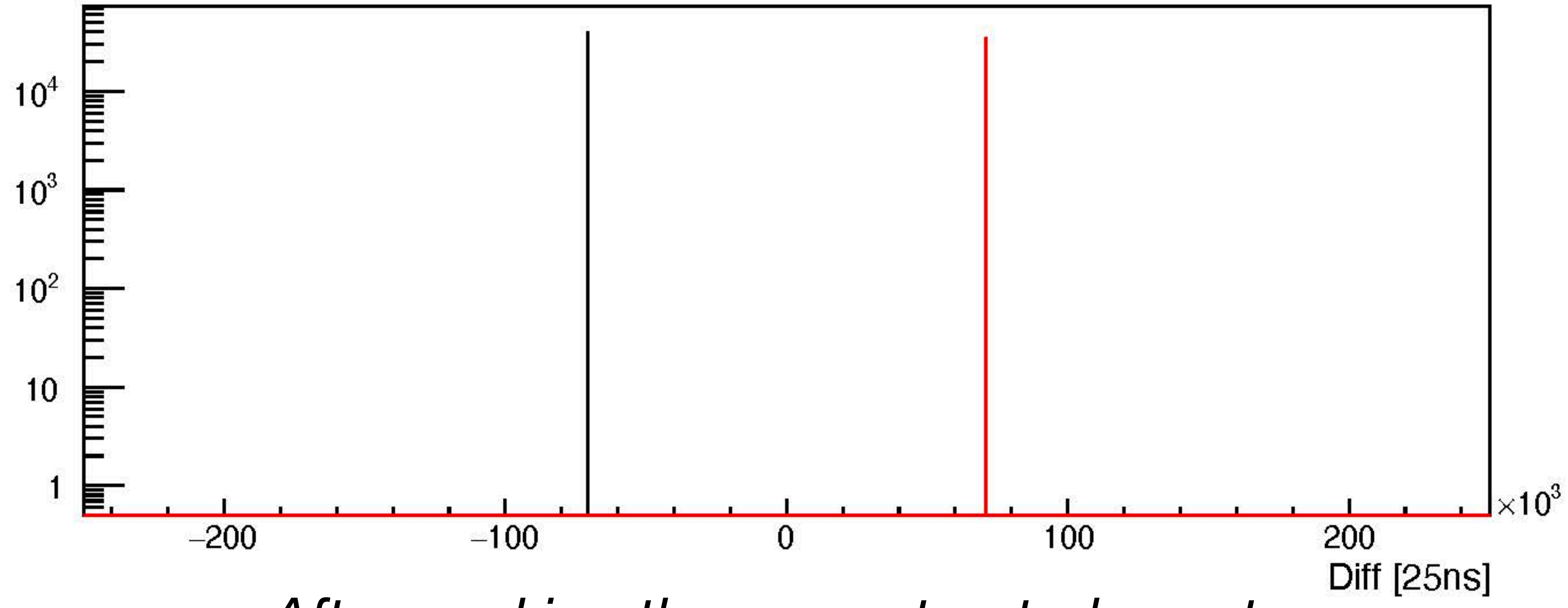




Stitching together the two KCU entries

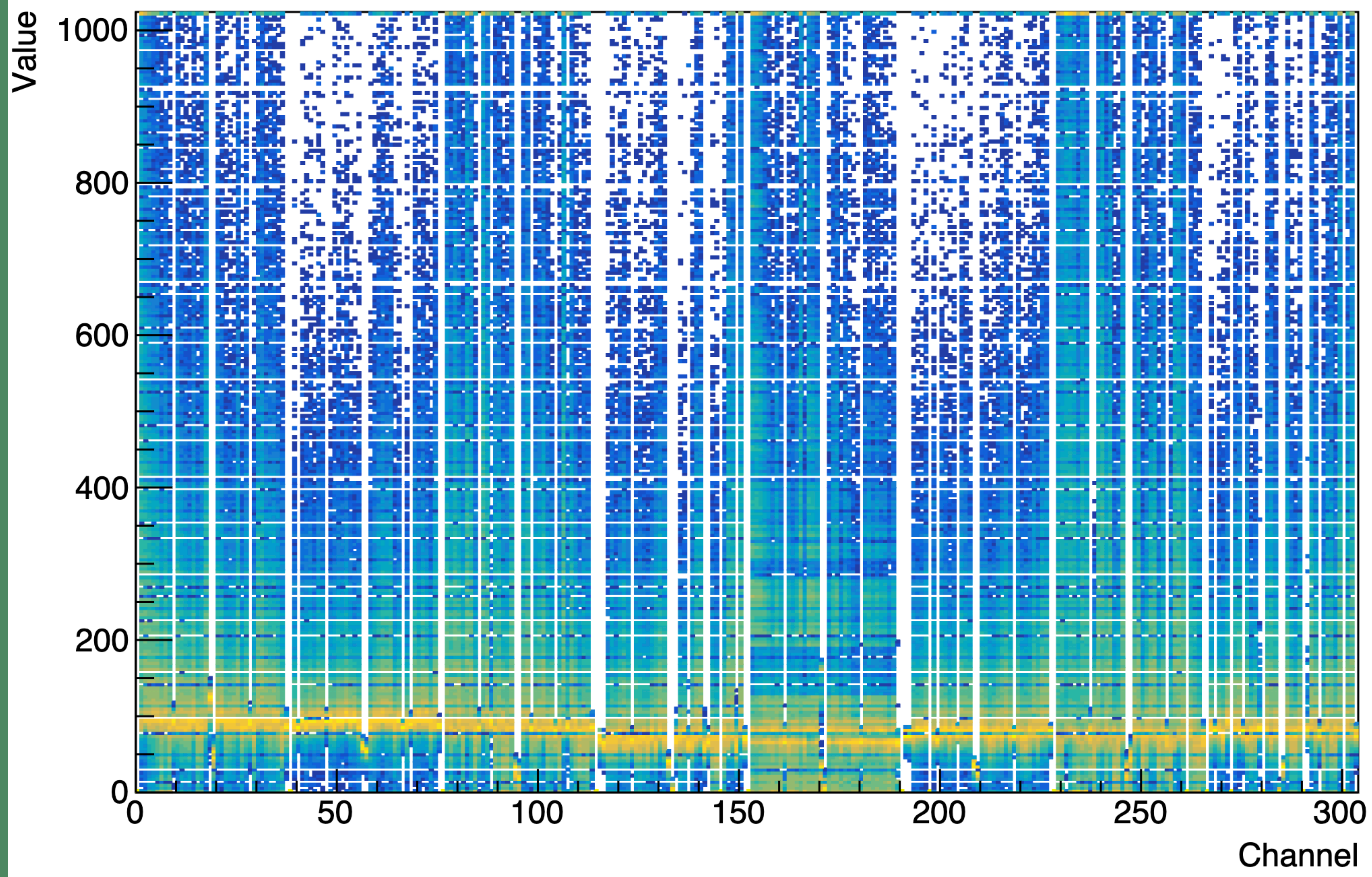
Just trying to find the offset between the two KCU's





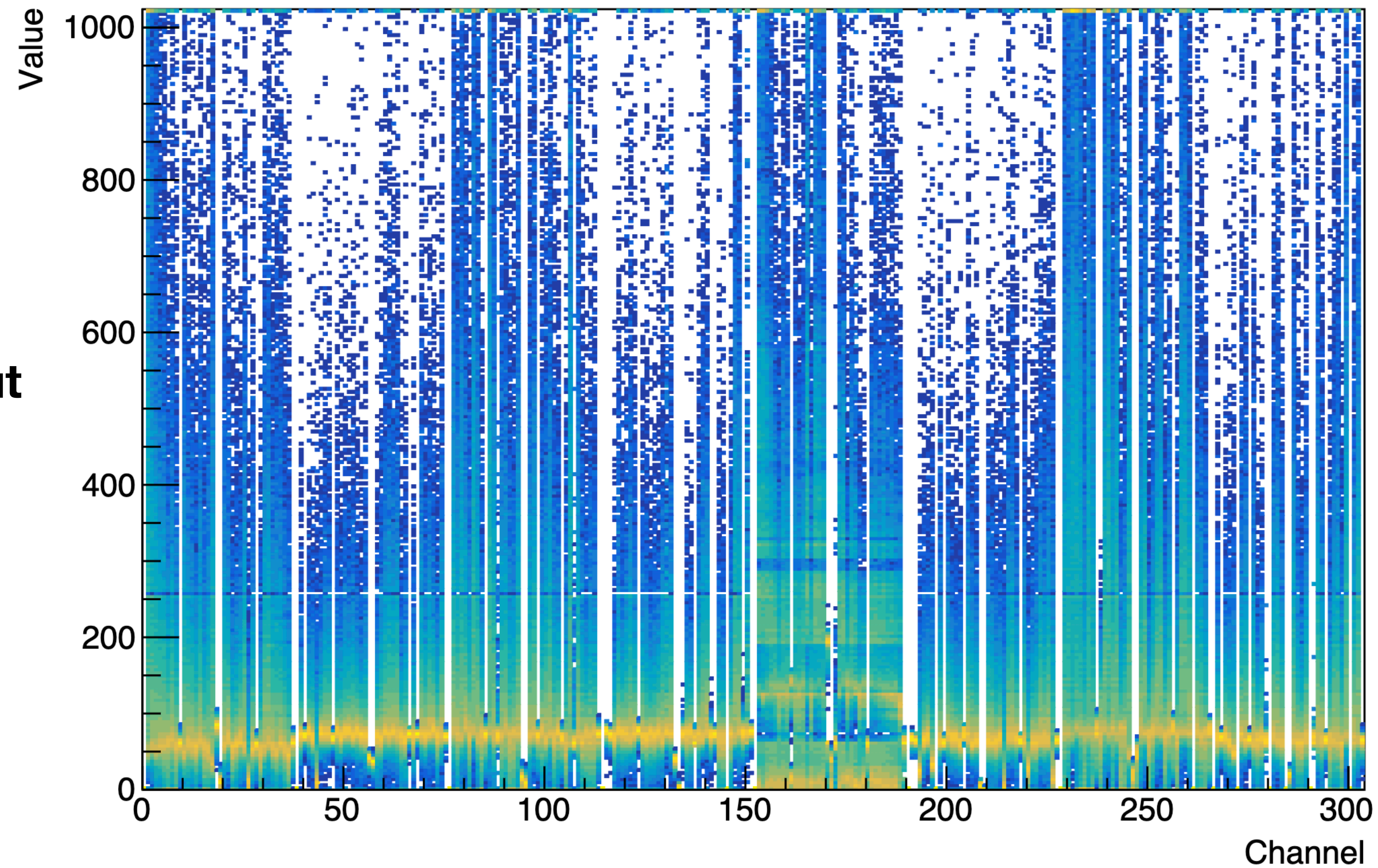
After marking the reconstructed events

Some fixes with the ADC delays



Lot of missing codes in the ADC distribution:

- This is completely normal, one could update the ADC delays in the I2C register

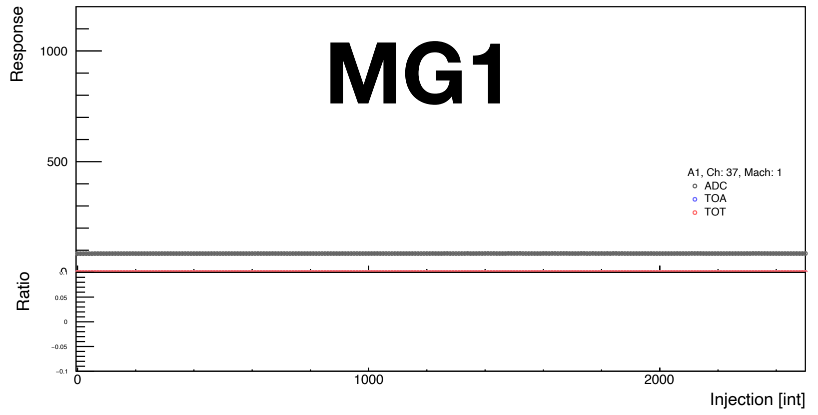


After adjustments, there is still one missing ADC code, but overall looks much better:

- Only drawback here is that the pedestals change and therefore one has to recalibrate everything again (pedestal, TOA, TOT thresholds)

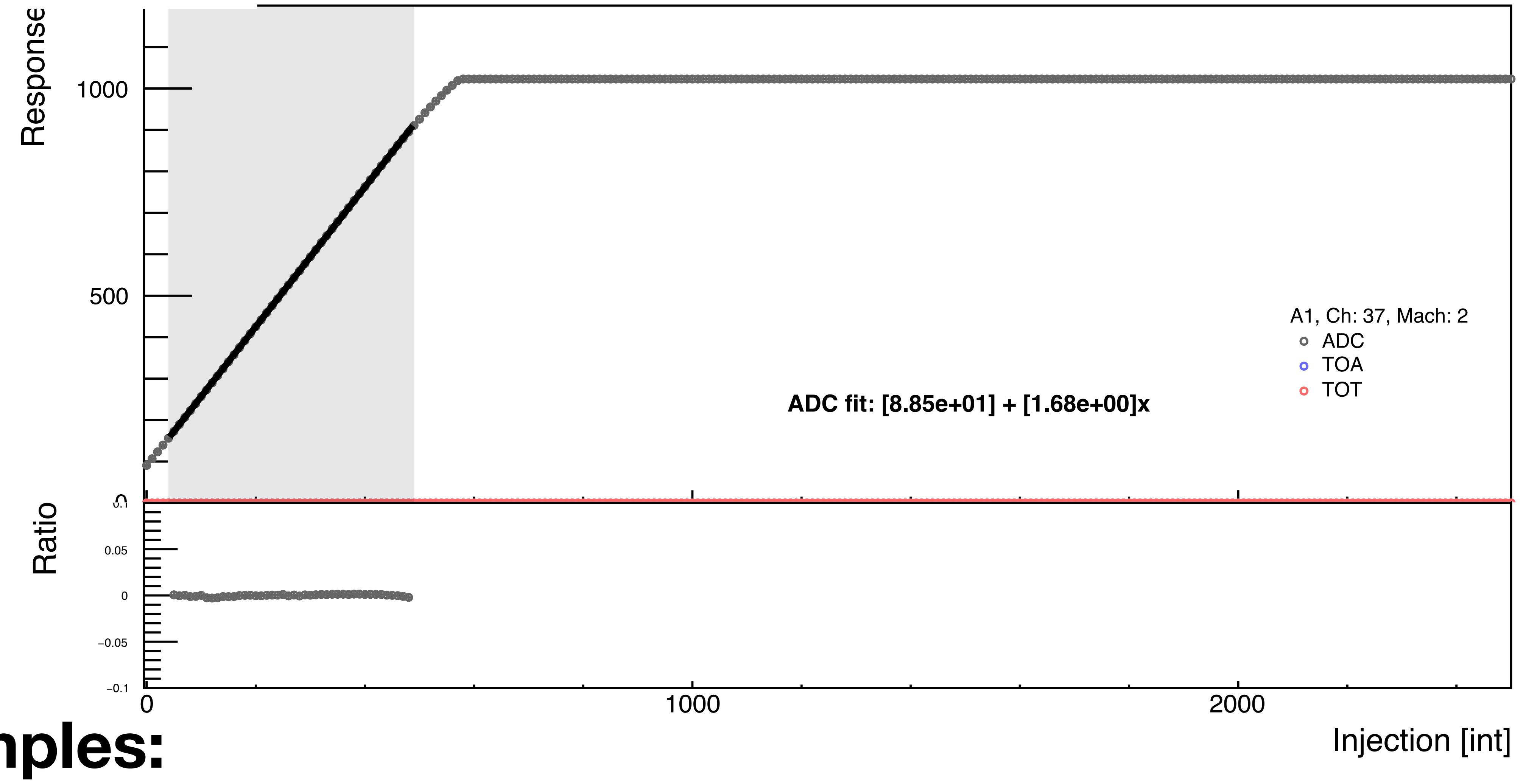
Calibration runs with TOT - this is to combine ADC-TOTs together

Even in calibration run, we run with machine gun trigger (multiple samples per hit)

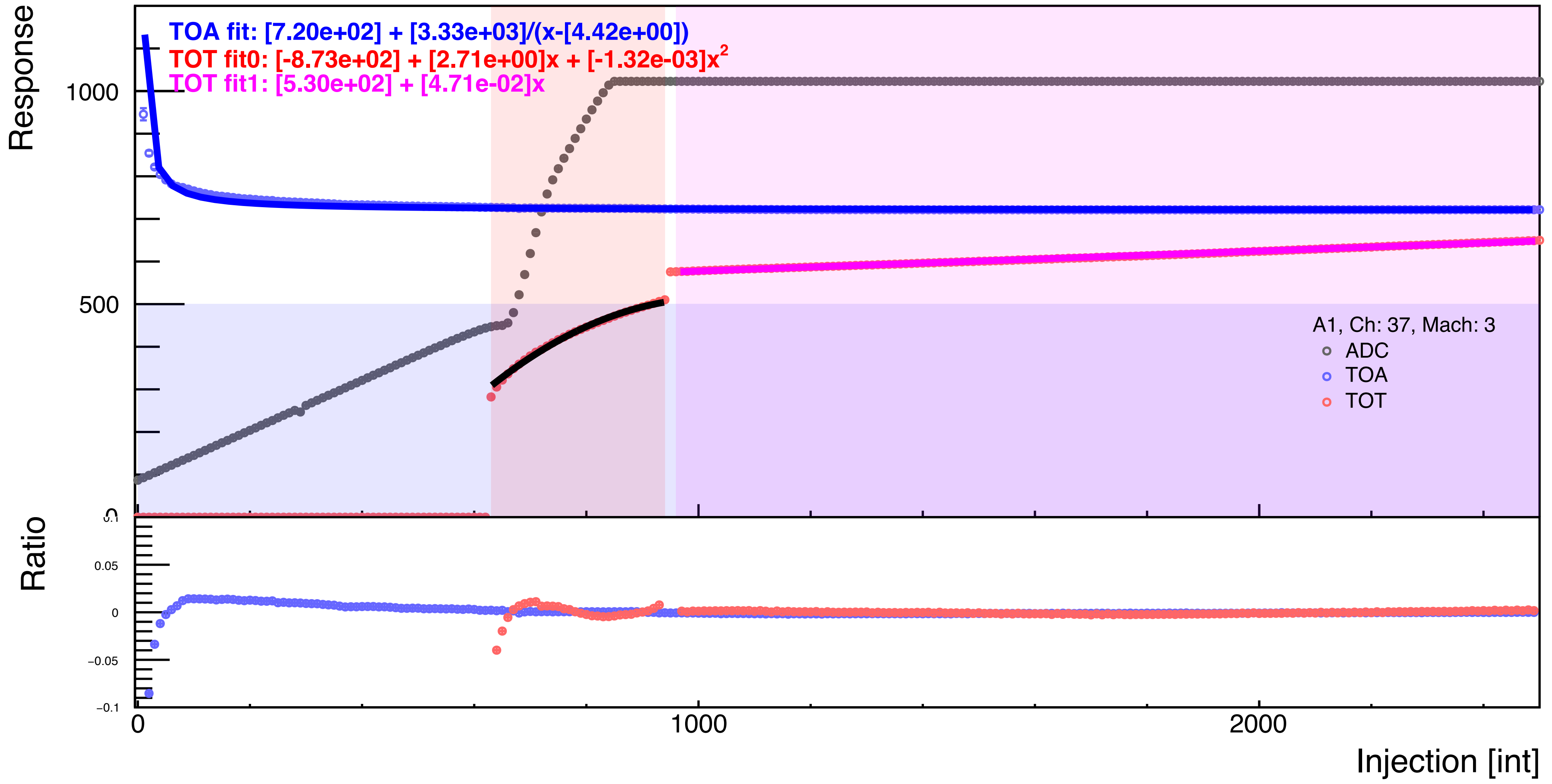


Internal injection of the chip can help identify the ADC/TOA/TOT behavior

MG2



MG3

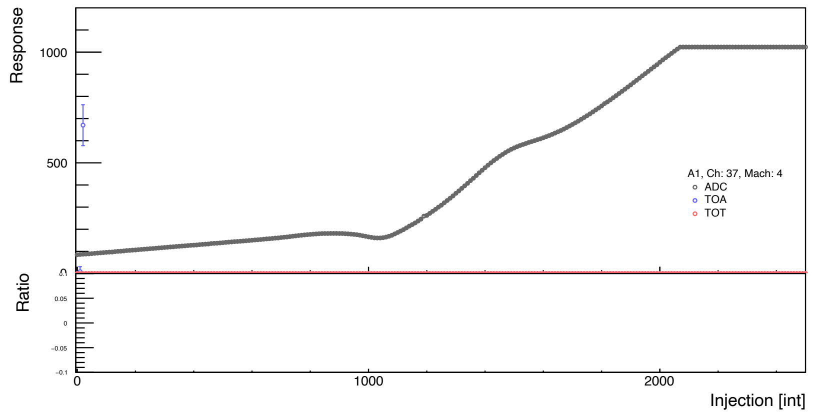


Different samples:

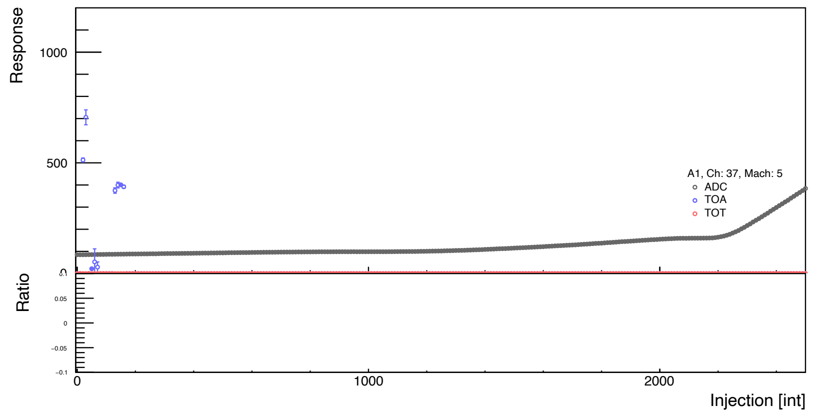
- MG0-1 - just pedestal before the signal
- MG2 - only ADC fires, it goes up to saturation
- MG3:
  - TOA has the slewing in the beginning
  - TOT has two regions:
    - < 512 - more quadratic function than linear
    - > 512 - very stable linear function

After the signal samples

MG4

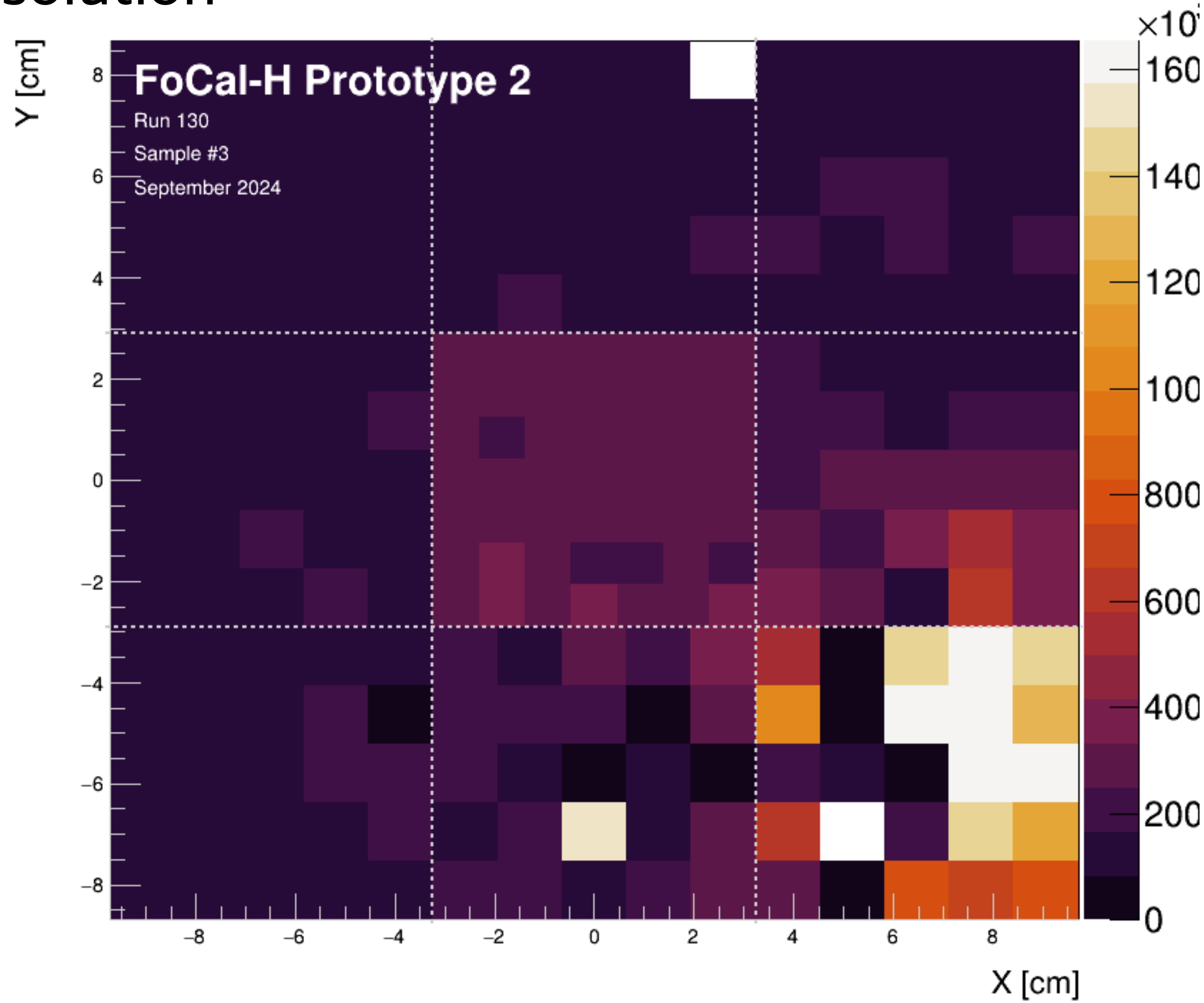


MG5



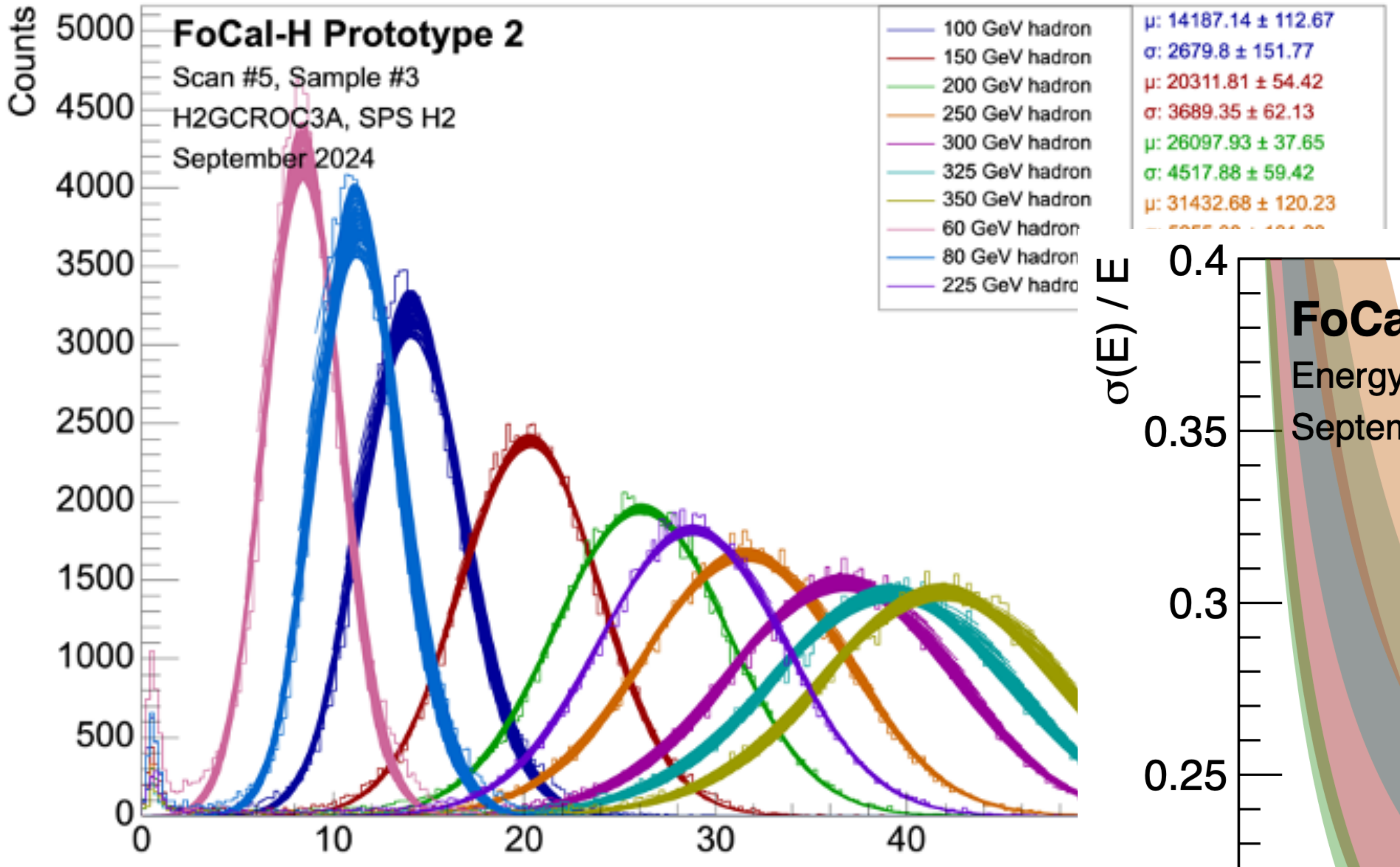
OAK RIDGE National Laboratory

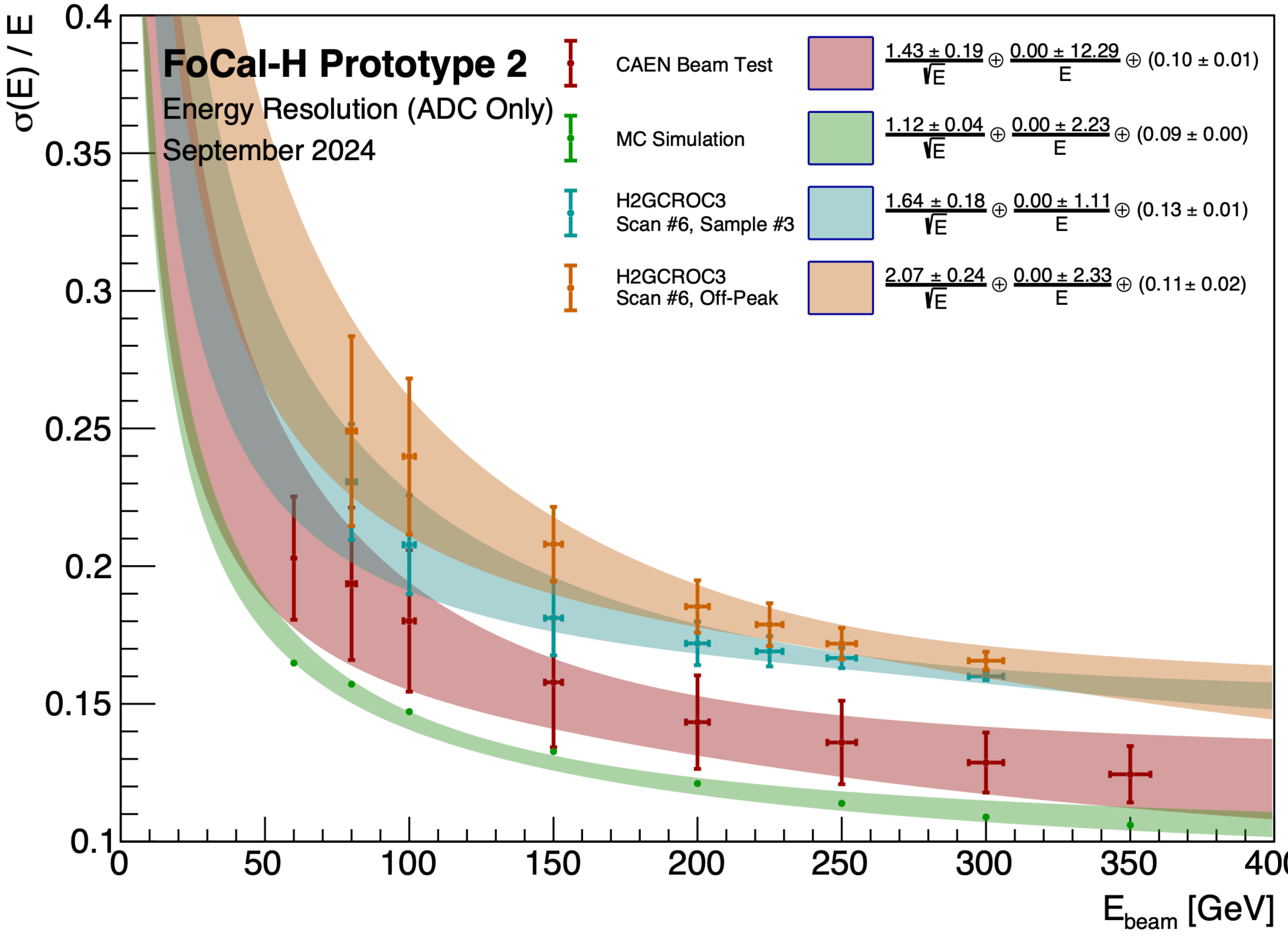
Position resolution

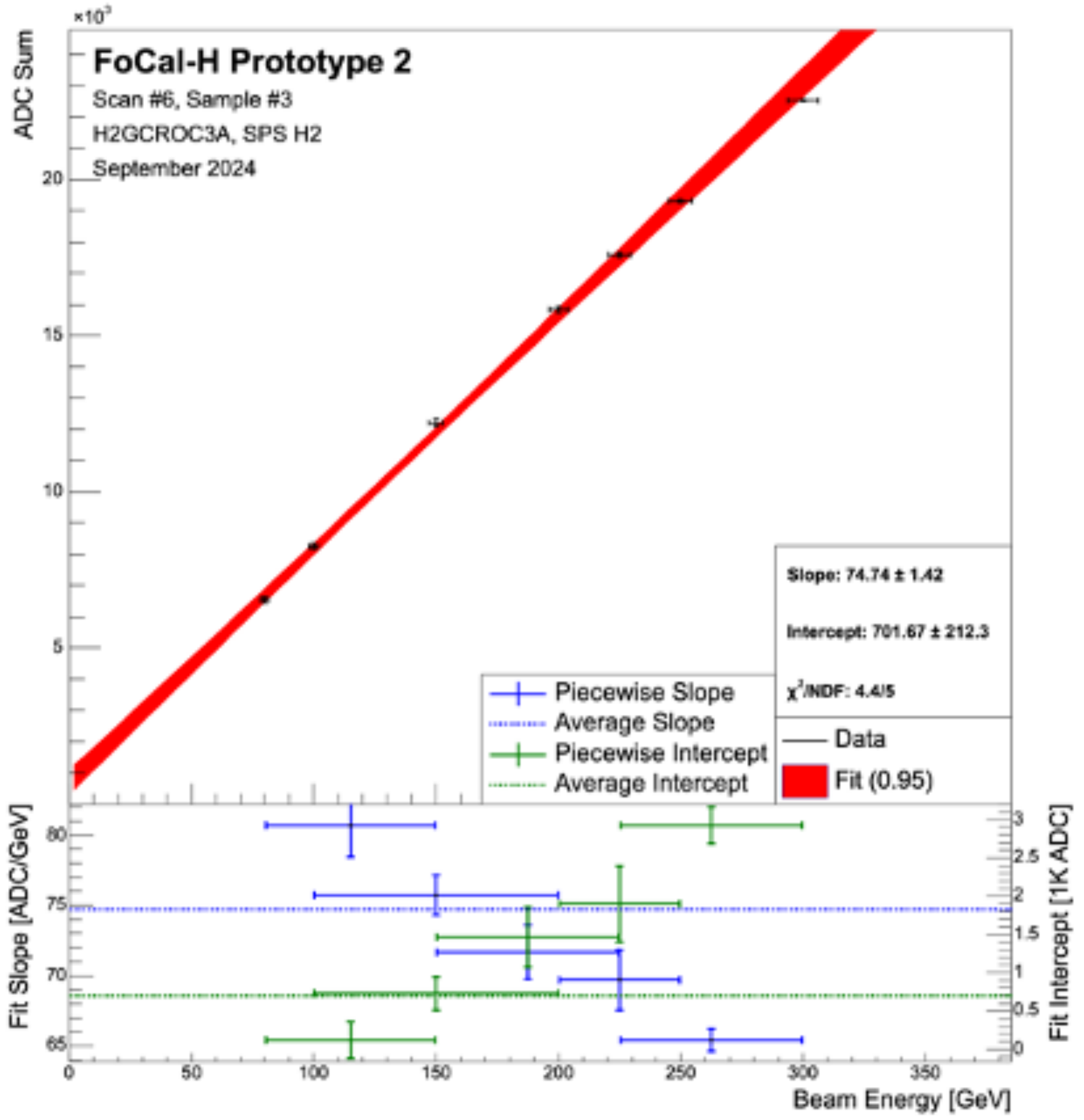


OAK RIDGE
National Laboratory

Some very early resolution figures







These are just teaser plots, the analysis is still ongoing and we are working on combining the TOT into the analysis

Not done yet, full signal reconstruction

$$\text{ADC} = f(x;\ \alpha, n, \bar{x}, \sigma, N, B) = \begin{cases} N \cdot \exp\left(-\frac{(x-\bar{x})^2}{2\sigma^2}\right) + B\,, & \text{for } \frac{x-\bar{x}}{\sigma} < \alpha \\ N \cdot A \cdot \left(B + \frac{x-\bar{x}}{\sigma}\right)^{-n} + B\,, & \text{for } \frac{x-\bar{x}}{\sigma} > \alpha \end{cases}$$

α: the point where the Gaussian transitions to the power-law tail, fixed by pre-fit
n : the exponent that determines the slope of the power-law tail, fixed by pre-fit
$\bar{x}$: peak position, free parameter
σ: the standard deviation of the Gaussian core, fixed
N: a normalization factor, or the amplitude of the peak, free parameter
B: baseline voltage, fixed

Starting to reconstruct the full signal:
Tried Semi Gaussian, but looks like the crystal ball function is more stable

Train the fitting on ADC only (left) then once the ADC is saturating

Still adding TOA and TOT into the mix