

# INTT various updates

Cheng-Wei Shih,  
National Central University/RIKEN

Dec 13th, 2024  
INTT meeting



國立中央大學  
National Central University

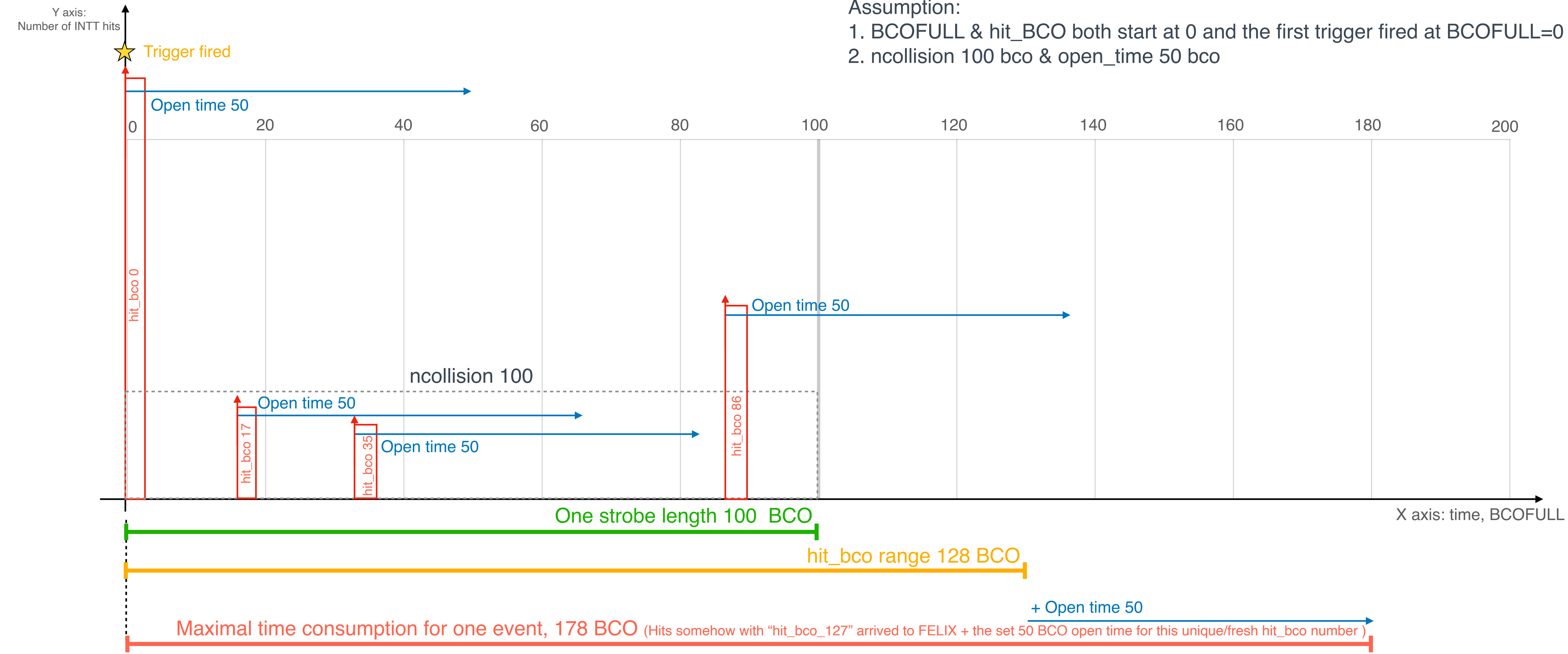


Note: the hit transmission from chip to ROC: 1 hit / 1 bco

In single event

Assumption:

1. BCOFULL & hit\_BCO both start at 0 and the first trigger fired at BCOFULL=0
2. ncollision 100 bco & open\_time 50 bco



- Current concept: when the first hit with “hit\_bco\_A” arrives at FELIX, FELIX waits for the other hits with same “hit\_bco\_A” coming for “50 BCO”, and if there is another hit with “hit\_bco\_B”, the FELIX would open another “50 BCO” for the hits with “hit\_bco\_B”

Code can be found in [GitHub](#)

Run 54280, first 3M events

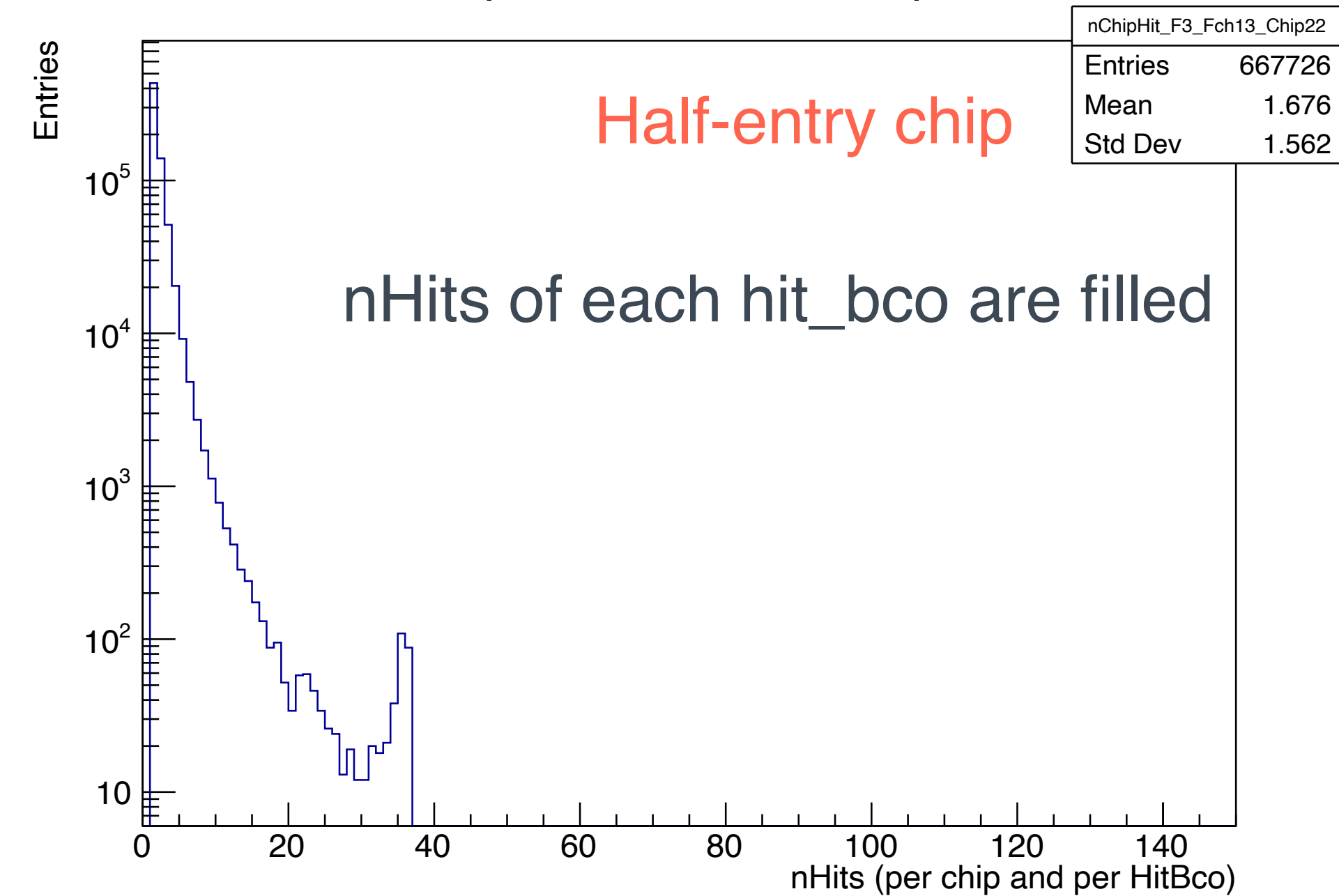
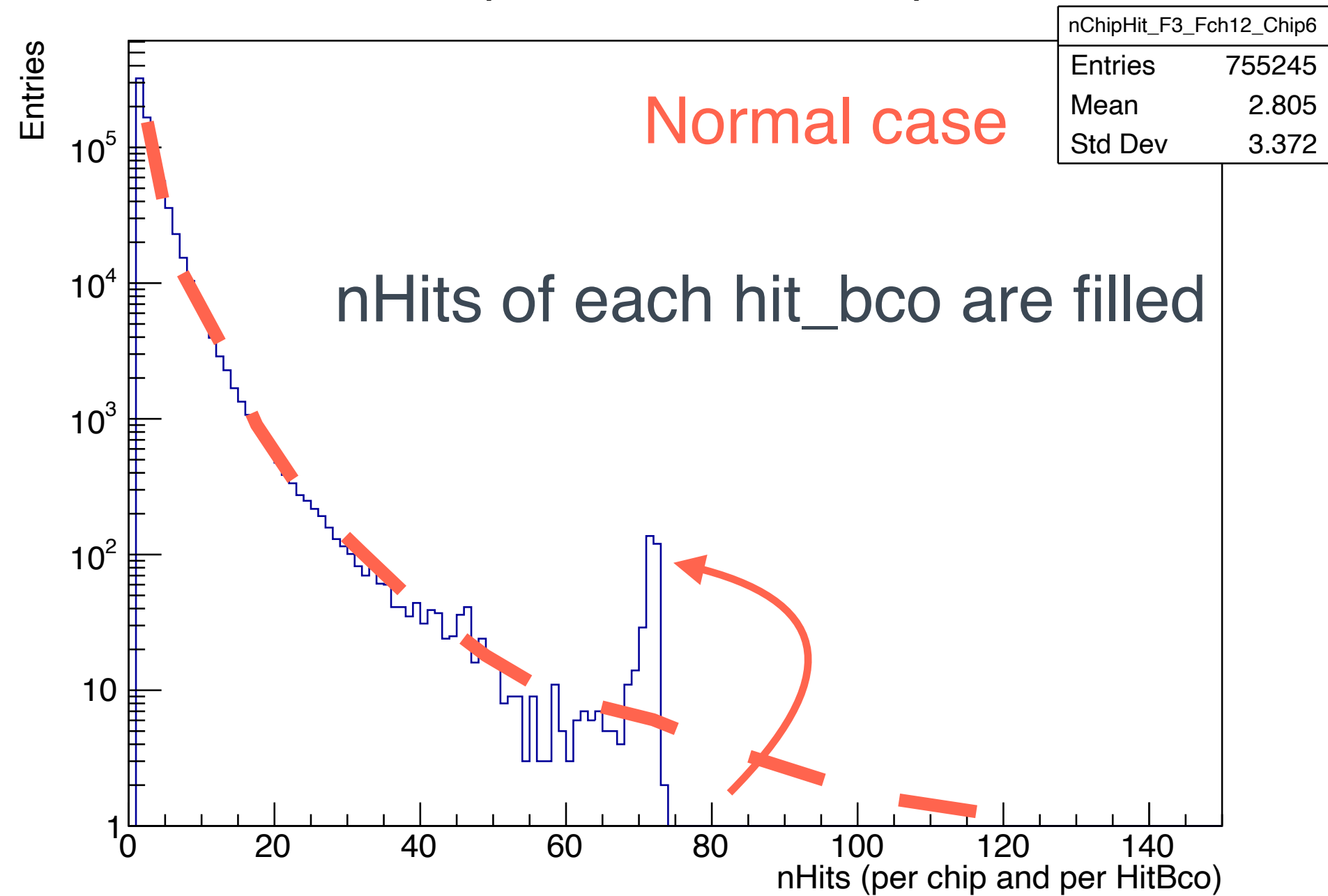
With HitQA and CloneHit Removal (CloneHit: same FELIX, FELIX\_ch, chip\_id, chan\_id, hit\_bco)

Count the number of hits of each chip, per hit\_bco

```
daq=> select * from intt_setting where runnumber = 54280;
runnumber | n_collisions | open_time | readout_mode | dac0 | dac1 | dac2 | dac3 | dac4 | dac5 | dac6 | dac7
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----
54280 | 100 | 60 | triggered | 35 | 45 | 60 | 90 | 120 | 150 | 180 | 210
```

nChipHit\_F3\_Fch12\_Chip6

nChipHit\_F3\_Fch13\_Chip22



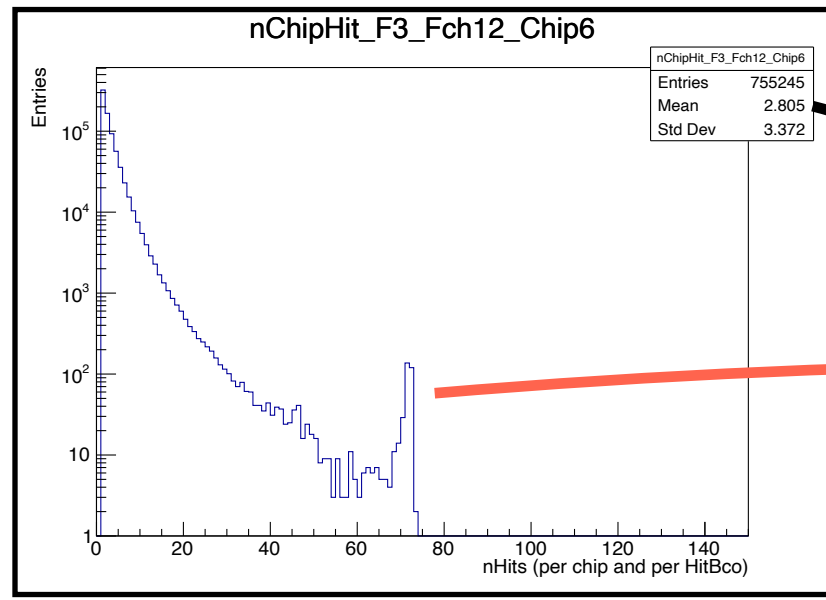
- The spike at nhits 73: hits been rejected due to the late arrival to the FELIX
  - In some extreme case, not all the hits are kept by the FELIX
- The maximal number of hits of each chip and per hit\_bco is 73
- Half-entry chips have similar structures → Hit missing happened before FELIX (at chip)

INTT has hit saturation issue

# Chip Occupancy - statistics

Run 54280, first 3M events

Total number of INTT chips:  $26 * 56 * 2 = 2912$



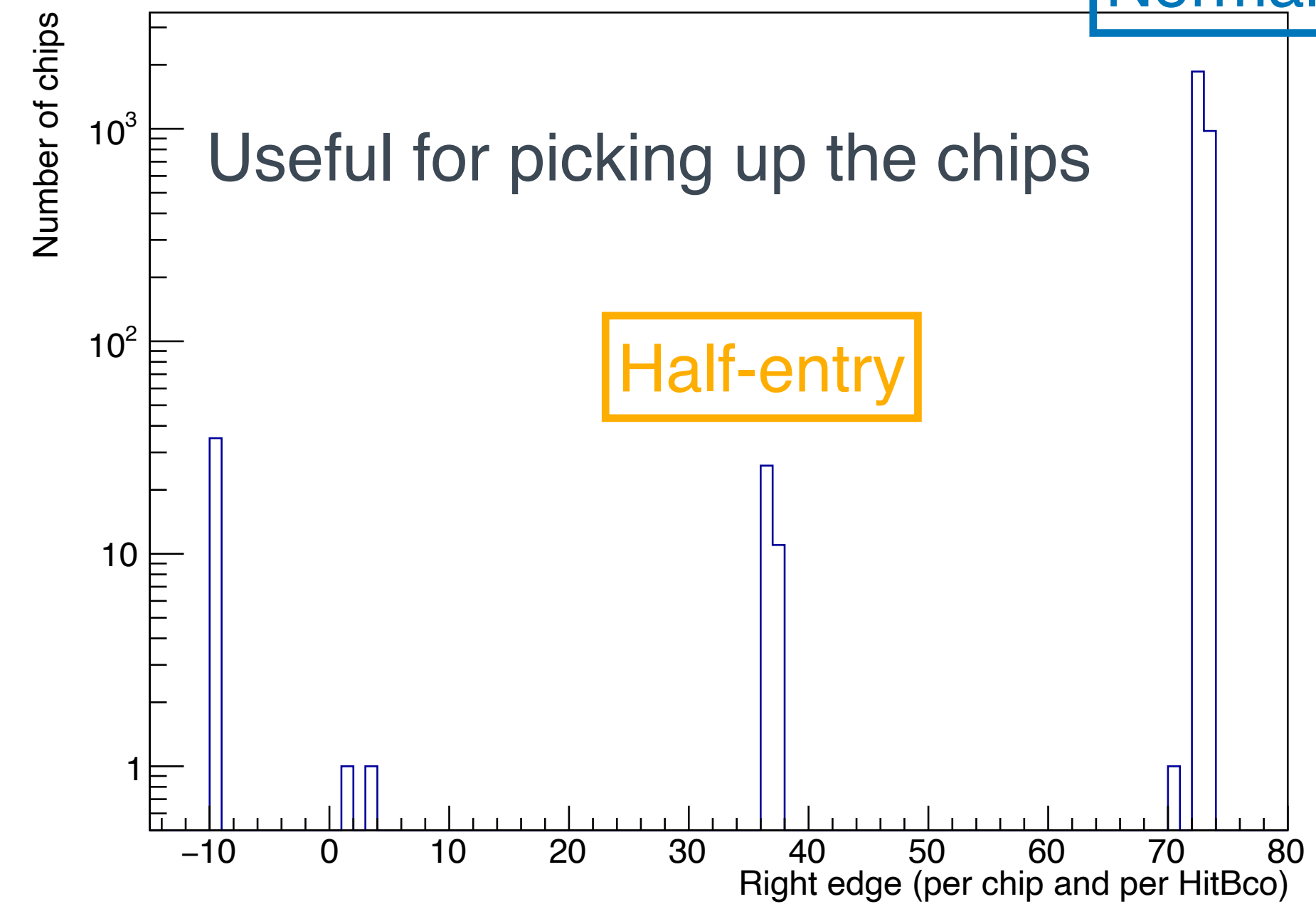
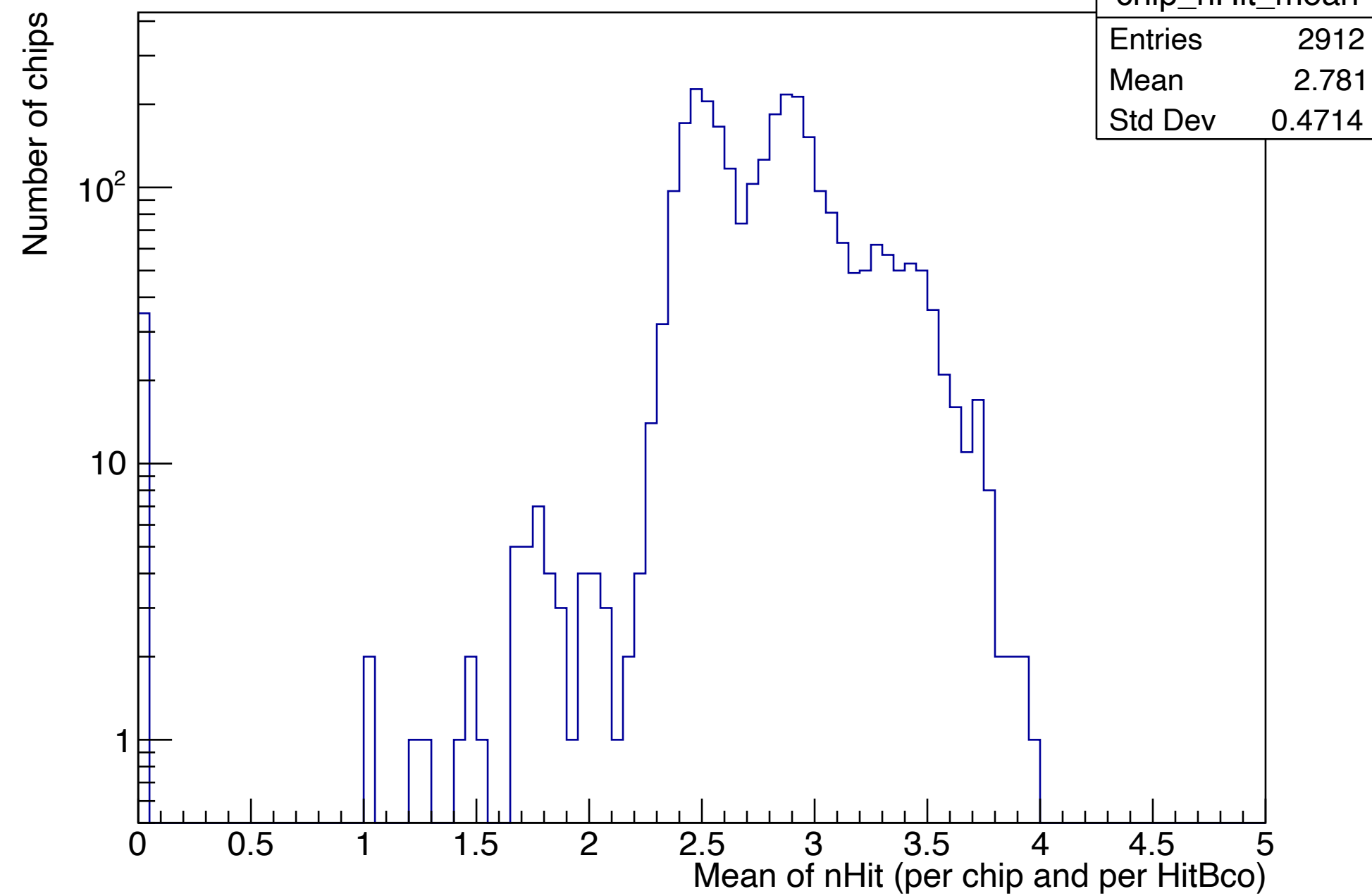
Chip mean

Right edge location

chip\_nHit\_mean

chip\_nHit\_RightEdge

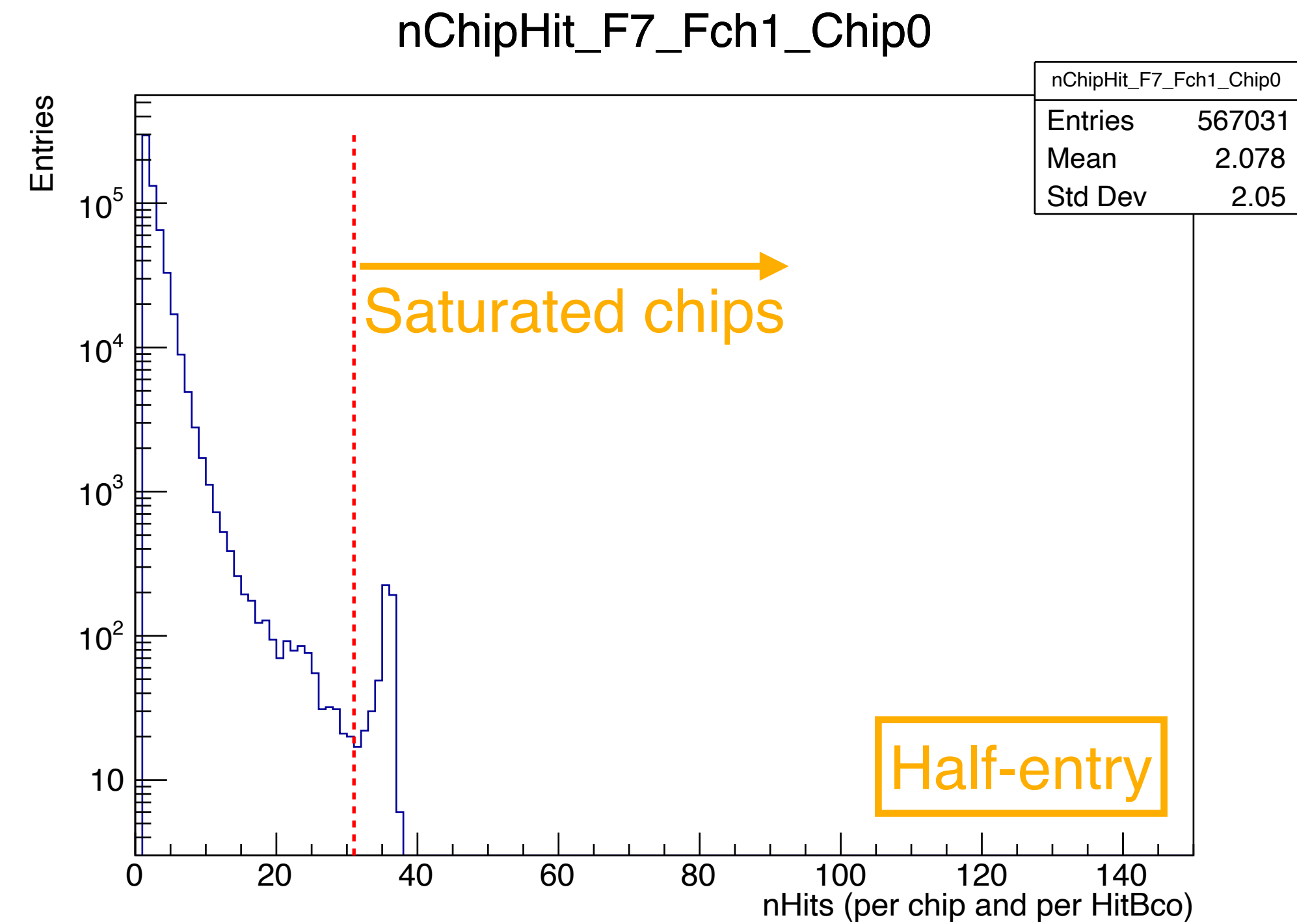
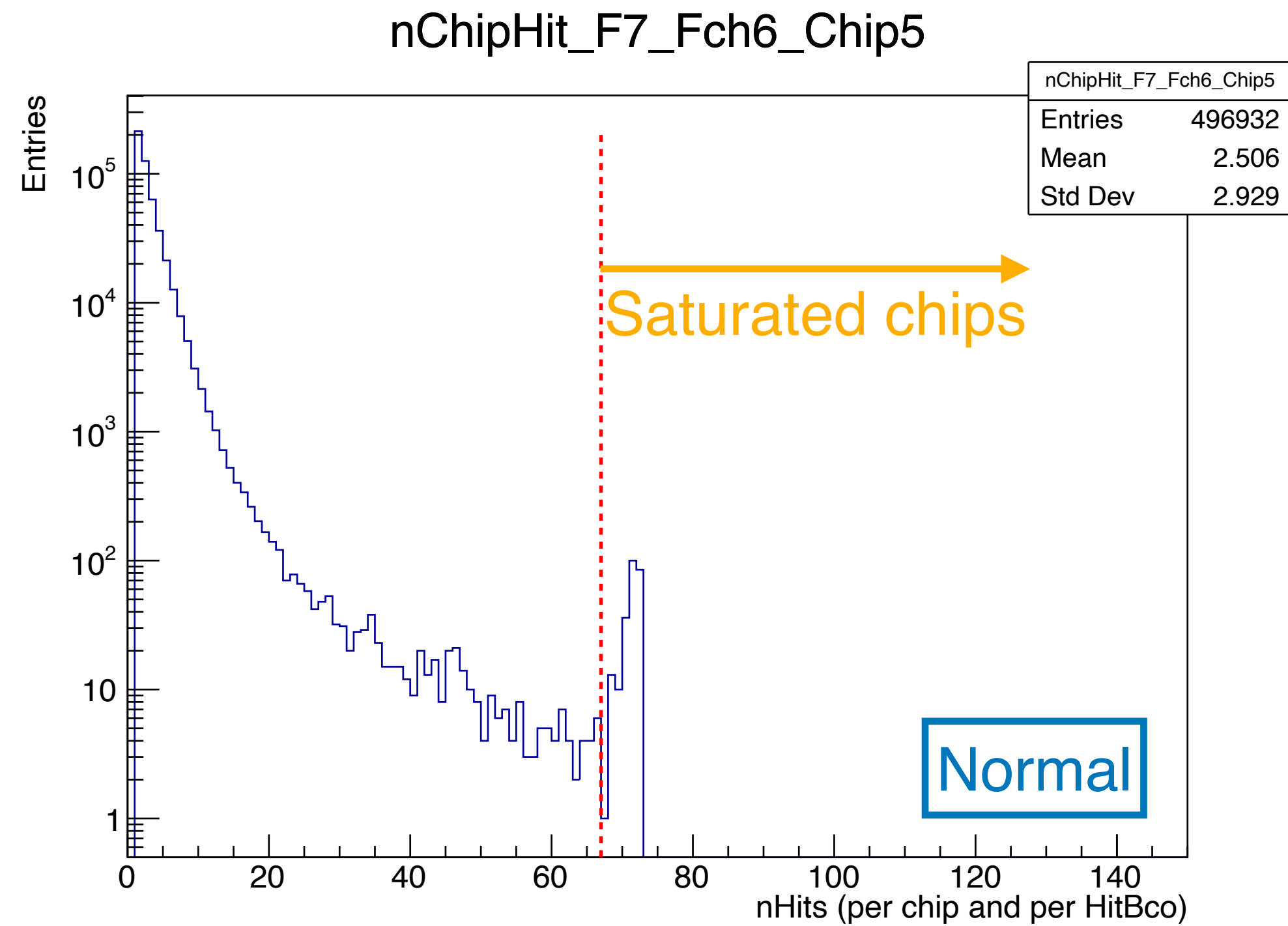
Normal



# Chip Occupancy - Saturated chips

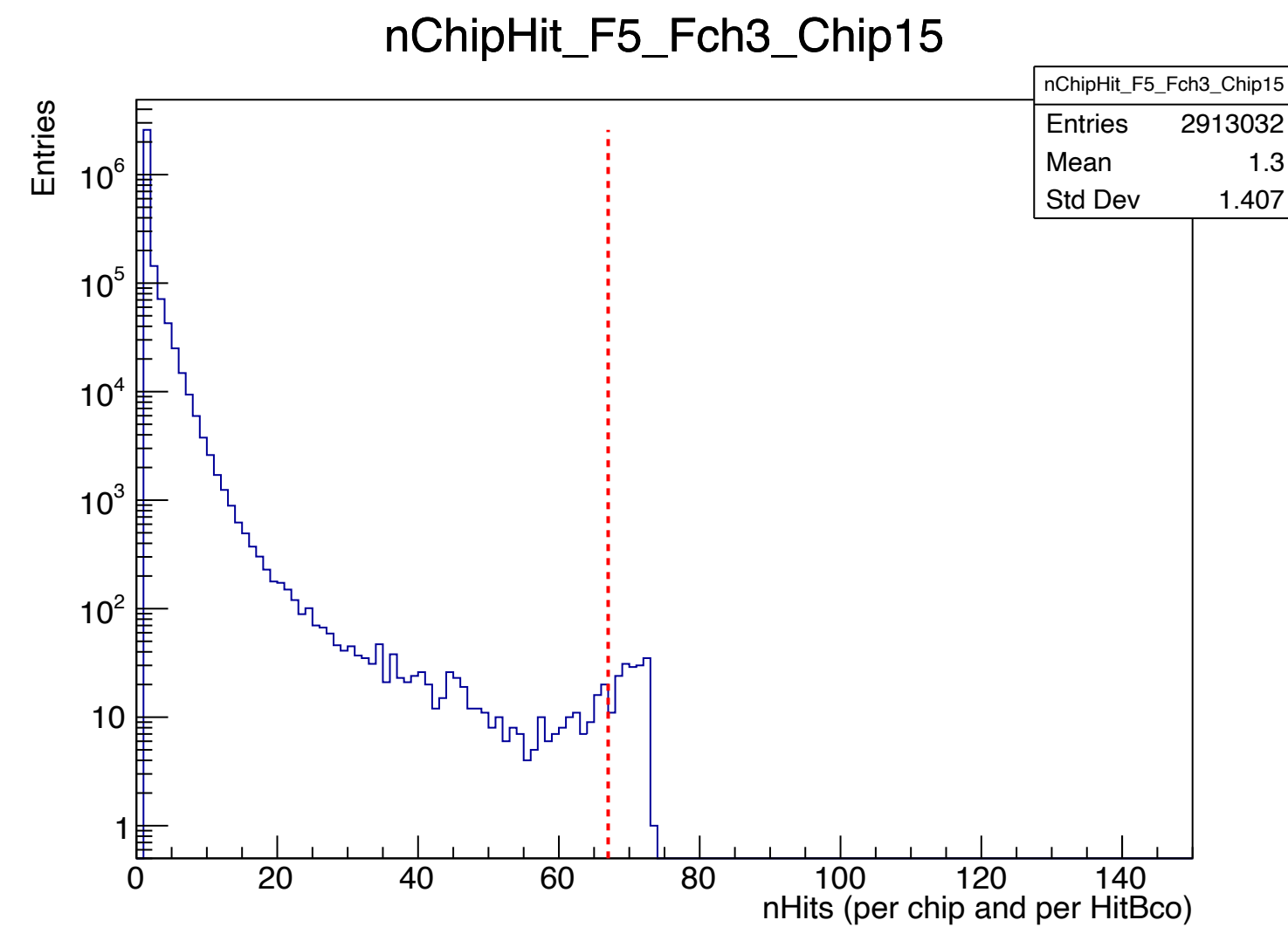
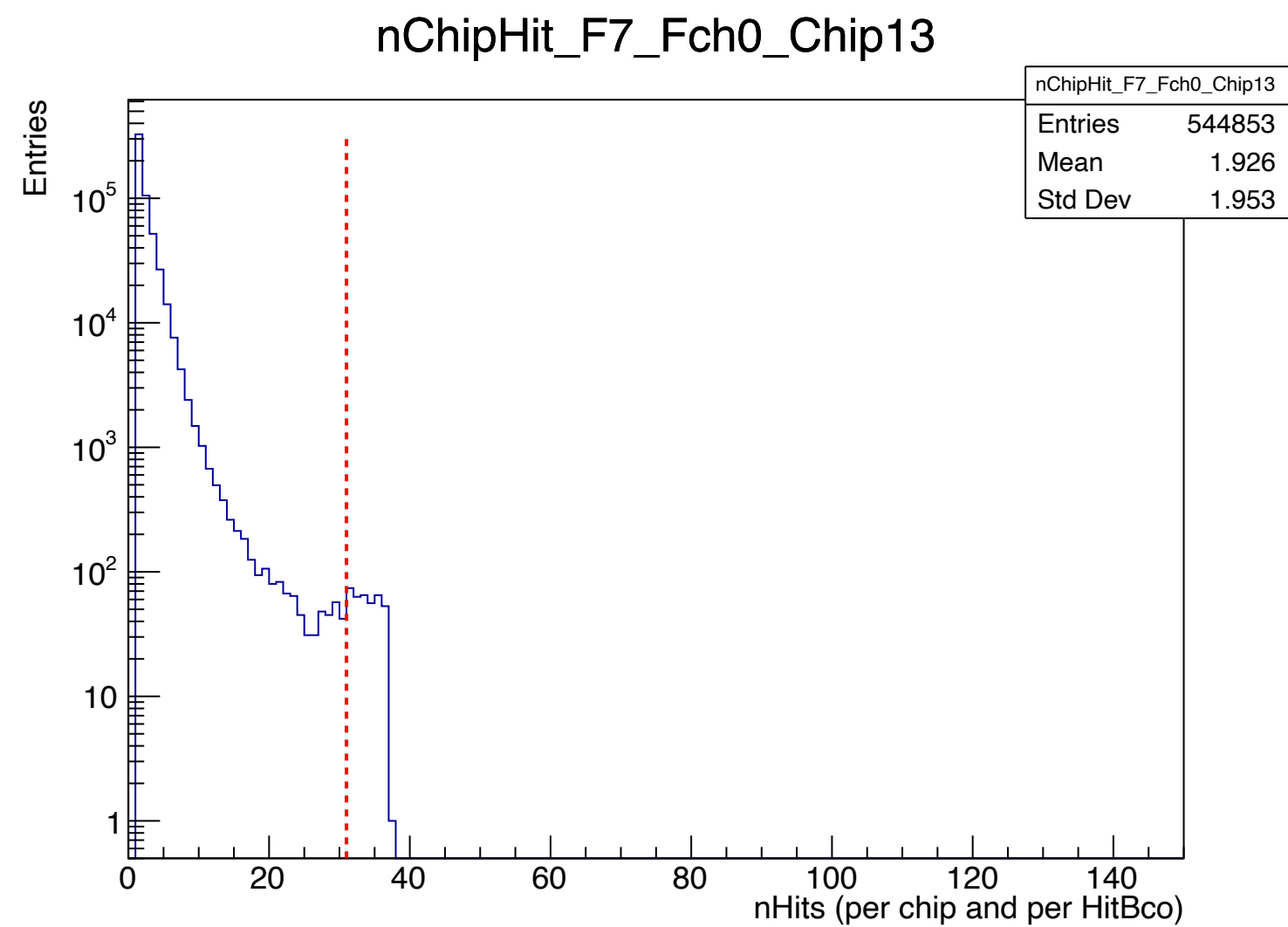
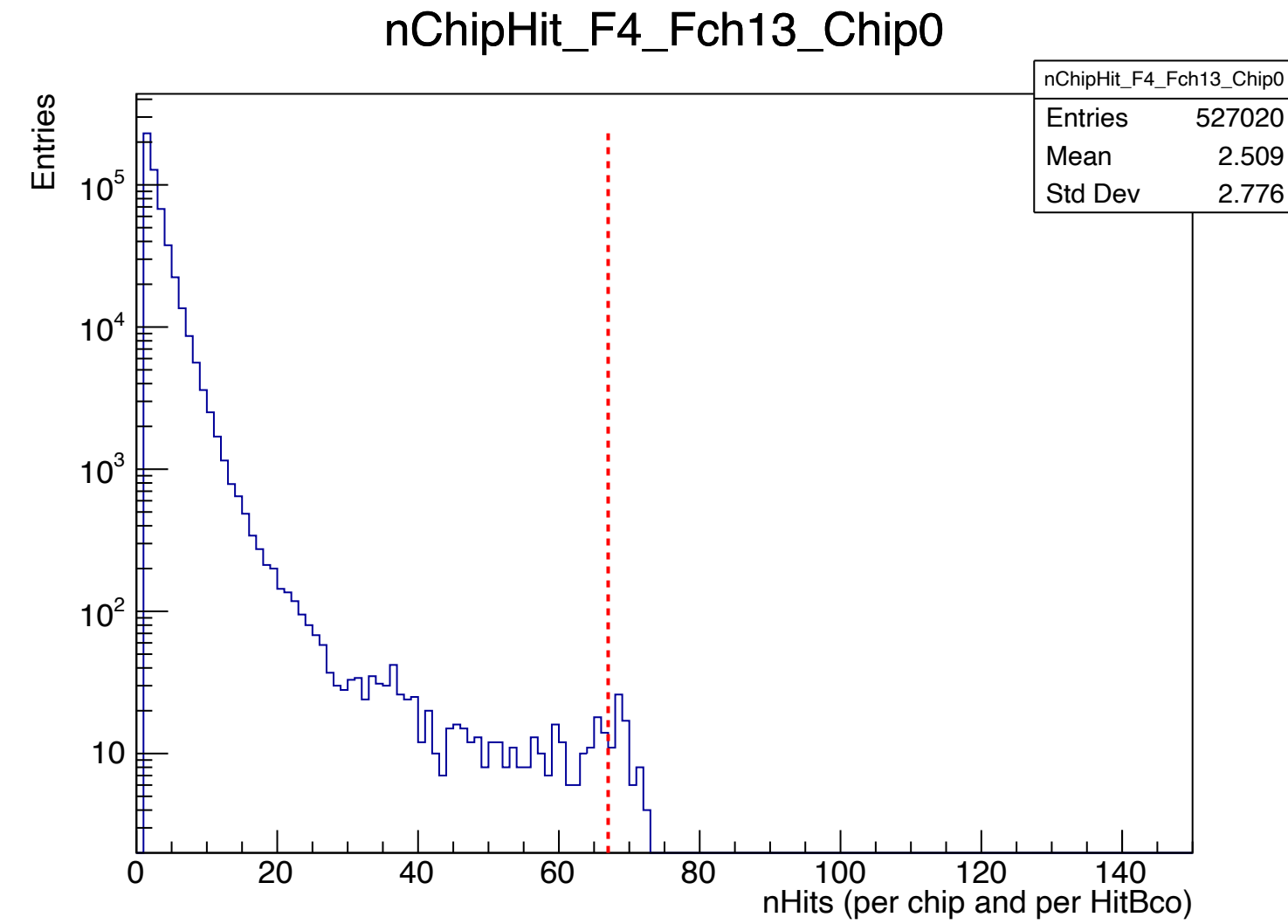
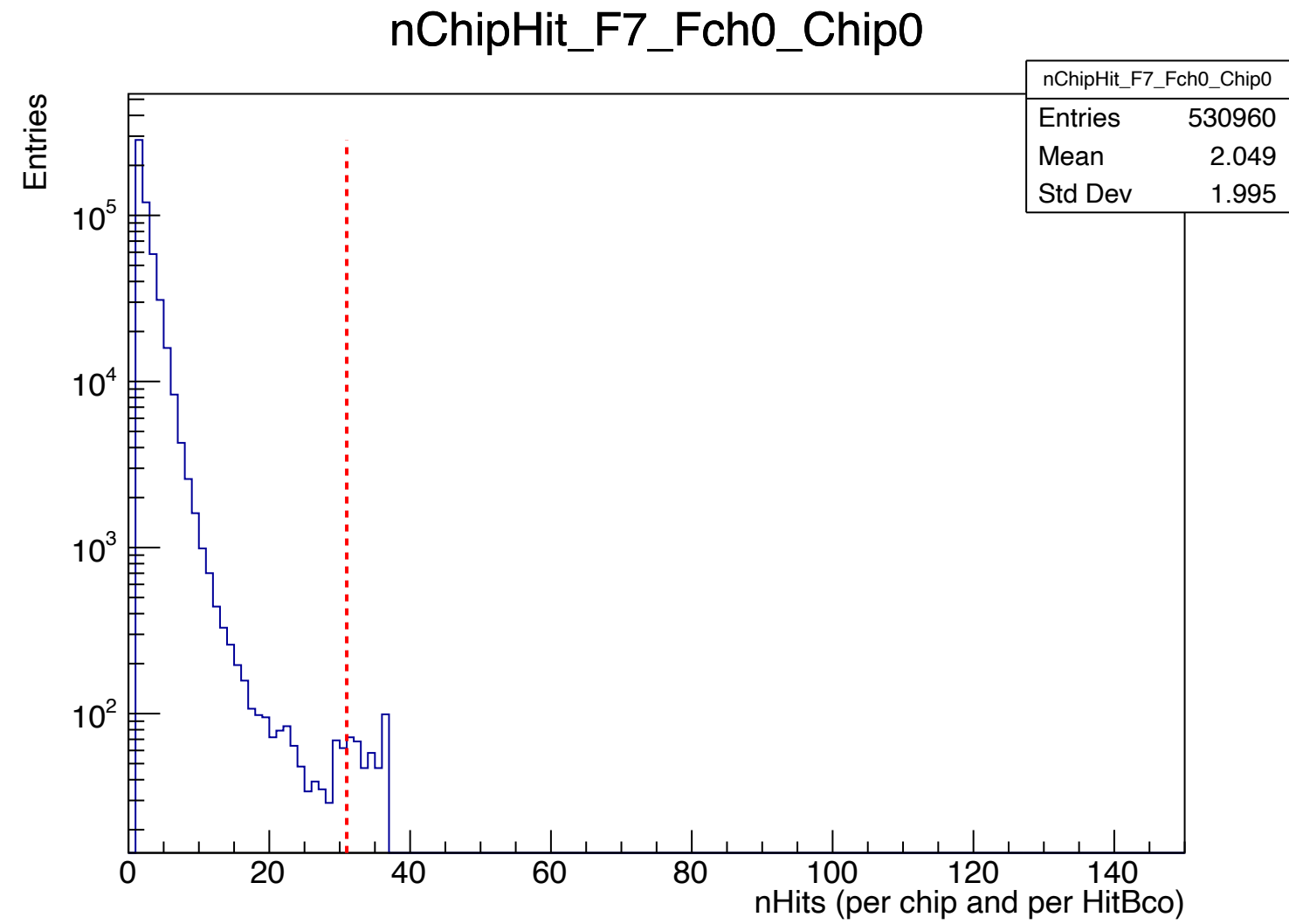
Selection

```
std::pair<double,double> normal_range = {60,80};  
double normal_threshold = 67;  
  
std::pair<double,double> halfentry_range = {30,40};  
double halfentry_threshold = 31;
```



Try to have the selections to pick up the chips saturated

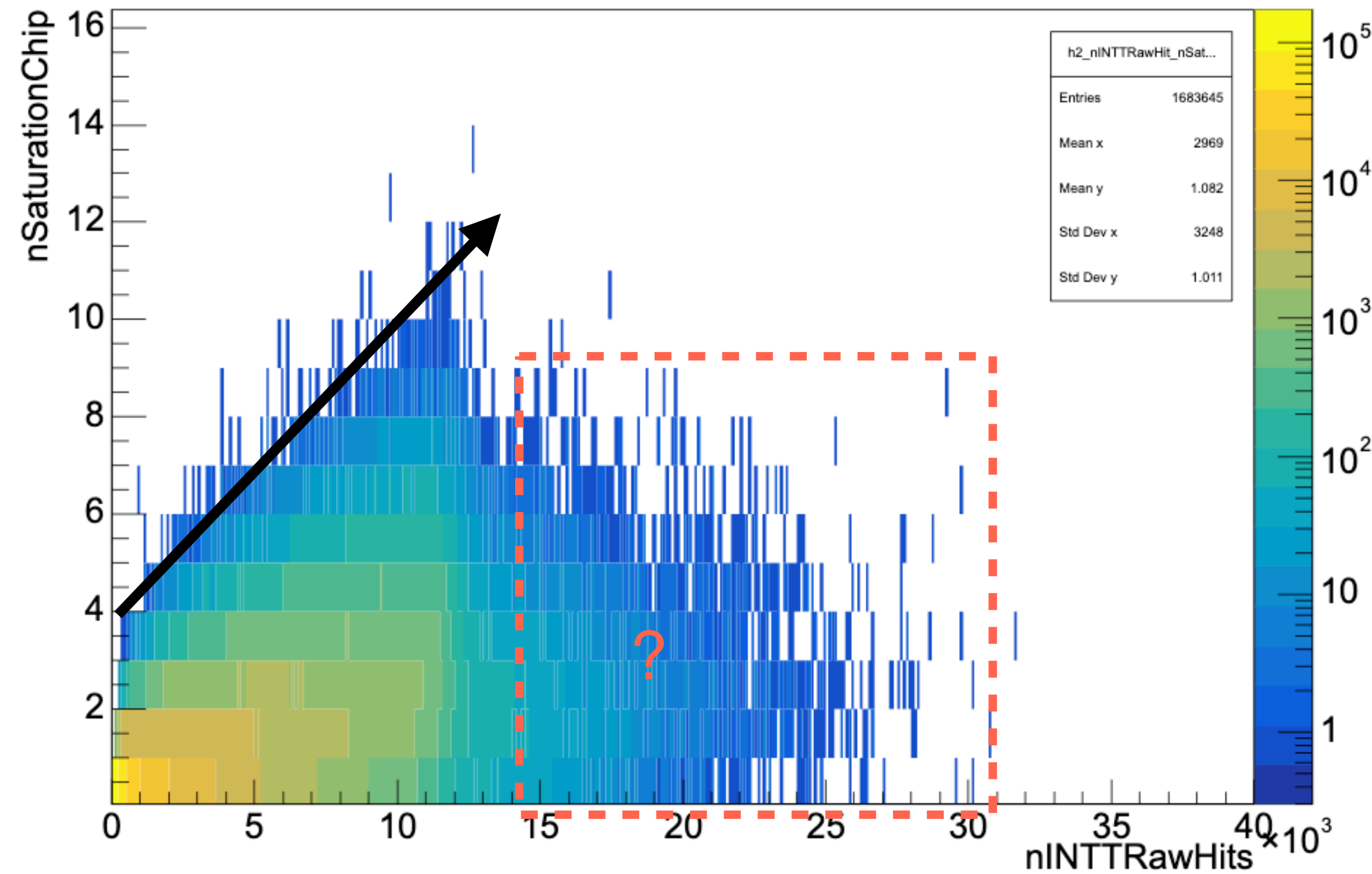
# Chip Occupancy - Saturated chips



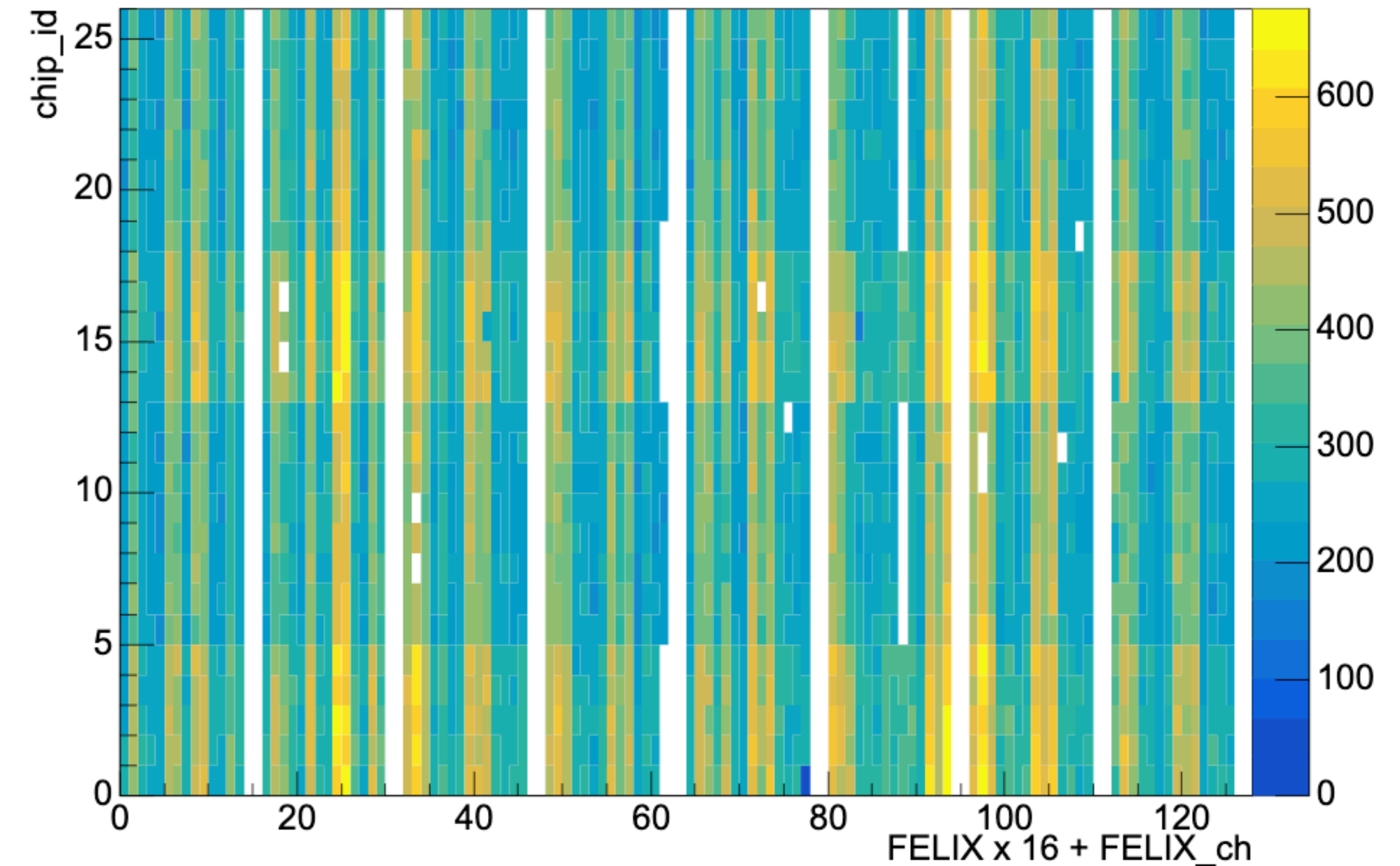
Some of the chips seem not to be suffered from the saturation problem that much, but most of the chips are suffered

# Chip Occupancy - Saturated chips

h2\_nINTTRawHit\_nSaturation



h2\_Saturation\_ChipMap



In the worse case, 12 out of 2912 chips are saturated in one event

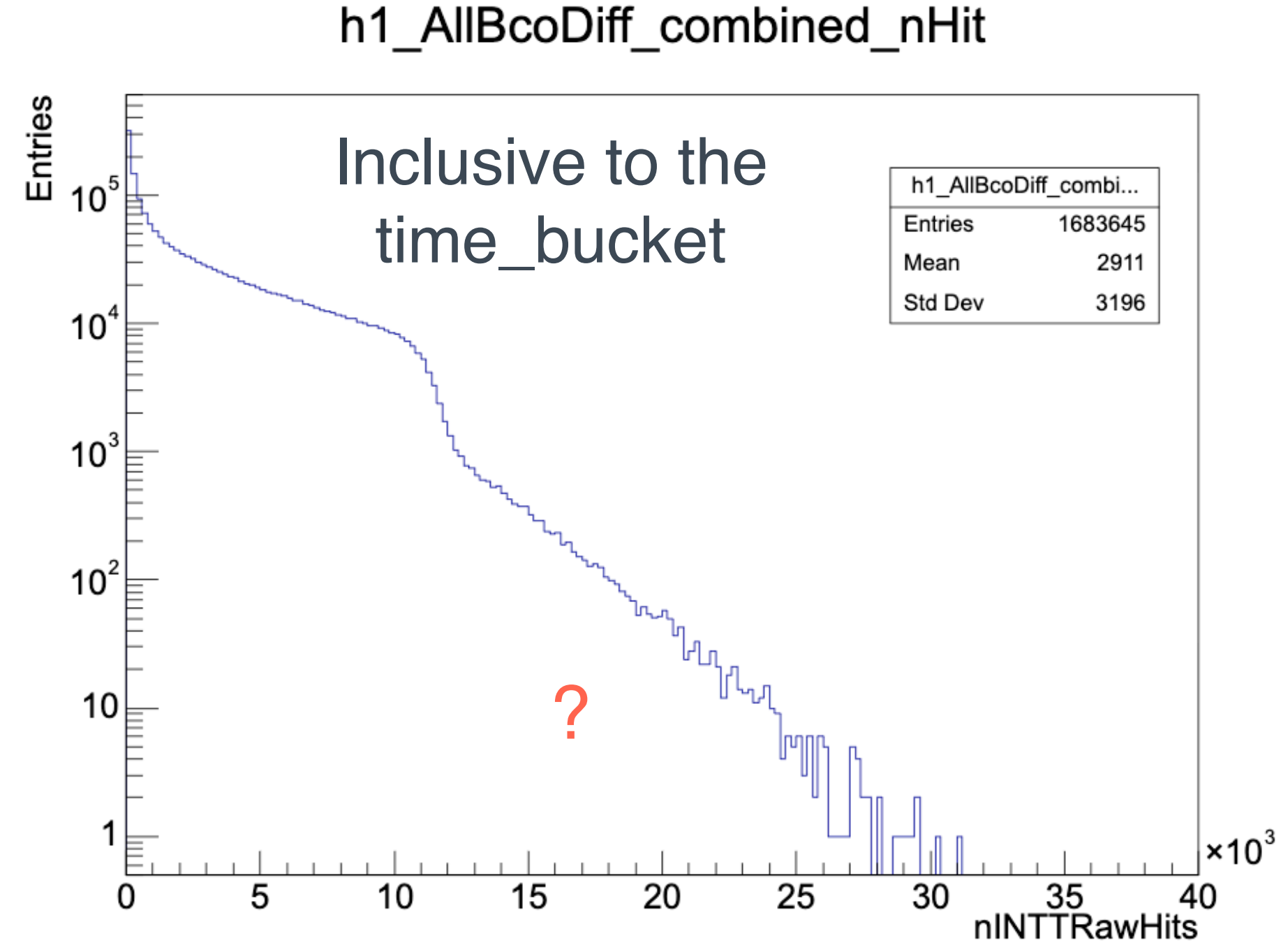
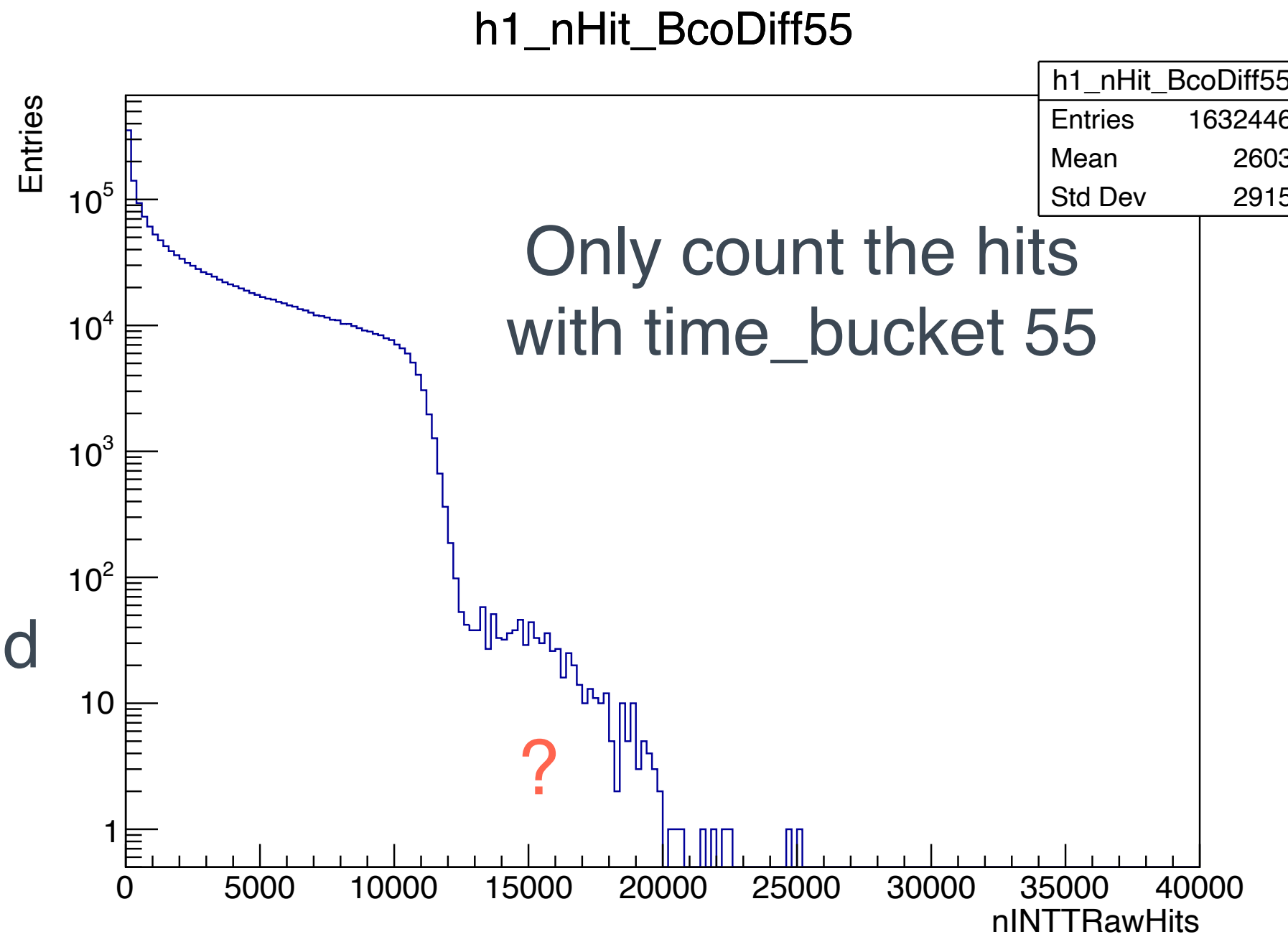
Assuming those chips have all channels fired,  $(128 - 73) * 12 = 660$  hits are dropped by FELIX servers

$660 / (13000 + 660) = \sim 5\%$  of the hits are missing

But we might gain more clusters (non-physical)

# nINTTRawHit, to check the exceed

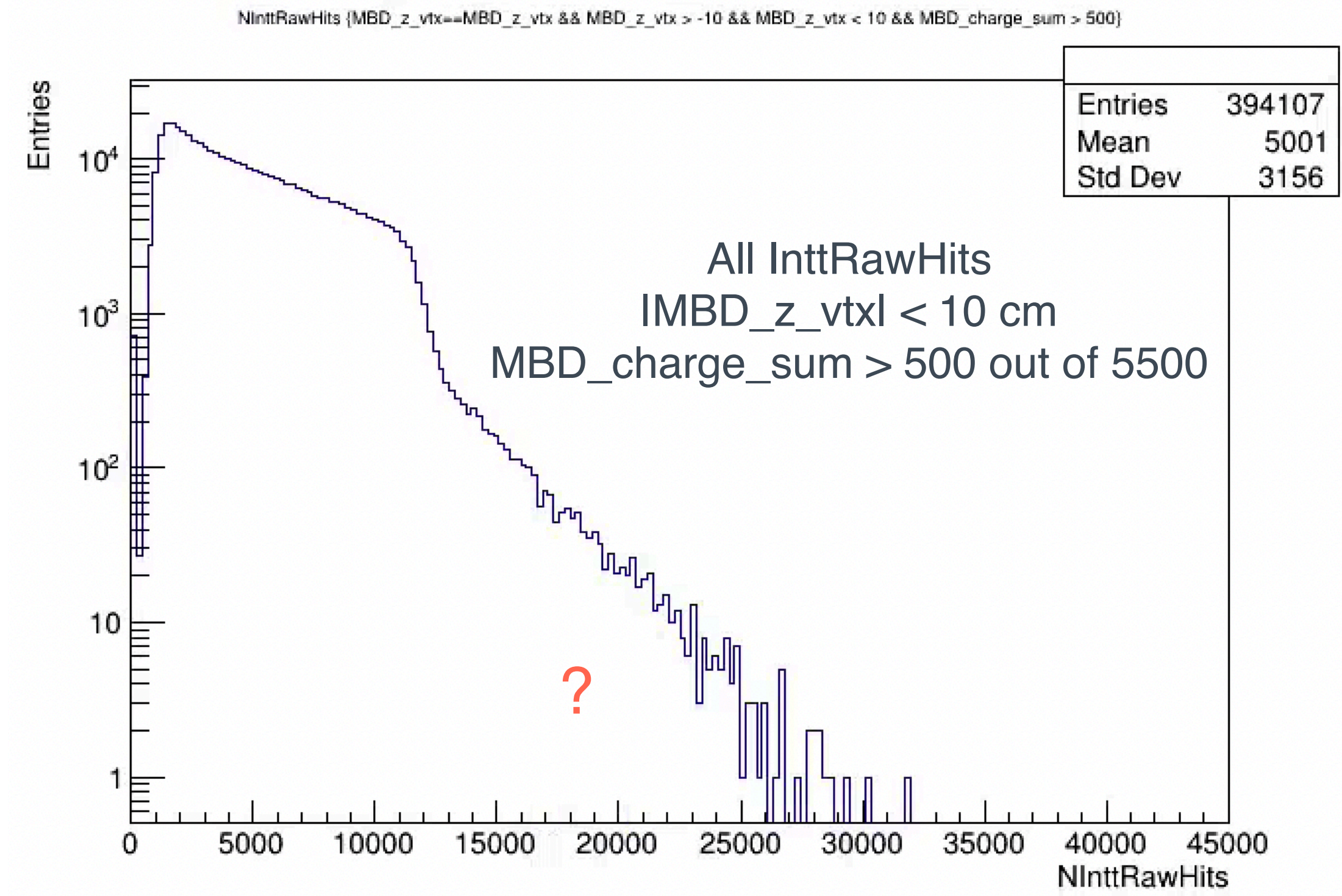
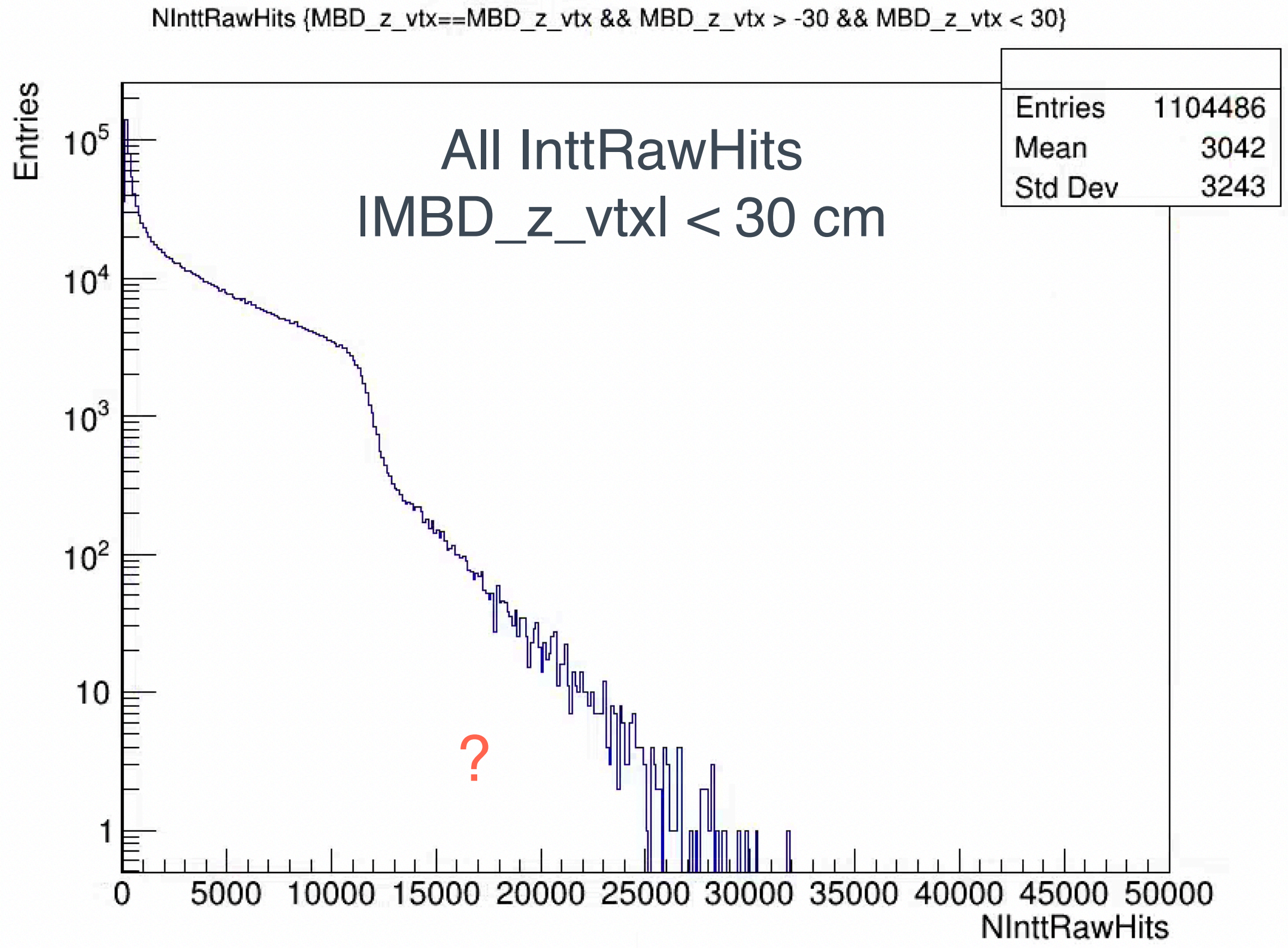
Run 54280  
Bad channel masked  
HitQA required  
Clone hit removal  
Trig\_MBDNS\_vtxZ30cm required



I so far not sure why there is a bump b/w nINTTRawHits 13k to 20k



# nINTTRawHit, to check the exceed

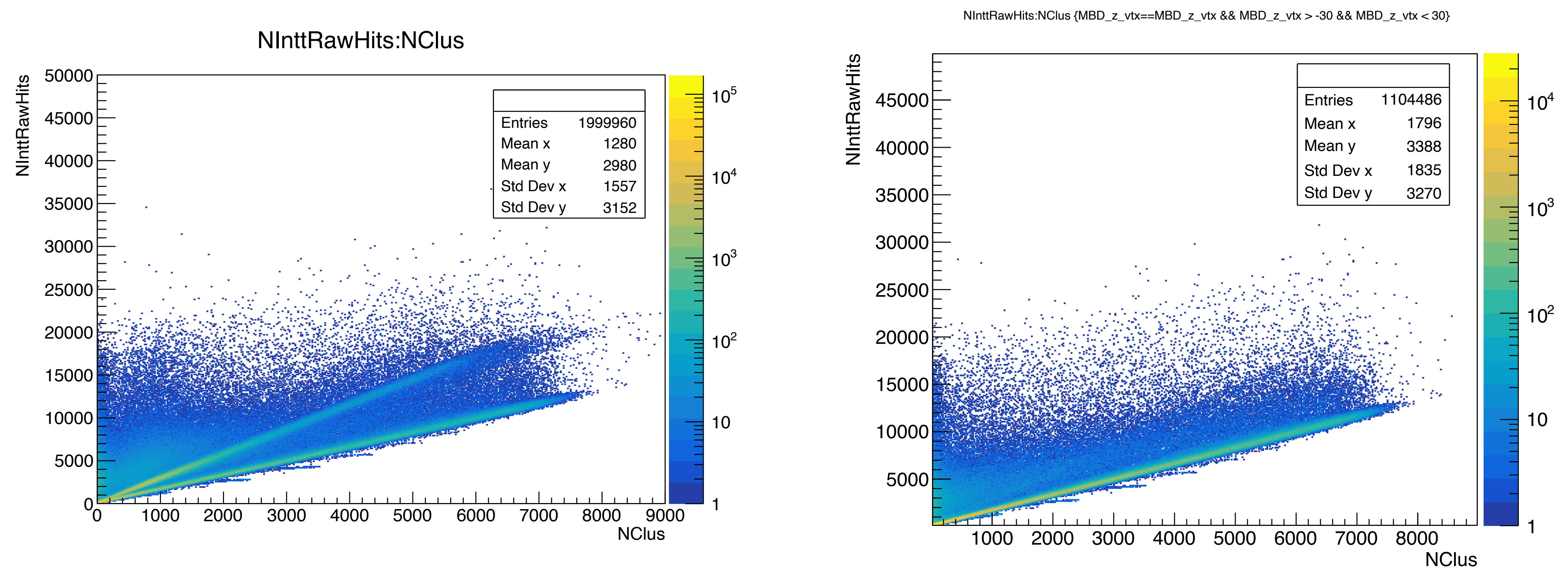


I so far not sure why there is a bump b/w nINTTRawHits 13k to 20k  
But it may not be urgent

# Correlation b/w nINTTRawHit and nClus



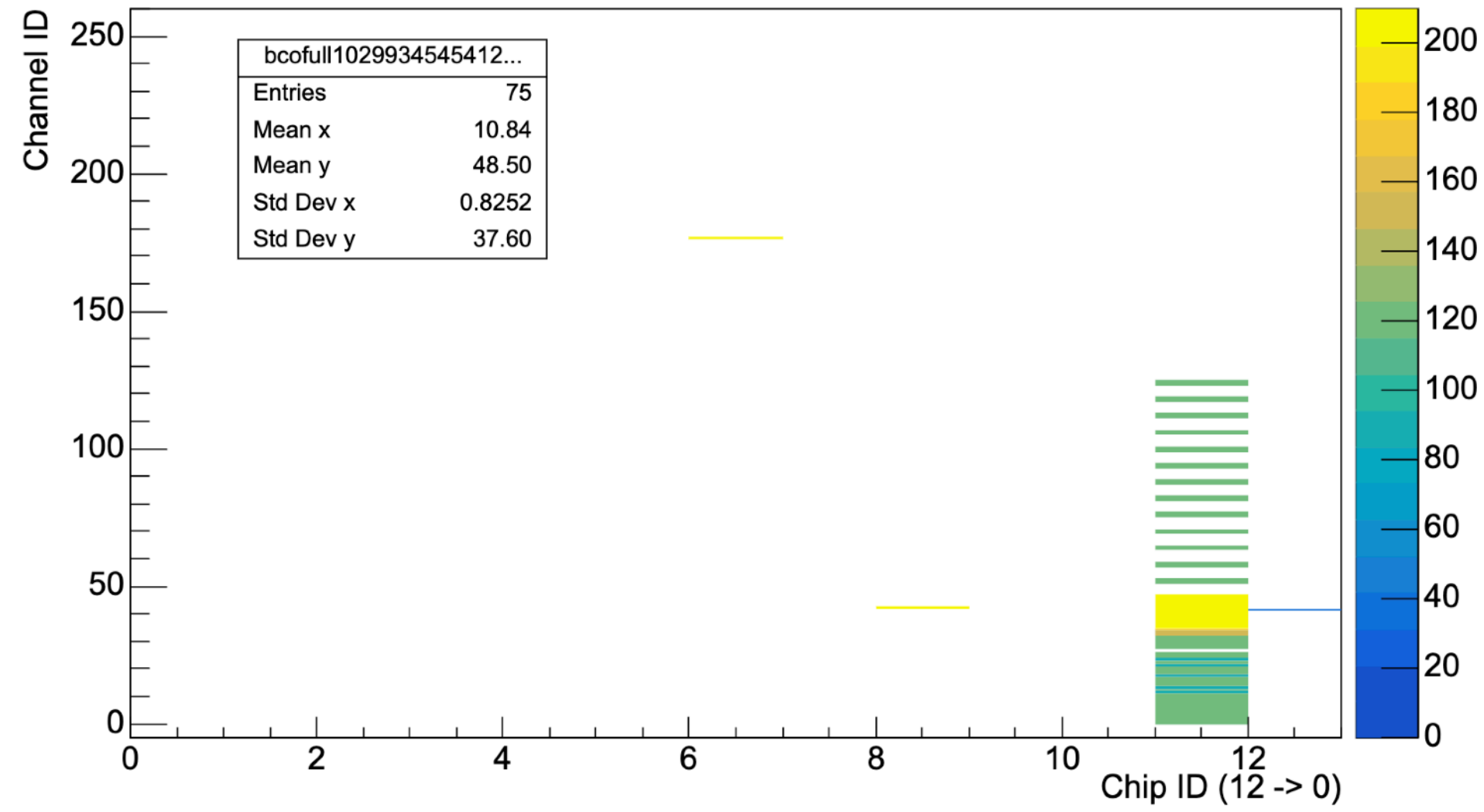
Run 54280  
All InttRawHit included  
Clustering in Z axis disable



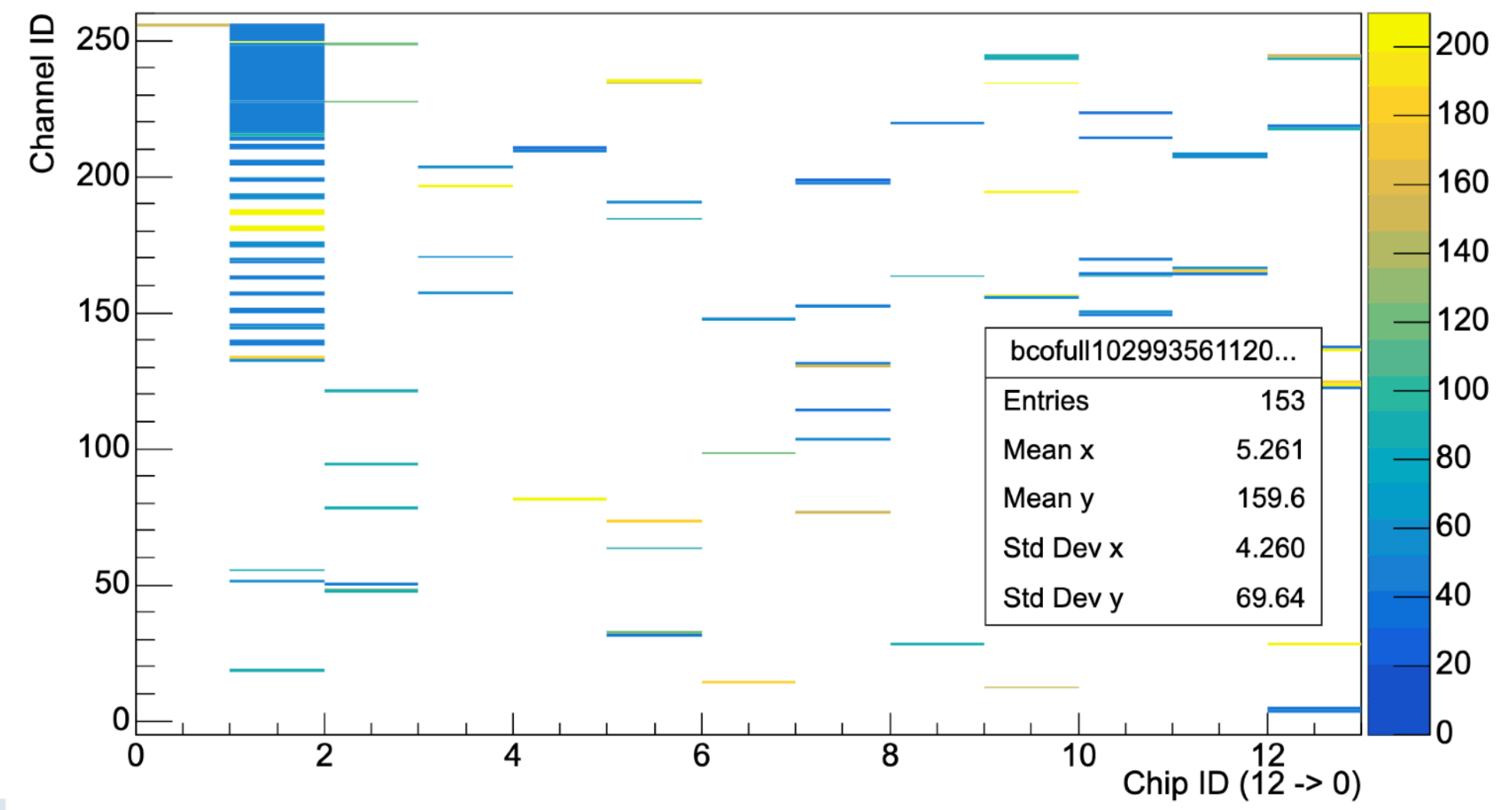
So far not sure why the vertex Z cut can eliminate the entire outlier branch  
The outlier groups are continuous in the Y axis view

# The patterns of the saturated chips

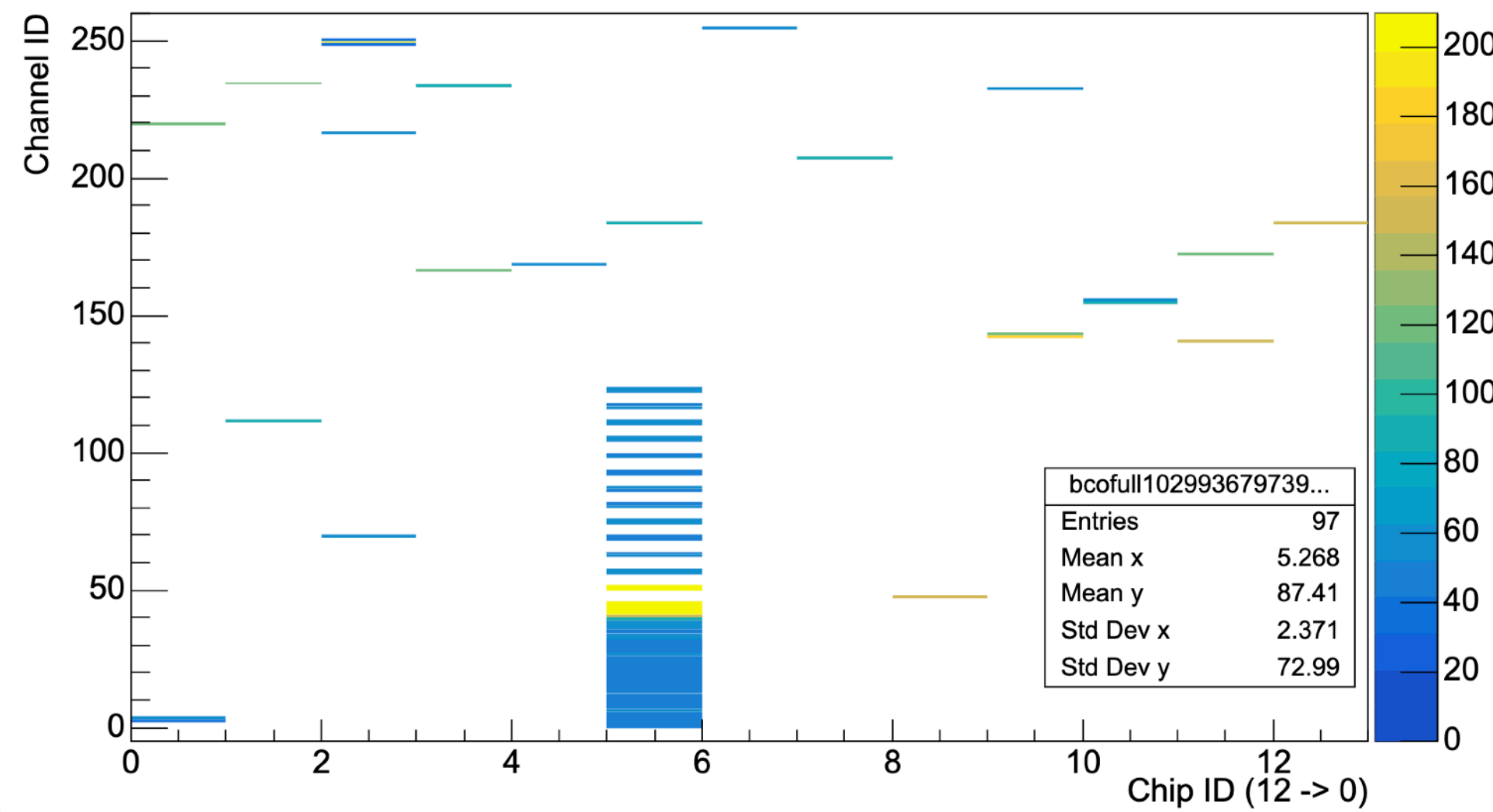
bcofull1029934545412\_F6\_Fch12



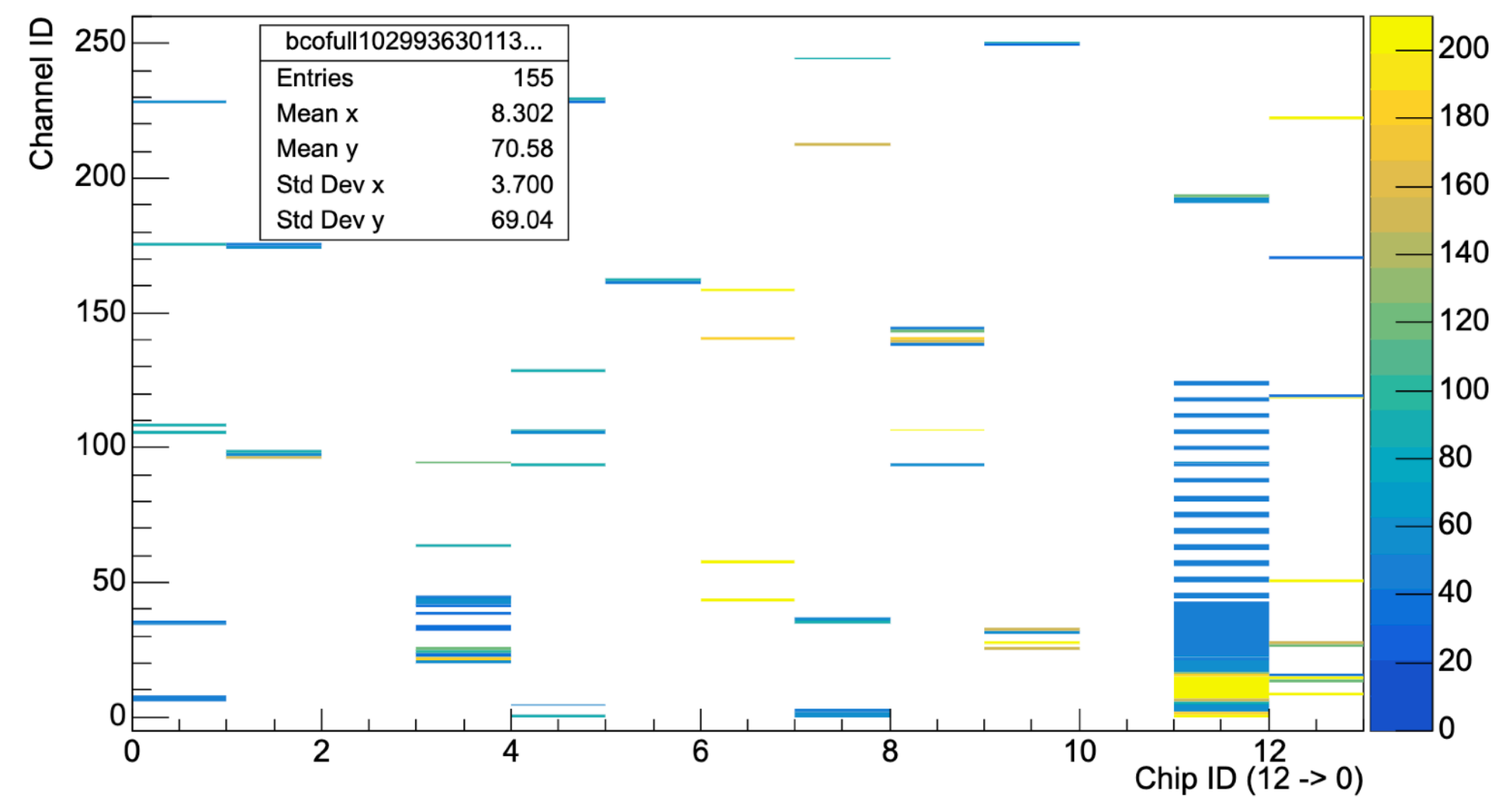
bcofull1029935611208\_F2\_Fch2



bcofull1029936797396\_F7\_Fch11

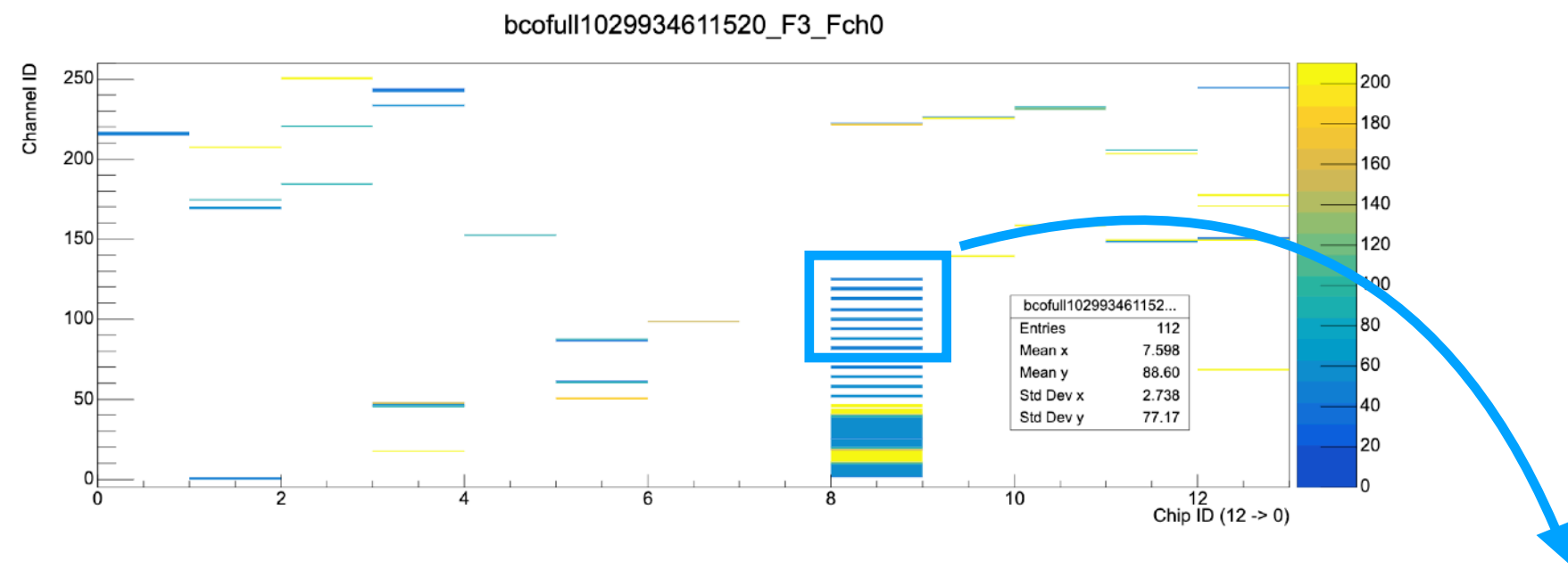


bcofull1029936301130\_F5\_Fch2

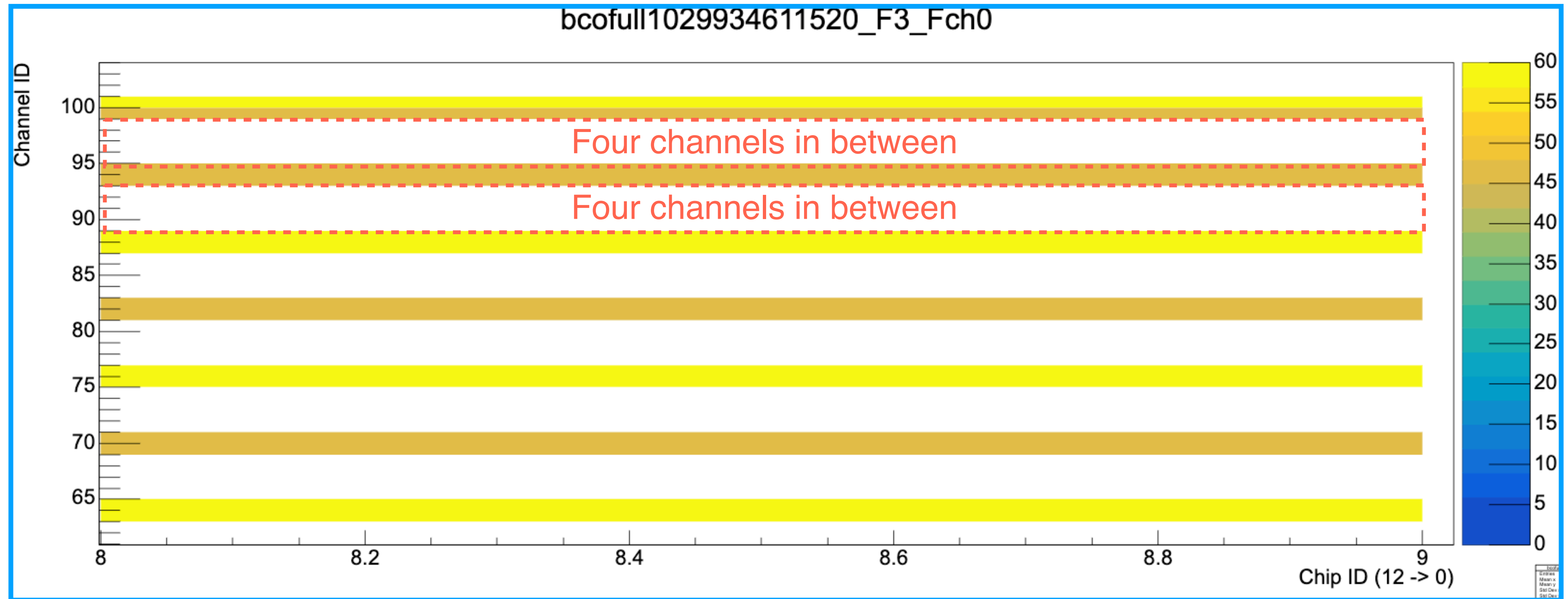


The pattern : big chunk + zebra crosswalk, and big chunk always closer to the edge

# The zebra crosswalk - normal



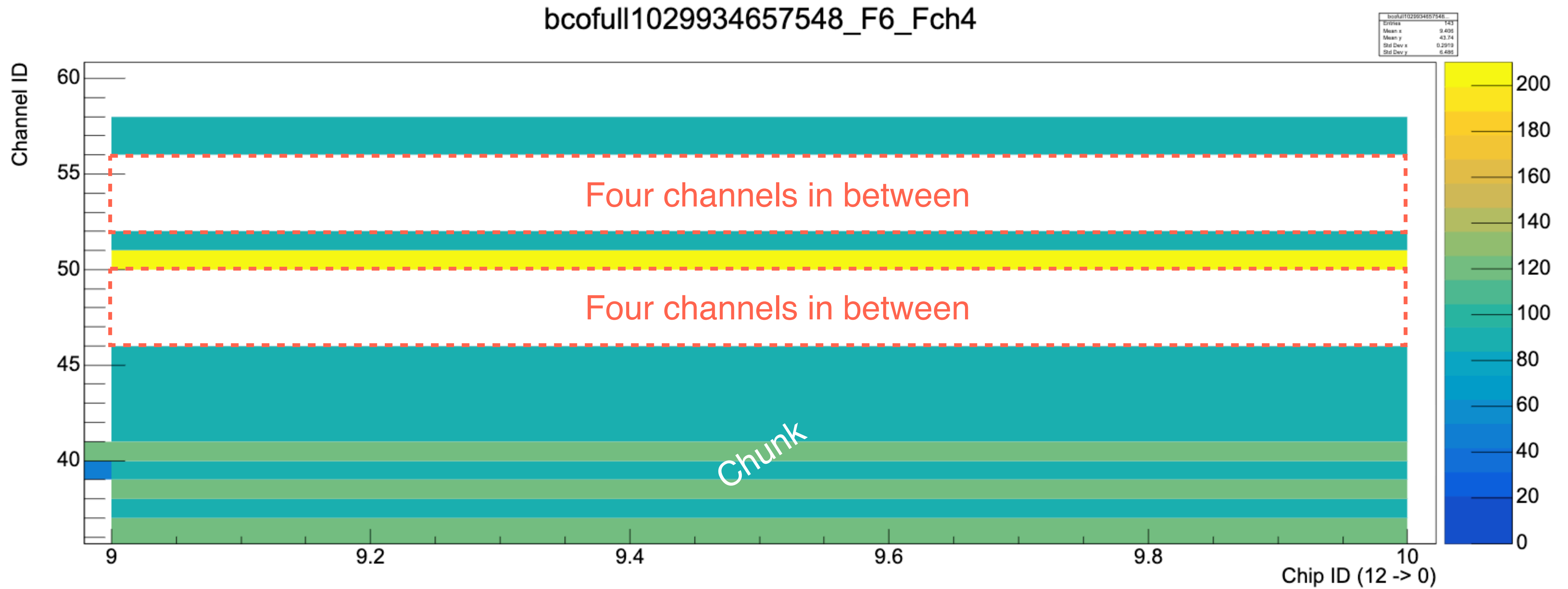
Two channels for each streak



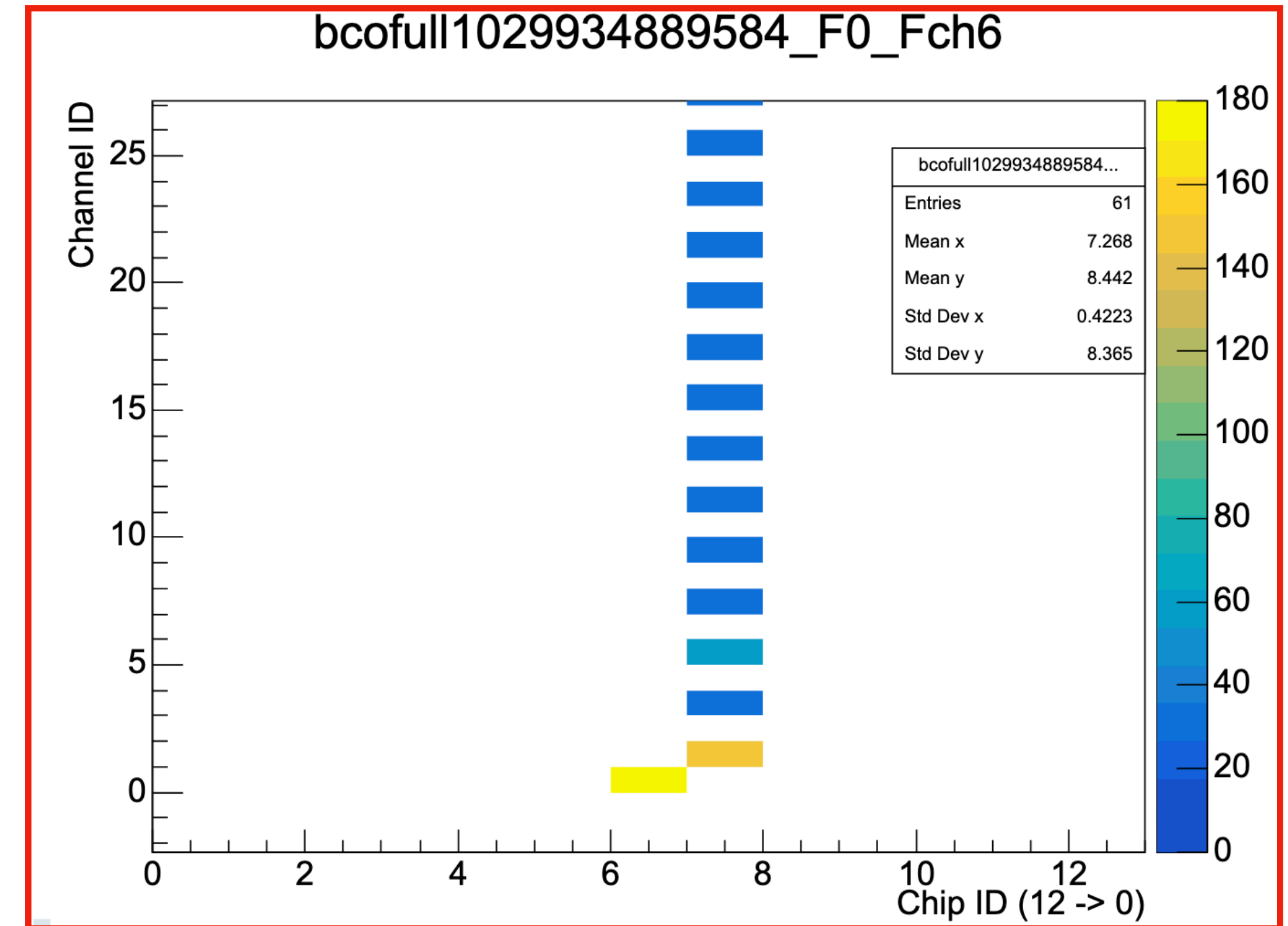
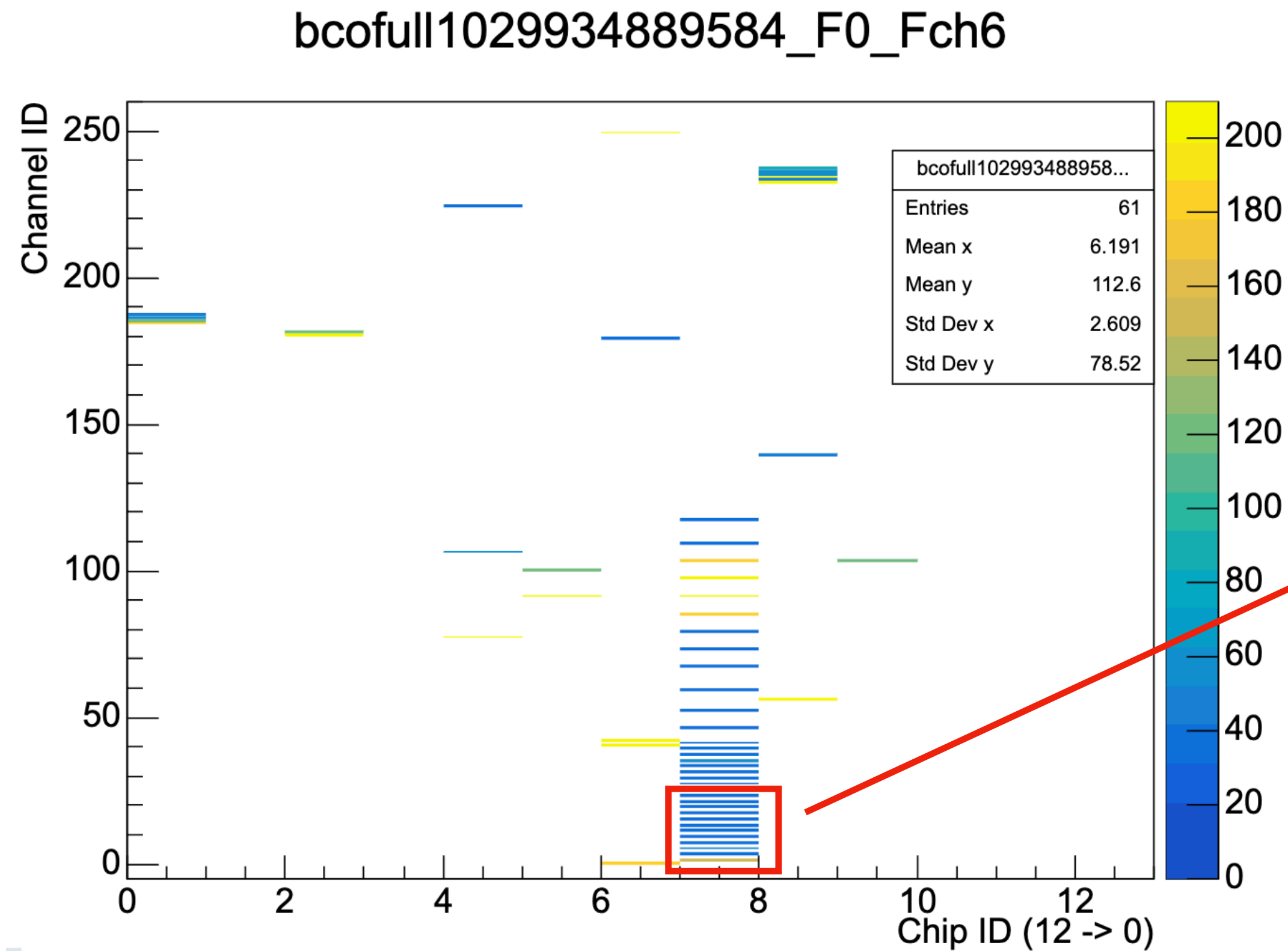
# B/w chunk and zebra crosswalk - normal



Run 54280



pattern of half-entry chip



Once again prove the working principle of the chip, it sends the hits in the alternative way

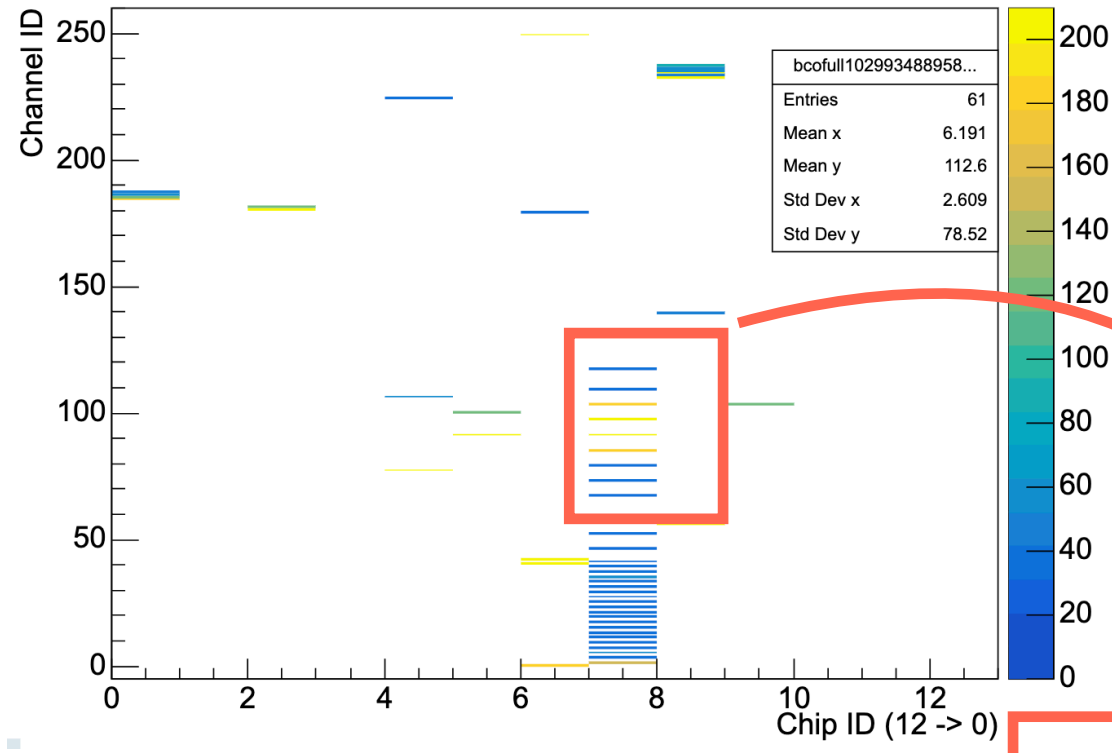
In this chip, all the even channels failed the signal transmission

It seems to be the case that one serial out takes care of even channels, one takes care of odd channels

# The zebra crosswalk - half entry

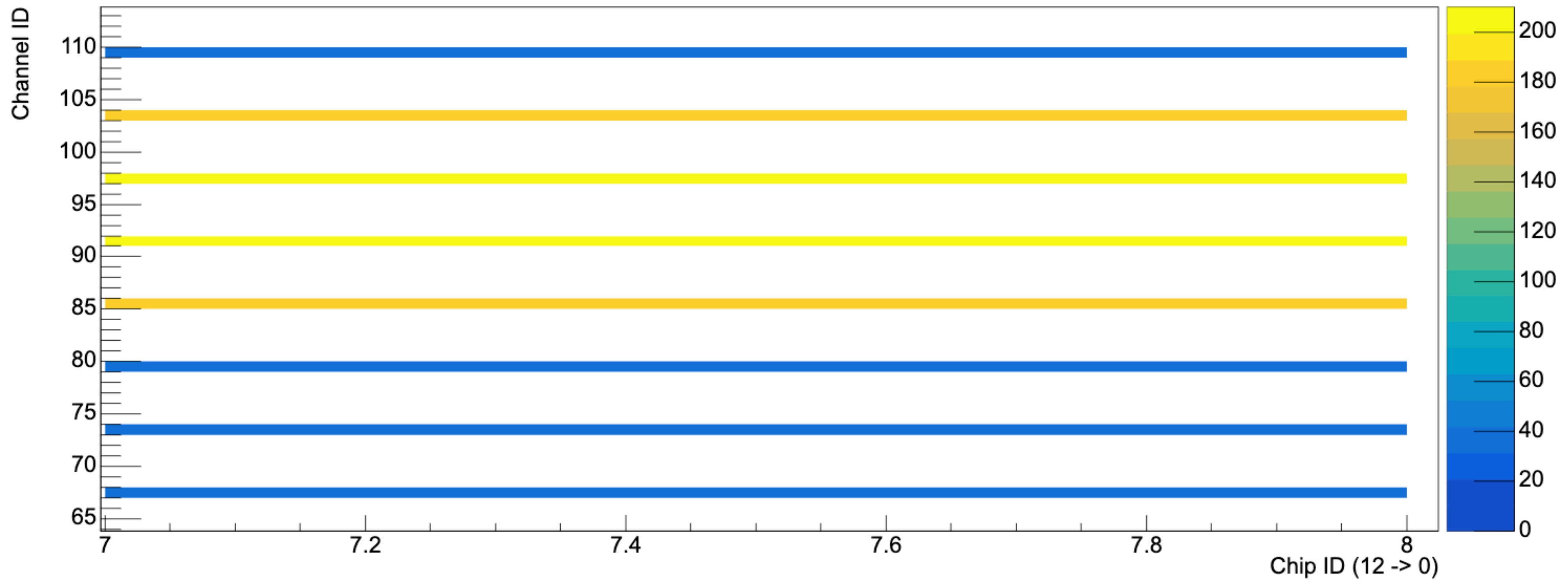


bcofull1029934889584\_F0\_Fch6



One channel for each streak

bcofull1029934889584\_F0\_Fch6

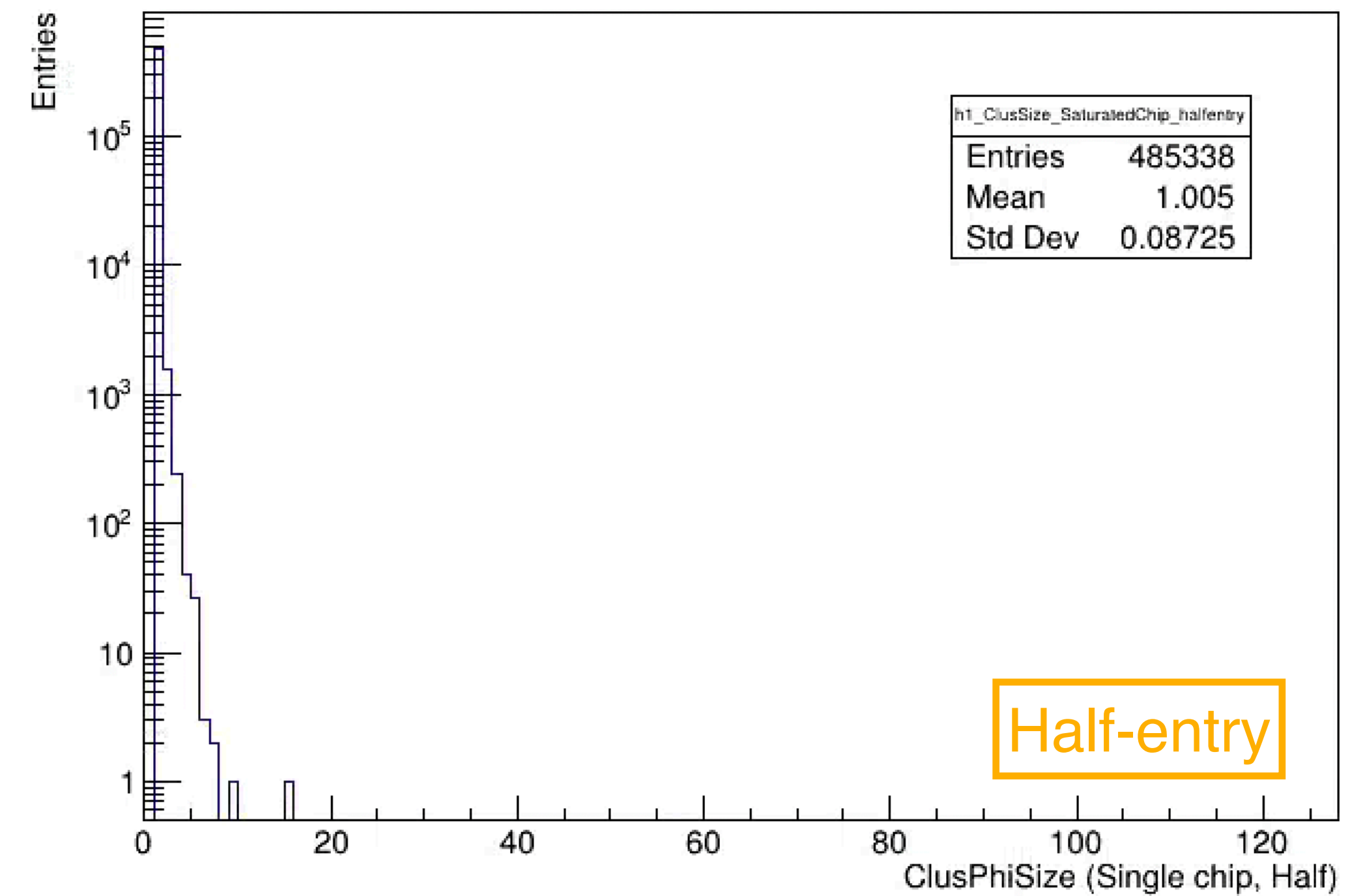
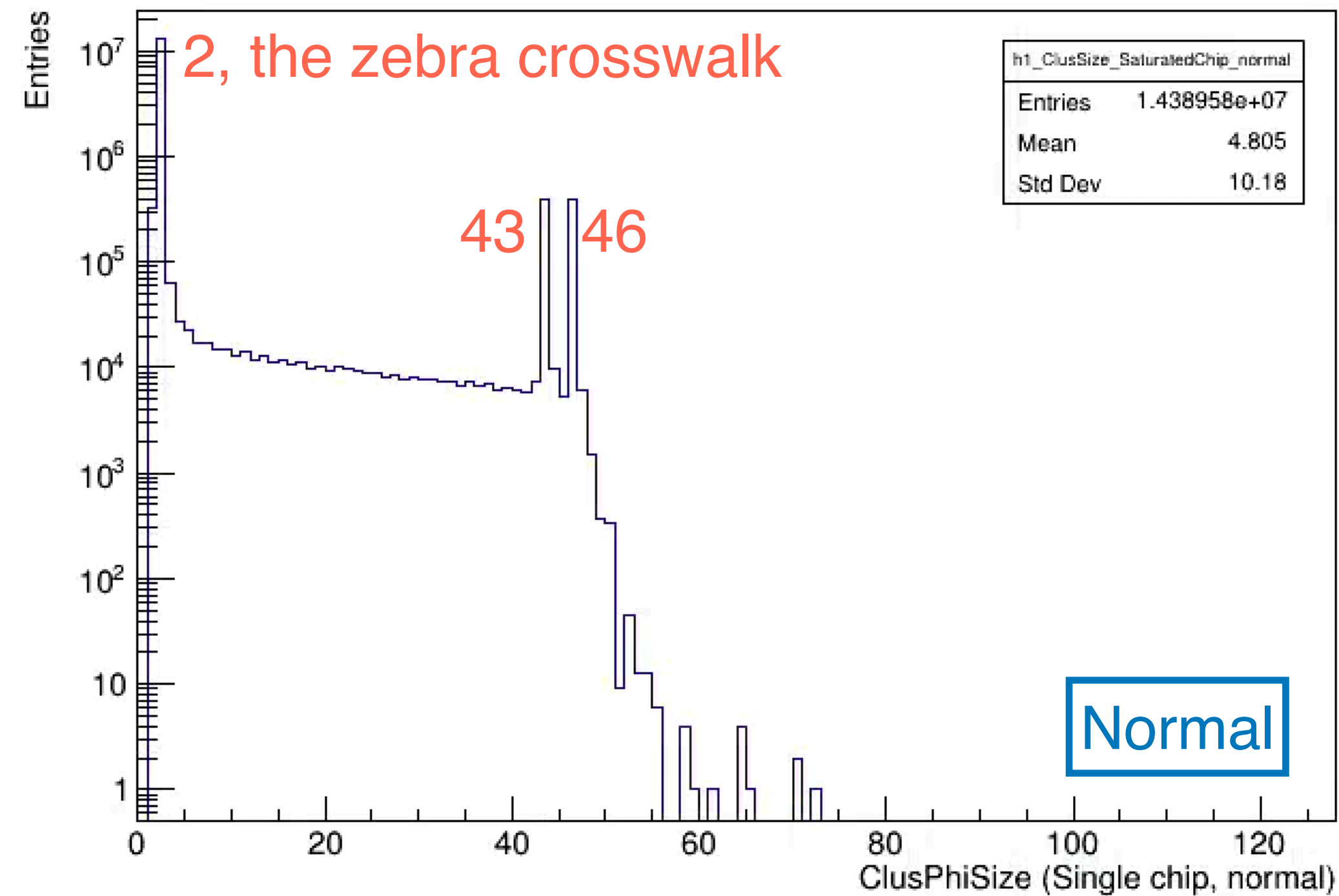


# Cluster phi size of the saturated chips

Private clustering (only do the clustering with single chip, 128 channels)

h1\_ClusSize\_SaturatedChip\_normal

h1\_ClusSize\_SaturatedChip\_halfentry

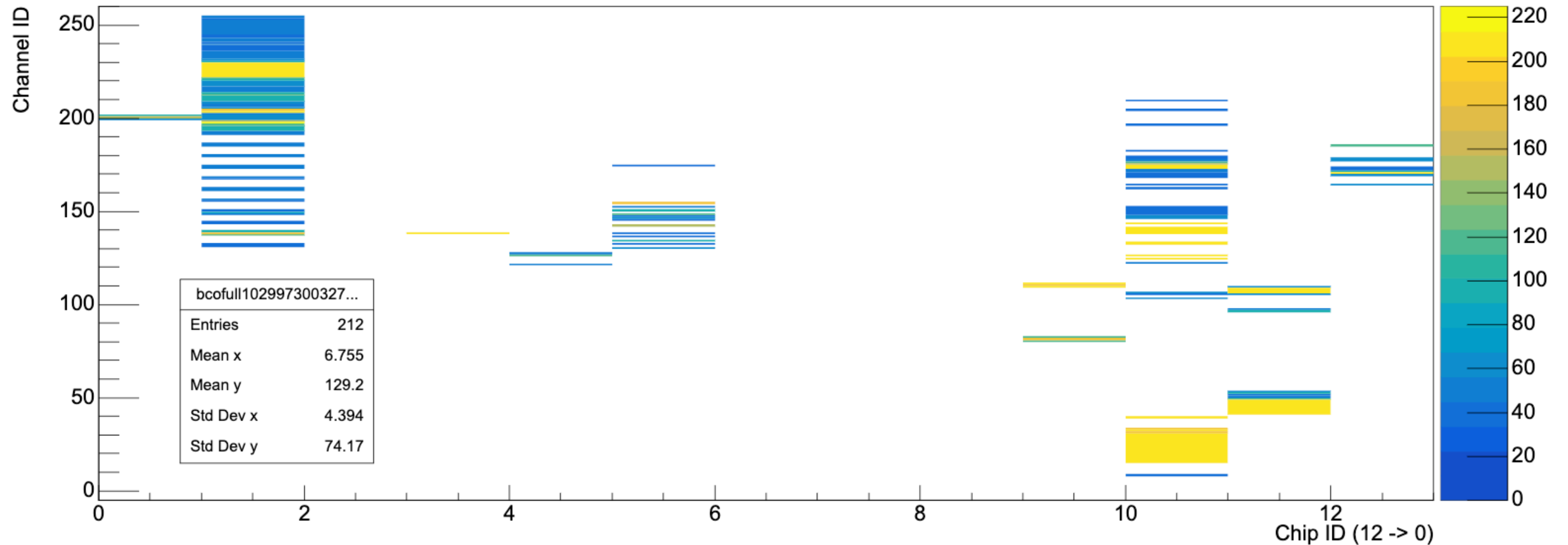


The big chunks in the hit maps of the saturated chips are with the phi size of 43 or 46 for most of the cases



# The patterns of the saturated chips

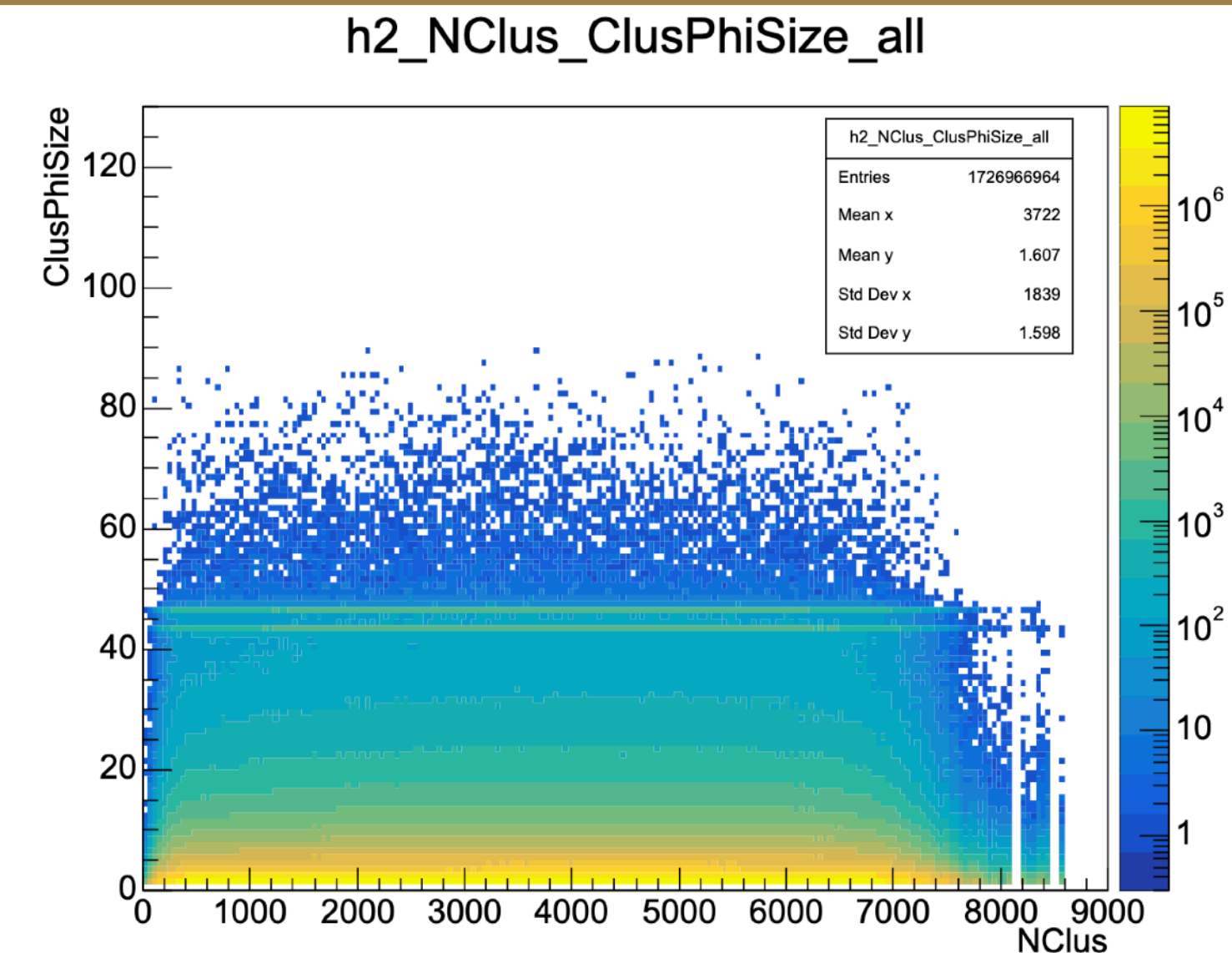
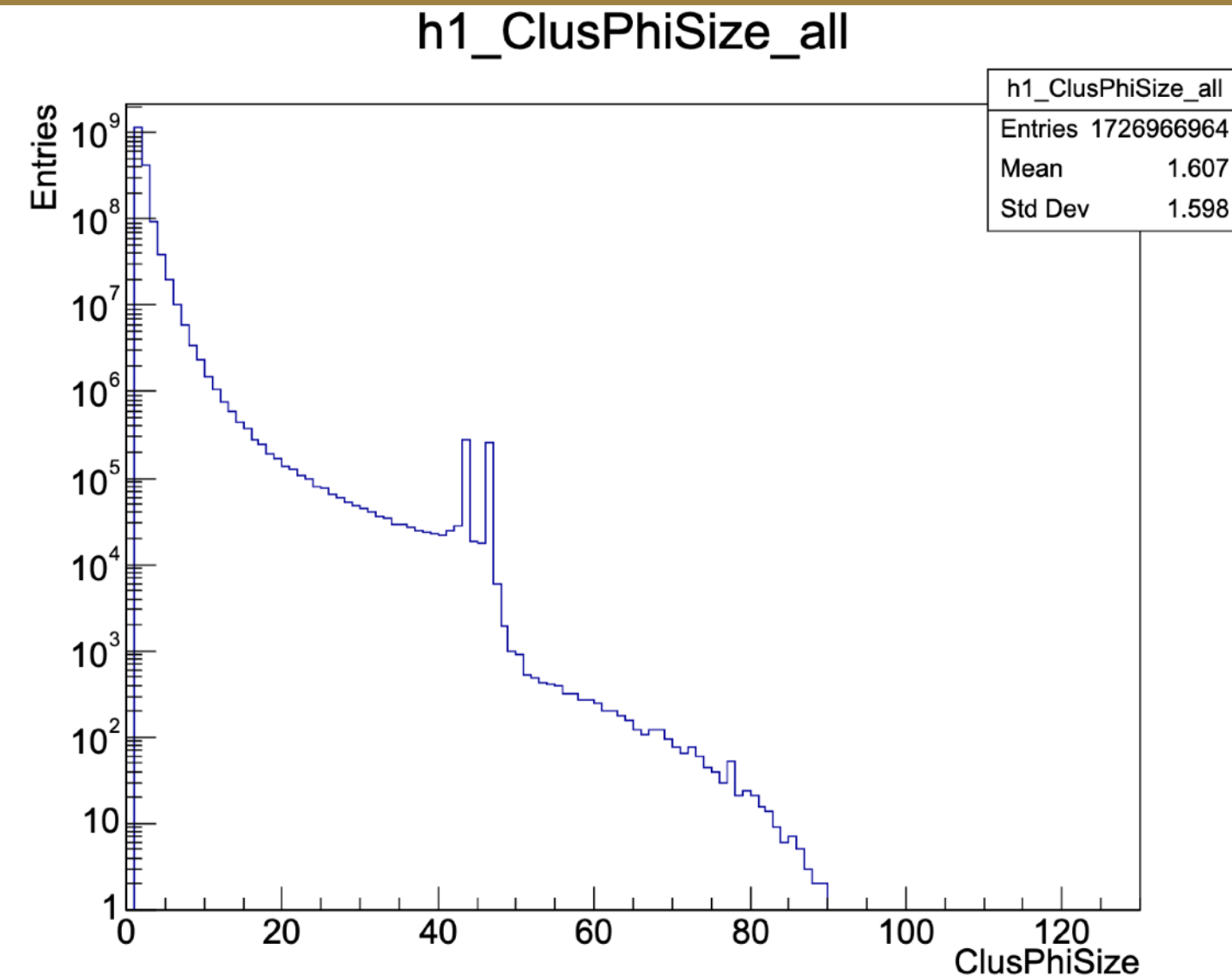
bcofull1029973003276\_F5\_Fch7



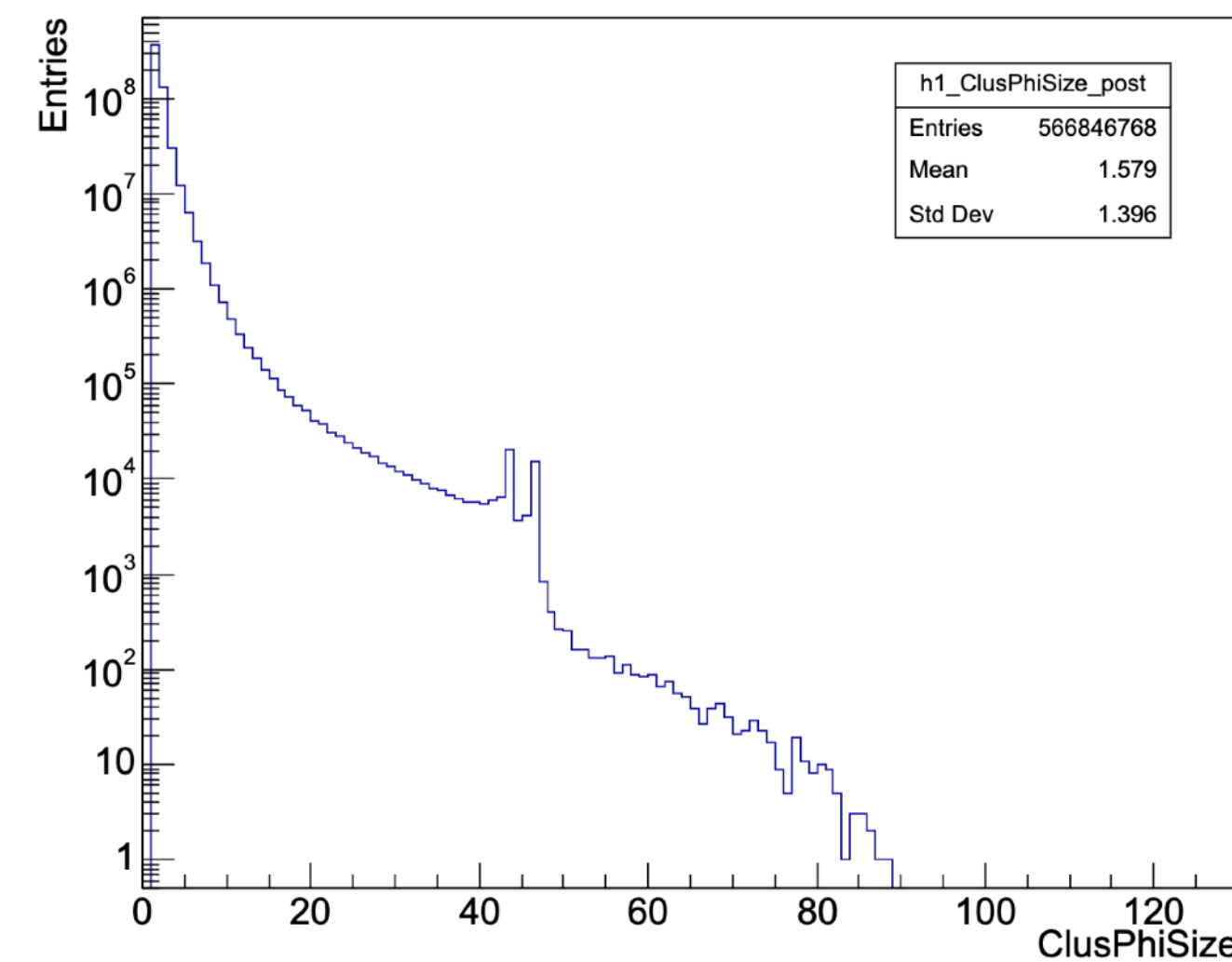
It's possible that the phi size of chunk can be larger than 46  
That chunk is with the size of 64

# Trial of event selection

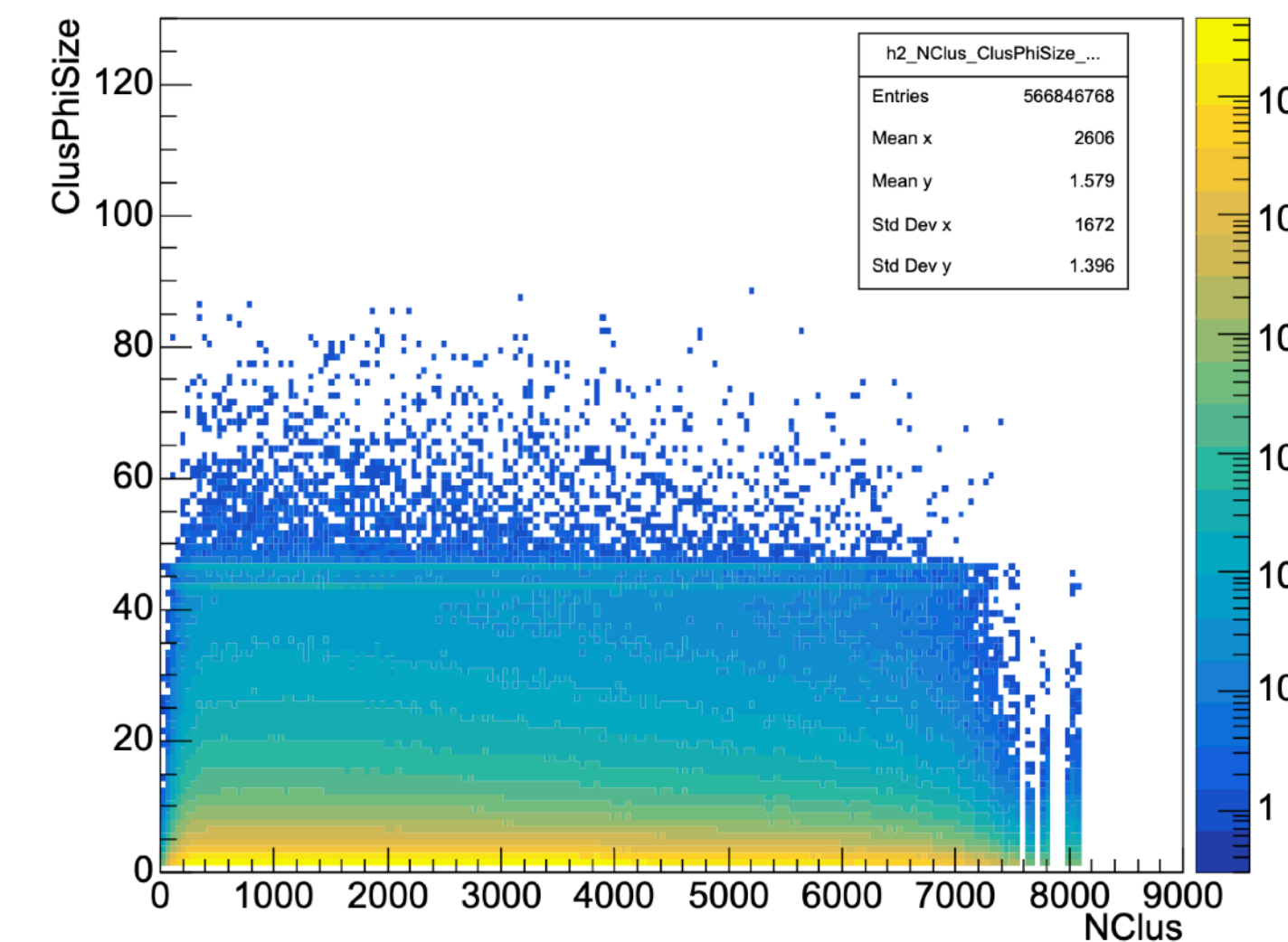
Inclusive



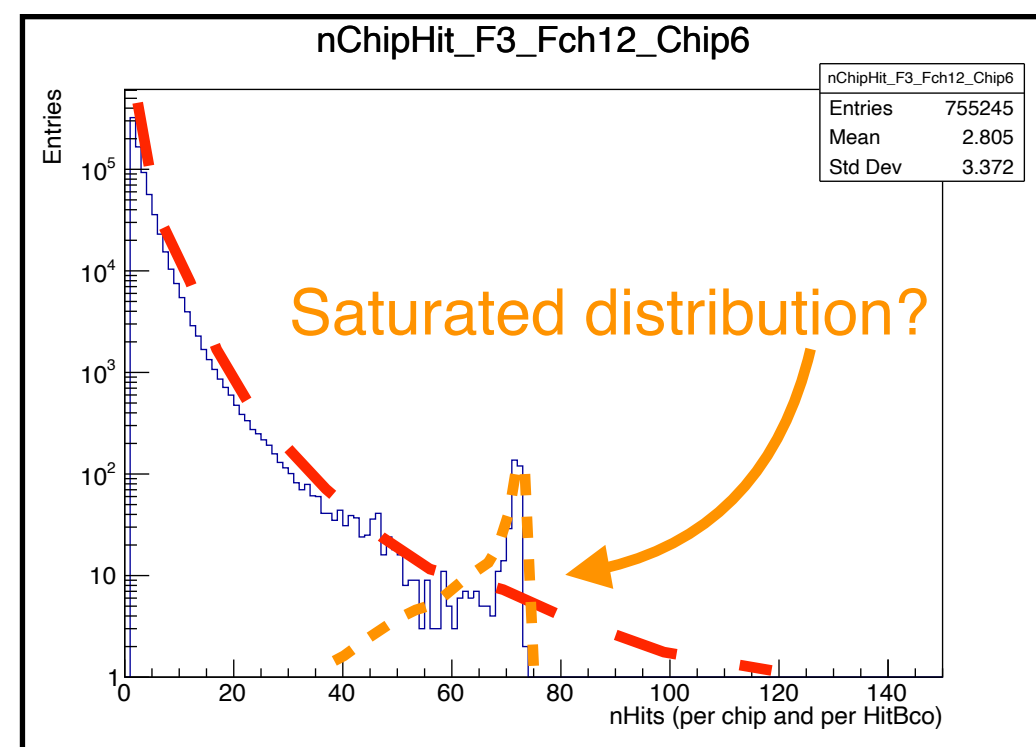
h1\_ClusPhiSize\_post



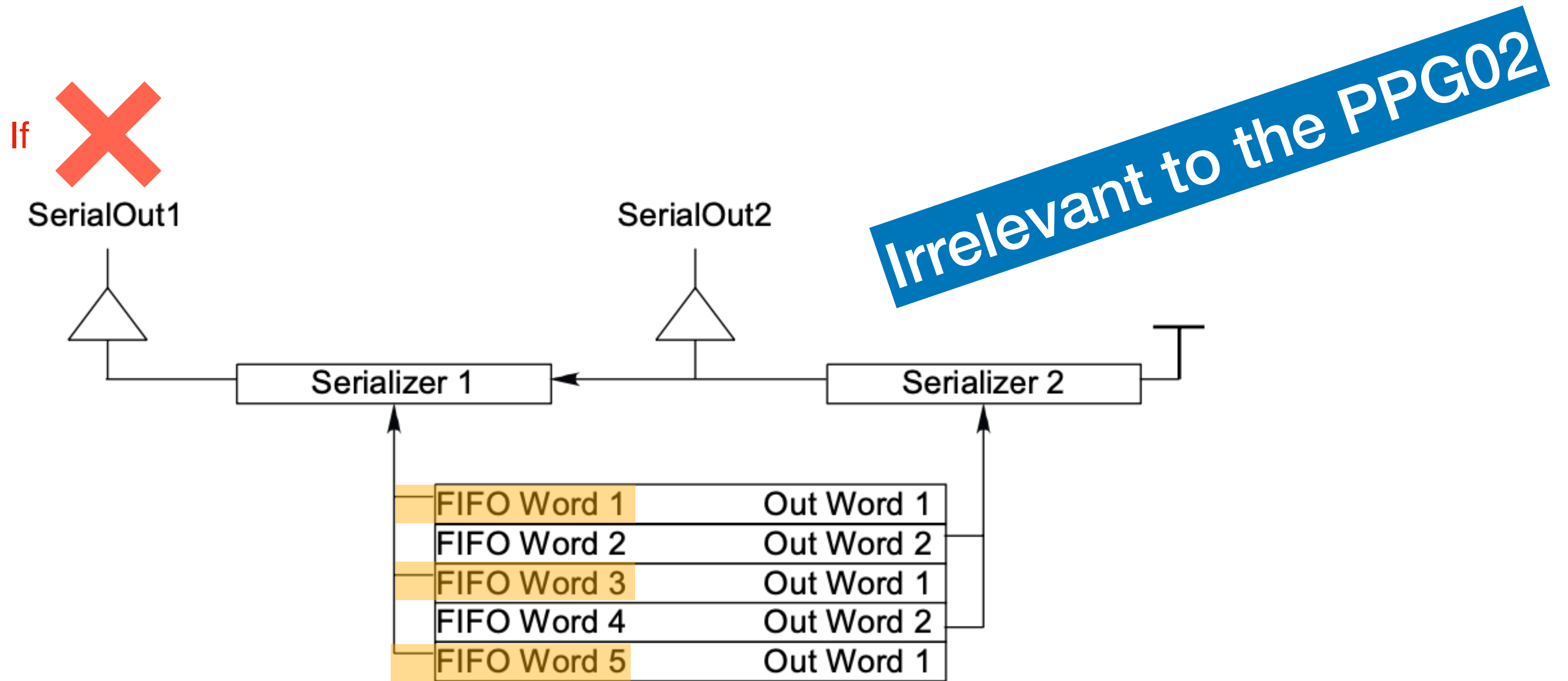
h2\_NClus\_ClusPhiSize\_post



As long as one chip classified as saturated chip, skip the event



The spikes become smaller, but still there. Might have the play with the zebra crosswalk if we really want to remove them



**Figure 16 - The Serializers**

If it's SerialOut1 dead, there is no hope to recover the half-entry chip by changing the `Digital Control setting`

## Phase Block

A primary requirement of the FPhx architecture is that it be able to read out within four beam cross-over periods an event that contained four hit strips. In other words, regardless of activity level, long latency cannot be tolerated. Hits must be sensed, amplified, discriminated, captured, sorted, serialized and read out of the chip. Moreover, the requirements do not allow for dead time, so if an FPhx chip receives an event in beam cross-over period “N”, it must be able to deal with an event in beam cross over period “N+1” and in beam cross-over period “N+2”, etc.

Requirement of PHENIX FVTX:  
read out 4 hits in 4 BCOs and no dead time

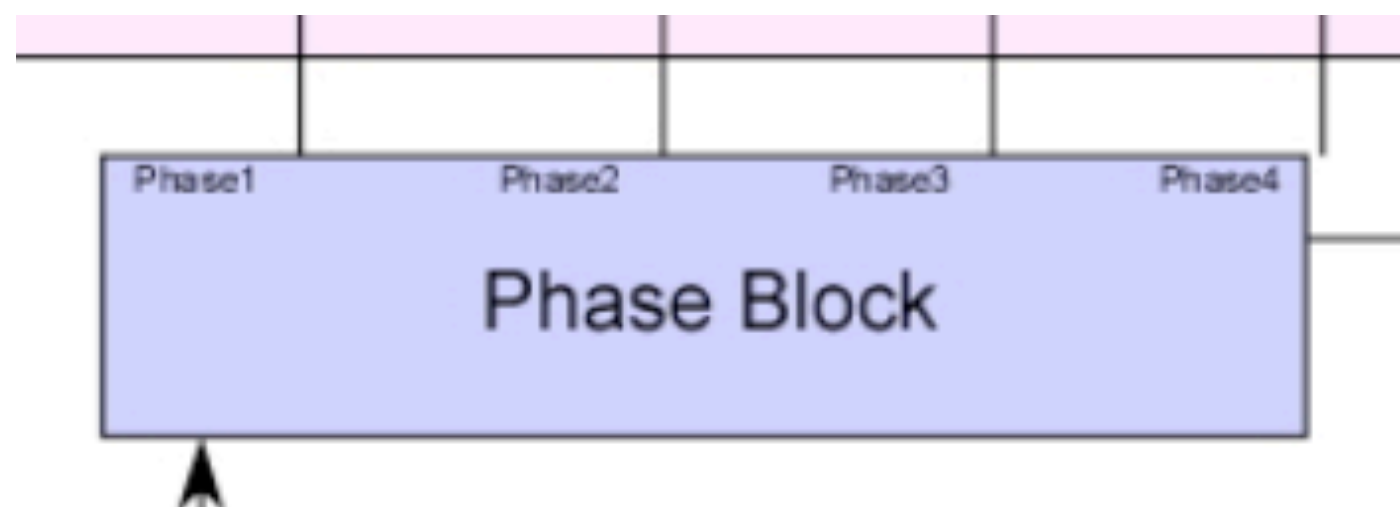
The procedures in FPFX chip: sensed → amplified → discriminated → acquisition → sorted → serialized → read out  
128 channels in parallel

If it's possible, it must be good to have the nhits and cluster size distributions of PHENIX FVTX 🙏🙏🙏

The procedures in FPHX chip: sensed → amplified → discriminated → acquisition → sorted → serialized → read out

128 channels in parallel

These twin requirements of low-latency and zero dead time give rise to the notion of phase architecture. During any given phase, amplification, discrimination, acquisition, sorting, serialization and output must happen. However, amplification, discrimination and acquisition are happening for hits that are occurring in *this* phase. Sorting is happening for hits that occurred in the last phase. Serialization and output are happening for hits that occurred at least two phases ago.



The requirement of “four-hits-in-four-beam-crossings” suggests that a cyclic progression of four phases (Phase 1 → Phase 2 → Phase 3 → Phase 4 → Phase 1 → etc) could be used as the cornerstone of an architecture built to work for Phenix. During

This approach would require redundant circuitry. For example, there would be Phase 1 acquisition circuitry and Phase 2 acquisition circuitry and Phase 3 acquisition circuitry and Phase 4 acquisition circuitry. Moreover, there would have to be additional circuitry to manage the flow of data through these different phases. However, this approach would enable the job to be done without requiring excessive speed and the consequent power that would require.

Though I didn't find the redundant circuitry in the block diagram

# FPHX chip manual - phase architecture



The procedures in FPHX chip: sensed → amplified → discriminated → acquisition → sorted → serialized → read out

In front end (sensed → amplified → discriminated)

Phase splitting occurs (acquisition → sorted)

128 channels in parallel (Acq: check mask)

unit : need to be confirmed

Time →

		BCO -2	BCO -1	BCO 0	BCO 1	BCO 2	BCO 3
Phase 1	ampl, disc, acqui			hit1_bco0			hit1_bco1
	sort		hit1_bco-1				
	serialization, output	hit1_bco-2					
Phase 2	ampl, disc, acqui						
	sort				hit1_bco0		
	serialization, output			hit1_bco-1			
Phase 3	ampl, disc, acqui						
	sort				hit1_bco1		
	serialization, output					hit1_bco0	
Phase 4	ampl, disc, acqui						
	sort					hit1_bco1	
	serialization, output						

In given phase, all the steps must happened, dealing with different hits from different phases

**Larger hits pose a bigger problem.** Of necessity, amplification and discrimination must occur in parallel for all 128 channels. Therefore, larger events will have no impact on this activity. Acquisition, even though it has four phases, also occurs in parallel for all 128 channels. **Sorting, serialization and output, however, will be impacted by the size of the event.** Larger events will take longer to sort, longer to serialize and longer to output. Serialization and output overloading can be mitigated by FIFOs which can be filled by large events and drained during empty or low occupancy beam cross-over periods. This could have an impact on the “four-hits-in-four-beam-crossings” requirement. For example, **if there are several large events in a row**, the first large event might output four hits in four beam-crossings, but, because of the FIFOs, the second and **subsequent hits might not get their first four hits out in four beam-crossings.**

I don't see the statement that one chip can only handle 4 hits in one event

Unfortunately, very little can be done about sorting (also known as zero suppression). Larger events will take longer to sort. Therefore, some mechanism must be in place to halt the advancing of the phase in the event that sorting cannot be completed in the required time. This is the principle responsibility of the Phase Block.

The Phase Block maintains the eight-bit beam cross-over counter or BCO Counter. This is the time stamp. It advanced on the rising edge of each BCO clock. The Phase Block also maintains the phase state machine which advances on the rising edge of each BCO clock *provided* that the sorting has been completed. If the sorting has not been completed, the phase does not advance, and the acceptance of further hits by the FPhx chip is suppressed until the phase can finally advance. Finally, the Phase Block operates as indirect addressing logic relating a particular phase to a particular time stamp so that when data is output, the hits are associated with the correct time stamp regardless of how many times the phase advance was blocked.

Inefficiency?

I don't see the statement that one chip can only handle 4 hits in one event



# The logic of chip (tentative, conceptual)

## The FPHX chip manual

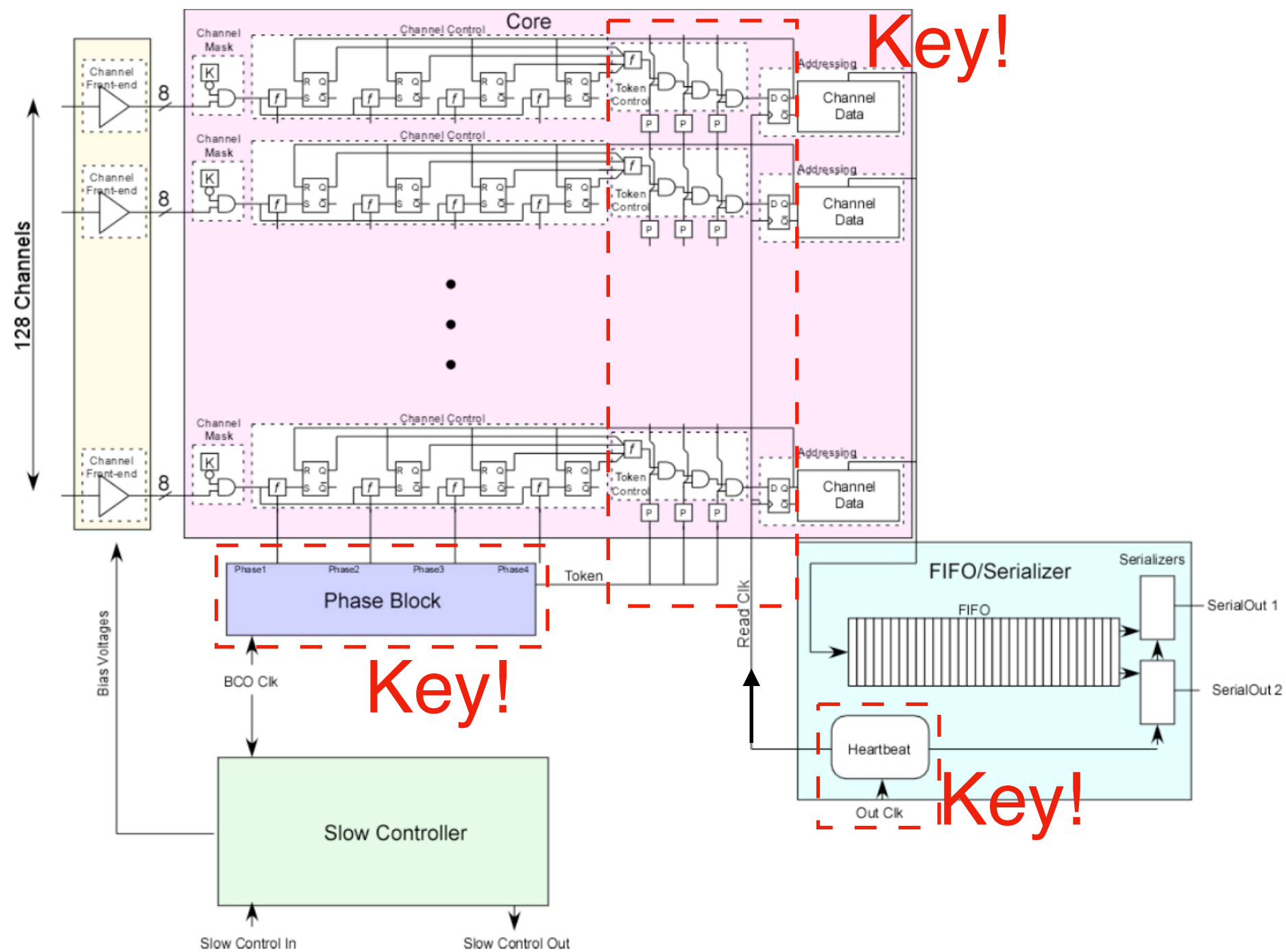


Figure 2 - The FPhx Block Diagram

\*Key: relevant parts for this issue but not fully understood

### Token Logic

The Token Logic divides the chip into individual channels, blocks of eight channels and banks of 32 channels. It performs this division with three token tiers. The Tier 3 token selects which bank of 32 has access to the bus. The Tier 2 token selects which block of 8 has access to the bus. Finally, the Tier 1 token selects which individual channel has the bus. A channel can only have the bus if it has all three tokens AND a hit to output.

## Tentative conceptual conclusion

(The followings are based on the observations and limited understanding in the FPHX chip manual)

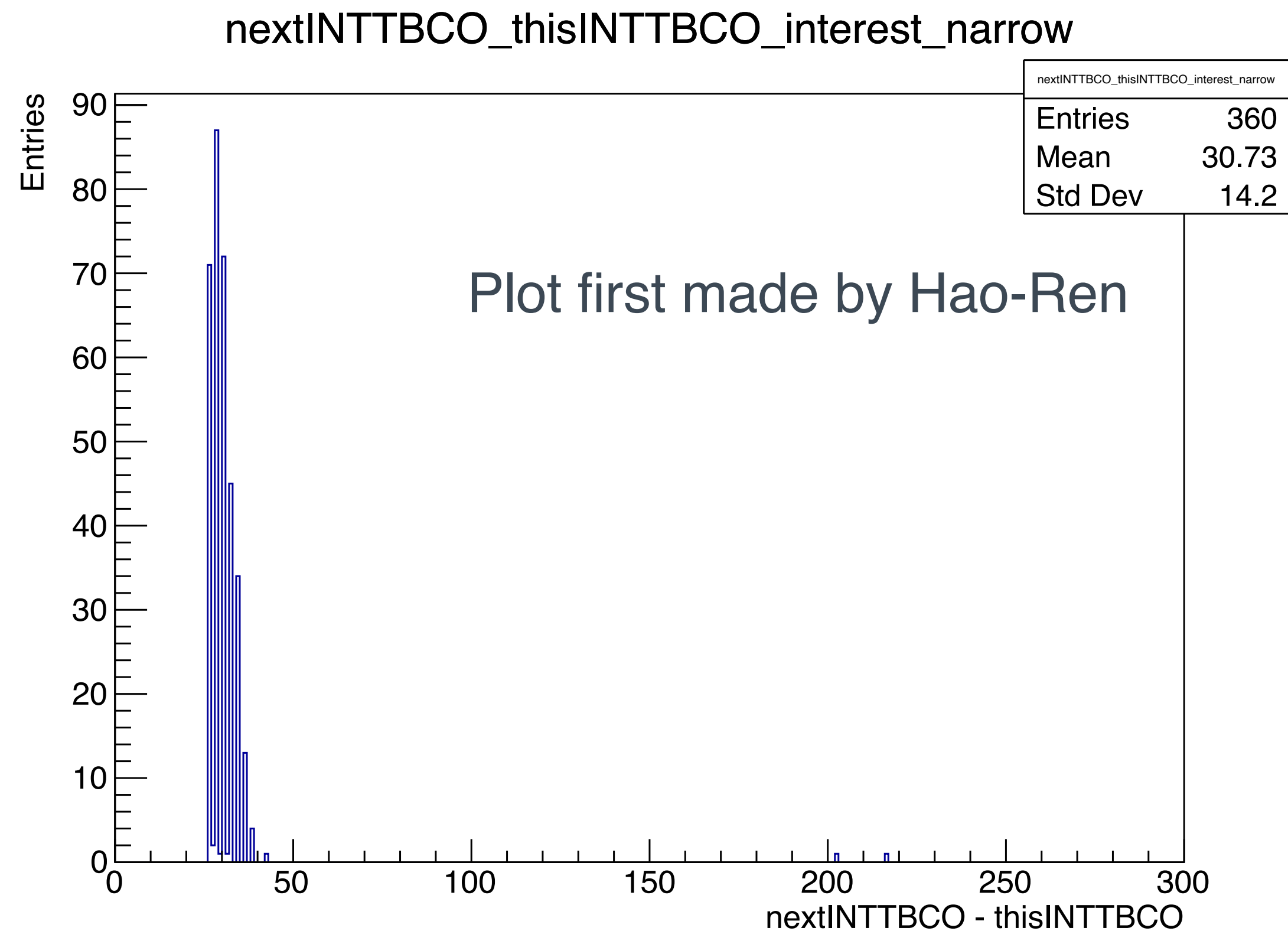
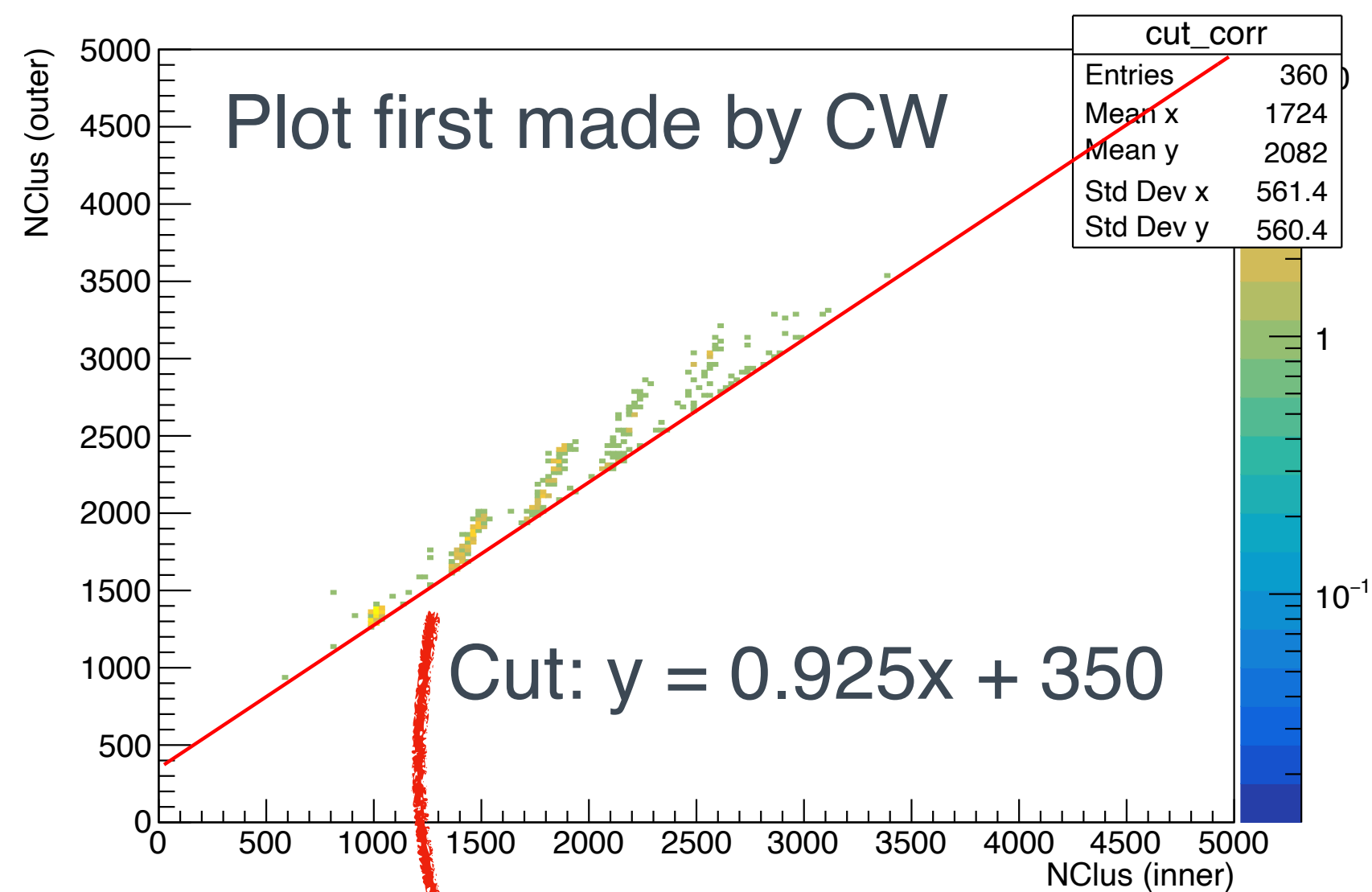
- The sequence of signal processing does not follow the channel ID
  - The channels are categorized into 4 groups (**phases**)
  - The phase where hit is assigned to is based on the phase state
- It seems that the sequence of data transmission of each phase is from the hit with smaller channel ID to the that of large channel ID
- It seems that the data transmission of the 4 phases follows some order (cannot be all the phases at once)
  - It's partially because of the FIFO
  - In one period of time (say 1 BCO), one chip can send out two hits by two data lines, `SerialOut1` and `SerialOut2`

### The hypothesis:

- 2 out of 4 phases successfully send out all the hits to the FELIX in time
  - Result in the structure of zebra crosswalk
- Rest 2 phases can only send of partial hits to FELIX in time
  - Upon some channel ID (some where channel ID 43 or 46), it's already out of time (open\_time). The FELIX servers therefore reject the hits
  - Result in the big chunk **Irrelevant to the PPG02 (maybe?)**

The direct question would be, how long does it take to process one hit, and in what sequence the hits are sent to ROC?

# Event of interest (EOI)



The very next events of the EOI are very close to EOI in time wise

Hypothesis: Hits in FELIX been assembled with INTTheader (INTT\_bcofull) and sent out to the down stream. Since FELIX receives new trigger, the previous INTT\_bcofull is overwritten. The hit assembly continues, but with the new INTT\_bcofull

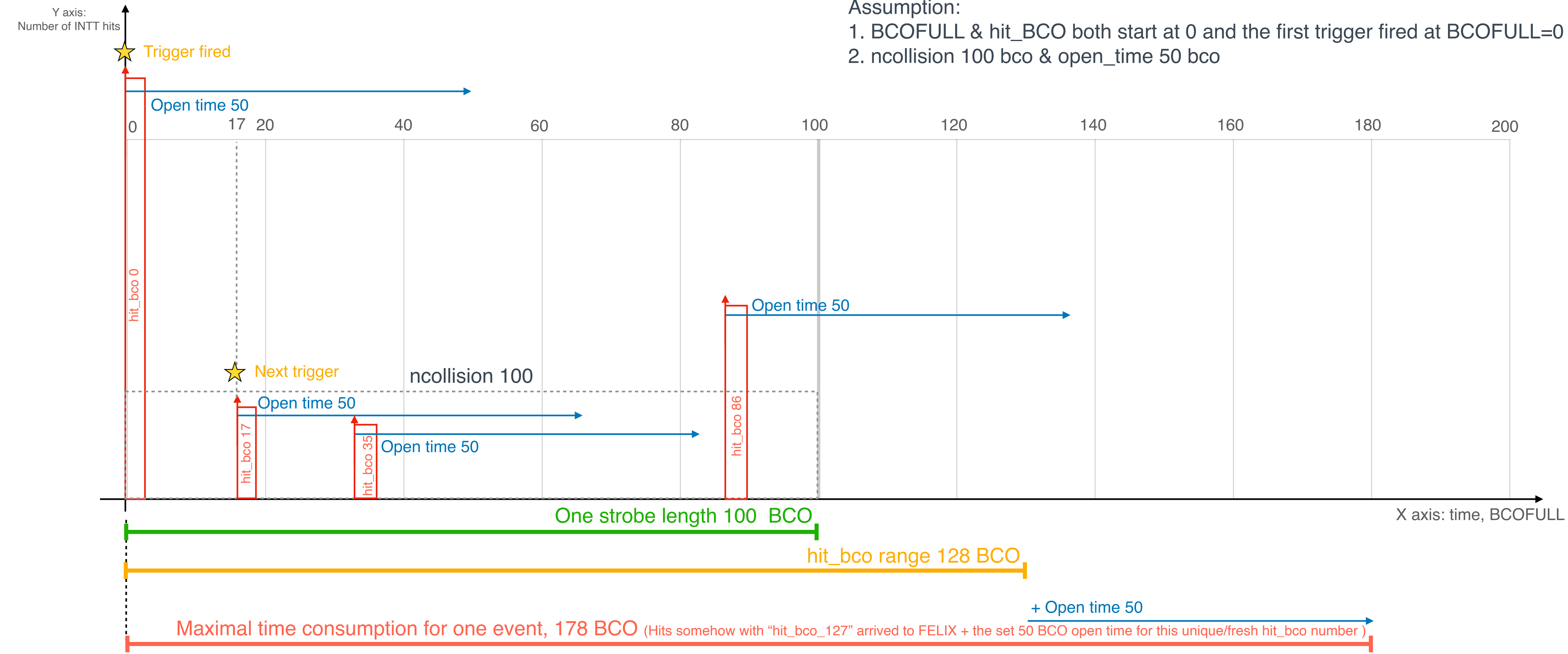
Can we probably just have a simple "BCOFULL\_diff" cut?

Note: the hit transmission from chip to ROC: 1 hit / 1 bco

In single event

Assumption:

1. BCOFULL & hit\_BCO both start at 0 and the first trigger fired at BCOFULL=0
2. ncollision 100 bco & open\_time 50 bco



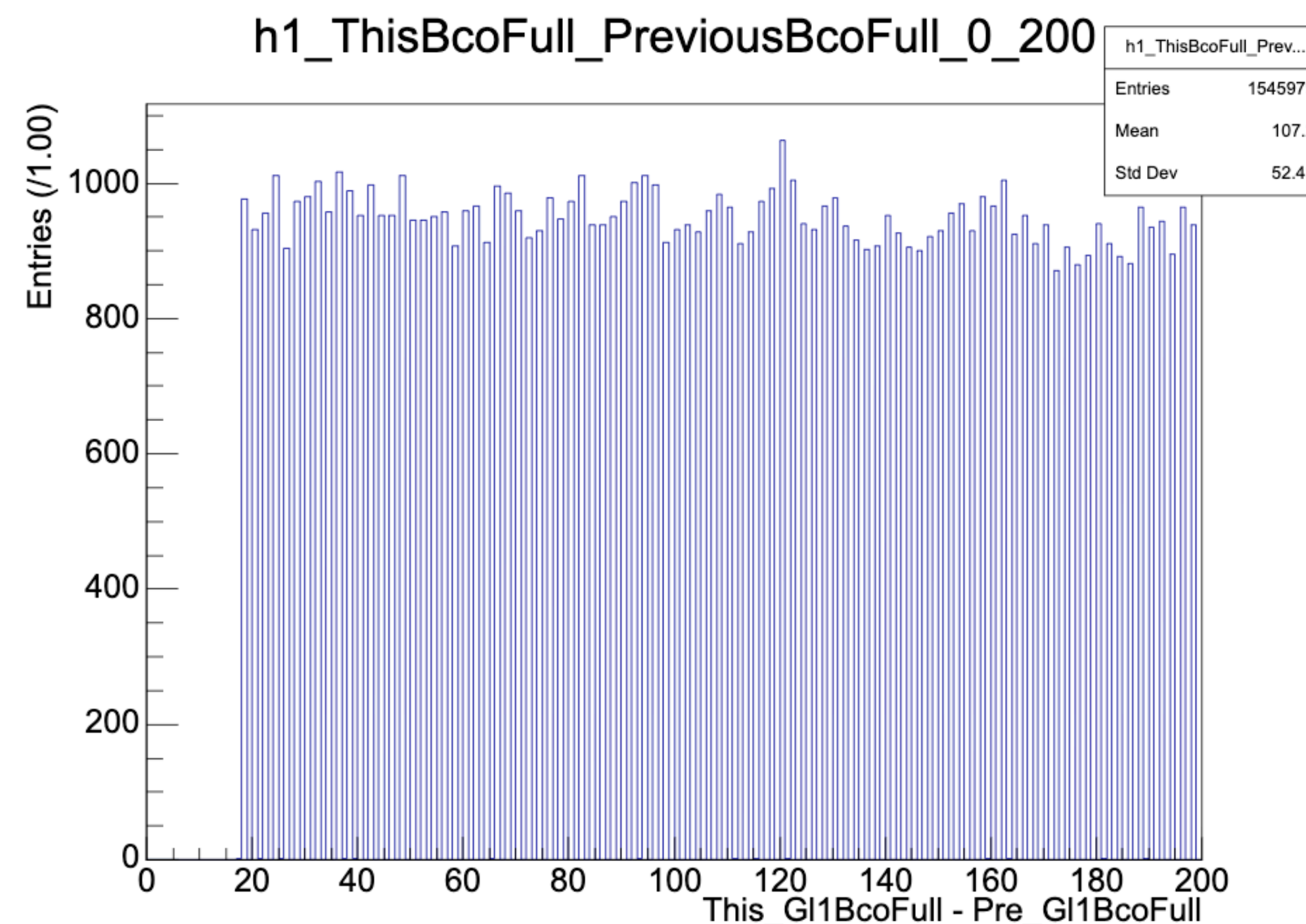
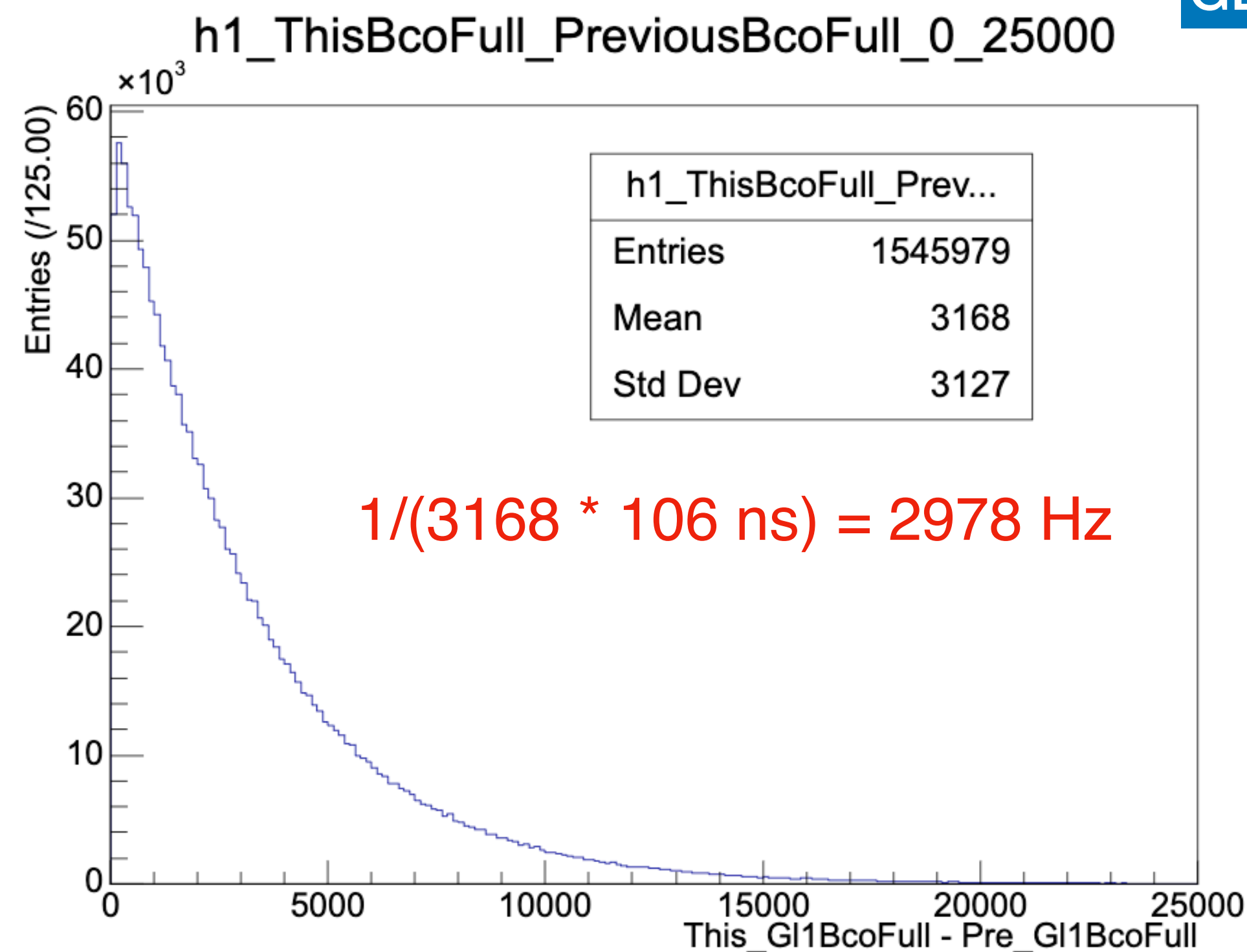
- Question 3.: As shown in cartoon, what if we have "hit\_bco\_0" in "this\_event", the FELIX is taking the hits with "hit\_bco\_0", but the next trigger happened within the "open-time"? what will happen?
- If it's the case, the issue we see in run 54280 will be different from that of run 20869 due to different trigger rate

# InttBcoFullDiff w.r.t previous events

Code in [GitHub](#)

Runnumber	run time (min)	nEvent	Rate (Hz)
54279	60.133	5842231	1619.253
54280	60.183	10610255	2938.331

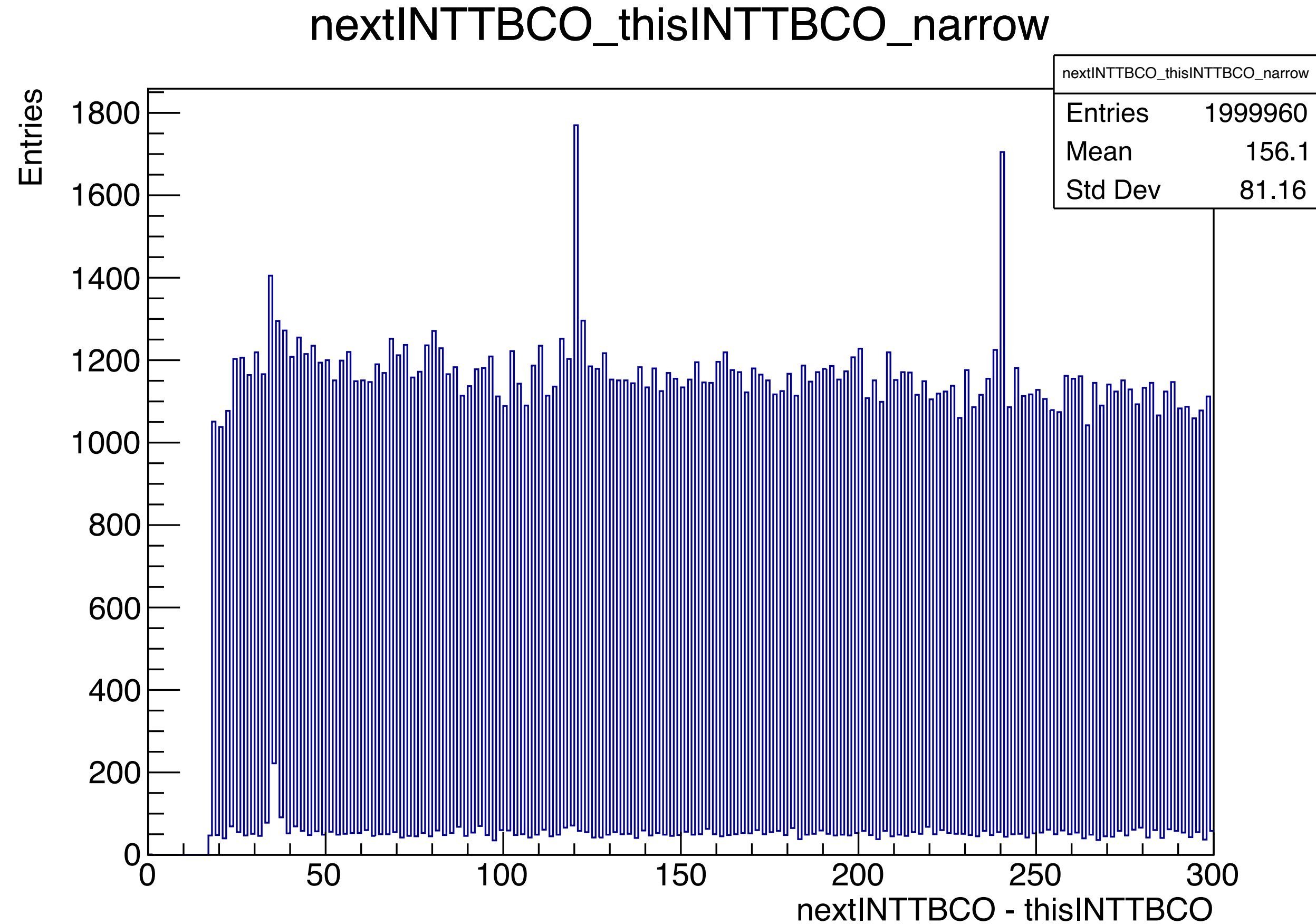
GL1BCO is used



Somehow the distribution event bco is different from what we expected  
But it seems to be the case, at least, the average trigger rate is matched

Somehow run54280 has higher trigger rate than the previous run  $\rightarrow$  could possibly by re-tune the scale-down factor

INTT BCOFULL (from "INTTEVENTHEADER->get\_bco\_full()")

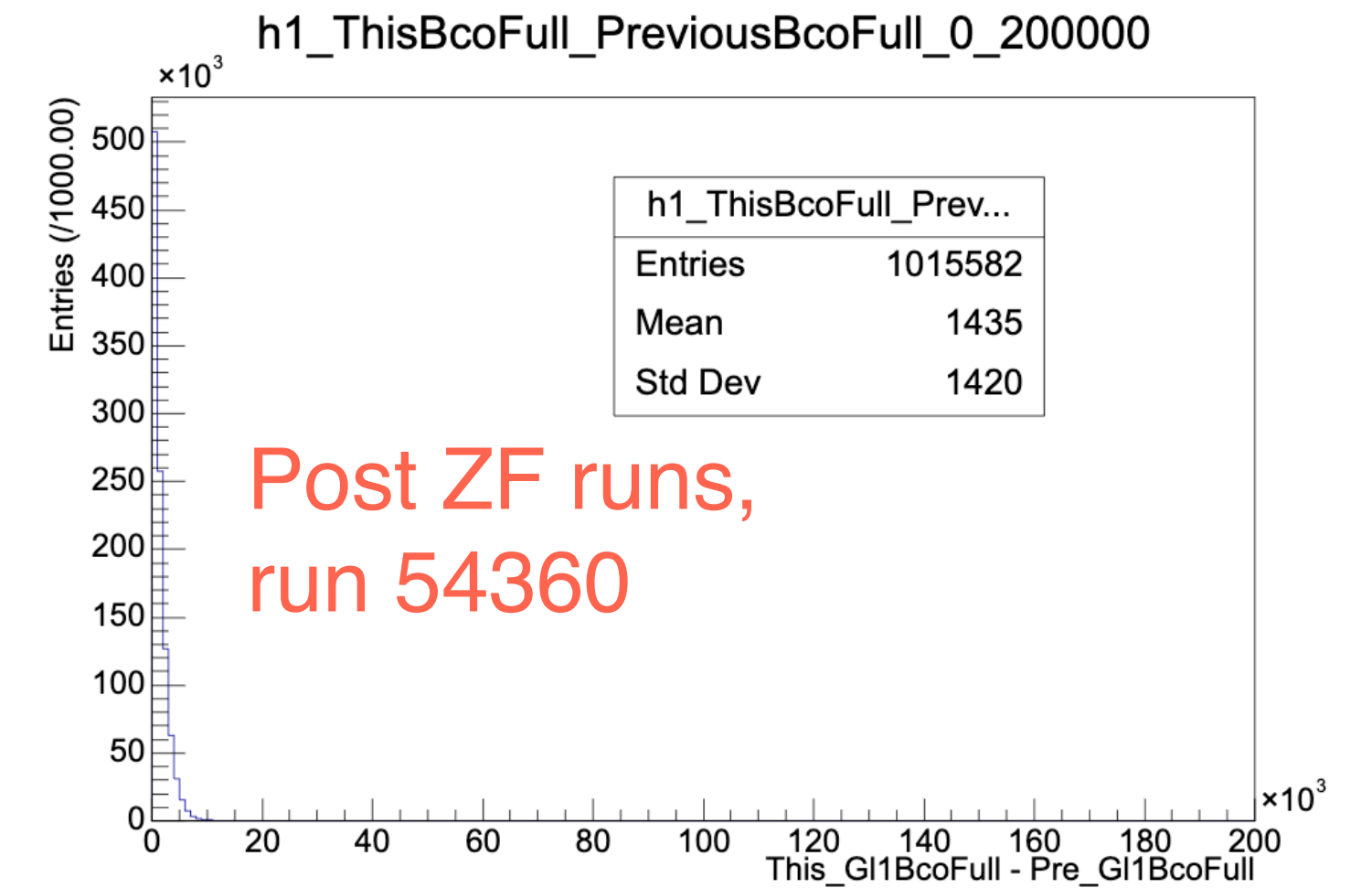
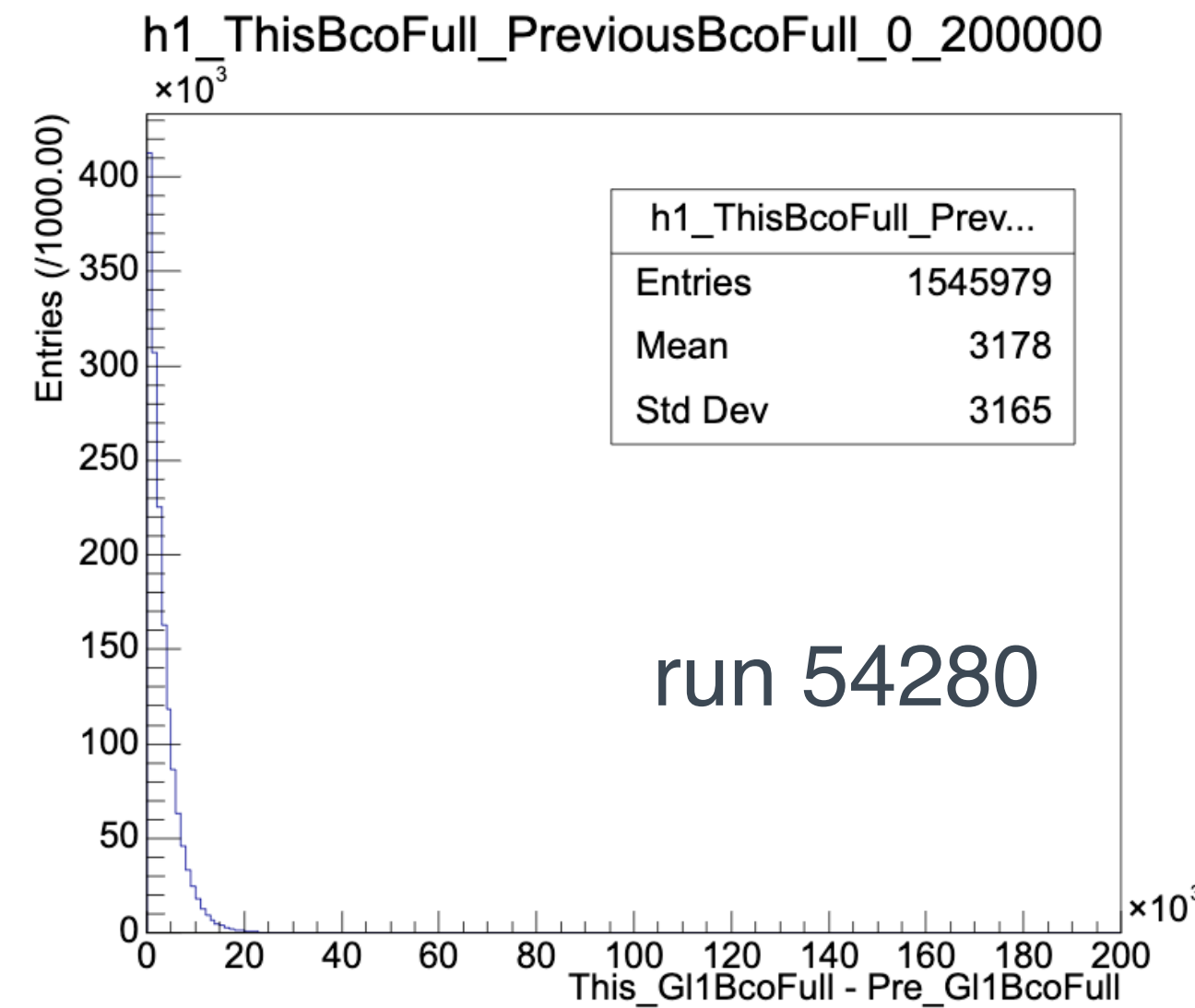
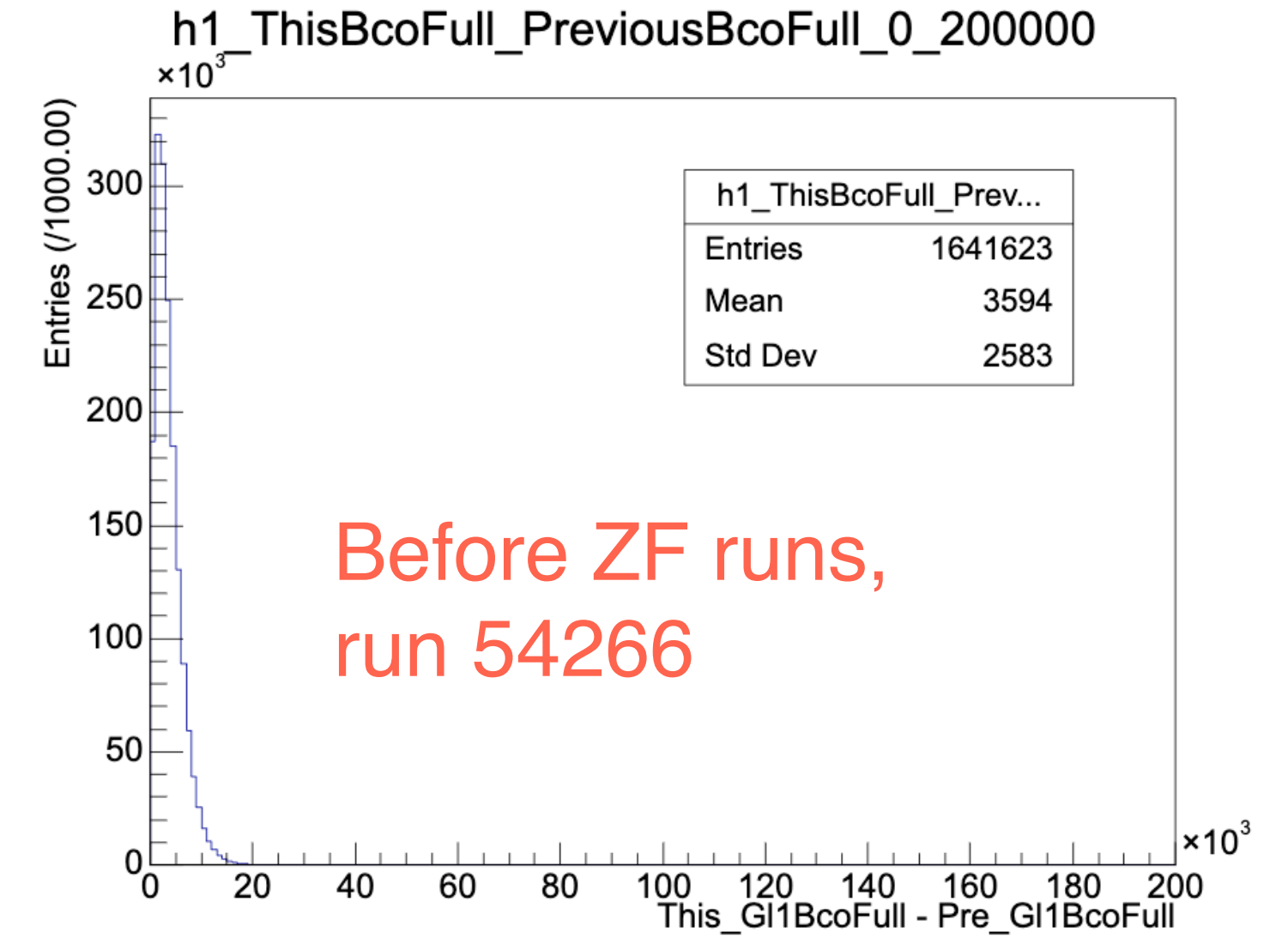
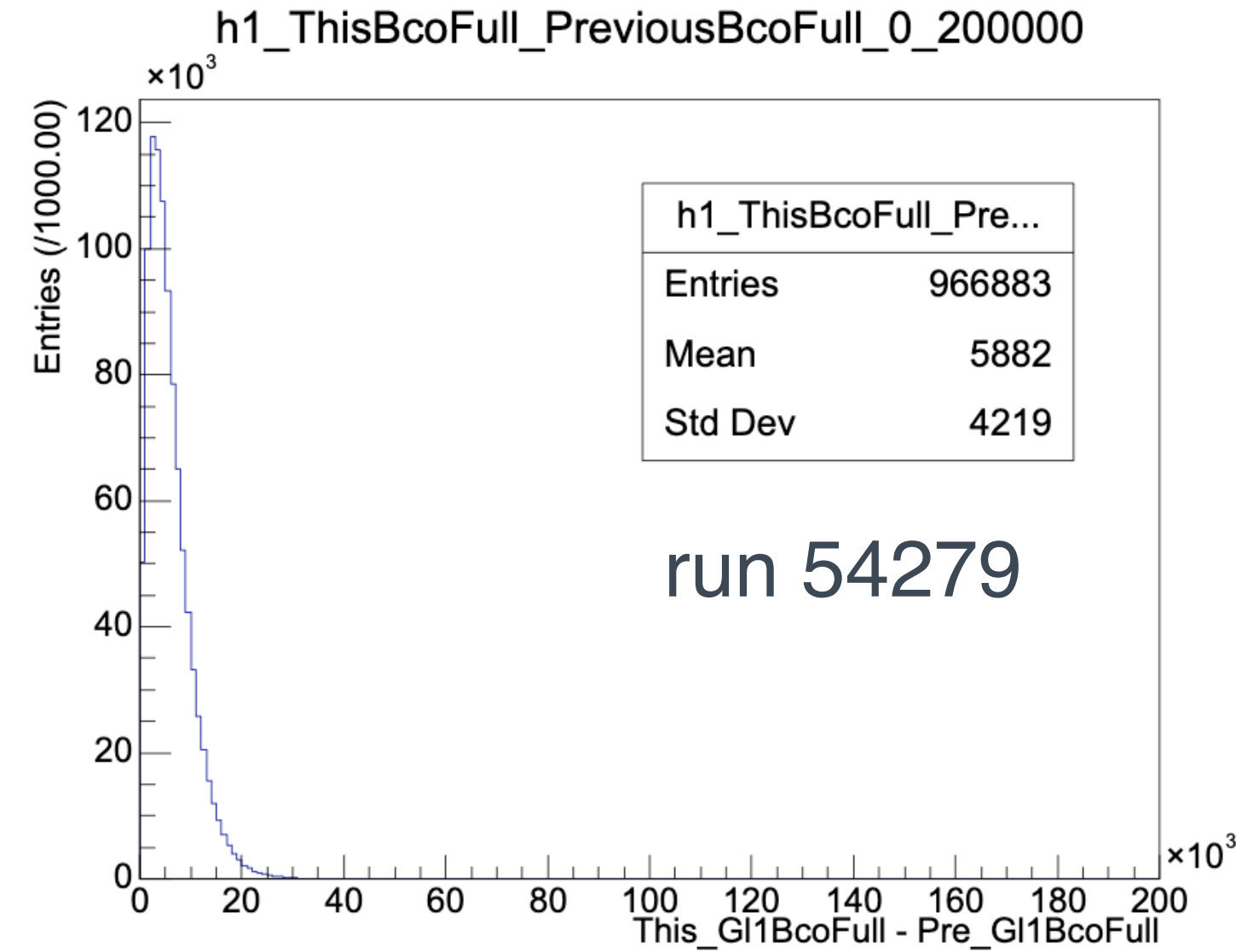
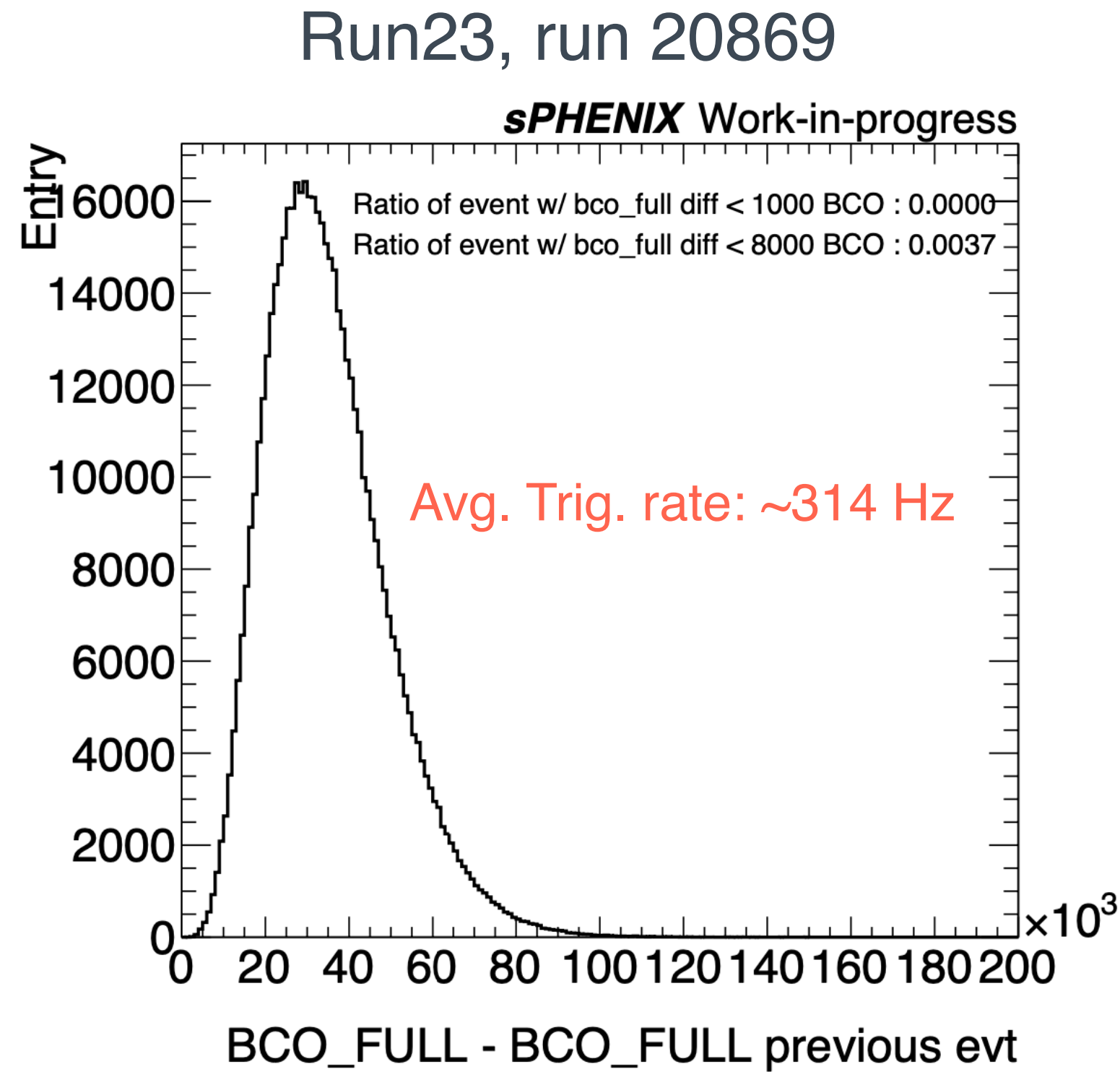


Still similar distribution comparing to that of made of GL1BCO

It seems that INTT FELIX servers don't deny the coming trigger signals even when the data processing is still ongoing

# InttBcoFullDiff w.r.t previous events

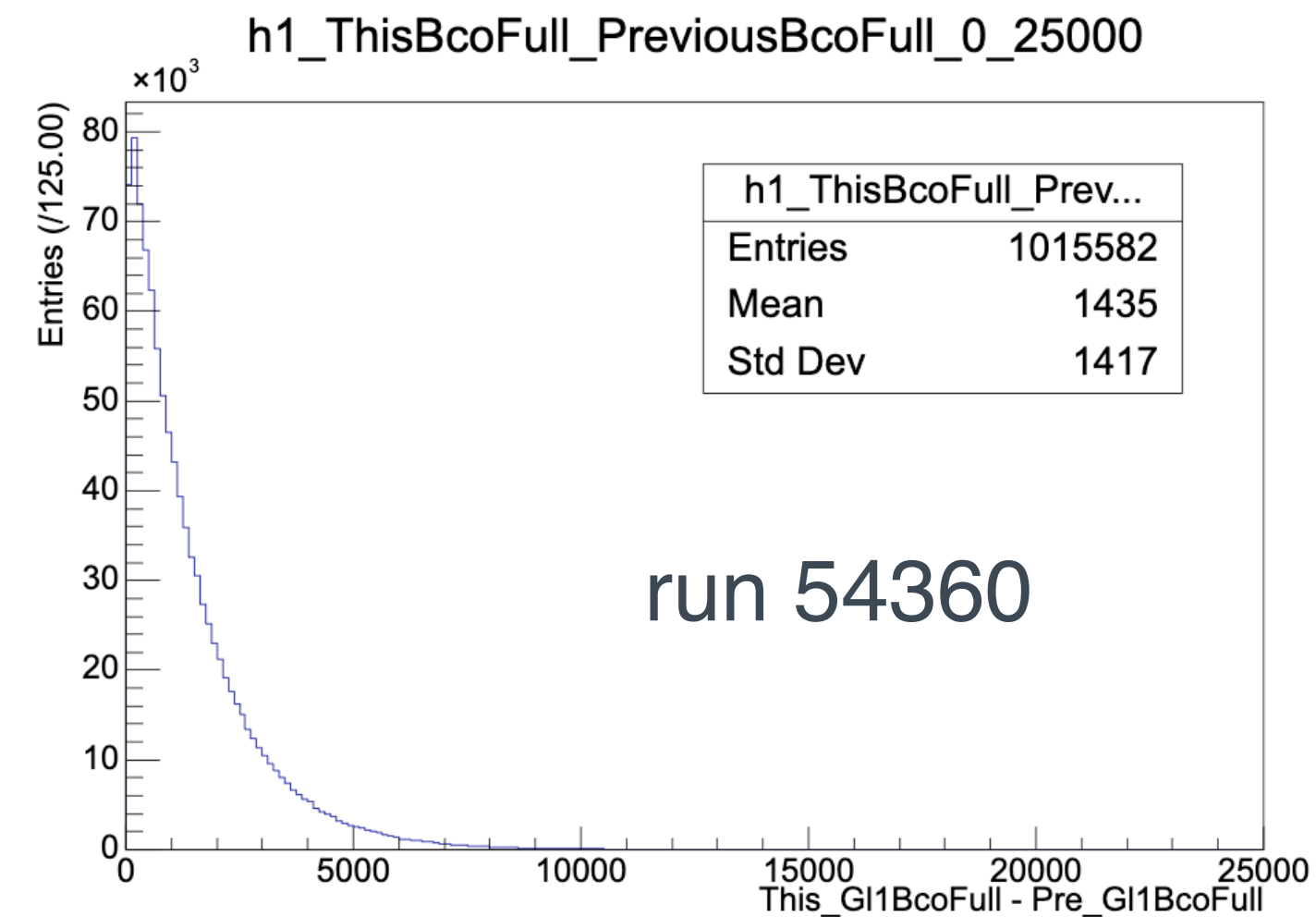
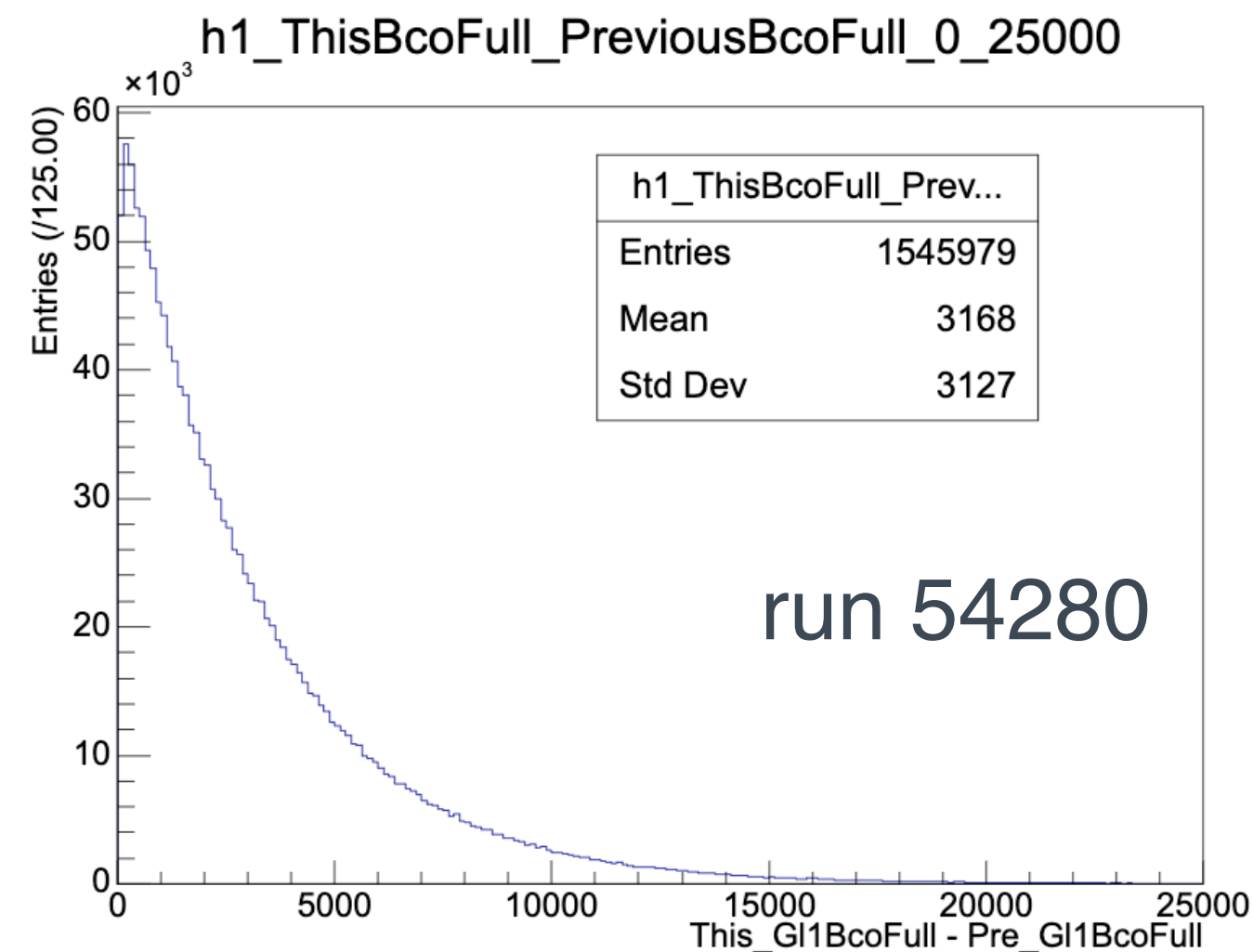
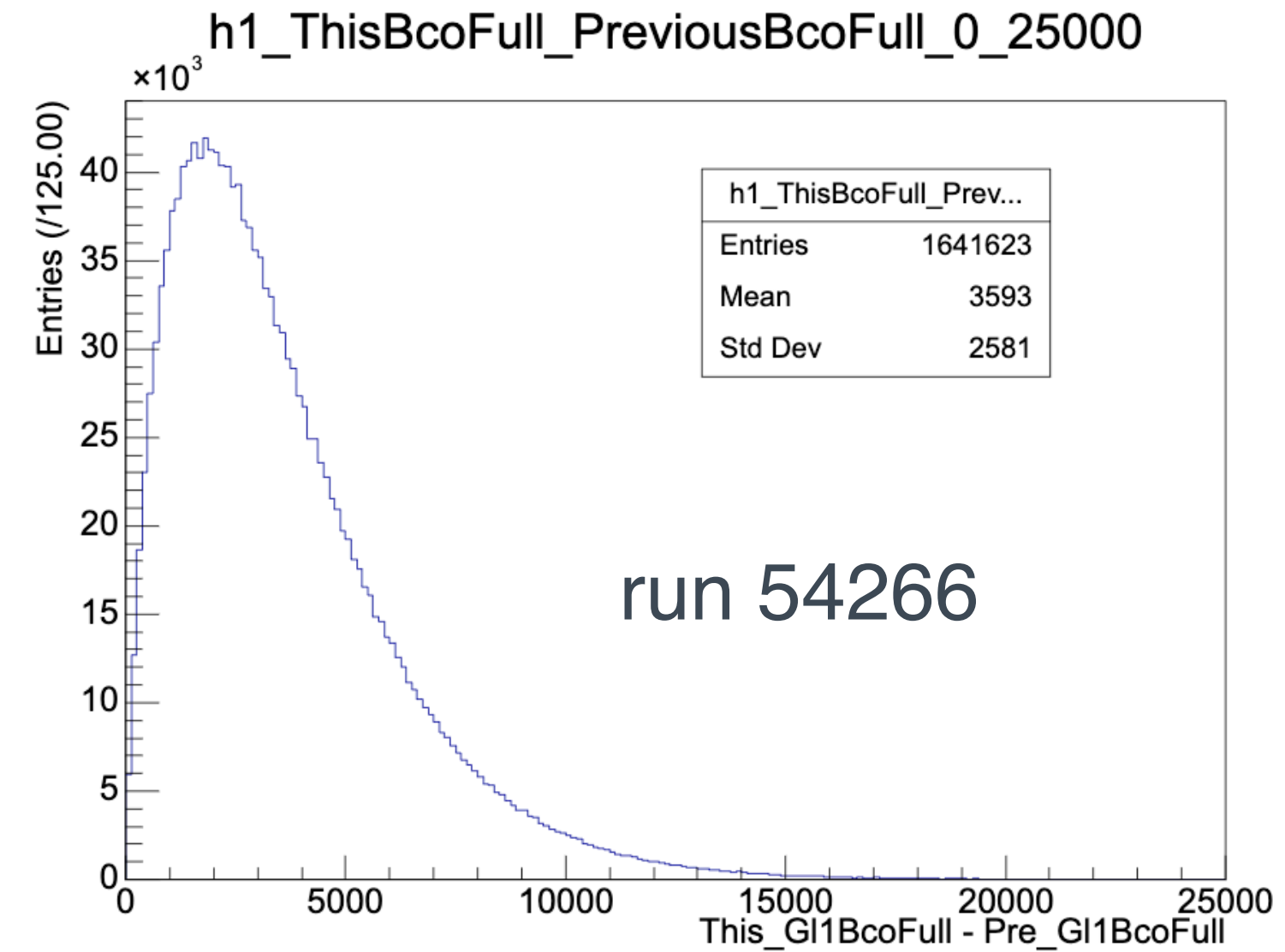
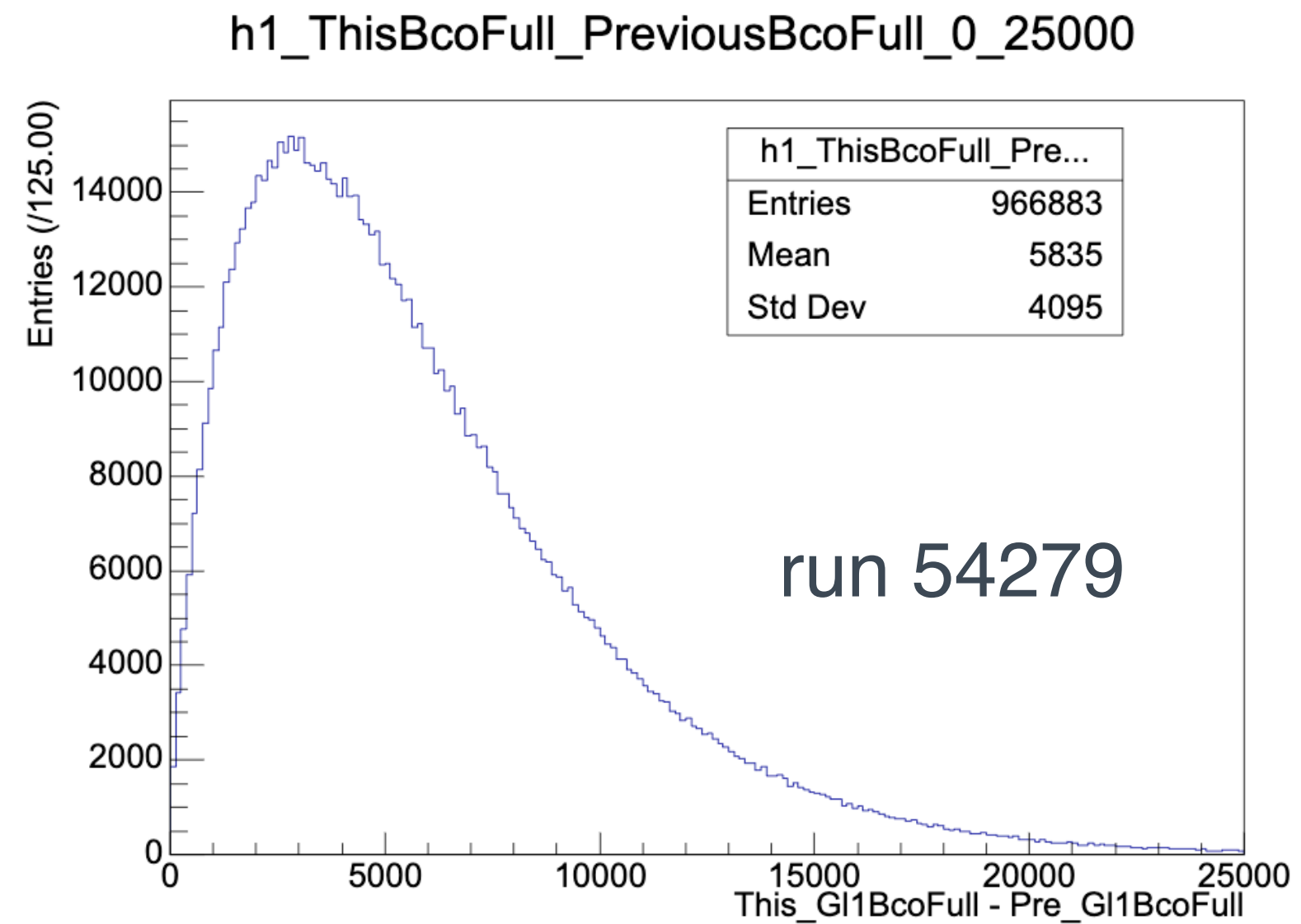
GL1BCO is used



# InttBcoFullDiff w.r.t previous events (narrow)



GL1BCO is used



The distributions look reasonable

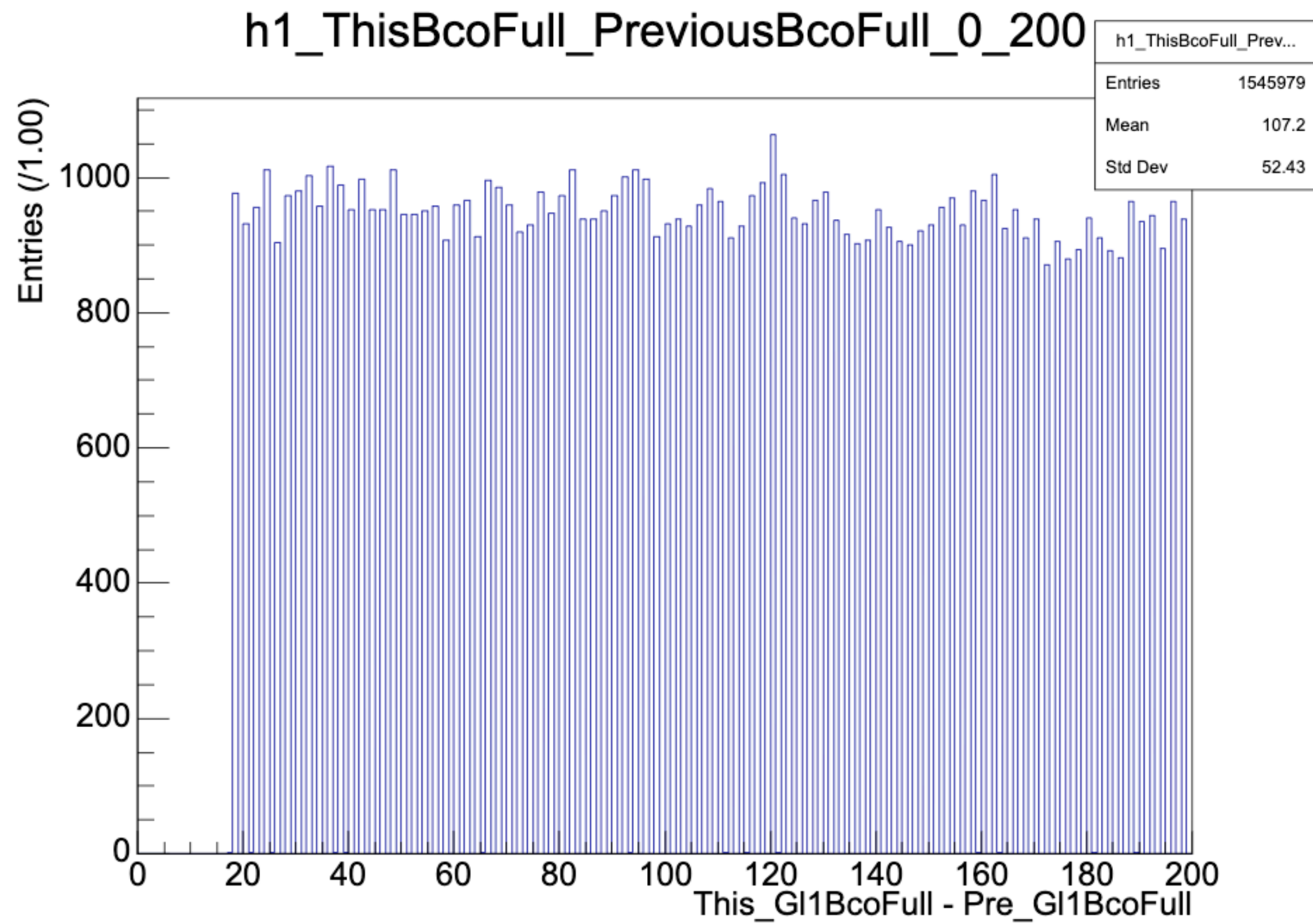
To have the Poisson distribution with large  $\lambda$ , to trigger rate has to be very low, few hundred Hz

# InttBcoFullDiff w.r.t previous events (narrow)

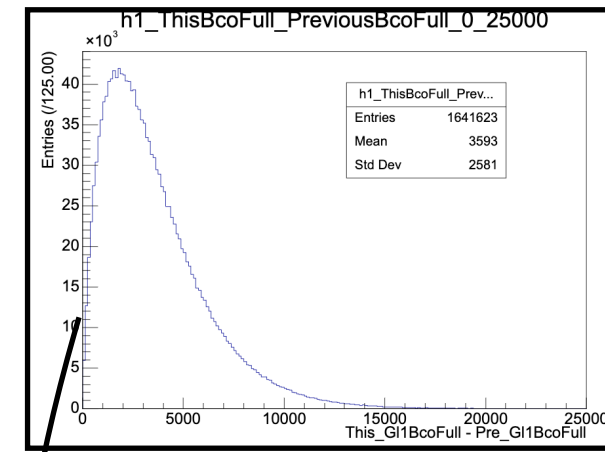
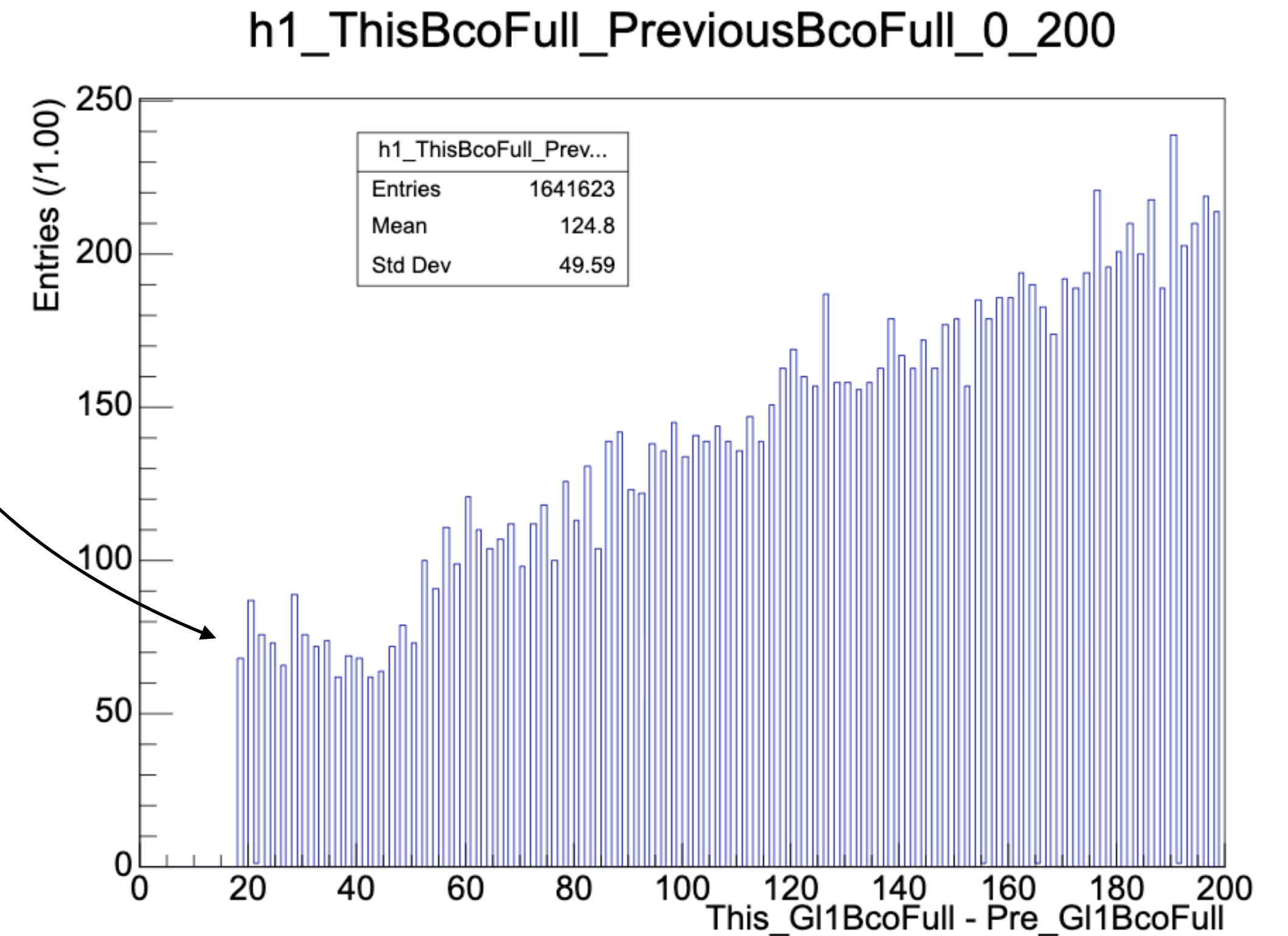


GL1BCO is used

run 54280



run 54266



Have the same dead time, 17 BCO (It may be the default set in the GTM? not due to the busy signal?)

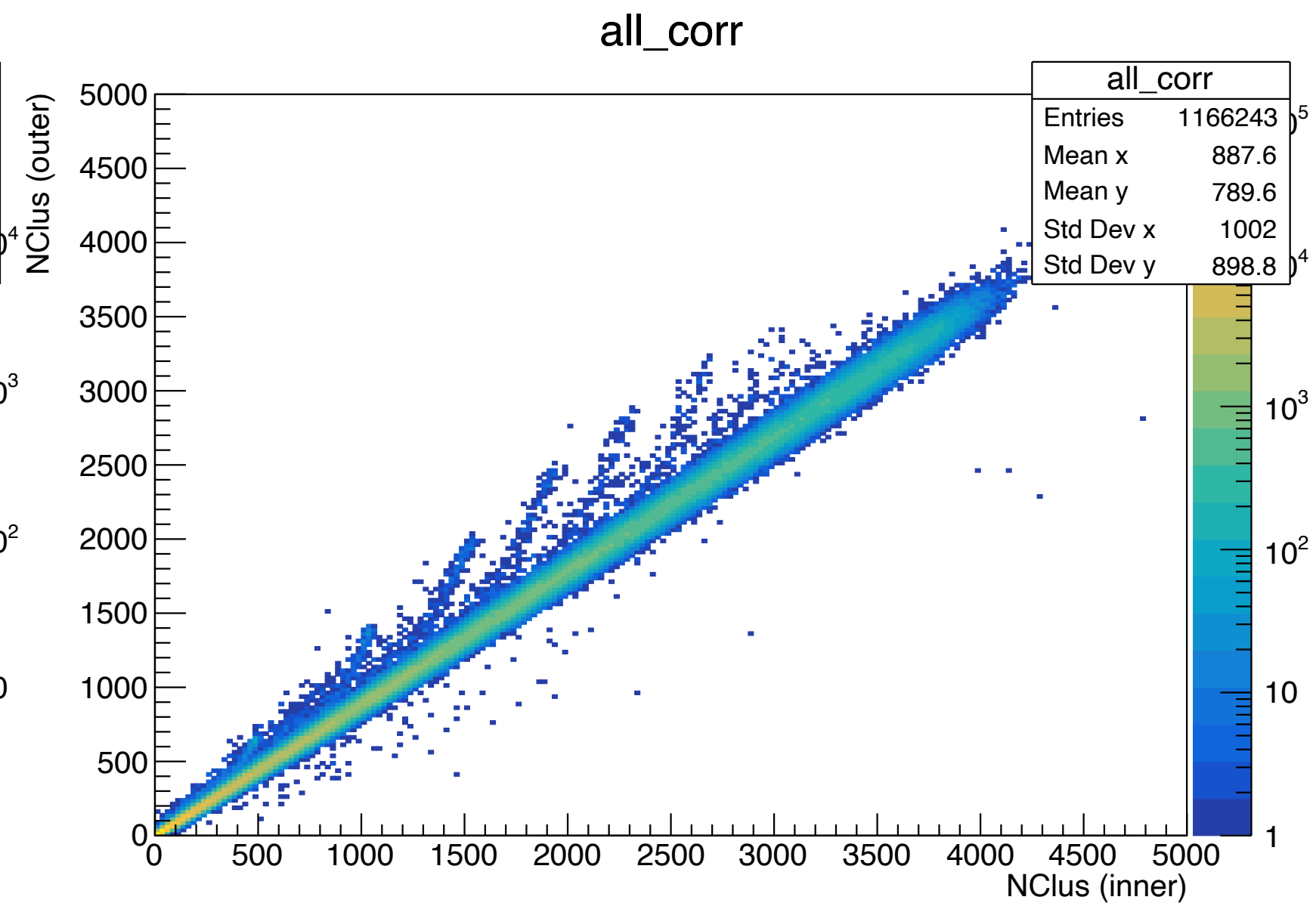
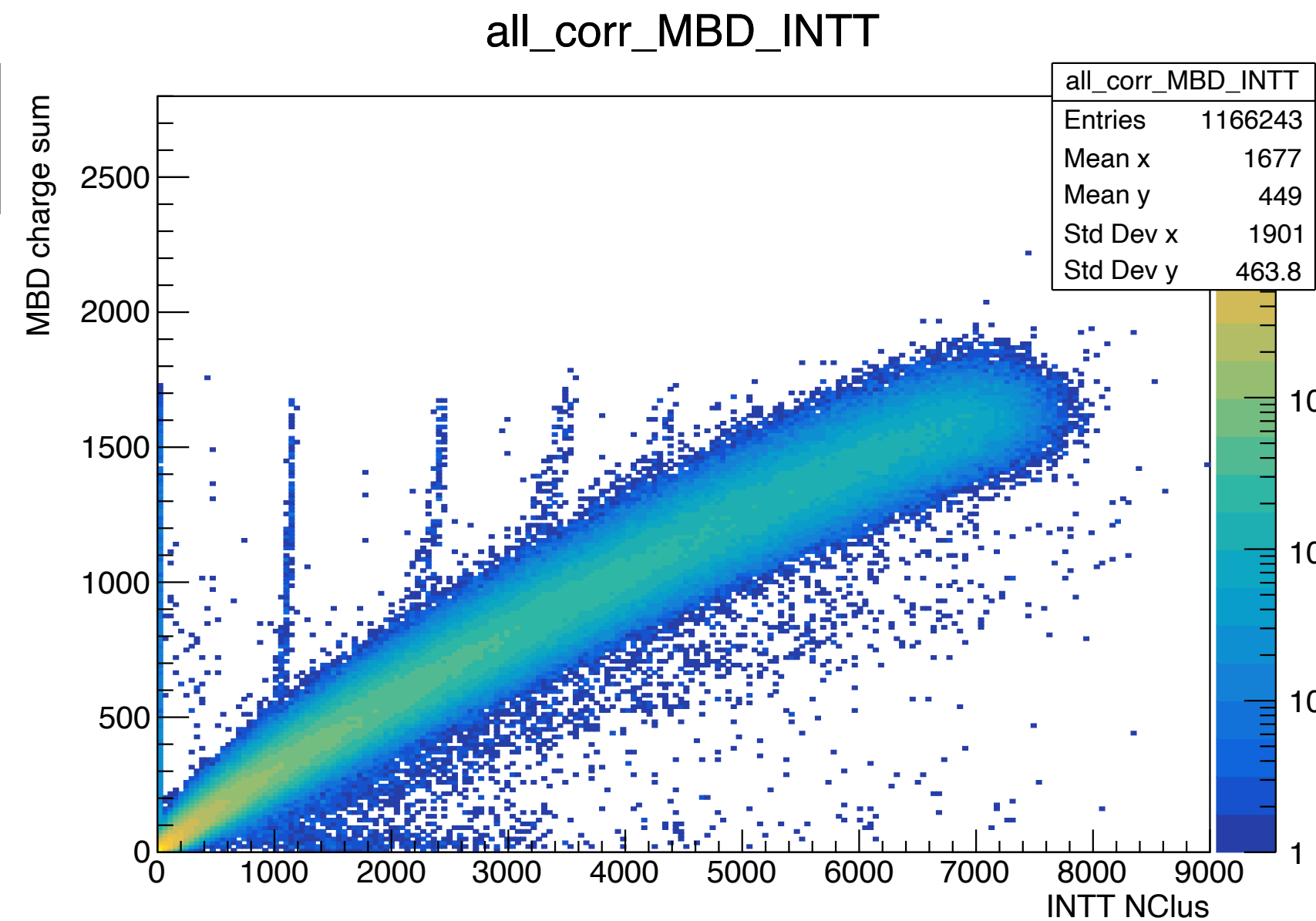
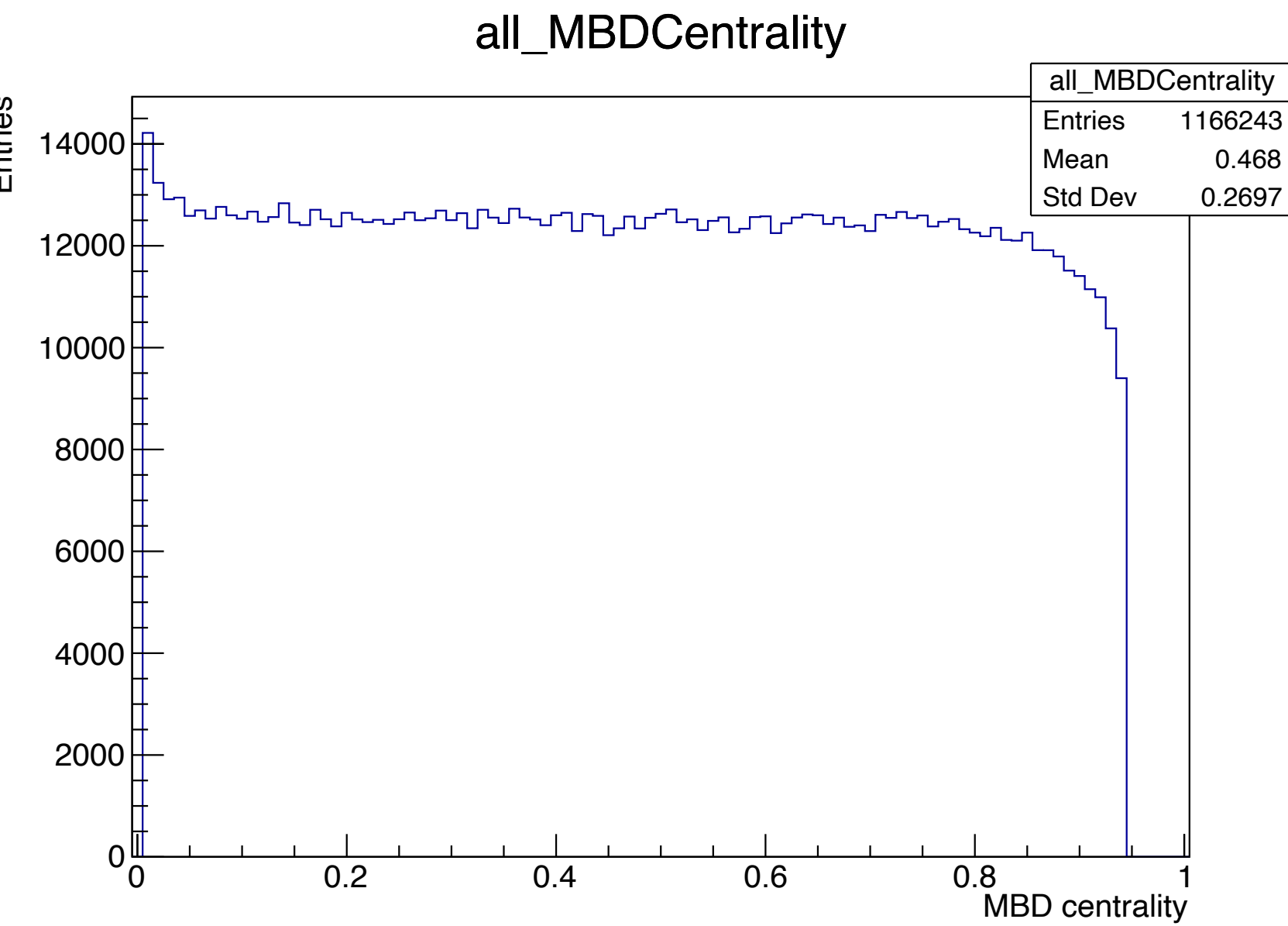


# Proposed selection: correlations, all events



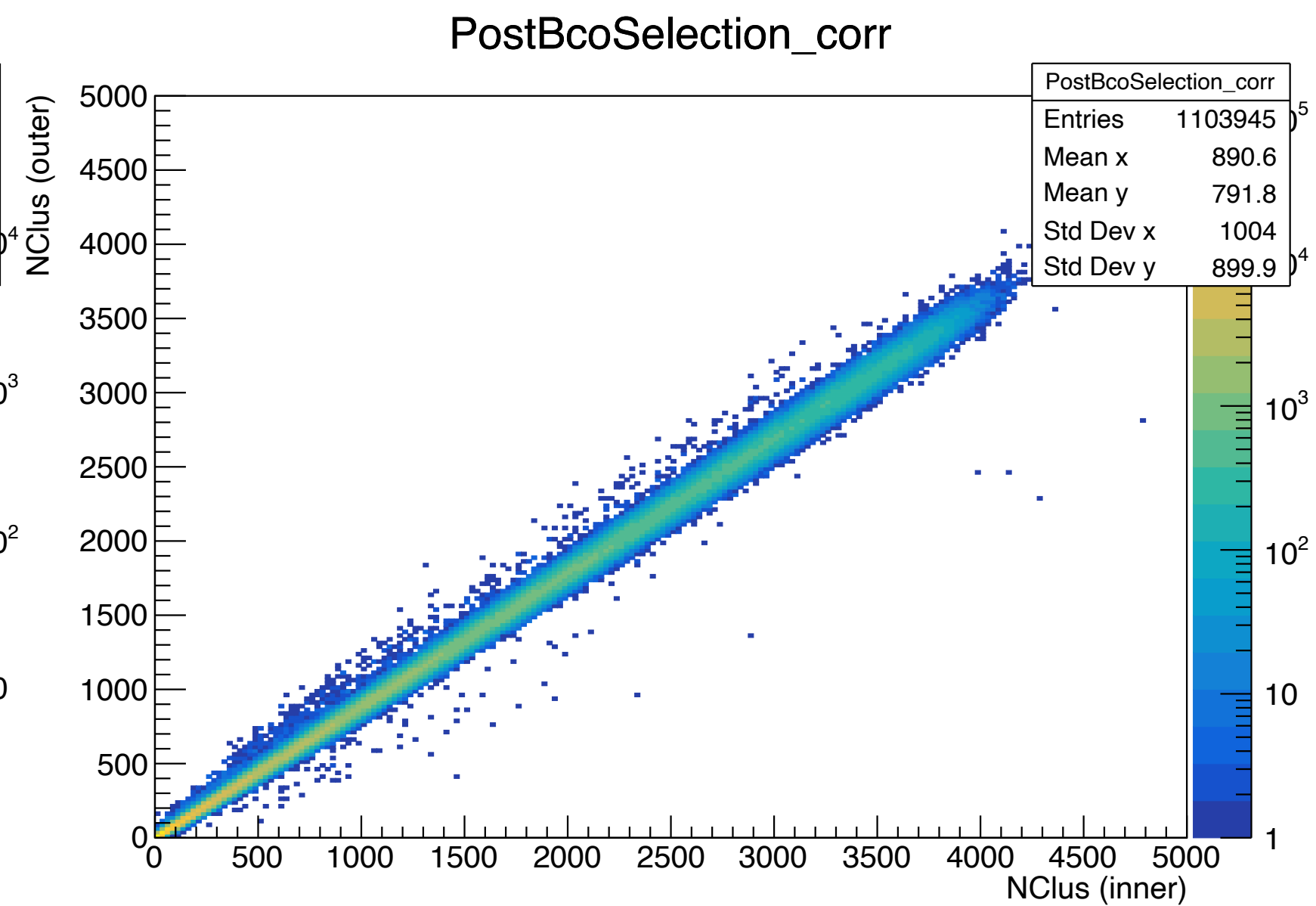
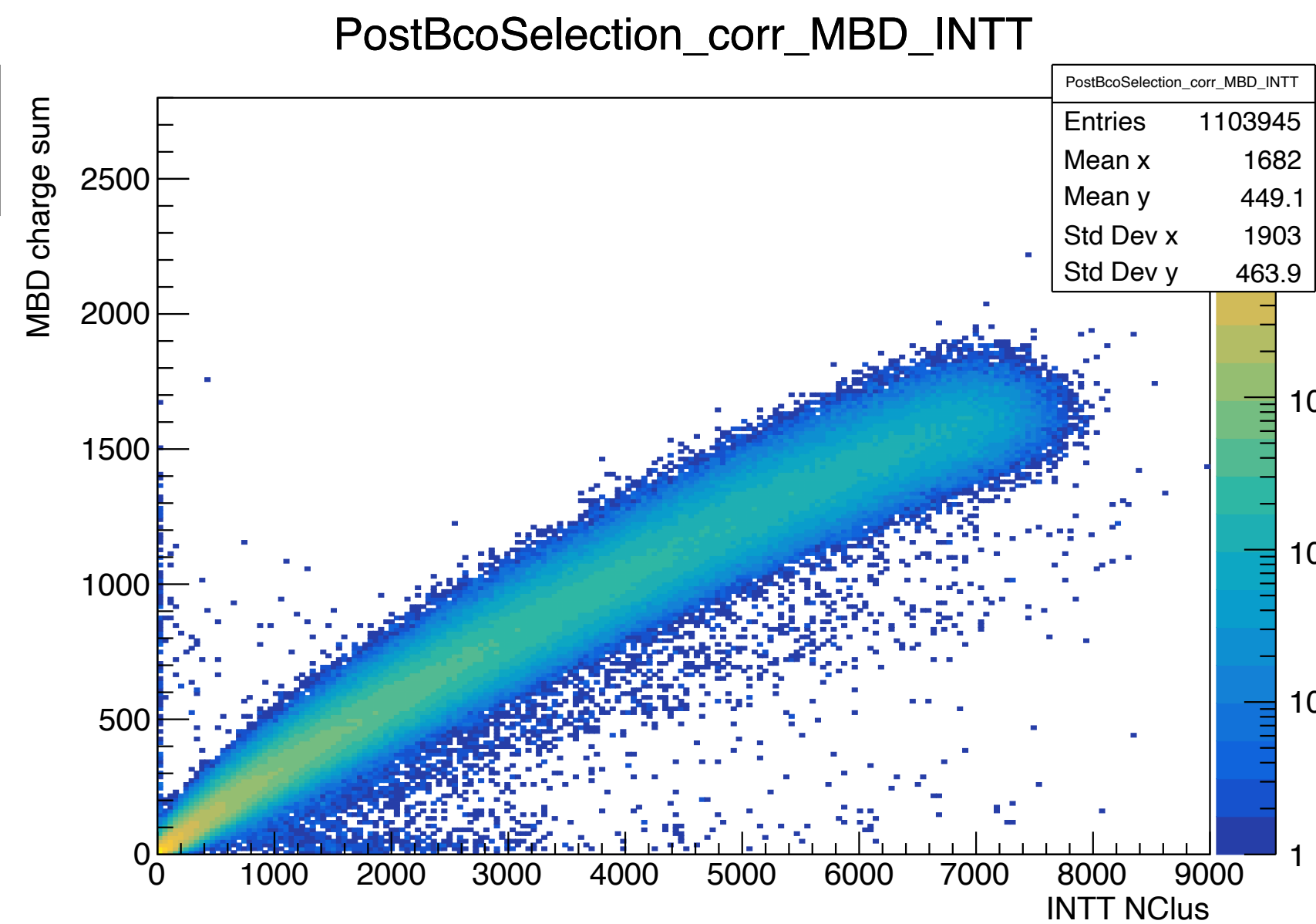
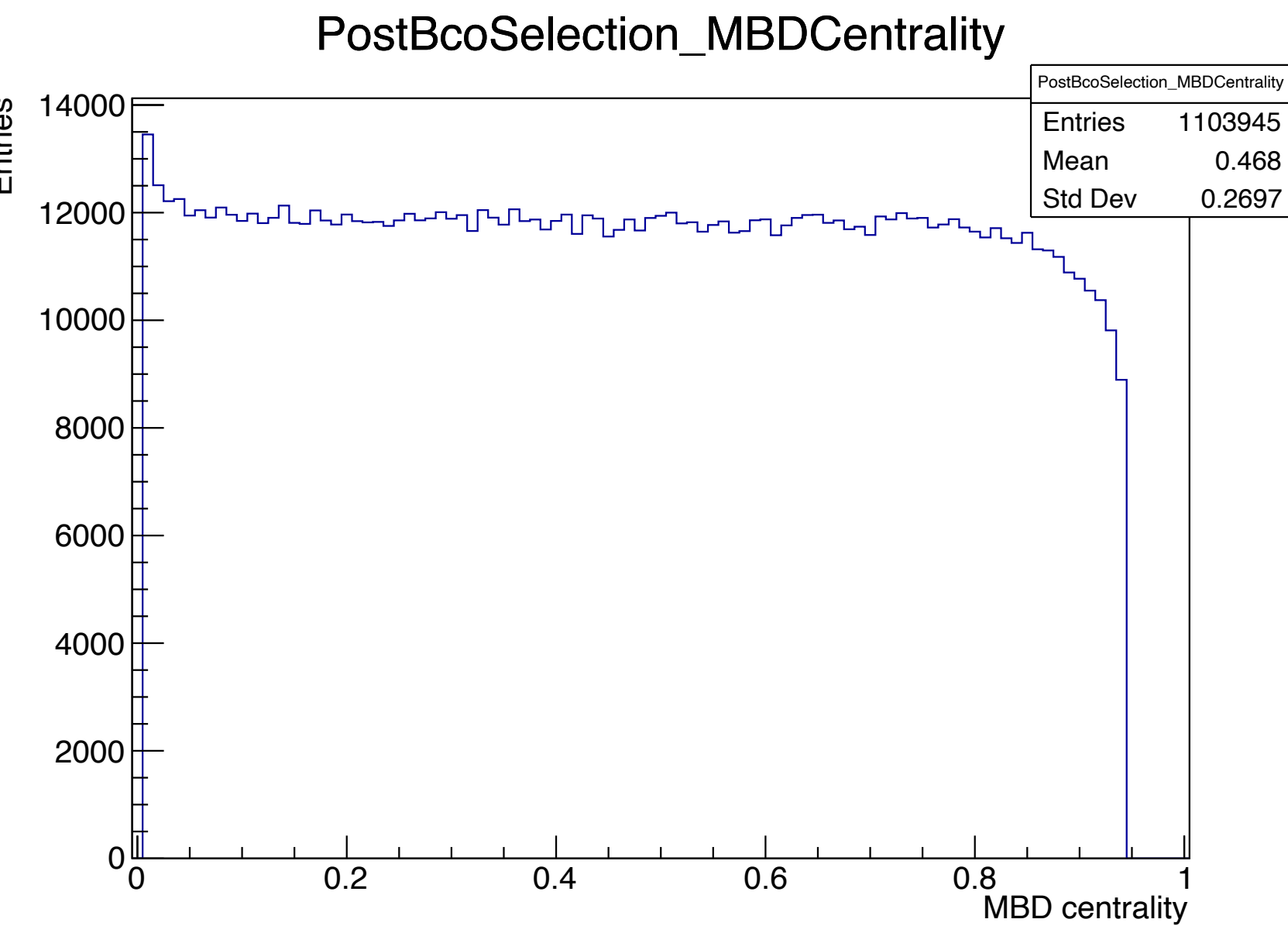
Run 54280

Only events with  $-10 \text{ cm} < \text{MBD\_z\_vtx} < 10 \text{ cm}$  are included



# Proposed selection: correlations, post cut

Only events with  $-10 \text{ cm} < \text{MBD\_z\_vtx} < 10 \text{ cm}$  are included  
Events w/  $\text{NextInttBcoFull} - \text{ThisInttBcoFull} > 188$  are kept

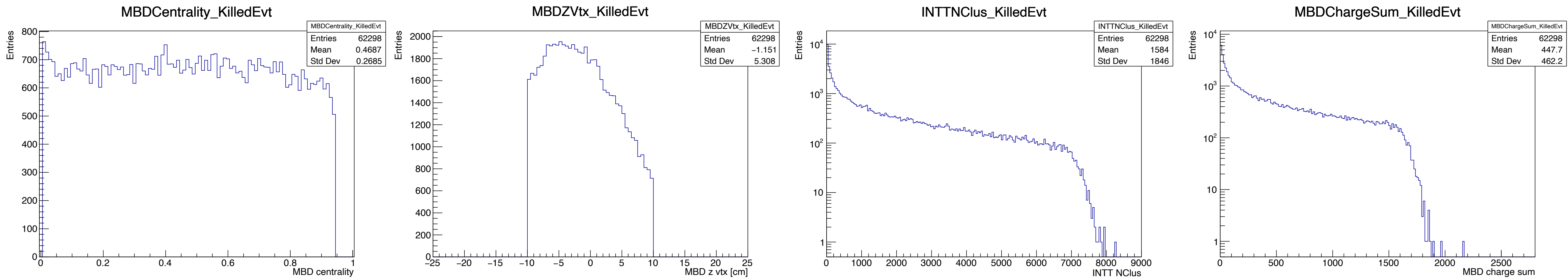


62,298 out of 1,166,243 events are excluded  $\rightarrow$  5.34%

# Distributions of the excluded events

Only evens with  $-10 \text{ cm} < \text{MBD\_z\_vtx} < 10 \text{ cm}$  are included  
Events w/  $\text{NextInttBcoFull} - \text{ThisInttBcoFull} > 188$  are kept

## The omitted events



62,298 out of 1,166,243 events are excluded  $\rightarrow$  5.34%  
Seems to be no strong multiplicity/centrality dependence

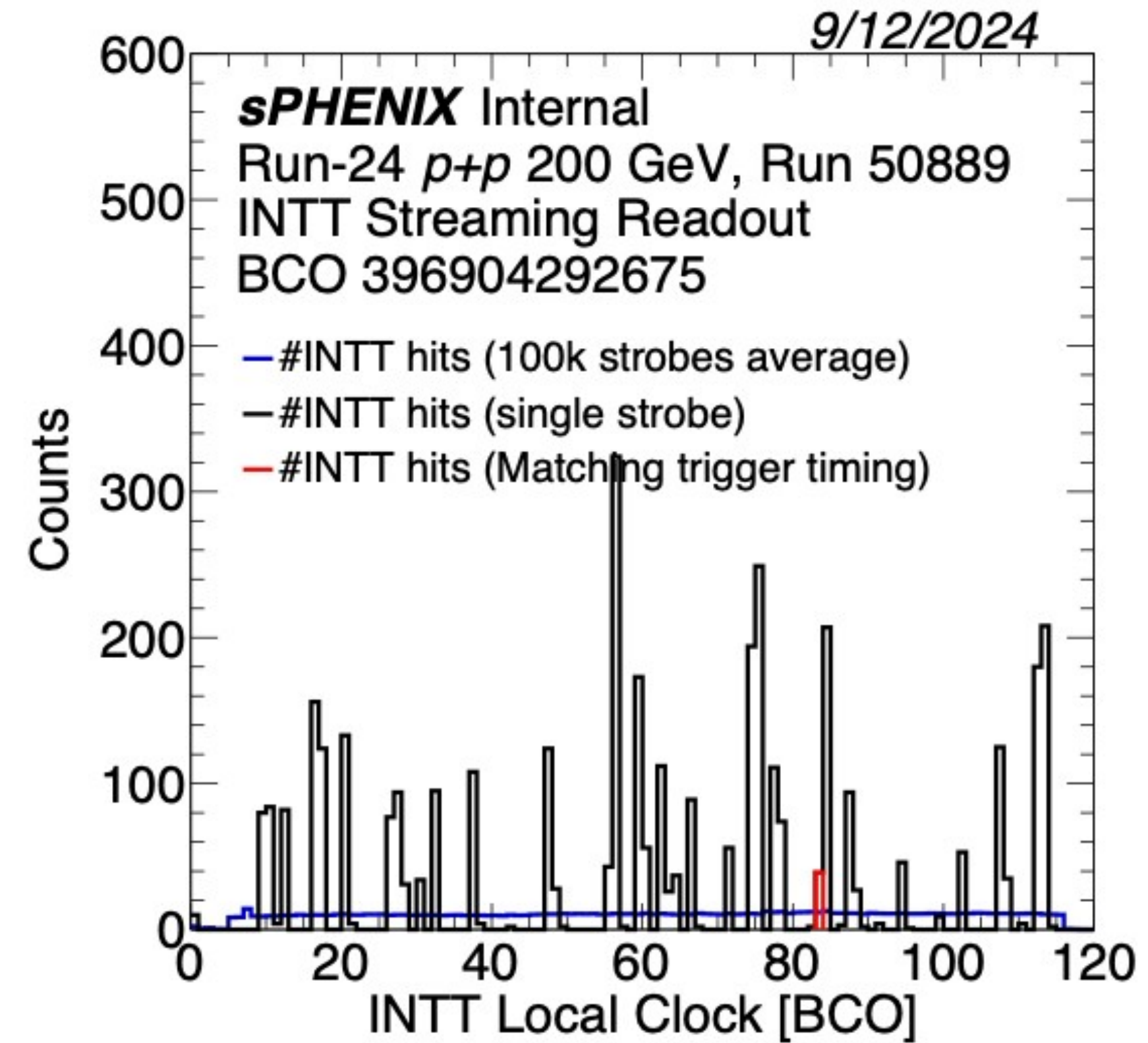
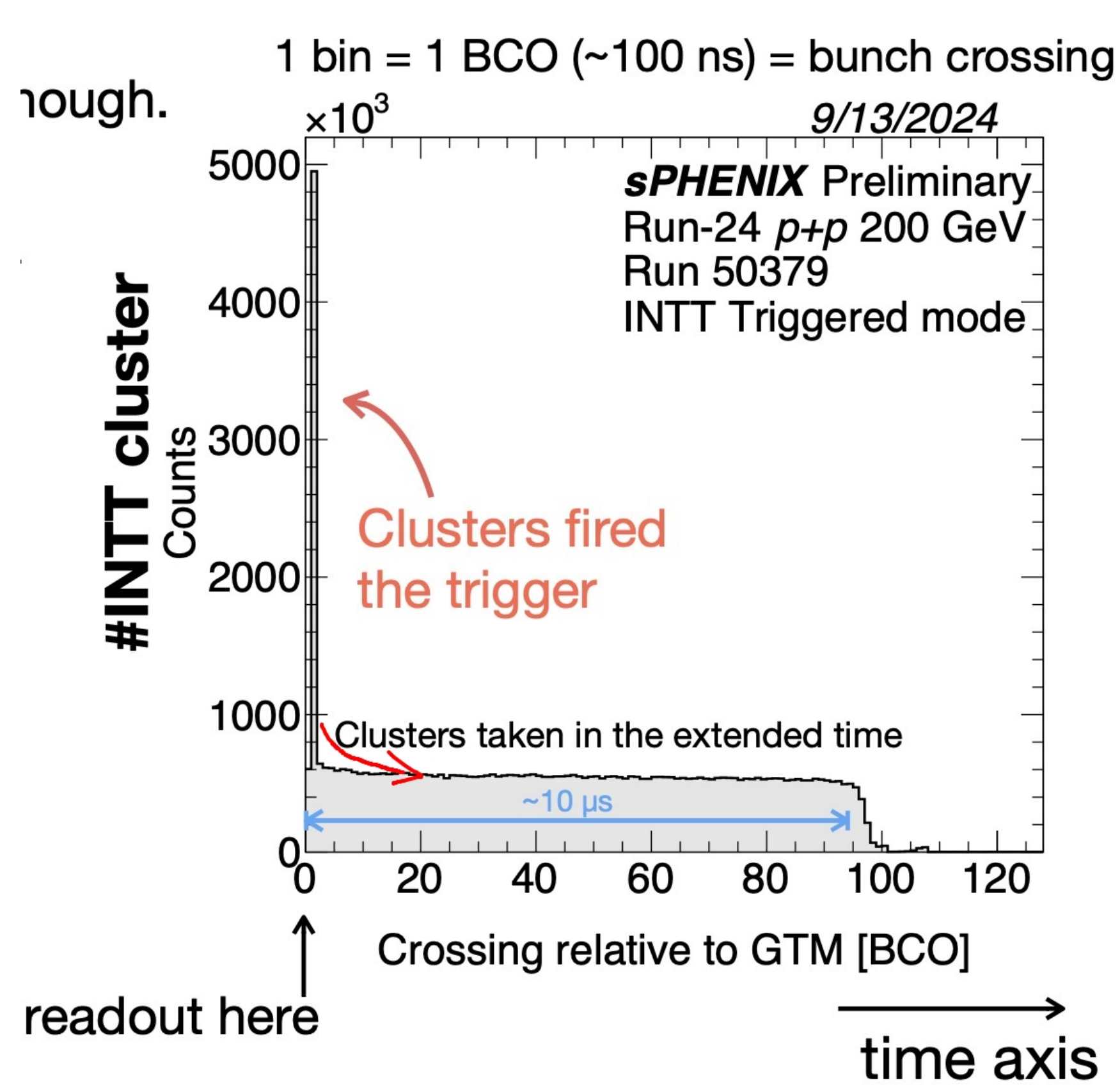
# Apple-to-apple comparison?



- High rate run in run 23 AuAu?
  - If the trigger rate is too low that there trigger span is longer than 1000, then it can be different
  - And see the behavior the fish-bone?

runnumber	runtype	brtimestamp	ertimestamp	updatetimestamp	eventsinrun	marked_invalid	has_comment	qcomment	trigger_rate	round
23901	beam	2023-07-27 00:38:16	2023-07-27 00:38:53		26994	-1	0		729.5675675675675676	0.617
23902	beam	2023-07-27 00:41:06	2023-07-27 00:42:07		54046	-1	0		886.0000000000000000	1.017
23903	beam	2023-07-27 01:08:11	2023-07-27 01:09:05		42592	-1	0		788.7407407407407407	0.900
23904	beam	2023-07-27 01:11:15	2023-07-27 01:12:20		92495	-1	0		1423.0000000000000000	1.083
23905	beam	2023-07-27 01:21:15	2023-07-27 01:26:27		968137	-1	0		3103.0032051282051282	5.200
23906	beam	2023-07-27 01:30:31	2023-07-27 01:36:51		1147432	-1	0		3019.5578947368421053	6.333
23907	beam	2023-07-27 01:40:36	2023-07-27 01:46:59		1172162	-1	0		3060.4751958224543081	6.383
23910	beam	2023-07-27 04:02:20	2023-07-27 04:04:06		50472	-1	0		476.1509433962264151	1.767
23911	beam	2023-07-27 04:06:59	2023-07-27 04:13:01		1232167	-1	0		3403.7762430939226519	6.033
23912	beam	2023-07-27 04:15:59	2023-07-27 04:21:40		1097876	-1	0		3219.5777126099706745	5.683
23913	beam	2023-07-27 04:24:02	2023-07-27 04:30:32		836609	-1	0		2145.1512820512820513	6.500
23914	beam	2023-07-27 04:33:02	2023-07-27 04:38:39		854458	-1	0		2535.4836795252225519	5.617
23915	beam	2023-07-27 04:43:39	2023-07-27 04:49:39		892075	-1	0		2477.9861111111111111	6.000
23916	beam	2023-07-27 04:52:22	2023-07-27 04:58:03		882601	-1	0		2588.2727272727272727	5.683
23917	beam	2023-07-27 05:00:32	2023-07-27 05:06:17		928719	-1	0		2691.9391304347826087	5.750
23918	beam	2023-07-27 05:09:16	2023-07-27 05:11:40		60886	-1	0		422.8194444444444444	2.400
23919	beam	2023-07-27 05:16:31	2023-07-27 05:22:19		898983	-1	0		2583.2844827586206897	5.800
23920	beam	2023-07-27 05:23:01	2023-07-27 05:23:48		32258	-1	0		686.3404255319148936	0.783
23921	beam	2023-07-27 05:29:39	2023-07-27 05:35:14		890837	-1	0		2659.2149253731343284	5.583
23922	beam	2023-07-27 05:39:05	2023-07-27 05:44:56		972576	-1	0		2770.8717948717948718	5.850
23923	beam	2023-07-27 05:47:09	2023-07-27 05:52:56		939211	-1	0		2706.6599423631123919	5.783

/sphenix/tg/tg01/commissioning/INTT/data/evt\_files/beam/beam\_intt{0..7}-00023911-0000.evt  
ncollision 4, open\_time 35, DAC0 18



The timing resolution of INTT is limited by the performance FPHX chip  
What important here is to understand the fraction of the hits moved to the next  
bco due to the imperfect cross/fine delay settings

- INTT has the chip saturation issue (FELIX rejects the hits if they are too late arrived)
  - One chip can have up to 73 hits with the given FELIX\_open\_time of 60 BCO\*
  - In the worst case, ~5% of the hits are rejected by FELIX in the run 54280
  - The patterns of the hit maps of the saturated chips are one chunk + zebra crosswalk
  - The chip saturation issue is correlated with the two spikes in the cluster phi size distribution (the sizes of the chunks are 43 or 46 predominantly)
- We have learnt that the InttBcoFullDiff w.r.t the next event of events of interest (EOI) is very short
  - This is the issue described in the slide 20 (Hypothesis: The INTTEventHeader is overwritten by the next trigger while still doing the hit assembly with the hits associated with previous trigger)
- The `InttBcoFullDiff` distribution is very different from that of run 20869
  - The same distributions of different runs are checked, not major issue spotted, look reasonable
  - The trigger rate of run 20869 is something like 300 Hz
- I would like to first come up with the proposal to have the `InttBcoFullDiff w.r.t next event` cut
  - Reject the events w/ `InttBcoFullDiff` < 188 → 5% of events are excluded
  - The performance looks good, the outliers are removed
  - And seems to be no centrality dependence

\*Need to confirm the unit of open\_time

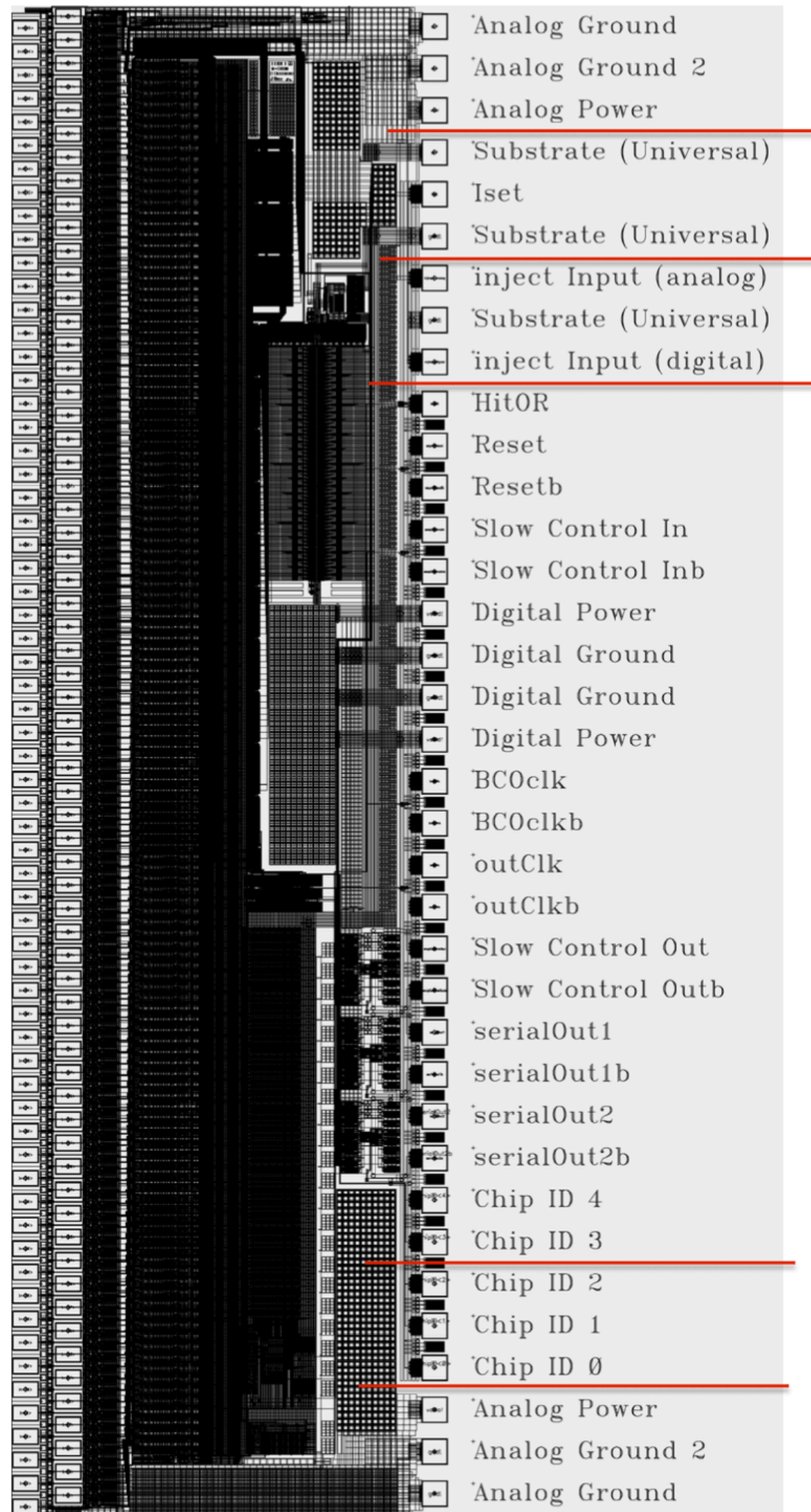
**Back up**

- The cut-off can be seen in the chip occupancy distributions, the work principle of open time is confirmed in some level
  - In run 54280, one chip can have up to 73 hits per event and per hit\_bco
  - The half-entry chips have similar structures. Half of hits cannot make it be received by ROCs, but the time is still spent
- The very next events of the event of interest (EOI) are very close to EOI in terms of the time span
  - Hypothesis: the INTT\_bcofull is overwritten when the next trigger is received by FELIX while FELIX is still proceeding the hit assembly with the rather late arrival hits corresponded to the previous INTT\_bcofull
  - Would it be a severe problem in the p+p data?
- We can possibly have a INTT\_bcofull\_diff cut. Some good events might be cut since the distribution is different from what we expect due to the rather higher collision rate
  - With the check of multiple runs, the distributions of event\_bco\_span look reasonable



One problem with this approach, however, is **how to make it deal with variable event sizes**. For example, even though “four-hits-in-four-beam-crossings” is the measuring stick for Phenix, **events of only one hit will be very common** and **events with more than four hits are actually desirable**.

Fortunately, this architecture can deal with events of less-than four hits easily. The only consequence is a slight inefficiency in the readout bandwidth. However, since synchronization words will be output whenever there is no data to be output, this slight inefficiency should allow the data acquisition system to remain in sync with the FPhx chips.



## The FPHX back-end

### Chip Organization

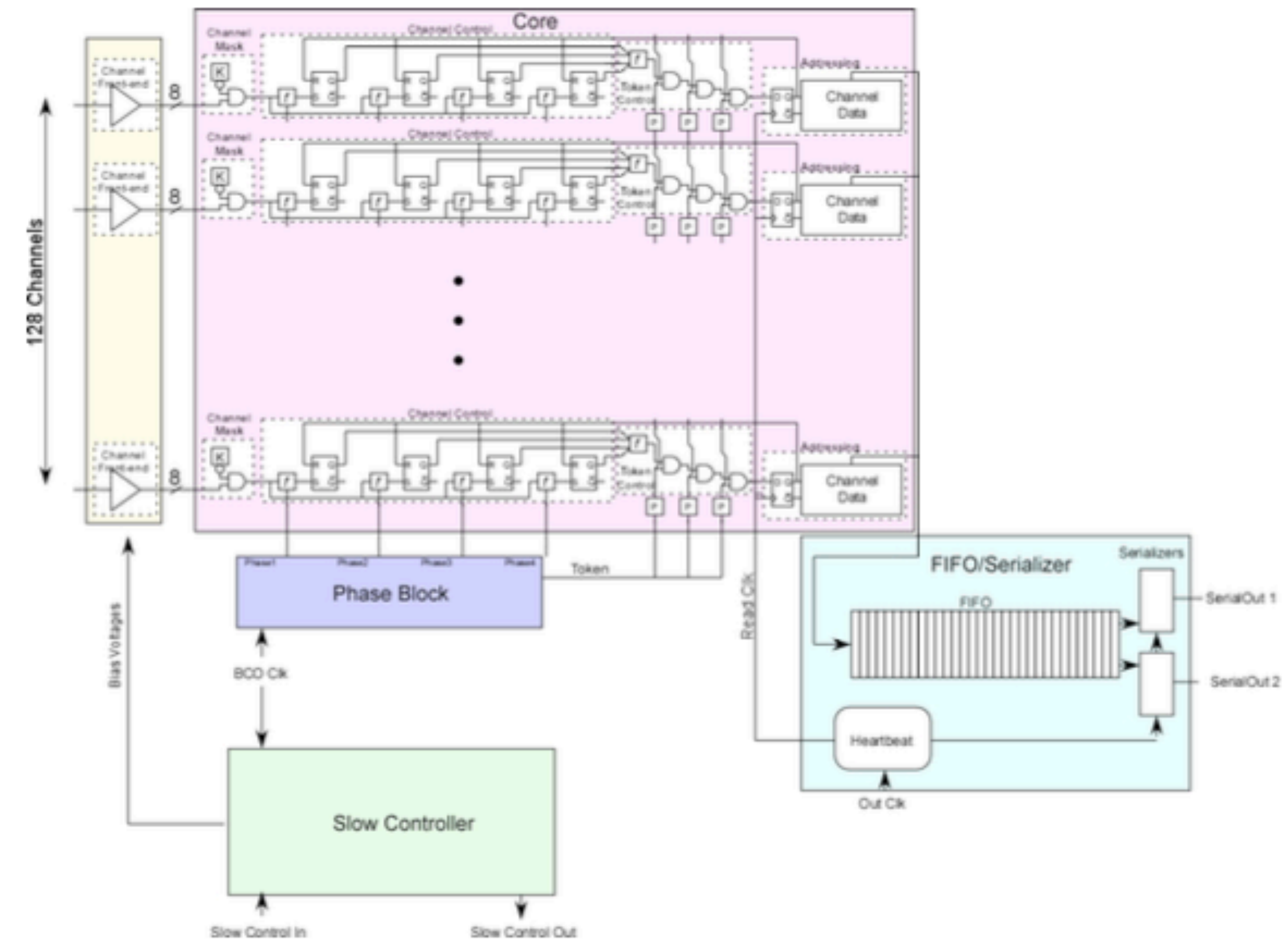


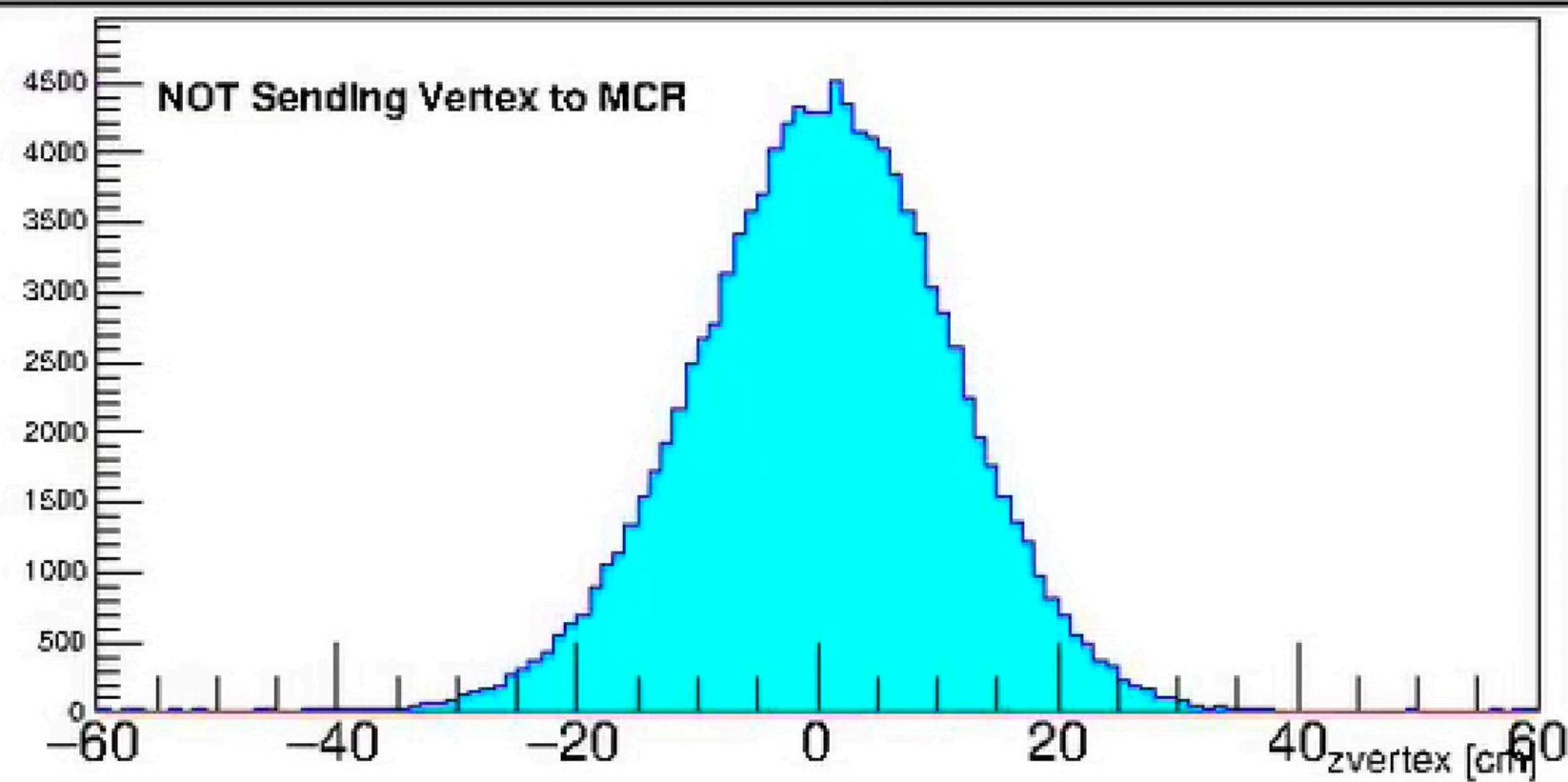
Figure 2 - The FPhx Block Diagram

# Run description - 54280

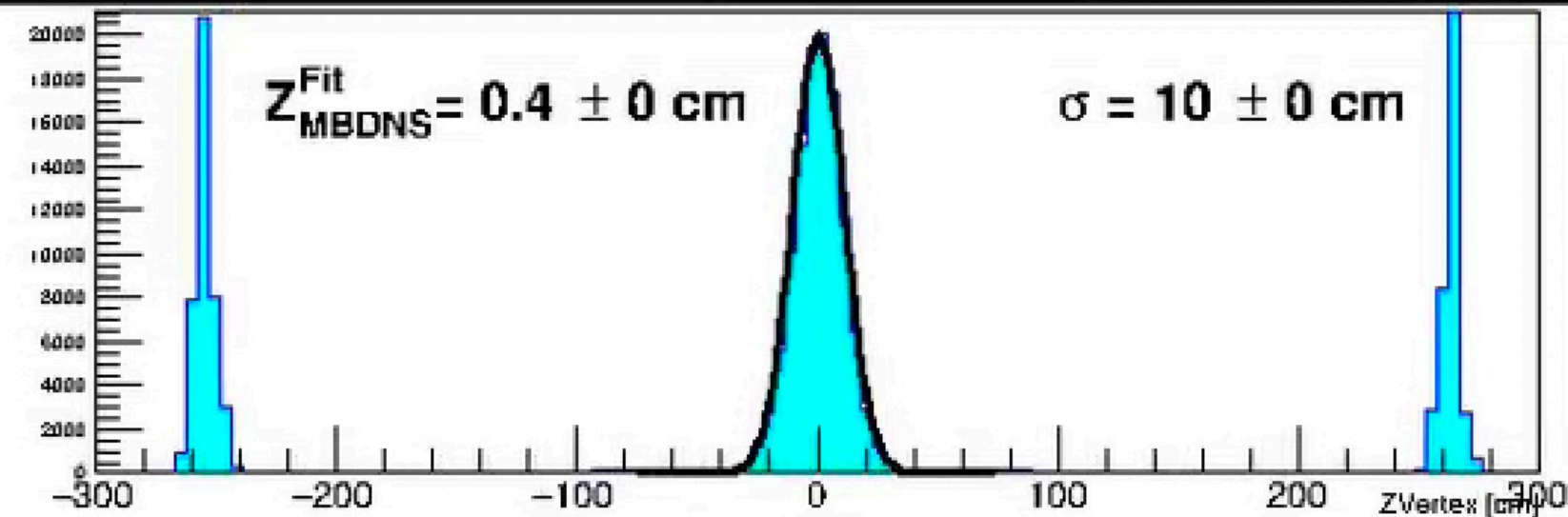
- Spike appears at each end of MBD
- The mini-bias definition is not yet available (as far as I know)
- Live trigger available to constraint the MBD vertex Z

Run #54280 Events: 204357 Date: Thu Oct 10 06:43:31 2011

MBD zvertex



MBD ZVertex (TRIG = MBDNS>=1)



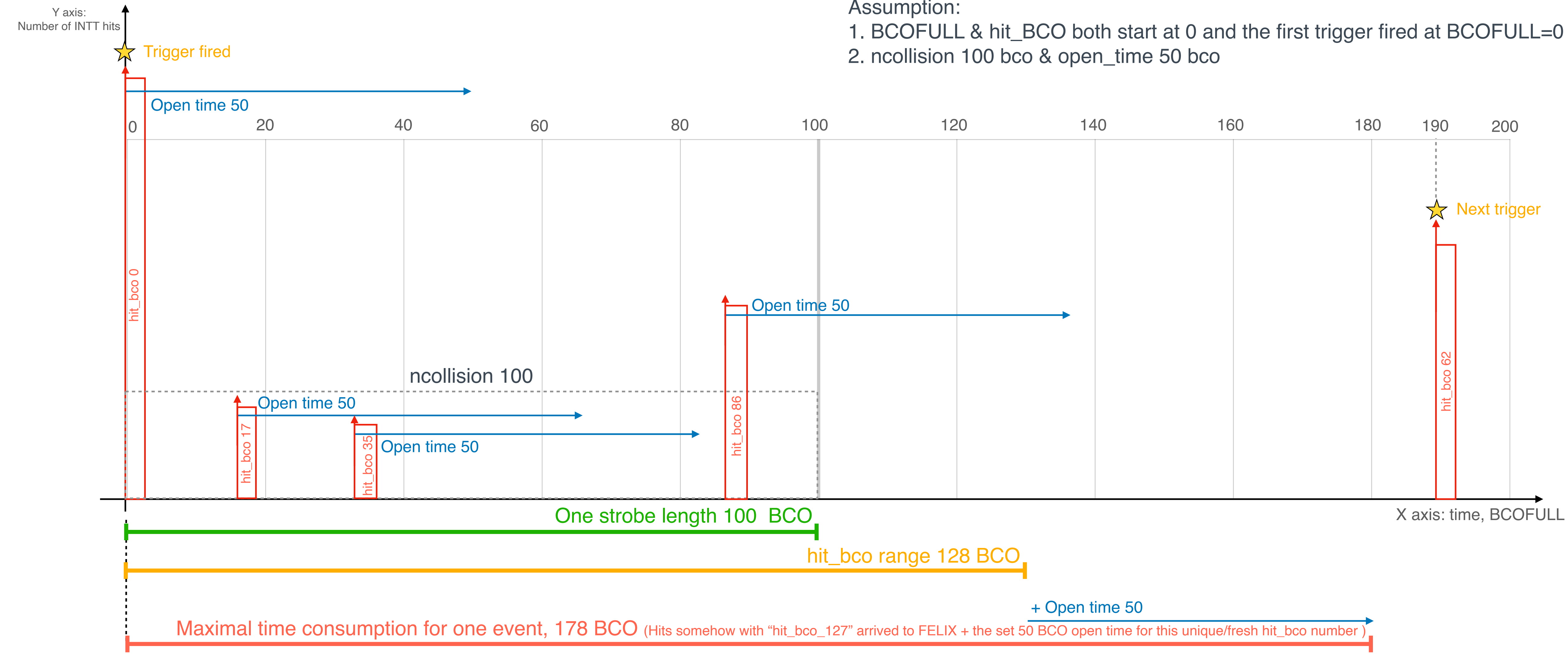
Trigger input channel	Name	enabled	Scaledown	Raw	Live $\langle \div \rangle$	Scaled	Live (%)
0	Clock	yes	93810	33836274325	33663041357	358838	99.5
1	ZDC South	yes	off	102829214	102308816	0	99.5
2	ZDC North	yes	off	98430768	95872319	0	97.4
3	ZDC Coincidence	yes	60	9417100	9370209	153672	99.5
4	HCAL Singles/Coincidence	yes	off	30282609	30125423	0	99.5
5		yes	off	33836274325	33663041357	0	99.5
6		yes	off	0	0	0	0
7		yes	off	0	0	0	0
8	MBD S >= 2	yes	off	86958423	86380777	0	99.3
9	MBD N >= 2	yes	off	85797943	85195687	0	99.3
10	MBD N&S >= 2	yes	0	10242665	10187457	10187457	99.5
11	MBD N&S >= 1	yes	off	18093659	17967450	0	99.3
12	MBD N&S >= 2, vtx < 10 cm	yes	off	4021509	4000602	0	99.5
13	MBD N&S >= 2, vtx < 30 cm	yes	off	5799143	5768655	0	99.5

Note: the hit transmission from chip to ROC: 1 hit / 1 bco

In single event

Assumption:

- 1. BCOFULL & hit\_BCO both start at 0 and the first trigger fired at BCOFULL=0
- 2. ncollision 100 bco & open\_time 50 bco



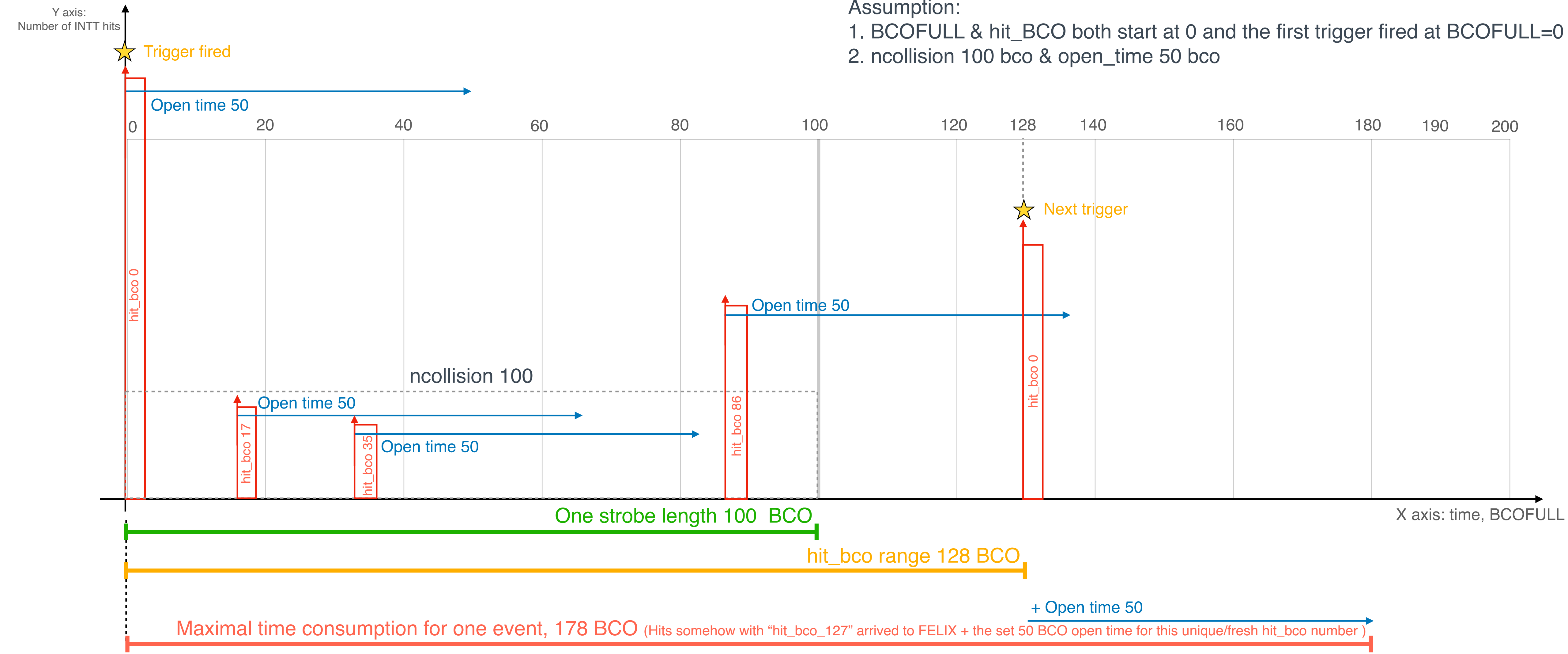
- Question 2.: As shown by cartoon, what if the "next trigger" is > 178 BCO away from the first trigger? (I assume this is the most safe case)

Note: the hit transmission from chip to ROC: 1 hit / 1 bco

In single event

Assumption:

- 1. BCOFULL & hit\_BCO both start at 0 and the first trigger fired at BCOFULL=0
- 2. ncollision 100 bco & open\_time 50 bco



- Question 3.: As shown in cartoon, what if we have hit\_bco\_0 in "this\_event", and the next trigger fired at "BCOFULL\_128 (hit\_bco\_0, again)". In addition, the FELIX is still taking the hits for hit\_bco\_86 for "this\_event". What will happen?