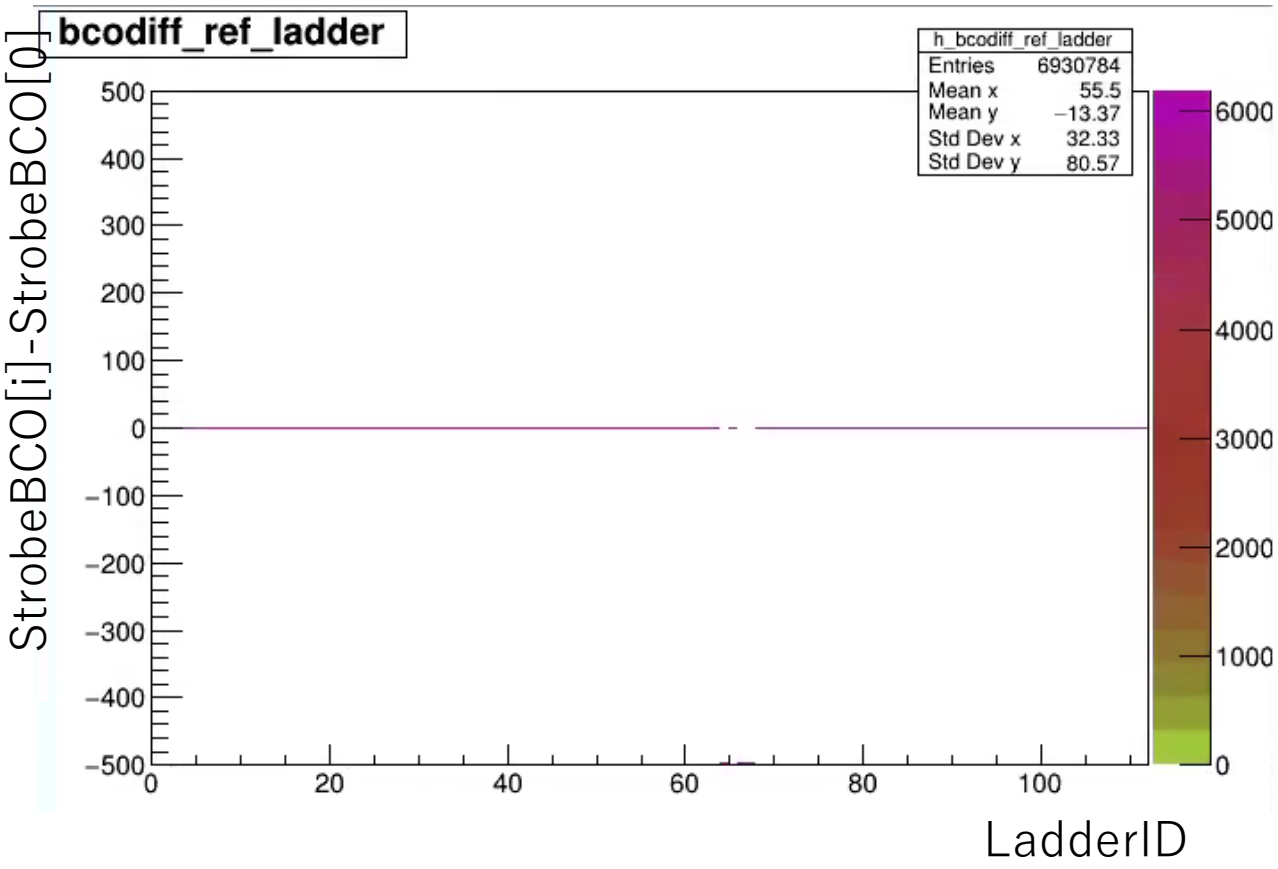
# Stream QA++

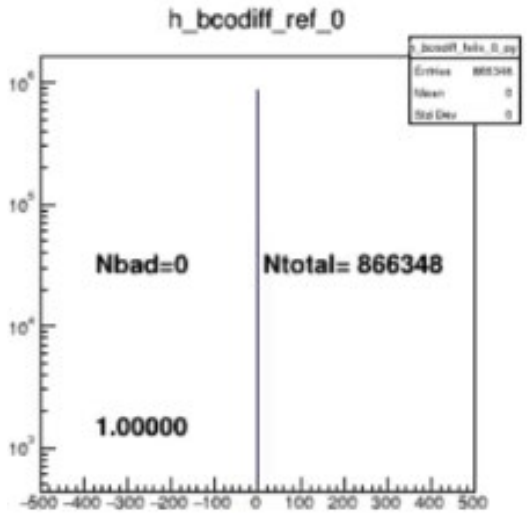
Takashi Hachiya
Nara Women's University

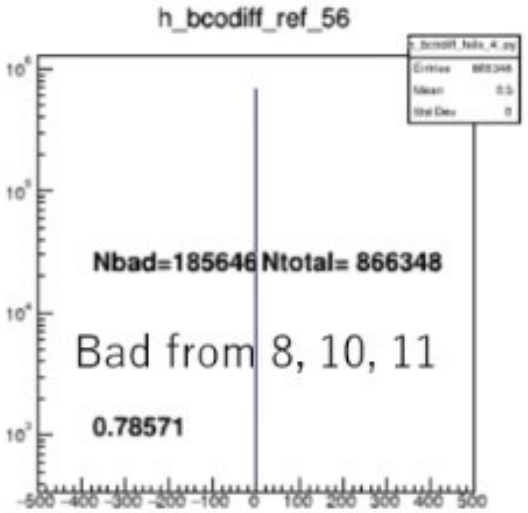
# Introduction

- 100% streaming coming. Verification that data is correct is necessary
  - All ladders receive the common BCO (no hit required)
  - Strobe is unique freq. (no hit required)
  - GL1 finding in INTT stream
  - Other issues
    - Production (Event combiner) is stuck at events ~78k

# Strobe BCO variation among 112 half ladders in stream data (run49660)

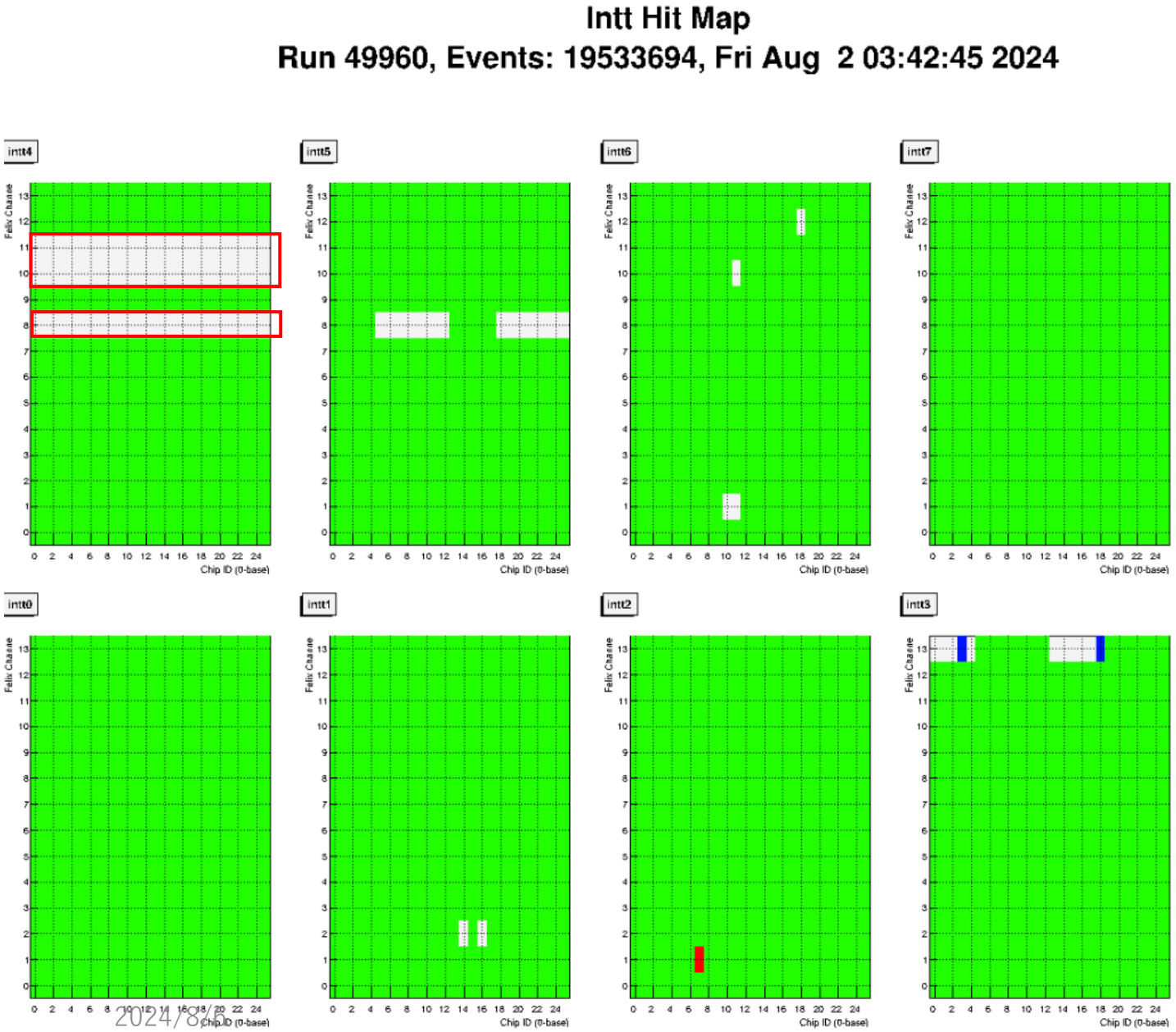


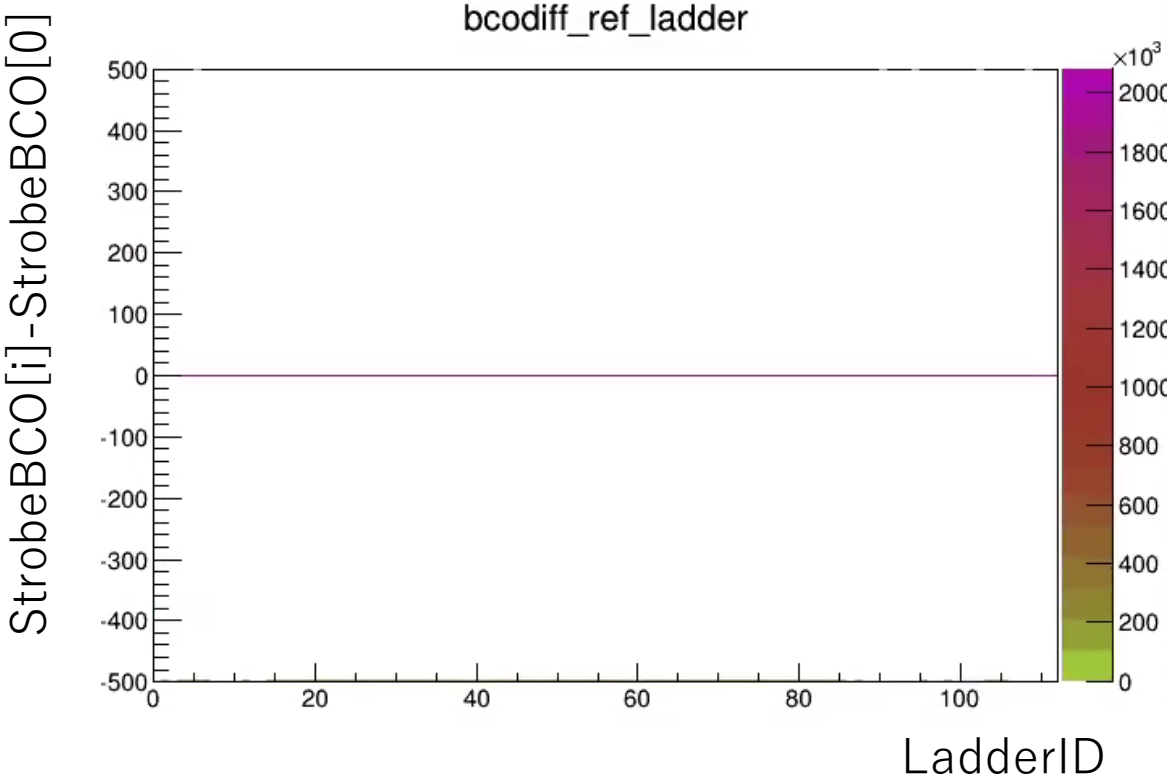

- All ladders have the same Strobe BCO (70k events analyzed)
  - My private decoder is used to get strobe BCO (no hit required)
  - Overflow/Underflow of the difference is shown at 499/-499 for clarity

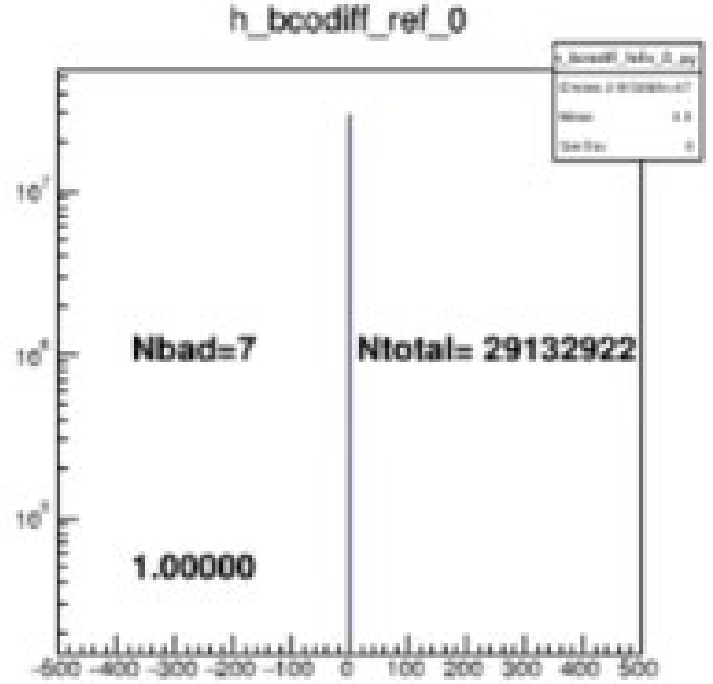
# Run 49660 streaming




- INTT4 8, 10, 11 are masked.
- Bad BCO comes from the masked ladders

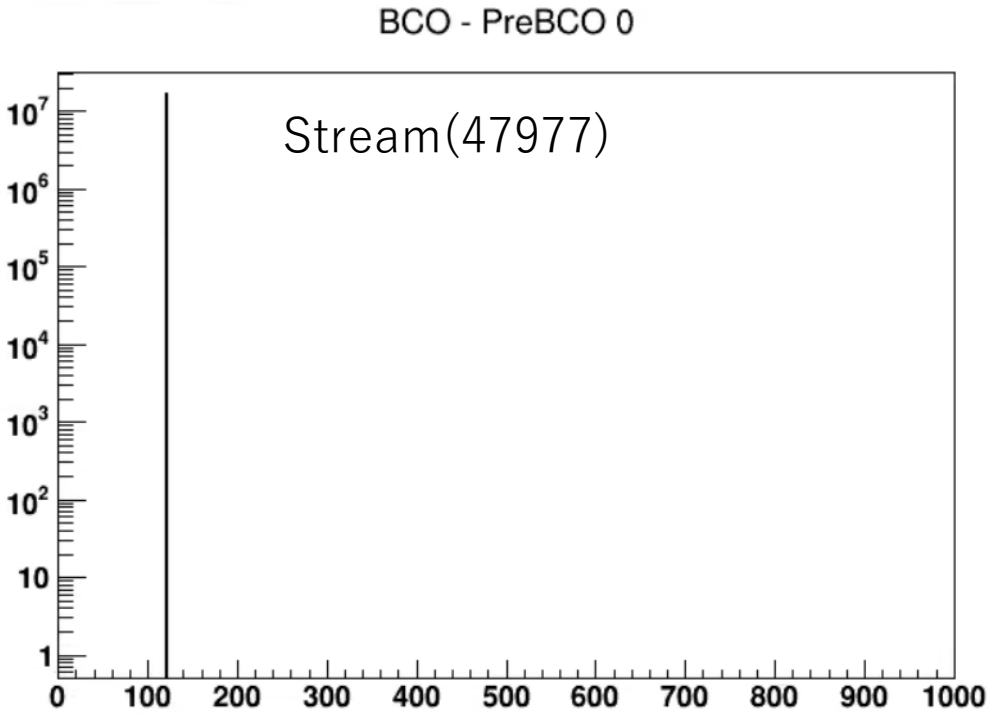
# GTM BCO variation among 112 half ladders in TRIGGERED data (run48007)

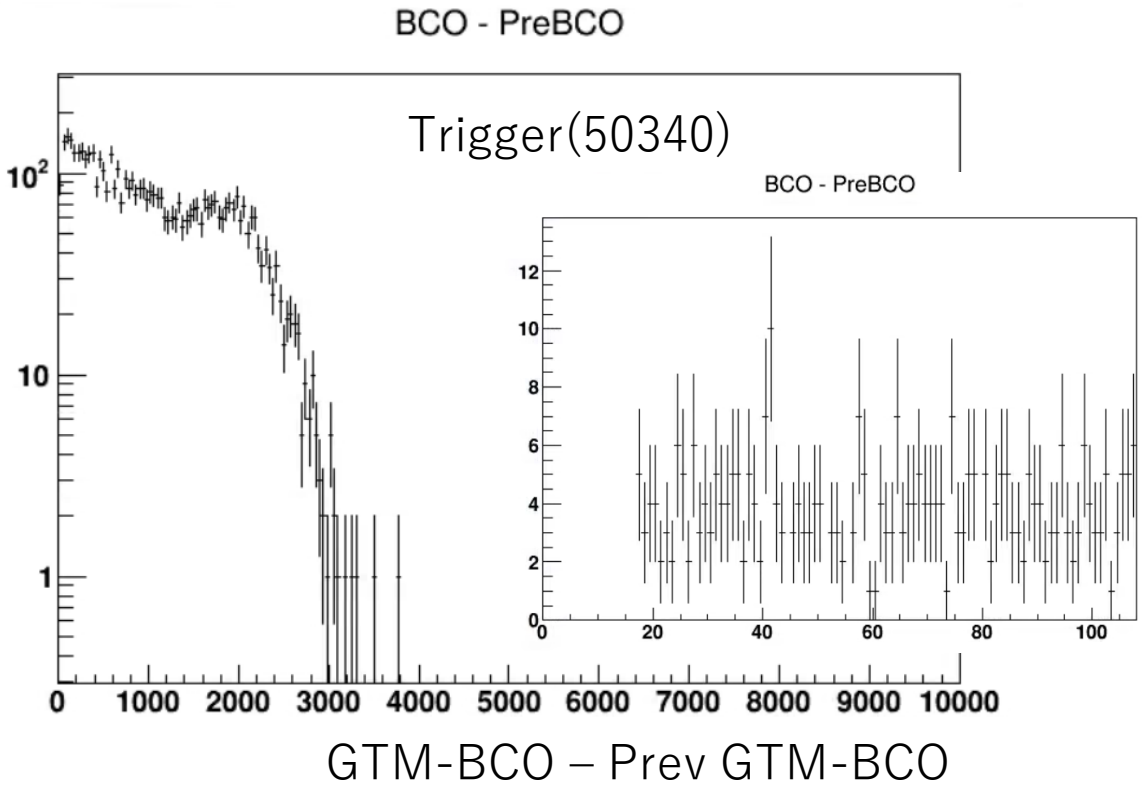



- Event where Difference = 0 comes from the last event or fragment of the event from previous run
  - N bad = 7 in this case

# BCO frequency

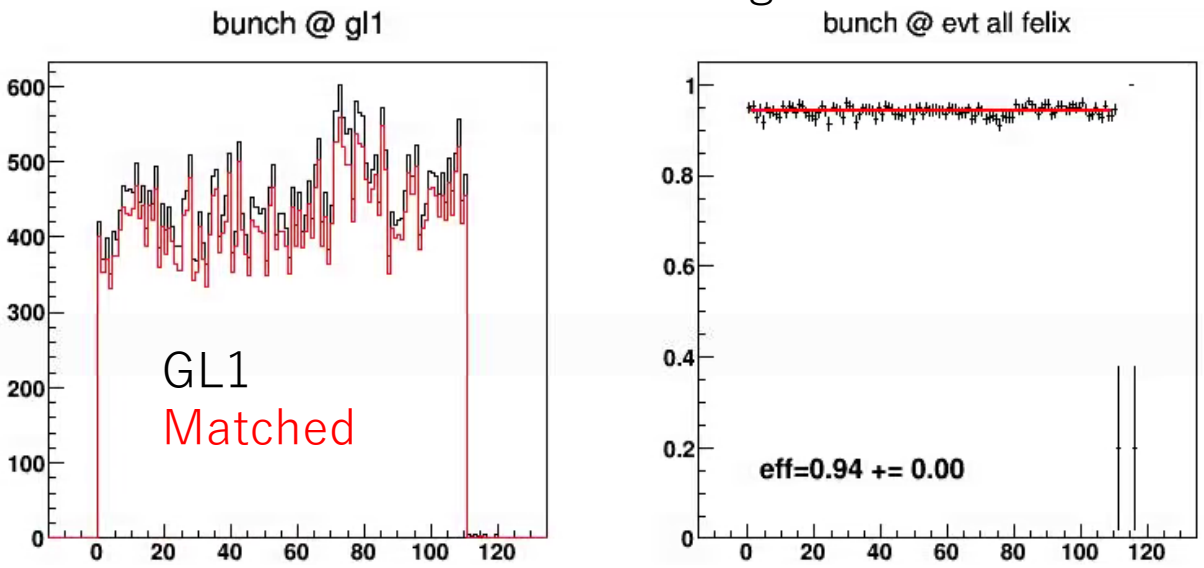



- Stream : Strobe freq = 120
- Trigger : GTM BCO freq = 16 ~ 3000 (dead 16x? in GL1)
  - No smaller than 16

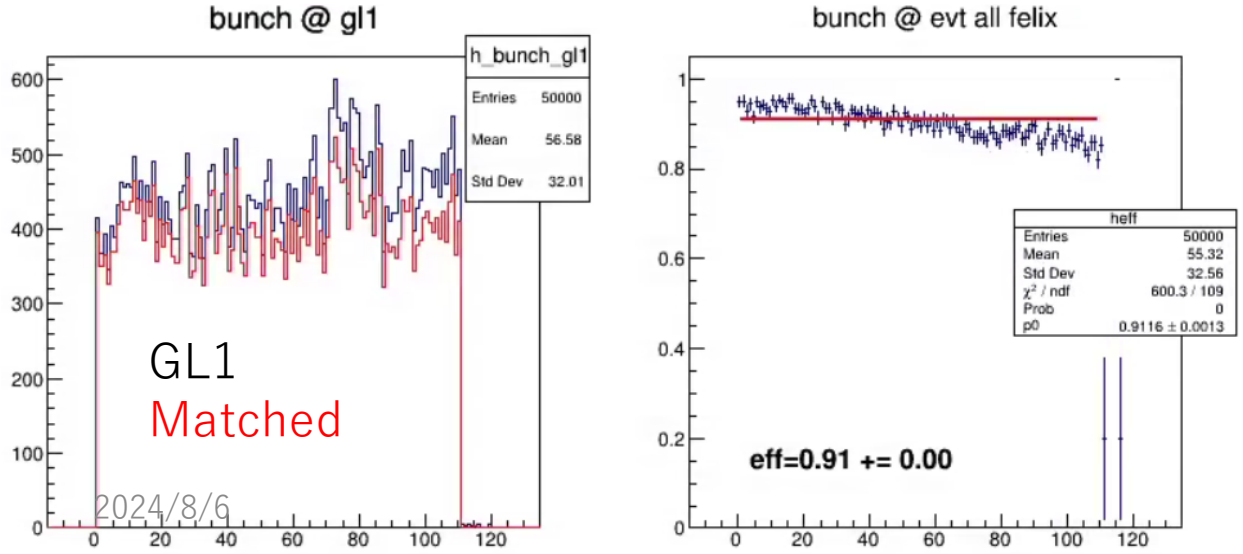
# GL1 finding efficiency by Fun4All

Takashi's GL1 finding code




- Efficiency is calculated using Fun4All
  - Efficiency ~ 94%
  - MC shows similar efficiency by Genki
    - PYTHIA + GEANT

Fun4All GL1 finding code




Results from F4A looks strange

- Slightly different with that from my version.
- Need to investigate

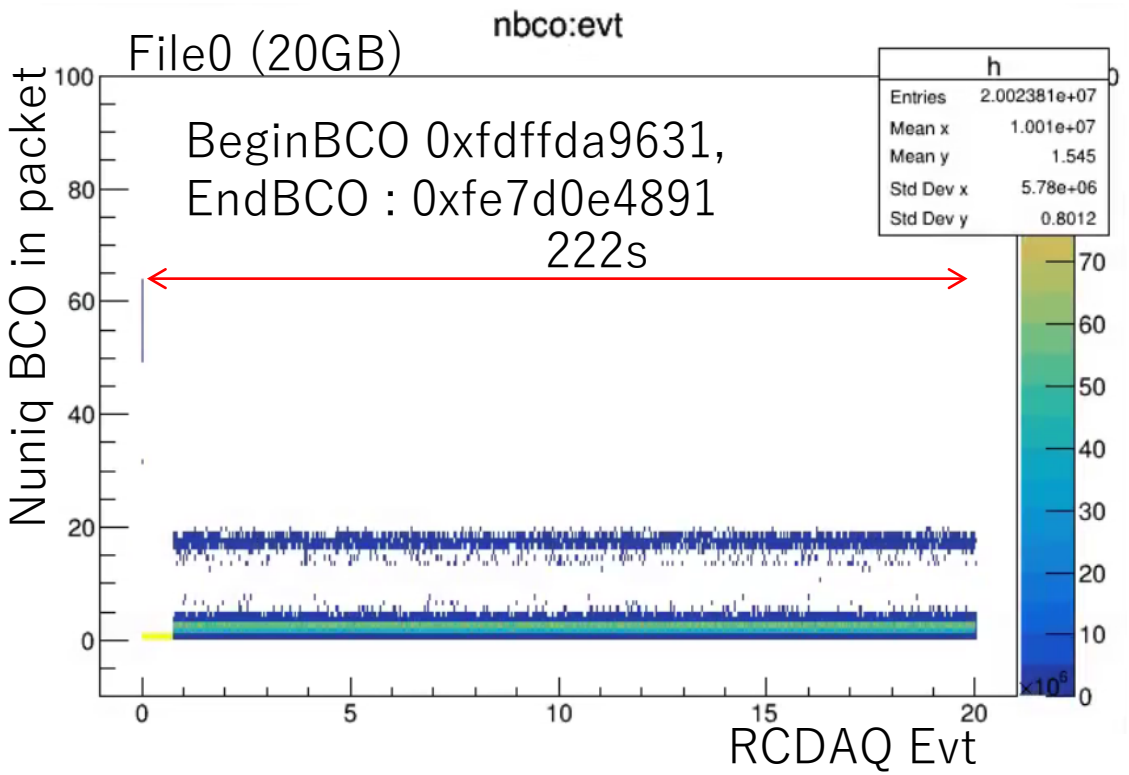
# Offline production issue for streaming data

- Event combiner (decoder) hanged up at event~78k always
  - Joseph and I are investigating this issue
- I analyzed run 49960,  6M events, 12m:50s, 4 files
- I used ddump to decode the evt file (event combiner not used)

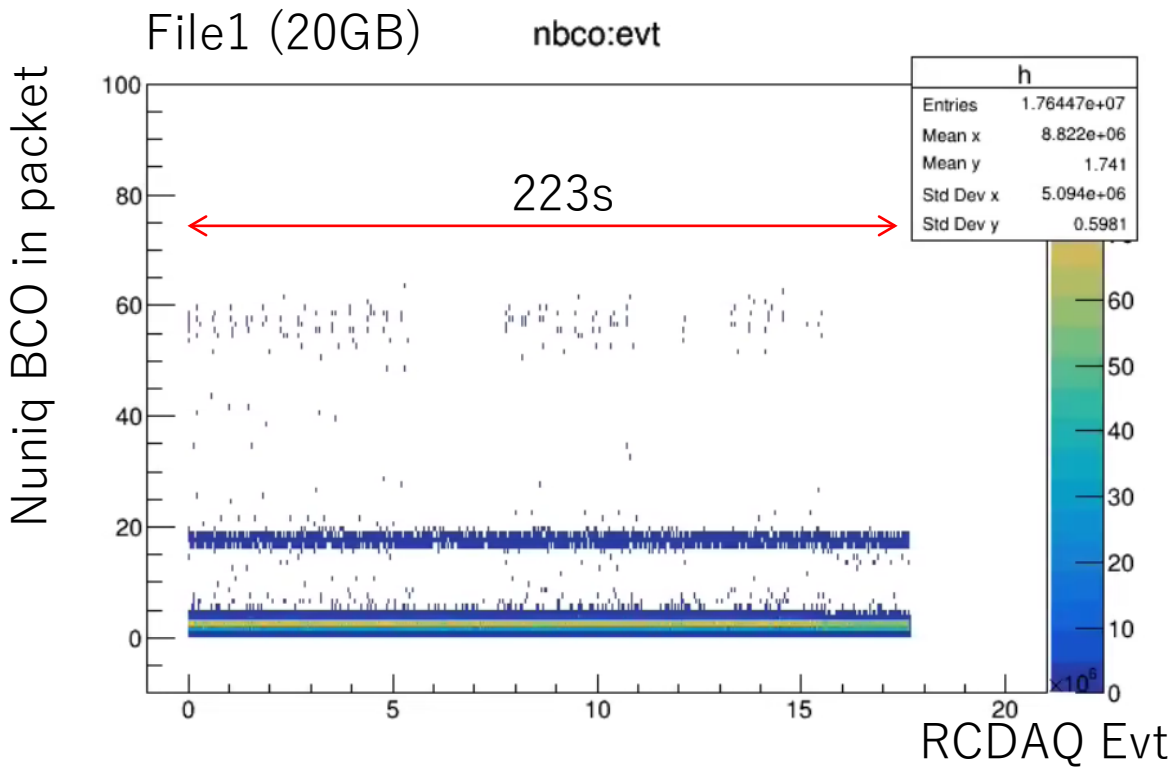
| | | | | | | |
|---|---|---|---|---|---|---|
| 49960 | physics | 2024-08-02 03:30:00 | 2024-08-02 03:42:50 | 6028353 | Trigger Info | Zero Suppression |

| Host: intt0 | Total Events: 65216349 | | | | |
|---|---|---|---|---|---|
| **Filename** | **Events** | **First Evt** | **Last Evt** | **in HPSS** | **in SDCC** |
| /bbox/bbox0/INTT/physics/physics_intt0-00049960-0000.evt | 20075803 | 1 | 20075803 | True | True |
| /bbox/bbox0/INTT/physics/physics_intt0-00049960-0001.evt | 17654566 | 20075804 | 37730369 | True | True |
| /bbox/bbox0/INTT/physics/physics_intt0-00049960-0002.evt | 18496515 | 37730370 | 56226884 | True | True |
| /bbox/bbox0/INTT/physics/physics_intt0-00049960-0003.evt | 8989465 | 56226885 | 65216349 | True | True |

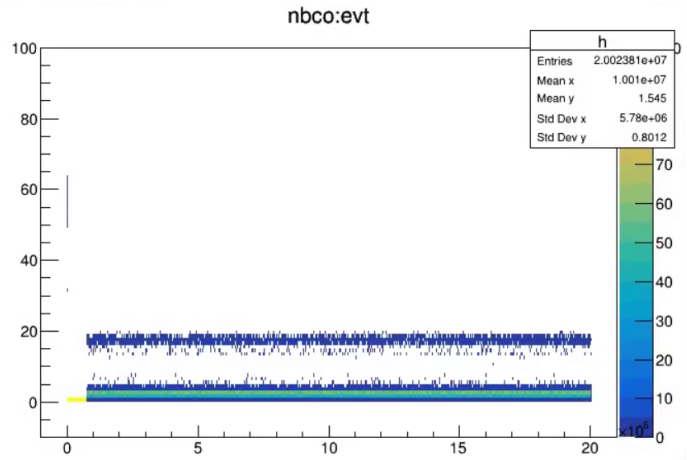
# Analysis by ddump

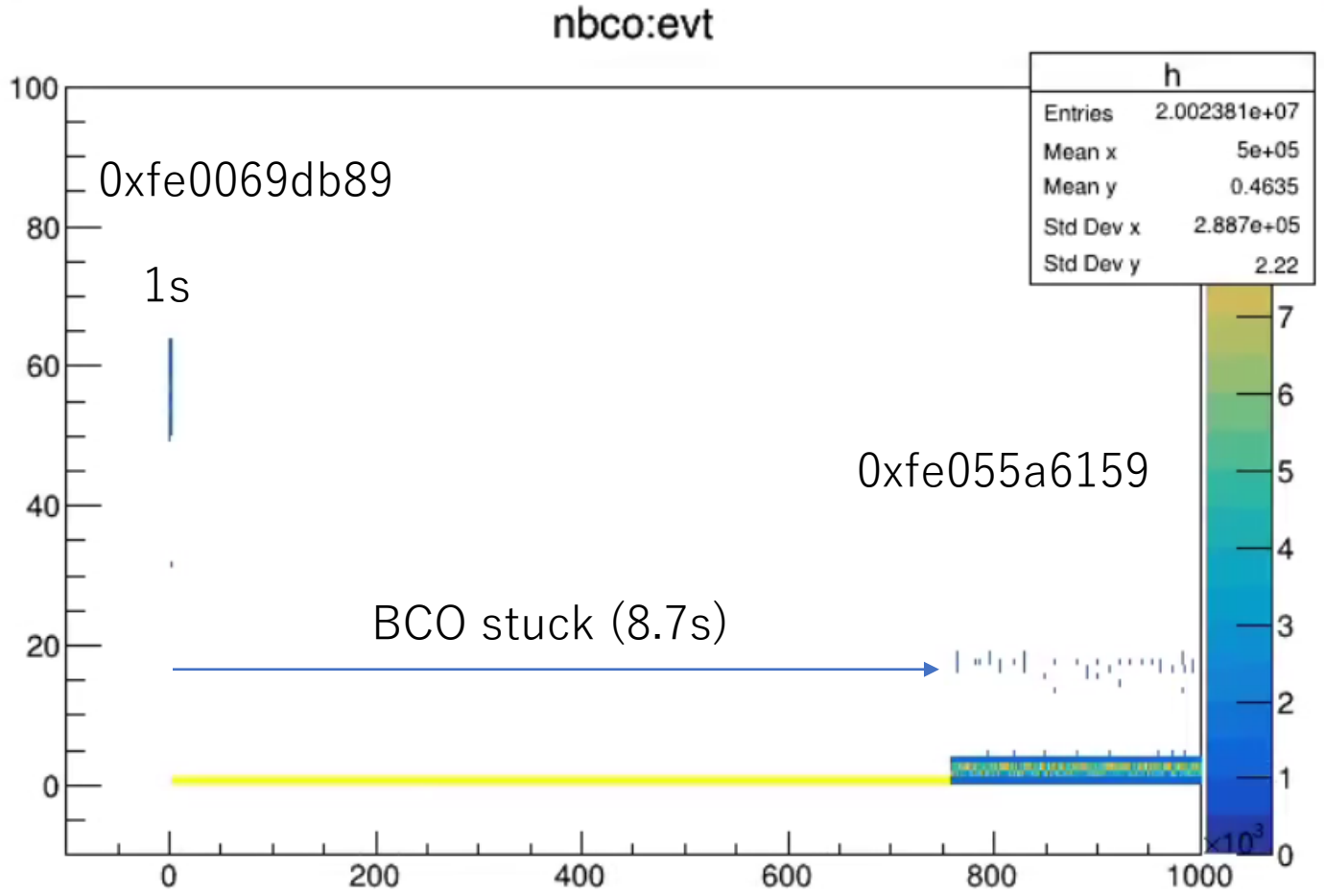



Begin 0xfe7d0e4891, end: 0xfefae9b561




- At the beginning of run, Nbco in packet ~ 60, then drop down to 0 , then going up to 1~20
  - Nbco in packet ~ 60 : 1s （up to 78k events)
  - Nbco in packet = 0 : 1-9s （no BCO found in ddump)
  - Nbco in packet = 1~20 : 9 – 222s
- This behavior seen in 1st file.




- This issue needs to be investigated in more detail
- BCO stuck (bad) but recovered later (good)

# Thought about this period

- What I saw
  - When I ran the fun4all event combiner, the process was always stuck at ~78k events. Process was killed by system after many hours
  - When I ran my private decoder, 30GB memory was used then killed by system
- From this, I suspect that BCO info may not be available in this stuck period
  - Both decoders look for the strobe/GTM BCO.
  - If next BCO is not found, all hits gets stored in the memory until next BCO is found.

**Fwd: ALERT: killed process with excessive memory footprint**

2024-08-04 12:06 に pinkenburg から

詳細 テキスト

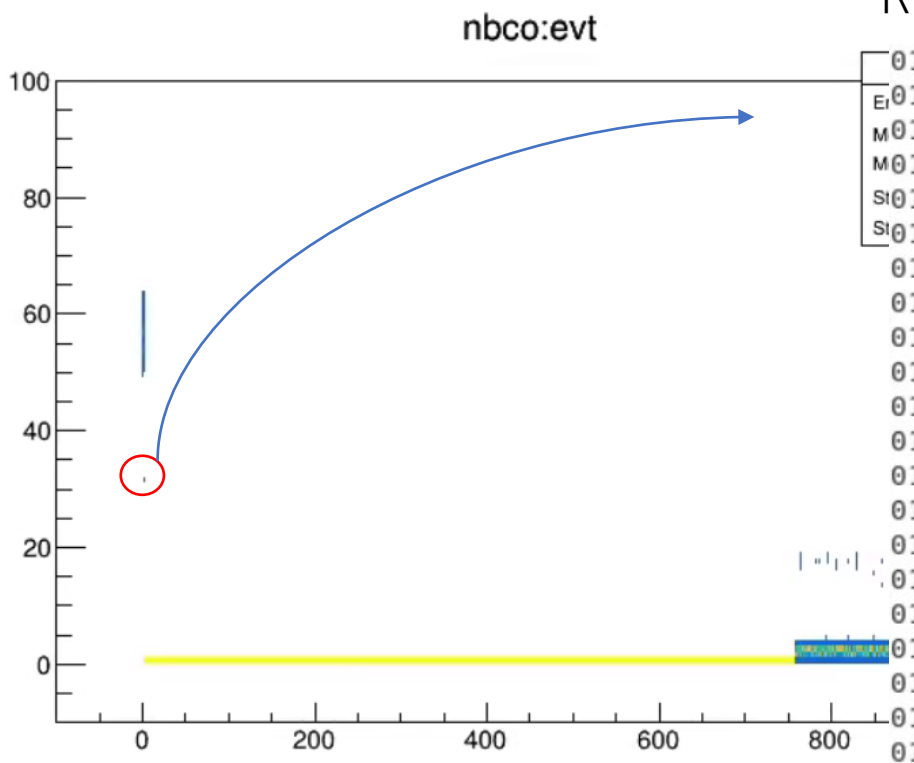
process_family.txt (~1015 バイト)

Hi Takashi,

I got a lot of these overnight. We have one node (sphnxdev01) which allows jobs up to 30GB for these cases. But you need to keep an eye on the job to make sure the machine doesn't start to swap. It looks like this happens during startup - if the job just reads until it gets killed, going to 30GB won't help

Chris

# No BCOFULL found in raw data in the yellow region




RCDAQ event=1415

```
0113500 f0afcaf0 511c0049 75020062 cafeff80
0113520 adfecade db890069 00010000 77890064
0113540 f0cfcaf0 d7fc0053 17fe0053 51ec0064
0113560 d1ef0064 f9ad0067 193b0073 193d0073
0113600 f0cfcaf0 2e8c0007 6d8d0007 2e330007
0113620 4e8b0007 2e8c0007 cd8d0007 8d8e0007
0113640 f0cfcaf0 2e8c0007 6d8d0007 2e330064
0113660 06680064 066f0064 26bf0007 26c00007
0113700 f0cfcaf0 2e8d0007 4e8e0007 6d8f0007
0113720 2d900007 4dc40007 4dc50007 4dc60007
0113740 f09fcaf0 98b70053 f2eb0053 f2ec0053
0113760 52ed0053 b2ee0064 53640007 14d80007
0114000 f0bfcaf0 4bf60013 cb760013 e3de0031
0114020 61ca0039 c1cb0039 01cc0039 623e0045
0114040 f05fcaf0 f5480023 f48d0023 59e7004e
0114060 52c6004e 93a80064 f3a90064 93d10064
0114100 f03fcaf0 2e620061 0eea0064 eeeb0064
0114120 73600064 57e40073 17e50073 953d000f
0114140 f0cfcaf0 4dc70007 2dc80007 2dc90007
0114160 4dca0007 4dcb0007 4dcc0007 0ec6000f
0114200 f00fcaf0 86020000 e6bd0000 4b8f000f
0114220 4ac4000f 0ac5000f 2b90000f 2b91000f
0114240 f06fcaf0 71aa0073 71ac0073 b1d00073
0114260 31d20073 f3e40009 f48d000f f48e000f
0114300 f08fcaf0 ba6e004e f6660050 95b90055
0114320 4dd30055 35ba0055 747d000f b39a000f
0114340 f0dfcaf0 c36c0050 a5d80055 45d90073
0114360 49760073 a9770007 e12a0007 413f000f
0114400 f01fcaf0 51a00073 11a30073 b45e0013
0114420 2e4b0013 ce4c0013 ee4d0013 ee4e0013
0114440 f03fcaf0 6e54000f 0f9a000f 0f9b000f
0114460 f53e000f 753f000f b1120016 91130016
0114500 f0cfcaf0 8ec7000f 8ec8000f 8ec9000f
```

Last BCO, after this, no BCO found in this event

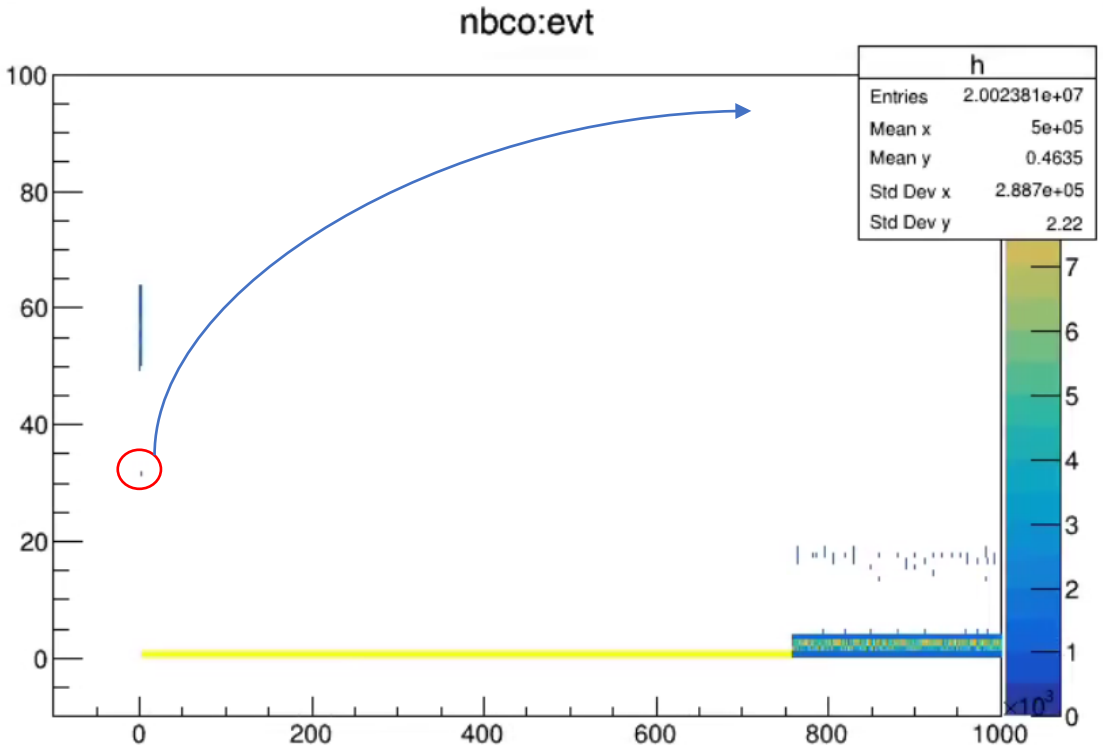
RCDAQ event=1416 (yellow region)

```
0000000 f0dfcaf0 03b2007a 63b30062 a69f0062
0000020 049d0077 251f0020 249e0020 c49f0020
0000040 f0afcaf0 f38c0053 f38d0053 538e0053
0000060 138f0053 8ec10071 34ee0028 f4ef0028
0000100 f08fcaf0 6e010068 4e020068 8fe00028
0000120 efe10028 efe20028 738b0028 338f0028
0000140 f06fcaf0 f007007a f11b0033 17c30045
0000160 d7c40045 f7f6006f 75550071 75560071
0000200 f09fcaf0 1a480025 ba49003c 71f60053
0000220 f3af0071 34930071 74bb0071 f4d3000b
0000240 f05fcaf0 b6650071 10860009 10870009
0000260 30880009 30890009 308d0009 108e0009
0000300 f00fcaf0 42490071 47ec0077 27ed0077
0000320 2a310077 4a320077 c7c4007e 6d1e0028
0000340 f07fcaf0 09990077 499e0077 23190077
0000360 02380077 42390077 29a10077 66630028
0000400 f0dfcaf0 e5200020 452e0020 84a00020
0000420 4d670020 a7a80028 c7a90030 85d70030
0000440 f03fcaf0 141f0077 11b20077 541e0077
0000460 ef620009 ef630009 325e0038 525f0038
0000500 f00fcaf0 a4fb0038 64fc0038 4b7b0038
0000520 eb7c0038 64fd0038 64fe0038 8b7d0038
0000540 f03fcaf0 72600038 52610038 52620038
0000560 72630038 72640038 92650038 92660038
0000600 f0bfcaf0 07340009 e4500009 24510009
0000620 6d260028 0bc20040 2bc30040 4c420040
0000640 f03fcaf0 72670038 72680038 72690038
0000660 726a0038 726b0038 926c0038 726d0038
0000700 f0bfcaf0 28a80040 48a90040 85720040
0000720 8be10040 ebe20040 65740040 45750040
0000740 f07fcaf0 06930039 012d0039 012e0039
0000760 01c80039 06970039 06950039 01c90039
```

There are data but No BCO

- “f0XYcaf0” is the header of FELIX payload
  - X : ladder ID
  - Y: Payload size always 0, = size = 7
- “cade” is the hit header, next word is BCOFULL
  - DATA is little endian

# Thought from the raw data




- No BCO is found in the yellow region. But some data there
  - No “cade” header (hit header)
- But data (payload) exist in the FELIX (payload always comes)
- Looks like “running crazy”
  - Saw similar behavior such as calibration mode last year

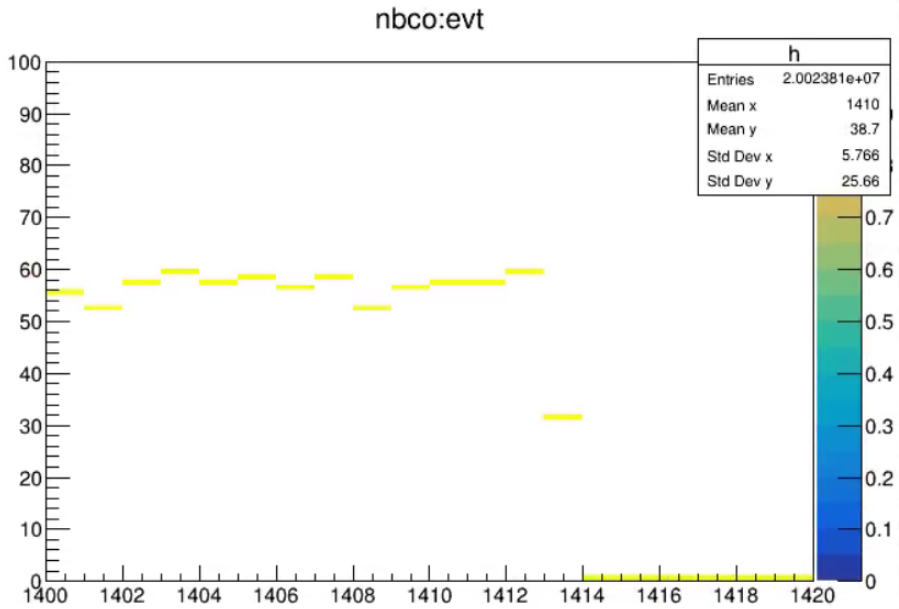
# How to make plot / check raw data

- Data:
  - “/sphenix/lustre01/sphnxpro/physics/INTT/physics/physics_intt0-00049960-0000.evt”
- Method
  - ddump to make text file (because fun4all decoder doesn’t work)
    - ddump -n 0 -f /sphenix/lustre01/sphnxpro/physics/INTT/physics/physics_intt0-00049960-0000.evt > data.txt
  - Extract particular character (I used “c++” code, need to compile)
    - /gpfs/mnt/gpfs02/sphenix/user/hachiya/INTT/INTT/general_codes/hachiya/InttEventTree/logana/logana.cc
    - ./a.out > data_bco.txt
  - Extract more
    - grep “Number” data_bco.txt > uniqbco.txt
  - Run macro to make TTREE
    - root –b –q  plot_uniqbco.C  -> uniqbco.root created
  - Make plot using TTREE
    - in uniqbco.root, ntp->Draw(“nbco:nevt”) // use draw option if you want
  - Find boundary of the “yellow” region by eye
- DDUMP for particular event
  - ddump -e 1415 -f /sphenix/lustre01/sphnxpro/physics/INTT/physics/physics_intt0-00049960-0000.evt | less
- DDUMP  binary data (you can see raw data w/ HEX format)
  - ddump -e 1417 -g -s -f /sphenix/lustre01/sphnxpro/physics/INTT/physics/physics_intt0-00049960-0000.evt | od -t x4 | less

# Summary

- What found
  - All ladder have the same BCO
  - Strobe BCO is always 120
  - GL1 finding efficiency ~ 94% but not consistent w/ Fun4All method
  - Event combiner stuck
  - Strobe BCO in stream data stuck and recovered later

# 1415




- 最後のBCOの後、CADEは無くなる。　しかし、データ自身はある。以前あった、CALIB時の大量データと同じように、見える
  - 発狂するときと同じ

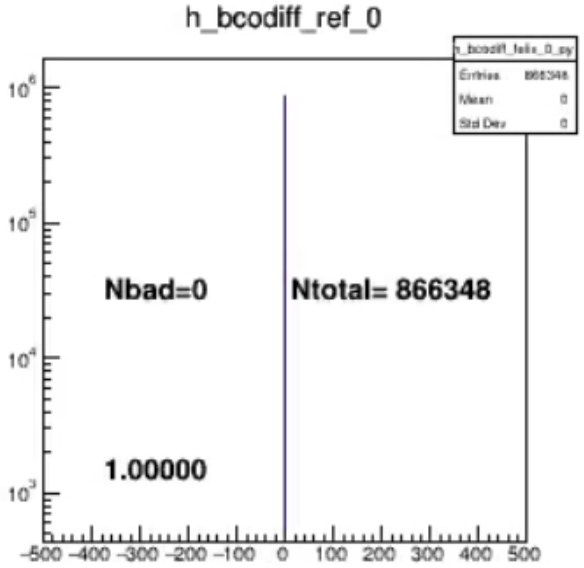
RCDAQ_Evt=1415

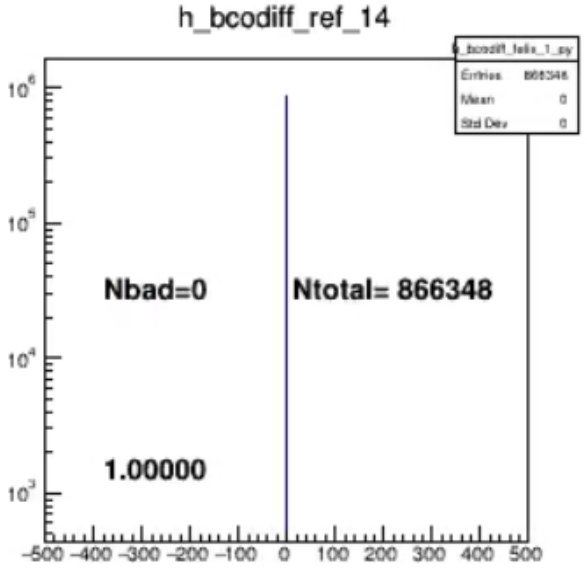
```
Packet  3001 16388 -1 (sPHENIX Packet) 110 (IDINTTV0)
  Number of unique BCOs: 31
   0 0xfe0069cd79
   1 0xfe0069cdf1
   2 0xfe0069ce69
   3 0xfe0069cee1
   4 0xfe0069cf59
   5 0xfe0069cfd1
   6 0xfe0069d049
   7 0xfe0069d0c1
   8 0xfe0069d139
   9 0xfe0069d1b1
  10 0xfe0069d229
  11 0xfe0069d2a1
  12 0xfe0069d319
  13 0xfe0069d391
  14 0xfe0069d409
  15 0xfe0069d481
  16 0xfe0069d4f9
  17 0xfe0069d571
  18 0xfe0069d5e9
  19 0xfe0069d661
  20 0xfe0069d6d9
  21 0xfe0069d751
  22 0xfe0069d7c9
  23 0xfe0069d841
  24 0xfe0069d8b9
  25 0xfe0069d931
  26 0xfe0069d9a9
  27 0xfe0069da21
  28 0xfe0069da99
  29 0xfe0069db11
  30 0xfe0069db89
  Number of hits: 12469
```

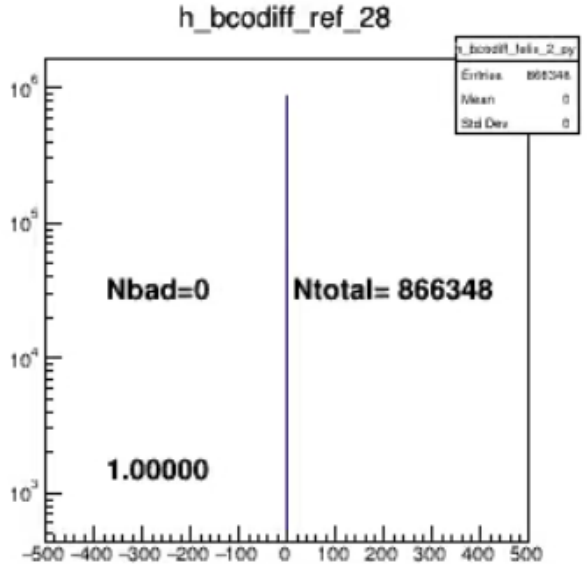
1415 last BCOFULL after this, no BCO FULL found

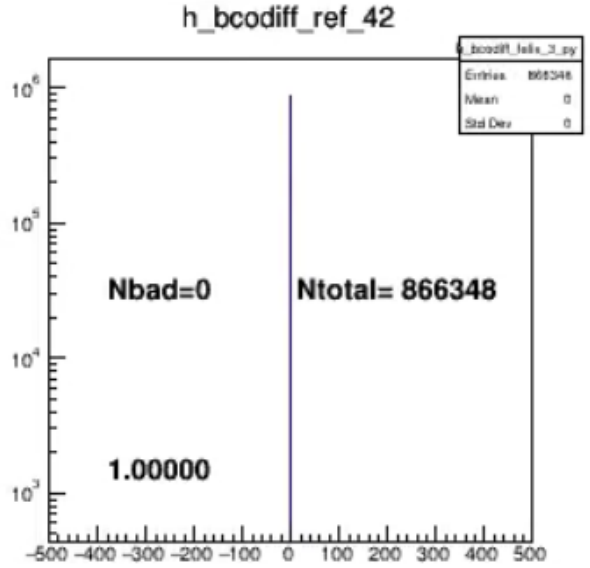
```
0113500 f0afcaf0 511c0049 75020062 cafeff80
0113520 adfecade db890069 00010000 77890064
0113540 f0cfcaf0 d7fc0053 17fe0053 51ec0064
0113560 d1ef0064 f9ad0067 193b0073 193d0073
0113600 f0cfcaf0 193e0073 6d8c0007 2e330007
0113620 4e8b0007 2e8c0007 cd8d0007 8d8e0007
0113640 f04fcaf0 265f0064 4ceb0064 06600064
0113660 06680064 066f0064 26bf0007 26c00007
0113700 f0cfcaf0 2e8d0007 4e8e0007 6d8f0007
0113720 2d900007 4dc40007 4dc50007 4dc60007
0113740 f09fcaf0 98b70053 f2eb0053 f2ec0053
0113760 52ed0053 b2ee0064 53640007 14d80007
0114000 f0bfcaf0 4bf60013 cb760013 e3de0031
0114020 61ca0039 c1cb0039 01cc0039 623e0045
0114040 f05fcaf0 f5480023 f48d0023 59e7004e
0114060 52c6004e 93a80064 f3a90064 93d10064
0114100 f03fcaf0 2e620061 0eea0064 eeeb0064
0114120 73600064 57e40073 17e50073 953d000f
0114140 f0cfcaf0 4dc70007 2dc80007 2dc90007
0114160 4dca0007 4dcb0007 4dcc0007 0ec6000f
0114200 f00fcaf0 86020000 e6bd0000 4b8f000f
0114220 4ac4000f 0ac5000f 2b90000f 2b91000f
0114240 f06fcaf0 71aa0073 71ac0073 b1d00073
0114260 31d20073 f3e40009 f48d000f f48e000f
0114300 f08fcaf0 ba6e004e f6660050 95b90055
0114320 4dd30055 35ba0055 747d000f b39a000f
0114340 f0dfcaf0 c36c0050 a5d80055 45d90073
0114360 49760073 a9770007 e12a0007 413f000f
0114400 f01fcaf0 51a00073 11a30073 b45e0013
0114420 2e4b0013 ce4c0013 ee4d0013 ee4e0013
0114440 f03fcaf0 6e54000f 0f9a000f 0f9b000f
0114460 f53e000f 753f000f b1120016 91130016
0114500 f0cfcaf0 8ec7000f 8ec8000f 8ec9000f
```

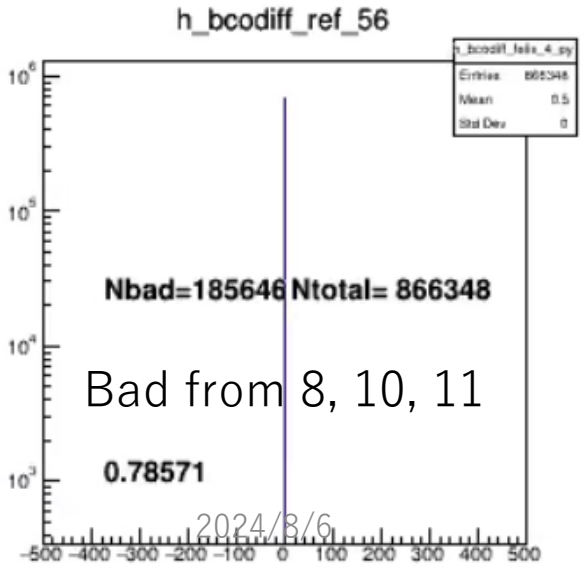
# Run 49660 streaming (68k evts)

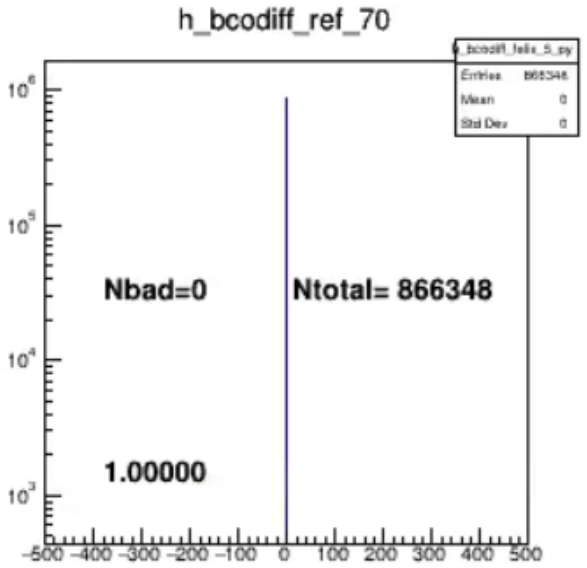


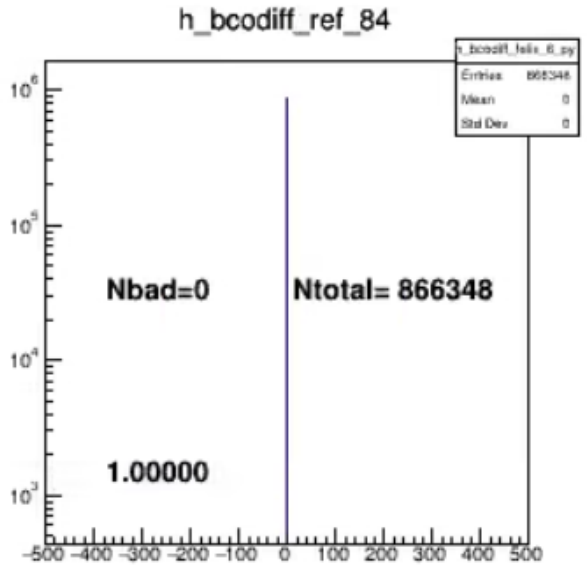



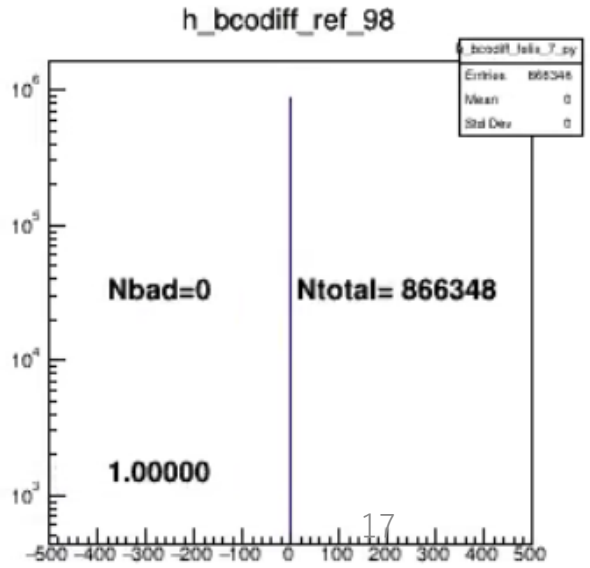



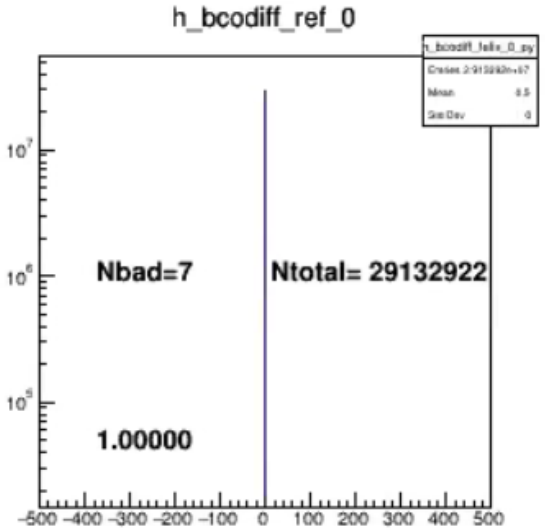



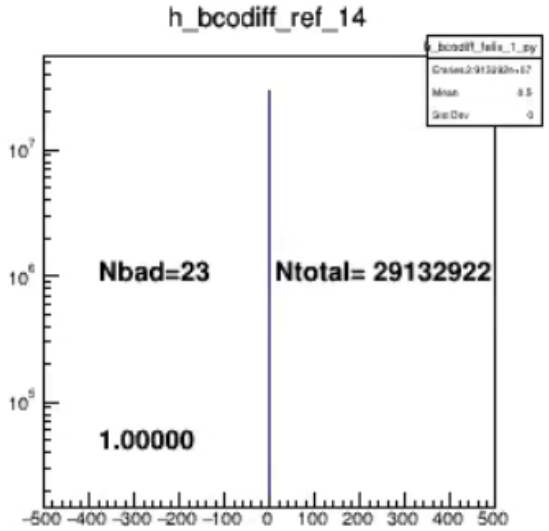



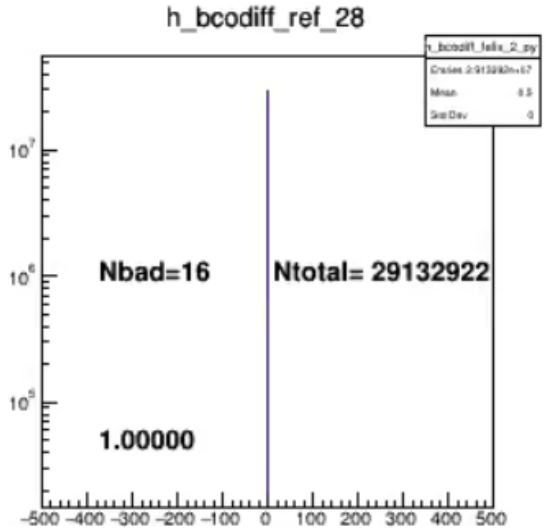



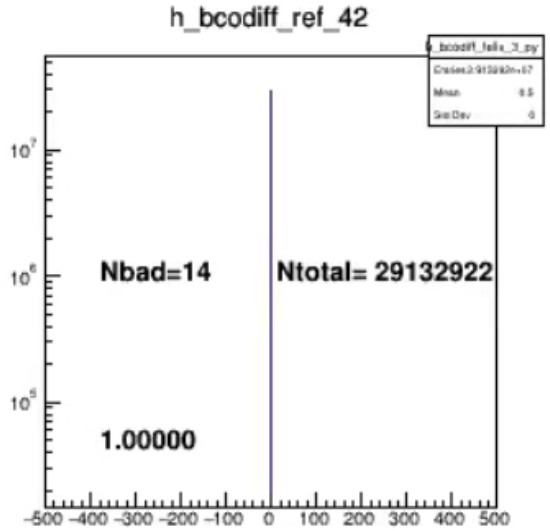

Bad from 8, 10, 11

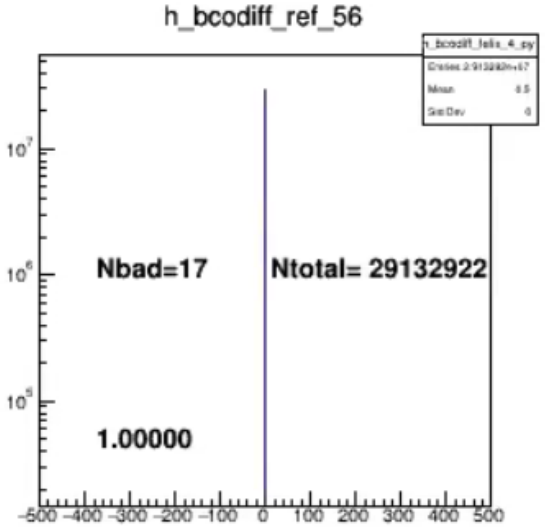
# Triggered

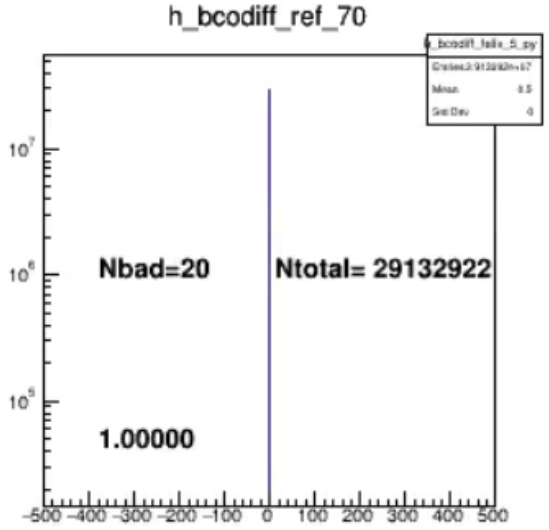


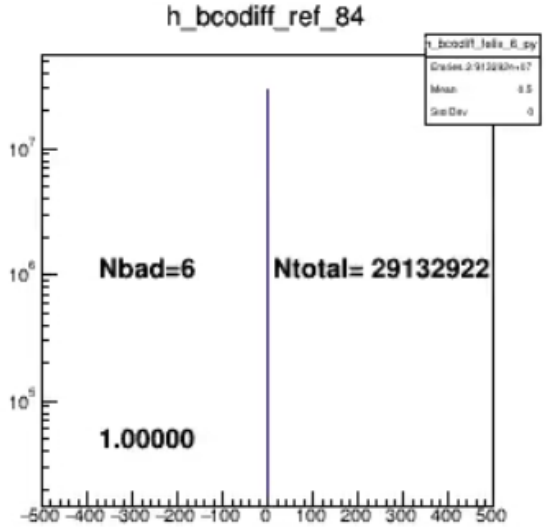



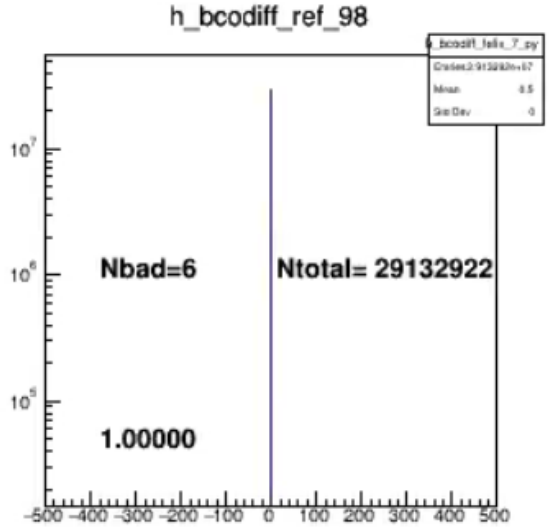



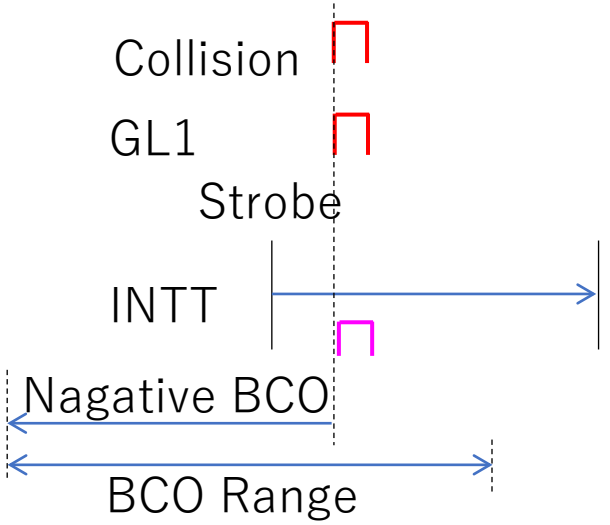

- Nbad appeared at event=-1 (fragment of the last run) and the last event based on the log file

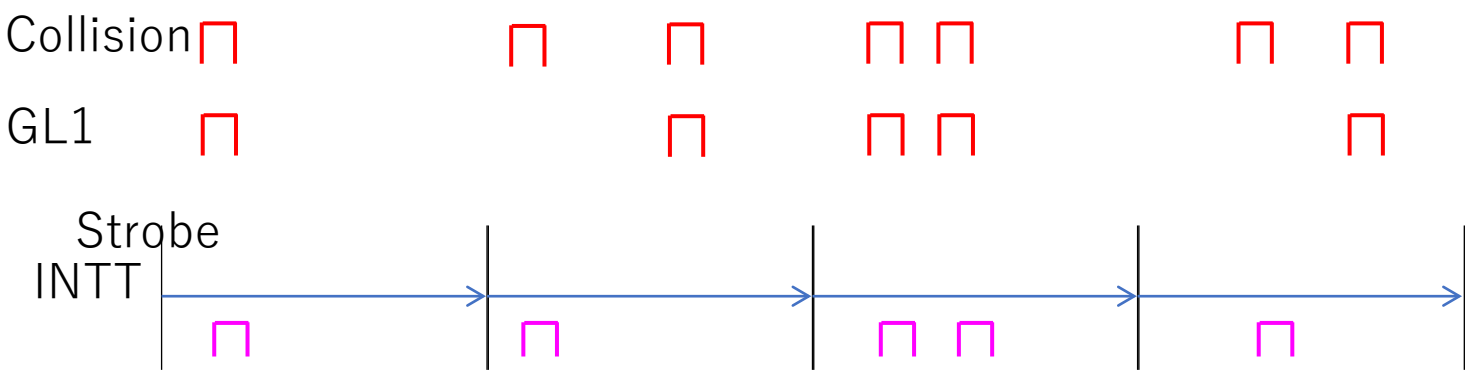
# backup

# 1 BCO DIFF QA plot as offline QA

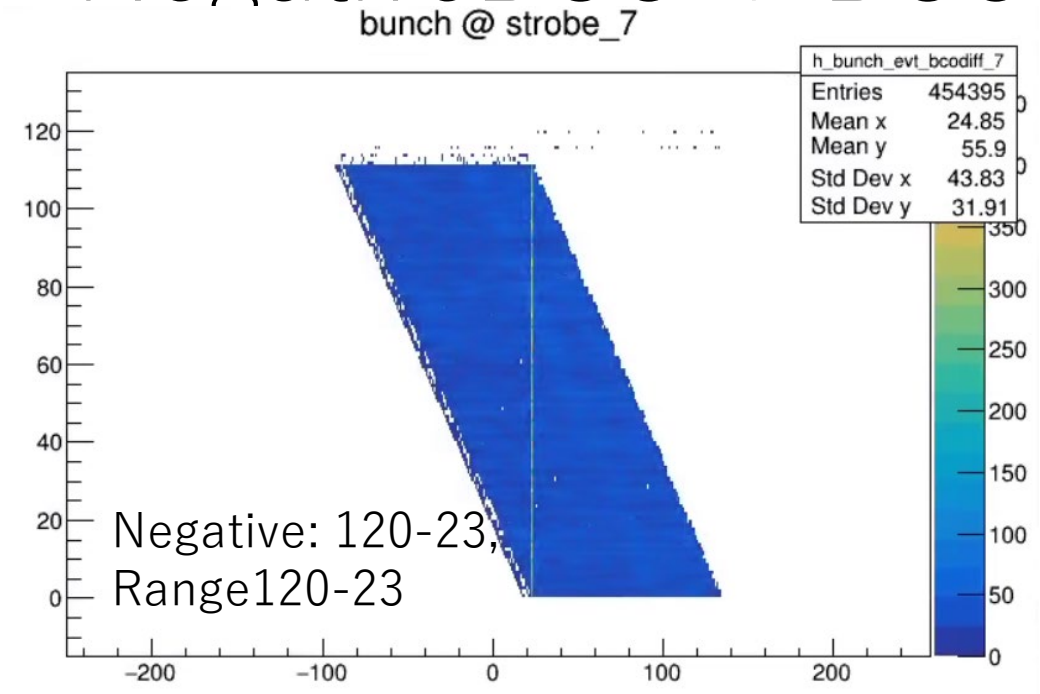
- BCO diff plots are required as offline QA
- For the streaming mode, I thought it cannot be made using F4A, but I found I can do it.

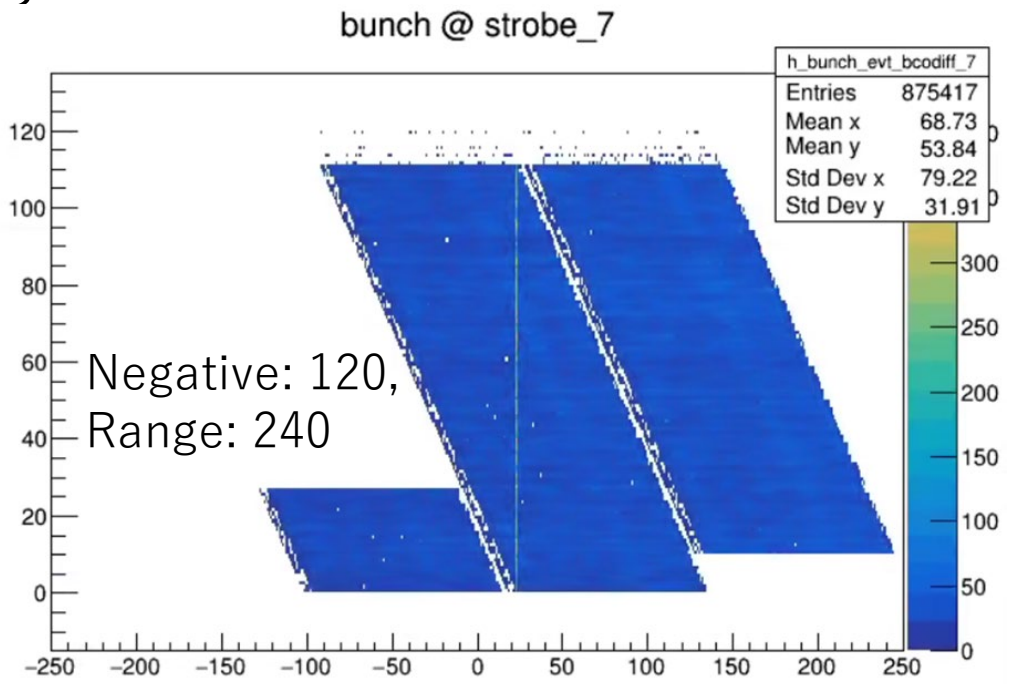
```
SingleInttPoolInput *intt_sngl= new SingleInttPoolInput("INTT_" + to_string(i));
intt_sngl->SetNegativeBco(120-23);
intt_sngl->SetBcoRange(120); // 128 + 256
intt_sngl->AddListFile(iter);
in->registerStreamingInput(intt_sngl, InputManagerType::INTT);
```

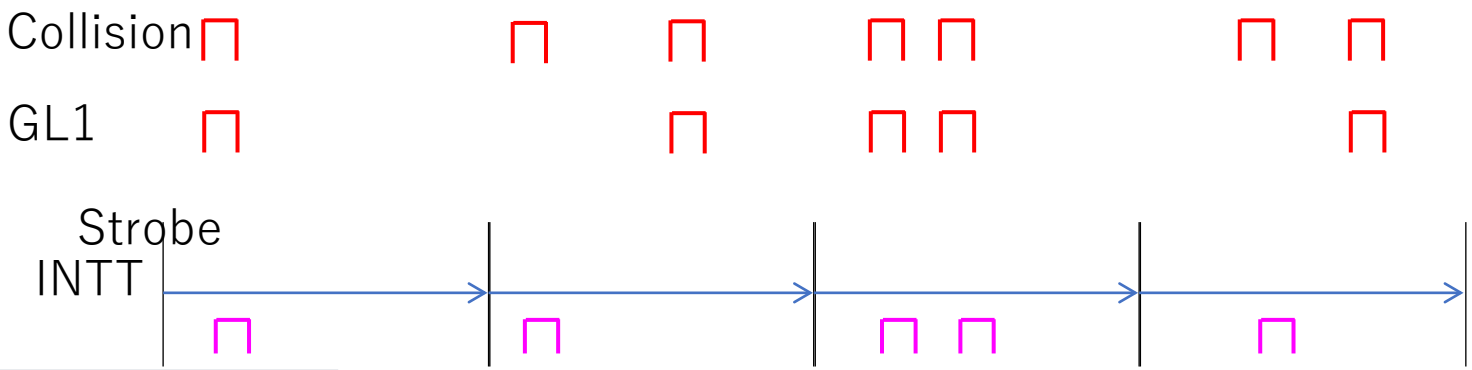


# NegativeBCO + BCORange

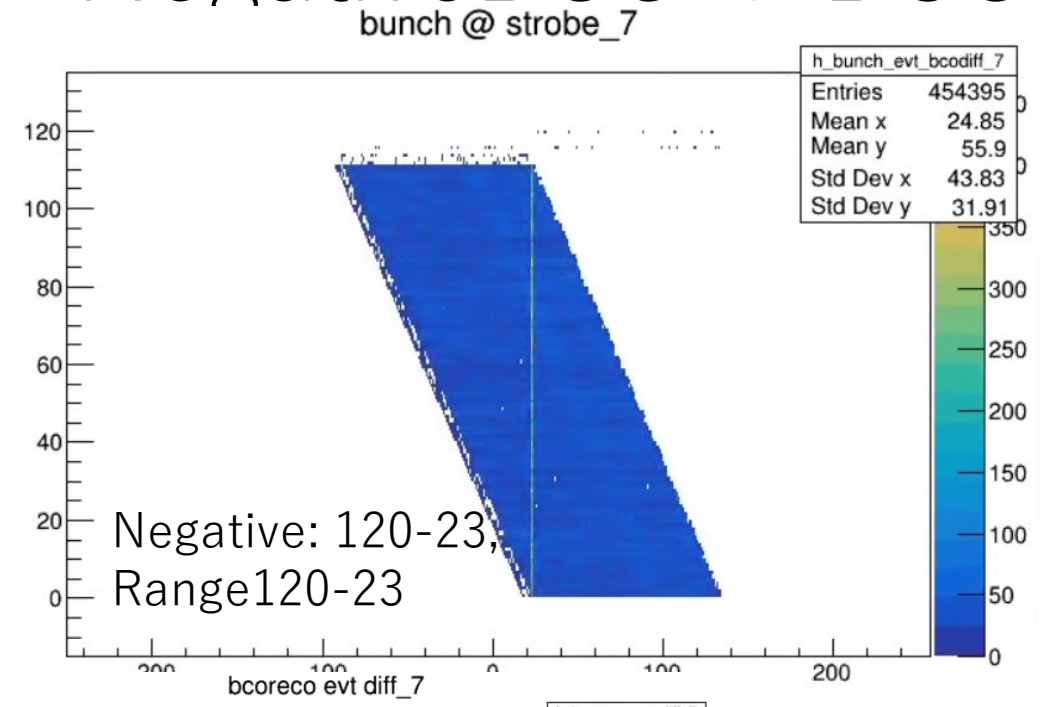


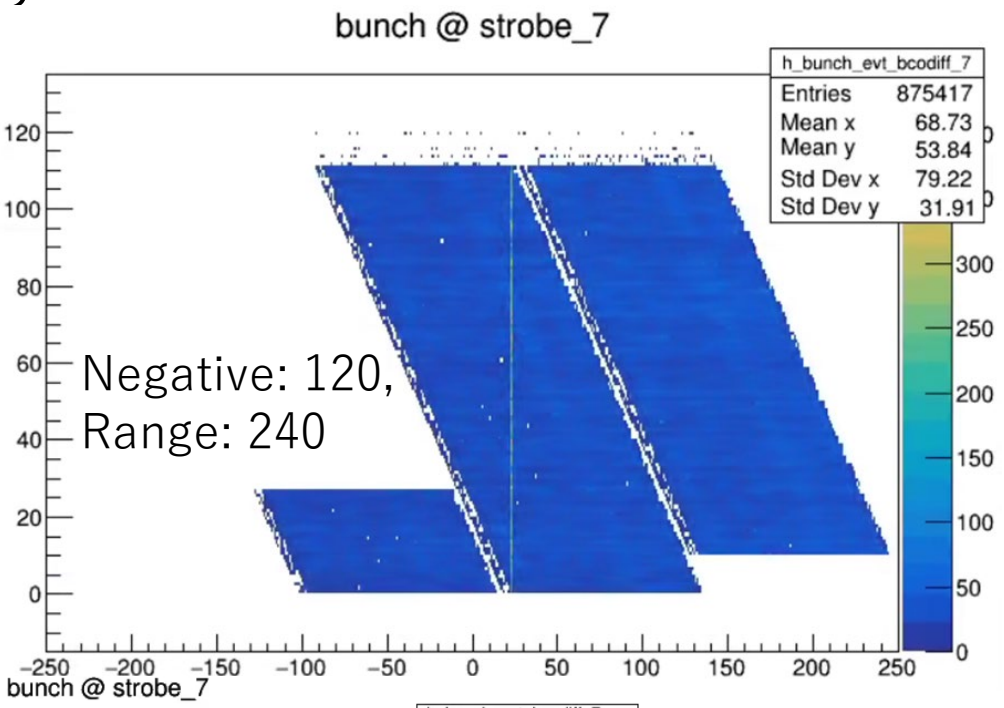



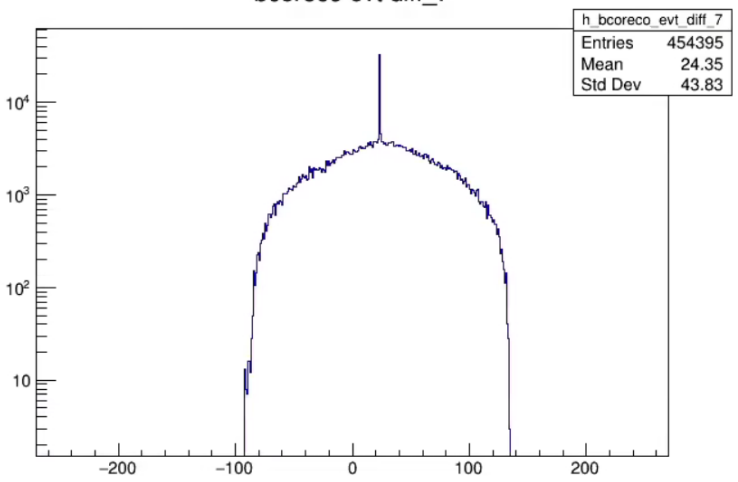



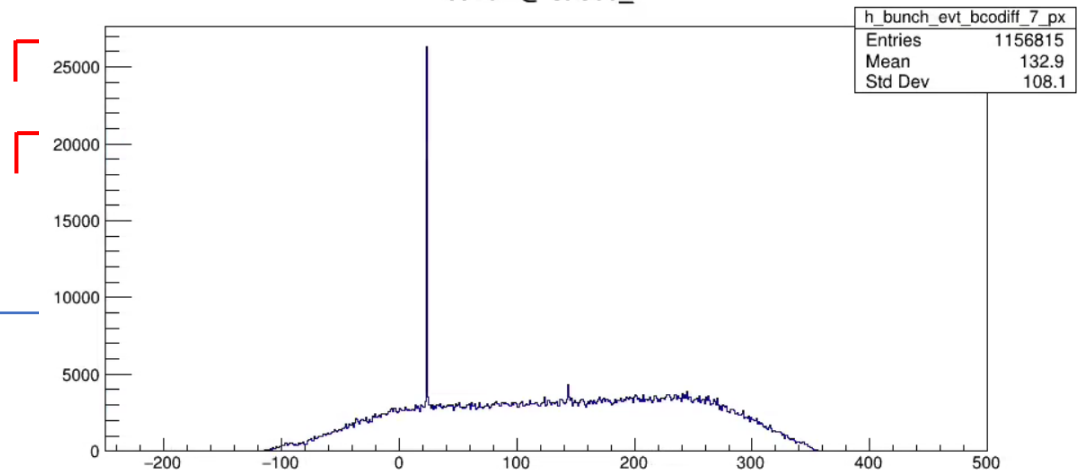

We can choose the range

# NegativeBCO + BCORange



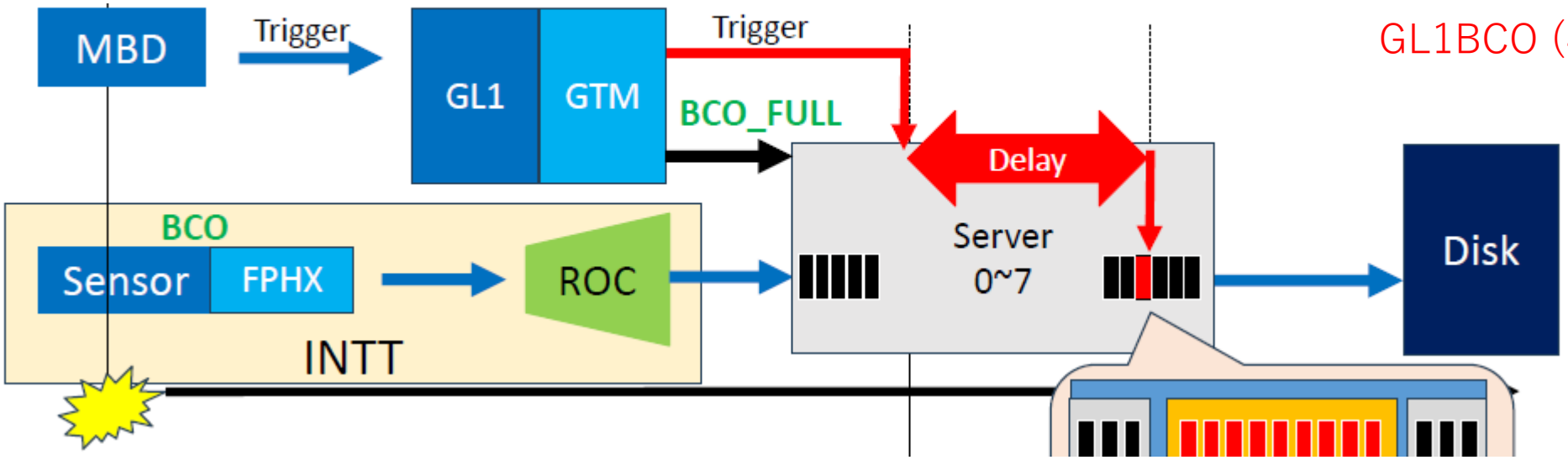




Second peak seen.
Could be mixup?

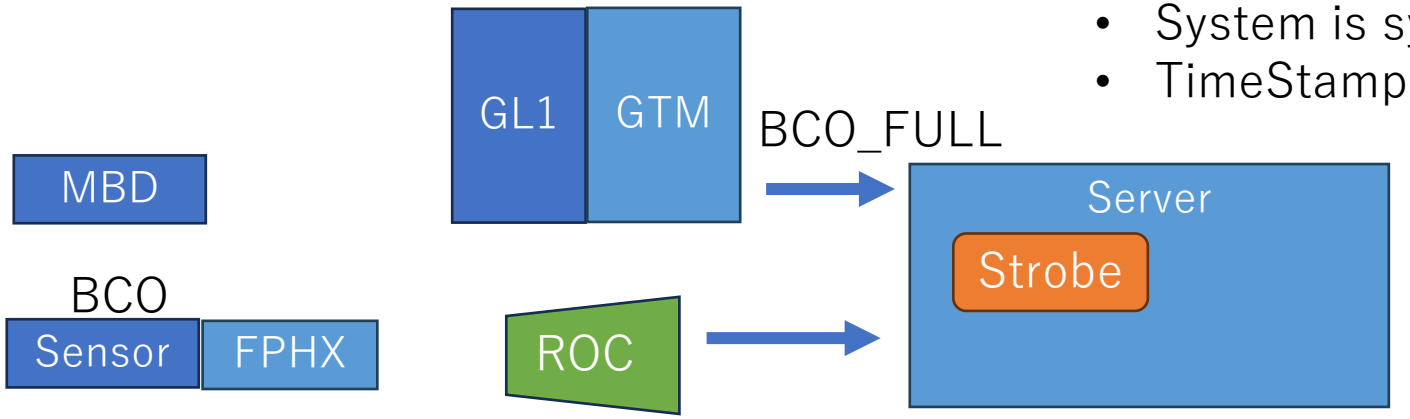
Right

# INTT readout diagram for (extended) trigger mode




GL1BCO (40bit) comes with trigger

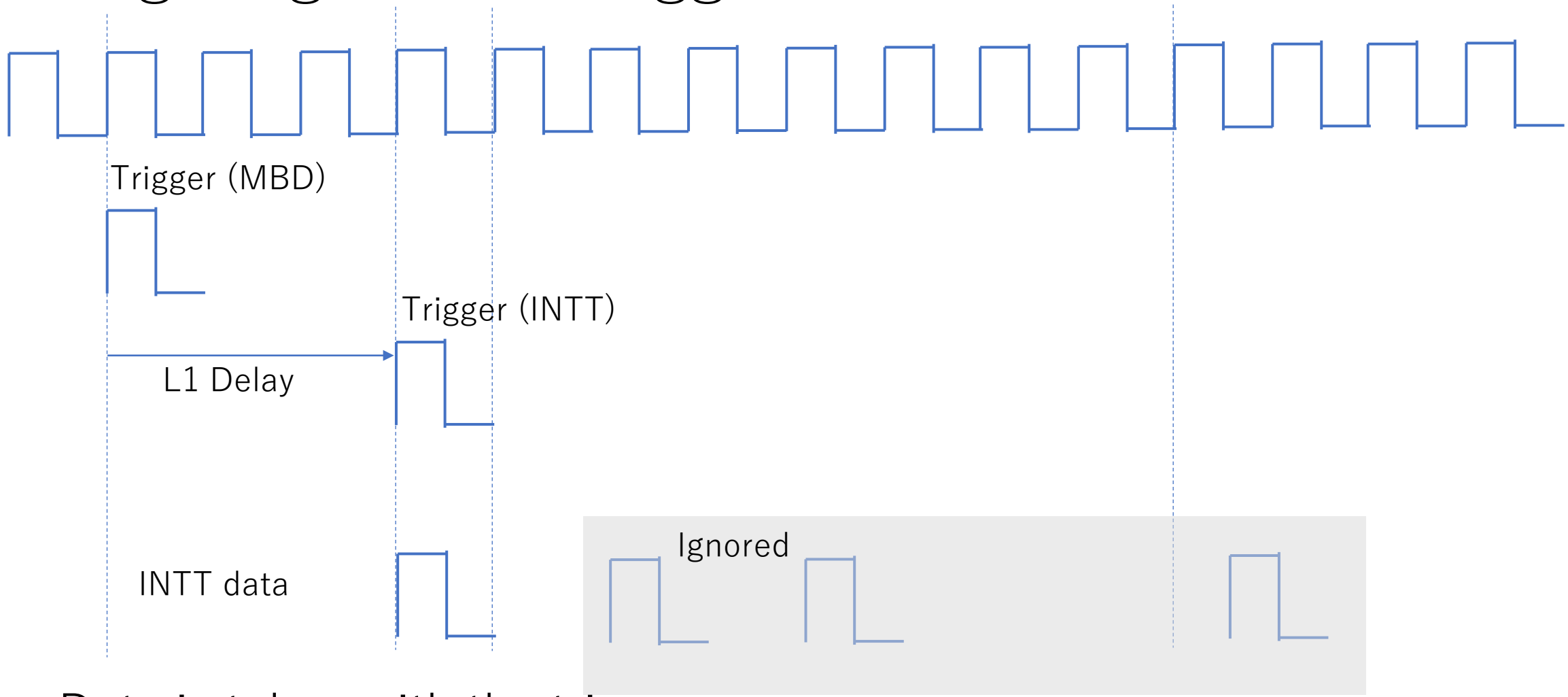
## Stream mode




Triggered GL1BCO is not available in INTT system

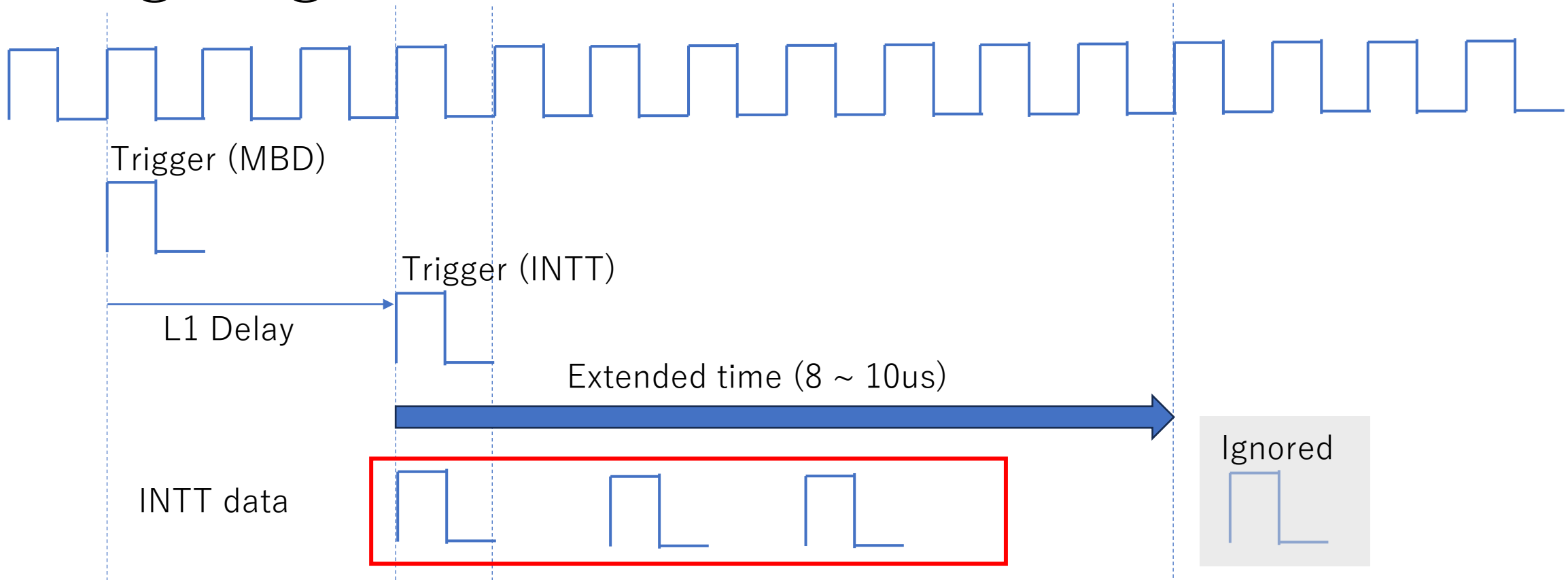
- GL1BCO is received from GTM
  - System is synchronized
  - TimeStamp (GL1BCO) is available but not sync to trigger

# Timing diagram for Trigger mode



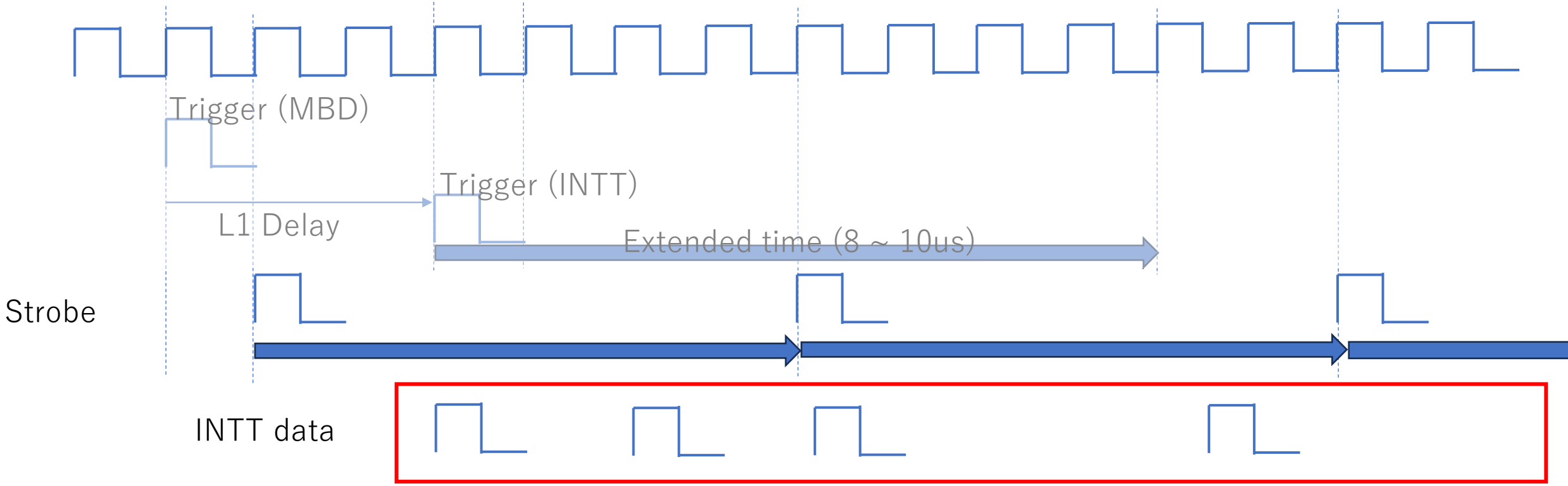

- Data is taken with the trigger
  - Any other hits are not recorded.
  - Standard way to take data. CALO system use this.

# Timing diagram for the extended mode




- Data taking window starts by the trigger and keep up to the extended time(10us)
  - All the hits are recorded within the extended period of time.  But some deadtime remain
  - INTT (and TPC) use this mode for now
  - DAQ rate is limited by 20KHz (50us)
- We have more data but still lose some data
  - 10% more MB data based on Jin's estimate

# Timing diagram for the stream mode

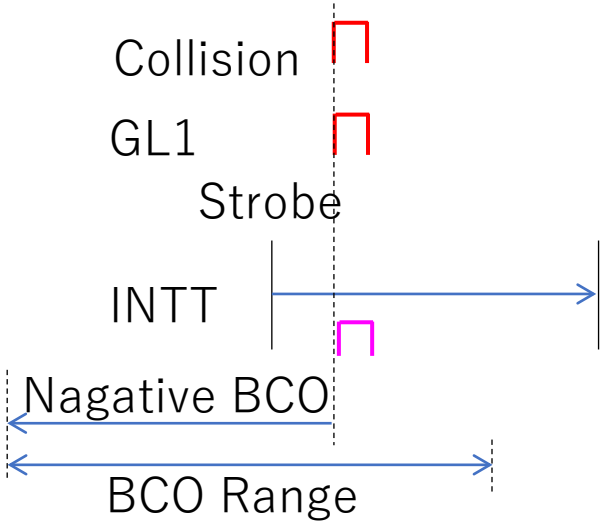


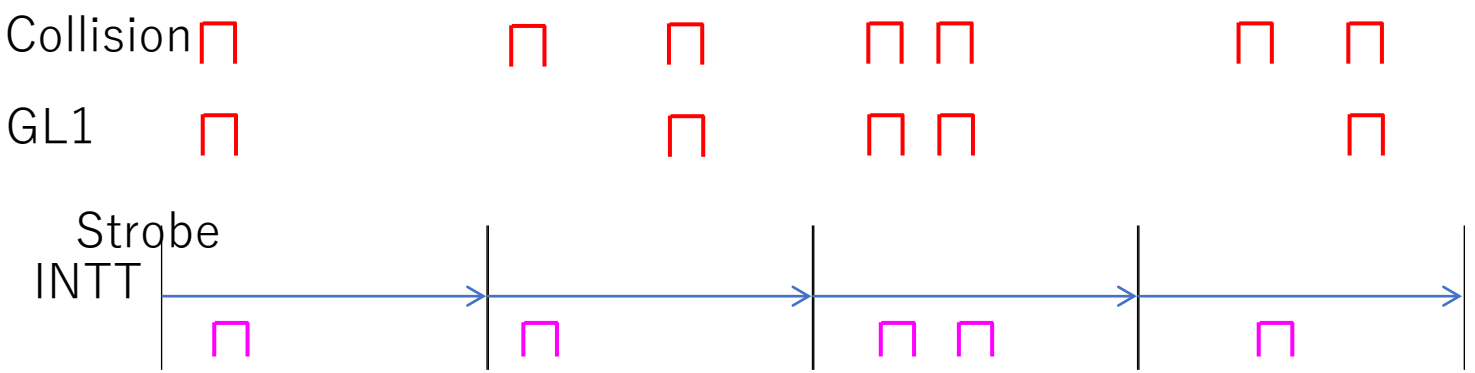

- Data is taking continuously without deadtime
- The strobe (start DAQ) is issued repeatedly (once per 120 BCO = 1 RHIC cycle) and open (120BCO periods)
  - All the hits are recorded without the dead time
  - New and ideal mode INTT (and TPC), need to test if it works

# 1 BCO DIFF QA plot as offline QA

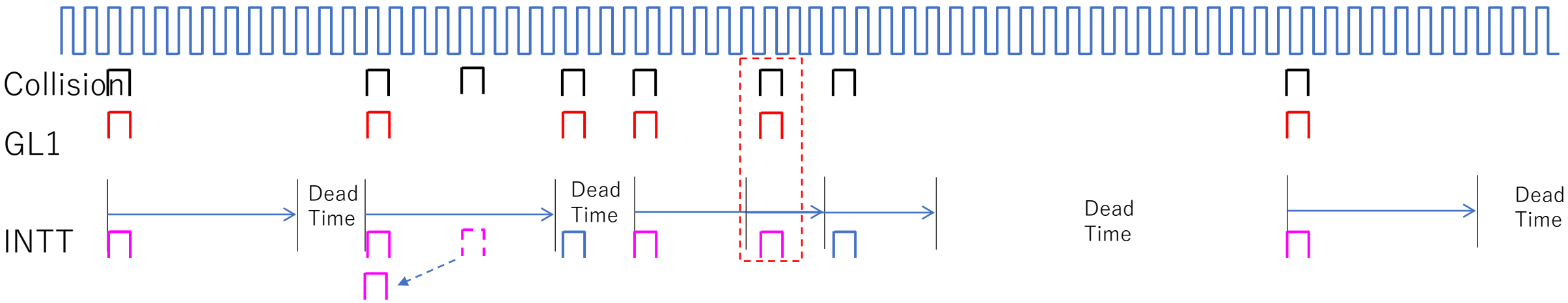
- BCO diff plots are required as offline QA
- For the streaming mode, I thought it cannot be made using F4A, but I found I can do it.

```
SingleInttPoolInput *intt_sngl= new SingleInttPoolInput("INTT_" + to_string(i));
intt_sngl->SetNegativeBco(120-23);
intt_sngl->SetBcoRange(120); // 128 + 256
intt_sngl->AddListFile(iter);
in->registerStreamingInput(intt_sngl, InputManagerType::INTT);
```



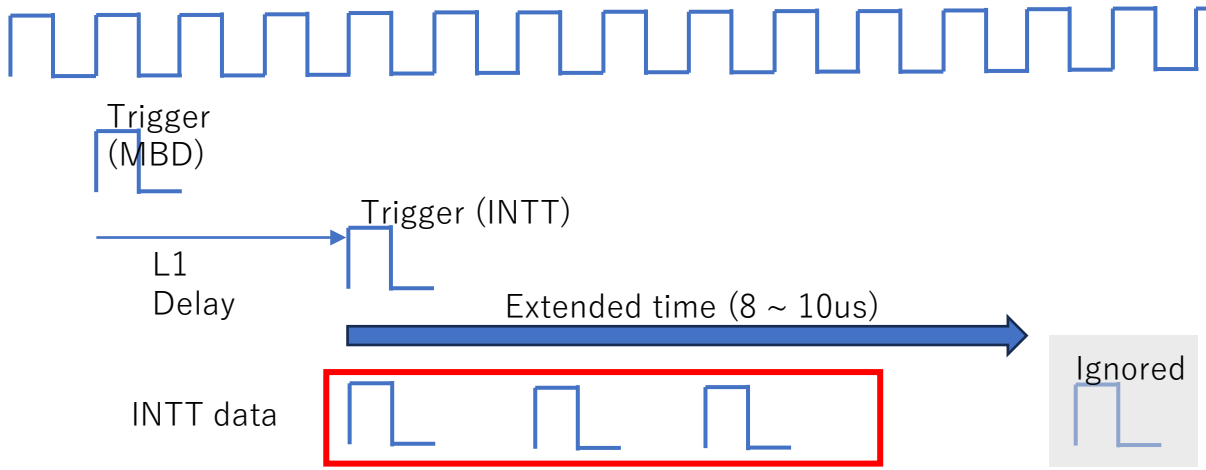



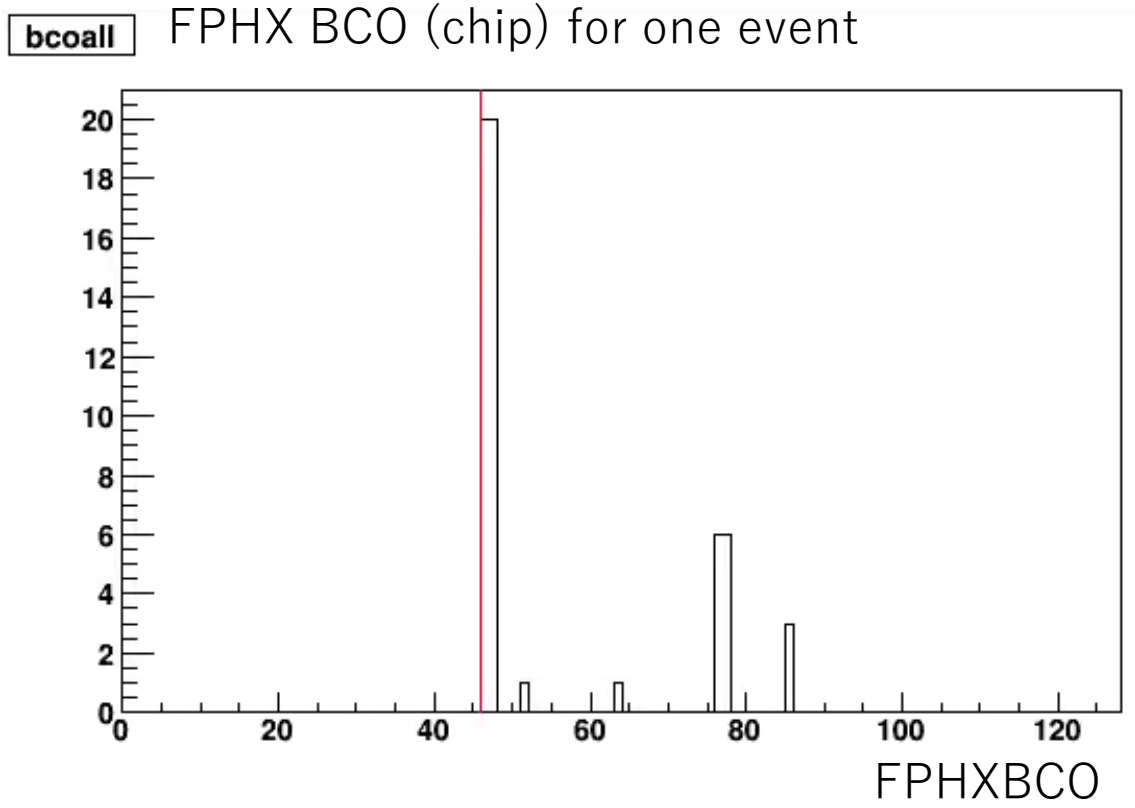
# Extended mode




- BCOFULL in data is the same with GL1BCO
- Issue/Questions
  - If second GL1 comes during the extended time, what happen?

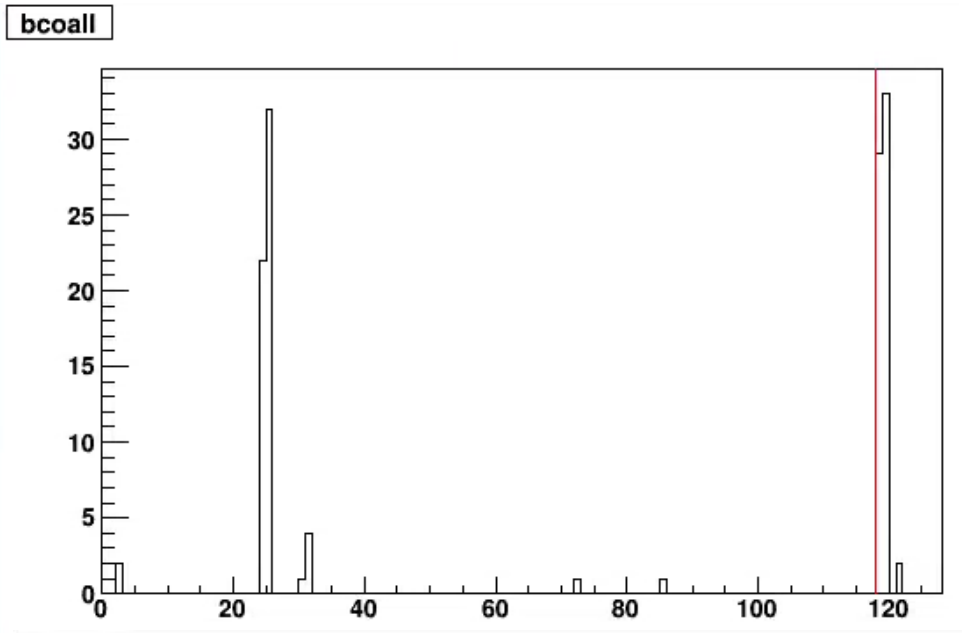
# Snapshot of the event reconstruction for the extended mode

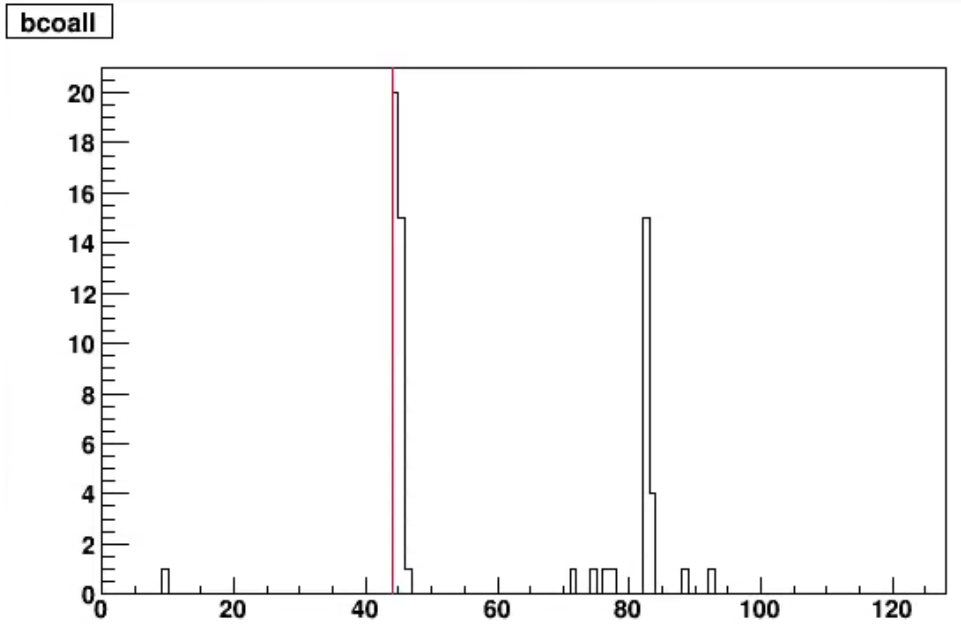


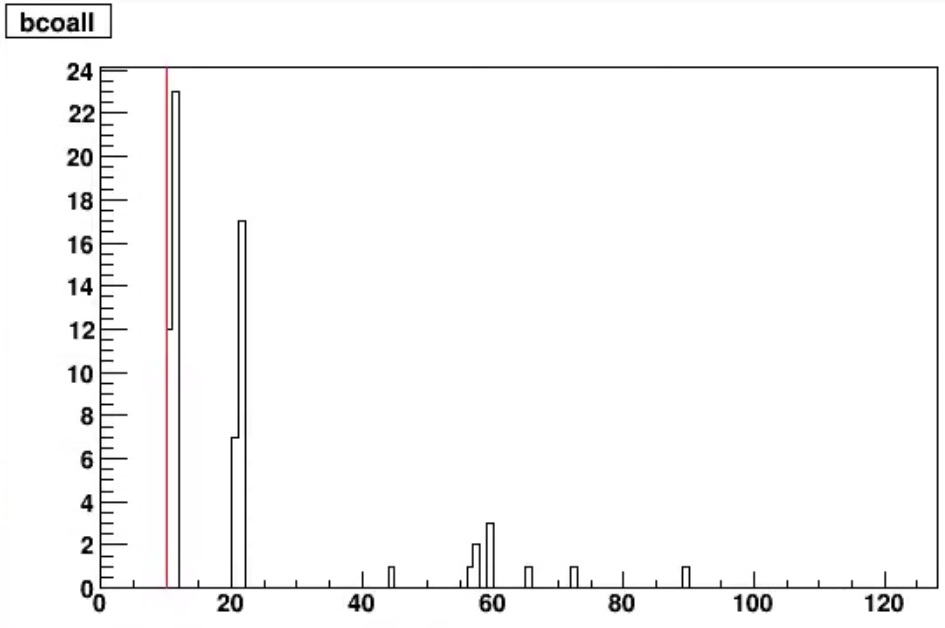



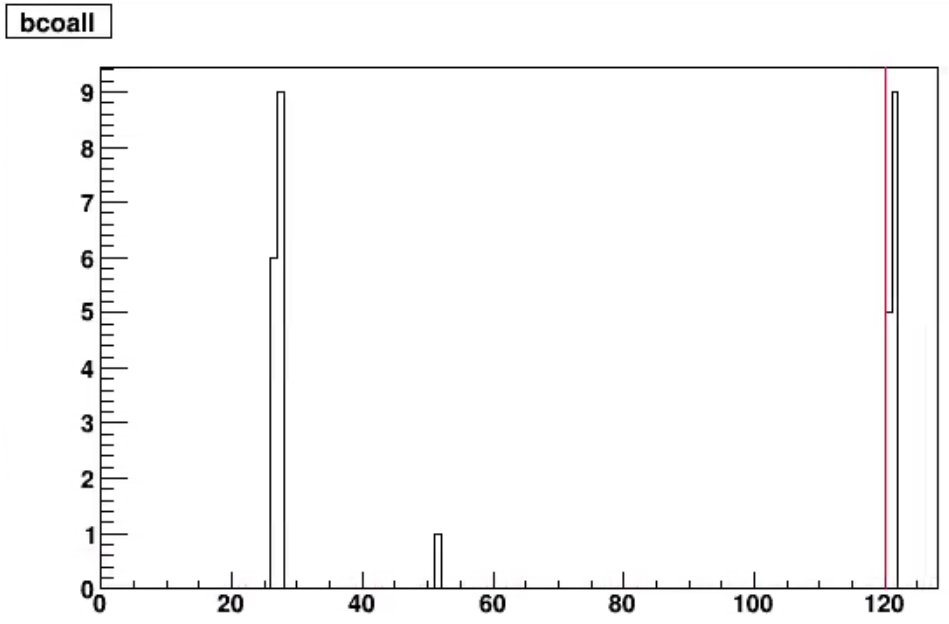

- Extended mode is analyzed (Run40742)
- FPHX BCO (7bit) for single event
  - Red : Trigger
- Shows the BCO peak is synchronized with the trigger and additional peak are there which is the additional data taken during the extended time

# More examples for the extended time



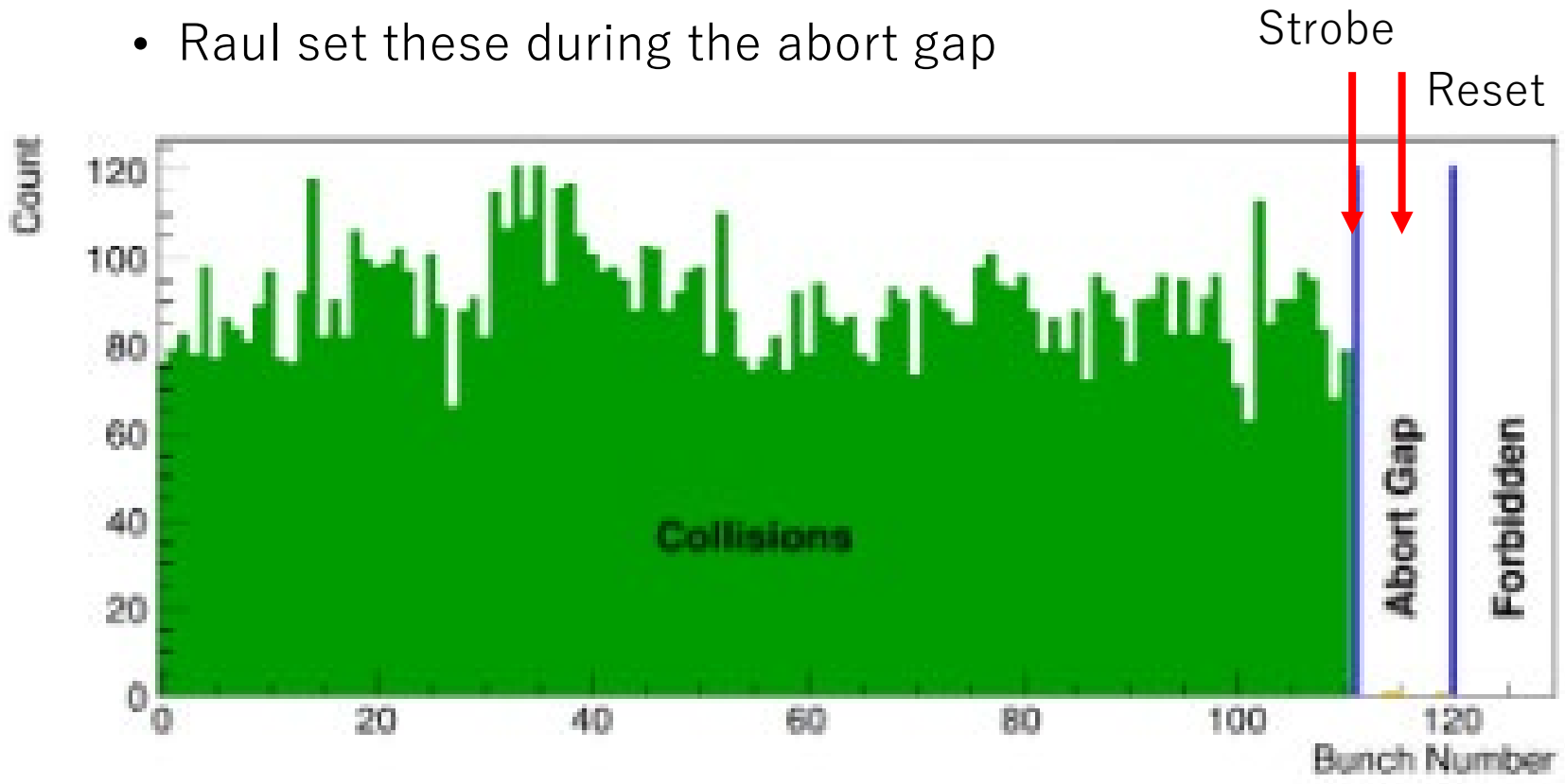



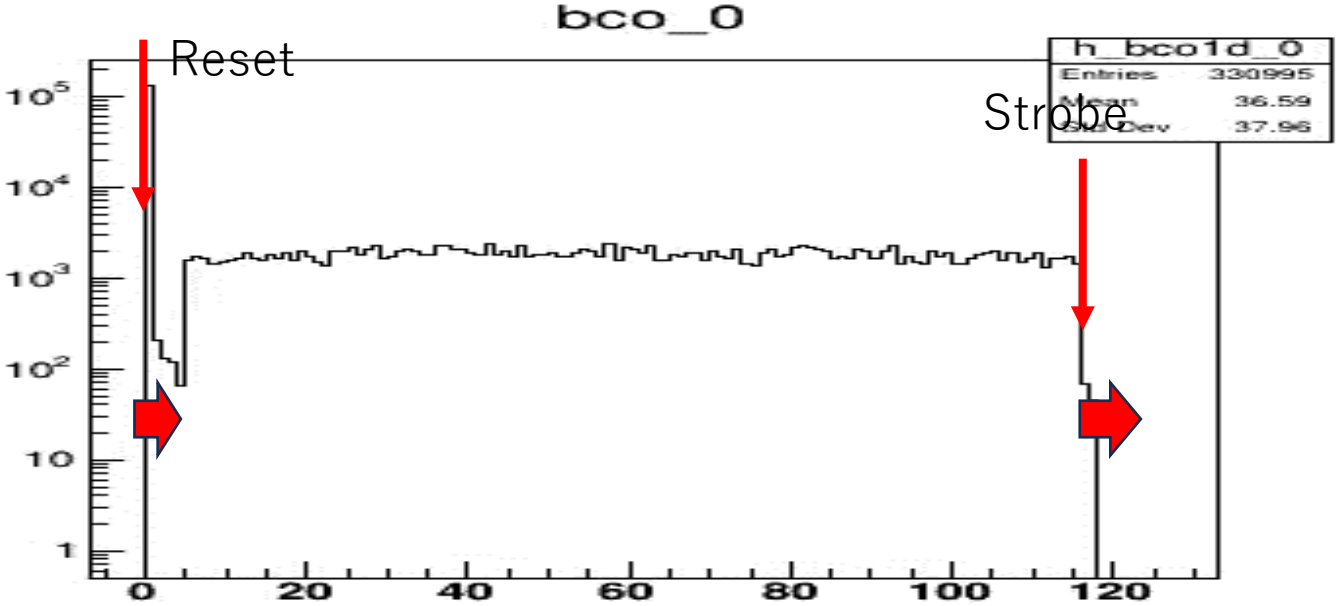



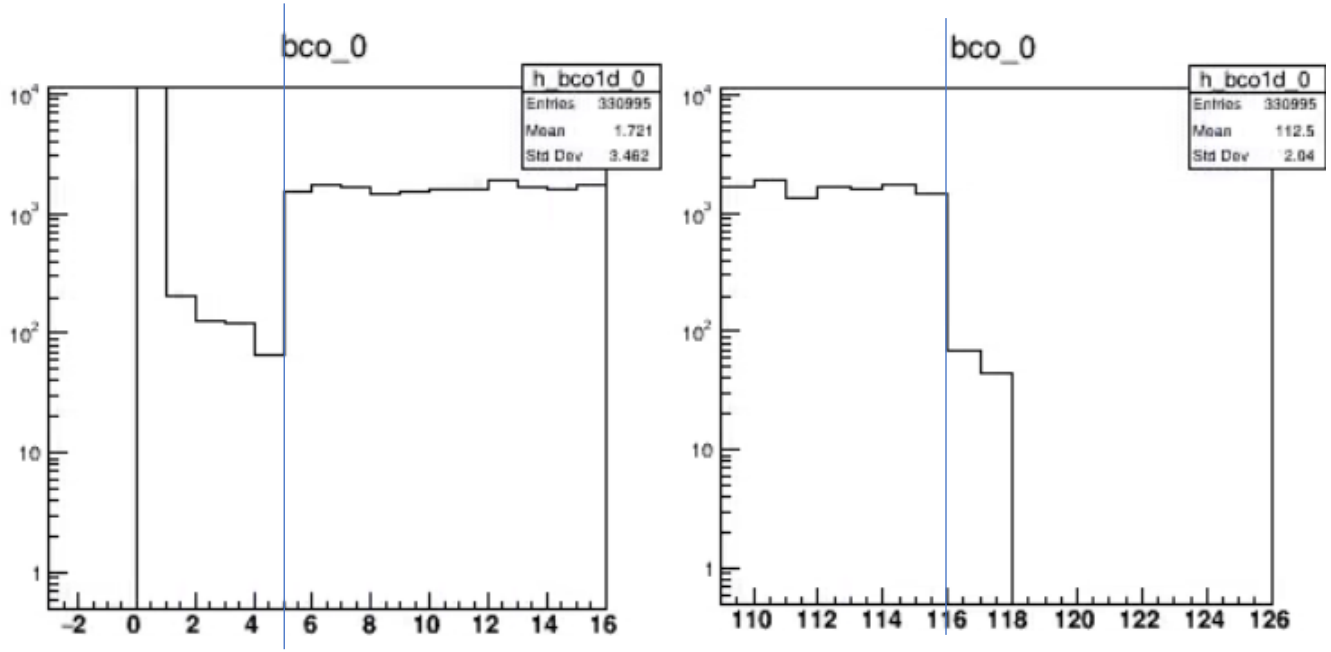
# Testing stream mode

- To make the stream mode simple, Raul proposed
  - Strobe is issued once per 120 BCO (= one RHIC cycle)
  - FPHX BCO is reset by 120 BCO (=one RHIC cycle)
    - These happens at the fixed bunch ID
    - Raul set these during the abort gap



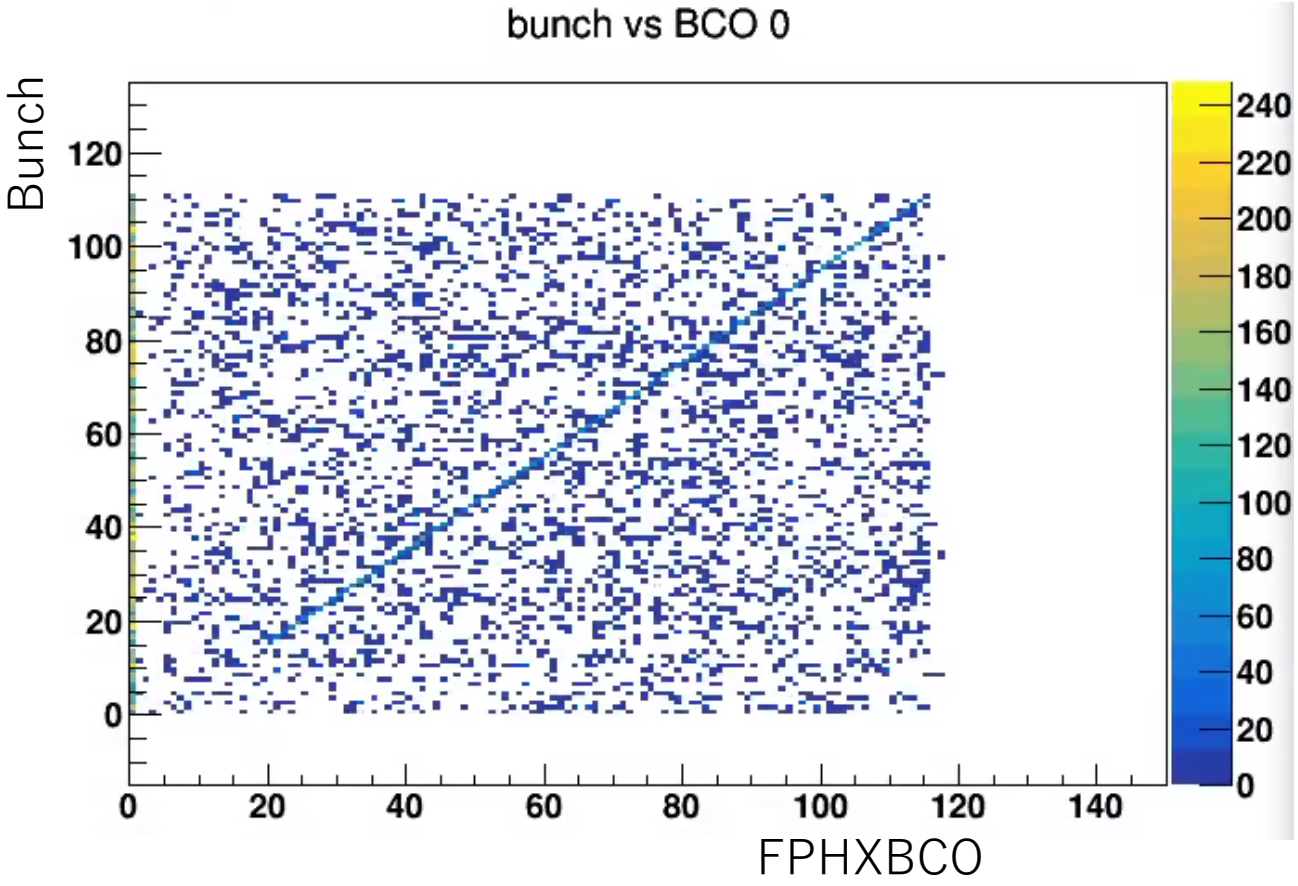
# FPHX BCO distribution from the stream mode




- Abort GAP clearly seen
- Strobe and Reset works as expected

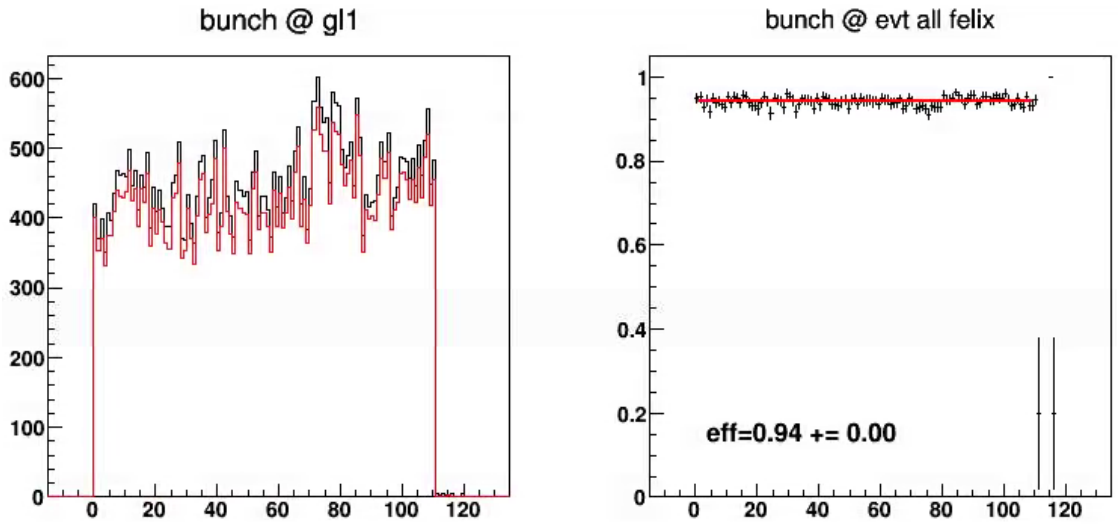
# Bunch vs FPHX BCO



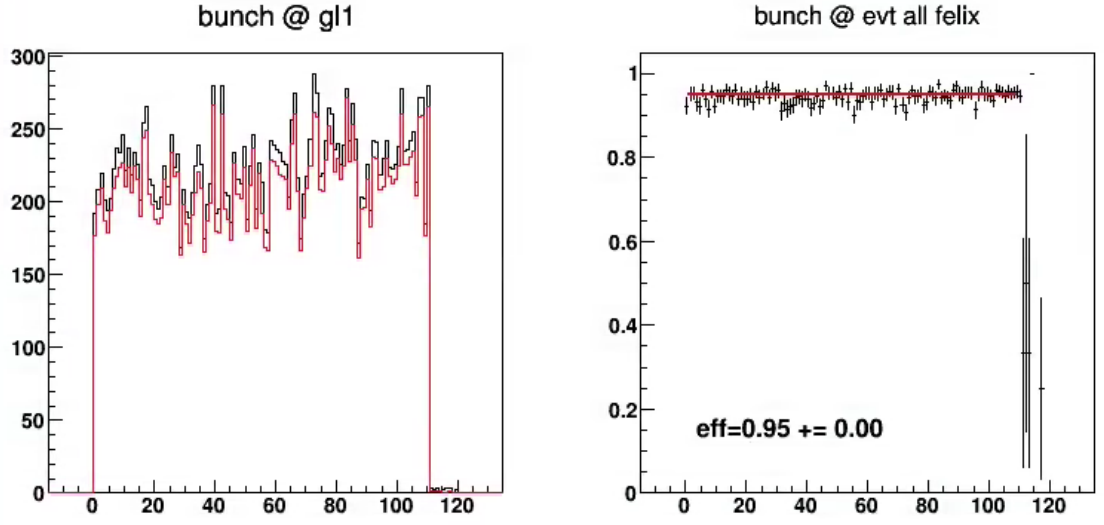

- Bunch and BCO is synchronized because the BCO is reset at the BUNCH

# GL1 finding efficiency

Streaming (run47977)




Triggered (run47982)




- Finding efficiency is similar for both the stream and triggered
  - It looks that the stream mode works properly
  - Previous study : 97%. Why?

- Offset between GL1 and recoBCO is changed. Why?
  - previous offset was 19, now it is 23
  - If we find the reason, it can use it. Otherwise this should be a calibration parameter

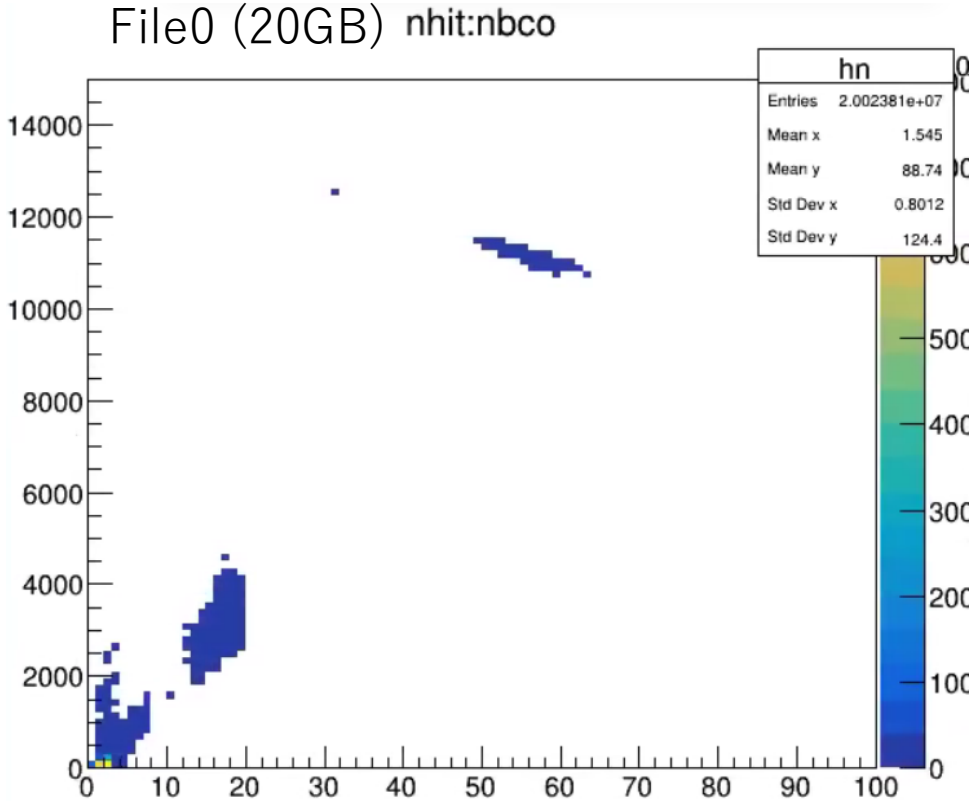
# Run 49660 streaming INTT0 evt file check

| | | | | | | |
|---|---|---|---|---|---|---|
| 49960 | physics | 2024-08-02 03:30:00 | 2024-08-02 03:42:50 | 6028353 | Trigger Info | Zero Suppression |

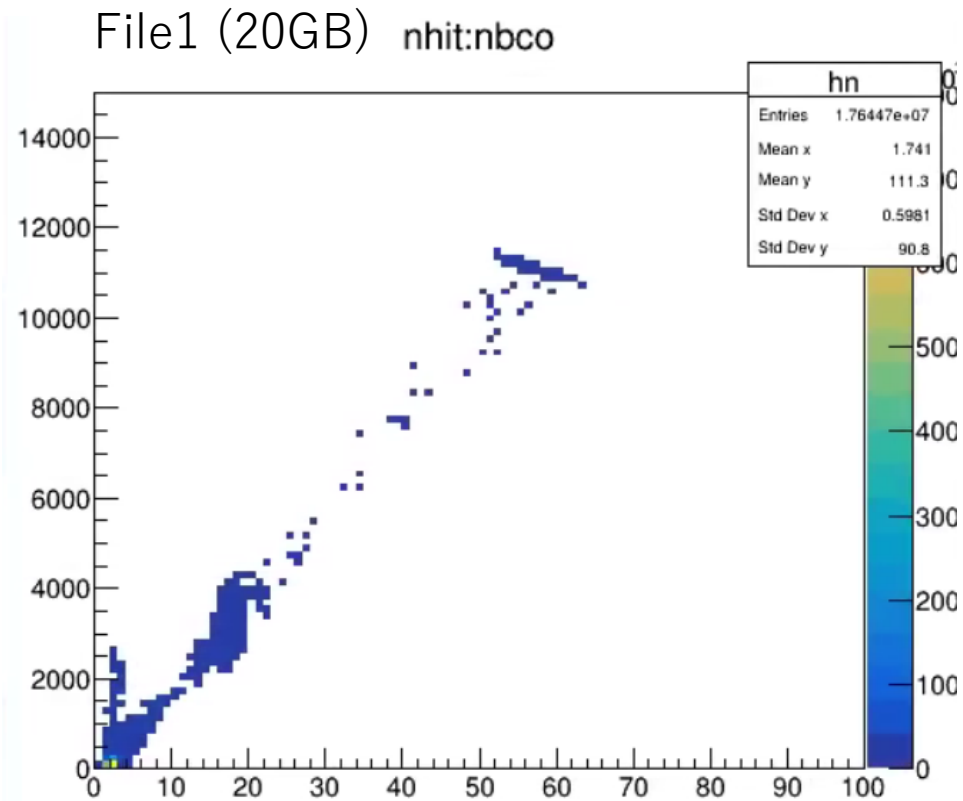
- File0 last BCO
  - 29 0xfe0069db11
  - 30 0xfe0069db89
  - この後、BCOがジャンプし、次に出てくるのが以下。
    - この期間のBCOFULLがない。
  - 0 0xfe055a6159
- File1 first BC0
  - 0 0xfe7d0e4891
  - 1 0xfe7d0e4909

- Duration time: 12m50s
- File0 のLast BCOがあるのは77k evtのあたり
- File0の最後のBCOとFile1の最初のBCOの時間差は220s

File0 (20GB) nhit:nbco

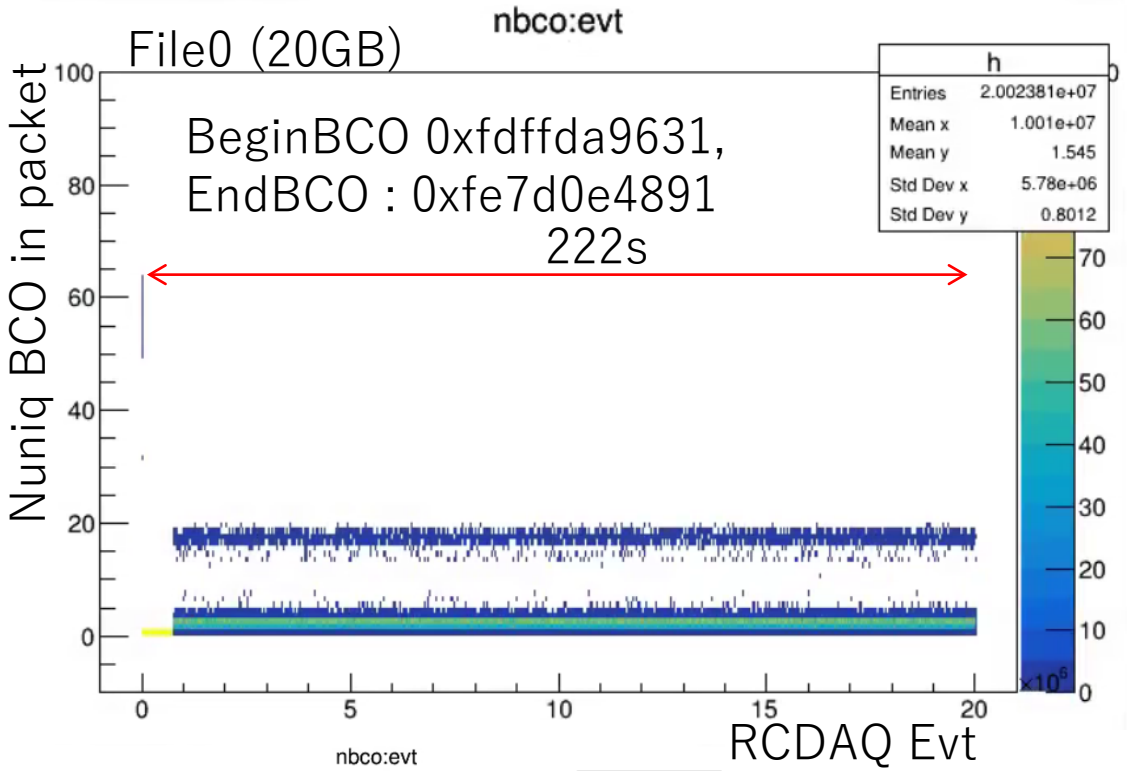


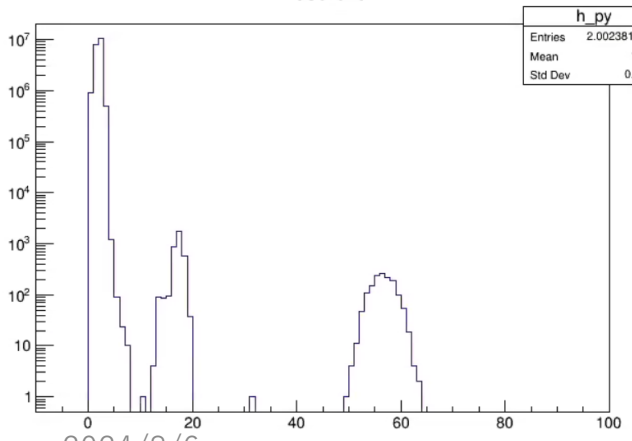

File1 (20GB) nhit:nbco



# Analysis by ddump

File0 (20GB)




Begin 0xfe7d0e4891, end: 0xfefae9b561

File1 (20GB)