Enhancement of the Belle II High Level Trigger using GPUs

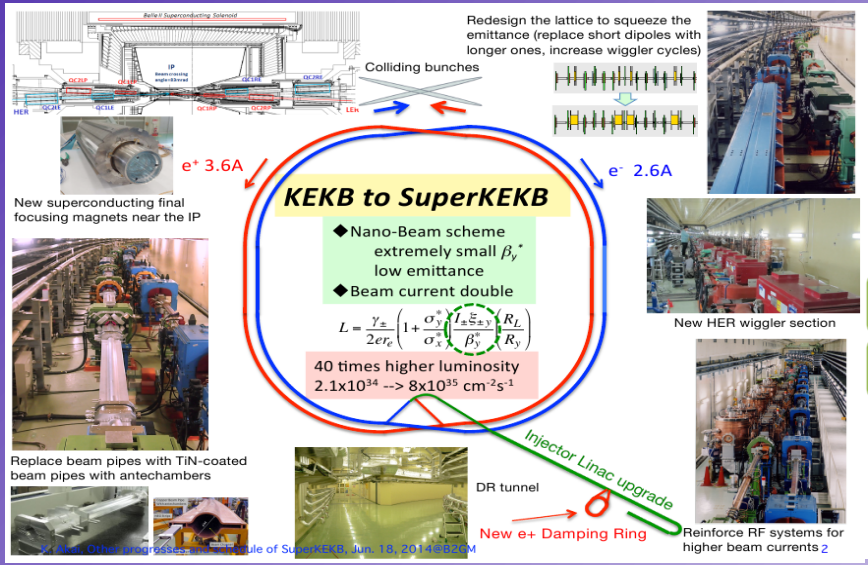
Ryosuke Itoh, KEK
on behalf of
Belle II DAQ group

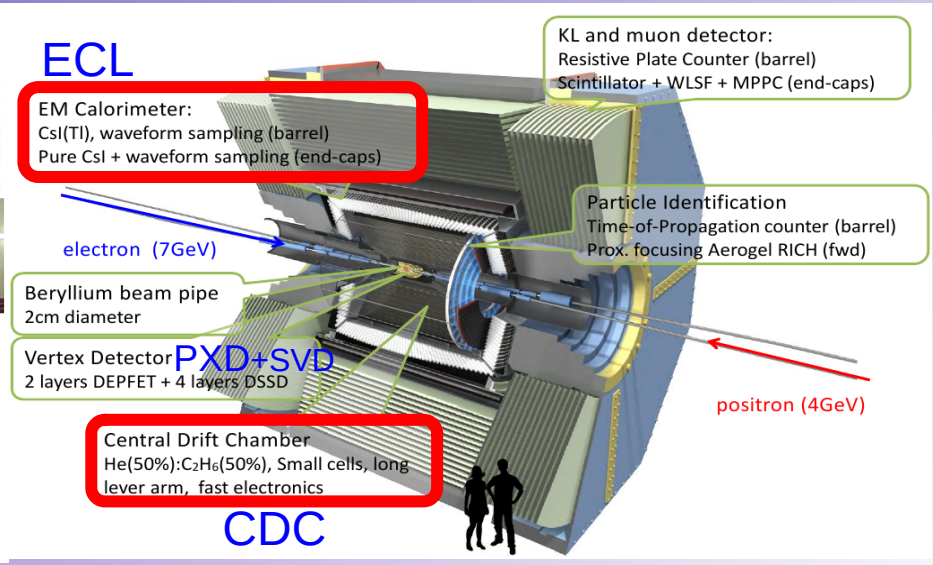
CEPC Workshop 2026, Shanghai, China,
Oct.26, 2026

1. The Belle II experiment

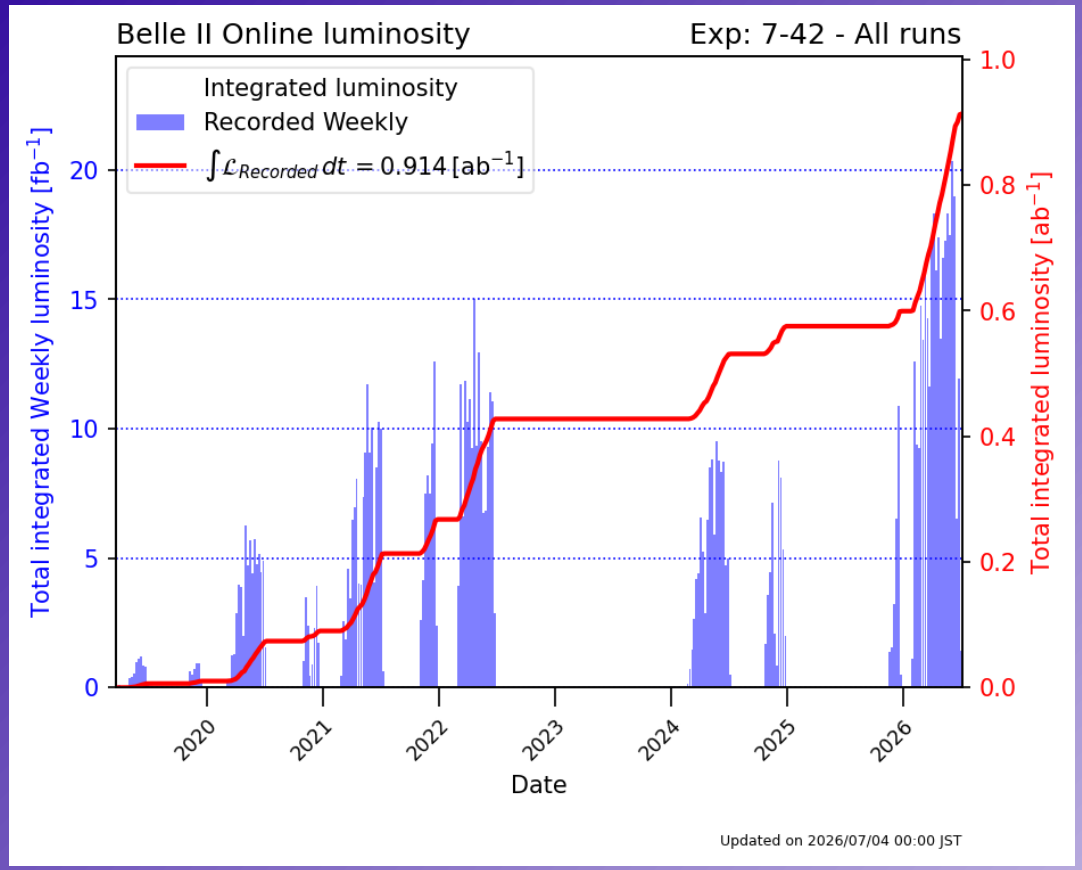
- The Belle II experiment is a new generation B-factory experiment at KEK in Japan to search for New Physics(NP).

- A huge number of B mesons are produced in e^+e^- collisions in the SuperKEKB accelerator and NP is searched for in B meson decays by the Belle II detector complex.



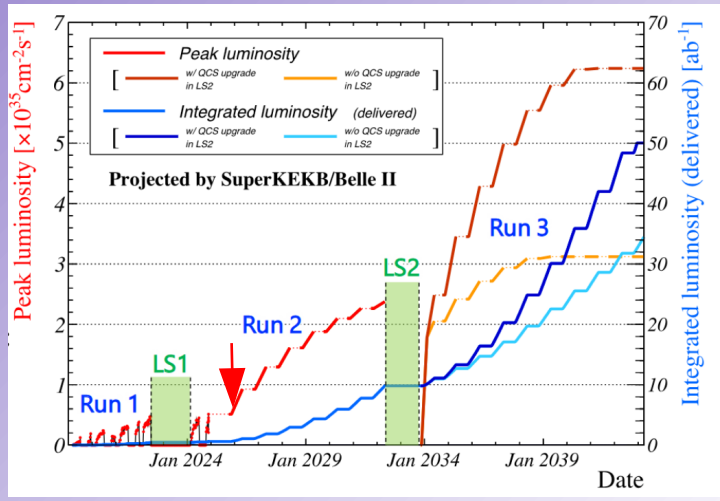


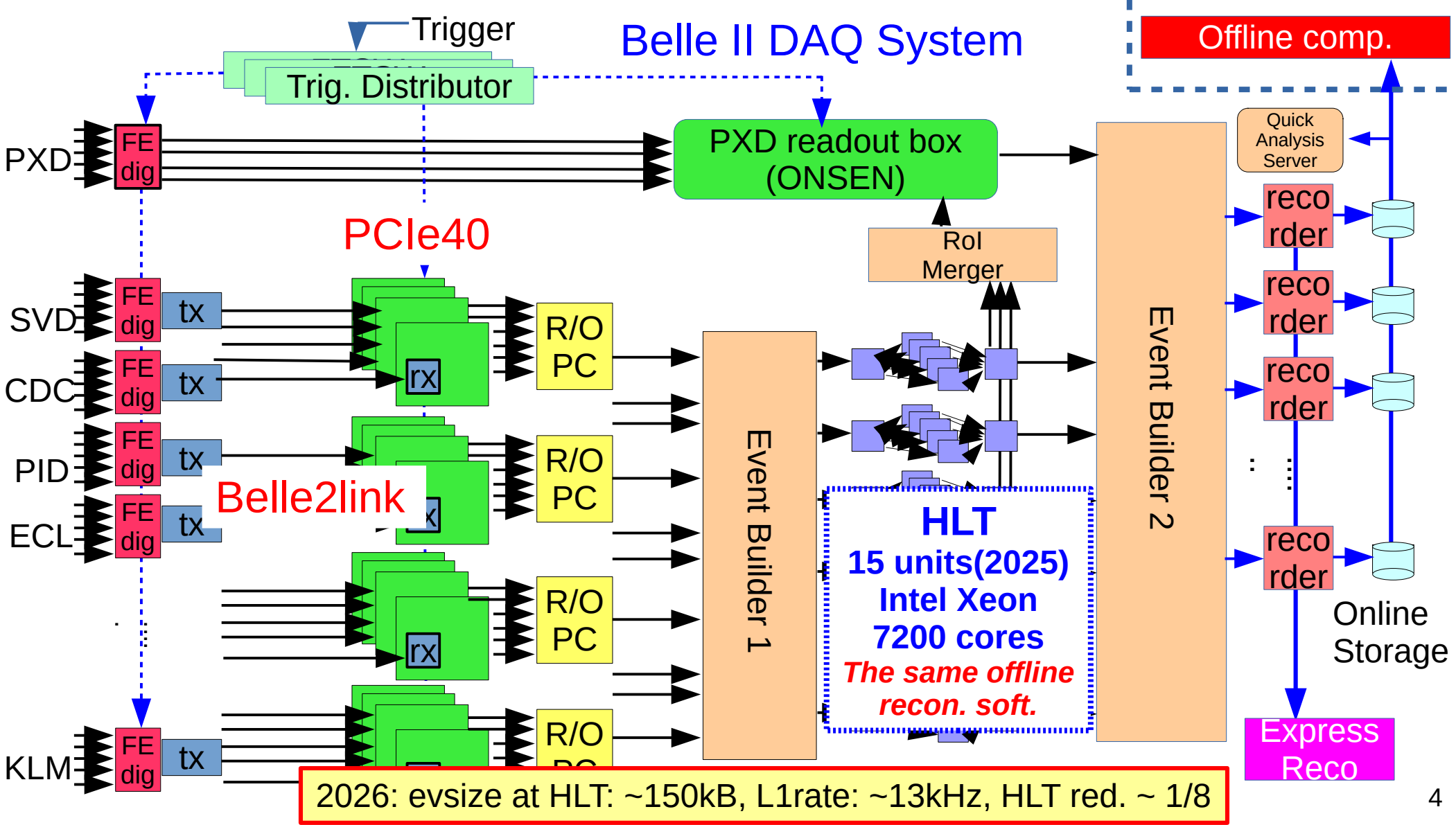
Belle II operation history

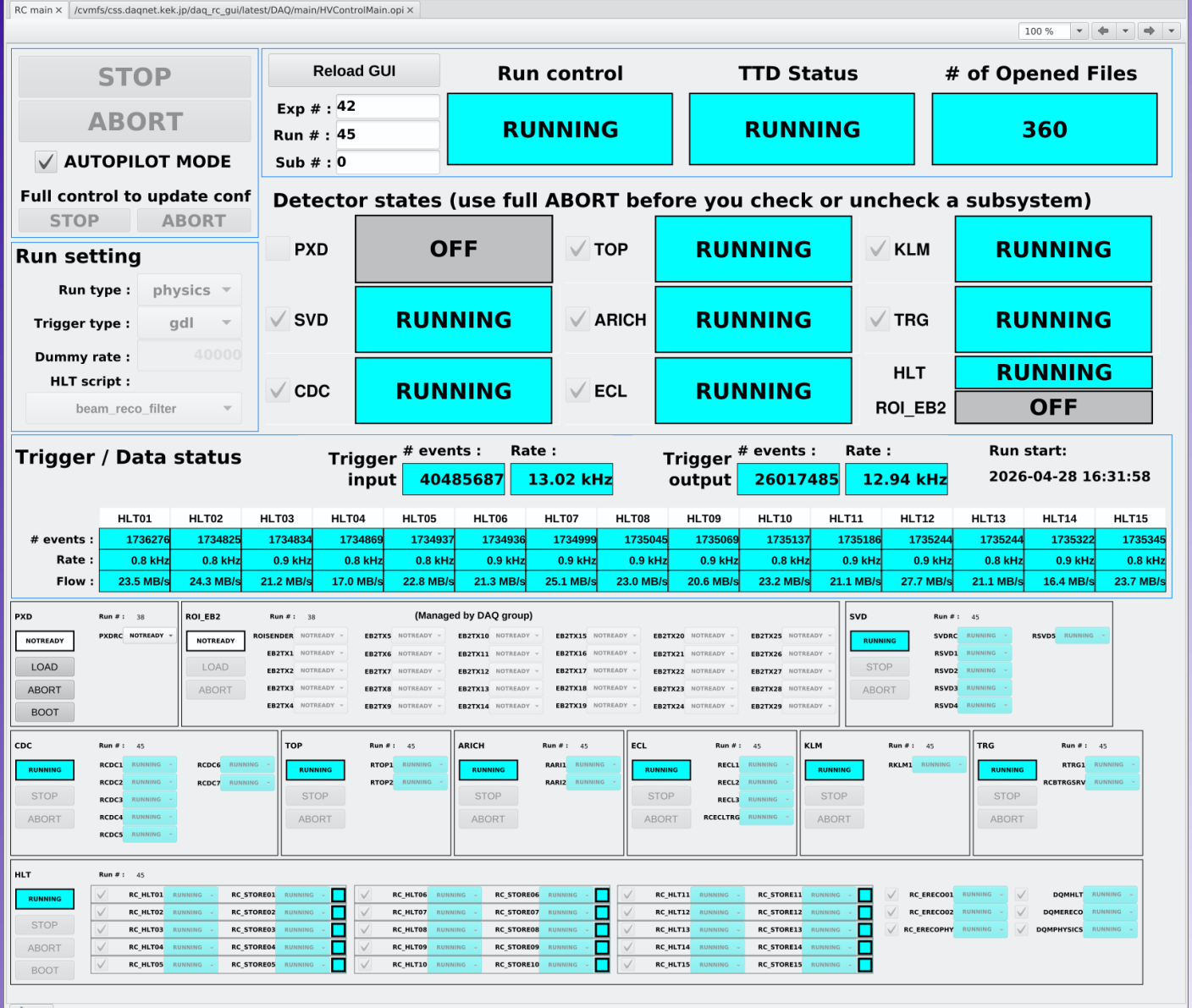


* Run 2 (Jan, 2024 -)
  L_{peak} = 5.32 x 10^{34} $cm^{-2}sec^{-1}$
  Integrated L ~ 914 fb^{-1}
  (Target luminosity 6 x $10^{35}cm^{-2}sec^{-1}$)

* Many difficulties in accelerator operation.
  -> Pixel Detector (PXD) is temporarily turned off.




Belle II DAQ System
Offline comp.
Trigger
Trig. Distributor
PXD
FE dig
PXD readout box (ONSEN)
PCIe40
RoI Merger
Quick Analysis Server
SVD
CDC
PID
ECL
KLM
tx
rx
Belle2link
R/O PC
Event Builder 1
Event Builder 2
HLT
15 units(2025)
Intel Xeon
7200 cores
The same offline recon. soft.
reco rder
Online Storage
Express Reco
2026: evsize at HLT: ~150kB, L1rate: ~13kHz, HLT red. ~ 1/8



PXD is turned off

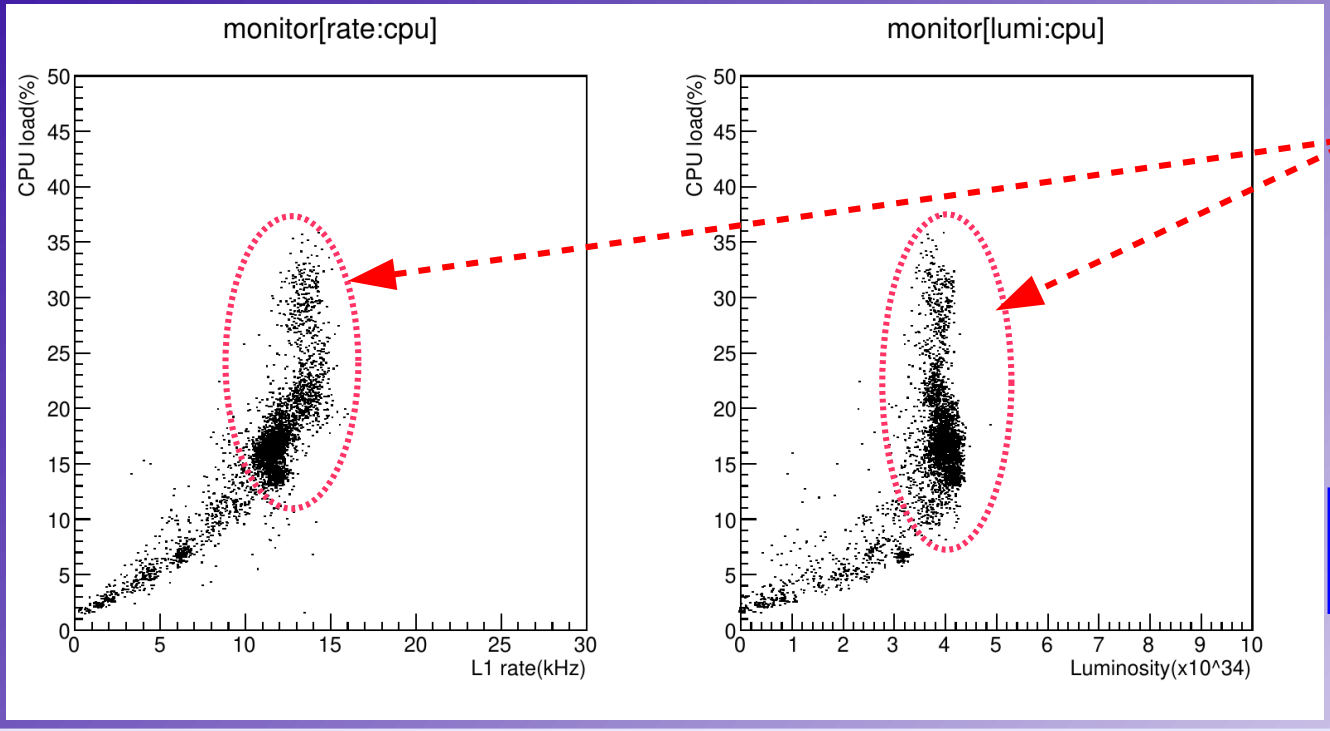
5

2. Motivation

- Current SuperKEKB status
  * Although the luminosity is still low, **the background is unexpectedly higher**.
  * This situation may persist as the luminosity continues to rise.

- The HLT CPU load is higher even though the physics event rate is far below the design value.
  * Current HLT (with 15 units) already has a sufficient processing power to manage “physics rate” at the final design luminosity.
  * The main issue is
    the higher CPU consumption caused by the “unexpectedly high background”.

Current HLT performance



Higher CPU consumption than expected

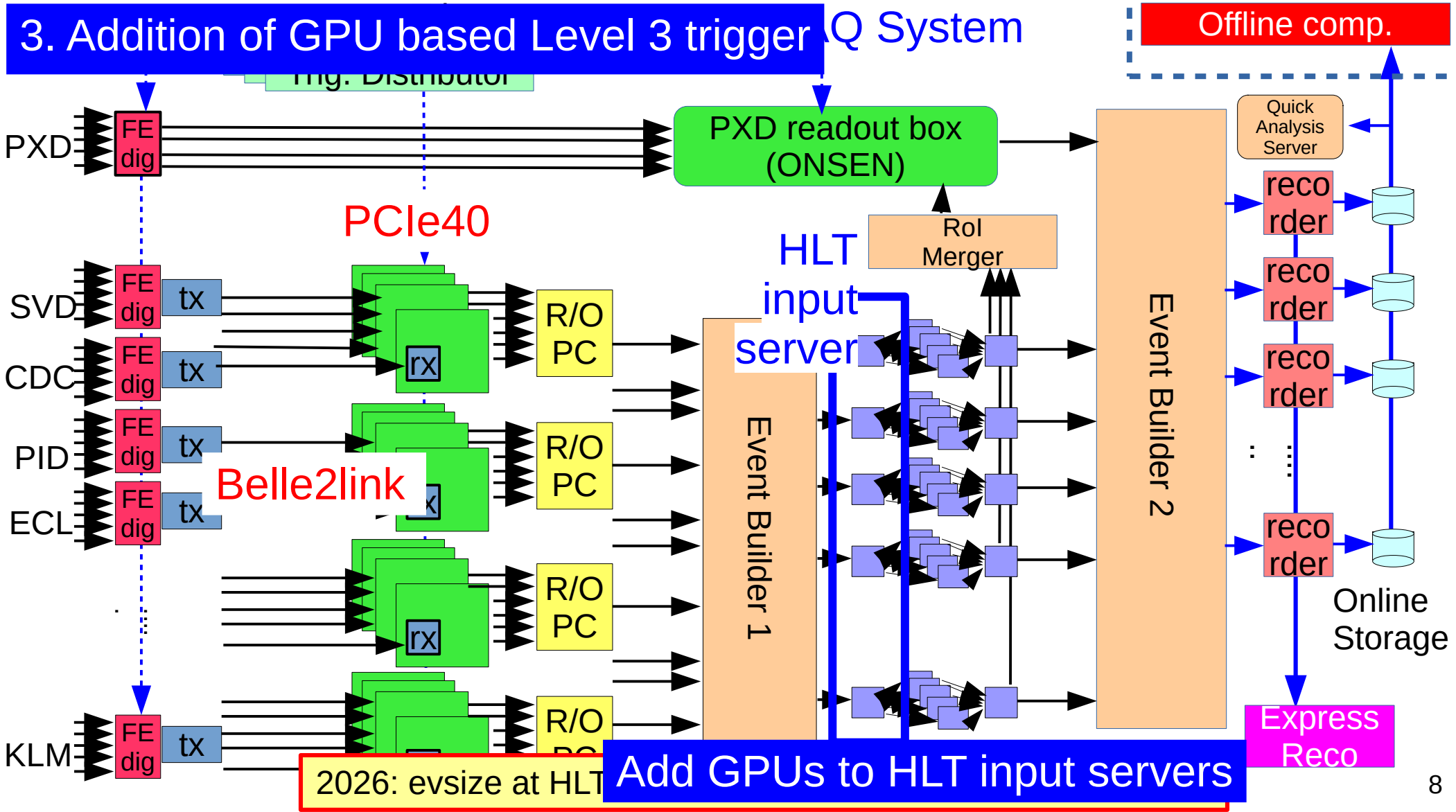
Data taken in early 2026 runs

- Heavy noise hits caused by the background are contaminated in the taken events.

  ==> Source of the unexpected higher CPU consumption.

Efficient supression of such noise hits/events is the key to mitigate the problem.

3. Addition of GPU based Level 3 trigger



Design Policy

- Keep existing HLT as is.
  * No change in the HLT hardware/software.
    Current HLT software processing is not touched at all.

- Add GPU processing before the existing HLT framework as a part of the HLT system.
  * **“Level 3 trigger(pre-filter)” software is implemented in GPU.**
  * Filter out background events before the actual HLT processing.

- Build GPU software development environment fully compatible with the Belle II software library coded mostly in C++.
    “Seamless software environment between CPU and GPU”

GPU-L3 project

4. Seamless Software Development Environment

- The software on GPU is desired to be compatible with the existing Belle2 software coded mostly in C++.

- GPU is supposed to be programmed in CUDA.

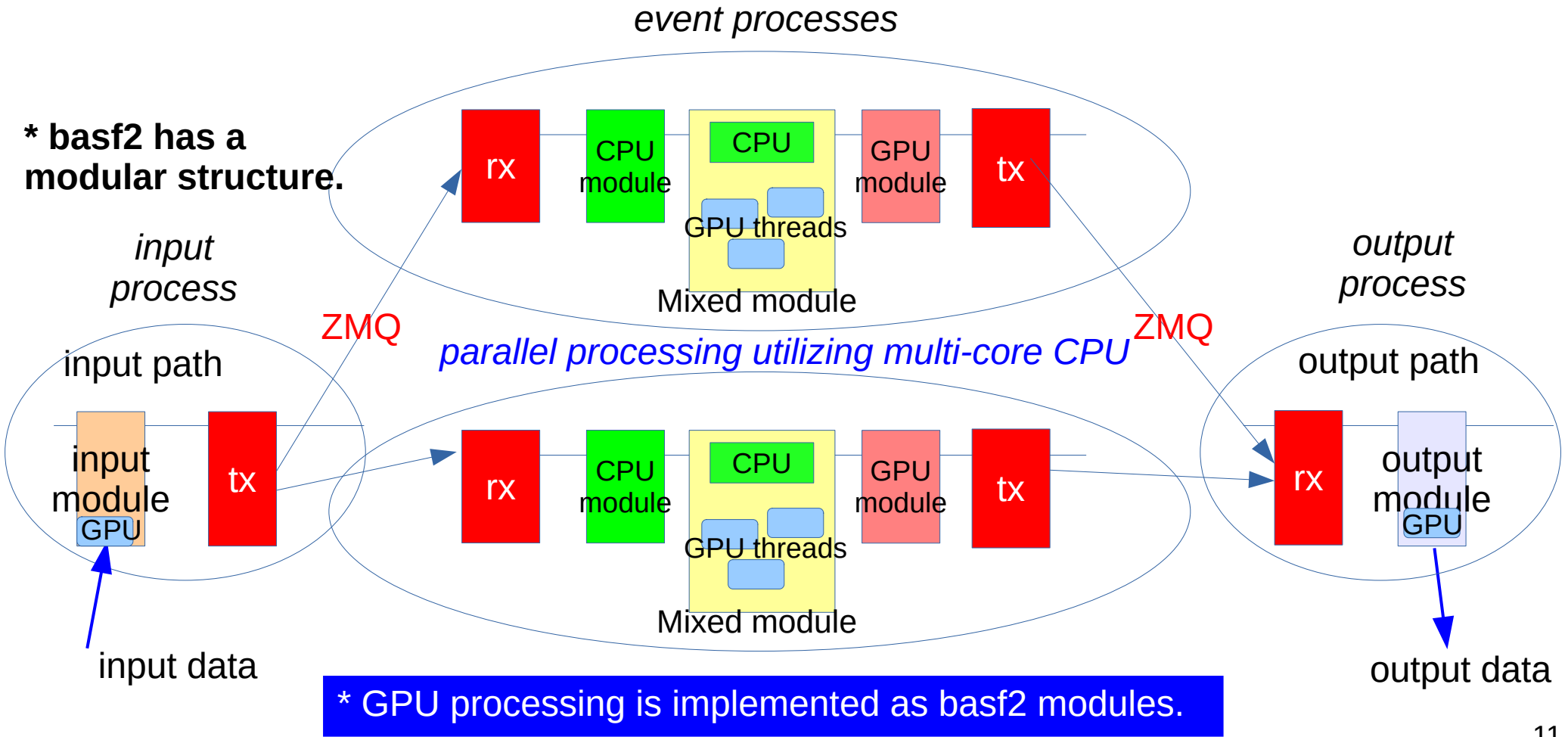
- The Belle2 software is developed using “SCons” and runs on a framework called “basf2” as a set of dynamically-linked modules.

- CUDA compiler (nvcc) was added to SCons build system.

- Development environment was built so that mixed C++ and CUDA codes can co-exist in Belle2 library and dynamically loaded in basf2.

=> Seamless execution of CUDA-coded GPU modules and C++-coded CPU modules on the same basf2 became possible.

basf2: Belle 2 event processing framework commonly used in online and offline with the parallel processing capability on multicore CPU.



* GPU processing is implemented as basf2 modules.

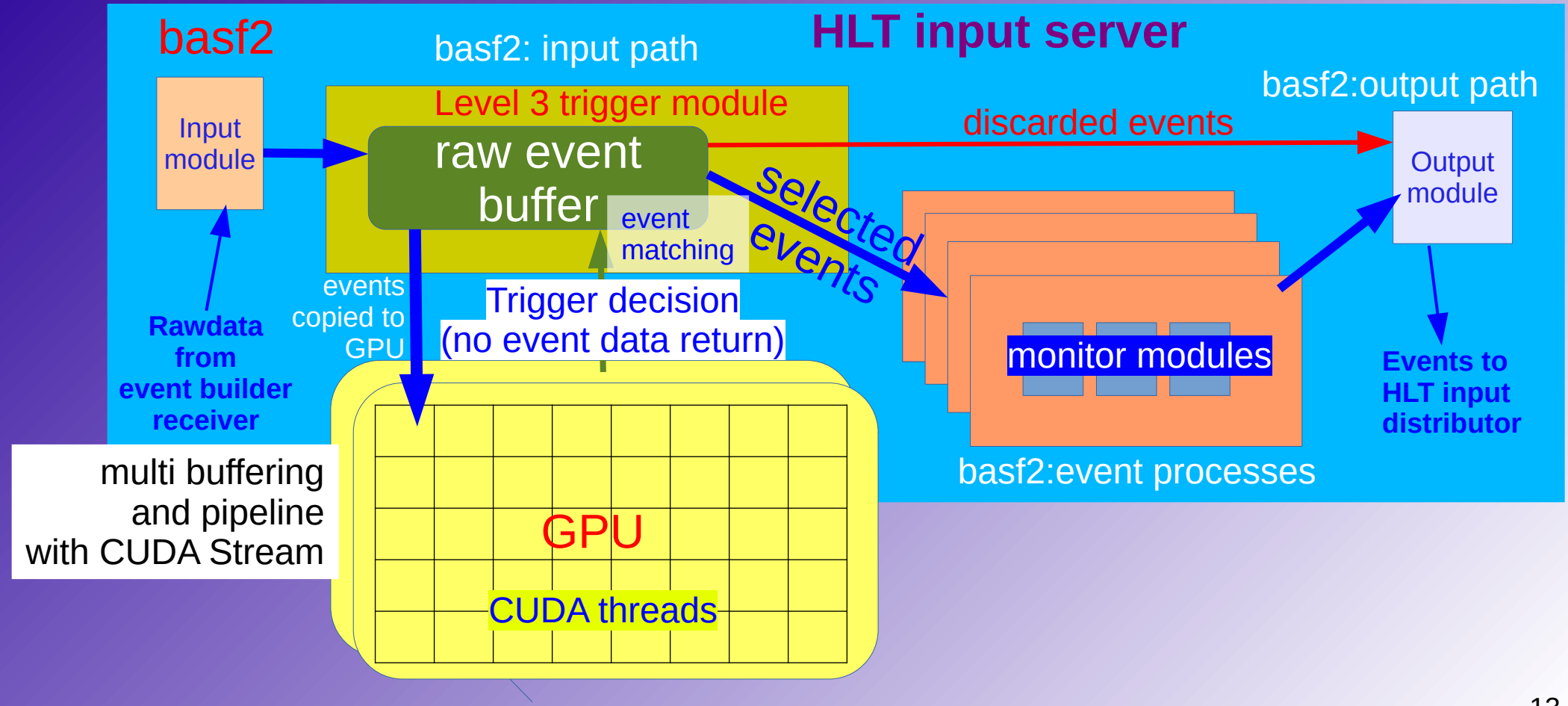
5. Data Flow for GPU

- Batch processing nature of GPU
  * GPU executes the same instruction chain for a number of different input events on corresponding CUDA threads in parallel (SIMD).
    -> “batch” processing of many events started at the same time.
  * Processing of one “batch”

1. buffer a number of events in the host memory.
2. copy them to GPU memory
3. start GPU batch processing for multiple events at the same time
4. transfer results back and compare with events in host memory
5. send only selected events to next basf2 module

- Pipelining is necessary to average out the bursty flow rate.
  ->”**CUDA Stream**” is utilized to implement multi-buffering.

Implementation of GPU Data flow in basf2



6. Implementation of Level 3 trigger code

- “Demonstration” Level 3 trigger software is now being implemented on GPU by recycling old code developed in C++ for CPU/basf2.
  <- *Consisting of fast CDC(Drift Chamber) tracking and ECL(Calorimeter) clustering.*
  => Practice of code porting to GPU and performance evaluation.

Various tips in the porting of C++ code to CUDA

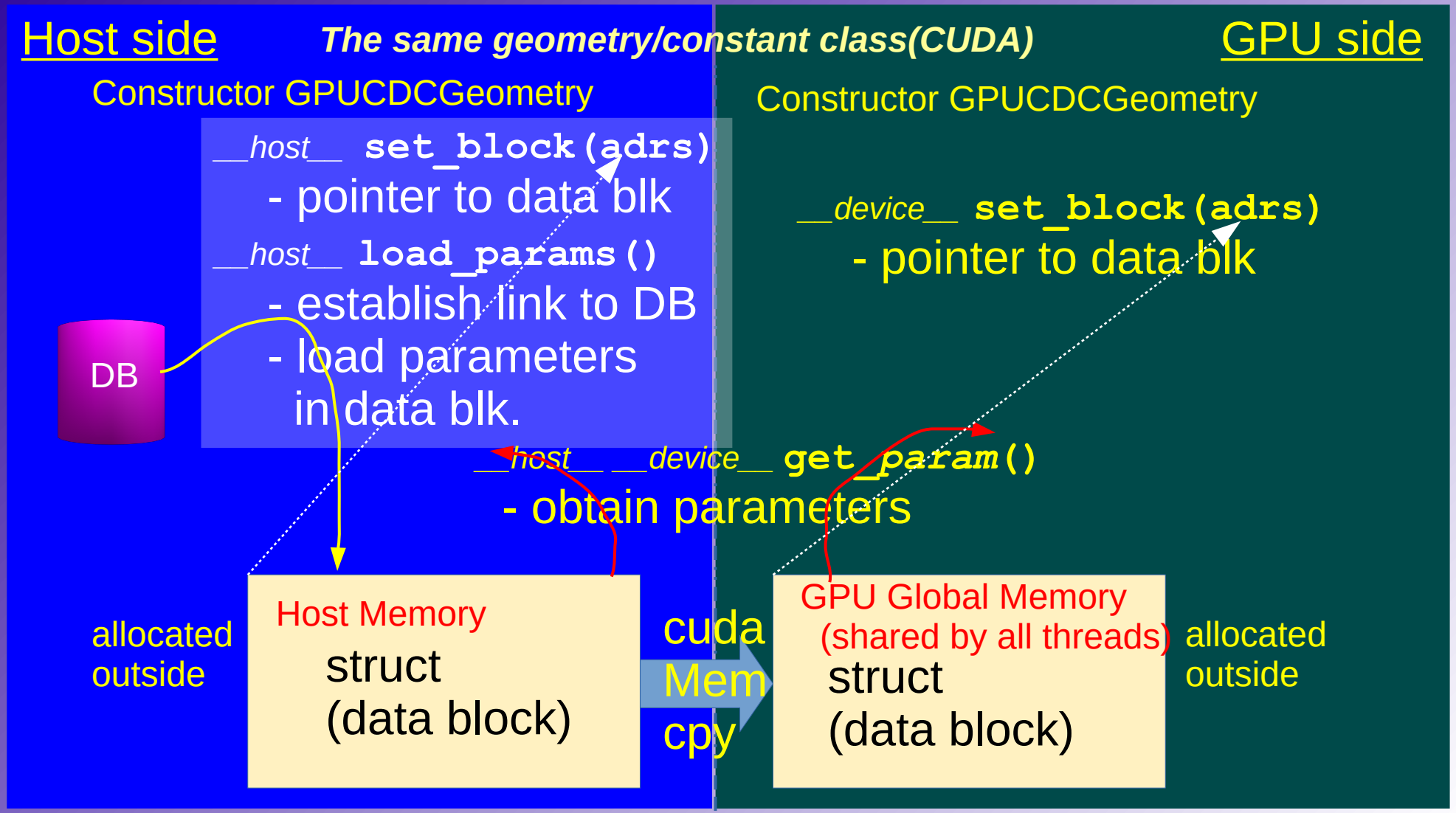
- Database access from GPU is implemented by copying the parameter memory block initialized on host from the database to GPU’s global memory (POD method).

- Dynamic object creation by C++ `new()` is replaced with a private GPU memory management using the “placement `new()`” so that the objects are created in the GPU’s global memory.

- STL (such as std::vector) / ROOT functions are replaced with home-grown libraries.

- “Double precision” floating point operations are replaced with single precision ones so as to maximize the GPU performance.

Geometry/Constants DB access from GPU (based on POD copy)



Current status of GPU-L3 project

Item	Status	Remark
Seamless software environment	Done and working	Still some problem (-j 1 is necessary...)
Data transport from host to GPU (and reverse)	Done and working	Pipelines data flow is implemented/tested.
Raw data unpacking (CDC/CL/TR)	CDC/ECL -> done TRG -> not yet	channel->wire conversion with DB
Geometry database access	CDC -> done ECL->done	Full CDC/ECL geometry is available on GPU
CDC Track Finding/Fitting	Mostly completed	confirmed CPU/GPU reproduceability
ECL clustering	Mostly completed	Validation in progress
TRG processing	not yet	
Level 3 decision	not yet	

Jingxuan Deng (Shandong U.) is working on ECL data processing.

7. Performance Evaluation

- The performance study was done using the preliminary implementation of CDC tracking code.
  * Comparison of reconstruction results (GPU and CPU)
  * Performance acceleration

- Data : Recorded beam run data taken w/o HLT filtering.

- CDC tracking code ported to CUDA was executed on a testbench GPU (A2) with a number of threads.

- The same code compiled by nvcc is also used for CPU (with `__host__ __device__` functions).

Test bench system

recycled from another purpose



```
itohepyc% lscpu
Architecture:             x86_64
  CPU op-mode(s):         32-bit, 64-bit
  Address sizes:          48 bits physical, 48 bits virtual
  Byte Order:             Little Endian
CPU(s):                   96
  On-line CPU(s) list:    0-95
Vendor ID:                AuthenticAMD
  Model name:             AMD EPYC 7643 48-Core Processor
    CPU family:           25
    Model:                1
    Thread(s) per core:   2
    Core(s) per socket:   48
    Socket(s):            1
    Stepping:             1
    Frequency boost:      enabled
    CPU(s) scaling MHz:   63%
    CPU max MHz:          3640.9170
    CPU min MHz:          1500.0000
    BogoMIPS:             4599.87
```

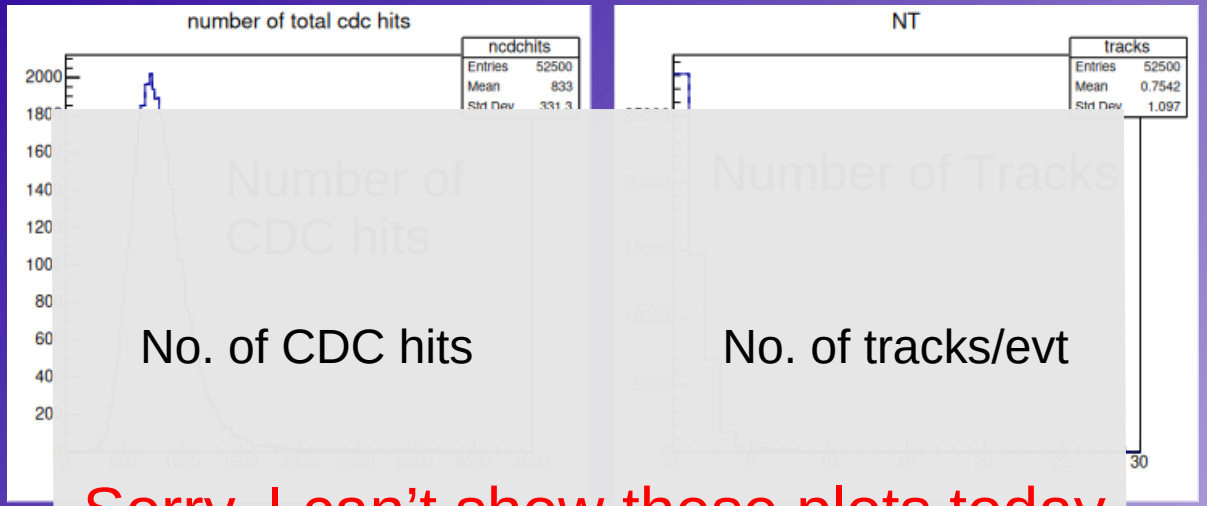
```
itohepyc% nvidia-smi
Thu Jan 30 13:59:22 2025
+-----------------------------------------------------------------------------------------+
| NVIDIA-SMI 565.57.01              Driver Version: 565.57.01      CUDA Version: 12.7     |
|-----------------------------------------+------------------------+----------------------+
| GPU  Name                 Persistence-M | Bus-Id          Disp.A | Volatile Uncorr. ECC |
| Fan  Temp   Perf          Pwr:Usage/Cap |           Memory-Usage | GPU-Util  Compute M. |
|                                         |                        |               MIG M. |
|=========================================+========================+======================|
|   0  NVIDIA A2                      Off |   00000000:81:00.0 Off |                    0 |
|  0%   33C    P8              5W /   60W |       1MiB /  15356MiB |      0%      Default |
|                                         |                        |                  N/A |
+-----------------------------------------+------------------------+----------------------+
```

CPU : AMD EPYC 7643
(48 cores/96 threads)

```
itohepyc% cat /etc/redhat-release
Rocky Linux release 9.5 (Blue Onyx)
```

GPU : NVIDIA A2 Tensor
(1280 CUDA cores;
10 RT cores;
40 Tensor cores)

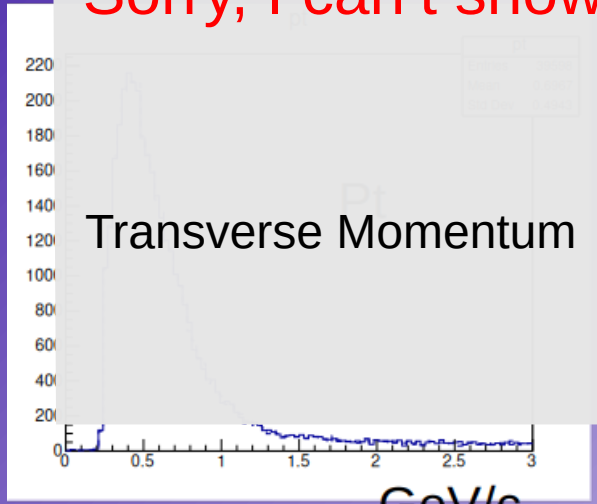
a) Reconstruction result



No. of CDC hits

No. of tracks/evt

Sorry, I can't show these plots today.



Transverse Momentum

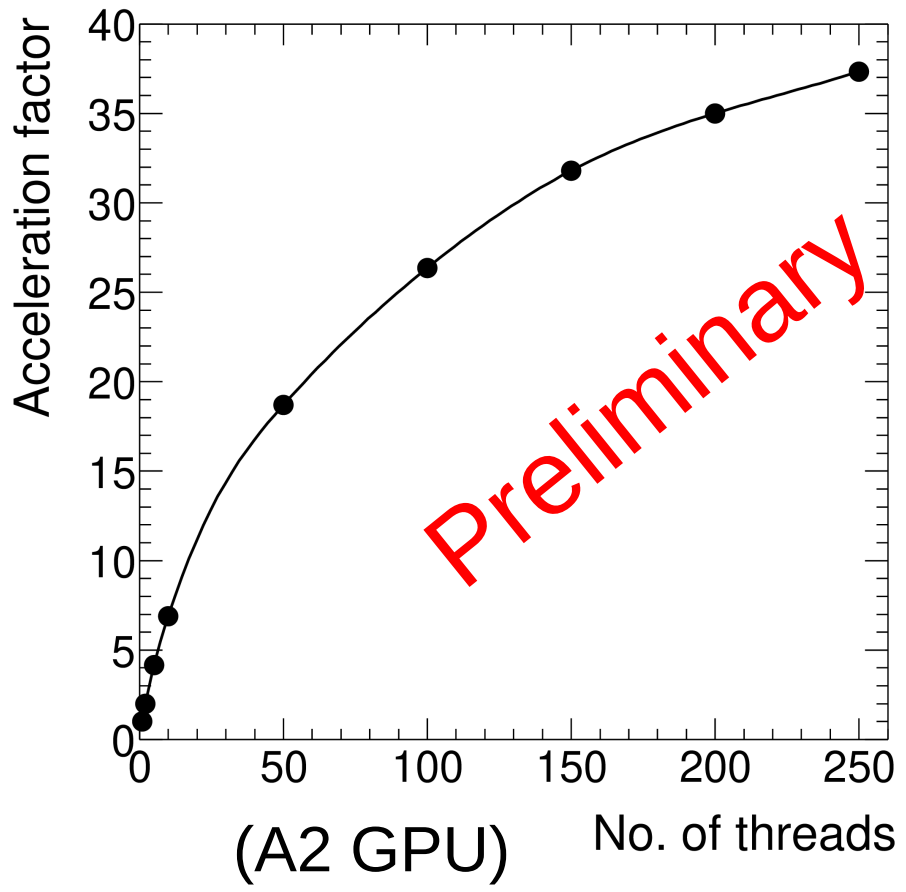
* The CDC tracking code is still under development and its implementation is preliminary.

* Tracking performance is not optimized at all.

* Good agreement between GPU and CPU processing results.

* Further detailed check is on-going.

b) GPU acceleration



- Processing rate was measured by varying the number of CUDA threads.

- Acceleration factor is obtained from the processing time relative to that with a single thread.

* The performance was measured **without pipelining**.
  = synchronous exec. of
  - data copy to GPU
  - GPU processing
  - result return to host.

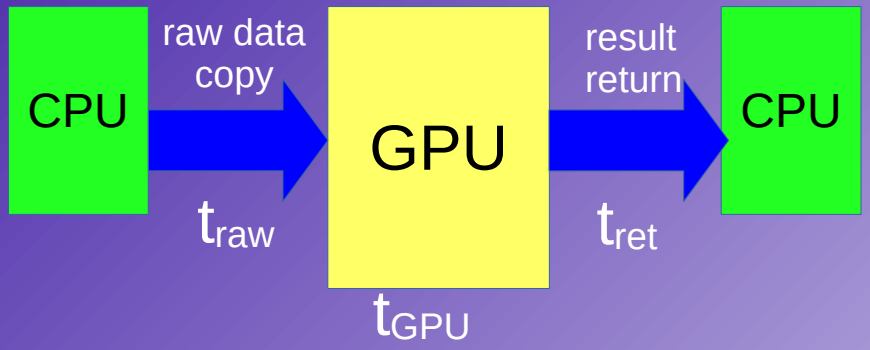
* Currently the number of events / batch = GPU threads

Observation:

* Rapid performance increase when no. of threads is small, but the increase gradually saturates for larger number of threads.

Explanation:

The GPU data flow is:



For small no. of threads

$t_{GPU} >> t_{(raw+ret)}$

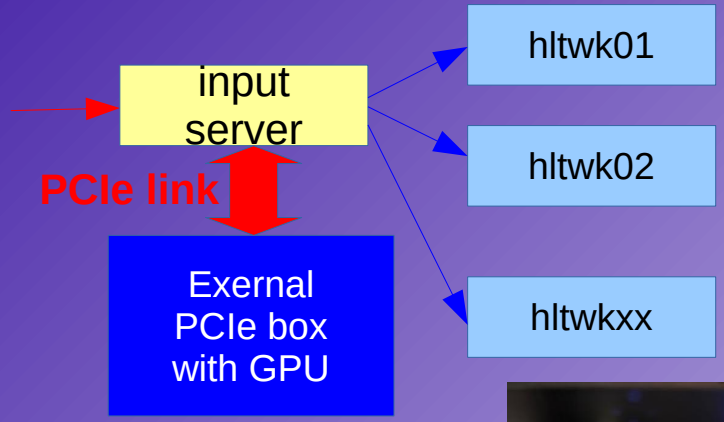
But when no. of thread becomes larger, $t_{(raw+ret)}$ dominates.

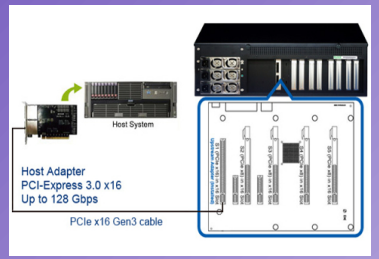
<- Data size becomes larger and the time for data copy becomes longer.

Mitigation : Pipelining using CUDA Stream

8. GPU Implementation in HLT unit

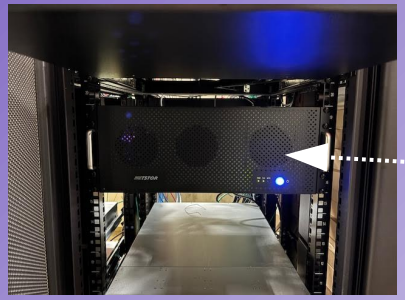
- Place GPU in an external rack-mount 4U unit (eGPU).
- Connect it to existing HLT input server via a special PCIe link.

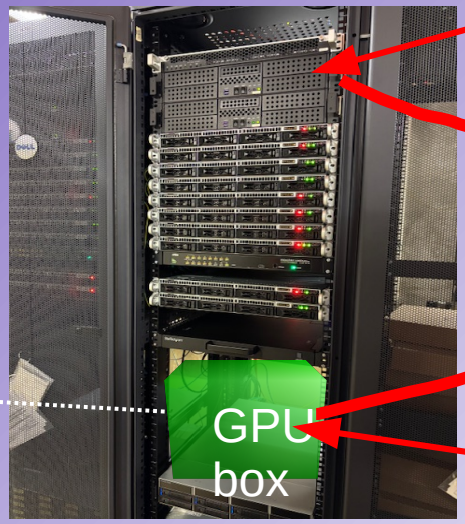




netstor NA265A-G4







input server

PCIe-link

GPU box

* Currently implemented in one HLT unit (out of 15).

Choice of GPU for Belle II HLT

* A GPU houses >10,000 CUDA cores.
* Adding 1 GPU for each of 15 HLT input servers is enough for our purpose.

-> in total of 150,000 CUDA cores

* GPU with better performance in fp64 is desired but hopefully less dependent (assumption to rely mostly on fp32).
* Vector/Tensor cores are not supposed to be used for the planned processing.

RTX 5090 : 21,760 cores, 32GB, ~100TFLOPS(fp32) / Blackwell

RTX 5000Ada : 12,448 cores, 32GB, 61TFLOPS(fp32)

RTX 6000Ada : 18,176 cores, 48GB, 91.1TFLOPS(fp32)

H100 : 14,592 cores, 80GB, 30TFLOPS(fp32)

A2 : 1,280 cores, 16GB, 4.5TFLOPS(fp32) : (used in test bench)

9. Summary

- An implementation of the GPU based Level 3 trigger in the existing HLT is planned to mitigate the unexpected high BGs.

- A seamless CPU-GPU software environment has been successfully implemented in the Belle II software framework.

- A preliminary CDC tracking code was implemented in the test bench GPU and the system performance was evaluated.

- The performance test in beam is planned next year with the full implementation of the Level 3 trigger in the target GPU.