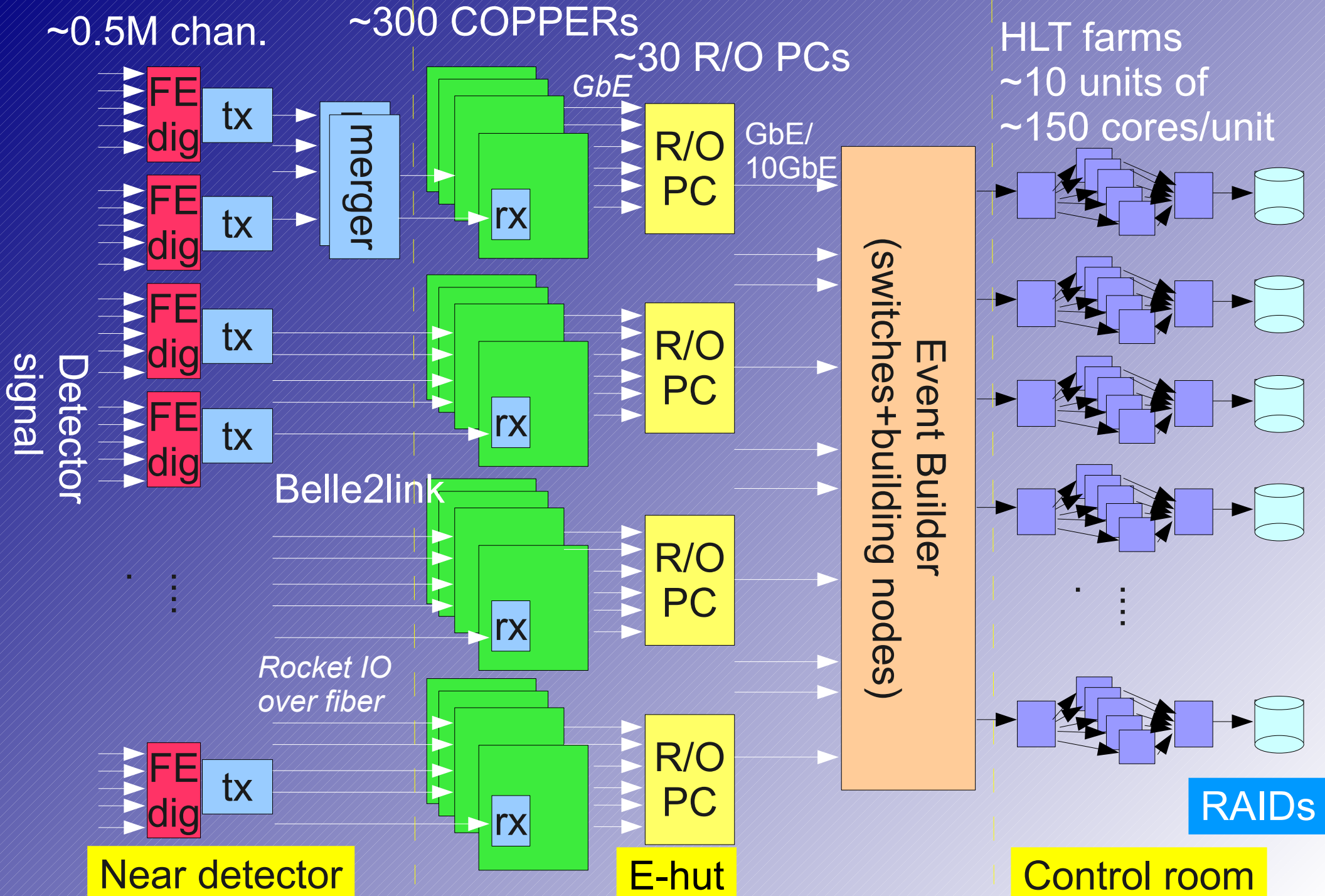


# Data flow and HLT in Belle II DAQ

R.Itoh, KEK

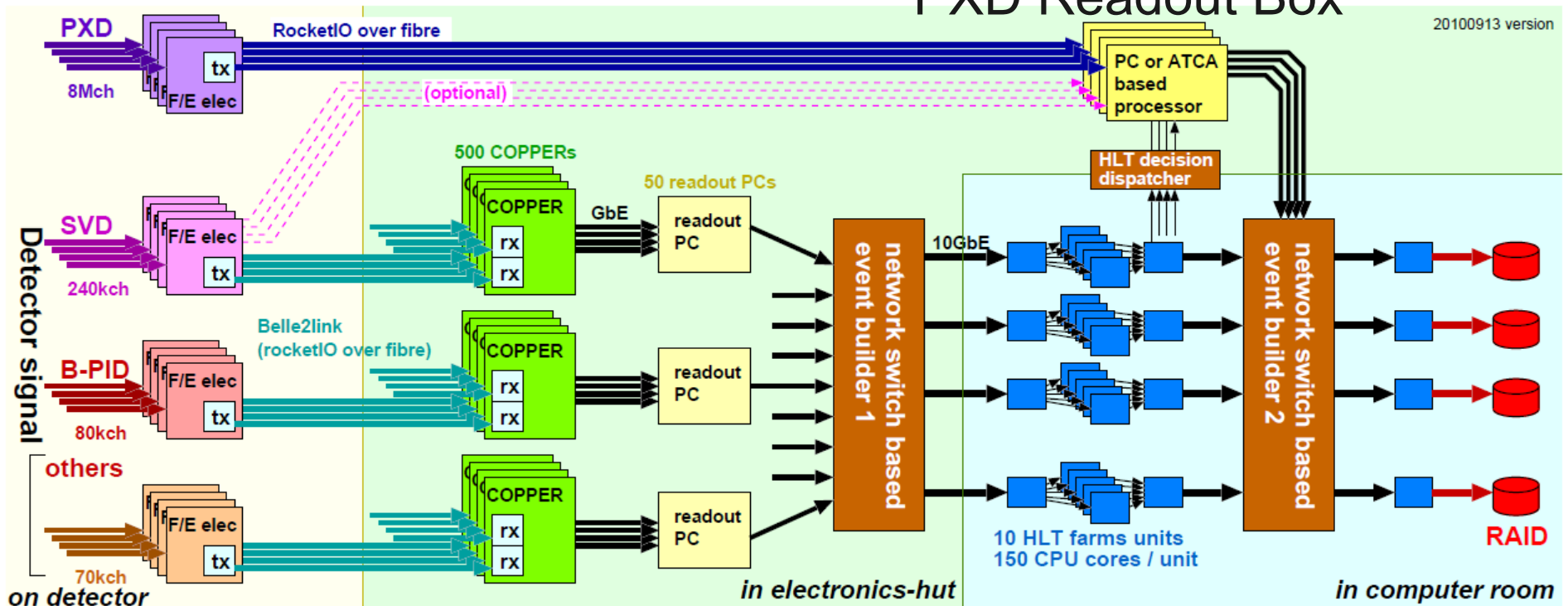
# Global DAQ Design (w/o PXD)

\* Timing dist. scheme is not included in this figure.



# “PXD Readout Box”

20100913 version



## Role of “PXD Readout Box”

- Associate PXD hits with tracks and reduce data size to be sent.

## 3 options:

- ATCA: Self tracking using SVD feed + reduction (Challenge)
- ATCA: HLT tracking + reduction (Baseline)
- PC: HLT tracking + reduction (Backup)

3

- \* Two ATCA options are merged as “Option 12” with SVD feed.
- \* HLT interface is implemented regardless of self tracking or not.

## Detail of detector readout requirements

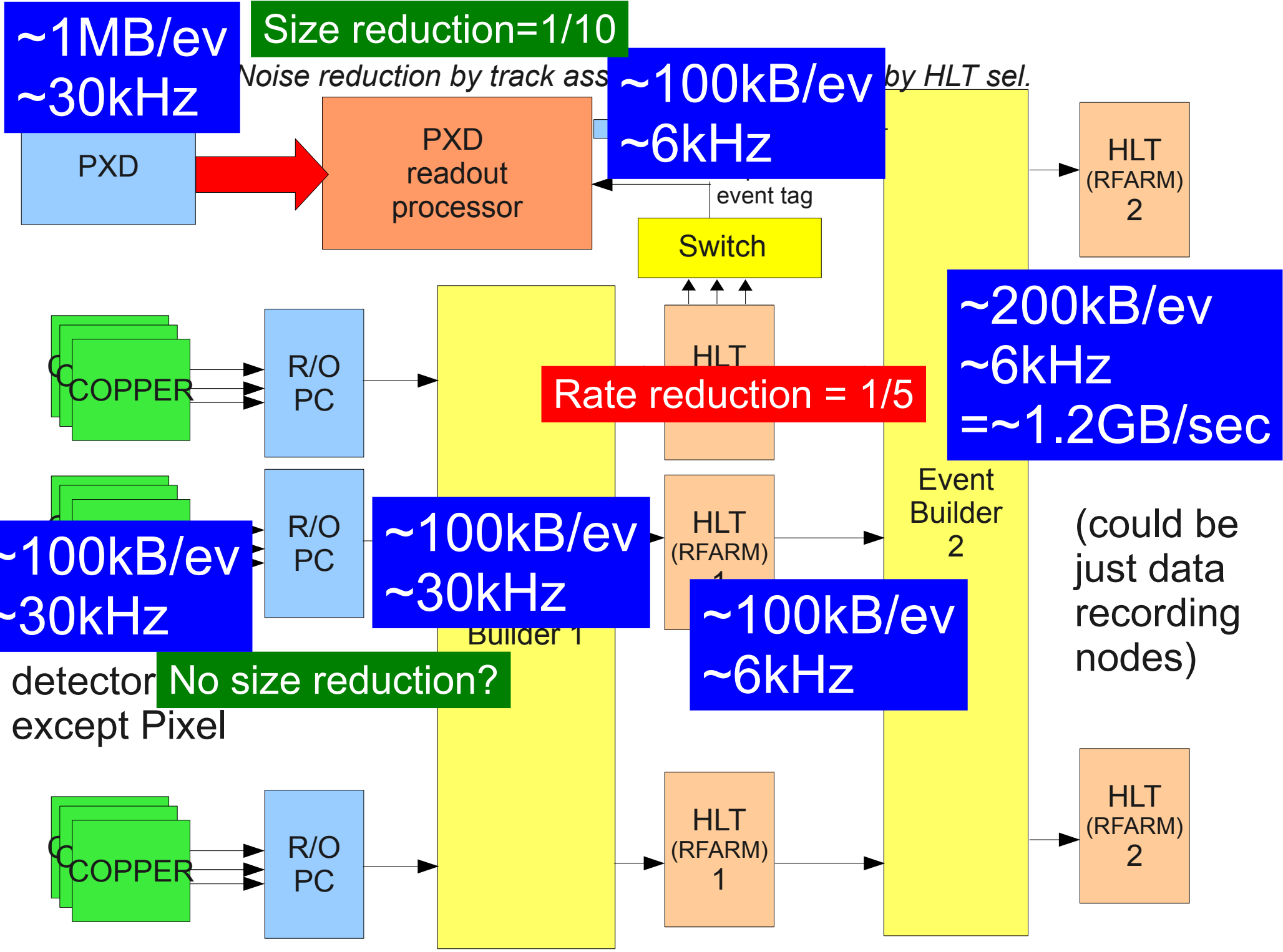
	#ch	occ [%]	#link	/link [B/s]	FNS	#CPR	ch sz [B]	ev sz [B]	total [B/s]	/CPR [B/s]
PXD	8M	2	40	600M/DHH	—	—	4	800k	24G	—
SVD	243456	1.9	40	13.8M	HSLB	40	4	18.5k	555M	13.8M
CDC	15104	10	302	0.6M	HSLB	75	4	6k	175M	2.3M
BPID	8192	2.5	128	7.5M	DSP	16	16	4k	120M	8M
EPID	77760	1.3	138	0.87M	HSLB	35	0.5	4k	120M	15M
ECL	8736	33	52	7.7M	HSLB	26	4	12k	360M	15M
BKLM	21696	1	86	9.7M	HSLB	6	8	2K	60M	10M
EKLM	16800	2	66	19.5M	DSP/HSLB	5	4	1.4k	42M	5.3M
TRG					HSLB	60				

\* These numbers are still preliminary and subject to change

Current assumption for DAQ design

Event Size : 1MB from PXD, 100kB from other detectors

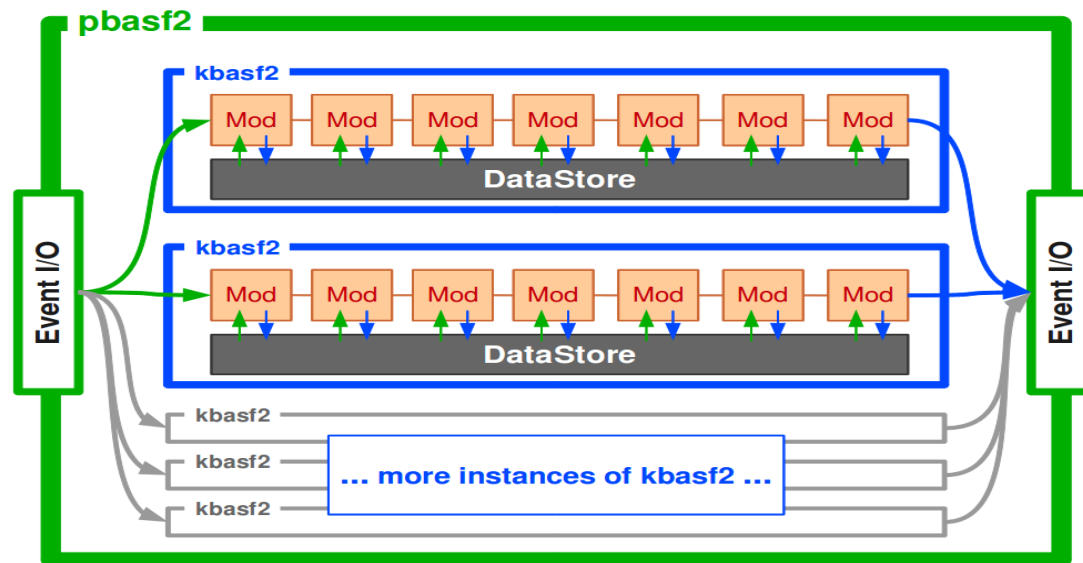
L1 trigger : max 30kHz (ave. rate)



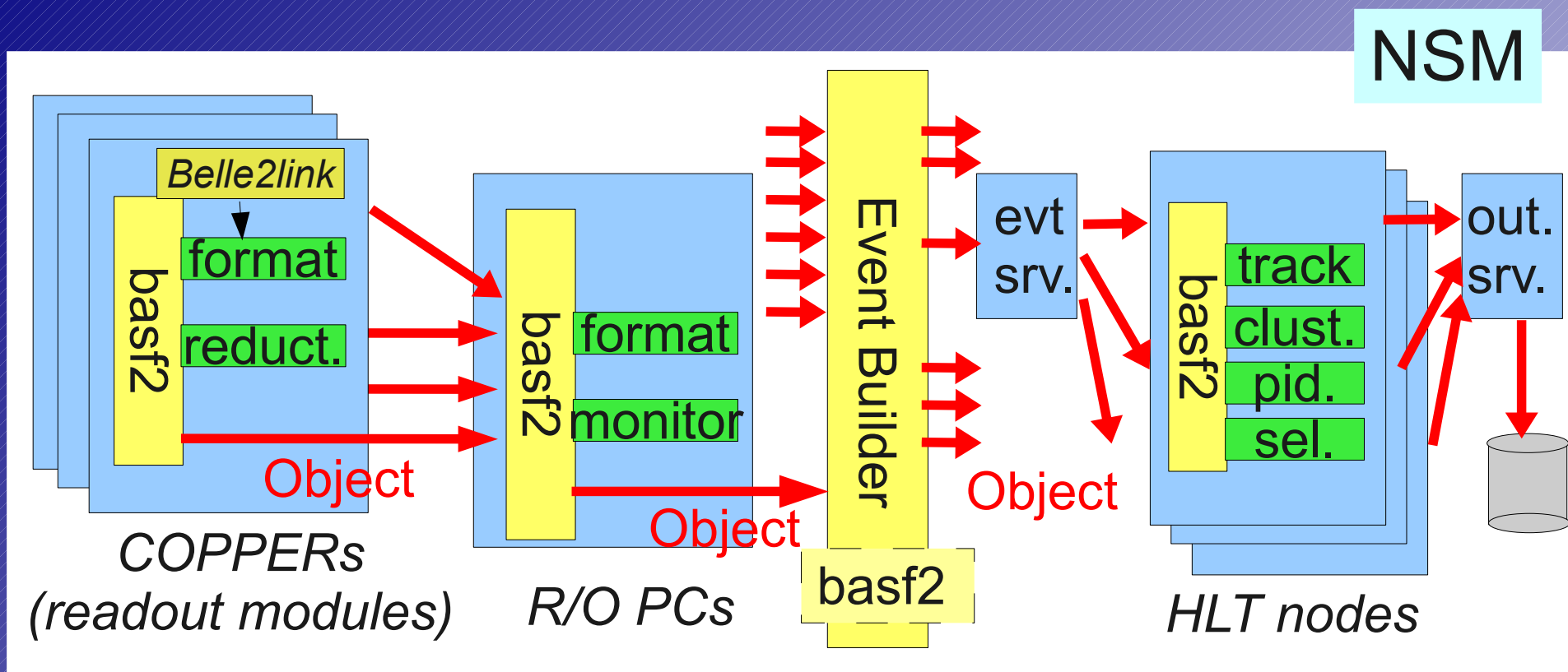
The data processing in DAQ is supposed to be done by using “basf2”.

## basf2: unified framework in Belle II

- Successor of BASF in Belle
- Used both in DAQ and offline. No differences/boundaries,
- Object oriented data handling using ROOT I/O
- Python is used for scripting
- Parallel processing is implemented as a super framework for basf2 kernel (but the boundary is not recognized by users).
- Extension of functionality to the framework is supposed to be done by adding modules.
  - > HLT framework is built outside basf2 by adding interface I/O modules to talk to socket.



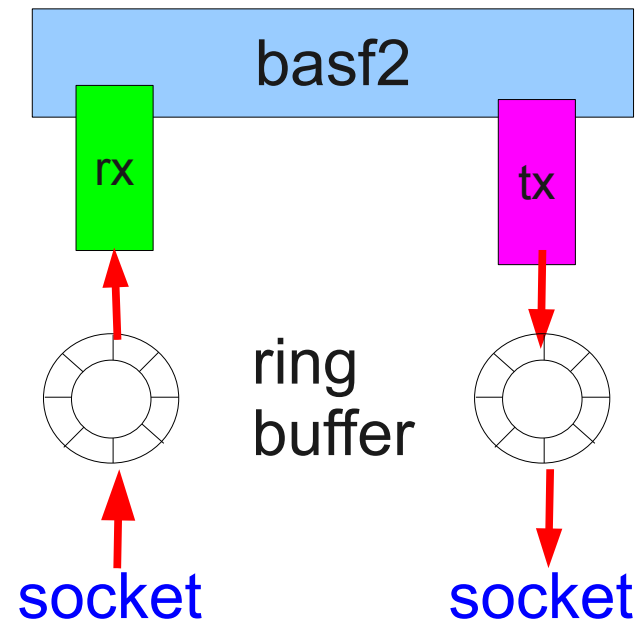
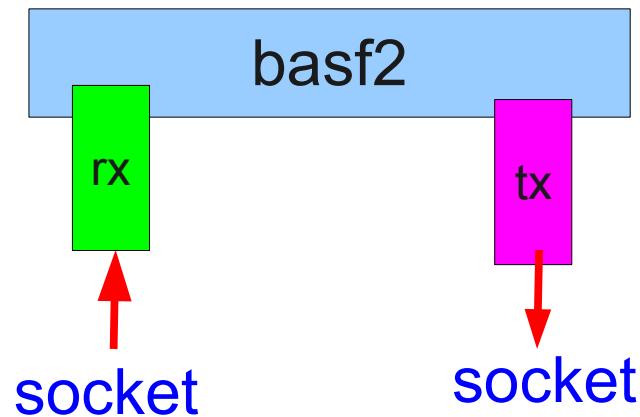
# Object-oriented data flow and processing in Belle II DAQ



- The same software framework from COPPER to HLT (and offline).
- “Object” based data flow throughout DAQ
  - > class to contain streamed object
- ROOT IO based object persistency at storage
- NSM for the system-wide control

## Object transfer over network

- Streamed (serialized) object are transferred using standard UNIX socket. (ROOT's TSocket is not used.)
- Low level socket interface (B2Socket) is being prepared by Soohyung. Possibility to use IPv6 as suggested by Yamagata-san.
- basf2 I/O interface is provided as modules “tx” and “rx” which serialize/deserialize objects
- In some cases, a Ring Buffer is sandwiched between socket tx/rx and actual basf2 module to average the data flow rate.





# EvtMessage

- serialized object format passed between cores/PC nodes

Header (16words) (tentative, something similar to d2packet\_header)

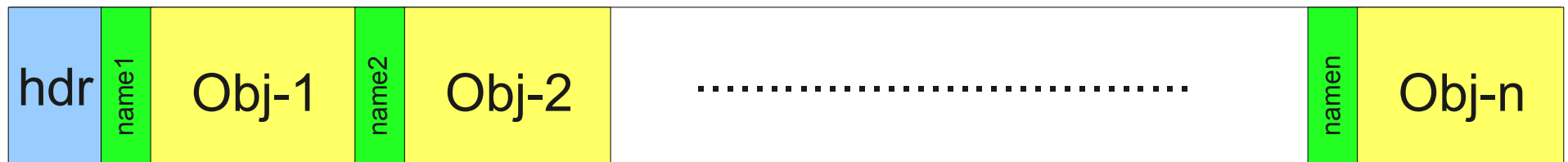
word 0 : Number of bytes in this record  
word 1 : RECORD\_TYPE (begin\_run, event, end\_run, others)  
word 2,3 : Time stamp (gettimeofday() format)  
word 4 : source of this message  
word 5 : dest. of this message  
word 6-15 : Reserved (used to store durability, nobjs and narrays for DataStore objects)

List of serialized objects (word 16-)

Streamed object :

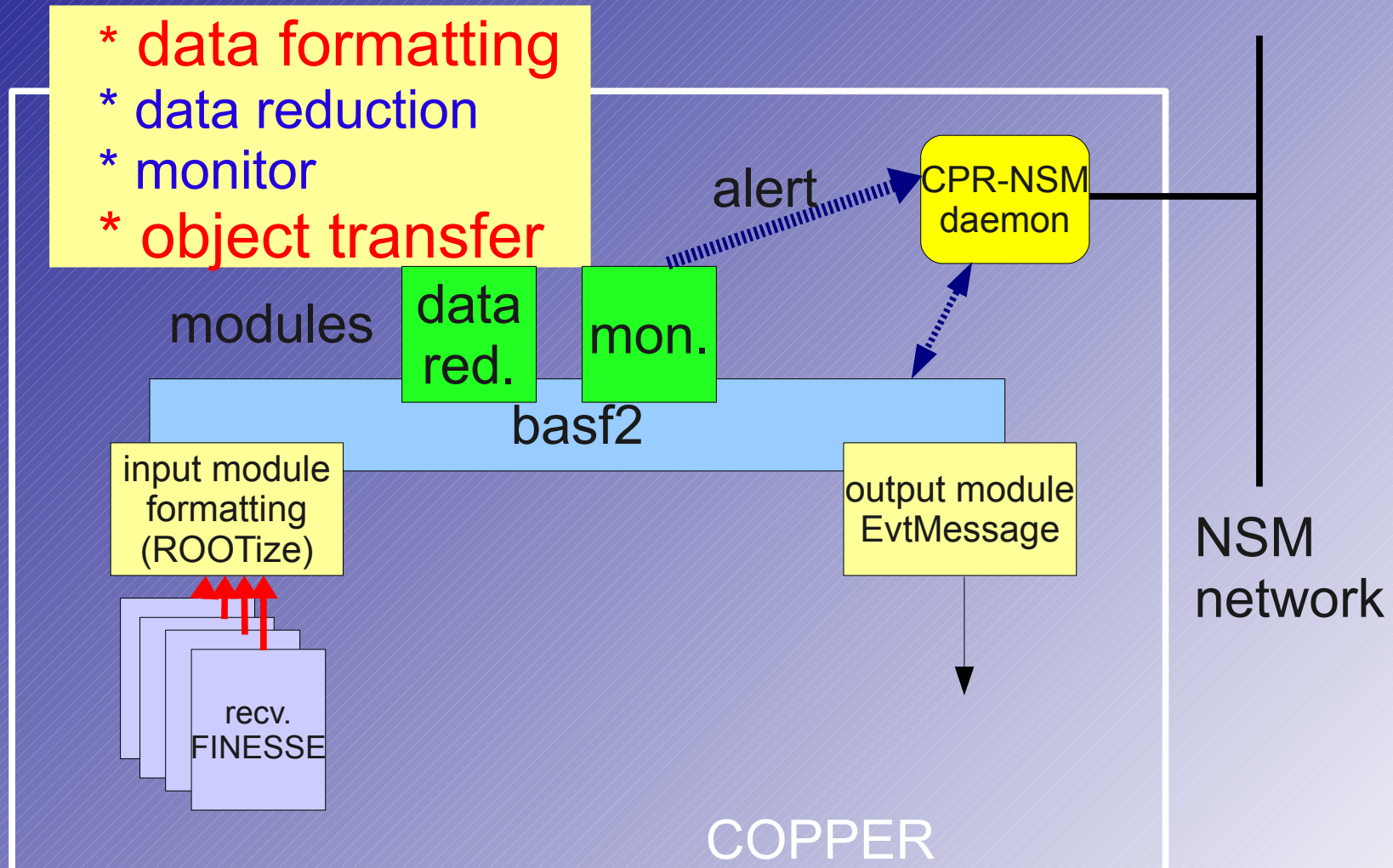
word 1 : nbytes of object name  
bytes : object name

word n : nbytes of streamed object  
bytes : streamed object



# COPPER

- Input : read receiver FINESSE and format it to rawdata object of each detector (ROOTization).
- Output: stream objects and send EvtMessage to raw socket
- Modules :



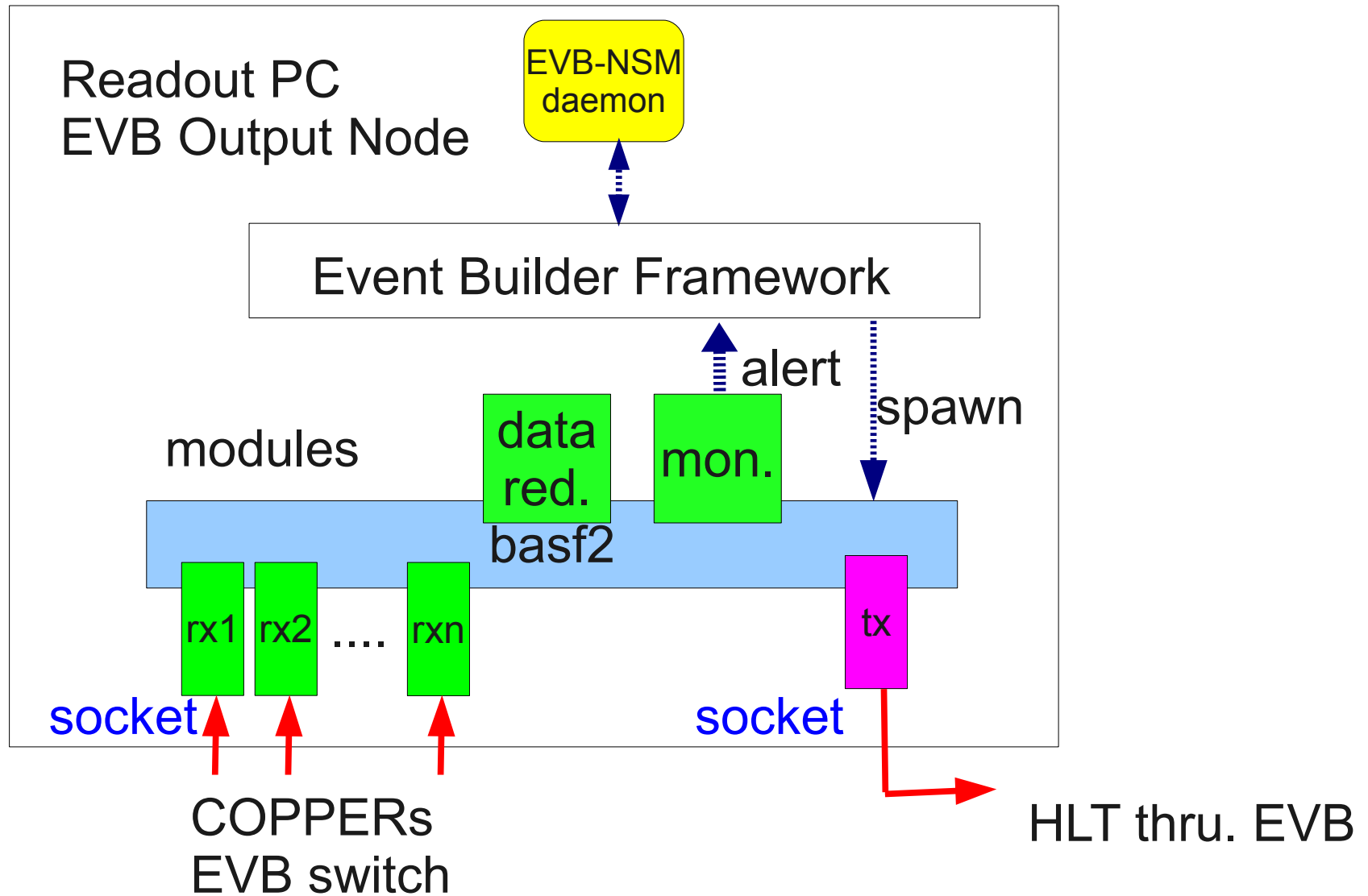
## Streaming raw data

- \* Raw event data have to be stored in “DataStore” as an object so as to be managed by basf2.
- \* To transfer the raw data to different node, the DataStore has to be streamed (serialized) and destreamed(deserialized).
- \* Streaming using ROOT is reported to be CPU consuming.  
-> could be an issue for COPPER CPU (ATOM@1.6GHz).



- \* Use the simplest structure of raw data :  
“variable sized array of integer”
  - Overhead of ROOT streaming can be minimized.
  - If still slow, optimized streamer (=handwritten streamer) will be implemented.
  - Data access through accessing class (just like Belle's TdcUnpacker class)

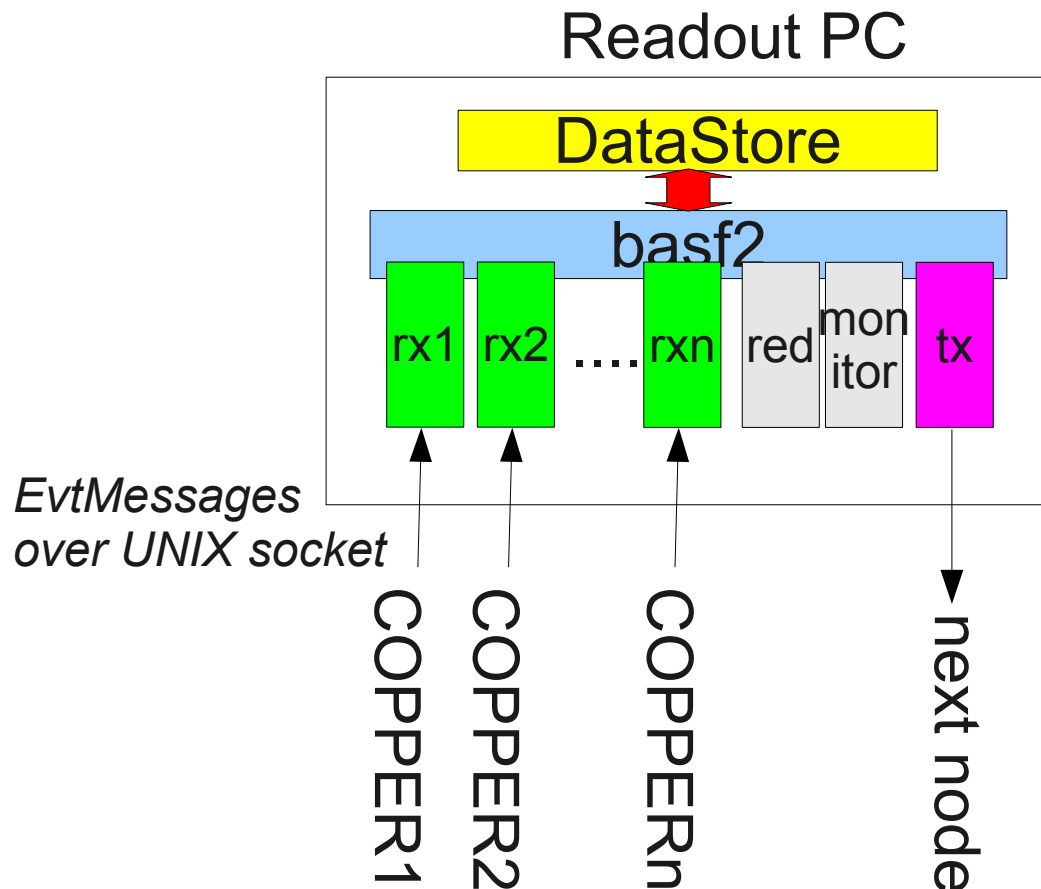
# Event Building





Event sender(tx) and receiver(rx) are coded as “modules” for basf2.

## Event Building with “DataStore” object

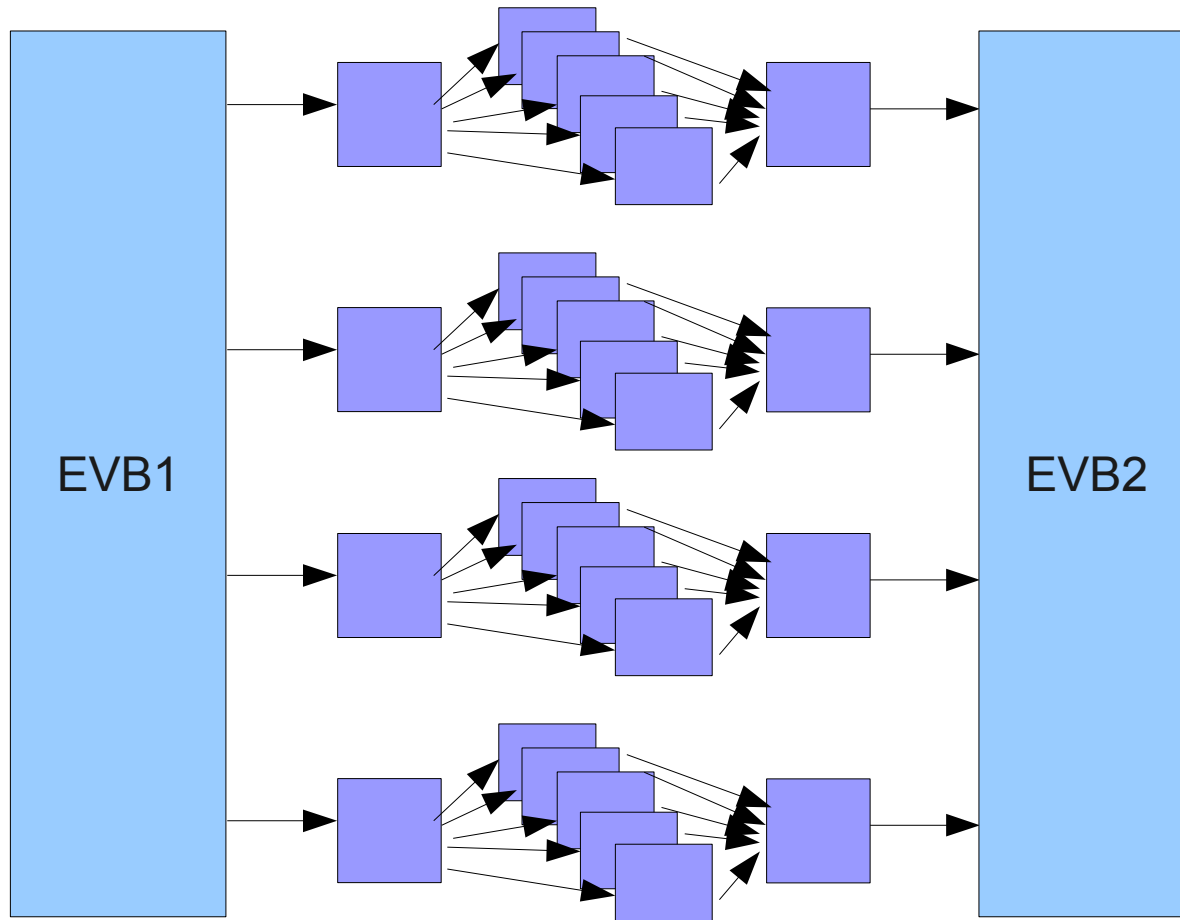
- On each node of Readout PC and Event Builder output PC, software event building in has to be implemented in basf2.
  - <-> Transport of event fragment is taken care by “Event Builder” (Yamagata-san)
- Event builder interface (“tx” and “rx”) is provided as **basf2 modules**.



-  Input module of basf2
  - \* Read one event fragment from a socket and save it in DataStore
-  Output module of basf2
  - \* Stream DataStore and send it to a socket.

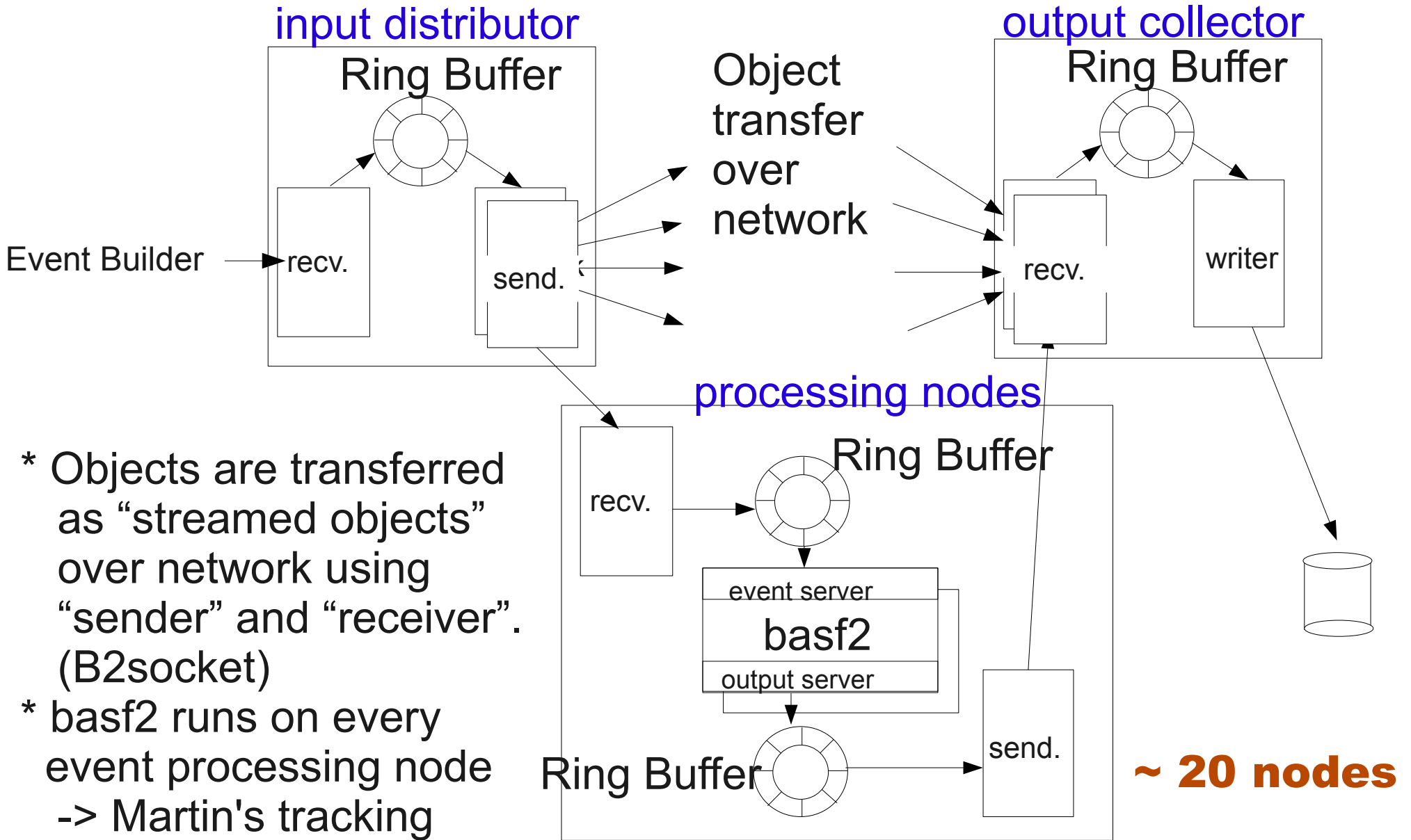
- By reading all sockets and placing them in DataStore, the event building is automatically ensured.

# HLT Configuration



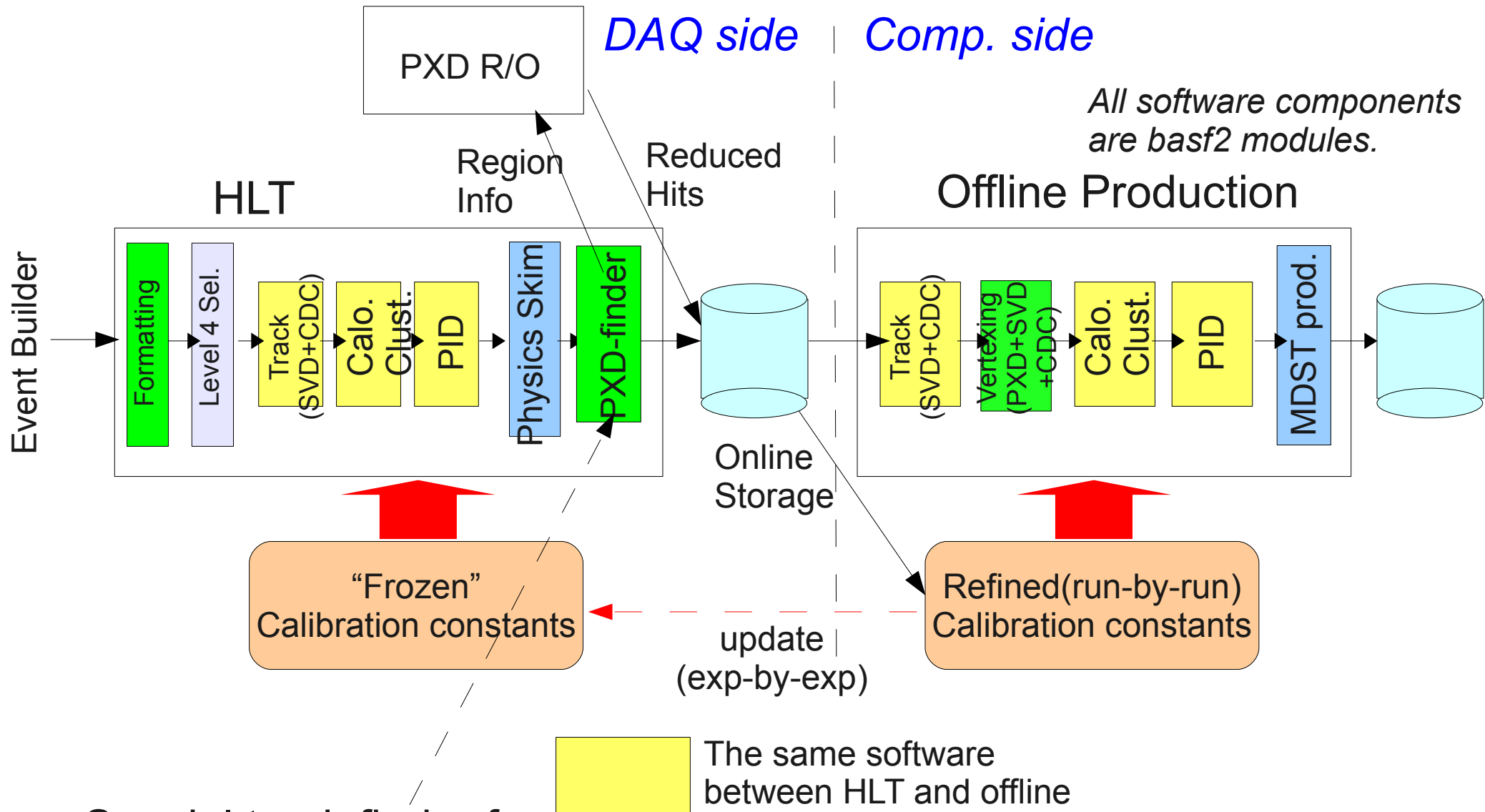
- Consists of  $O(10)$  HLT units
- One unit is supposed to process  $L=3\sim 5 \times 10^{34}$
- Units can be easily added latest to keep up with the luminosity increase.

# HLT data flow



- \* Objects are transferred as “streamed objects” over network using “sender” and “receiver”. (B2socket)
- \* basf2 runs on every event processing node -> Martin's tracking framework runs “as is”.

# HLT Event Processing chain



Special track finder for track-PXD hit association

\* Only for the HLT-triggered events *already parallel in 200 \* 10 cores*

\* Can make use of SVD+CDC raw data + full tracking results also



## Required processing power for HLT

- Estimation of “number of cores” required for HLT.
- The experience with Belle's RFARM.
  - \* We had 2 units of RFARM in Belle.
  - \* One unit consists of 40 servers each of which is equipped with dual **Xeon@3.4GHz**. They are operated w/o HyperThreading.
  - \* The performance of one unit almost saturated a data feed with a luminosity of  $1.5 \times 10^{34}$
- Phase 1(t=0) target luminosity of HLT :  $2 \times 10^{35}$ 
  - > Keep up with further luminosity increase by adding more HLT units later.
- The reconstruction software could become slower at Belle II because of higher detector granularity -> assume 50% increase.
- Estimated number of cores (3GHz) required for phase 1.  
$$40 \times 2(\text{dual}) \times (2 \times 10^{35}) / (1.5 \times 10^{34}) (\text{Lum}) \times 1.5(\text{soft}) \times (3.4/3.0)(\text{clk})$$
$$= 1813 \text{ cores@3GHz. } \Rightarrow \sim 2000 \text{ cores@3GHz are required.}$$

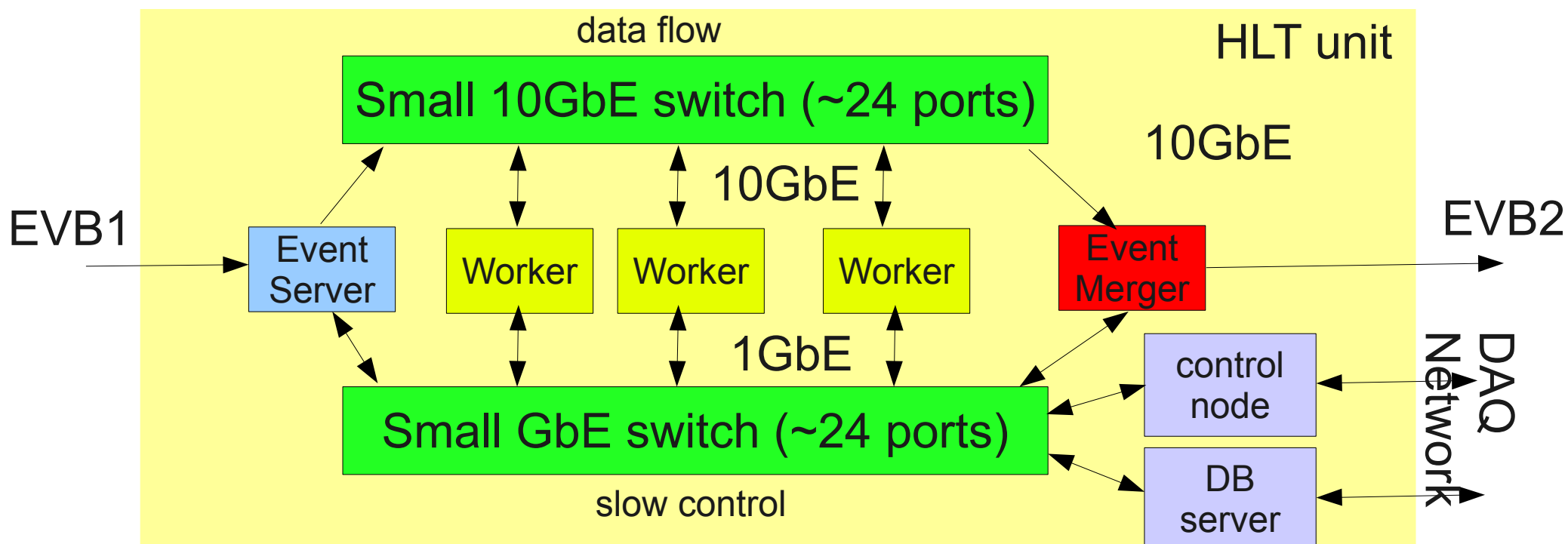
## HLT unit

- HLT has “unit” structure so that the processing power can be easily added.
- It is preferred that the number of unit is kept within a reasonable numbers considering the event builder cost.
  - > Current design:
    - \* ~ 5 units for phase 1.
    - \* Up to ~10-15 units at maximum.
- The number of cores / unit for phase 1.
  - >  $2000/5 = 400$  cores / unit.
- How do we house 400 cores in a unit?
  - \* The number of PC servers in a unit is preferred to be less to reduce the # of network port to use cheap switch.

## The data flow rate/server

- Maximum input data flow for full HLT can reach 3GB/sec ( $\leftarrow 100\text{kB} * 30\text{kHz}$ ). The output could be similar if we keep HLT processing results together with raw data ( $\leftarrow$  normally removed but might be left in some cases.)
- With 10 HLT units, the data flow/unit is 300MB/sec for both input and output.  $\rightarrow$  600MB/sec for in-out.
- With 20 servers/unit, the data flow/server  $\sim$ 30MB/sec for in-out.  
 $\Rightarrow$  This rate is within GbE bandwidth, but not enough margin.....
- Recently small 10GbE switch becomes cheaper (like Dell's PowerConnect 8024, 24port 10GbE-T switch @ 100-man-yen)

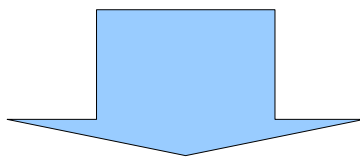
## Realistic configuration of one HLT unit



- One unit is equipped with ~20 event processing servers.  
=> required cores / server =  $400/20 = 20/\text{server}$ .  
=> necessity of “many core” server
- The number may have to be increased because of the lower clock rate for a core in “many core” CPUs.

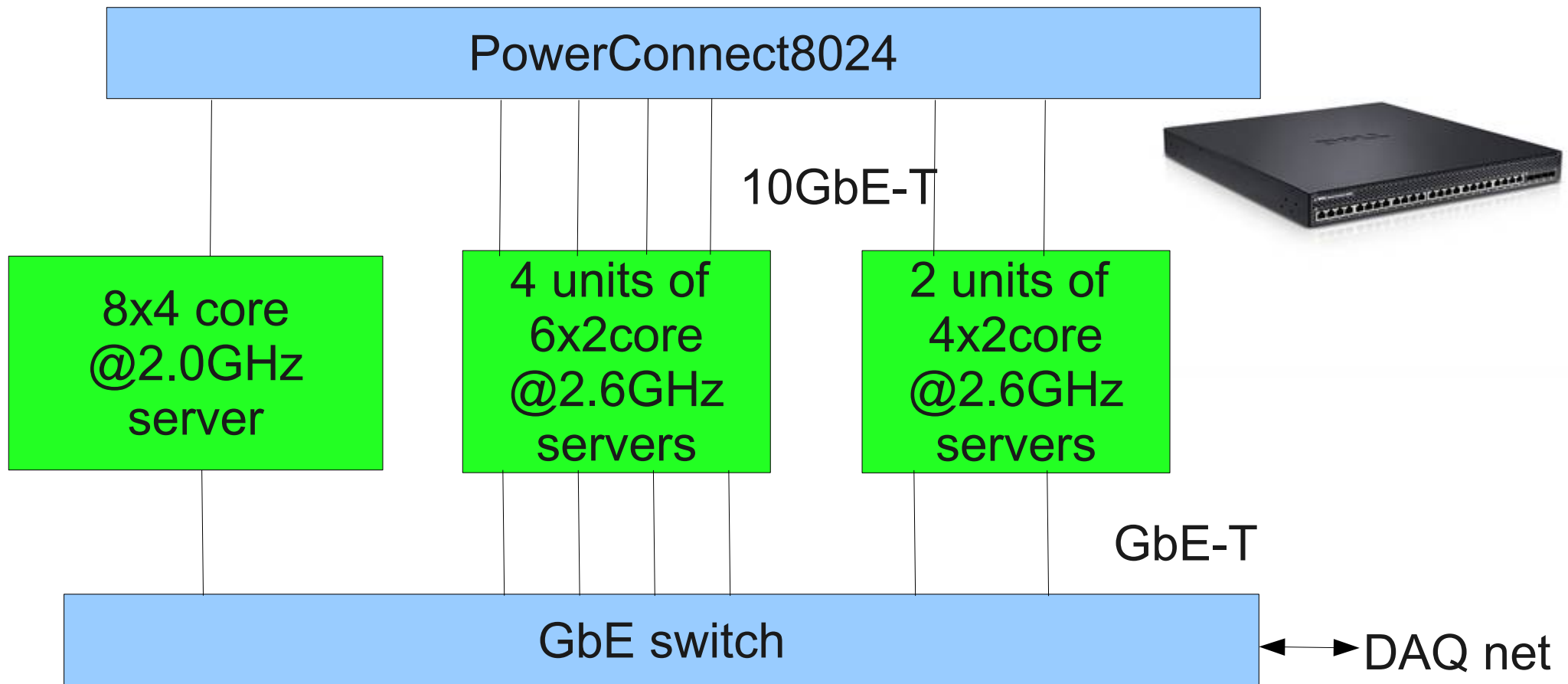
## “Many core” HLT test bench

- “Many core” servers are the essential component for the event processing nodes.
- Parallel processing performance on “many core” servers is a key to have enough processing power / unit.

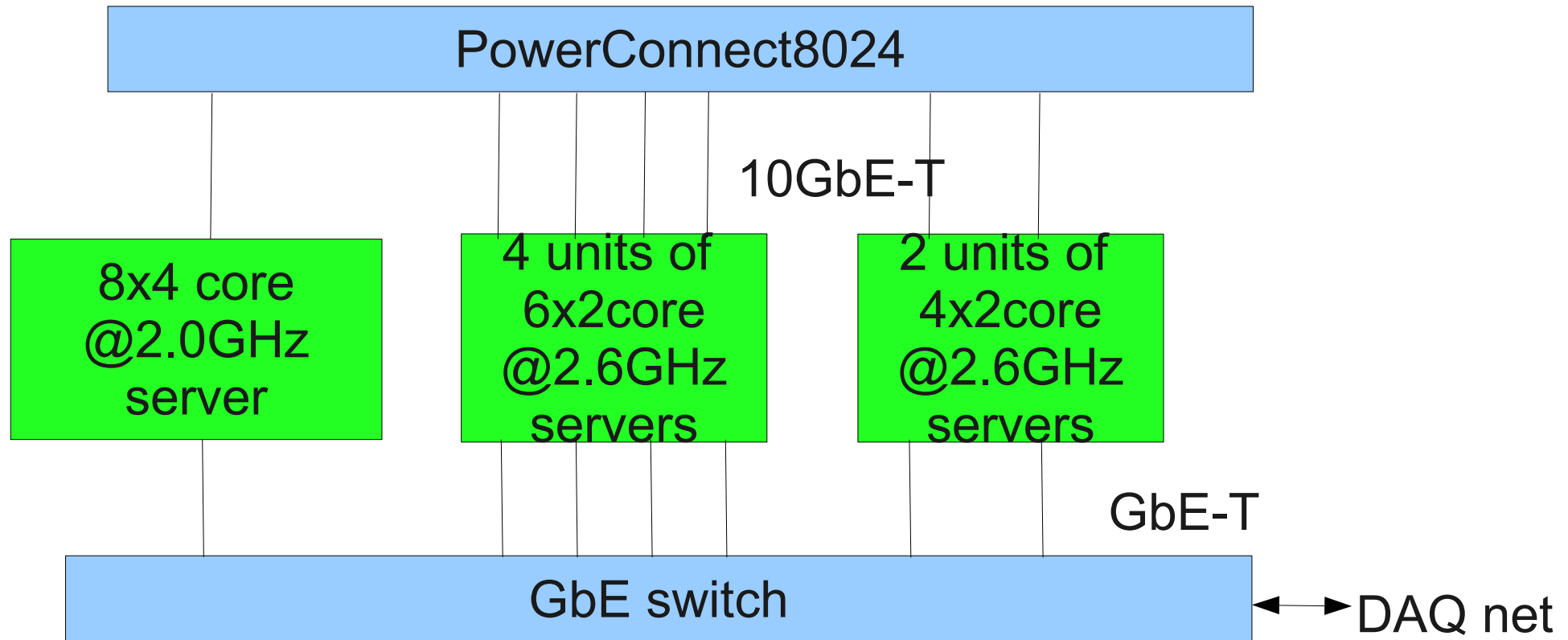


New HLT test bench with a set of “many core” servers connected via 10GbE-T.

\* This test bench is shared with Event Builder R&D for the test of 10GbE-T.



- PowerConnect8024 and 8x4=32 core server has been delivered.
- 32 core server was broken after one day operation
  - > sent back to the manufacturer
- 12 core servers will be delivered in this month.
- 8 core servers (purchased for other purpose) will be delivered in January.



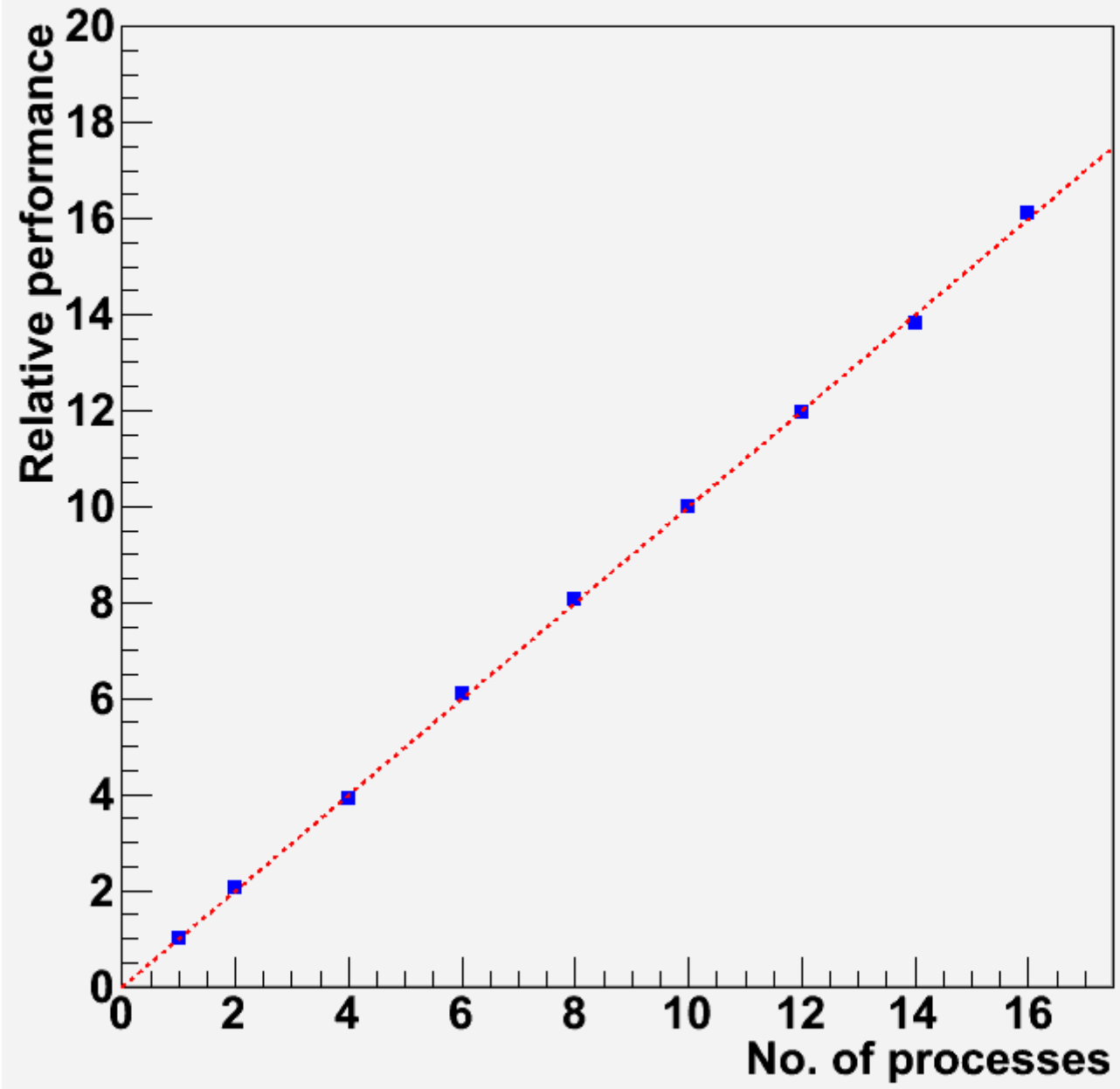
- Will replace existing HLT test bench built by recycling backup DAQ servers for Belle in coming months.
- In next JFY, more servers will be added together with a test recording RAID setup to form a full single HLT unit.
  
- The software development is in progress
  - \* Parallel processing on many cores
    - > integrated in basf2, being tested.
  
  - \* Parallel processing on network cluster
  - \* HLT configuration scheme
  - \* Real time monitoring scheme.
    - > “hbasf2” by Soohyung, which is a superframework for basf2
  
- The preliminary performance test of “multicore” server has been done (last Monday!),



## Processing performance of the 32-core server

- Benchmark : basf2 with parallel processing turned on
  - Used script :
    - \* Read event data file using SeqRootInput(same as output)
    - \* Belle II Geant4 simulation (particle gun, np=30)
    - \* Write CDC hits into a file using SeqRootOutput
    - \* Time for 1000 events is measured
  - Processing load : 3.5 sec/event
  - Data flow (the same for in and out) : 1.5MB/event
  - Linearity is measured by varying the number of processes for parallel processing
- \* Currently basf2 has problem in parallel processing for more than 16 processes and the performance up to 16 has been measured.
- \* The performance is observed to increase clearly linear to the number of processes.
- > “Many core server” approach seems to be promising, but >16 performance should be checked carefully.

# Linearity



- hbasf2: Super-framework supposed to be used in HLT

- Features

- Data transferring via network
- Node management/monitoring

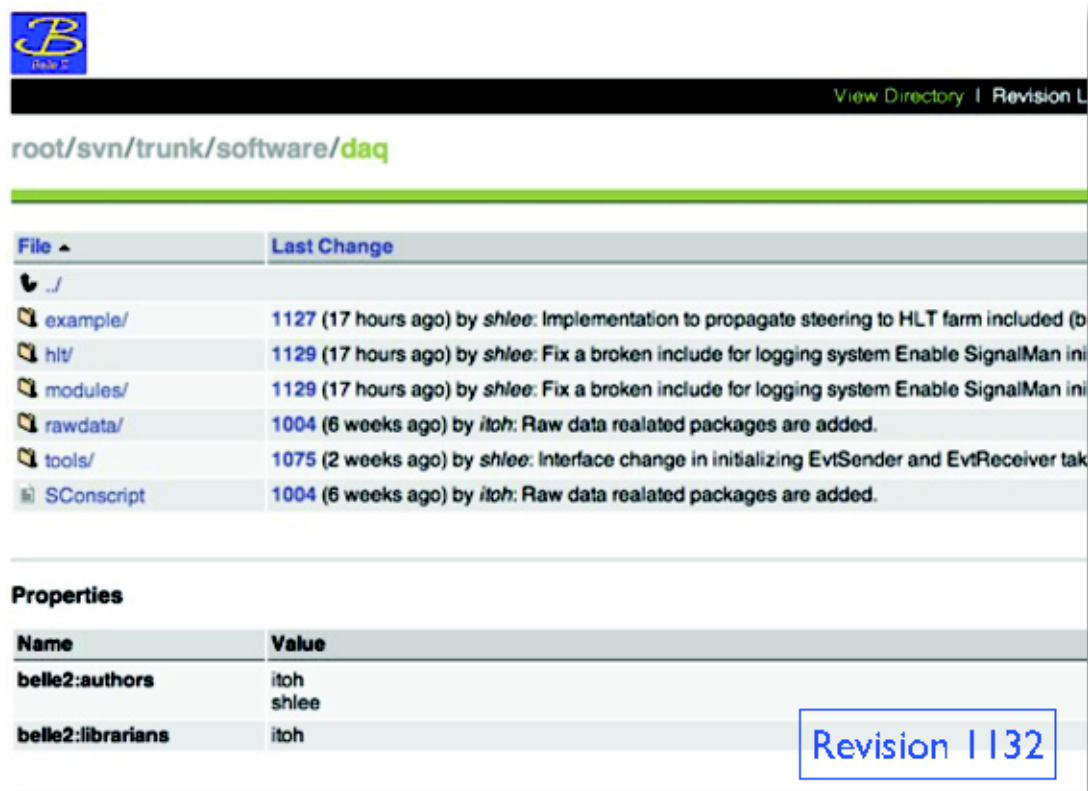
- Part of DAQ package

- daq/hlt: main components
- daq/tools: hbasf2 source code

- Two running modes

- Manager mode: for manager node
- Process mode: for event separator/worker node/event merger

- The implementation is still going on!



View Directory | Revision L

root/svn/trunk/software/daq

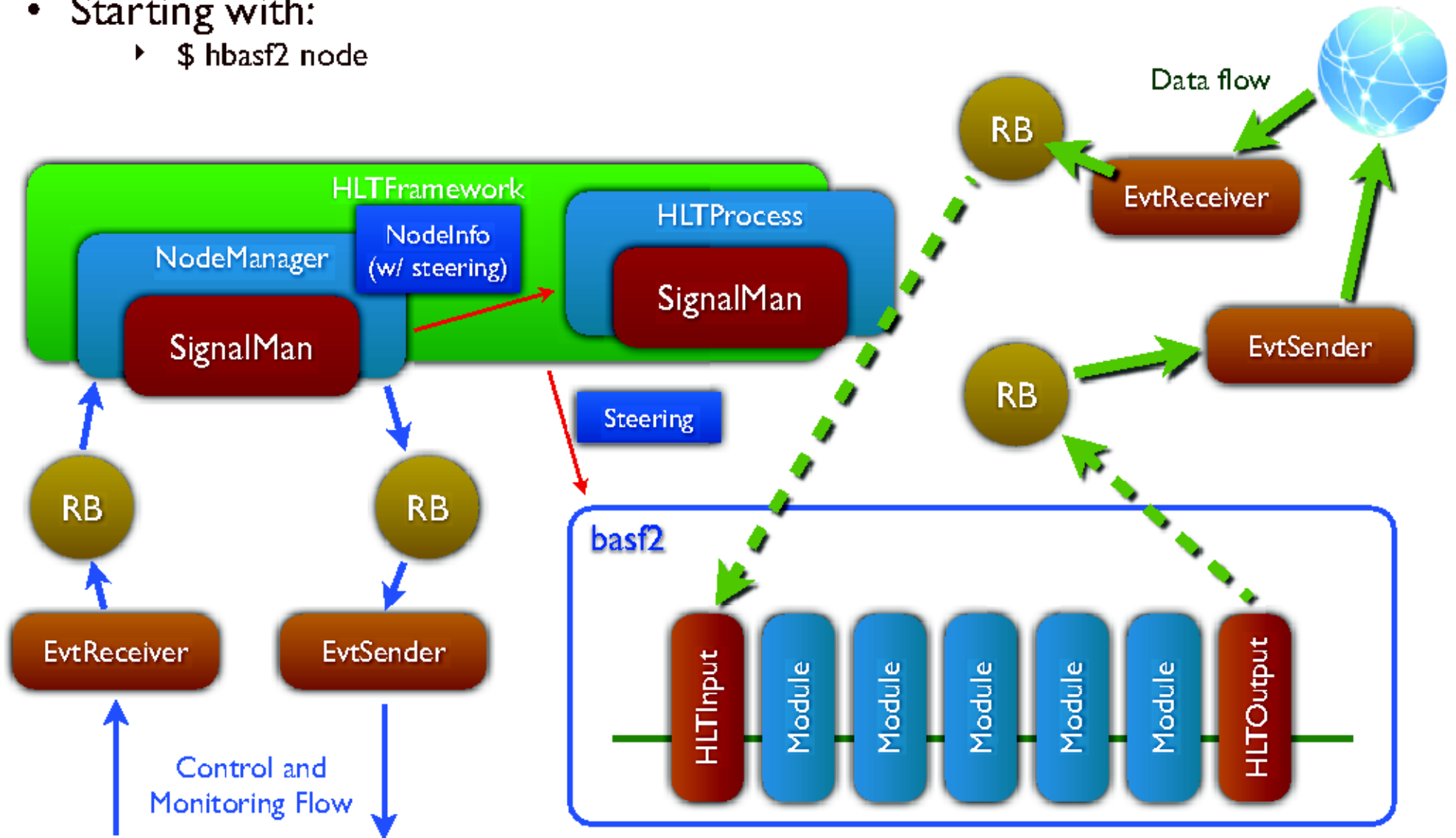
File ^	Last Change
./	
example/	1127 (17 hours ago) by shlee: Implementation to propagate steering to HLT farm included (b
hlt/	1129 (17 hours ago) by shlee: Fix a broken include for logging system Enable SignalMan ini
modules/	1129 (17 hours ago) by shlee: Fix a broken include for logging system Enable SignalMan ini
rawdata/	1004 (6 weeks ago) by itoh: Raw data realated packages are added.
tools/	1075 (2 weeks ago) by shlee: Interface change in initializing EvtSender and EvtReceiver tak
SConscript	1004 (6 weeks ago) by itoh: Raw data realated packages are added.

**Properties**

Name	Value
belle2:authors	itoh shlee
belle2:librarians	itoh

Revision 1132

- Starting with:
  - \$ hbasf2 node



# Summary

- The development of the data flow in Belle II DAQ is going on.
- The data processing are supposed to be performed on COPPERs, Readout PCs, and High Level Trigger nodes utilizing the same software framework “basf2” as that in offline.
- The event data are treated as “ROOT objects” and passed over the network using UNIX sockets by streaming them.
- Development of software components on each platform is going on, which includes “basf2 on COPPER” and “event building using basf2”.
- A new HLT test bench consisting of “multicore” servers is being built based on the “small switch” strategy.
- The preliminary performance test of a 32 core server has been done and a promising result is obtained.

# Backup Slides

```
struct RawCDC
{
    struct RawHeader header[16];
    unsigned int data[];
}
```

```
struct RawHeader
{
    unsigned int IDENT
    unsigned int nword;
    unsigned int .....
}
```

# Estimation of reduction rate by HLT

## Experience at Belle

- Two level reduction

a) “Level 4” selection (= Conventional HLT trigger software)

\* Cut in event vertex obtained using fast tracking

\* Cut in total energy sum of calorimeter

- Reduction rate is dependent on the beam condition

Typical reduction factor ~ 50% (2006 beam condition)

b) “Physics skim”

\* Physics level event selection using full reconstruction results.

\* Almost 100% of physics analysis use so-called “hadronBJ” and “low multiplicity skims + some scaled monitor events.

HadronBJ	: 14.2%	
Low mult. ( $\tau\tau$ , 2photon)	: 9.6%	2004 experience
Monitor events (ee, $\mu\mu$ ...)	: ~1%	
Total	: ~25% of L4 passed events	

➔ Order of 1/10 reduction at HLT is possible in some cases.

Assume 1/5 reduction as a realistic number.



- “Level 4” filter should be useful at Belle II.
- Based on fast tracking using CDC hits only.
- Close relationship with CDC trigger.
  - => Asked Iwasaki-san to take care of it.