

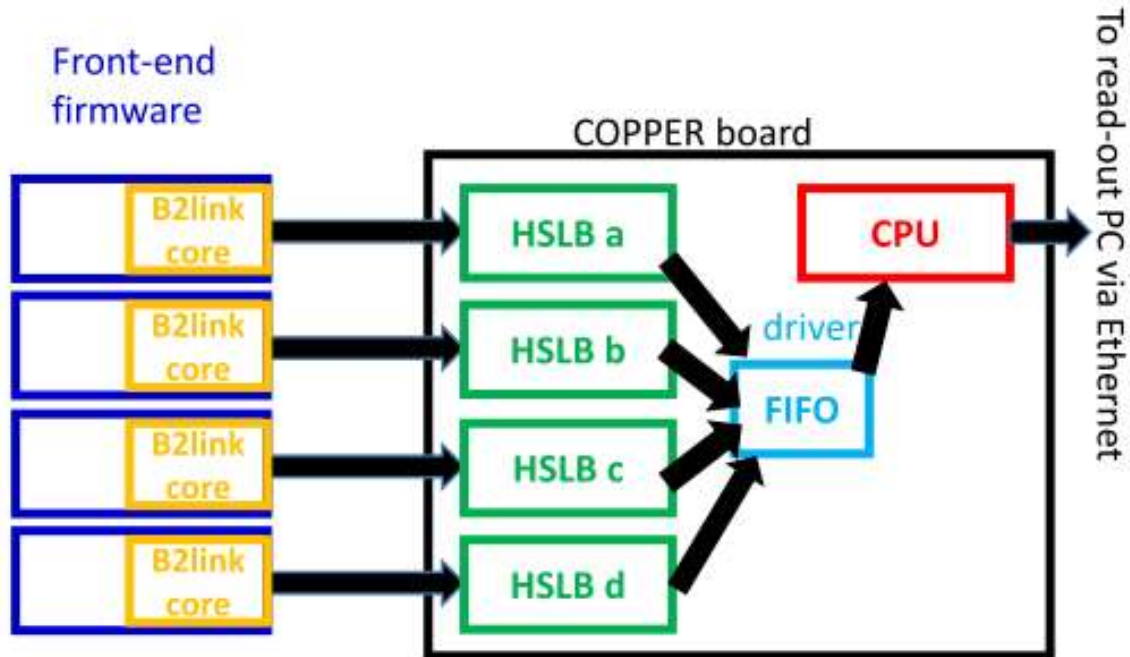
RawCOPPER data format

Aug. 27, 2015 (svn rev. 17269)

Satoru Yamada

1, Overview of RawCOPPER format (one data block from a COPPER board)

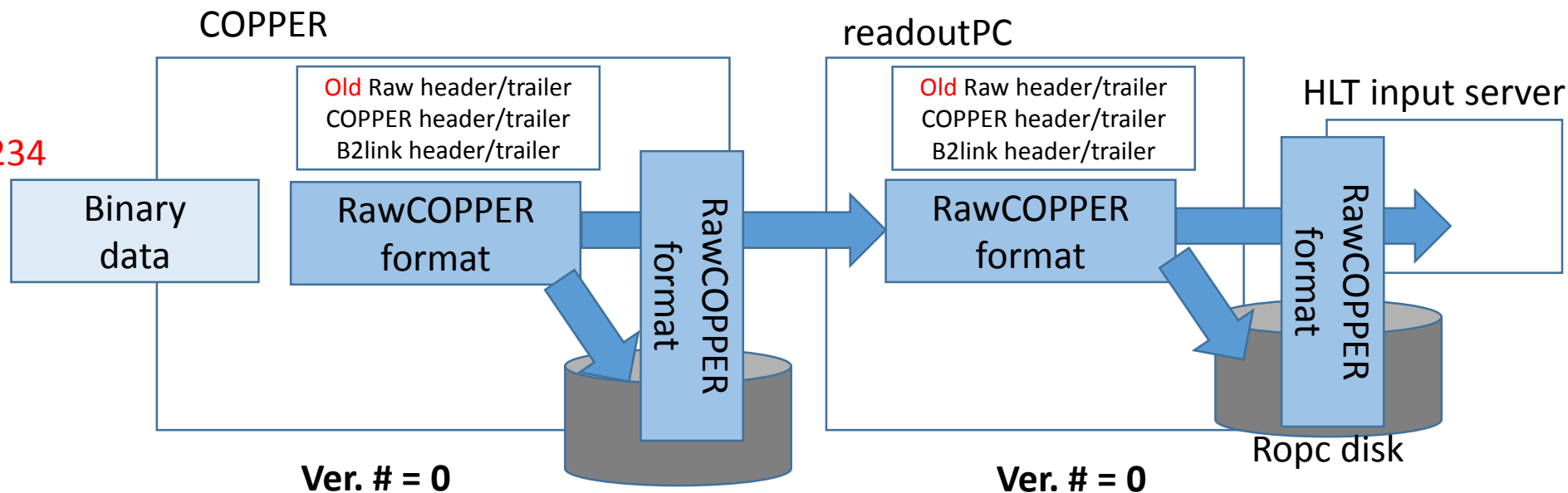
RawCOPPER header/trailer -> See Sec. 2
COPPER header/trailer -> See Sec.3
B2link(FEE+HSLB) header/trailer -> See Sec.4
Detector buffer -> Untouched by DAQ



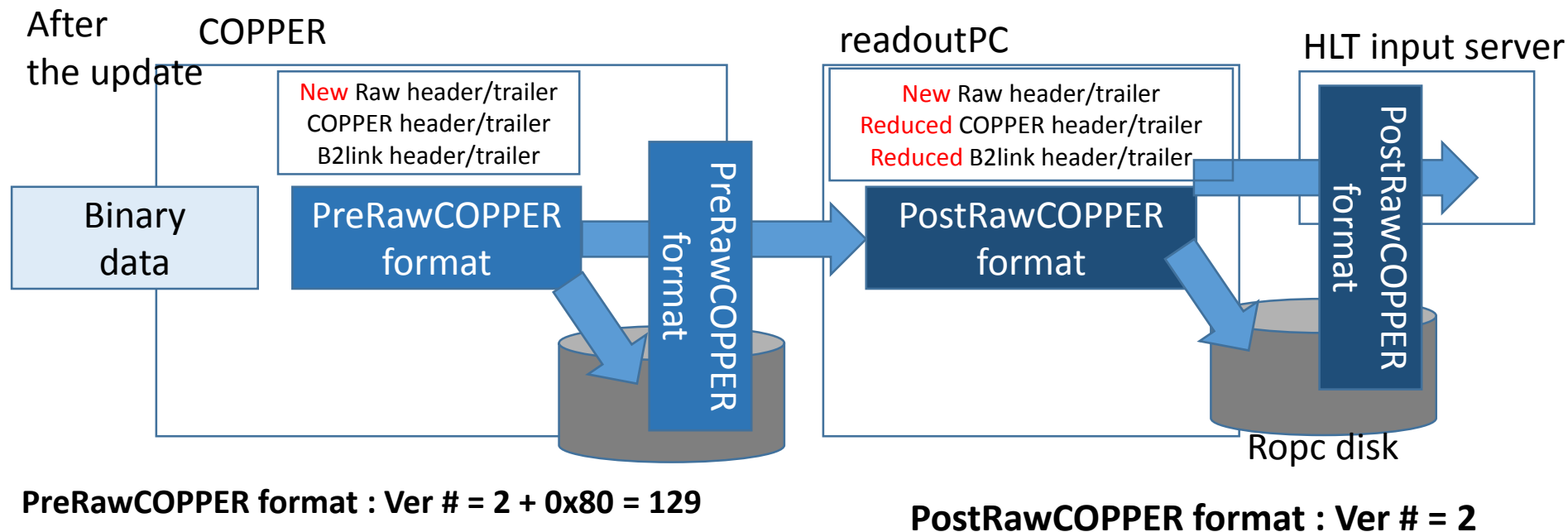
- **RawCOPPER header**
 - **COPPER header**
 - B2link HSLB header (slot A FINNESSE)
 - B2link FEE header(slot A FINNESSE)
 - Data contents(Detector buffer) (slot A FINNESSE)
 - B2link FEE trailer (slot A FINNESSE)
 - B2link HSLB trailer (slot A FINNESSE)
 - B2link HSLB header (slot B FINNESSE)
 - B2link FEE header(slot B FINNESSE)
 - Data contents(Detector buffer) (slot B FINNESSE)
 - B2link FEE trailer (slot B FINNESSE)
 - B2link HSLB trailer (slot B FINNESSE)
 - B2link HSLB header (slot C FINNESSE)
 - B2link FEE header(slot C FINNESSE)
 - Data contents(Detector buffer) (slot C FINNESSE)
 - B2link FEE trailer (slot C FINNESSE)
 - B2link HSLB trailer (slot C FINNESSE)
 - B2link HSLB header (slot D FINNESSE)
 - B2link FEE header(slot D FINNESSE)
 - Data contents(Detector buffer) (slot D FINNESSE)
 - B2link FEE trailer (slot D FINNESSE)
 - B2link HSLB trailer (slot D FINNESSE)
 - **COPPER trailer**
- **RawCOPPER trailer**

1-1, Online header/trailer reduction

Before rev. 11234
(June, 2014)



After rev. 11234
(June, 2014)



- PreRawCOPPER format
 - If you store data by COPPER CPU, then output data will be in Pre(reduction)RawCOPPER format.
- PostRawCOPPER format
 - Store the data downstream from readout PC, the output data will be in Post(reduction)RawCOPPERFormat

2-1, “RawCOPPER header/trailer” format in PreRawCOPPER format (ver. 2+0x80)

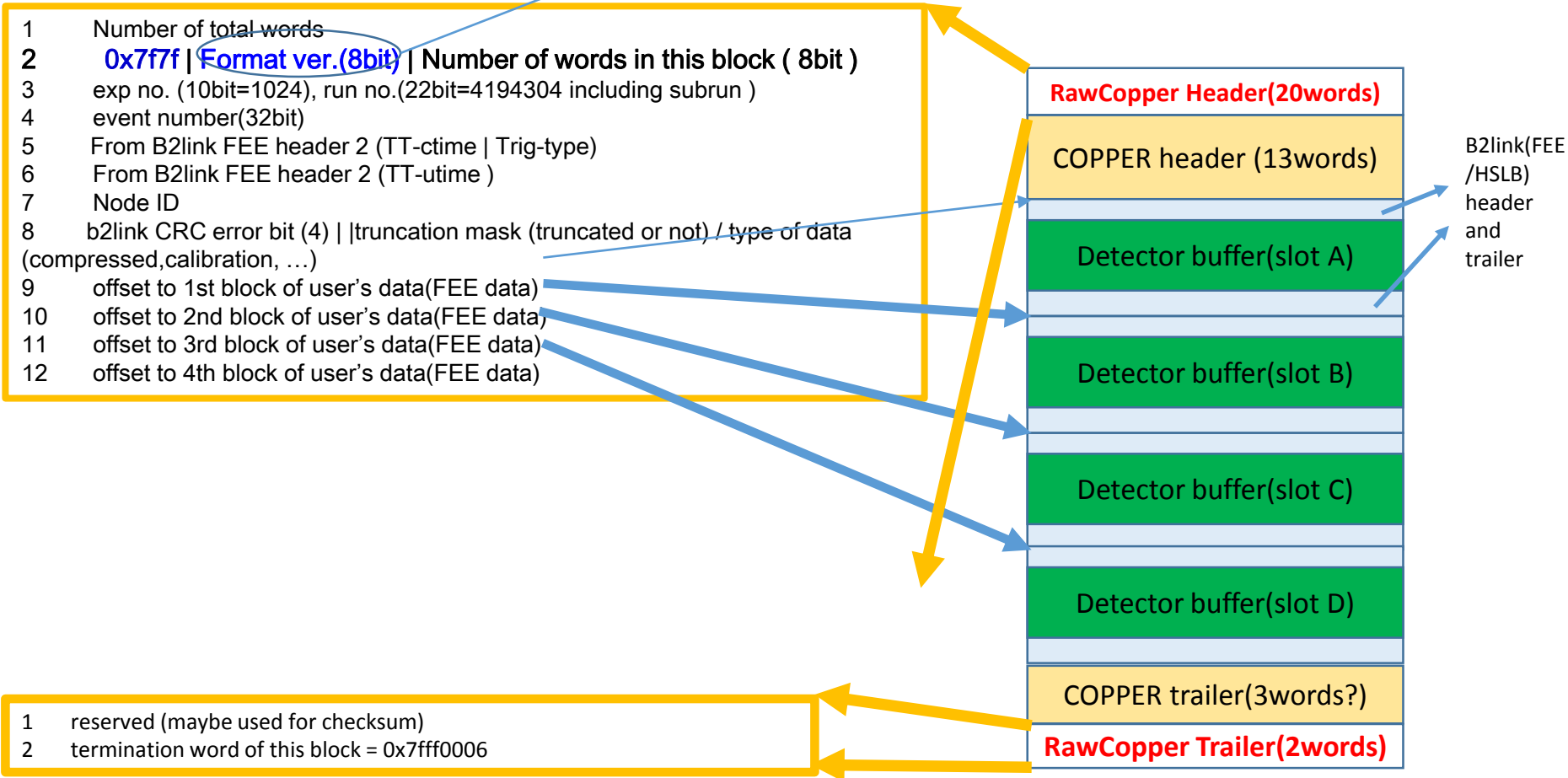
* Same as ver.1

Use this version number to distinguish
Different data format.

Ver.0 : to 2014. June(including DESY test)

ver.1 : from June.2014

Ver.2 : from Apr. 2015 (rev.17269)



2-2, “RawCOPPER header” and trailer format in PostRawCOPPER format (ver.0x02)

* Same as ver.1

Same as PreRawCOPPER format

2-3, tentative format of 32bit node ID

Format : (31-24) Detector ID : 8bit=256 : detector ID
(9-0) lower bits of COPPER ID : 10bit (1024)

Detector ID :

Detector ID (Defined in rawdata/dataobjects/include/RawCOPPERFormat.h)

- #define SVD_ID 0x01000000 // tentative
- #define CDC_ID 0x02000000 // tentative
- #define BPID_ID 0x03000000 // tentative
- #define EPID_ID 0x04000000 // tentative
- #define BECL_ID 0x05000000 // tentative
- #define EECL_ID 0x06000000 // tentative
- #define BKLM_ID 0x07000000 // tentative
- #define EKLM_ID 0x08000000 // tentative
- #define TRGDATA_ID 0x10000000 // tentative
 - #define CDCTRDATA_ID 0x11000000 // tentative
 - #define ECLTRGDATA_ID 0x12000000 // tentative
 - #define TOPTRGDATA_ID 0x13000000 // tentative
 - #define KLMTRGDATA_ID 0x14000000 // tentative
 - #define GDLTRGDATA_ID 0x15000000 // tentative

Full COPPER ID :

- **Except for TRG**, full COPPER ID can be reconstructed by “(Detector ID >> 24) * 1000 + COPEPR ID(12bit) ”
e.g. NodeID = 0x0600000a -> COPPER ID = cpr6010
NodeID = 0x0100000a -> COPPER ID = cpr1010
- ~~For TRG data, full COPPER ID can be reconstructed by “ 9000 + COPEPR ID(12bit) ”~~
 - Following a request from SLC part, hostname of TRG coppers will be assigned as follows: (Aug.27, 2015)
CDC TRG : cpr1100*
ECL TRG : cpr1200*
...
GDLTRG : cpr1500*

A label of COPPER ID will be attached on the front of a COPPER board

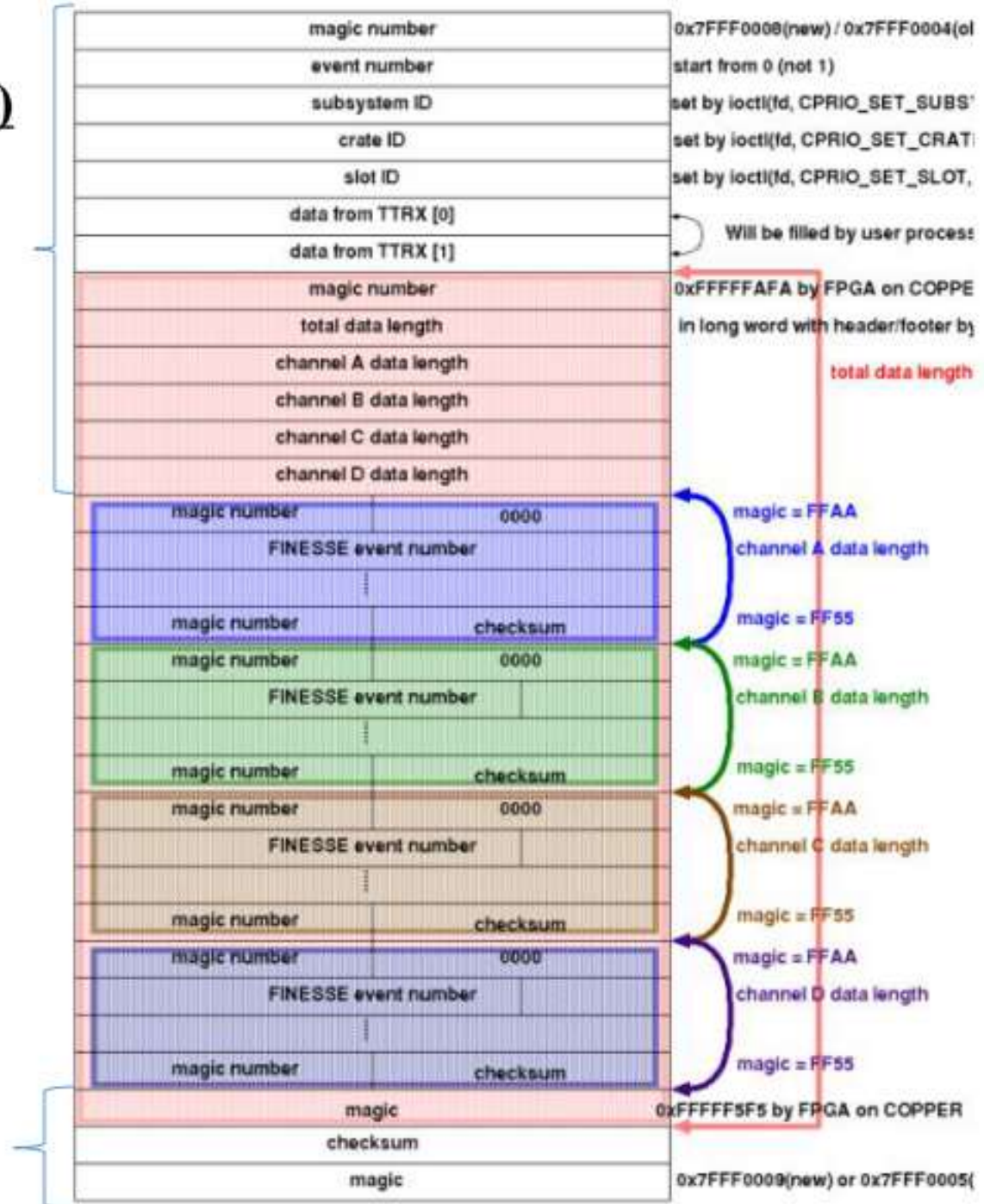
Node ID = “TTD ” = 0x54544420 and “FTSW” are reserved for VME CPU and FTSW now.

3-1, COPPER header and trailer in **PreRawCOPPER** format (ver. 2 + 0x80)

* Same as ver.1

COPPER header

COPPER Trailer



3-2, COPPER header and trailer in PostRawCOPPER format (ver. 0x02)

* Same as ver.1

No COPEPR header and trailer in Post reduction rawcopper format.

4-1, B2link FEE header/Trailer, B2link HSLB header/Trailer in PreRawCOPPERFormat (ver. 0x02 + 0x80)

* Same as ver.1

From Nakao-san's Belle2link User guide (June 10, 2014):

You can download from 18 th B2GM indico page

<http://kds.kek.jp/getFile.py/access?contribId=132&sessionId=28&resId=0&materialId=0&confId=15329>

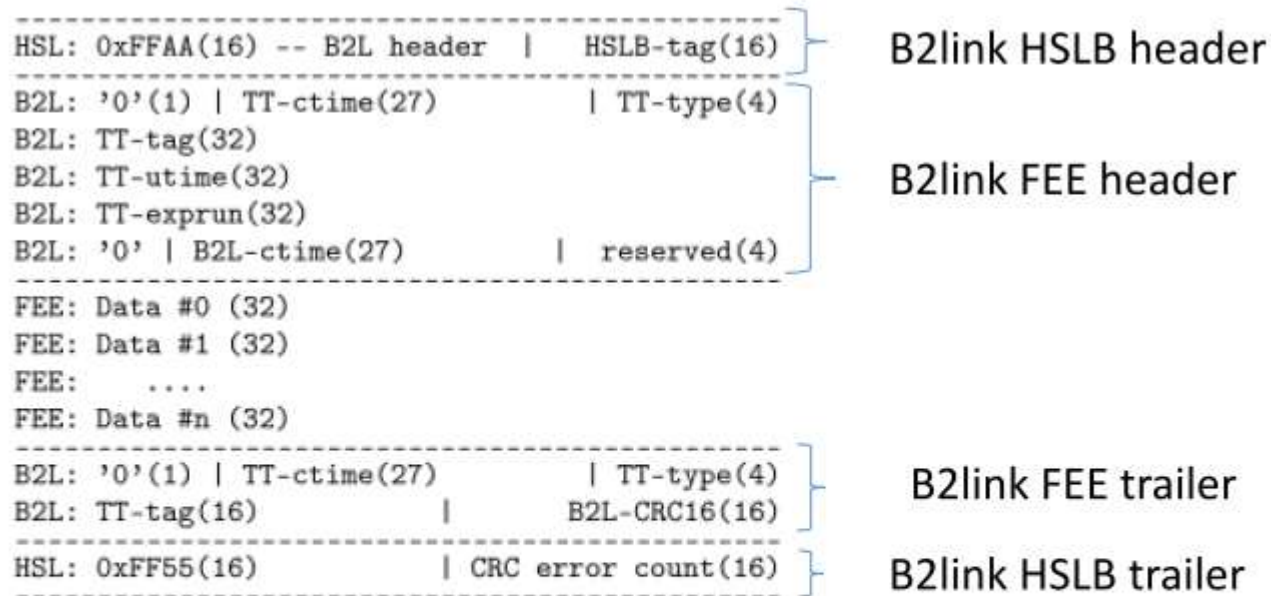


Figure 5: Data format as read out by the COPPER. The header and trailer words labelled with **HSL** are attached by HSLB, the words with **B2L** are attached by the belle2link component, and the words with FEE are those written into the belle2link component by the frontend firmware.

NOTICE :

To produce this format, the b2tt core used in the FEE firmware should be the latest.

Please see Nakao-san's following e-mails :

[b2link_ml:0143] Belle2link version 0.01 - SVN update

And

[b2link_ml:0144] Re: Belle2link version 0.01 - SVN update .

4-2, B2link FEE header/Trailer, B2link HSLB header/Trailer in PostRawCOPPERFormat (ver. 0x02)

*** Modified from ver.1**

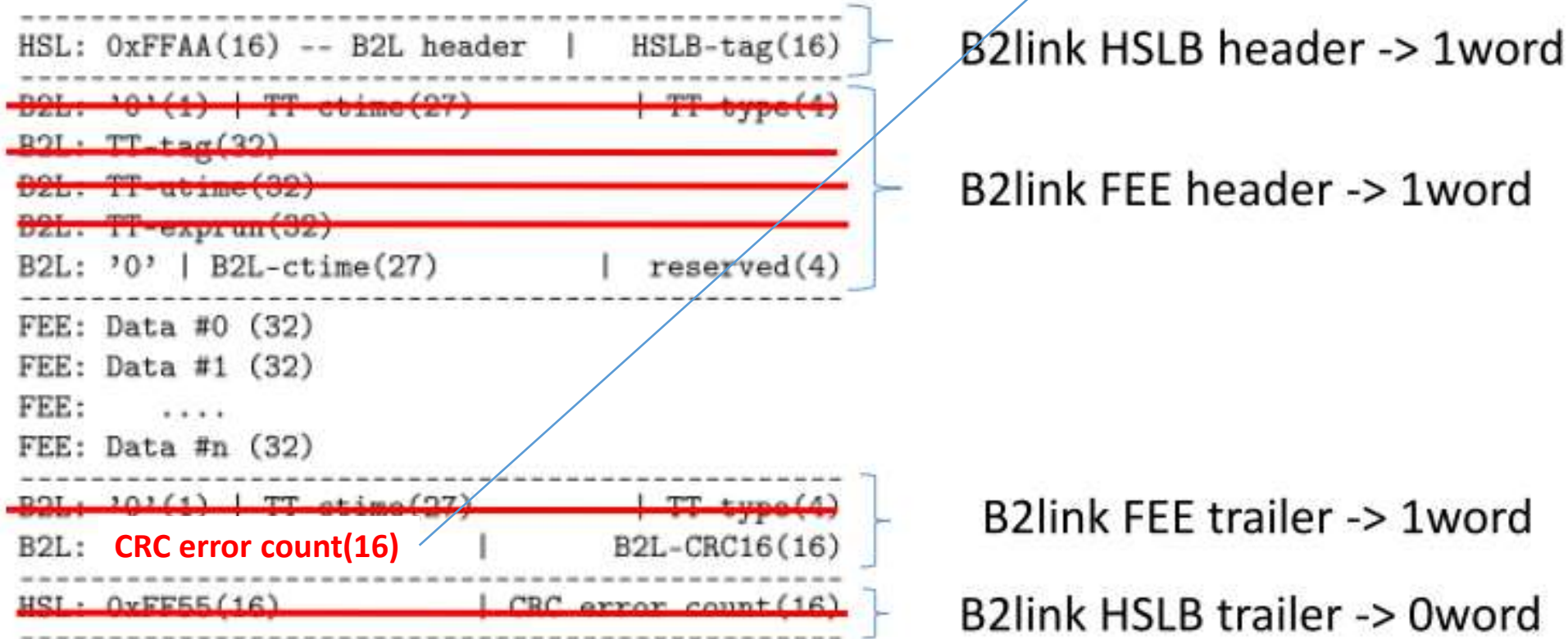


Figure 5: Data format as read out by the COPPER. The header and trailer words labelled with **HSL** are attached by HSLB, the words with **B2L** are attached by the belle2link component, and the words with FEE are those written into the belle2link component by the frontend firmware.

4-3, Older B2link header/trailer formats

At DESY test in January of 2014

From Nakao-san's B2GM slides:

<http://kds.kek.jp/getFile.py/access?contribId=143&sessionId=38&resId=0&materialId=slides&onfId=13911>

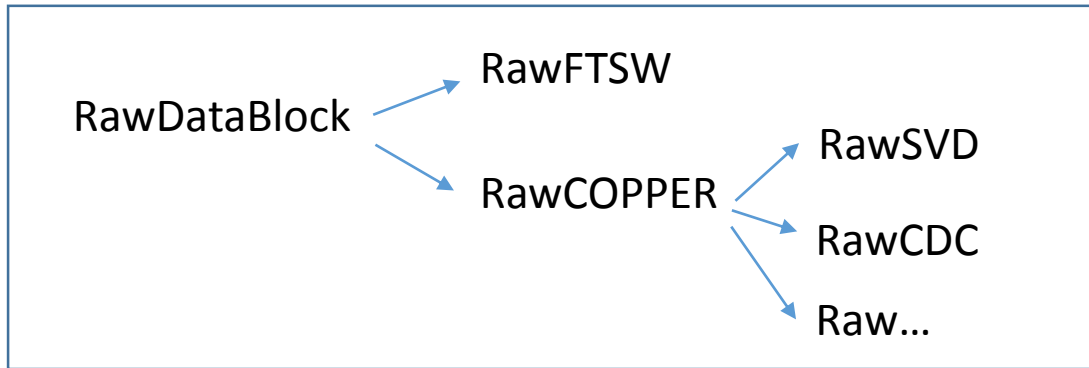
Data format (Final?)

The format used at the telescope test

```
-----  
HSL: 0xFFAA(16) --- B2L header | HSLB-tag(16)  
-----  
B2L: '0'(1) | TT-ctime(27) | TT-type(4)  
B2L: TT-tag(32)  
B2L: TT-utime(32)  
B2L: TT-exprun(32)  
B2L: '0' | B2L-ctime(27) | debug-flag(4)  
-----  
FEE: Data #0 (32)  
FEE: Data #1 (32)  
FEE: ....  
FEE: Data #n (32)  
-----  
B2L: TT-tag(16) | B2L-checksum(16)  
-----  
HSL: 0xFF55(16) | HSLB checksum(16)  
-----
```

- tag (event number) and utime to be increased to 32-bit (done),
HSLB-checksum, B2L-checksum to be added

5-1, RawDataBlock object (to handle Raw data from COPPER board)



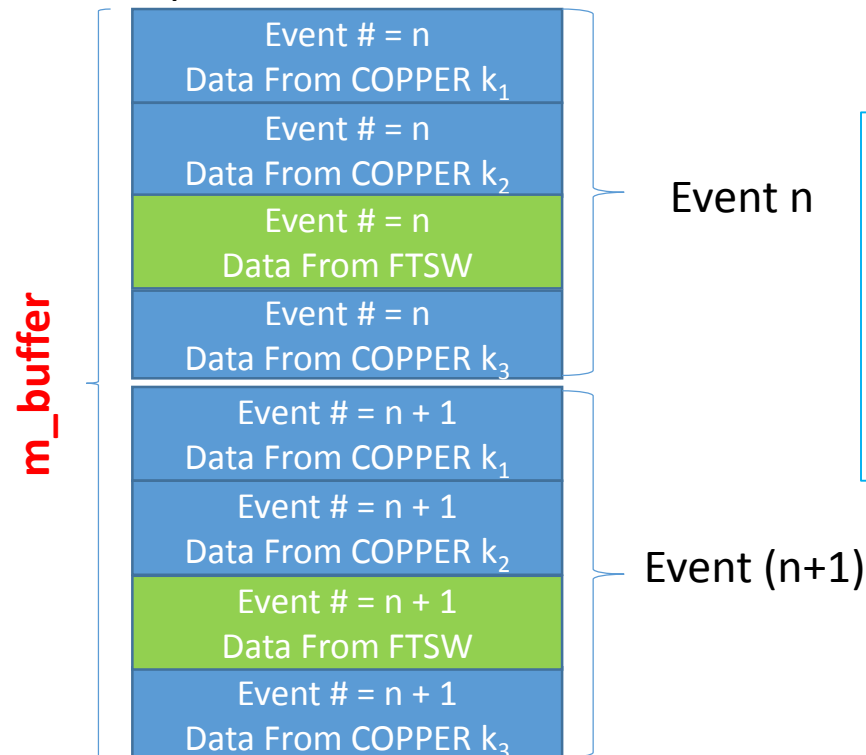
Source code :

<https://belle2.cc.kek.jp/svn/trunk/software/rawdata/dataobjects/>

```
RawDataBlock{
    methods to access data;
    int m_num_nodes; // # of nodes
    int m_num_events; // # of events

    int* m_buffer; -> buffer for data
}
```

Example of data structure

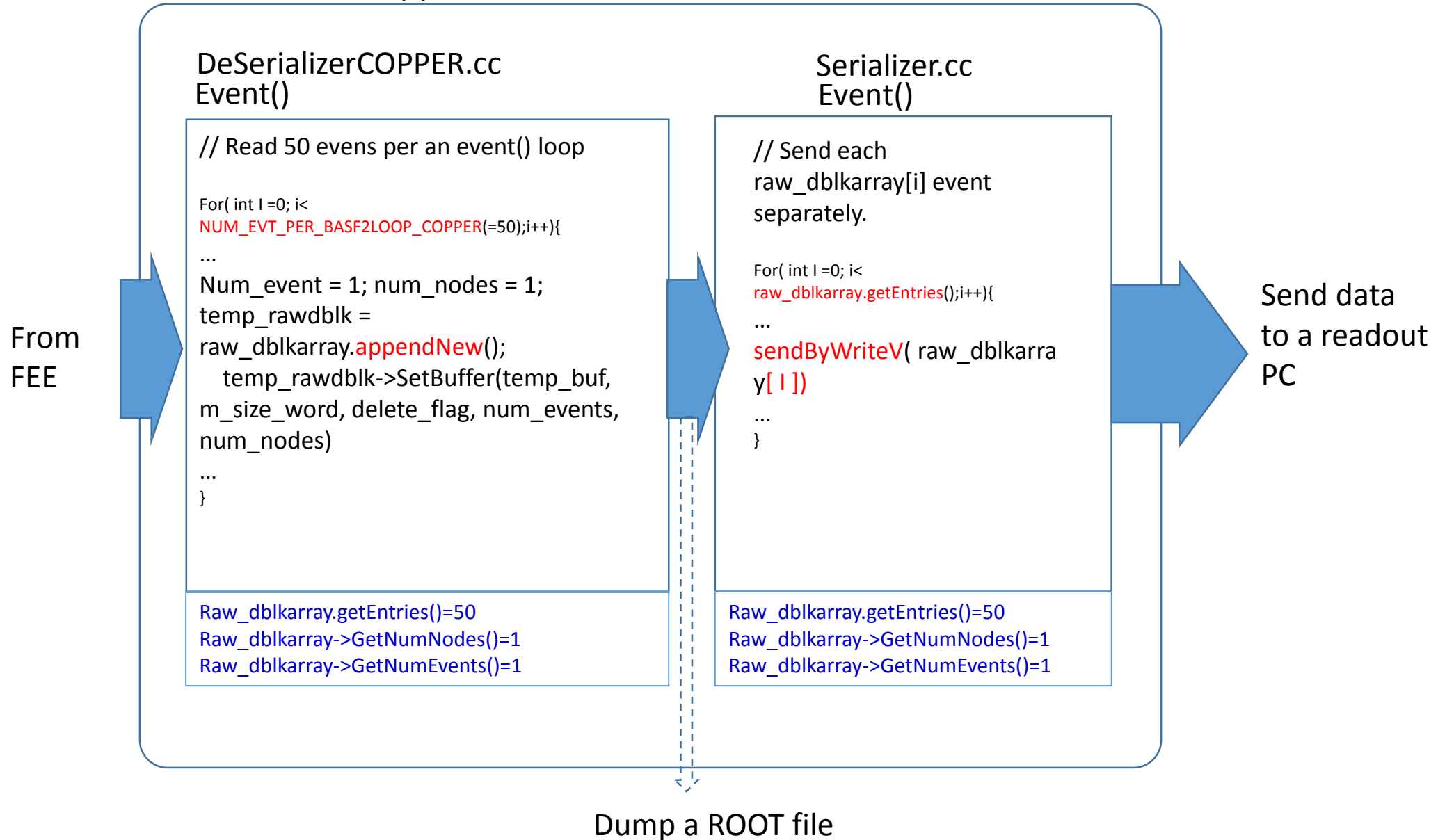


In this example,
 $M_num_nodes = 4$
 $M_num_events = 2$.

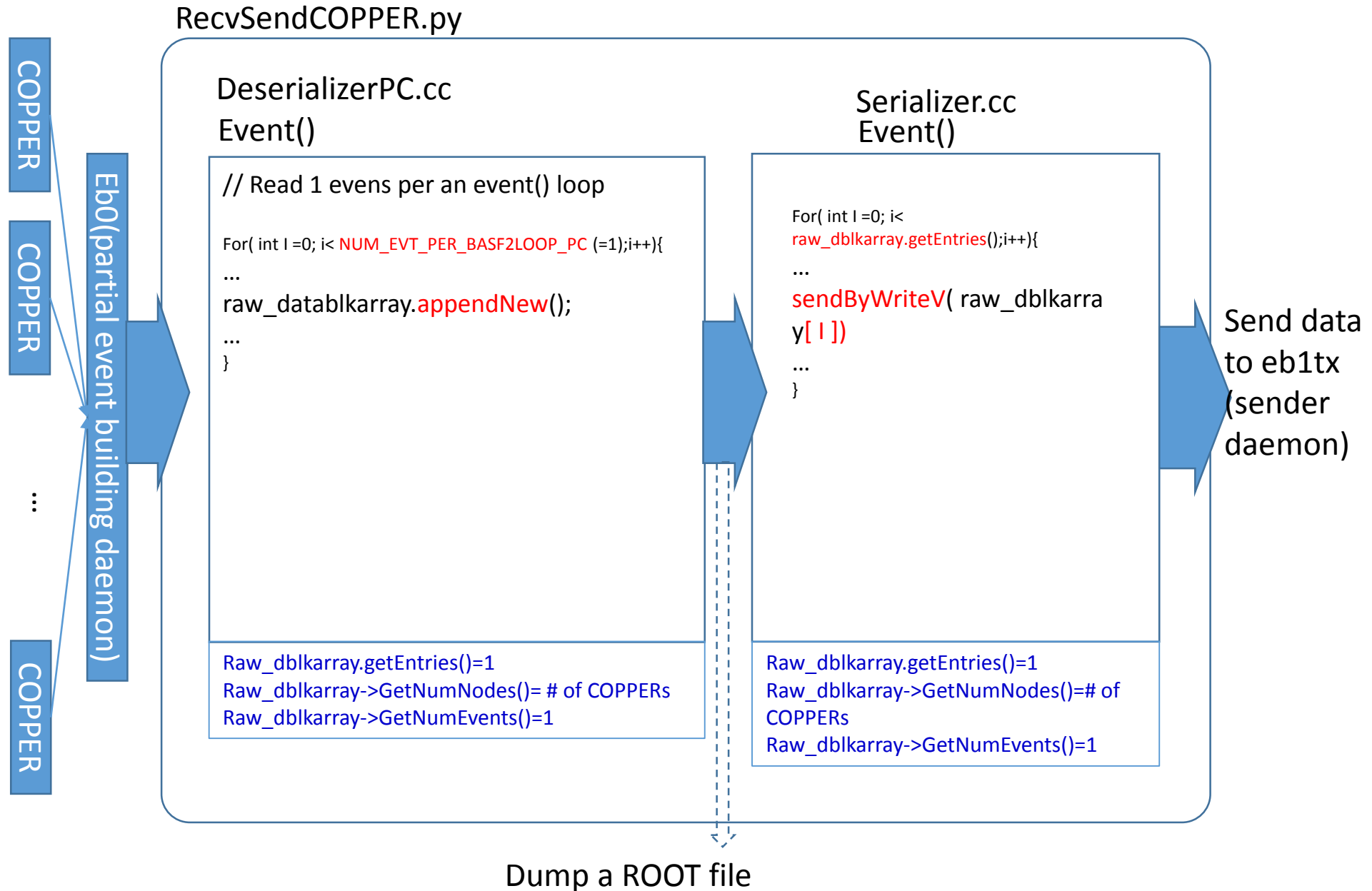
of data blocks = $4 * 2 = 8$

Example of Data handling on COPPER (as of rev.12453)

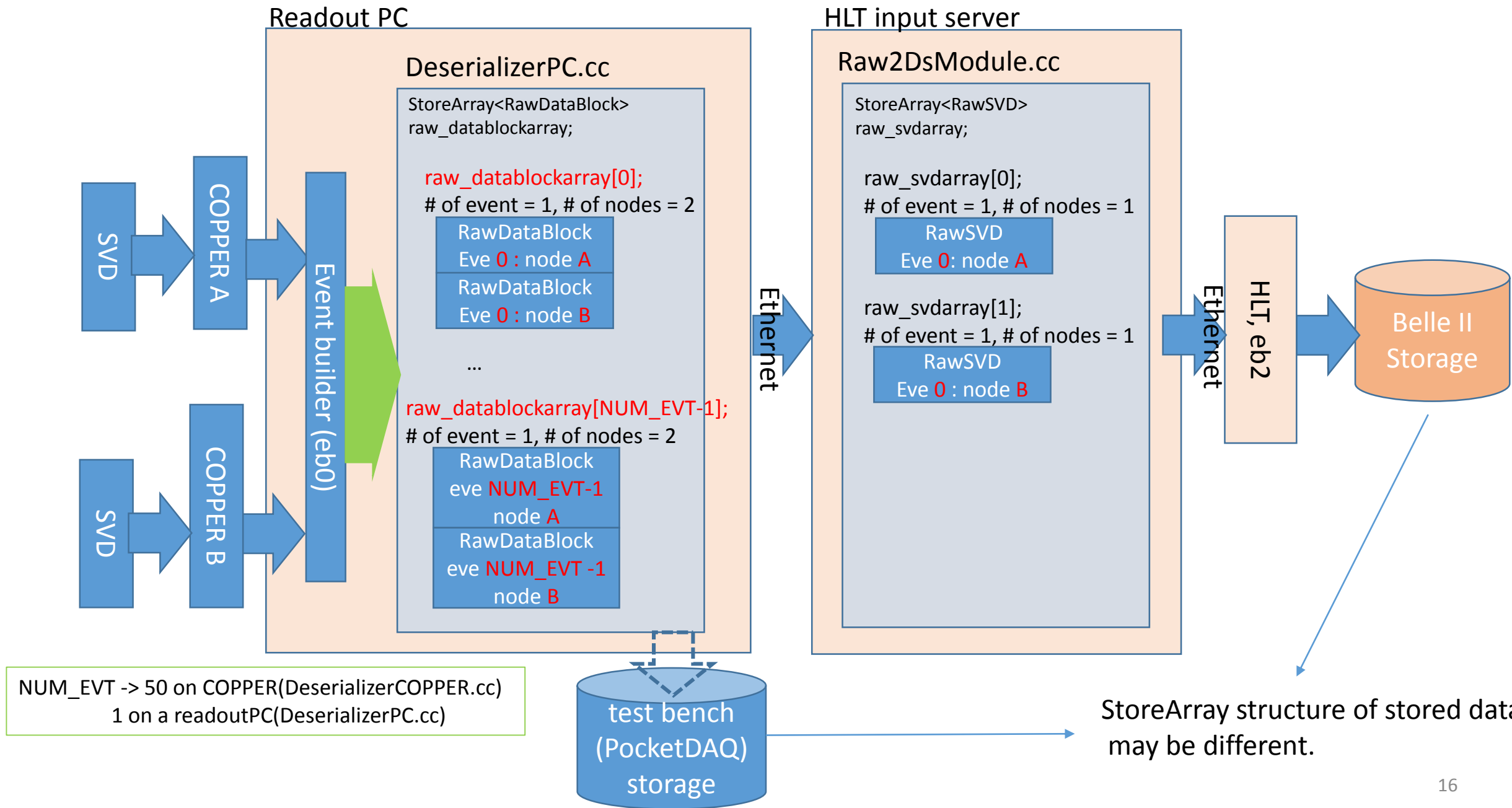
RecvSendCOPPER.py



Example of Data handling on a readout PC (as of rev.12453)



Example : # of events and nodes in one RawDataBock(Detector) object and StoreArray :



Example : how to store rawdata in RawDetector object by DAQ program

- Full Belle II DAQ
 - HLT(High Level Trigger) receives serialized binary data from readout PCs and stores them in RawDetector Class. This RawDetector objects will be stored in storage.
 - Module : `daq/rfarm/event/modules/src/Raw2DsModule.cc`
- Pocket DAQ
 - DATA are stored as a RawDataBlock object
 - Example program can be used to convert RawDataBlock/COPPER to RawDetector objects
 - Module : `daq/rawdata/modules/src/Convert2RawDet.cc`

2-2, Rawdata Unpacker for new and old data formats

Data taken at the DESY beam test(old format) can be read with the latest rawdata package
-> by checking data ver. In header.

New RawCOPPER class

- No change in style of the member functions -> No effect on derived class
- Does not have a format information in itself
 - Format class contains format information
 - RawCOPPERformat.cc -> the latest format
 - RawHeader.cc
 - RawCOPPERformat_v0.cc -> an old format
 - RawHeader_v0.cc
 - Assign a format class to `m_access` in `CheckVersionSetBuffer()`
 - Use `m_access` to access buffer contents

```
inline int RawCOPPER::GetExpNo(int n)
{
    CheckVersionSetBuffer();
    return m_access->GetExpNo(n);
}
```

```
inline int RawCOPPER::GetRunNo(int n)
{
    CheckVersionSetBuffer();
    return m_access->GetRunNo(n);
}
```

Notice :

- RawCOPPER class supports both formats for a while (0.5-1 year after the format becomes stable?).
- In that case, the latest RawCOPPER class cannot be used to read old format
- Of course, you can use old rawdata repository to read old format
- For ver.0 format, use rawdata repository before 11228

Revision History of this document(1)

- Jan.5, 2014 rev. 8376 : Add definition of tentative subsysID format
- Dec. 16, 2013 rev.7974 :
 - Add B2linkFEE header format
 - Add comments about handling StoreArray when unpacking Raw*** data.
- Oct.21, 2013 :rev.7133
 - Add instruction about Rawdata unpacking program
- Oct. 18, 2013 :rev. 7095
 - 1 st draft
- Jun. 23, 2014 : rev. 11234
 - Online (header/trailer) reduction scheme on readout PC is introduced
 - RawHeader format is changed
 - COPPER header/trailer format is changed
 - Nakao-san updated B2LFEE/HSLB header/trailer format
 - See [b2link_ml:0144] Re: Belle2link version 0.01 - SVN update
- Aug. 23, 2014: rev. 12453
 - Add a description of how RawDataBLock objects are handled by the actual DAQ program.
- Sep. 26, 2014 : rev.13065
 - Add TRG ID definition (0x09000000)
- Oct. 24, 2014 : rev. 13460
 - Add a slide about # of events and nodes in one RawDataBock(Detector) object and StoreArray and how to store raw data in RawDetector object by DAQ program (p.16. p.17)
- Jan. 23, 2015: rev. 15030
 - Modify TRGDATA_ID and add ***TRGDATA_ID for trigger from sub-detectors
- Mar.3, 2015 : rev. 15988
 - Use “node ID” instead of subsystem ID.

Revision History of this document(2)

- Apr. 21, 2015 rev. **17269** : Description about a new data format (ver.2)
- Aug. 27, 2015 : Hostname of TRG COPPER are revised.

Backup

Modified part of data-format in the change from ver.1 to ver.2

4-2, B2link FEE header/Trailer, B2link HSLB header/Trailer in PostRawCOPPERFormat (ver. 0x01)



Figure 5: Data format as read out by the COPPER. The header and trailer words labelled with **HSL** are attached by HSLB, the words with **B2L** are attached by the belle2link component, and the words with FEE are those written into the belle2link component by the frontend firmware.