Information in EventMetaData and ‘Error event’

1, Storing data in EventMetaData object by online DAQ
2, Definition of ‘Error event’ and how to avoid the event
2-1, Pocket DAQ case

S. Yamada (KEK)

Jul. 1, 2016

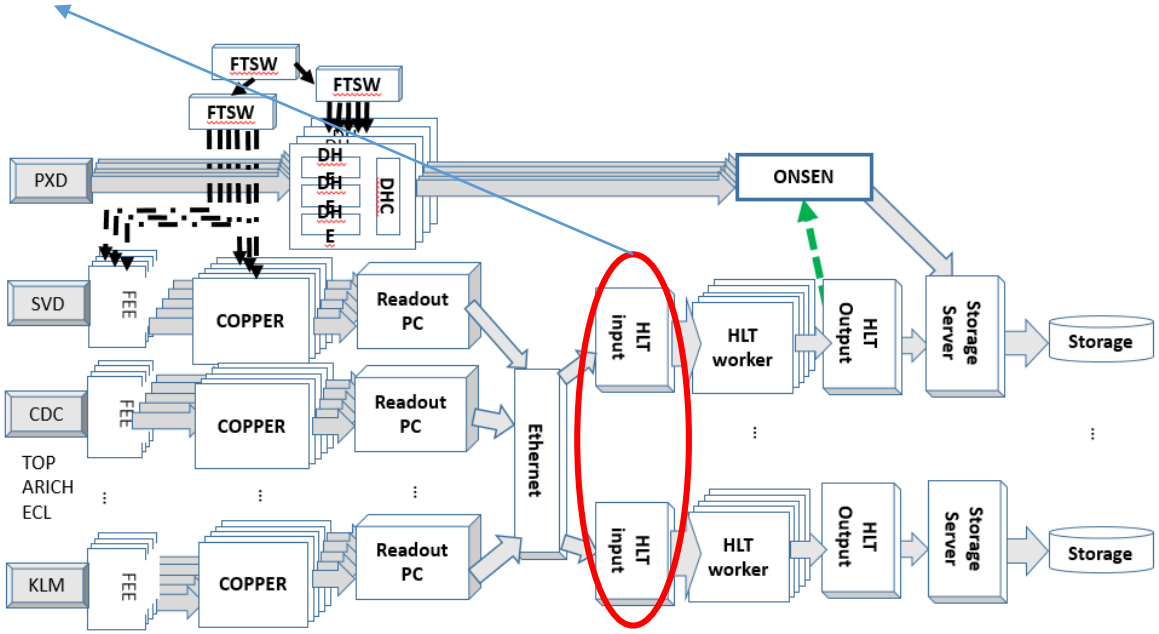
1, Storing data in EventMetaData object by online DAQ

Member variables of EventMetaData

Rev. 28703 (framework/dataobjects/include/EventMetaData.h)

private:
unsigned int m_event; /**< Event number ('normal' data has values > 0). */
int m_run; /**< Run number (usually > 0, run-independent MC has run == 0). */
int m_subrun; /**< Sub-run number, increases indicate recovery from DAQ-internal trouble without change to detector constants. Not supposed to be used by offline analysis. */
int m_experiment; /**< Experiment number. (valid values: [0, 1023], run-independent MC has exp == 0) */
int m_production; /**< Unique identifier of the production of the event. */
unsigned long long int m_time; /**< Time in ns since epoch (1970-01-01). */
std::string m_parentLfn; /**< LFN of the parent file */
double m_generatedWeight; /**< Generated weight. */
unsigned int m_errorFlag; /**< Indicator of error conditions during data taking, ORed combination of EventErrorFlag values. */

Red variables are supposed to be stored in EventMetadata by Raw2DsModule.cc of online DAQ program at an HLT input server. Currently m_time and m_production are not filled yet.



What daq/rfarm/event/modules/src/Raw2DsModule.cc is doing :

Fill values in each variable as follows :

```
evtmetadata->setExperiment(sndhdr.GetExpNum()); // -> Added at rev. 8338
evtmetadata->setRun(sndhdr.GetRunNum()); // -> Added at rev. 8701
evtmetadata->setSubrun(sndhdr.GetSubRunNum());   // -> Added at rev. 28343
evtmetadata->setEvent(sndhdr.GetEventNumber());  // -> Added at rev. 8338
if (error_flag) evtmetadata->addErrorFlag(EventMetaData::c_B2LinkCRCError); // Added at rev.18569
```

Pocket DAQ case :

- There is no HLT input server and filling EventMetaData is done by DeSerializerPC.cc not Raw2DsModule.cc.
- EventMetaData->getErrorFlag() can work if you use the DAQ program for data-taking after rev. 29676.

2, Definition of ‘Error event’ and how to avoid the event

Type of errors

'Error event' in this slides means that its EventMetaData.m_errorFlag is nonzero.

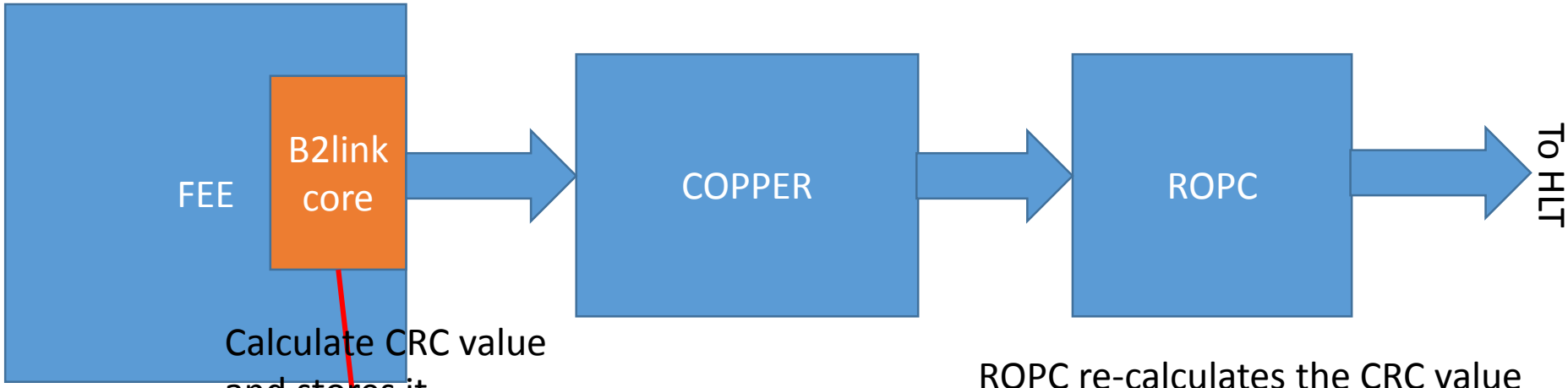
In EventMetaData.h,

```
/** bit-flag format of m_error_flag  */
enum EventErrorFlag {
  c_B2LinkCRCError = 0x1, /**< Belle2link CRC error  is detected in the event */
  c_HLTError       = 0x2  /**< Error is returned from HLT modules. */
};
```

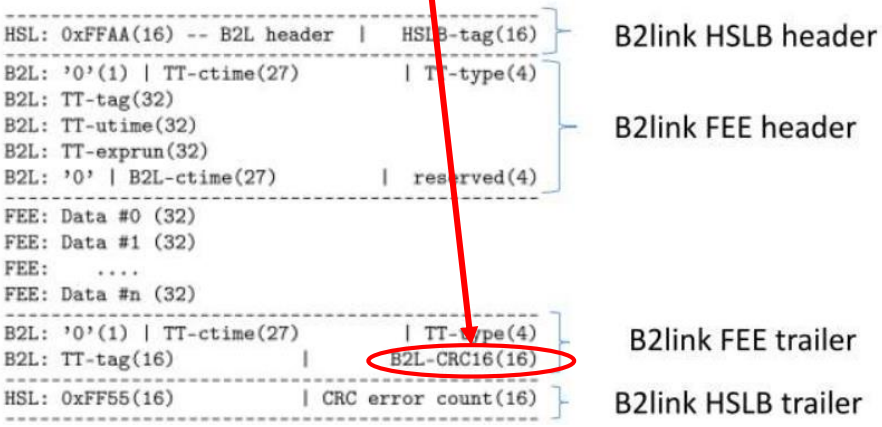
As of Jun. 2016, only c_B2LinkCRCError is implemented if using rev. 18569 or later.

What is ‘m_errorFlag= c_B2LinkCRCError = 0x1’

- Data-corruption during B2link data transfer is detected by CRC check.
  -> m_errorFlag = 1
- Data-corruption during B2link data transfer is not detected by CRC check.
  -> m_errorFlag = 0



Calculate CRC value and stores it.



ROPC re-calculates the CRC value and compare it with the CRC value in a trailer.
If they don’t coincide, error bit in the RawCOPPER header(not EventMetaData) is set.
Later, the header is checked by Raw2DsModule on HLT and m_errorFlag is set to 1.

How to avoid CRC 'error events' :

call getErrorFlag() of EventMetaData:

```
event(){
…
                if( EventMetaData->getErrorFlag() == 0 ){
                // No CRC error
                }else{
                  // CRC error event  }
… }
```

If the ver. of your DAQ program is old: (PocketDAQ before 29676, Belle2DAQ before rev. 18569)

You need to check every Raw*** objects except for RawPXD.

- For data taken with older DAQ program, you need to check each Raw*** object like the following; (*** is COPPER/SVD/CDC/TOP/ARICH/KLM)

```
StoreArray<Raw***> raw_ ***array;
 for (int i = 0; i < raw_ *** array.getEntries(); i++) {
  for (int k = 0; k < raw_ *** array[ i ]->GetNumEntries(); j++) {
   if (raw_***array[ i ]->GetEventCRCError(k) != 0 ){
                // CRC Error event !!
   }
  }
 }
```

Test program to check error flag in EventMetaData in .(s)root files

$ cd ${BELLE2_LOCAL_DIR}/daq/rawdata/examples
$ basf2 CheckErrorEvent.py –i <.sroot file>

…
[INFO] Reading StreamerInfo
[INFO] SeqRootInput: initialized.
[INFO] CheckErrorEvent: initialize() started.
[INFO] CheckErrorEvent: initialize() done.
[INFO] SeqRootInput: beginRun called.
[INFO] Begin of new run
[INFO] Processed: 1 runs, 1 events
[INFO] Processed: 1 runs, 2 events
…
[INFO] Processed: 1 runs, 7000 events
[INFO] Seq File 1 closed
[INFO] SeqRootInput : 7984 events read with total bytes of 113506 kB
[INFO] SeqRootInput : event rate = 10.2758 (KHz)
[INFO] SeqRootInput : flow rate = 146.088 (MB/s)
[INFO] SeqRootInput : event size = 14.2167 +- 1.26828 (kB)
[INFO] SeqRootInput: endRun done.

Event CRC error 0 CPRs 0 Events

-> In this case, No CRC 'error events' stored in EventMetaData are detected.

end