Towards non-stop DAQ

Mikihiko Nakao (KEK)

mikihiko.nakao@kek.jp

2019.8.28
Trigger/DAQ workshop at Yonsei

Auto restart

- **Motivation**
  - Data taking should not rely on diligence of the shifter, computers should be able to react faster
  - Most of the loss is probably due to known problem, which will be fixed by a known procedure: SALS
- **How was in Belle**
  - We did not do auto-restart since chance was bigger to get into the same trouble just by restart
  - A typical recovery procedure was a bit more complex than SALS
  - But shifters heard the alarm sound when the run is stopped by an error

Why not possible at Belle II so far?

- **Who to control**
  - TTD knows most of the errors, but it should be the master RC to take control
- **Difficulty in TTD code**
  - As discussed yesterday, pocket_ttd and ttctrld programs are not well maintained, especially in terms of communication to RC
- **Difficulty in RC code**
  - runcontrold is designed to be the master RC and at the same time the subdetector RC or readoutpc RC
  - Error / restart handling is probably a non-generic code which spoils the generality of runcontrold
  - Keeping the balance between generality and necessity and ease of code maintenance is the key

Things need to be avoided

- **Infinite loop**
  - If auto restart is tried and not successful, it should not be tried again
  - Definition of “success” is not so obvious. Maybe something like running more than 10 minutes with deadtime less than 10%?
- **Race condition**
  - Shifter or daqcore member may want to react upon error and may compete with the master RC who also tries the same

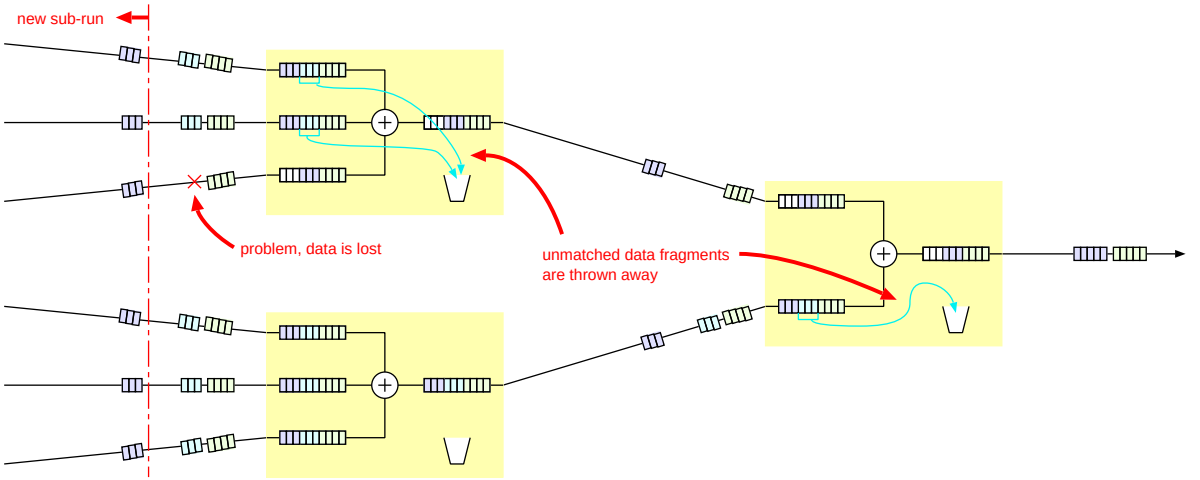
Towards auto-restart

- **ttctrld**
  - should send something to master RC, but RCCommand::STOP is not the right one. Probably need to add RCCommand::ERROR or something?
- **runcontrold**
  - it is a very passive process, and probably not a place to add many conditions such as infinite loop check
  - still some locking mechanism is needed in runcontrold
- **another app?**
  - Probably we need a process to check status of many things and tell master runcontorld what to do next (i.e., replacement of CSS GUI)
  - Who is in charge?

Idea of non-stop DAQ

- **Motivation**
  - Error happens only in one small corner while the rest is healthy
  - It takes some time to restart, and we can save some time if we do not need to restart HLT
  - In Belle, I was working both on TTD and Run Control and it was easy to coordinate RC and TTD, but it was not possible to fix a problem without starting a new run
  - And run restart often caused another problem somewhere else
- **What was vaguely planned**
  - Run can be kept running while recovering the error
  - Pause the run, and resynchronize event building after resumed
  - exp+run+subrun+event (64 bit in total) is always incremented

EB recovery



#1	#2	#3	EB		
Run 638 sub-run 0 - run reset					
0	0	0	0	0	0
1	1	1	1	1	1
2	2	2	2	2	2
3	3	3	3	3	3
4	4	4	4	4	4
5	x	5	(throw away)		
6	x	6	:		
7	x	7	:		
Run 638 sub-run 1 - sub-run reset					
8	8	8	8	8	8
9	9	9	9	9	9
10	10	10	10	10	10
:	:	:	:	:	:
:	:	:	:	:	:
9700	9700	9700	9700	9700	9700
9850	9850	(lost)	(throw away)		
10000	(lost)	(lost)	:		
Run 638 sub-run 0 - run reset					
0	0	0	0	0	0
1	1	1	1	1	1
2	2	2	2	2	2
3	3	3	3	3	3

- **Data-driven recovery of EB buffers**
  - Data flow is blocked by a failure at one point
  - By finding new “sub-run” event fragment, read out data from the previous sub-run will be simply discarded
- **events are realigned at EB**
  - Done by throwing away events from previous (sub)run at EB input

Towards non-stop DAQ

- **TTD**
  - Possible to send runreset to only one FEE, or only one detector
  - Possible to set next event number in FEE to resume
- **FEE**
  - Need to check if it works if started from non-zero event number
- **Event-builder**
  - It is already designed to be able to merge events, even if part of the event is lost
- **COPPER**
  - Event-building (HSLBs) does not care the event numbers, so all 4 HSLBs have to be reset
- **ONSEN**
  - Need to check if it works if a bulk of event numbers are skipped

Non-stop DAQ upgrade

- **Chance to fix**
  - If the firmware for the DAQ upgrade board cares the exp/run/subrun/event number in the data stream, it can do the same thing as
- **Channel-by-channel recovery**
  - It should be able to reset an individual link while rest of the links are untouched
  - After link is reestablished, TTD will set the new event number to FEE from which data taking resumes
  - Therefore it would be nice if the link recovery can be also controlled from TTD

Summary

- Auto restart should be a simple problem if things are coordinated, but the problem is the lack of coordination
- Non-stop DAQ is a more challenging project