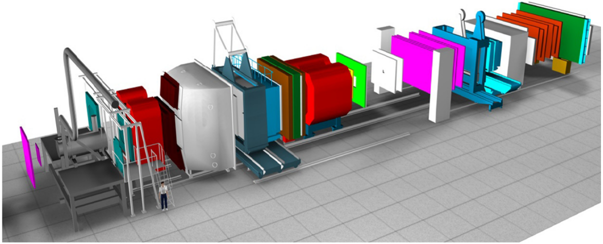
Plans for DAQ software beyond LS2

A. Kveton, M. Jandek, J. Novy

Faculty of Nuclear Sciences and Physical Engineering
Czech Technical University in Prague, Czech Republic
&
Charles University, Czech Republic
&
European Organization for Nuclear Research – CERN, Switzerland

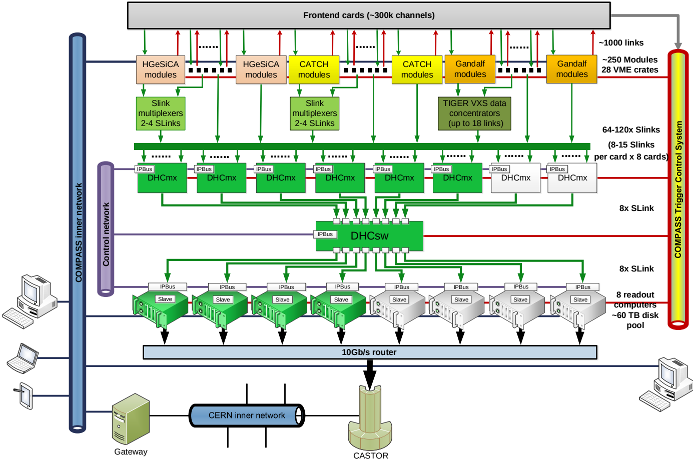
Targets of development

- Improve speed - upto 1 GB/s per readout computer
- Improve modularity - configurable dataflow through software
- Improve robustness - isolation of crucial tasks to specialized processes



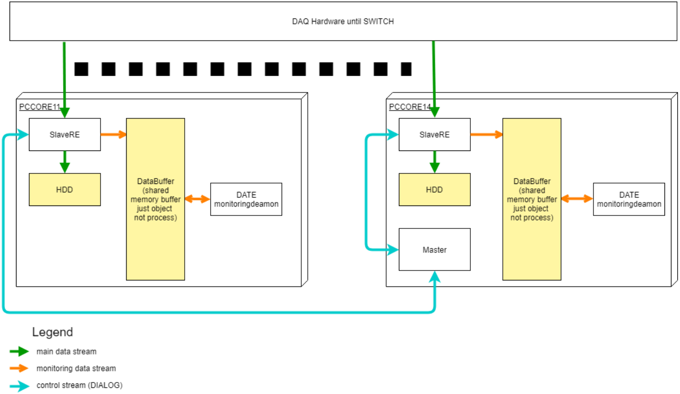
Present Structure

- Hardware based E.B.
- Data concentrated by 6 (up to 8) DAQ modules with multiplexer firmware
- Distribution to 4 (up to 8) readout computers by DAQ module switch firmware
- Full events received by servers
- Consistency check at many layers
- Events checked and transferred to DATE data format



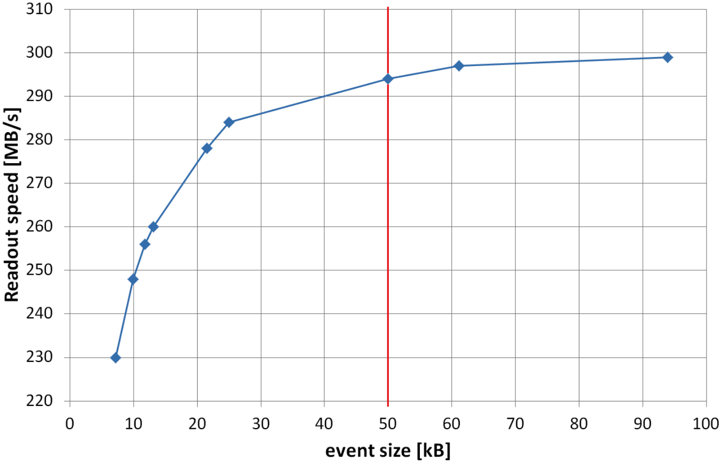
Present Structure

- Simple structure
- Obsolete monitoring tool
- No possibility of load balancing in software
- Inefficient data format



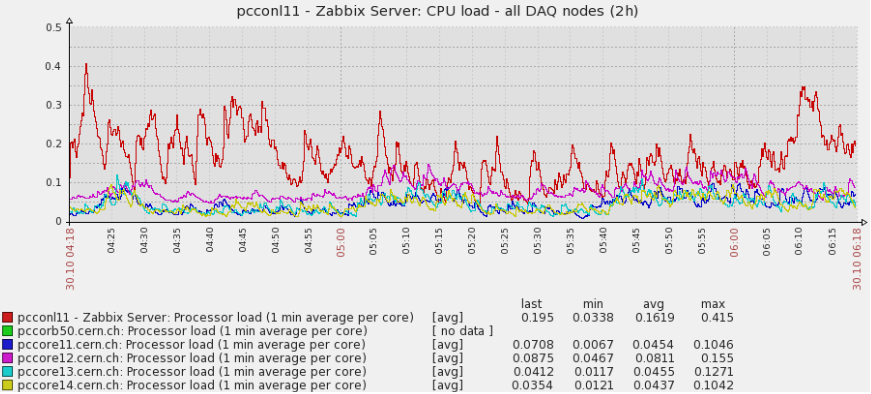
Present speed limit

- Up to 150 MB/s readout/processing speed per computer
- Limited by HDD and Spillbuffer



Present speed limit

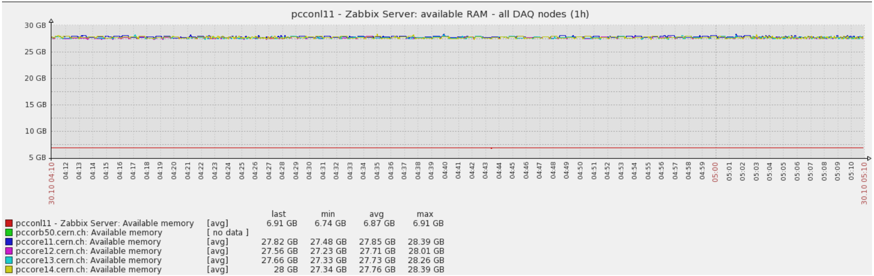
- Low CPU usage - with 6 core 2 GHz Xeon E5-2620 peak at 15 %



		last	min	avg	max
🟥 pcconl11 - Zabbix Server: Processor load (1 min average per core)	[avg]	0.195	0.0338	0.1619	0.415
🟩 pccorb50.cern.ch: Processor load (1 min average per core)	[no data]				
🟦 pccore11.cern.ch: Processor load (1 min average per core)	[avg]	0.0708	0.0067	0.0454	0.1046
🟪 pccore12.cern.ch: Processor load (1 min average per core)	[avg]	0.0875	0.0467	0.0811	0.155
▨ pccore13.cern.ch: Processor load (1 min average per core)	[avg]	0.0412	0.0117	0.0455	0.1271
🟨 pccore14.cern.ch: Processor load (1 min average per core)	[avg]	0.0354	0.0121	0.0437	0.1042

Present speed limit

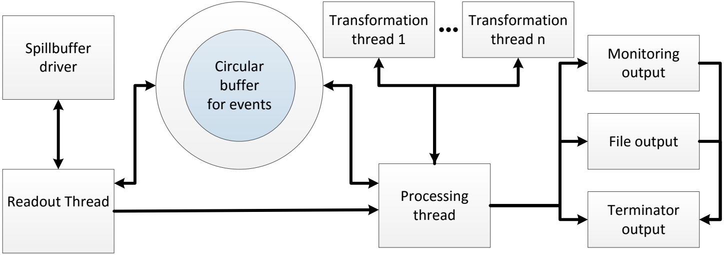
- Low RAM usage - less than 1 GB



		last	min	avg	max
🟥 pcconl11 - Zabbix Server: Available memory	[avg]	6.91 GB	6.74 GB	6.87 GB	6.91 GB
🟩 pccorb50.cern.ch: Available memory	[no data]				
🟦 pccore11.cern.ch: Available memory	[avg]	27.82 GB	27.48 GB	27.85 GB	28.39 GB
🟪 pccore12.cern.ch: Available memory	[avg]	27.56 GB	27.23 GB	27.71 GB	28.01 GB
🟦 pccore13.cern.ch: Available memory	[avg]	27.66 GB	27.33 GB	27.73 GB	28.26 GB
🟨 pccore14.cern.ch: Available memory	[avg]	28 GB	27.34 GB	27.76 GB	28.39 GB

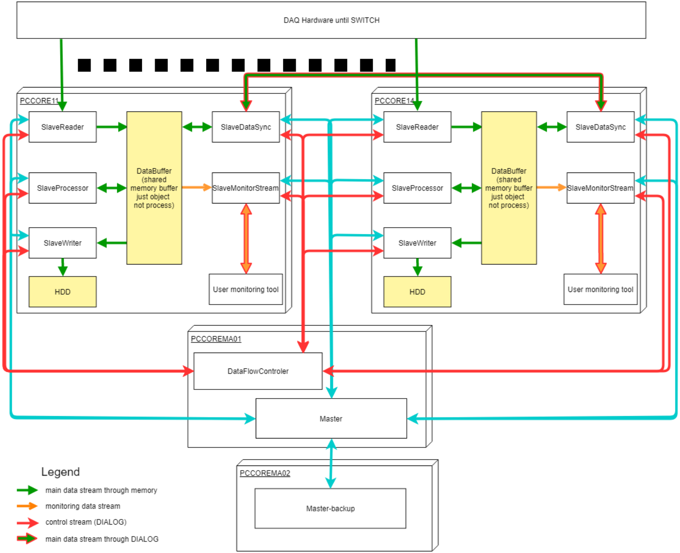
Present readout process

- Readout data from spillbuffer to server memory (internal circular buffer)
- Transformation to DATE format
- Data error checks
- File writing and distribution to other outputs



New software architecture

- Readout process divided to several sub-processes -> Spreading of the load
- Configurable storage target
- Possibility to add filtration processes
- Backup master process
- Scheduling -> load equalization
- New monitoring channel through DIALOG



New software architecture

SlaveReader

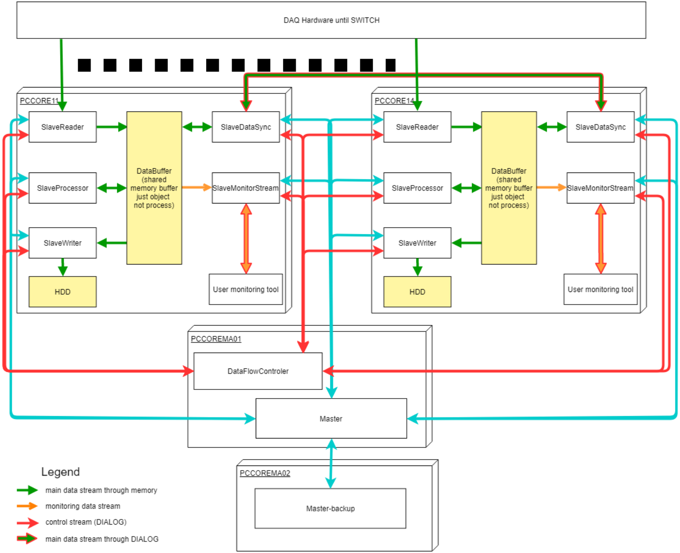
- Reads messages to shared DataBuffer
- Needed for each spillbuffer

SlaveDataSync

- Transfer between databuffers
- Needed for each server

SlaveMonitorStream

- Interface to monitoring tools



DAQ Hardware until SWITCH

PCCORE11: SlaveReader, SlaveProcessor, SlaveWriter, HDD, DataBuffer (shared memory buffer just object not process), SlaveDataSync, SlaveMonitorStream, User monitoring tool

PCCORE14: SlaveReader, SlaveProcessor, SlaveWriter, HDD, DataBuffer (shared memory buffer just object not process), SlaveDataSync, SlaveMonitorStream, User monitoring tool

PCCOREMA01: DataFlowControler, Master

PCCOREMA02: Master-backup

Legend

- main data stream through memory
- monitoring data stream
- control stream (DIALOG)
- main data stream through DIALOG

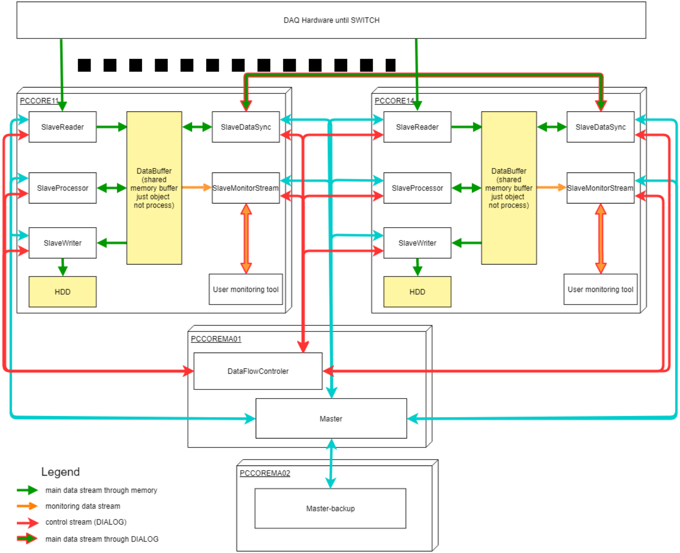
New software architecture

SlaveProcessor

- Not just one type of sub-process
- HLT
- Event building
- Format transformation

SlaveWriter

- Writing of data designated for storage
- Possibility to write data of one spill to one file



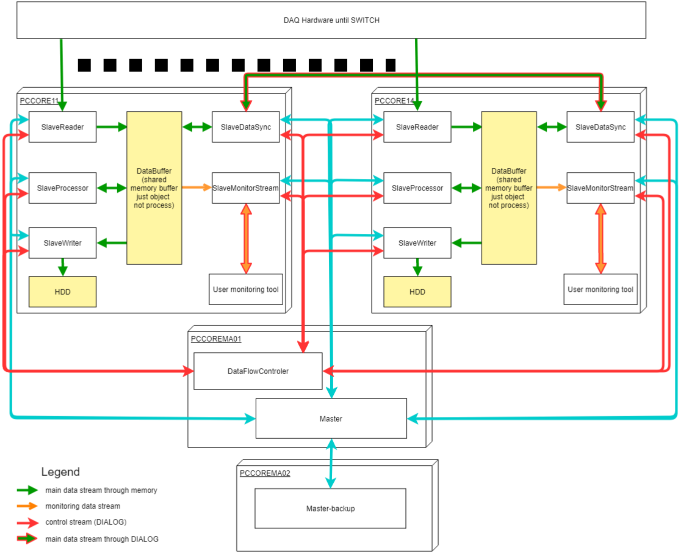
New software architecture

DataFlowControler

- Control of data transfer between buffers
- Scheduling
- Controls dataflow not states

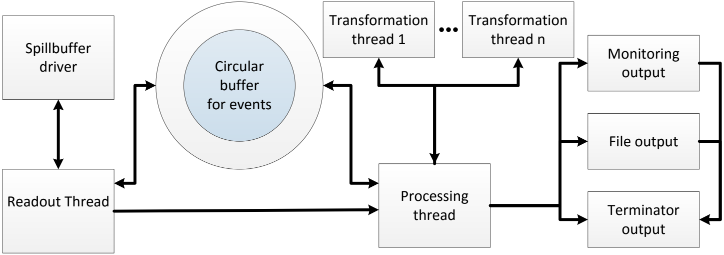
Master-backup

- Fail-safe



Transformation plan

- Possible to transform step by step
- Firstly separating monitoring and preparation of DataFlowControler
- Secondly separating file output
- Thirdly separating processing

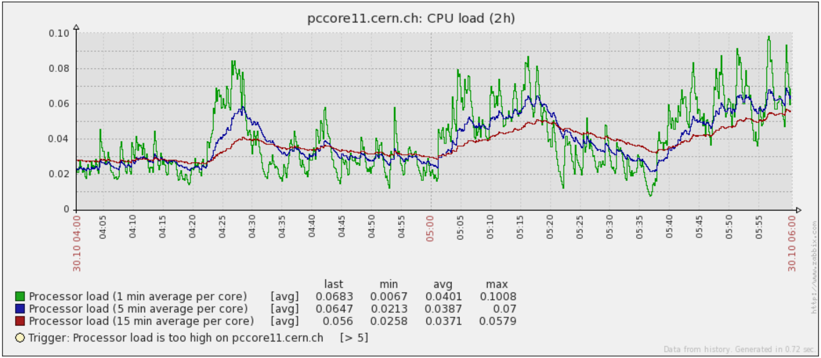


Triggerless 1 GB/s goal

- Processing speed not a problem
- New spillbuffer needed - new readout driver implementation
- Significant investment in HDD/SSD needed
  - at least two servers with 8 HDD or one with 4 SSD in RAID 10 per one 1 GB/s readout card
  - SSD faster, smaller, more expensive, reliability to be investigated (3 Drive Writes Per Day for 3 years)
  - NVMe SSD fastest, the most expensive
- Faster network connection to CASTOR or significant event filtration needed
- Store also raw data?

Data filtering

- SlaveProcessor sub-process could contain routines for data filtering at start



		last	min	avg	max
🟩 Processor load (1 min average per core)	[avg]	0.0683	0.0067	0.0401	0.1008
🟦 Processor load (5 min average per core)	[avg]	0.0647	0.0213	0.0387	0.07
🟥 Processor load (15 min average per core)	[avg]	0.056	0.0258	0.0371	0.0579

○ Trigger: Processor load is too high on pccore11.cern.ch [> 5]

Data from history. Generated in 0.72 sec.

Techniques of Data Filtering

- Possible approaches:
  - Pattern recognition
    - Neural networks
    - k-NN
    - kernel PCA
    - ...
  - Anomaly detection
    - Similar methods with different configuration may be used for pattern recognition
  - Searching for specific "known" data words and their combinations
- The basic problem is always the amount of available computational resources

Implementation

- Current architecture
  - Event building is done in DAQ hardware
  - Readout slaves receive whole events, analysis of events is therefore possible
- Triggerless DAQ
  - Data are received in "frames". Physics events may be spread over multiple frames
  - Event preparation from "frames" needed
  - All frames containing physics event must be available to a readout slave machine
  - New DAQ architecture contains a shared memory buffer and flowcontroler, which should take care of frames availability
  - All frames needed for an analysis of the event are available as a result

Scheduling (with triggers)

- Data read out by the SlaveReader process -> the scheduler decides what to do with them
- The scheduler has to make an educated guess of the processing time in order to be able to balance the load effectively.
- Information available to the scheduler:
  - Progress status of currently scheduled jobs on the individual pccores
  - Queued jobs per pccore
  - Event metadata
    - Event size
    - Trigger flags
    - Possible pattern recognition flags (filtration)
- The scheduler's estimation of processing time will be determined by a statistical (ML) approach

Scheduling (triggerless)

- Same idea, but we no longer have the trigger flags to use for the statistical model? -> increase need for some other evaluation -> possible usage of neural network
- HLT adds another stage of processing – which metadata can we use for the model?

General plans for DAQ SW

- Easier configuration (remove ALL the hardcoded constants)
- Continuous integration (GitLab)
  - Automated reformatting of code to comply with code style standards
  - Sanity tests (automated compilation)
  - In the future, possibly even more advanced automated testing
  - Command-line tools for faster deployment of new versions

General plans for DAQ SW

- Complete rework of run number incrementing logic as well as the logbook-accessing logic
- Logging improvements (especially the msglogger DB)
- User experience improvements
    - GUI threading
    - State machine timeout info
    - Video tutorial
    - Possibly ACK logic for MSGbrowser

Conclusion

- Huge amount of work in front of us
- Discussion about storage policy needed (where? when? what?)
- Discussion about resources needed for processing of triggerless data
- Step-by-step transition possible
- Many changes can be very useful also for 2021 run

The End

References

P. Abbon, et al.(the COMPASS collaboration): *The COMPASS experiment at CERN*. In: Nucl. Instrum. Methods Phys. Res., A 577, 3 (2007) pp. 455–518.

V. Y. Alexakhin, et al. (the COMPASS Collaboration): *COMPASS-II Proposal*. CERN-SPSC-2010-014, SPSC-P-340. May 2010.

M. Nakao , S. Y. Suzuki: *Network shared memory framework for the Belle data acquisition control system*. Real Time Conference, 1999. Santa Fe 1999. 11th IEEE NPSS.

C. Gaspar, M. Dönszelmann, Ph. Charpentier: *DIM, a Portable, Light Weight Package for Information Publishing, Data Transfer and Inter-process Communication*. International Conference on Computing in High Energy and Nuclear Physics, Padova, Italy, 1-11th February 2000.

C. Gaspar, M. Dönszelmann: *DIM – A Distributed Information Management System for the DELPHI Experiment at CERN*. Proceedings of the 8th Conference on Real-Time Computer applications in Nuclear, Particle and Plasma Physics, Vancouver, Canada, June 1993.

C. Gaspar, J. J. Schwarz: *A Highly Distributed Control System for a Large Scale Experiment*. 13th IFAC workshop on Distributed Computer Control Systems – DCCS'95, Toulouse, France, 27-29th September 1995.

M. Bodlak, et al.: *Development of new data acquisition system for COMPASS experiment*. Nuclear and Particle Physics Proceedings, 37th International Conference on High Energy Physics (ICHEP). April–June 2016, vol. 273–275, pp. 976–981. Available at: `http://dx.doi.org/10.1016/j.nuclphysbps.2015.09.153`.

M. Bodlak, et al.: *FPGA based data acquisition system for COMPASS experiment*. Journal of Physics: Conference Series. 2014-06-11, vol. 513, issue 1, s. 012029-. DOI: 10.1088/1742-6596/513/1/012029. Available at: `http://stacks.iop.org/1742-6596/513/i=1/a=012029?key=crossref.78788d23de2b4a6a34d127c361123b8c`.

M. Bodlak, et al.: *New data acquisition system for the COMPASS experiment*. Journal of Instrumentation. 2013-02-01, vol. 8, issue 02, C02009-C02009. DOI: 10.1088/1748-0221/8/02/C02009. Available at: `http://stacks.iop.org/1748-0221/8/i=02/a=C02009?key=crossref.a76044facdf29d0fb21f9eefe3305aa5`.

References

M. Bodlak, et al.: *Developing Control and Monitoring Software for the Data Acquisition System of the COMPASS Experiment at CERN*. Acta polytechnica: Scientific Journal of the Czech Technical University in Prague. Prague, CTU, 2013, issue 4. Available at: http://ctn.cvut.cz/ap/.

T. Anticic, et al. (ALICE DAQ Project): *ALICE DAQ and ECS User's Guide* CERN, EDMS 616039, January 2006.

C. Ghabrous Larrea, et al.: *IPbus: a flexible Ethernet-based control system for xTCA hardware*, 2015 JINST 10 C02019. doi:10.1088/1748-0221/10/02/C02019.

CASTOR – CERN Advanced Storage manager. Available at: http://castor.web.cern.ch. (Accessed: 2017-05-01).

Electronic developments for COMPASS at Freiburg. Available at: http://hpfr02.physik.uni-freiburg.de/projects/compass/electronics/catch.html. (Accessed: 2017-05-01).

The GANDALF Module. (online). Available at: http://hpfr03.physik.uni-freiburg.de/gandalf/pages/information/about-gandalf.php?lang=EN. (Accessed: 2017-05-01).

iMUX/HGESICA module. (online). Available at: https://twiki.cern.ch/twiki/pub/Compass/Detectors/FrontEndElectronics/imux_manual.pdf. (Accessed: 2017-05-01).

Linux at CERN. (online). Available at: http://linux.web.cern.ch/linux/scientific6/. (Accessed: 2017-05-01).

S-Link – High Speed Interconnect. (online). Available at: http://hsi.web.cern.ch/HSI/s-link/. (Accessed: 2017-05-01).