CLAS12 Software and Computing Deep Dive

Graham's piece

CLAS12 Software and Computing Deep Dive

- DAQ
  - Bit packing
  - Trigger improvements
  - Firmware road finding
  - Lessons learned from Spring 2018 running.
    - Improvements to file handling and organization.
- Code review and testing
- Reconstruction improvement summary
- Presentation of current CLAS12 computing requirements.
  - Reconstruction
  - Analysis
  - Simulation
- Progress towards meeting the requirements.
- Summary

Charge items:
1) b, c, f
2) a, b
3) a, b, c

DAQ

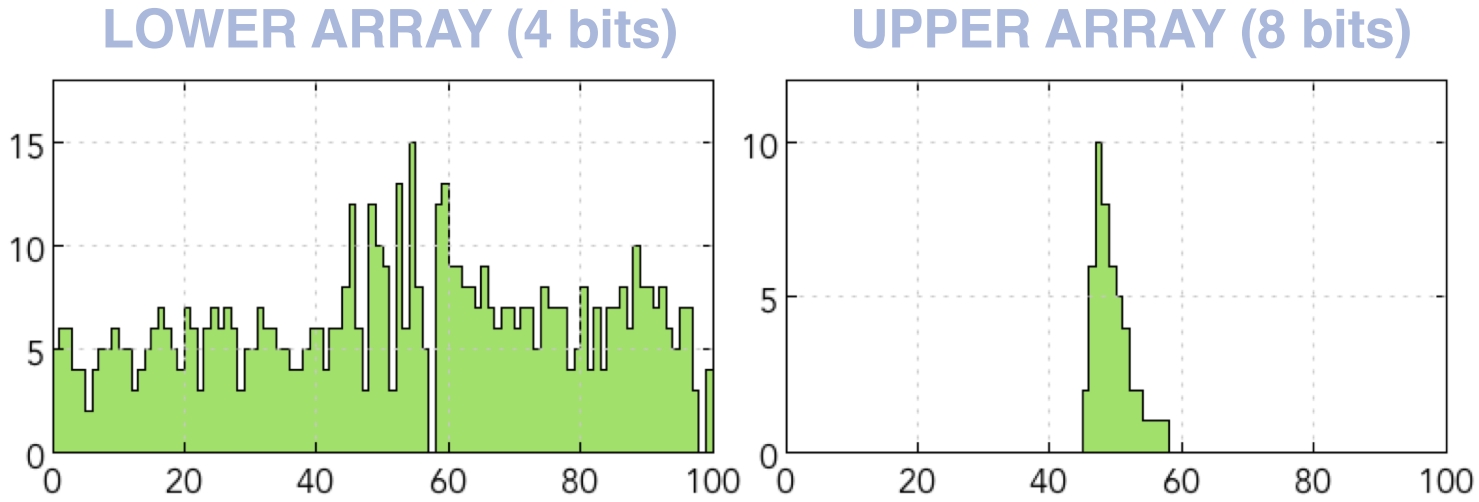
- The DAQ system for CLAS12 is based on the standard CODA architecture used by all halls but is one generation behind.
- Highly optimized for CLAS12's hardware configuration.
- During the Spring 2018 run was taking data at 600+ MB/s.
  - More than the rate of 200MB/s quoted in the last review.
  - Increase due to desire to keep all samples from FADCs.
- Some remaining bugs in software and firmware have been fixed and the system runs reliably.
- Earlier review had concerns over spares and manpower.
  - Spares have been bought.
  - DAQ support group is helping out until manpower issue is resolved by hiring or reassignment.

Bit packing

- Over 60% of the data in an event is generated by the Flash ADCs.
  - CLAS12 experiments want to keep all samples in the raw data.
- A more efficient packing of the data compared with the raw output of the fADCs allows a reduction in the size of the fADC data.
- The following table compares the event size before and after bit packing (BP) and for BP plus other optimizations. These are at the same beam energy and intensity. (Optimizations were improved thresholds, timing windows, etc. which lead to ~20% smaller events).
- Compared with Spring 2018 the current event significantly reduces the data rate for the same trigger rate.

Run period	Run number	Event size (kB)
Spring	4013	41.6
Fall no BP	4934	48.0
Fall BP	4930	31.1
Fall BP + Optim.	5030	26.2

How bit-packing works



- The fADCs are 12-bits yet the upper 8 bits are zero for most samples.
- Split upper 8 and lower 4 bits into two arrays :
- First, encodes in **52** bytes lower 4 bits of samples:
  - **2** bytes for pedestal height
  - **50** bytes for **100** samples of **4** bits
- Second array is encoded into series of bytes
  - leading and trailing zeros are suppressed
  - **1** byte is index of first non-zero byte
  - **N** following bytes are the top section of the pulse.

Current Example:
63 bytes used
52 for lower
10 bytes for high bits
1 byte for first non-zero

Jefferson Lab

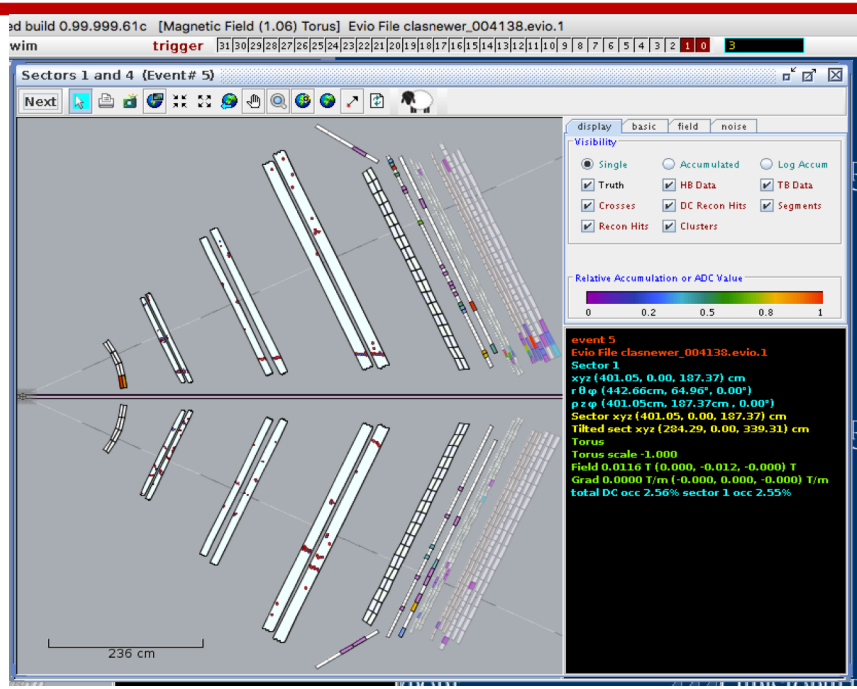
2016 review recommendation – consider compression

Method	File Size	Ratio
Raw	1.87 MB	1.00
Raw (LZ4)	0.81 MB	0.43
Raw (GZIP)	0.71 MB	0.37
Bit Packed	0.63 MB	0.34
Bit Packed (LZ4)	0.55 MB	0.29
Bit Packed (GZIP)	0.48 MB	0.26

- Packed fADC data volume is 34% of unpacked
- Using GZIP compression we could get close to this size.
  - CPU intensive
  - Not easy to code on an FPGA.
- Final bit packed data can still be compressed as well for writing to data file.

CLAS12 Drift Chamber-based trigger

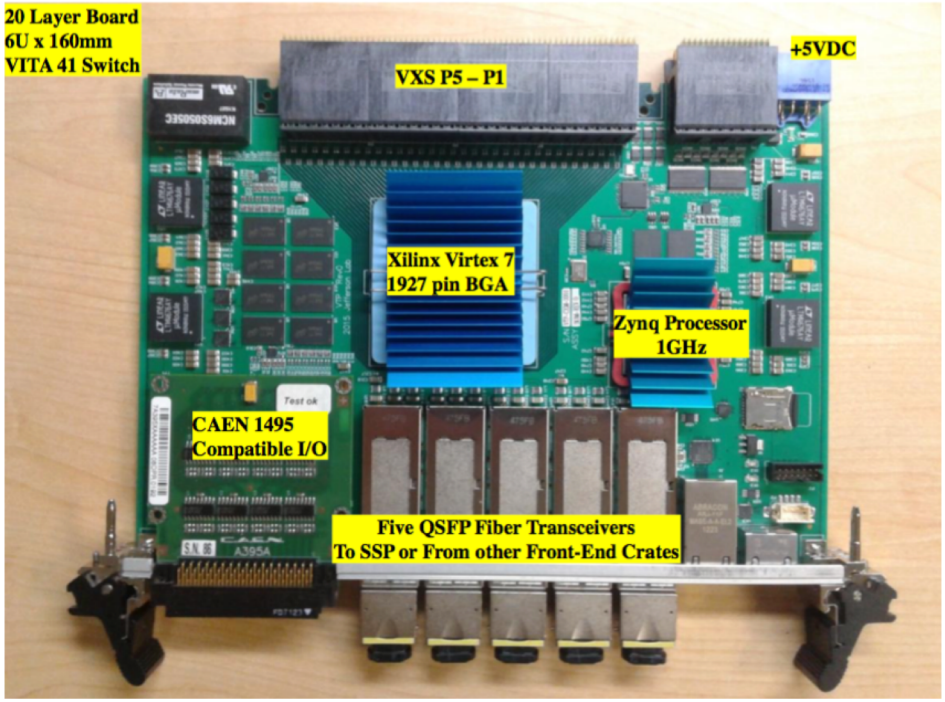
- CLAS12 use a custom VXS based board, the VTP, to read out the chambers. It was realized that a significant reduction to the trigger rate could be achieved using the ASIC on the VTP board.
- A dictionary based segment finder in each of 6 superlayers; every superlayer has 6 layers, trigger requires at least 4 layers to have hits
- Segments multiplicity (5 out of 6 superlayers must have segments), or road finder (based on road dictionary)
- Stage2: timing (and possible geometry) coincidence with other detectors.



- Segment dictionary and road dictionary was generated using MC.
- VHDL code generated using C to VHDL generator program and incorporated into VTP FPGAs.
- Trigger validation performed using beam data with trigger from random pulser.
- Resulting trigger decreases event rate up to 30% depending on run conditions, with efficiency close to 100% for electrons with momentum above 1GeV
- Road finder gives extra 10-12% event rate decrease for inbending electrons in compare with segments multiplicity, but only 2-3% for outbending electrons; it is mostly efficient for noisy events where segment multiplicity alone does not work very well

VTP

- The board below is a VTP which plugs into a VXS switch slot.
- It was designed as a general purpose trigger processor.
- It is configurable using firmware to read out data over the VXS serial backplane and make trigger decisions
- It can read data over the serial lanes of VXS faster than a regular CPU on VME.
- The Virtex 7 has enough capacity for complex algorithms and large lookup tables.
- The Zynq processor provides configuration and control.



Jefferson Lab

DAQ lessons learned from Spring 2018 running.

- Rate in Spring was ~600 MB/s
  - Data staged on local disk in counting house.
  - Script is triggered periodically to copy data to data center.
- MSS saw a huge dump of data and assigned several tape drives to archive to tape.
  - Data from a run spread randomly over several tapes (as many as ten).
- Tape replay problem - requests of 10 input files for decoding required mounting many tapes. Process I/O limited. Solutions:
  - Request entire tapes and stage to disk write decoded HIPO files back to tape in sequence – needs a lot of space.
  - Read back all of Spring 2018 files, sort raw and rewrite to tape.
- Going forward data will be moved more frequently to the datacenter to prevent spread over many tapes.
  - Aided by factor of ~2x lower DAQ data rate.
  - All data files from a run will now be written in a subdirectory named with the run number – prevents Lustre problems due to thousands of files in the same directory.
  - Data files in the same run are named in a fixed format to make file sorting simpler. (e.g. clas_1234.evio.00041 instead of clas_1234.evio.41).

Code review and testing

- The CLARA framework allows for :
  - Code profiling – how long do individual algorithms take.
  - Code isolation
    - Algorithms can be run independently of each other.
    - Algorithm code is self contained inside the service class.
- Our best Java programmers were tasked to review the code of individual services.
  - Many “interesting” code constructs found and fixed.
- Profiling showed some services slowed by thread contention.
  - Worked through them one by one and optimized code
- Found that some code was much slower on the new 2018 nodes than the 2016 ones due to changes in CPU architecture.
  - CLARA now detects different node types and adjusts thread affinity to optimize execution.

Summary of Reconstruction Improvements

- Results of improvements to the reconstruction code.
  - Measured on a Farm16 node
  - RK4 = Runge Kutta 4 algorithm – swimming in field.

Release/Branch	5b.6.2	vg-optimized	vg-optimized	5c.7.0
RK4 implementation	-	X	X	X
Limit on KF iterations	-	-	X	X
Code clean-up	-	-	X	X
Fast Math libraries	-	-	X	X
JRE warmup latency	-	-	-	X
Thread affinity	-	-	-	X
Object pool implementation	-	-	-	-
Code vectorization	-	-	-	-
KF service on GPU	-	-	-	-
1 Thread [mS]	1855 + (unstable)	1104 + (unstable)	787+ (unstable)	289.9 (stable)
Rate/Node [Hz]	19.0	33.0	45.6	94.2

Current reconstruction rates.

- The current farm is a heterogenous mix of three node types.
- The measured throughput of fully loaded nodes are shown in the table below.
- Average reconstruction rate of is ~92 Hz per node.

Node Type	Number of nodes	job slots/node	total job slots	Rate/node (Hz)	Rate/job slot(Hz)	Rate/flavor (Hz)
farm14	83.0	48.0	3984.0	61.5	1.28	5104.5
farm16	46.0	72.0	3312.0	94.2	1.31	4333.2
farm18	87.0	80.0	6960.0	119.6	1.50	10405.2
total	216.0*		14256.0	91.8**		91.9***

* total number of farm nodes
** arithmetical average of reconstruction rates
*** weighted average of reconstruction rates

Reconstruction requirements.

- CLAS allocation of local farm is 45% = 97 nodes.
- At 92 Hz per node this represents ~9 kHz reconstruction rate.
- Data backlog from 2018 and all 2019 production data will require 147 days to reconstruct. -> factor of two headroom…
  - Needs an efficient workflow to keep the farm busy.
  - Practical experience shows ~50% efficiency in keeping farm busy.
  - 147 compute days ~300 wall clock days per year, which is sustainable.

- Experience with current run shows validation and calibration to be ~10% of the reconstruction workload.
  - Can dedicate Farm nodes for calibration ~ 10 nodes

Run		Cal. days	rate kHz	Billions of events	Days to process
RG- A Spring 2018	2018 Spring	44	13	22.5	29
RG-A Fall 2018	2018 Fall	59	13	33	42
RG-K	2019	18	13	10	13
RG-B	2019	98	7.8	33	42
RG-A Spring 2019	2019	28	13	16	21
			Total	141.5	147

Analysis requirements.

- Analysis trains are fast but I/O intensive. Contribute to MSS bandwidth requirement but not a major contributor to compute workload.
- Analysis of train output
  - RG-A and RG-B are the major contributors to the workload. Use experience from RG-A to estimate analysis workload.
  - Expect only 20% of events will survive analysis train filtering.
  - Of these half will be rejected early in analysis.
  - So only 10% of the events are fully analyzed.
  - Per node analysis rate 10x reconstruction rate.
  - Even multiple analysis passes are not a significant contributor to the compute workload.
  - Input datafiles are small – not a contributor to bandwidth needs.

Non-simulation summary.

		2018		2019		2020	
Run		Spring	Fall	Spring+Fall	Summer	Spring	Fall
Exp		RG-A	RG-A/K	RG-A/B	RG-I (HPS)	RG-F	?
events	Billions	22.5	43	49	36	8	?
Load	M core hr	2.4	4.5	5.2	3.8	0.85	?

- Reconstruction rate has been significantly improved.
- RG-A and RG-B generate the largest compute loads – high rate, long duration.
- Backlog from spring is ~22 B events or 2.4 M core hours.
- Backlog + added load through end of 2019 is 141.5 B events ~15 M core hours
- Each full year adds 7 to 9 M core hours.
- Hall-B's current quota of 45% of the local 80M core hr/yr. cluster, is 36 M core hours.

Simulation requirements

- Event generation 200-500 mS/event – depends on channel studied
  - Up from last review but more realistic.
  - Generation is followed by reconstruction – 250 mS/event
  - Total CPU cost per event 500+250 mS = 750 mS or 3x reconstruction.

- Overall statistics for reactions analyzed is ~10% of all raw events
  - Estimated from observed Spring run skim sizes.
- Analyzers would like 10 times larger statistics for MC events.
- Typical acceptance ratio between recon. and MC events is ~30%,
- This leads to a ratio between MC and raw of: 10% x 10 x 3 = **3**
  - This is lower than 6 we had in 2016 and likely conservatively high.

- **Bottom line - simulation workload 9x reconstruction.**
  - 3x the statistics and 3x more costly per event.

- Using data from previous slide:
  - **63 M core hours simulation associated with 1 yr of raw data.**
  - This 63 M hours of work will be spread over more than one year.
  - RG-A and RG-B are the largest contributors, load will decrease after 2019.

Progress towards meeting the requirements.

- Local
  - The local Farm is currently adequately sized to calibrate and reconstruct the raw data.
    - Reconstruction ~150 days at 45% of existing farm.
    - Calibration and validation ~ 10 dedicated nodes while running.
      - Looking at making this an online resource.

- Plan to provision for simulation offsite ~63 M core hours per year
  - OSG – pioneered by GLUEX, we will follow.
    - Submit jobs from JLab OSG submit node.
    - Collaborators contribute to OSG – in particular MIT.
  - Other
    - Submit simulation in Docker container locally at the remote site using remote site's staff and batch system. Return results to Jlab.
    - INFN and others interested in this.
  - NERSC
    - Submit via SWIF2 workflow tool (follow GLUEX).
    - Have requested NERSC allocation of 30M core hours.
      - Enough to cover 50% of the annual simulation workload.

Summary

- Significant progress in several areas:
  - Understanding and optimizing DAQ and trigger.
  - Implementation of bit-packing and road finding.
    - Bit-packing significantly reduces event size.
    - Road finding reduces event rate.
  - Reconstruction
    - Optimized code and algorithms to significantly speed recon. rate.
    - Workload now easily fits into allocation of 45% of existing farm (for a single pass).
  - Analysis trains
    - Operational and significantly increasing efficiency by sorting data into small files containing only the data of interest to experiments.
  - Simulation
    - Code containerized
    - Needs of run groups better understood.
    - Simulation requirements understood
    - Offsite resources are identified which will allow all of the required simulation
- Software and the resources to support them are in great shape!