

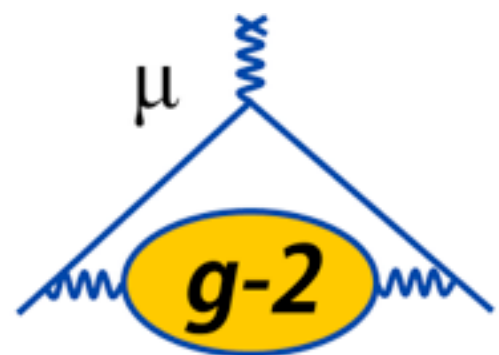


Data Acquisition and Online Monitoring Tools

Wes Gohn

University of Kentucky

2 October 2017

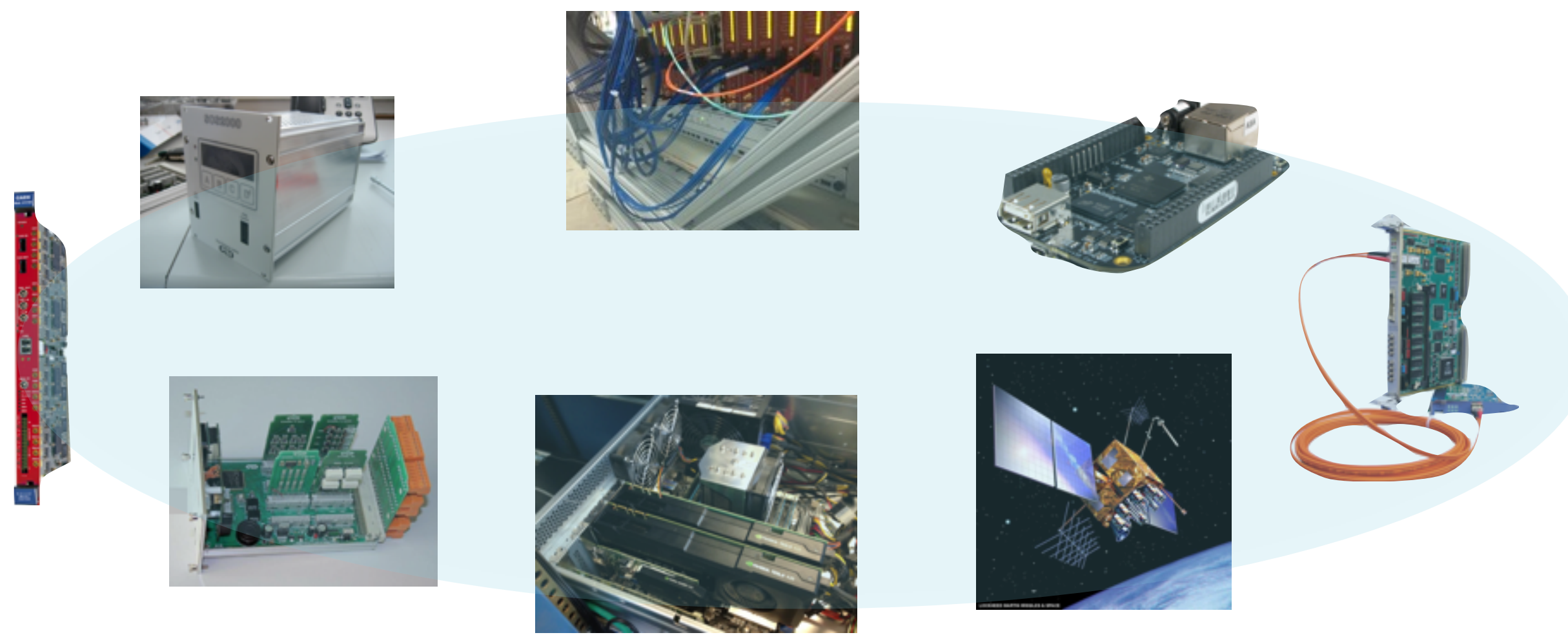


Charge Questions



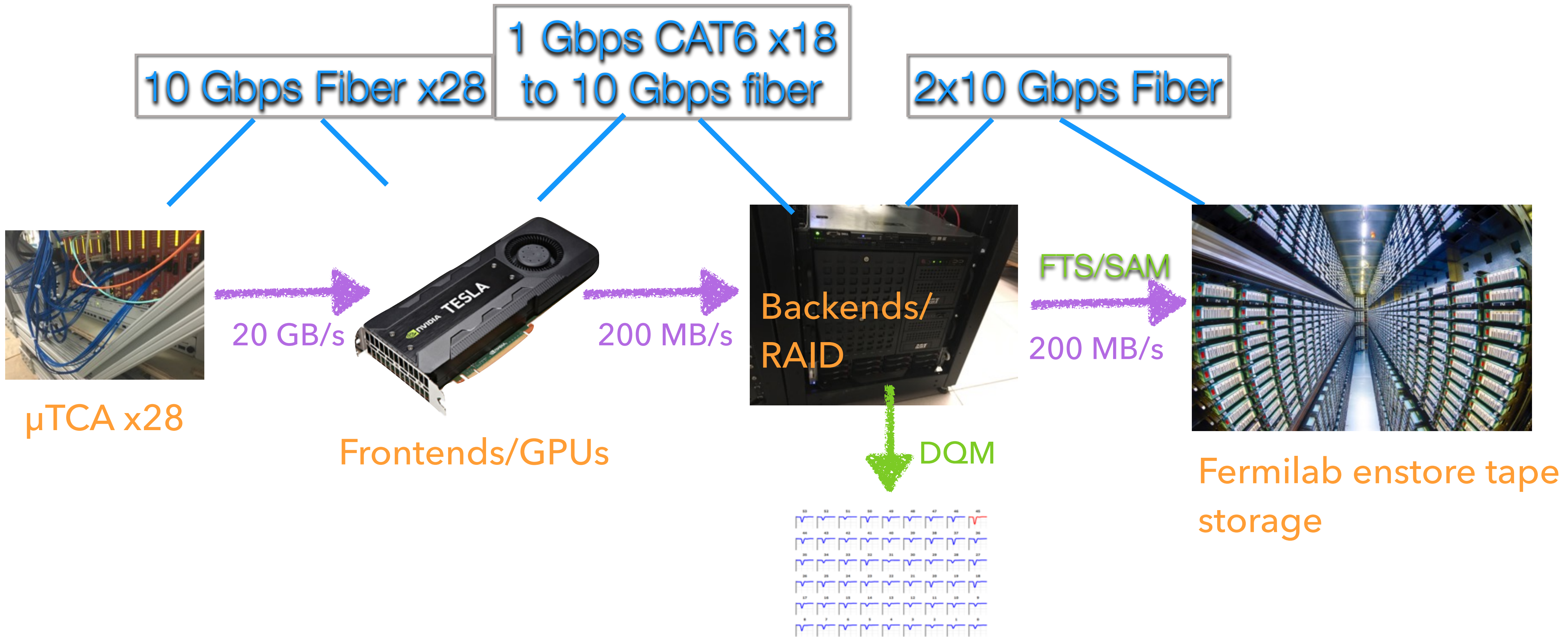
- Charge Questions 1h: “a description of the roles and responsibilities of each institution.”
 - will list institutions with contributions to DAQ systems
- Charge Question 2: “Has it been demonstrated that the experiment is ready for physics-quality data taking? If not, what actions are required to make it ready? **Is there a clear plan for monitoring (the beam and) the data quality and has the associated infrastructure been tested?** If not, what actions are required to adequately monitor the data quality?”

DAQ Overview



- The DAQ must assemble data from an “Internet of things” to form a complete picture of each muon fill.
- Expect an average data rate of 12 Hz for muon fills, plus some laser and pedestal fills.
- This provides data at a rate of 20 GB/s, which must be reduced via processing of data in GPUs.
- An art-based online data quality monitor allows us to monitor the data.

Data Flow



Data storage



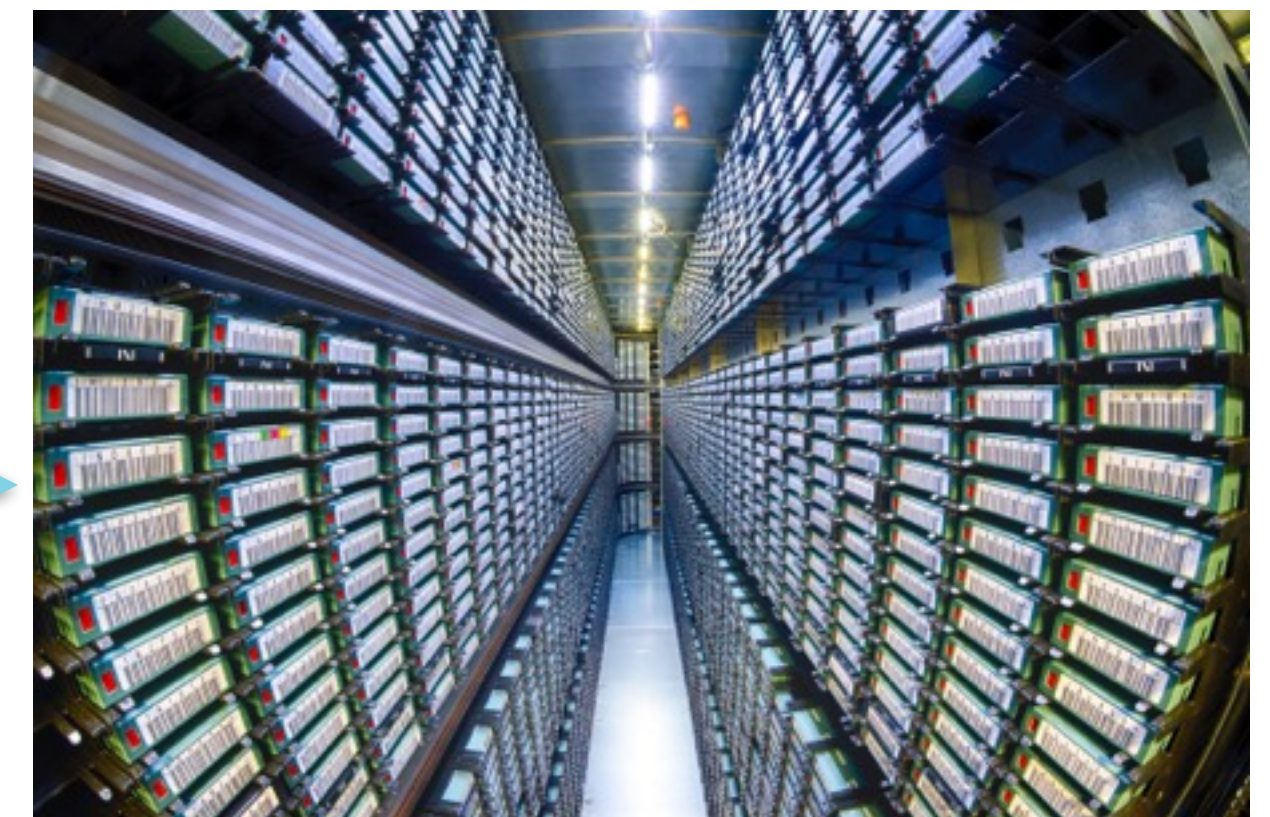
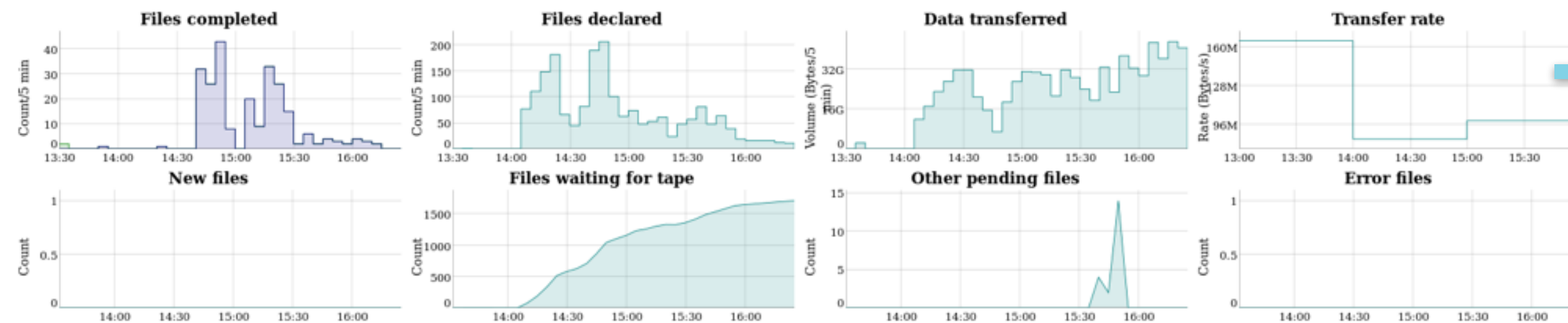
- Data is copied from local RAID using FTS/SAM to the Fermilab enstore tape system.
- The local RAID can accommodate four days of running if the link goes down.
- Both CRC32C and enstore checksums are computed for each file.
- A python plugin extracts the SAM metadata from a MIDAS JSON configuration file after each run.
- We wrote 20 TB of data during the commissioning run.
- We expect to write 6 PB of data in total.
- Two copies of the raw data will be stored in the MIDAS format as well as derived data in the *art* format.

FTS status for gm2-fts-gm2samgpvm01

Generated at 2016-06-24 16:27:04 CDT [\(update\)](#)

Summary

FTS: OK	SAM: OK
Completed files:	244
Failed transfers:	0
All error files:	0
Waiting on tape:	1719
Other pending files:	0
New files:	0



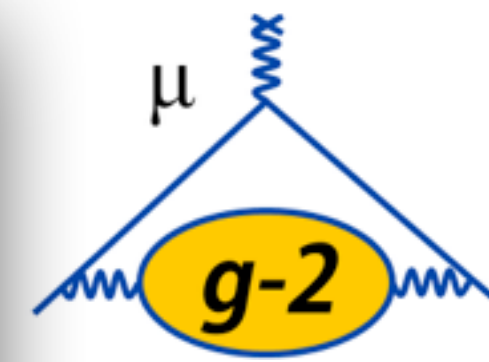
MIDAS configuration

- 32 fast frontends (data at beam fill rate).
- 35 slow control frontends.
- Midas alarm system.
- Midas sequencer used for calibration runs.
- ODB dumped to JSON file and saved to Postgres database at each end of run.
- Online analyzer using *art* and javascript.
- Separate MIDAS experiment running for magnetic field DAQ.

The screenshot displays the MIDAS Run Status interface. At the top, there are navigation buttons for Status, ODB, Messages, Alarms, Programs, Sequencer, Config, and Help. Below this, a 'Run Status' section shows the current run is 'Running' (Run 1618) with a start time of Wed Jun 28 05:38:50 2017 and a running time of 0h29m03s. The experiment name is GM2, and the data directory is /data2/gm2. A table below lists the status of various equipment, including MasterGM2, EB, AMC1300-1326, StrawTrackerLVandSC03, StrawTrackerDAQ, StrawTrackerHV03, IBMS Detector, CaloSC01-24, ESQ, IFIX, mscb110-174, and KickerSC_mscb282. The table columns are Equipment, Status, Events, Events[/s], and Data[MB/s]. At the bottom, there is a 'Logging Channels' section showing the channel #0: gm2_run01618_48.mid with 4992 events, 48906.884 MiB written, and a disk level of 81.1%.

Equipment +	Status	Events	Events[/s]	Data[MB/s]
MasterGM2	MasterGM2@g2be1.fnal.gov	1222	0.7	0.000
EB	Ebuilder@g2be1.fnal.gov	1221	0.0	0.000
AMC1300	AMC1300@g2aux-priv	1223	1.0	0.001
AMC1301	AMC1301@g2calo0102-data	1223	1.0	2.016
AMC1302	AMC1302@g2calo0102-data	1222	0.7	1.949
AMC1303	AMC1303@g2calo0304-data	1221	0.7	0.096
AMC1304	AMC1304@g2calo0304-data	1221	0.7	0.095
AMC1305	AMC1305@g2calo0506-data	1222	0.9	1.780
AMC1306	AMC1306@g2calo0506-data	1223	1.0	1.986
AMC1307	AMC1307@g2calo-spare-priv	1222	0.7	1.939
AMC1308	AMC1308@g2calo-spare-priv	1221	0.7	0.093
AMC1309	AMC1309@g2calo0910-data	1221	0.7	0.094
AMC1310	AMC1310@g2calo0910-data	1222	0.7	1.968
AMC1311	AMC1311@g2calo1112-data	1222	0.7	1.940
AMC1312	AMC1312@g2calo1112-data	1223	1.0	1.944
AMC1313	AMC1313@g2calo1314-data	1223	1.0	1.972
AMC1314	AMC1314@g2calo1314-data	1222	0.7	1.963
AMC1315	AMC1315@g2calo1516-data	1221	0.7	0.095
AMC1316	AMC1316@g2calo1516-data	1223	0.7	1.911
AMC1317	AMC1317@g2calo1718-data	1222	0.7	1.954
AMC1318	AMC1318@g2calo1718-data	1223	1.0	1.928
AMC1319	AMC1319@g2calo1920-data	1222	0.7	1.892
AMC1320	AMC1320@g2calo1920-data	1221	0.7	0.092
AMC1321	AMC1321@g2calo2122-data	1221	0.7	0.093
AMC1322	AMC1322@g2calo2122-data	1221	0.7	0.125
AMC1323	AMC1323@g2calo2324-data	1222	0.7	1.858
AMC1324	AMC1324@g2calo2324-data	1223	1.0	1.977
AMC1325	AMC1325@g2laserdaq-data	1221	0.7	0.069
AMC1326	AMC1326@g2aux-priv	1221	0.7	5.735
StrawTrackerLVandSC03	StrawTrackerLVandSC03@g2tracker1.fnal.gov	0	0.0	0.000
StrawTrackerDAQ	StrawTrackerDAQ@g2tracker0.fnal.gov	1221	0.7	0.006
StrawTrackerHV03	StrawTrackerHV03@g2tracker1.fnal.gov	0	0.0	0.000
IBMS Detector	IBMS Detector@g2ibms-priv	1223	0.7	0.121
CaloSC01	CaloSC01@g2sc-priv	0	0.0	0.000
CaloSC02	CaloSC02@g2sc-priv	0	0.0	0.000
CaloSC03	CaloSC03@g2sc-priv	0	0.0	0.000
CaloSC04	CaloSC04@g2sc-priv	0	0.0	0.000
CaloSC05	CaloSC05@g2sc-priv	0	0.0	0.000
CaloSC06	CaloSC06@g2sc-priv	0	0.0	0.000
CaloSC07	CaloSC07@g2sc-priv	0	0.0	0.000
CaloSC08	CaloSC08@g2sc-priv	0	0.0	0.000
CaloSC09	CaloSC09@g2sc-priv	0	0.0	0.000
CaloSC10	CaloSC10@g2sc-priv	0	0.0	0.000
CaloSC11	CaloSC11@g2sc-priv	0	0.0	0.000
CaloSC12	CaloSC12@g2sc-priv	0	0.0	0.000
CaloSC13	CaloSC13@g2sc-priv	0	0.0	0.000
CaloSC14	CaloSC14@g2sc-priv	0	0.0	0.000
CaloSC15	CaloSC15@g2sc-priv	0	0.0	0.000
CaloSC16	CaloSC16@g2sc-priv	0	0.0	0.000
CaloSC17	CaloSC17@g2sc-priv	0	0.0	0.000
CaloSC18	CaloSC18@g2sc-priv	0	0.0	0.000
CaloSC19	CaloSC19@g2sc-priv	0	0.0	0.000
CaloSC20	CaloSC20@g2sc-priv	0	0.0	0.000
CaloSC21	CaloSC21@g2sc-priv	0	0.0	0.000
CaloSC22	CaloSC22@g2sc-priv	0	0.0	0.000
CaloSC23	CaloSC23@g2sc-priv	0	0.0	0.000
CaloSC24	CaloSC24@g2sc-priv	0	0.0	0.000
ESQ_slow	ESQ_slow@g2quad-01	1729	1.0	0.000
ESQ	ESQ@g2quad-02-priv	1223	0.7	0.000
IFIX	Ok	173	0.0	0.000
mscb110	Ok	29	0.0	0.000
mscb13e	Ok	2871	0.0	0.000
mscb319	Ok	29	0.0	0.000
mscb323	Ok	29	0.0	0.000
KickerSC_mscb282	Ok	29	0.0	0.000
mscb174	Ok	29	0.0	0.000
Beam	Beam@g2sc-priv	346	0.3	0.000

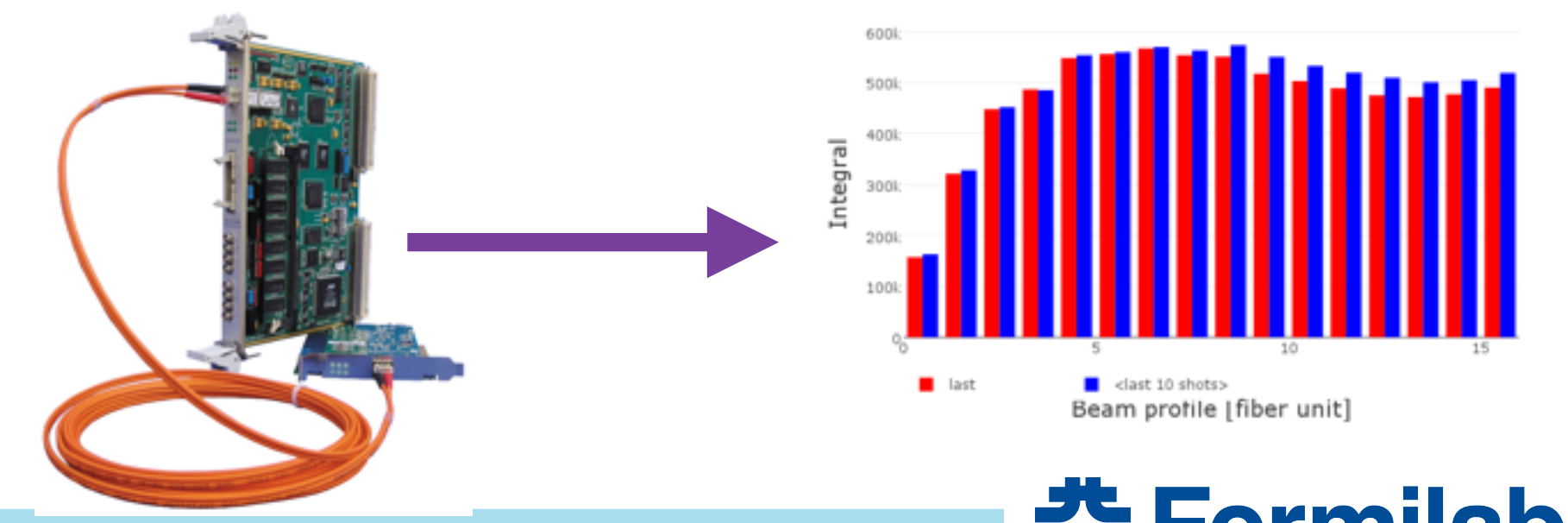
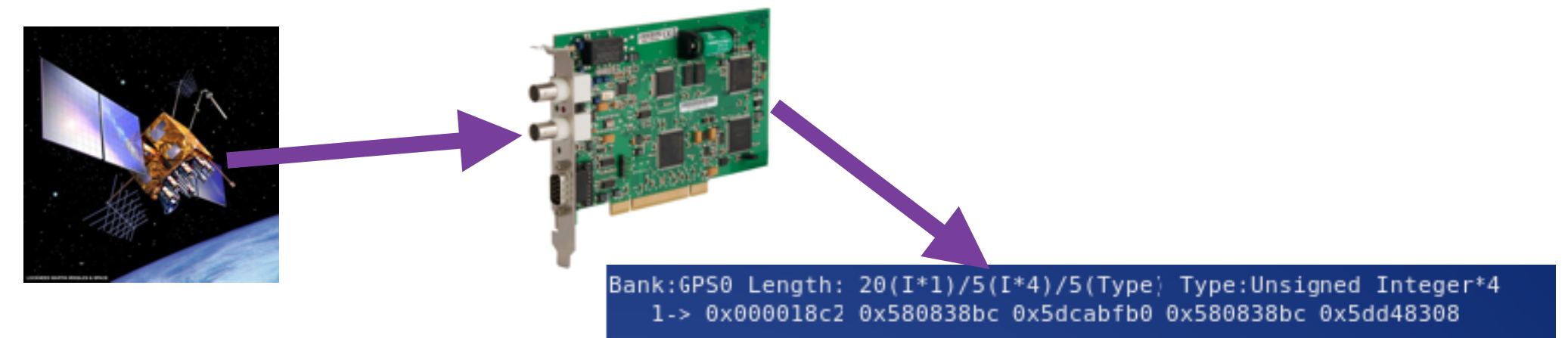
Channel	Events	MiB written	Compr.	Disk level
#0: gm2_run01618_48.mid	4992	48906.884	N/A	81.1%



MIDAS Frontends



- Master:
 - Communicates with other frontends with RPCs, sets up CCC, and writes GPS timestamps from Meinberg GPS unit.
- AMC13 frontend:
 - Main frontend for processing data from calorimeters, laser, fiber harps, quads, and kickers.
 - Processes the data with Nvidia Tesla K40 GPUs
- Tracker frontend:
 - Data comes from multihit TDCs that are read via FC7 cards.
- IBMS frontend:
 - Data from the inflector beam monitoring system (IBMS) is read out via a CAEN digitizer.



Slow Controls



- Slow control data is read from six SCS3000 mscb devices and 24 beaglebones.
- HV and LV frontends for tracker system.
- Slow frontend reading magnet properties from IFIX via an OPC client.
- Slow control data is stored in a Postgres database and displayed using a custom Django web display.



Field DAQ



- Field DAQ runs in independent MIDAS experiment.
- Contains seven asynchronous frontends reading data from fixed magnetic field probes and from a trolley that periodically transverses the ring to perform precision measurements of magnetic field.
- Data is correlated with the fast DAQ offline using GPS timestamps.

The image shows two screenshots from the MIDAS control interface. The top screenshot displays the 'Run Status' section, indicating that Run 395 is running. It shows the start time (Thu Jun 1 08:52:27 2017), running time (0h26m35s), and data directory (/home/newg2/gm2Data/). Below this is the 'Equipment' section with a table of system components and their statuses.

Equipment	Status	Events	Events[/s]	Data[MB/s]
Fixed Probes	Fixed Probes@g2field-fe2-priv	398	0.3	0.995
TrolleyInterface	Frontend stopped	0	0.0	0.000
GalilFermi	Frontend stopped	0	0.0	0.000
Surface Coils	Frontend stopped	0	0.0	0.000
Monitor	Frontend stopped	0	0.0	0.000
Fluxgate	Frontend stopped	0	0.0	0.000
PS Feedback	Frontend stopped	0	0.0	0.000

The bottom screenshot shows the 'Logging Channels' section, displaying a table with columns for Channel, Events, MiB written, Compr., and Disk level.

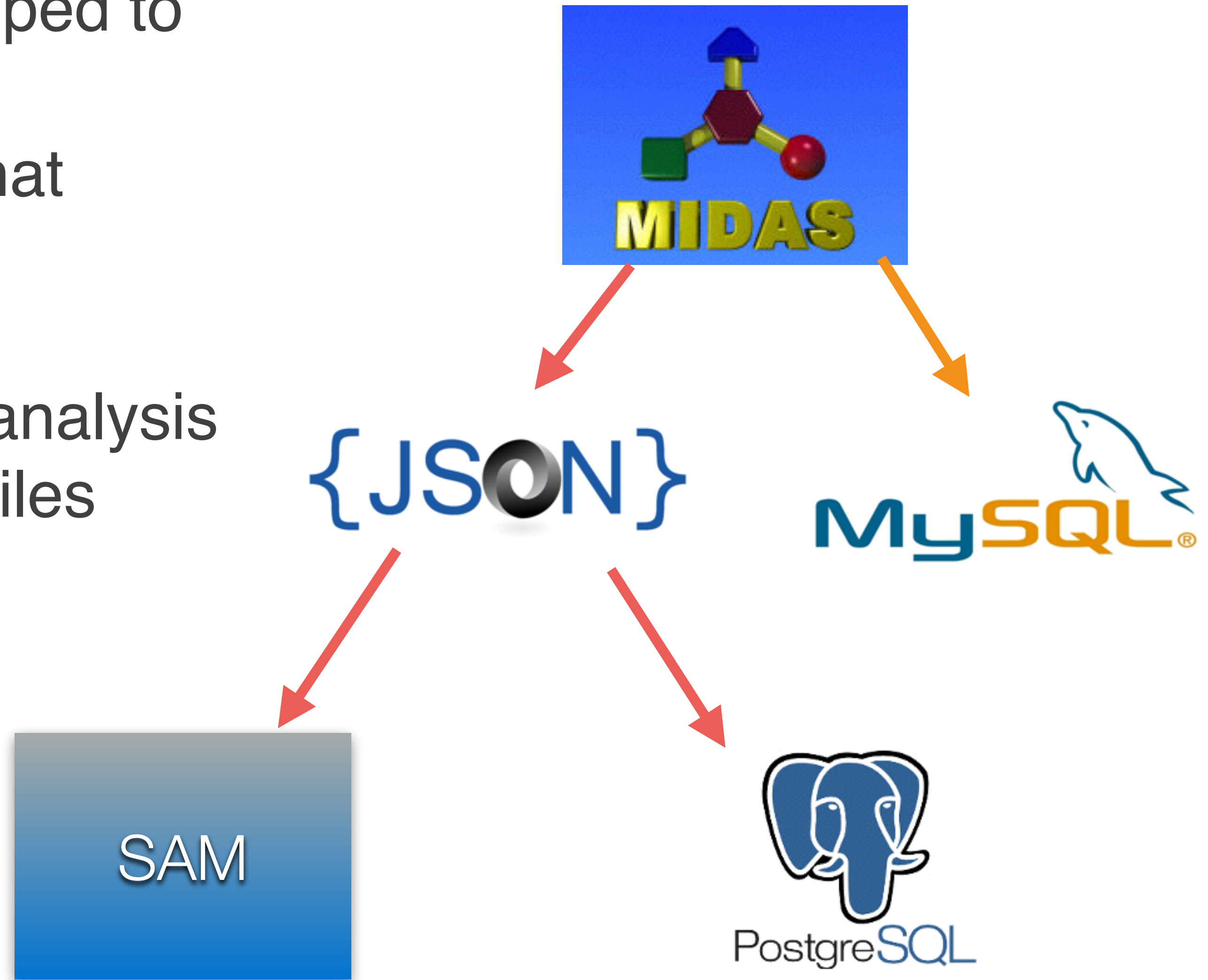
Channel	Events	MiB written	Compr.	Disk level
#0: run00395_00.mid.gz	399	949.078	N/A	17.1 %

The image shows the 'Muon g-2 DQM Run395 Event5 Subsystems' interface. It features a central circular diagram of the ring yoke probe with segments labeled Yoke A through Yoke J. To the right of the diagram are several yellow status boxes: 'Average Frequency: ---', 'Healthy Probes : 324', 'Front-end Online', 'Run in Progress', and 'Trolley Run'. On the left, there are control options for 'Update Options' (Refresh, Automatic On, Every 5 seconds) and 'Display Options' (All Probes).

MIDAS ODB Archive



- At each end of run, the ODB is dumped to a JSON file.
- A python routine is then executed that imports the entire JSON file into a PostgreSQL database.
- Metadata that is used in the offline analysis is also extracted from these JSON files using python plugins.



MIDAS Alarm System



- The MIDAS alarm system is used as the primary alarm system.
- Alarms are set on temperatures and voltages from slow control devices.
- Other slow frontends set alarms automatically when encountering an error.
- Periodic alarms reminded shifters to perform shift checks.
- Each alarm sounds an audible beep and speaks a message, while also displaying a banner at the top of the main DAQ page.
- Some alarms post automatically to the Fermilab electronic logbook.

The screenshot shows the MIDAS DAQ interface with a navigation bar at the top containing buttons for Status, ODB, Messages, Alarms, Programs, Sequencer, Config, and Help. Below this are buttons for Test, Restart Fast Frontends, ChanMap, Straw Tracker Power, Straw Tracker Settings, and WFD5. Two prominent banners are displayed: a yellow one for a 'Warning: Computer room temperature high' and a red one for an 'Alarm: Computer room temperature too high', both with 'Reset' buttons. Below the banners is a 'Run Status' section for 'Run 2075 Stopped'. It shows 'Start: Fri Jul 14 23:39:34 2017' and 'Stop: Fri Jul 14 23:34:51 2017'. A red box on the left contains 'Run 2075 Stopped' and a 'Start' button. To the right, there are green buttons for 'Alarms: On' and 'Restart: Yes'. Other fields include 'Experiment Name: GM2', 'Data dir: /data2/gm2', 'git hash:', and 'CCC Run State: Idle'. At the bottom, a green banner displays the message: '20:28:42 mhttpd:Alarm: Computer room temperature too high'.

Online Databases



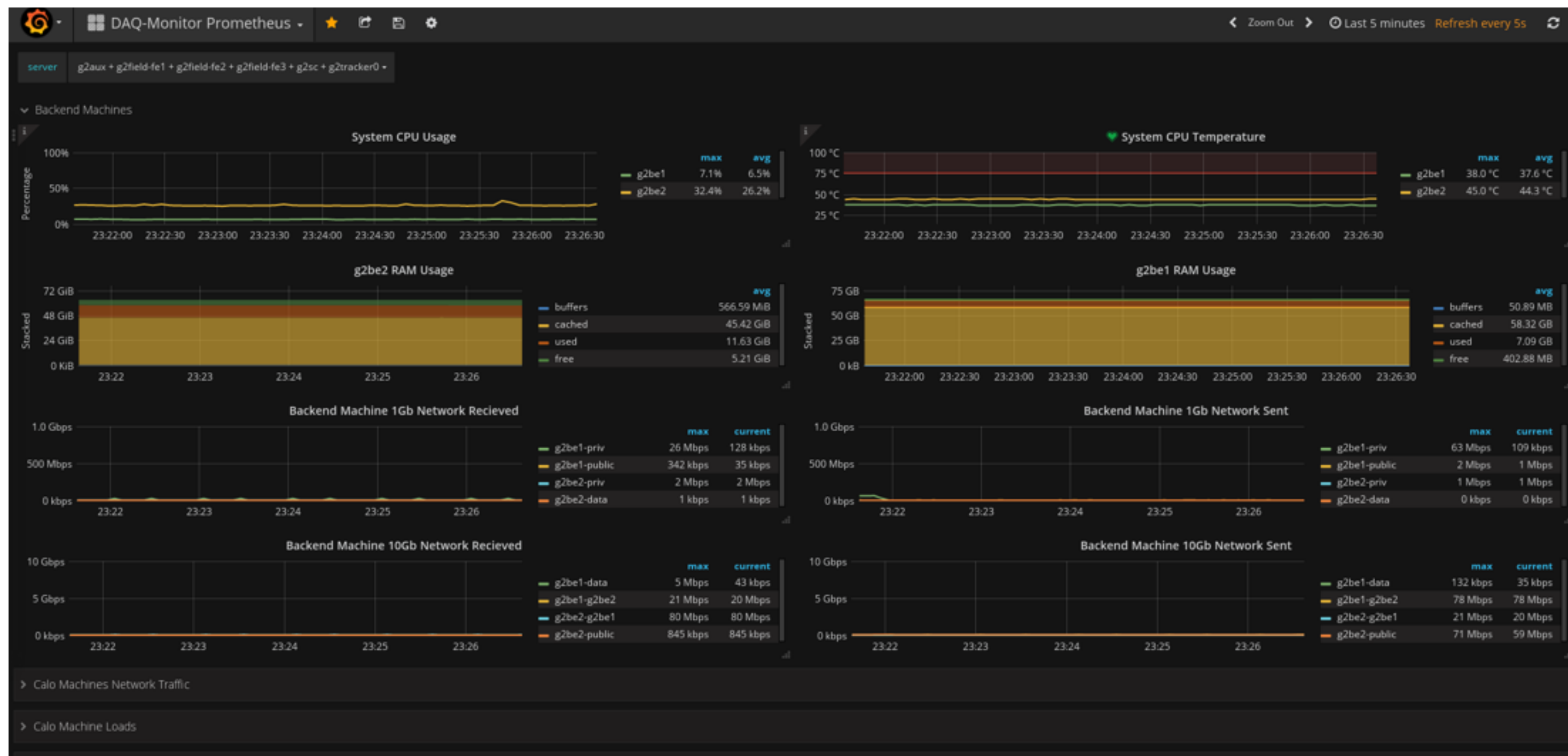
- Slow control data is stored in Postgres database.
- Entire MIDAS configuration is exported to separate table in Postgres database after each run.
- Postgres database is stored locally on 10 TB RAID
- Real-time backup to FNAL server.
- Plan to add redundant local database server.
- Access via custom Django-based web display.

Run	Start time	Stop time	Quality	Shift	Events	Run comment
4135	2017-02-03T17:41:15	2017-02-03T17:47:59	T	das	4558	CCC data format test
4134	2017-02-03T17:25:42	2017-02-03T17:27:24	T	das	1093	CCC data format test
4133	2017-02-03T17:24:27	2017-02-03T17:25:06	T	das	100	CCC data format test
4132	2017-02-03T17:21:59	2017-02-03T17:22:14	T	das	4	CCC data format test

DAQ Health Monitor



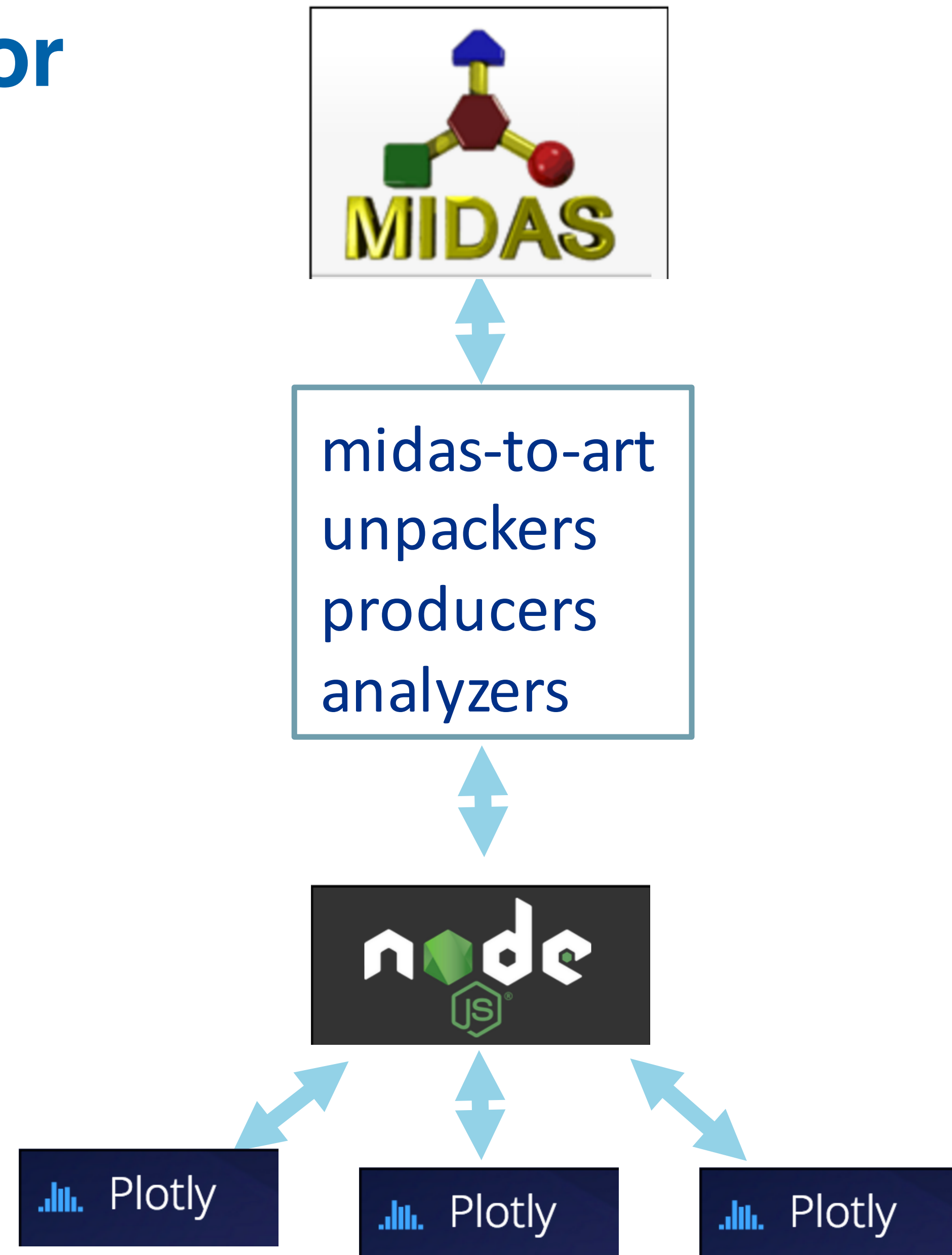
- Monitored health of DAQ systems using netdata for system monitoring, prometheus for short term data storage, and grafana to display data.



- Sets alarms and notifies experts when values go out of range.
- Satisfies recommendation from 2016 computing review.

art-based Data Quality Monitor

- Online DQM streams data from MIDAS mserver.
- Uses existing art modules to process the data.
- Data is extracted from the art job using ZeroMQ and published to a web GUI using node.js and plotly.



Online monitoring of the experiment

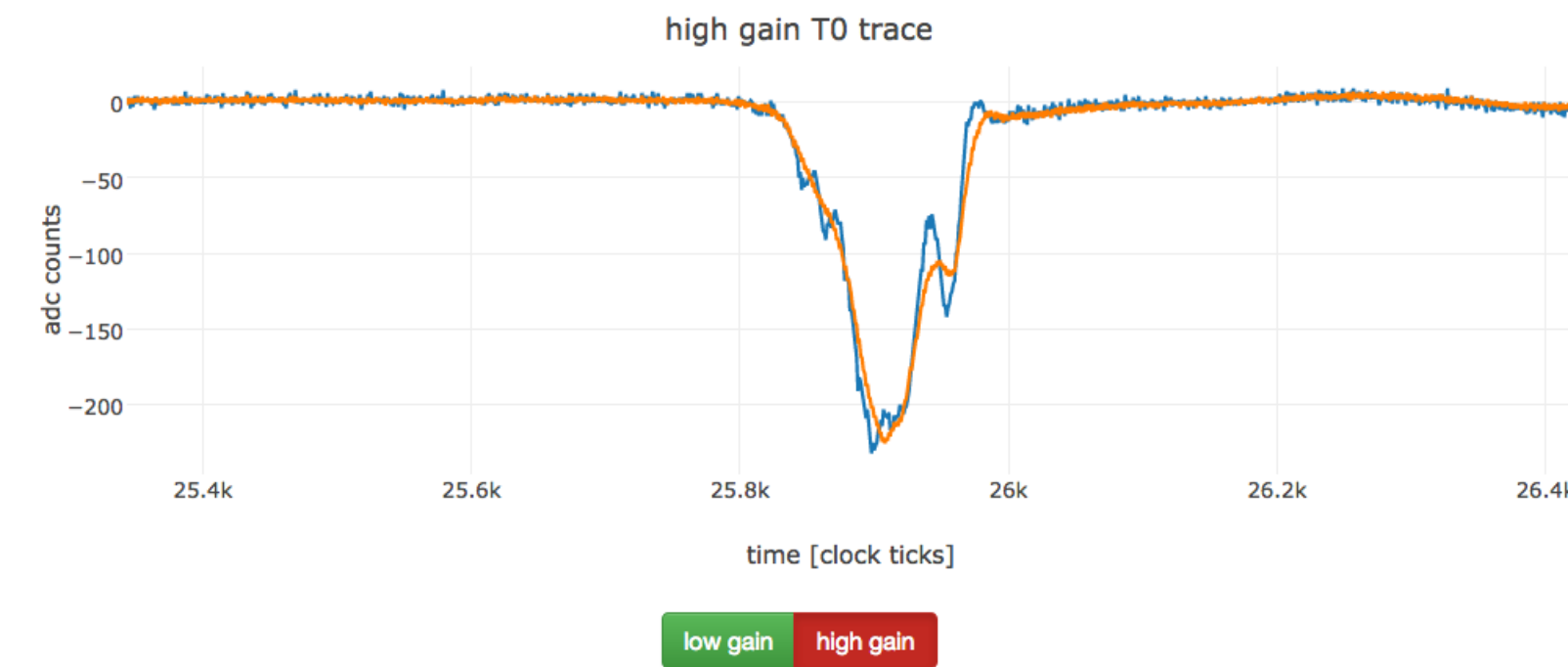


Muon g-2 DQM Run 1911 Event 348 100% of events processed Subsystem ▾

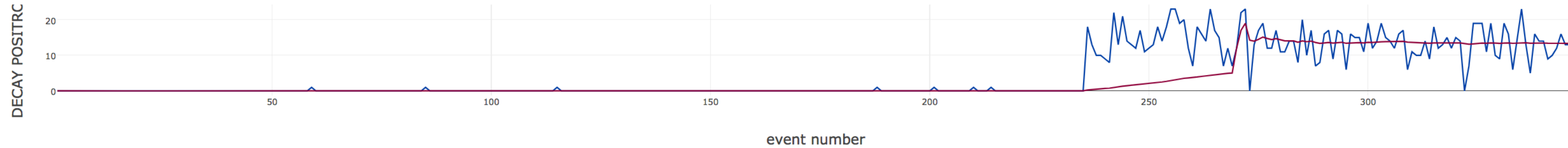
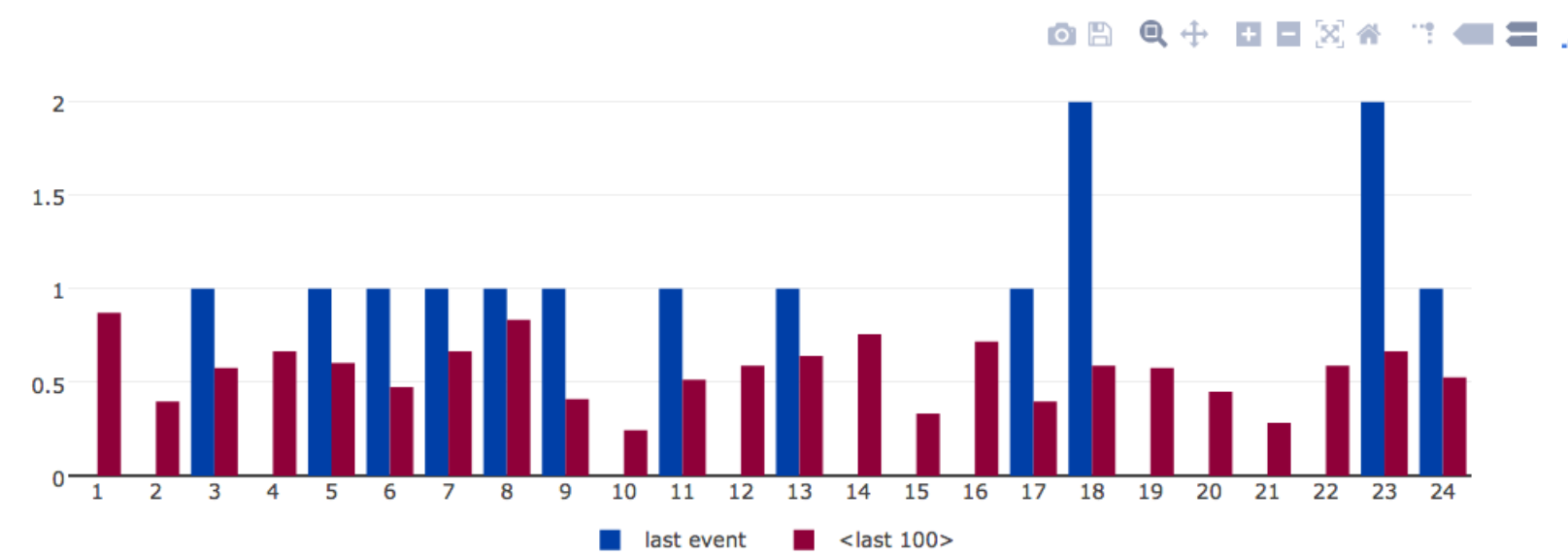
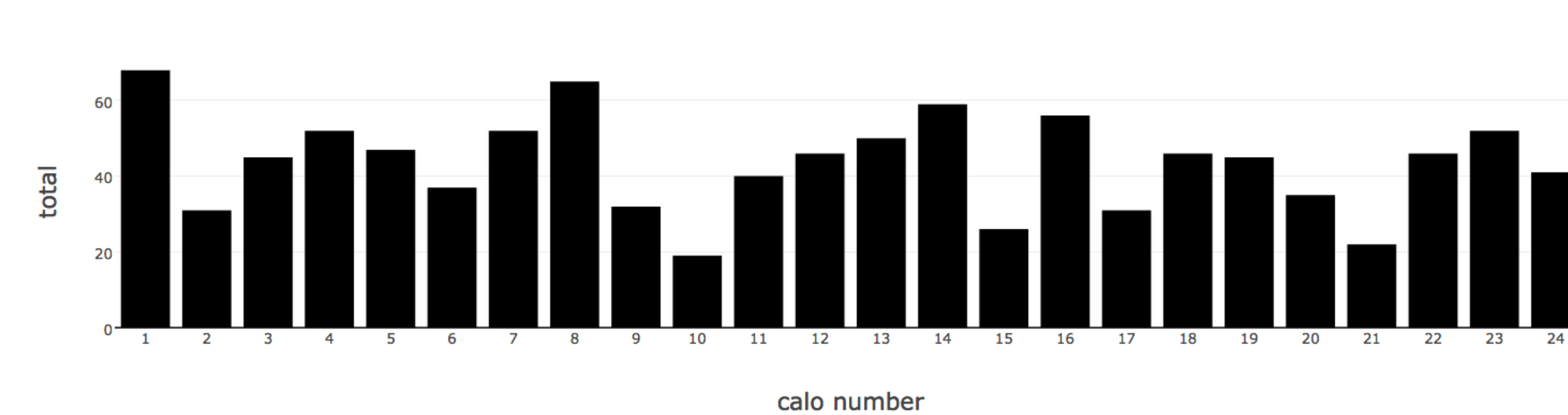
ctag islands daq recon kicker timing Select Calo ▾ **Connected**

CTAG

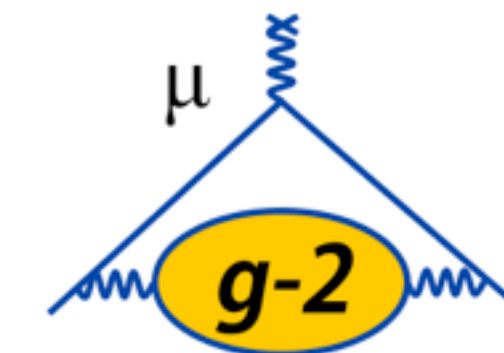
clear	NFILLS	TOTAL	LAST	AVG (last 100)
TO INTEGRAL	78	5.03e+6	6.48e+4	6.45e+4
ALL CALO SPLASH	78	50733	666	650.4
ACROSS RING SPLASH	78	3657	49	46.9
ALL CLUSTERS	78	14749	239	189.1
PROTON LAUNCH	78	7189	112	92.2
DECAY POSITRONS	78	1043	14	13.4



DECAY POSITRONS



Position of pulse in calorimeter



Muon g-2 DQM Run 1911 Event 348 100% of events processed

Subsystem ▾

ctag islands daq recon kicker timing

Calo 8 ▾ **Connected**

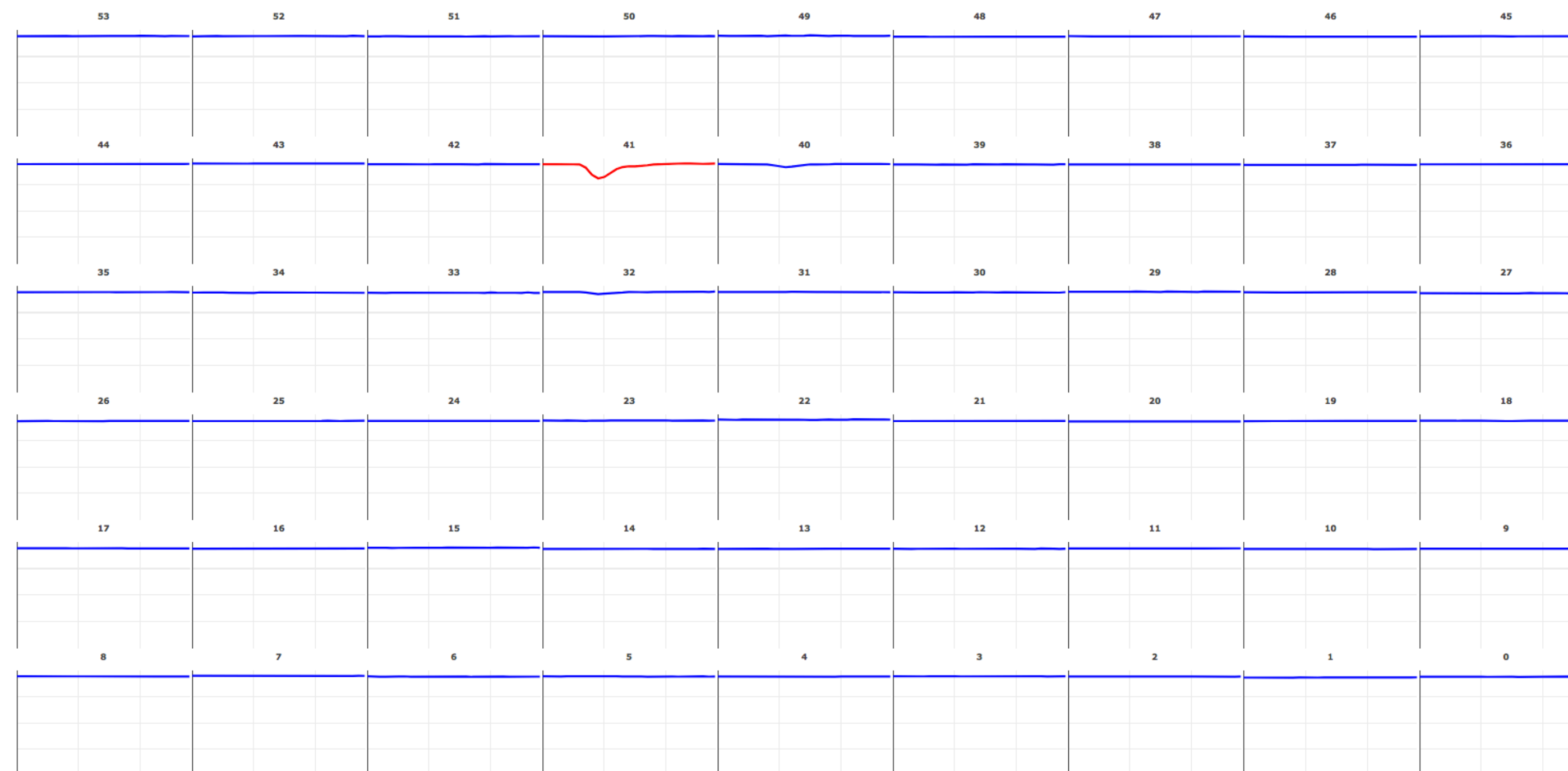
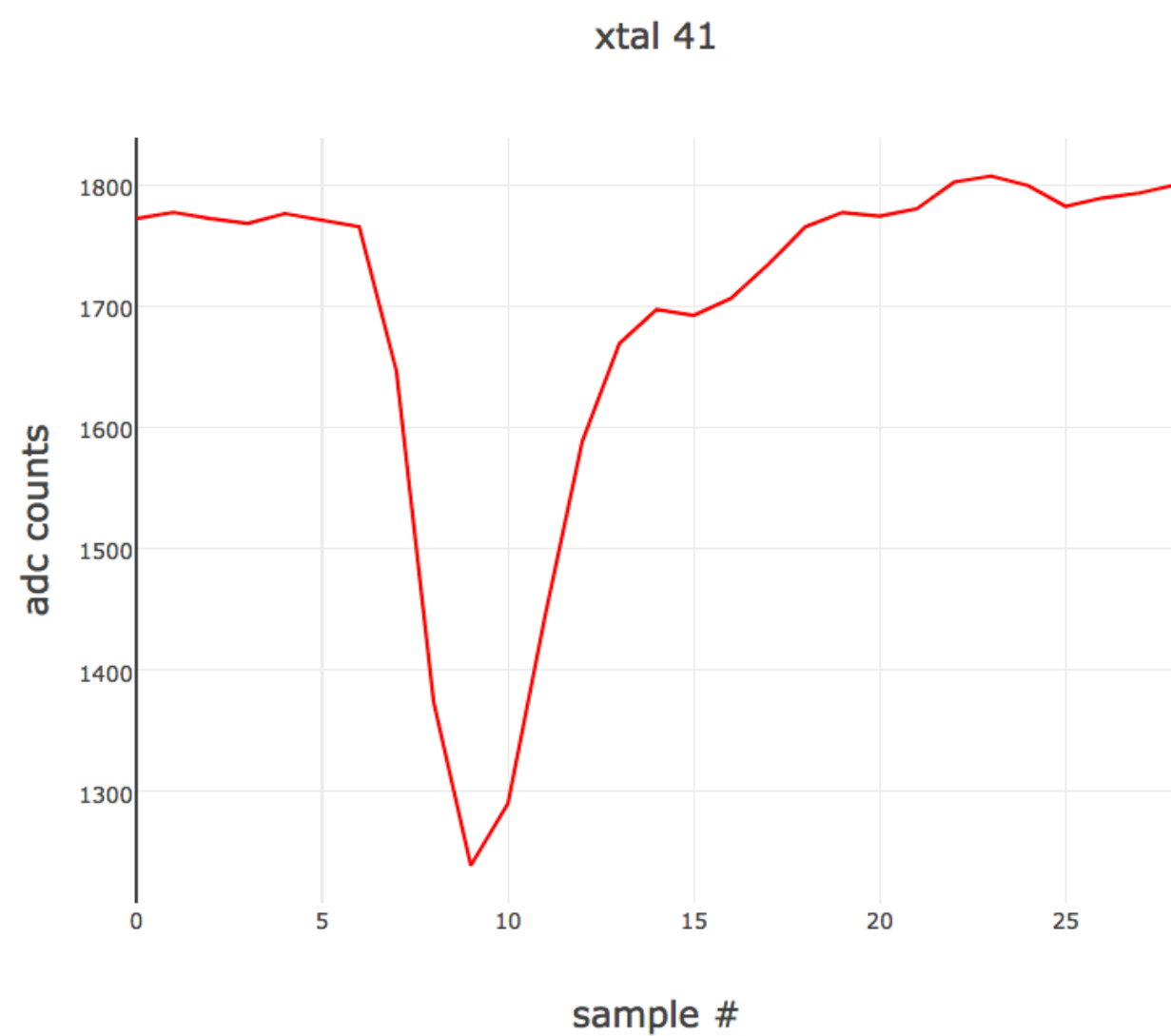
calo 8 traces

RUN 1911 EVENT 347 ISLAND 12

summary traces Q S recon laser headers

auto update: ON **pause** update late ▾

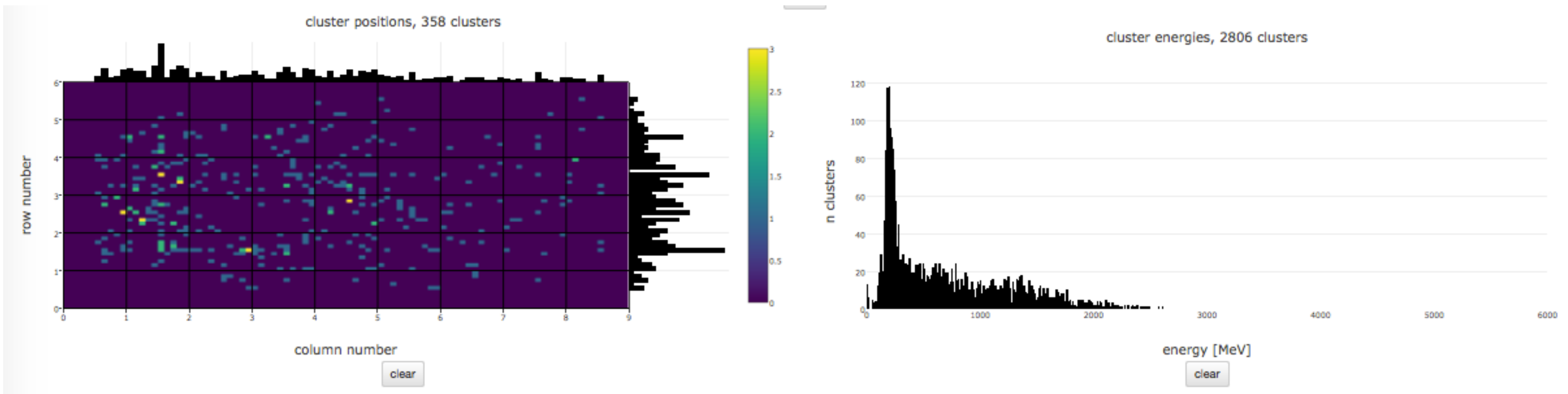
late island: first sample number 124428



Online reconstruction



- The online DQM uses GPU-based template fits to do clustering and energy reconstruction.



Tracker Monitor



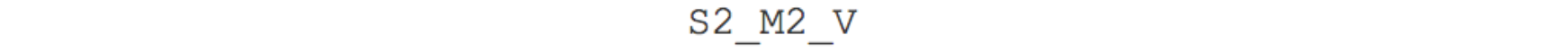
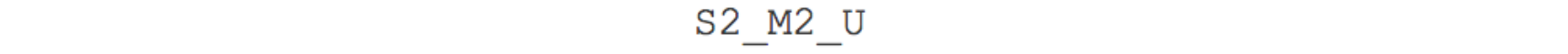
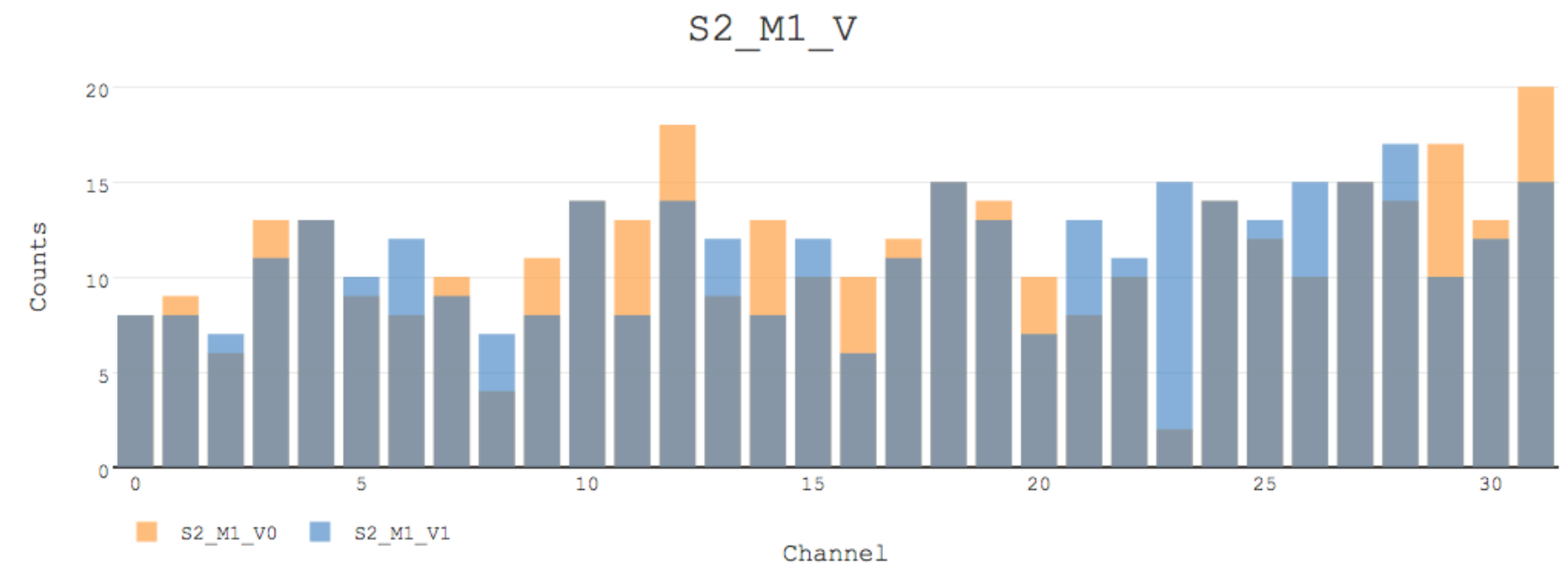
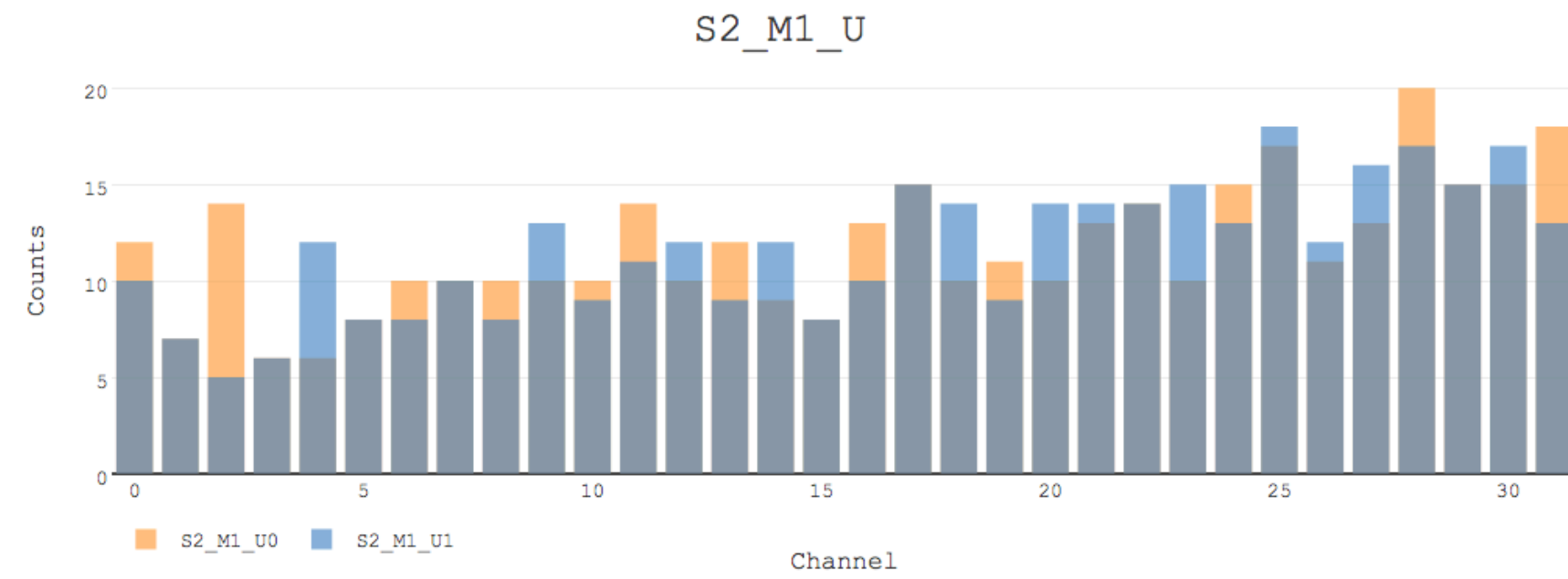
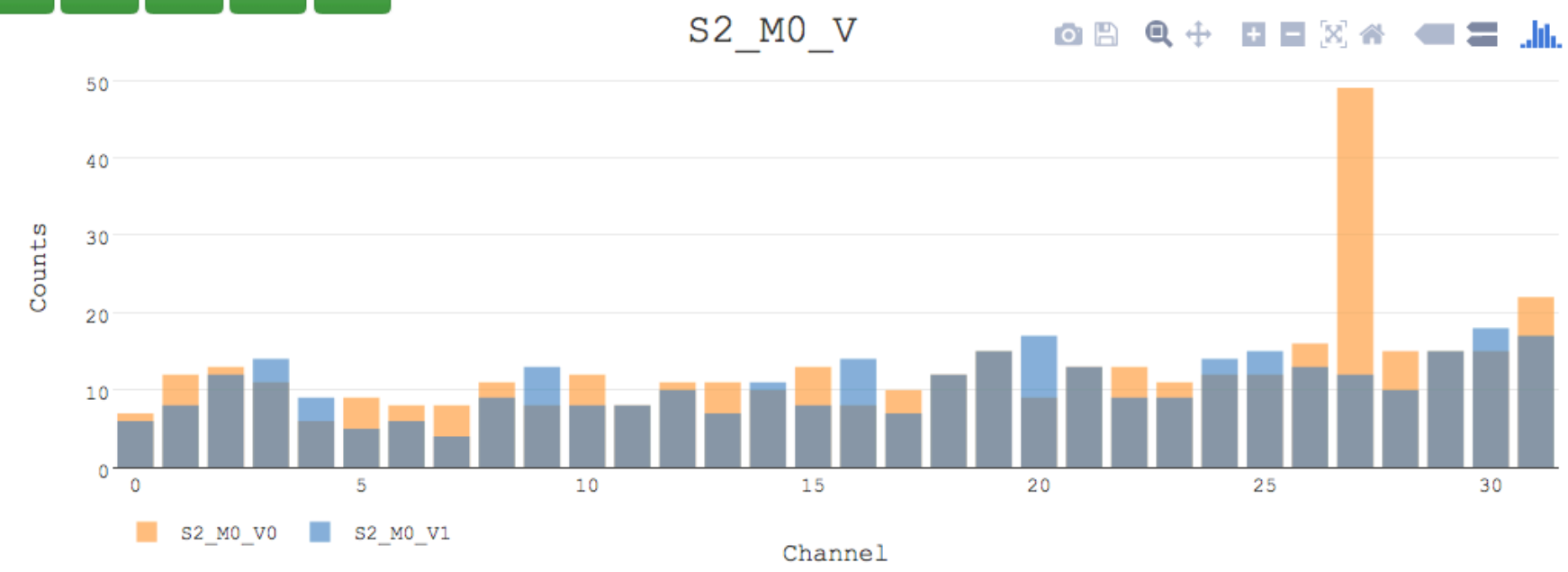
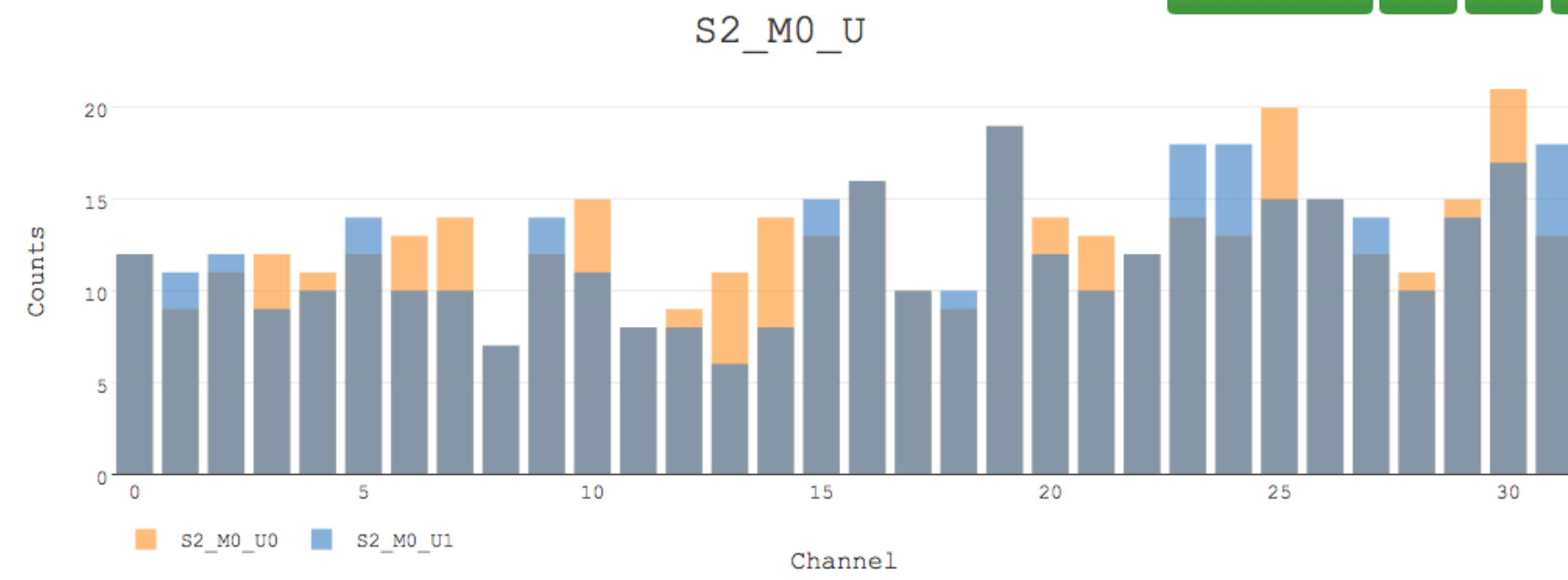
Muon g-2 DQM Run 1912 Event 5 Subsystem -

Straw Tracker Overview Station 2 Physics Plots Connected Clear Histos

Module 0 Straw Tracker Module 9 Fitted 127/128 Straws Present VB12 Not Present	Module 1 Straw Tracker Module 2 Fitted 127/128 Straws Present VB24 Not Present	Module 2 Straw Tracker Module 0 Fitted X/128 Straws Present	Module 3 Straw Tracker Module 1 Fitted 127/128 Straws VB14 Not Present
Module 4 Straw Tracker Module 6 Fitted 128/128	Module 5 Straw Tracker Module 5 Fitted 126/128 Straws UA8, UA17 Not Present	Module 6 Straw Tracker Module 00 Fitted 127/128 Straws (Straw 31)	Module 7 Straw Tracker Module 4 Fitted 127/128

Channels

Tracker 2 INFO M0 M1 M2 M3 M4 M5 M6 M7

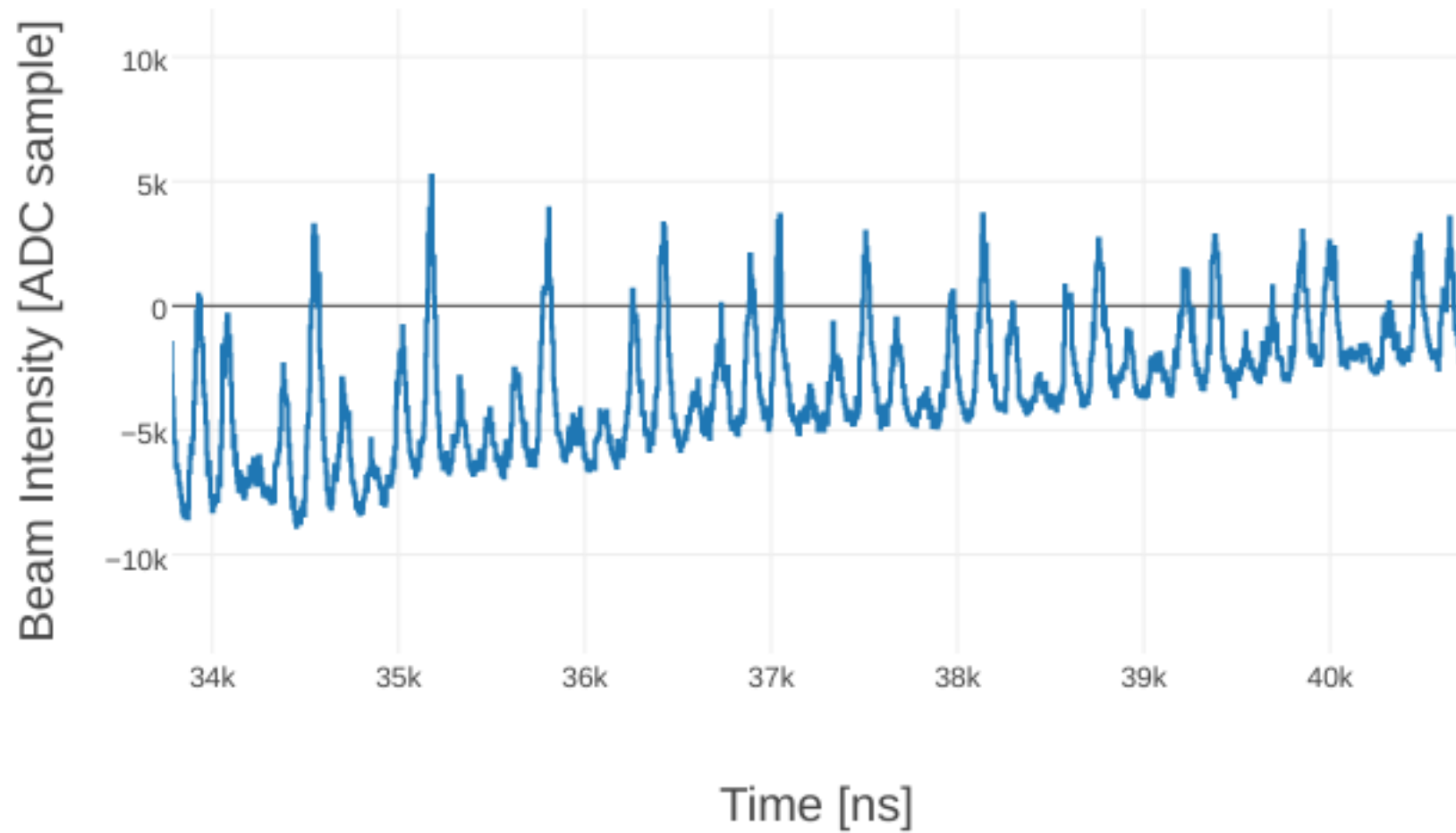


0
1

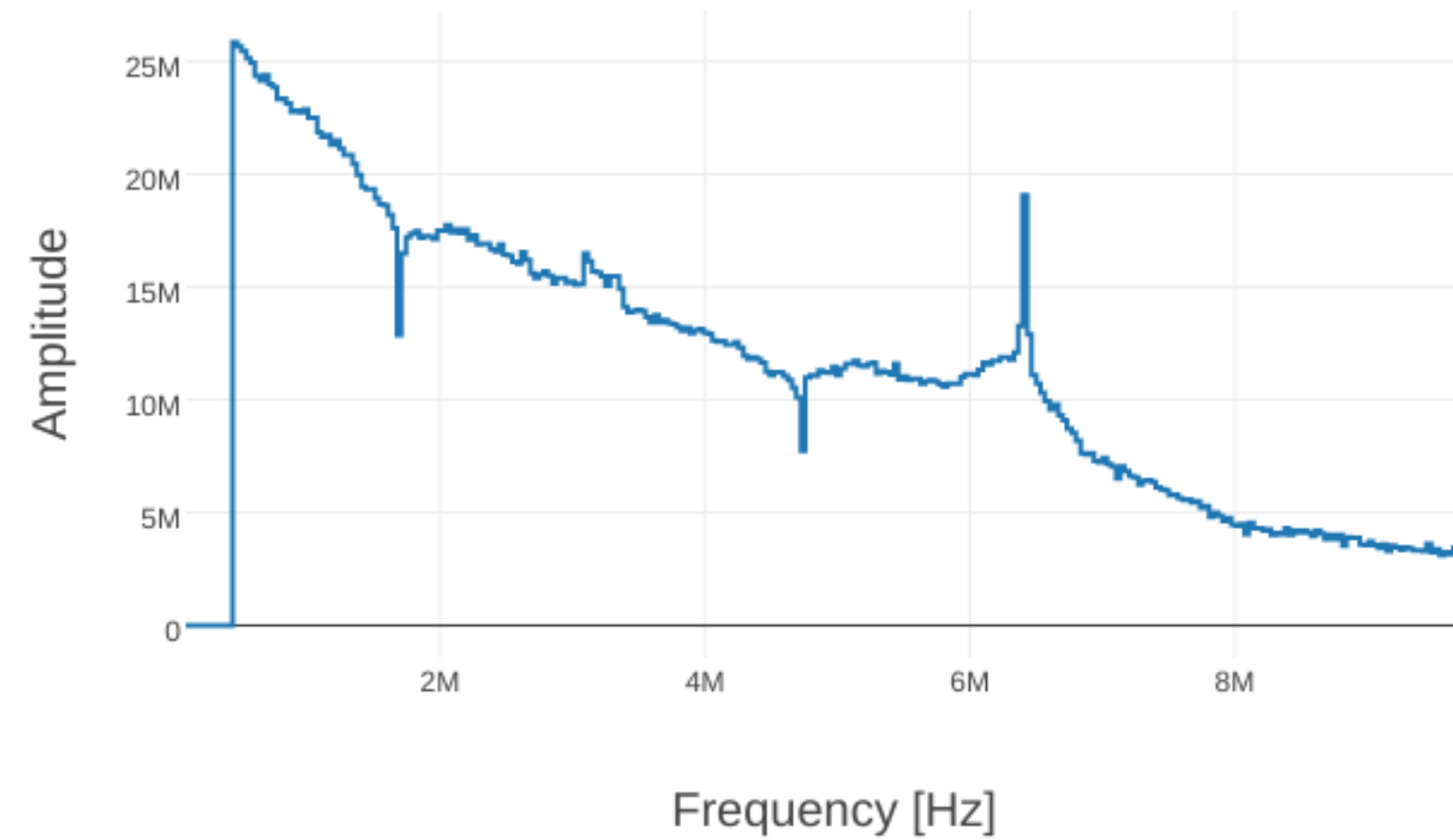


IBMS and Fiber Harps

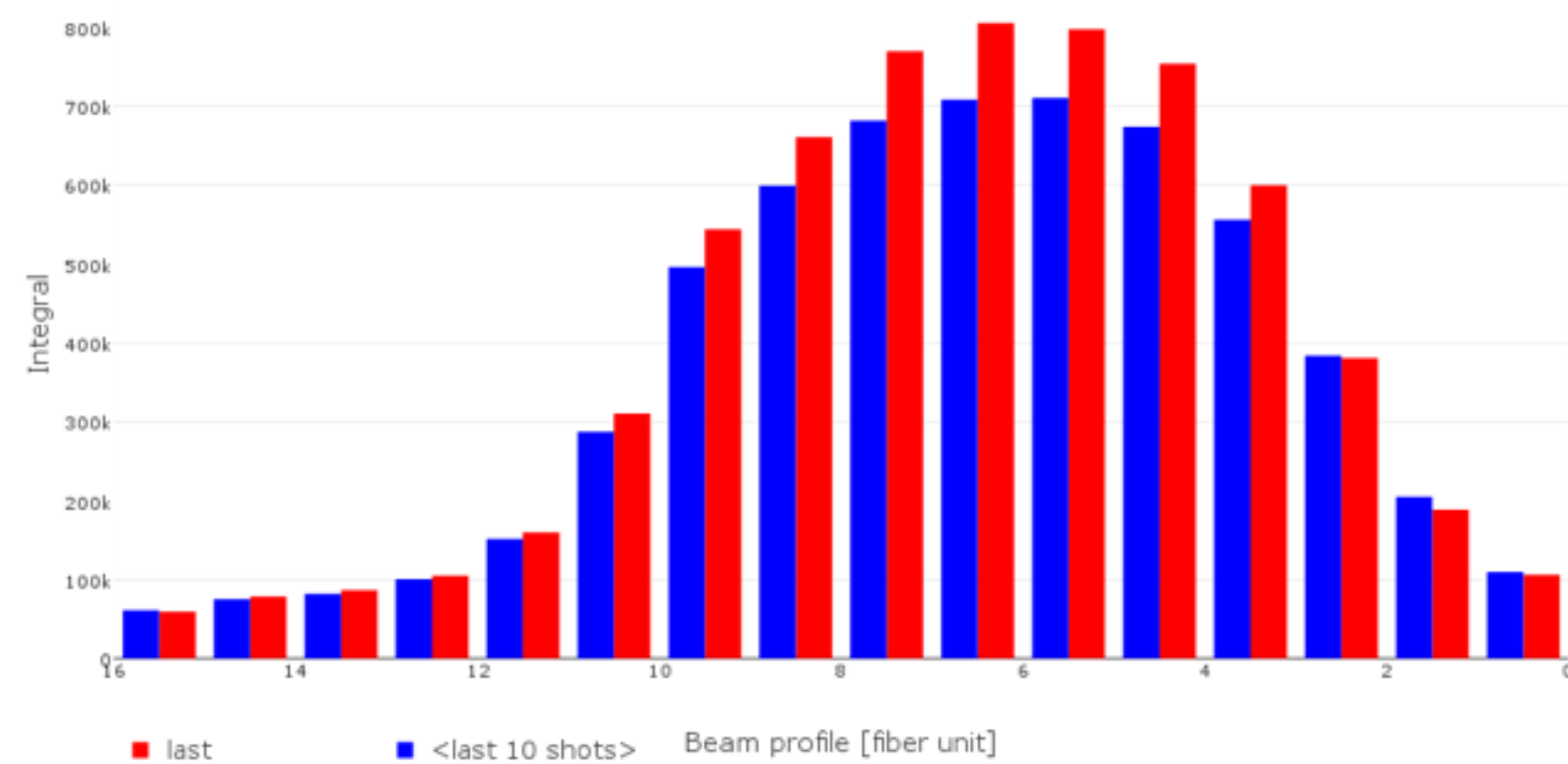
270 deg Y-profile Harp



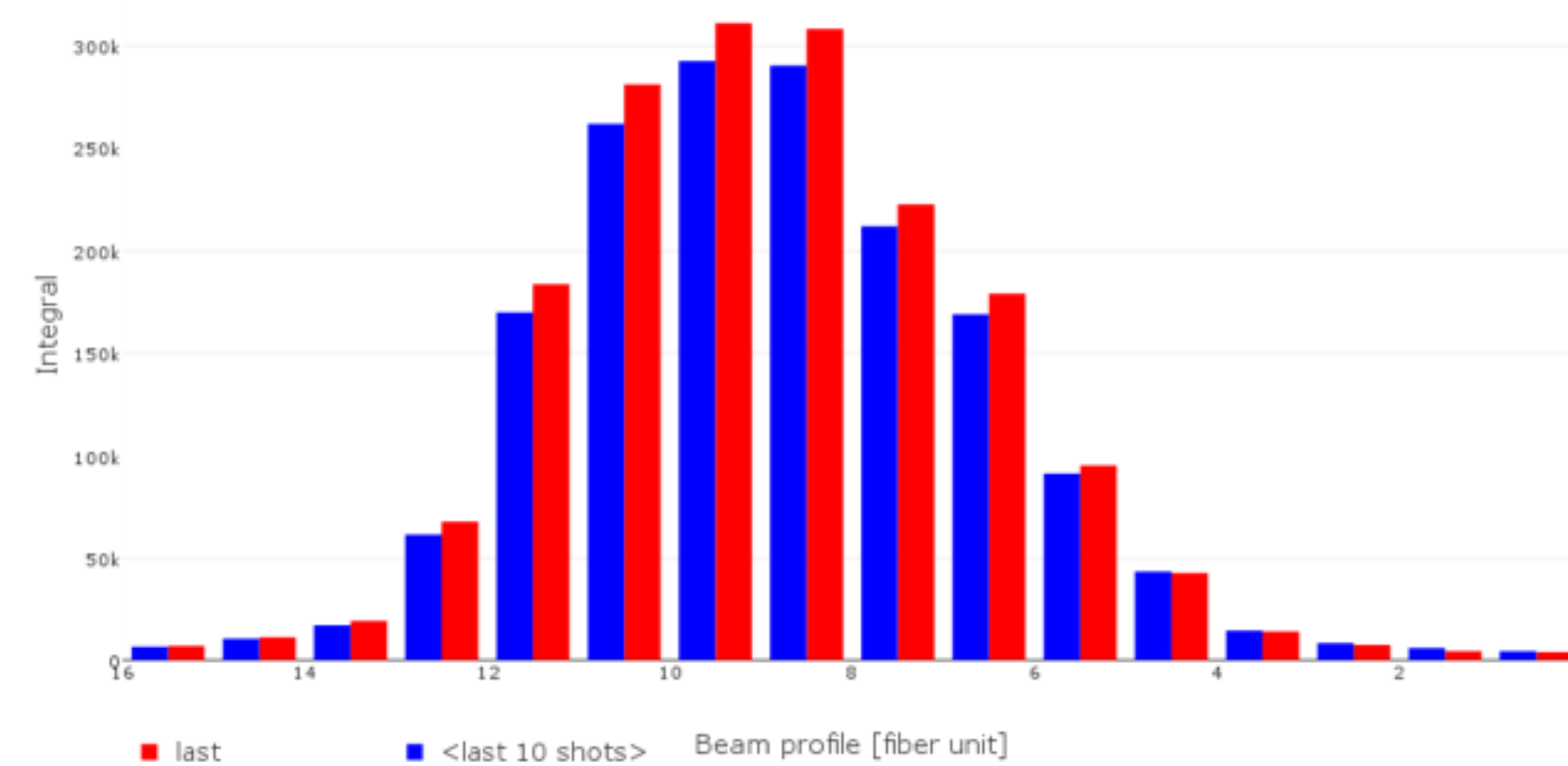
FFT BEAM INTENSITY (Y-profile Harp at 180 deg)



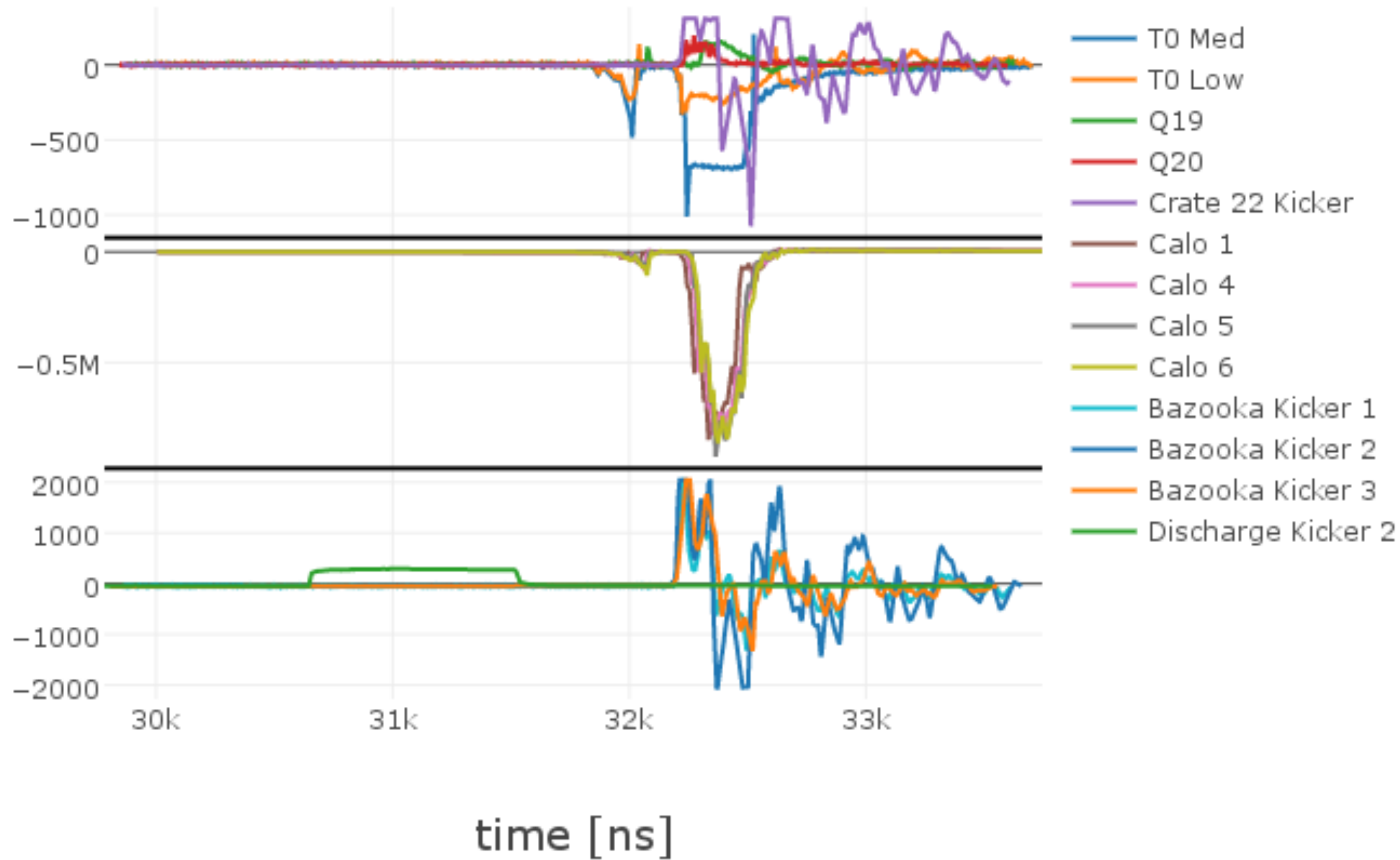
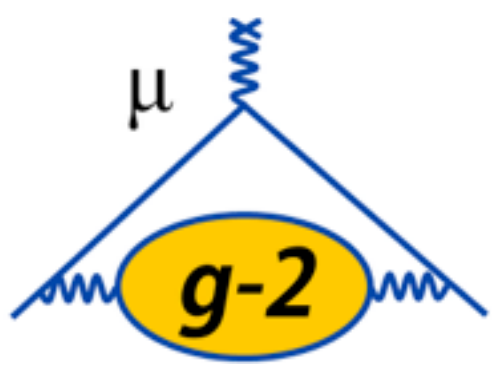
IBMS 1 Y Profile (0 up)



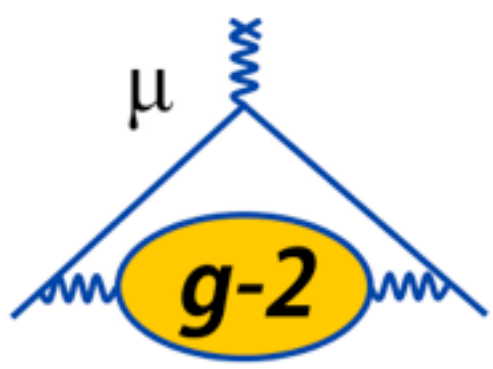
IBMS 1 X Profile (0 radially out)



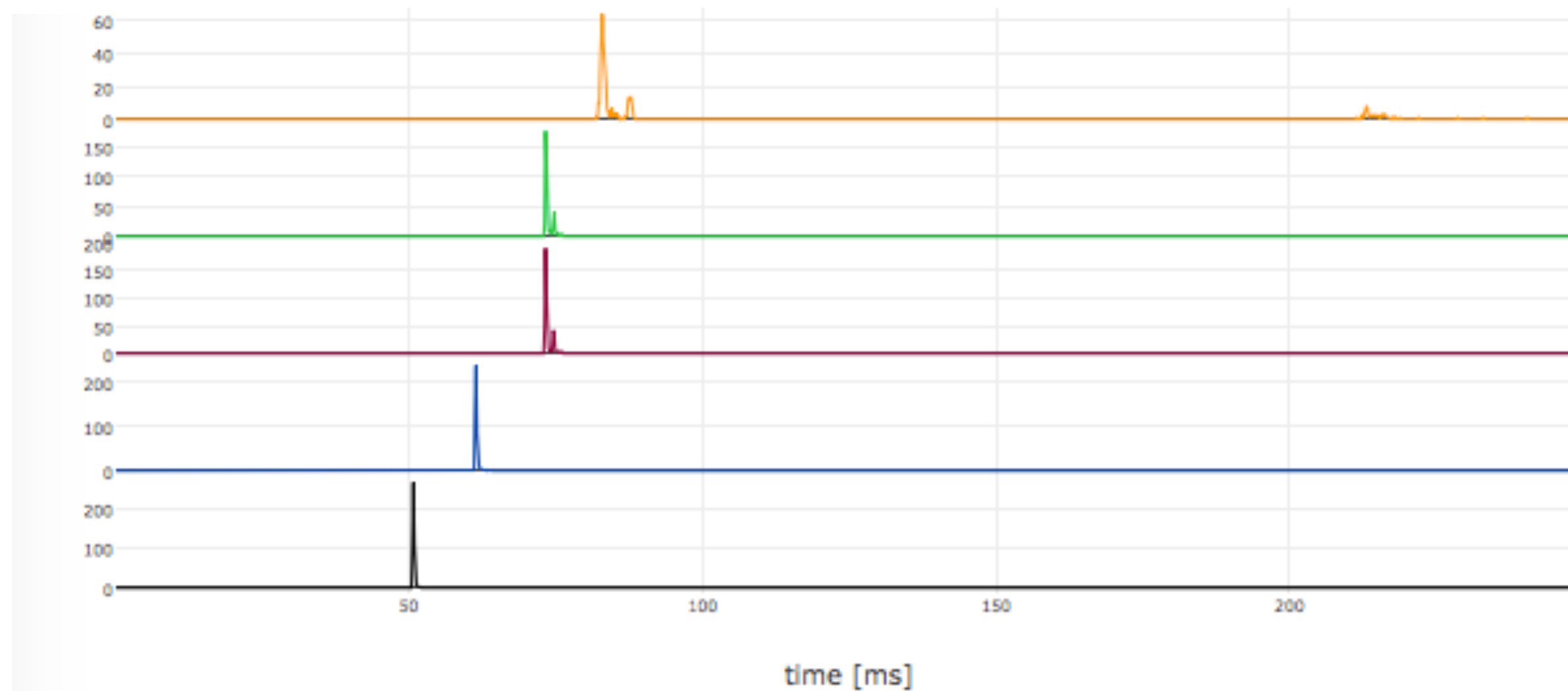
Monitoring Beam Injection



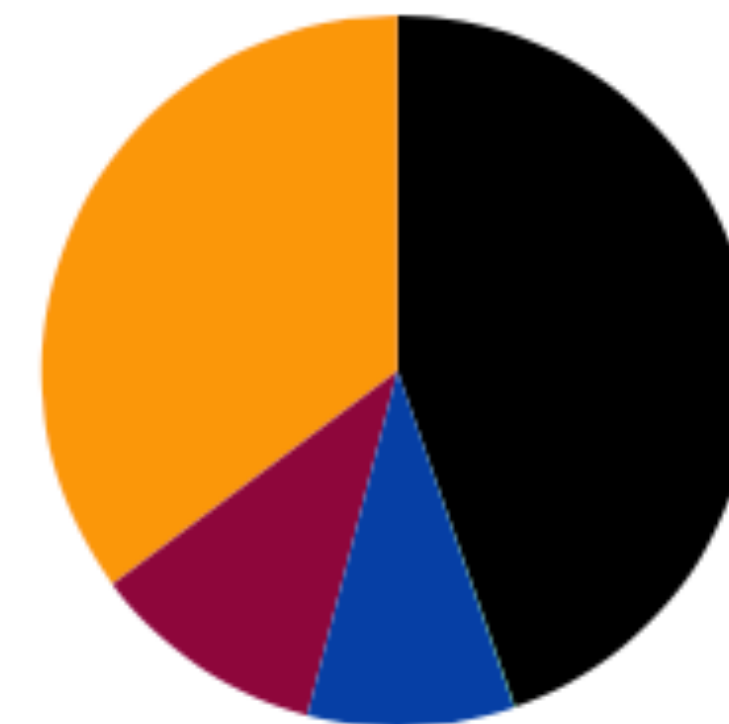
Online DAQ Profiling



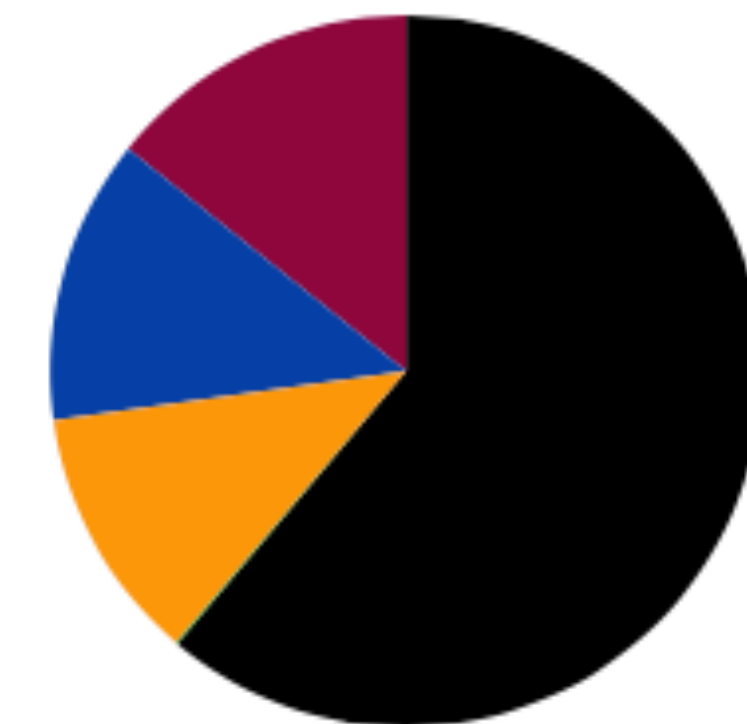
- Must process each event in 83 ms to keep up with average beam rate of 12 Hz.
- Most time is spent reading data from TCP socket and copying it to the GPU.
- Processing time in the GPU is very small.



Average

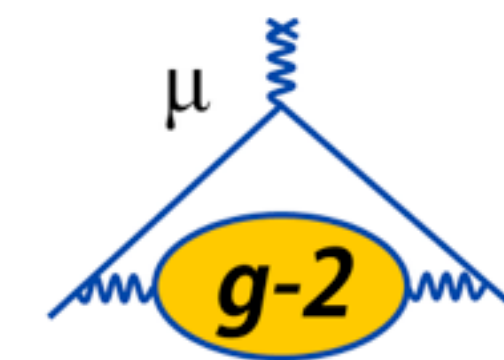


Last Event

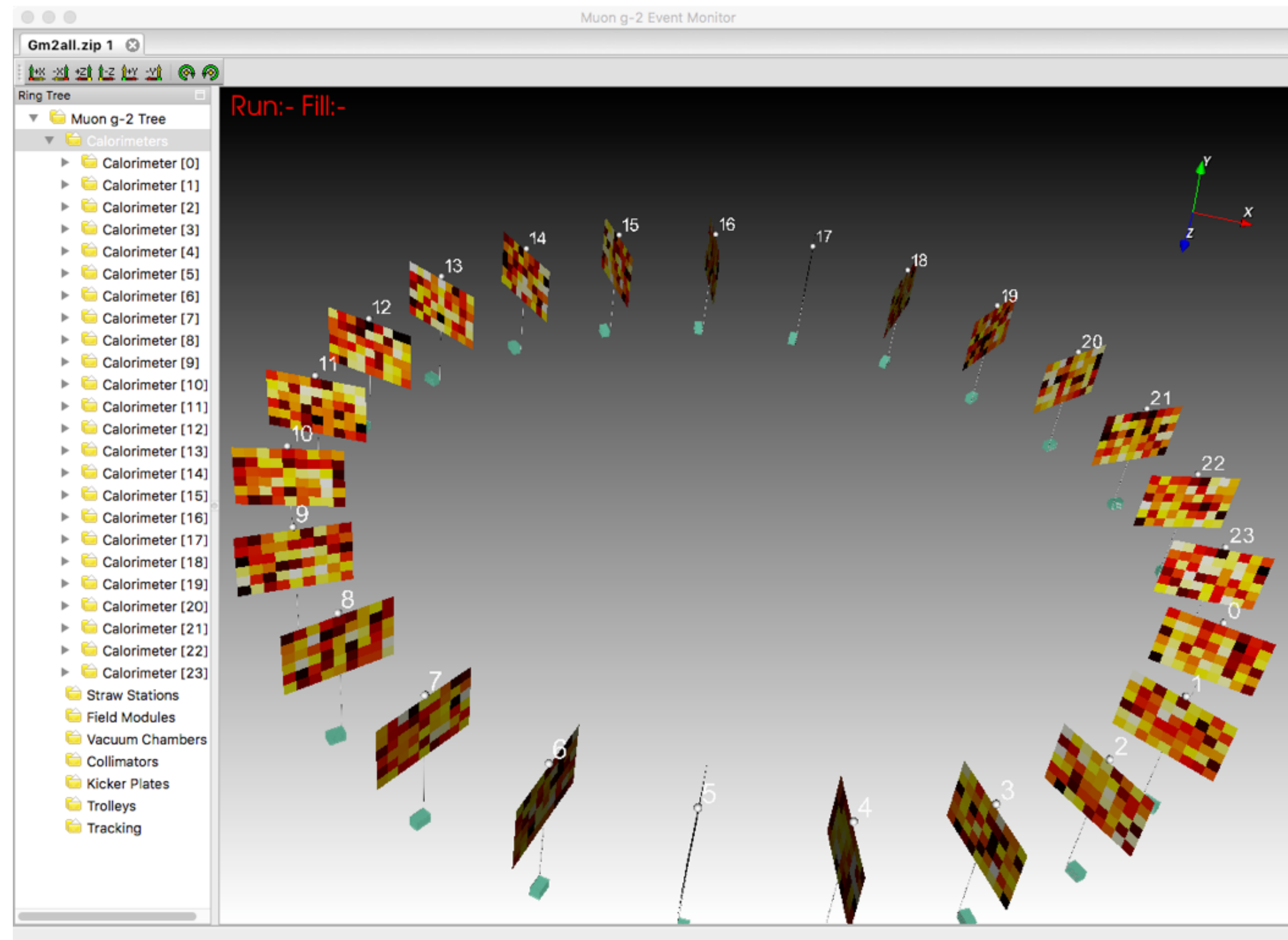


- got tcp data
- compression done
- processed gpu data
- copied gpu data
- mfe start

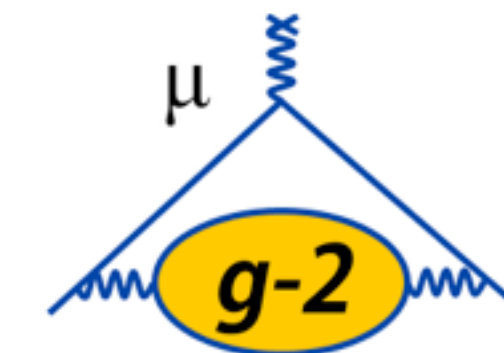
Event Display



- Based on Paraview (vtk).
- Reads data from same *art* jobs as DQM.
- Not ready for commissioning run, but it is being implemented now.
- Currently includes only calorimeters, but plan to add more subsystems soon.



Nearline analysis

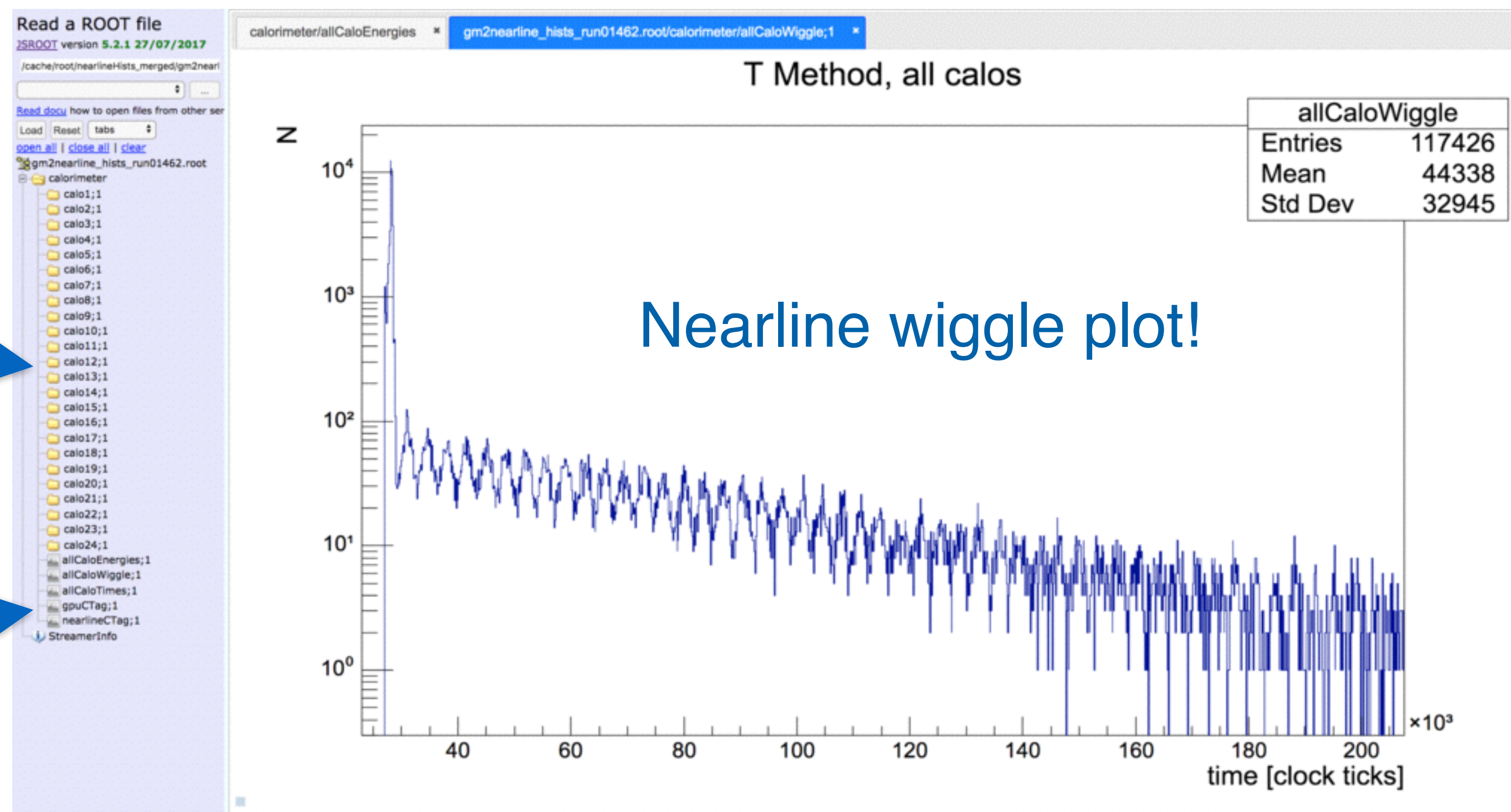


- Two dedicated machines in the control room are used for near line analysis.
- The near line analysis processes data from the most recent run, but performs the full art-based analysis on every run it processes.
- It is allowed to get behind the current run, but so far it can keep up with the live running.
- A set of histograms is made available for each processed run via the Django based web display utilizing JSROOT.

Select run,
detector, and plot
from menu



Plots combining
data from multiple
detectors.



Beam monitoring



- ACNET

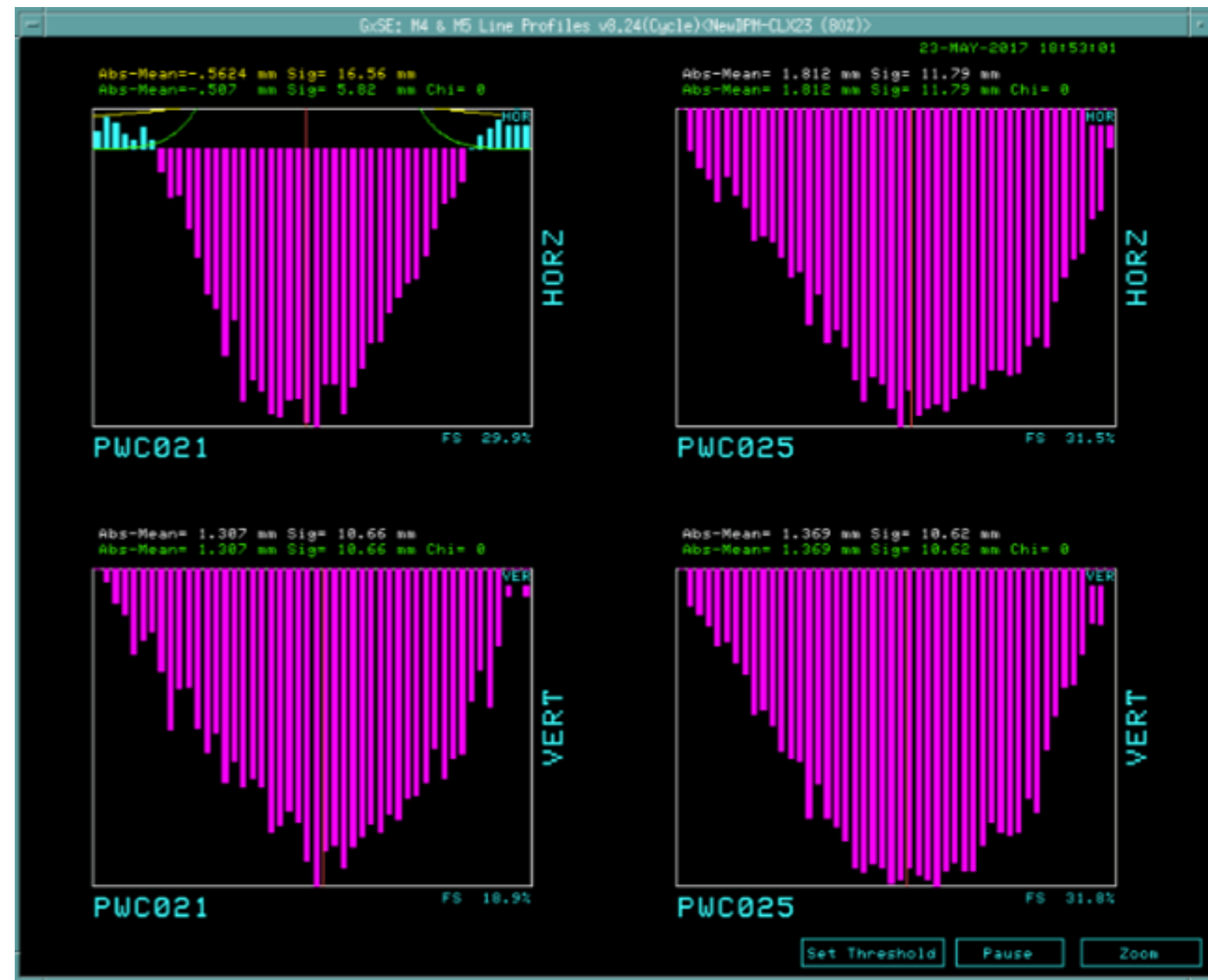
- A dedicated ACNET terminal is in the control room as is used by shifters to observe the status of beam line components.

- IFBeam

- An IFBeam display is kept visible for the shifter on one of the control room monitors to view trend plots of beam line measurements such as protons on target.

- MIDAS Beam frontend

- A dedicated MIDAS frontend reads data from the IFBeam service and writes it in the data stream.

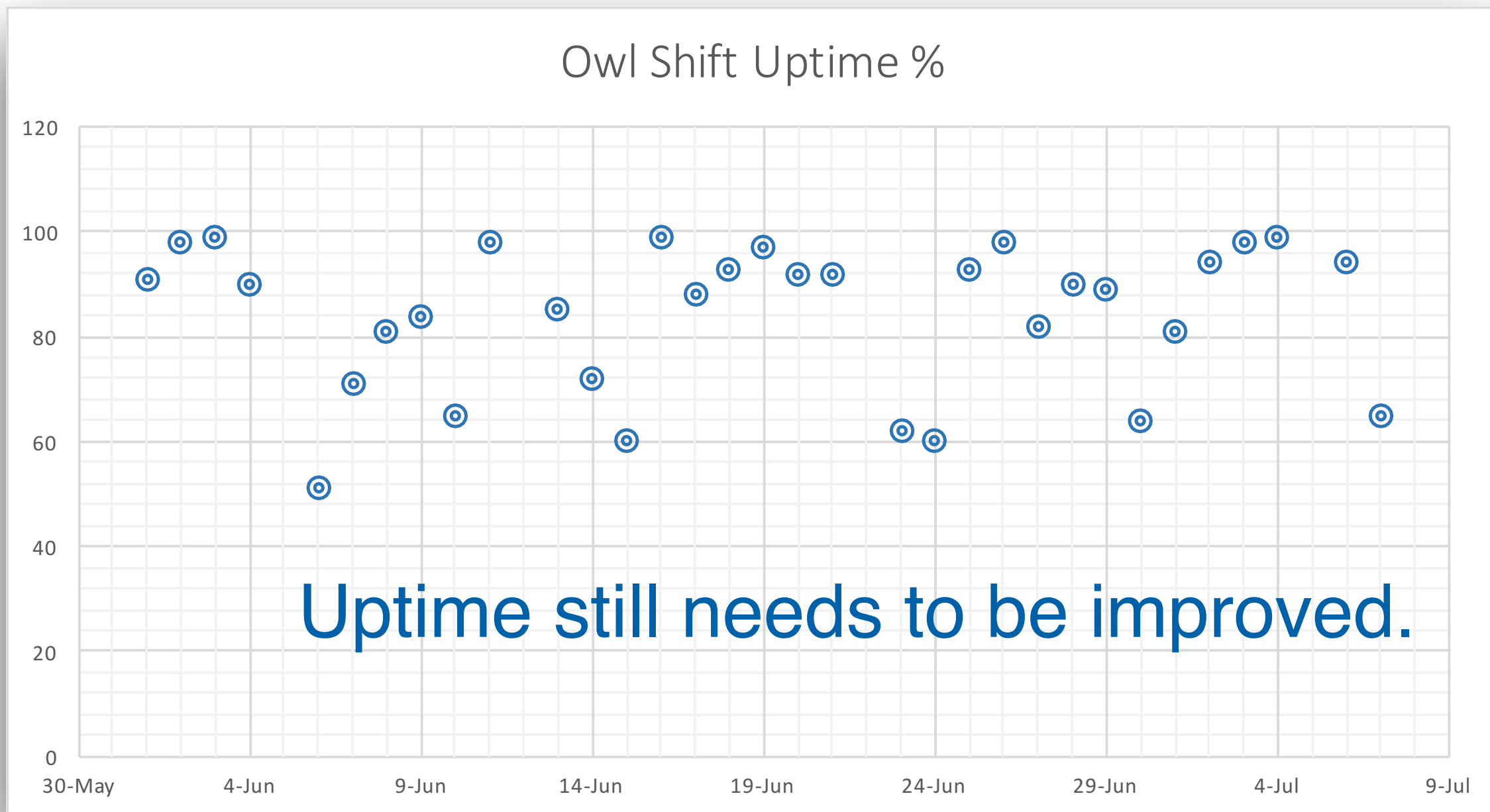
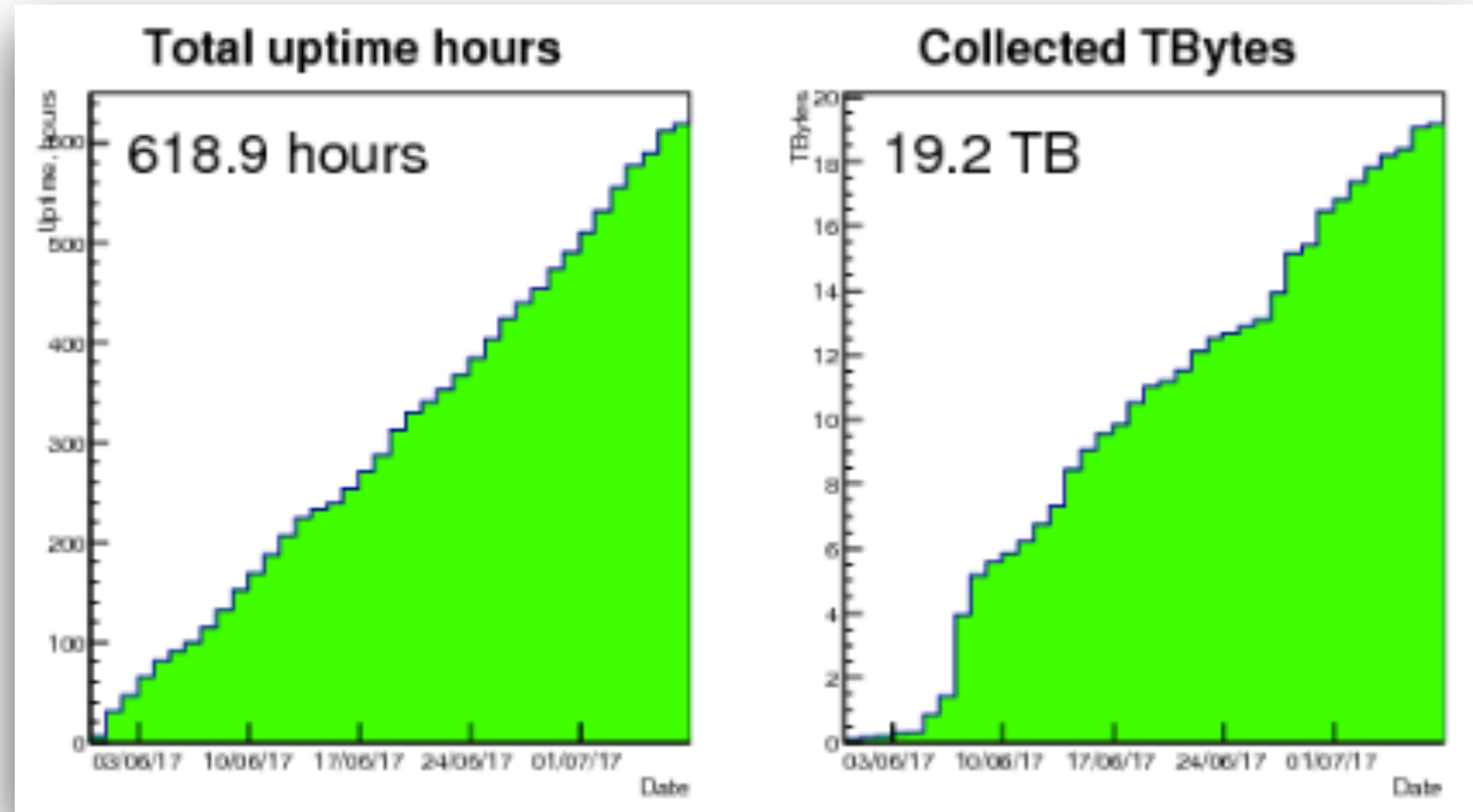


To-do before next run: Update MIDAS frontend to account for relative timing between beam line components.

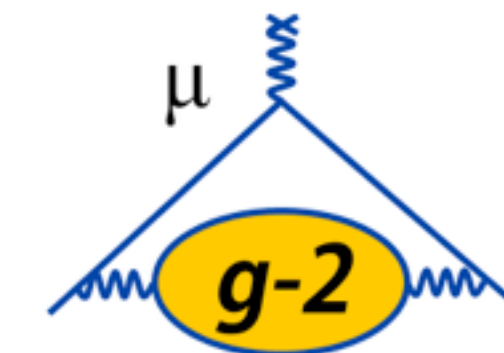
DAQ Performance During Commissioning Run



- The MIDAS DAQ performed well during our first run.
- Day shifts were mostly dedicated to beam line commissioning, so production data was taken at night.



Hardware redundancy plan



- Working with SLAM to develop a detailed plan for the replacement of any machine in the DAQ system.
 - Hot spare for GPU-based frontend machine for calorimeter readout (hot spare was successfully used during commissioning!)
 - Redundant backend systems that could share load temporarily if necessary.
 - Plan to add second database server.
 - IBMS requires custom hardware — only one machine now that can handle it.
- All computers are under warranty.
- Agreement negotiated with Dell on-site support to provide routine maintenance for hardware issues to reduce the likelihood of sending a machine off-site for repair.

Planned hardware upgrades before next run



- Setting up test stand for software development.
- Adding redundant database server.
- Adding dedicated web server with SSO.
- Improving control room hardware (double memory and add SSDs).
- Adding additional control room machine.

Contributing Institutions and Responsibilities



<i>Institution</i>	<i>Responsibilities</i>	<i>Personnel</i>
University of Kentucky	<i>AMC13 Readout, GPS, DAQ Hardware, IFIX integration, Online Systems Management</i>	<i>PI + PD (on site) + GS (on site)</i>
University College London	<i>Tracker readout, Online Management</i>	<i>PI (on site) + PD (on site) + GS (on site)</i>
University of Washington	<i>DQM, IBMS readout, Field, Nearline</i>	<i>GS + 2x GS + PD</i>
Northern Illinois University	<i>Slow controls</i>	<i>PI</i>
Argonne National Lab	<i>Field DAQ</i>	<i>PD (on site)</i>
JINR, Dubna	<i>Web interface, Event display</i>	<i>2x Scientist</i>
Novosibirsk	<i>Django Web Display</i>	<i>PD</i>
University of Michigan	<i>Field</i>	<i>GS</i>
Shanghai	<i>Database management</i>	<i>PD</i>
Italy (Frascati + Napoli)	<i>Laser system integration</i>	<i>PI + PD (on site) + PD + GS</i>
Cornell	<i>Auxiliary detector integration, Clock</i>	<i>PD + GS (on site)</i>
Fermilab	<i>Beamline integration</i>	<i>Scientist</i>

~24 total

RED = DAQ on-call

DAQ On-site support



Level 1 Shifter tries to debug the problem. ← 2 shifters in control room at all times

Level 2 DAQ On-calls are called. ← 2 DAQ experts on-call at all times

Level 3 Local online experts (Wes and Becky) are called.

Level 4 SLAM is paged by Wes or Becky. ← SLAM provides 24/7 support

Level 5 Dell on-site hardware support is contacted. ← Maintenance agreement to have computer hardware serviced locally.

High-rate tests



- During the commissioning run, we had data at ~ 0.1 Hz instead of 12 Hz, and the DAQ performed very well.
- We are using the laser calibration system to test the DAQ at 12 Hz.
- This data is being used for fine tuning our optimum DAQ settings for data collection such as the Q-method decimation and flush rate.
- We have observed that the DAQ typically runs for ~ 5 hours without human intervention, punctuated by TCP/IP errors and WFD5 errors that are fixed by restarting the offending process.
- Efforts to resolve these errors and improve uptime are in progress.

Summary



- MIDAS is working very well as the DAQ software for Muon g-2.
- The DAQ hardware includes 17 frontend machines, 5 backend machines, 2 dedicated near line analysis machines, 3 computers for slow control, 3 servers, and 24 beagle bones running 67 MIDAS frontends.
- We take an input data rate of 20 GB/s, and reduces that to < 200 MB/s in the event builder via GPU processing and lossless compression.
- We have a clear plan for monitoring the beam and data quality and the associated infrastructure has been tested.



backup

Outline

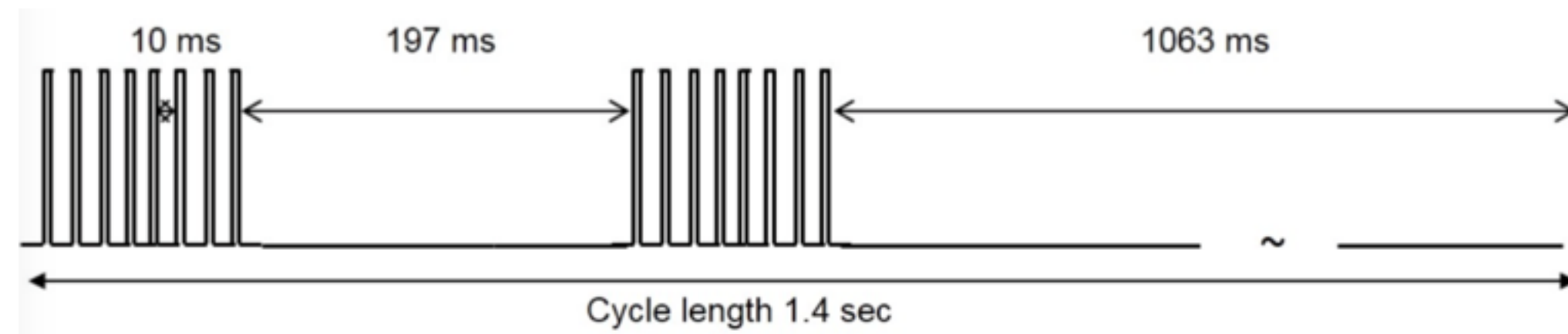
- Overview of DAQ/DQM Systems (and nearline?)
- Alarm system
- Data path and storage
- Slow controls/databases
- DAQ health monitoring
- What works
- What we have done to fix the things that didn't
- Interface between online/offline systems
- High rate tests
- Hardware redundancy plan + DAQ maintenance agreement
- Hardware upgrades planned before next run
- Personnel needed and SLAM contribution



Rate requirements



- Accommodate 12 Hz average rate of muon fills that consist of sequences of eight successive 700 μs fills with 10 ms fill-separations.



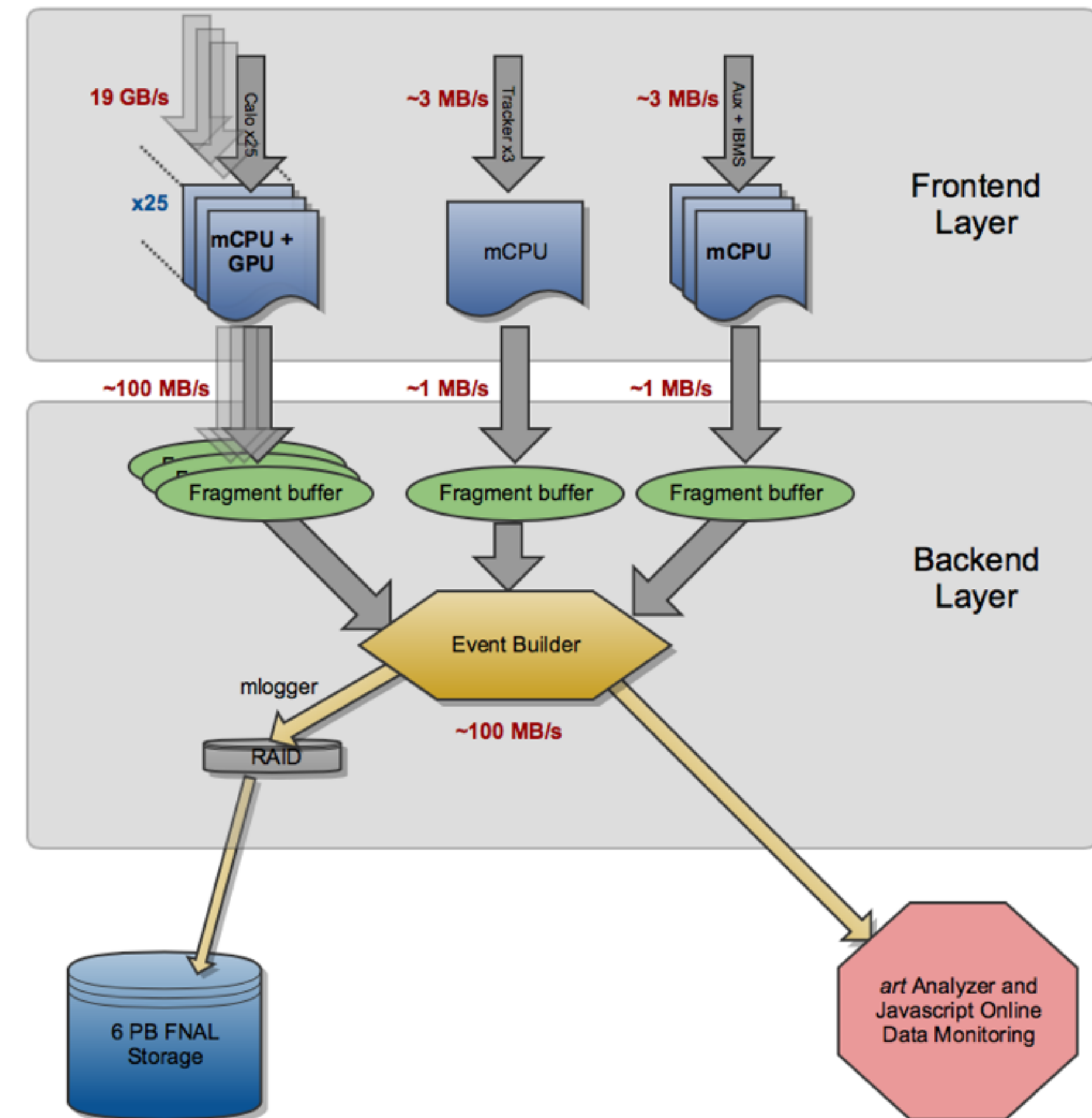
- Time-averaged rate of raw ADC samples is 20 GB/s, which must be reduced by a factor of 100.
- Data is processed in GPUs to accomplish this task.
- Total data on tape after 2 years of running will be 7 PB.

Source	MB Per Fill	MB Per Second
Raw data	1,600	19,400
T-Method	9.4	112.5
Q-Method	4.0	48.5
Prescaled Raw	1.6	19.4
Tracker	0.75	9
Laser Monitor	0.08	1
Auxiliary	0.33	4
Event Builder:	16.2	194.4

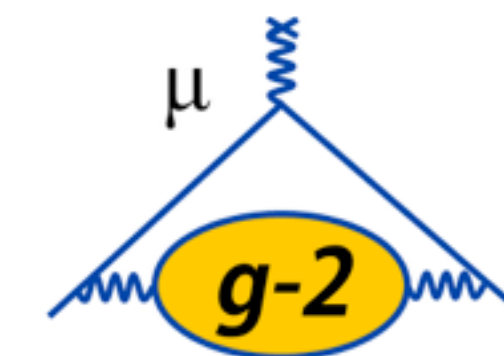
DAQ Design



- Layered array of commodity, networked processors
- Frontend layer for readout of detectors.
- Backend layer for assembly of event fragments.
- Slow control layer.
- Online analysis layer using *art*+JS.
- Field DAQ operates independently, but with a similar design.



Input sources

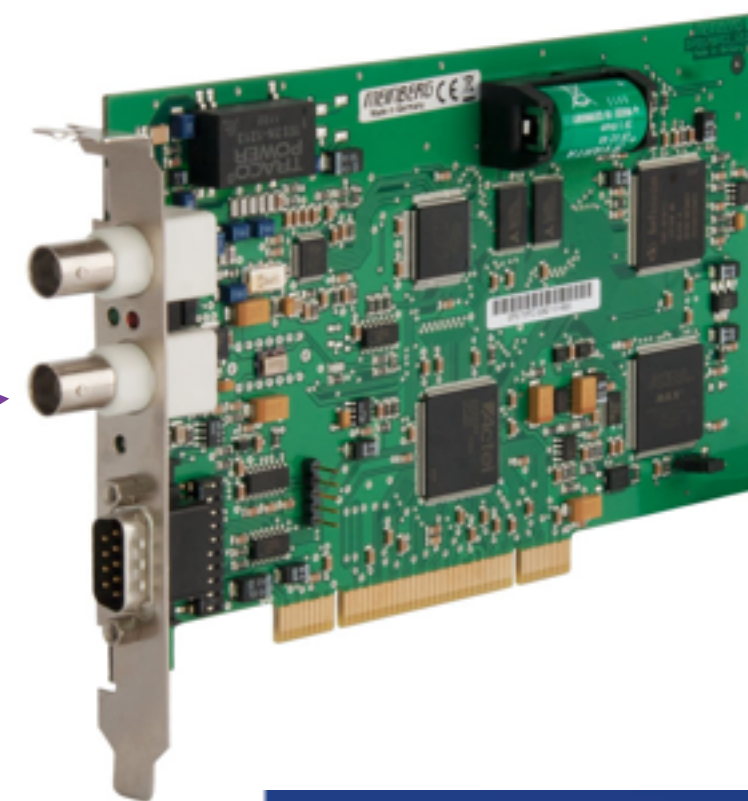


- Digitization is performed in custom uTCA based waveform digitizers.
- Each digitizer runs at 800 MSPS, so each time bin is 1.25 ns, and a 700 us fill is 560,000 clock ticks.
- Each uTCA crate contains 12 WFD5s or 60 channels of digitization.
 - Crate 0 reads data from the clock and control center (CCC)
 - Crates 1-24 each read data from one calorimeter (+ spare channels)
 - Crate 25 reads data from the laser system
 - Crate 26 reads data from the Auxiliary detectors (Harps, Quads, and Kickers)
 - Crate 27 reads data from the three tracker detectors.
- Data from each crate is sent to a DAQ computer via a dedicated 10 Gb fiber. The total data rate is 20 GB/s.
- The data is then processed in Nvidia K40 GPUs.

Master frontend



- Communicates with other frontends using RPC calls.
- Provides begin of run and end of run RPCs to all frontends.
- Provides end of fill trigger to synchronous frontends.
- Configures clock and control system.
- Reads trigger times from Meinberg GPS unit and writes them to a MIDAS bank.

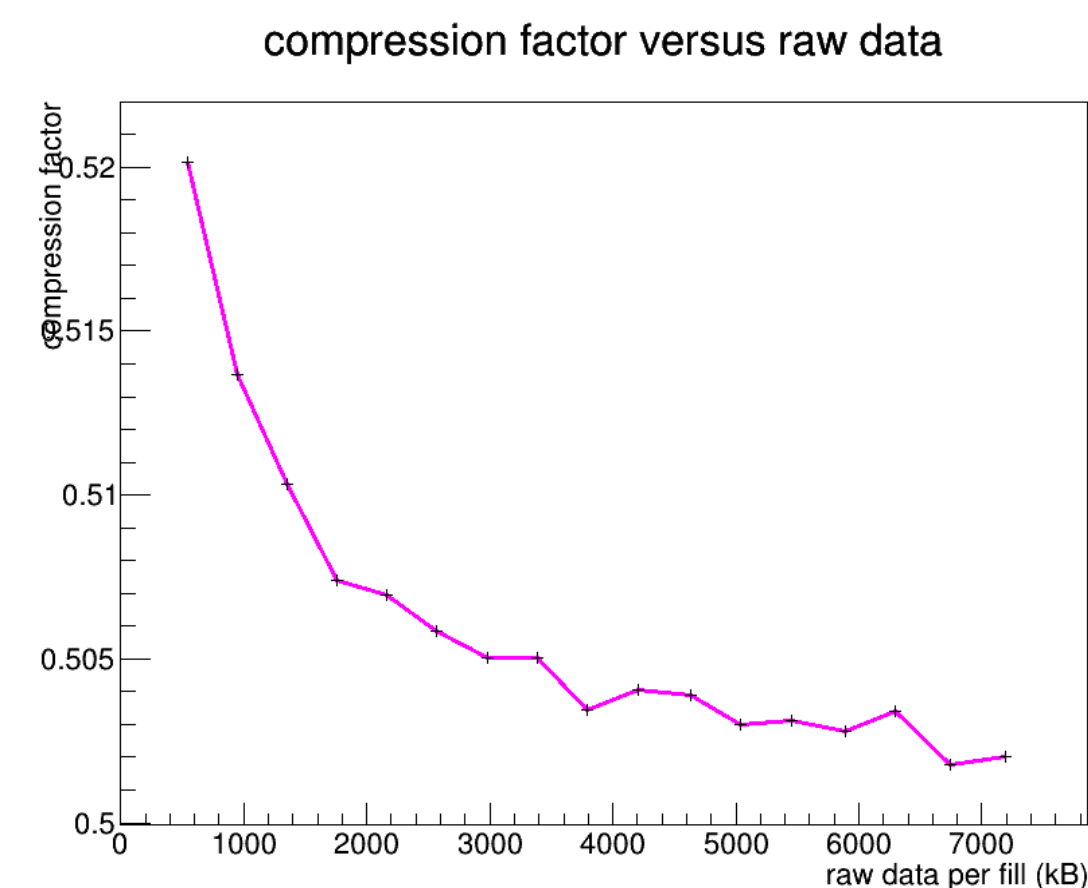
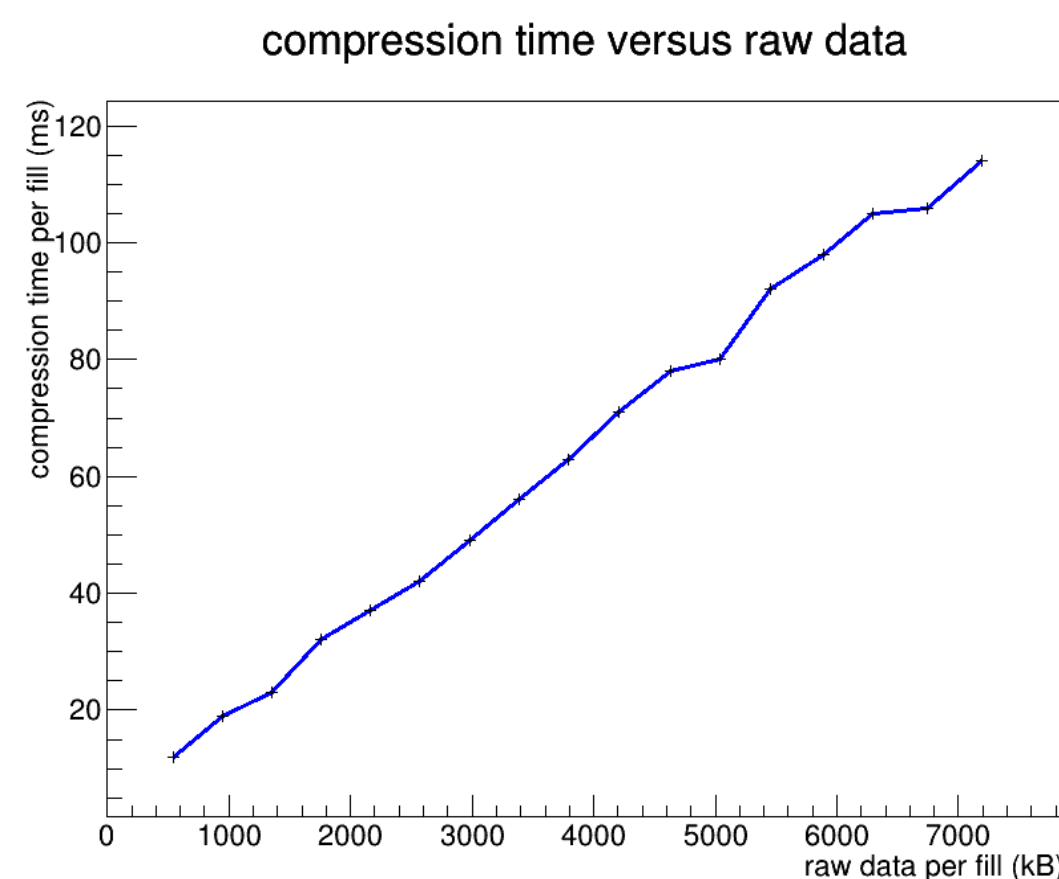
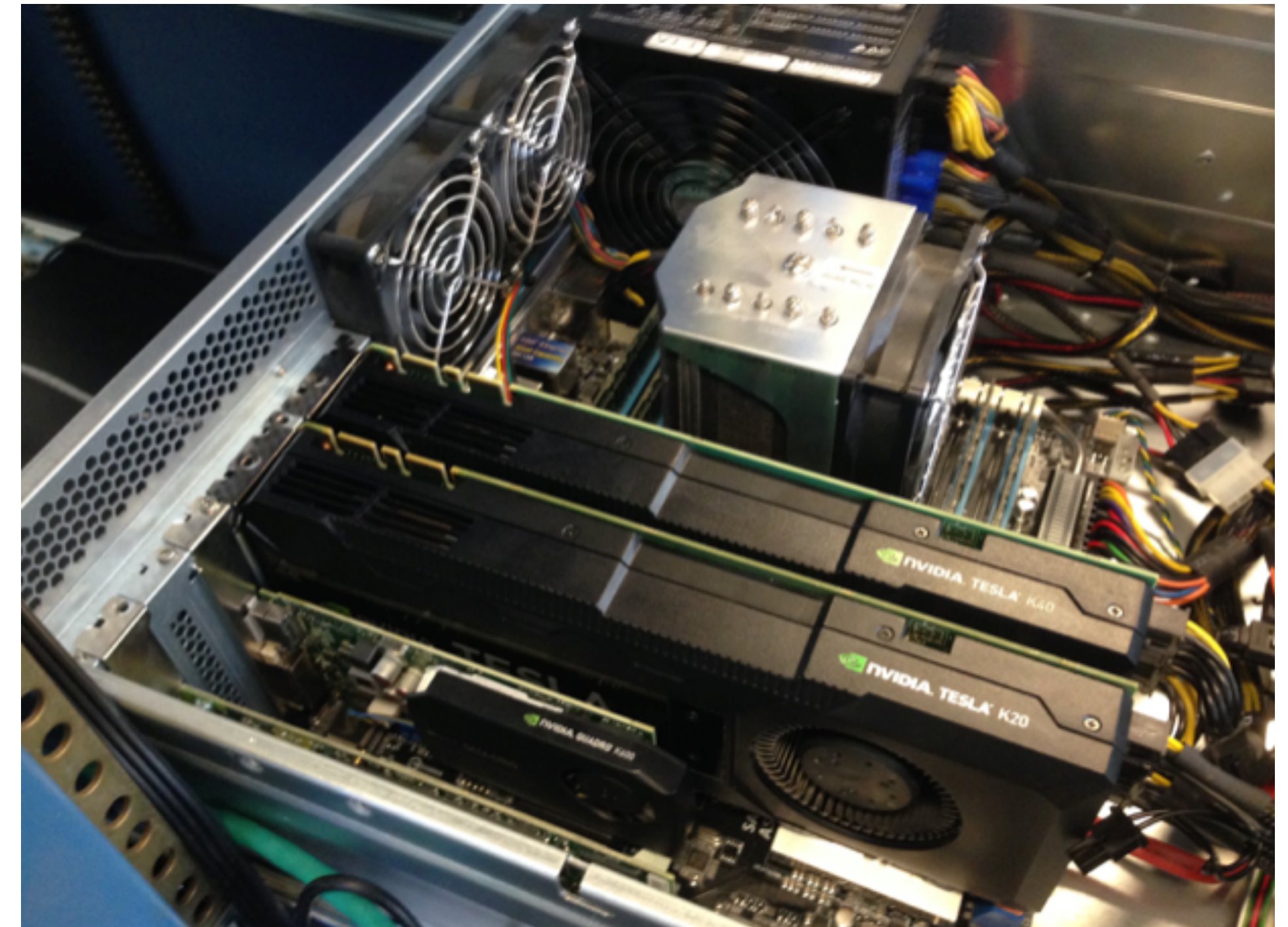


```
Bank:GPS0 Length: 20(I*1)/5(I*4)/5(Type) Type:Unsigned Integer*4  
1-> 0x00018c2 0x580838bc 0x5dcabfb0 0x580838bc 0x5dd48308
```

AMC13 Frontend

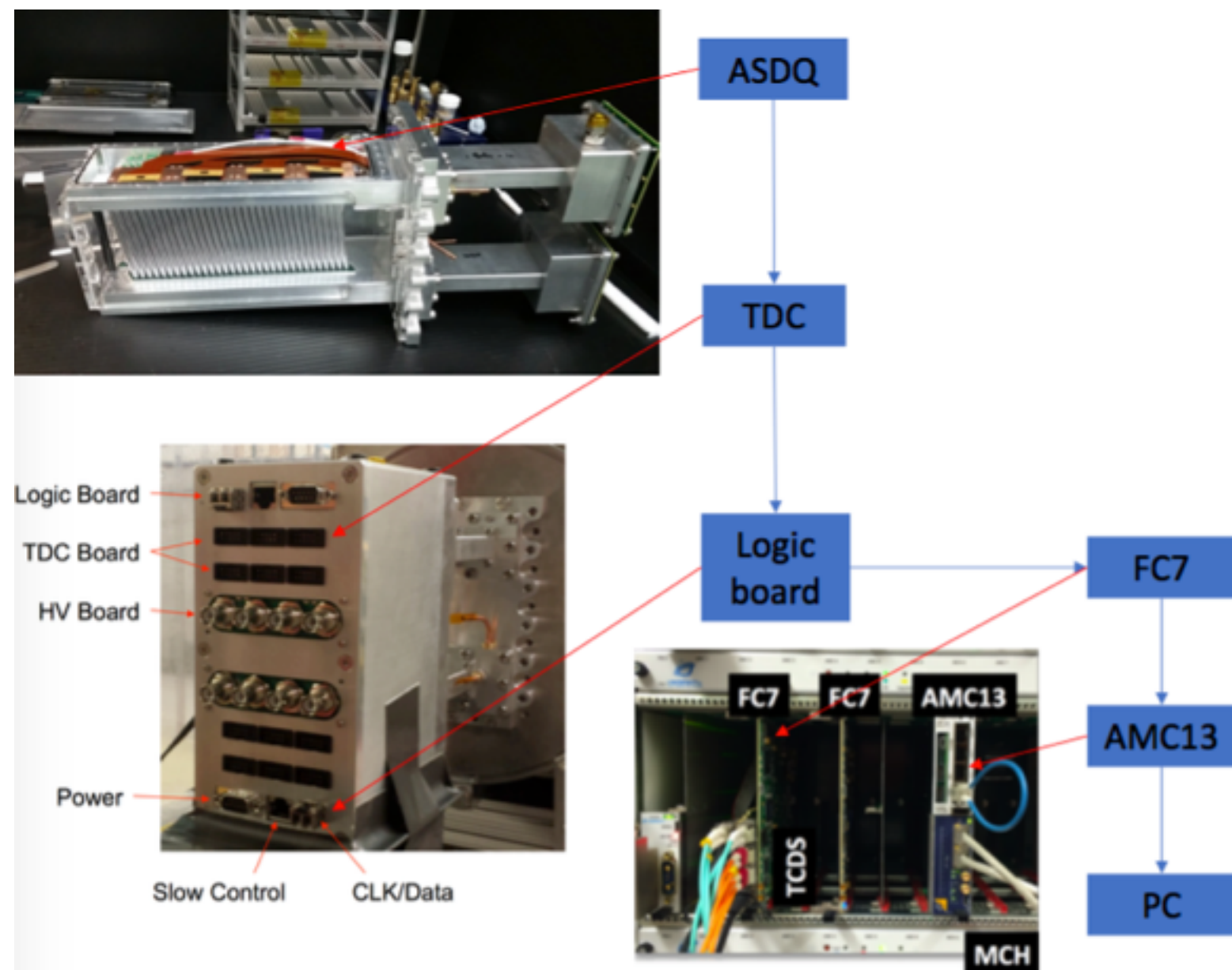


- Main frontend for processing data from calorimeters, laser, fiber harps, quads, and kickers.
- Each frontend process reads data from one uTCA crate over 10 Gb ethernet with TCPIP.
- Data is processed in Nvidia Tesla K40 GPUs using CUDA code that is integrated into the frontend.
- Midas banks are losslessly compressed using zlib.
- Full configuration of the uTCA crate is performed via the MIDAS ODB.



Tracker Frontend

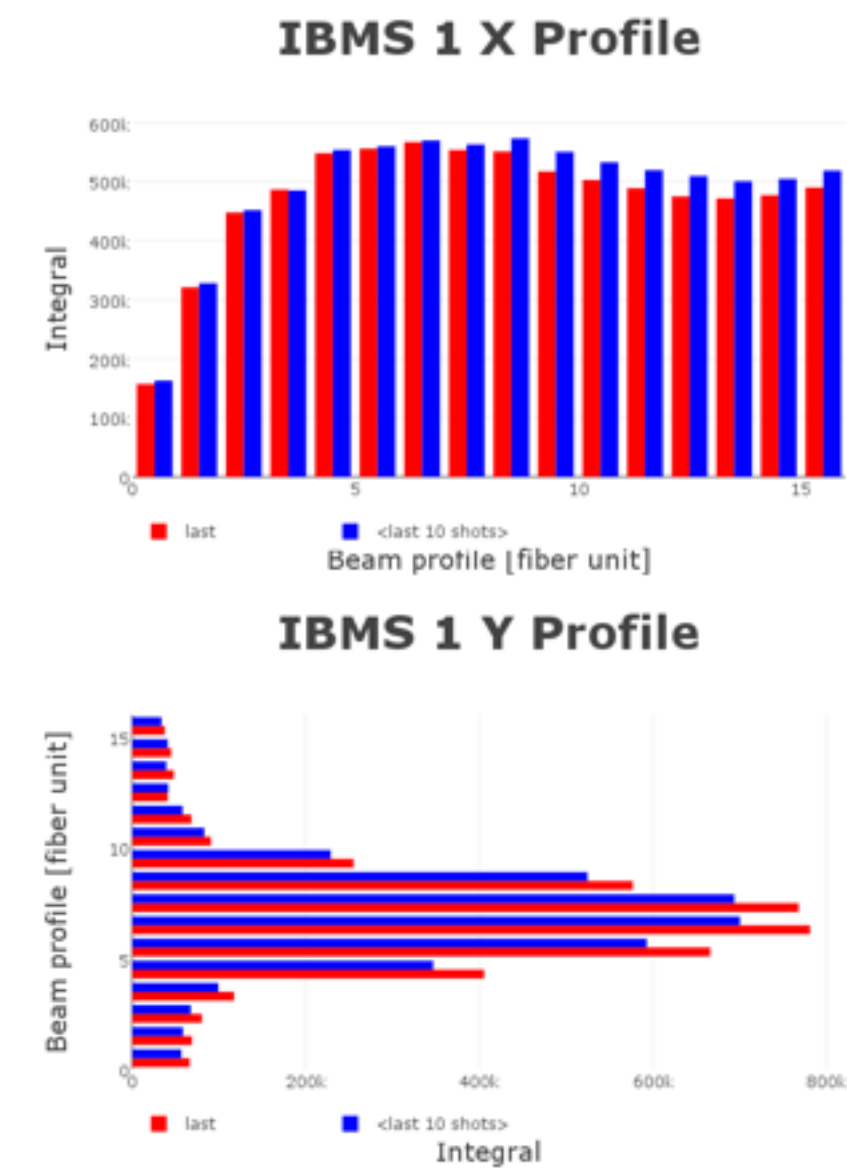
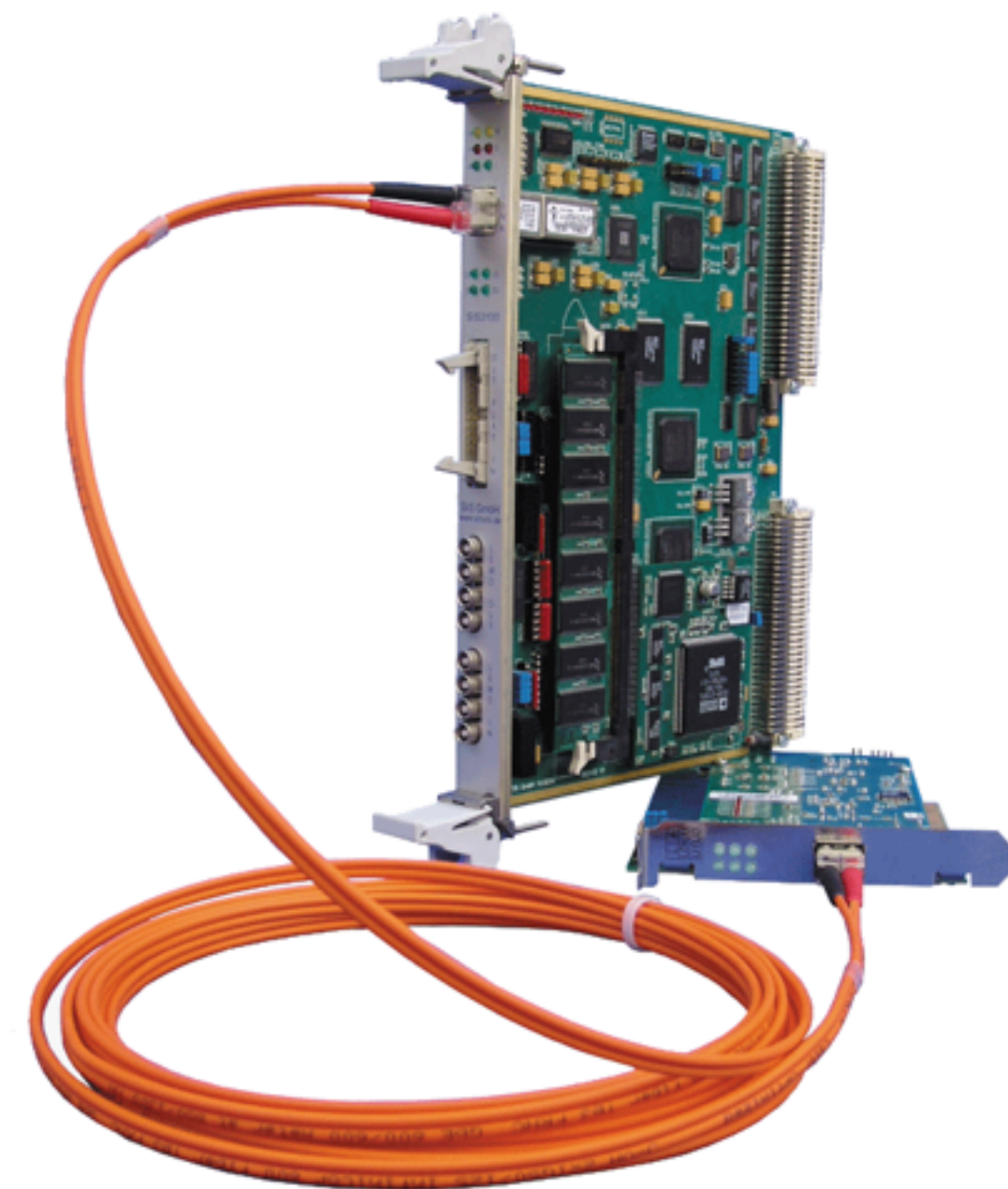
- Three tracker stations will be read via one uTCA crate.
- Reads data from AMC13.
- Instead of digitizers, data comes from multihit TDCs that are read via FC7 cards.



IBMS Frontend



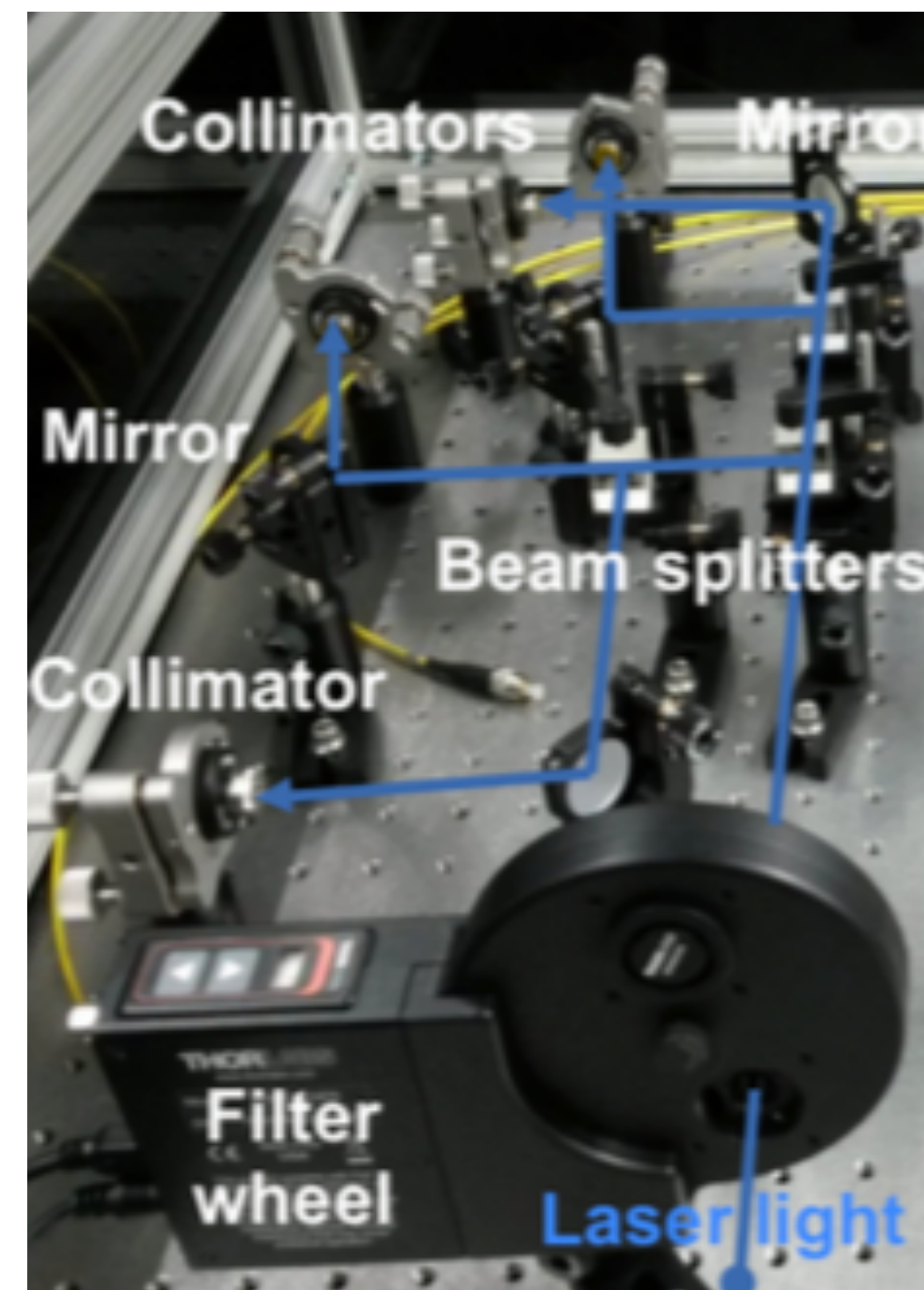
- Data from the inflector beam monitoring system (IBMS) is read out via a CAEN digitizer.
- A custom MIDAS frontend was written to integrate this detector into the DAQ.



MIDAS Sequencer



- The sequencer was used extensively for calibration runs such as filter wheel scans and bias voltage scans.
- A typical sequence would be:
 - Execute script to move wheel.
 - Update ODB values
 - Take data for 10 minutes
 - Repeat



Progress

Sequence is finished

Sequencer File

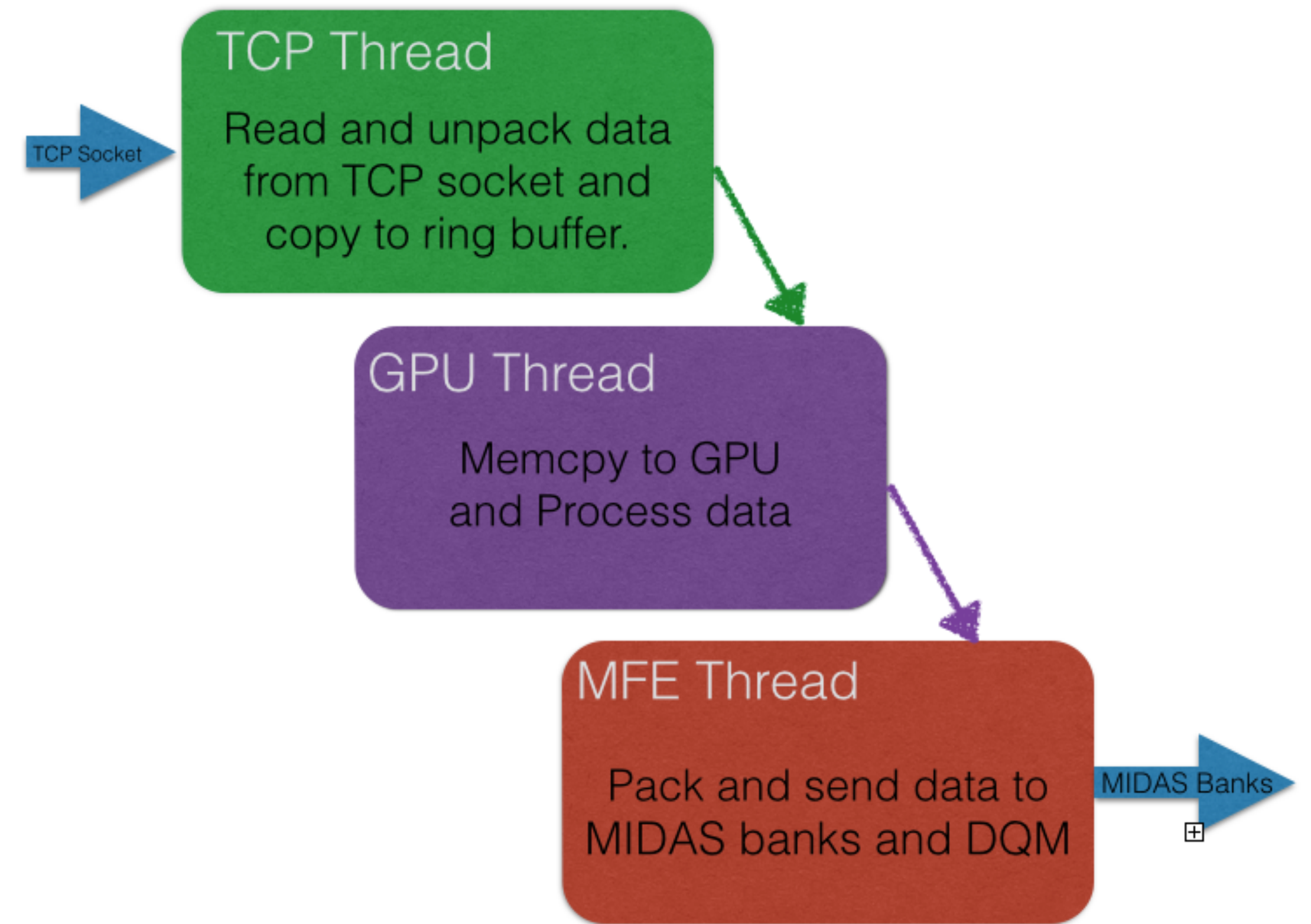
Filename: wheel_setting.msl [Show XML](#)

```
1 COMMENT "Run a sequence of laser runs"
2 RUNDESCRIPTION "Calibration run"
3
4 LOOP angle, 1, 2, 3, 4, 5, 6, 7, 1
5   ODBSET "/Experiment/Edit on start/Comment (256 max)", "Automated run"
6   ODBSET "/Experiment/Edit on start/Quality YNCT", "C"
7   SCRIPT /home/daq/test_wheel.sh, $angle
8   WAIT Seconds 2
9   TRANSITION START
10  WAIT Seconds 200
11  TRANSITION STOP
12 ENDLOOP
13 ODBSET "/Experiment/Edit on start/Comment (256 max)", "Enter comment"
14 ODBSET "/Experiment/Edit on start/Quality YNCT", "T"
```

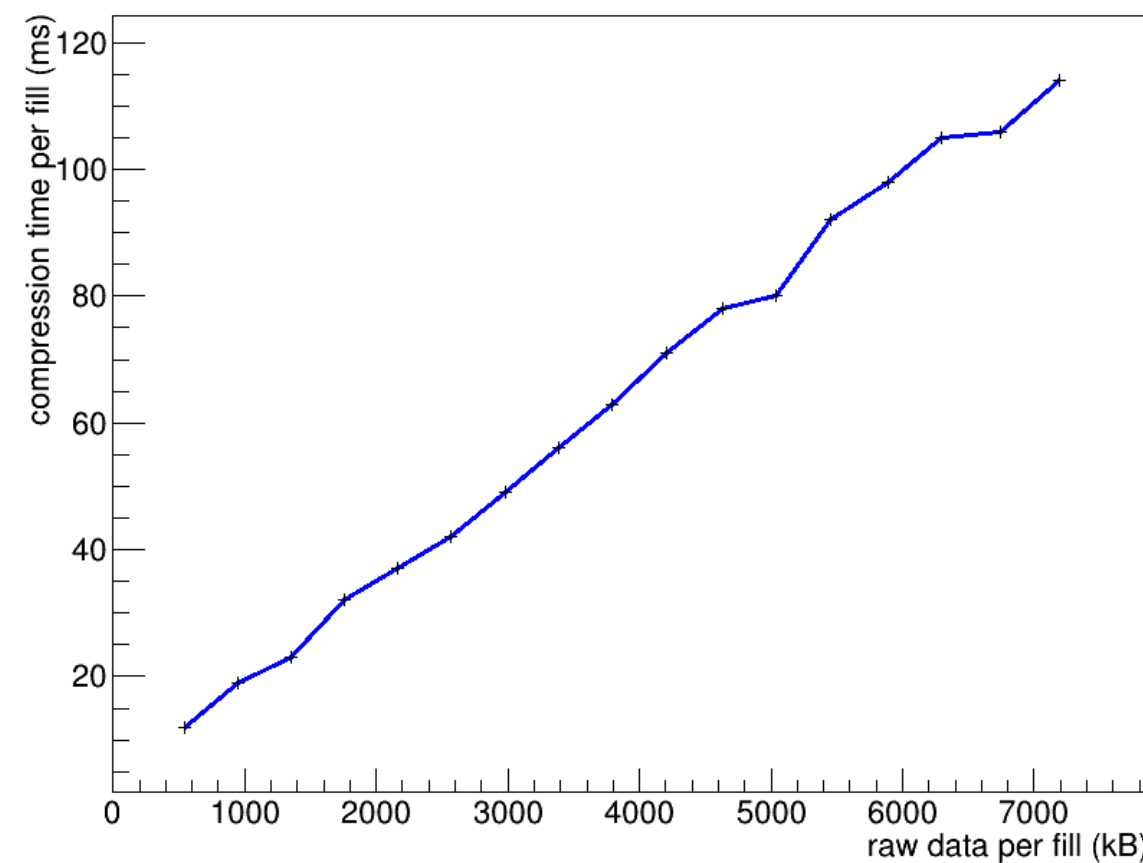

Calorimeter-GPU Frontend



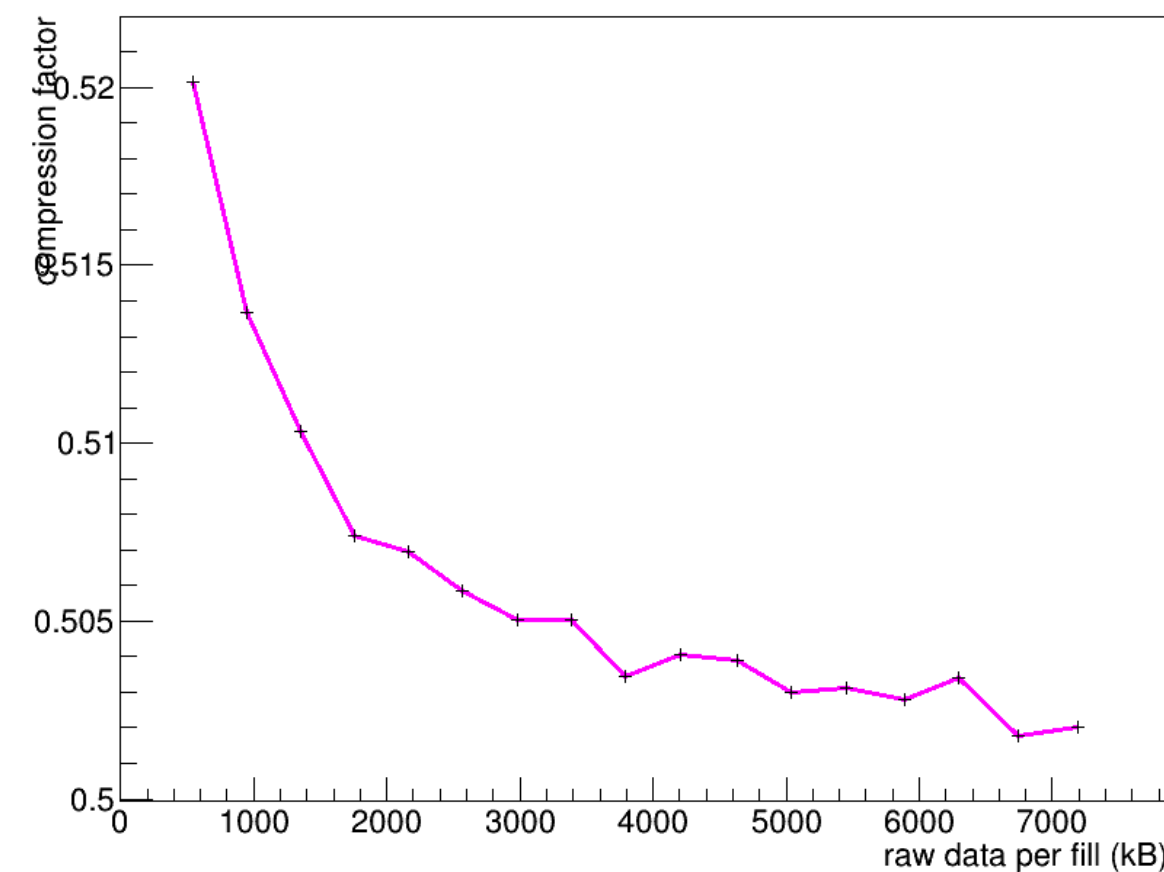
- Each frontend process reads data from one uTCA crate over 10 Gb ethernet with TCPIP.
- Frontend is multithreaded with mutex locks.
- Data is processed in Nvidia Tesla K40 GPUs using CUDA code that is integrated into the frontend.
- Midas banks are losslessly compressed using zlib.
- Full configuration of the uTCA crate is performed via the MIDAS ODB.



compression time versus raw data



compression factor versus raw data



Why GPUs?

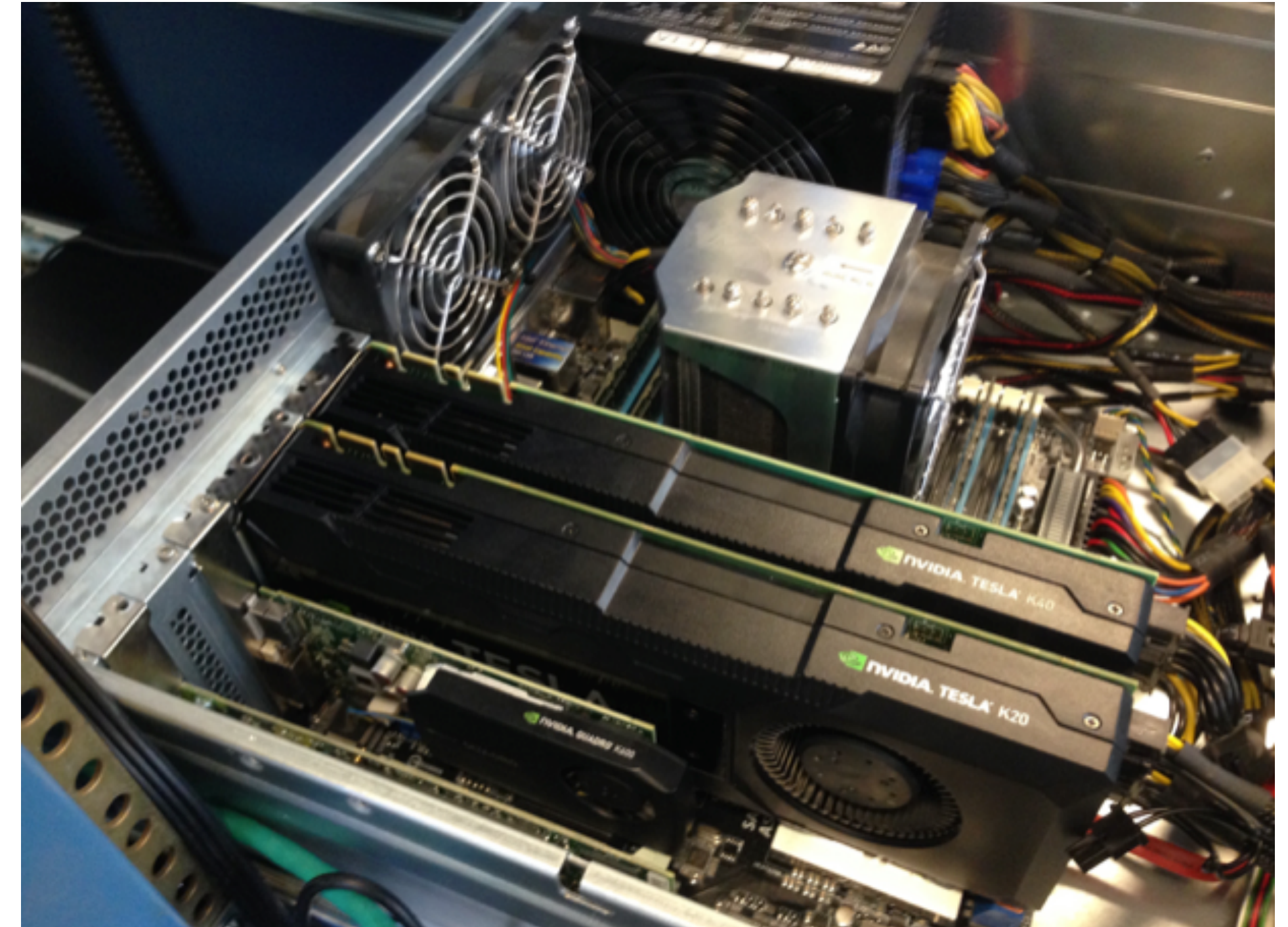
- The GPUs dramatically improve performance by parallelizing processing.
- Technology was developed for commercial applications, so it is well supported.
- Easier to code in C++ than to learn specialized language.
- Without GPUs, we could not keep up with our data rates.



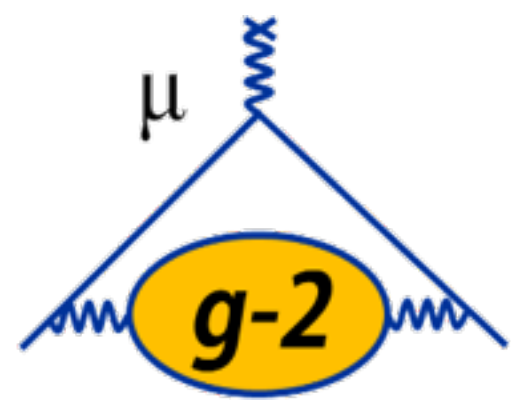
GPU Processing



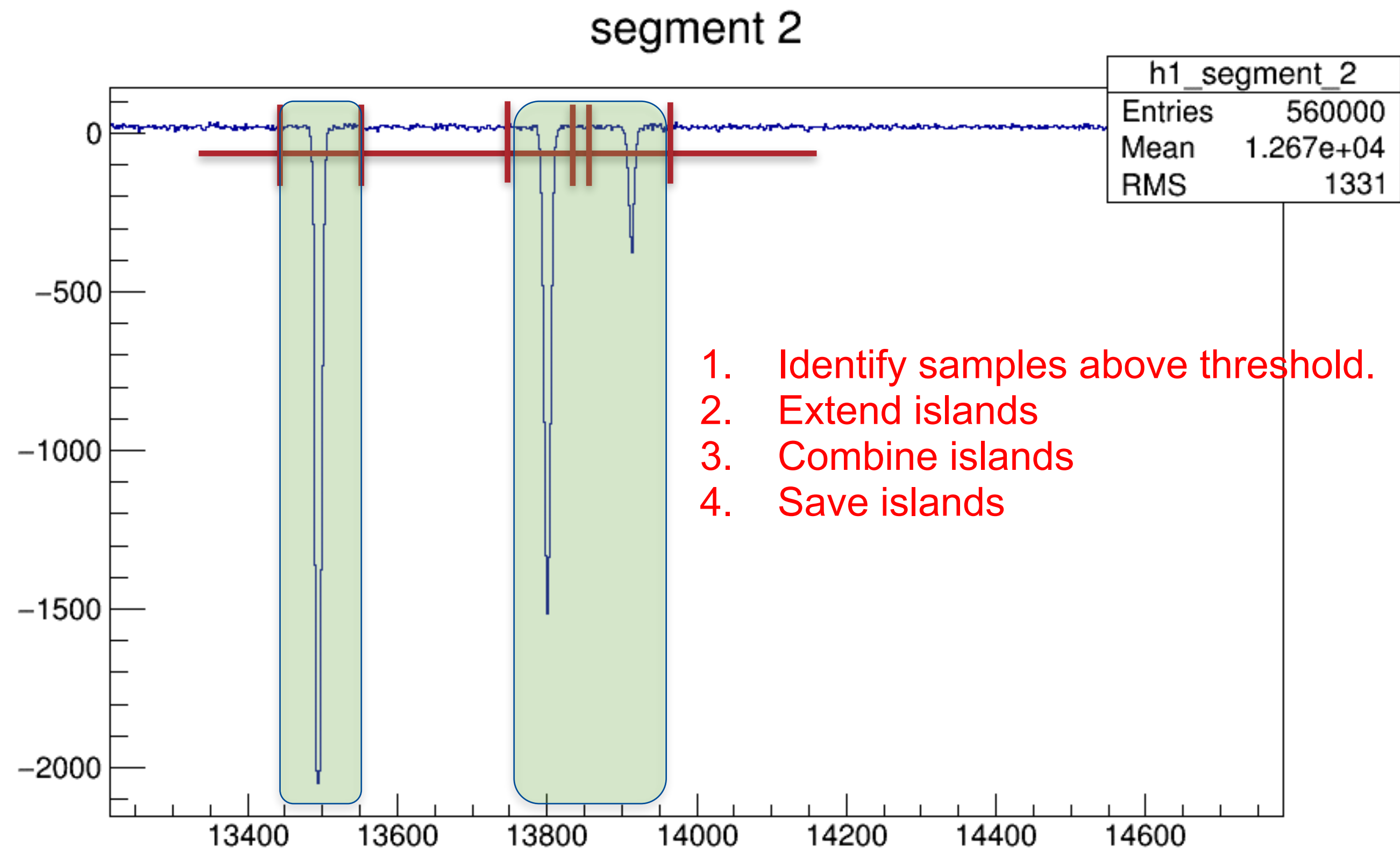
- The frontend includes CUDA routines for data processing.
- Each GPU processes data from one calorimeter.
- Raw fill is copied to GPU memory, where it is reduced using T-method (island chopping), Q-method (histogramming), pedestal calculation, and template fitting.
- The output of each process is written in one MIDAS bank.



T-Method



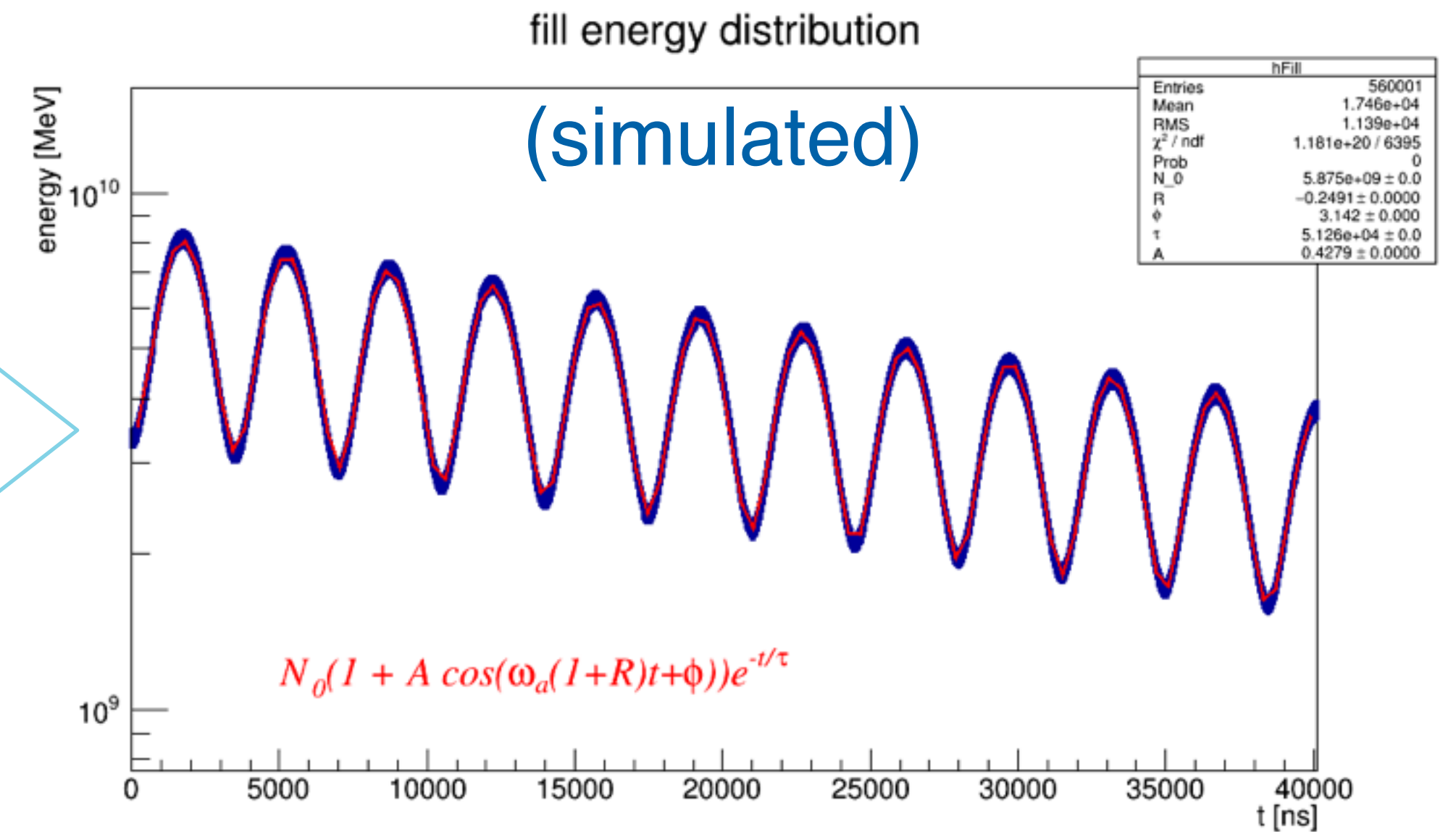
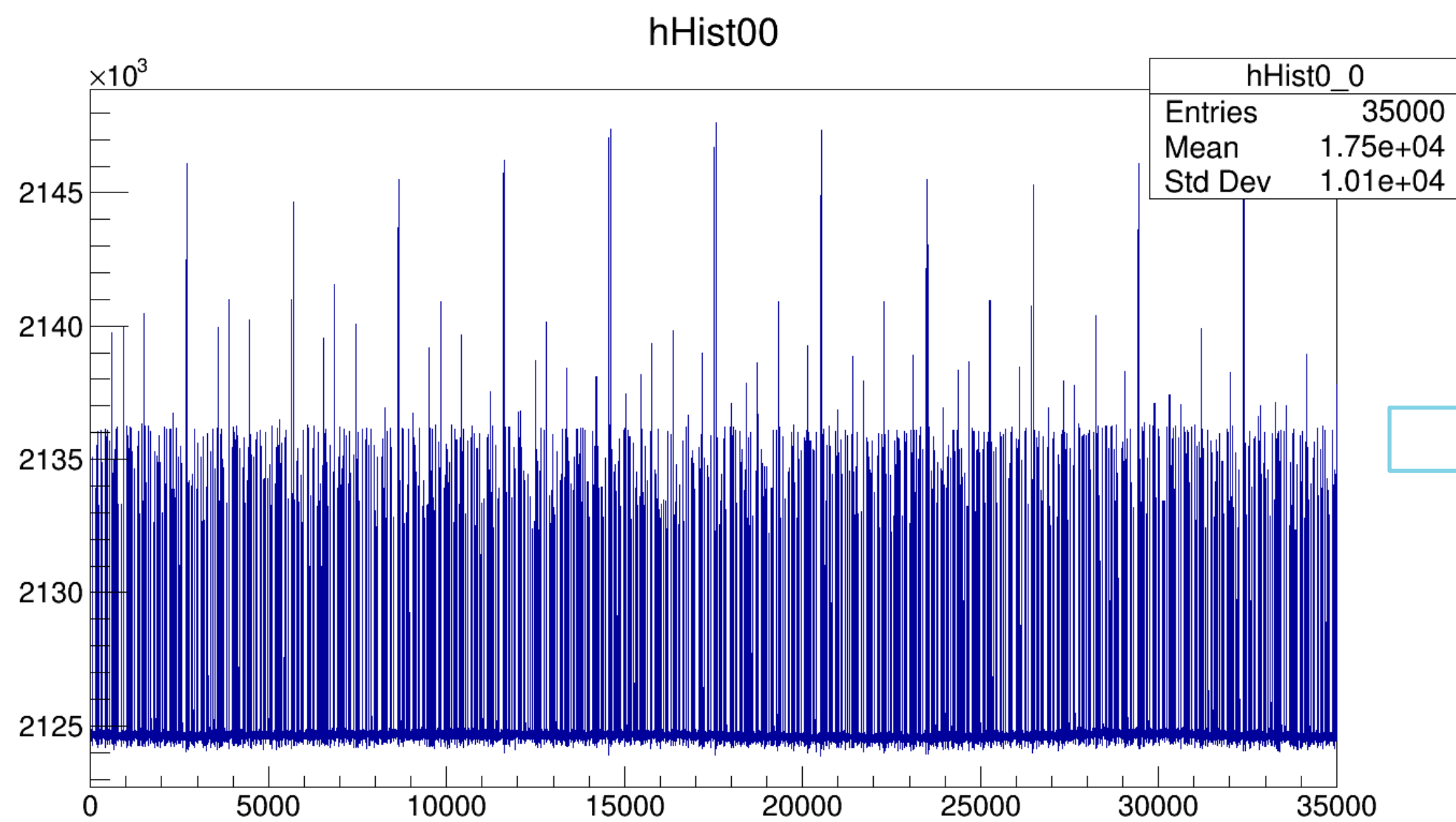
- Identify and save regions of the waveform containing positron hits.
- A typical waveform will have ~ 180 islands.



Q-method



- Full waveforms are decimated in time and summed over many fills to create a histogram that is saved in the data file.
 - i.e. If we decimate in time by 10 and flush every 100 fills, we reduce the data rate by a factor of 1000, so from 20 GB/s to 20 MB/s.
- Use smaller bins at lower times and wider bins at later times to insure that we can extract the pedestal.



GPU Template Fits



- Fit templates are loaded into the GPU memory and used to fit each peak above a certain threshold.
- A bank containing the fit results will be saved for each fill in addition to the chopped islands.
- Reduces the processing necessary in the online DQM.

