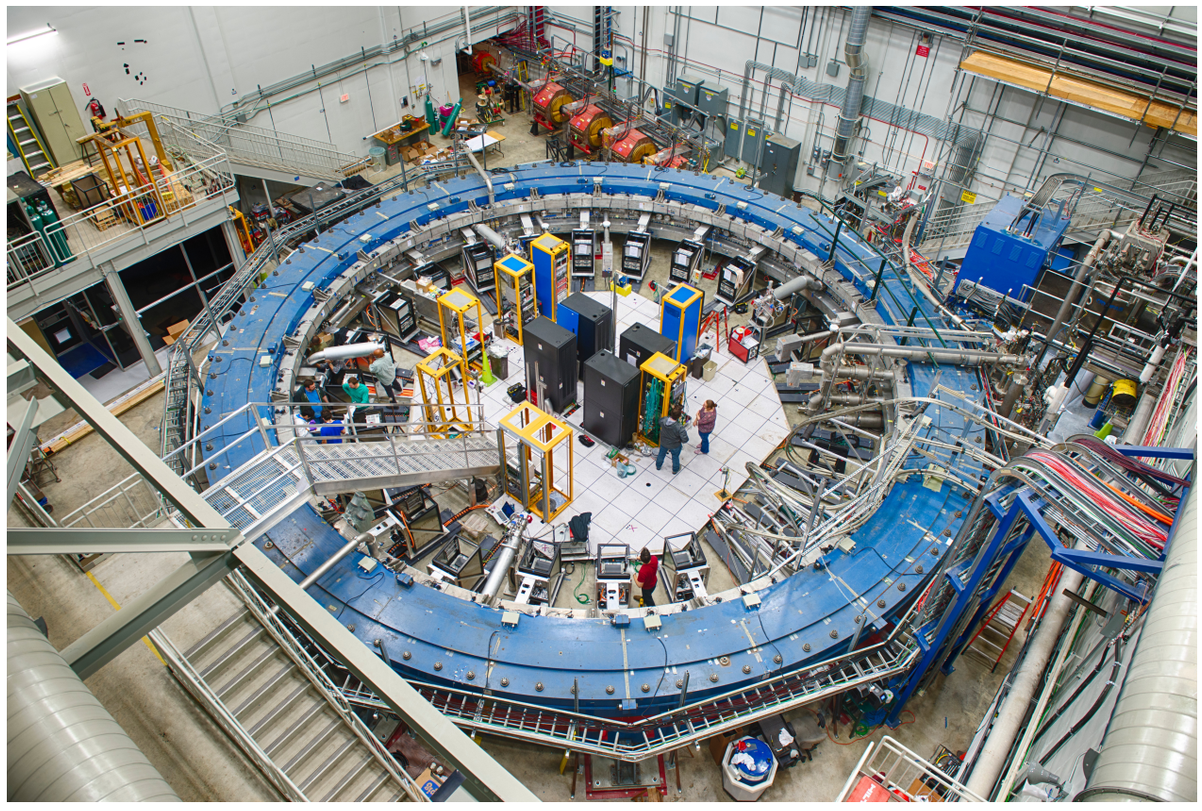
Использование программного пакета MIDAS в эксперименте «Muon g-2»

Н.В. Хомутов

Muon g-2 Experiment Overview

- Goal is to measure the anomalous magnetic moment of the muon to 140 ppb, which is a factor of 4 better than has been previously measured.
- Muon fills are injected into the ring at a rate of 12 Hz.
- The precession frequency of the muons is measured by detecting decay positrons in 24 segmented calorimeters inside the ring.

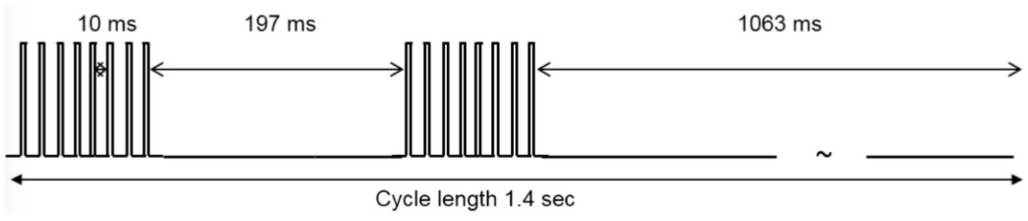


DAQ Input Sources

- 24 Calorimeters
  - 1 uTCA crate for each calorimeter
  - 54 channels * 24 calos = 1296 channels of digitized data.
  - Data processed by 12 Cornell WFD5s @800 MHz and sent from AMC13.
- 4 Fiber Harps
  - 7 channels * 4 harps = 28 channels
  - Data processed by Cornell WFD5s
- Quads and Kickers
  - 4 quad channels and 15 kicker channels
  - Data processed by Cornell WFD5s
- 3 Trackers
  - Data from Multihit TDCs sent from FC7s in a uTCA crate
- IBMS (inflector beam monitoring system)
  - Running on CAEN digitizer

Rate requirements

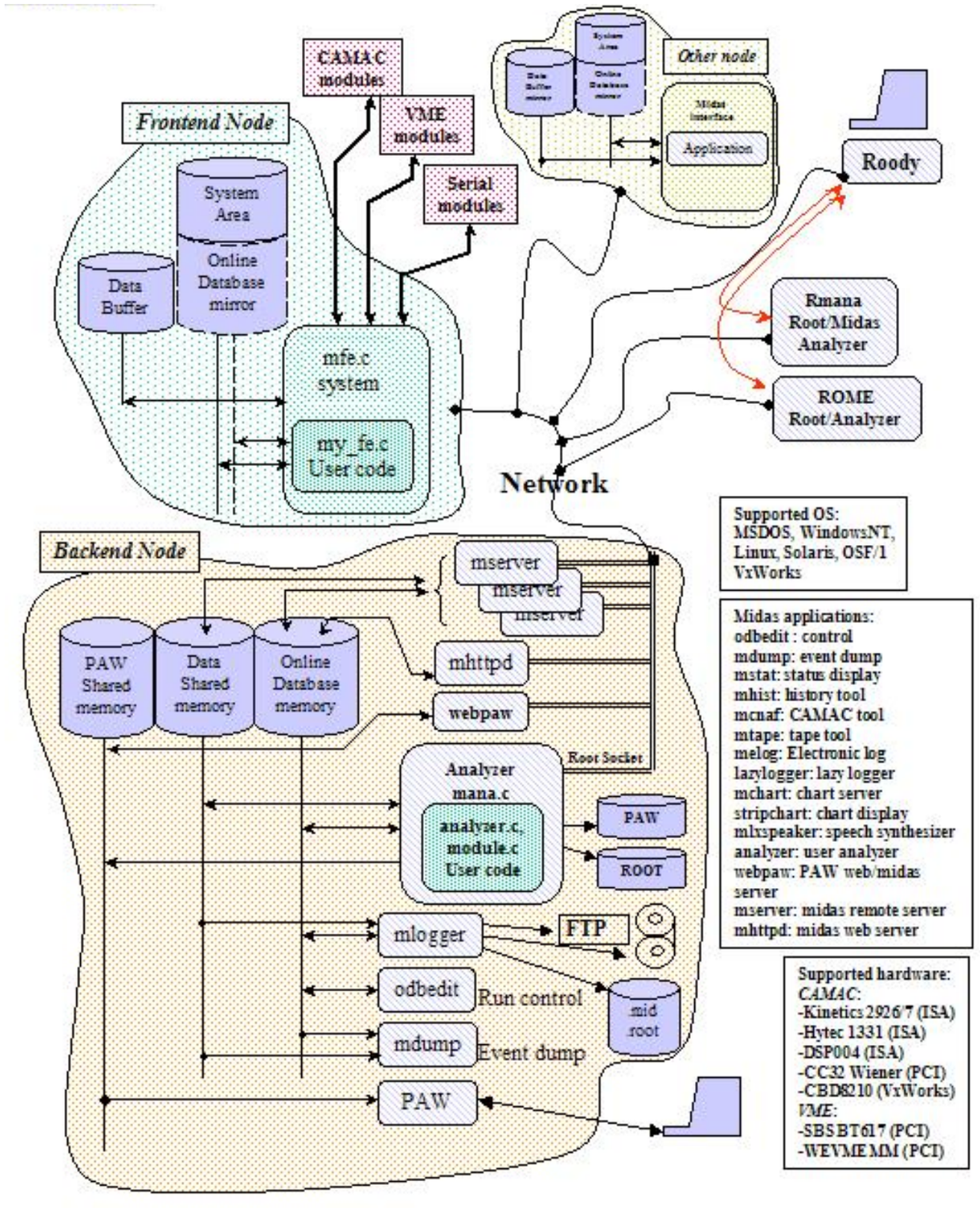
- Accommodate 12 Hz average rate of muon fills that consist of sequences of eight successive 700 μs fills with 10 ms fill-separations.



- 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 10 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**

Maximum Integrated Data Acquisition System



https://midas.triumf.ca

https://bitbucket.org/tmidas/midas.git

What is MIDAS?

- Maximum Integrated Data Acquisition System
- Maintained by TRIUMF and PSI
- Reference at midas.triumf.ca
- An Online Database (ODB) stores the real-time experiment configuration.
- Experiment is controlled via a web interface that interfaces with the ODB.
- Users write frontend programs that read data from the electronics and write it in MIDAS banks.

MIDAS Processes

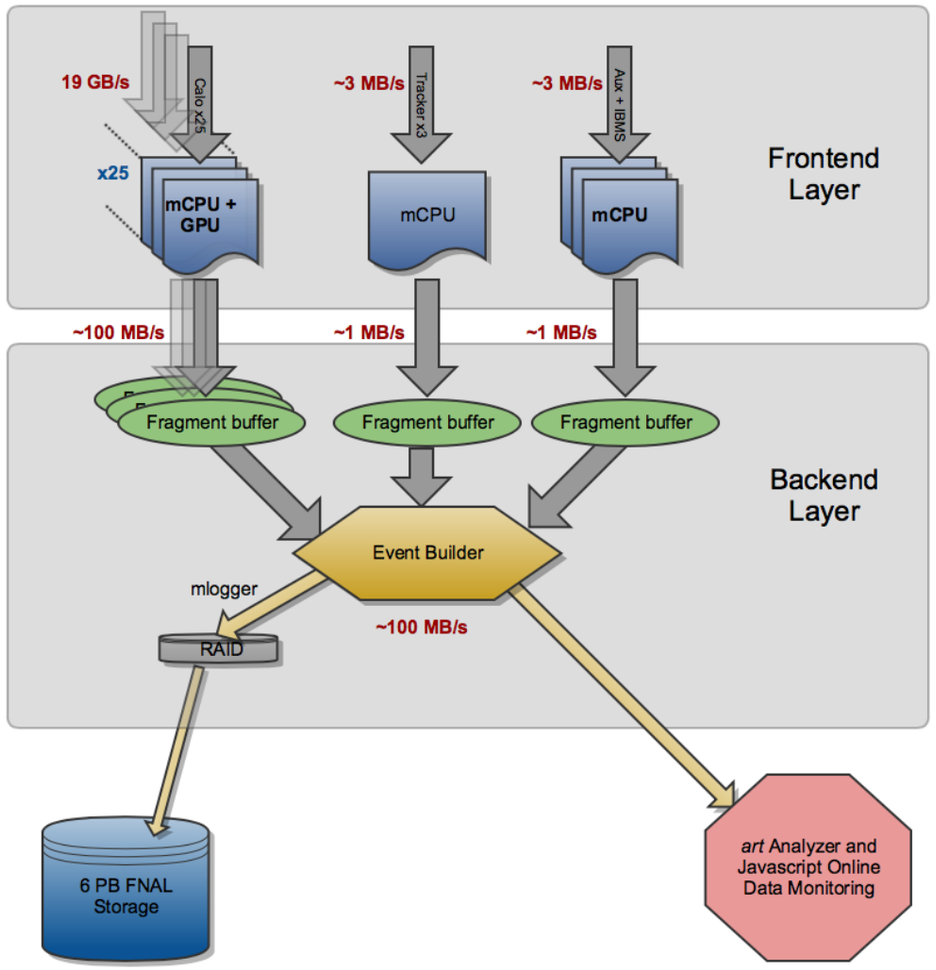
- mlogger: writes midas data files, writes select ODB parameters to MySQL database.
- mhttpd: MIDAS web interface.
- mserver: Central server that handles communication between all MIDAS processes.
- EventBuilder: Combines data fragments from multiple frontends into single event.

Muon g-2 experiment specific code:

- MasterGM2: Interfaces with CCC, trigger, and GPS. Communicates with frontends using RPC calls.
- Fast frontends: Write data at beam rate (i.e. 12 Hz)
- Slow frontends: Write data periodically (i.e. 1 Hz)

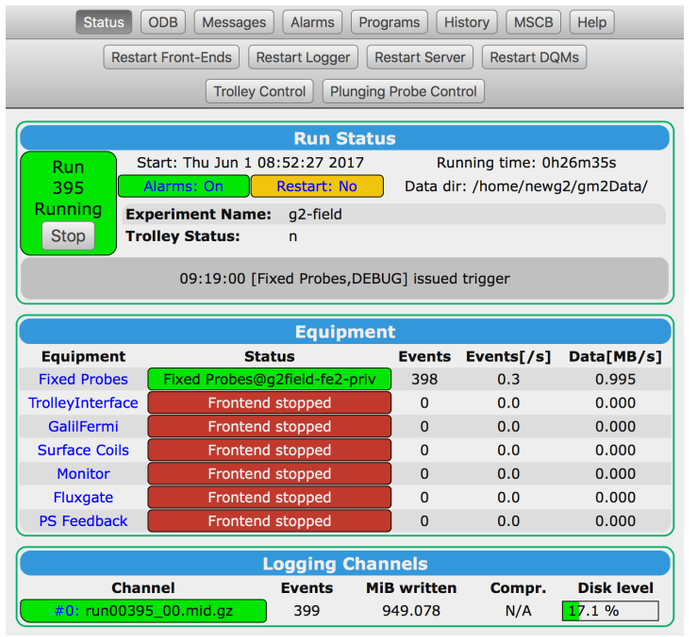
DAQ Design

- 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.



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.



DAQ Architecture

10 Gbps Fiber x28

1 Gbps CAT6 x18
to 10 Gbps fiber

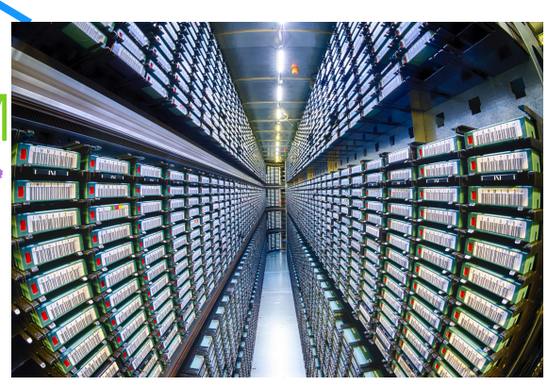
2x10 Gbps Fiber

20 GB/s

200 MB/s

FTS/SAM

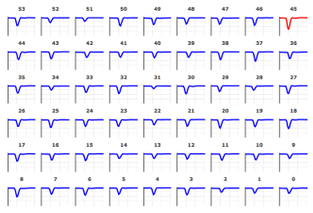
200 MB/s



μTCA x28

Frontends/
GPUs

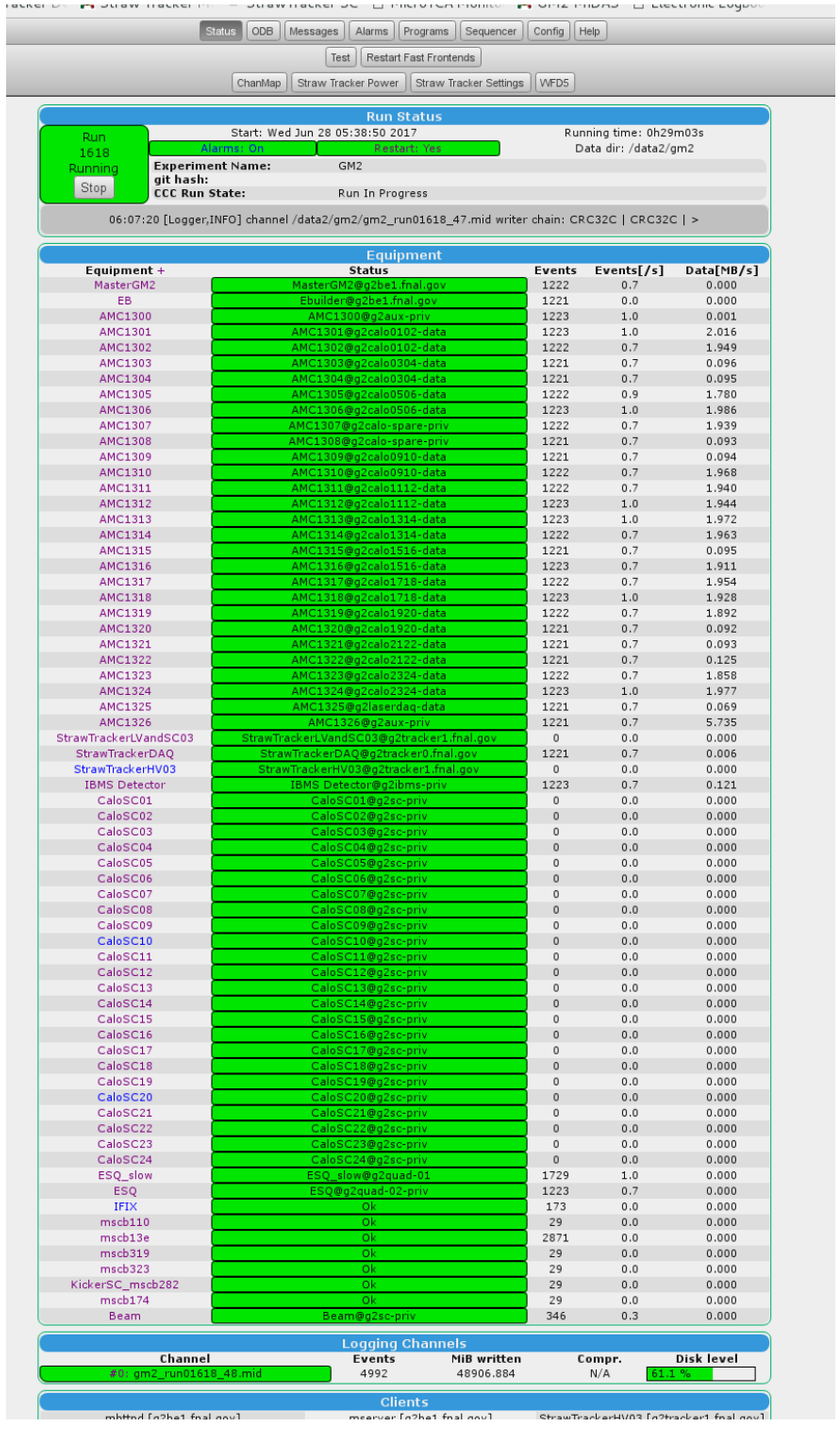
DQM



Fermilab tape
storage

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.

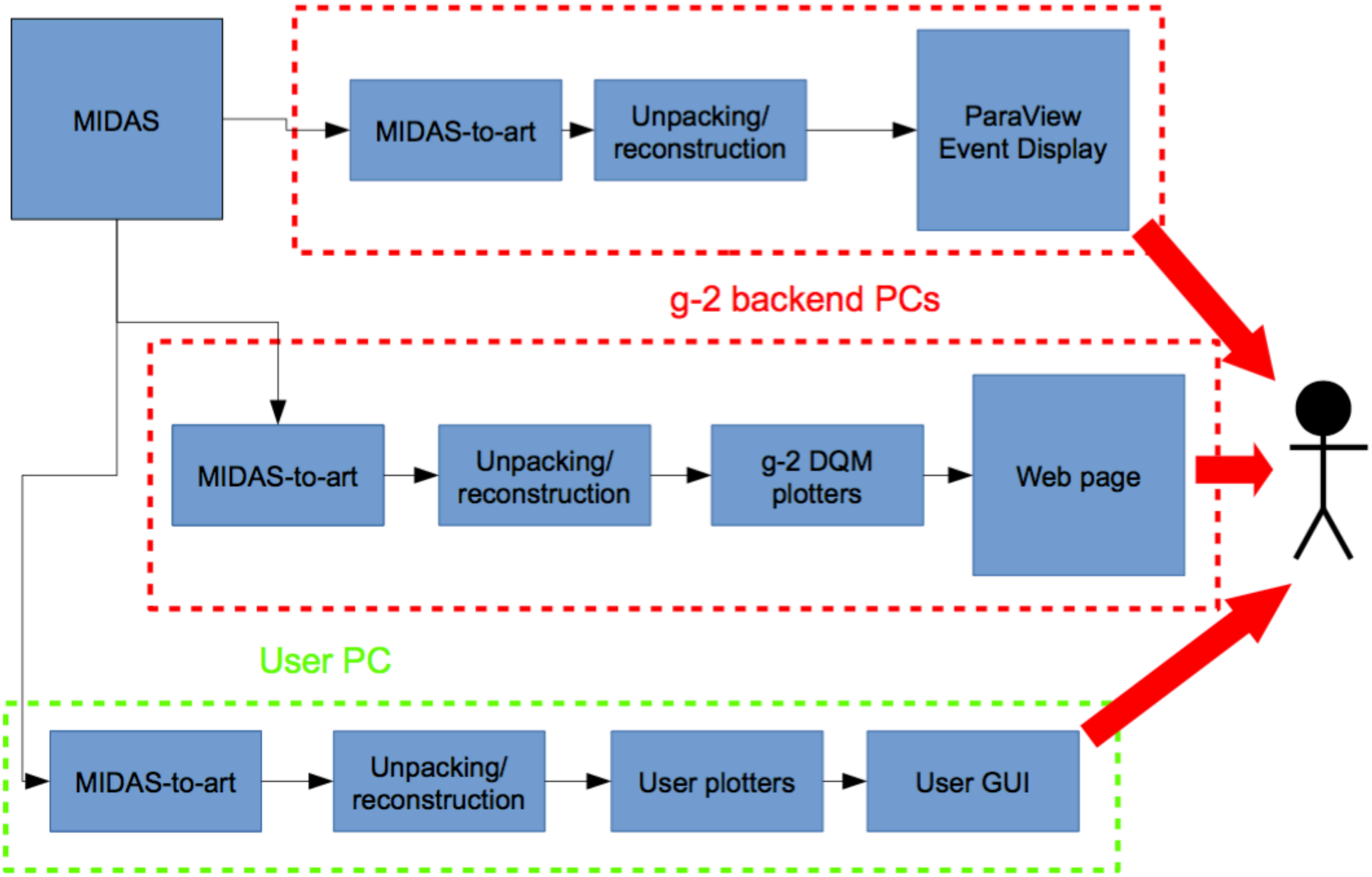


Fermilab *art* framework and *midas-to-art*

art is an event-processing framework developed/maintained by Fermilab SCD

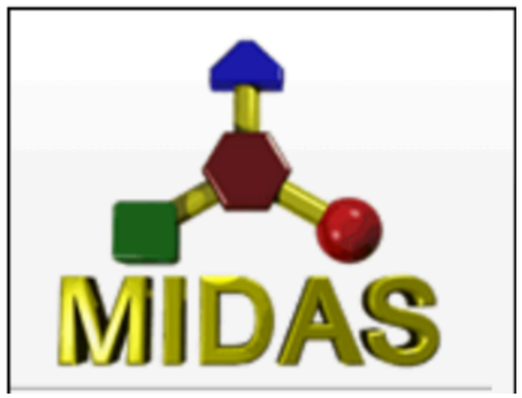
- used by (g-2), mu2e, NOvA, and others
- (g-2) unpacking/recon/analysis done in *art* modules
- *art* has its own file/data format
- *midas-to-art* plugin: reads MIDAS files into *art* jobs, translates MIDAS banks into *art* event data
- strong preference in collaboration to develop *art*-based DQM so we can reuse offline modules
- online *art* can be used as backend for number of different frontend DQM pages/event-displays, etc.

Data flow with MIDAS-to-art



General Architecture

bidirectional communication



midas experiment/mserver

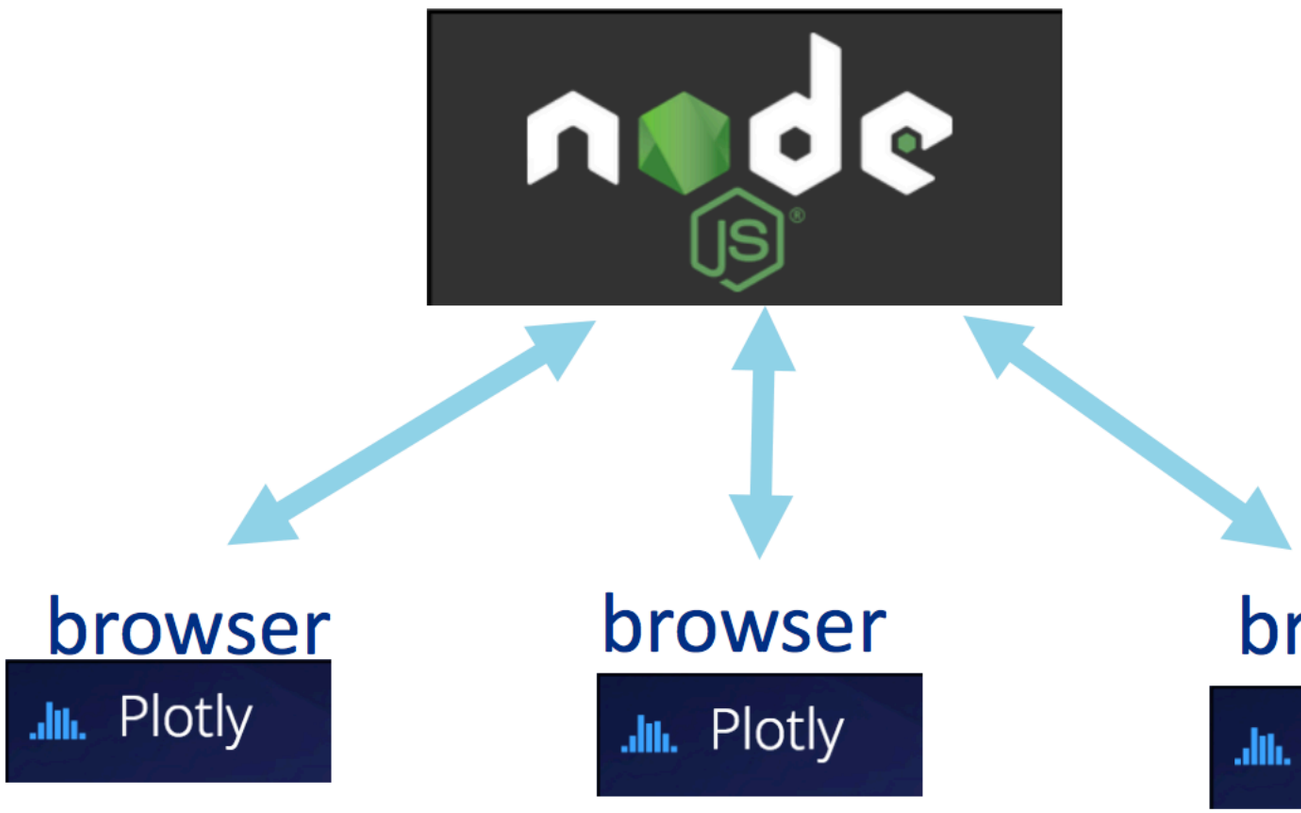
midas-to-art
unpackers
producers
analyzers

any module we've written can run online
(limited only by speed)

node.js webservers
data aggregation
client communication

connections to multiple clients

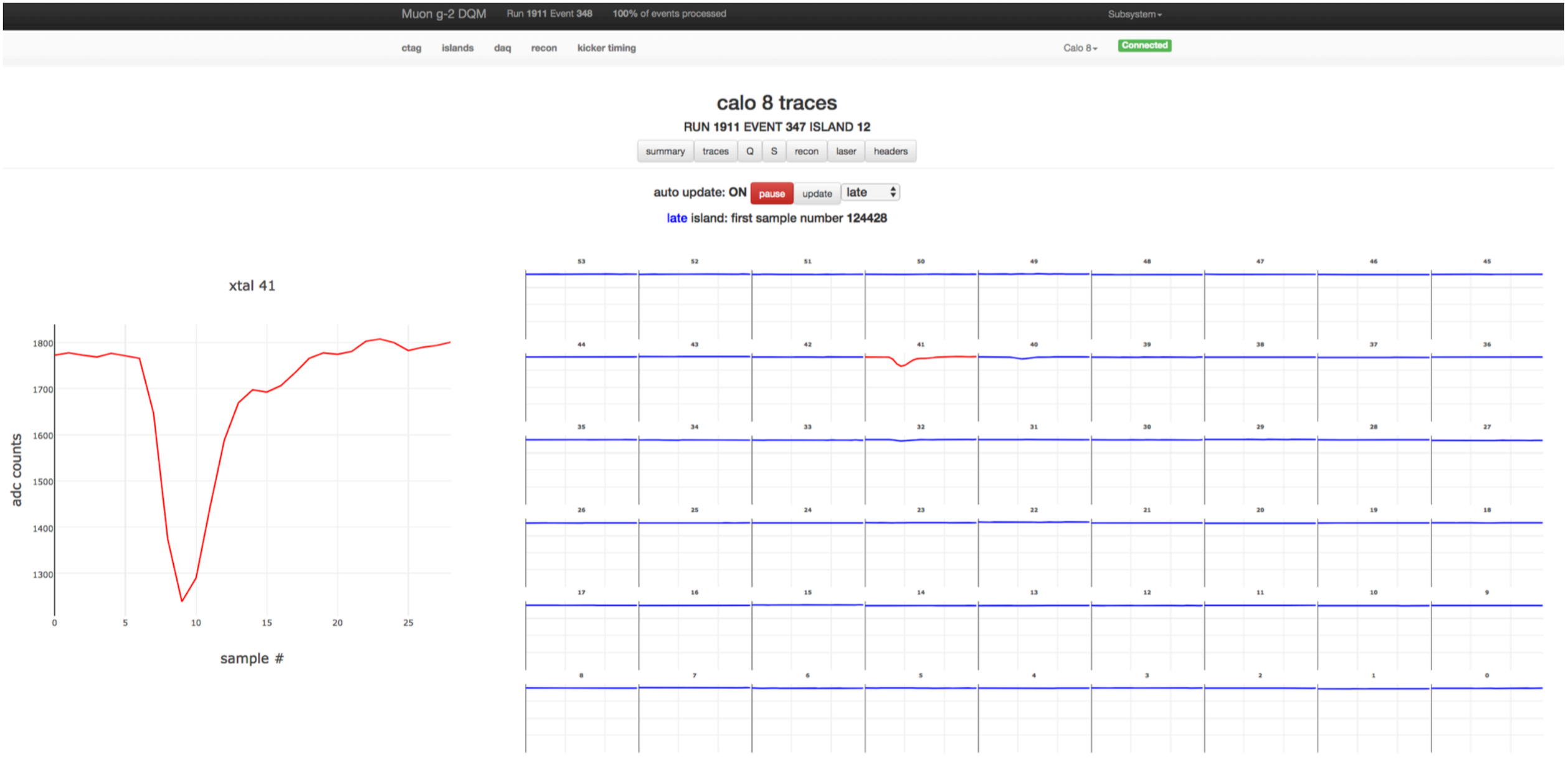
plot rendering is outsourced to clients



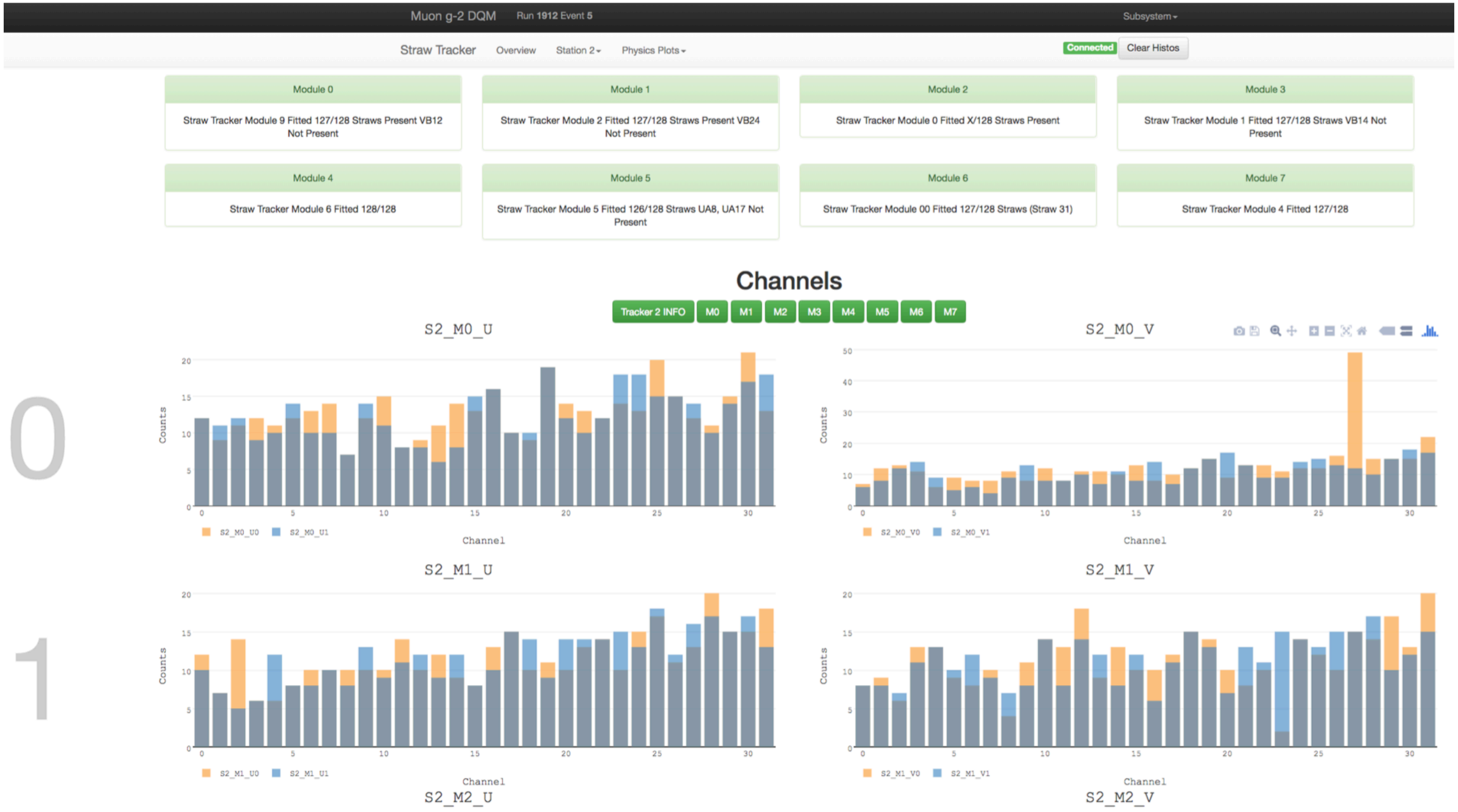
mainly using plotly.js for interactive plots

- server sends data through WebSockets on client request
- clients are insulated from both *art* job and MIDAS
- we use some additional js visualization tools (D3.js)

positron pulse in a calorimeter

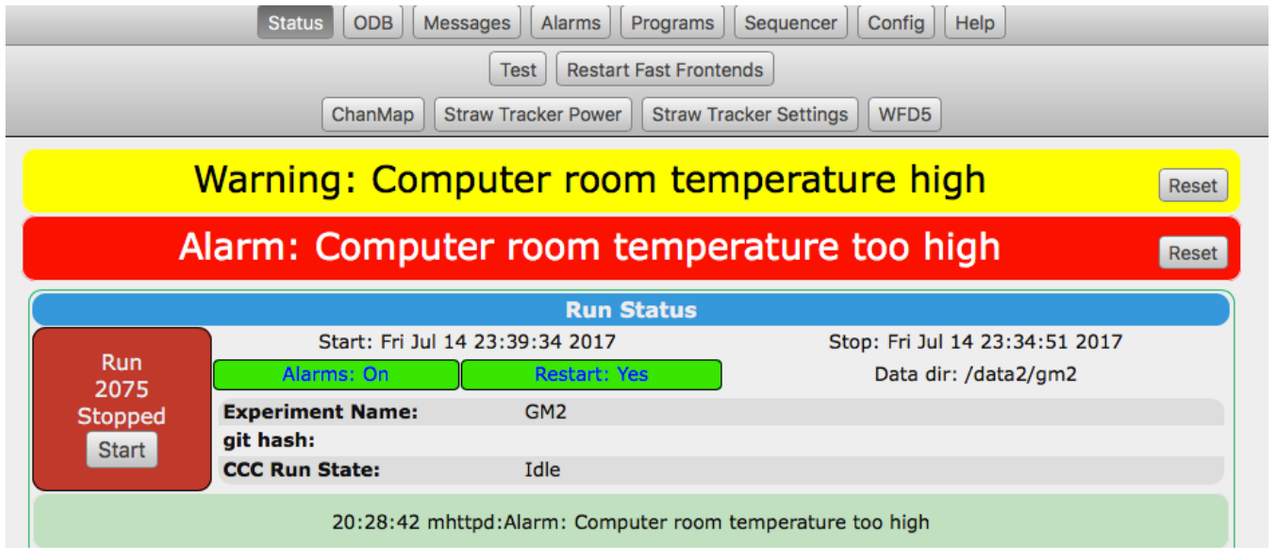


straw tracker TDC hits



MIDAS Alarms

- The MIDAS alarm system was used as the primary alarm system.
- Alarms were set on temperatures and voltages from MSCB devices.
- Other slow frontends set alarms automatically when encountering an error.



MIDAS web alarm page

Alarms - Mozilla Firefox

Mjsonrpc - MidasWiki | Alarms

localhost:8082/?cmd=Alarms

Search

2017 | Asus-2017 | MIX | g-2 | Work | Midas | IT | IL | JS | MIEX | CR | YT | Sh | ALT

Status | ODB | Messages | Alarms | Programs | Sequencer | Config | Help

Reset all alarms | Disable the alarm system | Speak

Evaluated alarms

Alarm	State	First triggered	Class	Condition	Current value	
Demo ODB	Disabled	-	Alarm	/Runinfo/Run number > 100	6667	Reset
EB	OK	-	Warning	/Equipment/EB/Statistics/kBytes per sec. > 2000000	0	Reset
RateL	OK	-	Alarm Low	/Equipment/EB/Statistics/kBytes per sec. > 3000000	0	Reset
RateH	OK	-	Alarm High	/Equipment/EB/Statistics/kBytes per sec. > 5000000	0	Reset
Test Alarm	OK	-	Test	/Runinfo/Run number = 0	6667	Reset
Magnet_sectorB_temp	OK	-	Warning	/Equipment/mscb13e/Variables/Input[8-9] > 30	24.89489	Reset
Magnet_sectorB_temp2	OK	-	Alarm	/Equipment/mscb13e/Variables/Input[8-9] > 30	24.89489	Reset
Magnet_sectorB_bottom_temp	Disabled	-	Warning	/Equipment/mscb13e/Variables/Input[10] > 27	25.36532	Reset
Magnet_sectorB_bottom_temp2	Disabled	-	Alarm	/Equipment/mscb13e/Variables/Input[10] > 30	25.36532	Reset
mscb110-8_temp	OK	-	Warning	/Equipment/mscb110/Variables/Input[8] > 27	21.21	Reset
mscb110-8_temp2	OK	-	Alarm	/Equipment/mscb110/Variables/Input[8] > 30	21.21	Reset
Kicker_OilLevel_1	Disabled	-	Kicker Alarm	/Equipment/KickerSC_mscb282/Variables/Input[35] < 1	1	Reset
Kicker_OilLevel_2	Disabled	-	Kicker Alarm	/Equipment/KickerSC_mscb282/Variables/Input[36] < 1	1	Reset
Kicker_OilLevel_3	Disabled	-	Kicker Alarm	/Equipment/KickerSC_mscb282/Variables/Input[37] < 1	1	Reset
Straw Tracker HV	OK	-	Tracker Data Alarm	/Equipment/StrawTrackerHV03/Variables/Channel tripped > 0	0	Reset
IBMS rate	OK	-	Warning	/Equipment/IBMS Detector/Statistics/Events per sec. > 50	0.3332222592469177	Reset
FiberHarp180Y	OK	-	Warning	/Equipment/FiberHarpControl/Variables/Input[7] == 0	1	Reset
FiberHarp180X	OK	-	Warning	/Equipment/FiberHarpControl/Variables/Input[3] == 0	1	Reset
FiberHarp270Y	OK	-	Warning	/Equipment/FiberHarpControl/Variables/Input[16] == 0	1	Reset
FiberHarp270X	OK	-	Warning	/Equipment/FiberHarpControl/Variables/Input[12] == 0	1	Reset

Program alarms

MIDAS Alarm system capabilities

- Alarm setting on any ODB variable against a threshold parameter
- Alarm triggered by evaluated condition
- Selection of Alarm check frequency
- Selection of Alarm trigger frequency
- Customization alarm scheme: multiple choices of alarm type
- Selection of alarm message destination (to system message log or to elog)
- Email or SMS alerts can be sent
- Alarm triggered when a Program is not running

A MIDAS experiment uses the ODB (fast online database) for storing various equipment parameters such as electronics settings and slow control variables.

Only some basic tools are available for viewing and editing the ODB data out of the box: a command-line **odbedit** application and a web based odb editor provided by the standard MIDAS **mhttpd** web server. These tools are good enough only for a relatively small odb with simple data organisation.

A custom solution is necessary for larger experiments (like “Muon g-2”@Fermilab or MEG-II @PSI), where many thousands of settings and channels should be controlled. Such settings could be also more naturally structured not as 1-D arrays, but as n-tuples or trees or multidimensional tables.

In the following slides custom MIDAS pages developed for controlling DAQ electronics for the “Muon g-2” experiment are presented.

Some features implemented in custom MIDAS pages

- Different types of data (boolean, integer, string etc.) could be edited.
- The same value (or a structured set of values) could be written to a subset of an array defined by a range selector. Example: **1,3,5-9,22**.
- Any values before writing to the ODB could be checked for validity.

Setting Mixed Type Parameters in 3-D table (top)

WFD5

localhost:8082/CS/wfd5 90% Search

Status | ODB | Messages | Alarms | Programs | Sequencer | Config | Help

WFD5 Channels

Configure 1 crate: Crate # 1 Configure multiple crates: Crate # 1 VALIDATE SELECTION

WRITE to ODB DATA LAST WRITTEN on 0

/Equipment/AMC1301/Settings/RiderXX/Board/

Key name	Rider 01	Rider 02	Rider 03	Rider 04	Rider 05	Rider 06	Rider 07	Rider 08	Rider 09	Rider 10	Rider 11	Rider 12	WFD5 ##	Value	SET
Enabled	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☐	1	1	S
Frontend Configuration Enabled	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	1	1	S
FPGA Firmware Version Required	2.5.1	2.5.1	2.5.1	2.5.1	2.5.1	2.5.1	2.5.1	2.5.1	2.5.1	2.5.1	2.5.1	2.5.1	1	2.5.1	S
Digitization Frequency (MHz)	800	800	800	800	800	800	800	800	800	800	800	800	1	800	S
Front Panel Clock Enabled	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	1	1	S
Front Panel Clock Frequency	ttc	ttc	ttc	ttc	ttc	ttc	ttc	ttc	ttc	ttc	ttc	ttc	1	ttc	S
ADC Data Endianness	little	little	little	little	little	little	little	little	little	little	little	little	1	little	S
Trigger Delay (ns)	0	0	0	0	0	0	0	0	0	0	0	0	1	0	S
Async Mode: Enabled	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	1	0	S
Async Mode: Waveform Length	80	80	80	80	80	80	80	80	80	80	80	80	1	80	S
Async Mode:	4	4	4	4	4	4	4	4	4	4	4	4	1	4	S

Setting Mixed Type Parameters in 3-D table (bottom)

WFD5

localhost:8082/CS/wfd5 90% Search

Presamples

Trigger Type 1: Enabled	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	1	1	S
Trigger Type 1: Waveform Count	1	1	1	1	1	1	1	1	1	1	1	1	1	1	S
Trigger Type 1: Waveform Length	560000	560000	560000	560000	560000	560000	560000	560000	560000	560000	560000	560000	1	560000	S
Trigger Type 1: Waveform Gap	16	16	16	16	16	16	16	16	16	16	16	16	1	16	S
Trigger Type 2: Enabled	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	1	1	S
Trigger Type 2: Waveform Count	4	4	4	4	4	4	4	4	4	4	4	4	1	4	S
Trigger Type 2: Waveform Length	800	800	800	800	800	800	800	800	800	800	800	800	1	800	S
Trigger Type 2: Waveform Gap	159200	159200	159200	159200	159200	159200	159200	159200	159200	159200	159200	159200	1	159200	S
Trigger Type 3: Enabled	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	1	1	S
Trigger Type 3: Waveform Count	1	1	1	1	1	1	1	1	1	1	1	1	1	1	S
Trigger Type 3: Waveform Length	800	800	800	800	800	800	800	800	800	800	800	800	1	800	S
Trigger Type 3: Waveform Gap	16	16	16	16	16	16	16	16	16	16	16	16	1	16	S
Error Threshold: Corrupt Data	10	10	10	10	10	10	10	10	10	10	10	10	1	10	S
Error Threshold: Unknown TTC	10	10	10	10	10	10	10	10	10	10	10	10	1	10	S
Error Threshold: DDR3 Overflow	7549747	7549747	7549747	7549747	7549747	7549747	7549747	7549747	7549747	7549747	7549747	7549747	1	7549747	S

Setting Boolean Parameters in 3-D table

WFD5 Channels

localhost:8082/CS/wfd5c

Search

Status | ODB | Messages | Alarms | Programs | Sequencer | Config | Help

WFD5 Riders

Configure 1 crate: Crate # 1 Configure multiple crates: Crate # 1 VALIDATE SELECTION

WRITE to ODB DATA LAST WRITTEN on 0

Channel Setting: Enabled

/Equipment/AMC1301/Settings/RiderXX/ChannelXX/Enabled

Ch #	Rider 01	Rider 02	Rider 03	Rider 04	Rider 05	Rider 06	Rider 07	Rider 08	Rider 09	Rider 10	Rider 11	Rider 12	WFD5 ##	Value	SET
00	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	1	1	S
01	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	1	1	S
02	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	1	1	S
03	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	1	1	S
04	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	1	1	S

Experiment GM2 Help Wed Oct 04 2017 18:27:53 GMT+0200 (CEST)

Setting Integer Parameters in 3-D table

Setting String Parameters in 3-D table

WFD5 Channels

localhost:8082/CS/wfd5c

Search

Status | ODB | Messages | Alarms | Programs | Sequencer | Config | Help

WFD5 Riders

Configure 1 crate: Crate # 1 Configure multiple crates: Crate # 1 VALIDATE SELECTION

WRITE to ODB DATA LAST WRITTEN on 0

Channel Setting: FPGA Firmware Version Required

/Equipment/AMC1301/Settings/RiderXX/ChannelXX/FPGA Firmware Version Required

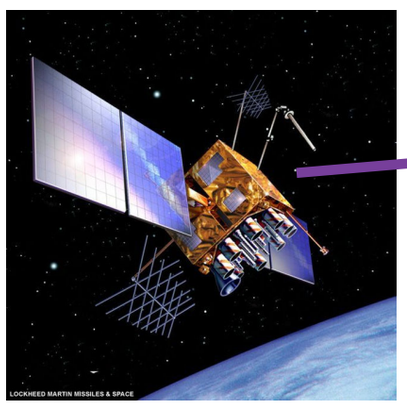
Ch #	Rider 01	Rider 02	Rider 03	Rider 04	Rider 05	Rider 06	Rider 07	Rider 08	Rider 09	Rider 10	Rider 11	Rider 12	WFD5 ##	Value	SET
00	2.3.2	2.3.2	2.3.2	2.3.2	2.3.2	2.3.2	2.3.2	2.3.2	2.3.2	2.3.2	2.3.2	2.3.2	1	2.3.2	S
01	2.3.2	2.3.2	2.3.2	2.3.2	2.3.2	2.3.2	2.3.2	2.3.2	2.3.2	2.3.2	2.3.2	2.3.2	1	2.3.2	S
02	2.3.2	2.3.2	2.3.2	2.3.2	2.3.2	2.3.2	2.3.2	2.3.2	2.3.2	2.3.2	2.3.2	2.3.2	1	2.3.2	S

Two table layout with common selectors

BACKUP

MasterGM2 frontend

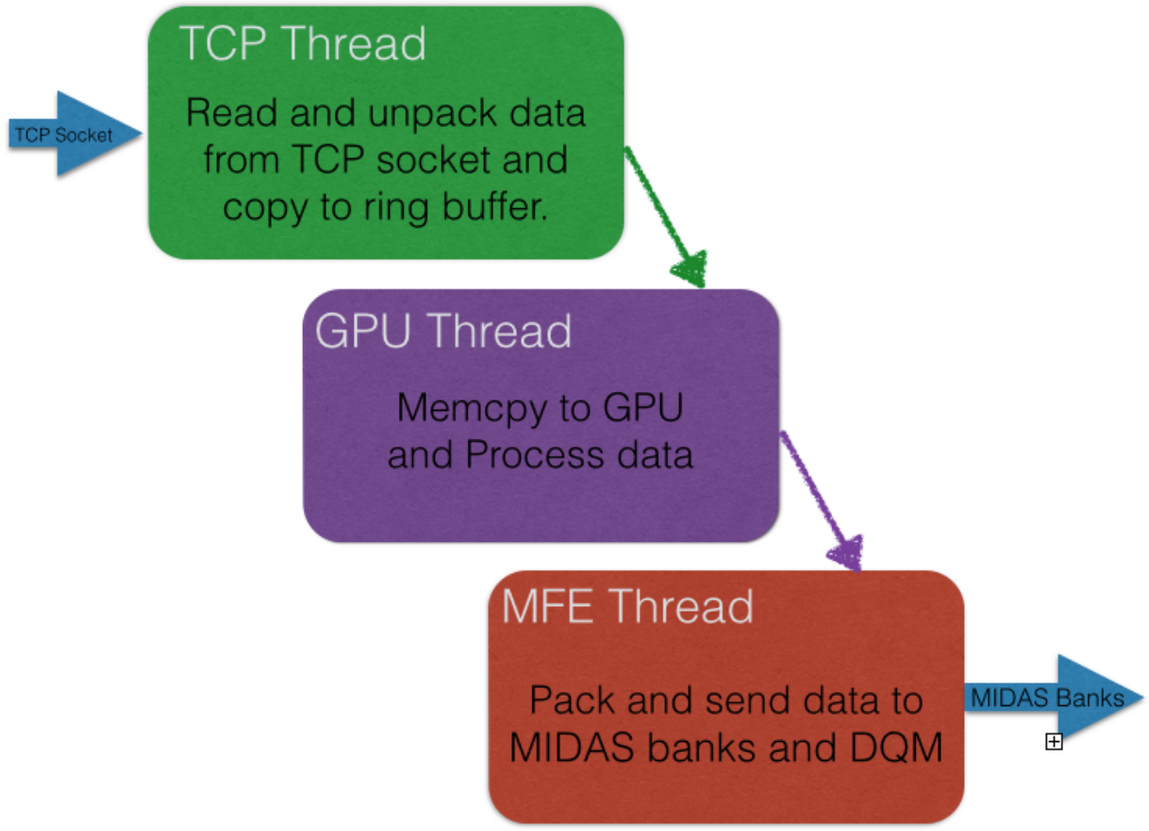
- Communicates with other frontends using RPC calls.
- Provides begin of run and end of run RPCs to all frontends.
- Provides end of fill trigger to synchronous frontends.
- Configures clock and control system.
- Reads trigger times from Meinberg GPS unit and writes them to a MIDAS bank.

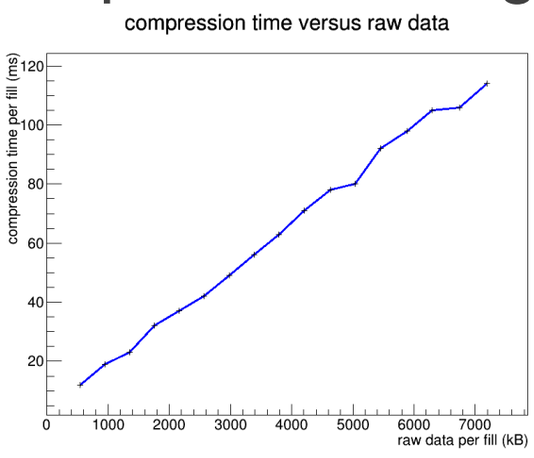


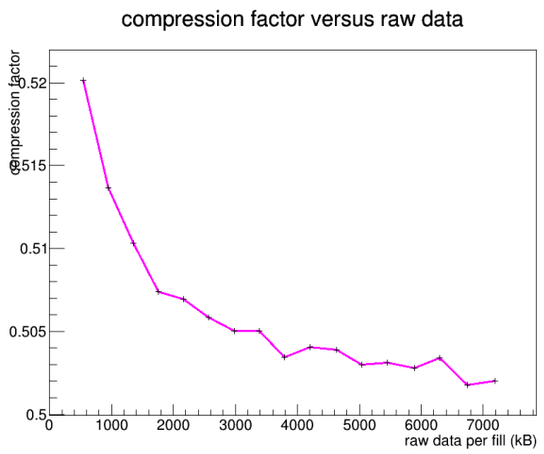
```
Bank:GPS0 Length: 20(I*1)/5(I*4)/5(Type) Type:Unsigned Integer*4
   1-> 0x000018c2 0x580838bc 0x5dcabfb0 0x580838bc 0x5dd48308
```

AMC13 Frontend

- 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.
- F
- F

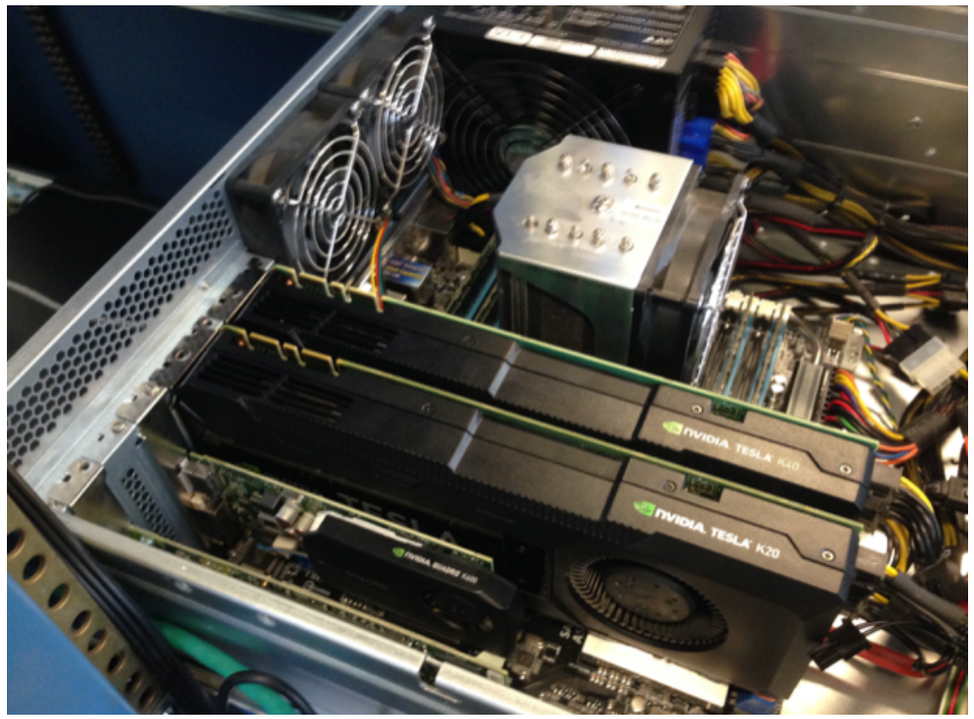






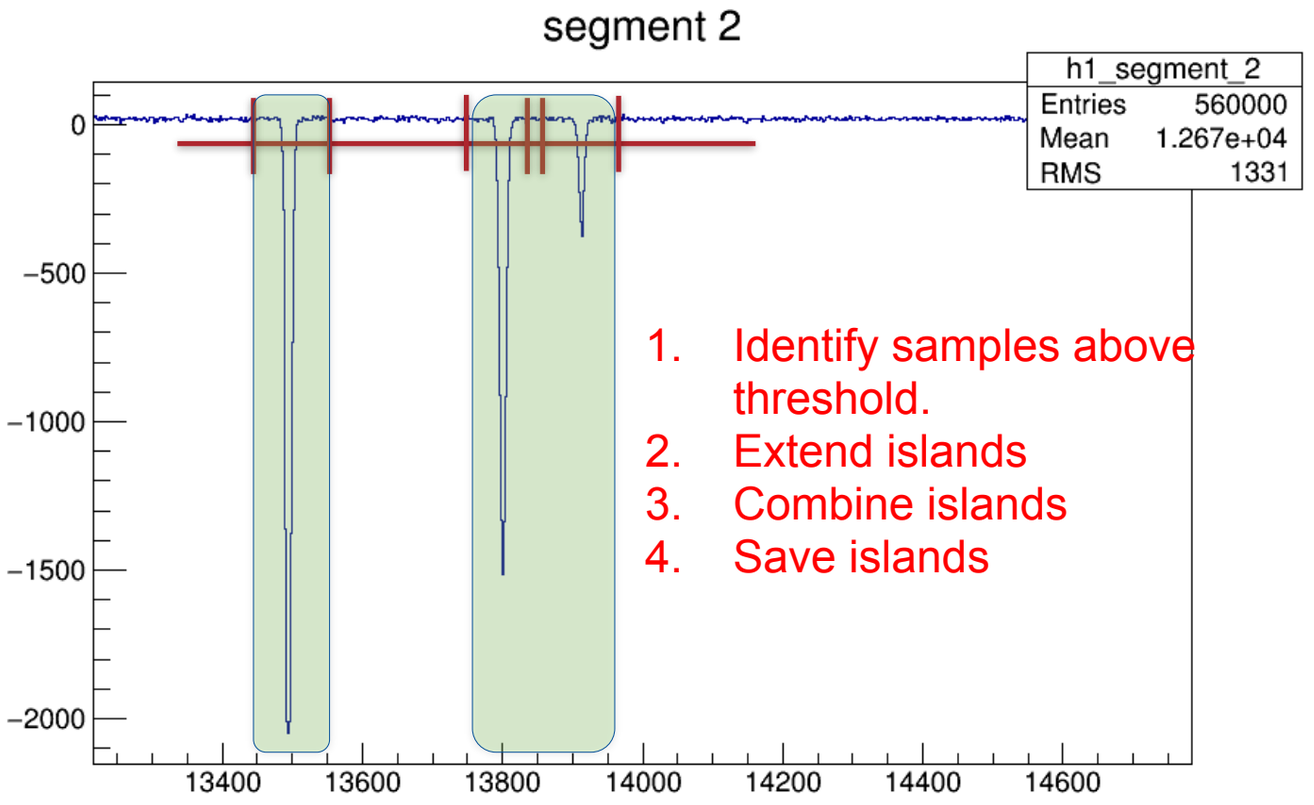
GPU Processing

- 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.



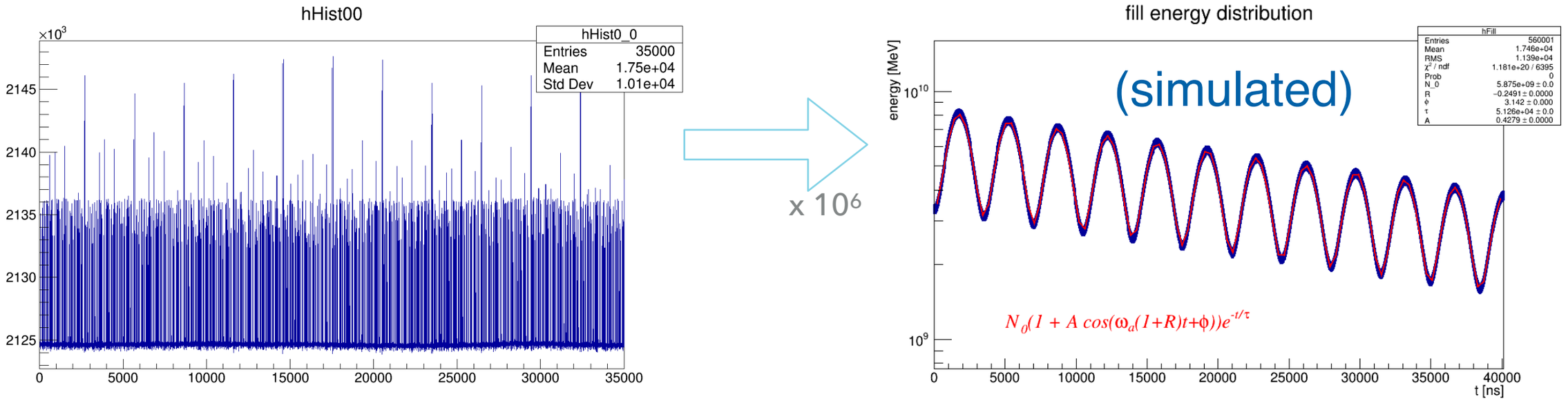
T-Method

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



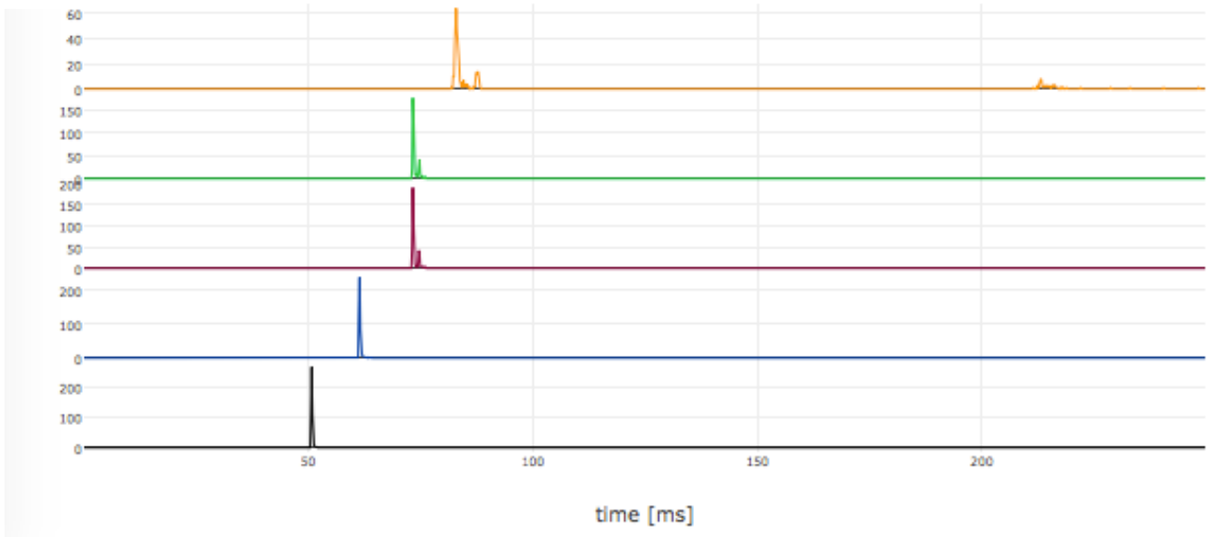
Q-method

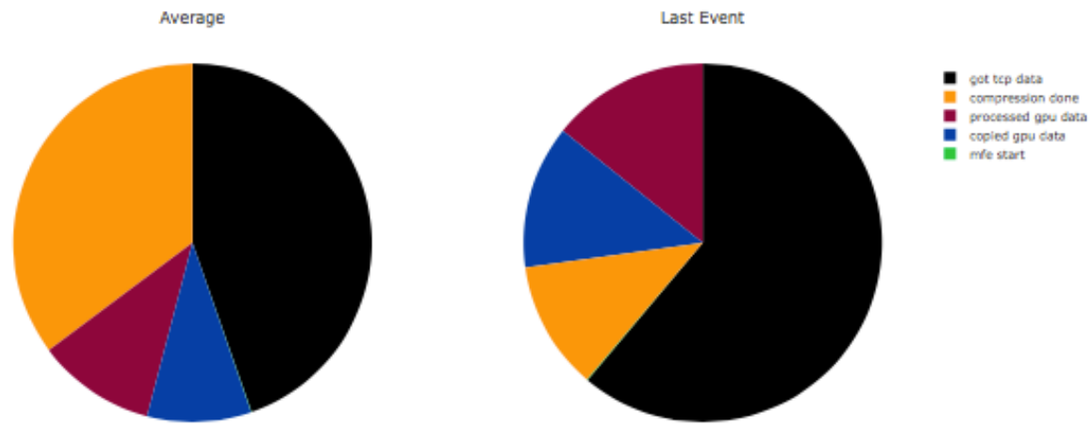
- Full waveforms are decimated in time and summed over many fills to create a histogram that is saved in the data file.
  - i.e. If we decimate in time by 10 and flush every 100 fills, we reduce the data rate by a factor of 1000, so from 20 GB/s to 20 MB/s.
- Use smaller bins at lower times and wider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



Processing time

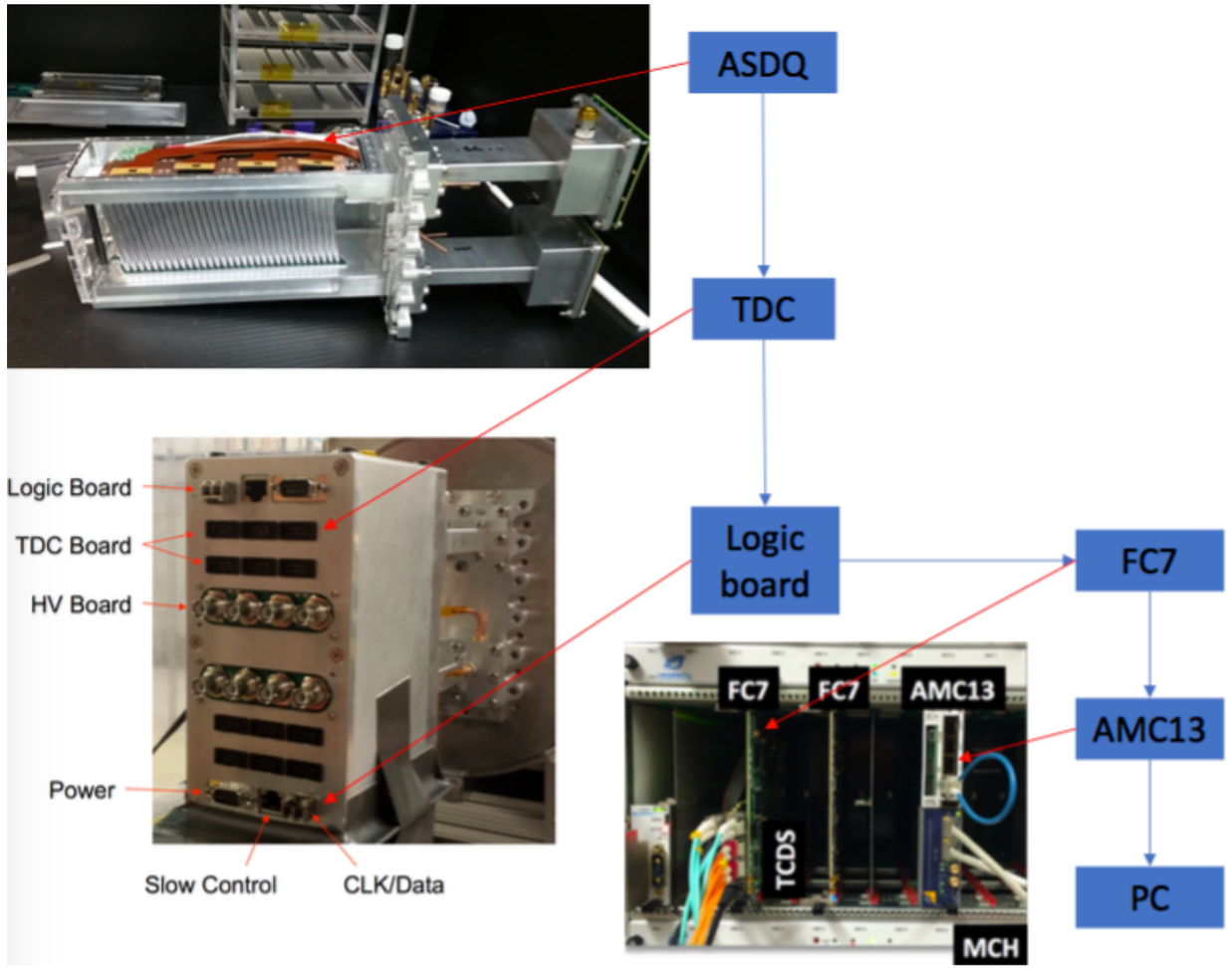
- Must process each event in 83 ms to keep up with average beam rate of 12 Hz.
- Most time is spent reading data from TCP socket and copying it to the GPU.
- Processing time in the GPU is very small.





Tracker Frontend

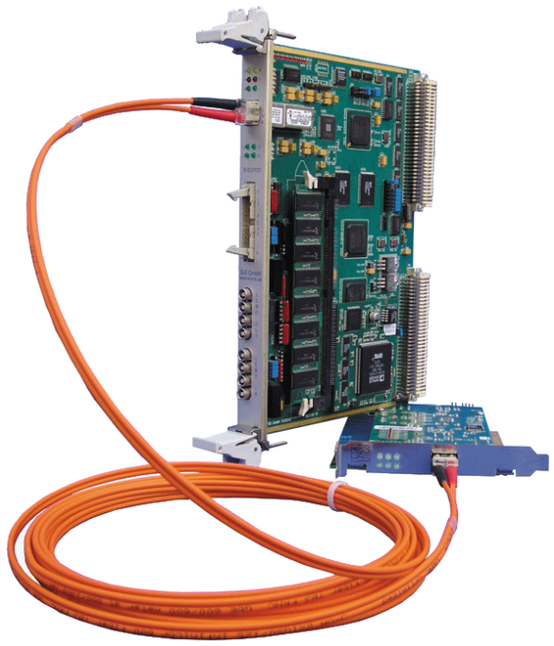
- Three tracker stations will be read via one uTCA crate.
- Reads data from AMC13.
- Instead of digitizers, data comes from multihit TDCs that are read via FC7 cards.

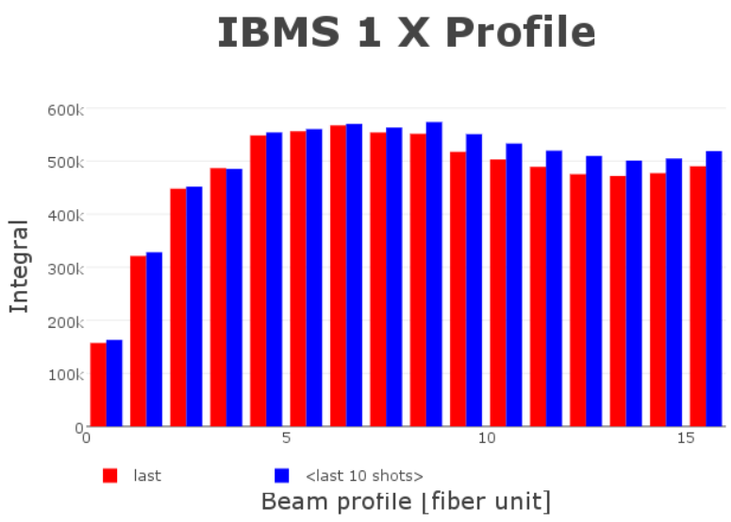


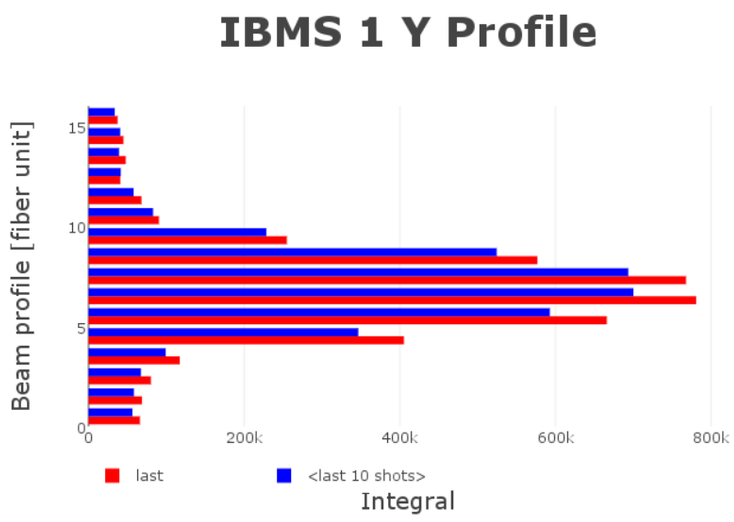
(Thanks R. Chislett for the diagram)

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.







Slow Controls

- DAQ includes six SCS3000 mscb devices.
- 24 beaglebones reading slow control data from calorimeters.
- HV and LV frontends for tracker system.
- Slow frontend reading magnet properties from IFIX via an OPC client.
- Beamline frontend periodically reading output of beam components from database.
- Slow control data is stored in a Postgres database and displayed using a custom Django web display.