

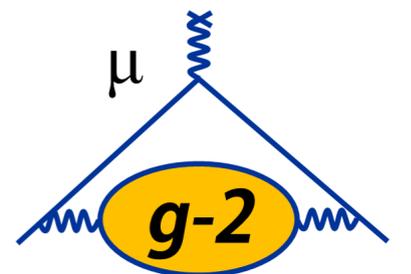


# Nearline with focus on fast physics processing

**Kim Siang Khaw**

**Muon  $g-2$  Computing Review**

**Nov 08, 2016**



# Content

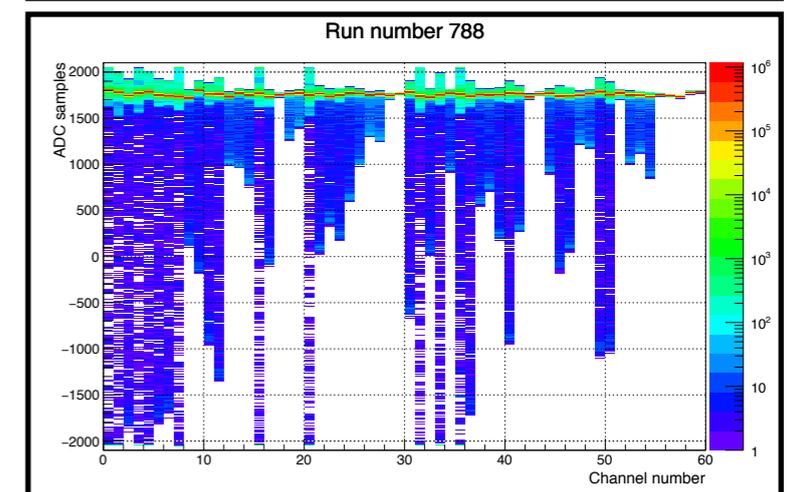
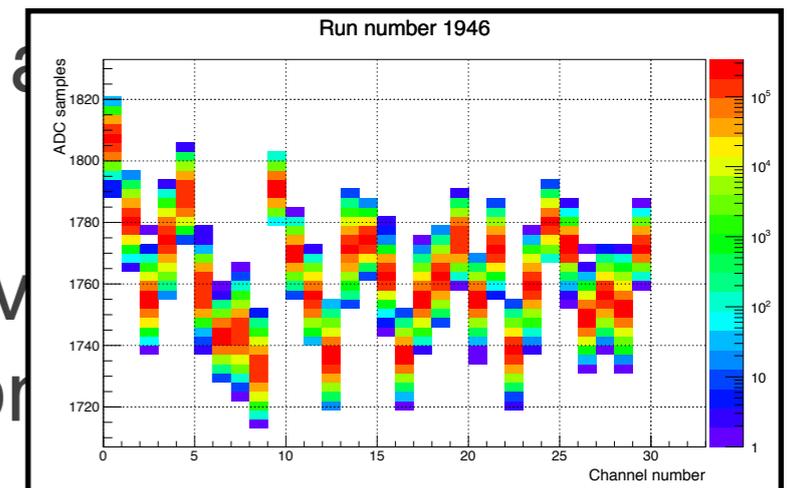
- Introduction: why nearline?
- Physics and technical goals
- Timing test from SLAC test run
- Optimization and multithreading
- Projected processing time
- Summary

# Introduction: why nearline?

- Based on our experience at SLAC test run, having a fast turnaround physics analysis is extremely crucial
- Help to diagnose performance and hardware problems with the detectors, especially during the first few months of data taking
- Differs from online data quality monitor (DQM) as it entails user interactivity
- Differs from offline data analysis as it has no overhead for file staging from tape, file transfer and grid job submission
- **And it will use the full Muon g-2 offline codes (we do not reinvent our own wheel!)**

# Introduction: why nearline?

- Based on our experience at SLAC test run, having a fast turnaround physics analysis is extremely crucial
- Help to diagnose performance and hardware problems with the detectors, especially during the first few months of data taking
- Differs from online data quality monitor (DQM) and interactivity
- Differs from offline data analysis as it has no over staging from tape, file transfer and grid job submission
- Example of contributions at SLAC
  - provided prompt feedback for digitizer information
  - provided prompt feedback for laser intensity settings
  - provided prompt feedback for SiPM gain settings
  - detected missing data from a digitizer crate



# Physics and technical goals

- Physics goals
  - to provide a fast handle for the equalization of SiPM gains
  - to provide “fresh” calibration data for online DQM and offline analysis
  - to provide out-of-the-box “wobble” plot (without corrections)
- Technical goals
  - maximize utilization of multi-core processors (using Thread Building Block - Intel TBB)
  - optimize codes for parallelization at the calorimeter-level
  - data processing rate matching DAQ rate of 12 Hz (not required but good to have)

# From DAQ to storage

## MIDAS DAQ

Run Status	
Run 2149	Start: Sun Jun 5 16:58:00 2016 Stop: Sun Jun 5 17:04:22 2016
Stopped	Alarms: On Restart: No Data dir: /data/slac/
Start	Experiment Name: SLAC
	Rider status: 0
17:04:33 [Logger,INFO] Run #2149 stopped	

Equipment				
Equipment	Status	Events	Events[/s]	Data[MB/s]
EB	Ebuilder@g2be	1880	0.0	0.000
MasterGM2	MasterGM2@g2be	0	0.0	0.000
AMC1301	AMC1301@g2calo	1880	0.0	0.000
AMC1302	AMC1302@g2calo	1880	0.0	0.000
AMC1303	AMC1303@g2calo	1880	0.0	0.000
Temperature	Ok	28	0.9	0.000

SAM Database

SAM

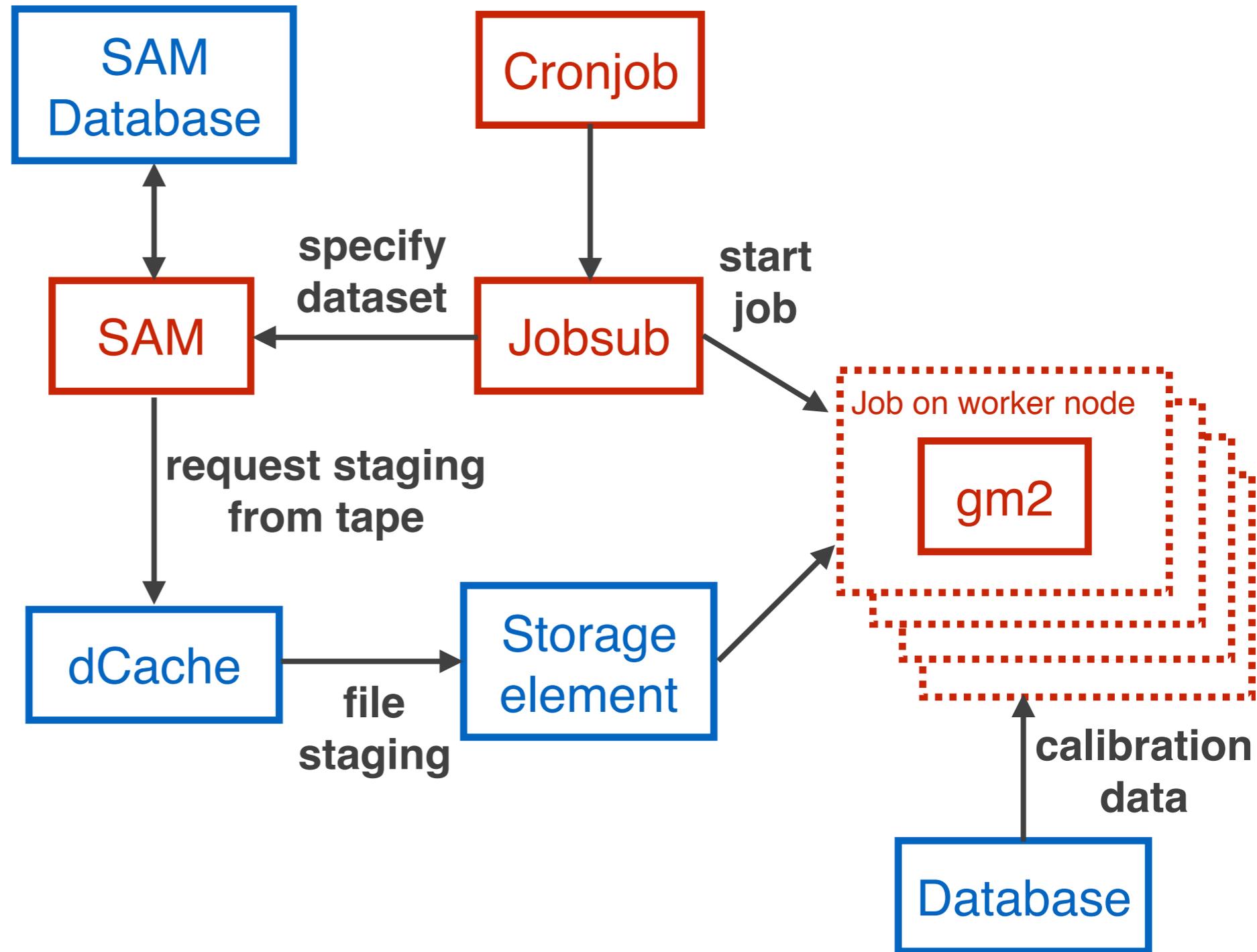
local RAID

file transfer

dCache

FTS

# From storage to job submission



# Nearline framework

## MIDAS DAQ

Run Status				
Run 2149	Start: Sun Jun 5 16:58:00 2016	Stop: Sun Jun 5 17:04:22 2016	Data dir: /data/slac/	
Stopped	Alarms: On	Restart: No		
Start	Experiment Name: SLAC	Rider status: 0		
17:04:33 [Logger,INFO] Run #2149 stopped				
Equipment				
Equipment	Status	Events	Events[/s]	Data[MB/s]
EB	Ebuilder@g2be	1880	0.0	0.000
MasterGM2	MasterGM2@g2be	0	0.0	0.000
AMC1301	AMC1301@g2calo	1880	0.0	0.000
AMC1302	AMC1302@g2calo	1880	0.0	0.000
AMC1303	AMC1303@g2calo	1880	0.0	0.000
Temperature	Ok	28	0.9	0.000

local RAID

nearline machine

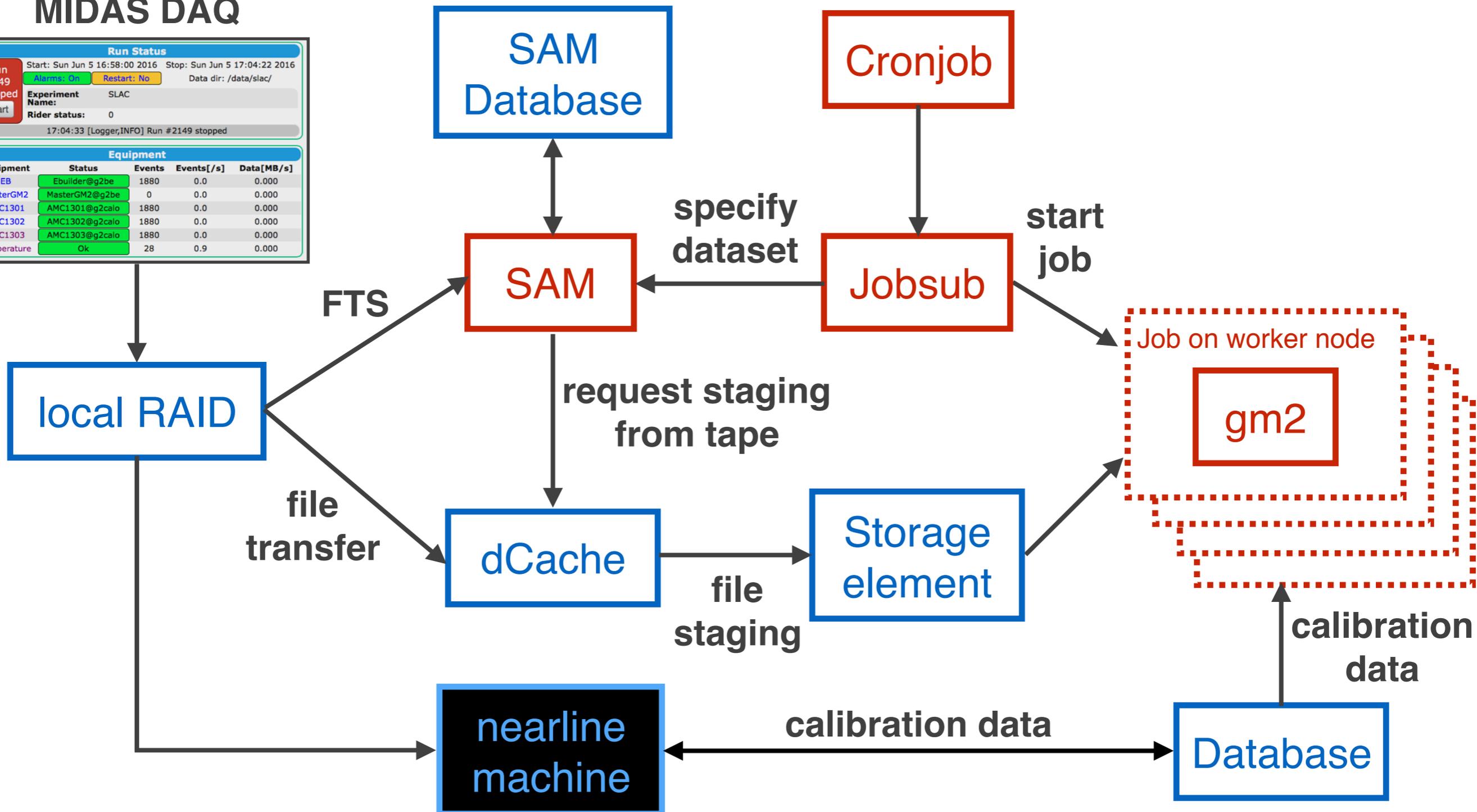
Database

calibration data

# Complete framework

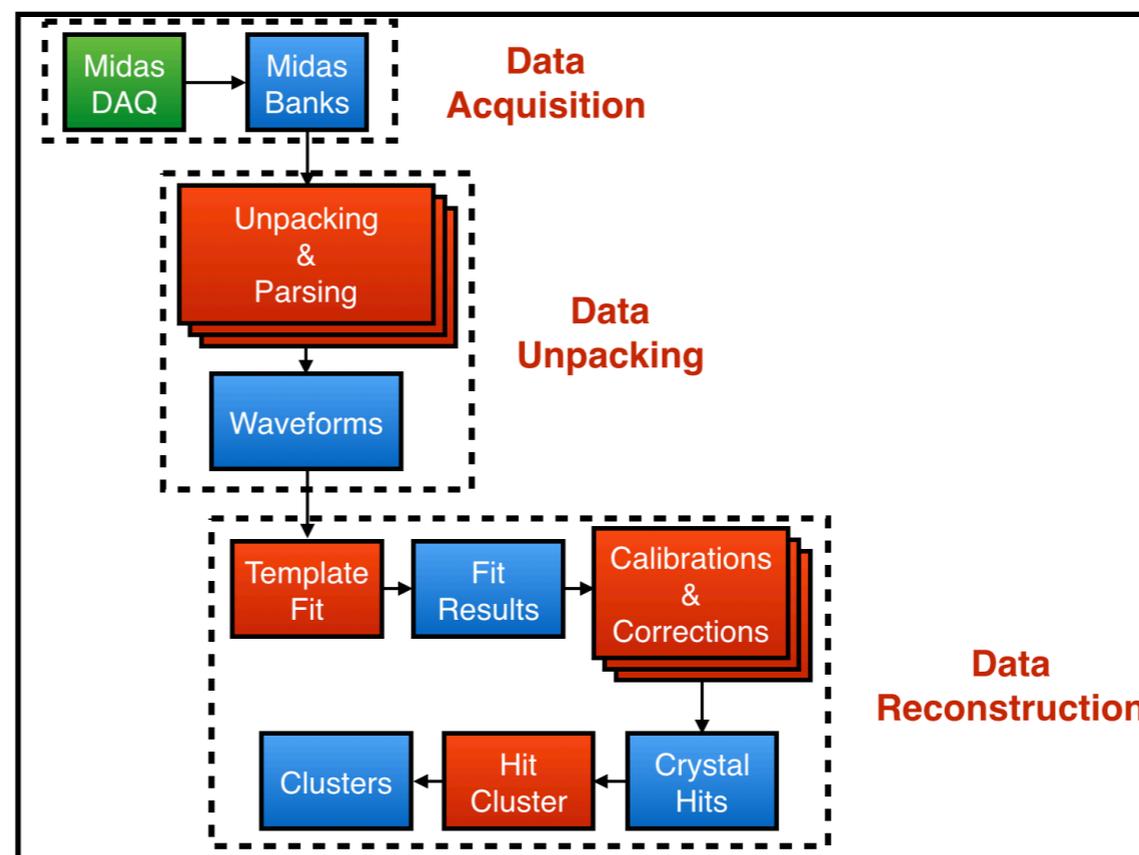
## MIDAS DAQ

Run Status				
Run 2149	Start: Sun Jun 5 16:58:00 2016	Stop: Sun Jun 5 17:04:22 2016	Data dir: /data/slac/	
Stopped	Alarms: On	Restart: No		
Start	Experiment Name: SLAC	Rider status: 0		
17:04:33 [Logger,INFO] Run #2149 stopped				
Equipment				
Equipment	Status	Events	Events[/s]	Data[MB/s]
EB	Ebuilder@g2be	1880	0.0	0.000
MasterGM2	MasterGM2@g2be	0	0.0	0.000
AMC1301	AMC1301@g2calo	1880	0.0	0.000
AMC1302	AMC1302@g2calo	1880	0.0	0.000
AMC1303	AMC1303@g2calo	1880	0.0	0.000
Temperature	Ok	28	0.9	0.000



# How fast are we right now?

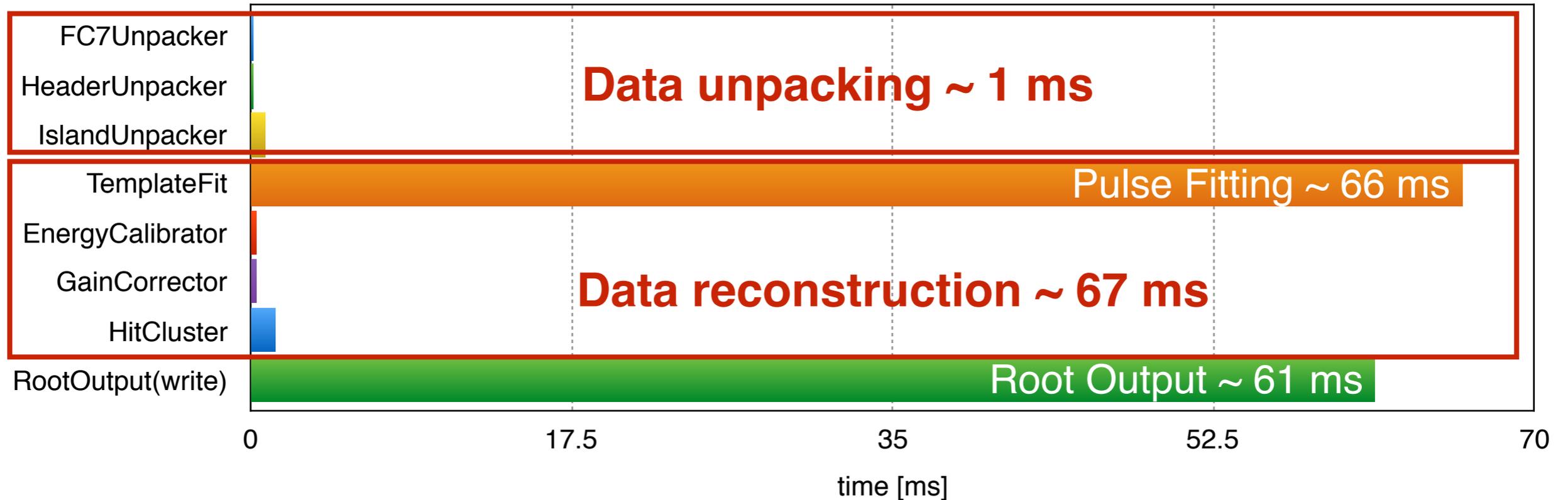
- The birth place of this idea - SLAC calorimeter test run
- We used the SLAC dataset for timing test and projected the time needed for 24 calorimeters and number of expected crystal hits
- (post-SLAC) Have been optimizing our offline codes



# How fast are we right now?

- The birth place of this idea - SLAC calorimeter test run
- We used the SLAC dataset for timing test and projected the time needed for 24 calorimeters and number of expected crystal hits (also hacked into unpacking routine to produce 24-calorimeter worth of data)
- Have been optimizing our offline codes since SLAC
  - **Data products** - removed virtual destructors for auto-generating special move functions (constructor and assignment operator) **1.1 to 4x speedup**
  - **Data unpacking** - removed redundant steps in data copying **2x speedup**

# Timing test (as of this review)



TimeTracker printout (sec)	Min	Avg	Max	Median	RMS	nEvts
Full event	0.0606531	0.0689881	0.501452	0.063798	0.0435458	100
unpackerPath:fc7Unpacker:FC7Unpacker	7.604e-05	0.000102685	0.00109262	9.0768e-05	0.000100388	100
unpackerPath:headerUnpacker:HeaderUnpacker	8.3824e-05	0.000107083	0.000941674	8.80715e-05	8.62767e-05	100
unpackerPath:islandUnpacker:OnlineIslandUnpacker	0.000642255	0.000724877	0.00135163	0.000711773	7.16665e-05	100
unpackerPath:islandFitter:TemplateFit	0.0579027	0.0661387	0.494832	0.0609873	0.0431672	100
unpackerPath:energyCalibrator:EnergyCalibrator	0.000242105	0.000267431	0.000674476	0.000253493	4.61028e-05	100
unpackerPath:gainCorrector:GainCorrector	0.000232165	0.000259864	0.000460291	0.000242228	3.48295e-05	100
unpackerPath:hitCluster:HitCluster	0.00115472	0.00130267	0.00187135	0.00128985	8.2678e-05	100
unpackerPath:TriggerResults:TriggerResultInserter	1.3608e-05	1.48846e-05	3.0534e-05	1.41715e-05	2.91178e-06	100
end_path:outfile:RootOutput	2.015e-06	2.57126e-06	1.4517e-05	2.267e-06	1.51856e-06	100
end_path:outfile:RootOutput(write)	0.0597246	0.0613416	0.0658187	0.0611854	0.0011628	100

# Current processing time (single-core)

\* SSD/HDD = 0.7

Conditions	Offline processing [ms]	*RootOutput [ms]
1 calorimeter	69	42/61
24 calorimeters	1656	1008/1464
50% less pulses (Muon g-2)	<b>828</b>	504/732
<b>Total</b>	<b>1332/1560 ms</b>	

- 0.75/0.6 Hz (if we write out everything)
- 1.1 Hz (if we skip the root file output)
- write speed ~ 10 MB/s (SSD, c.f. gm2 simulation ~ 40 MB/s)

Q: How do we go from 1 Hz towards 12 Hz?

# Current processing time (single-core)

\* SSD/HDD = 0.7

Conditions	Offline processing [ms]	*RootOutput [ms]
1 calorimeter	69	42/61
24 calorimeters	1656	1008/1464
50% less pulses (Muon g-2)	<b>828</b>	504/732
<b>Total</b>	<b>1332/1560 ms</b>	

- 0.75/0.6 Hz (if we write out everything)
- 1.2 Hz (if we skip the root file output)
- write speed ~ 10 MB/s (SSD, c.f. gm2 simulation ~ 40 MB/s)

**A: Multi-threading + multi-core Processor!**

# Multi-threading in *art* module

- Very limited support on multithreading in art
- “Event parallel” multi-threading is still work in progress
- Parallelization within a single module using Intel Threading Building Blocks (TBB)
- Have been testing multi-threading in our offline framework
  - **Data unpacking and reconstruction** - using tbb multithreading for parallelization especially for pulse fitting

Private communication  
with M. Paterno



2-3x speedup in  
processing time  
for a  
quadcore machine

# Nearline machine (candidate)

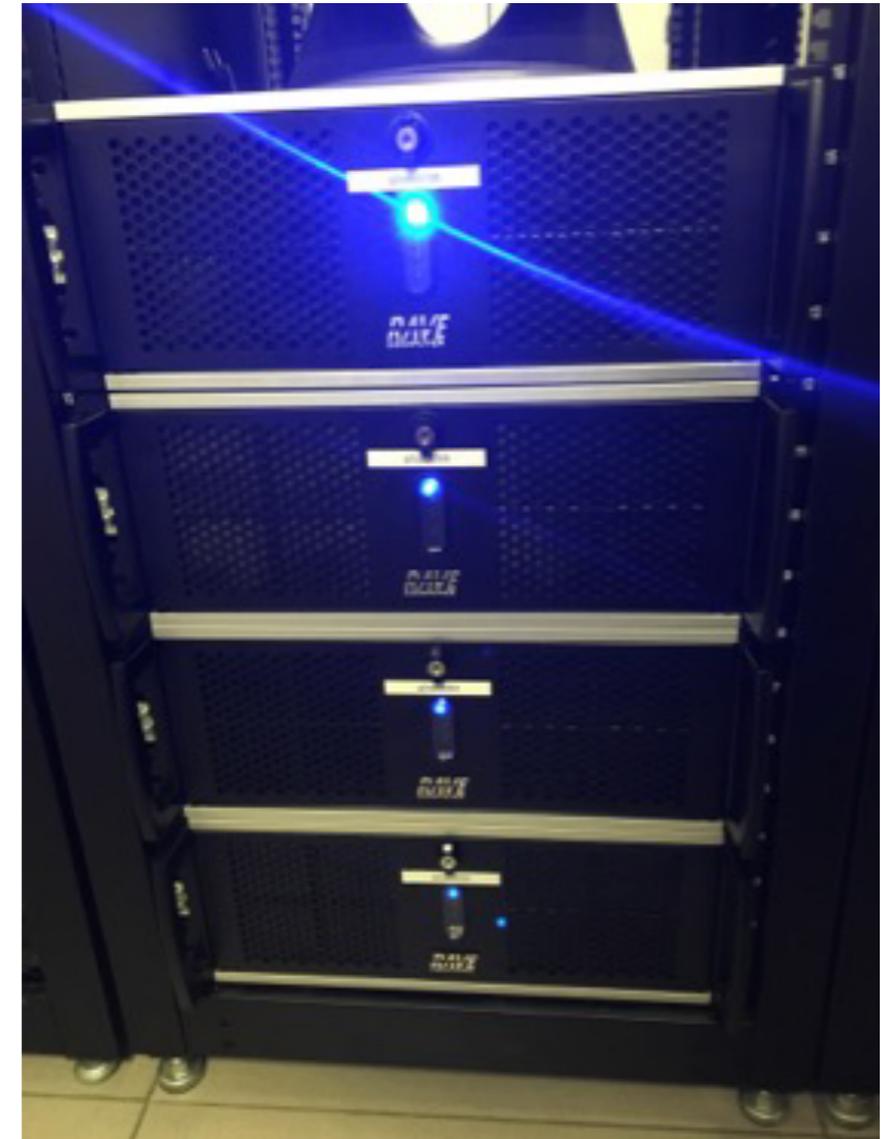
## RR-4U-UOFK-04

Rev NS

U/M EA

RR-4U-UOFK-04 TO INCLUDE:

- (1) 4U/Tower 920W 1+1 HE 8x35in Bays 2x5.25in Bays
- (1) 2U+ Passive Heatsink for LGA2011
- (1) Adaptor HDD carrier to install 2.5" HDD in 3.5" HDD tray
- (1) 4U Chassis Rail Kits SC743,745 (w/ 2pcs Black Chassis Ears)
- (1) ATX C612 Motherboard 7xPCIe Slots 2x1GbE IPMI USB3.0
- (1) Xeon E5-2667 v3 (20M Cache, 3.2-3.6 GHz) 8C/16T 135W
- (3) 3TB 3.5" Enterprise Constellation 128MB 7200RPM
- (1) 850 Pro 512GB SATA6Gb/s 2.5in SSD
- (8) 8GB DDR4-2133 ECC/REG SRx4 (64GB Total)



- We will need a machine with specs similar to our frontend machines (with supports from onsite Dell group)

# Processor (Candidates)

Compare Intel® Products

<http://ark.intel.com/compare/>

Highlight rows with differences

	Intel® Xeon® Processor E5-2667 v3 (20M Cache, 3.20 GHz)	Intel® Xeon® Processor E7-8867 v3 (45M Cache, 2.50 GHz)	Intel® Xeon® Processor E7-8890 v4 (60M Cache, 2.20 GHz)
▶ Product Name	Intel® Xeon® Processor E5-2667 v3 (20M Cache, 3.20 GHz)	Intel® Xeon® Processor E7-8867 v3 (45M Cache, 2.50 GHz)	Intel® Xeon® Processor E7-8890 v4 (60M Cache, 2.20 GHz)
▶ Code Name	Haswell	Haswell	Broadwell
<b>Essentials</b>			
▶ Processor Number	E5-2667V3	E7-8867V3	E7-8890V4
▶ Status	Launched	Launched	Launched
▶ Launch Date	Q3'14	Q2'15	Q2'16
▶ Lithography	22 nm	22 nm	14 nm
▶ Recommended Customer Price	\$2057.00	\$4672.00	\$7174.00
<b>Performance</b>			
▶ # of Cores	8	16	24
▶ # of Threads	16	32	48
▶ Processor Base Frequency	3.20 GHz	2.50 GHz	2.20 GHz
▶ Max Turbo Frequency	3.60 GHz	3.30 GHz	3.40 GHz
▶ Cache	20 MB SmartCache	45 MB	60 MB

- more cores means lower base frequency, need to figure out which works best within our budget

# Projected processing time

\* SSD

Conditions	Offline processing [ms]	*RootOutput [ms]	*PartialOutput [ms]
single core	828	504	216
4-core (25% overhead)	276		
8-core (25% overhead)	138		
16-core (25% overhead)	69		

Machine	nCores	Processing only [Hz]	+ partial output [Hz]	+ full output [Hz]
single core	1	1.2	<b>1.0</b>	0.8
SLAC machine	4	3.6	2.0	1.3
DAQ Frontend (FE)	8	7.2	2.8	1.6
FE with 16-core	16	14.5	<b>3.5</b>	1.7

# Projected processing time (faster I/O)

\*4x write speed

Conditions	Offline processing [ms]	*RootOutput [ms]	*PartialOutput [ms]
single core	828	126	54
4-core (25% overhead)	276		
8-core (25% overhead)	138		
16-core (25% overhead)	69		

Machine	nCores	Processing only [Hz]	+ partial output [Hz]	+ full output [Hz]
single core	1	1.2	<b>1.1</b>	1.0
SLAC machine	4	3.6	3.0	2.5
DAQ Frontend (FE)	8	7.2	5.2	3.8
FE with 16-core	16	14.5	<b>8.1</b>	5.1

# Summary

- Nearline analysis will be an important tool to kick start the data taking of the Muon g-2 experiment
- It will provide fast feedbacks to the DAQ & detector teams
- Current plan
  - optimization of codes for TBB multithreading
  - procurement of a nearline machine (8-core or 16-core)
  - improvement in Root file write speed (10 MB/s to 40 MB/s)
- Our current goal is to achieve 12 Hz data processing rate to match the DAQ rate
- Will do more tests with the current frontend machines before purchasing a new nearline machine