A Final Review of the Performance of the CDF Run II Data Acquisition System

William Badgett
Fermilab
for the CDF Collaboration
Computing in High Energy and Nuclear Physics 2012
New York City, New York, U.S.A.

"The King is Dead! Long Live the King!"

Introduction – Contents

- Quick Review of CDF Run][DAQ
- Performance, Records, Efficiency
- A look at event rates and dynamic prescaling
- Upgrades over the years: Bus *vs* Serial Link
- Experience with online computing & platforms
- Evolution of control computing
- Computing in a radiation environment
- Plea for common frameworks and conclusions

Design overview

Dataflow of CDF "Deadtimeless" Trigger and DAQ

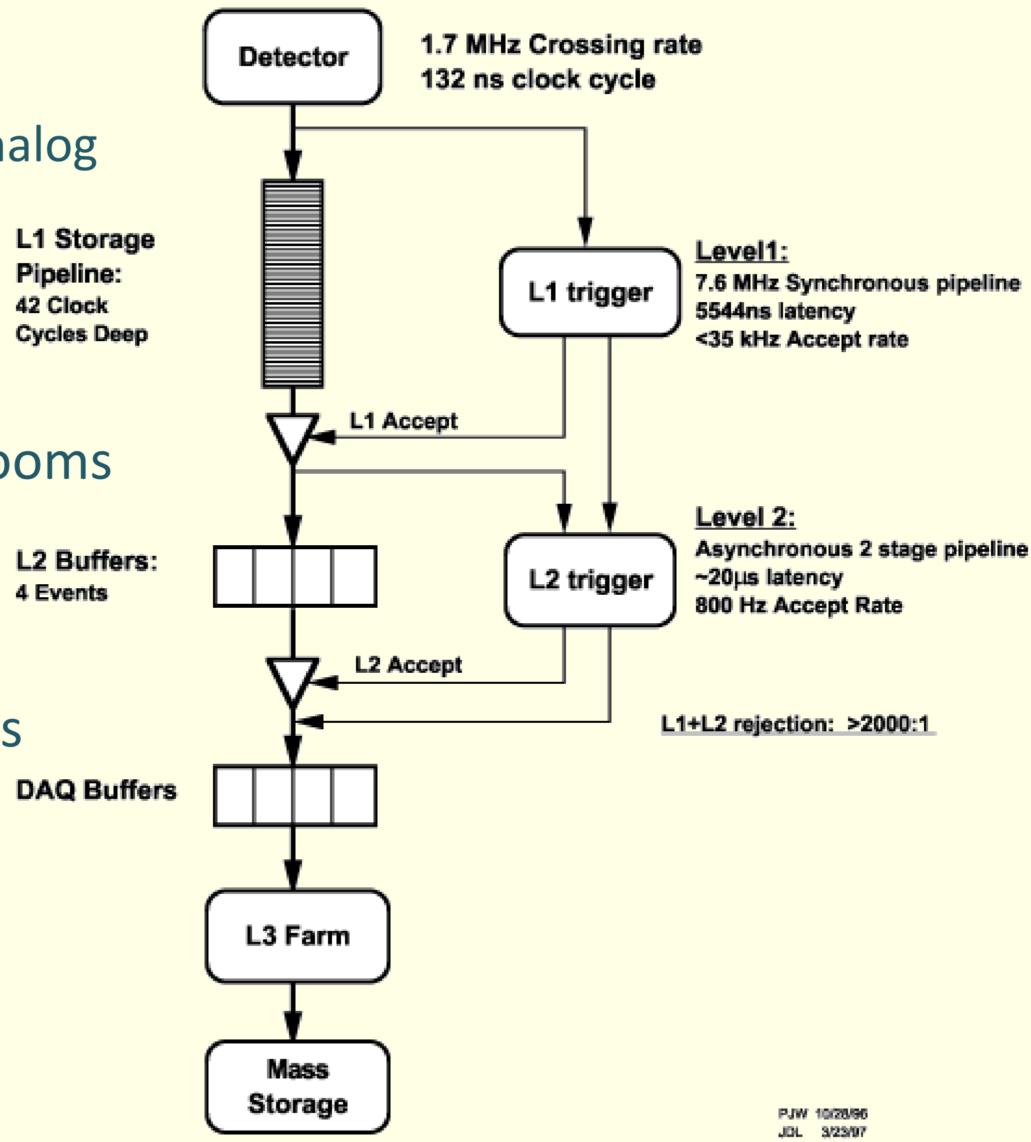
L1 Synchronous with Beam crossing

- Digitize input signals
  - For silicon, sample and hold in analog pipeline
- Push data along digital pipeline
- In parallel, form Level 1 trigger primitives
- Send primitive data to counting rooms to form trigger objects
- Form global trigger decision

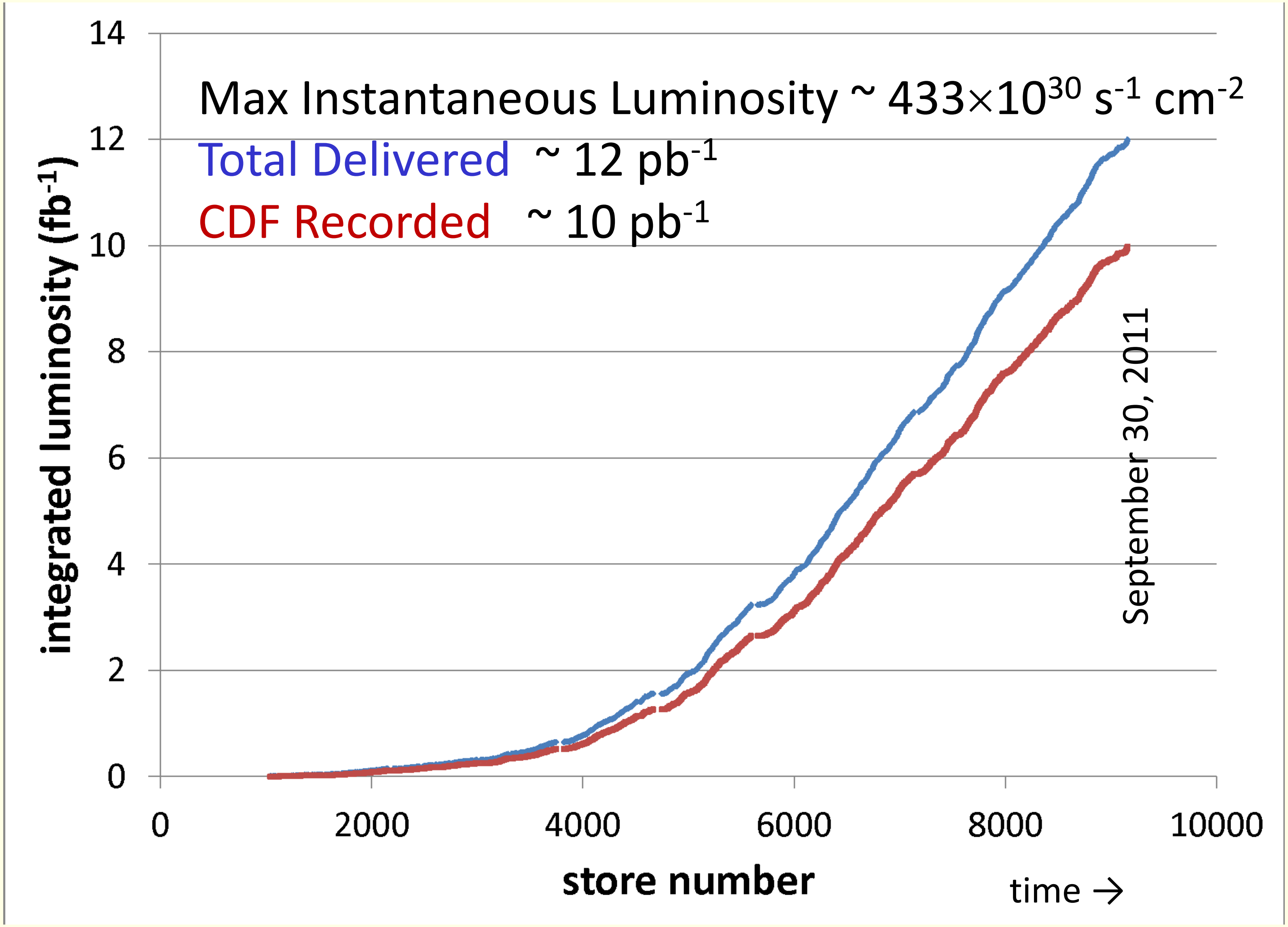
L2 Asynchronous

- Latch and hold in front end buffers
- Separate trigger data path
- Hybrid hardware and software

L3 Software Farm with offline type selection all in software



Records and Data Taken



Book Keeping – Run II Totals

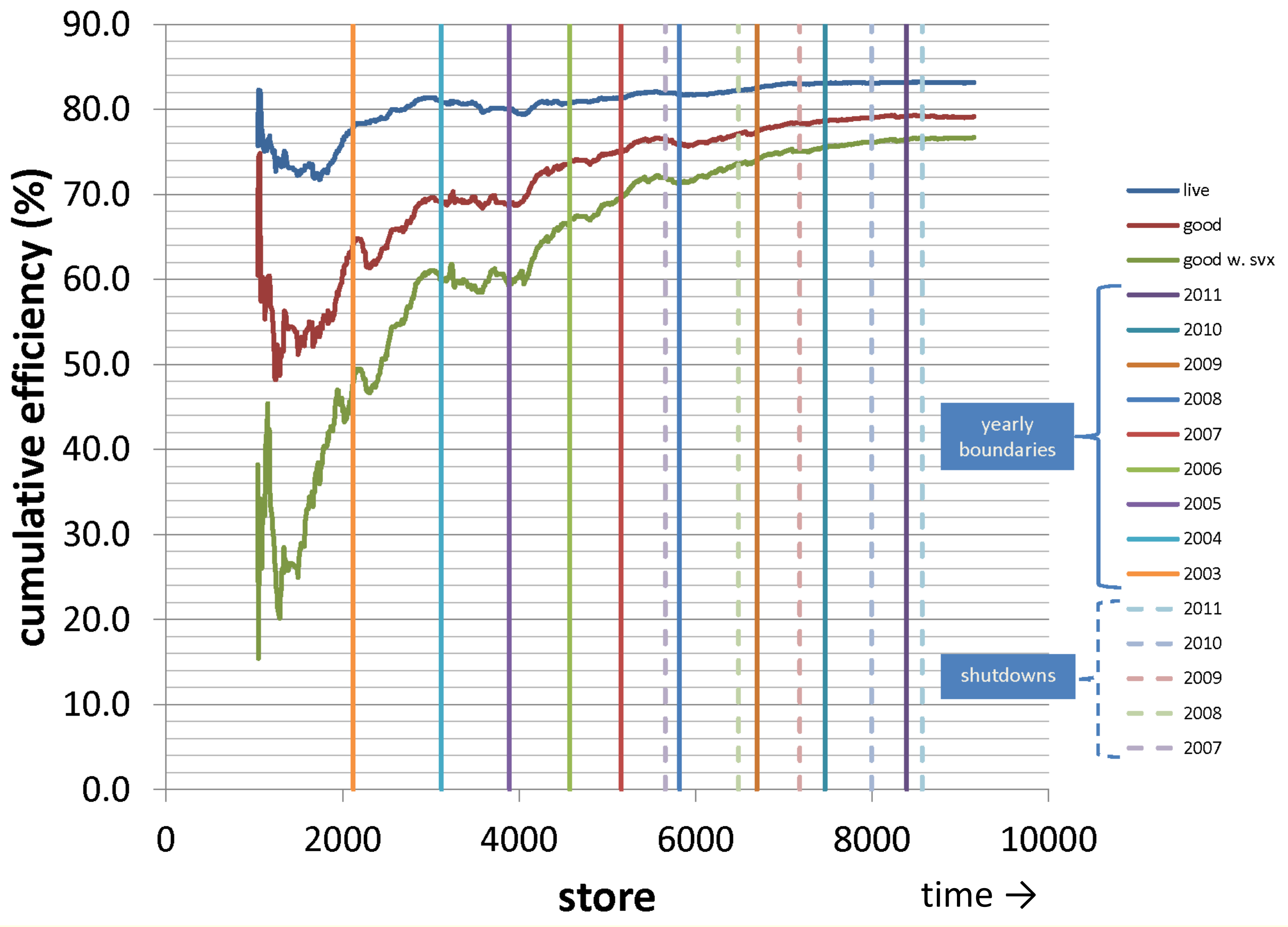
Level	Events	Rate	Rejection Factor
Beam Crossings	258,224,721,879,337	1.7 MHz	
Level 1 Accepts	2,382,669,811,547	15.7 kHz	108.4
Level 2 Accepts	50,250,240,098	330.8 Hz	47.4
Level 3 Accepts “to tape”	12,086,159,337 ~150 kB per event	79.6 Hz	4.2

Records and Limitations

Level	Avg Rate	Max Rate*	Limitation
1	17.6 kHz	40 – 50 kHz	L2 processing time; vertex finding; shower max data transfer; danger of wire bond damage
2	439 Hz	950 Hz	TDC hit readout central tracker; VME bus data transfer
3	105 Hz	400 Hz+	Offline computing resources

* Capabilities strongly inter-dependent on luminosity, beam conditions, physics selection

Cumulative Data Taking Efficiency



Performance Breakdown, "dirty laundry"

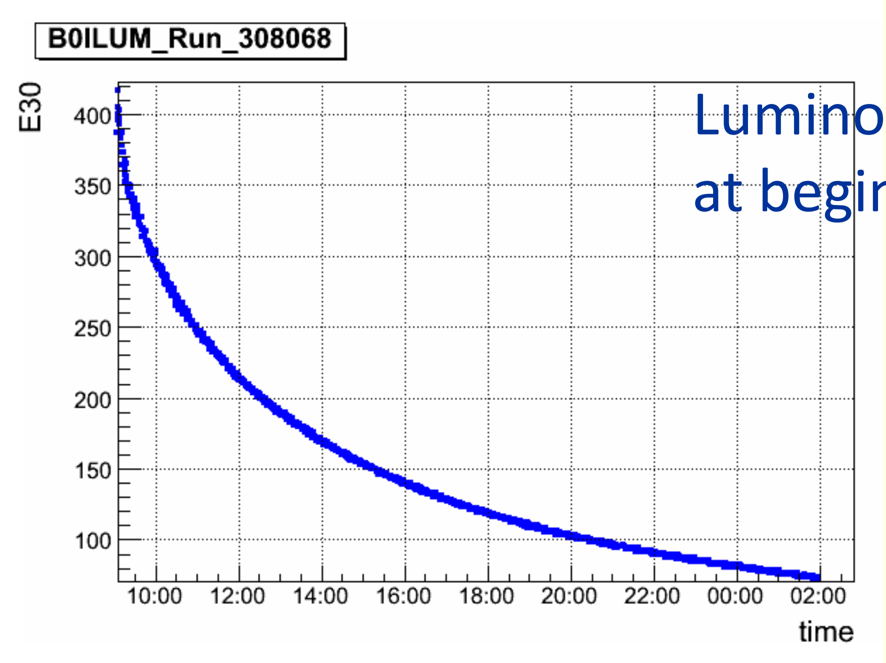
Category		Hours	Lumi, pb^{-1}	Lumi, %	
Total Delivered		42,255	11,994.6	100%	
Total Recorded		36,490	9,997.4	83.2%	
Intrinsic Deadtime *		1,344	782.0	6.5%	← next
System faults and conditions		4,420	1,179.4	9.8%	
Category	**Events**	**Hours**	**Lost Lumi pb^{-1}**	**Lumi Lost, %**	
HV Problems**	2970	598	200.3	1.667%	
Startup	1899	265	197.1	1.643%	
L2 Trigger	1115	253	51.6	0.430%	
Event Builder	944	274	49.6	0.413%	
Shower Max R/O	534	139	48.4	0.404%	← later
Silicon R/O	872	153	44.5	0.371%	
Beam Losses	692	227	44.4	0.371%	
18 More Categories 0.1% – 0.3% luminosity losses					

* Deadtime due to DAQ, trigger and selection physics vs. system capability

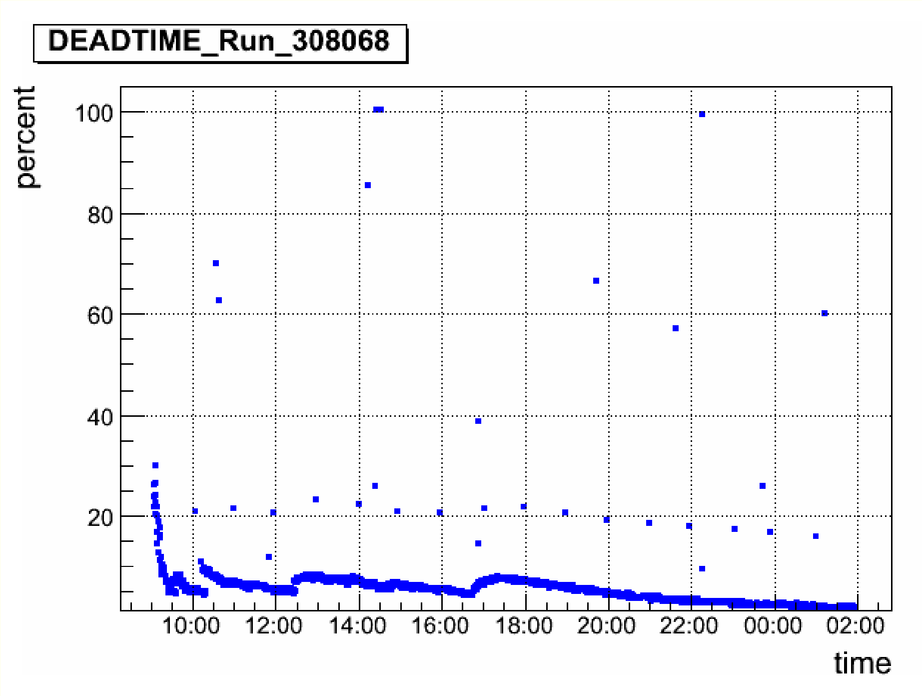
** Top three contenders, all supplies in Collision Hall

Sum is not 100% due to unattributed downtimes

Luminosity Evolution



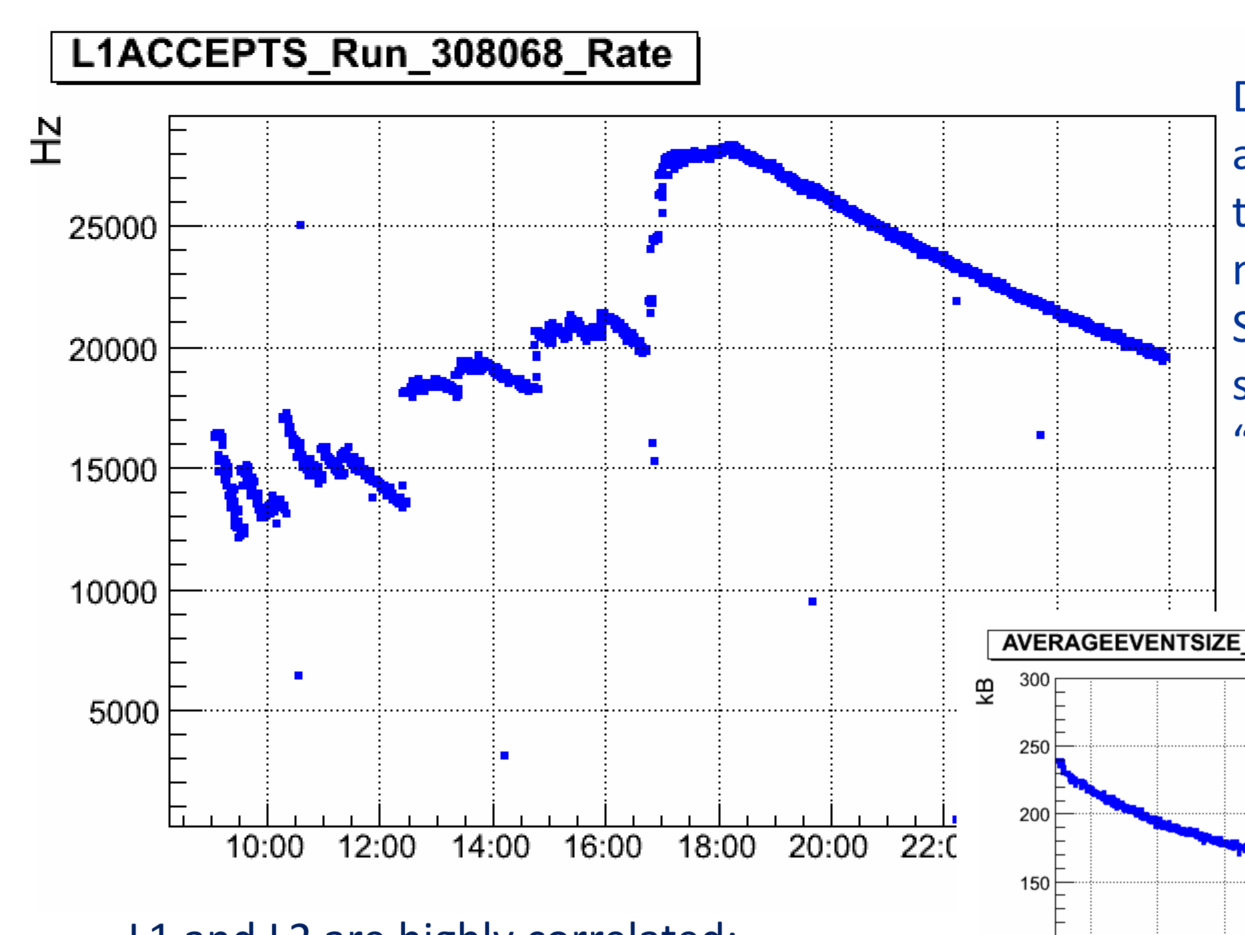
Luminosity drops rapidly at beginning of store (fill)



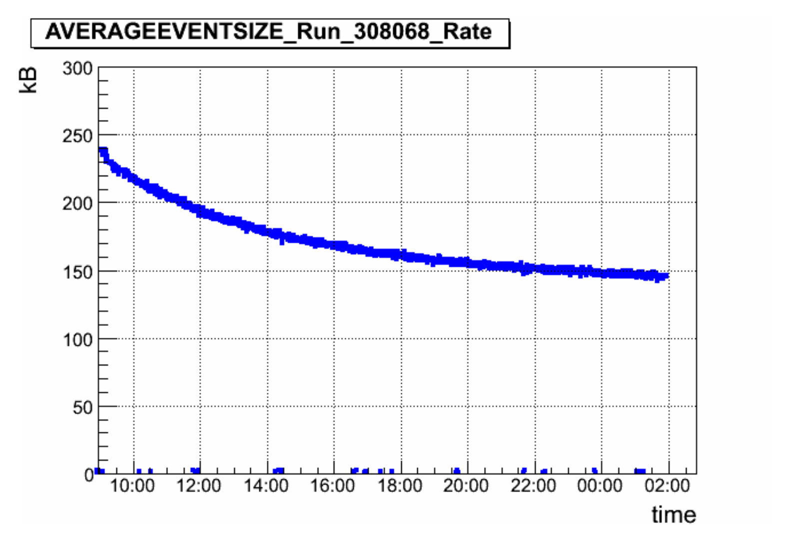
Highest luminosities saturated system – produces deadtime

Physics choices must be made – data taking efficiency vs. physics yield – a modest inefficiency yields copious B physics in right rate hadronic modes, low P_T muons, even Higgs searches

Dynamic Prescaling and Rate Evolution



Dynamic prescaling algorithms automatically open valve with time, using redundant safety mechanisms to avoid overflow System is pushed in a sophisticated way to maximize “physics througput”



L1 and L2 are highly correlated; occupancies decline with luminosity drops, allowing higher throughput later in the store (fill)

CDF DAQ and Trigger Upgrades

- CDF Run II straddled old VME bus readout and high speed serial links worlds
  - Intermediate VME bus readout before shipping over fiber link
- All upgrades took advantage of increasingly cheaper, faster and more standard links, both for increased throughput and reliability
  - ATM Event builder replaced with dedicated gigabit ethernet switch
  - Central L2 trigger process replaces custom bus with sLink and commercial PCs
  - L2 Calorimeter trigger replaces parallel I/O with sLinks
  - Silicon Vertex Trigger, ditto
  - Also advantage taken for faster, larger FPGAs

DAQ Control High Level Software Experience

- Top Level RunControl and Monitoring Performed by stand-alone Java Applications running on Linux SL4 PCs
  - Excellent experience; Linux PCs impervious to crashing
  - Java: ease of programming ; but device access hard
  - But system crashes replaced by Java Virtual Machines hang-ups and mystery crashes
  - Java and OO well suited for HEP experiment with subdetectors – inherit and reuse
  - Persistency is Oracle 10 (from 8) – well worth it
  - Java makes Threads very easy to use
    - Good: Lots of flexibility, parallel processing
    - Bad: Tempted to use lots of threads; ~ 12 persistent, plus ephemerals
  - Java GUIs operated only with private network, so:
  - Is there a need to move to a web interface?

DAQ Control, low level hardware end

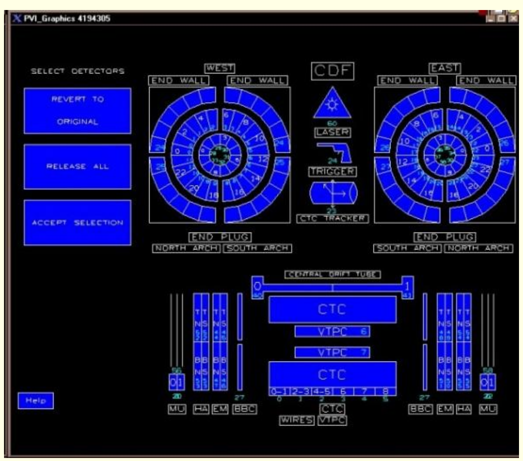
- Front end hardware control and configuration MVME Motorola processors running VxWorks
  - Crate processor conducted readout over VME bus
  - Bus transfer was an important readout rate limitation, impossible to reduce latency to < ~ 200 μsec with a fully loaded crate
  - In spite of the eccentric nature of VxWorks, nice to have an operating system simple enough to understand most of what was happening
  - Primary system readout limitation due to hit processing times in the TDCs and the unexpectedly large scaling of hits vs. luminosity
  - Good bye to VME…

RunControl Through the Ages (personal experience)

CDF Run I (0) RunControl designed to run on VT100 terminals

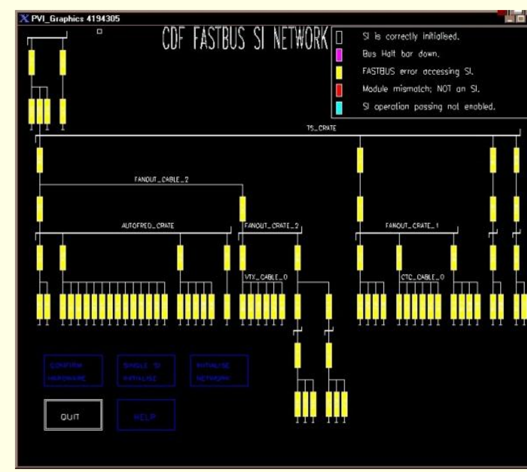
Tightly coupled to Fastbus branch network hardware, tied to VMS operating system

Fortran for low and high level interfaces

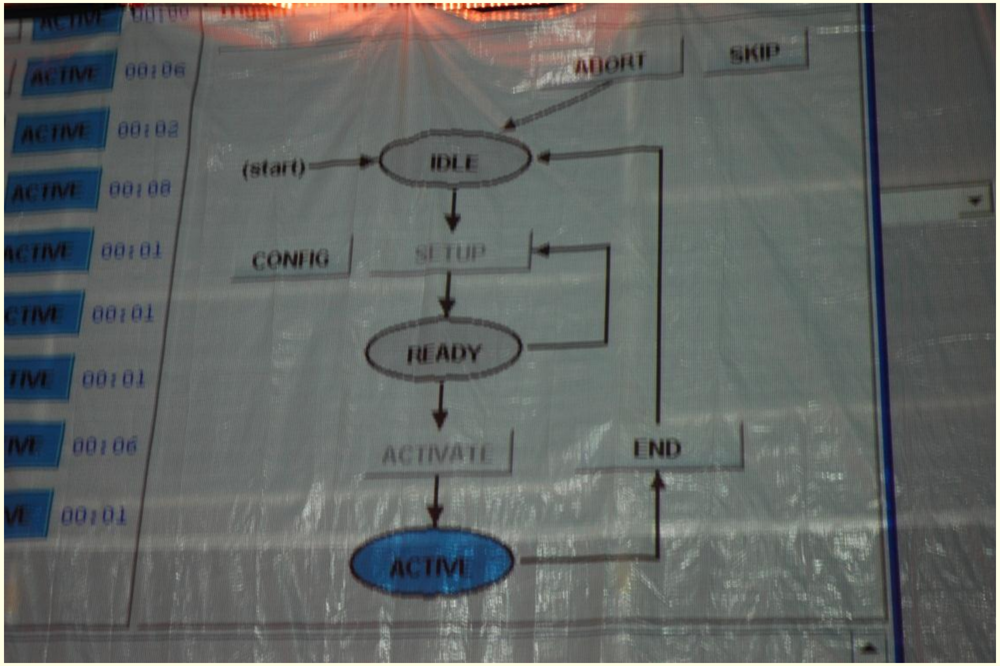


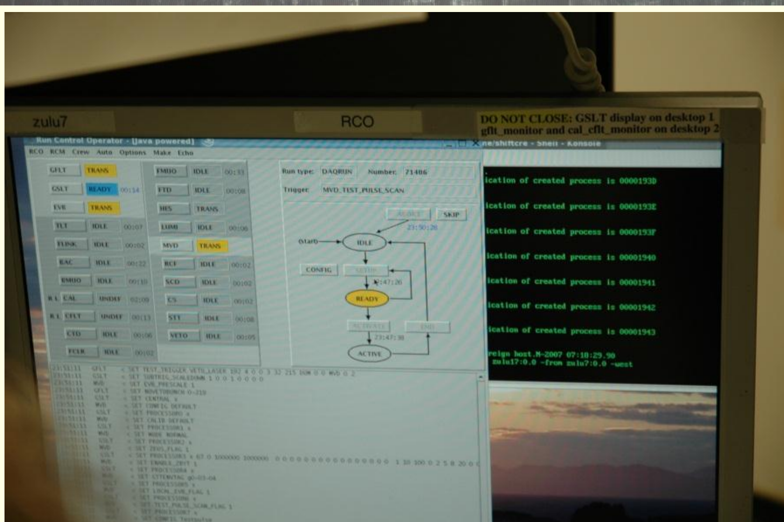
Early GUI attempt for selecting subdetectors – ran on more advanced terminals with graphics capabilities

J.Patrick



RunControl History, Zeus (2)



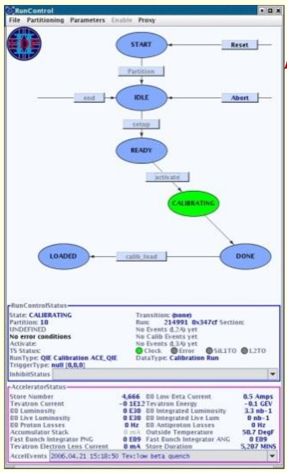


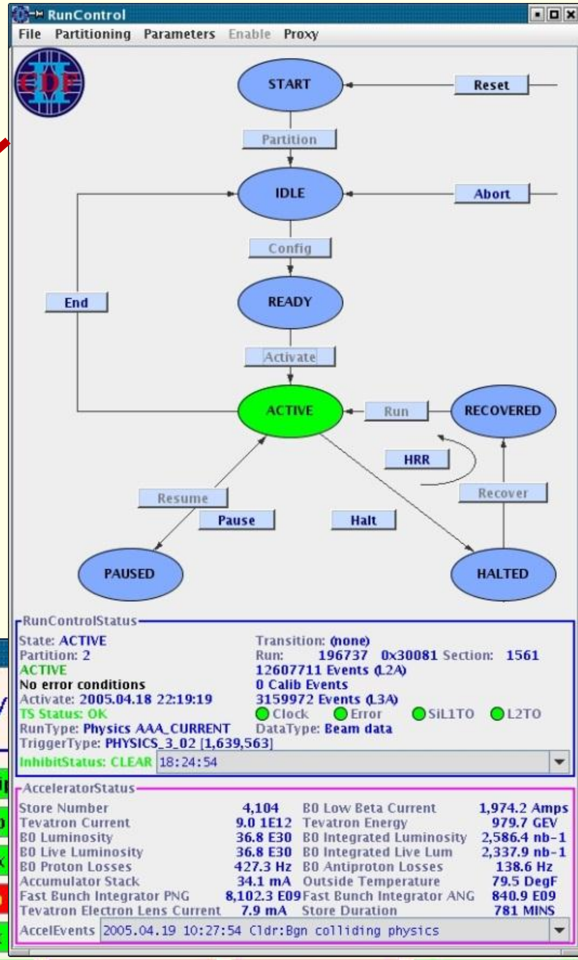
- Zeus main RunControl (2) – distributed command with little knowledge of subdetectors
- Not coupled to front end hardware at all; ran on VMS but easy to port
- TCL/TK for GUI and subdet write their own
- C and Occam for front end control of hardware; VME crates processors connected by hypercube of links (inmos transputers)

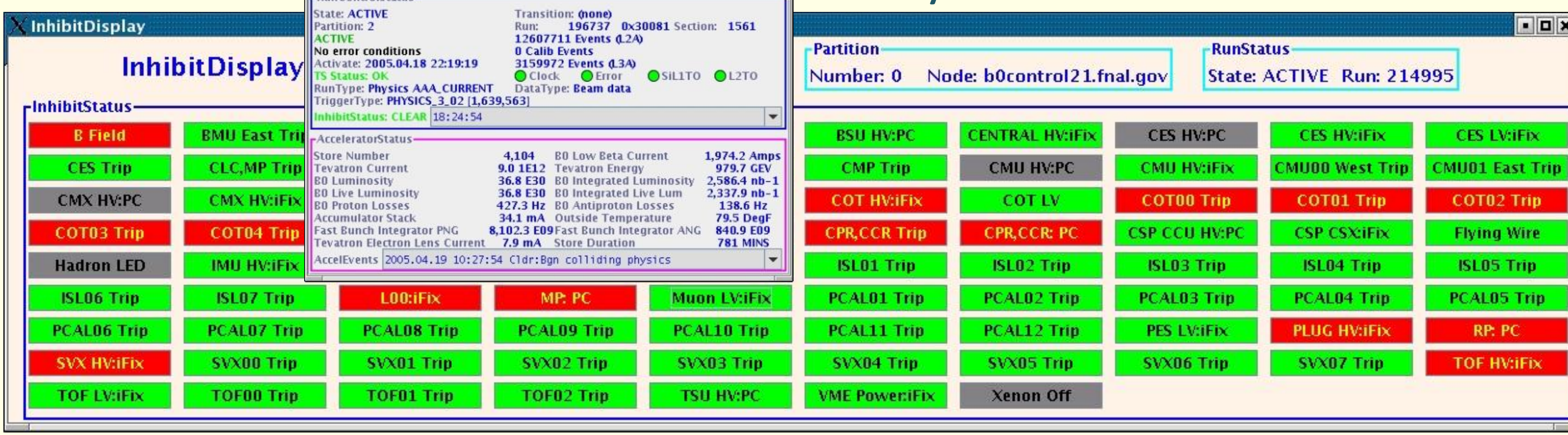
C. Youngman, J. Milewski, D. Boscherini

RunControl History, CDF II

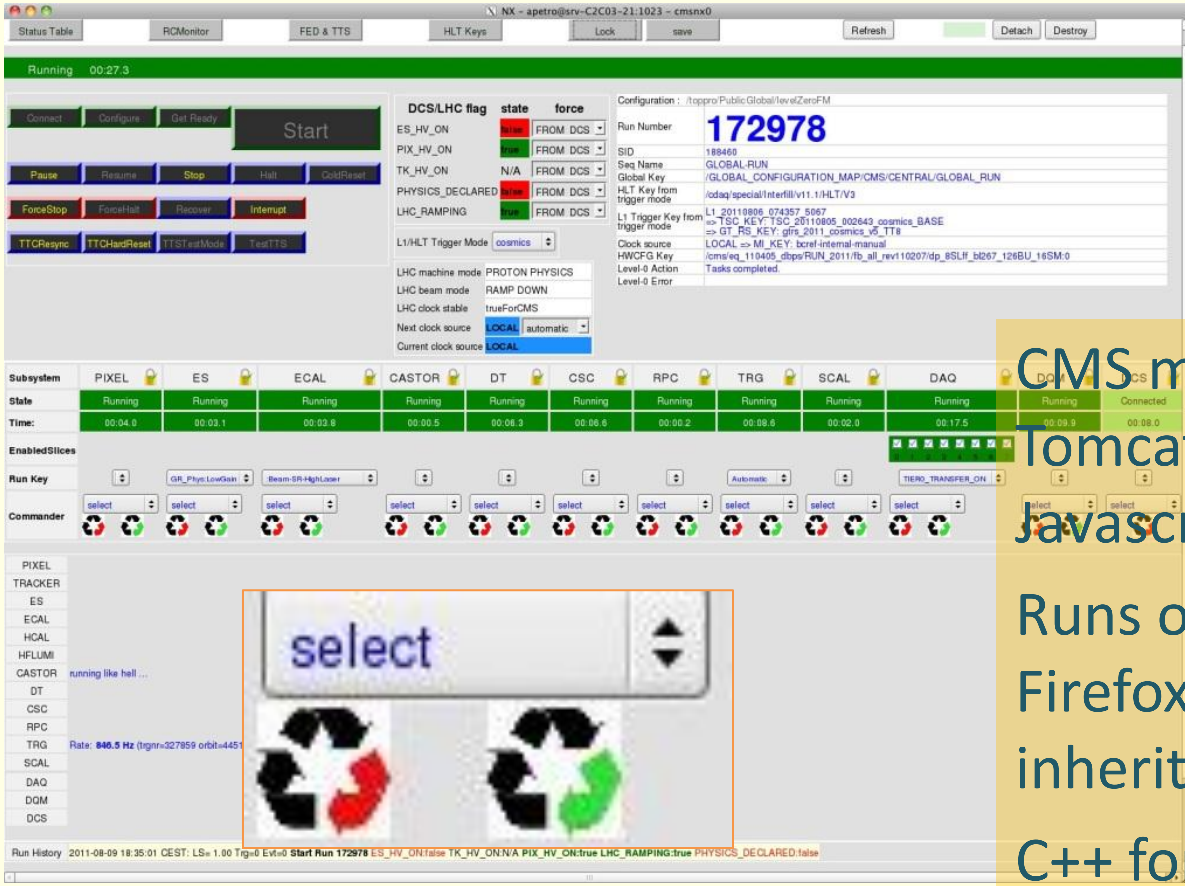
- CDF II RunControl – pure Java control and display with OO
- Ran on Linux
- C on VxWorks for front-end control; embedded MVME processor served by ethernet
- Subdetectors inherit and easily extend state machines







RunControl history, CMS



CMS main RunControl – Tomcat/Java backend with Javascript/HTML web display

Runs on any browser, usually Firefox on Linux; subdetectors inherit and extend for local RC

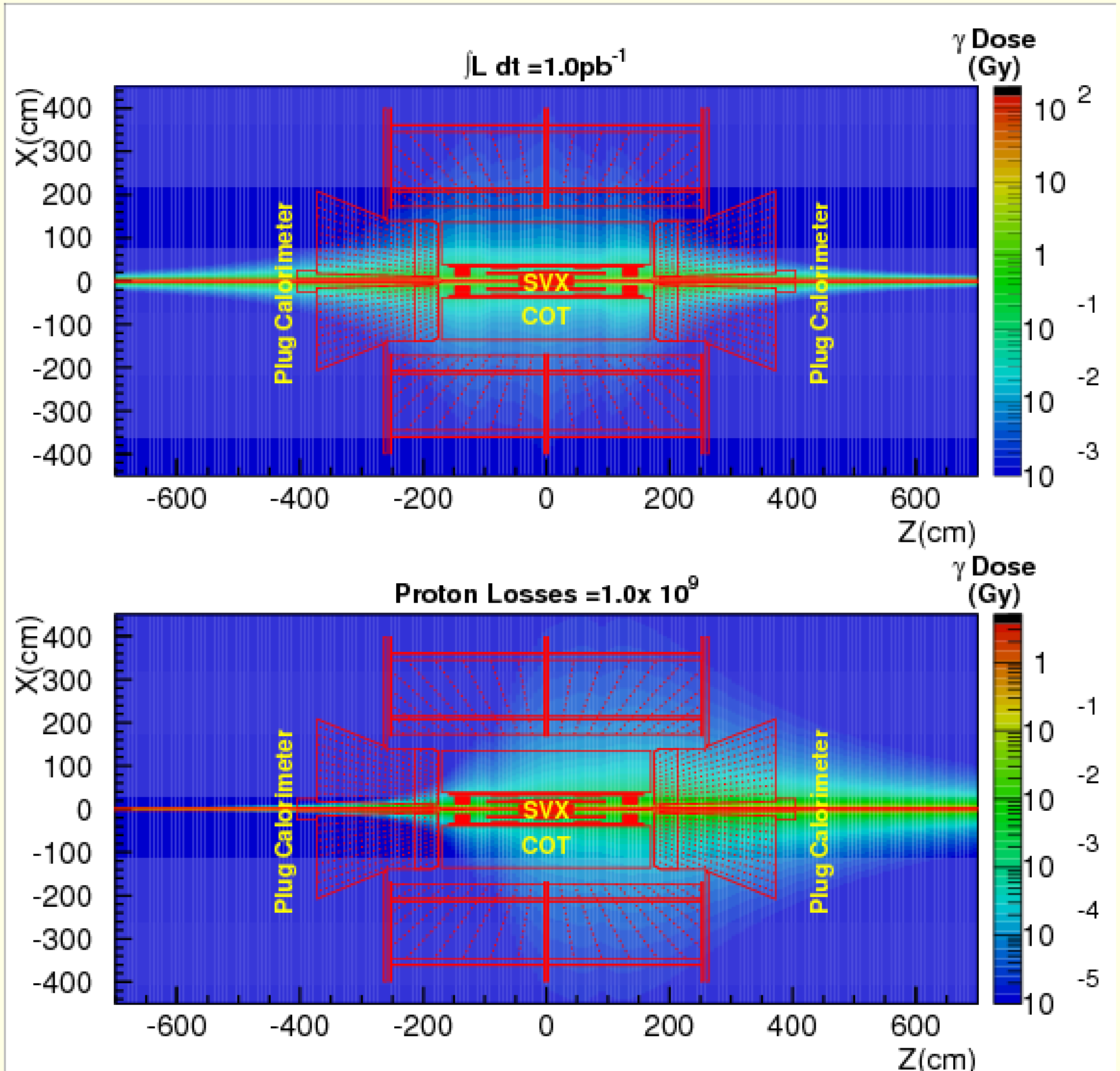
C++ for front-end control *xDAQ*; VME crates serviced by bridge to PCI on Linux

What is the future?

A. Petrucci, H. Sakulin, A.,Oh

See talks earlier this week

Computing in a Radiation Environment!



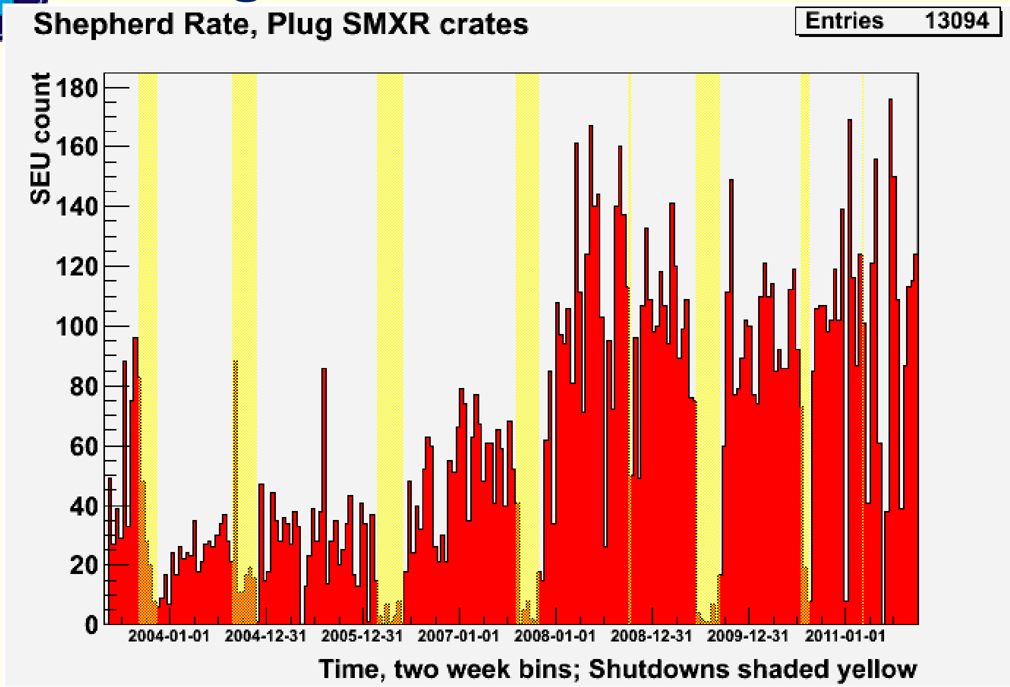
R. Tesarek, et al

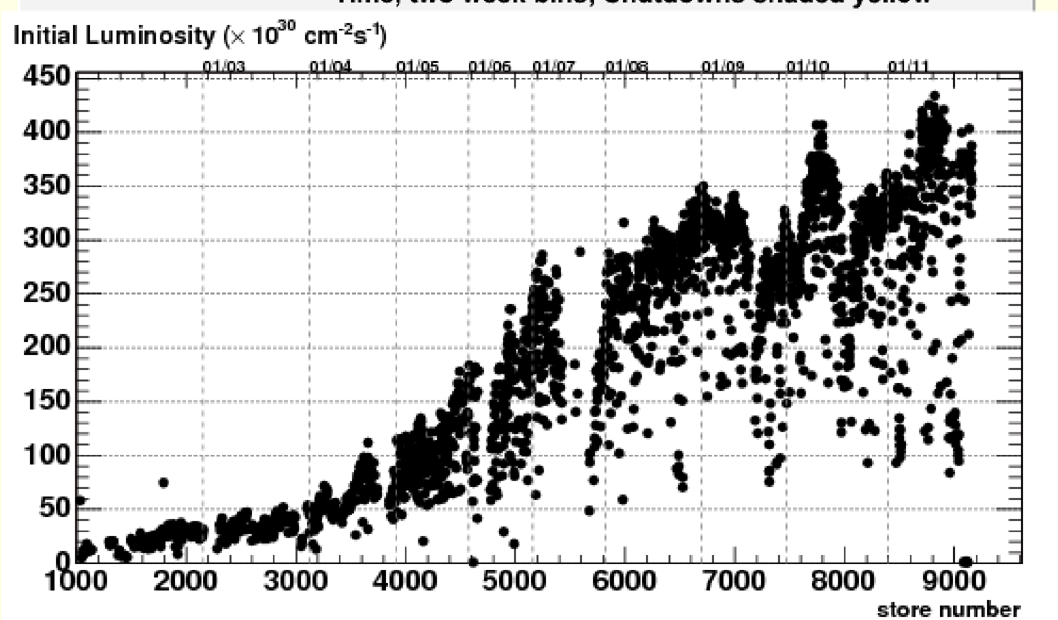
With a significant amount of electronics, HV, HV controllers in the collision hall we had to deal with many “single event upsets” believed caused by radiation

SEUs can cause faults requiring significant recovery time, thus reducing efficiency

Radiation field measured in two cases at left; radiation could be from normal running or beam losses, or perhaps uncorrelated

Plug Calorimeter Readout





SEU occurrences requiring complete reset of VME crate, CPU and electronics

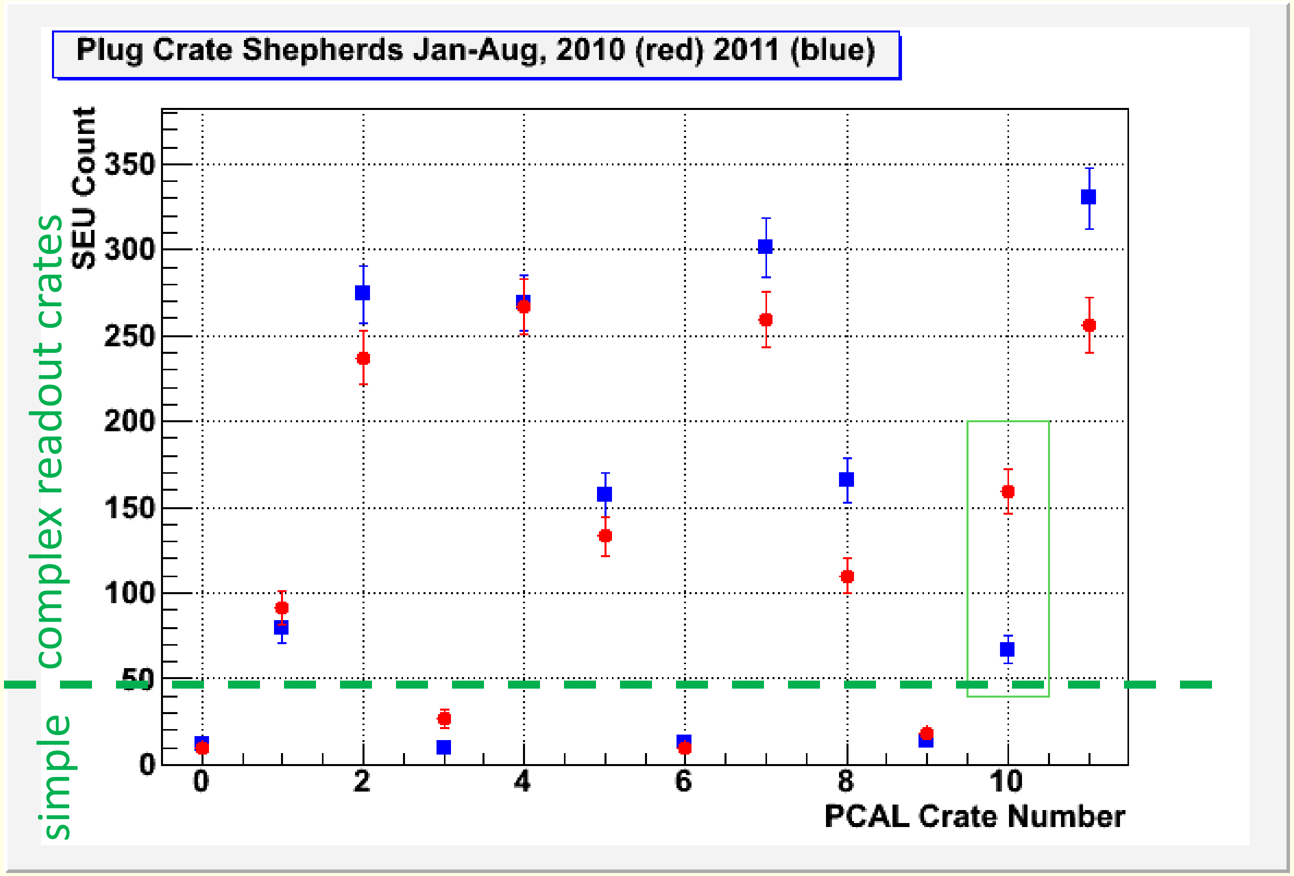
SEU rate tracks increase in Luminosity

Does not have p-pbar bias

Suspect SEUs from normal physics running

Can we reduce the SEU rate?

Non shower-max readout crates in same rack have much lower SEU rate, and use slower MVME2300 CPU compared to MVME2400. Also have very different I/O controllers. Try swapping and measuring rates!



Do see a significant drop in SEU rate (see green box)

Compare 4 non-shower-max crates near zero of y axis

ShowerMax readout significantly more complex: calibration corrections; zero suppression, etc.

SEU Conclusions

- SEUs most likely from radiation during normal physics beam running
- Can be alleviated by choice of hardware
  - But not eliminated
  - Very time consuming to test ~months; unlikely to see in test beam
  - Have potential penalties in readout rate – having to slow down clock rate, I/O changes, e.g.
- Crates without complex readout algorithms had much lower rate
  - Even with rate reduction, higher than simple readout crate
  - “Simple” readout crates also had a non-zero SEU rate
- ∴ Don’t put complex computing into a radiation environment!

Common Framework Plea

Online software that runs large, complex experiments is also large and complex

- But the functions are all very similar
- We would benefit from common solutions
- E.g. Network message passing, error logger, even state machines and monitoring
  - CDF Web Based Monitoring → CMS WBM
- Offline world has examples of commonly used packages: Root, Geant, MCs, etc.
  - See R. Brun's talk, Monday, for review of success
  - See G. Crawford's talk, Monday, for needs for future
- How would we start such an effort?
  - Need a small critical mass
  - But not a big committee (cf. R. Brun, Monday)

Conclusions

- CDF data collection concluded on September 30, 2011 after nearly 25 years of operation
- The CDF Run II DAQ and Trigger operated successfully and efficiently over 10+ years of running
- Looking forward to next generation of experiments and their DAQ and Triggers, hardware and software
  - ATCA, 10^n gigE, GPUs
  - New display/GUI platforms?
- Let's just hope there will be more HEP experiments