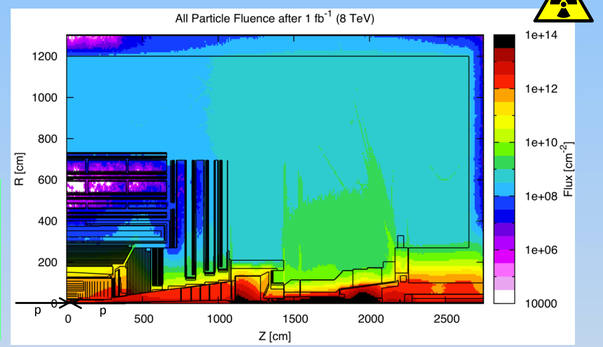
Soft Error Recovery during Operation of the Compact Muon Solenoid Experiment

Greg Rakness
University of California, Los Angeles

Radiation in the CMS experimental cavern

A soft error can be caused by ions or electromagnetic radiation striking a sensitive node in an electronic device

This plot shows a FLUKA simulation of the total particle flux as a result of 1/fb proton collisions. It is shown in a ¼-cutout of the CMS experimental hall.



Electronics in the experiment are exposed to radiation from proton collisions

Soft errors: what are they?

From wikipedia…

Soft error = data or signal in an electronic device which is wrong
A soft error can be fixed by resetting the device

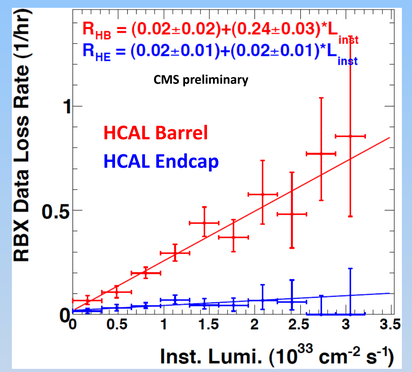
From operational experience…

During proton collisions, CMS experiences errors which…

- … interrupt the flow of the CMS data collection machine
- … are not correlated with a change in running conditions
- … in some cases, are correlated with luminosity
- … in some cases, are in agreement with expectations
- … in some cases, are expected
- … in all cases, are fixed by resetting the device

CMS observes soft errors that require fixing in the midst of collecting data

Soft errors increase with luminosity



This plot shows the rate of single event transients in optocouplers that cause soft errors in data from Readout Boxes (RBX) in the CMS Hadron Calorimeter.

The linear dependence of rate with instantaneous luminosity implies that soft errors are caused by radiation which originates from the proton collisions

Soft errors are unavoidable in particle physics experiments

Types of recovery

Two categories of soft error recovery

1. Hardware: electrical/optical signals sent from firmware
2. Software: commands sent from a computer

Two timescales of soft error recovery

1. Periodic: fix before the soft errors reach an intolerable level
2. On-demand: fix because the number or type of soft error is intolerable

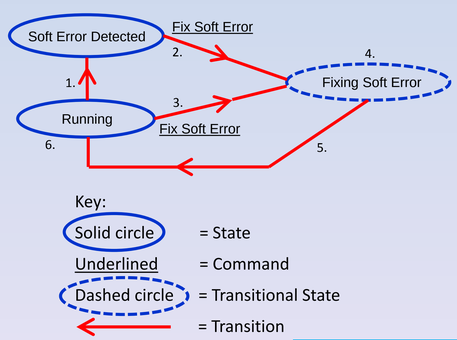
This implies…

- Fastest: hardware sends signal (periodic)
- Fast: hardware detects signal, hardware responds
- Slow: software detects signal, hardware responds
- Slowest: software detects signal, software responds

The recovery needs are unique to each subsystem

Soft error recovery: a generalized state machine

Pictorial description



Text description

1. Subsystem detects a soft error. Subsystem transitions from "Running" to "Soft Error Detected"
   - There is a problem with data integrity for which the recovery procedure is known
   - This state is a request to pause the trigger and data acquisition of the full system
2. Send command "Fix Soft Error" to subsystem in "Soft Error Detected"
   - Explicit confirmation that neither trigger nor data acquisition are running
3. In parallel, send "Fix Soft Error" to other subsystems
   - "Fix Soft Error" can be sent to those who want it… even if another subsystem has requested it
   - Can be used by subsystems to preventatively fix problems before *they* reach the level requiring "Soft Error Detected"
4. Subsystems do what they need to do while "Fixing Soft Error"
   - This mechanism is intended to recover from specific problems with minimal interruption to the system.
   - This does not mean "completely reconfigure" but may effectively mean "partially reconfigure"
5. Soft error fixed. Subsystem transition to "Running"
   - Subsystems must be able to satisfy central requirements to allow data to flow again
6. All subsystems back in "Running". We are ready to go…
   - Send signals to align all event counters in the full system
   - Resume trigger and data acquisition

System requirements

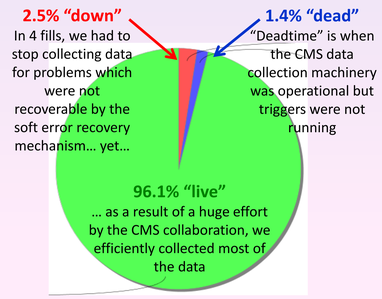
- Trigger, data acquisition (DAQ), and subsystem controlled from a centralized source
- Subsystem can detect soft errors themselves
- Subsystem can fix soft error themselves
- Subsystem can return to Running state after fixing soft error
- Trigger and DAQ can pause and allow subsystem to perform any necessary action
- Full system can continue Running even after reconfiguration of a single subsystem

These states and transitions can be inserted into any centrally controlled State Machine to recover from soft errors

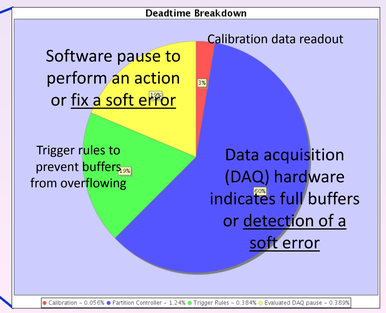
Total time spent recovering from soft errors

From 3-10 Sept 2012, the LHC delivered 1.02/fb over 10 fills lasting a total of 66h 45min.

How much of that data was not collected by CMS?



Breakdown the "deadtime"



~ 50% of the software pauses are soft error recovery

~ 20% of DAQ deadtime is soft error detection/recovery

Time spent recovering from soft errors per fb^{-1}

Detection technique	Response technique	Average number of incidents	Time per incident (sec)	Average time dead (sec)
Periodic	Firmware	130	0.100	13
Firmware	Firmware	3500	0.050	175
Software	Firmware	150	1.5	225
Software	Software	15	10	150
All	All	~3800		~560

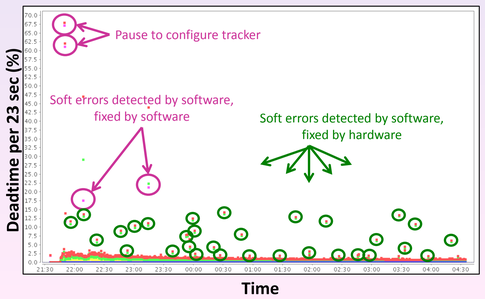
The total time spent recovering from soft errors is 0.2-0.3%
With a modest amount of work, it is possible to reduce this time by about a factor of ~2

Deadtime during a fill

Any data not collected for a reason not understood?

This plot shows the CMS deadtime per 23 second interval as a function of time during an LHC fill.

Red = Total deadtime
Green = Hardware indicates full buffer or soft error detected
Pink = Software pause to perform action or soft error recovery
Yellow = Trigger rules
Blue = Calibration data readout



Soft error recovery is responsible for ~all spikes in deadtime