SVT DAQ

Per Hansson Adrian

HPS Collaboration Meeting

10/27/2015

U.S. DEPARTMENT OF ENERGY
Office of Science

SLAC NATIONAL ACCELERATOR LABORATORY

Overview

SLAC

Trigger rate improvements

Optimized data format

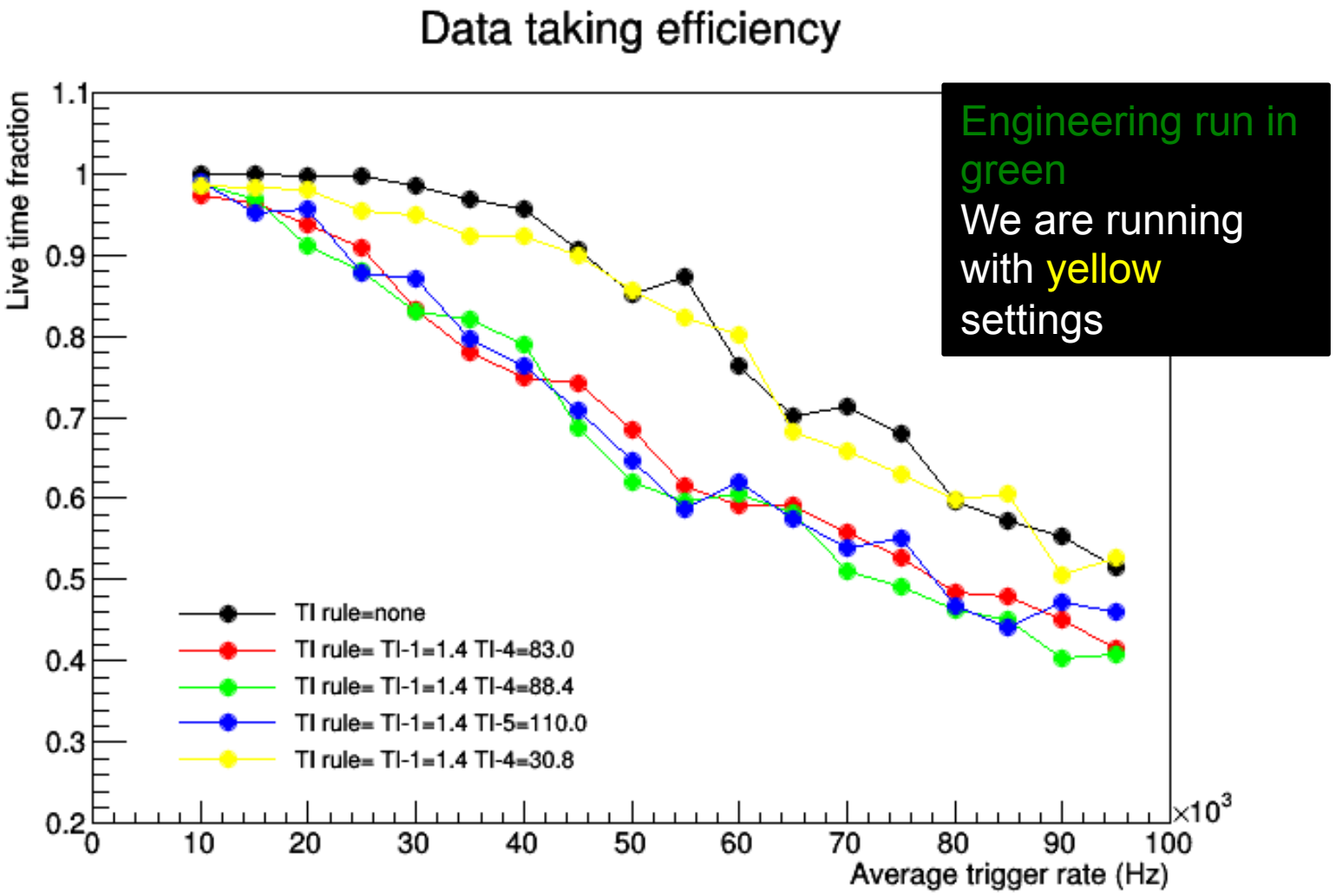
Shorter APV25 shaping time

Single event upset monitor

Data integrity

Plans

Deadtime Improvements: Simulation



NOTE: the average rate here is input or "ungated" rate

Deadtime Improvements: DAQ setup

SLAC

Trigger using special ecal configuration

- HV OFF
- Two crystals with low thresholds (all other super large)
- Singles-0 and singles-1 active with different thresholds (to allow “fine” tuning of rates)
- Event size with ECal only is ~ 1kB per event

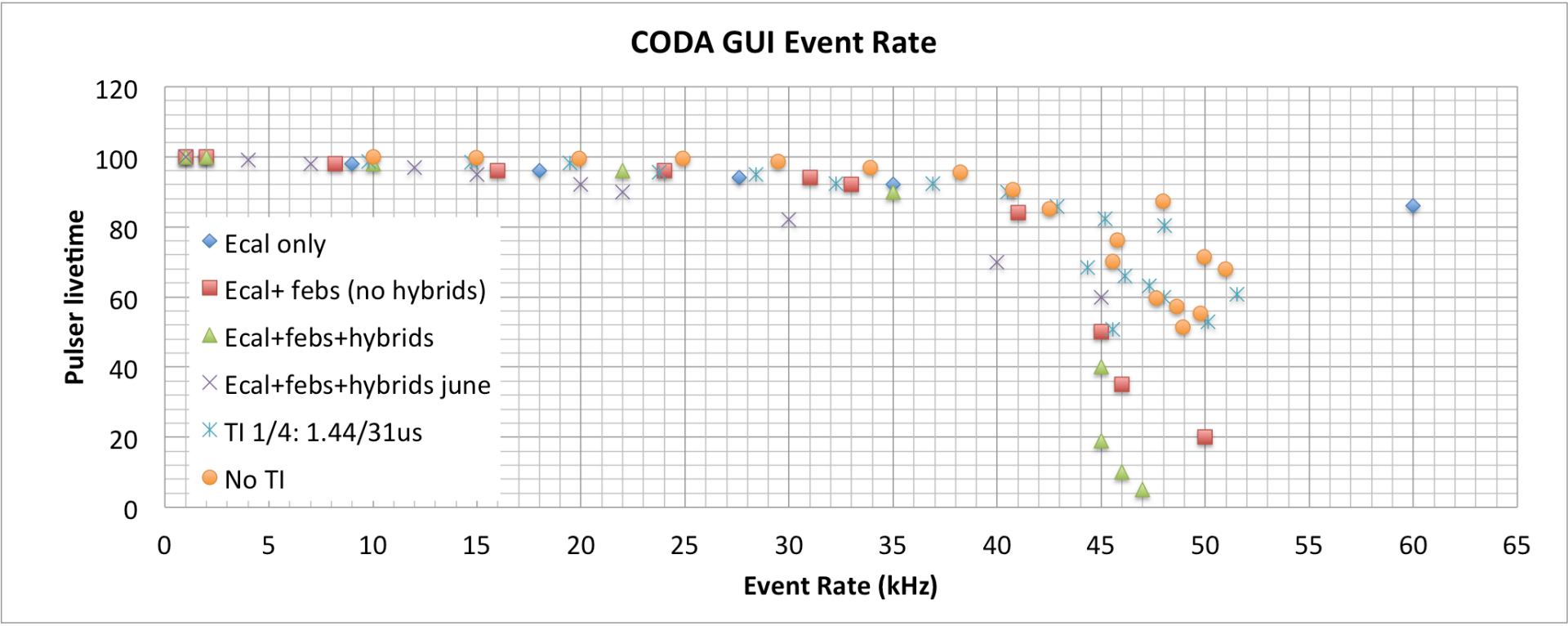
SVT is configured normally

- Chiller at +5C: hybrids at ~8C while running)
- Bias at 180V
- All FEBs and hybrids included (if not otherwise specified)
- New thresholds derived and applied at this temperature
- Event size with SVT was ~4kB per event

SVT asserts busy based on internal APV25 pipeline status

Good news is that we never crashed with the SVT in (run for hours with lots of high rates) => Busy is working.

Deadtime Improvements: GUI rates



"No TI" and "TI 1/4" are simulated rates, stat. uncertainty obvious at high rates
Live time seems turns down at the same place (except June tests)

Deadtime Improvements

SLAC

Big improvement

- DAQ live time with this setup is about 92% at 35kHz. Drops to about 45% at 45kHz
- Up to 35kHz within few percent from simulation. At 45kHz we are around 40% lower than simulation. Not clear why.

Issues

- Hard to get a good measurement at 40-50kHz because rates as measured fluctuates wildly
- One idea is that rate measurements are not integrating long enough (we are close to the limit)
- We also had an issue correlating the CODA GUI and EPICS variable rates (we sum all gated rates)

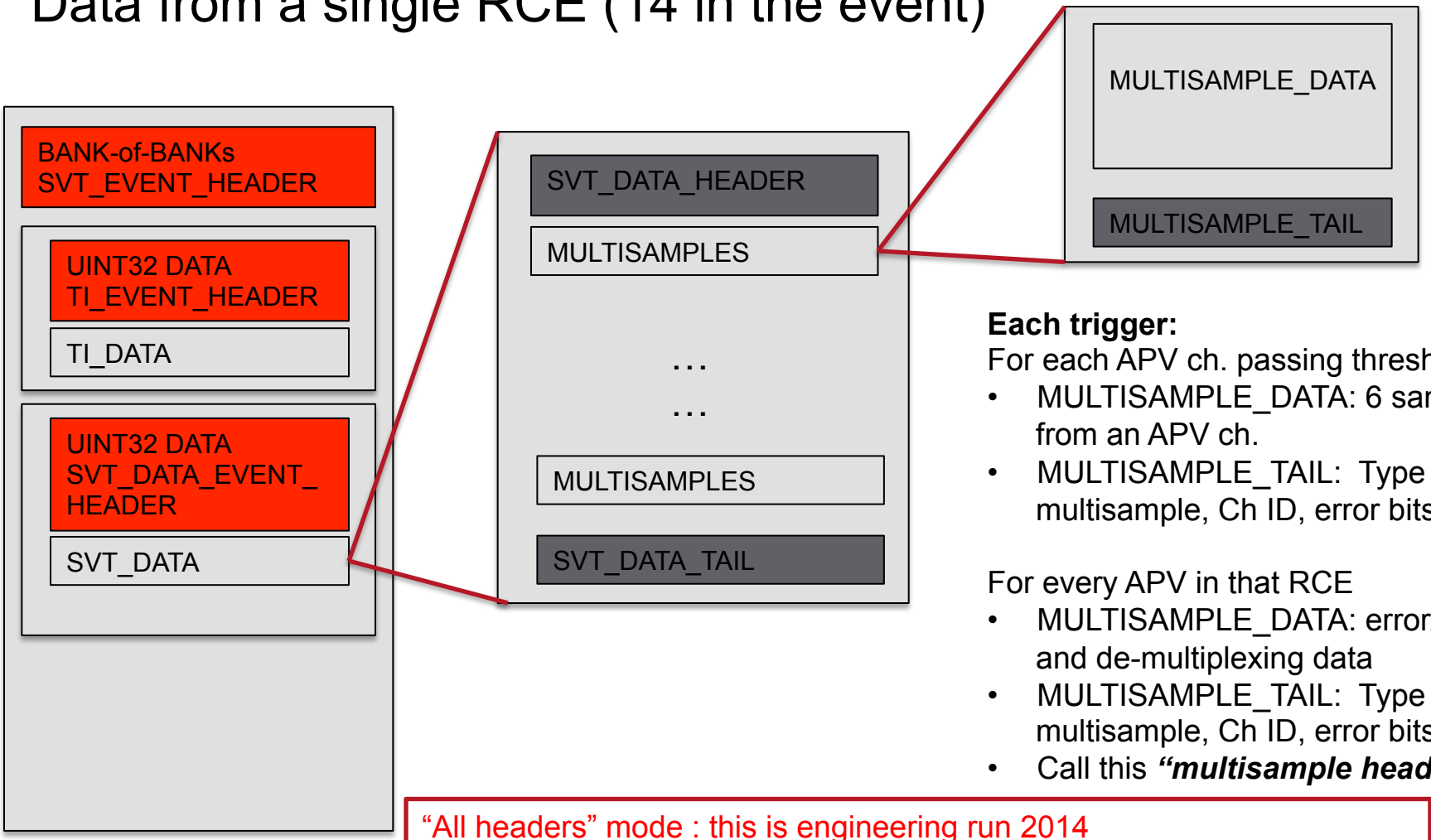
Next steps

- Hold off more tests until DAQ has been upgraded (new OS and network)
- Figure out any last (significant) difference w.r.t. simulation

SVT Data Format

SLAC

Data from a single RCE (14 in the event)



Each trigger:
For each APV ch. passing thresholds:

- MULTISAMPLE_DATA: 6 samples from an APV ch.
- MULTISAMPLE_TAIL: Type of multisample, Ch ID, error bits

For every APV in that RCE

- MULTISAMPLE_DATA: error/sync and de-multiplexing data
- MULTISAMPLE_TAIL: Type of multisample, Ch ID, error bits
- Call this ***"multisample header"***

"All headers" mode : this is engineering run 2014
"First header" mode removes all but one multisample headers per RCE
"No header" mode removes all but one multisample headers

SVT Data Format

```
<!-- header mode "All headers" -->

<bank content="bank" data_type="0xe" tag="51" padding="0" num="64" length="72" ndata="70">
    <uint32 data_type="0x1" tag="57610" padding="0" num="51" length="6" ndata="4">
            0x20010003          0x40      0xcde82c9           0x1
    </uint32>
    <uint32 data_type="0x1" tag="3" padding="0" num="51" length="64" ndata="62">
          0x100003f     0x96f594f7     0x9aff98fd     0x9ef99cfb     0x42000800
          0x36f534f7     0x3aff38fd     0x3ef93cfb     0x48800800     0x76f574f7
          0x7aff78fd     0x7ef97cfb     0x4d800800     0x56f554f7     0x5aff58fd
          0x5ef95cfb     0x49000800     0x96f594f7     0x9aff98fd     0x9ef99cfb
          0x4e000800     0x16f514f7     0x1aff18fd     0x1ef91cfb     0x40000800
          0x16f514f7     0x1aff18fd     0x1ef91cfb     0x4c000800     0x36f534f7
          0x3aff38fd     0x3ef93cfb     0x40800800     0x76f574f7     0x7aff78fd
          0x7ef97cfb     0x49800800     0x56f554f7     0x5aff58fd     0x5ef95cfb
          0x41000800     0x96f594f7     0x9aff98fd     0x9ef99cfb     0x4a000800
          0x36f534f7     0x3aff38fd     0x3ef93cfb     0x4c800800     0x16f514f7
          0x1aff18fd     0x1ef91cfb     0x48000800     0x56f554f7     0x5aff58fd
          0x5ef95cfb     0x4d000800     0x76f574f7     0x7aff78fd     0x7ef97cfb
          0x41800800           0xf
    </uint32>
  </bank>
  <bank content="bank" data_type="0xe" tag="52" padding="0" num="64" length="92" ndata="90">
    <uint32 data_type="0x1" tag="57610" padding="0" num="52" length="6" ndata="4">
            0x20010003          0x40      0xcde82c9           0x1
    </uint32>
    <uint32 data_type="0x1" tag="3" padding="0" num="52" length="84" ndata="82">
          0x100003f     0x56f554f7     0x5aff58fd     0x5ef95cfb     0x45000801
          0x36f534f7     0x3aff38fd     0x3ef93cfb     0x48800701     0x76f574f7
          0x7aff78fd     0x7ef97cfb     0x4d800701     0x76f574f7     0x7aff78fd
          0x7ef97cfb     0x4d800301     0x56f554f7     0x5aff58fd     0x5ef95cfb
          0x49000701     0x96f594f7     0x9aff98fd     0x9ef99cfb     0x4e000701
          0x96f594f7     0x9aff98fd     0x9ef99cfb     0x4e000301     0x76f574f7
          0x7aff78fd     0x7ef97cfb     0x45800801     0x16f514f7     0x1aff18fd
          0x1ef91cfb     0x4c000701     0x96f594f7     0x9aff98fd     0x9ef99cfb
          0x46000801     0x76f574f7     0x7aff78fd     0x7ef97cfb     0x49800701
          0x16f514f7     0x1aff18fd     0x1ef91cfb     0x4c000301     0x16f514f7
          0x1aff18fd     0x1ef91cfb     0x44000801     0x96f594f7     0x9aff98fd
          0x9ef99cfb     0x4a000701     0x36f534f7     0x3aff38fd     0x3ef93cfb
          0x4c800701     0x36f534f7     0x3aff38fd     0x3ef93cfb     0x4c800301
          0x16f514f7     0x1aff18fd     0x1ef91cfb     0x48000701     0x56f554f7
          0x5aff58fd     0x5ef95cfb     0x4d000701     0x56f554f7     0x5aff58fd
          0x5ef95cfb     0x4d000301     0x36f534f7     0x3aff38fd     0x3ef93cfb
          0x44800801           0x14
    </uint32>
  </bank>
```

```
<!-- header mode "First header" -->

<bank content="bank" data_type="0xe" tag="51" padding="0" num="15" length="16" ndata="14">
    <uint32 data_type="0x1" tag="57610" padding="0" num="51" length="6" ndata="4">
            0x20010003          0xf     0xdfb04f26           0
    </uint32>
    <uint32 data_type="0x1" tag="3" padding="0" num="51" length="8" ndata="6">
          0x100000e     0x8a938891     0x8e958c97     0x929f909d     0x42000800
                0xf
    </uint32>
  </bank>
  <bank content="bank" data_type="0xe" tag="52" padding="0" num="15" length="16" ndata="14">
    <uint32 data_type="0x1" tag="57610" padding="0" num="52" length="6" ndata="4">
            0x20010003          0xf     0xdfb04f26           0
    </uint32>
    <uint32 data_type="0x1" tag="3" padding="0" num="52" length="8" ndata="6">
          0x100000e     0x4a934891     0x4e954c97     0x529f909d     0x45000801
                0x14
    </uint32>
  </bank>
```

SVT overhead reduction ~ 0.075

- Engineering run had up to 80 overhead words from APV25 chips per ROC
- New format has 4

Large fraction of SVT data was overhead

- Needed for data integrity checks: implemented those in firmware now
- We have typically more APV25 chips than hits in the SVT
- With ECal in special mode, and NO beam, we see a 86% reduction in SVT bank size and ~70% reduction in HPS event size.

Trigger Delay and Faster Shaping Time

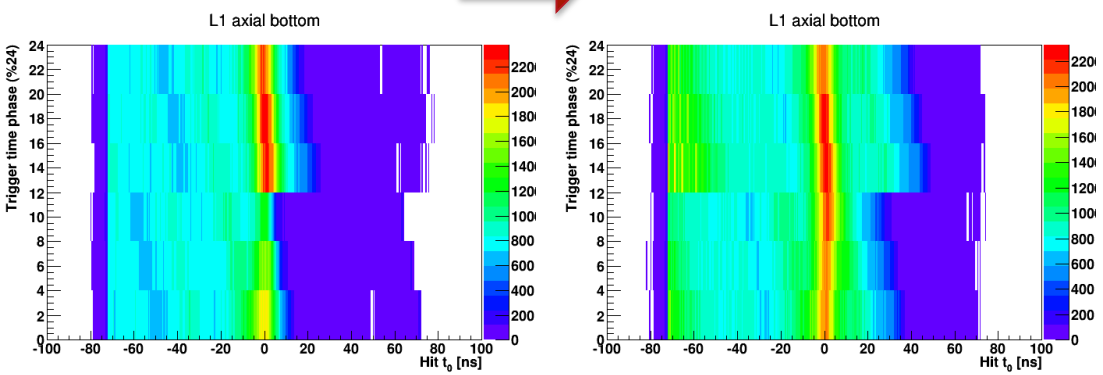
Increase adjustment of sampling point: DONE – not tested

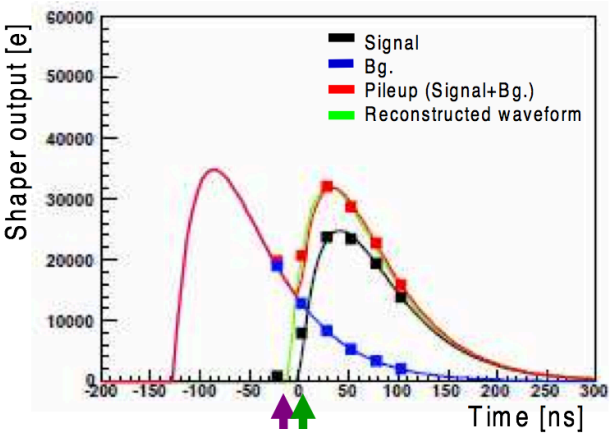
- Added firmware option to delay triggers being sent to Front End Board in steps of 8 ns.
- Allows for more optimal timing-in of SVT to improve pile-up rejection: we are picky

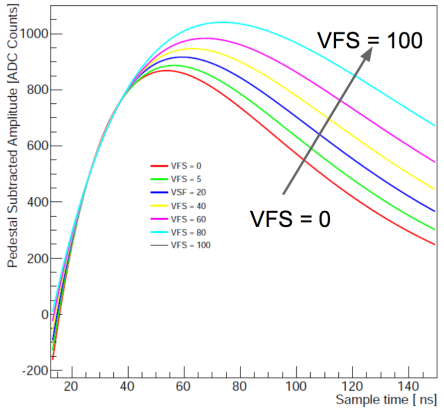
Reduce APV25 shaping time to 35ns

- Less sensitive to pile-up
- Hopefully slightly better time resolution
- Ongoing work

24ns shift







Single Event Upset Monitor

SLAC

Monitor the FPGA for errors

- Xilinx FPGA in FEB has ability to continually monitor the configuration for errors caused by Single Event Upsets (SEUs), typically caused by neutron radiation
- Not enabled during the Engineering run

Implementation

- Option 1: stream asynchronous data to mounted filesystem continuously (time stamped)
- Implement registers for monitoring and alarms (EPICS and ALH)
- DAQ will lock up in case of errors
- Ongoing work

Data Integrity

SVT DAQ Errors

Latency setting

Burst mode noise

SVT Data Integrity Analysis

SLAC

Analysis is ongoing

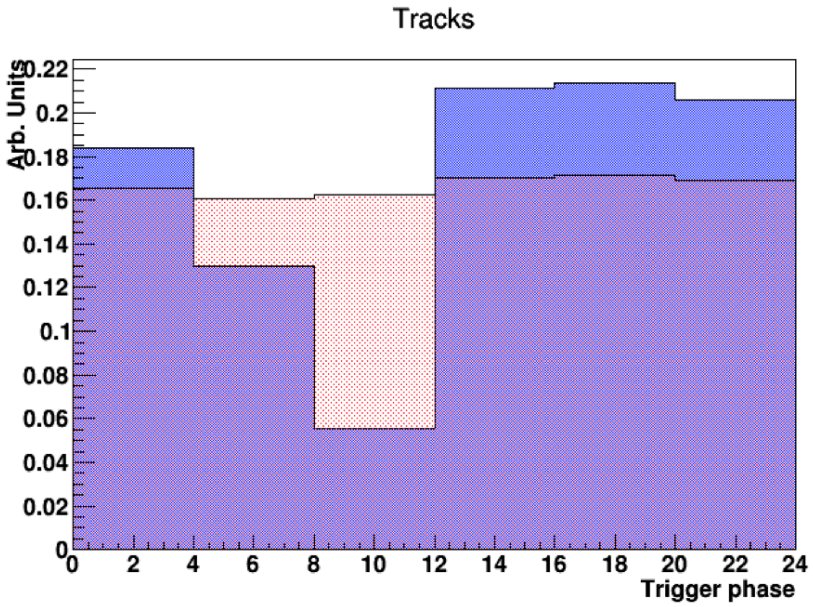
- Initially looked at subset of errors in 10% of the data
  - Saw very few (~500) data link error issues
  - More importantly saw ~2% of the events affected by "SyncError"; this error latches on (and was responsible for missing hybrids that some shifters found)
- Work is ongoing to look at all runs and all errors; will give an update soon

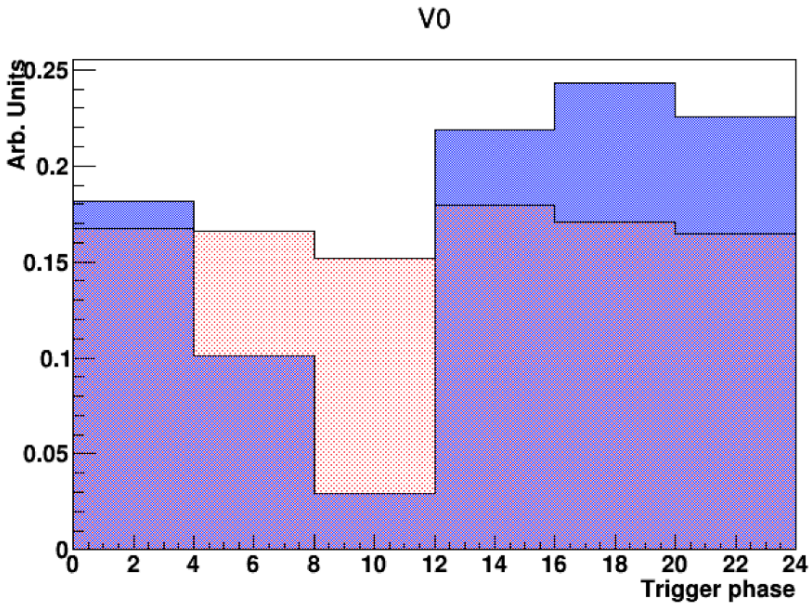
Physics analysis

- There is now (~pass3+) an event flag (Icio &DST) that tells you if there was any issues related to the SVT DAQ

Efficiency loss due to latency setting

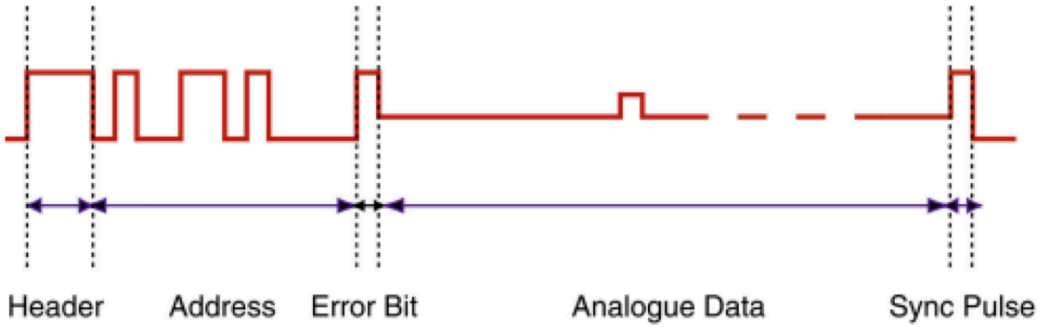
- 1.7G 0.5 mm events before fix, 2.3G events after
- Lost 25% of vertex candidates in affected runs: 10% loss overall (would want to simulate this behavior in MC)
- Can just drop events in the 2 (of 6) affected trigger phases: lose 33% of events in affected runs, 15% overall

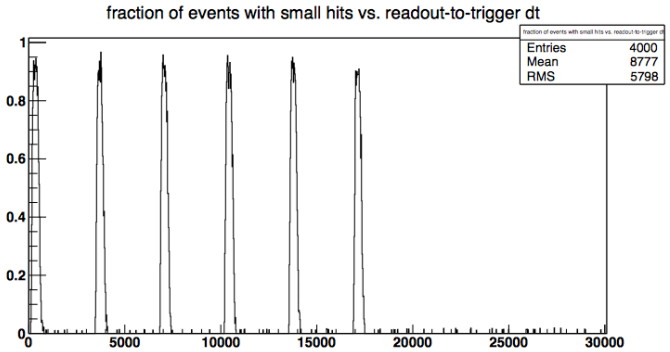




Burst-mode noise

- APV25 readout chip puts analog and digital signals on the same output line: 128 clocks of analog data, 12 clocks of digital data
- Preamp crosstalk from large currents/voltage swings: events coinciding with digital output will have extra noise
- “Burst mode” (allowing a trigger while the APV25 is still sending data) increases noise in the SVT (only in a subset of events)
  - Noisy events can be identified by looking for noise hits in the SVT (normally 0-2; noisy events have up to 100), and are correlated with time between triggers
  - From TI timestamps, we can identify all events where hits overlapped with digital output
  - Noisy events are 4-6% of data: just discard these





Operations and Plans

SLAC

Currently operating with SVT at -20C

Plan is to continue with this for a few more weeks

- Adjust APV25 shaping time
- Tune ADC sampling points across the system
- Investigate hybrid with handful of APV25 channels showing abnormal behavior

Misc.

- Need to replace broken RTD
- Need to add new flow meter (at JLab now)
- We plan not to test the spare flange due to unnecessary risks (unless analysis show link errors more prevalent)
- GUI updates

Summary

Large improvement in rates using TI busy

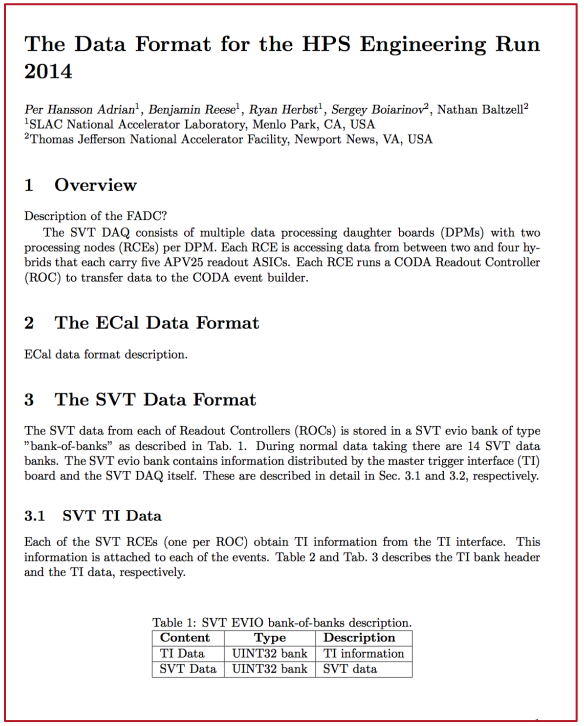
Expect reduction of SVT event size with about 80%

Ongoing work this fall

- Tune SVT operating points
- Implement SEU software and test
- More rate tests with updated DAQ
- Spares…

Documentation

- Data format note
- DAQ notes/papers



The Data Format for the HPS Engineering Run 2014

Per Hansson Adrian[1], *Benjamin Reese*[1], *Ryan Herbst*[1], *Sergey Boiarinov*[2], Nathan Baltzell[2]
[1]SLAC National Accelerator Laboratory, Menlo Park, CA, USA
[2]Thomas Jefferson National Accelerator Facility, Newport News, VA, USA

1 Overview

Description of the FADC?

The SVT DAQ consists of multiple data processing daughter boards (DPMs) with two processing nodes (RCEs) per DPM. Each RCE is accessing data from between two and four hybrids that each carry five APV25 readout ASICs. Each RCE runs a CODA Readout Controller (ROC) to transfer data to the CODA event builder.

2 The ECal Data Format

ECal data format description.

3 The SVT Data Format

The SVT data from each of Readout Controllers (ROCs) is stored in a SVT evio bank of type "bank-of-banks" as described in Tab. 1. During normal data taking there are 14 SVT data banks. The SVT evio bank contains information distributed by the master trigger interface (TI) board and the SVT DAQ itself. These are described in detail in Sec. 3.1 and 3.2, respectively.

3.1 SVT TI Data

Each of the SVT RCEs (one per ROC) obtain TI information from the TI interface. This information is attached to each of the events. Table 2 and Tab. 3 describes the TI bank header and the TI data, respectively.

Table 1: SVT EVIO bank-of-banks description.

Content	Type	Description
TI Data	UINT32 bank	TI information
SVT Data	UINT32 bank	SVT data