GLAST Large Area Telescope

LAT Deadtime

Warren Focke
SLAC
I&T Science Verification Analysis and Calibration Engineering Physicist
focke@slac.stanford.edu
650-926-4713

Outline

- What are we measuring and why?
- How?
- Full LAT deadtime measurements
- Interesting stuff
- Wrapup

It's Not Measured in %

- What?
  - Want to measure how long detector is unresponsive after an event
- Isn't the livetime counter good enough?
  - It's fine if you want to make an energy spectrum or image. But timing properties are affected by deadtime that is correlated with the signal.

How?

- Two ways to measure deadtime:
  - realtime – livetime
  - minimum event separation
    - delta EvtTicks
    - GemDeltaEventTime
- Most of this will not be possible offline in flight
  - onboard filter will discard many events
    - true previous/next events will usually not be available
    - so we won't get deadtime per event unless it's done onboard

Variables Used

- All measured in LAT ticks (50 ns)
- All in SVAC tuple
- GemDeltaEventTime (GDET)
  - direct from GEM
  - time since last event
    - only if triggered & read out
  - saturates
  - 16 bits = 3.3ms
- GemLiveTime (LIVE)
  - direct from GEM
  - only increments when LAT not busy
  - running counter (rolls over)
  - 25 bits = 1.7s

More Variables

- **EvtTicks**
  - Calculated offline
    - from GemOnePpsSeconds, GemOnePpsTime and GemTriggerTime
    - use EvtSecond, EvtNanoSecond to catch rollovers
  - Elapsed ticks since arbitrary point (<128s) before run start
  - Will need a new algorithm when we get GPS
    - 1pps signals are currently generated from GEM clock and are always exactly 20,000,000 ticks apart
    - This will not be true with GPS, as the GEM clock isn't that good
      - But it doesn't have to be
    - Absolute times will involve interpolating from 1pps signals to determine actual clock rate

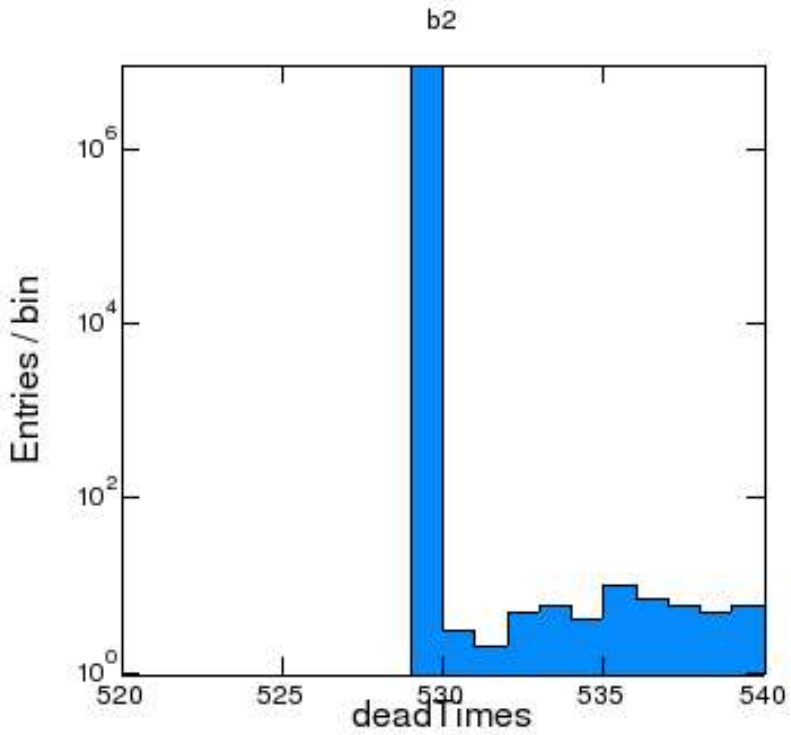
Calculating Deadtime

- LIVE is running livetime counter from GEM
  - GemLiveTime in SVAC tuple
- $DLT_i = LIVE_i - LIVE_{i-1}$
  - + 2**25 if < 0
- $DET_i = EvtTicks_i - EvtTicks_{i-1}$
- $DeadTime_i = DET_i - DLT_i$
- Make histogram
-
- Can't do this in flight

Other Methods

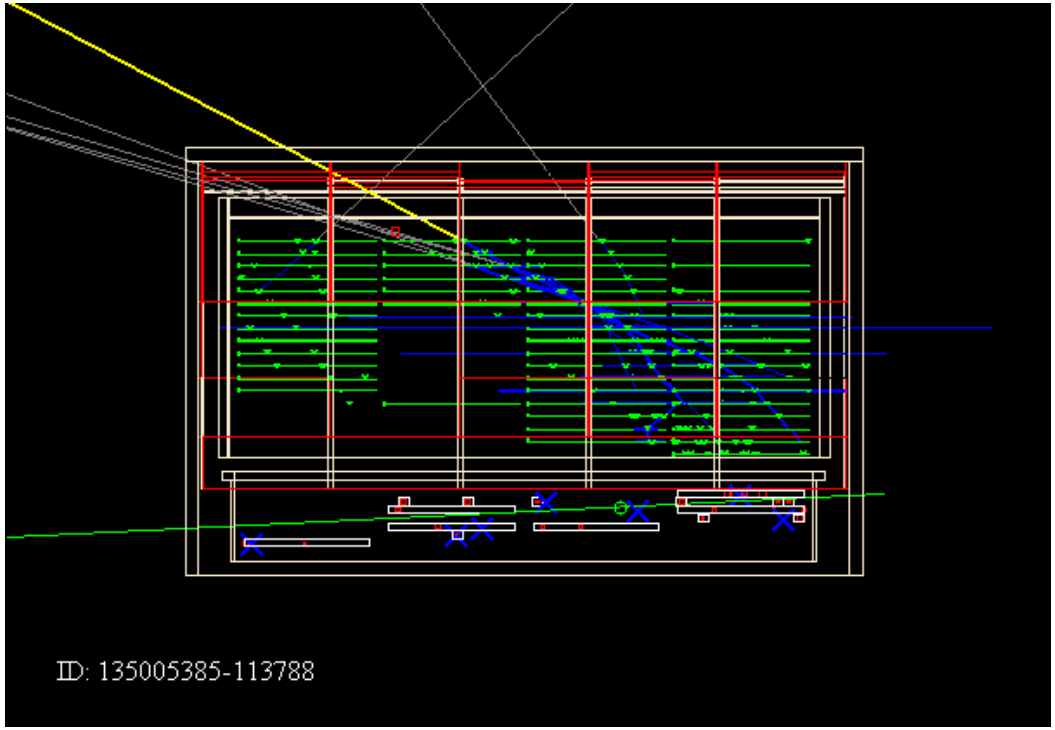
- Deadtime can also be estimated by looking at time intervals between successive events
  - Smallest value observed is upper limit
    - deadtime is actually 1-(smallest value)
- GemDeltaEventTime measures this directly
- Difference in EvtTicks for successive events gives another measure
-
- Both of these measurements agree with deadtime as calculated on previous slide for end2end full LAT runs

Full LAT B/2 Deadtime



- Minimum = 529 ticks (26.45μs)
  - This is the predicted value
- 10.8M measured times
- 238 of them != 529
- Max = 5697 ticks (285μs)

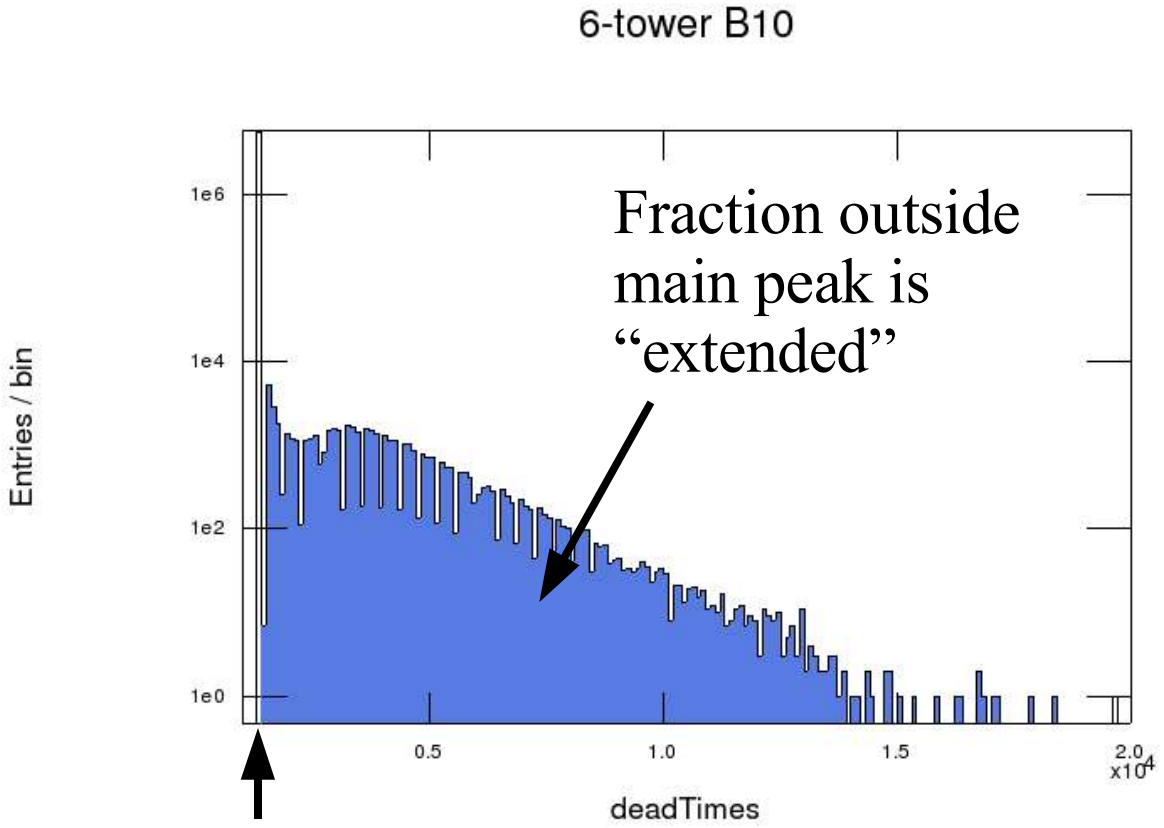
B/2 Extended Deadtime



- 238/10.8M not minimum
- Due to
  - Large events
  - Backpressure from previous large events
- Max = 5697 ticks (285μs)
  - previous event was not reconstructed
- Pictured event had 550 ticks

Full LAT Deadtime

	min Deadtime	min GDET	min delta EvtTicks	Extended fraction
B/2	529	530	530	2.20E-005
B/13	13129	13130	13130	25.00%
B/30	1309	1310	1310	1.10%



Deadtime Evolution

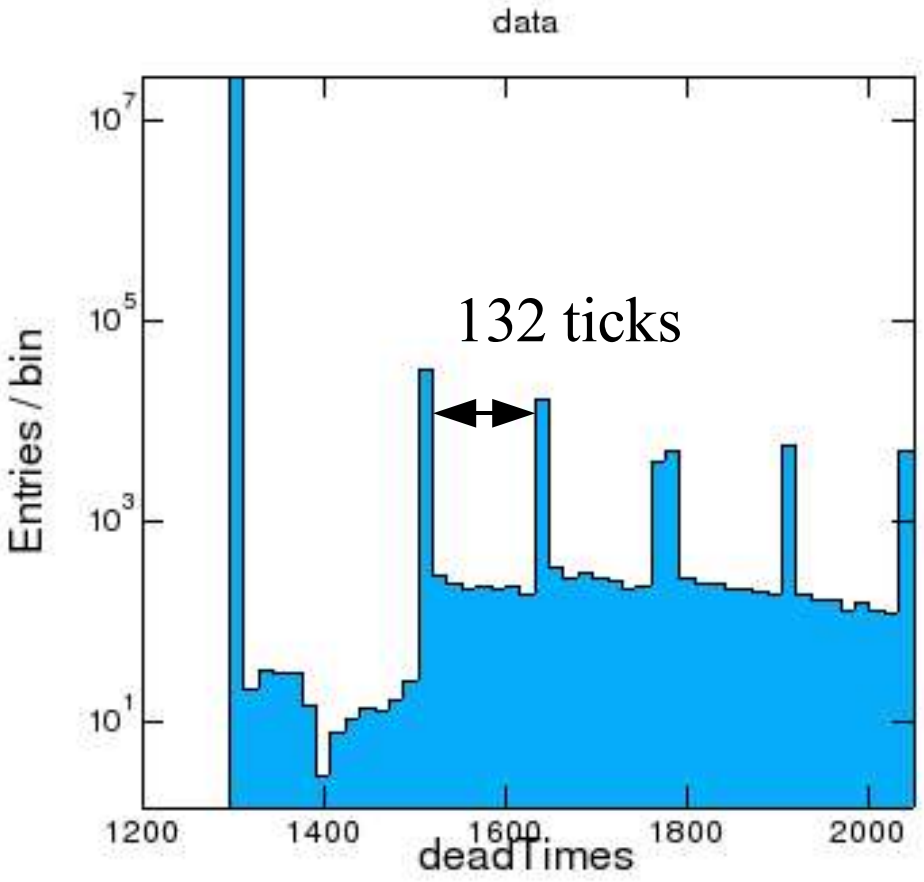
# Towers	B2	B10/B30	B13
2	529	1308	13128
4	529	9167	13128
6	529	1309	13129
8	529	1309	13129
LAT	529	1309	13129

Change from 1308/13128 to 1309/13129 is due to different CAL TACK delays in new towers. Maximum was 45 ticks for 2 and 4 towers, 46 ticks from 6 towers on.

Bad LAC threshholds caused about half the logs to be read out for every event

Deadtime remained @ 529 even for 20kHz external trigger (2 tower run)

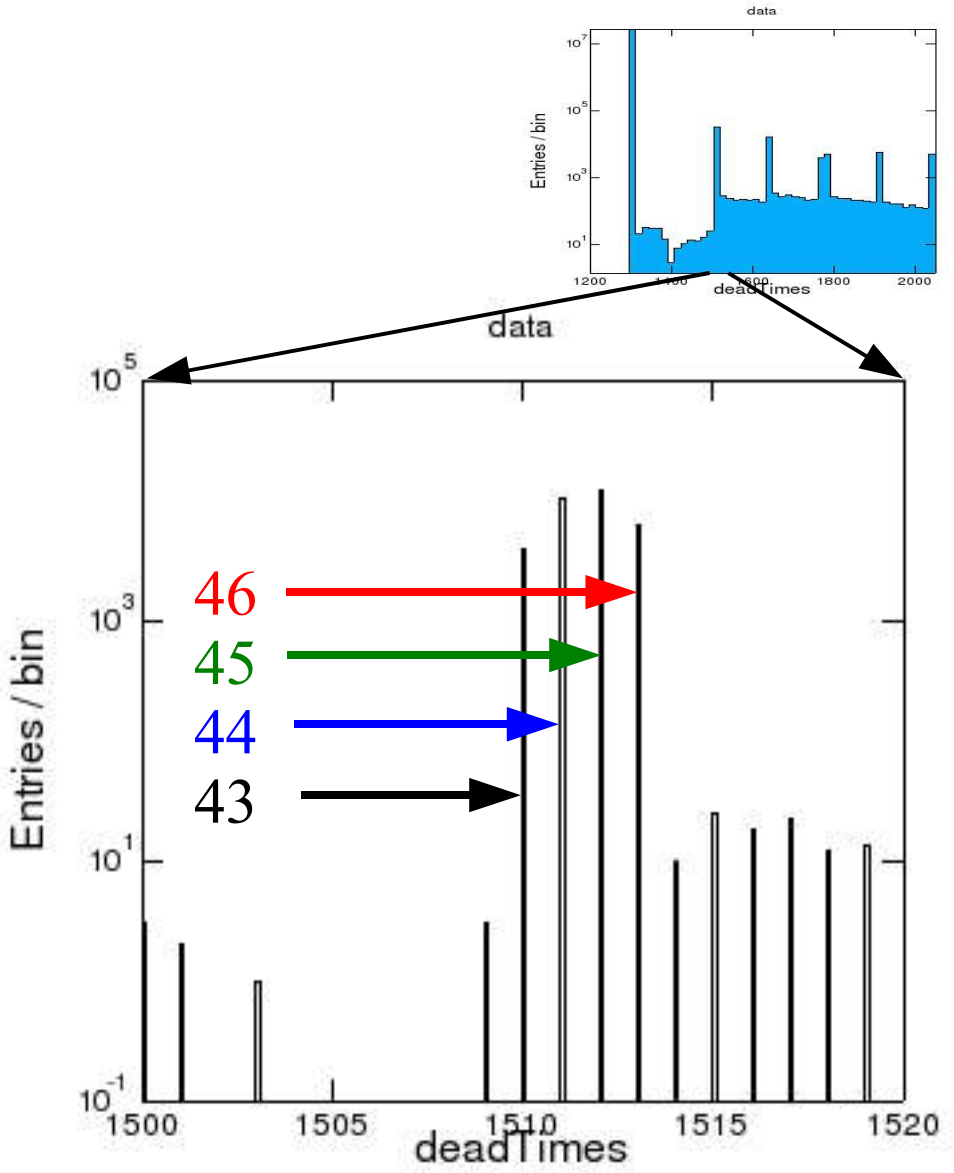
Quantized Deadtime in B10/B30



- Secondary peaks are separated by 132 ticks
  - this is the time required to read out 4 CAL logs
- These are 4-range runs, so CAL data is always quantized in 4-log chunks

Peak Widths

- Secondary peaks are > 1 tick wide due to different CAL TACK delays in different towers.
- 2 towers:
  - tack delays = 44, 45
  - peaks at 1511, 1512
- 16 towers:
  - tack delays = 43-46
  - peaks at 1510-1513
- Main peak is 1 tick wide because all towers contribute – longest delay wins
  - that's why the deadtime grew (by 1 tick) when we added towers 8 & 9



Conclusions

- Deadtime is stable at the predicted value
  - even at high rate
- Requirement: < ~~100~~ 30 μs
  - we're well under
- Goal: < 20μs
  - Missed it by *that* much