Hall A Analyzer - Bug #63

Scaler event count wrong in end-of-run counter summary

05/18/2017 02:26 PM - Ole Hansen

Status:	In Progress	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	10%
Category:		**Estimated time:**	8.00 hours
Target version:	1.7	**Spent time:**	0.00 hour
Responsible:			

Description

I get this with analyzer-1.5.25:

Normal end of file /home/ole/tutorial/data/g2p_3132.dat.0 encountered
End of file
Counter summary:
314292 events read
314292 events decoded
313476 physics events
314172 scaler events
115 slow control events
5 other event types
313476 physics events analyzed
314292 events accepted

314172 is not the number of scaler events, but "all"-"slow control"-"other". There should be 696 scaler events. Looks like a bug.

History

#1 - 05/18/2017 02:32 PM - Ole Hansen

This is not a bug in THaAnalyzer, but it looks like it could be one in the decoder. Maybe Bob can help:

THaAnalyzer counts a scaler event whenever fEvData->IsScalerEvent() is true. As implemented in THaEvData.h, this function is true for event type 140 OR if member variable evscaler 1. A comment says that "a scaler event can also be a physics event". I assume that these "dual" events are the ones with evscaler 1.

evscaler is set to 1 in THaCodaDecode::vme_decode whenever the event contains data from at least one ROC that is defined as type "scaler" in the crate map and that has at least 16 bytes of data:

if (fMap->isScalerCrate(roc) && GetRocLength(roc) >= 16) evscaler = 1;

This is true even if none of the scalers in this crate actually have any data. The crate map used for the replay above contains three such crates:

==== Crate 10 type scaler "evright"

# slot	model	clear	header	mask	nchan	ndata
1	3801	1	0xceb00000	0xffff0000	32	32
2	3801	1	0xceb10000	0xffff0000	32	32
3	3800	1	0xceb20000	0xffff0000	32	32
4	1151	1	0xceb30000	0xffff0000	16	16
5	1151	1	0xceb40000	0xffff0000	16	16
6	1151	1	0xceb50000	0xffff0000	16	16
7	560	1	0xceb60000	0xffff0000	16	16
8	560	1	0xceb70000	0xffff0000	16	16

==== Crate 11 type scaler "evleft"

# slot	model	clear	header	mask	nchan	ndata
1	3801	1	0xabc10000	0xffff0000	32	32
2	3801	1	0xabc20000	0xffff0000	32	32
3	3800	1	0xabc30000	0xffff0000	32	32
4	3800	1	0xabc40000	0xffff0000	32	32
5	3800	1	0xabc50000	0xffff0000	32	32

6	3800	1	0xabc60000	0xffff0000	32	32

==== Crate 12 type scaler "ev3arm"

# slot	model	clear	header	mask	nchan	ndata
1	3801	1	0xddd10000	0xffff0000	32	32
2	3801	1	0xddd30000	0xffff0000	32	32
7	775	1	0xda00bdc0	0xffffffff	32	2048
9	792	1	0xda00adc0	0xffffffff	32	2048

Models 1151, 560 and 3801 are handled in vme_decode, but 3800 is not implemented. (What is it?) 775 and 792 are a TDC and a QDC, respectively, presumably not present in the g2p setup. (Are they?) In any case, it looks like scaler crates can have both scaler and non-scaler modules. (Can they?) If so, then I wonder if evscaler should only be set if an actual scaler module reports data in the event, otherwise THaAnalyzer::ScalerAnalysis will be called for nothing, and too many events with scaler data are reported.

Bob: How do you explain the scaler event count in the above analysis? Does every physics event really have data from scalers in the scaler crates?

#2 - 05/18/2017 02:34 PM - Ole Hansen

From Bob Michaels 30-Jan-2014:

The simplest answer is that for some experiments, a fraction of our scaler channels were read for every event. The scaler events were evtype 140, occur every 4 sec approx and were typically 200 channels of scalers integrated for 4 sec or so. The scalers in events were evtype 1 - 7 normally, typically at 1 kHz and with perhaps 16 chan. I don't think there were "bugs" but this will get reorganized this year as a byproduct I'd what I'm doing. There are parts of ThaEvdata related to scalers which actually don't get used, and the work done by THaScaler instead. In the future I plan to have a THaScalerEventHandler for the evtype 140, hall C can have their own EvHandler, and if there are "scalers in Physics triggers" they will be handled like another VME module.

#3 - 05/18/2017 02:34 PM - Ole Hansen

From Ole Hansen 30-Jan-2014:

I guess at the minimum we need to make the functions THaEvData::IsPhysicsTrigger(), THaEvData::IsScalerEvent() etc. more adaptable since their behavior is currently hardcoded. Perhaps they should report bits from an internal bitfield or similar, with each bit corresponding to a certain event class. The event type handlers then can set those bits as appropriate.

In your new decoder scheme, do you still want scalers to be analyzed within THaAnalyzer, using something similar to THaScaler? I suggest we come up with a universal design for handling scalers, regardless of whether they are read in "evtype 140" or physics events. It sounds like you want the "evtype 140" scalers to be handled by an event type handler and scalers in physics events by something (some "scaler apparatus"?) in THaAnalyzer? Is that the plan? Why not let the scaler event handler take care of them as well?

BTW, what type of module is model "3800"?

#4 - 05/18/2017 02:34 PM - Ole Hansen

From Bob Michaels 30-Jan-2014:

The 3800 is a type of scaler (we have 4 types in Hall A,
each with different properties).

I'll try to come up with some working code for scalers soon
and then we can discuss it.

I think the "scaler events" should be handled by a "event type handler", while a "scaler module" is a kind of module. The same module class used by scaler events can be used in physics events if there are scalers in the event stream.

#5 - 05/18/2017 02:35 PM - Ole Hansen

From Ed Brash 30-Jan-2014:

I think the "scaler events" should be handled by a "event type handler", while a "scaler module" is a kind of module. The same module class used by scaler events can be used in physics events if there are scalers in the event stream.
That logic makes sense to me.

#6 - 05/18/2017 02:35 PM - Ole Hansen

From Ole Hansen 30-Jan-2014:

Makes sense to me, too. My point is: there shouldn't be two places in the analyzer where scaler modules are processed. In fact, maybe the distinction of "scaler events" could even be dropped. Maybe scaler events are just like physics events, except that ALL modules read in the scaler event are scaler modules. Physics events may contain a mixture of physics modules (ADCs, TDCs etc.) and scaler modules. Data from scaler modules are always processed by the same code in the same place (presumably somewhere in the decoder), regardless of the event type. Would that work?

Also, how do we make scaler data available to the user? If they should go into a separate tree in the output ROOT file, maybe processing scaler modules in THaAnalyzer is the way to go after all?

#7 - 05/19/2017 05:01 PM - Ole Hansen

- Start date deleted (12/17/2013)