

dunetpc - Bug #23807

Out of range clock errors in dataprep

12/26/2019 01:02 PM - David Adams

Status: Closed	Start date: 12/26/2019
Priority: Normal	Due date:
Assignee:	% Done: 0%
Category:	Estimated time: 0.00 hour
Target version:	
Description	
When I run dataprep jobs, I sometimes see the reported channel-timing clock differences appear to be outside integer range, e.g.	
<pre>DataPrepByApaModule::produce: WARNING: Channel clocks for APA 3 are not consistent. DataPrepByApaModule::produce: WARNING: Clock ticks count DataPrepByApaModule::produce: WARNING: -78116685949206656-3.12467e+15 128 DataPrepByApaModule::produce: WARNING: -12513 -500.52 2432</pre>	
This type of report appears whenever the ADC channel clocks for an APA do not all have the same value. Here, 128 channels (1 FEMB) have a calculated difference that is ridiculously large. Checking for one event, I found the ADC channel clock was zero. Presumably there and here, the problem is FEMB 302 although I do see this problem in other APAs.	
There are (at least) two issues to resolve: if we should expect clocks set to zero and to improve the reporting in DataPrepByApaModule.	
Tom, can you comment on the first of these?	

History

#1 - 12/26/2019 01:37 PM - David Adams

- File valgrind4517-87.out added

I have modified DataPrepByApaModule so warnings like the above now appear like this:

```
DataPrepByApaModule::produce: WARNING: Channel clocks for APA 3 are not consistent.
DataPrepByApaModule::produce: WARNING:      Clock      ticks      count
DataPrepByApaModule::produce: WARNING: -999999999      -4e+06      128 Channel clock is zero.
DataPrepByApaModule::produce: WARNING:      -12505      -500.2      2432
```

A maximum abs difference of 99,999,999 timing ticks (2 sec) is reported and an extra message displayed if that threshold is exceeded.

The above is for run 4517 event 87.

I ran this event with valgrind and some errors in the decoder are detected. I attach the report.

#2 - 12/26/2019 03:22 PM - Thomas Junk

By "clock" you mean the timestamp in raw::RDTimeStamp? I am not surprised the value is not filled properly for FEMB302's data. Does this happen every event for all 128 channels in FEMB302? I added a timestamp consistency filter to ProtoDUNEFembFilter_module.cc, and the example in runProtoDUNEFembFilter.fcl requires all channels on the beam side to have the same timestamps. I tested it on a small number of events and it worked.

The zero timestamp values are good catches. I can ask the DAQ people, though JJ Russell is no longer working on project-paid work, so it may be hard to get his attention. The invalid reads in the valgrind output are also concerning. When we were struggling with corrupt data, it was often the timestamp-getting method that segfaulted before we made enough integrity checks before calling it. More investigation coming -- I'm on vacation this week.

#3 - 12/26/2019 04:05 PM - David Adams

Yes, clock is the timestamp. I was surprised to see that it is often (mostly?) filled correctly. Other events log no warning messages implying the clocks for all channels agree and others show nonzero but discrepant values.

I have not confirmed the problem for APA 3 here is in FEMB 302. This problem (very large discrepancy) also appears in APA 4 so it is not just FEMB 302.

I have many examples of this problem and of crashes in the decoder. I was going to open a separate ticket for the latter. The crashes are all in early data.

I think someone should investigate and try hard to remove the memory errors. I don't know if this should be JJ, you, I or someone else but I would like to eliminate the memory errors. If need be, I am willing to work on this. I think I need to learn how to build proto-dune-dam-lib and link against it. I have been poking around with valgrind and gdb but it is difficult with everything optimized out. If I use the the debug version of dunetpc, will I see more symbols in proto-dune-dam-lib?

#4 - 12/26/2019 04:38 PM - Thomas Junk

I looked at the e-log for run 4517 and nearby entries. It was taken Sep. 20, 2018. I didn't find much, other than the cathode voltage was -140 kV, the cameras were on but lights were off. Giovanna requested a power cycle of cob6_rce02 at around that time and a reset, after which the shifter reported a trigger rate of 5 Hz. From the shifter: "cob6_rce02 fragment size was almost zero last run, which is why we stopped it. its APA-US-DaS which isnt configuring".

I agree it would be good to bulletproof proto-dune-dam-lib better. I went back and forth with JJ about this last year and the best I got was methods that checked the beginning and the end of the fragmnet for size and expected bytes. Crashed RCE's tend to drop data and the fragments are often short when the RCE's are not working properly.

dunepdsprce is also built with the debug qualifier. If you setup dunetpc with debug, you will get proto-dune-dam-lib (ups product named dunepdsprce) also in debug.

There is a makefile that comes with it in github -- just search github for proto-dune-dam-lib, clone it, find the makefile, and run make. larutils/buildScripts/build-dunepdsprce.sh does exactly that and also makes a UPS product out of it. If we fix bugs in it or otherwise make it more robust, we may have to give JJ a pull request so the mainline repo has the fixes. Or just fork it. I even had a little e-mail chat with Kirby about this piece of code being "brittle" and a long-term risk. I built it with c7 and e19 last week in anticipation of using those compilers, but sometime far in the future something mysterious could break with it.

#5 - 12/26/2019 04:48 PM - Thomas Junk

JJ's makefile is in proto-dune-dam-lib/dam/source/cc/make. I just cd to that directory and run it, but it has some options for using gcc or clang, or whether to use avx2 instructions or "gen" without avx2, and also a switch for turning on and off optimization (PROD=1 means optimize it). One of the instances of brittleness in the code is some historical difference in the behavior of the optimized and non-optimized versions which JJ ironed out after some investigation by me. I am not convinced that the code will continue to function as the compiler optimizations evolve, especially if it is giving invalid reads in valgrind. The data are probably just corrupt and we are better off skipping around it, which is what we try to do whenever we detect corrupt data, since we want the jobs to finish with as much good data as there is.

#6 - 12/26/2019 07:17 PM - David Adams

I did figure out how to grab and build proto-dune-dam-lib but have to figure out how to get dunetpc to build and run with my version. And when we do have changes, we need a place to put them. If JJ is not going to provide support, then we should copy into a dune repository and add the instructions to make a ups package. This will make maintenance much easier, at least for me. I suppose we already have the ups instructions in our Jenkins scripts.

I agree that we can skip events (or APAs or FEMBs) with corrupt data but we should have a way to identify these that does not require running code memory errors that may corrupt data in other parts of the detector, later in reco or in other events. I have been running single-APA jobs until they crash and then adding the event to a skip list but this is rather cumbersome. I also added event with the error reported here. This worked for all but one APA (3) in one run (5581). I am still nervous that memory errors causing subtle problems may remain.

#7 - 12/26/2019 09:29 PM - Thomas Junk

Yes, that has always been the strategy -- somehow test the data for integrity before unpacking it which can cause a segfault or more subtle errors. But even basic stuff like retrieving the number of ticks has caused crashes in the past. The build model was to use JJ's version as the reference, but JJ had different requirements than we do. He had to support some embedded platforms too. If all that is now gone, probably the best thing to do is to just copy his source files and include files into one of our repositories or make a new one. There is a little snip of code in dune_raw_data that injects itself into the process -- rce_fragment.cc, .hh (maybe miscapitalized). That's a little thing Jingbo had put in there in 2018. We could just imagine putting all of this into dunetpc, but the online monitor has to be able to run as well. It might be easier to fork JJ's github repo into our own and build the same.

#8 - 12/26/2019 09:46 PM - Thomas Junk

Oh and to use your privately built version, just put the .so's in LD_LIBRARY_PATH ahead of dunepdsprce's.

#9 - 12/27/2019 10:35 AM - David Adams

I would prefer to keep the code in a separate repository rather than add to the bloat of dunetpc.

I have added Tingjun and Mike to this report. I understand that Tingjun has identified some effort to work on dataprep performance and IMHO unpacking is the place to start.

As for including changes via the library path, the reported memory errors are in include (.hh) files. Are these only include in code compiled into the proto_dune_dam_lib package or are they also included elsewhere?

#10 - 12/27/2019 12:08 PM - Tingjun Yang

Hi David,

Thanks for adding me to the issue. I run valgrind on a regular basis and I am also concerned about those memory errors. I think copying the relevant code to a separate repository and making a ups product is a good first step. If SCD is available to help us improve the robustness of the code, that would be great.

Tingjun

#11 - 12/27/2019 01:14 PM - Thomas Junk

Regarding David's question about .hh's in proto-dune-dam-lib which also get into dunepdsprce. Indeed, they are included by code in dune-raw-data and also by the raw decoder tools and modules. Putting lots of executable code in header files is not my favorite, and so it is easy to get version shear by not re-compiling the products that depend on dunepdsprce when you make a change to it. UPS requires rebuilds which is a good thing, but it can make the debug cycle a little slow -- you'll have to rebuild dunepdsprce, dune_raw_data and dunetpc if you change a header file.

You can run the Jenkins script interactively and it'll build a UPS product you can copy into your localProducts directory. Here's a little script I use to run the Jenkins script interactively. Copy the latest build-dunepdsprce.sh out of larutils to an empty directory, and run this. It will make a tarball suitable for uploading to scisoft, and unpacking to a UPS products directory. build-dunepdsprce.sh clones JJ's github repo -- you'll have to hack it to use your own version of the source code. The reason to go through all of this is so when you rebuild dune_raw_data an dunetpc, the .hh files are in the right include path.

```
#!/bin/sh
```

```
touch workspace
rm -rf workspace
mkdir workspace
```

```
export QUAL=e17
export COMPILERQUAL_LIST="e17 c2 e19 c7"
export SIMDQUALIFIER=gen
export JJVERSION=V1.1.0
export BUILDTYPE=debug
export WORKSPACE=`pwd`/workspace
export VERSIONSUFFIX=
```

```
./build-dunepdsprce.sh
```

#12 - 12/27/2019 01:35 PM - Thomas Junk

That said, JJ does take pride in his work being efficient and robust, though I got profuse apologies from him about some of the things we had to solve. Since RCE's are not the chosen technology moving forwards, and the SLAC folks could see that coming, the investment was low. But we have to read this data for many years to come of course.

There is a fcl parameter to the decoder module and tool that exports the fragments to files that can be read with the standalone reader program in dunepdsprce. I used to give these to JJ to debug. The fcl parameter is RCESaveFragmentsToFiles and the standalone program is PdReaderTest.

#13 - 12/27/2019 04:22 PM - David Adams

It sounds like we should make a dunepdsprce repository that holds your scripts and wrappers so a user could build (or rebuild) with a one line command (./dunepdsprce/build my_install_dir) or with the mrb interface so we could check it out and build like any other mrb package. Or we could do both.

The dunepdsprce repository could hold a copy of all the code in proto_dune_dam_lib or might continue to do the clone and just hold selected files that replace the cloned files before building.

Or we could put all this into dune_raw_data and include the products of dunepdsprce in the dune_raw_data product.

#14 - 12/30/2019 01:52 PM - David Adams

I would like to try some mods of proto_dune_dam_lib and am trying to start by building against the unchanged head of that package.

I built dunepdsprce from proto_dune_dam_lib following Tom's instructions with the following mods:

```
COMPILERQUAL_LIST="e17"
BUILDTYPE=proc
```

I then untarred the product (dunepdsprce-1.1.0-slf7-x86_64-gen-e17-proc.tar.bz2) in the products area in development area.

I checked out dune-raw-data and dunetpc in my local build area. I renamed the former to dune_raw_data and had to construct workdir/srcs/CMakeLists.txt by hand with these contents:

```
# Master build script for CMake (automatically generated by "mrb newDev")
```

```

# Require minimum cmake version
CMAKE_MINIMUM_REQUIRED (VERSION 2.8)

# Add top level include
include_directories ( ${PROJECT_SOURCE_DIR} )

# Enable testing
enable_testing()

# add_subdirectory and other commands will be added below by @mrb gitCheckout@ for each package

# If you add a package by hand, you MUST also add the appropriate package block below
# Remove the appropriate package block below if you delete a product from your srcs area
# Or, use mrb uc to update this file safely

# DO NOT DELETE
# dunepdsprce package block
set(dunepdsprce_not_in_ups true)
# dune_raw_data package block
set(dune_raw_data_not_in_ups true)
include_directories ( ${CMAKE_CURRENT_SOURCE_DIR}/dune_raw_data )
include_directories ( $ENV{MRB_BUILDDIR}/dune_raw_data )
# dunetpc package block
set(dunetpc_not_in_ups true)
include_directories ( ${CMAKE_CURRENT_SOURCE_DIR}/dunetpc )
include_directories ( $ENV{MRB_BUILDDIR}/dunetpc )

# DO NOT DELETE
ADD_SUBDIRECTORY(dune_raw_data)
ADD_SUBDIRECTORY(dunetpc)

```

The build succeeded (this was not my first attempt) but I realized that somewhere along the line, I lost the local installation of dunepdsprce. When I put that back, I can no longer build and, when I try to set the product up by hand, I get this error:

```

bash-4.2$ ups list -aK+ dunepdsprce
ERROR: No instance matches were made between the
version file (/nashome/d/dladams/dev/dudevUnp/workdir/localProducts_larsoft_v08_38_01_e17_prof/dunepdsprce/v1_
1_0.version) and the
table file (dunepdsprce.table) for flavor (Linux64bit+3.10-2.17) and qualifiers (e17:gen:proc)
ERROR: Possible UPS database (/nashome/d/dladams/dev/dudevUnp/workdir/localProducts_larsoft_v08_38_01_e17_prof
) corruption in product 'dunepdsprce'.

```

Do I have to do more than just untar in local_products?

I add Lynn who may be able to help with some of the ups issues.

Thanks for any guidance.

#15 - 12/30/2019 03:14 PM - Thomas Junk

Hi David,

I think you want "prod" and not "proc"

You still need to supply a QUAL even with the compilerquallist. The compiler qual list is just there for making a complete table file on each build.

The switches mrb needs to indicate that a product is not in ups (seems poorly named since the products are in ups, just in the local products directory, which should come earlier in the PRODUCTS search list than say those in CVMFS) may perhaps be avoided if you put the dunepdsprce product in its own product area and not in localProducts.

So, in a empty directory,

```

mkdir dunepdsprce_product_dir
cd dunepdsprce_product_dir
unwind the dunepdsprce .bz2 tarfile
cp /cvmfs/dune.opensciencegrid.org/products/dune/.upsfiles ./upsfiles
export PRODUCTS=`pwd`:$PRODUCTS

```

```

cd your_work_area
ups list -aK+ dunepdsprce

```

should show your new product.

```

setup dunepdsprce <version> -q <qualifiers>

```

#16 - 12/30/2019 04:22 PM - David Adams

Actually I think I want "prof". With that, I don't get the listing error and this works:

```
bash-4.2$ setup dunepdsprce v1_1_0 -q e17:gen:prof
bash-4.2$ set | grep DUNEPD
DUNEPDSPRCE_DIR=/nashome/d/dladams/dev/dudevUnp/workdir/localProducts_larsoft_v08_38_01_e17_prof/./dunepdsprce/v1_1_0
DUNEPDSPRCE_FQ_DIR=/nashome/d/dladams/dev/dudevUnp/workdir/localProducts_larsoft_v08_38_01_e17_prof/./dunepdsprce/v1_1_0/Linux64bit+3.10-2.17-e17-gen-prof
DUNEPDSPRCE_INC=/nashome/d/dladams/dev/dudevUnp/workdir/localProducts_larsoft_v08_38_01_e17_prof/./dunepdsprce/v1_1_0/include
DUNEPDSPRCE_LIB=/nashome/d/dladams/dev/dudevUnp/workdir/localProducts_larsoft_v08_38_01_e17_prof/./dunepdsprce/v1_1_0/Linux64bit+3.10-2.17-e17-gen-prof/lib
DUNEPDSPRCE_VERSION=v1_1_0
SETUP_DUNEPDSPRCE='dunepdsprce v1_1_0 -f Linux64bit+3.10-2.17 -z /nashome/d/dladams/dev/dudevUnp/workdir/localProducts_larsoft_v08_38_01_e17_prof -q e17:gen:prof'
```

Thanks for pointing out the problem. Seems like something the build script could have caught.

Now I will see if mrb find the my version...

#17 - 12/31/2019 02:28 PM - David Adams

I finally managed to build against my mods in proto_dune_dam_data. I modified build-dunepdsprce.sh to build from my area instead of a new git clone and the rebuilt dune_raw_data and dunetpc after installing dunepdsprce in my local_products directory (that precedes cvmfs in PRODUCTS).

I modified TpcTrimmedRange to make its guess for index \geq size instead of $>$ size to try to fix the problem reported in [#23811](#). There is a test in that block that gives up if the guess is negative. I also modified that to give up if the new index is \geq size. With one event, that was giving a valgrind error, there is no longer an error but we get these messages:

```
TpcTrimmedRange::Location::locate: WARNING: Invalid frame index: 6144  $\geq$  6144
%MSG-w _process_RCE_AUX:: DataPrepByApaModule:dataprep@BeginModule 31-Dec-2019 14:02:13 CST run: 4517 subRun
: 1 event: 148 PDSPTPCDataInterface_tool.cc:451
n_ch*nticks too large: 128 * 18446744073709551472 = 18446744073709533184 larger than: 10000000. Discarding th
is fragment
%MSG
<pre>
The first is from my change showing the guess was made but index value remained bad. The second shows unpackin
g tool recognizes there is a problem.
```

#18 - 02/22/2021 08:13 AM - David Adams

- Status changed from New to Closed

I suppose we can close this.

#19 - 02/22/2021 10:29 AM - Thomas Junk

Yes, fixed long ago.

Files

valgrind4517-87.out	45.1 KB	12/26/2019	David Adams
---------------------	---------	------------	-------------