Timestamp Data Product Proposal

Tom Junk

LArSoft Coordination Meeting

July 3, 2018

Many Timestamps in ProtoDUNE-SP

- ProtoDUNE-SP has 15360 channels divided into 6 APA's of 2
- Five of Six APA's are read out using RCE's (SLAC)
- One of Six APA's is read out using FELIX (NIKHEF)
- Timing, Trigger, Cosmic-Ray Tagger, Photon systems read out in the same data stream.
- Beam Instrumentation data collected via a separate data stream and associated offline.
- art EventAuxiliary branch has a time (2x32 bits)
  - in ProtoDUNE-SP, the event builder fills this with a read from the system clock when the first fragment arrives. Resolution ~1 sec.

Timestamps in ProtoDUNE-SP

- Even the RCE's are not monolithic
  - they run freely, when booted, they pack up 1024-tick packets for the 512 channels they see.
  - Events are 10,000 ticks long. Enough packets are sent from the RCEs to the boardreaders to make a whole event. These are stored in the output file.
  - Offline, these are trimmed to size and aligned.
  - Individual ticks may drop from the WIBs (Warm Interface Boards) to the RCEs
- Separate 64-bit timestamp is sent up for each 128-channel data block. I believe this comes from the timing system (next page)
- At the very least, 50 different timestamps possible in an event just from the RCE's.
- FELIX also sends up timestamps with its raw data.
- FELIX times are also "supposed" to match.

Current Timestamp Storage

- The timing system writes a fragment that has a 64-bit timestamp, and a four-bit timing command word
- This 64-bit timestamp is a counter of a 50 MHz clock (internal, but later synched to the SPS).
- This counter is reset infrequently (does not zero at 1970 for example)
- The TPC ticks at 2 MHz, so there are 25 possible phases.
- We can read the artdaq fragment to get this value.
- Or we can copy it into another data product (similar content), but it may reduce dependency on dune_raw_data, where the artdaq fragment overlay is defined.
- No place to put any of these timestamps into the event

Previous Work

- In lardataobj/RawData/DAQHeader.h, there is a candidate data product.
  - It has a time_t in it, but also lots of other words we don't need from the DAQ480 software. Status, format, software version, event, spare, nchans, "fFixed".
  - Would be a waste to create one of these for each channel for each event.
- lardataobj/RawData/ExternalTrigger.h is almost exactly what we need. It has a 64-bit timestamp and an unsigned int. They're called "TrigTime" and "TrigID" though, and we'd like to put other things in them. Just names though. Would be odd to associate these with raw digits.
- MicroBooNE has uboone/RawData/utils/DAQHeaderTimeUBooNE.h with two time_t's. But it says "MicroBooNE"

Proposal

- We would like a timestamp per raw digit:
  - New data product, RDTimeStamp, with one 64-bit timestamp storage and one 16-bit set of flags (raw::RawDigits have no flags)
  - Defined in lardataobj near Raw::RawDigit
  - ProtoDUNE-SP would like to create one of these per TPC channel and associate them with the corresponding RawDigit
  - Granularity is a bit fine – one could imagine a many-to-one association for all the raw digits guaranteed to have the same timestamp.
  - But the timestamps (and flags) compress very well – many duplicate values.
  - In fact, the associations take more space
- Data product RDTimeStamp.h, classes_def.xml, and classes.h for lardataobj/rawdata is in a feature branch trj_rawdata_timestamp_jun2018

Product Sizes

- 10 events, 1 APA, ROOT compression level=1

Size in bytes	Fraction	Data Product Name
322364908	0.664	artdaq::Fragments_daq_ContainerTPC_DAQ.
162995998	0.336	raw::RawDigits_tpcrawdecoder_daq_RunRawDecoder.
75285	0.000	raw::RDTimeStampraw::RawDigitvoidart::Assns_tpcrawdecoder_daq_RunRawDecoder.
5236	0.000	raw::RDTimeStamps_tpcrawdecoder_daq_RunRawDecoder.

Mitigation

- We plan on re-unpacking daq fragments when we need to process raw digits. So we may not even store raw digits and these data products to disk/tape in bulk.
- The bulk of the processed data may not have raw digits saved. Andrew predicted a factor of 2x data inflation for reco passes.
- This data product may exist primarily in memory.

- Having a separate data product with associations also means no impact for other experiments.

Alternative Proposal

- Our original idea was to add the timestamp and flags to the raw::RawDigit data product
- Compresses well, no associations
- Might cause confusion for experiments not needing it.
- Backwards compatibility okay, forwards compatibility no – force everyone to move to new version of code reading persisted raw digits.

Comments

- Timestamp meaning not defined. It's declared as a uint64_t, which could be anything. Experiment specific (and possibly subsystem specific).
- Flags meaning is also not defined. People have to settle on their own conventions
- Other information that doesn't quite fit into raw::RawDigits: dropped WIB frames (1 tick x 128 channels, may come in clusters or sporadically). We can fill in the missing digits with zeros, keeping the rest aligned, and triggering our stuck-code mitigator. Our firmware person, JJ Russell at SLAC, wants to provide descriptors for every contiguous data packet. But we are limited to raw::RawDigit (plus things we can associate with it). And we don't want to have to rewrite downstream software. How to indicate missing data on the event display for example?