Storing the initial tick of TPC waveform in `raw::RawDigit`

Gianluca Petrillo

Fermi National Accelerator Laboratory

LArSoft coordination meeting, March 27th, 2018

Timing in LArSoft

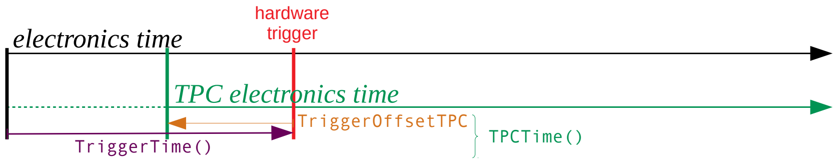
Documentation on LArSoft time frames can be found at:

- → MicroBooNE DocDB 12290 (Herbert Greenlee, October 2017)
- → `detinfo::DetectorClocks` documentation in LArSoft Doxygen

Relevant to this proposal are the time frames:

"electronics" time kind of glue

TPC electronics time when TPC waveforms are expected to start



Timing in LArSoft: more than you want to know

Time frames:

"electronics" time	kind of glue
TPC electronics time	when TPC waveforms are expected to start

In `detinfo::DetectorClocksStandard` ("standard" implementation) their relation is determined by two configuration parameters:

`TriggerOffsetTPC`	start of TPC electronics time with respect to the hardware trigger instant
`DefaultTrigTime`	*default* hardware trigger instant

- in simulation, default time is commonly used
- in data,
  1. time from a `raw::RawTrigger` object is used if available
  2. otherwise, the configured default trigger time is used

  $\rightarrow$ time internal from the start of the TPC waveform to the trigger is always the same for all events

  $\rightarrow$ time interval from the electronics time to the TPC waveform may change event by event

Reducing the raw data size

Data products:

`raw::RawDigit` ticks start from `TPCTime()` (implicit convention)

`recob::Wire` ticks are measured from `TPCTime()`

`recob::Hit` and derivates are measured from `TPCTime()`

When MicroBooNE decided to “chop” the start of `raw::RawDigit`, they had to:

- reconfigure the *global* timing setting (`TriggerOffsetTPC`)
- reprocess `raw::RawDigit`
- reprocess the reconstruction, which would have all times shifted

$\rightarrow$ proper time alignment depends on the amount of chopping

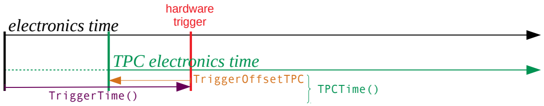
$\rightarrow$ data products from chopped and unchopped waveforms can’t be used in the same job

I have received a lot of support requests related to this, and I assume so did MicroBooNE people.

The proposal

I would like a more robust system to cope with the “chopping”...
My proposal:

- stage 1: a new “degree of freedom”: store the value of the first tick in `raw::RawDigit`
  - first tick is *with respect to* `TPCTime()`
  - a default value of `0` makes the change backward-compatible
  - bonus: allows different chopping for different channels
  - reconstructed data products *still measured in TPC electronics time*[1]
- stage 2: store reconstructed quantities in the same time frame
  - suggesting *electronics time frame*
  - allows to disregard which offset was used, all data products are on equal footing
  - breaking change: it’s a convention change



[1]Note that the default settings of most experiments set the electronics time and TPC electronics time frames to match.

Summary and discussion

- the amount of issues caused by `raw::RawDigit` chopping betray a design problem
- adding a bit of information to the data product might be a simple and *good* solution
- this can be implemented in a backward-compatible way...
- ... or, with more ambition, as breaking change
- question to the stakeholders: is this worth?

Disclaimer

Note: this is *my personal proposal*:

- MicroBooNE has not requested any action
- I have not previously discussed this proposal with them

Thank you for your consideration!

Why is the “stage 2” a breaking change

The change of convention moving the reference time frame from *TPC electronics* to *electronics* time is breaking:

- it changes the *interpretation* of the data product information
- the code will require a different time conversion
  - no conversion at all when comparing with *electronics time*
- only for experiments where TPC electronics and electronics times do not match
- only in data sets where TPC electronics and electronics times do not match