RCDAQ and EICrecon integration

Martin Purschke

with a lot of help from Dmitry, Derek, and Wouter

We have supported EIC-themed test beams with RCDAQ for ages, so *taking* the data is not an issue.

We have been going on about making a full “in-house” raw data -> EICrecon analysis toolchain

We really want and need to gain experience with a standard ePIC analysis for raw data so people get proficient with the tools that we have

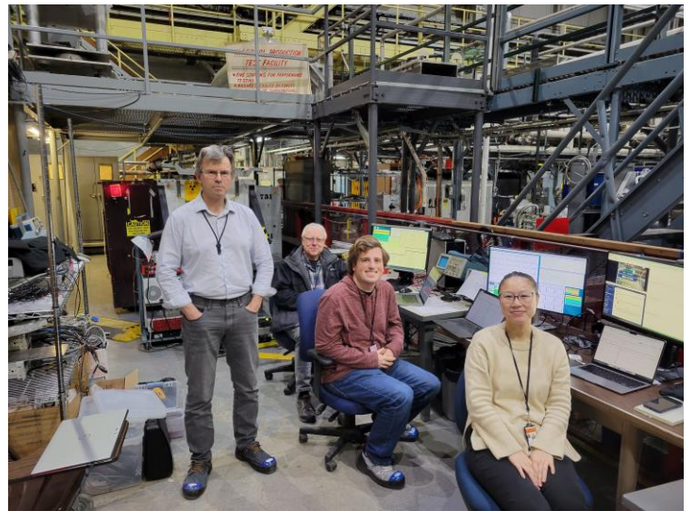
We will also see what features need to be implemented or need improvement

We have 3 data taking campaigns going on, all using RCDAQ and the assorted online monitoring tools:

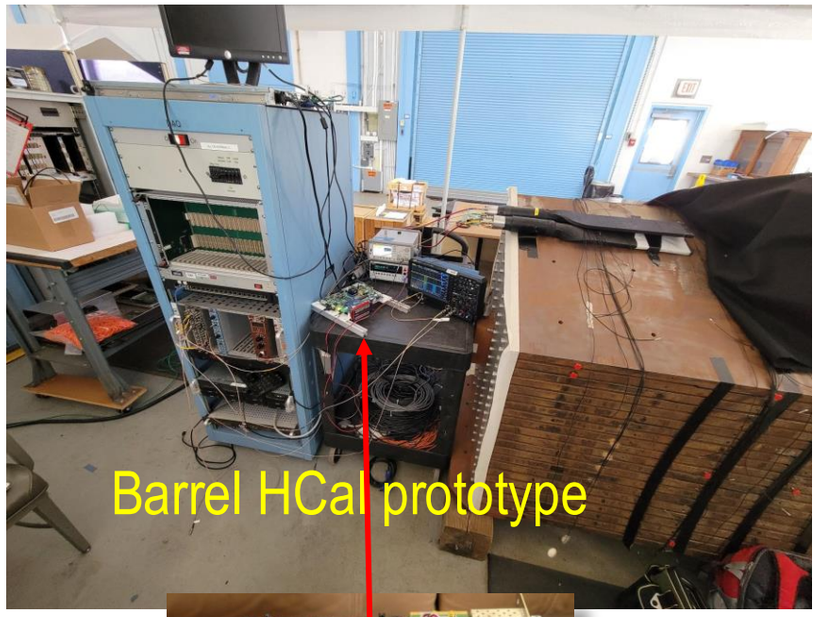
- the ”ePIC ZDC” test beam at the BNL NASA Space Radiation Lab (NSRL);
- the BHcal test setup in the Phys. Dept. HighBay area;
- the HRPPD high-field test in the BNL Magnet Division (Bldg 902) for the pfRICH

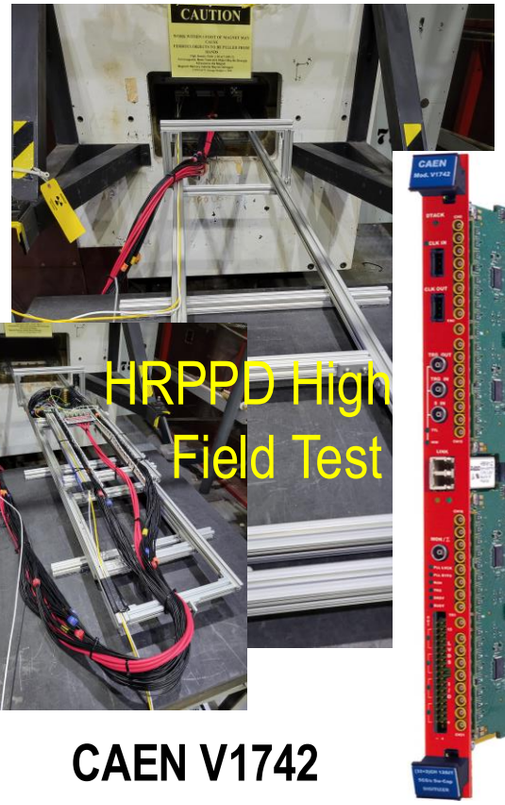
Current Data Taking Campaigns

While the ZDC test is an actual test beam, the HRPPD feels most like an actual one (data taking shifts and all…)

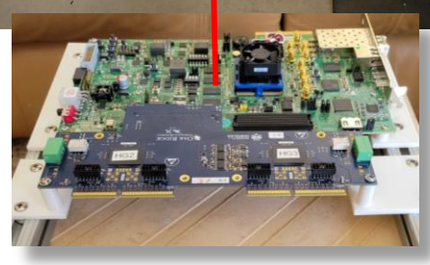


Beam Dump



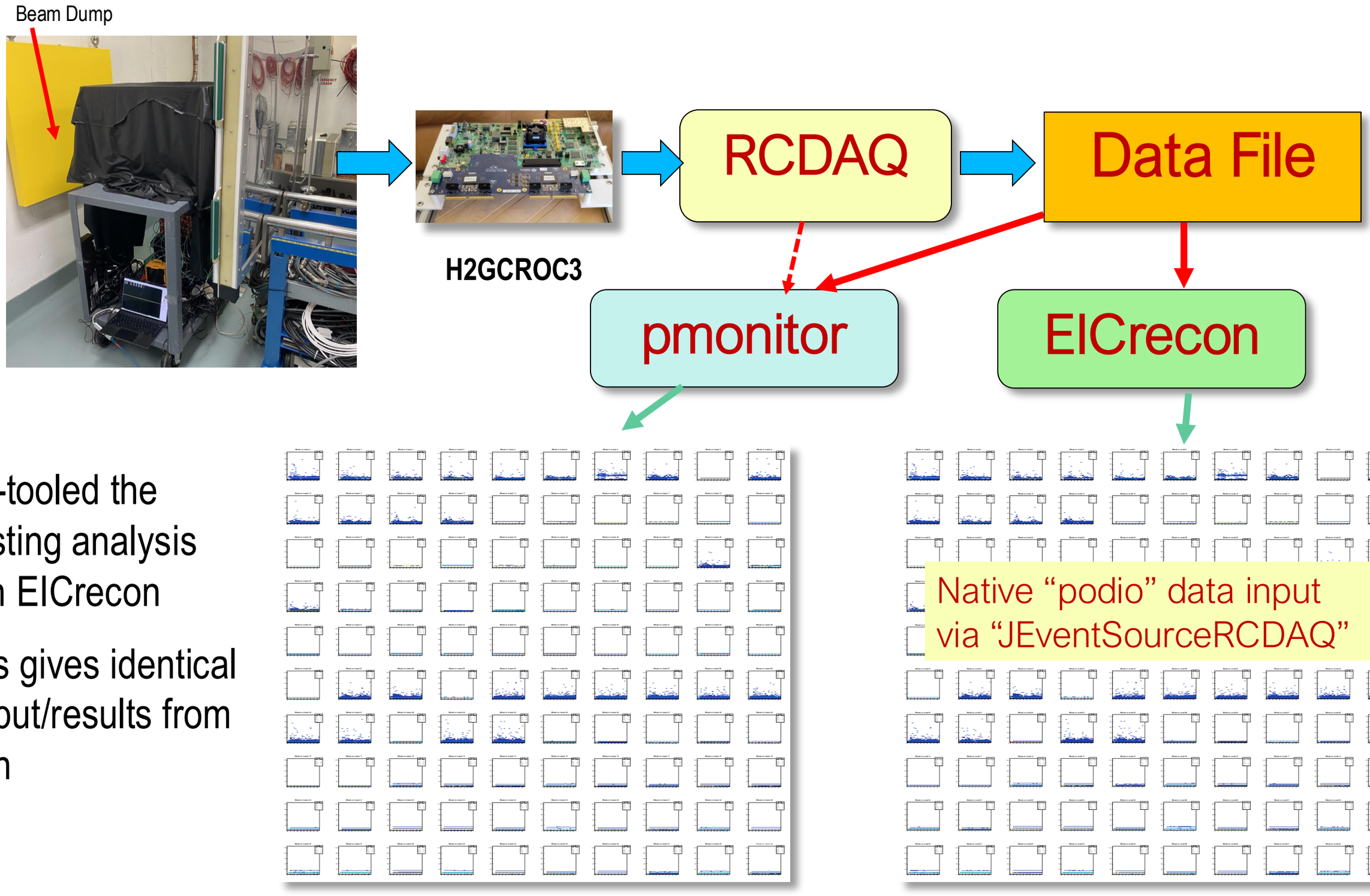


H2GCROC3 readout



CAEN V1742 waveform digitizer

Just to jump way ahead:



I re-tooled the existing analysis with EICrecon

This gives identical output/results from both

Quick recap of RCDAQ's data format

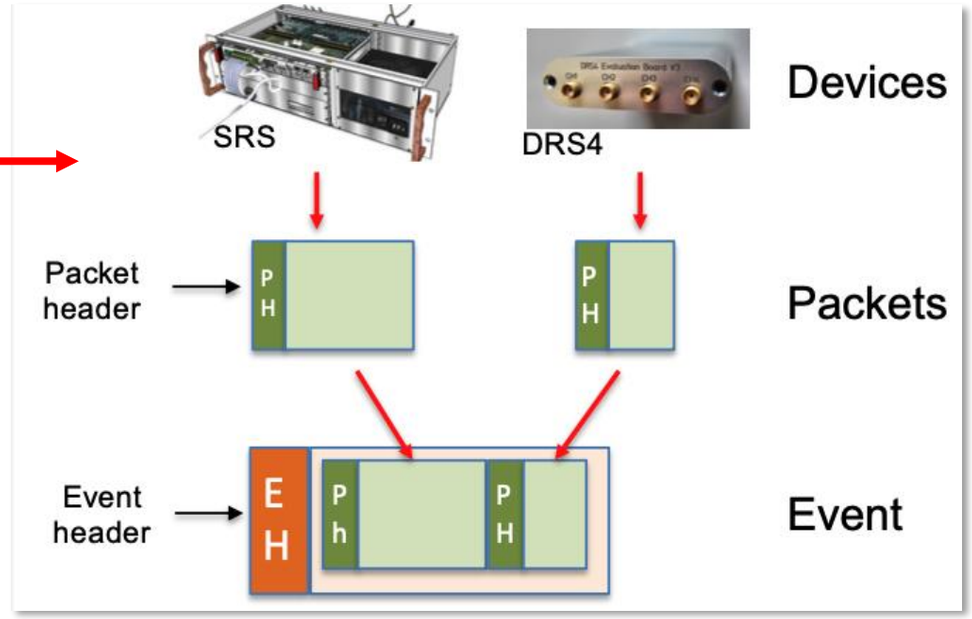
You see the data as "Events", each of which contain "Packets"

"Event" has become a bit of a mis-nomer since we have streaming data, but old names die hard

Each readout device that's registered with RCDAQ adds a "Packet" to the Event

The devices can be diverse, and usually are

This is from an early EIC-themed test beam at Fermi




```
$ dlist beam_0000004094-0000.evt
Packet  1010 23262 -1 (sPHENIX Packet)  70 (IDSRSV01)
Packet  1020  5126 -1 (sPHENIX Packet)  81 (IDDRS4V1)
```

Each packet has an ID that uniquely identifies what part of the detector is connected here

The ID is just a number - we typically impose an algorithm to pick IDs

For the H2GCROC3 we (arbitrarily) chose 12001

And with just one device, there is only one packet here:

```
$ dlist beam_ZDC-00001112-0000.evt
Packet 12001 36504 -1 (ePIC Packet) 301 (IDH2GCROC3)
```

There is no prescribed structure to the Packet payload

For all intents and purposes, the data payload is treated as a binary blob without a fixed structure

That “structure” comes in via data access APIs that standardize the access to the data

It also allows us to change the internal encoding of the payload without breaking user code – the APIs still give you the same answer. Your code will not depend on the inner encoding of the data.

That API keys on the “hitformat”. Change the inner encoding, change the hitformat, make a new decoder.

```
Packet 12001 36504 -1 (ePIC Packet) 301 (IDH2GCROC3)
```

As we evolve the H2GCROC3 hardware and firmware and the inner encodings, there will be more hitformats coming our way. *

The “301” is just an enumeration and has no further deep meaning. We assign it a mnemonic.

None of those future changes will break our code.

* For the sPHENIX TPC (FELIX) format, we are currently on our 5th hitformat, often for denser packing of the data. Any reconstruction/user code still works the same.

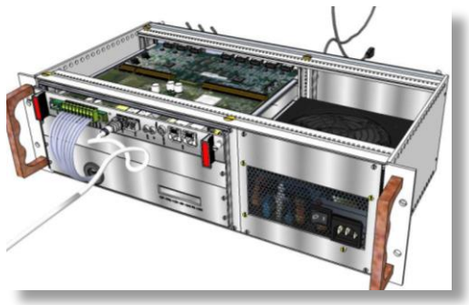
Common Analysis API for Waveform Samplers

What's a waveform sampler? A thing with many channels, each of which captures a waveform

Your digital scope is a waveform sampler, typically with 4 channels (and a screen attached) ☺

Here are a few examples (all supported in RCDAQ)

DRS4 Eval board



CERN RD51 SRS System

CAEN V1742 waveform digitizer

SAMPA/SALSA ASIC

H2GCROC3

If p is a pointer to your Packet object, then

```
p->iValue(0,"CHANNELS")       tells you how many channels that device has  (H2GCROC3:144)

p->iValue(channel,"SAMPLES")  tells you how many samples are present for that channel

p->iValue(c,s)                gives you sample s from channel c  (different APIs available)
```

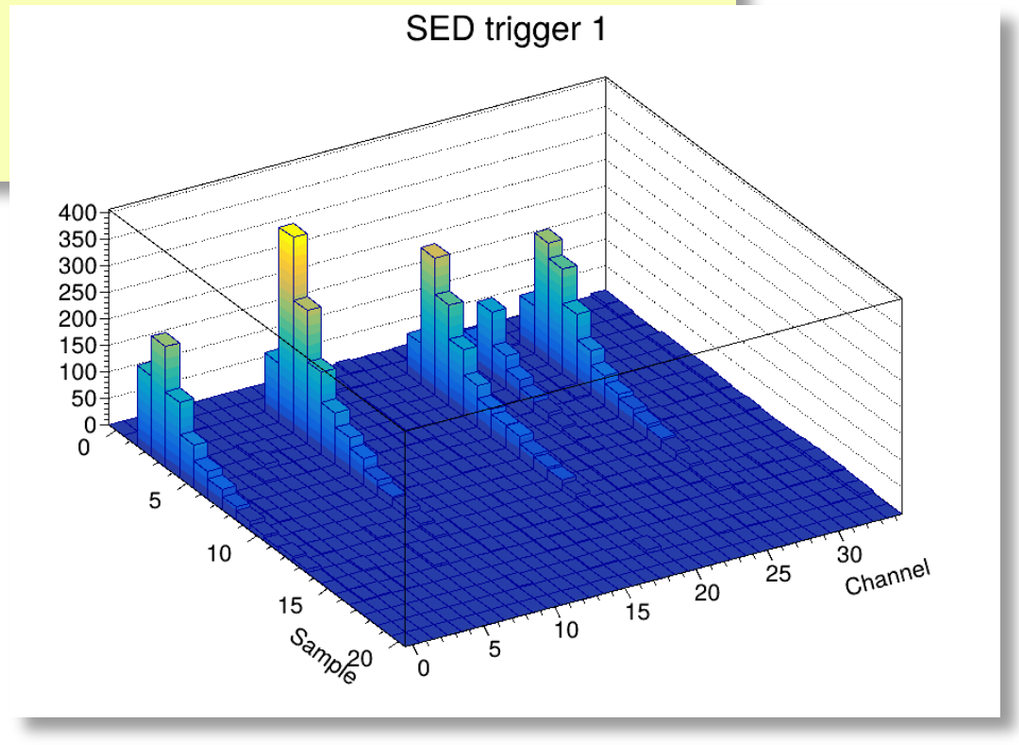
Often the #samples are the same for all channels, so you can get away with asking only once, as I'll show in the JEventsource code

And all those devices can share this kind of code

```
Packet *p = e->getPacket(12001);
 if (p)
   {
     for ( int c = 0; c < p->iValue(0,"CHANNELS"); c++)
     {
       for (int s = 0; s < p->iValue(c,"SAMPLES"); s++)
       {
         h2->Fill(c,s, p->iValue(c,s));
       }
     }
     delete p;
  }
```

I virtually always make what I call the “mlp standard plot” with Single Event Displays

The above code works the same for all of the devices from before



The H2GCROC3 has 2 additional fields

Instead of just a “waveform” vector of ADC values, the H2GCROC3 vector is triplet of

{ADC, “Time of Arrival”, “Time over Threshold”} instead

```
p->iValue(c,s)              gives you sample s from channel c

p->iValue(c,s,”TOA”)        gives you TOA value s from channel c

p->iValue(c,s,”TOT”)        gives you TOT s from channel
```

These are excerpts from the JEventSourceRCDAQ that you can now recognize:

```
 int channels = p->iValue(0,"CHANNELS");
 int samples =  p->iValue(0,"SAMPLES");
// . . .
 for (int ch = 0; ch < channels; ch++)
             {
               edm4eic::MutableRawHGCROCHit obj = coll.create(1200 + ch, 0);
               for ( int s = 0; s < samples; s++)
                 {
                   uint16_t adc = p->iValue(ch, s);
                   uint16_t toa = p->iValue(ch, s, "TOA");
                   uint16_t tot = p->iValue(ch, s, "TOT");
                   obj.addToSamples({adc, toa, tot, totp, totc});
                  }
```

Data access

A simple EICrecon run with our data file as input:

```
$ eicrecon -Ppodio:output_collections=p_1200 beam_ZDC-00001112-0000.evt
```

Resulting in a podio root file

```
*.....................................................................*
*Br    6 :_p_1200_samples : Int_t _p_1200_samples_                     *
*Entries :     3699 : Total  Size=     209008 bytes  File Size  =       7090 *
*Baskets :        3 : Basket Size=      32000 bytes  Compression=   4.21     *
*.....................................................................*
*Br    7 :_p_1200_samples.ADC : UShort_t ADC[_p_1200_samples_]         *
*Entries :     3699 : Total  Size=  192730021 bytes  File Size  =   56652292 *
*Baskets :     1759 : Basket Size=   25600000 bytes  Compression=   3.40     *
*.....................................................................*
*Br    8 :_p_1200_samples.timeOfArrival :                              *
*         | UShort_t timeOfArrival[_p_1200_samples_]                   *
*Entries :     3699 : Total  Size=  192747651 bytes  File Size  =    1664426 *
*Baskets :     1759 : Basket Size=   25600000 bytes  Compression= 115.78     *
*.....................................................................*
*Br    9 :_p_1200_samples.timeOverThreshold :                          *
*         | UShort_t timeOverThreshold[_p_1200_samples_]               *
*Entries :     3699 : Total  Size=  192754703 bytes  File Size  =    1100669 *
*Baskets :     1759 : Basket Size=   25600000 bytes  Compression= 175.09     *
*.....................................................................*
*
```

Summary

Full “in-house” analysis toolchain RCDAQ->EICrecon analysis: check.

(that’s what we wanted to demonstrate).

What’s next? (after some more polishing…)

I’ll re-tool another existing analysis (likely the magnet test where I’m currently most invested in) in EICrecon.

- Done for the ZDC beam data as shown before
- Good to be able to compare results that Alexander’s cohort gets and see that it’s the same.

We should have a discussion (not the right forum here I think) about the general strategy for raw data. Raw data have some additional requirements over simulation data, chiefly the meta-data/special events etc

I’ll need some additional classes for the generic waveform samplers, and some mods to HGCROCSample class

Some PRs coming our way…

Add this support (chiefly the ”Event Libraries” that decode the data) to the standard container in some form

Minus some polishing/tooling, and the PRs, we are set.

If you need data taking support, please get in touch and the DAQ group will find you a solution.

Backups

Generic waveform sampler data

You are tempted to use the “sample number” as a proxy for “time”

That assumes that the ASIC’s/FPGA’s system clock absolutely accurately drives the “sample time”

We currently make that assumption in the H2GCROC3 data we take

That is not really true – in each ASIC, the assorted storage cells have a personality and the times are not equidistant. This is from a CAEN 1742 unit running at 5GS, so each time diff should be 200ps – not so.

sample	time [ns]	ADC	time delta
0	0	2093	
1	0.199501	2091	0.199501
2	0.399002	2093	0.199501
3	0.598495	2095	0.199493
4	0.797997	2093	0.199501
5	0.997299	2094	0.199303
6	1.1965	2092	0.199203
7	1.3958	2093	0.199295
8	1.5949	2093	0.199104
9	1.794	2095	0.199097
10	1.993	2096	0.198997
11	2.1921	2094	0.199104
12	2.3909	2095	0.198799

...

```
## An individual sample for generic waveform samplers
edm4eic::GenericWFSample:
  Members:
    - float timeStamp
    - int16_t ADC          // [ADC Counts], signed int
b/c after pedestal subtraction it can be negative
```

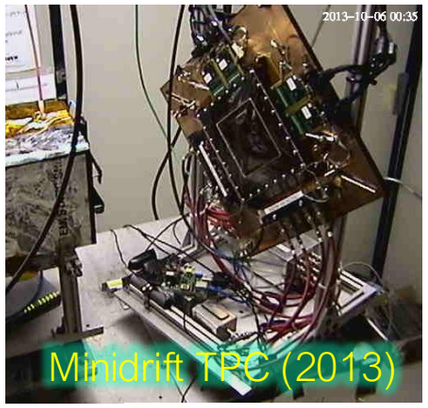
EIC-themed test beams with RCDAQ

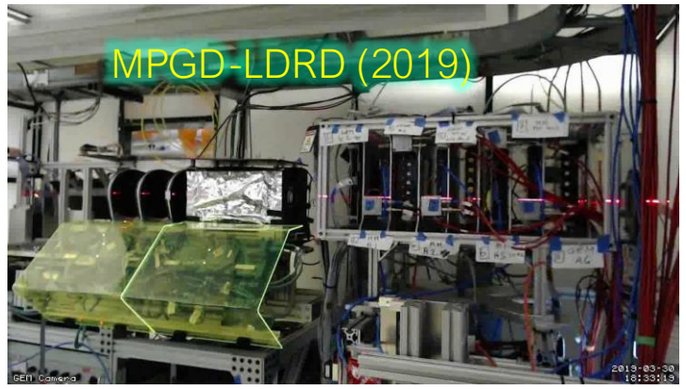
The RCDAQ system has been a pillar of EIC-themed data taking for R&D, test beams etc since 2013 – eRD1, eRD6, LDRDs, …

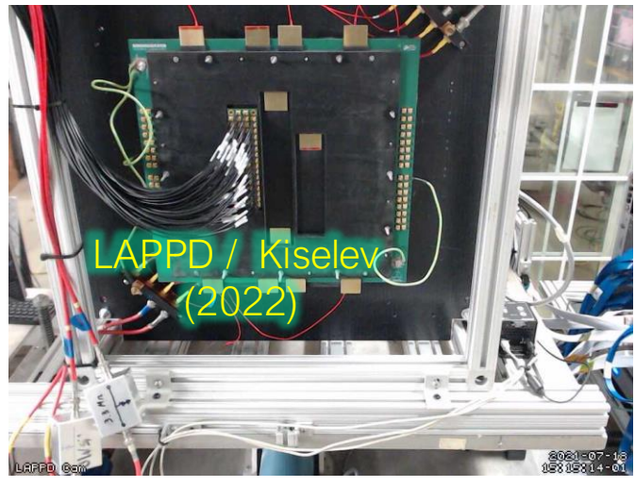
Many active RCDAQ installations in the ePIC orbit + ~30 elsewhere

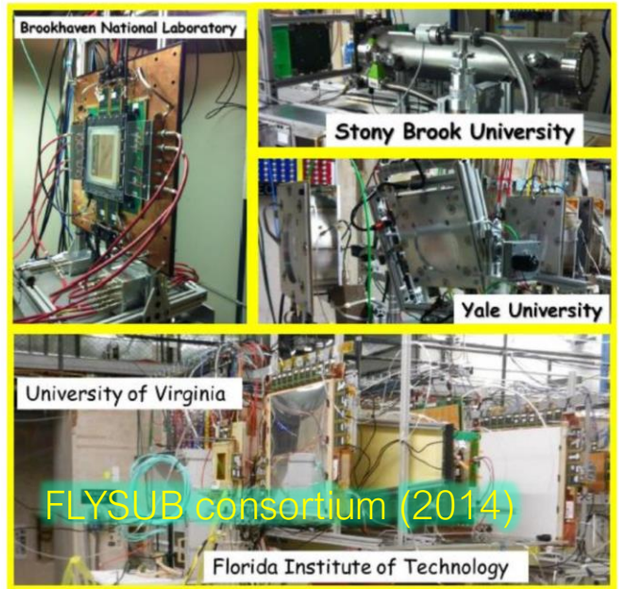
Usual entry by ease-of-use for standard devices (DRS, SRS, CAEN, …) and support for fully automated measurement campaigns

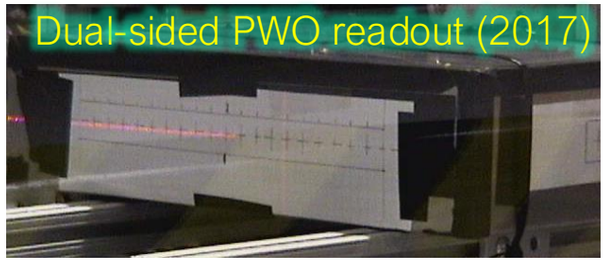
Manuals and “application notes” at https://www.phenix.bnl.gov/~purschke/rcdaq

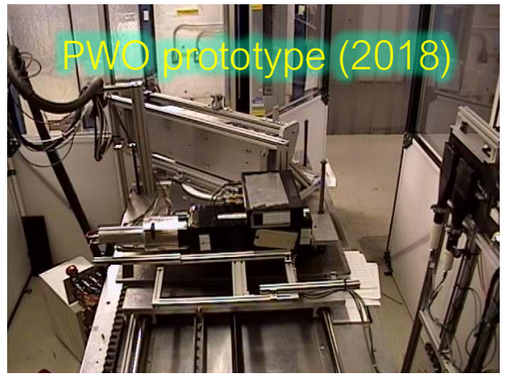












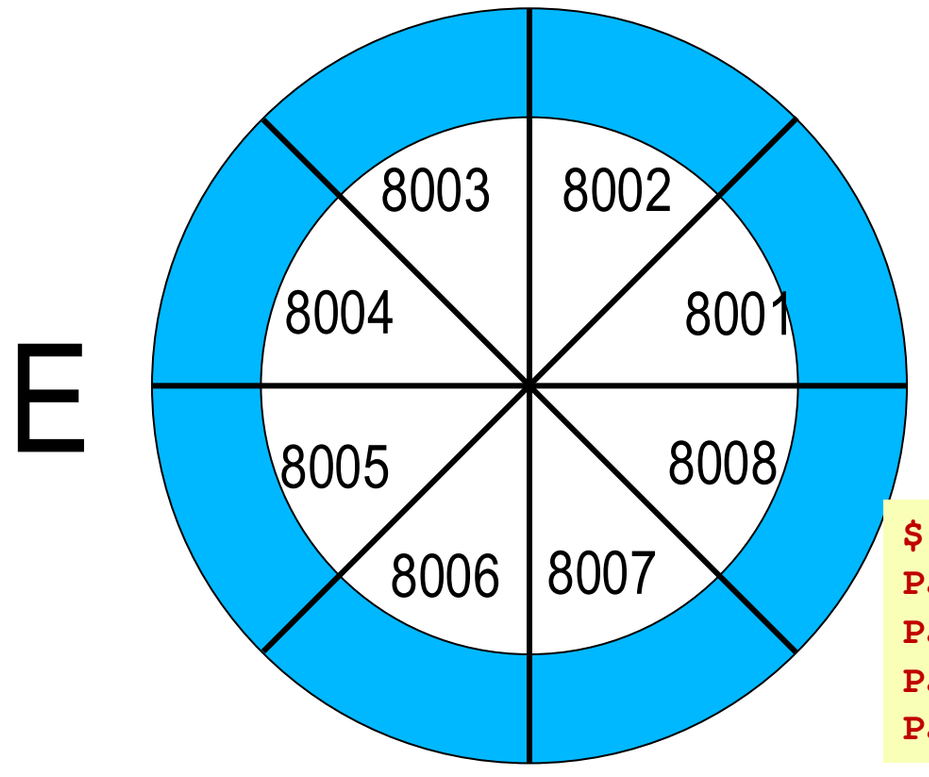
Thanks to Jin Huang for this collection

PacketID Assignment Example (BHCal)

In sPHENIX we assign each detector system a “Detector Number” – our Hcal happens to be #8

Each detector then can assign packet IDs in the d_nr * 1000 +1 …. d_nr * 1000 +999 range

8001… 8999 for the Hcal. Much more than we need



So we have 8001, 8002, 8007 and 8008 on the west side.

8003, 8004, 8005, 8006 on the east side

All 8 together constitute the data of the entire HCal

```
$ dlist /bbox/bbox6/W/HCal/cosmics/cosmics_seb16-064906-0000.prdf
Packet   8001   788  0 (Unformatted)   172 (IDDIGITIZERV3_12S)
Packet   8002   676  0 (Unformatted)   172 (IDDIGITIZERV3_12S)
Packet   8007   753  0 (Unformatted)   172 (IDDIGITIZERV3_12S)
Packet   8008   753  0 (Unformatted)   172 (IDDIGITIZERV3_12S)
```

```
$ dlist   /bbox/bbox7/W/HCal/cosmics/cosmics_seb17-064906-0000.prdf
Packet   8003   746  0 (Unformatted)   172 (IDDIGITIZERV3_12S)
Packet   8004   732  0 (Unformatted)   172 (IDDIGITIZERV3_12S)
Packet   8005   662  0 (Unformatted)   172 (IDDIGITIZERV3_12S)
Packet   8006   690  0 (Unformatted)   172 (IDDIGITIZERV3_12S))
```

The channel mapping for one device /packet repeats for each packet – super-easy to deal with