Status of SRS/APV25 integration into EUDAQ2/Corryvreckan

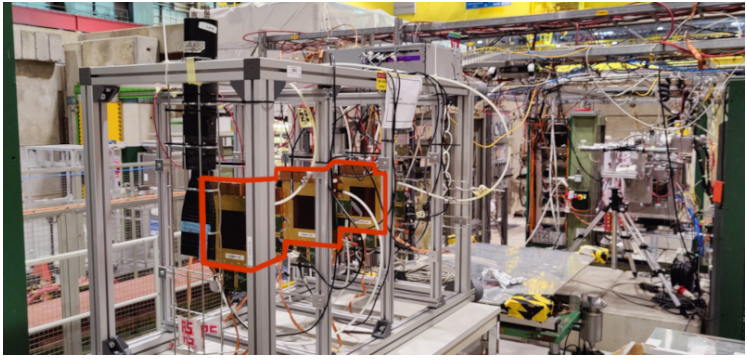
RD51 Collaboration Meeting

D. Figueiredo (INFN Pisa) L. Forthomme (AGH) F. Garcia (HIP)
<laurent.forthomme@cern.ch>

4-8 Dec 2023

Introduction: scope and goals

Development of a new reference tracking system for the TOTEM-CMS R&D programme @ SPS-H8



[F. Garcia]

Helsinki triple-GEM detector

- 10×10 cm^2 area coverage, with ~60 μm transverse spatial resolution
- low-cost, $ArCO_2$ (70%/30% mixing) gas telescope
- readout through APV25 hybrids, large user "know-how" allowing for fast applications, and still widely available at CERN store
- SRS technology for readout, also compatible with RD51's VMM3/VMM3a
- major drawback: **outdated SW chain** downstream SRS (DATE+AMORE/mmDAQ), decade-long unmaintained

Past + present

Present scheme: DATE/AMORE

ALICE's DATE ("Data Acquisition and Testing Environment")

- Designed for **multi-processor data acquisition**, suitable for large systems involving 100s of CPUs
- Multiple subsystems for triggering (CTP/LTU/TTC), assembling of fragments (event builder), distribution of sub-events (LDC), transient and permanent storage (TDS/PDS)
- **Interprocess communication** through TCP/IP link
- Includes run control, load balancer, monitoring and reporting tools

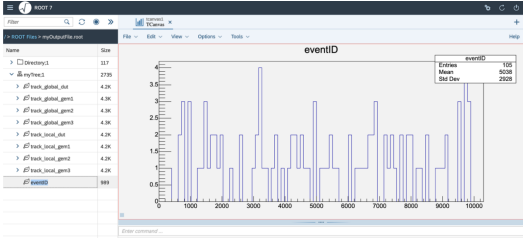
ALICE's AMORE ("Automatic Monitoring Environment")

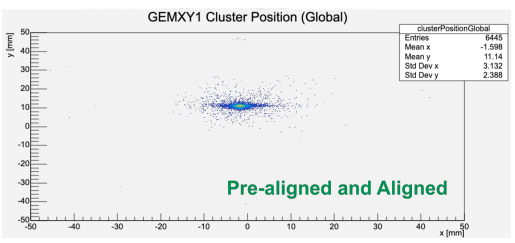
- Online SW for the **monitoring** (DQM) and **online reconstruction** of early ALICE data flow (2008→~2015)
- Modular, "publisher"/"subscriber" agents model ; interprocess communication through CERN-DIM layer
- Distributed as RPM for *very* outdated OS/software stacks
  - practically runs only on "ancient" x86 SLC5 configurations → virtual environments
  - does not survive any "critical" safety update, not fit for remote controlling
- custom `amoreSRS` overlay developed for RD51 [K. Gnanvo, S. Colafranceschi *et al.*]
- **Input**: set of scripts publishing stream from DATE acquisition
- **Output**: trigger-granular ROOT trees, containing clusters + pedestals data for offline processing

Present scheme: DATE/AMORE ↦ Corryvreckan

Corryvreckan - flexible, fast and lightweight test beam data reconstruction FW

- Modern equivalent of a same, modular reconstruction chain: event loaders, analysis modules, output file writers
- Development of an **event loader** based on AMORE cluster tree [D. Figueiredo]
  - propagation of AMORE event ID leaf to Corryvreckan's `eventID`
  - new "`gemrd51`" detector type for **alignment** and "4D-tracking"
  - new "`TreeWrite`" module saving all "useful" operational parameters (track intercept parameters for all GEMXYs and DUT)
  - DATE/AMORE equivalent of `mmDAQ`-level work reported by Elena *et al.* this morning





- Several monitoring quantities implemented: beam profile from clusters, ...
  - still a work in progress: specific *"à la AMORE"* plots to be implemented
  - (month)-old development, alread mature for analysis of multiple test beam campaigns for TOTEM GEM reference tracker

Future

Reminder: a driver for SRS communication[1]

Development of a `srsdriver`, handling all parts of UDP communication between DAQ and SRS server

- single library managing configuration/run registers definition & handling, **slow control** and **data collection**/parsing
  - shared `.so` object + collection of `.h` headers, released under **GPLv3**
  - shipped with several debugging and testing executables, including its own implementation of the `slow_control` utility, 1-to-1 compatibility with all `start.sh`/`stop.sh`/`physicsRun.sh`/… "control room scripts"
- **Python bindings** for SRS/APV25 configuration scripting
  - allows generation of "standard" ASCII configuration files
  - work in progress: direct configuration through UDP transfer with slow control object

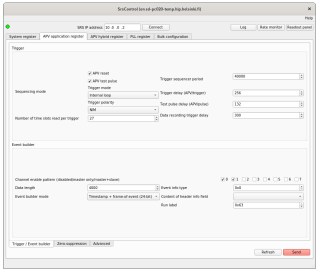
```
import pysrsdriver as srs
sc = srs.SlowControl('10.0.0.2')
apvapp = sc.readApvAppRegister()
apvapp.triggerDelay = 0x100 # BCKL_TRGDELAY
apvapp.triggerSeqPeriod = 40000 # BCKL_FREQ
apvapp.save('apv0.txt', '10.0.0.2', 6263)
sc.addFec(6006)
if sc.readout:
    frames = sc.read(0)
    # (...)
```

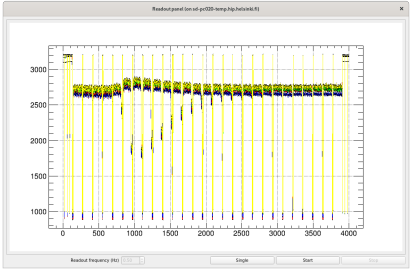
- shipped with a "trivial" C++ SRS **frames unpacker**
  - accepting any base class-derivative for custom, user-defined data format cases
  - can be interfaced to any C++-based readout component

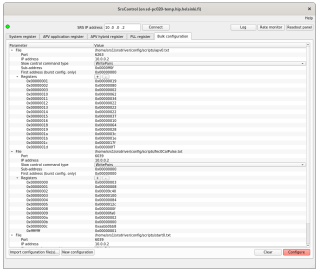
[1] Follow-up from Feb 2021 RD51 mini-week

(Other) reminder: slow control

`srscontrol`: Qt5-based GUI for the determination of all run parameters, and *live* testing of various configurations







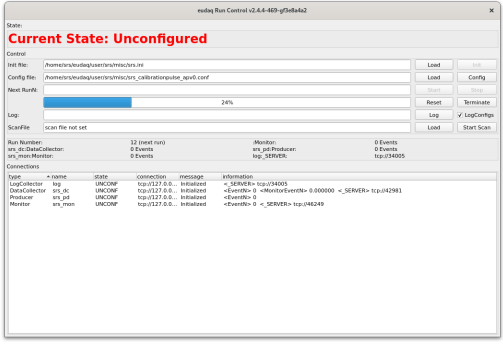
- designed as a possible replacement for the LabView-based application
- directly connects all `srsdriver` registers objects methods to a user-friendly interface
- ROOT-based monitoring, can be easily extended for more advanced online features (fits, pedestal subtraction, . . .)

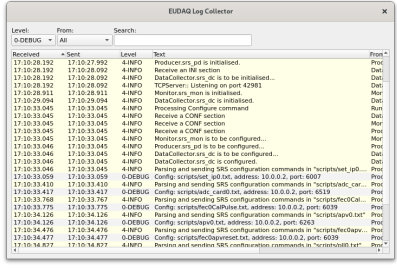
Direct loading of system/APV/PLL/APV-application registers for parameterisation before “reupload” to SRS

- **advanced GUI** for **simple configuration**
- **simple key/value configuration editor** for more **advanced usages**

New: EUDAQ2 integration[2]

Integration of the present SRS configuration/running/acquisition scheme into a EUDAQ2 user module

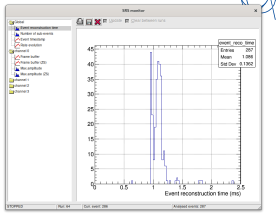




- Introduction of a new `eudaq::Producer` and `eudaq::DataCollector` for the SRS **configuration**, and **collection**, **unpacking**, and **storage** of all SRS frames
  - configuration through a collection of input ASCII files, similar as “console-based” slow control
  - making use of standard `srsdriver` event unpacking capability ; APV25 frames unpacking supported natively
- Nicely fits into TOTEM test beams’ environment, EUDAQ2 producer for SAMPIC/time reference

[2] Still a work in progress! Lots of little tasks to be foreseen along development work.

EUDAQ2 integration (cont.) → Corryvreckan integration



- Online monitoring through a `eudaq::ROOTMonitor` (as introduced in EUDAQ v2.4.2)
  - supports “on-the-fly”, automatic discovery/unpacking of all channels present in data stream
  - allows for offline “replay” of acquisition, running on RAW EUDAQ files, and generating a ROOT `TDirectory` of summary plots

- Can profit from `amoreSRS` publisher implementation to port main analysis/monitoring features into a modern environment
- Outside SW developments, still a bit of “massaging” for the clock distribution/trigger logic
  - TLU/SRS interfacing to be implemented ; in the meantime, can run in SRS-synchronous mode
  - handshake procedure, event counter+builder synchronisation to be developed

```
[EventLoaderEUDAQ2]
name = "gemrd51"
```

Corryvreckan interfacing

- EUDAQ2 file import already handled natively, interfacing work for SRS/APV25 frames unpacking to be foreseen
- large overlap in geometry definition for GEM-XY/DUT, will profit from experience developed in DATE/amoreSRS interfacing
- direct usage of a good variety of alignment/clustering/tracking (including GBL) algorithms

"Take-home messages"

- new DATE/AMORE import module for Corryvreckan
- `srsdriver` development stabilised, can now act as a "library-in-the-middle" for further developments
  - 1-to-1 handling of all parameters/modes covered in the SRS/APV25 manual
  - further development of Corryvreckan reconstruction may highlight some required features
- candidate GUI for control and operation parameters decision: `srscontrol`
- EUDAQ2-SRS module already able to operate/configure/acquire/store without external control

Lots of *little* interfacing tasks to be covered…

To quote a few…

- porting of former `amoreSRS`/`mmDAQ` preprocessing tools into EUDAQ2/Corryvreckan environment
- interfacing of full APV25 frames unpacking before propagation to Corryvreckan

Any help is more than welcomed!

Thanks a lot for your attention!

Spares