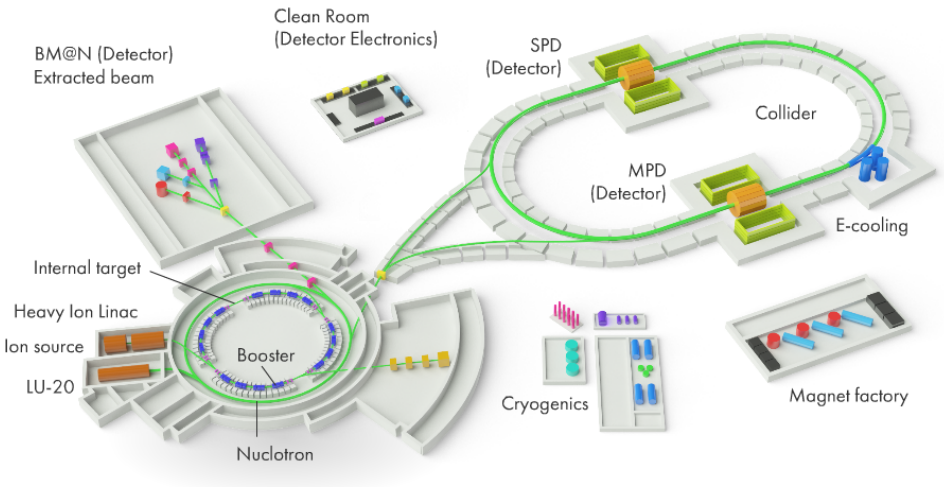
The BM@N experiment online data processing and QA system

Ilnur Gabdrakhmanov

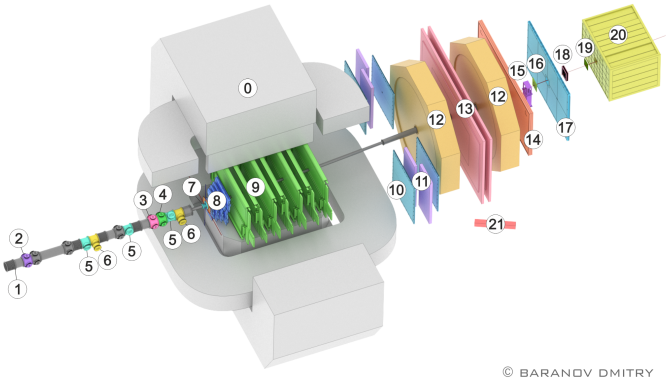
Veksler and Baldin Laboratory of High Energy Physics
Joint Institute for Nuclear Research
Dubna

The 7th International Conference on
Particle Physics and Astrophysics
Moscow, October 25 2024

Nuclotron based Ion Collider fAcility complex

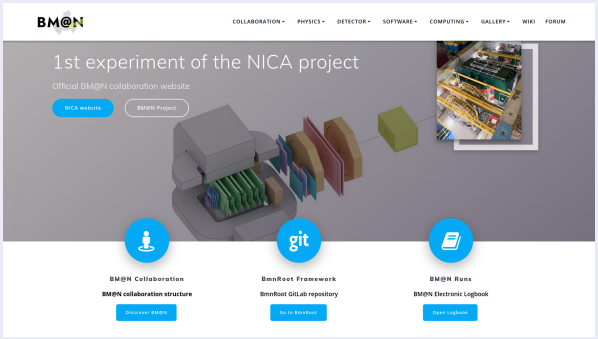


Baryonic Matter at Nuclotron



- Magnet SP-41 (0)
- Vacuum Beam Pipe (1)
- BC1, VC, BC2 (2-4)
- SiBT, SiProf (5, 6)
- Triggers: BD + SiMD (7)
- FSD, GEM (8, 9)
- CSC 1x1 m^2 (10)
- TOF 400 (11)
- DCH (12)
- TOF 700 (13)
- ScWall (14)
- FD (15)
- Small GEM (16)
- CSC 2x1.5 m^2 (17)
- Beam Profilometer (18)
- FQH (19)
- FHCal (20)
- HGN (21)

BM@N Framework BMNROOT



BM@N experiment home web-page:

https://bmn.jinr.ru

- News
- Software repositories
- Software tests
- Forums
- Database for physics run
- E.t.c.

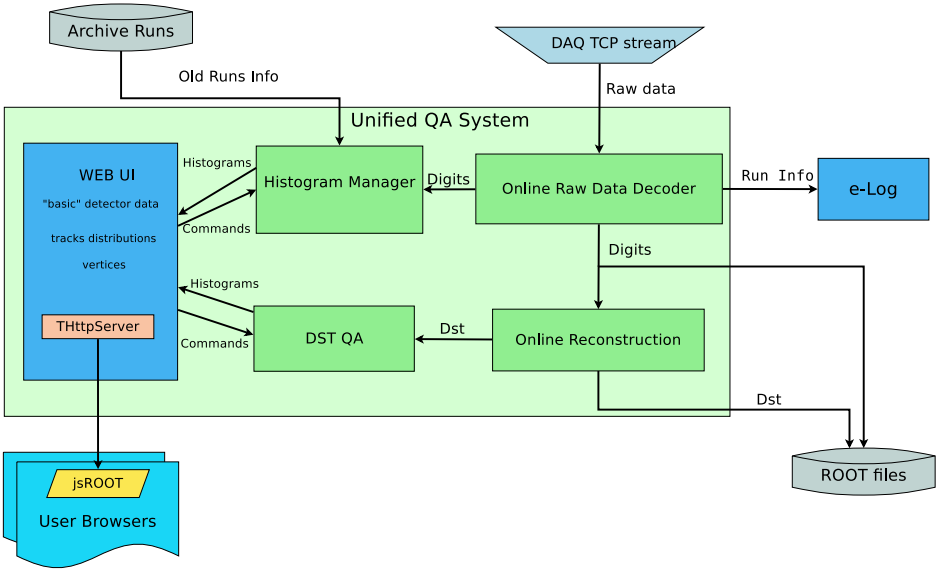
Benefits:

- Inherits basic properties from FairRoot (https://fairroot.gsi.de/), C++ classes
- Detector composition and geometry; particle propagation by GEANT3/4
- Advanced detector response functions, realistic tracking and PID included
- Event display for Monte-Carlo and experimental data
- QA system

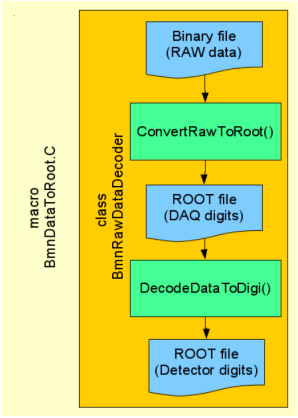
BmnROOT repository

https://git.jinr.ru/nica/bmnroot

General system scheme



Decoding scheme (currently under refactoring)



First step (Data Converter):

- Read a binary data file with RAW-data.
- Parse the data blocks: run/spill/event/module.
- Create «DAQ-digits» (ADC, TDC, TQDC, HRB, TTVXS, etc.) accordingly DAQ-data-format and write them into a tree.

Second step (Data Decoder):

- Read detector mappings (channel-to-strip) from the Unified Database
- Calculate pedestals and common modes of channels
- Clear noisy channels
- Decode DAQ-digits into detector-digits (BmnGemDigit, BmnTofDigit, etc.)
- Write the tree with detector-digits to a ROOT-file

Basic QA frontend with hardcoded histograms

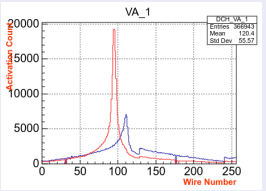
Implementation details:

- The data processed and transferred from the previous stage is used to fill ROOT histograms. Which in turn are sent to the end users via http.
- CERN jsROOT library is used to transform the ROOT object to the html histograms.
- Base class for histogram sets BmnHist is used in:
  - BmnHistTrigger
  - BmnHistGem
  - BmnHistToF
  - ...

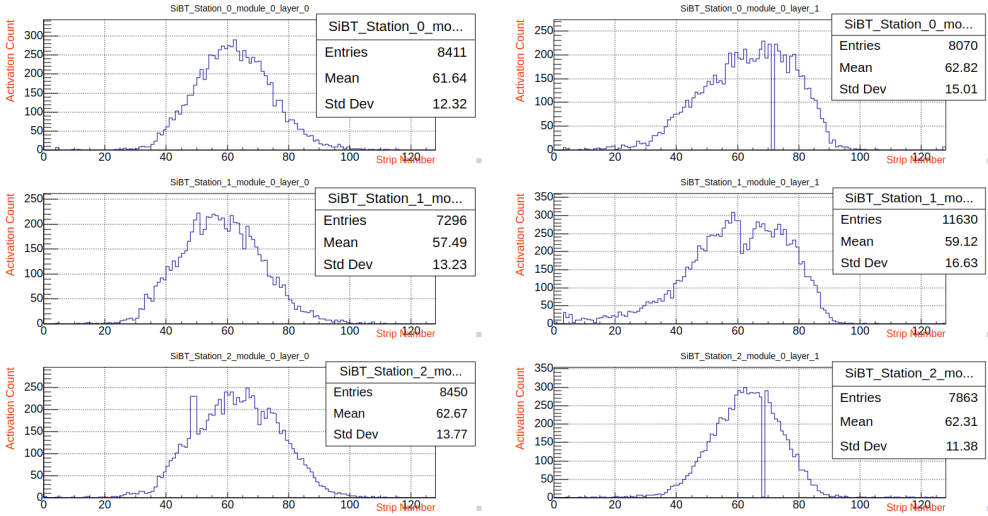
Thus addition of the new detector histogram set is rather simple.

Reference run:

- ✓ Ref run imposition
- ✓ Autoselection of similar runs

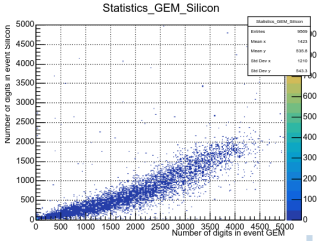


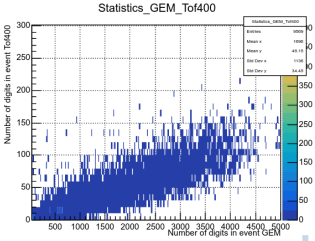
Live example of SiBT digits online decoding

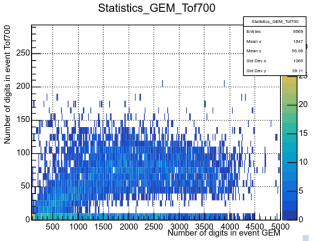


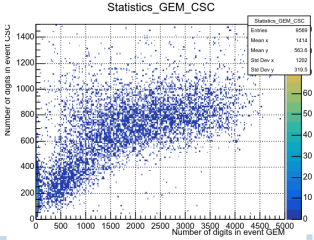
There were some problems with local channel maps at the beginning of the Xe run.

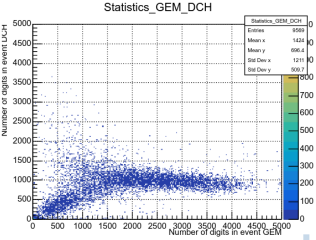
Live example of detectors correlation

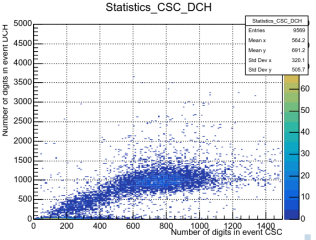












Existing alternative online processing frameworks

- TDAQ (ATLAS)
  - tightly integrated with other ATLAS software
  - thus it is rather difficult to deploy in other program environment
- FairMQ (GSI FAIR) work in progress: I.Romanov, K.Gertsenberger
  - seems to be quite flexible in deployment and settings (with DDS as an option)
  - but requires additional wrapper code
  - seems not to work in an interactive ROOT macros

FairRoot way of analysis via FairTask's
(Extensively being used in the BmnRoot)

- FairRunAna - task manager class
- FairSource - abstract class for a data source
- FairSink - abstract class for a data destination manager

Typical analysis macro workflow:

- BmnFileSource/FairFileSource (input data file)
- Task1 (executed event-by-event)
- Task2
- Task3
- ...
- FairRootFileSink (output data file)

Simplest way to move existing reconstruction code to online

Less code $\rightarrowtail$ Less errors

ZMQ transfer classes for FairRunAna

- BmnMQSource - ZeroMQ SUB socket[1] based source class
- BmnMQSink - ZeroMQ PUB socket based sink class

Benefits

- No need to rewrite existing bmnroot analysis code. (No need to touch any working task)
- It became possible to combine several analysis macros by source/sink network interfaces

[1] https://zeromq.org

BmnRoot QA structure

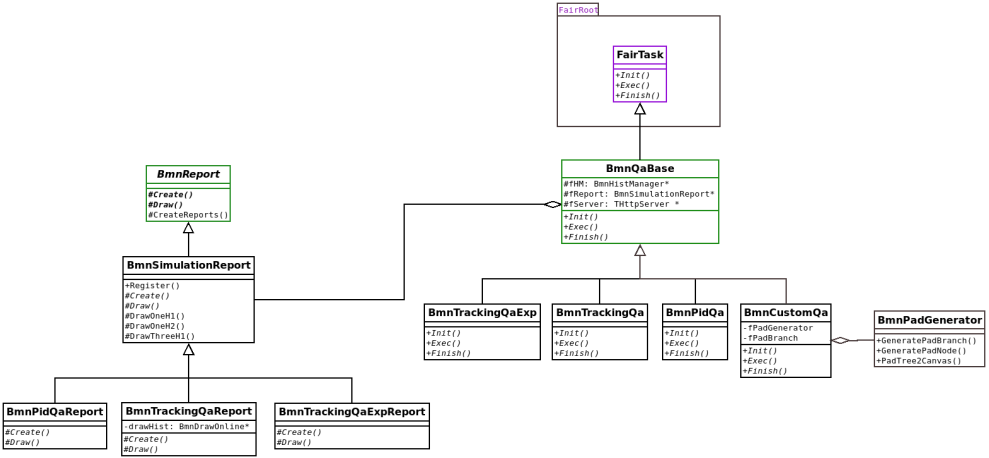
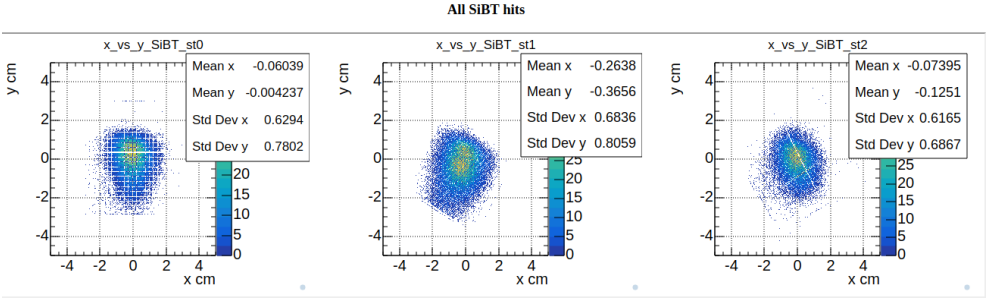
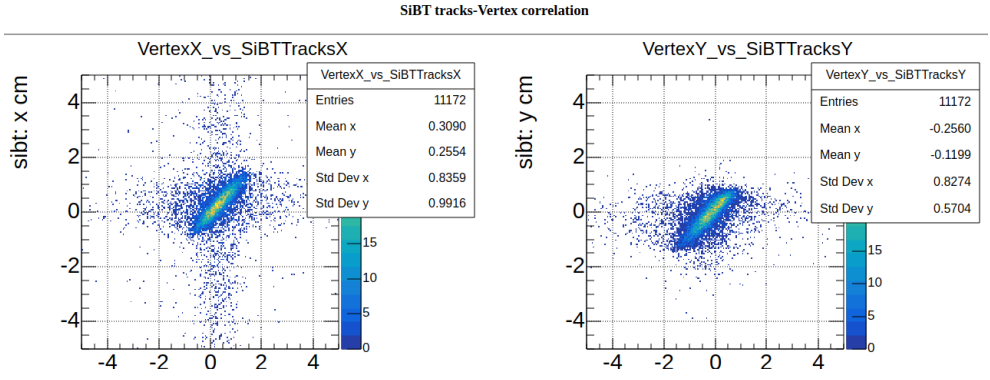


Figure: QA main classes (green ones were forked from CbmRoot)

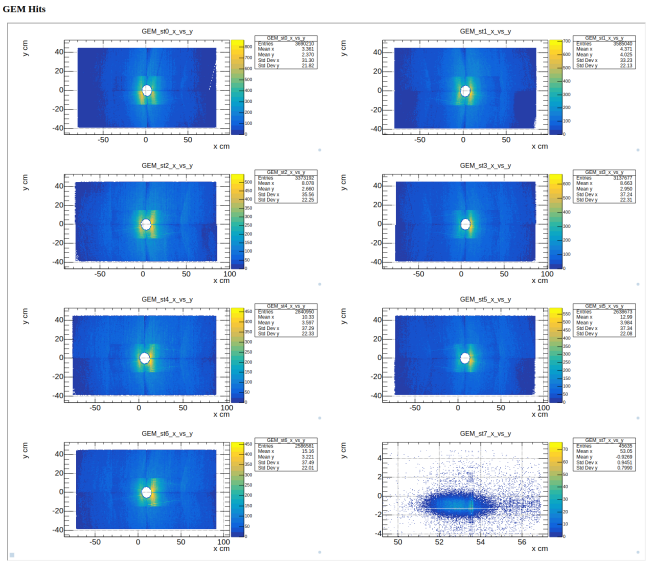
Live example of SiBT hits online reconstruction (Xe+CsI 2023 Run)



Live example of SiBT hits vs Vertex coordinates (Xe+CsI 2023 Run)

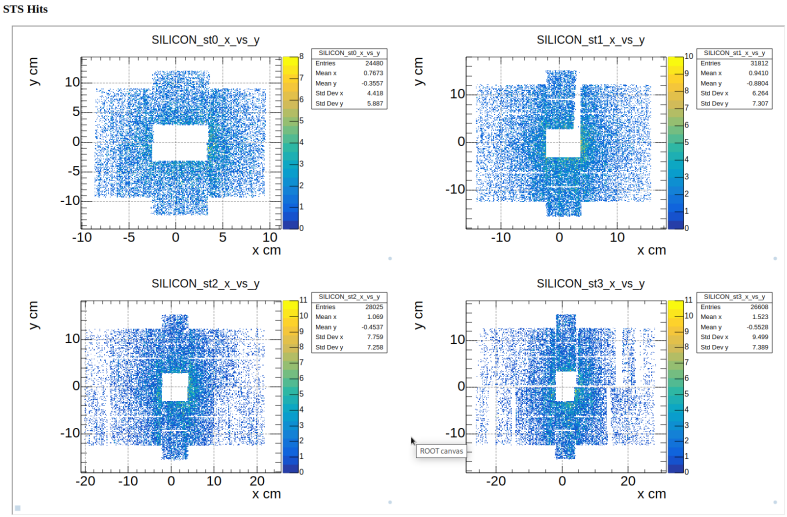


Live example of GEM hits online reconstruction (Xe+CsI 2023 Run)



The noise channel detection logic should be reevaluated for strip detectors on the beam

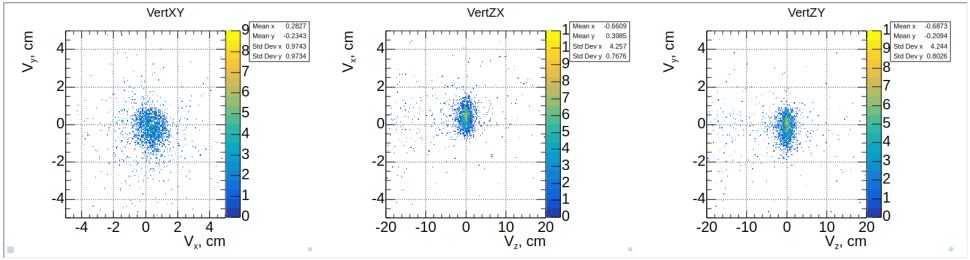
Live example of FSD hits online reconstruction (Xe+CsI 2023 Run)



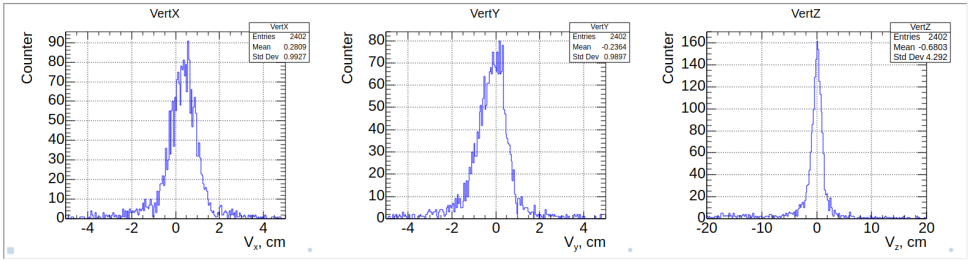
Live example of the primary vertex online reconstruction (Xe+CsI 2023 Run)

Vertex profile

Vertex profile 2D



Vertex profile 1D



Custom «no code» histograms. Motivation

Why?

Experiment upgrade as well as conduction of two experimental setups require distribution of work on the development of the online QA system.
Namely each detector team should be able to extend system's functionality easily.

Custom «no code» histograms. Motivation

Why?

Experiment upgrade as well as conduction of two experimental setups require distribution of work on the development of the online QA system.
Namely each detector team should be able to extend system's functionality easily.

Main objectives:

- Move monitoring configuration outside of the code
- Make addition of histogram simple and flexible (It should not require code rebuild)
- Implement filling logic configurable as well (thanks to ROOT TTree::Draw text parser it was possible)

Custom «no code» histograms. Motivation

Why?

Experiment upgrade as well as conduction of two experimental setups require distribution of work on the development of the online QA system.
Namely each detector team should be able to extend system's functionality easily.

Main objectives:

- Move monitoring configuration outside of the code
- Make addition of histogram simple and flexible (It should not require code rebuild)
- Implement filling logic configurable as well (thanks to ROOT TTree::Draw text parser it was possible)

Implementation

BmnPadGenerator class - creates a pad structure in the canvas on the basis of json scheme.

Test code example:

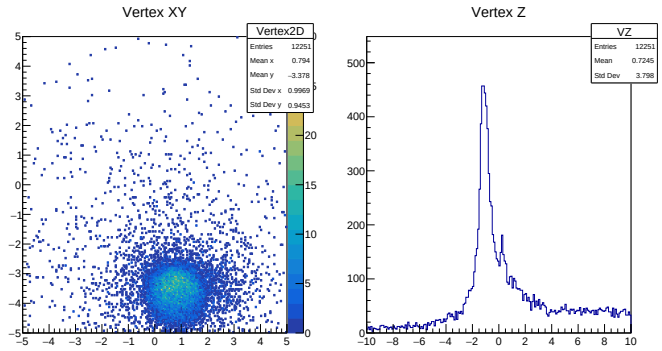
```
BmnPadGenerator *g = new BmnPadGenerator();
g->LoadPTFrom(FileName);
BmnPadBranch * br = g->GetPadBranch();
TCanvas* can = new TCanvas("canHits", "", 1920, 1080);
g->PadTree2Canvas(br, can);
BmnHist::DrawPadTree(br);
```

Simple configuration

JSON scheme:

```
{
    "Name": "Custom canvas",
    "Title": "Custom Canvas",
    "DivX": "2",
    "DivY": "1",
    "Pads": [
        {
            "Class": "TH2F",
            "Name": "Vertex2D",
            "Title": "Vertex XY",
            "Variable": "BmnVertex.fY:BmnVertex.fX",
            "Selection": "(BmnVertex.fZ>-10 && BmnVertex.fZ<10)",
            "Options": "colz",
            "Dimensions": [
                200,
                -5,
                5,
                200,
                -5,
                5
            ]
        },
        {
            "Class": "TH1F",
            "Name": "VZ",
            "Title": "Vertex Z",
            "Variable": "BmnVertex.fZ",
            "Selection": "(BmnVertex.fZ>-10 && BmnVertex.fZ<10)",
            "Dimensions": [
                200,
                -10,
                10
            ]
        }
    ]
}
```

Canvas structure:



Conclusion

- Unified online/offline QA system is being developed
  - Converter/decoder, event reconstruction, histogram management work in separate processes
  - ZeroMQ transfer source/sink classes were developed for FairRunManager based analysis.
  - Experimental "no code" approach were developed in order to simplify extension of the system.
- Decoder future work:
  - Full decoder parallelization
  - Improve logic for calibration/noise automation
- QA system future work:
  - Frontend possible move to Grafana or VisionForge (by MIPT team). CERN jsROOT turned out to be slow and laggy.
  - Automate starting/restarting of a QA components

Conclusion

- Unified online/offline QA system is being developed
  - Converter/decoder, event reconstruction, histogram management work in separate processes
  - ZeroMQ transfer source/sink classes were developed for FairRunManager based analysis.
  - Experimental "no code" approach were developed in order to simplify extension of the system.
- Decoder future work:
  - Full decoder parallelization
  - Improve logic for calibration/noise automation
- QA system future work:
  - Frontend possible move to Grafana or VisionForge (by MIPT team). CERN jsROOT turned out to be slow and laggy.
  - Automate starting/restarting of a QA components
- Thanks to summer students helped with the work:
  - A. Islentev (Edinburgh Un.) - Data converter parallelization, improvements and fixes of the ADC decoder
  - K. Mashitsin (SPbSU) - GEM decoding algorithm improvements and fixes
  - A. Driuk (SPbSU) - Digi correlation histograms, SiBT histogram
  - A. Iufriakova (SPbSU) - ADC decoder SIMD optimisation

Conclusion

- Unified online/offline QA system is being developed
  - Converter/decoder, event reconstruction, histogram management work in separate processes
  - ZeroMQ transfer source/sink classes were developed for FairRunManager based analysis.
  - Experimental "no code" approach were developed in order to simplify extension of the system.
- Decoder future work:
  - Full decoder parallelization
  - Improve logic for calibration/noise automation
- QA system future work:
  - Frontend possible move to Grafana or VisionForge (by MIPT team). CERN jsROOT turned out to be slow and laggy.
  - Automate starting/restarting of a QA components
- Thanks to summer students helped with the work:
  - A. Islentev (Edinburgh Un.) - Data converter parallelization, improvements and fixes of the ADC decoder
  - K. Mashitsin (SPbSU) - GEM decoding algorithm improvements and fixes
  - A. Driuk (SPbSU) - Digi correlation histograms, SiBT histogram
  - A. Iufriakova (SPbSU) - ADC decoder SIMD optimisation

Thanks for your attention!