JOINT INSTITUTE FOR NUCLEAR RESEARCH
Veksler and Baldin laboratory of High Energy Physics

FINAL REPORT ON THE START PROGRAMME

BM@N experiment
Separation of one and two charged particles in Silicon and CSC detectors

Supervisor:
Vasilii Plotnikov

Student:
Roman Leva
Lugansk State University

Participation period:
July 17 – August 31, 2022

Dubna, 2022

1. Introduction

BM@N experiment stands for Barionic Matter At Nuclotron. Investigation of strongly interacting matter at high densities and temperature is one of the main theme of modern high-energy nuclear physics. In this project, we will create programming algorithms for processing and analyzing of the experimental data. The main task is to separate one and two charged particles in Silicon and CSC detectors. Programs are written in C++ with using CERN ROOT and BM@N frameworks.

2. About BM@N installation and terms used in our project

Fixed targets of C, Al, Cu, Sn, Pb are bombarded with argon ion beam produced from the Nuclotron with energy 3.2 GeV per nucleon. After the collision, yield of collision product: Λ hyperons, π^+, K^+, p, He^3, d/He^4, t and others are passing through Silicon and GEM detectors (Figure1), that are used for reconstruction particles trajectory and momentum, that is very important for our analysis.

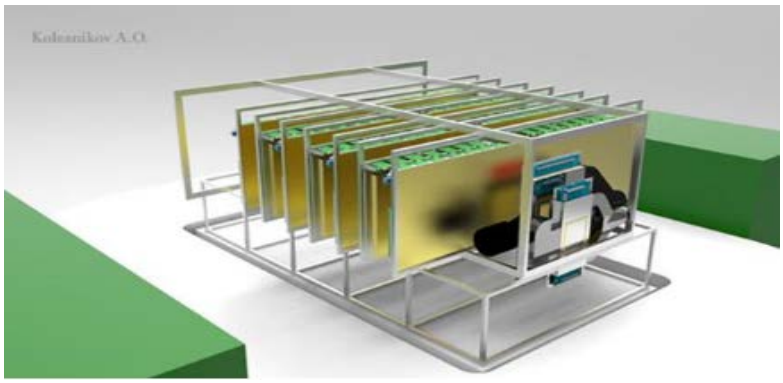

Figure 1. Schematic view of the initial hybrid tracker based on 3 forward silicon planes and 7 GEM planes. The carbon vacuum beam pipe follows the trajectory of the beam.

Figure 2 shows the ToF-400 and ToF-700 systems based on mRPC detectors used for particle time-of-flight measurements. ToFs helps to separate and identify particles. CSC detector approves hit in ToF.

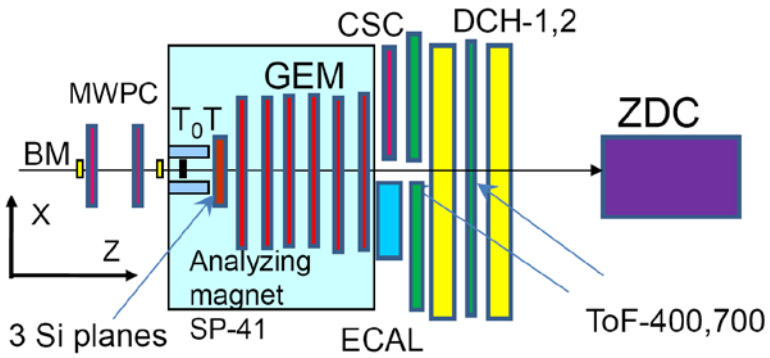

Figure 2. Schematic view of the BM@N setup in the experimental run with the argon and krypton beams.

After the collision, yield of nuclear fragments passes through different detector planes such as GEM, Silicon or Cathode Strip Chamber. Each particle can produce a Hit. A signal of

each detector strip called Digit. Particle can excite more than one detector strip. Hit consists from two overlapping Clusters. One Cluster belongs to the front readout plane. The other Cluster belongs to the back readout plane. Some hits can be fakes. Collection of Hits is processed by different algorithms to make a collection of particle Tracks.

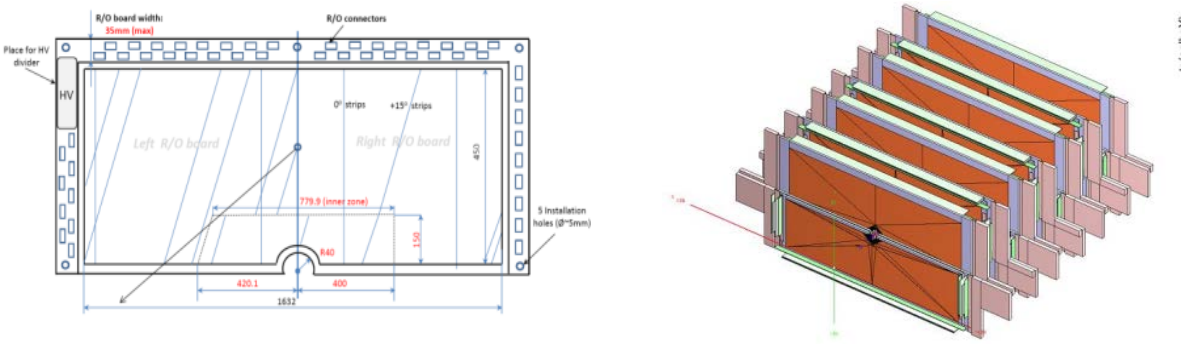

Figure 3. GEM detectors strips (left) and full GEM installation (right). Now only upper planes are installed and used.

Central tracker system consist of GEM and Silicon. GEM (Figure 3) is used for track reconstruction. GEM is also used for momentum measurements as they are placed inside the magnet.
Silicon detectors (Figure 4) are installed between the target and the GEM stations, they participates in track reconstruction and improve the precision of the primary vertex reconstruction.

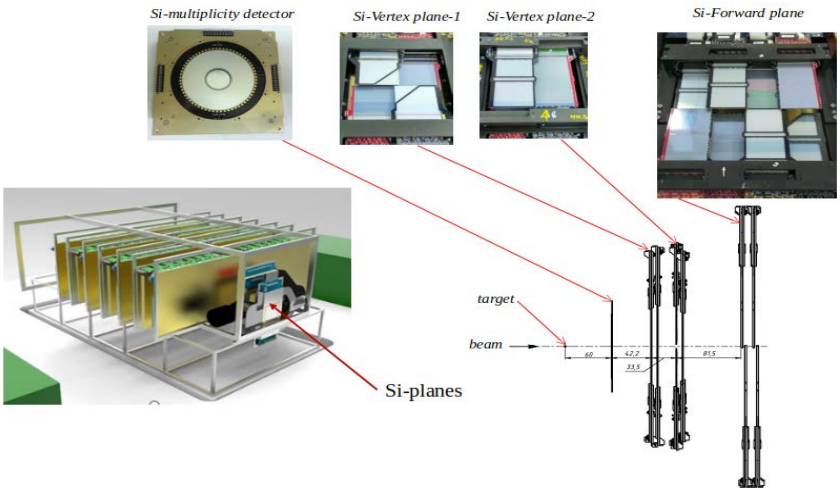

Figure 4. Silicon detectors.

Outer tracker is used to provide link between tracks measured in the central tracker and hits in the ToF-400 and ToF-700 detectors. It consisted of two large aperture drift chambers and Cathode Strip Chamber (Figure 5). CSC is used in our analysis.

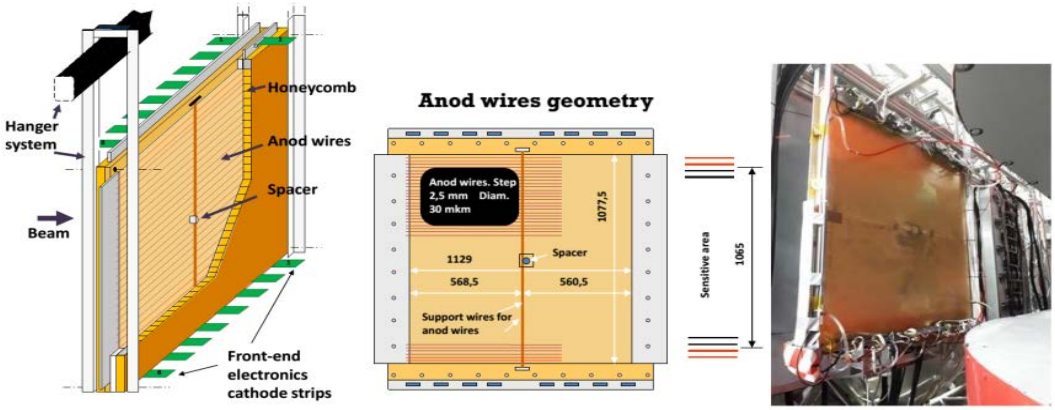

Figure 5. Cathode Strip Chamber (CSC) detectors.

Tracks of charged particles were reconstructed in the Si/GEM central tracking system and extrapolated into the CSC, where special algorithms matches CSC Hits with extrapolated tracks.

ToF time measurements are used to separate particles. From the formula (from the right) seen, that after measurement particle momentum, flight distance and time-of-flight we can obtain mass of the particle.

$$\tau = \frac{L}{\beta c};\ \beta = \frac{p}{\sqrt{p^2 + m^2}}$$

dE/dx information is available to us through the amplitudes of the hit signals. In Figure 6-A can be seen specific losses for different particles. This is used in dE/dX method.

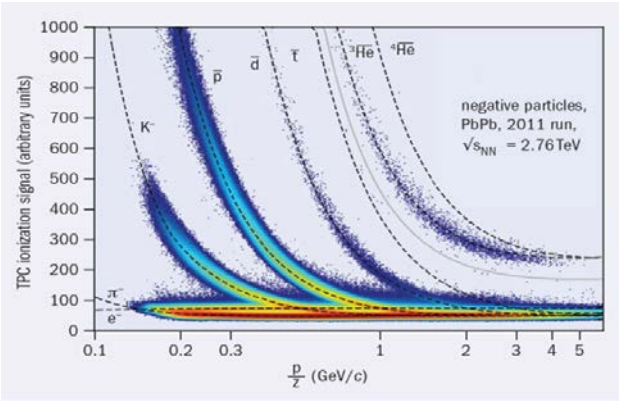

Figure 6-A. Ionization signals as function of particle momentum.

Below in Figure 6-B is an example of particle separations using time-of-flight method.

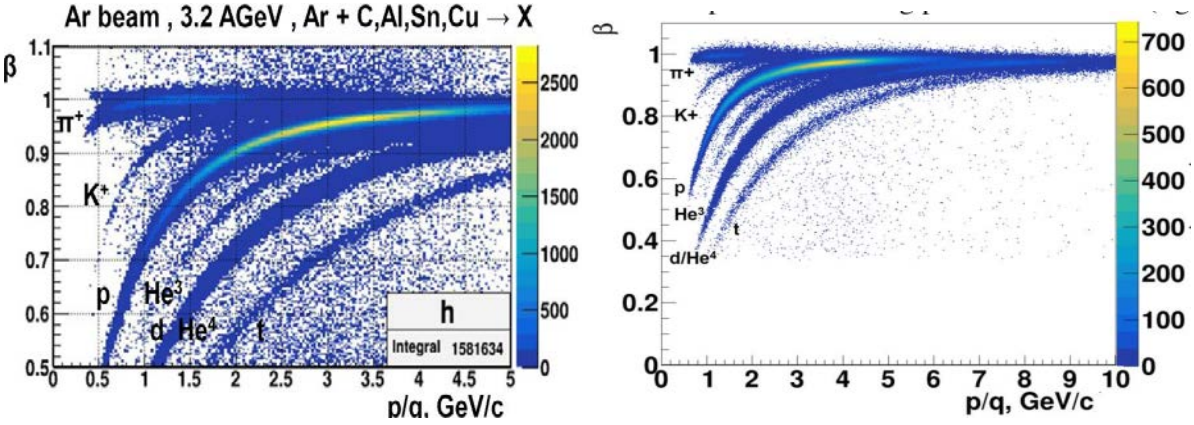

Figure 6-B. Velocity β = v/c as a function of the momentum-to-charge ratio p/q of positive charged particles measured in the ToF-400 system (left panel) and ToF-700 (right panel). Identified π + , K + , p, He 3 , d/He 4 , t are visible as populated bands of particles.

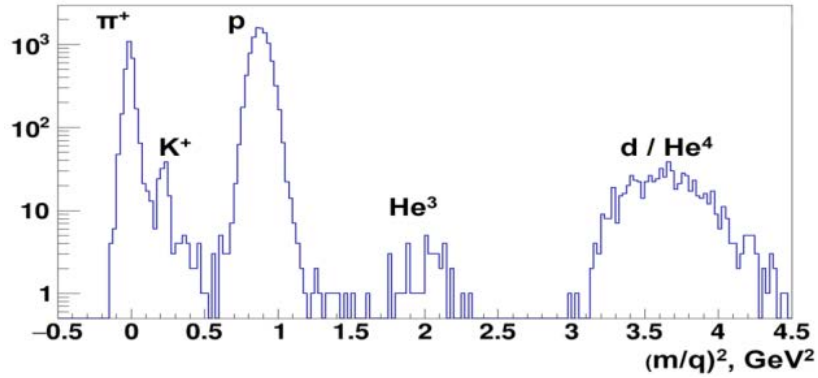

Figure 7. Distribution of reconstructed particle mass to charge squared.

From Figure 6 and 7, we can see that we need more precise separation, especially in higher momentum. For a more detailed separation of particles, information about the amplitudes of hits in GEM detectors is already used. In the present work, we analyze the possibility of using information about the amplitudes of hits in silicon detectors and CSCs in addition to information from GEM detectors.

3. Task description and solving.

As input data to our analysis, we have collections of: tracks (identified with help of Tof and not identified), hits, digits and clusters, primary vertexes. We need to compose identified and unidentified tracks. For each Event, we have identified Track loop (identified with ToF). For each such an identified track, we need to seek for this track in collection of unidentified tracks.

For every Event, we compare equality by number of hits in ident and unidentified track, and for every track pair we compare hit index arrays as in piece of code below:

```
Int_t nhitsCsc = trCsc->GetNStsHits();
for (Int_t it = 0; it < stsTrack->GetEntriesFast(); ++it)
{
  CbmStsTrack *tr = (CbmStsTrack *)stsTrack->UncheckedAt(it);
  if (tr->GetNStsHits() != nhitsCsc)
    continue;
  Int_t n = 0;
  while (n < nhitsCsc && tr->GetStsHitIndex(n) == trCsc->GetStsHitIndex(n))
    n++;
  if (n != nhitsCsc)
    continue;
```

Next, we impose constraints on vertexes from vertex collection such as in code below:

```
// vertex contition
if (primVtx->GetNTracks() < 2 && fabs(VertPos[2]) > 5.0)
  continue;

Bool_t trackIsInVertexTracksVec = kFALSE;
for (Int_t i = 0; i < (int)primVtx->trkID.size(); i++)
{
  if (primVtx->trkID[i] == trPid->GetTrack()->GetTrkID())
  {
    trackIsInVertexTracksVec = kTRUE;
  }
}
if (!trackIsInVertexTracksVec)
  continue;

Double_t maxZpv = 6.; // or 10.
if (!((primVtx->GetZ() > 0.584 - maxZpv /*1.5*0.65*/) && (primVtx->GetZ() < 0.584 + maxZpv /*1.5*0.65*/)) ||
    !((primVtx->GetX() > 0.28 - 2.5 * 0.93) && (primVtx->GetX() < 0.28 + 2.5 * 0.93)) ||
    !((primVtx->GetY() > 2.46 - 2.5 * 0.97) && (primVtx->GetY() < 2.46 + 2.5 * 0.97)))
  continue;

if ((trPid->GetXpv() < primVtx->GetX() - 1) && (trPid->GetXpv() > primVtx->GetX() + 1) &&
    (trPid->GetYpv() < primVtx->GetY() - 1) && (trPid->GetYpv() > primVtx->GetY() + 1) &&
    (trPid->GetDx() < -TMath::Sqrt(trPid->GetDdx())) && (trPid->GetDx() > TMath::Sqrt(trPid->GetDdx())) &&
    (trPid->GetDy() < -TMath::Sqrt(trPid->GetDdy())) && (trPid->GetDy() > TMath::Sqrt(trPid->GetDdy())))
  continue;
// end vertex condition
```

Meaning of such constraints is that vertex must have minimum 3 tracks and Z<5 cm. Vertex need to fall into certain spatial region conditioned by distribution area of X,Y,Z + 2.5 sigma of the on-target Argon beam. Distribution by Z is much wider, because of low accuracy of vertex finding algorithms.
Next we use Kalman filter to extrapolate our track to vertex:

```
CbmKFTrack kfTr = CbmKFTrack(*tr);

FairTrackParam trParam;
Int_t kfStatus = kfTr.Extrapolate(primVtx->GetZ()); // 0) success; 1) fault (10%)
kfTr.GetTrackParam(trParam);

double dx = trParam.GetX() - primVtx->GetX(), dy = trParam.GetY() - primVtx->GetY();

hTrdXYVert->Fill(dx, dy);

Double_t trBimp = TMath::Sqrt(dx * dx + dy * dy);
```

Our extrapolated track need to deviate from vertex less than 1 cm, so variable *trBimp<1* 1 cm is the accuracy of vertex finding algorithm.

Kalman filter helps to smooth the mean of track points (*measurements line*) array as in Figure 8 from the right:

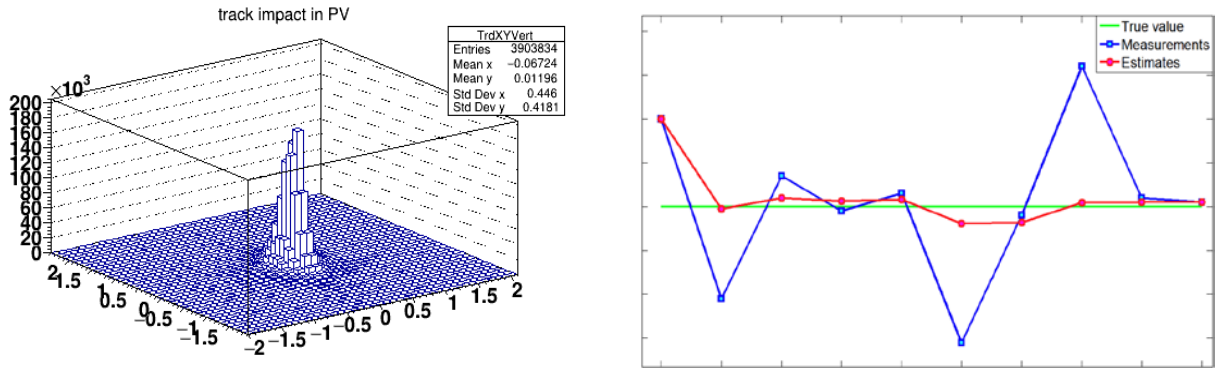

Figure 8. Residuals in Tof-400 (left), Kalman filter (right).

Histogram hTrdXYVert (Figure 8 from the left) describes distances between event primary vertex and track extrapolated to the vertex Z coordinate. The last two conditions in the restricted listing correspond to the experimental momentum dependencies of the residuals between the track and the ToF-400 hit.

Next, we evaluating each track many algorithms where we find track slop, segment length, selecting hits and clusters, doing truncated mean dE / dX calculation.

Therefore, we have found good identified track. This constraint helps for more clear analysis.

For each of that track, we have measured the mass of the particle. This mass will be compared to the table masses as in example of code below:

```
CbmStsTrack *tr = (CbmStsTrack *)stsTrack->UncheckedAt(indStsTrack);

if (fabs(tr->GetParamLast()->GetQp()) < 0.005)
  continue; // some direct track
//double m2= tr->GetB();
double m2 = trPid->GetMassQ();
int ton = 0;
if (m2 != 0)
  ton = 1;       // TOF700 association with m2 value
int tofPid = 0; // none
if (m2 > 0.1 && m2 < 0.35)
  tofPid = 1; // pion
else if (m2 > 0.5 && m2 < 1.2)
  tofPid = 2; // proton
else if (m2 > 1.2 && m2 < 2.5)
  tofPid = 3; // 3He
//else if(m2>2.5 && m2<4.5) tofPid=4; // d/He
else if (m2 > 2.5 && m2 < 7.5)
  tofPid = 4; // d/He
```

We use hits which section number corresponds to Silicon station (for Silicon) and Z>400 cm (for CSC). After processing all the tracks and its hits in Sil and CSC, we got a couple of histograms.

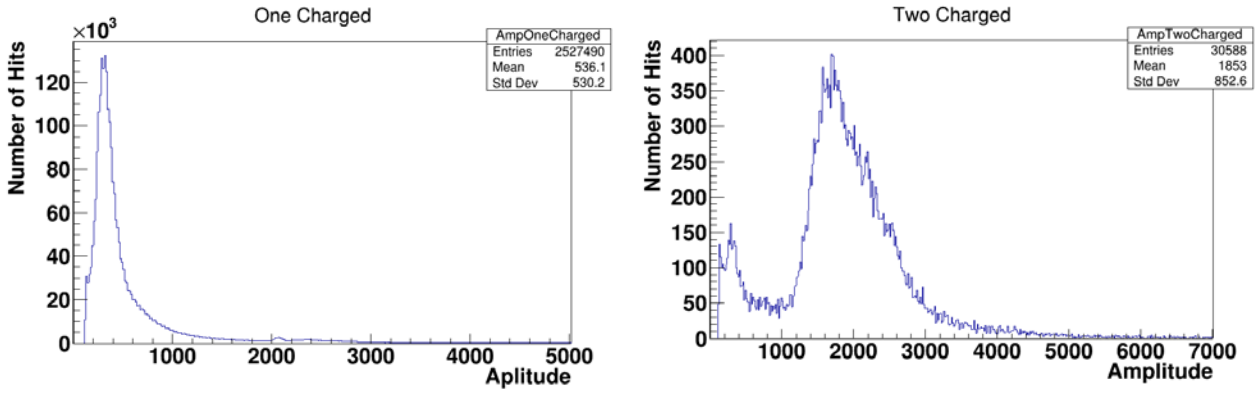

Figure 9. Integral histograms of one-charged (left) and two-charged (right) particle amplitudes in Silicon detectors by all stations.

In Figure 9 from the left, it is clearly seen a very good resolution of one-charged particles in Silicon. Peak value is near 400. There is no signal below 100 because of filtering of electronics noise. Also, a little overflow peak is seen near 2048 caused by ADC channel overflow and presence of more than one digit in a cluster.

In Figure 9, the leftmost very small peak is a noise from electronics that passes through a threshold. The Plato between one and two-charged peaks caused by broken clusters, where signal is observed not so clearly, is also seen. Also seen purity between left and right histograms for one and two-charged particles. This caused by the fact, that there is more cluster strips for two-charged and more width of each cluster respectively.

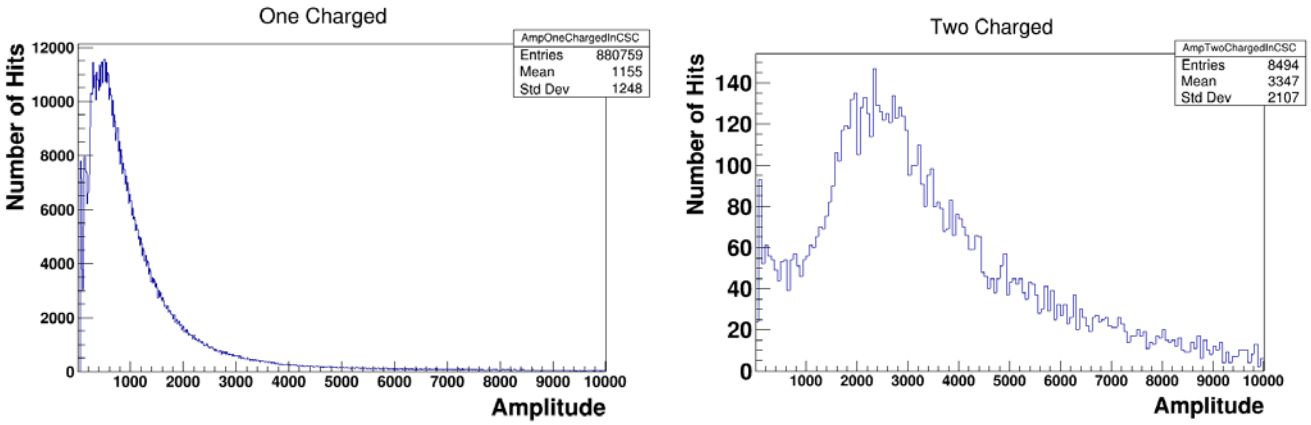

Figure 10. Integral histograms of one-charged (left) and two-charged (right) particle amplitudes in CSC detectors by all sectors.

In Figure 10 for CSC detector can be seen clear enough peak of one-charged tracks (without overflow peak) and two-charged tracks.

Next, we want to show distributions by individual sectors to better understand the spectra of signals amplitudes on them.

In comparing one-charged peak in Silicon and CSC, it can be seen that in CSC peak is not so clear, because of about five strips in CSC cluster that is more than in Silicon cluster. For two-charged clusters, there are about 10 strips. It causes peak widening for CSC.
Next, we want to get more clear statistics. We analyzed the number of hits in detector stations and sectors. In Figure 11, it is seen that the second station with first sector is the most populated at Silicon detectors.

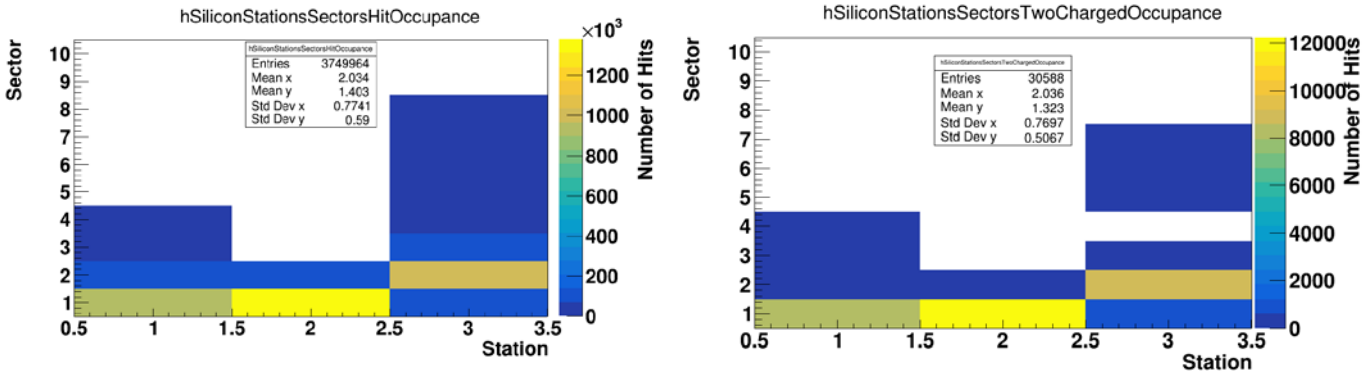

Figure 11. Histograms of one-charged (left) and two-charged (right) particle population in Silicon detector by station and sector number.

The most populated sector at CSC is the third sector. There are only a few two-charged hits in the 1st sector. That was due to the fact that the experiment used only upper part of GEM installation that intersects with CSC only partially. In Figures 12 and 13, the amplitude of two-charged and one-charged tracks in CSC is shown by sectors.

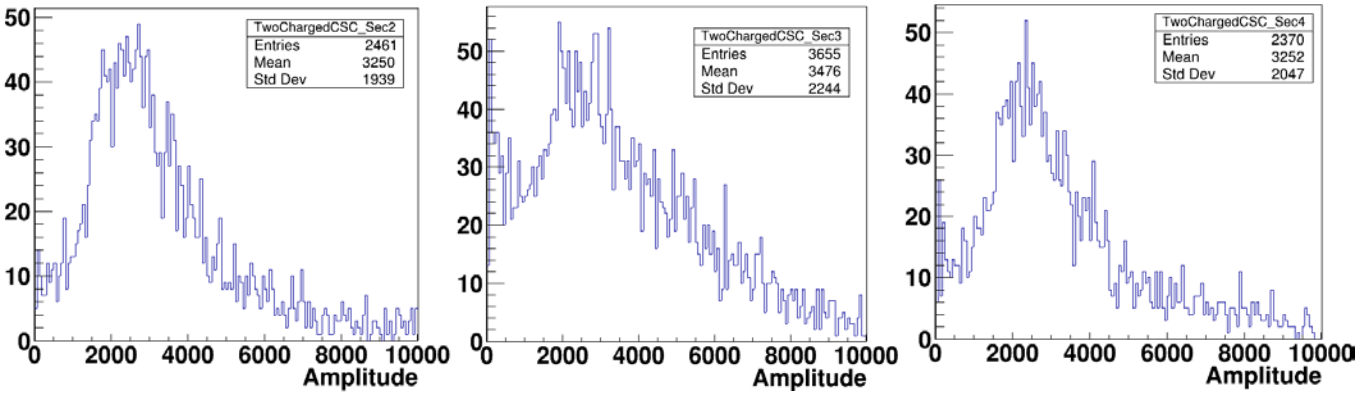

*Figure 12. Histograms of **two**-charged particle population in CSC detector by sector*

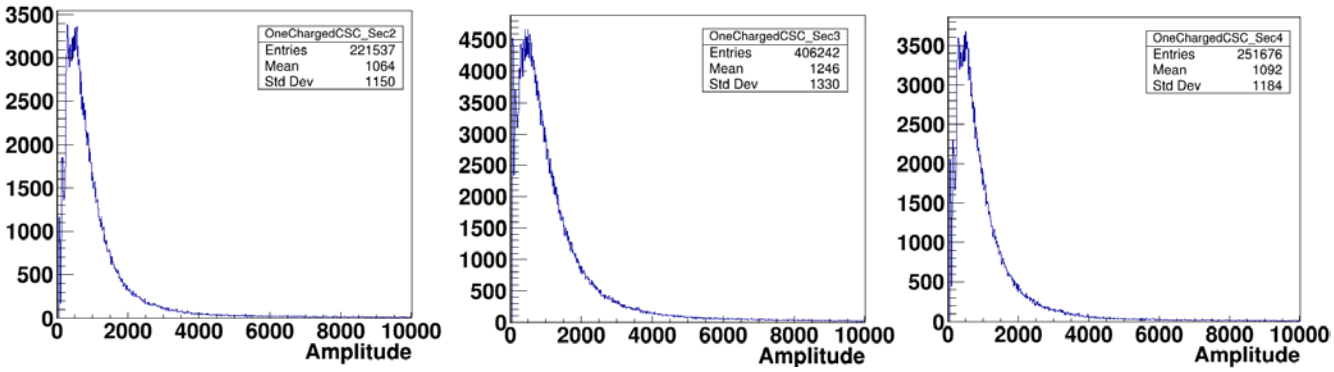

*Figure 13. Histograms of **one**-charged particle population in CSC detector by sector*

The amplitudes of one and two charged tracks in the most populated Silicon sector is shown in Figure 14.

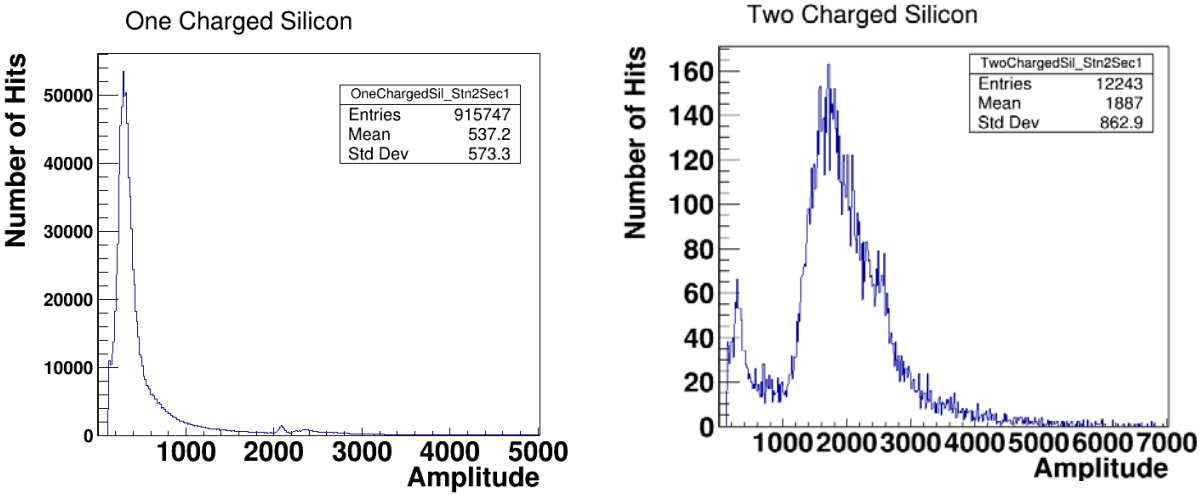

Figure 14. Histograms of one-charged (left) and two-charged (right) particle amplitudes in Silicon 2-nd station with 1-st sector.

We can see quite the same peak purity in sectors as in integral data.

4. Calculation of the ratio of one-charged to two-charged for Silicon detectors and CSC for the most populated sector

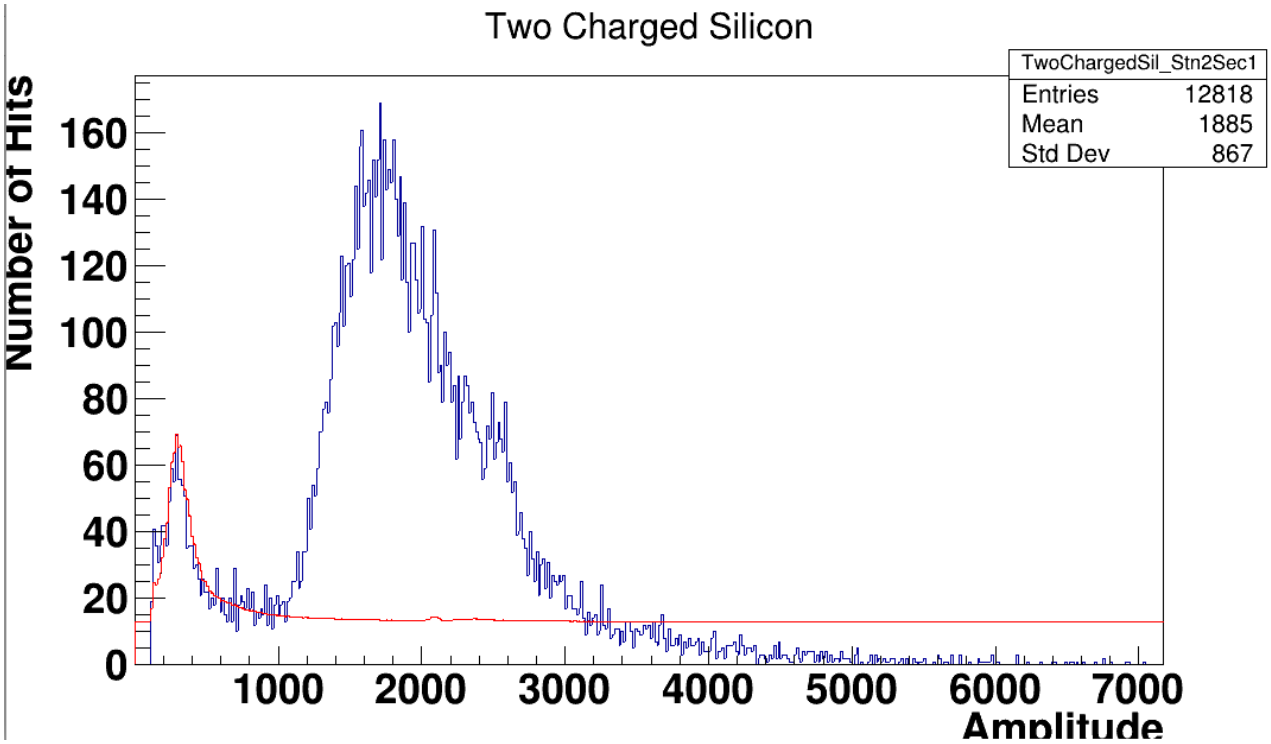

Figure 15. Decomposition of the cluster amplitude spectrum in a Silicon detector for two-charged tracks into signals from different Sources.

In Figure 15, we combine scaled left and right histograms from Figure 14. We also add a constant background whose value we extracted from the Plato between the two peaks. We can see a good fit for the left peak in Figure 15, which confirms our assumption about its origin.

```
root [1] TF1 *fu2 = new TF1("fu","13",0,10000)
root [2] OneChargedSil_Stn2Sec1->Scale(0.001)
root [3] OneChargedSil_Stn2Sec1->Add(fu2)
root [4] TwoChargedSil_Stn2Sec1->Draw()
root [5] OneChargedSil_Stn2Sec1→Draw("same")
```

Now we will look for the ratio of peak squares of two-charged and one-charged particles in the Silicon detector and CSC.

For one-charged in Silicon from c in the left (Ampl 100 to 1000), the amount under the peak = 880000.
For two-charged in Silicon from Figure 14 in the right (Ampl 1000 to 3200), the amount under the peak (excepting one-charged peak) 12100 – 1600 = 10500
Result 10500 / 880000 = 0.012 is the ratio of peak squares of two-charged and one-charged particles in Silicon.

For one-charged in CSC from Figure 13 in the middle (Ampl 50 to 4000), the amount under the peak = 400000.
For two-charged in CSC from Figure 12 in the middle (Ampl 200 to 9000), the amount under the peak (excepting noise peak) 3500 – 250 = 3250
Result 3250 / 400000 = 0.008 is the ratio of peak squares of two-charged and one-charged particles in CSC.

On Figure 15, the one and two charged tracks, and the tracks with bad clusters are seen.

Lets look at Figure 15:
S = Square of all (Ampl from 0 to 3200) = 12000
Sp1 = Square of all substrate (Ampl from 0 to 3200) = 2500
Sp2 = Square of substrate (Ampl from 1100 to 3200) = 1750
Sp3 = Square of substrate (Ampl from 0 to 600) = 440
Stwo = Square of two-charged above substrate (Ampl from 1100 to 3200) = 8000
Sone = Square of one-charged above substrate (Ampl from 0 to 600) = 700

Sp1 / S = 0.20
Stwo / S = 0.66
Sone / S = 0.05

Sp1/S is the part of the distribution associated with the background. Sone/S is the part of distribution relevant to one-charged tracks. Stwo/S is the part of the distribution corresponding to the two-charged tracks.

Values 0.08 and 0.012 are close enough that is expected.

5. Speedup algorithms for Run processing

For each Event Id, we have collection of identified tracks. We need to seek them in collection of unidentified tracks. Therefore, we can accelerate seeking by creating a Tree object, where for each Event Id we will store track index from the array of unidentified tracks.

```
TTree *eventIdToRecId = (TTree *)out->Get("eventIdToRecId");
if (!eventIdToRecId)
{
  eventIdToRecId = new TTree("eventIdToRecId", "Indexing eventId to recId");
}
```

When processing a Run, if it is calculated for the first time, then a new Branch *recId* of type unsigned int is created, the value of the *recIdInput* variable will be written to the Branch when the Fill method is called. If the second time, then when you call *GetEntry*, the *recIdOutput* variable will be filled with the value from the Branch.

```
UInt_t recIdInput, recIdOutput;
if (eventIdToRecId->GetEntries() == 0)
{
  eventIdToRecId->Branch("recId", &recIdInput, "recId/i");
  creatingEventIndexTree = true;
}
else
{
  eventIdToRecId->SetBranchAddress("recId", &recIdOutput);
  creatingEventIndexTree = false;
}
```

All Events have a sequential index 0,1,2,3…. Therefore, a direct comparison is obtained: by the number of the entry in the tree (coincides with the number of the event), the found index for the array of unidentified tracks is stored.

```
if (creatingEventIndexTree)
{
  do
  {
    rec->GetEntry(RecoEventNumber++);
    UInt_t bmnEventNumber = ((BmnEventHeader *)BmnH->UncheckedAt(0))->GetEventId();
  } while (((BmnEventHeader *)BmnH->UncheckedAt(0))->GetEventId() < PidEventNumber);

  recIdInput = RecoEventNumber - 1;
  eventIdToRecId->Fill();
}
else
{
  eventIdToRecId->GetEntry(iev);
  rec->GetEntry(recIdOutput);
  UInt_t bmnEventNumber = ((BmnEventHeader *)BmnH->UncheckedAt(0))->GetEventId();
}
```

```
  if (creatingEventIndexTree)
  {
    eventIdToRecId->Write();
  }

  out->Write();
  out->Close();
```

The result of the measurement of the processing of the Run file:
- the first time the file is processed in 60 seconds
- the second time in 45 seconds!

When you increase the file with tracks, the difference in speed will begin to appear in proportion to the number of tracks.

A similar Tree was also created to store the unidentified track index matched by *HitIndex*. In this tree, we will create a double index from the Event Id and at the same time from the number of the identified track. But in this situation, since the arrays being sorted are not large, there was no noticeable acceleration.

```
  for (Int_t it = 0; it < stsTrack->GetEntriesFast(); ++it)
  {
    CbmStsTrack *tr = (CbmStsTrack *)stsTrack->UncheckedAt(it);
    if (tr->GetNStsHits() != nhitsCsc)
      continue;
    Int_t n = 0;
    while (n < nhitsCsc && tr->GetStsHitIndex(n) == trCsc->GetStsHitIndex(n))
      n++;
    if (n != nhitsCsc)
      continue;

    indStsTrack = it;

    eventIdInput = iev;
    pidTrackIdInput = itPid;
    trackIdInput = it;

    break;
  }
  if (indStsTrack == -1)
  {
    cout << "ERROR: associated STS tracks is not found!" << endl;
  }

  pidTrackIdToUnindTrackId->Fill();
}
else
{
  pidTrackIdToUnindTrackId->GetEntryWithIndex(iev, itPid);

  indStsTrack = trackIdOutput;
}

CbmStsTrack *tr = (CbmStsTrack *)stsTrack->UncheckedAt(indStsTrack);
```

```
  if (creatingTrackIndexTree)
  {
    pidTrackIdToUnindTrackId->BuildIndex("eventId", "pidTrackId");
    pidTrackIdToUnindTrackId->Write();
  }
```

6. Conclusion

Silicon detectors can be used to gain additional information for separation of particles. The problem in using CSC consists in a significant intersection of the peaks for one charged and two charged particles, which is clearly seen in Figure 10.

7. Acknowledgments

I deeply grateful for my mentor Vasilii Plotnikov, he helps me understand the project with a lot of lectures and answers on my questions. Vasily showed me the BM@N installation and tells a lot of how it works. Great thanks for JINR organizers that help me to travel to beautiful city Dubna and gave comfortable hotel, gave me participation in excursions on JINR main facilities.