Status on TRGECL DQM

HanEol Cho
Hanyang University
TRG meeting

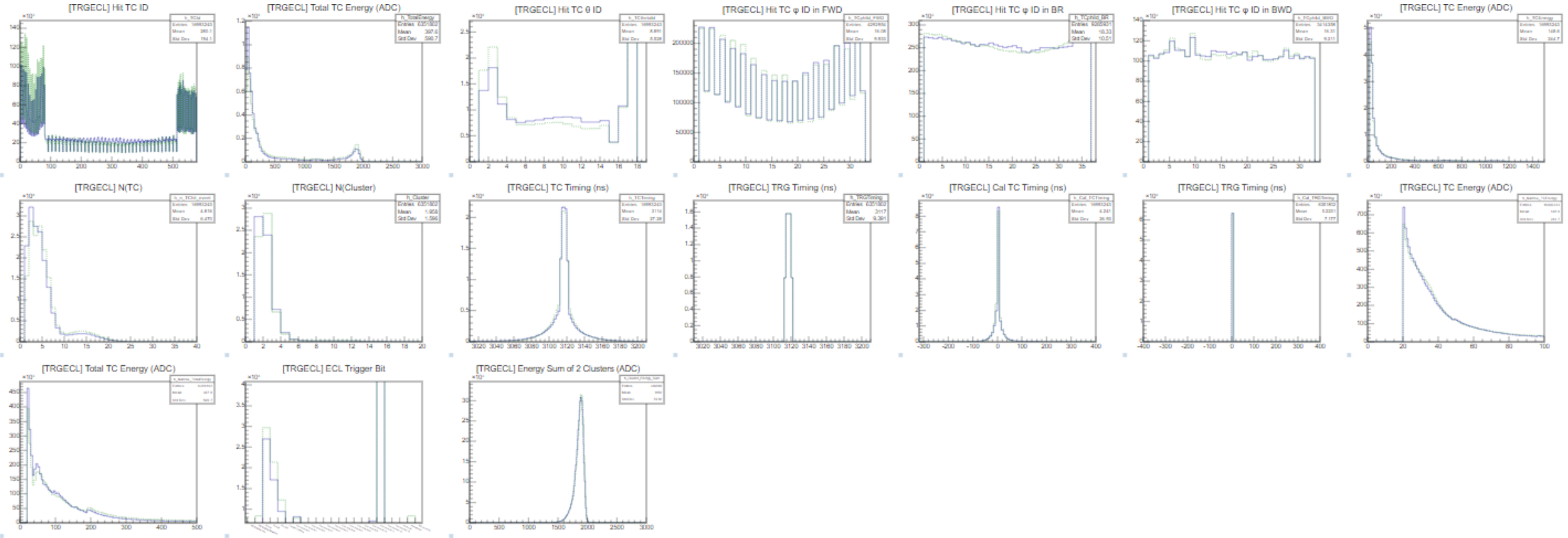
Seoul, 23.03.16

Status on TRGECL DQM

- TRGECL DQM
  - Current status
  - Fine Event
    - ► Fine event definition
    - ► Fraction of Fine Event plot
  - Event T0
    - ► The kind of Event T0
    - ► Event T0 plot
    - ► Histogram Analysis Event T0 resolution [on going]
- To do
  - QAM
    - ► QAM for Event T0 resolution
  - Fraction of ECL TRG timing
    - ► Other exp
    - ► The Fraction as a function of statistic

TRGECL DQM

Current status



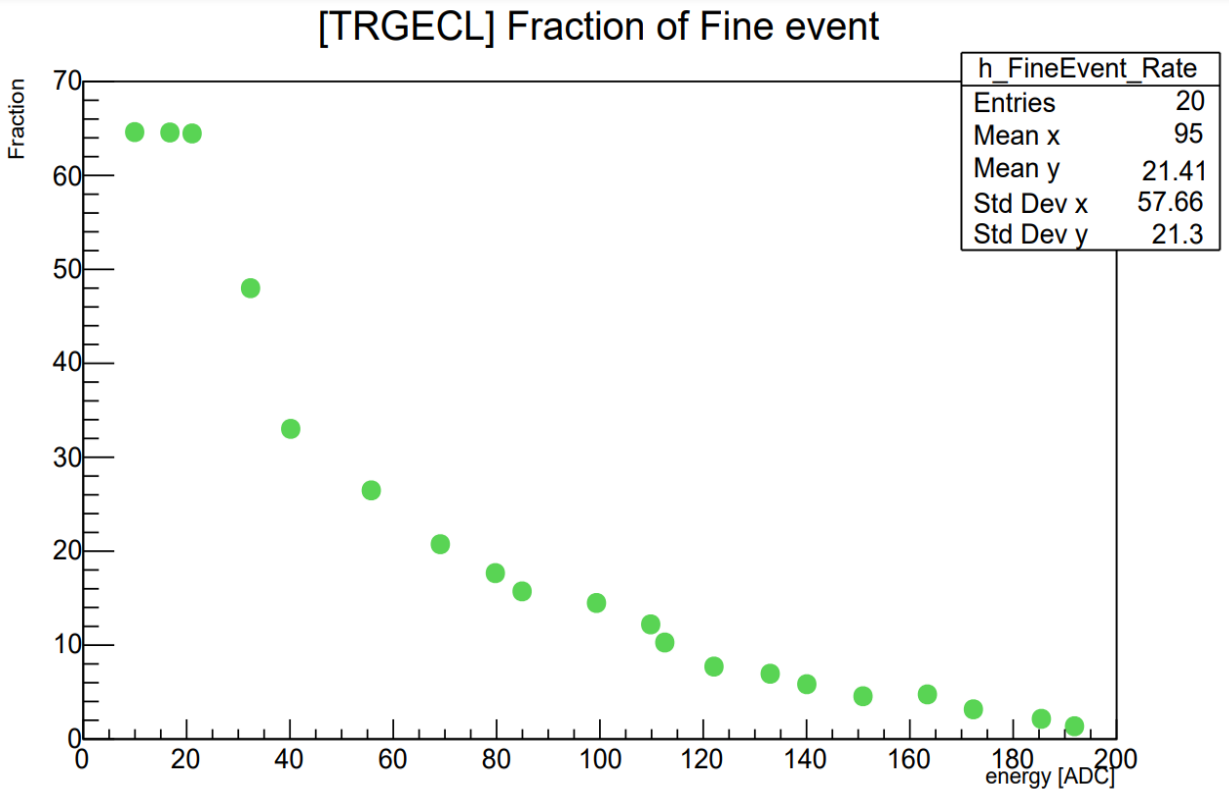
Fine Event

The ECL trigger is the main L1 trigger timing source. The ECL trigger determines trigger timing as the timing of the highest energy TC with two data clocks(256ns).

- Definition
  - The event whose most energetic TC is larger than 200 ADC (Not fixed).
  - SVD group will be select the condition.

- Applications
  - 6/3 mixed sample mode of SVD.
    - ► 3-sample based on fine event ($E_{max} > 200\ ADC$)
    - ► 6-sample based on course event ($E_{max} < 200\ ADC$)

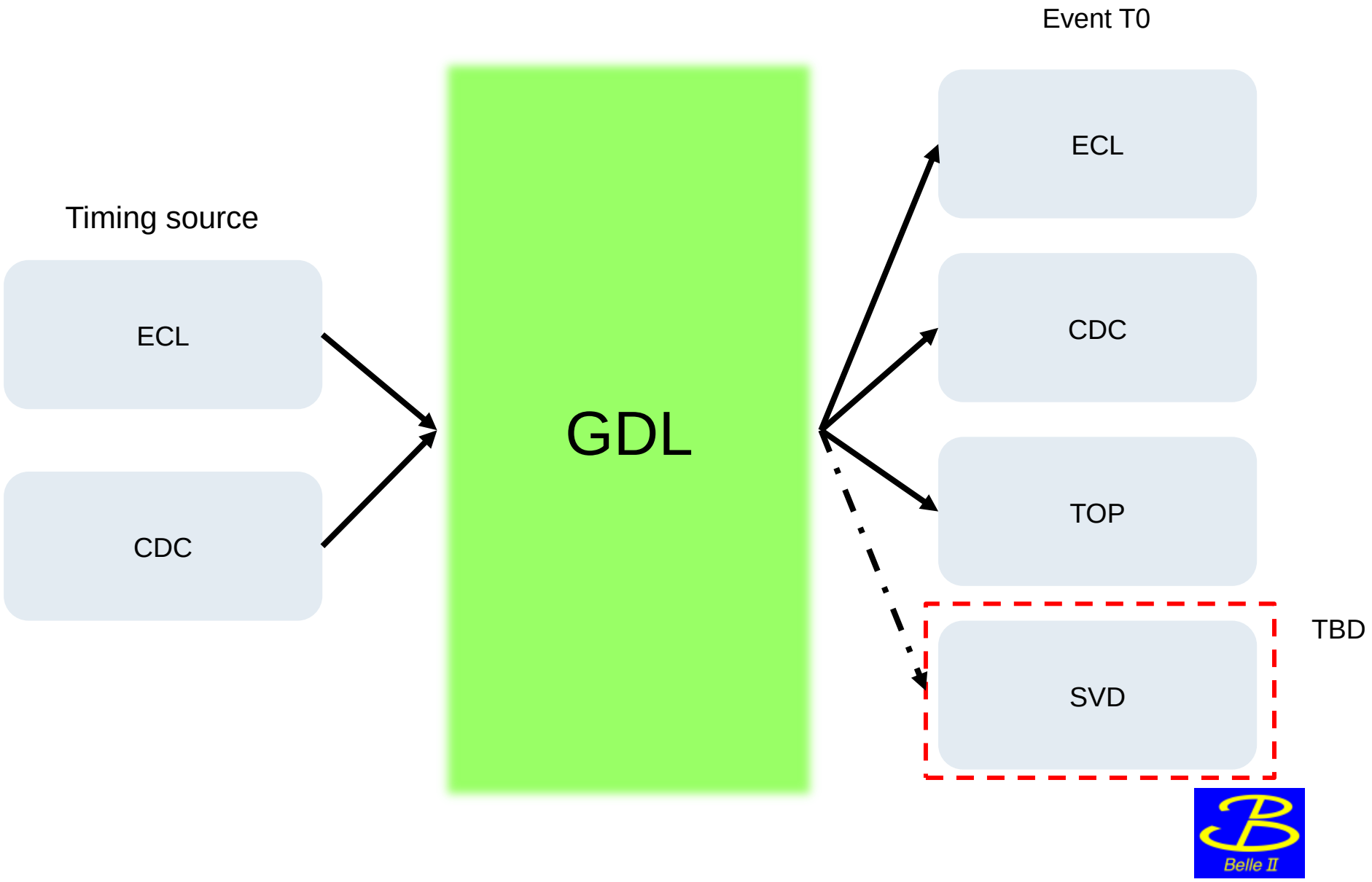
Fraction of Fine Event



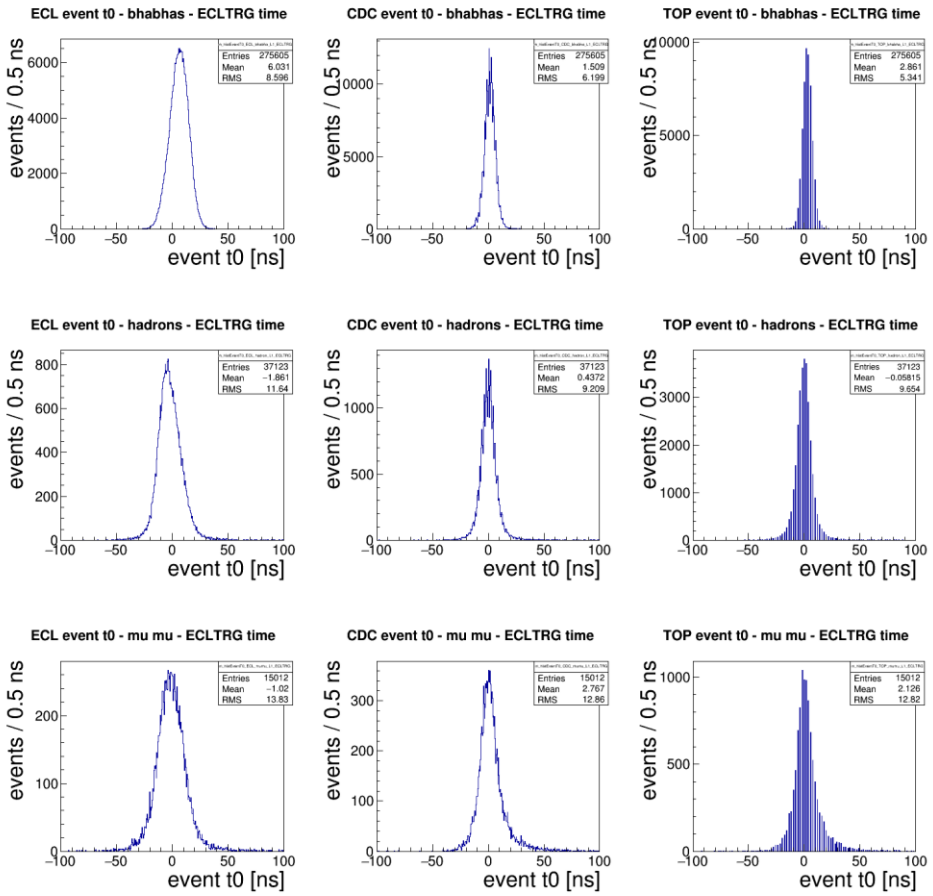
Fraction of fine event

- First version plot
  - X axis
    - $E_{max}\ cut$
    - range : 0 ~ 200 ADC
    - Can change depending on the SVD decision
  - Y axis
    - $Fraction = \frac{N_{ECL\ trigger}\ \&\ N_{E_{max}>X\ ADC}}{N_{ECL\ trigger}}$

Event T0



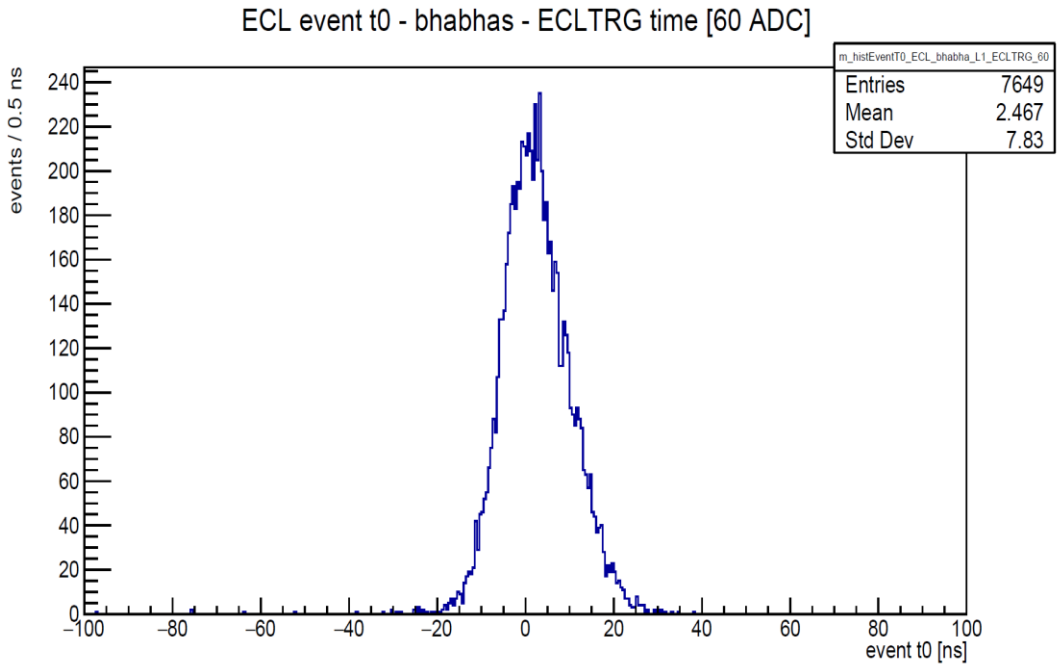
Event T0 plot



EventT0 ECLTRG of Physics run

- The current status
  - Total plot 18
  - ECL timing source : 9
  - CDC timing source : 9
  - Skim type : Bhabha, hadron, mumu

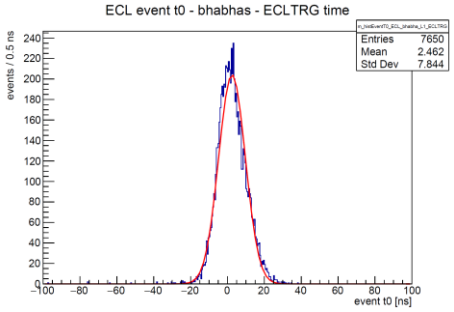
Update plot

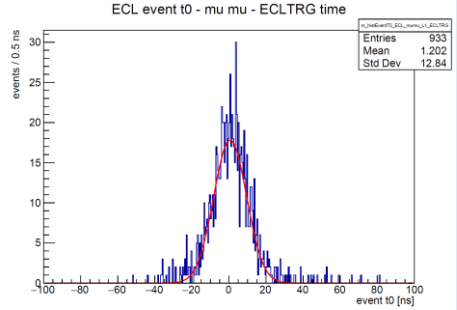


EventT0 ECLTRG with Emax > 60 ADC

- The plots for Event T0 resolution
  - ECL trigger
  - 9 plots per Emax cut
    - ► Emax cut
      - 20, 40, 60, 80, 100 ADC
  - Considering don't show these plot on DQM page

Extracting resolution



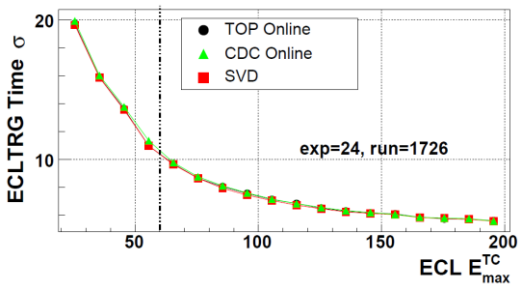


There is a module for checking ECL trigger jitter. This module, “DQMHistAnalysisEventT0”, make plots for Mirabelle. It use the double-gaussian fit.

For extracting resolution, used the single-gaussian fit. We checked if single gaussian working well. The single-gaussian worked well not only Bhabha skim but also mumu skim. We set the minimum the number of entry, 200. because it is likely that it will not work well in low statistic.

The modularization up to here was done.

Histogram Analysis Event T0 resolution



To do

To compare the single-gaussian and the double-gaussian.

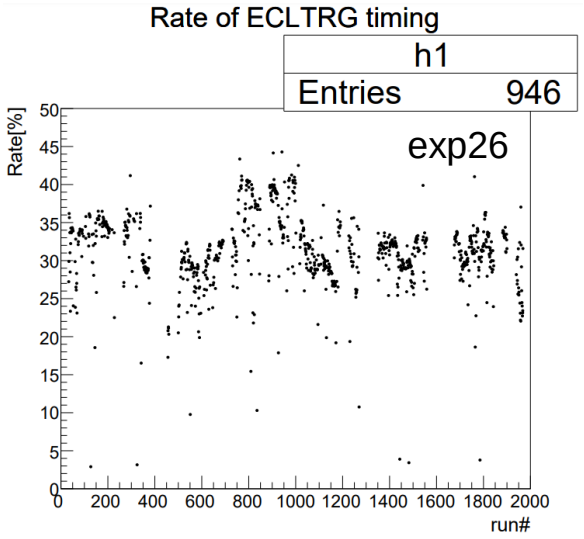
To check the effectiveness of the single-gaussian in low statistic.

To make the resolution plot by using the fit results.

To do

To do

- ECL trigger DQM
  - To make the resolution plot by using the fit results
  - To compare the single-gaussian and the double-gaussian.
  - To check the effectiveness of the single-gaussian in low statistic.
- QAM for Event T0 resolution
- Fraction of ECL TRG timing
  - Other exp
  - The Fraction as a function of statistic



Back up

Single gaussian

```
double TRGECLDQMHistAnalysisEventT0Module::ExtractingResolution(TH1* hist){

    TString tag = "gaussian";
  if (hist == nullptr) {
    B2DEBUG(20, "hist == nullptr");
    m_monObj->setVariable(Form("fit_%s", tag.Data()), 0);
    return  false;
  }

  int nToFit = hist->GetEntries();// - h->GetBinContent(0) - h->GetBinContent(h->GetNbinsX()+1);
  if (nToFit < m_nEntriesMin) {
    B2DEBUG(20, "not enough entries");
    m_monObj->setVariable(Form("fit_%s", tag.Data()), 0);
    return false;
  }


  //scale the histogram
  hist->Scale(1. / hist->GetEntries());
  hist->GetXaxis()->SetRangeUser(-50, 50);


  TF1 fitf("fit_gaus", TRGECLDQMHistAnalysisEventT0Module::fGaus, -50, 50, 6);
  fitf.SetParNames("N", "#mu_{1}", "#sigma_{1}");
  fitf.SetParameters(0.1, hist->GetMean(), hist->GetStdDev());
  fitf.SetParLimits(2, 0, 100); //sigma1


  hist->Fit("fit_gaus");


  Double_t par[3];
  fitf.GetParameters(&par[0]);

  return par[2];
}
```

Drawing histogram

```
void TRGECLDQMHistAnalysisEventT0Module::endRun()
{
  m_cECLTRG->cd();


        TString cut ="";

    for( int j = 0; j< 6; j++){

        if( j == 0){
           TH1* h = findHist(Form("TRGECLEventT0DQMdir/m_histEventT0_TOP_hadron_L1_ECLTRG%s", cut.Data()));
            h_TRGECLEventT0_TOP_hadron_L1_ECLTRG_resolution->Fill(j*20, ExtractingResolution(h));

            TH1* h1 = findHist(Form("TRGECLEventT0DQMdir/m_histEventT0_CDC_hadron_L1_ECLTRG%s", cut.Data()));
            h_TRGECLEventT0_CDC_hadron_L1_ECLTRG_resolution->Fill(j*20, ExtractingResolution(h1));
         }else {
           cut = Form("_%d",j*20);
           TH1* h = findHist(Form("TRGECLEventT0DQMdir/m_histEventT0_TOP_hadron_L1_ECLTRG%s", cut.Data()));
           h_TRGECLEventT0_TOP_hadron_L1_ECLTRG_resolution->Fill(j*20, ExtractingResolution(h));

           TH1* h1 = findHist(Form("TRGECLEventT0DQMdir/m_histEventT0_CDC_hadron_L1_ECLTRG%s", cut.Data()));
           h_TRGECLEventT0_CDC_hadron_L1_ECLTRG_resolution->Fill(j*20, ExtractingResolution(h1));
         }
    }// for end

  h_TRGECLEventT0_TOP_hadron_L1_ECLTRG_resolution->Draw();
 h_TRGECLEventT0_CDC_hadron_L1_ECLTRG_resolution->SetMarkerStyle(7);
    h_TRGECLEventT0_CDC_hadron_L1_ECLTRG_resolution->SetMarkerColor(7);
  h_TRGECLEventT0_CDC_hadron_L1_ECLTRG_resolution->Draw("same");

  if (m_printCanvas)
    m_cECLTRG->Print(Form("%s_ECLTRG.pdf", m_prefixCanvas.c_str()));

}
```

Event T0 generator

EventT0Generator

Module generates discrete event t0 in ~4ns steps (bunch spacing) according to double gaussian distribution and adds it to the production and decay times of MCParticles. This means that after this module the time origin (t = 0) is set to what L1 trigger would give as the collision time. In case of cosmics, the L1 triggerjitter is generated according to a continuos double gaussian distribution