Status of GRL firmware

Yun-Tsung Lai

KEK

yuntsung@post.kek.jp

Belle II TRG/DAQ workshop 2017

August 23, 2017

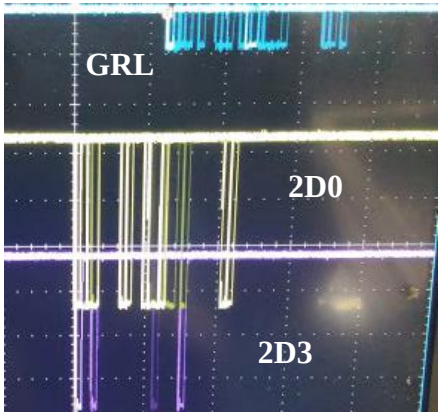
Outline

- Status and plan of GRL operation in E-Hut
- Development of GRL core logic: matching algorithm in VHDL
  - ISIM results
  - Generator interface
  - Resource usage
- Summary & To do

Status and plan of GRL operation in E-Hut

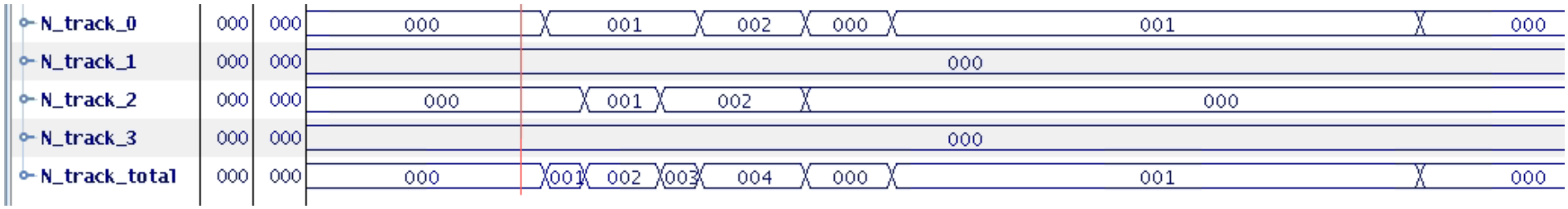
- In the current E-Hut setup, the CDCTRG data flow reaches GRL. GRL is receiving tracking results from the 4 2D boards.
  - In addition, the links from ETM and to GDL were built.
  - In GCRT, we use GRL with chipscope to monitor 2D output (tracking result and corresponding TSF info)

Track trigger signals



- GRL can provide the summary of 2D tracking results. Track counting within a 500 ns window is also implemented.

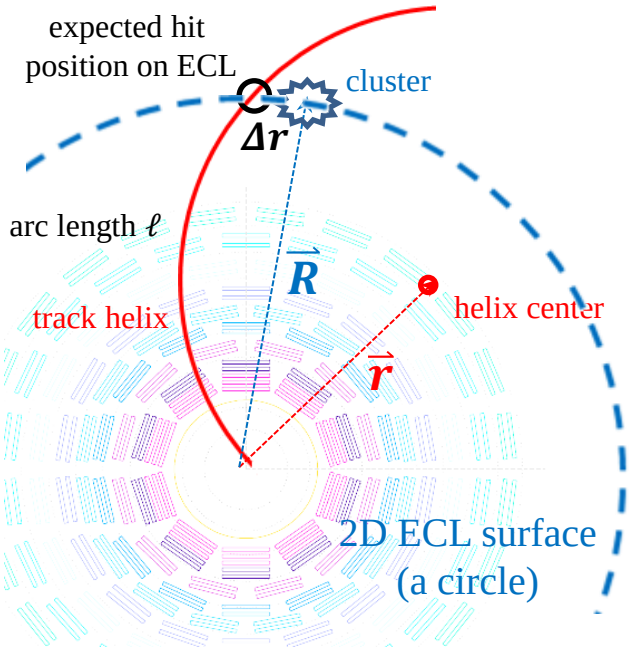
- Plan in near future: After 3D boards are installed and core logic is online, GRL will be responsible to provide various trigger signals about CDCTRG summary information. e.g. # of tracks



Matching algorithm – 2D

21st B2GM

- 2D tracker output from TRGCDC:
  - center of track helix: $\vec{r} = (r, \phi)$
  - p_t
  - Track is extrapolated as a circle/helix inside ECL.
- Output from TRGECL:
  - cluster position on ECL: $\vec{R} = (x, y) = (R, \theta)$ (Cluster position is with **15cm** depth to the inner surface.)
  - Deposit Energy



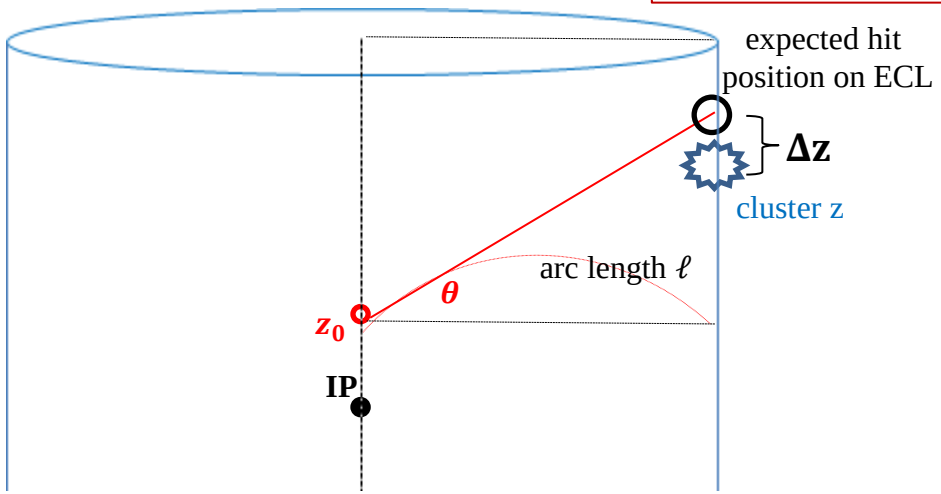
- Solve the expect hit position for a track on ECL:
  - $\overrightarrow{R'} = (x', y') = (R, \theta')$, where $\theta' = \mathbf{cos^{-1}}(\boldsymbol{R/2r}) + \boldsymbol{\phi}$ or $\boldsymbol{\phi} - \boldsymbol{cos^{-1}}(\boldsymbol{R/2r})$

- $\Delta r \equiv |\vec{R} - \overrightarrow{R'}|$: the deviation between cluster position and expected hit position.

Matching algorithm – Z

21st B2GM

- Z output from TRGCDC:
  - $z_0,\ tan\theta, p_z$
  - $tan\theta = \frac{p_z}{p_t} = \frac{z}{\ell}$
- Z output from TRGECL:
  - cluster z position



- Solve the expect hit position for a track on ECL:
  - $\boldsymbol{z' = z_0 + \ell tan\theta}$, where $\ell = 2rsin^{-1}(R/2r)$
  - $\boldsymbol{p_z = p_t tan\theta}$
  - $\boldsymbol{p} = \sqrt{p_t^2 + p_z^2}$
- $\Delta z \equiv z - z'$: the deviation between cluster z position and expected z position

Matching algorithm with FPGA

- GRL input:
  - 2D: 6*4 tracks' info ($p_t,\ \phi$)
  - 3D: 6*4 tracks' info ($z_0,\ tan\theta$)
  - ETM: ECL clusters' info (Hit position in Cartesian coordinate, energy)

- Matching between a track and a cluster: a single VHDL module.
  - Totally $N_{track}^{total} * N_{cluster}^{total}$ matching modules in parallel in firmware.
  - Resource usage issue: either trying to reduce the resource usage of a single module, or applying energy threshold to the cluster input to reduce the number.

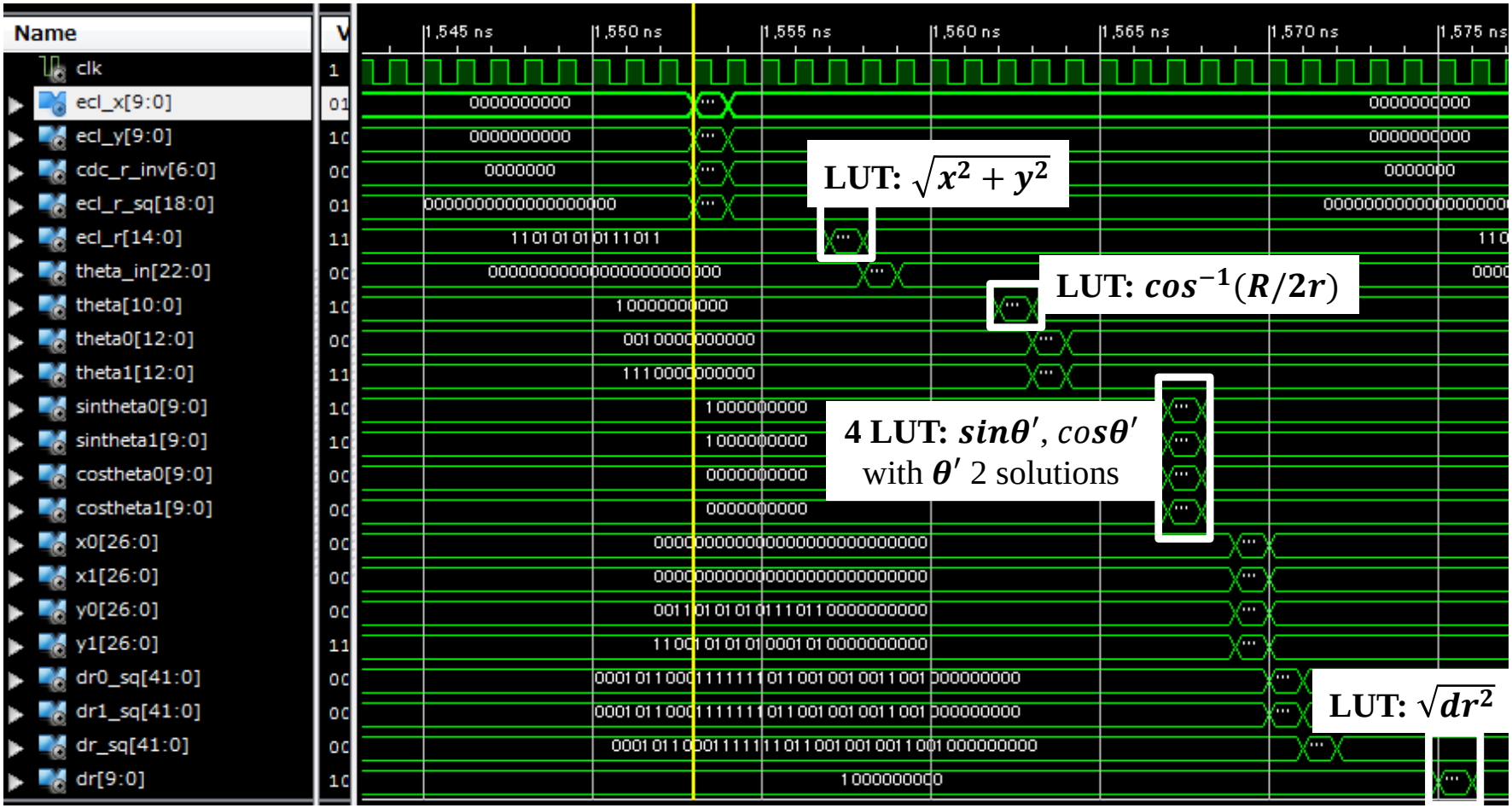
- The VHDL module is developed by the JSignal class provided by Jae-Bak.
  Special arithmetic operations are done by using LUT.
  e.g. sin, cos, sqrt, 1/x (for division), etc

- The logic uses 127 MHz sysclk to reduce latency.

ISIM result: 2D matching

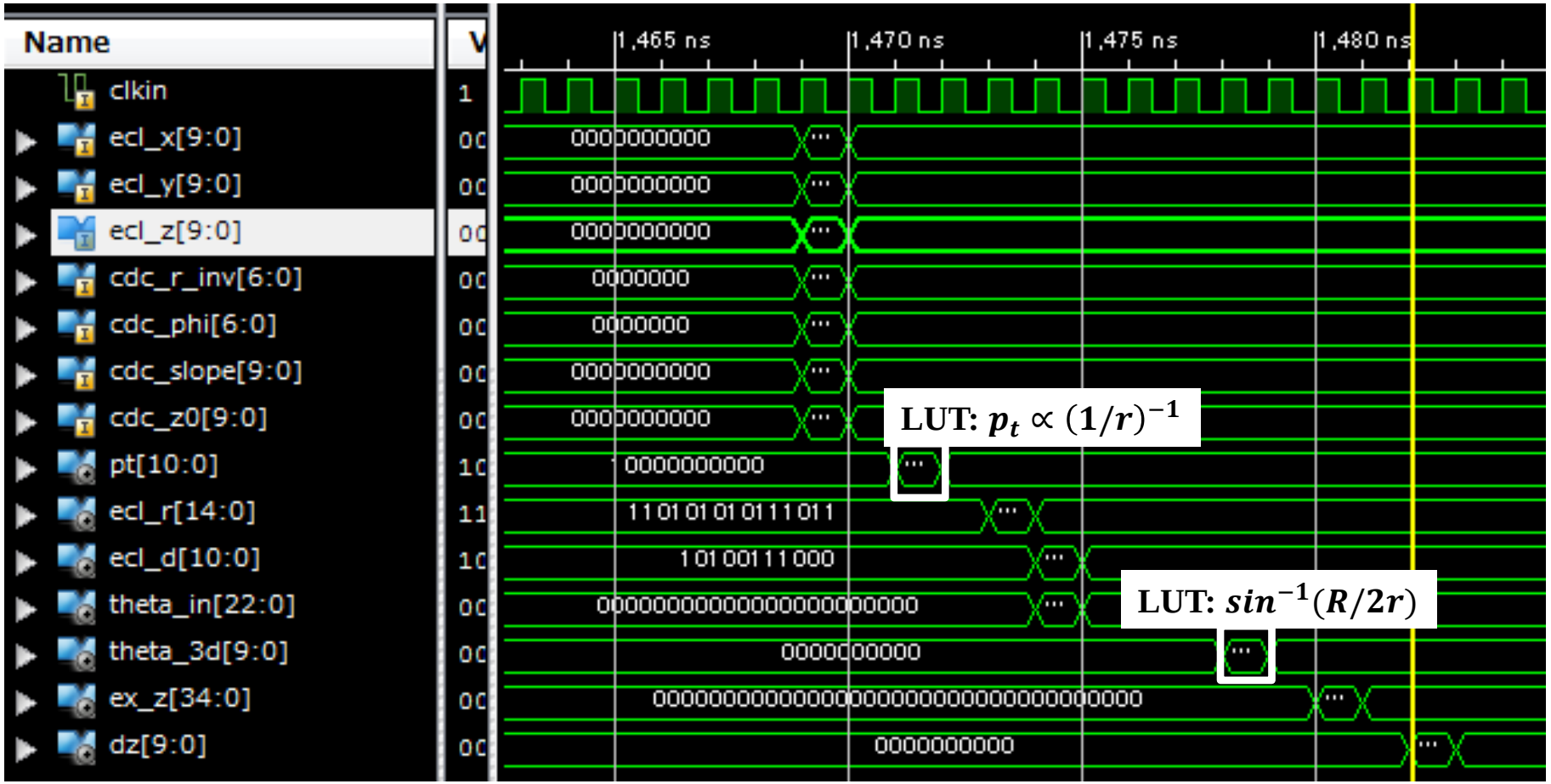
- CDC input: $\frac{1}{r}$ $(p_t), \phi$
- ECL input: x, y position
- Latency: 22 sysclk ~ 173 ns

- A multiplication takes 1 clk.
- A LUT procession takes 4 clks.



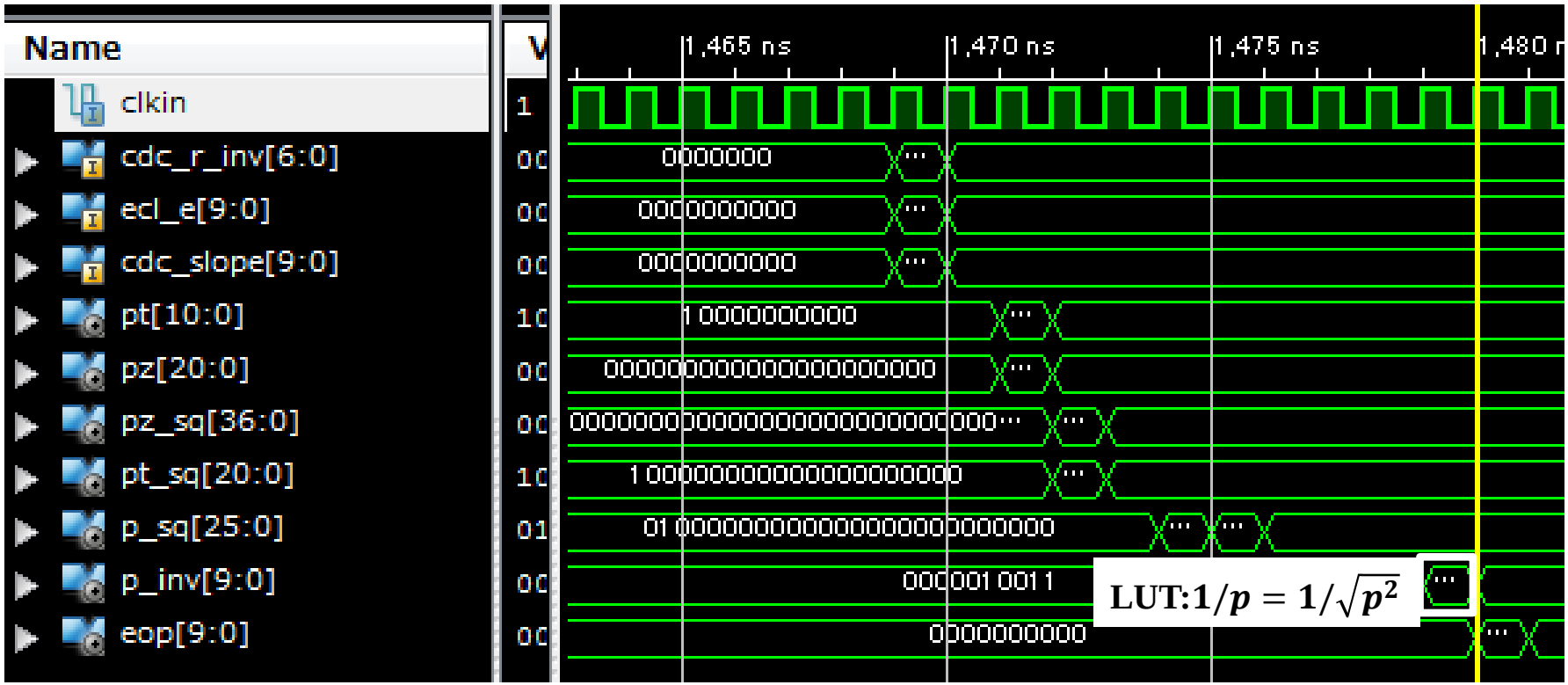
ISIM result: Z matching

- CDC input: $\frac{1}{r}$ $(p_t), z_0, tan\theta$
- ECL input: x, y, z position
- Latency: 13 sysclk ~ 102 ns



ISIM result: E/p

- CDC input: $\frac{1}{r}$ (p_t), $tan\theta$
- ECL input: energy
- Latency: 11 sysclk ~ 87 ns



Interface of the generator in TSIM

- The VHDL and LUT generator is implemented in TSIM TRGGRLMatch.

- The interface is parametrized to fit in modifications in the future:
  - Max, min, and bit width of the input variables, and LUT IO (depth and width).

```
void
TRGGRLMatch::VHDL_constant(std::map<std::string, int> & m_mInt, std::map<std::string, double> & m_mDouble)
{

  // Input signals: determined by sub-trigger systems
  m_mDouble["ECL_X_max"] = 130.0; m_mDouble["ECL_X_min"] = -130.0;
  m_mDouble["ECL_Y_max"] = 130.0; m_mDouble["ECL_Y_min"] = -130.0;
  m_mDouble["ECL_Z_max"] = 116.5; m_mDouble["ECL_Z_min"] = -75.5;
  m_mDouble["ECL_E_max"] = 10.0; m_mDouble["ECL_E_min"] = 0.0;
  m_mDouble["CDC_r_max"] = 3000.0; m_mDouble["CDC_r_min"] = 0.0;
  m_mDouble["CDC_r_inv_max"] = 0.035; m_mDouble["CDC_r_inv_min"] = 0.0;
  m_mDouble["CDC_phi_max"] = M_PI; m_mDouble["CDC_phi_min"] = -M_PI;
  m_mDouble["CDC_slope_max"] = 1.2; m_mDouble["CDC_slope_min"] = -1.2;
  m_mDouble["CDC_z0_max"] = 30.0; m_mDouble["CDC_z0_min"] = -30.0;

  m_mInt["ECL_X_bitsize"] = 10;
  m_mInt["ECL_Y_bitsize"] = 10;
  m_mInt["ECL_Z_bitsize"] = 10;
  m_mInt["ECL_E_bitsize"] = 10;
  m_mInt["CDC_r_bitsize"] = 7;
  m_mInt["CDC_r_inv_bitsize"] = 7;
  m_mInt["CDC_phi_bitsize"] = 7;
  m_mInt["CDC_slope_bitsize"] = 10;
  m_mInt["CDC_z0_bitsize"] = 10;
```

Resource usage issue

- The resource usage of a single module: ~ 0.6% of LUT.
  At most 136 parallel modules → $N_{track}^{total} * N_{cluster}^{total}$ is limited.

- Both the LUT IO bit width (depth and width) were set to be 10 bits for simplicity.
  More detailed estimation on the signals' resolution is necessary to reduce the LUT size effectively.
  - LUT info parameters are also adjustable:

```
// Bitsize of LUT input : Used in the constraint be
m_mInt["ECL_R_sq_c_bitsize"] = 10;
m_mInt["ECL_D_sq_c_bitsize"] = 10;
m_mInt["theta_in_c_bitsize"] = 10;
m_mInt["theta0_c_bitsize"] = 10;
m_mInt["theta1_c_bitsize"] = 10; // same as theta0_
m_mInt["dr_sq_c_bitsize"] = 10;
m_mInt["p_sq_c_bitsize"] = 10;

// Bitsize of LUT output : Determined by yourself,
m_mInt["ECL_R_bitsize"] = 10;
m_mInt["ECL_D_bitsize"] = 10;
m_mInt["theta_bitsize"] = 10;
```

- The total number of ECL clusters would be large (5120 bits for ETM→GRL).
  To reduce the number:
  - Energy threshold.
  - Pick up a certain number of clusters with the highest energy.

Summary & To do – Operation in E-Hut

- Summary:
  - CDCTRG data flow reaches GRL for the current GCRT.
    We use GRL to monitor the tracking result from the 4 2D boards and to summarize the track information.

- To do:
  - Provide 2D track trigger in near future.
  - After 3D boards are ready in E-Hut with core logic. GRL will summarize the CDCTRG information and prepare various types of triggers related to tracking.

Summary & To do – Core logic

- Summary:
  - The VHDL matching module is implemented by using Jae-Bak's class, which can output dr, dz and E/p.
  - Overall latency is 22 sysclk

- To do:
  - Try to reduce the resource usage by different approaches.
  - Make the delay and persistor modules between CDC and ECL inputs.
  - Test the logic after ETM provides clustering output.