# Task 2: Spatial resolution simulation

## Introduction

### Sensor resolution

One of the most pressing goals of silicon sensor R&D is to develop position-resolving detectors with better and better resolutions. The spatial resolution of silicon pixel detectors however depends on many parameters: There are obvious ones, such as the pixel pitch, but also more tricky ones like the incidence angle, the influence of a magnetic fields or the signal threshold of the readout chip.

For a binary read out pixel detector with a pitch of $w$, the resolution lies around $\sigma = w/\sqrt{12}$. In this task, we will use a simulation framework dedicated to silicon pixel detectors both to determine this standard resolution, but also to find ways to improve the resolution.

### Allpix$^2$

Allpix$^2$ is a modular simulation framework for silicon detectors.

Its main functionality is to simulate the detector response to a traversing particle, but due to its modularity also other functionalities could be added.

For simulating such a detector response, the following steps are performed:

- A user-defined particle is simulated to traverse the user-defined silicon detector. This is done using the Geant4 libraries, which then return the amount and location of the energy *deposited* in the sensor via ionization processes. The number of electron-hole pairs is calculated.
- In the (of course user-defined) electric field inside the active sensor volume, the charge carriers are *propagated* towards the electrodes. Several methods for this propagation exist, including a fourth-order Runge-Kutta stepping method.
- Depending on the final position of the charge carriers, they are associated with / *transferred* to a readout channel (a pixel).
- The behavior of the readout chip is simulated, *digitizing* the signal with a customized conversion from charge carriers to ADC counts (gain), dispersed thresholds and noise.

In all of these steps there are various parameters that can be customized by the user in order to either adapt it to a specific sensor, the details required as an output or for an optimization of the performance. More information can be found in this tutorial.

A detailed user manual is provided for the software.

## Tasks (in a nutshell)

1. Simulate the response of a silicon pixel detector to a traversing highly relativistic particle
2. Find a way to determine the resolution of a binary planar silicon pixel detector
3. Check the influence on the resolution by ...
    a. moving from a binary read out detector to a several-bit digital output
    b. using an inclined particle incidence
    c. defining a magnetic field
    d. altering the pixel pitch

## Instructions

### Set the environment

An installation of Allpix$^2$ is obviously required for this tutorial. We recommend using the environment of our NAF (National Analysis Facility) servers and the installation on CVMFS (Cern VM File System) for this, as described in the following steps. If you have an installation ready on your computer, feel free to use it. You might still need some files from the NAF to follow the tutorial, as you can also read in the following.

Log in to the NAF with the credentials you are given at the tutorial (the part inside the brackets is required when being outside the DESY network (remove the brackets when using this option.).),

```
ssh -XY (-J username@bastion.desy.de) username@naf-fhlabs.desy.de
```

and go to the working directory:

```
cd /nfs/dust/fhlabs/group/EDIT/allpix-track
```

Create your own subdirectory and *cd* into it.

Allpix$^2$ does not necessarily have to be installed on your working PC. There's an installation available within the CVMFS that is accessible from the NAF, as well as docker images. We'll use the CVMFS installation by sourcing the corresponding setup script:

```
source /cvmfs/clicdp.cern.ch/software/allpix-squared/1.4.3/x86_64-centos7-gcc7-opt/setup.sh
```

Check that the executable *allpix* is now available.

Copy the configuration and geometry files *detector.conf*, *visual.conf*, *start.conf* and *replay.conf* from the directory */nfs/dust/fhlabs/group/EDIT/allpix-track/start* to your working directory.

*Hint for Mac OSX users:* X-forwarding might not work properly. We could solve the issue by downgrading XQuartz (duh) to version 2.7.8, see here: https://www.xquartz.org/releases/XQuartz-2.7.8.html

### Visual inspection of the setup

Look at these files:

- *detector.conf*: Contains the geometry information of the setup, with the type, position and orientation of your detector. The detector used here follows the geometry of a CMS Phase I Pixel detector, with 52 x 80 pixels of the size 150 x 100 $\mu m^2$.
- *visual.conf*: Simplified simulation file for visualization:
  - **Allpix**: Generic parameters such as file locations, number of events, ...
  - **GeometryBuilderGeant4**: Translates the defined geometry into a Geant4 compatible geometry.
  - **MagneticFieldReader**: Defines a magnetic field for the G4 simulation and the charge propagation in the sensor.
  - **DepositionGeant4**: Runs G4 to determine the particle trajectory and the energy deposited inside the active sensor volumes.
  - **VisualizationGeant4**: Enables to visually inspect the geometry setup.

Run Allpix giving it the configuration file to be used:

```
allpix -c visual.conf
```

An error message should pop up. What does this error message tell you?

Play around with the geometry and the beam description if you like.

### First simulation

Look at the following configuration file, containing the complete workflow of a detector response simulation for a single detector:

- *start.conf*: In this file, the modules used for the simulation and their parameters are defined.
  - **Allpix**: Generic parameters such as file locations, number of events, ...
  - **GeometryBuilderGeant4**: Translates the defined geometry into a Geant4 compatible geometry.
  - **MagneticFieldReader**: Defines a magnetic field for the G4 simulation and the charge propagation in the sensor.
  - **DepositionGeant4**: Runs G4 to determine the particle trajectory and the energy deposited inside the active sensor volumes.
  - **ElectricFieldReader**: Defines an electric field for the charge propagation in the sensor, either calculated as a linear function or read in as a more realistic, CAD-simulated field distribution.
  - **GenericPropagation**: Propagation of the charge carriers in the sensor. One can switch on/off the propagation of electrons and holes, define step sizes, propagate groups of charge carriers, ...
  - **SimpleTransfer**: Association of propagated charge carriers at the sensor surface to the corresponding pixel.
  - **DefaultDigitizer**: Digitization of the charge information, emulating the logic of a pixel read out. Define noise, thresholds and the ADC parameters.
  - **DetectorHistogrammer**: Provides plots for an overview of the simulated detector performance.

Run this simulation for 2000 events (fyi: When the simulation is interrupted with *Ctrl+c*, the data is saved up to the current event). Have a look at the generated root file (use a *TBrowser*), which is located in */your/working/directory/output*. Discuss the plots and especially look at cluster sizes and residuals.

The output file name can be change with the *root_file* description.

Play around the the "log_level" parameter to change the level of output details printed on the terminal. The list of possible logging levels can be found in the manual.

### A closer look at the charge propagation (optional)

Start from the file *start.conf* and switch on the options *output_animations* and *output_linegraphs* (module *GenericPropagation*) and run this simulation for **one individual** event. Have a look at the output root file and at your output directory.

*Hint: One can adjust the parameter timestep_max to achieve more meaningful results.*

### Extracting the detector resolution

In Allpix it is possible to store and recall the data of intermediate simulation steps using the *ROOTObjectWriter* and the *ROOTObjectReader*. For the sensor under investigation we have created files containing objects of the type *DepositedCharge* and *PropagatedCharge* for different sensor rotations and magnetic fields. This should make it faster to investigate the detector resolution as a function of these two parameters while sparing the two most time consuming steps *DepositionGeant4* and *GenericPropagation*. The files are called *EDIT_<rotation>deg_<BField>T_data.root* and are located in the directory */nfs/dust/fhlabs/group/EDIT/allpix-track/simFiles/data* .

Look at the configuration file:

- replay.conf: Configuration loading the simulated propagated charges.
  - **Allpix**: Generic parameters such as file locations, number of events, ...
  - **ROOTObjectReader**: Read in intermediate simulation results from previous runs. Choose which objects to load.
  - **SimpleTransfer**: Association of propagated charge carriers at the sensor surface to the corresponding pixel.
  - **DefaultDigitizer**: Digitization of the charge information, emulating the logic of a pixel read out. Define noise, thresholds and the ADC parameters.
  - **DetectorHistogrammer**: Provides plots for an overview of the simulated detector performance.

Start without rotation and without magnetic field (*EDIT_0deg_0T_data.root*) and calculate the RMS of the residual ( resolution) on both *x* and *y* (that's something you could automate/script).

*Hint: Consider using not the full pre-simulated 100 kEvents, since this might take a bit too long.*

Then check the following parameters:

- Digitization:
  - By default our silicon detector gives us binary hit information (*adc_resolution = 1* (given in bits)). Go to typical ADC resolutions for charge readout. How does it influence the resolution?
  - Does the threshold have an influence?
- Rotation:
  - At which rotation angle does one achieve the best resolution? Why?
  - Is this influenced by the resolution of the ADC?
- Magnetic field:
  - At which magnetic field does one achieve the best resolution? Why?
  - Is this influenced by the resolution of the ADC?

***Hint**: It's not necessary to create one configuration file for every set of parameters. Any parameter can be overruled when calling Allpix using the following syntax:*

```
allpix -c some/configuration.conf -o generic_parameter=value -o CorrespondingModule.desired_parameter=value
```

### Extra task (optional)

Can you set up and simulate a typical test beam experiment with a six-plane EUDET-type (Mimosa26 sensors) beam telescope and a DUT (Device under test)?

*Hint: Have a look at the list of predefined sensor geometries in Allpix: https://gitlab.cern.ch/allpix-squared/allpix-squared/tree/master/models*