The
Allpix Squared Detector Framework

What A Year!

Simon Spannagel, **Paul Schütze**

3rd Allpix Squared User Workshop
9. - 11. May 2022

Software Overview Talk FAQs

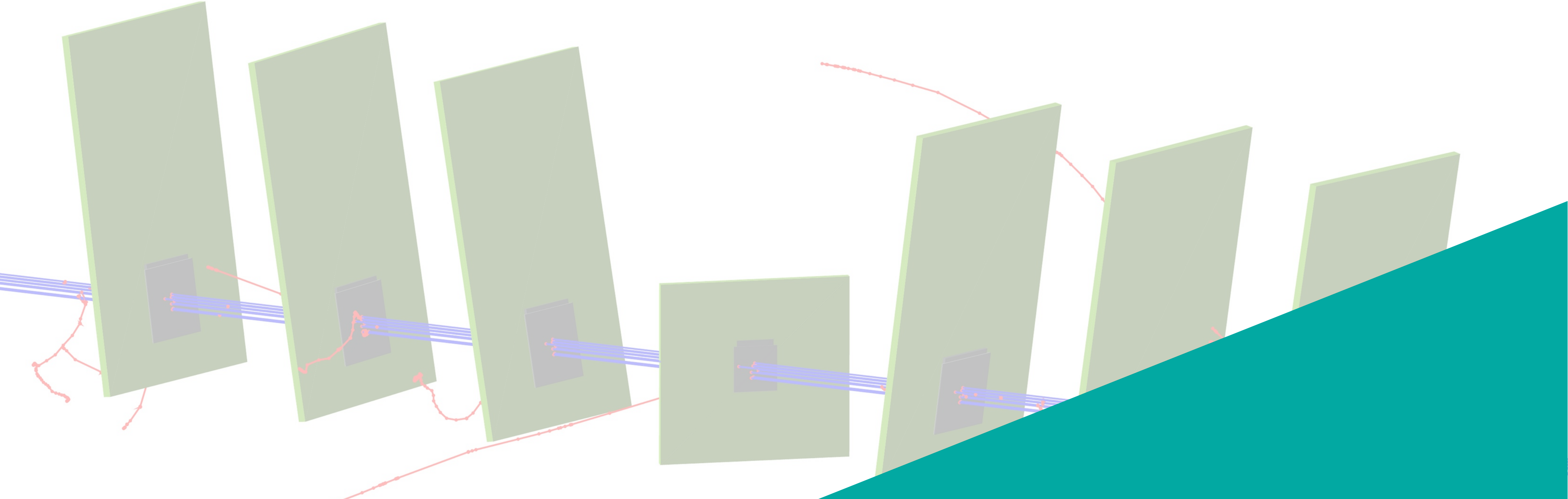
- **What's Allpix Squared & how does it work?**
  - ➔ Part I

- **What's new in Allpix Squared?**
  - ➔ Part III

- **What's next on Allpix Squared?**
  - A few things are already merged …
    - ➔ Part III
  - Many features under development
    - ➔ See several presentations in the next three days, i.a. by Simon on Wednesday

Allpix Squared

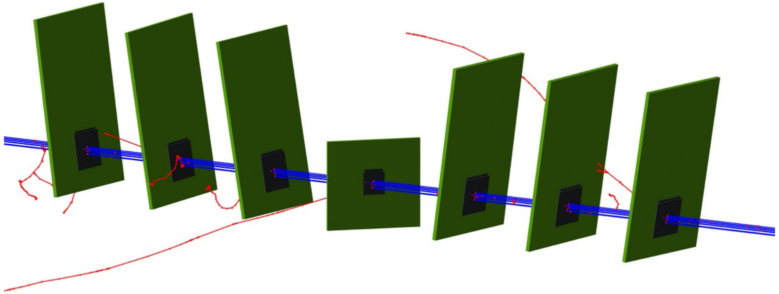
A Detector Simulation Framework



Motivation & History

Initial Motivation: **Monte Carlo simulation of silicon pixel detectors!**

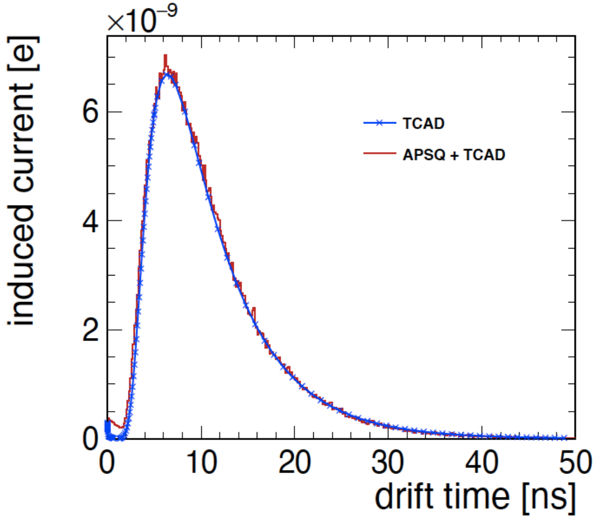
- Started at CERN EP-LCG – different groups from High Energy Physics got involved
  - ➔ Different phases of detector R&D to cover
  - Required a tool that at the same time is useful for …
    - Generic sensor R&D
    - Integration of detector systems, e.g. test beam setups
    - Validating simulation algorithms



➔ 2017: Allpix Squared 1.0 released with modular design & basic set of modules

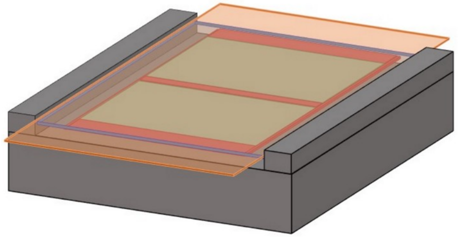
Motivation & History

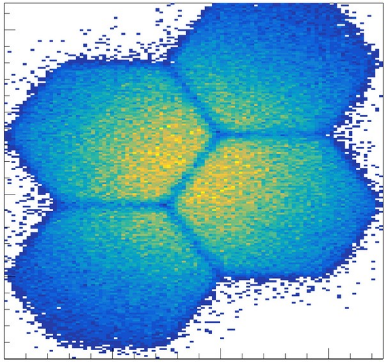
- The devil's in the details:
  HEP community targets high-precision simulations & realistic behaviour
  - Access to time-resolved information
    ➔ Transient simulation, pulse storage, amplifier simulation …
  - Inclusion of further physics effects
    ➔ User-selected recombination, mobility and trapping models, Shockley-Ramo theorem …
  - Interfaces to other frameworks
    ➔ TCAD electric fields, weighting potentials



Motivation & History

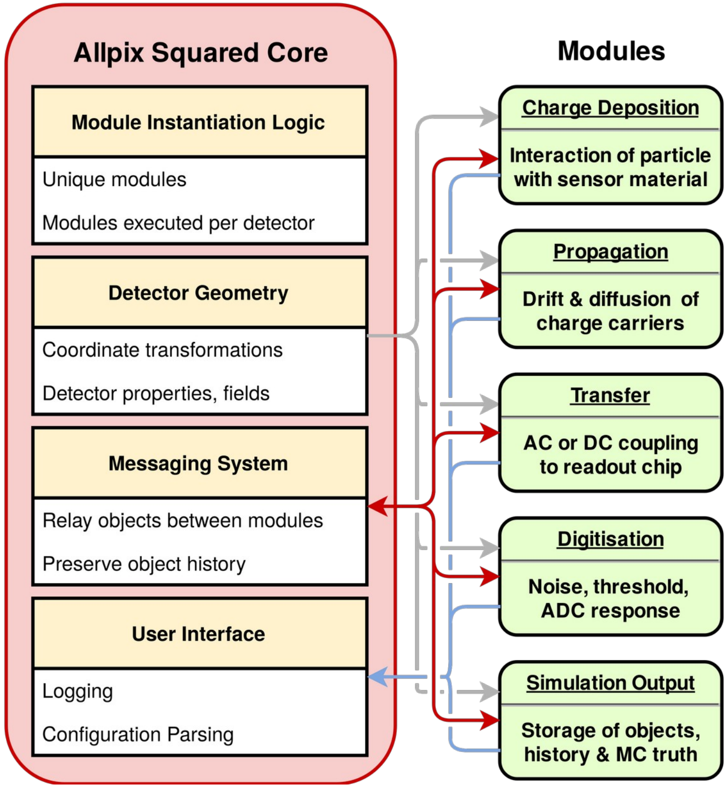
- The more the merrier:
  New users & applications – many of them outside the HEP community
  - Demands interfaces to other software and frameworks
    - ➔ Charge carrier input from file, different particle sources, flexible G4 interface
    - ➔ Various output formats, storage options, interfaces to analysis frameworks
  - Different detector types & geometries
    - ➔ Monolithic & hybrid sensors, radial strips,
      3D pixels, hexagonal pixels …
  - Different detector materials
    - ➔ Sensor material as a simulation parameter
  - Various applications
    - ➔ Passive materials, magnetic field, cosmic rays, …



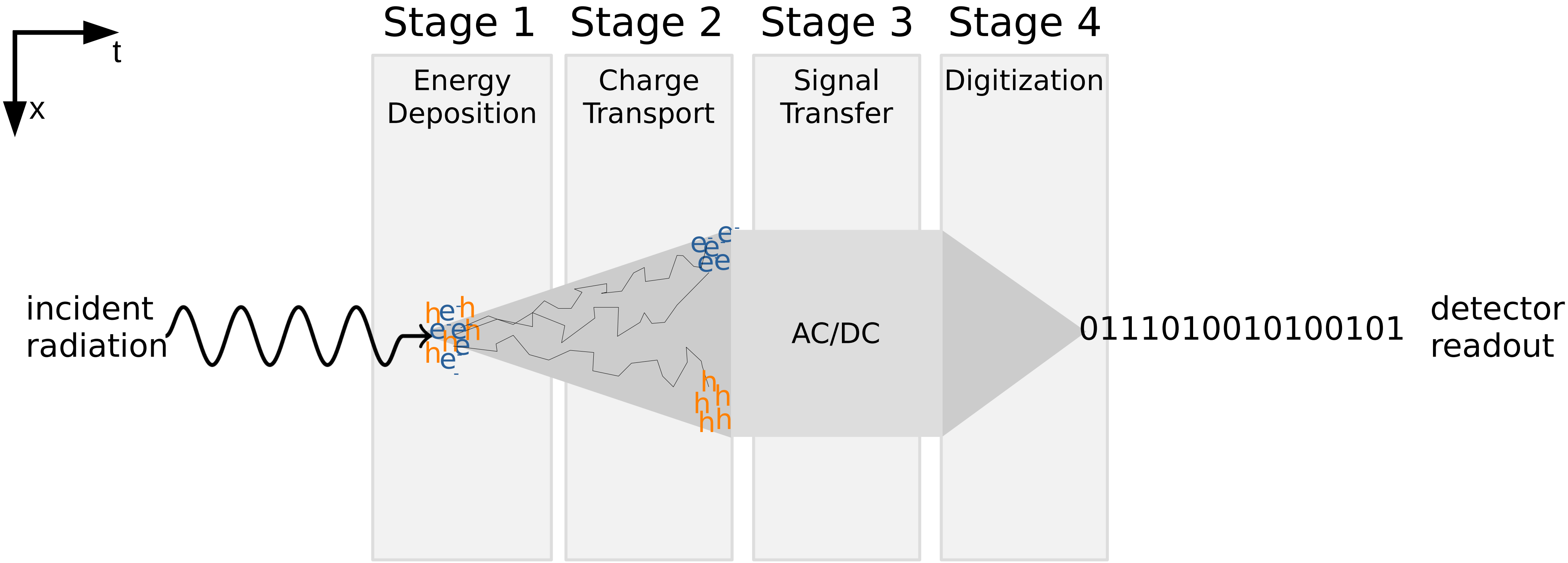


The Allpix² Framework

- Modular & flexible Monte Carlo simulation software
  - Modularity:
    separate infrastructure from physics
- Focus on usability & stability
  - Easy setup & configuration
  - Provide documentation (190p. user manual)
  - Regular patch & feature releases
  - User support in forum



Particle Detection in Silicon Sensors

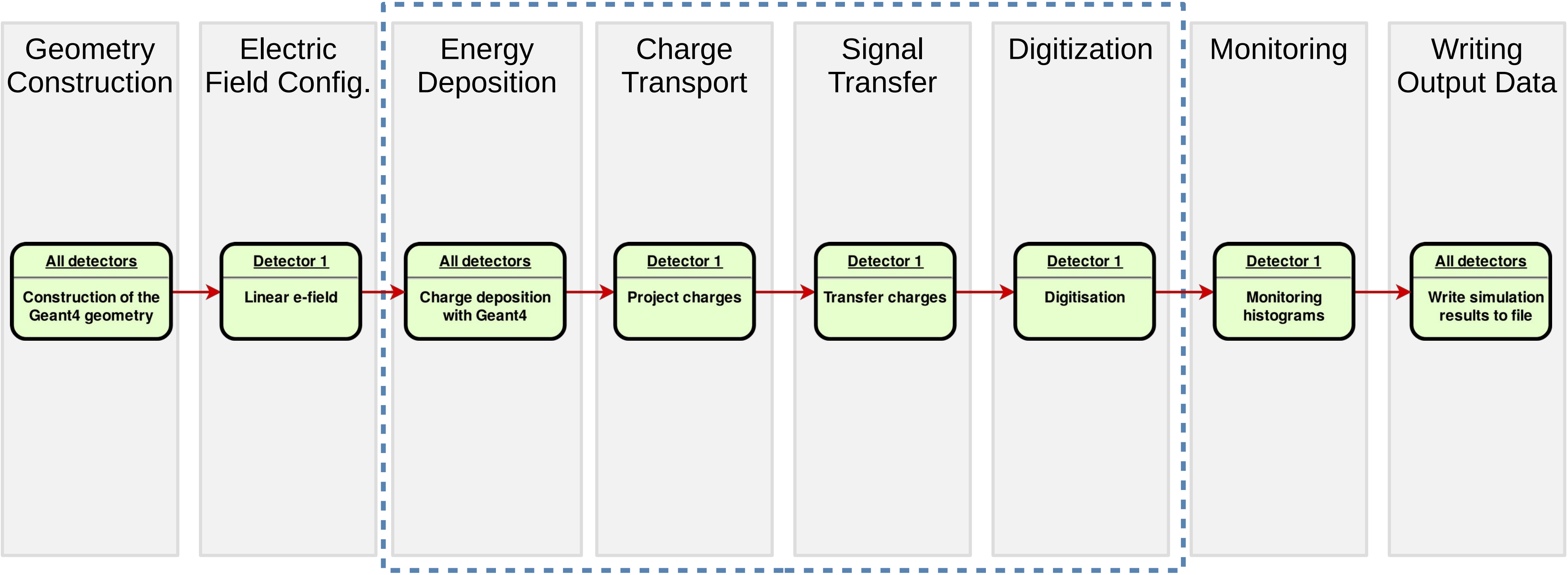


The Simulation Chain

- Geometry Construction
- Electric Field Config.
- Energy Deposition
- Charge Transport
- Signal Transfer
- Digitization
- Monitoring
- Writing Output Data

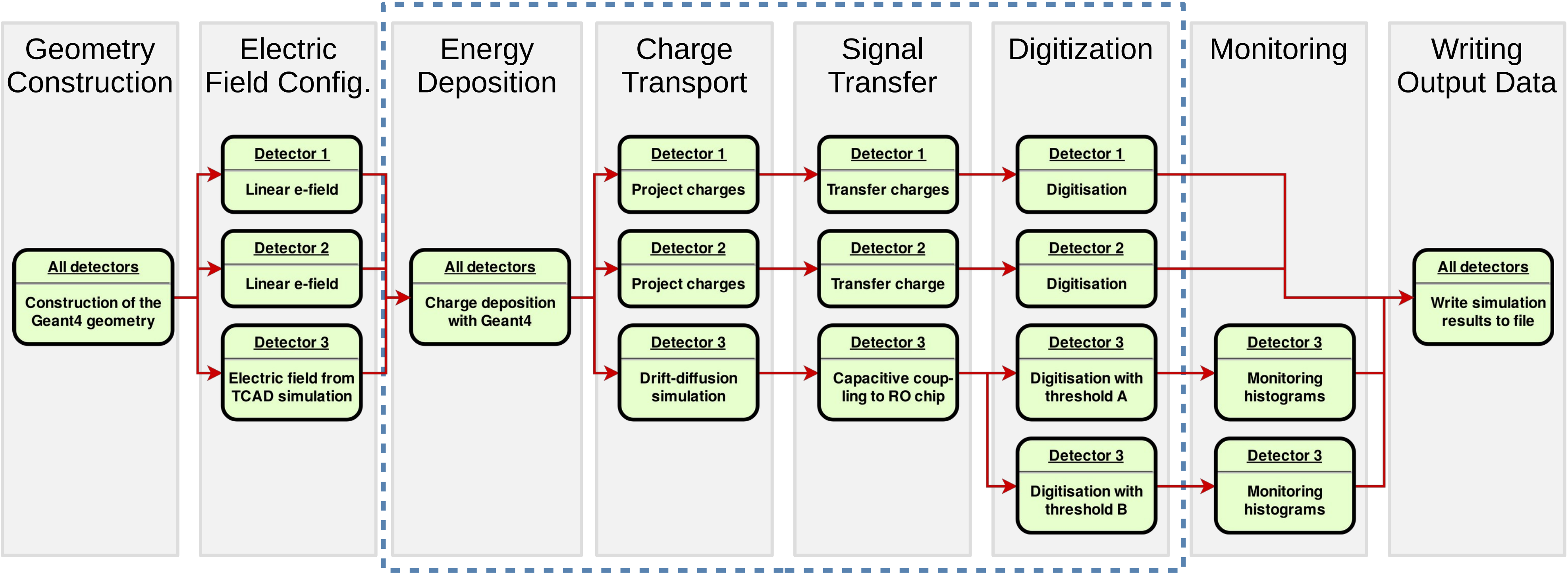
The Simulation Chain

- Building blocks follow individual steps of the signal formation in detectors
- Algorithms for each step can be chosen independently



The Simulation Chain

- Flexible simulation: modules configurable on per-detector level
- Multiple instances can be run in parallel (e.g. to simulate different front-ends)



Configuration of the Simulation Chain

- Building simulation chain from individual modules
  - Configuration file with modules in order of execution
  - Support for physical units
- *Every parameter documented in manual*
- Geometry configuration
  - File with position/orientation of individual detectors
  - Model files define detector geometries
  - Several detector models pre-configured

```
[AllPix]
log_level = "INFO"
number_of_events = 500000
detectors_file = "telescope.conf"

[GeometryBuilderGeant4]
world_material = "air"

[DepositionGeant4]
physics_list = FTFP_BERT_LIV
particle_type = "Pi+"
number_of_particles = 1
beam_energy = 120GeV
# ...

[ElectricFieldReader]
model="linear"
bias_voltage=150V
depletion_voltage=50V

[GenericPropagation]
temperature = 293K
charge_per_step = 10
spatial_precision = 0.0025um
timestep_max = 0.5ns

[SimpleTransfer]
```

Sustainable Development of Allpix^2

- Goal: develop a sustainable software – that is…
  - **validated** with prototype data & device simulations
  - **maintainable** over a period longer than O(1 fellow) / O(1 PhD)
  - well **documented** – 160p user manual

- Achieve with…
  - Extensive documentation → Low barrier for new users
  - Continuous integration → Automated tests
  - Rigorous code review → High code-quality
  - Clear & permissive license → Re-usability

GenericPropagation

Maintainer: Koen Wolters (koen.wolters@cern.ch), Simon Spannagel (simon.spannagel@cern.ch)
Status: Functional
Input: DepositedCharge
Output: PropagatedCharge

Description

Simulates the propagation of electrons and/or holes through the sensitive sensor volume of the detector. It allows to propagate sets of charge carriers together in order to speed up the simulation while maintaining the required accuracy. The propagation process for these sets is fully independent and no interaction is simulated. The maximum size of the set of propagated charges and thus the accuracy of the propagation can be controlled.

The propagation consists of a combination of drift and diffusion simulation. The drift is calculated using the charge carrier velocity derived from the charge carrier mobility parameterization by C. Jacoboni et al. @jacoboni. The correct mobility for either electrons or holes is automatically chosen, based on the type of the charge carrier under consideration. Thus, also input with both electrons and holes is treated properly.

The two parameters `propagate_electrons` and `propagate_holes` allow to control which type of charge carrier is propagated to their respective electrodes. Either one of the carrier types can be selected, or both can be propagated. It should be noted that this will slow down the simulation considerably since twice as many carriers have to be handled and it should only be used where sensible. The direction of the propagation depends on the electric field configured, and it should be ensured that the carrier types selected are actually transported to the implant side. For linear electric fields, a warning is issued if a possible misconfiguration is detected.

A fourth-order Runge-Kutta-Fehlberg method with fifth-order error estimation is used to integrate the electric field. After every Runge-Kutta step, the diffusion is accounted for by applying an offset drawn from a Gaussian distribution calculated from the Einstein relation

$$\sigma = \sqrt{\frac{2kT}{e}\mu t}$$

using the carrier mobility μ, the temperature T and the time step t. The propagation stops when the set of charges reaches any surface of the sensor.

The propagation module also produces a variety of output plots. These include a 3D line plot of the path of all separately propagated charge carrier sets from their point of deposition to the end of their drift, with nearby paths having different colors. In this coloring scheme, electrons are marked in blue colors, while holes are presented in different shades of orange. In addition, a 3D GIF animation for the drift of all individual sets of charges (with the size of the point proportional to the number of charges in the set) can be produced. Finally, the module produces 2D contour animations in all the planes normal to the X, Y and Z axis, showing the concentration flow in the sensor. It should be noted that generating the animations is very time-consuming and should be switched off even when investigating drift behavior.

Dependencies

This module requires an installation of Eigen3.

Parameters

- `temperature` : Temperature of the sensitive device, used to estimate the diffusion constant and therefore the strength of the diffusion. Defaults to room temperature (293.15K).
- `charge_per_step` : Maximum number of charge carriers to propagate together. Divides the total number of deposited charge carriers at a specific point into sets of this number of charge carriers and a set with the remaining charge carriers. A value of 10 charges per step is used by default if this value is not specified.
- `spatial_precision` : Spatial precision to aim for. The timestep of the Runge-Kutta propagation is adjusted to reach this spatial precision after calculating the uncertainty from the fifth-order error method. Defaults to 0.1nm.
- `timestep_start` : Timestep to initialize the Runge-Kutta integration with. Appropriate initialization of this parameter reduces the time to optimize the timestep to the spatial_precision parameter. Default value is 0.01ns.
- `timestep_min` : Minimum step in time to use for the Runge-Kutta integration regardless of the spatial precision. Defaults to 0.5ps.
- `timestep_max` : Maximum step in time to use for the Runge-Kutta integration regardless of the spatial precision. Defaults to 0.1ns.
- `integration_time` : Time within which charge carriers are propagated. After exceeding this time, no further propagation is performed for the respective carriers. Defaults to the LHC bunch crossing time of 25ns.
- `propagate_electrons` : Select whether electron-type charge carriers should be propagated to the electrodes. Defaults to true.
- `propagate_holes` : Select whether hole-type charge carriers should be propagated to the electrodes. Defaults to false.
- `output_plots` : Determines if output plots should be generated for every event. This causes a significant slow down of the simulation, it is not recommended to enable this option for runs with more than a couple of events. Disabled by default.
- `output_plots_step` : Timestep to use between two points plotted. Indirectly determines the amount of points plotted. Defaults to timestep_max if not explicitly specified.
- `output_plots_theta` : Viewpoint angle of the 3D animation and the 3D line graph around the world X-axis. Defaults to zero.
- `output_plots_phi` : Viewpoint angle of the 3D animation and the 3D line graph around the world Z-axis. Defaults to zero.
- `output_plots_use_pixel_units` : Determines if the plots should use pixels as unit instead of metric length scales. Defaults to false (thus using the metric system).
- `output_plots_use_equal_scaling` : Determines if the plots should be produced with equal distance scales on every axis (also if this implies that some points will fall out of the graph). Defaults to true.
- `output_plots_align_pixels` : Determines if the plot should be aligned on pixels, defaults to false. If enabled the start and the end of the axis will be at the split point between pixels.
- `output_animations` : In addition to the other output plots, also write a GIF animation of the charges drifting towards the electrodes. This is very slow and writing the animation takes a considerable amount of time, therefore defaults to false. This option also requires `output_plots` to be enabled.
- `output_animations_time_scaling` : Scaling for the animation used to convert the actual simulation time to the time step in the animation. Defaults to 1.0e9, meaning that every nanosecond of the simulation is equal to an animation step of a single second.
- `output_animations_marker_size` : Scaling for the markers on the animation, defaults to one. The markers are already internally scaled to the charge of their step, normalized to the maximum charge.
- `output_animations_contour_max_scaling` : Scaling to use for the contour color axis from the theoretical maximum charge at every single plot step. Default is 10, meaning that the maximum of the color scale axis is equal to the total amount of charges divided by ten (values above this are displayed in the same maximum color). Parameter can be used to improve the color scale of the contour plots.
- `output_animations_color_markers`: Determines if colors should be for the markers in the animations, defaults to false.

Usage

A example of generic propagation for all sensors of type "Timepix" at room temperature using packets of 25 charges is the following:

```
[GenericPropagation]
type = "timepix"
temperature = 293K
charge_per_step = 25
```

How to get started …

- CVMFS: binaries and dependencies available

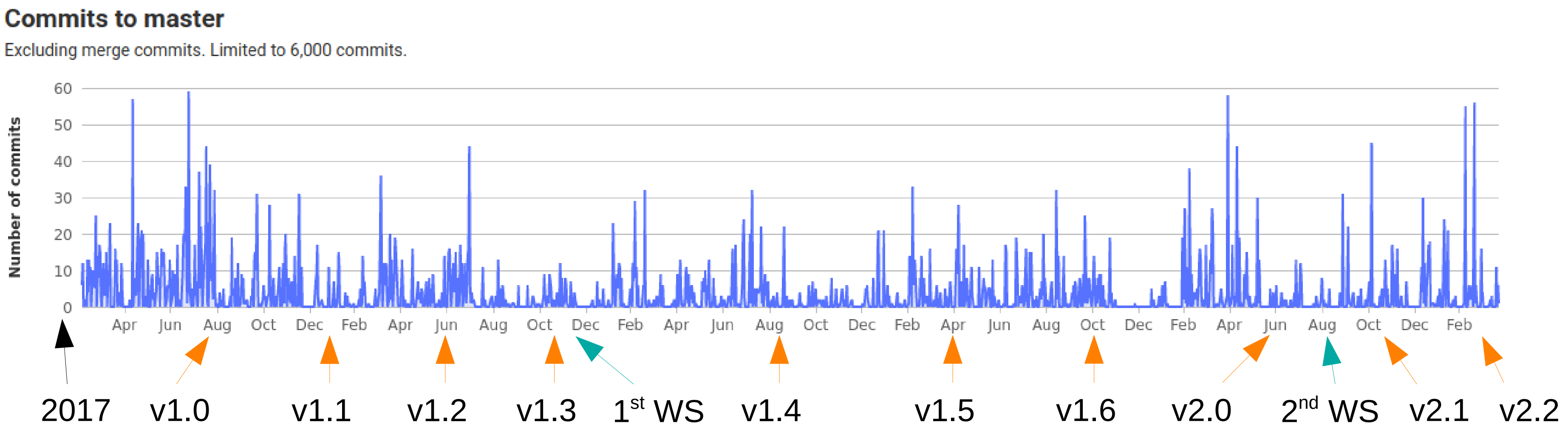
```
$ source /cvmfs/clicdp.cern.ch/software/allpix-squared/2.2.2/x86_64-centos7-clang12-opt/setup.sh
$ allpix --version
Allpix Squared version v2.2.2
               built on 2022-04-01, 12:43:46 UTC
```

- Docker images

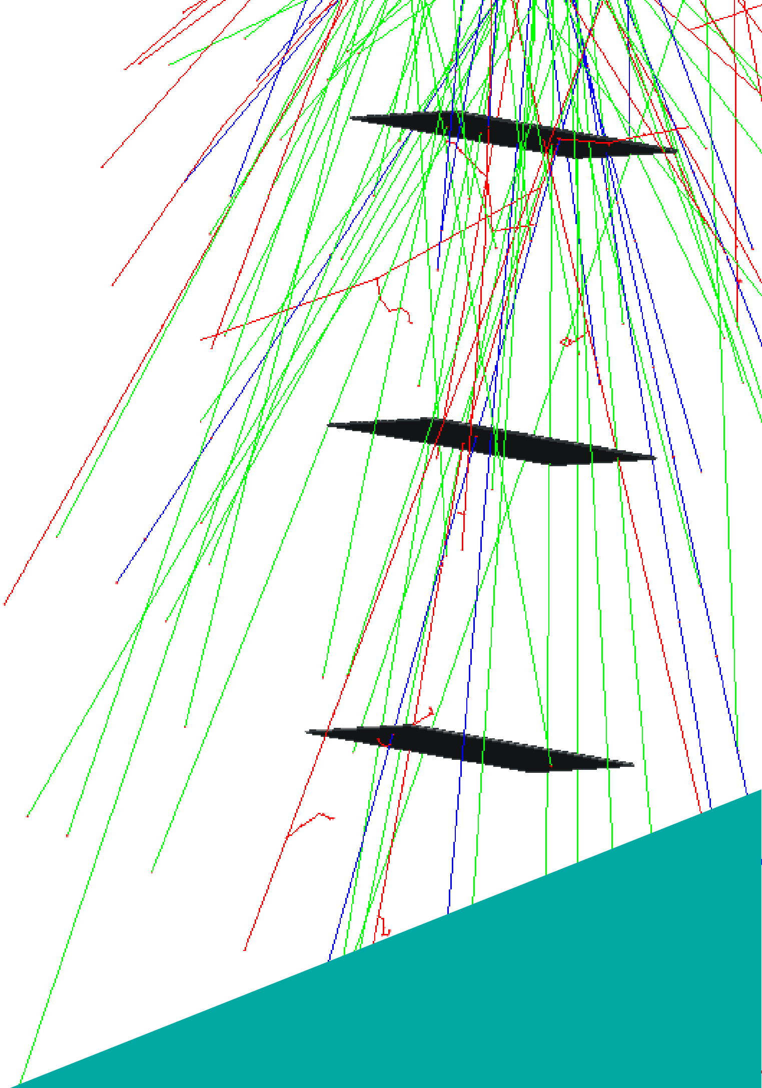
- Compile from source
    - Dependencies on *ROOT* & *Boost.Random*
    - *Geant4* & *Eigen* are optional

Allpix Squared Through the Ages

- Steady development since 2016 – 43 releases
  - Started within CERN EP-LCG group – now main development at DESY
- User-driven:
  - 45 contributors from various fields



Latest Releases



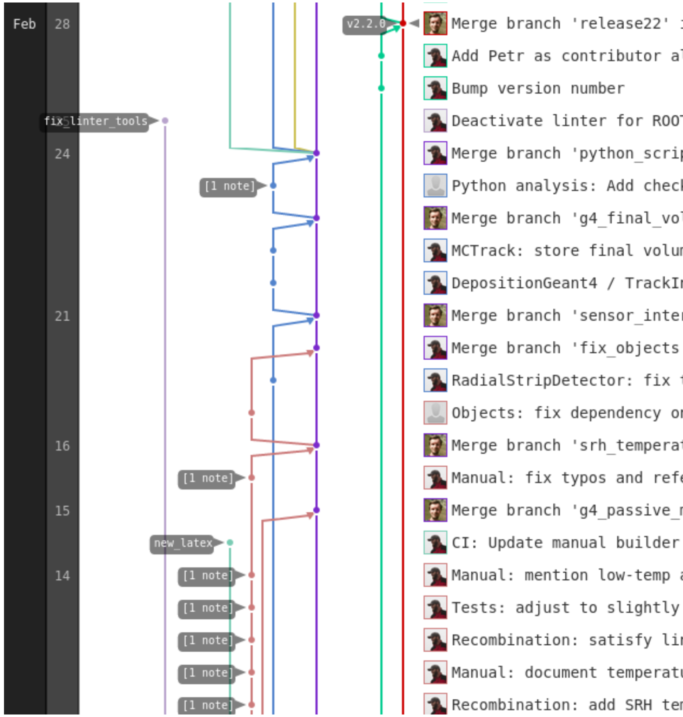
Allpix^2 – Recent Releases

Allpix2 Version 2.1

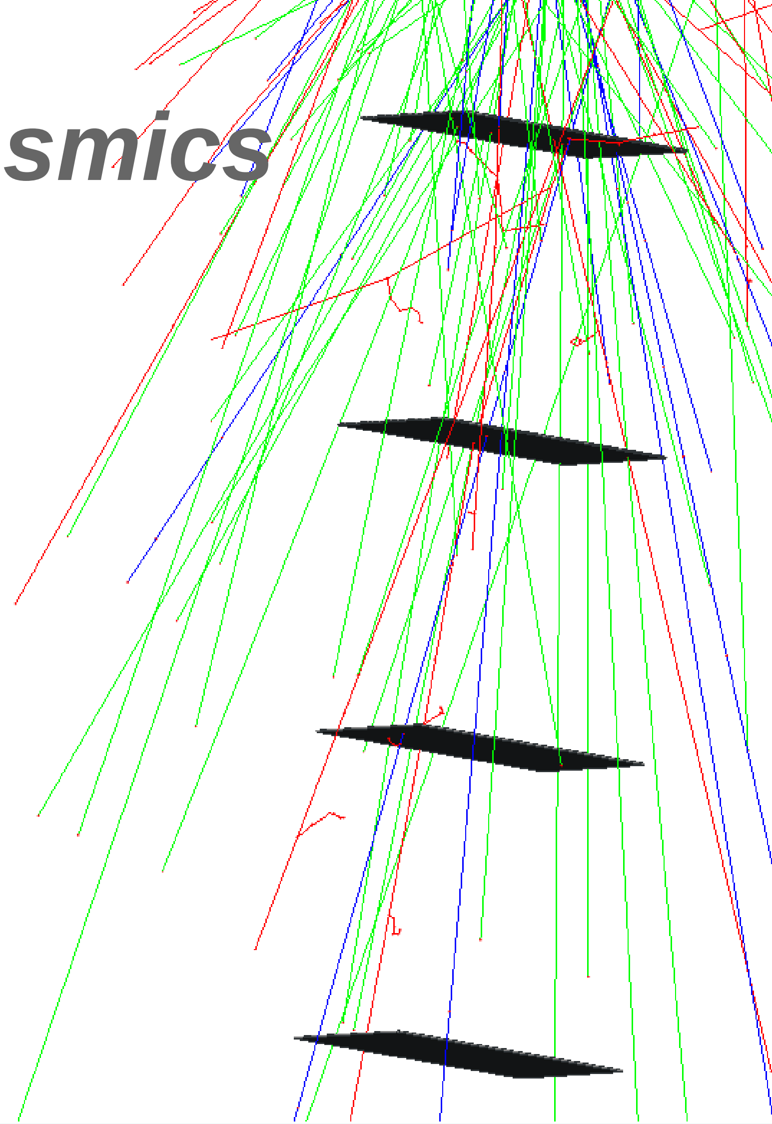
- Released in November 2021
- New module: DepositionCosmics
- Many other improvements to modules & framework

Allpix2 Version 2.2

- Released in February 2022
- Import of GDML geometries
- Custom mobility models



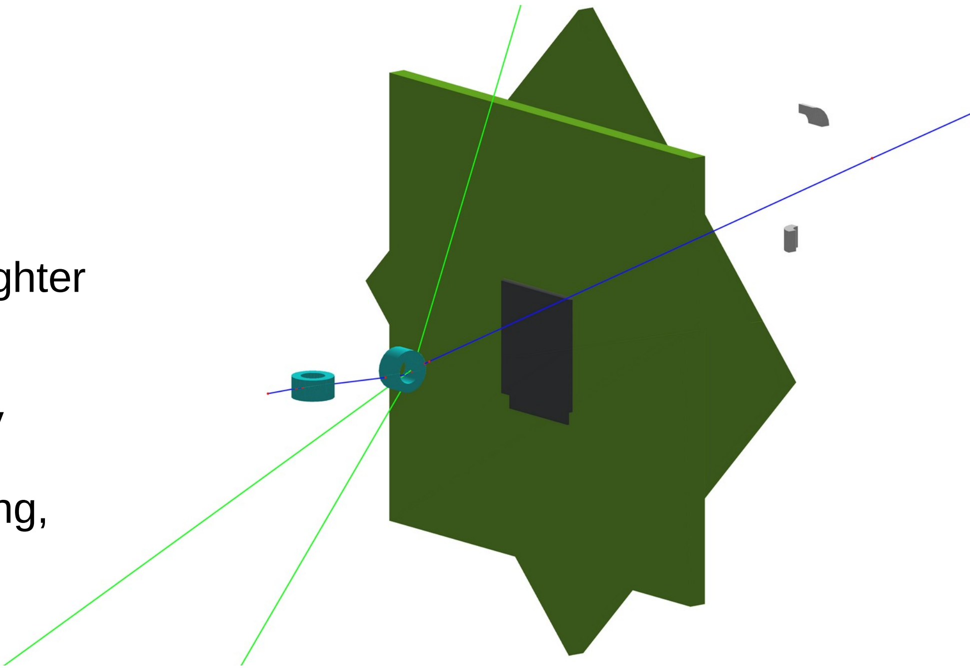
New Module: *DepositionCosmics*



- Simulation of cosmic rays with realistic particle and energy composition
- Utilises the CRY framework
  - Database of cosmic ray composition and spectrum depending on altitude, latitude and date
  - Interfaces to Geant4
  - ➔ Inherits from *DepositionGeant4* for sensor handling and energy deposition
- See application in M. Caspar's talk

Import of GDML Geometries

- GDML: Geometry Description Markup Language
  - Library of basic geometrical shapes
  - XML formatted geometry description:
    shape, dimensions, positioning and orientation
  - Features volume subtractions and mother/daughter volumes
- GDML imports are treated as passive volumes only
- Application examples: phantom definition for imaging, import of CAD models via GDML
- See presentation by F. Iguaz Gutierrez
- *Side fact:* included in the framework as first merge request on github repo



Custom Mobility Models

- From v2.0 on, mobility models are defined as individual classes and are loaded by modules
- New model: Custom mobility
- Example: replicate Jacoboni mobility model at 293 K:

```
mobility_model = "custom"
mobility_function_electrons = "[0]/[1]/pow(1.0+pow(x/[1],[2]),1.0/[2])"
mobility_parameters_electrons = 1.0927393e7cm/s, 6729.24V/cm, 1.0916
mobility_function_holes = "[0]/[1]/pow(1.0+pow(x/[1],[2]),1.0/[2])"
mobility_parameters_holes = 8.447804e6cm/s, 17288.57V/cm, 1.2081
```

- Applications: definition of custom mobility model without compilation

Other Notable Features

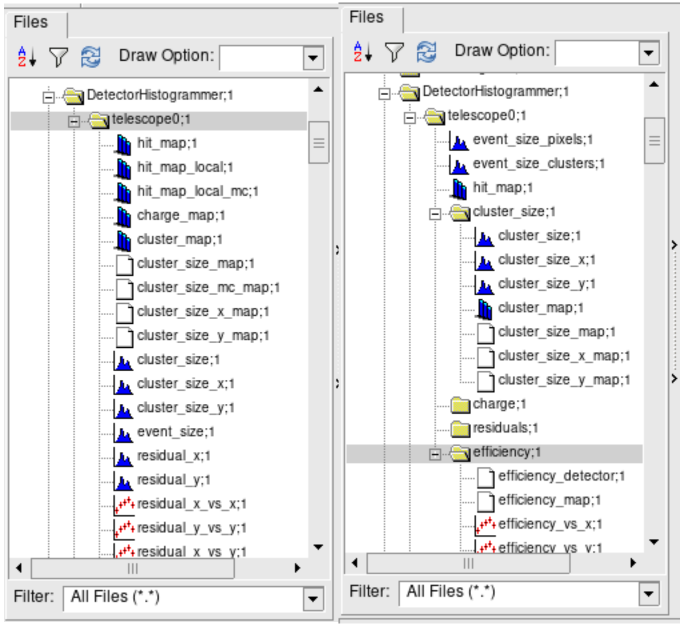
- MCTrack History:
  Add option to store *all* track objects – enables backtrace for secondary particles

- *DatabaseWriter*:
  Parallel Database Access for multithreading capability

- *DetectorHistogrammer*:
  Group histograms

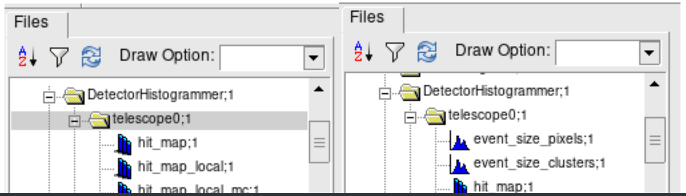
- Version & Dependency Reporting:
  allpix --version prints version of framework & dependendenies

- Many more …
  See release notes v2.1 & v2.2



Other Notable Features

- MCTrack History:
  Add option to store *all* track objects – enables backtrace for secondary particles
- *DatabaseWriter*:
  Parallel Database Access for multithreading capability
- *DetectorHistogrammer*:
  Group histograms
- Version & Dependency Reporting:
  allpix --version prints version of framework & dependendenies
- Many more …
  See release notes v2.1 & v2.2



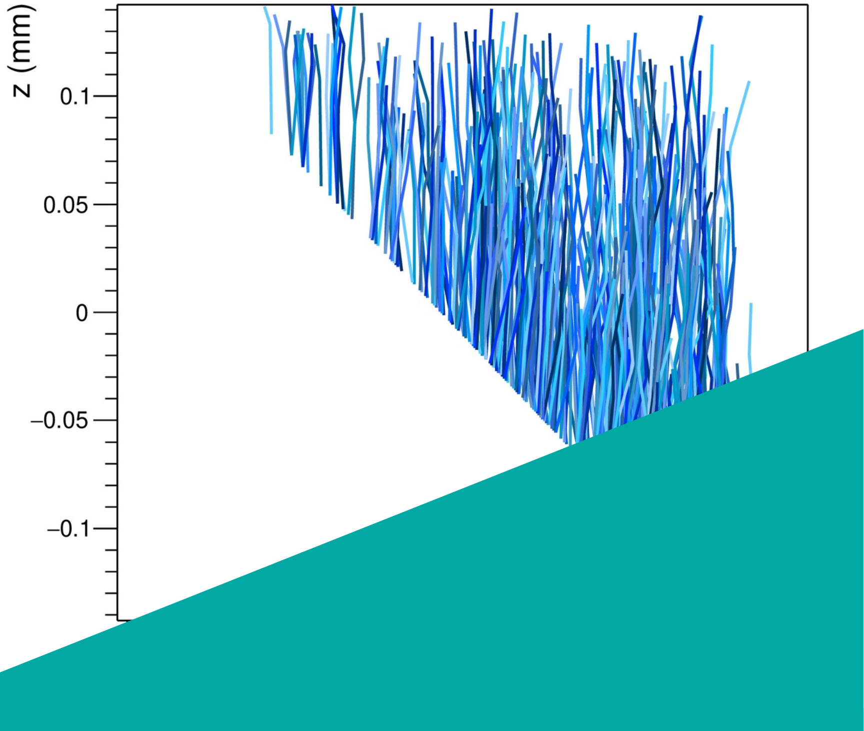

```
~/software/allpix-squared $ allpix --version
Allpix Squared version v2.0.0-927-g98404d8f6-dirty
               built on 2022-04-13, 08:22:45 UTC
               using Boost.Random 1.71.0
                     ROOT 6.24/06
                     Geant4 10.7.2
               running on 8x 11th Gen Intel(R) Core(TM) i7-1185G7 @ 3.00GHz

Copyright (c) 2016-2022 CERN and the Allpix Squared authors.

This software is distributed under the terms of the MIT License.
In applying this license, CERN does not waive the privileges and immunities
granted to it by virtue of its status as an Intergovernmental Organization
or submit itself to any jurisdiction.
```

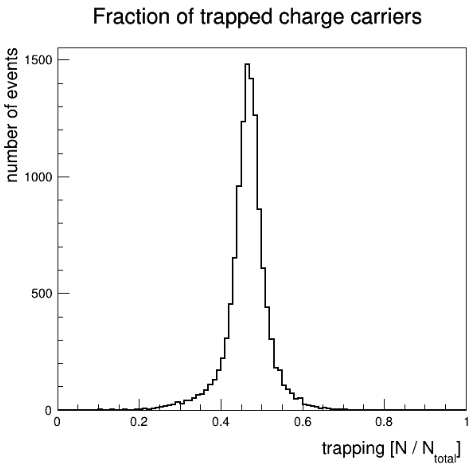
Sneak Preview

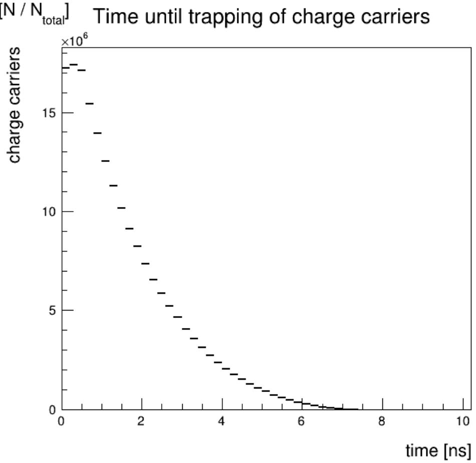
v2.3



Charge Carrier Trapping

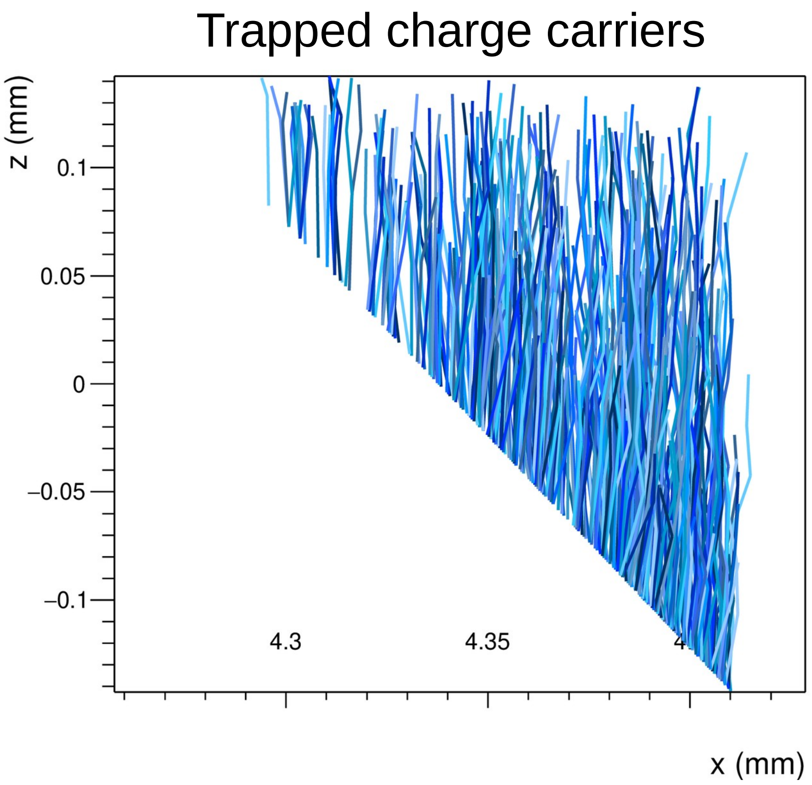
- Different trapping models implemented
  - Ljubljana / Kramberger
  - Dortmund / Krasel
  - Interpolation of CMS Tracker measurements
  - Mandic / high fluences
- Possibility to define custom trapping functions via configuration
- Scaling with fluence & temperature (where applicable)
- Note: this only describes trapping!
  Effects such as changed electric fields have to be provided separately, either through field map from TCAD or analytic approximation of E-field
- Merged: MR !624





Charge Carrier Trapping

- Different trapping models implemented
  - Ljubljana / Kramberger
  - Dortmund / Krasel
  - Interpolation of CMS Tracker measurements
  - Mandic / high fluences
- Possibility to define custom trapping functions via con
- Scaling with fluence & temperature (where applicable)
- Note: this only describes trapping!
  Effects such as changed electric fields have to be pro
  either through field map from TCAD or analytic approx
- Merged: MR !624



Sensor Materials

- The Allpix Squared ~~Silicon~~ Semiconductor Detector Simulation Framework now allows for the definition of other sensor materials than silicon
- Definition of sensor materials impacts …
  - Material in Geant4 geometry
  - Charge carrier creation energy default
  - Fano factor default
- Short list of supported materials
  ➔ New materials can easily be added by users (see FAQs in manual)
- See contribution by P. Smolyanskiy on GaAs:Cr Timepix3 detectors

Table 6.1: List of default sensor material properties implemented in Allpix^2

Material	Charge Creation Energy [eV]	Fano factor	Sources
Silicon	3.64	0.115	[25], [26]
Gallium Arsenide	4.2	0.14	[27]
Cadmium Telluride	4.43	0.24	[28], [29]
Cadmium Zinc Telluride $Cd_{0.8}Zn_{0.2}Te$	4.6	0.14	[30], [31]
Diamond	13.1	0.382	[32], [32]
Silicon Carbide (4H-SiC)	7.6	0.1	[33], [34]

Summary

- Monte Carlo simulations remain a crucial tool in the development cycle of particle detectors
- Allpix Squared sees and benefits from users from different applications & fields
  - ➔ Plenty of new features have been triggered by users' ideas or requests
- Steady development & support
  - Two feature releases 2.1 & 2.2 since 2nd Allpix Squared Workshop in 2021
  - Allpix Squared forum increasingly active – user support & bug reporting
- Plenty of new features on the horizon
  - Let's gather some more ideas in the coming three days

Allpix Squared Resources

Website
https://cern.ch/allpix-squared

List of reference publications
https://cern.ch/allpix-squared/page/publications/

Repository
https://gitlab.cern.ch/allpix-squared/allpix-squared

Docker Images
https://gitlab.cern.ch/allpix-squared/allpix-squared/container_registry

User Forum:
https://cern.ch/allpix-squared-forum/

Mailing Lists:

allpix-squared-users https://e-groups.cern.ch/e-groups/Egroup.do?egroupId=10262858

allpix-squared-developers https://e-groups.cern.ch/e-groups/Egroup.do?egroupId=10273730

User Manual:
https://cern.ch/allpix-squared/usermanual/allpix-manual.pdf