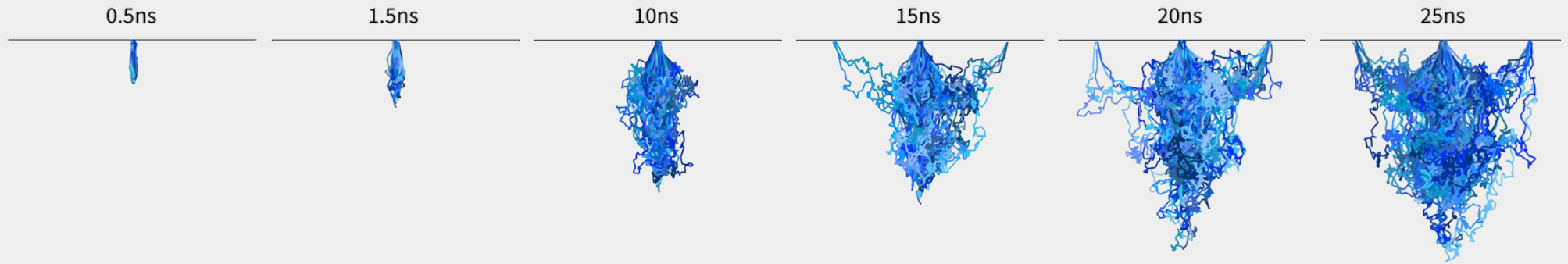




[cern.ch/allpix-squared](http://cern.ch/allpix-squared)



# What's New With Allpix Squared?

A Brief Overview of New Features, Releases and Plans

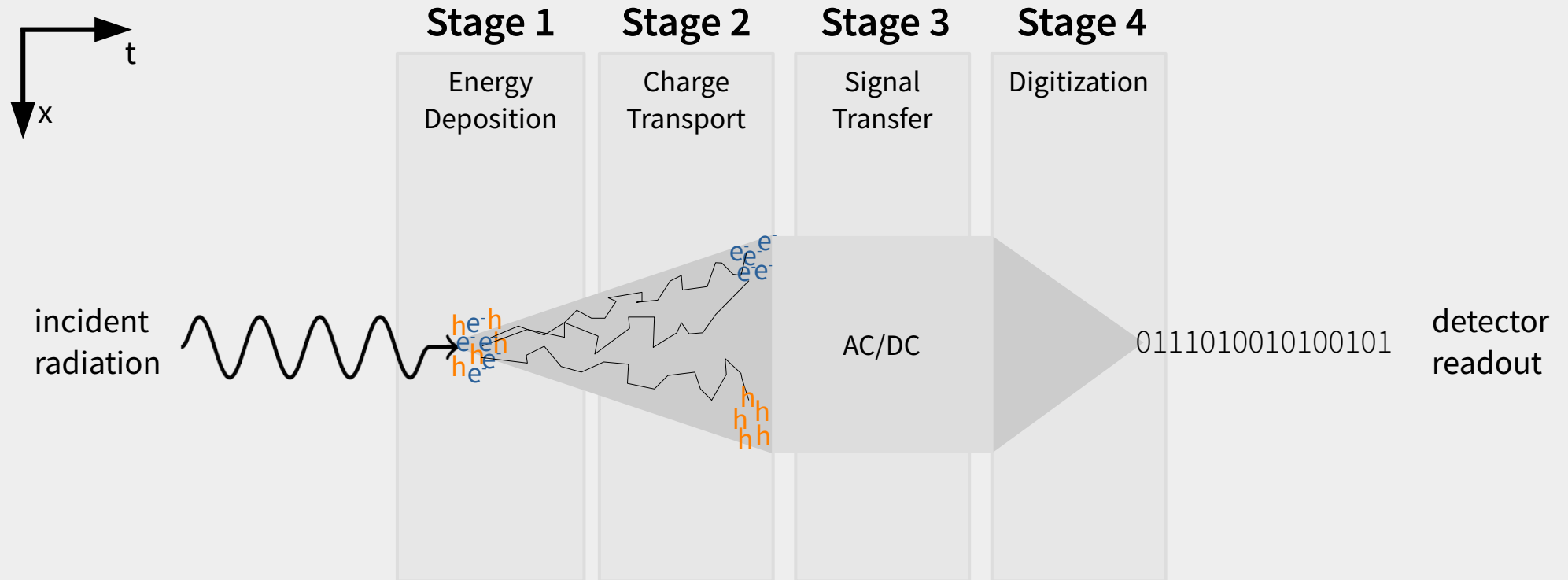
**Paul Schütze & Simon Spannagel, DESY**

9<sup>th</sup> Beam Telescopes & Test Beams

10 February 2021

Virtual Lecce

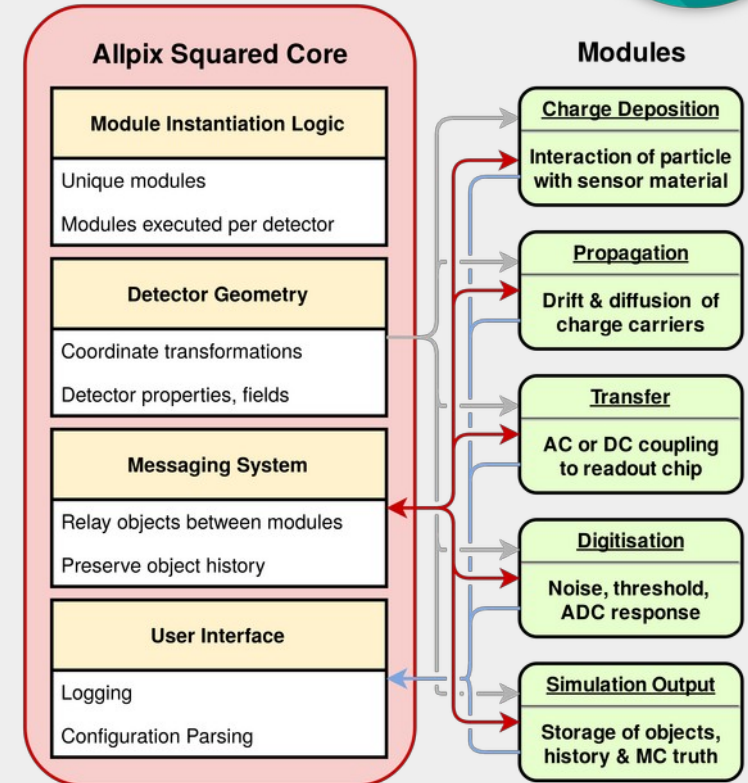
# Introduction: Silicon Detectors Monte Carlo Simulation



# Introduction: The Allpix<sup>2</sup> Framework

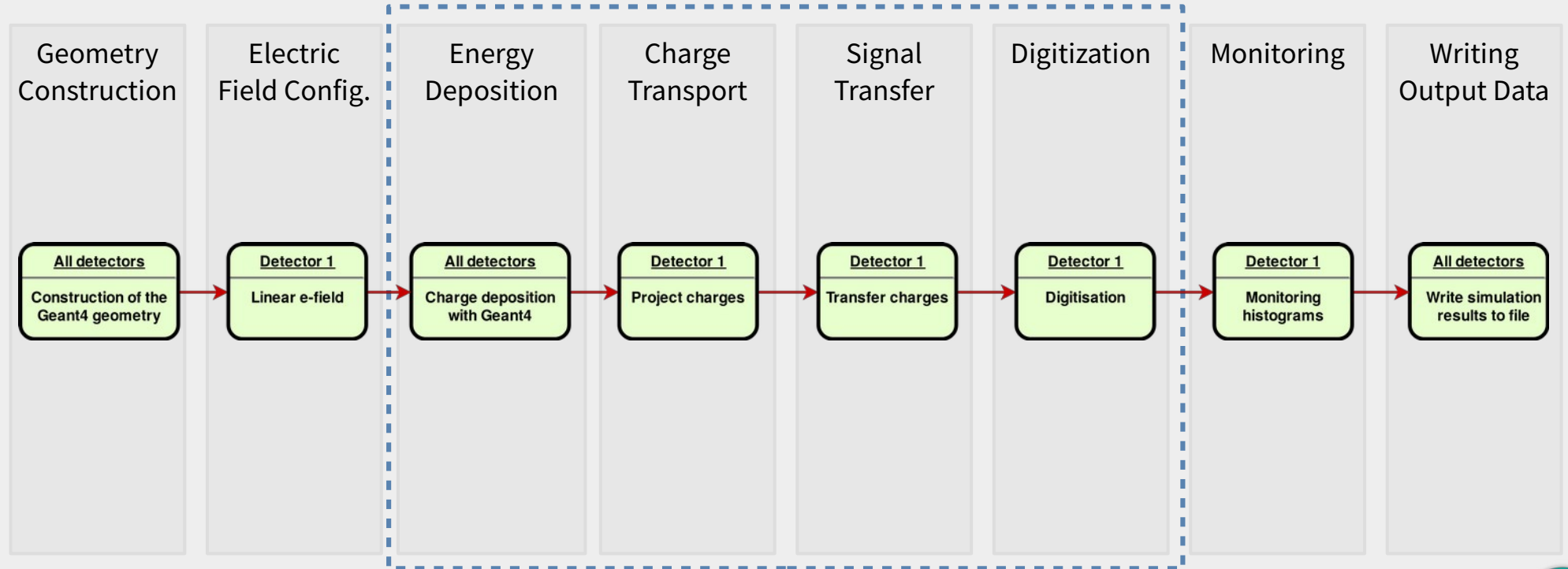


- Silicon Pixel Detector MC simulation software, that
  - ...provides a **modular tool kit** to simulate signal formation in silicon detectors
  - ...implements parametrized **detector models**
  - ...facilitates usage of **precise electric fields**
- Focus on usability & stability
  - Provide documentation (170p. [user manual](#))
  - **Regular patch & feature releases**,  
7 feature releases, 16 patch releases in 4 years
  - Community-driven, with by now **more than 30 contributors**



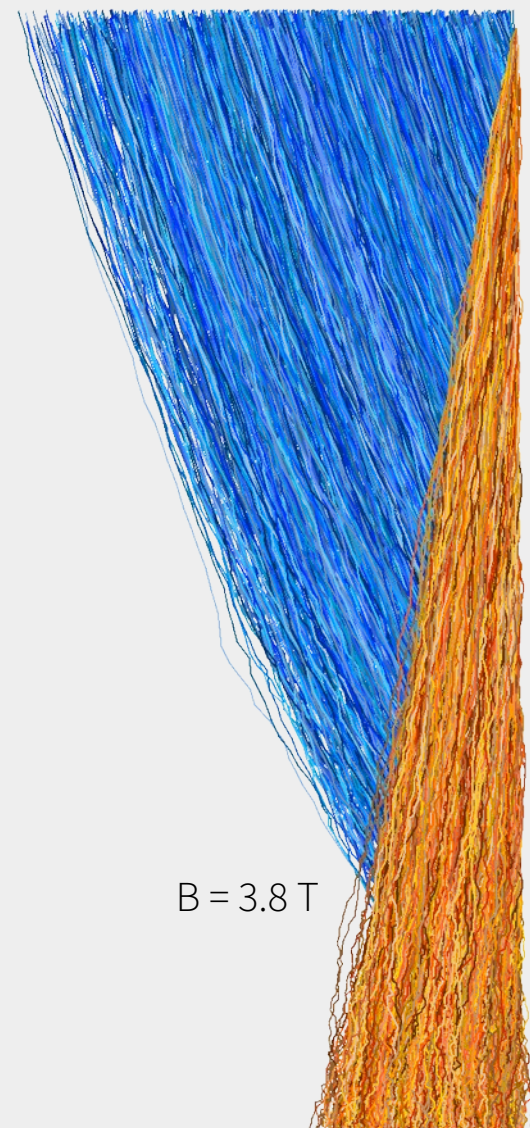
# Introduction: The Simulation Chain

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



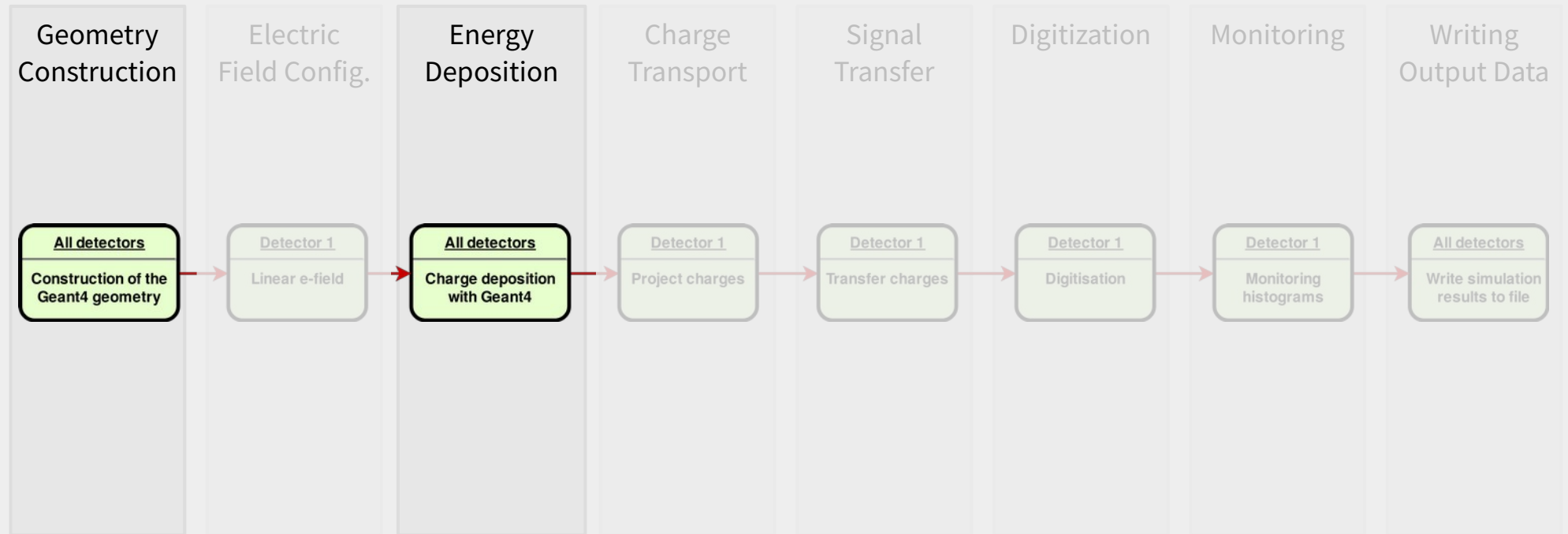
# Recently Added Features

Releases 1.4, 1.5 & 1.6



B = 3.8 T

# Geometry / Energy Deposition

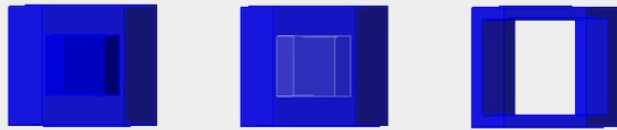


# Passive Materials: Things in the Beamline

Paul Schütze, DESY

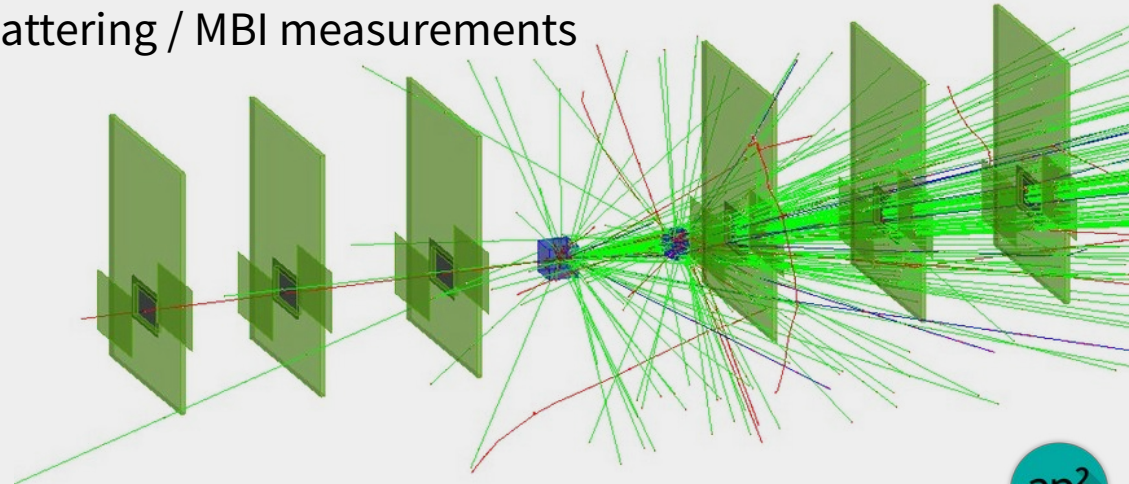
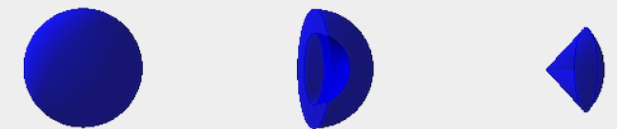
Koen van den Brandt, Nikhef

- Added possibility to define passive material in the geometry
- Different shapes, automatic merging of multiple shapes / hierarchy resolution
- Completely transparent to core framework through new parameter “*role*”



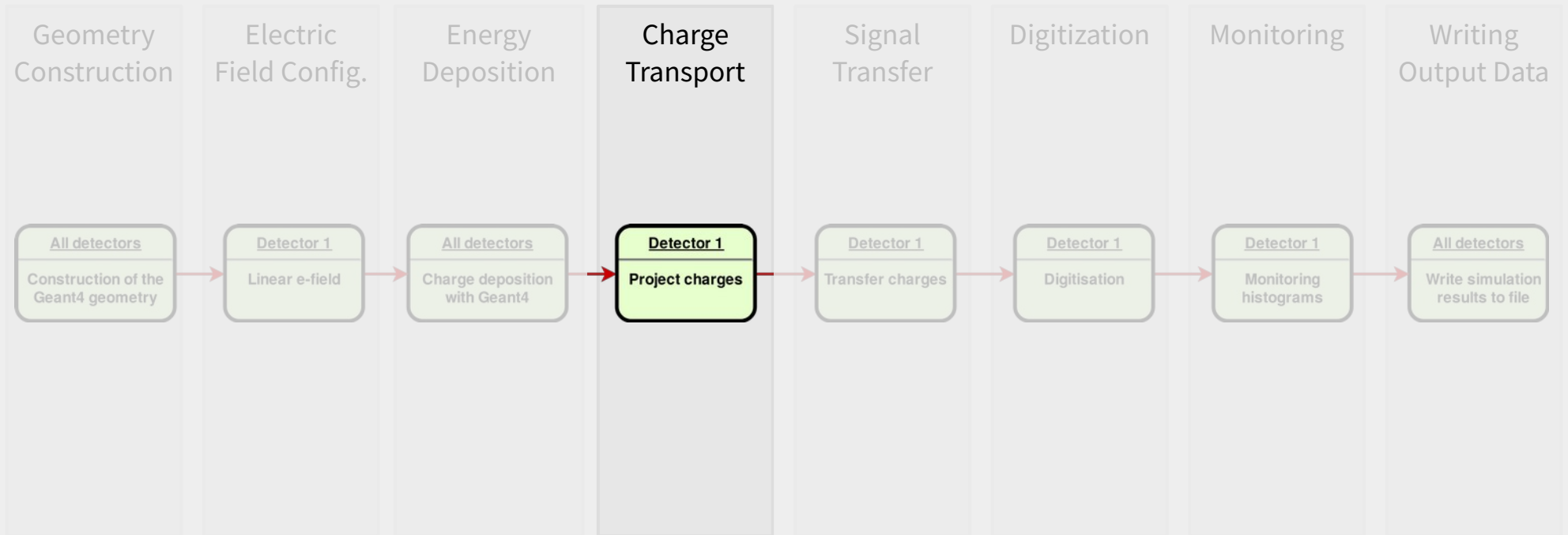
Usage examples:

- Realistic test beam setup (cooling box)
- Calorimeter simulation
- Multiple Scattering / MBI measurements



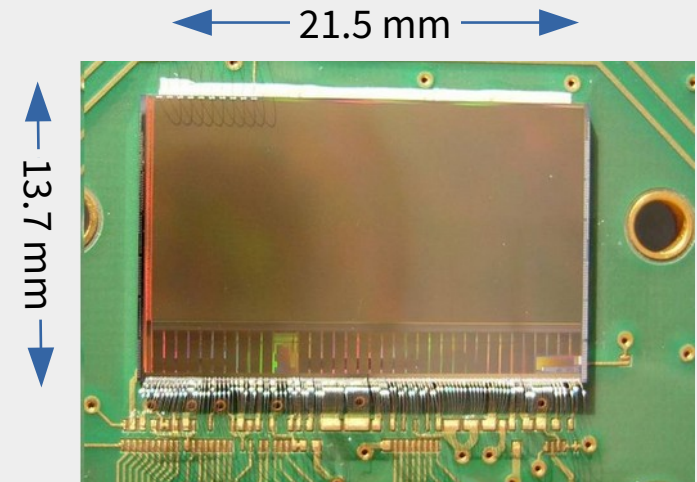
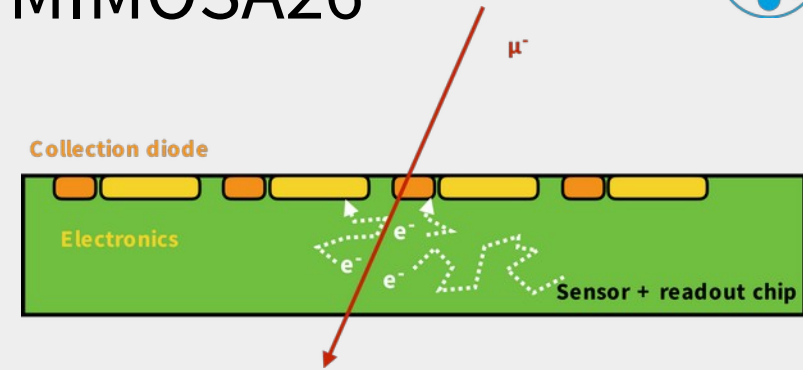


# Charge Transport



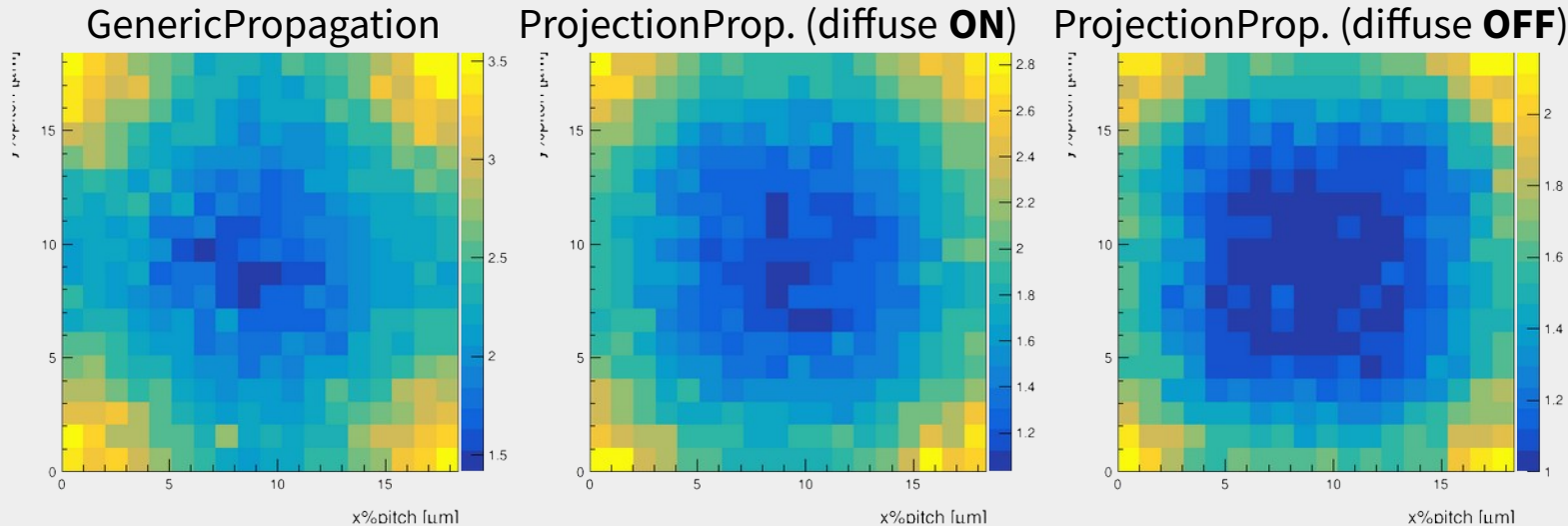
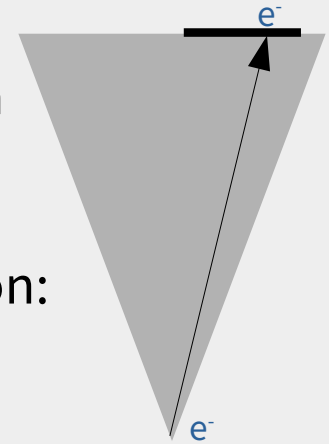
# Simulations of Telescopes @ DESY – MIMOSA26

- MIMOSA26: Monolithic active pixel sensor
    - 18.4  $\mu\text{m}$  x 18.4  $\mu\text{m}$  pitch, binary readout
    - Limited bias voltage: small depletion volume
    - Well known from beam telescopes
- used as reference detectors, simulation should be fast!



# ProjectionPropagation: Diffuse Before Project

- *ProjectionPropagation*: simplest & fastest charge transport module
  - Calculate total drift time, move to sensor surface, smear for diffusion
- Problem: does not work in partially-depleted sensors (e.g. CMOS)
- Solution: diffuse charge carriers in zero-field region before projection:

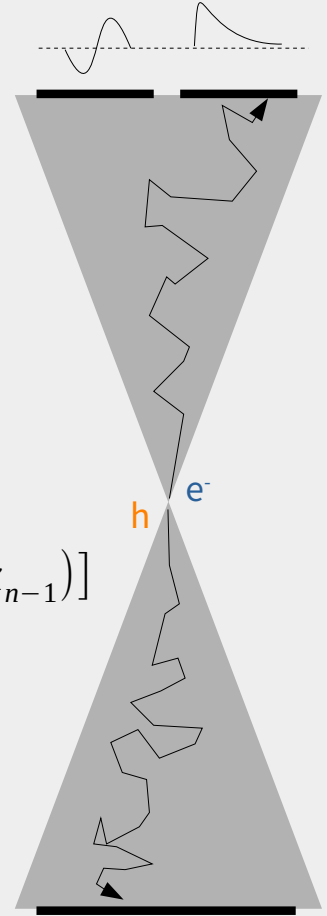


Better description of under-depleted sensors, new example in repository!

# Transient Simulation – Induced Signal at Electrodes

- Successive integration of motion, calculating induced charge per step
- Take each (group of) charge carrier
  - Calculate mobility & velocity from local fields
  - Make step, add diffusion offset from Gaussian distribution
  - Get induced charge from weighting potential difference for all neighbors
  - Repeat until sensor surface is reached
- Allows time-resolved simulation
  - Requires weighting potential, might not be trivial to obtain
- Time consuming:
  - Calculation for all neighboring electrodes for every step
  - Requires propagating both electrons and holes

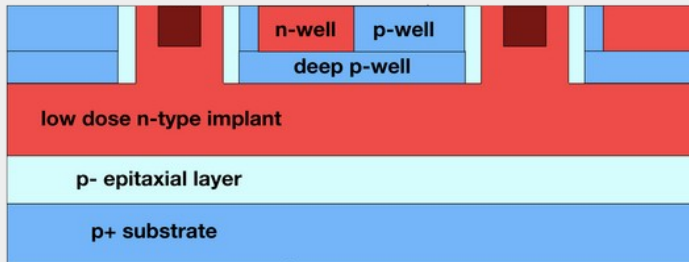
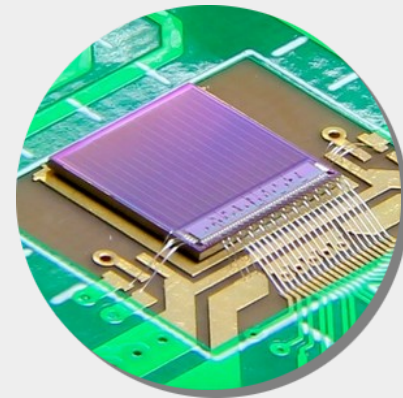
$$Q_n^{ind.} = \int_{t_{n-1}}^{t_n} I_n^{ind.} dt = q [\Phi(x_n) - \Phi(x_{n-1})]$$



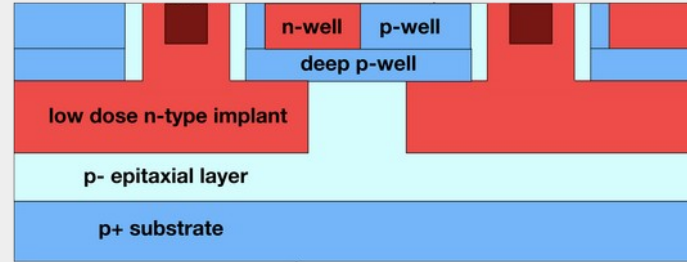
# Transient Simulations – The CLICTD MAPS Prototype

Katharina Dort, CERN  
Magdalena Munker, CERN

- Goal: understand & predict timing performance of CMOS prototypes
- Example: CLICTD for CLIC tracking detector → [Talk by Katharina yesterday](#)
  - 180 nm CMOS imaging process, small collection electrode
  - Pixel pitch:  $37.5 \mu\text{m} \times 30 \mu\text{m}$ ,  $30 \mu\text{m}$  epitaxial layer
  - Fully-integrated sensors, simultaneous ToA/ToT measurement
- Test bench for testing different sensor designs:



“continuous *n*-layer”

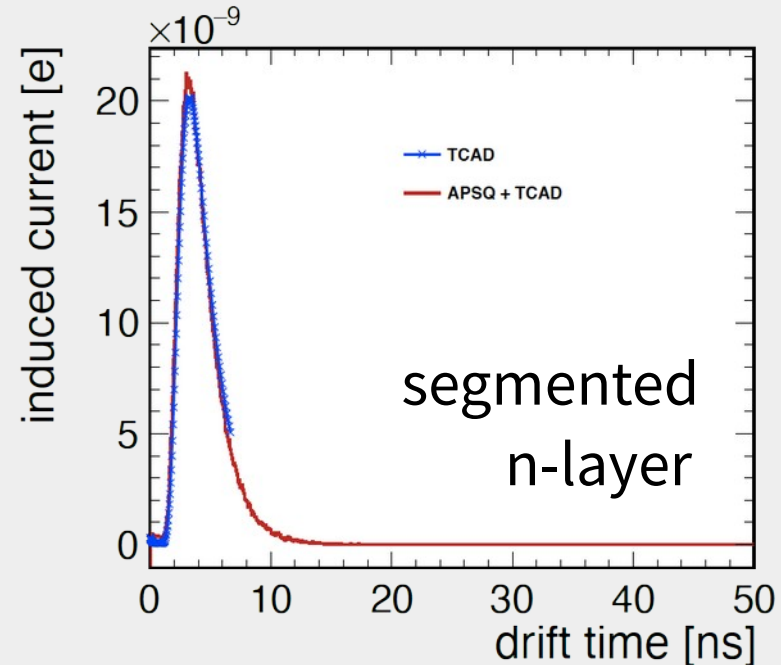
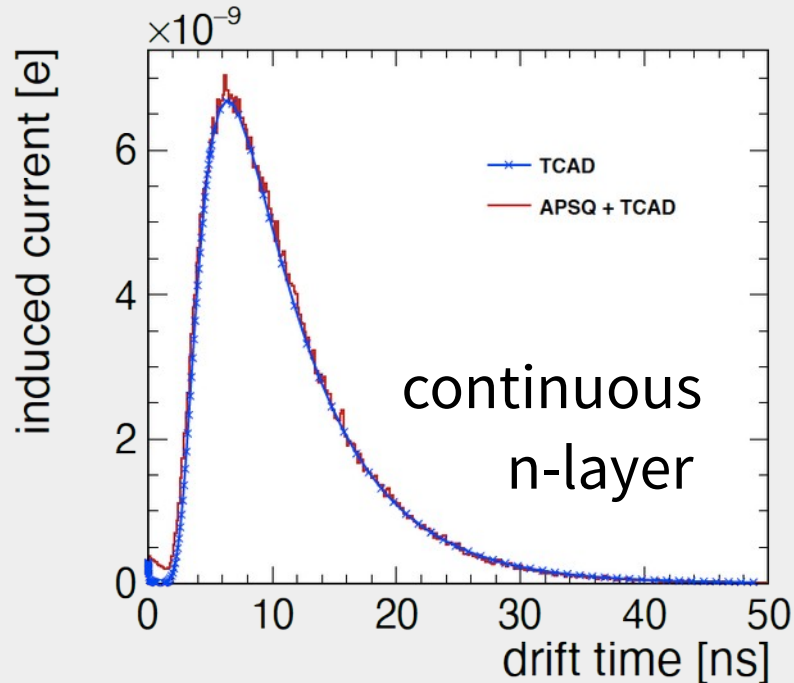


“gap in *n*-layer”

IEEE TNS, vol. 67, no. 10 (2020), 2263  
[doi:10.1109/TNS.2020.3019887](https://doi.org/10.1109/TNS.2020.3019887)

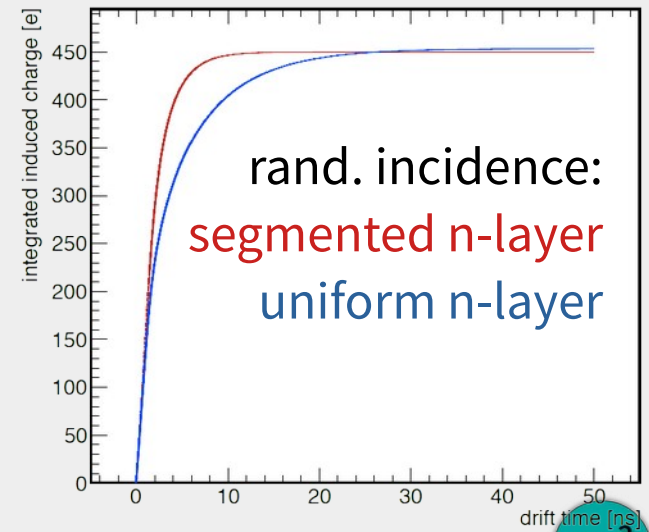
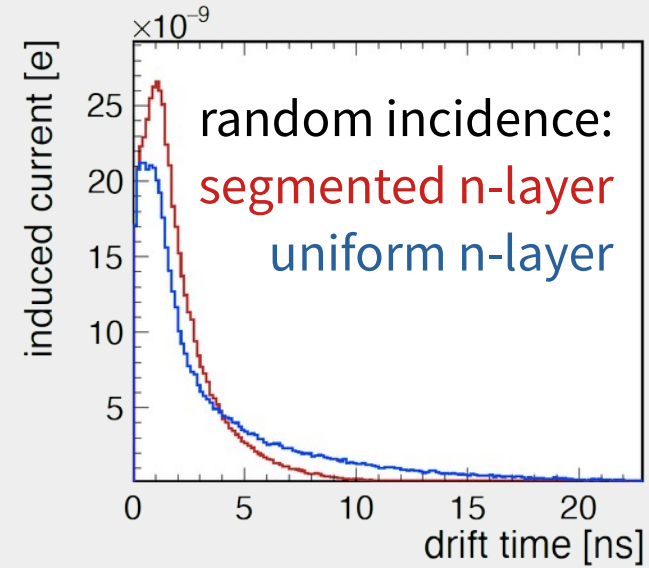
# Transient MC Simulation: Validation

- Comparison: TCAD transient / Allpix Squared transient + TCAD static
  - Comparing different CMOS sensor designs in worst-case scenario (pixel corner)
  - Uniform deposition of e.g. 63e/h per um along line to replicate TCAD transient

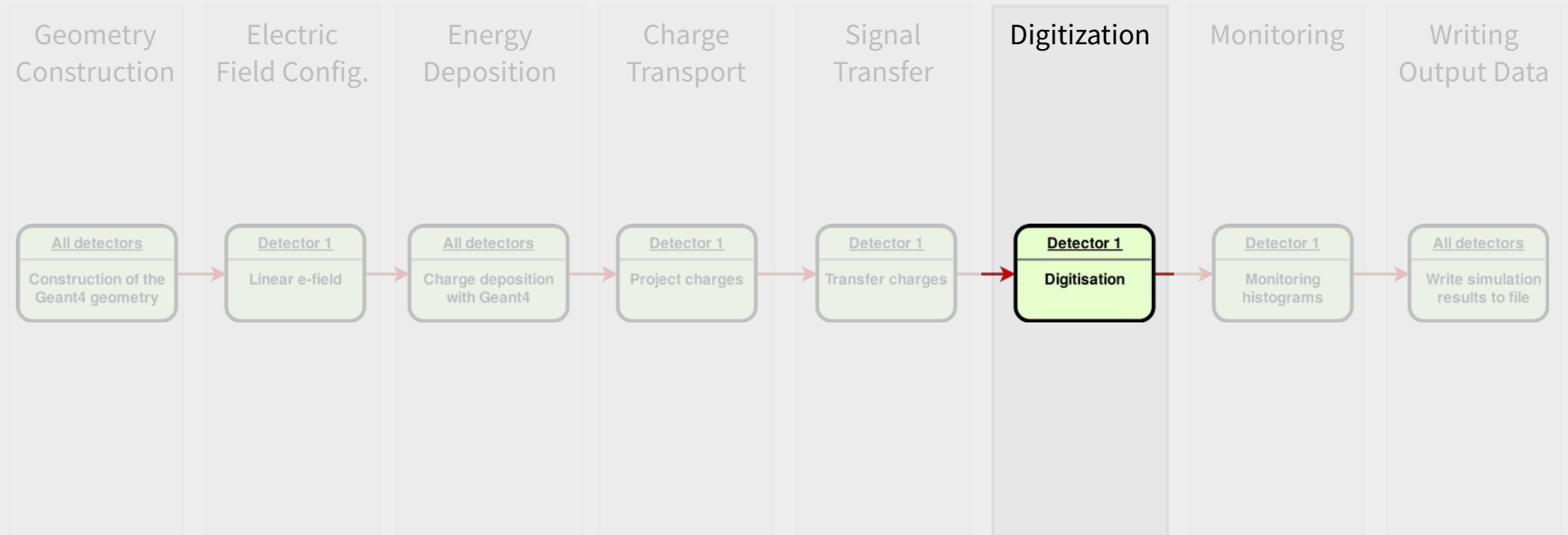


# Transient MC Simulations: Scaling Out

- Validated MC+TCAD simulation allows:
  - Random sampling of position in pixel cell
    - Too time-consuming in TCAD
    - Obtain **full picture**, not individual scenarios
  - Usage of Geant4 for **realistic performance**
    - Landau fluctuations
    - Secondaries
- Compatibility should be validated for every design
- Have seen some deviations in extreme cases, e.g. doping-dependent mobility in TCAD etc



# Digitization



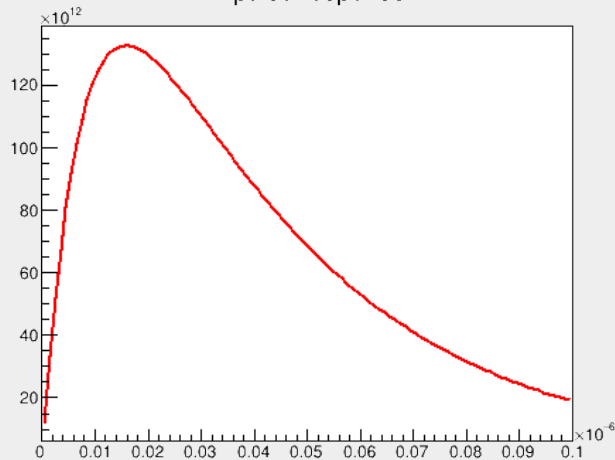


# Digitizing Pulses: CSADigitizer

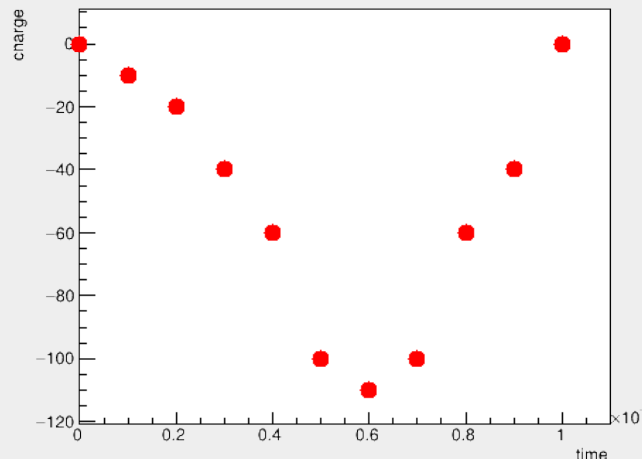
- Implementation of charge-sensitive amplifier with Krummenacher feedback, configuration via:
  - Rise time, feedback time & capacitance – “simple”
  - Detector cap., Krum. current, transconductance – “csa”
- Integrated ToT / ToA sampling on different clocks

```
[CSADigitizer]
model = "simple"
feedback_capacitance = 5e-15C/V
rise_time_constant = 1e-9s
feedback_time_constant = 10e-9 s
integration_time = 0.5e-6s
threshold = 10e-3V
clock_bin_toa = 1.5625ns
clock_bin_tot = 25.0ns
```

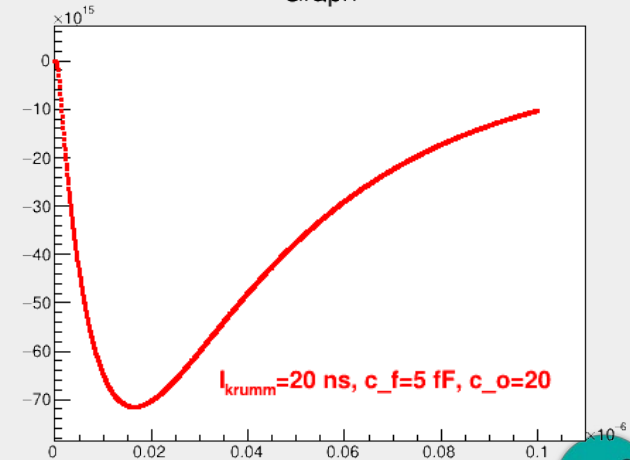
impulse response



pulse

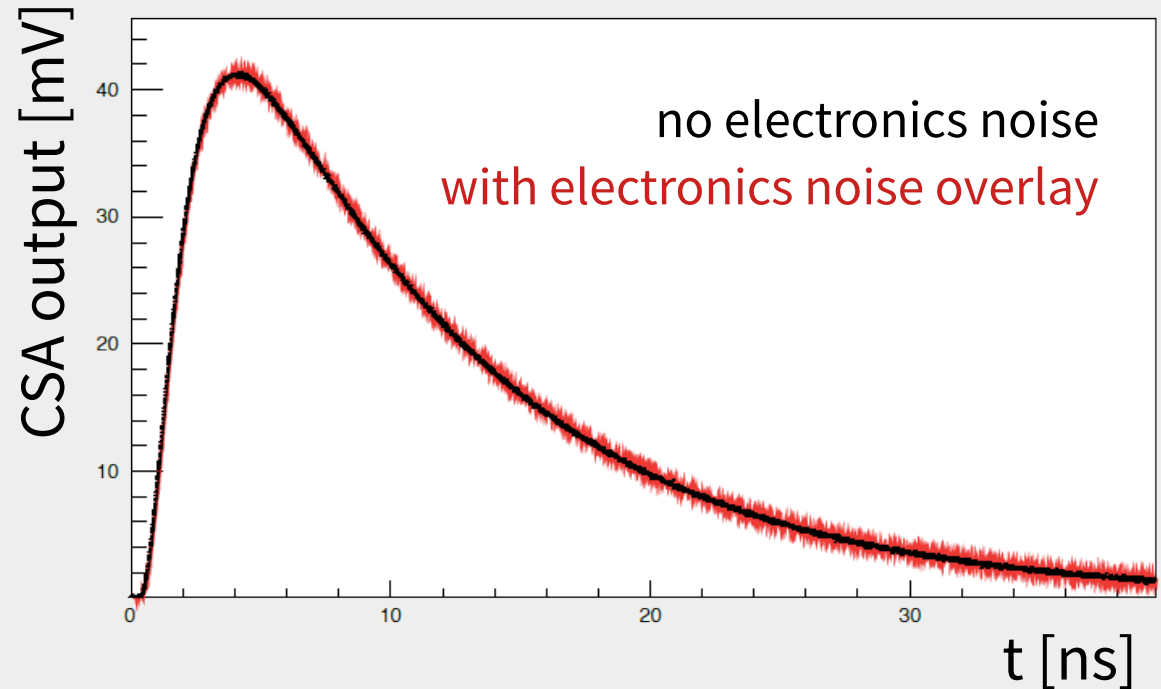


Graph



# Digitizing Pulses: CSADigitizer Example

- Single pulse of CLICTD, modified process, bias -6V/-6V at p-wells/substrate
- 5.4 GeV electron beam
- Electric field & weighting potential imported from electrostatic TCAD
- CSADigitizer with *simple* model and default parameters:
  - Rise time: 1ns
  - Feedback: 10ns



# And Much More...

## Documentation, Modules

```
Module {
    Module() {}
};

class ModuleManager;
class Messenger;

// Base constructor for unique modules
// param config Configuration for this module
Module(Configuration& config);

// Base constructor for detector modules
// param config Configuration for this module
// param detector Detector bound to this module
// Note: Detector modules should not forget to forward their detector to the base class
// \ref InvalidModuleStateException will be raised if the module failed to load
Module(Configuration& config, std::shared_ptr<Detector> detector);

// virtual destructor.
// Note: This destructor has all delegates linked to this module
Module::~Module();

// Note: Copying a module is not allowed
Module(const Module&) = delete;
Module(const Module&) const = delete;

// Note: Move behaviour (not possible with references)
Module(Module&) = delete;
Module(Module&) noexcept = delete;
```



# Many New Simulation Modules

- DepositionPointCharge:
  - Deposit energy at a single point or along line, e.g. for comparison with TCAD
- DepositionReader
  - Generate energy depositions externally (e.g. full-experiment G4 simulation)
  - Read deposited energy from file and dispatch for configured detectors
- DatabaseWriter (Enrico Jr. Schioppa, Uni Salento)
  - Write simulation result directly into PostgreSQL databases



# Some Selected Outside-HEP Application Highlights

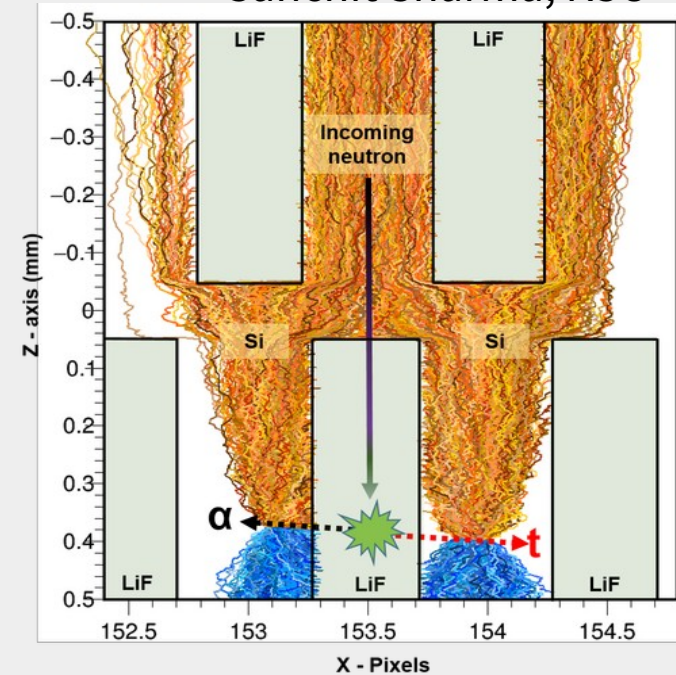


- Outside particle physics
  - NASA / Space Radiation Analysis Group
    - ISS radiation monitor simulations
  - Kansas State University
    - Silicon neutron detector with LiF trenches
- Education
  - EDIT Detector School & Bonn-Cologne Graduate School
    - Lab exercise on resolution of pixel detectors
  - Beamline for Schools 2019
    - Simulation of beam telescope with absorbers
  - Uni Dortmund
    - Bachelor thesis on time-of-flight measurements



**KANSAS STATE**  
UNIVERSITY

Sanchit Sharma, KSU



# A Word on Writing Code for MC Simulations

- Implementation of algorithms is not the most time-consuming part
- Most time-consuming part is to do it such, that the algorithms are...
  - ...validated with prototype data & device simulations
  - ...well documented
  - ...maintainable over a period longer than O(1 fellow) / O(1 PhD)
- Development of Allpix<sup>2</sup>: we spend considerable time on
  - Writing documentation → lower barrier for new users
  - Implementing automated testing, compilation → ensure software always works
  - Code review for new features → ensure functionality/compatibility

# User Manual & Code Documentation

- Focus from the very beginning on well-documented framework
- Source code documentation for every class, method
  - Doxygen markup for code reference
  - Deployed to the website for tags
- Extensive User Manual in LaTeX
  - Automatically compiled by CI
  - Module documentation as Markdown
    - Document module parameters, algorithms
    - Included in manual via Pandoc

```
namespace allpix {  
  
    /**  
     * @brief Instantiation of a detector mode  
     *  
     * Contains the detector in the world with  
     * (like the electric field). All model sp  
     * properties are stored in its DetectorMo  
     */  
    class Detector {  
        friend class GeometryManager;  
  
    public:  
        /**  
         * @brief Constructs a detector in the  
         * @param name Unique name of the dete  
         * @param model Model of the detector  
         * @param position Position in the wor  
         * @param orientation Rotation matrix  
         */  
        Detector(std::string name,  
                std::shared_ptr<DetectorModel  
                ROOT::Math::XYZPoint position  
                const ROOT::Math::Rotation3D&  
  
        /**  
         * @brief Get name of the detector  
         * @return Detector name  
         */  
        std::string getName() const;
```

### GenericPropagation

Maintainer: Koen Wilms (kwilms@cern.ch), Simon Spannagel (spannagel@cern.ch)  
Status: Functional  
Input: EvolvedCharge  
Output: PropagatedCharge

**Description**  
Simulates the propagation of electrons and 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 interactions are 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 determined from the charge carrier mobility parameter  $\mu$ , and the electric field  $E$ . The carrier mobility for other electrons or holes is automatically chosen based on the type of the charge carrier under consideration. This, along with both electrons and holes is treated equally.

The two parameters, `propagate_electrons` and `propagate_holes` allow to control which type of charge carrier is propagated to the respective electrodes. Other than 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 configuration, and should be ensured that the carrier types selected are actually transported to the relevant side for given electric fields, a warning is issued if a possible recombination 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 from a Gaussian distribution calculated from the Einstein relation:

$$\sigma = \sqrt{2Dt}$$

using the carrier mobility  $\mu$ , the temperature  $T$  and the time step. The propagation steps when the set of charges reaches any of the edges.

The propagation module also produces a variety of output plots. These include a 3D layout 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 the coloring scheme, electrons are marked in blue colors, while holes are presented in different shades of orange. In addition, a 2D GC simulation for the drift of all individual sets of charges (with the size of the plot proportional to the number of charges in the set) can be produced. Finally, the module produces 2D contour simulations 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 avoided if even when using `propagate` below.

**Dependencies**  
This module requires an installation of Diges.

**Parameters**

- `temperature`: Temperature of the sensitive device, used to estimate the diffusion constant and therefore the strength of the diffusion. Default is room temperature (300.15K).
- `charge_per_step`: Maximum number of charge carriers to propagate together. Overrides the total number of deposited charge carriers at a specific point in time if the number of charge carriers is not 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 use for the reweighting of the Runge-Kutta propagation to adjust to reaching the spatial precision after calculating the necessary force from the fifth-order error method. Default is 0.1 cm.
- `time_step_start`: Time step to initiate the Runge-Kutta integration with. An aggressive estimation of the parameter reduces the time to reach the spatial precision parameter. Default value is 0.1 ns.
- `time_step_min`: Minimum step time to use for the Runge-Kutta integration regardless of the spatial precision. Default is 1 ps.
- `time_step_max`: Maximum step time to use for the Runge-Kutta integration regardless of the spatial precision. Default is 0.1 ns.
- `convergence_cuts`: Time with which charge carriers are propagated. After exceeding this time, no further propagation is performed for the respective carriers. Default is the LHC bunch crossing time of 25 ns.
- `propagate_electrons`: Select whether electron-type charge carriers should be propagated to the electrodes. Default is true.
- `propagate_holes`: Select whether hole-type charge carriers should be propagated to the electrodes. Default is false.
- `output_plots`: Determine 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. Default is false.
- `output_plots_line`: Tendency to connect points plotted indirectly. Determines the amount of points plotted. Default is reverse, `none` if not explicitly specified.
- `output_plots_theta`: Viewpoint angle of the 3D animation and the 2D line graph around the world Z-axis. Default is zero.
- `output_plots_phi`: Viewpoint angle of the 3D animation and the 2D line graph around the world X-axis. Default is zero.
- `output_plots_use_pixel_units`: Determine if the plots should use pixels as unit instead of metric length scales. Default is false (this using the metric system).
- `output_plots_use_equal_scaling`: Determine 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). Default is true.
- `output_plots_align_pixels`: Determine if the plot should be aligned on pixels. Default is false if enabled the start and the end of the axis will be at the edge pixel between pixels.
- `output_animation_line`: In addition to the other output plots, also write a GIF animation of the charge drifts towards the electrodes. This is very slow and writing the animation takes a considerable amount of time, therefore default is false. This option also requires `output_plots` to be enabled.
- `output_animation_line_scaling`: Scaling for the animation used to convert the actual animation time to the time step the animation default is 1:1, meaning that every second of the animation requires an animation of a single second.
- `output_animation_color`: Scaling for the markers on the animation. Default is one. The markers are already internally scaled to the charge of the step, normalized to the maximum charge.
- `output_animation_color_use_scaling`: Scaling to use for the color scale 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 (in units) per volume above the one is depicted in the same maximum color. Parameter can be used to improve the color scale of the contour plots.
- `output_animation_color_markers`: Determine if colors should be for the markers in the animations. Default is false.

**Usage**  
An example of generic propagation for all sensors of type "Trapez" at room temperature using pixels of 25 charges in the following:

```
{GenericPropagation  
  type = "Trapez"  
  temperature = 300K  
  charge_per_step = 25
```

# Google Season of Docs

...is **not** Google Summer of Code!

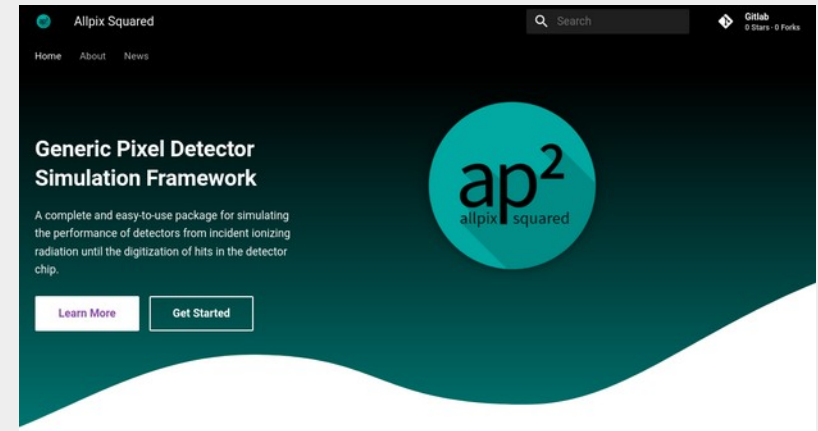
- Scholarship for experienced technical writers to work on documentation of open source projects
- Allpix Squared is participating through HEP Software Foundation
- Technical Writer **Sabita Rao** worked for three months on documentation/website
  - Goal: revision of online appearance
  - Focus on integration of online user manual
  - Improvements to tutorials/examples
- Finalization still pending



Sabita Rao

Paul Schütze, DESY

Simon Spannagel, DESY





In a nutshell...



# Summary

- Allpix Squared continues to be developed & used by broad community
- Many new features added recently
  - Passive materials can be added to geometry
  - Better treatment of partially depleted sensors in fast simulation
  - Transient simulations
  - Digitization with a charge-sensitive amplifier
  - Many new modules
- Several ongoing projects

## Get involved!

# Allpix Squared Resources



Website

<https://cern.ch/allpix-squared>



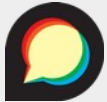
Repository

<https://gitlab.cern.ch/allpix-squared/allpix-squared>



Docker Images

[https://gitlab.cern.ch/allpix-squared/allpix-squared/container\\_registry](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>

