Status Report

SiD simulation/ reconstruction in DD4HEP

Aidan ROBSON and Dan PROTOPOPESCU - GLASGOW, UK

LCUK Meeting | Liverpool, February 2016

SiD simulation

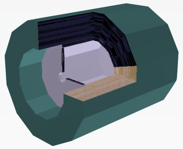
SiD Background

SiD has been using the SLIC framework (from SLAC) for simulation. Support effort has been reducing and at LCWS15 the collaboration decided to switch to the new DD4HEP framework, along with CLICdp and ILD.

-> plan is to use this moving towards TDR.

Glasgow is supporting this transition based on our existing CLICdp DD4HEP involvement.

DD4hep Implementation/validation



The Glasgow Linear Collider Group is working with SiD software coordinator (Jan Strube) to lead the conversion of the SiD model from Mokka to DD4hep. We are coordinating our activity with the wider SiD developer community (bi-weekly meetings).

What we are doing:

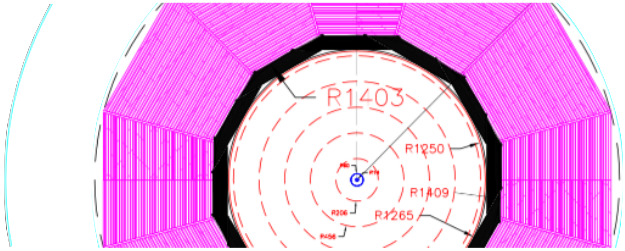
1) verifying the SiD geometry ported from Mokka
2) updating code to the latest SiD model
3) validating the new detector implementation
4) maintaining software and configuration compatibility with the latest DD4hep/DDSim versions
5) providing the validated model to the detector optimisation group
6) goto 2) if geometry or software changes are proposed

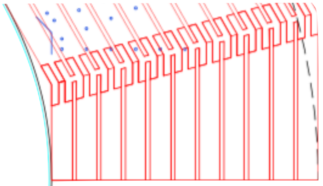
Along the way

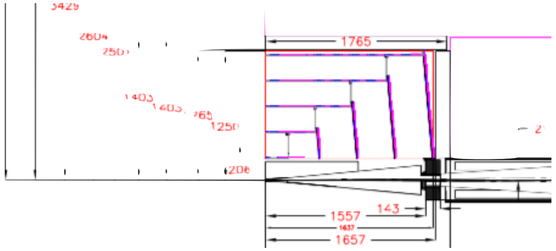
We incorporate detector geometry and spec changes as the SiD model evolves.

We document every step of the process for other users/developers within the SiD and the wider LC communities.

Maintain contact with ILC, CLIC, ILD developers.







SiD_o1_v01

All sub-detectors are built at the moment using generic DD4hep drivers. Certain drivers will be customised to include the desired level of detail once the DDReco performance of our implementation is assessed.

SiD Subcomponent	XML	Driver
Vertex Barrel	LoI3/in work	DD4hep_SiTrackerBarrel
Vertex Endcap	LoI3/in work	DD4hep_SiTrackerEndcap2
Tracker Barrel	LoI3/in work	DD4hep_SiTrackerBarrel
Tracker Endcap	LoI3/in work	DD4hep_SiTrackerEndcap2
ECal Barrel	EcalBarrel_o1_v01_00	GGenericCalBarrel_o1_v01
ECal Endcap	EcalEndcap_o1_v01_00	GGenericCalEndcap_o1_v01
HCal Barrel	HcalBarrel_o1_v01_00	GGenericCalBarrel_o1_v01
HCal Endcap	HcalEndcap_o1_v01_00	GGenericCalEndcap_o1_v01
Muon Barrel	LoI3/in work	DD4hep_PolyhedraBarrelCalorimeter2
Muon Endcap	LoI3/in work	DD4hep_PolyhedraEndcapCalorimeter2
LumiCal	LumiCal_o1_v01_00	DD4hep_CylindricalEndcapCalorimeter
BeamCal	BeamCal_o1_v01_00	DD4hep_ForwardDetector
Beam pipe	LoI3/in work	DD4hep_PolyconeSupport/TubeSegment
Supports and readout	LoI3/in work	none

■ ▣ Mokka port, ■ ▣ updated XML, ■ ▣ DD4hep generic drivers, ■ ▣ Generic ILD/CLIC/SiD drivers that include Pandora extensions

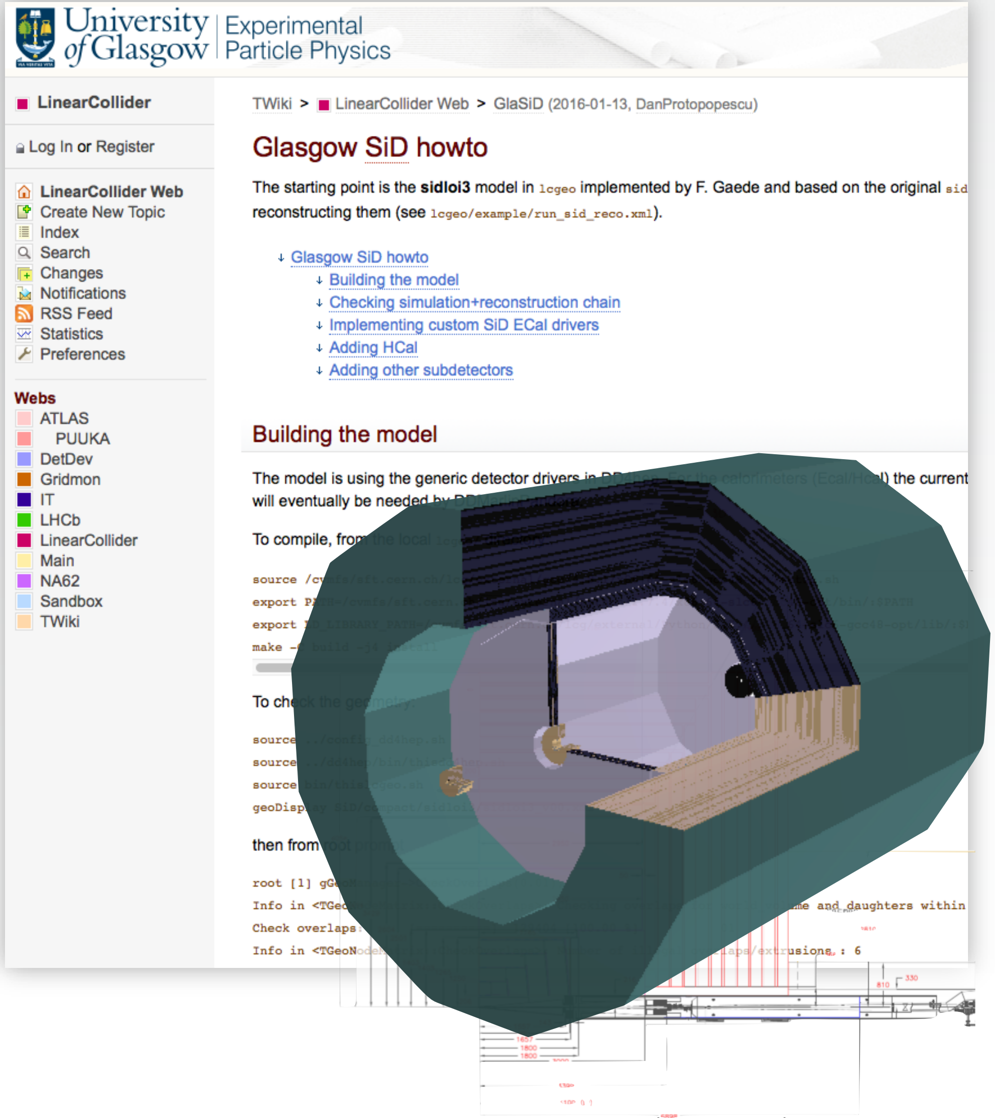
Current status

Done:

- geometry ported from Mokka by F. Gaede
- generic DD4hep drivers used for all subdetectors
- updated XMLs to the latest variable structure and naming conventions
- checked all numbers against the latest engineering drawings
- documented everything on our wiki

To do:

- finalise DD4hep implementation of all SiD subdetectors
- customise C++ drivers to include more detailed geometry
- provide code for physics simulations and optimisation studies
- update everything if/when software and/or geometry are modified
- document everything along the way



Glasgow SiD howto

The starting point is the **sidloi3** model in `lcgeo` implemented by F. Gaede and based on the original `sid`

reconstructing them (see `lcgeo/example/run_sid_reco.xml`).

- Glasgow SiD howto
  - Building the model
  - Checking simulation+reconstruction chain
  - Implementing custom SiD ECal drivers
  - Adding HCal
  - Adding other subdetectors

Building the model

The model is using the generic detector drivers in DD4hep. For the calorimeters (Ecal/Hcal) the current will eventually be needed by DDMarlinPandora

To compile, from the local `lcgeo` directory

```
source /cvmfs/sft.cern.ch/l
export PATH=/cvmfs/sft.cern.
export LD_LIBRARY_PATH=
make -C build -j4 install
```

To check the geometry:

```
source ../config_dd4hep.sh
source ../dd4hep/bin/thisdd4hep.sh
source bin/thislcgeo.sh
geoDisplay SiD/compact/sidloi
```

then from root prompt

```
root [1] gGeo
Info in <TGeo                                 me and daughters within
Check overlaps:
Info in <TGeoNode                                 aps/extrusions : 6
```

-> separately the CLICdp group is working intensively on the reconstruction

Aim is to have model complete and reconstruction working ~in the summer
LC s/w workshop in DESY later in February

BACKUP

Geometry implementation workflow

Obtaining the latest geometry

The detector geometry is still being optimised, so it's a work in progress.

The starting point is the geometry implemented in Mokka, and the most recent engineering drawings.

Implementing the latest SiD model in DD4hep

This means updating or rewriting:

1) the C++ drivers that actually build the detector modules
2) XML description(s)
3) compatibility with latest DDSim features

Validating the newly implemented SiD

This is done by:

1) visual comparison with a previous implementation
2) checking overlaps
3) comparing simple particle tracks
4) full physics simulations

Committing the changes to DESY SVN and/or Github

Central repository where:

1) latest geometries can be stored
2) subdetector conflicts can be checked
3) compatibility with other DD4hep software modules is tested

Geometry implementation workflow

Obtaining the latest geometry

The detector geometry is still being optimised, so it's a work in progress.

The starting point is the geometry implemented in Mokka, and the most recent engineering drawings.

Implementing the latest SiD model in DD4hep

This means updating or rewriting:

1) the C++ drivers that actually build the detector modules
2) XML description(s)
3) compatibility with latest DDSim features

Validating the newly implemented SiD

This is done by:

1) visual comparison with a previous implementation
2) checking overlaps
3) comparing simple particle tracks
4) full physics simulations

At the moment, all new code is committed to the DESY SVN repository, where the core DD4hep software and the CLIC and ILD models are stored as well. The working version is labelled **SiD_o1_v01**. The XML configuration is located in `lcgeo/SiD/compact/SiD_o1_v01` and the C++ drivers are located in `lcgeo/detector/`