Experience from running the CEPCSW

this slide is not a complete guide but just telling my first touch on the current CEPCSW developed by the CEPCSW software team.
In short, it shows how I use it recently.

05/21/2021

Setup

- Central repository

  https://github.com/cepc/CEPCSW

here, you need to fork first

- Procedure to run (from the README)

Quick start

SSH to lxslc7 (CentOS 7).

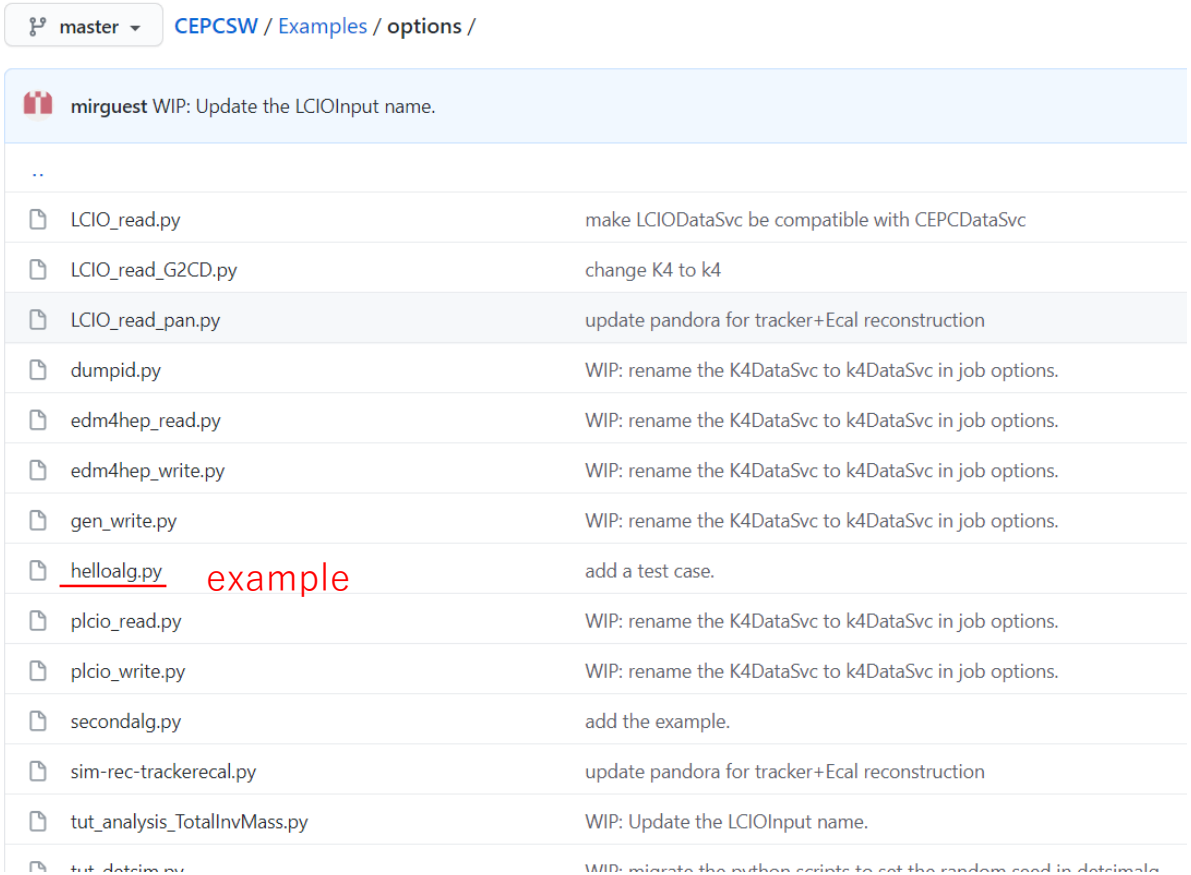
Before run following commands, please make sure you setup the CVMFS:

```
$ git clone git@github.com:cepc/CEPCSW.git
$ cd CEPCSW
$ git checkout master # branch name
$ source setup.sh
$ ./build.sh
$ ./run.sh Examples/options/helloalg.py
```

Run Script

- Many examples scripts can be found under “Examples/options/”

- We can refer those and prepare our own one



Settings in the run script

- Take an example script from “sdt” branch (To where I have uploaded the script recently)

-- (sdt)/Examples/options/fit_DC+Si.py

```
#!/usr/bin/env python
from Gaudi.Configuration import *

from Configurables import k4DataSvc
dsvc = k4DataSvc("EventDataSvc")

from Configurables import RndmGenSvc, HepRndm__Engine_CLHEP__RanluxEngine_
seed = [10]
# rndmengine = HepRndm__Engine_CLHEP__RanluxEngine_() # The default engine in Gaudi
rndmengine = HepRndm__Engine_CLHEP__HepJamesRandom_("RndmGenSvc.Engine") # The default engine in Geant4
rndmengine.SetSingleton = True
rndmengine.Seeds = seed

rndmgensvc = RndmGenSvc("RndmGenSvc")
rndmgensvc.Engine = rndmengine.name()

geometry_option = "CRD_o1_v01/CRD_o1_v01.xml"

if not os.getenv("DETCRDROOT"):
    print("Can't find the geometry. Please setup envvar DETCRDROOT." )
    sys.exit(-1)

geometry_path = os.path.join(os.getenv("DETCRDROOT"), "compact", geometry_option)
if not os.path.exists(geometry_path):
    print("Can't find the compact geometry file: %s"%geometry_path)
    sys.exit(-1)

from Configurables import GeomSvc
geosvc = GeomSvc("GeomSvc")
geosvc.compact = geometry_path
```

set the geometry option file

Environment variable :
“DETCRDROOT” = Detector/DetCRD/

Driver file (“CRD_o1_v01.xml”) location :
Detector/DetCRD/compact/CRD_o1_v01/CRD_o1_v01.xml

Settings in the run script

- Settings related to particle injection

```
32  ##############################################################################
33  # Physics Generator
34  ##############################################################################
35  from Configurables import GenAlgo
36  from Configurables import GtGunTool
37  from Configurables import StdHepRdr
38  from Configurables import SLCIORdr
39  from Configurables import HepMCRdr
40  from Configurables import GenPrinter
41  gun = GtGunTool("GtGunTool")
42  gun.Particles = ["mu-"]
43  gun.EnergyMins = [100.] # GeV
44  gun.EnergyMaxs = [100.] # GeV
45  gun.ThetaMins  = [85]    # deg
46  gun.ThetaMaxs  = [90]  # deg
47  gun.PhiMins    = [0]    # deg
48  gun.PhiMaxs    = [360]  # deg
49  # stdheprdr = StdHepRdr("StdHepRdr")
50  # stdheprdr.Input = "/cefs/data/stdhep/CEPC250/2fermions/E250.Pbhabha.e0.p0.whizard195/bhabha.e0.p0.00001.stdhep"
51  # lciordr = SLCIORdr("SLCIORdr")
52  # lciordr.Input = "/cefs/data/stdhep/lcio250/signal/Higgs/E250.Pbbh.whizard195/E250.Pbbh_X.e0.p0.whizard195/Pbbh_X.e
53  # hepmcrdr = HepMCRdr("HepMCRdr")
54  # hepmcrdr.Input = "example_UsingIterators.txt"
55
56  genprinter = GenPrinter("GenPrinter")
57
58  genalg = GenAlgo("GenAlgo")
59  genalg.GenTools = ["GtGunTool"]
```

Settings in the run script

- Output root file name

- At the last part of this script, we can specify which algorithms to be executed as well as number of events, services, output level
(but I only know/use portion of this functionality)

```
302
303 # output
304 from Configurables import PodioOutput
305 out = PodioOutput("outputalg")
306 #out.filename = "CRD_o1_v02-SimRecFit00_SDT.root"
307 out.filename = "test.root"
308 out.outputCommands = ["keep *"]
309
310 # ApplicationMgr
311 from Configurables import ApplicationMgr
312 ApplicationMgr(
313     TopAlg = [genalg, detsimalg,
314     digiVXD, digiSIT, digiSET, digiFTD,
315     spSIT, spSET, spFTD, tracking, forward, subset, full,
316     dCHDigiAlg, truthTrackerAlg, recGenfitAlgSDT,
317     out],
318     EvtSel = 'NONE',
319     EvtMax = 10,
320     ExtSvc = [rndmengine, rndmgensvc, dsvc,
321               evtseeder, geosvc, gearsvc, tracksystemsvc, ntsvc],
322     HistogramPersistency = 'ROOT',
323     OutputLevel = ERROR
324 )
```

last part of this script

Settings in the run script

- Settings for each algorithm in the middle part of the script

Example A: settings for VXD, SIT detectors

```
118 from Configurables import PlanarDigiAlg
119 digiVXD = PlanarDigiAlg("VXDDigi")
120 digiVXD.SimTrackHitCollection = "VXDCollection"
121 digiVXD.TrackerHitCollection = vxdhitname
122 digiVXD.ResolutionU = [0.0028, 0.006, 0.004, 0.004, 0.004, 0.004]
123 digiVXD.ResolutionV = [0.0028, 0.006, 0.004, 0.004, 0.004, 0.004]
124 digiVXD.UsePlanarTag = True
125 #digiVXD.OutputLevel = DEBUG
126
127 digiSIT = PlanarDigiAlg("SITDigi")
128 #digiSIT.IsStrip = False
129 digiSIT.IsStrip = True
130 digiSIT.SimTrackHitCollection = "SITCollection"
131 digiSIT.TrackerHitCollection = sithitname
132 digiSIT.TrackerHitAssociationCollection = "SITTrackerHitAssociation"
133 digiSIT.ResolutionU = [0.007]
134 #digiSIT.ResolutionV = [0.050]
135 digiSIT.ResolutionV = [0.000]
136 digiSIT.UsePlanarTag = True
```

Example B: settings for Tracking

```
221 from Configurables import FullLDCTrackingAlg
222 full = FullLDCTrackingAlg("FullTracking")
223 full.VTXTrackerHits = vxdhitname
224 #full.SITTrackerHits = sithitname
225 full.SITTrackerHits = sitspname
226 full.TPCTrackerHits = "NULL" # add TPC or DC tracker hit here, if TPC
227 full.SETTrackerHits = setspname
228 full.FTDPixelTrackerHits = ftdhitname
229 full.FTDSpacePoints = ftdspname
230 full.SITRawHits     = sithitname
231 full.SETRawHits     = sethitname
232 full.FTDRawHits     = ftdhitname
233 full.TPCTracks = "NULL" # add standalone TPC or DC track here
234 full.SiTracks  = "SubsetTracks"
235 full.OutputTracks  = "MarlinTrkTracks"
236 full.SETHitToTrackDistance = 5.
237 #full.OutputLevel = DEBUG
```

 # for those, we need to understand what input parameters are there, and so on . . .

Geometry Settings

sdt/Detector/DetCRD/compact/CRD_o1_v01/CRD_o1_v01.xml

1. Common (something like top-level) settings

2. Settings for each detector

   -- they can be ON/OFF

```
27
28  <include ref="./CRD_Dimensions_v01_01.xml"/>
29
30  <include ref="../CRD_common_v01/Beampipe_v01_01.xml"/>
31  <include ref="../CRD_common_v01/VXD_v01_01.xml"/>
32  <include ref="../CRD_common_v01/FTD_SimpleStaggered_v01_01.xml"/>
33  <include ref="../CRD_common_v01/SIT_SimplePlanar_v01_01.xml"/>
34  <include ref="../CRD_common_v01/DC_Simple_v01_01.xml"/>
35  <include ref="../CRD_common_v01/SET_SimplePlanar_v01_01.xml"/>
36  <include ref="../CRD_common_v01/Ecal_Crystal_Barrel_v01_01.xml"/>
37  <!--include ref="../CRD_common_v01/Ecal_Crystal_Endcap_v01_01.xml"/-->
38  <include ref="../CRD_common_v01/Coil_Simple_v01_01.xml"/>
39  <include ref="../CRD_common_v01/Hcal_Rpc_Barrel_v01_01.xml"/>
40  <include ref="../CRD_common_v01/Hcal_Rpc_Endcaps_v01_01.xml"/>
41  <!--include ref="../CRD_common_v01/Hcal_Rpc_EndcapRing_v01_01.xml"/-->
42  <include ref="../CRD_common_v01/Yoke_Barrel_v01_01.xml"/>
43  <include ref="../CRD_common_v01/Yoke_Endcaps_v01_01.xml"/>
44  <!--include ref="../CRD_common_v01/Lcal_v01_01.xml"/-->
45
46  <fields>
47    <field name="InnerSolenoid" type="solenoid"
```

e.g. sdt/Detector/DetCRD/compact/CRD_common_v01/SIT_SimplePixel_v01_01.xml

```
<detectors>
  <detector id="DetID_SIT" name="SIT" type="SIT_Simple_Pixel" vis="SITVis" readout="SITCollection" insideTrackingVolume="true">
    <envelope>
      <shape type="Assembly"/>
      <!--shape type="BooleanShape" operation="Union" material="Air" >
        <shape type="Tube" rmin="SIT_inner_radius"   rmax="SIT_outer_radius_1" dz="SIT_half_length_1" />
        <shape type="Tube" rmin="SIT_inner_radius_2" rmax="SIT_outer_radius"   dz="SIT_half_length" />
      </shape-->
    </envelope>

    <type_flags type="DetType_TRACKER + DetType_BARREL + DetType_PIXEL "/>

    <reconstruction strip_width="0." strip_length="0." strip_pitch="0." strip_angle="0*deg"  />

    <global sensitive_thickness="SIT_sensitive_thickness" support_thickness="SIT_support_thickness" sensor_length="SIT_sensor_length"
            sensitive_mat="G4_Si" support_mat="G4_C" sensitive_threshold_KeV="64*keV"  />
    <display ladder="SeeThrough" support="SITSupportVis" sens_env="SeeThrough" sens="SITSensitiveVis" />

    <layer layer_id="0" sensitive_radius="SIT1_inner_radius+0.5*SIT_sensitive_thickness" n_sensors_per_ladder="SIT1_half_length*2/SIT_sensor_length"
           n_ladders="2*pi*SIT1_inner_radius/SIT_sensor_length" ladder_clearance="0.1*mm" faces_IP="1" is_SIT1="1" is_SIT2="0"  />
    <layer layer_id="1" sensitive_radius="SIT2_inner_radius+0.5*SIT_sensitive_thickness" n_sensors_per_ladder="SIT2_half_length*2/SIT_sensor_length"
           n_ladders="2*pi*SIT2_inner_radius/SIT_sensor_length" ladder_clearance="0.1*mm" faces_IP="1" is_SIT1="0" is_SIT2="1"  />
    <layer layer_id="2" sensitive_radius="SIT3_inner_radius+0.5*SIT_sensitive_thickness" n_sensors_per_ladder="SIT3_half_length*2/SIT_sensor_length"
           n_ladders="2*pi*SIT3_inner_radius/SIT_sensor_length" ladder_clearance="0.1*mm" faces_IP="1" is_SIT1="0" is_SIT2="0"  />
    <layer layer_id="3" sensitive_radius="SIT4_inner_radius+0.5*SIT_sensitive_thickness" n_sensors_per_ladder="SIT4_half_length*2/SIT_sensor_length"
           n_ladders="2*pi*SIT4_inner_radius/SIT_sensor_length" ladder_clearance="0.1*mm" faces_IP="1" is_SIT1="0" is_SIT2="0"  />

  </detector>
</detectors>
```

Editing xml geometry files is (I think) within user's boundary

e.g. sdt/Detector/DetCEPCv4/src/tracker/SIT_Simple_Pixel_geo.cpp

Need to know the "DD4HEP"

```
62  static dd4hep::Ref_t create_element(dd4hep::Detector& theDetector, xml_h e
63
64    //---------------------------------------
65    //  See comments starting with '//**' for
66    //     hints on porting issues
67    //---------------------------------------
68
69
70    xml_det_t    x_det = e;
71    string       name  = x_det.nameStr();
72
73    dd4hep::DetElement   sit(  name, x_det.id()  ) ;
74
75    // --- create an envelope volume and position it into the world ----------
76
77    dd4hep::Volume envelope = dd4hep::xml::createPlacedEnvelope( theDetector,  e , sit
78
79    dd4hep::xml::setDetectorTypeFlag( e, sit ) ;
80
81    if( theDetector.buildType() == dd4hep::BUILD_ENVELOPE ) return sit ;
82
83    //-----------------------------------------------------------------------------
84
85    dd4hep::PlacedVolume pv;
86
87
88    sens.setType("tracker");
89
90
91    dd4hep::rec::ZPlanarData*  zPlanarData = new dd4hep::rec::ZPlanarData ;
```

```
93    //#################################################################################
94    //  code ported from SIT_Simple_Pixel::construct() :
95    //#################################
96
97    //  extended_reconstruction_parameters _e_r_p;
98
99    // ********************
100   //  Read and Store the Extended Reconstruction Parameters which are passed dir
101   // db->exec("select * from extended_reconstruction_parameters;");
102   // db->getTuple();
103   XMLHandlerDB db = XMLHandlerDB(  x_det.child( _Unicode( reconstruction ) ) ) ;
104
105   zPlanarData->widthStrip  = db->fetchDouble("strip_width")  ;
106   zPlanarData->lengthStrip = db->fetchDouble("strip_length") ;
107   zPlanarData->pitchStrip  = db->fetchDouble("strip_pitch")  ;
108   zPlanarData->angleStrip  = db->fetchDouble("strip_angle") ;
109   double strip_angle = zPlanarData->angleStrip ;
110   // ********************
111
112
113   //... db common_parameters
114   // // db->exec("select * from global;");
115   // // db->getTuple();
116   db = XMLHandlerDB(  x_det.child( _Unicode( global ) ) ) ;
117
118   // Sensitive Thickness
119   double sensitive_thickness = db->fetchDouble("sensitive_thickness") ;
120   // Support Thickness
121   double support_thickness = db->fetchDouble("support_thickness") ;
```

Summary of steps as a user

(・ Setup CEPCSW)

- Edit/Arrange the geometry xml files (for each detector)
- Arrange the run python script

  -- Specify the top level geometry files

  -- Settings for run (energy/number of events, root filenames, algorithm etc.)

  -- (if necessary) Arrange the input parameters of algorithm
  (preparation of new algorithm into the run script is ,,, I would expect supplied from developers)

- Run the simulation

Summary of steps – about difficult points

(・ Setup CEPCSW)

In many cases, we need helps from developers to update code of detector to set the configuration we want.

- Edit/Arrange the geometry xml files (for each detector)
- Arrange the run python script

  -- Specify the top level geometry files

  -- Settings for run (energy/number of events, root filenames, algorithm etc.)

  -- (if necessary) Arrange the input parameters of algorithm
  (preparation of new algorithm into the run script is ,,, I would expect supplied from developers)

- Run the simulation

Fundamental parameters (like track parameters) are stored in output roofiles, but there need some efforts if we want to have new variables etc.