Table of Contents

VeloPix in LHCb simulations

For the outline of this option see Paula Collins presentation at LHCb upgrade meeting on July 15, 2009.

First Gauss implementation was presented by Victor Coco at VELO upgrade meeting on Sept. 15, 2009.

First Boole implementation was presented by Marcin Kucharczyk at VELO upgrade meeting on Nov 24, 2009.

This page aims to be a recipe for using already implemented VeloPix solutions in the LHCb simulations. For those who might want to implement a new type of pixel sensors, modify the positions of modules, improve the already implemented solutions, etc. , a full description of the VeloPix implementation can be found in generic VeloPix description documentation. We will concentrate here on the different steps to follow to get VeloPix set up and used by LHCb simulations.

VeloPix in Panoramix

The following instructions were made using Panoramix v19r4 and CondDB tags: DDDB tag "velopix-mul-20100909" and SIMCOND tag "sim-20100510-vc-md100". If you wish to modify the xml description of the VeloPix then you must follow these steps.

Get the .db for VeloPix

1) Grab the DDDB_upgrade.db xml code and store it to a local area to work from ("SomeLocalPath" in the example I used $HOME/VELOPIX/dddb-velopix-mul-20100909/).

```
SetupProject LHCb
dump_db_to_files.py -c sqlite_file:$SQLITEDBPATH/DDDB_upgrade.db/DDDB -T "velopix-mul-20100909" -
dump_db_to_files.py -c sqlite_file:$SQLITEDBPATH/SIMCOND.db/SIMCOND -T "sim-20100510-vc-md100" -d
```

This saves the geometry and conditions database in xml format where it can be modified by the user.

Setup the Panoramix environment

2) We must now specify these tags which we have used so that Panoramix can build the xml and display the correct geometry.

```
SetupProject Panoramix v19r4
```

Then simply copy $WORKDIR/myPanoramix-upgrade-dddb.py file to your work directory. This file is a slightly modified version of $myPanoramix which sets the tags and also points Panoramix to the location of your xml description. Lets look at the additions to the code in a bit more detail

start Panoramix configurable

```
from Gaudi.Configuration import *
from Configurables import LHCbApp,CondDB
DDDBtag = "velopix-mul-20100909"
CondDBtag = "sim-20100510-vc-md100"
lhcbApp  = LHCbApp()
```

for simulation, tag should be the same as used in Gauss

```
lhcbApp.DDDBtag = DDDBtag
lhcbApp.CondDBtag = CondDBtag
```

Later in the file we specify the location of the xml to be used by Panoramix, we pass the lhcb.xml file containing the entire tree structure for the geometry and conditions

#--Geometry dependent options, use information from SIMCOND for positions

```
privatedb_path = "<SomeLocalPath>/"
from Configurables import DDDBConf
DDDBConf(DbRoot = privatedb_path + "lhcb.xml")
```

Running Panoramix with VeloPix

3) Finally to run Panoramix we must specify the head tag, which in this case, is the "Upgrade" tag. Enter the below into the terminal

```
python myPanoramix-upgrade-dddb.py -D Upgrade
```

You will probably see lots of errors but stick with it, eventually Panoramix will pop up and you can begin to check the changes you have made in the xml.

VeloPix in Gauss

The following instructions are given for Gauss v38r0p1 or higher. VeloPix is not supported on older version.

Set Gauss Environment

If you don't have Gauss environment already set, let's choose a version. Here we take v38r0p1but latest is usually the best choice:

```
SetupProject Gauss v38r0p1
```

Configure and run Gauss for VeloPix simulations

In order to specify to Gauss that you want to run on VeloPix, you will need to create an option file, say $YOURPATH/VeloPixSettings.py. Then you will be able to run Gauss:

```
gaudirun.py $GAUSSROOT/options/Gauss-Upgrade.py $YOURPATH/VeloPixSettings.py $GAUSSROOT/options/G
```

Let's detail the contents of VeloPixSettings.py:

To specify the tag of database to use, one needs the following lines.

```
from Configurables import LHCbApp
LHCbApp().DDDBtag   = "velopix-mul-20091116"
LHCbApp().CondDBtag = "sim-20091112-vc-md100"
```

"velopix-mul-20091116" is for the moment the only available official VeloPix tag. Of course if you are running on a private VeloPix geometry, you need to specify your own tag, see previous section.

To specify that one want to use VeloPix as vertex locator instead of the traditionnal Velo+!PuVeto, one need to replace the "VELO" field of the list of detector of which geometry is described as well as to the list of detectors one want to simulate and monitor. That is done through the Gauss configurable by adding to your $YOURPATH/VeloPixSettings.py the lines:

```
from Gauss.Configuration import *
```

```
Gauss().DetectorSim["VELO"]=['VeloPix']
Gauss().DetectorMoni["VELO"]=['VeloPix']
Gauss().DetectorGeo["VELO"]=['VeloPix']
```

For more details on the Gauss configuration for upgrade simulations please refer to this page.

Specify to Gauss a private DDDB with modified VeloPix

The normal way to do will be to specify the tag and the path of the upgrade DDDB containing the VeloPix you want to use. Since for the moment this is not completely supported, we will cheat by changing the default path of the DDDB_upgrade.db to point on your VeloPix-friendly DDDB_upgrade.db, located for example in ~/database/DDDB_upgrade_Private.db.

First of all save the default DDDB location and then getpack the package Det/SQLDDDB

```
setenv OLD_SQLITEDBPATH $SQLITEDBPATH
getpack Det/SQLDDDB
cd Det/SQLDDDB/cmt
source setup.csh
gmake
```

then in Det/SQLDDDB/db create links to the databases stroed in the default DDDB location (they are not copied to your Det/SQLDDDB/db folder since they are very heavy)

```
cd $SQLITEDBPATH
ln -s $OLD_SQLITEDBPATH/*.db .
```

Delete the link corresponding to the default DDDB_upgrade.db and reorient it to your local VeloPix-friendly database.

```
cd $SQLITEDBPATH
ln -s ~/database/DDDB_upgrade_Private.db DDDB_upgrade.db
```

In the end compile the package. Now when Gauss will call the Upgrade database at $SQLITEDBPATH/DDDB_upgrade.db/DDDB it will see your private VeloPix-friendly database as default. Don't forget to adapt the DDDBtag to the tag present in your private database, as explained on the next section. Usually the tag to use is "HEAD", unless you explicitly specified another one.

VeloPix in Boole

Different packages are available in Boole from release v20r1 on:

VP/VPAlgorithms - Digitization and Clustering code
VP/VPAssociators - Associators from clusters and digits to MCHits and MCParticles

In Lbcom from release v8r0 on you will find:
VP/VPDAQ - Preparing raw data for the VP and the decoding of the VP clusters (Brunel level)

The algorithms from this package are automatically taken by Boole or Brunel, if these algorithms are added to the Boole/Brunel VP sequences in Configuration.py (they are already added to the Boole VP sequence).

Set Boole Environment

In order to set Boole environment, you have to choose Boole version (v20r1 or higher), e.g.:

```
setenvBoole v20r1
```

Then you have to getpack Boole, i.e.:

```
getpack Digi/Boole v20r1
```

Configure and run Boole with VeloPix

To run Boole with VeloPix, you need to create an option file, let say $YOURPATH/VeloPixBoole.py, where the L0 is switched off and VeloPix algorithms are taken instead of present Velo. Then you may run Boole:

```
gaudirun.py $BOOLEROOT/options/Boole-Default.py  $YOURPATH/VeloPixBoole.py
```

The contents of VeloPixBoole.py in detail:

To specify the tag of database to use, one needs the following lines (see also VeloPix in Gauss).

```
LHCbApp().DataType  = "Upgrade"
LHCbApp().DDDBtag   = "velopix-mul-20100909"
LHCbApp().CondDBtag = "sim-20100510-vc-md100"
```

Then you have to specify that you want to use VeloPix instead of Velo and you have to switch the L0 off:

```
theApp.DetectorDigi["VELO"]=['VeloPix']
theApp.DetectorDigi["L0"]=[]
theApp.DetectorLink["VELO"]=['VeloPix']
theApp.DetectorLink["L0"]=[]
theApp.DetectorLink["TR"]=[]
theApp.DetectorMoni["VELO"]=['VeloPix']
theApp.DetectorMoni["L0"]=[]
```

Next step is to choose the spillover:

```
theApp.UseSpillover = True
DigiConf().SpilloverPaths = ["Prev","PrevPrev","Next"]
```

and specify the input sim file which should be generated using Gauss with VeloPix geometry (see VeloPix in Gauss).

To produce the digi output file, you have to specify its type and destination, as it is done in the last line of of VeloPixBoole.py.

In the case of any problems please contact MarcinKucharczyk

VeloPix in Brunel

Set Brunel Environment

Brunel v is the first Brunel release fully compatible with VeloPix. You can use every version posterior to it but also work with the nightlies:

```
SetupProject Brunel HEAD --nightly lhcb-head (add --build-env to create the environment the first
```

Brunel option files

In order to run Brunel with VeloPix, one need to specify the "Upgrade" datatype as well as the fact one want to use VeloPix instead of Velo. Option file should contains:

```
RecSysConf().RecoSequence = ["Decoding", "VELOPIX","TT","IT","OT","Tr","Vertex","RICH","CALO","MU
```

Make sure that the correct tag are used for the database (see....). First tag compatible with Brunel is velopix-20100909

Configure and run Brunel with VeloPix

An example of option file to run the VeloPix in Brunel is given in Rec/RecoUpgrade/options/Brunel_UpgradeVeloPix.py.

The key points are to specify that VeloPix have to be used in pattern recognition

```
TrackSys().TrackPatRecAlgorithms = ["VeloPix"]
```

and that the reconstruction sequence take the VeloPix into account:

```
RecSysConf().RecoSequence = ["Decoding", "VELOPIX","TT","IT","OT","Tr","Vertex","RICH","CALO","PR
```

Note that MUON have been removed because it's sequence call the nominal Velo. This have to be fixed at one point. Finally the output type and database should be specified:

```
Brunel().OutputType = "VELOPIXDST"
from Configurables import LHCbApp
LHCbApp().DDDBtag   = "velopix-mul-20091116"
LHCbApp().CondDBtag = "sim-20091112-vc-md100"
```

IMPORTANT: since for the time beuing the Configurables are not finalised, some of the nominal settings of the sequence (forward tracking, MUON, etc...) are not yet available. This work is in progress...

TODO list for VeloPix in Brunel

Reconstruction

VeloPix Monitoring

Track Monitoring

-- VictorCoco - 21-Apr-2010 -- MatthewReid - 19-Jul-2011

- myPanoramix-upgrade-dddb.py.txt: python script for running Panoramix usin velopix geometry

---

This topic: LHCb > VeloPix
Topic revision: r26 - 2013-06-18 - HellaSnoek