

HPS Simulation + Recon HOWTO Guide

Jeremy McCormick, SLAC

Goals

- Build the detector simulation program slic.
- Run some simulated physics events in an HPS test detector using slic and get LCIO output data.
- Run sample hps-java codes on the slic output to produce recon objects like Clusters, TrackerHits, and Tracks.
- Write some of your own simple analysis code, including simple histogramming of event data.

What will you need to start?

- A PC on which you can download and install software. (No iPhone or Android apps yet!)
 - But you should NOT need root/admin access.
- Linux or OSX
 - Java recon should work okay on Windows, as this language is platform independent. Windows is not at all recommended for building the simulation tools.
- An internet connection.
 - Once the framework is built and installed, you should be able to work offline.
- A working directory with sufficient space, that will be abbreviated as “/myworkdir” in subsequent instructions.

What software should be pre-installed before I begin?

- For building slic...
 - cvs
 - make
 - cmake 2.8.2 or greater
 - <http://www.cmake.org>
 - wget
 - svn
 - Java SDK 1.6+ (still needed for LCIO?)
- For recon using lcsim...
 - Java SDK 1.6+
 - Maven 2.2.1 (This version number is important!)
 - <http://maven.apache.org>
 - IDE such as Eclipse or Netbeans
 - optional but very useful for development

CVS Access

- We will use anonymous access to the SLAC CVS for now, which is enough to get started.
- Contact Tony Johnson at SLAC if you need write access. (Some may already have this.)
- Set the CVSROOT variable in your environment.

Bash

```
export CVSROOT=:pserver:anonymous@cvs.freehep.org:/cvs/lcd
```

C Shell

```
setenv CVSROOT :pserver:anonymous@cvs.freehep.org:/cvs/lcd
```

Now we can use commands like...

```
cvcs co lcsim
```

Building slic with SimDist

Build SimDist

```
cd /myworkdir  
cvs co SimDist  
cd SimDist  
./configure --with-geant4-version=v9r3p02  
make
```

Test slic

```
./scripts/slic.sh
```

Common build errors...


- No internet connection or lost it mid-build
- Some mysterious platform incompatibility (please tell me)
- Missing built-in libraries (such as getopt)

Accessing HPS Detector Data

Checkout the detector data CVS project

```
cd /myworkdir  
cvs co hps-detectors
```

Detectors each have their own directory
in hps-detectors/detectors where their data is stored.

detectors/HPS-Test-JLAB-v3pt1  HPS-Test-JLAB-v3pt1.lcdd
SamplingFractions
compact.xml
detector.properties

Set a local alias for the detector you are working on

```
cd detectors/HPS-Test-JLAB-v3pt1  
echo "HPS-Test-JLAB-v3pt1: file://$(pwd)" >> ~/.lcsim/alias.properties
```

The above command makes sure that the Java recon can find
the local copy of your detector data.

Generating Physics Events in StdHep Format

- MADGRAPH
- FTP area for event files
 - ftp://ftp-hps.slac.stanford.edu/hps/hps_data/MadGraph/
- Instructions for StdHep gen???? (Matt?)

Geant4 GPS

Sample GPS Macro for generating single electrons with circular beam spot...

```
/gps/particle e-  
/gps/number 1  
/gps/ene/type Mono  
/gps/pos/type Beam  
  
# The incident surface is in the x-y plane  
/gps/pos/rot1 1 0 0  
/gps/pos/rot2 0 1 0  
  
# Beam characteristics  
/gps/pos/shape Circle  
/gps/pos/centre 0 0 -.1  
/gps/pos/sigma_r 0.20 mm  
  
# The beam is travelling along the z-axis  
/gps/ang/rot1 -1 0 0  
/gps/ang/rot2 0 1 0  
/gps/ang/type beam1d  
  
# Beam energy  
/gps/energy 2.2 GeV
```

Running the Detector Simulation

Use a pre-generated StdHep file for physics events...

```
cd /myworkdir
./scripts/slic.sh \
-g ./hps-detectors/detectors/HPS-Test-JLAB-v3pt1/HPS-Test-JLAB-v3pt1.lcdd \
-i physicsEvents.stdhep \
-o slicHpsOutput -x -r 10000
```

- G4 macro example using GPS????

```
cd /myworkdir
./scripts/slic.sh \
-g ./hps-detectors/detectors/HPS-Test-JLAB-v3pt1/HPS-Test-JLAB-v3pt1.lcdd \
-m gps.mac \
[...rest is the same...]
```

Building HPS Java

```
cvs co hps-java
cd hps-java
mvn install
```

Common build errors...

- Not connected to internet.
- Dependency problem such as missing jars. (Please let me know about this as all deps should be deployed.)
- Wrong version of Maven (need 2.2.1).

LCSim does not know about hps-java, so this line needs to be put into your XML job description...

```
<classpath>
  <jar>~/m2/repository/org/lcsim/hps-java/1.0-SNAPSHOT/hps-java-1.0- SNAPSHOT.jar</jar>
</classpath>
```

Setting up LCSim

Option #1: Download a runnable jar

```
wget http://www.lcsim.org/maven2/org/lcsim/lcsim/1.19-SNAPSHOT/lcsim-1.19-SNAPSHOT-bin.jar
```

Option #2: Build it yourself

```
cd /myworkdir  
cvs co lcsim  
cd lcsim  
mvn install -DskipTests=true
```

Running LCSim

- LCSim is distributed as a runnable, “uber” jar, which means that all of its dependencies are included.
- It is run with the “java” command which executes a default main method.
- You may also run your own main methods using standard java syntax.
- Command line guide...

<https://confluence.slac.stanford.edu/display/ilc/lcsim+xml>

LCSim Command Line Syntax

```
[$] java -jar ./target/lcsim-1.19-SNAPSHOT-bin.jar
```

```
java -jar lcsim-bin.jar [options] steeringFile.xml
```

usage:

- D Define a variable with form [name]=[value]
- n Set the max number of events to process.
- p Load a properties file containing variable definitions
- q Turn on quiet mode.
- s Set the number of events to skip.
- v Turn on verbose mode
- w Rewrite the XML file with variables resolved
- x Perform a dry run which does not process events

!!!! Put link to LCSim CL confluence docs here. !!!!

Running the ECal Reconstruction

```
cd /myworkdir
java -jar lcsim/target/lcsim-1.19-SNAPSHOT-bin.jar \
  hps-java/src/main/resources/org/lcsim/hps/steering/ecal_example.lcsim \
  -DinputFile=/myworkdir/slicHPSOutput
```

- Location of lcsim may be different, depending on how you built it.
- The **inputFile** variable is defined within the XML steering file and used to specify the input events as well as a tag for output data such as new LCIO files or histograms.

Accessing and Histogramming CalorimeterHit Data

```
class SimpleCalAnalDriver extends Driver
{
static AIDA aida = AIDA.defaultInstance();
void process(EventHeader event)
{
    List<CalorimeterHit> hits = event.get(CalorimeterHit.class, "Ecal");
    for (CalorimeterHit hit : hits)
    {
        aida.cloud1d("Corr Energy").fill(hit.getCorrectedEnergy());
        aida.cloud1d("Raw Energy").fill(hit.getRawEnergy());
        aida.cloud2d("Pos XY").fill(hit.getPosition()[0], hit.getPosition()[1]);
    }
}
}
```


Histogramming using AIDA

- Provides common analysis classes such as 1 and 2D histograms and clouds.
- Can be viewed in JAS3 or a number of other AIDA-compatible analysis tools.
 - There is even a way to load directly into ROOT.

To save your histos at the end of an LCSim job...

```
<driver name="AidaSaveDriver"  
  type="org.lcsim.job.AidaSaveDriver">  
  <outputFileName>myhistos</outputFileName>  
</driver>
```