EvtGen inside the PandaRoot Framework

Björn Spruck
II. Physikalisches Institut, Uni Gießen

Outline

- Status up to now
- Work done
- Changes for the User

Panda Collaboration Meeting, Stockholm, 14-18.6.2010

EvtGen?

- Event Generator (better: Decayer)
- Used by a large number of HE experiments
- Large database of decays
  - decay of one particle in all known channels
- Particle and decay properties listed in test files
  - easy to modify
- Can (re)define you own decays
  - Well defined decay for MC studies

Status up to now

- EvtGen code shipped within PandaRoot svn
- But: User has to compile it himself!
- Result: standalone binary, f.e. simpleEvtGen

- Depend on cernlib
  - to be installed by the user
  - versions, 32/64, compiler
- Mix of C and Fortran code
  - g77/gfortran problem (better: libgfortran, libg2c)
  - g77 not (officially) supported anymore since gcc 4.0

Goal

- Get it compiled automagically on all systems
  - different compilers, 32/64, g77/gfortran
  - compile it together with PandaRoot
    - Cmake
    - Shared Lib
- Have a PandaRoot Task
  - useable inside macros
  - keep standalone binary (optional)

Problem

- Precompiled Cernlib
  - many versions, depending on compiler and system
  - future support?
  - a lot of problems
- Compiling Cernlib from source is not a solution
  - complicated, too
- Again:
  - g77/gfortran problem (better: libgfortran, libg2c)
  - g77 not (officially) supported anymore since gcc 4.0

Solution

- Get rid of precompiled Cernlib!!!
- What is EvtGen depending on?
  - Pythia
  - Photos
  - and a few more things...
- And:
  - Pythia is already in FairSoft as shared lib, why not use it?
  - Cernlib available as source code
    - use/copy only needed parts

Making It Compile

- Use existing **libPythia** instead of Cernlib
- Photos not depending on other Cernlib modules
  - add as source code and build shared **libPhotos**
- EvtGen contains “unused” code which depend on Cernlib
  - get rid of it (hbook...)
- Left over:
  - ddilog and ranf function missing
  - add als source code
- Works!!!

Code structure

- /trunk/pgenerators/
  - EvtGen/
    - photos/ contains code for **libPhotos**, but could be put anywhere
    - EvtGenBase/ -> **libEvtGen**
    - EvtGenModels/ -> **libEvtGen**
    - Cernlib/ (only two files) -> **libEvtGen**
  - EvtGenDirect/
    - PndEvtGenDirect.cxx/h -> **libEvtGen**
- Note:
  - The whole EvtGen Directory can be put to any location, e.g. fairroot

Short Summary

- No dependence on precompiled Cernlib anymore!!!
- Ships and compiles with PandaRoot
  - on all tested system
  - gcc 3&4, g77/gfortran/ifort
  - 32/64 bit, SLC4&5, Ubuntu 9&10, Suse ...
- What you get:
  - Shared library
  - Standalone binary (simpleEvtGen and others)
  - PandaRoot task (taking the same parameters as simpleEvtGen)

Changes for the User

- None (if you want)
  - (nearly) no change on EvtGen package
  - “old” way of compiling/linking against cernlib still works
  - all changes are by defines in CMakeFile.txt
- Standalone binaries still in package
  - compiled automatically (which are needed???)
  - compiled from Photos/Pythia/EvtGen
  - located in build/bin/
- Or:
  - Use the new Task ->

Example:

- Using example code from tutorials/charmonium/
  - run_sim_tpccombi.C
  - Before simulation (same in both cases):
    - Prepare your user.dec, DECAY.DEC and evt.pdl file (optional)
    - See EvtGen Manual / Wiki / Torino Tutorial
    - esp the one for simpleEvtGen

```
Old Way:
run simpleEvtGen particle userdecayfile #nrevts [mom] [seed]
(the file output.evt is created)
in the macro, use:
FairEvtGenGenerator* evtGen = new FairEvtGenGenerator("output.evt");

New Way:
in the macro, use:
PndEvtGenDirect* evtGen = new PndEvtGenDirect("particle","userdecayfile"[,mom,seed]);
```

Backup and further notes

Remarks and Tests of EvtGen in PandaRoot

Try out and check

- If you want to try:
  - Add to rootlogon.C (to be done in svn)
    - if(isLibrary("libPhotos"))gSystem->Load("libPhotos");
    - if(isLibrary("libEvtGen"))gSystem->Load("libEvtGen");
    - if(isLibrary("libEvtGenDirect"))gSystem->Load("libEvtGenDirect");
  - ... and change the generator line in your macro.

- Check it!!!
  - old way simpleEvtGen with CernLib against new SimpleEvtGen
  - can be done on text output basis without analysis

Checks done sor far

- old way simpleEvtGen with CernLib against new SimpleEvtGen ... compare output.evt
- Simple decay J/psi->ee,mumu including photos, 1000Evts
  - SLC 4.7 (32), “old” cernlib 2005?
  - SLC 5.4 (64), Ubuntu 9.10(64), cernlib2006
  - no difference between old and new!
  - one rounding(?) error difference between the SLC4 and other systems
- Pythia decays not done
  - same error on all systems... decay file problem?
  - either “pythia cannot decay this particle” or “no requested process has non-vanishing cross-section.”

INTELs FORTRAN compiler ifort

- add library path, compiler name and path to CMakeLists
- w/o additional parameters -> compiler and runs
- with “-fltconsitency -zero – auto” -> compiles and run
- with “-check all” -> Problem, but why???
  - compiles but does not run
  - Error: Symbol #include is not defined in current scope
-

More Checks done...

- using runtime checks: -fstack-protector -fbounds-check -fcheck-array-temporaries -frange-check -fcheck-data-deps
  - (-fcheck=all not supported?)
- **No** error detected on running without Photos
  - Psi(3770) all decays from DECAY.DEC
- But with Photos: (**solved!**)
  - At line 64 of file /home/bjoern/pandaroot/trunk/pgenerators/EvtGen/photos/photos_make.F
  - Fortran runtime error: Array reference out of bounds for array 'jmohep', lower bound of dimension 2 exceeded (0 < 1)
  - IF (JDAHEP(1,I).NE.0.AND.JMOHEP(1,JDAHEP(1,I)).EQ.I) THEN
  - for me the code looks o.k. if and is done left before right nad righ only if left succeeded (like in C code).
    - But this is fortran!
  - Error is gone if code is splitted into two lines
    - IF (JDAHEP(1,I).NE.0) THEN
    - IF (JMOHEP(1,JDAHEP(1,I)).EQ.I) THEN
- check in old precompiled version not possible...
  - however the interface can be compiler with flags, they do not show an error!
- Other fortran compilers might have more thorough checks.
  - ifort ... works

Known Problems

- Due to global fortran variables, only **one** EvtGen can be run at the time! (but why would you like to run two at the same time?)
- There is no real error msg in that case, but maybe one can set a flag to prevent this?