

Changes to the MIPP offline fluka and e907mc packages

Holger Meyer (WSU)

April 4, 2011

Abstract

Changes to several MIPP offline analysis packages[1] are described. These are *fluka*, *e907mc*, *E907MCInterface*. All modifications are fully backwards compatible.

1 Background and problem description

The simulation tool chain in MIPP data analysis consists of event generation (resulting in a stdhep file), tracking of events through the detector geometry in e907mc which is based on Geant3 (resulting in root data files with MC-truth and hit information) and digitization and reconstruction in an anamipp job using the offline framework (resulting in root data files with digits and tracks added and in root dst files). Event generation is done mainly with fluka and dpmjet. With dpmjet a primary interaction is generated and primary particles are written to the stdhep file. With fluka the primary interaction is tracked in fluka through the experimental target volume and tracks are written to the stdhep file as they cross the target volume boundary. The advantage is that all of fluka's physics models for interactions are taken into account inside the target. A disadvantage is that information on primary vertex position and multiplicity used to not be available in the later stages as it was not stored in the stdhep files. This hindered among other things the study of KNO scaling behavior in MIPP data. The modifications to fluka now write out the complete event tree of interactions so that the history of particles crossing the target volume boundary is available.

It is also desirable to retain the information on what kind of interaction (elastic, inelastic, ...) created the particles in the event. This information is accessible in fluka and is preserved with the modifications presented here.

2 Problem solution overview

It was decided[2] to add the vertex information from fluka to the stdhep files while retaining the particles tracked by fluka through the target. In

e907mc the particles from the target volume boundary crossing will be tracked as before (and output placed in folders “kine” and “hits” in the output root file) while additional information on the history of particles inside the target is ignored during tracking in geant. The primary vertex particles that decay inside the target are ignored and simply passed through into the output file into the “kine” folder. Care is taken to correctly assign all particles to the correct vertices.

If vertex information is not available (as in the case of stdhep files before the modifications described here or for stdhep files from dpmjet), then e907mc behaves fully backwards compatible. Even if the *old* e907mc encounters an stdhep file produced by the *new* fluka (a highly unusual, unnecessary use-case), despite the limitations in the stdhep file format, the new information in the stdhep files will be ignored because the old hep2geant routine skips over particles with stdhep status flags other than 1 or the MIPP beam status flag (usually 301) as they are considered to be decayed particles.

In the further offline analysis the changes are also completely transparent. No geant hits are generated from the particle tracks fully contained inside the target and thus there will not be any changes in the reconstruction.

Fluka has been modified to write out the vertex particle information. The input file must specify a USERDUMP card in the fluka input file to activate this. The previously used USERWEIG card should be removed to avoid duplicate output.

In the process of this update some other bugs have been identified and fixed. These include the assignment of kinetic energy thresholds for electrons and photons in the .inp files to drive fluka. Thresholds have been set to coincide between fluka and e907mc.

3 Problem solution details

3.1 fluka

Limits of stdhep files. Use status flag to encode particle number. The STDHEP manual[3] defines the ISTHEP integer as follows:

ISTHEP value	-	definition
0	-	null
1	-	final state particle
2	-	intermediate state
3	-	documentation line
4-10	-	reserved for future use
11-200	-	reserved for specific model use
201-...	-	reserved for users

To this was added before the MIPP convention:

ISTHEP value	-	definition
301	-	beam particle to be tracked backwards (deprecated, for backwards compatibility only)

This code is now depreciated, but retained for backwards compatibility.

Now the following codes are used. Particles from the stdhep file with these codes must not be tracked in e907mc.

ISTHEP value	-	definition
350	-	particle produced at primary interaction if it does not cross the target volume boundary
351	-	particle produced at secondary interaction if it does not cross the target volume boundary
...	-	...
387	-	particle produced in thirty seventh or higher level of interaction if it does not cross target volume boundary
389	-	particle produced below threshold set in fluka
<hr/>		
		beam particle, event primary interaction is.... (see fluka manual p.361, USDRAW entry to MGDRAW routine)
399	-	no event, uninteracted beam particle
40x	-	interaction in fluka subroutine KASKAD (hadron and muon interactions)
400	-	elastic interaction
401	-	inelastic interaction
402	-	particle decay
403	-	delta ray generation
404	-	pair production
405	-	bremsstrahlung
410	-	radioactive decay
50x	-	interaction in fluka subroutine EMFSCO (electron, positron and photon interactions)
508	-	bremsstrahlung
510	-	Mller scattering
512	-	Bhabha scattering
514	-	in-flight annihilation
515	-	annihilation at rest
517	-	pair production
519	-	Compton scattering
521	-	photoelectric interaction
525	-	Rayleigh scattering interaction
60x	-	interaction in fluka subroutine KASNEU (low-energy neutron interactions)
600	-	neutron interaction
70x	-	interaction in fluka subroutine KASHEA (heavy ion interactions)
700	-	delta ray generation

Some of these codes could be argued to duplicate definitions provided in the standard. For example ISTHEP code 2 might be used for those primary particles that are decayed or interacted by fluka. However, it was deemed a cleaner implementation to consistently use codes in the range reserved for users. Of the ISTHEP codes for beam particles encoding

the type of interaction only codes 399, 400, and 401 are expected to be used in MIPP. However, the other codes are defined here because fluka can issue these codes. They are the codes defined on page 361 of the fluka manual[4] with an offset of 300 added because the lower codes would otherwise conflict with other STDHEP ISTHEP codes.

Here is a flow example of how an event might get generated with particular attention to the call to user routines and their order. All the xxDRAW routines are actually separate entry points to the MGDRAW routine and the code resides in mgdraw.f:

1. Fluka starts the event. The routine SODRAW is called and puts the beam particle on the stdhep particle stack with ISTHEP code ISTHEPNOINT (399).
2. If there is no interaction then BXDRAW is called as the beam particle leaves the target volume and it is added to the stdhep stack with ISTHEP code ISTHEPTRACK (1). – OR – UGDRAW is called on the primary interaction. The ISTHEP flag and vertex position of the beam particle (which is already on the stdhep stack from the call to SODRAW) are updated with the position and type of interaction. Also all the produced particles are put onto the fluka stack with isthep code ISTHEPPRIMARY (350) and vertex position that is irrelevant as it will be updated when the particles interact or leave the target volume. If a produced particle is below threshold it will not be tracked by fluka. These particles are nonetheless put in the stdhep output with the vertex position of their origin and isthep code ISTHEPBELOWTHRESHOLD (389). JMOHEP indices of the produced particles and JDAHEP indices of the beam particle are set with the correct values.
3. The first particle of the interaction is tracked in fluka. (Actually it may be called the last particle as the fluka particle stack is processed in a FILO order as would be expected for a stack buffer.) Either it hits the target boundary. Then, in a call to BXDRAW, its vertex position is set to the target boundary crossing position and the ISTHEP value is set to ISTHEPTRACK. Or it undergoes an interaction or decay within the target volume. Then, in a call to UGDRAW, its vertex position is updated with the final point on the track (decay or interaction position) and all daughters are placed on the stdhep stack and JDAHEP for the mother and JMOHEP for the daughters are set. The fluka commons FLKSTK and GENSTK contain information on primaries and generated particles.
4. Other particles are processed in a similar manner until all particles have crossed out of the target volume.
5. The entry ENDRAW to the MGDRAW routine is called at the end of the event. This correctly sets the generation (primary, secondary,...) for particles that are fully contained inside the target. Also some consistency checks are performed and problems reported to stdout.

The interaction multiplicity for the event can now be determined from the stdhep data by the number of daughters of the beam particle: multiplicity = $jdahep(2,1) - (jdahep(1,1) + 1 = jdahep(2,1) - 1$ because the

first daughter particle of the beam track will always be stored with index 2. The beam track itself is stored first and has index 1.

3.2 e907mc

The e907mc program structure as relevant here is as follows: For each event the routine hep2geant3 is called to get the stdhep particles copied into the Geant3 data structures. Then Geant3 routines track all the particles in Geant3 common blocks (and add secondary particles and hits). Finally the routines in the E907MCInterface package (MCIDoKine and MCISoreHits) copy the information to the root data structures used in the MIPP offline software. Thus, in order to add the vertex particle information to the root output file without tracking these particles in Geant3, they will get handed to the E907MCInterface directly from hep2geant3 and never get put into common blocks or other data structures used by Geant3. The hep2geant3 routine has been modified in the past by MIPP to allow for the inverse beam flag and distribution of vertex positions throughout the target volume (for stdhep files from dpmjet).

3.3 E907MCInterface

- output into root file

3.4 DST

As we want this new information to be available in the DSTs the MIPPEventSummary and DSTMaker have been modified....

References

- [1] For general information on MIPP and MIPP data analysis see <http://www-mipp.fnal.gov/> and documents linked there.
- [2] The decision was based on discussion at weekly MIPP phone meetings. See H. Meyer, "Vertex information in stdhep files from fluka", MIPP-Doc-1062-v2
- [3] Lynn Garren, "StdHep 5.06.01 Monte Carlo Standardization at FNAL Fortran and C Implementation" (20 November 2006), FNAL-CD-Doc-903-v15, page 5, available at http://cd-docdb.fnal.gov/cgi-bin/RetrieveFile?docid=903&version=15&filename=stdhep_50601_manual.pdf
- [4] Alfredo Ferrari, Paola R. Sala, Alberto Fassò, Johannes Ranft, "Fluka: a multi-particle transport code (Program version 2008)", available at <http://www.fluka.org/content/manuals/FM.pdf>