

FLUKA

a multipurpose Monte Carlo Code



G. Battistoni

Handling of errors and crashes



We often receive request of help from users

- 90% of times these requests are simple problems that the user could have located himself: the first purpose of this lecture is to help you how to spot an error which depends on the user.
- In the 5% of cases a real FLUKA problem is found. To locate this kind of error is slightly more complicated, but the second purpose of this lecture is exactly concerning this.
- In the other 5% of cases, the debate is on the meaningfulness of results. This has nothing to do with this lecture....



A first kind of crash

- If the program crashes immediately (usually giving “segmentation fault”) without giving any file: it might be a problem of compilation, or conflict with respect to your architecture:
 1. You have not downloaded the right version for your architecture: Download, recompile (if you have user routines) and link.
 2. *There could be no FLUKA version suitable for your architecture...*



The first thing to do in case of crash...

- In any case you are left with some file: ***.log** (at least!) and ***.out**
- **Always look at the content of these files!**

A very simple case: you ask to run a FLUKA executable:

\$FLUPRO/flutil/rfluka -e myfluka -NO -M3 myinput
and all is immediately stopped.

```
Initial seed already existing
```

```
Running fluka in /home/battist/flukacourse/Pavia/examples/errors/fluka_31884
```

```
===== Running FLUKA for cycle # 1 =====
```

You are left with a **fluka_****** directory where just a ***.log** file is there. You might find a message of this kind:

```
../rfluka: line 259: ../myfluka: No such file or directory
```

The program myfluka did not exist or the path to it is missing!!



FLUKA version is expired...

You get the following message:

```
**** This version is obsolete and is  
**** going to expire in few days.  
**** Please contact Alfredo Ferrari  
**** INFN-Milan, tel.+39-02-503-17374  
**** or look for an updated version  
**** at http://www.fluka.org
```

You have to download an updated version:
Download, recompile (if you have user routines) and
relink.



There is something wrong in the input file!

- By far this is the most common problem. This is easily recognizable, since in the `*.out` file the echo of all lines of the input file is reported. In case of problem, the output stops with the last card correctly interpreted
- Be careful: sometimes **non visible control characters may appear in a file sent via-mail (not for all mail clients!). In this case use "dos2unix"** (a specific rpm exists for linux) They can be eliminated also using simple perl commands)



A few typical examples of this case:

```
***** Next control card *****  BEAM      -3.500      -8.2425E-02  -1.700      0.000      0.000      1.000      PROTON
Abort called from FLUKAM reason UNKNOWN PROJECTILE OR "OLD" HEAVY ION OPTION (NO LONGER SUPPORTED) Run stopped!
STOP UNKNOWN PROJECTILE OR "OLD" HEAVY ION OPTION (NO LONGER SUPPORTED)
```

You are using fixed format, but the SDUM identifying the particle is not well aligned (see [example16_1.inp](#))

```
*** Unable to resolve name element BLCKHOL in card ***
ASSIGNMA      BLCKHOLE  BLKHOLE
*** run stopped_***
```

Another misalignment in fixed format the finale E of BLACKHOLE is lost (see [example16_2.inp](#))



A few typical examples of this case: (continues...)

```
*** Unknown control code: ROT-DEF , ignored ***
```

```
***** Next control card *****  ROT-DEF  0.000  0.000  0.000  0.000  0.000  0.000  0.000  NEW-DEFA
```

```
**** Unknown Input card !!!!!!!!!!! ****
```

**You mistyped a FLUKA command (ROT-DEF instead of ROT-DEFI
(see example16_4.inp)**

```
*** The 3th field      -50  of the following input card ***
USRBDX      99.0      218.0      -50      TARGS1      TARGS2      78.5398 Sp1ChH
*** does not contain a valid formatted fortran real number!!! ***
*** It is ambiguous and it could be read differently on different compilers ***
*** depending whether it defaults or not to the blank=0 formatted input rule ***
```

You gave a numerical value without the "." (see example16_7.inp)



Have you debugged your geometry?

- Sometimes input errors are in the geometry
- 2 order of mistakes:
 1. trivial (wrong data cards)
 2. essential (undefined space points, multiple definition of space points): see the lectures on geometry and the use of DEBUG in the GEOEND card

See also the use of RAY to trace and analyze your geometry



An example of trivial mistake in geometry input

(see example16_6.inp) There a ZCC is defined with only 2 numbers instead than 3.

```
* Lead target  
ZCC TARG      0.0 0.0
```

You will find:

```
For the body = TARG      type = ZCC      2 numeric parameters were given instead of 3
```

The important geometry errors may appear as a lot of messages in the *.out and *.err file, complaining that a point in space is not defined, cannot be reached, etc.

Remember the need for proper normalization of direction cosines! (magfld, source etc.) You will get a lot of (warning) messages also in those cases, but this is not a true error...



Another frequent error...

```
*** (n,p) proton production activated for Xsec mat. # 3 ***  
**** Low energy neutron xsec not found for some media 12 13 ****  
POTASS
```


(see `example16_3.inp`) There a material has been defined using a name which has no correspondance with the low energy neutron cross sections available in FLUKA (see chap. 10 of the manual). In this case **POTASS** instead of **POTASSIUM**

Check the names of the materials.

Check the use of **LOW-MAT** cards: check if are using the updated parameters listed in the manual. Do you really need **LOW-MAT**?

This is necessary only in a limited number of cases and it is easy to misinterpret this command if you are not an expert user.

[Read carefully the manual about this]



What is left after these problems (handled by FLUKA automatically)

These errors are not generating exceptions. The temporary directory is erased and you remain with this message on the standard output:

```
Removing links
```

```
Removing temporary files
```

```
Saving output and random number seed  
No ranexample16_2002 generated!
```



FLUKA users and the manual

- All reported here so far implies that a FLUKA user is living symbiotically with the Manual....



Run time errors with exceptions...

- These are caused in 90% of cases by problems in user's code or user's configuration

You will see a message of this kind:

```
==== Running FLUKA for cycle # 1 =====  
ka: line 309: 30048 Aborted (core dumped) ${EXE} <${INPF 2}>${LOGF }>${LOGF
```

The temporary directory remains there. You will find the *.log, *.out, *.err files, a core.* file and the last random seed

The last random seed allows to restart your run from a configuration which was the one occurring at maximum 5 minutes of CPU before the error!

You can easily spot the problem using the GNU debugger (gdb). See `example16_5.inp` and the use of `source.f` (gaussian sampling with μ and σ passed by SOURCE command)



Using gdb...

- 1) Go inside the temporary directory
- 2) type: `gdb ../flukamy core.nnnn`



```
GNU gdb Red Hat Linux (6.0post-0.20040223.19rh)
Copyright 2004 Free Software Foundation, Inc.
GDB is free software, covered by the GNU General Public License, and you are
welcome to change it and/or distribute copies of it under certain conditions.
Type "show copying" to see the conditions.
There is absolutely no warranty for GDB. Type "show warranty" for details.
This GDB was configured as "i386-redhat-linux-gnu"...Using host libthread_db library "/lib/tls/libt
Core was generated by `/home/battist/flukacourse/Pavia/examples/errors/flukamy'.
Program terminated with signal 6, Aborted.
Reading symbols from /usr/lib/libg2c.so.0...done.
Loaded symbols for /usr/lib/libg2c.so.0
Reading symbols from /lib/tls/libm.so.6...done.
Loaded symbols for /lib/tls/libm.so.6
Reading symbols from /lib/libgcc_s.so.1...done.
Loaded symbols for /lib/libgcc_s.so.1
Reading symbols from /lib/tls/libc.so.6...done.
Loaded symbols for /lib/tls/libc.so.6
Reading symbols from /lib/ld-linux.so.2...done.
Loaded symbols for /lib/ld-linux.so.2
#0  0x009c37a2 in _dl_sysinfo_int80 () from /lib/ld-linux.so.2
```


Using gdb (2)

3) give the command bt (or where)



```
(gdb) bt
#0 0x009c37a2 in _dl_sysinfo_int80 () from /lib/ld-linux.so.2
#1 0x00a02e59 in raise () from /lib/tls/libc.so.6
#2 0x00a04882 in abort () from /lib/tls/libc.so.6
#3 0x005f0baf in sig_die () from /usr/lib/libg2c.so.0
#4 0x005f0c4b in f_setarg () from /usr/lib/libg2c.so.0
#5 <signal handler called>
#6 0x080496f4 in source_ (nomore=0x1) at source.f:123
BFD: BFD 20040223 20040223 assertion fail /usr/src/build/392707-i386/BUILD/gdb+dejagnum-20040223/bfd/libbfd.c:551
BFD: BFD 20040223 20040223 assertion fail /usr/src/build/392707-i386/BUILD/gdb+dejagnum-20040223/bfd/libbfd.c:551
BFD: BFD 20040223 20040223 assertion fail /usr/src/build/392707-i386/BUILD/gdb+dejagnum-20040223/bfd/libbfd.c:551
BFD: BFD 20040223 20040223 assertion fail /usr/src/build/392707-i386/BUILD/gdb+dejagnum-20040223/bfd/libbfd.c:551
BFD: BFD 20040223 20040223 assertion fail /usr/src/build/392707-i386/BUILD/gdb+dejagnum-20040223/bfd/libbfd.c:551
#7 0x0806cb04 in feeder_ (kendcn=0x91520e4) at feeder.FOR:186
BFD: BFD 20040223 20040223 assertion fail /usr/src/build/392707-i386/BUILD/gdb+dejagnum-20040223/bfd/libbfd.c:551
BFD: BFD 20040223 20040223 assertion fail /usr/src/build/392707-i386/BUILD/gdb+dejagnum-20040223/bfd/libbfd.c:551
BFD: BFD 20040223 20040223 assertion fail /usr/src/build/392707-i386/BUILD/gdb+dejagnum-20040223/bfd/libbfd.c:551
BFD: BFD 20040223 20040223 assertion fail /usr/src/build/392707-i386/BUILD/gdb+dejagnum-20040223/bfd/libbfd.c:551
BFD: BFD 20040223 20040223 assertion fail /usr/src/build/392707-i386/BUILD/gdb+dejagnum-20040223/bfd/libbfd.c:551
BFD: BFD 20040223 20040223 assertion fail /usr/src/build/392707-i386/BUILD/gdb+dejagnum-20040223/bfd/libbfd.c:551
BFD: BFD 20040223 20040223 assertion fail /usr/src/build/392707-i386/BUILD/gdb+dejagnum-20040223/bfd/libbfd.c:551
BFD: BFD 20040223 20040223 assertion fail /usr/src/build/392707-i386/BUILD/gdb+dejagnum-20040223/bfd/libbfd.c:551
#8 0x0804d2cb in flukam_ (iflgeo=0x864e948) at flukam.FOR:3228
#9 0x080492f2 in MAIN__ () at fluka.FOR:278
#10 0x0864e836 in main ()
```

Here it is!
frame #6 in source.f
at line 123



Using gdb (3)

4) give frame 6

```
(gdb) frame 6
#6  0x080496f4 in source_ (nomore=0x1) at source.f:123
123      PMOFLK (NPFLKA) = SQRT ( TKEFLK (NPFLKA) * ( TKEFLK (NPFLKA)
Current language:  auto; currently fortran
```

5) you can inspect variable content, for instance:

```
(gdb) p TKEFLK (NPFLKA)
$1 = -0.1541494786490023
```

A negative kinetic energy!!!!

Using gdb (4)

6) you can list the nearby lines:

```
(gdb) list
```

```
118     IF ( ABS(RGAUSS) .GT. 3.D+00) GO TO 555
119     *   TKEFLK (NPFLKA) = SQRT ( PBEAM**2 + AM (IONID)**2 ) - AM (IONID)
120     TKEFLK (NPFLKA) = WHASOU(1) + WHASOU(2)*RGAUSS
121     * Particle momentum
122     *   PMOFLK (NPFLKA) = PBEAM
123     PMOFLK (NPFLKA) = SQRT ( TKEFLK (NPFLKA) * ( TKEFLK (NPFLKA)
124     &   + TWOTWO * AM (ILOFLK(NPFLKA)) ) )
125     WRITE(*,*) TKEFLK (NPFLKA), PMOFLK (NPFLKA)
126     * Cosines (tx,ty,tz)
127     TXFLK (NPFLKA) = UBEAM
```

It's quite obvious that this was not a protected statement!



Using gdb (5)

- In the remaining 5% of cases there will be an error inside a FLUKA routine.
- In that case, if you have not the source, prepare a report with the gdb results and send it to fluka-discuss@fluka.org together with a tar file containing: input, user routines (if any), additional auxiliary files, last random seed, and any other possible useful information.



Using gdb (6)

- A bit of warning for experienced users: the compiling options of FLUKA scripts call for optimization
- In these conditions, if you want to follow step by step what happens inside a routine of yours, the debugger “seems confused”: you are not sure of the line that you are looking to
- If this is your need, then prepare a different version of compiling/linking script (fff, lfluka) substituting the **-O** option with **-O0**, then recompile and relink



Other cases:

In case you are using SOURCE, and by mistake you sample an energy (or momentum) larger than the WHAT(1) value of the BEAM card (maximum of dE/dx tabulation), then an ABORT will be generated at run time.

The *.out file will contain the following message:

```
Abort called from STEPOP reason dp/dx=<0 Run stopped!  
STOP dp/dx=<0
```



Other cases (2):

- **An error in reading Unit 14 refers to nuclear.bin:** your nuclear.bin file might be corrupted or missing (check all your binary files in \$FLUPRO)
- **Error in reading Unit 1: a problem in the random seed!** Often in case when you start in with rfluka using `-N n` with $n > 0$: **check for the existence of `ran***n` in the launch directory!**



Other cases (3):

```
**** No Random file available !!!!! ****  
Abort called from FLRM64 reason NO RANDOM FILE Run stopped!  
STOP NO RANDOM FILE
```

The above message occurs whenever the seed for the run is not available (or is corrupted).

Example: if, for instance you ask for n runs, and run $\#n-k$ ends with an error, then the seed for run $\#n-k+1$ is not produced and therefore all runs from $n-k+1$ to n will crash in this way...

The *.err file: is does not contain necessarily error messages!

The *.err file will report errors, but also a lot of warning which have a meaning mostly for the developers.

For instance: the following messages are not errors!

```
*** Frmbrk: we are dealing with a bag of 8 164.306992
*** Frmbrk: we are dealing with a bag of 7 186.47261
*** Frmbrk: a bag of 10 identical nucleons, cannot be managed ***
```

```
NEXT SEEDS:171B5708      1      0      0      0      0 33B49B1      0      0      0
220000      780000      780000      1.6305137E-02      1.00
NEXT SEEDS:1E5C731D      1      0      0      0      0 33B49B1      0      0      0
*** EVENTD: IJ,IBAR(IJ),ICH(IJ),IBTAR,ICHTAR,PPERNU -6 4 2 12 6 67.0629729
ECKDPM,PXKDPM,PYKDPM,PZKDPM -2.61119861 0.000579929462 -0.0154642238
-2.52471092
KP,IBRSNC(KP),ICRSNC(KP),TVRSNC(KP),ANRSNC(KP),EKRSNC(KP) 1 9 5 0.0885864878
8.39381037 0.00395470202
```

No. of events simulated so far

No. of events remaining to be simulated



A malicious case (expecially for old users)

- This is not exactly an error, and we have probably to correct the FLUKA behaviour for this.
- What happens if you use **EMF-CUT** without never giving the card with **SDUM=PROD-CUT**?



You are going to run FLUKA with unpredicted EMF cuts but there is no crash or stop



Doubts on the results?

If you really cannot understand, if you need to ask about physics related problems, once again write to fluka-discuss@fluka.org