Managing FLUKA Simulation Output Files for FOOT

G.B. S.M.

Available (lightweight) files in **CNAO2020 campaign**

/gpfs_data/local/foot/Simulation/Tutorial/CNAO2020

/gpfs_data/local/foot/Simulation/Tutorial/Fulll

Full detector (magnets etc.)

^{12}C at 200 MeV/u on C (5 mm ρ=1.83 g/cm^3):

12C_C_200.root *(from Txt2Root)*

12C_C_200shoe.root *(from Txt2NtuRoot)*

CNAO2020: 1339 events

Full: 1706 events

Introduction

This short tutorial is meant to explain how to use the MC data output produced for FOOT *(Electronic Spectrometer)*.

The purpose is not to teach how to perform a correct FOOT Simulation using FLUKA. In case there are people interested in that, a dedicated course can be organized, provided that they have previously attended at least a FLUKA Basic Course

The main topics today are:

- **Give some basic info specific of FLUKA MC what everybody needs to know**
- **The structure of data produced by MC for FOOT**
- **Provide examples about the use and interpretation of these data, and the connection of detector hits and particle properties at MC-truth level**

A few specific things of FLUKA MC that you need to know

Default units

the most important are:

time→ s, length → cm, energy → GeV, momentum → GeV/c

masses → GeV/c^2 B → Tesla

Reference frame: (cartesian, right-handed)

z is primary beam direction
y is pointing upwards

It coincides with the global reference frame used in SHOE, with origin (0,0,0,) <u>at the center of target</u>

Particles:

each particle is identified by a number

A few specific things of FLUKA MC that you need to know

Fluka name	Fluka no.	Common name
4-HELIUM	-6	Alpha
3-HELIUM	-5	Helium-3
TRITON	-4	Triton
DEUTERON	-3	Deuteron
HEAVYION	-2	Generic heavy ion with Z > 2
OPTIPHOT	-1	Optical Photon
RAY	0	Pseudoparticle
PROTON	1	Proton
APROTON	2	Antiproton
ELECTRON	3	Electron
POSITRON	4	Positron
NEUTRIE	5	Electron Neutrino
ANEUTRIE	6	Electron Antineutrino
PHOTON	7	Photon
NEUTRON	8	Neutron
ANEUTRON	9	Antineutron
MUON+	10	Positive Muon
MUON-	11	Negative Muon

Fluka name	Fluka no.	Common name
PION+	13	Positive Pion
PION-	14	Negative Pion
KAON+	15	Positive Kaon
KAON-	16	Negative Kaon
LAMBDA	17	Lambda
ALAMBDA	18	Antilambda
KAONLONG	12	Kaon-zero long
KAONSHRT	19	Kaon zero short
NEUTRIM	27	Muon neutrino
ANEUTRIM	28	Muon antineutrino
TAU+	41	Positive Tau
TAU-	42	Negative Tau
NEUTRIT	43	Tau neutrino
ANEUTRIT	44	Tau antineutrino

Here only the most importat

Since we are mostly interested to nuclear fragments, notice:

for p, n, d, t, ^{3}He, ^{4}He there is a specific FLUKA particle number

For A>4: FLUKA particle numbers is always -2, and nucleus is identified by Z and A

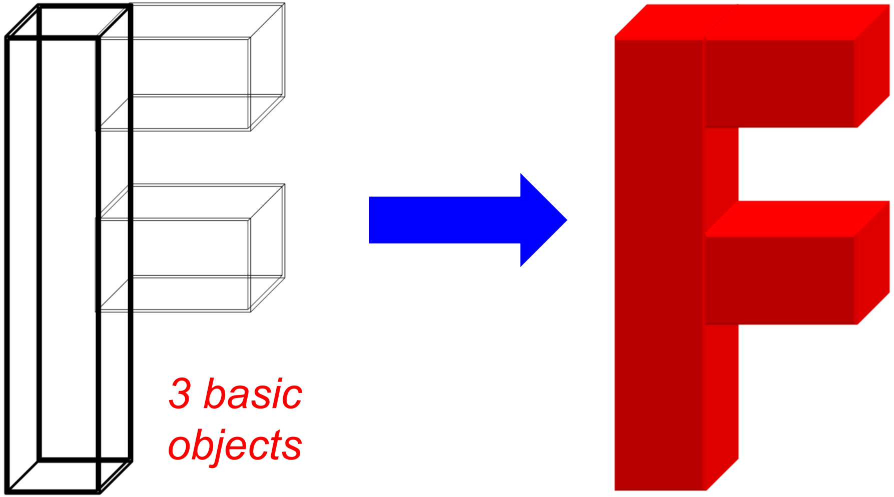
Very low energy fragments and nucleons originating in the "nuclear evaporation" phase are identified with a particle number in the range from -39 to -7. Again identified by Z and A.

there would be also a way to identify isomers, but we can omit this now

The concept of “Region”

FLUKA “Combinatorial Geometry”

Basic objects called bodies (such as cylinders, spheres, parallelepipeds, etc.) are combined to form more complex objects called **Regions**



1 complex object **= REGION**

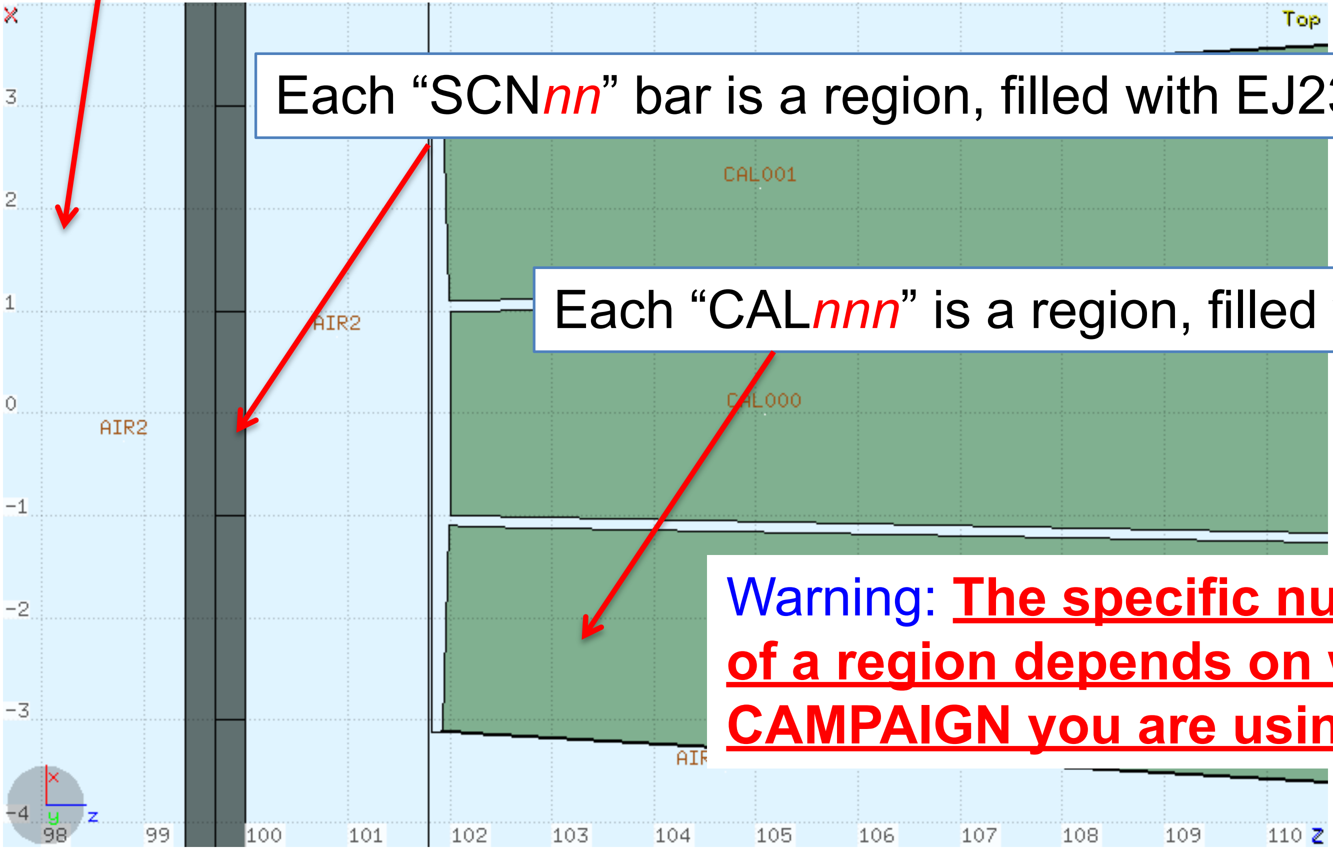
- The user knows the region usually by **name**, but internally (and in SHOE) it is identified by a **number**

- to each region is assigned a single **Material** (chemical element or compound or mixture)

"AIR2" is a region, filled with air (N, O, Ar @ STP)

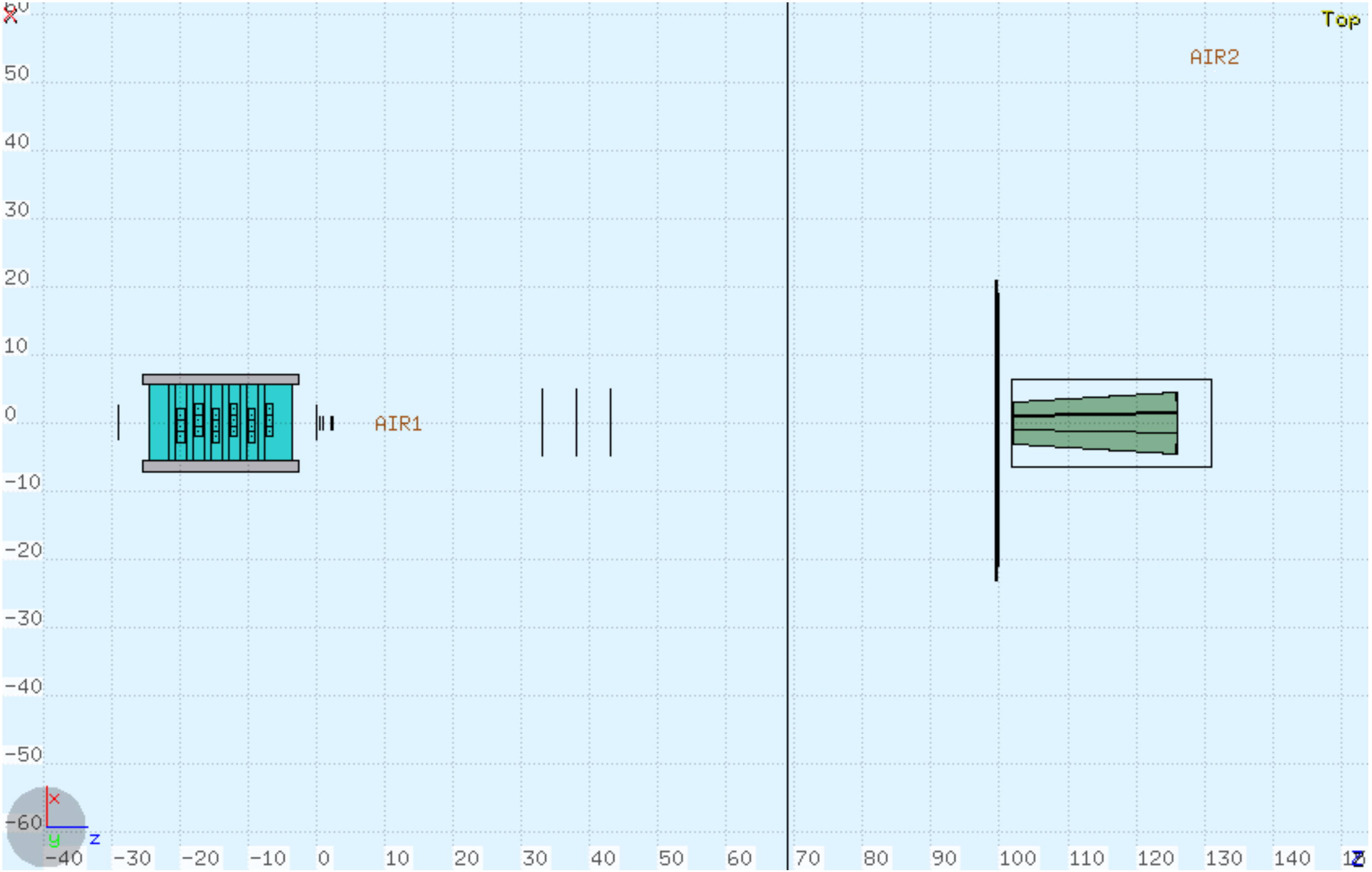
Each "SCN*nn*" bar is a region, filled with EJ232 scintillator

Each "CAL*nnn*" is a region, filled with BGO



Warning: **<u>The specific number identifier of a region depends on which CAMPAIGN you are using!</u>**

Detector in CNAO2020 Campaign



Where the FOOT user can retrieve relevant infos about geometry and materials used in simulation

For a given Campaign XXX:

In shoe/build/Reconstruction/level0/config/XXXX/FootGlobal.par
you see the detectors selected for simulation (*y* or *n* in a list)

In shoe/build/Reconstruction/level0/geomaps there are:

- FOOT_geo.map which contains the positions (of the "center"), dimensions and rotation angles in global coordinates of all FOOT detectors and magnets
- TA*detector.map which contain, for each single detector (or magnet system), the relative coordinates and rotation angle of every element composing the detector itself, together with the material description.
- TAGdetector.map contains info about target and primary beam

Example from config/CNAO2020

inside **FootGlobal.par** you can see:

```
IncludeDI:                                  n
IncludeST:                                  y
IncludeBM:                                  y
IncludeTG:                                  y
IncludeVT:                                  y
IncludeIT:                                  n
IncludeMSD:                                 y
IncludeTW:                                  y
IncludeCA:                                  y
```

Examples from geomaps

FOOT_geo.map

```
StartBaseName: "ST"
StartPosX: 0. StartPosY: 0. StartPosZ: -29.
StartAngX: 0. StartAngY: 0. StartAngZ: 0.

TargetBaseName: "TG"
TargetPosX: 0. TargetPosY: 0. TargetPosZ: 0.
TargetAngX: 0. TargetAngY: 0. TargetAngZ: 0.

BmBaseName: "BM"
BmPosX: 0. BmPosY: 0. BmPosZ: -14.
BmAngX: 0. BmAng:Y 0. BmAngZ:   0.

VertexBaseName: "VT"
VertexPosX: 0. VertexPosY:   0. VertexPosZ: 1.5
VertexAngX: 0. VertexAngY:   0. VertexAngZ: 0.

MagnetsBaseName: "DI"
MagnetsPosX: 0. MagnetsPosY:   0. MagnetsPosZ: 16.5
MagnetsAngX: 0. MagnetsAngY:   0. MagnetsAngZ: 0.

InnerTrackerBaseName: "IT"
InnerTrackerPosX: 0. InnerTrackerPosY: 0. InnerTrackerPosZ: 16.5
InnerTrackerAngX: 0. InnerTrackerAngY: 0. InnerTrackerAngZ: 0.

MultiStripBaseName: "MSD"
MultiStripPosX: 0. MultiStripPosY: 0. MultiStripPosZ: 38.02
MultiStripAngX: 0. MultiStripAngY: 0. MultiStripAngZ: 0.

TofWallBaseName: "TW"
TofWallPosX: -1. TofWallPosY:  -1. TofWallPosZ: 99.7
TofWallAngX: 0. TofWallAngY:  0. TofWallAngZ: 0.

CaloBaseName: "CA"
CaloPosX: 0. CaloPosY:  0. CaloPosZ: 114
CaloAngX: 0. CaloAngY:  0. CaloAngZ: 0.
```

TATWdetector.map

```
// -+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-
// Parameters of the TW
// -+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-
LayersN:      2
BarsN:        20
Material:   "EJ232"
Density:     1.023
Excitation: 4.8e-5
BirkFac:     0.0138

// -+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-
// Parameter of the TW (cm)
// -+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-
Width:         2.0      Height:      44.0       Thick:     0.3

// -+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-
// Parameter of the Detector bar used in the run
// -+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-
// vertical layer
Layer:              0
Bar:                0
PositionX: 19.0  PositionY:    0.0 PositionZ: 0.15
TiltX:       0.0  TiltY:        0.0 TiltZ:       0.0
Layer:              0
Bar:                1
PositionX: 17.0  PositionY:    0.0 PositionZ: 0.15
TiltX:       0.0  TiltY:        0.0 TiltZ:       0.0
Layer:              0
Bar:                2
PositionX: 15.0  PositionY:    0.0 PositionZ: 0.15
TiltX:       0.0  TiltY:        0.0 TiltZ:       0.0
Layer:              0
```

Configuration of Magnets

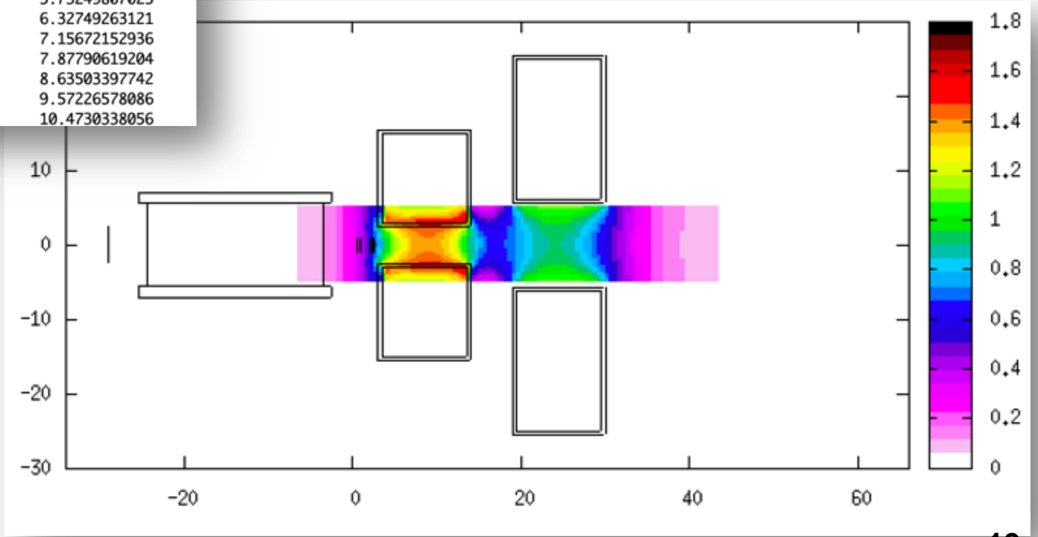
magnetic map name and numbers for magnets are in geomaps/XXXX/TADIdetector.map.

88641 21 21 201

x	y	z	Bx	By	Bz
-5.00000000000	-5.00000000000	-50.0000000000	1.36243669343	-30.6770814637	0.459840156935E-16
-5.00000000000	-5.00000000000	-49.5000000000	1.35464451635	-30.8022112988	0.528218691543
-5.00000000000	-5.00000000000	-49.0000000000	1.34685233926	-30.9273411338	1.05643738309
-5.00000000000	-5.00000000000	-48.5000000000	1.35906061221	-31.0238462683	1.64156333646
-5.00000000000	-5.00000000000	-48.0000000000	1.40067845000	-31.2369115992	2.25326833731
-5.00000000000	-5.00000000000	-47.5000000000	1.43021238978	-31.4820544068	2.78929569702
-5.00000000000	-5.00000000000	-47.0000000000	1.45974632956	-31.7271972143	3.32532305672
-5.00000000000	-5.00000000000	-46.5000000000	1.48928026933	-31.9723400218	3.86135041643
-5.00000000000	-5.00000000000	-46.0000000000	1.56924075328	-32.2800339546	4.50091052135
-5.00000000000	-5.00000000000	-45.5000000000	1.65712860643	-32.6735541981	5.15804511664
-5.00000000000	-5.00000000000	-45.0000000000	1.71563615145	-33.1755302923	5.73249807025
-5.00000000000	-5.00000000000	-44.5000000000	1.80905040239	-33.6194597825	6.32749263121
-5.00000000000	-5.00000000000	-44.0000000000	1.93494986313	-34.1492021605	7.15672152936
-5.00000000000	-5.00000000000	-43.5000000000	2.04046917779	-34.7722498674	7.87790619204
-5.00000000000	-5.00000000000	-43.0000000000	2.20446996652	-35.4680040321	8.63503397742
-5.00000000000	-5.00000000000	-42.5000000000	2.38506450881	-36.3909573966	9.57226578086
-5.00000000000	-5.00000000000	-42.0000000000	2.47932864781	-37.4124786747	10.4730338056

The map of magnetic file is contained in shoe/build/Reconstruction/fullrec/data

(at present is "AsymmetricDipoles.table")

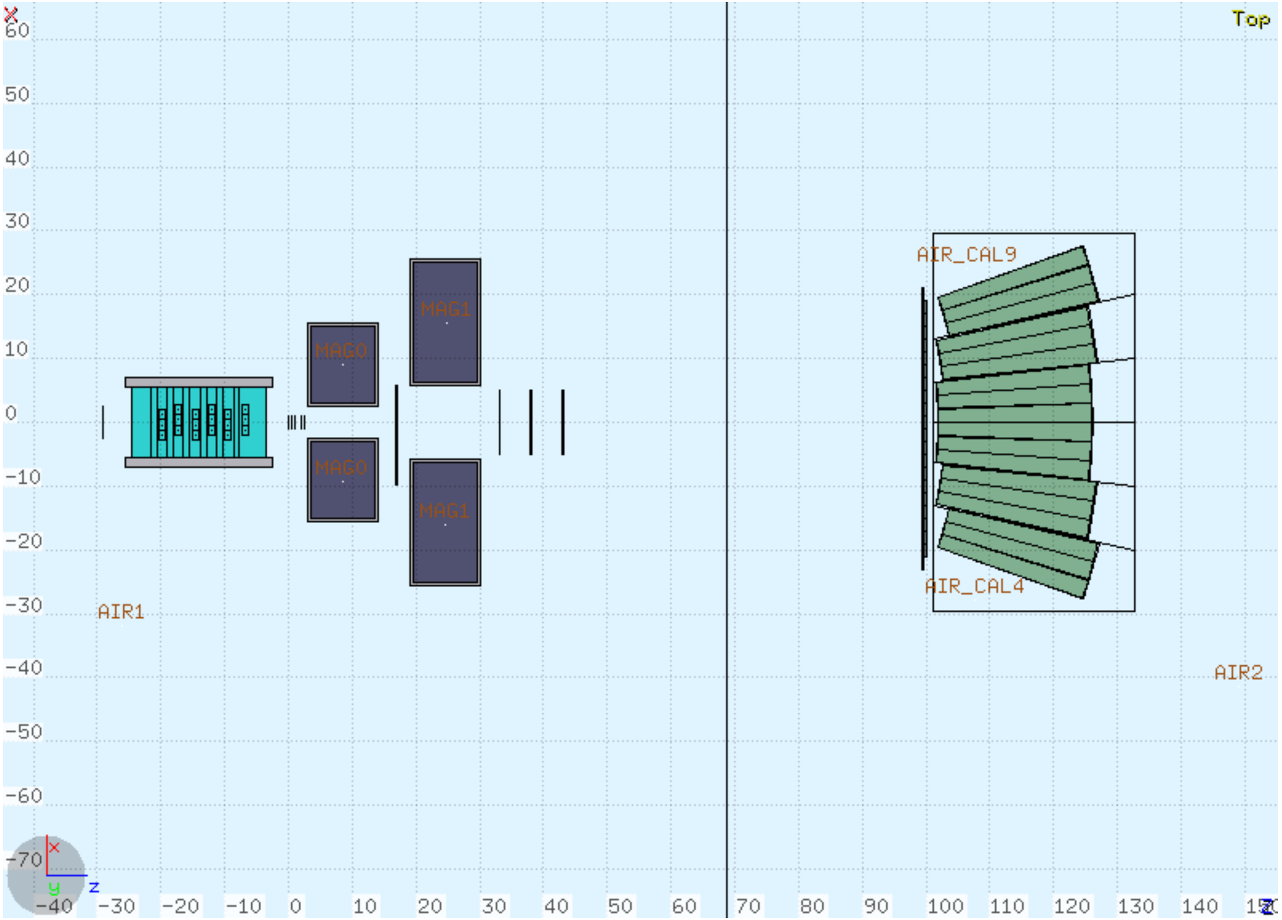


Region Numbering for CNAO2020 Campaign

At present not availble in SHOE
It can be useful in MC studies

Region n.	1	BLACK	"Black Hole"
Region n.	2	AIR1	Air
Region n.	3	AIR2	Air
Region n.	4	AIR_CAL0	Air around Calo
Region n.	5	STC	Start Counter
Region n.	6	STCMYL1	Mylar foil in front of Start Counter
Region n.	7	STCMYL2	Mylar foil on the back of SC
Region n.	8	BMN_SHI	BM Al Shield
Region n.	9	BMN_MYL0	BM Mylar foil at the entrance
Region n.	10	BMN_MYL1	BM Mylar foil at the exit
Region n.	11-46	BMN_C000 – BMN_C117	BM Cells
Region n.	47	BMN_FWI	BM Field wires
Region n.	48	BMN_SWI	BM Sense wires
Region n.	49	BMN_GAS	BM gas (non – sensitive)
Region n.	50	TARGET	Target
Region n.	51-62	VTXE0 – VTXP3	All different parts of VTX
Region n.	63-80	MSDS0 – MSDM4	All different parts of MSD
Region n.	81-120	SCN000 – SCN119	TW bars
Region n.	121-129	CAL000 – CAL008	BGO crystals
Region n.	130	ACAL_00	AIR around the BGO crystals

Detector Full

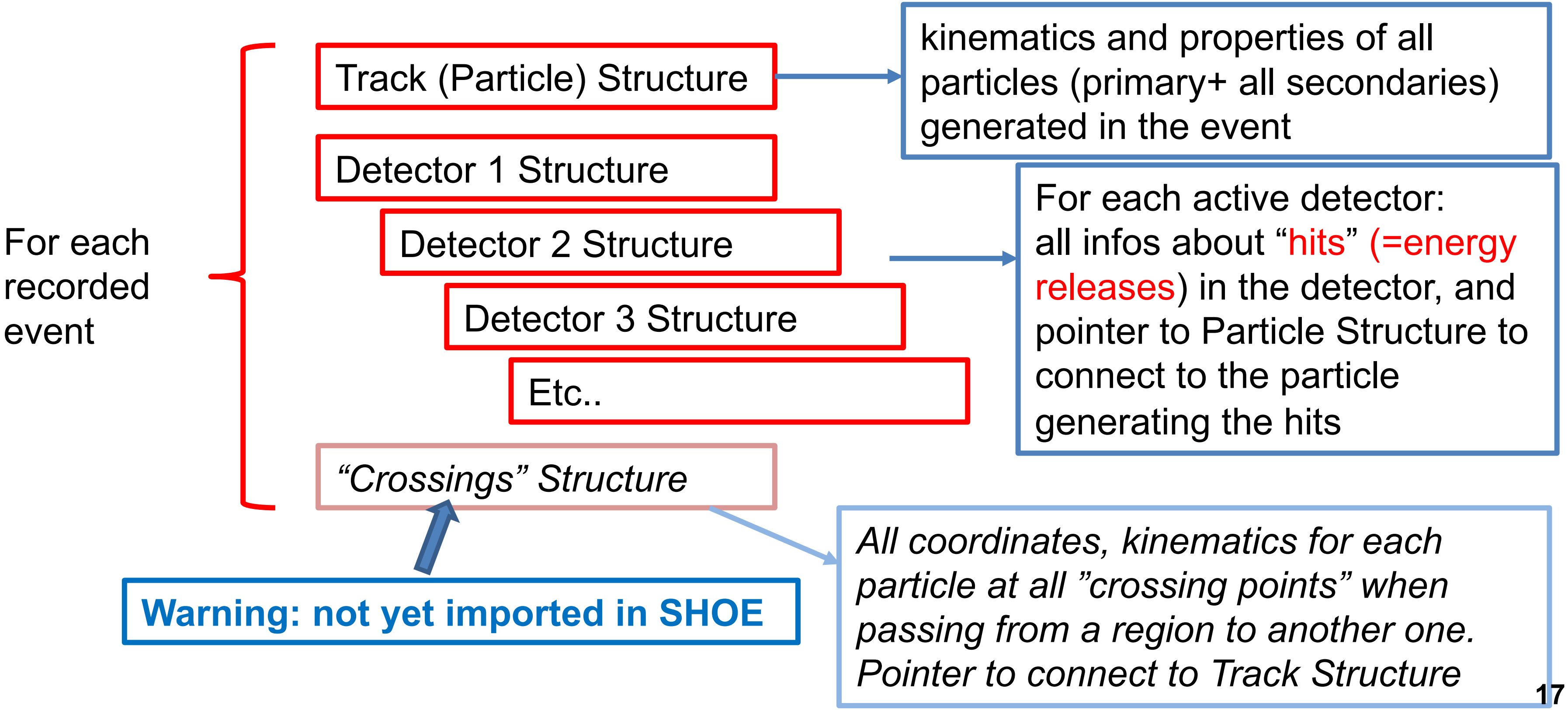


Region Numbering for FULL Campaign

At present not availble in SHOE
It can be useful in MC studies

Region n.	1	BLACK	“Black Hole”
Region n.	2	AIR1	Air
Region n.	3	AIR2	Air
Region n.	4-13	AIR_CAL0 - AIR_CAL9	Air around Calo
Region n.	14	STC	Start Counter
Region n.	15	STCMYL1	Mylar foil in front of Start Counter
Region n.	16	STCMYL2	Mylar foil on the back of SC
Region n.	17	BMN_SHI	BM Al Shield
Region n.	18	BMN_MYL0	BM Mylar foil at the entrance
Region n.	19	BMN_MYL1	BM Mylar foil at the exit
Region n.	20-55	BMN_C000 - BMN_C117	BM Cells
Region n.	56	BMN_FWI	BM Field wires
Region n.	57	BMN_SWI	BM Sense wires
Region n.	58	BMN_GAS	BM gas (non – sensitive)
Region n.	59	TARGET	Target
Region n.	60-71	VTXE0 - VTXP3	All different parts of VTX
Region n.	72-219	ITRE00 - ITRY112	All different parts of ITR
Region n.	220-237	MSDS0 - MSDM5	All different parts of MSD
Region n.	238-241	MAG0 – MAG_SH1	Different parts of the Magnets
Region n.	242-281	SCN000 - SCN119	TW bars
Region n.	282-569	CAL000 - CAL287	BGO crystals
Region n.	570-601	ACAL_00 - ACAL_31	AIR around the BGO crystals

The EVENT_STRUCT content (the one adopted in the traditional MC files)



The particle structure

for each of the produced particles we register the info in arrays: i.e.
TRmass[2] is the mass of the 3rd produced particle

EventNumber = FLUKA event number:

TRn= number of particles produced in the event

TRpaid= index in the part common of the particle parent

TRcha = charge (Z)

TRbar = barionic number (A)

TRfid = FLUKA code for the particle (*for example: photon=7*)

TRgen = generation number

TRreg = number of the region where the particle has been produced

TRix, TRiy, TRiz = production position of the particle (cm)

TRfx, TRfy, TRfz = final position of the particle (cm)

TRipx,TRipy,TRipz = production momentum of the particle (GeV/c)

TRifx,TRfpy,TRfpz = final momentum of the particle (GeV/c)

TRmass = particle mass (GeV/c^2)

TRtime = production time of the particle (s)

TRtof = time between death and birth of the particle (s)

TRtrlen = Track lenght of the particle (cm)

```
Int_t EventNumber;
Int_t TRn;
Int_t TRpaid[MAXTR];
Int_t TRgen[MAXTR];
Int_t TRcha[MAXTR];
Int_t TRreg[MAXTR];
Int_t TRbar[MAXTR];
Int_t TRdead[MAXTR];
Int_t TRfid[MAXTR];
Double_t TRix[MAXTR];
Double_t TRiyi[MAXTR];
Double_t TRiz[MAXTR];
Double_t TRfx[MAXTR];
Double_t TRfy[MAXTR];
Double_t TRfz[MAXTR];
Double_t TRipx[MAXTR];
Double_t TRipy[MAXTR];
Double_t TRipz[MAXTR];
Double_t TRfpx[MAXTR];
Double_t TRfpy[MAXTR];
Double_t TRfpz[MAXTR];
Double_t TRmass[MAXTR];
Double_t TRtime [MAXTR];
Double_t TRtof[MAXTR];
Double_t TRlen[MAXTR];
```

Retrieving the MC particle structure in SHOE

When processing a rootple obtained after processing with DecodeMC, you can use in your macro the methods defined in shoe/libs/src/TAMCbase (TAMCntuEve.hxx, TAMCntuEve.cxx)

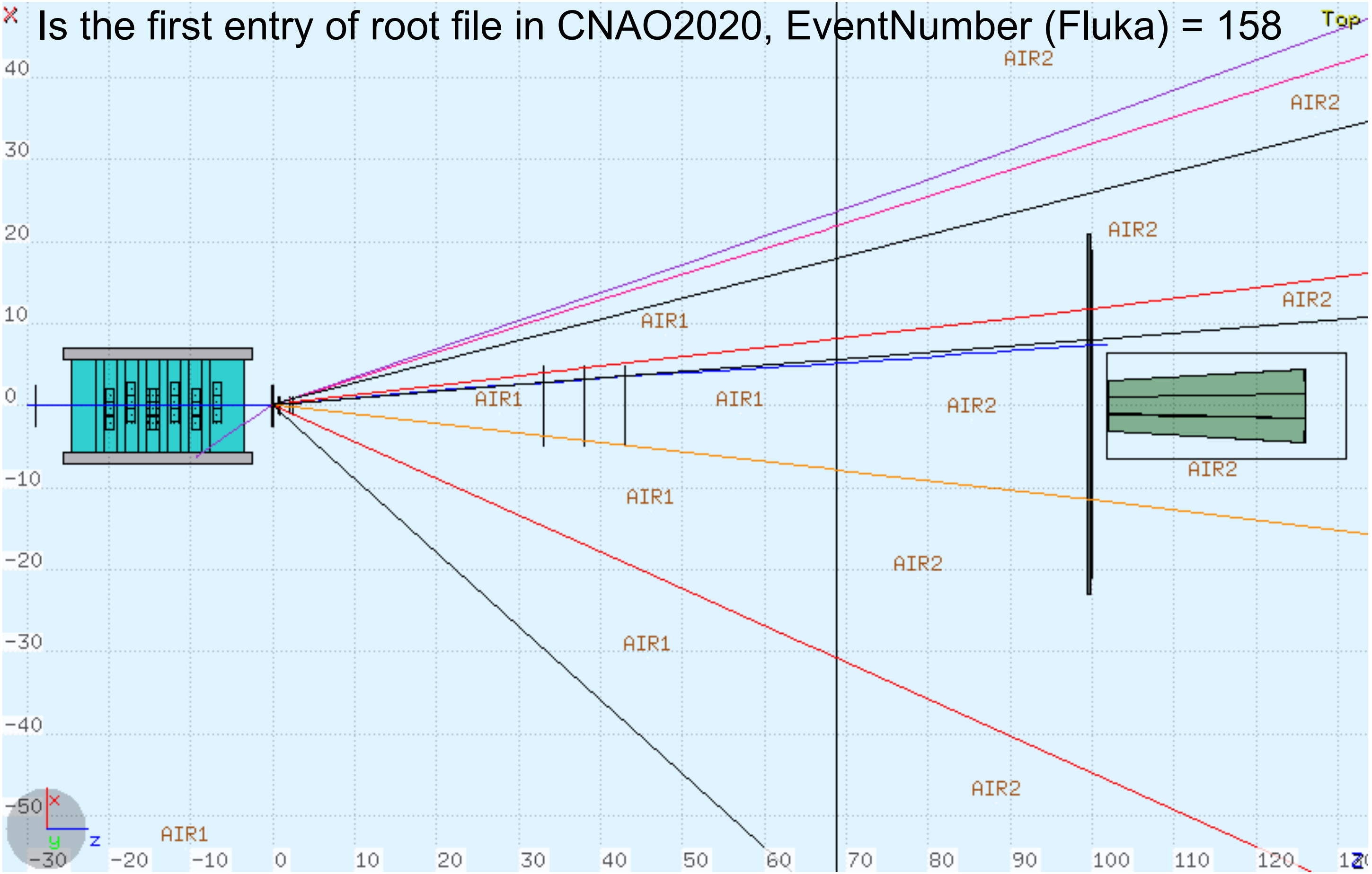
For example:

thanks to Y.D.

```
TTree *tree;
 tree->SetBranchAddress(TAMCntuEve::GetBranchName(), &mcNtuEve);
TAMCntuEve *mcNtuEve;
mcNtuEve = new TAMCntuEve(); gets the Event Structure
….
Somewhere inside a Loop on the events:
….
int  Nmctrack = mcNtuEve->GetTracksN();        retrieves TRn
for( Int_t iTrack = 0; iTrack < mcNtuEve->GetTracksN(); ++iTrack ) {  loop on the tracks in the event
   TAMCeveTrack* track = mcNtuEve->GetTrack(iTrack);   gets the track
   Int_t FLid = track->GetFlukaID();                  retrieves TRfid[iTrack]
   Int_t Mid = track->GetMotherID();                  retrieves TRpaid[iTrack]-1
   Int_t Charge = track->GetCharge();                 retrieves TRcha[iTrack]
   Int_t BarNum = track->GetBaryon();                 retrieves TRbar[iTrack]
   Int_t BarNum = track->GetRegion();                 retrieves TRreg[iTrack]
   Double_t Mass = track->GetMass();                  retrieves TRmass[iTrack]
   TVector3 InitPos = track->GetInitPos();            retrieves TRix[iTrack], TRiy[iTrack], TRiz[iTrack]
   TVector3 FinalPos = track->GetFinalPos();          retrieves TRfx[iTrack], TRfy[iTrack], TRfz[iTrack]
   TVector3 InitP = track->GetInitP();                retrieves TRipx[iTrack], TRipy[iTrack], TRipz[iTrack]
   TVector3 FinalP = track->GetFinalP();              retrieves TRfpx[iTrack], TRfpy[iTrack], TRfpz[iTrack]
   etc. etc.
```

An example

Is the first entry of root file in CNAO2020, EventNumber (Fluka) = 158



id = 0 is the primary. The first track in the structure (row=0)

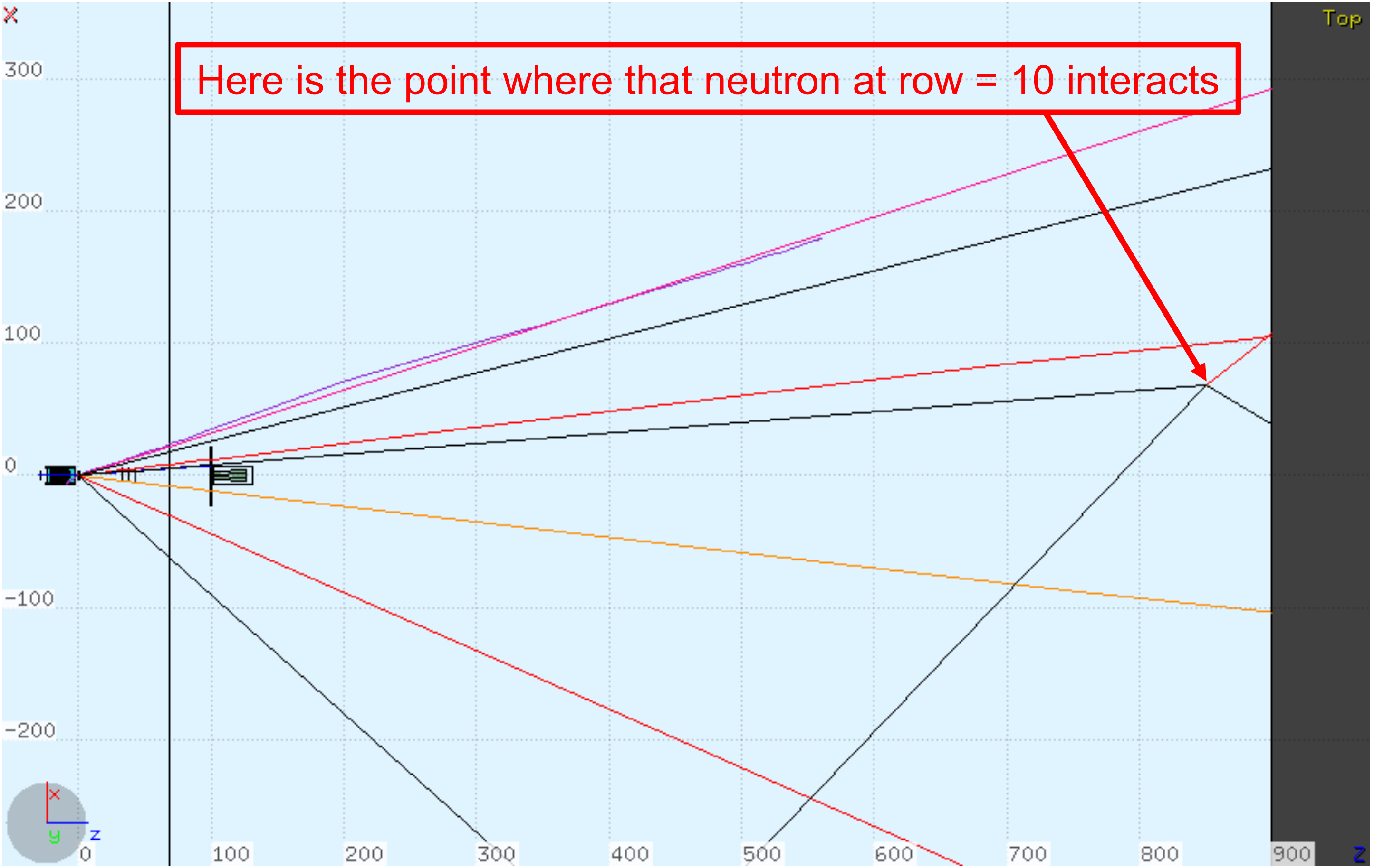
```
root [2] EventTree->Scan("TRn:TRpaid:TRfid:TRcha:TRbar:TRiz:TRfz","EventNumber==158")
********************************************************************************************************
*    Row   * Instance *       TRn *     TRpaid *      TRfid *      TRcha *      TRbar *       TRiz *       TRfz *
********************************************************************************************************
*        0 *        0 *        18 *          0 *         -2 *          6 *         12 *        -30 * -0.082383 *
*        0 *        1 *        18 *          1 *         -3 *          1 *          2 * -0.082383 * -9.383039 *
*        0 *        2 *        18 *          1 *         -3 *          1 *          2 * -0.082383 * 560.54650 *
*        0 *        3 *        18 *          1 *         -2 *          3 *          7 * -0.082383 * 101.68155 *
*        0 *        4 *        18 *          1 *         -5 *          2 *          3 * -0.082383 *       900 *
*        0 *        5 *        18 *          1 *         -6 *          2 *          4 * -0.082383 *       900 *
*        0 *        6 *        18 *          1 *          1 *          1 *          1 * -0.082383 *       900 *
*        0 *        7 *        18 *          1 *          1 *          1 *          1 * -0.082383 *       900 *
*        0 *        8 *        18 *          1 *          8 *          0 *          1 * -0.082383 *       900 *
*        0 *        9 *        18 *          1 *          8 *          0 *          1 * -0.082383 *       900 *
*        0 *       10 *        18 *          1 *          8 *          0 *          1 * -0.082383 * 850.46374 *
*        0 *       11 *        18 *         11 *         -6 *          2 *          4 * 850.46374 * 851.90222 *
*        0 *       12 *        18 *         11 *         -6 *          2 *          4 * 850.46374 * 850.58166 *
*        0 *       13 *        18 *         11 *         -6 *          2 *          4 * 850.46374 * 850.41534 *
*        0 *       14 *        18 *         11 *          8 *          0 *          1 * 850.46374 * -71.90350 *
*        0 *       15 *        18 *         11 *          1 *          1 *          1 * 850.46374 *       900 *
*        0 *       16 *        18 *         11 *          8 *          0 *          1 * 850.46374 *       900 *
*        0 *       17 *        18 *          1 *          1 *          1 *          1 * -0.082383 * -0.092312 *
********************************************************************************************************
```

For id>0: **id-1 points to the parent particle** (row id-1 in the structure)

All particles with id = 1 have been generated by the primary (id=0)

```
root [2] EventTree->Scan("TRn:TRpaid:TRfid:TRcha:TRbar:TRiz:TRfz","EventNumber==158")
*********************************************************************************************************
*    Row   * Instance *        TRn *     TRpaid *      TRfid *      TRcha *      TRbar *       TRiz *       TRfz *
*********************************************************************************************************
*        0 *        0 *         18 *          0 *         -2 *          6 *         12 *        -30 * -0.082383 *
*        0 *        1 *         18 *          1 *         -3 *          1 *          2 * -0.082383 * -9.383039 *
*        0 *        2 *         18 *          1 *         -3 *          1 *          2 * -0.082383 * 560.54650 *
*        0 *        3 *         18 *          1 *         -2 *          3 *          7 * -0.082383 * 101.68155 *
*        0 *        4 *         18 *          1 *         -5 *          2 *          3 * -0.082383 *       900 *
*        0 *        5 *         18 *          1 *         -6 *          2 *          4 * -0.082383 *       900 *
*        0 *        6 *         18 *          1 *          1 *          1 *          1 * -0.082383 *       900 *
*        0 *        7 *         18 *          1 *          1 *          1 *          1 * -0.082383 *       900 *
*        0 *        8 *         18 *          1 *          8 *          0 *          1 * -0.082383 *       900 *
*        0 *        9 *         18 *          1 *          8 *          0 *          1 * -0.082383 *       900 *
*        0 *       10 *         18 *          1 *          8 *          0 *          1 * -0.082383 * 850.46374 *
*        0 *       11 *         18 *         11 *         -6 *          2 *          4 * 850.46374 * 851.90222 *
*        0 *       12 *         18 *         11 *         -6 *          2 *          4 * 850.46374 * 850.58166 *
*        0 *       13 *         18 *         11 *         -6 *          2 *          4 * 850.46374 * 850.41534 *
*        0 *       14 *         18 *         11 *          8 *          0 *          1 * 850.46374 * -71.90350 *
*        0 *       15 *         18 *         11 *          1 *          1 *          1 * 850.46374 *       900 *
*        0 *       16 *         18 *         11 *          8 *          0 *          1 * 850.46374 *       900 *
*        0 *       17 *         18 *          1 *          1 *          1 *          1 * -0.082383 * -0.092312 *
*********************************************************************************************************
```

These particles with **id = 11** have been generated by the particle at row id-1 = 10 *(a neutron which interacts in air far away)*



Notice:

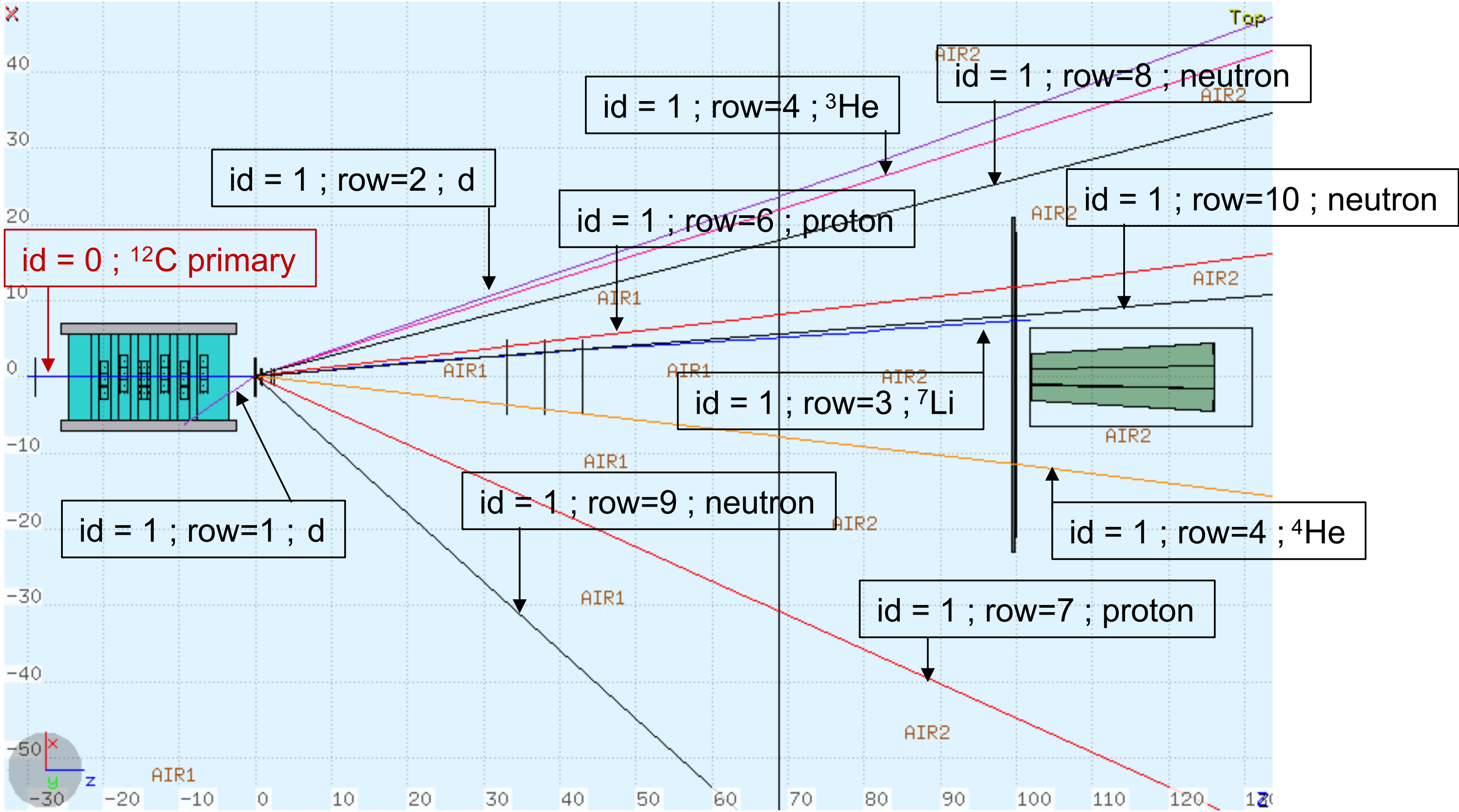
<u>At present simulation is generic and does not include a realistic room size: this means, for example, that no possible back-splash from walls is considered</u>

These particles exit from the geometry far away (z=900 cm)

```
root [2] EventTree->Scan("TRn:TRpaid:TRfid:TRcha:TRbar:TRiz:TRfz","EventNumber==158")
************************************************************************************************************
*    Row   * Instance *       TRn *    TRpaid *     TRfid *     TRcha *     TRbar *      TRiz *      TRfz *
************************************************************************************************************
*        0 *        0 *        18 *         0 *        -2 *         6 *        12 *       -30 * -0.082383 *
*        0 *        1 *        18 *         1 *        -3 *         1 *         2 * -0.082383 * -9.383039 *
*        0 *        2 *        18 *         1 *        -3 *         1 *         2 * -0.082383 * 560.54650 *
*        0 *        3 *        18 *         1 *        -2 *         3 *         7 * -0.082383 * 101.68155 *
*        0 *        4 *        18 *         1 *        -5 *         2 *         3 * -0.082383 *       900 *
*        0 *        5 *        18 *         1 *        -6 *         2 *         4 * -0.082383 *       900 *
*        0 *        6 *        18 *         1 *         1 *         1 *         1 * -0.082383 *       900 *
*        0 *        7 *        18 *         1 *         1 *         1 *         1 * -0.082383 *       900 *
*        0 *        8 *        18 *         1 *         8 *         0 *         1 * -0.082383 *       900 *
*        0 *        9 *        18 *         1 *         8 *         0 *         1 * -0.082383 *       900 *
*        0 *       10 *        18 *         1 *         8 *         0 *         1 * -0.082383 * 850.46374 *
*        0 *       11 *        18 *        11 *        -6 *         2 *         4 * 850.46374 * 851.90222 *
*        0 *       12 *        18 *        11 *        -6 *         2 *         4 * 850.46374 * 850.58166 *
*        0 *       13 *        18 *        11 *        -6 *         2 *         4 * 850.46374 * 850.41534 *
*        0 *       14 *        18 *        11 *         8 *         0 *         1 * 850.46374 * -71.90350 *
*        0 *       15 *        18 *        11 *         1 *         1 *         1 * 850.46374 *       900 *
*        0 *       16 *        18 *        11 *         8 *         0 *         1 * 850.46374 *       900 *
*        0 *       17 *        18 *         1 *         1 *         1 *         1 * -0.082383 * -0.092312 *
************************************************************************************************************
```

This proton has been generated by primary in the target, but dies in the target

Our example



id = 0 ; ¹²C primary
id = 1 ; row=1 ; d
id = 1 ; row=2 ; d
id = 1 ; row=3 ; ⁷Li
id = 1 ; row=4 ; ³He
id = 1 ; row=4 ; ⁴He
id = 1 ; row=6 ; proton
id = 1 ; row=7 ; proton
id = 1 ; row=8 ; neutron
id = 1 ; row=9 ; neutron
id = 1 ; row=10 ; neutron
AIR1
AIR2
Top

Which events are recorded?

1) At present, our default is to write on file only the events in which the primary particle had an inelastic interaction in the Target Region (~1.2% of all primaries considered)

2) It is possible to perform other choices upon request (*for example, in trigger or efficiency studies we write all events*)

Warning:
Asking for an inelastic interaction in the Target does not mean that it is the only inelastic interaction there:

a) in (very) few cases you may have another secondary interacting in the target
b) Primary can perform an elastic interaction before the inelastic one (see later)

What does it mean "Initial" or "Final"?

Initial coordinates: the coordinates of the point in the global reference frame where a particle is injected, or generated by interaction or decay

Initial momentum: the 3-vector P components at the point of injection or generation

Final coordinates: the coordinates of the point in the global reference frame where a particle "dies". A particle dies when: 1) has an inelastic interaction; 2) decays; 3) exits from the geometry; 4) its energy goes below the transport threshold which has been set in simulation: it is then propagated to the end of the remaining CSDA range.

Final momentum: the 3-vector P components at the point of death. In case 4) P_{final} components are 0.

About the meaning of time:

In a single event, Time starts from 0 in the point where the primary particle is injected.

TRtime: it has to be **0** for the primary. If the primary travels with velocity β, and interacts after a length **L**, the secondaries will be generated at $t = L/(\beta c)$ and that will be the value inside their TRtime

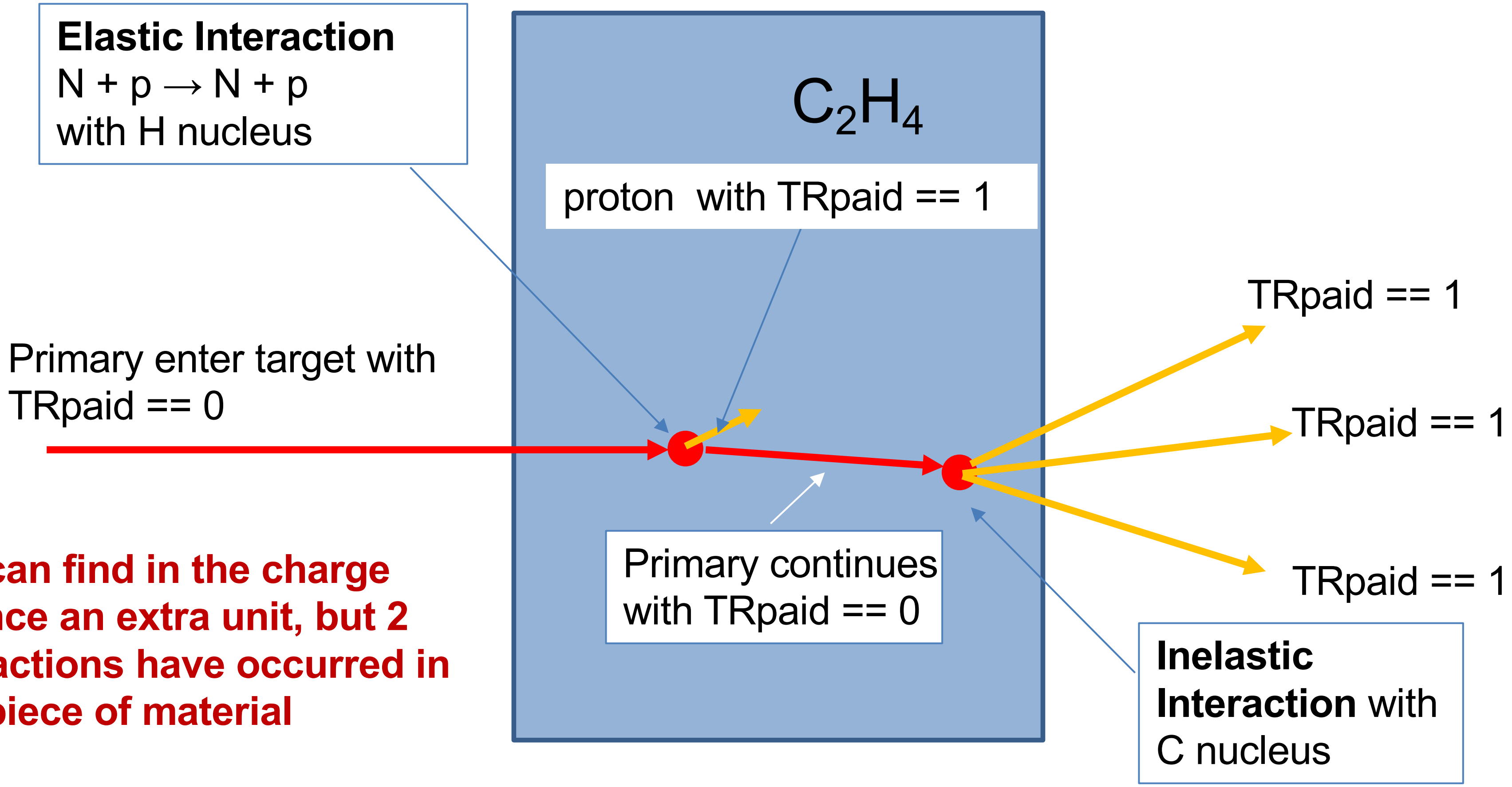
TRtof: it is the time difference between the ”death” (see slide before) and creation of a particle

Data omitted in the event recording

1. Unfortunately, we never included (so far) Z, A of the target nucleus where interaction occur. At present Target Nucleus can be often reconstructed by checking Z and A conservation: $\sum Z_i$ of secondary particles having id=1 has to be equal to the sum of Z of primary and Z of target. The same for baryonic number conservation.

2. We have not marked in any way elastic scattering. Be careful when interaction occurs in materials where Hydrogen is present: the recoiling proton (H) from elastic scattering of the primary (or of a secondary fragment) may appear from coordinates where no inelastic interaction occurred…

An example of the “elastic interaction problem”



You can find in the charge balance an extra unit, but 2 interactions have occurred in this piece of material

The individual detectors structures

For each detector with n energy releases (hits) the infos are stored in arrays (x, p, ΔE, time, etc...) with the i-th component related to the i-th release .

*DET*n = number of hits (energy releases) in the detector *DET*

*DET*id = to build the pointer to the particle responsible of the hit

*DET*xin, *DET*yin, *DET*zin = initial position of hit

*DET*xout, *DET*yout, *DET*zout = final position of hit

*DET*pxin, *DET*pyin, *DET*pzin = initial momentum of hit

*DET*pxout, *DET*pyout, *DET*pzout = final momentum

*DET*de = energy release ← *Remember it is in GeV*

*DET*tim = initial time of the energy release

There can be specific variables depending on the type of *DET* when needed.
For example: Layer, View, ….

Start Counter: STC

```
Int_t    STCn;
Int_t    STCid[MAXSTC];
Double_t STCxin[MAXSTC];
Double_t STCyin[MAXSTC];
Double_t STCzin[MAXSTC];
Double_t STCxout[MAXSTC];
Double_t STCyout[MAXSTC];
Double_t STCzout[MAXSTC];
Double_t STCpxin[MAXSTC];
Double_t STCpyin[MAXSTC];
Double_t STCpzin[MAXSTC];
Double_t STCpxout[MAXSTC];
Double_t STCpyout[MAXSTC];
Double_t STCpzout[MAXSTC];
Double_t STCde[MAXSTC];
Double_t STCal[MAXSTC];
Double_t STCtim[MAXSTC];
```

Simple case of non-segmented detector

Beam Monitor: BMN

```
Int_t     BMNn;
Int_t     BMNid[MAXBMN];
Int_t     BMNilay[MAXBMN];
Int_t     BMNicell[MAXBMN];
Int_t     BMNiview[MAXBMN];
Double_t BMNxin[MAXBMN];
Double_t BMNyin[MAXBMN];
Double_t BMNzin[MAXBMN];
Double_t BMNxout[MAXBMN];
Double_t BMNyout[MAXBMN];
Double_t BMNzout[MAXBMN];
Double_t BMNpxin[MAXBMN];
Double_t BMNpyin[MAXBMN];
Double_t BMNpzin[MAXBMN];
Double_t BMNpxout[MAXBMN];
Double_t BMNpyout[MAXBMN];
Double_t BMNpzout[MAXBMN];
Double_t BMNde[MAXBMN];
Double_t BMNal[MAXBMN];
Double_t BMNtim[MAXBMN];
```

Layer

Drift Cell (in a given layer)

View (X or Y in a given layer)

This is a segmented detector
Additional specific variables are needed

Vertex Tracker: VTX

```
Int_t    VTXn;
Int_t    VTXid[MAXVTX];
Int_t    VTXilay[MAXVTX];
Double_t VTXxin[MAXVTX];
Double_t VTXyin[MAXVTX];
Double_t VTXzin[MAXVTX];
Double_t VTXxout[MAXVTX];
Double_t VTXyout[MAXVTX];
Double_t VTXzout[MAXVTX];
Double_t VTXpxin[MAXVTX];
Double_t VTXpyin[MAXVTX];
Double_t VTXpzin[MAXVTX];
Double_t VTXpxout[MAXVTX];
Double_t VTXpyout[MAXVTX];
Double_t VTXpzout[MAXVTX];
Double_t VTXde[MAXVTX];
Double_t VTXal[MAXVTX];
Double_t VTXtim[MAXVTX];
```

Layer (0,1,2,3)

Identify the pixel

This is a segmented (=pixelated) detector
One additional specific variable is enough

Inner Tracker: ITR

```
Int_t    ITRn;
Int_t    ITRid[MAXITR];
Int_t    ITRsens[MAXITR];
Double_t ITRxin[MAXITR];
Double_t ITRyin[MAXITR];
Double_t ITRzin[MAXITR];
Double_t ITRxout[MAXITR];
Double_t ITRyout[MAXITR];
Double_t ITRzout[MAXITR];
Double_t ITRpxin[MAXITR];
Double_t ITRpyin[MAXITR];
Double_t ITRpzin[MAXITR];
Double_t ITRpxout[MAXITR];
Double_t ITRpyout[MAXITR];
Double_t ITRpzout[MAXITR];
Double_t ITRde[MAXITR];
Double_t ITRal[MAXITR];
Double_t ITRtim[MAXITR];
```

Mimosa28 Chip (0 – 31)

Identify
the pixel

Another pixelated detector with specific features

MicroStrip Detector: MSD

```
Int_t    MSDn;
Int_t    MSDid[MAXMSD];
Int_t    MSDilay[MAXMSD];
Double_t MSDxin[MAXMSD];
Double_t MSDyin[MAXMSD];
Double_t MSDzin[MAXMSD];
Double_t MSDxout[MAXMSD];
Double_t MSDyout[MAXMSD];
Double_t MSDzout[MAXMSD];
Double_t MSDpxin[MAXMSD];
Double_t MSDpyin[MAXMSD];
Double_t MSDpzin[MAXMSD];
Double_t MSDpxout[MAXMSD];
Double_t MSDpyout[MAXMSD];
Double_t MSDpzout[MAXMSD];
Double_t MSDde[MAXMSD];
Double_t MSDal[MAXMSD];
Double_t MSDtim[MAXMSD];
```

Layer (0 – 6)

Identify the strip (X or Y)

Another type of segmentation

Tof Wall: SCN

```
Int_t    SCNn;
Int_t    SCNid[MAXSCN];
Int_t    SCNibar[MAXSCN];
Int_t    SCNiview[MAXSCN];
Double_t SCNxin[MAXSCN];
Double_t SCNyin[MAXSCN];
Double_t SCNzin[MAXSCN];
Double_t SCNxout[MAXSCN];
Double_t SCNyout[MAXSCN];
Double_t SCNzout[MAXSCN];
Double_t SCNpxin[MAXSCN];
Double_t SCNpyin[MAXSCN];
Double_t SCNpzin[MAXSCN];
Double_t SCNpxout[MAXSCN];
Double_t SCNpyout[MAXSCN];
Double_t SCNpzout[MAXSCN];
Double_t SCNde[MAXSCN];
Double_t SCNal[MAXSCN];
Double_t SCNtim[MAXSCN];
```

SCN bar (0 – 19)

SCN view (0 – 1)

rear layer (0) **front layer** (1)

Inverted!

Another type of segmentation

Calorimeter: CAL

```
Int_t    CALn;
Int_t    CALid[MAXCAL];
Int_t    CALicry[MAXCAL];
Double_t CALxin[MAXCAL];
Double_t CALyin[MAXCAL];
Double_t CALzin[MAXCAL];
Double_t CALxout[MAXCAL];
Double_t CALyout[MAXCAL];
Double_t CALzout[MAXCAL];
Double_t CALpxin[MAXCAL];
Double_t CALpyin[MAXCAL];
Double_t CALpzin[MAXCAL];
Double_t CALpxout[MAXCAL];
Double_t CALpyout[MAXCAL];
Double_t CALpzout[MAXCAL];
Double_t CALde[MAXCAL];
Double_t CALal[MAXCAL];
Double_t CALtim[MAXCAL];
```

Crystal number

Most simple segmentation

The crossing data structure
Not yet inherited in SHOE

This structure registers the info on the particles that cross the boundaries between the different regions of the setup (detector elements, air, target).

CROSSn = number of boundary crossing
CROSSid = position of the crossing particle in the particle block
CROSSnreg = no. of region in which the particle is entering
CROSSnregold = no. of region the particle is leaving
CROSSpx, CROSSpy, CROSSpz = mom. at the boundary crossing
CROSSx, CROSSy, CROSSz = position of the boundary crossing
CROSSt = time of the boundary crossi
CROSSch = charge of crossing particle
CROSSm = mass of the crossing particle

```
Int_t CROSSn;
Int_t CROSSid[MAXCROSS];
Int_t CROSSnreg[MAXCROSS];
Int_t CROSSnregold[MAXCROSS];
Double_t CROSSx[MAXCROSS];
Double_t CROSSy[MAXCROSS];
Double_t CROSSzMAXCROSS];
Double_t CROSSpx[MAXCROSS];
Double_t CROSSpy[MAXCROSS];
Double_t CROSSpz[MAXCROSS];
Double_t CROSSm[MAXCROSS];
Double_t CROSSch[MAXCROSS];
Double_t CROSSt[MAXCROSS];
```

Very useful for many analyses about MC truth

Retrieving MC HITS from Detector Structures in SHOE

When processing a rootple obtained after processing with DecodeMC, you can use in your macro the methods defined in shoe/libs/src/TAMCbase (TAMCntuHit.hxx, TAMCntuHit.cxx)

GetHitsN() returns the no. of hits for the selected detector in the event

Int_t	**GetID()** →	returns DETid
Int_t	**GetTrackIdx()** →	returns pointer to the track that generated the hit
Int_t	**GetSensorId()** →	returns sensor no. when relevant (e.g. ITR)
Int_t	**GetBarId()** →	returns bar no. (SCN)
Int_t	**GetCrystaId()** →	returns CALicry
Int_t	**GetLayer()** →	returns layer no. (meaning changes with detector}
Int_t	**GetView()** →	returns SCNview
Int_t	**GetCell()** →	returns BMN cell
TVector3	**GetInPosition()** →	returns DETxin, DETyin, DETzin
TVector3	**GetOutPosition()** →	returns DETxout, DETyout, DETzout
TVector3	**GetInMomentum()** →	returns DETpxin, DETpyin, DETpzin
TVector3	**GetOutMomentum()** →	returns DETpxout, DETpyout, DETpzout
Double_t	**GetDeltaE()** →	returns DETde **(for TW is already converted in MeV !)**
Double_t	**GetTof()** →	returns DETtim

(See later also slide #45)

About the energy release in simulation - 1

<u>Charged particles</u>

Suppose for simplicity to neglect δ-rays:

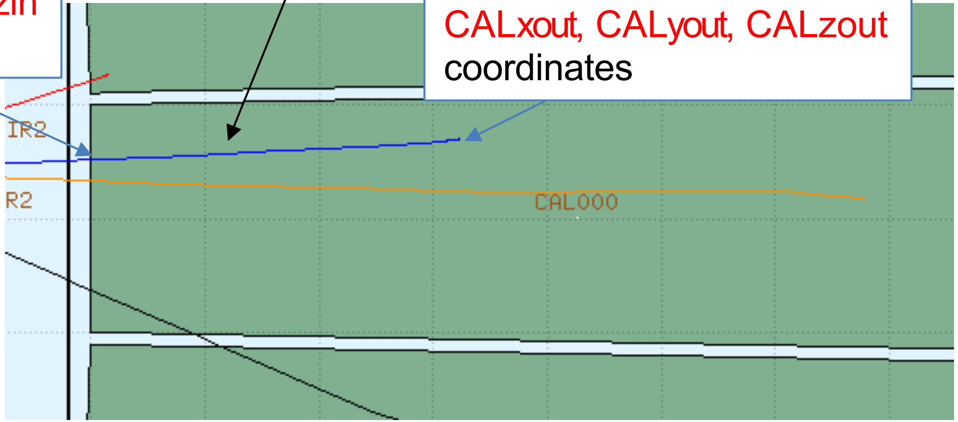
A ”hit” will be the energy lost during a “step” (with fluctuations of dE/dx properly considered in a continuous way)

Example for DET=CAL

<u>Assume this is a single step of a particle in a crystal</u>

CALxin, CALyin, CALzin coordinates

CALxout, CALyout, CALzout coordinates



CALpxin, CALpyin, CALpzin are the momentum components of the particle releasing energy had at CALxin, CALyin, CALzin

Analogous for CALpxout, CALpyout, CALpzout

No electronics/detector effects → no experimental resolution

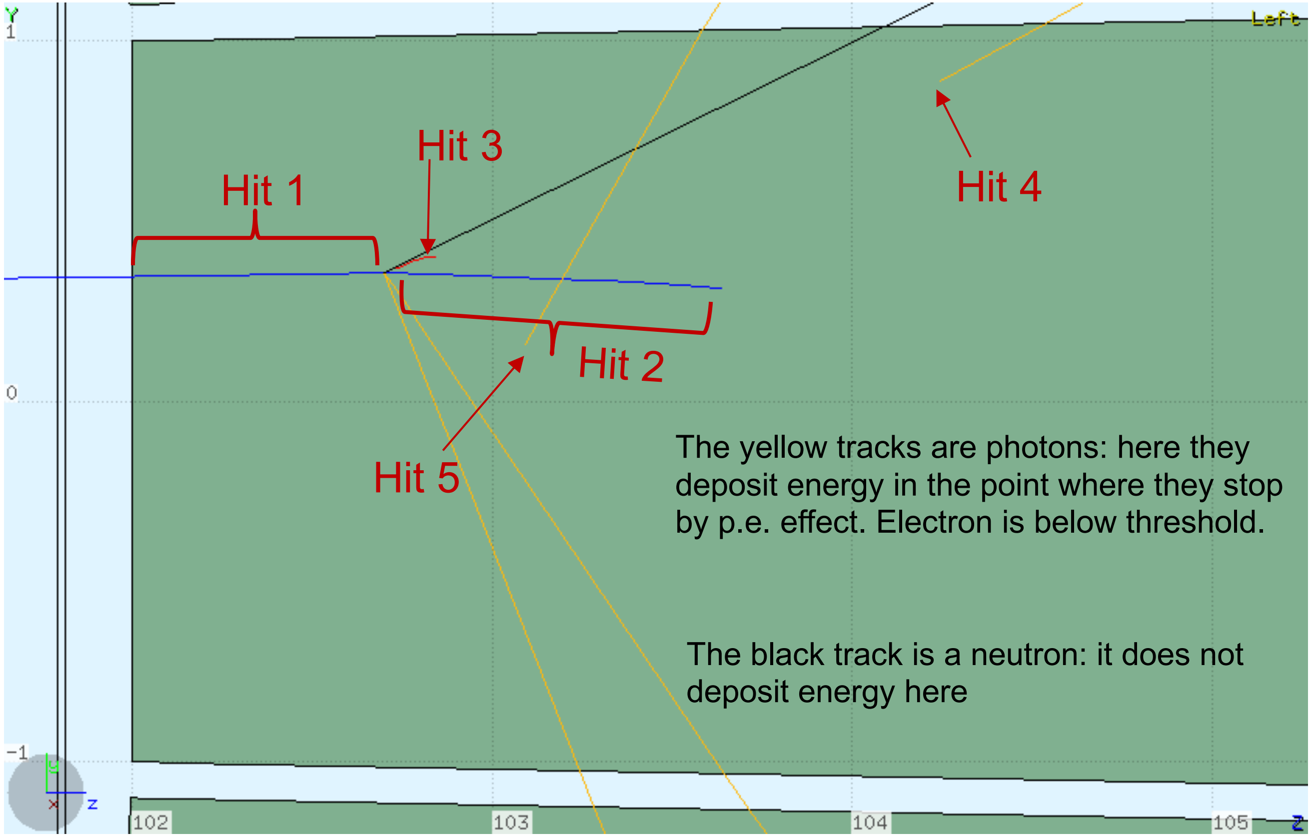
No quenching factors introduced so far

These have to be introduced in your post-processing macros

<u>Only physics intrisic fluctuations</u> (i.e. ”*Landau*” fluct.)

About the energy release in simulation - 2

More complex cases



Of course:

1) the energy released per event in the same detector is the **sum of ΔE** (here CALde)
2) The energy released per event in a single element of the detector is obtained by restricting the sum to a selected element. In this case you could use CALicry to select a given crystal

About the energy release in simulation - 3

There are cases in which the Hits (Energy depositions) have point-like space dimension.

In Fluka this may occurr in some cases. For example:

a) for e^+/e^-/photons which go below transport energy threshold *(see Hit #4 and Hit #5 in the previous slide)*
b) "Low Energy" neutrons (E<20 MeV) which deposit energy by kerma factors

Connecting Hits in Detectors to Track Structure

To find which particle released energy in a detector we can build a pointer to the Particle structure. Given the j-th energy release in the detector DET, then we build:

```
pointer= pevstr->DETid[j]–1;
```

Then the features of the particles responsible of the release (for example the mass and the charge) can be retrieved from the Particle structure as in the following examples:

```
Double_t Mass     = pevstr->TRmass[pointer];
Int_t    Charge = pevstr->TRcha[pointer];
```

To get the pointer in SHOE you make use of GetTrackIdx()

Connecting Hits in Detectors to Track Structure: example in a SHOE macro

thanks to Y.D.

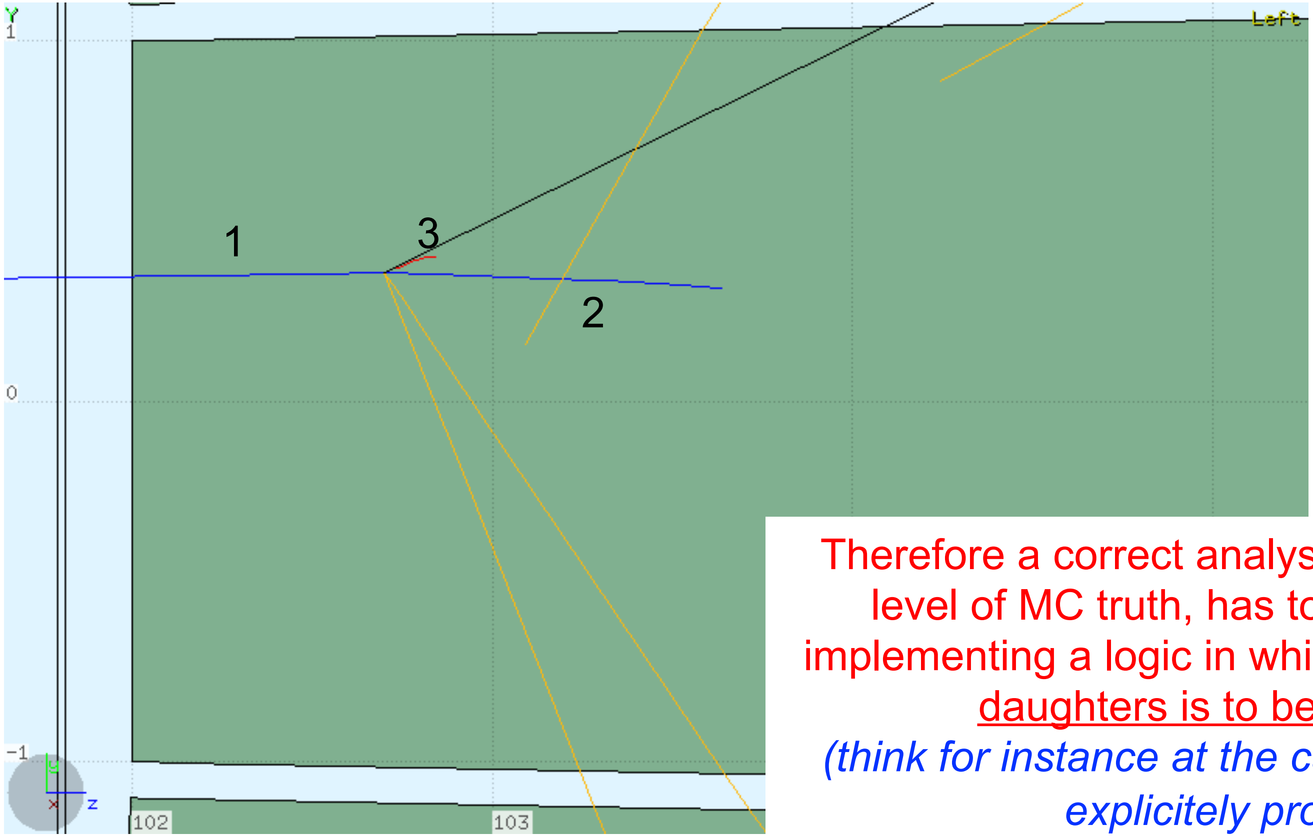
TAMCntuEve *mcNtuEve;
mcNtuEve = new TAMCntuEve(); gets the Event Structure

TAMCntuHit *bmNtuEve
bmNtuEve = new TAMCntuHit();
tree->SetBranchAddress(TAMCntuHit::GetBmBranchName(), &bmNtuEve); gets the Hits of BM
….
Somewhere inside a Loop on the events:
….
Int_t nbmMCHits=bmNtuEve->GetHitsN(); gets the number of Hits in the event

for(Int_t i=0;i<nbmMCHits;i++){ loop on the number of Hits
TAMChit* bmMChit=bmNtuEve->GetHit(i); gets the Hit
TAMCeveTrack* mctrack=mcNtuEve->GetTrack(bmMChit->GetTrackIdx()-1); gets the pointer to the track (particle) which generated the Hit
Int_t charge **mctrack->GetCharge();** retrieves the charge of the particle
….
etc. etc.

On the question of associating Hits with Particles

The issus is not so simple.
In this example the incoming particle releases energy with 3 Hits



Only one of them is directly associated to this particle (no. 1)

The other 2 Hits are associated to daughters of the incoming particle (products of an interaction)

Therefore a correct analysis of this kind, at the level of MC truth, has to be performed by implementing a logic in which the whole chain of daughters is to be considered

(think for instance at the case when δ-rays are explicitely produced)

Possible Basic Exercises using SHOE

1. Make a plot of the multiplicity per event of tracks produced anywhere in the detector
2. Make a plot of the multiplicity per event of tracks produced by the primary in the target
3. Make the previous plot only for those particle which exit the target going in the forward region and are produced with E>50 MeV/u
4. Make a plot of the energy distribution of fragments produced in target for a few different Z and/or A
5. Make a plot of the energy released per event in the TW
6. Make a plot of the energy released per event in the CA and for a selected crystal of your choice

Slightly Increasing Difficulty:

7. Compare the distribution of energy released by p and ^{4}He in the 1^{st} layer of MSD *(in the approximation that they do not produce daughters there)*

8. Select particles produced in the target which arrive at TW and make a plot of the energy that they have lost in the path from target to TW