Technical description of the implementation of the IR7 section at LHC with the FLUKA transport code

Magistris M., Santana Leitner M, Vlachoudis V., Ferrari A., Tsoulou K.

November 15, 2006

Abstract

This document has been written as a handbook to analyze, understand or modify the heat deposition Monte Carlo calculations performed for a wide variety of objects in the IR7 section of the LHC accelerator, in construction at CERN. The work includes references to the prototyping schemes and the implementation of a complex set-up for FLUKA, which deals with lists of objects and properties defined in the Twiss parameters through the use of the LATTICE concept and of a broad collection of user written subroutines.

Contents

1 Introduction

1.1 Simulations of the IR7 insertion

The collimation system of the future Large Hadron Collider (LHC) at CERN is a challenging project since the transverse intensities of the LHC beams are three orders of magnitude greater than those of other current facilities. Two insertions (IR3, IR7) of LHC are dedicated to beam cleaning with the design goal of absorbing part of the primary beam halo and of the secondary radiation. These insertions will house 54 movable two-sided collimators, and will be among the most radioactive areas of LHC. The collimators should withstand the deposited power, which for phase I can reach values of about 25 kW in the upstream units ($\sim$3 kW in the jaws).

The tertiary halo that escapes the collimation system in IR7 could heat some fragile elements up to unacceptable levels, if no additional absorber were used. In order to assess the energy deposition in sensitive components, extensive simulations were run with the Monte Carlo cascade code FLUKA[1, 2].

1.2 IR7 collimation system

The straight section and the dispersion suppressors (DS) of IR7 were fully implemented with FLUKA up to a high degree of sophistication, including all relevant components and details up to realistic limits[1]. A modular approach in the geometry definition and an extensive use of user-written programs allowed the implementation of all magnets and collimators with high precision, including flanges, steel supports and magnetic fields.

1.3 Normalization of the results

All simulations were scaled the following normalization factors, Fn:

energy deposition in superconducting objects [5], counts $\left[\frac{GeV}{cm^3\ p}\right]$ $\rightarrow$ power density $\left[\frac{W}{cm^3}\right]$:

$$\begin{aligned} Fn_e = 4.0 \cdot 10^{11}\ \left[\frac{p}{s}\right] \cdot 1.6 \cdot 10^{-10}\ \left[\frac{J}{GeV}\right] \cdot 0.9 \\ Fn_e = 57.6\ \left[\frac{W\ p}{GeV}\right] \end{aligned} \tag{1}$$

Where:

[1] This enabled to produce a *Povray* based [3] video walk along the geometry, that can be downloaded at [4].

- $4.0 \cdot 10^{11}$ $\left[\frac{p}{s}\right]$ is the proton loss rate per beam for the 0.1h beam lifetime at top energy and ultimate intensity [6][2,3].
- 0.9 is kept as historic loss normalization factor in order to being able to compare with previous calculated results presented at various meetings.

annual dose on warm elements [7], $\left[\frac{GeV}{cm^3\ p}\right] \rightarrow$ dose $\left[\frac{MGy}{y}\right]$

$$Fn_d = 2.0 \cdot 10^{16} \left[\frac{p}{y}\right] \cdot \rho^{-1} \left[\frac{cm^3}{Kg}\right] \cdot 1.6 \cdot 10^{-13} \left[\frac{MJ}{GeV}\right]$$
$$Fn_d = \frac{3200}{\rho \left[\frac{Kg}{cm^3}\right]} \cdot \left[\frac{MJ\ p}{GeV\ y}\right] \quad (2)$$

Where:

- $2.0 \cdot 10^{16}$ $\left[\frac{p}{y}\right]$ is the yearly proton loss rate per beam at ultimate intensity [6].

The former factors indeed correspond to ultimate intensity, 40 % above the level established for nominal intensity. This shall confere a certain security factor for all computations, which are originally expressed in $\left[\frac{GeV}{cm^3\ p}\right]$.

2 Implementation of the FLUKA geometry

2.1 Choice of the input geometry format

One of the very first things that must be agreed on when implementing a FLUKA geometry, is the choice of the format. By default, FLUKA format is fixed, which means that all parameters have to be specified with a fixed numerical format. Regions and bodies are referred to on the basis of their position in the input list. Due to the high complexity of the IR7 geometry, the fixed format was not considered an appropriate choice. Two alternatives were evaluated: ALIFE and free format.

2.1.1 ALIFE FLUKA GUI editor

ALIFE is a GUI editor for the FLUKA geometry and detector definitions. The user can define geometrical bodies by selecting the FLUKA body type from a menu and entering the parameters in a corresponding entry panel. The program automatically

[2]For injection the losses are higher: $8.6 \cdot 10^{11}$ $\left[\frac{p}{s}\right]$

[3]These numbers has been refined since the completion of our computations.

converts the definitions into FLUKA fixed format and allows the editing of already existing FLUKA body lists. FLUKA region definitions and assignments are written in an intermediate format. Instead of referring to the bodies by their position in the body list they are named by character strings. This has the advantage that the objects can be put in any order, and new objects can be inserted at any place. The editor takes care of the right numbering in the FLUKA region definitions. ALIFE has been extensively used in previous FLUKA studies and was deemed as a valid option for IR7. However, the recent FLUKA free format offers the same advantages of ALIFE without the need of relying on an external editor.

2.1.2 The FLUKA free format

Free format has been introduced only recently, although it is expected to soon supersede the fixed format. Free format has the great advantage of making the geometry definition flexible. For example, it enables the user to add or remove a body without affecting the whole region definitions. Moreover, bodies and regions are assigned a name instead of a sequential number, which makes it easier for the user to associate the FLUKA region to the real object. Furthermore, a well-chosen name greatly simplifies the geometry debugging and development. If there are two elements that are very similar (e.g., a horizontal and a vertical MQW), the implementation of the second element can be done by copying the first element, modifying one character in all the body and region names to differentiate them from those of the first element and making the necessary modification in the body definition. With the fixed format, the body and region names (numbers) would need to be changed completely.

It was decided that the whole geometry was to be implemented with the free format. For a matter of consistency, all the existing FLUKA models of LHC elements were to be translated into free format. The geometry of the tunnel and of the warm quadrupole MQW were provided by [8]. These elements had been implemented with ALIFE and needed 90° rotation to be consistent with our reference system ($\hat{x}$ horizontal, $\hat{y}$ vertical and $\hat{z}$ parallel to the beam). A specific FORTRAN program was written to read the ALIFE files, rotate all bodies of 90° and provide the geometry input in free format, preserving the original region names. If this operation were to be repeated, it is the opinion of the authors that script programming languages like PERL, REXX and PYTHON would be much more appropriate than FORTRAN.

2.2 A modular approach

The IR7 section of LHC consists of more than 200 elements, including collimators, dipoles, quadrupoles, absorbers and multipole magnets. Most of these elements appear several times in the IR7 layout; for example, there are four couples of MBW in the straight section. Sometimes all elements of the same type are completely identical (e.g., the MB) although commonly they show some differences, like the orientation (e.g., collimators), material (e.g., active absorbers and collimators) and beam pipes (e.g., MBW). In this paper the name "prototype" will refer to an element which is representative of all other similar elements in IR7, which will be named "replica". Despite some slight differences between a prototype and its replica, a modular approach [9] was adopted. Prototypes were defined with full details and replicated along the tunnel with minor changes, if need be.

There are at least two ways to implement a FLUKA geometry with a modular approach. The technically easier one, which only applies to straight sections, is to place the prototype inside the tunnel. The prototype will be defined only by infinite bodies (in the beam direction) and an upstream and a downstream cutting-plate, perpendicular to the beam. All replica of this element can be obtained by simply using the same body and region definition and modifying the position of the two cutting planes. However, this method was not chosen due to its intrinsic limitations. For example, it does not allow any rotation nor deviation from the beam axis, as it is needed for implementing the collimators.

The second way, the one chosen, is to place the prototypes in a "parking" near the simulating geometry and replicate them along the tunnel by means of *Lattice*. Whenever a particle enters a replica in the tunnel (which is implemented as an empty box as big as the prototype to be replicated), it is virtually transported to the prototype in the parking. The change of reference system, including translation and rotation, is defined by the user in a routine. As soon as the particle escapes from the prototype, it is transported back to the exit of the replica in the tunnel. Although particles entering any of the replicas are actually transported inside the same prototype, it is still possible to distinguish at scoring time between events occurring in two different replica.

In principle, any analytical symmetry transformation can be implemented via Lattice (rotation, translation, reflection or combination of these). Materials, thresholds and other settings are assigned to the prototype. Therefore all replica of the same prototype share the same properties. For example, because TCS and TCLA have a different jaw material, they need two separate prototypes which only differ in material assignment. However, both for the primary and secondary collimators, it would be pointless to make a dedicated prototype for each jaw halfwidth. It was thus decided to move the plane cutting each jaw at runtime. Whenever a particle

enters a TCS, a TCP or a TCLA, the half-gap is automatically corrected depending on the specific replica. The on-line corrections are very practical in cases like this one, but should not be abused of. Indeed, the user must have some expertise to fully understand how these run-time corrections affect the geometry, unless s/he is the one who implemented them.

This techique is limited to 2D, here the objects are placed longitudinally along the z coordinate.

3 Prototypes

The following sections describe the FLUKA geometry of each prototype of the IR7 model. Such description is meant to help the user to distinguish among different regions and to associate the area of interest to the FLUKA name. However, the implementation of the geometry is in continuous development and from the publication of the present report changes may have occurred.

3.1 Primary Collimators. See figs. 1(a), 9(o)

3.1.1 General view

The collimator delimiting virtual box is TCP2. The collimators are contained within a steel tank (TCPbox) that has a circular hole on each extreme (TCPplho) to admit the pipe of beam 1 or of beam 2[4]. The tank is connected to the other elements through an entry and an exit tube (TCP{entr/exit}) adapted by flanges that are composed of two concentric rings, {i/o}, upstreams and downstream {u/d}: TCPfla{u/d}{i/o}

3.1.2 Components inside the vacuum tank

All the vacuum within the entry an exit couplings is defined in TCPV01 and the rest of the vacuum outside the tank is TCPV02. Inside the tank, there are two symmetric pieces (R/L) composed by an steel clamp serrage (TCPclS{R/L}), a plaque support (TCPpla{R/L}) evenly divided into 5 central sections with 4 holes each and 2 extreme sections with 2 holes each (TCPho{1-5}{R/L}), a jaw, a clamp support, 5 x-transverse springs (TCPspr{R/L}), 6 longitudinal water pipes, one on top of the other one (TCPpiJ{R/L}).

[4] An LHC collimator is a single beam object.

3.1.3 The TCP jaws

The jaws TCPjaw{R/L} are elongated blocks with conical tapering, and about 12 cm superficial skimming (TCPisk{R/L}) that leave an active length of the jaw of 60 cm, the interactions taking place in a thin over-layer (TCPsca{R/L}) delimited by the planes TCPP{R/L}1 and TCPP{R/L}S, which are moved at runtime by the configuration set in the file *absorber_summary.dat* by the routine *lattic.f*. The aperture of the jaws introduced in *absorber_summary.dat* is taken from the Twiss files, which provide the x (BETX) and y (BETY) component of the normalized Beta function in terms of zz [cm]. The optical Sigma is given by:

SigmaX = Sqrt(E BETX) **SigmaY** = Sqrt(E BETY)

Where E is the emittivity:

$$E = \frac{E_n}{\gamma\beta} \tag{3}$$

At 7 TeV $E_n = 3.75E - 6m$ (normalized emittivity), $\gamma = 7461$ and $\beta \sim 1$. The file [IR7]/[Twiss]/tensigma.txt gives SigmaX and SigmaY already multiplied by a factor 10. Values are written in microns for Sigma and cm for position of the objects.

The transition from 7 TeV (lwb) to 450 GeV (injection) implies a huge variation in the half-widths of the collimators and absorbers. The geometry of the collimators cannot absorb such a change based exclusively in the moving planes (and leaving everything else untouched) because then the jaws of the collimators at injection would almost be completely "eaten" by the invading moving planes, which would be too open. Thus, the *ir7.fluka* file includes two separate definitions of the collimators, inserted between "define" cards, one for low beta, the other for injection. This is illustrated in 3.2 for the secondary collimators.

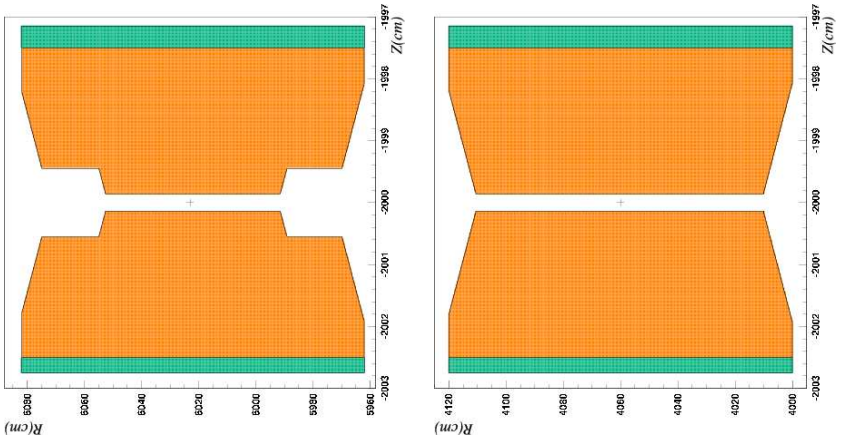
◇ The primary collimators have a half-gap of about 6 Sigma.
◇ There is a definition for top energy and another one for injection.

3.2 Secondary Collimators. **See figs. 2(a), 2(b), 3.2**

The prototype for the secondary collimators is identical to that of the primary ones, with three exceptions:

(i) All prefix TCP are replaced by TCS.

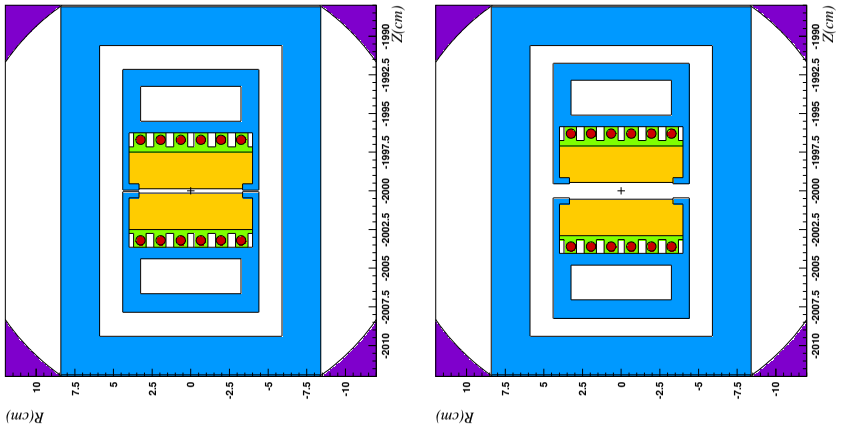
(ii) The first rule does not apply to the interaction planes TCPP{R/L}1 and TCPP{R/L}S, used by both objects.



(a) Shimmed jaws, 60 cm long, in the Primary collimators

(b) 100 cm long jaws in the Secondary collimators

Figure 1: z,x-stretched views of the jaws in the collimators.



(a) Low beta, 7000 GeV

(b) Injection, 450 GeV

Figure 2: x,y cross plots of the TCS at full energy and at injection.

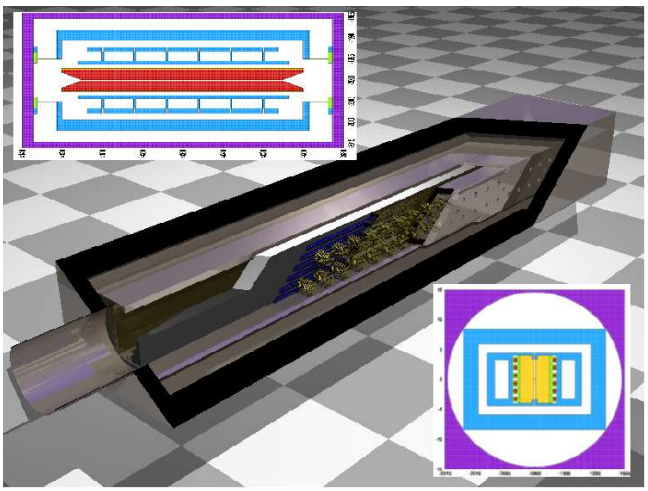


Figure 3: Horizontal and vertical cross sections and full view of the secondary collimator prototype

(iii) The jaws of the TCS don't have shimmed layers because the interaction length is the full length (see figure 3.1.3).

◇ The secondary collimators have a half-gap of 7 Sigma
◇ There is a definition for top energy and another one for injection (see fig. 3.2.)

3.3 Active Absorbers. See figs. 4(a), 4(b)

The active absorbers (tertiary collimators) are almost like the secondary collimators so the same rules are applicable (TCP → TCL). The singularities of the active absorbers are the following:

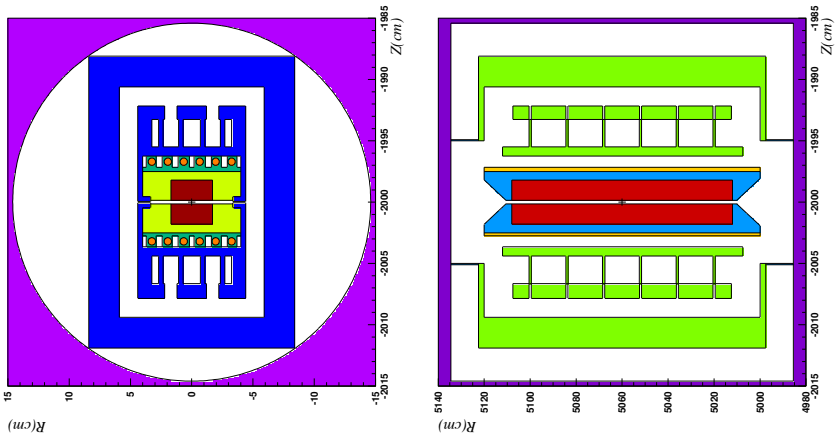
(i) The base material of the jaws is Cu, instead of carbon (collimators).

(ii) The jaws may have W insertion blocks (as shown in fig. 4(a),4(b)). These W inserts are 90 cm long (z), 3.4 cm wide (y) and have a thickness *lower than* 1.8 cm (x)[5]:

```
* Tungsten box for tungsten jaw
RPP TCLW    -2001.8  -1998.2 -1.7  1.7  5012.  5108.
```

◇ The jaws of the active absorbers are sitting at 10 Sigma.

[5]The prototype is centered around x = -2000 cm



(a) xy (front) stretched view of an active absorber

(b) yz (side) stretched view of an active absorber

Figure 4: Stretched frontal and side views of an active absorber with a W insertion block.

3.4 Magnets

Each LHC magnet has its own magnetic field mapping that is defined in various ways, e.g. analytically or with 2D interpolation of a grid. A special file format was created to allow a common and flexible description of each single case, including symmetries and anti-symmetries. The format describes a field by using a 2D linearly interpolated form, an analytical form (for constant fields and perfect quadrupoles) or both. As an example, the field of the quadrupoles is described with an analytical formula inside the vacuum pipes and it is linearly interpolated from a 2D grid outside the pipes. The intensity of each magnetic field as well as the rotation and the position in the arcs are dynamically assigned by a REXX[10] script for each element to ensure the correct optics and beta function of the beam.

3.4.1 MQW. See figs. 3.4.1, 8(d) 9(b)

An MQW is a warm quadrupole with elliptical beam chambers. There are two classes of MQW: MQWA and MQWB, the difference being the intensity of the magnetic field. In addition, there are two classes of prototypes, depending on whether the vacuum chamber of beam 1 has the major axis laying on the vertical axis or in the horizontal one. In the former case, the box containing the prototype is B_QUAD1 and in the file *prototype.pos* it is associated to MQWA+ AND MQWB+ (the '+' indicates that the magnetic field map is used with positive sign).

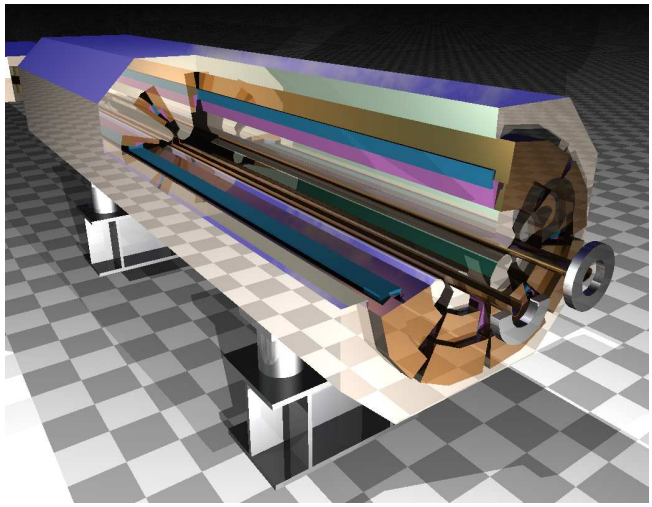


Figure 5: Open PovRay view into an MQW magnet as implemented in IR7 with FLUKA

All regions belonging to this prototype begin with the letter 'M', for example MAIROUT. In the latter case, the box is called B_QUAD2, the relative name in *prototype.pos* is MQWA- or MQWB-. All regions begin with 'N', for example NAIROUT.

The quadrupole yoke is divided into four quadrants:

MULYOKE upper-left **MURYOKE** upper-right

MLLYOKE lower-left **MLRYOKE** lower-right

There are eight pairs of coils, air gaps and insulators, whose name is divided into 4 parts:

1. The first character ('M' or 'N') indicates, as usual, the type of magnet.
2. The following part indicates the location

   UL for upper left **TO** for top **UR** for upper right

   LE for left **RI** for right

   LL for lower left **BO** for bottom **LR** for lower right

3. The third group of letters gives the kind of region:

COIL for coils **INSU** for insulators **AIR** for the air gap.

4. The last character or number is for the specific element: '1' and '2' for the two coils; '1', '2' and 'M' for the insulation; 'IN' and 'OU' for the inner and outer air-gap.

As an example, MBOAIRIN is the inner ('IN') air gap ('AIR') between the coil pair in the bottom ('BO') and belongs to the quadrupole with a vertical chamber for beam 1 ('M').

In the central part of the magnet lies the pole face, which is divided into eight regions: upper or lower ('U' or 'L'), left or right ('L' or 'R') and inner or outer ('I' or 'O'). For example, MLRFACEI is the inner lower right pole face.

Inside the pole faces there are 11 air gaps (from MGAP70 to MGAP80), the left and right vacuum chambers (MLEVC and MRIVC) and the left and right beam vacuum (MLEBV and MRIBV). There are two pairs of flanges: one pair upstream of the magnet ('F', which stays for front) and one pair downstream ('E', for end). Each pair consists of a right ('RI' and '1' for right and beam 1) and a left ('LE' and '2' for left and beam 2) flange. For example, MLE_FL2E is the left (beam 2) flange downstream of the magnet.

3.4.2 MBW. See figs. 3.4.2, 8(b), 9(e)

An MBW is a warm dipole and it is often referred to as 'dogleg bending magnet'. It is located upstream (downstream) of the primary collimators to increase (decrease) the distance between beam 1 and beam 2. There are four pairs of MBW in IR7. The first and the last pairs are contained by the box MBW1_BB and are associated to MBW- in the file *prototype.pos*, where the sign '-' indicates that the magnetic field is opposite to the reference one. The central axis of the vacuum chambers are at 19.4 cm distance between each-other. The remaining pairs of magnets are in the box MBW2_BB and are associated to MBW+. The distance between the beams is in this case 22.4 cm.

The MBW are 380 cm long, 74 cm wide and 104 cm high. Their active length is 340 cm: the front and rear 20 cm thick layers contain the top and bottom coils, that is the connection of the left with the right coil which passes near the beam pipes. The dipole is perfectly symmetric in x and y (apart for the magnetic field, which is antisymmetric in x).

All region names begin with MBW1 for the first and last pairs and MBW2 for the pairs in the middle. The following three or four characters characterize the specific region of the magnet. The steel body is called (MBW1)BODY and the air inside and surrounding the magnet is AIRI (for 'air inside') and AIR, respectively.

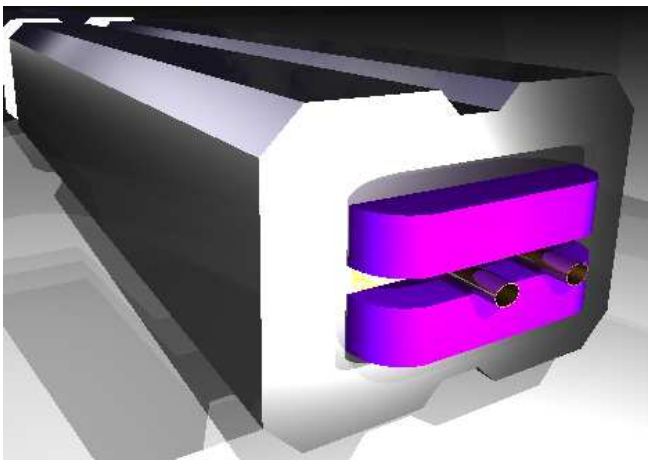


Figure 6: PovRay view of the MBW prototype as implemented in IR7 with FLUKA

Each of the two beam tubes is divided into the part lying inside the yoke (TBM1 and TMB2) and the rest, which exits the central body (TB1 and TB2). These tubes are both elliptical in shape, with an horizontal semi-axis of 2.95 cm and a vertical semi-axis of 2.2 cm. As for the vacuum chambers, the beam vacuum is divided into the regions VC1 and VCM1 (for beam 1) and VC2 and VCM2 (for beam 2).

YYY. The coils are a mixture of copper and water and are defined as one unique region, COIL. The two epoxy layers which separate the top from the bottom coils are represented by the region EPOX.

3.4.3 Cold magnets. See figs. 9(l), 8(c), 9(k), 9(l), 9(m), 9(f), 8(a), 9(n)

All cold magnets (with the exception of the MB and MS, which will be discussed in a separate section) have a similar structure in the FLUKA implementation. All regions begin with Cn, where n is a number from 1 to 7 which identifies the magnet. Table 1 gives the correspondence between the magnet, the FLUKA body and the first two alphanumerical characters identifying the FLUKA regions.

An MQTL (see figs. 9(l), 8(c)) is a superconducting quadrupole and is located along the dispersion suppressor (DS) section. It is defined inside a cylinder (MQTCYLO) with 130 cm length and 30 cm radius. These magnets are found in groups, one next to the other, and share the same cryostat. A group of MQTL is called Q: for example, the first group of cold quadrupoles at the end of the straight section is named Q6.

Some elements of the magnets are replicated for each beam, as it is for the coils that may surround both vacuum chambers. All regions referring to beam

Magnet	FLUKA	Alphanum.
MQTL	MQTCYLO	C1
MCBC	MCBCBODY	C2
MCS	MCS_BODY	C3
MCD	MCDOBODY	C4
MQ	MQ_BODY	C5
MQT	MQT_BODY	C6
MCBH	MCBHBODY	C7

Table 1: correspondence between cold magnets, FLUKA bodies and regions.

1 are called B1, whilst those for beam 2 are called B2. In addition, all vacuum chambers are called VC and all regions with vacuum are called BV (for 'beam vacuum'). For example, the region C6B2BV is the vacuum contained in the pipe for beam 2, in the magnet MQT.

Most of the magnets also contain elements like the cooling system (VH, with liquid helium), the coils (COIL, implemented as one region), the insulator near the coils (INSU, one global region), the air gap between coils (AIRG), iron structures (YOKE and IRON) and air surrounding the magnet (AIRO). Some magnets have special regions like a bottom and a front face (BOFA and TOFA), inner and outer yokes (YOKI and YOKO) and specific coils (OCTU for octupole and DECA for decapole). For example, C3C1COIL is the coil around beam 1 in the sextupole MCS. Some massive material like aluminum or steel is often implemented as a layer around the coil or the beam chamber. These layers are simply called LAY and their material assignment varies from one magnet to the other.

3.4.4 MB and MS. See figs. 9(f), 8(a), 9(n)

Only two cold magnets are implemented with a structure which is different from the previous one: the superconducting bending magnet MB and the sextupole MS.

4 An example of geometry development: Inserting a group of BLM's through a lattice

The insertion of an array of BLM is now shown as an example of the discussed methodology. First of all, the BLM prototype, defined as usual by bodies and regions, is inserted in the *.fluka file, the full object being contained within a BLM bounding box (i.e. BLMBODY) that is substracted from the logics of the parking tunnel definition. The BLMBODY is then referred[6] in the *prototypes.pos* file and

[6] with the parking to tunnel translation.

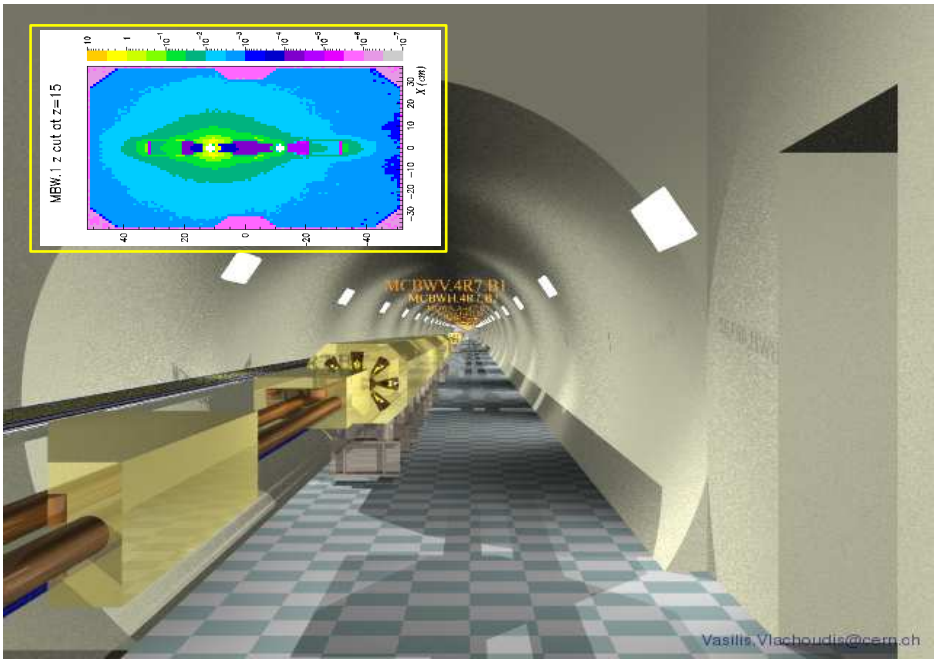


Figure 7: A beam element

associated to a shorter prefix, e.g. BLM (1).

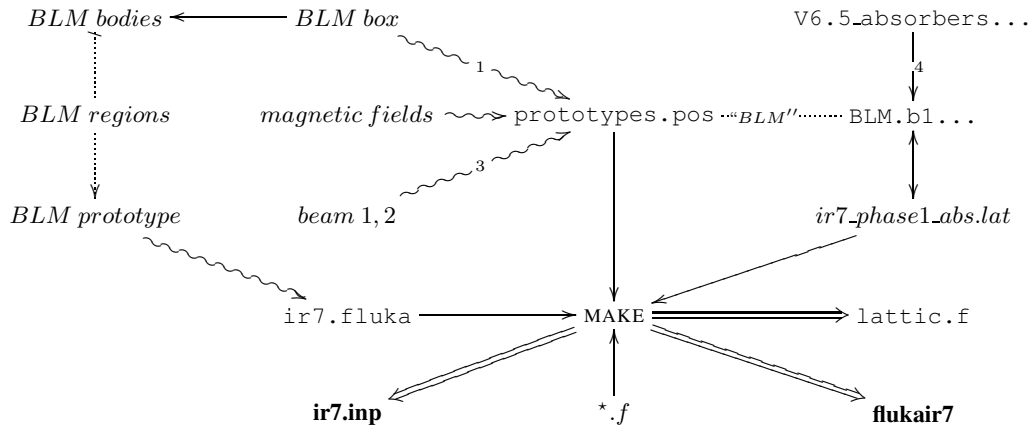
```
# BLMs box is +-30, BLM starts at -25, at 20 cm from FLANGES
BLM  1 0 BLMBODY      -2000.0    0.0  6600.0      60.0  -
#
```

A flag describing the magnetic length is included and it is specified whether the object concerns beam 1, beam 2 or both beams (2). Then, a Twiss file like *V6.5_absorbers.b1.phase1.data* can be used as template, and modified in the following way (4):

- There are as many BLM's as primary (TCP) and secondary (TCS) collimators, so only these objects and the end markers are left in the file.
- The prefix TCP and TCS of each name is replaced by BLM
- The position is shifted by 1.13 m, this being the distance between the center of the BLM's with respect to the preceding collimators.
- The file is saved under the name *BLM.b1.phase1.data*

Finally, the new file *BLM.b1.phase1.data* needs to be linked in the *ir7_phase1_abs.lat* file and then, everything is ready to produce the FLUKA input file, **.inp*, and the

executable, *flukair7*, from the FORTRAN programs, $\star.f$. The lattice patterns (*lattic.f*) are automatically written from the Twiss files and can then be used during execution together with *flukair7*.



5 Definition of the magnetic field

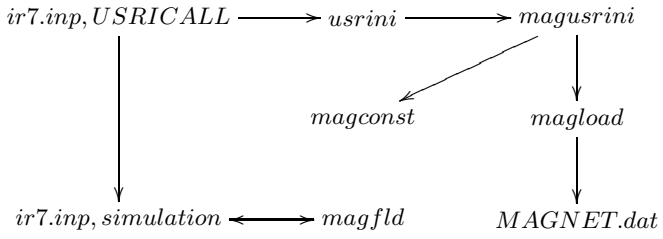
An accurate treatment of the magnetic field is essential in tracking particles along the tunnel, since it has a major impact in the inelastic interaction distribution. In the FLUKA model, depending on the magnet the field is constant, analytical or a 2D interpolation from a grid of values. In the case of an interpolation, a grid is provided with values ranging from -1 to 1. These values are then multiplied by a constant which represents the field intensity inside the beam pipes. By keeping the map unaltered and by fine-tuning the constant values, the beam was kept into the right trajectory along the entire section, within a few micron error.

5.1 Sequence of called routines

All the information about the magnetic field of a family of magnets (e.g., MBW) is written in a *.dat* file (e.g., *MBW.dat*) in the [MagneticField] directory. These files contain a grid (for a 2D interpolation) and indications about symmetries in the field. As an example, if the field is antisymmetric with respect to the $\hat{x}$ axis, the grid is then provided for the area $y > 0$, and an automated routine calculates the remaining semi-plane $y < 0$. These grids are read by FLUKA during initialization. One pair of cards USRICALL per magnet call the routine *usrini*, which in turn calls the routine *magusrini* and loads the field information. The routine *magusrini*, to

be precise, calls the routine *magload* for reading the grid of a 2D interpolated field from the relative *.dat* file, or the routine *magconst* to assign a constant value all over the magnet.

The initialization is done at the beginning of the simulation for all magnets. During the simulation, whenever a particle is inside a region with a magnetic field, *magfld* is invoked and a field vector[7] (calculated from values loaded as above) is returned as a function of the position of the particle and the magnet. The *.dat* files are written in a specific format that is compatible with the *magload*. Further information on how to implement a new magnetic field is included in any *.dat* file.



5.2 Magnetic Fields and Gradients

The fields in the magnets expressed in terms of the K multipole constants:

K0L Field. It corresponds to the angular kick in radian. For example, the 1232 MB.XXXX produce a total kick of 2 π rad, so for every dipole magnet $KOL = \frac{2\pi}{1232} = 5.1\ mrad$. The magnetic induction used to produce this kick is $B = 8.3\ T$

K1L Gradients of the field in the Quadru/Sextu/Decapoles. Some useful relations:

$$K1L = \frac{\int_0^L \frac{dB}{dl} \cdot dl}{Br} \tag{4}$$

Where $Br = B \cdot r$ is the *magnetic rigidity* of the beam [Tm]

$$Br = \frac{p}{e} = 7000 \cdot 10^9/c \tag{5}$$

The *magnetic induction* B in the dipoles is:

$$B = \frac{K0L \cdot Br}{L} = 8.3T \tag{6}$$

[7] Intensity and direction.

Another useful relation is:

$$B = \frac{K1L \cdot Br}{100 \cdot L} \tag{7}$$

Positive K1L value means focusing (defocusing) field on the horizontal (vertical) axis for *Beam1*. The opposite is true for *Beam2*.

5.3 Magnetic Fields Maps

The magnetic field maps are transformed to the BFLUKA format with the *mkfieldmap.r* script. i.e.:

5.3.1 MBs → MB → MB.dat

1. First the field anomalies are corrected with *mkfieldmap.r*. The program prints all the positions where the value is at least a 50 % off the average value of its neighbours. Then manually the i,j position is manually corrected with the average value.

2. Next we search for the constant part of the field which should be enclosed in the region $19 \leq i \leq 24$ and $j \leq 4$ with a small `rexx` script:

```
  rexx - | sum.r -avg 7
  "cat MB.corr.matrf (stack"
  do queued()
  parse pull i j x y bx by b
if i>=19 & i<=24 & j<=4 then say b
  end
  <Ctrl-D>
```

3. Finally we run *mkfieldmap* with the correct scaling factor:

```
mkfieldmap.r MB.corr.matrf MB.dat 1/<avg>
```

Cross sectional plots with a representation of the fields on several magnetic objects can be found in fig. 5.3.2. These fields are discussed below:

5.3.2 MQTL → MQ.dat

First we have to find the field gradient, by using only the value with $y = 0$:

```
grep "^ *31 *[ 0-9][0-9]" MQTL.matrf  > spam.dat
paw
v/read i,j,x,y,bx,by,b spam.dat
sigma y=y/10.0 | convert to cm
v/fit y(31:40) b(31:40) ! p1 ! 10
```

The found value is taken as scaling factor for the field:

```
mkfieldmap.r MQTL.matrf MQTL.dat 1/<P2>
```

6 Beam Generation

Each of the simulated showers was initiated in FLUKA by a nuclear interaction of a proton with the coordinates/direction provided by the COLLTRACK V5.4[8] code [11].

6.1 Defining the beam loss source

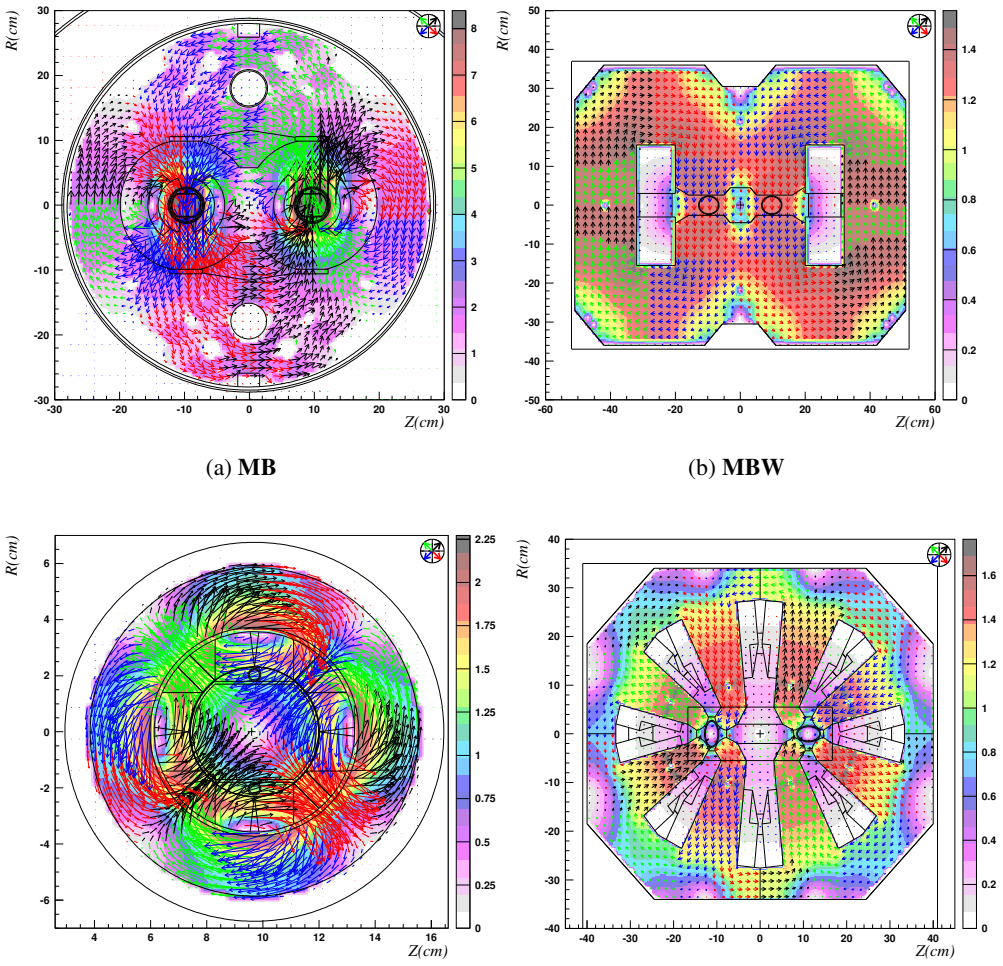
First of all, a scheme had been implemented for a set of interacting coordinates (positions and momenta) stored in files typically called $FLUKA.\{^{lwb}_{inj}\}\_\{^{hori}_{vert}\}\_\{^{nom}_{acc}\}.dat$[9] and for the corresponding description of the collimators (half-widths) stored in files named as $FLUKA.\_summary.\{^{lwb}_{inj}\}\_\{^{hori}_{vert}\}\_\{^{nom}_{acc}\}.dat$.

The transition from COLLTRACK to SIXTRACK implied a complete change in the input files, in format, number of columns, ordering, units and precision. In order to avoid further complications deep down into the scripting trees, after thorough and lengthy checks, a program, *transform.e*, has been created in order to translate the new formating into the old one. Then (*crsource.r*) transforms the coordinates into FLUKA-IR7 reference system. The interaction points are then tested with *checkbeam.e* in order to see the proportion of impacts in several collimators and absorbers. Finally the particles are sampled from the beam file by ([src]/*source.f*). The beam generation sequence is the following:

[8] a multi-turn beam optics program that computes a map of protons lost in the collimators by tracking up to 100 turns the 7 TeV beam at low beta settings.

[9] The naming convention that has been established is the following:

lwb 7 TeV	**hori** horizontal losses	**nom** nominal case
inj injection, 450 GeV	**vert** vertical losses	**acc** accident case, TCS off



(a) **MB** (b) **MBW**

(c) **MQTL** (d) **MQW**

Figure 8: Magnetic field lines in the MB (8(a)), MBW (8(b)), MQTL (8(c)) and MQW (8(d)) magnets

1. Produce the beam loss data *summary* file $FLUKA\_summary...dat$ and *interaction* file $FLUKA.\{^{lwb}_{inj}\}\_.\{^{hori}skew_{vert}\}\_\{^{nom}ini_{acc}\}.dat$, and store them in IR7/Twiss
2. Set the link to the corresponding (*lwb*,*inj*,*hori*,*vert*...) collimator *summary* twiss file in [IR7]/ir7_phase1_abs.lat
3. Recompile to produce the *input* file (*.inp) (`make proper; make`)
4. Transform the format of the *interaction* file (`transform.e`)
5. Generate the beam *source* file events for FLUKA (`crsource.r interaction_file summary_file input_file source_file`)
6. Test the beam with *checkbeam.e*

Steps 2 to 6 are handled by the script *makebeam.sh* shown below:

```
#!/bin/bash
echo "energy: injection [inj], nominal [lwb]?"
read en
  e=`echo $en | sed "s/\([a-Z]\)[a-Z]*/\1/"`
echo "loss plane: horizontal [hori], vertical [vert], skew [skew]?"
read pl
  p=`echo $pl | sed "s/\([a-Z]\)[a-Z]*/\1/"`
echo "operation: nominal [nom], no TCS [acc], no TCLA [ini]?"
read op
  o=`echo $op | sed "s/\([a-Z]\)[a-Z]*/\1/"`
#
cp $IR7/Twiss/FLUKA.$en.$pl.$op.dat a
echo using file $IR7/Twiss/FLUKA.$en.$pl.$op.dat
transform.e
cd $IR7 # the collimator Twiss file changes with the loss scenario
  sed -i "s/FLUKA_summary.[a-z]*\.[a-z]*\.[a-z]*\.dat/
    FLUKA_summary.$en.$pl.$op.dat/" ir7_phase1_abs.lat
  sed -i "s/FLUKA_summary.[a-z]*\.[a-z]*\.[a-z]*\.dat/
    FLUKA_summary.$en.$pl.$op.dat/" ir7_phase1.lat
  go.sh
cd -
if [ $# -eq 0 ]
then
  crsource.r b $IR7/Twiss/FLUKA_summary.$en.$pl.$op.dat $IR7/ir7.inp
    $IR7/Twiss/$pl$e$o.dat
  echo created file $IR7/Twiss/$pl$e$o.dat
  cd $IR7/Twiss
  echo "$pl$e$o.dat" | checkbeam.e
  cd -
elif [ $# -eq 1 ]
then
  file=`echo $1 | sed "s/TCSG/TCS/" | sed "s/TCLA/TCL/" |
    sed "s/\([a-Z]*[0-9]\)[a-Z0-9]*/\1/"`
  crsource.r b $IR7/Twiss/FLUKA_summary.$en.$pl.$op.dat $IR7/ir7.inp
    $IR7/Twiss/$file$e$p$o.dat $1
```

```
  echo created file $IR7/Twiss/$file$e$p$o.dat  filtered by object $1
  cd $IR7/Twiss
  echo "$file$e$p$o.dat" | checkbeam.e
  cd -
fi                                                                    17
rm a b
```

The needed input files, which can be downloaded from [12] are:

- $\diamond$ $FLUKA.\{^{lwb}_{inj}\}.\{^{hori}_{vert}\}\{^{nom}_{acc}\}.dat$
- $\diamond$ $FLUKA\_summary.\{^{lwb}_{inj}\}.\{^{hori}_{vert}\}\{^{nom}_{acc}\}.dat$

The testing scheme is standard:

i) `./scripts/swap.sh source.f sourcetest.f`

ii) compile: `make proper; make`

iii) run

iv) count the number of errors (lines in error file *ir7001_zxye.dat*).

If the number of errors is low (smaller that 20 for 10^6 particles) then the beam is OK. You can then restore the *source.f* from the *.bck* copy, recompile and run the simulations.

6.2 Filtering the beam loss on a specific object

It may be useful to filter the beam source to the losses in a specific object. This is the case in cross-talk studies for the calibration of the beam loss monitors [7, 13]. The script *makebeam.sh* accepts an argument for the filtering object. The name that needs to be provided is that of the enclosing box of the object (as spelled in the *lattice* section in *ir7.inp*).

For example, to obtain the losses in TCPB6L1, execute

```
makebeam.sh TCPB6L1
```

This will trigger the following instruction for *crsource*

```
crsource.r b $IR7/Twiss/FLUKA_summary...
  $IR7/ir7.inp $IR7/Twiss/TCPB6 $1
```

6.3 Activating a given beam loss source

The user can store different source files in the [Twiss] directory and choose to use one or another for every simulation.

These are the steps to activate any beam loss file:

1. In the SOURCE card of the input file write in the SDUM the file name without the *.dat extension. e.g. for the *hori.dat* file:

```
...
BEAM            7000.0                                          PROTON
SOURCE                                                          hori
...
```

2. In the IR7 launching scheme, the *file.dat* file (here *hori.dat*) will be only found if this file is added to the repository of data files in the script *run.sh*:

```
...
        SOURCES="Twiss/hori.dat Twiss/vert.dat /Twiss/skew.dat"
...
```

3. The last thing to remember, is to edit the $ir7\_phase1\_abs.lat$ file so that the correct $FLUKA\_summary$ file is pointed at before compiling (*make*) the input file (*ir7.fluka*). If you use *makebeam.sh* this step is automatically done.

7 Key files, scripts and folder structure

This section briefly describes some key files, including some general purpose scripts, and it displays the simulation tree structure of the IR7 directory.

7.1 IR7 system files

7.1.1 Input Files

ir7.fluka (,v - RCS versioning) Initial file containing the complete geometry and prototypes. The files contains tags inside comments:

```
* @XXXX:START@
* @STOP@
```

which define regions that the *mklattic.r* program will touch.

prototypes.pos contain all prototype position, names, bounding boxes and rotation information.

coll.pemf EMF XS file (obsolete).

ir7_xxxx.lat rexx callable script that contains the file definitions to use, including for example the collimator gap.

material.txt temporary file for defining the BIASING importance's.

7.1.2 Set-up scripts

mklattic.r (,v - RCS versioning) The master script takes as input several files defined in the file given as argument (typically ir7.lat) and produces the input file for fluka to run and the necessary *lattic.f* file for the lattice transformations.

makefile creates all the necessary files to run "make help". In principle the command "make" should create all the files needed. In case of doubt, please do a "make proper" before.

crsource.r (see sec.6) creates the source data file in the fluka geometry coordinates from the file provided by R.A.and the collimator gaps and fluka input file. It is found in ./Processing. Syntax:

```
rexx crsource.r <inelast_coord> <coll_summ> <flukainput>
  <fileout> [<collimator-name>]
```

Example:

```
rexx crsource.r 131657_inelastic_coord-B1-V.dat
  131657_coll_summary-B1-V.dat ../ir7.inp beam1.dat
```

mkfieldmap.r creates a field map for FLUKA from the .matrf files. Also reports possible position which need correction.

run.sh The running script:

```
run.sh [-bi] [-w workdir] [-r rundir] [-s randomseed]
   [-N #] [-M #]
```

Creates a temporary directory, copying all necessary files and submit PBS job (run.sh -b -s 5.0). Thinks to check before running:

- SOURCE card for the correct source distribution.
- BEAMPOS for the SDUM to be NEGATIVE in the case of Beam2 simulation

patch.so Intermediate VI-editor patch file to convert the names to numbers.

7.1.3 Source code

SiDa/sidain.f The following 4 cards are used to estimate the damage in the Si in the detector areas RR, UJ.

SiDa/fsidmc.f

SiDa/fsidmn.f

SiDa/fsithn.f Load the Si Damage XS to convert the fluence to 1 MeV damage. CALL SIDAIN. Defined inside the *fluscw.f*

SiDa/fluscw.f Perfom the Si Damage weighting for all USRxxx scoring cards containing the "Dm." keyword.

MagneticFields/magfld.f Magnetic field definitions

src/source.f Load the inelastic scattering positions and sample randomly from this distribution and force the inelastic interaction at the specified position. The Z direction is defined in the BEAMPOS card at SDUM. The beam energy is defined with the BEAM card.

src/fusrbv.f Do the scoring per region/lattice for USRBIN type 8.

7.1.4 Data Files

SiDa/sidan.dat, sidap.dat, sidae.dat

SiDa/sidapi.dat Si-Damage weighting functions for neutrons, protons, electrons and pions.

MagneticFields/MQW.dat, MB.dat, MCBC.dat, MQTL Magnetic field maps.

Twiss/V6.5_centre_lowb.b1.twiss

Twiss/V6.5_centre_lowb.b2.twiss

Twiss/twiss* Twiss files containing the beam elements, position and magnetic field intensity.

Twiss/∗_coll_summary.dat Collimator index, rotation, tilts and half-gaps.

Twiss/∗_inelastic_coord.dat Inelastic coordinate position in the reference frame of the collimator. RA simulated 450k protons out of which 90% = (wc -l Twiss/*_inelastic_coord.dat)/450000 is interacting in the IR7. Use the cr-source.r program to process the file.

Twiss/beam*.dat See sec.6. Beam files used by SOURCE contain 5 columns:

- x,y,z position in fluka geometry (cm).
- u,v direction cosines for vx,vy.

Twiss/Distr_MBW.A6L7.B1.dat Text file with the [x, xp, y, yp] coordinates corresponding to the particle (20000 particles) distribution at the end of D3 (MBW.A6L7.B1), ie at the longitudinal position 19825.4239m w.r.t. IP1:

```
Beam parameters:
   em=3.75e-6
   g(rel)=7460.52319
   E=7000TeV
Units: x,y: micron
      xp,yp: murad
```

Twiss/beamMBW.dat The beam distribution Distr_MBW.A6L7.B1.dat the FLUKA coordinate format. To be used as a SOURCE.

7.1.5 Autogenerated files

The following files are generated when running some scripts.

lattic.f Lattice transformation and dynamic change of the collimator gap.

ir7.inp Fluka input file containing the auto-defined LATTICE cards, bodies and regions, and numbers for all the cards apart the geometry.

ir7.mcnp MCNP conversion of the ir7.inp file used for debugging.

ir7_tmp.inp Intermediate file between ir7.fluka and ir7.inp

7.2 Data analysis scripts

7.2.1 A perl script to analyse energy deposition by region

EnLattice.pl requires the input file *ir7.inp*, and the energy deposition per region[18] to produce a table with the energy deposition in every single object of the lattice and the total energy in every box of the lattice.

```
#!/usr/bin/perl -w

print("Fluka input file (*.inp) is:\n");
$inputfile=<STDIN>;                                                    19
chomp($inputfile);
$inputfile =~ s/\s//g;
print ("The file with results (USRBIN, formatted) is:\n");
$resultfile=<STDIN>;
print ("Your results refer to 1 beam [1] or both beams [2]?\n");      20
       $beamoption=<STDIN>;
       chomp($beamoption);
       $beamoption=1 unless $beamoption==2;
print("Mumble, mumble .... difficult calculation ...\n");
chomp($resultfile);                                                    21
$resultfile =~ s/\s//g;
open (OUT1, ">LatticeWatt");
open (OUT2, ">CumulativeWatt");
open (IN, $resultfile);
open (FLINP, $inputfile);                                              22
print(OUT1 "Fluka input file (*.inp) is: $inputfile\n");
print(OUT2 "Fluka input file (*.inp) is: $inputfile\n");
print(OUT1 "The file with the results (USRBIN, formatted) is: $resultfile\n");
print(OUT2 "The file with the results (USRBIN, formatted) is: $resultfile\n");
if ($beamoption == 2)                                                  23
{print(OUT1 "Beam 1 and Beam 2 simulated\n");
print(OUT2 "Beam 1 and Beam 2 simulated\n");
} else
{print(OUT1 "Only one beam simulated\n\n");
print(OUT2 "Only one beam simulated\n\n");                             24
}
# initialisation
$regmin=$regmax=$latmin=$latmax=0;
@values=();
# creates an array with number -> region name, @regname                25
@regname=listregion();
# creates a hash with region name -> number, %regnumber
$regi=0;
while (defined($regname[$regi]))
{ $regnumber{$regname[$regi]}=$regi;                                   26
  $regi++;
}
# creates an array with number -> lattice name, @latname
@latname=listlattice();
# creates a hash with lattice name -> number, %latnumber               27
```

[18] For IR7 studies unit 39 has been used.

```
$lati=0;
while (defined($latname[$lati]))
{ $latnumber{$latname[$lati]}=$lati;
  $lati++;
}
# prints title and date
$title=<IN>;
$date=<IN>;
print OUT1 "$title$date";
print OUT2 "$title$date";
# looks for energy deposition by region
while ($read=<IN>)
{ chomp ($read);
  @read = split (/\s+/, $read);
 if ($read[1] eq 8 && $read[2]==208)
 {
# calls createlist
# reads values
($regmin,$regmax,$latmin,$latmax,@values)= createlist ();
# read statistics
(@statistics)= createstatistics($regmin,$regmax,$latmin,$latmax);
giveresults ($latmin,$latmax,$regmin,$regmax);
cumulateresults ($latmin,$latmax,$regmin,$regmax);
last;
 } # end if 8 and 208
}  # end of file (while)
close(OUT1);
close(OUT2);
close(IN);
print ("I am finished!\n");

# End of the main program

sub createlist
{
    my($regmin, $regmax, $latmin, $latmax, @listvalues);
($regmin, $regmax, $latmin, $latmax, @listvalues) = @_;
# Index
    my($a);
# Temporary variables
    my($read,@read,$numval,$numlin);
chomp($read=<IN>);
$read =~ s/^ *//;
@read = split (/\s+/, $read);
($regmin,$regmax) = @read[0,1];
chomp($read=<IN>);
$read =~ s/^ *//;
@read = split (/\s+/, $read);
($latmin,$latmax) = @read[0,1];
$numval= ($regmax-$regmin +1)*($latmax-$latmin +1);
$numlin = ($numval-$numval%5)/5;
$numlin++ unless ($numval%5) == 0;
#   skipping the next two lines
<IN>; <IN>;
$a=0;
while (++$a <= $numlin)
```

```
{ chomp($read=<IN>);
  $read =~ s/^ *//;
  @read = split (/\s+/, $read);
  push (@listvalues,@read);
}
return ($regmin,$regmax, $latmin,$latmax,@listvalues);
}

sub listregion
{
    my($readr, @readr, @regnumber);
    do     {$readr=<FLINP>;}
    until ($readr =~ /^END/);
        do     { chomp($readr=<FLINP>);
                 @readr = split (/\s+/, $readr);
                 if ( (defined($readr[2]) && $readr[2] =~ /^[|+-]/ ) &&
                      ($readr[1] =~ /^\d+$/)&& $readr[0] =~ /^\w/ )
                 { push(@regnumber,$readr[0]);
                   }
             }
    until ($readr =~ /^END/);
# @regnumber is such as $regnumber[1] = region 1
    unshift (@regnumber,"whatever");
        return @regnumber;
}


sub listlattice
{
    my($readl, @readlat, @latnames);
  while(defined($readl=<FLINP>))
   {  chomp $readl;
      if ($readl =~ /^LATTICE/)
      {   @readlat = split (/\s+/, $readl);
          push(@latnames,$readlat[1]);
      }
  }
# @latnames is defined as to be $latnames[0]=NOLATTICE
    unshift(@latnames,"NOLATTICE");
        return @latnames;
}

sub giveresults
{
    my($latmin,$latmax,$regmin,$regmax);
($latmin,$latmax,$regmin,$regmax)=@_;
# global: @regname, @latname
    my($lat, $reg,@results,$total,$error,$totalerror);
# printing information for cumulative regions
    for ($lat=$latmin;$lat<=$latmax;$lat++)
{   last unless (defined($latname[$lat]));
    print(OUT1 "\n*****   $latname[$lat]   *******\n\n");
    $total=0;
    $totalerror=0;
    $enne=0;
  for ($reg=$regmin;$reg<=$regmax;$reg++)
```

```
  {@required=("$lat","$reg");
# findvalues give the energy deposition if flag is 0, the error if
# the flag is 1
   $stat=0;
      $results=findvalues ($stat,$latmin,$regmin,$regmax,@required);
   $stat=1;
      $error=findvalues ($stat,$latmin,$regmin,$regmax,@required);
 if ($results > 0)
 {    $results=$results*(4.0E11)*(0.9)*(1.6E-10);  # Watt
      $results=$results*2 if $beamoption == 2;
      printf (OUT1 "$regname[$reg]: $reg    %10.3e +- %3d",$results,$error);
      print (OUT1 "%\n");
  $total= $total+ $results; # Watt
      $totalerror=$totalerror+($error*$results/100.)*($error*$results/100.);
      $enne++;
  } #endif
} # for region
    $totalerror=sqrt($totalerror);
    if($total==0.)
        {$totalerror=0.;
         }
    else
        {$totalerror=$totalerror/$total*100.;
     }
    print(OUT1 "\n Total: $total W    +- $totalerror %\n");
 } # for lattice
}

sub findvalues
{
    my($stat,$latmin,$regmin,$regmax,@lrneeded);
($stat,$latmin,$regmin,$regmax,@lrneeded) = @_;
# @values and @statistics are external
# internal variables
    my ($position,$latpos,$regpos);
# return variable
    my ($listresults);
$latpos=shift(@lrneeded);
$regpos=shift(@lrneeded);
$listresults=34.;
while (defined ($latpos))
{
    $position = ($latpos-$latmin)*($regmax-$regmin+1)+$regpos -1;
    $listresults=$values[$position] if $stat == 0;
    $listresults=$statistics[$position] if $stat == 1;
    $latpos=shift(@lrneeded);
    $regpos=shift(@lrneeded);
}
return ($listresults)
}

sub cumulateresults
{
    my($latmin,$latmax,$regmin,$regmax);
($latmin,$latmax,$regmin,$regmax)=@_;
# global: @regname, @latname
```

```
    my($lat , $reg , $results , $total , $stat , $error);
# printing information for cumulative regions
    for ($reg=$regmin ; $reg<=$regmax ; $reg++)
{   last unless (defined($regname[$reg]));
    print(OUT2 "\n*****___$regname[$reg]___*******\n\n");
    $total=0;
  for ($lat=$latmin ; $lat<=$latmax ; $lat++)
  {@required=("$lat","$reg");
   $stat=0;
      $results=findvalues ($stat , $latmin , $regmin , $regmax , @required);
   $stat=1;
      $error=findvalues ($stat , $latmin , $regmin , $regmax , @required);
 if ($results > 0)
 {    $results=$results*(4.0E11)*(0.9)*(1.6E-10);  # Watt
      $results=$results*2 if $beamoption == 2;
      printf (OUT2 "$latname[$lat]: $lat ____%10.3e __+-%3d", $results , $error);
      print (OUT2 "%\n");
  $total= $total+ $results; # Watt
  } #endif
} # for region
    print(OUT2 "\n Total: $total W\n");
 }
}

sub createstatistics
{
    my($regmin , $regmax , $latmin , $latmax , @statistics);
($regmin , $regmax , $latmin , $latmax , @statistics) = @_;
# temporary variables
my ($read , $a , $numval , $numlin);
while ($read=<IN>)
{last if $read =~ /^STAT/;
}
$numval= ($regmax-$regmin +1)*($latmax-$latmin +1);
$numlin= ($numval-$numval%5)/5;
$numlin++ unless ($numval%5) == 0;
$a=0;
while (++$a <= $numlin)
{ chomp($read=<IN>);
  $read =~ s/^ *//;
  @read = split (/\s+/, $read);
  push (@statistics , @read);
}
$a=0;
foreach(@statistics)
{   $statistics[$a]=$statistics[$a]*100+1;
    $a++;
}
return (@statistics);
}$
```

7.2.2 Other useful scripts

Scripts to produce the energy deposition/dose/track length summary files and corresponding plots, as well as other useful scripts are stored in [scripts]. See the appendix for the description of some of them.

7.3 Simulation results

[TCP60new] Simulations with 60 cm long TCP active absorber jaws.

[hori] Horizontal beam loss, 7 TeV, nominal case.

[WarmSection] Passive absorber optimisations. Surveyed entity: peak dose in coils of MBWB, MBWA and MQWAE5

[PAlimWsWmW] 1 m long W (W) + 20 cm long W (sW)+ 60 cm long W (mW) absorbers

[...]

[Cold_Section]

[BLM]

[TCPB6h] Losses (nom, 7TeV, hori) in TCPB6R7L1

[TCPC6h] Losses (nom, 7TeV, hori) in TCPC6R7L1

[...]

[vert] Same as hori, but only some cases studied.

[WarmSection] Passive absorber optimisations. Vertical losses.

[PAlimWsWmW] ...

[...]

[Cold_Section]

[BLM]

[TCPB6v] Losses (nom, 7TeV, vert) in TCPB6R7L1

[...]

[horiI] Same with horizontal losses (nominal) at **I**njection

[vertI] ...

[AbsSim] Simulations (old files, short TCP) of cold section and MQ6 for the first 4 active absorbers

[NBEAM] like [AbsSim] but with corrected beam and with W active absorbers

[doc] Documents, figures and publications

[figs] Figures

[PAC] Article and poster presented for PAC05 conference

[ACCAP] Article and presentation for ACCAC05 conference

[MC2005] Article and presentation for MC2005 conference

[EPAC] Abstracts for EPAC06 conference

[reports] This report and its source files

[Plotgeom] Some plotgeom files.

8 Launching a job

Before launching a simulation job, the user should define the zone of IR7 where most of the CPU effort is to be spent (this implies to change EMFCUT cards, biasing, eventually define black-hole regions, ...), what will be scored (write the *usrbin*, *usrtrack* ... cards and suppress those of lack of interest), what beam source will be used (hori, vert, vertin, ...), what is the energy of the beam (7000 or 450 GeV) and modify the collimator half-width accordingly, etc.

In order to avoid errors for misconfigured files (e.g. *1)* collimator jaws opened for 7000 GeV beam and beam defined at injection, 450 GeV *2)* source file inexistent, *3)* variance reduction not appropriate for zone of study, ...) a script has been written to automatize all tasks, *go.sh go.sh* asks the user the following questions:

```
echo "energy: injection [inj], nominal [lwb] ?"
read en
  e=`echo $en | sed "s/\([a-Z]\)[a-Z]*/\1/"`
echo "loss plane: horizontal [hori], vertical [vert], skew [skew]?"
read pl
  p=`echo $pl | sed "s/\([a-Z]\)[a-Z]*/\1/"`
echo "operation: nominal [nom], noTCS (accident) [acc], noTCLA [ini]?"
read op
  o=`echo $op | sed "s/\([a-Z]\)[a-Z]*/\1/"`
 echo "scoring: warm part [warm], cold part [cold], BLM scoring [BLM],
   UJ's RR's [sida], radioactive nuclei [resn]?"
read sc $
```

71

72

In the example above the user is performing a simulation for **inj**ection with a **hori**zontal beam loss scenario in **nom**inal conditions[73] for the **warm** section of IR7. The script will take care to activate/deactivate the appropriate "#define" cards in *ir7.fluka*, change the beam energy in *mklattic.r*, add the link to the "beam_file.dat" in emphrun.sh, compile and, finally, make the proper readjustments (BEAM, SOURCE, DOSE CARDS) in *ir7.inp*.

[73] **acc**ident here represents the commisioning case where the TCS are retracted.

Then everything (*ir7.inp, lattic.f, run.sh, *.dat*) will be ready to submit the job to the batch queue. Use for instance the following instruction to submit 10 jobs for the previous example:

```
auto.sh 1 10 TCP60new/horiI/PAlimWsWmW
```

In order to add new cases (e.g. beam 2, phase2,...) in *go.sh*, use *go.sh*:

```
#!/bin/bash

# use to compile, link beam files, set aperture for energy...
# for total beams (vertical, horizontal...) use as:
#          > go.sh
# for beams filtered in a collimator use the name of the object ej:
#          > go.sh TCSGA6L1
############ Preparing Twiss files and beam files
echo "energy: injection [inj], nominal [lwb]?"
read en
  e=`echo $en | sed "s/\([a-Z]\)[a-Z]*/\1/"`
echo "loss plane: horizontal [hori], vertical [vert], skew [skew]?"
read pl
  p=`echo $pl | sed "s/\([a-Z]\)[a-Z]*/\1/"`
echo "operation: nominal [nom], noTCS (accident) [acc], noTCLA [ini]?"
read op
  o=`echo $op | sed "s/\([a-Z]\)[a-Z]*/\1/"`
 echo "scoring: warm part [warm], cold part [cold], BLM scoring [BLM],
   UJ's RR's [sida], radioactive nuclei [resn]?"
read sc
#
#  cp ~/.bash_profile ~/.bash_profile.before
#  if [ $sc = 'resn' ]
#  then
#    cp ~/.bash_profile.flukadev ~/.bash_profile
#  else
#    cp ~/.bash_profile.fluka ~/.bash_profile
#  fi
#  diff --brief ~/.bash_profile ~/.bash_profile.before
#  if [ $? -eq 1 ]
#  then
#    rm ~/.bash_profile.before
#    echo "type: 'source ~/.bash_profile' and relaunch the sript go.sh"
#    exit 0
#  fi
#rm ~/.bash_profile.before
echo '#################### runing with version ....  ############'
echo $FLUPRO
#
ls $IR7/Twiss/FLUKA.$en.$pl.$op.dat $IR7/Twiss/FLUKA_summary.$en.$pl.$op.dat
if [ $? -ne 0 ]
then
  echo The file "$IR7/Twiss/FLUKA.$en.$pl.$op.dat" or/and
    "$IR7/Twiss/FLUKA_summary.$en.$pl.$op.dat" does/do not exist !!
  exit
fi
```

```
echo '#################### Adjusting TCP/TCS/TCL prototypes #####'

if [ $en = 'inj' ]                                                          83
then
  sed -i "s/^\#define LOWBETA/\*\#define LOWBETA/" ir7.fluka
elif [ $en = 'lwb' ]
then
  sed -i "s/^\*\#define LOWBETA/\#define LOWBETA/" ir7.fluka                84
fi

if [ $op = 'acc' ]
then
  sed -i "s/^\*\#define NoTCS/\#define NoTCS/" ir7.fluka                    85
  sed -i "s/^\#define NoTCLA/\*\#define NoTCLA/" ir7.fluka
elif [ $op = 'nom' ]
then
  sed -i "s/^\#define NoTCS/\*\#define NoTCS/" ir7.fluka
  sed -i "s/^\#define NoTCLA/\*\#define NoTCLA/" ir7.fluka                  86
elif [ $op = 'ini' ]
then
  sed -i "s/^\#define NoTCS/\*\#define NoTCS/" ir7.fluka
  sed -i "s/^\*\#define NoTCLA/\#define NoTCLA/" ir7.fluka
fi                                                                          87

echo '#################### Setting up scoring zone ##############'

if [ $sc = 'cold' ]
then                                                                        88
  sed -i "s/^\#define BLM/\*\#define BLM/" ir7.fluka
  sed -i "s/^\*\#define SCRCOLD/\#define SCRCOLD/" ir7.fluka
  sed -i "s/^\#define SCRWARM/\*\#define SCRWARM/" ir7.fluka
  sed -i "s/^\#define WARMSCORE/\*\#define WARMSCORE/" ir7.fluka
  sed -i "s/^\#define KATERINA/\*\#define KATERINA/" ir7.fluka                89
  sed -i "s/^\*\#define TUNBLACK/\#define TUNBLACK/" ir7.fluka
  sed -i "s/^\*\#define LOWIMPRR/\#define LOWIMPRR/" ir7.fluka
  sed -i "s/^\#define RESNUCLEI/\*\#define RESNUCLEI/" ir7.fluka
elif [ $sc = 'warm' ]
then                                                                        90
  sed -i "s/^\#define BLM/\*\#define BLM/" ir7.fluka
  sed -i "s/^\#define SCRCOLD/\*\#define SCRCOLD/" ir7.fluka
  sed -i "s/^\*\#define SCRWARM/\#define SCRWARM/" ir7.fluka
  sed -i "s/^\*\#define WARMSCORE/\#define WARMSCORE/" ir7.fluka
  sed -i "s/^\#define KATERINA/\*\#define KATERINA/" ir7.fluka                91
  sed -i "s/^\*\#define TUNBLACK/\#define TUNBLACK/" ir7.fluka
  sed -i "s/^\*\#define LOWIMPRR/\#define LOWIMPRR/" ir7.fluka
  sed -i "s/^\#define RESNUCLEI/\*\#define RESNUCLEI/" ir7.fluka
elif [ $sc = 'BLM' ]
then                                                                        92
  sed -i "s/^\*\#define BLM/\#define BLM/" ir7.fluka
  sed -i "s/^\#define SCRCOLD/\*\#define SCRCOLD/" ir7.fluka
  sed -i "s/^\*\#define SCRWARM/\#define SCRWARM/" ir7.fluka
  sed -i "s/^\#define WARMSCORE/\*\#define WARMSCORE/" ir7.fluka
  sed -i "s/^\#define KATERINA/\*\#define KATERINA/" ir7.fluka                93
  sed -i "s/^\*\#define TUNBLACK/\#define TUNBLACK/" ir7.fluka
  sed -i "s/^\*\#define LOWIMPRR/\#define LOWIMPRR/" ir7.fluka
  sed -i "s/^\#define RESNUCLEI/\*\#define RESNUCLEI/" ir7.fluka
```

```
elif [ $sc = 'sida' ]
then
  sed -i "s/^\#define␣BLM/\*\#define␣BLM/" ir7.fluka
  sed -i "s/^\#define␣SCRCOLD/\*\#define␣SCRCOLD/" ir7.fluka
  sed -i "s/^\#define␣SCRWARM/\*\#define␣SCRWARM/" ir7.fluka
  sed -i "s/^\*\#define␣WARMSCORE/\#define␣WARMSCORE/" ir7.fluka
  sed -i "s/^\#define␣KATERINA/\*\#define␣KATERINA/" ir7.fluka
  sed -i "s/^\#define␣TUNBLACK/\*\#define␣TUNBLACK/" ir7.fluka
  sed -i "s/^\#define␣LOWIMPRR/\*\#define␣LOWIMPRR/" ir7.fluka
  sed -i "s/^\#define␣RESNUCLEI/\*\#define␣RESNUCLEI/" ir7.fluka
elif [ $sc = 'resn' ]
then
  sed -i "s/^\#define␣BLM/\*\#define␣BLM/" ir7.fluka
  sed -i "s/^\#define␣SCRCOLD/\*\#define␣SCRCOLD/" ir7.fluka
  sed -i "s/^\*\#define␣SCRWARM/\#define␣SCRWARM/" ir7.fluka
  sed -i "s/^\#define␣WARMSCORE/\*\#define␣WARMSCORE/" ir7.fluka
  sed -i "s/^\#define␣KATERINA/\*\#define␣KATERINA/" ir7.fluka
  sed -i "s/^\*\#define␣TUNBLACK/\#define␣TUNBLACK/" ir7.fluka
  sed -i "s/^\*\#define␣LOWIMPRR/\#define␣LOWIMPRR/" ir7.fluka
  sed -i "s/^\*\#define␣RESNUCLEI/\#define␣RESNUCLEI/" ir7.fluka
fi

echo '#################### Changing ir7_phase lat files #########'

cd $IR7 # the collimator Twiss file changes with the loss scenario
  sed -i "s/FLUKA_summary.[a-z]*\.[a-z]*\.[a-z]*\.dat/
␣␣␣␣␣FLUKA_summary.$en.$pl.$op.dat/" ir7_phase1_abs.lat
  sed -i "s/FLUKA_summary.[a-z]*\.[a-z]*\.[a-z]*\.dat/
␣␣␣␣␣FLUKA_summary.$en.$pl.$op.dat/" ir7_phase1.lat
  if [ $en = 'inj' ]
  then
    sed -i "s/p_beam␣␣␣␣=␣7000.0/p_beam␣␣␣␣=␣␣450.0/" mklattic.r
    sed -i 's/BEAM             7000./BEAM              450./' ir7.fluka
#    sed -i "s/^\(BEAM[␣]*\)␣7000.0/\1␣␣450.0/" ir7.fluka
  elif [ $en = 'lwb' ]
  then
    sed -i "s/p_beam␣␣␣␣=␣␣450.0/p_beam␣␣␣␣=␣7000.0/" mklattic.r
    sed -i 's/BEAM              450./BEAM             7000./' ir7.fluka
#   sed -i "s/^\(BEAM[␣]*\)␣␣450.0/\1␣7000␣␣/" ir7.fluka
  fi
cd -

cd $IR7/Twiss
grep 'TCLA.*R7.B1' /home/LHC/IR7/Twiss/FLUKA_summary.$en.$pl.$op.dat
   > absorber_summary.dat

echo '#################### cleaning and compiling ###############'

cd $IR7
source cluster_path
version=`echo $FLUPRO | sed 's/soft\//#/' | cut -d# -f2`
echo $version
if [ $version = 'flukadev' ]
then
  echo 'Running activation calculations, FLUKA development version.'
  cp makefile.activation makefile
```

```
  cp rundev.sh run.sh
else
  echo 'Running energy deposition calculations, FLUKA standard version.'
  cp makefile.energy makefile
  cp runnor.sh run.sh
fi
make proper
make
j=0
chmod a+x $IR7/flukair7
echo ' '

echo '#################### adding usrbdx for end current ########'
sed -i "s/\*CSRBDX/USRBDX/" ir7_tmp.inp
make

echo '#################### Checking type of simulation ##########'

grep '^ASSIGNMAT' ir7.fluka | grep BLCKHOLE | grep -v ' BLACK'
  | grep -v 'TUN_' | grep 'UJ' > t0055
if [ $? -eq 0 ]
then
  echo 'NO SIMULATION IN UJs'
fi
grep '^ASSIGNMAT' ir7.fluka | grep BLCKHOLE | grep -v ' BLACK'
  | grep -v 'TUN_' | grep 'RR' > t0055
if [ $? -eq 0 ]
then
  echo 'NO SIMULATION IN RRs'
fi
grep '^ASSIGNMAT' ir7.fluka | grep BLCKHOLE | grep -v ' BLACK'
  | grep -v 'TUN_' | grep 'TCP' > t0055
if [ $? -eq 0 ]
then
  echo 'NO SIMULATION from TCP?? on'
  echo 'SIMULATION IN STRAIGHT SECTION'
  j=1
fi
grep '^ASSIGNMAT' ir7.fluka | grep BLCKHOLE | grep -v ' BLACK'
  | grep -v 'TUN_' | grep 'TCS' > t0055
if [ $? -eq 0 ]
then
  echo 'NO SIMULATION from TCS?? on'
  echo 'SIMULATION IN STRAIGHT SECTION'
  j=1
fi
grep '^ASSIGNMAT' ir7.fluka | grep BLCKHOLE | grep -v ' BLACK'
  | grep -v 'TUN_' | grep 'TCL' > t0055
if [ $? -eq 0 ]
then
  echo 'NO SIMULATION from TCL?? on'
  echo 'SIMULATION IN STRAIGHT SECTION'
  j=1
fi
grep '^ASSIGNMAT' ir7.fluka | grep BLCKHOLE | grep 'C5AIRO  C5B2LAY1' > t0055
if [ $? -eq 0 ]
```

```
then
  echo 'SIMULATION IN STRAIGHT SECTION'
  j=1
fi

grep '^#define EMFCUT' ir7.fluka > t0055
if [ $? -eq 0 -a $j -eq 1 ]
then
  echo 'WARNING!! only simulating straight section but with high EMFCUTS!!'
fi
grep '^#define EMFCUT' ir7.fluka > t0055
if [ $? -eq 1 -a $j -eq 0 ]
then
  echo 'activate define EMFCUT to gain speed'
fi
grep '^#define EMFCUT' ir7.fluka > t0055
grep '^USRBIN' ir7.fluka | grep 'D:' > t0055

echo '#################### Checking special dose cards  #########'
if [ $? -eq 1 -a $j -eq 0 ]
then
  echo 'ERROR! Dose card (D:) should be added manually to .inp file'
fi
echo ' '
rm t0055

echo '#################### Editing ir7.inp & run.sh #############'

cd $IR7
if [ $# -eq 0 ]
then
 sed -i "s/\(^SOURCE[ ]*\)\([0-9a-Z]*\)/\1$pl$e$o/" ir7.inp
 grep $pl$e$o run.sh
 if [ $? -ne 0 ]
 then
   echo ADDING $pl$e$o.dat to run.sh
   sed -i "s/SOURCES=\"/SOURCES=\"Twiss\/$pl$e$o.dat /" run.sh
 fi
else
 file=`echo $1 | sed "s/TCSG/TCS/" | sed "s/TCLA/TCL/"
   | sed "s/\([a-Z]*[0-9]\)[a-Z0-9]*/\1/"`
 sed -i "s/\(^SOURCE[ ]*\)\([0-9a-Z]*\)/\1$file$e$p$o/" ir7.inp
 grep $file$e$p$o run.sh
 if [ $? -ne 0 ]
 then
   echo ADDING $file$e$p$o.dat to run.sh
   sed -i "s/SOURCES=\"/SOURCES=\"Twiss\/$file$e$p$o.dat /" run.sh
 fi
fi

if [ $en = 'inj' ]
then
  sed -i "s/^\(BEAM[ ]*\) 7000.0/\1  450.0/" ir7.inp
elif [ $en = 'lwb' ]
then
  sed -i "s/^\(BEAM[ ]*\)  450.0/\1 7000.0/" ir7.inp
```

```
fi

if [ $sc = 'warm' ]
then
  sed -i 's/200.0MBWB6L/200.0D:WB6L/' ir7.inp
  if [ $? -ne 0 ]
  then
    echo 'D: replacement in MBW pipe could not be done'
  fi
  sed -i 's/160.0MQWAE5L/160.0D:WAE5L/' ir7.inp
  if [ $? -ne 0 ]
  then
    echo 'D: replacement in MQW pipe could not be done'
  fi
  sed -i 's/157.4MQWAE5L/157.4D:WAE5L/' ir7.inp
  if [ $? -ne 0 ]
  then
    echo 'D: replacement in MQW could not be done'
  fi
  sed -i 's/155.4MQWAE5L/155.4D:WAE5L/' ir7.inp
  if [ $? -ne 0 ]
  then
    echo 'D: replacement in MQW could not be done'
  fi
  sed -i 's/159.4MQWAE5L/159.4D:WAE5L/' ir7.inp
  if [ $? -ne 0 ]
  then
    echo 'D: replacement in MQW could not be done'
  fi
  sed -i 's/157.4MQWAD5L/157.4D:WAD5L/' ir7.inp
  if [ $? -ne 0 ]
  then
    echo 'D: replacement in MQW could not be done'
  fi
  sed -i 's/155.4MQWAD5L/155.4D:WAD5L/' ir7.inp
  if [ $? -ne 0 ]
  then
    echo 'D: replacement in MQW could not be done'
  fi
  sed -i 's/159.4MQWAD5L/159.4D:WAD5L/' ir7.inp
  if [ $? -ne 0 ]
  then
    echo 'D: replacement in MQW could not be done'
  fi
fi
```

9 Conclusions

The tertiary halo which escapes the collimation system in IR7 may heat some fragile elements up to unacceptable levels, if no additional absorber is used. The assessment of the energy deposition in sensitive components requires extensive simulations with a multi particle cascade code like FLUKA. The exact shape of the

elements, their location and the used materials play an important role in the development of the hadronic shower. For this reason the straight section and the dispersion suppressors (DS) of IR7 were fully implemented with a high degree of sophistication, including all relevant components and details up to realistic limits.

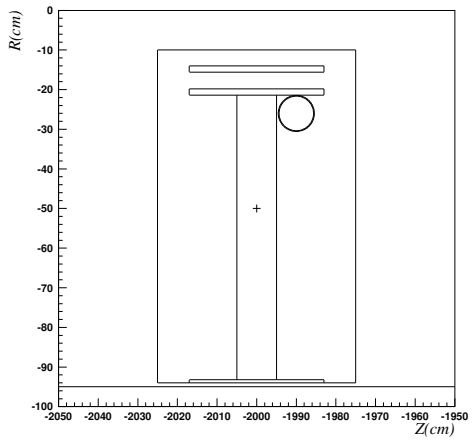
Special attention was paid to the definition of the magnetic field. Algorithms for linear interpolation as well as analytical expressions were employed to provide the field intensity inside the magnets. The orientation and the jaw aperture of all primary and secondary collimators is defined in such a way that it is easy to make any change in their definition, if any modification applies to the layout. After several tests the region-importance biasing could be effectively set, in order to minimize the CPU time without losing in accuracy of the simulations. Special options allow focusing the CPU effort on selected areas of the tunnel, depending on the needs of the user.

The FLUKA model of IR7 can be used for any energy deposition study, including radiation damage to electronics and response of detectors and monitors along the beam line. With extra effort on the material assignment, it can also be used for estimating the material activation and the dose rate expected after ceasing operation. Although the complexity of the system requires some expertise, the intrinsically modular approach that has been adopted allows the user to modify the geometry according to her/his needs. Moreover, special user written routines can be included to treat specific cases and make the study more flexible. In addition, the whole geometry is automatically implemented from a file with the location of the elements. With minor changes the geometry of similar sections of LHC (e.g., IR3) could be implemented with the same level of accuracy.

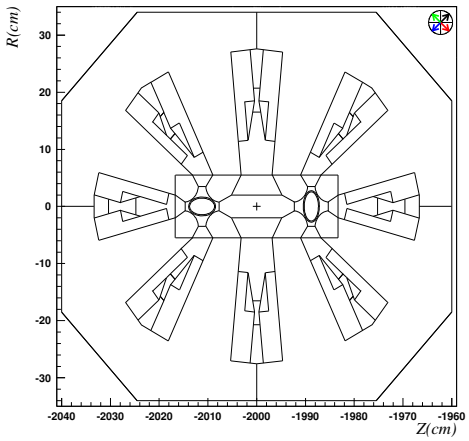
A Front cross-sectional views of the FLUKA prototypes

This appendix provides a cross-sectional front view for every prototype that was used to implement the IR7 section. In the rendering, attention was payed to the FLUKA regions rather than the material assignments: this explains the presence of division lines between parts of the same elements (e.g., the iron yoke of the warm quadrupole). Scales are in cm. The following list gives the prototype name, its position in the FLUKA parking (z coordinate in cm), the real element it represents and the label of the corresponding picture.

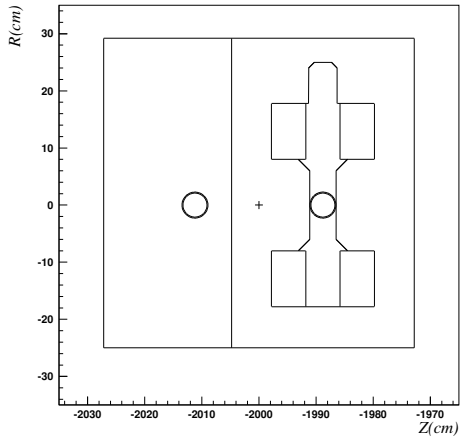
1. BLMBODY, 6570, BLM, fig. 9(a)
2. B_QUAD1, -20, MQW, fig. 9(b)
3. CD2MB_R1, 650, MCBWV, fig. 9(c)
4. CD_MB_R1, 400, MCBWH, fig. 9(d)
5. MBW1_BB, 2980, MBW, fig. 9(e)
6. MB_BODY, -4445, MB, fig. 9(f)
7. MCBCBODY, -600, MCBC, fig. 9(g)
8. MCBHBODY, 2200, MCBC, fig. 9(h)
9. MCDOBODY, -400, MCDO, fig. 9(i)
10. MCS_BODY, -500, MCS, fig. 9(j)
11. MQ_BODY, -380, MQ, fig. 9(k)
12. MQT_BODY, -850, MQT, fig. 9(l)
13. MQT_CYLO, -900, MQTL, fig. 9(m)
14. MS_BODY, 2000, MS, fig. 9(n)
15. TCP60, 5947.5, TCP, fig. 9(o)



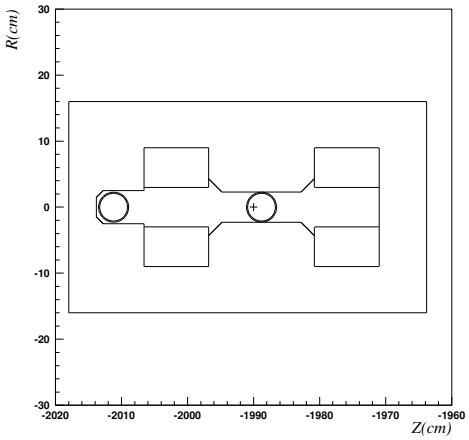
(a) **BLM**



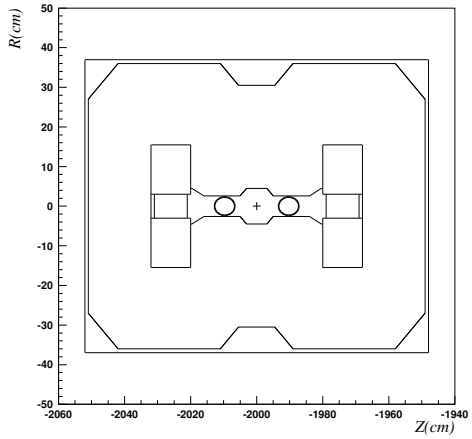
(b) **B_QUAD1**



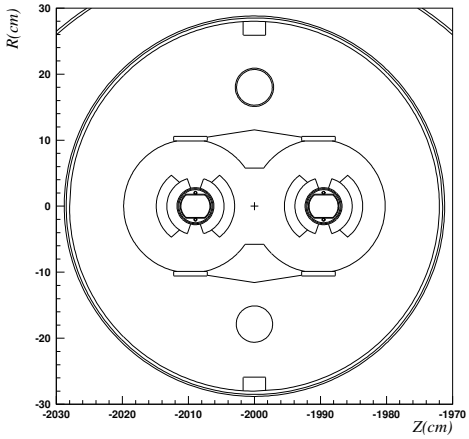
(c) **CD2MB_R1**



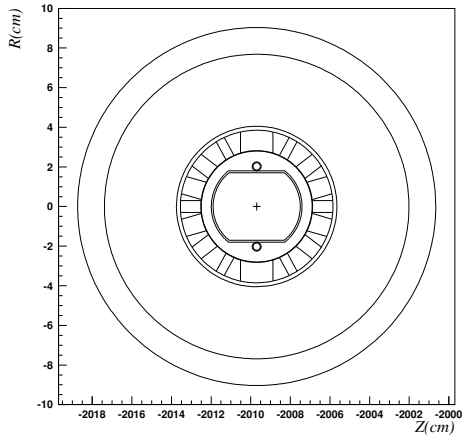
(d) **CD_MB_R1**



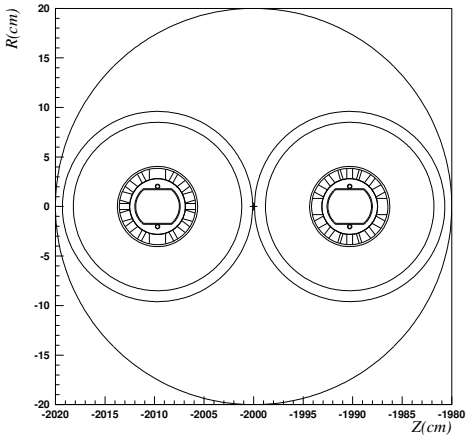
(e) **MBW1_BB**



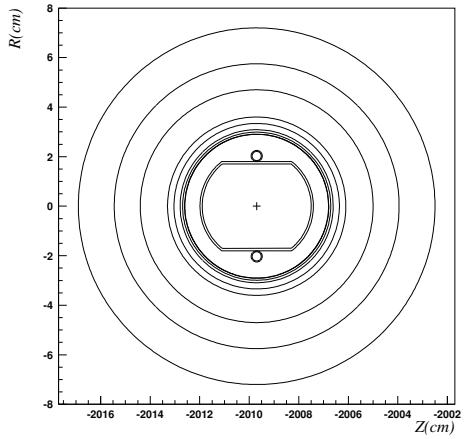
(f) **MB_BODY**



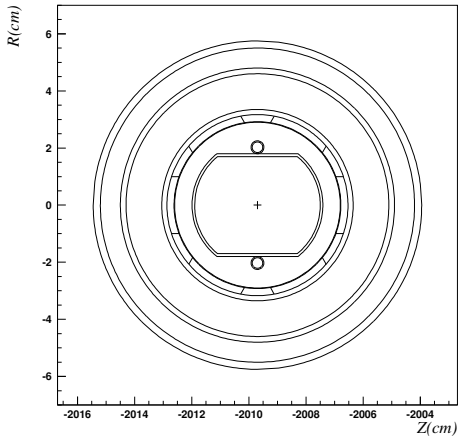
(g) **MCBCBODY**



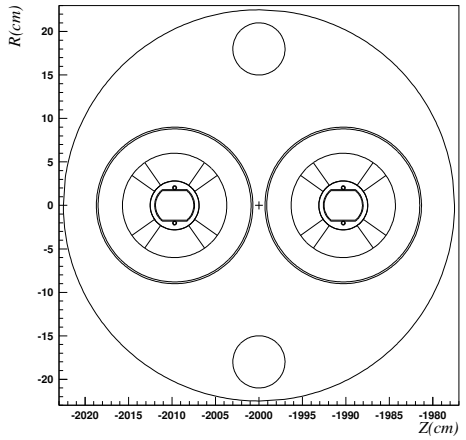
(h) **MCBHBODY**



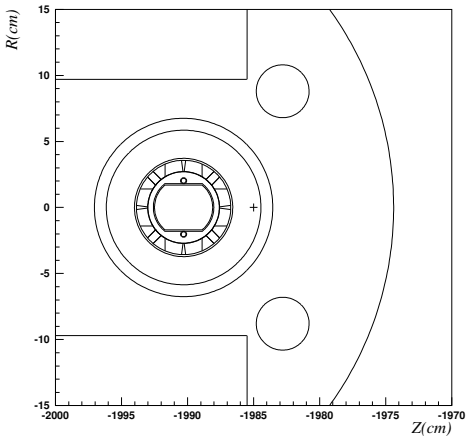
(i) **MCDO_BODY**



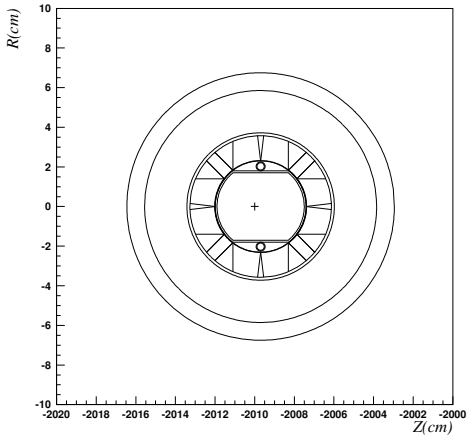
(j) **MCS_BODY**



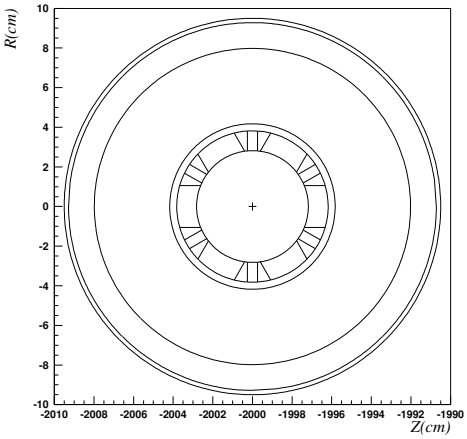
(k) **MQ_BODY**



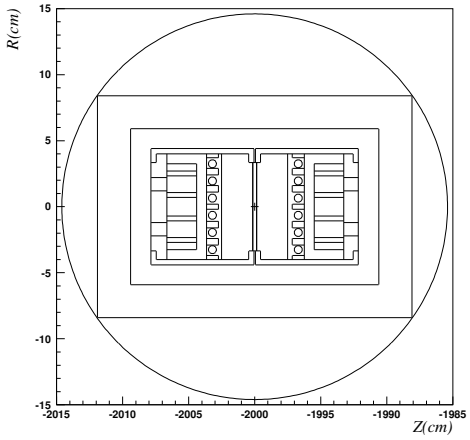
(l) **MQT_BODY**



(m) **MQTCYLO**



(n) **MS_BODY**



(o) **TCP60**

References

[1] A. Fassò, A. Ferrari, S. Roesler, P.R. Sala, G. Battistoni, F. Cerutti, E. Gadioli, M.V. Garzelli, F. Ballarini, A. Ottolenghi, A. Empl, and J. Ranft. The physics models of FLUKA: Status and recent developments. In *Computing in High Energy and Nuclear Physics 2003 Conference (CHEP2003), La Jolla, CA, USA, March 24-28, 2003*, volume arXiv:hep-ph/0306267, March 2003.

[2] A. Fassò, A. Ferrari, J. Ranft, and P.R. Sala. *FLUKA: a multi-particle transport code*. CERN, INFN, SLAC, 2005.

[3] Persistence Of Vision Pty. Ltd. *Persistence Of Vision*. Pty. Ltd, Williamstown, Victoria, Australia, 2004. Software.

[4] V. Vlachoudis. Movie of the fluka ir7 layout. url: http://lhc-collimation-project.web.cern.ch/lhc-collimation-project/movie-IR7-FLUKA.htm.

[5] M. Magistris, M. Santana-Leitner, V. Vlachoudis, and A. Ferrari. Optimization of the active absorber scheme for the protection of the dispersor supressor. Technical report, CERN-AB-ATB, 2006.

[6] Lamont M. Estimates of annual proton doses in the lhc. Lhc project note, CERN AB/OP, October 2005. 375.

[7] M. Santana-Leitner, M. Magistris, V. Vlachoudis, and A. Ferrari. Fluka simulations for the optimization of the beam loss monitors. Technical report, CERN-AB-ATB, 2006.

[8] Markus Brugger and Stefan Roesler. Personal Communication, 2004.

[9] V. Vlachoudis, A. Ferrari, M. Magistris, M. Santana-Leitner, and K. Tsoulou. Consequences of regular and irregular beam impact on the LHC collimators. In IL American Nuclear Society, LaGrange Park, editor, *MC2005 proceedings*, CERN CH-1211 Geneva-23 Switzerland, 2005. CERN-AB.

[10] V. Vlachoudis. Brexx overview. In Rexx Symposium 2001, editor, *Rexx Symposium 2001 proceedings*, CERN CH-1211 Geneva-23 Switzerland, 2001. CERN-AB. Published.

[11] G. Robert-Demolaize, R. Assmann, S. Redaelli, and F. Schmidt. A new version of sixtrack with collimation and aperture interface. In Particle Accelerator Conference 05, editor, *PAC05 proceedings*, CERN CH-1211 Geneva-23 Switzerland, 2005. CERN. Published.

[12] Collimation Site. New long collimators. url: http://lhc-collimation-project.web.cern.ch.

[13] Magistris M., Santana-Leitner M., Dehning B., Ferrari A., Stockner M., and Vlachoudis V. Simulations for the response of beam loss monitors in ir7 insertion in lhc. In *EPAC06*, June 2006. Submitted.