Alma Mater Studiorum · University of Bologna

School of Science
Department of Physics and Astronomy
Master Degree in Physics

GEANT4 Radioprotection studies of the ERHA system for protontherapy treatment

Supervisor:
Prof. Mauro Villa

Co-supervisor:
Prof. Roberto Spighi

Submitted by:
Nicola Ferrara

Academic Year 2021/2022

Abstract

This work is focused on the radiation protection for a protontherapy facility. The aim is to simulate with the best accuracy the prompt radiation field of the proton accelerator situed in Ruvo di Puglia, owned by Linearbeam s.r.l. company. In order to simulate it, is used Geant4, a software for interaction simulations of particles with matter. Thanks to internship work, thesis speaks about cancer therapy with a new method for particle acceleration, a linear beam. For a complete overview of the therapy, this work starts with a crush course on interactions of particle with matter, goes specifically to biological matter, then is shown a brief introduction to shielding studies for a particle acceleration facility, and then a presentation of Geant4. At the end, the main aspects of the proton accelerator are simulated, from proton hitting material of beam-pipe to detectors used to measure dose.

Contents

Introduction

Cancer is a large group of diseases that can start in almost any organ or tissue of the body when abnormal cells grow uncotrollably, go beyond their usual boundaries to invade adjoining parts of the body.
Cancer is the second leading cause of death globally, accounting for an estimated 9.6 million deaths, in 2018 [1]. There are numerous methods of treatment, like for examples surgery, chemiotherapy, immunotherapy, radiotherapy ($\approx$ 50% of all cancer patients) and hadrontherapy, also called Ion Beam Therapy.
Hadrontherapy had a rapid development in recent years due to the progresses in technology and radiation oncology techniques. In the last 50 years there was a continuous expansion, at the end of 2016 there were 61 proton centres and 9 heavy ion centres, with 149345 patients treated with proton and 21580 patients treated with ^{12}C.
Hadrontherapy represents a valid alternative to the conventional radiotherapy that uses photons or, more rarely, electrons. Comparing these therapies, hadrontherapy advantages comes from the different lost energy mechanism. The dose release profile of radiation with photons presents a peak at short distance from the patients's skin followed by a decreasing release of the radiation in accordance with the absorption law. On the other hand, hadrontherapy presents a low dose profile at the beginning of the path and a sharp maximum, *Bragg peak*, near the end. This therapy provides a high irradiation accuracy of the tumor volume and a reduced damage to the surrounding healthy tissues. The particles involved in this therapy present high capability of inducing a direct damage to the DNA of cancerous cells.
Unfortunately, when using these beams, there is a major increase in the presence of fragments derived from the nuclear interactions of the beam and the patient tissues. Consequently a dose is released in the entry region and beyond the Bragg peak, still not completely studied. Proton treatments are the most widespread but even in their employment occurs the problem of fragmentation.
FOOT (FragmentatiOn Of Target) experiment was designed with purpose of providing for the lack of experimental measurements of nuclear reaction cross sections for fragments produced in the interaction between tissue nuclei and charged particles of the beam, with the task of study both projectile and target fragmentation where the latter was neglected by previous experiments. Fragmentation problem vanishes for protontherapy, because proton interaction are mostly due to ionization.
In Chapter 1 is described how particles interact with matter, with an overview of the

most probable mechanisms. In Chapter 2 is presented an introduction to the effects at different energies, for different particles, of the interaction of particles with biological matter, focused on what happens when radiation is used to kill cancers.
In Chapter 3 is described the Linearbeam accelerator facility, an overview of the ERHA project, a detailed description of the components. In Chapter 4 is explained how to shield from steady-state radiation fields, with a focus on the protection study of an accelerator facility, involving some fundamentals concepts and quantities. In Chapter 5 is shown the GEANT4 simulation work, main object of this thesis, in which in three steps is built a very similar model of the accelerator facility in order to obtain the gamma and neutron dose distribution inside bunker.
In Chapter 6 are reported measured quatities, dosimeter and set-up description, in order to compare them with simulation.

Chapter 1

Radiation interaction with matter

A brief overview of the physical process that occurs when charged and neutral particles cross matter is given in this chapter. Charged particles interact via Coulomb force and nuclear fragmentation, while neutrons could scatter with nuclei and produce secondary charged particles.

I. Cross Section

The cross section is a measurement of the probability that a reaction happens. Let's imagine a diffusion experiment in which a beam of particles smashes on the surface of the target material and interacts all along the thickness δx, like in figure 1.1[2]. N_i is the number of particles of the beam, and n_b is the density of scattering centers, so the product $n_b \delta x$ is the number of scattering centers for a unitary surface. Now supposing that each particle in the beam can interact on average with at most one and only one particle of the target, the number of reactions in unit of time $\frac{dN_r}{dt}$ is proportional to number of particle of the beam in the same unit of time, and $n_b \delta x$. In order to write the proportionality:

$$\frac{dN_r}{dt} = \sigma \frac{dN_i}{dt} n_b \delta x \tag{1.1}$$

So σ is like a surface, and it is measured in *barn*, *1 barn* corresponds to $10^{-24} cm^2$.

II. Interaction of charged particles

Particles with electric charge could interact with electrons and nuclei, depending on cross section: when a particle interacts with the entire atom cross section is much bigger than when it interacts with the nucleus, so the most probable process happens with atoms, and for the sake of this thesis it is possible to distinguish between three main fenomena: *bremsstrahlung*, cherenkov and ionization.

i. Bremsstrahlung

Bremsstrahlung consists of light emission from a charged particle of mass m that interacts with an electromagnetic field, specifically the Z charged nucleus's field, equal to:

$$-\frac{dE}{dx} = \frac{4N_a Z^2 \alpha^3 (hc)^2}{m^2 c^4} E \ln(\frac{183}{Z^{\frac{1}{3}}}) \sim \frac{Z^2}{m^2} \tag{1.2}$$

This contribution is negligible for protons and ions due to $\frac{1}{m^2}$.

ii. Cherenkov

This fenomena is due to light emission that occurs if a charged particle travels a medium faster than speed of light in that medium. The cause of this emission is linked to the polarization and depolarization of matter when particle crosses it. Every point of the trajectory gives rise to a spherical wave front, with $v = c/n$. So the threshold for which fenomena happens is $\beta > 1/n$.

iii. Ionization

Charged particles main process is the ionization or excitation of atoms, and the mean rate of energy lost per unit lenght of material is called *stopping power* and it is expressed by the Bethe-Block formula:

$$\begin{aligned} -< \frac{dE}{dx} >= \frac{2\pi n_e r_e^2 m_e c^2 z^2}{\beta^2} \Bigg[\ln\left(\frac{2m_e c^2 \beta^2 T_{max}}{I^2(1-\beta^2)}\right) - 2\beta^2 + 2zL_1(\beta) + 2z^2 L_2(\beta) \\ - 2\frac{C}{Z} - \delta + G \Bigg] \end{aligned} \tag{1.3}$$

- β charged particle velocity in c units
- I mean excitation energy, material-dependent

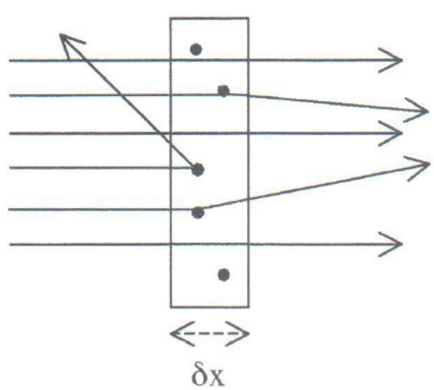

Figure 1.1: *Diffusion experiment on a target.*

- δ density correction
- C is the shell correction, important at low energies
- T_{max} maximum energy transfer to an electron (from kinematics)
- L_1 Barkas correction (z^3) responsible for the difference in stopping power for particles-antiparticles
- L_2 Bloch (z^4) correction
- G Mott corrections

The formula is valid for particles with mass bigger than the electron mass. The *Barkas effect* was discovered by Walter Barkas, who saw that positive and negative pions with same energy that went through emulsion material had different ranges. This difference relies on the sign of the electromagnetic force, positive particles bring near electrons while negative particles repel them, changing the mean charge density of the material.
The energy lost shows a distribution that depends upon the thickness of the material. For thick materials a large number of ions was created (from 30000 to 70000 ions pair for α particle in air), leading to a Gaussian distribution. For thin materials, the energy straggling in statistical fluctuations depends upon a little number of interactions with a huge loss of energy. This distribution is called Landau 1.2. In Figure 1.3 there is a plot of BetheBlock for different materials.

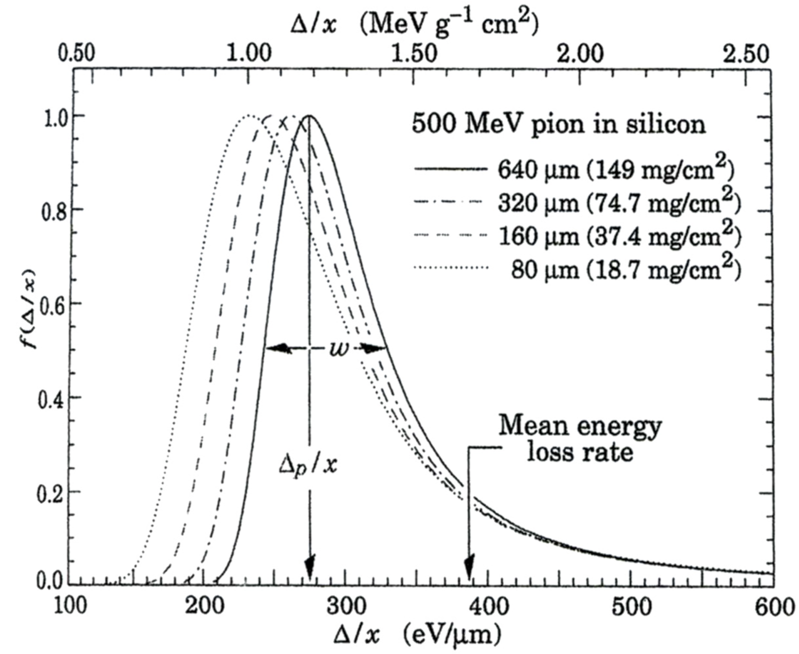

Figure 1.2: *Plot of the Landau distribution, where Δ is the energy loss and x is the thickness[3].*

This shows also that particles lose their energy for very low energy values, so at the end of the *Range*. This quantity is the mean lenght of a particle inside materials. In order to obtain an analytic formula for the *Range* a method is integrate over all the deposited energy, but it's not easy.

$$R = \int_0^{T_0} \left(-\frac{dE}{dx} \right)^{-1} dE \tag{1.4}$$

This is the formula for the range, where E_0 is the starting energy. This approach neglects that energy loses are stochastic in nature and that secondary particles posses an energy and range distribution. In addition electrons do not have a straight path but suffer multiple scatter with many changes in direction [4]. In order to simplify the integral it could be useful the *Bragg-Kleeman approximation*, under the condition of a *Continuous Slowing Down Approximation* (CSDA) in which particles lose their energy slowly and continuously. The condition led to a simplification in a very common situation, in which particles go through a compound with different stopping power, giving this formula:

$$\frac{dE}{d\chi} = \sum_i^N W_i \left(\frac{dE}{d\chi} \right)_i \tag{1.5}$$

in which the χ is the massive lenght, $\chi = \rho x$, and W_i is the fraction of material in the compound.

Massive particles Particles that have got a mass bigger than electrons are heavy or massive, like protons or α. They lose energy inside materials by ionization. When particles lose energy, their velocity decreases and lose more and more energy, with a ionization density peaked at the end of range, so the deposited energy quantity reached a peak, called *Bragg peak* (see Figure 2.6), where $\beta \sim \frac{z^{2/3} v_0}{c}$, and $v_0 = \frac{e^2}{\hbar}$. At peak is reached the maximum effect, as is explained in Chapter 2.

Electrons and positrons Electrons lose energy by Bethe-Block with a change, due to the identity between projectile and target particles. Bremsstrahlung overcomes ionization at a critic energy value, that depends on material, but in general it is comparable with ionization at $10 MeV$.

III. Interaction of neutral particles

Neutral particles don't lose energy ionizing matter, processes are more complicated. First of all, it's fundamental to distinguish between photons and neutrons: photons haven't mass, so main interactions with ordinary matter is pure electromagnetic. Neutrons are massive particle that could interact with nuclei, by elastic or anelastic collisions.

i. Photons

Photons interact with matter at atomic and nuclear level, with different processes. First photon could be absorbed, giving rise to an electron, or it could scatter on an electron, while at nuclear level, a photon could disapper in a couple of electron-positron. These particles go deeper into matter more than charged particles, and they don't lose energy smoothly but all the energy is lost. So, in order to understand a photon beam interaction, is useful to derive the intensity of the beam:

$$I(x) = I_0 e^{-\mu x} \tag{1.6}$$

where I_0 is the incident intensity, x is the thickness of material, μ is the *attenuation coefficient*. So it is linked to the cross section: $\mu = \frac{\rho N_A \sigma}{A}$.

Photoelectric effect The photoeffect is the first demonstration of quantum physics, photon is absorbed by atom, and the electron ejected has got an energy:

$$E = h\nu - \phi \tag{1.7}$$

where ϕ is the binding energy. The cross section over the threshold goes down as energy increases. Electron leaves a hole inside atom, that will be filled by an electron of another shell, and this process goes on in order to ri-estabilish an equilibrium of electrons, giving rise to emission of photons. Cross section of photoeffect is $\sigma \propto \frac{Z^{4\div5}}{E^{3,5}}$

Compton effect When electrons have more energy and very small binding energy, they are free. A photon hitting these electrons may be scattered and lose energy, changing its frequency. The cross section is $\sigma \propto \frac{Z}{E}$.

Pair production Fenomena of pair production happens when energy of photons overcomes $2m_ec^2$. It's made possible by the presence of matter: a nucleus interaction that absorbs the momentum. Just a simplified description, but a very clear explenation is possible in the framework of Quantum Field Theory. Cross section is $\frac{Z^2}{\ln E}$. In order to visualize different interaction it is needed to define *mass attenuation coefficient* $\frac{\mu}{\rho}$, as showned in figure 1.4.

ii. Neutrons

Neutrons could interact only by strong force with nuclei. This interaction is a short range force, so neutrons have to reach a very small distance from nuclei on order to interact, less than $10^{-15}m$. For this reason neutrons go very deep into matter. Neutrons should be divided by their energy, like in Table 1.1 All the possible interaction of neutrons with nuclei are listed below:

1. ***Elastic Scattering***: main lost energy mechanism into MeV region.
2. *Anelastic Scattering*: nuclei is excited and decays with gamma or other type of decay. Always possible into MeV region.
3. ***Radiative capture***: $n + (Z, A) \rightarrow \gamma + (Z, A + 1)$, and $\sigma \propto \frac{1}{v}$, with v velocity of neutron.

4. ***Nuclear Reactions***: neutron is captured and other charged particles are emitted. Cross section goes as the radiative capture, but there could be some resonances, into a range from eV to keV.

5. ***Fission***: this fenomena happens for thermal neutrons.

6. ***Hadronic cascade***: over 300 MeV.

In many situation, like detectors or nuclear plants, a very important process to study is the *moderation* of neutrons, a method to decrease their energy. When they go inside matter, they scatter with nuclei before they reach thermal or cold energy. At this point it is very probable a capture reaction. A specific explenation of nuclear reaction in a detector is written in chapter 6.

IV. Nuclear fragmentation

The problem of hadrontherapy is that at energy of 10^2 MeV/nucleon, fragmentation is the most frequent nuclear interaction: the projectile collides with target pheriferically, with few nucleons participating and products are separated in quasi projectile fragment and quasi target fragment both spectator of the interaction, while the region of interaction fragments in few nucleons like in Figure 1.5. The primary beam particle, like ^{12}C nuclei, can produce lighter fragments which, with lower energy, will continue to move through the material releasing energy inside it. The dose released beyond the Bragg peak depends on the charge of the particle: it is smaller if particles are protons (around 15 %), larger if they are ^{12}C nuclei and even more if ^{20}Ne nuclei. Depending on impact parameter between the particle and other nuclei of the material we could have a situation with one single fragment or many much lighter fragments, depending on the impact parameter. Fragments from quasi-projectile have velocity almost equal to velocity of the beam and are emitted in a narrow angle and then those fragments have a larger range with respect to the beam, because they are lighter and have almost same velocity.
The other fragments from quasi-target have a wider angular distribution and lower energies so they will stop before the quasi-projectile. This is the case of light particles such as protons, deuterons and helium. The dose beyond the Bragg peak comes from the quasi-projectile contribution whereas the wide angular halo comes from the quasi-target

Table 1.1: *Classification of neutrons by their energies.*

Neutron	Energy
High Energy	> 100 MeV
Fast	100 keV $\div$ 100 MeV
Epithermal	100 meV $\div$ 100 keV
Thermal or Slow	25 meV
Cold or Ultracold	1 μeV $\div$ 1 meV

fragments which have larger lateral displacement.
It's possible to study with Monte Carlo simulation the informations about cross-sections and other relevant parameters. The goal is to study how many fragments are obtained and of which energy. An example of fragments distributions is in Figure 1.6. Fragments play an important role in spreading collateral damage to the healthy tissues crossed by the incident particles. Since the fragments have higher range and different directions, they cannot be neglected in treament planning.

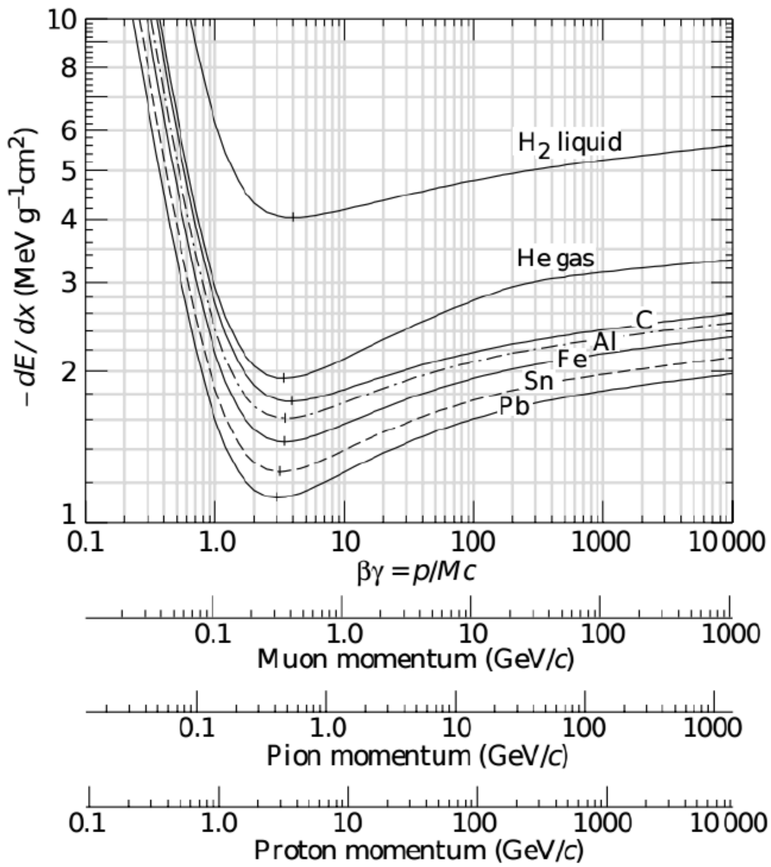

Figure 1.3: *Plot of the BetheBlock formula[4].*

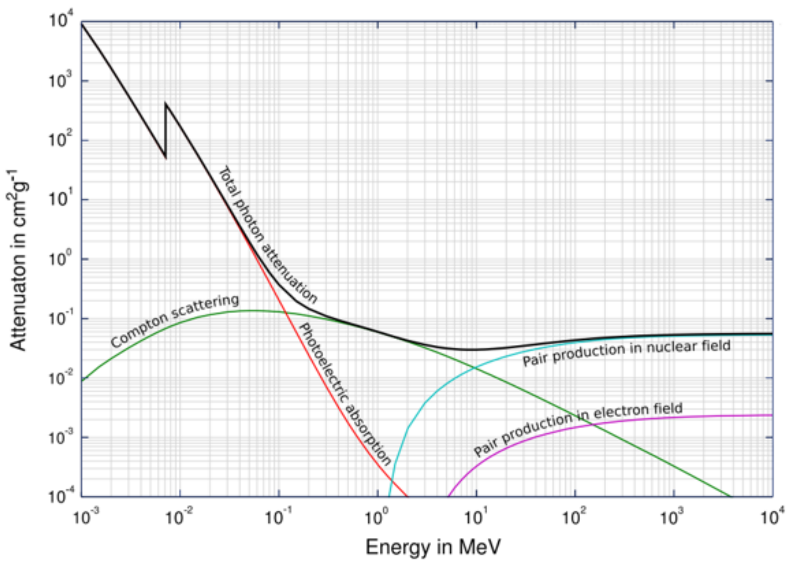

Figure 1.4: *Plot of the attenuation coefficients for Fe [5].*

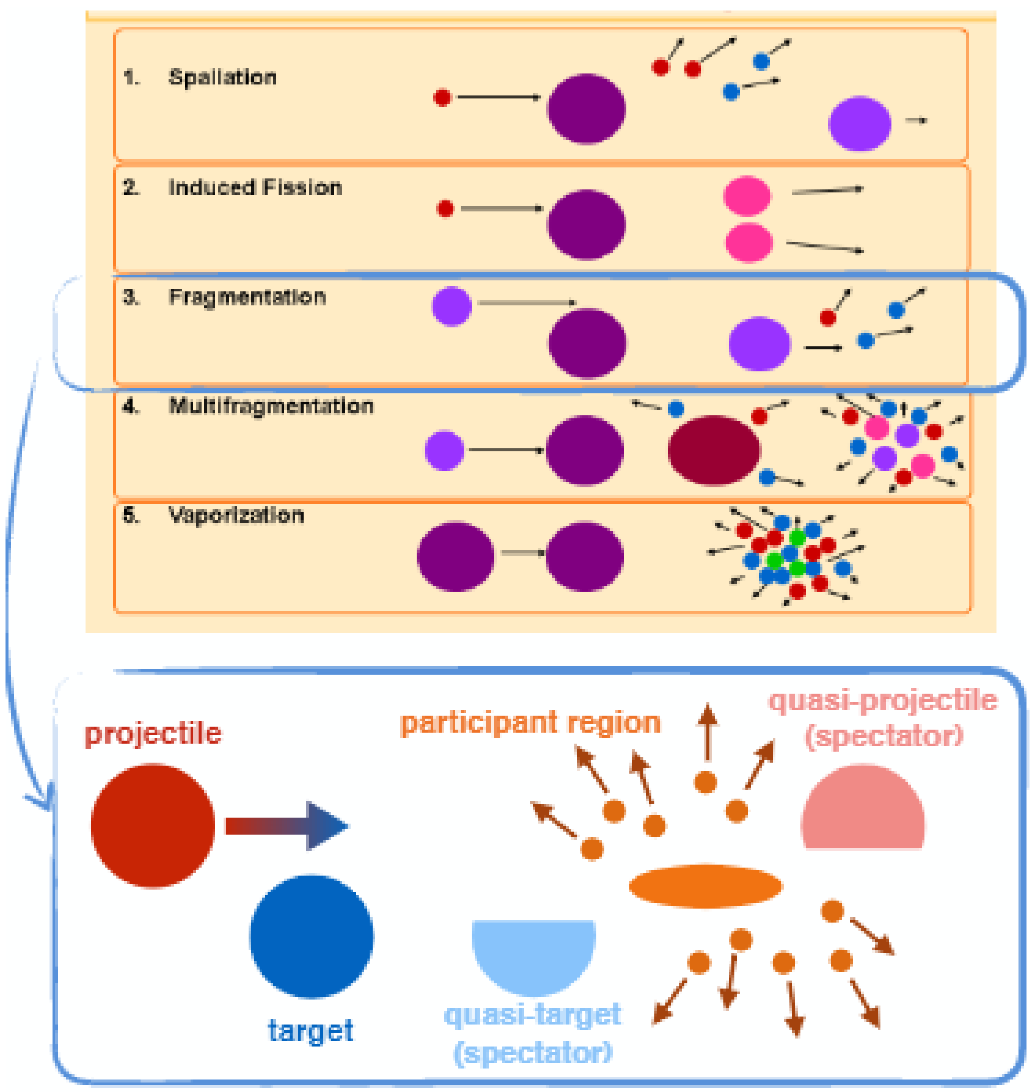

Figure 1.5: *An illustration of the nuclear interactions with a zoom on the fragmentation.*

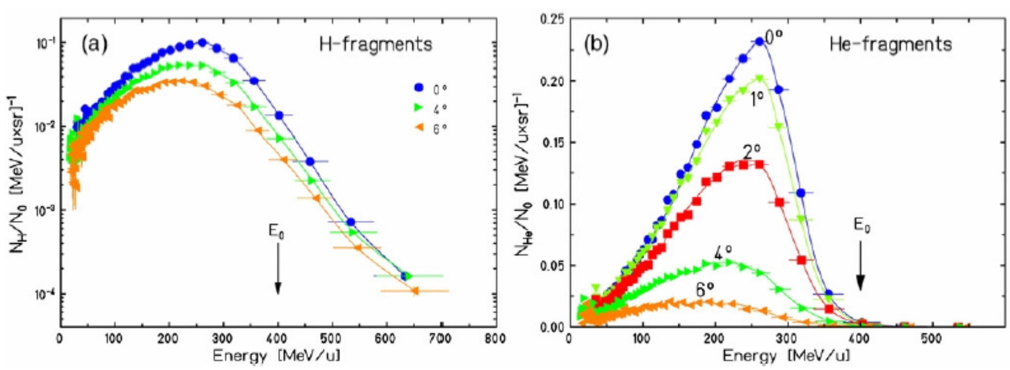

Figure 1.6: *Energy spectrum of secondary particles, Hydrogen (a) and Helium (b), produced in the fragmentation of a* ^{12}C *beam at 400 MeV/u in a water absorber 27,9 cm deep; distinct curves correspond to different emission angles (*0°, 4°, 6° *on the left and* 0°, 1°, 2°, 4°, 6° *on the right)[6].*

Chapter 2

Radiation biology

Biological radiation actions comprises all levels of biological organization. The interaction of radiation and biological system is as old as life itself, it certainly played a central role in evolution of self-organizing structures. The first effect of radiation action on biological systems is the transfer of energy to essential cellular components. This requires the introduction of different concepts and quantities.

I. Biological radiation effects

Ioniziting radiations are made of particles that ionize atoms while they cross matter. The interaction of these particles, charged or not, light or heavy depends upon several factors, like energy. Photons with MeV energy can mainly do Compton and pair production and produce electrons and positrons of not so high energies which are light particles so they would lose energy through Bethe-Block. On the other hand a MeV neutron can interact with nuclei scattering on them. Nuclei are charged more than one so they will lose their energy in a small portion of tissue.

i. Dose

Every physical or biological effect induced in the tissue is a consequence of the energy deposited in the matter by radiation. It's important to define the quantity of energy deposited by radiation per unit mass: the absorbed dose

$$D = \frac{dE}{dm} \tag{2.1}$$

defined as the expectation value of the absorbed energy divided by the mass of the volume. The unit of dose is the *Gray* (Gy) which equals 1 J/kg. An older and officially outdated unit is the *rad* (rd):

$$1rd = 100erg/g = 0.01Gy \tag{2.2}$$

Equivalent dose The radiation received corresponds to dose, but different exposure (electrons, protons, neutrons, gamma, ...) results into different physical damages. Stochastic effects occurrence depends not only on the amount of energy absorbed, but also on the nature of the radiation generating the dose. This difference is taken into account by weighting the absorbed dose by a factor related to the quality of the radiation, called the radiation weighting factor and written w_R. The equivalent dose, written H_T, in a tissue or organ is given by the following expression:

$$H_T = \sum_R D_{T,R} * w_R \tag{2.3}$$

where H_T is the equivalent dose for organ T, w_R is the weighting factor for radiation R, $D_{T,R}$ is the average absorbed dose in tissue or organ T due to radiation R. ICRP Publication 60 assigns the weighting factors w_R with each type of radiation, as shown in Table 2.1. Unit for equivalent doses is the Sievert (Sv). The values of w_R are the result of estimates

Table 2.1: *Weighting factor w_R for different types of radiation, ICRP Publication 60.*

Radiation	Energy	Values of w_R
Photons	-	1
Electrons	-	1
	< 10 keV	5
	10 - 100 keV	10
Neutrons	100 keV - 2 MeV	20
	2 - 20 MeV	10
	> 20 MeV	5
Alpha, fission fragments, heavy nuclei	-	20

conducted by comparing the relative biological effectiveness (RBE) of different radiations in inducing cancer in an organ. According to experiment, values are significant only at low doses of radiation, which lead to stochastic effects[7].

Effective dose Epidemiological studies have shown that occurrence of cancers depends on the intrinsic sensitivity of each organ. So, each tissue or organ is associated with a weighting factor w_T which takes into account the probability of radiation-induced stochastic effects in the organ or tissue. This factor allows calculation of the effective dose E, a hypothetical dose which, administered uniformly to the entire body, would cause the same late-onset damage as all the doses received by the same individual on the different organs separately at different times, so it makes possible to estimate the risk for humans from a measurable quantity, the absorbed dose. In case of partial exposure of several organs, the effective dose is:

$$E = \sum_T w_T H_T \tag{2.4}$$

in Sievert unit. Values of some tissue are shown in Table 2.2.

Kerma In order to give a conceptionally clear description of what could happen when a neutral radiation (gamma or neutron) interacts with matter a special quantity is defined which comprises the total kinetic energy transferred to secondary particles per mass element. It is called *KERMA* (kinetic energy released per mass) and is measured in Jkg^{-1}. Each mass element is, of course, not isolated but part of its environment to which it loses particles and from where others enter. Secondary particle equilibrium is obtained if every particle leaving the element is compensated by an entering one of exactly the same type and energy. A necessary - but not sufficient - condition for this to occur is that the mass element is part of a homogeneous medium at a depth that is larger than the range of the most energetic particle. It is immediately clear that this can never happen at or near surfaces. Kerma and dose, however, are even in the case of secondary particle equilibrium not generally identical; this is only if *bremsstrahlung* losses are negligible. At the surface there is the largest difference because of the lack of equilibrium, see Figure 2.1 . They

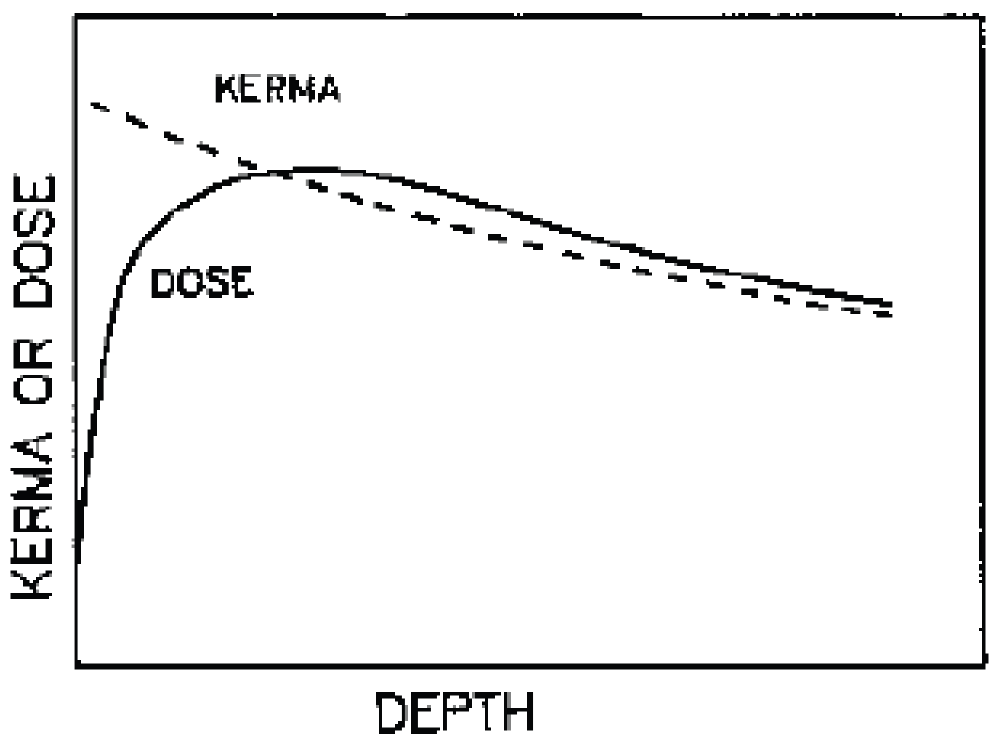

Figure 2.1: *Difference between Kerma and Dose[9].*

become smaller with greater depths where they then are only due to bremsstrahlung.

ii. LET

The energy deposition in an exposed body is mediated almost eexclusively by charged particles. These ionize atoms on their way loosing parts of their energy in successive steps until they reach the end of their range. Depending on the type of particle, ionizations are more or less closely spaced, which is very important if one considers energy deposition onto very small sites. This situation may be described by the energy loss of a particle per distance travelled. Corresponding quantity is called *linear energy transfer* (LET) which is defined as the amount of *locally* absorbed energy per unit length. Locally means that energy absorbed is a fraction of total energy transferred in which there is a limit on energy deposited in a local site of medium; 100 eV has been widely accepted, which corresponds to an electron range of about 5 nm. LET without restriction on energy is equal to the stopping power:

$$LET_{\infty} = -\frac{dE}{dx} \tag{2.5}$$

This physical quantity is the first step to evaluate the biological damage. Since the LET value is proportional to the energy transferred by the radiation, it is related to the ionization density and to biological effects[10].

iii. RBE

For a simple quantitative description of the dependence of radiation quality, the concept of "relative biological effectiveness" (RBE) has been introduced. This parameter is defined as the ratio of doses which yield the same effect if one compares the test radiation with 250 kV X rays or $cobalto - 60$ gamma rays:

$$RBE = \frac{D(250kVXrays)}{D(test \quad radiation)} \tag{2.6}$$

RBE shows a dependence on LET. In fact, to a higher LET corresponds a greater number of ionizations along the path inducing a more important damage and this, as mentioned before, is associated with a high RBE value. This parameter is plotted versus LET_{∞} in Figure 2.2. The overall behaviour is: RBE increases with LET, passes a maximum around 200 keV/μm from where it declines. This last part has been related to the *overkill* situation where more energy is deposited per particle traversal than actually required for the inactivation of cells. The initial rise is commonly interpreted to mean that essential lesions, like double strand breaks of DNA, are formed with greater efficiency with higher ionization density[12].

Proton RBE Proton effects on tissue is still an open question of physics and biology. Since initial studies, dose specification in proton radiation therapy was fixed to a constant RBE value, while after a lot of studies is obtained that other variables influence RBE, like position in the Bragg Peak, initial beam energy and tissue. The physics process that could influence the most is fragmentation. When the incident beam interacts with the target in an inelastic collision, a non-negligible amount of fragments is produced outside of the planned area. They can derive from the target or from the projectile disintegration, according to the different conditions in which the fragmentation takes places (see Section

IV). In case of a proton beam, target nuclei could fragment and change RBE, as is shown in Figure 2.3.

iv. DNA damage

In DNA, radiation alterations may be classified in *single strand breaks* and *double strand breaks*:

Single strand breaks this is a scission in one chain, while the other remains unaffected.

Double strand breaks this is usually caused by a single energy deposition event or by the interaction of two SSB formed individually in close proximity.

DBSs are more likely to happen when heavy ions interact with matter, because ionized electrons, created in this process, have a mean free path of the order of few nanometers. This process cause permanent damage to the DNA because both chains are broken on the same spot and it is not possible to retrieve the genetic information. This means that heavy ions have a higher damage capability than photons, which, instead, are involved in the SSB (Figure 2.4). An indirect effect on DNA is the ionization of molecules such as H_2O, giving H^+ or OH^-, that can attack other molecules. For example OH^- attacks the DNA molecules, which have very soft electrical bounds. Damage depends on the kind of alteration of the DNA, causing cancer or long term genetic mutations.

II. Application in cancer therapy

Treatment with radiation plays an important role. The most used radiations are photons, because of the advanced knowledge on how to produce them and modulate in energy.

i. Radiotherapy

Radiotherapy uses photon beams with energy between 6 MeV and 25 MeV like in Figure 2.5 and also electron beams in combination with the other therapy. Initially X rays were mainly used but are now substituted with γ rays that are far more energetic. The photon beam used in radiotherapy shows a characteristic depth-dose profile that follows the absorption law and, therefore, displays a steep exponential decrease of dose with depth, like in Figure fig:radio. It presents a maximum between 1 cm and 5 cm. The problem of radiotherapy is that localization of the maximum dose release that, even at greater energies, remains near the surface of the target while tumors are often found in depth[15].

ii. Hadrontherapy

Hadrontherapy is an oncological technique that uses hadrons beams, not photon beams, accelerated by cyclotrons or synchrotrons to energies from 50 MeV/u to 400 MeV/u (per nucleon). The strenght of hadrontherapy lies in the unique physical and radiobiological

properties of these particles; they can penetrate the tissues with little diffusion and deposit the maximum energy just before stopping. This allows a precise definition of the specific region to be irradiated.
The idea of using protons for cancer treatment was first proposed in 1946 by the physicist Robert Wilson, who later became the founder and first director of the Fermi National Accelerator Laboratory. The first patients were treated in the 1950s in nuclear physics facilities by means of non-dedicated accelerators. In the late 1970s improvements in accelerator technology, coupled with advances in medical imaging and computing, made proton therapy a viable option for routine medical applications.
In Italy there are three facilities: CATANA Proton Therapy beam line in Catania, with a energy up to 60 MeV, CNAO in Pavia with a energy up to 250 MeV for protons and 400 Mev/u for carbon beam and APSS in Trento with energy between 60 and 230 MeV for protons.
This therapy shows a distinctive dose release profile that makes it more effective than radiotherapy; it is characterized by a distinct narrow peak, *Bragg peak*, at the end of the particles path with a sharp fall-off at the distal edge (Figure 2.6). Distribution shows a small entrance dose, a well-defined range and a small lateral beam spread.
Despite being both employed in the therapy, protons and carbon ions have different features for what concerns the biological effects and the dose tail. At same absorbed dose, protons have a similar biological response to photons, while heavy ions show higher effectiveness. In addition heavy ions exhibit a distinctive dose tail beyond the Bragg peak, caused by secondary fragments produced in nuclear reactions along the stopping path of ions[16].

iii. Boron Neutron Capture Therapy

The *Boron Neutron Capture therapy* (BNCT) is a sperimental radiotherapy based on irradiation with termal neutrons on a cancer region enriched with ^{10}B. Neutron capture reaction with boron lead to formation of ^{11}B* excited which decays suddendly into two products with an high LET very energetic, an alpha particle and a ^{7}Li by means of two nuclear reactions:

$$^{10}\mathrm{B} + n \to {}^{11}\mathrm{B}^* \to \alpha + {}^{7}\mathrm{Li}^* \to \alpha + {}^{7}\mathrm{Li} + \gamma(94\%) \tag{2.7}$$

where products energies are:

- $E_{alpha} \simeq 1.47 MeV$
- $E_{^7\mathrm{Li}} \simeq 0.84 MeV$
- $E_{gamma} \simeq 0.48 MeV$

and

$$^{10}\mathrm{B} + n \to {}^{11}\mathrm{B}^* \to \alpha + {}^{7}\mathrm{Li}(6\%) \tag{2.8}$$

where products energies are:

- $E_{alpha} \simeq 1.78 MeV$

- $E_{^7\mathrm{Li}} \simeq 1.01 MeV$

The products ionize matter near capture region, into $5\mu m$ for α and $9\mu m$ for ^{7}Li, a not much small volume of a cell ($\sim 10\mu m$). The advantages to use boron are that it is not a radioactive isotope, it is available in nature and very easy to link into molecules, and than products of reaction have high values of LET and small ranges, so they release energy inside cancer cells. Although cross section of capture for other nuclei are much smaller than boron ($\sigma_{^{10}\mathrm{B}} \simeq 3.84 \cdot 10^3 barn$), two of these, hydrogen and nitrogen ($\sigma_{^1\mathrm{H}} \simeq 0.33 barn, \sigma_{^{14}\mathrm{N}} \simeq 1.75 barn$), are more concentrated than boron and absorb neutron. So the treatment is optimazed in order to have a letal effect on cancer by using $\sim 10^9$ atoms of ^{10}B for a cell with a thermal neutron fluence of $10^{10} - 10^{12} \frac{neutrons}{cm^2}$ and a boron concentration inside tumor region of $35 - 50\frac{\mu g}{g}$. Disadvantages of this therapy are linked to boron distribution, because an ideal drug should fill tumor region and stay outside blood-brain barrier with a small concentration into blood. Another problem is how to control neutron flux from a nuclear reactor and focus it on the tumor region[17]. In figure 2.7 is shown main reaction for BNCT.

iv. Proton Boron Capture Therapy

Besides the advantages of using a neutron-free nuclear fusion reaction, the relevance of this method stems from the fact that the $p + {}^{11}\mathrm{B} \to 3\alpha$ cross section becomes significantly high at relatively low incident proton energy, for example around the Bragg peak region. As shown in Figure 2.8 a proton beam as conventionally used in protontherapy is drastically slowed down across the tumor (the Bragg peak region). Thus, most of its energy (dose) is delivered to the tumor cells. Under the assumption that a given concentration of ^{11}B nuclei is present preferentially, but not exclusively, in the tumor, the arrival of slow protons could trigger a series of fusion events generating several alpha particles that are localized in the tumor region. In fact, most of the alpha particles generated in the proton-boron reaction have an average range in water of less than 30 μm, thus comparable with the typical cell size. Hence, even if such particles are mainly produced outside the cell cytoplasm due to sub-optimal boron uptake, the probability that they would reach the nucleus and damage the DNA remains very high. Moreover, even if a non-negligible concentration of ^{11}B nuclei is present in the healty tissues surrounding the tumor, the number of generated alpha particles, would be relatively low, or completely absent, due the non-favourable incident proton energy spectrum away from the tumor region. This would lead to a more biologically effective particle dose localization, higher than the one currently achievable with conventional protontherapy, thus to a more efficient treatment in terms of an *enhancement* in cancer cell lethality, especially because of the clustered nature of the DNA damage, caused by the high-LET alpha particles emitted in the tumor region. Hence, protontherapy could acquire the benefits of an enhanced efficiency in cancer cell killing, moving close to ^{12}C ion hadrontherapy but without the above-mentioned complications of the latter.
Ballistic advantage granted by the inverted dose-depth profile of charged particles is

such that in protontherapy most of the dose is released mainly in the tumor region (left panel). If cancer cells are loaded with ^{11}B-delivering agents, unrepairable DNA clustered lesions will be also produced by High-LET alpha particles. For a given clinical case, such higher *dose modifying factor* can potentially allow to reduce the overall dose delivered to the patient compared to a standard treatment without the presence of ^{11}B-delivering agents in the tumor.

Table 2.2: *Weighting factor w_T for different types of tissue [8].*

Tissue	Values of w_T
Gonads	0.08
Bone marrow	0.12
Colon	0.12
Lung	0.12
Stomach	0.12
Bladder	0.04
Breast	0.12
Live	0.04
Oesophagus	0.12
Thyroid	0.04
Skin	0.01
Bone surface	0.01
Brain	0.01
Salivar glands	0.01
Other tissues or organs	0.12

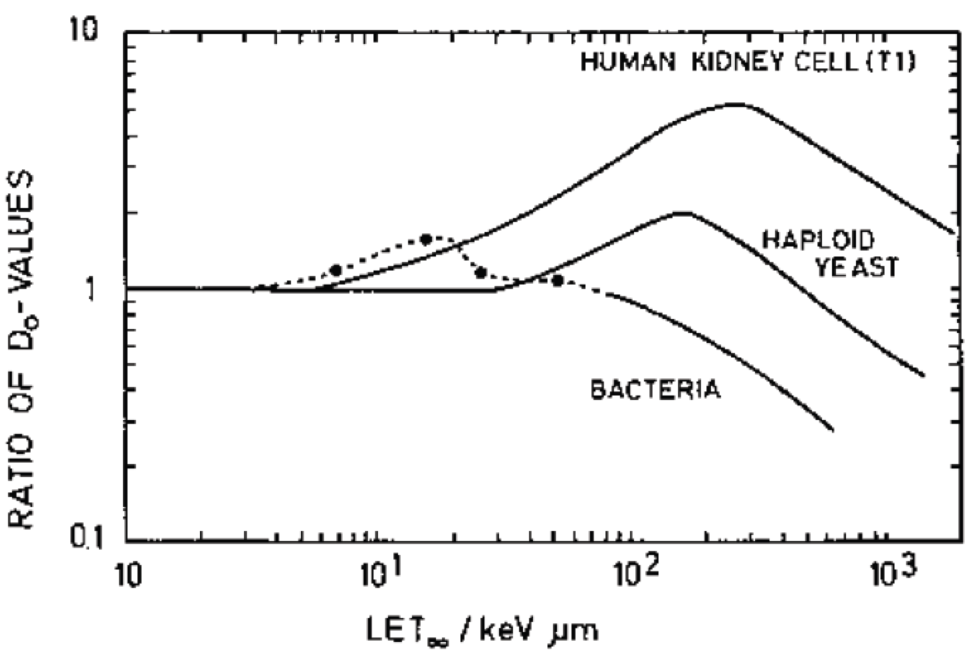

Figure 2.2: *Relative radiation sensitivity of various cell types as a function of LET[11].*

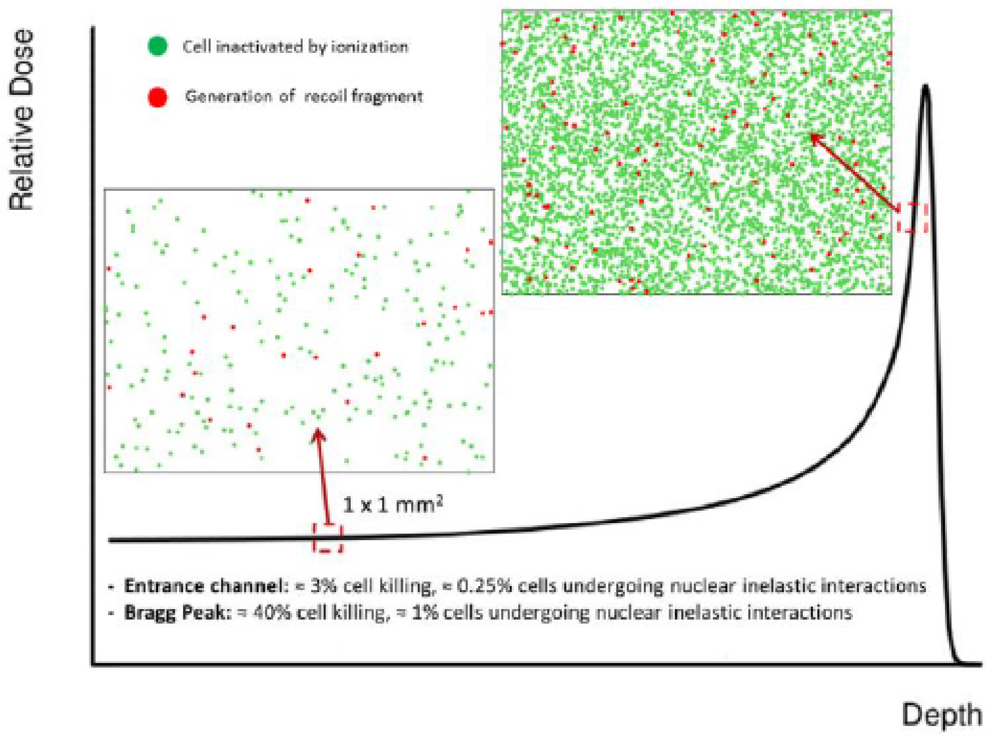

Figure 2.3: *Cell inactivated by ionization (green) and target fragmentation (red) in tissue of 1 mm^2 [13].*

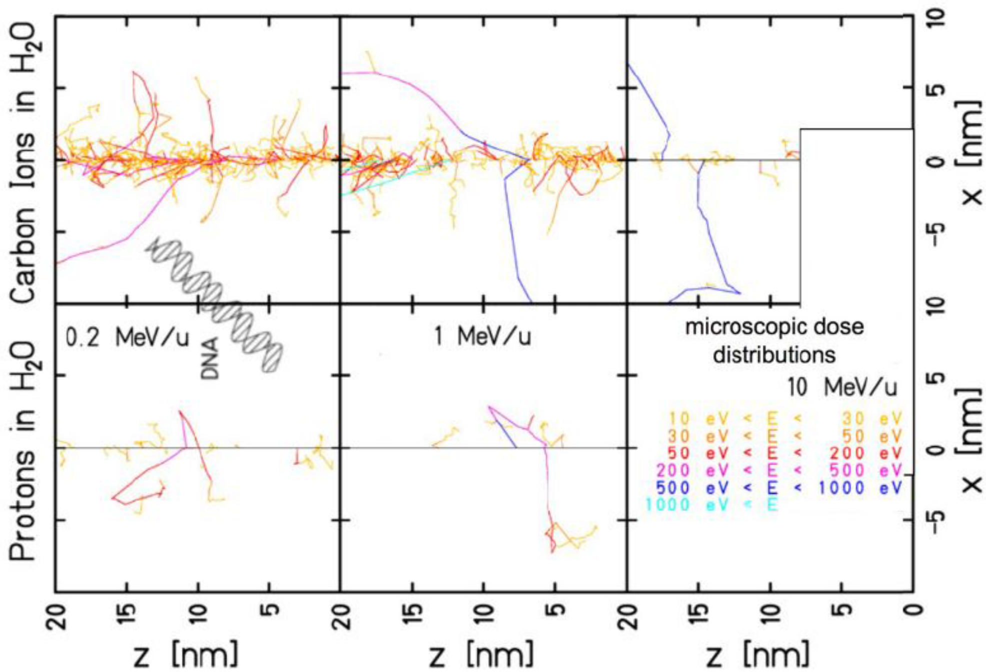

Figure 2.4: *Simulation with the Montecarlo code of the interaction between carbon ions and protons with matter in proximity of the range end, where particles slow down and their energy is of few MeV per nucleon. The graph shows the tracks of single secondary electrons. Carbon ions produce more tracks than protons, increasing the probability of having direct damage to the DNA[14].*

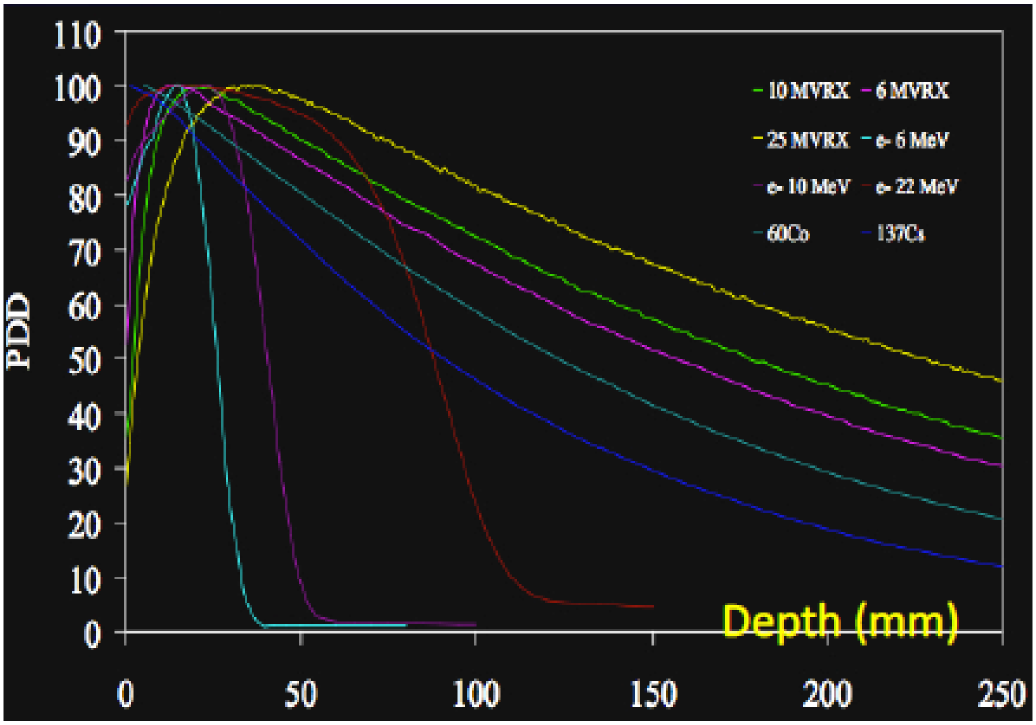

Figure 2.5: *Percent Depth Dose of photons beams for several photon energy.*

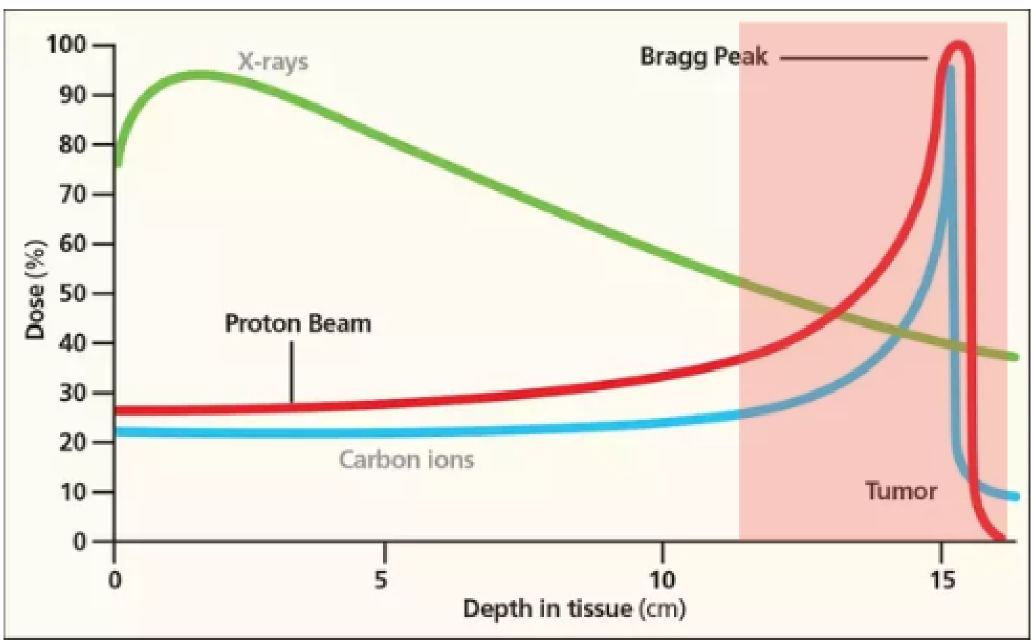

Figure 2.6: *Comparison of depth-dose profiles in water of photons, protons and high-energy carbon ions.*

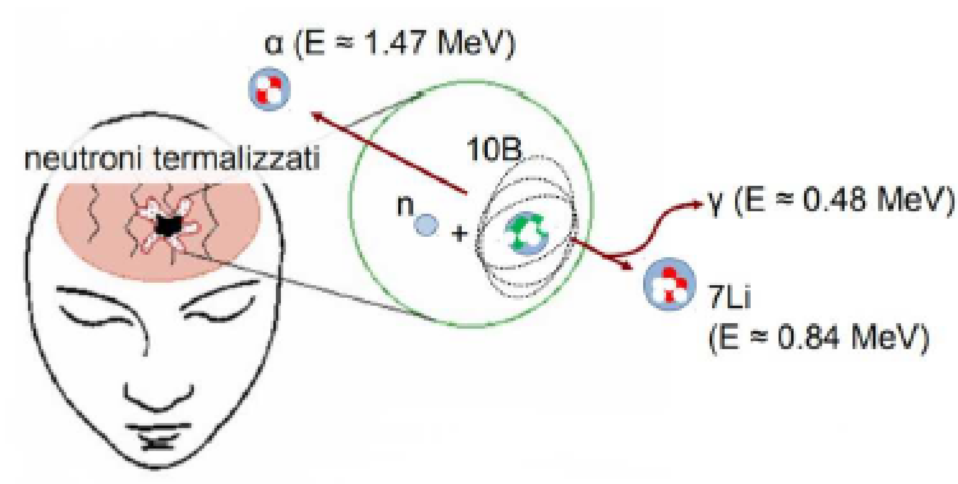

Figure 2.7: *Rappresentation of BNCT.*

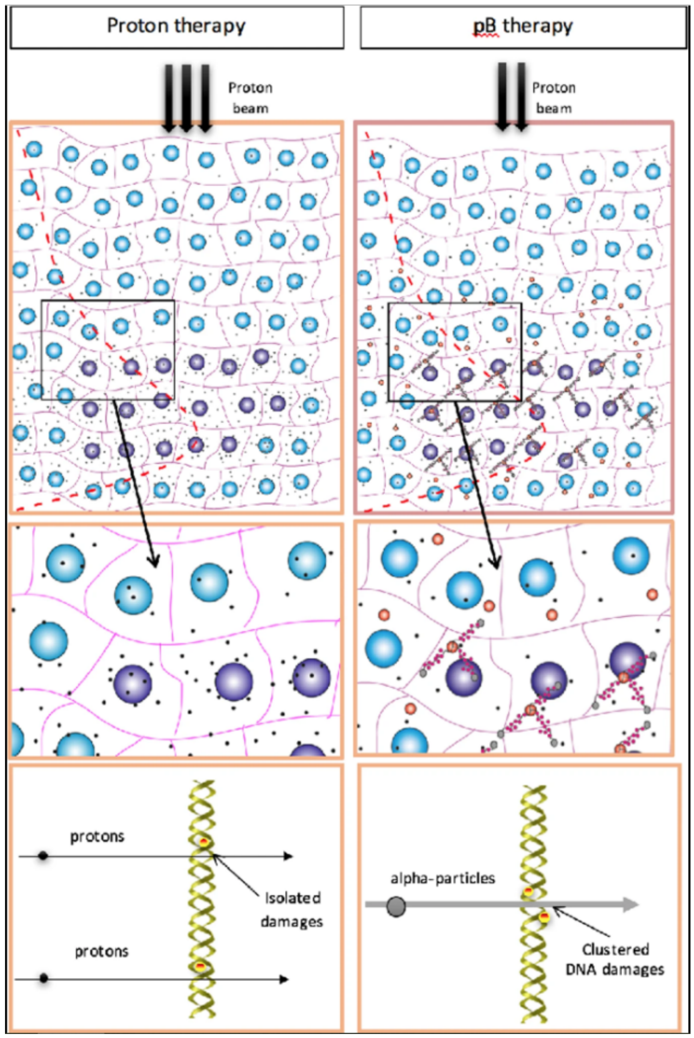

Figure 2.8: *Rappresentation of PBCT: left panel shows conventional protontherapy, in which incident proton beam mainly results in isolated, repairable DNA breaks; right panel shows how proton-boron extremely localized alpha emission in the Bragg peak causes irreparable clustered DNA damage [18].*

Chapter 3

ERHA project

The technology of proton's production and acceleration is a part of several research projects. The ERHA is focused on the realisation of an hadrontherapy center, in which the accelerator is linear and compact. The main reason is the possibility to install this machine inside a hospital centre. In order to build this type of accelerator, there are some specification to be setted: type of particle, for example protons, but also ions, energy of the beam, number of particle accelerated in a second, and repetition frequency. In all the situation accelerator is placed inside a 100 m^2 room, with all the instrumentation.

I. LINAC accelerator

LINAC is a type of particle accelerator that is based on the multiple acceleration method: in the gap between two beam pipe tubes there is an electric potential difference $V = V_0 \cos \omega t$ that estabilishes a variable electric field, so the particle is accelerated if it crosses gaps in a time syncronous with field, so for every wave period. After a period, particle gains velocity, so the distance between gaps has to increase, roughly as $L = vT/2$, where v is the particle velocity[19]. ERHA project is based on some working conditions which are shown in Table 3.1 Some characteristics are described in a simplified way:

- ***Peak current***: Mean current, that correspond to mean number of particles for a treatment, is 1 ÷ 10 nA. A pulsed beam with a repetition rate of 10 Hz and with a pulse width of $4\mu s$, has a peak current of:

$$\frac{1nA}{10Hz \cdot 4\mu s} = 25\mu A \tag{3.1}$$

- ***Number of protons per injector pulse***: Peak current of the injector is 1.0 mA. If pulse width is $4\mu s$, number of protons is:

$$\frac{1mA * 4\mu s}{1,6 * 10^{-19}C} = 2,5 * 10^{10} protons \tag{3.2}$$

per pulse, while per second are $2,5 * 10^{10} * 10Hz = 2,5 * 10^{11} p/s$

- ***Approssimative lenght***: A general solution for lenght of an accelerator is:

$$L = \frac{T_{final}}{qE_{0T}\cos\phi_s} \tag{3.3}$$

  where q is the charge of proton, E_{0T} is the effective accelerating field of $10 \div 15$ MV/m and ϕ_s is the sincronous phase of $-20°$. Typical lenght is 20 meters.
- ***RF Power***: In order to accelerat it needs a radio frequency power at peak:

$$P_p = \frac{(E_{0T})^2}{Z_{eff}} L \tag{3.4}$$

  where Z_{eff} is the shunt resistance for unit lenght. So P_p is some MW. In order to obtain mean power, P_p must be multiplied by duty cicle.
- ***Accelerator modules***: Protons are initially accelerated by the Radiofrequency Quadrupole (RFQ) to 4 MeV. For energies from 5 MeV and 100 MeV are usually used Drift Tube Linac modules (DTL), working at less than 500 MHz. After these energies are used Side Couple Linac modules (SCL), working at 3 GHz.
- ***Frequency***: In general accelerators work at high frequency, because it reduces dimensions in which is located, but also RF power, and break-down limit is higher, as obtained by Kilpatrick law $E_{break} \propto f^{0.4}$.

i. Injector

The first module for injecting protons into accelerating pipe is the Accsys-Hitachi PL-7, a commercial model made of a duoplasmatron source of 30 keV and a RFQ of 4 MeV operating at a 428,27 MHz, seventh subarmonic of 2997,92 MHz, the operating frequency of the LINAC. The power supply consists of 15 planar triods, EIMAC tubes (CPI-YU176A), 1 for pre-amplificator, 2 for second pre-amplificator and 12 for the final amplificator[20].

Duoplasmatron Proton source is a ions source supplied with gas, tipically hydrogen. A schematic figure is 3.1. An electric arc, with low pressure, between anode and cathode is produced. On cathode there is a thin wire, tipically made of tungsten, in which flows current which produces electrons by thermoionic effect. On cathode flows gas which is ionized by electric arc and produced plasma, which is confined by an intermediate electrode conic shaped and stay in little region A. Although plasma is globally neutral, particles inside are charged and let it expand. In order to confine plasma, it is used a magnetic field that mantain a pressure on plasma. On electrode there is a little hole that made particle flowing to anode. Between intermediate electrode and anode there is an other electrode that is positive and pick electrons. In region B protons are accelerated towards estraction electrode and goes into a region with a lense and a diafram, which are used to focus the beam[21].

RFQ The RFQ is a linear accelerator that accelerates, focuses and make bunches of protons with a radiofrequency field. It is the best way to accelerate ions at low energies, and it is simple to design and build, and accelerate intense beams with 90% of efficiency. RFQ structure consists of a radiofrequency cavity made by four electrodes (Figure 3.2) formed and shaped properly in order to obtain right acceleration and focusing. Protons are focused polarizing electrodes simmetrically, in order to create a quadrupole field. Protons are simultaneously accelerated by the same field by shaping electrodes longitudinally like a sinusoid function, in Figure 3.3. Peaks of electrodes are separated by a lenght of $\beta\lambda$, in order to shape bunches of protons[22].

ii. SCDTL

The end of injector is matched with an ERHA project developed module which is based on a Drift Tube Linac in which accelerating tanks are alternated to pairing tanks situated off axis, in a way that is possible to pair fields inside tanks. Between two accelerating tanks there are quadrupole permanent magnets in a FODO cell, that focus the beam. This is an innovative module, because it is designed to be modular and easy to set, see Figure 3.4 that shows the Module 0, an SCDTL (*Side Coupled Drift Tube Linac*). A schematic visualization of how SCDTL works is shown in Figure 3.5 where are visible tanks that consist of a series of drift tube cell, from 3 to 7 $\beta\lambda$ long, while the space between tanks is used to focus the beam by means of a quadrupole (PMQ) which is 3 cm long and with a diameter 2 cm long. The Drift Tube Linac is made by an array of resonant cavities, at RF frequency. The syncronization of the particle motion with electromagnetic oscillations is made by changing lenght of drift tubes between gaps. The lenght increasing of the cavities increases also the capacity, and decreases resonance frequency. In order to keep frequency to a constant value, gap geometry in tanks have to be shaped reducing drift tube diameters progressively, like in Figure 3.6. Design of cavities creates also the right distribution of the field and estabilishes the parassite currents on the separation surface between two cavities. An electromagnetic wave that is resonant in a cavity is depicted by a *mode*, that could be *Transverse Electric* (TE) if the components of the field that is directed along propagation direction is $\vec{B}$, otherwise *Transverse Magnetic* (TM), in order to satisfy contour conditions where field is null. Electrif field distribution and parassite currents for the TM_{010} are in a shape that leaves field distribution equal when the separtion surfaces are eliminated. In order to hold drift tubes there are sustain bars in a position in which field is radial and is not perturbated.
A big problem on DTL is to sustain the mode TM_{010}. In a complex structure is a very common problem that different resonant modes are generated which could give rise to significatively beam losses. In order to resolve it, some tuning elements are inserted in the structure, so called *post couplers*, orthogonal to sustain bars[23].

iii. CCL

CCL module is developed at Los Alamos Laboratory. The structure is similar to SCDTL, it is compact, every cell is $\beta\lambda/2$ long assembled in tanks of limited lenght (in a tank there are 16 cells). Between tanks are placed quarupoles, every tank is made by n accelerating cell and n-1 pairing cell in $\pi/2$ field mode, while a bridge coupler pairs the two tanks (see Figure 3.8). An efficient gain of the structure for small energies (bigger than 25-30 MeV) is possible with a "PALME" structure, much more complex to design, as shown in Figure 3.7 High values of the field E_0 can be set to obtain smaller accelerator, but RF power increases as E_0^2. Kilpatrick's limit of break-down E_k is of $47MV/m$ for a $3GHz$ frequency, with a superficial field $E_s \leq 2E_k$. For a geometry reason, $E_s \simeq 6E_0$, for a value of $15MV/m$. When the field is fixed, an alghoritm SUPERFISH calculates geometry of the cells. These are assembled in tank by an other program, which calculates energy gain for every tank and focusing force of the quadrupole. At the end a simulation program PARMILA, in which are simulated hundreds of particles, can simulate trajectory inside beam-pipe. A CCL, *coupled cavity linac*, is made by resonant cavities coupled by a hole in the structure. They are used to accelerate protons in a range $0.4 < \beta < 1.0$. Single cavity is called cell, in an excited mode very similar to TM_{010}. Coupling is generated by a hole like in Figure 3.9 where in (a) is shown an electric or capacitive coupling due to a higher electric field than magnetic component, while in (b) is shown a magnetic or inductive coupling, due to a higher magnetic field[25].

Table 3.1: *Initial LINAC specifications.*

Proton Spot Size	2 mm
RF frequency	3 GHz
Mean beam current	1 $\div$ 10 nA
Peak current (50 Hz)	0.7 $\div$ 33 μA
Injector peak current	1.2 mA
Repetition rate	20 $\div$ 200 Hz
Pulse duration	2 -20 μs
Energy precision	$\pm$ 0.2 MeV
Water range	0.1 $\div$ 3.3 g/cm^2
Distal decrease dose (80% - 20%)	< 2 mm
Normalized transverse emittance	< 2 π mm mrad
Duty cicle	0.01%

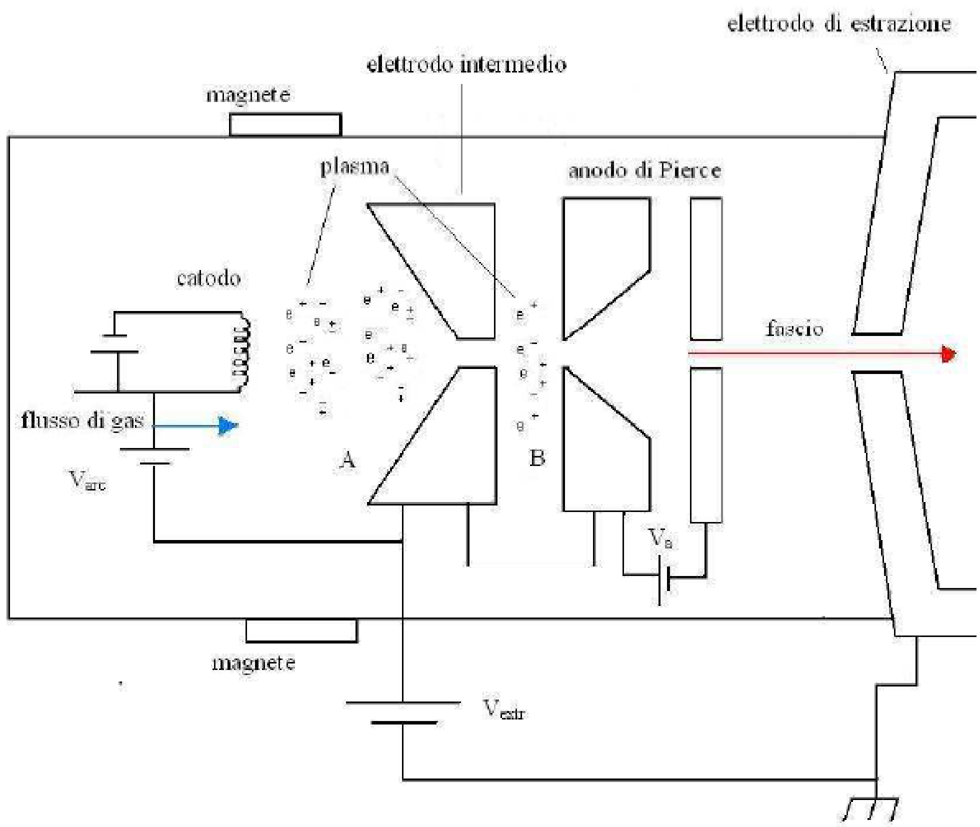

Figure 3.1: *Proton source* duoplasmatron.

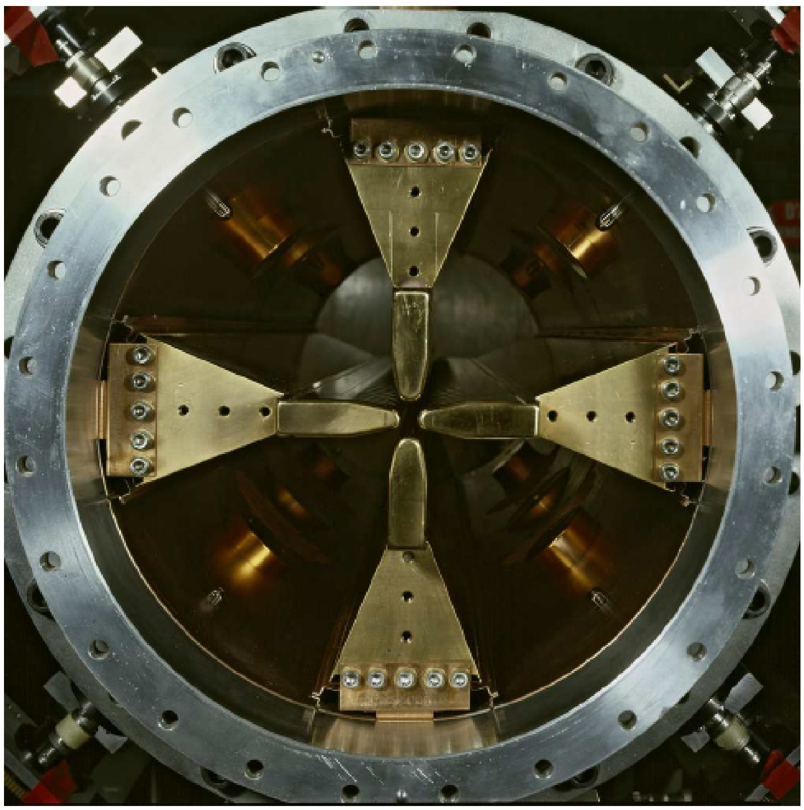

Figure 3.2: *RFQ structure.*

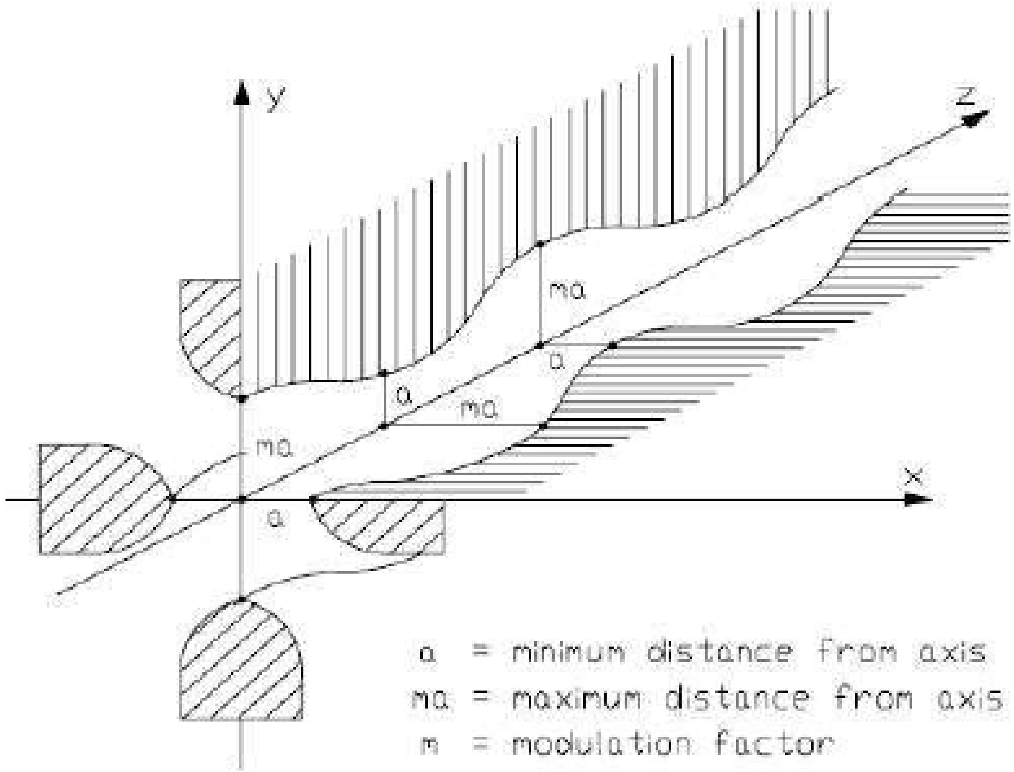

Figure 3.3: *Longitudianl modulation of electrodes.*

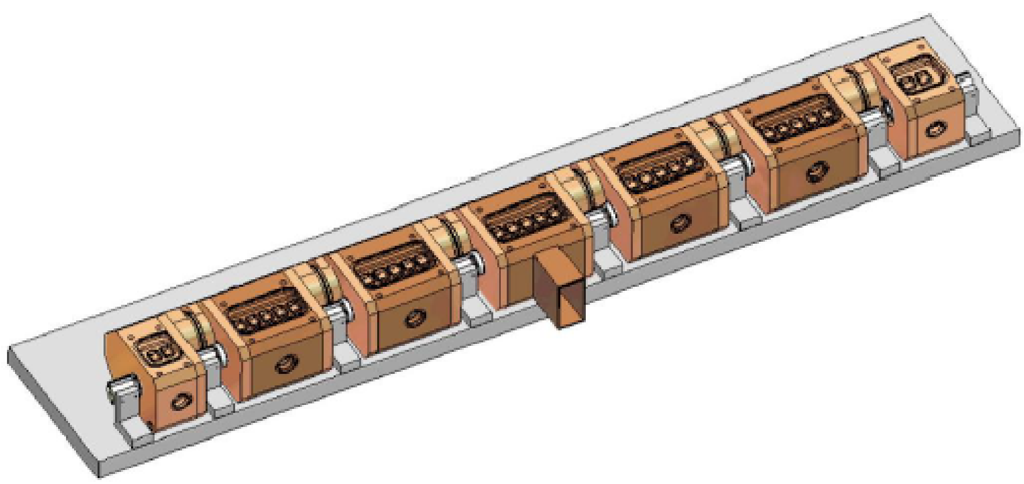

Figure 3.4: *Module 0, SCDTL.*

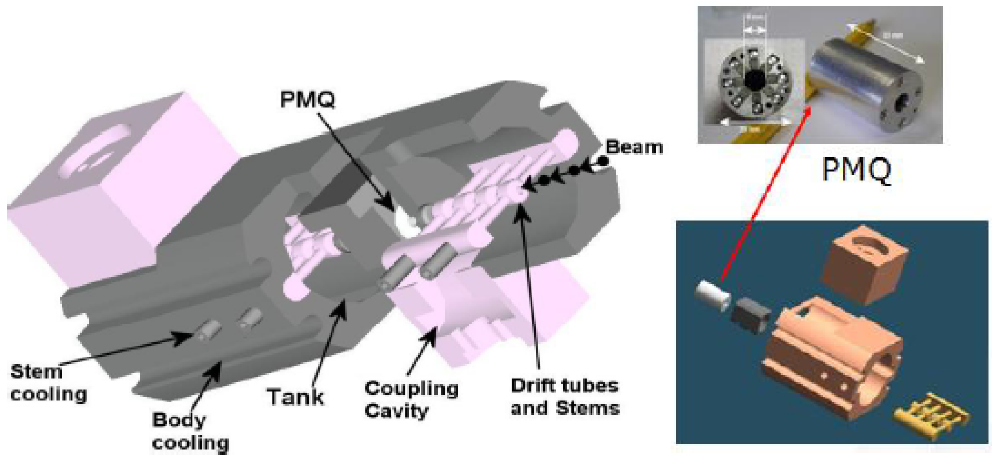

Figure 3.5: *Interior part of tanks and PMQ.*

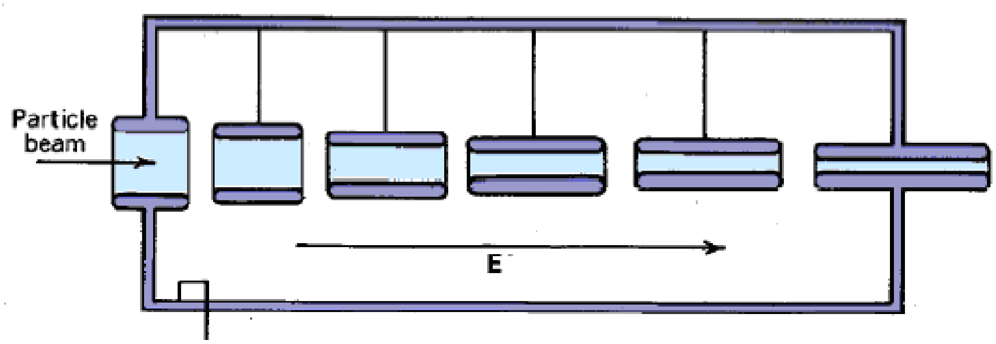

Figure 3.6: *Progressive shaped tank.*

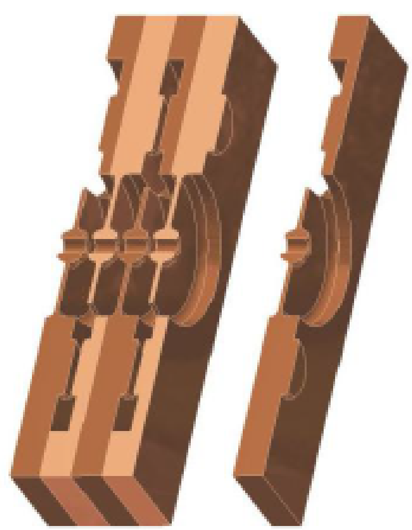

Figure 3.7: *PALME structure of CCL.*

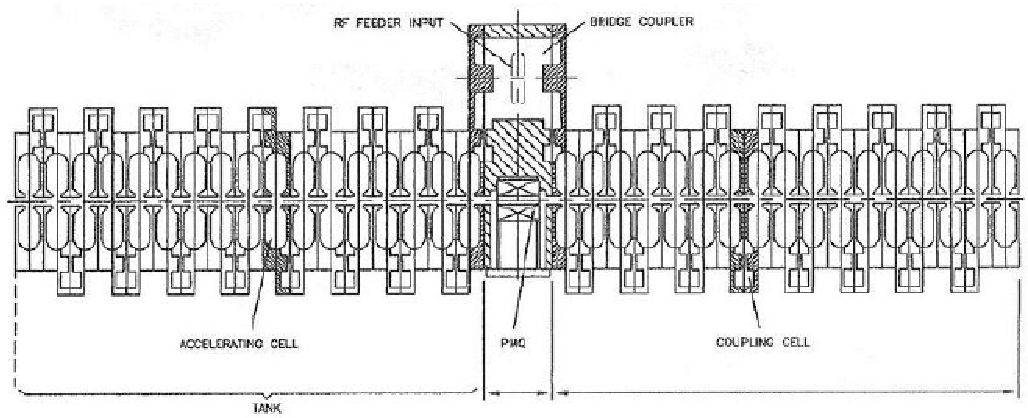

Figure 3.8: *[24]Assembled tanks in CCL.*

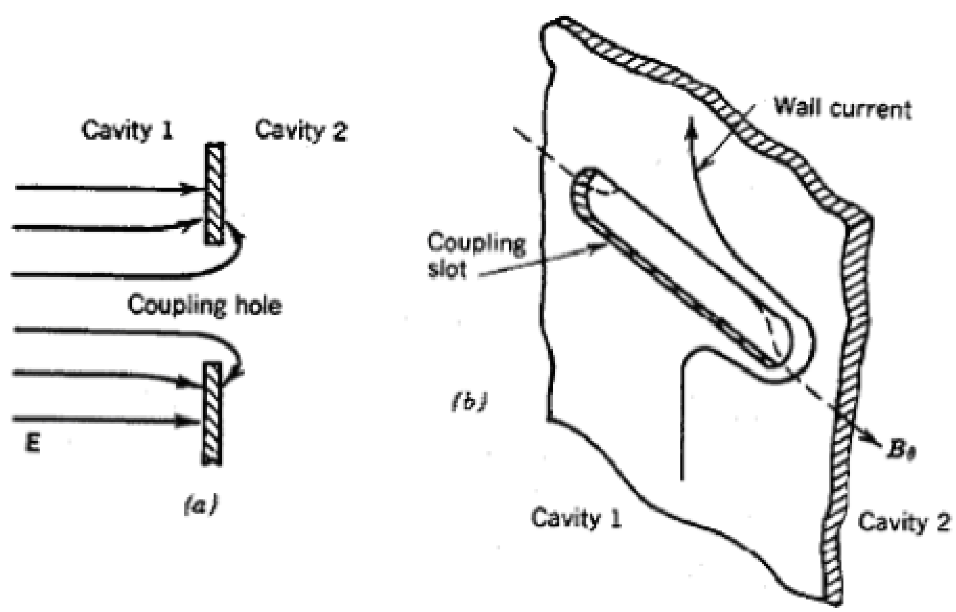

Figure 3.9: *Coupling between cavities,* electric *to left,* magnetic *to right.*

Chapter 4

Radiation shielding

The radiological protection aspects of an accelerators are extremely important in the design of these machines. There are many parameters by which particle accelerators may be classified. For example, they may be classified in terms of the acceleration technology, such as power source or acceleration path geometry. Also they may be classified by their application, by types of particle, maximal energy, maximal intensity and duty factor of the beams. In this chapter is described the main radiation source in the bunker and the protection aspects.

I. Prompt radiation

A basic knowledge of the nuclear reaction mechanisms that happens in the energy range of 1 - 10 MeV is required in order to understand shielding of an accelerator. When a proton hits matter it could undergo a nuclear reaction and form a compound state. The process could give rise to an inelastic scattering, in which the compound could change number of protons and neutrons but A doesn't change, a so called *charge-exchange reaction*. When A changes, there is a transfer reaction (*stripping* or *pick-up*) or a *knockout reaction*. The angular distribution of the particle is characteristically anisotropic, peaking in the forward direction. In this situation, each nucleon will undergo further collisions, gradually spreading its excitation energy over the whole nucleus. For a certain time (during pre-equilibrium phase) nuclear state will become increasingly complex, but after a certain relaxation time, statistical equilibrium will be reached. A fraction of the mixture of nuclear states consists of configurations in which energy is concentrated on one nucleon so that it may escape from the nucleus. Similarly, kinetic energy may be concentrated on groups of particles and lead to the emission of α particles, tritons, deuteron etc. This is similar to evaporation and may be charachterized by a nuclear temperature $\Theta \approx 2 - 8$ MeV, so that spectrum of the emitted neutrons may be described by the following Maxwellian distribution:

$$dN(E_n) \propto \frac{E_n}{\Theta^2} \exp\left(\frac{-E_n}{\Theta}\right) dE_n \tag{4.1}$$

Compound reactions may occur during the *pre-equilibrium* phase, before statistical equilibrium is achieved. In such cases the angle of emission may still be correlated with the direction of the incident particle. On the other hand, once statistical equilibrium was obtained, the emitted particles have no memory of incident particle and angular distribution is isotropic. The particles emitted could participate in similar reactions resulting in an intranuclear cascade, which develops through interaction of individual nucleons inside nucleus.
Prompt radiation field near interaction point of accelerated protons with matter is complex and becomes more complex as energy is increased. The field consists of a mixture of *charged and neutral particles and photons*. Several simulation codes are available which include all the interaction described above and which allow estimates of the radiation field, such as GEANT4 and Fluka.
In order to attenuate prompt field, in particular as seen above neutron field, two criteria must be satisfied: interpose sufficient mass between the source and the field point and attenuate effectively neutrons of all energies. First criterion is easily obtained by dense material of high atomic mass, while second is most easily met by hydrogen, which attenuates neutrons of all energies via elastic scattering. The two criteria are met by concrete because it contains hydrogen in water form.
In Figure 4.1 is shown variation of the mass attenuation lenght, $\rho\lambda$ for monoenergetic

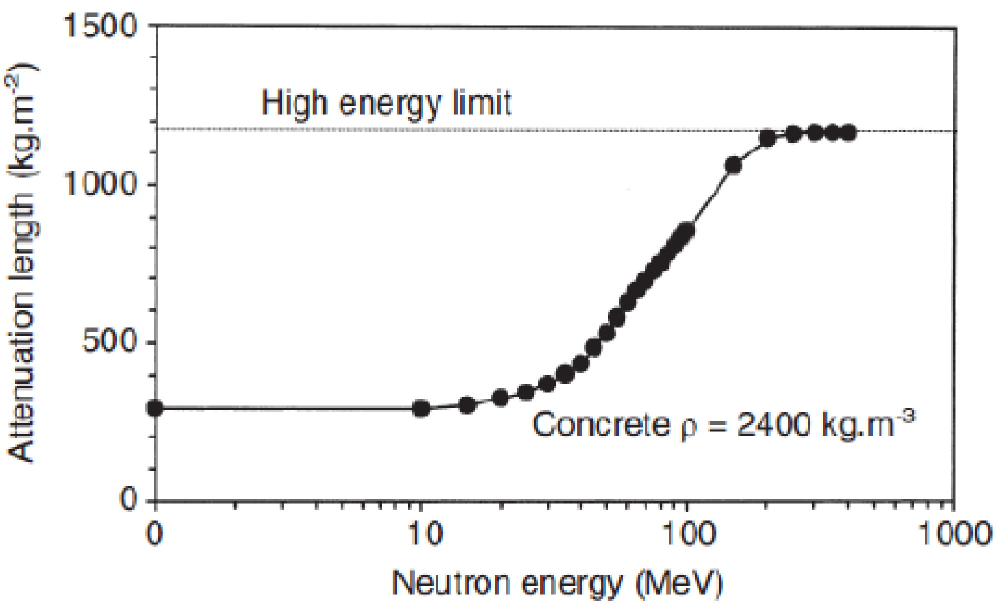

Figure 4.1: *Attenuation lenght $\rho\lambda$ of monoenergetic neutrons in concrete of $\rho = 2400$ kg/m^3 in function of energy.*

neutrons in concrete as function of energy. Below 20 MeV, $\rho\lambda$ is 200 kg/m^2. Above this energy neutron cross section changes from a interaction with target nuclei as a whole, so an elastic scattering, to an interaction with nucleons.

The lower energy neutrons and charged particles are regenerated at all depths in the shield by the inelastic interactions of the neutrons with the shielding material. In other words, at any field point outside the shielding, the highest energy neutrons will be those that have come directly from the source without interaction, or that have undergone only elastic scattering or direct inelastic scattering with little loss of energy and only small angular deflection. Any low energy neutrons and charged particles detected outside the shielding will have been generated by the intranuclear cascade near the outer surface of the shield. In Figure 4.2 the neutron yield is normalized per interacting proton and has a

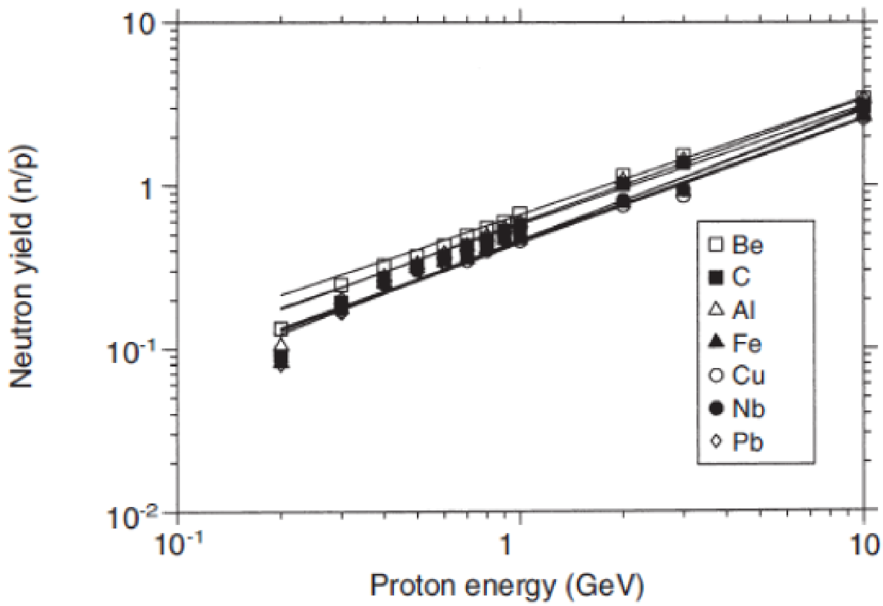

Figure 4.2: *The yield of neutrons with energy $E_n > 100$ MeV per interacting proton in stopping targets of a number of materials as a function of proton energy. The points are the results of calculations with the FLUKA Monte Carlo code and the lines are best fits to these points to the relation $n(E_p) = n_0 E_p^m$, [26].*

dependence on the proton energy of the form:

$$n(E_p) = n_0 E_p^m \tag{4.2}$$

i. Operational quantities

Body-related protection quantities, effective dose, equivalent dose, have drawback of not being amenable to measurement. To meet the requirements of the organizations charged with monitoring workforce exposures, the concept of *operational quantities* was introduced, used to arrive at reasonable assessments of the protection quantities. These operational quantities involve the following characteristic features: they are based on dose at depths of 10 mm and 0,07 mm, respectively, as measured in the ICRU sphere, or in the human body. The ICRU sphere is a reference sphere, 30 cm in diameter, made of tissue-equivalent

material with a density of 1 gcm^{-3}. They can be measured at the workplace, by means of external radiation detectors (rate meters, dosimeters), and may be used for individual ambient monitoring purposes at workstations. They serve as estimators, yielding as a rule overestimates ("conservative" estimates), for the effective dose, and the organ-related equivalent doses. When a variety of radiations, energies, and angles of incidence are involved, the respective quantities relating to each of these are additive.
Three major operational quantities are used: two are used for area, or ambient monitoring purposes:

- ambient dose equivalent $H^*(d)$
- directional dose equivalent $H'(d, \Omega)$

and third one is used for individual monitoring purposes:

- personal dose equivalent $H_p(d)$

These quantities correspond to the dose equivalent produced at a point located at a depth d in a phantom (e.g. the ICRU sphere), or in the human body, which in turn depends on the energy of the radiation involved, and the geometric conditions pertaining to the exposure (direction of irradiation). For routine monitoring purposes, the values found for these operational quantities are deemed to provide adequately accurate estimates for the effective dose, and for the dose to the skin, especially if they are lower than the radiological protection limits.

Ambient dose equivalent $H^*(d)$ is the reference quantity for strongly penetrating radiation, for ambient monitoring purposes. $H^*(d)$ provides a good estimator for the *effective dose*. As the recommended depth d, in that case, is 10 mm, this quantity may then be noted $H^*(10)$. Many detectors used as dose-rate meters are calibrated with reference to $H^*(10)$.

Directional dose equivalent $H'(d, \Omega)$ is the quantity used for low-penetrating radiation, for ambient monitoring purposes. $H'(d, \Omega)$ stands as an estimator for the equivalent dose to the skin H_{skin}. Consequently, the recommended depth, in this case, is 0,07 mm. This quantity may thus be noted $H'(0.07, \Omega)$.

Personal dose equivalent $H_p(d)$ is the quantity used for personal monitoring purposes. Two cases may arise: for strongly penetrating radiation, the recommended depth stands equal to 0.07 mm, the quantity is then noted $H_p(10)$, this providing a good estimator for the effective dose. For low-penetrating radiation, the recommended depth stands equal to 0.07 mm, the quantity is then noted $H_p(0.07)$, which is a good estimator for the equivalent dose to the skin H_{skin}.

Dosimeters worn on the surface of the body, serving for workforce monitoring purposes, are calibrated with reference to $H_p(10)$ and $H_p(0.07)$, they thus yield good estimates for the effective dose, and the equivalent dose to the skin. Such dosimeters are, as a rule,

covered with a tissue-equivalent material.
The depth d, or indeed any thickness of any given material may be stated in terms of the corresponding density thickness, expressed in gcm^{-2}, or in mgcm^{-2}. There are also detectors used for individaul monitoring purposes, serving to measure $H_p(3)$, this being an estimator for the equivalent dose to the lens of the eye H_{lens}. In Figure 4.3 is summa-

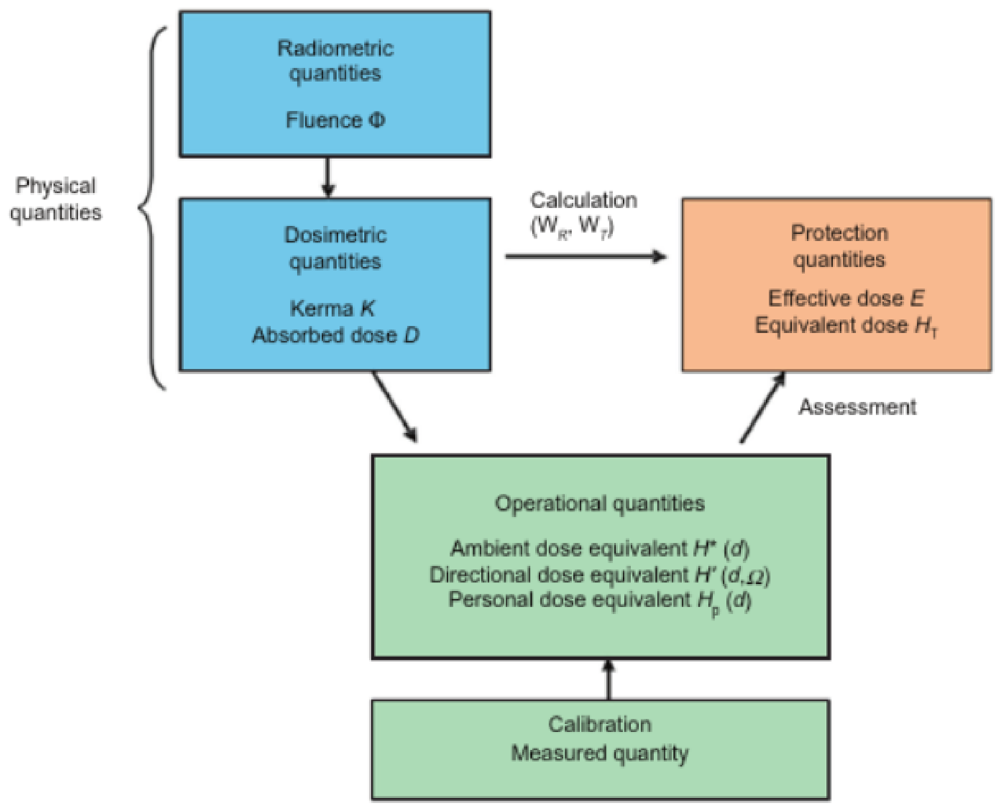

Figure 4.3: *Relationships between physical quantities, protection quantities, and operational quantities [7].*

rized the relationships between physical quantities, protection quantities and operational quantities.

ii. Gamma Shielding

Attenuation of photons by various absorbing materials under ideal narrow-beam conditions satisfy the relationship

$$I(x) = I_0 e^{-\mu x} \tag{4.3}$$

where I_0 is the initial photon intensity, a fluence or a flux, $I(x)$ is the photon intensity after passing through an absorber of thickness x in narrow-beam geometry, and $\mu(cm^{-1})$ is the total attenuation coefficient, which accounts for all interaction processes, including scattering reactions, that remove photons from the beam. The attenuation coefficient μ is

dependent on the particular absorber medium and the photon energy.
Values of μ generally increase as the Z of the absorber increases because photoelectric interactions are increased in high-Z materials especially for low-energy photons, and high-Z materials yield increases in pair production interactions for high-energy photons. Because of the high-Z effect, lead is often used to line the walls of x-ray rooms, made into lead aprons for personnel protection, and incorporated into leaded glass, and $BaSO_4$ is incorporated into concrete (called barite or barytes concrete) to increase its effectiveness as a photon shield.
Photon attenuation coefficients in various materials can be calculated from measurements of the intesity of a narrow beam of photons of a given energy. Many calculations of radiation exposure/dose from photon sources are straightforward once the flux is known. A useful formulation is the flux at a distance r from an attenuated point source:

$$\phi(x) = \phi_0 \frac{e^{-\mu x}}{4\pi r^2} \tag{4.4}$$

The expression $e^{-\mu x}/4\pi r^2$ is referred to as the *point kernel* which is the response at a point r from a source of unit strenght. The point kernel is used extensively in developing relationships between flux and exposure for various source geometries and absorbing media.
Although many radiation sources can, with its ease and utility, be represented as a point or an approximates a line source, a contaminated area that is representative of a disc or infinite planar source, and various volume sources. Practical approaches can be used to determine the photon flux from point kernels spread over such geometries, and once the flux has been determined it than be applied in the usual way to calculate radiation exposure [27].

iii. Neutron Shielding

The interactions that slow neutrons down and cause their eventual removal from a beam are probabilistic: they either occur or they do not. Consequently, a flux of neutrons of intensity I will be diminished in a thickness x of absorber proportional to the intensity of the neutron source and the neutron removal coefficient Σ_{nr} of the absorbing material:

$$-\frac{dI}{dx} = \Sigma_{nr} I \tag{4.5}$$

which has a solution

$$I(x) = I_0 e^{-\Sigma_{nr} x} \tag{4.6}$$

like a photon attenuation process, where I_0 is the initial intensity and $I(x)$ refers to those neutrons that penetrate a distance x in an absorber without a collision. Therefore $e^{-\Sigma_{nr} x}$ represents the probability that a given neutron travels a distance x without an interaction. Conceptually, Σ_{nr} can be thought of as the probability per unit path lenght that a neutron will undergo an interaction as it moves through an absorber and be removed from the

beam either by absorption or scattering. In this context then it very much resembles the attenuation coefficient for photons in *good (narrow-beam) geometry*, and can be similarity developed and used for neutron shielding and dosimetry.
The features of neutron beams, including the concept of *narrow-beam* effects, are shown in Figure 4.4, where different interactions remove a neutron which doesn't reach the detector.

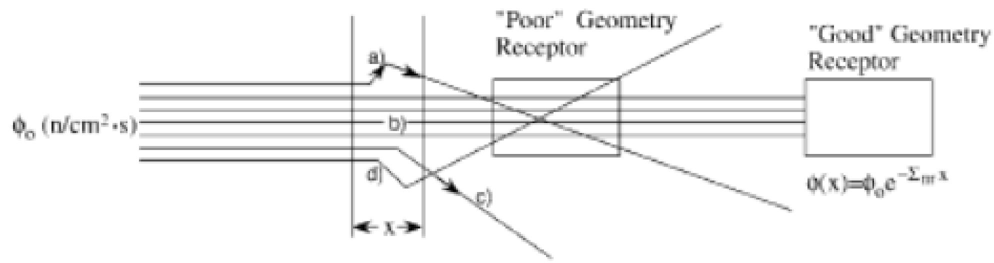

Figure 4.4: *Absorber of thickness x that depletes a beam of neutron because of (a) an inelastic scattering followed by a photon emission and a scattering reaction back into beam, or (b) an absorption or capture (grey lines), (c) elastic scattering out of the beam, or (d) elastic scattering with additional scattering back into the beam.*

As a consequence of the complicated resonance structure of the neutron cross-section, neutron removal coefficient Σ_{nr} can change irregularly. Neutron removal is determined experimentally for each shield material as a combined removal coefficient. These are included in Table 4.1, for shields with a thickness of less than mean free paths and at least 6 g/cm^2 of hydrogenous material, also included are values for shielding materials, only substance [28].

II. Enviromental Impact

A proton accelerator could impact on environment because of the prompt radiation and the emission of radioactive effluents, each of which may have an off-site radiological impact. A component of the direct radiation field is the *Skyshine*, due to the fact that the shielding in the vertical direction is not always constrained in this way so that more radiation (usually neutrons) may be emitted from the roof shielding of an accelerator at levels that may have an off-site impact.
Because the thresholds for nuclear reactions for neutrons with the constituents of air all lie near or above 20 MeV, the interactions below this energy are restricted to elastic scattering. The high energy nuclear interaction lenght for N_2 and O_2 are of the order of 90 gcm^{-2}, which for the density of air is of the order of 750 m and hence the high energy neutrons effectively escape to great distances. Because only low energy neutrons could be scattered backward, these neutrons predominate.
Due to mass ratio of neutrons to nitrogen and oxygen nuclei, many elastic scatters are needed in order to reduce the neutron energy. It follows that, as a first approximation, there is no effective attenuation of neutrons in air and the primary reduction in fluence

out to a few hundred meters derives from geometrical factors. Dependence of neutron dose on distance from the source is therefore in the first instance a purely geometrical effect. As particle number must be conserved, the dose is inversely proportional to the area over which the particles are dispersed

$$H = \frac{H_a A}{2\pi r^2} \tag{4.7}$$

where $H_a \cdot A$ is the mean dose on A area of the roof and r is the distance from the source to the field point of interest [29].

III. Induced radioactivity

In an accelerator bunker for energies $E < 30$ MeV, radionuclide production by direct reactions such as single and multi-nucleon transfer as well as processes such as (p, γ) are of principal concern. The systematic and approximate energy dependences of these processes are well understood. Reactions that occur inside bunker are endoergic nuclear reactions with a E_{th} equal to:

$$E_{th} = \frac{m_p + M}{M}|Q| \tag{4.8}$$

The Q-value is the difference between the separation energy of the in-going and out-going particles in the absence of excitation energy in either the entrance or the exit channels.
It is also quite common for thermal neutrons to produce significant levels of induced radioactivity in the accelerator room. Such radioactivity results from thermal neutron capture reactions that sometimes can have relatively large cross sections. As the energy of the incident radiation increases, the number of possible reaction channels increases, with a corresponding increase in the number of radionuclides produced. The variety of radionuclides that can be produced becomes higher as one raises the bombarding energy because more reaction thresholds are exceeded. Induced radioactivity produced by neutron irradiation is more probable compared with gamma and charged particles. This is explained by the fact that gamma and charged particles must have high energy of at least a few MeV to start a nuclear reaction with the creation of radioactive nuclei, whereas neutrons are electrically neutral and easily penetrate the nuclei to make it radioactive. For protons or heavy ions, the nuclear rupture caused by the collision between the initial particle and the target material and the nuclear reaction between the generated secondary neutrons and the material are also important ways to generate induced radioactivity. Heavy ion projectiles are fragmented and remain implanted in the target after hitting it. Induced activity at modern proton and heavy ion accelerators is characterized by levels of the order of $10^2 - 10^3$ μSv/h (depending on cooling times and irradiation scenarios) and causes significant challenges during commissioning and repair work. The ratio only applicable to induced radioactivity in air, water and unshielded accelerator structures, while the ratio in targert or material test areas is often much higher. Sometimes, for such work, shielding is required to reduce radiation to levels that are acceptable in terms of dose limits and optimization of staff, or, with adequate consideration for the area's

intended level of human occupancy. The contribution from the induced activity of the concrete shielding to the total radiation background in the tunnels of the accelerator after it ceases can be significant, comparable to the background from the activation of various metal parts of the accelerator. The activation of soil and water ponds near accelerators creates significant problems. Due to their large dimensions and the level of prompt radiation generated during operation, high-energy accelerators are generally located underground. Following soil activation, generated radionuclides may leak and migrate into numerous streams and bodies of water. Neutrons passing through accelerator foundations constitute the most significant contribution to soil activation. The soil is often weakly activated, and its activation is determined by the long-lived radionuclides such as ^{3}H ,^{7}Be ,^{22}Na ,^{45}Ca ,^{54}Mn ,^{55}Fe. The main contribution to the activation of ground water is made by ^{3}H and ^{22}Na, which form readily soluble compounds. In Table 4.2 are summarized radioactive nuclides produced in a particle accelerator.

Table 4.1: *Neutron removal coefficients Σ_{nr} for thermal neutrons for materials which are surrounded by sufficient hydrogenous material to absorb neutrons that are degraded in energy due to scattering interactions.*

Material	$\Sigma_{nr}(cm^{-1})$
Sodium	0.032
Graphite	0.078
Carbon	0.084
Concrete	0.089
Heavy Water	0.092
Zirconium	0.101
Water	0.103
Paraffin	0.106
Polyethylene	0.111
Lead	0.118
Beryllium	0.132
Iron	0.156
Copper	0.167
Uranium	0.182
Tungsten	0.212

Table 4.2: *Induced radioactivity in proton accelerators facilities [30].*

Material	Isotope	Threshold (MeV)	Half-life	σ(mb)	Decay mode
Plastic, oils	^{3}H	11	12.33 y	10	$\beta-$
-	^{7}Be	2	53.22 d	10	EC
Al	^{22}Na	30	2.60 y	10	$\beta+$
Iron	^{44m}Sc	-	2.44 d	-	IT
-	^{46}Sc	-	83.8 d	-	$\beta-$
-	^{47}Sc	-	3.35 d	-	$\beta-$
-	^{48}Sc	-	1.82 d	-	$\beta-$
-	^{48}V	20	15.97 d	6	$\beta+$
-	^{51}Cr	30	27.7 d	6	EC
-	^{52}Mn	20	5.59 d	30	$\beta+$
-	^{54}Mn	30	312.1 d	30	EC
-	^{55}Fe	-	2.74 y	-	EC
-	^{59}Fe	-	44.5 d	-	$\beta-$
-	^{56}Co	5	77.2 d	30	$\beta+$
-	^{57}Co	30	271.7 d	30	EC
-	^{58}Co	30	70.9 d	25	$\beta+$
Steel	^{59}Ni	-	75 y	-	EC
-	^{60}Co	30	5.27 y	15	$\beta-$
Copper	^{63}Ni	-	100 y	-	$\beta-$
-	^{65}Zn	-	243.7 d	100	EC

Chapter 5

GEANT4 simulation

The GEANT4 (GEometry ANd Tracking) is a software coded in a object oriented programming language, C++ [31]. It can simulate the passage of particles through matter, with a lot of application in high energy, nuclear and accelerator physics, and medical and space science. It is made for detector design and response to particles, signal amplitudes, energy spectra, resolutions, efficiencies. The minimal tasks to provide in order to have a complete process are:

- provide geometrical and material description of apparatus
- provide event generator
- choose an appropriate physics list and procution cuts

The software produce particles and simulate interactions, and propagate the secondary particles. At the heart of Geant4 is an abundant set of physics models to handle the interactions of particles with matter across a very wide energy range. Geant4 is written in C++ and exploits advanced software-engineering techniques and object-oriented technology to achieve transparency. In order to show the class description and utilization, is useful to put them in a diagram, as in figure 5.1. Here are showned categories and ereditariety. So at the bottom of the diagram there are categories used by virtually all higher categories and provide the foundation of the toolkit.

Global category covers the system of units, constant, numeric and random number handling.

Material and Particle implement facilities necessary to describe the physical properties of particles and materials for the simulation of particle-matter interactions.

Geometry module offers the ability to describe a geometrical structure and propagate particles efficiently through it. Above these reside categories required to describe the tracking of particles and the physical processes they undergo.

Track category contains classes for tracks and steps, used by processes.

Processes category, which contains implementations of models of physical interactions: electromagnetic interactions of leptons, photons, hadrons and ions, and hadronic interactions.

Tracking category, which manages their contribution to the evolution of a track's state and provides information in sensitive volumes for hits and digitization.

Event category manages events in terms of their tracks.

Run category manages collections of events that share a common beam and detector implementation.

Readout category allows the handling of pile-up.

All of these categories use some capabilites to connect to facilities outside toolkit through abstract interfaces, in order to provide *visualization, persistency and user interface.*

I. First step simulation

The aim of this simulation project is the study of ambiental dose inside bunker, due to proton beam smashing on the matching line at the end of the RFQ-injector. So the first simulation is developed in order to obtain a contour map of the ambiental dose produced by a pulse of the beam, and a mean number of secondary particles produced. In this starting phase, is designed not only the geometry, but also the electromagnetic field, in order to simulate the lost number of protons in the beam pipe for every tank. So at initial step isn't simulated the beam outside accelerator line, simulation is stopped at the end of Module 0.

i. Geometry

The simulation of the accelerator LINAC of the ERHA project starts with the geometry construction of the matching line between the RFQ output and the input of *Module 0*, which specifications are listed in Table 5.1. The *Module 0* is designed in a semplified way,

Table 5.1: *Matching Line specifications.*

Element	Lenght	In. Radius	Ext. Radius	Material	Gradient
PMQ 0	20 mm	3 mm	10 mm	Alluminium	-160 T/m
Drift	65 mm	3 mm	3,2 mm	Stainless-Steel	-
PMQ 1	20 mm	3 mm	10 mm	Alluminium	196 T/m

in which tanks are cilindric and have same size for the entire module. This semplification is of course a limit, but as showned after field is resized and constant in the tank. So *Module 0* is descripted in Table 5.2. This assembly is copied 5 times and it is 1 meter long. The class in which is defined is *LXeAccelerator()*, written in Appendix A.

ii. Materials

The beam pipe is the volume with a vacuum density material, and it is made with materials in Table 5.3 with a pressure of 10^{-7} Pa. The classe in which materials are defined is *LXeDetectorConstruction()*, written in Appendix B.

iii. Fields

The beam pipe is also the volume in which protons are accelerated, so here is defined the field. Because of difficulties in setting a non-uniform and non-constant field, the volumes of the beam-pipe inside the tanks are filled with electric field with a constant value along z-axis that is the mean value of the original time-dependent field in the tanks. Class *ElectricFieldSetup()* can perform this operation.
The beam pipe of the quadrupole is filled with a quadrupole magnetic field, thanks to *QuadrupoleMagneticField()* class used in *FieldSetup()*. In order to build a multithreading executable object, fields are defined *static* and *G4ThreadLocal*.

iv. Results

A first interesting result is obtained by a *voxelitation* of the space of the envelope around accelerator. In this way is studied the dose released for a bunch of 10^6 protons. In figure 5.2 is clear that all the dose is released at the center in the beam pipe. In figure 5.3 dose is released at first and second quadrupole, as is showned also in figure 5.4.

II. Second step simulation

The second step of the simulation project is focused on a detector design. The gamma and neutron prompt radiation fields were measured by two different detectors. Simulation starts from the construction of detector, in order to simulate event detection of a gamma or a neutron.

Table 5.2: *Module 0 specifications.*

Element	Lenght	In. Radius	Ext. Radius	Material	Gradient(or $\vec{E}$)
Tank 0	60 mm	1.5 mm	30 mm	Copper	4.8 MV/m
Drift	10 mm	3 mm	3.2 mm	Stainless-Steel	-
PMQ 2	20 mm	3 mm	10 mm	Alluminium	-179 T/m
Drift	10 mm	3 mm	3.2 mm	Stainless-Steel	-
Tank 1	60 mm	1.5 mm	30 mm	Copper	4.8 MV/m
Drift	10 mm	3 mm	3.2 mm	Stainless-Steel	-
PMQ 3	20 mm	3 mm	10 mm	Alluminium	186 T/m
Drift	10 mm	3 mm	3.2 mm	Stainless-Steel	-

i. Geometry

Detectors are placed near the accelerator at 1 meter, at same altitude of the beam-pipe as showned in figure 5.5. Simulation in this step is focused on the interaction between beam and materials around beam-pipe, and the tracks are stopped outside bunker without a interaction with the walls. Geometry description is different for gamma and neutron detectors: they are parallelepipedal shaped, gamma is 6x6x20 cm^3, while neutron is 20x20x40 cm^3. They consist of a scintillation material and a photomultiplier of 2.3 cm of radius on one side, covered by aluminium. Neutron detector is surrounded by 3.0 cm of polyethylene material. In figure 5.6 is visible the gamma detector.

ii. Materials

These dose monitor detectors are scintillating one (see Table 5.4). The glass scintillator inside neutron detector (density 2.4 g/cm^3) is composed as descripted in Table 5.5.

iii. Results

First is simulated gamma detection, then neutron detection, by changing the *MainVolume* class.

Gamma Output is a table of simulated quantities (see Table 5.6), where *protons* is the number of particle injected, *hits* is the total number of optical photons that hit the pmt in the scintillator, *hits over threshold* is the total number of events detected, *scintillation ev,* is the number of optical photons produced by scintillation, *cherenkov ev.* is the number of optical photons produced by cherenkov effect, *absorbed ev.* is the number of optical photons absorbed by scintillator material, *absorbed energy* is the total energy absorbed in the scintillator and *integrated dose* is the total dose released in the simulated world.
In order to make a comparisation, linear extrapolation gives $\sim 10^3$ gamma with respect to 10^{11} protons, which is the number of protons accelerated in one second by injector. At this value correspond an equivalent dose of:

$$1 * 10^{-12} Sv \cdot cm^2 * \frac{10^3}{cm^2 \cdot s} = 10^{-9} \frac{Sv}{s} = 3.6 \frac{\mu Sv}{h} \tag{5.1}$$

Table 5.3: *Vacuum composition.*

Element	Fraction
Oxygen	30%
Nitrogen	69%
Hydrogen	1%

Neutron The simulation of neutron detector is a little more complicated, because of the small rate of production. Indeed, in order to be sure is applied a window on total hits on pmt in scintillator detector, after a check on the peak position, like in figure 5.7. So when hits for simulated detector are $800 < hits < 1200$, it is a neutron. A first simulation is shown in Table 5.7, while a more statistically efficient simulation gives the results in Table 5.8. Again this value must be compared with the measurement. So with 10^{11} protons per second, the number of neutrons could be obtained by a linear fit of data. Conversion to equivalent dose gives:

$$391 * 10^{-12} Sv \cdot cm^2 * \frac{170}{cm^2 \cdot s} = 66470 * 10^{-12} \frac{Sv}{s} = 239 \frac{\mu Sv}{h} \tag{5.2}$$

III. Third step simulation

The last step include the simulation of all the other modules up to 50 MeV of acceleration. The Module with CCL tanks is constructed with the use of a opensource library *CADMesh* [32]. For a simulation of the entire process is required a lot of CPU-time, as long as 5 days for 10^9 protons, with results shown in Table 5.9. The entire accelerator room is a bunker with thick walls (1.20 meters), of concrete material with a composition like in Table 5.10, as is shown in figure 5.8. A very important result is the study of energy of the beam for all the acceleration modules. In order to compute the energy of the beam is useful to plot the kinetic energy of the protons inside beam-pipe during a *step*, a GEANT4 object that memorize several parameteres between two interaction. In figure 5.9 is shown in the x-axis the energy, while on the y-axis there is number of steps. $N(T)$ is the number of steps at T kinetic energy, and this is proportional to $N_p(T)$ that is the number of protons, proportional to $I(T)$, current of the beam:

$$\frac{N(T)}{N(0)} = \frac{N_p(T)}{N_p(0)} = \frac{I(T)}{I(0)} \tag{5.3}$$

where $I(0)$ is the mean peak current of the RFQ. At the end of *Module 0* the current is:

$$I(T) = 0.25 * 1.2mA = 300\mu A \tag{5.4}$$

As shown results aren't statistically good, because of CPU-time consumption. So in order to reduce time is possible to change run from *singlethread* to *multithread*.

Table 5.4: *Scintillator specifications:* R.Ind *is the refractive index,* Abs. Lenght *is the absorbtion lenght,* Yield *is the number of gamma per unit of energy.*

Scintill.	R.Ind.	Abs. Lenght	Yield	τ	Birks constant
Polystyrene	1.58	250 cm	10 g/keV	2.4 ns	0.126 mm/MeV
Glass	1.59	420 cm	10 g/keV	18 -/- 45 ns	0.126 mm/MeV

i. Multithreading

The multithreading approach is obtained using *G4MTRunManager* class. Use of shared object defined *static* and *G4ThreadLocal* in *LXeDetectorConstruction.hh*, gives rise to a memory reduction, but in primis delete errors during run due to field and geometry. It needs 12 hours for a run with 10^9 protons, a very time-consuption reduction. In Table 5.11 are shown results of simulation and in Figure 5.10 is shown the linear fit used to derive the coefficient for the expected dose rate.

ii. Superposition of sources

As a consequence of the high number of protons accelerated by simulation with respect to real current at the end of module 7, as is shown in figure 5.9, a new method of simulation could be a separation between injection and beam output in air. This could save a lot of time, because very high time-consuption is needed for simulation in air.

Table 5.5: *Glass scintillator specifications.*

Element	Fraction
C	84.18%
H	7.82%
Li6	7.5%
Ce	0.5%

Table 5.6: *Results with gamma detector.*

Protons	10^9
Hits	1250
Hits over threshold	11
Scintillation ev.	3100
Cherenkov ev.	0
Absorbed ev.	1870
Absorbed Energy	1170 keV
Integrated dose	$1.1 * 10^{-12}$ Gy

Table 5.7: *Second step neutron simulation* 10^9 *protons.*

Protons	10^9
Hits	17600
Hits over threshold	17
Scintillation ev.	38600
Cherenkov ev.	704
Absorbed ev.	21400
Absorbed Energy	9357 keV
Integrated dose	$1.02 * 10^{-12}$ Gy
Neutrons	1

Table 5.8: *Second step neutron simulation* 10^{10} *protons.*

Protons	10^{10}
Hits	3729000
Hits over threshold	2200
Scintillation ev.	8206000
Cherenkov ev.	70000
Absorbed ev.	4500000
Absorbed Energy	1243000 keV
Integrated dose	$1.2 * 10^{-11}$ Gy
Neutrons	17

Table 5.9: *Third step neutron simulation.*

Protons	10^9
Hits over threshold	34
Neutrons	3

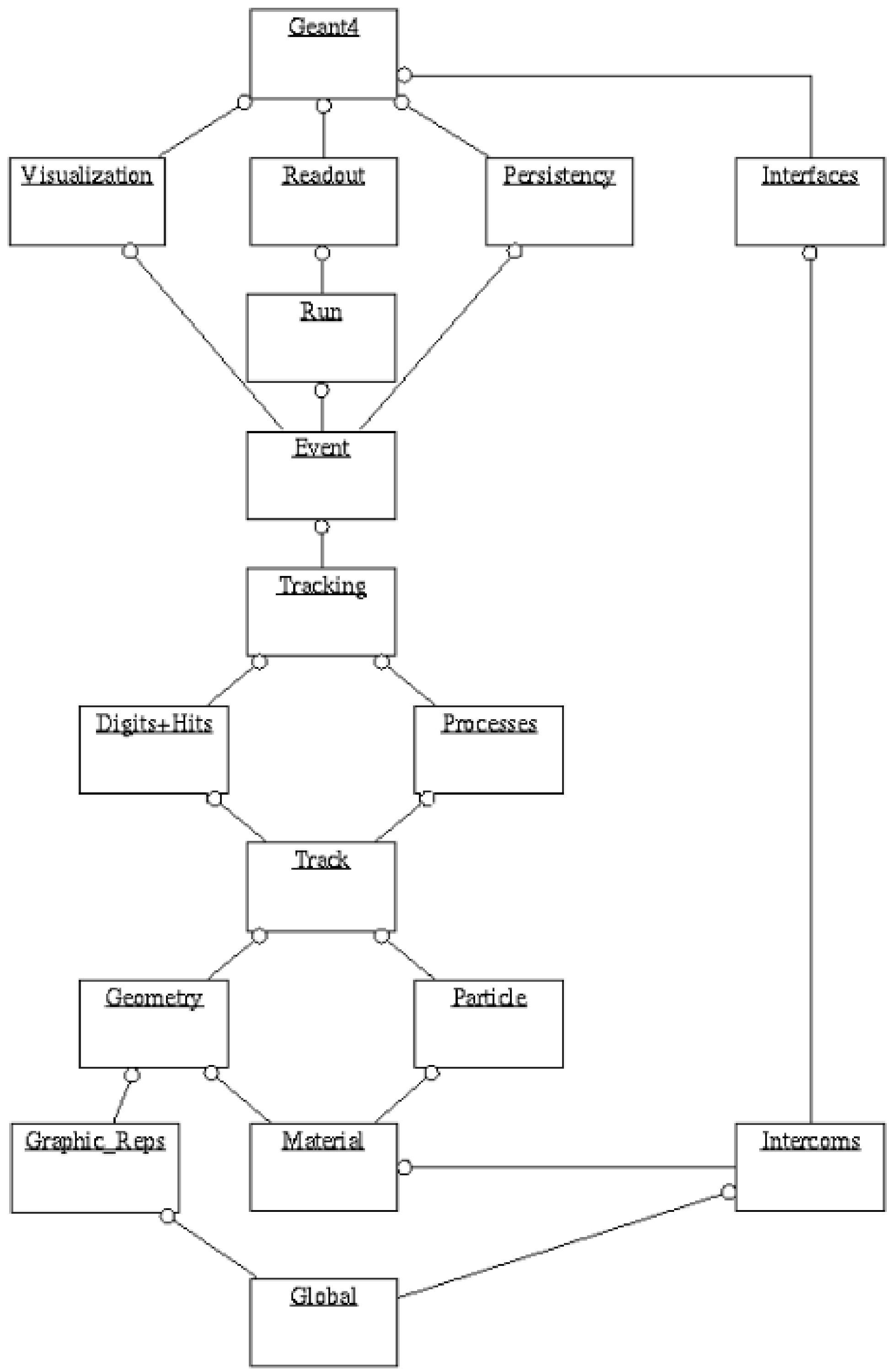

Figure 5.1: *Geant4 class category diagram.*

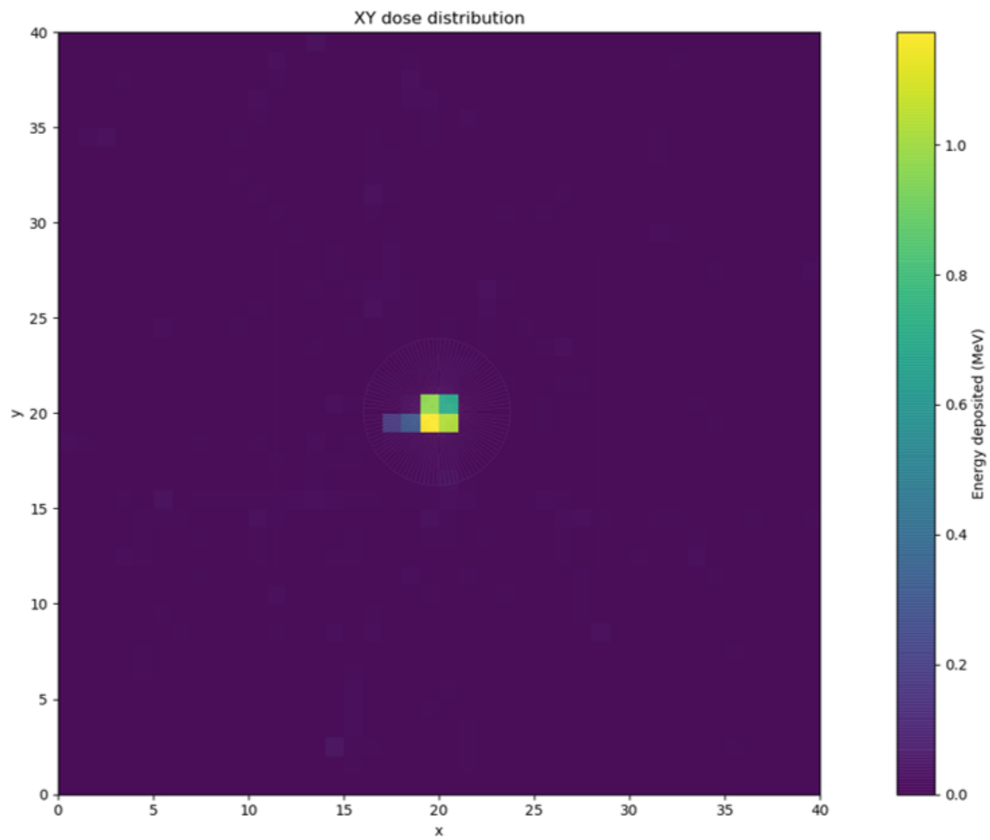

Figure 5.2: *The map of the dose in xy plane.*

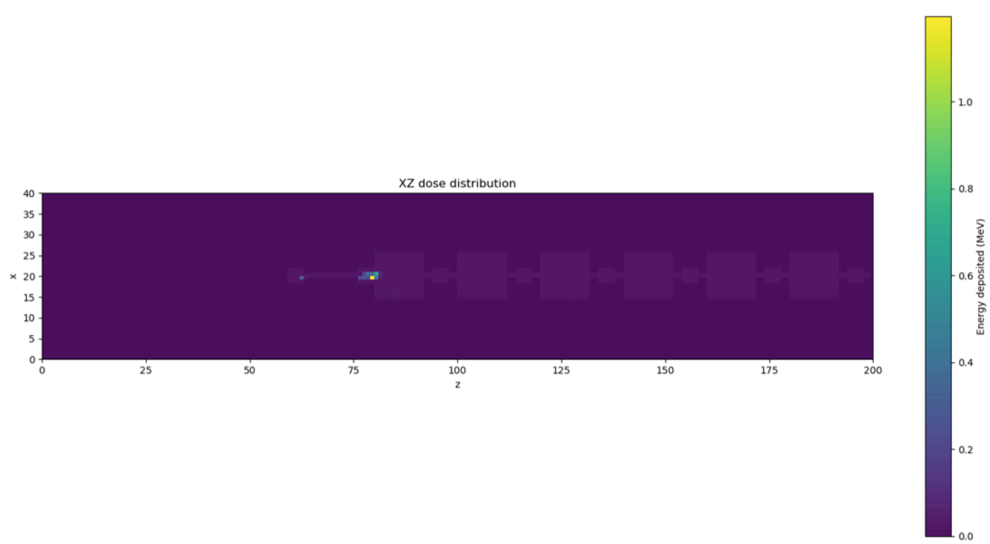

Figure 5.3: *The map of the dose in xz plane.*

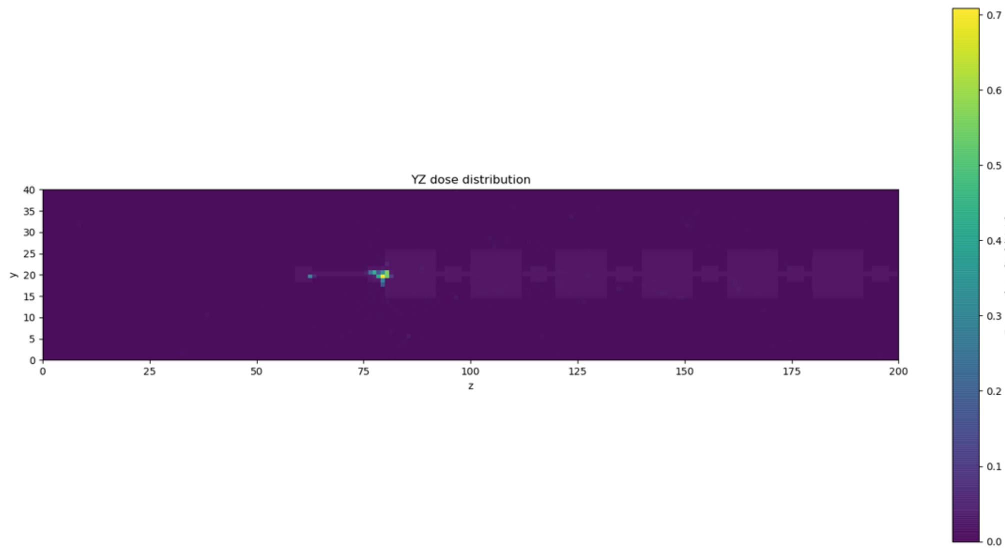

Figure 5.4: *The map of the dose in yz plane.*

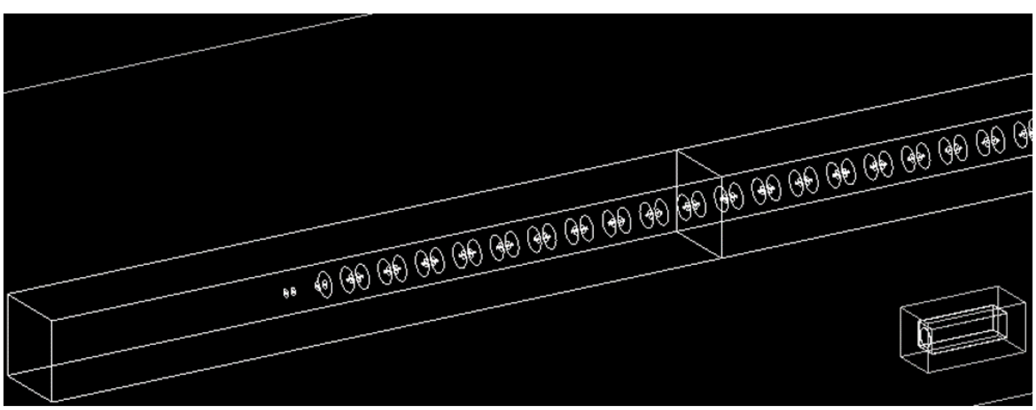

Figure 5.5: *A screen of the neutron detector position.*

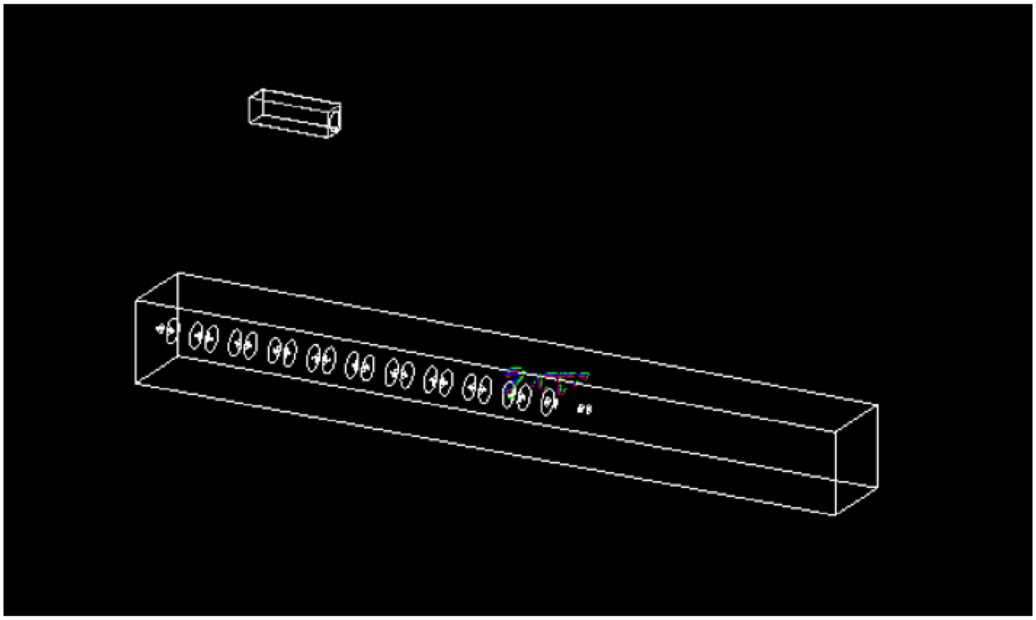

Figure 5.6: *A screen of the gamma detector position.*

Table 5.10: *Concrete composition.*

Oxygen	49.2875%
Carbon	5.62%
Hydrogen	0.6%
Sodium	0.453%
Magnesium	0.663%
Alluminium	2.063%
Silicon	18.867%
Potassium	0.656%
Calcium	20.091%
Iron	1.118%
Phosphorus	0.048%
Sulphur	0.012%
Titanium	0.347%
Manganese	0.0387%
Zinc	0.0241%
Zirconium	0.0074%
Barium	0.0179%
Lead	0.0464%
Strontium	0.04%

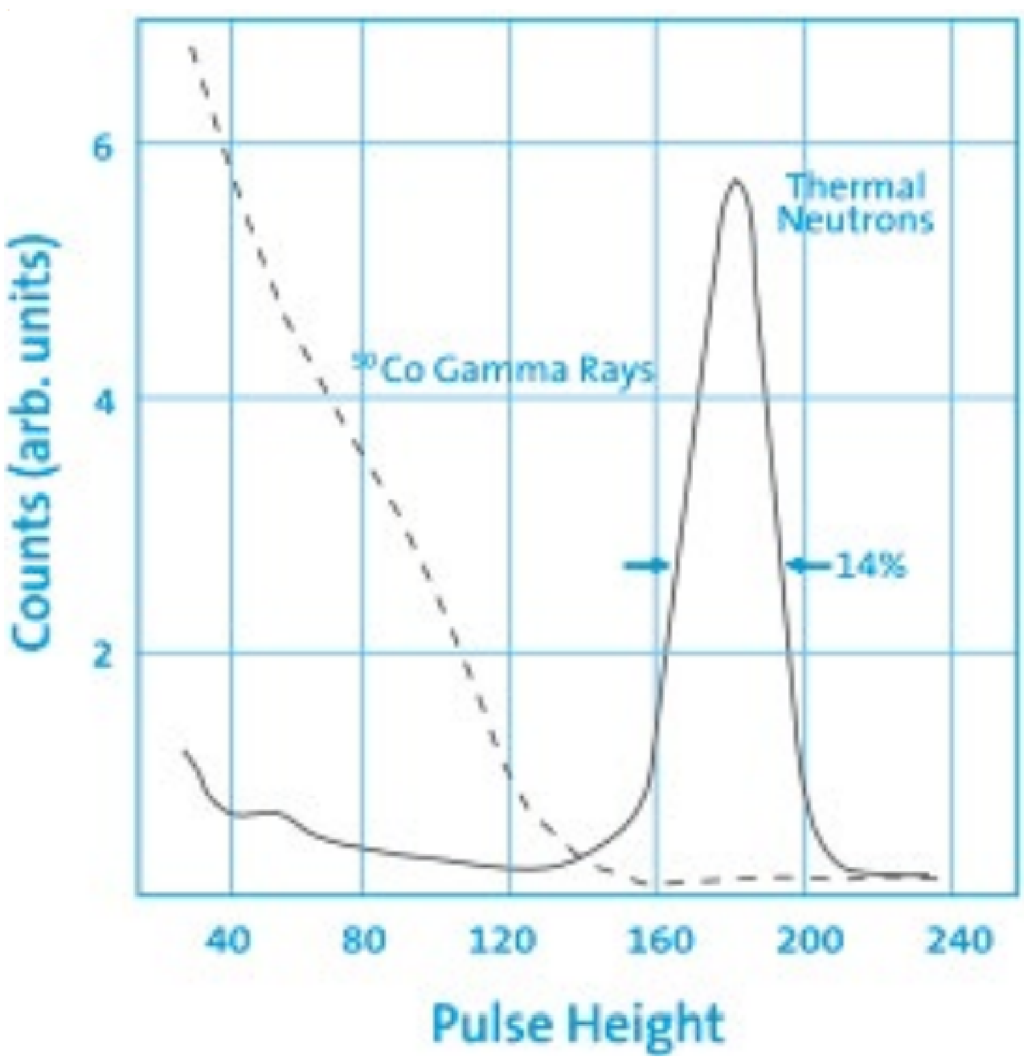

Figure 5.7: *Pulse height curve of a glass scintillator filled with lithium.*

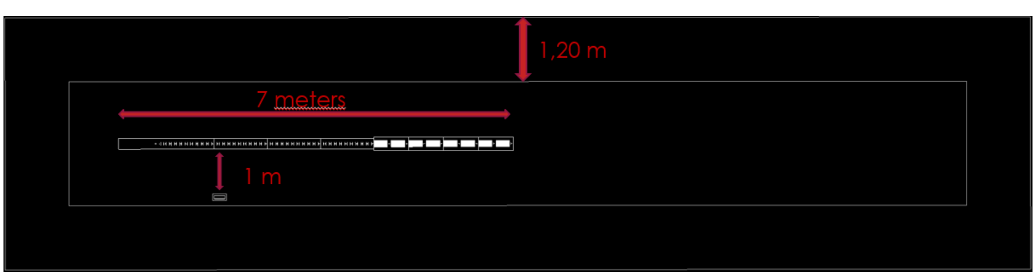

Figure 5.8: *Top to bottom view of the simulated bunker.*

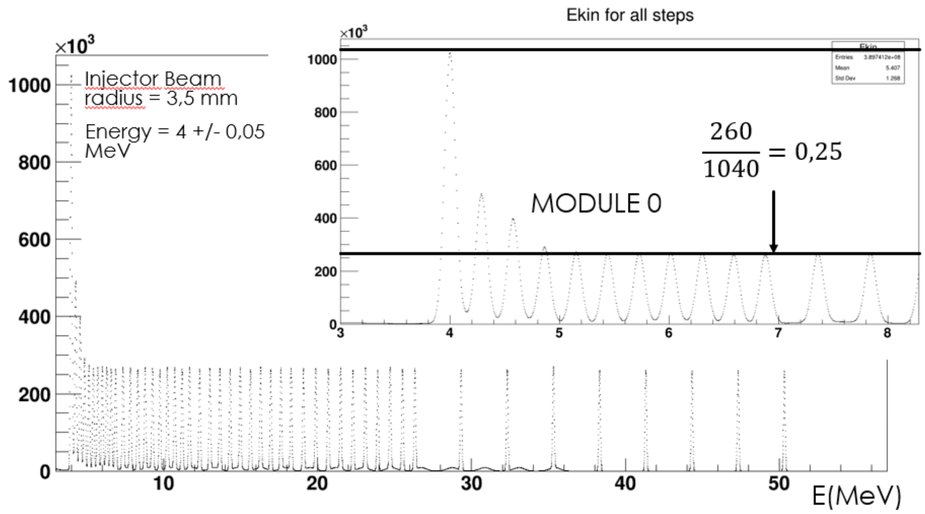

Figure 5.9: *The number of steps inside beam-pipe releated to kinetic energy of the beam.*

Table 5.11: *Number of neutrons into detector simulated for different numbers of protons.*

Protons	Neutrons
10^9	3
10^{10}	19
10^{11}	315
10^{12}	2836
Intercept	(-0,9342 $\pm$ 1,6150) n
Slope	(2,853 10^{-9} $\pm$ 5,103 10^{-11}) n/p

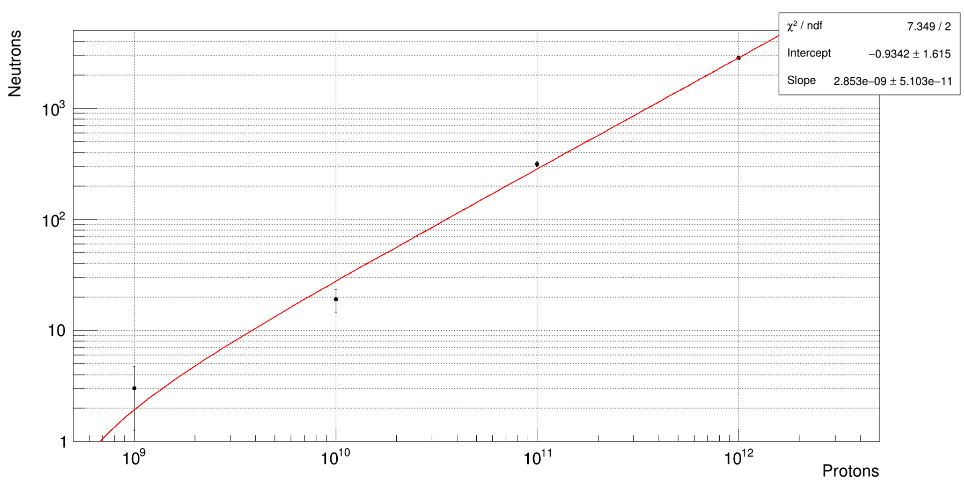

Figure 5.10: *Linear fit, parameter extraction with MIGRAD-ROOT.*

Chapter 6

Gamma and Neutron detection

Detection of particles and dose measurement in a pulsed neutron field is a issue for particle accelerators, where beam is lost near targets, collimators and beam dumps. The interest in active detectors to be employed in pulsed neutron field is constantly increasing due to the growing number of applications where the pulse structure of the radiation field hinders the use of active detectors operating in pulse mode in the halls housing acceleratos, including the treatment rooms.
Although the $H^*(10)$ which characterises a single burst is usually low, and would no constitute a problem in terms of averaged $H^*(10)$ rate over the entire measurement period, the $H^*(10)$ rate during the radiation burst can reach extremely high values, up to 100 Sv/h in typical medical diagnostic applications and up to 10^7 Sv/h [33] in facilities such as the ones used for material testing, and this usually leads to severe underestimations of the $H^*(10)$. Recent measurements around research particle accelerators reported severe under reponse in commercial active neutron detectors.
Main cause of this underestimation can be attributed to dead time losses, which are a distinctive feature of active detectors working in pulse mode. An active radiation detector can in fact operate in three modes: current, mean square voltage (MSV) and pulse mode. Current mode averages out the fluctuations in the intervals between individual interactions and is usually employed with high interaction rates when there is no need of preserving the information on the amplitude and timing of single interactions. MVS mode, whose detection principle is based on a special processing of the fluctuating component of the detector current signal, becomes useful when making measurements in mixed radiation environments when the charge produced by one type of radiation is much different than that from the second type. Pulse mode is the most commonly applied, for preserving information on the amplitude and timing of individual events. In this detector each pulse represents the results of a single interaction. Disadvantage of this property is dead time and losses induced by the counting system, which can be correctly compensated only in the case of steady-state sources of constant intensity, but not in the case of pulsed sources. An ideal detector would in fact count every event that occurs. However, a real detector and its read-out electronics need a specific amount of time to create and process an output pulse. An event that occurs during this time cannot be

registered. Depending on the detector system, it is either suppressed (non paralysable systems) or changes the shape of the previously detected pulse, resulting in a pile up (paralysable systems)[34]. The minimal time between two separately detectable events is called the dead time and it ranges from 1 to 10 μs for neutron dose detectors. It is possible to define five requirements that an active neutron detector should show for efficiently working in pulsed neutron field:

- capability to withstand very high instantaneous neutron fluxes with little or no saturation;
- high sensitivity, usually expressed in nSv^{-1}, at least comparable with that of commercially available rem counters, i.e. about $1\,\mathrm{nSv}^{-1}$;
- capability to reject the photon contribution that usually accompanies the neutron fields;
- capability to measure correctly the intensity of a single neutron burst;
- good sensitivity over the entire neutron energy range, especially to high-energy neutrons.

I. Dead-time correction factor

The difficulty which arises in electrical counting when the source is pulsed is due to the finite resolution of any counting system. Thus, following any count, there is a dead-time during which the system would be unable to record a further count, should one occur. It needs a careful study of the counting losses expected under that situation. It is seen that one cannot simply apply equations for continuous counting at the enhanced counting speed, the error in doing so would be greatest for conditions which are very likely to arise in practice, for example, when the pulse lenght does not greatly exceed the dead-time of the system. It is also shown that it is desirable to reduce the dead-time to the minimum. The studies of C. H. Westcott ([35]) on the soppression factor introduced when $\frac{T}{\tau}$ (T is the pulse period), although greater than unity, cannot be considered large, lead to the simple formula

$$E' = E\frac{1}{X}\left(\frac{1}{2} + \frac{X - \frac{1}{2}}{1+z}\right) \tag{6.1}$$

where:

- E is the expected total count, including suppressed counts;
- E' is the expected total of recorded counts;
- X is $\frac{T}{\tau}$ the duration of the pulse in units of the dead-time;
- z is $\nu\tau = \frac{N}{T}\tau$ is the expected mean counting rate during the pulse (ν) in units of the dead-time, indeed the average expectation of counts within one dead-time.

The factor for the standard condition of 0,6 mA of injector current could be calculated for a dead-time value of 1,0 μs as measured in [36], and it gives

$$\frac{E'}{E} = \frac{1\mu s}{4\mu s}\left(\frac{1}{2} + \frac{4 - \frac{1}{2}}{1 + \frac{43}{4\mu s}1\mu s}\right) \sim 0,2 \tag{6.2}$$

while for a current of 1,2 mA it gives 0,17.

II. Measured equivalent dose

of accelerator started with an evaluation of gamma dose recorded in real time with a safety system monitor builded by *Atomtek*, a plastic scintillator detector. In order to measure the neutron dose in the bunker are set 3 different neutron probe in different positions. *Atomtek* built the first detector, a proportional counter with ^{3}He in a polyethylene moderator, which is set using an FPGA with a RS232 protocol port by which are inserted commands using a LabView interface. After a first measure, a complete set of measurement is obtained with a *Thermo-BIOREM 752* probe, a proportional counter with BF_3 placed in a cylindrical moderator containing polyethylene and boron carbide.

i. Gamma Atomtek BDKG-04

Atomtek BDKG-04 x-ray and gamma probe measures equivalent ambient dose and equivalent dose rate with features specified in Table 6.1 and shown in Figure 6.1. The monitor

Table 6.1: *Atomtek BDKG-04 features.*

Detector	scintillation plastic
Dose equivalent rate	0,05μSv/h - 10 Sv/h
Instrinsic measurement error	$\leq$ 20%
Energy range	15 keV - 3 MeV
Energy sensitivity response	15 - 60 keV $\pm$ 35% 60 keV - 3 MeV $\pm$ 20%
Sensitivity on ^{137}Cs	70 cps for 1μSv/h
Operating temperature range	$-30\,°C + 50\,°C$
Relative humidity	35 °C up to 98%
Protection class	IP64
Radio disturbance	CEI/IEC CESPR 22:1997
Electromagnetic compatibility	CEI/IEC 61000-4-2:1995 IEC 61000-4-3:1995
Weight	0.45 kg
Dimensions	$\oslash$ 60x200 mm

system consists of a set of gamma-probes located in different positions in the bunker. In Figure 6.2 is shown where are placed the detectors. The detector is placed at 1 meter from beam-pipe at link point between Module 0 and Module 1. Gamma measurements are summarized in Table 6.2.

Table 6.2: *Measured gamma dose for* 10^{11} *protons.*

Protons	10^{11}
Dose rate	3,07μSv/h

ii. Neutron Atomtek BDKN-03

Neutron detector BDKN-03 is a proportional counter with ^{3}He in a polyethylene moderator, as depicted in Table 6.3 where are summarized the detector features. In Figure 6.3 is

Table 6.3: *Atomtek BDKN-03 features.*

Detector	^{3}He proportional counter
Dose equivalent rate range	0,1μSv/h - 10 mSv/h
Dose equivalent range	0,1μSv - 10 Sv
Instrinsic measurement error	$\pm$ 20%
Energy range	0,025 eV - 14 MeV
Sensitivity to Pu$-$Be (dose mode)	0.355 cps for 1μSv/h
Flux density range	0.1 - 10^4 neutron$\cdot s^{-1} \cdot cm^{-2}$
Sensitivity to Pu$-$Be (flux mode)	0.5 cps for 1μSv/h
Operating temperature range	$-30\,°C + 50\,°C$
Relative humidity	$\leq$ 35 °C up to 95%
Protection class	IP64
Weight	8 kg
Dimensions	316x220x265 mm

shown a picture of the detector with LabView interface. Measured dose rate is plotted in Figure 6.4, 6.5 and 6.6. In Table 6.4 are summarized measured dose rate parameters. The analysis is made with RooFit, with a Gaussian Model for the signal and a Uniform model for the background.

iii. Neutron ThermoScientific-BIOREM 752

The ThermoScientific FHT 752 BIOREM in Figure 6.7 is a commercial neutron dose rate meter for stationary and portable use, especially suited for environmental measurements.

In Table 6.5 the probe features are summarized. It employs a BF_3 proportional counter placed in a cylindrical moderator containing polyethylene and boron carbide. The output is given in $H^*(10)$, but an internal calibration factor, expressed in nSv/count, can be set by the user, by calibrating it with a neutron source. The response function is given in Figure 6.8.

Linearity plot The expected $H^*(10)$ is calculated by appling to the beam intensity a coefficient, in n/p, neutron per proton, derived from a linear fit of the simulated neutron detection in Figure 5.10, and then by calculating the simulated dose with equation 5.2, and in the end by appling the dead-time factor in equation 6.1. In Table 6.6 are shown these values for every beam intensities. The results of the measurements are shown in Table 6.7. In Figure 6.9 is shown the linearity plot[36]. The lines fitting the experimental data points are compared with a straight line that represents the bisector of the first quadrant, i.e. the ideal linear reponse. The equation used to fit the data were chosen only as visual guides and are linear of the form $y = A \cdot x$. The measured values refer to the dose rate $H^*(10)$ detected in the stray field generated by a sequence of single beam pulses with a fixed repetition frequency. Uncertainty is statistical.

Table 6.4: *Measured dose parameters: second value is more precise than first one because it is obtained in floating mean mode.*

Current	Frequency	Dose rate	Standard Deviation
0.6 mA	10 Hz	133,67 μSv/h	36,64 μSv/h
0.6 mA	10 Hz	134,66 μSv/h	7,01 μSv/h
1.0 mA	10 Hz	200.05 μSv/h	15,01 μSv/h
1.2 mA	10 Hz	224,49 μSv/h	51,52 μSv/h

Figure 6.1: *X-ray and gamma probe.*

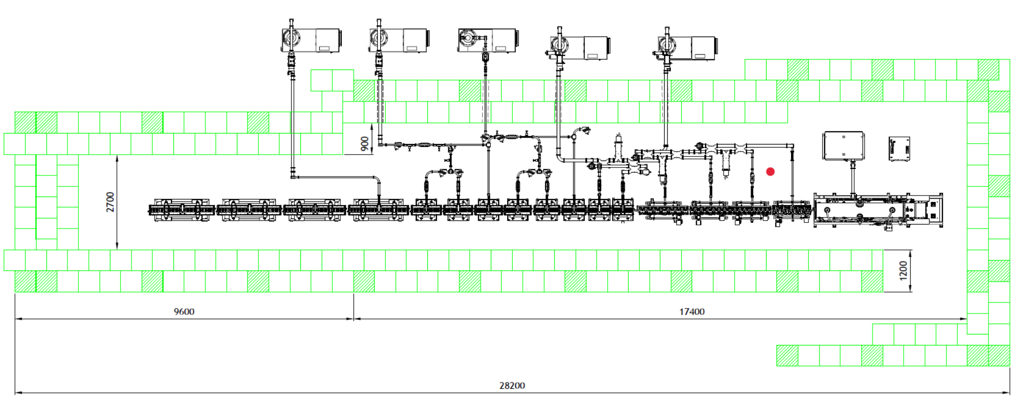

Figure 6.2: *Model shows the position of gamma and neutron detectors (red point), at 1 meter from beam-pipe and between Module 0 and Module 1.*

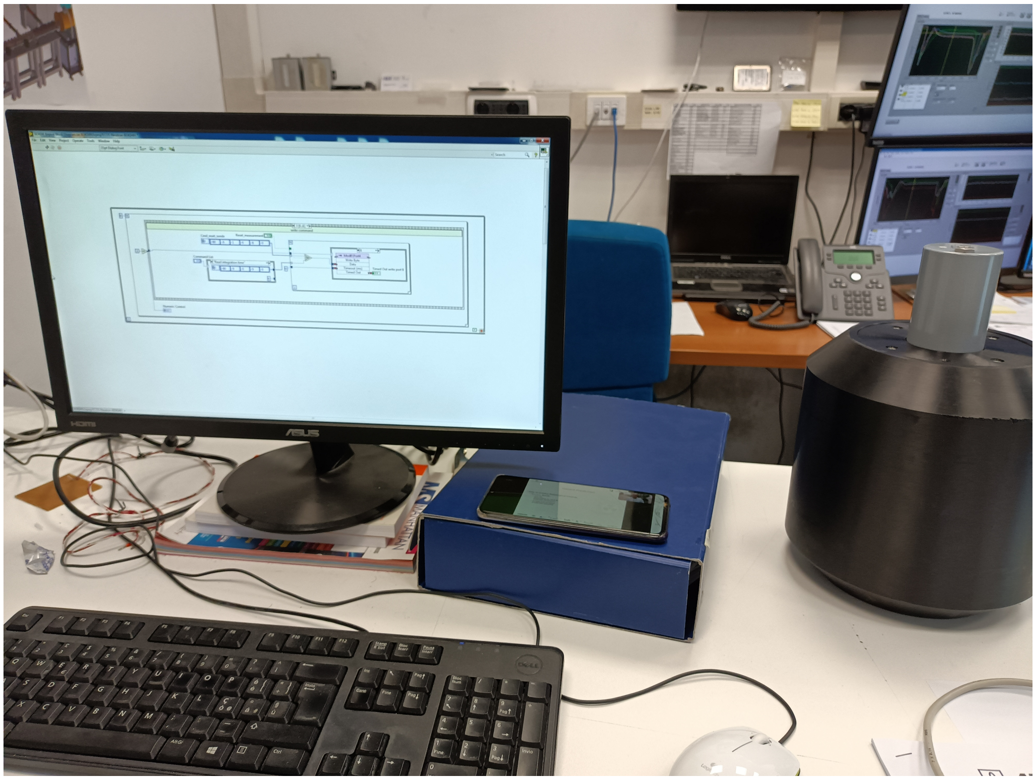

Figure 6.3: *Atomtek neutron probe and Labview interface with RS232 protocol.*

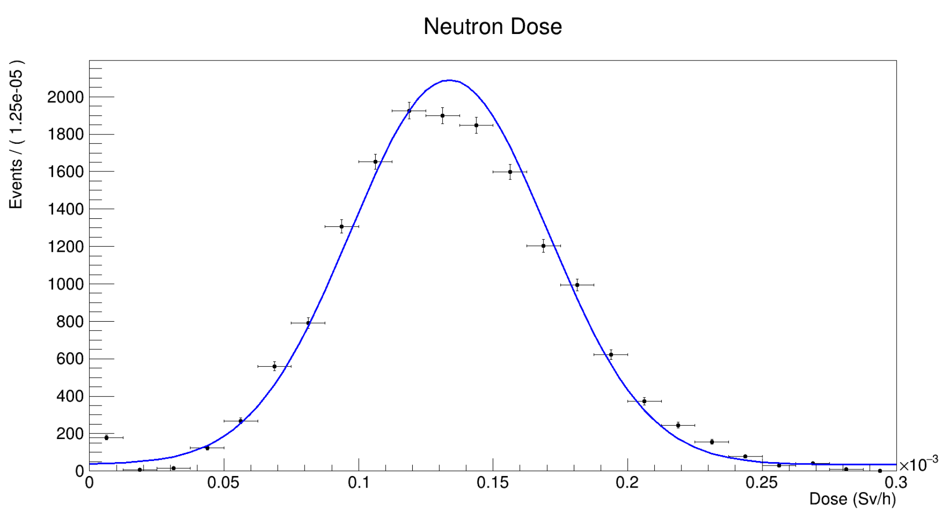

Figure 6.4: *Neutron dose rate for an injection current of 0.6 mA and 10 Hz of repetition rate.*

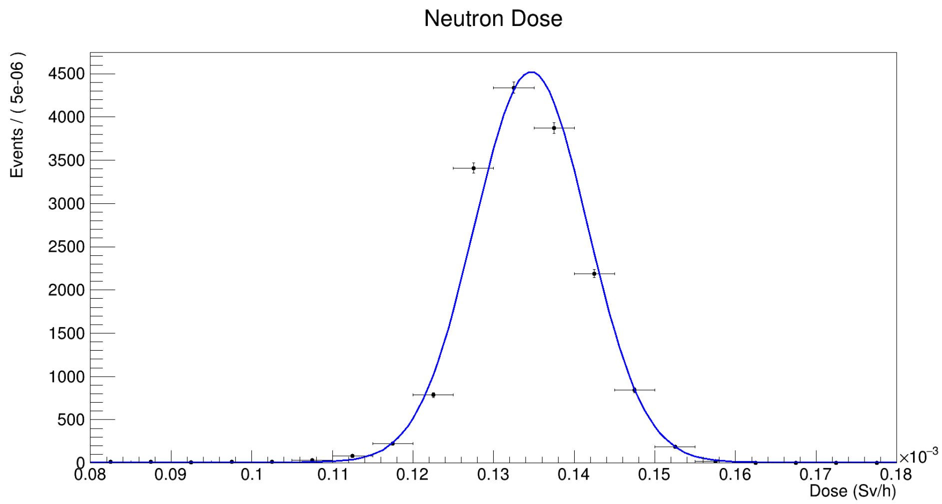

Figure 6.5: *Neutron dose rate for an injection current of 0.6 mA and 10 Hz of repetition rate, in floating mean acquisition mode.*

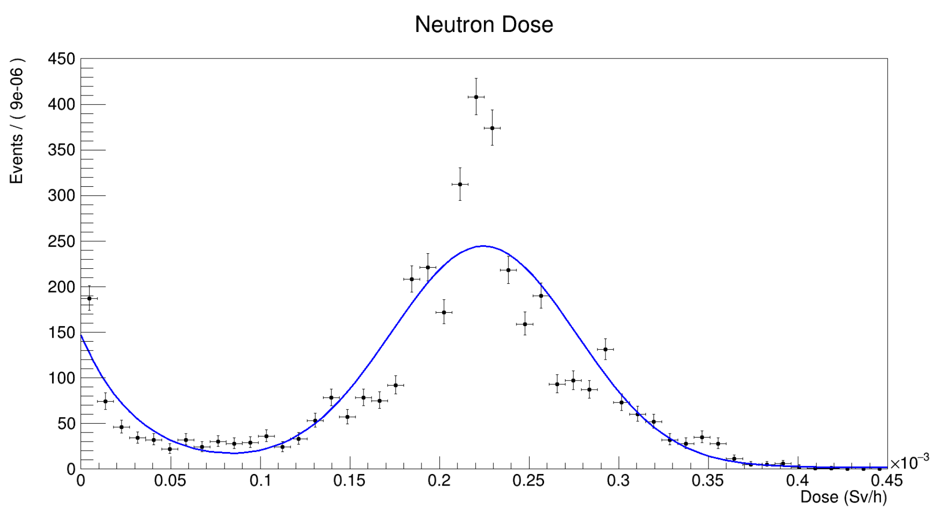

Figure 6.6: *Neutron dose rate for an injection current of 1.2 mA and 10 Hz of repetition rate. Fit model is a Gaussian for signal, a Uniform for background and a Exponential for low rate events.*

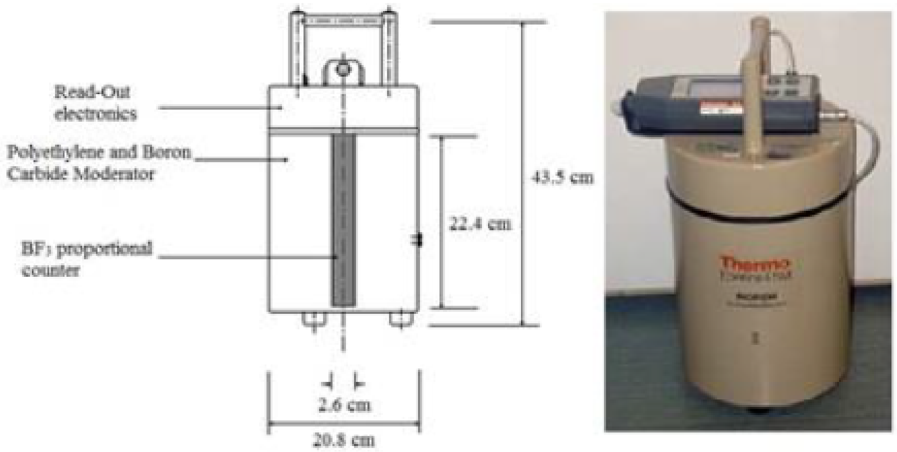

Figure 6.7: *Geometry structure to the left, a picture to the right.*

Table 6.5: *Thermo-Scientific BIOREM FHT 752.*

Detector	BF_3 proportional counter
Dose equivalent rate range	0,0001μSv/h - 400 mSv/h
Instrinsic measurement error	$\pm$ 20%
Energy range	0,025 eV - 10 MeV
Sensitivity to ^{252}Cf (dose mode)	0.56 cps for 1μSv/h
Protection class	IP64
Weight	11,5 kg
Dimensions	$\oslash$ 208x435 mm

Table 6.6: *Calculated expected dose rate values from the simulated ones.*

I (mA)	ν (Hz)	P/pulse	P/s	N/s	S. Dose (μSv/h)	E. Dose (μSv/h)
0.6	10	1,5 10^{10}	1,5 10^{11}	428	602,45	120,5
0.6	20	1,5 10^{10}	3 10^{11}	856	1204,90	241,0
0.6	30	1,5 10^{10}	4,5 10^{11}	1284	1807,35	361,5
1.2	10	3 10^{10}	3 10^{11}	856	1204,9	205,0
1.2	20	3 10^{10}	6 10^{11}	1712	2409,8	409,7
1.2	30	3 10^{10}	9 10^{11}	2568	3614,7	614,5

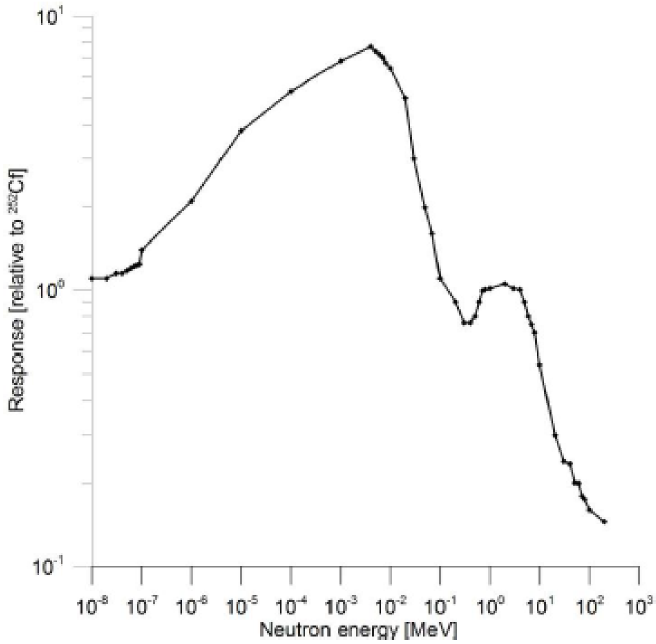

Figure 6.8: *Response function to neutrons of BIOREM, expressed in units relative to the response of moderated* ^{252}Cf *[34].*

Table 6.7: *Measured dose rate values from two Thermo BIOREM probes.*

I (mA)	ν (Hz)	BIOREM-1 (μSv/h)	BIOREM-2 (μSv/h)
0.6	10	44	50
0.6	20	85	98
0.6	30	120	150
1.2	10	140	200
1.2	20	280	400
1.2	30	420	600

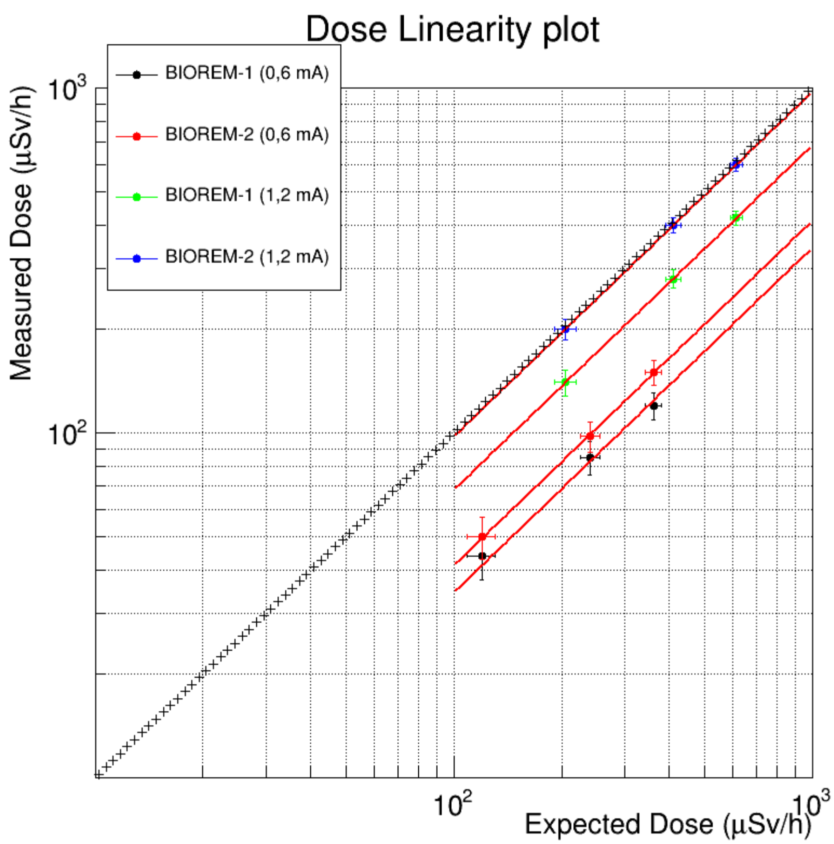

Figure 6.9: *Linearity plot for the BIOREM-1 and BIOREM-2 probes.*

Conclusion and Outlook

In order to resume this work, it is fundamental to say that simulation plays a central role in every situation in which there are radiation fields propagating where people have to be kept in safe. The reason is that particles such as neutrons behave in a way that is difficult to roughly estimate, and are responsible for the main part of the amount of dose absorbed by tissue. Shieldings are required to protect body and materials from damage, but they are very useful where there are steady-state field, so a radiation coming from nuclear decay. But in an accelerator facility there are other sources of neutrons and gamma, in general pulsed radiation fields, which are difficult to estimate. This work is focused to obtain a estimation of exposure dose due to the interaction between protons and materials around beam-pipe. Simulation is obtained with different approaches and than is compared with measured values of dose obtained with different probes. Results are very similar, if a factor that is used to get rid of undetected particles is applied, because of detector dead-time. Finally this work is useful to estabilish work conditions and verify measurements with probes. In order to go on with this project it could be useful to improve the simulation with a multithreading process and run it on a cluster for HPC. A further study on the proton beam and its interaction in air is of fundamental importance for an overall study of the dose released by the machine.

Appendix A

LXeAccelerator

GEANT4, C++ LXeAccelerator.cc class:

```
//
// ********************************************************************
// * License and Disclaimer                                           *
// *                                                                  *
// * The  Geant4 software  is  copyright of the Copyright Holders  of *
// * the Geant4 Collaboration.  It is provided  under  the terms  and *
// * conditions of the Geant4 Software License,  included in the file *
// * LICENSE and available at  http://cern.ch/geant4/license .  These *
// * include a list of copyright holders.                             *
// *                                                                  *
// * Neither the authors of this software system, nor their employing *
// * institutes,nor the agencies providing financial support for this *
// * work  make  any representation or  warranty, express or implied, *
// * regarding  this  software system or assume any liability for its *
// * use.  Please see the license in the file  LICENSE  and URL above *
// * for the full disclaimer and the limitation of liability.         *
// *                                                                  *
// * This  code  implementation is the result of  the  scientific and *
// * technical work of the GEANT4 collaboration.                      *
// * By using,  copying,  modifying or  distributing the software (or *
// * any work based  on the software)  you  agree  to acknowledge its *
// * use  in  resulting  scientific  publications,  and indicate your *
// * acceptance of all terms of the Geant4 Software license.          *
// ********************************************************************
//
//
/// \file optical/LXe/src/LXeMainVolume.cc
/// \brief Implementation of the LXeMainVolume class
//
//
#include "LXeAccelerator.hh"

#include "G4Box.hh"
#include "G4Colour.hh"
#include "G4LogicalSkinSurface.hh"
```

```
#include "G4LogicalBorderSurface.hh"
#include "G4LogicalVolume.hh"
#include "G4Material.hh"
#include "G4MaterialPropertiesTable.hh"
#include "G4OpticalSurface.hh"
#include "G4Sphere.hh"
#include "G4SystemOfUnits.hh"
#include "G4Tubs.hh"
#include "G4VisAttributes.hh"
#include "ElectricFieldSetup.hh"
#include "F03FieldSetup.hh"
#include "G4TransportationManager.hh"
#include "G4UniformMagField.hh"
#include "G4FieldManager.hh"
#include "G4RunManager.hh"
#include "G4NistManager.hh"
#include "G4Cons.hh"
#include "G4Orb.hh"
#include "G4Trd.hh"
#include "G4PVPlacement.hh"
#include "G4AutoDelete.hh"
#include "G4AssemblyVolume.hh"
#include "G4Types.hh"

//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo......

LXeAccelerator::LXeAccelerator(G4RotationMatrix* pRot, const G4ThreeVector& tlate,
G4LogicalVolume* pMotherLogical, G4bool pMany,
G4int pCopyNo, LXeDetectorConstruction* c)
// Pass info to the G4PVPlacement constructor
: G4PVPlacement(pRot, tlate,
// Temp logical volume must be created here
new G4LogicalVolume(new G4Box("temp2", 1.*cm, 1.*cm, 1.*cm),
G4Material::GetMaterial("Vacuum"), "temp2",
0, 0, 0),
"housing2", pMotherLogical, pMany, pCopyNo)
, fConstructor(c)
{
  // Get nist material manager
  G4NistManager* nist = G4NistManager::Instance();

  // Envelope parameters
  //
  G4double env_sizeXY = 20.*cm, env_sizeZ = (0.99+0.80)*m;
  G4Material* env_mat = G4Material::GetMaterial("Air");

  // Option to switch on/off checking of volumes overlaps
  //
  G4bool checkOverlaps = true;

```

```
//
// Envelope
//
G4Box* solidEnv =
new G4Box("Envelope",                     //its name
0.5*env_sizeXY, 0.5*env_sizeXY, 0.5*env_sizeZ); //its size

G4LogicalVolume* logicEnv =
new G4LogicalVolume(solidEnv,             //its solid
env_mat,              //its material
"Envelope");          //its name


//Material - Air vacuum
G4Material* vacuum = G4Material::GetMaterial("Vacuum");


//
// Shape 0 - Quadrupole 1
//
G4Material* shape0_mat = nist->FindOrBuildMaterial("G4_Al");
G4double shape0_rmin =  3.*mm, shape0_rmax = 1.*cm;
G4double shape0_hz = 1.*cm;
G4double shape0_phimin = 0.*deg, shape0_phimax = 360.*deg;
G4ThreeVector pos0 = G4ThreeVector(0*cm, 0*cm, -20.*cm);

G4Tubs* solidShape0 =
new G4Tubs("Shape0",                      //its name
shape0_rmin, shape0_rmax, shape0_hz, shape0_phimin, shape0_phimax);//its size

G4LogicalVolume* logicShape0 =
new G4LogicalVolume(solidShape0,          //its solid
shape0_mat,           //its material
"Shape0");            //its name

new G4PVPlacement(0,                        //no rotation
pos0,                   //at position
logicShape0,            //its logical volume
"Shape0",               //its name
logicEnv,               //its mother  volume
false,                  //no boolean operation
0,                      //copy number
checkOverlaps);         //overlaps checking


// Shape 1 - Beam pipe 1

//Position
G4ThreeVector pos1 = pos0;

G4double shape1_rmin =  0.*mm, shape1_rmax = 3.*mm;
G4double shape1_hz = 1.*cm;
```

```
G4double shape1_phimin = 0.*deg, shape1_phimax = 360.*deg;
G4Tubs* solidShape1 =
new G4Tubs("Shape1",
shape1_rmin, shape1_rmax, shape1_hz, shape1_phimin, shape1_phimax);

flogicShape1 =
new G4LogicalVolume(solidShape1,         //its solid
vacuum,               //its material
"Shape1");            //its name

new G4PVPlacement(0,                         //no rotation
pos1,                     //at position
flogicShape1,              //its logical volume
"Shape1",                 //its name
logicEnv,                 //its mother  volume
false,                    //no boolean operation
0,                        //copy number
checkOverlaps);           //overlaps checking


//
// Beam pipe envelope
//
G4Material* shape_mat = nist->FindOrBuildMaterial("G4_STAINLESS-STEEL");
G4double shape_rmin =  3.*mm, shape_rmax = 3.2*mm;
G4double shape_hz = 3.25*cm;
G4double shape_phimin = 0.*deg, shape_phimax = 360.*deg;

//Position
G4ThreeVector pos = G4ThreeVector(0*cm, 0*cm, -20.*cm + shape1_hz + shape_hz)
  ;

G4Tubs* solidShape =
new G4Tubs("Shape",
shape_rmin, shape_rmax, shape_hz, shape_phimin, shape_phimax);

G4LogicalVolume* logicShape =
new G4LogicalVolume(solidShape,          //its solid
shape_mat,            //its material
"Shape");             //its name

new G4PVPlacement(0,                         //no rotation
pos,                      //at position
logicShape,               //its logical volume
"Shape",                  //its name
logicEnv,                 //its mother  volume
false,                    //no boolean operation
0,                        //copy number
checkOverlaps             //overlaps checking
);

//
```

```
// Shape 2 - Beam pipe 2
//
G4double shape2_rmin =  0.*mm, shape2_rmax = 3.*mm;
G4double shape2_hz = 3.25*cm;
G4double shape2_phimin = 0.*deg, shape2_phimax = 360.*deg;

//Position
G4ThreeVector pos2 = G4ThreeVector(0*cm, 0*cm, -20.*cm + shape1_hz +
  shape2_hz);

G4Tubs* solidShape2 =
new G4Tubs("Shape2",
shape2_rmin, shape2_rmax, shape2_hz, shape2_phimin, shape2_phimax);

G4LogicalVolume* logicShape2 =
new G4LogicalVolume(solidShape2,         //its solid
vacuum,                //its material
"Shape2");             //its name

new G4PVPlacement(0,                           //no rotation
pos2,                  //at position
logicShape2,           //its logical volume
"Shape2",              //its name
logicEnv,              //its mother  volume
false,                 //no boolean operation
0,                     //copy number
checkOverlaps          //overlaps checking
);

//Add air layer
G4double layer = 0.*mm;
//
// Shape 3 - Quadrupole 2
//
G4double shape3_rmin =  3.*mm, shape3_rmax = 1.*cm;
G4double shape3_hz = 1.*cm;
G4double shape3_phimin = 0.*deg, shape3_phimax = 360.*deg;
G4ThreeVector pos3 = G4ThreeVector(0*cm, 0*cm, -20.*cm + shape1_hz + 2*
  shape2_hz + layer + shape3_hz);
G4Tubs* solidShape3 =
new G4Tubs("Shape3",
shape3_rmin, shape3_rmax, shape3_hz, shape3_phimin, shape3_phimax);

G4LogicalVolume* logicShape3 =
new G4LogicalVolume(solidShape3,         //its solid
shape0_mat,            //its material
"Shape3");             //its name

new G4PVPlacement(0,                           //no rotation
pos3,                  //at position
logicShape3,           //its logical volume
"Shape3",              //its name
```

```
logicEnv,                //its mother  volume
false,                   //no boolean operation
0,                       //copy number
checkOverlaps);          //overlaps checking
//
// Shape 4 - Beam pipe 3
//
G4double shape4_rmin =  0.*mm, shape4_rmax = 3.*mm;
G4double shape4_hz = 1.*cm;
G4double shape4_phimin = 0.*deg, shape4_phimax = 360.*deg;
G4ThreeVector pos4 = G4ThreeVector(0*cm, 0*cm, -20.*cm + shape1_hz + 2*shape2_hz + layer + shape3_hz);
G4Tubs* solidShape4 =
new G4Tubs("Shape4",
shape4_rmin, shape4_rmax, shape4_hz, shape4_phimin, shape4_phimax);

flogicShape4 =
new G4LogicalVolume(solidShape4,         //its solid
vacuum,               //its material
"Shape4");            //its name

new G4PVPlacement(0,                         //no rotation
pos4,                    //at position
flogicShape4,             //its logical volume
"Shape4",                //its name
logicEnv,                //its mother  volume
false,                   //no boolean operation
0,                       //copy number
checkOverlaps);          //overlaps checking


//
// Shape 5 - Module 0
//
G4Material* shape5_mat = nist->FindOrBuildMaterial("G4_Cu");
G4double shape5_rmin =  1.5*mm, shape5_rmax = 3.*cm;
G4double shape5_hz = 3.*cm;
G4double shape5_phimin = 0.*deg, shape5_phimax = 360.*deg;
G4ThreeVector pos5 = G4ThreeVector(0*cm, 0*cm, -20.*cm + shape1_hz + 2*shape2_hz + layer + 2*shape3_hz + layer + shape5_hz );
G4Tubs* solidShape5 =
new G4Tubs("Shape5",                     //its name
shape5_rmin, shape5_rmax, shape5_hz, shape5_phimin, shape5_phimax);//its size

G4LogicalVolume* logicShape5 =
new G4LogicalVolume(solidShape5,         //its solid
shape5_mat,           //its material
"Shape5");            //its name

new G4PVPlacement(0,                         //no rotation
pos5,                    //at position
logicShape5,             //its logical volume
```

```
"Shape5",                    //its name
logicEnv,                    //its mother  volume
false,                       //no boolean operation
0,                           //copy number
checkOverlaps);              //overlaps checking
//
// Shape 6 - Beam pipe 4
//
G4double shape6_rmin =  0.*mm, shape6_rmax = 1.5*mm;
G4double shape6_hz = 3.*cm;
G4double shape6_phimin = 0.*deg, shape6_phimax = 360.*deg;
G4ThreeVector pos6 = G4ThreeVector(0*cm, 0*cm, -20.*cm + shape1_hz + 2*
  shape2_hz + layer + 2*shape3_hz + layer + shape5_hz);
G4Tubs* solidShape6 =
new G4Tubs("Shape6",                          //its name
shape6_rmin, shape6_rmax, shape6_hz, shape6_phimin, shape6_phimax);//its size

flogicShape6 =
new G4LogicalVolume(solidShape6,          //its solid
vacuum,                 //its material
"Shape6");              //its name

new G4PVPlacement(0,                           //no rotation
pos6,                        //at position
flogicShape6,                 //its logical volume
"Shape6",                    //its name
logicEnv,                    //its mother  volume
false,                       //no boolean operation
0,                           //copy number
checkOverlaps);              //overlaps checking


//
// Beam pipe between Module 0 and Quadrupole 1
//
//
// Beam pipe envelope
//
G4Material* beampipe1_mat = nist->FindOrBuildMaterial("G4_STAINLESS-STEEL");
G4double beampipe1_rmin =  3.*mm, beampipe1_rmax = 3.2*mm;
G4double beampipe1_hz = 0.5*cm;
G4double beampipe1_phimin = 0.*deg, beampipe1_phimax = 360.*deg;

//Position
G4ThreeVector beampipe1_pos = G4ThreeVector(0*cm, 0*cm, -20.*cm + shape1_hz +
   2*shape2_hz + layer + 2*shape3_hz + layer + 2*shape5_hz + beampipe1_hz);

G4Tubs* beampipe1_solid =
new G4Tubs("beampipe1solid",
beampipe1_rmin, beampipe1_rmax, beampipe1_hz, beampipe1_phimin,
  beampipe1_phimax);

```

```
G4LogicalVolume* beampipe1_logic =
new G4LogicalVolume(beampipe1_solid,     //its solid
beampipe1_mat,       //its material
"beampipe1logic");   //its name

new G4PVPlacement(0,                          //no rotation
beampipe1_pos,            //at position
beampipe1_logic,          //its logical volume
"beampipe1physical",      //its name
logicEnv,                 //its mother  volume
false,                    //no boolean operation
0,                        //copy number
checkOverlaps             //overlaps checking
);

//
// Vacuum inside beampipe
//
G4double vacuum1_rmin =  0.*mm, vacuum1_rmax = 3.*mm;
G4double vacuum1_hz = 0.5*cm;
G4double vacuum1_phimin = 0.*deg, vacuum1_phimax = 360.*deg;

//Position
G4ThreeVector vacuum1_pos = G4ThreeVector(0*cm, 0*cm, -20.*cm + shape1_hz +
  2*shape2_hz + layer + 2*shape3_hz + layer + 2*shape5_hz + vacuum1_hz);

G4Tubs* vacuum1_solid =
new G4Tubs("vacuum1solid",
vacuum1_rmin, vacuum1_rmax, vacuum1_hz, vacuum1_phimin, vacuum1_phimax);

G4LogicalVolume* vacuum1_logic =
new G4LogicalVolume(vacuum1_solid,       //its solid
vacuum,              //its material
"vacuum1logic");     //its name

new G4PVPlacement(0,                          //no rotation
vacuum1_pos,              //at position
vacuum1_logic,            //its logical volume
"vacuum1physical",        //its name
logicEnv,                 //its mother  volume
false,                    //no boolean operation
0,                        //copy number
checkOverlaps             //overlaps checking
);

//
// Shape 7 - Quadrupole 3
//
G4double shape7_rmin =  3.*mm, shape7_rmax = 1.*cm;
G4double shape7_hz = 1.*cm;
G4double shape7_phimin = 0.*deg, shape7_phimax = 360.*deg;
G4ThreeVector pos7 = G4ThreeVector(0*cm, 0*cm, -20.*cm + shape1_hz + 2*
```

```
    shape2_hz + layer + 2*shape3_hz + layer + 2*shape5_hz + 2*vacuum1_hz +
    shape7_hz);

  G4Tubs* solidShape7 =
  new G4Tubs("Shape7",                        //its name
  shape7_rmin, shape7_rmax, shape7_hz, shape7_phimin, shape7_phimax);//its size

  G4LogicalVolume* logicShape7 =
  new G4LogicalVolume(solidShape7,            //its solid
  shape0_mat,           //its material
  "Shape7");            //its name

  new G4PVPlacement(0,                          //no rotation
  pos7,                     //at position
  logicShape7,              //its logical volume
  "Shape7",                 //its name
  logicEnv,                 //its mother  volume
  false,                    //no boolean operation
  0,                        //copy number
  checkOverlaps);           //overlaps checking

  //
  // Shape 8 - Beam pipe 5
  //
  G4double shape8_rmin =  0.*mm, shape8_rmax = 3.*mm;
  G4double shape8_hz = 1.*cm;
  G4double shape8_phimin = 0.*deg, shape8_phimax = 360.*deg;
  G4ThreeVector pos8 = G4ThreeVector(0*cm, 0*cm, -20.*cm + shape1_hz + 2*
    shape2_hz + layer + 2*shape3_hz + layer + 2*shape5_hz + 2*vacuum1_hz +
    shape7_hz);

  G4Tubs* solidShape8 =
  new G4Tubs("Shape8",                        //its name
  shape8_rmin, shape8_rmax, shape8_hz, shape8_phimin, shape8_phimax);//its size

  flogicShape8 =
  new G4LogicalVolume(solidShape8,            //its solid
  vacuum,               //its material
  "Shape8");            //its name

  new G4PVPlacement(0,                          //no rotation
  pos8,                     //at position
  flogicShape8,              //its logical volume
  "Shape8",                 //its name
  logicEnv,                 //its mother  volume
  false,                    //no boolean operation
  0,                        //copy number
  checkOverlaps);           //overlaps checking

  //
  // Beam pipe between Quadrupole 3 and Module 1
  //
```

```
//
// Beam pipe envelope
//
G4Material* beampipe2_mat = nist->FindOrBuildMaterial("G4_STAINLESS-STEEL");
G4double beampipe2_rmin =  3.*mm, beampipe2_rmax = 3.2*mm;
G4double beampipe2_hz = 0.5*cm;
G4double beampipe2_phimin = 0.*deg, beampipe2_phimax = 360.*deg;

//Position
G4ThreeVector beampipe2_pos = G4ThreeVector(0*cm, 0*cm, -20.*cm + shape1_hz +
   2*shape2_hz + layer + 2*shape3_hz + layer + 2*shape5_hz + 2*vacuum1_hz +
  2*shape7_hz + beampipe2_hz);

G4Tubs* beampipe2_solid =
new G4Tubs("beampipe2solid",
beampipe2_rmin, beampipe2_rmax, beampipe2_hz, beampipe2_phimin,
  beampipe2_phimax);

G4LogicalVolume* beampipe2_logic =
new G4LogicalVolume(beampipe2_solid,     //its solid
beampipe2_mat,       //its material
"beampipe2logic");   //its name

new G4PVPlacement(0,                         //no rotation
beampipe2_pos,           //at position
beampipe2_logic,         //its logical volume
"beampipe2physical",     //its name
logicEnv,                //its mother  volume
false,                   //no boolean operation
0,                       //copy number
checkOverlaps            //overlaps checking
);

//
// Vacuum inside beampipe
//
G4double vacuum2_rmin =  0.*mm, vacuum2_rmax = 3.*mm;
G4double vacuum2_hz = 0.5*cm;
G4double vacuum2_phimin = 0.*deg, vacuum2_phimax = 360.*deg;

//Position
G4ThreeVector vacuum2_pos = G4ThreeVector(0*cm, 0*cm, -20.*cm + shape1_hz +
  2*shape2_hz + layer + 2*shape3_hz + layer + 2*shape5_hz + 2*vacuum1_hz + 2*
  shape7_hz + vacuum2_hz);

G4Tubs* vacuum2_solid =
new G4Tubs("vacuum2solid",
vacuum2_rmin, vacuum2_rmax, vacuum2_hz, vacuum2_phimin, vacuum2_phimax);

G4LogicalVolume* vacuum2_logic =
new G4LogicalVolume(vacuum2_solid,       //its solid
vacuum,              //its material
```

```
"vacuum2logic");      //its name

new G4PVPlacement(0,                          //no rotation
vacuum2_pos,                //at position
vacuum2_logic,              //its logical volume
"vacuum2physical",          //its name
logicEnv,                   //its mother  volume
false,                      //no boolean operation
0,                          //copy number
checkOverlaps               //overlaps checking
);

//                               //
// -- Modulo 1 --- Quadrupole 3 //
//                               //

//
// Shape 9 - Module 1
//
G4Material* shape9_mat = nist->FindOrBuildMaterial("G4_Cu");
G4double shape9_rmin =  1.5*mm, shape9_rmax = 3.*cm;
G4double shape9_hz = 3.*cm;
G4double shape9_phimin = 0.*deg, shape9_phimax = 360.*deg;
G4ThreeVector pos9 = G4ThreeVector(0*cm, 0*cm, -20.*cm + shape1_hz + 2*
  shape2_hz + layer + 2*shape3_hz + layer + 2*shape5_hz + 2*vacuum1_hz + 2*
  shape7_hz + 2*vacuum2_hz + shape9_hz);
G4Tubs* solidShape9 =
new G4Tubs("Shape9",                          //its name
shape9_rmin, shape9_rmax, shape9_hz, shape9_phimin, shape9_phimax);//its size

G4LogicalVolume* logicShape9 =
new G4LogicalVolume(solidShape9,          //its solid
shape9_mat,           //its material
"Shape9");            //its name

new G4PVPlacement(0,                          //no rotation
pos9,                       //at position
logicShape9,                //its logical volume
"Shape9",                   //its name
logicEnv,                   //its mother  volume
false,                      //no boolean operation
0,                          //copy number
checkOverlaps);             //overlaps checking
//
// Shape 10 - Beam pipe 6
//
G4double shape10_rmin =  0.*mm, shape10_rmax = 1.5*mm;
G4double shape10_hz = 3.*cm;
G4double shape10_phimin = 0.*deg, shape10_phimax = 360.*deg;
G4ThreeVector pos10 = G4ThreeVector(0*cm, 0*cm, -20.*cm + shape1_hz + 2*
  shape2_hz + layer + 2*shape3_hz + layer + 2*shape5_hz + 2*vacuum1_hz + 2*
  shape7_hz + 2*vacuum2_hz + shape9_hz);
```

```
G4Tubs* solidShape10 =
new G4Tubs("Shape10",                   //its name
shape10_rmin, shape10_rmax, shape10_hz, shape10_phimin, shape10_phimax);//its size

flogicShape10 =
new G4LogicalVolume(solidShape10,       //its solid
vacuum,              //its material
"Shape10");          //its name

new G4PVPlacement(0,                    //no rotation
pos10,                  //at position
flogicShape10,           //its logical volume
"Shape10",              //its name
logicEnv,               //its mother  volume
false,                  //no boolean operation
0,                      //copy number
checkOverlaps);         //overlaps checking


//
// Beam pipe between Module 1 and Quadrupole 4
//
//
// Beam pipe envelope
//
G4Material* beampipe3_mat = nist->FindOrBuildMaterial("G4_STAINLESS-STEEL");
G4double beampipe3_rmin =  3.*mm, beampipe3_rmax = 3.2*mm;
G4double beampipe3_hz = 0.5*cm;
G4double beampipe3_phimin = 0.*deg, beampipe3_phimax = 360.*deg;

//Position
G4ThreeVector beampipe3_pos = G4ThreeVector(0*cm, 0*cm, -20.*cm + shape1_hz + 2*shape2_hz + layer + 2*shape3_hz + layer + 2*shape5_hz + 2*vacuum1_hz + 2*shape7_hz + 2*beampipe2_hz + 2*shape9_hz + beampipe3_hz);

G4Tubs* beampipe3_solid =
new G4Tubs("beampipe3solid",
beampipe3_rmin, beampipe3_rmax, beampipe3_hz, beampipe3_phimin, beampipe3_phimax);

G4LogicalVolume* beampipe3_logic =
new G4LogicalVolume(beampipe3_solid,     //its solid
beampipe3_mat,       //its material
"beampipe3logic");   //its name

new G4PVPlacement(0,                    //no rotation
beampipe3_pos,          //at position
beampipe3_logic,        //its logical volume
"beampipe3physical",    //its name
logicEnv,               //its mother  volume
false,                  //no boolean operation
```

```
0,                         //copy number
checkOverlaps              //overlaps checking
);

//
// Vacuum inside beampipe
//
G4double vacuum3_rmin =  0.*mm, vacuum3_rmax = 3.*mm;
G4double vacuum3_hz = 0.5*cm;
G4double vacuum3_phimin = 0.*deg, vacuum3_phimax = 360.*deg;

//Position
G4ThreeVector vacuum3_pos = G4ThreeVector(0*cm, 0*cm, -20.*cm + shape1_hz +
  2*shape2_hz + layer + 2*shape3_hz + layer + 2*shape5_hz + 2*vacuum1_hz + 2*
  shape7_hz + 2*vacuum2_hz + 2*shape9_hz + vacuum3_hz);

G4Tubs* vacuum3_solid =
new G4Tubs("vacuum3solid",
vacuum3_rmin, vacuum3_rmax, vacuum3_hz, vacuum3_phimin, vacuum3_phimax);

G4LogicalVolume* vacuum3_logic =
new G4LogicalVolume(vacuum3_solid,        //its solid
vacuum,                 //its material
"vacuum3logic");        //its name

new G4PVPlacement(0,                          //no rotation
vacuum3_pos,               //at position
vacuum3_logic,             //its logical volume
"vacuum3physical",         //its name
logicEnv,                  //its mother  volume
false,                     //no boolean operation
0,                         //copy number
checkOverlaps              //overlaps checking
);

//
// Shape 11 - Quadrupole 4
//
G4double shape11_rmin =  3.*mm, shape11_rmax = 1.*cm;
G4double shape11_hz = 1.*cm;
G4double shape11_phimin = 0.*deg, shape11_phimax = 360.*deg;
G4ThreeVector pos11 = G4ThreeVector(0*cm, 0*cm, -20.*cm + shape1_hz + 2*
  shape2_hz + layer + 2*shape3_hz + layer + 2*shape5_hz + 2*vacuum1_hz + 2*
  shape7_hz + 2*vacuum2_hz + 2*shape9_hz + 2*vacuum3_hz + shape11_hz);

G4Tubs* solidShape11 =
new G4Tubs("Shape11",                     //its name
shape11_rmin, shape11_rmax, shape11_hz, shape11_phimin, shape11_phimax);//its
   size

G4LogicalVolume* logicShape11 =
new G4LogicalVolume(solidShape11,         //its solid
```

```
shape0_mat,           //its material
"Shape11");           //its name

new G4PVPlacement(0,                          //no rotation
pos11,                    //at position
logicShape11,             //its logical volume
"Shape11",                //its name
logicEnv,                 //its mother  volume
false,                    //no boolean operation
0,                        //copy number
checkOverlaps);           //overlaps checking

//
// Shape 12 - Beam pipe 7
//
G4double shape12_rmin =  0.*mm, shape12_rmax = 3.*mm;
G4double shape12_hz = 1.*cm;
G4double shape12_phimin = 0.*deg, shape12_phimax = 360.*deg;
G4ThreeVector pos12 = G4ThreeVector(0*cm, 0*cm, -20.*cm + shape1_hz + 2*
  shape2_hz + layer + 2*shape3_hz + layer + 2*shape5_hz + 2*vacuum1_hz + 2*
  shape7_hz + 2*vacuum2_hz + 2*shape9_hz + 2*vacuum3_hz + shape11_hz);

G4Tubs* solidShape12 =
new G4Tubs("Shape12",                        //its name
shape12_rmin, shape12_rmax, shape12_hz, shape12_phimin, shape12_phimax);//its
   size

flogicShape12 =
new G4LogicalVolume(solidShape12,         //its solid
vacuum,               //its material
"Shape12");           //its name

new G4PVPlacement(0,                          //no rotation
pos12,                    //at position
flogicShape12,             //its logical volume
"Shape12",                //its name
logicEnv,                 //its mother  volume
false,                    //no boolean operation
0,                        //copy number
checkOverlaps);           //overlaps checking

//
// Beam pipe between Quadrupole 4 and Module 2
//
//
// Beam pipe envelope
//
G4Material* beampipe4_mat = nist->FindOrBuildMaterial("G4_STAINLESS-STEEL");
G4double beampipe4_rmin =  3.*mm, beampipe4_rmax = 3.2*mm;
G4double beampipe4_hz = 0.5*cm;
G4double beampipe4_phimin = 0.*deg, beampipe4_phimax = 360.*deg;
```

```
//Position
G4ThreeVector beampipe4_pos = G4ThreeVector(0*cm, 0*cm, -20.*cm + shape1_hz +
   2*shape2_hz + layer + 2*shape3_hz + layer + 2*shape5_hz + 2*vacuum1_hz +
  2*shape7_hz + 2*beampipe2_hz + 2*shape9_hz + 2*beampipe3_hz + 2*shape11_hz
  + beampipe4_hz);

G4Tubs* beampipe4_solid =
new G4Tubs("beampipe4solid",
beampipe4_rmin, beampipe4_rmax, beampipe4_hz, beampipe4_phimin,
  beampipe4_phimax);

G4LogicalVolume* beampipe4_logic =
new G4LogicalVolume(beampipe4_solid,     //its solid
beampipe4_mat,        //its material
"beampipe4logic");    //its name

new G4PVPlacement(0,                         //no rotation
beampipe4_pos,            //at position
beampipe4_logic,          //its logical volume
"beampipe4physical",      //its name
logicEnv,                 //its mother  volume
false,                    //no boolean operation
0,                        //copy number
checkOverlaps             //overlaps checking
);

//
// Vacuum inside beampipe
//
G4double vacuum4_rmin =  0.*mm, vacuum4_rmax = 3.*mm;
G4double vacuum4_hz = 0.5*cm;
G4double vacuum4_phimin = 0.*deg, vacuum4_phimax = 360.*deg;

//Position
G4ThreeVector vacuum4_pos = G4ThreeVector(0*cm, 0*cm, -20.*cm + shape1_hz +
  2*shape2_hz + layer + 2*shape3_hz + layer + 2*shape5_hz + 2*vacuum1_hz + 2*
  shape7_hz + 2*vacuum2_hz + 2*shape9_hz + 2*vacuum3_hz + 2*shape11_hz +
  beampipe4_hz);

G4Tubs* vacuum4_solid =
new G4Tubs("vacuum4solid",
vacuum4_rmin, vacuum4_rmax, vacuum4_hz, vacuum4_phimin, vacuum4_phimax);

G4LogicalVolume* vacuum4_logic =
new G4LogicalVolume(vacuum4_solid,       //its solid
vacuum,               //its material
"vacuum4logic");      //its name

new G4PVPlacement(0,                         //no rotation
vacuum4_pos,              //at position
vacuum4_logic,            //its logical volume
"vacuum4physical",        //its name
```

```
    logicEnv,                 //its mother  volume
    false,                    //no boolean operation
    0,                        //copy number
    checkOverlaps             //overlaps checking
    );


  //
  // Construct Assembled Volume
  //
  G4AssemblyVolume* assembly = new G4AssemblyVolume();

  assembly->AddPlacedVolume(logicShape5,     pos5,          0);
  assembly->AddPlacedVolume(flogicShape6,     pos6,          0);
  assembly->AddPlacedVolume(beampipe1_logic, beampipe1_pos, 0);
  assembly->AddPlacedVolume(vacuum1_logic,   vacuum1_pos,   0);
  assembly->AddPlacedVolume(logicShape7,     pos7,          0);
  assembly->AddPlacedVolume(flogicShape8,     pos8,          0);
  assembly->AddPlacedVolume(beampipe2_logic, beampipe2_pos, 0);
  assembly->AddPlacedVolume(vacuum2_logic,   vacuum2_pos,   0);
  assembly->AddPlacedVolume(logicShape9,     pos9,          0);
  assembly->AddPlacedVolume(flogicShape10,    pos10,         0);
  assembly->AddPlacedVolume(beampipe3_logic, beampipe3_pos, 0);
  assembly->AddPlacedVolume(vacuum3_logic,   vacuum3_pos,   0);
  assembly->AddPlacedVolume(logicShape11,    pos11,         0);
  assembly->AddPlacedVolume(flogicShape12,    pos12,         0);
  assembly->AddPlacedVolume(beampipe4_logic, beampipe4_pos, 0);
  assembly->AddPlacedVolume(vacuum4_logic,   vacuum4_pos,   0);

  G4double z_pos = 2*shape5_hz + 2*vacuum1_hz + 2*shape7_hz + 2*vacuum2_hz + 2*
    shape9_hz + 2*vacuum3_hz + 2*shape11_hz + 2*beampipe4_hz;

  for (int i = 0; i < 4; i++)
  {
    G4ThreeVector position = G4ThreeVector(0*cm, 0*cm, z_pos*(i+1));
    assembly->MakeImprint(logicEnv, position, 0);
  }

  SetLogicalVolume(logicEnv);
}

//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo
    ......
```

Appendix B

LXeDetectorConstruction

GEANT4, C++ LXeDetectorConstruction.cc class:

```
//
// ********************************************************************
// * License and Disclaimer                                           *
// *                                                                  *
// * The  Geant4 software  is  copyright of the Copyright Holders  of *
// * the Geant4 Collaboration.  It is provided  under  the terms  and *
// * conditions of the Geant4 Software License,  included in the file *
// * LICENSE and available at  http://cern.ch/geant4/license .  These *
// * include a list of copyright holders.                             *
// *                                                                  *
// * Neither the authors of this software system, nor their employing *
// * institutes,nor the agencies providing financial support for this *
// * work  make  any representation or  warranty, express or implied, *
// * regarding  this  software system or assume any liability for its *
// * use.  Please see the license in the file  LICENSE  and URL above *
// * for the full disclaimer and the limitation of liability.         *
// *                                                                  *
// * This  code  implementation is the result of  the  scientific and *
// * technical work of the GEANT4 collaboration.                      *
// * By using,  copying,  modifying or  distributing the software (or *
// * any work based  on the software)  you  agree  to acknowledge its *
// * use  in  resulting  scientific  publications,  and indicate your *
// * acceptance of all terms of the Geant4 Software license.          *
// ********************************************************************
//
//
/// \file optical/LXe/src/LXeDetectorConstruction.cc
/// \brief Implementation of the LXeDetectorConstruction class
//
//
#include "LXeDetectorConstruction.hh"

#include "LXeDetectorMessenger.hh"
#include "LXeMainVolume.hh"
#include "LXePMTSD.hh"
```

```
#include "LXeScintSD.hh"
#include "LXeWLSSlab.hh"
#include "LXeAccelerator.hh"
#include "LXeModule1.hh"
#include "LXeModule2.hh"
#include "LXeModule3.hh"
#include "LXeModuleCCL.hh"

#include "G4Types.hh"
#include "ElectricFieldSetup.hh"
#include "F03FieldSetup.hh"

#include "LXeSteppingAction.hh"

#include "globals.hh"
#include "G4Box.hh"
#include "G4GeometryManager.hh"
#include "G4LogicalBorderSurface.hh"
#include "G4LogicalSkinSurface.hh"
#include "G4LogicalVolume.hh"
#include "G4LogicalVolumeStore.hh"
#include "G4Material.hh"
#include "G4MaterialTable.hh"
#include "G4NistManager.hh"
#include "G4OpticalSurface.hh"
#include "G4PhysicalConstants.hh"
#include "G4PhysicalVolumeStore.hh"
#include "G4PVPlacement.hh"
#include "G4RunManager.hh"
#include "G4SDManager.hh"
#include "G4SolidStore.hh"
#include "G4Sphere.hh"
#include "G4SystemOfUnits.hh"
#include "G4ThreeVector.hh"
#include "G4Tubs.hh"
#include "G4UImanager.hh"
#include "G4VisAttributes.hh"
#include "G4AutoDelete.hh"

using namespace CLHEP;

G4bool LXeDetectorConstruction::fSphereOn = true;

//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo......

LXeDetectorConstruction::LXeDetectorConstruction()
: fLXe_mt(nullptr)
, fMPTPStyrene(nullptr)
, fScoringVolume(nullptr)
, fMPTGlassSci(nullptr)
{
```

```
fExperimentalHall_box  = nullptr;
fExperimentalHall_log  = nullptr;
fExperimentalHall_phys = nullptr;

fLXe = fAl = fAir = fVacuum = fGlass = fGlassSci = nullptr;
fPstyrene = fPMMA = fPethylene1 = fPethylene2 = nullptr;
fConcrete = nullptr;

fN = fO = fC = fH = fLi6 = fCe = nullptr;
fAll = nullptr;
fNa = fMg = fSi = fK = fCa = fFe = fP = fS = fTi = fMn = fZn = fZr = fBa =
  fPb = fSr = nullptr;

fSaveThreshold = 0;
SetDefaults();

DefineMaterials();
fDetectorMessenger = new LXeDetectorMessenger(this);

felFieldSetup1_0 = 0;
felFieldSetup2_0 = 0;
femFieldSetup1_0 = 0;
femFieldSetup2_0 = 0;
femFieldSetup3_0 = 0;
femFieldSetup4_0 = 0;

felFieldSetup1_1 = 0;
felFieldSetup2_1 = 0;
femFieldSetup1_1 = 0;
femFieldSetup2_1 = 0;

felFieldSetup1_2 = 0;
felFieldSetup2_2 = 0;
femFieldSetup1_2 = 0;
femFieldSetup2_2 = 0;

felFieldSetup1_3 = 0;
felFieldSetup2_3 = 0;
femFieldSetup1_3 = 0;
femFieldSetup2_3 = 0;

felFieldSetup1_4 = 0;
felFieldSetup2_4 = 0;
femFieldSetup1_4 = 0;
femFieldSetup2_4 = 0;

felFieldSetup1_5 = 0;
felFieldSetup2_5 = 0;
femFieldSetup1_5 = 0;
femFieldSetup2_5 = 0;

felFieldSetup1_6 = 0;
```

```
  felFieldSetup2_6 = 0;
  femFieldSetup1_6 = 0;
  femFieldSetup2_6 = 0;

  felFieldSetup1_7 = 0;
  felFieldSetup2_7 = 0;
  femFieldSetup1_7 = 0;
  femFieldSetup2_7 = 0;
}

//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo......

LXeDetectorConstruction::~LXeDetectorConstruction()
{
  if(fMainVolume)
  {
    delete fMainVolume;
  }
  delete fLXe_mt;
  delete fDetectorMessenger;
  delete fMPTPStyrene;
  delete fScoringVolume;
  delete fMPTGlassSci;
}

//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo......

void LXeDetectorConstruction::DefineMaterials()
{
  G4double a;  // atomic mass
  G4double z;  // atomic number
  G4double density;

  G4int polyPMMA = 1;
  G4int nC_PMMA  = 3 + 2 * polyPMMA;
  G4int nH_PMMA  = 6 + 2 * polyPMMA;

  G4int polyeth = 1;
  G4int nC_eth  = 2 * polyeth;
  G4int nH_eth  = 4 * polyeth;

  //***Elements
  fH = new G4Element("H", "H", z = 1., a = 1.01 * g / mole);
  fC = new G4Element("C", "C", z = 6., a = 12.01 * g / mole);
  fN = new G4Element("N", "N", z = 7., a = 14.01 * g / mole);
  fO = new G4Element("O", "O", z = 8., a = 16.00 * g / mole);
  fLi6 = new G4Element("Li6", "Li6", z = 3., a = 6.00 * g / mole);
  fCe = new G4Element("Ce", "Ce", z = 58., a = 140.12 * g / mole);

  fAl1 = new G4Element("Al", "Al", z = 13., a = 26.98 * g / mole);
```

```
fNa = new G4Element("Na", "Na", z = 11., a = 22.99 * g / mole);
fMg = new G4Element("Mg", "Mg", z = 12., a = 24.30 * g / mole);
fSi = new G4Element("Si", "Si", z = 14., a = 28.08 * g / mole);
fK = new G4Element("K", "K", z = 19., a = 39.10 * g / mole);
fCa = new G4Element("Ca", "Ca", z = 20., a = 40.078 * g / mole);
fFe = new G4Element("Fe", "Fe", z = 26., a = 55.84 * g / mole);
fP = new G4Element("P", "P", z = 15., a = 30.97 * g / mole);
fS = new G4Element("S", "S", z = 16., a = 32.065 * g / mole);
fTi = new G4Element("Ti", "Ti", z = 22., a = 47.867 * g / mole);
fMn = new G4Element("Mn", "Mn", z = 25., a = 54.94 * g / mole);
fZn = new G4Element("Zn", "Zn", z = 30., a = 65.409 * g / mole);
fZr = new G4Element("Zr", "Zr", z = 40., a = 91.224 * g / mole);
fBa = new G4Element("Ba", "Ba", z = 56., a = 137.327 * g / mole);
fPb = new G4Element("Pb", "Pb", z = 82., a = 207.2 * g / mole);
fSr = new G4Element("Sr", "Sr", z = 38., a = 87.62 * g / mole);

//***Materials
// Liquid Xenon
fLXe = new G4Material("LXe", z = 54., a = 131.29 * g / mole,
density = 3.020 * g / cm3);
// Aluminum
fAl = new G4Material("Al", z = 13., a = 26.98 * g / mole,
density = 2.7 * g / cm3);
// Vacuum
fVacuum = new G4Material("Vacuum", density = universe_mean_density, 3,
  kStateGas,
0.1 * kelvin, 1.e-7 * pascal);
fVacuum->AddElement(fH, 1. * perCent);
fVacuum->AddElement(fN, 69. * perCent);
fVacuum->AddElement(fO, 30. * perCent);
// Air
fAir = new G4Material("Air", density = 1.29 * mg / cm3, 3);
fAir->AddElement(fH, 1. * perCent);
fAir->AddElement(fN, 69. * perCent);
fAir->AddElement(fO, 30. * perCent);
// Glass
fGlass = new G4Material("Glass", density = 1.032 * g / cm3, 2);
fGlass->AddElement(fC, 91.533 * perCent);
fGlass->AddElement(fH, 8.467 * perCent);
// Glass scintillator
fGlassSci = new G4Material("GlassSci", density = 2.40 * g / cm3, 4);
fGlassSci->AddElement(fC, 84.18 * perCent);
fGlassSci->AddElement(fH, 7.82 * perCent);
fGlassSci->AddElement(fLi6, 7.5 * perCent);
fGlassSci->AddElement(fCe, 0.5 * perCent);
// Polystyrene
fPstyrene = new G4Material("Polystyrene", density = 1.03 * g / cm3, 2);
fPstyrene->AddElement(fC, 8);
fPstyrene->AddElement(fH, 8);
// Fiber (PMMA)
fPMMA = new G4Material("PMMA", density = 1190. * kg / m3, 3);
```

```
fPMMA->AddElement(fH, nH_PMMA);
fPMMA->AddElement(fC, nC_PMMA);
fPMMA->AddElement(fO, 2);
// Cladding(polyethylene)
fPethylene1 = new G4Material("Pethylene1", density = 1200. * kg / m3, 2);
fPethylene1->AddElement(fH, nH_eth);
fPethylene1->AddElement(fC, nC_eth);
// Double cladding(flourinated polyethylene)
fPethylene2 = new G4Material("Pethylene2", density = 1400. * kg / m3, 2);
fPethylene2->AddElement(fH, nH_eth);
fPethylene2->AddElement(fC, nC_eth);
// Concrete composition
fConcrete = new G4Material("Concrete", density = 2.42 * g / cm3, 19);
fConcrete->AddElement(fO, 49.2875 * perCent);
fConcrete->AddElement(fC, 5.62 * perCent);
fConcrete->AddElement(fH, 0.6 * perCent);
fConcrete->AddElement(fNa, 0.453 * perCent);
fConcrete->AddElement(fMg, 0.663 * perCent);
fConcrete->AddElement(fAll, 2.063 * perCent);
fConcrete->AddElement(fSi, 18.867 * perCent);
fConcrete->AddElement(fK, 0.656 * perCent);
fConcrete->AddElement(fCa, 20.091 * perCent);
fConcrete->AddElement(fFe, 1.118 * perCent);
fConcrete->AddElement(fP, 0.048 * perCent);
fConcrete->AddElement(fS, 0.012 * perCent);
fConcrete->AddElement(fTi, 0.347 * perCent);
fConcrete->AddElement(fMn, 0.0387 * perCent);
fConcrete->AddElement(fZn, 0.0241 * perCent);
fConcrete->AddElement(fZr, 0.0074 * perCent);
fConcrete->AddElement(fBa, 0.0179 * perCent);
fConcrete->AddElement(fPb, 0.0464 * perCent);
fConcrete->AddElement(fSr, 0.04 * perCent);

//***Material properties tables

std::vector<G4double> lxe_Energy = { 7.0 * eV, 7.07 * eV, 7.14 * eV };

std::vector<G4double> lxe_SCINT = { 0.1, 1.0, 0.1 };
std::vector<G4double> lxe_RIND  = { 1.59, 1.57, 1.54 };
std::vector<G4double> lxe_ABSL  = { 35. * cm, 35. * cm, 35. * cm };
fLXe_mt = new G4MaterialPropertiesTable();
fLXe_mt->AddProperty("SCINTILLATIONCOMPONENT1", lxe_Energy, lxe_SCINT);
fLXe_mt->AddProperty("SCINTILLATIONCOMPONENT2", lxe_Energy, lxe_SCINT);
fLXe_mt->AddProperty("RINDEX", lxe_Energy, lxe_RIND);
fLXe_mt->AddProperty("ABSLENGTH", lxe_Energy, lxe_ABSL);
fLXe_mt->AddConstProperty("SCINTILLATIONYIELD", 12000. / MeV);
fLXe_mt->AddConstProperty("RESOLUTIONSCALE", 1.0);
fLXe_mt->AddConstProperty("SCINTILLATIONTIMECONSTANT1", 20. * ns);
fLXe_mt->AddConstProperty("SCINTILLATIONTIMECONSTANT2", 45. * ns);
fLXe_mt->AddConstProperty("SCINTILLATIONYIELD1", 1.0);
fLXe_mt->AddConstProperty("SCINTILLATIONYIELD2", 0.0);
fLXe->SetMaterialPropertiesTable(fLXe_mt);
```

```

// Set the Birks Constant for the LXe scintillator
fLXe->GetIonisation()->SetBirksConstant(0.126 * mm / MeV);

std::vector<G4double> glass_RIND      = { 1.49, 1.49, 1.49 };
std::vector<G4double> glass_AbsLength = { 420. * cm, 420. * cm, 420. * cm };
G4MaterialPropertiesTable* glass_mt    = new G4MaterialPropertiesTable();
glass_mt->AddProperty("ABSLENGTH", lxe_Energy, glass_AbsLength);
glass_mt->AddProperty("RINDEX", lxe_Energy, glass_RIND);
fGlass->SetMaterialPropertiesTable(glass_mt);

std::vector<G4double> vacuum_Energy  = { 2.0 * eV, 7.0 * eV, 7.14 * eV };
std::vector<G4double> vacuum_RIND    = { 1., 1., 1. };
G4MaterialPropertiesTable* vacuum_mt = new G4MaterialPropertiesTable();
vacuum_mt->AddProperty("RINDEX", vacuum_Energy, vacuum_RIND);
fVacuum->SetMaterialPropertiesTable(vacuum_mt);
fAir->SetMaterialPropertiesTable(vacuum_mt);  // Give air the same rindex

std::vector<G4double> wls_Energy = { 2.00 * eV, 2.87 * eV, 2.90 * eV,
  3.47 * eV };

std::vector<G4double> rIndexPstyrene = { 1.58, 1.58, 1.58, 1.58 };
std::vector<G4double> absorption1    = { 250. * cm, 250. * cm, 250. * cm,
  250. * cm };
std::vector<G4double> scintilFast    = { 0.0, 0.0, 1.0, 1.0 };
fMPTPStyrene = new G4MaterialPropertiesTable();
fMPTPStyrene->AddProperty("RINDEX", wls_Energy, rIndexPstyrene);
fMPTPStyrene->AddProperty("ABSLENGTH", wls_Energy, absorption1);
fMPTPStyrene->AddProperty("SCINTILLATIONCOMPONENT1", wls_Energy, scintilFast)
  ;
fMPTPStyrene->AddConstProperty("SCINTILLATIONYIELD", 10. / keV);
fMPTPStyrene->AddConstProperty("RESOLUTIONSCALE", 1.0);
fMPTPStyrene->AddConstProperty("SCINTILLATIONTIMECONSTANT1", 2.4 * ns);
fPstyrene->SetMaterialPropertiesTable(fMPTPStyrene);

// Set the Birks Constant for the Polystyrene scintillator
fPstyrene->GetIonisation()->SetBirksConstant(0.126 * mm / MeV);

//New Material
std::vector<G4double> rIndex_GlassSci = { 1.59, 1.59, 1.59, 1.59 };
std::vector<G4double> AbsLength_GlassSci    = { 420. * cm, 420. * cm, 420. *
  cm, 420. * cm };
std::vector<G4double> scintilFast_GlassSci    = { 0.0, 0.0, 1.0, 1.0 };
fMPTGlassSci = new G4MaterialPropertiesTable();
fMPTGlassSci->AddProperty("RINDEX", wls_Energy, rIndex_GlassSci);
fMPTGlassSci->AddProperty("ABSLENGTH", wls_Energy, AbsLength_GlassSci);
fMPTGlassSci->AddProperty("SCINTILLATIONCOMPONENT1", wls_Energy,
  scintilFast_GlassSci);
fMPTGlassSci->AddProperty("SCINTILLATIONCOMPONENT2", wls_Energy,
  scintilFast_GlassSci);
fMPTGlassSci->AddConstProperty("SCINTILLATIONYIELD", 10000. / MeV);
fMPTGlassSci->AddConstProperty("RESOLUTIONSCALE", 1.0);
```

```
  fMPTGlassSci->AddConstProperty("SCINTILLATIONTIMECONSTANT1", 18. * ns);
  fMPTGlassSci->AddConstProperty("SCINTILLATIONTIMECONSTANT2", 45. * ns);
  fMPTGlassSci->AddConstProperty("SCINTILLATIONYIELD1", 1.);
  fMPTGlassSci->AddConstProperty("SCINTILLATIONYIELD2", 0.);
  fGlassSci->SetMaterialPropertiesTable(fMPTGlassSci);

  // Set the Birks Constant for the GlassSci scintillator
  fGlassSci->GetIonisation()->SetBirksConstant(0.126 * mm / MeV);


  std::vector<G4double> RefractiveIndexFiber = { 1.6, 1.6, 1.6, 1.6 };
  std::vector<G4double> AbsFiber     = { 9.0 * m, 9.0 * m, 0.1 * mm, 0.1 * mm };
  std::vector<G4double> EmissionFib = { 1.0, 1.0, 0.0, 0.0 };
  G4MaterialPropertiesTable* fiberProperty = new G4MaterialPropertiesTable();
  fiberProperty->AddProperty("RINDEX", wls_Energy, RefractiveIndexFiber);
  fiberProperty->AddProperty("WLSABSLENGTH", wls_Energy, AbsFiber);
  fiberProperty->AddProperty("WLSCOMPONENT", wls_Energy, EmissionFib);
  fiberProperty->AddConstProperty("WLSTIMECONSTANT", 0.5 * ns);
  fPMMA->SetMaterialPropertiesTable(fiberProperty);

  std::vector<G4double> RefractiveIndexClad1 = { 1.49, 1.49, 1.49, 1.49 };
  G4MaterialPropertiesTable* clad1Property    = new G4MaterialPropertiesTable();
  clad1Property->AddProperty("RINDEX", wls_Energy, RefractiveIndexClad1);
  clad1Property->AddProperty("ABSLENGTH", wls_Energy, AbsFiber);
  fPethylene1->SetMaterialPropertiesTable(clad1Property);

  std::vector<G4double> RefractiveIndexClad2 = { 1.42, 1.42, 1.42, 1.42 };
  G4MaterialPropertiesTable* clad2Property    = new G4MaterialPropertiesTable();
  clad2Property->AddProperty("RINDEX", wls_Energy, RefractiveIndexClad2);
  clad2Property->AddProperty("ABSLENGTH", wls_Energy, AbsFiber);
  fPethylene2->SetMaterialPropertiesTable(clad2Property);
}

//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo
    ......

G4VPhysicalVolume* LXeDetectorConstruction::Construct()
{
  // The experimental hall walls are all 1m away from housing walls
  G4double expHall_x = fScint_x + fD_mtl + 1.1 * m;
  G4double expHall_y = fScint_y + fD_mtl + 1.1 * m;
  G4double expHall_z = ((1.79 + 6. + 4.*1.31 + 0.67 + 3.2)/4.)*m;

  G4double wall_x = expHall_x + 1.20*m;
  G4double wall_y = expHall_y + 1.20*m;
  G4double wall_z = expHall_z + 1.20*m;

  //Create wall over experimental hall
  fWall_box =
  new G4Box("wall_box", wall_x, wall_y, wall_z);
  fWall_log =
  new G4LogicalVolume(fWall_box, fConcrete, "wall_log", 0, 0, 0);
```

```
fWall_phys =
new G4PVPlacement(0, G4ThreeVector(), fWall_log, "wall", 0, false, 0);

//Create experimental hall
fExperimentalHall_box =
new G4Box("expHall_box", expHall_x, expHall_y, expHall_z);
fExperimentalHall_log =
new G4LogicalVolume(fExperimentalHall_box, fAir, "expHall_log", 0, 0, 0);
fExperimentalHall_phys =
new G4PVPlacement(0, G4ThreeVector(), fExperimentalHall_log, "expHall",
  fWall_log, false, 0);

fExperimentalHall_log->SetVisAttributes(G4Colour(0.8, 0.8, 0.8));
fWall_log->SetVisAttributes(G4Colour(0.8, 0.8, 0.8));

// Place the main volume
if(fMainVolumeOn)
{
  fMainVolume = new LXeMainVolume(0, G4ThreeVector(1. *m, 0. *m , -6.60 *m +
  6.0*m), fExperimentalHall_log, false, 0, this);
}

fLXeAccelerator = static_cast<const LXeAccelerator*>(new LXeAccelerator(0,
  G4ThreeVector( 0. , 0. , -6.60*m + 5.0*m), fExperimentalHall_log, false, 0,
   this));

fLXeModule1 = static_cast<const LXeModule1*>(new LXeModule1(0, G4ThreeVector(
   0. , 0. , 0.395*m - 5.60*m + 5.0*m), fExperimentalHall_log, false, 0, this
  ));

fLXeModule2 = static_cast<const LXeModule2*>(new LXeModule2(0, G4ThreeVector(
   0. , 0. , 0.395*m - 4.6*m + 5.0*m), fExperimentalHall_log, false, 0, this)
  );

fLXeModule3 = static_cast<const LXeModule3*>(new LXeModule3(0, G4ThreeVector(
   0. , 0. , 1.395*m - 4.6*m + 5.0*m), fExperimentalHall_log, false, 0, this)
  );

fLXeModuleCCL1 = static_cast<const LXeModuleCCL*>(new LXeModuleCCL(0,
  G4ThreeVector( 0. , 0. , 2.2225*m - 4.6*m + 5.0*m), fExperimentalHall_log,
  false, 0, this));

fLXeModuleCCL2 = static_cast<const LXeModuleCCL*>(new LXeModuleCCL(0,
  G4ThreeVector( 0. , 0. , 2.8775*m - 4.6*m + 5.0*m), fExperimentalHall_log,
  false, 0, this));

fLXeModuleCCL3 = static_cast<const LXeModuleCCL*>(new LXeModuleCCL(0,
  G4ThreeVector( 0. , 0. , 3.5325*m - 4.6*m + 5.0*m), fExperimentalHall_log,
  false, 0, this));

fLXeModuleCCL4 = static_cast<const LXeModuleCCL*>(new LXeModuleCCL(0,
  G4ThreeVector( 0. , 0. , 4.1875*m - 4.6*m + 5.0*m), fExperimentalHall_log,
```

```
    false, 0, this));

  // Place the WLS slab
  if(fWLSslab)
  {
    G4VPhysicalVolume* slab = new LXeWLSSlab(
    0, G4ThreeVector(0., 0., -fScint_z / 2. - fSlab_z - 1. * cm),
    fExperimentalHall_log, false, 0, this);

    // Surface properties for the WLS slab
    G4OpticalSurface* scintWrap = new G4OpticalSurface("ScintWrap");

    new G4LogicalBorderSurface("ScintWrap", slab, fExperimentalHall_phys,
    scintWrap);

    scintWrap->SetType(dielectric_metal);
    scintWrap->SetFinish(polished);
    scintWrap->SetModel(glisur);

    std::vector<G4double> pp           = { 2.0 * eV, 3.5 * eV };
    std::vector<G4double> reflectivity = { 1.0, 1.0 };
    std::vector<G4double> efficiency   = { 0.0, 0.0 };

    G4MaterialPropertiesTable* scintWrapProperty =
    new G4MaterialPropertiesTable();

    scintWrapProperty->AddProperty("REFLECTIVITY", pp, reflectivity);
    scintWrapProperty->AddProperty("EFFICIENCY", pp, efficiency);
    scintWrap->SetMaterialPropertiesTable(scintWrapProperty);
  }

  fScoringVolume = fExperimentalHall_log;

  return fWall_phys;
}

//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo
    ......

void LXeDetectorConstruction::ConstructSDandField()
{
  if(!fMainVolume)
  return;

  // PMT SD

  LXePMTSD* pmt = fPmt_SD.Get();
  if(!pmt)
  {
    // Created here so it exists as pmts are being placed
    G4cout << "Construction /LXeDet/pmtSD" << G4endl;
    LXePMTSD* pmt_SD = new LXePMTSD("/LXeDet/pmtSD");
```

```
    fPmt_SD.Put(pmt_SD);

    pmt_SD->InitPMTs();
    pmt_SD->SetPmtPositions(fMainVolume->GetPmtPositions());
  }
  else
  {
    pmt->InitPMTs();
    pmt->SetPmtPositions(fMainVolume->GetPmtPositions());
  }
  G4SDManager::GetSDMpointer()->AddNewDetector(fPmt_SD.Get());
  // sensitive detector is not actually on the photocathode.
  // processHits gets done manually by the stepping action.
  // It is used to detect when photons hit and get absorbed & detected at the
  // boundary to the photocathode (which doesn't get done by attaching it to a
  // logical volume.
  // It does however need to be attached to something or else it doesn't get
  // reset at the begining of events

  SetSensitiveDetector(fMainVolume->GetLogPhotoCath(), fPmt_SD.Get());

  // Scint SD

  if(!fScint_SD.Get())
  {
    G4cout << "Construction /LXeDet/scintSD" << G4endl;
    LXeScintSD* scint_SD = new LXeScintSD("/LXeDet/scintSD");
    fScint_SD.Put(scint_SD);
  }
  G4SDManager::GetSDMpointer()->AddNewDetector(fScint_SD.Get());
  SetSensitiveDetector(fMainVolume->GetLogScint(), fScint_SD.Get());

  // LXeAccelerator
  if(!felFieldSetup1_0){
    static G4LogicalVolume* flogicShape6 = fLXeAccelerator->getlogicShape6();
    felFieldSetup1_0 = new ElectricFieldSetup();
    felFieldSetup1_0->SetLocalFieldValue(G4ThreeVector(0.,0.,(4.8*megavolt/m)))
    ;
    flogicShape6->SetFieldManager(felFieldSetup1_0->GetLocalFieldManager(),
    true );
    G4AutoDelete::Register(felFieldSetup1_0);
  }
  if(!felFieldSetup2_0){
    static G4LogicalVolume* flogicShape10 = fLXeAccelerator->getlogicShape10();
    felFieldSetup2_0 = new ElectricFieldSetup();
    felFieldSetup2_0->SetLocalFieldValue(G4ThreeVector(0.,0.,(4.8*megavolt/m)))
    ;
    flogicShape10->SetFieldManager(felFieldSetup2_0->GetLocalFieldManager(),
    true );
    G4AutoDelete::Register(felFieldSetup2_0);
  }
  if(!femFieldSetup1_0){
```

```
    static G4LogicalVolume* flogicShape1 = fLXeAccelerator->getlogicShape1();
    femFieldSetup1_0 = new F03FieldSetup();
    femFieldSetup1_0->SetLocalFieldValue(160.*tesla/(1.*m), G4ThreeVector() , 0.*deg);
    flogicShape1->SetFieldManager(femFieldSetup1_0->GetLocalFieldManager(), true );
    G4AutoDelete::Register(femFieldSetup1_0);
  }
  if(!femFieldSetup2_0){
    static G4LogicalVolume* flogicShape4 = fLXeAccelerator->getlogicShape4();
    femFieldSetup2_0 = new F03FieldSetup();
    femFieldSetup2_0->SetLocalFieldValue(196.*tesla/(1.*m), G4ThreeVector() , 90.*deg);
    flogicShape4->SetFieldManager(femFieldSetup2_0->GetLocalFieldManager(), true );
    G4AutoDelete::Register(femFieldSetup2_0);
  }
  if(!femFieldSetup3_0){
    static G4LogicalVolume* flogicShape8 = fLXeAccelerator->getlogicShape8();
    femFieldSetup3_0 = new F03FieldSetup();
    femFieldSetup3_0->SetLocalFieldValue(179.*tesla/(1.*m), G4ThreeVector() , 0.*deg);
    flogicShape8->SetFieldManager(femFieldSetup3_0->GetLocalFieldManager(), true );
    G4AutoDelete::Register(femFieldSetup3_0);
  }
  if(!femFieldSetup4_0){
    static G4LogicalVolume* flogicShape12 = fLXeAccelerator->getlogicShape12();
    femFieldSetup4_0 = new F03FieldSetup();
    femFieldSetup4_0->SetLocalFieldValue(186.*tesla/(1.*m), G4ThreeVector() , 90.*deg);
    flogicShape12->SetFieldManager(femFieldSetup4_0->GetLocalFieldManager(), true );
    G4AutoDelete::Register(femFieldSetup4_0);
  }

  // LXeModule1
  if(!felFieldSetup1_1){
    static G4LogicalVolume* flogicShape6 = fLXeModule1->getlogicShape6();
    felFieldSetup1_1 = new ElectricFieldSetup();
    felFieldSetup1_1->SetLocalFieldValue(G4ThreeVector(0.,0.,8.*megavolt/m));
    flogicShape6->SetFieldManager(felFieldSetup1_1->GetLocalFieldManager(), true );
    G4AutoDelete::Register(felFieldSetup1_1);
  }
  if(!felFieldSetup2_1){
    static G4LogicalVolume* flogicShape10 = fLXeModule1->getlogicShape10();
    felFieldSetup2_1 = new ElectricFieldSetup();
    felFieldSetup2_1->SetLocalFieldValue(G4ThreeVector(0.,0.,8.*megavolt/m));
    flogicShape10->SetFieldManager(felFieldSetup2_1->GetLocalFieldManager(), true );
    G4AutoDelete::Register(felFieldSetup2_1);
```

```
  }
  if(!femFieldSetup1_1){
    static G4LogicalVolume* flogicShape8 = fLXeModule1->getlogicShape8();
    femFieldSetup1_1 = new F03FieldSetup();
    femFieldSetup1_1->SetLocalFieldValue(179.*tesla/(1.*m), G4ThreeVector() ,
    0.*deg);
    flogicShape8->SetFieldManager(femFieldSetup1_1->GetLocalFieldManager(),
    true );
    G4AutoDelete::Register(femFieldSetup1_1);
  }
  if(!femFieldSetup2_1){
    static G4LogicalVolume* flogicShape12 = fLXeModule1->getlogicShape12();
    femFieldSetup2_1 = new F03FieldSetup();
    femFieldSetup2_1->SetLocalFieldValue(186.*tesla/(1.*m), G4ThreeVector() ,
    90.*deg);
    flogicShape12->SetFieldManager(femFieldSetup2_1->GetLocalFieldManager(),
    true );
    G4AutoDelete::Register(femFieldSetup2_1);
  }

  // LXeModule2
  if(!felFieldSetup1_2){
    static G4LogicalVolume* flogicShape6 = fLXeModule2->getlogicShape6();
    felFieldSetup1_2 = new ElectricFieldSetup();
    felFieldSetup1_2->SetLocalFieldValue(G4ThreeVector(0.,0.,11.*megavolt/m));
    flogicShape6->SetFieldManager(felFieldSetup1_2->GetLocalFieldManager(),
    true );
    G4AutoDelete::Register(felFieldSetup1_2);
  }
  if(!felFieldSetup2_2){
    static G4LogicalVolume* flogicShape10 = fLXeModule2->getlogicShape10();
    felFieldSetup2_2 = new ElectricFieldSetup();
    felFieldSetup2_2->SetLocalFieldValue(G4ThreeVector(0.,0.,11.*megavolt/m));
    flogicShape10->SetFieldManager(felFieldSetup2_2->GetLocalFieldManager(),
    true );
    G4AutoDelete::Register(felFieldSetup2_2);
  }
  if(!femFieldSetup1_2){
    static G4LogicalVolume* flogicShape8 = fLXeModule2->getlogicShape8();
    femFieldSetup1_2 = new F03FieldSetup();
    femFieldSetup1_2->SetLocalFieldValue(179.*tesla/(1.*m), G4ThreeVector() ,
    0.*deg);
    flogicShape8->SetFieldManager(femFieldSetup1_2->GetLocalFieldManager(),
    true );
    G4AutoDelete::Register(femFieldSetup1_2);
  }
  if(!femFieldSetup2_2){
    static G4LogicalVolume* flogicShape12 = fLXeModule2->getlogicShape12();
    femFieldSetup2_2 = new F03FieldSetup();
    femFieldSetup2_2->SetLocalFieldValue(186.*tesla/(1.*m), G4ThreeVector() ,
    90.*deg);
    flogicShape12->SetFieldManager(femFieldSetup2_2->GetLocalFieldManager(),
```

```
    true );
    G4AutoDelete::Register(femFieldSetup2_2);
  }

  // LXeModule3
  if(!felFieldSetup1_3){
    static G4LogicalVolume* flogicShape6 = fLXeModule3->getlogicShape6();
    felFieldSetup1_3 = new ElectricFieldSetup();
    felFieldSetup1_3->SetLocalFieldValue(G4ThreeVector(0.,0.,13.4*megavolt/m));
    flogicShape6->SetFieldManager(felFieldSetup1_3->GetLocalFieldManager(),
    true );
    G4AutoDelete::Register(felFieldSetup1_3);
  }
  if(!felFieldSetup2_3){
    static G4LogicalVolume* flogicShape10 = fLXeModule3->getlogicShape10();
    felFieldSetup2_3 = new ElectricFieldSetup();
    felFieldSetup2_3->SetLocalFieldValue(G4ThreeVector(0.,0.,13.4*megavolt/m));
    flogicShape10->SetFieldManager(felFieldSetup2_3->GetLocalFieldManager(),
    true );
    G4AutoDelete::Register(felFieldSetup2_3);
  }
  if(!femFieldSetup1_3){
    static G4LogicalVolume* flogicShape8 = fLXeModule3->getlogicShape8();
    femFieldSetup1_3 = new F03FieldSetup();
    femFieldSetup1_3->SetLocalFieldValue(179.*tesla/(1.*m), G4ThreeVector() ,
    0.*deg);
    flogicShape8->SetFieldManager(femFieldSetup1_3->GetLocalFieldManager(),
    true );
    G4AutoDelete::Register(femFieldSetup1_3);
  }
  if(!femFieldSetup2_3){
    static G4LogicalVolume* flogicShape12 = fLXeModule3->getlogicShape12();
    femFieldSetup2_3 = new F03FieldSetup();
    femFieldSetup2_3->SetLocalFieldValue(186.*tesla/(1.*m), G4ThreeVector() ,
    90.*deg);
    flogicShape12->SetFieldManager(femFieldSetup2_3->GetLocalFieldManager(),
    true );
    G4AutoDelete::Register(femFieldSetup2_3);
  }

  // LXeModuleCCL1
  if(!felFieldSetup1_4){
    static G4LogicalVolume* flogicShape1 = fLXeModuleCCL1->getlogicShape1();
    felFieldSetup1_4 = new ElectricFieldSetup();
    felFieldSetup1_4->SetLocalFieldValue(G4ThreeVector(0.,0.,12.*megavolt/m));
    flogicShape1->SetFieldManager(felFieldSetup1_4->GetLocalFieldManager(),
    true );
    G4AutoDelete::Register(felFieldSetup1_4);
  }
  if(!felFieldSetup2_4){
    static G4LogicalVolume* flogicShape4 = fLXeModuleCCL1->getlogicShape4();
    felFieldSetup2_4 = new ElectricFieldSetup();
```

```
    felFieldSetup2_4->SetLocalFieldValue(G4ThreeVector(0.,0.,12.*megavolt/m));
    flogicShape4->SetFieldManager(felFieldSetup2_4->GetLocalFieldManager(),
   true );
    G4AutoDelete::Register(felFieldSetup2_4);
  }
  if(!femFieldSetup1_4){
    static G4LogicalVolume* flogicShape3 = fLXeModuleCCL1->getlogicShape3();
    femFieldSetup1_4 = new F03FieldSetup();
    femFieldSetup1_4->SetLocalFieldValue(160.*tesla/(1.*m), G4ThreeVector() ,
   0.*deg);
    flogicShape3->SetFieldManager(femFieldSetup1_4->GetLocalFieldManager(),
   true );
    G4AutoDelete::Register(femFieldSetup1_4);
  }
  if(!femFieldSetup2_4){
    static G4LogicalVolume* flogicShape6 = fLXeModuleCCL1->getlogicShape6();
    femFieldSetup2_4 = new F03FieldSetup();
    femFieldSetup2_4->SetLocalFieldValue(160.*tesla/(1.*m), G4ThreeVector() ,
   90.*deg);
    flogicShape6->SetFieldManager(femFieldSetup2_4->GetLocalFieldManager(),
   true );
    G4AutoDelete::Register(femFieldSetup2_4);
  }

  // LXeModuleCCL2
  if(!felFieldSetup1_5){
    static G4LogicalVolume* flogicShape1 = fLXeModuleCCL2->getlogicShape1();
    felFieldSetup1_5 = new ElectricFieldSetup();
    felFieldSetup1_5->SetLocalFieldValue(G4ThreeVector(0.,0.,12.*megavolt/m));
    flogicShape1->SetFieldManager(felFieldSetup1_5->GetLocalFieldManager(),
   true );
    G4AutoDelete::Register(felFieldSetup1_5);
  }
  if(!felFieldSetup2_5){
    static G4LogicalVolume* flogicShape4 = fLXeModuleCCL2->getlogicShape4();
    felFieldSetup2_5 = new ElectricFieldSetup();
    felFieldSetup2_5->SetLocalFieldValue(G4ThreeVector(0.,0.,12.*megavolt/m));
    flogicShape4->SetFieldManager(felFieldSetup2_5->GetLocalFieldManager(),
   true );
    G4AutoDelete::Register(felFieldSetup2_5);
  }
  if(!femFieldSetup1_5){
    static G4LogicalVolume* flogicShape3 = fLXeModuleCCL2->getlogicShape3();
    femFieldSetup1_5 = new F03FieldSetup();
    femFieldSetup1_5->SetLocalFieldValue(160.*tesla/(1.*m), G4ThreeVector() ,
   0.*deg);
    flogicShape3->SetFieldManager(femFieldSetup1_5->GetLocalFieldManager(),
   true );
    G4AutoDelete::Register(femFieldSetup1_5);
  }
  if(!femFieldSetup2_5){
    static G4LogicalVolume* flogicShape6 = fLXeModuleCCL2->getlogicShape6();
```

```
    femFieldSetup2_5 = new F03FieldSetup();
    femFieldSetup2_5->SetLocalFieldValue(160.*tesla/(1.*m), G4ThreeVector() ,
   90.*deg);
    flogicShape6->SetFieldManager(femFieldSetup2_5->GetLocalFieldManager(),
   true );
    G4AutoDelete::Register(femFieldSetup2_5);
  }

  // LXeModuleCCL3
  if(!felFieldSetup1_6){
    static G4LogicalVolume* flogicShape1 = fLXeModuleCCL3->getlogicShape1();
    felFieldSetup1_6 = new ElectricFieldSetup();
    felFieldSetup1_6->SetLocalFieldValue(G4ThreeVector(0.,0.,12.*megavolt/m));
    flogicShape1->SetFieldManager(felFieldSetup1_6->GetLocalFieldManager(),
   true );
    G4AutoDelete::Register(felFieldSetup1_6);
  }
  if(!felFieldSetup2_6){
    static G4LogicalVolume* flogicShape4 = fLXeModuleCCL3->getlogicShape4();
    felFieldSetup2_6 = new ElectricFieldSetup();
    felFieldSetup2_6->SetLocalFieldValue(G4ThreeVector(0.,0.,12.*megavolt/m));
    flogicShape4->SetFieldManager(felFieldSetup2_6->GetLocalFieldManager(),
   true );
    G4AutoDelete::Register(felFieldSetup2_6);
  }
  if(!femFieldSetup1_6){
    static G4LogicalVolume* flogicShape3 = fLXeModuleCCL3->getlogicShape3();
    femFieldSetup1_6 = new F03FieldSetup();
    femFieldSetup1_6->SetLocalFieldValue(160.*tesla/(1.*m), G4ThreeVector() ,
   0.*deg);
    flogicShape3->SetFieldManager(femFieldSetup1_6->GetLocalFieldManager(),
   true );
    G4AutoDelete::Register(femFieldSetup1_6);
  }
  if(!femFieldSetup2_6){
    static G4LogicalVolume* flogicShape6 = fLXeModuleCCL3->getlogicShape6();
    femFieldSetup2_6 = new F03FieldSetup();
    femFieldSetup2_6->SetLocalFieldValue(160.*tesla/(1.*m), G4ThreeVector() ,
   90.*deg);
    flogicShape6->SetFieldManager(femFieldSetup2_6->GetLocalFieldManager(),
   true );
    G4AutoDelete::Register(femFieldSetup2_6);
  }

  // LXeModuleCCL4
  if(!felFieldSetup1_7){
    static G4LogicalVolume* flogicShape1 = fLXeModuleCCL4->getlogicShape1();
    felFieldSetup1_7 = new ElectricFieldSetup();
    felFieldSetup1_7->SetLocalFieldValue(G4ThreeVector(0.,0.,12.*megavolt/m));
    flogicShape1->SetFieldManager(felFieldSetup1_7->GetLocalFieldManager(),
   true );
    G4AutoDelete::Register(felFieldSetup1_7);
```

```
  }
  if(!felFieldSetup2_7){
    static G4LogicalVolume* flogicShape4 = fLXeModuleCCL4->getlogicShape4();
    felFieldSetup2_7 = new ElectricFieldSetup();
    felFieldSetup2_7->SetLocalFieldValue(G4ThreeVector(0.,0.,12.*megavolt/m));
    flogicShape4->SetFieldManager(felFieldSetup2_7->GetLocalFieldManager(),
    true );
    G4AutoDelete::Register(felFieldSetup2_7);
  }
  if(!femFieldSetup1_7){
    static G4LogicalVolume* flogicShape3 = fLXeModuleCCL4->getlogicShape3();
    femFieldSetup1_7 = new F03FieldSetup();
    femFieldSetup1_7->SetLocalFieldValue(160.*tesla/(1.*m), G4ThreeVector() ,
    0.*deg);
    flogicShape3->SetFieldManager(femFieldSetup1_7->GetLocalFieldManager(),
    true );
    G4AutoDelete::Register(femFieldSetup1_7);
  }
  if(!femFieldSetup2_7){
    static G4LogicalVolume* flogicShape6 = fLXeModuleCCL4->getlogicShape6();
    femFieldSetup2_7 = new F03FieldSetup();
    femFieldSetup2_7->SetLocalFieldValue(160.*tesla/(1.*m), G4ThreeVector() ,
    90.*deg);
    flogicShape6->SetFieldManager(femFieldSetup2_7->GetLocalFieldManager(),
    true );
    G4AutoDelete::Register(femFieldSetup2_7);
  }
}
//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo
    ......

void LXeDetectorConstruction::SetDimensions(G4ThreeVector dims)
{
  fScint_x = dims[0];
  fScint_y = dims[1];
  fScint_z = dims[2];
  G4RunManager::GetRunManager()->ReinitializeGeometry();
}

//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo
    ......

void LXeDetectorConstruction::SetHousingThickness(G4double d_mtl)
{
  fD_mtl = d_mtl;
  G4RunManager::GetRunManager()->ReinitializeGeometry();
}

//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo
    ......

void LXeDetectorConstruction::SetNX(G4int nx)
```

```
{
  fNx = nx;
  G4RunManager::GetRunManager()->ReinitializeGeometry();
}

//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo......

void LXeDetectorConstruction::SetNY(G4int ny)
{
  fNy = ny;
  G4RunManager::GetRunManager()->ReinitializeGeometry();
}

//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo......

void LXeDetectorConstruction::SetNZ(G4int nz)
{
  fNz = nz;
  G4RunManager::GetRunManager()->ReinitializeGeometry();
}

//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo......

void LXeDetectorConstruction::SetPMTRadius(G4double outerRadius_pmt)
{
  fOuterRadius_pmt = outerRadius_pmt;
  G4RunManager::GetRunManager()->ReinitializeGeometry();
}

//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo......

void LXeDetectorConstruction::SetDefaults()
{
  // Resets to default values
  fD_mtl = 0.0635 * cm;
  fD_moderator = 3. * cm;

  fScint_x = 6. * cm;
  fScint_y = 6. * cm;
  fScint_z = 20. * cm;

  fNx = 1;
  fNy = 1;
  fNz = 0;

  fOuterRadius_pmt = 2.3 * cm;

  fSphereOn = false;
```

```
  fRefl       = 1.0;

  fNfibers       = 1;
  fWLSslab       = false;
  fMainVolumeOn  = true;
  fMainVolume    = nullptr;
  fSlab_z        = 2.5 * mm;

  G4UImanager::GetUIpointer()->ApplyCommand(
  "/LXe/detector/scintYieldFactor 1.");

  if(fLXe_mt)
  fLXe_mt->AddConstProperty("SCINTILLATIONYIELD", 12000. / MeV);
  if(fMPTPStyrene)
  fMPTPStyrene->AddConstProperty("SCINTILLATIONYIELD", 10. / keV);
}

//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo......

void LXeDetectorConstruction::SetSphereOn(G4bool b)
{
  fSphereOn = b;
  G4RunManager::GetRunManager()->ReinitializeGeometry();
}

//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo......

void LXeDetectorConstruction::SetHousingReflectivity(G4double r)
{
  fRefl = r;
  G4RunManager::GetRunManager()->ReinitializeGeometry();
}

//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo......

void LXeDetectorConstruction::SetWLSSlabOn(G4bool b)
{
  fWLSslab = b;
  G4RunManager::GetRunManager()->ReinitializeGeometry();
}

//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo......

void LXeDetectorConstruction::SetMainVolumeOn(G4bool b)
{
  fMainVolumeOn = b;
  G4RunManager::GetRunManager()->ReinitializeGeometry();
}
```

```
//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo......

void LXeDetectorConstruction::SetNFibers(G4int n)
{
  fNfibers = n;
  G4RunManager::GetRunManager()->ReinitializeGeometry();
}

//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo......

void LXeDetectorConstruction::SetMainScintYield(G4double y)
{
  fLXe_mt->AddConstProperty("SCINTILLATIONYIELD", y / MeV);
}

//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo......

void LXeDetectorConstruction::SetWLSScintYield(G4double y)
{
  fMPTPStyrene->AddConstProperty("SCINTILLATIONYIELD", y / MeV);
}

//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo......

void LXeDetectorConstruction::SetSaveThreshold(G4int save)
{
  // Sets the save threshold for the random number seed. If the number of
  // photons generated in an event is lower than this, then save the seed for
  // this event in a file called run###evt###.rndm

  fSaveThreshold = save;
  G4RunManager::GetRunManager()->SetRandomNumberStore(true);
}

//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo......
```

Bibliography

[1] World Health Organisation

[2] C. A. Bertulani, P. Danielewicz *Introduction to Nuclear Reactions*

[3] C. Patrignani et al. (Particle Data Group), Chinese Physics C, 40, 100001 (2016)

[4] S. Braibant, G. Giacomelli, M. Spurio *Particles and Fundamental Interactions*

[5] M.J. Berger; J.H. Hubbell; S.M. Seltzer; J. Chang; J.S. Coursey; R. Sukumar; D.S. Zucker. *XCOM: Photon Cross Sections Database*, National Institute of Standards and Technology (NIST), Retrieved 2 Nov 2007

[6] E. Haettner, H. Iwase, M. Krämer, G. Kraft and D. Schardt, *Experimental study of nuclear fragmentation of 200 and 400 MeV/u ^{12}C ions in water for applications in particle therapy*, Physics in Medicine and Biology, Volume 58, Number 23, 2013

[7] M. Philippe, C. Jimonet *European Radiation Protection Course: Basics*

[8] ICRP, 2007. The 2007 Recommendations of the International Commission on Radiological Protection. ICRP Publication 103. Ann. ICRP 37 (2-4).

[9] M. Pelliccioni *Elementi di Dosimetria delle Radiazioni*

[10] P. Dendy, B. Heaton *Physics for Diagnostic Radiology, 2nd Edition*

[11] Dr. Jurgen Kiefer *Biological Radiation Effects* (1989)

[12] H. Paganetti 2002 *Phys. Med. Biol. 47 747*

[13] P. Tommasino, M. Durante *Proton radiobiology*, Cancers, vol. 7, no. 1, pp. 353-381, 2015

[14] C. Ottaviani *The Inverse Kinematics approach in the FOOT experiment*, Alma Mater Studiorum-Università di Bologna

[15] A. Sibtain, A. Morgan, N. MacDougall *Physics for Clinical Oncology*

[16] M. Krengli, R. Orecchia *Medical aspects of the national centre for oncological hadrontherapy (CNAO) in Italy*

[17] D. Alberti, A. Michelotti, A. Lanfranco, N. Protti, S. Altieri, A. Deagostino, S. Geninatti Crich *In vitro and in vivo BNCT investigations using a carborane containing sulfonamide targeting CAIX epitopes on malignant pleural mesothelioma and breast cancer cells*, Sci. Rep. 2020; 10:19274

[18] G.A.P. Cirrone, ..., G. Cuttone, G. Korn *First experimental proof of Proton Boron Capture Therapy (PBCT) to enhance protontherapy effectiveness*

[19] E. Segrè *Nuclei e Particelle*, Zanichelli

[20] L. Picardi, C. Ronsivalle, A. Vignati *Progetto del TOP Linac, ENEA Technical Report RT/INN/97/17*

[21] M. E. Abdelaziz, A. M. Ghander *A study of a duoplasmatron ion source with an expansion cup*, IEEE transaction on nuclear science, June 1967

[22] H. Wiedemann *Particle Accelerator Physics*, Springer

[23] L. Picardi, C. Ronsivalle, B. Spataro *Design development of the SCDTL structure for the TOP linac*

[24] G. De Michele *Un sistema di misura per l'ottimizzazione del profilo di campo accelerante in un linac sc per protoni*, Università di Napoli Federico II

[25] S. Hanna *RF linear accelerators for medical and industrial applications*, Artech House

[26] E. Mauro *Radiation Protection studies for CERN LINAC4/SPL accelerator complex*, EPFL

[27] Shielding aspects of accelerators, targets and irradiation facilities (SATIF-10), Workshop proceedings Geneva, Switzerland 2-4 June 2010

[28] J. E. Martin *Physics for Radiation Protection*, Wiley-VCH

[29] F. M. Khan *The physics of radiation therapy*, Lippincott Williams & Wilkins

[30] Khasanov, S., Yang, B., Su, Y. *et al.* Induced radioactivity at particle accelerators: a short review. *Radiat Detect Technol Methods 5, 481-492 (2021).*

[31] Geant4, a simulation toolkit.

[32] CADMesh, a CAD interface for GEANT4

[33] C. Schmidt, T. Stahl and E. Mergel, Current problems in the field of radiation protection technique - Use of active personal dosimeters (APD) in pulsed radiation fields, in: Proceedings of the 2009 EUROSAFE Forum for nuclear Safety, Brussels, Belgium, 2-3 November2009 (2009).

[34] G. P. Manessi *Development of advanced radiation monitors for pulsed neutron fields*, University of Liverpool

[35] C. H. Westcott *A study of expected loss rates in the counting of particles from pulsed sources*

[36] Aza, E., Caresana, M., Cassell, C., Charitonidis, N., Harrouch, E., Manessi, G.P., Pangallo, M., Perrin, D., Samara, E., Silari, M., 2013. Instrument intercomparison in the Pulsed Neutron Fields at the CERN HiRadMat Facility. CERN. Technical Note CERN-RP-2013-037-REPORTS-TN.