



Universiteit Gent
Faculteit Wetenschappen
Vakgroep Fysica en Sterrenkunde

² No title yet

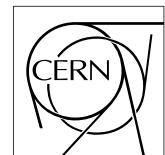
³ No sub-title neither, obviously...

⁴ Alexis Fagot

5



Thesis to obtain the degree of
Doctor of Philosophy in Physics
Academic years 2012-2017



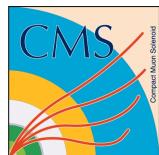


Universiteit Gent
Faculteit Wetenschappen
Vakgroep Fysica en Sterrenkunde

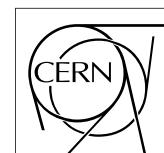
Promotoren: Dr. Michael Tytgat
Prof. Dr. Dirk Ryckbosch

Universiteit Gent
Faculteit Wetenschappen
Vakgroep Fysica en Sterrenkunde
Proeftuinstraat 86, B-9000 Gent, België
Tel.: +32 9 264.65.28
Fax.: +32 9 264.66.97

17



Thesis to obtain the degree of
Doctor of Philosophy in Physics
Academic years 2012-2017



Acknowledgements

¹⁹ Ici on remerciera tous les gens que j'ai pu croiser durant cette aventure et qui m'ont permis de passer
²⁰ un bon moment

²¹ *Gent, ici la super date de la mort qui tue de la fin d'écriture*
²² *Alexis Fagot*

Table of Contents

24	Acknowledgements	i
25	Nederlandse samenvatting	vii
26	English summary	ix
27	1 Introduction	1-1
28	1.1 A story of High Energy Physics	1-1
29	1.2 Organisation of this study	1-1
30	2 Investigating the TeV scale	2-1
31	2.1 The Standard Model of Particle Physics	2-1
32	2.2 The Large Hadron Collider and the Compact Muon Solenoid	2-1
33	2.2.1 CMS, a multipurpose experiment	2-1
34	2.3 Muon Phase-II Upgrade	2-1
35	3 Physics of Resistive plate chambers	3-1
36	3.1 Principle	3-1
37	3.1.1 Electron drift velocity	3-4
38	3.2 Rate capability and time resolution of Resistive Plate Chambers	3-4
39	3.2.1 Operation modes	3-4
40	3.2.2 Detector designs and performance	3-6
41	3.2.2.1 Double-gap RPC	3-6
42	3.2.2.2 Multigap RPC (MRPC)	3-7
43	3.2.2.3 Charge distribution and performance limitations	3-8
44	3.3 Signal formation	3-11
45	3.4 Gas transport parameters	3-11
46	4 Longevity studies and Consolidation of the present CMS RPC subsystem	4-1
47	4.1 Resistive Plate Chambers at CMS	4-1
48	4.1.1 Overview	4-1
49	4.1.2 The present RPC system	4-2
50	4.1.3 Pulse processing of CMS RPCs	4-3
51	4.2 Testing detectors under extreme conditions	4-3
52	4.2.1 The Gamma Irradiation Facilities	4-6
53	4.2.1.1 GIF	4-6
54	4.2.1.2 GIF++	4-8
55	4.3 Preliminary tests at GIF	4-10
56	4.3.1 Resistive Plate Chamber test setup	4-10
57	4.3.2 Data Acquisition	4-12
58	4.3.3 Geometrical acceptance of the setup layout to cosmic muons	4-12

59	4.3.3.1	Description of the simulation layout	4-13
60	4.3.3.2	Simulation procedure	4-15
61	4.3.3.3	Results	4-16
62	4.3.4	Photon flux at GIF	4-16
63	4.3.4.1	Expectations from simulations	4-16
64	4.3.4.2	Dose measurements	4-21
65	4.3.5	Results and discussions	4-22
66	4.4	Longevity tests at GIF++	4-23
67	4.4.1	Description of the Data Acquisition	4-26
68	4.4.2	RPC current, environmental and operation parameter monitoring	4-27
69	4.4.3	Measurement procedure	4-28
70	4.4.4	Longevity studies results	4-28
71	5	Investigation on high rate RPCs	5-1
72	5.1	Rate limitations and ageing of RPCs	5-1
73	5.1.1	Low resistivity electrodes	5-1
74	5.1.2	Low noise front-end electronics	5-1
75	5.2	Construction of prototypes	5-1
76	5.3	Results and discussions	5-1
77	6	Conclusions and outlooks	6-1
78	6.1	Conclusions	6-1
79	6.2	Outlooks	6-1
80	A	A data acquisition software for CAEN VME TDCs	A-1
81	A.1	GIF++ DAQ file tree	A-1
82	A.2	Usage of the DAQ	A-2
83	A.3	Description of the readout setup	A-3
84	A.4	Data read-out	A-3
85	A.4.1	V1190A TDCs	A-4
86	A.4.2	DataReader	A-6
87	A.4.3	Data quality flag	A-10
88	A.5	Communications	A-12
89	A.5.1	V1718 USB Bridge	A-13
90	A.5.2	Configuration file	A-13
91	A.5.3	WebDCS/DAQ intercommunication	A-17
92	A.5.4	Example of inter-process communication cycle	A-18
93	A.6	Software export	A-18
94	B	Details on the offline analysis package	B-1
95	B.1	GIF++ Offline Analysis file tree	B-1
96	B.2	Usage of the Offline Analysis	B-2
97	B.2.1	Output of the offline tool	B-3
98	B.2.1.1	ROOT file	B-3
99	B.2.1.2	CSV files	B-5
100	B.3	Analysis inputs and information handling	B-6
101	B.3.1	Dimensions file and IniFile parser	B-6
102	B.3.2	TDC to RPC link file and Mapping	B-7
103	B.4	Description of GIF++ setup within the Offline Analysis tool	B-9
104	B.4.1	RPC objects	B-9

105	B.4.2 Trolley objects	B-10
106	B.4.3 Infrastructure object	B-11
107	B.5 Handeling of data	B-12
108	B.5.1 RPC hits	B-13
109	B.5.2 Clusters of hits	B-14
110	B.6 DAQ data Analysis	B-15
111	B.6.1 Determination of the run type	B-16
112	B.6.2 Beam time window calculation for efficiency runs	B-17
113	B.6.3 Data loop and histogram filling	B-18
114	B.6.4 Results calculation	B-19
115	B.6.4.1 Rate normalisation	B-19
116	B.6.4.2 Rate and activity	B-21
117	B.6.4.3 Strip masking tool	B-23
118	B.6.4.4 Output CSV files filling	B-25
119	B.7 Current data Analysis	B-29

¹²⁰

Nederlandse samenvatting
–Summary in Dutch–

¹²¹

¹²² Le resume en Neerlandais (j'aurais peut-être pu apprendre la langue juste pour ça...).

¹²³

English summary

¹²⁴ Le meme résume mais en Anglais (on commencera par la hein!).

1

Introduction

125

126

¹²⁷ **1.1 A story of High Energy Physics**

¹²⁸ **1.2 Organisation of this study**

2

129

130

Investigating the TeV scale

131 **2.1 The Standard Model of Particle Physics**

132 **2.2 The Large Hadron Collider and the Compact Muon Solenoid**

133 **2.2.1 CMS, a multipurpose experiment**

134 Along the path of the LHC beams, 4 main experiments have been built in order to investigate new
135 physics:

- 136 • ALICE
- 137 • ATLAS
- 138 • CMS
- 139 • LHCb

140 **2.3 Muon Phase-II Upgrade**

141 After the more than two years lasting First Long Shutdown (LS1), the Large Hadron Collider (LHC)
142 delivered its very first Run-II proton-proton collisions early 2015. LS1 gave the opportunity to the
143 LHC and to the its experiments to undergo upgrades. The accelerator is now providing collisions
144 at center-of-mass energy of 13 TeV and bunch crossing rate of 40 MHz, with a peak luminosity
145 exceeding its design value. During the first and upcoming second LHC Long Shutdown, the Compact
146 Muon Solenoid (CMS) detector is also undergoing a number of upgrades to maintain a high system
147 performance [1].

148 From the LHC Phase-2 or High Luminosity LHC (HL-LHC) period onwards, i.e. past the Third
149 Long Shutdown (LS3), the performance degradation due to integrated radiation as well as the average
150 number of inelastic collisions per bunch crossing, or pileup, will rise substantially and become a

major challenge for the LHC experiments, like CMS that are forced to address an upgrade program for Phase-II [2]. Simulations of the expected distribution of absorbed dose in the CMS detector under HL-LHC conditions, show in figure 4.16 that detectors placed close to the beamline will have to withstand high irradiation, the radiation dose being of the order of a few tens of Gy.

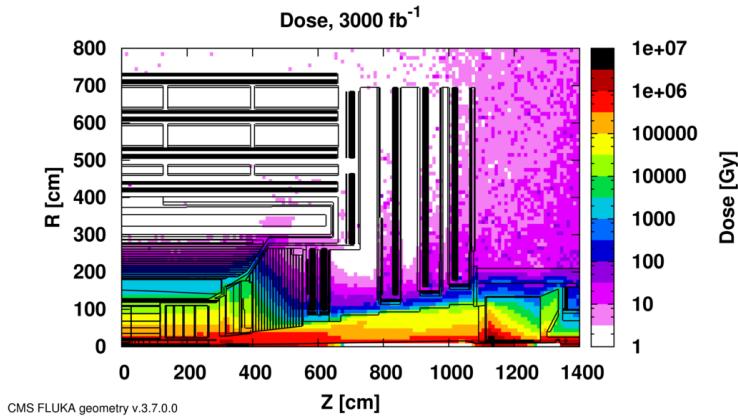


Figure 2.1: Absorbed dose in the CMS cavern after an integrated luminosity of 3000 fb. R is the transverse distance from the beamline and Z is the distance along the beamline from the Interaction Point at Z=0.

The measurement of small production cross-section and/or decay branching ratio processes, such as the Higgs boson coupling to charge leptons or the $B_s \rightarrow \mu^+\mu^-$ decay, is of major interest and specific upgrades in the forward regions of the detector will be required to maximize the physics acceptance on the largest possible solid angle. To ensure proper trigger performance within the present coverage, the muon system will be completed with new chambers. In figure 2.2 one can see that the existing Cathode Strip Chambers (CSCs) will be completed by Gas Electron Multipliers (GEMs) and Resistive Plate Chambers (RPCs) in the pseudo-rapidity region $1.6 < |\eta| < 2.4$ to complete its redundancy as originally scheduled in the CMS Technical Proposal [3].

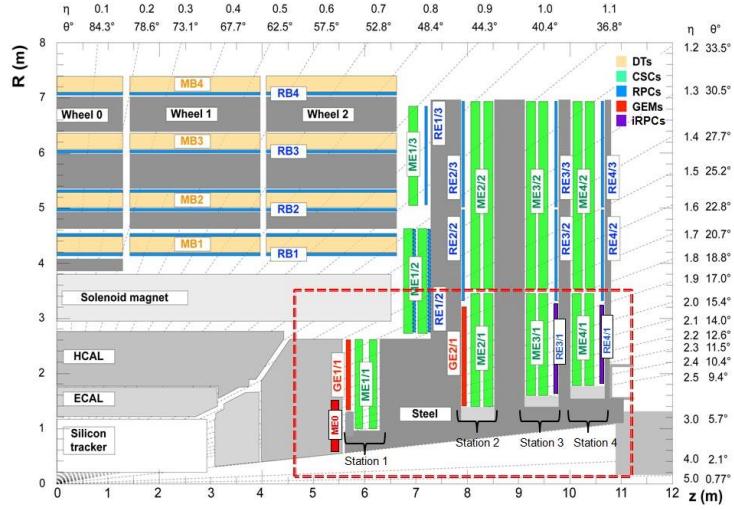


Figure 2.2: A quadrant of the muon system, showing DTs (yellow), RPCs (light blue), and CSCs (green). The locations of new forward muon detectors for Phase-II are contained within the dashed box and indicated in red for GEM stations (ME0, GE1/1, and GE2/1) and dark blue for improved RPC (iRPC) stations (RE3/1 and RE4/1).

163 RPCs are used by the CMS first level trigger for their good timing performances. Indeed, a very
 164 good bunch crossing identification can be obtained with the present CMS RPC system, given their
 165 fast response of the order of 1 ns. In order to contribute to the precision of muon momentum mea-
 166 surements, muon chambers should have a spatial resolution less or comparable to the contribution
 167 of multiple scattering [1]. Most of the plausible physics is covered only considering muons with
 168 $p_T < 100$ GeV thus, in order to match CMS requirements, a spatial resolution of $\mathcal{O}(\text{few mm})$ the
 169 proposed new RPC stations, as shown by the simulation in figure 2.3. According to preliminary
 170 designs, RE3/1 and RE4/1 readout pitch will be comprised between 3 and 6 mm and 5 η -partitions
 171 could be considered.

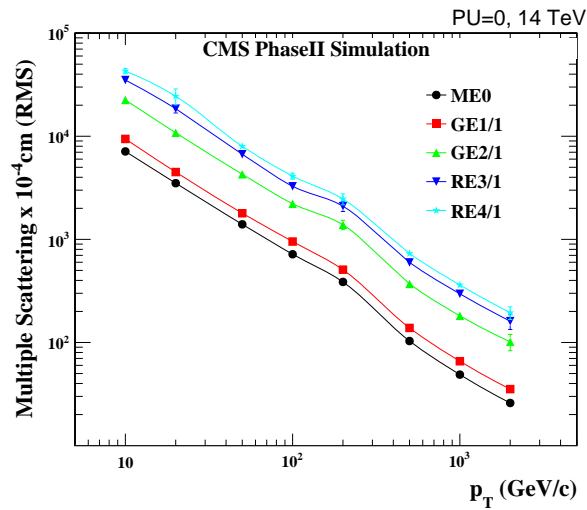


Figure 2.3: RMS of the multiple scattering displacement as a function of muon p_T for the proposed forward muon stations. All of the electromagnetic processes such as bremsstrahlung and magnetic field effect are included in the simulation.

3

172

173

Physics of Resistive plate chambers

174 A Resistive Plate Chamber (RPC) is a gaseous detector using the same physical processes described
175 in Chapter 3. It has been developed in 1981 by Santonico and Cardarelli [4], under the name of
176 *Resistive Plate Counter*, as an alternative to the local-discharge spark counters proposed in 1978
177 by Pestov and Fedotovich [5, 6]. Working with spark chambers implied using high-pressure gas
178 and high mechanical precision which the RPC simplified by formerly using a gas mixture of argon
179 and butane flowed at atmospheric pressure and a constant and uniform electric field propagated
180 in between two parallel electrode plates. Moreover, a significant increase in rate capability was
181 introduced by the use of electrode plate material with high bulk resistivity, preventing the discharge
182 from growing throughout the whole gas gap. Indeed, the effect of using resistive electrodes is that
183 the constant electric field is locally canceled out by the development of the discharge, limiting its
184 growth.

185 Through its development history, different operating modes [7–9] and new detector designs [10–
186 12] have been discovered, leading to further improvement of the rate capability of such a detector.
187 Moreover, the addition of SF_6 into the gas mix improved the stability of operation of the RPC [13,
188 14].

189 The low developing costs and easily achievable large detection areas offered by RPCs, as well
190 as the wide range of possible designs, made them a natural choice to as muon chambers and/or
191 trigger detectors in multipurpose experiments such as CMS [1] or ATLAS [15], time-of-flight detec-
192 tors in ALICE [16], calorimeter with CALICE [17] or even detectors for volcanic muography with
193 ToMuVol [18].

194 3.1 Principle

195 RPCs are ionisation detectors composed of two parallel resistive plate electrodes in between which
196 a constant electric field is set. The space in between the electrodes, referred as *gap*, is filled with a
197 dense gas that is used to generate primary ionization into the gas volume. The free charge carriers
198 (electrons and cations) created by the ionization of the gas molecules are then accelerated towards

¹⁹⁹ the electrodes by the electric field, as shown in Figure 3.1 [19].

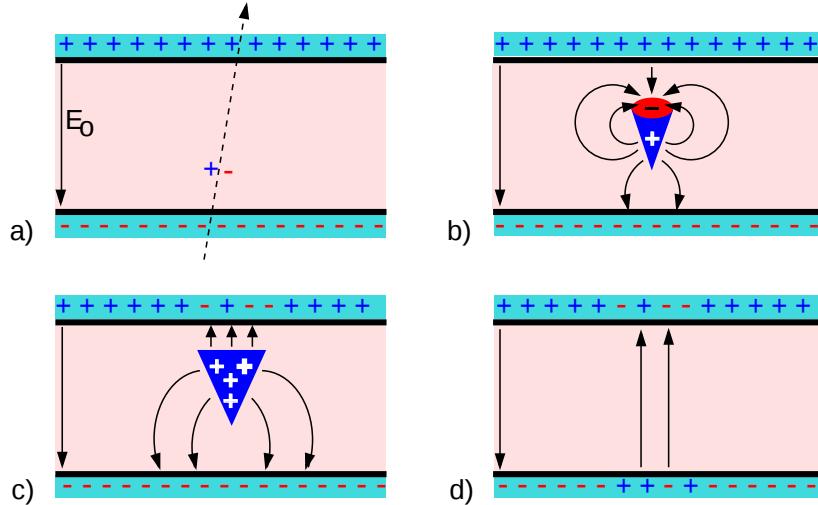


Figure 3.1: Different phases of the avalanche development in the RPC gas volume subjected to a constant electric field E_0 . a) An avalanche is initiated by the primary ionisation caused by the passage of a charged particle through the gas volume. b) Due to its growing size, the avalanche starts to locally influence the electric field. c) The electrons, lighter than the cations reach the anode first. d) The ions reach the cathode. While the charges have not recombined, the electric field in the small region around the avalanche stays affected and locally blind the detector.

²⁰⁰ RPCs being passive detectors, a current on pick-up copper read-out placed outside of the gas
²⁰¹ volume is induced by the charge accumulation during the growth of the avalanche. As a result,
²⁰² the time resolution of the detector is substantially increased as the output signal is generated while
²⁰³ the electrons are still in movement. The advantage of a constant electric field, over multi-wire
²⁰⁴ proportional chambers, is that the electrons are being fully accelerated from the moment charge
²⁰⁵ carriers are freed and feel the full strength of the electric field that doesn't depend on the distance to
²⁰⁶ the readout and that the output signal doesn't need for the electrons to be physically collected.

²⁰⁷ The typical gas mixture RPCs are operated with is generally composed of 3 gas compounds.

- ²⁰⁸ • Tetrafluoroethane ($C_2F_4H_2$), also referred to as *Freon*, is the principal compound of the RPC
²⁰⁹ gas mixtures, with a typical fraction above 90%. It is used for it's high effective Townsend
²¹⁰ coefficient and the great average fast charge that allows to operate the detector with a high
²¹¹ threshold with respect to argon, for example, that has similar effective Townsend coefficient
²¹² but suffers from a lower fast charge. To operate with similar conditions, argon would require a
²¹³ higher electric field leading to a higher fraction of streamers, thus limiting the rate capability
²¹⁴ of the detector [20].
- ²¹⁵ • Isobutane (i- C_4H_{10}), only present in a few percent in the gas mixtures, is used for its UV
²¹⁶ quenching properties [21] helping to prevent streamers due to UV photon emission during the
²¹⁷ avalanche growth.
- ²¹⁸ • Sulfur hexafluoride, (SF_6), referred to simply as *SF6*, is used in very little quantities for its
²¹⁹ high electronegativity. Excess of electrons are being absorbed by the compound and streamers

are suppressed [14]. Nevertheless, a fraction of SF_6 higher than 1% will not bring any extra benefit in terms of streamer cancelation power but will lead to higher operating voltage [13], as can be understood through Figure 3.2.

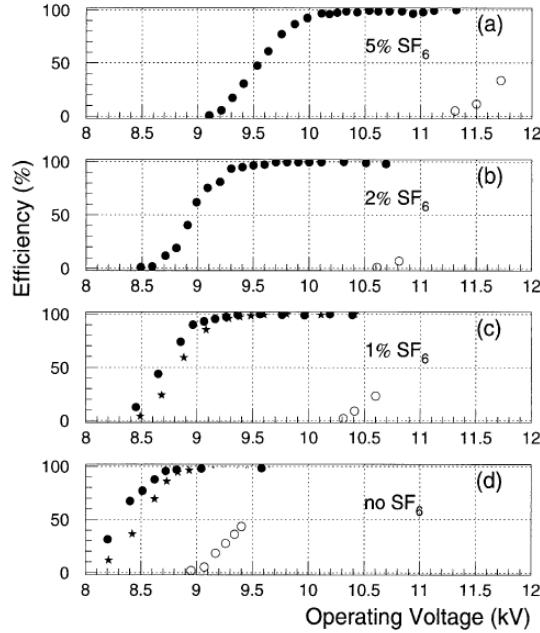


Figure 3.2: Effeciency (circles and stars with 30 mV and 100 mV thresholds respectively) and streamer probability (opened circles) as function of the operating voltatge of a 2 mm single gap HPL RPC flushed with a gas mixture containing (a) 5%, (b) 2%, (c) 1% and (d) no SF_6 [13].

After an avalanche developed in the gas, a time long compared to the development of a discharge is needed to recombine the charge carriers in the electrode material due to their resistivity. This property has the advantage of affecting the local electric field and avoiding sparks in the detector but, on the other hand, the rate capability is intrinsically limited by the time constant τ_{RPC} of the detector. Using a quasi-static approximation of Maxwell's equations for weakly conducting media, it can be shown that the time constant τ_{RPC} necessary to the charge recombination at the interface in between the electrode and the gas volume is given by the Formula 3.1 [22].

$$\tau_{RPC} = \frac{\epsilon_{electrode} + \epsilon_{gas}}{\sigma_{electrode} + \sigma_{gas}} \quad (3.1)$$

A gas can be assimilated to vacuum, leading to $\epsilon_{gas} = \epsilon_0$ and $\sigma_{gas} = 0$, and the electrodes permittivity and conductivity can be written as $\epsilon_{electrode} = \epsilon_r \epsilon_0$ and $\sigma_{electrode} = 1/\rho_{electrode}$, showing the strong dependence of the time constant to the electrodes resistivity in Formula 3.2.

$$\tau_{RPC} = (\epsilon_r + 1)\epsilon_0 \times \rho_{electrode} \quad (3.2)$$

Very few materials with a low enough resistivity exist in nature. The resistivity targeted to build RPCs ranges from 10^9 to $10^{12} \Omega \cdot \text{cm}$. The most common RPC electrode materials are displayed in Table 3.1. When the doped glass and ceramics can offer short time constants of the order of 1 ms,

236 the developing cost of such materials is quite high due to the very low demand. Thus, High-pressure
 237 laminate (HPL) is often the choice for high rate experiments using very large RPC detection areas.
 238 Other experiments working at cosmic muon fluxes can safely operate with ordinary float glass.

Material	$\rho_{electrode}$ ($\Omega \cdot \text{cm}$)	ϵ_r	τ_{RPC} (ms)
Float glass	10^{12}	~ 7	~ 700
High-pressure laminate	10^{10} to 10^{12}	~ 6	~ 6 to 600
Doped glass (LR S)	10^9 to 10^{11}	~ 10	~ 1 to 100
Doped ceramics (SiN/SiC)	10^9	~ 8.5	~ 1
Doped ceramics (Ferrite)	10^8 to 10^{12}	~ 20	~ 0.2 to 2000

Table 3.1: Properties of the most used electrode materials for RPCs.

3.1.1 Electron drift velocity

240 Talk about the electron drift velocity and mention the time resolution of RPCs.

3.2 Rate capability and time resolution of Resistive Plate Chambers

243 As already previously discussed, the electrode material plays a key role in the max intrinsic rate
 244 capability of RPCs. R&D is being done to develop at always cheaper costs material with lower
 245 resistivity. Nevertheless, the amount of charge released, i.e. the size of the discharge, if reduced
 246 leads to a smaller blind area in the detector, increasing the rate capability of the detector.

3.2.1 Operation modes

248 RPCs where developed early 1980s. At that time it was using an operating mode now referred to
 249 as *streamer mode*. Streamers are large discharges that develop in between the 2 electrodes enough
 250 to locally discharge the electrodes. If the electric field inside of the gas volume is strong enough,
 251 with electrons being fast compared to ions, a large and dense cloud of positive ions will develop
 252 nearby the anode and extend toward the cathode while the electrons are being collected, eventually
 253 leading to a streamer discharge due to the increase of field seen at the cathode. the field is then strong
 254 enough so that electrons are pulled out of the cathode. Electrodes, though they are a unique volume
 255 of resistive material, can be assimilated to capacitors. At the moment an electric field is applied in
 256 between their outer surfaces, the charge carriers inside of the volume will start moving leading to
 257 a situation where there is no voltage across the electrodes and a higher density of negative charges,
 258 i.e. electrons, on the inner surface of the cathode. Finally, when a streamer discharge occurs, these
 259 electrons are partially released in the gas volume contributing to increase the discharge strength until
 260 the formation of a conductive plasma, the streamer. This can be understood through Figure 3.3 [7].
 261 Streamer signals are very convenient in terms of read-out as no amplification is required with output
 262 pulses amplitudes of the order of a few tens to few hundreds of mV as can be seen on Figure 3.4.

263 When the electric field is reduced though, the electronic gain is small until the electrons get close
 264 enough to the anode and the positive ion cloud is much smaller. The electric field cannot rise to the
 265 point a field emission of electrons on the cathode is possible. The resulting signal is weak, of the

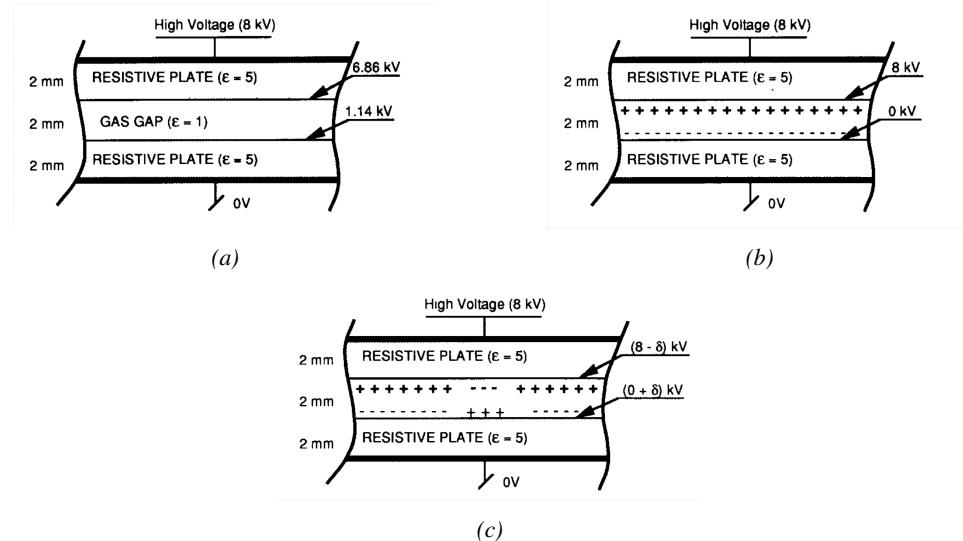


Figure 3.3: Movement of the charge carriers in an RPC. Figure 3.3a: Voltage across an RPC whose electrode have a relative permittivity of 5 at the moment the tension is applied. Figure 3.3b: After the charge carriers moved, the electrodes are charged and there is no voltage drop over the electrodes anymore. The full potential is applied on the gas gap only. Figure 3.3c: The streamer discharge initiated by a charged particle transports electrons and cations towards the anode and cathode respectively.

order of a few mv as shown on Figure 3.4, and requires amplification. This is the *avalanche mode* of RPC operation.

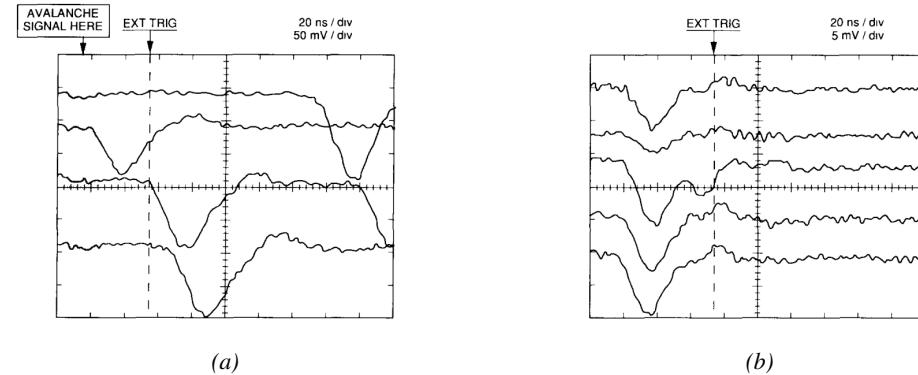


Figure 3.4: Typical oscilloscope pulses in streamer mode (Figure 3.4a) and avalanche mode (Figure 3.4b). In the case of streamer mode, the very small avalanche signal is visible.

This mode offers a higher rate capability by providing smaller discharges that don't affect the electrodes charge and are more locally contained in the gas volume as was demonstrated by Crotty with Figure 3.5 [7]. The detector only stays locally blind the time the charge carriers are recombined and there is no need for electrode recharge which is a long process affecting a large portion of the detector. Another advantage of avalanche signals over streamer is the great time consistency. Figure 3.4 shows very clearly that avalanche signals have a very small time jitter. This is a natural choice for high rate experiments.

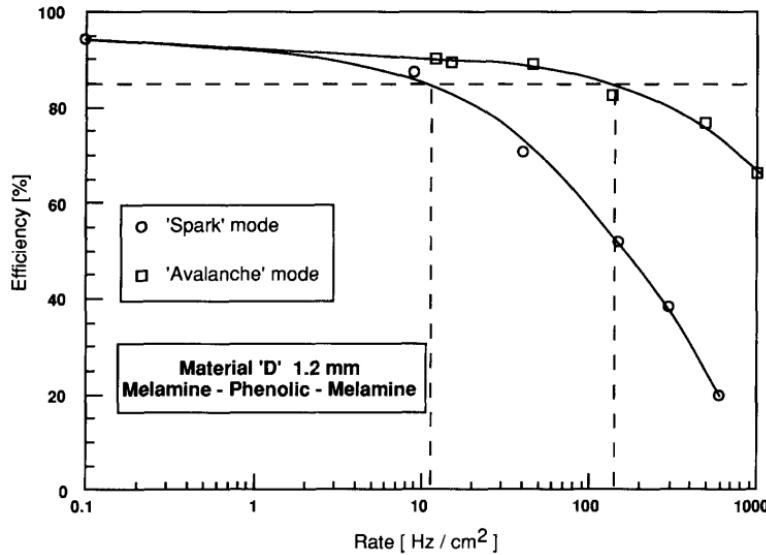


Figure 3.5: Rate capability comparison for the streamer and avalanche mode of operation. An order of magnitude in rate capability for a maximal efficiency drop of 10% is gained by using the avalanche mode over the streamer mode.

275 3.2.2 Detector designs and performance

276 Different RPC design have been used and each of them present its own advantages. Historically, the
 277 first type of RPC to have been developed is what is referred now to *no narrow gap* RPC [4, 23]. After
 278 the avalanche mode has been discovered [7], it has been proven that increasing the width of the gas
 279 gap lead to higher rate capability, due to lower charge deposition per avalanche, and lower power
 280 dissipation [23]. Nevertheless, by increasing the gas gap width, the time resolution of the detector
 281 decreases. This is a natural result if the increase of active gas volume in the detector is taken into
 282 account. Indeed, for a given threshold, only the small fraction of gas closest to the cathode will
 283 provide enough gain to have a detectable signal. In the case of a wider gas volume, the active
 284 region is then larger and a larger time jitter is introduced with the variation of starting position of the
 285 avalanche, as discussed in [10]. To solve improve both the time resolution and the rate capability,
 286 different methods were used trying to get advantages of both narrow and wide gap RPCs.

287 3.2.2.1 Double-gap RPC

288 Double-gap RPCs are made out of 2 narrow RPC detectors. The 2 RPC gaps are stacked on top of
 289 each other as shown in Figure 3.6. This detector layout, popularized by the two multipurpose experiments
 290 CMS [1] and ATLAS [15] at LHC, can be used as an OR system in which each individual
 291 chamber participates in the output signal. The gain of such a detector is greatly reduced with respect
 292 to single-gap RPCs with an efficiency plateau reached at lower voltage, as visible on Figure 3.7.

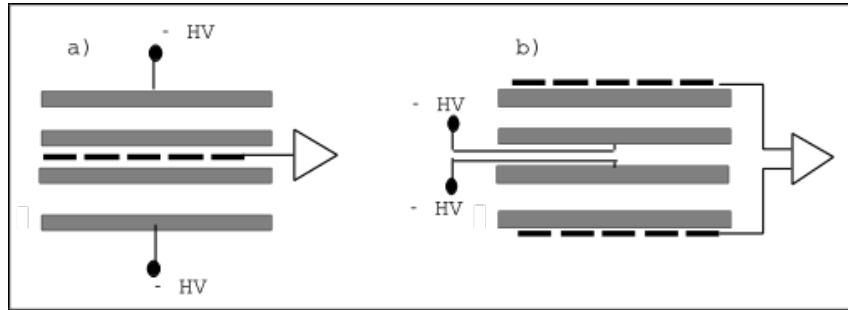


Figure 3.6: Possible double-gap RPC layouts: a) "standard" 1D double-gap RPC, as used in CMS experiment, where the anodes are facing each other and a 1D read-out plane is sandwiched between them, b) double read-out double-gap RPC as used in ATLAS experiment, where the cathodes are facing each other and 2 read-out planes are used on the outer surfaces. This last layout can offer the possibility to use a 2D reconstruction by using orthogonal read-out planes.

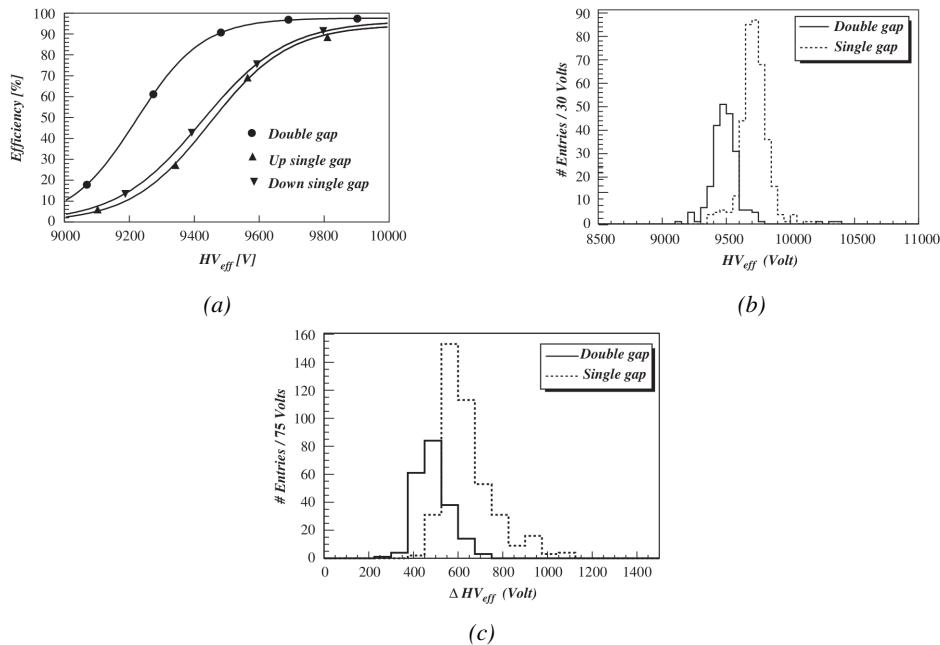


Figure 3.7: Comparison of performance of CMS double and single gap RPCs using cosmic muons [24]. Figure 3.7a: Comparison of efficiency sigmoids. Figure 3.7b: Voltage distribution at 95% of maximum efficiency. Figure 3.7c: $\Delta_{10\%}^{90\%}$ distribution.

293 3.2.2.2 Multigap RPC (MRPC)

294 MRPCs are layouts in which floating sub electrode plates are placed into a wide gap RPC to divide
 295 the gas volume and create a sum of narrow gaps [10, 11]. The time resolution of such a detector can
 296 reach of few tens of ps, with gas gaps of the order of a few hundred μm as shown in Figure 3.8
 297 representing ALICE Time-of-flight (ToF) MRPCs.

298 Sometimes used as a double multigap RPC, taking advantage of the OR of double gap RPCs,
 299 the MRPC is mainly used as ToF detector [25–29] due to its excellent timing properties that allow

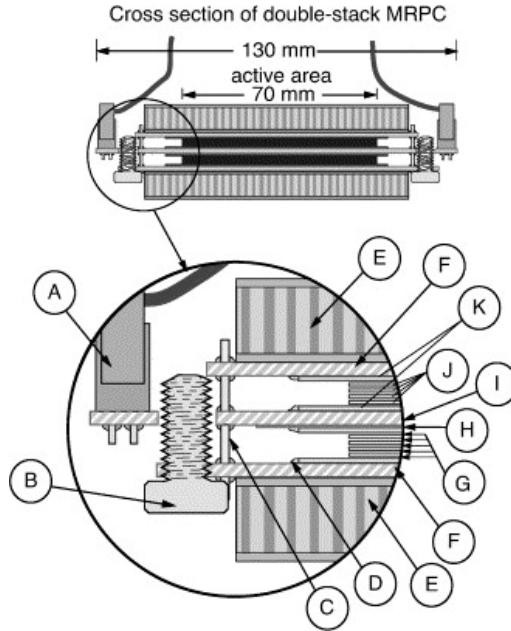


Figure 3.8: Presentation of ALICE MRPC using 250 μm gas gaps, 620 μm outer glass electrodes and 550 μm inner floating electrodes. More details on the labels are given in [25].

300 to perform particle identification as explained by Williams in [30]. The principle of particle iden-
 301 tification using ToF is simply the measurement of the velocity of a particle. Indeed, particles are
 302 defined by their mass (for the parameter of interest here, their electric charge being measured using
 303 the bending angle of the particles traveling through a magnetic field) and this mass can be calculated
 304 by measuring the velocity β and momentum of the particle:

$$\beta = \frac{p}{\sqrt{p^2 + m^2}} \quad (3.3)$$

305 Intuitively, it is trivial to understand that 2 different particles having the same momentum will
 306 have a different velocity due to the mass difference and thus a different flight time T_1 and T_2 through
 307 the detector and this is used to separate and identify particles. The better the time resolution of the
 308 ToF system used, the stronger will the separation be:

$$T = \frac{L}{v} = \frac{L}{c \cdot \beta}, \quad \Delta T = T_1 - T_2 = \frac{L}{c} \left(\sqrt{1 + m_1^2/p^2} - \sqrt{1 + m_2^2/p^2} \right) \cong (m_1^2 - m_2^2) \frac{L}{2cp^2} \quad (3.4)$$

309 An example of particle identification is given for the case of STAR experiment in Figure 3.9.
 310 Another benefit of using such small gas gaps is the strong reduction of the average avalanche
 311 volume and thus of the blind spot on MRPCs leading to an improved rate capability. Multigaps can
 312 sustain backgrounds of several kHz/cm² as demonstrated in Figure 3.10.

313 3.2.2.3 Charge distribution and performance limitations

314 The direct consequence of the different RPC layouts is a variation of intrinsic time resolution of the
 315 RPC as the gap size decreases. An advantage is given to multigaps whose design use sub-millimeter

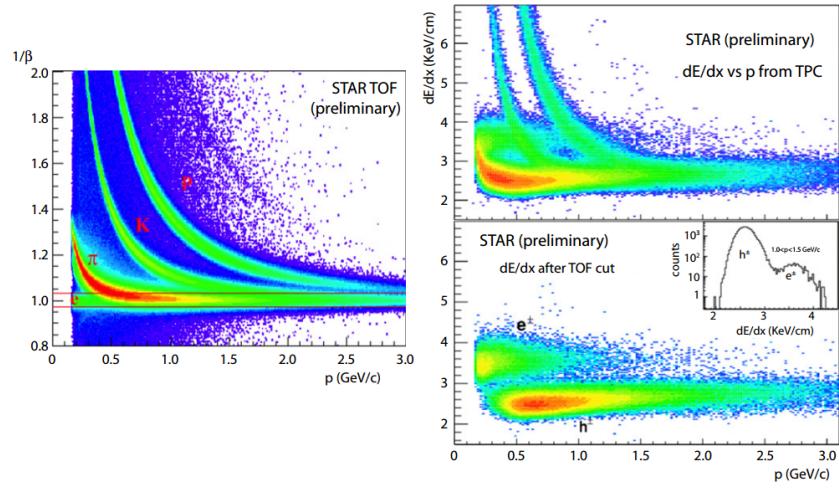


Figure 3.9: Particle identification applied to electrons in the STAR experiment. The identification is performed combining ToF and dE/dx measurements [30].

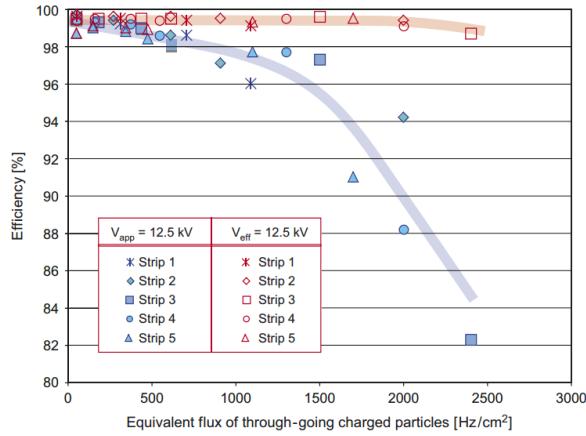


Figure 3.10: Comparison of the detector performance of ALICE ToF MRPC [31] at fixed applied voltage (in blue) and at fixed effective voltage (in red). The effective voltage is kept fixed by increasing the applied voltage accordingly to the current drawn by the detector.

316 gas volumes providing very consistent signals.

317 On the charge spectrum point of view, each layout has its own advantages. When the double-gap
 318 has the highest induced over drifting charge ratio, as seen in Figure 3.11, the multigap has a charge
 319 spectrum strongly detached from the origin, as visible in Figure 3.12. A high induced over drifting
 320 charge ratio means that the double gap can be safely operated at high threshold or that at similar
 321 threshold it can be operated with a twice smaller drifting charge, meaning a higher rate capability.
 322 On the other hand, the strong detachment of the charge spectrum from the origin in the MRPC case
 323 allows to reach a higher efficiency with increasing threshold as most of the induced charge is not low
 324 due to the convolution of several single gap spectra. The range of stable efficiency increases with
 325 the number of gap, as presented in Figure 3.13.

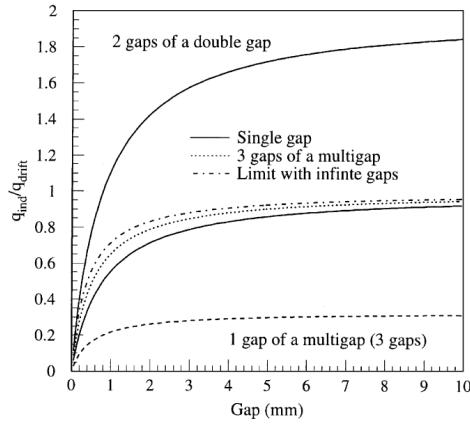


Figure 3.11: Ratio between total induced and drifting charge have been simulated for single gap, double-gap and multigap layouts [32]. The total induced charge for a double-gap RPC is a factor 2 higher than for a multigap.

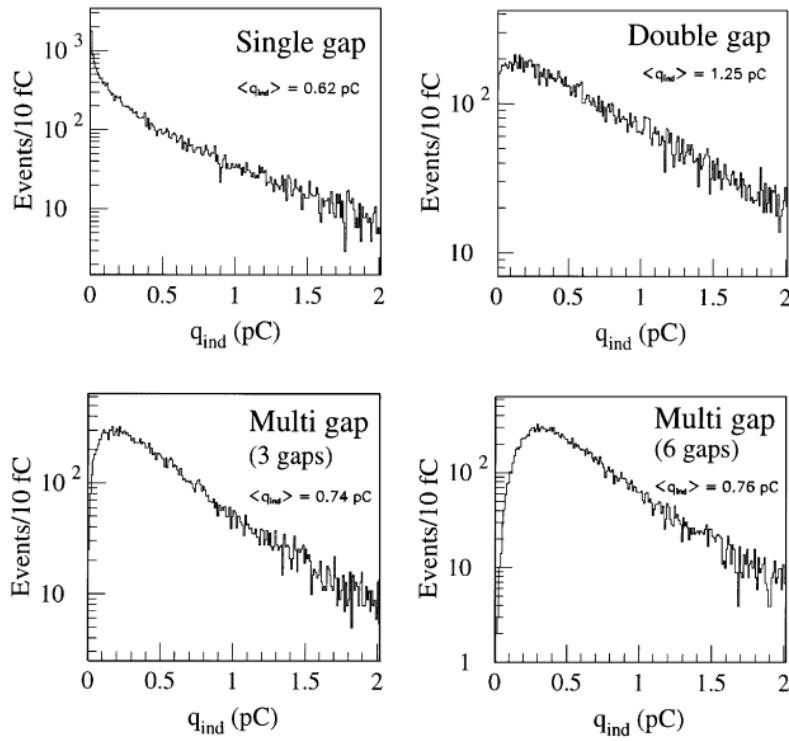


Figure 3.12: Charge spectra have been simulated for single gap, double-gap and multigap layouts [32]. It appears that when single gap shows a decreasing spectrum, double and multigap layouts exhibit a spectrum whose peak is detached from the origin. The detachment gets stronger as the number of gaps increases.

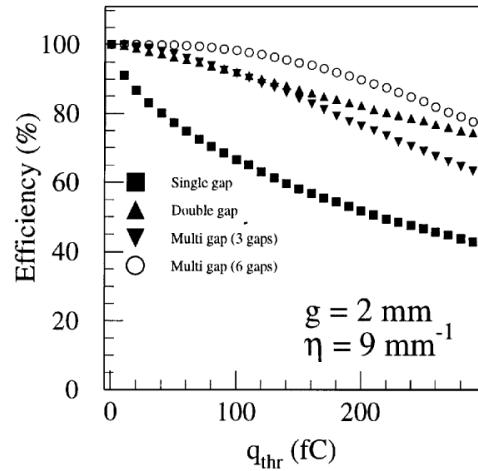


Figure 3.13: The maximal theoretical efficiency is simulated for single gap, double-gap and multigap layouts [32] at a constant gap thickness of 2 mm and using an effective Townsend coefficient of 9 mm^{-1} .

326 3.3 Signal formation

327 3.4 Gas transport parameters

4

328

329

330

Longevity studies and Consolidation of the present CMS RPC subsystem

331

4.1 Resistive Plate Chambers at CMS

332

4.1.1 Overview

333

The Resistive Plate Chambers (RPC) system, located in both barrel and endcap regions, provides a fast, independent muon trigger with a looser p_T threshold over a large portion of the pseudorapidity range ($|\eta| < 1.6$) [add reconstruction].

336

337

During High-Luminosity LHC (HL-LHC) operations the expected conditions in terms of background and pile-up will make the identification and correct P_T assignment a challenge for the Muon system. The goal of RPC upgrade is to provide additional hits to the Muon system with precise timing. All these informations will be elaborated by the trigger system in a global way enhancing the performance of the trigger in terms of efficiency and rate control. The RPC Upgrade is based on two projects: an improved Link Board System and the extension of the RPC coverage up to $|\eta| = 2.4$. [FIXME 2.4 or 2.5?]

344

345

The Link Board system, that will be described in section xxx, is responsible to process, synchronize and zero-suppress the signals coming from the RPC front end boards. The Link Board components have been produced between 2006 and 2007 and will be subjected to aging and failure in the long term. The upgraded Link Board system will overcome the aging problems described in section xxx and will allow for a more precise timing information to the RPC hits from 25 to 1 ns [ref section xxx].

350

351

352

353

354

The extension of the RPC system up to $|\eta| = 2.1$ was already planned in the CMS TDR [ref cmstdr] and staged because of budget limitations and expected background rates higher than the rate capability of the present CMS RPCs in that region. An extensive R&D program has been done in order to develop an improved RPC that fulfills the CMS requirements. Two new RPC layers in the innermost ring of stations 3 and 4 will be added with benefits to the neutron-induced background

355 reduction and efficiency improvement for both trigger and offline reconstruction.

356 4.1.2 The present RPC system

357 The RPC system is organized in 4 stations called RB1 to RB4 in the barrel region, and RE1 to RE4
 358 in the endcap region. The innermost barrel stations, RB1 and RB2, are instrumented with 2 layers
 359 of RPCs facing the innermost (RB1in and RB2in) and outermost (RB1out and RB2out) sides of the
 360 DT chambers. Every chamber is then divided from the read-out point of view into 2 or 3 η partitions
 361 called “rolls”. The RPC system consist of 480 barrel chambers and 576 endcap chambers. Details
 362 on the geometry are discussed in the paper [ref to geo paper].

363 The CMS RPC chamber is a double-gap, operated in avalanche mode to ensure reliable operation
 364 at high rates. Each RPC gap consists of two 2-mm-thick resistive High-Pressure Laminate (HPL)
 365 plates separated by a 2-mm-thick gas gap. The outer surface of the HPL plates is coated with a thin
 366 conductive graphite layer, and a voltage is applied. The RPCs are operated with a 3-component,
 367 non-flammable gas mixture consisting of 95.2% freon ($C_2H_2F_4$, known as R134a), 4.5% isobutane
 368 ($i-C_4H_{10}$), and 0.3% sulphur hexafluoride (SF_6) with a relative humidity of 40% - 50%. Readout
 369 strips are aligned in η between the 2 gas gaps. [\[Add a sentence on FEBs.\]](#)

370 The discriminated signals coming from the Front End boards feed via twisted cables (10 to 20 m
 371 long) the Link Board System located in UXC on the balconies around the detector. The Link System
 372 consist of the 1376 Link Boards (LBs) and the 216 Control Boards (CBs), placed in 108 Link Boxes.
 373 The Link Box is a custom crate (6U high) with 20 slots (for two CBs and eighteen LBs). The Link
 374 Box contains custom backplane to which the cables from the chambers are connected, as well as the
 375 cables providing the LBs and CBs power supply and the cables for the RPC FEBs control with use
 376 of the I2C protocol (trough the CB). The backplane itself contains only connectors (and no any other
 377 electronic devices).

378 The Link Board has 96 input channels (one channel corresponds to one RPC strip). The input
 379 signals are the ~ 100 ns binary pulses which are synchronous to the RPC hits, but not to the LHC
 380 clock (which drives the entire CMS electronics). Thus the first step of the FEB signals processing
 381 is synchronization, i.e. assignment of the signals to the BXes (25 ns periods). Then the data are
 382 compressed with a simple zero-suppressing algorithm (the input channels are grouped into 8 bit
 383 partitions, only the partitions with at least one nonzero bit are selected for each BX). Next, the non-
 384 empty partitions are time-multiplexed i.e. if there are more than one such partition in a given BX,
 385 they are sent one-by-one in consecutive BXes. The data from 3 neighbouring LBs are concentrated
 386 by the middle LB which contains the optical transmitter for sending them to the USC over a fiber at
 387 1.6 Gbps.

388 The Control Boards provide the communication of the control software with the LBs via the
 389 FEC/CCU system. The CBs are connected into token rings, each ring consists of 12 CBs of one
 390 detector tower and a FEC mezzanine board placed on the CCS board located in the VME crate in
 391 the USC. In total, there are 18 rings in the entire Link System. The CBs also perform automatic
 392 reloading of the LB's firmware which is needed in order to avoid accumulation of the radiation
 393 induced SEUs in the LBs firmware.

394 Both LBs and CB are based on the Xilinx Spartan III FPGAs, the CB additionally contains
 395 radiation-tolerant (FLASH based) FPGA Actel ProAsicPlus.

396 The High Voltage power system is located in USC, not exposed to radiation and easily accessible
 397 for any reparation. A single HV channel powers 2 RPC chambers both in the barrel and endcap
 398 regions. The Low Voltage boards are located in UXC on the balconies and provide the voltage to the

399 front end electronics.

400 4.1.3 Pulse processing of CMS RPCs

401 Signals induced by cosmic particle in the RPC strips are shaped by standard CMS RPC Front-End
 402 Electronics (FEE) following the scheme of Figure 4.1. On a first stage, analogic signals are amplified
 403 and then sent to the Constant Fraction Discriminator (CFD) described in Figure 4.2. At the end of
 404 the chain, 100 ns long pulses are sent in the LVDS output. These output signal are sent on one side to
 405 a V1190A Time-to-Digital Converter (TDC) module from CAEN and on the other to an OR module
 406 to count the number of detected signals. Trigger and hit coïncidences are monitored using scalers.
 407 The TDC is used to store the data into ROOT files. These files are thus analysed to understand the
 408 detectors performance.

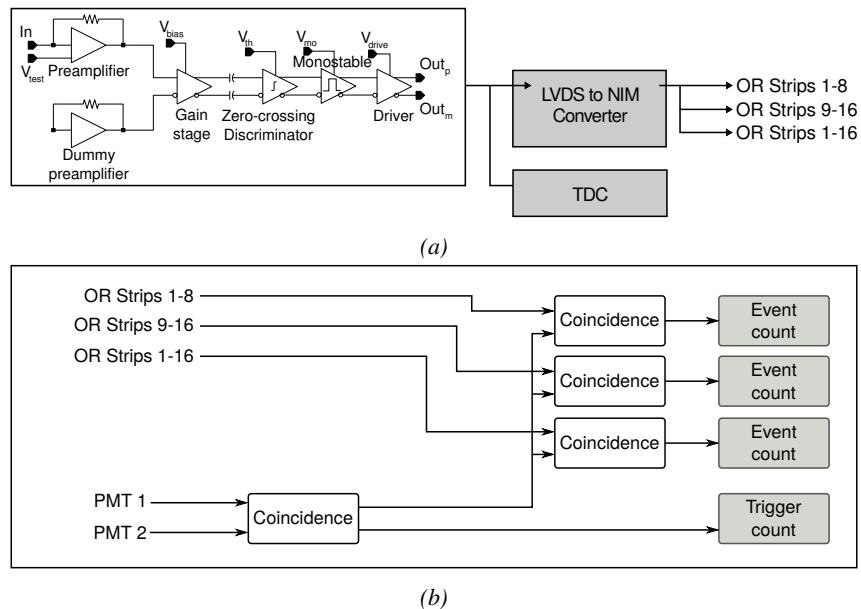


Figure 4.1: Signals from the RPC strips are shaped by the FEE described on Figure 4.1a. Output LVDS signals are then read-out by a TDC module connected to a computer or converted into NIM and sent to scalers. Figure 4.1b describes how these converted signals are put in coincidence with the trigger.

409 4.2 Testing detectors under extreme conditions

410 The upgrade from LHC to HL-LHC will increase the peak luminosity from $10^{34} \text{ cm}^{-2} \text{ s}^{-1}$ to reach
 411 $5 \times 10^{34} \text{ cm}^{-2} \text{ s}^{-1}$, increasing in the same way the total expected background to which the RPC
 412 system will be subjected to. Composed of low energy gammas and neutrons from p - p collisions, low
 413 momentum primary and secondary muons, puch-through hadrons from calorimeters, and particles
 414 produced in the interaction of the beams with collimators, the background will mostly affect the
 415 regions of CMS that are the closest to the beam line, i.e. the RPC detectors located in the endcaps.
 416 [To update.]

417

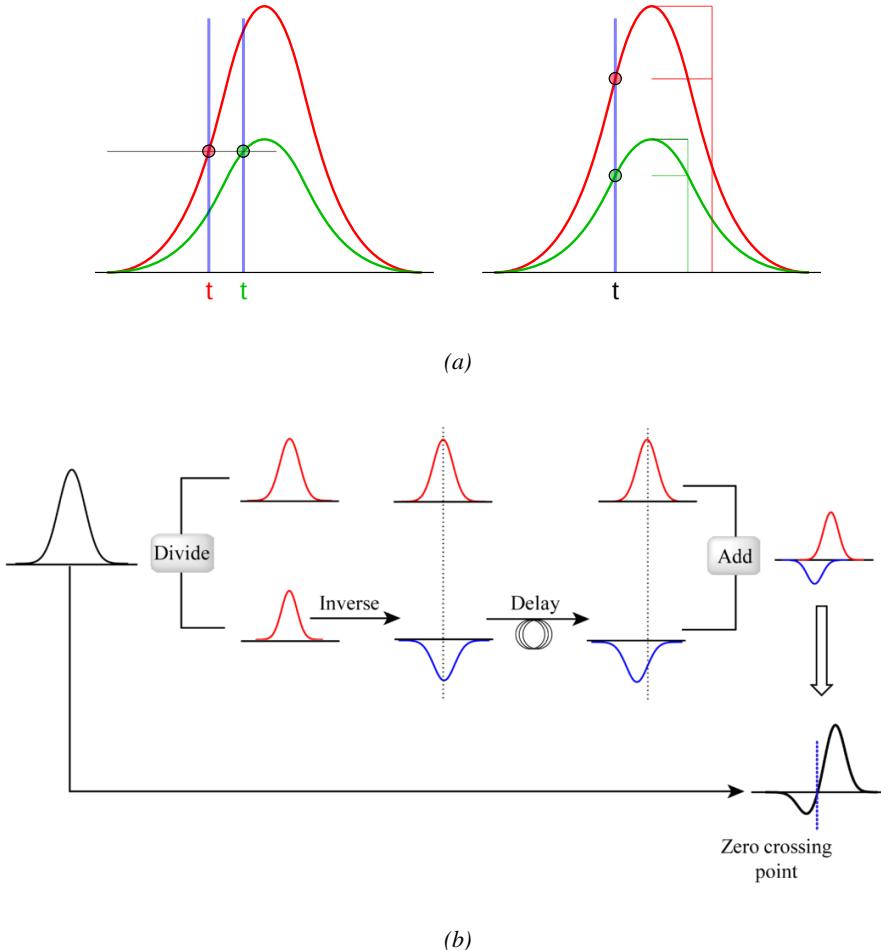
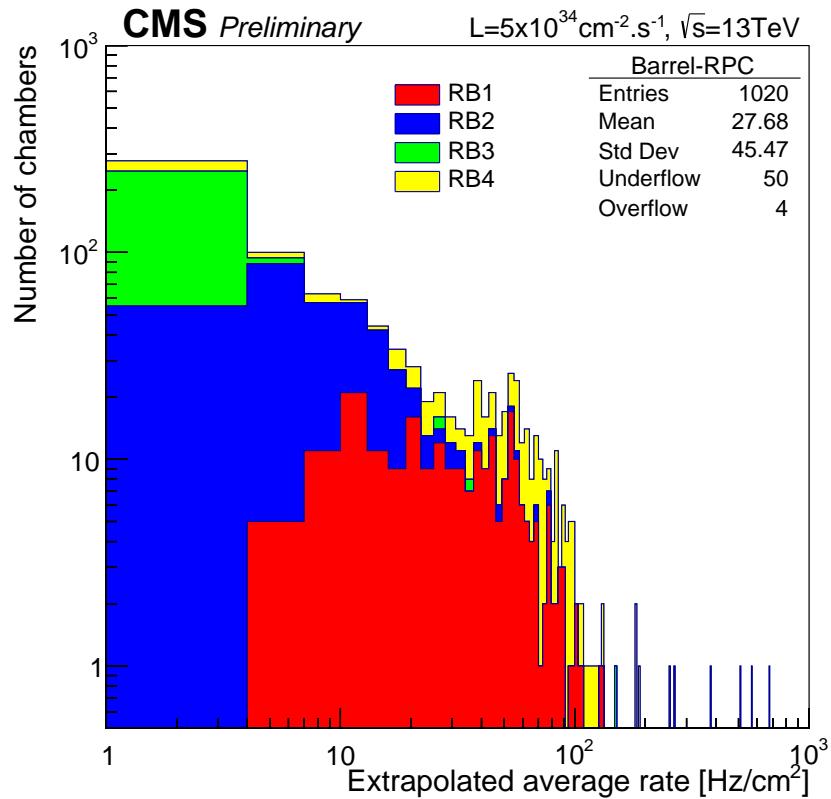
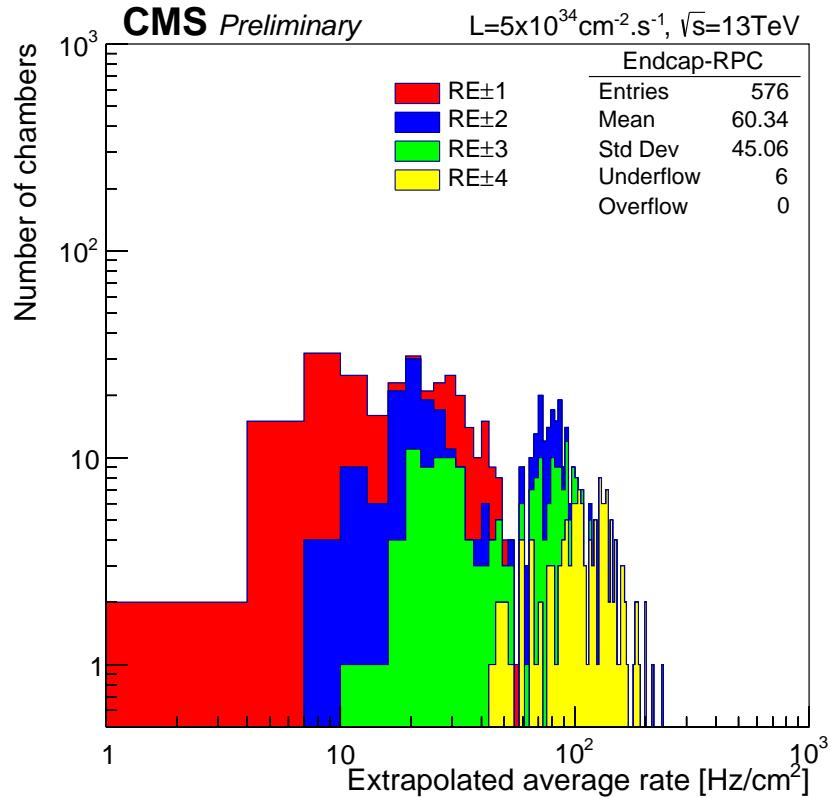


Figure 4.2: Description of the principle of a CFD. A comparison of threshold triggering (left) and constant fraction triggering (right) is shown in Figure 4.2a. Constant fraction triggering is obtained thanks to zero-crossing technique as explained in Figure 4.2b. The signal arriving at the input of the CFD is split into three components. A first one is delayed and connected to the inverting input of a first comparator. A second component is connected to the noninverting input of this first comparator. A third component is connected to the noninverting input of another comparator along with a threshold value connected to the inverting input. Finally, the output of both comparators is fed through an AND gate.

418 The 2016 data allowed to study the values of the background rate in all RPC system. In Figure
 419 the distribution of the chamber background hit rate per unit area is shown at a luminosity
 420 of $5 \times 10^{34} \text{ cm}^{-2} \cdot \text{s}^{-1}$ linearly extrapolating from data collected in 2016 [ref mentioning the linear
 421 dependency of rate vs lumi]. The maximum rate per unit area at HL-LHC conditions is expected to
 422 be of the order of 600 Hz/cm^2 (including a safety factor 3). Nevertheless, Fluka simulations have
 423 conducted in order to understand the background at HL-LHC conditions. The comparison to the
 424 data has shown, in Figure 4.4, a discrepancy of a factor 2 even though the order of magnitude is
 425 consistent. [Understand mismatch.]



(a)



(b)

Figure 4.3: (4.3a) Extrapolation from 2016 data of single hit rate per unit area in the barrel region. (4.3b) Extrapolation from 2016 data of single hit rate per unit area in the endcap region.

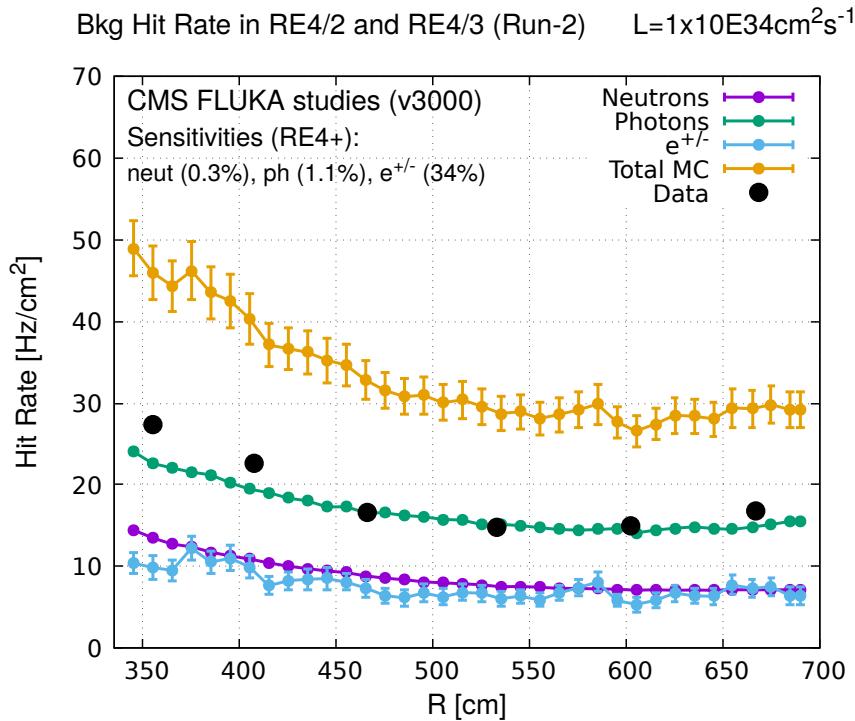


Figure 4.4: Background Fluka simulation compared to 2016 Data at $L = 10^{34} \text{ cm}^{-2} \cdot \text{s}^{-1}$ in the fourth endcap disk region. A mismatch in between simulation and data can be observed. [\[To be understood.\]](#)

In the past, extensive long-term tests were carried out at several gamma and neutron facilities certifying the detector performance. Both full size and small prototype RPCs have been irradiated with photons up to an integrated charge of $\sim 0.05 \text{ C/cm}^2$ and $\sim 0.4 \text{ C/cm}^2$, respectively [33, 34]. During Run-I, the RPC system provided stable operation and excellent performance and did not show any aging effects for integrated charge of the order of 0.01 C/cm^2 . Projections on currents from 2016 Data, has allowed to determine that the total integrated charge, by the end of HL-LHC, would be of the order of 1 C/cm^2 (including a safety factor 3). [\[Corresponding figure needed.\]](#)

434

4.2.1 The Gamma Irradiation Facilities

4.2.1.1 GIF

Located in the SPS West Area at the downstream end of the X5 test beam, the Gamma Irradiation Facility (GIF) was a test area in which particle detectors were exposed to a particle beam in presence of an adjustable gamma background [35]. Its goal was to reproduce background conditions these detectors would suffer in their operating environment at LHC. GIF layout is shown in Figure 4.5. Gamma photons are produced by a strong ^{137}Cs source installed in the upstream part of the zone inside a lead container. The source container includes a collimator, designed to irradiate a $6 \times 6 \text{ m}^2$ area at 5 m maximum to the source. A thin lens-shaped lead filter helps providing with a uniform outgoing flux in a vertical plane, orthogonal to the beam direction. The principal collimator hole provides a pyramidal aperture of $74^\circ \times 74^\circ$ solid angle and provides a photon flux in a pyramidal vol-

ume along the beam axis. The photon rate is controled by further lead filters allowing the maximum rate to be limited and to vary within a range of four orders of magnitude. Particle detectors under test are then placed within the pyramidal volume in front of the source, perpendicularly to the beam line in order to profit from the homogeneous photon flux. Adjusting the background flux of photons can then be done by using the filters and choosing the position of the detectors with respect to the source.

451

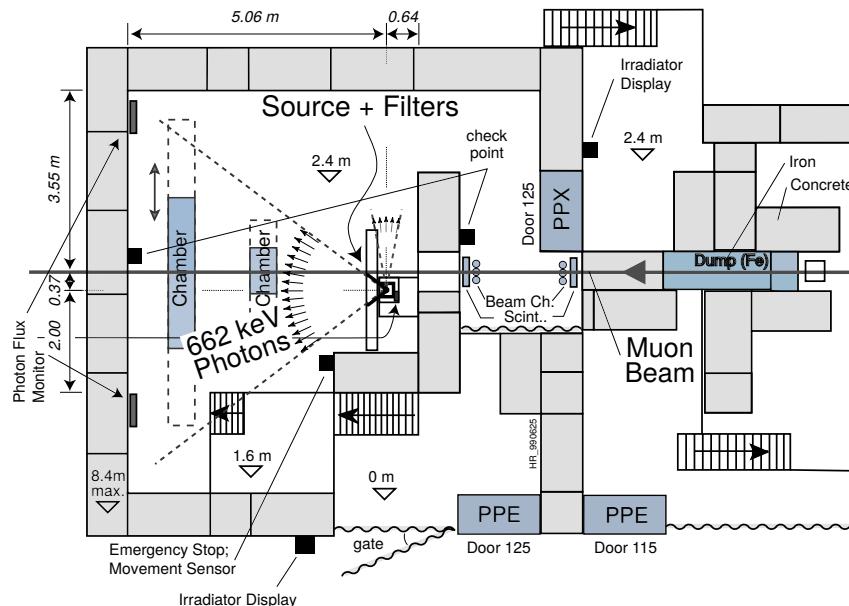


Figure 4.5: Layout of the test beam zone called X5c GIF at CERN. Photons from the radioactive source produce a sustained high rate of random hits over the whole area. The zone is surrounded by 8 m high and 80 cm thick concrete walls. Access is possible through three entry points. Two access doors for personnel and one large gate for material. A crane allows installation of heavy equipment in the area.

As described on Figure 4.6, the ^{137}Cs source emits a 662 keV photon in 85% of the decays. An activity of 740 GBq was measured on the 5th March 1997. To estimate the strength of the flux in 2014, it is necessary to consider the nuclear decay through time assiciated to the Cesium source whose half-life is well known ($t_{1/2} = (30.05 \pm 0.08)$ y). The GIF tests where done in between the 20th and the 31st of August 2014, i.e. at a time $t = (17.47 \pm 0.02)$ y resulting in an attenuation of the activity from 740 GBq in 1997 to 494 GBq in 2014.

458

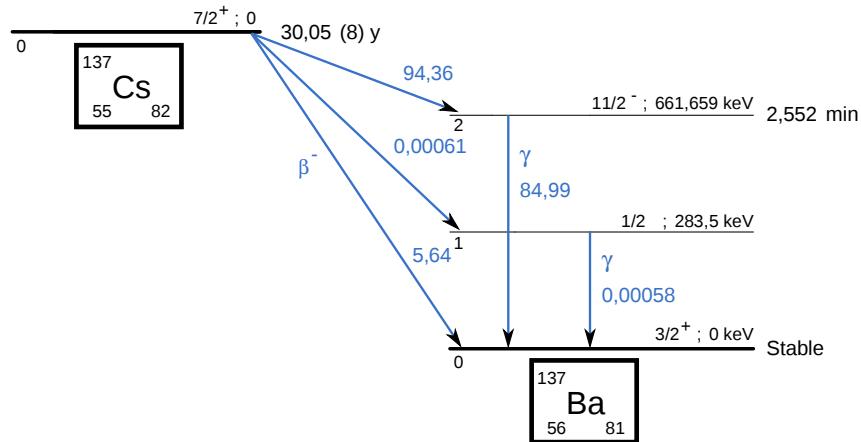


Figure 4.6: ^{137}Cs decays by β^- emission to the ground state of ^{137}Ba (BR = 5.64%) and via the 662 keV isomeric level of ^{137}Ba (BR = 94.36%) whose half-life is 2.55 min.

4.2.1.2 GIF++

The new Gamma Irradiation Facility (GIF++), located in the SPS North Area at the downstream end of the H4 test beam, has replaced its predecessor during LS1 and has been operational since spring 2015 [36]. Like GIF, GIF++ features a ^{137}Cs source of 662 keV gamma photons, their fluence being controlled with a set of filters of various attenuation factors. The source provides two separated large irradiation areas for testing several full-size muon detectors with continuous homogeneous irradiation, as presented in Figure 4.7.

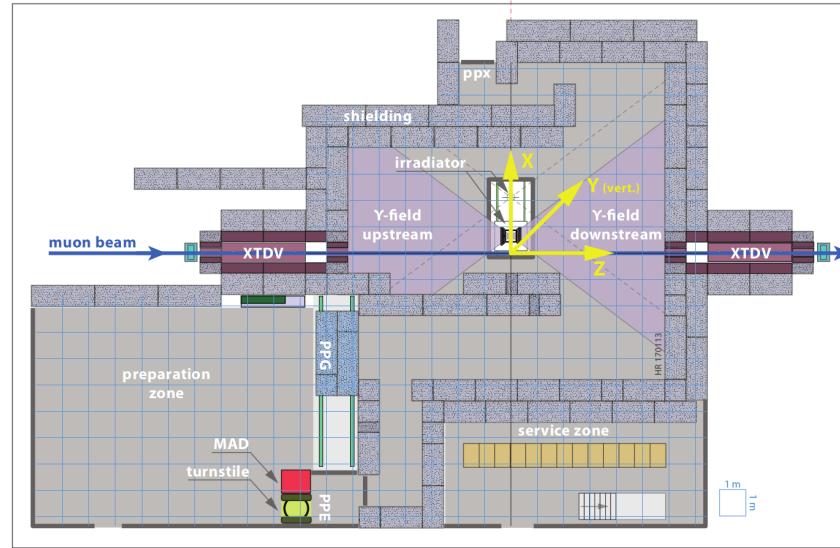


Figure 4.7: Floor plan of the GIF++ facility. When the facility downstream of the GIF++ takes electron beam, a beam pipe is installed along the beam line (z -axis). The irradiator can be displaced laterally (its center moves from $x = 0.65 \text{ m}$ to 2.15 m), to increase the distance to the beam pipe.

467 The source activity was measured to be about 13.5 TBq in March 2016. The photon flux being
 468 far greater than HL-LHC expectations, GIF++ provides an excellent facility for accelerated aging
 469 tests of muon detectors.

470

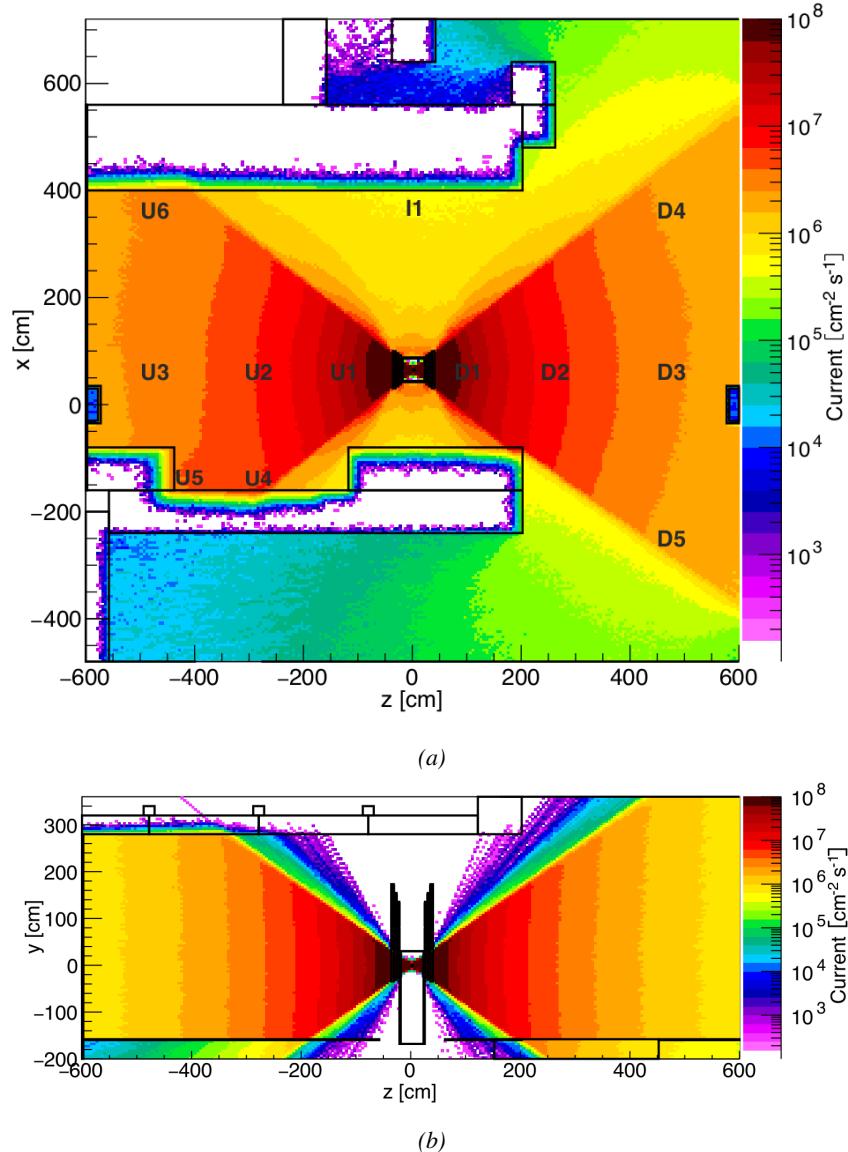


Figure 4.8: Simulated unattenuated current of photons in the xz plane (Figure 4.8a) and yz plane (Figure 4.8b) through the source at $x = 0.65 \text{ m}$ and $y = 0 \text{ m}$. With angular correction filters, the current of 662 keV photons is made uniform in xy planes.

471 The source is situated in the muon beam line with the muon beam being available a few times a
 472 year. The H4 beam, composed of muons with a momentum of about 150 GeV/c, passes through the
 473 GIF++ zone and is used to study the performance of the detectors. Its flux is of 104 particles/ s cm^2

474 focused in an area similar to $10 \times 10 \text{ cm}^2$. Therefore, with properly adjusted filters, one can imitate
 475 the HL-LHC background and study the performance of muon detectors with their trigger/readout
 476 electronics in HL-LHC environment.

477

478 4.3 Preliminary tests at GIF

479 4.3.1 Resistive Plate Chamber test setup

480 During summer 2014, preliminary tests have been conducted in the GIF area on a newly produced
 481 RE4/2 chamber labelled RE-4-2-BARC-161. This chamber has been placed into a trolley covered
 482 with a tent. The position of the RPC inside the tent and of the tent related to the source is described
 483 in Figure 4.9. To test this CMS RPC, three different absorber settings were used. First of all,
 484 measurements were done with fully opened source. Then, to complete this preliminary study, the
 485 gamma flux has been attenuated from a factor 2 and a factor 5. The expected gamma flux at the level
 486 of our detector will be discussed in subsection ??.

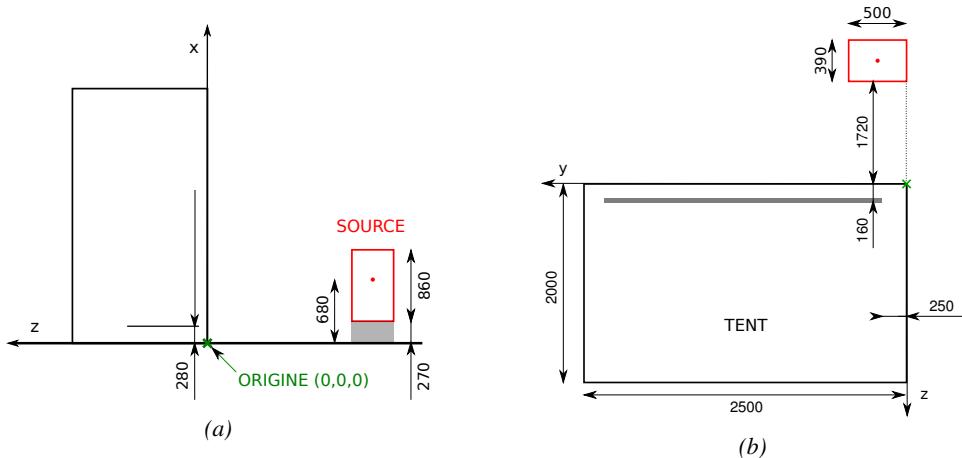


Figure 4.9: Description of the RPC setup. Dimensions are given in mm. A tent containing RPCs is placed at 1720 mm from the source container. The source is situated in the center of the container. RE-4-2-BARC-161 chamber is 160 mm inside the tent. This way, the distance between the source and the chambers plan is 2060 mm. Figure 4.9a provides a side view of the setup in the xz plane while Figure 4.9b shows a top view in the yz plane.



Figure 4.10: RE-4-2-BARC-161 chamber is inside the tent as described in Figure 4.9. In the top right, the two scintillators used as trigger can be seen. This trigger system has an inclination of 10° relative to horizontal and is placed above half-partition B2 of the RPCs. PMT electronics are shielded thanks to lead blocks placed in order to protect them without stopping photons from going through the scintillators and the chamber.

487 At the time of the tests, the beam not being operationnal anymore, a trigger composed of 2 plastic
 488 scintillators has been placed in front of the setup with an inclination of 10 deg with respect to the
 489 detector plane in order to look at cosmic muons. Using this particular trigger layout, shown on Fig-
 490 ure 4.10, leads to a cosmic muon hit distribution into the chamber similar to the one in Figure 4.11.
 491 Measured without gamma irradiation, two peaks can be seen on the profil of partition B, centered
 492 on strips 52 and 59. Section ?? will help us understand that these two peaks are due respectively to
 493 forward and backward coming cosmic particles where forward coming particles are first detected by
 494 the scintillators and then the RPC while the backward coming muons are first detected in the RPC.

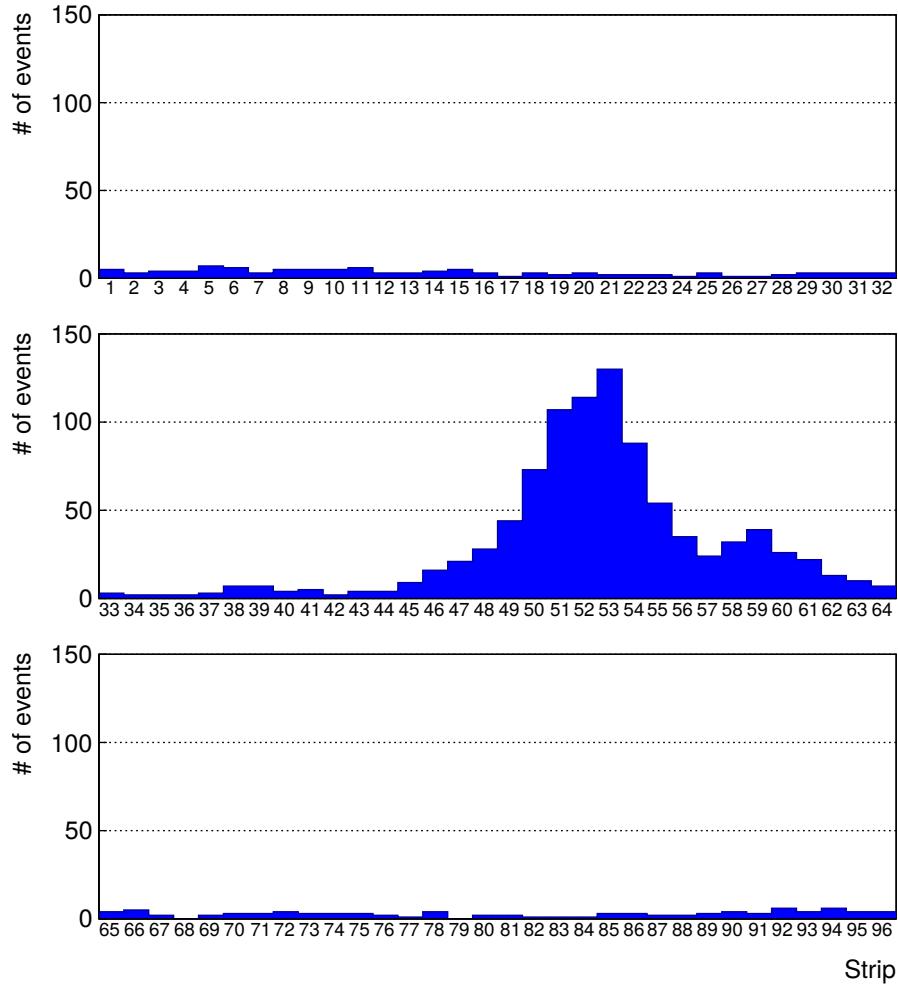


Figure 4.11: Hit distributions over all 3 partitions of RE-4-2-BARC-161 chamber is showed on these plots. Top, middle and bottom figures respectively correspond to partitions A, B, and C. These plots show that some events still occur in other half-partitions than B2, which corresponds to strips 49 to 64, in front of which the trigger is placed, contributing to the inefficiency of detection of cosmic muons. In the case of partitions A and C, the very low amount of data can be interpreted as noise. On the other hand, it is clear that a little portion of muons reach the half-partition B1, corresponding to strips 33 to 48.

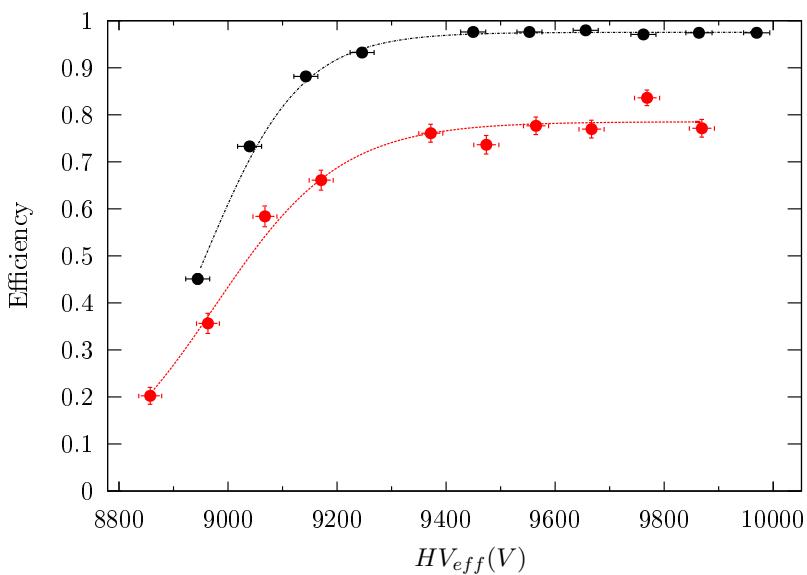
495 4.3.2 Data Acquisition

496 4.3.3 Geometrical acceptance of the setup layout to cosmic muons

497 In order to profit from a constant gamma irradiation, the detectors inside of the GIF bunker need
 498 to be placed in a plane orthogonal to the beam line. The muon beam that used to be available was
 499 meant to test the performance of detectors under test. This beam not being active anymore, another
 500 solution to test detector performance had to be used. Thus, it has been decided to use cosmic muons
 501 detected through a telescope composed of two scintillators. Lead blocks were used as shielding to

502 protect the photomultipliers from gammas as can be seen from Figure 4.10.

503 An inclination has been given to the cosmic telescope to maximize the muon flux. A good com-
 504 promise had to be found between good enough muon flux and narrow enough hit distribution to
 505 be sure to contain all the events into only one half partitions as required from the limited available
 506 readout hardware. Nevertheless, a consequence of the misplaced trigger, that can be seen as a loss
 507 of events in half-partition B1 in Figure 4.11, is an inefficiency. Nevertheless, the inefficiency of ap-
 508 proximately 20 % highlighted in Figure 4.12 by comparing the performance of chamber BARC-161
 509 in 904 and at GIF without irradiation seems too important to be explained only by the geometri-
 510 cal acceptance of the setup itself. Simulations have been conducted to show how the setup brings
 511 inefficiency.



512 *Figure 4.12: Results are derived from data taken on half-partition B2 only. On the 18th of June 2014, data
 513 has been taken on chamber RE-2-BARC-161 at building 904 (Prevessin Site) with cosmic muons providing us a
 514 reference efficiency plateau of $(97.54 \pm 0.15)\%$ represented by a black curve. A similar measurement has been
 515 done at GIF on the 21st of July with the same chamber giving a plateau of $(78.52 \pm 0.94)\%$ represented by a
 516 red curve.*

517 4.3.3.1 Description of the simulation layout

518 The layout of GIF setup has been reproduced and incorporated into a Monte Carlo (MC) simulation
 519 to study the influence of the disposition of the telescope on the final distribution measured by the
 520 RPC. A 3D view of the simulated layout is given into Figure 4.13. Muons are generated randomly
 521 in a horizontal plane located at a height corresponding to the lowest point of the PMTs. This way,
 522 the needed size of the plane in order to simulate events happening at very big azimuthal angles (i.e.
 $\theta \approx \pi$) can be kept relatively small. The muon flux is designed to follow the usual $\cos^2\theta$ distribution
 for cosmic particle. The goal of the simulation is to look at muons that pass through the muon
 telescope composed of the two scintillators and define their distribution onto the RPC plane. During
 the reconstruction, the RPC plane is then divided into its strips and each muon track is assigned to a
 strip.

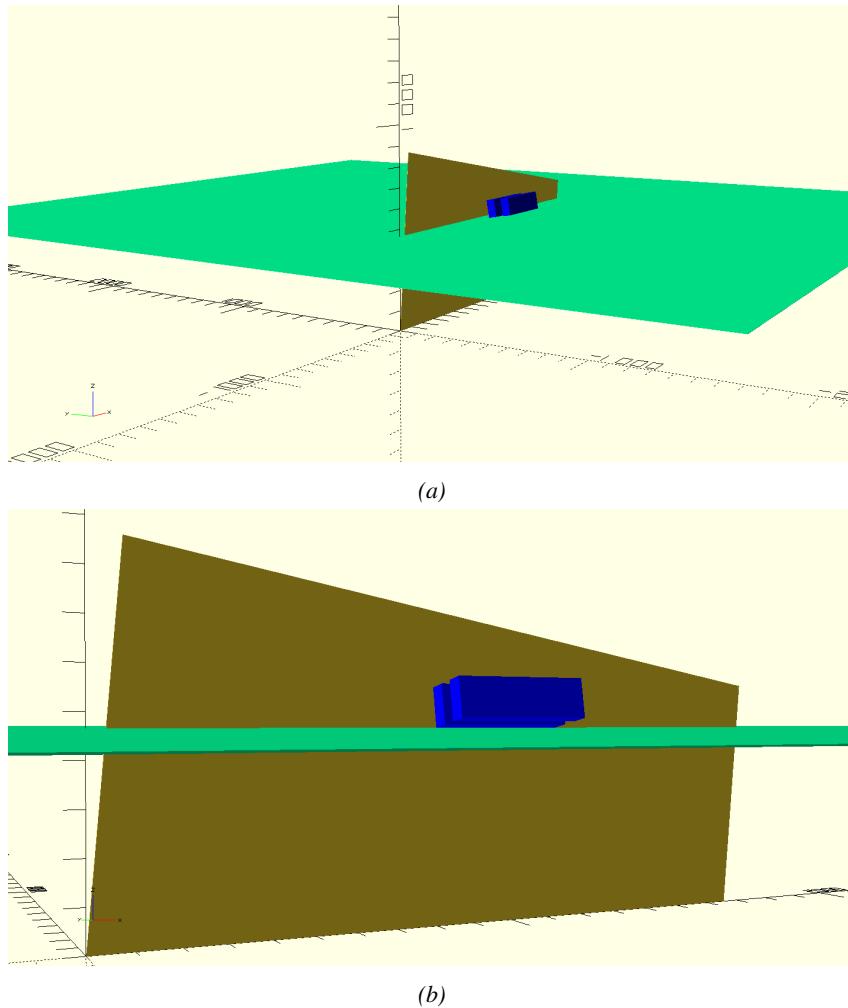


Figure 4.13: Representation of the layout used for the simulations of the test setup. The RPC is represented as a yellow trapezoid while the two scintillators as blue cuboids looking at the sky. A green plane corresponds to the muon generation plane within the simulation. Figure 4.9a shows a global view of the simulated setup. Figure 4.9b shows a zoomed view that allows to see the 2 scintillators as well as the full RPC plane.

523 In order to further refine the quality of the simulation and understand deeper the results the
 524 dependance of the distribution has been studied for a range of telescope inclinations. Moreover,
 525 the threshold applied on the PMT signals has been included into the simulation in the form of a
 526 cut. In the approximation of uniform scintillators, it has been considered that the threshold can be
 527 understood as the minimum distance particles need to travel through the scintillating material to give
 528 a strong enough signal. Particles that travel a distance smaller than the set "threshold" are thus not
 529 detected by the telescope and cannot trigger the data taking. Finally, the FEE threshold also has
 530 been considered in a similar way. The mean momentum of horizontal cosmic rays is higher than
 531 those of vertical ones but the stopping power of matter for momenta ranging from 1 GeV to 1 TeV
 532 stays comparable. It is then possible to assume that the mean number of primary e^- /ion pairs per
 533 unit length will stay similar and thus, depending on the applied discriminator threshold, muons with

534 the shortest path through the gas volume will deposit less charge and induce a smaller signal on the
 535 pick-up strips that could eventually not be detected. These two thresholds also restrain the overall
 536 geometrical acceptance of the system.

537 **4.3.3.2 Simulation procedure**

538 The simulation software has been designed using C++ and the output data is saved into ROOT
 539 histograms. Simulations start for a threshold T_{scint} varying in a range from 0 to 45 mm in steps
 540 of 5 mm, where $T_{scint} = 0$ mm corresponds to the case where there isn't any threshold apply on
 541 the input signal while $T_{scint} = 45$ mm, which is the scintillator thickness, is the case where muons
 542 cannot arrive orthogonally onto the scintillator surface. For a given T_{scint} , a set of RPC thresholds
 543 are considered. The RPC threshold, T_{RPC} varies from 2 mm, the thickness of the gas volume, to
 544 3 mm in steps of 0.25 mm. For each $(T_{scint}; T_{RPC})$ pair, $N_\mu = 10^8$ muons are randomly generated
 545 inside the muon plane described in the previous paragraph with an azimuthal angle θ chosen to follow
 546 a $\cos^2\theta$ distribution.

547 Planes are associated to each surface of the scintillators. Knowing muon position into the muon
 548 plane and its direction allows us, by assuming that muons travel in a straight line, to compute the
 549 intersection of the muon track with these planes. Applying conditions to the limits of the surfaces
 550 of the scintillator faces then gives us an answer to whether or not the muon passed through the
 551 scintillators. In the case the muon has indeed passed through the telescope, the path through each
 552 scintillator is computed and muons whose path was shorter than T_{scint} are rejected and are thus
 553 considered as having not interacted with the setup.

554 On the contrary, if the muon is labeled as good, its position within the RPC plane is computed
 555 and the corresponding strip, determined by geometrical tests in the case the distance through the
 556 gas volume was enough not to be rejected because of T_{RPC} , gets a hit and several histograms
 557 are filled in order to keep track of the generation point on the muon plane, the intersection points
 558 of the reconstructed muons within the telescope, or on the RPC plane, the path traveled through
 559 each individual scintillator or the gas volume, as well as other histograms. Moreover, muons fill
 560 different histograms whether they are forward or backward coming muons. They are discriminated
 561 according to their direction components. When a muon is generated, an (x, y, z) position is assigned
 562 into the muon plane as well as a $(\theta; \phi)$ pair that gives us the direction it's coming from. This way,
 563 muons satisfying the condition $0 \leq \phi < \pi$ are designated as backward coming muons while muons
 564 satisfying $\pi \leq \phi < 2\pi$ as forward coming muons.

565 This simulation is then repeated for different telescope inclinations ranging in between 4 and 20°
 566 and varying in steps of 2°. Due to this inclination and to the vertical position of the detector under
 567 test, the muon distribution reconstructed in the detector plane is asymmetrical. The choice as been
 568 made to chose a skew distribution formula to fit the data built as the multiplication of gaussian and
 569 sigmoidal curves together. A typical gaussian formula is given as 4.1 and has three free parameters
 570 as A_g , its amplitude, \bar{x} , its mean value and σ , its root mean square. Sigmoidal curves as given by
 571 formula 4.2 are functions converging to 0 and A_s as x diverges. The inflection point is given as x_i
 572 and λ is proportional to the slope at $x = x_i$. In the limit where $\lambda \rightarrow \infty$, the sigmoid becomes a
 573 step function.

$$g(x) = A_g e^{-\frac{(x-\bar{x})^2}{2\sigma^2}} \quad (4.1)$$

$$s(x) = \frac{A_s}{1 + e^{-\lambda(x-x_i)}} \quad (4.2)$$

574 Finally, a possible representation of a skew distribution is given by formula 4.3 and is the product
 575 of 4.1 and 4.2. Naturally, here $A_{sk} = A_g \times A_s$ and represents the theoretical maximum in the limit
 576 where the skew tends to a gaussian function.

$$sk(x) = g(x) \times s(x) = A_{sk} \frac{e^{\frac{-(x-\bar{x})^2}{2\sigma^2}}}{1 + e^{-\lambda(x-x_i)}} \quad (4.3)$$

577 **4.3.3.3 Results**

578 **Influence of T_{scint} on the muon distribution**

579 **Influence of T_{RPC} on the muon distribution**

580 **Influence of the telescope inclination on the muon distribution**

581 **Comparison to data taken at GIF without irradiation**

582 **4.3.4 Photon flux at GIF**

583 **4.3.4.1 Expectations from simulations**

584 In order to understand and evaluate the γ flux in the GIF area, simulations had been conducted in
 585 1999 and published by S. Agosteo et al [35]. Table 4.1 presented in this article gives us the γ flux
 586 for different distances D to the source. This simulation was done using GEANT and a Monte Carlo
 587 N-Particle (MCNP) transport code, and the flux F is given in number of γ per unit area and unit time
 588 along with the estimated error from these packages expressed in %.

Nominal ABS	Photon flux F [$s^{-1}cm^{-2}$]			
	at $D = 50$ cm	at $D = 155$ cm	at $D = 300$ cm	at $D = 400$ cm
1	$0.12 \cdot 10^8 \pm 0.2\%$	$0.14 \cdot 10^7 \pm 0.5\%$	$0.45 \cdot 10^6 \pm 0.5\%$	$0.28 \cdot 10^6 \pm 0.5\%$
2	$0.68 \cdot 10^7 \pm 0.3\%$	$0.80 \cdot 10^6 \pm 0.8\%$	$0.25 \cdot 10^6 \pm 0.8\%$	$0.16 \cdot 10^6 \pm 0.6\%$
5	$0.31 \cdot 10^7 \pm 0.4\%$	$0.36 \cdot 10^6 \pm 1.2\%$	$0.11 \cdot 10^6 \pm 1.2\%$	$0.70 \cdot 10^5 \pm 0.9\%$

Table 4.1: Total photon flux ($E\gamma \leq 662$ keV) with statistical error predicted considering a ^{137}Cs activity of 740 GBq at different values of the distance D to the source along the x -axis of irradiation field [35].

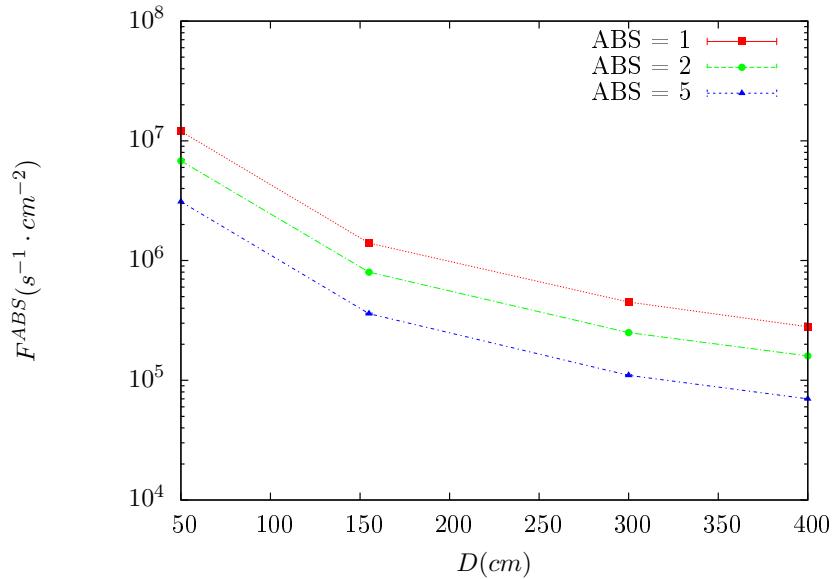


Figure 4.14: γ flux $F(D)$ is plot using values from table 4.1. As expected, the plot shows similar attenuation behaviours with increasing distance for each absorption factors.

The simulation doesn't directly provide us with an estimated flux at the level of our RPC. First of all, it is needed to extract the value of the flux from the available data contained in the original paper and then to estimate the flux in 2014 at the time the experimentation took place. Figure 4.14 that contains the data from Table 4.1. In the case of a pointlike source emitting isotropic and homogeneous gamma radiations, the gamma flux F at a distance D to the source with respect to a reference point situated at D_0 where a known flux F_0 is measured will be expressed like in Equation 4.4, assuming that the flux decreases as $1/D^2$, where c is a fitting factor.

$$F^{ABS} = F_0^{ABS} \times \left(\frac{cD_0}{D} \right)^2 \quad (4.4)$$

By rewriting Equation 4.4, it comes that :

$$c = \frac{D}{D_0} \sqrt{\frac{F^{ABS}}{F_0^{ABS}}} \quad (4.5)$$

$$\Delta c = \frac{c}{2} \left(\frac{\Delta F^{ABS}}{F^{ABS}} + \frac{\Delta F_0^{ABS}}{F_0^{ABS}} \right) \quad (4.6)$$

Finally, using Equation 4.5 and the data in Table 4.1 with $D_0 = 50$ cm as reference point, we can build Table 4.2. It is interesting to note that c for each value of D doesn't depend on the absorption factor.

Nominal ABS	Correction factor c		
	at $D = 155$ cm	at $D = 300$ cm	at $D = 400$ cm
1	$1.059 \pm 0.70\%$	$1.162 \pm 0.70\%$	$1.222 \pm 0.70\%$
2	$1.063 \pm 1.10\%$	$1.150 \pm 1.10\%$	$1.227 \pm 0.90\%$
5	$1.056 \pm 1.60\%$	$1.130 \pm 1.60\%$	$1.202 \pm 1.30\%$

Table 4.2: Correction factor c is computed thanks to formulae 4.5 taking as reference $D_0 = 50$ cm and the associated flux F_0^{ABS} for each absorption factor available in table 4.1.

600 For the range of D/D_0 values available, it is possible to use a simple linear fit to get the evolution
 601 of c . The linear fit will then use only 2 free parameters, a and b , as written in Equation 4.7. This gives
 602 us the results showed in Figure 4.15. Figure 4.15b confirms that using only a linear fit to extract c is
 603 enough as the evolution of the rate that can be obtained superimposes well on the simulation points.

$$c \left(\frac{D}{D_0} \right) = a \frac{D}{D_0} + b \quad (4.7)$$

$$F^{ABS} = F_0^{ABS} \left(a + \frac{bD_0}{D} \right)^2 \quad (4.8)$$

$$\Delta F^{ABS} = F^{ABS} \left[\frac{\Delta F_0^{ABS}}{F_0^{ABS}} + 2 \frac{\Delta a + \Delta b \frac{D_0}{D}}{a + \frac{bD_0}{D}} \right] \quad (4.9)$$

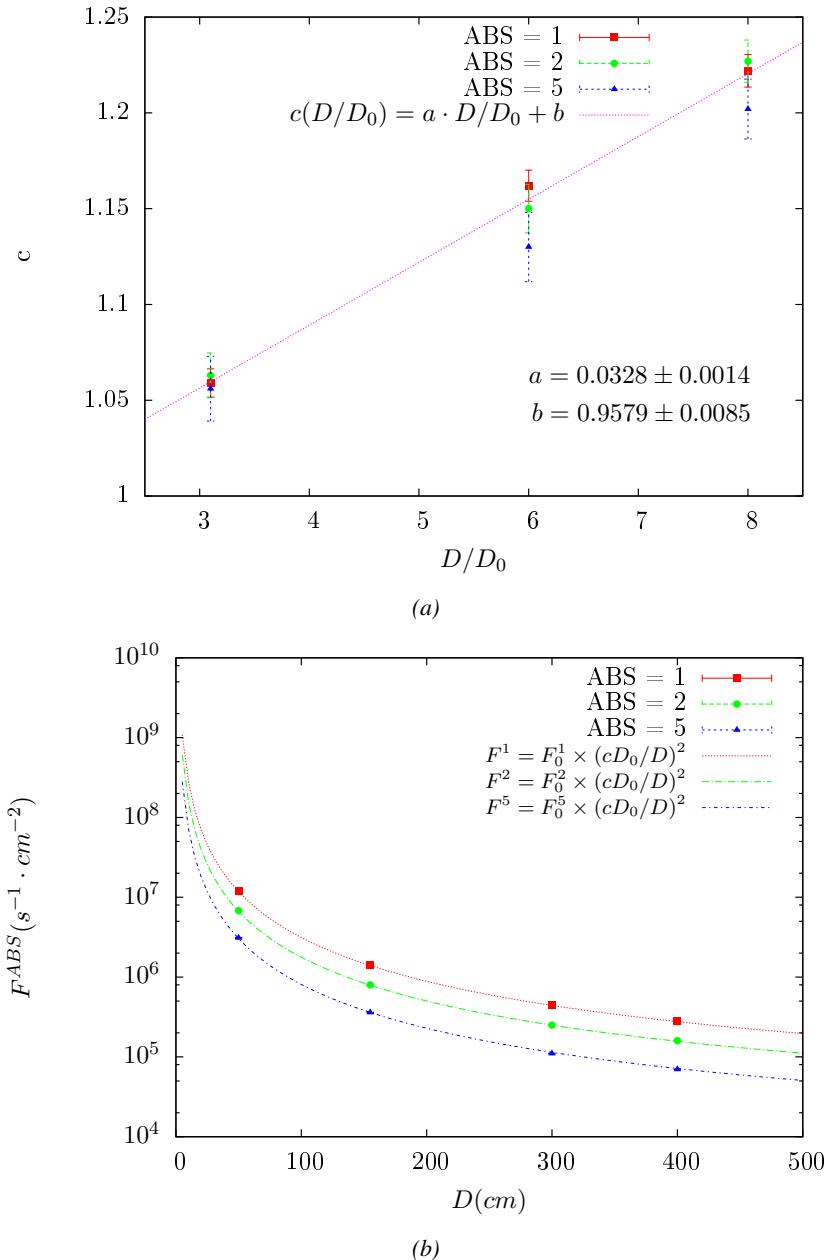


Figure 4.15: Figure 4.15a shows the linear approximation fit done via formulae 4.7 on data from table 4.2. Figure 4.15b shows a comparison of this model with the simulated flux using a and b given in figure 4.15a in formulae 4.4 and the reference value $D_0 = 50\text{cm}$ and the associated flux for each absorption factor F_0^{ABS} from table 4.1

In the case of the 2014 GIF tests, the RPC plane is located at a distance $D = 206$ cm to the source. Moreover, to estimate the strength of the flux in 2014, it is necessary to consider the nuclear decay through time associated to the Cesium source whose half-life is well known ($t_{1/2} = (30.05 \pm 0.08)$ y). The very first source activity measurement has been done on the 5th of March 1997 while the GIF

608 tests were done in between the 20th and the 31st of August 2014, i.e. at a time $t = (17.47 \pm 0.02)$ y
 609 resulting in an attenuation of the activity from 740 GBq in 1997 to 494 GBq in 2014. All the needed
 610 information to extrapolate the flux through our detector in 2014 has now been assembled, leading
 611 to the Table 4.3. It is interesting to note that for a common RPC sensitivity to γ of $2 \cdot 10^{-3}$, the
 612 order of magnitude of the estimated hit rate per unit area is of the order of the kHz for the fully
 613 opened source. Moreover, taking profit of the two working absorbers, it will be possible to scan
 614 background rates at 0 Hz, ~ 300 Hz as well as ~ 600 Hz. Without source, a good estimate of the
 615 intrinsic performance will be available. Then at 300 Hz, the goal will be to show that the detectors
 616 fulfill the performance certification of CMS RPCs. Then a first idea of the performance of the
 617 detectors at higher background will be provided with absorption factors 2 (~ 600 Hz) and 1 (no
 618 absorption). [Here I will also put a reference to the plot showing the estimated background rate at
 619 the level of RE3/1 in the case of HL-LHC but this one being in another chapter, I will do it later.]

Nominal ABS	Photon flux F [$s^{-1}cm^{-2}$]			Hit rate/unit area [$Hz cm^{-2}$] at $D^{2014} = 206$ cm
	at $D_0^{1997} = 50$ cm	at $D^{1997} = 206$ cm	at $D^{2014} = 206$ cm	
1	$0.12 \cdot 10^8 \pm 0.2\%$	$0.84 \cdot 10^6 \pm 0.3\%$	$0.56 \cdot 10^6 \pm 0.3\%$	1129 ± 32
2	$0.68 \cdot 10^7 \pm 0.3\%$	$0.48 \cdot 10^6 \pm 0.3\%$	$0.32 \cdot 10^6 \pm 0.3\%$	640 ± 19
5	$0.31 \cdot 10^7 \pm 0.4\%$	$0.22 \cdot 10^6 \pm 0.3\%$	$0.15 \cdot 10^6 \pm 0.3\%$	292 ± 9

Table 4.3: The data at D_0 in 1997 is taken from [35]. In a second step, using Equations 4.8 and 4.9, the flux at D can be estimated in 1997. Then, taking into account the attenuation of the source activity, the flux at D can be estimated at the time of the tests in GIF in 2014. Finally, assuming a sensitivity of the RPC to γ $s = 2 \cdot 10^{-3}$, an estimation of the hit rate per unit area is obtained.

620 **4.3.4.2 Dose measurements**

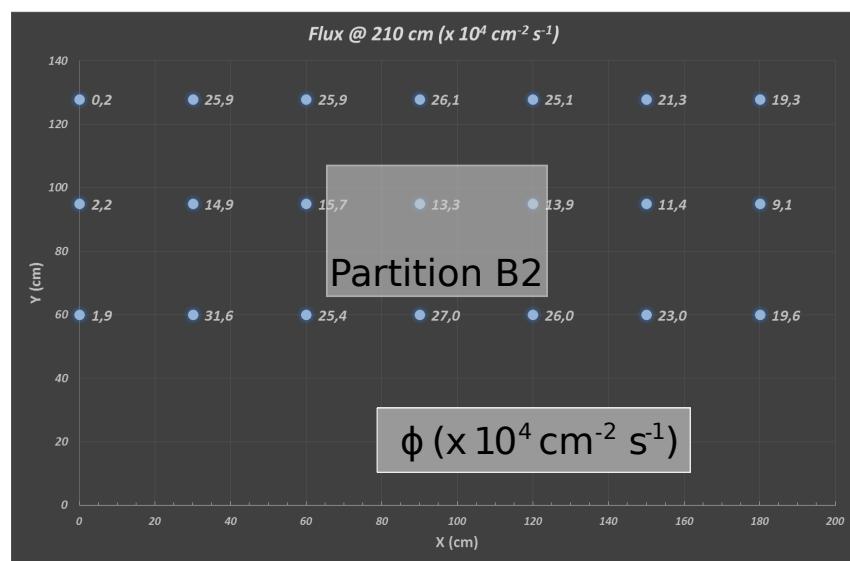


Figure 4.16: Dose measurements has been done in a plane corresponding to the tents front side. This plan is 1900 mm away from the source. As explained in the first chapter, a lens-shaped lead filter provides a uniform photon flux in the vertical plan orthogonal to the beam direction. If the second line of measured fluxes is not taken into account because of lower values due to experimental equipments in the way between the source and the tent, the uniformity of the flux is well showed by the results.

⁶²¹ **4.3.5 Results and discussions**

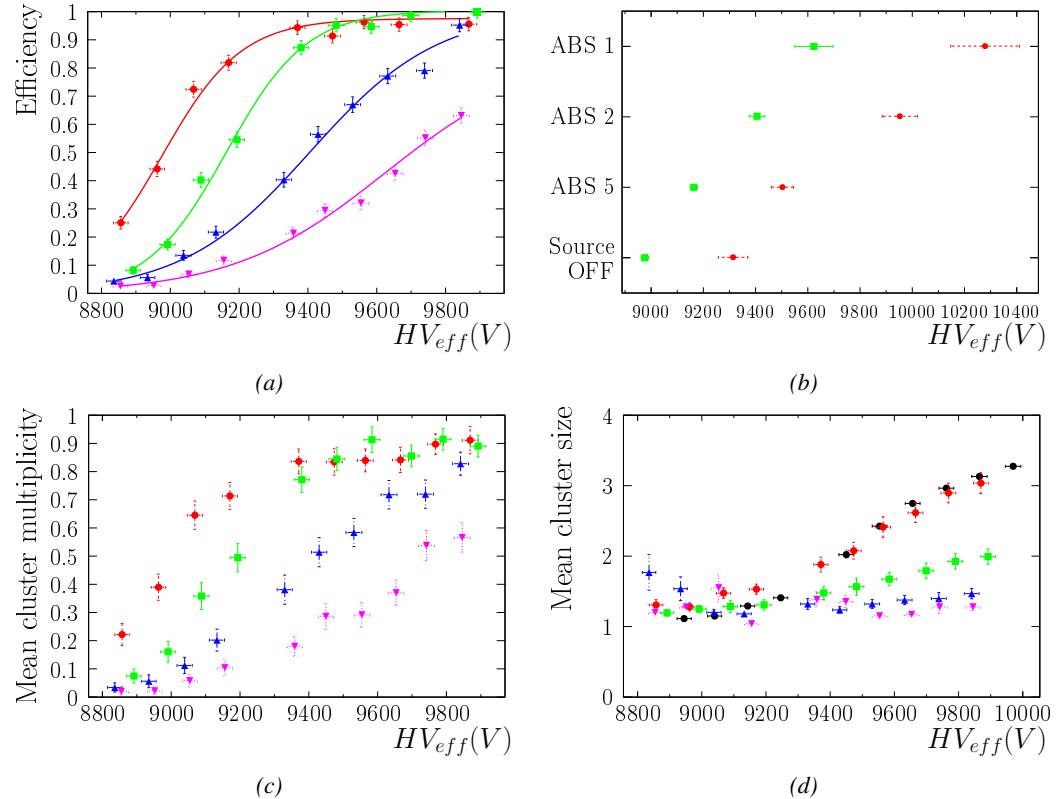


Figure 4.17

622 **4.4 Longevity tests at GIF++**

623 Longevity studies imply a monitoring of the performance of the detectors probed using a high inten-
624 sity muon beam in a irradiated environment by periodically measuring their rate capability, the dark
625 current running through them and the bulk resistivity of the Bakelite composing their electrodes.
626 GIF++, with its very intense ^{137}Cs source, provides the perfect environment to perform such kind
627 of tests. Assuming a maximum acceleration factor of 3, it is expected to accumulate the equivalent
628 charge in 1.7 years.

629 As the maximum background is found in the endcap, the choice naturally was made to focus the
630 GIF++ longevity studies on endcap chambers. Most of the RPC system was installed in 2007. Nev-
631 ertheless, the large chambers in the fourth endcap (RE4/2 and RE4/3) have been installed during
632 LS1 in 2014. The Bakelite of these two different productions having different properties, four spare
633 chambers of the present system were selected, two RE2,3/2 spares and two RE4/2 spares. Having
634 two chambers of each type allows to always keep one of them non irradiated as reference, the per-
635 formance evolution of the irradiated chamber being then compared through time to the performance
636 of the non irradiated one.

637 The performance of the detectors under different level of irradiation is measured periodically dur-
638 ing dedicated test beam periods using the H4 muon beam. In between these test beam periods, the
639 two RE2,3/2 and RE4/2 chambers selected for this study are irradiated by the ^{137}Cs source in order
640 to accumulate charge and the gamma background is monitored, as well as the currents. The two
641 remaining chambers are kept non-irradiated as reference detectors. Due to the limited gas flow in
642 GIF++, the RE4 chamber remained non-irradiated until end of November 2016 where a new mass
643 flow controller has been installed allowing for bigger volumes of gas to flow in the system.

644 Figures 4.18 and 4.19 give us for different test beam periods, and thus for increasing integrated
645 charge through time, a comparison of the maximum efficiency, obtained using a sigmoid-like func-
646 tion, and of the working point of both irradiated and non irradiated chambers [**SIGMOID2005**]. No
647 aging is yet to see from this data, the shifts in γ rate per unit area in between irradiated and non
648 irradiated detectors and RE2 and RE4 types being easily explained by a difference of sensitivity due
649 to the various Bakelite resistivities of the HPL electrodes used for the electrode production.

650 Collecting performance data at each test beam period allows us to extrapolate the maximum effi-
651 ciency for a background hit rate of $300\text{ Hz}/\text{cm}^2$ corresponding to the expected HL-LHC conditions.
652 Aging effects could emerge from a loss of efficiency with increasing integrated charge over time,
653 thus Figure 4.20 helps us understand such degradation of the performance of irradiated detectors in
654 comparison with non irradiated ones. The final answer for an eventual loss of efficiency is given in
655 Figure 4.21 by comparing for both irradiated and non irradiated detectors the efficiency sigmoids
656 before and after the longevity study. Moreover, to complete the performance information, the Bake-
657 lite resistivity is regularly measured thanks to Ag scans (Figure 4.22) and the noise rate is monitored
658 weekly during irradiation periods (Figure 4.23). At the end of 2016, no signs of aging were observed
659 and further investigation is needed to get closer to the final integrated charge requirements proposed
660 for the longevity study of the present CMS RPC sub-system.

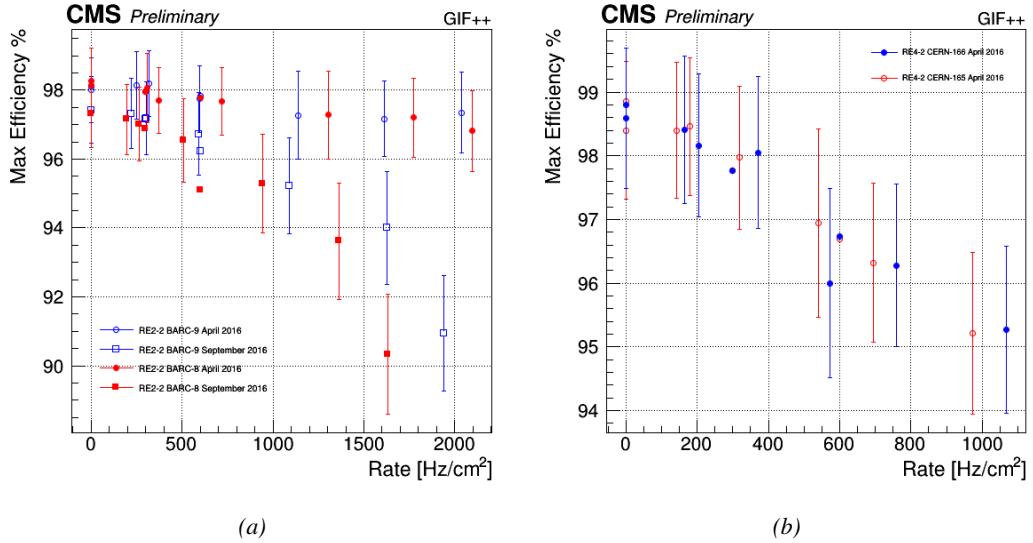


Figure 4.18: Evolution of the maximum efficiency for RE2 (4.18a) and RE4 (4.18b) chambers with increasing extrapolated γ rate per unit area at working point. Both irradiated (blue) and non irradiated (red) chambers are shown.

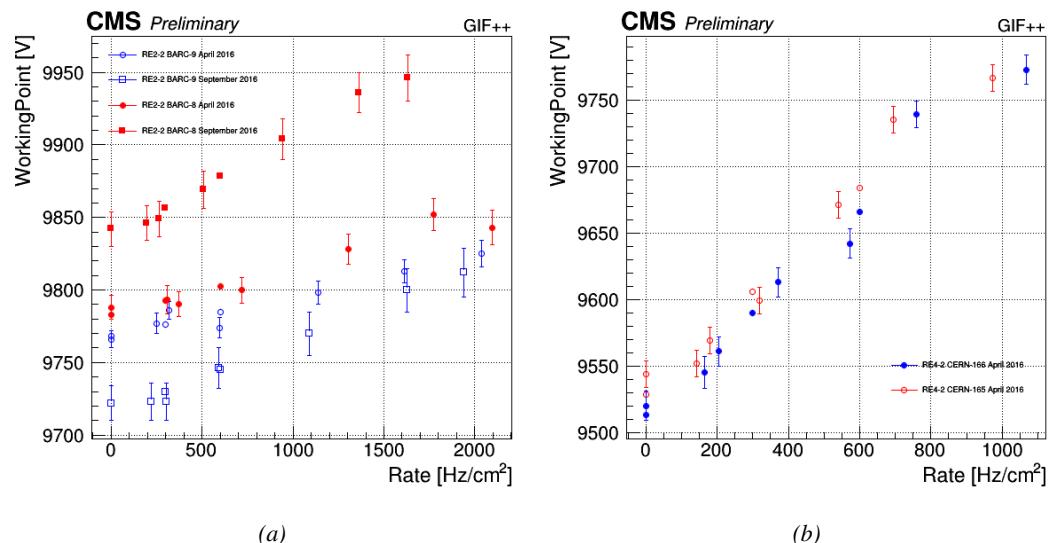


Figure 4.19: Evolution of the working point for RE2 (4.19a) and RE4 (4.19b) with increasing extrapolated γ rate per unit area at working point. Both irradiated (blue) and non irradiated (red) chambers are shown.

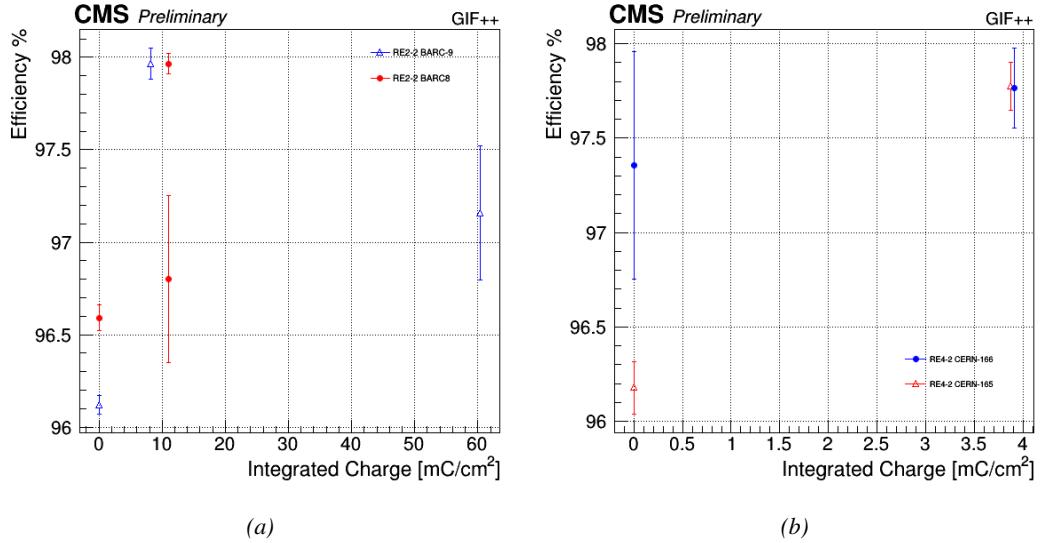


Figure 4.20: Evolution of the maximum efficiency at HL-LHC conditions, i.e. a background hit rate per unit area of 300 Hz/cm², with increasing integrated charge for RE2 (4.20a) and RE4 (4.20b) detectors. Both irradiated (blue) and non irradiated (red) chambers are shown. The integrated charge for non irradiated detectors is recorded during test beam periods and stays small with respect to the charge accumulated in irradiated chambers.

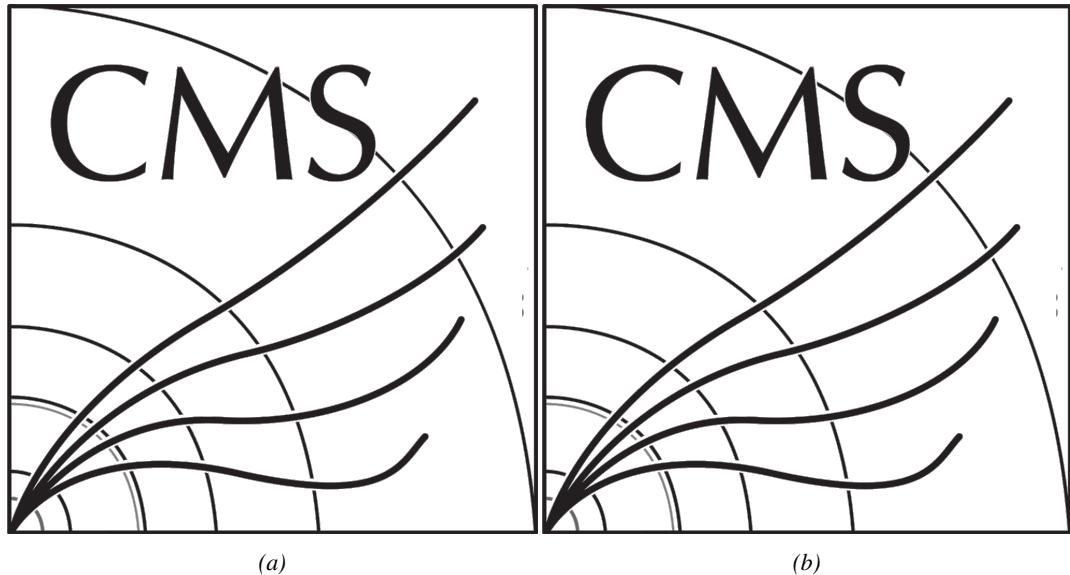


Figure 4.21: Comparison of the efficiency sigmoid before (triangles) and after (circles) irradiation for RE2 (4.21a) and RE4 (4.21b) detectors. Both irradiated (blue) and non irradiated (red) chambers are shown.

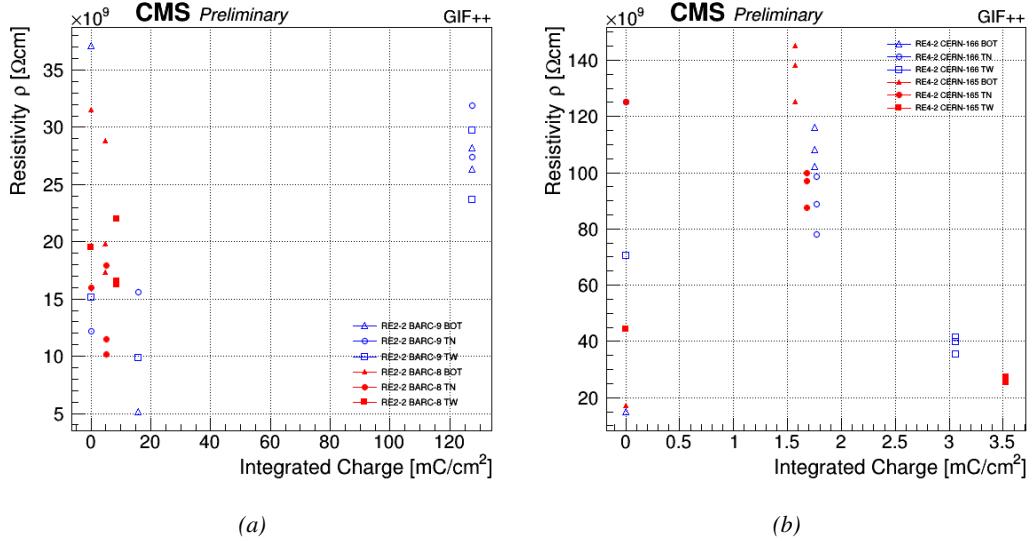


Figure 4.22: Evolution of the Bakelite resistivity for RE2 (4.22a) and RE4 (4.22b) detectors. Both irradiated (blue) and non-irradiated (red) chambers are shown.

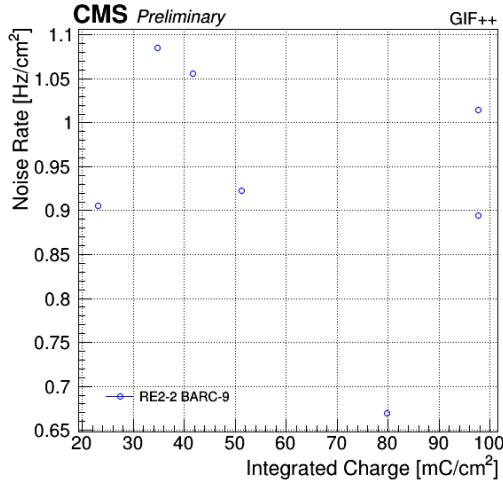


Figure 4.23: Evolution of the noise rate per unit area for the irradiated chamber RE2-2-BARC-9 only.

4.4.1 Description of the Data Acquisition

For the longevity studies, four spare chambers of the present system are used. Two spare RPCs of the RE2,3 stations as well as two spare RPCs from the new RE4 stations have been mounted in a Trolley. Six RE4 gaps are also placed in the trolley. The trolley is placed inside the GIFT++ in the upstream region of the bunker, taking the cesium source as a reference. The trolley is oriented for the detection surface of the chambers to be orthogonal to the beam line. The system can be moved along the orthogonal plane in order to have the beam in all η -partitions. For the aging the trolley is

669 moved outside the beam line and is placed in a distance of 5.2 m to the source, which irradiates the
 670 bunker using an attenuation filter of 2.2 which corresponds to a fluence of 10^7 gamma/cm^2 .

671 During GIF++ operation, the data collected can be divided into different categories as several
 672 parameters are monitored in addition to the usual RPC performance data. On one hand, to know
 673 the performance of a chamber, it is need to measure its efficiency and to know the background
 674 conditions in which it is operated. To do this, the hit signals from the chamber are recorded and
 675 stored in a ROOT file via a Data Acquisition (DAQ) software. On the other hand, it is also very
 676 important to monitor parameters such as environmental pressure and temperature, gas temperature
 677 and humidity, RPC HV, LV, and currents, or even source and beam status. This is done through the
 678 GIF++ web Detector Control Software (DCS) that stores this information in a database.

679 Two different types of tests are conducted on RPCs via the DAQ. Indeed, the performance of the
 680 detectors is measured periodically during dedicated test beam periods using the H4 muon beam. In
 681 between these test beam periods, when the beam is not available, the chambers are irradiated by the
 682 ^{137}Cs in order to accumulate deposited charge and the gamma background is measured.

683 RPCs under test are connected through LVDS cables to V1190A Time-to-Digital Converter
 684 (TDC) modules manufactured by CAEN. These modules, located in the rack area outside of the
 685 bunker, get the logic signals sent by the chambers and save them into their buffers. Due to the
 686 limited size of the buffers, the collected data is regularly erased and replaced. A trigger signal is
 687 needed for the TDC modules to send the useful data to the DAQ computer via a V1718 CAEN USB
 688 communication module.

689 In the case of performance test, the trigger signal used for data acquisition is generated by the
 690 coincidence of three scintillators. A first one is placed upstream outside of the bunker, a second one
 691 is placed downstream outside of the bunker, while a third one is placed in front of the trolley, close by
 692 the chambers. Every time a trigger is sent to the TDCs, i.e. every time a muon is detected, knowing
 693 the time delay in between the trigger and the RPC signals, signals located in the right time window
 694 are extracted from the buffers and saved for later analysis. Signals are taken in a time window of
 695 400 ns centered on the muon peak (here we could show a time spectrum). On the other hand, in the
 696 case of background rate measurement, the trigger signal needs to be "random" not to measure muons
 697 but to look at gamma background. A trigger pulse is continuously generated at a rate of 300 Hz using
 698 a dual timer. To integrate an as great as possible time, all signals contained within a time window of
 699 10us prior to the random trigger signal are extracted form the buffers and saved for further analysis
 700 (here another time spectrum to illustrate could be useful, maybe even place both spectrum together
 701 as a single Figure).

702 The signals sent to the TDCs correspond to hit collections in the RPCs. When a particle hits
 703 a RPC, it induce a signal in the pickup strips of the RPC readout. If this signal is higher than the
 704 detection threshold, a LVDS signal is sent to the TDCs. The data is then organised into 4 branches
 705 keeping track of the event number, the hit multiplicity for the whole setup, and the time and channel
 706 profile of the hits in the TDCs.

707 **4.4.2 RPC current, environmental and operation parameter monitoring**

708 In order to take into account the variation of pressure and temperature between different data taking
 709 periods the applied voltage is corrected following the relationship :

$$709 \quad HV_{eff} = HV_{app} \times \left(0.2 + 0.8 \cdot \frac{P_0}{P} \times \frac{T}{T_0} \right) \quad (4.10)$$

⁷¹⁰ where T_0 (=293 K) and P_0 (=990 mbar) are the reference values.

⁷¹¹ **4.4.3 Measurement procedure**

- ⁷¹² Insert a short description of the online tools (DAQ, DCS, DQM).
- ⁷¹³ Insert a short description of the offline tools : tracking and efficiency algorithm.
- ⁷¹⁴ Identify long term aging effects we are monitoring the rates per strip.

⁷¹⁵ **4.4.4 Longevity studies results**

5

716

717

Investigation on high rate RPCs

718 **5.1 Rate limitations and ageing of RPCs**

719 **5.1.1 Low resistivity electrodes**

720 **5.1.2 Low noise front-end electronics**

721 **5.2 Construction of prototypes**

722 **5.3 Results and discussions**

6

723

724

Conclusions and outlooks

725 **6.1 Conclusions**

726 **6.2 Outlooks**

A

727

728

A data acquisition software for CAEN VME TDCs

729

730 Certifying detectors in the perspective of HL-LHC required to develop tools for the GIF++ experiment.
731 Among them was the C++ Data Acquisition (DAQ) software that allows to make the communications
732 in between a computer and TDC modules in order to retrieve the RPC data [37]. In this
733 appendix, details about this software, as of how the software was written, how it functions and how
734 it can be exported to another similar setup, will be given.

735 A.1 GIF++ DAQ file tree

736 GIF++ DAQ source code is fully available on github at https://github.com/afagot/GIF_DAQ. The software requires 3 non-optional dependencies:

- 738 • CAEN USB Driver, to mount the VME hardware,
739 • CAEN VME Library, to communicate with the VME hardware, and
740 • ROOT, to organize the collected data into a TTree.

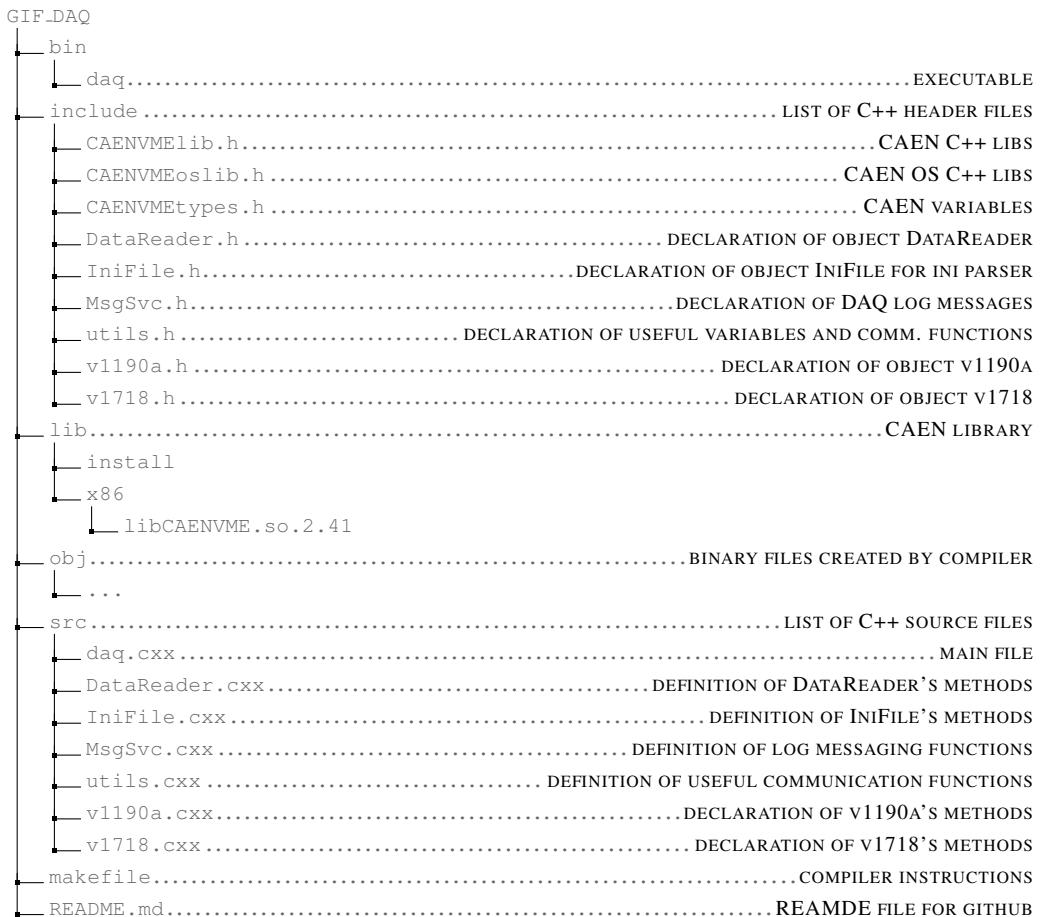
741 The CAEN VME library will not be packaged by distributions and will need to be installed man-
742 ually. To compile the GIF++ DAQ project via a terminal, from the DAQ folder use the command:

743

```
744     make
```

745 The source code tree is provided below along with comments to give an overview of the files' con-
746 tent. The different objects created for this project (`v1718`, `v1190a`, `IniFile` & `DataReader`) will be
747 described in details in the following sections.

748



749 A.2 Usage of the DAQ

750 GIF++ DAQ, as used in GIF++, is not a standalone software. Indeed, the system being more complexe,
 751 the DAQ only is a sub-layer of the software architecture developped to control and monitor
 752 the RPCs that are placed into the bunker for performance study in an irradiated environment. The top
 753 layer of GIF++ is a Web Detector Control System (webDCS) application. The DAQ is only called
 754 by the webDCS when data needs to be acquired. The webDCS operates the DAQ through command
 755 line. To start the DAQ, the webDCS calls:

756
 757 bin/daq /path/to/the/log/file/in/the/output/data/folder

758 where /path/to/the/log/file/in/the/output/data/folder is the only argument required. This
 759 log file is important for the webDCS as this file contains all the content of the communication of the
 760 webDCS and the different systems monitored by the webDCS. Its content is constantly displayed
 761 during data taking for the users to be able to follow the operations. The communication messages
 762 are normally sent to the webDCS log file via the functions declared in file MsgSvc.h, typically
 763 MSG_INFO(string message).

764

765 A.3 Description of the readout setup

766 The CMS RPC setup at GIF++ counts 5 V1190A Time-to-Digital Converter (TDC) manufactured
 767 by CAEN [38]. V1190A are VME units accepting 128 independent Multi-Hit/Multi-Event TDC
 768 channels whose signals are treated by 4 100 ps high performance TDC chips developed by CERN
 769 / ECP-MIC Division. The communication between the computer and the TDCs to transfer data is
 770 done via a V1718 VME master module also manufactured by CAEN and operated from a USB
 771 port [39]. These VME modules are all hosted into a 6U VME 6021 powered crate manufactured by
 772 W-Ie-Ne-R than can accommodate up to 21 VME bus cards [40]. These 3 components of the DAQ
 773 setup are shown in Figure A.1.

774

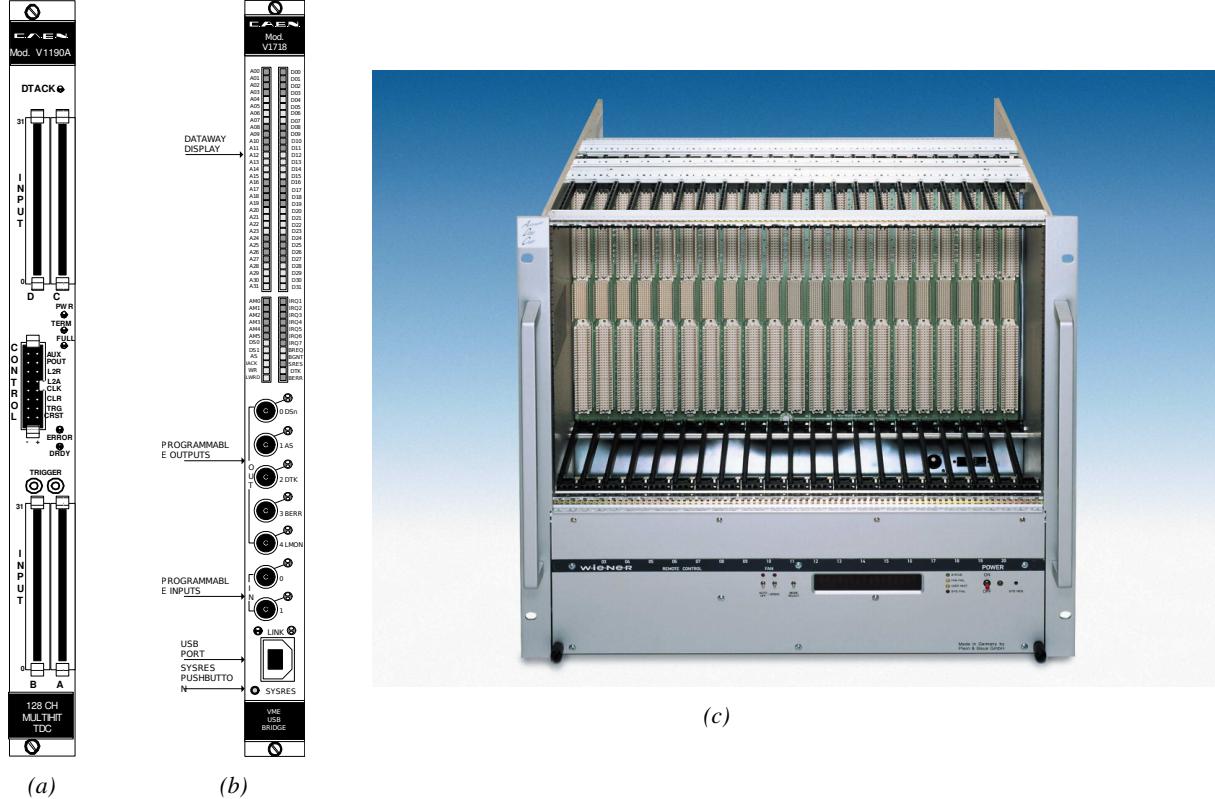


Figure A.1: (A.1a) View of the front panel of a V1190A TDC module [38]. (A.1b) View of the front panel of a V1718 Bridge module [39]. (A.1c) View of the front panel of a 6U 6021 VME crate [40].

775

A.4 Data read-out

776 To efficiently perform a data readout algorithm, C++ objects to handle the VME modules (TDCs
 777 and VME bridge) have been created along with objects to store data and read the configuration file

778 that comes as an input of the DAQ software.

779

780 A.4.1 V1190A TDCs

781 The DAQ used at GIF takes profit of the *Trigger Matching Mode* offered by V1190A modules.
 782 This setting is enabled through the method `v1190a::SetTrigMatching (int ntdcs)` where `ntdcs`
 783 is the total number of TDCs in the setup this setting needs to be enabled for (Source Code A.1). A
 784 trigger matching is performed in between a trigger time tag, a trigger signal sent into the TRIGGER
 785 input of the TDC visible on Figure A.1a, and the channel time measurements, signals recorded from
 786 the detectors under test in our case. Control over this data acquisition mode, explained through
 787 Figure A.2, is offered via 4 programmable parameters:

- 788 • **match window:** the matching between a trigger and a hit is done within a programmable time
 789 window. This is set via the method

```
790     void v1190a::SetTrigWindowWidth(UINT windowWidth, int ntdcs)
```

- 791 • **window offset:** temporal distance between the trigger tag and the start of the trigger matching
 792 window. This is set via the method

```
793     void v1190a::SetTrigWindowWidth(UINT windowWidth, int ntdcs)
```

- 794 • **extra search margin:** an extended time window is used to ensure that all matching hits are
 795 found. This is set via the method

```
796     void v1190a::SetTrigSearchMargin(UINT searchMargin, int ntdcs)
```

- 797 • **reject margin:** older hits are automatically rejected to prevent buffer overflows and to speed
 798 up the search time. This is set via the method

```
799     void v1190a::SetTrigRejectionMargin(UINT rejectMargin, int ntdcs)
```

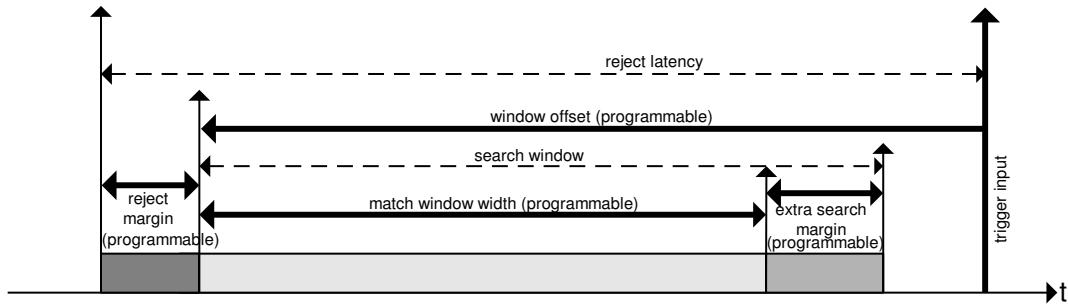


Figure A.2: Module V1190A Trigger Matching Mode timing diagram [38].

800 Each of these 4 parameters are given in number of clocks, 1 clock being 25 ns long. It is easy to
 801 understand at this level that there are 3 possible functioning settings:

- 802 • **1:** the match window is entirely contained after the trigger signal,

- 803 • **2:** the match window overlaps the trigger signal, or

- 804 • **3:** the match window is entirely contained before the trigger signal as displayed on Figure A.2.

805 In both the first and second cases, the sum of the window width and of the offset can be set to
806 a maximum of 40 clocks, which corresponds to 1 μ s. Evidently, the offset can be negative, allowing
807 for a longer match window, with the constraint of having the window ending at most 1 μ s after the
808 trigger signal. In the third case, the maximum negative offset allowed is of 2048 clocks (12 bit) cor-
809 responding to 51.2 μ s, the match window being strictly smaller than the offset. In the case of GIF++,
810 the choice has been made to use this last setting by delaying the trigger signal. During the studies
811 performed in GIF++, both the efficiency of the RPCs, probed using a muon beam, and the noise or
812 gamma background rate are monitored. The extra search and reject margins are left unused.
813 To probe the efficiency of RPC detectors, the trigger time tag is provided by the coïncidence of
814 scintillators when a bunch of muons passes through GIF++ area is used to trigger the data acquisi-
815 tion. For this measurement, it is useful to reduce the match window width only to contain the muon
816 information. Indeed, the delay in between a trigger signal and the detection of the corresponding
817 muon in the RPC being very contant (typically a few tens of ns due to jitter and cable length), the
818 muon signals are very localised in time. Thus, due to a delay of approximalety 325 ns in between
819 the muons and the trigger, the settings where chosen to have a window width of 24 clocks (600 ns)
820 centered on the muon peak thanks to a negative offset of 29 clocks (725 ns).
821 On the otherhand, monitoring the rates don't require for the DAQ to look at a specific time window.
822 It is important to integrate enough time to have a robust measurement of the rate as the number of
823 hits per time unit. The triggerring signal is provided by a pulse generator at a frequency of 300 Hz
824 to ensure that the data taking occurs in a random way, with respect to beam physics, to probe only
825 the irradiation spectrum on the detectors. The match window is set to 400 clocks (10 μ s) and the
826 negative offset to 401 clocks as it needs to exceed the value of the match window.

```

827
class v1190a
{
    private :
        long             Handle;
        vector<Data32>   Address;
        CVDataWidth      DataWidth;
        CVAddressModifier AddressModifier;

    public:

        v1190a(long handle, IniFile *inifile, int ntdcs);
        ~v1190a();
        Data16 write_op_reg(Data32 address, int code, string error);
        Data16 read_op_reg(Data32 address, string error);
        void Reset(int ntdcs);
        void Clear(int ntdcs);
        void TestWR(Data16 value,int ntdcs);
        void CheckTDCStatus(int ntdcs);
        void CheckCommunication(int ntdcs);
        void SetTDCTestMode(Data16 mode,int ntdcs);
        void SetTrigMatching(int ntdcs);
        void SetTrigTimeSubtraction(Data16 mode,int ntdcs);
        void SetTrigWindowWidth(Uint windowHeight,int ntdcs);
        void SetTrigWindowOffset(Uint windowOffset,int ntdcs);
        void SetTrigSearchMargin(Uint searchMargin,int ntdcs);
        void SetTrigRejectionMargin(Uint rejectMargin,int ntdcs);
        void GetTrigConfiguration(int ntdcs);
        void SetTrigConfiguration(IniFile *inifile,int ntdcs);
        void SetTDCDetectionMode(Data16 mode,int ntdcs);
        void SetTDCResolution(Data16 lsb,int ntdcs);
        void SetTDCDeadTime(Data16 time,int ntdcs);
        void SetTDCHeadTrailer(Data16 mode,int ntdcs);
        void SetTDCEventSize(Data16 size,int ntdcs);
        void SwitchChannels(IniFile *inifile,int ntdcs);
        void SetIRQ(Data32 level, Data32 count,int ntdcs);
        void SetBlockTransferMode(Data16 mode,int ntdcs);
        void Set(IniFile *inifile,int ntdcs);
        void CheckStatus(CVErrorCodes status) const;
        int ReadBlockD32(Uint tdc, const Data16 address,
                         Data32 *data, const Uint words, bool ignore_berr);
        Uint Read(RAWData *DataList,int ntdcs);
};

828

```

829 *Source Code A.1: Description of C++ object v1190a.*

830 The v1190a object, defined in the DAQ software as in Source Code A.1, offers the possibility to
 831 concatenate all TDCs in the readout setup into a single object containing a list of hardware addresses
 832 (addresses to access the TDCs' buffer through the VME crate) and each constructor and method acts
 833 on the list of TDCs.
 834

835 **A.4.2 DataReader**

836 Enabled thanks to v1190a::SetBlockTransferMode(Data16 mode, int ntdcs), the data transfer
 837 is done via Block Transfer (BLT). Using BLT allows to tranfer a fixed number of events called a
 838 *block*. This is used together with an Almost Full Level (AFL) of the TDCs' output buffers, defined

839 through `v1190a::SetIRQ(Data32 level, Data32 count, int ntdcs)`. This AFL gives the maxi-
 840 mum amount of 32735 words (16 bits, corresponding to the depth of a TDC output buffer) that can
 841 written in a buffer before an Interrupt Request (IRQ) is generated and seen by the VME Bridge,
 842 stopping the data acquisition to transfer the content of each TDC buffers before resuming. For each
 843 trigger, 6 words or more are written into the TDC buffer:

- 844 • a **global header** providing information of the event number since the beginning of the data
 acquisition,
- 846 • a **TDC header**,
- 847 • the **TDC data** (*if any*), 1 for each hit recorded during the event, providing the channel and the
 time stamp associated to the hit,
- 849 • a **TDC error** providing error flags,
- 850 • a **TDC trailer**,
- 851 • a **global trigger time tag** that provides the absolute trigger time relatively to the last reset,
 and
- 853 • a **global trailer** providing the total word count in the event.

854 As previously described in Section ??, CMS RPC FEEs provide us with 100 ns long LVDS out-
 855 put signals that are injected into the TDCs' input. Any avalanche signal that gives a signal above the
 856 FEEs threshold is thus recorded by the TDCs as a hit within the match window. Each hit is assigned
 857 to a specific TDC channel with a time stamp, with a precision of 100 ps. The reference time, $t_0 = 0$,
 858 is provided by the beginning of the match window. Thus for each trigger, coming from a scintillator
 859 coïncidence or the pulse generator, a list of hits is stored into the TDCs' buffers and will then be
 860 transferred into a ROOT Tree.

861 When the BLT is used, it is easy to understand that the maximum number of words that have
 862 been set as ALF will not be a finite number of events or, at least, the number of events that would
 863 be recorded into the TDC buffers will not be a multiple of the block size. In the last BLT cycle to
 864 tranfer data, the number of events to transfer will most propably be lower than the block size. In that
 865 case, the TDC can add fillers at the end of the block but this option requires to send more data to the
 866 computer and is thus a little slower. Another solution is to finish the transfer after the last event by
 867 sending a bus error that states that the BLT reached the last event in the pile. This method has been
 868 chosen in GIF++.

870 Due to irradiation, an event in GIF++ can count up to 300 words per TDC. A limit of 4096 words
 871 (12 bits) has been set to generate IRQ which represent from 14 to almost 700 events depending on
 872 the average of hits collected per event. Then the block size has been set to 100 events with enabled
 873 bus errors. When an AFL is reached for one of the TDCs, the VME bridge stops the acquisition by
 875 sending a BUSY signal.

877 The data is then transferred one TDC at a time into a structure called `RAWData` (Source Code A.2).

```
878
879   struct RAWData{
880     vector<int>           *EventList;
881     vector<int>           *NHitsList;
882     vector<int>           *QFlagList;
883     vector<vector<int>>  *Channellist;
884     vector<vector<float>> *TimeStampList;
885   };
```

880 *Source Code A.2: Description of data holding C++ structure `RAWData`.*

881 In order to organize the data transfer and the data storage, an object called `DataReader` was
882 created (Source Code A.3). On one hand, it has `v1718` and `v1190a` objects as private members for
883 communication purposes, such as VME modules settings via the configuration file `*iniFile` or data
884 read-out through `v1190a::Read()` and on the other hand, it contains the structure `RAWData` that allows
885 to organise the data in vectors reproducing the tree structure of a ROOT file.

```
886
887   class DataReader
888   {
889     private:
890       bool      StopFlag;
891       IniFile *iniFile;
892       Data32  MaxTriggers;
893       v1718   *VME;
894       int      nTDCs;
895       v1190a  *TDCs;
896       RAWData TDCData;
897
898     public:
899       DataReader();
900       virtual ~DataReader();
901       void      SetIniFile(string inifilename);
902       void      SetMaxTriggers();
903       Data32  GetMaxTriggers();
904       void      SetVME();
905       void      SetTDC();
906       int      GetQFlag(Uint it);
907       void      Init(string inifilename);
908       void      FlushBuffer();
909       void      Update();
910       string  GetFileName();
911       void      WriteRunRegistry(string filename);
912       void      Run();
913   };
```

888 *Source Code A.3: Description of C++ object `DataReader`.*

889 Each event is transferred from `TDCData` and saved into branches of a ROOT `TTree` as 3 integers
890 that represent the event ID (`EventCount`), the number of hits read from the TDCs (`nHits`), and the
891 quality flag that provides information for any problem in the data transfer (`qflag`), and 2 lists of
892 `nHits` elements containing the fired TDC channels (`TDCCh`) and their respective time stamps (`TDCTS`),
893 as presented in Source Code A.4. The ROOT file file is named using information contained into
894 the configuration file, presented in section A.5.2. The needed information is extracted using method
895 `DataReader::GetFileName()` and allow to build the output filename format `ScanXXXXXX_HVX_DAQ.root`

896 where ScanXXXXXX is a 6 digit number representing the scan number into GIFT++ database and HVX
 897 the HV step within the scan that can be more than a single digit. An example of ROOT data file is
 898 provided with Figure A.3.

```
899
  RAWData TDCData;
  TFile *outputFile = new TFile(outputFileName.c_str(),"recreate");
  TTree *RAWDataTree = new TTree("RAWData","RAWData");

  int          EventCount = -9;
  int          nHits    = -8;
  int          qflag    = -7;
  vector<int>   TDCCh;
  vector<float> TDCTS;

  RAWDataTree->Branch("EventNumber",&EventCount, "EventNumber/I");
  RAWDataTree->Branch("number_of_hits",&nHits, "number_of_hits/I");
  RAWDataTree->Branch("Quality_flag",&qflag, "Quality_flag/I");
  RAWDataTree->Branch("TDC_channel",&TDCCh);
  RAWDataTree->Branch("TDC_TimeStamp",&TDCTS);

900
  //...
  //Here read the TDC data using v1190a::Read() and place it into
  //TDCData for as long as you didn't collect the requested amount
  //of data.
  //...

  for(Uint i=0; i<TDCData.EventList->size(); i++){
    EventCount = TDCData.EventList->at(i);
    nHits     = TDCData.NHitsList->at(i);
    qflag     = TDCData.QFlagList->at(i);
    TDCCh    = TDCData.ChannelList->at(i);
    TDCTS    = TDCData.TimeStampList->at(i);
    RAWDataTree->Fill();
  }
}
```

901 *Source Code A.4: Highlight of the data transfer and organisation within DataReader::Run() after the data has been collected into TDCData.*

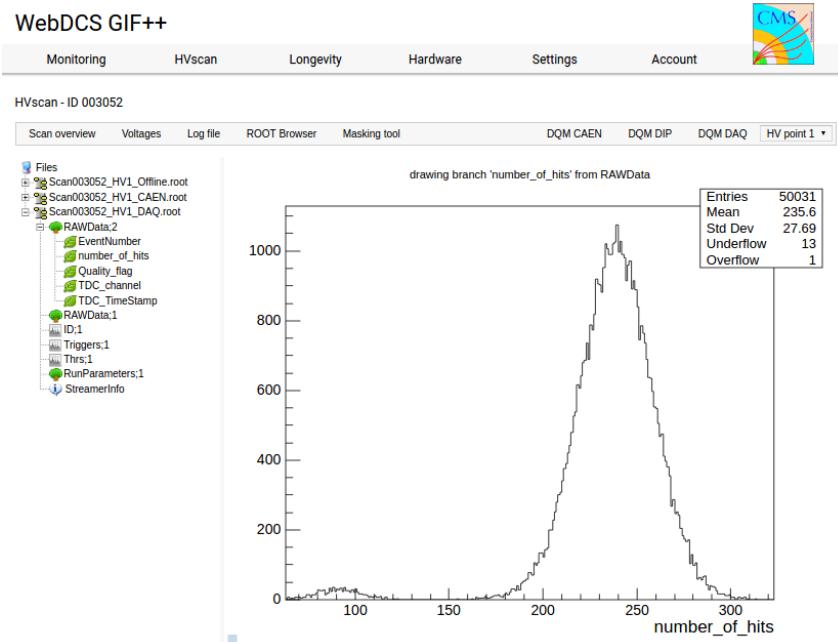


Figure A.3: Structure of the ROOT output file generated by the DAQ. The 5 branches (EventNumber, number_of_hits, Quality_flag, TDC_channel and TDC_TimeStamp) are visible on the left panel of the ROOT browser. On the right panel is visible the histogram corresponding to the variable nHits. In this specific example, there were approximately 50k events recorded to measure the gamma irradiation rate on the detectors. Each event is stored as a single entry in the TTree.

902 A.4.3 Data quality flag

903 Among the parameters that are recorded for each event, the quality flag, defined in Source Code A.5,
 904 is determined on the fly by checking the data recorded by every single TDC. From method `v1190a::Read()`,
 905 it can be understood that the content of each TDC buffer is readout one TDC at a time. Entries are
 906 created in the data list for the first TDC and then, when the second buffer is readout, events corre-
 907 sponding to entries that have already been created to store data for the previous TDC are added to
 908 the existing list element. On the contrary, when an event entry has not been yet created in the data
 909 list, a new entry is created.

```
910
 911 typedef enum _QualityFlag {
 912     GOOD      = 1,
 913     CORRUPTED = 0
 914 } QualityFlag;
```

912 *Source Code A.5: Definition of the quality flag `enum`.*

913 It is possible that each TDC buffer contains a different number of events. In cases where the first
 914 element in the buffer list is an event for corresponds to a new entry, the difference in between the
 915 entry from the buffer and the last entry in the data list is recorded and checked. If it is greater than 1,
 916 what should never be the case, the quality flag is set to CORRUPTED for this TDC and an empty entry
 917 is created in the place of the missing ones. Missing entries are believe to be the result of a bad hold

918 on the TDC buffers at the moment of the readout. Indeed, the software hold is effective only on 1
 919 TDC at a time and no solution as been found yet to completely block the writting in the buffers when
 920 an IRQ is received.

921 At the end of each BLT cycle, the ID of the last entry stored for each TDC buffer is not recorded.
 922 When starting the next cycle, if the first entry in the pile corresponds to an event already existing
 923 in the list, the readout will start from this list element and will not be able to check the difference
 924 in between this entry's ID and the one of the last entry that was recorded for this TDC buffer in
 925 the previous cycle. In the case events were missing, the flag stays at its initial value of 0, which is
 926 similar to CORRUPTED and it is assumed that then this TDC will not contribute to `number_of_hits`,
 927 `TDC_channel` or `TDC_TimeStamp`.

928 Finally, since there will be 1 `RAWData` entry per TDC for each event (meaning `nTDCs` entries,
 929 referring to `DataReader` private attribute), the individual flags of each TDC will be added together.
 930 The final format is an integer composed `nTDCs` digits where each digit is the flag of a specific TDC.
 931 This is constructed using powers of 10 like follows:

```
932     TDC 0: QFlag = 100 × _QualityFlag
 933     TDC 1: QFlag = 101 × _QualityFlag
 934     ...
 935     TDC N: QFlag = 10N × _QualityFlag
```

936 and the final flag to be with N digits:

```
937     QFlag = n....3210
```

938 each digit being 1 or 0. Below is given an example with a 4 TDCs setup.

939 If all TDCs were good : `QFlag = 1111`,

940 but if TDC 2 was corrupted : `QFlag = 1011`.

941 When data taking is over and the data contained in the dynamical `RAWData` structure is transferred
 942 to the ROOT file, all the 0s are changed into 2s by calling the method `DataReader::GetQFlag()`.
 943 This will help translating the flag without knowing the number of TDCs beforehand. Indeed, a flag
 944 111 could be due to a 3 TDC setup with 3 good individual TDC flags or to a more than 3 TDC setup
 945 with TDCs those ID is greater than 2 being CORRUPTED, thus giving a 0.

946 The quality flag has been introduced quite late, in October 2017 only, to the list of GIFT++ DAQ
 947 parameters to be recorded into the output ROOT file. Before this addition, the missing data, corrupting
 948 the quality for the offline analysis, was contributing to artificially fill data with lower multiplicity.
 949 Looking at `TBranch number_of_hits` provides an information about the data of the full GIFT++
 950 setup. When a TDC is not able to transfer data for a specific event, the effect is a reduction of the
 951 total number of hits recorded in the full setup, this is what can be seen from Figure A.4. After offline
 952 reconstruction detector by detector, the effect of missing events can be seen in the artificially filled
 953 bin at multiplicity 0 shown in Figure A.5. Nonetheless, for data with high irradiation levels, as it is
 954 the case for Figure A.5a, discarding the fake multiplicity 0 data can be done easily during the offline
 955 analysis. At lower radiation, the missing events contribution becomes more problematic as the multi-
 956 tiplicity distribution overlaps the multiplicity 0 and that in the same time the proportion of missing

events decreases. Attempts to fit the distribution with a Poisson or skew distribution function were not conclusive and this very problem has been at the origin of the quality flag that allows to give a non ambiguous information about each event quality.

960

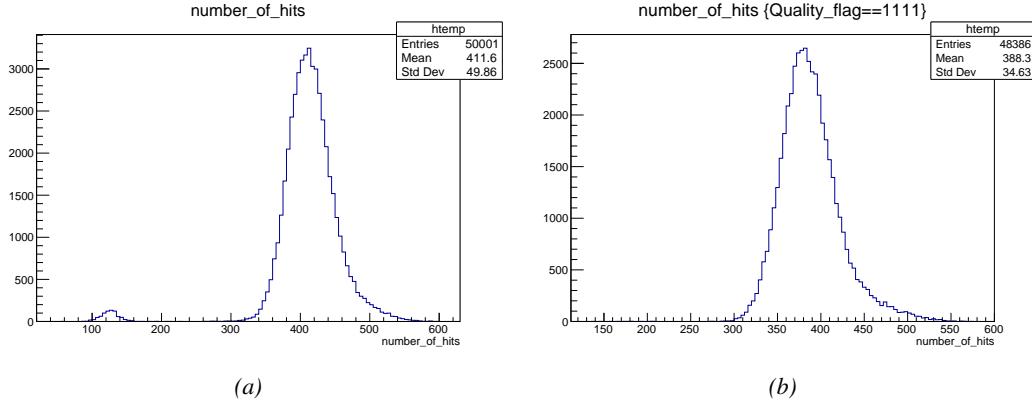


Figure A.4: The effect of the quality flag is explained by presenting the content of TBranch `number_of_hits` of a data file without `Quality_flag` in Figure A.4a and the content of the same TBranch for data corresponding to a `Quality_flag` where all TDCs were labelled as `GOOD` in Figure A.4b taken with similar conditions. It can be noted that the number of entries in Figure A.4b is slightly lower than in Figure A.4a due to the excluded events.

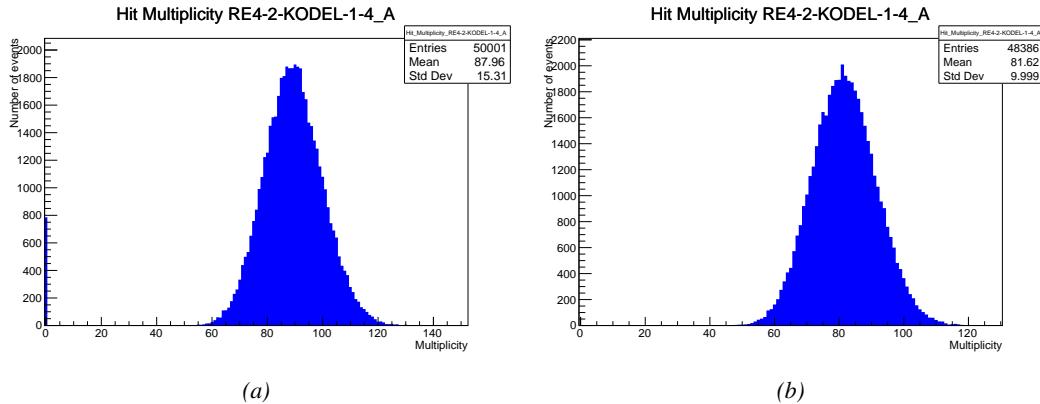


Figure A.5: Using the same data as previously showed in Figure A.4, the effect of the quality flag is explained by presenting the reconstructed hit multiplicity of a data file without `Quality_flag` in Figure A.5a and the reconstructed content of the same RPC partition for data corresponding to a `Quality_flag` where all TDCs were labelled as `GOOD` in Figure A.5b taken with similar conditions. The artificial high content of bin 0 is completely suppressed.

A.5 Communications

961 To ensure data readout and dialog in between the machine and the TDCs or in between the webDCS
962 and the DAQ, different communication solutions were used. First of all, it is important to have a
963

964 module to allow the communication in between the TDCs and the computer from which the DAQ
 965 operates. When this communication is effective, shifters using the webDCS to control data taking
 966 can thus send instructions to the DAQ.

967

968 A.5.1 V1718 USB Bridge

969 In the previous section, the data transfer has been discussed. The importance of the `v1718` object
 970 (Source Code A.6), used as private member of `DataReader`, was not explicated. VME master
 971 modules are used for communication purposes as they host the USB port that connects the pow-
 972 ered crate buffer to the computer where the DAQ is installed. From the source code point of view,
 973 this object is used to control the communication status, by reading the returned error codes with
 974 `v1718::CheckStatus()`, or to check for IRQs coming from the TDCs through `v1718::CheckIRQ()`.
 975 Finally, to ensure that triggers are blocked at the hardware level, a NIM pulse is sent out of one of the
 976 5 programmable outputs (`v1718::SendBUSY()`) to the VETO of the coincidence module where the
 977 trigger signals originate from. As long as this signal is ON, no trigger can reach the TDCs anymore.
 978

```
979
  class v1718{
    private:
      int Handle;
      Data32 Data;           // Data
      CVIRQLevels Level;    // Interrupt level
      CVAddressModifier AM;  // Addressing Mode
      CVDataWidth dataSize; // Data Format
      Data32 BaseAddress;   // Base Address

    public:
      v1718(IniFile *inifile);
      ~v1718();
      long GetHandle(void) const;
      int SetData(Data16 data);
      Data16 GetData(void);
      int SetLevel(CVIRQLevels level);
      CVIRQLevels GetLevel(void);
      int SetAM(CVAddressModifier am);
      CVAddressModifier GetAM(void);
      int SetDatasize(CVDataWidth datasize);
      CVDataWidth GetDataSize(void);
      int SetBaseAddress(Data16 baseaddress);
      Data16 GetBaseAddress(void);
      void CheckStatus(CVErrorCodes status) const;
      void CheckIRQ();
      void SetPulsers();
      void SendBUSY(BusyLevel level);
  };

```

980 *Source Code A.6: Description of C++ object v1718.*

981 A.5.2 Configuration file

982 The DAQ software takes as input a configuration file written using INI standard [41]. This file is
 983 partly filled with the information provided by the shifters when starting data acquisition using the
 984 webDCS, as shown by Figure A.6. This information is written in section [`General`] and will later

985 be stored in the ROOT file that contains the DAQ data as can be seen from Figure A.3. Indeed,
 986 another `TTree` called `RunParameters` as well as the 2 histograms `ID`, containing the scan number,
 987 start and stop time stamps, and `Triggers`, containing the number of triggers requested by the shifter,
 988 are available in the data files. Moreover, `ScanID` and `HV` are then used to construct the file name
 989 thanks to the method `DataReader::GetFileName()`.

Chamber	RE2-2-NPD-BARC-8	RE2-2-CERN-106	RE2-3-NPD-BARC-9	RE4-2-CERN-165	RE4-2-KODEL-1-4	Max triggers
HV _{eff} 1	8600	8500	8600	8500	6500	
HV _{eff} 2	8700	8600	8700	8600	6600	
HV _{eff} 3	8800	8700	8800	8700	6700	
HV _{eff} 4	8900	8800	8900	8800	6800	
HV _{eff} 5	9000	8900	9000	8900	6900	
HV _{eff} 6	9100	9000	9100	9000	7000	
HV _{eff} 7	9200	9100	9200	9100	7100	
HV _{eff} 8	9300	9200	9300	9200	7200	
HV _{eff} 9	9400	9300	9400	9300	7300	
HV _{eff} 10	9500	9400	9500	9400	7400	

Figure A.6: WebDCS DAQ scan page. On this page, shifters need to choose the type of scan (Rate, Efficiency or Noise Reference scan), the gamma source configuration at the moment of data taking, the beam configuration, and the trigger mode. These information will be stored in the DAQ ROOT output. Are also given the minimal measurement time and waiting time after ramping up of the detectors is over before starting the data acquisition. Then, the list of HV points to scan and the number of triggers for each run of the scan are given in the table underneath.

990 The rest of the information is written beforehand in the configuration file template, as explicated
 991 in Source Code A.7, and contains the hardware addresses to the different VME modules in the
 992 setup as well as settings for the TDCs. As the TDC settings available in the configuration file are not
 993 supposed to be modified, an improvement would be to remove them from the configuration file and
 994 to hardcode them inside of the DAQ code itself or to place them into a different INI file that would
 995 host only the TDC settings to lower the probability for a bad manipulation of the configuration file
 996 that can be modified from one of webDCS' menus.

997

```
[General]
TdcS=4
ScanID=$scanid
HV=$HV
RunType=$runtype
MaxTriggers=$maxtriggers
Beam=$beam
[VMEInterface]
Type=V1718
BaseAddress=0xFF0000
Name=VmeInterface
[TDC0]
Type=V1190A
BaseAddress=0x00000000
Name=Tdc0
StatusA00-15=1
StatusA16-31=1
StatusB00-15=1
StatusB16-31=1
StatusC00-15=1
StatusC16-31=1
StatusD00-15=1
StatusD16-31=1
[TDC1]
Type=V1190A
BaseAddress=0x11110000
Name=Tdc1
StatusA00-15=1
StatusA16-31=1
StatusB00-15=1
StatusB16-31=1
StatusC00-15=1
StatusC16-31=1
StatusD00-15=1
StatusD16-31=1
[TDC2]
Type=V1190A
BaseAddress=0x22220000
Name=Tdc2
StatusA00-15=1
StatusA16-31=1
StatusB00-15=1
StatusB16-31=1
StatusC00-15=1
StatusC16-31=1
StatusD00-15=1
StatusD16-31=1
[TDC3]
Type=V1190A
BaseAddress=0x44440000
Name=Tdc3
StatusA00-15=1
StatusA16-31=1
StatusB00-15=1
StatusB16-31=1
StatusC00-15=1
StatusC16-31=1
StatusD00-15=1
StatusD16-31=1
[TDCSettings]
TriggerExtraSearchMargin=0
TriggerRejectMargin=0
TriggerTimeSubtraction=0b1
TdcDetectionMode=0b01
TdcResolution=0b10
TdcDeadTime=0b00
TdcHeadTrailer=0b1
TdcEventSize=0b1001
TdcTestMode=0b0
BLTMode=1
```

998

Source Code A.7: INI configuration file template for 4 TDCs. In section [General], the number of TDCs is explicitated and information about the ongoing run is given. Then, there are sections for each and every VME modules. There buffer addresses are given and for the TDCs, the list of channels to enable is given. Finally, in section [TDCSettings], a part of the TDC settings are given.

1000 In order to retrieve the information of the configuration file, the object `IniFile` has been developed
 1001 to provide an INI parser, presented in Source Code A.8. It contains private methods returning a
 1002 boolean to check the type of line written in the file, whether a comment, a group header or a key line
 1003 (`IniFile::CheckIfComment()`, `IniFile::CheckIfGroup()` and `IniFile::CheckIfToken()`). The
 1004 key may sometimes be referred to as *token* in the source code. Moreover, the private element
 1005 `FileData` is a map of `const string` to `string` that allows to store the data contained inside the
 1006 configuration file via the public method `IniFile::GetFileData()` following the formatting (see
 1007 method `IniFile::Read()`):

```
1008
  1008     string group, token, value;
  // Get the field values for the 3 strings.
  // Then concatenate group and token together as a single string
  // with a dot separation.
  token = group + "." + token;
  FileData[token] = value;
```

1010 More methods have been written to translate the different keys into the right variable format
 1011 when used by the DAQ. For example, to get a `float` value out of the configuration file data, knowing
 1012 the group and the key needed, the method `IniFile::floatType()` can be used. It takes 3 arguments
 1013 being the group name and key name (both `string`), and a default `float` value used as exception in
 1014 the case the expected combination of group and key cannot be found in the configuration file. This
 1015 default value is then used and the DAQ continues on working after sending an alert in the log file for
 1016 further debugging.

```

1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
typedef map< const string, string > IniFileData;

class IniFile{
    private:
        bool      CheckIfComment(string line);
        bool      CheckIfGroup(string line, string& group);
        bool      CheckIfToken(string line, string& key, string& value);
        string    FileName;
        IniFileData FileData;
        int       Error;

    public:
        IniFile();
        IniFile(string filename);
        virtual ~IniFile();

        // Basic file operations
        void      SetFileName(string filename);
        int       Read();
        int       Write();
        IniFileData GetFileData();

        // Data readout methods
        Data32    addressType (string groupname, string keyname, Data32
        → defaultvalue);
        long      intType     (string groupname, string keyname, long
        → defaultvalue);
        long long  longType   (string groupname, string keyname, long long
        → defaultvalue );
        string    stringType  (string groupname, string keyname, string
        → defaultvalue );
        float     floatType   (string groupname, string keyname, float
        → defaultvalue );

        // Error methods
        string    GetErrorMsg();
};

Source Code A.8: Description of C++ object IniFile used as a parser for INI file format.
```

A.5.3 WebDCS/DAQ intercommunication

When shifters send instructions to the DAQ via the configuration file, it is the webDCS itself that gives the start command to the DAQ and then the 2 softwares use inter-process communication through file to synchronise themselves. This communication file is represented by the variable `const string __runstatuspath`.

On one side, the webDCS sends commands or status that are readout by the DAQ:

- INIT, status sent when launching a scan and read via function `CtrlRunStatus(...)`,
- START, command to start data taking and read via function `CheckSTART()`,
- STOP, command to stop data taking at the end of the scan and read via function `CheckSTOP()`, and
- KILL, command to kill data taking sent by user and read via function `CheckKILL()`

1031 and on the other, the DAQ sends status that are controled by the webDCS:

- 1032 ● `DAQ_RDY`, sent with `SendDAQReady()` to signify that the DAQ is ready to receive commands
1033 from the webDCS,
- 1034 ● `RUNNING`, sent with `SendDAQRunning()` to signify that the DAQ is taking data,
- 1035 ● `DAQ_ERR`, sent with `SendDAQError()` to signify that the DAQ didn't receive the expected com-
1036 mand from the webDCS or that the launch command didn't have the right number of argu-
1037 ments,
- 1038 ● `RD_ERR`, sent when the DAQ wasn't able to read the communication file, and
- 1039 ● `WR_ERR`, sent when the DAQ wasn't able to write into the communication file.

1040 **A.5.4 Example of inter-process communication cycle**

1041 Under normal conditions, the webDCS and the DAQ processes exchange commands and status via
1042 the file hosted at the address `__runstatuspath`, as explained in subsection A.5.3. An example of
1043 cycle is given in Table A.1. In this example, the steps 3 to 5 are repeated as long as the webDCS tells
1044 the DAQ to take data. A data taking cycle is the equivalent as what is called a *Scan* in GIFT++ jargon,
1045 referring to a set a runs with several HV steps. Each repetition of steps 3 to 5 is then equivalent to a
1046 single *Run*.

1047

1048 At any moment during the data taking, for any reason, the shifter can decide that the data taking
1049 needs to be stopped before it reached the end of the scheduled cycle. Thus at any moment on the
1050 cycle, the content of the inter-process communication file will be changed to `KILL` and the DAQ will
1051 shut down right away. The DAQ checks for `KILL` signals every 5s after the TDCs configuration is
1052 over. So far, the function `CheckKILL()` has been used only inside of the data taking loop of method
1053 `DataReader::Run()` and thus, if the shifter decides to KILL the data taking during the TDC con-
1054 figuration phase or the HV ramping in between 2 HV steps, the DAQ will not be stopped smoothly
1055 and a *force kill* command will be sent to stop the DAQ process that is still awake on the computer.
1056 Improvements can be brought on this part of the software to make sure that the DAQ can safely
1057 shutdown at any moment.

1058

1059 **A.6 Software export**

1060 In section A.2 was discussed the fact that the DAQ as written in its last version is not a standalone
1061 software. It is possible to make it a standalone program that could be adapted to any VME setup
1062 using V1190A and V1718 modules by creating a GUI for the software or by printing the log mes-
1063 sages that are normally printed in the webDCS through the log file, directly into the terminal. This
1064 method was used by the DAQ up to version 3.0 moment where the webDCS was completed. Also, it
1065 is possible to check branches of DAQ v2.X to have example of communication through a terminal.

1066

1067 DAQ v2.X is nonetheless limited in it's possibilities and requires a lot of offline manual interven-
1068 tions from the users. Indeed, there is no communication of the software with the detectors' power
1069 supply system that would allow for a user a predefine a list of voltages to operate the detectors at

step	actions of webDCS	status of DAQ	<code>__runstatuspath</code>
1	launch DAQ ramp voltages ramping over wait for currents stabilization	readout of IniFile configuration of TDCs	INIT
2		configuration done send DAQ ready wait for START signal	DAQ_RDY
3	waiting time over send START		START
4	wait for run to end monitor DAQ run status	data taking ongoing check for KILL signal	RUNNING
5		run over send DAQ_RDY wait for next DCS signal	DAQ_RDY
6	ramp voltages ramping over wait for currents stabilization		DAQ_RDY
3	waiting time over send START		START
4	wait for run to end monitor DAQ run status	update IniFile information data taking ongoing check for KILL signal	RUNNING
5		run over send DAQ_RDY wait for next DCS signal	DAQ_RDY
7	send command STOP	DAQ shuts down	STOP

Table A.1: Inter-process communication cycles in between the webDCS and the DAQ through file string signals.

1070 and loop over to take data without any further manual intervention. In v2.X, the data is taken for a
1071 single detector setting and at the end of each run, the softwares asks the user if he intends on taking
1072 more runs. If so, the software invites the user to set the operating voltages accordingly to what is
1073 necessary and to manual update the configuration file in consequence. This working mode can be a
1074 very first approach before an evolution and has been successfully used by colleagues from different
1075 collaborations.

1076

1077 For a more robust operation, it is recommended to develop a GUI or a web application to inter-
1078 face the DAQ. Moreover, to limit the amount of manual interventions, and thus the probability to
1079 make mistakes, it is also recommended to add an extra feature into the DAQ by installing the HV
1080 Wrapper library provided by CAEN of which an example of use in a similar DAQ software devel-
1081 opped by a master student of UGent, and called TinyDAQ, is provided on UGent's github. Then, this
1082 HV Wrapper will help you communicating with and give instructions to a CAEN HV powered crate
1083 and can be added into the DAQ at the same level where the communication with the user was made
1084 in DAQ v2.X. In case you are using another kind of power system for your detectors, it is stringly
1085 adviced to use HV modules or crates that can be remotely controled via a using C++ libraries.

1086

B

1087

1088

Details on the offline analysis package

1089 The data collected in GIF++ thanks to the DAQ described in Appendix A is difficult to interpret by
1090 a human user that doesn't have a clear idea of the raw data architecture of the ROOT data files. In
1091 order to render the data human readable, a C++ offline analysis tool was designed to provide users
1092 with detector by detector histograms that give a clear overview of the parameters monitored during
1093 the data acquisition [42]. In this appendix, details about this software in the context of GIF++, as of
1094 how the software was written and how it functions will be given.

1095 B.1 GIF++ Offline Analysis file tree

1096 GIF++ Offline Analysis source code is fully available on github at https://github.com/aafagot/GIF_OfflineAnalysis. The software requires ROOT as non-optionnal dependency
1097 as it takes ROOT files in input and write an output ROOT file containing histograms. To compile the
1098 GIF++ Offline Analysis project is compiled with cmake. To compile, first a build/ directory must
1099 be created to compile from there:

```
1101 mkdir build  
1102 cd build  
1103 cmake ..  
1104 make  
1105 make install
```

1103 To clean the directory and create a new build directory, the bash script cleandir.sh can be used:

```
1104  
1105 ./cleandir.sh
```

1106 The source code tree is provided below along with comments to give an overview of the files' con-
1107 tent. The different objects created for this project (`Infrastructure`, `Trolley`, `RPC`, `Mapping`, `RPCHit`,
1108 `RPCCluster` and `Inifile`) will be described in details in the following sections.

1109

```

GIF_OfflineAnalysis
├── bin
│   └── offlineanalysis ..... EXECUTABLE
├── build..... CMAKE COMPILATION DIRECTORY
└── ...
    ├── include..... LIST OF C++ HEADER FILES
    │   ├── Cluster.h..... DECLARATION OF OBJECT RPCCLUSTER
    │   ├── Current.h..... DECLARATION OF GETCURRENT ANALYSIS MACRO
    │   ├── GIFTrolley.h..... DECLARATION OF OBJECT TROLLEY
    │   ├── Infrastructure.h..... DECLARATION OF OBJECT INFRASTRUCTURE
    │   ├── IniFile.h..... DECLARATION OF OBJECT INI FILE FORINI PARSER
    │   ├── Mapping.h..... DECLARATION OF OBJECT MAPPING
    │   ├── MsgSvc.h..... DECLARATION OF OFFLINE LOG MESSAGES
    │   ├── OfflineAnalysis.h..... DECLARATION OF DATA ANALYSIS MACRO
    │   ├── RPCTracker.h..... DECLARATION OF OBJECT RPC
    │   ├── RPCHit.h..... DECLARATION OF OBJECT RPCHIT
    │   ├── types.h..... DEFINITION OF USEFUL VARIABLE TYPES
    │   └── utils.h..... DECLARATION OF USEFUL FUNCTIONS
    ├── obj..... BINARY FILES CREATED BY COMPILER
    └── ...
        ├── src..... LIST OF C++ SOURCE FILES
        │   ├── Cluster.cc..... DEFINITION OF OBJECT RPCCLUSTER
        │   ├── Current.cc..... DEFINITION OF GETCURRENT ANALYSIS MACRO
        │   ├── GIFTrolley.cc..... DEFINITION OF OBJECT TROLLEY
        │   ├── Infrastructure.cc..... DEFINITION OF OBJECT INFRASTRUCTURE
        │   ├── IniFile.cc..... DEFINITION OF OBJECT INI FILE FORINI PARSER
        │   ├── main.cc..... MAIN FILE
        │   ├── Mapping.cc..... DEFINITION OF OBJECT MAPPING
        │   ├── MsgSvc.cc..... DEFINITION OF OFFLINE LOG MESSAGES
        │   ├── OfflineAnalysis.cc..... DEFINITION OF DATA ANALYSIS MACRO
        │   ├── RPCTracker.cc..... DEFINITION OF OBJECT RPC
        │   ├── RPCHit.cc..... DEFINITION OF OBJECT RPCHIT
        │   └── utils.cc..... DEFINITION OF USEFUL FUNCTIONS
        ├── cleandir.sh..... BASH SCRIPT TO CLEAN BUILD DIRECTORY
        ├── CMakeLists.txt..... SET OF INSTRUCTIONS FOR CMAKE
        ├── config.h.in..... DEFINITION OF VERSION NUMBER
        └── README.md..... README FILE FOR GITHUB

```

1110

B.2 Usage of the Offline Analysis

1111

In order to use the Offline Analysis tool, it is necessary to know the Scan number and the HV Step of the run that needs to be analysed. This information needs to be written in the following format:

1113

1114

```
Scan00XXXX_HVY
```

1115

where XXXX is the scan ID and Y is the high voltage step (in case of a high voltage scan, data will be taken for several HV steps). This format corresponds to the base name of data files in the database

1116

1117 of the GIF++ webDCS. Usually, the offline analysis tool is automatically called by the webDCS at
 1118 the end of data taking or by a user from the webDCS panel if an update of the tool was brought.
 1119 Nonetheless, an expert can locally launch the analysis for tests on the GIF++ computer, or a user can
 1120 get the code on its local machine from github and download data from the webDCS for its own anal-
 1121 ysis. To launch the code, the following command can be used from the `GIF_OfflineAnalysis` folder:

```
1122
1123 bin/offlineanalysis /path/to/Scan00XXXX_HVY
```

1124 where, `/path/to/Scan00XXXX_HVY` refers to the local data files. Then, the offline tool will by itself
 1125 take care of finding all available ROOT data files present in the folder, as listed below:

- 1126 • `Scan00XXXX_HVY_DAQ.root` containing the TDC data as described in Appendix ?? (events, hit
 1127 and timestamp lists), and
- 1128 • `Scan00XXXX_HVY_CAEN.root` containing the CAEN mainframe data recorded by the monitor-
 1129 ing tool webDCS during data taking (HVs and currents of every HV channels). This file is
 1130 created independently of the DAQ.

1131 **B.2.1 Output of the offline tool**

1132 **B.2.1.1 ROOT file**

1133 The analysis gives in output ROOT datafiles that are saved into the data folder and called using the
 1134 naming convention `Scan00XXXX_HVY_Offline.root`. Inside those, a list of `TH1` histograms can be
 1135 found. Its size will vary as a function of the number of detectors in the setup as each set of histograms
 1136 is produced detector by detector. For each partition of each chamber, can be found:

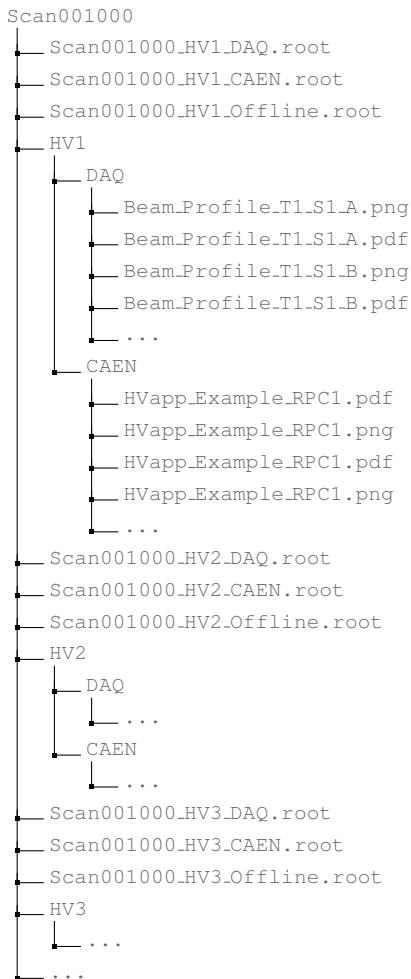
- 1137 • `Time_Profile_Tt_Sc_p` shows the time profile of all recorded events (number of events per
 1138 time bin),
- 1139 • `Hit_Profile_Tt_Sc_p` shows the hit profile of all recorded events (number of events per chan-
 1140 nel),
- 1141 • `Hit_Multiplicity_Tt_Sc_p` shows the hit multiplicity (number of hits per event) of all recorded
 1142 events (number of occurrences per multiplicity bin),
- 1143 • `Strip_Mean_Noise_Tt_Sc_p` shows noise/gamma rate per unit area for each strip in a se-
 1144 lected time range. After filters are applied on `Time_Profile_Tt_Sc_p`, the filtered version
 1145 of `Hit_Profile_Tt_Sc_p` is normalised to the total integrated time and active detection area
 1146 of a single channel,
- 1147 • `Strip_Activity_Tt_Sc_p` shows noise/gamma activity for each strip (normalised version of
 1148 previous histogram - strip activity = strip rate / average partition rate),
- 1149 • `Strip_Homogeneity_Tt_Sc_p` shows the *homogeneity* of a given partition ($\text{homogeneity} = \exp(-\text{strip rates standard deviation(strip rates in partition/average partition rate)})$),
- 1150 • `mask_Strip_Mean_Noise_Tt_Sc_p` shows noise/gamma rate per unit area for each masked
 1152 strip in a selected time range. Offline, the user can control the noise/gamma rate and decide to
 1153 mask the strips that are judged to be noisy or dead. This is done via the *Masking Tool* provided
 1154 by the webDCS,

- 1155 ● `mask_Strip_Activity_Tt_Sc_p` shows noise/gamma activity per unit area for each masked
1156 strip with respect to the average rate of active strips,
- 1157 ● `NoiseCSize_H_Tt_Sc_p` shows noise/gamma cluster size, a cluster being constructed out of
1158 adjacent strips giving a signal at the *same time* (hits within a time window of 25 ns),
- 1159 ● `NoiseCMult_H_Tt_Sc_p` shows noise/gamma cluster multiplicity (number of reconstructed
1160 clusters per event),
- 1161 ● `Chip_Mean_Noise_Tt_Sc_p` shows the same information than `Strip_Mean_Noise_Tt_Scp` us-
1162 ing a different binning (1 chip corresponds to 8 strips),
- 1163 ● `Chip_Activity_Tt_Sc_p` shows the same information than `Strip_Activity_Tt_Scp` using
1164 chip binning,
- 1165 ● `Chip_Homogeneity_Tt_Sc_p` shows the homogeneity of a given partition using chip binning,
- 1166 ● `Beam_Profile_Tt_Sc_p` shows the estimated beam profile when taking efficiency scan. This
1167 is obtained by filtering `Time_Profile_Tt_Sc_p` to only consider the muon peak where the
1168 noise/gamma background has been subtracted. The resulting hit profile corresponds to the
1169 beam profile on the detector channels,
- 1170 ● `L0_Efficiency_Tt_Sc_p` shows the level 0 efficiency that was estimated **without** muon track-
1171 ing,
- 1172 ● `MuonCSize_H_Tt_Sc_p` shows the level 0 muon cluster size that was estimated **without** muon
1173 tracking, and
- 1174 ● `MuonCMult_H_Tt_Sc_p` shows the level 0 muon cluster multiplicity that was estimated **without**
1175 muon tracking.

1176 In the histogram labels, t stands for the trolley number (1 or 3), c for the chamber slot label in
1177 trolley t and p for the partition label (A, B, C or D depending on the chamber layout) as explained
1178 in Chapter ??.

1179 In the context of GIF++, an extra script called by the webDCS is called to extract the histograms
1180 from the ROOT files. The histograms are then stored in PNG and PDF formats into the correspond-
1181 ing folder (a single folder per HV step, so per ROOT file). the goal is to then display the histograms
1182 on the Data Quality Monitoring (DQM) page of the webDCS in order for the users to control the
1183 quality of the data taking at the end of data taking. An example of histogram organisation is given
1184 below:

1185
1186



1187 *Here can put some screens from the webDCS to show the DQM and the plots available to users.*

1188

1189 **B.2.1.2 CSV files**

1190 Moreover, up to 3 CSV files can be created depending on which ones of the 3 input files were in the
 1191 data folder:

- 1192 ● `Offline-Corrupted.csv` , is used to keep track of the amount of data that was corrupted and
 1193 removed from old data format files that don't contain any data quality flag.
- 1194 ● `Offline-Current.csv` , contains the summary of the currents and voltages applied on each
 1195 RPC HV channel.
- 1196 ● `Offline-L0-EffC1.csv` , is used to write the efficiencies, cluster size and cluster multiplicity
 1197 of efficiency runs. Note that `L0` refers here to *Level 0* and means that the results of efficiency and
 1198 clusterization are a first approximation calculated without performing any muon tracking in

1199 between the different detectors. This offline tool provides the user with a preliminar calculation
 1200 of the efficiency and of the muon event parameters. Another analysis software especially
 1201 dedicated to muon tracking is called on selected data to retrieve the results of efficiency and
 1202 muon clusterization using a tracking algorithm to discriminate noise or gamma from muons
 1203 as muons are the only particles that pass through the full setup, leaving hits than can be used
 1204 to reconstruct their tracks.

- 1205 • `Offline-Rate.csv`, is used to write the noise or gamma rates measured in the detector readout
 1206 partitions.

1207 Note that these 4 CSV files are created along with their *headers* (`Offline-[...]-Header.csv`
 1208 containing the names of each data columns) and are automatically merged together when the offline
 1209 analysis tool is called from the webDCS, contrary to the case where the tool is runned locally from
 1210 the terminal as the merging bash script is then not called. Thus, the resulting files, used to make
 1211 official plots, are:

- 1212 • `Corrupted.csv`,
 1213 • `Current.csv`,
 1214 • `L0-EffCl.csv`.
 1215 • `Rate.csv`.

1216 B.3 Analysis inputs and information handling

1217 The usage of the Offline Analysis tool as well as its output have been presented in the previous section.
 1218 It is now important to dig further and start looking at the source code and the inputs necessary
 1219 for the tool to work. Indeed, other than the raw ROOT data files that are analysed, more information
 1220 needs to be imported inside of the program to perform the analysis such as the description of the
 1221 setup inside of GIFT++ at the time of data taking (number of trolleys, of RPCs, dimensions of the
 1222 detectors, etc...) or the mapping that links the TDC channels to the coresponding RPC channels in
 1223 order to translate the TDC information into human readable data. 2 files are used to transmit all this
 1224 information:

- 1225
 1226 • `Dimensions.ini`, that provides the necessary setup and RPC information, and
 1227 • `ChannelsMapping.csv`, that gives the link between the TDC and RPC channels as well as the
 1228 *mask* for each channel (masked or not?).

1229 B.3.1 Dimensions file and InFile parser

1230 This input file, present in every data folder, allows the analysis tool to know of the number of active
 1231 trolleys, the number of active RPCs in those trolleys, and the details about each RPCs such as
 1232 the number of RPC gaps, the number of pseudo-rapidity partitions (for CMS-like prototypes), the
 1233 number of strips per partion or the dimensions. To do so, there are 3 types of groups in the INI file
 1234 architecture. A first general group, appearing only once at the head of the document, gives information
 1235 about the number of active trolleys as well as their IDs, as presented in Source Code B.1. For

1236 each active trolley, a group similar to Source Code B.2 can be found containing information about
 1237 the number of active detectors in the trolley and their IDs. Each trolley group as a `Tt` name format,
 1238 where `t` is the trolley ID. Finally, for each detector stored in slots of an active trolley, there is a group
 1239 providing information about their names and dimensions, as shown in Source Code B.3. Each slot
 1240 group as a `TtSs` name format, where `s` is the slot ID of trolley `t` where the active RPC is hosted.

```
1241 [General]
1242 nTrolleys=2
  TrolleysID=13
```

1243 *Source Code B.1: Example of `[General]` group as might be found in `Dimensions.ini`. In Gif++, only 2 trolleys are available to hold RPCs and place them inside of the bunker for irradiation. The IDs of the trolleys are written in a single string as "13" and then read character by character by the program.*

```
1244 [T1]
  nSlots=4
  SlotsID=1234
```

1245 *Source Code B.2: Example of trolley group as might be found in `Dimensions.ini`. In this example, the file tells that there are 4 detectors placed in the holding slots of the trolley `T1` and that their IDs, written as a single string variable, are 1, 2, 3 and 4.*

```
1246 [T1S1]
  Name=RE2-2-NPD-BARC-8
  Partitions=3
  Gaps=3
  Gap1=BOT
  Gap2=TN
  Gap3=TW
  AreaGap1=11694.25
  AreaGap2=6432
  AreaGap3=4582.82
  Strips=32
  ActiveArea-A=157.8
  ActiveArea-B=121.69
  ActiveArea-C=93.03
```

1247 *Source Code B.3: Example of slot group as might be found in `Dimensions.ini`. In this example, the file provides information about a detector named `RE2-2-NPD-BARC-8`, having 3 pseudo-rapidity readout partitions and stored in slot `S1` of trolley `T1`. This is a CMS RE2-2 type of detector. This information will then be used for example to compute the rate per unit area calculation.*

1248 This information is readout and stored in a C++ object called `IniFile`, that parses the information
 1249 in the INI input file and stores it into a local buffer for later use. This INI parser is the exact same
 1250 one that was previously developed for the Gif++ DAQ and described in Appendix A.5.2.

1251 **B.3.2 TDC to RPC link file and Mapping**

1252 The same way the INI dimension file information is stored using `map`, the channel mapping and mask
 1253 information is stored and accessed through `map`. First of all, the mapping CSV file is organised into
 1254 3 columns separated by tabulations (and not by commas, as expected for CSV files as it is easier using
 1255 streams to read tab or space separated data using C++):

1256

1257 RPC_channel TDC_channel mask

1258 using as formatting for each field:

1259
1260 TSCCC TCCC M

1261 TSCCC is a 5-digit integer where T is the trolley ID, s the slot ID in which the RPC is held insite
1262 the trolley T and ccc is the RPC channel number, or *strip* number, that can take values up to
1263 3-digits depending on the detector,

1264 TCCC is a 4 digit integer where T is the TDC ID, ccc is the TDC channel number that can take values
1265 in between 0 and 127, and

1266 M is a 1-digit integer indicating if the channel should be considered ($M = 1$) or discarded ($M = 0$)
1267 during analysis.

1268 This mapping and masking information is readout and stored thanks to the object `Mapping`, pre-
1269 sented in Source Code B.4. Similarly to `IniFile` objects, this class has private methods. The first
1270 one, `Mapping::CheckIfNewLine()` is used to find the newline character '`\n`' or return character
1271 '`\r`' (depending on which kind of operating system interacted with the file). This is used for the
1272 simple reason that the masking information has been introduced only during the year 2017 but the
1273 channel mapping files exist since 2015 and the very beginning of data taking at GIF++. This means
1274 that in the older data folders, before the upgrade, the channel mapping file only had 2 columns, the
1275 RPC channel and the TDC channel. For compatibility reasons, this method helps controling the
1276 character following the readout of the 2 first fields of a line. In case any end of line character is
1277 found, no mask information is present in the file and the default $M = 1$ is used. On the contrary, if
1278 the next character was a tabulation or a space, the mask information is present.

1279 Once the 3 fields have been readout, the second private method `Mapping::CheckIfTDCCh()` is
1280 used to control that the TDC channel is an existing TDC channel. Finally, the information is stored
1281 into 3 different maps (`Link`, `ReverseLink` and `Mask`) thanks to the public method `Mapping::Read()`.
1282 `Link` allows to get the RPC channel by knowing the TDC channel while `ReverseLink` does the op-
1283 posite by returning the TDC channel by knowing the RPC channel. Finally, `Mask` returns the mask
1284 associated to a given RPC channel.

```

1285 typedef map<Uint,Uint> MappingData;
1286
class Mapping {
    private:
        bool          CheckIfNewLine(char next);
        bool          CheckIfTDCCh(Uint channel);
        string         FileName;
        MappingData   Link;
        MappingData   ReverseLink;
        MappingData   Mask;
        int           Error;
1287
    public:
        Mapping();
        Mapping(string baseName);
        ~Mapping();

        void SetFileName(const string filename);
        int Read();
        Uint GetLink(Uint tdcchannel);
        Uint GetReverse(Uint rpcchannel);
        Uint GetMask(Uint rpcchannel);
    };

```

1287 *Source Code B.4: Description of C++ object Mapping used as a parser for the channel mapping and mask file.*

1288 B.4 Description of GIF++ setup within the Offline Analysis tool

1289 In the previous section, the tool input files have been discussed. The dimension file information is
 1290 stored in a map hosted by the `IniFile` object. But this information is then used to create a series of
 1291 new objects that helps defining the GIF++ infrastructure directly into the Offline Analysis. Indeed,
 1292 from the `RPC`, to the more general `Infrastructure`, every element of the GIF++ infrastrucutre is
 1293 recreated for each data analysis based on the information provided in input. All this information
 1294 about the infrastructure will be used to assign each hit signal to a specific strip channel of a specific
 1295 detector, and having a specific active area. This way, rate per unit area calculation is possible.
 1296

1297 B.4.1 RPC objects

1298 `RPC` objects have been developped to represent physical active detectors in GIF++ at the moment
 1299 of data taking. Thus, there are as many `RPC` objects created during the analysis than there were
 1300 active `RPCs` tested during a run. Each `RPC` hosts the information present in the corresponding INI
 1301 slot group, as shown in B.3, and organises it using a similar architecture. This can be seen from
 1302 *Source Code B.5*.

1303 To make the object more compact, the lists of gap labels, of gap active areas and strip active
 1304 areas are stored into `vector` dynamical containers. `RPC` objects are always contructed thanks to the
 1305 dimension file information stored into the `IniFILE` and their ID, using the format `TtSs`. Using the
 1306 `RPC` ID, the constructor calls the methods of `IniFILE` to initialise the `RPC`. The other constructors
 1307 are not used but exist in case of need. Finally, some getters have been written to access the different
 1308 private parameters storing the detector information.

```

1309 class RPC{
1310     private:
1311         string      name;           //RPC name as in webDCS database
1312         Uint        nGaps;          //Number of gaps in the RPC
1313         Uint        nPartitions;    //Number of partitions in the RPC
1314         Uint        nStrips;        //Number of strips per partition
1315         vector<string> gaps;       //List of gap labels (BOT, TOP, etc...)
1316         vector<float>  gapGeo;        //List of gap active areas
1317         vector<float>  stripGeo;      //List of strip active areas
1318
1319     public:
1320         RPC();
1321         RPC(string ID, IniFile* geofile);
1322         RPC(const RPC& other);
1323         ~RPC();
1324         RPC& operator=(const RPC& other);
1325
1326         string GetName();
1327         Uint GetNGaps();
1328         Uint GetNPartitions();
1329         Uint GetNStrips();
1330         string GetGap(Uint g);
1331         float GetGapGeo(Uint g);
1332         float GetStripGeo(Uint p);
1333     };

```

1311 *Source Code B.5: Description of C++ objects RPC that describe each active detectors used during data taking.*

1312 B.4.2 Trolley objects

1313 Trolley objects have been developed to represent physical active trolleys in GIFT++ at the moment
 1314 of data taking. Thus, there are as many trolley objects created during the analysis than there were
 1315 active trolleys hosting tested RPCs during a run. Each Trolley hosts the information present in the
 1316 corresponding INI trolley group, as shown in B.2, and organises it using a similar architecture. In
 1317 addition to the information hosted in the INI file, these objects have a dynamical container of RPC
 1318 objects, representing the active detectors the active trolley was hosting at the time of data taking.
 1319 This can be seen from Source Code B.6.

1320 Trolley objects are always constructed thanks to the dimension file information stored into the
 1321 IniFILE and their ID, using the format Tt. Using the Trolley ID, the constructor calls the methods
 1322 of IniFILE to initialise the Trolley. Retrieving the information of the RPC IDs via SlotsID, a new
 1323 RPC is constructed and added to the container RPCs for each character in the ID string. The other
 1324 constructors are not used but exist in case of need. Finally, some getters have been written to access
 1325 the different private parameters storing the trolley and detectors information.

```

1326 class Trolley{
1327     private:
        Uint          nSlots; //Number of active RPCs in the considered trolley
        string        SlotsID; //Active RPC IDs written into a string
        vector<RPC*> RPCs;   //List of active RPCs
1328
    public:
        //Constructors, destructor and operator =
        Trolley();
        Trolley(string ID, IniFile* geofile);
        Trolley(const Trolley& other);
        ~Trolley();
        Trolley& operator=(const Trolley& other);
1329
        //Get GIFTrolley members
        Uint  GetNSlots();
        string GetSlotsID();
        Uint   GetSlotID(Uint s);
1330
        //Manage RPC list
        RPC*  GetRPC(Uint r);
        void  DeleteRPC(Uint r);
1331
        //Methods to get members of RPC objects stored in RPCs
        string GetName(Uint r);
        Uint   GetNGaps(Uint r);
        Uint   GetNPartitions(Uint r);
        Uint   GetNStrips(Uint r);
        string GetGap(Uint r, Uint g);
        float  GetGapGeo(Uint r, Uint g);
        float  GetStripGeo(Uint r, Uint p);
    };

```

Source Code B.6: Description of C++ objects Trolley that describe each active trolley used during data taking.

1329 B.4.3 Infrastructure object

1330 The `Infrastructure` object has been developped to represent the GIFT++ bunker area dedicated to
 1331 CMS RPC experiments. With this very specific object, all the information about the CMS RPC
 1332 setup within GIFT++ at the moment of data taking is stored. It hosts the information present in the
 1333 corresponding INI general group, as shown in B.1, and organises it using a similar architecture. In
 1334 addition to the information hosted in the INI file, this object have a dynamical container of `Trolley`
 1335 objects, representing the active tolleys in GIFT++ area. This can be seen from Source Code B.7.

1336 The `Infrastructure` object is always contructed thanks to the dimension file information stored
 1337 into the `IniFILE`. Retrieving the information of the trolley IDs via `TrolleysID`, a new `Trolley` is
 1338 constructed and added to the container `Trolleys` for each character in the ID `string`. By extension,
 1339 it is easy to understand that the process described in Section B.4.2 for the construction of RPCs
 1340 takes place when a trolley is constructed. The other constructors are not used but exist in case of
 1341 need. Finally, some getters have been written to access the different private parameters storing the
 1342 infrastructure, tolleys and detectors information.

```

1343
class Infrastructure {
    private:
        Uint             nTrolleys;   //Number of active Trolleys in the run
        string          TrolleysID; //Active trolley IDs written into a string
        vector<Trolley*> Trolleys; //List of active Trolleys (struct)

    public:
        //Constructors and destructor
        Infrastructure();
        Infrastructure(IniFile* geofile);
        Infrastructure(const Infrastructure& other);
        ~Infrastructure();
        Infrastructure& operator=(const Infrastructure& other);

        //Get Infrastructure members
        Uint GetNTrolleys();
        string GetTrolleysID();
        Uint GetTrolleyID(Uint t);

1344
        //Manage Trolleys
        Trolley* GetTrolley(Uint t);
        void DeleteTrolley(Uint t);

        //Methods to get members of GIFTrolley objects stored in Trolleys
        Uint GetNSlots(Uint t);
        string GetSlotsID(Uint t);
        Uint GetSlotID(Uint t, Uint s);
        RPC* GetRPC(Uint t, Uint r);

        //Methods to get members of RPC objects stored in RPCs
        string GetName(Uint t, Uint r);
        Uint GetNGaps(Uint t, Uint r);
        Uint GetNPartitions(Uint t, Uint r);
        Uint GetNStrips(Uint t, Uint r);
        string GetGap(Uint t, Uint r, Uint g);
        float GetGapGeo(Uint t, Uint r, Uint g);
        float GetStripGeo(Uint t, Uint r, Uint p);
};


```

Source Code B.7: Description of C++ object *Infrastructure* that contains the full information about CMS RPC experiment in GIF++.

1346 B.5 Handeling of data

1347 As discussed in Appendix A.4.2, the raw data as a `TTree` architecture where every entry is related to
 1348 a trigger signal provided by a muon or a random pulse, whether the goal of the data taking was to
 1349 measure the performance of the detector or the noise/gamma background respectively. Each of these
 1350 entries, referred also as events, contain a more or less full list of hits in the TDC channels to which
 1351 the detectors are connected. To this list of hits corresponds a list of time stamps, marking the arrival
 1352 of the hits within the TDC channel.

1353 The infrastructure of the CMS RPC experiment within GIF++ being defined, combining the
 1354 information about the raw data with the information provided by both the mapping/mask file and the
 1355 dimension file allows to build new physical objects that will help in computing efficiency or rates.

1356 B.5.1 RPC hits

1357 The raw data stored in the ROOT file as output of the GIFT++ DAQ, is readout by the analysis tool
 1358 using the structure `RAWData` presented in Source Code B.9 that differs from the structure presented
 1359 in Appendix A.4.2 as it is not meant to hold all of the data contained in the ROOT file. In this sense,
 1360 this structure is in the case of the offline analysis tool not a dynamical object and will only be storing
 1361 a single event contained in a single entry of the `TTree`.

```
1362
1363 class RPCHit {
    private:
        Uint Channel;      //RPC channel according to mapping (5 digits)
        Uint Trolley;     //0, 1 or 3 (1st digit of the RPC channel)
        Uint Station;     //Slot where is held the RPC in Trolley (2nd digit)
        Uint Strip;       //Physical RPC strip where the hit occurred (last 3
                           → digits)
        Uint Partition;   //Readout partition along eta segmentation
        float TimeStamp; //Time stamp of the arrival in TDC

    public:
        //Constructors, destructor & operator =
        RPCHit();
        RPCHit(Uint channel, float time, Infrastructure* Infra);
        RPCHit(const RPCHit& other);
        ~RPCHit();
        RPCHit& operator=(const RPCHit& other);

        //Get RPCHit members
        Uint GetChannel();
        Uint GetTrolley();
        Uint GetStation();
        Uint GetStrip();
        Uint GetPartition();
        float GetTime();
    };

    typedef vector<RPCHit> HitList;
    typedef struct GIFHitList { HitList rpc[NTROLLEYS][NSLOTS][NPARTITIONS]; } →
        GIFHitList;

    bool SortHitbyStrip(RPCHit h1, RPCHit h2);
    bool SortHitbyTime(RPCHit h1, RPCHit h2);
}
```

1364 *Source Code B.8: Description of C++ object `RPCHit`.*

```
1365
1366 struct RAWData{
    int iEvent;           //Event i
    int TDCNHits;        //Number of hits in event i
    int QFlag;            //Quality flag list (1 flag digit per TDC)
    vector<Uint> *TDCCh; //List of channels giving hits per event
    vector<float> *TDCTS; //List of the corresponding time stamps
};
```

1366 *Source Code B.9: Description of C++ structure `RAWData`.*

1367 Each member of the structure is then linked to the corresponding branch of the ROOT data tree,
 1368 as shown in the example of Source Code B.10, and using the method `GetEntry(int i)` of the ROOT
 1369 class `TTree` will update the state of the members of `RAWData`.

```

1370   TTree* dataTree = (TTree*)dataFile.Get("RAWData");
1371   RAWData data;
1372
1373   dataTree->SetBranchAddress("EventNumber", &data.iEvent);
1374   dataTree->SetBranchAddress("number_of_hits", &data.TDCNHits);
1375   dataTree->SetBranchAddress("Quality_flag", &data.QFlag);
1376   dataTree->SetBranchAddress("TDC_channel", &data.TDCCh);
1377   dataTree->SetBranchAddress("TDC_TimeStamp", &data.TDCTS);

```

1372 *Source Code B.10: Example of link in between RAWData and TTree.*

1373 The data is then analysed entry by entry and to each element of the TDC channel list, a `RPCHit` is
 1374 constructed by linking each TDC channel to the corresponding RPC channel thanks to the `Mapping`
 1375 object. The information carried by the RPC channel format allows to easily retrieve the trolley and
 1376 slot from which the hit was recorded (see section B.3.2). Using these 2 values, the readout partition
 1377 can be found by knowing the strip channel and comparing it with the number of partitions and strips
 1378 per partition stored into the `Infrastructure` object.

1379 Thus `RPCHit` objects are then stored into 3D dynamical list called `GIFHitList` (Source Code B.9)
 1380 where the 3 dimensions refer to the 3 layers of the readout in `GIF++` : in the bunker there are *trolleys*
 1381 (τ) holding detectors in *slots* (s) and each detector readout is divided into 1 or more pseudo-rapidity
 1382 *partitions* (p). Using these 3 information allows to assign an address to each readout partition and
 1383 this address will point to a specific hit list.

1384

1385 **B.5.2 Clusters of hits**

1386 All the hits contained in the ROOT file have been sorted into the different hit lists through the
 1387 `GIFHitList`. At this point, it is possible to start looking for clusters. A cluster is a group of adjacent
 1388 strips getting hits within a time window of 25 ns. These strips are then assumed to be part of the same
 1389 physical avalanche signal generated by a muon passing through the chamber or by the interaction of
 1390 a gamma stopping into the electrodes of the RPCs.

1391 To keep the cluster information, `RPCCluster` objects have been defined as shown in Source
 1392 Code B.11. Using the information of each individual `RPCHit` taken out of the hit list, it stores
 1393 the cluster size (number of adjacent strips composing the cluster), the first and last hit, the center for
 1394 spatial reconstruction and finally the start and stop time stamps as well as te time spread in between
 1395 the first and last hit.

```

1396
class RPCCluster{
    private:
        Uint ClusterSize; //Size of cluster #ID
        Uint FirstStrip; //First strip of cluster #ID
        Uint LastStrip; //Last strip of cluster #ID
        float Center; //Center of cluster #ID ((first+last)/2)
        float StartStamp; //Time stamp of the earliest hit of cluster #ID
        float StopStamp; //Time stamp of the latest hit of cluster #ID
        float TimeSpread; //Time difference between earliest and latest hits
                           //of cluster #ID
    public:
        //Constructors, destructor & operator =
        RPCCluster();
        RPCCluster(HitList List, Uint cID, Uint cSize, Uint first, Uint firstID);
        RPCCluster(const RPCCluster& other);
        ~RPCCluster();
        RPCCluster& operator=(const RPCCluster& other);

        //Get Cluster members
        Uint GetID();
        Uint GetSize();
        Uint GetFirstStrip();
        Uint GetLastStrip();
        float GetCenter();
        float GetStart();
        float GetStop();
        float GetSpread();
    };

typedef vector<RPCCluster> ClusterList;

//Other functions to build cluster lists out of hit lists
void BuildClusters(HitList &cluster, ClusterList &clusterList);
void Clusterization(HitList &hits, TH1 *hcSize, TH1 *hcMult);

```

1398 *Source Code B.11: Description of C++ object Cluster.*

1399 To investigate the hit list of a given detector partition, the function `Clusterization()` defined
 1400 in `include/Cluster.h` needs the hits in the list to be time sorted. This is achieved by calling func-
 1401 tion `sort()` of library `<algorithm>` using the comparator `SortHitbyTime(RPCHit h1, RPCHit h2)`
 1402 defined in `include/RPCHit.h` that returns `true` if the time stamp of hit `h1` is lower than that of `h2`.
 1403 A first isolation of strips is made only based on time information. All the hits within the 25 ns win-
 1404 dow are taken separately from the rest. Then, this sub-list of hits is sorted this time by ascending
 1405 strip number, using this time the comparator `SortHitbyStrip(RPCHit h1, RPCHit h2)`. Finally, the
 1406 groups of adjacent strips are used to construct `RPCCluster` objects that are then stored in a temporary
 1407 list of clusters that is at the end of the process used to know how many clusters were reconstructed
 1408 and to fill their sizes into an histogram that will allows to know the mean size of muon or gamma
 1409 clusters.
 1410

1411 **B.6 DAQ data Analysis**

1412 All the ingredients to analyse GIF++ data have been defined. This section will focus on the different
 1413 part of the analysis performed on the data, from determining the type of data the tool is dealing with

1414 to calculating the rate in each detector or reconstructing muon or gamma clusters.

1415 B.6.1 Determination of the run type

1416 In GIF++, both the performance of the detectors in detecting muons in an irradiated environment and
 1417 the gamma background can be independantly measured. These corresponds to different run types
 1418 and thus, to different TDC settings giving different data to look at.

1419
 1420 In the case of performance measurements, the trigger for data taking is provided by the coïncidence
 1421 of several scintillators when muons from the beam passing through the area are detected. Data
 1422 is collected in a 600 ns wide window around the arrival of muons in the RPCs. The expected time
 1423 distribution of hits is shown in Figure B.1a. The muon peak is clearly visible in the center of the
 1424 distribution and is to be extracted from the gamma background that composes the flat part of the
 1425 distribution.

1426 On the other hand, gamma background or noise measurements are focussed on the non muon
 1427 related physics and the trigger needs to be independant from the muons to give a good measurement
 1428 of the gamma/noise distribution as seen by the detectors. The trigger is then provided by a pulse
 1429 generator at a frequency of 300 Hz whose pulse is not likely to be on time with a muon. In order
 1430 to increase the integrated time without increasing the acquisition time too much, the width of the
 1431 acquisition windows are increased to 10 μ s. The time distribution of the hits is expected to be flat, as
 1432 shown by Figure B.1b.

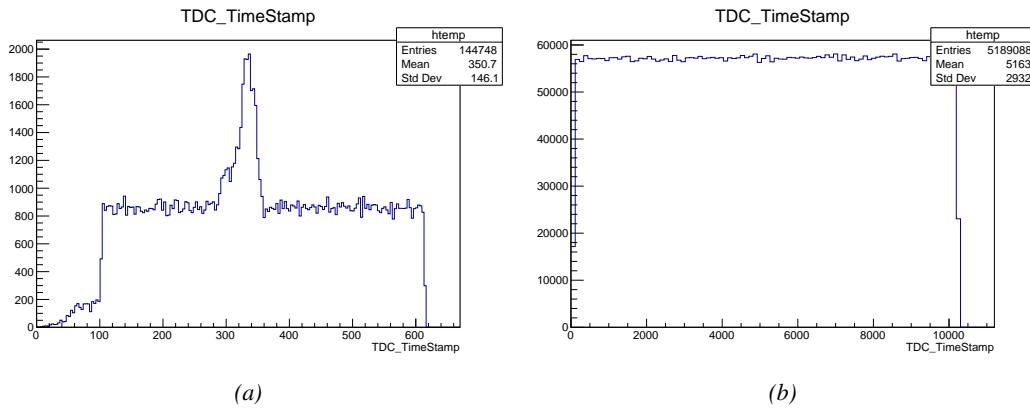


Figure B.1: Example of expected hit time distributions in the cases of efficiency (Figure B.1a) and noise/gamma rate per unit area (Figure B.1b) measurements as extracted from the raw ROOT files. The unit along the x-axis corresponds to ns. The fact that "the" muon peak is not well defined in Figure B.1a is due to the contribution of all the RPCs being tested at the same time that don't necessarily have the same signal arrival time. Each individual peak can have an offset with the ones of other detectors. The inconsistency in the first 100 ns of both time distributions is an artefact of the TDCs and are systematically rejected during the analysis.

1433 The ROOT files include a TTree called RunParameters containing, among other things, the in-
 1434 formation related to the type of run. The run type can then be accessed as described by Source
 1435 Code B.12 and the function IsEfficiencyRun() is then used to determine if the run file is an effi-
 1436 ciency run or, on the contrary, another type of run (noise or gamma measurement).

```

1437
1438     TTree* RunParameters = (TTree*)dataFile.Get("RunParameters");
1439     TString* RunType = new TString();
1440     RunParameters->SetBranchAddress("RunType", &RunType);
1441     RunParameters->GetEntry(0);

```

1439 *Source Code B.12: Access to the run type contained in TTree* RunParameters.*

1440 Finally, the data files will have a slightly different content whether it was collected before or after
1441 October 2017 and the upgrade of the DAQ software that brought a new information into the ROOT
1442 output. This is discussed in Appendix A.4.3 and implies that the analysis will differ a little depending
1443 on the data format. Indeed, as no information on the data quality is stored, in older data files, the cor-
1444 rections for missing events has to be done at the end of the analysis. The information about the type
1445 of data format is stored in the variable **bool** `isNewFormat` by checking the list of branches contained
1446 in the data tree via the methods `TTree::GetListOfBranches()` and `TCollection::Contains()`.

1447 **B.6.2 Beam time window calculation for efficiency runs**

1448 Knowing the run type is important first of all to know the width of the acquisition window to be used
1449 for the rate calculation and finally to be able to seek for muons. Indeed, the peak that appears in the
1450 time distribution for each detectors is then fitted to extract the most probable time window in which
1451 the tool should look for muon hits. The data outside of this time window is then used to evaluate the
1452 noise or gamma background the detector was subjected to during the data taking. Computing the
1453 position of the peak is done calling the function `SetBeamWindow()` defined in file `src/RPCHit.cc` that
1454 loops a first time on the data. The data is first sorted in a 3D array of 1D histograms (`GIFH1Array`, see
1455 `include/types.h`). Then the location of the highest bin is determined using `TH1::GetMaximumBin()`
1456 and is used to define a window in which a gaussian fit will be applied to compute the peak width.
1457 This window is a 80 ns defined by Formula B.1 around the central bin.

$$t_{center}(ns) = bin \times width_{bin}(ns) \quad (\text{B.1a})$$

$$[t_{low}; t_{high}] = [t_{center} - 40; t_{center} + 40] \quad (\text{B.1b})$$

1458 Before the fit is performed, the average number of noise/gamma hits per bin is evaluated using
1459 the data outside of the fit window. Excluding the first 100 ns, the average number of hits per bin
1460 due to the noise or gamma is defined by Formula B.2 after extracting the amount of hits in the time
1461 windows $[100; t_{low}]$ and $[t_{high}; 600]$ thanks to the method `TH1::Integral()`. This average number
1462 of hits is then subtracted to every bin of the 1D histogram, in order to *clean* it from the noise or
1463 gamma contribution as much as possible to improve the fit quality. Bins where $\langle n_{hits} \rangle$ is greater
1464 than the actual bin content are set to 0.

$$\Delta t_{noise}(ns) = 600 \overbrace{-t_{high} + t_{low}}^{-80ns} - 100 = 420ns \quad (\text{B.2a})$$

$$\langle n_{hits} \rangle = width_{bin}(ns) \times \frac{\sum_{t=100}^{t_{low}} + \sum_{t=t_{high}}^{600}}{\Delta t_{noise}(ns)} \quad (\text{B.2b})$$

1465 Finally, the fit parameters are extracted and saved for each detector in 3D arrays of **float**
1466 (`muonPeak`, see `include/types.h`), a first one for the mean arrival time of the muons, `PeakTime`,

1467 and a second one for the width of the peak, `PeakWidth`. The width is defined as 6σ of the gaussian
 1468 fit. The same settings are applied to every partitions of the same detector. To determine which one
 1469 of the detector's partitions is directly illuminated by the beam, the peak height of each partition is
 1470 compared and the highest one is then used to define the peak settings.

1471 **B.6.3 Data loop and histogram filling**

1472 3D arrays of histogram are created to store the data and display it on the DQM of GIF++ webDCS
 1473 for the use of shifters. These histograms, presented in section B.2.1.1, are filled while looping on
 1474 the data. Before starting the analysis loop, it is necessary to control the entry quality for the new
 1475 file formats featuring `QFlag`. If the `QFlag` value for this entry shows that 1 TDC or more have a
 1476 `CORRUPTED` flag, then this event is discarded. The loss of statistics is low enough to be neglected.
 1477 `QFlag` is controled using the function `IsCorruptedEvent()` defined in `src/utils.cc`. As explained
 1478 in Appendix A.4.3, each digit of this integer represent a TDC flag that can be 1 or 2. Each 2 is
 1479 the sign of a `CORRUPTED` state. Then, the data is accessed entry by entry in the ROOT `TTree` using
 1480 `RAWData` and each hit in the hit list is assigned to a detector channel and saved in the corresponding
 1481 histograms. In the first part of the analysis, in which the loop over the ROOT file's content is
 1482 performed, the different steps are:

1483 **1- RPC channel assignment and control:** a check is done on the RPC channel extracted thanks
 1484 to the mapping via the method `Mapping::GetLink()`. If the channel is not initialised and is 0, or if
 1485 the TDC channel was greater than 5127, the hit is discarded. This means there was a problem in the
 1486 mapping. Often a mapping problem leads to the crash of the offline tool.

1487 **2- Creation of a `RPCHit` object:** to easily get the trolley, slot and partition in which the hit has
 1488 been assigned, this object is particularly helpful.

1489 **3- General histograms are filled:** the hit is filled into the time distribution and the general hit
 1490 distribution histograms, and if the arrival time is within the first 100 ns, it is discarded and nothing
 1491 else happens and the loop proceeds with the next hit in the list.

1492 **4- Multiplicity counter:** the hit multiplicity counter of the corresponding detectors incremented.

1493 **5-a- Efficiency runs - Is the hit within the peak window? :** if the peak is contained in the peak
 1494 window previously defined in section B.6.2, the hit is filled into the beam hit profile histogram of
 1495 the corresponding chamber, added into the list of muon hits and increments the counter of *in time*
 1496 hits. The term *in time* here refers to the hits that are likely to be muons by arriving in the expected
 1497 time window. If the hit is outside of the peak window, it is filled into the noise profile histogram
 1498 of the corresponding detector, added into the list of noise/gamma hits and increments the counter of
 1499 noise/gamma hits.

1500 **5-b- Noise/gamma rate runs - Noise histograms are filled:** the hit is filled into the noise profile
 1501 histogram of the corresponding detector, added into the list of noise/gamma hits and increments the
 1502 counter of noise/gamma hits.

1504 After the loop on the hit list of the entry is over, the next step is to clusterize the 3D lists filled
 1505 in the previous steps. A 3D loop is then started over the active trolley, slot and RPC partitions to
 1506 access these objects. Each `NoiseHitList` and `MuonHitList`, in case of efficiency run, are clusterized
 1507 as described in section B.5.2. There corresponding cluster size and multiplicity histograms are filled
 1508 at the end of the clustering process. Then, the efficiency histogram is filled in case of efficiency run.
 1509 The selection is simply made by checking whether the RPC detected signals in the peak window
 1510 during this event. Nevertheless, it is useful to highlight that at this level, it is not possible yet to
 1511 discriminate in between a muon hit and noise or gamma hit. Thus, `MuonCSize_H`, `MuonCMult_H`
 1512 and `Efficiency0_H` are subjected to noise and gamma contamination. This contamination will be
 1513 estimated and corrected at the moment the results will be written into output CSV files. Finally, the
 1514 loop ends on the filling of the general hit multiplicity histogram.

1515 **B.6.4 Results calculation**

1516 As mentioned in section B.2.1, the analysis of DAQ data provides the user with 3 CSV files and
 1517 a ROOT file associated to each and every ROOT data file. The fourth CSV file is provided by the
 1518 extraction of the CEAN main frame data monitored during data taking and will be discussed later.
 1519 After looping on the data in the previous part of the analysis macro, the output files are created and a
 1520 3D loop on each RPC readout partitions is started to extract the histograms parameters and compute
 1521 the final results.

1522

1523 **B.6.4.1 Rate normalisation**

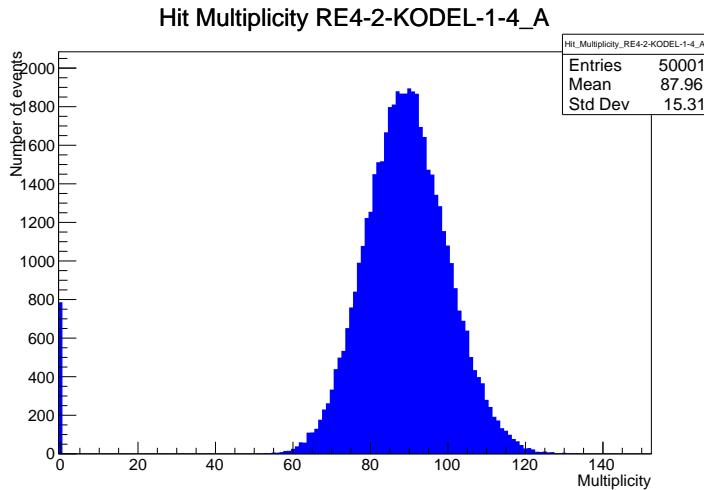


Figure B.2: The effect of the quality flag is explained by presenting the reconstructed hit multiplicity of a data file without `Quality_flag`. The artificial high content of bin 0 is the effect of corrupted data.

1524 To analyse old data format files, not containing any quality flag, it is needed to estimate the amount
 1525 of corrupted data via a fit as the corrupted data will always fill events with a fake "0 multiplicity".
 1526 Indeed, as no hits were stored in the DAQ ROOT files, these events artificially contribute to fill
 1527 the bin corresponding to a null multiplicity, as shown in Figure B.2. In the case the mean of the

hit multiplicity distribution is high, the contribution of the corrupted data can easily be evaluated for later correction by comparing the level of the bin at multiplicity 0 and of a skew fit curve that should indicate a value consistent with 0. A skew fit has been chosen over a Poisson fit as it was giving better results for lower mean multiplicity values. Nevertheless, for low irradiation cases, as explained in Appendix A.4.3, the hit multiplicity distribution mean is, on the contrary, rather small and the probability to record events without hits can't be considered small anymore, leading to a difficult and non-reliable estimation of the corruption. As can be seen in Source Code B.13, conditions have been applied to prevent bad fits and wrong corruption estimation in cases where :

- The difference in between the data for multiplicity 1 and the corresponding fit value should be lower than 1% of the total amount of data : $\frac{|n_{m=1} - sk(1)|}{N_{tot}} < 0.01$ where $n_{m=1}$ is the number of entries with multiplicity 1, $sk(1)$ the value of the skew fit, as defined by Formula 4.3, for multiplicity 1 and N_{tot} the total number of entries.

- The amount of data contained in the multiplicity 0 bin should not exceed 40% : $\frac{n_{m=0}}{N_{tot}} \leq 0.4$ where $n_{m=0}$ is the number of entries with multiplicity 0. This number has been determined to be the maximum to be able to separate the excess of data due to corruption from the hit multiplicity distribution.

Those 2 conditions need to be fulfilled to estimate the corruption of old data format files. If the fit was successful, the level of corruption is written in `Offline-Corrupted.csv` and the number of corrupted entries, refered as the integer `nEmptyEvent`, is subtracted from the total number of entries when the rate normalisation factor is computed as explicit in Source Code B.13. Note that for new data format files, the number of corrupted entries being set to 0, the definition of `rate_norm` stays valid.

```

1550
    if(!isNewFormat){
        TF1* GaussFit = new TF1("gaussfit","[0]*exp(-0.5*((x-[1])/[2])**2)",0,Xmax);
        GaussFit->SetParameter(0,100);
        GaussFit->SetParameter(1,10);
        GaussFit->SetParameter(2,1);
        HitMultiplicity_H.rpc[T][S][p]->Fit(GaussFit,"LIQR","",0.5,Xmax);

        TF1* SkewFit = new TF1("skewfit","[0]*exp(-0.5*((x-[1])/[2])**2) / (1 +
→      exp(-[3]*(x-[4])))",0,Xmax);
        SkewFit->SetParameter(0,GaussFit->GetParameter(0));
        SkewFit->SetParameter(1,GaussFit->GetParameter(1));
        SkewFit->SetParameter(2,GaussFit->GetParameter(2));
        SkewFit->SetParameter(3,1);
        SkewFit->SetParameter(4,1);
        HitMultiplicity_H.rpc[T][S][p]->Fit(SkewFit,"LIQR","",0.5,Xmax);

        double fitValue = SkewFit->Eval(1,0,0,0);
        double dataValue = (double)HitMultiplicity_H.rpc[T][S][p]->GetBinContent(2);
        double difference = TMath::Abs(dataValue - fitValue);
        double fitTOdataVSentries_ratio = difference / (double)nEntries;
        bool isFitGOOD = fitTOdataVSentries_ratio < 0.01;

1551
        double nSinglehit = (double)HitMultiplicity_H.rpc[T][S][p]->GetBinContent(1);
        double lowMultRatio = nSinglehit / (double)nEntries;
        bool isMultLOW = lowMultRatio > 0.4;

        if(isFitGOOD && !isMultLOW){
            nEmptyEvent = HitMultiplicity_H.rpc[T][S][p]->GetBinContent(1);
            nPhysics = (int)SkewFit->Eval(0,0,0,0);
            if(nPhysics < nEmptyEvent)
                nEmptyEvent = nEmptyEvent-nPhysics;
        }
    }

    double corrupt_ratio = 100.*(double)nEmptyEvent / (double)nEntries;
    outputCorrCSV << corrupt_ratio << '\t';

    float rate_norm = 0.;
    float stripArea = GIFIInfra->GetStripGeo(tr,sl,p);

    if(IsEfficiencyRun(RunType)){
        float noiseWindow = BMTDCWINDOW - TIMEREJECT - 2*PeakWidth.rpc[T][S][p];
        rate_norm = (nEntries-nEmptyEvent)*noiseWindow*1e-9*stripArea;
    } else
        rate_norm = (nEntries-nEmptyEvent)*RDMNOISEWDW*1e-9*stripArea;

```

Source Code B.13: Definition of the rate normalisation variable. It takes into account the number of non corrupted entries and the time window used for noise calculation, to estimate the total integrated time, and the strip active area to express the result as rate per unit area.

1553 B.6.4.2 Rate and activity

1554 At this point, the strip rate histograms, StripNoiseProfile_H.rpc[T][S][p], only contain an in-
 1555 formation about the total number of noise or rate hits each channel received during the data taking.
 1556 As described in Source Code B.14, a loop on the strip channels will be used to normalise the content
 1557 of the rate distribution histogram for each detector partitions. The initial number of hits recorded for
 1558 a given bin will be extracted and 2 values will be computed:

- 1559 ● the strip rate, defined as the number of hits recorded in the bin normalised like described in
 1560 the previous section, using the variable `rate_norm`, and

- 1561 ● the strip activity, defined as the number of hits recorded in the bin normalised to the average
 1562 number of hits per bin contained in the partition histogram, using the variable `averageNhit`.
 1563 This value provides an information on the homogeneity of the detector response to the gamma
 1564 background or of the detector noise. An activity of 1 corresponds to an average response.
 1565 Above 1, the channel is more active than the average and bellow 1, the channel is less active.

```

int nNoise = StripNoiseProfile_H.rpc[T][S][p]->GetEntries();
float averageNhit = (nNoise>0) ? (float)(nNoise/nStripsPart) : 1.;

for(Uint st = 1; st <= nStripsPart; st++) {
    float stripRate =
        StripNoiseProfile_H.rpc[T][S][p]->GetBinContent(st)/rate_norm;
    float stripAct =
        StripNoiseProfile_H.rpc[T][S][p]->GetBinContent(st)/averageNhit;

    StripNoiseProfile_H.rpc[T][S][p]->SetBinContent(st,stripRate);
    StripActivity_H.rpc[T][S][p]->SetBinContent(st,stripAct);
}

```

1567 *Source Code B.14: Description of the loop that allows to set the content of each strip rate and strip activity
 channel for each detector partition.*

1568 On each detector partitions, which are readout by a single FEE, all the channels are not processed
 1569 by the same chip. Each chip can give a different noise response and thus, histograms using a chip
 1570 binning are used to investigate chip related noise behaviours. The average values of the strip rate
 1571 or activity grouped into a given chip are extracted using the using the function `GetChipBin()` and
 1572 stored in dedicated histograms as described in Source Codes B.15 and B.16 respectively.

```

float GetChipBin(TH1* H, Uint chip){
    Uint start = 1 + chip*NSTRIPSCHIP;
    int nActive = NSTRIPSCHIP;
    float mean = 0.;

    for(Uint b = start; b <= (chip+1)*NSTRIPSCHIP; b++) {
        float value = H->GetBinContent(b);
        mean += value;
        if(value == 0.) nActive--;
    }

    if(nActive != 0) mean /= (float)nActive;
    else mean = 0.;

    return mean;
}

```

1575 *Source Code B.15: Function used to compute the content of a bin for an histogram using chip binning.*

```

1576   for(UInt ch = 0; ch < (nStripsPart/NSTRIPSCHIP); ch++) {
1577     ChipMeanNoiseProf_H.rpc[T][S][p]->
1578       SetBinContent(ch+1,GetChipBin(StripNoiseProfile_H.rpc[T][S][p],ch));
1579     ChipActivity_H.rpc[T][S][p]->
1580       SetBinContent(ch+1,GetChipBin(StripActivity_H.rpc[T][S][p],ch));
1581   }

```

Source Code B.16: Description of the loop that allows to set the content of each chip rate and chip activity bins for each detector partition knowing the information contained in the corresponding strip distribution histograms.

The activity variable is used to evaluate the homogeneity of the detector response to background or of the detector noise. The homogeneity h_p of each detector partition can be evaluated using the formula $h_p = \exp(-\sigma_p^R / \langle R \rangle_p)$, where $\langle R \rangle_p$ is the partition mean rate and σ_p^R is the rate standard deviation calculated over the partition channels. The more homogeneously the rates are distributed and the smaller will σ_p^R be, and the closer to 1 will h_p get. On the contrary, if the standard deviation of the channel's rates is large, h_p will rapidly get to 0. This value is saved into histograms as shown in Source Code B.17 and could in the future be used to monitor through time, once extracted, the evolution of every partition homogeneity. This could be of great help to understand the apparition of eventual hot spots due to ageing of the chambers subjected to high radiation levels. The monitored homogeneity information could then be combined with a monitoring of the activity of each individual channel in order to have a finer information. Monitoring tools have been suggested and need to be developed for this purpose.

```

1590   float MeanPartSDev = GetTH1StdDev(StripNoiseProfile_H.rpc[T][S][p]);
1591   float strip_homog = (MeanPartRate==0)
1592     ? 0.
1593     : exp(-MeanPartSDev/MeanPartRate);
1594   StripHomogeneity_H.rpc[T][S][p]->Fill("exp -#left(#frac{\#sigma_{Strip}
1595     \rightarrow Rate}{\#mu_{Strip Rate}}\#right)",strip_homog);
1596   StripHomogeneity_H.rpc[T][S][p]->GetYaxis()->SetRangeUser(0.,1.);
1597
1598   float ChipStDevMean = GetTH1StdDev(ChipMeanNoiseProf_H.rpc[T][S][p]);
1599   float chip_homog = (MeanPartRate==0)
1600     ? 0.
1601     : exp(-ChipStDevMean/MeanPartRate);
1602   ChipHomogeneity_H.rpc[T][S][p]->Fill("exp -#left(#frac{\#sigma_{Chip}
1603     \rightarrow Rate}{\#mu_{Chip Rate}}\#right)",chip_homog);
1604   ChipHomogeneity_H.rpc[T][S][p]->GetYaxis()->SetRangeUser(0.,1.);

```

Source Code B.17: Storage of the homogeneity into dedicated histograms.

B.6.4.3 Strip masking tool

The offline tool is automatically called at the end of each data taking to analyse the data and offer the shifter DQM histograms to control the data quality. After the histograms have been published online in the DQM page, the shifter can decide to mask noisy or dead channels that will contribute to bias the final rate calculation by editing the mask column of `ChannelsMapping.csv` as can be seen in Figure B.3.

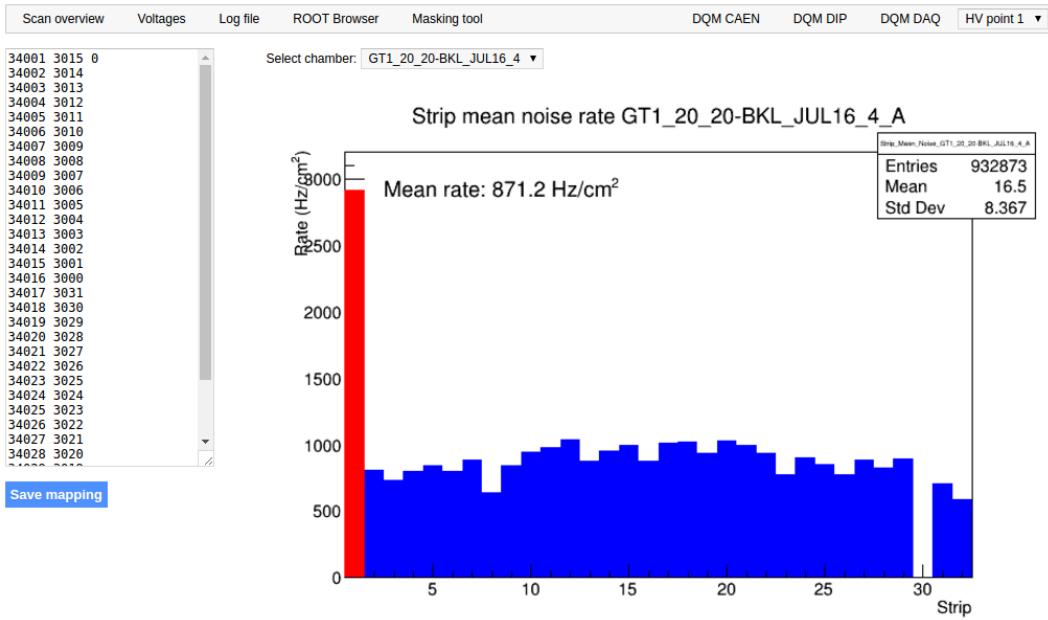


Figure B.3: Display of the masking tool page on the webDCS. The window on the left allows the shifter to edit ChannelsMapping.csv. To mask a channel, it only is needed to set the 3rd field corresponding to the strip to mask to 0. It is not necessary for older mapping file formats to add a 1 for each strip that is not masked as the code is versatile and the default behaviour is to consider missing mask fields as active strips. The effect of the mask is directly visible for noisy channels as the corresponding bin turns red. The global effect of masking strips will be an update of the rate value showed on the histogram that will take into consideration the rejected channels.

1599 From the code point of view, the function `GetTH1Mean()` is used to retrieve the mean rate par-
 1600 tition by partition after the rates have been calculated strip by strip and filled into the histograms
 1601 `StripNoiseProfile_H.rpc[T][S][p]`, as described through Source Code B.18.

1602 Once the mask for each rejected channel has been updated, the shifter can manually run the of-
 1603 fline tool again to update the DQM plots, now including the masked strips, as well the rate results
 1604 written in the output CSV file `Offline-Rate.csv`. If not done during the shifts, the strip masking
 1605 procedure needs to be carefully done by the person in charge of data analysis on the scans that were
 1606 selected to produce the final results.

```

1607
1608     float GetTH1Mean(TH1* H) {
1609         int nBins = H->GetNbinsX();
1610         int nActive = nBins;
1611         float mean = 0.;
1612
1613         for(int b = 1; b <= nBins; b++) {
1614             float value = H->GetBinContent(b);
1615             mean += value;
1616             if(value == 0.) nActive--;
1617         }
1618
1619         if(nActive != 0) mean /= (float)nActive;
1620         else mean = 0.;
1621
1622         return mean;
1623     }

```

Source Code B.18: The function `GetTH1Mean()` is used to return the mean along the y-axis of `TH1` histograms containing rate information. In order to take into account masked strips whose rate is set to 0, the function looks for masked channels and decrement the number of active channels for each null value found.

1610 B.6.4.4 Output CSV files filling

1611 All the histograms have been filled. Parameters will then be extracted from them to compute the
1612 final results that will later be used to produce plots. Once the results have been computed, the very
1613 last step of the offline macro is to write these values into the corresponding CSV outputs. Aside of
1614 the file `Offline-Corrupted.csv`, 2 CSV files are being written by the macro `offlineAnalysis()`,
1615 `Offline-Rates.csv` and `Offline-L0-EffCl.csv` that respectively contain information about noise
1616 or gamma rates, cluster size and multiplicity, and about level 0 reconstruction of the detector effi-
1617 ciency, muon cluster size and multiplicity. Details on the computation and file writing are respec-
1618 tively given in Sources Codes B.19 and B.20.

1619 **Noise/gamma background variables** are computed and written in the output file for each detector
1620 partitions. A detector average of the hit and cluster rate is also provided, as shown through Sources
1621 Code B.19. The variables that are written for each partition are:

- 1622 • The mean partition hit rate per unit area, `MeanPartRate`, that is extracted from the histogram
1623 `StripNoiseProfile_H` as the mean value along the y-axis, as described in section B.6.4.3. No
1624 error is recorded for the hit rate as this is considered a single measurement. No statistical error
1625 can be associated to it and the systematics are unknown.
- 1626 • The mean cluster size, `cSizePart`, is extracted from the histogram `NoiseCSize_H` and it's
1627 statistical error, `cSizePartErr`, is taken to be 2σ of the total distribution.
- 1628 • The mean cluster multiplicity per trigger, `cMultPart`, is extracted from the histogram `NoiseCMult_H`
1629 and it's statistical error, `cMultPartErr`, is taken to be 2σ of the total distribution. It is impor-
1630 tant to point to the fact that this variable gives an information that is dependent on the buffer
1631 window width used for each trigger for the calculation.
- 1632 • The mean cluster rate per unit area, `ClustPartRate`, is defined as the mean hit rate normalised

1633 to the mean cluster size and it's statistical error, `ClustPartRateErr`, is then obtained using the
 1634 relative statistical error on the mean cluster size.

```

for (UInt tr = 0; tr < GIFInfra->GetNTrolleys(); tr++) {
    UInt T = GIFInfra->GetTrolleyID(tr);

    for (UInt sl = 0; sl < GIFInfra->GetNSlots(tr); sl++) {
        UInt S = GIFInfra->GetSlotID(tr,sl) - 1;

        float MeanNoiseRate = 0.;
        float ClusterRate = 0.;
        float ClusterSDev = 0.;

        for (UInt p = 0; p < GIFInfra->GetNPartitions(tr,sl); p++) {
            float MeanPartRate = GetTH1Mean(StripNoiseProfile_H.rpc[T][S][p]);
            float cSizePart = NoiseCSIZE_H.rpc[T][S][p]->GetMean();
            float cSizePartErr = (NoiseCSIZE_H.rpc[T][S][p]->GetEntries()==0)
                ? 0.
                : 2*NoiseCSIZE_H.rpc[T][S][p]->GetStdDev() /
                    sqrt(NoiseCSIZE_H.rpc[T][S][p]->GetEntries());
            float cMultPart = NoiseCMult_H.rpc[T][S][p]->GetMean();
            float cMultPartErr = (NoiseCMult_H.rpc[T][S][p]->GetEntries()==0)
                ? 0.
                : 2*NoiseCMult_H.rpc[T][S][p]->GetStdDev() /
                    sqrt(NoiseCMult_H.rpc[T][S][p]->GetEntries());
            float ClustPartRate = (cSizePart==0) ? 0.
                : MeanPartRate/cSizePart;
            float ClustPartRateErr = (cSizePart==0) ? 0.
                : ClustPartRate * cSizePartErr/cSizePart;

            outputRateCSV << MeanPartRate << '\t'
                << cSizePart << '\t' << cSizePartErr << '\t'
                << cMultPart << '\t' << cMultPartErr << '\t'
                << ClustPartRate << '\t' << ClustPartRateErr << '\t';

            RPCarea += stripArea * nStripsPart;
            MeanNoiseRate += MeanPartRate * stripArea * nStripsPart;
            ClusterRate += ClustPartRate * stripArea * nStripsPart;
            ClusterSDev += (cSizePart==0)
                ? 0.
                : ClusterRate*cSizePartErr/cSizePart;
        }

        MeanNoiseRate /= RPCarea;
        ClusterRate /= RPCarea;
        ClusterSDev /= RPCarea;

        outputRateCSV << MeanNoiseRate << '\t'
            << ClusterRate << '\t' << ClusterSDev << '\t';
    }
}

```

1636 *Source Code B.19: Description of rate result calculation and writing into the CSV output Offline-Rate.csv.
 Are saved into the file for each detector, the mean partition rate, cluster size and cluster mutiplicity, along with
 their errors, for each partition and as well as a detector average.*

1637 **Muon performance variables** are computed and written in the output file for each detector parti-
 1638 tions as shown through Sources Code B.20. The variables that are written for each partition are:

- 1639 ● The muon efficiency, `eff`, extracted from the histogram `Efficiency0_H`. It is reminded that
1640 this offline tool doesn't include any tracking algorithm to identify muons from the beam and
1641 only relies on the hits arriving in the time window corresponding to the beam time. The con-
1642 tent of the efficiency histogram is thus biased by the noise/gamma background contribution
1643 into this window and is thus corrected by estimating the muon data content in the peak re-
1644 gion knowing the noise/gamma content in the rate calculation region. Both time windows
1645 being different, the choice was made to normalise the noise/gamma background calculation
1646 window to it's equivalent beam window in order to have comparable values using the variable
1647 `windowRatio`. Finally, to estimate the data ratio in the peak region, the variable `DataRatio`
1648 is defined as the ratio in between the estimated mean cluster multiplicity of the muons in the
1649 peak region, `MuonCM`, and of the total mean cluster multiplicity in the peak region, `PeakCM`.
1650 `MuonCM` is itself defined as the difference in between the total mean cluster multiplicity in the
1651 peak region and the normalised mean noise/gamma cluster multiplicity calculated outside of
1652 the peak region. The statistical error related to the efficiency, `eff_err`, is computed using a
1653 binomial distribution, as the efficiency measure the probability of "success" and "failure" to
1654 detect muons.
- 1655 ● The mean muon cluster size, `MuonCS`, is calculated using the total mean cluster size and multi-
1656 plicity in the peak region, respectively extracted from histograms `MuonCSize_H` and `MuonCMult_H`,
1657 the noise/gamma background mean cluster size and normalised multiplicity, extracted from
1658 `NoiseCSize_H` and `NoiseCMult_H`, and of the estimated muon cluster multiplicity `MuonCM` pre-
1659 viously explicated. The associated statistical error, `MuonCM_err`, is calculated using the propa-
1660 gation of errors of the mentioned variables.
- 1661 ● The mean muon cluster multiplicity in the peak region, `MuonCM`, explicated above whose sta-
1662 tistical error, `MuonCM_err`, is the sum of statistical error associated to the total mean clus-
1663 ter multiplicity in the peak reagion, `PeakCM_err`, and of the mean noise/gamma cluster size,
1664 `NoiseCM_err`.

1665 In addition to these 2 CSV files, the histograms are saved in ROOT file `Scan00XXXX_HVY_Offline.root`
1666 as explained in section B.2.1.1.

1667

```

for (UInt tr = 0; tr < GIFInfra->GetNTrolleys(); tr++) {
    UInt T = GIFInfra->GetTrolleyID(tr);
    for (UInt sl = 0; sl < GIFInfra->GetNSlots(tr); sl++) {
        UInt S = GIFInfra->GetSlotID(tr,sl) - 1;
        for (UInt p = 0; p < GIFInfra->GetNPartitions(tr,sl); p++) {
            float noiseWindow =
                BMTDCWINDOW - TIMEREJECT - 2*PeakWidth.rpc[T][S][p];
            float peakWindow = 2*PeakWidth.rpc[T][S][p];
            float windowRatio = peakWindow/noiseWindow;

            float PeakCM = MuonCMult_H.rpc[T][S][p]->GetMean();
            float PeakCS = MuonCSize_H.rpc[T][S][p]->GetMean();
            float NoiseCM = NoiseCMult_H.rpc[T][S][p]->GetMean() *windowRatio;
            float NoiseCS = NoiseCSize_H.rpc[T][S][p]->GetMean();
            float MuonCM = (PeakCM<NoiseCM) ? 0. : PeakCM-NoiseCM;
            float MuonCS = (MuonCM==0 || PeakCM*PeakCS<NoiseCM*NoiseCS)
                ? 0.
                : (PeakCM*PeakCS-NoiseCM*NoiseCS) / MuonCM;
            float PeakCM_err = (MuonCMult_H.rpc[T][S][p]->GetEntries()==0.)
                ? 0.
                : 2*MuonCMult_H.rpc[T][S][p]->GetStdDev() /
                    sqrt(MuonCMult_H.rpc[T][S][p]->GetEntries());
            float PeakCS_err = (MuonCSize_H.rpc[T][S][p]->GetEntries()==0.)
                ? 0.
                : 2*MuonCSize_H.rpc[T][S][p]->GetStdDev() /
                    sqrt(MuonCSize_H.rpc[T][S][p]->GetEntries());
            float NoiseCM_err = (NoiseCMult_H.rpc[T][S][p]->GetEntries()==0.)
                ? 0.
                : windowRatio*2*NoiseCMult_H.rpc[T][S][p]->GetStdDev() /
                    sqrt(NoiseCMult_H.rpc[T][S][p]->GetEntries());
            float NoiseCS_err = (NoiseCSize_H.rpc[T][S][p]->GetEntries()==0.)
                ? 0.
                : 2*NoiseCSize_H.rpc[T][S][p]->GetStdDev() /
                    sqrt(NoiseCSize_H.rpc[T][S][p]->GetEntries());
            float MuonCM_err = (MuonCM==0) ? 0. : PeakCM_err+NoiseCM_err;
            float MuonCS_err = (MuonCS==0 || MuonCM==0) ? 0.
                : (PeakCS*PeakCM_err + PeakCM*PeakCS_err +
                    NoiseCS*NoiseCM_err + NoiseCM*NoiseCS_err +
                    MuonCS*MuonCM_err) / MuonCM;

            float DataRatio = MuonCM/PeakCM;
            float DataRatio_err = (MuonCM==0) ? 0.
                : DataRatio*(MuonCM_err/MuonCM + PeakCM_err/PeakCM);
            float eff = DataRatio*Efficiency0_H.rpc[T][S][p]->GetMean();
            float eff_err = DataRatio*2*Efficiency0_H.rpc[T][S][p]->GetStdDev() /
                sqrt(Efficiency0_H.rpc[T][S][p]->GetEntries()) +
                Efficiency0_H.rpc[T][S][p]->GetMean()*DataRatio_err;

            outputEffCSV << eff << '\t' << eff_err << '\t'
                << MuonCS << '\t' << MuonCS_err << '\t'
                << MuonCM << '\t' << MuonCM_err << '\t';
        }
    }
}

```

1668

Source Code B.20: Description of efficiency result calculation and writing into the CSV output Offline-L0-EffCl.csv. Are saved into the file for each detector, the efficiency, corrected taking into account the background in the peak window of the time profile, muon cluster size and muon cluster multiplicity, along with their errors, for each partition and as well as a detector average.

1669

1670 B.7 Current data Analysis

1671 Detectors under test at GIF++ are connected both to a CAEN HV power supply and to a CAEN
1672 ADC that reads the currents inside of the RPC gaps bypassing the supply cable. During data tak-
1673 ing, the webDCS records into a ROOT file called `Scan00XXXX_HVY_CAEN.root` histograms with the
1674 monitored parameters of both CAEN devices. Are recorded for each RPC channels (in most cases,
1675 a channel corresponds to an RPC gap):

- 1676 • the effective voltage, HV_{eff} , set by the webDCS using the PT correction on the CAEN power
1677 supply,
- 1678 • the applied voltage, HV_{app} , monitored by the CAEN power supply, and the statistical error
1679 related to the variations of this value through time to follow the variation of the environmental
1680 parameters defined as the RMS of the histogram divided by the square root of the number of
1681 recorded points,
- 1682 • the monitored current, I_{mon} , monitored by the CAEN power supply, and the statistical error
1683 related to the variations of this value through time to follow the variation of the environmental
1684 parameters defined as the RMS of the histogram divided by the square root of the number of
1685 recorded points,
- 1686 • the corresponding current density, J_{mon} , defined as the monitored current per unit area,
1687 $J_{mon} = I_{mon}/A$, where A is the active area of the corresponding gap,
- 1688 • the ADC current, I_{ADC} , recorded through the CAEN ADC module that monitors the dark
1689 current in the gap itself. First of all, the resolution of such a module is better than that of
1690 CAEN power supplies and moreover, the current is not read-out through the HV supply line
1691 but directly at the chamber level giving the real current inside of the detector. The statistical
1692 error is defined as the RMS of the histogram distribution divided by the square root of the
1693 number of recorded points.

1694 Once extracted through a loop over the element of GIF++ infrastructure via the C++ macro
1695 `GetCurrent()`, these parameters, organised in 9 columns per detector HV supply line, are written in
1696 the output CSV file `Offline-Current.csv`. The macro can be found in the file `Current.cc`.

List of Figures

1697

1698	2.1	Absorbed dose in the CMS cavern after an integrated luminosity of 3000 fb. R is the transverse distance from the beamline and Z is the distance along the beamline from the Interaction Point at Z=0.	2-2
1699	2.2	A quadrant of the muon system, showing DTs (yellow), RPCs (light blue), and CSCs (green). The locations of new forward muon detectors for Phase-II are contained within the dashed box and indicated in red for GEM stations (ME0, GE1/1, and GE2/1) and dark blue for improved RPC (iRPC) stations (RE3/1 and RE4/1).	2-3
1700	2.3	RMS of the multiple scattering displacement as a function of muon p_T for the proposed forward muon stations. All of the electromagnetic processes such as bremsstrahlung and magnetic field effect are included in the simulation.	2-4
1701	3.1	Different phases of the avalanche development in the RPC gas volume subjected to a constant electric field E_0 . a) An avalanche is initiated by the primary ionisation caused by the passage of a charged particle through the gas volume. b) Due to its growing size, the avalanche starts to locally influence the electric field. c) The electrons, lighter than the cations reach the anode first. d) The ions reach the cathode. While the charges have not recombined, the electric field in the small region around the avalanche stays affected and locally blind the detector.	3-2
1702	3.2	Effeciency (circles and stars with 30 mV and 100 mV thresholds respectively) and streamer probability (opened circles) as function of the operating voltatge of a 2 mm single gap HPL RPC flushed with a gas mixture containing (a) 5%, (b) 2%, (c) 1% and (d) no SF_6 [13].	3-3
1703	3.3	Movement of the charge carriers in an RPC. Figure 3.3a: Voltage across an RPC whose electrode have a relative permittivity of 5 at the moment the tension s applied. Figure 3.3b: After the charge carriers moved, the electrodes are charged and there is no voltage drop over the electrodes anymore. The full potential is applied on the gas gap only. Figure 3.3c: The streamer discharge initiated by a charged particle transports electrons and cations towards the anode and cathode respectively.	3-5
1704	3.4	Typical oscilloscope pulses in streamer mode (Figure 3.4a) and avalanche mode(Figure 3.4b). In the case of streamer mode, the very small avalanche signal is visible.	3-5
1705	3.5	Rate capability comparison for the streamer and avalanche mode of operation. An order of magnitude in rate capability for a maximal efficiency drop of 10% is gained by using the avalanche mode over the streamer mode.	3-6

1730	3.6 Possible double-gap RPC layouts: a) "standard" 1D double-gap RPC, as used in CMS experiment, where the anodes are facing each other and a 1D read-out plane is sandwiched in between them, b) double read-out double-gap RPC as used in ATLAS experiment, where the cathodes are facing each other and 2 read-out planes are used on the outer surfaces. This last layout can offer the possibility to use a 2D reconstruction by using orthogonal read-out planes.	3-7
1731		
1732		
1733		
1734		
1735		
1736	3.7 Comparison of performance of CMS double and single gap RPCs using cosmic muons [24]. Figure 3.7a: Comparison of efficiency sigmoid. Figure 3.7b: Voltage distribution at 95% of maximum efficiency. Figure 3.7c: $\Delta_{10\%}^{90\%}$ distribution.	3-7
1737		
1738		
1739		
1740	3.8 Presentation of ALICE MRPC using 250 μm gas gaps, 620 μm outer glass electrodes and 550 μm inner floating electrodes. More details on the labels are given in [25].	3-8
1741		
1742		
1743	3.9 Particle identification applied to electrons in the STAR experiment. The identification is performed combining ToF and dE/dx measurements [30].	3-9
1744		
1745		
1746	3.10 Comparison of the detector performance of ALICE ToF MRPC [31] at fixed applied voltage (in blue) and at fixed effective voltage (in red). The effective voltage is kept fixed by increasing the applied voltage accordingly to the current drawn by the detector.	3-9
1747		
1748		
1749	3.11 Ratio between total induced and drifting charge have been simulated for single gap, double-gap and multigap layouts [32]. The total induced charge for a double-gap RPC is a factor 2 higher than for a multigap.	3-10
1750		
1751		
1752		
1753	3.12 Charge spectra have been simulated for single gap, double-gap and multigap layouts [32]. It appears that when single gap shows a decreasing spectrum, double and multigap layouts exhibit a spectrum whose peak is detached from the origin. The detachment gets stronger as the number of gaps increases.	3-10
1754		
1755	3.13 The maximal theoretical efficiency is simulated for single gap, double-gap and multigap layouts [32] at a constant gap thickness of 2 mm and using an effective Townsend coefficient of 9 mm^{-1}	3-11
1756		
1757		
1758		
1759		
1760	4.1 Signals from the RPC strips are shaped by the FEE described on Figure 4.1a. Output LVDS signals are then read-out by a TDC module connected to a computer or converted into NIM and sent to scalers. Figure 4.1b describes how these converted signals are put in coincidence with the trigger.	4-3
1761		
1762		
1763		
1764		
1765		
1766		
1767		
1768	4.2 Description of the principle of a CFD. A comparison of threshold triggering (left) and constant fraction triggering (right) is shown in Figure 4.2a. Constant fraction triggering is obtained thanks to zero-crossing technique as explained in Figure 4.2b. The signal arriving at the input of the CFD is split into three components. A first one is delayed and connected to the inverting input of a first comparator. A second component is connected to the noninverting input of this first comparator. A third component is connected to the noninverting input of another comparator along with a threshold value connected to the inverting input. Finally, the output of both comparators is fed through an AND gate.	4-4
1769		
1770		
1771		
1772	4.3 (4.3a) Extrapolation from 2016 data of single hit rate per unit area in the barrel region. (4.3b) Extrapolation from 2016 data of single hit rate per unit area in the endcap region.	4-5
1773		
1774	4.4 Background Fluka simulation compared to 2016 Data at $L = 10^{34} \text{cm}^{-2} \cdot \text{s}^{-1}$ in the fourth endcap disk region. A mismatch in between simulation and data can be observed. [To be understood.]	4-6

LIST OF FIGURES

1775	4.5	Layout of the test beam zone called X5c GIF at CERN. Photons from the radioactive source produce a sustained high rate of random hits over the whole area. The zone is surrounded by 8 m high and 80 cm thick concrete walls. Access is possible through three entry points. Two access doors for personnel and one large gate for material.	4-7
1776		A crane allows installation of heavy equipment in the area.	
1777	4.6	^{137}Cs decays by β^- emission to the ground state of ^{137}Ba (BR = 5.64%) and via the 662 keV isomeric level of ^{137}Ba (BR = 94.36%) whose half-life is 2.55 min.	4-8
1778			
1779	4.7	Floor plan of the GIF++ facility. When the facility downstream of the GIF++ takes electron beam, a beam pipe is installed along the beam line (z-axis). The irradiator can be displaced laterally (its center moves from $x = 0.65$ m to 2.15 m), to increase the distance to the beam pipe.	4-8
1780			
1781	4.8	Simulated unattenuated current of photons in the xz plane (Figure 4.8a) and yz plane (Figure 4.8b) through the source at $x = 0.65$ m and $y = 0$ m. With angular correction filters, the current of 662 keV photons is made uniform in xy planes.	4-9
1782			
1783	4.9	Description of the RPC setup. Dimensions are given in mm. A tent containing RPCs is placed at 1720 mm from the source container. The source is situated in the center of the container. RE-4-2-BARC-161 chamber is 160 mm inside the tent. This way, the distance between the source and the chambers plan is 2060 mm. Figure 4.9a provides a side view of the setup in the xz plane while Figure 4.9b shows a top view in the yz plane.	4-10
1784			
1785	4.10	RE-4-2-BARC-161 chamber is inside the tent as described in Figure 4.9. In the top right, the two scintillators used as trigger can be seen. This trigger system has an inclination of 10° relative to horizontal and is placed above half-partition B2 of the RPCs. PMT electronics are shielded thanks to lead blocks placed in order to protect them without stopping photons from going through the scintillators and the chamber.	4-11
1786			
1787	4.11	Hit distributions over all 3 partitions of RE-4-2-BARC-161 chamber is showed on these plots. Top, middle and bottom figures respectively correspond to partitions A, B, and C. These plots show that some events still occur in other half-partitions than B2, which corresponds to strips 49 to 64, in front of which the trigger is placed, contributing to the inefficiency of detection of cosmic muons. In the case of partitions A and C, the very low amount of data can be interpreted as noise. On the other hand, it is clear that a little portion of muons reach the half-partition B1, corresponding to strips 33 to 48.	4-12
1788			
1789	4.12	Results are derived from data taken on half-partition B2 only. On the 18 th of June 2014, data has been taken on chamber RE-2-BARC-161 at building 904 (Prevessin Site) with cosmic muons providing us a reference efficiency plateau of $(97.54 \pm 0.15)\%$ represented by a black curve. A similar measurement has been done at GIF on the 21 st of July with the same chamber giving a plateau of $(78.52 \pm 0.94)\%$ represented by a red curve.	4-13
1790			
1791	4.13	Representation of the layout used for the simulations of the test setup. The RPC is represented as a yellow trapezoid while the two scintillators as blue cuboids looking at the sky. A green plane corresponds to the muon generation plane within the simulation. Figure 4.9a shows a global view of the simulated setup. Figure 4.9b shows a zommed view that allows to see the 2 scintillators as well as the full RPC plane.	4-14
1792			
1793	4.14	γ flux $F(D)$ is plot using values from table 4.1. As expected, the plot shows similar attenuation behaviours with increasing distance for each absorption factors.	4-17
1794			

1821	4.15	Figure 4.15a shows the linear approximation fit done via formulae 4.7 on data from table 4.2. Figure 4.15b shows a comparison of this model with the simulated flux using a and b given in figure 4.15a in formulae 4.4 and the reference value $D_0 = 50\text{cm}$ and the associated flux for each absorption factor F_0^{ABS} from table 4.1	4-19
1822			
1823			
1824			
1825	4.16	Dose measurements has been done in a plane corresponding to the tents front side. This plan is 1900 mm away from the source. As explained in the first chapter, a lens-shaped lead filter provides a uniform photon flux in the vertical plan orthogonal to the beam direction. If the second line of measured fluxes is not taken into account because of lower values due to experimental equipments in the way between the source and the tent, the uniformity of the flux is well showed by the results.	4-21
1826			
1827			
1828			
1829			
1830			
1831	4.17	4-22
1832	4.18	Evolution of the maximum efficiency for RE2 (4.18a) and RE4 (4.18b) chambers with increasing extrapolated γ rate per unit area at working point. Both irradiated (blue) and non irradiated (red) chambers are shown.	4-24
1833			
1834			
1835	4.19	Evolution of the working point for RE2 (4.19a) and RE4 (4.19b) with increasing extrapolated γ rate per unit area at working point. Both irradiated (blue) and non irradiated (red) chambers are shown.	4-24
1836			
1837			
1838	4.20	Evolution of the maximum efficiency at HL-LHC conditions, i.e. a background hit rate per unit area of 300 Hz/cm ² , with increasing integrated charge for RE2 (4.20a) and RE4 (4.20b) detectors. Both irradiated (blue) and non irradiated (red) chambers are shown. The integrated charge for non irradiated detectors is recorded during test beam periods and stays small with respect to the charge accumulated in irradiated chambers.	4-25
1839			
1840			
1841			
1842			
1843			
1844	4.21	Comparison of the efficiency sigmoid before (triangles) and after (circles) irradiation for RE2 (4.21a) and RE4 (4.21b) detectors. Both irradiated (blue) and non irradiated (red) chambers are shown.	4-25
1845			
1846			
1847	4.22	Evolution of the Bakelite resistivity for RE2 (4.22a) and RE4 (4.22b) detectors. Both irradiated (blue) and non irradiated (red) chambers are shown.	4-26
1848			
1849	4.23	Evolution of the noise rate per unit area for the irradiated chamber RE2-2-BARC-9 only.	4-26
1850			
1851	A.1	(A.1a) View of the front panel of a V1190A TDC module [38]. (A.1b) View of the front panel of a V1718 Bridge module [39]. (A.1c) View of the front panel of a 6U 6021 VME crate [40].	A-3
1852			
1853			
1854	A.2	Module V1190A <i>Trigger Matching Mode</i> timing diagram [38].	A-4
1855	A.3	Structure of the ROOT output file generated by the DAQ. The 5 branches (<code>EventNumber</code> , <code>number_of_hits</code> , <code>Quality_flag</code> , <code>TDC_channel</code> and <code>TDC_TimeStamp</code>) are visible on the left panel of the ROOT browser. On the right panel is visible the histogram corresponding to the variable <code>nHits</code> . In this specific example, there were approximately 50k events recorded to measure the gamma irradiation rate on the detectors. Each event is stored as a single entry in the <code>TTree</code>	A-10
1856			
1857			
1858			
1859			
1860			

LIST OF FIGURES

- 1861 A.4 The effect of the quality flag is explained by presenting the content of `TBranch`
1862 `number_of_hits` of a data file without `Quality_flag` in Figure A.4a and the con-
1863 tent of the same `TBranch` for data corresponding to a `Quality_flag` where all TDCs
1864 were labelled as `GOOD` in Figure A.4b taken with similar conditions. It can be noted
1865 that the number of entries in Figure A.4b is slightly lower then in Figure A.4a due
1866 to the excluded events. A-12
1867 A.5 Using the same data as previously showed in Figure A.4, the effect of the quality
1868 flag is explained by presenting the reconstructed hit multiplicity of a data file with-
1869 out `Quality_flag` in Figure A.5a and the reconstructed content of the same RPC
1870 partition for data corresponding to a `Quality_flag` where all TDCs were labelled as
1871 `GOOD` in Figure A.5b taken with similar conditions. The artificial high content of bin
1872 0 is completely suppressed. A-12
1873 A.6 WebDCS DAQ scan page. On this page, shifters need to choose the type of scan
1874 (Rate, Efficiency or Noise Reference scan), the gamma source configuration at the
1875 moment of data taking, the beam configuration, and the trigger mode. These in-
1876 formation will be stored in the DAQ ROOT output. Are also given the minimal
1877 measurement time and waiting time after ramping up of the detectors is over before
1878 starting the data acquisition. Then, the list of HV points to scan and the number of
1879 triggers for each run of the scan are given in the table underneath. A-14
- 1880 B.1 Example of expected hit time distributions in the cases of efficiency (Figure B.1a)
1881 and noise/gamma rate per unit area (Figure B.1b) measurements as extracted from
1882 the raw ROOT files. The unit along the x-axis corresponds to ns. The fact that
1883 "the" muon peak is not well defined in Figure B.1a is due to the contribution of all
1884 the RPCs being tested at the same time that don't necessarily have the same signal
1885 arrival time. Each individual peak can have an offset with the ones of other detectors.
1886 The inconsistance in the first 100 ns of both time distributions is an artefact of the
1887 TDCs and are systematically rejected during the analysis. B-16
1888 B.2 The effect of the quality flag is explained by presenting the reconstructed hit multi-
1889 plicity of a data file without `Quality_flag`. The artificial high content of bin 0 is the
1890 effect of corrupted data. B-19
1891 B.3 Display of the masking tool page on the webDCS. The window on the left allows the
1892 shifter to edit `ChannelsMapping.csv`. To mask a channel, it only is needed to set the
1893 3rd field corresponding to the strip to mask to 0. It is not necessary for older mapping
1894 file formats to add a 1 for each strip that is not masked as the code is versatile and
1895 the default behaviour is to consider missing mask fields as active strips. The effect
1896 of the mask is directly visible for noisy channels as the corresponding bin turns red.
1897 The global effect of masking strips will be an update of the rate value showed on the
1898 histogram that will take into consideration the rejected channels. B-24

List of Tables

1899

1900	3.1 Properties of the most used electrode materials for RPCs.	3-4
1901	4.1 Total photon flux ($E\gamma \leq 662$ keV) with statistical error predicted considering a	
1902	^{137}Cs activity of 740 GBq at different values of the distance D to the source along	
1903	the x-axis of irradiation field [35].	4-16
1904	4.2 Correction factor c is computed thanks to formulae 4.5 taking as reference $D_0 =$	
1905	50 cm and the associated flux F_0^{ABS} for each absorption factor available in table 4.1. .	4-18
1906	4.3 The data at D_0 in 1997 is taken from [35]. In a second step, using Equations 4.8	
1907	and 4.9, the flux at D can be estimated in 1997. Then, taking into account the	
1908	attenuation of the source activity, the flux at D can be estimated at the time of the	
1909	tests in GIF in 2014. Finally, assuming a sensitivity of the RPC to γ $s = 2 \cdot 10^{-3}$,	
1910	an estimation of the hit rate per unit area is obtained.	4-20
1911	A.1 Inter-process communication cycles in between the webDCS and the DAQ through	
1912	file string signals.	A-19

1913

List of Acronyms

1914

List of Acronyms

1915

1916

1917

A

1918

1919 AFL

Almost Full Level

1920

1921

1922

B

1923

1924 BARC

Bhabha Atomic Research Centre

1925 BLT

Block Transfer

1926 BR

Branching Ratio

1927

1928

1929

C

1930

1931 CAEN

Costruzioni Apparecchiature Elettroniche Nucleari S.p.A.

1932 CERN

European Organization for Nuclear Research

1933 CFD

Constant Fraction Discriminator

1934 CMS

Compact Muon Solenoid

1935 CSC

Cathode Strip Chamber

1936

1937

1938

D

1939

1940 DAQ

Data Acquisition

1941 DCS

Detector Control Software

1942 DQM

Data Quality Monitoring

1943 DT

Drift Tube

1944

1945

1946

F

1947

LIST OF ACRONYMS

1948	FEE	Front-End Electronics
1949	FEB	Front-End Board
1950		
1951		
1952	G	
1953		
1954	GE-/-	Find a good description
1955	GE1/1	Find a good description
1956	GE2/1	Find a good description
1957	GEANT	GEometry ANd Tracking - a series of software toolkit platforms developed by CERN
1958		
1959	GEM	Gas Electron Multiplier
1960	GIF	Gamma Irradiation Facility
1961	GIF++	new Gamma Irradiation Facility
1962		
1963		
1964	H	
1965		
1966	HL-LHC	High Luminosity LHC
1967	HPL	High-pressure laminate
1968	HV	High Voltage
1969		
1970		
1971	I	
1972		
1973	iRPC	improved RPC
1974	IRQ	Interrupt Request
1975		
1976		
1977	L	
1978		
1979	LHC	Large Hadron Collider
1980	LS1	First Long Shutdown
1981	LS3	Third Long Shutdown
1982	LV	Low Voltage
1983	LVDS	Low-Voltage Differential Signaling
1984		
1985		
1986	M	
1987		
1988	MC	Monte Carlo

LIST OF ACRONYMS

1989	MCNP	Monte Carlo N-Particle
1990	ME-/-	Find good description
1991	ME0	Find good description
1992	MRPC	Multigap RPC
1993		
1994		
1995	N	
1996		
1997	NIM	Nuclear Instrumentation Module logic signals
1998		
1999		
2000	P	
2001		
2002	PMT	PhotoMultiplier Tube
2003		
2004		
2005	R	
2006		
2007	RE-/-	Find a good description
2008	RE2/2	Find a good description
2009	RE3/1	Find a good description
2010	RE3/2	Find a good description
2011	RE4/1	Find a good description
2012	RE4/2	Find a good description
2013	RE4/3	Find a good description
2014	RMS	Root Mean Square
2015	ROOT	a framework for data processing born at CERN
2016	RPC	Resistive Plate Chamber
2017		
2018		
2019	S	
2020		
2021	SPS	Super Proton Synchrotron
2022		
2023		
2024	T	
2025		
2026	TDC	Time-to-Digital Converter
2027	ToF	Time-of-flight

LIST OF ACRONYMS

2028

2029

2030

2031

2032 webDCS

Web Detector Control System

W

References

- 2034 [1] CERN. Geneva. LHC Experiments Committee. *The CMS muon project : Technical Design*
2035 *Report*. Tech. rep. CERN-LHCC-97-032. CMS Collaboration, 1997.
- 2036 [2] CERN. Geneva. LHC Experiments Committee. *Technical Proposal for the Phase-II Upgrade*
2037 *of the CMS Detector*. Tech. rep. CERN-LHCC-2015-010. CMS Collaboration, 2015.
- 2038 [3] CERN. Geneva. LHC Experiments Committee. *CMS, the Compact Muon Solenoid : technical*
2039 *proposal*. Tech. rep. CERN-LHCC-94-38. CMS Collaboration, 1994.
- 2040 [4] R. Santonico and R. Cardarelli. “Development of resistive plate counters”. In: *Nucl. Instr.*
2041 *Meth. Phys. Res.* 187 (1981), pp. 377–380.
- 2042 [5] Yu.N. Pestov and G.V. Fedotovich. *A picosecond time-of-flight spectrometer fot eh VEPP-2M*
2043 *based on local-discharge spark counter*. Tech. rep. SLAC-TRANS-0184. SLAC, 1978.
- 2044 [6] W.W. Ash, ed. *Spark Counter With A Localized Discharge*. Vol. SLAC-R-250. 1982, pp. 127–
2045 131.
- 2046 [7] I. Crotty et al. “The non-spark mode and high rate operation of resistive parallel plate cham-
2047 bers”. In: *NIMA* 337 (1993), pp. 370–381.
- 2048 [8] I. Crotty et al. “Further studies of avalanche mode operation of resistive parallel plate cham-
2049 bers”. In: *NIMA* 346 (1994), pp. 107–113.
- 2050 [9] R. Cardarelli et al. “Avalanche and streamer mode operation of resistive plate chambers”. In:
2051 *NIMA* 382 (1996), pp. 470–474.
- 2052 [10] E. Cerron Zeballos et al. “A new type of resistive plate chamber: The multigap RPC”. In:
2053 *NIMA* 374 (1996), pp. 132–135.
- 2054 [11] M.C.S. Williams. “The development of the multigap resistive plate chamber”. In: *Nucl. Phys.*
2055 *B* 61 (1998), pp. 250–257.
- 2056 [12] H. Czyrkowski et al. “New developments on resistive plate chambers for high rate operation”.
2057 In: *NIMA* 419 (1998), pp. 490–496.
- 2058 [13] P. Camarri et al. “Streamer suppression with SF6 in RPCs operated in avalanche mode”. In:
2059 *NIMA* 414 (1998), pp. 317–324.
- 2060 [14] E. Cerron Zeballos et al. “Effect of adding SF6 to the gas mixture in a multigap resistive plate
2061 chamber”. In: *NIMA* 419 (1998), pp. 475–478.
- 2062 [15] CERN. Geneva. LHC Experiments Committee. *ATLAS muon spectrometer: Technical design*
2063 *report*. Tech. rep. CERN-LHCC-97-22. ATLAS Collaboration, 1997.
- 2064 [16] CERN. Geneva. LHC Experiments Committee. *ALICE Time-Of-Flight system (TOF) : Tech-*
2065 *nical Design Report*. Tech. rep. CERN-LHCC-2000-012. ALICE Collaboration, 2000.
- 2066 [17] The CALICE collaboration. “First results of the CALICE SDHCAL technological proto-
2067 type”. In: *JINST* 11 (2016).

- 2068 [18] PoS, ed. *Density Imaging of Volcanoes with Atmospheric Muons using GRPCs*. International
2069 Europhysics Conference on High Energy Physics - HEP 2011. 2011.
- 2070 [19] C. Lippmann. “Detector Physics of Resistive Plate Chambers”. PhD thesis. Johann Wolfgang
2071 Goethe-Universität, 2003.
- 2072 [20] M. Abbrescia et al. “Properties of C₂H₂F₄-based gas mixture for avalanche mode operation
2073 of resistive plate chambers”. In: *NIMA* 398 (1997), pp. 173–179.
- 2074 [21] G. Battistoni et al. “Sensitivity of streamer mode to single ionization electrons”. In: *NIMA*
2075 235 (1985), pp. 91–97.
- 2076 [22] W. Riegler. “Induced signals in resistive plate chambers”. In: *NIMA* 491 (2002), pp. 258–271.
- 2077 [23] E. Cerron Zeballos et al. “A comparison of the wide gap and narrow gap resistive plate
2078 chamber”. In: *NIMA* 373 (1996), pp. 35–42.
- 2079 [24] M. Abbrescia et al. “Cosmic ray tests of double-gap resistive plate chambers for the CMS
2080 experiment”. In: *NIMA* 550 (2005), pp. 116–126.
- 2081 [25] ALICE Collaboration. “A study of the multigap RPC at the gamma irradiation facility at
2082 CERN”. In: *NIMA* 490 (2002), pp. 58–70.
- 2083 [26] B. Bonner et al. “A multigap resistive plate chamber prototype for time-of-flight for the STAR
2084 experiment at RHIC”. In: *NIMA* 478 (2002), pp. 176–179.
- 2085 [27] S. Yang et al. “Test of high time resolution MRPC with different readout modes for the
2086 BESIII upgrade”. In: *NIMA* 763 (2014), pp. 190–196.
- 2087 [28] A. Akindinov et al. “RPC with low-resistive phosphate glass electrodes as a candidate for
2088 the CBM TOF”. In: *NIMA* 572 (2007), pp. 676–681.
- 2089 [29] JINST, ed. *Development of the MRPC for the TOF system of the MultiPurpose Detector*.
2090 RPC2016: XII Workshop on Resistive Plate Chambers and Related Detectors. 2016.
- 2091 [30] M.C.S. Williams. “Particle identification using time of flight”. In: *Journal of Physics G* 39
2092 (2012).
- 2093 [31] A. Alici et al. “Aging and rate effects of the Multigap RPC studied at the Gamma Irradiation
2094 Facility at CERN”. In: *NIMA* 579 (2007), pp. 979–988.
- 2095 [32] M. Abbrescia et al. “The simulation of resistive plate chambers in avalanche mode: charge
2096 spectra and efficiency”. In: *NIMA* 431 (1999), pp. 413–427.
- 2097 [33] M. Abbrescia et al. “Study of long-term performance of CMS RPC under irradiation at the
2098 CERN GIF”. In: *NIMA* 533 (2004), pp. 102–106.
- 2099 [34] H.C. Kim et al. “Quantitative aging study with intense irradiation tests for the CMS forward
2100 RPCs”. In: *NIMA* 602 (2009), pp. 771–774.
- 2101 [35] S. Agosteo et al. “A facility for the test of large-area muon chambers at high rates”. In: *NIMA*
2102 452 (2000), pp. 94–104.
- 2103 [36] PoS, ed. *CERN GIF ++ : A new irradiation facility to test large-area particle detectors for*
2104 *the high-luminosity LHC program*. Vol. TIPP2014. 2014, pp. 102–109.
- 2105 [37] A. Fagot. *GIF++ DAQ v4.0*. 2017. URL: https://github.com/afagot/GIF_DAQ.
- 2106 [38] CAEN. *Mod. V1190-VX1190 A/B, 128/64 Ch Multihit TDC*. 14th ed. 2016.
- 2107 [39] CAEN. *Mod. V1718 VME USB Bridge*. 9th ed. 2009.
- 2108 [40] W-Ie-Ne-R. *VME 6021-23 VXI*. 5th ed. 2016.
- 2109 [41] Wikipedia. *INI file*. 2017. URL: https://en.wikipedia.org/wiki/INI_file.

BIBLIOGRAPHY

- 2110 [42] S. Carrillo A. Fagot. *GIF++ Offline Analysis v6*. 2017. URL: https://github.com/afagot/GIF_OfflineAnalysis.
2111