lpc caen
laboratoire de physique corpusculaire

ROOT
An Object-Oriented Data Analysis Framework

cnrs IN2P3
Les deux infinis

RHB

ROOT Histogram Builder

Quick Start Guide

[September 2012]

This document describes **R**OOT **H**istogram **B**uilder (**RHB**) developed at LPC Caen.
It gives information for RHB development releases (version < 1.0).

This guide is divided in six parts:

- the first one gives two ways to install RHB,
- then you will find an introduction to RHB,
- in the third part, the main RHB analysis components are presented,
- the following part presents the main RHB GUI,
- the fifth part shows some common use cases to start analysis with RHB,
- the references to contact the RHB team are indicated in the last part.

LPC Caen
6 Boulevard Maréchal Juin
14050 – CAEN Cedex

Tel: +33 (0)2 31 45 25 00

Table of Contents

I – Installing RHB

Regardless the installation mode of RHB, note that this software is developed on Linux Ubuntu LTS platform, so it is Debian compliant.

The last Long Term Support version of Ubuntu was 10.04 (*lucid lynx*), and RHB is now available for Ubuntu 12.04 (*precise pangolin*).

I – 1. From source codes

You need first to get a file like “**rhb-1.0.tar.gz**”. If you don't have any, you can provide RHB tarball file by contacting us (see VI – References chapter).

This archive is built with Autotools and you may have to install some required libraries and packages on your desktop/laptop, such as “libtool”, “autoconf”and “automake”.

To install RHB from source codes, follow these steps:

- open a Linux Terminal
- create a target folder: mkdir *myRHB*
- open this folder: cd *myRHB*
- extract RHB archive: tar zxvf rhb-1.0.tar.gz .
- open *myRHB* folder: cd rhb-1.0
- run autotools build:
  - ./configure
  - make
  - sudo make install

I – 2. From LPC repository as Linux package

If you work under a Linux Ubuntu platform, this is the easiest way to get and update the RHB package. It ensures furthermore package dependencies (ROOT from CERN).

CAUTION: Up to now, package is only for **x86_64** ! That is to say only for x86 Intel architecture and for AMD 64 bits.

To install RHB from LPC repository, follow these steps:

- sudo apt-add-repository 'deb http://rhb.in2p3.fr/distribution/ubuntu/ precise main'
- sudo apt-get update
- sudo apt-get install rhb

For example, for the up-to-date version in september 2012, a “**rhb_0.16-9~precise_amd64.deb**” file should be loaded and installed.

To finish installation, you can now verify RHB is successfully installed by simply launching “RHB” command in a Linux Terminal: the main RHB GUI should display.

II – Introduction to RHB

ROOT **H**istogram **B**uilder (**RHB**) is based on the ROOT framework from CERN. Its purpose is to simply declare and fill histograms, without writing any line of source code. Histograms declarations are thus performed using a graphical interface (RHB GUI). The software packages (RHB Core) is in charge of updating declared parameters and filling histograms built by the user.

RHB can work off-line from data files, or on-line directly from DAQ systems such as DAS (GANIL/LPC), FASTER (LPC) or GANDALF (LPSC). The user has to configure a Data Reader (data format) and a Device (data source) declared in a resources file (.RHBrc).

CAUTION: With this version, RHB should be **restarted** each time the format or the source of data is changed.

RHB can be launched in basic or mixed modes from a Linux Terminal:

- **RHB**: standalone mode without any visualization.
- **RHB --visu**: runs RHB with histograms visualization and display a ROOT TBrowser or **RHB –v** (old).
- **RHB --oscillo**: runs RHB with a 2 channels oscilloscope visualization. or **RHB -o**
- **RHB -- RHV**: runs RHB with a dedicated tabs visualization fully configurable by user or **RHB –r** (NEW).

You can get help by passing **--help** or **-h** option to RHB command.
More details are given in manpage, try **man RHB**.

III – Main RHB analysis components description

The main parameters and histograms components are described here, as they should be sufficient to start analysis with RHB. Uses cases are given in chapter V.

III – 1. Parameters

RHB can treat different types of parameters that can be single, vector or persistent. Most of them can be edited by dedicated RHB GUI tabs.

III – 1. a. Raw parameters

Raw parameters are provided by the data source. The user has thus no way to modify them, but they can be used to build calculated parameters (additional or intermediate ones).

III – 1. b. Calculated parameters

Calculated parameters open possibilities to the user to get more "user friendly" parameters (unit conversions for example). They are simple mathematical expressions of other parameters.

They can include variable coefficients whose values can be modified while the analysis is under process.

CAUTION: Calculated parameters rely on the ROOT::TFormula class and their expression are therefore limited to **4 parameters**. If you need more, you will have to introduce intermediate parameters.

III – 1. c. Conditions

Conditions are calculated parameters whose result is 0 (false) or 1 (true).

A condition can be applied to a histogram. In that case, the histogram is filled when this condition is true.

III – 1. d. Graphical cuts

Graphical cuts are conditions drawn on a 2D histogram. Such a condition will be true if a x/y point is inside this graphical curve.

III – 2. Histograms

In RHB, two histograms types exist: ROOT standard ones (1D, 2D, Profile) which are introduced here, and specific ones, such as the following Oscillogram type.

Whatever its type, a condition can be associated to one histogram. In that case, the histogram is filled when this condition is true.

NOTE: A **Generic Building** tab is provided in RHB GUI to define many histograms with a single declaration.

III – 2. a. FH1F

Inherits from the ROOT::TH1F class. This is the standard **1D** histogram class for RHB.

III – 2. b. FH2F

Inherits from the ROOT::TH2F class. This is the standard **2D** histogram class for RHB.

III – 2. c. FProfile

Inherits from the ROOT::TProfile class. **Profile** histograms are histograms in which each channel contains the mean value and the spread of y-axis's parameter distribution for the x-axis's parameter values contained in the channels range.

III – 2. d. FOscillogram

Oscillograms are histograms in which contents is updated each time its vector parameter is set, i.e. each time the parameter is read from source.

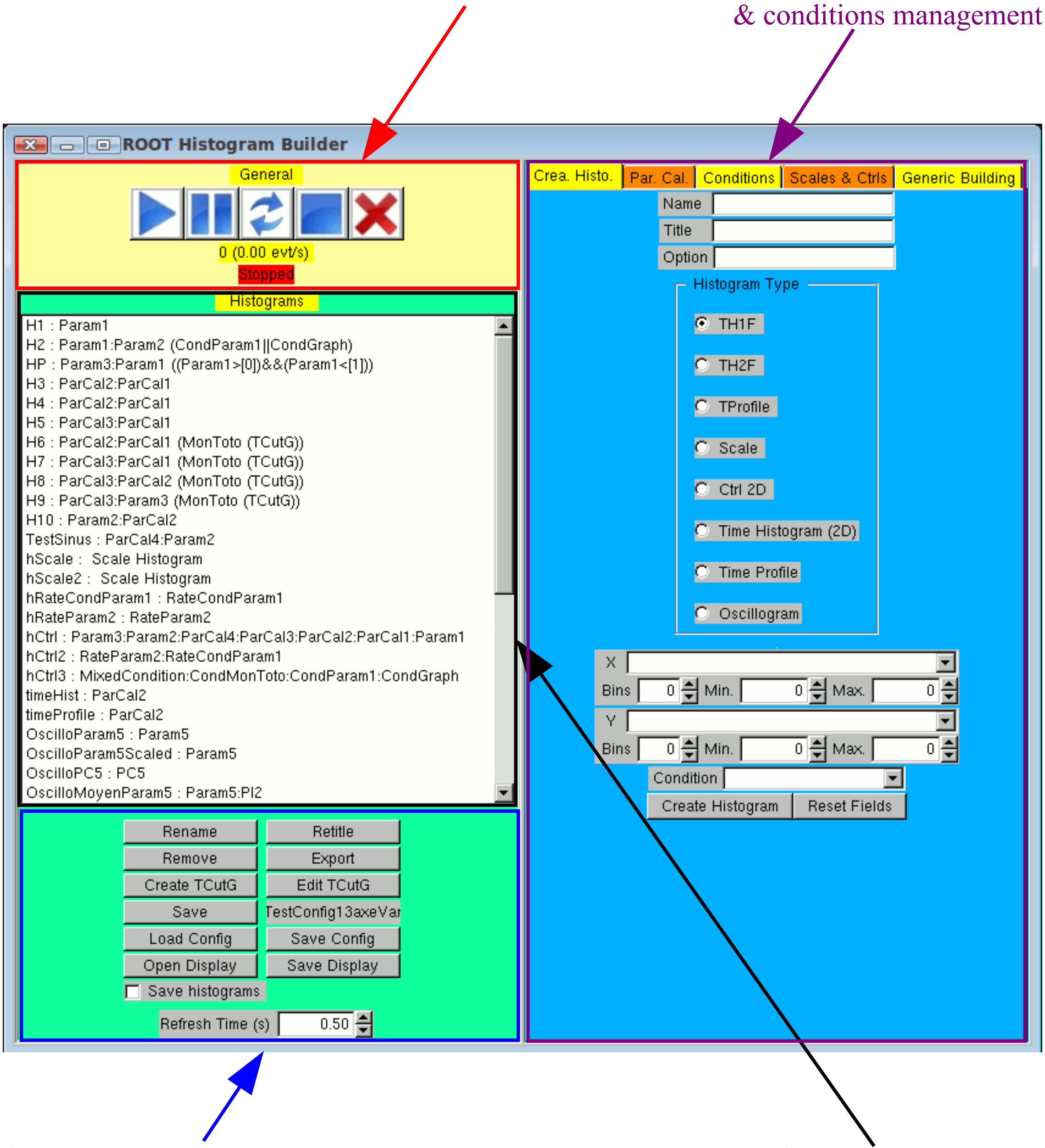
IV – RHB GUI

IV – 1. GUI presentation

The RHB GUI is divided in **4 parts**:

❶ Run control

❹ Histograms, parameters & conditions management



0 (0.00 evt/s)

Stopped

H1 : Param1
H2 : Param1:Param2 (CondParam1||CondGraph)
HP : Param3:Param1 ((Param1>[0])&&(Param1<[1]))
H3 : ParCal2:ParCal1
H4 : ParCal2:ParCal1
H5 : ParCal3:ParCal1
H6 : ParCal2:ParCal1 (MonToto (TCutG))
H7 : ParCal3:ParCal1 (MonToto (TCutG))
H8 : ParCal3:ParCal2 (MonToto (TCutG))
H9 : ParCal3:Param3 (MonToto (TCutG))
H10 : Param2:ParCal2
TestSinus : ParCal4:Param2
hScale : Scale Histogram
hScale2 : Scale Histogram
hRateCondParam1 : RateCondParam1
hRateParam2 : RateParam2
hCtrl : Param3:Param2:ParCal4:ParCal3:ParCal2:ParCal1:Param1
hCtrl2 : RateParam2:RateCondParam1
hCtrl3 : MixedCondition:CondMonToto:CondParam1:CondGraph
timeHist : ParCal2
timeProfile : ParCal2
OscilloParam5 : Param5
OscilloParam5Scaled : Param5
OscilloPC5 : PC5
OscilloMoyenParam5 : Param5:PI2

TestConfig13axeVar

□ Save histograms

Refresh Time (s) 0.50

● TH1F
○ TH2F
○ TProfile
○ Scale
○ Ctrl 2D
○ Time Histogram (2D)
○ Time Profile
○ Oscillogram

X: Bins 0 Min. 0 Max. 0

Y: Bins 0 Min. 0 Max. 0

❷ Environment & display management

❸ Histograms list

IV – 2. GUI parts introduction

❶ Run control:

This part drives the analysis flow. The user can start, stop, pause or refresh reading. It shows an additional event count rate to supervise the experiment.

❷ Environment & display management:

Here are contained all functionalities to set a RHB run. User can set several options, such as the display refresh time of histograms, load or save a configuration, etc …

You will find here action buttons to manage histograms (create, remove, rename or retitle) and to create and edit graphical cuts.

❸ Histograms list:

All declared histograms are listed here. Their name are displayed with possibly condition into brackets. By clicking one of them, user can manage one histogram properties as introduced previously in ❷ .

❹ Histograms, parameters & conditions management:

This part groups analysis components displayed in tabs. The user can manage separately histograms, define calculated parameters and conditions.

A “Generic Building” tab allows to declare several histogram using a single declaration.

Note that a **dedicated syntax** is proposed in RHB GUI to formulate complex expressions for these components.

V – First RHB use cases

Simple builds are described below. They should give you entry points to begin analysis with RHB. Requirements are first introduced.

V – 1. Requirements

When RHB is launched, the main GUI frame is always displayed. But some extra frames may be displayed depending on the options as described in chapter II.

Moreover, if the first launch of RHB is fully empty (without any component declarations loaded), in practice you will probably load a saved configuration, including your own experiment components.

In this case, RHB relies on a **.RHBrc** file containing all the resources to set. This file will specify what Data Reader to use, what Device type to use, and eventually a configuration file ended with a **.facqConf** extension.

The configuration file contains all parameters and histograms description and is written in a **special syntax**. If you are interested in, you can have a look in such a file after a configuration is saved, but editing a configuration file is generally reserved for advanced users.

CAUTION: All RHB required files are **locally** loaded. Take care to be in the good directory before launching RHB with your requirements, otherwise RHB will be launched with default settings.

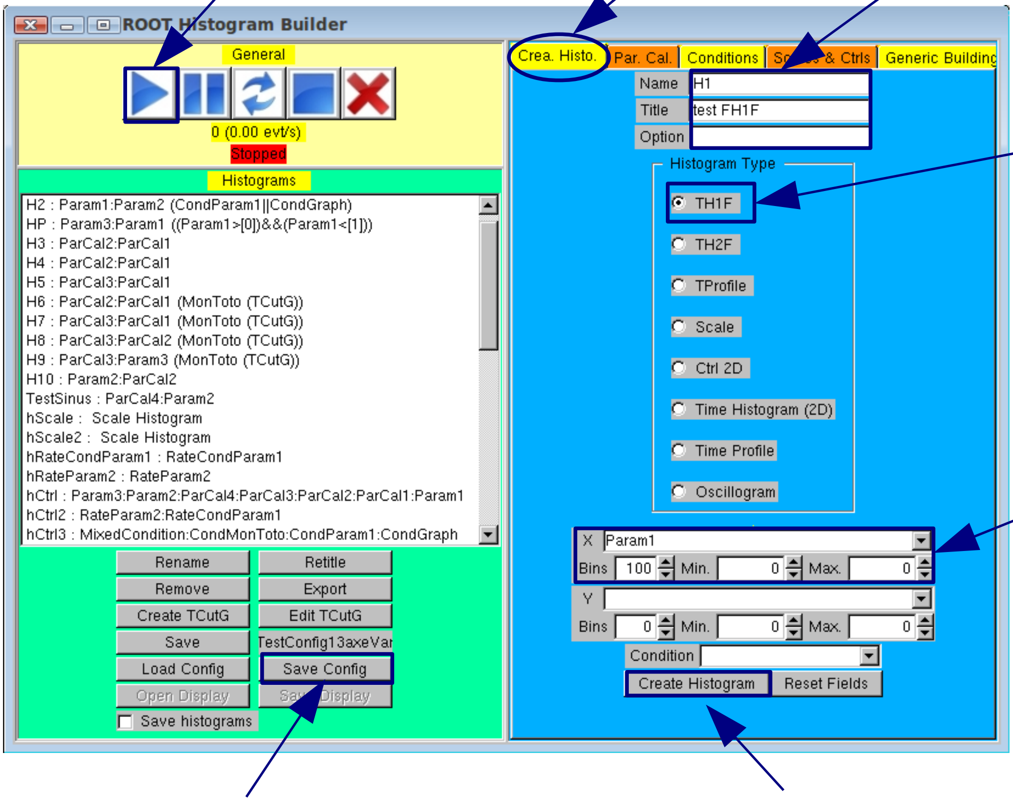
V – 2. Simple 1D histogram

To build your first simple **1D** histogram, follow these steps:

❻ Now, to start running and fill the histogram, you just have to click on this “Play” button.
(assuming RHB has been launched with a valid configuration)

❶ Click on “Crea. Histo” tab

❷ Enter histogram name and title, and choose a display option



0 (0.00 evt/s)

Stopped

```
H2 : Param1:Param2 (CondParam1||CondGraph)
HP : Param3:Param1 ((Param1>[0])&&(Param1<[1]))
H3 : ParCal2:ParCal1
H4 : ParCal2:ParCal1
H5 : ParCal3:ParCal1
H6 : ParCal2:ParCal1 (MonToto (TCutG))
H7 : ParCal3:ParCal1 (MonToto (TCutG))
H8 : ParCal3:ParCal2 (MonToto (TCutG))
H9 : ParCal3:Param3 (MonToto (TCutG))
H10 : Param2:ParCal2
TestSinus : ParCal4:Param2
hScale :  Scale Histogram
hScale2 :  Scale Histogram
hRateCondParam1 : RateCondParam1
hRateParam2 : RateParam2
hCtrl : Param3:Param2:ParCal4:ParCal3:ParCal2:ParCal1:Param1
hCtrl2 : RateParam2:RateCondParam1
hCtrl3 : MixedCondition:CondMonToto:CondParam1:CondGraph
```

☐ Save histograms

Name: H1

Title: test FH1F

Option:

Histogram Type

● TH1F

○ TH2F

○ TProfile

○ Scale

○ Ctrl 2D

○ Time Histogram (2D)

○ Time Profile

○ Oscillogram

X: Param1 — Bins: 100 — Min.: 0 — Max.: 0

Y: — Bins: 0 — Min.: 0 — Max.: 0

Condition:

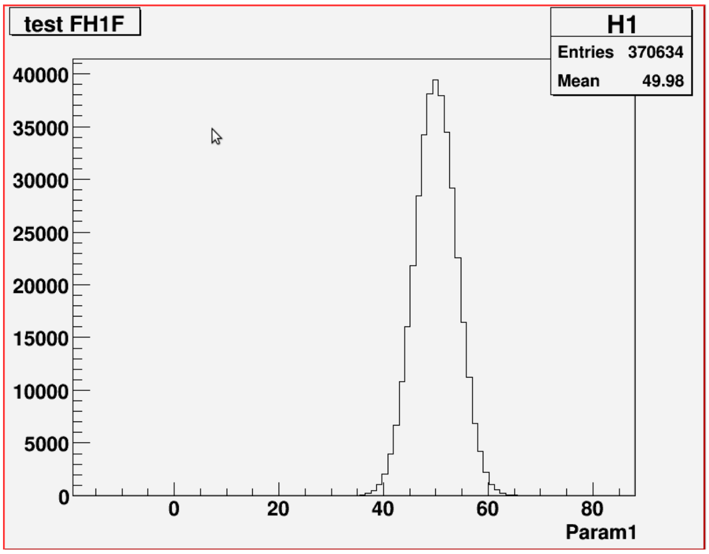
❸ Select histogram type

❹ Select x-axis parameter and set bins and range.
(param1 is a **raw** parameter)

❼ And finally click here if you want to save this histogram in a configuration file.
(TestConfig13axeVar here)

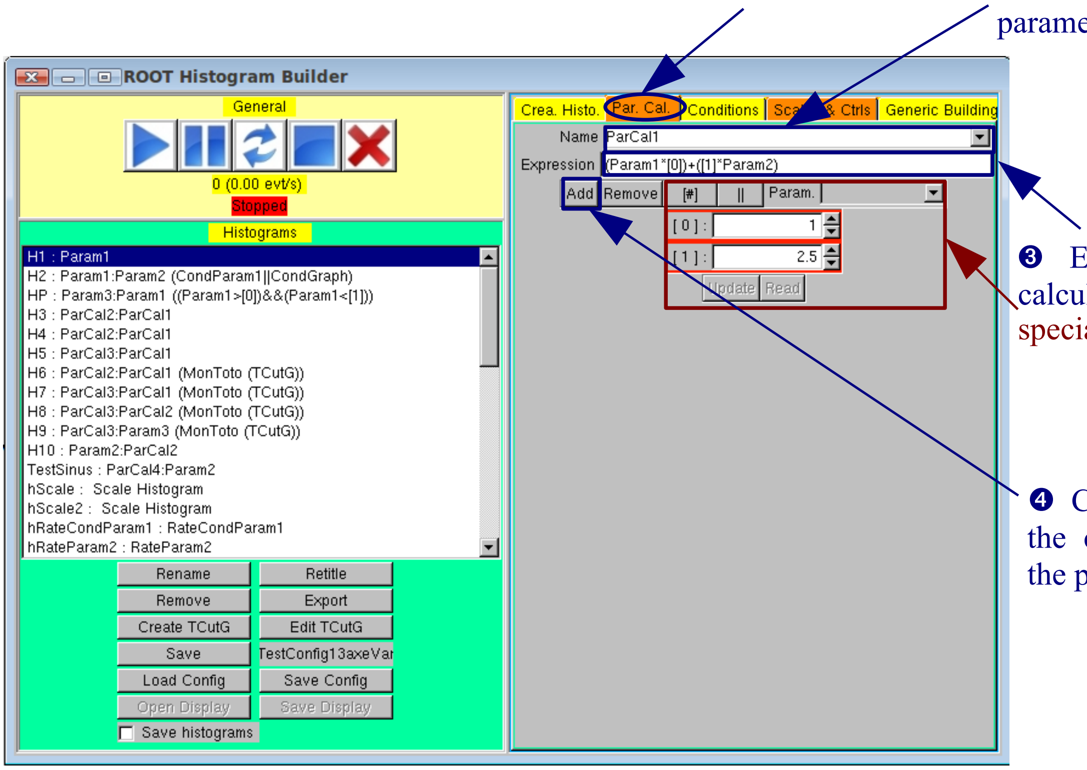
❺ Click here to create histogram.
As a result, a histogram named “H1” is added to the histograms list on the left.

Here is a display example for the created 1D histogram.



V – 2. Calculated parameter

Now, if we want to add a **calculated** parameter:


ROOT Histogram Builder

General

0 (0.00 evt/s)

Stopped

Histograms

H1 : Param1
H2 : Param1:Param2 (CondParam1||CondGraph)
HP : Param3:Param1 ((Param1>[0])&&(Param1<[1]))
H3 : ParCal2:ParCal1
H4 : ParCal2:ParCal1
H5 : ParCal3:ParCal1
H6 : ParCal2:ParCal1 (MonToto (TCutG))
H7 : ParCal3:ParCal1 (MonToto (TCutG))
H8 : ParCal3:ParCal2 (MonToto (TCutG))
H9 : ParCal3:Param3 (MonToto (TCutG))
H10 : Param2:ParCal2
TestSinus : ParCal4:Param2
hScale : Scale Histogram
hScale2 : Scale Histogram
hRateCondParam1 : RateCondParam1
hRateParam2 : RateParam2

Name ParCal1

Expression (Param1*[0])+([1]*Param2)

[0] : 1

[1] : 2.5

☐ Save histograms

❶ Click on “Par. Cal.” tab

❷ Enter the calculated parameter name

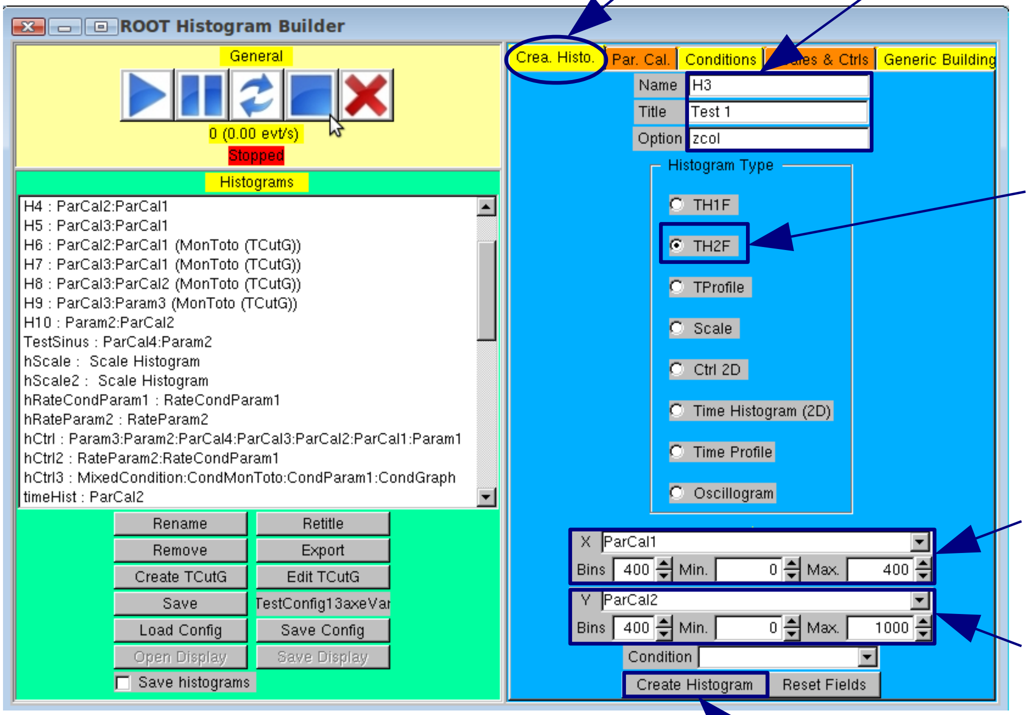
❸ Enter an expression to calculate parameter using the special syntax widgets

❹ Click this button to add the calculated parameter to the parameters list

NOTE: **conditions** rely on the same GUI management, provided user has clicked ❶ “Conditions” tab.

V – 3. First 2D histogram

In the same way than for 1D histograms, follow these steps to build your first **2D** histogram:



ROOT Histogram Builder

General

0 (0.00 evt/s)

Stopped

Histograms

H4 : ParCal2:ParCal1
H5 : ParCal3:ParCal1
H6 : ParCal2:ParCal1 (MonToto (TCutG))
H7 : ParCal3:ParCal1 (MonToto (TCutG))
H8 : ParCal3:ParCal2 (MonToto (TCutG))
H9 : ParCal3:Param3 (MonToto (TCutG))
H10 : Param2:ParCal2
TestSinus : ParCal4:Param2
hScale : Scale Histogram
hScale2 : Scale Histogram
hRateCondParam1 : RateCondParam1
hRateParam2 : RateParam2
hCtrl : Param3:Param2:ParCal4:ParCal3:ParCal2:ParCal1:Param1
hCtrl2 : RateParam2:RateCondParam1
hCtrl3 : MixedCondition:CondMonToto:CondParam1:CondGraph
timeHist : ParCal2

TestConfig13axeVar

☐ Save histograms

Name: H3

Title: Test 1

Option: zcol

Histogram Type

○ TH1F
● TH2F
○ TProfile
○ Scale
○ Ctrl 2D
○ Time Histogram (2D)
○ Time Profile
○ Oscillogram

X: ParCal1 Bins 400 Min. 0 Max. 400

Y: ParCal2 Bins 400 Min. 0 Max. 1000

Condition:

❶ Click on “Crea. Histo” tab

❷ Enter histogram name and title, and choose a display option

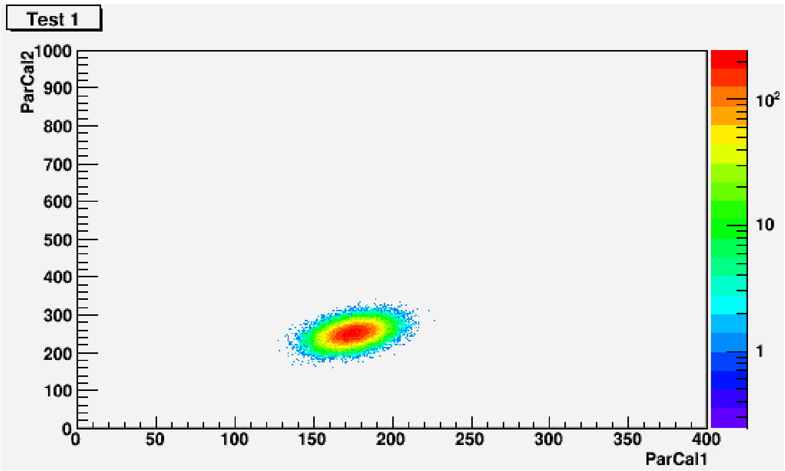
❸ Select histogram type

❹ Select x-axis parameter and set bins and range (**calculated** parameter here)

❺ Select y-axis parameter and set bins and range (**calculated** parameter too)

❻ Click here to create histogram. As a result, named histogram “H3” is added to the histograms list on the left.

A 2D histogram display example with its previous settings is here:



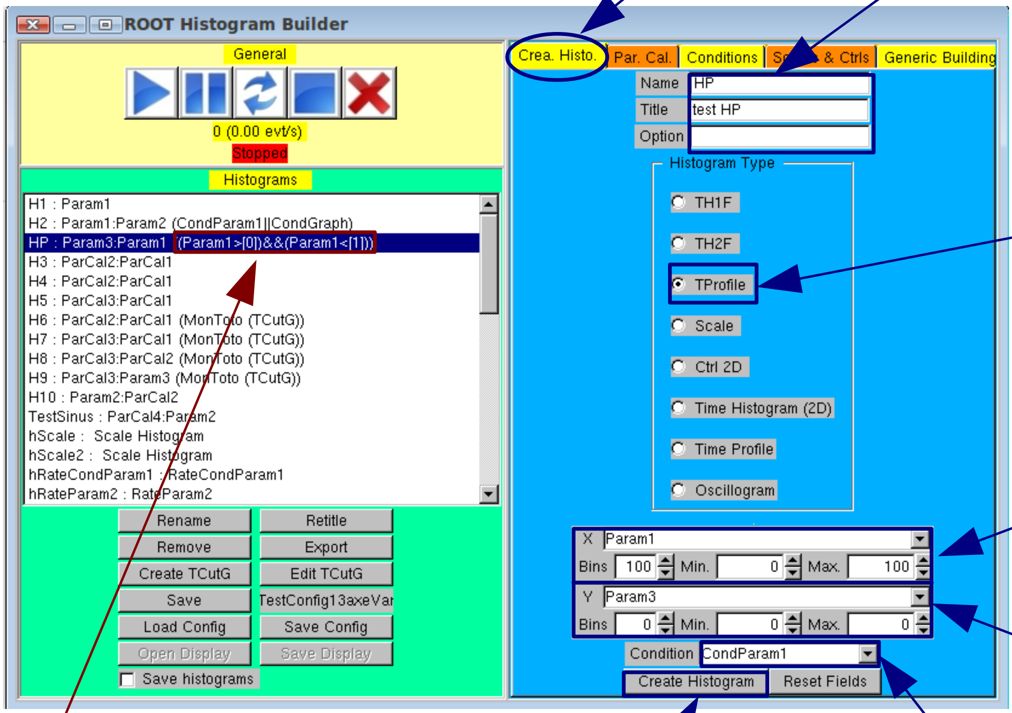
V – 4. Conditioned Bidim

We assume here that a “CondParam1” condition has been created. Remember that a **condition** is declared like a calculated parameter (see V – 2 Calculated parameter).

Now to set this condition to a **2D** histogram, follow this:

❶ Click on “Crea. Histo” tab

❷ Enter histogram name and title, and choose a display option



0 (0.00 evt/s)

Stopped

H1 : Param1
H2 : Param1:Param2 (CondParam1||CondGraph)
HP : Param3:Param1 (Param1>[0])&&(Param1<[1]))
H3 : ParCal2:ParCal1
H4 : ParCal2:ParCal1
H5 : ParCal3:ParCal1
H6 : ParCal2:ParCal1 (MonToto (TCutG))
H7 : ParCal3:ParCal1 (MonToto (TCutG))
H8 : ParCal3:ParCal2 (MonToto (TCutG))
H9 : ParCal3:Param3 (MonToto (TCutG))
H10 : Param2:ParCal2
TestSinus : ParCal4:Param2
hScale : Scale Histogram
hScale2 : Scale Histogram
hRateCondParam1 : RateCondParam1
hRateParam2 : RateParam2

TestConfig13axeVar

□ Save histograms

Name: HP
Title: test HP
Option:

Histogram Type
○ TH1F
○ TH2F
● TProfile
○ Scale
○ Ctrl 2D
○ Time Histogram (2D)
○ Time Profile
○ Oscillogram

X: Param1
Bins 100 Min. 0 Max. 100

Y: Param3
Bins 0 Min. 0 Max. 0

Condition: CondParam1

❸ Select histogram type

❹ Select x-axis parameter and set bins and range (Param1 is a **raw** parameter)

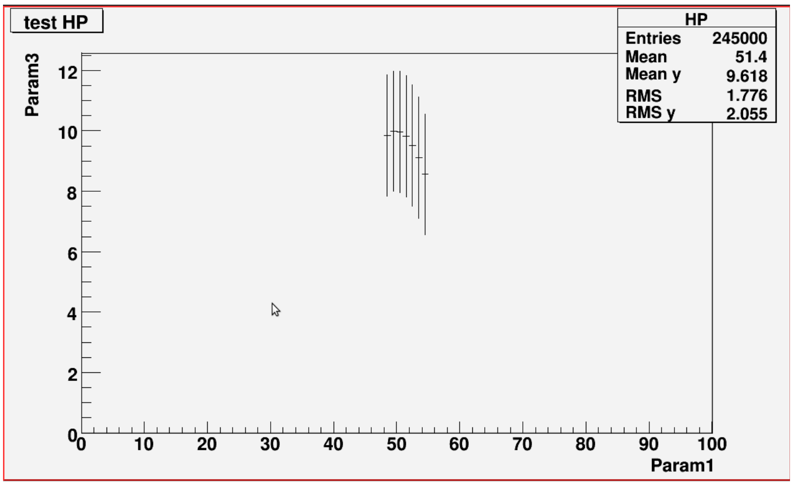
❺ Select y-axis parameter and set bins and range (Param3 is a **raw** parameter too)

❻ Select the condition to apply

❼ Click here to create histogram. As a result, named histogram “HP” is added to the histograms list on the left.

Note: Expression of “CondParam1” condition appears into brackets in the histogram name (with here [0]: 48.5 and [1]: 55)

As a result, the corresponding TProfile histogram should be:

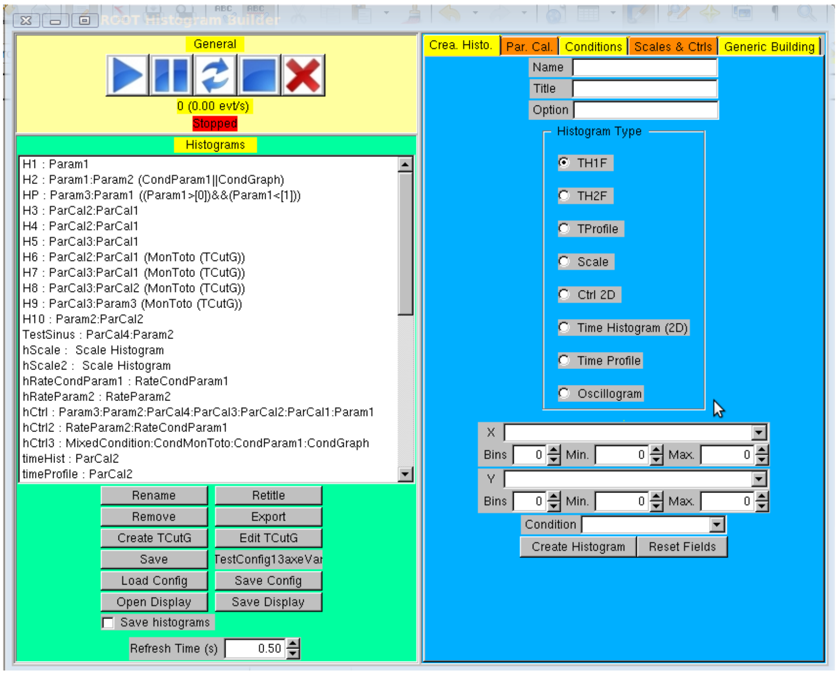


V – 5. Visualization basis

As introduced in chapter “II - Introduction to RHB”, to have visualization facilities, RHB have to be launched with “**-v**” or “**--visu**” option.

In this way, 2 frames are displayed:

… the main **RHB GUI**:

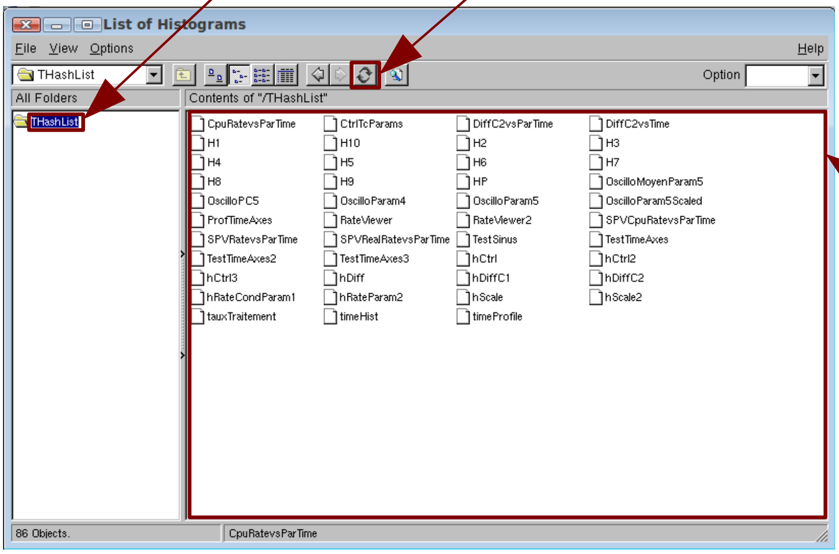


```
H1 : Param1
H2 : Param1:Param2 (CondParam1||CondGraph)
HP : Param3:Param1 ((Param1>[0])&&(Param1<[1]))
H3 : ParCal2:ParCal1
H4 : ParCal2:ParCal1
H5 : ParCal3:ParCal1
H6 : ParCal2:ParCal1 (MonToto (TCutG))
H7 : ParCal3:ParCal1 (MonToto (TCutG))
H8 : ParCal3:ParCal2 (MonToto (TCutG))
H9 : ParCal3:Param3 (MonToto (TCutG))
H10 : Param2:ParCal2
TestSinus : ParCal4:Param2
hScale :  Scale Histogram
hScale2 :  Scale Histogram
hRateCondParam1 : RateCondParam1
hRateParam2 : RateParam2
hCtrl : Param3:Param2:ParCal4:ParCal3:ParCal2:ParCal1:Param1
hCtrl2 : RateParam2:RateCondParam1
hCtrl3 : MixedCondition:CondMonToto:CondParam1:CondGraph
timeHist : ParCal2
timeProfile : ParCal2
```

… **and a ROOT TBrowser:**

Click on “THashList” to refresh the histograms list in the right part

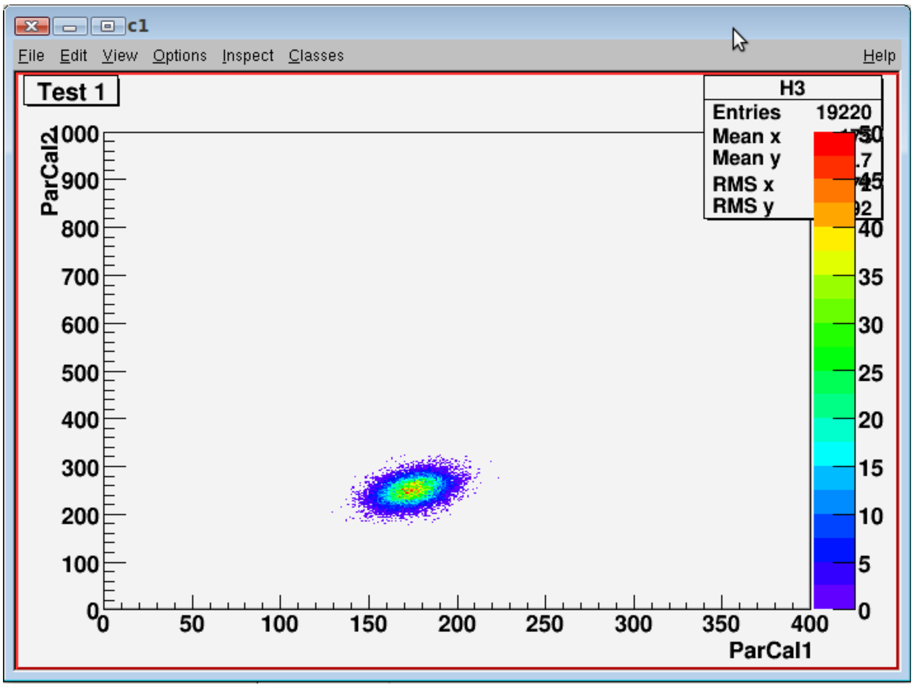
Note: You will need sometimes to refresh the histograms list after creating/removing histograms.



```
CpuRatevsParTime    CtrlTcParams          DiffC2vsParTime   DiffC2vsTime
H1                  H10                   H2                H3
H4                  H5                    H6                H7
H8                  H9                    HP                OscilloMoyenParam5
OscilloPC5          OscilloParam4         OscilloParam5     OscilloParam5Scaled
ProfTimeAxes        RateViewer            RateViewer2       SPVCpuRatevsParTime
SPVRatevsParTime    SPVRealRatevsParTime  TestSinus         TestTimeAxes
TestTimeAxes2       TestTimeAxes3         hCtrl             hCtrl2
hCtrl3              hDiff                 hDiffC1           hDiffC2
hRateCondParam1     hRateParam2           hScale            hScale2
tauxTraitement      timeHist              timeProfile
```

Histograms list: double click on one of them to display it in a canvas.

For example, by double clicking on “H3”, the following display should appear on the screen:



H3

Entries 19220

Mean x

Mean y

RMS x

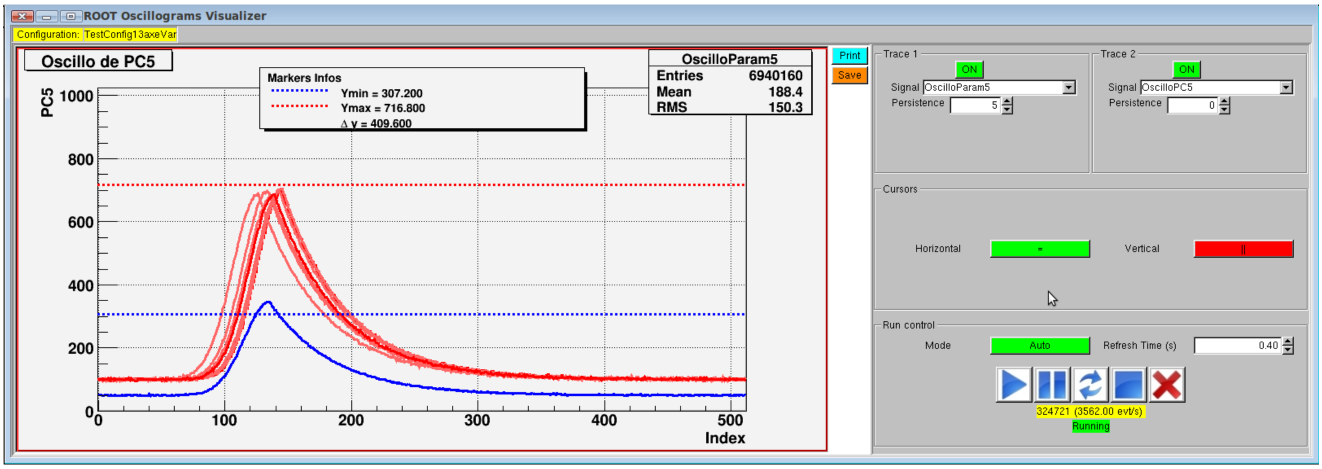
RMS y

NOTE: user can define its own dedicated display environment by writing a C++ source file ended with “**.C**”. It will be loaded and executed by ROOT as a macro file. To use it do not forget to setup the “**ConfigFile**” item in the **.RHBrc** file.

V – 6. Oscilloscope run

As said in chapter II, you have to specify the “**-o**” or the “**--oscillo**” option to open the RHB Oscilloscope.

In this way, a typical frame like this one will be opened:



Configuration: TestConfig13axeVar

Markers Infos

Ymin = 307.200

Ymax = 716.800

Δ y = 409.600

OscilloParam5	
Entries	6940160
Mean	188.4
RMS	150.3

Trace 1: Signal OscilloParam5, Persistence 5

Trace 2: Signal OscilloPC5, Persistence 0

Mode Auto

Refresh Time (s) 0.40

324721 (3562.00 evt/s)

Running

User gets thus expected standard oscilloscope functions to select signals, set cursors on axis and select a persistence value for both signals.

V – 7. Offline reading

Reading offline data means reading from a **data file**. In order to read a data file, you will have to set some items in your .RHBrc file. Here is an example to read a FASTER data file:

# character is used for comments

```
.
.
.
#========================================
# Configuration file
#-----------------------

ConfigFile: TestConfig13axeVar.facqConf
.
.
.
#========================================
# Data Reader
#-----------------
#  Experiment with FASTER:
# UDP/IP Netwowk -> FIPFasterDataReader
# File           -> FFileFasterDataReader
#
DataReader: FFileFasterDataReader
.
.
.
#========================================
# Device
#---------
# For FASTER experiments (FFasterDataReader):
# UDP/IP Netwowk -> FIPDevice
# File           -> FFileDevice
Device: FFileDevice
.
.
.
#========================================
# File name to read
#----------------------
FFileDevice.FileName: /test/or1_0001.fast
.
.
.
```

Put here a valid File Data Reader name

Always put here FFileDevice to read data from files

Specify here the data filename, including path name

By associating this .RHBrc file with a saved configuration file (.facqConf), you are able to re-run your experiment as if it was on-line.

<u>CAUTION</u>: RHB will be in general slower than a dedicated analysis software. It may be nevertheless easier to use since no programming skills are required to use RHB.

Now you should know how to build and manage analysis with RHB, so enjoy it !

VI – References

If you need more information, if you have any question or if you find bugs, you can contact RHB team:

- Authors: D. Cussol (cussol@lpccaen.in2p3.fr), RHB project leader
  T. Chaventré (chaventre@lpccaen.in2p3.fr), RHB developer

- Project web site: https://rhb.lpc-caen.in2p3.fr