LABORATORI NAZIONALI DI FRASCATI

FINUDA Note 53/LNF/STB/07
18 Luglio 2007

Procedure for the calibration of FINUDA straw tubes

Simone Bianco

INFN, Laboratori Nazionali di Frascati, Frascati, Italy.

Abstract

This note describes the new programs developed to check the straw tube T_0 stability during the data taking. It allows to monitor T_0 variation and to produce new calibration tables to be inserted into the database.

1 Introduction

The determination of the time origin (T_0) of the TDC distribution produced by the FINUDA straw tubes is a really important operation. In fact it allows to have correct time measurements overcoming the differences between each electronic channel. To reach this goal we have to put together the data coming from a lot of runs (in such a way we can have good statistics) and fit the resulting distribution with a peculiar function (see [1] chapter 5.3). In this way we get the T_0 parameter (the second flex point of the fit function) which is our estimate of the time origin.

T_0 is not a constant, it changes moving from straw to straw and it can also change during time. Therefore, it is really important to follow its trend during all the experimental data taking. The program which I developed reads data files written in text form (I used `StbT0Fit` by Alexey Krasnoperov to merge and fit all the data) and allows to plot the evolution of a fit parameter as a function of the run number.

2 Program's details

The program is placed in the following folder:
`finudast@pcfin3:~/StbT0Calibration/PlotT0Evolution`.
The source file **main.cxx** can be found in the subfolder *src*. The data files to be processed must be placed in the main folder of the program. Then all the names of theese files have to be put in the code in such a way:

```
const char *fname[ ] =
{
"N_FINU_3500_3699_FitParams.txt",
"N_FINU_3700_3899_FitParams.txt",
"N_FINU_3900_4099_FitParams.txt",
...
0};
```

If you want to do this operation in a simplier way, you can use the *MakeSortedList.csh* script present in the main folder. This script produces a list of names (in the correct format and order) of the files present in the folder. You can modify this script to make it working with files named with a different structure. At the end you can insert the produced list in the code.

2.1 How to use MakeSortedList

MakeSortedList is a script based on the awk tool. You should modify it to make it working with your text files. You have to look at the names of your files. The script sets "_" as divider, so every "_" present in file names begins a new block in the name structure. We want to print the first block containing numbers (in such a way we can order numerically file names), so the first awk print instruction must be referred to this block. Also the *ls* instruction has to be correctly set. For example if your data are saved in a file named "New_FINU_3000_3199 ... .txt" you will have to modify the script in this way (using Emacs or other editors):

```
#!/bin/csh -f
#
#

ls New_FINU_*FitParams.txt | awk -F "_" '{print $3,$0}'
 | sort -n \\| awk '{print "\"\"$2"\","}' > LIST
```

Otherwise if your files are named "FINU_3000_3199 ... .txt" the correct sintax for the script is:

```
#!/bin/csh -f
#
#

ls FINU_*FitParams.txt | awk -F "_" '{print $2,$0}'
 | sort -n \\| awk '{print "\"\"$2"\","}' > LIST
```

In both the examples the output list is called *LIST*.

2.2 How to modify reading operations

At the beginning the program defines a structure:

```
struct StbInfo // declaring a structure for the data
{
   int   id;
   int   layer;
   int   straw;
   float chi2;
   float chi2ndof;
   int   ndof;
   float param[6];
   float error[6];
   string fittype;
   void clean(void)
   {
     chi2=0;
     chi2ndof=0;
     ndof=0;
     bzero(param,sizeof(param));
     bzero(error,sizeof(error));
   }
  }
```

If you change the format of the text file you will have to modify also this structure and the reading instructions. In the default configuration we read data in this order: layer (the layer number), straw (the straw number), fittype (a string declaring the fit type used: COSM or HYP), chi2 (the χ^2 value), chi2ndof (the value of χ^2 divided by the number of degrees of freedom), ndof (the number of degrees of freedom), param[0] (the first fit parameter), error[0] (the error on the first fit parameter), ... , param[5], error[5].

3 How to run the program

From the program folder type the command *./Evolution* to start the application. The program displays the loaded text files (advising of any possible error occurred opening or reading the files). Then the user will be asked to choose a layer and a straw tube. Finally a ROOT's canvas is created with the plot of the T_0 evolution (which in the default configuration corresponds to *param[1]*). If you want to plot another parameter you just have to change the parameters' counter in all the instructions regarding the graph creation (for example param[3] and error[3] in lieu of param[1] and error[1]). Obviously, you can divide your canvas and plot the evolution of different parameters. It is also possible to display them on the same graph, you just have to add graphs to the already exsisting multigraph *mg*.

4 Multiplot

Another program has been developed in a similar way, it is named *Multiplot* (source file *src/Multiplot.cpp*). *Multiplot* allows to make a lot of plots (corresponding to different straw tubes) not having to load all the text files any time. The program, after having shown the first plot, asks again for the layer number. You just have to select a layer and a tube on it to see the new evolution plot in the canvas. In this way you are not able to work in the canvas (you cannot zoom, save, print, ...); you have to confirm the selection of the tube to enable all the GUI tools. So, if you want to work on a graph, after having plotted it assign 999 as layer number. In such a way you will have the use of all GUI instruments.

If you want to compile *Multiplot* you have to rename *Multiplot.cpp* as *Multiplot.cxx* and *main.cxx* as *main.cpp*. In such a way the `Makefile` will work only on Multiplot source code. If you want to set a different name for the produced file you can modify the `Makefile` (PROGRAM := new name).

5 Calibration of the straw tubes

The calibration of the FINUDA straw tubes can be completed adding to the databases all the obteined T_0 values. To do this we just have to insert the UNIX time references in the text files conteining T_0 values (for example FINU_10000_10199_DbT0.txt). In this way we will be able to correct every time measurements of the TDC modules of the straw tubes. Therefore the database will contain the correct T_0 value for each straw tube for the whole run period.

References

[1] L. Benussi Doctoral Thesis,
http://www.lnf.infn.it/esperimenti/finuda/tesi_dot_benussi.pdf.

[2] R. Brun and F. Rademakers, “ROOT Referece Guide”,
http://root.cern.ch.

[3] R. Brun and F. Rademakers, “The ROOT System”,
http://root.cern.ch.