CLAS_NOTE 97-007

Drift Chamber Pulser Analysis Code

F. Roudot
Centre d'Etudes de Saclay

October 27, 1997

Contents

1 Introduction

This documentation describes the Fortran code developped to analyse the **Drift Chambers calibration runs** (using the pulser). This code also fills the other drift chamber calibration banks (geometry and time–to–distance parameters).

This program is RECSIS based and sits on the DCCALIB account under the directory RECCAL.

The idea of this program is to analyse the pulser data on a sector by sector basis. Three steps are needed to analyse an entire pulser run.

- Create a set of calibration reference files (if this is the first analysis) for each sector.
- Run the code for the 6 sectors again to extract the new calibration constants and to create a new set of calibration reference files for the next calibration analysis.
- Create the global Drift Chamber Calibration file, including *geometry* (misalignment and misorientation) *banks* and *time–to–distance* parameter *banks*.

At the end of this documentation, I will explain how to proceed, but let me start with a general presentation of the DCCALIB account features.
The chapter 2 of this documentation is devoted to the the DCCALIB account directory structure. In the Chapter 3, the general scheme of the RECCAL software is exposed, the flow chart is shown, a summary of the *TCL variables* is given and the bos banks are detailed. In chapter 4 the *calibration procedures* are explained in detail. The chapter 5 summarizes the steps required for a complete analysis of a *Drift Chamber Calibration run*. The chapter 6 shows how to vizualise the Delays Calibration results some results of the last date pulser data run are given has an example. In The last chapter an overview of the Time–to–Distance calibration is given.

2 DCCALIB directory structure

Several directories are at the same level as RECCAL. This scheme was design to simplify the information flow, and also to separate utilities, output files and information files.

2.1 RECCAL

The calibration analysis is done in this directory. The RECCAL directory contains the known RECSIS directory structure.
In order to simplify the calibration code (and also historicaly) RECCAL was developped in the RECSIS USER subdirectory.
As there is no link to the recsis's detector packages libraries in RECCAL, I have modified some code in the RECSIS, RECUTL and CMS subdirectories; that is why they are in the RECCAL subdirectory. If the INCLUDE area is also present it's due to the removal of the tcl

variables related to the detector packages.
The USER subdirectory is the central piece of this code, all the calibrations routines are there (together with the usual user routines) and also several tcl initialisation files (this will be describe in the following section).

2.2 INPUT

The input data files directory contains.

- The 10 crate maps *dc_fb1.map* ... *dc_fp10.map* files.
  The contents of the maps are summarized in the Table 1. Each slot in a given *FastBus crate* is associated with a given part of the drift chamber. This information is summarized in the Table 2.

TDC slot	ADB width	FB	TDC channel	layer	wire	sector
$4 \rightarrow 23$ 10 & 17 not used	0: short pulse 1: long pulse	$1 \rightarrow 10$	$0 \rightarrow 95$	$1 \rightarrow 36$	$1 \rightarrow 192$	$1 \rightarrow 6$

Table 1: Variation range of MAP parameters

FASTBUS	SLOT $4 \rightarrow 9$	SLOT $11 \rightarrow 16$	SLOT $18 \rightarrow 23$
1	R3_S3_AX	R3_S4_AX	R3_S3_ST
2	R3_S5_AX	R3_S4_ST	R3_S5_ST
3	R3_S2_AX	R3_S1_AX	R3_S2_ST
4	R3_S6_AX	R3_S1_ST	R3_S6_ST
5	R1_S2_AX_ST	R1_S1_AX_ST	R1_S3_AX_ST
6	R1_S6_AX_ST	R1_S4_AX_ST	R1_S5_AX_ST
7	R2_S3_AX	R2_S4_AX	R2_S3_ST
8	R2_S5_AX	R2_S4_ST	R2_S5_ST
9	R2_S2_AX	R2_S1_AX	R2_S2_ST
10	R2_S6_AX	R2_S1_ST	R2_S6_ST

Table 2: FastBus Slot to **R**egion, **S**ector, **AX**ial or **ST**ereo conversion

- The input files for the geometry calibration *chamber.dat* and *torus.dat*.
  The CHAMBER.DAT file contains the mis–position and mis–orientation of the 18 chambers (1 row per chamber). The TORUS.DAT file contains the mis–position (1^{st} row) and the mis–orientation (2^{nd} row) of the toroid.

- The input files *dc_xvst_ri.dat* (i = *1, 2* and *3*) for the distance to time calibration. One file for each region to take into acount the gas admixture difference between regions. The region 2 has more parameters to take into account the magnetic field

dependance.
The contents of these files is the same as the one described in the *Time-to-Distance Calibration Bank* section of the next chapter. There is one row per sector and layer in each data file.

- The badwire list (Richard Thompson's PDU output file) *badwire.dat.*

These input files have to be modified by hand if a change occure in the *Fast Bus* addressing, but also each time a *survey or an external geometry calibartion* of the drift chamber has been done. The Time–to–Distance parameters (smear time, maximum drift time and drift velocity) has to be modified in the input file in order to feed the bos banks with the proper parameters if someone notice a change.

2.3 OUTPUT

The output BosFPack files directory contains.

- *dc_cal_ref_s1.bfp* ... *dc_cal_ref_s6.bfp* (bank DCCR).
- *calib_out_s1.bfp* ... *calib_out_s6.bfp* (bank DCCR).
- *dc_calib_s1.bfp* ... *dc_calib_s6.bfp* (bank TDLY).
- *dc_calib.bfp* (global file).

These files are produced by RECCAL and contains the internal (bank DCCR) and external (bank DDLY) useful calibration parameters. The **dc_calib.bfp** file contains all the calibration BOS banks and will be used to feed the MAPMANAGER.

2.4 REPORT

This directory contains the report text files requested by the user.
For each crate and slot an indexed file is created. The index value is $num = 100 \times (crate\_number - 1) + slot\_number$.

- *cal_num.report* A full report of the peak_search procedure for a given crate and slot.
- *cal_num_br.report* A brief report of the peak_search procedure for a given crate and slot.
- *res_num.report* A brief report of the residual_ana procedure for a given crate and slot.

The contents of this huge number of report files will be explained in the **Calibration Procedure** chapter.

2.5 KUMAC

Several kumac files are on this directory, it is useful to visualize the results at different step of the calibration analysis.

- *tdc_s1.kumac ... tdc_s6.kumac* the raw tdc information decoded in a crate and slot basis for each sector.
- *del_s1.kumac ... del_s6.kumac* the extracted cable length and cable delay value displayed in a crate and slot basis for each sector.
- *tdel_s1.kumac ... tdel_s6.kumac* the calculated cable delay and associated status displayed in a layer versus wire plot for each sector.
- *tdif_s1.kumac ... tdif_s6.kumac* the calculated cable length and associated status displayed in a layer versus wire plot for each sector.

All these kumac files use the **.rzn files** sitting on the *dccalib* area of the *scratch* disk.

2.6 DOC

The documentation sits here, it will be the place to put some important results in the futur.

2.7 THE SCRATCH AREA

As the calibration code is RECSIS based, it produces a *rzn file* (containing *1-d* or *2-d histograms* and *ntuples*). In the case of the calibration, these files are to big to be somewhere in the DCCALIB tree.
I decided to put the RECCAL's *rzn files* on the scratch disk associated to the machine from where the code was launched under the directory DCCALIB. The name of the *rzn file* **isn't** indexed by the job number as in recsis but by the *analysed sector number* in case on single sector analysis **sect1.rzn ... sect6.rtzn**. In case of complete analysis the name of the rzn file is **global.rzn**.
The contents of the **.rzn files** will be discused later in the **Calibration Procedure** chapter.

3 General Calibration Scheme

In this section the structure of the calibration code is described, a general flow chart is presented, control *TCL* parameters are discussed and the, *BOS* banks are detailed.

3.1 Flow Chart

The calibration analysis flow chart is schematized in the Fig. 1.

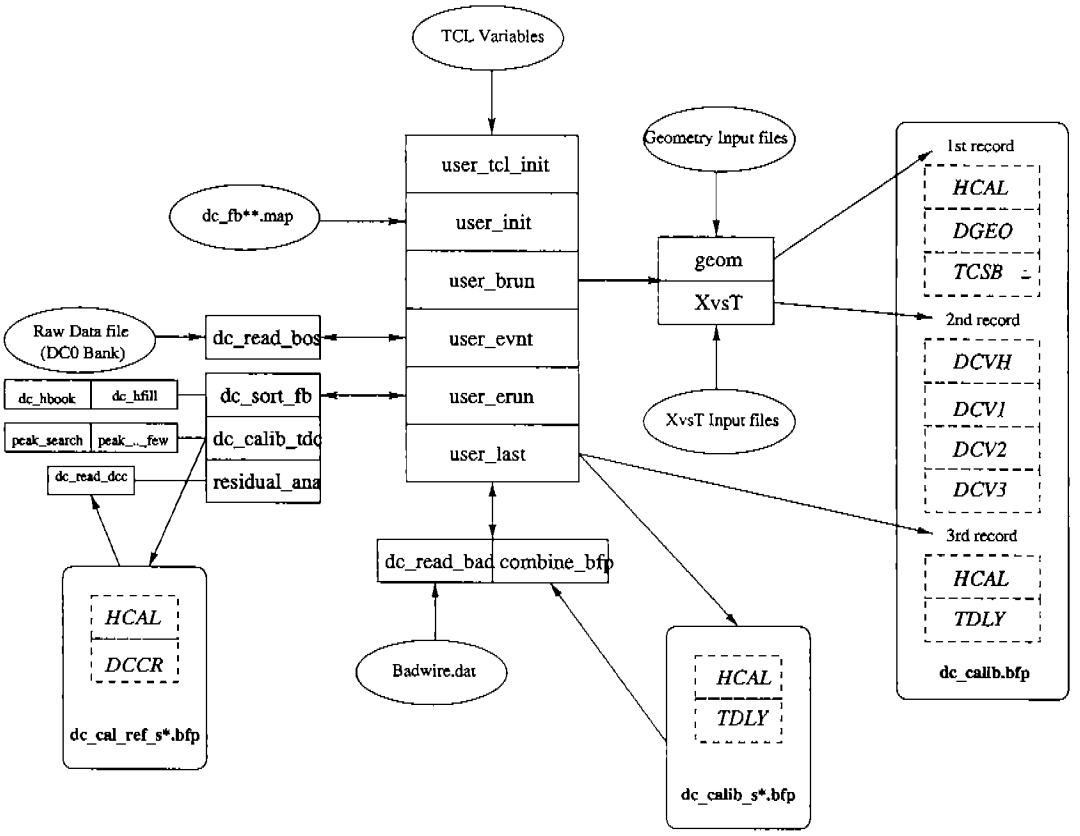

Figure 1: Input, Output and modules of **REC_CAL**

The *full line boxes* represents the **subroutines**.
The *dashed line boxes* represents the **bos banks**.
The *rounded edge boxes* represents the output **BosFPack** file.
The *ovals* represents the **input files**.
The left hand side of this scheme is related to the pulser analysis.
The central part is the common user flow chart.
The XvsT–geom section is related to the geometry and time–to–distance calibration.
The right hand side represents the contents of the general CALIBRATION OUTPUT FILE.

This program is controled via numerous *TCL variables*. The calibration results are stored in several *BOS banks* in order to be used in further calibration analysis or in REConstruction and analySIS code: RECSIS. The tcl variables and the Bos banks are described in the following sections.

3.2 *Calibration* TCL Variables

The *TCL variables* describe in this section are used everywhere in the code, they are defined and some default values are buried in the code in the routine USER_TCL_INIT. Some of these

tcl variables control the analysis and are reinitialized in the INIT_CAL_***.TCL files.

3.2.1 Specific Calibration TCL variables

- **num_peaks** (integer) The number of peak seek by the peak_search routine.
  (*Default 2*)

- **t_off_pulser** (real) The pulser offset. Used to give absolute calibration constants.
  (*Default 3600*)

- **global_cal** (logical) The switch to produce the full analysis BFP FILE.
  (*default false*)

- **sect_ana** (integer) To analyze the given sector (global_cal has to be false).
  (*default 6*)

- **creat_ref** (logical) To creat a set of reference bank for a given sector. Useful for a first pulser run analysis for example. (*default false*)

- **resid_ana** (logical) To make a residual analysis on the result of the calibration. A set of reference bank should be present in the OUTPUT subdirectory.
  (*default false*)

- **brief_out** (logical) To control the level of information in the report files present in the REPORT subdirectory.
  (*default true*: A short report file is created for each crate and slot during the peak searching procedure). If this tcl variable is set to false a more complete set of file will be produced (see analysis section in this documentation).

- **only_xvst** (logical) To fill the Time–to–Distance banks.
  (*default false*)

- **only_geom** (logical) To fill the Geometry banks.
  (*default false*)

3.2.2 Recsis Used TCL variables

The common RECSIS tcl variables are used let me just highlight some useful ones.

- **ddl_file** We have to use that tcl variable in order to use only the **DC0** and **HEAD** bos banks.
  (*default clasbanks.ddl* defined in the tcl initialization files)

- **inputfile** The name of the input data file.
  (defined in the tcl initialization files)

- **rec_output** The name of one of the *BosFPack* output files.
(defined in the tcl initialization files). This file is used to store the internal calibration banks (DCCR) stored in the **JW** array.

3.3 *Calibration* BOS Bank

All banks are defined and commented using *ddl files*. These files are:

- clasbanks.ddl (for the raw data HCAL and DC0 banks).
- geom.ddl (for the *geometry* calibration banks).
- dc_calib.ddl (for the *delay* and *time–to–distance* calibration banks).

A second version of txt2bos have been written to recognize calibration banks (**C list, array JW**) and raw data banks (**E list, IW array**). To avoid mixing between raw data and calibration data, is safer to use the **JW** array to store calibration parameters during calibration run analysis. In order to use a third kind of calibration variable (the reference value) I have created a third bos array for the calibration reference values: the **KW** array.

3.3.1 Header Banks

HREF: the Header CALibration bank. *2 words*

- Version number
- Run number

HCAL & DCVH: the Header CALibration banks. (*5 words*)

- Version
- Low run number
- High run number
- Calibration time
- Calibration type
  - 1: Internal Calibration bank
  - 2: Time Delay Calibration bank
  - 3: Geometry bank
  - 4: XvsT bank

3.3.2 Internal Delay Calibration Banks

DCCR: the Drift Chamber Calibration Reference bank.
(*crate and slot indexed bank* $\mathrm{rec_{num}} = \mathrm{crate_{num}} + 2^5 \times \mathrm{slot_{num}}$, *768 words per record*).

- Electronic Channel
- ADB to STB board Cable length
- Half sum of the pulsing in and out peak position.
- Status.

3.3.3 External Delay Calibration Banks

DDLY: Drift chamber time DeLaY bank.
(*sector indexed bank* $\mathrm{rec_{num}} = \mathrm{sect_{num}}$ *20436 words per record*).

- Wire Id. (as defined in 94.012 Clas note)
- Time Delay.
- Status Word.

3.3.4 Geometry Calibration Banks

DGEO: Drift chamber GEOmetry survey bank (*8 words* $\times$ *18 chambers*).

- Sector Id.
- Region Id.
- Mis-alignment $\vec{X}$. (of the Bogdan Center)
- Small sines of Mis-orientation $\vec{s}$. (arround the 3 directions)

TCSB: Toroidal Coordinate System mis-alignment and mis-orientation Bank (*6 words*).

- Mis-alignment of toroid $\vec{X}_t$. (in the HCS)
- Small sines Mis-orientation $\vec{s}_t$. (arround the 3 directions)

3.3.5 Time–to–Distance Calibration Banks

DCV1: Region 1 parameters for Time–to–Distance function.
(*sector indexed bank* $\mathrm{rec_{num}} = \mathrm{sect_{num}}$ *5 words* per record).

- Smear Time
- Drift velocity
- Maximum drift time
- spare parameter
- spare parameter

DCV2: Region 2 parameters for Time–to–Distance function.
(*sector indexed bank* $\mathrm{rec_{num}} = \mathrm{sect_{num}}$ *14 words* per record).

- Smear Time (0^o)
- Drift velocity (0^o)
- Drift velocity function first parameter (0^o)
- Drift velocity function second parameter (0^o)
- Maximum drift time (0^o)
- Tmax function first parameter (0^o)
- Tmax function second parameter (0^o)
- Smear Time (30^o)
- Drift velocity (30^o)
- Drift velocity function first parameter (30^o)
- Drift velocity function second parameter (30^o)
- Maximum drift time (30^o)
- Tmax function first parameter (30^o)
- Tmax function second parameter (30^o)

DCV3: Region 3 parameters for Time–to–Distance function.
(*sector indexed bank* $\mathrm{rec_{num}} = \mathrm{sect_{num}}$ *5 words* per record).

- Smear Time

- Drift velocity (slope)
- Maximum drift time (Tmax)
- spare parameter
- spare parameter

3.4 Relation to the MAPMANAGER

The code was developed during the last 2 years, much before the selection of the MAPMANAGER as a run indexed database for the whole *CLAS Detector* calibration constants. That's why the code use such number of *BOS banks* and *BosFPack files*.

In order to use the code as it is, Joe manak set some routines to fill the MAPMANAGER with the output banks of the calibration code. These routines are in the DC_MAP subdirectory of the UTILITIES CLAS packages ($CLAS_PACK).

4 Calibration procedure

In this chapter, I will focus on the specific routines for the extraction of the so called **t_0** the **cable delays** used in the REConstruction and analySIS code. The Fig. 2 is an enlargment of the Fig. 1 in the delay calibration area.

In this chapter the purpose of the major subroutine of the delay calibration part of the code are described. In the first section, the sector by sector analysis routines are highlighted. In the second part the complete calibration procedure are explained.

4.1 Sector by Sector analysis

The Delay Calibration is done in a sector by sector basis, meaning the code has to be run one time for each sector; the analysed sector is defined with the *sect_ana TCL* variable. The analyis is done in a crate and slot basis, for each slot 192 channels are analysed (96 $\times$ 2 – long and short pulses –); the intermediate (*see* internal calibration banks) result are stored in this crate and slot basis but the final results are stored in a sector, layer and wire basis as RECSIS will expect them.

In the following the major part of the code will be describe in a routine by routine basis.

4.1.1 user_init

RECCAL has to know the *FastBus crate* configuration in order to proceed, this is done by reading the map *dc_fbi.map* ($\mathbf{i} = 1 \ldots 10$) located in the DCCALIB's INPUT directory. The contents of thes map are stored for the *FastBus crate* information in the huge *crate_id,*

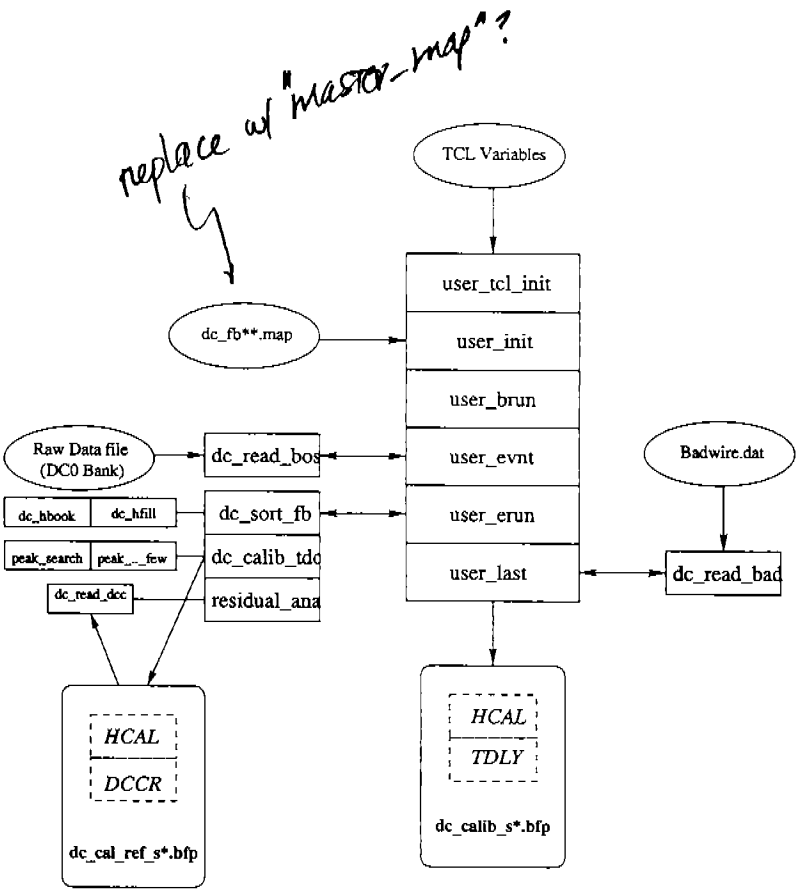

Figure 2: Delay Calibration Highlight

slot_id, chan_id, wid_id (sector, layer and wire) 3–D arrays; for the *drift chamber addressing* information in the huge *addr_id, sect_id, lay_id, wi_id* (slot, width, crate and channel) 4–D arrays. These arrays are saved in the dc_sort common block in the dc_sort.inc include file.

In this routine the **dc_calib_*si*.bfp** (i = *sect_ana TCL var.* file is also open. This file is define as unit 3, and will contain the external bos bank information stored in the **JW** array.

4.1.2 user_evnt

At event time, the raw data are read using the **dc_read_bos** routine.

dc_read_bos: The purpose of **dc_read_bos** is to dump the drift chamber raw data into the *dc_channel_number* and *dc_tdc_value* array (event number and hit number 2–D arrays) **for the current analyzed sector**. Based on that information the *dc_nchannels* and *dc_ok* (event number 1–D array) are created. These arrays are saved in the dc_raw_event common block in the dc_raw_event.inc include file.

Back to the **user_evnt** routine, if dc_ok was true the number of channel fired for the current event is dumped in an ntuple for analysis purpose.

The event loop stops when the run is over or when the number of event exceed the size of the defined number of event *dc_maxevent* (21000).

4.1.3 user_erun

The Delay Analysis is done in that routine. The first step is to sort the information which is curently known for the analyzed sector in a *tdc address basis* in a crate and slot basis. It is the Purpose of the **dc_sort_fb** routine.

dc_sort_fb: The purpose of the **dc_sort_fb** routine is to store the data in a *FastBus crate* and *slot* basis.
By decoding the raw *dc_channel_number* and *dc_ok* information and using the *crate_id* and *slot_id* tables the *fb_ok* crate and slot 2–D array is created. Then the **dc_hbook** routine is called.

dc_hbook: The purpose of **dc_hbook** is to create a paw directory if *fb_ok(crate,slot)* was true. The name of this paw directory is **CAL_num** (num = 100 × crate - 1) + slot). In each directory 192 1–D histograms are created corresponding to the 96 addresses in a given slot times 2 for the short and the long pulses. The names of these histograms are *tdc i* (*i* goes from 1 to 192).

By reading the raw *dc_channel_number* again, and using the raw event information another crate and slot dependant array is created: *fb_nchannels*. Then the tdc value and the corresponding *FastBus* channel arrays are created, these arrays: *fb_tdc_value* and *fb_chan_wid* are 3–D (crate, slot and hit) arrays, they are saved in the dc_evnt and dc_evnt2 common blocks in the dc_evnt.inc include file.
When the loop is over and all events in the given analyzed sector are done, **dc_hfill** is called.

dc_hfill: The purpose of *dc_hfill* is to fill the proper histogram (2048 bins plots) in the proper paw directory for all events of the curent analyzed drift chamber sector.

A bunch of histograms have been created and filled in several paw directories, the real calibration procedure can start.
A loop over all the crate and slot is done in order to calibrate all drift chamber electronic channels. If *fb_ok(crate,slot)* is true the **dc_calib_tdc** routine is called.

dc_calib_tdc: This routine is designed to calibrate one crate and slot set of 192 channel at the time. All channels has already been histogramed into corresponding 2048 bins plots (cf **dc_hfill**). Then one channel at a time the peaks are searched. The called procedure is **peak_search**. The number of peak to search for *num_peaks* is defined via a tcl variable.

peak_search: In this procedure, the histograms are loaded into a vector. Then the program steps up along the values finding those histogram channels with more counts in them than a specific threshold *tdc_threshold*. If the value in the channel is above this threshold this location is marked as a peak. Then the program skips a number of channels and starts looking for the next peak. This routine returns a number of informations.

- The location of these peaks.
- The number of peaks found.
- An error flag indicating whether the routine has found the right number of peaks or too many or too few.

The position of the peaks is stored in the peak_search_re common block of the peak_search_stuff.inc include file.

dc_calib_tdc routine takes control again. If the correct number of peaks has been found it proceeds normally. If too few have been found, the programs looks for peaks again, but in decreasing order. If too few peaks have been found it was due to the fact that the start of the tdc was to close to the first stop. By decreasing that order, we found that it is possible to associate correctly the true time peaks with the TDC channels. The subroutine used for this is **peak_search_few**.
On the other hand if the program finds too many peaks. It runs again through **peak_search**, but increases the threshold by a value of *tdc_thres_var* (this procedure is iterated 5 times at maximum).

Once this process has been completed for all channels, the program matches each found peak with the appropriate pulse corresponding to the **in_pulsing** and **out_pulsing**. Then for each channel, the position of the incoming pulse and the outgoing pulse are used to calculate the mean sum and the mean difference between these positions (which gave us respectively the cable delay and the length of the 17^{th} pair cable).

These values are saved for each channel in the dc_calib_tdc_re common block in the dc_calib_tdc_stuff.inc include file. An additional information is passed through the user_erun routine via the dc_calib_tdc_in common block of the same include file: The status word that indicates if the peak search procedure was sucessful and how.
This information is also dumped into the *Extracted parameters* ntuple which contains 5 parameters tagged as *channel, peak_out, peak_in, length* and *delay*. This ntuple is also in the **CAL_num** paw directory under the identifier 1000.

A peak searching report is also given on the *cal_num_br.report* or *cal_num.report* file depending on the choosen option. The brief output **cal_num_br.report** is only a summary of the pathologies found in the num channels. The full output **cal_num.report** is a summary of each peak position (including the pathologies) and a summary of the extracted peak_out, peak_in, mean_diff, mean_sum and status for all channnels of the current crate and slot analyzed. All these files are located in the DCCALIB's REPORT directory (cf chapter 2).

Back to the **user_erun** routine, depending on the *TCL* variable choice, it's possible either to produce a reference file *creat_ref TCL* for further analysis or a residual analysis *resid_ana TCL* if a reference file already exist. **It's not possible to combine the creat_ref and resid_ana TCL parameters.**

To produce a reference file, the good channel defined as 2 peaks found in the *dc_calib_tdc* routine and 17^{th} pair wire length (so called the difference in the previous description) in a good range (50 to 75 ns) are averaged per group of 8 wires. The electronic channel, the average difference, the average sum and the status are then dump into the **DCCR** bos bank **IW** array's.

If the residual analysis was choosen, the **residual_ana** routine is called.

residual_ana: The purpose of this routine is to made a residual analysis of the difference between the calculated new values and the old one stored in the **DCCR** bos banks for the mean_diff and the mean_sum for each channel of a given crate and slot. The old stored parameter are retrived using the **dc_read_dcc** procedure.

dc_read_dcc: This routine open a third kind of *BosFPack* file: the old reference file and use a thirt BOS array: the **KW** array. For the current crate and slot analysed, this routine returne the values of the calble mean_diff and mean_sum together with the associated status for each of the 192 channels.

Then the **residual_ana** routine take the values to build the difference and made the residual analysis on both parameters using the **hfithn** *hbook* procedure. The fit is performed on the histograms *2000* and *2001* located in the **CAL_num** paw directory; the results of the fit are saved with the histograms. An algorythme is then used to chose if the new value for the mean_diff and mean_sum are valid or not.

- If the difference between the old and the new values are out of the range of 3 standard deviation from the calculated mean value; the new values are taken.
- If the deviation is more important, the values are not modified the old ones are kept as new calibration values.
- The status of old and new calibration parameters are also check in order to take the proper parameters.

A report of the fitting parameters and the modified mean_diff and mean_sum is given in the **res_num.report** file located on the DCCALIB'S REPORT directory (see chapter 2) if the *brief_out TCL* variable is set to false.

user_erun routine takes control again, the residual analysis resulting electronic channel, mean_diff, mean_sum and status are then dump into the **DCCR** bos bank **IW** array's (that's why we have to use 2 different arrays for the reference calibration parameters).

The parameter we are intersted on for RECSIS is the so called **cable delay** which is the sum of the mean_diff and the mean_sum determined by the **user_erun** routine. The last procedure of the **user_erun** is to fill the *t_del, t_diff* and *t_sta* 3–D (sect,layer,wire) arrays with the proper information in order to pass them to the **user_last** routine. This is done through the dc_res_re and dc_res_in common blocks in the dc_res.inc include file.

4.1.4 user_last

The purpose of that routine is to create the calibration bank in order to fill the calibration file **dc_calib_s*i*.bfp** (i = *sect_ana TCL*).
The cable length extracted in the calibration routine are not the exact one needed by RECSIS. Some correction has to be done before filling the **DDLY** bos bank with the proper information. Two additionnal delays has to be taken into acount.

- An overall pulser offset the *t_off_pulser TCL* variable set to 3600 ns. in the code.
- A region dependent offset depending on the *FastBus cable length* named *t_off* in the code and set respectivly to 500, 1900 and 2900 ns for the Region I, II and III.

The cable length is corrected for each region on the drift chamber to take into acount these additionnal cable length:

$$new\_delay = delay + t\_off - t\_off\_pulser$$

Based on the each wire status, when a calibration wasn't sucessful even after the residual analysis procedure, it is possible in this code to use predefine values of the delays in order to fill the bosbanks (in that case the status is kept as it was in order for RECSIS to know wire is somehow wrong). The expected value for each sector are kept in the 6–D arrays *t_expi* (i = 1 ... 6) and are buried in the code.

The other purpose of that routine is to modify the status of each wire, in case of pathologies detected by **PDU**, by reading the **PDU** output file *badwire.dat* located in the INPUT directory. This is the purpose of the **dc_read_bad** routine. If a wire is tag as pathologic by PDU, the actual status word is modify to be

$$new\_status = 10 \times pdu\_status + cal\_status$$

When all that is done, the **HCAL** Header CALibration bank and the **DDLY** Drift chamber DeLaY bank for the analysed sector are created and filled using the BOS **JW** array.

In order to quickly check the parameters, a 5–D **ntuple** is also created (*layer*, *wire*, *delay*, *status* and *length*). This ntuple sits in the paw directory **sect*i*.rzn** (i = *sect_ana TCL var.*)

scratch file tagged as *Delay Sector sect_ana* (The ntuple's identifier is 300 + *sect_ana*).

The single sector pass of the calibration code is completed.

4.2 Full Drift Chamber analysis

A few other routines are provided to make the full cable delay calibration analysis. The filling of the *geometry* and *Time to Distance* calibration banks defined in the section 3.3. This analysis is done by using the *global_cal TCL* variable.

4.2.1 user_init

The only action taken, if **global_cal** is true is the opening of **dc_calib.bfp** file. This file is define as unit 3, and will contain the external bos bank information stored in the **JW** array. This file will be used to fill the MAPMANAGER.

A new feature has been added in order to fill only the Time–to–Distance table parameters or the Geometry mis–position and mis–orientation baks (respectively **only_xvst** and **only_geom**). **These options don't allows to fill the Delay Calibration Bank.**
If *only_xvst TCL* var. is true, the output file is **dc_xvst.bfp**, if *only_geom TCL* var. is true the output file is **dc_geom.bfp**, if both *TCL* vars. are true the output file is **dc_xvst_geom.bfp**. This file is define as unit 3, and will contain the external bos bank information stored in the **JW** array and will be used to fill the MAPMANAGER.

4.2.2 user_brun

The **Geometry** and **Time–to–Distance** calibration are done in this routine by calling **geom** and **dc_xvst_write** if the *only_xvst* and *only_geom* parameters are true.

geom: For initialization purpose, the **geom_write_ini** routine is called. The **geom** procedure then reads the *chamber.dat* and *torus.dat* file located in the INPUT directory and fill the *xt, st, xpos* and *spos* arrays in order to fill the geometry calibration bannks. This is done in the **geom_write_bos**.

geom_write_ini: The purpose of this procedure is to read the geometry ddl (**geom.ddl**) file using **txt2bos**. The geometry bos banks are then known by the system and the **HCAL** Heater CALibration bank is created (**JW** array).

geom_write_bos: This procedure simply dump the *xt, st, xpos* and *spos* arrays in the geometry bos banks **TCSB** and **DGEO** using the (**JW** array).
The Geometry bos banks are then write to the calibration file using the FWBOS Fpack routine.

dc_xvst_write: For initialization purpose this routine calls **dc_xvst_write_header** and then to file the calibration banks ,the **dc_xvst_write_r*i*** (i = 1 ...3) are called.

dc_xvst_write_header: The purpose of that routine is to create the **DCVH** Time to distance Heater calibration bank (**JW** array).

dc_xvst_write_r*i* (*i = 1 ... 3*)**:** These 3 routines are similare. The *dc_xvst_ri.dat* input files (located in the INPUT directory) are read and then the contents of the file are dump into the corresponding bos banks **DCV1**, **DCV2** and **DCV3** (**JW** array).
The Time–to–Distance bos banks are then write to the calibration file using the FWBOS Fpack routine.

4.2.3 user_last

Controled by the *global_cal TCL* variable nothing is done in the **user_evnt** and **user_erun** procedure. The only purpose of the **user_last** is to combine the results of the sector by sector analysis (descibed in the previous sections). This is done in the **combine** routine.

combine: In this routine, the 6 external calibration file **dc_calib_s*i*.bfp** (*i = 1 ... 6*) are read and dump into the *adr, del* and *stat* 2–D arrays (sector, wire_id) in order to be pass to the user_last routine. This is done through the whole_ch_re and whole_ch_in common blocks in the whole_chamber.inc include file. (The **KW** array is used to avoid any mixing between the bos arrays.)

The user_last routine takes control again and then create the **DDLY** banks for the 6 sectors and fill the **JW** array with the *adr, del* and *stat* for the 6 sectors. This information will be aviable in the **dc_calib.bfp** BosFPack file.

In order to quickly check the parameters, a 5–D **ntuple** is also created (*sector*, *layer*, *wire*, *delay* and *status*). This ntuple is in the paw directory of **global.rzn** scratch file tagged as *Delays and Status Full system* (The ntuple's identifier is 1000).

5 Steps to Achieve a Full Calibration

In order to run the calibration code. One has to log into the DCCALIB acount and then do the following:

dccalib @ jlabs2> **cd RECCAL**
dccalib @ jlabs2> **source .CALcshrc**
dccalib @ jlabs2> **cd user**

5.1 Sector by Sector Analysis

One has to edit the tcl initialisation files in order to modify the name of the input file and set the proper analysis option.

dccalib @ jlabs2> **emacs init_cal_s*i*.tcl** (*i = 1 ... 6*).

Modify the *name of the input file* **inputfile /net/farms0/work/ ...**
Chose the *reference creation option* by setting **set creat_ref -1** if this is a first path of the analysis.
Chose the *residual analysis option* by setting **set resid_ana -1** if a reference file already exist.

Then one can run the calibration code. *The next procedure has to be done 6 times (1 for each sector).* A first step is to clean up the scratch disk area and the OUTPUT directory using the *pre_run shell.*

dccalib @ jlabs2> **pre_run_s*i*.sh** (*i = sector number*).
dccalib @ jlabs2> **../bin/SunOS/user -t init_cal_s*i*** (*i = sector number*).

If this was a first calibration analysis (reference creation), the previous procedure has to be done one more time setting the **resid_ana TCL var.** to true and the **creat_ref TCL var.** to false.

At this time we have on the scratch area 6 *hbook files* **sect*i*.rzn** (*i = 1 ... 6*). The contents of these file can be check using the kumac files provided in the KUMAC directory.

Two *BosFPack* files which contain the **DDLY** and **DCCR** bos banks are also produced in the OUTPUT directory, respectively **dc_calib_s*i*.bfp** and **calib_out_s*i*.bfp** (*i = 1 ... 6*).

5.2 Global Analysis

We are now in our way to produce the global calibration file. One has to edit the tcl proper initialisation files and modify the name of the input file and set the proper option.

dccalib @ jlabs2> **emacs init_cal_global.tcl.**

Modify the *name of the input file* **inputfile /net/farms0/work/ ...**
Choose the *global analysis option by* by setting **set global_ana -1.**

To modify the *Geometry* or *Time–to–Distance* INPUT parameter, we have to edit the data

input files and modify the proper parameter. (I assume the bookkeeping of the modification is maintain by the calibration responsible person)

dccalib @ jlabs2> **emacs ../../INPUT/chambre.dat.**
dccalib @ jlabs2> **emacs ../../INPUT/torus.dat.**
dccalib @ jlabs2> **emacs ../../INPUT/dc_xvst_r1.dat.**
dccalib @ jlabs2> **emacs ../../INPUT/dc_xvst_r2.dat.**
dccalib @ jlabs2> **emacs ../../INPUT/dc_xvst_r3.dat.**

Then one can run the calibration code a last time to produce the global calibration file. A first step is to clean up the scratch disk area and the OUTPUT directory using the *pre_run shell.*

dccalib @ jlabs2> **pre_run_global.sh**
dccalib @ jlabs2> **../bin/SunOS/user -t init_cal_global**

At this time we have on the scratch area one *hbook files* **global.rzn.** The contents of these file can be check using the kumac files provided in the KUMAC directory.

Two *BosFPack* files which contain the **DDLY** and **DCCR** bos banks are also produced in the OUTPUT directory, respectively **dc_calib.bfp** and **calib_out.bfp.**

5.3 Geometry and Time–to–Distance Bank filling

A new option allows to fill only the Geometry and Time–to–Distance banks.

dccalib @ jlabs2> **emacs init_xvst_geom.tcl.**

Choose the *Time–to–Distance option by* by setting **set only_xvst -1.**
Choose the *geometry option by* by setting **set only_geom -1.**

To modify the *Geometry* or *Time–to–Distance* INPUT parameter, we have to edit the data input files and modify the proper parameter. (I assume the bookkeeping of the modification is maintain by the calibration responsible person)

dccalib @ jlabs2> **emacs ../../INPUT/chambre.dat.**
dccalib @ jlabs2> **emacs ../../INPUT/torus.dat.**
dccalib @ jlabs2> **emacs ../../INPUT/dc_xvst_r1.dat.**
dccalib @ jlabs2> **emacs ../../INPUT/dc_xvst_r2.dat.**
dccalib @ jlabs2> **emacs ../../INPUT/dc_xvst_r3.dat.**

Then one can run the calibration code a last time to produce a Geometry and/or Time–to–Distance calibration file. A first step is to clean up the OUTPUT directory using the *pre_run*

shell.

dccalib @ jlabs2> **pre_run_xvst_geom.sh**
dccalib @ jlabs2> **../bin/SunOS/user -t init_xvst_geom**

At this time we have produced the *BosFPack* files in the OUTPUT directory **dc_geom.bfp**, **dc_xvst.bfp** or **dc_geom_xvst.bfp** which contains the **DCV1**, **DCV2**, **DCV3**, **GEOM** and **TCSB** bos banks.

5.4 Feeding the MapManager

The last step of the calibration procedure is to fill the MAPMANAGER. An action is required for each type of calibration parameter.

5.4.1 Using the Global dc_calib.bfp file

The first step is to copy the **dc_calib.bfp** BosFPack file in the **$CLAS_PARMS** area.

dccalib @ jlabs2> **cp dc_calib.bfp $CLAS_PARMS/dc_calib.bfp**
dccalib @ jlabs2> **cd $CLAS_PARMS**

To fill the Mapmanager with the t_0:
dccalib @ jlabs2> **fill_t0 -m<DC_TDLY.map>** -r<*run*> **-i<dc_calib.bfp>**

To fill the Mapmanager with the status:
dccalib @ jlabs2> **fill_stat -m<DC_STATUS.map>** -r<*run*> **-i<dc_calib.bfp>**

To fill the Mapmanager with the geometry parameters:
dccalib @ jlabs2> **fill_geom -m<DC_GEOM.map>** -r<*run*> **-i<dc_calib.bfp>**

To fill the Mapmanager with the Time–to–Distance parameters:
dccalib @ jlabs2> **fill_xvst -m<DC_TIMEDIST.map>** -r<*run*> **-i<dc_calib.bfp>**

In the above *run* refers to the run_number for whom the analysis has been done (it could be different for each type of calibration constant).

5.4.2 Partial Feeding

As we have the oportunity to produce several type of calibration **BosFPack** files, it is also possible to feed the MAPMANAGER with these files but to avoid a proliferation of files in

the $CLAS_PARMS area, it will be better to keep the files in the OUTPUT directory of the DCCALIB account.

Then, to produce a new set of *geometry parameters* in the DC_GEOM.map line of the MAPMANAGER it is possible to use again the **fill_geom** utility.

dccalib @ jlabs2> **cd**
dccalib @ jlabs2> **cd OUTPUT**
dccalib @ jlabs2> **fill_geom -m<DC_GEOM.map> -r<*run*> -i<dc_geom.bfp>**
or
dccalib @ jlabs2> **fill_geom -m<DC_GEOM.map> -r<*run*> -i<dc_xvst_geom.bfp>**

To produce a new set of *Time–to–Distance parameters* in the DC_TIMEDIST.map line of the MAPMANAGER it is possible to use again the **fill_xvst** utility.

dccalib @ jlabs2> **cd**
dccalib @ jlabs2> **cd OUTPUT**
dccalib @ jlabs2> **fill_xvst -m<DC_TIMEDIST.map> -r<*run*> -i<dc_xvst.bfp>**
or
dccalib @ jlabs2> **fill_xvst -m<DC_TIMEDIST.map> -r<*run*> -i<dc_xvst_geom.bfp>**

It is also possible to modify the *Time Delay & Status* for a given sector by modifying the DC_TDLY.map or DC_STATUS.map of the MAPMANAGER using the sector dependant **dc_calib_s*i*.bfp** file ($i = 1 \ldots 6$).

dccalib @ jlabs2> **cd**
dccalib @ jlabs2> **cd OUTPUT**
dccalib @ jlabs2> **fill_t0 -m<DC_TDLY.map> -r<*run*> -i<dc_calib_s*i*.bfp>**
or for the status
dccalib @ jlabs2> **fill_status -m<DC_STATUS.map> -r<*run*> -i<dc_calib_s*i*.bfp>**

5.5 General Comments

If someone modify something, somewhere in the USER, RECUTL or RECSIS code (or also in the INCLUDE area), the best way to produce the new libraries and executables is:

dccalib @ jlabs2> **cd**
dccalib @ jlabs2> **cd RECCAL**
dccalib @ jlabs2> **make -f $CLAS_CMS/clas.mk user**

This will rebuild the libraries of the modified packages and also all the Fortran files which include a modified include file, and the **user** exectutable.

The libraries and the exec are located in the bin/SunOS subdirectory of RECCAL (*and not in the* DCCALIB*'s bin directory !*).

6 Visualization of the Delay calibration Analysis

A list of kumac files (see section 2.5) provide an easy way to visualize the Time Delay calibration results. In This chapter, I will illustrated the use of these *kumac* files by using the *hbook* files **secti.rzn** ($i = 1 \ldots 6$) and **global.rzn** locted in the *scratch area.*
The report files (see section 2.4) is provided to have a quick check of the result of the *channel by channel calibration* and *residual analysis.* The contents of these files will also be discussed in this chapter.

As an example I will use the results of the september **dc_pulser_run 5952 analysis.**

6.1 Contents of the hbook files

Let me start with a brief description of the contents of the *sector by sector analysis* and the *global analysis* hbook; this will help the understanding of the next sections.

6.1.1 secti.rzn file

In the *secti.rzn* file ($i = 1 \ldots 6$), the LUN1 root directory contains **2 Ntuples**. (In all the following **i** is the analysed sector number)

- The *3–tuple* Channel Fired Si (Id 10i) **contents: (run, event, wire_fired)**
- The *5–tuple* Delay Secti (Id 30i) **contents: (layer, wire, status, delay, length)**

In the LUN1 directory, there is also a subdirectory for each crate and slot involved in the curent analysed sector **i** (this represents 30 subdirectories). The name of these directories are **CAL_num** where **num** is a 3–digit number equal to *(crate - 1)* $\times$ *10 + slot.*
Each *secti.rzn* file then contains 30 directory filled with:

- 192 *1–D Histogram* labeled **Channel j** (Id j) representing the contents of the **j–electronic channel** ($j = 1 \ldots 192$)
- The *5–tuple* Extracted parameters (Id 1000) **contents: (channel, peak_out, peak_in, length, delay**. These parameters are the results of the peak_search procedure.
- The *1–D Histogram* labeled Δ diff (Id 2000) representing the 17^{th} pair length residual analysis.
- The *1–D Histogram* labeled Δ sum (Id 2001) representing the cable delay residual analysis.

6.1.2 global.rzn file

In the *global.rzn*, the LUN1 root directory contains only **1 Ntuple** which is the result of the complete analysis. There is no paw subdirectory in this file.

The *5–tuple* is named **Delays and Status Full system**, has the ID 1000 and contains (Sector, Layer, Wire, Status, T_delay).

6.2 General purpose spectra

The 1st spectra to look at is the number of channel fired per event, to check the integrity of the pulser run. This is done by using the *general.kumac* file. The fig. 3 is sample of the *general.ps* file.

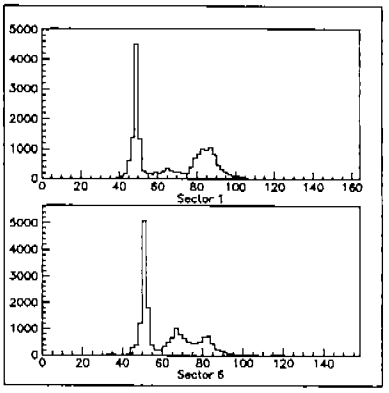

Figure 3: Number of Channel Fired per Event (sector 1 and 6)

The number of channel should be a thin peak. On these two spectra we saw some noise.

6.3 Crate and Slot basis analysis

6.3.1 Raw TDC

It is possible to take a look at all the *raw TDC* in a given sector using the *tdc_si.kumac* file. The corresponding *.ps* file is huge. One histogram is created for each elecronic channel of a given Crate an slot involved in the **sector i**. The fig. 4 is a sample of some tdc values from the sector 6 analysis.

In these plot we expected 2 peaks corresponding to the incomming pulse and the outgoing pulse. The difference between the 2 peaks position should be almost the same in every crate and slot picture.
In this sector 6 sample we can see some good channel in region II Axial (crate 10 and slots 5 and 9 *the channel 18 seems bad*), in the region II Stereo (crate 10 slot 19) the peak positions seems good too.

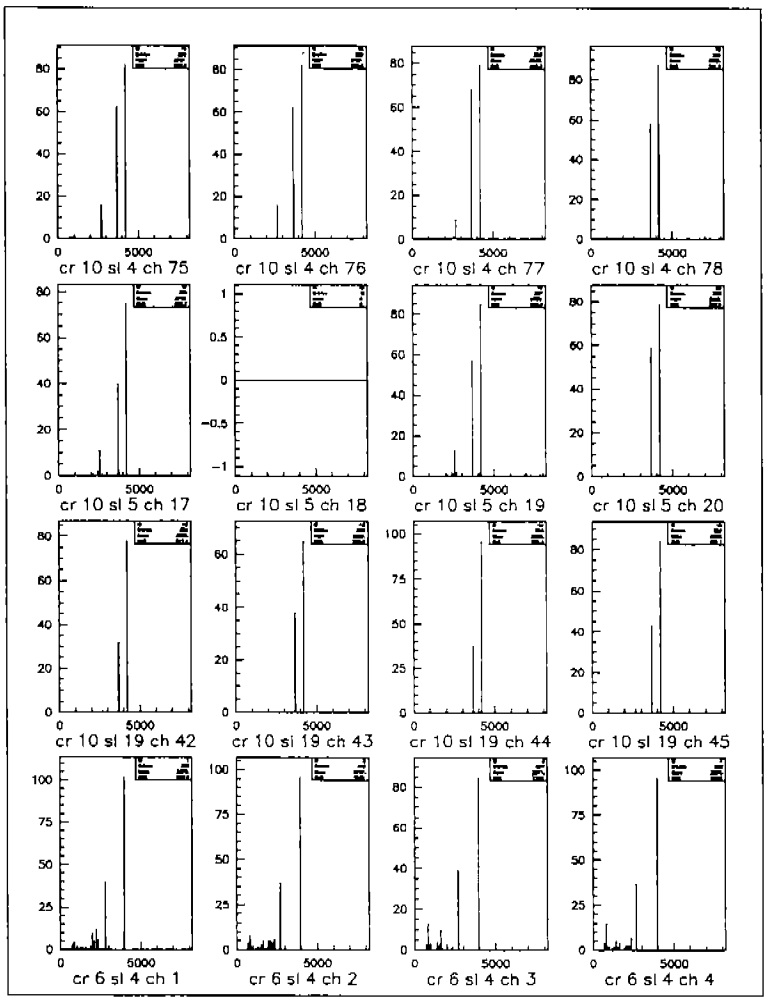

Figure 4: Sample from the tdc_s6.kumac file

In the region I a real problem occurs (and has been solve now) there was no outgoing pulse in the spectrum, it's then **not posible to calibrate the region I Drift chamber TDC's.**

6.3.2 Extracted Lenght and Delay

The 2 relevant parameters of the calibration are the length of the 17^{th} pair cable (the mean difference between the incomming and outgoing pulse peak positions) and the Delay of the analysed channel (the mean sum between the incomming and outgoing pulses peak positions). The *del_si.kumac* file are provided to take a look at this information in a crate and slot basis.

The fig. 5 is a sample of results for the sector 6 region I and II.

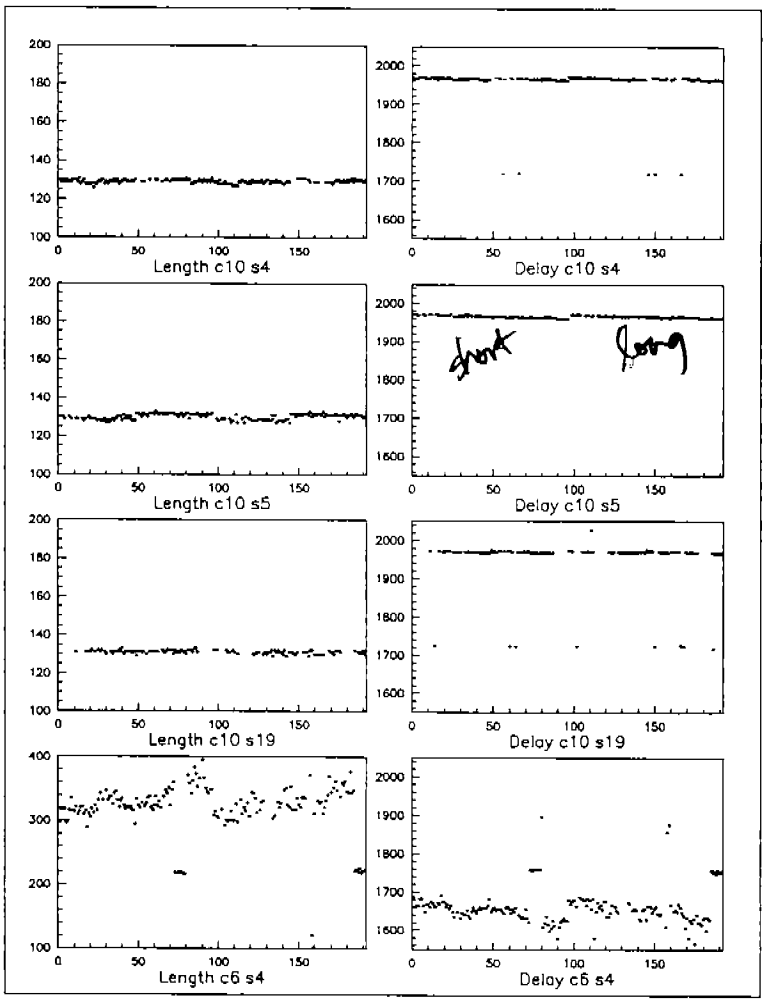

Figure 5: Sample from the del_s6.kumac file

I pick again the crate 10, slot 4 and 5 (sector 6 region II Axial) and slot 19 (sector 6 region II stereo) and the crate 6 slot 4 (sector 6 region I) as an example. In these 2–D representation the extracted lenght (left) and delay are ploted versus the electronic channel number in each crate and slot.

In the region II the results are fine even if some channels are missing (meaning some individuals problems). The extracted cable lenght is close to the expected value $\approx$ 75.ns (the times in the plot are in TDC counts). In the region I the expected value should be slightly lower and we observe a fuzzy 150.ns cloud. This reflect the fact that the program have found 2 peaks in the region I TDC spectra which are not related to the in and out pulses.

That leads me to make a cut on the extarcted length in the code and tag as bad any channel with a length smaller than 40 ns and greater than 80 ns. (this information is kept in the channel status word)

Concerning the Delay, we can notice that within a given crate and slot the values are quite close when there is no problem with the out pulse.

6.3.3 Residual Analysis

The last crate and slot dependant interseting results are the Residual analysis one. The *res_si.kumac* file are provided to take a look at this information in a crate and slot basis.

The fig. 6 is a sample of results for the sector 6 region I and II.

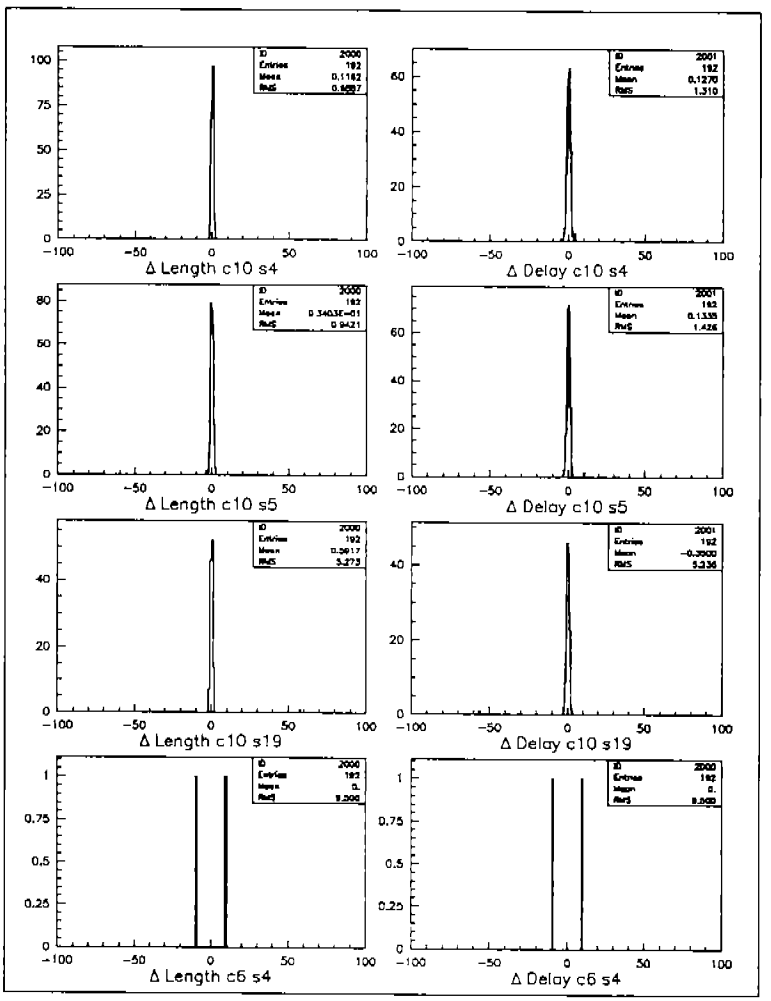

Figure 6: Sample from the res_s6.kumac file

The crate 10, slot 4 and 5 (sector 6 region II Axial) and slot 19 (sector 6 region II stereo) and the crate 6 slot 4 (sector 6 region I) are again the choosen example.
The residual analysis is successful for the lenght and delays in region II where the data are good, but for region I the discrepency betweem the reference data and the current analysed

data is so big that the residual analysis has no meaning.

6.3.4 Comments

Taking a quick look at the **dc pulser run 5952** using this calibration and especially this crate and slot based analyis allows us to find some mis–functionning of the pulser. Especially the **region I** polarity mismatch between the pulse to the STB board and the STB SIP assembly. Some other crate and slot has to be investigated in order to find other pathologies.

Unfortunately for this run only 2 region were pulsed for each sector. It will be usefull to have a general check of an entire pulser run as soon as possible.

Concerning the crate and slot analyis there is nothing more to say. I'm confident that more spectra will be useful and I encourage anyone to play with the code and add as many plot as needed.

6.4 Sector, Layer and wire basis analysis

The following section describes the other *calibration kumac files* but for hardware debugging purpose I will suggest to use the previous kumac files since the real cabling dependance is on a crate and slot basis and not on a sector, layer and wire basis.

6.4.1 Status

In the sector, layer and wire basis analysis, the status of the analysis are always shown with the checked parameter. Let me here summarize the meaning of these status.

- 0 Wire corresponding to a good electronic channel analysis.
- $1 \rightarrow 4$ Wire corresponding to a noisy channel but the analysis was succesful.
- 5 Channel to noisy even after noise supression.
- 6 The code was not able to find 1 peak (problem either with in or out pulsing)
- 7 No peak at all (channel disconected)
- 8 Reference, good channel
- 9 Reference, bad single channel
- 10 Reference, bad group of 8 channel

The reference part means the reference value has been taken instead of the calibartion analysis one.

6.4.2 Cable Lenght

As we know the corespondance between the *crate, slot, channel* and *sector, layer, wire* it's then possible to plot the already shown parameters in a layer and wire basis for a given sector. The *tdif_si.kumac* file are provided to take a look at the extracted cable lenght in a layer and wire basis.

The fig. 7 is a sample of results for the sector 6.

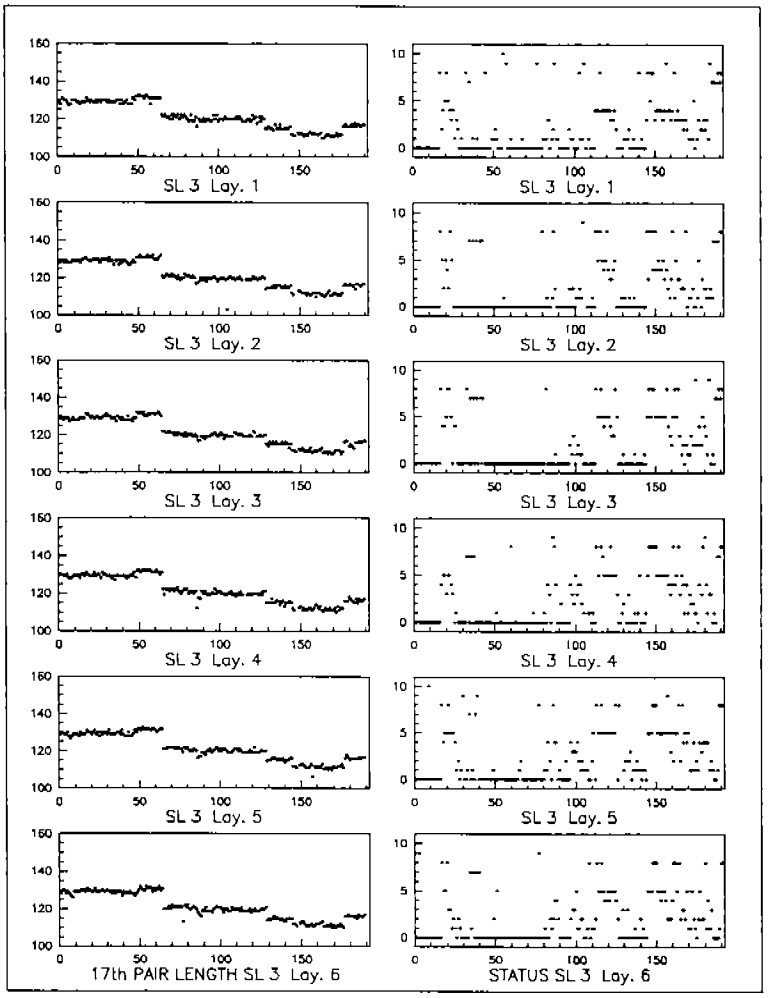

Figure 7: Sample from the tdif_s6.kumac file

This figure shows the 17^{th} *pair cable length* and the corresponding *status* for the 6 layers of the 3^{rd} Super Layer of the drift chambers (the X axix is the wire number).
In the plot we show the expected backward, forward dependance of the 17^{th} pair cable length.

6.4.3 Cable Delays

The same kind of plot is avaiable for the t_0 information. The *tdel_si.kumac* files are provided to take a look at the extracted cable delay in a layer and wire basis.

The fig. 8 is a sample of results for the sector 6.

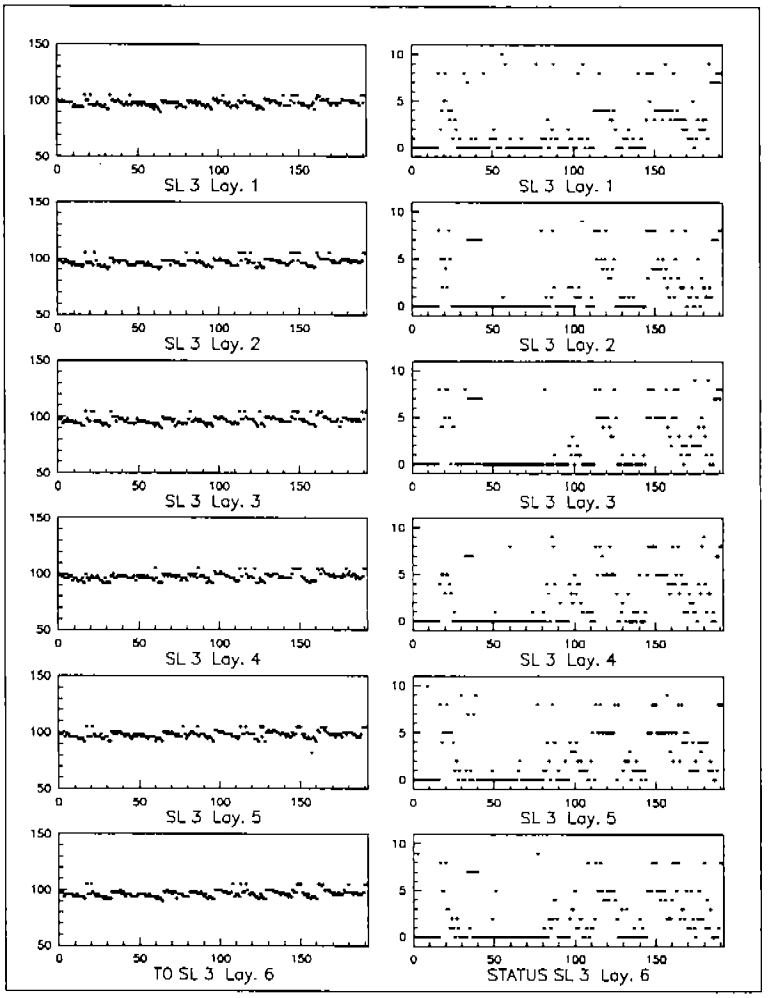

Figure 8: Sample from the tdel_s6.kumac file

This figure shows the T_0 and the corresponding *status* for the 6 layers of the 3rd Super Layer of the drift chambers (the X axix is the wire number).
In this plot a non physical dependance is shown. That means that we have to modify the T_0 in order to take into account the **STB board** cable dependance.

The T_0 differs from Super–Layer to Super–Layer and Sector to Sector in the Table 3 the expected values are summerised.

When for a given wire the analysis failed the T_0 is taken in the previous table. The fig. 9

	SL-1	SL-2	SL-3	SL-4	SL-5	SL-6
1	-1500	-1495	90	85	1105	1090
2	-1500	-1495	80	95	1095	1105
3	-1500	-1495	-15	-5	1000	1015
4	-1500	-1495	-20	-5	1000	1000
5	-1500	-1495	-60	-10	1000	1010
6	-1500	-1495	105	110	1100	1100

Table 3: Default T_0 values

is a sample of result for the sector 6 Region I and III where the calibration analysis failed.

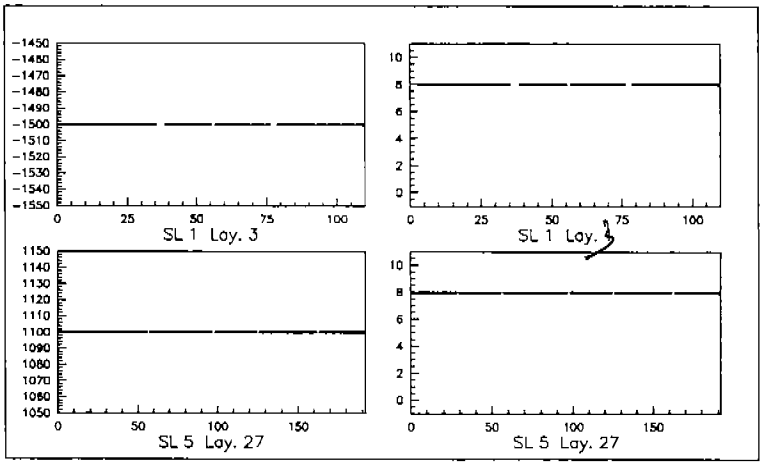

Figure 9: Sample from the tdel_s6.kumac file

For the Super Layer 1 and 5 the calibration analysis failed (the status is 8), in this case the reference values for the T_0 are taken into account (this number is the same for an entire super layer of a given drift chamber sector).

6.5 Complete analysis

The *global.rzn* file is a small file (compared to the sector ones) with allows a quick look at all the T_0. Fore exemple the **g_del.kumac** file plots T_0 for each Super–Layer in a sector basis.

This file is only useful in order to test the pathologies of the analysis, for more detailed result it's better to refer to the previous parts.

6.6 The report files

As I have already mentionned, the *cal_***.report*, *cal_br_***.report*, and *res_***.report* files are an easy way to check the analysis procedure in a crate and slot basis.

6.6.1 cal_br_***.report

This file contains the problems encontred during the peak search analysis.
This is a few line extracted from the *cal_905_br.report* file. (**crate 10 slot 5**: Region 2 Axial Sector 6)

```
Problem with channel:    138 ----> too few peaks
Channel   138 no peaks at all

Problem with channel:    148 ----> too many peaks
Problem with channel:    148 ----> fixed after    2 iteration (s)

Problem with channel:    151 ----> too many peaks
Problem with channel:    151 ----> fixed after    1 iteration (s)

Problem with channel:    191 ----> too few peaks
Warning on channel   191  problem either with pulsing in or out
```

6.6.2 cal_***.report

This file contains, in addition to the previous information more details about the calibration such as the extracted position of the peaks.
The next lines are extracted from the *cal_905.report*

```
TDC_Channel:   147 peak position 3667.0 4195.0

Problem with channel:    148 ----> too many peaks
Problem with channel:    148 ----> fixed after    2 iteration (s)
TDC_Channel:   148 peak position 3671.0 4191.0
TDC_Channel:   149 peak position 3667.0 4195.0
TDC_Channel:   150 peak position 3667.0 4191.0

Problem with channel:    151 ----> too many peaks
Problem with channel:    151 ----> fixed after    1 iteration (s)
TDC_Channel:   151 peak position 3663.0 4191.0
TDC_Channel:   152 peak position 3667.0 4191.0
TDC_Channel:   184 peak position 3663.0 4183.0
TDC_Channel:   185 peak position 3663.0 4187.0
```

```
TDC_Channel:   186 peak position 3655.0 4183.0
TDC_Channel:   187 peak position 3659.0 4183.0
TDC_Channel:   188 peak position 3659.0 4183.0
TDC_Channel:   189 peak position 3659.0 4183.0
TDC_Channel:   190 peak position 3663.0 4183.0

Problem with channel:    191 ----> too few peaks
Warning on channel   191  problem either with pulsing in or out
TDC_Channel:   191 peak position    0.0 4187.0
TDC_Channel:   192 peak position 3663.0 4183.0
```

Then a full set of information is printed in the full calibration report file.

```
                ********   CALIBRATION REPORT   ********

Channel:     1  peak: out  1840.0  in  2100.0  diff   130.0  sum  1970.0  stat  101
Channel:     2  peak: out  1840.0  in  2100.0  diff   130.0  sum  1970.0  stat -200
Channel:     3  peak: out  1840.0  in  2100.0  diff   130.0  sum  1970.0  stat  100

Channel:   147  peak: out  1834.0  in  2098.0  diff   132.0  sum  1966.0  stat  101
Channel:   148  peak: out  1836.0  in  2096.0  diff   130.0  sum  1966.0  stat  102
Channel:   149  peak: out  1834.0  in  2098.0  diff   132.0  sum  1966.0  stat  100
Channel:   150  peak: out  1834.0  in  2096.0  diff   131.0  sum  1965.0  stat  100
Channel:   151  peak: out  1832.0  in  2096.0  diff   132.0  sum  1964.0  stat  101
Channel:   152  peak: out  1834.0  in  2096.0  diff   131.0  sum  1965.0  stat  100

Channel:   184  peak: out  1832.0  in  2092.0  diff   130.0  sum  1962.0  stat  100
Channel:   185  peak: out  1832.0  in  2094.0  diff   131.0  sum  1963.0  stat  100
Channel:   186  peak: out  1828.0  in  2092.0  diff   132.0  sum  1960.0  stat  100
Channel:   187  peak: out  1830.0  in  2092.0  diff   131.0  sum  1961.0  stat  100
Channel:   188  peak: out  1830.0  in  2092.0  diff   131.0  sum  1961.0  stat  100
Channel:   189  peak: out  1830.0  in  2092.0  diff   131.0  sum  1961.0  stat  100
Channel:   190  peak: out  1832.0  in  2092.0  diff   130.0  sum  1962.0  stat  100
Channel:   191  peak: out     0.0  in     0.0  diff     0.0  sum     0.0  stat -300
Channel:   192  peak: out  1832.0  in  2092.0  diff   130.0  sum  1962.0  stat  100
```

6.6.3 cal_***.report

In the *cal_***.report* after a dump of the 2 residual analysis histograms, a summary of the channel change is given.

This is again a few line extracted from the *res_905report* file.

```
              ***** ELECTRONIC MODIFICATION REPORT   *****

delta_diff mean :     0.070  sigma delta_diff :     0.867
delta_sum  mean :     0.099  sigma  delta_sum :     0.981

TDC Channel   1 diff =  130.00 Analyzed Value Taken.
                sum  = 1970.00 Analyzed Value Taken.
TDC Channel   2 diff =  130.00 Analyzed Value Taken.
                sum  = 1970.00 Analyzed Value Taken.
TDC Channel   3 diff =  130.00 Analyzed Value Taken.
                sum  = 1970.00 Analyzed Value Taken.

TDC Channel 190 diff =  130.00 Analyzed Value Taken.
                sum  = 1962.00 Analyzed Value Taken.
TDC Channel 191 diff =  130.86 value at  150.8 sigma: Previous Stored Value Taken
                sum  = 1961.43 value at 1998.6 sigma: Previous Stored Value Taken
TDC Channel 192 diff =  130.00 Analyzed Value Taken.
                sum  = 1962.00 Analyzed Value Taken.
```

7 Time–to–Distance Calibration Overview