Kloe Note 94-137

March 95

# KLOE Data Structure December 94

A. Andryakov, A. Antonelli, C. Bloise, A. Calcaterra,
G. Capon, G. Cataldi, G.Chiefari, F. Donno, E.Drago, A. Farilla,
E. Gorini, F. Grancagnolo, W. Kim, J. Lee-Franzini, S. Miscetti,
F. Pelucchi, M. Primavera, S. Wölfle

March 24, 1995

# Contents

# 1 INTRODUCTION

The aim of this document is to introduce users and code writers to the KLOE event data structure, describing the hierarchical scheme which represents different stages of the reconstruction process and the banks containing the information. The idea is to have a coherent way of storing events which is the same at each stage of the software, from Monte-Carlo and simulation analysis to "real" data. At each level of the analysis, we find a logical progression of YBOS banks from the most elementary level of the data structure to the sequentially more highly refined levels. The first and most elementary level in the topological hierarchy is the Raw Data Bank, a YBOS Bank which contains detector information as it comes out from the data acquisition system. Immediately above this is the element level. An Element Bank is a construct of the raw data which retains most of the granularity of the detector information which it represents, but uses the calibration database to correct the digitized information into real numbers corresponding to physics quantities. Elements are organized by detector reflecting that calibration is invariably done on a detector by detector basis. Above this level is the segment level. Segments are the combination of the information from one or more detector. As an example, segments may be helices in a tracking chamber or energy clusters in a calorimeter. Objects are the lowest level in this hierarchy. Objects are the items which are used by physicists for analysis and should have transparent interpretations in terms of the physics of the event. However, they are conceptually different from segments in that they are associated with a physics interpretation of the event and have physics dependent assumptions regarding their kinematic attributes. At each stage of this chain a mechanism to go back to the parent bank is available. Before a detailed discussion of this data structure, general considerations about the YBOS banks will follow.

# 2 Data Structure: Software Consideration for YBOS Banks

In order to better organize the information in the YBOS banks and get the most out of the code we cannot avoid some software considerations:

## 2.1 Optimization of CPU and Disk Space Usage

On machines with 64 bits data registers like the ones based on the Alpha chip it pays to avoid using a Fortran INTEGER*2 and C short data types. The Alpha chip issues LDx_U instructions and some bitmasking instructions to get at the data, rather than a single LDL instruction for the 32 bit datatype. Also data within structures (YBOS common blocks) should be aligned to at least 64 bit boundaries. The considerations above facilitate a better use of the CPU but penalize a lot the use of the available mass storage. In fact, data occupy double the space that they would if stored in INTEGER*4 longwords. We should decide what is most critical for KLOE. For the Cosmic Ray Stand and Test Beam 1994 the use of Alpha based and 64 bits HP machines will be limited: data coming from the DAQ and RAW data banks will be stored in INTEGER*2 words to save disk space.

## 2.2 Integer Convertions

It is also important to avoid divisions between integers and, in general, integer to floating point conversions. Integer division is achieved by converting the integer to a float and dividing and converting back to an integer. Whenever possible, integers should be recoded to a floating point data type to avoid the conversion. In fact the current Alpha CPU chip has no direct connection between its floating point unit and its integer unit, which means such conversions require a load/store operation and are thus less efficient. Information will not be compressed when dealing with Element Banks but real numbers are used where necessary.

### 2.3 Multidimensional Arrays

Multidimensional arrays should be stored following the FORTRAN 'natural' order so that they can be traversed avoiding trashing the CPU's cache. Columns in an array should be scanned first.

### 2.4 Bit Encoding

Few conventions need to be established for bit encoding. Bit encoding of 32 bit words (longwords) shall be described according the VAX representation of 32 bit integers. The lower half word is the less significant 16 bits of an integer and the upper half word is the more significant 16 bits of an integer. The least significant bit is called bit 0, while the most significant bit is called bit 31. Since all data structures employ YBOS for their representation, the interpretation of such bit encoded meanings is machine independent, regardless of the precise internal machine representation of the integer.

## 3 General rules for banks creation

In this section general rules for creating banks are given. Such rules should be strictly respected to grant coherence among banks and avoid collisions.

### 3.1 Naming conventions

Banks should be named using the following rules:

1. The first letter describes the detector part:

   A - Anticoincidence
   C - Calorimeter
   D - Drift Chamber
   I - Inner (Vertex) Chamber
   Q - Quadrupole Instrumentation
   T - Trigger

2. The last letter classifies the type of the bank:

   R - Raw
   E - Element
   S - Segment
   O - Object

3. Squeezed banks made using some specific algorithm to save space in the output file should have the same name as the originating bank but in LOWER CASE. The convention explained in point 2 should be respected. Raw banks will be squeezed and their name does not follow this rule.

## 3.2 Structure of data inside a bank

A YBOS bank is made by a YBOS header that the user can just read but not modify, and a user data block. In KLOE the user data block of a bank is logically broken in two pieces, a user header and a data body. The user header contains information about the bank itself: the size of the header MUST be the first word; the bank version number described later follows the header size; optionally the number of information stored inside the bank; other optional information. Storing the size of the header inside the bank allows users to increase the words in the header so that new code is backward compatible through the use of the bank version number. Header size and version number are integer numbers that can be eventually packed in one YBOS longword. The header block and the data body of a bank MUST be alligned to 32 bits.

## 3.3 Backward compatibility

All YBOS banks shall be numbered starting with 1 and each bank is given its own, independent version number. This version number is stored inside the bank, thereby allowing that programs capable of decoding older as well as newer banks may be written, without requiring any recompilation(s) in the older environment(s). Everytime one changes the structure of a KLOE bank by changing some values of parameters in the Include File and/or introducing new parameters, one must also increment the parameter for the version number.

```
     CODEING SUGGESTION for routines reading data from a specific bank:

C
C--Check if old version of bank structure:
        IF (IW(INDDAT+VRSDxx) .NE. VRSNxx) THEN
           CALL ERLOGR(Subnam,ERWARN,0,0,
      &       "Bank structure not up to date; "
      &      //"- bank (xxxx,n)")

C--Correct the base index to the data
C--if (size of KLOE header .ne. HDRSxx):

           IF (IW(INDDAT+HDRDxx) .NE. HDRSxx) THEN
              BASIND = INDDAT+IW(INDDAT+HDRDxx)-HDRSxx
           ELSE
              BASIND = INDDAT
           ENDIF
C
C--Version number of bank is up to date:

        ELSE
           BASIND = INDDAT
        END IF
```

### 3.4 Tabular Banks

Wherever blocks of information are repeated in the user data body of a bank a tabular structure is recommended. In a tabular structure information is stored by rows and columns. The number of columns is fixed, while the number of rows varies. A block of information is given by the data stored in a raw. In this case, the number of rows and columns is stored in the bank header while data are stored in the data body one row (block) of information after the other. The reason for storing the number of rows and columns in the bank header is code backward compatibility: a new column can be easily added to the bank and the new code, checking the version number, can still jump between rows using the number of columns as the block information size. New variables MUST be added only at the end of header or body blocks.

### 3.5 Parameters and Documentation

Banks should be described by mean of include files available to FORTRAN programs and by Latex documentation files to be put in KLOE$DOC. In each bank include files parameters which give the displacement of the various words in the bank from the main YBOS INDDAT pointer are given. These parameters must be named following the convention that the first three letters of their names MUST be the same as the bank name.

### 3.6 Bank Aliases

At the Segment level, different physics algorithms produce banks having the same internal structure but different contents. Because users may want to save the results of several algorithms in the same output file, the banks MUST differ in the name. The concept of a bank alias has been introduced in KLOE through the usage of bank names parameters. In the include file describing the bank different parameters, each referring to a different physics algorithm, may define bank names for the segment banks. The name of the parameter variable is meaningful and identifies the algorithm used to create the bank in question. All names for the same bank MUST be defined in the same include file. It is the responsability of each detector group leader to ensure no collisions among bank names. The naming convention rules for a bank MUST be respected also in this case.

### 3.7 Link Mechanisms

Links can be estabished between banks. This should be done using the YBOS bank name and number or via internal bank offsets. Work link banks (obtained by calls to BLINK) are only provvisory and have the life of a program.

## 4 Logical Record

This section is addressed to give a description of the Logical Run and Event Records for the Raw and Element Data. The following description of the banks within an event record uses a parameterisation of the Bank Type that is also described in the include file YBOS$LIBRARY:BNKTYP.INC. Furthermore, displacements are given relative to the Bank Data Index (INDDAT) returned by BLOCAT, rather than relative to the basic Bank Index (IND). Thus displacements are all relative to zero (in other words, the first word data word in the bank is located at offset 0).

## 4.1 LRID Bank - Logical Record IDentifier Bank

The LRID Bank is the first Bank in any logical record (Run or Event) and is the bank through which ANALYSIS_CONTROL recognizes the Record Type. It has the following Bank Header Characteristics:-

| | | |
|---|---|---|
| Bank Name | : | "LRID" |
| Bank Number | : | 1 |
| Bank Data Length | : | 8 |
| Bank Type | : | BNKTI4 (Integer*4) |

The Bank contains the following information:-

| Displacement (I*4) | Word Type | Description |
|---|---|---|
| 0 | I*4 | Run Number |
| 1 | I*4 | Run Type |
| 2 | I*4 | Record No. |
| 3 | I*4 | Record No. of this Type |
| 4 | I*4 | Record Type |
| 5 | I*4 | Creating Process ID |
| 6 | I*4 | Creation Date |
| 7 | I*4 | Creation Time |

1. Run Number. Run Numbers for runs writing data to tape shall be unique within each Experiment Type and shall monatonically increase.

2. Run Type. This word has the following fields:-

| | | |
|---|---|---|
| Bits | 0-11 | Run Type |
| Bits | 12-15 | Run Modifier |
| Bits | 16-19 | Spare |
| Bits | 20-23 | Spare |
| Bits | 24-31 | Experiment Type |

For KLOE the following Experiment and Run Types have been defined:-

| Experiment Type | |
|---|---|
| 0 | KLOE Physics |
| 1 | Cosmic Ray Test Stand |
| 2 | Offline Simulation |
| 3 | Test Beam |

| Run Type | |
|---|---|
| 0 | Undefined |
| 1 | Physics |
| 2 | Cosmic Ray |
| 3 | Calibration |

The include file K$INC:RUNTYP.INC contains parameters with the definitions above.

3. Record No. This field indexes each event (logical record) sequentially as it is written to tape. It is only specified for logical records written to tape and indexes records commencing at 1 within each run.

4. Record No. of this type within Run. This field indexes each logical record of a particular record type sequentially as it is written to tape. It is only specified for logical records written to tape and indexes records commencing at 1 within each run.

5. Record Type. Identifies Logical Records as Run, Events, Calibration Records, Comments etc. This word has the following fields:-

   | | | |
   |---|---|---|
   | Bits | 0-15 | Record Type |
   | Bits | 16-31 | Spare |

   The possible record types for the KLOE experiment are defined in the include files K$INC:RECTYP.INC. A C version of the same include file is available in the same area.

6. Creating Process ID. This identifies uniquely the process that was the creator of this logical record. It is made up of the following fields:-

   | | | |
   |---|---|---|
   | Bits | 0-15 | Process ID |
   | Bits | 16-23 | Release Number |
   | Bits | 24-31 | Version Number |

   Thus each process that generates an event record has assigned to it a unique Identifier, being in the range 0-65535. Additionally, each process is stamped by Version and Release Numbers in the range 0-255. The Version Number should be incremented for major changes to the software, whereas the Release Number should track such things as bug fixes etc. within a Version.

7. Creation Date. Date of creation of the record in the form:-

   $$65536*\text{Years} + 256*\text{Months} + \text{Days}$$

   Where the Years should be in 4 digit form (e.g. 1983). This format has the property that it monatonically increases with increasing date and therefore enables easy searches to be made.

8. Creation Time. Time of creation to nearest second in the form:-

   $$65536*\text{Hours} + 256*\text{Minutes} + \text{Seconds}$$

   Where a 24 hour clock should be used. This format has the property that it monatonically increases with increasing time and therefore enables easy searches to be made.

Include files describing the parameters for LRID may be found in K$INC.

### 4.2 EVCL Bank - EVent CLassification Bank

The EVCL Bank contains information from various sources that serve to classify the event. This Bank is only present in Event Logical Records and is used by Analisys_Control to classify events. The Bank has the following Bank Header Characteristics:-

| | | |
|---|---|---|
| Bank Name | : | "EVCL" |
| Bank Number | : | 1 |
| Bank Data Length | : | ? |
| Bank Type | : | BNKTI4 (Integer*4) |

The EVCL bank has not yet been defined. A non detailed proposal comes from the DAQ and Offline groups:-

| Displacement (I*4) | Word Type | Description |
|---|---|---|
| 0 | I*4 | Trigger Number |
| 1 | I*4 | Trigger Descriptor(1) |
| 2 | I*4 | Trigger Descriptor(2) |
| 3 | I*4 | Processor ID |
| 4 | I*4 | Event Builder Conditions |
| 5 | I*4 | Machine Luminosity |
| 6 | I*4 | Bunch Number |
| 7 | I*4 | Tune Shift |
| 8 | I*4 | Background |
| 9 | I*4 | Clock Source |
| 10 | I*4 | Error Summary |
| 11 | I*4 | Spare |
| 12 | I*4 | Spare |
| 13 | I*4 | Spare |
| 14 | I*4 | Spare |

## 5 Monte Carlo Banks

The Monte Carlo banks do not follow the definition given for true data banks. The Monte Carlo original information can be stored together with the reconstruction banks to evaluate performances and efficiencies of different algorithms. We propose to store the banks concerning the kinematics and the hits for each event.

The kinematical banks will have the same contents of the actual KINE and VERT bank from GEANT and all the existing software can be used to retrieve this information. The links between KINE and VERT is automathically done using GUNPAK-like software, the link between the raw or element banks and KINE is provided by banks for each subdetector (DHKA for drift chamber and ... for calorimeter).

For the HITS banks we propose to have a different bank for each subdetector containing the information on the true hit values. The link mechanism between HITS and the raw or element banks is not yet established.

## 5.1 KINE banks

There are as many KINE banks as tracks in the event; the number of the KINE bank is the same as the track number given by GEANT. Each bank contains the following information :

```
    YBOS Bank header:
    -----------------
Bank Name:          KINE
Bank Number:        Track number
Bank Data Length:
Bank Type:          BNKTR4  !  R*4
```

Each word in the bank can be located through the offsets shown below.

```
-------------------------------------------------------------------
  offset           word type         description
-------------------------------------------------------------------
  KINPPX              R*4   Particle momentum along x axis.
  KINPPY              R*4   Particle momentum along y axis.
  KINPPZ              R*4   Particle momentum along z axis.
  KINPEN              R*4   Energy of the particle.
  KINPTY              R*4   Particle type.
  KINVXO              R*4   Vertex origin number.
  KINNVX              R*4   Number of produced vertices (NVTX)
-------------------------------------------------------------------
  KINVLS              R*4   First offset to vertices number list
  ....                      ....
```

User words could be stored at the end of the vertices list, for them is not possible to define offsets as parameters because the vertices list can be different for each track. A track dependent offset can be built like this:

KINVLS+NVTX

that gives the number of user words, NUSW the remaining words in the bank are the user words obtained using the offsets:

KINVLS+NVTX+I (I=1,NUSW)

## 5.2 VERT banks

There are as many VERT banks as vertices in the event; the number of the VERT bank is the same as the vertex number given by GEANT. Each bank contains the following information :

```
    YBOS Bank header:
    -----------------
Bank Name:          VERT
```

```
Bank Number:        Vertex number
Bank Data Length:
Bank Type:          BNKTR4  !  R*4
```

Each word in the bank can be located through the offsets shown below.

```
-------------------------------------------------------------------
  offset       word type        description
-------------------------------------------------------------------
  VERXCO          R*4   X coordinate of the vertex.
  VERYCO          R*4   Y coordinate of the vertex.
  VERZCO          R*4   Z coordinate of the vertex.
  VERTOF          R*4   Vertex time of fligth.
  VERTNO          R*4   Beam track number origin of the vertex.
  VERTAR          R*4   Target track number origin of the vertex.
  VERNTR          R*4   Number of tracks generated at vertex.
-------------------------------------------------------------------
  VERTLS          R*4   First offset to tracks number list.
  ....                  ....
```

As for the KINE bank, user words could be stored at the end of the bank and retrieved with the same mechanism.

### 5.3 HITS banks

There is one HITS bank per detector. To be continued................

# 6 Detector Banks - Raw and Element Format

In this section a description of all detector banks follows for both formats, raw and element. The banks are also described by include FORTRAN files available in the general include area K$INC. For the raw banks the following is a first draft proposal. In fact, the general structure of data coming from DAQ is not yet completely defined. We would like to propose the following:

1) It is desirable that the format of the raw data coming from the online Data Aquisition System is YBOS. We are aware that this could have a (small) impact on the speed of the DAQ and on the space occupied by the raw data, but nevertheless this option should be taken seriously.

2) The raw data banks should be separated for each subdetector in TDC and ADC banks.

3) The addresses for TDC and ADC should be the electronic ones, not the geometric ones, because this will make a lot easier to correct for miscabling, which is a very likely possibility, especially at the beginning of the experiment.

If the first two requests can not be met for some reasons, another stage of processing will be necessary before the ones described here.

## 6.1 CADR Bank – Calorimeter ADc Raw data bank

The CADR bank stores the ADC values for the calorimeter. It has the following YBOS header characteristics:

```
Bank Name:          "CADR"
Bank Number:          1
Bank Data Length:   HDSIZE+NADC*NVAL      (in 4 bytes words)
Bank Type:          BNKTI4                (integer*4)
```

The Bank is structured in a header block followed by a data block. Each word in the bank can be located through the displacement pointers shown below.

```
--------------------------------------------------------------------
 Offset      Word type   Description
--------------------------------------------------------------------
 CADHDS         I*4     Header size HDSIZE
 CADVRN         I*4     Bank version number
 CADNRW         I*4     Number of ADC values (NADC)     \ might be
 CADNCL         I*4     Number of values per ADC (NVAL) / packed
--------------------------------------------------------------------
 CADVAL         I*4     First Packed word of electronic
                address and ADC Pulse height
--------------------------------------------------------------------
```

Notes:

- Version number, header size could be packed in one word, also the number of ADCs and the number of values per ADC.
- The data words are packed as coming from the DAQ i.e. 13 bits will be reserved to the ADC value and the remaining bit to the electronic address.
- The subsequent packed words can be obtained using the offset:

```
OFFVAL = IW(CADHDS) + IW(CADNCL)*(I-1) + CADVAL
```

### 6.2 CTDR Bank – Calorimeter TDc Raw data bank

The CTDR bank stores the TDC values for the calorimeter, it has the following YBOS header characteristics:

```
Bank Name:          "CTDR"
Bank Number:          1
Bank Data Length:   HDSIZE+NTDC*NVAL     (in 4 bytes words)
Bank Type:          BNKTI4               (integer*4)
```

The Bank is then structured in a header block followed by a data block. Each word in the bank can be located through the displacement pointers shown below.

```
------------------------------------------------------------------
 Offset      Word type    Description
------------------------------------------------------------------
 CTDHDS        I*4     Header size HDSIZE
 CTDVRN        I*4     Bank version number
 CTDNRW        I*4     Number of TDC values (NTDC)     \ might be
 CTDNCL        I*4     Number of values per TDC (NVAL) / packed
------------------------------------------------------------------
 CTDVAL        I*4     First Packed word of electronic
               address and TDC value
------------------------------------------------------------------
```

Notes:

- Version number, header size could be packed in one word, also the number of TDCs and the number of values per TDC.
- The data words are packed as coming from the DAQ i.e. 13 bits will be reserved to the ADC value and the remaining bit to the electronic address.
- The subsequent packed words can be obtained using the offset:

```
OFFVAL = IW(CTDHDS) + IW(CTDNCL)*(I-1) + CTDVAL
```

### 6.3 CELE Bank - Calorimeter ELEment bank

The CELE bank stores calibrated energy and time information for each element of the calorimeter. It has the following YBOS header characteristics:

```
Bank Name:          "CELE"
Bank Number:          1
Bank Data Length:   HDSIZE+NELE*NVAL  (in 4 bytes words)
Bank Type:          MIXED   (integer*4 + real4)
```

The Bank is structured in a header block followed by a data block. Each word in the bank can be located through the offset parameters shown below.

```
-------------------------------------------------------------------
 Offset            Word type   Description
-------------------------------------------------------------------
 CELHDS            I*4     Header size HDSIZE
 CELVRN            I*4     Bank version number
 CELCAL            I*4     Calibration information
 CELNRW            I*4     Number of elements (NELE)  \ might be
 CELNCL            I*4     Values per element (NVAL)  / packed
-------------------------------------------------------------------
 HDSIZE+CELADR+(I-1)*NVAL  I*4     Packed address \
 HDSIZE+CELEA +(I-1)*NVAL  R*4     Energy side A   \
 HDSIZE+CELEB +(I-1)*NVAL  R*4     Energy side B    - NELE Blocks
 HDSIZE+CELTA +(I-1)*NVAL  R*4     Time   side A   /
 HDSIZE+CELTB +(I-1)*NVAL  R*4     Time   side B  /
-------------------------------------------------------------------
```

Notes:

- Header and data block size could be packed in one word, also number of elements and number of values per element.
- Calibration information could be packed as follows:

| Bits range | Information |
|---|---|
| 0 – 7 | Pedestal version |
| 8 – 15 | Energy threshold in multiples of 0.1 MeV |
| 16 – 19 | Flag for dead channel correction |
| 20 – 23 | Flag for energy calibration |
| 24 – 31 | Flag for time calibration |

- The address for the element is packed as follows:

| Bits range | Information |
|---|---|
| 0 – 3 | Column number |
| 4 – 6 | Plane number |
| 7 – 11 | Wedge number |
| 12 – 13 | not used at the moment |
| 14 – 15 | Calorimeter number |

## 6.4 CEKA Bank - Calorimeter ELEment into MC-KINE bank

The CEKA bank links the element bank CELE to the KINE montecarlo banks by storing the addresses of the KINE tracks contributing to each element. The address for the track is the KINE bank number.

It has the following YBOS header characteristics:

```
Bank Name:          "CEKA"
Bank Number:          1
Bank Data Length:   HDSIZE+NELE*NVAL  (in 4 bytes words)
Bank Type:          BNKTI4   (integer*4 )

-------------------------------------------------------------------
 Offset           Word type   Description
-------------------------------------------------------------------
 CEKHDS              I*4    Header size HDSIZE
 CEKVRN              I*4    Bank version number
 CEKNCL              I*4    Number of elements (NELE)
 CEKNTR              I*4    Number of KINE tracks per element (NVAL)
 HDSIZE+CEKADR
 +(I-1)*NVAL         I*4    Montecarlo Kine Bank Number
-------------------------------------------------------------------
```

## 6.5 ITDR - Inner Chamber TDC Raw bank

The ITDR bank contains the TDC values of the Inner Chamber

```
------------------------------------------------------------------
 Offset      Word type   Description
------------------------------------------------------------------
 ITDHDS         I*2     Header size ICTRHS
 ITDVRN         I*2     Bank version number
 ITDNRW         I*2     Number of TDC values (NTDCIC)     \ might be
 ITDNCL         I*2     Number of values per TDC (ICTREL) / packed
--------------------------------------------------------------------
    Bank header:
    ------------
Bank Name:          ITDR
Bank Number:          ?
Next Bank:            ?
Bank Data Length:   NTDCIC*ICTREL+ICTRHS
Bank Type:          BNKTI2         !  I*2

    Bank structure:
    ---------------
(the parameters that describe the bank are in the file ICTDCR.CIN)
----------------------------------------------------------------
          displacement         word type        description
----------------------------------------------------------------
```

```
        _ ICTRPO+
       |  ICTREL*(iblock-1)+1     I*2        TDC address
       |
NTDCIC -|
blocks  |  ICTRVA+
        - ICTREL*(iblock-1)+1     I*2        TDC count
```

where NTDCIC is the number of fired TDCs and ICTREL is the size of a block of information.

### 6.6 IADR - Inner Chamber ADC Raw bank

The IADR bank contains the ADC values of the Inner Chamber

```
-----------------------------------------------------------------
 Offset     Word type   Description
-----------------------------------------------------------------
 IADHDS        I*2     Header size ICARHS
 IADVRN        I*2     Bank version number
 IADZSF        I*2     Zero suppression information
 IADNRW        I*2     Number of ADC values (NADCIC)     \ might be
 IADNCL        I*2     Number of values per ADC (ICAREL) / packed
------------------------------------------------------------------

    Bank header:
    ------------
Bank Name:          IADR
Bank Number:          ?
Next Bank:            ?
Bank Data Length:   NADCIC*ICAREL+ICARHS
Bank Type:          BNKTI2          !  I*2

    Bank structure:
    ---------------
(the parameters that describe the bank are in the file ICADCR.CIN)
----------------------------------------------------------------
          displacement          word type        description
----------------------------------------------------------------


        _ ICARPO+
       |  ICAREL*(iblock-1)+1     I*2        ADC address
       |
NADCIC -|
blocks  |  ICARVA+
        - ICAREL*(iblock-1)+1     I*2        ADC count
```

where NADCIC is the number of fired ADCs and ICAREL is the size of a block of information. The address of a TDC or of an ADC is packed as follows:

| bit range | Information |
|---|---|
| 0 – 0 | part number |
| 1 – 5 | layer number |
| 6 – 10 | tube number |
| 11 –11 | side number |

## 6.7 ITAE - Inner Chamber TDC and ADC Element bank

The ITAE bank contains the TDC and ADC values for each side of a wire of the Inner Chamber.

```
---------------------------------------------------------------
 Offset         Word type   Description
---------------------------------------------------------------
 IELHDS            I*2     Header size ICELHS
 IELVRN            I*2     Bank version number
 IELCAL            I*2     Calibration information
 IELNRW            I*2     Number of elements (NTAEIC)\ might be
 IELNCL            I*2     Number of values  (ICTAEL) / packed
--------------------------------------------------------------

    Bank header:
    ------------
Bank Name:          ITAE
Bank Number:          ?
Next Bank:            ?
Bank Data Length:   NTAEIC*ICTAEL+ICELHS
Bank Type:          BNKTI2         !  I*2

    Bank structure:
    ---------------
(the parameters that describe the bank are in the file ICTAEL.CIN)
----------------------------------------------------------------
          displacement          word type        description
----------------------------------------------------------------


         _ ICTAPO+
        |  ICTAEL*(iblock-1)+1     I*2        wire address
        |
        |
        |  ICTAVA+                            propagation time
```

```
        |  ICTAEL*(iblock-1)+1     I*2        side A
        |
        |
NTAEIC -|  ICTAVB+                            propagation time
blocks  |  ICTAEL*(iblock-1)+1     I*2        side B(if we will use
        |                                     TDCs on both sides)
        |
        |  ICQAVA+                            charge
        |  ICTAEL*(iblock-1)+1     I*2        side A
        |
        |
        |  ICQAVB+                            charge
         - ICTAEL*(iblock-1)+1     I*2        side B
```

### 6.8 ICEL - Inner Chamber Element bank

The ICEL bank contains some quantities related to the Inner Chamber wires.

```
    Bank header:
    ------------
Bank Name:          ICEL
Bank Number:          1
Next Bank:            0
Bank Data Length:   NELBIC*ICEBEL+1 ! NELBIC from the detector map
Bank Type:          Mixed Type

    Bank structure:
    ---------------
(the pointers to move inside the bank will be described in ICEL.CIN)
----------------------------------------------------------------
            displacement        word type         description
----------------------------------------------------------------

            ICEBVN                 I*4         Bank Version Number
           -
          | ICEBPO+
          | ICEBEL*(iblock-1)+1    I*4         Wire address
          |
NELBIC ---| ICEBDT+
          | ICEBEL*(iblock-1)+1    R*4         Drift time
          |
          | ICEBDD+
          | ICEBEL*(iblock-1)+1    R*4         Drift Distance
          |
          | ICEBZC+
          | ICEBEL*(iblock-1)+1    R*4         z coordinate
           -
```

### 6.9 DTDR - Drift Chamber TDC Raw bank

The DTDR bank contains the TDC values of the Drift Chamber as they are coming from data acquisition.

For debugging purposes, in a subset of all events, TDC values coming from DAQ will not be zero suppressed, and the zero suppression flag will indicate this case. We do not think it is necessary to add informations on the set of zero suppression constants, because it must be possible to retrieve them from the database via the run number. The same arguments apply also for ADC pedestal subtraction. The word containing the address for each sense wire and the TDC value is one 32 bit word packed as follows:

| Bits range | Information |
|---|---|
| 0 – 19 | crate/slot/channel |
| 20 – 31 | TDC value |

```
    YBOS Bank header:
    -----------------
Bank Name:          DTDR
Bank Number:          1
Bank Data Length:
Bank Type:          BNKTI4  !  I*4
```

The Bank is structured in a header block followed by a data block. Each word in the bank can be located through the offset shown below.

```
---------------------------------------------------------------
  offset         word type     description
---------------------------------------------------------------


  DTDHDS            I*4   header size (HDSIZE)
  DTDVRN            I*4   bank Version Number
  DTDNRO            I*4   number of rows=number of fired TDC
                          (NROW)
  DTDNCO            I*4   number of data word/row (NCOL)
  DTDZES            I*4   zero suppression flag
---------------------------------------------------------------
  DTDADR            I*4   first packed word containing
                          sense wire electronic address and
                          TDC value
  ....                    ....
```

The subsequent packed word values can be obtained using the offsets
OFFADR= HDSIZE+DTDADR+(I-1)*NCOL (where I runs from 1 to NROW)

### 6.10 DADR - Drift Chamber ADC Raw bank

The DADR bank contains the ADC values of the Drift Chamber The word containing the address for each sense wire and the ADC value is one 32 bit word packed as follows:

| Bits range | Information |
|---|---|
| $0 - 19$ | crate/slot/channel |
| $20 - 31$ | ADC value |

```
    YBOS Bank header:
    -----------------
Bank Name:          DADR
Bank Number:          1
Bank Data Length:
Bank Type:          BNKTI2  !  I*4
```

The Bank is structured in a header block followed by a data block. Each word in the bank can be located through the offset shown below.

```
---------------------------------------------------------------
  offset      word type         description
---------------------------------------------------------------


  DADHDS         I*4   header size (HDSIZE)
  DADVRN         I*4   bank Version Number
  DADNRO         I*4   number of rows=number of ADC fired (NROW)
  DADNCO         I*4   number of data word/rows (NCOL)
  DADPES         I*4   pedestal subtraction flag
---------------------------------------------------------------
  DADADR         I*4   first packed word containing
                       sense wire electronic address and
                       ADC value
  ....                 ....
```

The subsequent packed words values can be obtained using the offsets
OFFADR= HDSIZE+DADADR+(I-1)*NCOL (where I runs from 1 to NROW)

### 6.11 DTCE - Drift Chamber Time and Charge element bank

The DTCE bank contains the time and the charge values of the Drift Chamber obtained with calibration from the raw ADC and TDC banks.

The data calibration algorithm (see fig 1) could take care of the following:

1)TDC's zero suppression or ADC's pedestal subtraction for the special debugging events.

2)Suppression of noisy channels.

3)Unpacking of the raw data word containing the address and the TDC,ADC values.

4)Address pairing of TDC and ADC values.

5)Application of (RUN dependent) calibration constants from raw counts to ns and pC.

6)Decoding of electronic address into geometric address, and possible correction for miscabling (see subsection 0.5).

The address for each sense wire could be packed (if necessary) as follows:

| Bits range | Information |
|---|---|
| 0 – 15 | Sense wire number |
| 16 – 31 | Layer number |

```
    YBOS Bank header:
    -----------------
Bank Name:          DTCE
Bank Number:          1
Bank Data Length:
Bank Type:            mixed
```

The Bank is structured in a header block followed by a data block. Each word in the bank can be located through the offset shown below.

```
----------------------------------------------------------------
  offset          word type         description
----------------------------------------------------------------


  DTCHDS             I*4   header size (HDSIZE)
  DTCVRN             I*4   bank Version Number
  DTCNRO             I*4   number of rows (NROW)
  DTCNCO             I*4   number of data word/rows (NCOL)
----------------------------------------------------------------
  DTCADR             I*4   first sense wire address
  DTCTIM             R*4   first time value
  DTCCHA             R*4   first charge value
  ....                     ....
```

The subsequent values can be obtained using the offsets
OFFADR= HDSIZE+DTCADR+(I-1)*NCOL,
OFFTIM= HDSIZE+DTCTIM+(I-1)*NCOL,
OFFCHA= HDSIZE+DTCCHA+(I-1)*NCOL,
where I runs from 1 to NROW

## 6.12 DHRE - Drift Chamber Hit Radius and associated error element bank

The DHRE bank contains the values of the hit radius and relative error obtained applying an averaged time to space relation. This bank is an input to the pattern recognition module.

```
    YBOS Bank header:
    -----------------
```

```
Bank Name:          DHRE
Bank Number:        1
Bank Data Length:
Bank Type:          Mixed
```

The Bank is structured in a header block followed by a data block. Each word in the bank can be located through the offset shown below.

```
--------------------------------------------------------------
  offset         word type        description
--------------------------------------------------------------


  DHRHDS            I*4   header size (HDSIZE)
  DHRVRN            I*4   bank Version Number
  DHRNRO            I*4   number of rows (NROW)
  DHRNCO            I*4   number of data word/rows (NCOL)
--------------------------------------------------------------
  DHRADR            I*4   first sense wire address
  DHRRAD            R*4   first hit radius
  DHRERR            R*4   first radius error
  DHRTRN            I*4   track candidate number (link)
  ....                    ....
```

The subsequent values can be obtained using the offsets
OFFADR= HDSIZE+DHRADR+(I-1)*NCOL,
OFFRAD= HDSIZE+DHRRAD+(I-1)*NCOL,
OFFERR= HDSIZE+DHRERR+(I-1)*NCOL,
OFFTRN= HDSIZE+DHRTRN+(I-1)*NCOL,
where I runs from 1 to NROW

The track candidate number (t.c.n.) is assigned to the hit by the pattern recognition module. Its main purpose is to declare to possible iterations of the pattern recognition whether this hit is available (t.c.n.= 0) or not (t.c.n.> 0).

### 6.13 DHKA - Drift chamber hit address into the MC-KINE bank

The DHKA bank contains the addresses to retrieve information on the track kinematics stored in the KINE bank for Monte Carlo data. As shown in fig.2, there is a link to the KINE bank for each hit. The address value is the number of the KINE bank.

```
    YBOS Bank header:
    -----------------
Bank Name:          DHKA
Bank Number:        1
Bank Data Length:
Bank Type:          BNKTI4
```

The Bank is structured in a header block followed by a data block. Each word in the bank can be located through the offset shown below.

```
--------------------------------------------------------------
  offset              word type        description
--------------------------------------------------------------


  DHKHDS                  I*4   header size
  DHKVRN                  I*4   bank Version Number
  DHKNRO                  I*4   number of rows
  DHKNCO                  I*4   number of data words/row
--------------------------------------------------------------
  DHKADR                  I*4   Monte Carlo KINE bank number
  ....                          ....
```

### 6.14 DHHA - Drift chamber hit address into the MC-HITS bank

The DHHA bank contains the the addresses to retrieve information on the hit stored in the HITS bank for Monte Carlo data. As shown in fig.2, there is a link to the HITS bank for each hit. The address value is the number of the hit in the HITS bank.

```
    YBOS Bank header:
    -----------------
Bank Name:          DHHA
Bank Number:        1
Bank Data Length:
Bank Type:          BNKTI4
```

The Bank is structured in a header block followed by a data block. Each word in the bank can be located through the offset shown below.

```
--------------------------------------------------------------
  offset        word type        description
--------------------------------------------------------------


  DHHHDS           I*4   header size
  DHHVRN           I*4   bank Version Number
  DHHNRO           I*4   number of rows
  DHHNCO           I*4   number of data word/rows
--------------------------------------------------------------
  DHHADR           I*4   hit address into Monte Carlo HITS bank
  ....                   ....
```

# 7 Segment banks

Above the element banks are the segment. Segments are the combination of the information from one or more detector. The following is a description of the banks defined for the E.M. Calorimenter and for the Drift Chamber.

## 7.1 Segment Banks-Processed Data from the Calorimeters

In this section we describe the "segment" level of the data structure for the e.m. calorimeter. All the information obtained by grouping together more calorimeter elements to form the energy clusters belongs to this structure. No attempt is done in using information coming from other detectors or in assigning a particle identification.

A cluster is thought as the collection of energy deposition in the calorimeter cells which are related by spatial contiguity and time coincidence. Such a correspondence can be found with different criteria and we think that a simple pattern of adjacency of the fired cells in the r-phi plane can be used as a starting point in the clustering procedure. The result of such a pre-clustering stage will be stored separately from the final cluster especially if more than one attempt will be done in the merging of the energy clumps and in the determination of the final energy, position and direction of each particle.

From a technical point of view, various solutions for the banks' structure have been discussed and the final choice has been taken together with the other detector experts to construct a similar and integrated data structure for the whole KLOE apparatus. Essentially, we have decided to store the cluster number into the bank number and to have two different banks for each cluster (or precluster). One contains the physical information of the cluster while the other one links backward the cluster to the element bank CELE.

In the following we present the details of such banks both for the pre-clustering and the merging stage.

## 7.2 CPPS-CPLS Bank - Calorimeter Pre-clustering data Banks

As already mentioned, two different banks can be envisaged for each cluster, the CPPS (Pre-clustering parameter segment) and the CPLS (Pre-clustering list segment) bank.

The CPPS bank contains a fixed number of real words that are the cluster parameters. The CPLS bank consists in the list of cells belonging to the cluster. This is achieved technically by storing the IBLOCK pointer in the CELE bank.

*CPPS Bank* The Bank has the following YBOS Header Characteristics:

```
Bank Name:     "CPPS"
Bank Number:      pre-cluster number
Bank Data Length:   CPPHS+1+CPPBS        ! in 4 bytes words
Bank Type:          BNKT..               ! Real*4
```

The Bank is then structured in a Version Block followed by a Data Block. Each Word in the bank can be located through the displacement pointers shown below.

```
Displacement         Word Type    Word Description
-----------------------------------------------------------------
Version Header Block (size = CPPHS)
-----------------------------------------------------------------
  ?CPPHL               I*4         Header Block Length (=CPPHS)
   CPPVN               I*4         Version Number
   CPPFL               I*4         Adjacency Flag
  ?CPPDL               I*4         Data Block Length (=CPPBS)
------------------------------------------------------------------------
       Data Block
------------------------------------------------------------------------

  CPPHS+CPPENE         R*4      EPCL = Pre-cluster Energy

  CPPHS+CPPPOS+
  (0:2)                R*4      XPCL,YPCL,ZPCL = Pre-cluster
                                          centroid

  CPPHS+CPPTIM         R*4      TPCL = time of the Pre-cluster
                                      centroid
  CPPHS+CPPTSI+
  (0:3 )               R*4      T1PCL,ST1PCL,
                                T2PCL,ST2PCL = one-side
                                time and RMS of the Pre-cluster
                                centroid
  CPPHS+CPPTFL         I*4      TFLPCL = topological flag
```

*CPLS Bank*

The Bank has the following YBOS Header Characteristics:

```
Bank Name:    "CPLS"
Bank Number:     Pre-cluster number
Bank Data Length:   CPLHS+1+NELEPC      ! in 2 bytes words
Bank Type:          BNKTI2              ! I*2

  Displacement         Word Type    Word Description
-------------------------------------------------------------------
Version Header Block (size = CPLHS)
-------------------------------------------------------------------
  ?CPLHL               I*2         Header Block Size (=CPLHS)
   CPLVN               I*2         Version Number
   CPLDL               I*2         Data Block size (=NELEPC)
-------------------------------------------------------------------
       Data Block
-------------------------------------------------------------------
NELEPC
blocks:
```

```
  CPLHS+CPLPO+
  CPLBS*(Iblock-1)      I*2         CPLPOI = pointer (iblock)
                                             to CELE bank
```

### 7.3 CLPS-CLLS Bank - Clustering data Banks

These data sets, apart from some different parameters appearing in the CLPS bank, closely reproduce the previous structures.

*CLPS Bank*

The Bank has the following YBOS Header Characteristics:

```
Bank Name:     "CLPS"
Bank Number:      cluster number
Bank Data Length:   CLPHS+1+CLPBS       ! in 4 bytes words
Bank Type:          BNKT..              ! Real*4
```

The Bank is then structured in a Version Block followed by a Data Block. Each Word in the bank can be located through the displacement pointers shown below.

```
   Displacement         Word Type    Word Description
-----------------------------------------------------------------
Version Header Block (size = CLPHS)
-----------------------------------------------------------------
   ?CLPHL                I*4          Header Block Size (=CLPHS)
    CLPVN                I*4          Version Number
   ?CLPDL                I*4          Data Block Size (=CLPBS)
-----------------------------------------------------------------
       Data Block
-----------------------------------------------------------------
    CLPHS+CLPENE      R*4          ECL = cluster Energy

    CLPHS+CLPPOS+
    (0:5)            R*4         XCL,SXCL,YCL,SYCL,ZCL,SZCL =
                                 cluster centroid and RMS
    CLPHS+CLPTIM+
    (0:1)            R*4         TCL,STCL = time of the
                                 cluster and RMS

    CLPHS+CLPTAP+
    (0:5)            R*4         XACL,SXACL,YACL,
                                 SYACL,ZACL,SZACL =
                                 cluster apex and RMS
    CLPHS+CLPCOS+
    (0:5)            R*4         CX,SCX,CY,
                                 SCY,CZ,SCZ = direction cosines
                                 of the particle and RMS
    CLPHS+CLPEFL     I*4         INEXF = Flag for fully/partially
                                         contained events.
```

*CLLS Bank*

The bank fully reproduces the structure of the CPLS bank.

The Bank has the following YBOS Header Characteristics:

```
Bank Name:    "CLLS"
Bank Number:     cluster number
Bank Data Length:   CLLHS+1+ NELECL     ! in 2 bytes words
Bank Type:          BNKTI2              ! I*2
```

The Bank is then structured in a Version Block followed by a Data Block. Each Word in the bank can be located through the displacement pointers shown below.

```
Displacement         Word Type    Word Description
---------------------------------------------------------------
Version Header Block (size = CLLHS)
---------------------------------------------------------------
  ?CLLHL                I*2         Header Block Size (=CLLHS)
   CLLVN                I*2         Version Number
   CLLDL                I*2         Data Block Size (=NELECL)
---------------------------------------------------------------
       Data Block
---------------------------------------------------------------
NELECL blocks:
  CLLHS+CLLPO+
  CLLBS*(iblock-1)   I*2        CLPPOI = Pointer (iblock)
                                         to CELE bank
```

The locations beginning with a question mark are those filled by non-variable numbers: the presence of these words, as it has been discussed, can be convenient but it is not mandatory.

## 7.4 DPRS - Drift chamber track candidate banks from Pattern Recognition.

The DPRS bank is an output of the pattern recognition module and an input to the track fit module. It contains:

1) the values of the helix parameters from the raw fit

2) number of hits assigned to this track candidate

3) list of hit addresses inside the DHRE bank

```
    YBOS Bank header:
    -----------------
Bank Name:          DPRS
Bank Number:        Track candidate number
Bank Data Length:
Bank Type:          Mixed (5R4,I4)
```

The Bank is structured in a header block followed by a data block. Each word in the bank can be located through the offset shown below.

```
--------------------------------------------------------------
     offset              word type        description
--------------------------------------------------------------

     DPRHDS                  I*4   header size
     DPRVRN                  I*4   bank Version Number
--------------------------------------------------------------
     DPRXCO                  R*4   X coordinate of point of closest
                                   approach to the first hit (PCA).
     DPRYCO                  R*4   Y coordinate of PCA
     DPRZCO                  R*4   Z coordinate of PCA
     DPRCUR                  R*4   curvature
     DPRPHI                  R*4   azimuth angle at PCA
     DPRCTT                  R*4   cot(theta)
     DPRNHI                  I*4   number of hits assigned to this
                                   track candidate
     DPRFHA                  I*4   first hit address
     ....                          ....
```

## 7.5 DTFS - Drift chamber Track Fit results banks.

The DTFS bank contains track fit results. This bank is an output of the track fit module and an input to the vertex fit module. After a successful vertex fit, track parameters at the vertex point are added. The DTFS bank also contains the list of hit addresses inside the DHRE bank for all the hits actually accepted by the fit (it could be shorter than the list of hits received from pattern recognition).

```
    YBOS Bank header:
    -----------------
Bank Name:          DTFS
Bank Number:        Track number
Bank Data Length:
Bank Type:          Mixed
```

The Bank is structured in a header block followed by a data block. Each word in the bank can be located through the offset shown below.

```
---------------------------------------------------------------
  offset           word type         description
---------------------------------------------------------------
  DTFHDS              I*4          header size
  DTFVRN              I*4          bank Version Number
---------------------------------------------------------------
  DTFNCO              I*4          number of rows for
```

| | | |
|---|---|---|
| | | kinematic information |
| DTFPRN | I*4 | original track candidate number from pattern recognition |
| DTFQFL | I*4 | Track fit quality flag |
| DTFINH | I*4 | Initial number of hits |
| DTFNHF | I*4 | Number of hits after track fit has converged |
| DTFXCO | R*4 | X coordinate of PCA |
| DTFYCO | R*4 | Y coordinate of PCA |
| DTFZCO | R*4 | Z coordinate of PCA |
| DTFCUR | R*4 | Curvature (signed relatively to the charge of the particle) |
| DTFCTT | R*4 | cot(theta) \| of the particle |
| DTFPHI | R*4 | azimuth \| movement at PCA |
| DTFCVM** | R*4 | covariance matrix (15 words) |
| DTFCHI | R*4 | chi**2 of the track fit |
| DTFPCH | R*4 | chi**2 probability of the fit |
| DTFXLP | R*4 | x- \| coordinate of the |
| DTFYLP | R*4 | y- \| last track |
| DTFZLP | R*4 | z- \| fit point |
| DTFCLP | R*4 | curvature \| at |
| DTFTLP | R*4 | cot(theta) \| the last |
| DTFPLP | R*4 | azimuth \| point |
| DTFMSK | I*4 | number of the multiple scattering kinks |
| DTFCHM | R*4 | chi**2 of the multiple scattering |
| DTFCRV | R*4 | Curvature (signed relatively to the charge of the particle) |
| DTFCTV | R*4 | cot(theta) |
| DTFPHV | R*4 | azimuth at the vertex point |
| DTFCOV** | R*4 | covariance matrix |
| DTFLNV | R*4 | track length after vertex fit |
| DTFPID | I*4 | Particle id |
| DTFPVM | I*4 | pointer to the mother vertex bank |
| DTFPVS | I*4 | pointer to the secondary vertex bank |
| DTFNHR | I*4 | pointer to the secondary vertex |
| DTFFHA | I*4 | first hit address |
| .... | | .... |

At some point of the track fit iteration the parameters of the track movement can be considered to be defined with accuracy good enough to correct the T0 and to use finer time-space relations, depending on the space position of the point of closest approach to each wire. At that point two new banks will be created: one is a DHRE bank with bank number 2 and new values of the hit radius and its error. This information will be updated each time a new time-space relation will be applied (in other words we propose to have only two DHRE banks: the initial one and a current one); the other is a DHSP bank.

### 7.6 DHSP - Drift chamber Hit Space Points bank.

The DHSP bank contains the space points of the hits defined during the fit procedure. It is created and updated together with the DHRE bank number 2. It contains the same number of hits as the DHRE bank, but if some hit is not used by the track fit, the corresponding row of the DHSP bank contains zeros.

```
    YBOS Bank header:
    -----------------
Bank Name:          DHSP
Bank Number:        1
Bank Data Length:
Bank Type:          Mixed
```

The Bank is structured in a header block followed by a data block. Each word in the bank can be located through the offset shown below.

```
-----------------------------------------------------------------
  offset              word type         description
-----------------------------------------------------------------

  DHSHDS                  I*4   header size
  DHSVRN                  I*4   bank Version Number
  DHSNRO                  I*4   number of rows (NROW)
  DHSNCO                  I*4   number of data words/row (NCOL)
-----------------------------------------------------------------
  DHSADR                  I*4   first sense wire address
  DHSXCO                  R*4   first hit x-coordinate
  DHSYCO                  R*4   first hit y-coordinate
  DHSZCO                  R*4   first hit z-coordinate
  ....                          ....
```

The subsequent values can be obtained using the offsets
OFFADR= DHSADR+(I-1)*NCOL,
OFFXCO= DHSXCO+(I-1)*NCOL,
OFFYCO= DHSYCO+(I-1)*NCOL,
OFFZCO= DHSZCO+(I-1)*NCOL,
where I runs from 1 to NROW

### 7.7 DVFS - Drift chamber Vertex Fit results banks.

The DVFS bank is an output of the vertex fit module. This bank contains vertex fit results on the space position of the vertex, the list of tracks (incoming to and originating from the vertex) and vertex ID flag from the fit.

```
    YBOS Bank header:
    -----------------
Bank Name:          DVFS
Bank Number:        Vertex number
Bank Data Length:
Bank Type:            mixed
```

The Bank is structured in a header block followed by a data block. Each word in the bank can be located through the offset shown below.

```
--------------------------------------------------------------------
  offset              word type         description
--------------------------------------------------------------------
  DVFHDS                  I*4          header size
  DVFVRN                  I*4          bank Version Number
--------------------------------------------------------------------
  DVFQFL                  I*4          Vertex fit quality flag
  DVFXCO                  R*4          X coordinate of the vertex fit
  DVFYCO                  R*4          Y coordinate of the vertex fit
  DVFZCO                  R*4          Z coordinate of the vertex fit
  DVFCHI                  R*4          Chi**2 of the fit
  DVFCOV**                R*4          covariance matrix (6 words)
  DVFIDF                  I*4          Vertex fit identification
                                       flag
  DVFMTR                  I*4          Pointer onto the mother
                                       track bank
  DVFNTR                  I*4          Number of secondary tracks
                                       originating from that vertex
  DTFTRA                  I*4          pointer to the 1st DTFS bank
  ....                                 ....
```

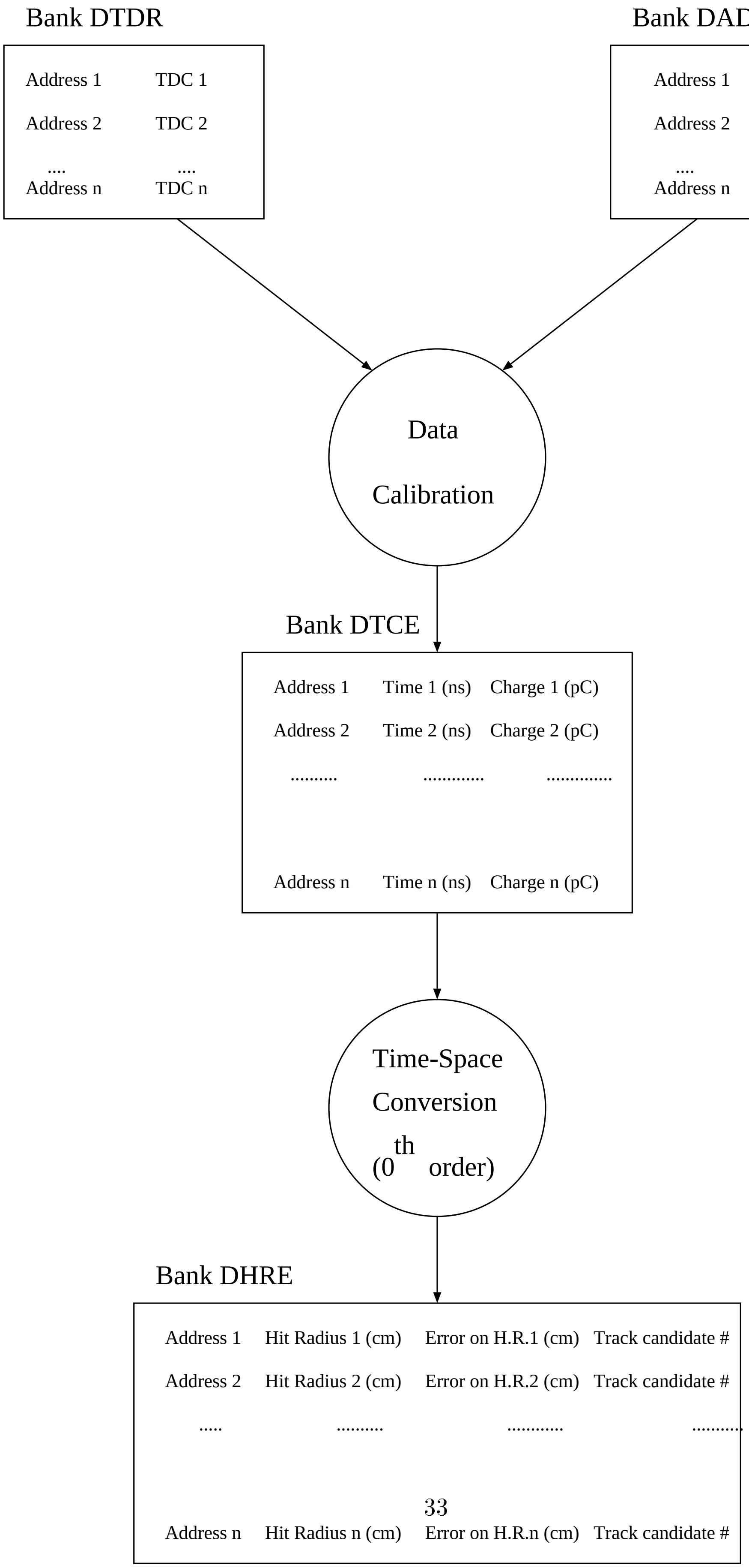




Figure 1: *Drift Chamber data reconstruction flow chart*

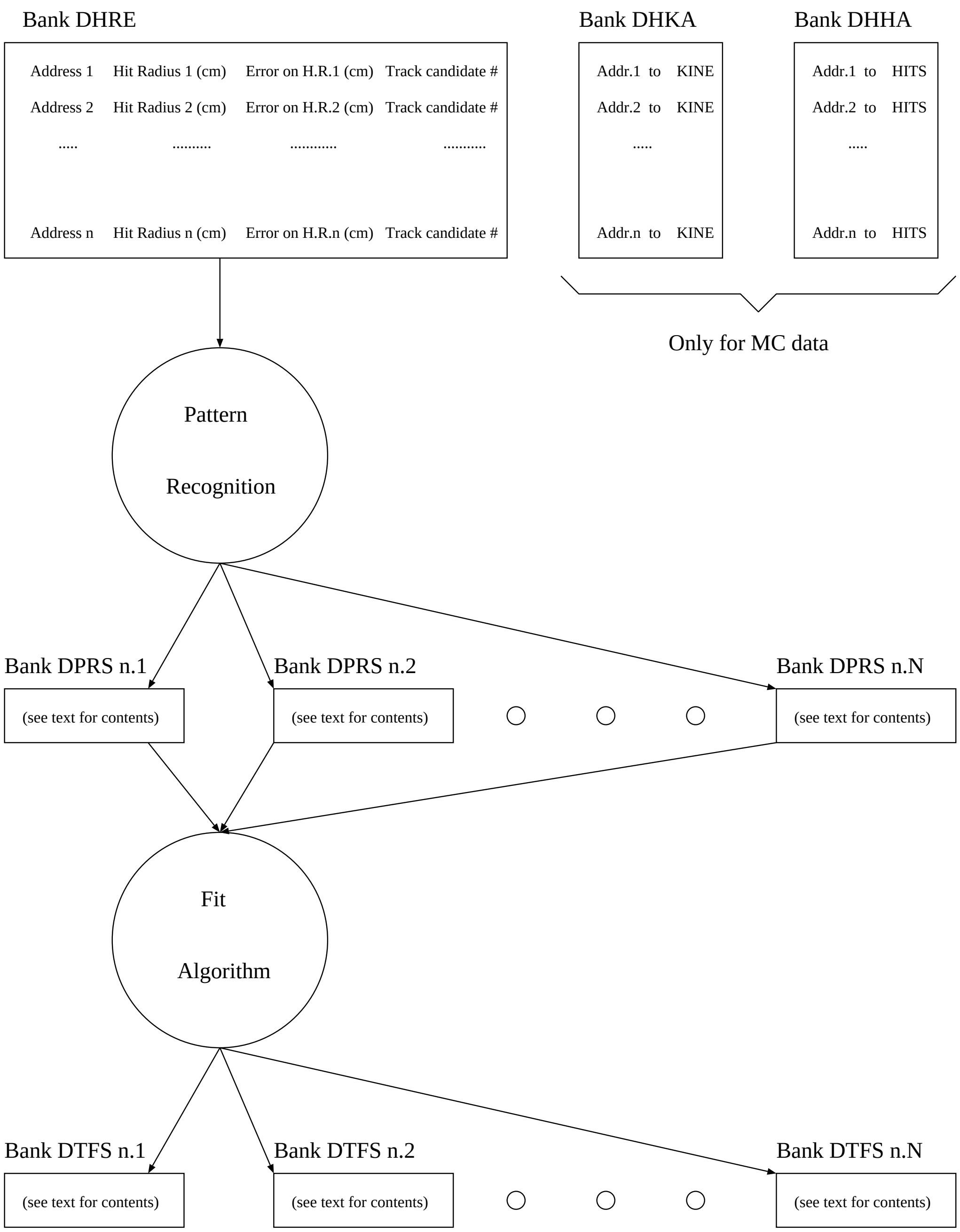


Figure 2: *Drift Chamber data reconstruction flow chart*

# 8 Databases

This section describes the Databases structure itself and the bank structure which can be put in these databases. More information about database can be found in the RTDB$DOC:RT_DATABASE.TEX.

There are three kinds of database, one for the Run Condition, one for the Calibration and one for the Geometry. Each database is divided in directories and has its own "keys" definition to characterise an object inside the database.

## 8.1 Databases directories description

First here is the description of the directories :

For the run condition database :

```
//CDRT
      /HELP
      /DICTIONARY
      /RUN_COND
               /TRIGGER_ST
               /TRIGGER_LT
               /TRIGGER_MT
               /DRCH_LT
               /DRCH_MT
               /DRCH_ST
               /CALO_LT
               /CALO_MT
               /CALO_ST
               /BEAM_LT
               /BEAM_MT
               /BEAM_ST
```

For the calibration database :

```
//CDCT
      /HELP
      /DICTIONARY
      /CALIBRATION
                  /DRCH_LT
                  /DRCH_MT
                  /DRCH_ST
                  /CALO_LT
                  /CALO_MT
                  /CALO_ST
                  /TRIG_LT
                  /TRIG_MT
                  /TRIG_ST
```

For the geometry database :

```
//CDGT
      /HELP
      /DICTIONARY
      /GEOMETRY
               /DRCH
                    /PARAM_LT
                    /DESCR_MT
                    /DESCR_LT
               /CALO
                    /DESCR_MT
                    /DESCR_LT
               /TRIG
                    /DESCR_MT
                    /DESCR_LT
```

The objects contained in the databases are ZEBRA banks. The format of these banks can be INTEGER*4 or REAL or mixed case if necessary. The name is a 4 characters string, the first letter refers to the detector (C for the Calorimeter, D for the Drift chamber, T for the Trigger), the three remaining letter ”describes” the object. The displacements given in the following description are with respect to the pointer to the data banks.

## 8.2 Database KEYS definitions

The keys definitions are the following :

| System keys | | |
|---|---|---|
| Key Number | Meaning | Parameter |
| 1 | Serial Number | IDHKSN |
| 2 | Pointer | IDHPTR |
| 3 | Flags | IDHFLG |
| 4 | Insertion Time | IDHINS |
| 5 | *reserved* | |
| Special User keys | | |
| 6 | Unique source identifier | IDHUSI |
| 7 | Software reference number | IDHSRN |
| 8 | *Unused* | |
| 9 | Run Type (or Trigger Number) | DBRUNT |
| 10 | OFFline-ONline Flag | DBOFON<br>(Only for calibration) |
| Validity range pair | | |
| 11 | Run Number Start of validity | DBSRTR |
| 12 | Run Number End of validity | DBENDR |
| Normal User keys | | |
| 13 | Bank Name | DBBKNA |
| 14 | Version number<br>(if needed)<br>or Part. Id Number<br>or Run number | DBCPID<br><br>(for calorimeter calib.)<br>(for calibration) |

Most of the key are common of all the databases, only the 10th key is particular to the calibration database and the 14th key has different meaning depending to the database and also to the detector which are considered.

## 8.3 Databases banks description

The existing banks are :

1) For the Run Condition database

**Calorimeter**

**CPED** is the bank containing the Pedestal values and their sigmas. It is a bank of 2*(number of channels) or 4*(number of channels) reals, depending if there are one or two sets of ADC. The directory pathname to access this bank is "//CDRT/RUN_COND/CALO_ST".
The desription of the bank in the case of one set of ADC is:

| Displacement (I*4) | Word Type | Description |
|---|---|---|
| +1 | R*4 | Pedestal value of the first ADC |
| +2 | R*4 | Pedestal sigma of the first ADC |
| . | R*4 | |
| . | R*4 | |
| . | R*4 | |
| +2*(i-1)+1 | R*4 | Pedestal value of the i th ADC |
| +2*(i-1)+2 | R*4 | Pedestal sigma of the i th ADC |

In the case of two sets of ADC :

| Displacement | Word Type | Description |
|---|---|---|
| +1 | R*4 | First Pedestal value of the first ADC channel |
| +2 | R*4 | First Pedestal sigma of the first ADC channel |
| +3 | R*4 | Second Pedestal value of the first ADC channel |
| +4 | R*4 | Second Pedestal sigma of the first ADC channel |
| . | R*4 | |
| . | R*4 | |
| . | R*4 | |
| +4*(i-1)+1 | R*4 | First Pedestal value of the i th ADC channel |
| +4*(i-1)+2 | R*4 | First Pedestal sigma of the i th ADC channel |
| +4*(i-1)+3 | R*4 | Second Pedestal value of the i th ADC channel |
| +4*(i-1)+4 | R*4 | Second Pedestal sigma of the i th ADC channel |

**CMOF** gives the corresponding offline number of an ADC described by its "hardware" pointer The directory pathname to access this bank is "//CDRT/RUN_COND/CALO_ST".

| Displacement | Word Type | Description |
|---|---|---|
| +1 | I*4 | Offline IDentifier for ADC channel pointed by the number 1 |
| . | I*4 | |
| . | I*4 | |
| . | I*4 | |
| 65535 | I*4 | Offline IDentifier for ADC channel pointed by the number 65535 |

**CMAP** is bank containing the Calorimeter Map file. It is like a integer matrix with (number of channels) rows and 11 columns.

| Displacement | Word Type | Description |
|---|---|---|
| +1 | I*4 | Version number of the map file |
| +2 | I*4 | Total number of channels NC |
| +3 | I*4 | First DAQ Electronic Pointer "n" |
| +4 | I*4 | Cart type of Electronic channel pointed by "n" |
| +5 | I*4 | VME address of the "Electronic channel" |
| +6 | I*4 | Crate Number of the "Electronic channel" |
| +7 | I*4 | Slot Number of the "Electronic channel" |
| +8 | I*4 | Subaddress of the "Electronic channel" |
| +9 | I*4 | Corresponding Detector Number of the "Electronic channel" |
| +10 | I*4 | Corresponding Side of the "Electronic channel" |
| +11 | I*4 | Module Number of the "Electronic channel" |
| +12 | I*4 | Row corresponding to the "Electronic channel" |
| +13 | I*4 | Column of the "Electronic channel" |
| . | I*4 | |
| . | I*4 | |
| . | I*4 | |
| 11*(i-1)+3 | I*4 | ith DAQ Electronic Pointer "m" |
| 11*(i-1)+4 | I*4 | Cart type of the ith "Electronic channel" |
| 11*(i-1)+5 | I*4 | VME address of the ith "Electronic channel" |
| 11*(i-1)+6 | I*4 | Crate Number of the ith "Electronic channel" |
| 11*(i-1)+7 | I*4 | Slot Number of the ith "Electronic channel" |
| 11*(i-1)+8 | I*4 | Subaddress of the ith "Electronic channel" |
| 11*(i-1)+9 | I*4 | Detector Number of the ith "Electronic channel" |
| 11*(i-1)+10 | I*4 | Side (1:2) of the ith "Electronic channel" |
| 11*(i-1)+11 | I*4 | Module Number of the ith "Electronic channel" |
| 11*(i-1)+12 | I*4 | Row of the ith "Electronic channel" |
| 11*(i-1)+13 | I*4 | Column of the ith "Electronic channel" |
| . | I*4 | |
| . | I*4 | |
| . | I*4 | |

**CHVM** is the Calorimeter High Voltage Map for the photo multiplier.

**Drift Chamber**

### Trigger

**TSTC** is the Tubes STroc Configuration bank

```
=======================================================================
        TSTC (ZEBRA) bank - Tubes STroc Configuration bank
=======================================================================
The TSTC bank stores the Stroc Configuration (# of read channels and
4 values for each channel) of the Cosmic Ray Stand.

The bank has the following Characteristics :

Bank Name :         "TSTC"
Bank Number :          1
Bank Data Length :  1+TSTCBS*NSTRO                     ! in 4 bytes words
Bank Type :         Integer                            ! I*4
```

The bank is then structured in 1 data block. Each word in the bank can be located through the displacement pointers shown below. A description of these parameters can be found in K$CRS:CRTSTC.INC.

```
Displacement         Word Type    Word Description
C-----------------------------------------------------------------------
C Data Block
C-----------------------------------------------------------------------
           STRNUC              I*4          STRoc NUmber of Channels

           STRCNU+                          STRoc Channel NUmber
         / TSTCBS*(Iblock-1)   I*4
        /
NSTRO  /   COORCH+                          COORdinate Read by the
blocks \   TSTCBS*(Iblock-1)   I*4          CHannel
        \
         \ NUSTRC+                          NUmber of STrips Read by
           TSTCBS*(Iblock-1)   I*4          a Channel

           CHSTAT+                          CHannel STATus
           TSTCBS*(Iblock-1)   I*4          (0=O.K., 1=Ovf, -1=Unf)
C-----------------------------------------------------------------------
```

### Beam

2) For the Calibration Database

### Calorimeter

**CPMG** contains the constants to compute the photo multiplier gains following the equation $gain = (constant * HV)^{\kappa}$. It is like a real matrix NC*3, where NC is the total number of PMs. The directory pathname to access this bank is "//CDRT/CALIBRATION/CALO_LT".

| Displacement | Word Type | Description |
|---|---|---|
| +1 | R*4 | Photo multiplier number of the first PM |
| +2 | R*4 | first constant to compute the gain of the first PM |
| +3 | R*4 | second constant "$\kappa$ to compute the gain of the first PM |
| . | R*4 | |
| . | R*4 | |
| +3*(i-1)+1 | R*4 | Photo multiplier number of the ith PM |
| +3*(i-1)+2 | R*4 | first constant to compute the gain of the ith PM |
| +3*(i-1)+3 | R*4 | second constant "$\kappa$ to compute the gain of the ith PM |
| . | R*4 | |
| . | R*4 | |

**CMIP** is the bank containing the Minimum Ionizing Peaks values and their sigmas. It is a bank of 2*(number of channels) or 4*(number of channels) reals, depending if there are one or two sets of ADC. The directory pathname to access this bank is "//CDRT/CALIBRATION/CALO_ST". The desription of the bank in the case of one set of ADC is:

| Displacement (I*4) | Word Type | Description |
|---|---|---|
| +1 | R*4 | MIPs value of the first ADC |
| +2 | R*4 | MIPs sigma of the first ADC |
| . | R*4 | |
| . | R*4 | |
| . | R*4 | |
| +2*(i-1)+1 | R*4 | MIPs value of the ith ADC |
| +2*(i-1)+2 | R*4 | MIPs sigma of the ith ADC |

In the case of two sets of ADC :

| Displacement | Word Type | Description |
|---|---|---|
| +1 | R*4 | First MIPs value of the first ADC channel |
| +2 | R*4 | First MIPs sigma of the first ADC channel |
| +3 | R*4 | Second MIPs value of the first ADC channel |
| +4 | R*4 | Second MIPs sigma of the first ADC channel |
| . | R*4 | |
| . | R*4 | |
| . | R*4 | |
| +4*(i-1)+1 | R*4 | First MIPs value of the ith ADC channel |
| +4*(i-1)+2 | R*4 | First MIPs sigma of the ith ADC channel |
| +4*(i-1)+3 | R*4 | Second MIPs value of the ith ADC channel |
| +4*(i-1)+4 | R*4 | Second MIPs sigma of the ith ADC channel |

**Drift Chamber**

**Trigger**

3) For the Geometry database

**Calorimeter**

**Drift Chamber**

4 parameters banks used for the cell parametrisation.
DPR1,DPR2,DPR3,DPR4

1 bank containing the radii used for the cell parametrisation
DPRD

The directory pathname to access thess banks is "//CDRT/GEOMETRY/DRCH_LT". These 5 banks have the same real format corresponding to a matrix 6 by 36 as follows :

| Displacement | Word Type | Description |
|---|---|---|
| +1 | R*4 | Number of different cell shapes: NC |
| +2 | R*4 | Number of $\phi$-section per shape: NP |
| +3 | R*4 | 1st $\phi$-section Parameter for the First kind of cell shape |
| +4 | R*4 | 2nd $\phi$-section Parameter for the First kind of cell shape |
| . | R*4 | |
| . | R*4 | |
| . | R*4 | |
| +NP+2 | R*4 | NPth $\phi$-section Parameter for the First kind of cell shape |
| +NP+3 | R*4 | 1st $\phi$-section Parameter for the Second kind of cell shape |
| . | R*4 | |
| . | R*4 | |
| . | R*4 | |
| +2*NP+2 | R*4 | NPth $\phi$-section Parameter for the Second kind of cell shape |
| . | R*4 | |
| . | R*4 | |
| +NP*(i-1)+2+j | R*4 | j th $\phi$-section parameter for the i th kind of cell shape |
| . | R*4 | |
| . | R*4 | |

**Trigger**

**SCPO** gives the Scintillators Positions.

```
C=====================================================================
        SCPO (ZEBRA) bank - Scintillators Position bank
C=====================================================================
The SCPO bank stores the Scintillators Position vs the "sostegno" ;
the "sostegno" position vs the calorimeter being stored on TUPO bank.

The bank has the following Characteristics :

Bank Name :         "SCPO"
Bank Number :          1
```

```
Bank Data Length :  SCPOBS*NSCI                          ! in 4 bytes words
Bank Type :         Real                                 ! R*4
```

The bank is then structured in 1 data block. Each word in the bank can be located through the displacement pointers shown below. A description of these parameters can be found in K$CRS:CRSCPO.INC.

```
Displacement         Word Type    Word Description
C-----------------------------------------------------------------------
C Data Block
C-----------------------------------------------------------------------
         / XSCSOP+                             X SCintillator vs SOstegno
        /  SCPOBS*(Iblock-1)   R*4             Position
       /
NSCI  /    YSCSOP+                             Y SCintillator vs SOstegno
blocks\    SCPOBS*(Iblock-1)   R*4             Position
       \
        \  ZSCSOP+                             Z SCintillator vs SOstegno
         \ SCPOBS*(Iblock-1)   R*4             Position
C-----------------------------------------------------------------------
Note : The positions are in cm.
```

**TVSZ** gives the streamer tubes number in function of the z position.

```
C=======================================================================
        TVSZ (ZEBRA) bank - Tubes Vs Strips Z position bank
C=======================================================================
The TVSZ bank stores the Z Strips Position vs the Streamer Tubes border
(1st strip position, 2 relevant distances between strips) for each of
the 4 plans.

The bank has the following Characteristics :

Bank Name :         "TVSZ"
Bank Number :          1
Bank Data Length :  1+TVSZBS*NPLZ                        ! in 4 bytes words
Bank Type :         Real                                 ! R*4
```

The bank is then structured in 1 data block. Each word in the bank can be located through the displacement pointers shown below. A description of these parameters can be found in K$CRS:CRTVSZ.INC.

```
Displacement         Word Type    Word Description
C-----------------------------------------------------------------------
C Data Block
C-----------------------------------------------------------------------
           ZSTRWI              R*4             Z STRips WIdth

           1ZSTPO+                             1st Z STrip PPsition
         / TVSZBS*(Iblock-1)   R*4
        /
NPLZ   /   D16ZST+                             Distance between 16th Z STrip
```

```
blocks \   TVSZBS*(Iblock-1)   R*4         and the following one
        \
         \ DIZSTR+                         DIstance between Z STRips
           TVSZBS*(Iblock-1)   R*4
C----------------------------------------------------------------------
Note : The width and positions of the strips are in cm.
       Iblock = 1 (same positions for the 4 streamer tubes)
```

**TVSY** gives the streamer tubes number in function of the y position.

```
C======================================================================
        TVSY (ZEBRA) bank - Tubes Vs Strips Y position bank
C======================================================================
The TVSY bank stores the NSTY (NSTY = 256) Y Strips Position vs the
border of each Streamer Tube plan of the Cosmic Ray Stand.

The bank has the following Characteristics :

Bank Name :         "TVSY"
Bank Number :          1
Bank Data Length :  1+TVSYBS*NSTY            ! in 4 bytes words
Bank Type :         Real                            ! R*4
```

The bank is then structured in 1 data block. Each word in the bank can be located through the displacement pointers shown below. A description of these parameters can be found in K$CRS:CRTVSY.INC.

```
Displacement         Word Type    Word Description
C----------------------------------------------------------------------
C Data Block
C----------------------------------------------------------------------
           YSTRWI              R*4         Y STRips WIdth

  NSTY   | YSTPBP+                         Y STrip vs Plane Border
  blocks | TVSYBS*(Iblock-1)   R*4         Position
C----------------------------------------------------------------------
Note : The width and positions of the strips are in cm.
```

**TUDI** gives the streamer tubes dimension.

```
C======================================================================
        TUDI (ZEBRA) bank - Tubes DImension bank
C======================================================================
The TUDI bank stores the Streamer Tubes Planes DImension
(display purposes).

The bank has the following Characteristics :

Bank Name :         "TUDI"
Bank Number :          1
Bank Data Length :  TUDIBS*(NPL)             ! in 4 bytes words
```

```
Bank Type :          Real                             ! R*4
```

The bank is then structured in 1 data block. Each word in the bank can be located through the displacement pointers shown below. A description of these parameters can be found in K$CRS:CRTUDI.INC.

```
Displacement         Word Type    Word Description
C----------------------------------------------------------------------
C Data Block
C----------------------------------------------------------------------
           XPLDIM+                            X PLane DIMension
         / TUDIBS*(Iblock-1)   R*4
        /
 1     /   YPLDIM+                            Y PLane DIMension
 block \   TUDIBS*(Iblock-1)   R*4
        \
         \ ZPLDIM+                            Z PLane DIMension
           TUDIBS*(Iblock-1)   R*4
C----------------------------------------------------------------------
Note : The dimensions are in cm.
       Iblock = 1 (same dimensions for the 4 streamer tubes)
```

**TUPO** gives the streamer tubes position.

```
C======================================================================
        TUPO (ZEBRA) bank - Tubes POsition bank
C======================================================================
The TUPO bank stores the Streamer Tubes Planes Position vs the
"sostegno" and the "sostegno" position vs the calorimeter.

The bank has the following Characteristics :

Bank Name :          "TUPO"
Bank Number :           1
Bank Data Length :  TUPOBS*(NPL+NSOS)                ! in 4 bytes words
Bank Type :          Real                             ! R*4
```

The bank is then structured in 1 data block. Each word in the bank can be located through the displacement pointers shown below. A description of these parameters can be found in K$CRS:CRTUPO.INC.

```
Displacement         Word Type    Word Description
C----------------------------------------------------------------------
C Data Block
C----------------------------------------------------------------------
           XPLSOP+                            X PLane vs SOstegno Position
         / TUPOBS*(Iblock-1)   R*4
        /
NPL    /   YPLSOP+                            Y PLane vs SOstegno Position
blocks \   TUPOBS*(Iblock-1)   R*4
        \
         \ ZPLSOP+                            Z PLane vs SOstegno Position
```

```
            TUPOBS*(Iblock-1)   R*4

            XSOCAP+                                 X SOstegno vs CAlorimeter
          / TUPOBS*(Iblock-1)   R*4             Position
         /
NSOS    /   YSOCAP+                                 Y SOstegno vs CAlorimeter
blocks  \   TUPOBS*(Iblock-1)   R*4             Position
         \
          \ ZSOCAP+                                 Z SOstegno vs CAlorimeter
            TUPOBS*(Iblock-1)   R*4             Position
C-----------------------------------------------------------------------
Note : - The positions are in cm.
       - NSOS = 1
```