EUROPEAN ORGANIZATION FOR NUCLEAR RESEARCH

CB-note 288

Introduction to SVX in Locater and Cboff

R. Ouared, M.Doser

CERN, European Organisation for Nuclear Research, Geneva, Switzerland

September 20, 1996

Abstract : The integration of the Silicon Vertex detector in the offline reconstruction package is described to the user. Initialisation, pre-data handling and reconstruction processes are described and complete details provided.

1 Introduction

The introduction of the SVX related code in CBOFF and LOCATER requires to define new commons, to update old others, to initialize the geometry properly, to handle the raw data and the lookup table so that one can deal directly with genuine clusters and eventually to perform the reconstruction processes sequentially and coherently. Sections **2**, **3** and **4** discuss **initialization**, **pre-data handling** and **reconstruction** processes respectively. Sections **5** and **6** give respectively digest and conclusion of all the forthcoming discussion. Flowcharts and tables are drawn at the very end. The description of appropriate **commons** is coming up along with the reliable topics.

2 Initialization

The initialization procedure aims to setting on the geometrical parameters of the Silicon Vertex Detector for either strip and to loading in pedestal and threshold information in specific commons.

2.1 Geometrical Initialization

The lack of precise alignment information regarding the SVX detector allows one to assume at this starting point an ideal detector centered right at the origin of the Crystal Barrel detector and a perfect alignment along with the beam axis (z). As the result of that, one is able to draw the whole 15-PADs SVX detector by simple and periodical rotations of the first pad around the z–axis once the original transverse–plane $r - \phi$ position of the latter is known. This operation is performed in the subroutine TCINIT in two steps the first dealing with the strips belonging to the first pad and the second dealing with the strips of the remaining pads. The algorithm used is the following:

1. The first pad:

   - Find out the polar coordinates $r - \phi$ of strip number **127** in the first pad and calculate the corresponding cartesian coordinates x, y so that :

     $$x(127) = r(127)\cos\phi$$

     $$y(127) = r(127)\sin\phi$$

     The polar coordinates are by default set to 1.2 m and 90° centigrades. They are stored in common TPPRMS through the 2–dimensional array OSVXTP(2). Possibility to tune theses values is given using the CBOFF input cards with the following syntax: SVXY 1.2 90.0.

- Compute the cartesian and polar coordinates for the strips in the first pad starting from those of strip 127 and deduce the corresponding polar coordinates. For the i^{th} strip the cartesian and polar coordinates are explicitely calculated this way:

$$\begin{aligned}
x(i) &= x(127) + i\ a\ \cos\gamma \\
y(i) &= y(127) + i\ a\ \sin\gamma \\
r(i) &= \sqrt{x(i)^2 + y(i)^2} \\
\varphi(i) &= \arctan\frac{y(i)}{x(i)}
\end{aligned}$$

  a and γ being respectively the $50\mu m$ strip interspace and the angular direction of the pad plane with respect of x axis. The direction of the pad has its origin at strip 127 and points to the left–hand side towards strip 0 . The variable γ takes into account an additional 1° inclination of the pad plane in the transverse plane. The values of a, $\frac{e}{2}$ with e the $370\mu m$ module depth and the inclination are respectively stored in the common TPPRMS through the 3–dimensional variable ASVXTP(3).

2. The other pads:

- Compute the cartesian and polar coordinates for the strips of the remaining pads using a simple clockwise rotation around the z–axis of the first–pad strips coordinates. The rotation angle depends on the relative locations of the strip in a given pad and of the pad within the 15-PADs detector. More explicitely the algorithm is as follows:

$$\begin{aligned}
\varphi(i) &= \varphi(j) - 0.41888\ k \qquad i \geq\ 128; k \geq\ 1 \\
x(i) &= r(j)\ \cos\varphi(i) \\
y(i) &= r(j)\ \sin\varphi(i)
\end{aligned}$$

  where the indices i, j, k are respectively the absolute strip number **(0–1919)**, the local strip number **(0–127)** and the local pad number **(0-14)**. The rounded number 0.41888 is the one equivalent to 24° centigrades (the 15 modules accomodate uniformly in the 360° transverse plane).

- The coordinates x, y, r and φ for all the strips are respectively stored in the common TPPRMS in the arrays XSVXTP(0:1919), YSVXTP(0:1919), RSVXTP(0:127) and FSVXTP(0:1919). Note that in the ideal case all the strips with the same locations in their respective pads have the same radii.

2.2 Loading SVX table

Pedestal Tables are calculated and stored at the beginning of each run. Pedestals and Thresholds, **(2048)** short words a piece, are stored sequentially by full **7+3+1=11** Bits logical address. The values are multiplied by **16** (4 *bits fractional resolution*) before storage [2] . Information provided by the 'RTVX' table bank (the structure is drawn in table 1) is loaded right at the begining of the running code in common CBTBSV. That is performed in routine CBCASV if the 'RTVX' Zebra address is found out. Pedestals and Thresholds are filled up for all the strips either. Possibility is given to the user to loading in an external table by means of the following locater input card for the syntax is: SVTB .T.. In the later, the external file must be set to logical unit **84** and must contain pedestals and thresholds respectively in the free format for either strip. Following the same definition of the threshold used in 'RTVX' table, the one $th(i)$ written down in the external file should comply with the following equation:

$$th(i) = ped(i) + 5\,\sigma(i)$$

where $ped(i)$ is the value of the pedestal of strip i and $\sigma(i)$ the effective pedestal standard deviation. Then the Locater routine TCSVTA will load up the common CBTBSV in this particular case with the values of $ped(i)$ and $\sigma(i)$. That will be used in the Pre-data handling part of the SVX related code.

3 Pre-Data Handling

The Pre-Data Handling part of this code has the capability to dealing with all kind of Raw Data whatever they are, regarding the **0–suppression** and the **Pedestal–subtraction** features of the online digitization process. The data are online, originally registered in 'RVTX' Zebra bank according to the bank structure definition drawn in tables (3,4,5). Offline, a preliminary pass steered by the Locater routine TSNRAW is made so as to keep in the higher level Raw Bank 'RSVX' information for **genuine clusters** only, with the same structure as in 'RVTX' bank.The data in 'RSVX' are now completely 0–suppressed and pedestal–subtracted. 'RSVX' bank will be the starting point for the reconstruction process regarding the vertex detector later on. The 0–supression, pedestal–suppression procedure is made sequentially starting with the pedestal subtraction performed by the routine TSPSUB, then the common mode suppression is performed by the routine TSMODE and finally the 0–suppression is implemented by the routine TSZERO in which 'RSVX' is lifted up and prepared for later use.

3.1 Pedestal subtraction

At this stage the pedestal subtraction is performed for the vertex detector frontside only. The treatment of backplane amplitudes are postponed till in TSZERO. The procedure in TSPSUB is done as follows:

- Check the format of the data. That will initiate the implementation of the subtraction procedure if the format is even(**a600, a602**). The subtraction's already done for odd format data (**a601, a603**) (see table 2).

- Get the 10–bits Raw amplitude $ampls(i)$ from 'RVTX' bank and implement the subtraction. The new strip pedestal–subtracted amplitude is stored in 1920–dimensional array *pampl* in common TCSVRA and is derived this way:

$$pampl(i) = ampls(i) - ped(i)$$

  where ped(i) is the pedestal value for strip i stored in common CBTBSV. The values in *pampl* array provide the entries for the common mode suppression procedure which is the next step. In the case where no pedestal subtraction is needed the array *pampl* is equivalent to *ampls*.

- To handle properly the signal, **sign (overthreshold)**, **valid (over pedestal)** and **over (overflow)** bits are stored also in common TCSVRA through the 1920–dimensional arrays LSIGN, LVALID and LOVER respectively for later use in TSZERO.

3.2 Common mode suppression

The common mode suppression is implemented in routine TSMODE for non 0–suppressed data type wich have the format **a600, a601** either. The algorithm of the procedure, with a preliminary status, comply with the following steps:

- For each pad, the mean value of the pedestal–subtracted amplitude is computed.
- The pedestal–subtracted amplitudes are compared each to the mean value. Only those with a deviation less than **80** in absolute value are kept in the set of strips which will serve to compute the baseline shift. The shift is simply the amplitude average corresponding to this particular set.
- The common mode suppression is implemented and the new value of the amplitude is stored in the 1920–dimensional array *campl* defined in common TCSVRA and computed as follows:

  $$campl(j + 128\ (i - 1)) \ = \ pampl(j + 128\ (i - 1)) - cmod(i)$$

  where i the pad number index **(1–15)**, j the local strip index **(1–128)** and $cmod(i)$ the baseline shift for pad number i. The array *cmod* is defined in common TCSVRA.

3.3 Zero suppression

The zero suppression conducted by the subroutine TSZERO has to keep significant signals only leading to the storage of **real clusters** in 'RSVX' bank. Even 'RVTX' $pedestal - unsubtracted$; $zero - suppressed$ Data need to pass through this level to garantee the purity of the cluster[1]. This is achieved in 2 rounds, the first keeping all strips with more than 2σ signals and the last defining and keeping clusters for one of the contigous strip has more than 5σ. The gap between clusters is a crucial parameter to be set by the user as well as the heigths of significant signals. To choose externally the minimum gap between clusters use the following locater input card syntax: SVCL 1. The value by default is 1. The thresholds are chosen using the syntax SVTH 2. 5., the first number dealing with first level signals and the second dealing with the higher level signals, that is a proper definition of a cluster. The default is respectively 2. and 5.. The algorithm for such exercice is implemented as follows:

- Select strips with amplitudes *campl* larger than 2σ, valid and not overflown. The threshold σ must be strictly larger than zero. For each selected strips, amplitudes, absolute strip numbers (1–1920) and effective thresholds are stored in temporarily variables.

[1]Sometimes, whole pads fire and 'RVTX' **a603**-format Data taking become completely misleading then.

- The **'clusterization'** process looks for any signal larger than 5σ amongst the set of contigous strips selected as discussed above. The total number of clusters and hit wires that come out are used in the next step for 'RSVX' Zebra bank space allocation.
- Lift up the higher level bank 'RSVX' with the same structure as 'RVTX' and load it up with information regarding the brand newly defined clusters.

4 Reconstruction process

The basic Locater routines for the reconstruction process related to the vertex detector are namely TSVPOS and TSCNCT both called by TCTRAK. Shortly speaking, TSVPOS makes up out of the raw bank 'RSVX' 3 extra banks each with specific information useful for the whole event reconstruction process. Those banks deals respectively with **'hits'**, **'clusters'** and **'backplane–amplitudes'** features of the detector. They are bound to the detector master bank 'TVXT' along with specific address offsets, repectively at **-1**, **-2** and **-3**. The **'clusters'** bank is however the basic one used for the tracking. TSCNCT connects the clusters to the relialable JDC tracks before the helix fit tracking procedure is being implemented.

4.1 TSVPOS

The key job of TSVPOS is to compute the position of each 'RSVX' defined cluster in the Crystal Barrel reference frame. This is done using the absolute strip locations for the given cluster and the amplitude associated to each strip, the latter providing weighted contribution to the center of gravity calculation of the cluster position. Some other useful information also are derived and filed where they belong. Debug options are available and can be actived by +SELF cards in the kumac file. The whole exercice is done by TSVPOS in 4 steps:

1. **Lift up the master bank** 'TVXT'

   - The top level 'TVXT' bank is lifted right at the beginning of the code at offset **-8** of 'HTJD' bank:

   $$\mathrm{LTVXT} = \mathrm{LQ}(\mathrm{LHTJD} - 8)$$

   Its structure is drawn in table 6.

2. **Lift and fill up the** 'BACKPLANE–AMPLITUDES' **bank**

   - Lift at offset **-3** from 'TVXT' the $(\mathbf{16} \times \mathbf{2})$ long words BACKPLANE–AMPLITUDE bank. In this bank, for each of the possible 16 pads 2 long words are reserved: the first, **pedestal subtracted amplitude** and the second, **signal** $\div$ **background** [table 8]. The second word is null for the time being. Only the first word is filled if the signal is valid. The **valid** and **overflow** bits are stamped in bits **14** and **15** of the first word respectively. Note that backplane amplitudes are measured with **12–bits** ADC's.

3. **Lift and fill up the** 'HITS' **bank**

- Lift at offset **-1** from 'TVXT' the (**number of fired strips** $\times$ **2**) long words 'HITS' bank. In this bank, every strip records with 2 long words: the first, **the absolute strip number (1-1920)** and the second, its **amplitude** [table 7].

4. **Lift and fill up the** 'CLUSTERS' **bank**

- For each cluster k compute its local center of gravity as in the following:

$$\begin{aligned} \mathrm{X_L}(k) &= \frac{\sum_i x_i\, A_i}{\sum_i A_i} \\ \mathrm{Y_L}(k) &= 0. \end{aligned}$$

where i, x_i and A_i are respectively the fired strip index in the cluster, the local position of the strip and its amplitude. The local position of the strip is given by

$$x_i = j\ a$$

where j and a are respectively the local strip number **(0-127)** and the **50μm** strip interspacing.

- Compute the center gravity of the cluster both cartesian and polar coordinates in CB reference frame as follows:

$$\begin{aligned} \mathrm{X_{CB}}(k) &= \mathrm{X_0} + \mathrm{X_L}(k) * \cos\alpha \\ \mathrm{Y_{CB}}(k) &= \mathrm{Y_0} + \mathrm{X_L}(k) * \cos\beta \\ \varphi(k) &= \arctan\frac{\mathrm{Y_{CB}}(k)}{\mathrm{X_{CB}}(k)} \\ \mathrm{R}(k) &= \sqrt{\mathrm{X_{CB}}(k)^2 + \mathrm{Y_{CB}}(k)^2} \end{aligned}$$

where $\mathrm{X_0}$ and $\mathrm{Y_0}$ are the CB-coordinates of the first strip in the corresponding pad, $\cos\alpha$ and $\cos\beta$ the director cosines of the pad line pointing to the last strip, $\varphi(k)$ and R(k) are the CB-polar coordinates of the cluster. Turning on the +self card TSVPOS allows to print out the intermediate results of the computing process.

- Compute the errors on the centre of gravity of the clusters. This is very preliminary and only a rough estimate is given in this first round:

$$\begin{aligned} \delta x &= 0.0100cm \\ \delta y &= 0.0100cm \\ \delta\varphi &= \frac{\sqrt{\delta y^2\ \mathrm{X_{CB}}^2 + \delta x^2\ \mathrm{Y_{CB}}^2}}{\mathrm{R}^2} \\ \delta\mathrm{R} &= \delta y \end{aligned}$$

- Lift the **'clusters'** bank at address offset -2 from LTVXT and fill it up with cluster's information (**16-words length per cluster**). See table 9 for details. The content of this bank can be printed out by turning on in the kumac file the +self card TSTVTX.The printout of the raw bank RSVX is also possible if the +self card RSRSVX in turned on in the kumac file.

4.2 TSCNCT

As it's been mentionned in the former, the aim of TSCNCT is to match clusters with the relevant JDC tracks. Practically speaking, that will affect the content of only the first word in **'clusters'** bank, which is set to 0 at the TSVPOS level and should contain the track number associated with the given cluster. This will complete the set of hits feeding on the helix fit. The +SELF card TSCNCT let it printing out the intermediate results. The matching process which has been simplified down to an acceptable level for the first use of the Silicon Vertex detector in CRYSTAL BARREL experiment, goes through the following steps:

1. Computing the distance of clusters to tracks

   - Each track is extrapolated inwards to get the polar position of its impact in the crossed pads. First, the parameters of the circle fit (κ,c^2,ψ_0) are taken out from TCTK bank. Each point on the track for which r and ϕ are the polar coordinates complies with the constraint[1] :

     $$\kappa\ r + \frac{\kappa\ c^2}{r} + \sin(\phi - \psi_0) \quad = \quad 0$$

     Particularly, the expected azimuthal angles ϕ_{ex} of clusters are find out by setting r to their corresponding center of gravity radius as measured in TSVPOS. The angular distance *dist* of each cluster to this track is given by

     $$dist \quad = \quad |\,\varphi - \phi_{ex}\,|$$

     where φ is the measured value of the azimuthal angle of the cluster centre of gravity. Only distances smaller than a given value MATCH are stored in 2-dimensional array MATCH(I,J) where the index I stands for cluster number **(1-50)** and J stands for track number **(1-20)**. The value of MATCH is set to 0.6 radians and is not restrictive at all at this level where no SVX calibration's available.
     One should note at this stage of the matching process that a cluster could be associated to more than one track or/and a track associated with more than 2 clusters. The next steps tackle these problems.

2. Matching the cluster to one track

   - The closest track to the cluster is matched to the latter. The slected track ID is then stored in the first word of **'clusters'** bank. However, many clusters could be attached to the same track while normally not more than 2 clusters are allowed, the extrem case being when the supporting pads overlap.

3. Matching the track with no more than 2 clusters

   - At this level of the matching process only cases with more than 2 clusters per track are treated. The 2 closest clusters are kept attached to the track and the farthest released. The track ID allocated to the latter is 0 and this kind of clusters remains free and won't participate to the helix fitting process.

5 The digest

In this paragraph, the main parts of the Silicon Vertex Detector related code are summarized. These parts are essentially **commons**, **routines**, **environment setting**, **input cards**, **+self cards** and **bank addresses**. Further, the flowcharts are corresponding to CBMAIN, CBLOOP and CBPHYS branches and are drawn in figures **1**, **2** and **3** respectively. These branches only quote the main routines used to implement the SVX related code which are commented in the next subsections.

5.1 New commons

- TCSVXY: defines the variable FSTRIP (**2-dim,real**) to steers the (r,φ) position of strip 128 in the first pad.
- TCSVTH: declares the variable SVSIGM (**2-dim,real**) to define clusters using 2 levels of signals.
- TCSVTB: defines the variable LSVTBX (**logical**) to attach external pedestal table.
- TCSVOF: defines the variable IOFSET (**2-dim,integer**) to set the pointers to both the first short word fired strip address in 'RVTX' bank and the first short word Pedestal Data address in 'RTVX' table bank. This was useful essentially when the code modelling process was running out simultaneously with the vertex detector bank structures buildup. Indeed, the word offsets were changing in such a way that makes it mandatory the introduction of a new steering card SVOF letting the code fluently adjust to this dynamical situation. This is now useless for the user.
- TCSVRA: allows the communication between the **Pre-Data Handling** routines dealing with **Pedestal-Subtraction** and **Zero-Suppression**. AMPLS(1920), PAMPL(1920), CAMPL(1920), CMOD(15) are the **real**-type variables, NSHORT and NFORM the **integer** ones, LSUBPED, LVALID(1920), LOVER(1920) and LSIGN(1920) the **logical** ones.
- TCSVCL : defines the variable ISGAP (**integer**) to steer the minimum gap between different clusters.
- CBTBSV: stores out of the lookup tables Pedestal and threshold informations with proper conversions during the initialization phase in either CBCASV or TCSVTA routines. The variables used are ISPEDS(0:1919) and ISTHRE(0:1919) and both are **integers**.

5.2 Updated Commons

- CBLINK: defines the new bank addresses LRVTX and LRSVX for the raw banks and LTVXT for the reconstruction bank.
- CBDBCO: defines new SVX variables if possibly the control of inputs from database is made. The variables are GTVTCC and TBVTCC both (**logicals**), and IMVTCC (**integer**).
- TPPRMS: defines the Vertex Detector positionning parameterization as performed in TCINIT routine during the geometrical initialization phase. The variables used are all **real**: XSVXTP(0:1919), YSVXTP(0:1919), RSVXTP(0:127), FSVXTP(0:1919), ASVXTP(3) and OSVXTP(2).

- TCLIFT: defines the Zebra bank formats for 'RSVX' and 'TVXT' in the routine ZBFORM. The variables are both **integers**: MRSVX(5) and MTVXT(5).
- TRKPRM: defines the length size of the 'BACKPLANE-AMPLITUDES', 'HITS' and 'CLUSTERS' banks. The variables used are all **integers** and are respectively: LENBP, LENSV and LENCL. They are definitely set to **2**, **2** and **16** respectively and recorded in the data part of the header bank 'TVXT' [table 6].
- TCFLAG: allows the switch to the SVX code instead of the PWC one in TCTRAK and TCHXLD routines.

5.3 New routines

- RSRVTX: dumps 'RVTX' content in usable format.
- RSRSVX: called by TSVPOS to dump 'RSVX' content in usable format.
- TSTVTX: called by TSVPOS to dump 'TVXT' content in usable format.
- TSTVTA: called by TCINIT to load up Pedestal and Threshold information from external lookup tables.
- CBCASV: called by CBTABS to load up Pedestal and Threshold information from tape lookup tables.
- TSNRAW: main steering routine for Pre-Data Handling process called by TCTRAK.
- TSPSUB: called by TSNRAW to perform *pedestal substraction*.
- TSMODE: called by TSNRAW to perform *common mode suppression*.
- TSZERO: called by TSNRAW to perform *zero suppression*.
- TSVPOS: called by TCTRAK to work out *Backplane – Amplitudes*, *Clusters* and *Hits* information from 'RSVX' bank.
- TSCNCT: called by TCTRAK to perform the SVX–JDC *matching process*.

5.4 Updated routines

- CBINIT: called by CBMAIN to set the defaults for SVX calibration constants.
- CBFFGO: called by CBINIT to get the SVX steering data cards with FFREAD.
- ZBINIT: called by CBINIT to define individual division pointers for the banks 'RVTX', 'RSVX' and 'TVXT'.
- GETEVT: called by CBLOOP to get events and connect on the standard bank addresses 'LRVTX', 'LRSVX' and 'LTVXT'. Refers to subsection **5.6** for bank address settlement.
- ZBFORM: called by CBINIT to format the banks 'RSVX' and 'TVXT'.

- TCINIT: called by CBINIT to initialize the SVX tracking section and to set on its geometrical configuration.
- CBSRUN: called by CBLOOP to printout some SVX information.
- CBTABS: called by CBLOOP to load in SVX table lookup information.
- CJINIT: called by TCINIT to process the SVX steering input cards.
- TCTRAK: called by CBPHYS to track in the *Silicon Vertex Detector*.
- TCHELX: called by TCTRAK to process the helix fit including in the SVX clusters.

5.5 Setting Environnement, Input Cards and +Self Cards

Besides all the usual environnement setting, an extra one is also needed if any external SVX lookup table's used for the syntacs is:

setenv FORT84 filename

The basic input cards are used with the following syntax:

SVTB .F.	**(the default)**	
SVCL 1	**(the default)**	
SVXY 1.2 90.	**(the default)**	
SVOF 2 130	**(the default)**	**(don't use it)**
SVTH 2. 5.	**(the default)**	

To turn on the debug options some +self cards could be used:

+SELF, TSVPOS; +SELF, TSCNCT; +SELF, RSRSVX and +SELF, TSTVTX.

5.6 Bank Addresses

The banks addresses are defined in the following way:

$$\begin{aligned}
\mathrm{LRVTX} &= \mathrm{LQ}(\mathrm{LHRAW}-15)\\
\mathrm{LRSVX} &= \mathrm{LQ}(\mathrm{LHRAW}-16)\\
\mathrm{LTVXT} &= \mathrm{LQ}(\mathrm{LHTJD}-8)\\
'\mathrm{HITS}' &= \mathrm{LQ}(\mathrm{LTVXT}-1)\\
'\mathrm{CLUSTERS}' &= \mathrm{LQ}(\mathrm{LTVXT}-2)\\
'\mathrm{BACKPLANE}-\mathrm{AMPLITUDES}' &= \mathrm{LQ}(\mathrm{LTVXT}-3)
\end{aligned}$$

It is not mandatory to define 'RTVX' address by hand. The Zebra function LZFIDH deals with that automatically in routine CBTABS.

6 Conclusion

This version of the SVX related code has been achieved during the early use of the Silicon Vertex Detector Hardware. The reconstruction efficiency of this code should then improve after solving definitely both alignment and digitization uncoding problems. Making reasonable errors estimate and improving the matching process will also to some extent improve the situation.
The geometrical configuration of the detector is to be sure different from the setup defined in TCINIT for an ideal situation is being assumed :**(the center of SVX is at (x=0.,y=0.), z-axis is not taken into account, the positions of all pads are given by simple geometrical rotation from the first pad**).
During the 0-suppressed Data-Taking runs, the 'RVTX' clusters' were enlarged to include 2 neighbour strips (one for each side) to allow for proper common-mode supression. This case is not treated by TSMODE routine because this couple of strips yet have not been included in 'RSVX' clusters made up through the different Pre-Data Handling stages.
Moreover, the center of gravity of clusters have been measured with very rough errors in TSVPOS although a formalism of error estimate taking into account correlations between variables has been implemented in the code but not used so far.
Last but not the least, the matching process in routine TSCNCT is somehow inefficient to dealing with very close tracks crossing the same pad although Monte–Carlo events accomodate with that rather well. However, further steps in the code should be implemented to handle properly those released clusters associated willy nilly with JDC tracks. The efficiency for each strip is assumed to be 1. Also, some extra code should be implemented to account for variables $\frac{\mathbf{signal}}{\mathbf{background}}$ in $'\mathbf{backplane-amplitudes}'$ bank and both φ' and $\frac{\mathbf{dE}}{\mathbf{dx}}$ in $'\mathbf{clusters}'$ bank, all having been temporarily set to zero.

7 Acknowledgement

We want to thank Bruce Barnett and Christian Voelcker for very useful discussions, comments and achievements which eased very much the implementation of the SVX related code in a usable fashion. Many thanks also to Lucciano Pavarotti for his particular tribute to *Miss Sarajevo* particularly enjoyed during this technical note typing phase.

References

[1] Locater Manual.

[2] Bruce Barnett e-mails to Crystal Barrel collaboration

Short Word number	**Content**
1	$Format:\ Matches\ Data\ Format$
2	$Short\ Word\ Count\ (\mathbf{4232})$
3, 4	$Bitmask\ showing\ detectors\ readout$
5	$Number\ of\ detectors$
6	$Number\ of\ strips\ per\ detector\ (\mathbf{128})$
7	$Tailcount$
8	$threshold\ offset$
9	$sigma\ factor$
10	$alignement$
11 $\rightarrow$ 14	$time/date\ of\ pedestal\ data$
15 $\rightarrow$ 40	$reserved\ for\ future\ use$
41 $\rightarrow$ 56	$Backplane\ ADC\ Pedestal\ Data$
57 $\rightarrow$ 72	$Backplane\ ADC\ Threshold\ Data$
73 $\rightarrow$ 104	$\mathbf{16}\ Long\ words,\ CRAM\ Ident$
105 $\rightarrow$ 136	$\mathbf{16}\ Long\ words,\ Backplane\ ADC\ Ident$
137 $\rightarrow$ 2184	$\mathbf{2048}\ Short\ Words\ Pedestal\ Data$
2185 $\rightarrow$ 4232	$\mathbf{2048}\ Short\ Words\ Threshold\ Data$

Table 1: **'RTVX' Bank Structure**

Format Word	**Meaning**
$a600$	$Pedestal-Unsubtracted;\ Non-Zero-Suppressed$
$a601$	$Pedestal-Subtracted;\ Non-Zero-Suppressed$
$a602$	$Pedestal-Unsubtracted;\ Zero-Suppressed$
$a603$	$Pedestal-Subtracted;\ Zero-Suppressed$

Table 2: **Format Words Definition**

Short Word number	**Content**
1	$Format\ Word$
2	$Total\ number\ of\ short\ words\ (including\ this\ word\ and\ the\ format\ word)$
3	$Bitmask:\ Backplane\ ADCs\ above\ threshold$
4	$Padding$
$5 \rightarrow 20$	$Backplane\ ADCs.\ One\ for\ each\ Multistrip\ detector$
$21 \rightarrow 36$	$Baseline\ offset:\ One\ for\ each\ Multistrip\ detector.\ The\ average\ value\ for\ channels\ not\ overthreshold\ or\ overflowing,\ multiplied\ by$ **16**
37	$Address\ of\ first\ hit\ in\ this\ cluster$
38	$Number\ of\ short\ data\ words\ in\ first\ cluster\ of\ contigous\ hits\ (\ excluding\ this\ world\ and\ the\ address\ word\ )$: **N**
$38+1 \rightarrow 38+N$	$N\ Data\ Words\ for\ this\ cluster$
$39+N$	$Address\ of\ first\ hit\ in\ the$ **2nd** $cluster$
$40+N$	$Number\ of\ short\ data\ words\ in$ **2nd** $cluster$: **M**
$40+N+1 \rightarrow 40+N+M$	$M\ Data\ Words\ for\ this\ cluster$
...	...

Table 3: **'RVTX' Bank Structure**

Bit Number	**Content**
15	$Flag:\ 0$
14	$Overflow$
13	$Valid\ (\ Over\ pedestal\ )$
12	$Over\ Threshold$
$11-0$	**12** $Data\ bits$

Table 4: **Data Word Format**

Bit Number	**Content**
15	$Flag:\ 1$
$13-14$	$Reserved\ Flag\ Bits\ \ (\ zero\ )$
$11-12$	$Reserved\ Address\ Bits\ \ (\ zero\ )$
$8-10$	$Module\ ID$ **(0-7)**
7	$Sub-Module$ **(0: Top, 1: Bottom)**
$6-0$	$Strip/Channel\ number$ **(0-127)**

Table 5: **Address Word Format**

Offset	Content
LQ(LTVTX − 3)	*Link to* 'BACKPLANE AMPLITUDES' *bank address*
LQ(LTVTX − 2)	*Link to* 'CLUSTERS' *bank address*
LQ(LTVTX − 1)	*Link to* 'HITS' *bank address*
IQ(LTVTX + 1)	*Total number of fired strips in event*
IQ(LTVTX + 2)	*Total number of clusters in event*
IQ(LTVTX + 3)	*Length size per strip in* 'HITS' *bank* (LENSV = 2)
IQ(LTVTX + 4)	*Length size per cluster in* 'CLUSTERS' *bank* (LENCLS = 16)
IQ(LTVTX + 5)	*Length size per pad in* 'BACKPLANE − AMPLITUDES' *bank* (LENBP = 2)

Table 6: **'TVXT' Bank Structure**

Long Word Number	Content
1	*First strip number* **(0-1919)**
2	*First strip amplitude*
3	*Second strip number*
4	*Second strip amplitude*
...	...

Table 7: **'Hits' Bank Structure**

Long Word Number	Content
1	*First pad Backplane Amplitude*
2	*First pad Signal* $\div$ *Background*
3	*Second pad Backplane Amplitude*
4	*Second pad Signal* $\div$ *Background*
...	...
29	*Backplane Amplitude for pad* 15
30	*Signal* $\div$ *Background for pad* 15

Table 8: **'Backplane-Amplitudes' Bank Structure. All the even words are set to 0 for the time being.**

Long Word Number	**Content**
1	*track number for first cluster. It is set to* **0** *in* TSVPOS. *Its value will change later in* TSCNCT *if successful matching occurs*
2	*First strip number in first cluster* **(0-1919)**
3	*Number of hits in first cluster*
4	*Center of gravity* $x -$ *coordinate for first cluster*
5	*Center of gravity* $y -$ *coordinate for first cluster*
6	*Center of gravity* $z -$ *coordinate for first cluster. It is set to* **0** *in* TSVPOS. *Its value will change later in* TCHELX *if helix fit successfully done*
7	*First cluster Radius*
8	*First cluster* φ
9	*First cluster* φ'. *It is set to* **0** *in* TSVPOS *for the time being. This variable will help if anymore precise cluster – track matching required. It provides the direction of the track if energy loss rate in* SVX *known*
10	δx *for first cluster*
11	δy *for first cluster*
12	xy *covariance for first cluster. it is set to* **0** *for the time being*
13	$\delta\varphi$ *for first cluster*
14	δr *for first cluster*
15	*Energy loss rate* $\frac{dE}{dx}$ *of matching track in* SVX. *It is set to* **0** *for the time being*
16	*Total amplitude of first cluster*
1	*Track number for second cluster*
2	*First strip number in second cluster*
3	*Number of hits in second cluster*
...	...
...	...

Table 9: **'Clusters' Bank Structure.** $\frac{dE}{dx} = \frac{e\Delta}{\cos\varphi'}$ **where e is the thickness of** SVX **and** Δ **the average energy loss rate.**

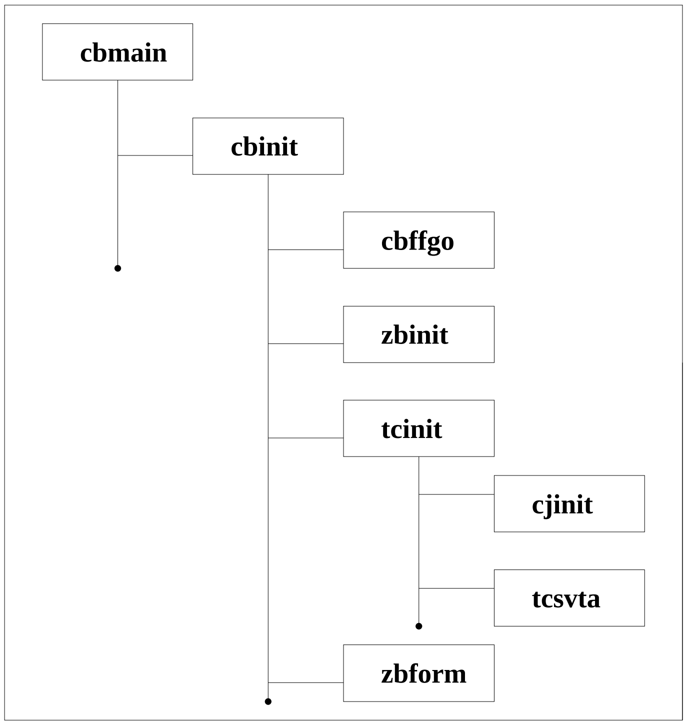

Figure 1: CBMAIN *branch*

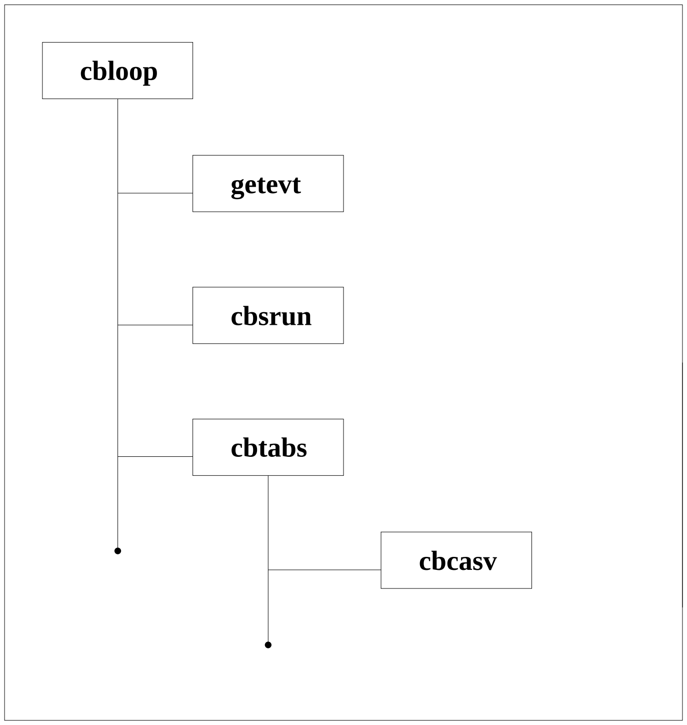

Figure 2: CBLOOP *branch*

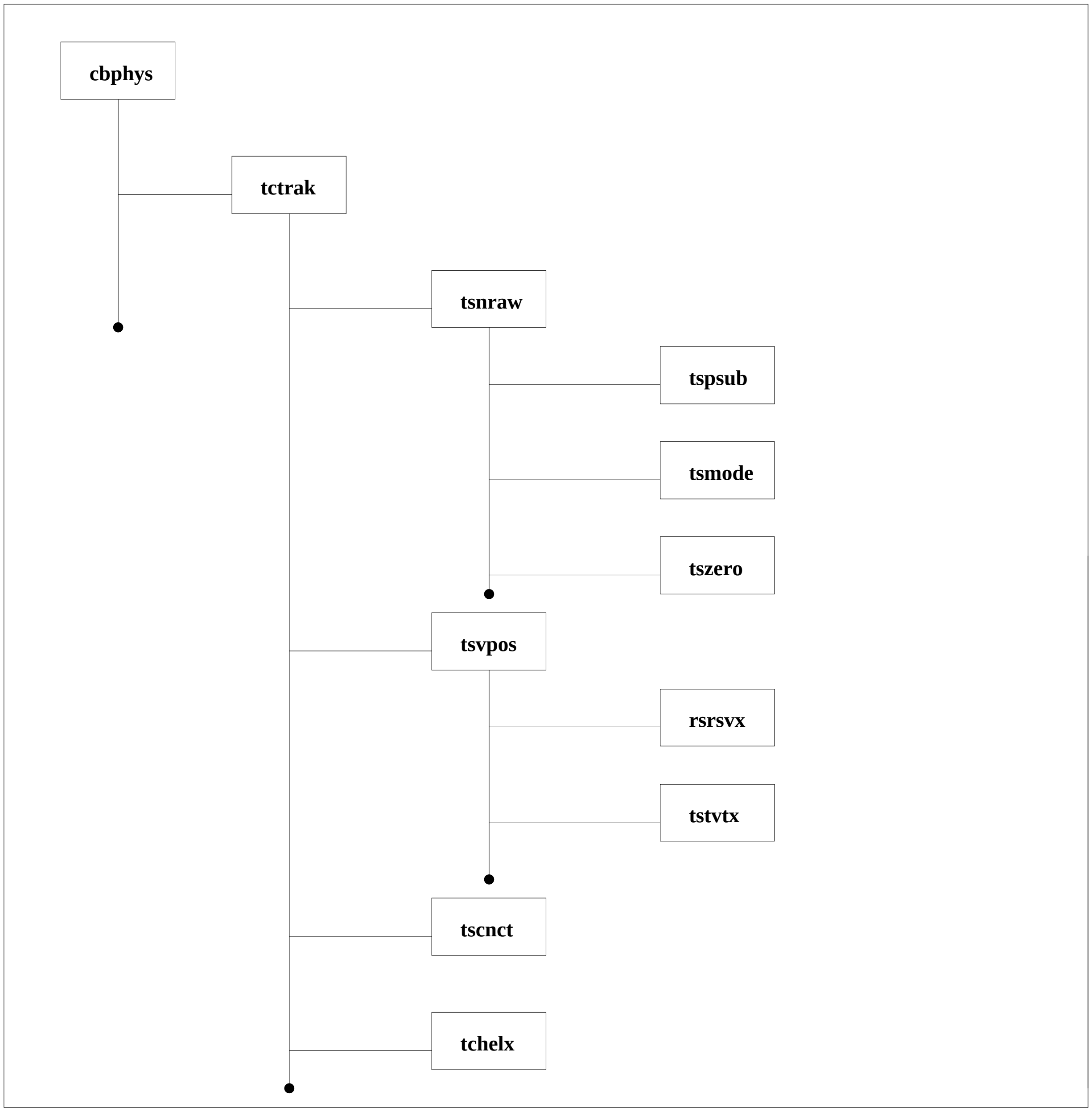

Figure 3: CBPHYS *branch*