16/OFFL-0318

13-Dec-94

The OPAL DST (OD) processor in ROPE

(Version 4.08)

A. Buijs

Abstract

The aim of the OD processor is to collect a summary of the event data in a well defined format and to provide routines for easy access to these data. The user will need to have knowledge of FORTRAN and PATCHY, but not of ZEBRA. This summary will be written to the Data Summary Tapes (DSTs).

Contents

Chapter 1

Introduction.

The OD processor consists of a set of routines which unpack the various ROPE detector banks and store a selected set of quantities in the DST banks. Banks provided are: 'GNRL', 'CTRK', 'PRES', 'ECAL', 'TOF ', 'HCAL', 'MUON', 'FDET', 'VRTX', 'ATRK', 'SIW' and 'DTE'. About 99% of the proposed quantities are currently filled.

Following is an overview of the various pieces of information available in the DST banks.

CHARGED TRACKS: the DST information is obtained from the CT banks, with the exception of the endpoint information, which comes from CJ. Also when there is no momentum in CT, the CJ momentum is taken. Vertex constraint fits are done by default, and can be undone by a call to ODUNCZ. Silicon information is stored in a separate temporary bank. It can be swapped with the real CTRK bank by a call to ODTOSI

ELECTROMAGNETIC CLUSTERS: the em clusters are obtained from the merged em banks. The corrections applied to the energies are for angles only. all clusters, including clusters with negative energy (noise hits) are stored. In addition, from OD400 onwards, compressed raw block data are stored as well.

TIME-OF-FLIGHT COUNTERS: Some general timing information is stored in the general bank. TOF hits are also stored separately. BETA is calculated using the extrapolated track length. There is probably more stored for the TOF than for any other device...

PRESAMPLER CLUSTERS: Presampler clusters are stored in separate

banks.

HADRONIC CLUSTERS: the hadronic clusters are obtained from the HB, HE and HP banks. The banks allow for strip clusters and tower clusters in the same format. At present, no merging between strips and towers is done. Therefore simply adding all energies of all clusters will lead to double counting.

MUONS: muons are obtained from muon chamber banks. These banks have already been merged with the hadron calorimeter muons. However, the track parameters are only taken from the muon chambers, because these are more reliable.

FORWARD DETECTOR: A short block with calorimeter information is provided. Some drift chamber information is also stored, but is not yet (OD404) calibrated for physics use.

VERTICES: The parameters for the event vertex are stored in the GNRL block. For each secondary vertex, a VRTX block and a corresponding ATRK block are stored. Track parameters for the two parts of kinked tracks are stored in the ATRK block. Vertices can be remade by calling CXBAT and then ODVRTX.

SILICON-TUNGSTEN CALORIMETER: Tracks, electromagnetic and hadronic clusters are stored in the SIW block (starting with OD407). These are obtained from the SW global cluster banks.

EXTRAPOLATION and MERGING. The merging of detector components is done by the CE processor. A weight is stored for each association, using the errors from the track parameters, and the cluster position, and the uncertainty due to multiple scattering. The bank of the detector with the finer resolution stores stores the weight for the best association to a device with lower resolution. The bank of the latter stores the number of associations with the former.

NOTE: in this scheme, one hit (cluster) can be associated to several tracks.

It is NOT possible to run the OD processor without the CE processor.

Chapter 2

FFREAD Cards

In order to activate the DST processor, the ODBLK card is required. When reading

ODBLK To build the 'DST' blocks, accepts:

'GNRL'	To get general event information (event No., time, energy ...) in the GNRL block.
'CTRK'	To get charge track information in the CTRK blocks (one per CT track).
'TOF.'	To get time-of-flight information in the TOF blocks (one per TOF bar hit).
'PRES'	To get presampler information in the PRES blocks (one per presampler cluster).
'ECAL'	To get electromagnetic calorimeter information in the ECAL blocks (one per EM cluster).
'HCAL'	To get hadron calorimeter information in the HCAL blocks (one per hadronic cluster).
'MUON'	To get muon tracking information in the MUON blocks (one per muon candidate).
'VRTX'	To get vertex information in the VRTX blocks (one per vertex found).

'ATRK' To get pseudo track (possible primary corresponding to measured secondary) information in the ATRK blocks (one per pseudo track found).

'FDET' To get forward detector information in the FDET blocks (one per forward detector cluster). (Silicon-tungsten clusters were also included in the FDET block for versions 404-406.)

'SIW' To get silicon-tungsten calorimeter clusters and tracks (one per cluster or track).

'USER' To get user defined DST blocks. ODSBLK('USER',...) must then be called in a user routine as described in chapter 3.

'ALL.' To get all mentioned blocks.

ODPBLK Accepts 3 integers n1,n2,n3 and prints the 'DST' blocks for events from n1 to n2 with an increment of n3 (default is 0 0 1).

ODHIST ON Books and fills some histograms in a directory '//PAWC/OD'.

ODCUTS Sets the values of the cuts used by ODGBLK and ODGBLH. These routines are used in ODPHYS to calculate sphericity etc., and also in the terse printout. See section **??** for more information.

ODBKTH 1=50 Will set the EB packing threshold to 50 MeV

ODBKTH 2=20 Will set the EE packing threshold to 20 MeV

ODSMGLP Sets the values of the smearing parameters used by ODSMGL for tracks in Monte Carlo events. By default all the parameters are 1. The five parameters correspond to the smearing of kappa, phi0, d0, tanlam and z0. See section **??** for more information.

Chapter 3

USAGE and ACCESS to the DATA

The user writes a set of USER routines to do his analysis.

USER routines are in the standard form foreseen by ROPE, *i.e.* a set of routines USINIT, USSETR, USBAT, USFIN, etc. The user links (loads, binds) his routines and the OD library into ROPE. It should then be transparent to him whether the input is DST only or raw (processed) data.

There are three methods to access the data:

- Through statement functions.
- By access routines ODFBLK, ODNBLK.
- By means of a COMMON block.

The three methods can be mixed within the same USER code. Access through statement functions is most efficient.

Please note that the **order** of the various quantities on the DST is not the same as in the documentation and may change. Each variable must be accessed using the predefined PARAMETER. (Right: PY=RC(JCPY,ITR), wrong: PY=RC(JCPX+1,ITR)).

3.1 STATEMENT FUNCTIONS

Statement functions are a feature of FORTRAN to allow the user to use simple functions in his code without actually CALLing the routine. Decent FORTRAN compilers will replace statements functions by inline code, which is very efficient.

Statement functions MUST be included AFTER the last type declaration and BEFORE the first executable statement. The OD statement functions are declared in the PATCHY sequence ODSFUN. In addition, once per USBAT, the routine ODFUNS should be called to set up the parameters for the statement functions.

Statement functions exist for all DST blocks. The names of these functions are:

```
IG, IC, IP, IE, IT, IH, IM, IF, IV, IA, IS, ITMC, IJMC       and
RG, RC, RP, RE, RT, RH, RM, RF, RV, RA, RS, RTMC, RJMC, RXMC
```

The ones starting with I are the integers, the ones with R are the reals. They correspond to the usual blocks G=GNRL, C=CTRK, etc. ITMC and RTMC refer to the TREE bank, and IJMC and RJMC to the JETS bank. RXMC gives the information of the XTRA banks. (See GOPAL Primer)

3.1.1 Example using statement functions.

```
      SUBROUTINE USBAT(LEVEL)
+SEQ,DECLARE.
+SEQ,ODPAR.
      INTEGER ITRK,LEVEL
+SEQ,ODSFUN.
      IF (LEVEL .NE. 0) RETURN
      CALL ODFUNS
      IF (NGNRL .EQ. 0) RETURN
      CALL HFILL (11,RG(jgebea),0.,1.)
      DO 10 ITRK = 1,NCTRK
        CALL HFILL(12,ALOG10(RC(jcp,ITRK)),RC(jcdedx,ITRK),1.)
   10 CONTINUE
      END
```

This very simple subroutine first checks if there is a GNRL block. block.

If it exists, it histograms the beam energy, and for each track the dE/dx versus the log of the momentum.

The lower-case variables are defined in ODPAR. Lower-case is only used here for demonstration purpose; do not use it in real code. The variables starting with J are offsets in the data blocks, as explained below. The variables NGNRL, NCTRK, NPRES, NECAL *etc.*, are the number of blocks present for each type. These words are filled in ODSFUN by ODFUNS. Note that all variables declared in ODSFUN cannot be declared as local variables anymore. Note that statement functions cannot be used on the left of an equal sign, and note that they are not the same as an array. The latter means that you cannot call a routine FRED(RC(jcpx,ITRK)) if FRED expects an array.

The ODSFUN keep sequence contains declarations AND statement functions. Therefore it must be put **after** declarations, COMMON and SAVE statements, but **before** other statement functions and executable code. You may also use separately ODSFUN1, which contains all the declarations and ODSFUN2, which contains all the statement functions. In that case you need +SEQ,ROBANK as well. ROBANK is already included in ODSFUN, although that is against the rules. Similarly, GCBANK is included in ROBANK, so you need both the ROPE CARDS file and the GEANT CARDS file in your PATCHY step.

When accessing a data record without OD blocks (this happens frequently in Monte Carlo events), care must be taken not to access non-existing storage locations with statement functions. The check on NGNRL as shown above is sufficient for this.

3.2 ACCESS ROUTINES

Access to OPAL DST data is possible through a set of utility routines.

3.2.1 ODFBLK

```
      CALL ODFBLK(BLKTYP,IBLK,NINTE,IDAT*,NIMOV*,NREA,RDAT*,NRMOV*)

function        Retrieves data for given DST blocks.

input  BLKTYP   Block type, ('CTRK','GNRL',etc...)
```

```
        IBLK      Block number (track number in case of 'CTRK')
        NINTE     Expected number of integer data words
        NREA      Expected number of real data words

output  IDAT      Array containing INTEGER data on return
        NIMOV     Number of INTEGER data words moved (=0 if not existing)
        RDAT      Array containing REAL data on return
        NRMOV     Number of REAL data words moved (=0 if not existing)
```

3.2.2 ODNBLK

```
        CALL ODNBLK(BLKTYP,NBLK*)

function          Returns the number of blocks of type BLKTYP

input   BLKTYP    Block type

output  NBLK      Number of blocks found
```

3.2.3 Example Using Access Routines

```
      SUBROUTINE USBAT(LEVEL)
+SEQ,ODPAR.
      REAL         RGDAT(nrgnrl), RTKDAT(nrctrk), P
      INTEGER      IGDAT(nignrl), ITKDAT(nictrk)
      INTEGER      NIMOV, NRMOV, NCTRS, ITRK, LEVEL
      IF (LEVEL .NE. 0) RETURN
      CALL ODFBLK ('GNRL',0,mignrl,IGDAT,NIMOV,mrgnrl,RGDAT,NRMOV)
      CALL HFILL (11,RGDAT(jgebea),0.,1.)
      CALL ODNBLK ('CTRK',NCTRS)
      DO 10 ITRK = 1,NCTRS
        CALL ODFBLK ('CTRK',ITRK,mictrk,ITKDAT,NIMOV,mrctrk,RTKDAT,NRMOV)
        IF (NRMOV .EQ. 0) GOTO 10
        P = SQRT(RTKDAT(jcpx)**2+ RTKDAT(jcpy)**2+ RTKDAT(jcpz)**2)
        CALL HFILL (12,P,RTKDAT(jcdedx),1.)
   10 CONTINUE
      END
```

This very simple subroutine first fetches the general ('GNRL') block. If it exists, it histograms the beam energy, and for each track the dE/dx versus momentum.

The lower-case variables are defined in ODPAR. The variables starting with J are offsets in the data blocks, as explained below. The PARAMETERS starting with N are the maximum size of a block. (Since OD400 words may be deleted from the blocks) The parameters starting with M are the current number of data words in a particular block. The second character is I or R, referring to either the integer or the real data part of the block. So, MIGNRL is the number of integer data words in the General block, MRECAL is the number of real data words in the electro magnetic cluster block.

NOTE: The character arguments in all OD subroutines must be UPPER CASE!

3.3 COMMON BLOCKS

3.3.1 USER Defined Common Blocks.

The DST data can also be passed through a user defined common block. In that mode the user defines his common block in USSETR (called by ROPE) using the routine ODBOOK:

```
        CALL ODBOOK(BLKTYP,VARTYP,VARADR,VARLEN)

function         Book a variable from the DST banks.

input  BLKTYP    Block type (e.g. 'CTRK', 'ECAL', etc...)
       VARTYP    Variable type (e.g. 'PX  ','DEDX', etc...)
       VARADR    Address of the variable (array) receiving the data.
       VARLEN    length of the array VARADR.
```

The VARTYPes allowed are given by the last four characters of the offset parameters, see DATA definitions.

3.3.2 Example Using Common Blocks

This example performs the same function as in 4.1.3.

```
      SUBROUTINE USSETR
      COMMON /COUSER/NCTRS,EBEAM,P(100),DEDX(100)
      CALL ODBOOK('GNRL','NCTR',NCTRS,1)
      CALL ODBOOK('GNRL','EBEA',EBEAM,1)
      CALL ODBOOK('CTRK','P   ',P,    100)
      CALL ODBOOK('CTRK','DEDX',DEDX, 100)
      END

      SUBROUTINE USBAT(LEVEL)
      COMMON /COUSER/NCTRS,EBEAM,P(100),DEDX(100)
      INTEGER ITRK
      IF (LEVEL .NE. 0) RETURN
      CALL ODFILL
      CALL HFILL (11,EBEAM,0.,1.)
      DO 10 ITRK = 1,NCTRS
   10 CALL HFILL (12,P(ITRK),DEDX(ITRK),0.,1.)
      END
```

The routine USSETR is used to define the common block /COUSER/, containing the DST variables NCTR, EBEA, P and DEDX. Note that another user may define a different common block with different DST variables. Before USBAT is called, the common block is filled as requested.

Chapter 4

Utilities.

4.1 ODPBLK

```
      CALL ODPBLK(BLKTYP,IBLK)

function        Prints a block contents.

input  BLKTYP   Block type
       IBLK     Block number (0=all blocks)
```

The printout gives for each block first the integer part, then the real part. The mnemonics used are the same as described below.

CALL ODPEVT prints the whole event.

4.2 ODGBLK and ODGBLH

These routines are used to define good tracks and clusters in the context of computing event quantities such as total energy, thrust, sphericity, etc. They may not be ideally suited to one's analysis.

```
   LOGICAL LGOOD, ODGBLK
   LGOOD = ODGBLK(BLKTYP,IBLK)

function        Selects good tracks and good ECAL and HCAL clusters
```

```
input  BLKTYP    Block type
       IBLK      Block number (0=all blocks)
```

The cuts are based on note OPAL/PHYS/0020 by P. Maettig, D. Schaile and G. VanDalen and are controlled by the FFREAD card `ODCUTS`. The following cuts are applied to charged tracks (the default values are listed in parentheses):

PT	Minimum transverse momentum. (0.15 GeV/c)
CST	Maximum $\|cos(\theta)\|$. (0.94)
D0	Maximum $\|d_0\|$. (5.0 cm)
Z0	Maximum $\|z_0\|$. (99.99 cm)
CH2	Maximum $\|\chi^2\|$ in r-ϕ plane. (999.0)
NHT	Minimum number of CJ hits. (40.0)

Cuts on ECAL and HCAL quantities are:

ERW	Minimum raw energy for ECAL clusters. (0 GeV)
EEB	Minimum (corrected) energy for EB clusters. (100 MeV)
EEE	Minimum (corrected) energy for EE clusters. (200 MeV)
EHC	Minimum energy for HCAL clusters. (400 MeV)

In addition, EE clusters are required to contain at least two blocks. To avoid double-counting, ECAL and HCAL clusters with associated charged tracks are flagged as bad by ODGBLK.

The cuts above appear in the order they are specified on the ODCUTS card.

An additional routine, ODGBLH, provides a second set of cuts used by ODPHYS in computing some event quantities.

```
LOGICAL LGOOD, ODGBLH
LGOOD = ODGBLH(BLKTYP,IBLK)
```

```
function         Selects good tracks and good ECAL and HCAL clusters

input  BLKTYP    Block type
       IBLK      Block number (0=all blocks)
```

The cuts for this routine can be set with arguments 11 through 19 of the ODCUTS card and are based on those of the Higgs group as defined in Phys. Lett. B253 (1991) 511-523. Those pertaining to charged tracks are:

PTH	Minimum transverse momentum. (0.1 GeV/c)
CTH	Maximum $\|cos(\theta)\|$. (0.966)
D0H	Maximum $\|d_0\|$. (2.5 cm)
Z0	Maximum $\|z_0\|$. (50.0 cm)
NHH	Minimum number of CJ hits. (20.0)

In ODGBLH the cut on CJ hits is on the maximum of NHH and half the total possible number of hits.

Cuts on ECAL and HCAL quantities are:

EBH	Minimum raw energy for EB clusters. (170 MeV)
EEH	Minimum raw energy for EE clusters. (250 MeV)
HTH	Minimum uncorrected energy for HB and HE clusters. (600 MeV)
HPH	Minimum uncorrected energy for HP clusters. (2 GeV)

In ODGBLH ECAL and HCAL clusters may be associated with charged tracks.

4.3 ODPHYS

ODPHYS computes event quantities which are stored in the event header as well as in the GNRL block. For a description of event header quantities see the ROPE manual.

Three different sets of track and cluster selection criteria are used in computing event quantities in ODPHYS. For header words IINCTR, IINMUO and IINVTX, all corresponding DST blocks are used. The ECAL energy sum (word IIECAL) use corrected energies. The HCAL energy sum (word IIHCAL) uses the corrected energies of tower clusters.

The thrust (IITHRU), cos θ of the thrust axis (IICTHR) and missing energy vector components (IIMEPx) are computed using good clusters and tracks as defined by ODGBLK (see section **??**). The sums of energies of good clusters and of momenta of good charged tracks (IISxx) as well as the counts of good clusters and charged tracks (IINxx) use the criteria of ODGBLH. (Here xx is CT, EB, EE, HT, HPT, HS or HPS.)

4.4 Numbering of Clusters.

Presampler and Hadron Calorimeter Banks: The presampler banks PBGC and PEGC are numbered in a single sequence, according to their appearance in the linear chain, starting with PBGC. The number of the Hadron calorimeter clusters is similarly counted starting with the hadron barrel, then the endcap, and finally the pole tip.

Electromagnetic Calorimeter Banks: The lead glass clusters are numbered as they appear in the EMCL banks. All EMCL banks go to DST for now.

4.5 Constraint Track Refitting

A user callable routine is provided to refit the charged tracks in the CTRK blocks. Details of this routine are given in OPAL technical note TN029.

The calling sequence for `ODCTRE` is

```
ODCTRE( ITRACK, IOPTZ, IOPTXY, IERR)
```

The input arguments are defined as follows:

ITRACK number of CTRK block to update. Non-positive values of ITRACK are interpreted as follows:

$i > 0$ update track **i**

0	update all tracks
-1	update all tracks assigned to primary vertex
-2	update all tracks passing ODGBLK cuts

IOPTZ steering code for Z constraints. If lowest order bit is set, a constraint is done to the primary vertex in Z. Setting the next higher bit results in a constraint to the CJ endpoint, provided it has a valid value. Higher order bits are ignored.

IOPTXY steering code for XY constraints. If lowest order bit is set, a constraint is done to the primary vertex in XY. Higher order bits are ignored.

The routine updates the contents of the CTRK blocks specified by the input arguments. The CTRK words updated are:

```
(for Z constraints)              (for XY constraints)
 JCFTYP                           JCFTYP
                                  JCPX   JCPY
 JCPZ   JCP    JCDP               JCPZ   JCP    JCDP
                                  JCKAPP JCPHI0 JCD0
 JCTLAM JCZ0
                                  JCEM11 JCEM21 JCEM22
                                  JCEM31 JCEM32 JCEM33
 JCEM44 JCEM54 JCEM55
 JCCHI3                           JCCHIR
 JCDEDX JCDDED

          The fit type JCFTYP is updated as follows:
 new_fit_type = old_fit_type + 16*IOPTZ + 256*IOPTXY
```

The fit type is updated in order to prevent the same constraint from being applied twice. In addition an error flag `IERR` is returned. If all is well `IERR` will be zero. A return value less than zero indicates a problem with the input information; for example, it is forbidden to apply the same constraint to a track more than once. If `IERR`> 0, there was a problem in applying the constraint to CTRK block `IERR` (perhaps due to the constrained fit failing).

The constraint fit can be undone by a call to ODUNCZ(ITRK,ZUNC), where ITRK is the track number and ZUNC the $z-$value after the unconstraining.

4.6 Usage of Silicon data in the track fit.

The DST blocks output from the current PASS3 re-ROPE of the 1991 data contain the usual non-silicon track and vertex parameters. However, some extra information is being output along with the normal DST banks which allows the SILICON track and vertex parameters to be accessed as well.

A utility routine ODTOSI is now available for use with DST analysis programs to convert the DST track parameters from non-silicon to silicon and vice-versa. Physics quantities such as thrust and sphericity are also updated using the silicon tracks, and an option 'CX' in ODTOSI allows primary and secondary vertices to be updated also if needed. ODTOSI usage is described below.

The silicon alignment constants being used for the re-ROPE are believed to be good to 15 microns internally and 45 microns externally. The track fit quality is still under study, but early indications from a study of about 500 mu-pair events are that the impact parameter resolution is improved by 20-30 percent with silicon.

The DST arrangement described above is intended as a temporary measure while the silicon detector alignment is still in progress. For the longer term, two new integer words (JCSIH1 and JCSIH2) have been added to the CTRK DST track block. These contain the SI hit number for the inner and outer silicon barrels respectively. If these words are non-zero, then an inner (JCSIH1) and/or outer (JCSIH2) barrel silicon hit was included in the track fit. These words should be zero for the current re-ROPE DST output but are filled by ODTOSI if silicon tracks are requested.

Silicon data begin during period 26.

To fill all the standard DST blocks with silicon tracks and vertices, use

```
CALL ODTOSI ('SI','CX',IFAIL)
```

where IFAIL is returned non-zero if an error is detected. After this call, all the standard DST access methods (ODFBLK, ODSFUN internal functions etc) should work as usual, but now the silicon rather than non-silicon parameters should be returned. If you are not interested in primary or secondary vertex information use:

```
CALL ODTOSI ('SI','  ',IFAIL)
```

instead. To convert the DST information back to its original non-silicon state, use:

```
      CALL ODTOSI (’  ’,’  ’,IFAIL)
```

where the second argument is now irrelevant.

ODTOSI can be called as often as necessary on each event - it simply toggles back and forth between SI and non-SI states.

Starting in 1993, the silicon microvertex detector read out hits in both rphi and z. The user has the option to update tracks for the rphi hits only, or for both the rphi and z hits. To include only rphi hits, use:

```
      CALL ODTOSI(’si’,’  ’,IFAIL)
```

To include both rphi and z hits use:

```
      CALL ODTOSI(’SI’,’  ’,IFAIL)
```

as before. For data from runs before 1993, ’si’ and ’SI’ are equivalent. One can also use ’SX’ in place of ’si’. When using only rphi hits for 1993 data and later it is essential to include CX in one’s link list.

4.6.1 Example of ODTOSI

```
 +SEQ,ODSFUN.
         ...
*--      find the non-silicon primary vertex
      CALL ODTOSI (’  ’,’  ’,IFAIL)
      X = RG(JGVX)
      Y = RG(JGVY)
      Z = RG(JGVZ)
*--      find the SI primary vertex
      CALL ODTOSI (’SI’,’CX’,IFAIL)
      XSI = RG(JGVX)
      YSI = RG(JGVY)
      ZSI = RG(JGVZ)
```

```
         ...
      DO 100 I = 1, NCTRK
*--      find d0 for non-SI track fit for track I
      CALL ODTOSI ('  ','  ',IFAIL)
      D0   = RC(JCD0,I)
*--      find d0 for SI track fit for track I
      CALL ODTOSI ('SI','CX',IFAIL)
      D0SI = RC(JCD0,I)
  100 CONTINUE
         ...
```

Using ODTOSI should involve only a negligible CPU overhead, though the first call for an event with the 'CX' option will invoke a CALL to CXBAT. Second and subsequent calls for a given event involve only a few ZSHUNTs. If you use the 'CX' option, the CX library should be included in your LINK/LOAD/BIND... step.

The subroutine ODCSHI retrieves SI hit numbers and positions from the ODCS bank. This routine does not access the GNRL or CTRK DST banks and so can be used independently from ODTOSI.

Calling sequence (to be called within USBAT):

```
      SUBROUTINE ODCSHI (ICTRK,IBARRL,IHITX,IHITZ,HLOCAL,HOPAL)
 Input arguments:

      ICTRK     = track number
      IBARRL    = Barrel number (1 is inner barrel (ladders  1-11)
                                 2 is outer barrel (ladders 12-25))
 Output arguments:

      IHITX     = SI xy hit number (=0 if none found)
      IHITZ     = SI rz hit number (=0 if none found)
      HLOCAL(3) = x,y,z of SI hit in LOCAL coordinates
      HOPAL(3)  = x,y,z of SI hit in OPAL  coordinates
```

4.6.2 Silicon raw data.

For debugging purposes, some silicon raw data are very temporarily stored on the DST as well. A new bank, ODSI hangs from position 3 (LLODSI)

of the GNRL bank. This bank contains two packed integers per SI cluster. The encoding is as follows (all entries are integers):

```
Word 1: (the least significant bit is #1)
   Bits  1-16: 100*X(cluster), X=local SI coordinate
   Bits 17-19: Number of Channels
   Bits 20-23: Offset, int(X(cluster)) - X(first channel in cluster)
   Bits 24-31: Resolution (NYI)

Word 2:
   Bits  1-10: f(Pulse Height), logarithmic conversion good to +- 0.4 %
   Bits 11-18: f(Noise),            "           "        "   + +- 1.5 %
   Bits 19-23: Ladder Number
   Bits 24-25: Detector Number
   Bits 26-31: Hit Number (0==> Original hit number >63).
```

The last three entries when used as ILAD*1000 + IDET*100 + IHIT recover the original hit number. The current SI hit pointers in the CTRK banks can thus be mapped to the entries in the ODSI banks. In both words, the sign bit is unused to insure correct representation on machines which might automatically extend a 32 bit signed integer to 64 bits.

4.7 Likelihood of matching between CV, CJ and CZ

This likelihood is small. However, the results are stored in a packed integer which can be unpacked with a routine ODCTMA:

```
      CALL ODCTMA(IC(JCMLLH,ICTRK),CVLLV,CVLLW,CZLL)

 INPUT :  IC(JCMLLH,ICTRK)   The packed likelihood
 OUTPUT:  CVLLV              Matching LLH for CV axial wires - CJ
 OUTPUT:  CVLLW              Matching LLH for CV stereo wires - CJ
 OUTPUT:  CVLLZ              Matching LLH for CJ - CZ
```

4.8 Refitting of secondary vertices

Martin Jimack has provided a routine which will refit secondary vertices constraining the resultant vector to the primary vertex:

```
      CALL ODV0ZF(ITRCK1,ITRCK2,PVTX,ITK1,ITK2,
     +                          RTK1,RTK2,VTX,DZ,P1,P2,IERR)

  INPUT    : ITRCK1,ITRCK2 - DST track numbers of the two candidates
  INPUT    : PVTX(3)       - Real primary vertex position
  OUTPUT   : ITK1()        - Integer DST array for track 1
  OUTPUT   : ITK2()        - Integer DST array for track 2
  OUTPUT   : RTK1()        - Real DST array for track 1
  OUTPUT   : RTK2()        - Real DST array for track 2
  OUTPUT   : VTX(3)        - Secondary vertex
  OUTPUT   : DZ            - Error on z
  OUTPUT   : P1(3),P2(3)   - Momenta of tracks at secondary vertex
  OUTPUT   : IERR          - Integer error code (0=Ok)
```

4.9 Improvement of dE/dx measurements.

A subroutine is provided by M. Hauschild to reduce the dE/dx systematic errors and to improve the dE/dx resolution. The corrections are optimised for 1990 PASS4 and 1991 PASS3 data but might be used also for 1990 PASS3 data (or previous passes) with less performance.

In order to have a consistent set of both data and Monte Carlo the correction routine has to be applied also for Monte Carlo (PASS3 as well as PASS4).

Calling sequence (to be called within USBAT):

```
      SUBROUTINE ODDXBT(ICTRK,IMODE,IWT,IERR)

 Input arguments:

      ICTRK  =  0  update ALL  charged tracks
             >  0  update ONLY charged track no. ICTRK
      IMODE  =  1 update dE/dx and error only
```

```
            =  2  update weights only
            =  3  update dE/dx, error and weights
            <  0  don't report updates of CTRK blocks
                    abs(IMODE) gives mode option
      IWT   =  1  FAST option for weights (1-dim, less accurate, fast)
            =  2  FULL option for weights (2-dim, standard DST, slow)
            =  3  automatic selection of FULL/FAST option
                    according to momentum and dE/dx
                    see routine ODDXWT for more information
            <  0  calculate signed weights
                    abs(IWT) gives weight option

Output arguments:

      IERR  =  0  everything o.k.
            >  0  some CTRK blocks not updated (dE/dx or error = 0)
                    IERR = number of not updated CTRK blocks

- e.g. CALL ODDXBT(0,-3,2,IERR)

      - corrects dE/dx and errors of ALL charged tracks
      - calculates dE/dx weights using dE/dx errors and momentum errors
      - updates the DST words
      - does not print any report messages if a DST word has changed

- e.g. CALL ODDXBT(22,3,-3,IERR)

      - corrects dE/dx and error of track no. 22
      - does automatically select the method for weight calculation:
          if dE/dx >= 12 keV/cm use dE/dx error and momentum error (2-dim)
          if dE/dx <  12 keV/cm use dE/dx error only (1-dim)
      - calculates SIGNED weights where:
          SIGN(weight) = SIGN(dE/dx(meas) - dE/dx(expected))
      - updates the DST words
      - prints a report message if a DST word has changed
```

The option of SIGNED weights gives the opportunity to calculate a normalised dE/dx-distribution (dE/dx-norm) as used in the electron-identification:

```
    DXNORM = SQRT(CHISIN(1.-ABS(WEIGHT),1))     ! CHISIN = CERNLIB function
```

```
DXNORM = SIGN(DXNORM,WEIGHT)
```

Hints and notes:

- The 2-dim weight calculation is CPU intensive (5 x just DST reading). Use 1-dim weights or automatic selection (0.5 x just DST reading), if possible.
- The routine checks experiment + run number, ROPE version and date of calibration data base used in order to find out data type and ROPE pass to set-up the right corrections and correction parameters. If you use any 'private' reROPEd data/MC sets please watch report messages from `ODDXDT` if the data type is identified correctly.
- In order not to apply the corrections twice or after a future reROPE where it might be obsolete the information about the performed dE/dx corrections is stored on DST in word `JCDXBT` as a bit pattern (see routine `ODDXCR` for details). After a future reROPE also the record number of the DX data base record used for ROPEing will be stored in `JCDXBT` in order to have full information available for possible future corrections.

For users more familiar with dE/dx here is a list of systematic effects presently corrected on DST level:

- Integer rounding error at truncated mean (DATA + MC)
- Shift in truncted mean for less number of hits (DATA + MC)
- Logarithmic path length correction in Θ (DATA only)
- Logarithmic path length correction in phi (bad approximation, DATA only)
- Asymmetry in $\cos(\Theta)$, quadrant based (DATA only)
- Revised saturation correction (DATA only)
- Dependence of dE/dx on the number of dE/dx hits (DATA only)
- New dE/dx parametrisation for weight calculation (DATA + MC)

- Monte Carlo dE/dx rescaled with new dE/dx parametrisation using TREE information (consistent mean dE/dx in DATA and MC)

Summary of performance features:

- SIGNED weights possible (easy to get dE/dx-norm)
- Improved resolution (now 3.5% for 159 dE/dx hits in GPMH, was 3.8%)
- Less discrepancy of dE/dx of electrons in multi hadrons compared to isolated electrons (< 0.1 keV/cm)
- Less dependence of dE/dx on the number of hits
- No asymmetry in Θ
- Improved saturation correction for higher ionising tracks
- Less dE/dx systematics in end caps (but not yet perfect) due to logarithmic path length correction (cut on $\geq$ 20 dE/dx hits is recommended to reduce systematics further)

4.10 Improvement of TOF measurements.

A subroutine is provided by Mr. Graham Marquis de Wilson to calculate the "best" TOF estimate incorporating a correction for the time vs z systematic and proper weighting of left and right time estimates. The user is expected to look at the distribution of TOFB - TOFGEO to see if the events are consistent with photons. This code was developed for the single photon analysis and uses parametrisations of the timing response and resolution which have been tuned on single electron events.

The corrections are also useful for improving the TOF resolution for charged tracks but the resolution function and response function given are only strictly applicable to electrons. A detailed comparison of the response and resolution for different particle types and environments is in progress and something which is more general purpose will be provided in the near future. The code here should only be used on re-ROPED data (i.e. PASS3)

Calling sequence:

```
      SUBROUTINE ODBTOF(ICASE,ITFLAG,THETA,PHI,TOF,DTHT,
     +                        TOFB,TOFGEO,TOFERR,ZDIF,IRCODE)

 Input arguments -

      ICASE  - Chooses which time vs z correction, and resolution
               function to use
             = 1  for charged tracks  (particle id applications)
             = 2  for single electrons (also useful for photons)
                  (i.e. e-m showers)
             = 3  for all TOF hits in multi-hadron events
                  (mix of tracks and showers) - useful for
                  high statistics calibration
      ITFLAG - TOF DST flag     JTFLAG
      THETA  - TOF DST quantity JTTHET
      PHI    -                  JTPHI
      TOF    -                  JTTOF
      DTHT   -                  JTDTHT

 Output arguments - TOFB   - Best estimate of the time-of-flight
                             (ie. time of arrival of the particle at
                              the TOF scintillator)
                    TOFGEO - Expected time of arrival for a photon.
                             (ie. beta=1, linear trajectory)
                    TOFERR - Calculated error on TOFB - use with a
                             10-20\% pinch of salt.
                    ZDIF   - ZTOFP - ZEB - the residual difference
                             in the TOF z estimate and that from EB.
                             If this is not consistent with zero within
                             the resolution of about 7.5 cm, then
                             it is unlikely that TOFB is reliable.

 Output arguments are in ns,cm.
```

4.11 Secondary Vertices.

The results of the vertex finding algorithms are stored in the VRTX and ATRK blocks. In addition, in the CTRK block, words can be found which

point to the vertex from which the track comes. Most useful is the mask JCVPAT, in which one bit is set for each vertex. The lowest order bit corresponds to the primary vertex, the remaining bits to secondary vertices. One track can belong to more than one vertex. For example, a track which was fitted to the primary vertex and also to secondary vertex number 3, will have the word JCVPAT set to 9. The word JCIVT1 is the vertex number for the beginning of the track. It is useful when a track belongs to one vertex only.

4.12 Storage of raw ECAL data on the DST.

For some analyses it is desirable to have the **raw** energies in each ECAL block. This is especially true for electron id. The DD banks for EB and EE contain the raw ADC values for each block. To transform the ADC values into energies requires running the individual processor and hence a fair amount of CPU time. In order to address this problem the block energy banks (EBNG and EENG) are stored in a compressed form with the DST from OD version 4.00 onwards.

Two routines steer the packing and unpacking of the block energy banks. The routine ODPKNG steers the packing and ODUPNG steers the unpacking of the blocks. The actual packing algorithm is very simple. For each block above a threshold (20 MeV for EB and 40 MeV for EE) the block energy and block number are stored in one integer word. The packing thresholds are settable with the ODBKTH FFREAD card.

- ODBKTH 1=50 Will set the EB packing threshold to 50 MeV
- ODBKTH 2=20 Will set the EE packing threshold to 20 MeV

Currently the packed blocks are stored at link -2 of the GNRL bank, in the ODCE bank. The ODCE bank contains a word denoting the packing version and then two words containing the number of packed EB and EE blocks. Following these words are the packed block words. The blocks are packed and written out if ECAL processing has been selected in ODBAT.

To unpack the blocks the user must call ODUPNG from within USBAT. Tests are be made to see if the ODCE bank exists and whether or not the EBNG and EENG banks exist. ODUPNG will not overwrite existing energy banks. EBNG and EENG banks created by ODUPNG will contain the block

codes and energies for those blocks that were above the packing threshold. For the EENG bank, the energy error word is set to zero for every block.

4.13 ECAL Block Cheat Information.

Marcello Mannelli has provided routines to store Monte Carlo cheat information for each ECAL cluster in the DST. This information is stored in the ODEM bank which hangs from the fifth down link of the MC bank. The first part of the bank contains one word for each ECAL cluster. The first eight bits of each of these words contains the ECAL cluster number (as a cross-check), the second eight bits contains the number of KINE particles contributing to the cluster, and the final sixteen bits contains an offset within the ODEM bank to where the list of energies contributed by each particle are stored. (If this offset is IOFF, the first word of the list of energies is IQ(LODEM+IOFF+1)).

The list of energies contributed by each particle are stored in the second part of the bank. The first sixteen bits of each word contains the KINE particle number, and the final sixteen bits of each word contains the fraction, multiplied by 1000, of *raw* cluster energy contributed by that particle.

Note that this information is incomplete in Monte Carlo generated with OD versions 407 and earlier. The cheat energies may not reflect the full amount attributable to a given KINE track, and not all KINE tracks which contributed energy to a cluster may be listed. This was fixed in OD408. Note also that it is the KINE particle number which is stored in the ODEM bank and not the TREE particle number. The KINE particle number of each TREE particle is stored in word 7 (JTKINE in sequence ODTPAR) of its TREE record. The KINE number will be zero if the particle is not long-lived.

There are no generic access routines for the ODEM bank available in the OD library. An example access routine appears below. This same routine is available for downloading from World Wide Web.

```
      SUBROUTINE USODEM(ITREE,IECAL,ENERGY,IERR)
*.
*...USODEM   Return energy contributed by tree particle ITREE to
*.           cluster IECAL. If ITREE is 0, return the sum of energy
*.           contributed by all tree particles to that cluster.
```

```
+SEQ,ODPAR.
+SEQ,ODTPAR.
+SEQ,RBITFUNC.
+SEQ,RCREP.
      INTEGER ITREE, IECAL, IERR
      INTEGER LODEM, LLODEM
      PARAMETER (LLODEM=5)
      INTEGER KCTR, ICLUS, ICTR, NCTR, IOFF, I
      REAL    ENERGY
      LOGICAL FOUND
+SEQ,ODSFUN.
      IERR = -1
      ENERGY = 0.
*-
*-- Assume user has already called ODFUNS
*     CALL ODFUNS
*
      IF (ITREE.LT.0 .OR. ITREE.GT.NTREE) GOTO 999
      IF (IECAL.LE.0 .OR. IECAL.GT.NECAL) GOTO 999
      IF (LMC.LE.0) GOTO 999
      IF (IQNL(LMC).LT.LLODEM) GOTO 999
      LODEM = LQ(LMC-LLODEM)
      IF (LODEM.LE.0) GOTO 999
*-
      IERR = 0
      IF (ITREE.GT.0) THEN
        KCTR  = ITMC(JTKINE,ITREE)
        IF (KCTR.LE.0) THEN
          IERR = 2
          GOTO 999
        ENDIF
      ELSE
        KCTR = 0
      ENDIF
*-
      ICLUS = IBITS(IQ(LODEM+IECAL),0,8)
      IF (ICLUS.NE.IECAL) THEN
        WRITE (CHREP,1000) IECAL, ICLUS
        CALL REPORT('MYODEM',1,'W')
        IERR = -1
```

```
        GOTO 999
      ENDIF
*-
      NCTR = IBITS(IQ(LODEM+IECAL),8,8)
      IOFF = IBITS(IQ(LODEM+IECAL),16,16)
      FOUND = .FALSE.
      DO 100 I = 1, NCTR
        ICTR = IBITS(IQ(LODEM+IOFF+I),0,8)
        IF (KCTR.GT.0 .AND. ICTR.NE.KCTR) GOTO 100
        FOUND = .TRUE.
        ENERGY = ENERGY + FLOAT(IBITS(IQ(LODEM+IOFF+I),16,16))/1000.
  100 CONTINUE
*-
      IF (.NOT.FOUND) THEN
        IERR = 1
      ELSE
        ENERGY = RE(JEERAW,IECAL)*ENERGY
      ENDIF
*-
  999 RETURN
 1000 FORMAT('Inconsistent ECAL clus no ',I3,' in ODEM bank at posn ',
     + I3)
      END
```

4.14 PE Data Analysis Utilities

These tools have been provided by Yoram Rozen for those using Presampler Endcap clusters in their analyses. The first returns corrected coordinates for PE PRES clusters in data, and the second randomly zeroes the word JPMULT in PE PRES clusters in Monte Carlo DSTs to better model PE sectors missing in data for part or all of the same year.

```
      CALL ODPEAL(ICLUS,THETNW,PHINEW,IERR)

   function         Apply alignment corrections for PE clusters on DST

   input  ICLUS     Presampler cluster number
```

```
output THETNW    New cluster theta position
       PHINEW    New cluster phi position
       IERR      0  Problems with data, use with care
                 1  OK, correction made
                 2  Not a PE cluster, no correction made
                 3  MC Data, no correction made
                 4  N/A, no correction made
```

In cases where IERR is 1-4, the values of THETNW and PHINEW can be used. Note that a return code of zero is an error.

```
   CALL ODPEON(PEON,IERR)

function         Correct MC for sectors unused in data

input  PEON      Integer = 0 Do not use the routine
                           1 Make corrections

output IERR      0  For MC, all is OK; otherwise, no modifications made
                 1  OK, but routine called more than once per event
```

This is currently only available in OD versions 4.08 and up.

4.15 Global Smearing of Track Parameters

Chris Darling has modified the routine BTSMGL so that constraints to the z of the primary vertex are removed before the smearing is applied. The constraints are reapplied when possible after the track parameters are smeared. The smearing is in the form of constant global factors which multiply the differences between the reconstructed and true track parameters, the latter being derived from information stored in the TREE bank. By default these parameters are 1, i.e. no smearing is done. The parameters can be set with the ODSMGLP FFREAD card (see chapter **??**). The smeared track parameters are copied back into the CTRK block along with updated values for the momentum components and charge. No random numbers are used. The calling sequence is

```
CALL ODSMGL
```

Note that ODSMGL uses routines in the BT library, and therefore the BT library must be included in one's link list.

4.16 Known Shortcomings of the OD Processor.

Only the shortcomings of the OD processor itself are described. Shortcomings resulting from detector code, *e.g.* not filling certain data words, are not mentioned.

- Data compression is not yet provided. (This is now available with the DC processor.)
- In merging the muon segments with ECAL and HCAL clusters, the direction information of the muon segments is not used.
- Noise hits in the presampler and electromagnetic calorimeter are stored indiscriminantly.
- Some quantities are not implemented. (see DATA definitions).
- The concept of a flag to indicate whether a charged track extrapolates to the fiducial volume of an outer detector is still rudimentary.
- Users who selectively refill sets of DST blocks (e.g. PRES, ECAL, etc.) and who switch between SI and non-SI tracks may occasionally find inconsistencies in the corresponding block counts in the GNRL block. This is because only the active copy of the GNRL block is updated when a set of blocks is refilled. The user is responsible for updating the other copy if he or she needs these words. The block counters in sequence ODSFUN are correct as long as ODFUNS has been called after the blocks are refilled.
- Rerunning the CX processor with DST tracks as input and refilling the VRTX and ATRK blocks with a call to ODVRTX will result in the loss of the kinked track information in the ATRK blocks. Calling ODTOSI with the CX option will have this effect, although the kinks will still be available for the non-SI configuration in that case.

- A full set of track quantities is not available for the two parts of kinked tracks stored in ATRK blocks (for example, there is no dE/dx). It is foreseen to treat the pieces as separate tracks and store them in CTRK blocks or in an equivalent structure once more experience is gained.
- Calling ODTOSI does not refill USER common blocks with the SI track parameters and primary vertex. One has to call ODFILL after the call to ODTOSI to update the commons. This applies even when ROPE calls ODTOSI.

Chapter 5

DATA definitions

In the OD package, the data are passed in blocks. Each BANK consists of an INTEGER BLOCK followed by a REAL BLOCK of data. The first word of each BANK is the number of INTEGER words in the BLOCK, according to the format '*I-F'.

In the block description, a mnemonic is given for each data word. This mnemonic is defined in the KEEP sequence ODPAR. For usage, see EXAMPLES.

5.1 GNRL

```
Integer:
JGDTYP   Data type
JGDSTY   Data sub type  (obsolete)
JGIEXP   Experiment number
JGIRUN   Run number
JGIFIL   LEP fill number
JGIEVT   Event number
JGPVER   Program version
JGITAP   Tape number (Maxi-DST)
JGIBUN   Bunch number
JGDETS   Detector status mask
JGTRGS   Trigger status mask
JGEVTY   Event type
JGFITY   Filter mask
```

```
JGDATE   Date
JGTIME   Time
JGNCTR   Number of central tracks
JGNPRE   Number of presampler clusters
JGNECA   Number of e.m. clusters
JGNTOF   Number of TOF hits
JGNFDE   Number of forward detector segments.
JGNHCA   Number of HCAL clusters
JGNMUO   Number of MUON segmenst
JGNVTX   Number of vertices
JGNATR   Number of alternative tracks
JGTRM1   Trigger mask word 1
JGTRM2   Trigger mask word 2
JGTRM3   Trigger mask word 3
JGTRM4   Trigger mask word 4
JGNFCL   Number of forward detector e.m. clusters
JGNTBR   Number of recontsructed TOF bars
JGNTBC   Number of coplanar TOF combinations
JGNJET   Number of reconstructed jets.  Not implemented.
JGTPT1   Trigger theta-phi input mask, Track trigger, theta bin 1
JGTPT2   Trigger theta-phi input mask, Track trigger, theta bin 2
JGTPT3   Trigger theta-phi input mask, Track trigger, theta bin 3
JGTPT4   Trigger theta-phi input mask, Track trigger, theta bin 4
JGTPT5   Trigger theta-phi input mask, Track trigger, theta bin 5
JGTPT6   Trigger theta-phi input mask, Track trigger, theta bin 6
JGTPOB   Trigger theta-phi input mask, t-O-f, barrel bin (theta 2 to 5)
JGTPE1   Trigger theta-phi input mask, Ecal trigger,  theta bin 1
JGTPE2   Trigger theta-phi input mask, Ecal trigger,  theta bin 2
JGTPE3   Trigger theta-phi input mask, Ecal trigger,  theta bin 3
JGTPE4   Trigger theta-phi input mask, Ecal trigger,  theta bin 4
JGTPE5   Trigger theta-phi input mask, Ecal trigger,  theta bin 5
JGTPE6   Trigger theta-phi input mask, Ecal trigger,  theta bin 6
JGTPH1   Trigger theta-phi input mask, Hcal trigger,  theta bin 1
JGTPH2   Trigger theta-phi input mask, Hcal trigger,  theta bin 2
JGTPH3   Trigger theta-phi input mask, Hcal trigger,  theta bin 3
JGTPH4   Trigger theta-phi input mask, Hcal trigger,  theta bin 4
JGTPH5   Trigger theta-phi input mask, Hcal trigger,  theta bin 5
JGTPH6   Trigger theta-phi input mask, Hcal trigger,  theta bin 6
JGTPM1   Trigger theta-phi input mask, ME   trigger,  theta bin 1
JGTPM2   Trigger theta-phi input mask, ME   trigger,  theta bin 2
```

```
JGTPMB   Trigger theta-phi input mask, Muon Barrel trigger (theta 2 to 5)
JGTPM5   Trigger theta-phi input mask, ME   trigger,  theta bin 5
JGTPM6   Trigger theta-phi input mask, ME   trigger,  theta bin 6
JGNCT    Number of good charged tracks
JGNEB    Number of good EB clusters
JGNEE    Number of good EE clusters
JGNHBT   Number of good HB tower clusters
JGNHET   Number of good HE tower clusters
JGNHPT   Number of good HP tower clusters
JGTRES   Trigger result mask word
JGDAQS   Data acquisition status mask
JGFIDD   Filter dense data status mask
JGFIST   Filter status mask
JGROST   ROPE status mask
JGEVT2   Second word for event type
JGLEPQ   Lep beam pos quality (0=unk,1=bad,2=soso,3=ok)
JGCDP1   Packed CD hit information
JGCDP2   Packed CD hit information
JGCDP3   Packed CD hit information
JGCDP4   Packed CD hit information
JGCDP5   Packed CD hit information
JGCDP6   Packed CD hit information
JGCDP7   Packed CD hit information
JGPTR1   Pretrigger bits 1-32
JGPTR2   Pretrigger bits 33-64
JGPTP1   Pretrigger Phi; bits 1-12:CV 17-28 CJ
JGPTP2   Pretrigger Phi; bits 1-12:TB 17-28 EB
JGPTP3   Pretrigger Phi; bits 1-12:EEL 17-28 EER
JGPTP4   Pretrigger Phi; bits 1-12:MEL 17-28 MER
JGNSIW   Number of SIW blocks
JGTETR   TE trigger word (since OD413)
JGTENC   Number of TE clusters (since OD413)
JGTPD1   Theta-phi TE word1
JGTPD2   Theta-phi TE word2
JGTRPU   Trigger Pattern Unit ?

Real:
JGBZ     z-component of B field (signed)         [Gauss]
JGEBEA   Beam energy                               [GeV]
JGDEBE   Error on beam energy                      [GeV]
```

JGEVIS	Visible energy	[GeV]
JGPOL1	Polarization of electrons (n.i.)	
JGPOL2	Polarization of positrons (n.i.)	
JGVX	x-coordinate of average vertex	[cm]
JGVY	y-coordinate of average vertex	[cm]
JGVZ	z-coordinate of average vertex	[cm]
JGVE11	<xx>	
JGVE21	<yx>	
JGVE22	<yy>	
JGVE31	<zx>	
JGVE32	<zy>	
JGVE33	<zz>	
JGVWXY	Weight (probability) of x-y fit	
JGVW3D	Weight (probability) of 3-D fit	
JGSPHE	Sphericity of the event	
JGSPCT	Sum of charged momenta	[GeV]
JGSECA	Sum of electromagnetic energy	[GeV]
JGSHCA	Sum of hadronic energy	[GeV]
JGAPLA	Aplanarity	
JGCSPH	Cosine of the sphericity axis	
JGTHRU	Thrust	
JGOBLA	Oblateness	
JGCTHR	Cosine of the thrust axis	
JGT0AV	Average t0 over all TOF hits	[ns]
JGDT0A	Average t0 diff. between copl. TOF hits	[ns]
JGDT0M	Minimum t0 diff. between copl. TOF hits	[ns]
JGPSPH	Phi angle of sphericity axis	[rad]
JGPTHR	Phi angle of thrust axis	[rad]
JGMEPX	x-component of missing energy vector	[GeV]
JGMEPY	y-component of missing energy vector	[GeV]
JGMEPZ	z-component of missing energy vector	[GeV]
JGBX	Beam crossing time	[ns]
JGBXCJ	Beam crossing time for CJ	[TDC counts]
JGBXCV	Beam crossing time for CV	[TDC counts]
JGDECV	CV gas density (p/t)	[?]
JGSCT	Scalar sum of good charged track momenta	[GeV]
JGSEB	Sum of good EB cluster energy	[GeV]
JGSEE	Sum of good EE cluster energy	[GeV]
JGSHBS	Sum of HB strip cluster energy	[GeV]
JGSHES	Sum of HE strip cluster energy	[GeV]

```
JGSHPS   Sum of HP strip cluster energy            [GeV]
JGSHBT   Sum of good HB tower cluster energy       [GeV]
JGSHET   Sum of good HE tower cluster energy       [GeV]
JGSHPT   Sum of good HP tower cluster energy       [GeV]
JGSFE    Sum of FD gamma catchers cluster energy   [GeV]
JGSFK    Sum of FD calorimeters cluster energy     [GeV]
JGLEPX   X position of vertex from LEP              [cm]
JGLEPY   Y position of vertex from LEP              [cm]
JGBXDI   BX direct signal seen in TOF              [TDC]
JGSTCN   Time offset for TOF calc                   [ns]
JGOPMX   Maximum opening angle of CV tracks        [rad]
JGBTZ    Time offset for bunchlet                   [ns]
```

The vertex parameters JGVX, JGVY and JGVZ are calculated by CX for each event. If no primary vertex was found, for example because there are not enough tracks, a search is done in bank GNCT for the default values for the vertex. The error matrix then becomes diagonal.

```
JGDETS, JGTRGS
   Bitmasks with the detector status and the trigger status.
   Two bits per detector, allowing status values 0-3. (3=all Ok)
   CV 1  CJ 3  CZ 5  TB 7  PB 9  EB 11 PE 13 EE 15 HT 17 HS 19
   HP 21 MB 23 ME 25 FD 27 SI 29 SW 31

   Example IPBDST= JBYT(IG(JGDETS),9,2)
```

5.2 CTRK

Charged track block.

```
Integer:
JCNHCV   Number of axial CV hits + 100* stereo hits
JCNHCJ   Number of CJ hits
JCNHCZ   Number of CZ hits
JCFTYP   Fit type
JCPTYP   PDG particle type.
JCIMCT   Monte Carlo (TREE) track number
```

```
JCMMSK   HCAL and MUON chamber hit mask
JCIVT1   Vertex number start of track
JCDXBT   dE/dx bit pattern
JCVPAT   Vertex pattern (bit 0 is primary vertex)
JCIPRE   Pointer to presampler cluster
JCIECA   Pointer to ECAL cluster
JCITOF   Pointer to TOF hit
JCIHCA   Pointer to HCAL cluster
JCIMUO   Pointer to MUON segment
JCIJET   Jet number
JCCVHT   Vertex chamber hit pattern, see below.
JCNHDE   Number of hits used for dE/dx in CJ
JCNBEC   Number of EC blocks used in the E flow         (Deleted)
JCMLLH   Log likelyhood for matching CV-CJ-CZ. (see below)
JCSIH1   Hit number silicon barrel 1
JCSIH2   Hit number silicon barrel 2

Real:
JCPX     Best value for px at primary vertex       [GeV]
JCPY     Best value for py at primary vertex       [GeV]
JCPZ     Best value for pz at primary vertex       [GeV]
JCQ      Charge                                      [e]
JCKAPP   Kappa=1/(2rho)                           [1/cm]
JCPHI0   Phi0                                      [rad]
JCD0     d0 = (phi x d).z                           [cm]
JCTLAM   TAN(lambda) = COT(theta) track polar angle
JCZ0     z at p.c.a of track                        [cm]
JCCHIR   Chisq/DGF in r-phi
JCCHI3   Chisq/DGF in S-Z
JCEM11   <kappa0 kappa0>                       [1/cm**2]
JCEM21   <phi0 kappa0>                          [rad/cm]
JCEM22   <phi0 phi0>                            [rad**2]
JCEM31   <d0 kappa0>                             [cm/cm]
JCEM32   <d0 phi0>                              [rad*cm]
JCEM33   <d0 d0>                                 [cm**2]
JCEM41   <tdip kappa0>                           [1./cm]
JCEM42   <tdip phi0>                               [rad]
JCEM43   <tdip d0>                                 [cm]
JCEM44   <tdip tdip>                                 []
JCEM51   <z0 kappa0>                            [cm/cm]
```

JCEM52	<z0 phi0>	[cm*rad]
JCEM53	<z0 d0>	[cm**2]
JCEM54	<z0 tdip>	[cm]
JCEM55	<z0 z0>	[cm**2]
JCR1	Radius of first measured point in CJ	[cm]
JCR2	Radius of last measured point in CJ	[cm]
JCDEDX	dE/dx in CJ	[keV/cm]
JCDDED	error in dE/dx	[keV/cm]
JCWDE	dE/dx weight for electron	
JCWDMU	dE/dx weight for muon	
JCWDPI	dE/dx weight for pion	
JCWDK	dE/dx weight for kaon	
JCWDP	dE/dx weight for proton	
JCTLEN	CT Track length in r-phi	[cm]
JCBETA	TOF beta	[v/c]
JCDBET	TOF error in beta	[v/c]
JCWPRE	Weight for best presampler cluster	
JCWECA	Weight for match with ECAL cluster	
JCWTOF	Weight for match with TOF hit	
JCWHCA	Weight for match with HCAL cluster	
JCWMUO	Weight for match with MUON segment	
JCTHEC	Theta at intersection point with EB EE	[rad]
JCDTHE	Error in Theta at int point with EB EE	[rad]
JCPHEC	Phi at intersection point with EB EE	[rad]
JCDPHE	Error in phi at int point with EB EE	[rad]
JCTHHC	Theta at intersection point with HB HE HP	[rad]
JCDTHH	Error in theta at int point with HB HE HP	[rad]
JCPHHC	Phi at intersection point with HB HE HP	[rad]
JCDPHH	Error in phi at int point with HB HE HP	[rad]
JCTHMU	Theta at intersection point with MB ME	[rad]
JCDTHM	Error in Theta at int point with MB ME	[rad]
JCPHMU	Phi at intersection point with MB ME	[rad]
JCDPHM	Error in phi at int point with MB ME	[rad]
JCP	Total momentum	[GeV]
JCDP	Error in momentum	[GeV]
JCRECA	R(x,y,z) at int point with EB EE	[cm]
JCRHCA	R(x,y,z) at int point with HB HE HP	[cm]
JCRMUO	R(x,y,z) at int point with MB ME	[cm]
JCTHDM	Polar direction of Muon at MB/ME int point	[rad]
JCPHDM	Azimuthal direction of Muon at MB/ME int point	

```
JCVTMU   Mult scat par. at Mu chambers: var(t)
JCVPMU   Mult scat par. at Mu chambers: var(psi)
JCTPMU   Mult scat par. at Mu chambers: cov(t,psi)
JCECON   Energy in 30 degree cone              (Deleted)
JCECO2   Energy flow in cone; shape variable.  (Deleted)
JCEPZ    Z coordinate of endpoint in endcap         [cm]
JCEPDZ   Error in z-coordinate in endcap            [cm]
JCFMOM   First moment EM-CTRK                  (Deleted)
JCDIST   Distance centroids EM-CTRK            (Deleted)
JCRUNT   Run test on CJ hits (avorton)
JCECOR   Corrected ECAL energy                 (Deleted)
JCZOCJ   Z0 value from CJ                           [cm]
JCTLCJ   Tan(lambda) value from CJ
JCQEFF   dE/dx track density (effective charge)    [keV]
JCMWD1   Sum of PB U strip multiplicity about track
JCMWD2   Sum of PB V strip multiplicity about track
JCMWD3   Sum of PB wire multiplicity about track
```

JxSOUR

For (almost) all blocks, a bitmask is defined to indicate which detectors participated. Parameters for all detectors exist in ODPAR:

Name	Hex value	Name	Hex value	Name	Hex Value
MSKCV	1	MSKCJ	2	MSKCZ	4
MSKTB	8	MSKPB	10	MSKPE	20
MSKEB	80	MSKEE	100	MSKHB	200
MSKHE	400	MSKHP	800	MSKMB	2000
MSKME	4000	MSKFD	10000	MSKSW	80000

JCCVHT

bits 1-18	bits 19-22	bits 23-25
bit set for each hit wire in CV	N2HITS in axial	N2HITS in stereo

where N2HITS is a counter for the number of hits which followed one or more preceding hits on the same CV wire. This is an interesting quantity for people doing impact parameter analyses because the intrinsic CV resolution is considerably worse for hits which are not the first to arrive on a given wire in a given event.

JCIECA (TOF) (HCA) (MUO)

This pointer can have three values:

>0 The cluster to which the track points.
=0 The track does not point to a cluster.
<0 The track extrapolation thought that the track did not not make it into the ECA (TOF),... Note that this is not always correct for tracks below 2 GeV!

JCFTYP

The lowest four bits, interpreted as an integer in the range 0-15, define the fit type:

1) Circle fit using OUKARF
2) Billoir fit through all points using OUBILF
3) Billoir fit in CJ, then add discrete scatters, then Billoir in CV
4) same as (3) but systematic errors are added onto CJ pars before CV hits are put on the track. This is the default.

The fifth lowest bit is set if the track is constrained to the vertex in Z.

The sixth lowest bit is set if the track is constrained to the CJ endpoint.

If the CT track fit fails, the fit code is set to -1, in which case the parameters in the CT track bank are taken directly from the CJ fit.

JCMMSK

A word with one bit set for each layer in the hadron calorimeter or muon chambers which has one or more hits.

Bits 1-10 HB or HP hits.
Bits 11-18 HE hits
Bits 19-23 MB hits
Bits 24-31 ME hits

Roads: HB 20 mrad + 100 mrad/p(GeV)

```
HE     20 cm   + 45  cm/p(GeV)
MB     100mrad + 100 mrad/p(GeV)
ME     55 cm   + 55  cm/p(GeV)
```

JCVPAT

A word with a bit set for each vertex the track may originate from. The first bit corresponds to the primary vertex, the remaining bits to secondary vertices.

JCMLLH

Matching likelihood between central detectors:

first ten bits for CV-CJ matching likelihood,
next ten bits for CW-CJ matching likelihood,
last ten bits for CZ-CJ matching likelihood

unpack: likelihood = 10bits/10.0 - 2.3, or use ODCTMA (see above)

JCSIH1, JCSIH2

Packed Silicon hit numbers for barrel 1 and 2

```
unpack: z hit number = MOD( IC(JCSIH1,.)/100000,40)
        ladder number= MOD( IC(JCSIH1,.)/1000  ,100)
        xy hit number= MOD( IC(JCSIH1,.)       ,1000)
```

JCECON

The corrected Ecal energy within a cone of 30 mrad around the track at EB entrance, i.e. the difference between the extrapolated track points and the block center must be less than 30 mrad

In the endcap region the number of blocks used for determining ECONE are those which would be hit by the track, assuming a straight line extrapolation, in a radius of 50cm along this track, counting from the EE entrance point, and 80mrad in phi angle.

JCECO2

The corrected ECAL energy deposit in all blocks contributing to the ECONE variable, plus the energy deposit in all neighbour blocks, i.e. in respect of the ECONE contributing blocks.

JCRUNT
   difference between the observed and expected numbers of "runs"
   in the signs of the track residual distribution divided by the
   expected uncertainty in this difference. For a good track with
   no kinks, we expect it to be distributed according to a
   gaussian with zero mean and unit width. Decays in flight and
   tracking/calibration problems are likely to result in more
   negative values of SIGMA.

JCCHI3
   Up to and including OD313, this word IS actually filled with
   the ZS fit chisquared.

JCDXBT
   Bit pattern filled by the DX processor or
                     by the dE/dx DST correction routines:

   Bits 0-17 bits with performed corrections (see routine ODDXCR)
   Bits 18-22 not used (reserved for future corrections)
   Bit   23 flag for dE/dx DST correction
   Bits 24-31 record number of DX data base record used for ROPEing

JCQEFF
   dE/dx track density (effective charge) [keV]

   For each track the amount of charge deposited by all other tracks
   within the same CJ sectors = effective charge caused by other tracks.

   A large effective charge indicates a large track density/activity in the
   neighbourhood of that specific track (jet core) resulting in increasing
   dE/dx systematics caused by FADC pedestal shifts. In addition the dE/dx
   resolution is degraded.

JCMWD1, JCMWD2, JCMWD3
   Sums of the multiplicities of PB U and V strips and wires in 5 mrad
half-angle cones about the point to which the track is extrapolated at
the barrel presampler. These quantities were added in OD407.

   Note that the multiplicity is the charge accumulated on the set of
strips or wires falling within the wedge corrected for the path length

of particles traversing the chambers which increases with |cos(theta)|.
Nominally the wires run parallel to the Z direction and the U and V
strips lie along the lines:

```
       z + rphi = constant  ( U strips )
       z - rphi = constant  ( V strips )
```

5.3 PRES

```
Integer:
JPSOUR   Detector bit mask
JPNCTR   Number of pointing CTRK tracks.
JPIECA   Pointer to EM cluster

Real:
JPMULT   Observed multiplicity
JPDMUL   Error in observed multiplicity
JPTHET   Theta of cluster                          [rad]
JPDTHE   Error in theta                            [rad]
JPPHI    Phi of cluster                            [rad]
JPDPHI   Error in phi                              [rad]
JPSIZU   Size in U - strips (PB), vertical strips (PE)
JPSIZV   Size in V - strips (PB), horizontal strips (PE)
JPWECA   Weight of match to EM cluster
```

5.4 TOF

```
Integer:
JTFLAG     Flag
Real:
JTMULT     TOF pulse height in M.I. units
JTDMUL     Error in JTMULT
JTTHET     Theta of hit                          [rad]
JTDTHE     Error in theta                        [rad]
JTPHI      Phi of hit (0-2pi)                    [rad]
JTTOF      Measured time                          [ns]
```

```
JTDTOF     Error in measured time                  [ns]
JTDTHT     Theta difference used to calculate TOF[rad]
```

5.5 ECAL

```
Integer:
JESOUR   Source (detector bit mask)
JEITOF   Pointer to TOF hit
JENBLO   Number of blocks in cluster
JENB90   Number of blocks with 90% of energy
JENCTR   Number of matching CTRK tracks
JENPRE   Number of matching presampler clusters
JENMUO   Number of matching muon hits
JEIHCA   Pointer to best HCAL cluster

Real:
JEE      Energy corrected for angles              [GeV]
JEDE     Error in corrected energy                [GeV]
JEERAW   Raw Energy                               [GeV]
JEDERA   Error in raw energy.                     [GeV]
JETHET   Theta of the cluster                     [rad]
JEDTHE   Error in theta                           [rad]
JEPHI    Phi of cluster (no angle corr.)          [rad]
JEDPHI   Error in phi                             [rad]
JETHE1   Min theta                                [rad]
JETHE2   Max theta                                [rad]
JEPHI1   Min phi                                  [rad]
JEPHI2   Max phi                                  [rad]
JEFMOM   First moment of cluster                  [rad]
JECDST   Distance of most energetic block from cl...      (deleted)
JEFRMX   Fraction of (raw) energy in most energetic block
JEWHCA   Weight for best HCAL cluster
JEFSMX   Fraction of raw energy in second most en...
JEDFSB   Distance between most and 2nd en. blo...         (deleted)
JEINVM   Invariant cluster mass                   [GeV]

JEINVM
    The invariant cluster mass, i.e. calculationg the 4-vector for the
```

cluster (sum of all blocks) and then calculating the invariant mass
by sqrt(ee*2-p**2)

5.6 HCAL

Integer:

JHSOUR	Detector bit mask
JHNSTR	Number of strips in cluster
JHNTOW	Number of towers in cluster
JHTWBT	Mask of operational layers
JHNCTR	Number of pointing CTRK tracks
JHNECA	Number of pointing EM clusters
JHNMUO	Number of pointing MUON segments
JHNL14	No of hits in layers 1-4 (bytepacked)
JHNL58	No of hits in layers 5-8
JHNL90	No of hits in layer 9

Real:

JHE	Energy (corrected for gain and dead layers)	[GeV]
JHDE	Error in corrected energy	[GeV]
JHTHET	Theta of the cluster	[rad]
JHDTHE	Error in theta	[rad]
JHPHI	Phi of cluster	[rad]
JHDPHI	Error in phi	[rad]
JHTHE1	Min theta	[rad]
JHTHE2	Max theta	[rad]
JHPHI1	Min phi	[rad]
JHPHI2	Max phi	[rad]
JHFIN1	Fraction of raw energy in layer 1.	
JHET	Tower energy	[GeV]
JHES	Strip equivalent energy	[GeV]
JHEUNC	Uncorrected energy	[GeV]
JHDYSC	Halfwidth of endcap strip cluster in y	[cm]
JHDXSC	Halfwidth of endcap strip cluster in x	[cm]

JHNL14

A four-byte word, where the first byte contains the number of hits in layer 1, the second byte the number of hits in layer 2, etc. JHNL58 and JHNL90 are the same for layers 5 to 8 and 9 to 10. The first byte is the lowest order byte.

JHDYSC, JHDXSC
: The y and x extents of the cluster. These values are meaningful on for endcap strip clusters. In the endcap, the strip are parallel to the x-axis. In HB and HP, they are at constant phi.

5.7 MUON

Integer:

JMSOUR	Detector bit mask
JMNCTR	Number of pointing CTRK tracks
JMNPRE	Number of pointing presampler clusters (not used)
JMIMCT	Monte Carlo track number (not for purely HCAL muons)
JMCASE	Case (not for purely HCAL muons) *
JMIECA	Pointer to EM cluster
JMITOF	Pointer to TOF hit (not used)
JMIHCA	Pointer to HCAL cluster
JMKODE	Hit code *
JMNMBH	Number of barrel hits
JMNEXZ	Number of x-z hits in endcap
JMNEYZ	Number of y-z hits in endcap
JMNHST	Number of hits in HCAL
JMNSEX	Number of endcap x muon segments
JMNSEY	Number of endcap y muon segments
JMNDOF	Degrees of freedom for muon barrel fit
JMHNBY	Number of nearby hits in HB segment
JMFITZ	Flag for segments which fail z fit

Real:

JMX	x Coordinate of MUON segment	[cm]
JMY	y Coordinate	[cm]
JMZ	z Coordinate *	[cm]
JMDX	Error in x	[cm]

JMDY	Error in y	[cm]
JMDZ	Error in z *	[cm]
JMCHMB	Chisq of the muon barrel fit	
JMCIXZ	Chisq of the muon endcap x-z fit	
JMCIYZ	Chisq of the muon endcap y-z fit	
JMPHI	Phi of segment	[rad]
JMDPHI	Variance in phi.	[rad]
JMMTHE	Theta of segment *	[rad]
JMDCOT	Variance in cot(theta) *	[]
JMCOXY	Correlation in muon barrel x-y fit	[cm**2]
JMWECA	Weight of match to EM cluster	
JMWTOF	Weight of match to TOF hit	
JMWHCA	Weight of match to HCAL cluster	
JMCHHB	Chisquare of HB muon segment	
JMJUNK	Junk	

JMCASE A word of up to 4 digits (one per hit), each digit coded as follows.

1 Hit from a prompt particle.
2 Hit from a hadronic interaction product.
3 Hit from the decay of a prompt particle.
4 Hit from an electron/positron (delta-ray etc.).
5 Failure.

e.g. 1133 primary has decayed in the muon chambers and the segment was produced with hits from the prompt particle and its decay product.

JMKODE A word of 5 digits (one per detector) coded as follows.

1st digit	Muon barrel
2nd digit	Muon endcap
3rd digit	Hadron barrel
4th digit	Hadron poletip
5th digit	Hadron endcap

with the codes

0	no detector information

```
         1           detector hits on the segment
         2           detector track on the segment

         e.g.  00212 muon barrel and hadron barrel tracks with
               muon endcap hits. No hadron poletip information.

         JMCASE and JMKODE are decimal codes!

JMZ      Can be zero when the muon barrel fails to find a z-fit
JMDZ                          ''
JMMTHE                        ''
JMDCOT                        ''
```

5.8 FDET

```
Integer:
JFSOUR     Detector bit mask:
JFDTI1     Packed drift time information  (see below)
JFDTI2     Packed drift time information
JFPAD1     Packed cathode pad information (see below)
JFPAD2     Packed cathode pad information

Real:
JFE        Energy                                  [GeV]
JFDE       Error in energy                         [GeV]
JFTHET     Theta                                   [rad]
JFDTHE     Error in theta                          [rad]
JFPHI      Phi                                     [rad]
JFDPHI     Error in phi                            [rad]
JFEPRE     Presampler energy                       [GeV]

  JFSOUR     Detector bit mask:
         1st bit     Gamma catcher (FE) hit (see release notes)
         2nd bit     Fine luminosity monitor (FL) hit
         3rd bit     Drift chambers (FT) hit
         4th bit     Tube chambers (FB) hit
         5th bit     Calorimeter (FK) hit
         6th bit     Far-forward monitor (FF) hit (not yet)
```

```
        7th bit     FB horizontal plane hit
        8th bit     FB vertical plane hit
        9th bit     FB diagonal plane hit
       10th bit     FL A-counter hit (only for FL-lumi events)
       11th bit     FL C-counter hit (only for FL-lumi events)
   12- 13th bits    number of FT hits in front plane
   14- 15th bits    number of FT hits in rear plane
       16th bit     lumi tag, cluster used forming MAINBAR
       17th bit     lumi tag, cluster used forming TOTBAR
       18th bit     lumi tag, cluster used forming CUBED
       19th bit     lumi tag, cluster used forming FLLUMI
       20th bit     cluster found by silicon-tungsten
       21st bit     silicon-tungsten trigger status
   22- 24th bits    particle identification code
                       1 = electromagnetic shower
                       2 = track
                       3 = hadronic shower
```

JFDTI1

```
  Bits  1-10   Drift time from reference chamber (SARC) wire 1.
  Bits 11-20   Drift time from reference chamber (SARC) wire 2.
  Bits 21-30   Drift time from tracking chamber (TC2) wire 1.
```

JFDTI2

```
  Bits  1-10   Drift time from tracking chamber (TC2) wire 2.
  Bits 11-20   Drift time from tracking chamber (TC2) wire 3.
  Bits 21-30   Drift time from tracking chamber (TC2) wire 4.
```

JFPAD1

```
  Bits  1-10   Signal from reference chamber (SARC0 pad 1.
  Bits 11-20   Signal from reference chamber (SARC) pad 2.
  Bits 21-30   Signal from tracking chamber (TC2) pad 1.
```

JFPAD2

```
  Bits  1-10   Signal from tracking chamber (TC2) pad 2.
```

```
Bits 11-20   Signal from tracking chamber (TC2) pad 3.
Bits 21-30   Signal from tracking chamber (TC2) pad 4.


Packed drift times. Each time occupies 10 bits, and is stored
in units of 3ns.

e.g. to get the time for SARC wire 1,
      ITPACK = IF(JFDTI1,IBLK)
      CALL MVBITS(ITPACK,0,10,ITIME1,0)
      TIME1 = 3.*FLOAT(ITIME1)

Packed diamond pad signals. Each signal occupies 10 bits; the
first 9 are the absolute value in units of 1/500, while the
most significant bit gives the sign; 0 => +ve, 1=> -ve.

e.g. to get the pad signal for TC2 wire 4,
      IPPACK = IF(JFPAD2,IBLK)
      CALL MVBITS(IPPACK,20,9,IPAD4,0)
      CALL MVBITS(IPPACK,30,1,ISIGN,0)
      PAD4 = 0.002*FLOAT(ITIME1)
      IF (ISIGN .EQ. 1) PAD4 = -PAD4
```

For DSTs produced with OD version 406 and before, the SiW clusters and tracks were also stored in the FDET block. For these blocks the quantities are defined as below. (See section **??** for information about the SIW block.)

```
There are several different types of SW clusters that can be found
in the FDET bank:
                                                        Source bits
                                                        -----------
       Cluster type                                     2 2 2 2 2 2
     -----------------                                  0 1 2 3 4 5
                       __
                      /   electromagnetic showers       1 x 1 0 0 0
     data clusters   <    tracks                        1 x 0 1 0 0
                      \__ hadronic showers              1 x 1 1 0 0

     trigger clusters                                   0 x 0 0 0 1
```

where x is the bit indicating whether the trigger status was correct for the event.

In other words, if bit 20 is set, then the cluster has been constructed from the SW data, and all the DST quantities are filled with their appropriate values. Note that the track-finder cannot calculate an energy for tracks, so they have an energy of exactly 0. The values of the DST words are as follows:

```
JFE     = energy of the cluster (0 for tracks)
JFDE    = 0. (not calculated)
JFTHET  = theta of the cluster (pi - theta for the left end)
JFDTHE  = error on theta
JFPHI   = phi of the cluster
JFDPHI  = error on phi
JFDTI1  = row number (0..31) of the maximum in each of the layers
          from layers 4 to 9 as follows:
                   ..999998 88887777 76666655 55544444
JFDTI2  = internal fitted slope of cluster (in microradians)
JFPAD1  = longitudinal shower profile parameters
JFPAD2  = event pedstal (lower 16 bits) and number of pads used
          forming the cluster (high 16 bits)
```

If bit 25 is set, then the cluster was constructed from the trigger information. In this case, only the energy and phi coordinates are available. The quantities that are filled are:

```
JFE     = 0.
JFDE    = energy of the trigger cluster
JFTHET  = 0. for right end, pi for left end
JFPHI   = phi of the trigger cluster
JFDTI1  = cluster type (1=current event, 2=accidental event)
```

This convoluted state of affairs is designed so that the casual user who simply looks at all quantities in the FDET block will not need to check bits. If there is a mixture of data and trigger clusters in the FDET block, then the user will only pick up the real reconstructed showers, since the trigger clusters are marked as having 0 energy. This, of course, also prevents double-counting of

```
the energy since the data clusters usually correspond to the clusters
found from the trigger information.
```

The SW data and trigger cluster DST information may be accessed via ODSW routines described below in table **??**. These routines are compatible with either version of the DST (i.e. with SiW quantities stored in the FDET or SIW blocks).

5.9 VRTX

```
Integer:
JVTYPE   Vertex type
JVALGO   Algorithm
JVNCTR   Number of CTRK track used

Real:
JVCSRP    Cos(r.p) in the x-y plane
JVD0      Abs(d0) w.r.t. event vertex            [cm]
JVFLEN    Flight length w.r.t. event vertex      [cm]
JVSD0     Sum of abs(d0) w.r.t. event vertex     [cm]
JVCSVP    Cosine of opening angle at vertex point
JVDXY     separation in x-y at vertex point      [cm]
JVDZ      Separation in z at vertex point        [cm]
JVPSC2    Pseudo-chisquared for V
JVMEE     Invariant mass assuming e+e-          [GeV]
JVMPP     Invariant mass assuming pi+pi-        [GeV]
JVMPRP    Invariant mass assuming p pi-         [GeV]
JVMPPR    Invariant mass assuming pi+ p-bar     [GeV]
```

5.10 ATRK

```
Integer:
JAIVT1   Vertex number of origin (1=primary)
JAIVT2   Vertex number of end point (zero for kinks)
JAICTR   Original track number (only non-zero for kinks)

Real:
```

```
JAPX    best value for px                              [GeV]
JAPY    best value for py                              [GeV]
JAPZ    best value for pz                              [GeV]
JAQ     charge                                           [e]
JAKAPP  Kappa=1/2rho                                  [1/cm]
JAPHI0  phi0                                           [rad]
JAD0    d0 = phi x d.z                                  [cm]
JATLAM  TAN(lambda) = COT(theta) track polar angle
JAZ0    z at p.c.a of track                             [cm]
JACHIR  Chisq/DGF in r-phi
JACHIZ  Chisq/DGF in s-z
JAEM11  <kappa0 kappa0>                            [1/cm**2]
JAEM21  <phi0 kappa0>                               [rad/cm]
JAEM22  <phi0 phi0>                                 [rad**2]
JAEM31  <d0 kappa0>                                  [cm/cm]
JAEM32  <d0 phi0>                                   [rad*cm]
JAEM33  <d0 d0>                                      [cm**2]
JAEM44  <tdip tdip>                                        []
JAEM54  <z0 tdip>                                       [cm]
JAEM55  <z0 z0>                                      [cm**2]
JAVX    x of the secondary vertex                       [cm]
JAVY    y of the secondary vertex                       [cm]
JAVZ    z of the secondary vertex                       [cm]
```

Beginning with overall program version 1784 the ATRK blocks also contain the track parameters of the two parts of kinked tracks found by the CX processor using CJ hits. These follow the pseudo-tracks from secondary vertex finding in the ATRK blocks. For vertices the correspondence between VRTX blocks and ATRK blocks is preserved. (This correspondence is somewhat circumstantial; it is best to check the value of word JAIVT2 to get the VRTX block number corresponding to a given ATRK.) For the (two) pieces of a kinked track, word JAIVT2 is zero while word JAICTR contains the number of the kinked CTRK.

5.11 SIW

Beginning with OD407 the SW clusters are stored in the SIW block. More detailed quantities are also stored in the "hidden" ODSL bank; the ODSW

access routines described briefly below in table **??** provide access to quantities stored in the SIW block and ODSL bank.

```
Integer:
JSSOUR  Cluster type bit mask (see below)

Real:
JSE     Cluster energy (zero for tracks)          [GeV]
JSTHET  Cluster theta                             [rad]
JSPHI   Cluster phi                               [rad]
JSR     Cluster radius                             [cm]
JSDR    Uncertainty in the cluster radius          [cm]
JSLSHD  Fraction of energy in first four layers

JSSOUR  Cluster type bit mask:
   Bit  1      0 = right end, 1 = left end
   Bits 2-4    1 = Electromagnetic cluster
               2 = Track
               3 = Hadronic cluster
               4 = Unidentified cluster (i.e. blob)
   Bits 5-11   Corresponding cluster number in ODSL bank
   Bit 20      SW detector mask
   Bit 21      SW trigger status bit
```

Many SW analyses use the SW routines SWGNGC, SWGTGC, SWGNTC, and SWGTTC, to extract data and trigger clusters from the results banks after running the SW processor on dense data. The following OD routines have the same calling sequences and return information in the same format as the corresponding SW routines, but starting with the information on the DST:

`ODSWNG (NCLUS)`	returns the number of data clusters
`ODSWGC (ICLUS, GCLUS)`	puts information from the `ICLUS`th data cluster into array `GCLUS`
`ODSWNT (NCLUS)`	the number of trigger clusters
`ODSWTC (ICLUS, TCLUS)`	puts information from the `ICLUS`th trigger cluster into array `TCLUS`

Here is an example of their use:

```
     +SEQ,SWGCPM.
           INTEGER NCLUS, ICLUS
```

```
      REAL GCLUS(NGCSIZ), E
*
      CALL ODSWNG (NCLUS)
      DO 100 ICLUS = 1,NCLUS
        CALL ODSWGC (ICLUS, GCLUS)
        E = GCLUS(JGCENR)
          (et cetera)
  100 CONTINUE
```

The array GCLUS contains as much of the original cluster information as is possible to reconstruct from the DST information. See the SW manual (SWMANUAL TEX on the WRITEUP disk) or the definition of the SWGCPM sequence in the SW ROPE card file for a description of what information is in a cluster.

5.12 DTE

Since OD413 the TE cluster bank (DTE) is filled.

```
Integer:
JDITH   Theta angle of cluster
JDIPH   Phi angle of cluster
JDICAP  Endcap ID of cluster
JDIADC  Integrated charge of cluster
JDITD0  Time zero of first pulse in cluster
JDIDW   Time width of first pulse in cluster
JDNOTP  Number of time pulses in cluster

Real:
None so far
```

5.13 USER

The user has the possibility of defining his own block which will have the name 'USER'. For this, he has to prepare an integer and a real array, and store these with ODSBLK. The user block is retrieved with ODFBLK.

5.14 Monte Carlo Information.

By default, the Monte Carlo TREE and a rudimentary *cheat* bank for the electromagnetic calorimeters are saved in the DST. (See the GOPAL user's manual). The TREE, JETS and XTRA information can be retrieved as the other DST banks with a call to ODFBLK:

```
CALL ODFBLK('TREE',ITREE,NITREE,ITDAT,NIMOVE,NRTREE,RTDAT,NRMOVE)
```

and similar for JETS and XTRA. The TREE index ITREE is the number as it appears in the TREE bank. It is also the number stored under JCIMCT and JMIMCT in the CTRK and MUON blocks. The total number of TREE, JETS and XTRA entries can be retrieved by a call to ODNBLK. Now the parameters NITREE and NRTREE and the offsets in the data arrays ITDAT and RTDAT are stored in a sequence ODTPAR, in the OD PAM file:

```
TREE integer:
JTPID   Particle Data Group ID.
JTIPAR  Parent Number.
JTISEC  Number of first secondary in tree.
JTNSEC  Number of secondaries.
JTICTR  Track number in DST.     THIS QUANTITY IS NOT SET !!!
JTIJET  Jet Number.
JTKINE  Number of KINE bank.
JT0000  <spare>
JTSFLG  Start flag.
JTEFLG  End flag.

Reals:
JTPX    Px                        [GeV]
JTPY    Py                        [GeV]
JTPZ    Pz                        [GeV]
JTE     Energy.                   [GeV]
JTMASS  Mass.                     [GeV]
JTQ     Charge.                     [e]
JTX0    x start.                   [cm]
JTY0    y start.                   [cm]
JTZ0    z start.                   [cm]
JTXEND  x end.                     [cm]
```

```
JTYEND  y end.                      [cm]
JTZEND  z end.                      [cm]

JETS integers:
JJFLAV  Flavour (PDG code).
JJIPAR  Number of parent particle in JETS.
JJIDEC  Number of parent particle in TREE.

Reals:
JJPX    Px                          [GeV]
JJPY    Py                          [GeV]
JJPZ    Pz                          [GeV]
JJE     Energy.                     [GeV]
JJMASS  Mass                        [GeV]

XTRA reals only:
JXZB    B-quark z value
JXZAB   Anti B-quark z value
JXMTB   B-quark transverse mass      [GeV]
JXMTAB  Anti B-quark transverse mass [GeV]
JXZC    C-quark z value
JXZAC   Anti C-quark z value
JXMTC   C-quark transverse mass      [GeV]
JXMTAC  Anti C-quark transverse mass [GeV]
JXTMHL  Tau minus helicity
JXTPHL  Tau plus helicity
```

Chapter 6

Release Notes

6.1 Release notes version 4.14

1. Theta-phi inputs for TE in GNRL bank (TPD1,TPD2).
2. New global fix-up routine : ODFIX

Chapter 7

Release Notes

7.1 Release notes version 4.13

1. New DTE bank for TE clusters, new words TETR and TENC in GNRL bank.

7.2 Release notes version 4.12

1. A few minor bug fixes.

7.3 Release notes version 4.11

1. New versions of ODCSHI,ODCSPR.
2. New routines ODSIZL,ODPTRK,ODCTOC,ODADMS.
3. New routine ODPSCA to scale MC track momenta.
4. Correct arguments in ZSHUNT calls in ODTOSI.

7.4 Release notes version 4.10

1. Mods to ODDEDX.

2. Corrected ODGLOB default values.

3. Change ODCB bank structure.

4. Mods to ODLOSS.

7.5 Release notes version 4.09

1. New word in GNRL block: JGBTZ - T-zero of bunchlet.

2. New global smearing routine ODGLOB.

3. New correction routine for EB clusters, ODEBCO.

4. A few utility routines for ODSMGL moved from BT package.

7.6 Release notes version 4.08

1. New routines ODPEAL and ODPEON provided by Yoram Rozen to correct PE cluster positions with better alignment information, and to randomly kill Monte Carlo PE clusters in sectors which were dead in data. (See section **??**.)

2. New routine ODSMGL from Chris Darling provides smearing of track parameters by constant global factors. Handles tracks with z vertex constraints properly. (See section **??**.)

3. Modified ODTOSI to handle addition of SI(z) hits and z vertex constraints in a consistent way. Also properly treats addition of SI(x) hits from 1991 and 1992 DSTs.

7.7 Release notes version 4.07

1. The SW clusters and tracks are now stored in the SIW block. Additional cluster information useful for the SiW luminosity analysis is stored in a hidden ODSL bank.

2. The CTRK block has been extended to include three PB multiplicity wedge sums corresponding to the total multiplicity in U strips, V strips

and wires in a 5 mrad half-angle cone about the position to which a track is extrapolated at the barrel presampler.

3. ODTOSI now accepts 'SX' in addition to 'si' to switch to SI tracks with rphi hits only.

4. The tube chamber hit bits in the FDET block are now stored correctly (beginning with overall program version 1777).

5. Track parameters for the two parts of kinked tracks are stored in the ATRK block (beginning with overall program version 1784).

7.8 Release notes version 4.06

1. The SiW part of the FDET block has changed slightly.

7.9 Release notes version 4.05

1. Remove the tube chamber words from FDET

7.10 Release notes version 4.04

1. The FDET block has been extended to include more detailed information from the FD calorimeter, tube chambers, and drift chambers. The change is backwards-compatible, in the sense that the seven real words of the old FDET block are unchanged; the new data are added to the end of the previous real and integer words. The drift-chamber-only blocks no longer exist.

2. The FD Gamma Catcher is now included in the DST. It uses the FDET block, with the appropriate bit set in the bit-mask IF(JFSOUR,Iblk).

3. The variables JCZ0UC and JCTLUC were not what was claimed. The names have now changed to JCZ0CJ and JCTLCJ, because that's what they are.

4. The raw SI hits are stored.

5. The value of JCCHI3 is actually the χ^2 for the $s - z$ fit, not for the 3-d fit. This has always been the case and will remain so.

6. The dE/dx weights are now SIGNED quantities!

7.11 Release notes version 4.00

1. Users of FDET blocks *must* now use the bit mask `IF(JFSOUR,iclus)` to select the FD subdetectors they require. The new drift-chamber blocks are provided for DST-level monitoring, but are not yet calibrated for physics analyses. The other FDET blocks are unaffected.

2. Offsets are no longer parameters; they are variables, filled according to the number of words present. NOTE: this means that offsets like JCDEDX can no longer be used as dimensions in array declarations; one has to use parameters like NRCTRK.

3. A separate bank contains the raw ECAL block information.

4. The routine ODV0ZF was introduced for vertex fitting.

5. The ODFDET routine has changed completely.

7.12 Release notes version 3.13

1. Another forward detector block for driftchambers.

2. Another energy cluster word.

3. New version of ODTOSI to switch to SI data.

4. Include the ODDX patch with dE/dx improvements by M. Hauschild.