COMPACT 7.3 USER GUIDE

C. Biino, G. Bocquet, M. Clemencic, B. Hay, V. Kozhuharov,
A. Maier, M. Martini, R. Moore, I. Wingerter

F77 version

May 17, 2013

Contents

1 Installing COmPACT

The COmPACT installation files are all stored in the NA48 AFS offline directory called */afs/cern.ch/na48/offline2/compact/compact-7.3*. If the computer you work on has access to afs, you only need the reader file: *reader-7.3.tar.gz* which allows you to read COmPACT files. The description of the installation is in section 1.2. The user only needs the user routines, the analysis routines and the main program; these files are included in the *reader-7.3.tar.gz*. If you do not have access to afs, then you need the full compact file: *compact-7.3.tar.gz*. The description of the installation is in section 1.3.
When there is a compact update, the entire tar file is rebuild.

1.1 Compact directories on public afs

The compact public afs area is */afs/cern.ch/na48/offline2/compact/*. In this directory there are subdirectories which can be useful to the user, in particular since by default most of the COmPACT code is not copied to the user area.

- compact-n.m for previous compact versions (before 7.3).
- compact-7.3: contains the sources of the version 7.3. There are subdirectories:

  compact/lib : The COmPACT library (with 2 sub-directories *inc* and *src*).

  compact/rlib : The COmPACT reader library (with 4 sub-directories *inc*, *userinc*, *src* and *anasrc*).

  compact/zlib : The compression library.

  compact/doc : The documentation files.

  compact/tools: The compact tools directory (relevant to build the compact libraries).

- lib directory: contains the compact libraries.
- rlib directory: contains the compact reader libraries.
- There is one directory per operating system containing links the COmPACT libraries (libcompact and libreader).
  - .sun4m_53/lib
  - .alpha_osf20/lib
  - .hp700_ux90/lib
  - .linux/lib
  - .aix/lib

By default, the Makefile in the reader directory points to these directories on public afs.

1.2 Installing COmPACT for Reading with access to afs

Copy the file *reader-7.3.tar.gz* to your home directory then *gunzip* the file before running it through *tar*. To do this you type the following two commands:

```
gunzip reader-7.3.tar.gz
tar xvf reader-7.3.tar
```

or

```
tar xvzf reader-7.3.tar.gz
```

These commands uncompress the distribution file and create the directory structure for COmPACT. The result is a single directory called *reader* which contains the COmPACT code usefull to the user. An example of the output generated by these commands is shown below:

```
> cp /afs/cern.ch/na48/offline2/compact/compact-7.2/reader-7.2.tar.gz .
> gunzip reader-7.2.tar.gz
> tar xvf reader-7.2.tar
x reader, 0 bytes, 0 tape blocks
x reader/userinc, 0 bytes, 0 tape blocks
x reader/userinc/user.h, 12 bytes, 1 tape blocks
x reader/userinc/constants.h, 474 bytes, 1 tape blocks
x reader/userinc/F77ana_define.h, 269 bytes, 1 tape blocks
x reader/userinc/F77_ana.h, 1858 bytes, 4 tape blocks
x reader/userinc/user_cuts.h, 2470 bytes, 5 tape blocks
x reader/usersrc, 0 bytes, 0 tape blocks
x reader/usersrc/fuser_burst.F, 1554 bytes, 4 tape blocks
x reader/usersrc/fuser_cmpEvent.F, 1189 bytes, 3 tape blocks
x reader/usersrc/fuser_cmpEvent.example.F, 11167 bytes, 22 tape blocks
x reader/usersrc/fuser_cmpFilter.F, 1104 bytes, 3 tape blocks
.
.
.
.
x reader/obj, 0 bytes, 0 tape blocks
x reader/depends, 0 bytes, 0 tape blocks
x reader/.last.f77.compile.h, 0 bytes, 0 tape blocks
x reader/.last.c.compile.h, 0 bytes, 0 tape blocks
x reader/compact.job, 5320 bytes, 11 tape blocks
x reader/runcompact symbolic link to
/afs/cern.ch/na48/offline2/compact/tools/compact.pl
x reader/CompactUG-7.2-F.ps symbolic link to
/afs/cern.ch/na48/offline2/compact/compact-7.2/doc/CompactUG-7.2-F.ps
x reader/src, 0 bytes, 0 tape blocks
x reader/src/compact_main.c, 42856 bytes, 84 tape blocks
x reader/.__templist, 138 bytes, 1 tape blocks
x reader/CompactUG-7.2-C.ps symbolic link to
/afs/cern.ch/na48/offline2/compact/compact-7.2/doc/CompactUG-7.2-C.ps
x reader/.compact-status, 3 bytes, 1 tape blocks
x reader/test.list, 472 bytes, 1 tape blocks
x reader/Makefile, 10708 bytes, 21 tape blocks
x reader/userana, 0 bytes, 0 tape blocks
x reader/last.kumac, 152 bytes, 1 tape blocks
>
```

To perform an initial test of COmPACT change to the new *reader* directory and run the GNU make program. The commands to do this are:

```
cd reader
gmake
```

COmPACT will now compile with the default C user routines. These will print out a text dump of the first 20 events of a burst to a file called *compact.txt*. If COmPACT fails to compile ensure that you are running on a supported machine (Linux, DEC Unix, SunOS or HP-UX9) and then that you have the GNU C compiler and CERN libraries installed in the standard places, if not you may need to edit the Makefile.
To use rfio (i.e. to have the possiblity to access data remotely and data stored inside the *CASTOR* system) one needs to modify the Makefile (around line 14):
change

```
USE_RFIO = no
```

to

```
USE_RFIO = yes
```

IMPORTANT

Only the C or F77 user routines are compiled in to COmPACT, not both. The default is to use the C routines, however uncommenting the line containing “F77DEF=....” in *Makefile* by removing the “#” will cause the F77 routines to be called instead.

```
> cd reader
> gmake
Makefile:319: depends/compact_main.d: No such file or directory
Makefile:319: depends/user_init.d: No such file or directory
Makefile:319: depends/user_burst.d: No such file or directory
Makefile:319: depends/user_superBurst.d: No such file or directory
Makefile:319: depends/user_cmpEvent.d: No such file or directory
Makefile:319: depends/user_ke3Event.d: No such file or directory
Makefile:319: depends/user_kmu3Event.d: No such file or directory
Makefile:319: depends/user_mcEvent.d: No such file or directory
.
.
.
.
gmake compact
gmake[1]: Entering directory ‘/u/vl/wingerte/testcmp/reader’
gcc -O
> -I. -I/afs/cern.ch/na48/offline2/compact/compact-7.2/compact/lib/inc
-I/afs/cern.ch/na48/offline2/compact/compact-7.2/compact/rlib/inc
-I/afs/cern.ch/na48/offline2/compact/compact-7.2/compact/rlib/anainc
-I./userinc
> -I/afs/cern.ch/na48/offline2/compact/compact-7.2/compact/zlib
 -c src/compact_main.c -o ./obj/compact_main.o
.
.
.
.
>
```

1.3 Installing COmPACT for Reading without afs

Copy the file *compact-7.3.tar.gz* to your home directory then *gunzip* the file before running it through *tar*. To do this you type the following two commands:

```
gunzip compact-7.3.tar.gz
tar xvf compact-7.3.tar
```

or

```
tar xvzf compact-7.3.tar.gz
```

These commands uncompress the distribution file and then create the directory structure for COmPACT. The result is a single directory called *compact* which contains all the COmPACT code. An example of the output generated by these commands is shown below:

```
blocksize = 128
x compact/
x compact/lib/
x compact/lib/Makefile, 4907 bytes, 10 tape blocks
x compact/lib/compact-7.2.x, 38680 bytes, 76 tape blocks
x compact/lib/inc/
x compact/lib/inc/F77common.h, 4886 bytes, 10 tape blocks
x compact/lib/inc/geometry.h, 595 bytes, 2 tape blocks
x compact/lib/inc/physics.h, 593 bytes, 2 tape blocks
x compact/lib/inc/compact-2.2.h symbolic link to compact.h
x compact/lib/inc/compact.h, 23712 bytes, 47 tape blocks
x compact/lib/inc/offsets.h, 18579 bytes, 37 tape blocks
x compact/lib/inc/compact-3.0.h symbolic link to compact.h
x compact/lib/inc/compact-3.1.h symbolic link to compact.h
x compact/lib/inc/compact-3.2.h symbolic link to compact.h
x compact/lib/inc/compact-3.3.h, 23712 bytes, 47 tape blocks
.
.
.
.
x compact/reader/.compact-status, 3 bytes, 1 tape blocks
x compact/reader/test.list, 472 bytes, 1 tape blocks
x compact/reader/Makefile, 10708 bytes, 21 tape blocks
x compact/reader/userana, 0 bytes, 0 tape blocks
x compact/reader/last.kumac, 152 bytes, 1 tape blocks
```

You will then need to compile the two COmPACT libraries:

- libcompact: go into compact/lib directory.
  - Change in the Makefile USE_RFIO =yes to USE_RFIO =no (line18) if your computer dont support rfio.
  - `gmake`
  - When the Makefile has finished, you should have a file called: libcompact.7.3.a
- rlib: go into compact/rlib directory.
  - Change PUBLIC_AFS =yes to PUBLIC_AFS =no in the Makefile (line 16).
  - `gmake`
  - When the Makefile has finished, you should have a file called: libreader.7.3.a

When the two COmPACT libraries are done, you need to go to the directory compact/reader and edit the Makefile to ensure that the libraries are taken from your account and not from afs. You simply need to change the line (17):

```
PUBLIC_AFS = yes
```

to

```
PUBLIC_AFS = no
```

Then the procedure is as explained in the previous section 1.2. The tar file can then be removed.

2 Running COmPACT

2.1 The COmPACT executable

If all went well in the installation phase you now have and executable program called "compact" sitting in the directory. To have a look at the list of command line options available just type:

```
compact -h
```

or just

```
compact
```

To run COmPACT on a data file the most useful options are "-i", which allows you to specify a single input file, and "-l" which specifies a text file containing the name of a data file on every line.
In the directories /afs/cern.ch/na48/offline2/compact/FilterProd/summaryXX one can find lists of compact and super-compact files available on disk (.list) or through the *CASTOR* system (.castor).
To make compact read one of these files and produce a print out of the first 20 events type of a file:

```
compact -i /castor/cern.ch/na48/na48prod/2004/run/SC/gcmp2416702_1090132191.scmp.1
```

This produces the text file *compact.txt* which contains the event print out.
The online "help" (generated when you type `compact` or `compact -h` shows:

```
user@lxplus007 1:50pm [reader]compact -h
*COmPACT*  - Compact command arguments:
*COmPACT*  -    -i   <file>      declare input file name (replaces s/e/n options)
*COmPACT*  -    -l   <file>      give input file list (replaces s/e/n/i options)
*COmPACT*  -    -co  <file>      declare COmPACT output file name
*COmPACT*  -    -ko  <file>      declare Ke3 output file name
*COmPACT*  -    -kmo <file>      declare Kmu3 output file name
*COmPACT*  -    -mo  <file>      declare MC output file name
*COmPACT*  -    -so  <file>      declare SuperCOmPACT output file name (user filter)
*COmPACT*  -    -soc <file>      declare SuperCOmPACT output file name (charged filte
*COmPACT*  -    -son <file>      declare SuperCOmPACT output file name (neutral filte
*COmPACT*  -    -ho  <file>      declare HyperCOmPACT output file name (user filter)
*COmPACT*  -    -hoc <file>      declare HyperCOmPACT output file name (charged filte
*COmPACT*  -    -hon <file>      declare HyperCOmPACT output file name (neutral filte
*COmPACT*  -    -scprod-cuts     set default for looser filter cuts (SCProd)
*COmPACT*  -    -nokl2           disable kl2 filter in the neutral split (enabled by 
*COmPACT*  -    -smo <file>      declare MC output file name
*COmPACT*  -    -sl  #           set SuperCOmPACT output level
*COmPACT*  -                     default (no option): epsilon standard super-compact
*COmPACT*  -                     #=16+iflag (iflag=1,2,4) for 2pi0, 3pi0, 2 gamma res
*COmPACT*  -                     #=32 for Dalitz summary
*COmPACT*  -                     #=64 for rare decay summary
*COmPACT*  -                     #=128 for four-trackssummary
*COmPACT*  -                     #=a combination of 16+iflag, 32, 64, 128 is also acc
*COmPACT*  -    -srw             rewrite SuperCOmPACT integer event structure
*COmPACT*  -                     from float version
*COmPACT*  -    -db              access cdb (compact database)
*COmPACT*  -    -ndb             do not access cdb (compact database) This is the def
*COmPACT*  -    -cheat           transform compact events into super-compact
*COmPACT*  -    -skip-kab-rec    turn off Kabes reconstruction (when reading cmp even
*COmPACT*  -    -rmin <RMIN>     Only analysis bursts for which run>=RMIN
*COmPACT*  -    -rmax <RMAX>     Only analysis bursts for which run<=RMAX
*COmPACT*  -    -nevt <NEVT>     Only analysis NEVT events and then stops
*COmPACT*  -    -empty #         Empty Burst, EoB, evt lists structures:
*COmPACT*  -                     to save disk space when filtering events
*COmPACT*  -                     #=0: empty burst, evt lists, EoB
*COmPACT*  -                     #=1: empty burst
*COmPACT*  -                     #=2: empty evt lists
*COmPACT*  -                     #=4: empty EoB
*COmPACT*  -                     #=6,3,.. a combination of above is also accepted
*COmPACT*  -   -string <string>  Pass an arbitrary string. This string is available a
*COmPACT*  -   -h                print this help message
```

2.2 Description of the command line options

FIX-ME: must be updated

- **-i** Selection of one compact input file:
  compact -i /castor/cern.ch/na48/na48prod/2004/run/SC/gcmp2416702_1090132191.scmp.1

- **-l** Run compact with a list of input files:
  compact -l /afs/cern.ch/na48/offline2/compact/FilterProd/summary2003/SC_SS1-SS2-SS3_pass2.list: compact will read the SS1,SS2,SS3 super-compact data.
- **-co** Output compact events into the file specified.
  compact -l input.list -co high-mass.cmp:
  compact will read files in *input.list* and write required events (selected from

user_cmpFilter routine) into the file *high-mass.cmp* inside the working directory.

- **-ko** Same as for option -co but for events with K_e^3 compact format. The selection should be done from user_ke3Filter routine.
- **-kmo** Same as for option -co but for events with K_μ^3 compact format. The selection should be done from user_kmu3Filter routine.
- **-mo** Same as for option -co but for monte-carlo events. The selection should be done from user_mcFilter routine.
- **-so** Same as for option -co but the output will have super-compact format. This option is used when producing super-compact samples. As for -co option, the user_cmpFilter routine is used to select events if input data have compact format; if input events have super-compact format, routine user_userCmpFilter should be used.
- **-soc** Output selected events into supercompact. Selection is done by user_superSel3pic() routine in the standard compact library (so-called charged split).
- **-son** The same as -soc option but the selection is done by user_superSel3pin() routine in the standard library (so-called neutral split).
- **-ho** This option is included for filling of user-defined hyperCompact. The selection is done by user_userCmpFilter routine where the hyperCmpEvent structure should be filled. It is the one written to the output file.
- **-hoc** Writing hyperCompact events to the output file starting from SuperCompact. Selection is done inside user_superSel3pic() routine (so-called charged split).
- **-hon** Same as -hon but the selection is done by user_superSel3pin() routine in the standard library (so-called neutral split).
- **-scprod-cuts** An initialization of the cuts for sel3pin and sel3pic routines is done. Cuts are reset to be the ones used during SuperCompact splits production.
- **-scprod-mode** Used during main production of SuperCompact, HyperCompact, Charged and Neutral splits.
- **-nokl2** Doesn't write the Kl2 events into the neutral split. By the default it is not set (i.e the neutral split contains Kl2 events)
- **-smo** Same as for option -so but for super-compact monte-carlo events. The selection should be done from user_superMcFilter routine if input events have super-compact format or from user_McFilter if input events have mc-compact format.
- **-rwts** Rewrite the time stamp of the previous event with the time stamp of the previous event in the input file. Value '-1' is used if a first event in a burst. This option is valid only for Compact and SuperCompact input files

- **-b** This option is used in conjonction with the previous six options to create output files with a fixed size. *compact -l input.list -co high-mass.cmp -b 200MB*: compact will create multiple consecutive output files with 200Mbytes size (the output files will be called high-mass.cmp.1, high-mass.cmp.2, ...).
- **-ndb** With this option (*compact -l input.list -ndb*) the compact (or burst) database is not read - default.
- **-cheat** This option allows the user to read input files with compact format (cmpEvent or mcEvent) but to analyse them as super-compact events (superCmpEvent or superMcEvent). This is usefull when one has a running super-compact program but super-compact have not be produced. Of course, the program takes longer than if the input data were super-compact. . . . . . .
- **-rmin** This option allows to specify a minimum run number. *compact -l input.list -rmin 10540*: only data with $RunNumber >= 10540$ will be analysed. If the input list contains data with $RunNumber < 10540$, data will be readin by compact but not given to analysis routines (i.e. the user routines will only *see* data for the requested runs.
- **-rmax** Same as *rmin* but for maximum run number.
- **-nevt** Limits the number of events to be analysed. When the limit is reached, compact stops.*compact -l input.list -nevt 9999*
- **-empty** This allows to suppress the content of some big compact structure which are known to be no use for the foreseen analysis. The above listing is clear to understand how to handle this option. *compact -l input.list -co rare.cmp -empty 0*: the output file will not contain any data inside the *Burst*, *EndofBurst* structures and the *event lists* will be empty. This is particularly useful when filtering compact data (like rare decays comapct filters) for which this history information is not useful.

2.3 Accessing the run database

FIX-ME: must be updated In the reader directory there is perl script called runcompact_pc. It invokes another perl script called sms_select.pl which connects to the run database. These scripts allow to find out where the data produced by L3 (goldraw,goldcmp,compact) are.

If one just types:

```
runcompact_pc
```

one gets

```
Usage: runcompact\_pc selections [options]
 selections: -f FirstRun[:FirstBurstInRun]
             -l LastRun[:LastBurstInRun]
             -s StartTime (format YYYY_MM_DD_HH_MM_SS or UNIX timestamp)
             -e EndTime (format YYYY_MM_DD_HH_MM_SS or UNIX timestamp)
```

```
            -t BeamType e.g KS, KSKL, test, ...
            -u file_name containing bursts to be processed
options:    -n Max_Number_of_Bursts
            -o Max_Number_of_Bursts with tape optimization
            -p stream_name change input stream (default goldcompact)
            -b send your jour to NQS batch system (qsub)
            -i run interactively
            -d dry run: generate compact.list and quit
            -D debugger program   run your favourite debugger

You have to make at least one selection,
if you specify more criteria, they are put in AND
```

3 Writing Code for COmPACT

The directory structure has been designed so that all the COmPACT library source code and header files are stored in the *lib* directory on /afs/cern.ch/na48/offline2/compact/compact-7.3.

The COmPACT reader code is kept in a library as well; the source code and header files are stored in the *rlib* directory on /afs/cern.ch/na48/offline2/compact/compa
7.3.

The user code and any user header files required are placed in the *usersrc* and *userinc* directories where they will be automatically compiled and linked by the make file. The user code consists of several routines:

Routine	**Called when**
user_init	program startup
user_burst	new burst header read
user_superBurst	new superCOmPACT burst header read
user_hyperBurst	new hyperCOmPACT burst header read
user_cmpEvent	COmPACT format event read
user_cmpFilter	COmPACT event read and COmPACT output file active
user_ke3Event	Ke3 format event read
user_ke3Filter	Ke3 event read and Ke3 output file active
user_mcEvent	MC format event read
user_mcFilter	MC event read and MC output file active
user_superCmpEvent	SuperCOmPACT format event read
user_superCmpFilter	SuperCOmPACT event read and SuperCOmPACT output file acti
user_hyperCmpEvent	HyperCOmPACT format event read
user_eob	end of burst structure read
user_superEob	SuperCOmPACT end of burst structure read
user_exit	end of program

All these routines must return an integer which is zero if no errors are encountered. For the filter routines a negative number indicates that the event should be written to the output file. Any other number will generate a warning message from COmPACT.

To add your own code modify the relevant dummy user routine supplied. These are carefully commented to show where to add your code and contain a small header needed to interface to COmPACT. The variable names of the various quantities stored for each event and burst are given in appendix I. (C and FORTRAN names are on different files: CompactUG-7.3-C.pdf and CompactUG-7.3-F.pdf)

IMPORTANT
Only the C or F77 user routines are compiled in to COmPACT, not both. The default is to use the C routines, however uncommenting the line containing “F77DEF=....” in *Makefile* by removing the “#” will cause the F77 routines to be called instead.

Each user routine has it’s own file. For the C interface these are called “user_xxxx.c” and for the F77 interface “fuser_xxxx.F”.

4 Analysis routines in COmPACT 7.3

From version 4.1 the analysis routines are in the reader library rlib, i.e. they are not anymore in the reader/userana directory. The user has the possibility to modify the routines:

- copy the relevant routine from /afs/cern.ch/na48/offline2/compact/compact-7.3/compact/rlib/anasrc to reader/userana (and the necessary include files from ...../rlib/anainc to reader/userinc
- add the copied file name in the Makefile in the section:

  `UCASRCS =`

  if its a C routine.

  `UFASRCS =`

  if its a fortran routine.
- gmake

IMPORTANT

In version 4.1, most of the analysis routines are called by default, some are called on conditions, *tagtime* is called by default ONLY for good charged and neutral events, because of its CPU use. A selection mechanism has been set-up; its description is given in 8.7 to alter the COmPACT default. The routine steering the analysis routines is in /afs/cern.ch/na48/offline2/compact/compact-7.3/compact/rlib/src/cmpAnalysis.c

4.1 user_stdcmpch.c (H. Fox)

This file contains routines:

- USER_STD_EP which computes quantities related to the tracks: $\frac{E}{p}$ stored in $evt->track[i].EovP$ [TRACK_EOVP(i)], Cluster associated to the track stored in $evt->track[i].LKRclu$ [TRACK_LKRCLU(i)].

4.2 fuser_hodotime.F (M. Lenti)

The USER_HODOTIME routine computes the event time measured with the charged hodoscope which is stored in $evt->achod.hodotime$ [ACHOD_HODOTIME] and other quantities which are stored in structure anacharghod [ACHOD_xxx].

4.3 fuser_lkrtime.F (Lydia Fayard)

The USER_LKRTIME routines computes the event time for neutral events, measured using at most 8 cells from the LKR clusters selected by the neutral selection routine (USER_SEL2PI0). Variables $evt->aneut.LKRtime$ [ANEUT_LKRTIME], $evt->aneut.LKRNHODtime$ [ANEUT_LKRNHODTIME] and $evt->aneut.ntUsed$ [ANEUT_NTUSED] (part of the ananeut structure) are filled by LKR_TIME. The routine is documented in the code.

4.4 fuser_badburst.F (Lydia Fayard)

The USER_BADBURST routine fills the $bur->BadB$ [BUR_BADB_xxx] structure with non zero values if the burst is declared *bad* for the analysis. This routine has to be called inside the USER_BURST routine and then events can be rejected, according to the flags, inside USER_CMPEVENT routine.

4.5 fuser_espy.F (M. Velasco)

This routine computes the energy behind the tracks, measured by the Neutral trigger; this quantity is stored in $evt- > track[itrack].Espy$ [TRACK_ESPY(itrack)] which is a volatile variable i.e. is not saved on disk. If there was no NUT data in at least one view -8888. is filled. If the distance between the track extrapolation to LKR and the column center is larger than 4cm for BOTH views -9999. is filled. If $|Ex-Ey|/Emax >$ 20%, the smallest energy is filled with negative sign.

4.6 fuser_bluefield.F (J.B. Cheze)

This routine has to be applied for 1997 data but not for the next years; for 1998 and later data, the correction from the field in the blue tube is already applied at the reconstruction level. This routine recomputes quantities related to vertex and tracks, taking into account the magnetic field in the blue tube; for the time being, the recomputed quantities are stored in compact free variables; they may, one day, replace the standard variables. c evt-¿vertex[ivertex].anavar[1] = corrected evt-¿vertex[ivertex].y

4.7 fuser_lkrpedcor.F (A. Ceccucci)

```
/************************************************************/
/* Compact routines to correct 1997 data for LKR pedestal     */
/* variations.  (Augusto Ceccucci)                            */
/*                                                            */
/* The corrections have been measured on groups               */
/* of 40 bursts. Files containing the averages are stored in  */
/* the compact/GeomFiles directory.                           */
/* The correction is not applied to the compact event but     */
/* the correction are stored in a "volatile" structure        */
/*                                                            */
/* LKR_ANAVAR(iclu,1): ecorrke3 corrected for ped. shift (GeV) */
/*                     (evt-lkr[iclu].anavar[0])              */
/* LKR_ANAVAR(iclu,2): pedestal correction (in MeV)           */
/*                     (evt-lkr[iclu].anavar[1])              */
/************************************************************/
```

From 23-06-98, use latest updated correction flags.

4.8 user_eobdec (G. Barr)

```
/**********************************************************************/
/* int errdvec_(int v[],int vlen);                                     */
/* v[]     input  Array containing the ProcError array                  */
/* vlen    input  Length of data in v[] (=NProcError)                   */
/* This routine fills the structure eob->aerrdec.decRes[idec]           */
/* (i.e. in fortran EOB_AERRDEC_DECRES_XXXX(idec+1)                     */
/*                                                                      */
/* eob->aerrdec.decRes[idec].code    : y    y    y  0=root,1=echan,2=log */
/* eob->aerrdec.decRes[idec].echan   : n    y    y  echan               */
/* eob->aerrdec.decRes[idec].log     : n    n    y  log                 */
/* eob->aerrdec.decRes[idec].Nerr    : y    y    y  #errors = #calls    */
/* eob->aerrdec.decRes[idec].StatFlag: y    y    n  sat flag            */
/* eob->aerrdec.decRes[idec].evtno   : y    y    n  evtno               */
/* eob->aerrdec.decRes[idec].ts      : y    y    n  ts                  */
/* eob->aerrdec.decRes[idec].x1      : y    y    n  ???                 */
/* eob->aerrdec.decRes[idec].x2      : y    y    n  ????                */
/*                                                                      */
/* EOB_AERRDEC_DECRES_CODE(idec)     : y    y    y  0=root,1=echan,2=log */
/* EOB_AERRDEC_DECRES_ECHAN(idec)    : n    y    y  echan               */
/* EOB_AERRDEC_DECRES_LOG(idec)      : n    n    y  log                 */
/* EOB_AERRDEC_DECRES_NERR(idec)     : y    y    y  #errors = #calls    */
/* EOB_AERRDEC_DECRES_STATFLAG(idec) : y    y    n  sat flag            */
/* EOB_AERRDEC_DECRES_EVTNO(idec)    : y    y    n  evtno               */
/* EOB_AERRDEC_DECRES_TS(idec)       : y    y    n  ts                  */
/* EOB_AERRDEC_DECRES_X1(idec)       : y    y    n  ???                 */
/* EOB_AERRDEC_DECRES_X2(idec)       : y    y    n  ????                */
/************************************************************************/
```

4.9 fuser_nhodtime (M. Lenti)

4.10 fuser_aklflag (F. Marchetto)

```
C*** Routine to interface Compact output to Super-Compact
C*** It should be used also to over-write Variable EVT_AKLTIME
C*** and EVT_AKLERRFLAG in the COmPACT Structure cmpEvent
C***
C*** TimeAkl is the time of the Akl hit closest to Hodoscope
C*** time, i.e. EVT_HODTIME. If EVT_HODTIME is zero, then EVT_NHOTIME is
C*** used. With a proper and small adjustment one could use the event
C*** time as reference.
C*** FlagAkl is an overall flag which is built acording to the number of hit
C*** in a given window:
C*** FlagAkl = 100*Nhit1rms + 10*Nhit2rms + Nhit3rms
```

```
C*** where Nhit1rms is the number of hits within a 1 rms ( 1 ns),
C*** Nhit2rms is the number of hits in the 1-2 rms window (1 to 2 ns)
C*** Nhit3rms is the number of hits in the 2-3 rms window (2 to 3 ns)
C*** [There is a undetermination in this definition: in example 10 hits
C*** in the 1 to 2 rms range give a flag as a single hit in the 1 rms
C*** window. This is extremely unlikely]
```

4.11 fuser_lkrposcor (G. Unal)

This routine corrects the cluster position (applied to 1997, 1998 and 1999 data) using measurements obtained from Ke3 analysis. The corrected x and y positions are stored respectively into LKR_ANAVAR(i,3) and LKR_ANAVAR(i,4) ($lkr[iclu]->anavar[2]$ and $lkr[iclu]->anavar[3]$). Value used are in the files compact/GeomFiles/cpd_pos_97.txt, cpd_pos_98.txt., cpd_pos_99.txt.

4.12 user_TrackVertexCor (I. Wingerter)

This routine steers the correction routines for the spectrometer: geometry, magnet and field in blue field corrections. It is called for all events and calls the required correction routines. Vertices are recomputed after the corrections were applied

Please note that from the 19-11-98, the correction for the field in the blue tube is NOT available in the evt->trackCorr.xxx (TRACKCORR_XXX) structure; it is available for vertices in the evt->vertexCorr.xxx (VERTEXCORR_XXX) if the correction was required. This remark also applies to super-compact: the tracks dont have the correction due to the field in the blue tube applied; the charged summaries do. This is because this correction is applied to the tracks belonging to a given vertex; if a track belongs to more than one vertex, the correction would be applied twice; therefore to correct the vertex it is applied to tracks, then the vertex is computed, then the correction is removed from the track.

4.13 fuser_lkrcalcor (G. Unal)

```
      subroutine user_lkrcalcor_event(IB, QB, IE, QE)
c
c correct cluster energy for calibration effect
c      (1) eta factors for intercalibration 97
c      (2) Time variation of energy scale
c      (3) Overall energy scale factor=1.0029 (should be correct to =< 5*10*
c
```

```
This routine (valid only for 1997 data) works as the previous lkr
correction routines: it overwrites LKR_ANAVAR(i,1) (lkr[i]->anavar[0])
with the corrected value.
```

4.14 fuser_lkrcalcor98 (G. Unal)

This routine contains all the applied neutral corrections for 1998 and 1999 data. It also applies the necessary correction to monte-carlo events.

```
c
c correct cluster energy
c   (0) E vs radius close to beam tube
c   (1) Zero suppression effect
c   (2) Energy loss at low photon energy (from MC)
c   (3) Sharing correction, bias in reconstruction from MC
c   (4) Sharing correction, data/MC shower profile
c   (5) Sharing correction, beam tube effect
c   (6) Energy non linearity (E/p correction)
c   (7) Time variation of Escale (makes sense only if Pedestal correction do
c   (8) Overall Escale
c   (9) Space charge effect correction
c  (10) Pedestal shift variation
c  (11) Eta intercalibration for 99
c
c if icorr = 0, apply corrections on data/MC according to
c    best knowledge
c if icorr !=0 apply only specified corrections (according to
c    bits above)   !! You should know what you are doing !!
c
c  returns in function value word corresponding to corrections done
c
```

The variable LKR_ANAVAR(i,5) (lkr[i]-¿anavar[4]) is filled with the energy measured in the two samples before the pulse.

4.15 murec0999 (T. Gershon)

This routine is the updated version of the muon reconstruction in order to handle 1999 data and MC data.

4.16 user_lkrcalhi2k (A. Ceccucci)

This routine applies a correction to the cluster energy, following the change of calibration due to the change in Lkr temperature which occured during August 2000.

```
/*****************************************************/
/* COmPACT user routine: user_lkrcalhi2k_init()      */
/*                                                   */
/* User routine called to read the correction to the */
/* LKr calibration for the data collected in 2000    */
/*                                        AC 30/1/01 */
/*****************************************************/
```

4.17 fuser_ghost.F (G.Unal)

This routine was developped in order to reject ghost track. This routine is called for super-compact events and fills sevt->summary[isum].Char[0].anaflag[1] with 1 if the vertex is made with one ghost track (from K_{e3} or $K_{\mu3}$).

4.18 fuser_lkrsmear.F (G.Unal)

This routine smears the cluster energy (in evt-¿lkr[i].anavar[0] [LKR_ANAVAR(1)])for monte-carlo events.

4.19 mc_etail.F (G.Unal)

This routine compute s effect on energy from resolution tails; the modified energy is stored in sevt-¿cluster[iclu].spare[0] [SCLUSTER_SPARE(1)].

4.20 fuser_pedint.F (M.Scarpa)

```
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c  Feb. 2002 Marcella S.                                         c
c  Ferrara pedestals and bad runs/bursts list (epsi,KS,KL runs)  c
c  4 channels ks/l : 1us 200ns 3us 15us                          c
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
```

This routine subtract pedestal values and stores bad bursts/runs list. The 4 KS(L) channels' pedestal values are stored in pedks(l) burst by burst and subtracted to beam-monitor intragrator compact variables. The compact variables are over-written, so the old value must be stored before. The routine fill also FEBAD_KS(L): negative value (-1) means bad burst or run.

```
c ----------------------------------------------------------------------
c       FEBADR_KS/L: bad runs & burst ks/l channels
c                    -1 bad
c                     0 ok
c                     2 oscillatory behavior but peds computed anyway

        INTEGER FEBAD_KS,FEBAD_KL
        REAL pedks,pedkl
        COMMON/FEMON/pedks(4),pedkl(4),FEBAD_KS,FEBAD_KL
c -----------------------------------------------------------------------
```

IN COMPACT the routine (if wanted) must be called by the user in fuser_cmpEvent.F (user_cmpEvent.c) as

```
CALL USER_PEDINT(bur,evt)   (USER_PEDINT(bur,evt);)
```

IN SUPERCOMPACT the pedestals are always applyed ad the routine is called in cmp2scmp.c
If the routine is called a bit (27) is set in the evt->FlagCorr variable. This will be more usefull when we will make the new compact version and we will introduce a flag to call by default the routine.

4.21 fuser_dcholes.F (M.Scarpa)

From version 6.2.

```
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c  Sept. 2002 Marcella S.                                              c
c  catch the hole(s) in dch r/o wire maps through eob rate infos       c
c  1 connector = 16 wires = 1 card in old r/o picture                  c
c  new r/o 1 cmc = 2 connectors                                        c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
```

In *fuserhole.h* the COMMON/HOLESdch/holes(24,16) is defined, used to house the hole-nothole definition for each card(16) in each plane(24: dch1-2-4). A card is a 16 wires group as was defined in the old DCH read-out picture.
The *fuserhole.F* routine detects the holes and fills the array holes(24,16). The strategy is to look at the rate per card(16), in each plane(24), and 'catch' the holes in two steps:

- 1) flag (flag1) the low rate cards w.r.t cut1
  Also the low rate cards at the edges are flagged
- 2) flag (flag2) the asymmetric cards w.r.t cut2
  The edges are excluded from this flag. An hole is found if both flag1 and flag2 are fired and the correspondig card index is filled in holes(plane,card)

The rate per card is computed in raw and stored in the DCHRATES2 common block according to the following relation:
100*(number of hits per card)/(number of hits per plane), where

- #hits per card is stored in eob->SGNeff ($EOB\_SGNEFF(384)$) (16 cards X 24 planes=384)
- #hits per plane is stored in eob->SGNineff ($EOB\_SGNINEFF(384)$) (24 planes from DCH 1-2-4)

All these variables are end-of-burst information and they will be trasmitted to the SuperCompact *eob*-structure.

4.21.1 6.2 new compact variable based on DCH info

In order to allow multiplicity and accidental related study new compact structure *DCHmult* has been introduced which variables are taken from raw common blocks. The same structure is then passed to SuperCompact $SCDCHmult$.

1. The number of hits in the $[-25, 175]$ ns MBX time window, per plane for DCH1-2-4 are coded in:
   evt->DCHEFFmult.MBXPlaneEff[4] (DCHEFFMULT_MBXPLANEEFF(4))
   The maximum number of hit per plane is 64, i.e. 6 bits. We limit to 5 bits. The 24 numbers are coded in 4 words. The less significant bit of the first word is for the first plane od DCH 1, see 4.22. These information is elaborated in 4.22 to have an hint about the accidental activity (see evt->DCHEFFmult.MBXmult).
2. multiplicity in the tighter MBX time window $[-10, 155]$ ns, per plane for DCH 1:
   evt->DCHEFFmult.L1Trk24Eff[2] (DCHEFFMULT_L1TRK24EFF(2)) coded as before. These information are elaborated in 4.23 to determine which events should have been triggered by the L1=2trk+4trk and the result stored in a flag passed to SuperCompact evt->Trk24ON (DCHEFFMULT_TRK24ON).
3. DCH snow effect flagging evt->DCHEFFmult.DCHSnowErr[2] (DCHEFFMULT_DCHSNOWERR(2)) two bit words. The bit meaning is described in table 1
   First word: bit 0-14 dcherr1; bit 16-31 dcherr2.
   Second word: bit 0-14 dcherr3; bit 16-31 dcherr4. These words are in superCompact as well.

4.22 fuser_mboxeff_SC.F (F.Marchetto - M.Holder)

From version 6.2.

0	bit error in 9th time bit plane 8 dch2
1	first word is not a header
2	plane number not as expected
3	chamber number not as expected
4	wrong word count
5	header TimeStamp different from global TS
6	TS difference between headers
7	crate collect status word TS different from global TS
8	token time out
9	data ready time out
10	header-CSC time mismatch
11	less headers than expected
12	more headers than expected
13	hit time out of [-150,350] ns window
14	only for DCH2 or DCH4: snow flag (1—2—4—6—7—11—12).and.14

Table 1: evt->DCHEFFmult.DCHSnowErr description

```
      Function user_mboxeff_SC(ISB,QSB,ISE,QSE)
C***     The output of the routine is a flag stored in user_mboxeff.
C***     ONLY DCH1, DCH2, and DCH4 are considered
C***     The procedure is the following:
C***  a) get the number of hits per plane (hit -> drift time between
C***    -25 and 175)
C***  b) for each view is extracted the number of hits defined as
C***     the minimum number of hits between the two planes making a view
C***  c) the number of extra hits for each view is defined as
C***     the number of hits per view - 2 (where two is the number of hits
C***     expected for a decay channel with two charged particles in the
C***     final state)
C***  d) result is coded into user_mboxeff: for example
C***     user_mboxeff = 3 -> at least 3 views in DCH1 have .ge. Nextra_hit
C***     user_mboxeff = 15 -> at least 3 views in DCH2 have .ge. Nextra_hit
C***     user_mboxeff = 75 -> at least 3 views in DCH4 have .ge. Nextra_hit
C***     etc...
```

4.23 fuser_L1trk24Eff.F (N.Cartiglia)

From version 6.2.

```
      function is_it_l1ok(IB,QB,IE,QE)
C---
C--- Routine to determine which events should have been triggered
C--  by the L1=2trk+4trk
```

Table 2: Routines for the selection of 3π events

Event type		Charged	Neutral
cmpEvent	C	user_sel3pic(bur,evt)	user_sel3pin(bur,evt)
	F	USER_SEL3PIC(IB,IE)	USER_SEL3PIN(IB,IE)
superCmpEvent	C	user_superSel3pic(sbur,sevt)	user_superSel3pin(sbur,sevt)
	F	USER_SUPERSEL3PIC(ISB,ISE)	USER_SUPERSEL3PIN(ISB,ISE)
hyperCmpEvent	C	user_hyperSel3pic(hbur,hevt)	user_hyperSel3pin(hbur,hevt)
	F	USER_HYPERSEL3PIC(IHB,IHE)	USER_HYPERSEL3PIN(IHB,IHE)

```
C---
C     The output of the subroutine is in L1eff
C     l1eff = 1 then the event was trigger also by "ge2trk+ge4trk"
C
```

This routine uses the packed words DCHEFFMULT_MBXPLANEEFF(i) and the array DCHEFFMULT_L1TRK24EFF(i) as input.
See the code for the unpacking of DCHEFFMULT_MBXPLANEEFF.
The same routine exists in SuperCompact and it is called by syperCmpAnanlysis.c and it is filling the variable sevt$-$ >SCDCHEFFmult.Trk24ON (SCDCHEFFMULT_TRK24ON).

5 Analysis routines in COmPACT (versions $\geq$ 7.1)

5.1 Selection routines for 3π events (M. Sozzi)

For the standard selections of $K^{\pm} \to \pi^{\pm}\pi^{+}\pi^{-}$ and $K^{\pm} \to \pi^{\pm}\pi^{0}\pi^{0}$ a set of routines is provided (see Table 2).
Those reoutines fill the C structures (FORTRAN common blocks) ana3pic and ana3pin (ANA3PIC and ANA3PIN), see Table 3 and Table 4.

5.2 Filter routine for $Kl2$ decays (L. Fiorini)

The routine `KL2FILTER`[1], defined as

```
INTEGER FUNCTION KL2FILTER(ISB,QSB,ISE,QSE)
INTEGER ISB(2),ISE(2)
REAL*4  QSB(2),QSE(2)
```

inside the file `rlib/anasrc/kl2filter.F`, has been provided by Luca Fiorini to filter $Kl2$ events. It takes as input the superBurst and the

[1]C/C++ programmers should call it according to the prototype:
`int kl2filter_(superBurst*, superBurst*, superCmpEvent*, superCmpEvent*).`

Table 3: Sturucture filled by the 3π charged selection

C	FORTRAN	type	meaning
ana3pic.called	ANA3PIC_CALLED	integer	Non-zero if user_sel3pic was called
ana3pic.flag	ANA3PIC_FLAG	integer	Non-zero if event is selected
ana3pic.ghost	ANA3PIC_GHOST	integer	Non-zero if the first best vtx had a ghost track
ana3pic.ivertex	ANA3PIC_IVERTEX	integer	Index of the best vertex
ana3pic.ikabtrk	ANA3PIC_IKABTRK	integer	Index of the best KABES track
ana3pic.charge	ANA3PIC_CHARGE	integer	K charge
ana3pic.pk	ANA3PIC_PK	float	Reconstructed K momentum
ana3pic.pt2	ANA3PIC_PT2	float	Transverse momentum squared
ana3pic.mass	ANA3PIC_MASS	float	3pi mass
ana3pic.cog1x	ANA3PIC_COG1X	float	COG at DCH1 (x)
ana3pic.cog1y	ANA3PIC_COG1Y	float	COG at DCH1 (y)
ana3pic.cog4x	ANA3PIC_COG4X	float	COG at DCH4 (x)
ana3pic.cog4y	ANA3PIC_COG4Y	float	COG at DCH4 (y)
ana3pic.u	ANA3PIC_U	float	U (this is U0)
ana3pic.v	ANA3PIC_V	float	V (absolute value)
ana3pic.u1	ANA3PIC_U1	float	U from invariant mass of even pions
ana3pic.u2	ANA3PIC_U2	float	U from CM energy of odd pion

Table 4: Sturucture filled by the 3π neutral selection

C	FORTRAN	type	meaning
ana3pin.called	ANA3PIN_CALLED	integer	Non-zero if user_sel3pin was called
ana3pin.flag	ANA3PIN_FLAG	integer	Non-zero if event is selected
ana3pin.ghost	ANA3PIN_GHOST	integer	Non-zero if the first best vtx had a ghost track
ana3pin.ivertex	ANA3PIN_IVERTEX	integer	Index of the best vertex
ana3pin.ikabtrk	ANA3PIN_IKABTRK	integer	Index of the best KABES track
ana3pin.charge	ANA3PIN_CHARGE	integer	K charge
ana3pin.pk	ANA3PIN_PK	float	Reconstructed K momentum
ana3pin.pt2	ANA3PIN_PT2	float	Transverse momentum squared
ana3pin.mass	ANA3PIN_MASS	float	3pi mass
ana3pin.cog1x	ANA3PIN_COG1X	float	COG at DCH1 (x)
ana3pin.cog1y	ANA3PIN_COG1Y	float	COG at DCH1 (y)
ana3pin.cog4x	ANA3PIN_COG4X	float	COG at DCH4 (x)
ana3pin.cog4y	ANA3PIN_COG4Y	float	COG at DCH4 (y)
ana3pin.u	ANA3PIN_U	float	U
ana3pin.v	ANA3PIN_V	float	V (absolute value)

superCmpEvent structures and returns -1 for accepted events or 0 for rejected events.

Since COmPACT 7.1.1, the events selected by this filter are stored in the neutral scmp split.

6 Detailed description of super-compact and compact structure

This section aims at giving more details on the compact variables than the single line listed with the compact structure list. It is divided into three sub-sections: the first one 6.1 describes the super-compact structure; the second one 6.2 describes the compact structure; and the third one 6.3 describes routines that are used to fill the variables (either compact or super-compact or both).

6.1 Super-compact structure

The routine to fill the super-compact structure is *rlib/src/cmp2scmp.c*.

6.1.1 superCmpEvent structure

- struct rndm: for overlayed events.This structure contains information on the random used to overlay (overflows,....).
- nEvt: Event number (0 for overlayed events) .
- trigWord: Bit coded word for trigger, overlaying and filters.
  - bits 0-15.(trigger definition can vary year to year; available from na48 web information).
- timeStamp: The event timestamp in seconds since ....
- nTrigBef: coded word for the number of triggers in the previous 20ns (bits 0-7), 60ns (bits 8-15), 100 ns(bits 16-23) and 200ns (bits 24-32) (computed in *rlib/src/getTrigBef.c*).
- timeToPrev: time in timestamp units (25ns units) between the event and the previous trigger (bits 0-15) and the next to previous trigger (bits 16-31).
- SPSPhase: SPS frequency phase to $40MHz$ clock.
- MainsPhase: Phase inside the $50Hz$ cycle.
- QXdNdt: QX intensity between two L2 triggers in Hz.
- struct PUtslice : pattern-unit bits (cf sect. 6.1.9).
- LKRenergy: Total Energy in the Lkr calorimeter (Sum of cluster uncorrected energies).
- HACenergy: Total Energy in the HAC calorimeter

- struct proton: contains description of reconstructed protons from tagger (cf sect. 6.1.7).
- NprotLadder: Number of proton hits on ladder.
- DCHbz: track z position before magnet.
- DCHz: track z position after magnet.
- struct trak (cf sect. 6.1.5).
- struct SCvertex (cd sect. 6.1.6).
- struct FourVertex (cd sect. ??).
- struct muon (cd sect. 6.1.3).
- struct cluster (cf sect. 6.1.8).
- struct chamber (cf sect. 6.1.10).
- struct DCHcluster: This structure is not filled.
- NovrflwSim: Number of overflows symetrized for charged and neutrals.
- ovrflwSimBef: Time of closest symetrized overflow before the event.
- ovrflwSimAft: Time of closest symetrized overflow after the event.
- tsPrev: Timestamp of previous (L2) trigger.
- tsNext: Timestamp of next (L2) trigger.
- twPrev: Trigger word of previous (L2) trigger.
- twNext: Trigger word of next (L2) trigger.
- nHACcut: Number of HAC clusters with $E > 3\ GeV$ (*getnHACcut in rlib/src/cmp2scmp.c*).
- spareInt[2]: spare variables (integer).
- spareFloat[2]: spare variables (float).

6.1.2 DETstatus sub-structure

This structure holds information describing the sub-detector status (table 5).

6.1.3 muon sub-structure

Valid from version 6.0. From version 6.2 on, this structure is filled at run time according to a new muon-reconstruction routine. The information used to fill the Super-Compact muon structure is taken from the pmuon structure.

6.1.4 pmuon sub-structure

From version 6.2. This structure contains the photo-multiplier hits of the muon veto detector.

TAG	Not filled
AKS	Not filled
AKL	Not filled
DCH	MBOX simulation word (evt->DCHstatus)
HOD	Not filled
HAC	Not filled
LKR	decoding error flag (evt->LKRerrflag)
NHO	Not filled
MUV	decoding error flag (evt->MUVerrflag)
MBX	dead time information: bit 0 ON: MBX alive. (rlib/src/getMBOXdeadTime(bur,evt))
NTR	0: OK; 1: empty; 2: decoding error (rlib/anasrc/nuterr.c))
L3	16 low bits (0-15): Golden Neutral filter bits (see table 2 of Na48 Note 98-28) 16 high bits (16-31): Golden Charged filter bits (see table 3 of Na48 Note 98-28)
LV3Trig	16 low bits(0-15):evt->L3trigword[0] 16 high bits (16-31): evt->L3FilterDownScale
LV3TrigRare	L3 rare filter bits (evt->L3trigword[1])
LV3ABTrig	16 low bits(0-15):evt->L3ONLINEtrigword[0] 16 high bits (16-31): evt->L3Btrigword[0]
LV3ATrigRare	L3 online rare filter bits (evt->L3ONLINEtrigword[1])
LV3BTrigRare	L3 B rare filter bits (evt->L3Btrigword[1])
ChTrEff[10]	Extract from PU info (see table 6) for 10 time slices around the event

Table 5: sevt->DETstatus[0].xxx description

6.1.5 Trak sub-structure

6.1.6 SCvertex sub-structure

Introduced with version 6.0.

6.1.7 Proton sub-structure

Valid up to version 5.2.

6.1.8 Cluster sub-structure

6.1.9 PUtslice sub-structure

From version 6.1 the pattern-unit in Super-Compact is a copy of the pattern-unit in Compact (for the meaning of the bit see the Web address http://na48.web.cern.ch/NA48/private/Trigger/Overview.html). However, in order to save space certain channels have been zeroed and only timeslices 3-5 (counting from zero) are kept. In version 6.2 the following bits are kept:

```
Channel  4 all bits
Channel  5 all bits
Channel  6 all bits
Channel  7 only bit 5 (TON*NAKL)
Channel 12 all bits
Channel 13 all bits
Channel 14 only bits 16-20
Channel 15 all bits
```

6.1.10 Chamber sub-structure

6.1.11 Overflow sub-structure

6.1.12 Accidental sub-structure

6.1.13 Random for overlay sub-structure

6.2 Compact structure

6.2.1 Beam integrator sub-structure

The routine /lib/src/getBINT.c fills the variables for beam monitors integrator. The meaning of the variables is changed in 2001 as listed in table 6.2.1. In particular 8 more words have been added in order to get the samples for the channel with 200 nsec integration time. The 12 samples (10 bits each) are packed in 4 words, the least significant bits being always the earlier of the 3 samples.

6.2.2 Treatment of the charged vertex from version 6.2

From version 6.2 the charged vertex reconstruction is redone at the compact level. This has been introduced to treat vertices made from tracks with the same sign. For each pair of tracks the vertex reconstruction is attempted calling the routine FD_VTFIT3 in fuser_vertexntrk.F (see description below).

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C These routines calculate the vertex coordinates and some other
C parameters related to the tracks attached to this vertex.
C The number of tracks can be 2,3,4 or more ( To use it with at least
C 5 or more tracks , please contact the author )
C The following is a Kalman filter calculation which takes into
```

```
C account multiple scattering in the Kevlar window, He and chambers
C and also measurement errors in the chambers
C It is a generalisation of the code written for 2 tracks which is
C extensively used in the reconstruction of raw data events
C Main routine is FD_VTFIT3
C All calculated values appear in the VERTEXCORR_NTRK array
C Author : J.B.Cheze : November 1999
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
```

A vertex is saved if the closest distance of approach is less than 10 cm and the time difference between the two tracks is less than 30 ns. Positive and negative track indexes have the same meaning for the opposite sign vertexes: the track sign has to be deduced from $evt->track[].pq$ (TRACK_PQ).

6.3 Routines used to fill compact and super-compact variables

1) The routine /lib/src/getPuNut.c (from G. Fischer) to detect consecutive triggers; this routine is called when a super-compact event is built and bits are set into $sevt->CMPstatus$; this routine can also be called from compact (getPuNut(evt) for the C); it returns a bit coded integer:

```
bit 0: No PU data
bit 1: nut2pi0 two timeslices before the trigger
bit 2: nut2pi0 one timeslices before the trigger
bit 3: nut2pi0 at timeslices of the trigger
bit 4: nut2pi0 one timeslices after the trigger
bit 5: nut2pi0 two timeslices after the trigger
```

2)
3)
4)
5)
6)

7 HyperCOmPACT structures

This section provides descriptions for some of the variables in the HyperCOmPACT structures.

7.1 Charged hyperCmpEvent

7.1.1 Treatment of extra tracks/clusters

- Definition of extra cluster:
  $E > 1.5GeV$, separation @LKr R>15cm from each of the 3 tracks belonging to the good vertex. The first 4 extra clusters recorded into hevt->cluster.
  X : **hevt->cluster[i].x;**
  Y : **hevt->cluster[i].y;**
  time : **hevt->cluster[i].time;**
  energy : **hevt->cluster[i].energy;**
- Definition of extra track:
  A track not belonging to the good vertex. The first 4 extra tracks recorded into hevt->cluster structure.
  X @ LKr plane:**hevt->cluster[i].rmsx;**
  Y @ LKr plane: **hevt->cluster[i].rmsy;**
  time : **hevt->cluster[i].dDeadCell;**
  momentum :**hevt->cluster[i].dTrack;**

The numbers of extra tracks and clusters are coded into hevt->flag (like in the previous version of the routine).

7.1.2 hevt->flag (HEVT_FLAG)

The integer variable hevt-¿flag is used to store useful informations about the event. In order to save space, each bit has a different meaning, which is explained in table 7.1.2.

In order to extract the number the function `int get_n_of_tracks(hyperCmpEvent *hevt)` is provided (`GET_N_OF_TRACKS(QE)` in FORTRAN).

7.1.3 hevt->PUpack[3], (HEVT_PUPACK)

The first column is bit number in HEVT_PUPACK, the second is standard channel and bit numbers. HEVT_PUPACK should contain 3 words, corresponding to contents of timeslice 3,4,5. (Exception from this rule: channel N9, there 2 timeslices are packed into single bit).

NB: C indexing used everywhere! should add +1 for fortran indexing!

/* Packed Bit — PU Channel.Bit */

* 0 — 5.0 *

* 1 — 5.1 *

* 2 — 5.2 *

* 3 — 5.3 *

* 4 — 5.4 *

* 5 — 5.5 *

* 6 — 5.6 *
* 7 — 5.7 *
* 8 — 5.8 *
* 9 — 5.9 *
* 10 — 5.10 *
* 11 — 5.12 *
* 12 — 5.13 *
* 13 — 9.10 *
* 14 — 9.11 *
* 15 — 9.12 *
* 16 — 9.17 *
* 17 — 9.18 *
* 18 — 12.1 *
* 19 — 12.3 *
* 20 — 12.4 *
* 21 — 12.5 *
* 22 — 12.13 *
* 23 — 12.18 *
* 24 — 13.1 *
* 25 — 13.2 *
* 26 — 14.0 *
* 27 — 14.1 *
* 28 — 14.16 *
* 29 — 14.17 *
* 30 — 14.18 *
* 31 — 14.20 *

7.2 Neutral hyperCmpEvent

8 Useful informations when running compact and/or super-compact

This sections describes a few informations usefull when running analysis jobs.

8.1 Input/Output flags

In figure 1, it is shown how the output flags -so/-ho… do depending on the type of input the reader gets.

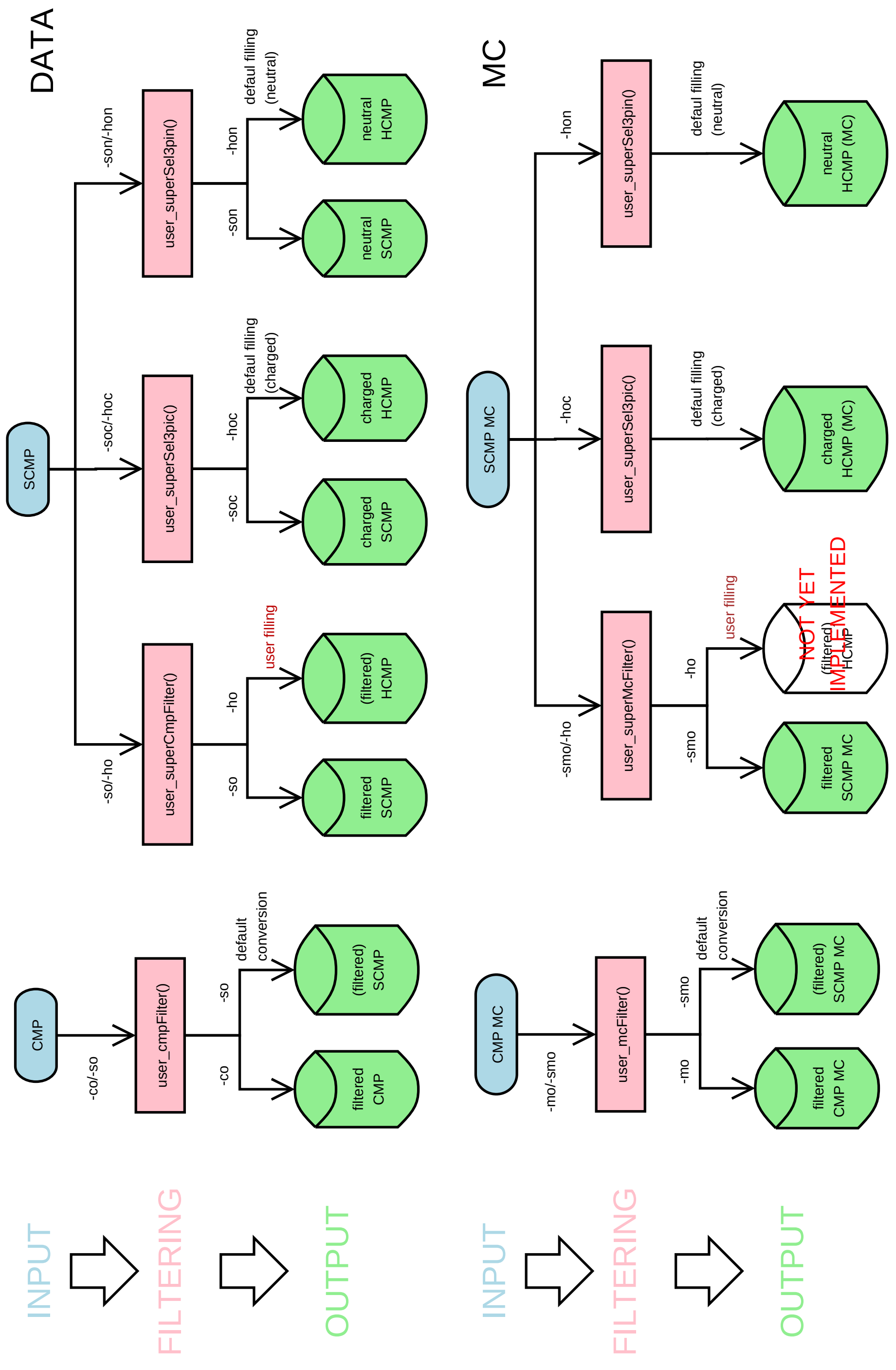

Figure 1: Scheme of the behaviours triggered by the output flags in the COmPACT reader.

8.2

The busrt database (also sometimes called compact database) is automatically accessed from the compact reader and 2 structures are filled: rdb (for run database) et bdb (for burst database); the variable names and descriptions are available at the end of this manual. In the four routines, user_burst.c, user_superBurst.c, fuser_burst.F and fuser_superBurst.F, it is shown how to access the data in the database.

The result of the routine in prepration to decide whether a burst is good or not, will fill the variable bur->dberr (BUR_DBERR) or sbur->dberr (SBUR_DBERR), depending whether the event is of compact or super-compact format.

It is possible to run compact without accessing the compact database (this is sometimes usefull when reading already filtered data or when the database is not yet ready):

```
compact -l input.list -ndb
```

(ndb stands for NoDataBase).

8.3 constants.h file

A set of usefull constants (internal to compact or physical constants like the kaon mass) is available from file lib/inc/constants.h.

8.4 Decoding errors words

The word DCHerrflagPDS: $evt- > DCHerrflagPDS$ (EVT_DCHERRFLAGPDS) copy of Q(PDCH+8), the summary of DCH decoding errors (cf Table 8.4).

8.5 Which corrections were applied to this event

The word FlagCorr: $evt- > FlagCorr$ (EVT_FLAGCORR) is bit coded and indicates which analysis routines were called (cf Table 10). It is copied to super-compact ($sevt- > FlagCorr$ (SEVT_FLAGCORR)).

This information allows to find out *a posteriori* which compact corrections were applied to a given event.

8.6 Event lists

There are 6 *event lists* which contains information for ALL the events in the rawdata files:

8.7 Flags to select analysis routines

- Super Compact has been modified; the structure has changed (and is not backward compatible.
- The analysis routines have been moved to the rlib library for ease on maintenance; i.e. the library can be modified without any intervention from the user; this should make updates easier.
- Control on analysis routines: the variables $bur- > CallAnaRoutine$ (BUR_CALLANAROUTINE) allows the user to decide how to call the analysis routines; this has to be done from the user_burst routine. The routine (f)user_burst.example.c(F) shows how to setup the flags. *The routines badburst, CmpLkrDead (for 1997), CmpTimeOffset, epsy and std_ep, std_rcog, std_ptsq are called antway.*
  - $bur- > CallAnaRoutine = 1$ (the default in compact): let COmPACT decides.
  - $bur- > CallAnaRoutine = 0$: NO analysis routine called.
  - $bur- > CallAnaRoutine = -1$: the user selects the analysis routines by setting the following variables to a non zero value:

```
bur->CallAnaRoutine=-1;  /* the user wants to choose the routines */
bur->tocall.selcharged=1; /* to call selcharged */
bur->tocall.sel2pi0=1;    /* to call sel2pi0 */
bur->tocall.sel3pi0=1;    /* to call sel3pi0 - only if no 2pi0 cand
bur->tocall.bluefield=1;  /* to call correction for Bfield in blue t
bur->tocall.lkrpedcor=1;  /* to call lkr pedestal correction (1997)
bur->tocall.hodotime=1;   /* to call hodotime */
bur->tocall.nhodtime=1;   /* to call nhodtime */
bur->tocall.lkrtime=1;    /* to call lkrtime only if there is a 2pi
bur->tocall.tagtime=0;    /* to call tagtime */
bur->tocall.muon_rec=0;   /* to call "new" muon reconstruction (199
bur->tocall.aksflag=1;    /* to call flagging of good AKS events */
bur->tocall.lkrposcor=1;  /* to call lkr position correction (1997)
bur->tocall.lkrsharing=1; /* to call lkr energy sharing between clu
bur->tocall.muon_rec=1;   /* to call "new" muon reconstruction (199
bur->tocall.muon_reject=1;/* to call muon Rejection routines (1997)
bur->tocall.geomcor=1;    /* to call geometry correction routine (1
bur->tocall.newcharged=1; /* to call the new charged analysis
routine - NOT YET */
```

or

```
C*the user wants to choose the routines
      BUR_CALLANAROUTINE=-1
C*to call selcharged
      BUR_TOCALL_SELCHARGED=1
C*to call sel2pi0
      BUR_TOCALL_SEL2PI0=1
```

```
C*to call sel3pi0 - only if no 2pi0 cand.
      BUR_TOCALL_SEL3PI0=1
C*to call correction for Bfield in blue tube (1997)
      BUR_TOCALL_BLUEFIELD=1
C*to call lkr pedestal correction (1997)
      BUR_TOCALL_LKRPEDCOR=1
C*to call hodotime
      BUR_TOCALL_HODOTIME=1
C*to call nhodtime
      BUR_TOCALL_NHODTIME=1
C*to call lkrtime only if there is a 2pi0 cand.
      BUR_TOCALL_LKRTIME=1
C*to call tagtime
      BUR_TOCALL_TAGTIME=0
C*to call "new" muon reconstruction (1997)
      BUR_TOCALL_MUON_REC=0
C*to call flagging of good AKS events
      BUR_TOCALL_AKSFLAG=1
C* to call lkr position correction (1997)
      BUR_TOCALL_LKRPOSCOR=1
C* to call lkr energy sharing between clusters
      BUR_TOCALL_LKRSHARING=1
C*to call "new" muon reconstruction (1997)
      BUR_TOCALL_MUON_REC=1
C*to call muon rejection (1997)
      BUR_TOCALL_MUON_REJECT=1
C*to call geometry correction routine
      BUR_TOCALL_GEOMCOR=1
C*to call NEW charged event selection routine (NOT YET)
      BUR_TOCALL_NEWCHARGED=1
```

9 Super-compact scaling factors

Important: SuperCOmPACT reduces data volume and uncompression time by using integer variables for all i/o to and from disk files. This is implemented by multiplying floats by 10^n then casting as integers before copying the structure to disk. When reading the file the integer variables are converted to floats and then divided by the same scaling factor. This naturally results in truncation of a certain number of decimal places. It is intended that only non-significant figures are removed.

A list of the scaling factors can be found in the file $/afs/cern.ch/na48/offline/compact/com$ $6.0/lib/compact.x$. The list is also given below with explanations of where these factors are applied:

Here are given the lists of labels for the TS scalers (TS and L2TS).

A 1997 configuration

```
C
C *** L2TS FILE
C
      INTEGER L2TS_NL
      PARAMETER (L2TS_NL=17)
      CHARACTER*32  L2TS_LAB(L2TS_NL)
      DATA L2TS_LAB/'TS_PI+PI-','TS_L1_PI+PI-','TS_L1PP_WDOG',
     +    'TS_QOR', 'TS_PI0PI0', 'TS_NHOD', 'TS_LKRMBIAS',
     +    'TS_EPS_MBIAS', 'TS_PI+PI-PI0', 'TS_MUMUG',
     +    'TS_DALITZ', 'TS_SUSY', 'TS_RanKL/KS',
     +    'TS_L1ON_CNT', 'TS_XLOSTTR', 'TS_FLOSTTR',
     +    'TS_2PI0_NOPK'/


      INTEGER TS_NL
      PARAMETER (TS_NL=93)
      CHARACTER*32  TS_LAB(TS_NL)
      DATA TS_LAB/'TS_Q2*1mu*!AKL', 'TS_Q2*!1mu*!AKL',
     + 'TS_QX*!1mu*!AKL', 'TS_QX*!1mu', 'TS_QX/2',
     + 'TS_QX', 'TS_QX_FTIME_1', 'TS_QX_FTIME_0',
     + 'TS_QX_FT0_to_MB', 'TS_QX_FT1_to_MB',
     + 'TS_MBOX_DEBREQ', 'TS_L1_COD1_to_MB',
     + 'TS_L1_COD2_to_MB', 'TS_L1_COD3_to_MB',
     + 'TS_L1_STRB*L1ON', 'TS_L1_STRB_to_MB',
     + 'TS_<=1TRACK_DCH1', 'TS_2/3TRACKS_DC1',
     + 'TS_>=4TRACKS_DC1', 'TS_OVFLOW_DCH1', 'TS_TOTOR_AKL',
     + 'TS_QX/128', 'TS_Q2*2mu*!AKL', 'TS_TON*!1mu*!AKL',
     + 'TS_TRIG_NO_XOFF', 'TS_TRIG_NOX_WRIT', 'TS_LASER',
     + 'TS_LKR_CAL_(NZS)', 'TS_LKR_CAL_(ZS)', 'TS_HAC_PULSER',
     + 'TS_PMB_PULSER', 'TS_RANDOM_KL', 'TS_RANDOM_KS',
     + 'TS_S2HVspark', 'TS_FULL_TSlost', 'TS_XOFF_TSlost',
     + 'TS_L1_LASER', 'TS_L1_LKRCAL_NZS', 'TS_L1_LKRCAL_ZS',
     + 'TS_L1_HAC_PULS', 'TS_L1_PMB_PULS', 'TS_L1_RAND_KL',
     + 'TS_L1_RAND_KS', 'TS_L1_L1ON_TRANS', 'TS_L1_Q2*1u*!AK',
     + 'TS_L1_Q2*!1u*!AK', 'TS_L1_QX*!1u*!AK', 'TS_L1_QX*!1mu',
     + 'TS_L1_QX/2', 'TS_L1_QX', 'TS_L1_QX_FTIME_1',
     + 'TS_L1_QX_FTIME_0', 'TS_L1_OVF_DCH1', 'TS_L1_>=4TR_DCH1',
     + 'TS_L1_2/3TR_DCH1', 'TS_L1_<=1TR_DCH1', 'TS_L1_TON*!1u*!A',
     + 'TS_L1_Q2*2u*!AK', 'TS_L1_QX/128', 'TS_L1_AKL', 'TS_NT_2PI0',
     + 'TS_NT_LKR_MBIAS', 'TS_NT_2PI0_NOPK', 'TS_NT_ETOT',
     + 'TS_NT_ACCID', 'TS_NT_MUMUGAM', 'TS_NT_MUMUEE',
     + 'TS_NT_DALITZ', 'TS_NT_SUSY_2G', 'TS_NT_FTIME_0',
     + 'TS_NT_FTIME_1', 'TS_NT_TIMPEAK', 'TS_NT_SUSY_3P',
     + 'TS_CPD_PATTERN', 'TS_Q1*!1mu', 'TS_Q2', 'TS_QOR',
     + 'TS_1MU', 'TS_TON*!1MU', 'TS_Q1*1MU', 'TS_MB_OVFLOW12',
     + 'TS_MB_OVFLOW4', 'TS_MB_TOO_CPLX', 'TS_MB_INTIME',
     + 'TS_MB_ZOK_RARE', 'TS_MB_PI+PI-PI0 ', 'TS_MB_PI+PI-_OK ',
     + 'TS_MB_FATAL', 'TS_MB_FORCE_READ', 'TS_MB_ZOK', 'TS_MB_MASS_OK',
     + 'TS_MB_STRB_TO_TS','TS_NT_ETOT_BIAS'/
```

B 1998 configuration

```
C
C *** L2TS FILE
C
      INTEGER L2TS_NL
      PARAMETER (L2TS_NL=16)
      CHARACTER*32  L2TS_LAB(L2TS_NL)
      DATA L2TS_LAB/'TS_PI+PI-','TS_L1_PI+PI-','TS_L1PP_WDOG',
     +    'TS_CHAR_MBias', 'TS_PI0PI0', 'TS_NHODO','TS_LKRMBIAS',
     +    'TS_3PI0', 'TS_PI+PI-PI0', 'TS_MUMUG',
     +    'TS_DALITZ', 'TS_SUSY', 'TS_RanKL/KS',
     +    'TS_L1ONCNT', 'TS_XLOSTTR', 'TS_FLOSTTR'/


      INTEGER TS_NL
      PARAMETER (TS_NL=92)
      CHARACTER*32  TS_LAB(TS_NL)
      DATA TS_LAB/'TS_Q2*1mu*!AKL', 'TS_Q2*!1mu*!AKL',
     + 'TS_QX*!1mu*!AKL', 'TS_QX*!1mu', 'TS_TOTOR_AKL',
     + 'TS_QX/D', 'TS_QX_FTIME_1', 'TS_QX_FTIME_0',
     + 'TS_QX_FT0_to_MB', 'TS_QX_FT1_to_MB',
     + 'TS_MBOX_DEBREQ', 'TS_L1_COD1_to_MB',
     + 'TS_L1_COD2_to_MB', 'TS_L1_COD3_to_MB',
     + 'TS_L1_STRB*L1ON', 'TS_L1_STRB_to_MB',
     + 'TS_GE1TRACK_DCH1', 'TS_GE2TRACKS_DC1',
     + 'TS_GE4TRACKS_DC1', 'TS_OVFLOW_DCH1',
     + 'TS_QX', 'TS_Q2*2mu*!AKL','TS_TON',
     + 'TS_TRIG_NO_XOFF', 'TS_TRIG_NOX_WRIT','TS_LASER',
     + 'TS_LKR_CAL_(NZS)', 'TS_LKR_CAL_(ZS)','TS_HAC_PULSER',
     + 'TS_PMB_PULSER', 'TS_RANDOM_KL','TS_RANDOM_KS',
     + 'TS_S2HVspark', 'TS_FULL_TSlost','TS_XOFF_TSlost',
     + 'TS_L1_LASER', 'TS_L1_LKRCAL_NZS','TS_L1_LKRCAL_ZS',
     + 'TS_L1_HAC_PULS', 'TS_L1_PMB_PULS','TS_L1_RAND_KL',
     + 'TS_L1_RAND_KS', 'TS_L1_L1ON_TRANS','TS_L1_Q2*1u*!AK',
     + 'TS_L1_Q2*!1u*!AK', 'TS_L1_QX*!1u*!AK','TS_L1_QX*!1u',
     + 'TS_L1_QX/D', 'TS_L1_QX', 'TS_L1_QX_FTIME_1',
     + 'TS_L1_QX_FTIME_0', 'TS_L1_OVF_DCH1','TS_L1_GE4TR_DCH1',
     + 'TS_L1_GE2TR_DCH1', 'TS_L1_GE1TR_DCH1','TS_L1_TON',
     + 'TS_L1_Q2*2u*!AK', 'TS_L1_TOTOR_AKL','TS_NT_2PI0',
     + 'TS_NT_LKR_MBIAS', 'TS_NT_3PI0','TS_NT_ETOT',
     + 'TS_NT_ACCID', 'TS_NT_MUMUGAM', 'TS_NT_MUMUEE',
     + 'TS_NT_DALITZ', 'TS_NT_SUSY_2G','TS_NT_FTIME_O',
     + 'TS_NT_FTIME_1', 'TS_NT_TIMPEAK','TS_NT_SUSY_3P',
     + 'TS_CPD_PATTERN', 'TS_Q1*!1mu', 'TS_Q2','TS_QOR',
     + 'TS_1MU', 'TS_TON*!1MU', 'TS_Q1*1MU','TS_MB_OVFLOW12',
     + 'TS_MB_OVFLOW4', 'TS_MB_TOO_CPLX','TS_MB_INTIME',
     + 'TS_MB_ZOK_RARE', 'TS_MB_PI+PI-PI0 ','TS_MB_PI+PI-_OK ',
     + 'TS_MB_FATAL', 'TS_MB_FORCE_READ', 'TS_MB_ZOK','TS_MB_MASS_OK',
     + 'TS_MB_STRB_TO_TS','TS_MB_4TRACKS','TS_MB_mbias'/


-----------------------------------------------------
```

C 1999 configuration

```
C
C *** L2TS FILE
C
      INTEGER L2TS_NL
      PARAMETER (L2TS_NL=17)
      CHARACTER*32  L2TS_LAB(L2TS_NL)
      DATA L2TS_LAB/'TS_PI+PI-','TS_L1_PI+PI-','TS_L1PP_WDOG',
     +    'TS_CHAR_MBias','TS_PI0PI0','TS_NHODO','TS_LKR_MBIAS',
     +    'TS_3PI0','TS_KE4','TS_MUMUG','TS_DALITZ','TS_MB_4TRACK',
     +    'TS_RanKL/KS','TS_L1CNT','TS_L1ONCNT',
     +    'TS_XLOSTTR','TS_FLOSTTR'/


      INTEGER TS_NL
      PARAMETER (TS_NL=93)
      CHARACTER*32  TS_LAB(TS_NL)
      DATA TS_LAB/'TS_Q2*1mu*!AKL','TS_Q2*!1mu*!AK/D',
     +  'TS_QX*!1mu*!AKL','TS_QX*!1mu','TS_TOTAND_AKL','TS_QX/D',
     +  'TS_Q1*2mu*!AKL','TS_TON','TS_QX_FT0_to_MB',
     +  'TS_QX_FT1_to_MB','TS_MBOX_DEBREQ','TS_L1_COD1_to_MB',
     +  'TS_L1_COD2_to_MB','TS_L1_COD3_to_MB','TS_L1_STRB*L1ON',
     +  'TS_L1_STRB_to_MB','TS_GE1TRACK_DCH1','TS_GE2TRACKS_DC1',
     +  'TS_GE4TRACKS_DC1','TS_OVFLOW_DCH1','TS_DEBU_pulser',
     +  'TS_QX','TS_QX_FTIME_0','TS_QX_FTIME_1',
     +  'TS_TRIG_NO_XOFF','TS_TRIG_NOX_WRIT','TS_LASER',
     +  'TS_LKR_CAL_(NZS)','TS_LKR_CAL_(ZS)','TS_HAC_PULSER',
     +  'TS_PMB_PULSER','TS_RANDOM_KL','TS_RANDOM_KS',
     +  'TS_S2HVspark','TS_FULL_TSlost','TS_XOFF_TSlost',
     +  'TS_L1_LASER','TS_L1_LKRCAL_NZS','TS_L1_LKRCAL_ZS',
     +  'TS_L1_HAC_PULS','TS_L1_PMB_PULS','TS_L1_RAND_KL',
     +  'TS_L1_RAND_KS','TS_L1_L1ON_TRANS','TS_L1_Q2*1u*!AK',
     +  'TS_L1_Q2*!uAKL/D','TS_L1_QX*!1u*!AK','TS_L1_QX*!1u',
     +  'TS_L1_TOTAND_AKL','TS_L1_QX/D','TS_L1_Q1*2u*!AK',
     +  'TS_L1_TON','TS_L1_OVF_DCH1','TS_L1_GE4TR_DCH1',
     +  'TS_L1_GE2TR_DCH1','TS_L1_GE1TR_DCH1','TS_L1_DEBU_puls',
     +  'TS_L1_QX','TS_L1_QX_FTIME_1','TS_L1_QX_FTIME_0',
     +  'TS_NT_2PI0','TS_NT_LKR_MBIAS','TS_NT_3PI0','TS_NT_ETOT',
     +  'TS_NT_ACCID','TS_NT_MUMUGAM','TS_NT_MUMUGG',
     +  'TS_NT_DALITZ','TS_NT_KE4NT','TS_NT_FTIME_0',
     +  'TS_NT_FTIME_1','TS_NT_TIMPEAK','TS_NT_PI0nunu',
     +  'TS_CPD_PATTERN','TS_Q1*!1mu','TS_Q2','TS_QOR','TS_1MU',
     +  'TS_TON*!1MU','TS_Q1*1MU','TS_MB_OVFLOW12',
     +  'TS_MB_OVFLOW4','TS_MB_TOO_CPLX','TS_MB_INTIME',
     +  'TS_MB_4TRACKS','TS_MB_ZOK_RARE','TS_MB_PI+PI-PI0',
     +  'TS_MB_PI+PI-_OK','TS_MB_FATAL','TS_MB_FORCE_READ',
     +  'TS_MB_ZOK','TS_MB_MASS_OK','TS_MB_STRB_TO_TS'/


--------------------------------------------------
```

D 2001 configuration

```
C
C *** L2TS FILE
C
      INTEGER L2TS_NL
      PARAMETER (L2TS_NL=18)
      CHARACTER*32  L2TS_LAB(L2TS_NL)
      DATA L2TS_LAB/'TS_PI+PI-', 'TS_L1_PI+PI-', 'TS_L1PP_WDOG',
     +     'TS_RADHYP', 'TS_SEMHYP', 'TS_NHODO', 'TS_DELTAS2',
     +     'TS_2PI0', 'TS_3PI0', 'TS_MUMU', 'TS_DALITZ', 'TS_MB_4TRACK',
     +     'TS_MULTI', 'TS_Random', 'TS_L1CNT', 'TS_L1ONCNT',
     +     'TS_XLOSTTR', 'TS_FLOSTTR'/


      INTEGER TS_NL
      PARAMETER (TS_NL=93)
      CHARACTER*32  TS_LAB(TS_NL)
      DATA TS_LAB/'TS_Q2*1mu*!AKL','TS_Q2*!1mu*!AK/D','TS_Q1*!1mu*!AKL',
     +     'TS_QX', 'TS_TOTAND_AKL', 'TS_Q1/D', 'TS_Q1*2mu', 'TS_TON',
     +     'TS_Q1_FT0_to_MB', 'TS_Q1_FT1_to_MB', 'TS_MBOX_DEBREQ',
     +     'TS_L1_COD1_to_MB', 'TS_L1_COD2_to_MB', 'TS_L1_COD3_to_MB',
     +     'TS_L1_STRB*L1ON', 'TS_L1_STRB_to_MB', 'TS_GE1TRACK_DCH1',
     +     'TS_GE2TRACKS_DC1', 'TS_GE4TRACKS_DC1', 'TS_OVFLOW_DCH1',
     +     'TS_DEBU_pulser', 'TS_Q1', 'TS_Q1_FTIME_0', 'TS_Q1_FTIME_1',
     +     'TS_TRIG_NO_XOFF', 'TS_TRIG_NOX_WRIT', 'TS_LASER',
     +     'TS_LKR_CAL_(NZS)', 'TS_LKR_CAL_(ZS)', 'TS_HAC_PULSER',
     +     'TS_PMB_PULSER', 'TS_RANDOM_KL', 'TS_RANDOM_KS',
     +     'TS_S2HVspark', 'TS_FULL_TSlost', 'TS_XOFF_TSlost',
     +     'TS_L1_LASER', 'TS_L1_LKRCAL_NZS', 'TS_L1_LKRCAL_ZS',
     +     'TS_L1_HAC_PULS', 'TS_L1_PMB_PULS', 'TS_L1_RAND_KL',
     +     'TS_L1_RAND_KS', 'TS_L1_L1ON_TRANS', 'TS_L1_Q2*1u*!AK',
     +     'TS_L1_Q2*!uAKL/D', 'TS_L1_Q1*!1u*!AK', 'TS_L1_QX',
     +     'TS_L1_TOTAND_AKL', 'TS_L1_Q1/D', 'TS_L1_Q1*2u',
     +     'TS_L1_TON', 'TS_L1_OVF_DCH1', 'TS_L1_GE4TR_DCH1',
     +     'TS_L1_GE2TR_DCH1', 'TS_L1_GE1TR_DCH1', 'TS_L1_DEBU_puls',
     +     'TS_L1_Q1', 'TS_L1_Q1_FTIME_1', 'TS_L1_Q1_FTIME_0',
     +     'TS_NT_2PI0', 'TS_NT_LKR_MBIAS', 'TS_NT_3PI0', 'TS_NT_ETOT',
     +     'TS_NT_ACCID', 'TS_NT_EHACLOW', 'TS_NT_MUMUGG',
     +     'TS_NT_DALITZ', 'TS_NT_3PI0_PEAK', 'TS_NT_FTIME_0',
     +     'TS_NT_FTIME_1', 'TS_NT_TIMPEAK', 'TS_NT_PI0nunu',
     +     'TS_CPD_PATTERN', 'TS_Q1*!1mu', 'TS_Q2', 'TS_QOR', 'TS_1MU',
     +     'TS_TON*!1MU', 'TS_Q1*1MU', 'TS_MB_MLAMBDA',
     +     'TS_MB_M>LAMBDA', 'TS_MB_DCH1_DIST', 'TS_MB_INTIME',
     +     'TS_MB_4TRACKS', 'TS_MB_ZOK_RARE', 'TS_MB_P_RATIO',
     +     'TS_MB_PI+PI-_OK', 'TS_MB_FATAL', 'TS_MB_PT', 'TS_MB_ZOK',
     +     'TS_MB_MASS_OK', 'TS_MB_STRB_TO_TS'/

--------------------------------------------------
```

E 2002 configuration

```
C
C *** L2TS FILE
C


      INTEGER L2TS_NL
      PARAMETER (L2TS_NL=18)
      CHARACTER*32  L2TS_LAB(L2TS_NL)
      DATA L2TS_LAB/'TS_PI+PI-', 'TS_L1_PI+PI-', 'TS_L1PP_WDOG',
     +     'TS_RADHYP', 'TS_SEMHYP', 'TS_NHODO', 'TS_DELTAS2',
     +     'TS_2PI0', 'TS_3PI0', 'TS_MUMU', 'TS_DALITZ', 'TS_MB_4TRACK',
     +     'TS_MULTI', 'TS_Random', 'TS_L1CNT', 'TS_L1ONCNT',
     +     'TS_XLOSTTR', 'TS_FLOSTTR'/



      INTEGER TS_NL
      PARAMETER (TS_NL=93)
      CHARACTER*32  TS_LAB(TS_NL)
      DATA TS_LAB/'TS_Q2*1mu*!AKL','TS_Q2*!1mu*!AK/D','TS_Q1*!1mu*!AKL',
     +     'TS_QX', 'TS_TOTAND_AKL', 'TS_Q1/D', 'TS_Q1*2mu', 'TS_TON',
     +     'TS_Q1_FT0_to_MB', 'TS_Q1_FT1_to_MB', 'TS_MBOX_DEBREQ',
     +     'TS_L1_COD1_to_MB', 'TS_L1_COD2_to_MB', 'TS_L1_COD3_to_MB',
     +     'TS_L1_STRB*L1ON', 'TS_L1_STRB_to_MB', 'TS_GE1TRACK_DCH1',
     +     'TS_GE2TRACKS_DC1', 'TS_GE4TRACKS_DC1', 'TS_OVFLOW_DCH1',
     +     'TS_DEBU_pulser', 'TS_Q1', 'TS_Q1_FTIME_0', 'TS_Q1_FTIME_1',
     +     'TS_TRIG_NO_XOFF', 'TS_TRIG_NOX_WRIT', 'TS_LASER',
     +     'TS_LKR_CAL_(NZS)', 'TS_LKR_CAL_(ZS)', 'TS_HAC_PULSER',
     +     'TS_PMB_PULSER', 'TS_RANDOM_KL', 'TS_RANDOM_KS',
     +     'TS_S2HVspark', 'TS_FULL_TSlost', 'TS_XOFF_TSlost',
     +     'TS_L1_LASER', 'TS_L1_LKRCAL_NZS', 'TS_L1_LKRCAL_ZS',
     +     'TS_L1_HAC_PULS', 'TS_L1_PMB_PULS', 'TS_L1_RAND_KL',
     +     'TS_L1_RAND_KS', 'TS_L1_L1ON_TRANS', 'TS_L1_Q2*1u*!AK',
     +     'TS_L1_Q2*!uAKL/D', 'TS_L1_Q1*!1u*!AK', 'TS_L1_QX',
     +     'TS_L1_TOTAND_AKL', 'TS_L1_Q1/D', 'TS_L1_Q1*2u',
     +     'TS_L1_TON', 'TS_L1_OVF_DCH1', 'TS_L1_GE4TR_DCH1',
     +     'TS_L1_GE2TR_DCH1', 'TS_L1_GE1TR_DCH1', 'TS_L1_DEBU_puls',
     +     'TS_L1_Q1', 'TS_L1_Q1_FTIME_1', 'TS_L1_Q1_FTIME_0',
     +     'TS_NT_2PI0', 'TS_NT_LKR_MBIAS', 'TS_NT_3PI0', 'TS_NT_ETOT',
     +     'TS_NT_ACCID', 'TS_NT_EHACLOW', 'TS_NT_MUMUGG',
     +     'TS_NT_DALITZ', 'TS_NT_3PI0_PEAK', 'TS_NT_FTIME_0',
     +     'TS_NT_FTIME_1', 'TS_NT_TIMPEAK', 'TS_NT_PI0nunu',
     +     'TS_CPD_PATTERN', 'TS_Q1*!1mu', 'TS_Q2', 'TS_QOR', 'TS_1MU',
     +     'TS_TON*!1MU', 'TS_Q1*1MU', 'TS_MB_MLAMBDA',
     +     'TS_MB_M>LAMBDA', 'TS_MB_DCH1_DIST', 'TS_MB_INTIME',
     +     'TS_MB_4TRACKS', 'TS_MB_ZOK_RARE', 'TS_MB_P_RATIO',
     +     'TS_MB_PI+PI-_OK', 'TS_MB_FATAL', 'TS_MB_PT', 'TS_MB_ZOK',
     +     'TS_MB_MASS_OK', 'TS_MB_STRB_TO_TS'/

--------------------------------------------------
```

F 2003 configuration

L1TS Scalers

id	label	id	label	id	label
1	TS_CH_KE2-PRE/D3	32	TS_RANDOM_KL	63	TS_NT_PEAK
2	TS_Q1/D1	33	TS_RANDOM_KS	64	TS_NT_Etot
3	TS_CH_KU2-PRE/D4	34	TS_S2HVspark	65	TS_NT_Etot_short
4	TS_Q2*!AKL	35	TS_FULL_TSlost	66	TS_NT_KMU3
5	TS_Q2*!AKL_d	36	TS_XOFF_TSlost	67	TS_NT_KMU2
6	TS_Q1/D2	37	TS_L1_LASER	68	TS_NT_KE2-PRE
7	TS_CH_KE2-PRE	38	TS_L1_LKRCAL_NZS	69	TS_NT_NOPEAK
8	TS_CH_KMU2-PRE	39	TS_L1_LKRCAL_ZS	70	TS_NT_FTIME_0
9	TS_Q1+Q2_FT0_MB	40	TS_L1_HAC_PULS	71	TS_NT_FTIME_1
10	TS_Q1+Q2_FT1_MB	41	TS_L1_PMB_PULS	72	TS_NT_TIMEPEAK
11	TS_MBOX_DEBREQ	42	TS_L1_RAND_KL	73	TS_NT_Zvtx
12	TS_L1_COD1_to_MB	43	TS_L1_RAND_KS	74	TS_Q1
13	TS_L1_COD2_to_MB	44	TS_L1_L1ON_TRANS	75	TS_Q1*!Q2
14	TS_L1_COD3_to_MB	45	TS_L1_KE2-PRE/D3	76	TS_Q1*QAND
15	TS_L1_STRB*L1ON	46	TS_L1_Q1/D1	77	TS_1MU
16	TS_L1_STRB_to_MB	47	TS_L1_KMU2-PR/D4	78	TS_TON
17	TS_1trk_loose	48	TS_L1_Q2*!AKL	79	TS_Q2
18	TS_2-3trk_loose	49	TS_L1_Q2*!AKL_d	80	TS_MB_OVFLOW1-2
19	TS_HI_multipl.	50	TS_L1_Q1/D2	81	TS_MB_OVFLOW4
20	TS_OVFLOW_DCH1	51	TS_L1_KE2-PRE	82	TS_MB_TOO_CPLX
21	TS_Random_L1	52	TS_L1_KMU2-PRE	83	TS_MB_IN_TIME
22	TS_Q1+Q2	53	TS_L1_OVF_DCH1	84	TS_MB_MBX-2VTX
23	TS_Q1+Q2_FTIME_0	54	TS_L1_HI_mult	85	TS_MB_1TRK-3
24	TS_Q1+Q2_FTIME_1	55	TS_L1_2-3trk_loo	86	TS_MB_Zok_rare
25	TS_TRIG_NO_XOFF	56	TS_L1_1trk_loose	87	TS_MB_MBX-1VTX
26	TS_TRIG_NOX_WRIT	57	TS_L1_Random_L1	88	TS_MB_1TRK-2
27	TS_LASER	58	TS_L1_Q1+Q2	89	TS_MB_FATAL
28	TS_LKR_CAL_(NZS)	59	TS_L1_Q1+Q2_FT_1	90	TS_MB_FORCE_R/O
29	TS_LKR_CAL_(ZS)	60	TS_L1_Q1+Q2_FT_0	91	TS_MB_ZFAKE
30	TS_HAC_PULSER	61	TS_NT_COG	92	TS_MB_1TRK-P
31	TS_PMB_PULSER	62	TS_NT_LKR_MBIAS	93	TS_MB_STRB_TO_TS

L2TS Scalers

id	label	id	label	id	label
1	TS_MB-2VTX	7	TS_MB-1TRK-2	13	TS_2BODY-PRE
2	TS_MB-1VTX	8	TS_MB-ZFAKE	14	TS_Random
3	TS_L1PP_WDOG	9	TS_C-MBIAS	15	TS_L1CNT
4	TS_C-PRE	10	TS_N-MBIAS	16	TS_L1ONCNT
5	TS_MB-1TRK-P	11	TS_N-PRE	17	TS_XLOSTTR
6	TS_MB-1TRK-3	12	TS_KMU3-PRE	18	TS_FLOSTTR

NOTE: in C you have to use (id-1) because the arrays are 0-based.

G Common blocks description

Here is a description, as complete as possible, prepared by G. Bocquet, of the common blocks from which data were transfered from the raw0xx program to compact. This descr

```
*************************************************************************
* COMMON /DCHSAC_CMPBLK/     rsa car file
*************************************************************************
      INTEGER CMAXHIT           ! maximum number of hits
      INTEGER MAXTRACK          ! maximum number of tracks
      INTEGER NDCH              ! number of chambers
      PARAMETER (CMAXHIT=64)
      PARAMETER (MAXTRACK=40)
      PARAMETER (NDCH=4)
      INTEGER NHITS             ! nb of hits/view(1->4 ch1,5->8 ch2,.)
      INTEGER NHT               ! nb of hits on track
      INTEGER JPLANE            ! planes of track hits
      INTEGER JWIRE             ! wires of track hits
      REAL JTIME_C              ! drift time of track hits
      REAL JDIST_C              ! drift distance of track hits
      REAL JCOOR_C              ! coordinate of track hits
      REAL XDCH                 ! track space points in drift chambers
      INTEGER NOVERFL,OVFLBIT   ! nb of overflows and overflow bits
      INTEGER IEFF              ! efficiency

      COMMON /DCHSAC_CMPBLK/ NHITS(16),NHT(MAXTRACK),
     +     JPLANE(MAXTRACK,CMAXHIT),
     +     JWIRE(MAXTRACK,CMAXHIT),JTIME_C(MAXTRACK,CMAXHIT),
     +     JDIST_C(MAXTRACK,CMAXHIT),JCOOR_C(MAXTRACK,CMAXHIT),
     +     XDCH(MAXTRACK,NDCH,3),NOVERFL,OVFLBIT,
     +     IEFF(MAXTRACK,2)



*************************************************************************
* COMMON /DCHCLU_CMPBLK/     rsa car file
*************************************************************************
      INTEGER NCLUSTER,MAXCLUSTER
      PARAMETER (MAXCLUSTER=64)
      INTEGER DCH_CLUS
      COMMON/DCHCLU_CMPBLK/NCLUSTER,DCH_CLUS(3,MAXCLUSTER)
```

```
c      DCH_CLUS(1,NCLUSTER) =  view number ( 1:16)
c      DCH_CLUS(2,NCLUSTER) =  measured coodinate
c      DCH_CLUS(3,NCLUSTER) =  cluster time (0.:100.ns)
```

```
************************************************************************
* COMMON /OVRFLWSIM_CMPBLK/     res car file
************************************************************************
      integer NSOVF       ! simulated overflow condition with > 5 hits
      real    TOVF_CLOSEB     ! ovf closest to zero, but Before 0
      real    TOVF_CLOSEA     ! ovf closest to zero, but After 0
                              ! these 2 variables are obtained looping
                              ! on the chambers but number 3

      COMMON /OVRFLWSIM_CMPBLK/ NSOVF,TOVF_CLOSEB,TOVF_CLOSEA
```

```
************************************************************************
* COMMON /OVLRND_CMPBLK/        ovl car file
************************************************************************
      Integer  RNDty          ! random type: 1=Ks, 2=Kl (from PRE bank)
      Integer  RNDts          ! timestamp (from PRE bank)
      Integer  RNDrun         ! run# of the random used from (PRE bank)
      Integer  RNDbur         ! burst# of the random used(from PRE bank)
      Integer  RNDused        ! #times random used so far(from PRE bank)
      Integer  PDSused        !
      Real     RNDsps         ! sps phase  (from PSCA bank)
      Real     RND50          ! main phase (from PSCA bank)
      Integer  RND_ROVF(4)    ! ring overflow info
      Real     RND_TOVF(64)   ! TDC overflow info
      REAL     RNDKlmon_DNDT  ! Kl monitor intensity (from PSCA bank)
      REAL     RNDKsmon_DNDT  ! Ks monitor intensity (from PSCA bank)
      REAL     RNDTagmon_DNDT ! tagger monitor intensity(from PSCA)
      REAL     RNDQx_DNDT     ! Qx intensity (from PSCA bank)
      REAL     RNDAKs_DNDT    ! AKS intensity (from PSCA bank)

      COMMON /OVLRND_CMPBLK/ RNDty, RNDts, RNDrun, RNDbur, RNDused,
     +                       RNDsps, RND50, RND_TOVF, RND_ROVF,
     +                       PDSused, RNDKlmon_DNDT, RNDKsmon_DNDT,
     +                       RNDTagmon_DNDT, RNDQx_DNDT, RNDAKs_DNDT
```

```
************************************************************************
* COMMON /AKS_REC/     rec car file
************************************************************************
      integer nchann
      parameter (nchann=5)
      real aksrec(nchann,4)
      real aks_eff(2)
      integer iflag_aksrec
C
      COMMON /aks_REC/aksrec,aks_eff,iflag_aksrec
C
C     aksrec(n,1) = reconstructed number of mips for counter n
C     aksrec(n,2) = reconstructed time for counter n
C     aksrec(n,3) = pmb reconstruction error flag for counter n
C     aksrec(n,4) = reconstructed second time for counter n
C     aks_eff(1) = efficiency of counter n.4
C     aks_eff(2) = efficiency sigma of counter n.4
C     iflag_aksrec = not yet implemented
```

H SQLITE database variables

H.1 Variables on a Compact Structure

```
 - Bad bursts for each detector:       in BadBurst   structure
     defined on a burst or run basis
 - Time offsets per Run:               in TimeOffset structure
```

H.2 Variables on COMMON ABCOG_PARAMS

```
  ABCOG\_PARAMS\_ALPHA       Alpha parameter
  ABCOG\_PARAMS\_ALPHA_COEFF Coefficient used for alpha determination
  ABCOG\_PARAMS\_BETA        Beta parameter
  ABCOG\_PARAMS\_BETA_COEFF  Coefficient used for beta determination
  ABCOG\_PARAMS\_MKP         Mass for K+
  ABCOG\_PARAMS\_MKPERR      Error on mass shift for K+
  ABCOG\_PARAMS\_MKN         Mass shift for K-
  ABCOG\_PARAMS\_MKNERR      Error on mass shift for K-
  ABCOG\_PARAMS\_COGX1P      X of COG at DCH1 for pos. track
  ABCOG\_PARAMS\_COGY1P      Y of COG at DCH1 for pos. track
  ABCOG\_PARAMS\_COGX1N      X of COG at DCH1 for neg. track
  ABCOG\_PARAMS\_COGY1N      Y of COG at DCH1 for neg. track
  ABCOG\_PARAMS\_COGX4P      X of COG at DCH4 for pos. track
  ABCOG\_PARAMS\_COGY4P      Y of COG at DCH4 for pos. track
  ABCOG\_PARAMS\_COGX4N      X of COG at DCH4 for neg. track
  ABCOG\_PARAMS\_COGY4N      Y of COG at DCH4 for neg. track
  ABCOG\_PARAMS\_STATUS      =0 if query of db ok, =1 if problems
  ABCOG\_PARAMS\_pkp         Momentum for positive kaon beam
  ABCOG\_PARAMS\_pkdxdzp     dxdz for positive kaon beam
  ABCOG\_PARAMS\_pkdydzp     dydz for positive kaon beam
  ABCOG\_PARAMS\_pkxoffp     x offset for positive kaom beam at z=0
  ABCOG\_PARAMS\_pkyoffp     y offset for positive kaom beam at z=0
  ABCOG\_PARAMS\_pkm         Momentum for negative kaon beam
  ABCOG\_PARAMS\_pkdxdzm    dxdz for negative kaon beam
  ABCOG\_PARAMS\_pkdydzm    dydz for negative kaon beam
  ABCOG\_PARAMS\_pkxoffm    x offset for negative kaom beam at z=0
  ABCOG\_PARAMS\_pkyoffm    y offset for negative kaom beam at z=0
```

The syntax of the beam parameter variables listed above is for FORTRAN user, for C the corresponding variables can be deduced in the following way:

ABCOG_PARAMS_pkdxdzm → abcog_params.pkdxdzm

bit	1997 data	1998+1999 data
0	chan 0-bit 4 HOD-L1in Qx/2	chan 0-bit 5 HOD-L1in Qx/D
1	chan 0-bit 5 HOD-L1in Qx	chan 1-bit 9 HOD-L1in Qx
2	chan 0-bit 6 HOD-L1in FT0	chan 1-bit10 HOD-L1in FT0
3	chan 0-bit 7 HOD-L1in FT1	chan 1-bit11 HOD-L1in FT1
4	chan 3-bit11 NUT-L1in Etot	chan 3-bit11 NUT-L1in Etot
5	chan 3-bit18 NUT-L1in FT0 NUT	chan 3-bit18 NUT-L1in FT0 NUT
6	chan 3-bit19 NUT-L1in FT1 NUT	chan 3-bit19 NUT-L1in FT1 NUT
7	chan 4-bit 4 SUB1-L2in Qx/2	chan 4-bit 5 SUB1-L2in Qx/D
8	chan 4-bit 5 SUB1-L2in Qx	chan 4-bit13 SUB1-L2in Qx
9	chan 4-bit 6 SUB1-L2in FT0	chan 4-bit15 SUB1-L2in FT0
10	chan 4-bit 7 SUB1-L2in FT1	chan 4-bit14 SUB1-L2in FT1
11	chan 6-bit 3 SUB3-L2in Etot	chan 6-bit 3 SUB3-L2in Etot
12	chan14-bit 0 L1 out to MBOX FT0	chan14-bit 0 L1 out to MBOX FT0
13	chan14-bit 1 L1 out to MBOX FT1	chan14-bit 1 L1 out to MBOX FT1
14	chan14-bit16 L1 out to MBOX Trg bit 0 to MBX	chan14-bit16 L1 out to MBOX Trg bit 0 to
15	chan14-bit17 L1 out to MBOX Trg bit 1 to MBX	chan14-bit17 L1 out to MBOX Trg bit 1 to
16	chan14-bit18 L1 out to MBOX Trg bit 2 to MBX	chan14-bit18 L1 out to MBOX Trg bit 2 to
17	chan14-bit19 L1 out to MBOX Strobe to MBX	chan14-bit19 L1 out to MBOX Strobe to M
18		chan14-bit20 L1 out to MBOX L1on
19		chan12-bit10 L1 out: 2tracks
20		chan12-bit 5 L1 out: QX/D
21		chan12-bit13 L1 out: QX
22		chan12-bit15 L1 out: FT0
23		chan12-bit14 L1 out: FT1
24		chan13-bit 0 L1 out: Etot
25		chan 4-bit10 SUB1-L2in: 2tracks
26		chan 5-bit 2 L2 in: TCPLX
27		chan 5-bit 3 L2 in: IN-TIME
28	chan 5-bit 9 L2 in: Pi+Pi- OK	chan 5-bit 9 L2 in: Pi+Pi- OK
29		chan 5-bit10 L2 in: fatal
30	chan 1-bit 9 DCH-L1in Qx/32	chan10-bit24 AKS PU
31	chan 4-bit14 SUB1-L2in Qx/32	

Table 6: sevt->DETstatus[0].ChTrEff[its] description

name x=l/s	meaning	Integration time	
		<2001	2001
evt-> beamIntKx.Integ2us (BEAMINTKX_INTEG2US)	average	2μsec	1 μsec
evt-> beamIntKx.Integ15us(BEAMINTKX_INTEG15US)	average	15μsec	200 nsec
evt-> beamIntKx.Integ30us(BEAMINTKX_INTEG30US)	average	30μsec	3 μsec
evt-> beamIntKx.Integ60us(BEAMINTKX_INTEG60US)	average	60μsec	15 μsec
evt-> beamIntKx.Qual2us (BEAMINTKX_QUAL2US)	rms	2μsec	1 μsec
evt-> beamIntKx.Qual15us (BEAMINTKX_QUAL15US)	rms	15μsec	200 nsec
evt-> beamIntKx.Qual30us (BEAMINTKX_QUAL30US)	rms	30μsec	3 μsec
evt-> beamIntKx.Qual60us (BEAMINTKX_QUAL60US)	rms	60μsec	15 μsec
evt-> beamIntKx.packsampl[1-4] (BEAMINTKX_PACKSAMP(1-4))	samples		200 nsec

Table 7: Beam intensity integrators meanings

bit	mask	mnemonic	description
0	0x1	—HF_NEUTRAL—	$\pi^{\pm}\pi^0\pi^0$ event
1	0x2	—HF_LKRDNSC—	the event was recorded without reading the LKR
2-5	0x3c	—HF_NTRACKS—	Number of tracks
6	0x40	—HD_GHOSTTR—	the first best vertex was discarded because of a ghost track
7			Empty
8-11	0xf00	—HF_DTSTAMP—	Distance to previous event (if <= 15 timestamps)
12-15	0xf000	—HF_CLUSTER—	Number of non-assoc. LKR clusters
16-19	0xf0000	—HF_CLINTIM—	Non-assoc. LKR clusters in ± 15 ns window
20-23	0xf00000	—HF_TRINTIM—	Extra tracks in ± 15 ns window

Table 8: Meaning of the bits of $hevt->flag$ (HEVT_FLAG).

<2002	Bit	meaning	verdict	2002	Bit	meaning
	0	Data corruption	no-fatal	0	Tstamp difference between headers	no-fatal
	1	Internal mixing	no-fatal	1	Wrong word count in one plane	no-fatal
	2	Global Mixing	no-fatal	2	Token time out	no-fatal
	3	Data corruption	no-fatal	3	Data ready time out	no-fatal
	4	Data corruption	no-fatal	4	Tstamp unrecovered mismatch	no-fatal
	5	Internal mixing	no-fatal	5	Wrong number of headers	no-fatal
	6	Data corruption	no-fatal	6	Header-CSC mismatch	no-fatal
	7			7		
	8			8	First word is not a header	no-fatal
	9	Data corruption	no-fatal	9	Plane number mismatch	no-fatal
	10	Global Mixing	no-fatal	10	TStamp differs from most frequent one	no-fatal
	11	Data corruption	no-fatal	11	Chamber number mismatch	no-fatal

Table 9: DCH decoding error bits description

Bit set	analysis routine called
0	fuser_selcharged
1	fuser_sel2pi0
2	fuser_sel3pi0
3	fuser_bluefield
4	fuser_lkrpedcor
5	fuser_lkrposcor
6	fuser_lkrsharing
7	fuser_hodotime
8	fuser_nhodtime
9	fuser_lkrtime
10	fuser_tagtime
11	fuser_aksflag
12	muon_rec
13	muonReject
14	user_GeomCor
15	fuser_newcharged
16	user_magnetcorr
17	muon_trackrec (uncorr. tracks)
18	muon_vertexrec
19	fuser_lkrcalcor
20	subset1 of fuser_lkrcalcor
21	subset2 of fuser_lkrcalcor
21	subset3 of fuser_lkrcalcor
23	fuser_sel2gam
24	muon_trackrec (corr. tracks)
25	fuser_lkrcalhi2k

Table 10: Correspondance bit in FlagCorr - analysis routines

$evt->TrigWordL(i)$ EVT_TRIGWORDL(i)	Trigger words	
$evt->TimeStampL(i)$ EVT_TIMESTAMPL(i)	Time stamps	
$evt->DCHDecErrorL(i)$ EVT_DCHDECERRORL(i)	DCHerrflagPDS (see table above)	
$evt->LKRHADecErrorL(i)$	=Q(LPLKR+9) if=0 decoding OK	
EVT_LKRHACDECERRORL(i)	bit set	
	0	timestamp not consistent among the links
	1	trigger word not consistent among the links
	2	local event number not consistent among the links
	3	decoding error: recover missing trailer
	4	decoding errors
	5	No data (no LKR event header)
	6	NHitcell=0
	7	Too much data
	8	Not enough data
	9	decoding error from DCP header
	10	decoding error from channel header
	11	decoding error from trailer
	14	HAC decoding called
	15	HAC decoding problem
$evt->NeutralInfoL(i)$	bit set	
EVT_NEUTRALINFOL(i)	0	Routine called
	1	downscaled event
	2	KS candidate
	3	KS candidate flagged
	4	KL candidate
	5	KL candidate passing all physics cuts
	6	no KS or no KL found
	7	Pb. with REC or DCHREC or RDTK or VTX banks
$evt->ChargedInfoL(i)$	= iflag1(L3B filter) + 16*iflag2(golden filter)	
EVT_CHARGEDINFOL(i)	iflag1	
	1	routine called
	2	$\leq$ 4 clusters-evt kept
	3	$>$ 4clusters with one comb. in time-evt kept
	4	no comb. with geq5 clusters in time-evt kept
	5	geq5 clu. in time and fullfilling $2\pi^0$ cut.
	iflag2	
	1	routine called
	2	all banks OK
	3	geq 4 clusters
	4	geq 4 clusters in time
	5	above energy cut
	6	+ 1 comb. fullfilling $E_{tot} > cut$
	7	+ fullfilling cog cut
	8	+ fullfilling $c.\tau$ cut
	9	+ fullfilling π^0 mass cut

Table 11: Description of the content of the complete lists(i.e. one entry per recorded event)

Scaling factor	Usage
const SCF_EK = 1000;	kaon energy
const SCF_SPSPHASE = 1000;	SPS phase wrt to 40MHz clock
const SCF_EVTTIME = 1000;	event time
const SCF_MREC = 100000;	reconstructed mass
const SCF_VTXX = 100;	charged vertex: x-coord.
const SCF_VTXY = 100;	charged vertex: y-coord.
const SCF_VTXZ = 100;	charged vertex: z-coord.
const SCF_VTXCDA = 100;	charged vertex: cda
const SCF_VTXTIME = 1000;	charged vertex: time
const SCF_VTXPTSQ = 10000000;	charged vertex: p_t^2
const SCF_AKSA = 100;	z AKS
const SCF_ESC = 10000;	energy scale
const SCF_MPI0 = 100000;	π^0 mass
const SCF_RELLI = 1000;	ellipse number
const SCF_PI0MERR = 100000;	error on π^0 mass
const SCF_TAGTIME = 1000;	tagger proton time
const SCF_AKSTIME = 1000;	AKS hit time
const SCF_AKLTIME = 1000;	AKL hit time
const SCF_TRKP = 10000;	charged track momentum
const SCF_TRKQL = 1000;	track quality
const SCF_TRKX = 1000;	track x-coord. before/after magnet
const SCF_TRKY = 1000;	track y-coord. before/after magnet
const SCF_TRKZ = 100;	track z-coord. before/after magnet
const SCF_TRKDXDZ = 100000;	track slope before/after magnet
const SCF_TRKDYDZ = 100000;	track slope before/after magnet
const SCF_TRKTIME = 100;	track time
const SCF_TRKDDEAD = 1000;	d(cm) to closest dead cell
const SCF_HODTIME = 1000;	hodoscope hit time
const SCF_NHODTIME = 1000;	neutral hodoscope hit time
const SCF_LKRE = 1000;	Lkr total/cluster energy
const SCF_LKRX = 1000;	Lkr cluster x-coord.
const SCF_LKRY = 1000;	Lkr cluster y-coord.
const SCF_LKRTIME = 1000;	Lkr cluster time
const SCF_ESPY = 1000;	Neutral Trigger energy
const SCF_EOVP = 1000;	E/p
const SCF_HACE = 1000;	Hac total/cluster energy
const SCF_HACBF = 1000;	Hac E_{back}/E_{front}
const SCF_MUVTIME = 1000;	Muon veto time
const SCF_OVFLTIME = 1000;	Overflow time
const SCF_OVFLX = 100;	
const SCF_OVFLY = 100;	
const SCF_KSMTIME = 1000;	KSM time
const SCF_DTARG = 1000;	D_{target}
const SCF_PTPRIME = 1000000;	$\tilde{P}_{\perp}^2$

I F77 Interface

The tables below list the names of the variables used to access the event data in COmPACT. These variable names are case sensitive, unlike normal F77 variables, and so *must be typed in as they appear here.* Failure to do this will generate compile time errors with the F77 compiler.

I.1 Calling analysis routines

Some analysis routines are now in compact. Important information valid for both the fortran and the C interfaces have been given in section 4. The way to call the routines is given as an example in *fuser_cmpEvent.example.F*.

J COmPACT Structures

J.1 NUTkmu3

Variable	Type	Description
KMU3_NUT_TIMESTAMP	int	NUT time stamp = IQ(LPNUT+1)
KMU3_NUT_MAXE	int	1000*Energy at IPC (GeV) = IQ(LPNUT+6)
KMU3_NUT_MAXCOG	int	1000*COG at IPC (cm) = IQ(LPNUT+7)
KMU3_NUT_MAXD	int	1000*d at IPC (cm) = IQ(LPNUT+8)
KMU3_NUT_MAXZ	int	1000*z at IPC (cm) = IQ(LPNUT+9)
KMU3_NUT_MAXL	int	1000*l at IPC (lifetime units) = IQ(LPNUT+10)
KMU3_NUT_MAXTF	int	1000*time corr. at IPC (ns) = IQ(LPNUT+11)
KMU3_NUT_SPY_X(64)	int	spy channels (X view) = IQ(LPNUT+28+i)
KMU3_NUT_SPY_Y(64)	int	spy channels (Y view) = IQ(LPNUT+92+i)
KMU3_NUT_SPY_XCBIT(6)	int	spy ctrl bits (X view) = IQ(LPNUT+162+i)
KMU3_NUT_SPY_YCBIT(6)	int	spy ctrl bits (Y view) = IQ(LPNUT+168+i)

J.2 RNDMsummary

Variable	Type	Description
RNDM_TYPE	int	type of random used(KS=1, KL=2) = Q(LPRE+1)
RNDM_TIMESTAMP	int	timestamp of random used = Q(LPRE+3)
RNDM_RUN_BURST	int	Not used in versions $\geq$ 3.4
RNDM_RUN	int	Run Number of random used = Q(LPRE+9)

Variable	Type	Description
RNDM_BURST	int	Burst time stamp of random used = Q(LPRE+10)
RNDM_NUSED	int	Nb of times RNDM evt was used so far=Q(LPRE+8)
RNDM_SPSPHASE	float	SPS phase of random evt
RNDM_MAINPHASE	float	main(50Hz) phase of random evt
RNDM_PDSUSED	int	one bit/detector overlayed as in LQ(LPRE)
RNDM_TOVRFLW(64)	float	time of nearest ovflw to rndm evt
RNDM_ROVRFLW(4)	int	ring buffer overflow for each chamber
RNDM_KLMONDNDT	float	intensity in KLmon for random event (cf doc230698)
RNDM_KSMONDNDT	float	intensity in KSmon for random event
RNDM_TAGMONDNDT	float	intensity in TAGGERmon for random event
RNDM_QXDNDT	float	intensity in QX for random event
RNDM_AKSDNDT	float	intensity in AKS for random event

J.3 L3TRIGhist

Variable	Type	Description
TRHIST_TRIGWORD(*ievt*)	int	Trigger word for previous event ievt
TRHIST_TSTAMP_DIFF(*ievt*)	int	Timestamp difference from current event

J.4 EVTtimestamp

Variable	Type	Description
TSTAMP_TIMESTAMP(n)	int	Time stamp which is different
TSTAMP_SOURCE(n)	int	source of timestamp

J.5 PMBscaler

Variable	Type	Description
SCALER_N	int	counts this timeslice =Q(PSCA+x+2) (cf doc 230698)
SCALER_DNDT	float	difference/timestamp diff. =Q(PSCA+x+3)

J.6 PMBtimeslice

Variable	Type	Description
TRIG_CHAN(*tslice*,16)	int	16 channels * 24 bits=Q(PCAT+19+x)

J.7 TAGhit

Variable	Type	Description
TAG_HIT_AMPL(*chan*,*hit*)	float	amplitude of hit = Q(RTAG+x+1)
TAG_HIT_TIME(*chan*,*hit*)	float	time of hit = Q(RTAG+x+3)
TAG_HIT_CHI2(*chan*,*hit*)	float	χ^2 of hit = Q(RTAG+x+5)
TAG_HIT_STATUS(*chan*,*hit*)	int	status code of hit = Q(RTAG+x+6)

J.8 TAGchannel

Variable	Type	Description
TAG_N(*chan*)	int	channel number=Q(RTAG+n+1)
TAG_NHIT(*chan*)	u_int	Number of tagger hits in this channel

J.9 KSMhit

Variable	Type	Description
KSM_COUNTER(*hit*)	int	counter number(1-8) = Q(PKSM+x+2)
KSM_PHEIGHT(*hit*)	float	pulse height(GeV) = Q(PKSM+x+3)
KSM_TIME(*hit*)	float	time of hit(ns) = Q(PKSM+x+4)
KSM_PDSFLAG(*hit*)	int	decoding flag=Q(PKSM+x+1)

J.10 BeamInt

Variable	Type	Description
BEAMINTKX_INTEG2US	int	Integration over 2μs
BEAMINTKX_INTEG15US	int	Integration over 15μs
BEAMINTKX_INTEG30US	int	Integration over 30μs
BEAMINTKX_INTEG60US	int	Integration over 60μs
BEAMINTKX_QUAL2US	int	quality of integration over 2μs
BEAMINTKX_QUAL15US	int	quality of integration over 15μs
BEAMINTKX_QUAL30US	int	quality of integration over 30μs
BEAMINTKX_QUAL60US	int	quality of integration over 60μs
BEAMINTKX_PACKSAMP(4)	int	packed samples for channel 2

J.11 AKScounter

Variable	Type	Description
AKS_TIME(*counter*)	float	time of hit in counter (CB aks_rec_)
AKS_TIME2(*counter*)	float	time of snd hit (available from 4.2)

Variable	Type	Description
AKS_MIPS(*counter*)	float	number of m.i.p.s (CB aks_rec_)
AKS_ERROR(*counter*)	int	error flag (CB aks_rec_)

J.12 AKLhit

Variable	Type	Description
AKL_POCKET(*hit*)	int	pocket number (1-7) = Q(RAHI+2)
AKL_LAYER(*hit*)	int	layer number (1,2) = Q(RAHI+3)
AKL_COUNTER(*hit*)	int	counter number (1-12) = Q(RAHI+4)
AKL_MIPS(*hit*)	float	number of m.i.p.s = Q(RAHI+5)
AKL_TIME(*hit*)	float	time of AKL hit = Q(RAHI+6)
AKL_ONTFLAG(*hit*)	int	on time flag = Q(RAHI+7)

J.13 DCHcluster

Variable	Type	Description
DCHCLU_VIEW(*clu*)	int	View (1-16: 1-4 ch1, 5-8 ch2, 9-10 ch3, 13-16 ch4)
DCHCLU_COORD(*clu*)	int	cluster coordinate for view ($\in [0;256]cm$)
DCHCLU_TIME(*clu*)	int	cluster time (ns) (CB dchclu_cmpblk_ for entire struct.)

J.14 DCHhit

Variable	Type	Description
TRACK_HIT_PLANE(*track,hit*)	int	plane number (CB dchsac_cmpblk_)

Variable	Type	Description
TRACK_HIT_WIRE(*track*,*hit*)	int	wire number (CB dchsac_cmpblk_)
TRACK_HIT_TIME(*track*,*hit*)	float	time of hit (CB dchsac_cmpblk_)
TRACK_HIT_DIST(*track*,*hit*)	float	distance of hit (CB dchsac_cmpblk_)
TRACK_HIT_POS(*track*,*hit*)	float	position of hit (CB dchsac_cmpblk_)

J.15 DCHspacepoint

Variable	Type	Description
TRACK_SPNT_X(*track*,*dch*)	float	x coord (CB dchsac_cmpblk_)
TRACK_SPNT_Y(*track*,*dch*)	float	y coord (CB dchsac_cmpblk_)
TRACK_SPNT_Z(*track*,*dch*)	float	z coord (CB dchsac_cmpblk_)

J.16 DCHtrack

Variable	Type	Description
TRACK_PQ(*track*)	float	track mom.*charge=Q(RDTK+4)*Q(RDTK+3)
TRACK_P(*track*)	float	track momentum=Q(RDTK+4)
TRACK_Q(*track*)	int	not FILLED
TRACK_PERR(*track*)	float	error on momentum=Q(RDTK+5)
TRACK_CHI2(*track*)	float	χ^2 of track=Q(RDTK+6)
TRACK_BX(*track*)	float	x position before magnet=Q(RDTK+7)
TRACK_BY(*track*)	float	y position before magnet=Q(RDTK+8)
TRACK_BDXDZ(*track*)	float	dx/dz before magnet=Q(RDTK+9)

Variable	Type	Description
TRACK_BDYDZ(*track*)	float	dy/dz before magnet=Q(RDTK+10)
TRACK_X(*track*)	float	x position after magnet=Q(RDTK+15)
TRACK_Y(*track*)	float	y position after magnet=Q(RDTK+16)
TRACK_DXDZ(*track*)	float	dx/dz after magnet=Q(RDTK+17)
TRACK_DYDZ(*track*)	float	dy/dz after magnet=Q(RDTK+18)
TRACK_TIME(*track*)	float	track time=Q(RDTK+23)
TRACK_QUALITY(*track*)	float	quality of track=Q(RDTK+24)
TRACK_HODTIME(*track*)	float	hod. time ass. with track=Q(RDTK+25)
TRACK_HODSTATUS(*track*)	int	hod. flags used for time calc.=Q(RDTK+26)
TRACK_NHITS(*track*)	int	Nb of hits - always filled (CB dchsac_cmpblk_)
TRACK_NHIT(*track*)	u_int	Number of track hits - only for DCH streams
TRACK_EXHAC(*track*)	float	energy in HAC x strip=Q(RHCL+13)
TRACK_EYHAC(*track*)	float	energy in HAC y strip=Q(RHCL+14)
TRACK_DDEADCELL(*track*)	float	Dist. to closest dead cell(cm)=Q(RDTK+27)
TRACK_SIGXX(*track*)	float	Error on x (cm)=Q(RDTK+11)
TRACK_SIGYY(*track*)	float	Error on y (cm)=Q(RDTK+12)
TRACK_SIGDXDX(*track*)	float	Error on dx/dz =Q(RDTK+13)
TRACK_SIGDYDY(*track*)	float	Error on dy/dz =Q(RDTK+14)

Variable	Type	Description
TRACK_SIGXDX(*track*)	float	correlation sigma(x,dx/dz)=Q(RDTK+28)
TRACK_SIGXY(*track*)	float	correlation sigma(x,y)=Q(RDTK+29)
TRACK_SIGDXY(*track*)	float	correlation sigma(dx/dz,y)=Q(RDTK+30)
TRACK_SIGXDY(*track*)	float	correlation sigma(x,dy/dz)=Q(RDTK+31)
TRACK_SIGDXDY(*track*)	float	correlation sigma(dx/dz,dy/dz)=Q(RDTK+32)
TRACK_SIGYDY(*track*)	float	correlation sigma(y,dy/dz) =Q(RDTK+33)
TRACK_HITPATTERN(*track*)	int	one bit per wire for efficiency studies
TRACK_EFFICIENCY(*track*,2)	int	bit coded words for eff. studies
TRACK_SPAREINT(*track*,2)	int	2 spare integers
TRACK_SPAREFLOAT(*track*,2)	int	2 spare floats
TRACK_LKRCLU(*track*)	int	LKR clu. index=0 $\rightarrow NLkr - 1$ (std_ep)
TRACK_EOVP(*track*)	float	E/p (comp. in std_ep)
TRACK_ESPY(*track*)	float	Energy in NUTspy (comp in espy)
TRACK_MUVTIME(*track*)	float	time(ns) of closest muon in time (comp murec1198)
TRACK_ANAVAR(*track*,20)	float	Provision for analysis variables
TRACK_ANAFLAG(*track*,5)	int	Provision for analysis flags

J.17 DCHvertex

Variable	Type	Description
VERTEX_X(*vertex*)	float	x position(cm)=Q(RDVX+6)

Variable	**Type**	**Description**
VERTEX_Y(*vertex*)	float	y position(cm)=Q(RDVX+7)
VERTEX_Z(*vertex*)	float	z position(cm)=Q(RDVX+8)
VERTEX_CHI2(*vertex*)	float	chi2 vertex fit =Q(RDVX+24)
VERTEX_BDXDZPOS(*vertex*)	float	dx/dz bef.mag. Pos. track =Q(RDVX+10)
VERTEX_BDYDZPOS(*vertex*)	float	dy/dz bef.mag. Pos. track =Q(RDVX+11)
VERTEX_BDXDZNEG(*vertex*)	float	dx/dz bef.mag. Neg. track =Q(RDVX+12)
VERTEX_BDYDZNEG(*vertex*)	float	dy/dz bef.mag. Neg. track =Q(RDVX+13)
VERTEX_NBDXDZTRACK(*vertex*)	u_int	Number of dxdz for track i
VERTEX_BDXDZTRACK(*vertex*,7)	float	dxdz for track i
VERTEX_NBDYDZTRACK(*vertex*)	u_int	Number of dxdz for track i
VERTEX_BDYDZTRACK(*vertex*,7)	float	dxdz for track i
VERTEX_CDA(*vertex*)	float	closest approach(cm)=Q(RDVX+9)
VERTEX_ERRORFLAG(*vertex*)	int	1:Error ana. done, 0: not done=Q(RDVX+23)
VERTEX_BLUEFLAG(*vertex*)	int	1:BlueField used, 0: not used=Q(RDVX+14)
VERTEX_PPIPI(*vertex*)	float	$\pi^+\pi^-$ vertex momentum(GeV)=Q(RDVX+3)
VERTEX_MPIPI(*vertex*)	float	$\pi^+\pi^-$ invariant mass(GeV)=Q(RDVX+4)
VERTEX_MLAMBDA(*vertex*)	float	Λ^0 mass(GeV) - **All the following variables**
VERTEX_MALAMBDA(*vertex*)	float	Anti-Λ^0 mass - **computed from**
VERTEX_IPTRK(*vertex*)	int	index of the positive track

Variable	**Type**	**Description**
VERTEX_INTRK(*vertex*)	int	index of the negative track
VERTEX_NITRACK(*vertex*)	u_int	Number of Index of tracks i in this vertex
VERTEX_ITRACK(*vertex*,7)	int	Index of tracks i in this vertex
VERTEX_RCOG(*vertex*)	float	cog(cm) **rlib/srcVertexVariables.c**
VERTEX_PTSQKS(*vertex*)	float	_$P_{\perp}^{2}$ (GeV^2) for K_s
VERTEX_PTSQKL(*vertex*)	float	_$P_{\perp}^{2}$ (GeV^2) for K_l
VERTEX_PTPRIMEKS(*vertex*)	float	_$\tilde{P_{\perp}}^{2}$ (GeV^2) for K_s
VERTEX_PTPRIMEKL(*vertex*)	float	_$\tilde{P}_{\perp}^{2}$ (GeV^2) for K_l
VERTEX_DTINTKS(*vertex*)	float	_$D_{target}^{in}(cm)$ for K_s
VERTEX_DTINTKL(*vertex*)	float	_$D_{target}^{in}(cm)$ for K_l
VERTEX_DTOUTTKS(*vertex*)	float	_$D_{target}^{out}(cm)$ for K_s
VERTEX_DTOUTTKL(*vertex*)	float	_$D_{target}^{out}(cm)$ for K_l
VERTEX_DTINVKS(*vertex*)	float	_$D_{vertex}^{in}(cm)$ for K_s
VERTEX_DTINVKL(*vertex*)	float	_$D_{vertex}^{in}(cm)$ for K_l
VERTEX_DTOUTVKS(*vertex*)	float	_$D_{vertex}^{out}(cm)$ for K_s
VERTEX_DTOUTVKL(*vertex*)	float	_$D_{vertex}^{out}(cm)$ for K_l
VERTEX_EANGLE(*vertex*)	float	Kaon Ener.(GeV) (open angle)
VERTEX_CTAU(*vertex*)	float	(lifetime units)
VERTEX_PHIDECAY(*vertex*)	float	Decay plane phi (rd)
VERTEX_ASP(*vertex*)	float	_$(P_- - P_+)/(P_- + P_+)$: signed asymetry
VERTEX_HODOTIMEANA(*vertex*)	float	(comp. by USER_HODOTIME routine) (ns)
VERTEX_TYPE(*vertex*)	int	vertex type - (fuser_newcharged)
VERTEX_CUTS(*vertex*)	int	bit coded: which cuts passed (fuser_newcharged)

Variable	Type	Description
VERTEX_IFLAG(*vertex*)	int	see comments in fuser_newcharged.F
VERTEX_ANAVAR(*vertex*,20)	float	Provision for analysis variables
VERTEX_ANAFLAG(*vertex*,5)	int	Provision for analysis flags

J.18 DCHNROtdof

Variable	Type	Description
DCHNRO_DOF_TDOF_TTIME(*idof,it*)	float	Q(LPDOFNR+10) tdc dof time relative to the Timestamp(ns)
DCHNRO_DOF_TDOF_TSTATUS(*idof,it*)	int	Q(LPDOFNR+11) status word for tdc dof

J.19 DCHNROdof

Variable	Type	Description
DCHNRO_DOF_DCHERROR(*idof*)	int	Q(LPDOFNR+1) DECDCH error code, 0=raw data format OK
DCHNRO_DOF_NRO(*idof*)	int	Q(LPDOFNR+2) Nb of planes having ring ovf (=1)
DCHNRO_DOF_IPL(*idof*)	int	Q(LPDOFNR+7) plane number= only 1
DCHNRO_DOF_RSTATUS(*idof*)	int	Q(LPDOFNR+8) status word for ring dof
DCHNRO_DOF_NTDOF(*idof*)	u_int	Number of tdc overflows info

J.20 DCHNROhit

Variable	Type	Description
DCHNRO_HIT_ID(*ihit*)	float	Q(LPDCHNR+x+1)
DCHNRO_HIT_IPLANE(*ihit*)	float	Q(LPDCHNR+x+2) plane number (increasing order)

Variable	Type	Description
DCHNRO_HIT_IWIRE(*ihit*)	float	Q(LPDCHNR+x+3) wire number (increasing order)
DCHNRO_HIT_TIME(*ihit*)	float	Q(LPDCHNR+x+4) drift time (nsec)
DCHNRO_HIT_ITRACK(*ihit*)	float	Q(LPDCHNR+x+5)
DCHNRO_HIT_ITSLOT(*ihit*)	float	Q(LPDCHNR+x+6) time slot info

J.21 DCHNROgen

Variable	Type	Description
DCHNRO_EVTIME	float	Q(LPDCHNR+1)
DCHNRO_TIMEST	int	Q(LPDCHNR+5) .and. Q(LPDCHNR+6)
DCHNRO_TRIGW	int	Q(LPDCHNR+7)
DCHNRO_FLAG	int	Q(LPDCHNR+8)
DCHNRO_NHIT	u_int	Number of hits info
DCHNRO_NDOF	u_int	Number of overflows info for the nro plane

J.22 DCHFEstatus

Variable	Type	Description
DCHFE_INTTRIG	int	Q(LPDOFE+8)
DCHFE_F1PAR	int	
DCHFE_PIPEFULL	int	
DCHFE_PIPEFULLWIRE(8)	int	

J.23 DCHmult

Variable	Type	Description
DCHEFFMULT_MBXPLANEEFF(6)	int	number of hit for plane packed
DCHEFFMULT_L1TRK24EFF(2)	int	multiplicity in DCH1 for L1 trigger packed
DCHEFFMULT_DCHSNOWERR(2)	int	16 bit for DCH snow effect(2-1)(4-3)

Variable	**Type**	**Description**
DCHEFFMULT_MBXMULT	int	multiplicity in DCH
DCHEFFMULT_TRK24ON	int	L1Trk24=1 –¿ trigger OK

J.24 SGNwire

Variable	**Type**	**Description**
CHAMBER_SPNT_WIRE_NUMBER(*dch*,*spnt*,*n*)	int	wire numb. (CB dchsgn_cmpblk_)
CHAMBER_SPNT_WIRE_DRIFTTIME(*dch*,*spnt*,*n*)	float	drift time (CB dchsgn_cmpblk_)

J.25 SGNspacepoint

Variable	**Type**	**Description**
CHAMBER_SPNT_X(*dch*,*spnt*)	float	x coordinate (CB dchsgn_cmpblk_)
CHAMBER_SPNT_Y(*dch*,*spnt*)	float	y coordinate (CB dchsgn_cmpblk_)
CHAMBER_SPNT_NWIRE(*dch*,*spnt*)	u_int	Number of Wire Nb + drift time
CHAMBER_SPNT_QUALITY(*dch*,*spnt*,4)	int	qual. flag/view (CB dchsgn_cmpblk_)

J.26 SGNaccidental

Variable	**Type**	**Description**
CHAMBER_ACC_X(*dch*,*acc*)	float	x coordinate (CB dchsgn_cmpblk_)
CHAMBER_ACC_Y(*dch*,*acc*)	float	y coordinate (CB dchsgn_cmpblk_)
CHAMBER_ACC_TIME(*dch*,*acc*)	float	Accidental time (CB dchsgn_cmpblk_)

J.27 SGNchamber

Variable	Type	Description
CHAMBER_NSPNT(*dch*)	u_int	Number of space point array
CHAMBER_NACC(*dch*)	u_int	Number of accidental array
CHAMBER_TOVRFLW(*dch*,16)	float	nearest ovflw times to evt(CB dchsgn_cmpblk_)
CHAMBER_ROVRFLW(*dch*)	int	ring buffer ovrflw (CB dchsgn_cmpblk_)

J.28 SGNtrack

Variable	Type	Description
SGNTRK_P(*track*)	float	momentum*charge of track (CB dchsgn_cmpblk_)
SGNTRK_CHI2(*track*)	float	χ^2 of track - NOT filled
SGNTRK_BX(*track*)	float	x position before magnet (CB dchsgn_cmpblk_)
SGNTRK_BY(*track*)	float	y position before magnet (CB dchsgn_cmpblk_)
SGNTRK_BDXDZ(*track*)	float	dx/dz before magnet (CB dchsgn_cmpblk_)
SGNTRK_BDYDZ(*track*)	float	dy/dz before magnet (CB dchsgn_cmpblk_)
SGNTRK_X(*track*)	float	x position after magnet (CB dchsgn_cmpblk_)
SGNTRK_Y(*track*)	float	y position after magnet (CB dchsgn_cmpblk_)
SGNTRK_DXDZ(*track*)	float	dx/dz after magnet (CB dchsgn_cmpblk_)
SGNTRK_DYDZ(*track*)	float	dy/dz after magnet (CB dchsgn_cmpblk_)
SGNTRK_SPNT(*track*,4)	int	space points on track (CB dchsgn_cmpblk_)

J.29 SGNvertex

Variable	Type	Description
SGNVTX_X(*vertex*)	float	x position (CB dchsgn_cmpblk_)
SGNVTX_Y(*vertex*)	float	y position (CB dchsgn_cmpblk_)
SGNVTX_Z(*vertex*)	float	z position (CB dchsgn_cmpblk_)
SGNVTX_CDA(*vertex*)	float	closest approach (CB dchsgn_cmpblk_)
SGNVTX_FLAG(*vertex*)	int	vertex flag - NOT filled

J.30 HODhit

Variable	Type	Description
HOD_PLANE(*hit*)	int	plane number - NOT used
HOD_COUNTER(*hit*)	int	counter hit = Q(RHHI+5)
HOD_X(*hit*)	float	x coord of DCH track = Q(RHHI+6)
HOD_Y(*hit*)	float	y coord of DCH track = Q(RHHI+7)
HOD_PHEIGHT(*hit*)	float	pulse height (from PDS) = Q(RHHI+10)
HOD_TIME(*hit*)	float	time of hit (all corrections added) = Q(RHHI+13)
HOD_UCTIME(*hit*)	float	uncorrected time of hit = Q(RHHI+11)
HOD_PDSFLAG(*hit*)	int	flag from PDS data structure = Q(RHHI+3)

J.31 HODneuthit

Variable	Type	Description
HODNEUT_PLANE(*hit*)	int	Plane number (1=V, 2=H) = Q(LPHOD+x+2)

Variable	Type	Description
HODNEUT_COUNTER(*hit*)	int	scintillator number (1-64) = Q(LPHOD+x+3)
HODNEUT_PULSE(*hit*)	float	pulse height (mips) = Q(LPHOD+x+4)
HODNEUT_TIME(*hit*)	float	time (ns) from ramp = Q(LPHOD+x+5)

J.32 LKRcluster

Variable	Type	Description
LKR_ENERGY(*cluster*)	float	Cluster energy(GeV)=Q(RLCL+4)
LKR_EENERGY(*cluster*)	float	NOT filled from 4.5
LKR_E2SAMPALL(*cluster*)	float	$< E >$ (samp. 1+2) all cells in clu.(GeV)=Q(RLCL+16)
LKR_E77(*cluster*)	float	clu. energy in 7x7 cells(GeV)=Q(RLCL+17)
LKR_X(*cluster*)	float	x coord of cluster(cm)=Q(RLCL+7)
LKR_Y(*cluster*)	float	y coord of cluster(cm)=Q(RLCL+8)
LKR_RMSX(*cluster*)	float	X cluster width (cell u.) =Q(RLCL+9)
LKR_RMSY(*cluster*)	float	Y cluster width (cell u.) =Q(RLCL+10)
LKR_IMAX(*cluster*)	int	index of maximum energy cell=Q(RLCL+3)
LKR_TIME(*cluster*)	float	Time(ns)=Q(RLCL+11)
LKR_ETIME(*cluster*)	float	NOT filled from 4.5
LKR_TLATCELL(*cluster*)	float	time of most energetic lateral cell(ns)=Q(RLCL+13)
LKR_DDEADCELL(*cluster*)	float	distance of closest dead cell(cm)=Q(RLCL+14)

Variable	Type	Description
LKR_UCENERGY(*cluster*)	float	uncorrected cluster energy(GeV)=Q(RLCL+15)
LKR_CELLSREAD(*cluster*)	int	Number of cells read out=Q(RLCL+2)
LKR_STATUS(*cluster*)	int	status bits=Q(RLCL+6)
LKR_SPACHACORR(*cluster*)	float	corr. applied for spacecharge effect=Q(RLCL+18)
LKR_ECORRKE3(*cluster*)	float	cl. energy corrected with ke3 factor(GeV)=Q(RLCL+19)
LKR_SPAREINT1(*cluster*)	int	1 spare integers in case
LKR_SPAREFLOAT1(*cluster*)	float	Chi2 shower shape = Q(RLCL+12)
LKR_GAINMAX(*cluster*)	int	gain of cell with max. energy
LKR_ECELLMAX(*cluster*)	float	energy of the highest energy cell
LKR_TIMERAW(*cluster*)	float	cluster time before t0 corr.
LKR_MCTAILCORR(*cluster*)	float	energy tail corr from MC
LKR_ANAVAR(*cluster*,5)	float	Provision for analysis variables
LKR_ANAFLAG(*cluster*,5)	int	Provision for analysis flags

J.33 KABhit

Variable	Type	Description
KAB_STRIP_HIT_TLEAD(*idet*,*istrip*,*hit*)	float	Leading time of hit = Q(PKAB+xxx+4)
KAB_STRIP_HIT_TTRAIL(*idet*,*istrip*,*hit*)	float	Trailing time of hit = Q(PKAB+xxx+5)
KAB_STRIP_HIT_TRACKID(*idet*,*istrip*,*hit*)	int	MC Track index (1 $\rightarrow$ *gen_track*) = Q(PKAB+xxx+1)

J.34 KABstrip

Variable	Type	Description
KAB_STRIP_NHIT(*idet,istrip*)	u_int	Number of hits in this strip

J.35 KABdet

Variable	Type	Description

J.36 KABtrack

Variable	Type	Description
KABTRK_PQ(*track*)	float	track mom.*charge
KABTRK_P(*track*)	float	track momentum
KABTRK_Q(*track*)	int	not FILLED
KABTRK_UPORDOWN(*track*)	int	first station UP=1 or DOWN=2
KABTRK_PERR(*track*)	float	error on momentum
KABTRK_CHI2(*track*)	float	χ^2 (quality) of track
KABTRK_X(*track*)	float	x position in second station
KABTRK_Y(*track*)	float	y position in second station
KABTRK_XUORD(*track*)	float	x position in UP or Down station
KABTRK_YUORD(*track*)	float	y position in UP or Down station
KABTRK_SIGXXUORD(*track*)	float	Error on xUorD
KABTRK_SIGYYUORD(*track*)	float	Error on yUorD
KABTRK_TIME(*track*)	float	track time
KABTRK_TIMEUORD(*track*)	float	track time at UP or Down station
KABTRK_TIMEST2(*track*)	float	track time at second station
KABTRK_DXDZ(*track*)	float	dx/dz
KABTRK_DYDZ(*track*)	float	dy/dz
KABTRK_RECFLAG(*track*)	int	reconstruction flag
KABTRK_SIGXX(*track*)	float	Error on x (cm)

Variable	Type	Description
KABTRK_SIGYY(*track*)	float	Error on y (cm)
KABTRK_SIGTT(*track*)	float	Error on time (ns)
KABTRK_SIGDXDX(*track*)	float	Error on dx/dz
KABTRK_SIGDYDY(*track*)	float	Error on dy/dz
KABTRK_SIGPY(*track*)	float	correlation sigma(p,y)
KABTRK_SIGXT(*track*)	float	correlation sigma(x,time)
KABTRK_SIGXDX(*track*)	float	correlation sigma(x,dx/dz)
KABTRK_SIGTDX(*track*)	float	correlation sigma(time,dx/dz)
KABTRK_SIGPX(*track*)	float	correlation sigma(p,x) not FILLED
KABTRK_SIGXY(*track*)	float	correlation sigma(x,y) not FILLED
KABTRK_SIGPT(*track*)	float	correlation sigma(p,time) not FILLED
KABTRK_SIGYT(*track*)	float	correlation sigma(y,time) not FILLED
KABTRK_SIGPDX(*track*)	float	correlation sigma(p,dx/dz) not FILLED
KABTRK_SIGYDX(*track*)	float	correlation sigma(y,dx/dz) not FILLED
KABTRK_SPAREINT(*track*,2)	int	2 spare integers
KABTRK_SPAREFLOAT(*track*,2)	float	2 spare floats
KABTRK_ANAVAR(*track*,20)	float	Provision for analysis variables
KABTRK_ANAFLAG(*track*,5)	int	Provision for analysis flags

J.37 KABFTWindow

Variable	Type	Description
KABFADC_CHANNEL_TWINDOW_MAXPULSEHEIGHT(0,i,i)	int	max pulseheight in this window

Variable	Type	Description
KABFADC_CHANNEL_TWINDOW_TFIRST(0,i,i)	float	time of 1st sample (w/r to timestamp)
KABFADC_CHANNEL_TWINDOW_TMAX(0,i,i)	float	time of max pulseheight in this window
KABFADC_CHANNEL_TWINDOW_NPULSEHEIGHT(0,i,i)	u_int	Number of pulseheight of sample
KABFADC_CHANNEL_TWINDOW_PULSEHEIGHT(0,i,i,300)	int	pulseheight of sample

J.38 KABFChannel

Variable	Type	Description
KABFADC_CHANNEL_ID(0,i)	int	channel number
KABFADC_CHANNEL_MODE(0,i)	int	sampling mode (time step size in ns (1/2)
KABFADC_CHANNEL_DECFLAG(0,i)	int	error code from dectag routine
KABFADC_CHANNEL_NTWINDOW(0,i)	u_int	Number of time windows in this channel

J.39 KABFADC

Variable	Type	Description
KABFADC_TIMESTAMP(0)	int	timestamp
KABFADC_OFFSET(0)	int	r/o offset for timestamp
KABFADC_NCHANNEL(0)	u_int	Number of FADC channels

J.40 NHOhit

Variable	Type	Description
NHO_COUNTER(hit)	int	counter number(1-32) = Q(RNHI+4)
NHO_X(hit)	float	x coord of hit(cm) = Q(RNHI+5)
NHO_Y(hit)	float	y coord of hit(cm) = Q(RNHI+6)
NHO_PHEIGHT(hit)	float	pulse height (from PDS) = Q(RNHI+9)

Variable	Type	Description
NHO_TIME(*hit*)	float	time of hit(ns) (all corr. added) = Q(RNHI+11)
NHO_UCTIME(*hit*)	float	uncorr. time of hit(ns) = Q(RNHI+10)
NHO_PDSFLAG(*hit*)	int	flag from PDS data structure = Q(RNHI+3)

J.41 HACcluster

Variable	Type	Description
HAC_ENERGY(*cluster*)	float	Energy = Q(RHCL+4)
HAC_BFRATIO(*cluster*)	float	back to front ratio of energy = Q(RHCL+5)
HAC_X(*cluster*)	float	x coord of cluster = Q(RHCL+7)
HAC_Y(*cluster*)	float	y coord of cluster = Q(RHCL+8)
HAC_RMSX(*cluster*)	float	cluster x-RMS(strips) = Q(RHCL+11)
HAC_RMSY(*cluster*)	float	cluster y-RMS(strips) = Q(RHCL+12)
HAC_EMAXX(*cluster*)	float	Max Ener. in strip at extrapolated track x = Q(RHCL+13)
HAC_EMAXY(*cluster*)	float	Max Ener. in strip at extrapolated track y= Q(RHCL+14)
HAC_TIME(*cluster*)	float	Cluster time = Q(RHCL+6)
HAC_RECFLAG(*cluster*)	int	Status bits = Q(RHCL+15) (cf NA48 note 98-4)

J.42 HACcell

Variable	Type	Description
PHAC_IDCELL(*icell*)	int	cell id = Q(PHAC+x+1)

Variable	Type	Description
PHAC_ENERGY(*icell*)	float	cell energy = Q(PHAC+x+3)

J.43 MUVhit

Variable	Type	Description
MUV_X(*hit*)	float	x position of hit = Q(RMUH+1)
MUV_Y(*hit*)	float	y position of hit = Q(RMUH+2)
MUV_TIME(*hit*)	float	time of hit = Q(RMUH+7)
MUV_DTIME(*hit*)	float	accuracy of the time = Q(RMUH+8)
MUV_STATUS(*hit*)	int	Number of hits in time with track
MUV_PLANE(*hit*,2)	int	Bit coded word for the planes used in the hit
MUV_CHI2(*hit*)	float	χ^2. Meaningful only if trackID != -1
MUV_TRACKID(*hit*)	int	Track index ($0 \rightarrow evt- > N\_track - 1$)
MUV_VERTEXID(*hit*)	int	Vertex index ($0 \rightarrow evt- > N\_track - 1$)

J.44 PMUVhit

Variable	Type	Description
PMUV_HIT_TIME(*chan*,*hit*)	float	time of hit = Q(PMUV+xxx+1)
PMUV_HIT_WIDTH(*chan*,*hit*)	float	time over threshold = Q(PMUV+xxx+2)
PMUV_HIT_INFO(*chan*,*hit*)	float	pulse status = Q(PMUV+xxx+3)

J.45 PMUVchannel

Variable	Type	Description
PMUV_N(*chan*)	int	channel number = Q(PMUV+x+1)
PMUV_NHIT(*chan*)	u_int	Number of hits in this channel

J.46 L3filter

Variable	Type	Description
FILTER_INDEX(*n*)	int	filter index
FILTER_INFO(*n*)	int	filter information

J.47 L3particle

Variable	Type	Description
PART_TYPE(*n*)	int	particle type

J.48 etaCluster

Variable	Type	Description
ETASUMM_CLU_FLAG(iclu)	int	0: not meson cand.; 1: meson cand.-Q(RETA+x+1)
ETASUMM_CLU_NCELL(iclu)	int	Ncell readout Q(RETA+x+2)
ETASUMM_CLU_IMAX(iclu)	int	Index of cell with max. energy Q(RETA+x+3)
ETASUMM_CLU_NREC(iclu)	int	Nb of cells with recov. pb in clu. rec. Q(RETA+x+4)
ETASUMM_CLU_ECELL(iclu,25)	float	E(GeV) each cell around clu. centre-Q(RETA+x+5+i)

J.49 etaSummary

Variable	Type	Description
ETASUMM_FLAG(0)	int	1=$\pi^0 \to \gamma\gamma$, 2=$\eta \to \gamma\gamma$, 3=$\eta \to 3\pi^0$,4: ch. =Q(RETA+1)
ETASUMM_XVTXZ(0)	float	Neutral vertex (from Ks target)=Q(RETA+2)
ETASUMM_XCOG(0)	float	Meson center of gravity=Q(RETA+3)
ETASUMM_EMESON(0)	float	Meson energy (GeV)=Q(RETA+4)
ETASUMM_NCLU(0)	u_int	Number of List of eta clusters

J.50 ananeut

Variable	Type	Description
ANEUT_ESC	float	Energy scale factor (input to user_sel2pi0)
ANEUT_IFLAG	int	1 for $2\pi^0$ cand., 2 for $3\pi^0$,0 if not, -1 no recbank
ANEUT_CEM(3)	float	$\pi^0\pi^0$ masses - sel2pi0/3pi0
ANEUT_INC(6)	int	clusters ids - sel2pi0/3pi0
ANEUT_RELLI	float	R ellipse - sel2pi0/3pi0
ANEUT_CTAU	float	c*tau (lifetimes) - sel2pi0/3pi0
ANEUT_EKAON	float	Kaon energy (GeV) - sel2pi0/3pi0
ANEUT_LKRTIME	float	time from LKR (ns) - LKRtime
ANEUT_LKRNHODTIME	float	time from LKR+NHOD (ns) - LKRtime
ANEUT_NTUSED	int	Nb of cells used to compute time - LKRtime
ANEUT_NHODTIME	float	time from NHOD - nhodotime

Variable	Type	Description
ANEUT_CUTS	int	To store bits for selection

J.51 anacharg

Variable	Type	Description
ACHARG_KTYPE	int	vertex type: 1 for Ks, 2 for Kl
ACHARG_IFLAG	int	bit0: 1 for $\pi^+\pi^-$ cand., 0 if not
ACHARG_IVERTEX	int	best vertex index
ACHARG_CUTS	int	bits describing cuts (newcharged)

J.52 anachodcount

Variable	Type	Description
ACHOD_ACOUNT_PLANE(*icount*)	float	Planes indices used
ACHOD_ACOUNT_COUNTER(*icount*)	float	Counters indices used
ACHOD_ACOUNT_TIME(*icount*)	float	Time for counters used
ACHOD_ACOUNT_SIG(*icount*)	float	sig

J.53 anacharghod

Variable	Type	Description
ACHOD_HODOTIME	float	Charged hod. event time
ACHOD_ITRACE(2)	int	tracks index to be used by hodotime
ACHOD_NACOUNT	u_int	Number of Counters used for hodotime
ACHOD_HSIGG	float	hsigg
ACHOD_VSIGG	float	vsigg

J.54 anaprothit

Variable	Type	Description
ATAG_APROT_PROTHIT_INDEXHIT(*proton*,*prothit*,2)	int	1st and Snd index of hit

J.55 anaproton

Variable	Type	Description
ATAG_APROT_PROTTIME(*proton*)	float	proton time
ATAG_APROT_PROTSTAT(*proton*)	int	proton code
ATAG_APROT_NPROTHIT(*proton*)	u_int	Number of hits belonging to proton

J.56 anatagger

Variable	Type	Description
ATAG_NAPROT	u_int	Number of identified protons
ATAG_NPROTLADDER	int	nprotLadder
ATAG_NPROTMONITOR	int	nprotMonitor

J.57 anaaks

Variable	Type	Description
AAKS_FLAG	int	see doc. 04-06-98 fuser_aksflag description

J.58 anacalled

Variable	Type	Description
ACALL_SELCHARGED	int	0: not; >0: called
ACALL_SEL2PI0	int	0: not; >0: called
ACALL_SEL3PI0	int	0: not; >0: called
ACALL_BLUEFIELD	int	0: not; 1: called
ACALL_LKRPEDCOR	int	0: not; 1: called
ACALL_LKRPOSCOR	int	0: not; 1: called
ACALL_LKRSHARING	int	0: not; 1: called
ACALL_HODOTIME	int	0: not; 1: called
ACALL_NHODTIME	int	0: not; 1: called
ACALL_LKRTIME	int	0: not; 1: called
ACALL_TAGTIME	int	0: not; 1: called
ACALL_AKSFLAG	int	0: not; 1: called

Variable	Type	Description
ACALL_MUON_REC	int	0: not; 1: called
ACALL_MUON_REJECT	int	0: not; 1: called
ACALL_GEOMCOR	int	0: not; 1: called
ACALL_MAGNETCOR	int	0: not; 1: called
ACALL_NEWCHARGED	int	0: not; 1: called
ACALL_MUON_TRACKREC	int	0: not; 1: called
ACALL_MUON_VTXREC	int	0: not; 1: called
ACALL_LKRCALCOR	int	0: not; 1: called
ACALL_LKRCALCOR1	int	0: not; 1: called
ACALL_LKRCALCOR2	int	0: not; 1: called
ACALL_LKRCALCOR3	int	0: not; 1: called
ACALL_LKRCALHI2K	int	0: not; 1: called
ACALL_SEL2GAM	int	0: not; >0: called
ACALL_SEL3PIC	int	0: not; 1: called
ACALL_SEL3PIN	int	0: not; 1: called

J.59 cmpEvent

Variable	Type	Description
EVT_NEVENT	int	trigger event number
EVT_TRIGWORD	int	trigger word (bits 16 and 17 for RaNDoMs)
EVT_TIMESTAMP	int	event timestamp
EVT_RECFLAG	int	reconstruction flags
EVT_FLAGCORR	int	1bit/analysis routine called
EVT_EVTFSPARE(10)	float	dummy spare floats
EVT_EVTISPARE(10)	int	dummy spare ints
EVT_DBERR	int	non-zero if error from SQLITE db
EVT_NTSTAMP	u_int	Number of time stamps!=evt->timestamp
EVT_MAINSPHASE	float	50Hz mains phase to 40MHz clock - from PSCA

Variable	Type	Description
EVT_SPSPHASE	float	SPS frequency phase to 40MHz clock - from PSCA
EVT_MAINSPHASERAW	int	raw mains phase
EVT_SPSPHASERAW	int	raw SPS phase
EVT_FEMAINSPHASE	float	mains phase (1st event method) - from PSCA
EVT_FESPSPHASE	float	SPS phase (1st event method) - from PSCA
EVT_NTRIG	u_int	Number of PMB pattern unit data
EVT_PATERRFLAG	int	Pattern unit error flag = IQ(PCAT+18)
EVT_PATSTATUS	int	Pattern unit status bits - NOT filled
EVT_TAGANTIC	int	anti-counter hit bits - NOT filled
EVT_NTAG	u_int	Number of Tagger channel data
EVT_TAGERRFLAG	int	Tagger error flag = Q(RTAG+3)
EVT_TAGSTATUS	int	Tagger status bits = Q(RTAG+1)
EVT_NKSM	u_int	Number of KS monitor hit data
EVT_KSMERRFLAG	int	KSM error flag = Q(PKSM+1)
EVT_KSMSTATUS	int	KSM status bits = Q(RANT+4)
EVT_AKLTIME	float	AKL event time = Q(RANT+2)
EVT_NAKL	u_int	Number of Anti-counter hit data
EVT_AKLERRFLAG	int	AKL error flag = Q(RANT+1)
EVT_AKLSTATUS	int	Anti-counter status bits = Q(RANT+4)
EVT_AKSMIPS(5)	float	NOT USED
EVT_AKSTIME(5)	float	NOT USED

Variable	**Type**	**Description**
EVT_AKSERRFLAG	int	AKS error flag - NOT filled
EVT_AKSSTATUS	int	AKS status flag (CB aks_rec_)
EVT_DCHBZ	float	z before magnet in DCHs=Q(RDCH+2)
EVT_DCHZ	float	z after magnet in DCHs=Q(RDCH+3)
EVT_DCHBZCORR	float	Corr. z bef. magnet (comp. in TrackVertexCorr)
EVT_DCHZCORR	float	Corr. z aft. magnet (comp. in TrackVertexCorr)
EVT_DCHNHITS(16)	int	Nhits/view×4 chambers (CB dchsac_cmplk_)
EVT_DCHOVRFLOW	int	overflow bits (CB dchsac_cmplk_)
EVT_BESTVERTEX	int	NOT used from 4.3.3
EVT_NDCHCLU	u_int	Number of Clusters in each view
EVT_NTRACK	u_int	Number of track structures
EVT_NTRACKCORR	u_int	Number of track structures after corr.
EVT_VTXFRRDVX	int	0: vtx built from compact; !=0: rec. from RDVX
EVT_NVERTEX	u_int	Number of vertex structures
EVT_NVERTEXCORR	u_int	Number of vertex structures after corr.
EVT_DCHERRFLAG	int	DCH error flag= Q(RDCH+5)
EVT_DCHERRFLAGPDS	int	DCH error flag from PDS = Q(PDCH+8)
EVT_DCHSTATUS	int	MBOX simu. bits - from RCAT or PU(ovl)

Variable	Type	Description
EVT_MBOXDEADTIME	int	MBOX deadtime ON (comp. in getMBOXdeadTime.c)
EVT_SGNTIME	int	DCH time reference
EVT_NSGNTRK	u_int	Number of track structures - NOT avail. from 4.3
EVT_NSGNVTX	u_int	Number of vertex structures - NOT avail. from 4.3
EVT_HODTIME	float	Event time (ns)= Q(RHOD+2)
EVT_NHOD	u_int	Number of hodoscope hit data
EVT_HODNEUTFLAG	int	set to 1 if Nhits was above limit
EVT_NHODNEUT	u_int	Number of hodoscope hit data
EVT_HODERRFLAG	int	Hodoscope error flag= Q(RHOD+1)
EVT_HODSTATUS	int	Hodoscope status bits= Q(RHOD+3)
EVT_LKRETOTCELL	float	Energy sum of cells (7^{th} samp.) (GeV) = Q(PLKR+7)
EVT_LKRENERGY	float	Clusters energy sum (GeV)=Q(RLKR+2)
EVT_NLKR	u_int	Number of LKR cluster data
EVT_LKRERRFLAG	int	LKR error flag= Q(RLKR+3)
EVT_LKRSTATUS	int	LKR status bits (set to zero)
EVT_LKRDOWNSCALED	int	LKR downscale flag (=1 downsc., =0 not downsc.)
EVT_NKABHIT	int	Total number of raw KABES hits

Variable	Type	Description
EVT_NKABTRK	u_int	Number of KABES track data
EVT_KABERRFLAG	int	KAB error flag= Q(RKAB+5)
EVT_KABERRFLAGPDS	int	KAB error flag from PDS = Q(PKAB+6)
EVT_NKABFADC	u_int	Number of KABES FADC data (filled only in the dedicated stream)
EVT_NUTHITMAP(4)	int	hit map (2 projections x 64 bits) - from PNUT
EVT_NUTXPEAK(4)	int	x peaks times=(-3→0)=IQ(PNUT+x+171)
EVT_NUTYPEAK(4)	int	y peaks times=(-3→0)=IQ(PNUT+x+172)
EVT_NUTNXPEAK(4)	int	x peaks (bef. LUT) (-3→ 0)=IQ(PNUT+x+20)
EVT_NUTNYPEAK(4)	int	y peaks (bef. LUT) (-3→ 0)=IQ(PNUT+x+21)
EVT_NUTM0(2)	int	0th moment (2 proj.)= IQ(PNUT+x+22,23)
EVT_NUTM1(2)	int	1st moment (2 proj.)= IQ(PNUT+x+24,25)
EVT_NUTM2(2)	int	2nd moment (2 proj.)= IQ(PNUT+x+26,27)
EVT_NUTM0SMP(6)	int	0th moment (sample -2,-1,0,+1,+2) - from PNUT
EVT_NUTM1SMP	int	1st moment (sample 0) - from PNUT
EVT_NUTM2SMP	int	2nd moment (sample 0) - from PNUT
EVT_NUTHACSMP	int	HAC (sample 0)= IQ(PNUT+x+170)-upto compact-4.3

Variable	Type	Description
EVT_NUTHACESUM(6)	int	HAC (sample -2,-1,0,+1,+2)= IQ(PNUT+x+170)-fr compact-4.4
EVT_NUTTS	int	Trigger control bits (sample 0) IQ(PNUT+x+19)
EVT_SPY_X(64)	int	spy chan.(X view)=IQ(PNUT+x+28+i)
EVT_SPY_Y(64)	int	spy chan.(Y view)=IQ(PNUT+x+92+i)
EVT_NUTERRFLAG	int	Neutral trigger error flag=IQ(PNUT+5)
EVT_NUTSTATUS	int	status bits - NOT filled
EVT_NHOTIME	float	Event time(ns)= Q(RNHO+2)
EVT_NNHO	u_int	Number of Neutral hod. hit data
EVT_NHOERRFLAG	int	Neutral hod. error flag= Q(RNHO+1)
EVT_NHOSTATUS	int	Neutral hod. status bits= Q(RNHO+3)
EVT_HACENERGY	float	Total energy in the HAC= Q(RHAC+2)
EVT_NGOODHAC	int	Nb of HAC clu. ass. to tracks=Q(RHAC+3)
EVT_NHAC	u_int	Number of HAC cluster data
EVT_HACERRFLAG	int	HAC error flag (set to 0)
EVT_HACSTATUS	int	HAC status bits (set to 0)
EVT_NPHAC	u_int	Number of HAC cell data
EVT_NMUV	u_int	Number of MUV hit data
EVT_NPMUV	u_int	Number of Muon veto channel PDS data
EVT_NPMUVNEW	u_int	Number of New Muon veto channel PDS data

Variable	Type	Description
EVT_MUVERRFLAG	int	MUV dec. status= Q(RMUV+3)
EVT_MUVSTATUS	int	MUV rec. status= Q(RMUV+1)
EVT_L3EVTYPE	int	Event type - NOT USED
EVT_L3STATUS	int	L3 status bits - NOT USED
EVT_L3ERRDEC	int	L3 flag for decoding errors
EVT_L3ERRREC	int	L3 flag for reconstruction errors
EVT_L3RTRSTATUS	int	real-time-rec. status bits - NOT USED
EVT_L3TRIGWORD(2)	int	Level 3 trigger words
EVT_L3ONLINETRIGWORD(2)	int	Level 3 trigger words at run time
EVT_L3BTRIGWORD(2)	int	Level 3 trigger words at repro
EVT_L3ACTIONDOWNSCALE	int	Downscaling flag for cuts per trigger in L3
EVT_L3ACTIONDOWNSCALE2	int	Downscaling flag for cuts per trigger in L3
EVT_L3FILTERDOWNSCALE	int	Downscaling flag for filter in L3
EVT_L3ACTIONOVL	int	Overlay flag per trigger in L3
EVT_L3ACTIONOVL2	int	Overlay flag per trigger in L3
EVT_L3FILTEROVL	int	Overlay flag per filter in L3
EVT_L3CUTFLAG	int	Flagged cuts in L3
EVT_L3CUTFLAG2(10)	int	Flagged cuts in L3
EVT_L3RAREDECAY	int	rare decay bits
EVT_L3EVSTEER	int	event steering information
EVT_NFILTER	u_int	Number of L3 filter information
EVT_NPART	u_int	Number of L3 particles

Variable	Type	Description
EVT_NTRHIST	u_int	Number of NOT USED
EVT_L3NCELLS	int	number of cells
EVT_L3NCELLABT	int	number of cells above threshold
EVT_L3RAWENERGY	float	raw LKR energy
EVT_L3DCHHITS(4)	int	number of DCH hits
EVT_L3EP	float	best vertex highest E/p
EVT_L3MINDIST	float	Minimum distance for E/p calculation
EVT_L3ZVERT	float	best vertex z-coord
EVT_L3CDA	float	best vertex closest distance of approach
EVT_L3PIPIMASS	float	2-track invariant mass for best vertex
EVT_L3PPAIR	float	2-track momentum for best vertex
EVT_L3COGC	float	momentum weighted cog for b vtx
EVT_L3TAU	float	proper lifetime of best vertex
EVT_L3NCLUS	int	number of LKR clusters
EVT_L3PI0PI0MASS(2)	float	two π^0 mass combinations
EVT_L3COGN	float	centre of gravity for neutrals
EVT_L2BSTATUS	int	L2B Bits 0-15:clust.,16:error,17:acc.,18:treat.
EVT_L2BCOMCLUST	int	L2B cluster combination
EVT_L2BSHADR(6)	int	L2B channel address of shower max.
EVT_L2BSHX(6)	int	L2B Shower x coord. ($\times$100)
EVT_L2BSHY(6)	int	L2B Shower y coord. ($\times$100)
EVT_L2BSHENERGY(6)	int	L2B Shower Energy ($\times$1000)
EVT_L2BSHTIME(6)	int	L2B Shower Time ($\times$10)
EVT_L2BSHXRMS(6)	int	L2B Shower x rms ($\times$100)

Variable	Type	Description
EVT_L2BSHYRMS(6)	int	L2B Shower y rms (×100)
EVT_L3SPARE(2)	float	spare L3 words
EVT_L3SHOWERWIDTH	float	Shower Width computed in L3 - NOT USED
EVT_STATUS	int	Overall event status bits
EVT_CMPFILTER	int	Compact Filter bits selection - NOT FILLED
EVT_NOVRFLWSIM	int	Number of simulated overflows
EVT_OVRFLWSIMBEF	float	overflow closest to 0, Before 0
EVT_OVRFLWSIMAFT	float	overflow closest to 0, After 0
EVT_NTRIGWORDL	u_int	Number of TrigWord (ALL evts in burst)
EVT_TRIGWORDL(70000)	int	TrigWord (ALL evts in burst)
EVT_NTIMESTAMPL	u_int	Number of TimeStamp (ALL evts in burst)
EVT_TIMESTAMPL(70000)	int	TimeStamp (ALL evts in burst)
EVT_NDCHDECERRORL	u_int	Number of DCH Dec. error (ALL evts in burst)
EVT_DCHDECERRORL(70000)	int	DCH Dec. error (ALL evts in burst)
EVT_NLKRHACDECERRORL	u_int	Number of LKR+HAC Dec. error (ALL evts in burst)
EVT_LKRHACDECERRORL(70000)	int	LKR+HAC Dec. error (ALL evts in burst)
EVT_NCHARGEDINFOL	u_int	Number of Charged Filter bits (ALL evts in burst)
EVT_CHARGEDINFOL(70000)	int	Charged Filter bits (ALL evts in burst)

Variable	Type	Description
EVT_NNEUTRALINFOL	u_int	Number of Neutral Filter bits (ALL evts in burst)
EVT_NEUTRALINFOL(70000)	int	Neutral Filter bits (ALL evts in burst)
EVT_NETASUMM	u_int	Number of struct. used for η runs
EVT_NTHMPACKEDDATA	u_int	Number of THM packed data
EVT_THMPACKEDDATA(90)	int	THM packed data
EVT_SPAREINT(10)	int	10 spare integers in case
EVT_SPARE01INT(10)	int	10 spare integers in case
EVT_SPAREFLOAT(10)	float	10 spare floats in case
EVT_SPARE01FLOAT(10)	float	10 spare floats in case
EVT_NTRIGBEF	int	Ntriggers in prev. 100ms-NOT YET FILLED
EVT_TIMETOPREV	int	time to prev. trigger(ns)-NOT YET FILLED

J.60 XoffDet

Variable	Type	Description
BUR_XOFFDET_DATA(3)	float	NOT used from 4.2
BUR_XOFFDET_NTRANSITION	float	Nb of XOFF transitions OFF $\rightarrow$ ON
BUR_XOFFDET_FRACTIMEON	float	Fraction of time XOFF was ON (rel. to XoffBurstLen)
BUR_XOFFDET_FIRSTTIMEON	float	First time in burst XOFF was ON (s)

J.61 NSstat

Variable	Type	Description
BUR_BEAMNS_(P42/K12)_BEND	int	0: OK; 1 out of toler. (magnets:1A,coll.:1mm

Variable	Type	Description
BUR_BEAMNS_(P42/K12)_QUAD	int	quad
BUR_BEAMNS_(P42/K12)_TRIM	int	trim
BUR_BEAMNS_(P42/K12)_SCRAPER	int	scraper
BUR_BEAMNS_(P42/K12)_COLL	int	coll
BUR_BEAMNS_(P42/K12)_COLR	int	colr

J.62 BeamNonStandard

Variable	Type	Description
BUR_BEAMNS_TARGT4	float	Intensity in T4 ($\times 10^{10} prot.$)
BUR_BEAMNS_TARGT10	float	Intensity in T10 ($\times 10^{10} prot.$)
BUR_BEAMNS_TARGKSY	float	KS target X position (mm)
BUR_BEAMNS_TARGKSX	float	KS target Y position (mm)
BUR_BEAMNS_TAXMOT7	float	Collimator postion (mm)
BUR_BEAMNS_TAXMOT8	float	Collimator postion (mm)
BUR_BEAMNS_TAXMOT17	float	Collimator postion (mm)
BUR_BEAMNS_TAXMOT18	float	Collimator postion (mm)
BUR_BEAMNS_T10COLLSTART	float	T10 Collimator positon
BUR_BEAMNS_T10COLLEND	float	T10 Collimator positon
BUR_BEAMNS_COLLPROTTAG	float	Collimator-proton tagging: 1(in)/0(out)
BUR_BEAMNS_COLLPROTTAGX	float	Coll.-proton tagging: x pos. (mm)
BUR_BEAMNS_COLLPROTTAGY	float	Coll.-proton tagging: y pos. (mm)
BUR_BEAMNS_COLLCLEANCOLL	float	Collimator: 1(in)/0(out)
BUR_BEAMNS_COLLCLEANCOLLX	float	Coll. : x pos. (mm)
BUR_BEAMNS_COLLCLEANCOLLY	float	Coll. : y pos. (mm)

Variable	Type	Description
BUR_BEAMNS_COLLDEFCOLL	float	Defining collimator: 1(in)/0(out)
BUR_BEAMNS_COLLDEFCOLLX	float	Defining coll.: x pos. (mm)
BUR_BEAMNS_COLLDEFCOLLY	float	Defining coll.: y pos. (mm)
BUR_BEAMNS_CRYDISHOR	float	Crystal x pos. (mm)
BUR_BEAMNS_CRYDISVER	float	Crystal y pos. (mm)
BUR_BEAMNS_CRYROTHOR	float	Crystal rot. in V. plane
BUR_BEAMNS_CRYROTVER	float	Crystal rot. in H. plane
BUR_BEAMNS_CRYTEMP	float	Crystal temperature
BUR_BEAMNS_XAKSDISTHOR	float	AKS x pos. (mm)
BUR_BEAMNS_XAKSROTHOR	float	AKS rot. in H. plane
BUR_BEAMNS_XAKSROTVER	float	AKS rot. in V. plane
BUR_BEAMNS_P42CONV	float	P42 converter: 1(in)/0(out)
BUR_BEAMNS_K12CONV	float	K12 converter: 1(in)/0(out)
BUR_BEAMNS_VACCPUMP55	float	Pressure
BUR_BEAMNS_VACCPUMP56	float	Pressure
BUR_BEAMNS_VACCPUMP98	float	Pressure
BUR_BEAMNS_VACCPUMP58	float	Pressure
BUR_BEAMNS_VACCVALVE56	float	Pressure
BUR_BEAMNS_VACCVALVE58	float	Pressure
BUR_BEAMNS_SPSBEND1K12	float	Current for bend 1 in K12 (K12-B1) (A). This is the Achromat 1
BUR_BEAMNS_SPSBEND2K12	float	Current for bend 2 in K12 (K12-B2) (A)
BUR_BEAMNS_SPSBEND3K12	float	Current for bend 3 in K12 (K12-B3) (A)
BUR_BEAMNS_SPSBEND4K12	float	Current for bend 4 in K12 (K12-B4) (A)
BUR_BEAMNS_SPSBEND5K12	float	Current for bend 5 in K12 (K12-B5) (A)
BUR_BEAMNS_P42FILENUMBER	float	File name
BUR_BEAMNS_K12FILENUMBER	float	File name

J.63 BeamStandard

Variable	Type	Description
BUR_BEAMS_SPSDATE	float	Date
BUR_BEAMS_SPSHOUR	float	Time
BUR_BEAMS_SPST4INT	float	Intensity on T4 ($10^{10} protons$)
BUR_BEAMS_SPST4SYM	float	Symmetry in T4 (<0 if ups. count. missing)
BUR_BEAMS_SPST10INT	float	Intensity in T10 ($10^{10} protons$)
BUR_BEAMS_SPST4	float	Nb of magnets with errors at T4
BUR_BEAMS_SPSK12	float	Nb of magnets with errors at K12
BUR_BEAMS_SPSP0	float	Nb of magnets with errors at P0
BUR_BEAMS_SPSMNP33COIL1	float	Current for coil1 of MNP33(K12-B6) (A)
BUR_BEAMS_SPSMNP33COIL2	float	Current for coil2 of MNP33(K12-B7) (A)
BUR_BEAMS_SPSBEND1K12	float	Current for bend 1 in K12 (K12-B1) (A)
BUR_BEAMS_SPSSTATUS	float	Machine Status Word
BUR_BEAMS_SPARE(4)	float	spare

J.64 TSscalars

Variable	Type	Description
BUR_TS_TSSCAL_NSCALER	u_int	Number of TS scalers
BUR_TS_TSSCAL_SCALER(100)	int	TS scalers

J.65 L2TSscalars

Variable	Type	Description
BUR_TS_L2TSSCAL_NSCALER	u_int	Number of L2 TS scalers
BUR_TS_L2TSSCAL_SCALER(20)	int	L2 TS scalers

J.66 AKLscalars

Variable	Type	Description
BUR_TS_AKLSCAL_AKL1OR	int	AKL: AKL1:OR scal.
BUR_TS_AKLSCAL_AKL2OR	int	AKL: AKL2:OR scal.
BUR_TS_AKLSCAL_AKL3OR	int	AKL: AKL3:OR scal.
BUR_TS_AKLSCAL_AKL4OR	int	AKL: AKL4:OR scal.
BUR_TS_AKLSCAL_AKL5OR	int	AKL: AKL5:OR scal.
BUR_TS_AKLSCAL_AKL6OR	int	AKL: AKL6:OR scal.
BUR_TS_AKLSCAL_AKL7OR	int	AKL: AKL7:OR scal.
BUR_TS_AKLSCAL_AKLOR	int	AKL: AKL :OR scal.
BUR_TS_AKLSCAL_AKL1ORAND	int	AKL: AKL1: OrAnd scal.
BUR_TS_AKLSCAL_AKL2ORAND	int	AKL: AKL2: OrAnd scal.
BUR_TS_AKLSCAL_AKL3ORAND	int	AKL: AKL3: OrAnd scal.
BUR_TS_AKLSCAL_AKL4ORAND	int	AKL: AKL4: OrAnd scal.
BUR_TS_AKLSCAL_AKL5ORAND	int	AKL: AKL5: OrAnd scal.
BUR_TS_AKLSCAL_AKL6ORAND	int	AKL: AKL6: OrAnd scal.
BUR_TS_AKLSCAL_AKL7ORAND	int	AKL: AKL7: OrAnd scal.
BUR_TS_AKLSCAL_AKLORAND	int	AKL: AKL : OrAnd scal.

J.67 TriggerSupervisor

Variable	Type	Description

J.68 BeamMonitor

Variable	Type	Description
BUR_BM_BADMAC	int	0: OK; > 0: bctr MAC had problem
BUR_BM_FASTSPILLKS	float	Eff. spill length seen by Delco-KS
BUR_BM_AVERAGEAKS	float	Average AKS intensity (Hz)

Variable	Type	Description
BUR_BM_MAXAKS	float	Maximum int. of AKS readings (Hz)
BUR_BM_RMSAKS	float	RMS int. of AKS readings (Hz)
BUR_BM_SLOWDCKS	float	Slow Duty Cycle-KS
BUR_BM_EXTRLENKS	float	Extraction Length-KS ($\in [0-1]$)
BUR_BM_FASTSPILLKL	float	Eff. spill length seen by Delco-KS
BUR_BM_AVERAGEBCTR	float	Average BCtr intensity (Hz)
BUR_BM_MAXBCTR	float	Maximum int. of BCtr readings (Hz)
BUR_BM_RMSBCTR	float	RMS int. of BCtr readings (Hz)
BUR_BM_SLOWDCKL	float	Slow Duty Cycle-KL
BUR_BM_EXTRLENKL	float	Extraction Length-KL ($\in [0-1]$)
BUR_BM_SPARE(3)	float	spares
BUR_BM_NSAMPLE	int	Nb of samples for vectors
BUR_BM_FCLOCK	int	Final clock
BUR_BM_FHAC1	int	Single part. trigger in HAC monitor
BUR_BM_FHAC2	int	≥ 2 part. in HAC (top$\times$bot + left$\times$right)
BUR_BM_FMUONS	int	Muon veto trigger
BUR_BM_FAKS	int	Final aks count
BUR_BM_FT1T2	int	Final t1 t2 count
BUR_BM_FQX	int	Final Qx count
BUR_BM_FBCTR	int	Final BCTR count
BUR_BM_FTWOMU	int	2 muons rate from MUV
BUR_BM_FHV1	int	BCtr downscaled
BUR_BM_FHVD1	int	BCtr downscaled further
BUR_BM_FHVD2	int	BCtr downscaled further
BUR_BM_FHVD3	int	BCtr downscaled further
BUR_BM_FHVD4	int	BCtr downscaled further
BUR_BM_FDELCO	int	Final Delco count

Variable	Type	Description
BUR_BM_FAKL	int	AKL total OR
BUR_BM_FKSMDELCO	int	Final ksmDelco count
BUR_BM_FKSM	int	Final KSM count
BUR_BM_CLOCK(400)	int	Clock counts (100kHz) bef. readings
BUR_BM_AKS(400)	int	AKS coincidence
BUR_BM_T1T2(400)	int	Tagger monitor counters (counts per interval)
BUR_BM_QX(400)	int	QX from charged hodoscope
BUR_BM_BCTR(400)	int	Direct beamcounter
BUR_BM_DELCO(400)	int	BCtr delayed self-coincidence (w=20ns, del=57ns)
BUR_BM_KSMDELCO(400)	int	KS target mon. delayed self-co (w=20ns del=??ns)
BUR_BM_KSM(400)	int	KS targer monitor

J.69 BeamMonitor01

Variable	Type	Description
BUR_BM01_BADMAC	int	0: OK; > 0: bctr MAC had problem
BUR_BM01_FASTSPILLKS	float	Eff. spill length seen by Delco-KS
BUR_BM01_AVERAGEAKS	float	Average AKS intensity (Hz)
BUR_BM01_MAXAKS	float	Maximum int. of AKS readings (Hz)
BUR_BM01_RMSAKS	float	RMS int. of AKS readings (Hz)
BUR_BM01_SLOWDCKS	float	Slow Duty Cycle-KS
BUR_BM01_EXTRLENKS	float	Extraction Length-KS ($\in [0-1]$)
BUR_BM01_FASTSPILLKL	float	Eff. spill length seen by Delco-KS
BUR_BM01_AVERAGEBCTR	float	Average BCtr intensity (Hz)

Variable	Type	Description
BUR_BM01_MAXBCTR	float	Maximum int. of BCtr readings (Hz)
BUR_BM01_RMSBCTR	float	RMS int. of BCtr readings (Hz)
BUR_BM01_SLOWDCKL	float	Slow Duty Cycle-KL
BUR_BM01_EXTRLENKL	float	Extraction Length-KL ($\in [0-1]$)
BUR_BM01_SPARE(3)	float	spares
BUR_BM01_NSAMPLE	int	Nb of samples for vectors
BUR_BM01_FCLOCK	int	Final clock
BUR_BM01_FHAC1	int	Single part. trigger in HAC monitor
BUR_BM01_FHAC2	int	$\geq$ 2 part. in HAC (top$\times$bot + left$\times$right)
BUR_BM01_FMUONS	int	Muon veto trigger
BUR_BM01_FAKS	int	Final aks count
BUR_BM01_FT1T2	int	Final t1 t2 count
BUR_BM01_FQX	int	Final Qx count
BUR_BM01_FBCTR	int	Final BCTR count
BUR_BM01_FTWOMU	int	2 muons rate from MUV
BUR_BM01_FHV1	int	BCtr downscaled
BUR_BM01_FHVD1	int	BCtr downscaled further
BUR_BM01_FHVD2	int	BCtr downscaled further
BUR_BM01_FHVD3	int	BCtr downscaled further
BUR_BM01_FHVD4	int	BCtr downscaled further
BUR_BM01_FDELCO	int	Final Delco count
BUR_BM01_FAKL	int	AKL total OR
BUR_BM01_FKSMDELCO	int	Final ksmDelco count
BUR_BM01_FKSM	int	Final KSM count
BUR_BM01_CLOCK(800)	int	Clock counts (100kHz) bef. readings
BUR_BM01_AKS(800)	int	AKS coincidence
BUR_BM01_T1T2(800)	int	Tagger monitor counters (counts per interval)
BUR_BM01_QX(800)	int	QX from charged hodoscope
BUR_BM01_BCTR(800)	int	Direct beamcounter

Variable	Type	Description
BUR_BM01_DELCO(800)	int	BCtr delayed self-coincidence (w=20ns, del=57ns)
BUR_BM01_KSMDELCO(800)	int	KS target mon. delayed self-co (w=20ns del=??ns)
BUR_BM01_KSM(800)	int	KS targer monitor

J.70 LkrCalib

Variable	Type	Description
BUR_LKRCALIB_TRIGGER(30)	int	trigger for calibration
BUR_LKRCALIB_DAC	float	DAC value

J.71 DataMerger

Variable	Type	Description
BUR_DM_ERROR	int	???
BUR_DM_DETERROR	int	???

J.72 TagClock

Variable	Type	Description
BUR_TAGCLK_PHASE	float	Clock0 Phase
BUR_TAGCLK_REFVOLT1	float	Reference Voltage
BUR_TAGCLK_REFVOLT2	float	Reference Voltage

J.73 MUVScalars

Variable	Type	Description
BUR_MUVSCAL_MUONS	int	μ's scal.
BUR_MUVSCAL_TWOMUONS	int	2 μ scal.

J.74 BadBurst

Variable	Type	Description
BUR_BADB_CALL	int	if >0: badburst routine called
BUR_BADB_SKIP	int	if >0 entire burst is skipped
BUR_BADB_LKR	int	LKR flag (comp. in fuser_badburst.F)
BUR_BADB_DCH	int	Dch flag (comp. in fuser_badburst.F)
BUR_BADB_NUT	int	Nut flag (comp. in fuser_badburst.F)
BUR_BADB_MBX	int	Mbx flag (comp. in fuser_badburst.F)
BUR_BADB_HAC	int	Hac flag (comp. in fuser_badburst.F)
BUR_BADB_TAG	int	Tag flag (comp. in fuser_badburst.F)
BUR_BADB_MUV	int	Muv flag (comp. in fuser_badburst.F)
BUR_BADB_HODC	int	Ch. Hod. flag (comp. in fuser_badburst.F)
BUR_BADB_HODN	int	Neutr. Hod. flag (comp. in fuser_badburst.F)
BUR_BADB_PMB	int	PMB flag (comp. in fuser_badburst.F)
BUR_BADB_AKS	int	AKS flag (comp. in fuser_badburst.F)
BUR_BADB_AKL	int	AKL flag (comp. in fuser_badburst.F)
BUR_BADB_CLK	int	Clock flag (comp. in fuser_badburst.F)
BUR_BADB_KSM	int	KSM flag (comp. in fuser_badburst.F)
BUR_BADB_KAB	int	Kabes flag (comp. in fuser_badburst.F)
BUR_BADB_NOEPS	int	Not a $\frac{\epsilon'}{\epsilon}$ burst

Variable	Type	Description
BUR_BADB_PHYS	int	Burst not good for physics (Kcharged)
BUR_BADB_SPARE1	int	spare
BUR_BADB_SPARE2	int	spare
BUR_BADB_SPARE3	int	spare
BUR_BADB_SPARE4	int	spare

J.75 TimeOffset

Variable	Type	Description
BUR_TOFFST_VERSION	float	Version of timin-offset file read-in
BUR_TOFFST_TAG	float	Tagger offset
BUR_TOFFST_AKS	float	Aks offset
BUR_TOFFST_AKL	float	Akl offset
BUR_TOFFST_HOD	float	Ch. hod. offset
BUR_TOFFST_NHO	float	Neutr. hod. offset
BUR_TOFFST_DCH	float	Dch offset
BUR_TOFFST_LKR	float	Lkr offset
BUR_TOFFST_HAC	float	Hac offset
BUR_TOFFST_MUV	float	Muv offset
BUR_TOFFST_LKRTAG	float	Lkr-Tag offset
BUR_TOFFST_LKRNHOD	float	Lkr-nhod offset
BUR_TOFFST_LKRAKL	float	Lkr-Akl offset
BUR_TOFFST_LKRHAC	float	Lkr-Hac offset
BUR_TOFFST_KABPLUS	float	Fine tuning of Kabes offset for K plus
BUR_TOFFST_KABMINUS	float	Fine tuning of Kabes offset for K Minus

J.76 superTimeOffset

This structure is used for compact supercompact and hypercompact bursts. Here the version for compact bursts is exemplified. In order to access time offsets from supercompact and hypercompact, you should substitute "BUR_" with "SBUR_" or "HBUR_" depending on which kind of data you are analyzing.

Variable	Type	Description
BUR_TOFFST_VERSION	float	Version of timin-offset file read-in
BUR_TOFFST_TAG	float	Tagger offset
BUR_TOFFST_AKS	float	Aks offset
BUR_TOFFST_KAB	float	Kabes offset
BUR_TOFFST_NMV	float	New Muon Veto offset
BUR_TOFFST_AKL	float	Akl offset
BUR_TOFFST_HOD	float	Ch. hod. offset
BUR_TOFFST_NHO	float	Neutr. hod. offset
BUR_TOFFST_DCH	float	Dch offset
BUR_TOFFST_LKR	float	Lkr offset
BUR_TOFFST_HAC	float	Hac offset
BUR_TOFFST_MUV	float	Muv offset
BUR_TOFFST_LKRTAG	float	Lkr-Tag offset
BUR_TOFFST_LKRNHOD	float	Lkr-nhod offset
BUR_TOFFST_LKRAKL	float	Lkr-Akl offset
BUR_TOFFST_LKRHAC	float	Lkr-Hac offset
BUR_TOFFST_KABPLUS	float	Fine tuning of Kabes offset for K plus
BUR_TOFFST_KABMINUS	float	Fine tuning of Kabes offset for K Minus

J.77 anatocall

Variable	Type	Description
BUR_TOCALL_SELCHARGED	int	0: not; >0: to call
BUR_TOCALL_SEL2PI0	int	0: not; >0: to call
BUR_TOCALL_SEL3PI0	int	0: not; >0: to call
BUR_TOCALL_BLUEFIELD	int	0: not; 1: to call
BUR_TOCALL_LKRPEDCOR	int	0: not; 1: to call
BUR_TOCALL_LKRPOSCOR	int	0: not; 1: to call
BUR_TOCALL_LKRSHARING	int	0: not; 1: to call
BUR_TOCALL_HODOTIME	int	0: not; 1: to call
BUR_TOCALL_NHODTIME	int	0: not; 1: to call
BUR_TOCALL_LKRTIME	int	0: not; 1: to call
BUR_TOCALL_TAGTIME	int	0: not; 1: to call

Variable	Type	Description
BUR_TOCALL_AKSFLAG	int	0: not; 1: to call
BUR_TOCALL_MUON_REC	int	0: not; 1: to call
BUR_TOCALL_MUON_REJECT	int	0: not; 1: to call
BUR_TOCALL_GEOMCOR	int	0: not; 1: to call
BUR_TOCALL_MAGNETCOR	int	0: not; 1: to call
BUR_TOCALL_NEWCHARGED	int	0: not; 1: to call
BUR_TOCALL_MUON_TRACKREC	int	0: not; 1: to call
BUR_TOCALL_MUON_VTXREC	int	0: not; 1: to call
BUR_TOCALL_LKRCALCOR	int	0: not; 1: to call
BUR_TOCALL_LKRCALCOR1	int	0: not; 1: to call
BUR_TOCALL_LKRCALCOR2	int	0: not; 1: to call
BUR_TOCALL_LKRCALCOR3	int	0: not; 1: to call
BUR_TOCALL_LKRCALHI2K	int	0: not; 1: to call
BUR_TOCALL_SEL2GAM	int	0: not; >0: to call

J.78 Burst

Variable	Type	Description
BUR_MAJORVER	int	COmPACT major version number
BUR_MINORVER	int	COmPACT minor version number
BUR_PATCH	int	COmPACT patch number
BUR_TIME	int	burst time
BUR_HEPDB(20)	int	Timestamps from HEPdb - not filled yet
BUR_BRTYPE	int	Type of burst (data/MC/..)
BUR_BMTYPE	int	Type of beam (K_S,K_L,K_S+K_L,....)
BUR_INTENSITY	float	number protons on T10
BUR_NRUN	int	run number
BUR_NEVENT	int	— Not filled
BUR_DATASET	int	Data set value
BUR_KSMSCA(32)	int	KS monitor scalers

Variable	Type	Description
BUR_NHOSCA(32)	int	Neutral hodoscope scalers
BUR_HODVSCA(64)	int	Hodoscope vertical scalers
BUR_HODHSCA(64)	int	Hodoscope horizontal scalers
BUR_LOGIC(128)	int	Logic scalers
BUR_SYPS(32)	int	Symmetric pre-trigger scaler
BUR_TSSCA(124)	int	NOT filled - back. compat.
BUR_XOFFBURSTLEN	float	Burst Length (s)
BUR_NTRIGWORD	u_int	Number of TrigWord (ALL evts in burst)
BUR_TRIGWORD(70000)	int	TrigWord (ALL evts in burst)
BUR_NTIMESTAMP	u_int	Number of TimeStamp (ALL evts in burst)
BUR_TIMESTAMP(70000)	int	TimeStamp (ALL evts in burst)
BUR_NDCHDECERROR	u_int	Number of DCH Dec. error (ALL evts in burst)
BUR_DCHDECERROR(70000)	int	DCH Dec. error (ALL evts in burst)
BUR_NLKRHACDECERROR	u_int	Number of LKR+HAC Dec. error (ALL evts in burst)
BUR_LKRHACDECERROR(70000)	int	LKR+HAC Dec. error (ALL evts in burst)
BUR_NCHARGEDINFO	u_int	Number of Charged Filter bits (ALL evts in burst)
BUR_CHARGEDINFO(70000)	int	Charged Filter bits (ALL evts in burst)
BUR_NNEUTRALINFO	u_int	Number of Neutral Filter bits (ALL evts in burst)

Variable	Type	Description
BUR_NEUTRALINFO(70000)	int	Neutral Filter bits (ALL evts in burst)
BUR_CHAMBERDZ(4)	float	Chamber Delta(z) w.r.t. to Geom file
BUR_CALLANAROUTINE	int	see text in user-guide
BUR_GETEOBBEFORE	int	NOT USED yet
BUR_DBERR	int	Database error code (0=OK;-1=NOK)

J.79 L3errorcounter

Variable	Type	Description
EOB_L3ERRCTR_ERROR	int	generic error
EOB_L3ERRCTR_DECEVT	int	event decoding errors
EOB_L3ERRCTR_DECTAG	int	tagger decoding errors
EOB_L3ERRCTR_DECAKS	int	AKS decoding errors
EOB_L3ERRCTR_DECAKL	int	anticounter decoding errors
EOB_L3ERRCTR_DECDCH	int	driftchamber decoding errors
EOB_L3ERRCTR_DECHOD	int	hodoscope decoding errors
EOB_L3ERRCTR_DECNHO	int	neutral hodoscope decoding errors
EOB_L3ERRCTR_DECLKR	int	liquid krypton decoding errors
EOB_L3ERRCTR_DECHAC	int	hadron calorimeter decoding errors
EOB_L3ERRCTR_DECMUV	int	muon veto decoding errors
EOB_L3ERRCTR_DECPAT	int	pattern unit decoding errors
EOB_L3ERRCTR_DECNUT	int	neutral trigger decoding errors
EOB_L3ERRCTR_DECL3	int	level 3 decoding errors
EOB_L3ERRCTR_RECTAG	int	tagger reconstruction errors
EOB_L3ERRCTR_RECAKS	int	AKS reconstruction errors

Variable	Type	Description
EOB_L3ERRCTR_RECAKL	int	AKL reconstruction errors
EOB_L3ERRCTR_RECDCH	int	driftchamber reconstruction errors
EOB_L3ERRCTR_RECHOD	int	hodoscope reconstruction errors
EOB_L3ERRCTR_RECNHO	int	neutral hodoscope reconstruction errors
EOB_L3ERRCTR_RECLKR	int	liquid krypton reconstruction errors
EOB_L3ERRCTR_RECHAC	int	hadron calorimeter reconstruction errors
EOB_L3ERRCTR_RECMUV	int	muon veto reconstruction errors
EOB_L3ERRCTR_RECBTH	int	both DCH reconstructions' errors
EOB_L3ERRCTR_COMDCH_NODCH	int	computation err. (no DCH info)
EOB_L3ERRCTR_COMEP_NODCH	int	comp. error on E/p (no DCH info)
EOB_L3ERRCTR_COMEP_NOLKR	int	comp. error on E/p (no LKR info)
EOB_L3ERRCTR_COMEP_DOUBLECLUS	int	comp. error on E/p (two clu.)
EOB_L3ERRCTR_COMEP_NOCLUS	int	comp. error on E/p (no clu.)
EOB_L3ERRCTR_COMEP_SAMECLUS	int	comp. error on E/p (2 tracks on clu.)
EOB_L3ERRCTR_COMMBOX_NODCH	int	COMMBOX error counters
EOB_L3ERRCTR_COMNUT_NOLKR	int	COMNUT error counters
EOB_L3ERRCTR_COML3B_NOLKR	int	COML3B error counters
EOB_L3ERRCTR_SPARE	int	spare variable for compatibility

J.80 decResult

Variable	Type	Description
EOB_AERRDEC_DECRES_CODE(*idec*)	int	Record code 0=master, 1=echan, 2=log
EOB_AERRDEC_DECRES_ECHAN(*idec*)	int	Error ID in call to errcountf()
EOB_AERRDEC_DECRES_LOG(*idec*)	int	Error log value in call to errcountf()
EOB_AERRDEC_DECRES_NERR(*idec*)	int	Number of errors
EOB_AERRDEC_DECRES_SATFLAG(*idec*)	int	Flag if log values truncated
EOB_AERRDEC_DECRES_EVTNO(*idec*)	int	if satflag=1,Event no. where sat. occurred
EOB_AERRDEC_DECRES_TS(*idec*)	int	if satflag=1,Timestamp where sat. occurred
EOB_AERRDEC_DECRES_X1(*idec*)	int	Internal variable (ask Giles)
EOB_AERRDEC_DECRES_X2(*idec*)	int	Internal variable (ask Giles)

J.81 anaerrdec

Variable	Type	Description
EOB_AERRDEC_NDECRES	u_int	Number of holds errors from decod. - (comp. in eobdec)

J.82 pisaCounter

Variable	Type	Description
EOB_PISAMON_MONXXX_COUNTS(iSample,64)	int	Number of counts per counter

J.83 pisaCounterSample

Variable	Type	Description
EOB_PISAMON_SAMPLE_CLOCK(*i*)	int	
EOB_PISAMON_SAMPLE_PHASE(*i*)	int	
EOB_PISAMON_SAMPLE_EFFSPILL(*i*,6)	int	
EOB_PISAMON_SAMPLE_MOM0_JURA(*i*)	float	
EOB_PISAMON_SAMPLE_MOM0_SALEVE(*i*)	float	

Variable	Type	Description
EOB_PISAMON_SAMPLE_MOM1_JURA(i,2)	float	
EOB_PISAMON_SAMPLE_MOM1_SALEVE(i,2)	float	
EOB_PISAMON_SAMPLE_MOM2_JURA(i,2)	float	
EOB_PISAMON_SAMPLE_MOM2_SALEVE(i,2)	float	

J.84 pisaMonitors

Variable	Type	Description
EOB_PISAMON_TIME	int	Burst time
EOB_PISAMON_NSAMPLE	u_int	Number of samples (7.2,2004)
EOB_PISAMON_MAINZ(16)	int	(new 7.2)
EOB_PISAMON_NMONJURA	u_int	Number of Jura counter samples (backw. compat.)
EOB_PISAMON_NMONSALEVE	u_int	Number of Saleve counter samples (backw. compat.)
EOB_PISAMON_NMONAUX	u_int	Number of Aux counter samples (backw. compat.)

J.85 EoBMagnet

Variable	Type	Description
EOB_MAGNETPROBES_TIME	int	time of measurement
EOB_MAGNETPROBES_HALLVSOB(6)	float	field values at start of burst
EOB_MAGNETPROBES_HALLTSOB(6)	float	probe temp at start of burst
EOB_MAGNETPROBES_HALLVEOB(6)	float	field values at end of burst
EOB_MAGNETPROBES_HALLTEOB(6)	float	probe temp at end of burst
EOB_MAGNETPROBES_VOLTMETERSOB	float	different probe type at sob
EOB_MAGNETPROBES_VOLTMETEREOB	float	different probe type at eob

J.86 EndofBurst

Variable	Type	Description
EOB_NEVENT	int	number of events in burst
EOB_NWRITE	int	number of events written to output
EOB_NKILL	int	number of events killed by cuts
EOB_STATUS	int	status bits of rec program
EOB_SGNNRESET(3)	int	number of reset vectors
EOB_SGNINEFF(384)	int	inefficiency for each group of 16 wires
EOB_SGNEFF(384)	int	efficiency for each group of 16 wires
EOB_SGNBERR(20)	int	number of r/o errors in burst
EOB_SGNBADBIT	int	plane with most readout errors
EOB_NEVENTLIST	int	Nb of events in the list
EOB_NTRIGWORD	u_int	Number of TrigWords (ALL evts in burst)
EOB_TRIGWORD(70000)	int	TrigWords (ALL evts in burst)
EOB_NTIMESTAMP	u_int	Number of TimeStamp (ALL evts in burst)
EOB_TIMESTAMP(70000)	int	TimeStamp (ALL evts in burst)
EOB_NDCHDECERROR	u_int	Number of DCH Dec. error (ALL evts in burst)
EOB_DCHDECERROR(70000)	int	DCH Dec. error (ALL evts in burst)
EOB_NLKRHACDECERROR	u_int	Number of LKR Dec. error (ALL evts in burst)
EOB_LKRHACDECERROR(70000)	int	LKR Dec. error (ALL evts in burst)

Variable	Type	Description
EOB_NCHARGEDINFO	u_int	Number of Charged Filter bits (ALL evts in burst)
EOB_CHARGEDINFO(70000)	int	Charged Filter bits (ALL evts in burst)
EOB_NNEUTRALINFO	u_int	Number of Neutral Filter bits (ALL evts in burst)
EOB_NEUTRALINFO(70000)	int	Neutral Filter bits (ALL evts in burst)
EOB_NPROCERROR	u_int	Number of Error List of original processing
EOB_PROCERROR(10000)	int	Error List of original processing
EOB_NPROCERRORREPRO	u_int	Number of Error List of reprocessing
EOB_PROCERRORREPRO(10000)	int	Error List of reprocessing
EOB_L3SPARE_INT(80)	int	spare space to insert extra variables!
EOB_L3SPARE_FLOAT(20)	float	spare space to insert extra variables!

J.87 ke3Event

Variable	Type	Description
KNEVENT	int	trigger event number
KTIMESTAMP	int	event timestamp
KECELL(49)	float	cell energies (7×7)
KFCELL(49)	int	cell flags (7×7)
KADC(45)	int	ADC counts ((3×3)×5)
KIXELEC	int	x index of cell hit by e
KIYELEC	int	y index of cell hit by e
KXCLUS	float	X coord of cluster centre
KYCLUS	float	Y coord of cluster centre
KEELEC	float	energy of electron
KPELEC	float	momentum of electron

Variable	**Type**	**Description**
KXELEC	float	extrapolated x position of e on LKR
KYELEC	float	extrapolated y position of e on LKR
KAPXELEC	float	angle of e wrt. LKr projectivity (x proj)
KAPYELEC	float	angle of e wrt. LKr projectivity (y proj)
KERAW	float	uncorrected cluster energy
KCELLSREAD	float	number of cells read out
KSPACHACORR	float	corr. applied for spacecharge effect=Q(RLCL+18)
KECORRKE3	float	cl. energy corrected with ke3 factor=Q(RLCL+19)
KE2SAMP7	float	energy in first smpl of 7*7 cells Q(RLCL+21)
KEHAC	float	HAC Clu. energy behind el.(GeV)=Q(LRKE3+13)
KDISTHAC	float	min. dist.(HAC clu.-elec track) (cm)=Q(LRKE3+14)
KNPMUV	int	number of hits PMUV Q(LRKE3+15)
KECLUNEAR	float	Closest cluster: E(GeV)=Q(LRLCL+4)
KIDCLUNEAR	int	Closest cluster: cell max id=Q(LRLCL+3)
KXCLUNEAR	float	Closest cluster: x(cm)=Q(LRCL+7)
KYCLUNEAR	float	Closest cluster: y(cm)=Q(LRCL+8)
KPPION	float	Pion: momentum(GeV)=Q(LRKE3+10)
KXPION	float	Pion: x(cm) (extrap. at LKR)=Q(LRKE3+11)

Variable	Type	Description
KYPION	float	Pion: y(cm) (extrap. at LKR)=Q(LRKE3+12)
KZVERTEX	float	z position of vertex

J.88 kmu3Event

Variable	Type	Description
KMU3_NNUT	u_int	Number of detailed NUT information for kum3 data

J.89 mcPlaneTrak

Variable	Type	Description
MPART_PLANE_ICODE(isim,ipart,ipl)	int	plane code - NOT USED from 4.0
MPART_PLANE_XYZ(isim,ipart,ipl,3)	float	NOT USED from 4.0
MPART_PLANE_ZPLANE(isim,ipart,ipl)	float	z of Plane = Q(LTRAK+x+4)
MPART_PLANE_DXDZ(isim,ipart,ipl)	float	dx/dz in plane= Q(LTRAK+x+5)
MPART_PLANE_DYDZ(isim,ipart,ipl)	float	dy/dz in plane= Q(LTRAK+x+6)
MPART_PLANE_EDEP(isim,ipart,ipl)	float	depos.Ener.= Q(LTRAK+x+7)
MPART_PLANE_ACCEPT(isim,ipart,ipl)	int	accept. flag= Q(LTRAK+x+9)

J.90 mcParticle

Variable	Type	Description
MPART_TYPE(isim,ipart)	int	particle type=Q(LPART+1)
MPART_P(isim,ipart,4)	float	four-mom.=Q(LPART+2,3,4,5)
MPART_PVERTEX(isim,ipart,3)	float	prod. vertex=Q(LPART+9,10,11)

Variable	Type	Description
MPART_DVERTEX(isim,ipart,3)	float	decay vertex=Q(LPART+12,13,14)
MPART_SPIN(isim,ipart,3)	float	spin of generated particle -x,y,z-
MPART_XBMAG(isim,ipart,3)	float	pos. bef. magnet - NOT USED
MPART_XAMAG(isim,ipart,3)	float	pos. aft. magnet - NOT USED
MPART_PBMAG(isim,ipart,3)	float	bef. magnet - NOT USED
MPART_PAMAG(isim,ipart,3)	float	mom. aft. magnet - NOT USED
MPART_ICODES(isim,ipart)	int	NOT USED from 4.0
MPART_NPLANE(isim,ipart)	u_int	Number of Traking planes

J.91 mcSIM

Variable	Type	Description
MSIM_MCTYPE(isim)	int	MC type (NMC/NASIM)=Q(LSIM+2)
MSIM_MCVERSION(isim)	int	version of MC used=Q(LSIM+3)
MSIM_GEOVERSION(isim)	int	version of geometry used=Q(LSIM+4)
MSIM_SHWVERSION(isim)	int	version of shower library used=Q(LSIM+5)
MSIM_SIMYEAR(isim)	int	data year simulated=Q(LSIM+1)
MSIM_ACCRATE(isim)	float	accidental rate=Q(LSIM+18)
MSIM_KSKLRATIO(isim)	float	K_S/K_L ratio=Q(LSIM+16)
MSIM_CNRATIO(isim)	float	charged/neutral decay ratio=Q(LSIM+17)
MSIM_RANDSEED(isim,3)	int	random seeds=Q(LSIM+7,8,9)
MSIM_MCWORD(isim,4)	int	Q(LSIM+10,11,12,13) (NASIM $\neq$ NMC)

Variable	Type	Description
MSIM_NASIM(isim,4)	int	NOT USED from 4.0
MSIM_TIME(isim)	int	event time=Q(LEVT+6)
MSIM_MAGNETSIM(isim)	int	magnet simulation used=Q(LSIM+6)
MSIM_OPTIONS(isim)	int	MC options=Q(LSIM+15)
MSIM_NPART(isim)	u_int	Number of 'true' particle data

J.92 mcEvent

Variable	Type	Description
MNMCSIM	u_int	Number of SIM structure

J.93 DETstatus

Variable	Type	Description
SDETSTATUS_AKL	int	AKL bit coded status
SDETSTATUS_DCH	int	DCH bit coded status
SDETSTATUS_HOD	int	CHOD bit coded status
SDETSTATUS_HAC	int	HAC bit coded status
SDETSTATUS_LKR	int	LKR bit coded status
SDETSTATUS_KAB	int	KAB bit coded status
SDETSTATUS_NHO	int	NHOD bit coded status
SDETSTATUS_MUV	int	MUV bit coded status
SDETSTATUS_MBX	int	MBX bit coded status
SDETSTATUS_NTR	int	NTR bit coded status
SDETSTATUS_LV3	int	L3 filter bits (0-15: neutral, 16-31: charged)
SDETSTATUS_LV3TRIG	int	L3trigword[0](bits 0-15) + L3FilterDownScale(bits 16-31)
SDETSTATUS_LV3TRIGRARE	int	L3trigword[1](bits 0-31)

Variable	Type	Description
SDETSTATUS_LV3ABTRIG	int	L3ONLINEtrigword[0](bits 0-15)+L3Btrigword[0](bits 16-31)
SDETSTATUS_LV3ATRIGRARE	int	L3ONLINEtrigword[1](bits 0-31)
SDETSTATUS_LV3BTRIGRARE	int	L3Btrigword[1](bits 0-31)
SDETSTATUS_CHTREFF(10)	int	Charged Trigger Efficiency

J.94 KABstrack

Variable	Type	Description
SKABTRAK_P(*track*)	float	track momentum
SKABTRAK_Q(*track*)	int	track charge
SKABTRAK_UPORDOWN(*track*)	int	first station UP=1 or DOWN=2
SKABTRAK_PERR(*track*)	float	error on momentum
SKABTRAK_CHI2(*track*)	float	χ^2 (quality) of track
SKABTRAK_X(*track*)	float	x position in second station
SKABTRAK_Y(*track*)	float	y position in second station
SKABTRAK_XUORD(*track*)	float	x position in UP or Down station
SKABTRAK_YUORD(*track*)	float	y position in UP or Down station
SKABTRAK_TIME(*track*)	float	track time
SKABTRAK_TIMEUORD(*track*)	float	track time at UP or Down station
SKABTRAK_TIMEST2(*track*)	float	track time at second station
SKABTRAK_DXDZ(*track*)	float	dx/dz
SKABTRAK_DYDZ(*track*)	float	dy/dz
SKABTRAK_SIGXX(*track*)	float	Squared error on x (cm^2)
SKABTRAK_SIGYY(*track*)	float	Squared error on y (cm^2)

Variable	Type	Description
SKABTRAK_SIGTT(*track*)	float	Squared error on time (ns^2)
SKABTRAK_SIGDXDX(*track*)	float	Squared error on dx/dz
SKABTRAK_SIGDYDY(*track*)	float	Squared error on dx/dz
SKABTRAK_RECFLAG(*track*)	int	reconstruction flag
SKABTRAK_ANAVAR(*track*,20)	float	Provision for analysis variables
SKABTRAK_ANAFLAG(*track*,5)	int	Provision for analysis flags

J.95 trak

Variable	Type	Description
STRACK_P(*track*)	float	momentum
STRACK_Q(*track*)	int	charge
STRACK_QUALITY(*track*)	float	track quality
STRACK_CHI2(*track*)	float	track chi2
STRACK_BX(*track*)	float	x-coordinate before magnet
STRACK_BY(*track*)	float	y-coordinate before magnet
STRACK_BDXDZ(*track*)	float	dx/dz before magnet
STRACK_BDYDZ(*track*)	float	dy/dz before magnet
STRACK_X(*track*)	float	x-coordinate after magnet
STRACK_Y(*track*)	float	y-coordinate after magnet
STRACK_DXDZ(*track*)	float	dx/dz after magnet
STRACK_DYDZ(*track*)	float	dy/dz after magnet
STRACK_PERR(*track*)	float	Error on track momentum
STRACK_SIGXX(*track*)	float	Error on x (cm)=Q(RDTK+11)
STRACK_SIGYY(*track*)	float	Error on y (cm)=Q(RDTK+12)
STRACK_SIGDXDX(*track*)	float	Error on dx/dz =Q(RDTK+13)

Variable	Type	Description
STRACK_SIGDYDY(*track*)	float	Error on dy/dz =Q(RDTK+14)
STRACK_SIGXDX(*track*)	float	NOT FILLED - correlation sigma(x,dx/dz)=Q(RDTK+28)
STRACK_SIGXY(*track*)	float	correlation sigma(x,y)=Q(RDTK+29)
STRACK_SIGDXY(*track*)	float	correlation sigma(dx/dz,y)=Q(RDTK+30)
STRACK_SIGXDY(*track*)	float	correlation sigma(x,dy/dz)=Q(RDTK+31)
STRACK_SIGDXDY(*track*)	float	correlation sigma(dx/dz,dy/dz)=Q(RDTK+32)
STRACK_SIGYDY(*track*)	float	correlation sigma(y,dy/dz) =Q(RDTK+33)
STRACK_TIME(*track*)	float	track time
STRACK_HODTIME(*track*)	float	track time in hodoscope
STRACK_HODSTATUS(*track*)	int	hod. flags used for time calc.=Q(RDTK+26)
STRACK_AKLTIME(*track*)	float	nearest akl hit around cluster time (see doc)
STRACK_AKLTIME67(*track*)	float	time of AKL hit closest to event time (see doc)
STRACK_AKLFLAG(*track*)	int	records th nuber of hit in various timeslices
STRACK_IMUON(*track*)	int	index of assciated muon (-1. if none)
STRACK_IHAC(*track*)	int	associated HAC cluster
STRACK_ICLUS(*track*)	int	assoc. LKR clu. $0 \leq index < Ncluster$ (-1 if none)
STRACK_DDEADCELL(*track*)	float	distance to nearest LKR dead cell
STRACK_HITPATTERN(*track*)	int	one bit per wire for efficiency studies
STRACK_EFFICIENCY(*track*,2)	int	bit coded words for eff. studies

Variable	Type	Description
STRACK_SPARE(*track*,5)	float	spare variables

J.96 cluster

Variable	Type	Description
SCLUSTER_ENERGY(*iclus*)	float	cluster energy
SCLUSTER_X(*iclus*)	float	x-coordinate of cluster
SCLUSTER_Y(*iclus*)	float	y-coordinate of cluster
SCLUSTER_TIME(*iclus*)	float	cluster time
SCLUSTER_HACTIME(*iclus*)	float	nearest hac hit around cluster time
SCLUSTER_HODTIME(*iclus*)	float	nearest ch. hodoscope hit around cluster time
SCLUSTER_AKLTIME(*iclus*)	float	nearest akl hit around cluster time (see doc)
SCLUSTER_AKLTIME67(*iclus*)	float	time of AKL hit closest to event time (see doc)
SCLUSTER_AKLFLAG(*iclus*)	int	number of hits in various time slices
SCLUSTER_DDEADCELL(*iclus*)	float	distance to closest dead cell
SCLUSTER_STATUS(*iclus*)	int	status bits
SCLUSTER_ITRACK(*iclus*)	int	index of associated track (-1 if none)
SCLUSTER_RMSX(*iclus*)	float	x cluster width (lkr-¿rmsx) - NOT filled for ε'
SCLUSTER_RMSY(*iclus*)	float	y cluster width (lkr-¿rmsy) - NOT filled for ε'
SCLUSTER_MCTAILCORR(*iclus*)	float	energy tail corr from MC
SCLUSTER_SPARE(*iclus*,5)	float	spare variables

J.97 vtxtracks

Variable	Type	Description
SVTX_VTXTRACKS_ITRACK(*ivertex,itrack)*	int	index of track
SVTX_VTXTRACKS_BDXDZ(*ivertex,itrack)*	float	corrected dxdz before the magnet at the vertex

Variable	Type	Description
SVTX_VTXTRACKS_BDYDZ(*ivertex,itrack)*	float	corrected dydz before the magnet at the vertex

J.98 SCvertex

Variable	Type	Description
SVTX_NVTXTRACK(*ivertex*)	u_int	Number of track data in this vertex
SVTX_CHARGE(*ivertex*)	int	total charge of this vertex
SVTX_CDA(*ivertex*)	float	closest approach
SVTX_COG(*ivertex*)	float	cog (cm)
SVTX_CHI2(*ivertex*)	float	χ^2 of vertex
SVTX_X(*ivertex*)	float	x position
SVTX_Y(*ivertex*)	float	y position
SVTX_Z(*ivertex*)	float	z position
SVTX_TVTX(*ivertex*)	float	vertex time from spectrometer times
SVTX_TVTXHODO(*ivertex*)	float	Vertex time defined by the hodoscope
SVTX_HACTIME(*ivertex*)	float	closest hit in the hac around tVtx
SVTX_AKLTIME(*ivertex*)	float	closest hit of AkL around tVtx
SVTX_AKLTIME67(*ivertex*)	float	closest hit of AkL around tVtx (see doc)
SVTX_AKLFLAG(*ivertex*)	int	aklflag as defined in fuser_akl.F. It is used for both aklTime and aklTime67
SVTX_ANAVAR(*ivertex*,5)	float	Provision for analysis variables
SVTX_ANAFLAG(*ivertex*,5)	int	Provision for analysis flags

J.99 muon

Variable	Type	Description
SMUON_X(*imuon*)	float	x position
SMUON_Y(*imuon*)	float	y position
SMUON_TIME(*imuon*)	float	time from muon detector
SMUON_CHI2(*imuon*)	float	chi2 from the muon reconstruction
SMUON_STATUS(*imuon*)	int	coded for hit planes
SMUON_PLANE(*imuon*,2)	int	bit coded for hit map
SMUON_ITRK(*imuon*)	int	track index

J.100 pmuon

Variable	Type	Description
SPMUON_TIME(*ipmuon*)	float	time of the photomultiplier channel.
SPMUON_CHANNEL(*ipmuon*)	int	photo multiplier channel

J.101 hacclus

Variable	Type	Description
SHACCLUS _ENERGY(*ihacclus*)	float	energy of cluster
SHACCLUS _X(*ihacclus*)	float	x position of cluster
SHACCLUS _Y(*ihacclus*)	float	y position of cluster
SHACCLUS _TIME(*ihacclus*)	float	time of cluster
SHACCLUS _BFRATIO(*ihacclus*)	float	back to front ratio

J.102 SCDCHmult

Variable	Type	Description
SSCDCHEFFMULT_MBXPLANEEFF(6)	int	number of hit for plane packed
SSCDCHEFFMULT_L1TRK24EFF(2)	int	multiplicity in DCH1 for L1 trigger packed
SSCDCHEFFMULT_DCHSNOWERR(2)	int	16 bit for DCH snow effect(2-1)(4-3)
SSCDCHEFFMULT_MBXMULT	int	multiplicity in DCH
SSCDCHEFFMULT_TRK24ON	int	L1Trk24=1 –¿ trigger OK

J.103 PUtslice

Variable	Type	Description
SPU_CHAN(*tslice*,16)	int	16 channels * 24 bits x 10 tslice

J.104 MaChit

Variable	Type	Description
SMCH_COUNTER(*hit*)	int	(ONLY MaC evs) counter number(1-8) Q(PKSM+x+2)
SMCH_PHEIGHT(*hit*)	float	(ONLY MaC evs) pulse height(GeV) = Q(PKSM+x+3)
SMCH_TIME(*hit*)	float	(ONLY MaC evs) time of hit(ns) = Q(PKSM+x+4)
SMCH_PDSFLAG(*hit*)	int	(ONLY MaC evs) decoding flag=Q(PKSM+x+1)

J.105 SRNDMsummary

Variable	Type	Description
SRNDM_TYPE	int	type of random used(KS=1, KL=2) = Q(LPRE+1)
SRNDM_TIMESTAMP	int	timestamp of random used = Q(LPRE+3)
SRNDM_RUN	int	Run Number of random used = Q(LPRE+9)
SRNDM_BURST	int	Burst time stamp of random used = Q(LPRE+10)
SRNDM_NUSED	int	Number of times this RNDM evt was used so far
SRNDM_SPSPHASE	float	SPS phase of random evt

Variable	Type	Description
SRNDM_MAINPHASE	float	main(50Hz) phase of random evt
SRNDM_TOVRFLW(64)	float	time of nearest ovflw to rndm evt
SRNDM_KLMONDNDT	float	intensity in KLmon for random event
SRNDM_KSMONDNDT	float	intensity in KSmon for random event
SRNDM_QXDNDT	float	intensity in QX for random event

J.106 superCmpEvent

Variable	Type	Description
SEVT_DBERR	int	non-zero if error from SQLITE db
SEVT_CMPSTATUS	int	COmPACT analysis bit coded status
SEVT_FLAGCORR	int	which corrections were applied to this event
SEVT_CMPFILTER	int	which filter was applied to compact event
SEVT_NEVT	int	trigger event number
SEVT_TRIGWORD	int	trigger word
SEVT_TIMESTAMP	int	time stamp
SEVT_NTRIGBEF	int	number of triggers in interval preceding this event
SEVT_TIMETOPREV	int	time to previous event
SEVT_SPSPHASE	float	SPS phase to 40MHz clock
SEVT_MAINSPHASE	float	Mains 50Hz phase to 40MHz clock
SEVT_SPSPHASERAW	int	raw SPS phase
SEVT_MAINSPHASERAW	int	raw mains phase
SEVT_KLMONDNDT	float	intensity in KLmon
SEVT_QXDNDT	float	intensity in QX
SEVT_NMCH	u_int	Number of Mannelli counter hit data

Variable	Type	Description
SEVT_LKRDOWNSCALED	int	LKR downscale flag (=1 downsc., =0 not downsc.)
SEVT_LKRENERGY	float	total energy in LKR
SEVT_HACENERGY	float	total energy in HAC
SEVT_NKABSTRAK	u_int	Number of charged object - Kabes tracks
SEVT_NTRACK	u_int	Number of charged object - tracks
SEVT_NVTX	u_int	Number of charged object -vertices
SEVT_NMUON	u_int	Number of charged object - muon
SEVT_NPMUON	u_int	Number of photomultiplier muon hits
SEVT_NHACCLUS	u_int	Number of HAC clusters
SEVT_NCLUSTER	u_int	Number of LKR objects - clusters
SEVT_TSPREV	int	previous trigger timestamp
SEVT_TSNEXT	int	next trigger timestamp
SEVT_TWPREV	int	previous trigger trigger word
SEVT_TWNEXT	int	next trigger trigger word
SEVT_SPAREINT(2)	int	2 spare integers
SEVT_SPAREFLOAT(2)	float	2 spare floats

J.107 STSscal

Variable	Type	Description
SBUR_TSSCAL_NSCALER	u_int	Number of TS scalers
SBUR_TSSCAL_SCALER(100)	int	TS scalers

J.108 SL2TSscal

Variable	Type	Description
SBUR_L2TSSCAL_NSCALER	u_int	Number of L2 TS scalers

Variable	Type	Description
SBUR_L2TSSCAL_SCALER(20)	int	L2 TS scalers

J.109 superBurst

Variable	Type	Description
SBUR_MAJORVER	int	COmPACT major version number
SBUR_MINORVER	int	COmPACT minor version number
SBUR_PATCH	int	COmPACT patch number
SBUR_TIME	int	burst time
SBUR_BRTYPE	int	Type of burst (data/MC/..)
SBUR_NRUN	int	run number
SBUR_NTRIGWORD	int	Number of trig word (all evts in burst)
SBUR_INTENSITY	int	number of protons on T10
SBUR_INTENSITYT4	int	number of protons on T4
SBUR_CTAUORIGIN	int	z(mm) used to compute $c.\tau$
SBUR_TOFFSTVER	int	tim. offset version used to prod. SC
SBUR_MNP33CURRENT	int	current of the spectrometer magnet
SBUR_BEND1CURRENT	int	current of achromat 1
SBUR_FLAGS	int	Flags for turning on and of correction routines
SBUR_DATASET	int	Data set value
SBUR_DBERR	int	Database error code (0=OK;-1=NOK)

J.110 superPisaCounter

Variable	Type	Description
SEOB_SPISAMON_SMONXXX_AVERAGE(64)	float	average over 10 samples for each counter

Variable	Type	Description
SEOB_SPISAMON_SMONXXX_STDDEV(64)	float	standard deviation over 10 samples for each counter

J.111 superPisaMonitors

Variable	Type	Description
SEOB_SPISAMON_JURACOUNTS(64)	int	sum of Jura counter information
SEOB_SPISAMON_SALEVECOUNTS(64)	int	sum of Saleve counter information
SEOB_SPISAMON_AUXCOUNTS(64)	int	sum of Aux counter information

J.112 superPisaMonitors04

Variable	Type	Description
SEOB_PISAMON_TIME	int	Burst time
SEOB_PISAMON_NSAMPLE	u_int	Number of samples, same as eob (7.2,2004)
SEOB_PISAMON_MAINZ(16)	int	(new 7.2)

J.113 superEndofBurst

Variable	Type	Description
SEOB_MAGIC	int	Magic number
SEOB_DCHPLANEHIT(32)	int	inefficiency for each group of 16 wires
SEOB_DCHCARDHIT(512)	int	efficiency for each group of 16 wires

J.114 superMcParticle

Variable	Type	Description
SMCEVENT_PART_TYPE(ipart)	int	particle type=Q(LPART+1)

Variable	Type	Description
SMCEVENT_PART_P(ipart,4)	float	four-mom.=Q(LPART+2,3,4,5) (E,px,py,pz)
SMCEVENT_PART_PVERTEX(ipart,3)	float	prod. vertex=Q(LPART+9,10,11) (x,y,z)
SMCEVENT_PART_DVERTEX(ipart,3)	float	decay vertex=Q(LPART+12,13,14) (x,y,z)
SMCEVENT_PART_SPIN(ipart,3)	float	spin of generated particle =Q(LPART+) (x,y,z)

J.115 superMcDecay

Variable	Type	Description
SMCEVENT_DECAY_KTYPE	int	Kaon type
SMCEVENT_DECAY_DTYPE	int	Decay type
SMCEVENT_DECAY_DVERTEX(3)	float	Decay vertex (x,y,z)
SMCEVENT_DECAY_P(4)	float	Kaon four-mom. (E,px,py,pz)

J.116 superMcEvent

Variable	Type	Description
SMCEVENT_NPART	u_int	Number of For each particle

J.117 hyperBurst

Variable	Type	Description
HBUR_MAJORVER	int	COmPACT major version number
HBUR_MINORVER	int	COmPACT minor version number
HBUR_TIME	int	burst time
HBUR_BRTYPE	int	Type of burst (data/MC/..)

Variable	Type	Description
HBUR_NRUN	int	run number
HBUR_SAMPLE	int	sample number
HBUR_FLAG	int	burst status flag
HBUR_SPARE(2)	float	2 spare floats

J.118 hyperKabTrk

Variable	Type	Description
HKABTRK_P(*iktrk*)	float	track momentum
HKABTRK_Q(*iktrk*)	int	track charge
HKABTRK_UPORDOWN(*iktrk*)	int	first station UP=1 or DOWN=2
HKABTRK_PERR(*iktrk*)	float	error on momentum
HKABTRK_CHI2(*iktrk*)	float	χ^2 (quality) of track
HKABTRK_X(*iktrk*)	float	x position in second station
HKABTRK_Y(*iktrk*)	float	y position in second station
HKABTRK_YUORD(*iktrk*)	float	y position in UP or Down station
HKABTRK_TIME(*iktrk*)	float	track time
HKABTRK_DXDZ(*iktrk*)	float	dx/dz
HKABTRK_DYDZ(*iktrk*)	float	dy/dz

J.119 hyperCluster

Variable	Type	Description
HCLUSTER__ENERGY(*iclus*)	float	energy
HCLUSTER__X(*iclus*)	float	X position
HCLUSTER__Y(*iclus*)	float	Y position
HCLUSTER__TIME (*iclus*)	float	time
HCLUSTER__RMSX(*iclus*)	float	RMS X
HCLUSTER__RMSY(*iclus*)	float	RMS Y
HCLUSTER__DDEADCELL(*iclus*)	float	distance to nearest dead cell
HCLUSTER__DTRACK(*iclus*)	float	dist from closest track
HCLUSTER__FLAG(*iclus*)	int	flag

Variable	Type	Description
HCLUSTER__SPARE(*iclus*,2)	float	2 spare floats

J.120 hyperTrack

Variable	Type	Description
HTRACK__Q(*itrack*)	int	charge
HTRACK__P(*itrack*)	float	momentum
HTRACK__PERR(*itrack*)	float	momentum error
HTRACK__TIME(*itrack*)	float	track time form dch
HTRACK__HODOTIME(*itrack*)	float	track time from hodoscope
HTRACK__BX(*itrack*)	float	x position before the magnet
HTRACK__BY(*itrack*)	float	y position before the magnet
HTRACK__X(*itrack*)	float	x position after the magnet
HTRACK__Y(*itrack*)	float	y position after the magnet
HTRACK__BDXDZ(*itrack*)	float	dx/dz before magnet
HTRACK__BDYDZ(*itrack*)	float	dy/dz before magnet
HTRACK__NBFBDXDZ(*itrack*)	float	dx/dz before magnet w/o bluefield
HTRACK__NBFBDYDZ(*itrack*)	float	dy/dz after magnet w/o bluefield
HTRACK__DXDZ(*itrack*)	float	dx/dz after magnet
HTRACK__DYDZ(*itrack*)	float	dy/dz after magnet
HTRACK__EOP(*itrack*)	float	E over p for track
HTRACK__FLAG(*itrack*)	int	flags
HTRACK__SPARE(*itrack*,2)	float	2 spare floats

J.121 hyperVertex

Variable	Type	Description
HVTX__Q(*vtx*)	int	sum of charge
HVTX__MASS(*vtx*)	float	invariant kaon mass

Variable	Type	Description
HVTX__P(*vtx*)	float	momentum of kaon
HVTX__CDA(*vtx*)	float	Closest distance of apporach
HVTX__POS(*vtx*,3)	float	x,y,z of vertex
HVTX__SPARE(*vtx*,2)	float	2 spare floats

J.122 hyperMcParticle

Variable	Type	Description
HPART__TYPE(ipart)	int	particle type (NASIM)
HPART__P(ipart,4)	float	four-mom. (E,px,py,pz)
HPART__PVERTEX(ipart,3)	float	prod. vertex (x,y,z)
HPART__DVERTEX(ipart,3)	float	decay vertex (x,y,z)

J.123 hyperCmpEvent

Variable	Type	Description
HEVT_DBERR	int	non-zero if error from SQLITE db
HEVT_FLAG	int	flag bit 0 on neutral, charged otherwise
HEVT_TIMESTAMP	int	time stamp
HEVT_SPSPHASE	float	sps phase
HEVT_MAINSPHASE	float	mains phase
HEVT_TRIGWORD	int	trigger word
HEVT_PUPACK(3)	int	packed Pattern Units bits (access with ...)
HEVT_PU04(3)	int	pattern unit 4, 3 time slots (not filled from in 7.1
HEVT_PU14(3)	int	pattern unit 14, 3 time slots (not filled from in 7.1
HEVT_PU06(3)	int	pattern unit 6, 3time slots (not filled in 7.1)

Variable	**Type**	**Description**
HEVT_NCLUSTER	u_int	Number of cluster info. Not filled for charged evts.
HEVT_NTRACK	u_int	Number of hyper tracks
HEVT_NMCPART	u_int	Number of MC generated particles
HEVT_SPARE(2)	float	2 spare floats

J.124 GeomPhaSpa

Variable	Type	Description
GEOM_BEAMK(L/S)_X	float	phase space dispersion *10**3: x
GEOM_BEAMK(L/S)_Y	float	phase space dispersion *10**3: y
GEOM_BEAMK(L/S)_DX	float	phase space dispersion *10**3: dx
GEOM_BEAMK(L/S)_DY	float	phase space dispersion *10**3: dy
GEOM_BEAMK(L/S)_ANGLE(2)	float	beam angle: [0]: absolute; [1]: relative

J.125 GeomPos1

Variable	Type	Description
GEOM_X_X	float	x position in cm
GEOM_X_Y	float	y position in cm
GEOM_X_Z	float	z position in cm
GEOM_X_R	float	Radius in cm
GEOM_X_L	float	toto

J.126 GeomPos2

Variable	Type	Description
GEOM_DETX_X	float	x position in cm
GEOM_DETX_Y	float	y position in cm
GEOM_DETX_Z	float	z position in cm
GEOM_DETX_RIN	float	Rin in cm
GEOM_DETX_OUTEROCTAGON(2)	float	Outer Octagon (square) 2 values
GEOM_DETX_THICKNESS	float	Thickness in

J.127 GeomPos3

Variable	Type	Description
GEOM_X_X	float	x position in cm
GEOM_X_Y	float	y position in cm

Variable	Type	Description
GEOM_X_Z	float	z position in cm
GEOM_X_ANGLE	float	Rotation

J.128 GeomPosAkl

Variable	Type	Description
GEOM_AKLPOCKET_X(ipock)	float	x position in cm
GEOM_AKLPOCKET_Y(ipock)	float	y position in cm
GEOM_AKLPOCKET_Z(ipock)	float	z position in cm
GEOM_AKLPOCKET_INNEROCTAGON(ipock,2)	float	Inner Octagon
GEOM_AKLPOCKET_OUTEROCTAGON(ipock,2)	float	Outer Octagon

J.129 GeomPosStrip

Variable	Type	Description
GEOM_KAB_STRIP_X(ikab,istrip)	float	x position in cm
GEOM_KAB_STRIP_Y(ikab,istrip)	float	y position in cm
GEOM_KAB_STRIP_Z(ikab,istrip)	float	z position in cm

J.130 GeomKabes

Variable	Type	Description

J.131 GeomPosPlane

Variable	Type	Description
GEOM_DCH_PLANE_X(ich,iplane)	float	x position in cm
GEOM_DCH_PLANE_Y(ich,iplane)	float	y position in cm
GEOM_DCH_PLANE_Z(ich,iplane)	float	z position in cm
GEOM_DCH_PLANE_X1(ich,iplane)	float	x1 position in cm
GEOM_DCH_PLANE_ANGLE(ich,iplane)	float	angle of plane (in degrees)

J.132 GeomChamber

Variable	Type	Description

J.133 GeomDecay

Variable	Type	Description
GEOM_DECAY_ZNEUTRAL	float	Begining of neutral decay region
GEOM_DECAY_ZCHARGED	float	Begining of charged decay region

J.134 GeomVirtualZ

Variable	Type	Description
GEOM_DET_Z	float	Virtual point after the magnet
GEOM_DET_BZ	float	Virtual point before the magnet

J.135 GeomCompact

Variable	Type	Description
GEOM_AKSCONVTHICK	float	AKS converter thickness

This is a commented version of the *cdb* header file `cdbmap.h` defining the run and burst variables. These are the names to be used for reading back these variables in COmPACT. A more detailed description is given in NA48 note 2000-11 on “The cdb COmPACT quality database”.

In compact the run structure is called rdb; data are accessed by `rdb->nBur` for instance. In compact the burst structure is called bdb; data are accessed by `bdb->time` for instance.

```
/*---------------------------------------------------------*/
/* Macros to allow f77 access to data from CDB.            */
/*                                                         */
/*                                     Bruce Hay 8.12.98  */
/*---------------------------------------------------------*/
/*---------------------------------------------------------*/
/* Run structures:                                         */
/*---------------------------------------------------------*/

 RDB_DUMMY

/* General info: ---------------------------------------*/

 RDB_NBUR        /* no. bursts */
 RDB_NBURBAD     /* no. of bad bursts */
 RDB_TYPBAD      /* type of bad bursts */

/* Good events: ----------------------------------------*/

 RDB_NKSPIPI     /* no. of Ks Pi+Pi- */
 RDB_NKLPIPI     /* no. of Kl Pi+Pi- */
 RDB_NKSPI0PI0   /* no. of Ks Pi0Pi0 */
 RDB_NKLPI0PI0   /* no. of Kl Pi0Pi0 */

/* Charged performance: --------------------------------*/

 RDB_OVFRAND     /* % overflow in random events */
 RDB_OVPI0PI0    /* % overflow in Pi0Pi0 events */
 RDB_MASSKS      /* Ks->Pi+Pi- mass */
 RDB_MERRKS      /* Ks->Pi+Pi- mass resolution */
 RDB_RESPT2      /* Pt2 resolution */
 RDB_RESBKC      /* residual background estimator */
 RDB_MBXDEAD     /* Massbox deadtime */
 RDB_TEFFC       /* trigger efficiency */

/* Neutral performance: --------------------------------*/
```

```
 RDB_MASSPI0     /* Pi0->gg mass */
 RDB_MERRPI0     /* Pi0->gg mass resolution */
 RDB_RESELL      /* Rellipse resolution */
 RDB_RESBKN      /* residual background estimator */
 RDB_TEFFN       /* trigger efficiency */

/* Tagger: ---------------------------------------------*/

 RDB_TAGEFF      /* tagger efficiency */
 RDB_TAGDIL      /* tagger dilution */
 RDB_MULTTS      /* % multiple time-stamp events */

/* Beam: -----------------------------------------------*/

 RDB_TOTT10      /* intensity: total # p on T10 */

/*-----------------------------------------------------*/
/* Burst structures:                                   */
/*-----------------------------------------------------*/

/* General info: ---------------------------------------*/

 BDB_TIME          /* burst time: 32-bit Unix time */
 BDB_BADBUR        /* bad burst: 1 bit/system */
 BDB_NEVT          /* no. events in L3 list */
 BDB_INTT10        /* intensity: ppp on T10 */

/* Charged chain: --------------------------------------*/

 BDB_NWDOG        /* no. watchdogs in L3 list */
 BDB_QX2SCAL      /* Qx/2 scaler */
 BDB_ETOTSCAL     /* Etot scaler */
 BDB_NL1PIPI      /* no. L1PiPi in L3 list */
 BDB_TSFL1PIPI    /* first charged time-stamp: L1PiPi */
 BDB_TSLL1PIPI    /* last  charged time-stamp: L1PiPi */
 BDB_MBXSCAL      /* Massbox trigger scaler */
 BDB_NMBX         /* no. Massbox triggers in L3 list */
 BDB_NFILTCH      /* no. of L3 filtered events */
 BDB_NGOODCH      /* no. of "good" charged events */
 BDB_TSFMBX       /* first charged time-stamp: MBX */
 BDB_TSLMBX       /* last  charged time-stamp: MBX */

/* Neutral chain: --------------------------------------*/

 BDB_NNHOD        /* no. NHOD in L3 list */
 BDB_TSFNHOD      /* first neutral time-stamp: NHOD */
 BDB_TSLNHOD      /* last  neutral time-stamp: NHOD */
 BDB_PI0PI0SCAL   /* Pi0Pi0 trigger scaler */
 BDB_NPI0PI0      /* no. Pi0Pi0 triggers in L3 list */
 BDB_NFILTNE      /* no. of L3 filtered events */
 BDB_NGOODNE      /* no. of "good" neutral events */
 BDB_TSFPI0PI0    /* first neutral time-stamp: Pi0Pi0 */
 BDB_TSLPI0PI0    /* last  neutral time-stamp: Pi0Pi0 */

/* Dedicated variables: --------------------------------*/

 BDB_DCHERR(x)     /* DCH readout errors */
 BDB_MBXDEAD       /* % Massbox deadtime */
 BDB_MBXILL        /* % Massbox illegal deadtime */
 BDB_MULTTS        /* % multiple time-stamp events */
 BDB_NHITSTRACK    /* average no. hits/track */
 BDB_MBXNOTRACK    /* % Massbox events with no track */
 BDB_XOFF_LKR      /* % Lkr Xoff time */
 BDB_XOFF_DCH      /* % Dch Xoff time */
 BDB_XOFF_NUT      /* % Nut Xoff time */
 BDB_XOFF_HAC      /* % Hac Xoff time */
 BDB_XOFF_AKL      /* % Akl Xoff time */
 BDB_XOFF_TAG      /* % Tag Xoff time */
 BDB_XOFF_MUV      /* % Muv Xoff time */
 BDB_XOFF_HOD      /* % Hod Xoff time */

/* Marco's variables: ----------------------------------*/

 BDB_BCORR(x)     /* beam correlation variables ??? */
 BDB_MULT(x)      /* detector multiplicities ??? */
 BDB_STAT(x)      /* Wilcoxon and Kolmogorov-Smirnov */
                  /* burst statistics test variables */

/* PMB variable: ---------------------------------------*/

 BDB_PMBERR       /* PMB error bit-packed word: bits 00-05 */

/* Lkr file summary variables: -------------------------*/

 BDB_SUMMSTAT     /* summary status */
 BDB_LKRDT1       /* (logfile burst time - FIC time) */
 BDB_LKRDT2       /* (logfile burst time - matched burst time) */
 BDB_NEVLKR       /* number of events seen by Lkr */

/* errcompact variables: ------------------------------*/
```

```
 BDB_NMIXEVT      /* number of mixed events */
 BDB_NIBWCBAD     /* number of IB/wordcount mismatches */
 BDB_NEVTNUMBAD   /* number of local/global event number mismatches */
 BDB_NEMPTYPDCH   /* number of empty list of hits from PDCH */
 BDB_NDECPLKRBAD  /* number of decoding problems in PLKR bank */
 BDB_NACCPLKRBAD  /* number of problems to access PLKR in CALREAD */
 BDB_NTOOMANYCELLS /* number of times too many cells in CALREAD */
 BDB_NDCMISSTHR   /* number of times DC missed being above threshold */
 BDB_NDCFALSETHR  /* number of times DC falsely above threshold */
 BDB_NCTSSNO      /* ctss: no : different M - que? */
 BDB_NCTSSYES     /* ctss: yes: different M - que? */
 BDB_NERRCNT

/* DCH inefficiency variable: -------------------------*/

 BDB_DCHINEFF     /* bit-packed (00-31) DCH plane > ineff threshold */

/* new DCH eff. variable from Eddy 02-6-99 ------------*/

 BDB_DCHEFF       /* drift chamber efficiency variable */
 BDB_DCHEFF2

/* tagger efficiency */

 BDB_BST_TAGEFF   /* tagger efficiency per burst */

/* 2 new Raphael variables 19-02-99 -------------------*/

 BDB_OVFNEUT      /* % of overflows in neutral events */ # NPIOOF
 BDB_FRACKS       /* % of Ks in good vertex events */
```

K Compact up-dates for 2007 analysis

Please note that the changes described in the next section on the compact structure are applied only on the production(s) coming from reprocessing and not in the production (goldcmp34) coming from L3 during 2007 data taking.

K.1 Compact structure

Only the contents of two variables are changed with respect to the previous (7.2) version. In the HODneuthit structure (charged hodoscope structure for all the hits, including the hits not associated to a track) <u>now</u>:

HODneuthit

Variable	**Type**	**Description**
HODNEUT_PLANE(hit)	int	Flag from PDS data
HODNEUT_COUNTER(hit)	int	Scintillator number (1-128)

K.2 SuperCompact structure

For the 2007 run the KABstrack, HACclus and MaChit structures, given that the corresponding sub-detectors where not present during 2007 data taking, were used to store other quantities. Then please take in mind that for the 2007 run the data included on such supercompact variables are meaningless for the user.
In the following please take care of the variables type, some quantities usually stored on integer variables are now stored on floating variables.

The following variables are added to the main structure superCmpEvent:

superCmpEvent

Variable	**Type**	**Description**
SEVT_NHOD	int	Number of Charged Hod. hit data
SEVT_HODERRFLAG	int	Charged Hod. error flag
SEVT_HODSTATUS	int	Charged Hod. status bits
SEVT_NNHO	int	Number of Neutral Hod. hit data
SEVT_NHOERRFLAG	int	Neutral Hod. error flag
SEVT_NHOSTATUS	int	Neutral Hod. status bits
SEVT_NHODNEUT	int	Number of Charged Hod. hit data (all)
SEVT_HODNEUTFLAG	float	Set to 1 if Nhits was above limit

The following variables are added to the Lkr's cluster structure:

cluster

Variable	Type	Description
SCLUSTER_TLATCELL(iclus)	float	Time of most energetic lateral cell (ns)
SCLUSTER_UCENERGY(iclus)	float	Uncorrected cluster energy (GeV)
SCLUSTER_SPACHACORR(iclus)	float	Correction applied for spacecharge effect
SCLUSTER_E77(iclus)	float	Cluster energy in 7×7 cells (GeV)
SCLUSTER_ECELLMAX(iclus)	float	Energy of the cell with highest energy (GeV)
SCLUSTER_E2SAMPALL(iclus)	float	Energy from the first two time slots (pedestal) of all cluster cells (GeV)
SCLUSTER_GAINMAX(iclus)	int	Gain of the cell with highest energy

Then already from 2003 in a spare variable of the superCompact cluster structure are coded two integer variables:

SCLUSTER_SPARE(iclus,2)=imax(iclus)+1000000*cellsread(iclus)

where imax is the index of the cell with highest energy in the cluster and cellsread is the number of cells read for each cluster.
A way to extract the two variables with FORTRAN code, once cellsread and imax are declared integer, is the following:

```
cellsread(iclus)=SCLUSTER_SPARE(iclus,2)/1000000
imax(iclus)=mod(SCLUSTER_SPARE(iclus,2),1000000)
```

The following new superCompact structures are then added:

HODhit

Variable	**Type**	**Description**
SHOD_PHEIGHT(hit)	float	Pulse height (ADC counts)
SHOD_COUNTER(hit)	int	Scintillator number (1-128)
SHOD_PDSFLAG(hit)	int	Flag from PDS data
SHOD_TIME(hit)	float	Time of hit (ns)
SHOD_X(hit)	float	x coord. of DCH track (cm)
SHOD_Y(hit)	float	y coord. of DCH track (cm)

NHOhit

Variable	**Type**	**Description**
SNHO_PHEIGHT(hit)	float	Pulse height (ADC counts)
SNHO_COUNTER(hit)	float	Counter number (1-32)
SNHO_PDSFLAG(hit)	int	Flag from PDS data
SNHO_TIME(hit)	float	Time of hit (ns)

HODneuthit

Variable	**Type**	**Description**
SHODNEUT_PHEIGHT(hit)	float	Pulse height (ADC counts)
SHODNEUT_COUNTER(hit)	float	Scintillator number (1-128)
SHODNEUT_PDSFLAG(hit)	float	Flag from PDS data
SHODNEUT_TIME(hit)	float	Time of hit (ns)