```
JADE COMPUTER NOTE 68                        THIS IS JADEPR.TEXT(NOTE68)
=========================

P. STEFFEN, 83/09/21


              THE JADE CALIBRATION SCHEME
             ==============================


The jade calibration constants are available to  off  line  programs  in
COMMON /CALIBR/. THE constants are  updated  whenever  KALIBR/KLREAD  is
called (for each run). As a  consequence  one  has  always  the  current
constants in /CALIBR/ available, which are relevant to the event  beeing
processed.
The different sets of constants are on the disk-file 'F11LHO.AUPDAT1'
(, or on the two files 'F11LHO.BUPDAT0', 'F11LHO.BUPDAT1').
The status and the recent changes of the calibration files are described
in 'F11PST.LHOLIB.S(#CALNEWS)'.


A.      Structure of COMMON /CALIBR/.
---------------------------------------

  1.    COMMON /CALIBR/ ACALIB(1000)
                        DIMENSION HCALIB(2000), ICALIB(1000)
                        EQUIVALENCE (ACALIB(1),ICALIB(1),HCALIB(1))
        The actual length of /CALIBR/ words is set in the
                                             SUBROUTINE KLREAD.

  2.    The first 100 locations are foreseen for pointers and
        administration:
        IDATA( 1): POINTER TO MUCA-constants
        IDATA( 2): POINTER TO LGMA-constants
        IDATA( 3): POINTER TO TAGS-constants
        IDATA( 4): POINTER TO JTPL-constants
        IDATA( 5): POINTER TO JTAB-constants
        IDATA( 6): POINTER TO TOFC-constants
        IDATA( 7): POINTER TO LGST-constants
        IDATA( 8): POINTER TO DEDX-constants
        IDATA( 9): POINTER TO SPTG-constants
        IDATA(10): POINTER TO RVTX-constants
        IDATA(11): POINTER TO RCON-constants
        IDATA(12): POINTER TO TAGF-constants

        e.g. IPVRVTX = IDATA(10)        : pointer to run-vertex coordinates
             XV      = ADATA(IPRVTX+1): 1. constant  =  x(vertex)

        If the constants consist of half-words one has
        e.g. IPJTPL = IDATA( 4)*2      : pointer to jet chamber constants
             IT0    = HDATA(IPJTPL+1) : 1. constant  =  T0(1.wire)

  3.    It is JADE convention that the MUCA-constants are the  first  set
        of constants and they always start at ADATA(100).

  4.    The following different sets of constants are at present
        available:
        MUCA : Mu-chamber constants                H. McCann    (Man.)
        LGMA : lead glass constants                M. Minowa    (Tok.)
        TAGS : tagging constants (obsolete)        H. Wriedt    (Lan.)
        JTPL : jet chamber wire constants          R.D. Heuer   (Hei.)
        JTAB : jet chamber cell constants          P. Steffen/J.Spitzer
                                                                (Hei.)
        TOFC : time of flight constants            B. Naroska   (Desy)
        LGST : lead glass "spinning blocks"        M. Minowa    (Tok.)
        DEDX : dE/dx calibration constants         S. Bethke    (Hei.)
        SPTG : tagging "spinning blocks"(obsolete) H. Wriedt    (Lan.)
        RCON : not jet established
        RVTX : run dependent event vertex          S. Komamiya  (Hei.)
```

```
        TAGF : tagging constants (1982 ...)        A. Finch     (Lan.)


B.      Structure of the Calibration Files
-------------------------------------------

  1.    Calibration data for different periods are stored on the
        files:
              F11LHO.BUPDAT0 : constants up to run 10 000
              F11LHO.BUPDAT1 : constants from run 10 000on
        A compressed version of both files is  on  F11LHO.AUPDAT1,  which
        contains no LGST and SPTG constants ("spinning block" constants).
        This file is commonly used. The constants, which  are  left  out,
        are used in general only in the REDUC1-step.

        A record  on  the  UPDAT-file  may  contain  a  complete  set  of
        constants as well as  update  values  for  a  limited  number  of
        constants.

  2.    The record of the calibration file have the folllowing format:
        1. word       : LENGTH = number of following words
        2. word     )
        3.          )
        :           ) : data
        :           )
        (length+1)  )

        The records can be read and written with the statements
        DIMENSION IBUF(2009)
        READ (22) LENGTH, (IBUF(I1),I1=1,LENGTH)
        WRITE(22) LENGTH, (IBUF(I1),I1=1,LENGTH)

  3.    The first record on a calibration file has only one data-word:
        LENGTH  = 1
        IBUF(1) = time at which the data on the calibration file
                  start to become valid.
        According to this time the KLREAD-subroutine selects the
        calibration file (e.g. BUPDAT0 or BUPDAT1).

  4.    The following records have a LENGTH >= 9.
        The first 9 data words  contain  a  header. The  following  words
        contain the calibration constants:
        IBUF( 1)  = name of constants (e.g. MUCA, DEDX, see A.4.)
        IBUF( 2)  = current number of records for the same set of const.
        IBUF( 3)  = total   number of records for the same set of const.
        IBUF( 4)  = unused
        IBUF( 5)  = time at which the constants have been established
        IBUF( 6)  = time from which on the constants are valid
        IBUF( 7)  = 0, if record contains updates of selected words
                       within a set of constants.
                  = I, if record contains updates of consecutive words
                       within a set of constants;
                       I = 1.location within the set of constants
                           which shall be replaced by the constants of
                           the record.
                    Exemples are given in part B.7.
        IBUF( 8)  = run number from which on the constants are valid
        IBUF( 9)  = unused
        IBUF(10)  = data
        IBUF(11)  =  ...
         ...
         ...

  5.    The last record has LENGTH = 9 and consists only of a header.
        The time (IBUF(6)) and the run number (IBUF(8)) are set to a
        very large value.

  6.    The records are limited in lenghth to 2000 words. A longer set
        of constants is split in two or more records.
        E.g. a set of 3000 constants(full words)
             1. record: LENGTH = 1509
```

```
                  words 1-9   : header (word 7(1.location) =     1)
                  words 10...: constants     1 - 1500
          2. record: LENGTH = 1509
                  words 1-9   : header (word 7(1.location) = 1501)
                  words 10...: constants 1501 - 3000
     Or a set of 6000 constants(half words)
          1. record: LENGTH = 1509
                  words 1-9   : header (word 7(1.location) =     1)
                  words 10...: constants     1 - 3000
          2. record: LENGTH = 1509
                  words 1-9   : header (word 7(1.location) = 3001)
                  words 10...: constants 3001 - 6000

 7.  Updates of subsets of constants.
     There is a complete set of constants for the different periods of
     data taking. Within a period it is in general only necessary to
     update a subset of the complete set of constants. There are two
     possibility for replacing such a subset of constants.
     Exemples:
     1. Replace a subset of 6 consecutive constants starting with
        the 27th-word of the complete set of constants:
        record: word 7     = 27
                word 10-15 = constants to replace
                        words 27-32 of the complete set.
     2. Replace the 4th, 26th and 138th constant(full words)
          record: word 7     = 0
                  word 10    =  4 : location within the complete set
                                    to be replaced
                  word 11    = replacement of   4th constant
                  word 12    =  26
                  word 13    = replacement of  26th constant
                  word 14    = 138
                  word 15    = replacement of 138th constant
                   ...          ...
        Replace the 4th, 26th and 138th constant(half words)
          record: word 7       = 0
                  halfword 19 =  4 : location within the complete set
                                    to be replaced
                  halfword 20 = replacement of   4th constant
                  halfword 21 =  26
                  halfword 22 = replacement of  26th constant
                  halfword 23 = 138
                  halfword 24 = replacement of 138th constant
                   ...          ...


C.   Correlation Files of Run Number and Time of Data Taking
-----------------------------------------------------------------

 1.  There exist two files, which contain for each run the time used
     for updating the calibration constants. The two files are
     F11LHO.DSKTI000  : runs  539 ... 9728
     F11LHO.DSKTI001  : runs 1000 ...
     The records are ordered according to the calibration time.

 2.  The records have a fixed length of 9 words:
     word 1 : run number
     word 2 : calibration time
     word 3 : second        )
     word 4 : minute        )
     word 5 : hour          )  time at run start
     word 6 : day           )
     word 7 : month         )
     word 8 : year          )
     word 9 : calibration time of REFORM step

 3.  The second file F11LHO.DSKTI001 is continuously updated by the
     REFORM-job. The new records are added to the end of the file.
     As the runs do not always come in the proper order the file
     must be reordered regularly (about once/month during data taking
     and after the last REFORM-job of a running period) (see C.5).
```

```
     If new constants for the current running period are installed
     it is always recommended to reorder the correlation file before
     installing the new constants.

 4.  With  the  help  of  the  correlation  file  the  subroutine
     CVDTRN(IRUN,ITIME) converts run# into calibration time  and  vice
     versa. If one of the arguments (IRUN or ITIME) is zero the proper
     value will be returned.

 5.  The reordering of  the  correlation  file  F11LHO.DSKTI001  is  a
     rather delicate operation, which must be left to specialists. For
     completeness of this note some gross details  of  this  operation
     are descibed:
     In order to avoid any interference between the REFORM-job,
     which adds new records to the end of the file, and the ordering
     we pursue the following procedure:
      1. Produce a backup copy of F11LHO.DSKTI001 for safety.
      2. Allocate F11LHO.DSKTI001 to your NEWLIB-session:
                              ALLOC F(SAFE) DA('F11LHO.DSKTI001').
      3. Run the job JBLORDER with high priority:
           writes ordered correlation file onto intermediate file.
      4. Check the printout of JBLORDER (no fatal error occurred).
      5. Submit the job JBLCOPY with high priority.
      6. Wait until FREE-request of F11LHO.DSKTI001:
              F11LHO** (....) needs F11LHO.DSKTI001,
              F11LHO** is the name of the submitted JBLCOPY-job.
      7. If any other job (e.g. a REFORM-job) request a FREE, one must
                CANCEL F11LHO**, the JBLCOPY-job,
                FREE DA('F11LHO.DSKTI001') and
                try later again starting from 1.
         In this case a destructive interference has occured.
      8. Otherwise one follows the FREE-request:
                FREE DA('F11LHO.DSKTI001')
      9. Check the printout of JBLcopy (no fatal error occurred).
     10. Don't forget to update #CALNEWS.


D.   Service of the Calibration Files
----------------------------------------

 0.  Also this operation is  rather  delicate  and  must  be  left  to
     specialists. Nevertheless  the  details  are  descibed  here  for
     completeness of this note.

 1.  One must avoid complications arising from jobs, which update  the
     calibration constants, and other  jobs,  which  merely  read  the
     calibration data.
     Therefore we use the file F11LHO.KALWRK0 for  updating  constants
     and as the master file for F11LHO.BUPDAT1 (AUPDAT1).  KALWRK0  is
     copied frequently to BUPDAT1 (AUPDAT1) when constants  are  added
     or changed.

 2.  Updating of the calibration files is performed by  two  different
     jobs:
     - The REFORM-job, which adds LGST-constants
                                   ("spinning block" constants.
     - The so-called KADD-jobs, which add or change
                                           all other constants.

 3.  In order to avoid any interference between the REFORM-job and the
     KADD-job one procedes as follows for the KADD-job:
      1. Produce a backup copy of F11LHO.KALWRK0 for safety.
      2. Allocate F11LHO.KALWRK0 to your NEWLIB-session:
                              ALLOC F(SAFE) DA('F11LHO.KALWRK0').
      3. Run the job JBKADD* with high priority:
           write updated calibration file onto intermediate file.
      4. Check the printout of JBKADD* (no fatal error occurred).
      5. Submit the job JBKALCOP with high priority.
      6. Wait until FREE-request of F11LHO.KALWRK0:
              F11LHO** (....) needs F11LHO.KALWRK0,
              F11LHO** is the name of the submitted JBKALCOP-job.
```

```
 7. If any other job (e.g. a REFORM-job) request a FREE, one
    one must CANCEL F11LHO**, the JBKALCOP-job,
             FREE DA('F11LHO.KALWRK0') and
             try later again starting from 1.
    In this case a destructive interference has occured.
 8. Otherwise one follows the FREE-request:
             FREE DA('F11LHO.KALWRK0')
 9. Check the printout of JBKALCOP (no fatal error occurred).
10. Copy KALWRK0 --> BUPDAT1  (JBKALCOP-job).
11. Produce new AUPDAT1- file from BUPDAT0+KALWRK0 (JBCOMB1-job).
12. Check printouts.
13. Don't forget to update #CALNEWS.
```