Database User's and Reference Manual

Frederique Pelucchi

September 20, 1994

This document describes the functionality and the use of the package RTDB which has been written to manipulate the KLOE database for the test Run 1994. It includes routines which allow user to store/retrieve object in/from a database.

Before describing these routines and how to use them, a short explanation on the databases used is given.

1 A brief description of HEPDB

HEPDB [1] is a database management system developed at CERN by L3, OPAL and the Application Software Group of the Computer and Network Division. It complies the requirements to store a large amount of data such as detector geometry and calibration constants, to have a fast access to the data at program execution and to limit the rate of transactions between the storage media where the database resides and the computer memory available to the user.

HEPDB is based upon the ZEBRA RZ [2] package and increases the functionalities of RZ providing data compression and optimization of performances through partitioned directories. It uses random access files which can reside on direct access devices or in memory. The data is accessed or stored using a UNIX-like pathname, plus some identifiers known as keys.

The fundamental mean for the operation of HEPDB is the database server i.e. an account where a process runs from in detached mode, performs regular tasks and updates the database on clients request.
HEPDB provides an interactive interface to check database status or to modify the database structure as well as Fortran library routines, and it also offers portability between different platforms (via ZEBRA FZ [2] sequential exchange mode files).

A Zebra dynamic store (fortran labeled common block) is used. HEPDB operates inside two divisions in that store: a HEPDB system division which the user is not concerned with and the division where, at user's request, data objects are dumped from the random access media and kept in memory as long as required by the user.

2 KLOE Test Run Databases

For the test run period, three databases are foreseen, one for Run Conditions, one for the Calibration and one for the Geometry. These databases are called respectivly RTRNC.DBS, RTCAL.DBS and RTGEO.DBS and reside in DISK$MF:[KLOEDB.DATABASE] on Frascati ALPHA and VAX, on Roma1 VAX, and on Pisa Alpha.

Each database is divided in directories and is defined for HEPDB by a "index" which is formed by two characters and allows HEPDB routines to locate the database. The directory names characterise the type of the database (Calibration, Run Condition or Geometry), the detector (or subdetector) and the update frequency of the objects which are in the directory. For instance _LT indicates some "Long Term" objects, _MT some "Medium Term" objects and _ST some "Short Term" objects. The meaning of long, medium and short term depends on the type of the database. The detector names are "CALO" for CALOrimeter, "DRCH" for DRift CHamber, "TRIG" for the trigger (which includes Scintillators and Streamer Tubes). For the 4 character objects names, the rule is, the first letter refers to the detector, the last three letters "describe" the object.

2.1 Directory structure

The directories structure is the following, where names between ' ' are the objects (in fact ZEBRA banks) :

For the calibration database (HEPDB index "CT") :

```
//CDCT
      /HELP
      /DICTIONARY
      /CALIBRATION
                  /DRCH_LT
                  /DRCH_MT
                  /DRCH_ST
                  /CALO_LT
                          -'CPMG' (Calorimeter PM Gain)
                  /CALO_MT
                  /CALO_ST
                          -'CMIP' (Calorimeter MIPs), 'CTWA' (Calorimeter
                           Time WAlk corrections), 'CT0S' (T0s),
                           'CVEL' (Calorimeter VELocities), 'CTOF'
                  /TRIG_LT
                  /TRIG_MT
                          -'TGAM' (gamma_1, gamma_2 time walk corrections
                           for the Scintillators),
                           'TTSC' (Time-Space Correlation),'SYTC'
                           (Scintillator ...)
                  /TRIG_ST
```

For the geometry database (HEPDB index "GT") :

```
//CDGT
      /HELP
      /DICTIONARY
      /GEOMETRY
               /DRCH
                    /PARAM_LT
                             -'DPR1', 'DPR2', 'DPR3', 'DPR4' (4 parameter
```

```
                     sets of the cell parametrization), 'DPRA'
                     (Cell radii). Each bank is equivalent of a
                     matrix 6 by 36
                     (6 different cell shapes, 36 phi-section).
               /DESCR_MT
               /DESCR_LT
          /CALO
               /DESCR_MT      Banks to describe Geometry
               /DESCR_LT
          /TRIG
               /DESCR_MT
                     -'SXPO' (Scintillator X position),
                      'SYPO' (Scintillator y position),
                      'TVSY' (Tube versus Y),
                      'TVSZ' (Tube versus Z),
                      'TUPO' (Tube Position),'TUDI' (Tube dimensions)
               /DESCR_LT
```

For the run condition database (HEPDB index "RT") :

```
//CDRT
      /HELP
      /DICTIONARY
      /RUN_COND
               /TRIGGER_ST
               /TRIGGER_LT
               /TRIGGER_MT
                          - 'TTMP' (Trigger streamer Tubes Map File),
                            'TSMP' (Trigger Scintillators Map file),
                            'TDCH' (Trigger Dead CHannels),
                            'THGV' (Trigger High Voltage).
               /DRCH_LT
               /DRCH_MT
                       -'DTHR' (thresholds)
               /DRCH_ST
                       -'DHGV' (High Voltage),
                        'DMAP' (Drift Chamber Map file),
                        'DGAZ' (Gaz parameters), 'DDCH' (Dead Channels),
                        'DDWR' (Dead Wire).
               /CALO_LT
               /CALO_MT
               /CALO_ST
                       -'CMAP' (Calorimeter Map file), 'CAEN' (High Voltage),
                        'CDCH' (Dead Channels), 'CPED' ( ADC Pedestals),
                        'CMOF' (Map Offline), 'CHVM'.
               /BEAM_LT
               /BEAM_MT
               /BEAM_ST
                       -'BRCF' (Beam Run Configuration, P_beam, x,y,z ..)
```

2.2 HEPDB Keys

Each object is defined in the database by a certain number of "keys", that means that you select your object through these keys, the user has up to 100 keys to define the object. In the following table, the keys adopted for the KLOE databases are described. The five first keys are reserved for the HEPDB system and must not be touched by the user or application program.

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

The 14th key can be fill, for instance, with the run number which is used to produce calibration values or with a version number of the object.

3 Database Manipulation

The database itself will never be directly manipulated by a user or an application program unless the update has to be in the database less than one minute after the request. All modifications of a database are done via an intermediate FZ file and a separate program will request the server to make the modification.

For this purpose, a set of routines was written and an independent program exists to allow the user asking update to a database.

Another set of routines are done to read in the database and, to test directly new sets of calibration without going through the database, there are also routines which read in intermediate FZ files.

All these routines are in the RTDB$library:RTDB.OLB, the logical names are defined after typing "setup rtdb".

3.1 Intermediate FZ file

FZ files are ZEBRA Exchange File to be read and written in machine independent format. This format allows user to have a header per data structure, the data structure being a single ZEBRA bank or a whole zebra bank structure. In our case the data structure is a single ZEBRA bank and the header is used to put all the information concerning the database keys in order to be able to put the object correctly in the database lateron.

The description of the header is the following :

1st word: Index of the database ('CT', 'GT' or 'RT')

2nd word: A 4 character flag 'OFF', 'ON' or '   ' (blank) to indicate if the values come from offline or online (for calibration only, in all other case this is simply blank)

3rd word: Bank Name (see below for the existing names)

4th word: Particle Identifier Number if any (for calorimeter, 0 in other case)

5th word: Run Number of the validity start for the object

6th word: Run Number of the validity end for the object

7th word: Run Type (or Trigger Number)

The first three words are 4 character strings, the last 4 words are integer.

To create a new FZ file or to read an existing FZ file, you have first to open it and at the end to close it correctly. You can put as many banks as you want in one FZ file. The routines to do so are now described.

Routines to open a FZ file

```
**********************************************************************
      INTEGER FUNCTION FZOPNW(FILENAM,LUNFZ)
**********************************************************************
C
C     Description :
C     ===========
C     Open a new FZ file
C
C     Input Parameter :
C     ===============
C     CHARACTER*(*) FILENAM ! File Name of the FZ file to be opened
C
C     Output parameters :
C     ==================
C     INTEGER LUNFZ      ! Logical Unit of the FZ file
C
C     Function values :
C     ===============
C                       FZOPNW : Return Code of FZFILE
C                                or error code given by the OPEN statement
```

```
C                                 In any case = 0 if success
*********************************************************************
      INTEGER FUNCTION FZOPOD(FILENAM,LUNFZ)
*********************************************************************
C
C     Description :
C     ===========
C     Open an existing  FZ file
C
C     Input Parameter :
C     ===============
C     CHARACTER*(*) FILENAM ! File Name of the FZ file to be opened
C
C     Output parameters :
C     ==================
C     INTEGER LUNFZ      ! Logical Unit of the FZ file
C
C     Function values :
C     ===============
C                        FZOPOD : Return Code of FZFILE
C                                 or error code given by the OPEN statement
C                                 In any case = 0 if success
```

Routines to close a FZ file

```
*********************************************************************
      INTEGER FUNCTION FZCLOU(LUNFZ)
*********************************************************************
C
C     Description :
C     ===========
C     Close the output FZFILE and UNIT LUNFZ
C
C     Input Parameter :
C     ===============
C     INTEGER LUNFZ      ! Logical Unit of the FZ file
C
C     Output parameters : none
C     ==================
C
C     Function values :
C     ===============
C                        FZCLOU : IQUEST(1) Error Code return by FZENDO
C                                 or error code given by the OPEN statement
C                                 In any case = 0 if success
*********************************************************************
```

```
      INTEGER FUNCTION FZCLIN(LUNFZ)
********************************************************************
C
C     Description :
C     ===========
C     Close the input FZFILE and UNIT LUNFZ
C
C     Input Parameter :
C     ===============
C     INTEGER LUNFZ      ! Logical Unit of the FZ file
C
C     Output parameters : none
C     ==================
C
C     Function values :
C     ===============
C                       FZCLIN : IQUEST(1) Error Code return by FZENDO
C                                or error code given by the OPEN statement
C                                In any case = 0 if success
```

Routines to create FZ file

```
********************************************************************
      INTEGER FUNCTION FZRTVS(LUNFZ,IDXID,SOURCE,NTRIG,BKID,PARTID,
     &                              VALRGR,NBCHL,VALUE,SIGMA)
********************************************************************
C
C     Description :
C     ===========
C     Creates a FZ file for one set of mean and sigmas values.
C
C     Input Parameter :
C     ===============
C     INTEGER LUNFZ      ! Logical Unit of the FZ file
C     CHARACTER*2 IDXID  ! Index of the database ('CT', 'RT', 'GT')
C     CHARACTER*4 SOURCE ! Give the source of the values
C                        ! 'OFF ' for Offline or 'ON  ' for Online
C                        ! or '  ' if you don't care
C     INTEGER    NTRIG   ! Trigger Number or Run Type
C     CHARACTER*4 BKID   ! Bank Identifier
C     INTEGER PARTID     ! Particle Identifier (For Calorimeter Only)
C                        ! Put 0 if you don't care
C     INTEGER VALRGR(2)  ! Array containing the Run Validity Range of the
C                        ! parameters, VALRGR(1) = Run # of Validity Start
C                        !             VALRGR(2) = Run # of Validity End
C                        !             (Put 999999 if it is undetermined)
```

```
C     INTEGER NBCHL       ! Number of Channels
C                         ! = Dimension of the array VALUE and SIGMA
C     REAL VALUE(NBCHL)   ! Array of the parameter values
C     REAL SIGMA(NBCHL)   ! Array of the parameter sigmas
C
C     Output parameters : none
C     ==================
C
C     Function values :
C     ===============
C                       FZRTVS : Error return Code given by FZOUT
C                                = 0 if success
************************************************************************
      INTEGER FUNCTION FZRTV2(LUNFZ,IDXID,SOURCE,NTRIG,BKID,PARTID,
     &                        VALRGR,NBCHL,VALUE1,SIGMA1,VALUE2,SIGMA2)
************************************************************************
C
C     Description :
C     ===========
C     Creates a FZ file for two sets of mean and sigmas values.
C
C     Input Parameter :
C     ===============
C     INTEGER LUNFZ         ! Logical Unit of the FZ file
C     CHARACTER*2 IDXID     ! Index of the database
C     CHARACTER*4 SOURCE    ! Give the source of the calibration values
C                           ! 'OFF ' for Offline or 'ON  ' for Online
C     INTEGER    NTRIG      ! Trigger Number
C     CHARACTER*4 BKID      ! Bank Identifier
C     INTEGER PARTID        ! Particle Identifier (For Calorimeter Only)
C                           ! Put 0 if you don't care
C     INTEGER VALRGR(2)     ! Array containing the Run Validity Range of the
C                           ! parameters, VALRGR(1) = Run # of Validity Start
C                           !             VALRGR(2) = Run # of Validity End
C                           !             (Put 999999 if you don't know)
C     INTEGER NBCHL         ! Number of Channels
C                           ! = Dimension of the array VALUE and SIGMA
C     REAL VALUE1(NBCHL)    ! Array of the First set of values
C     REAL SIGMA1(NBCHL)    ! Array of the First set of sigmas
C     REAL VALUE2(NBCHL)    ! Array of the Second set of values
C     REAL SIGMA2(NBCHL)    ! Array of the Second set of sigmas
C
C     Output parameters : none
C     ==================
C
C     Function values :
C     ===============
C                       FZRTC2 : Error return Code given by FZOUT
```

```
C                                = 0 if success
*******************************************************************
      INTEGER FUNCTION FZRTVA(LUNFZ,IDXID,SOURCE,BKID,PARTID,
     &                        VALRGR,NV,VALUES)
*******************************************************************
C
C     Description :
C     ===========
C     Creates a FZ file for one set of REAL values.
C
C     Input Parameter :
C     ===============
C     INTEGER LUNFZ      ! Logical Unit of the FZ file
C     CHARACTER*2 IDXID  ! Database Index ('RT','CT','GT')
C     CHARACTER*4 SOURCE ! flag off or on-line ('OFF ','ON ')
C     CHARACTER*4 BKID   ! Bank Identifier
C     INTEGER PARTID     ! Particle Identifier (Put 0 if you don't care)
C     INTEGER VALRGR(2)  ! Array containing the Run Validity Range of the
C                        ! parameters, VALRGR(1) = Run # of Validity Start
C                        !             VALRGR(2) = Run # of Validity End
C                        !             (Put 999999 if you don't know)
C     INTEGER NTRIG      ! Trigger Number or Run Type
C     INTERGER NV        ! Number of values
C     REAL VALUES(NV)    ! Parameter Values.
C
C     Output parameters : none
C     ==================
C
C     Function values :
C     ===============
C                       FZRTVA : Error return Code given by FZOUT
C                                = 0 if success
C
*******************************************************************
      INTEGER FUNCTION FZRTVI(LUNFZ,IDXID,SOURCE,BKID,PARTID,
     &                        VALRGR,NV,VALUES)
*******************************************************************
C
C     Description :
C     ===========
C     Creates a FZ file for one set of INTEGER values.
C
C     Input Parameter :
C     ===============
C     INTEGER LUNFZ      ! Logical Unit of the FZ file
C     CHARACTER*2 IDXID  ! Database Index ('RT','CT','GT')
C     CHARACTER*4 SOURCE ! flag off or on-line ('OFF ','ON ')
C     CHARACTER*4 BKID   ! Bank Identifier
```

```
C     INTEGER PARTID     ! Particle Identifier (Put 0 if you don't care)
C     INTEGER VALRGR(2)  ! Array containing the Run Validity Range of the
C                        ! parameters, VALRGR(1) = Run # of Validity Start
C                        !             VALRGR(2) = Run # of Validity End
C                        !             (Put 999999 if you don't know)
C     INTEGER NTRIG      ! Trigger Number or Run Type
C     INTERGER NV        ! Number of values
C     INTEGER VALUES(NV) ! Parameter Values.
C
C     Output parameters : none
C     ==================
C
C     Function values :
C     ===============
C                  FZRTVI: Error return Code given by FZOUT
C                           = 0 if success
C
*******************************************************************
      INTEGER FUNCTION FZCMAP(LUNFZ,NTRIG,VALRGR,NC,ADDPTR,DETPTR)
*******************************************************************
C
C     Description :
C     ===========
C     Creates a FZ file with the Calorimeter Map File for
C     RUN CONDITION database
C
C     Input Parameter :
C     ===============
C     INTEGER LUNFZ         ! Logical Unit of the FZ file
C     INTEGER NTRIG         ! Trigger Number or Run Type
C     INTEGER VALRGR(2)     ! Array containing the Run Validity Range of the
C                           ! parameters, VALRGR(1) = Run # of Validity Start
C                           !             VALRGR(2) = Run # of Validity End
C                           !               (Put 999999 if you don't know)
C     INTERGER NC           ! Number of Channels
C     INTEGER ADDPTR(NC,5)  ! "Hard Ware" address: 1- Cart type
C                           !                      2- VME address
C                           !                      3- Crate Number
C                           !                      4- Slot Number
C                           !                      5- Subaddress
C     INTEGER DETPTR(NC,5)  ! Detector Address: 1- Detector Number
C                           !                   2- Side (1:2)
C                           !                   3- Module Number (1:NC)
C                           !                   4- Row (1:5)
C                           !                   5- Column (1:12)
C
C     Output parameters : none
C     ==================
```

```
C
C     Function values :
C     ===============
C                       FZCMAP : Error return Code given by FZOUT
C                                  = 0 if success
******************************************************************
      INTEGER FUNCTION FZTMAP(LUNFZ,NTRIG,BKID,VALRGR,NC,ADDPTR,DETPTR)
******************************************************************
C
C     Description :
C     ===========
C     Creates a FZ file with Trigger Map File for
C     scintillators or Streamer Tubes.
C
C     Input Parameter :
C     ===============
C     INTEGER LUNFZ          ! Logical Unit of the FZ file
C     INTEGER NTRIG          ! Trigger Number or Run Type
C     CHARACTER*4 BKID       ! Bank Name (TSMP for Trigger Scintillator,
C                            !            TTMP for Trigger streamer Tubes)
C     INTEGER VALRGR(2)      ! Array containing the Run Validity Range of the
C                            ! parameters, VALRGR(1) = Run # of Validity Start
C                            !             VALRGR(2) = Run # of Validity End
C                            !               (Put 999999 if you don't know)
C     INTERGER NC            ! Total Number of Channels
C     INTEGER ADDPTR(NC,2)   ! "Hard Ware" address:
C                            !  For Scintillators : 1- Slot Number
C                            !                      2- Channel number
C                            !  For Streamer Tubes: 1- Stroc ch.
C                            !                      2- Address
C     INTEGER DETPTR(NC,4)   ! Detector Address:
C                            ! For Scintillators : 1- Card type (1=ADC, 2=TDC)
C                            !                     2- Up-Down position
C                            !                        (1=up, 2=down)
C                            !                     3- position
C                            !                     4- side (left=1, right=2)
C                            ! For Streamer Tubes: 1- Up-Down position
C                            !                        (1=up, 2=down)
C                            !                     2- plane
C                            !                     3- coordonate
C                            !                     4- strip number
C     Output parameters : none
C     ==================
C
C     Function values :
C     ===============
C                       FZTMAP : Error return Code given by FZOUT
C                                  = 0 if success
```

```
******************************************************************
      INTEGER FUNCTION FZRTMA(LUNFZ,IDXID,SOURCE,NTRIG,BKID,
     &                        VALRGR,NR,NC,VALUE)
******************************************************************
C
C     Description :
C     ===========
C     Creates a FZ file with a matrix of NR*NC values
C
C     Input Parameter :
C     ===============
C     INTEGER LUNFZ      ! Logical Unit of the FZ file
C     CHARACTER*2 IDXID  ! Index of the database ('CT', 'RT', 'GT')
C     CHARACTER*4 SOURCE ! Give the source of the values
C                        ! 'OFF ' for Offline or 'ON  ' for Online
C                        ! or '  ' if you don't care
C     INTEGER    NTRIG   ! Trigger Number or Run Type
C     CHARACTER*4 BKID   ! Bank Identifier
C     INTEGER PARTID     ! Particle Identifier (For Calorimeter Only)
C                        ! Put 0 if you don't care
C     INTEGER VALRGR(2)  ! Array containing the Run Validity Range of the
C                        ! parameters, VALRGR(1) = Run # of Validity Start
C                        !             VALRGR(2) = Run # of Validity End
C                        !             (Put 999999 if it is undetermined)
C     INTEGER NR         ! Number of rows
C                        ! = 1 st Dimension of the array VALUE
C     INTEGER NC         ! Number of Columns
C                        ! = 2 nd Dimension of the array VALUE
C     REAL VALUE(NR,NC)  ! Array of the parameter values
C
C     Output parameters : none
C     ==================
C
C     Function values :
C     ===============
C                       FZRTMA : Error return Code given by FZOUT
C                                = 0 if success
C
C  Warning : the array passed to the routine must have EXACTLY the
C  ========= same dimension of the matrix
C
******************************************************************
      INTEGER FUNCTION FZR4MA(LUNFZ,IDXID,SOURCE,NTRIG,BKID,PARTID,
     &                        VALRGR,NDIM,NR,NC,VALUE)
******************************************************************
C
C     Description :
C     ===========
```

```
C     Creates a FZ file with a REAL matrix of NR*NC values. The array
C     passed to fill the matrix is assumed to be an array dimensionned
C     as VALUE(NDIM,N), NDIM must be passed as argument.
C
C     Input Parameter :
C     ===============
C     INTEGER LUNFZ      ! Logical Unit of the FZ file
C     CHARACTER*2 IDXID  ! Index of the database ('CT', 'RT', 'GT')
C     CHARACTER*4 SOURCE ! Give the source of the values
C                        ! 'OFF ' for Offline or 'ON  ' for Online
C                        ! or '  ' if you don't care
C     INTEGER     NTRIG  ! Trigger Number or Run Type
C     CHARACTER*4 BKID   ! Bank Identifier
C     INTEGER PARTID     ! Particle Identifier (For Calorimeter Only)
C                        ! Put 0 if you don't care
C     INTEGER VALRGR(2)  ! Array containing the Run Validity Range of the
C                        ! parameters, VALRGR(1) = Run # of Validity Start
C                        !             VALRGR(2) = Run # of Validity End
C                        !             (Put 999999 if it is undeterminated)
C     INTEGER NDIM       ! The first dimension of the array passed
C                        ! to fill the matrix. It could be different
C                        ! from the first dimension of the matrix, if
C                        ! you declare a bigger array than you really use for
C                        ! the matrix.
C     INTEGER NR         ! Number of rows
C                        ! = 1 st Dimension of the matrix
C     INTEGER NC         ! Number of Columns
C                        ! = 2 nd Dimension of the matrix
C     REAL VALUE(*)      ! Array of the parameter values
C
C     Output parameters : none
C     ==================
C
C     Function values :
C     ===============
C                       FZR4MA : Error return Code given by FZOUT
C                                = 0 if success
*********************************************************************
```

__Routines to read in the FZ file__

```
*********************************************************************
      INTEGER FUNCTION GTFZNW(LUNFZ,IDXID,SOURCE,VERNUM,RUNNUM,NTRIG,
     &                        BNKNA,NW)
*********************************************************************
C
C
```

```
C     Description :
C     ===========
C     Retrieve the number of data words of a bank in a FZ file
C
C     Input Parameter :
C     ===============
C     INTEGER LUNFZ      ! Logical Unit of the FZ file
C     CHARACTER*2 IDXID  ! Database Index
C     INTEGER SOURCE     ! Source of the object 0 = ONline, 1 = OFFline
C     INTEGER  VERNUM    ! Version Number
C     INTEGER    RUNNUM  ! Run Number
C     INTEGER    NTRIG   ! Trigger number or Run Type
C     CHARACTER*4 BNKNA  ! Bank name
C
C     Output parameters :
C     ==================
C     INTEGER NW        ! Number of Data Words
C
C     Function values :
C     ===============
C                      GTFZNW : Return Code given by FZIN, 0 if success
C                               (ZEBRA Code)
*********************************************************************
      INTEGER FUNCTION GTFZHD(LUNFZ,IDXID,BNKNA,
     &                        SOURCE,RUNSRT,RUNEND,NTRIG,NVER)
*********************************************************************
C
C     Description :
C     ===========
C     Retrieve the FZ file header for the bank BNKNA, and index IDXID
C
C     Input Parameter :
C     ===============
C     INTEGER LUNFZ     ! Logical Unit of the FZ file
C     CHARCTER*2 IDXID  ! Database Index
C     CHARACTER*4 BNKNA ! Bank name
C
C     Output parameters :
C     ==================
C     CHARACTER*4 SOURCE ! OFFline-Online flag
C     INTEGER     RUNSRT ! Run Number for the start of validity
C     INTEGER     RUNEND ! Run Number for the end of validity
C     INTEGER     NTRIG  ! Trigger number or Run type
C     INTEGER     NVER   ! Version number or PID number
C
C     Function values :
C     ===============
C                      GTFZHD : Return Code given by FZIN, 0 if success
```

```
C                                (ZEBRA Code)
*********************************************************************
      INTEGER FUNCTION GTFZMV(LUNFZ,IDXID,VERNUM,RUNNUM,NTRIG,CHANUM,
     &                        BNKNA,VALUE,SIGMA)
*********************************************************************
C
C     Description :
C     ===========
C     Retrieve mean and sigma value for channel number CHANUM
C     from FZ File where there is, at least,
C     one bank containing one set of mean and sigma values
C
C     Input Parameter :
C     ===============
C     INTEGER LUNFZ     ! Logical Unit of the FZ file
C     CHARCTER*2 IDXID  ! Database Index
C     INTEGER  VERNUM   ! Version Number
C     INTEGER    RUNNUM ! Run Number
C     INTEGER    NTRIG  ! Trigger number
C     INTEGER   CHANUM  ! Channel Number
C     CHARACTER*4 BNKNA ! Bank name
C
C     Output parameters :
C     ==================
C     REAL VALUE  ! Parameter value
C     REAL SIGMA  ! Error on this value
C
C     Function values :
C     ===============
C                      GTFZMV : Return Code given by FZIN, 0 if success
C                               (ZEBRA Code)
C
*********************************************************************
      INTEGER FUNCTION GTFZVS(LUNFZ,IDXID,SOURCE,VERNUM,RUNNUM,NTRIG,
     &                        BNKNA,NBMAX,VALUE,SIGMA,NBCHA)
*********************************************************************
C
C
C     Description :
C     ===========
C     Retrieve All mean and sigma values from FZ File where there is,
C     at least, one bank containing one set of mean and sigma values
C
C     Input Parameter :
C     ===============
C     INTEGER LUNFZ      ! Logical Unit of the FZ file
C     CHARACTER*2 IDXID  ! Database Index
C     INTEGER SOURCE     ! Source of the object 0 = ONline, 1 = OFFline
```

```
C     INTEGER  VERNUM    ! Version Number
C     INTEGER    RUNNUM  ! Run Number
C     INTEGER    NTRIG   ! Trigger number or Run Type
C     CHARACTER*4 BNKNA  ! Bank name
C     INTEGER NBMAX      ! Maximum Number of Channels (Dimension of the arrays
C                        ! VALUE and SIGMA)
C
C     Output parameters :
C     =================
C     REAL VALUE(NBMAX) ! Parameter values
C     REAL SIGMA(NBMAX) ! Errors on the values
C     INTEGER NBCHA     ! Number of Channels actually found
C
C     Function values :
C     ===============
C                      GTFZVS : Return Code given by FZIN, 0 if success
C                               (ZEBRA Code)
*********************************************************************
      INTEGER FUNCTION GTFZV2(LUNFZ,IDXID,SOURCE,VERNUM,RUNNUM,NTRIG,
     &              BNKNA,NBMAX,VALUE1,SIGMA1,VALUE2,SIGMA2,NBCHA)
*********************************************************************
C
C
C     Description :
C     ===========
C     Retrieve two sets of mean and sigma values from an FZ File
C     where there is, at least,
C     one bank containing two sets of mean and sigma values
C
C     Input Parameter :
C     ===============
C     INTEGER LUNFZ      ! Logical Unit of the FZ file
C     CHARACTER*2 IDXID  ! Database Index
C     INTEGER SOURCE     ! Source of the object 0 = ONline, 1 = OFFline
C     INTEGER  VERNUM    ! Version Number
C     INTEGER    RUNNUM  ! Run Number
C     INTEGER    NTRIG   ! Trigger number or Run Type
C     CHARACTER*4 BNKNA  ! Bank name
C     INTEGER NBMAX      ! Maximum Number of Channels (Dimension of the arrays
C                        ! VALUE and SIGMA)
C
C     Output parameters :
C     =================
C     REAL VALUE1(NBMAX) ! 1st set of mean values
C     REAL SIGMA1(NBMAX) ! 1st corresponding set of Error on this value
C     REAL VALUE2(NBMAX) ! 2nd set of mean values
C     REAL SIGMA2(NBMAX) ! 2nd corresponding set of Error on this value
C     INTEGER NBCHA      ! Number of Channels actually found
```

```
C
C     Function values :
C     ===============
C                      GTFZV2 : Return Code given by FZIN, 0 if success
C                               (ZEBRA Code)
*********************************************************************
      INTEGER FUNCTION GTFZVA(LUNFZ,IDXID,SOURCE,VERNUM,RUNNUM,NTRIG,
     &                        BNKNA,NBMAX,VALUE,NBVA)
*********************************************************************
C
C     Description :
C     ===========
C     Retrieve All values from FZ File where there is,
C     at least, one bank containing one list of values.
C
C     Input Parameter :
C     ===============
C     INTEGER LUNFZ      ! Logical Unit of the FZ file
C     CHARACTER*2 IDXID  ! Database Index
C     INTEGER SOURCE     ! Source of the object 0 = ONline, 1 = OFFline
C     INTEGER  VERNUM    ! Version Number
C     INTEGER    RUNNUM  ! Run Number
C     INTEGER    NTRIG   ! Trigger number
C     CHARACTER*4 BNKNA  ! Bank name
C     INTEGER NBMAX      ! Maximum Number of Channels
C
C     Output parameters :
C     ==================
C     REAL VALUE(NBMAX) ! Parameter value
C     INTEGER NBVA      ! Number of Channels actually found
C
C     Function values :
C     ===============
C                      GTFZVA : Return Code given by FZIN, 0 if success
C                               (ZEBRA Code)
*********************************************************************
      INTEGER FUNCTION GTFZMA(LUNFZ,IDXID,SOURCE,VERNUM,RUNNUM,NTRIG,
     &                        BNKNA,NRI,NCI,VALUE,NR,NC)
*********************************************************************
C
C
C     Description :
C     ===========
C     Retrieve All values of a matrix from FZ File where there is,
C     at least, one bank containing a matrix.
C
C     Input Parameter :
C     ===============
```

```
C     INTEGER LUNFZ      ! Logical Unit of the FZ file
C     CHARACTER*2 IDXID  ! Database Index
C     INTEGER SOURCE     ! Source of the object 0 = ONline, 1 = OFFline
C     INTEGER  VERNUM    ! Version Number
C     INTEGER    RUNNUM  ! Run Number
C     INTEGER    NTRIG   ! Trigger number
C     CHARACTER*4 BNKNA  ! Bank name
C     INTEGER NRI        ! 1st dimension of the array value
C     INTEGER NCI        ! 2nd dimension of the array value
C
C     Output parameters :
C     ==================
C     REAL VALUE(NRI,NCI) ! value
C     INTEGER NR          ! Number of Row actually found
C     INTEGER NC          ! Number of Column actually found
C
C     Function values :
C     ===============
C                     GTFZMA : Return Code given by FZIN, 0 if success
C                              (ZEBRA Code)
C
*********************************************************************
      INTEGER FUNCTION GTFZML(LUNFZ,IDXID,SOURCE,VERNUM,RUNNUM,NTRIG,
     &                        BNKNA,NR,NC,VALUE)
*********************************************************************
C
C
C     Description :
C     ===========
C     Retrieve the value of the matrix element (NR,NC)
C      from FZ File where there is,
C     at least, one bank containing a matrix.
C
C     Input Parameter :
C     ===============
C     INTEGER LUNFZ     ! Logical Unit of the FZ file
C     CHARACTER*2 IDXID ! Database Index
C     INTEGER SOURCE    ! Source of the object 0 = ONline, 1 = OFFline
C     INTEGER  VERNUM   ! Version Number
C     INTEGER    RUNNUM ! Run Number
C     INTEGER    NTRIG  ! Trigger number
C     CHARACTER*4 BNKNA ! Bank name
C     INTEGER NR        ! Number of Row
C     INTEGER NC        ! Number of Column
C
C     Output parameters :
C     ==================
C     REAL VALUE ! value
```

```
C
C     Function values :
C     ===============
C                      GTFZML : Return Code given by FZIN, 0 if success
C                                (ZEBRA Code)
*********************************************************************
```

3.2 Database program and routines

3.2.1 Program to modify Database

To modify a database, the program RTDB$PROGRAMS:DBMENU.EXE has to be runned. This program displays a menu and you normally just have to answer the questions. All users cannot ask for a modification in a database, only user with a special "Rights Identifier" can. These persons must be designed by their own group.

All the routines used by the program are included in the RTDB library. The code source can be found in RTDB$REFERENCE_SOURCE:RTUPSH.KLOE.

Here are the list of the routines available at that time :

Routine to Up Date a database:

```
*********************************************************************
      INTEGER FUNCTION UDRTDB()
*********************************************************************
C
C     Description :
C     ===========
C     Updates the test beam database from a FZ file
*********************************************************************
```

Routine to Delete an object in database

```
*********************************************************************
      INTEGER FUNCTION DERTDB()
*********************************************************************
C
C     Description :
C     ===========
C     Deletes an object in the test beam database.
C
*********************************************************************
```

Routine to Replace an object

```
*********************************************************************
      INTEGER FUNCTION RERTDB()
*********************************************************************
C
```

```
C     Description :
C     ===========
C     Replace an object in the test beam database from a FZ
C     file
*********************************************************************
```

Routine to restore an object

```
*********************************************************************
      INTEGER FUNCTION RSRTDB()
*********************************************************************
C
C     Description :
C     ===========
C     Restore a deleted object in database.
C
*********************************************************************
```

Routine to Up Date the Calorimeter MAP file database:

```
***************************************************************************
      INTEGER FUNCTION UDRTMP(BNKNA,RAWMAP,NCHA,LBANK)
*********************************************************************
C
C     Description :
C     ===========
C     Updates the Detector MAP (I*2) FILE in the run test beam database
C
C     Input Parameter :
C     ===============
C     CHARACTER*4 BNKNA ! Bank name
C     INTEGER*2 RAWMAP  ! Map file array from the comment record.
C     INTEGER NCHA      ! Number of channels
C
C     Output parameters :
C     ==================
C     INTEGER LBANK ! Returned addresss of the Data bank
C
C     Function values :
C     ===============
C                       UDRTMP : IRC Error return by CDSTOR
C                                0 if successful
*********************************************************************
      INTEGER FUNCTION UDI4MP(IDXDB,BNKNA,RAWMAP,DIMMAP)
*********************************************************************
C
C     Description :
```

```
C     ===========
C     Updates Detector MAP FILE (integer*4)
C
C     Input Parameter :
C     ===============
C     CHARACTER*2 IDXDB   ! Index of the database
C     CHARACTER*4 BNKNA   ! Bank name
C     INTEGER*4 RAWMAP(*) ! Map file array from the comment record.
C     INTEGER*4 DIMMAP    ! Dimension of the map file
C
C     Output parameters :  none
C     ==================
C
C     Function values :
C     ===============
C                       UDI4MP : IRC Error returned by CDSTOR or CDVECT
C                                0 if successful
*********************************************************************
```

Useful Routines

- ```
  *********************************************************************
        INTEGER FUNCTION FZPATH(HEADCHAR,PATHNAM)
  *********************************************************************
  C
  C     Description :
  C     ===========
  C     Find the pathname with the FZ header information
  C
  C     Input Parameter :
  C     ===============
  C     CHARACTER*4 HEADCHAR(FZHDSZ) ! Header of the FZ file
  C
  C     Output parameters :
  C     ==================
  C     CHARACTER*100 PATHNAM ! Absolut Path name
  C
  C     Function values :
  C     ===============
  C                       FZPATH : 0 if the path name is found
  C                                1 if the database index is unknown
  C                                2 if the bank is unknown
  *********************************************************************
  ```

- ```
  *********************************************************************
        INTEGER FUNCTION PATHCH(PATH)
  ```

```
*******************************************************************
C
C     Description :
C     ===========
C     Check the path name of a directory
C
C     Input Parameter :
C     ===============
C     CHARACTER*100 PATH ! Path Name
C
C     Output parameters : none
C     =================
C
C     Function values :
C     ===============
C                       PATHCH : 0 if success
C                                1 if no database prefix
C                                2 if database prefix unknown
C
*******************************************************************
```

- ```
  *******************************************************************
        INTEGER FUNCTION RTPATH(IDXID,BNKNA,PATHNAM)
  *******************************************************************
  C
  C     Description :
  C     ===========
  C     Find the absolut pathname of an object from the bank Name
  C     and the database index.
  C
  C     Input Parameter :
  C     ===============
  C     CHARACTER*4 BNKNA ! Bank Name
  C
  C     Output parameters :
  C     =================
  C     CHARACTER*2 IDXID     ! Database Index
  C     CHARACTER*100 PATHNAM ! Absolut Pathname
  C
  C     Function values :
  C     ===============
  C                       RTPATH :  0 if success, 1 if the detector id unknown,
  C                                 2 if the bank is unknown
  *******************************************************************
  ```

- ```
  *******************************************************************
        INTEGER FUNCTION HDBINF(PATHNA,SOURCE,NTRIG,
       &                        VERNUM,RUNNUM,BNKNA,LBANK)
  ```

```
******************************************************************
C
C     Description :
C     ===========
C     Retrieve data bank from database
C
C     Input Parameter :
C     ===============
C     CHARACTER*100 PATHNA ! Absolut Path Name of the directory
C     INTEGER SOURCE       ! Source of the object 0 = ONline, 1 = OFFline
C     INTEGER NTRIG        ! Trigger Number
C     INTEGER  VERNUM      ! Version Number
C     INTEGER RUNNUM       ! Run Number
C     CHARACTER*4 BNKNA    ! Bank Identifier
C
C     Output parameters :
C     ==================
C     INTEGER LBANK        ! Pointer for the HEPDB keys Bank
C                          ! The data bank is stored as daughter bank
C                          ! on the first link of the key bank i.e.
C                          ! Data pointer = LQ(LBANK-1) = DPOINT
C                          ! Then data words are stored beginning with
C                          ! IQ(DPOINT+1) or Q(DPOINT+1) depending if
C                          ! it is an integer (IQ) or a real (Q), and
C                          ! ending with IQ(DPOINT+NW) or Q(DPOINT+NW),
C                          ! NW being the number of data words given by
C                          ! IQ(DPOINT-1). The LBANK pointer has to be protected
C                          ! against relocations or gabage collection, e.g
C                          ! must be in a link area. Such aerea is already
C                          ! existing, so if you use the pointer LKFZFN to
C                          ! keep this address (LBANK=LKFZFN),
C                          ! you should not have problem.
C     Function values :
C     ===============
C                      HDBINF : Return Code given by CDUSEM, 0 if success
C                               (HEPDB Code)
******************************************************************
```

To initialise ZEBRA and create the links for Database

```
******************************************************************
      INTEGER FUNCTION ZDBINI(HDBSIZ,LIST)
******************************************************************
C
C     Description :
C     ===========
C     Initialise ZEBRA and the link structure used by Database
```

```
C     routines
C
C     Input Parameter :
C     ===============
C     INTEGER HDBSIZ  ! Size of the memory used for the database
C     INTEGER LIST(*) ! Parameter list to set the log level for ZEBRA
C                     ! as given in the MZEBRA routine description
C
C     Output parameters : none
C     ==================
*********************************************************************
```

3.2.2 Routines for Users

For the moment, the routines to retrieve information from the database are:

```
*********************************************************************
      INTEGER FUNCTION GTRTMP(BNKNA,VERSION,NCHMAX,MAPFIL,NCHAN)
*********************************************************************
C
C     Description :
C     ===========
C     Retrieve the Detector MAP FILE from the run test beam database
C
C     Input Parameter :
C     ===============
C     CHARACTER*4 BNKNA ! Bank name
C     INTEGER  VERSION  ! Version number of the map file to be retrieved
C     INTEGER NCHMAX    ! Maximum number of channel to retrieve
C
C     Output parameters :
C     ==================
C     INTEGER*2 MAPFIL ! Map file array from the Run Condition Database.
C     INTEGER NCHAN    ! Number of channel really found (minimum between
C                      !  NCHMAX and NCHAN)
C
C     Function values :
C     ===============
C                       GTRTMP : IRC Error return by CDSTOR
C                                0 if successful
*********************************************************************
      INTEGER FUNCTION GTDBMV(IDXID,SOURCE,VERNUM,RUNNUM,NTRIG,CHANUM,
     &                        BNKNA,VALUE,SIGMA)
*********************************************************************
C
C     Description :
C     ===========
C     Retrieve mean and sigma value for channel CHANUM,
```

```
C     from database bank which contain one set of mean and sigma values.
C
C     Input Parameter :
C     ===============
C     CHARACTER*2 IDXID ! Database index
C     INTEGER  SOURCE   ! Calibration source, 0 = ONline, 1 = OFFline
C     INTEGER VERNUM    ! Version Number
C     INTEGER    RUNNUM ! Run Number
C     INTEGER    NTRIG  ! Trigger number
C     INTEGER   CHANUM  ! Channel Number
C     CHARACTER*4 BNKNA ! Bank name
C
C     Output parameters :
C     ==================
C     REAL VALUE  ! Parameter value
C     REAL SIGMA  ! Error on this value
C
C     Function values :
C     ===============
C                      GTDBMV : Return Code given by CDUSEM, 0 if success
C                               (HEPDB Code)
*******************************************************************
      INTEGER FUNCTION GTDBVS(IDXID,SOURCE,VERNUM,RUNNUM,NTRIG,BNKNA,
     &                        NBMAX,VALUE,SIGMA,NBCHA)
*******************************************************************
C
C     Description :
C     ===========
C     Retrieve All mean and sigma values from database bank which
C     contains one set of mean and sigma values.
C
C     Input Parameter :
C     ===============
C     CHARACTER*2 IDXID ! Database Index
C     INTEGER  SOURCE   ! Calibration source, 0 = ONline, 1 = OFFline
C     INTEGER VERNUM    ! Version Number
C     INTEGER    RUNNUM ! Run Number
C     INTEGER    NTRIG  ! Trigger number
C     CHARACTER*4 BNKNA ! Bank name
C     INTEGER NBMAX     ! Maximum Number of Channels
C
C     Output parameters :
C     ==================
C     REAL VALUE(NBMAX) ! Parameter value
C     REAL SIGMA(NBMAX) ! Error on this value
C     INTEGER NBCHA     ! Number of Channels actually found
C
C     Function values :
```

```
C     ===============
C                      GTDBVS : Return Code given by CDUSEM, 0 if success
C                               (HEPDB Code)
**********************************************************************
      INTEGER FUNCTION GTDBV2(IDXID,SOURCE,VERNUM,RUNNUM,NTRIG,BNKNA,
     &                        NBCHA,VALUE1,SIGMA1,VALUE2,SIGMA2)
**********************************************************************
C
C     Description :
C     ===========
C     Retrieve two sets of mean and sigma values from database bank which
C     contains two sets of mean and sigma values.
C
C     Input Parameter :
C     ===============
C     CHARACTER*2 IDXID ! Database Index
C     INTEGER  SOURCE   ! Calibration source, 0 = ONline, 1 = OFFline
C     INTEGER VERNUM    ! Version Number
C     INTEGER    RUNNUM ! Run Number
C     INTEGER    NTRIG  ! Trigger number or Run Type
C     CHARACTER*4 BNKNA ! Bank name
C     INTEGER NBMAX     ! Maximum Number of Channels
C
C     Output parameters :
C     ==================
C     REAL VALUE1(NBCHL)   ! Array of the First set of values
C     REAL SIGMA1(NBCHL)   ! Array of the First set of sigmas
C     REAL VALUE2(NBCHL)   ! Array of the Second set of values
C     REAL SIGMA2(NBCHL)   ! Array of the Second set of sigmas
C     INTEGER NBCHA        ! Number of Channels actually found
C
C     Function values :
C     ===============
C                      GTDBV2 : Return Code given by CDUSEM, 0 if success
C                               (HEPDB Code)
**********************************************************************
      INTEGER FUNCTION GTDBSR(IDXID,SOURCE,VERNUM,RUNNUM,NTRIG,BNKNA,
     &                        NUMSER)
**********************************************************************
C
C     Description :
C     ===========
C     Retrieve the serial number of a given object
C
C     Input Parameter :
C     ===============
C     CHARACTER*2 IDXID ! Database Index
C     INTEGER  SOURCE   ! Calibration source, 0 = ONline, 1 = OFFline
```

```
C     INTEGER VERNUM    ! Version Number
C     INTEGER    RUNNUM ! Run Number
C     INTEGER    NTRIG  ! Trigger number
C     CHARACTER*4 BNKNA ! Bank name
C
C     Output parameters :
C     ==================
C     INTEGER NUMSER ! Serial Number
C
C     Function values :
C     ===============
C                      GTDBSR : Return Code given by CDUSEM, 0 if success
C                               (HEPDB Code)
*************************************************************************
      INTEGER FUNCTION GTDBNW(IDXID,BNKNA,SOURCE,VERNUM,RUNNUM,NTRIG,NW)
*************************************************************************
C
C     Description :
C     ===========
C     Retrieve the number of Data words in the bank BNKNA
C
C     Input Parameter :
C     ===============
C     CHARACTER*2 IDXID ! Database Index
C     CHARACTER*4 BNKNA ! Bank Name
C     INTEGER VERNUM    ! Version Number
C     INTEGER    RUNNUM ! Run Number
C     INTEGER    NTRIG  ! Trigger number or Run Type
C
C     Output parameters :
C     ==================
C     INTEGER NW        ! Number of Data word
C
C     Function values :
C     ===============
C                       GTDBNW : Return Code given by CDUSEM, 0 if success
C                               (HEPDB Code)
********************************************************************
      INTEGER FUNCTION GTDBVA(IDXID,SOURCE,VERNUM,RUNNUM,NTRIG,BNKNA,
     &                         NBMAX,VALUE,NBVA)
********************************************************************
C
C     Description :
C     ===========
C     Retrieve all values from database bank containing
C     just one list of values.
C
C     Input Parameter :
```

```
C     ===============
C     CHARACTER*2 IDXID ! Database Index
C     INTEGER  SOURCE   ! Calibration source, 0 = ONline, 1 = OFFline
C     INTEGER VERNUM    ! Version Number
C     INTEGER    RUNNUM ! Run Number
C     INTEGER    NTRIG  ! Trigger number
C     CHARACTER*4 BNKNA ! Bank name
C     INTEGER NBMAX     ! Maximum Number of Values
C
C     Output parameters :
C     ==================
C     REAL VALUE(NBMAX) ! values
C     INTEGER NBVA      ! Number of Values actually found
C
C     Function values :
C     ===============
C                      GTCLAL : Return Code given by CDUSEM, 0 if success
C                               (HEPDB Code)
C
*********************************************************************
      INTEGER FUNCTION GTDBVL(IDXID,SOURCE,VERNUM,RUNNUM,NTRIG,BNKNA,
     &                        NV,VALUE)
*********************************************************************
C
C     Description :
C     ===========
C     Retrieve the value for "channel" NV from database bank containing
C     just one list of values.
C
C     Input Parameter :
C     ===============
C     CHARACTER*2 IDXID ! Database Index
C     INTEGER  SOURCE   ! Calibration source, 0 = ONline, 1 = OFFline
C     INTEGER VERNUM    ! Version Number
C     INTEGER    RUNNUM ! Run Number
C     INTEGER    NTRIG  ! Trigger number
C     CHARACTER*4 BNKNA ! Bank name
C     INTEGER NV     ! Number of Values
C
C     Output parameters :
C     ==================
C     REAL VALUE ! value
C
C     Function values :
C     ===============
C                      GTDBVL: Return Code given by CDUSEM, 0 if success
C                               (HEPDB Code)
*********************************************************************
```

```
      INTEGER FUNCTION GTDBML(IDXID,SOURCE,VERNUM,RUNNUM,NTRIG,BNKNA,
     &                        NR,NC,VALUE)
*******************************************************************
C
C     Description :
C     ===========
C     Retrieve the value of matrix element (NR,NC) from database bank
C     containing "Matrix".
C
C     Input Parameter :
C     ===============
C     CHARACTER*2 IDXID ! Database Index
C     INTEGER  SOURCE   ! Calibration source, 0 = ONline, 1 = OFFline
C     INTEGER VERNUM    ! Version Number
C     INTEGER    RUNNUM ! Run Number
C     INTEGER    NTRIG  ! Trigger number
C     CHARACTER*4 BNKNA ! Bank name
C     INTEGER NR        ! Number of Row
C     INTEGER NC        ! Number of Column
C
C     Output parameters :
C     ==================
C     REAL VALUE ! value
C
C     Function values :
C     ===============
C                     GTDBML : Return Code given by CDUSEM, 0 if success
C                              (HEPDB Code)
*******************************************************************
      INTEGER FUNCTION GTDBMA(IDXID,SOURCE,VERNUM,RUNNUM,NTRIG,BNKNA,
     &                        NRI,NCI,VALUE,NR,NC)
*******************************************************************
C
C     Description :
C     ===========
C     Retrieve All values of a matrix NR*NC from database bank
C     containing "Matrix".
C
C     Input Parameter :
C     ===============
C     CHARACTER*2 IDXID ! Database Index
C     INTEGER  SOURCE   ! Calibration source, 0 = ONline, 1 = OFFline
C     INTEGER VERNUM    ! Version Number
C     INTEGER    RUNNUM ! Run Number
C     INTEGER    NTRIG  ! Trigger number
C     CHARACTER*4 BNKNA ! Bank name
C     INTEGER NRI       ! 1st dimension of the array VALUE
C     INTEGER NCI       ! 2nd dimension of the array VALUE
```

```
C
C     Output parameters :
C     ==================
C     REAL VALUE(NRI,NCI) ! value
C     INTEGER NR          ! Number of Row actually found
C     INTEGER NC          ! Number of Column actually found
C
C     Function values :
C     ===============
C                      GTDBMA : Return Code given by CDUSEM, 0 if success
C                               (HEPDB Code)
*******************************************************************
      INTEGER FUNCTION GTDBKS(IDXID,BNKNA,VERNUM,NTRIG,SOURCE)
*******************************************************************
C
C     Description :
C     ===========
C     Retrieve the last value of the HEPDB keys for the "version number"
C     (key number 14), the Run Type and the source
C     for a given object defined by the bank name and the HEPDB index.
C
C     Input Parameter :
C     ===============
C     CHARACTER*2 IDXID ! Database Index
C     CHARACTER*4 BNKNA ! Bank Name
C
C     Output parameters :
C     ==================
C     INTEGER VERNUM ! Last Run Number or Version number put in the Database
C     INTEGER NTRIG  ! Trigger number or Run Type
C     INTEGER SOURCE ! Source of the object (0 = Online, 1 = Offline)
C
C     Function values :
C     ===============
C                       GTDBKS : Return Code given by CDUSEM, 0 if success
C                               (HEPDB Code)
*******************************************************************
      INTEGER FUNCTION GTDBRN(IDXID,BNKNA,NPART,NTRIG,SOURCE,
     &                        NSERIE)
*******************************************************************
C
C     Description :
C     ===========
C     Retrieve an object define by the bank name and the 14th key
C     corresponding to the Particle Identifier or the Run Number
C     or a version number used to produce this object.
C
C     Input Parameter :
```

```
C     ===============
C     CHARACTER*2 IDXID ! Database Index
C     CHARACTER*4 BNKNA ! Bank Name
C     INTEGER NPART     ! Part Id or Run Number used for the calibration
C
C     Output parameters :
C     ==================
C     INTEGER NTRIG  ! Trigger number or Run Type
C     INTEGER SOURCE ! Source of the object (0 = Online, 1 = Offline)
C     INTEGER NSERIE ! Serial number of the object, equal 0 if it does not
C                    ! exist
C
C     Function values :
C     ===============
C                     GTDBRN : Return Code given by CDUSEM, 0 if success
C                              (HEPDB Code)
********************************************************************
      INTEGER FUNCTION GTDBCM(IDXDB,BNKNA,RUNNUM,DIMMAX,MAPFIL,DIMMAP)
********************************************************************
C
C     Description :
C     ===========
C     Retrieve the Detector MAP FILE (INTEGER*4) from the
C     database
C
C     Input Parameter :
C     ===============
C     CHARACTER*2 IDXDB ! Database Index
C     CHARACTER*4 BNKNA ! Bank name
C     INTEGER  RUNNUM   ! Run number
C     INTEGER DIMMAX    ! Maximum dimension for the map file.
C
C     Output parameters :
C     ==================
C     INTEGER MAPFIL(DIMMAP)! Map file array from the Run Condition Database.
C     INTEGER DIMMAP        ! Dimension of the map file (number of I*4) found
C                           ! (Minimum between DIMMAX and the real dimension)
C
C     Function values :
C     ===============
C                     GTDBCM : IRC Error return by
C                               0 if successful
********************************************************************
```

Two other routines are available to OPEN and CLOSE a database. These routines should be normally called at program or run initialisation (or program termination for the close) by ANALYSIS_CONTROL and NOT by the user. If user's program doesnot run in the context of ANALYSIS_CONTROL, here are the routines to open then close a database.

Routine to open a database

```
**********************************************************************
      INTEGER FUNCTION RTDBOP(DBIDX)
**********************************************************************
C
C     Description :
C     ===========
C     Open a database.
C
C     Input Parameter :
C     ===============
C      CHARACTER*2 DBIDX ! Database Index, CT for the Calibration test beam
C                        ! database, RT for the Run Condition database and
C                        ! GT for the Geometry test beam database
C
C     Output parameters : none
C     ==================
C
C     Function values :
C     ===============
C                       RTDBOP : IRC  Return Code of CDOPEN, 0 if success
C                                     (HEPDB Code)
**********************************************************************
```

Routine to close a database

```
**********************************************************************
      INTEGER FUNCTION RTDBCL(DBIDX)
**********************************************************************
C
C     Description :
C     ===========
C     Close a database.
C
C     Input Parameter :
C     ===============
C      CHARACTER*2 DBIDX ! Database Index, CT for the Calibration test beam
C                        ! database, RT for the Run Condition database and
C                        ! GT for the Geometry test beam database
C
```

```
C     Output parameters : none
C     ==================
C
C     Function values :
C     ===============
C                      RTDBOP : Return Code of CDCL, 0 if success
C                                         (HEPDB Code)
************************************************************************
```

References

[1] *HEPDB, Database Management Package, Reference Manual* , Application Software Group, Computers and Network Division, CERN Program Library Long Writeup Q180.

[2] *ZEBRA, Overview of the ZEBRA System*, CN, ECP and PPE Division, CERN Program Library Long Writeup Q100/Q101.