184208

R 193-04

Donohoo ED35

FLUI

ASE SYS

REPO

y 1991

red by:

Reardon

Violett J

ntract:

-37403

or:

and Space

l Space

ight Cent

Contents

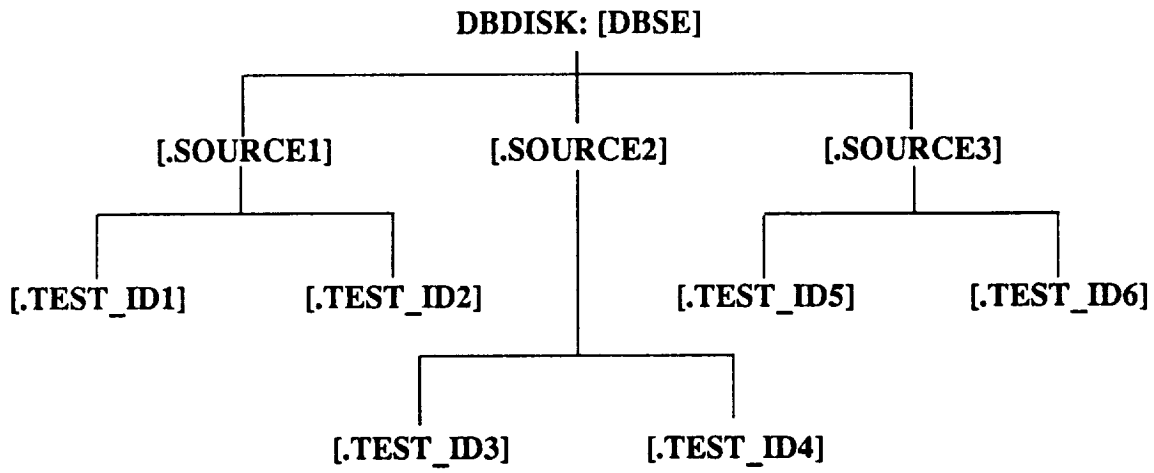
List of Tables

Section 1
INTRODUCTION

The AFAS Database System was developed to provide the basic structure of a comprehensive database system for the MSFC Structures and Dynamics Laboratory Aerophysics Division. The system is intended to handle all of the Aerophysics Division Test Facilities as well as data from other sources. The system has been written for the DEC VAX family of computers in FORTRAN -77 and utilizes the VMS indexed file system and screen management routines.

The database system recognizes three levels of interface to the system: the Database Manager who has overall system control, the Test Engineer who is responsible for the entry and validation of the data, and the User who accesses the data. Two types of test data are maintained: measured data with filenames using an "M" suffix, and engineering data with file names using an "E" suffix. The measured files contain test data. The engineering files contain data that have been reduced or manipulated by the Test Engineer.

The database directory structure has a root level directory with subdirectories representing each data source (facility). Each source directory contains subdirectories corresponding to specific tests. The directory tree structure is illustrated below.



At the highest directory level the system provides an index file to all tests available on the system. The directory for each test may contain a number of informational files. The four basic files involved in the stored data structure are: the VALUE file, LABEL file, the SET file, and the CHANGE LOG file.

The VALUE file stores test data as a matrix which can be visualized as columns representing measurements (or test parameters) and rows representing test conditions. The data values are stored in an indexed file using the data label describing the

measurement as the record key. The desired test condition is located by an offset in each record.

The LABEL file contains a record for each data label. Information in the record defines characteristics of the measured data values corresponding to the label. The purpose of this record is to provide the system and the user with information about the measurement. The structure of the label file record is defined by a PARAMETER file. This allows the Database Manager to specify field names and field lengths to be used for measurement description on a particular test. There are five required fields in the PARAMETER file : the key field ENGR_LABEL, which is an all uppercase representation of the data label; CASE_LABEL, which is the character representation of the label as the user will see it; DATATYPE which determines the data type of the label data; FLDLEN, which determines the data field length; DATA_SEGMENT used to tag the data retrieval path. Other optional fields might include: a short descriptor for the measurement, units for the measurement, or geometry values locating the measurement.

The SET file contains groupings of data labels which are desired for applications such as plotting of geometrically related data and sequencing of input data. This is an indexed file using the set name as the key. The set name is followed in the record by the X, Y, and Z parameter names. The balance of the record consists of fields of data label names and their associated X, Y, and Z parameter values.

The CHANGE LOG contains a history of modifications made to the VALUE file data. This is an indexed file using the data label as the key. A log entry consists of a data label and offset to determine the cell position, the original cell value, the number of modifications made, and, for each modification: the new value, the user name of process that modified the data, the date the cell was modified, and the time the cell was modified. The CHANGE LOG is used to track modifications made to the VALUE file and to retrieve previous values.

The balance of this document contains: a description of the user interface, lists of all code structure elements, descriptions of the file structures, a description of the security system operation, a detailed description of the data retrieval tasks, a description of the session log, and a description of the archival system.

Section 2
USER INTERFACE

The AFAS Database System uses a menu-driven interface to a network of data files and information files that can provide data on any test in the database. Database access is organized into three levels: Database Manager, Test Engineer, and User. The tasks associated with each level are summarized in Table 2.1.

When a user accesses the database system a top level menu (Table 2.2) lists the options for which the user is privileged. The User level functions are available to all users, but the database manager functions are not available to the Test Engineer or User access levels. Each of the options presented in the top level menu represent a separate executable file. There are a number of key functions that are common to all menus and may not be listed on the display due to space restrictions. Table 2.3 lists the universal key functions.

The user may obtain on-line help by entering "?" at any point where the system is requesting input. The information pertaining to the current option will be displayed on the screen. While viewing the on-line help, the user may obtain help information from subsequent option levels by typing the desired level option. A list of additional help items, if available, is displayed at the bottom of the help screen. The user exits the help facility by entering CNTRL/K. A CNTRL/K is entered by striking the "Ctrl" and "K" keys simultaneously.

Table 2.1: Database System Main Menu

To gain access to the database system enter "DBS." This will display the database options menu. The menu options vary according to the privilege assigned to the user gaining access. The complete menu contains the following options.

M — Management Services
E — Test Engineer Tasks
A — Archival Tasks
R — Test Data Load/Retrieval
T — View Data Table
V — View Reports
Select — Select/Browse Available Tests
Q — Quit

Option	Description
M	Executes the Database Manager Tasks.
E	Executes the Test Engineer Tasks.
A	Executes the Data Archival Tasks.
R	Executes the Data Retrieval Tasks which enable a user to create RS/1 compatible data files from the test data in the database.
T	Allows the user to view the VALUE file for the currently selected test.
V	Allows the user to view the available reports for the currently selected test.
Select	Allows the user to view a list of tests available on the system and to select one of these tests as the current test.
Q	Exits the AFAS Database System and returns control to the DCL environment.

Table 2.2: Database Function Summary

Database Manager Functions

- Maintain the database index file describing all tests in the system.
- Create parameter files which specify fields to be used in describing Labels.
- Create and load the "engineering" files.
- Remove invalid data from value files.
- Control archival of test data.
- Maintain database system security.
- Install text files for reports.

Test Engineer Functions

- Create and maintain the Label file of unique labels with descriptive information.
- Create and maintain the Set file to be used for associating plotting groups and specifying input formats.
- Mark invalid data in the value file.
- Rebuild the value file as necessary to maintain proper organization.
- Enter measured data into the value file.
- Prepare engineering data in RS/1 Tables which the Database Manager can load into the engineering value file.

User Level Functions

- Begin a new session or continue a previous data retrieval session by specifying the previous session name.
- View all data or a data summary to determine the data of interest.
- Select data by entering data labels and/or sets, or select the labels and sets from scrolled lists. The user may restrict the data retrieved for a label by entering ranges or requesting that data cells that do not have valid data be removed.
- Select output. The user has the option of creating a printed report of the data selected, loading the data into BBN RS/1 tables, creating meta-files, or storing the retrieval session in a session log.

Table 2.3: Universal Key Functions

Key Sequence	Result
CNTRL/K	Exit current menu.
CNTRL/E	Exit program.
"PF4"	Erase input field from cursor to end of field.
Left Arrow	Move left in input field.
Right Arrow	Move right in input field
"?"	Display help for current item or screen.

Notes: CNTRL/E may not be implemented at points where a valid input is necessary

Help may not be available where spawned tasks do not allow for trapping the "?" character.

Section 3
CODE STRUCTURE

The Database System is comprised of eight executable code sections. The source code for each section resides in a subdirectory of the directory DBDISK:[DBSE.SOURCE]. All routines common to two or more code sections reside in the library directory DBDISK:[DBSE.SOURCE.LIBRARY]. Table 3.1 lists the code sections and briefly describes each. A complete listing of all routines in the database system is provided in Appendix A. A listing of the arguments passed to each routine is provided in Appendix B.

A "make" facility has been provided to build the executable files when changes have been made to the source code. A make file has been provided for each executable file. The make files reside in the source directories and are named <executable_file>.MAK, where <executable_file> is the name of the executable. The files <executable_file>.COM and <executable_file>.OPT may also be present in the directory and are part of the make facility. The .COM file contains the link command. The .OPT file will be present whenever the link command in the .COM file exceeds the buffer capacity of the command interpreter (256 characters). The .OPT file will then be used to name the additional modules. After a source file has been modified, the executable is updated by typing "FMAKE <executable_name>". This will process the make file <executable_file>.MAK. Table 3.2 lists the symbol definitions required for database system maintenance. A brief description of the .MAK, .COM, and .OPT files follows.

<executable_file>.MAK

This file contains the commands for the make facility. The make facility looks at the target file date and checks the dependent file dates to see if any of the dependent files have later dates. If a dependent file date is more recent than the target file the next DCL command in the .MAK file is executed. If the date of the target file is the more recent the next DCL command in the .MAK file is skipped. The target files in the .MAK files are designated by the line "$!=<file_name>". The dependent files are designated by "$!><file_name>". The following is an example .MAK file that compiles and links the file TEST.FOR.

```
$!=TEST.OBJ
$!>TEST.FOR
$!>TEST.INC
$ FOR TEST
$!
$! Comment line
$!
$!=TEST.EXE
$!>TEST.OBJ
$ LINK TEST
```

<executable_file>.COM

This file contains the link command to build the executable file. This file is executed from within the .MAK file

<executable_file>.OPT

This file contains the link options for the link command. This file is specified in the .COM file. An options file will be present whenever the link command exceeded the buffer capacity of the command interpreter (256 characters).

Table 3.1: Database Code Sections

Code Section	Directory	Description
DBS_MGR	[...DBS_MGR	Database Manager source and executable.
DBS_DAT	[...DBS_DAT]	Test Engineer source and executable.
ARCHIVE	[...ARCHIVE]	Archival system source and executable.
USER	[...USER]	User source and executable.
DB_EDIT	[...DB_EDIT]	Tabled data display source and executable.
REPORTS	[...REPORTS]	Report viewer source and executable.
BROWSER	[...BROWSER]	Available tests browser source and executable.
DBS_LBR	[...DBS_LBR]	Source files for library routines.
MAIN_MENU	[...MAIN_MENU]	Main menu source and executable.

NOTE: [...<directory>] indicates that the directory resides under the parent directory DBDISK:[DBSE.SOURCE].

Table 3.2: Symbols and Logicals Used by Database

The AFAS Database System uses several logical and symbol definitions to locate the necessary files. DBDISK and DBS are required by all users. The remaining symbols and logicals are required for maintaining the code.

```
DBS == "@DBDISK:[DBSE.SOURCE.MAIN_MENU]MAIN_MENU"
FMAKE == "$DBDISK:[DBSE.SOURCE.MAKE]FMAKE"
DEFINE DBDISK <disk containing source code>
DEFINE INCLUDES DBDISK:[DBSE.SOURCE.INCLUDES]
DEFINE DBS_LBR DBDISK:[DBSE.SOURCE.DBS_LBR]
DBS_EDITOR == <session log editor>
```

DBS starts the AFAS Database System.
FMAKE executes the make facility.
DBDISK is the AFAS Database System "home" disk.
INCLUDES determines the location of the required include files.
DBS_LBR determines the location of the common subroutines and function.
DBS_EDITOR is a user-defined symbol that should point to the user's editor of preference. The editor should provide full-screen editing capability.

Section 4
FILE STRUCTURES

The AFAS Database System uses several file types to store test data and related information. Table 4.1 provides a list of the files and a brief description of their function. Appendix C describes each file, its record structure, and lists the FDL file and the structure description file, if one is available.

Table 4.1: System Files

Nomenclature:	SOURCE	— Facility or other data source, i.e., TWT.
	TEST	— Test identifier, i.e., 0023.
	*	— File type suffix: M — measured data E — engineering data
	<file>	— Denotes user specified path and name.
	<session>	— Denotes user-specified session name.

File Name and Path in DBDISK:[DBSE]	Description
System Files	
ARCHIVE.HLP	Help file for archival system.
DBS_DAT.HLP	Help file for Test Engineer code section.
DBS_MGR.HLP	Help file for Database Manager code section.
DB_EDIT.HLP	Help file for value file viewer/editor.
DATA_RETRIEVAL.HLP	Help file for User code section.
ARCH_MASTER.FDL	FDL file used to create master archival file.
CHANGE_LOG.FDL	FDL file used to create change log history files.
SET.FDL	FDL file used to create set files.
TEST.FDL	FDL file used to create test index file.
USER_LOG.FDL	FDL file used to create database user log.
VLU.FDL	FDL file used to create test value files.
ARCH_MASTER.FIL	Archive master file. Contains data archival information.
TEST.IND	Test index file. An entry is created for each test in the database.
Test-Specific Files	
[.SOURCE.TEST].CHL*	Value file change log. All changes made to value file are recorded.
[.SOURCE.TEST].GEO	Test geometry description file.
[.SOURCE.TEST].IND*	Label index file. Defines available labels for test data storage.
[.SOURCE.TEST].MRP*	Contains last offset position in value file.
[.SOURCE.TEST].PRM*	Label parameter file. Defines fields associated with the labels in the label index file.

Table 4.1: (Continued) System Files

File Name and Path in DBDISK:[DBSE]	Description
[.SOURCE.TEST].SET*	Data set file. Contains a record for each data set defined for the test.
[.SOURCE.TEST].VLU*	Test data value file. Contains actual test data.
User-Generated Files	
<file>.RS11	Value Table meta file. This is a meta file containing VALUE file data.
<file>.RS12	Set Table meta file. This is a meta file containing SET file data.
<file>.RS13	Label Table meta file. This is a meta file containing LABEL file data.
<session>.XSL	User-created session log. This file enables a user to resume a previous data retrieval session.

Section 5
SECURITY

The AFAS Database System provides three levels of access to the system: Database Manager, Test Engineer and Data-Retrieval User. The VAX system manager must assign identifiers to users requiring access higher than User level. The system manager creates the required identifiers by executing the Authorize utility from the SYS$SYSTEM directory. The commands for adding the required identifiers are:

```
UAF>add/identifier DB_MGR
UAF>add/identifier DB_TST
```

The identifier "DB_MGR" is required for a user to gain Database Manager access. The identifier "DB_TST" is required for a user to gain Test Engineer access. The system manager grants a user the required identifier by executing the authorize utility from the SYS$SYSTEM directory. The commands for granting the identifiers are:

```
UAF>grant/identifier DB_MGR user1
UAF>grant/identifier DB_TST user2
```

Where user1 and user2 are valid user names. In the above case "user1" would gain Database Manager privileges while "user2" would gain Test Engineer privileges.

The Database Manager may place restrictions on test data access by "protecting" data for specific tests. The test index file "TEST.IND" contains a record for each test loaded into the database. The "test_prot" field of the test index record is used to restrict User level access to test data. The following algorithm describes the test data protection scheme.

```
If "test_prot" field of test index record = "P" then

    If user holds "DB_MGR" identifier or
      user holds "DB_TST" identifier or
      username = "test_engr" field of test index record Then

      Access granted.

    Else
      If username = "test_cntr" field of test index record or
        user UIC = privileged group Then

        Access granted.

      Else
        If user holds identifier defined in "test_right"
          field of index record Then
```

```
            Access granted.

          Else
            Access denied.

          End if
        End if
      End if
    Else
      Access granted.

  End if
```

Since Test Engineers have access to all test data, it may be necessary to limit the ability of Test Engineers to modify test data. The following algorithm describes the requirements for test data write access by Test Engineers.

```
If user does not hold "DB_MGR" identifier Then

  If username same as "test_engr" field of test index record Then

    Write access granted.

  Else

    If user holds identifier defined in "test_right"
      field of index record Then

      Write access granted.

    Else
      Write access denied.

    End if
  End if

Else

  Write access granted.

End if
```

Section 6
DATA RETRIEVAL

The AFAS Database System provides a means of extracting data from the database in the USER code section (DATA_RETRIEVAL.EXE).

The data selected by the user are stored in two arrays - "sl_label_array" and "sl_relative_pos". The array "sl_label_array" is a list of data LABELS selected by the user. The array "sl_relative_pos" is a list of offsets that will be applied to the VALUE file record for each of the selected labels. There are other arrays that are maintained to enable the user to create or modify the data retrieval session.

When a LABEL is selected by the user, the corresponding VALUE file record is brought into memory. The offsets in "sl_relative_pos" are always applied to the VALUE file record currently in memory. The offsets in "sl_relative_pos" correspond to data cell locations in the VALUE file record. The structure is best described by a simple example.

Given the LABELS RUN, T1, T2

VALUE file records containing:

RUN = {10, 20, 30, 40, 50, 60, 70, 80, 90, 100}
T1 = {1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0}
T2 = {1.1, 2.2, 3.3, 4.4, 5.5, 6.6, 7.7, 8.8, 9.9, 10.1}

And the arrays:

"sl_label_array" = {"RUN", "T1", "T2"}
"sl_relative_pos" = {1, 3, 5, 7, 10}

Then the corresponding output table would contain:

RUN 10 30 50 70 100
T1 1.0 3.0 5.0 7.0 10.0
T2 1.1 3.3 5.5 7.7 10.1

The "sl_relative_pos" array can be restricted by:

1. Entering a range or series of ranges for the currently selected label. This will remove offsets in "sl_relative_pos" corresponding to values of the selected LABEL that are out of range.
2. Select "valid data only" for the currently selected LABEL. This will remove offsets in "sl_relative_pos" for which the selected label does not have data.
3. Combining 1 and 2.

The task which provides the Data Retrieval function (DATA_RETRIEVAL.EXE in the USER code section) needs to be understood thoroughly, so the algorithm for data selection and the corresponding routines are outlined on the following pages.

- Select option "R - Test Data Load/Retrieval" from the database system main menu. This starts the USER code section.

- Select option "D - Select Data" from the USER code main menu. Call UPDATE_DATA_POINTERS.

  Routine UPDATE_DATA_POINTERS :

  Call OPEN_DBSE to open the value file for the test selected.
  Call OPEN_SET_FILE to open set file for test selected.
  Display option menu to user.
  Display current number of labels and offsets selected.
  Prompt user for menu selection until CNTRL/K entered.

  - Select option "A - Select LABEL".
    Call VIEW_LABELS.
  - Select option "B - Enter LABEL".
    Enter label names until CNTRL/K entered.
    For each valid label entered call UPDATE_LABEL_POINTERS
  - Select option "C - List LABELs selected."
    Call LIST_LABELS.
  - Select option "D - Select SET."
    Call VIEW_SETS.
  - Select option "E - Enter SET."
    Enter set names until CNTRL/K entered.
    For each valid set entered call
    UPDATE_SET_POINTERS.
  - Select option "F - List SETs selected."
    Call LIST_SETS.
  - Select option "G - Edit current session."
    Call WRITE_SESSION_FILE to write current session log.
    Call EDIT_SESSION_FILE.

  Close value file.
  Close set file.
  Return to caller.

  End routine UPDATE_DATA_POINTERS.

ROUTINES

```
Routine ADD_UNBOUNDED_LABEL:
    If label not in "sl_label_array"
      Increment sl_num_labels.
      Load label into "sl_label_array".
      Load "sl_no_range" flag into "sl_lower_bound" array.
      Load "sl_no_range" flag into "sl_upper_bound" array.
      Set "sl_ranges(label_pos).num_ranges" to zero.
      Set "sl_ranges(label_pos).nonulls" to .FALSE.
    End if
    Return to caller.
  End routine ADD_UNBOUNDED_LABEL.

  Routine CVT_TOKEN_ATTR:
    Convert string to appropriate binary data type.
    Load I2_TYPE, I4_TYPE, R4_TYPE, R8_TYPE or string size for
      character data into "type".
    Return to caller.
  End routine CVT_TOKEN_ATTR.

  Routine EDIT_SESSION_FILE:
    Get editor symbol for user-defined editor.
    Spawn edit command.
    Call INIT_SESSION.
    Call READ_SESSION_FILE.
    Return to caller.
  End routine EDIT_SESSION_FILE.

  Routine ENTER_BOUND:
    Read input value string.
    Call CVT_TOKEN_ATTR to convert string to proper data type.
    Return to caller.
  End routine ENTER_BOUND.

  Routine ENTER_RANGE:
    While last character of bound value .ne. "+" do
      Prompt user for upper and lower bound values.
      Call ENTER_BOUND.
      Increment "n_bnd".
      Increment "bkt_pos".
    End do
    Load "sl_ranges(label_pos).num_ranges" + 1 into "lab_rng".
```

```
      Load "n_bnd" into "sl_ranges(label_pos).num_bnds(lab_rng)".
      Load "range_bkt_size" + 1 into
"sl_ranges(label_pos).rng_ptr(lab_rng)".
      Load "range_bkt_size" + "n_bnd" into "range_bkt_size".
      Load "lab_rng" into "sl_ranges(label_pos).num_ranges".
      Return to caller.
    End routine ENTER_RANGE.

    Routine FINISH_SESSION_FILE_SCAN.
      Close session log file.
      Return to caller.
    End routine FINISH_SESSION_FILE_SCAN.

    Routine GET_SET_LABELS:
      If set name has been loaded previously return to caller.
      Increment "sl_num_sets".
      Add set name to "sl_set_array" session log common.
      For each label in set file call ADD_UNBOUNDED_LABEL.
      Display current number of labels and offsets selected.
      Return to caller.
    End routine GET_SET_LABELS.

    Routine INITIALIZE_SESSION_FILE_SCAN.
      Get LUN for session log file.
      Open session log file.
      If cannot open session file then free LUN.
      Set "scan_cur_pos" to zero.
      Set "scan_length" to -1.
      Set "scan_line_num" to zero.
      Return to caller (Return open status).
    End routine INITIALIZE_SESSION_FILE_SCAN.

    Routine INIT_SESSION:
      Set "sl_num_labels" to zero.
      Set "sl_num_rrefs" to zero.
      Set "sl_num_sets" to zero.
      Set "sl_init_label" to .TRUE.
      Set "range_bkt_size" to zero.
    End routine INIT_SESSION.

    Routine LIST_LABELS:
      Display labels and ranges in current session.
      Allow scrolling of list.
      If an entry is deleted then
```

```
      Remove "sl_label_array" entry for current label.
      Remove "sl_upper_bound" entry for current label.
      Remove "sl_lower_bound" entry for current label.
      Remove "sl_ranges" entry for current label.
      Decrement "sl_num_labels".
    End if
End routine LIST_LABELS.

Routine LIST_SETS:
    Display sets in current session.
    Allow scrolling of list.
    If an entry is deleted then
      Remove "sl_set_array" entry for current set.
      Decrement "sl_num_sets".
    End if
End routine LIST_SETS.

Routine PROCESS_LABEL:
    Load "sl_upper_bound(label_pos)" and
        "sl_lower_bound(label_pos)" with appropriate flags or
        range values.
    If label is first label to be processed then
      Build "sl_relative_pos" array for current label and range.
    Else if label range specified
      Set "sl_relative_pos" array offsets that are out of
          range to zero.
    End if
    Call SHIFT_ARRAY to rid "sl_relative_pos" array of zero
        valued entries.
    If measured data accessed
      Call CHECK_VCOUNT to get rid of any "sl_relative_pos"
          offsets with valid count nonzero.
    End if
    Return to caller.
End routine PROCESS_LABEL.

Routine READ_SESSION_FILE:
    Call INITIALIZE_SESSION_FILE_SCAN.
    Read label from session file.
    Do while labels present in session file.
      If label is not in list call ADD_UNBOUNDED_LABEL.
      Increment "sl_ranges(lab).num_ranges".
      Load "sl_ranges(lab).num_ranges" into "rng".
      Set "sl_ranges(lab).num_bnds(rng)" to zero.
```

```
      Load "range_bkt_size" + 1 into "sl_ranges(lab).rng_ptr(rng)".
      Do while ranges present for current label.
        Increment "sl_ranges(lab).num_bnds(rng)".
        Load "sl_ranges(lab).num_bnds(rng)" into "bnd".
        Increment "range_bkt_size".
        Store current lower bound string.
        Call CVT_TOKEN_ATTR to convert string to proper type.
        If upper bound specified then
          Store current upper bound string.
          Call CVT_TOKEN_ATTR to convert string to proper type.
        Else
          Load current lower bound into upper bound.
        End if
      End do
      If current label is initial label call PROCESS_LABEL.
      If "valid data only" specified call REMOVE_INVALID_OFFSETS.
    End do
    Call FINISH_SESSION_FILE_SCAN.
    Return to caller.
End routine READ_SESSION_FILE.

Routine REMOVE_INVALID_OFFSETS:
    Load "sl_label_array(label_pos)" into "label_id".
    Make sure data for "label_id" is in memory.
    Remove offsets in "sl_relative_pos" that do not have valid data.
    Shift "sl_relative_pos" array to remove the invalid cells.
    Decrement "sl_num_rrefs" to reflect new "sl_relative_pos" array size.
    Return to caller.
End routine REMOVE_INVALID_OFFSETS.

Routine TOGGLE_NONULLS:
    Set "sl_ranges(label_pos).nonulls" to logical opposite.
    Return to caller.
End routine TOGGLE_NONULLS.

Routine UPDATE_LABEL_POINTERS:
    If wildcard character in label name
      For each label matching wildcard search call ADD_UNBOUNDED_LABEL
    Else
```

```
        Call ADD_UNBOUNDED_LABEL.
        Prompt user for range specification.
        If range specified call ENTER_RANGE.
        If "Valid Data Only" is specified call TOGGLE_NONULLS.
        If label loaded is first to be loaded or a range has been specified call
PROCESS_LABEL.
        If "valid data only" specified call REMOVE_INVALID_OFFSETS.
      End if
      Display current number of labels and offsets selected.
      Return to caller.
    End routine UPDATE_LABEL_POINTERS.

    Routine UPDATE_SET_POINTERS:
      If wildcard in set name
        For each set matching wildcard search call GET_SET_LABELS.
      Else if "ALL" specified
        For each set label in set file call GET_SET_LABELS.
      Else
        Call GET_SET_LABELS.
      End if
      Return to caller.
    End routine UPDATE_SET_POINTERS.

    Routine VIEW_LABELS:
      Display list of available labels.
      Select label from available labels list until CNTRL/K entered.
      If label selected from list call UPDATE_LABEL_POINTERS.
      Return to caller.
    End routine VIEW_LABELS.

    Routine VIEW_SETS:
      Display list of available sets.
      Select set from available sets list until CNTRL/K entered.
      If set selected from list call UPDATE_SET_POINTERS.
      Return to caller.
    End routine VIEW_SETS.
```

Section 7
THE SESSION LOG

The session log enables a user to store the current retrieval session, resume a previously stored retrieval session, or modify a retrieval session. When a user begins a data retrieval session all information pertaining to the current session is maintained in memory. The user may select to store the current session at any time or may edit the information stored in memory for the current session. Appendix D describes the session log file format.

- Storing a data retrieval session.

  The user may store the current retrieval session by selecting option "S" from the main menu of the USER code. The user will be prompted for the session name. The extension should not be part of the session name. A file <session_name>.SL2 will be created and will contain all information required to resume the session. The user will also have the option to create the session log when terminating the retrieval session.

- Editing a retrieval session.

  The user may modify the contents of the session log by selecting option "G" from the "Data Selection" menu. The "Data Selection" menu is presented when option "D" is selected from the USER code main menu. When option "G" is selected the system spawns the editor defined by the symbol "DBS_EDITOR" and opens the session log file. The session log may also be modified offline by any text editor.

- Resuming a data retrieval session.

  When starting a data_retrieval session, the user is prompted for "New" or "Previous" session. If previous session is selected the user may enter a valid session name or select a session from a list of available sessions. When a valid session name is entered the information contained in the session log is loaded into memory. The user may now resume the session from the point that it was stored.

- Initializing the current retrieval session.

  The user may initialize the current retrieval session at any time by selecting option "I" from the USER code main menu. This will return the retrieval session to the "New" session state.

Section 8
DATA ARCHIVAL

As the volume of data in the AFAS Database System grows it will be necessary to archive tests that are no longer accessed on a regular basis. An archival facility has been provided to enable the Database Manager to archive selected tests to optical media and/or magnetic tape.

The archival system allows the Database Manager to archive selected tests and either delete or retain the archived test in an active status. After a test has been successfully archived, a record* is entered into the archival master file (DBDISK:[DBSE]ARCH_MASTER.FIL). The test index record* in DBDISK:[DBSE]TEST.IND corresponding to the archived test is also updated to reflect the status of the archived data. The first six characters of the archival media label are entered into the "arch_m_vol" field of the test index record. The status of the archived data is recorded in the "arch_m_flag" field of the test index record.

The possible values for "arch_m_flag" are:

0 - Data not archived
1 - Data archived and retained
2 - Data archived and deleted

If engineering data are available for the archived test the fields "arch_e_vol" and "arch_e_flag" will be updated as well.

If a user selects a test that has been archived and deleted, a message will notify the user that the requested test has been archived and is not available. The user should notify the Database Manager to have the test restored.

The Database Manager restores an archived test by selecting the restore option from the archival menu. If a test has been archived more than once, a list of archival sets is presented. The Database Manager selects the appropriate archival set from this list.

* All record fields for all files are defined in Appendix C.

Appendix A
DATABASE SYSTEM MODULE LISTING

Nomenclature : The field "Section" refers to the directory as described in Table 3.1

Routine Name	Source File	Section	Description
ARCHIVE MAIN	ARCHIVE_MAIN.FOR	ARCHIVE	Main routine for ARCHIVE code section.
BROWSER MAIN	BROWSER.FOR	BROWSER	Main routine for BROWSER code section.
DBS_DAT MAIN	DBSE.FOR	DBS_DAT	Main routine for DBS_DAT code section.
DBS_MGR MAIN	MANAGER.FOR	DBS_MGR	Main routine for DBS_MGR code section.
DISPLAY MAIN	DISPLAY.FOR	REPORTS	Main routine for REPORTS code section.
MAIN_MENU MAIN	GET_OPTION.FOR	MAIN_MENU	Main routine for MAIN_MENU code section.
USER MAIN	DATA_RETRIEVAL.FOR	USER	Main routine for USER code section.
function ACCESS_OKAY	ACCESS_OKAY.FOR	DBS_LBR	Determine if user has access to specified test.
subroutine ACCESS_TEST_FILE	ACCESS_TEST_FILE.FOR	DBS_LBR	Provide interface to TEST.IND file.
subroutine ADD_UNBOUNDED_LABEL	ADD_UNBOUNDED_LABEL.FOR	USER	Add label with no range specification.
subroutine ADD_VALUE	LINKLIST.FOR	DBS_DAT	Add a value to the linked list.
subroutine ADJUST_LIST	ADJUST_LIST.FOR	DBS_DAT	Find label in list and display it.
subroutine ADJUST_LIST_U	ADJUST_LIST_U.FOR	USER	Find label in list and display it.
subroutine ADJUST_REPORTS	ADJUST_REPORTS.FOR	REPORTS	Scoll list of available reports and select one.
subroutine ALTER_DATABASE_REC	DB_EDIT.FOR	DBS_LBR	Update value file record.
subroutine ARCHIVE_DATA	ARCHIVE.FOR	ARCHIVE	Execute mount/dismount and archive commands.
subroutine BUILD_BACKUP_CMD	ARCHIVE.FOR	ARCHIVE	Build archive request backup command.
subroutine BUILD_INPUT_FILE	BUILD_INPUT_FILE.FOR	USER	Build RS/1 meta files.
subroutine BUILD_RESTORE_CMD	BUILD_RESTORE_CMD.FOR	ARCHIVE	Create archive restore command.
subroutine BUILD_SAVESET	ARCHIVE.FOR	ARCHIVE	Create archive save set name.
subroutine BUILD_SESSION_FILE	BUILD_SESSION_FILE.FOR	USER	Build session log *.XSL.
function CARRAY_BSEARCH	CARRAY_BSEARCH.FOR	DBS_LBR	Perform binary search on sorted string array.
subroutine CALL_DB_EXIT	CALL_DB.FOR	DB_EDIT	DB_EDIT Exit handler.
function CHAR_MATCH	CHAR_MATCH.FOR	DBS_LBR	See if character string and byte array match.
function CHAR_VAL	RANGE_CHECK.FOR	DBS_LBR	Convert a byte array to a character string.
function CHECK_FIELDS	ARCHIVE.FOR	ARCHIVE	Verify required fields present.
subroutine CHECK_VCOUNT	PROCESS_LABEL.FOR	USER	Remove offsets with VALID COUNT > 0.
subroutine CLEAR_BUF	SCROLL_VALIDATION.FOR	DBS_DAT	Clear buffer by setting to nulls.
subroutine CLEAR_DBSE_REC	RECORD.FOR	DBS_LBR	Load value file record with nulls.
subroutine CLEAR_LABEL_FIELD	LABEL.FOR	DBS_LBR	Load specified label record field with blanks.
function CLEAR_QUEUE	QUEUE.FOR	DBS_LBR	Clear RAB queue.
subroutine CLEAR_RECORD	LABEL.FOR	DBS_LBR	Load label record with blanks.
subroutine CLEAR_REC_FIELD	RECORD.FOR	DBS_LBR	Load value file record field with blanks.
function CLEAR_STACK	STACK.FOR	DBS_LBR	Clear RAB stack.
subroutine CLOSE_USER_LOG	USER_LOG.FOR	DBS_LBR	Close user log file.
function CMD_INDEX	GET_OPTION.FOR	MAIN_MENU	Determine the index of the DCL command.
subroutine COMBINE_SETS	COMBINE_SETS.FOR	DBS_DAT	Combine two or more sets into single set.
subroutine COMPLEMENT_CELL	COMPLEMENT_CELL.FOR	DBS_LBR	Toggle valid data cell flag.
subroutine COMPRESS	COMPRESS.FOR	DBS_LBR	Remove imbedded blanks and nulls from string.
subroutine CREATE_ARCHIVAL_SCREEN	CREATE_ARCH_SCREENS.FOR	ARCHIVE	Create archival command echo screen.
subroutine CREATE_CH_LOG_SCREEN	CREATE_CH_LOG_SCREEN.FOR	DBS_LBR	Create screen for viewing change log entries.
subroutine CREATE_D	CREATE_D.FOR	DBS_MGR	Create the screen for the "D" main menu option.
subroutine CREATE_DATA_INPUT_SCREEN	CREATE_DATA_INPUT_SCREEN.FOR	DBS_DAT	Create screen for "I" main menu option.
subroutine CREATE_GET_SCREEN	CREATE_ARCH_SCREENS.FOR	ARCHIVE	Create archive store prompt screen.
subroutine CREATE_HELP	CREATE_HELP.FOR	DBS_LBR	Create main help screen.
subroutine CREATE_KEY_SCROLL	CREATE_KEY_SCROLL.FOR	DBS_LBR	Create screen for scrolling available key values.
subroutine CREATE_L	CREATE_L.FOR	DBS_DAT	Create screen for "L" main menu option.

Routine Name	Source File	Section	Description
subroutine CREATE_MAIN_MENU	CREATE_ARCH_SCREENS.FOR	ARCHIVE	Create archive main menu screen.
subroutine CREATE_P	CREATE_P.FOR	DBS_MGR	Create the screen for the "P" main menu option.
subroutine CREATE_R	CREATE_R.FOR	DBS_MGR	Create the screen for the "R" main menu option.
subroutine CREATE_RECORD	CREATE_RECORD.FOR	DBS_MGR	Prompt user for test index record fields.
subroutine CREATE_RPT_SCREEN	CREATE_ARCH_SCREENS.FOR	ARCHIVE	Create archival report prompt screen.
subroutine CREATE_RPT_SCROLL	CREATE_RPT_SCROLL.FOR	ARCHIVE	Create archival report scrolling file.
subroutine CREATE_RS1_TABLES	CREATE_RS1_TABLES.FOR	USER	Load selected data into RS/1 tables.
subroutine CREATE_RTR_GET_SCREEN	CREATE_ARCH_SCREENS.FOR	ARCHIVE	Create archive restore prompt screen.
subroutine CREATE_SCN	DATA_RETRIEVAL.FOR	USER	Create main menu screen.
subroutine CREATE_SCRN	CREATE_SCRN.FOR	REPORTS	Create display for REPORTS code section.
subroutine CREATE_SCROLL	CREATE_SCROLL.FOR	DBS_MGR	Create test index record screen scrolling region.
subroutine CREATE_SCROLL_SCREEN	CREATE_ARCH_SCREENS.FOR	ARCHIVE	Create archive entries scrolling screen.
subroutine CREATE_SOURCE_SCREEN	CREATE_SOURCE_SCREEN.FOR	DBS_LBR	Create screen for entering source/testid.
subroutine CREATE_SOURCE_SCROLL	CREATE_SOURCE_SCROLL.FOR	DBS_LBR	Create screen for scrolling available tests.
subroutine CURSOR_DISPLAY	CURSOR_DISPLAY.FOR	DBS_LBR	Turn cursor on/off.
subroutine CVT_BYTES_TO_STR	CVT_BYTES_TO_STR.FOR	DBS_LBR	Convert byte array to string.
subroutine CVT_FIELD_TO_STR	CVT_FIELD_TO_STR.FOR	DBS_LBR	Convert numerical data to string.
subroutine CVT_I4_TO_STR	WRITE_SESSION_FILE.FOR	USER	Convert four byte integer to character string.
subroutine CVT_R8_TO_STR	WRITE_SESSION_FILE.FOR	USER	Convert eight byte real data to character string.
subroutine CVT_STR_TO_FIELD	CVT_STR_TO_FIELD.FOR	DBS_LBR	Convert string to appropriate data type.
subroutine CVT_STR_TO_NUM	CVT_STR_TO_NUM.FOR	DBS_LBR	Convert string to numerical data.
subroutine CVT_TOKEN_ATTR	READ_SESSION_FILE.FOR	USER	Load input field into appropriate byte field.
function DATE_VALID	GET_RPT_FIELDS.FOR	ARCHIVE	Verify that date field is valid.
subroutine DATLAB	DATLAB.FOR	DBS_DAT	Option "L" of main menu - Label Maintenance.
subroutine DATSET	DATSET.FOR	DBS_DAT	Option "S" of main menu - Set Maintenance.
subroutine DBS_EXIT	DBS_EXIT.FOR	DBS_DAT	DBS_DAT exit handler.
subroutine DB_EDIT	DB_EDIT.FOR	DBS_LBR	Read/Write editor for value file.
subroutine DELETE_DBSE_REC	RECORD.FOR	DBS_LBR	Delete a value file record.
subroutine DELETE_LABEL_RECORD	LABEL.FOR	DBS_LBR	Delete current label record.
function DELETE_QUEUE_POSITION	QUEUE.FOR	DBS_LBR	Remove entry from RAB queue.
subroutine DELETE_SET_REC	SET_DATA.FOR	DBS_LBR	Delete a set file record.
function DEVICE_SUPPORTED	ARCHIVE.FOR	ARCHIVE	Determine if requested device is supported.
subroutine DISPLAY_FIELD	DISPLAY_FIELD.FOR	DBS_DAT	Display label field.
subroutine DISPLAY_FILE	DISPLAY.FOR	REPORTS	Display selected report.
subroutine DISPLAY_MONITOR	ARCHIVE.FOR	ARCHIVE	Display current monitor mode.
subroutine DISPLAY_MSGS	ARCHIVE.FOR	ARCHIVE	Display completion status message.
subroutine DOWN	DISPLAY.FOR	REPORTS	Scroll display down
subroutine EDIT_SESSION_FILE	EDIT_SESSION_FILE.FOR	USER	Spawn editor to edit session log.
function EMPTY_STACK	STACK.FOR	DBS_LBR	See if RAB stack is empty.
subroutine ENTER_BOUND	ENTER_BOUND.FOR	USER	Read range value and convert to appropriate type.
subroutine ENTER_RANGE	ENTER_RANGE.FOR	USER	Prompt user for range values.
subroutine ERASE_GET_SCREEN	ARCHIVE.FOR	ARCHIVE	Erase prompt screen.
subroutine ERASE_HELP_REGION	HELP.FOR	DBS_LBR	Erase the help message display.
subroutine ERASE_IO_REGION	ERASE_IO_REGION.FOR	DBS_LBR	Erase io region of display.
subroutine ERASE_OPTION_REGION	ERASE_OPTION_REGION.FOR	DBS_LBR	Erase options region of display.
subroutine ERASE_OPTION_REGION2	ERASE_OPTION_REGION2.FOR	REPORTS	Erase menu option region.

Routine Name	Source File	Section	Description
subroutine ERASE_SCROLL_REGION	ERASE_SCROLL_REGION.FOR	DBS_LBR	Erase scrolling region of display.
function ERROR_HANDLE	ERROR_HANDLE.FOR	DBS_LBR	Error handling routine.
subroutine FIND_NUMBER_STOP	SCANNER.FOR	USER	Find end of a numerical field.
subroutine FIND_TYPES	FIND_TYPES.FOR	DBS_LBR	Determine available extension types for test.
function FIND_VALUE	FIND_VALUE.FOR	DBS_LBR	Search for requested value in value file record.
function FINISH_SESSION_FILE_SCAN	SCANNER.FOR	USER	End session file scan - close session log file.
subroutine FORMATTER	FORMATTER.FOR	DBS_LBR	Create output format string for REAL*8 value.
function FULL_STACK	STACK.FOR	DBS_LBR	See if RAB stack is full.
subroutine GEN_REPORT	GEN_REPORT.FOR	USER	Generate report of selected data to output file.
subroutine GEODAT	GEODAT.FOR	DBS_DAT	Define/display geometry data.
function GET_ARCHIVE_DATA	ARCHIVE.FOR	ARCHIVE	Get archival store fields.
subroutine GET_ARY_FIELD	SCROLL_VALIDATION.FOR	DBS_DAT	Load buffer with bytes from value file record.
subroutine GET_CNTRLK	GET_CNTRLK.FOR	USER	Intercept unsolicited input.
subroutine GET_DISK_NAME	GET_DISK_NAME.FOR	DBS_LBR	Get data storage device name for test.
function GET_FIND_RANGE	GET_FIND_RANGE.FOR	DBS_LBR	Prompt user for label, hi/low range and direction.
subroutine GET_FIRST_KEY	POSITION_TEST_FILE.FOR	DBS_LBR	Read first valid key from test index file.
subroutine GET_INPCONERR_FILE	ERROR_HANDLE.FOR	DBS_LBR	Get error message string.
subroutine GET_LABEL_FIELD	LABEL.FOR	DBS_LBR	Return specified label record field.
function GET_MAIN_OPTION	ARCHIVE.FOR	ARCHIVE	Get main archive menu option.
subroutine GET_NEW_MIN	RANGE_CHECK.FOR	DBS_LBR	Determine a new min for the current value record.
subroutine GET_NEW_MAX	RANGE_CHECK.FOR	DBS_LBR	Determine a new max for the current value record.
function GET_NEXT_ENTRY	POSITION_TEST_FILE.FOR	DBS_LBR	Get next record from test index file.
function GET_NEXT_KEY	POSITION_TEST_FILE.FOR	DBS_LBR	Get next key value from test index file.
subroutine GET_NUM_ENTRIES	ARCHIVE.FOR	ARCHIVE	Create scroll file of matching entries.
subroutine GET_NUMBER	GET_NUMBER.FOR	USER	Read numerical field from display.
subroutine GET_RECORD	ARCHIVE_MAIN.FOR	ARCHIVE	Load requested archive record into memory.
subroutine GET_REC_FIELD	RECORD.FOR	DBS_LBR	Get a value file record field.
function GET_RELATIVE_POS	ARCHIVE.FOR	ARCHIVE	Determine relative volume position of save set.
function GET_RPT_FIELDS	GET_RPT_FIELDS.FOR	ARCHIVE	Get archive report generation fields.
function GET_RTR_FIELDS	ARCHIVE.FOR	ARCHIVE	Get archival restore fields.
subroutine GET_SAVESET_EXT	ARCHIVE.FOR	ARCHIVE	Create unique save set extension.
subroutine GET_SET_LABELS	GET_SET_LABELS.FOR	USER	Load labels from set into selected label array.
subroutine GET_SET_VARS	DATSET.FOR	DBS_DAT	Define X, Y and Z set parameter field names.
function GET_SOURCE_TESTID	GET_SOURCE_TESTID.FOR	DBS_LBR	Get a valid source/testid/type.
subroutine GET_SUBPID	ARCHIVE.FOR	ARCHIVE	Get subprocess PID.
subroutine GET_TRGT_DEV	ARCHIVE.FOR	ARCHIVE	Get target device for restore operation.
subroutine GET_TYPE	GET_TYPE.FOR	DBS_LBR	Determine the data type to be used (M/E), etc.
subroutine GET_VAR_VALS	DATSET.FOR	DBS_DAT	Get X, Y and Z set parameter field values.
subroutine HELP	HELP.FOR	DBS_LBR	Display available help messages.
subroutine HLI_PROCESS_TABLE2	HLI_PROCESS_TABLE2.FOR	DBS_MGR	Load set data from RS/1 table 2.
subroutine HLI_PROCESS_TABLE3	HLI_PROCESS_TABLE3.FOR	DBS_MGR	Load data labels from RS/1 table 3.
function I2_VAL	RANGE_CHECK.FOR	DBS_LBR	Convert a two-byte array to a two-byte integer.
function I4_RUN	SCROLL_VALIDATION.FOR	DBS_DAT	Convert RUN number to integer*4.
function I4_VAL	RANGE_CHECK.FOR	DBS_LBR	Convert a four-byte array to a four-byte integer.
function INCREASING	INCREASING.FOR	DBS_LBR	Determine if a value is between ranges.
subroutine INIT_SESSION	INIT_SESSION.FOR	USER	Initialize session log information fields.
function INITIALIZE_SESSION_FILE_SCAN	SCANNER.FOR	USER	Init. session file scan - open session log file.
subroutine INSERT_PARM_DATA	INSERT_PARM_DATA.FOR	DBS_MGR	Load parameter data into parameter file.
function KEYCHANGE	CREATE_RPT_SCROLL.FOR	ARCHIVE	Determine if key field is still valid.
subroutine LABEL_PARAM	LABEL_PARAM.FOR	DBS_MGR	Create/View/Modify parameter files.

Routine Name	Source File	Section	Description
subroutine LEVEL_OF_ACCESS	LEVEL_OF_ACCESS.FOR	DBS_LBR	Determine user's level of access.
subroutine LINE2TESTREC	SCROLL_TESTS.FOR	DBS_MGR	Copy active screen record to test index record.
subroutine LIST2ARRAY	LINKLIST.FOR	DBS_DAT	Copy the linked list to a character string array.
subroutine LIST_LABELS	LIST_LABELS.FOR	USER	Display labels selected during retrieval session.
subroutine LIST_SETS	LIST_SETS.FOR	USER	Display sets selected during retrieval session.
subroutine LOAD_LABEL_LIST	LOAD_LABEL_LIST.FOR	DBS_DAT	Create list of available labels.
subroutine LOAD_RS1_DATA	LOAD_RS1_DATA.FOR	DBS_LBR	Load RS/1 meta-files into database.
subroutine LOAD_RS1_TABLES	LOAD_RS1_TABLES.FOR	DBS_MGR	Load RS/1 tables into database.
subroutine LOAD_SET_VAR_VALS	DATSET.FOR	DBS_DAT	Get default X, Y and Z set parameter field values.
subroutine LOG_MOD	LOG_MOD.FOR	DBS_LBR	Record a cell modification in change log.
subroutine MANAGE_ENGR_FILES	MANAGE_ENGR_FILES.FOR	DBS_MGR	Create "E" type - engineering data files.
subroutine MGR_EXIT	MGR_EXIT.FOR	DBS_MGR	DBS_MGR exit handler.
subroutine MIN_MAX_SCAN	MIN_MAX_SCAN.FOR	USER	Display min/max values for test data labels.
subroutine MIN_MAX_SCAN2	MIN_MAX_SCAN2.FOR	DBS_DAT	Display min and max values for each test label.
subroutine MODIFY_LABEL_DESCR	MODIFY_LABEL_DESC.FOR	DBS_DAT	View/Modify label parameter field values.
subroutine MOVE_BYTES	LOG_MOD.FOR	DBS_LBR	Copy bytes from one location to another.
subroutine MOVE_MEASURED_DATA	MOVE_MEASURED_DATA.FOR	DBS_MGR	Copy "M" information files into "E" region.
subroutine NEW_SPEC	NEW_SPEC.FOR	DBS_LBR	Determine file specs for new file.
function NEXT_TOKEN	SCANNER.FOR	USER	Read next field from session log file.
subroutine NULLIFY_LIST	LINKLIST.FOR	DBS_DAT	Initialize the linked list.
subroutine OPEN_ARCH_FILE	ACCESS_ARCH.FOR	ARCHIVE	Open archival record file.
subroutine OPEN_CH_LOG_FILE	ACCESS_CH_LOG.FOR	DBS_LBR	Open change log file.
subroutine OPEN_DBSE	RECORD.FOR	DBS_LBR	Open the specified value file.
subroutine OPEN_LABEL_FILE	OPEN_LABEL_FILE.FOR	DBS_LBR	Open label data file.
subroutine OPEN_SET_FILE	SET_DATA.FOR	DBS_LBR	Open set data file.
subroutine OPEN_USER_LOG	USER_LOG.FOR	DBS_LBR	Open user log file.
subroutine OPTION_D	OPTION_D.FOR	DBS_MGR	Option "D" of main menu - test index file.
subroutine OUTPUT_REPORT	OUTPUT_REPORT.FOR	USER	Prompt user for type of report to be generated.
function POP	STACK.FOR	DBS_LBR	Pop entry from RAB stack.
function POP_FIRST_POSITION	QUEUE.FOR	DBS_LBR	Pop entry from RAB stack and put in queue.
subroutine POSITION_DEVICE	ARCHIVE_MAIN.FOR	ARCHIVE	Position archival media to specified file ptr.
subroutine POSITION_TEST_FILE	POSITION_TEST_FILE.FOR	DBS_LBR	Set test index file to specified record and read.
subroutine PRINT_REPORT	PRINT_REPORT.FOR	ARCHIVE	Write archive report to ARCH_REPORT.RPT.
subroutine PROCESS_LABEL	PROCESS_LABEL.FOR	USER	Restrict offsets for label and range.
subroutine PROCESS_SESSION_FILE	PROCESS_SESSION_FILE.FOR	USER	Load and process a session log file.
subroutine PROCESS_TABLE2	PROCESS_TABLE2.FOR	DBS_LBR	Load the set data meta-file into database.
subroutine PROCESS_TABLE3	LOAD_RS1_DATA.FOR	DBS_LBR	Update label file with any new labels.
subroutine PURGE_RUNS	PURGE_RUNS.FOR	DBS_MGR	Remove specified runs from data value file.
function PUSH	STACK.FOR	DBS_LBR	Push entry on RAB stack.
function PUSH_FIRST_POSITION	QUEUE.FOR	DBS_LBR	Remove element from RAB queue and push on stack.
subroutine PUT_ARY_FIELD	SCROLL_VALIDATION.FOR	DBS_DAT	Load value file record with buffer contents.
subroutine PUT_BYTES	SCROLL_VALIDATION.FOR	DBS_DAT	Convert buffer to string.
subroutine PUT_INVALID_FIELD	RECORD.FOR	DBS_LBR	Load a value file record field with invalid data.
subroutine PUT_KEY_SCROLL_OPTIONS	PUT_KEY_SCROLL_OPTIONS.FOR	DBS_LBR	Create key scrolling display.
subroutine PUT_LABEL_FIELD	LABEL.FOR	DBS_LBR	Load value into specified label record field.
subroutine PUT_MENU	GET_OPTION.FOR	MAIN_MENU	Display main menu options.
subroutine PUT_OPTION_LIST	VALID_OPTION.FOR	DBS_LBR	Display a list of valid options.
subroutine PUT_PARAM_DATA	LABEL_PARAM.FOR	DBS_MGR	Display data from specified parameter file.
subroutine PUT_REC_FIELD	RECORD.FOR	DBS_LBR	Load a value file record field with valid data.
subroutine PUT_SUBOPTION_LIST	VALID_OPTION.FOR	DBS_LBR	Display a list of valid suboptions.

Routine Name		Source File	Section	Description
function	QUERY_BOUND	QUERY_BOUND.FOR	DBS_LBR	Get a boundary value (high/low).
function	R4_VAL	RANGE_CHECK.FOR	DBS_LBR	Convert a four-byte array to a four-byte real.
function	R8_VAL	RANGE_CHECK.FOR	DBS_LBR	Convert an eight-byte array to an eight-byte real.
subroutine	RANGE_CHECK	RANGE_CHECK.FOR	DBS_LBR	Determine new max/min for current value record.
subroutine	READ_ARCH_REC	ACCESS_ARCH.FOR	ARCHIVE	Read archival record.
subroutine	READ_CH_LOG_REC	ACCESS_CH_LOG.FOR	DBS_LBR	Read change log record.
subroutine	READ_DBSE_REC	RECORD.FOR	DBS_LBR	Read value file record.
subroutine	READ_DBSE_REC_HDR	RECORD.FOR	DBS_LBR	Read value file record header.
subroutine	READ_FIELDS	ARCHIVE.FOR	ARCHIVE	Read archival request fields.
subroutine	READ_FIRST_DBSE_REC	RECORD.FOR	DBS_LBR	Position value file pointer to first record.
subroutine	READ_LABEL	LABEL.FOR	DBS_LBR	Read label file record.
function	READ_SESSION_FILE	READ_SESSION_FILE.FOR	USER	Load session log into memory.
subroutine	READ_SET_REC	SET_DATA.FOR	DBS_LBR	Read a set file record.
subroutine	READ_SET_REC_HDR	SET_DATA.FOR	DBS_LBR	Read a set file record header.
subroutine	READ_USER_LOG	USER_LOG.FOR	DBS_LBR	Read a user log entry.
subroutine	REDRAW_SCREEN	REDRAW_SCREEN.FOR	USER	Repaint current display.
subroutine	REFRESH_MIN_MAX_SCREEN	REFRESH_MIN_MAX_SCREEN.FOR	DBS_DAT	Rewrite fields on min max display.
subroutine	REMOVE_DATA	REMOVE_DATA.FOR	DBS_MGR	Allow user to remove/modify data in value file.
subroutine	REMOVE_INVALID_OFFSETS	REMOVE_INVALID_OFFSETS.FOR	USER	Remove invalid data offsets from offset array.
function	REMOVE_LAST_QUEUE_ENTRY	QUEUE.FOR	DBS_LBR	Remove last entry from RAB queue.
function	RENAME_LABEL	RENAME_LABEL.FOR	DBS_DAT	Rename a label in the appropriate info. files.
function	REPLACE_QUEUE_ENTRY	QUEUE.FOR	DBS_LBR	Put entry in specified RAB queue position.
subroutine	REP_EXIT	REP_EXIT.FOR	REPORTS	Reports display exit handler.
function	RESET_POSITION	QUEUE.FOR	DBS_LBR	Pop RAB stack and set file to that position.
subroutine	RESTORE	ARCHIVE_MAIN.FOR	ARCHIVE	Generate archival restore commands.
subroutine	RESUME_SESSION	RESUME_SESSION.FOR	USER	Resume a previously stored session.
subroutine	RES_PARM_FILE	RES_PARM_FILE.FOR	DBS_LBR	Calculate byte position of fields in param. file.
function	RS1_COLNUM	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
function	RS1_CLOSE_TABLE	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
function	RS1_DELETE_TABLE	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
function	RS1_FINISH	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
function	RS1_GET_CELL	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
function	RS1_INIT	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
function	RS1_LASTCOL	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
function	RS1_LASTROW	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
function	RS1_LOCK	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
function	RS1_MAKE_DIR	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
function	RS1_MAKE_TABLE	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
function	RS1_OPEN_TABLE	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
function	RS1_SET_CELL	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
function	RS1_SET_DIR	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
function	RS1_TABLEEXISTS	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
function	RS1_UNLOCK	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
subroutine	RS1__BADOPTIONS	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
subroutine	RS1__CANTDELETE	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
subroutine	RS1__DISKSPACELOW	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
subroutine	RS1__EMPTY	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
subroutine	RS1__INVALIDCOLNUM	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
subroutine	RS1__INVALIDROWNUM	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.

Routine Name	Source File	Section	Description
subroutine RS1__NOGROUPDIR	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
subroutine RS1__NOSUCHDIR	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
subroutine RS1__NOTATEXT	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
subroutine RS1__NOTINGROUP	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
subroutine RS1__NOUSERDIR	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
subroutine RS1__PREVIOUSINIT	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
function RS1__SUCCESS	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
subroutine RS1__TABLEEOF	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
subroutine RS1__USERDIRLOCKED	RS1_FAKE.FOR	DBS_LBR	RS/1 compatibility test routine.
subroutine SAVE_FIELD	SAVE_FIELD.FOR	DBS_DAT	Store a value into the label parameter value field.
function SAVE_POSITION	QUEUE.FOR	DBS_LBR	Add current RAB to RAB queue.
subroutine SCAN_LIST	WILDCARD.FOR	DBS_LBR	Determine if an entry is in array of strings.
subroutine SCROLL_KEYS	SCROLL_SOURCES.FOR	DBS_LBR	Scroll available key values.
subroutine SCROLL_SOURCES	SCROLL_SOURCES.FOR	DBS_LBR	Scroll available test entries.
subroutine SCROLL_TESTS	SCROLL_TESTS.FOR	DBS_MGR	Scroll/Modify records in test index file.
subroutine SCROLL_VALIDATION	SCROLL_VALIDATION.FOR	DBS_DAT	Display RUN,RERUN,FRAME and VALID COUNT for test.
subroutine SELECT_MATCH	SELECT_MATCH.FOR	DBS_DAT	Display list entries matching a requested pattern.
subroutine SESS_FILE_READ_ERR	READ_SESSION_FILE.FOR	USER	Create session file read error message.
subroutine SET_DBSE_REC	RECORD.FOR	DBS_LBR	Set value file record pointer.
subroutine SET_FILE	SET_FILE.FOR	DBS_LBR	Set file pointer to a specific address.
subroutine SET_FILE_POINTER	SCROLL_TESTS.FOR	DBS_MGR	Set test index file pointer to specified record.
subroutine SET_MIN_MAX	SET_MIN_MAX.FOR	DBS_LBR	Preset min and max values.
function SET_MODE	SET_MODE.FOR	DBS_LBR	Set SMG pasteboard attributes.
function SET_PROT_USEROPEN	SET_PROT_USEROPEN.FOR	DBS_LBR	Define the protection mask for created file.
subroutine SET_VAL	SET_VAL.FOR	DBS_LBR	Update "VALID COUNT" entry .
subroutine SET_VALIDITY	SET_VALIDITY.FOR	DBS_DAT	Increment "VALID COUNT" if runs are reloaded.
subroutine SET_VOLPOS	ARCHIVE_MAIN.FOR	ARCHIVE	Verify and set volume to correct position.
subroutine SHIFT_ARRAY	PROCESS_LABEL.FOR	USER	Shift offset array to remove nonzero entries.
subroutine SKIP_WHITE_SPACE	SCANNER.FOR	USER	Skip over "white space" in session log file.
subroutine SORTER	SORTER.FOR	DBS_MGR	Sort the test index records.
subroutine SORT_LABEL_ARRAY	BUILD_INPUT_FILE.FOR	USER	Sort selected label array.
subroutine STAT_MSG	STAT_MSG.FOR	DBS_LBR	Signal SMG function completion errors.
subroutine STOP_PROC	ARCHIVE.FOR	ARCHIVE	Force executing image to exit.
function STRLEN	WILDCARD.FOR	DBS_LBR	Determine string length without trailing balnks.
subroutine SWAP	SCROLL_VALIDATION.FOR	DBS_DAT	Load buffer with contents of input buffer.
subroutine TOGGLE_NONULLS	UPDATE_LABEL_POINTERS.FOR	USER	Toggle "valid data only" retrieval flag.
subroutine TESTREC2LINE	SCROLL_TESTS.FOR	DBS_MGR	Copy test index record to screen display buffer.
function TEST_ENGR_ACCESS	TEST_ENGR_ACCESS.FOR	DBS_LBR	Determine if user has Test Engineer access.
subroutine UNLOCK_DBSE_REC	RECORD.FOR	DBS_LBR	Unlock value file record.
subroutine UNLOCK_LABEL_RECORD	LABEL.FOR	DBS_LBR	Unlock label file record.
subroutine UP	DISPLAY.FOR	REPORTS	Scroll display up.
subroutine UPCASE	UPCASE.FOR	DBS_LBR	Convert string to uppercase.
subroutine UPDATE_DATA_POINTERS	UPDATE_DATA_POINTERS.FOR	USER	Menu for data label/set view and selection.
function UPDATE_FILE_POSITION	QUEUE.FOR	DBS_LBR	Position file to specified address.
subroutine UPDATE_LABEL_POINTERS	UPDATE_LABEL_POINTERS.FOR	USER	Process user-selected label.
subroutine UPDATE_SCREEN	LOAD_RS1_DATA.FOR	DBS_LBR	Notify user of number of labels/frames loaded.
subroutine UPDATE_SELECT_LINE	UPDATE_SELECT_LINE.FOR	DBS_LBR	Update current source/testid/type line.
subroutine UPDATE_SET_POINTERS	UPDATE_SET_POINTERS.FOR	USER	Get selected set and corresponding labels.
subroutine UPDATE_TEST_FILE	UPDATE_TEST_FILE.FOR	ARCHIVE	Update archival fields in "TEST.IND" file.

Routine Name	Source File	Section	Description
subroutine UPDATE_WINDOW	UPDATE_WINDOW.FOR	DBS_LBR	Update window and scrolling queue.
subroutine USER_EXIT	USER_EXIT.FOR	USER	User code exit handler.
function VALID	GET_OPTION.FOR	MAIN_MENU	Determine if selected option is valid.
function VALID_OPTION	VALID_OPTION.FOR	DBS_LBR	Determine if the selected option is valid.
subroutine VERIFY_VOLUME	VERIFY_VOLUME.FOR	ARCHIVE	Verify mounted volume has correct volume name.
subroutine VIEW_CHANGE_LOG	VIEW_CHANGE_LOG.FOR	DBS_LBR	Display change log entry for current cell.
subroutine VIEW_LABELS	VIEW_LABELS.FOR	USER	Display list of available data labels.
subroutine VIEW_REPORT	VIEW_REPORT.FOR	ARCHIVE	Display archival report on screen.
subroutine VIEW_SETS	VIEW_SETS.FOR	USER	Display list of available data sets.
subroutine VIRTUAL_INPUT	VIRTUAL_INPUT.FOR	DBS_LBR	Get a string from keyboard input.
subroutine WEED_OUT_POS	BUILD_INPUT_FILE.FOR	USER	Remove invalid data offsets.
function WILDCARD	WILDCARD.FOR	DBS_LBR	Search string array using wildcards "*".
subroutine WRITE_ARCH_REC	ACCESS_ARCH.FOR	ARCHIVE	Write archival record.
subroutine WRITE_BUFFER	VIEW_CHANGE_LOG.FOR	DBS_LBR	Write value buffer to output buffer.
subroutine WRITE_CH_LOG_REC	ACCESS_CH_LOG.FOR	DBS_LBR	Write a change log record.
subroutine WRITE_DBSE_REC	RECORD.FOR	DBS_LBR	Write/rewrite value file record.
subroutine WRITE_INVTEX_MSG	ERROR_HANDLE.FOR	DBS_LBR	Write invalid text message.
subroutine WRITE_INVTEXREC_MSG	ERROR_HANDLE.FOR	DBS_LBR	Write invalid text encountered in record message.
subroutine WRITE_LABEL	LABEL.FOR	DBS_LBR	Write/rewrite a label file record.
subroutine WRITE_SESSION_FILE	WRITE_SESSION_FILE.FOR	USER	Write retrieval session data to session file.
subroutine WRITE_SET_REC	SET_DATA.FOR	DBS_LBR	Write/rewrite a set file record.
subroutine WRITE_USER_LOG	USER_LOG.FOR	DBS_LBR	Write a user log entry.

Appendix B
DATABASE SYSTEM ROUTINE ARGUMENT LISTING

Nomenclature : The "Type" field denotes the data type and field size.

Cn	- Character string of size n.
C*	- Character string with passed length.
I2	- Two-byte integer value.
I4	- Four-byte integer value.
R4	- Four-byte real value.
R8	- Eight-byte real value.
L1	- One-byte logical value.
L	- Compiler default sized logical value.
Record	- Record structure.

The "Access" field denotes the routine's access to the argument.

R	- Read only access - argument not modified.
W	- Write only access - argument modified.
R/W	- Read/Write access - argument modified.

Routine Name		Argument	Type	Access	Description
ARCHIVE	MAIN	None			
DB_EDIT	MAIN	None			
DBS_DAT	MAIN	None			
DBS_MGR	MAIN	None			
DISPLAY	MAIN	None			
MAIN_MENU	MAIN	None			
USER	MAIN	None			
function	ACCESS_OKAY	in_source	C8	R	Source name string.
		in_test_id	C8	R	Test id name string.
subroutine	ACCESS_TEST_FILE	lun	I4	R	Test file access LUN.
		function	I2	R	I/O access function (defined in INCLUDES:file_io_def.inc)
		stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
		status	I4	W	System completion status.
		key_mode	I2	R	Key type (Defined in INCLUDES:file_io_def.inc)
		key_len	I2	R	Length of passed key field.
		rdkey	C*	R	Key value.
subroutine	ADD_UNBOUNDED_LABEL	label_id	C*	R	Label name string.
		label_pos	I4	W	Position of label in "sl_label_array".
subroutine	ADD_VALUE	new_value	C12	R	Value to be added to linked list.
subroutine	ADJUST_LIST	scrn_id	I4	R	SMG display screen id.
		label_value	C12	R	Label name string.
		list_pos	I2	R/W	Position of label in "label_list".
		list_size	I2	R	Size of "label_list".
		label_list	C12	R	List of label names.
		active_row	I4	R/W	Current display row.
		scan_stat	I2	W	Status of label list scan.
		terminator	I2	W	Terminator encountered in SMG read.
		scrn_top_pos	I2	R/W	"label_list" index at top of screen.
		scrn_btm_pos	I2	R/W	"label_list" index at bottom of screen.
		line_length	I2	R	Length of display line.
		minmax	L1	R	Calling routine flag - TRUE if caller is MIN_MAX_SCAN.
		info1	C*	R	Character array to be viewed with each label.
		info2	C*	R	Character array to be viewed with each label.
		info1_elem_size	I2	R	Length of info1 character string.
		dbse_lun	I4	R	Value file access LUN.

Routine Name	Argument	Type	Access	Description
subroutine ADJUST_LIST_U	scrn_id	I4	R	SMG display screen id.
	label_value	C12	R	Label name string.
	list_pos	I2	R/W	Position of label in "label_list".
	list_size	I2	R	Size of "label_list".
	label_list	C12	R	List of label names.
	active_row	I4	R/W	Current display row.
	scan_stat	I2	W	Status of label list scan.
	terminator	I2	W	Terminator encountered in SMG read.
	scrn_top_pos	I2	R/W	"label_list" index at top of screen.
	scrn_btm_pos	I2	R/W	"label_list" index at bottom of screen.
	line_length	I2	R	Length of display line.
	minmax	L1	R	Calling routine flag - TRUE if caller is MIN_MAX_SCAN.
	info1	C*	R	Character array to be viewed with each label.
	info2	C*	R	Character array to be viewed with each label.
	info1_elem_size	I2	R	Length of info1 character string.
	dbse_lun	I4	R	Value file access LUN.
subroutine ADJUST_REPORTS	scrn_id	I4	R	SMG display screen id.
	label_value	C12	R	Report name string.
	list_pos	I2	R/W	Position of report in "label_list".
	list_size	I2	R	Size of "label_list".
	label_list	C12	R	List of report names.
	active_row	I4	R/W	Current display row.
	scan_stat	I2	W	Status of report list scan.
	terminator	I2	W	Terminator encountered in SMG read.
	scrn_top_pos	I2	R/W	"label_list" index at top of screen.
	scrn_btm_pos	I2	R/W	"label_list" index at bottom of screen.
	line_length	I2	R	Length of display line.
subroutine ALTER_DATABASE_REC	dbse_lun	I4	R	Value file access LUN.
	cs_lab	C*	R	Label name of record to be rewritten.
	new_dbse_rec	B	R	Array containing new value record information.
	max_changed_offset	I4	R	Number of offsets loaded from new value record.
	new_cell_count	I4	R	Number of data cells in new value record.
subroutine ARCHIVE_DATA	pbd_id	I4	R	SMG pasteboard id
	main_id	I4	R	SMG display screen id.
	arch_id	I4	R	SMG display screen id.
	monitor	L1	R	Display archival commands to screen.
subroutine BUILD_BACKUP_CMD	backup_cmd	C255	W	Archive command.
subroutine BUILD_INPUT_FILE	scrn_id	I4	R	SMG display screen id.
	kbd_id	I4	R	SMG keyboard id.
	pbrd_id	I4	R	SMG pasteboard id
subroutine BUILD_RESTORE_CMD	restore_cmd	C255	W	Archive restore command.
subroutine BUILD_SAVESET	None			
subroutine BUILD_SESSION_FILE	scrn_id	I4	R	SMG display screen id.
	kbd_id	I4	R	SMG keyboard id.
	pbrd_id	I4	R	SMG pasteboard id
	previous_screen	L1	W	Return to previous screen flag.
	session_name	C12	W	Session file name.
function CARRAY_BSEARCH	array	C*	R	Sorted array of character strings.
	num_elements	I4	R	Number of elements in "array".
	target	C*	R	String to be found.
subroutine CALL_DB_EXIT	None			

Routine Name		Argument	Type	Access	Description
function	CHAR_MATCH	target_string	C*	R	Character string to be compared.
		source_string	Record	R	Byte array to be compared.
		source_len	I2	R	Number of characters to compare.
function	CHAR_VAL	cell	Record	R	Defines a value file data cell.
function	CHECK_FIELDS	get_id	I4	R	SMG display screen id.
		first_row	I4	R	First row of field input.
		last_row	I4	R	Last row of field input.
		prompt_col	I4	R	Column at which field data starts.
subroutine	CHECK_VCOUNT	dbse_lun	I4	R	Value file access LUN.
subroutine	CLEAR_BUF	byte_buf	B	W	Byte array to be cleared.
		len	I2	R	Number of bytes to clear.
subroutine	CLEAR_DBSE_REC	stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
subroutine	CLEAR_LABEL_FIELD	field	C*	R	Name of field to be cleared.
		stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
function	CLEAR_QUEUE	None			
subroutine	CLEAR_RECORD	stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
subroutine	CLEAR_REC_FIELD	offset	I4	R	Position in value file record to be cleared.
		stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
function	CLEAR_STACK	None			
subroutine	CLOSE_USER_LOG	log_lun	I4	R	User log access LUN.
function	CMD_INDEX	terminator	I2	W	Terminator encountered in SMG read.
subroutine	COMBINE_SETS	set_file	C*	R	Set file name.
		set_lun	I4	R	Set file access LUN.
		pbrd_id	I4	R	SMG pasteboard id
		scrn_id	I4	R	SMG display screen id.
		kbd_id	I4	R	SMG keyboard id.
subroutine	COMPLEMENT_CELL	new_dbse_rec	B	R/W	Byte array containing the value file record.
		offset	I4	R	Cell offset position in "new_dbse_rec"
		cell_count	I4	R/W	Number of cells loaded into "new_dbse_rec".
		key_label	C12	R	Key field for value file record.
		dbse_lun	I4	R	Value file access LUN.
subroutine	COMPRESS	string	C*	R/W	Character string to be compressed.
subroutine	CREATE_ARCHIVAL_SCREEN				
		arch_id	I4	W	SMG display screen id.
subroutine	CREATE_CH_LOG_SCREEN				
		ch_log_id	I4	W	SMG display screen id.
subroutine	CREATE_D	scrn_id	I4	W	SMG display screen id.
subroutine	CREATE_DATA_INPUT_SCREEN				
		scrn_id	I4	W	SMG display screen id.
subroutine	CREATE_GET_SCREEN	get_id	I4	W	SMG display screen id.
subroutine	CREATE_HELP	scrn_id	I4	W	SMG display screen id.
subroutine	CREATE_KEY_SCROLL	scrn_id	I4	W	SMG display screen id.
subroutine	CREATE_L	scrn_id	I4	W	SMG display screen id.
subroutine	CREATE_MAIN_MENU	main_id	I4	W	SMG display screen id.
		monitor	L1	R	Display archival commands to screen.
subroutine	CREATE_P	scrn_id	I4	W	SMG display screen id.
subroutine	CREATE_R	scrn_id	I4	W	SMG display screen id.
subroutine	CREATE_RECORD	pbrd_id	I4	R	SMG pasteboard id
		scrn_id	I4	R	SMG display screen id.
		kbd_id	I4	R	SMG keyboard id.
		valid	B	W	Valid input flag.
subroutine	CREATE_RPT_SCREEN	rpt_id	I4	W	SMG display screen id.

Routine Name	Argument	Type	Access	Description
subroutine CREATE_RPT_SCROLL	error_cond	B	W	Completion status - not defined in include file.
subroutine CREATE_RS1_TABLES	scrn_id	I4	W	SMG display screen id.
subroutine CREATE_RTR_GET_SCREEN				
	rtg_id	I4	W	SMG display screen id.
subroutine CREATE_SCN	scrn_id	I4	W	SMG display screen id.
subroutine CREATE_SCRN	scrn_id	I4	W	SMG display screen id.
subroutine CREATE_SCROLL	scrn_id	I4	W	SMG display screen id.
subroutine CREATE_SCROLL_SCREEN				
	scroll_id	I4	W	SMG display screen id.
subroutine CREATE_SOURCE_SCREEN				
	scrn_id	I4	W	SMG display screen id.
subroutine CREATE_SOURCE_SCROLL				
	scrn_id	I4	W	SMG display screen id.
subroutine CURSOR_DISPLAY	mode	C	R	Flag to turn cursor on/off. On='h', Off='l'.
	pbrd_id	I4	R	SMG pasteboard id
subroutine CVT_BYTES_TO_STR	string	C*	W	String buffer to hold converted data.
	strlength	I4	W	Number of valid characters in "string".
	bytes	B	R	Byte array to be converted.
	form	C	R	Data type of "bytes" array.
	length	I2	R	Number of valid bytes in "bytes" array.
subroutine CVT_FIELD_TO_STR	field	Record	R	Data field to be converted.
	field_len	I2	R	Number of bytes in data field.
	field_form	C	R	Data type of data field.
	read_stat	I2	R	Completion status of previous READ.
	ret_val	C*	W	Converted value string.
	out_len	I4	W	Number of valid characters in "ret_val".
subroutine CVT_I4_TO_STR	num	I4	R	Integer value to be converted to string.
	str	C*	W	Character string that will hold converted integer.
	len_str	I4	W	Length of converted string field.
subroutine CVT_R8_TO_STR	num	R8	R	Real value to be converted to string.
	str	C*	W	Character string that will hold converted integer.
	len_str	I4	W	Length of converted string field.
subroutine CVT_STR_TO_FIELD	string	C*	R	String containing value to be converted.
	strlength	I4	R	Number of valid characters in "string".
	bytes	B	W	Byte array containing converted field.
	form	C	R	Data type of data field ("bytes").
	length	I4	R	Length of data field ("bytes").
	cvt_stat	I4	W	Completion status of internal READ.
subroutine CVT_STR_TO_NUM	string	C*	R	String value to be converted.
	strlength	I4	R	Number of valid characters in "string".
	num	Record	W	Converted value.
	form	C	R	Data type of data field.
	length	I2	R	Number of bytes in data field.
	cvt_stat	I4	W	Completion status of internal READ.
subroutine CVT_TOKEN_ATTR	str	C*	R	Field value string.
	elem	Record	W	Field element.
	form	C1	R	Format specifier.
	size	I2	R	Field size.
	type	B	W	Data type.
	stat	I4	W	Conversion completion status.
function DATE_VALID	date	C9	R	Date to be validated.

Routine Name	Argument	Type	Access	Description
subroutine DATLAB	pbrd_id	I4	R	SMG pasteboard id
	scrn_id	I4	R	SMG display screen id.
	kbd_id	I4	R	SMG keyboard id.
subroutine DATSET	pbrd_id	I4	R	SMG pasteboard id
	scrn_id	I4	R	SMG display screen id.
	kbd_id	I4	R	SMG keyboard id.
subroutine DBS_EXIT	None			
subroutine DB_EDIT	QUIT_RTN	External		Address of exit handler.
	view_only	L1	R	View/Edit data flag.
	ch_log_id	I4	R	SMG display screen id.
subroutine DELETE_DBSE_REC	dbse_lun	I4	R	Value file access LUN.
	stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
subroutine DELETE_LABEL_RECORD	lab_lun	I4	R	Label file access LUN.
	stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
function DELETE_QUEUE_POSITION				
	position	I4	R	Queue position to be deleted.
subroutine DELETE_SET_REC	set_lun	I4	R	Set file access LUN.
	stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
function DEVICE_SUPPORTED	device_name	C*	R	Name of archival device.
subroutine DISPLAY_FIELD	record	I2	R	Label field position.
	row	I4	R	Display output row.
	col	I4	R	Display output column.
	scrn_id	I4	R	SMG display screen id.
subroutine DISPLAY_FILE	file_name	C*	R	Name of file to be displayed.
	scrn_id	I4	R	SMG display screen id.
	kbd_id	I4	R	SMG keyboard id.
	pbrd_id	I4	R	SMG pasteboard id
subroutine DISPLAY_MONITOR	main_id	I4	R	SMG display screen id.
	monitor	L1	R	Display archival commands to screen.
subroutine DISPLAY_MSGS	main_id	I4	R	SMG display screen id.
	monitor	L1	R	Display archival commands to screen.
subroutine DOWN	num_rows	I4	R	Number of rows to scroll down.
	rab_addr	I4	R	Address of RAB record.
	lun	I4	R	Scroll file access LUN.
	scrn_id	I4	R	SMG display screen id.
subroutine EDIT_SESSION_FILE	fname	C*	R	Session file name.
	pbrd_id	I4	R	SMG pasteboard id
	kbd_id	I4	R	SMG keyboard id.
	scrn_id	I4	R	SMG display screen id.
	stat	I4	W	Completion status.
	dbse_lun	I4	R	Value file access LUN.
function EMPTY_STACK	None			

Routine Name	Argument	Type	Access	Description
subroutine ENTER_BOUND	str	C*	R	Input string field.
	which	I4	R	Upper/Lower bound specifier.
	bkt_pos	I4	R	Position in range bucket.
	exit_flg	L1	W	User requested exit.
	form	C1	R	Format specifier.
	size	I2	R	Field size.
	help_key	C*	R	Help key for current routine.
	help_lev	I4	R	Help level of current routine.
	EXIT_RTN	External		Address of exit handler.
	scrn_id	I4	R	SMG display screen id.
	pbrd_id	I4	R	SMG pasteboard id
	kbd_id	I4	R	SMG keyboard id.
	row	I4	R	Row for string prompt.
	col	I4	R	Column for string prompt.
	err_row	I4	R	Row for error message.
	err_col	I4	R	Column for error message.
subroutine ENTER_RANGE	scrn_id	I4	R	SMG display screen id.
	kbd_id	I4	R	SMG keyboard id.
	pbrd_id	I4	R	SMG pasteboard id
	label_pos	I4	R	Position of label in "sl_label_array".
	lower_bound	C12	W	Lower bound string value.
	upper_bound	C12	W	Upper bound string value.
subroutine ERASE_GET_SCREEN	get_id	I4	R	SMG display screen id.
	first_row	I4	R	First row of region to be erased.
	last_row	I4	R	Last row of region to be erased.
	prompt_col	I4	R	Display column at which prompt is displayed.
subroutine ERASE_HELP_REGION	scrn_id	I4	R	SMG display screen id.
	pbrd_id	I4	R	SMG pasteboard id
subroutine ERASE_IO_REGION	scrn_id	I4	R	SMG display screen id.
subroutine ERASE_OPTION_REGION	scrn_id	I4	R	SMG display screen id.
subroutine ERASE_OPTION_REGION2				
	scrn_id	I4	R	SMG display screen id.
subroutine ERASE_SCROLL_REGION	scrn_id	I4	R	SMG display screen id.
function ERROR_HANDLE	signal_args	I4	R	Error condition array.
	mech_args	I4	?	Compatibility variable.
subroutine FIND_NUMBER_STOP	line	C*	R	Input line for processing.
	cur_pos	I4	R	Current line position.
	stop_pos	I4	W	Line position at which processing terminated.
subroutine FIND_TYPES	prompt_str	C*	W	Prompt string.
	len_str	I4	W	Length of prompt string.
function FIND_VALUE	current_pos	I2	R	Current search list position.
	direction	C	R	Search direction.
	field_form	C	R	Data field type.
	field_len	I4	R	Dat field size.
	low_buf	Record	R	Low bound.
	high_buf	Record	R	High bound.
	value_buf	B	R	Value to be found.
	value_buf_size	I4	R	Size of data field to be found.
	value_field_len	I4	?	Not currently used.
function FINISH_SESSION_FILE_SCAN				
	None			

Routine Name	Argument	Type	Access	Description
subroutine FORMATTER	real_8	R8	R	Data to be formatted.
	size	I2	R	Data field size
	form	C6	W	Output format for WRITE statement.
	stat	I2	W	Completion status.
function FULL_STACK	None			
subroutine GEN_REPORT	refs	Record	R	Offsets for output.
	num_refs	I2	R	Number of offsets for output.
	labels	C*	R	Array of labels for output.
	num_labels	I2	R	Number of entries in "labels".
	dbse_lun	I4	R	Value file access LUN.
	refs_flag	I2	R	"refs" array data type flag.
	scrn_id	I4	R	SMG display screen id.
	kbd_id	I4	R	SMG keyboard id.
	pbrd_id	I4	R	SMG pasteboard id
subroutine GEODAT	pbrd_id	I4	R	SMG pasteboard id
	scrn_id	I4	R	SMG display screen id.
	kbd_id	I4	R	SMG keyboard id.
function GET_ARCHIVE_DATA	pbd_id	I4	R	SMG pasteboard id
	kbd_id	I4	R	SMG keyboard id.
	main_id	I4	R	SMG display screen id.
	get_id	I4	R	SMG display screen id.
subroutine GET_ARY_FIELD	offset	I4	R	Offset in value record.
	buffer	B	W	Target byte array buffer to hold field.
	size	I2	R	Number of bytes in field.
	valid	L1	R	Cell validity flag.
	rec	B	R	Byte array containing the value file record.
	stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
subroutine GET_CNTRLK	pbrd_id	I4	R	SMG pasteboard id
	kbd_id	I4	R	SMG keyboard id.
subroutine GET_DISK_NAME	in_source	C8	R	Source name string.
	in_test_id	C8	R	Test id name string.
	disk	C*	W	Storage disk name.
	stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
function GET_FIND_RANGE	fnd_scrn_id	I4	R	SMG display screen id.
	scrn_id	I4	R	SMG display screen id.
	pbrd_id	I4	R	SMG pasteboard id
	kbd_id	I4	R	SMG keyboard id.
	sea_label	C*	R	Current label.
	sea_field_form	C	R	Current label data field format.
	sea_field_len	I2	R	Current label data field length.
	low_buf	Record	W	Low range buffer.
	high_buf	Record	W	High range buffer.
	QUIT_RTN	External		Address of exit handler.
subroutine GET_FIRST_KEY	test_lun	I4	R	Test index file access LUN.
	test_stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
	key_pos	I2	R	Key position of "key"
	key	C*	R	Key value for TEST.IND file read.
subroutine GET_INPCONERR_FILE	str1	C*	W	Target string.
	str2	C*	R	Source string.

Routine Name	Argument	Type	Access	Description
subroutine GET_LABEL_FIELD	field	C*	R	Field name to be retrieved.
	buffer	B	W	Byte buffer to receive field.
	size	I2	R	Number of bytes in field.
	stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
function GET_MAIN_OPTION	main_id	I4	R	SMG display screen id.
	kbd_id	I4	R	SMG keyboard id.
subroutine GET_NEW_MIN	None			
subroutine GET_NEW_MAX	None			
function GET_NEXT_ENTRY	test_lun	I4	R	Test index file access LUN.
function GET_NEXT_KEY	test_lun	I4	R	Test index file access LUN.
	test_stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc).
	key_pos	I2	R	Key position of "key".
	key	C*	R	Key value for TEST.IND read.
subroutine GET_NUM_ENTRIES	error_cond	B	W	Completion status - not defined in include file.
subroutine GET_NUMBER	scrn_id	I4	R	SMG display screen id.
	row	I4	R	Display input row.
	col	I4	R	Display input column.
	num	I4	W	Numerical value entered by user.
	numlen	I4	R	Length of input field.
	terminator	I4	W	Terminator encountered in SMG read.
	help_key	C*	R	Help key for current routine.
	default	C*	R	Default input value.
subroutine GET_RECORD	pbd_id	I4	R	SMG pasteboard id
	kbd_id	I4	R	SMG keyboard id.
	main_id	I4	R	SMG display screen id.
	scroll_id	I4	R	SMG display screen id.
	valid	L	W	Valid record retrieved flag.
subroutine GET_REC_FIELD	offset	I4	R	Position of field in value file record.
	buffer	B	W	Byte array to receive data field.
	size	I2	R	Size of field buffer.
	valid	L1	W	Validity byte of field retrieved.
	stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
function GET_RELATIVE_POS	volnam	C*	R	Archival volume name.
function GET_RPT_FIELDS	pbd_id	I4	R	SMG pasteboard id
	kbd_id	I4	R	SMG keyboard id.
	main_id	I4	R	SMG display screen id.
	rpt_id	I4	R	SMG display screen id.
function GET_RTR_FIELDS	pbd_id	I4	R	SMG pasteboard id
	kbd_id	I4	R	SMG keyboard id.
	main_id	I4	R	SMG display screen id.
	rtg_id	I4	R	SMG display screen id.
subroutine GET_SAVESET_EXT	rdkey	C*	R	Key value.
	volnam	C*	R	Archival volume name.
	extension	C*	W	Archival save set extension.
subroutine GET_SET_LABELS	in_set_name	C12	R	Set name string.
	set_lun	I4	R	Set file access LUN.
	dbse_lun	I4	R	Value file access LUN.
	scrn_id	I4	R	SMG display screen id.
	kbd_id	I4	R	SMG keyboard id.

Routine Name	Argument	Type	Access	Description
subroutine GET_SET_VARS	terminator	I2	W	Terminator encountered in SMG read.
	scrn_id	I4	R	SMG display screen id.
	select_line_id	I4	R	SMG display screen id.
	pbrd_id	I4	R	SMG pasteboard id
	kbd_id	I4	R	SMG keyboard id.
	set_var_name	C*	W	Array of X, Y, and Z parameter names.
function GET_SOURCE_TESTID	primary_id	I4	R	SMG display screen id.
subroutine GET_SUBPID	None			
subroutine GET_TRGT_DEV	error_cond	B	W	Completion status - not defined in include file.
subroutine GET_TYPE	type	C	W	Test data type ((M)easured, (E)ngineering, etc.).
	terminator	I2	W	Terminator encountered in SMG read.
subroutine GET_VAR_VALS	terminator	I2	W	Terminator encountered in SMG read.
	scrn_id	I4	R	SMG display screen id.
	select_line_id	I4	R	SMG display screen id.
	pbrd_id	I4	R	SMG pasteboard id
	kbd_id	I4	R	SMG keyboard id.
	out_row	I4	R	Display row for input field.
	val_buff	R4	W	Array of X, Y, and Z values.
	lab_lun	I4	R	Label file access LUN.
	in_label	C*	R	Current label name.
subroutine HELP	pbrd_id	I4	R	SMG pasteboard id
	kbd_id	I4	R	SMG keyboard id.
	help_key	C*	R	Help key for current routine.
	level	I2	R	Level of help key path.
subroutine HLI_PROCESS_TABLE2	table2_id	I4	R	ID of RS/1 table one.
	table1_id	I4	R	ID of RS/1 table two.
	comblock	B	W	RS/1 status array.
subroutine HLI_PROCESS_TABLE3	scrn_id	I4	R	SMG display screen id.
	lab_lun	I4	R	Label file access LUN.
	table3_id	I4	R	ID of RS/1 table three.
	comblock	B	W	RS/1 status array.
	stat3	I2	W	Return status.
function I2_VAL	pointer	I2	R	First element of a two-byte array.
function I4_RUN	buf	B	R	Buffer containing RUN field.
	form	C	R	Data type of RUN field.
	size	I2	R	Number of bytes in RUN field.
function I4_VAL	pointer	I4	R	First element of a four-byte array.
function INCREASING	low	Record	R	Data cell value record.
	mid	Record	R	Data cell value record.
	high	Record	R	Data cell value record.
	form	C	R	Type of data in data cell (R,I,C).
	size	I2	R	Number of bytes in data cell.
subroutine INIT_SESSION	None			
function INITIALIZE_SESSION_FILE_SCAN				
	file_name	C*	R	Session file name.
subroutine INSERT_PARM_DATA	file_name	C*	R	Parameter file name.
	area	C*	R	Area in which data is to be inserted.
	name	C*	R	Variable name to be inserted.
	position	I2	R	Position in area where field to be inserted.
	length	I2	R	Number of bytes of data for field inserted.
	d_form	C	R	Format of data field inserted.
	description	C*	R	Description of field inserted.

Routine Name		Argument	Type	Access	Description
function	KEYCHANGE	key_field	I2	R	Key position.
		rdkey	C*	R	Key value.
		key_len	I2	R	Length of passed key field.
subroutine	LABEL_PARAM	scrn_id	I4	R	SMG display screen id.
subroutine	LEVEL_OF_ACCESS	None			
subroutine	LINE2TESTREC	line	C*	R	Character string containing test index record fields.
subroutine	LIST2ARRAY	array	C*	W	Array of character strings.
		array_cntr	I2	W	Number of character strings in "array".
subroutine	LIST_LABELS	scrn_id	I4	R	SMG display screen id.
		pbrd_id	I4	R	SMG pasteboard id
		kbd_id	I4	R	SMG keyboard id.
subroutine	LIST_SETS	scrn_id	I4	R	SMG display screen id.
		pbrd_id	I4	R	SMG pasteboard id
		kbd_id	I4	R	SMG keyboard id.
subroutine	LOAD_LABEL_LIST	lab_lun	I4	R	Label file access LUN.
		scrn_id	I4	R	SMG display screen id.
subroutine	LOAD_RS1_DATA	scrn_id	I4	R	SMG display screen id.
		kbd_id	I4	R	SMG keyboard id.
		pbrd_id	I4	R	SMG pasteboard id
		EXIT_RTN	External		Address of exit handler.
subroutine	LOAD_RS1_TABLES	scrn_id	I4	R	SMG display screen id.
		kbd_id	I4	R	SMG keyboard id.
		pbrd_id	I4	R	SMG pasteboard id
subroutine	LOAD_SET_VAR_VALS	in_label	C*	R	Current label.
		lab_lun	I4	R	Label file access LUN.
		val_buff	R4	W	Array of X, Y, and Z values.
subroutine	LOG_MOD	ch_log_lun	I4	R	Change log access LUN.
		label	C*	R	Label value of modified data cell.
		field_type	C	R	Data type of modified data cell.
		field_len	I2	R	Number of bytes for modified data cell.
		offset	I4	R	VALUE record offset of modified data cell.
		old_field	B	R	Array of bytes containing previous data cell value.
		new_field	B	R	Array of bytes containing new data cell value.
subroutine	MANAGE_ENGR_FILES	pbrd_id	I4	R	SMG pasteboard id
		scrn_id	I4	R	SMG display screen id.
		kbd_id	I4	R	SMG keyboard id.
subroutine	MGR_EXIT	None			
subroutine	MIN_MAX_SCAN	scrn_id	I4	R	SMG display screen id.
		kbd_id	I4	R	SMG keyboard id.
		pbrd_id	I4	R	SMG pasteboard id
subroutine	MIN_MAX_SCAN2	scrn_id	I4	R	SMG display screen id.
		kbd_id	I4	R	SMG keyboard id.
		pbrd_id	I4	R	SMG pasteboard id
subroutine	MODIFY_LABEL_DESCR	pbrd_id	I4	R	SMG pasteboard id
		scrn_id	I4	R	SMG display screen id.
		kbd_id	I4	R	SMG keyboard id.
		lab_lun	I4	R	Label file access LUN.
		dbse_lun	I4	R	Value file access LUN.
		param_file	C*	R	Parameter file name.
		data_file	C*	R	VALUE file name.

Routine Name	Argument	Type	Access	Description
subroutine MOVE_BYTES	from	B	R	Source byte array.
	to	B	W	Target byte array.
	num_bytes	I2	R	Number of bytes to copy.
subroutine MOVE_MEASURED_DATA	pbrd_id	I4	R	SMG pasteboard id
	scrn_id	I4	R	SMG display screen id.
	kbd_id	I4	R	SMG keyboard id.
subroutine NEW_SPEC	out_spec	C*	W	New file specs.
	in_spec	C*	R	Input file specs.
	def_spec	C*	R	Default file specs.
function NEXT_TOKEN	token	Record	W	Field read from session log.
subroutine NULLIFY_LIST	None			
subroutine OPEN_ARCH_FILE	arch_lun	I4	R	Archival file access LUN.
	stat	I4	W	Completion status returned by OPEN command.
subroutine OPEN_CH_LOG_FILE	ch_log_lun	I4	R	Change log access LUN.
	ch_log_file	C*	R	name of change log file.
	stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
subroutine OPEN_DBSE	dbse_lun	I4	R	Value file access LUN.
	dbse_file	C*	R	Name of file to be opened.
subroutine OPEN_LABEL_FILE	source	C8	R	Source name string.
	test_id	C8	R	Test id name string.
	lab_lun	I4	R	Label file access LUN.
	param_file	C*	R	Name of corresponding parameter file.
subroutine OPEN_SET_FILE	set_lun	I4	R	Set file access LUN.
	set_file	C*	R	Name of set file.
	access	I2	R	Read/Write access flag.
subroutine OPEN_USER_LOG	log_lun	I4	R	User log access LUN.
	user_name	C15	W	User name.
subroutine OPTION_D	pbrd_id	I4	R	SMG pasteboard id
	scrn_id	I4	R	SMG display screen id.
	kbd_id	I4	R	SMG keyboard id.
	scroll_id	I4	R	SMG scrolling region id.
subroutine OUTPUT_REPORT	scrn_id	I4	R	SMG display screen id.
	kbd_id	I4	R	SMG keyboard id.
	pbrd_id	I4	R	SMG pasteboard id
function POP	element	I4	W	Element "popped" from stack.
function POP_FIRST_POSITION	None			
subroutine POSITION_DEVICE	None			
subroutine POSITION_TEST_FILE	test_lun	I4	R	Test index file access LUN.
	read_mode	I2	R	Read access mode (defined in INCLUDES:file_io_def.inc)
	key	C*	R	Test index file READ key.
	key_position	I2	R	Key field position (primary, secondary, etc.)
	source	C8	R	Source name string for record match.
	test_id	C8	R	Test id name string for record match.
	system	C*	R	System field string for record match.
	component	C*	R	Component field string for record match.
	engineer	C*	R	Engineer field string for record match.
	contractor	C*	R	Contractor field string for record match.
subroutine PRINT_REPORT	open_stat	I4	W	Status returned by report file OPEN command.

Routine Name	Argument	Type	Access	Description
subroutine PROCESS_LABEL	dbse_lun	I4	R	Value file access LUN.
	scrn_id	I4	R	SMG display screen id.
	label_pos	I4	R	Position of label in "sl_label_array".
	upper_range	C12	R	Upper range for bound.
	lower_range	C12	R	Lower bound for range.
subroutine PROCESS_SESSION_FILE				
	sess_file	C*	R	Session file name.
	error	L1	W	Error flag.
	scrn_id	I4	R	SMG display screen id.
	root_session_name	C*	W	File name without node/disk info.
subroutine PROCESS_TABLE2	l_label_name	C12	R	Array of label names.
	l_num_units	I2	R	Number of labels in "l_label_name" array.
	load_lun	I4	R	Table file access LUN.
	load_name	C*	R	RS/1 meta file name.
subroutine PROCESS_TABLE3	scrn_id	I4	R	SMG display screen id.
	load_name	C*	R	RS/1 meta file name.
	lab_lun	I4	R	Label file access LUN.
	load_lun	I4	R	Table file access LUN.
	param_file	C*	R	Test parameter file name.
	stat3	I2	W	Return status.
subroutine PURGE_RUNS	scrn_id	I4	R	SMG display screen id.
function PUSH	element	I4	R	Element "pushed" on stack.
function PUSH_FIRST_POSITION	None			
subroutine PUT_ARY_FIELD	offset	I4	R	Field offset position in "rec".
	buffer	B	R	Byte array buffer that will be loaded into "rec".
	size	I2	R	Number of bytes to load into "rec".
	rec	B	R/W	Value file record byte array.
	stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
subroutine PUT_BYTES	out_buf	C12	W	Output buffer containing converted value.
	byte_buf	B	R	Byte array containing data to be converted to string.
	rec_field_form	C	R	Output field data type.
	rec_field_size	I2	R	Number of bytes in "byte_buf" to use.
subroutine PUT_INVALID_FIELD	offset	I4	R	Position in value file record to contain field.
	buffer	B	R	Field to be loaded into value file record.
	size	I2	R	Number of elements in "buffer".
	stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
subroutine PUT_KEY_SCROLL_OPTIONS				
	srce_scrn_id	I4	R	SMG display screen id.
	item	C*	R	Field for which options are displayed.
subroutine PUT_LABEL_FIELD	field	C*	R	Name of field to be modified.
	buffer	B	W	Byte buffer containg field data.
	size	I2	R	Number of bytes in field.
	stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
subroutine PUT_MENU	dsp_id	I4	R	SMG display screen id.
subroutine PUT_OPTION_LIST	srce_scrn_id	I4	R	SMG display screen id.
subroutine PUT_PARAM_DATA	par_lun	I4	R	Parameter file access LUN.
	param_file	C*	R	Parameter file name.
	scrn_id	I4	R	SMG display screen id.
	out_row	I4	W	Display output row.
	bottom_of_region	I4	W	Last row of display output region.

Routine Name	Argument	Type	Access	Description
subroutine PUT_REC_FIELD	offset	I4	R	Position in value file record to contain field.
	buffer	B	R	Field to be loaded into value file record.
	size	I2	R	Number of elements in "buffer".
	stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
subroutine PUT_SUBOPTION_LIST	srce_scrn_id	I4	R	SMG display screen id.
	item	C*	R	Current item to be selected - Displayed in prompt.
function QUERY_BOUND	scrn_id	I4	R	SMG display screen id.
	row	I4	R	Display row for prompt.
	col	I4	R	Display column for prompt.
	string	C*	W	Returned string.
	kbd_id	I4	R	SMG keyboard id.
	pbrd_id	I4	R	SMG pasteboard id
	rows	I4	R	Number of rows in display to change.
	cols	I4	R	Number of columns in display to change.
	QUIT_RTN	External		Address of exit handler.
function R4_VAL	pointer	R4	R	First element of a four-byte array.
function R8_VAL	pointer	R8	R	First element of an eight-byte array.
subroutine RANGE_CHECK	dbse_lun	I4	R	Value file access LUN.
	key_label	C12	R	Label used to retrieve value file record.
	cell	Record	R/W	Used to define a value file data cell.
	cell_format	C	R	Format of value file data cell.
	cell_size	I4	R	Number of bytes in value file data cell.
subroutine READ_ARCH_REC	arch_lun	I4	R	Archival file access LUN.
	function	I2	R	I/O access function (defined in INCLUDES:file_io_def.inc)
	rdkey	C*	R	Key value.
	key_len	I2	R	Length of passed key field.
	key_mode	I2	R	Key type (Defined in INCLUDES:file_io_def.inc)
	stat	I4	W	Completion status returned by READ command.
subroutine READ_CH_LOG_REC	ch_log_lun	I4	R	Change log access LUN.
	function	I2	R	I/O access function (defined in INCLUDES:file_io_def.inc)
	rdkey	C*	R	Key value.
	key_len	I2	R	Length of passed key field.
	stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
subroutine READ_DBSE_REC	rec_label	C*	R	Key value for keyed read.
	dbse_lun	I4	R	Value file access LUN.
	mode	I2	R	Read access mode (defined in INCLUDES:file_io_def.inc)
	stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
subroutine READ_DBSE_REC_HDR	rec_label	C*	R	Key value for keyed read.
	dbse_lun	I4	R	Value file access LUN.
	mode	I2	R	Read access mode (defined in INCLUDES:file_io_def.inc)
	stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
subroutine READ_FIELDS	get_id	I4	R	SMG display screen id.
	prompt_col	I4	R	Display column at which prompt is displayed.
subroutine READ_FIRST_DBSE_REC	dbse_lun	I4	R	Value file access LUN.
	stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
subroutine READ_LABEL	label	C*	R	Key value for keyed read.
	lab_lun	I4	R	Label file access LUN.
	mode	I2	R	Read access mode (defined in INCLUDES:file_io_def.inc)
	stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)

Routine Name		Argument	Type	Access	Description
function	READ_SESSION_FILE	filnam	C*	R	Session file name.
		dbse_lun	I4	R	Value file access LUN.
		scrn_id	I4	R	SMG display screen id.
		err_row	I4	R	Error message row.
		err_col	I4	R	Error message column.
subroutine	READ_SET_REC	set_label	C*	R	Key value for keyed read.
		set_lun	I4	R	Set file access LUN.
		mode	I2	R	Read access mode (defined in INCLUDES:file_io_def.inc)
		stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
subroutine	READ_SET_REC_HDR	set_label	C*	R	Key value for keyed read.
		set_lun	I4	R	Set file access LUN.
		mode	I2	R	Read access mode (defined in INCLUDES:file_io_def.inc)
		stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
subroutine	READ_USER_LOG	log_lun	I4	R	User log access LUN.
		log_stat	L1	W	Completion status (TRUE/FALSE)
		user_name	C15	W	User name.
		source	C8	W	Source name string.
		test_id	C8	W	Test id name string.
		type	C	W	Data type.
subroutine	REDRAW_SCREEN	None			
subroutine	REFRESH_MIN_MAX_SCREEN				
		scrn_id	I4	R	SMG display screen id.
		start_pos	I2	R	First "list" element to be displayed.
		stop_pos	I2	R	Last "list" element to be displayed.
		top_of_region	I2	R	Top of display output region.
		screen_column	I2	R	Output column.
		list	C*	R	Character string array.
		dbse_lun	I4	R	Value file access LUN.
subroutine	REMOVE_DATA	scrn_id	I4	R	SMG display screen id.
		ch_log_id	I4	R	SMG display screen id.
subroutine	REMOVE_INVALID_OFFSETS				
		label_pos	I4	R	Label position in "sl_label_array".
		dbse_lun	I4	R	Value file access LUN.
		label_id	C12	W	Label name at "label_pos".
function	REMOVE_LAST_QUEUE_ENTRY				
		None			
function	RENAME_LABEL	old_name	C*	R	Label name currently in file.
		new_name	C*	R	Label name to replace "old_name" entries.
function	REPLACE_QUEUE_ENTRY	q_pos	I4	R	Queue position to be replaced.
		rab	Record	R	Record Access Block.
subroutine	REP_EXIT	None			
function	RESET_POSITION	rab	Record	W	Record Access Block.
subroutine	RESTORE	pbd_id	I4	R	SMG pasteboard id
		main_id	I4	R	SMG display screen id.
		arch_id	I4	R	SMG display screen id.
		kbd_id	I4	R	SMG keyboard id.
		monitor	L1	R	Display archival commands to screen.

Routine Name	Argument	Type	Access	Description
subroutine RESUME_SESSION	scrn_id	I4	R	SMG display screen id.
	kbd_id	I4	R	SMG keyboard id.
	pbrd_id	I4	R	SMG pasteboard id
	previous_screen	L1	W	Return to previous screen flag.
	root_session_name	C*	W	Session file name.
subroutine RES_PARM_FILE	file_name	C*	R	Parameter file name.
subroutine SAVE_FIELD	active_record	I2	R	Current label parameter field.
	stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
	scrn_id	I4	R	SMG display screen id.
	message	L1	W	Message display flag.
function SAVE_POSITION	rab	Record	R	Record Access Block.
subroutine SCAN_LIST	token	C*	R	Item to be found in list.
	scan_stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc).
	list_item	C*	W	Item matching "token".
	list_len	I2	R	Length of list to be searched.
	list_pos	I2	R/W	List position of matching element.
	list	C*	R	Array of character strings to be searched.
subroutine SCROLL_KEYS	key_position	I2	R	Record key position.
	key	C*	R	Record key value.
	terminator	I2	W	Terminator encountered in SMG read.
subroutine SCROLL_SOURCES	read_mode	I2	R	Read access mode (defined in INCLUDES:file_io_def.inc)
	key	C*	R	Record key value.
	key_position	I2	R	Record key position.
	r_source	C8	R	Source name string.
	r_testid	C8	R	Test id name string.
	terminator	I2	W	Terminator encountered in SMG read.
subroutine SCROLL_TESTS	lun	I4	R	Scroll file access LUN.
	key_mode	I2	R	Key type (Defined in INCLUDES:file_io_def.inc)
	access_key	C*	R	Key to be matched by scrolled records.
	key_len	I2	R	Length of passed key field.
	dsp_id	I4	R	SMG display screen id.
	scrn_id	I4	R	SMG display screen id.
	kbd_id	I4	R	SMG keyboard id.
subroutine SCROLL_VALIDATION	dbse_lun	I4	R	Value file access LUN.
	scroll_id	I4	R	SMG display screen id.
	pbrd_id	I4	R	SMG pasteboard id
	kbd_id	I4	R	SMG keyboard id.
subroutine SELECT_MATCH	pattern	C*	R	Input pattern to be matched.
	scan_stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc).
	list_size	I2	R	Number of elements in "list".
	position	I2	W	Position in "list" that match is found.
	list	C*	R	Array of character strings to search.
	scrn_id	I4	R	SMG display screen id.
	kbd_id	I4	R	SMG keyboard id.
	pbrd_id	I4	R	SMG pasteboard id
subroutine SESS_FILE_READ_ERR	token	Record	R	Field processed from input.
	str	C*	R	Label name string.
	err_type	I4	R	Error type.
	scrn_id	I4	R	SMG display screen id.
	err_row	I4	R	Error message display row.
	err_col	I4	R	Error message display column.

Routine Name	Argument	Type	Access	Description
subroutine SET_DBSE_REC	rec_label	C*	R	Key value for value file read.
	dbse_lun	I4	R	Value file access LUN.
	mode	I2	R	Read access mode (defined in INCLUDES:file_io_def.inc)
	stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
subroutine SET_FILE	rab	Record	R	Record Access Block.
	posit	I4	R	File position.
	stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
subroutine SET_FILE_POINTER	lun	I4	R	VALUE file LUN.
	p_key_0	C8	R	Key 0 match value.
	p_key_1	C12	R	Key 1 match value.
	p_key_2	C12	R	Key 2 match value.
	p_key_3	C12	R	Key 3 match value.
	key_mode	I2	R	Key position.
subroutine SET_MIN_MAX	bytes	B	W	Byte array to receive min/max value.
	form	C1	R	Field data type (R,C,I).
	length	I2	R	Field length (bytes).
	minmax	I2	R	Minimum/maximum value flag.
function SET_MODE	pbd_id	I4	R	SMG pasteboard id
function SET_PROT_USEROPEN	fab	Record	W	File Attributes Block.
	rab	Record	R	Record Access Block.
	lun	I4	R	LUN of file to be opened.
subroutine SET_VAL	dbse_lun	I4	R	Value file access LUN.
	rel_pos	I2	W	Relative position after data loaded.
	max_rel_pos	I2	R	Maximum relative position before data loaded.
subroutine SET_VALIDITY	dbse_lun	I4	R	Value file access LUN.
	relative_pos	I2	R	Offset in VALUE file record.
subroutine SET_VOLPOS	None			
subroutine SHIFT_ARRAY	num_positions	I2	R/W	Number of offsets in offset array.
	pos_list	I2	R/W	Offset array.
subroutine SKIP_WHITE_SPACE	should_skip_comments	L1	R	Skip comments flag.
subroutine SORTER	lun	I4	R	Test index LUN.
	status	I4	W	System completion status.
subroutine SORT_LABEL_ARRAY	None			
subroutine STAT_MSG	stat	I4	R	SMG status to be displayed.
subroutine STOP_PROC	pbd_id	I4	R	SMG pasteboard id
function STRLEN	string	C*	R	Input string.
subroutine SWAP	byte_buf_1	B	R	Source buffer byte array.
	byte_buf_2	B	W	Target buffer byte array.
	len	I2	R	Number of bytes to copy.
subroutine TOGGLE_NONULLS	label_pos	I4	R	Label position in "sl_label_array".
	valid_only	L1	R/W	Valid data collection flag.
subroutine TESTREC2LINE	line	C*	W	Record output buffer.
function TEST_ENGR_ACCESS	in_source	C8	R	Source name string.
	in_test_id	C8	R	Test id name string.
subroutine UNLOCK_DBSE_REC	dbse_lun	I4	R	Value file access LUN.
	stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
subroutine UNLOCK_LABEL_RECORD	lab_lun	I4	R	Label file access LUN.
	stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)

Routine Name	Argument	Type	Access	Description
subroutine UP	num_rows	I4	R	Number of rows to scroll up.
	rab_addr	I4	R	Address of RAB record.
	lun	I4	R	Scroll file access LUN.
	scrn_id	I4	R	SMG display screen id.
subroutine UPCASE	instring	C*	R/W	String to be converted.
subroutine UPDATE_DATA_POINTERS				
	scrn_id	I4	R	SMG display screen id.
	pbrd_id	I4	R	SMG pasteboard id
	kbd_id	I4	R	SMG keyboard id.
function UPDATE_FILE_POSITION				
	rab	Record	W	Record Access Block.
subroutine UPDATE_LABEL_POINTERS				
	scrn_id	I4	R	SMG display screen id.
	kbd_id	I4	R	SMG keyboard id.
	p_label_id	C12	R	Label name.
	dbse_lun	I4	R	Value file access LUN.
	dbse_file	C*	R	Value file name.
	message	L1	W	Message present flag.
	valid_only	L1	R/W	Valid data collection flag.
subroutine UPDATE_SCREEN	scrn_id	I4	R	SMG display screen id.
	num_lab	I2	R	Number of labels loaded.
	max_relptr	I2	R	Largest relative pointer used.
subroutine UPDATE_SELECT_LINE	source	C8	R	Source name string.
	test_id	C8	R	Test id name string.
	type	C	R	Data type accessed (E,M,...).
subroutine UPDATE_SET_POINTERS	scrn_id	I4	R	SMG display screen id.
	kbd_id	I4	R	SMG keyboard id.
	set_id	C12	R	Set name string.
	set_lun	I4	R	Set file access LUN.
	set_file	C*	R	Set file name.
	dbse_lun	I4	R	Value file access LUN.
	message	L1	W	Message currently displayed flag.
subroutine UPDATE_TEST_FILE	arc_flag	B	R	Archival flag for "arch_rec".

Routine Name	Argument	Type	Access	Description
subroutine UPDATE_WINDOW	old_top	I4	R	Previous record index displayed at screen top.
	old_bot	I4	R	Previous record index displayed at screen bottom.
	old_left	I4	R	Previous cell position displayed on screen left.
	old_right	I4	R	Previous cell position displayed on screen right.
	new_top	I4	R	Current record index displayed at screen top.
	new_bot	I4	R	Current record index displayed at screen top.
	new_left	I4	R	Current cell position displayed on screen left.
	new_right	I4	R	Current cell position displayed on screen right.
	scrn_buf	B	R	Byte array containing display records.
	cs_lab	C12	R	Array of field names.
	field_form	C	R	Array of display cell data types.
	field_len	I4	R	Array of display cell field lengths.
	rec_len	I4	R	Array of display record lengths.
	rd_stat	I2	R	Array of data cell internal READ stats.
	first	I4	W	Position of first queue entry.
	last	I4	W	Position of last queue entry.
	label_list	C*	R	Array of key fields for screen record READs.
	scrn_id	I4	R	SMG display screen id.
	ch_log_lun	I4	R	Change log access LUN.
	ch_log_present	L1	R	Change log present flag.
	update_stat	I4	?	Left over from previous version.
	view_only	L1	R	View/Edit data flag.
subroutine USER_EXIT	None			
function VALID	terminator	I2	W	Terminator encountered in SMG read.
function VALID_OPTION	option	C	R	Option selected.
	option_list	C*	R	String of valid option characters.
	terminator	I2	R	Terminator encountered in SMG read.
	valid_terminator	I2	R	Valid terminator value.
subroutine VERIFY_VOLUME	error_cond	B	W	Completion status - not defined in include file.
	new_rec	Record	R	Archival record.
subroutine VIEW_CHANGE_LOG	pbrd_id	I4	R	SMG pasteboard id
	ch_log_id	I4	R	SMG display screen id.
	kbd_id	I4	R	SMG keyboard id.
	selected	L1	W	Field selected flag.
	return_buff	B	W	Byte buffer containing selected field.
subroutine VIEW_LABELS	scrn_id	I4	R	SMG display screen id.
	pbrd_id	I4	R	SMG pasteboard id
	kbd_id	I4	R	SMG keyboard id.
	dbse_lun	I4	R	Value file access LUN.
	dbse_file	C44	R	Value file name.
	valid_only	L1	R	Valid data collection flag.
subroutine VIEW_REPORT	pbd_id	I4	R	SMG pasteboard id
	kbd_id	I4	R	SMG keyboard id.
	main_id	I4	R	SMG display screen id.
	scroll_id	I4	R	SMG display screen id.

Routine Name	Argument	Type	Access	Description
subroutine VIEW_SETS	scrn_id	I4	R	SMG display screen id.
	pbrd_id	I4	R	SMG pasteboard id
	kbd_id	I4	R	SMG keyboard id.
	set_lun	I4	R	Set file access LUN.
	dbse_lun	I4	R	Value file access LUN.
	set_file	C44	R	Set file name.
subroutine VIRTUAL_INPUT	echo_def	B	R	Echo default string flag. "echo_def = 0" no echo.
	string	C*	W	Returned string.
	length	I2	R	Maximum input string length.
	dsp_id	I4	R	SMG display screen id.
	kbd_id	I4	R	SMG keyboard id.
	row	I4	R	Display row for data echo.
	col	I4	R	Display column for data echo.
	terminator	I2	W	Terminator encountered in SMG read.
	fill	C	R	String filler character.
	case	I2	R	Upper case conversion flag. (INCLUDES:file_io_def.inc)
subroutine WEED_OUT_POS	relative_pos	I2	W	Offset position array.
	rel_pos_num	I2	W	Number of entries in "relative_pos".
	dbse_lun	I4	R	Value file access LUN.
function WILDCARD	wild_pattern	C*	R	Original selection mask.
	test_pattern	C*	R	Item to be compared to "wild_pattern".
subroutine WRITE_ARCH_REC	arch_lun	I4	R	Archival file access LUN.
	stat	I4	W	Completion status returned by WRITE command.
subroutine WRITE_BUFFER	out_buff	C*	W	Output buffer - string version of field selected.
	ch_field_type	C	R	Data type of data cell.
	ch_field_len	I2	R	Field length of data cell.
	cnv_form	C*	R	Output format for Internal WRITE.
	val_buff	Record	E	Buffer containing selected field.
subroutine WRITE_CH_LOG_REC	ch_log_lun	I4	R	Change log access LUN.
	function	I2	R	I/O access function (defined in INCLUDES:file_io_def.inc)
	stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
subroutine WRITE_DBSE_REC	dbse_lun	I4	R	Value file access LUN.
	stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
	mode	I2	R	Write access mode (defined in INCLUDES:file_io_def.inc)
subroutine WRITE_INVTEX_MSG	text	C*	R	Invalid text string.
subroutine WRITE_INVTEXREC_MSG	text	C*	R	Ivalid text string.
	rec	I4	R	Invalid record number.
	file	C*	R	Name of file being processed.
subroutine WRITE_LABEL	lab_lun	I4	R	Label file access LUN.
	stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
	mode	I2	R	Write access mode (defined in INCLUDES:file_io_def.inc)
subroutine WRITE_SESSION_FILE	filnam	C*	R	Session file name.
	labels	C*	R	Array of label names.
	ranges	Record	R	Array of range record.
	num_labels	I2	R	Number of entries in "labels".
subroutine WRITE_SET_REC	set_lun	I4	R	Set file access LUN.
	stat	I2	W	Completion status (defined in INCLUDES:file_io_def.inc)
	mode	I2	R	Write access mode (defined in INCLUDES:file_io_def.inc)

Routine Name	Argument	Type	Access	Description
subroutine WRITE_USER_LOG	log_lun	I4	R	User log access LUN.
	user_name	C15	R	User name.
	source	C8	R	Source name string.
	test_id	C8	R	Test id name string.
	type	C	R	Data type accessed (E,M,...).

Appendix C
DATABASE SYSTEM FILE RECORD FORMATS

<code_section>.HLP

These are help files for the corresponding code sections. The files contain eighty column ASCII text records created by editor. Help entries are defined by creating a help "tag" followed by the associated help text. A tag is created by inserting a "#" followed by a unique help path string. Each path entry in the path string must be a single character and path entries must be separated by a period. An example of a valid help tag is "#1.A.G". Any text lines following a help tag are associated with that tag. The help text for a help tag is terminated when another help tag or "#end" is encountered.

```
      FILENAME : [DBSE]<code_section>.HLP
          TYPE : Sequential with variable record length.
RECORD LENGTH : Maximum 79 bytes.
```

MASTER ARCHIVAL FILE

FILENAME : [DBSE]ARCH_MASTER.FIL
TYPE : Indexed with fixed record length.
PRIMARY KEY : volnam
SECONDARY KEYS : src_testid
RECORD LENGTH : 200 bytes

FIELD NAME	TYPE	DESCRIPTION
volnam	char*12	Archival volume name.
src_testid	char*17	Archived source/test_id.
saveset	char*17	Archival save set name.
media	char*20	Archival media type.
date	char*9	Archival date.
text	char*60	Descriptive text from test index file.
comment	char*60	Archival comment field.
relpos	integer*4	Position of save set on archival volume.
filler	byte	Used to fill record.

FDL FILE FOR ARCH_MASTER.FIL

```
TITLE   "File Name : DBDISK:[DBSE]ARCH_MASTER.FDL"

IDENT   " 9-AUG-1990 14:21:54   VAX-11 FDL Editor"

SYSTEM
        SOURCE                  "VAX/VMS"

FILE
        NAME                    "dbdisk:[dbse]arch_master.fil"
        ORGANIZATION            indexed

RECORD
        CARRIAGE_CONTROL        none
        FORMAT                  fixed
        SIZE                    200

AREA 0
        ALLOCATION              15
        BEST_TRY_CONTIGUOUS     yes
        BUCKET_SIZE             3
        EXTENSION               3

AREA 1
        ALLOCATION              6
        BEST_TRY_CONTIGUOUS     yes
        BUCKET_SIZE             3
        EXTENSION               3

AREA 2
        ALLOCATION              12
        BEST_TRY_CONTIGUOUS     yes
        BUCKET_SIZE             3
        EXTENSION               6

KEY 0
        CHANGES                 no
        DATA_AREA               0
        DATA_FILL               100
        DATA_KEY_COMPRESSION    no
        DATA_RECORD_COMPRESSION no
        DUPLICATES              yes
        INDEX_AREA              1
        INDEX_COMPRESSION       no
        INDEX_FILL              100
        LEVEL1_INDEX_AREA       1
        NAME                    "Volume name"
        PROLOG                  3
        SEG0_LENGTH             12
        SEG0_POSITION           0
        TYPE                    string
```

FDL FILE FOR ARCH_MASTER.FIL (Concluded)

```
KEY 1
        CHANGES                 yes
        DATA_AREA               2
        DATA_FILL               100
        DATA_KEY_COMPRESSION    no
        DUPLICATES              yes
        INDEX_AREA              2
        INDEX_COMPRESSION       no
        INDEX_FILL              100
        LEVEL1_INDEX_AREA       2
        NAME                    "Source/Testid"
        SEG0_LENGTH             17
        SEG0_POSITION           12
        TYPE                    string
```

INCLUDE FILE FOR ARCH_MASTER.FIL

```
c
c  ARCH_REC_DEF.INC
c
c  This include file describes the archival file records.
c
      parameter       volume_key      =       0       !primary key pos.
      parameter       srctst_key      =       1       !secondary key pos.

      structure /arch/

         character    volnam          *12,    !Volume name     - Key 0
     1                src_tstid       *17,    !Source//Test ID - Key 1
     1                saveset         *17,    !Save set name
     1                media           *20,    !Media type
     1                date            *9,     !Archival date
     1                text            *60,    !Descriptive text
     1                comment         *60     !comment field

         integer*4    relpos          !Relative position on volume

         byte         filler                  !record filler

      end structure
      record /arch/ arch_rec

      character       source          *8,     !source
     1                testid          *8,     !test ID
     1                device          *10,    !device name
     1                dbs_dev         *5,     !test storage device
     1                req_volnam      *12,    !requested volume
     1                req_saveset     *17,    !requested save set
     1                req_date        *9      !requested date

      integer*2       marks_per_file          !device file markers

      integer*4       num_entries             !entries on device

      common /archival/ arch_rec,
     1                  source,
     1                  testid,
     1                  device,
     1                  dbs_dev,
     1                  req_volnam,
     1                  req_saveset,
     1                  req_date,
     1                  marks_per_file,
     1                  num_entries
```

TEST INDEX FILE

FILENAME : [DBSE]TEST.IND
TYPE : Indexed with fixed record length.
PRIMARY KEY : source
SECONDARY KEYS : test_id, system, component, test_engr, test_cntr
RECORD LENGTH : 196

FIELD NAME	TYPE	DESCRIPTION
source	char*8	Source of test data.
test_id	char*8	Test ID of test data.
system	char*12	System associated with test data.
component	char*12	Component associated with test data.
test_engr	char*16	Engineer associated with test data.
test_cntr	char*16	Contractor associated with test data.
num_runs	integer*4	Number of runs loaded for the test.
test_desc	char*60	Description of test.
dbs_device	char*5	Test storage device.
test_start	char*8	Test start date.
test_comp_date	char*8	Test completion date.
arc_m_flag	byte	Measured data archival flag.
arc_m_vol	char*6	Last measured archival volume name.
arc_e_flag	byte	Engineering data archival flag.
arc_e_vol	char*6	Last engineering archival volume name.
test_right	char*10	Access restriction identifier.
test_prot	char	Test protection flag. "P" indicates protected data.
num_dbs_labels	integer*4	Number of labels loaded for test.
dbs_scale	real*4	Scale factor.
dbs_exts	char*6	List of available type extensions.

FDL FILE FOR TEST.IND

```
TITLE   "File Name : DBDISK:[DBSE]TEST.FDL"

IDENT   "12-JAN-1990 09:39:27   VAX-11 FDL Editor"

SYSTEM
        SOURCE                  "VAX/VMS"

FILE
        ALLOCATION              28
        BEST_TRY_CONTIGUOUS     no
        BUCKET_SIZE             2
        CLUSTER_SIZE            2
        CONTIGUOUS              no
        EXTENSION               0
        GLOBAL_BUFFER_COUNT     0
        NAME                    "[DBSE]TEST.IND"
        ORGANIZATION            indexed
        OWNER                   [1,1]
        PROTECTION              (system:RWED, owner:RWED, group:RWE, world:RWE)

RECORD
        BLOCK_SPAN              yes
        CARRIAGE_CONTROL        none
        FORMAT                  fixed
        SIZE                    196

AREA 0
        ALLOCATION              28
        BUCKET_SIZE             2
        EXTENSION               0

KEY 0
        CHANGES                 no
        DATA_AREA               0
        DATA_FILL               100
        DATA_KEY_COMPRESSION    yes
        DATA_RECORD_COMPRESSION yes
        DUPLICATES              yes
        INDEX_AREA              0
        INDEX_COMPRESSION       yes
        INDEX_FILL              100
        LEVEL1_INDEX_AREA       0
        NAME                    ""
        NULL_KEY                no
        PROLOG                  3
        SEG0_LENGTH             8
        SEG0_POSITION           0
        TYPE                    string
```

FDL FILE FOR TEST.IND (Continued)

```
KEY 1
        CHANGES                 yes
        DATA_AREA               0
        DATA_FILL               100
        DATA_KEY_COMPRESSION    yes
        DUPLICATES              yes
        INDEX_AREA              0
        INDEX_COMPRESSION       yes
        INDEX_FILL              100
        LEVEL1_INDEX_AREA       0
        NAME                    ""
        NULL_KEY                no
        SEG0_LENGTH             8
        SEG0_POSITION           8
        TYPE                    string

KEY 2
        CHANGES                 yes
        DATA_AREA               0
        DATA_FILL               100
        DATA_KEY_COMPRESSION    yes
        DUPLICATES              yes
        INDEX_AREA              0
        INDEX_COMPRESSION       yes
        INDEX_FILL              100
        LEVEL1_INDEX_AREA       0
        NAME                    ""
        NULL_KEY                no
        SEG0_LENGTH             12
        SEG0_POSITION           16
        TYPE                    string

KEY 3
        CHANGES                 yes
        DATA_AREA               0
        DATA_FILL               100
        DATA_KEY_COMPRESSION    yes
        DUPLICATES              yes
        INDEX_AREA              0
        INDEX_COMPRESSION       yes
        INDEX_FILL              100
        LEVEL1_INDEX_AREA       0
        NAME                    ""
        NULL_KEY                no
        SEG0_LENGTH             12
        SEG0_POSITION           28
        TYPE                    string
```

FDL FILE FOR TEST.IND (Concluded)

```
KEY 4
        CHANGES                 yes
        DATA_AREA               0
        DATA_FILL               100
        DATA_KEY_COMPRESSION    yes
        DUPLICATES              yes
        INDEX_AREA              0
        INDEX_COMPRESSION       yes
        INDEX_FILL              100
        LEVEL1_INDEX_AREA       0
        NAME                    ""
        NULL_KEY                no
        SEG0_LENGTH             16
        SEG0_POSITION           40
        TYPE                    string

KEY 5
        CHANGES                 yes
        DATA_AREA               0
        DATA_FILL               100
        DATA_KEY_COMPRESSION    yes
        DUPLICATES              yes
        INDEX_AREA              0
        INDEX_COMPRESSION       yes
        INDEX_FILL              100
        LEVEL1_INDEX_AREA       0
        NAME                    ""
        NULL_KEY                no
        SEG0_LENGTH             16
        SEG0_POSITION           56
        TYPE                    string
```

INCLUDE FILE FOR TEST.IND

```
c  This file defines the TEST.IND data record contents.
c
c  Display sizes for test_rec fields
c
      parameter       source_len       = 8      !source
      parameter       test_id_len      = 8      !test id
      parameter       system_len       = 12     !system
      parameter       component_len    = 12     !component
      parameter       test_engr_len    = 16     !test engineer
      parameter       test_cntr_len    = 16     !contractor
      parameter       num_runs_len     = 8      !number of runs entered
      parameter       test_desc_len    = 60     !comment field
      parameter       dbs_device_len   = 5      !storage device
      parameter       test_start_len   = 8      !starting date
      parameter       test_comp_date_len = 8    !completion date
      parameter       arc_m_flag_len   = 1      !# of meas. data arch. sets
      parameter       arc_m_vol_len    = 6      !last meas. arch. set name
      parameter       arc_e_flag_len   = 1      !# of eng. data arch. sets
      parameter       arc_e_vol_len    = 6      !last eng. arch. set name
      parameter       test_right_len   = 10     !rights protection field
      parameter       test_prot_len    = 1      !proprietary protection field
      parameter       num_dbs_labels_len = 8    !# of labels loaded
      parameter       dbs_scale_len    = 8      !scale factor
      parameter       dbs_exts_len     = 6      !available file exts. (M,E,...)

      structure /test_ind_record/

            character       source           *(source_len)
            character       test_id          *(test_id_len)
            character       system           *(system_len)
            character       component        *(component_len)
            character       test_engr        *(test_engr_len)
            character       test_cntr        *(test_cntr_len)
            integer*4       num_runs         !# runs in the test
            character       test_desc        *(test_desc_len)
            character       dbs_device       *(dbs_device_len)
            character       test_start       *(test_start_len)
            character       test_comp_date   *(test_comp_date_len)
            byte            arc_m_flag       !measured archival flag
            character       arc_m_vol        *(arc_m_vol_len)
            byte            arc_e_flag       !eng. archival flag
            character       arc_e_vol        *(arc_e_vol_len)
            character       test_right       *(test_right_len)
            character       test_prot        *(test_prot_len)
            integer*4       num_dbs_labels   !# labels loaded
            real*4          dbs_scale        !scale factor
            character       dbs_exts         *(dbs_exts_len)

      end structure

      record /test_ind_record/ test_rec
      record /test_ind_record/ access_rec

      common /test_ind/ test_rec, access_rec
```

USER LOG FILE

This is the test access log. A record is entered for each user of the database system. The record stores the last test accessed by each user. Each record is 32 bytes long.

FILENAME : [DBSE]USER_LOG.LOG

TYPE : Indexed with fixed record length.
PRIMARY KEY : username
RECORD LENGTH : 32

FIELD NAME	TYPE	DESCRIPTION
user_name	char*15	User accessing database system.
source	char*8	Source last accessed by user.
test_id	char*8	Test last accessed by user.
type	char*1	Extension type last accessed by user.

FDL FILE FOR USER_LOG.LOG

```
TITLE   "File Name : DBDISK:[DBSE]USER.LOG"

IDENT   " 9-AUG-1990 14:21:54   VAX-11 FDL Editor"

SYSTEM
        SOURCE                  VAX/VMS

FILE
        ALLOCATION              10
        BEST_TRY_CONTIGUOUS     yes
        BUCKET_SIZE             2
        CLUSTER_SIZE            5
        CONTIGUOUS              no
        EXTENSION               5
        GLOBAL_BUFFER_COUNT     0
        NAME                    "USER_LOG.LOG"
        ORGANIZATION            indexed
        OWNER                   [200,1]
        PROTECTION              (system:RWED, owner:RWED, group:RW, world:RW)

RECORD
        BLOCK_SPAN              yes
        CARRIAGE_CONTROL        none
        FORMAT                  fixed
        SIZE                    32

AREA 0
        ALLOCATION              10
        BUCKET_SIZE             2
        EXTENSION               5

KEY 0
        CHANGES                 no
        DATA_KEY_COMPRESSION    yes
        DATA_RECORD_COMPRESSION yes
        DATA_AREA               0
        DATA_FILL               100
        DUPLICATES              no
        INDEX_AREA              0
        INDEX_COMPRESSION       yes
        INDEX_FILL              100
        LEVEL1_INDEX_AREA       0
        NAME                    ""
        NULL_KEY                no
        PROLOG                  3
        SEG0_LENGTH             15
        SEG0_POSITION           0
        TYPE                    string
```

TEST CHANGE LOG FILE

This file records modifications made to the VALUE file data cells.

Nomenclature : "source" is the facility or other data source.
"test" is the test identifier.

FILENAME : [DBSE.source.test]test.CHL*
TYPE : Indexed with variable record length.
PRIMARY KEY : label//offset
RECORD LENGTH : Maximum of 20544 bytes

FIELD NAME	TYPE	DESCRIPTION
mod_label	char*12	Label name of cell modified.
ch_offset	char*8	Offset value of cell modified - character form.
mod_offset	Integer*4	Offset value of cell modified - integer form.
ch_field_type	char	Data type of cell modified.
ch_field_len	Integer*2	Number of bytes in data cell.
mod_original	byte(12)	Byte array containing original cell value.
num_mods	Integer*2	Number of modifications made to data cell.
mod_user	char*12	User name of process that modified data cell.
mod_value	byte(12)	Array of modification values.
mod_date	char*9	Date that cell was modified.
mod_time	char*8	Time that cell was modified.

FDL FILE FOR TEST CHANGE LOG

```
TITLE   "File Name : DBDISK:[DBSE]CHANGE_LOG.FDL"

IDENT   "21-SEP-1990 11:04:56   VAX-11 FDL Editor"

SYSTEM
        SOURCE                  "VAX/VMS"

FILE
        ORGANIZATION            indexed

RECORD
        CARRIAGE_CONTROL        none
        FORMAT                  variable
        SIZE                    20544

AREA 0
        ALLOCATION              51
        BEST_TRY_CONTIGUOUS     yes
        BUCKET_SIZE             48
        EXTENSION               48

AREA 1
        ALLOCATION              48
        BEST_TRY_CONTIGUOUS     yes
        BUCKET_SIZE             48
        EXTENSION               48

KEY 0
        CHANGES                 no
        DATA_AREA               0
        DATA_FILL               100
        DATA_KEY_COMPRESSION    no
        DATA_RECORD_COMPRESSION no
        DUPLICATES              no
        INDEX_AREA              1
        INDEX_COMPRESSION       no
        INDEX_FILL              100
        LEVEL1_INDEX_AREA       1
        NAME                    "Label/Offset"
        PROLOG                  3
        SEG0_LENGTH             20
        SEG0_POSITION           0
        TYPE                    string
```

INCLUDE FILE FOR TEST CHANGE LOG

```
c
c                              CH_LOG_REC_DEF.INC
c
c  This file defines the records in the change log files.
c  A change log file is created for a test whenever a data cell in the
c  value file is modified.
c
      parameter       max_mods          =       1000     !maximum mods/cell

      character       mod_label                  *12,     !label of cell
      1               ch_offset                  *8,      !cell offset (char)
      1               ch_field_type,                      !cell data type
      1               mod_user (max_mods)        *12,     !user of modification
      1               mod_date (max_mods)        *9,      !date cell modified
      1               mod_time (max_mods)        *8       !time cell modified

      integer*4       mod_offset                          !cell offset

      integer*2       num_mods,                           !number of cell mods.
      1               ch_field_len                        !cell byte size

      byte            mod_original     (12),              !original value
      1               mod_value        (12,max_mods)      !array of mod. values

      common /ch_log/ mod_label, ch_offset, mod_offset, mod_original,
      1               num_mods, ch_field_type, ch_field_len,
      1               mod_user, mod_value, mod_date, mod_time
```

LABEL FILE RECORD PARAMETER FILE

FILENAME : [DBSE.source.test]test.PRM*
TYPE : Sequential with fixed record length.
RECORD LENGTH : 62 bytes

FIELD NAME	TYPE	DESCRIPTION
name	char*12	The name for the current field in .IND file.
position	Integer*2	The byte position of the field in .IND file.
length	Integer*2	The field length of the field - number of bytes.
d_form	char	The data type of the field (R, I, C).
description	char*40	Brief description of the field.

INCLUDE FILE FOR LABEL FILE RECORD PARAMETER FILE

```
c
c File name : includes:parameter_def.inc
c
      parameter   max_fields = 500                        !max fields per label
      parameter   max_record = 2000                       !max label rec. size
      parameter   lablen = 12                             !length of label name
c
      character       file_nm                   *44,      !.IND file name
      1               key_name                  *12,      !key field name
      1               field_name (max_fields)   *12,      !parameter field names
      1               field_form (max_fields),            !parameter data types
      1               key_form                            !key data type
c
      byte            label_rec       (max_record)        !label file record
c
      integer*2       record_len,                         !label file rec. len.
      1               key_size,                           !size of key field.
      1               num_fields,                         !# of parameter fields
      1               field_pos       (max_fields),       !parameter field
                                                          !positions.
      1               field_size      (max_fields)        !parameter field
                                                          !lengths - bytes.
c
      common /parameters/ file_nm, key_name, field_name, record_len,
      1                   key_size, num_fields, field_pos, field_size,
      1                   field_form, label_rec
```

```
        FILENAME : [DBSE.source.test]test.SET*
            TYPE : Indexed with variable record length.
     PRIMARY KEY : set_name
   RECORD LENGTH : Variable
```

FIELD NAME	TYPE	DESCRIPTION
set_name	char*12	Unique set name.
set_ext_num	byte	Extended record number.
set_ext	byte	Extended record flag.
num_units	integer*2	Number of labels for this set.
set_var_name	char*12	Array of X,Y,Z parameter names.
label_name	char*12	Array of labels to be included in set.
label_val	real*4	Array of X,Y,Z parameter values.
.	.	
.	.	
.	.	

Pairs of label_name/label_val are repeated for "num_units" labels.

FDL FILE FOR SET FILE

```
TITLE   "File Name : DBDISK:[DBSE]SET.FDL"

IDENT   "21-SEP-1990 12:24:56   VAX-11 FDL Editor"

SYSTEM
        SOURCE                  VAX/VMS

FILE
        ALLOCATION              60
        BEST_TRY_CONTIGUOUS     no
        BUCKET_SIZE             28
        CLUSTER_SIZE            5
        CONTIGUOUS              no
        EXTENSION               0
        FILE_MONITORING         no
        GLOBAL_BUFFER_COUNT     0
        NAME                    ""
        ORGANIZATION            indexed
        OWNER                   []
        PROTECTION              (system:RWED,owner:RWED,group:RWED,world:RWED)

RECORD
        BLOCK_SPAN              yes
        CARRIAGE_CONTROL        none
        FORMAT                  variable
        SIZE                    24000

AREA 0
        ALLOCATION              58
        BUCKET_SIZE             28
        EXTENSION               0

KEY 0
        CHANGES                 no
        DATA_KEY_COMPRESSION    yes
        DATA_RECORD_COMPRESSION yes
        DATA_AREA               0
        DATA_FILL               100
        DUPLICATES              no
        INDEX_AREA              0
        INDEX_COMPRESSION       yes
        INDEX_FILL              100
        LEVEL1_INDEX_AREA       0
        NAME                    ""
        NULL_KEY                no
        PROLOG                  3
        SEG0_LENGTH             12
        SEG0_POSITION           0
        TYPE                    string
```

INCLUDE FILE FOR SET FILE

```
c
c File name : INCLUDES:set_rec_def.inc
c
      parameter       max_set_labels  = 2000
c
c  There are 24 bytes per label:
c
c  label_name  : 12 bytes
c  value_array : 3-4 byte REALs
c
      parameter       max_set_store   = 1327          !max sets per rec.

      parameter       set_header_size = 52            !set record header size

      character       set_name                      *12,!name of set
      1               label_name (max_set_labels)   *12,!name of label
      1               set_var_name (3)              *12,!X,Y, Z parameter
                                                        !fields
      1               hold_set_var_name (3)         *12 !temp buffer

      byte            set_ext,                          !extended record flag
      1               set_ext_num                       !extended record num.

      integer*2       num_units,                        !num labels in set
      1               hold_num_units                    !temp buffer

      integer*4       set_rec_len                       !set record length

      real*4          label_val (3,max_set_labels)      !X,Y, Z parameter
                                                        !values.

      common /set_par/ set_name, num_units, set_var_name, label_name,
      1                label_val
```

VALUE FILE

FILENAME : [DBSE.source.test]test.VLU*
TYPE : Indexed with variable record length.
PRIMARY KEY : dbse_key//ext_number
RECORD LENGTH : Variable

FIELD NAME	TYPE	DESCRIPTION
dbse_key	C12	Label of data record.
ext_number	B	Extended record number.
extension	B	Extended record flag.
case_label	C12	Version of label to be displayed.
rec_field_form	C1	Data type of data cells for this record.
rec_field_size	I2	Number of bytes per data cell for this record.
cells_loaded	I4	Number of data cells in this record.
min_buf	B	12-byte array containing minimum value loaded.
max_buf	B	12-byte array containing maximum value loaded.
dbs_rec_len	I4	Number of bytes in cell data array.
dbse_rec	B	Array of bytes containing cell data.

FDL FILE FOR VALUE FILE

```
TITLE   "File Name : DBDISK:[DBSE]VLU.FDL"

IDENT   "18-JUN-1990 10:26:38   VAX-11 FDL Editor"

SYSTEM
        SOURCE                  "VAX/VMS"

FILE
        ORGANIZATION            indexed

RECORD
        CARRIAGE_CONTROL        none
        FORMAT                  variable
        SIZE                    31912

AREA 0
        ALLOCATION              5000
        BEST_TRY_CONTIGUOUS     yes
        BUCKET_SIZE             63
        EXTENSION               1000

AREA 1
        ALLOCATION              63
        BEST_TRY_CONTIGUOUS     yes
        BUCKET_SIZE             63
        EXTENSION               63

KEY 0
        CHANGES                 no
        DATA_AREA               0
        DATA_FILL               100
        DATA_KEY_COMPRESSION    no
        DATA_RECORD_COMPRESSION no
        DUPLICATES              no
        INDEX_AREA              1
        INDEX_COMPRESSION       no
        INDEX_FILL              100
        LEVEL1_INDEX_AREA       1
        PROLOG                  3
        SEG0_LENGTH             13
        SEG0_POSITION           0
        TYPE                    string
```

INCLUDE FILE FOR VALUE FILE

```
c
c File name : INCLUDES:dbse_rec_def.inc
c
      parameter       max_frames = 9800                !MUST HAVE SAME
                                                       !VALUE AS sl_num_rrefs
                                                       !frames allowed

      parameter       max_dbse_record =
      1               max_frames * 13                  !maximum record length

      parameter       max_store_size  = 31850          !max bytes per rec.

      parameter       header_size     = 61             ! the dbse header size

      parameter       max_labels      = 2000
      parameter       max_sets        = 1000

      character       rec_field_form,                  !data storage format
      1               dbse_key        *12,             !data access key
      1               case_label      *12,
      1               c_min           *12,             !char min
      1               c_max           *12              !char max

      byte            dbse_rec  (max_dbse_record),     !database record
      1               extension,
      1               ext_number,
      1               min_buf   (12),                  !min value buffer
      1               max_buf   (12)                   !max value buffer

      integer*2       rec_field_size,                  !size of field
      1               i2_min,                          !2 byte int min
      1               i2_max                           !2 byte int max

      integer*4       dbse_rec_len,
      1               cells_loaded,
      1               i4_min,                          !4 byte int min
      1               i4_max                           !4 byte int max

      real*4          r4_min,                          !4 byte real min
      1               r4_max                           !4 byte real max

      real*8          r8_min,                          !8 byte real min
      1               r8_max                           !8 byte real max

      logical*1       dbse_data_acquired

      equivalence (  c_min, min_buf )
      equivalence ( i2_min, min_buf )
      equivalence ( i4_min, min_buf )
      equivalence ( r4_min, min_buf )
      equivalence ( r8_min, min_buf )
      equivalence (  c_max, max_buf )
      equivalence ( i2_max, max_buf )
      equivalence ( i4_max, max_buf )
      equivalence ( r4_max, max_buf )
      equivalence ( r8_max, max_buf )


      common /dbse_par/ dbse_key, case_label,
      1       dbse_rec_len, rec_field_size, rec_field_form,
      1       extension, ext_number, cells_loaded, min_buf, max_buf,
      1       dbse_rec, dbse_data_acquired
```

MAXIMUM RELATIVE POSITION (MRP) FILE

FILENAME : [DBSE.source.test]test.MRP*
TYPE : Sequential - single record.
RECORD LENGTH : 4 bytes

FIELD NAME	TYPE	DESCRIPTION
MRP	I4	Maximum relative position. Defines the current number of values in the VALUE file (test_id.VLU*).

TEST REPORTS

The test reports are named in the report file name extension. For example, if a pre-test report was generated for TEST 0016 in the AFD facility it would be named 0016.TPRE-TEST. The report name follows ".T" in the report file extension. The reports can only contain ASCII text and the records must not exceed 80 characters each.

```
      FILENAME : [DBSE.source.test]test.T<report_name>*
          TYPE : Sequential with variable record length.
RECORD LENGTH : Variable
```

LABEL FILE

```
       FILENAME : [DBSE.source.test]test.IND*
           TYPE : Indexed with fixed record length.
    PRIMARY KEY : engr_label
  RECORD LENGTH : Defined in the corresponding .PRM* parameter file.
```

FIELD NAME	TYPE	DESCRIPTION
engr_label	C12	Uppercase label name used as key.
case_label	C12	Version of label name to be displayed.
datatype	C1	Data storage format (R,C,I).
fldlen	I2	Number of bytes required for data field.
data_segment		Used to tag data retrieval path.

These are the required fields. Other data and fields are added to the record as specified in the corresponding .PRM* parameter file.

TEST GEOMETRY FILE

FILENAME : [DBSE.source.test]test.GEO
TYPE : Indexed with fixed record length.
KEY : LCS
RECORD LENGTH : 102 Bytes

FIELD NAME	TYPE	DESCRIPTION
lcs	R4	Local coordinate system number.
rcs	R4	Reference coordinate system number.
scale	R4	Local scale = lcsdimension/rcsdimension.
type(3)	C1	Type of local system : type(1) = L for left-hand Cartesian. = X, Y, Z for cylindrical axis = other defaults to right-hand Cartesian. type(2) and type(3) are for cylindrical systems. type(2) = X, Y, or Z indicates the axis of origin for the cylindrical angle. type(3) = R for right-hand cylindrical angle. L for left-hand cylindrical angle.
origin(3)	R4	If rcs is Cartesian, these are DX, DY DZ. If rcs is cylindrical, these are axial distance, angle, and radius.
rorder(3)	C1	Order of axis rotations used to align the rcs to the lcs. YZX would indicate rotations about Y, then Z, and finally X.
rot(3)	C1	Rotation angles in the order specified by rorder. Signs of rotations are based on the right-hand sense.
geocom	C60	Descriptive comment for the lcs

SESSION LOG FILE

FILENAME : <user-specified>.XSL
TYPE : Sequential with variable record length.
RECORD LENGTH : Variable - maximum 80 bytes.

For a complete description of the session log file see "APPENDIX D".

RS/1 VALUE TABLE META FILE

FILENAME : <user-specified>.RS11
TYPE : Sequential with variable record length.
RECORD LENGTH : Variable

FIELD NAME	TYPE	DESCRIPTION
RECORD 1:		
rows	C6	Number of rows represented by the number of records in the file (= number of label names).
RECORD 2:		
columns	C6	Number of columns represented by the fields in the records (= number of data values plus 1 for the label).

Records 3 and 4 are output only if sets were chosen.
The purpose of records 3 and 4 is to provide a desired table spacing. The record numbering in descriptions of files .RS12 and .RS13 assume that these two records are used.

RECORD 3:		
label	C12	The entry "SET" provides a column in the resulting table for set names.
"~"	C1	A tilde is used as a data field separator.
data	C12	Data value. Should look "EMPTY" to RS/1.
"~"	C1	Data field separator.
data	C12	Data = "EMPTY".
.	.	.
.	.	Repeat empty data to length of other
.	.	data records (record 5 and subsequent).

RS/1 VALUE TABLE META FILE (Concluded)

RECORD 4:		
label	C12	The entry "PARAMETER" provides a column in the table for set plotting parameter names.
"␣"	C1	Data field separator.
data	C12	Data value. Should look "EMPTY" to RS/1.
"␣"	C1	Data field separator.
data	C12	Data value = "EMPTY".
.	.	.
.	.	Repeat empty data to length of other
.	.	data records (record 5 and subsequent).
RECORD 5:		
label	C12	Name of first label selected.
"␣"	C1	Data field separator.
data	C12	Data value.
"␣"	C1	Data field separator.
data	C12	Data value.
.	.	.
.	.	Repeat data values to the length
.	.	of the selected data.

RECORD 6 and subsequent records : Repeat record 5 for all selected labels.

RS/1 SET TABLE META FILE

FILENAME : <user-specified>.RS12
TYPE : Sequential with variable record length.
RECORD LENGTH : Variable

FIELD NAME	TYPE	DESCRIPTION
RECORD 1:		
rows	C6	Number of rows represented by the number of records in the file (= number of set names).
RECORD 2:		
columns	C6	Number of columns represented by the number of fields in the records (= number of label names in table .RS11 plus 2 for the "SET" and "PARAMETER" columns).
RECORD 3 and subsequent records.		
label	C12	Set name.
"~"	C1	Data field separator.
parameter	C12	Set plotting parameter name, e.g. THETA.
"~"	C1	Data field separator.
data	C12	Plotting parameter data value corresponding to the label in record 3 of file .RS11, or an "EMPTY" indication if the label was not entered as part of a set.
"~"	C1	Data field separator.
data	C12	Plotting parameter data value corresponding to the label in record 3 of file .RS11, or an "EMPTY" indication if the label was not entered as part of a set.
"~"	C1	Data field separator.
.	.	.
.	.	Repeat data values to equal the number of
.	.	labels entered in table .RS11.

RS/1 LABEL TABLE META FILE

FILENAME : <user-specified>.RS13
TYPE : Sequential with variable record length.
RECORD LENGTH : Variable

FIELD NAME	TYPE	DESCRIPTION
RECORD 1:		
rows	C6	Number of rows represented by the number of records in the file (= number of label names plus 1 for the header line from record 3).
RECORD 2:		
columns	C6	Number of columns represented by the number of fields in the records (= number of fields in the TEST_ID.IND file).
RECORD 3:		
field1	C12	First field name in the TEST_ID.PRM file.
","	C1	Data field separator.
field3	C12	Third field name in the TEST_ID.PRM file.
.	.	.
.	.	Repeat order of fields described in the
.	.	TEST_ID.PRM file to serve as headers for
.	.	the data to be provided in subsequent records.
.	.	

RS/1 LABEL TABLE META FILE (Concluded)

FIELD NAME	TYPE	DESCRIPTION
RECORD 4:		
data	C12	The value of the label name from record 5 of the .RS11 file.
"~"	C1	Data field separator.
data	C?	The value of the second field in the TEST_ID.IND file for the label in the field.

NOTE : The field length "?" indicates that the field length will vary for each test and field.

"~"	C1	Data field separator.
data	C?	The value of the third field in the TEST_ID.IND file for the label in the field.
"~"	C1	Data field separator.
data	C?	The value of the third field in the TEST_ID.IND file for the label in the field.
.	.	.
.	.	Repeat field values to the end of the
.	.	TEST_ID.IND record.

Record 5 and subsequent records : repeat record 4 for the labels from record 6 and subsequent in the .RS11 file.