K 6

```
EEEEEEEEEEEEEEE  RRRRRRRRRRRR     FFFFFFFFFFFFFFF
EEEEEEEEEEEEEEE  RRRRRRRRRRRR     FFFFFFFFFFFFFFF
EEEEEEEEEEEEEEE  RRRRRRRRRRRR     FFFFFFFFFFFFFFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEEEEEEEEEEE     RRRRRRRRRRRR     FFFFFFFFFFFF
EEEEEEEEEEEE     RRRRRRRRRRRR     FFFFFFFFFFFF
EEEEEEEEEEEE     RRRRRRRRRRRR     FFFFFFFFFFFF
EEE              RRR   RRR        FFF
EEE              RRR   RRR        FFF
EEE              RRR   RRR        FFF
EEE              RRR      RRR     FFF
EEE              RRR      RRR     FFF
EEE              RRR      RRR     FFF
EEEEEEEEEEEEEEE  RRR         RRR  FFF
EEEEEEEEEEEEEEE  RRR         RRR  FFF
EEEEEEEEEEEEEEE  RRR         RRR  FFF
```

D 2

```
**FILE**ID**ERF



          EEEEEEEEEE  RRRRRRRR    FFFFFFFFFF
          EEEEEEEEEE  RRRRRRRR    FFFFFFFFFF
          EE          RR      RR  FF
          EE          RR      RR  FF
          EE          RR      RR  FF
          EE          RR      RR  FF
          EEEEEEEE    RRRRRRRR    FFFFFFFF
          EEEEEEEE    RRRRRRRR    FFFFFFFF
          EE          RR  RR      FF
          EE          RR  RR      FF
          EE          RR    RR    FF                ....
          EE          RR    RR    FF                ....
          EEEEEEEEEE  RR      RR  FF                ....
          EEEEEEEEEE  RR      RR  FF                ....

          LL            IIIIII      SSSSSSSS
          LL            IIIIII      SSSSSSSS
          LL              II      SS
          LL              II      SS
          LL              II      SS
          LL              II      SS
          LL              II        SSSSSS
          LL              II        SSSSSS
          LL              II              SS
          LL              II              SS
          LL              II              SS
          LL              II              SS
          LLLLLLLLLL    IIIIII    SSSSSSSS
          LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
ER
VO
```

```
0 MODULE ERF (%TITLE 'Errorlog Report Formatter'
0               MAIN = ERF,
0               IDENT = 'V04-000' ) =
0 ! The version number above must be changed in two places.
0 ! Search for 'IDENT ='.
1 BEGIN
1
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1
1 !++
1 ! FACILITY:  ERF, Errorlog Report Formatter
1 !
1 ! ABSTRACT:
1 !       This is the main routine for ERF. All exit paths must return
1 !       here to exit.
1 !
1 !
1 ! ENVIRONMENT:
1 !
1 !       VAX/VMS operating system. unprivileged user mode,
1 !
1 ! AUTHOR:  Elliott A. Drayton
1 !
1 ! Modified by:
1 !
1 !       V03-026 EAD0197         Elliott A. Drayton              23-Jul-1984
1 !               Added code to check version numbers in VALIDATE_PACKET.
1 !
1 !       V03-025 EAD0193         Elliott A. Drayton               6-Jul-1984
1 !               Made LSTLUN own and initialized it from syecom. Cleared
1 !               MAILBOX_CHANNEL which is now in syecom.
1 !
1 !       V03-024 EAD0180         Elliott A. Drayton              26-Jun-1984
1 !               Add workstation device class and have /LOG report any
1 !               files that were created.
```

```
1 !
1 !      V03-023 EAD0170          Elliott A. Drayton                4-May-1984
1 !              Added context parameter to FILE_SCAN.
1 !
1 !      V03-022 EAD0140          Elliott A. Drayton               12-Apr-1984
1 !              Removed reference to EMBETDEF.
1 !
1 !      V03-021 EAD0130          Elliott A. Drayton                9-Apr-1984
1 !              Moved image_loader and its support routines in to ERFSHR.
1 !
1 !      V03-020 EAD0119          Elliott A. Drayton               22-Mar-1984
1 !              Add support for the UNKNOWN keyword.
1 !
1 !      V03-019 EAD0117          Elliott A. Drayton               21-Mar-1984
1 !              Fixed translation of loadable image name.
1 !
1 !      V03-018 SAR0209          Sharon A. Reynolds               10-Mar-1984
1 !              - Fixed a bug in write_msg so it checks the contents of
1 !              output_flag.
1 !              - Changed the call to write_msg so that the address of the
1 !              command line descriptor is passed.
1 !
1 !      V03-017 EAD0116          Elliott A. Drayton                9-Mar-1984
1 !              Removed emb_buf and syecom_buf.
1 !
1 !              JMG0013          Joel M. Gringorten                8-Mar-1984
1 !              Output command line at end of report if IO is directed
1 !              to a file.
1 !
1 !      V03-016 EAD0107          Elliott A. Drayton               29-Feb-1984
1 !              Fixed problem in validate_packet which prevented unknown
1 !              error packets from being identified.
1 !
1 !      V03-015 EAD0104          Elliott A. Drayton               29-Feb-1984
1 !              Major clean up.
1 !
1 !              SAR0189          Sharon A. Reynolds               13-Feb-1984
1 !              - Added the ERF_NOTFOUND message test at EOF.
1 !              - Added summary specific test for summary update calls.
1 !              - Added two parameters to the Brief_c_dispatcher calls.
1 !
1 !              JMG0012          Joel M. Gringorten                7-Feb-1984
1 !              Added support for /statistics qualifier.
1 !
1 !      V03-014 JMG0005          Joel M. Gringorten               29-Dec-1983
1 !              Added support for /summary=histogram.
1 !              - Added histo output dispatch to erf_control.
1 !              - Added histo update dispatch to process_file.
1 !
1 !      V03-013 SAR0172          Sharon A. Reynolds               16-Nov-1983
1 !              - Added 'logmscp' entry support.
1 !              - Updated the 'max_xxx_type' values for new devices.
1 !              - Completed the device tables.
1 !              - Fixed a bug with /include=mem/summary=mem, the summary
1 !                statistics were counted twice.
1 !
1 !      V03-012 SAR0165          Sharon A. Reynolds               14-Oct-1983
```

```
1 !                     - Removed the code that counts logmessage/logstatus
1 !                       entries to fix a bug. (/incl=logm/excl=DU).
1 !                     - Fixed a bug with the report type and /SID qualifier.
1 !                     - Changed erf_norep to erf_invreptyp with fatal severity.
1 !
1 !             V03-011 SAR0150         Sharon A. Reynolds              7-Oct-1983
1 !                     Fixed a bug in the 'validate_packet' routine.
1 !
1 !             V03-010 SAR0138         Sharon A. Reynolds              20-Sep-1983
1 !                     Fixed bug in code that determines whether to init commons.
1 !
1 !             V03-009 SAR0136         Sharon A. Reynolds              12-Sep-1983
1 !                     Added code that removed the mscp info msg first part.
1 !
1 !             V03-008 SAR0128         Sharon A. Reynolds              7-Sep-1983
1 !                     Added routine to initialize the qiocommon, opcode,
1 !                     and modes commons. (Init_commons routine). Replaced
1 !                     debugging error messages with permanent error messages.
1 !
1 !             V03-007 EAD0006         Elliott A. Drayton              23-Aug-1983
1 !                     Added routines to open and parse text library records which
1 !                     are used to build internal tables.
1 !
1 !             V03-006 SAR0062         Sharon A. Reynolds,             20-Jun-1983
1 !                     Fixed bug with processor type in 'validate_packet'.
1 !
1 !             V03-005 SAR0031         Sharon A. Reynolds,             2-Jun-1983
1 !                     Put in a permanent solution to syecom buffer address problem.
1 !                     Removed some unneccessary code.
1 !
1 !             V03-004 SAR0023         Sharon A. Reynolds,             11-May-1983
1 !                     Modified 'process_packet' and 'full_dispatcher' so that the
1 !                     summary information will be updated. Also modified
1 !                     'process_record' so that multiple part MSCP entries will
1 !                     be output. Fixed a problem passing record_length.
1 !                     Put in a temporary solution to syecom problem.
1 !
1 !             V03-003 SAR0011         Sharon A. Reynolds,             11-Apr-1983
1 !                     Added code to 'validate_packet' to set up the device
1 !                     class and type fields to the appropriate emb fields,
1 !                     as they are in different locations per entry type.
1 !
1 !             V03-002 SAR0010         Sharon A. Reynolds,             9-Apr-1983
1 !                     Intialized status, status2, status3, and status4. Also
1 !                     added code to ensure processing a 'device entry' before
1 !                     examining device class and device type in 'validate packet'.
1 !
1 !--
```

```
;  164      0163  1 REQUIRE 'SRC$:RECSELDEF.REQ';    ! Defines emb fields
;  165      0294  1 REQUIRE 'LIB$:PARSERDAT.R32';    ! Defines option_flag fields
;  166      0448  1 REQUIRE 'SRC$:ERFDEF.REQ';
;  167      0734  1
;  168      0735  1
;  169      0736  1 FORWARD ROUTINE
;  170      0737  1     Build_class_tables,           ! Allocates and inits device class tables
;  171      0738  1     Erf,                          ! Top level routine
;  172      0739  1     Erf_control,                  ! Main control loop
;  173      0740  1     Full_dispatcher,              ! Cases to correct EXEC_IMAGE call
;  174      0741  1     Get_library_text,             ! Reads library module record and calls parser
;  175      0742  1     Handler,                      ! Condition handler
;  176      0743  1     Init_commons,                 ! Initialize fortran data commons
;  177      0744  1     Open_text_lib,                ! Routine to open and init the text library
;  178      0745  1     Parse_text_record,            ! Routine to compress and parse text record
;  179      0746  1     Parse_max_table_size,         !
;  180      0747  1     Parse_module_names,           !
;  181      0748  1     Parse_device_desc_record,     !
;  182      0749  1     Parse_max_min_table_record,   !
;  183      0750  1     Process_file,                 ! Reads ERRLOG.SYS & loops till EOF
;  184      0751  1     Process_packet,               ! Cases on report type to dispatcher
;  185      0752  1     Validate_packet,              ! Checks packet for CPU,ENTRY,CLASS&TYPE
;  186      0753  1     Write_binary,                 ! Write packet as read, no text translation
;  187      0754  1     Write_err_msg;                ! Write error messages to output file
;  188      0755  1
;  189      0756  1 EXTERNAL ROUTINE
;  190      0757  1     Exec_image,                   ! Call loaded image with correct params.
;  191      0758  1     Device_type_entry,            !
;  192      0759  1     Get_vm,                       ! Allocates requested buffers
;  193      0760  1     Image_loader,                 ! Determines which image to load & loads
;  194      0761  1     Lbr$close,
;  195      0762  1     Lbr$get_record,
;  196      0763  1     Lbr$ini_control,
;  197      0764  1     Lbr$lookup_key,
;  198      0765  1     Lbr$open,
;  199      0766  1     Lbr$set_locate,
;  200      0767  1     Lib$cvt_dtb,                  ! Convert decimal to binary
;  201      0768  1     Lib$extzv,
;  202      0769  1     Lib$file_scan,
;  203      0770  1     Log_filename,                 ! Signals filenames and error messages
;  204      0771  1     Map_image,
;  205      0772  1     Parse_command,                ! Analyze command line
;  206      0773  1     Parse_output_files,           ! Handles the opening of output files
;  207      0774  1     Record_selected,              ! Determines if record should be processed
;  208      0775  1     Timrb,                        ! Runtime statistics package
;  209      0776  1     Timre,                        !
;  210      0777  1     Unknown_dispatcher,           ! Formats and outputs reports for unknown error packets
;  211      0778  1     Write_msg;                    !
;  212      0779  1
;  213      0780  1 EXTERNAL
;  214      0781  1     Class_dir:                    REF VECTOR[,WORD],
;  215      0782  1     EMB:                              $BBLOCK PSECT (EMB),
;  216      0783  1     Input_fab:                        $BBLOCK [],
;  217      0784  1     Input_rab:                        $BBLOCK [],
;  218      0785  1     Input_nam:                        $BBLOCK [],
;  219      0786  1     Input_xabfhc:                     $BBLOCK [],
;  220      0787  1     Lstlun_rab_address:           REF $BBLOCK [],
    Option_flag:                    REF $BBLOCK [],
    Output_fab:                         $BBLOCK [],
    Output_nam:                     REF $BBLOCK [],
    Output_rab:                         $BBLOCK [],
    Parser_data:                    REF $BBLOCK [],
    Parser_table:                   REF $BBLOCK [],
    Related_nam:                        $BBLOCK [],
    Rejected_fab:                       $BBLOCK [],
    Rejected_nam:                       $BBLOCK [],
    Rejected_rab:                       $BBLOCK [],
    Summary_flag:                   REF $BBLOCK [],
    Syecom:                             $BBLOCK PSECT (SYECOM),
    Sys$output_rab_address:         REF $BBLOCK,
    Worst_error:                        $BBLOCK [LONG],
    Lnm$file_dev_desc,
    Bus_image:                      REF BLOCKVECTOR[,2] FIELD (desc_fields),
    Bus_version:                    REF VECTOR[,WORD],
    Bus_xfer_addr:                  REF VECTOR[,LONG],      ! Address of bus xfer address table
    Disk_image:                     REF BLOCKVECTOR[,2] FIELD (desc_fields),
    Disk_version:                   REF VECTOR[,WORD],      ! Address of version number of device dependent code
    Disk_xfer_addr:                 REF VECTOR[,LONG],      ! Address of disk xfer address table
    Lp_image:                       REF BLOCKVECTOR[,2] FIELD (desc_fields),
    Lp_version:                     REF VECTOR[,WORD],
    Lp_xfer_addr:                   REF VECTOR[,LONG],      ! Address of lp xfer address table
    Max_misc_type:                      BYTE,
    Max_lp_type:                        BYTE,
    Packet_processor_xfer_addr:     REF VECTOR[,LONG],      ! Address of realtime xfer address table
    Packet_processor_image:         REF BLOCKVECTOR[,2] FIELD (desc_fields),
    Packet_processor_version:       REF VECTOR[,WORD],
    Realtime_image:                 REF BLOCKVECTOR[,2] FIELD (desc_fields),
    Realtime_version:               REF VECTOR[,WORD],
    Realtime_xfer_addr:             REF VECTOR[,LONG],      ! Address of realtime xfer address table
    Scom_image:                     REF BLOCKVECTOR[,2] FIELD (desc_fields),
    Scom_version:                   REF VECTOR[,WORD],
    Scom_xfer_addr:                 REF VECTOR[,LONG],      ! Address of scom xfer address table
    Tape_image:                     REF BLOCKVECTOR[,2] FIELD (desc_fields),
    Tape_version:                   REF VECTOR[,WORD],
    Tape_xfer_addr:                 REF VECTOR[,LONG],      ! Address of tape xfer address table
    Translate_entry_table:          REF VECTOR [,WORD],
    Workstation_image:              REF BLOCKVECTOR[,2] FIELD (desc_fields),
    Workstation_version:            REF VECTOR[,WORD],
    Workstation_xfer_addr:          REF VECTOR[,LONG];


LITERAL
    Erf$_facility = 8,              ! Facility code for ERF
    Word_size = 2,                  ! Number of bytes in a word
    Longword = 4,                               ! Number of bytes in a longword
    Descriptor_length = 8;          ! Number of bytes in a string descriptor

Own
    Lstlun:                             LONG;

EXTERNAL LITERAL
Erf_badevtyp,
Erf_badevval,
Erf_badmodnam,
1 Erf_clstblerr,
1 Erf_cvterr,
1 Erf_herald,
1 Erf_input,
1 Erf_loaderr,
1 Erf_notfound,
1 Erf_invreptyp,
1 Erf_toomancls,
1 Erf_total,
1 Erf_unkentry,
1 Erf_unkclass;
1
1
1 SD('ERFLIB');    ! Defines string descriptors
```

```
1 GLOBAL
1     Bus_devices:                   REF VECTOR[,WORD],
1     Desc_table_address:            REF BLOCKVECTOR[,2] FIELD (desc_fields),
1     Dev_addrs_ptr:                 REF VECTOR[,LONG],         ! Address device name pointers
1     Dev_class_ptr:                 REF VECTOR[,WORD],
1     Disk_devices:                  REF VECTOR[,WORD],         ! Address of disk device name table (e.g. DB)
1     Function,                                                 ! Function to be done
1     Herald:                            $BBLOCK [12],          ! Address of message vector
1     Inited_commons,
1     Input_desc:                        $BBLOCK [dsc$k_d_bln],! Allocate dynmaic descriptor
1     Input_file_count,
1     Item_count,                                               ! Used as index count
1     Library_func:                      INITIAL (Lbr$c_read),
1     Library_index,
1     Library_type:                      INITIAL (lbr$c_typ_txt),
1     Lp_devices:                    REF VECTOR[,WORD],
1     Max_bus_type:                      BYTE,
1     Max_classes:                       BYTE,
1     Max_cpu_types:                     WORD,
1     Max_disk_type:                     BYTE,
1     Max_range_table_addr:          REF VECTOR [,LONG],
1     Max_realtime_type:                 BYTE,
1     Max_scom_type:                     BYTE,
1     Max_tape_type:                     BYTE,
1     Max_Workstation_type:              BYTE,
1     Min_modules_desc:              REF BLOCKVECTOR[,2] FIELD (desc_fields),
1     Max_modules_desc:              REF BLOCKVECTOR[,2] FIELD (desc_fields),
1     Min_max_table_sizes:           REF VECTOR[,WORD],
1     Min_range_table_addr:          REF VECTOR[,LONG],
1     Module_name_desc,
1     Packet_processor_devices:      REF VECTOR[,WORD],
1     Processor_type_table:          REF VECTOR [,WORD],
1     Realtime_devices:              REF VECTOR[,WORD],
1     Record_desc:                       $BBLOCK [dsc$k_d_bln],
1     Scom_devices:                  REF VECTOR[,WORD],
1     Selected,                                                 ! Count of records selected for processing
1     Summary_dispatcher_addr,                                  ! Transfer address of summary dispatcher
1     Tape_devices:                  REF VECTOR[,WORD],
1     Table_address:                 REF VECTOR[,WORD],
1     Table_length:                          BYTE,
1     Text_rfa:                      VECTOR [2],
1     Token_desc:                    $BBLOCK [DSC$K_D_BLN],
1     Total_selected,
1     Total_rejected,
1     Unknown_entry:                 INITIAL (false),
1     Workstation_devices:           REF VECTOR[,WORD];
1
1 Builtin
1           FFS ;
1
```

```
1 Routine ERF =                                         ! Main routine for ERF
2 BEGIN
2 !++
2 ! Functional description
2 !
2 !       This is the top level routine for the ERF facility.
2 !       It calls the main control loop.  Any errors encountered
2 !       will be passed back to this routine.
2 !
2 ! Calling sequence
2 !
2 !       ERF () from the command language interpreter
2 !
2 ! Input parameters
2 !
2 !       AP = Address of argument list passed from CLI
2 !
2 ! Output parameters
2 !
2 !       None
2 !
2 ! Routine value
2 !
2 !       Worst error is returned.
2 !
2 !----
2
2 LOCAL
2       channel,
2       status;
2
2 !
2 !SET UP HANDLERS ---
2 ! Declare condition handler to record severest
2 ! error message issued, to be returned on exit of image.
2 !
2
2 ENABLE handler;
2
2
2 !
2 !CALL MAIN CONTROL ---
2 ! Invoke the main subroutine.  If any errors are encountered they
2 ! will be returned immediately, if fatal, or saved in WORST_ERROR
2 ! for exit processing.
2 !
2
2 Worst_error = Erf_control() ;
2
2
2 !
2 !RETURN TO USER ---
2 !       Return to the user.  Variable WORST_ERROR is maintained by
2 !       the error handler (see routine HANDLER).  If no messages have
2 !       been signaled then the initial value of WORST_ERROR, SS$_NORMAL,
2 !       will be returned.
2 !
```

```
2
2 Return .worst_error;                                   ! Return contents of WORST_ERROR
2
1 END;


                                                     .TITLE  ERF Errorlog Report Formatter
                                                     .IDENT  \V04-000\

                                                     .PSECT  $PLIT,NOWRT,NOEXE,  PIC,2

                42  49  4C  46  52  45  00000 P.AAB:  .ASCII  \ERFLIB\                               ;
                                        00006         .BLKB   2
                             00000006   00008 P.AAA:  .LONG   6                                      ;
                             00000000'  0000C         .ADDRESS P.AAB                                 ;

                                                     .PSECT  $OWN$,NOEXE,  PIC,2

                                        00000 LSTLUN: .BLKB   4

                                                     .PSECT  $GLOBAL$,NOEXE,  PIC,2

                                        00000 BUS_DEVICES::
                                                      .BLKB   4
                                        00004 DESC_TABLE_ADDRESS::
                                                      .BLKB   4
                                        00008 DEV_ADDRS_PTR::
                                                      .BLKB   4
                                        0000C DEV_CLASS_PTR::
                                                      .BLKB   4
                                        00010 DISK_DEVICES::
                                                      .BLKB   4
                                        00014 FUNCTION::
                                                      .BLKB   4
                                        00018 HERALD::.BLKB   12
                                        00024 INITED_COMMONS::
                                                      .BLKB   4
                                        00028 INPUT_DESC::
                                                      .BLKB   8
                                        00030 INPUT_FILE_COUNT::
                                                      .BLKB   4
                                        00034 ITEM_COUNT::
                                                      .BLKB   4
                             00000001   00038 LIBRARY_FUNC::
                                                      .LONG   1                                      ;
                                        0003C LIBRARY_INDEX::
                                                      .BLKB   4
                             00000004   00040 LIBRARY_TYPE::
                                                      .LONG   4                                      ;
                                        00044 LP_DEVICES::
                                                      .BLKB   4
                                        00048 MAX_BUS_TYPE::
                                                      .BLKB   1
                                        00049 MAX_CLASSES::
                                                      .BLKB   1
                                        0004A MAX_CPU_TYPES::
                                                      .BLKB   2
```

```
                  0004C MAX_DISK_TYPE::
                                  .BLKB   1
                  0004D           .BLKB   3
                  00050 MAX_RANGE_TABLE_ADDR::
                                  .BLKB   4
                  00054 MAX_REALTIME_TYPE::
                                  .BLKB   1
                  00055 MAX_SCOM_TYPE::
                                  .BLKB   1
                  00056 MAX_TAPE_TYPE::
                                  .BLKB   1
                  00057 MAX_WORKSTATION_TYPE::
                                  .BLKB   1
                  00058 MIN_MODULES_DESC::
                                  .BLKB   4
                  0005C MAX_MODULES_DESC::
                                  .BLKB   4
                  00060 MIN_MAX_TABLE_SIZES::
                                  .BLKB   4
                  00064 MIN_RANGE_TABLE_ADDR::
                                  .BLKB   4
                  00068 MODULE_NAME_DESC::
                                  .BLKB   4
                  0006C PACKET_PROCESSOR_DEVICES::
                                  .BLKB   4
                  00070 PROCESSOR_TYPE_TABLE::
                                  .BLKB   4
                  00074 REALTIME_DEVICES::
                                  .BLKB   4
                  00078 RECORD_DESC::
                                  .BLKB   8
                  00080 SCOM_DEVICES::
                                  .BLKB   4
                  00084 SELECTED::
                                  .BLKB   4
                  00088 SUMMARY_DISPATCHER_ADDR::
                                  .BLKB   4
                  0008C TAPE_DEVICES::
                                  .BLKB   4
                  00090 TABLE_ADDRESS::
                                  .BLKB   4
                  00094 TABLE_LENGTH::
                                  .BLKB   1
                  00095           .BLKB   3
                  00098 TEXT_RFA::
                                  .BLKB   8
                  000A0 TOKEN_DESC::
                                  .BLKB   8
                  000A8 TOTAL_SELECTED::
                                  .BLKB   4
                  000AC TOTAL_REJECTED::
                                  .BLKB   4
        00000000  000B0 UNKNOWN_ENTRY::
                                  .LONG   0                                    ;
                  000B4 WORKSTATION_DEVICES::
                                  .BLKB   4
```

```
                                                 ERFLIB_DESC==         P.AAA
                                                               .EXTRN  EXEC_IMAGE, DEVICE_TYPE_ENTRY
                                                               .EXTRN  GET_VM, IMAGE_LOADER
                                                               .EXTRN  LBR$CLOSE, LBR$GET_RECORD
                                                               .EXTRN  LBR$INI_CONTROL
                                                               .EXTRN  LBR$LOOKUP_KEY, LBR$OPEN
                                                               .EXTRN  LBR$SET_LOCATE, LIB$CVT_DTB
                                                               .EXTRN  LIB$EXTZV, LIB$FILE_SCAN
                                                               .EXTRN  LOG_FILENAME, MAP_IMAGE
                                                               .EXTRN  PARSE_COMMAND, PARSE_OUTPUT_FILES
                                                               .EXTRN  RECORD_SELECTED
                                                               .EXTRN  TIMRB, TIMRE, UNKNOWN_DISPATCHER
                                                               .EXTRN  WRITE_MSG, CLASS_DIR
                                                               .EXTRN  EMB, INPUT_FAB, INPUT_RAB
                                                               .EXTRN  INPUT_NAM, INPUT_XABFHC
                                                               .EXTRN  LSTLUN_RAB_ADDRESS
                                                               .EXTRN  OPTION_FLAG, OUTPUT_FAB
                                                               .EXTRN  OUTPUT_NAM, OUTPUT_RAB
                                                               .EXTRN  PARSER_DATA, PARSER_TABLE
                                                               .EXTRN  RELATED_NAM, REJECTED_FAB
                                                               .EXTRN  REJECTED_NAM, REJECTED_RAB
                                                               .EXTRN  SUMMARY_FLAG, SYECOM
                                                               .EXTRN  SYS$OUTPUT_RAB_ADDRESS
                                                               .EXTRN  WORST_ERROR, LNM$FILE_DEV_DESC
                                                               .EXTRN  BUS_IMAGE, BUS_VERSION
                                                               .EXTRN  BUS_XFER_ADDR, DISK_IMAGE
                                                               .EXTRN  DISK_VERSION, DISK_XFER_ADDR
                                                               .EXTRN  LP_IMAGE, LP_VERSION
                                                               .EXTRN  LP_XFER_ADDR, MAX_MISC_TYPE
                                                               .EXTRN  MAX_LP_TYPE, PACKET_PROCESSOR_XFER_ADDR
                                                               .EXTRN  PACKET_PROCESSOR_IMAGE
                                                               .EXTRN  PACKET_PROCESSOR_VERSION
                                                               .EXTRN  REALTIME_IMAGE, REALTIME_VERSION
                                                               .EXTRN  REALTIME_XFER_ADDR
                                                               .EXTRN  SCOM_IMAGE, SCOM_VERSION
                                                               .EXTRN  SCOM_XFER_ADDR, TAPE_IMAGE
                                                               .EXTRN  TAPE_VERSION, TAPE_XFER_ADDR
                                                               .EXTRN  TRANSLATE_ENTRY_TABLE
                                                               .EXTRN  WORKSTATION_IMAGE
                                                               .EXTRN  WORKSTATION_VERSION
                                                               .EXTRN  WORKSTATION_XFER_ADDR
                                                               .EXTRN  ERF_BADEVTYP, ERF_BADEVVAL
                                                               .EXTRN  ERF_BADMODNAM, ERF_CLSTBLERR
                                                               .EXTRN  ERF_CVTERR, ERF_HERALD
                                                               .EXTRN  ERF_INPUT, ERF_LOADERR
                                                               .EXTRN  ERF_NOTFOUND, ERF_INVREPTYP
                                                               .EXTRN  ERF_TOOMANCLS, ERF_TOTAL
                                                               .EXTRN  ERF_UNKENTRY, ERF_UNKCLASS

                                                               .PSECT  $CODE,NOWRT,  PIC,2

                                       0000 00000 ERF:         .WORD   Save nothing                                 : 0909
                          6D     0010    CF  DE 00002          MOVAL   1$, (FP)                                     : 0910
               00000000V  00             00  FB 00007          CALLS   #0, ERF_CONTROL                              : 0956
               00000000G  00             50  D0 0000E          MOVL    R0, WORST_ERROR                              :
                                             04 00015          RET                                                  : 0969
                                       0000 00016 1$:          .WORD   Save nothing                                 : 0910
```

```
                                         7E  D4 00018          CLRL    -(SP)                                              ;
                                         5E  DD 0001A          PUSHL   SP                                                 ;
                             7E       04 AC  7D 0001C          MOVQ    4(AP), -(SP)                                       ;
                00000000V  00            03  FB 00020          CALLS   #3, HANDLER                                        ;
                                             04 00027          RET                                                        ;

; Routine Size:  40 bytes,    Routine Base:  $CODE + 0000
```

```
1 Routine ERF_CONTROL =                                          ! Main control loop for ERF
2 BEGIN
2
2 !++
2 ! Functional description
2 !
2 !       This is the control routine for the ERF facility.
2 !       Any errors encountered will be passed back to this routine.
2 !
2 ! Calling sequence
2 !
2 !       ERF_CONTROL ()
2 !
2 ! Input parameters
2 !
2 !
2 ! Output parameters
2 !
2 !       None
2 !
2 ! Routine value
2 !
2 !       Worst error is returned.
2 !
2 !----
2
2 LOCAL
2       Field_size:             BYTE INITIAL (6),
2       Scan_context:                INITIAL (0),
2       Start_position:              INITIAL (0),
2       Status,                                 ! Returned status
2       Summary_index,                          ! Index for summary type
2       Text_lib_name,
2       Out_file;
2
2
2 !
2 ! Unconditional call to the runtime statistics package initialization routine.
2 !
2 TIMRB();
2
2
2 $INIT_DYNDESC (input_desc);
2
2 !
2 ! Initialize the number of lines output.
2 !
2 Syecom[sye$b_lines] = 1 ;
2
```

```
2
2 !
2 ! OPEN TEXT LIBRARY
2 !   Call LBR interface routines to prepare the library which
2 !   will provide text for output reports.
2
2 CALL_FUNCTION ( Open_text_lib () );
2
2
2 !
2 ! PROCESS COMMAND LINE
2 !   Call CLI interface routines to parse command line and setup
2 !   internal tables for further processing.
2
2 CALL_FUNCTION ( Parse_command () );
2
2 !
2 ! If /SUMMARY then load the ERFSUMM.EXE. If ERFSUMM.EXE not found clear
2 ! summary flag and continue.
2 !
2
2 If .option_flag [opt$v_summary_qual] then
3  Begin
3  Status = Map_image( AD ('SYS$SYSTEM:ERFSUMM.EXE'), Summary_dispatcher_addr) ;
3  If NOT .status then option_flag [opt$v_summary_qual] = 0;
2  End;
2
2
2 Syecom[sye$l_mailbox_channel] = 0;
2
2 !
2 ! While file names exist in the command line go process the file.
2 !
2
2 While GET_VALUE ('FILE_SPECS', input_desc ) do
3     Begin
3     input_fab [fab$b_fns] = .input_desc [dsc$w_length];
3     input_fab [fab$l_fna] = .input_desc [dsc$a_pointer];
3     input_fab [fab$l_ctx] = msg$_searchfail;       ! Setup the error message
3     LIB$FILE_SCAN (                                ! Call file scanner with -
3                 Input_fab,                         ! address of FAB, -
3                 Process_file,                      ! success action routine, -
3                 Log_filename,                      ! error action routine, -
3                 Scan_context );                    ! and scan context.
2     End;
2
2
2 !
2 ! If /SUMMARY then output summary report type requested.
2 !
2 If .option_flag[opt$v_summary_qual]
2 Then
3     Begin
3
3     Until .start_position GTR 6 do
4         Begin
4
```

```
4                 FFS (start_position,field_size,
4                       .summary_flag,summary_index) ;
4
4                 Case .summary_index from 0 to 5 of
4                     Set
4
4                     [0]:
4                     Function = all_summ_out;    ! Initialize function to output
4                                                 ! all possible summary information
4
4                     [1]:
4                     Function = dev_summ_out;    ! Initialize function to output
4                                                 ! device summary information only
4
4                     [2]:
4                     Function = entry_summ_out;  ! Initialize function to output
4                                                 ! entry summary information only
4
4                     [3]:
4                     Function = memory_summ_out; ! Initialize function to output
4                                                 ! memory summary information only
4
4                     [4]:
4                     Function = volume_summ_out; ! Initialize function to output
4                                                 ! volume summary information only
4
4                     [5]:
4                     Function = histo_summ_out;  ! Initialize function to output
4                                                 ! histogram summary information only
4
4                     [OUTRANGE]:
4                     EXITLOOP;
4
4                     TES;
4
4                 Exec_image ( Summary_dispatcher_addr, Lstlun, Function) ;
4                 Start_position = .summary_index + 1 ;
3                 End ;
2             End;
2
2
2   ! LOG MESSAGE
2   !  If /LOG requested and more then one input file
2   !  was processed then print total number of files processed,
2   !  total records selected and total records rejected.
2   !
2
2   If .option_flag[opt$v_log_qual] and .input_file_count gtru 1 then
2       signal (erf_total, 3, .total_selected, .total_rejected, .input_file_count);
2
2
2   !
2   ! If /STATISTICS was specified then call the runtime statistics display routine.
2   !
2   If .option_flag[opt$v_statistics_qual] then TIMRE(lstlun) ;
2
2
```

```
2 !
2 ! Write original command line, if /REJECTED was not specified.
2 !
2 If ( NOT .option_flag[opt$v_rejected_qual] ) then
2    write_msg ( parser_table[erl$r_cmd_line], 1 );
2
2
2 !
2 ! CLOSE OUTPUT FILES
2 !
2
2 If .syecom [sye$l_forms] OR
2    .option_flag [opt$v_binary_qual]
2 then
3    BEGIN
3    If .option_flag[opt$v_log_qual]
3    then
4       BEGIN
4       Local desc : VECTOR [2,long];
4       Desc[0] = .output_nam [nam$b_rsl];
4       Desc[1] = .output_nam [nam$l_rsa];
4       Signal (msg$_created, 1, desc);
3       END;
2    END;
2
2
2 Output_fab [fab$l_ctx] = msg$_closeout;         ! Assign error messages
2 Rejected_fab [fab$l_ctx] = msg$_closeout;       ! Assign error messages
2
2 If .option_flag [opt$v_output_qual] AND
2    .option_flag [opt$v_binary_qual]
2 then
2    CALL_FUNCTION ( $close (fab = output_fab, err = log_filename) );
2
2
2 If .option_flag[opt$v_rejected_qual]
2 then
3    BEGIN
3    If .option_flag[opt$v_log_qual]
3    then
4       BEGIN
4       Local desc : VECTOR [2,long];
4       Desc[0] = .rejected_nam [nam$b_rsl];
4       Desc[1] = .rejected_nam [nam$l_rsa];
4       Signal (msg$_created, 1, desc);
3       END;
3    CALL_FUNCTION ( $close ( fab = rejected_fab, err = log_filename) );
2    END;
2
2 Return .worst_error
2
1 End;


                                                       .PSECT  $PLIT,NOWRT,NOEXE, PIC,2
```

```
53  46  52  45  3A  4D  45  54  53  59  53  24  53  59  53  00010 P.AAD:  .ASCII  \SYS$SYSTEM:ERFSUMM.EXE\<0><0>           ;
                        00  00  45  58  45  2E  4D  4D  55  0001F                                                         ;
                                          00000016.  00028 P.AAC:  .LONG   22                                         ;
                                          00000000'  0002C         .ADDRESS P.AAD                                     ;
            00  00  53  43  45  50  53  5F  45  4C  49  46  00030 P.AAF:  .ASCII  \FILE_SPECS\<0><0>                  ;
                                          0000000A.  0003C P.AAE:  .LONG   10                                         ;
                                          00000000'  00040         .ADDRESS P.AAF                                     ;

                                                                   .EXTRN  CLI$GET_VALUE, SYS$CLOSE

                                                                   .PSECT  $CODE,NOWRT,  PIC,2

                                            0FFC 00000 ERF_CONTROL:
                                                                   .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11       ; 0970
                     5B 00000000'      00   9E 00002               MOVAB   LSTLUN, R11                                ;
                     5A 00000000G      00   9E 00009               MOVAB   LIB$SIGNAL, R10                            ;
                     59 00000000G      00   9E 00010               MOVAB   LOG_FILENAME, R9                           ;
                     58 00000000G      00   9E 00017               MOVAB   SYECOM+12, R8                              ;
                     57 00000000G      00   9E 0001E               MOVAB   INPUT_FAB+52, R7                           ;
                     56 00000000G      00   9E 00025               MOVAB   OPTION_FLAG, R6                            ;
                     55 00000000'      00   9E 0002C               MOVAB   FUNCTION, R5                               ;
                     5E                08   C2 00033               SUBL2   #8, SP                                     ;
                     54                06   90 00036               MOVB    #6, FIELD_SIZE                             ; 0971
                                       7E   D4 00039               CLRL    SCAN_CONTEXT                               ;
                                       53   D4 0003B               CLRL    START_POSITION                             ;
        00000000G    00                00   FB 0003D               CALLS   #0, TIMRB                                  ; 1009
               14    A5 020E0000       8F   D0 00044               MOVL    #34471936, INPUT_DESC                      ; 1012
                                18     A5   D4 0004C               CLRL    INPUT_DESC+4                               ;
                     68                01   90 0004F               MOVB    #1, SYECOM+12                              ; 1017
        00000000V    00                00   FB 00052               CALLS   #0, OPEN_TEXT_LIB                          ; 1025
                     07                50   E9 00059               BLBC    STATUS, 1$                                 ;
        00000000G    00                00   FB 0005C               CALLS   #0, PARSE_COMMAND                          ; 1033
                     01                50   E8 00063 1$:           BLBS    STATUS, 2$                                 ;
                                            04 00066               RET                                                ;
                     50                66   D0 00067 2$:           MOVL    OPTION_FLAG, R0                            ; 1040
     1B              60                0E   E1 0006A               BBC     #14, (R0), 3$                              ;
                                74     A5   9F 0006E               PUSHAB  SUMMARY_DISPATCHER_ADDR                    ; 1042
                          00000000'    00   9F 00071               PUSHAB  P.AAC                                      ;
        00000000G    00                02   FB 00077               CALLS   #2, MAP_IMAGE                              ;
                     08                50   E8 0007E               BLBS    STATUS, 3$                                 ; 1043
                     50                66   D0 00081               MOVL    OPTION_FLAG, R0                            ;
               01    A0         40     8F   8A 00084               BICB2   #64, 1(R0)                                 ;
                                13     A8   D4 00089 3$:           CLRL    SYECOM+31                                  ; 1047
                                14     A5   9F 0008C 4$:           PUSHAB  INPUT_DESC                                 ; 1053
                          00000000'    00   9F 0008F               PUSHAB  P.AAE                                      ;
        00000000G    00                02   FB 00095               CALLS   #2, CLI$GET_VALUE                          ;
                     27                50   E9 0009C               BLBC    R0, 5$                                     ;
                     67         14     A5   90 0009F               MOVB    INPUT_DESC, INPUT_FAB+52                   ; 1055
               F8    A7         18     A5   D0 000A3               MOVL    INPUT_DESC+4, INPUT_FAB+44                 ; 1056
               E4    A7 0008123A       8F   D0 000A8               MOVL    #528954, INPUT_FAB+24                      ; 1057
                              4200     8F   BB 000B0               PUSHR   #^M<R9,SP>                                 ; 1058
                          00000000V    00   9F 000B4               PUSHAB  PROCESS_FILE                               ;
                                CC     A7   9F 000BA               PUSHAB  INPUT_FAB                                  ;
        00000000G    00                04   FB 000BD               CALLS   #4, LIB$FILE_SCAN                          ;
                                       C6   11 000C4               BRB     4$                                         ; 1053
                     50                66   D0 000C6 5$:           MOVL    OPTION_FLAG, R0                            ; 1069
     53              60                0E   E1 000C9               BBC     #14, (R0), 15$                             ;
```

```
                     06             53  D1 000CD 6$:     CMPL    START_POSITION, #6                              ; 1073
                                    4E  14 000D0         BGTR    15$                                             ;
                     50 00000000G   00  D0 000D2         MOVL    SUMMARY_FLAG, R0                                ; 1077
   52          60    54             53  EA 000D9         FFS     START_POSITION, FIELD_SIZE, (R0), -             ; 1076
                                                                 SUMMARY_INDEX                                   ;
               05    00             52  CF 000DE         CASEL   SUMMARY_INDEX, #0, #5                           ; 1079
 001D        0018  0013           000E     000E2 7$:     .WORD   8$-7$,-                                         ;
                   0027           0022     000EA                 9$-7$,-                                         ;
                                                                 10$-7$,-                                        ;
                                                                 11$-7$,-                                        ;
                                                                 12$-7$,-                                        ;
                                                                 13$-7$                                          ;
                                    30  11 000EE         BRB     15$                                             ; 1107
                     65             01  D0 000F0 8$:     MOVL    #1, FUNCTION                                    ; 1083
                                    17  11 000F3         BRB     14$                                             ;
                     65             02  D0 000F5 9$:     MOVL    #2, FUNCTION                                    ; 1087
                                    12  11 000F8         BRB     14$                                             ;
                     65             04  D0 000FA 10$:    MOVL    #4, FUNCTION                                    ; 1091
                                    0D  11 000FD         BRB     14$                                             ;
                     65             06  D0 000FF 11$:    MOVL    #6, FUNCTION                                    ; 1095
                                    08  11 00102         BRB     14$                                             ;
                     65             07  D0 00104 12$:    MOVL    #7, FUNCTION                                    ; 1099
                                    03  11 00107         BRB     14$                                             ;
                     65             09  D0 00109 13$:    MOVL    #9, FUNCTION                                    ; 1103
                                    55  DD 0010C 14$:    PUSHL   R5                                              ; 1111
                                    5B  DD 0010E         PUSHL   R11                                             ;
                              74    A5  9F 00110         PUSHAB  SUMMARY_DISPATCHER_ADDR                         ;
              00000000G  00         03  FB 00113         CALLS   #3, EXEC_IMAGE                                  ;
                         53   01    A2  9E 0011A         MOVAB   1(R2), START_POSITION                           ; 1112
                                    AD  11 0011E         BRB     6$                                              ; 1073
                     50             66  D0 00120 15$:    MOVL    OPTION_FLAG, R0                                 ; 1123
                                    60  95 00123         TSTB    (R0)                                            ;
                                    19  18 00125         BGEQ    16$                                             ;
                     01       1C    A5  D1 00127         CMPL    INPUT_FILE_COUNT, #1                            ;
                                    13  1B 0012B         BLEQU   16$                                             ;
                              1C    A5  DD 0012D         PUSHL   INPUT_FILE_COUNT                                ; 1124
                     7E     0094    C5  7D 00130         MOVQ    TOTAL_SELECTED, -(SP)                           ;
                                    03  DD 00135         PUSHL   #3                                              ;
                        00000000G   8F  DD 00137         PUSHL   #ERF_TOTAL                                      ;
                     6A             05  FB 0013D         CALLS   #5, LIB$SIGNAL                                  ;
                     50             66  D0 00140 16$:    MOVL    OPTION_FLAG, R0                                 ; 1130
                     09       02    A0  E9 00143         BLBC    2(R0), 17$                                      ;
                                    5B  DD 00147         PUSHL   R11                                             ;
              00000000G  00         01  FB 00149         CALLS   #1, TIMRE                                       ;
                         50         66  D0 00150 17$:    MOVL    OPTION_FLAG, R0                                 ; 1136
               0F        60         0A  E0 00153         BBS     #10, (R0), 18$                                  ;
                                    01  DD 00157         PUSHL   #1                                              ; 1137
                        00000000G   00  DD 00159         PUSHL   PARSER_TABLE                                    ;
              00000000G  00         02  FB 0015F         CALLS   #2, WRITE_MSG                                   ;
                         07   F8    A8  E8 00166 18$:    BLBS    SYECOM+4, 19$                                   ; 1144
                         50         66  D0 0016A         MOVL    OPTION_FLAG, R0                                 ; 1145
               26        60         01  E1 0016D         BBC     #1, (R0), 20$                                   ;
                         50         66  D0 00171 19$:    MOVL    OPTION_FLAG, R0                                 ; 1148
                                    60  95 00174         TSTB    (R0)                                            ;
                                    1F  18 00176         BGEQ    20$                                             ;
                     50 00000000G   00  D0 00178         MOVL    OUTPUT_NAM, R0                                  ; 1152
               04    AE       03    A0  9A 0017F         MOVZBL  3(R0), DESC                                     ;
```

```
                    08   AE         04   A0  D0 00184           MOVL    4(R0), DESC+4                                 ; 1153
                                    04   AE  9F 00189           PUSHAB  DESC                                          ; 1154
                                         01  DD 0018C           PUSHL   #1                                            ;
                              00081073   8F  DD 0018E           PUSHL   #528499                                       ;
                          6A             03  FB 00194           CALLS   #3, LIB$SIGNAL                                ;
             00000000G   00   0008105A   8F  D0 00197 20$:      MOVL    #528474, OUTPUT_FAB+24                        ; 1159
             00000000G   00   0008105A   8F  D0 001A2           MOVL    #528474, REJECTED_FAB+24                      ; 1160
                         50              66  D0 001AD           MOVL    OPTION_FLAG, R0                               ; 1162
                         16         01   A0  E9 001B0           BLBC    1(R0), 21$                                    ;
          12             60              01  E1 001B4           BBC     #1, (R0), 21$                                 ; 1163
                                         59  DD 001B8           PUSHL   R9                                            ; 1165
                              00000000G  00  9F 001BA           PUSHAB  OUTPUT_FAB                                    ;
             00000000G   00              02  FB 001C0           CALLS   #2, SYS$CLOSE                                 ;
                         42              50  E9 001C7           BLBC    STATUS, 24$                                   ;
                         50              66  D0 001CA 21$:      MOVL    OPTION_FLAG, R0                               ; 1168
          34             60              0A  E1 001CD           BBC     #10, (R0), 23$                                ;
                                         60  95 001D1           TSTB    (R0)                                          ; 1171
                                         1E  18 001D3           BGEQ    22$                                           ;
                    04   AE   00000000G  00  9A 001D5           MOVZBL  REJECTED_NAM+3, DESC                          ; 1175
                    08   AE   00000000G  00  D0 001DD           MOVL    REJECTED_NAM+4, DESC+4                        ; 1176
                                    04   AE  9F 001E5           PUSHAB  DESC                                          ; 1177
                                         01  DD 001E8           PUSHL   #1                                            ;
                              00081073   8F  DD 001EA           PUSHL   #528499                                       ;
                          6A             03  FB 001F0           CALLS   #3, LIB$SIGNAL                                ;
                                         59  DD 001F3 22$:      PUSHL   R9                                            ; 1179
                              00000000G  00  9F 001F5           PUSHAB  REJECTED_FAB                                  ;
             00000000G   00              02  FB 001FB           CALLS   #2, SYS$CLOSE                                 ;
                         07              50  E9 00202           BLBC    STATUS, 24$                                   ;
                         50   00000000G  00  D0 00205 23$:      MOVL    WORST_ERROR, R0                               ; 1182
                                             04 0020C 24$:      RET                                                   ; 1184

; Routine Size:  525 bytes,    Routine Base:  $CODE + 0028


;  622           1185  1
```

```
1 Routine PROCESS_FILE (FAB) =
1
1 !++
1 ! Functional description
1 !
1 !       This routine processes one input file. It is
1 !       called as an action routine from LIB$FILE_SCAN.
1 !
1 ! Calling sequence
1 !
1 !       PROCESS_FILE (FAB)
1 !
1 ! Input parameters
1 !
1 !       FAB - address of input_fab.
1 !
1 ! Output parameters
1 !
1 !       None
1 !
1 ! Routine value
1 !
1 !       Worst error is returned.
1 !
1 !----
1
2 BEGIN
2
2 Map
2       Fab:            ref $bblock;
2
2
2 Local
2       Status:         $BBLOCK [LONG],
2       Desc:           VECTOR [2,LONG];        ! General purpose descriptor
2
2
2 Own
2       Class:          WORD,                   ! Class of the image to be loaded
2       First_time:     INITIAL (TRUE),         ! Flag to know if this is first_time
2       Type:           WORD,                   ! Type of the image to be loaded
2       Xfer_addr:      LONG ;                  ! Transfer address of the required image
2
2
```

```
  2 !
  2 ! Establish the input buffer address.
  2 !
  2 Input_rab [rab$l_ubf] = emb ;              ! Load input buffer addr. in RAB
  2
  2 !
  2 ! OPEN AND CONNECT ---
  2 !  To the input error log file.  Exit immediately if any errors
  2 !  are detected. The error handlers will have been invoked if an
  2 !  error occured, and the user will have been notified.  The error
  2 !  message used by the LOG_FILENAME routine is drawn from the
  2 !  CTX field of the FAB or INPUT_RAB as needed.
  2 !
  2
  2 Fab [fab$l_ctx] = msg$_openin;                    ! Specify the error message
P 2 CALL_FUNCTION ( $open (fab=.fab,                  ! OPEN the input file
  2                        err=log_filename) );
P 2 CALL_FUNCTION ( $connect (rab=input_rab,          ! CONNECT to input file
  2                        err=log_filename) );
  2
  2
  2
  2
  2 !
  2 ! INITIALIZE OUTPUT FILES --
  2 !  The processing of the output files has been deferred until now
  2 !  so that a fully parsed input file name would be available (as
  2 !  a related file name) for default file name components.
  2 !
  2
  2 If .input_file_count eql 0 then                   ! If this is the first pass
  2   CALL_FUNCTION ( parse_output_files(Lstlun) ); ! then open the output files
  2
  2 Input_file_count = .input_file_count + 1;         ! Count the number of files we process.
  2
  2 Lstlun = .syecom[sye$l_lstlun];                   ! Initialize the fortran logical unit number
  2
  2
  2 !
  2 ! RESET THE RECORD COUNTERS--
  2 !    Reset the file-relative record number such that records in each
  2 !    file will be numbered in ascending order beginning with one.
  2 !    Reset the selected count so it is also on a per-file basis.
  2 !
  2
  2 Syecom[sye$l_reccnt] = 0;                         ! Reset the record number
  2 Selected = 0;                                     ! Reset file select count
  2
```

```
2 !
2 ! READ AND LOOP UNTIL EOF OR ERROR ---
2 !   While status is true, get a record from the input file.
2 !     Increment the SYECOM[SYE$L_RECCNT] which is the record count.
2 !     If the user command specified this type of record needs processing then
2 !      Increment record selected count.
2 !      If /BINARY specified, then
2 !       write the record to the specified binary file,
2 !      else go process the record(error packet).
2 !     else write the error packet to the rejected records file if specified.
2 !   End while
2 !
2
2 While status = $GET (rab=input_rab, err=log_filename) do ! Read a record
3  BEGIN
3  Syecom[sye$l_reccnt] = .syecom[sye$l_reccnt] + 1;  ! Update the record number
3
3  If record_selected () then                   ! Process this record?
4    BEGIN                                      !  Yes
4    Selected = .selected + 1;                  !  Count how many we process
4    If .option flag[opt$v_binary_qual] then    !  If /BINARY write packet
5     CALL_FUNCTION ( write_binary (emb, output_rab) )
4    Else
4     CALL_FUNCTION ( process_packet () );      ! Analyze packet
4    End
3  Else
3    If .option_flag[opt$v_rejected_qual]       ! If /REJECTED write packet
3    then CALL_FUNCTION ( write_binary (emb, rejected_rab) );
3
3 !
3 ! Determine if the end value for a '/entry=end' was found.
3 ! Yes, set up status and exit as if end of file.
3 !
3  If .syecom[sye$b_end_value] then
4    Begin
4    Status = RMS$_EOF ;
4    Exitloop ;
3    End ;
2  END;
2
2 Total_selected = .total_selected + .selected;   ! Accumulate totals
2 Total_rejected = .total_rejected + (.syecom[sye$l_reccnt] - .selected);
2
2
2 !
2 !CHECK STATUS AT END OF LOOP ---
2 ! Now check the return status to make sure it was a normal EOF.  If not,
2 ! notify the user.
2 !
2
3 If not (.status eql rms$_eof)             ! If any status other than
2    then return .status;                   !  expected eof, return it
2
2 !
2 ! Indicate that end of file occurred.
2 !
2 Syecom[sye$b_eof_flag] = true ;
```

```
;  776      1335  2  !
;  777      1336  2  !  Display MSCP messages
;  778      1337  2  !  Determine if a full report should be generated, so that the MSCP
;  779      1338  2  !  messages can be de-queued and output. (This must be done after EOF
;  780      1339  2  !  because the number of entries for a given cmd ref number is unknown
;  781      1340  2  !  due to the number of retries for the I/O, etc.).
;  782      1341  2  !  ************************************
;  783      1342  2  !   Brief reports are handled in the root and don't seem to require this
;  784      1343  2  !   call back to de-queue any information.???????
;  785      1344  2  !   Summary information is updated when the first part of an MSCP message
;  786      1345  2  !   is seen. Summaries for device rollup worked before this code was put in.
;  787      1346  2  !
;  788      1347  2  If .parser_data[erl$b_rpt_type] EQLU FULL_REP
;  789      1348  2  Then
;  790      1349  2      !
;  791      1350  2      ! Set up the class/type and attempt to load the ERFPROC1 image. If the
;  792      1351  2      ! image was already loaded the xfer address will be returned.
;  793      1352  2      !
;  794      1353  3      Begin
;  795      1354  3      Class = 0 ;
;  796      1355  3      Type = EMB$C_SP ;
;  797      1356  3
;  798      1357  3      Worst_error = Image_loader ( type, class, xfer_addr );
;  799      1358  3
;  800      1359  3      If .Xfer_addr NEQU 0
;  801      1360  3      Then
;  802      1361  3          !
;  803      1362  3          ! Call the device dependent module to produce a full report.
;  804      1363  3          ! (Record_size is passed as a count of 1 and record number is unknown
;  805      1364  3          ! at this time and should not matter, because when ERLLOGSTS sees that
;  806      1365  3          ! EOF was seen it will call the DUDRIVER_MSCP_DQ routine to output the
;  807      1366  3          ! remainder of the MSCP message information).
;  808      1367  3          !
;  809      1368  4          BEGIN
;  810      1369  4          Syecom[sye$l_options] = %c'S';
;  811      1370  4          Exec_image ( Xfer_addr, Lstlun, %REF(1), syecom[sye$l_reccnt],
;  812      1371  4                       AD('S') );
;  813      1372  3          End;
;  814      1373  2      End ;
;  815      1374  2
;  816      1375  2  !
;  817      1376  2  ! Determine whether any of the specified enrtries were found.
;  818      1377  2  ! If no entries found. Output an informational message for the user.
;  819      1378  2  !
;  820      1379  2
;  821      1380  2  If .total_selected EQL 0 then signal (erf_notfound) ;
;  822      1381  2
;  823      1382  2  !
;  824      1383  2  ! CLOSE ---
;  825      1384  2  !       Input file processing is now complete.  Revise the stored error
;  826      1385  2  !       message (which is passed to the error routine via the 'user context'
;  827      1386  2  !       field [CTX] of the FAB) and $CLOSE the input file.
;  828      1387  2  !
;  829      1388  2
;  830      1389  2  Fab [fab$l_ctx] = msg$_closein;                       ! Assign error message
;  831      1390  2  CALL_FUNCTION ( $close (fab=.fab, err=log_filename) );  ! Close the input file
;  832      1391  2
;  833          1392  2
;  834          1393  2 If .option_flag[opt$v_log_qual] then               ! If /LOG requested
;  835          1394  3    BEGIN
;  836          1395  3    Desc [0] = .input_nam [nam$b_rsl];              !  then notify the user
;  837          1396  3    Desc [1] = .input_nam [nam$l_rsa];              !  with file name and counts
;  838          1397  3    Signal (erf_input, 3, desc, .selected, .syecom[sye$l_reccnt] - .selected);
;  839          1398  2    END;
;  840          1399  2
;  841          1400  2
;  842          1401  2
;  843          1402  2 Return true;
;  844          1403  2
;  845          1404  1 END;


                                                                   .PSECT  $PLIT,NOWRT,NOEXE,  PIC,2

                                  00  00  00  53  00044 P.AAH:     .ASCII  \S\<0><0><0>                              ;
                                      00000001  00048 P.AAG:     .LONG   1                                         ;
                                      00000000' 0004C            .ADDRESS P.AAH                                    ;

                                                                   .PSECT  $OWN$,NOEXE,  PIC,2

                                                00004 CLASS:     .BLKB   2
                                                00006            .BLKB   2
                                      00000001  00008 FIRST_TIME:
                                                                   .LONG   1                                         ;
                                                0000C TYPE:      .BLKB   2
                                                0000E            .BLKB   2
                                                00010 XFER_ADDR:
                                                                   .BLKB   4

                                                                   .EXTRN  SYS$OPEN, SYS$CONNECT
                                                                   .EXTRN  SYS$GET

                                                                   .PSECT  $CODE,NOWRT,  PIC,2

                                           0FFC 00000 PROCESS_FILE:
                                                                   .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11      ; 1186
                        5B 00000000G   00   9E 00002             MOVAB   EMB, R11                                  ;
                        5A 00000000G   00   9E 00009             MOVAB   LIB$SIGNAL, R10                           ;
                        59 00000000G   00   9E 00010             MOVAB   OPTION_FLAG, R9                           ;
                        58 00000000G   00   9E 00017             MOVAB   INPUT_RAB, R8                             ;
                        57 00000000G   00   9E 0001E             MOVAB   LOG_FILENAME, R7                          ;
                        56 00000000G   00   9E 00025             MOVAB   SYECOM, R6                                ;
                        55 00000000'   00   9E 0002C             MOVAB   SELECTED, R5                              ;
                        54 00000000'   00   9E 00033             MOVAB   LSTLUN, R4                                ;
                        5E             0C   C2 0003A             SUBL2   #12, SP                                   ;
                 24     A8             6B   9E 0003D             MOVAB   EMB, INPUT_RAB+36                         ; 1233
                        53       04    AC   D0 00041             MOVL    FAB, R3                                   ; 1244
                 18     A3 0008109A    8F   D0 00045             MOVL    #528538, 24(R3)                           ;
                               0088    8F   BB 0004D             PUSHR   #^M<R3,R7>                                ; 1246
           00000000G    00             02   FB 00051             CALLS   #2, SYS$OPEN                              ;
                        77             50   E9 00058             BLBC    STATUS, 6$                                ;
                                       57   DD 0005B             PUSHL   R7                                        ; 1248
                                       58   DD 0005D             PUSHL   R8                                        ;
```

```
          00000000G  00              02  FB 0005F          CALLS    #2, SYS$CONNECT                          ;
                     69              50  E9 00066          BLBC     STATUS, 6$                               ;
                               AC    A5  D5 00069          TSTL     INPUT_FILE_COUNT                         ; 1260
                                     0C  12 0006C          BNEQ     1$                                       ;
                                     54  DD 0006E          PUSHL    R4                                       ; 1261
          00000000G  00              01  FB 00070          CALLS    #1, PARSE_OUTPUT_FILES                   ;
                     58              50  E9 00077          BLBC     STATUS, 6$                               ;
                               AC    A5  D6 0007A  1$:     INCL     INPUT_FILE_COUNT                         ; 1263
                     64        27    A6  D0 0007D          MOVL     SYECOM+39, LSTLUN                        ; 1265
                                     66  D4 00081          CLRL     SYECOM                                   ; 1275
                                     65  D4 00083          CLRL     SELECTED                                 ; 1276
                                     57  DD 00085  2$:     PUSHL    R7                                       ; 1291
                                     58  DD 00087          PUSHL    R8                                       ;
          00000000G  00              02  FB 00089          CALLS    #2, SYS$GET                              ;
                     52              50  D0 00090          MOVL     R0, STATUS                               ;
                     4B              52  E9 00093          BLBC     STATUS, 8$                               ;
                                     66  D6 00096          INCL     SYECOM                                   ; 1293
          00000000G  00              00  FB 00098          CALLS    #0, RECORD_SELECTED                      ; 1295
                     1A              50  E9 0009F          BLBC     R0, 4$                                   ;
                                     65  D6 000A2          INCL     SELECTED                                 ; 1297
                     50              69  D0 000A4          MOVL     OPTION_FLAG, R0                          ; 1298
     08              60              01  E1 000A7          BBC      #1, (R0), 3$                             ;
                          00000000G  00  9F 000AB          PUSHAB   OUTPUT_RAB                               ; 1299
                                     16  11 000B1          BRB      5$                                       ;
          00000000V  00              00  FB 000B3  3$:     CALLS    #0, PROCESS_PACKET                       ; 1301
                                     16  11 000BA          BRB      6$                                       ;
                     50              69  D0 000BC  4$:     MOVL     OPTION_FLAG, R0                          ; 1304
     13              60              0A  E1 000BF          BBC      #10, (R0), 7$                            ;
                          00000000G  00  9F 000C3          PUSHAB   REJECTED_RAB                             ; 1305
                                     5B  DD 000C9  5$:     PUSHL    R11                                      ;
          00000000V  00              02  FB 000CB          CALLS    #2, WRITE_BINARY                         ;
                     01              50  E8 000D2  6$:     BLBS     STATUS, 7$                               ;
                                     04  000D5             RET                                               ;
                     AB        1E    A6  E9 000D6  7$:     BLBC     SYECOM+30, 2$                            ; 1311
                     52  0001827A    8F  D0 000DA          MOVL     #98938, STATUS                           ; 1313
                     50              65  D0 000E1  8$:     MOVL     SELECTED, R0                             ; 1318
                24   A5              50  C0 000E4          ADDL2    R0, TOTAL_SELECTED                       ;
     50              66              50  C3 000E8          SUBL3    R0, SYECOM, R0                           ; 1319
                28   A5              50  C0 000EC          ADDL2    R0, TOTAL_REJECTED                       ;
          0001827A   8F              52  D1 000F0          CMPL     STATUS, #98938                           ; 1328
                                     04  13 000F7          BEQL     9$                                       ;
                     50              52  D0 000F9          MOVL     STATUS, R0                               ; 1329
                                     04  000FC             RET                                               ;
                10   A6              01  90 000FD  9$:     MOVB     #1, SYECOM+29                            ; 1334
                     50  00000000G   00  D0 00101          MOVL     PARSER_DATA, R0                          ; 1347
                     02              60  91 00108          CMPB     (R0), #2                                 ;
                                     44  12 0010B          BNEQ     10$                                      ;
                               04    A4  B4 0010D          CLRW     CLASS                                    ; 1354
                0C   A4        63    8F  9B 00110          MOVZBW   #99, TYPE                                ; 1355
                               10    A4  9F 00115          PUSHAB   XFER_ADDR                                ; 1357
                               04    A4  9F 00118          PUSHAB   CLASS                                    ;
                               0C    A4  9F 0011B          PUSHAB   TYPE                                     ;
          00000000G  00              03  FB 0011E          CALLS    #3, IMAGE_LOADER                         ;
          00000000G  00              50  D0 00125          MOVL     R0, WORST_ERROR                          ;
                               10    A4  D5 0012C          TSTL     XFER_ADDR                                ; 1359
                                     20  13 0012F          BEQL     10$                                      ;
                2B   A6        53    8F  9A 00131          MOVZBL   #83, SYECOM+43                           ; 1369
```

```
                     00000000'  00  9F 00136          PUSHAB  P.AAG                                   ; 1371
                                56  DD 0013C          PUSHL   R6                                      ; 1370
              08   AE           01  D0 0013E          MOVL    #1, 8(SP)                               ;
                           08   AE  9F 00142          PUSHAB  8(SP)                                   ;
                                54  DD 00145          PUSHL   R4                                      ;
                           10   A4  9F 00147          PUSHAB  XFER_ADDR                               ;
                00000000G  00   05  FB 0014A          CALLS   #5, EXEC_IMAGE                          ;
                           24   A5  D5 00151 10$:     TSTL    TOTAL_SELECTED                          ; 1380
                                09  12 00154          BNEQ    11$                                     ;
                     00000000G  8F  DD 00156          PUSHL   #ERF_NOTFOUND                           ;
                           6A   01  FB 0015C          CALLS   #1, LIB$SIGNAL                          ;
              18   A3 00081052  8F  D0 0015F 11$:     MOVL    #528466, 24(R3)                         ; 1389
                         0088   8F  BB 00167          PUSHR   #^M<R3,R7>                              ; 1390
                00000000G  00   02  FB 0016B          CALLS   #2, SYS$CLOSE                           ;
                           31   50  E9 00172          BLBC    STATUS, 13$                             ;
                           50   69  D0 00175          MOVL    OPTION_FLAG, R0                         ; 1393
                                60  95 00178          TSTB    (R0)                                    ;
                                27  18 0017A          BGEQ    12$                                     ;
              04   AE 00000000G 00  9A 0017C          MOVZBL  INPUT_NAM+3, DESC                       ; 1395
              08   AE 00000000G 00  D0 00184          MOVL    INPUT_NAM+4, DESC+4                     ; 1396
                           50   65  D0 0018C          MOVL    SELECTED, R0                            ; 1397
         7E                66   50  C3 0018F          SUBL3   R0, SYECOM, -(SP)                       ;
                                50  DD 00193          PUSHL   R0                                      ;
                           0C   AE  9F 00195          PUSHAB  DESC                                    ;
                                03  DD 00198          PUSHL   #3                                      ;
                     00000000G  8F  DD 0019A          PUSHL   #ERF_INPUT                              ;
                           6A   05  FB 001A0          CALLS   #5, LIB$SIGNAL                          ;
                           50   01  D0 001A3 12$:     MOVL    #1, R0                                  ; 1402
                                04     001A6 13$:     RET                                             ; 1404

; Routine Size:  423 bytes,    Routine Base:  $CODE + 0235
```

```
Routine PROCESS_PACKET =
BEGIN
!++
! Functional description
!
!       This routine determines which dispatcher to call for processing
!       the specified report type. Any errors encountered will be passed
!       back to the caller.
!
! Calling sequence
!
!       Process_packet ( )
!
! Input parameters
!
!
! Output parameters
!
!       None
!
! Routine value
!
!       Worst error is returned.
!
!----

Literal
        No_full = 0 ;

Local
   Status;

Global
   Brief_xfer_addr;

syecom[sye$l_record_size] = .input_rab [rab$w_rsz];

If (NOT .unknown_entry) then                  ! If not an unknown entry then
   Begin
   Case .parser_data[erl$b_rpt_type]          ! Case on report type value
       From 0 to REG_DUMP_REP of
       Set

       [No_full]:
         If .option_flag[opt$v_summary_qual] then
            CALL_FUNCTION ( full_dispatcher () ) ;


       [Brief_rep]:                           ! Go output a Brief report
         Begin
         If .Brief_xfer_addr EQL 0 then
            Begin
            Status = map_image (AD ('SYS$SYSTEM:ERFBRIEF.EXE'),brief_xfer_addr);
            If NOT .status then return true;
            End;

         Syecom[sye$l_options] = %c'B';
```

```
	Exec_image( Brief_xfer_addr,
		    Lstlun,
		    AD('B'),
		    syecom[sye$l_record_size],
		    %REF(.option_flag[opt$v_summary_qual]),
		    .Summary_flag) ;
	End;

      [Full_rep]:			! Go output a Full report
	Begin
	!
	! Determine whether the fortran text commons (giocommon, opcodes, modes)
	! have been initialized, if not then call the init_commons routine to
	! initialize them. Then call the full dispatcher.
	!
	If NOT .inited_commons then CALL_FUNCTION (init_commons()) ;
	CALL_FUNCTION ( Full_dispatcher () );
	End ;


      [Reg_dump_rep]:			! Go output a Register Dump report
	Begin
	If .Brief_xfer_addr EQL 0 then
	    Begin
	    Status = map_image (AD ('SYS$SYSTEM:ERFBRIEF.EXE'),brief_xfer_addr);
	    If NOT .status then return true;
	    End;

	Syecom[sye$l_options] = %c'C';
	Exec_image( Brief_xfer_addr,
		    Lstlun,
		    AD('C'),
		    syecom[sye$l_record_size],
		    %REF(.option_flag[opt$v_summary_qual]),
		    .Summary_flag) ;

	End;

      [OUTRANGE]:
	Signal (erf_invreptyp);

      TES;
      End
Else
      If .parser_data[erl$b_rpt_type] NEQU NO_FULL then
	 Unknown_dispatcher ();		! Call unknown dispatcher


!
! If /summary was specified then call summary_dispatcher.
!
If .option_flag [opt$v_summary_qual] then
 Begin
 If (.summary_flag[sum$v_device]  OR .summary_flag[sum$v_all_summ]) then
 Exec_image ( Summary_dispatcher_addr, Lstlun, %REF(dev_summ_upd)) ;

 If (.summary_flag[sum$v_histogram] OR .summary_flag[sum$v_all_summ]) then
```

```
;   961          1519  3   Exec_image ( Summary_dispatcher_addr, Lstlun, %REF(histo_summ_upd)) ;
;   962          1520  2   End;
;   963          1521  2
;   964          1522  2 Return true;
;   965          1523  1 End;



                                                                      .PSECT  $PLIT,NOWRT,NOEXE,  PIC,2

42  46  52  45  3A  4D  45  54  53  59  53  24  53  59  53  00050 P.AAJ:  .ASCII  \SYS$SYSTEM:ERFBRIEF.EXE\<0>              ;
                        00  45  58  45  2E  46  45  49  52  0005F                                                        ;
                                         00000017  00068 P.AAI:  .LONG   23                                               ;
                                         00000000' 0006C         .ADDRESS P.AAJ                                          ;
                                  00  00  00  42  00070 P.AAL:  .ASCII  \B\<0><0><0>                                     ;
                                         00000001  00074 P.AAK:  .LONG   1                                                ;
                                         00000000' 00078         .ADDRESS P.AAL                                          ;
42  46  52  45  3A  4D  45  54  53  59  53  24  53  59  53  0007C P.AAN:  .ASCII  \SYS$SYSTEM:ERFBRIEF.EXE\<0>              ;
                        00  45  58  45  2E  46  45  49  52  0008B                                                        ;
                                         00000017  00094 P.AAM:  .LONG   23                                               ;
                                         00000000' 00098         .ADDRESS P.AAN                                          ;
                                  00  00  00  43  0009C P.AAP:  .ASCII  \C\<0><0><0>                                     ;
                                         00000001  000A0 P.AAO:  .LONG   1                                                ;
                                         00000000' 000A4         .ADDRESS P.AAP                                          ;

                                                                      .PSECT  $GLOBAL$,NOEXE,  PIC,2

                                                   000B8 BRIEF_XFER_ADDR::
                                                                      .BLKB   4


                                                                      .PSECT  $CODE,NOWRT,  PIC,2

                                              03FC 00000 PROCESS_PACKET:
                                                                      .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9          ; 1405
                          59 00000000G  00  9E 00002             MOVAB   MAP_IMAGE, R9                          ;
                          58 00000000G  00  9E 00009             MOVAB   EXEC_IMAGE, R8                         ;
                          57 00000000'  00  9E 00010             MOVAB   LSTLUN, R7                             ;
                          56 00000000G  00  9E 00017             MOVAB   SUMMARY_FLAG, R6                       ;
                          55 00000000'  00  9E 0001E             MOVAB   P.AAI, R5                              ;
                          54 00000000G  00  9E 00025             MOVAB   OPTION_FLAG, R4                        ;
                          53 00000000G  00  9E 0002C             MOVAB   SYECOM+35, R3                          ;
                          52 00000000'  00  9E 00033             MOVAB   BRIEF_XFER_ADDR, R2                    ;
                          5E             04  C2 0003A             SUBL2   #4, SP                                 ;
                          63 00000000G  00  3C 0003D             MOVZWL  INPUT_RAB+34, SYECOM+35                ; 1440
                          50 00000000G  00  D0 00044             MOVL    PARSER_DATA, R0                        ; 1444
                          03         F8  A2  E9 0004B             BLBC    UNKNOWN_ENTRY, 1$                      ; 1442
                                   009E  31 0004F             BRW     13$                                    ;
                   03          00    60  8F 00052 1$:         CASEB   (R0), #0, #3                           ; 1444
     0067          004C        0021        0018       00056 2$:         .WORD   4$-2$,-                                ;
                                                                              5$-2$,-                                ;
                                                                              7$-2$,-                                ;
                                                                              9$-2$                                  ;
                          00000000G  8F  DD 0005E             PUSHL   #ERF_INVREPTYP                         ; 1501
              00000000G  00          01  FB 00064             CALLS   #1, LIB$SIGNAL                         ;
                                   008D  31 0006B 3$:         BRW     14$                                    ;
```

```
                                                          H  4
ERF             Errorlog Report Formatter            15-Sep-1984 23:42:14    VAX-11 Bliss-32 V4.0-742                 Page  30
V04-000                                              14-Sep-1984 12:27:17    DISK$VMSMASTER:[ERF.SRC]ERF.B32;1         (11)

                         50              64  D0 0006E 4$:     MOVL    OPTION_FLAG, R0                          ; 1449
              F6         60              0E  E1 00071         BBC     #14, (R0), 3$                             ;
                                         3B  11 00075         BRB     8$                                        ; 1450
                                         62  D5 00077 5$:     TSTL    BRIEF_XFER_ADDR                           ; 1455
                                         0D  12 00079         BNEQ    6$                                        ;
                                         52  DD 0007B         PUSHL   R2                                        ; 1457
                                         55  DD 0007D         PUSHL   R5                                        ;
                         69              02  FB 0007F         CALLS   #2, MAP_IMAGE                             ;
                         51              50  D0 00082         MOVL    R0, STATUS                                ;
                         44              51  E9 00085         BLBC    STATUS, 10$                               ; 1458
                 08      A3        42    8F  9A 00088 6$:     MOVZBL  #66, SYECOM+43                            ; 1461
                                         66  DD 0008D         PUSHL   SUMMARY_FLAG                              ; 1467
                         50              64  D0 0008F         MOVL    OPTION_FLAG, R0                           ; 1466
  04    AE               60              01                                                                     ;
                                         0E  EF 00092         EXTZV   #14, #1, (R0), 4(SP)                      ;
                                   04    AE  9F 00098         PUSHAB  4(SP)                                     ;
                                         53  DD 0009B         PUSHL   R3                                        ; 1465
                                   0C    A5  9F 0009D         PUSHAB  P.AAK                                     ; 1464
                                         45  11 000A0         BRB     12$                                       ; 1462
                         0B      FF6C    C2  E8 000A2 7$:     BLBS    INITED_COMMONS, 8$                        ; 1477
             00000000V   00              00  FB 000A7         CALLS   #0, INIT_COMMONS                          ;
                         01              50  E8 000AE         BLBS    STATUS, 8$                                ;
                                             04 000B1         RET                                               ;
             00000000V   00              00  FB 000B2 8$:     CALLS   #0, FULL_DISPATCHER                       ; 1478
                         3F              50  E8 000B9         BLBS    STATUS, 14$                               ;
                                             04 000BC         RET                                               ;
                                         62  D5 000BD 9$:     TSTL    BRIEF_XFER_ADDR                           ; 1484
                                         0E  12 000BF         BNEQ    11$                                       ;
                                         52  DD 000C1         PUSHL   R2                                        ; 1486
                                   2C    A5  9F 000C3         PUSHAB  P.AAM                                     ;
                         69              02  FB 000C6         CALLS   #2, MAP_IMAGE                             ;
                         51              50  D0 000C9         MOVL    R0, STATUS                                ;
                         61              51  E9 000CC 10$:    BLBC    STATUS, 18$                               ; 1487
                 08      A3        43    8F  9A 000CF 11$:    MOVZBL  #67, SYECOM+43                            ; 1490
                                         66  DD 000D4         PUSHL   SUMMARY_FLAG                              ; 1496
                         50              64  D0 000D6         MOVL    OPTION_FLAG, R0                           ; 1495
  04    AE               60              01                                                                     ;
                                         0E  EF 000D9         EXTZV   #14, #1, (R0), 4(SP)                      ;
                                   04    AE  9F 000DF         PUSHAB  4(SP)                                     ;
                                         53  DD 000E2         PUSHL   R3                                        ; 1494
                                   38    A5  9F 000E4         PUSHAB  P.AAO                                     ; 1493
                                 0084    8F  BB 000E7 12$:    PUSHR   #^M<R2,R7>                                ; 1491
                         68              06  FB 000EB         CALLS   #6, EXEC_IMAGE                            ;
                                         0B  11 000EE         BRB     14$                                       ; 1442
                                         60  95 000F0 13$:    TSTB    (R0)                                      ; 1506
                                         07  13 000F2         BEQL    14$                                       ;
             00000000G   00              00  FB 000F4         CALLS   #0, UNKNOWN_DISPATCHER                    ; 1507
                         50              64  D0 000FB 14$:    MOVL    OPTION_FLAG, R0                           ; 1513
              2E         60              0E  E1 000FE         BBC     #14, (R0), 18$                            ;
                         50              66  D0 00102         MOVL    SUMMARY_FLAG, R0                          ; 1515
              03         60              01  E0 00105         BBS     #1, (R0), 15$                             ;
                         0D              60  E9 00109         BLBC    (R0), 16$                                 ;
                         6E              03  D0 0010C 15$:    MOVL    #3, (SP)                                  ; 1516
                                 4080    8F  BB 0010F         PUSHR   #^M<R7,SP>                                ;
                                   D0    A2  9F 00113         PUSHAB  SUMMARY_DISPATCHER_ADDR                   ;
                         68              03  FB 00116         CALLS   #3, EXEC_IMAGE                            ;
                         50              66  D0 00119 16$:    MOVL    SUMMARY_FLAG, R0                          ; 1518
              03         60              05  E0 0011C         BBS     #5, (R0), 17$                             ;
                         0D              60  E9 00120         BLBC    (R0), 18$                                 ;
```

```
                     6E        08  D0 00123 17$:    MOVL     #8, (SP)                                          ; 1519
                   4080        8F  BB 00126         PUSHR    #^M<R7,SP>                                        ;
                     D0        A2  9F 0012A         PUSHAB   SUMMARY_DISPATCHER_ADDR                           ;
                     68        03  FB 0012D         CALLS    #3, EXEC_IMAGE                                    ;
                     50        01  D0 00130 18$:    MOVL     #1, R0                                            ; 1522
                                   04 00133         RET                                                        ; 1523

; Routine Size:  308 bytes,    Routine Base:  $CODE + 03DC
```

```
1 Routine FULL_DISPATCHER =                                ! Full report dispatcher
2 BEGIN
2 !++
2 ! Functional description
2 !
2 !       This routine checks to see if the loadable module needed
2 !       to process the error packet is available. If it is not then
2 !       it tries to load the module. If successful then it call the module.
2 !       Any errors encountered will be passed back to this routine.
2 !
2 ! Calling sequence
2 !
2 !       Full_dispatcher ()
2 !
2 ! Input parameters
2 !
2 !
2 ! Output parameters
2 !
2 !       None
2 !
2 ! Routine value
2 !
2 !       Worst error is returned.
2 !
2 !----
2
2 Own
2    Xfer_addr:   LONG,                      ! Transfer address of the required image
2    Class:       WORD,                      ! Class of the image to be loaded
2    Type:        WORD;                      ! Type of the image to be loaded
2
2
2 !
2 ! If the packet entry type is a device error, device timeout, or
2 ! device attention then use the class and type specified in the packet as
2 ! class and type of the image to be loaded. Else use class = 0 and
2 ! type = entry type value.
2 !
2
2 If ( .emb[emb$w_hd_entry] EQLU EMB$C_DE ) OR
2    ( .emb[emb$w_hd_entry] EQLU EMB$C_DT ) OR
3    ( .emb[emb$w_hd_entry] EQLU EMB$C_DA )
2 Then
3    Begin
3    Type = .emb[emb$b_dv_type];
3    Class = .emb[emb$b_dv_class];
3    End
2 Else
3    Begin
3    Type = .emb[emb$w_hd_entry];
3    Class = 0;
2    End;
2
2 !
2 ! Try and load the image. If no image report error and return.
2 !
```

```
2 Worst_error = Image_loader ( type, class, xfer_addr );
2 If .Xfer_addr EQLU 0 then return .worst_error;
2
2 !
2 ! The error packet entry type will determine which call to EXEC_IMAGE should
2 ! be used in order to pass the necessary parameters to the loaded image
2 ! for translation of the error packet.
2 !
2 Case .emb[emb$w_hd_entry] from 1 to EMB$C_UBC of
2 SET
2
2 [ EMB$C_DE,                         ! Device Error 1
2   EMB$C_BE,                         ! Bus Error 4
2   EMB$C_AW,                         ! Asynchronous Write Error 7
2   EMB$C_CS,                         ! Cold start (ie: SYSTEM BOOT) 32 %x20
2   34,                               ! NOT IN DEFINITION FILE 34 %x22
2   EMB$C_NF,                         ! New errlod.sys file created 35 %x23
2   EMB$C_WS,                         ! Warm start (ie: SYSTEM POWER RECOVERY) 36 %x24
2   EMB$C_CR,                         ! Fatal bugcheck 37 %x25
2   EMB$C_TS,                         ! Time stamp entry 38 %X26
2   EMB$C_SS,                         ! System service message 39 %X27
2   EMB$C_SBC,                        ! System bugcheck 40 %X28
2   EMB$C_OM,                         ! Operator message 41 %X29
2   EMB$C_NM,                         ! Network message 42 %X2A
2   EMB$C_DT,                         ! Device Timeout 96 %X60
2   EMB$C_UI,                         ! Undefined interrupt 97 %X61
2   EMB$C_DA,                         ! Asynchronous Device Attention 98 %X62
2   EMB$C_UBC ] :                     ! User bugcheck 112 %X70
2
2     !
2     ! Determine if a full report should be generated.
2     ! Call the device dependent module to produce a full report.
2     ! Else return.
2     !
3     Begin
3     If .parser_data[erl$b_rpt_type] NEQ 0 then Exec_image ( Xfer_addr, Lstlun );
3     Return true;
2     End;
2
2 [ EMB$C_SP,                         !  Software Parameters 99 %X63
2   EMB$C_LM ,                        !  Logged Message 100 %X64
2   EMB$C_LOGMSCP ] :                 !  MSCP message without UCB  101 %x65
2
3     Begin
3     !
3     ! Determine if summary information should be updated.
3     !
3     If (.option_flag[opt$v_summary_qual]) AND
4        (.emb[emb$w_hd_entry] NEQU EMB$C_LOGMSCP)
3     Then
3          ! Yes, call the deivce dependent module for summary updates.
3          !
4          BEGIN
4          Syecom[sye$l_options] = %c'R';
4          Exec_image ( Xfer_addr, Lstlun, syecom[sye$l_record_size],
4                       syecom[sye$l_reccnt], AD('R') );
3          END;
```

```
3
3        !
3        ! If report type is not equal to NOFULL then call the device dependent
3        ! module to produce a full report.
3        !
3        If .parser_data[erl$b_rpt_type] NEQ 0
3        then
4           BEGIN
4           Syecom[sye$l_options] = %c'S';
4           Exec_image ( Xfer_addr, Lstlun, syecom[sye$l_record_size],
4                        syecom[sye$l_reccnt], AD('S') );
3           End;
3        Return true;
2        End;
2
2   [ EMB$C_MC,                          ! Machine check 2
2     EMB$C_SA,                          ! SBI Alert 5
2     EMB$C_SE,                          ! Soft ECC Error 6
2     EMB$C_HE,                          ! Hard ECC Error 8
2     EMB$C_UBA,                         ! 11/780 Unibus Adapter error 9
2     EMB$C_UE,                          ! 11/730 Unibus Error 11 %XB
2     EMB$C_MBA,                         ! 11/780 Massbus Adapter Error 12 %XC
2     EMB$C_VM,                          ! Volume mount 64 %X40
2     EMB$C_VD ] :                       ! Volume dismount 65 %X41
2
3        Begin
3        !
3        ! Determine if summary information should be updated.
3        !
3        If (.parser_data[erl$b_rpt_type] EQL 0) AND
4           (.option_flag[opt$v_summary_qual])
3        Then
3            ! Yes, call the deivce dependent module for summary updates
3            ! and return to the calling routine.
3            !
4            Begin
4            Syecom[sye$l_options] = %c'R';
4            Exec_image (Xfer_addr, Lstlun, AD ('R') );
4            Return true ;
3            End ;
3
3        !
3        ! Call the device dependent module to produce a full report.
3        !
3        Syecom[sye$l_options] = %c'S';
3        Exec_image (Xfer_addr, Lstlun, AD ('S') );
3        Return True;
2        End;
2
2    [EMB$C_SBIA,                                  ! SBI Adaptor error 13 %X0D
2     EMB$C_CRD,                                   ! CRD log 14 %X0E
2     EMB$C_EMM,                                   ! Environmental Monitor 15 %X0F
2     EMB$C_HLT,                                   ! Processor Error Halt 16 %20
2     EMB$C_CRBT]:                                 ! Console Reboot 17 %X21
3            Begin
3            Exec_image (Xfer_addr);
3            Return true;
```

```
; 1138          1695  2         End;
; 1139          1696  2
; 1140          1697  2   [ 3, 10, 18 to 31, 33, 43 to 63, 66 to 95, 102 to 111, outrange ]:
; 1141          1698  2     Return true;
; 1142          1699  2   TES;
; 1143          1700  2
; 1144          1701  1   End;



                                                                  .PSECT  $PLIT,NOWRT,NOEXE,  PIC,2

                                00  00  00  52   000A8 P.AAR:    .ASCII   \R\<0><0><0>                              ;
                                     00000001    000AC P.AAQ:    .LONG     1                                         ;
                                     00000000'   000B0           .ADDRESS P.AAR                                      ;
                                00  00  00  53   000B4 P.AAT:    .ASCII   \S\<0><0><0>                              ;
                                     00000001    000B8 P.AAS:    .LONG     1                                         ;
                                     00000000'   000BC           .ADDRESS P.AAT                                      ;
                                00  00  00  52   000C0 P.AAV:    .ASCII   \R\<0><0><0>                              ;
                                     00000001    000C4 P.AAU:    .LONG     1                                         ;
                                     00000000'   000C8           .ADDRESS P.AAV                                      ;
                                00  00  00  53   000CC P.AAX:    .ASCII   \S\<0><0><0>                              ;
                                     00000001    000D0 P.AAW:    .LONG     1                                         ;
                                     00000000'   000D4           .ADDRESS P.AAX                                      ;

                                                                  .PSECT  $OWN$,NOEXE,  PIC,2

                                                 00014 XFER_ADDR:
                                                                  .BLKB    4
                                                 00018 CLASS:     .BLKB    2
                                                 0001A TYPE:      .BLKB    2


                                                                  .PSECT  $CODE,NOWRT,  PIC,2

                                            03FC 00000 FULL_DISPATCHER:
                                                                  .WORD    Save R2,R3,R4,R5,R6,R7,R8,R9             ; 1524
                   59  00000000G    00   9E 00002                 MOVAB    OPTION_FLAG, R9                          ;
                   58  00000000G    00   9E 00009                 MOVAB    WORST_ERROR, R8                          ;
                   57  00000000G    00   9E 00010                 MOVAB    PARSER_DATA, R7                          ;
                   56  00000000G    00   9E 00017                 MOVAB    EMB+4, R6                                ;
                   55  00000000'    00   9E 0001E                 MOVAB    P.AAQ, R5                                ;
                   54  00000000G    00   9E 00025                 MOVAB    EXEC_IMAGE, R4                           ;
                   53  00000000G    00   9E 0002C                 MOVAB    SYECOM+43, R3                            ;
                   52  00000000'    00   9E 00033                 MOVAB    XFER_ADDR, R2                            ;
                   50               66   3C 0003A                 MOVZWL   EMB+2, R0                                ; 1564
                   01               50   B1 0003D                 CMPW     R0, #1                                   ;
                                    0E   13 00040                 BEQL     1$                                       ;
           0060    8F               50   B1 00042                 CMPW     R0, #96                                  ; 1565
                                    07   13 00047                 BEQL     1$                                       ;
           0062    8F               50   B1 00049                 CMPW     R0, #98                                  ; 1566
                                    0C   12 0004E                 BNEQ     2$                                       ;
             06    A2        19     A6   9B 00050 1$:             MOVZBW   EMB+29, TYPE                             ; 1569
             04    A2        18     A6   9B 00055                 MOVZBW   EMB+28, CLASS                            ; 1570
                                    07   11 0005A                 BRB      3$                                       ; 1564
             06    A2               50   B0 0005C 2$:             MOVW     R0, TYPE                                 ; 1574
```

```
ERF                Errorlog Report Formatter                  N 4
V04-000                                         15-Sep-1984 23:42:14   VAX-11 Bliss-32 V4.0-742               Page 36
                                                14-Sep-1984 12:27:17   DISK$VMSMASTER:[ERF.SRC]ERF.B32;1          (12)

                                          04      A2 B4 00060        CLRW    CLASS                            ; 1575
                                                  52 DD 00063 3$:    PUSHL   R2                               ; 1581
                                          04      A2 9F 00065        PUSHAB  CLASS                            ;
                                          06      A2 9F 00068        PUSHAB  TYPE                             ;
                      00000000G  00               03 FB 0006B        CALLS   #3, IMAGE_LOADER                 ;
                                 68               50 D0 00072        MOVL    R0, WORST_ERROR                  ;
                                                  62 D5 00075        TSTL    XFER_ADDR                        ; 1582
                                                  04 12 00077        BNEQ    4$                               ;
                                 50               68 D0 00079        MOVL    WORST_ERROR, R0                  ;
                                                     04 0007C        RET                                      ;
                                 51               66 3C 0007D 4$:    MOVZWL  EMB+4, R1                        ; 1589
                 006F     8F     01               51 AF 00080        CASEW   R1, #1, #111                     ; 1692
     00E2             0160       0133      00E2        00086 5$:     .WORD   6$-5$,-                          ;
     0133             00E2       0133      0133        0008E                 9$-5$,-                          ;
     0133             0133       0160      0133        00096                 13$-5$,-                         ;
     015B             015B       015B      015B        0009E                 6$-5$,-                          ;
     0160             0160       0160      015B        000A6                 9$-5$,-                          ;
     0160             0160       0160      0160        000AE                 9$-5$,-                          ;
     0160             0160       0160      0160        000B6                 6$-5$,-                          ;
     00E2             0160       0160      0160        000BE                 9$-5$,-                          ;
     00E2             00E2       00E2      0160        000C6                 9$-5$,-                          ;
     00E2             00E2       00E2      00E2        000CE                 13$-5$,-                         ;
     0160             0160       00E2      00E2        000D6                 9$-5$,-                          ;
     0160             0160       0160      0160        000DE                 9$-5$,-                          ;
     0160             0160       0160      0160        000E6                 12$-5$,-                         ;
     0160             0160       0160      0160        000EE                 12$-5$,-                         ;
     0160             0160       0160      0160        000F6                 12$-5$,-                         ;
     0133             0160       0160      0160        000FE                 12$-5$,-                         ;
     0160             0160       0160      0133        00106                 12$-5$,-                         ;
     0160             0160       0160      0160        0010E                 13$-5$,-                         ;
     0160             0160       0160      0160        00116                 13$-5$,-                         ;
     0160             0160       0160      0160        0011E                 13$-5$,-                         ;
     0160             0160       0160      0160        00126                 13$-5$,-                         ;
     0160             0160       0160      0160        0012E                 13$-5$,-                         ;
     0160             0160       0160      0160        00136                 13$-5$,-                         ;
     00E2             0160       0160      0160        0013E                 13$-5$,-                         ;
     00F3             00F3       00E2      00E2        00146                 13$-5$,-                         ;
     0160             0160       0160      00F3        0014E                 13$-5$,-                         ;
     0160             0160       0160      0160        00156                 13$-5$,-                         ;
     00E2             0160       0160      0160        0015E                 13$-5$,-                         ;
                                                                             13$-5$,-                         ;
                                                                             13$-5$,-                         ;
                                                                             13$-5$,-                         ;
                                                                             6$-5$,-                          ;
                                                                             13$-5$,-                         ;
                                                                             6$-5$,-                          ;
                                                                             6$-5$,-                          ;
                                                                             6$-5$,-                          ;
                                                                             6$-5$,-                          ;
                                                                             6$-5$,-                          ;
                                                                             6$-5$,-                          ;
                                                                             6$-5$,-                          ;
                                                                             6$-5$,-                          ;
                                                                             6$-5$,-                          ;
                                                                             13$-5$,-                         ;
                                                                             13$-5$,-                         ;
                                                                             13$-5$,-                         ;
```

```
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
9$-5$,-
9$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
13$-5$,-
6$-5$,-
6$-5$,-
6$-5$,-
7$-5$,-
7$-5$,-
7$-5$,-
13$-5$,-
```

```
                                                                     C  5
ERF              Errorlog Report Formatter                     15-Sep-1984 23:42:14    VAX-11 Bliss-32 V4.0-742               Page  38
V04-000                                                        14-Sep-1984 12:27:17    DISK$VMSMASTER:[ERF.SRC]ERF.B32;1           (12)

                                                                                 13$-5$,-                                  ;
                                                                                 13$-5$,-                                  ;
                                                                                 13$-5$,-                                  ;
                                                                                 13$-5$,-                                  ;
                                                                                 13$-5$,-                                  ;
                                                                                 13$-5$,-                                  ;
                                                                                 13$-5$,-                                  ;
                                                                                 13$-5$,-                                  ;
                                                                                 13$-5$,-                                  ;
                                                                                 6$-5$                                     ;
                                    7E  11 00166          BRB      13$                                       ; 1698
                   50               67  D0 00168  6$:     MOVL     PARSER_DATA, R0                           ; 1616
                                    60  95 0016B          TSTB     (R0)                                      ;
                                    77  13 0016D          BEQL     13$                                       ;
                             EC     A2  9F 0016F          PUSHAB   LSTLUN                                    ;
                                    52  DD 00172          PUSHL    R2                                        ;
                   64               02  FB 00174          CALLS    #2, EXEC_IMAGE                            ;
                                    6D  11 00177          BRB      13$                                       ; 1692
                   50               69  D0 00179  7$:     MOVL     OPTION_FLAG, R0                           ; 1628
        1B         60               0E  E1 0017C          BBC      #14, (R0), 8$                             ;
              0065 8F               51  B1 00180          CMPW     R1, #101                                  ; 1629
                                    14  13 00185          BEQL     8$                                        ;
                   63        52     8F  9A 00187          MOVZBL   #82, SYECOM+43                            ; 1634
                                    55  DD 0018B          PUSHL    R5                                        ; 1636
                             D5     A3  9F 0018D          PUSHAB   SYECOM                                    ;
                             F8     A3  9F 00190          PUSHAB   SYECOM+35                                 ; 1635
                             EC     A2  9F 00193          PUSHAB   LSTLUN                                    ;
                                    52  DD 00196          PUSHL    R2                                        ;
                   64               05  FB 00198          CALLS    #5, EXEC_IMAGE                            ;
                   50               67  D0 0019B  8$:     MOVL     PARSER_DATA, R0                           ; 1643
                                    60  95 0019E          TSTB     (R0)                                      ;
                                    44  13 001A0          BEQL     13$                                       ;
                   63        53     8F  9A 001A2          MOVZBL   #83, SYECOM+43                            ; 1646
                             0C     A5  9F 001A6          PUSHAB   P.AAS                                     ; 1648
                             D5     A3  9F 001A9          PUSHAB   SYECOM                                    ;
                             F8     A3  9F 001AC          PUSHAB   SYECOM+35                                 ; 1647
                             EC     A2  9F 001AF          PUSHAB   LSTLUN                                    ;
                                    52  DD 001B2          PUSHL    R2                                        ;
                   64               05  FB 001B4          CALLS    #5, EXEC_IMAGE                            ;
                                    2D  11 001B7          BRB      13$                                       ; 1692
                   50               67  D0 001B9  9$:     MOVL     PARSER_DATA, R0                           ; 1667
                                    60  95 001BC          TSTB     (R0)                                      ;
                                    10  12 001BE          BNEQ     10$                                       ;
                   50               69  D0 001C0          MOVL     OPTION_FLAG, R0                           ; 1668
        09         60               0E  E1 001C3          BBC      #14, (R0), 10$                            ;
                   63        52     8F  9A 001C7          MOVZBL   #82, SYECOM+43                            ; 1674
                             18     A5  9F 001CB          PUSHAB   P.AAU                                     ; 1675
                                    07  11 001CE          BRB      11$                                       ;
                   63        53     8F  9A 001D0  10$:    MOVZBL   #83, SYECOM+43                            ; 1682
                             24     A5  9F 001D4          PUSHAB   P.AAW                                     ; 1683
                             EC     A2  9F 001D7  11$:    PUSHAB   LSTLUN                                    ;
                                    52  DD 001DA          PUSHL    R2                                        ;
                   64               03  FB 001DC          CALLS    #3, EXEC_IMAGE                            ;
                                    05  11 001DF          BRB      13$                                       ; 1692
                                    52  DD 001E1  12$:    PUSHL    R2                                        ; 1693
                   64               01  FB 001E3          CALLS    #1, EXEC_IMAGE                            ;
                   50               01  D0 001E6  13$:    MOVL     #1, R0                                    ; 1694
```

```
                                        04 001E9          RET                                          ; 1701

; Routine Size:  490 bytes,    Routine Base:  $CODE + 0510

; 1145          1702  1
```

```
  1 Routine OPEN_TEXT_LIB =
  1
  1 !++
  1 ! Functional description
  1 !
  1 !           This routine set up the default library name then attempts
  1 !           to translate ERFLIB. If there is no translation then the
  1 !           default library name is used to open the text library.
  1 !           If there is a translation then that string is used instead.
  1 !           Once the library is opened, modules are read in and there
  1 !           records parsed. These records are used to build the tables
  1 !           which control device validation, device module secection,
  1 !           and CPU validation. MODULE_NAME_DESC points at the name of
  1 !           text module to be read and parsed. FUNCTION is a value which
  1 !           specifies which record parser to use for a particular text
  1 !           module.
  1 !
  1 ! Calling sequence
  1 !
  1 !           OPEN_TEXT_LIB ()
  1 !
  1 ! Input parameters
  1 !
  1 !
  1 ! Output parameters
  1 !
  1 !           None
  1 !
  1 ! Routine value
  1 !
  1 !           Worst error is returned.
  1 !
  1 !----
  1
  2 BEGIN
  2
  2 LOCAL
  2         Buff:               $BBLOCK [80],
  2         Desc:               VECTOR[2,LONG] INITIAL (80,buff),
  2         Function,
  2         Nocall_needed:  BYTE INITIAL (FALSE),
  2         Status,
  2         Text_library_name_desc,
  2         Trnlnmlst:          $itmlst_decl (items = 1);
  2
  2 Global
  2         Ident;
  2
  2 Ident = $descriptor('V03-026');
  2 Text_library_name_desc = $descriptor ('SYS$LIBRARY:ERFLIB.TLB') ;
  2
P 2 $Itmlst_init ( itmlst = trnlnmlst, (itmcod = lnm$_string, bufadr = .desc[1],
  2                                       bufsiz = .desc[0], retlen = desc[0]));
  2
P 2 If $trnlnm ( attr = %ref(lnm$m_case_blind),
P 2                 tabnam = lnm$file_dev_desc,
P 2                 lognam = erflib_desc,
3                               itmlst = trnlnmlst)
2 Then
2    Text_library_name_desc = desc;
2
2 !
2 ! Initialize text library control and open the library.
2 !
2 Status = LBR$INI_CONTROL ( Library_index, Library_func, Library_type ) ;
2 If NOT .status then Signal_stop (.status) ;
2
2 Status = LBR$OPEN ( Library_index, .Text_library_name_desc ) ;
2 If NOT .status then                              ! Could not open library.
3     Begin
3     Signal_stop (msg$_searchfail, 1, .text_library_name_desc, .status) ;
2     End;
2
2
2 !
2 ! Set to locate mode for reading records to parse.
2 !
2
2 CALL_FUNCTION ( LBR$SET_LOCATE (Library_index) );
2
2
2 !
2 ! Sequence thru the reading and parsing of the text modules.
2 !
2 Incr loop_count from 1 to 11 do
3    Begin
3    Case .loop_count from 1 to 11 of set
3      [1]: ( Function = 1; Module_name_desc = $descriptor ('MAX_CLASS_SIZE') );
3      [2]: ( Function = 1; Module_name_desc = $descriptor ('CLASS_VALUES') );
3      [3]: ( Function = 3; Module_name_desc = $descriptor ('TABLE_SIZES') );
3      [4]:
4           Begin
4           Herald[msg$w_msg_flg] = 1;          ! Message flages
4           Herald[msg$w_arg_cnt] = 3;          ! Argument count
4           Herald[msg$l_msg_id] = erf_herald;
4           Herald[msg$w_new_flg] = 1;          ! New message flages
4           Herald[msg$w_FAO_cnt] = 1;
4           Herald[msg$l_FAO_arg1] = .ident;
4           $Putmsg (msgvec = herald);
4           Function = 4;
4           Module_name_desc = $descriptor ('DEVICES') ;
3           End;
3      [5]:
4           Begin
4           Function = 2;
4           Module_name_desc = $descriptor ('TRANSLATE_ENTRY_TABLE');
4           Table_address = .translate_entry_table;
4           Table_length = .max_misc_type;
4           Item_count = 0;
3           End;
3      [6]:
4           Begin
4           Function = 2;
4           Module_name_desc = $descriptor ('CPU_TYPES');
```

```
	    Table_address = .processor_type_table;
	    Table_length = .max_cpu_types;
	    Item_count = 0;
	    End;

	[7]:
	    Begin
	    Function = 5;
	    Module_name_desc = $descriptor ('MIN_MODULE_NAMES');
	    Desc_table_address = .min_modules_desc;
	    Table_length = .max_cpu_types;
	    Item_count = 0;
	    End;
	[8]:
	    Begin
	    Function = 5;
	    Module_name_desc = $descriptor ('MAX_MODULE_NAMES');
	    Desc_table_address = .max_modules_desc;
	    Table_length = .max_cpu_types;
	    Item_count = 0;
	    End;


	[9]:
	    !
	    ! THE NEXT THREE SECTIONS MUST BE DONE IN THIS SEQUENCE.
	    ! This section loads the MIN_MAX_TABLE_SIZES table. Each
	    ! table entry specifies the number of range pairs that
	    ! exist for a particular CPU.
	    !
	    Begin
	    Function = 2;
	    Module_name_desc = $descriptor ('MIN_MAX_SIZES');
	    Table_address = .min_max_table_sizes;
	    Table_length = .max_cpu_types;
	    Item_count = 0;
	    End;

	[10]:
	    !
	    ! This section uses the contents of the MIN_MAX_TABLE_SIZES table
	    ! to determine the size of the range tables. The base address of
	    ! each range table is then saved.
	    !
	    Begin
	    Incr range_loop from 1 to .max_cpu_types do
	      Begin
	      If .min_max_table_sizes[.range_loop] NEQ 0 then
		Begin
		Max_range_table_addr[.range_loop] =
		    get_vm ( (.min_max_table_sizes[.range_loop] + 1 ) * word_size);
		Min_range_table_addr[.range_loop] =
		    get_vm ( (.min_max_table_sizes[.range_loop] + 1 ) * word_size);
		End
	      Else
		Begin
		Max_range_table_addr[.range_loop] = 0;
```

```
6                   Min_range_table_addr[.range_loop] = 0;
5                   End;
4                 End;
4
4               !
4               ! For each range table which has a non zero size, read a text library
4               ! module which will specify the min. ranges.
4               !
4               Incr range_loop from 1 to .max_cpu_types do
4                 If .min_max_table_sizes[.range_loop] NEQ 0 then
5                   Begin
5                   Function = 2;
5                   Module_name_desc = min_modules_desc[.range_loop,desc_one];
5                   Table_address = .min_range_table_addr[.range_loop];
5                   Table_length = .min_max_table_sizes[.range_loop];
5                   Item_count = 0;
5                   CALL_FUNCTION ( Get_library_text ( .Function, .Module_name_desc ));
4                   End;
4               Nocall_needed = True;
3               End;
3
3
3         [11]:
3               !
3               ! For each range table which has a non zero size, read a text library
3               ! module which will specify the max. ranges.
3               !
4               Begin
4               Incr range_loop from 1 to .max_cpu_types do
4                 If .min_max_table_sizes[.range_loop] NEQ 0 then
5                   Begin
5                   Function = 2;
5                   Module_name_desc = max_modules_desc[.range_loop,desc_one];
5                   Table_address = .max_range_table_addr[.range_loop];
5                   Table_length = .min_max_table_sizes[.range_loop];
5                   Item_count = 0;
5                   CALL_FUNCTION ( Get_library_text ( .Function, .Module_name_desc ));
4                   End;
4               Nocall_needed = True;
3               End;
3         TES;
3
3         If NOT .nocall_needed then                 ! If nocall_needed is false then
4           CALL_FUNCTION ( Get_library_text ( .Function, .Module_name_desc ))
3         Else                                       ! else its true and reset it to false.
3           Nocall_needed = false;
3
2         End;
2
2     Status = LBR$CLOSE ( Library_index ) ;
2     If NOT .status then Signal_stop (.status) ;
2
2     Return true;
1     End;
```

```
                                                                           .PSECT  $PLIT,NOWRT,NOEXE,  PIC,2

                                36  32  30  2D  33  30  56  000D8 P.AAZ:   .ASCII  \V03-026\                                      ;
                                                            000DF          .BLKB   1
                                                  00000007. 000E0 P.AAY:   .LONG   7                                              ;
                                                  00000000' 000E4          .ADDRESS P.AAZ                                         ;
46  52  45  3A  59  52  41  52  42  49  4C  24  53  59  53  000E8 P.ABB:   .ASCII  \SYS$LIBRARY:ERFLIB.TLB\                       ;
                                42  4C  54  2E  42  49  4C  000F7                                                                 ;
                                                            000FE          .BLKB   2
                                                  00000016. 00100 P.ABA:   .LONG   22                                             ;
                                                  00000000' 00104          .ADDRESS P.ABB                                         ;
    45  5A  49  53  5F  53  53  41  4C  43  5F  58  41  4D  00108 P.ABD:   .ASCII  \MAX_CLASS_SIZE\                               ;
                                                            00116          .BLKB   2
                                                  0000000E. 00118 P.ABC:   .LONG   14                                             ;
                                                  00000000' 0011C          .ADDRESS P.ABD                                         ;
            53  45  55  4C  41  56  5F  53  53  41  4C  43  00120 P.ABF:   .ASCII  \CLASS_VALUES\                                 ;
                                                  0000000C. 0012C P.ABE:   .LONG   12                                             ;
                                                  00000000' 00130          .ADDRESS P.ABF                                         ;
                53  45  5A  49  53  5F  45  4C  42  41  54  00134 P.ABH:   .ASCII  \TABLE_SIZES\                                  ;
                                                            0013F          .BLKB   1
                                                  0000000B. 00140 P.ABG:   .LONG   11                                             ;
                                                  00000000' 00144          .ADDRESS P.ABH                                         ;
                                53  45  43  49  56  45  44  00148 P.ABJ:   .ASCII  \DEVICES\                                      ;
                                                            0014F          .BLKB   1
                                                  00000007. 00150 P.ABI:   .LONG   7                                              ;
                                                  00000000' 00154          .ADDRESS P.ABJ                                         ;
59  52  54  4E  45  5F  45  54  41  4C  53  4E  41  52  54  00158 P.ABL:   .ASCII  \TRANSLATE_ENTRY_TABLE\                        ;
                                    45  4C  42  41  54  5F  00167                                                                 ;
                                                            0016D          .BLKB   3
                                                  00000015. 00170 P.ABK:   .LONG   21                                             ;
                                                  00000000' 00174          .ADDRESS P.ABL                                         ;
                        53  45  50  59  54  5F  55  50  43  00178 P.ABN:   .ASCII  \CPU_TYPES\                                    ;
                                                            00181          .BLKB   3
                                                  00000009. 00184 P.ABM:   .LONG   9                                              ;
                                                  00000000' 00188          .ADDRESS P.ABN                                         ;
45  4D  41  4E  5F  45  4C  55  44  4F  4D  5F  4E  49  4D  0018C P.ABP:   .ASCII  \MIN_MODULE_NAMES\                             ;
                                                        53  0019B                                                                 ;
                                                  00000010. 0019C P.ABO:   .LONG   16                                             ;
                                                  00000000' 001A0          .ADDRESS P.ABP                                         ;
45  4D  41  4E  5F  45  4C  55  44  4F  4D  5F  58  41  4D  001A4 P.ABR:   .ASCII  \MAX_MODULE_NAMES\                             ;
                                                        53  001B3                                                                 ;
                                                  00000010. 001B4 P.ABQ:   .LONG   16                                             ;
                                                  00000000' 001B8          .ADDRESS P.ABR                                         ;
        53  45  5A  49  53  5F  58  41  4D  5F  4E  49  4D  001BC P.ABT:   .ASCII  \MIN_MAX_SIZES\                                ;
                                                            001C9          .BLKB   3
                                                  0000000D. 001CC P.ABS:   .LONG   13                                             ;
                                                  00000000' 001D0          .ADDRESS P.ABT                                         ;

                                                                           .PSECT  $GLOBAL$,NOEXE,  PIC,2

                                                            000BC IDENT::  .BLKB   4

                                                                           .EXTRN  SYS$TRNLNM, SYS$PUTMSG

                                                                           .PSECT  $CODE,NOWRT,  PIC,2

                                                      0FFC  00000 OPEN_TEXT_LIB:
                                                           .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11              ; 1703
                    5B 00000000G   00  9E 00002            MOVAB   LIB$STOP, R11                                     ;
                    5A 00000000'   00  9E 00009            MOVAB   P.AAY, R10                                        ;
                    59 00000000'   00  9E 00010            MOVAB   MODULE_NAME_DESC, R9                              ;
                    5E         94  AE  9E 00017            MOVAB   -108(SP), SP                                      ;
              14    AE         50  8F  9A 0001B            MOVZBL  #80, DESC                                         ; 1737
              18    AE         1C  AE  9E 00020            MOVAB   BUFF, DESC+4                                      ;
                               58  94 00025                CLRB    NOCALL_NEEDED                                     ;
              54    A9         6A  9E 00027                MOVAB   P.AAY, IDENT                                      ; 1751
                    52         20  AA  9E 0002B            MOVAB   P.ABA, TEXT_LIBRARY_NAME_DESC                     ; 1752
                    50         04  AE  9E 0002F            MOVAB   TRNLNMLST, $$ITMBLKPTR                            ; 1755
                    80         14  AE  B0 00033            MOVW    DESC, ($$ITMBLKPTR)+                              ;
                    80             02  B0 00037            MOVW    #2, ($$ITMBLKPTR)+                                ;
                    80         18  AE  D0 0003A            MOVL    DESC+4, ($$ITMBLKPTR)+                            ;
                    80         14  AE  9E 0003E            MOVAB   DESC, ($$ITMBLKPTR)+                              ;
                                   80  D4 00042            CLRL    ($$ITMBLKPTR)+                                    ;
                               04  AE  9F 00044            PUSHAB  TRNLNMLST                                         ; 1760
                                   7E  D4 00047            CLRL    -(SP)                                             ;
                             FF28  CA  9F 00049            PUSHAB  ERFLIB_DESC                                       ;
                        00000000G  00  9F 0004D            PUSHAB  LNM$FILE_DEV_DESC                                 ;
              10    AE  02000000   8F  D0 00053            MOVL    #33554432, 16(SP)                                 ;
                               10  AE  9F 0005B            PUSHAB  16(SP)                                            ;
          00000000G  00            05  FB 0005E            CALLS   #5, SYS$TRNLNM                                    ;
                     04            50  E9 00065            BLBC    R0, 1$                                            ;
                     52        14  AE  9E 00068            MOVAB   DESC, TEXT_LIBRARY_NAME_DESC                      ; 1762
                               D8  A9  9F 0006C 1$:        PUSHAB  LIBRARY_TYPE                                      ; 1767
                               D0  A9  9F 0006F            PUSHAB  LIBRARY_FUNC                                      ;
                               D4  A9  9F 00072            PUSHAB  LIBRARY_INDEX                                     ;
          00000000G  00            03  FB 00075            CALLS   #3, LBR$INI_CONTROL                               ;
                     57            50  D0 0007C            MOVL    R0, STATUS                                        ;
                     05            57  E8 0007F            BLBS    STATUS, 2$                                        ; 1768
                                   57  DD 00082            PUSHL   STATUS                                            ;
                     6B            01  FB 00084            CALLS   #1, LIB$STOP                                      ;
                                   52  DD 00087 2$:        PUSHL   TEXT_LIBRARY_NAME_DESC                            ; 1770
                               D4  A9  9F 00089            PUSHAB  LIBRARY_INDEX                                     ;
          00000000G  00            02  FB 0008C            CALLS   #2, LBR$OPEN                                      ;
                     57            50  D0 00093            MOVL    R0, STATUS                                        ;
                     0F            57  E8 00096            BLBS    STATUS, 3$                                        ; 1771
                             0084  8F  BB 00099            PUSHR   #^M<R2,R7>                                        ; 1773
                                   01  DD 0009D            PUSHL   #1                                                ;
                         0008123A  8F  DD 0009F            PUSHL   #528954                                           ;
                     6B            04  FB 000A5            CALLS   #4, LIB$STOP                                      ;
                               D4  A9  9F 000A8 3$:        PUSHAB  LIBRARY_INDEX                                     ; 1781
          00000000G  00            01  FB 000AB            CALLS   #1, LBR$SET_LOCATE                                ;
                     01            50  E8 000B2            BLBS    STATUS, 4$                                        ;
                                       04 000B5            RET                                                       ;
                     55            01  D0 000B6 4$:        MOVL    #1, LOOP_COUNT                                    ; 1787
          0A         01            55  CF 000B9 5$:        CASEL   LOOP_COUNT, #1, #10                               ; 1789
0031      0028       001F      0016      000BD 6$:        .WORD   7$-6$,-                                           ;
009D      008E       007F      0065      000C5                     8$-6$,-                                           ;
          015B       00C4      00AC      000CD                     9$-6$,-                                           ;
                                                                   10$-6$,-                                          ;
                                                                   12$-6$,-                                          ;
                                                                   13$-6$,-                                          ;
                                                                   14$-6$,-                                          ;
                                                                   15$-6$,-                                          ;
                                                                   16$-6$,-                                          ;
```

```
                                                               20$-6$,-                                  ;
                                                               26$-6$                                    ;
                  53              01  D0 000D3 7$:    MOVL     #1, FUNCTION                              ; 1790
                  69        38    AA  9E 000D6        MOVAB    P.ABC, MODULE_NAME_DESC                   ;
                                  44  11 000DA        BRB      11$                                       ;
                  53              01  D0 000DC 8$:    MOVL     #1, FUNCTION                              ; 1791
                  69        4C    AA  9E 000DF        MOVAB    P.ABE, MODULE_NAME_DESC                   ;
                                  3B  11 000E3        BRB      11$                                       ;
                  53              03  D0 000E5 9$:    MOVL     #3, FUNCTION                              ; 1792
                  69        60    AA  9E 000E8        MOVAB    P.ABG, MODULE_NAME_DESC                   ;
                                  32  11 000EC        BRB      11$                                       ;
            B0    A9 00010003     8F  D0 000EE 10$:   MOVL     #65539, HERALD                            ; 1796
            B4    A9 00000000G    8F  D0 000F6        MOVL     #ERF_HERALD, HERALD+4                     ; 1797
            B8    A9 00010001     8F  D0 000FE        MOVL     #65537, HERALD+8                          ; 1798
            BC    A9        54    A9  D0 00106        MOVL     IDENT, HERALD+12                          ; 1800
                                  7E  7C 0010B        CLRQ     -(SP)                                     ; 1801
                                  7E  D4 0010D        CLRL     -(SP)                                     ;
                            B0    A9  9F 0010F        PUSHAB   HERALD                                    ;
     00000000G    00              04  FB 00112        CALLS    #4, SYS$PUTMSG                            ;
                  53              04  D0 00119        MOVL     #4, FUNCTION                              ; 1802
                  69        70    AA  9E 0011C        MOVAB    P.ABI, MODULE_NAME_DESC                   ; 1803
                                  5C  11 00120 11$:   BRB      19$                                       ; 1789
                  53              02  D0 00122 12$:   MOVL     #2, FUNCTION                              ; 1807
                  69      0090    CA  9E 00125        MOVAB    P.ABK, MODULE_NAME_DESC                   ; 1808
            28    A9 00000000G    00  D0 0012A        MOVL     TRANSLATE_ENTRY_TABLE, TABLE_ADDRESS      ; 1809
            2C    A9 00000000G    00  90 00132        MOVB     MAX_MISC_TYPE, TABLE_LENGTH               ; 1810
                                  3F  11 0013A        BRB      18$                                       ; 1811
                  53              02  D0 0013C 13$:   MOVL     #2, FUNCTION                              ; 1815
                  69      00A4    CA  9E 0013F        MOVAB    P.ABM, MODULE_NAME_DESC                   ; 1816
            28    A9        08    A9  D0 00144        MOVL     PROCESSOR_TYPE_TABLE, TABLE_ADDRESS       ; 1817
                                  2B  11 00149        BRB      17$                                       ; 1818
                  53              05  D0 0014B 14$:   MOVL     #5, FUNCTION                              ; 1824
                  69      00BC    CA  9E 0014E        MOVAB    P.ABO, MODULE_NAME_DESC                   ; 1825
            9C    A9        F0    A9  D0 00153        MOVL     MIN_MODULES_DESC, DESC_TABLE_ADDRESS      ; 1826
                                  1C  11 00158        BRB      17$                                       ; 1827
                  53              05  D0 0015A 15$:   MOVL     #5, FUNCTION                              ; 1832
                  69      00D4    CA  9E 0015D        MOVAB    P.ABQ, MODULE_NAME_DESC                   ; 1833
            9C    A9        F4    A9  D0 00162        MOVL     MAX_MODULES_DESC, DESC_TABLE_ADDRESS      ; 1834
                                  0D  11 00167        BRB      17$                                       ; 1835
                  53              02  D0 00169 16$:   MOVL     #2, FUNCTION                              ; 1848
                  69      00EC    CA  9E 0016C        MOVAB    P.ABS, MODULE_NAME_DESC                   ; 1849
            28    A9        F8    A9  D0 00171        MOVL     MIN_MAX_TABLE_SIZES, TABLE_ADDRESS        ; 1850
            2C    A9        E2    A9  90 00176 17$:   MOVB     MAX_CPU_TYPES, TABLE_LENGTH               ; 1851
                            CC    A9  D4 0017B 18$:   CLRL     ITEM_COUNT                                ; 1852
                          00D9    31  00 17E   19$:   BRW      30$                                       ; 1789
                  56        E2    A9  3C 00181 20$:   MOVZWL   MAX_CPU_TYPES, R6                         ; 1862
                                  52  D4 00185        CLRL     RANGE_LOOP                                ;
                                  4A  11 00187        BRB      23$                                       ;
                  54        E8    A9  D0 00189 21$:   MOVL     MAX_RANGE_TABLE_ADDR, R4                  ; 1866
                  50        F8    A9  D0 0018D        MOVL     MIN_MAX_TABLE_SIZES, R0                   ; 1864
                  50      6042    3C  00191           MOVZWL   (R0)[RANGE_LOOP], R0                      ;
                                  32  13 00195        BEQL     22$                                       ;
           7E     50              01  78 00197        ASHL     #1, R0, -(SP)                             ; 1867
                  6E              02  C0 0019B        ADDL2    #2, (SP)                                  ;
     00000000G    00              01  FB 0019E        CALLS    #1, GET_VM                                ;
                6442              50  D0 001A5        MOVL     R0, (R4)[RANGE_LOOP]                      ;
                  54        FC    A9  D0 001A9        MOVL     MIN_RANGE_TABLE_ADDR, R4                  ; 1868
```

```
                         50      F8    A9  D0 001AD        MOVL    MIN_MAX_TABLE_SIZES, R0                              ; 1869
                         50          6042  3C 001B1        MOVZWL  (R0)[RANGE_LOOP], R0                                 ;
              7E         50            01  78 001B5        ASHL    #1, R0, -(SP)                                        ;
                         6E            02  C0 001B9        ADDL2   #2, (SP)                                             ;
                  00000000G 00         01  FB 001BC        CALLS   #1, GET_VM                                           ;
                       6442            50  D0 001C3        MOVL    R0, (R4)[RANGE_LOOP]                                 ;
                                       0A  11 001C7        BRB     23$                                                  ; 1864
                                     6442  D4 001C9 22$:   CLRL    (R4)[RANGE_LOOP]                                     ; 1873
                         50      FC    A9  D0 001CC        MOVL    MIN_RANGE_TABLE_ADDR, R0                             ; 1874
                                     6042  D4 001D0        CLRL    (R0)[RANGE_LOOP]                                     ;
              B2         52            56  F3 001D3 23$:   AOBLEQ  R6, RANGE_LOOP, 21$                                  ; 1862
                         52      E2    A9  3C 001D7        MOVZWL  MAX_CPU_TYPES, R2                                    ; 1882
                                       54  D4 001DB        CLRL    RANGE_LOOP                                           ;
                                       33  11 001DD        BRB     25$                                                  ;
                         51      F8    A9  D0 001DF 24$:   MOVL    MIN_MAX_TABLE_SIZES, R1                              ; 1883
                                     6144  B5 001E3        TSTW    (R1)[RANGE_LOOP]                                     ;
                                       2A  13 001E6        BEQL    25$                                                  ;
                         53            02  D0 001E8        MOVL    #2, FUNCTION                                         ; 1885
                         50      F0    A9  D0 001EB        MOVL    MIN_MODULES_DESC, R0                                 ; 1886
                         69          6044  7E 001EF        MOVAQ   (R0)[RANGE_LOOP], MODULE_NAME_DESC                   ;
                         50      FC    A9  D0 001F3        MOVL    MIN_RANGE_TABLE_ADDR, R0                             ; 1887
                   28    A9          6044  D0 001F7        MOVL    (R0)[RANGE_LOOP], TABLE_ADDRESS                      ;
                   2C    A9          6144  33 001FC        CVTWB   (R1)[RANGE_LOOP], TABLE_LENGTH                       ; 1888
                                 CC    A9  D4 00201        CLRL    ITEM_COUNT                                           ; 1889
                                       69  DD 00204        PUSHL   MODULE_NAME_DESC                                     ; 1890
                                       53  DD 00206        PUSHL   FUNCTION                                             ;
                  00000000V 00         02  FB 00208        CALLS   #2, GET_LIBRARY_TEXT                                 ;
                            7A         50  E9 0020F        BLBC    STATUS, 34$                                          ;
              C9         54            52  F3 00212 25$:   AOBLEQ  R2, RANGE_LOOP, 24$                                  ; 1883
                                       3F  11 00216        BRB     29$                                                  ; 1892
                         52      E2    A9  3C 00218 26$:   MOVZWL  MAX_CPU_TYPES, R2                                    ; 1902
                                       54  D4 0021C        CLRL    RANGE_LOOP                                           ;
                                       33  11 0021E        BRB     28$                                                  ;
                         51      F8    A9  D0 00220 27$:   MOVL    MIN_MAX_TABLE_SIZES, R1                              ; 1903
                                     6144  B5 00224        TSTW    (R1)[RANGE_LOOP]                                     ;
                                       2A  13 00227        BEQL    28$                                                  ;
                         53            02  D0 00229        MOVL    #2, FUNCTION                                         ; 1905
                         50      F4    A9  D0 0022C        MOVL    MAX_MODULES_DESC, R0                                 ; 1906
                         69          6044  7E 00230        MOVAQ   (R0)[RANGE_LOOP], MODULE_NAME_DESC                   ;
                         50      E8    A9  D0 00234        MOVL    MAX_RANGE_TABLE_ADDR, R0                             ; 1907
                   28    A9          6044  D0 00238        MOVL    (R0)[RANGE_LOOP], TABLE_ADDRESS                      ;
                   2C    A9          6144  33 0023D        CVTWB   (R1)[RANGE_LOOP], TABLE_LENGTH                       ; 1908
                                 CC    A9  D4 00242        CLRL    ITEM_COUNT                                           ; 1909
                                       69  DD 00245        PUSHL   MODULE_NAME_DESC                                     ; 1910
                                       53  DD 00247        PUSHL   FUNCTION                                             ;
                  00000000V 00         02  FB 00249        CALLS   #2, GET_LIBRARY_TEXT                                 ;
                            39         50  E9 00250        BLBC    STATUS, 34$                                          ;
              C9         54            52  F3 00253 28$:   AOBLEQ  R2, RANGE_LOOP, 27$                                  ; 1903
                         58            01  90 00257 29$:   MOVB    #1, NOCALL_NEEDED                                    ; 1912
                         0F            58  E8 0025A 30$:   BLBS    NOCALL_NEEDED, 31$                                   ; 1916
                                       69  DD 0025D        PUSHL   MODULE_NAME_DESC                                     ; 1917
                                       53  DD 0025F        PUSHL   FUNCTION                                             ;
                  00000000V 00         02  FB 00261        CALLS   #2, GET_LIBRARY_TEXT                                 ;
                            03         50  E8 00268        BLBS    STATUS, 32$                                          ;
                                           04 0026B        RET                                                          ;
                                       58  94 0026C 31$:   CLRB    NOCALL_NEEDED                                        ; 1919
       FE45           55         01    0B  F1 0026E 32$:   ACBL    #11, #1, LOOP_COUNT, 5$                              ; 1787
```

```
                                  D4    A9  9F 00274        PUSHAB  LIBRARY_INDEX                        ; 1923
                  00000000G  00         01  FB 00277        CALLS   #1, LBR$CLOSE                        ;
                             57         50  D0 0027E        MOVL    R0, STATUS                           ;
                             05         57  E8 00281        BLBS    STATUS, 33$                          ; 1924
                                        57  DD 00284        PUSHL   STATUS                               ;
                             6B         01  FB 00286        CALLS   #1, LIB$STOP                         ;
                             50         01  D0 00289 33$:   MOVL    #1, R0                               ; 1926
                                            04 0028C 34$:   RET                                          ; 1927

; Routine Size:  653 bytes,    Routine Base:  $CODE + 06FA
```

```
1 Routine GET_LIBRARY_TEXT ( Function, module_name ) =
2 BEGIN
2 !++
2 ! Functional description
2 !
2 !       This routine looks up the text module name specified. It
2 !       reads the records from that text module. If a record
2 !       does not have a comment character in the first three
2 !       character positions and the record is not of length zero,
2 !       then the parsing routine specified by FUNCTION is called.
2 !
2 ! Calling sequence
2 !
2 !       GET_LIBRARY_TEXT ( Function, module_name )
2 !
2 ! Input parameters
2 !
2 !               Function : Value specifying;
2 !                       1 Build class tables
2 !                       2 Parse and convert to binary a list of values
2 !                       3 Allocate and initialize processor and device tables
2 !                       4 Parse device description records. See text module
2 !                          DEVICES for more information.
2 !
2 !             Module_name : Address of descriptor for module name.
2 !
2 ! Output parameters
2 !
2 !       None
2 !
2 ! Routine value
2 !
2 !       Worst error is returned.
2 !
2 !----
2
2 LOCAL
2       Offset,
2       Position,
2       Status;
2
2 !
2 ! Use MODULE_NAME as the key to find the text module in library.
2 !
2
2 Status = LBR$LOOKUP_KEY ( Library_index, .Module_name, Text_rfa ) ;
2 If NOT .status then Signal (erf_badmodnam, 1, .module_name, .status) ;
2
2
2 !
2 !READ A RECORD FROM THE TEXT LIBRARY
2 ! If the record length is not zero then case to a decode routine.
2 ! Search the record for the comment character '!'.
2 ! If "!" is in one of the first three positions get a new record.
2 !
2
2 While Status = LBR$GET_RECORD ( Library_index, Record_desc, Record_desc) do
```

```
; 1430          1985  3        Begin
; 1431          1986  3        If .Record_desc [dsc$w_length] NEQ 0 then
; 1432          1987  4          Begin
; 1433          1988  4          Position = CH$FIND_CH (.Record_desc [dsc$w_length],
; 1434          1989  4                                 .Record_desc [dsc$a_pointer], %c'!' );
; 1435          1990  4          If .position NEQ 0 then
; 1436          1991  4            Offset = CH$DIFF ( .position , .Record_desc [dsc$a_pointer])
; 1437          1992  4          Else
; 1438          1993  4            Offset = 4;
; 1439          1994  4
; 1440          1995  4          If .Offset GTR 3 then
; 1441          1996  4            Case .function from 1 to 5 of set
; 1442          1997  4              [1]: Build_class_tables ();           ! Inits device class table
; 1443          1998  4              [2]: Parse_max_min_table_record ();   ! Min max table parser
; 1444          1999  4              [3]: Parse_max_table_size ();         ! Parse & convert to binary
; 1445          2000  4              [4]: Parse_device_desc_record ();     ! Parse & return strings
; 1446          2001  4              [5]: Parse_module_names ();
; 1447          2002  4            Tes;
; 1448          2003  3          End;
; 1449          2004  2        End;
; 1450          2005  2
; 1451          2006  2  Item_count = 0;                                  ! Global used by several routines
; 1452          2007  2                                                   !  as there array index.
; 1453          2008  2  Return true ;
; 1454          2009  1  End ;



                                             007C 00000 GET_LIBRARY_TEXT:
                                                               .WORD   Save R2,R3,R4,R5,R6                          ; 1928
                   56 00000000'      00   9E 00002             MOVAB   RECORD_DESC, R6
                                20   A6   9F 00009             PUSHAB  TEXT_RFA                                     ; 1973
                                08   AC   DD 0000C             PUSHL   MODULE_NAME                                  ;
                                C4   A6   9F 0000F             PUSHAB  LIBRARY_INDEX                                ;
              00000000G   00         03   FB 00012             CALLS   #3, LBR$LOOKUP_KEY                           ;
                          55         50   D0 00019             MOVL    R0, STATUS                                   ;
                          14         55   E8 0001C             BLBS    STATUS, 1$                                   ; 1974
                                     55   DD 0001F             PUSHL   STATUS                                       ;
                                08   AC   DD 00021             PUSHL   MODULE_NAME                                  ;
                                     01   DD 00024             PUSHL   #1                                           ;
                          00000000G  8F   DD 00026             PUSHL   #ERF_BADMODNAM                               ;
              00000000G   00         04   FB 0002C             CALLS   #4, LIB$SIGNAL                               ;
                                     56   DD 00033 1$:         PUSHL   R6                                           ; 1984
                                     56   DD 00035             PUSHL   R6                                           ;
                                C4   A6   9F 00037             PUSHAB  LIBRARY_INDEX                                ;
              00000000G   00         03   FB 0003A             CALLS   #3, LBR$GET_RECORD                           ;
                          55         50   D0 00041             MOVL    R0, STATUS                                   ;
                          60         55   E9 00044             BLBC    STATUS, 11$                                  ;
                          50         66   3C 00047             MOVZWL  RECORD_DESC, R0                              ; 1986
                                     E7   13 0004A             BEQL    1$                                           ;
                          52    04   A6   D0 0004C             MOVL    RECORD_DESC+4, R2                            ; 1989
                 62       50         21   3A 00050             LOCC    #33, R0, (R2)                                ; 1988
                                     02   12 00054             BNEQ    2$                                           ;
                                     51   D4 00056             CLRL    R1                                           ;
                          54         51   D0 00058 2$:         MOVL    R1, POSITION                                 ;
```

```
                                              06  13 0005B        BEQL    3$                                                ; 1990
              53             54               52  C3 0005D        SUBL3   R2, POSITION, OFFSET                              ; 1991
                                              03  11 00061        BRB     4$                                                ;
                             53               04  D0 00063 3$:    MOVL    #4, OFFSET                                        ; 1993
                             03               53  D1 00066 4$:    CMPL    OFFSET, #3                                        ; 1995
                                              C8  15 00069        BLEQ    1$                                                ;
              04             01         04    AC  CF 0006B        CASEL   FUNCTION, #1, #4                                  ; 1996
     0025   001C           0013             000A     00070 5$:    .WORD   6$-5$,-                                           ;
                                            002E     00078                7$-5$,-                                           ;
                                                                          8$-5$,-                                           ;
                                                                          9$-5$,-                                           ;
                                                                          10$-5$                                            ;
                      00000000V  00               00  FB 0007A 6$:    CALLS   #0, BUILD_CLASS_TABLES                        ; 1997
                                              B0  11 00081        BRB     1$                                                ;
                      00000000V  00               00  FB 00083 7$:    CALLS   #0, PARSE_MAX_MIN_TABLE_RECORD                ; 1998
                                              A7  11 0008A        BRB     1$                                                ;
                      00000000V  00               00  FB 0008C 8$:    CALLS   #0, PARSE_MAX_TABLE_SIZE                      ; 1999
                                              9E  11 00093        BRB     1$                                                ;
                      00000000V  00               00  FB 00095 9$:    CALLS   #0, PARSE_DEVICE_DESC_RECORD                  ; 2000
                                              95  11 0009C        BRB     1$                                                ;
                      00000000V  00               00  FB 0009E 10$:   CALLS   #0, PARSE_MODULE_NAMES                        ; 2001
                                              8C  11 000A5        BRB     1$                                                ; 1996
                                        BC    A6  D4 000A7 11$:   CLRL    ITEM_COUNT                                        ; 2006
                             50               01  D0 000AA        MOVL    #1, R0                                            ; 2008
                                                  04 000AD        RET                                                       ; 2009

; Routine Size:  174 bytes,    Routine Base:  $CODE + 0987


; 1455         2010  1
```

```
1 Routine BUILD_CLASS_TABLES =
2 BEGIN
2 !++
2 ! Functional description
2 !
2 !       This routine allocates memory for tables which will be filled
2 !       with the information obtained from the ERF text library in
2 !       SYS$LIBRARY. See the text library modules for a description
2 !       of the text library records.
2 !
2 ! Calling sequence
2 !
2 !      Build_class_tables ()
2 !
2 ! Input parameters
2 !
2 !
2 ! Output parameters
2 !
2 !       Class_dir, class_names, dev_addrs_ptr, dev_class_ptr
2 !
2 ! Routine value
2 !
2 !       Worst error is returned.
2 !
2 !----
2
2 OWN
2         Context,                  ! Continuation flag if this flag is set then more
2                                   !   values in comma seperated list
2         Size,                     ! Length of the field pointed to by VALUE_ADDR
2         Status,                   ! Status after a convert
2         Index,                    ! The first field of a text record
2         Value_addr;               ! Pointer to current filed in the text record
2
2 !LOCAL
2 !       Class_names;              ! Address of DC$_xxx strings
2 !Class_names = Get_vm ((.size+1) * 13); ! Class name + string size = 13
```

```
; 1496          2049  2 !
; 1497          2050  2 ! Call to obtain the index and the current value address (value_addr).
; 1498          2051  2 !
; 1499          2052  2
; 1500          2053  2 Context = 0;
; 1501          2054  2 CALL_FUNCTION ( Parse_text_record ( context, index, value_addr, size ) );
; 1502          2055  2
; 1503          2056  2
; 1504          2057  2
; 1505          2058  2 !
; 1506          2059  2 ! Convert the current value from ASCII to binary and use it to allocate
; 1507          2060  2 ! memory for tables. Use the index value to determin which table to build.
; 1508          2061  2 !
; 1509          2062  2 Status = LIB$CVT_DTB ( .size, .Value_addr, size );
; 1510          2063  2 If NOT .status then Signal (erf_cvterr, 2,.size,value_addr) ;
; 1511          2064  2
; 1512          2065  2 If .class_dir EQL 0 then
; 1513          2066  3  Begin
; 1514          2067  3  Class_dir = Get_vm ((.size+1) * 6);     ! Device Class(2 bytes + device name table addr (longword)= 6
; 1515          2068  3  Max_classes = .size;                   ! Total number of dev. classes
; 1516          2069  3  Dev_addrs_ptr = Class_dir[.max_classes+1];
; 1517          2070  3  Dev_class_ptr = .class_dir ;
; 1518          2071  3  End
; 1519          2072  2 Else
; 1520          2073  3  Begin
; 1521          2074  3  If .index GTR .max_classes then signal (erf_toomancls, 1, .index);
; 1522          2075  3  Class_dir[.index] = .size; ! Set device class values
; 1523          2076  2  End;
; 1524          2077  2
; 1525          2078  2 Return true ;
; 1526          2079  1 End ;


                                                             .PSECT  $OWN$,NOEXE,  PIC,2

                                            0001C CONTEXT:.BLKB   4
                                            00020 SIZE:   .BLKB   4
                                            00024 STATUS: .BLKB   4
                                            00028 INDEX:  .BLKB   4
                                            0002C VALUE_ADDR:
                                                          .BLKB   4


                                                             .PSECT  $CODE,NOWRT,  PIC,2

                                       003C 00000 BUILD_CLASS_TABLES:
                                                          .WORD   Save R2,R3,R4,R5                       ; 2011
                    55 00000000G   00   9E 00002          MOVAB   LIB$SIGNAL, R5                         ;
                    54 00000000G   00   9E 00009          MOVAB   CLASS_DIR, R4                          ;
                    53 00000000'   00   9E 00010          MOVAB   MAX_CLASSES, R3                        ;
                    52 00000000'   00   9E 00017          MOVAB   SIZE, R2                               ;
                             FC    A2   D4 0001E          CLRL    CONTEXT                                ; 2053
                                   52   DD 00021          PUSHL   R2                                     ; 2054
                             0C    A2   9F 00023          PUSHAB  VALUE_ADDR                             ;
                             08    A2   9F 00026          PUSHAB  INDEX                                  ;
```

```
                                                                         F  6
ERF               Errorlog Report Formatter                       15-Sep-1984 23:42:14    VAX-11 Bliss-32 V4.0-742              Page  54
V04-000                                                           14-Sep-1984 12:27:17    DISK$VMSMASTER:[ERF.SRC]ERF.B32;1          (16)

                                          FC   A2  9F 00029          PUSHAB  CONTEXT                                    ;
                         00000000V  00         04  FB 0002C          CALLS   #4, PARSE_TEXT_RECORD                      ;
                                    76         50  E9 00033          BLBC    STATUS, 5$                                 ;
                                               52  DD 00036          PUSHL   R2                                         ; 2062
                                          0C   A2  DD 00038          PUSHL   VALUE_ADDR                                 ;
                                               62  DD 0003B          PUSHL   SIZE                                       ;
                         00000000G  00         03  FB 0003D          CALLS   #3, LIB$CVT_DTB                            ;
                                04  A2         50  D0 00044          MOVL    R0, STATUS                                 ;
                                    10    04   A2  E8 00048          BLBS    STATUS, 1$                                 ; 2063
                                          0C   A2  9F 0004C          PUSHAB  VALUE_ADDR                                 ;
                                               62  DD 0004F          PUSHL   SIZE                                       ;
                                               02  DD 00051          PUSHL   #2                                         ;
                                   00000000G   8F  DD 00053          PUSHL   #ERF_CVTERR                                ;
                                    65         04  FB 00059          CALLS   #4, LIB$SIGNAL                             ;
                                               64  D5 0005C  1$:     TSTL    CLASS_DIR                                  ; 2065
                                               26  12 0005E          BNEQ    2$                                         ;
                     50             62         06  C5 00060          MULL3   #6, SIZE, R0                               ; 2067
                                          06   A0  9F 00064          PUSHAB  6(R0)                                      ;
                         00000000G  00         01  FB 00067          CALLS   #1, GET_VM                                 ;
                                    64         50  D0 0006E          MOVL    R0, CLASS_DIR                              ;
                                    63         62  90 00071          MOVB    SIZE, MAX_CLASSES                          ; 2068
                                    51         64  D0 00074          MOVL    CLASS_DIR, R1                              ; 2069
                                    50         63  9A 00077          MOVZBL  MAX_CLASSES, R0                            ;
                                BF  A3    02 A140  3E 0007A          MOVAW   2(R1)[R0], DEV_ADDRS_PTR                   ;
                                C3  A3         51  D0 00080          MOVL    R1, DEV_CLASS_PTR                          ; 2070
                                               23  11 00084          BRB     4$                                         ; 2065
                                    50    08   A2  D0 00086  2$:     MOVL    INDEX, R0                                  ; 2074
         50          63                   08   00  ED 0008A          CMPZV   #0, #8, MAX_CLASSES, R0                    ;
                                               0D  18 0008F          BGEQ    3$                                         ;
                                               50  DD 00091          PUSHL   R0                                         ;
                                               01  DD 00093          PUSHL   #1                                         ;
                                   00000000G   8F  DD 00095          PUSHL   #ERF_TOOMANCLS                             ;
                                    65         03  FB 0009B          CALLS   #3, LIB$SIGNAL                             ;
                                    51         64  D0 0009E  3$:     MOVL    CLASS_DIR, R1                              ; 2075
                                    50    08   A2  D0 000A1          MOVL    INDEX, R0                                  ;
                                  6140         62  B0 000A5          MOVW    SIZE, (R1)[R0]                             ;
                                    50         01  D0 000A9  4$:     MOVL    #1, R0                                     ; 2078
                                                   04 000AC  5$:     RET                                                ; 2079

; Routine Size:  173 bytes,    Routine Base:  $CODE + 0A35
```

```
1 Routine PARSE_MAX_MIN_TABLE_RECORD =
2 BEGIN
2 !++
2 ! Functional description
2 !
2 !       This routine calls parse_text_record to obtain a value from
2 !       the comma seperated list of values. The value is converted
2 !       to binary and place in a table.
2 !
2 ! Calling sequence
2 !
2 ! Input parameters
2 !
2 !
2 ! Output parameters
2 !
2 !       Fills the table specified by TABLE_ADDRESS.
2 !
2 ! Routine value
2 !
2 !       Worst error is returned.
2 !
2 !----
2 Local
2       Context,
2       Index,
2       Size,
2       Status,
2       Value_addr;
2
```

```
; 1559          2110  2 Context = 0;      ! Clear the context
; 1560          2111  2
; 1561          2112  2 Do
; 1562          2113  3  Begin
; 1563          2114  3  Item_count = .item_count + 1;
; 1564          2115  3
; 1565          2116  3  If .item_count GTR .table_length then
; 1566          2117  3     (signal (erf_badevtyp, 2, .item_count, .Module_name_desc); Return true);
; 1567          2118  3
; 1568          2119  3  CALL_FUNCTION ( Parse_text_record ( context, index, value_addr, size ) );
; 1569          2120  3
; 1570          2121  3
; 1571          2122  3  Status = LIB$CVT_DTB ( .size, .Value_addr, size );
; 1572          2123  3  If NOT .status  then Signal (erf_cvterr, 2,.size,value_addr) ;
; 1573          2124  3
; 1574          2125  3  Table_address [.Item_count] = .size;
; 1575          2126  3
; 1576          2127  3  End
; 1577          2128  3
; 1578          2129  2 While .context EQL 1;
; 1579          2130  2
; 1580          2131  2 Return true ;
; 1581          2132  1 End ;




                                                      001C 00000 PARSE_MAX_MIN_TABLE_RECORD:
                                                                             .WORD   Save R2,R3,R4                                      ; 2080
                                  54 00000000G   00   9E 00002               MOVAB   LIB$SIGNAL, R4                                     ;
                                  53 00000000'   00   9E 00009               MOVAB   ITEM_COUNT, R3                                     ;
                                  5E             10   C2 00010               SUBL2   #16, SP                                            ;
                                           04    AE   D4 00013               CLRL    CONTEXT                                            ; 2110
                                                 63   D6 00016 1$:           INCL    ITEM_COUNT                                         ; 2114
                                  50             63   D0 00018               MOVL    ITEM_COUNT, R0                                     ; 2116
           50        60    A3     08             00   ED 0001B               CMPZV   #0, #8, TABLE_LENGTH, R0                           ;
                                                 12   18 00021               BGEQ    2$                                                 ;
                                           34    A3   DD 00023               PUSHL   MODULE_NAME_DESC                                   ; 2117
                                                 50   DD 00026               PUSHL   R0                                                 ;
                                                 02   DD 00028               PUSHL   #2                                                 ;
                                     00000000G   8F   DD 0002A               PUSHL   #ERF_BADEVTYP                                      ;
                                  64             04   FB 00030               CALLS   #4, LIB$SIGNAL                                     ;
                                                 4F   11 00033               BRB     4$                                                 ;
                                           08    AE   9F 00035 2$:           PUSHAB  SIZE                                               ; 2119
                                           10    AE   9F 00038               PUSHAB  VALUE_ADDR                                         ;
                                           08    AE   9F 0003B               PUSHAB  INDEX                                              ;
                                           10    AE   9F 0003E               PUSHAB  CONTEXT                                            ;
                        00000000V 00             04   FB 00041               CALLS   #4, PARSE_TEXT_RECORD                              ;
                                  3C             50   E9 00048               BLBC    STATUS, 5$                                         ;
                                           08    AE   9F 0004B               PUSHAB  SIZE                                               ; 2122
                                           10    AE   DD 0004E               PUSHL   VALUE_ADDR                                         ;
                                           10    AE   DD 00051               PUSHL   SIZE                                               ;
                        00000000G 00             03   FB 00054               CALLS   #3, LIB$CVT_DTB                                    ;
                                  52             50   D0 0005B               MOVL    R0, STATUS                                         ;
                                  11             52   E8 0005E               BLBS    STATUS, 3$                                         ; 2123
                                           0C    AE   9F 00061               PUSHAB  VALUE_ADDR                                         ;
                        0C   AE  DD 00064          PUSHL   SIZE                                              ;
                             02  DD 00067          PUSHL   #2                                                ;
                  00000000G  8F  DD 00069          PUSHL   #ERF_CVTERR                                       ;
          64                 04  FB 0006F          CALLS   #4, LIB$SIGNAL                                    ;
          51            5C   A3  D0 00072 3$:      MOVL    TABLE_ADDRESS, R1                                 ; 2125
          50                 63  D0 00076          MOVL    ITEM_COUNT, R0                                    ;
        6140            08   AE  B0 00079          MOVW    SIZE, (R1)[R0]                                    ;
          01            04   AE  D1 0007E          CMPL    CONTEXT, #1                                       ; 2129
                             92  13 00082          BEQL    1$                                                ;
          50                 01  D0 00084 4$:      MOVL    #1, R0                                            ; 2131
                                 04 00087 5$:      RET                                                       ; 2132

; Routine Size:  136 bytes,    Routine Base:  $CODE + 0AE2
```

```
1 Routine PARSE_MAX_TABLE_SIZE =
2 BEGIN
2 !++
2 ! Functional description
2 !
2 !       With the information that is returned from a call to
2 !       'PARSE_TEXT_RECORD', this routine allocates storage.
2 !       The index value of the library record being parsed
2 !       determines which table pointers are initialized.
2 !       It also sets up the table of addresses pointed to
2 !       by 'DEV_ADDR_PTR'.
2 !
2 ! Calling sequence
2 !
2 ! Input parameters
2 !
2 !       None.
2 !
2 ! Output parameters
2 !
2 !       All the table pointers for devices and cpu tables
2 !       are set up here and are global.
2 !
2 ! Routine value
2 !
2 !       Worst error is returned.
2 !
2 !----
2
2 OWN
2       Dc_class: word,         ! Temp for device class
2       Device_addr,            ! Temp for device table address
2       Context,                ! Flag which specifies if there are more items in the text record
2       Image_addr,
2       Index,                  ! Value of the first item in the text record
2       Size,                   ! Size of the item returned.
2       Status,                 ! Status of the LIB$ call
2       Value_addr,             ! Address of the current value in the text record
2       Version_addr,
2       Xfer_addr;
2
```

```
2  !
2  ! Call to obtain the index and the current value address (value_addr).
2  !
2  Context = 0;
2  CALL_FUNCTION ( Parse_text_record ( context, index, value_addr, size ) );
2
2  !
2  ! Convert the current value from ASCII to binary and use it to allocate
2  ! memory for tables. Use the index value to determin which table to build.
2  !
2  Status = LIB$CVT_DTB ( .size, .Value_addr, size );
2  If NOT .status then Signal (erf_cvterr, 2,.size,value_addr) ;
2
2  !
2  ! Update the size for non-used first locations in tables.
2  !
2  Size = .size + 1 ;
2
2  !
2  ! If INDEX is equal to one then allocate storage for the
2  ! CPU and verfification table and save the table addresses.
2  !
2  If .index EQL 1 then
3    Begin
3    Local Amount;
3    Amount = .size * longword;
3    Max_cpu_types = .size -1;
3    Min_range_table_addr = get_vm (.amount);
3    Max_range_table_addr = get_vm (.amount);
3    Min_modules_desc = get_vm (.amount*2);
3    Max_modules_desc = get_vm (.amount*2);
3    Processor_type_table = get_vm (.size * word_size);
3    Min_max_table_sizes = get_vm (.size * word_size);
3    Return true;
3    End;
2
2  !
2  ! Allocate the storage for the device, version number, xfer address,
2  ! and image name tables.
2  !
2  Device_addr = get_vm (.size * word_size);
2  Version_addr = get_vm (.size * word_size);
2  Xfer_addr = get_vm (.size * longword);
2  Image_addr = get_vm (.size * descriptor_length);
2
2
2  !
2  ! Via the index determine which entry is being processed and
2  ! copy the addresses/size of the tables to the appropriate places.
2  !
2  Case .index from 2 to 9 of set
3      [2]: Begin
3           Disk_devices = .device_addr;
3           Disk_version = .version_addr;
3           Disk_xfer_addr = .xfer_addr;
3           Disk_image = .image_addr;
3           Max_disk_type = .size - 1;
```

```
3               Disk_devices[0] = .size;
3               Dc_class = DC$_DISK;
2               End;
2
3       [3]:    Begin
3               Tape_devices = .device_addr;
3               Tape_version = .version_addr;
3               Tape_xfer_addr = .xfer_addr;
3               Tape_image = .image_addr;
3               Max_tape_type = .size - 1;
3               Tape_devices[0] = .size;
3               Dc_class = DC$_TAPE;
2               End;
2
3       [4]:    Begin
3               Scom_devices = .device_addr;
3               Scom_version = .version_addr;
3               Scom_xfer_addr = .xfer_addr;
3               Scom_image = .image_addr;
3               Max_scom_type = .size - 1;
3               Scom_devices[0] = .size;
3               Dc_class = DC$_SCOM;
2               End;
2
3       [5]:    Begin
3               Lp_devices = .device_addr;
3               Lp_version = .version_addr;
3               Lp_xfer_addr = .xfer_addr;
3               Lp_image = .image_addr;
3               Max_lp_type = .size - 1;
3               Lp_devices[0] = .size;
3               Dc_class = DC$_LP;
2               End;
2
3       [6]:    Begin
3               Realtime_devices = .device_addr;
3               Realtime_version = .version_addr;
3               Realtime_xfer_addr = .xfer_addr;
3               Realtime_image = .image_addr;
3               Max_realtime_type = .size - 1;
3               Realtime_devices[0] = .size;
3               Dc_class = DC$_REALTIME;
2               End;
2
3       [7]:    Begin
3               Bus_devices = .device_addr;
3               Bus_version = .version_addr;
3               Bus_xfer_addr = .xfer_addr;
3               Bus_image = .image_addr;
3               Max_bus_type = .size - 1;
3               Bus_devices[0] = .size;
3               Dc_class = DC$_BUS;
2               End;
2
3       [8]:    Begin
3               Packet_processor_devices = .device_addr;
3               Packet_processor_version = .version_addr;
```

```
3               Packet_processor_xfer_addr = .xfer_addr;
3               Packet_processor_image = .image_addr;
3               Max_misc_type = .size - 1;
3               Packet_processor_devices[0] = .size;
3               Translate_entry_table = get_vm (.size * word_size);
3               Dc_class = DC$_ZERO_CLASS;
2               End;
2
3       [9]: Begin
3               Workstation_devices = .device_addr;
3               Workstation_version = .version_addr;
3               Workstation_xfer_addr = .xfer_addr;
3               Workstation_image = .image_addr;
3               Max_Workstation_type = .size - 1;
3               Workstation_devices[0] = .size;
3               Dc_class = DC$_WORKSTATION;
2               End;
2
3       [OUTRANGE]:  Begin
3                    Signal (erf_badevval, 1, .index, .module_name_desc) ;
3                    Return true ;
2                    End;
2
2       TES;
2
2  !
2  ! Fill in the device class address of the 'class_dir' table. It
2  ! contains the pointers to the device class specific tables (devices,
2  ! version number, xfer address, and image name).
2  !
2  Incr count from 1 to .max_classes do
3   Begin
3   If .dev_class_ptr[.count] EQL .dc_class then    ! Make sure its the right slot
4      Begin                                        !  for the address.
4      Dev_addrs_ptr[.count] = .device_addr;        ! Save the address of the
4      Return true ;                                !  device name tables.
3      End;
2   End;
2
2  Signal (erf_clstblerr, 1,.dc_class) ;
2  Return true;
1  End ;


                                        .PSECT  $OWN$,NOEXE, PIC,2

                        00030 DC_CLASS:
                                        .BLKB   2
                        00032           .BLKB   2
                        00034 DEVICE_ADDR:
                                        .BLKB   4
                        00038 CONTEXT:.BLKB     4
                        0003C IMAGE_ADDR:
                                        .BLKB   4
                        00040 INDEX:    .BLKB   4
                        00044 SIZE:     .BLKB   4
```

```
                                                  00048 STATUS:  .BLKB   4
                                                  0004C VALUE_ADDR:
                                                                 .BLKB   4
                                                  00050 VERSION_ADDR:
                                                                 .BLKB   4
                                                  00054 XFER_ADDR:
                                                                 .BLKB   4


                                                                 .PSECT  $CODE,NOWRT,  PIC,2

                                            00FC 00000 PARSE_MAX_TABLE_SIZE:
                                                                 .WORD   Save R2,R3,R4,R5,R6,R7                    ; 2133
                          57 00000000G  00   9E 00002            MOVAB   LIB$SIGNAL, R7                            ;
                          56 00000000G  00   9E 00009            MOVAB   GET_VM, R6                                ;
                          55 00000000'  00   9E 00010            MOVAB   DISK_DEVICES, R5                          ;
                          54 00000000'  00   9E 00017            MOVAB   SIZE, R4                                  ;
                                  F4    A4   D4 0001E            CLRL    CONTEXT                                   ; 2177
                                        54   DD 00021            PUSHL   R4                                        ; 2178
                                  08    A4   9F 00023            PUSHAB  VALUE_ADDR                                ;
                                  FC    A4   9F 00026            PUSHAB  INDEX                                     ;
                                  F4    A4   9F 00029            PUSHAB  CONTEXT                                   ;
                     00000000V  00      04   FB 0002C            CALLS   #4, PARSE_TEXT_RECORD                     ;
                                01      50   E8 00033            BLBS    STATUS, 1$                                ;
                                             04 00036            RET                                               ;
                                        54   DD 00037 1$:        PUSHL   R4                                        ; 2184
                                  08    A4   DD 00039            PUSHL   VALUE_ADDR                                ;
                                        64   DD 0003C            PUSHL   SIZE                                      ;
                     00000000G  00      03   FB 0003E            CALLS   #3, LIB$CVT_DTB                           ;
                            04  A4      50   D0 00045            MOVL    R0, STATUS                                ;
                                10      04   A4   E8 00049       BLBS    STATUS, 2$                                ; 2185
                                  08    A4   9F 0004D            PUSHAB  VALUE_ADDR                                ;
                                        64   DD 00050            PUSHL   SIZE                                      ;
                                        02   DD 00052            PUSHL   #2                                        ;
                               00000000G 8F  DD 00054            PUSHL   #ERF_CVTERR                               ;
                                67      04   FB 0005A            CALLS   #4, LIB$SIGNAL                            ;
                                        64   D6 0005D 2$:        INCL    SIZE                                      ; 2190
                                01    FC A4   D1 0005F           CMPL    INDEX, #1                                 ; 2196
                                        4B   12 00063            BNEQ    3$                                        ;
                                50      64   D0 00065            MOVL    SIZE, R0                                  ; 2199
                        52      50      02   78 00068            ASHL    #2, R0, AMOUNT                            ;
                  3A    A5      50      01   A3 0006C            SUBW3   #1, R0, MAX_CPU_TYPES                     ; 2200
                                        52   DD 00071            PUSHL   AMOUNT                                    ; 2201
                                66      01   FB 00073            CALLS   #1, GET_VM                                ;
                          54    A5      50   D0 00076            MOVL    R0, MIN_RANGE_TABLE_ADDR                  ;
                                        52   DD 0007A            PUSHL   AMOUNT                                    ; 2202
                                66      01   FB 0007C            CALLS   #1, GET_VM                                ;
                          40    A5      50   D0 0007F            MOVL    R0, MAX_RANGE_TABLE_ADDR                  ;
                                52      02   C4 00083            MULL2   #2, R2                                    ; 2203
                                        52   DD 00086            PUSHL   R2                                        ;
                                66      01   FB 00088            CALLS   #1, GET_VM                                ;
                          48    A5      50   D0 0008B            MOVL    R0, MIN_MODULES_DESC                      ;
                                        52   DD 0008F            PUSHL   R2                                        ; 2204
                                66      01   FB 00091            CALLS   #1, GET_VM                                ;
                          4C    A5      50   D0 00094            MOVL    R0, MAX_MODULES_DESC                      ;
                  7E            64      01   78 00098            ASHL    #1, SIZE, -(SP)                           ; 2205
```

```
ERF               Errorlog Report Formatter                                B 7
V04-000                                                              15-Sep-1984 23:42:14    VAX-11 Bliss-32 V4.0-742               Page  63
                                                                     14-Sep-1984 12:27:17    DISK$VMSMASTER:[ERF.SRC]ERF.B32;1          (20)

                                      66          01  FB 0009C        CALLS   #1, GET_VM                                ;
                               60     A5          50  D0 0009F        MOVL    R0, PROCESSOR_TYPE_TABLE                  ;
                    7E                64          01  78 000A3        ASHL    #1, SIZE, -(SP)                           ; 2206
                                      66          01  FB 000A7        CALLS   #1, GET_VM                                ;
                               50     A5          50  D0 000AA        MOVL    R0, MIN_MAX_TABLE_SIZES                   ;
                                                  54  11 000AE        BRB     5$                                        ; 2207
                    7E                64          01  78 000B0 3$:    ASHL    #1, SIZE, -(SP)                           ; 2214
                                      66          01  FB 000B4        CALLS   #1, GET_VM                                ;
                               F0     A4          50  D0 000B7        MOVL    R0, DEVICE_ADDR                           ;
                    7E                64          01  78 000BB        ASHL    #1, SIZE, -(SP)                           ; 2215
                                      66          01  FB 000BF        CALLS   #1, GET_VM                                ;
                               0C     A4          50  D0 000C2        MOVL    R0, VERSION_ADDR                          ;
                    7E                64          02  78 000C6        ASHL    #2, SIZE, -(SP)                           ; 2216
                                      66          01  FB 000CA        CALLS   #1, GET_VM                                ;
                               10     A4          50  D0 000CD        MOVL    R0, XFER_ADDR                             ;
                    7E                64          03  78 000D1        ASHL    #3, SIZE, -(SP)                           ; 2217
                                      66          01  FB 000D5        CALLS   #1, GET_VM                                ;
                               F8     A4          50  D0 000D8        MOVL    R0, IMAGE_ADDR                            ;
                                      50     FC   A4  D0 000DC        MOVL    INDEX, R0                                 ; 2224
                               07     02          50  CF 000E0        CASEL   R0, #2, #7                                ;
       00B7                  0085   0053        0023     000E4 4$:    .WORD   6$-4$,-                                   ;
       0195                  0153   0120        00ED     000EC                7$-4$,-                                   ;
                                                                              8$-4$,-                                   ;
                                                                              10$-4$,-                                  ;
                                                                              12$-4$,-                                  ;
                                                                              14$-4$,-                                  ;
                                                                              16$-4$,-                                  ;
                                                                              18$-4$                                    ;
                                             58   A5  DD 000F4        PUSHL   MODULE_NAME_DESC                          ; 2307
                                                  50  DD 000F7        PUSHL   R0                                        ;
                                                  01  DD 000F9        PUSHL   #1                                        ;
                                      00000000G   8F  DD 000FB        PUSHL   #ERF_BADEVVAL                             ;
                                      67          04  FB 00101        CALLS   #4, LIB$SIGNAL                            ;
                                                01DB  31 00104 5$:    BRW     22$                                       ; 2308
                                      65     F0   A4  D0 00107 6$:    MOVL    DEVICE_ADDR, DISK_DEVICES                 ; 2226
                        00000000G     00     0C   A4  D0 0010B        MOVL    VERSION_ADDR, DISK_VERSION                ; 2227
                        00000000G     00     10   A4  D0 00113        MOVL    XFER_ADDR, DISK_XFER_ADDR                 ; 2228
                        00000000G     00     F8   A4  D0 0011B        MOVL    IMAGE_ADDR, DISK_IMAGE                    ; 2229
                                      50          64  D0 00123        MOVL    SIZE, R0                                  ; 2230
                    3C      A5        50          01  83 00126        SUBB3   #1, R0, MAX_DISK_TYPE                     ;
                                      51          65  D0 0012B        MOVL    DISK_DEVICES, R1                          ; 2231
                                      61          50  B0 0012E        MOVW    R0, (R1)                                  ;
                               EC     A4          01  B0 00131        MOVW    #1, DC_CLASS                              ; 2232
                                      62          11     00135        BRB     9$                                        ; 2224
                               7C     A5     F0   A4  D0 00137 7$:    MOVL    DEVICE_ADDR, TAPE_DEVICES                 ; 2236
                        00000000G     00     0C   A4  D0 0013C        MOVL    VERSION_ADDR, TAPE_VERSION                ; 2237
                        00000000G     00     10   A4  D0 00144        MOVL    XFER_ADDR, TAPE_XFER_ADDR                 ; 2238
                        00000000G     00     F8   A4  D0 0014C        MOVL    IMAGE_ADDR, TAPE_IMAGE                    ; 2239
                                      50          64  D0 00154        MOVL    SIZE, R0                                  ; 2240
                    46      A5        50          01  83 00157        SUBB3   #1, R0, MAX_TAPE_TYPE                     ;
                                      51     7C   A5  D0 0015C        MOVL    TAPE_DEVICES, R1                          ; 2241
                                      61          50  B0 00160        MOVW    R0, (R1)                                  ;
                               EC     A4          02  B0 00163        MOVW    #2, DC_CLASS                              ; 2242
                                      66          11     00167        BRB     11$                                       ; 2224
                               70     A5     F0   A4  D0 00169 8$:    MOVL    DEVICE_ADDR, SCOM_DEVICES                 ; 2246
                        00000000G     00     0C   A4  D0 0016E        MOVL    VERSION_ADDR, SCOM_VERSION                ; 2247
                        00000000G     00     10   A4  D0 00176        MOVL    XFER_ADDR, SCOM_XFER_ADDR                 ; 2248
```

```
              00000000G  00      F8   A4  D0 0017E          MOVL    IMAGE_ADDR, SCOM_IMAGE                          ; 2249
                         50           64  D0 00186          MOVL    SIZE, R0                                        ; 2250
         45   A5         50           01  83 00189          SUBB3   #1, R0, MAX_SCOM_TYPE                           ;
                         51      70   A5  D0 0018E          MOVL    SCOM_DEVICES, R1                                ; 2251
                         61           50  B0 00192          MOVW    R0, (R1)                                        ;
                  EC     A4           20  B0 00195          MOVW    #32, DC_CLASS                                   ; 2252
                         67           11  00199 9$:         BRB     13$                                             ; 2224
                  34     A5      F0   A4  D0 0019B 10$:     MOVL    DEVICE_ADDR, LP_DEVICES                         ; 2256
              00000000G  00      0C   A4  D0 001A0          MOVL    VERSION_ADDR, LP_VERSION                        ; 2257
              00000000G  00      10   A4  D0 001A8          MOVL    XFER_ADDR, LP_XFER_ADDR                         ; 2258
              00000000G  00      F8   A4  D0 001B0          MOVL    IMAGE_ADDR, LP_IMAGE                            ; 2259
                         50           64  D0 001B8          MOVL    SIZE, R0                                        ; 2260
   00000000G  00         50           01  83 001BB          SUBB3   #1, R0, MAX_LP_TYPE                             ;
                         51      34   A5  D0 001C3          MOVL    LP_DEVICES, R1                                  ; 2261
                         61           50  B0 001C7          MOVW    R0, (R1)                                        ;
                  EC     A4      43   8F  9B 001CA          MOVZBW  #67, DC_CLASS                                   ; 2262
                         64           11  001CF 11$:        BRB     15$                                             ; 2224
                  64     A5      F0   A4  D0 001D1 12$:     MOVL    DEVICE_ADDR, REALTIME_DEVICES                   ; 2266
              00000000G  00      0C   A4  D0 001D6          MOVL    VERSION_ADDR, REALTIME_VERSION                  ; 2267
              00000000G  00      10   A4  D0 001DE          MOVL    XFER_ADDR, REALTIME_XFER_ADDR                   ; 2268
              00000000G  00      F8   A4  D0 001E6          MOVL    IMAGE_ADDR, REALTIME_IMAGE                      ; 2269
                         50           64  D0 001EE          MOVL    SIZE, R0                                        ; 2270
         44   A5         50           01  83 001F1          SUBB3   #1, R0, MAX_REALTIME_TYPE                       ;
                         51      64   A5  D0 001F6          MOVL    REALTIME_DEVICES, R1                            ; 2271
                         61           50  B0 001FA          MOVW    R0, (R1)                                        ;
                  EC     A4      60   8F  9B 001FD          MOVZBW  #96, DC_CLASS                                   ; 2272
                         73           11  00202 13$:        BRB     17$                                             ; 2224
                  F0     A5      F0   A4  D0 00204 14$:     MOVL    DEVICE_ADDR, BUS_DEVICES                        ; 2276
              00000000G  00      0C   A4  D0 00209          MOVL    VERSION_ADDR, BUS_VERSION                       ; 2277
              00000000G  00      10   A4  D0 00211          MOVL    XFER_ADDR, BUS_XFER_ADDR                        ; 2278
              00000000G  00      F8   A4  D0 00219          MOVL    IMAGE_ADDR, BUS_IMAGE                           ; 2279
                         50           64  D0 00221          MOVL    SIZE, R0                                        ; 2280
         38   A5         50           01  83 00224          SUBB3   #1, R0, MAX_BUS_TYPE                            ;
                         51      F0   A5  D0 00229          MOVL    BUS_DEVICES, R1                                 ; 2281
                         61           50  B0 0022D          MOVW    R0, (R1)                                        ;
                  EC     A4      80   8F  9B 00230          MOVZBW  #128, DC_CLASS                                  ; 2282
                         75           11  00235 15$:        BRB     19$                                             ; 2224
                  5C     A5      F0   A4  D0 00237 16$:     MOVL    DEVICE_ADDR, PACKET_PROCESSOR_DEVICES           ; 2286
              00000000G  00      0C   A4  D0 0023C          MOVL    VERSION_ADDR, PACKET_PROCESSOR_VERSION          ; 2287
              00000000G  00      10   A4  D0 00244          MOVL    XFER_ADDR, PACKET_PROCESSOR_XFER_ADDR           ; 2288
              00000000G  00      F8   A4  D0 0024C          MOVL    IMAGE_ADDR, PACKET_PROCESSOR_IMAGE              ; 2289
                         51           64  D0 00254          MOVL    SIZE, R1                                        ; 2290
   00000000G  00         51           01  83 00257          SUBB3   #1, R1, MAX_MISC_TYPE                           ;
                         50      5C   A5  D0 0025F          MOVL    PACKET_PROCESSOR_DEVICES, R0                    ; 2291
                         60           51  B0 00263          MOVW    R1, (R0)                                        ;
              7E         51           01  78 00266          ASHL    #1, R1, -(SP)                                   ; 2292
                         66           01  FB 0026A          CALLS   #1, GET_VM                                      ;
              00000000G  00           50  D0 0026D          MOVL    R0, TRANSLATE_ENTRY_TABLE                       ;
                                 EC   A4  B4 00274          CLRW    DC_CLASS                                        ; 2293
                         33           11  00277 17$:        BRB     19$                                             ; 2224
                00A4     C5      F0   A4  D0 00279 18$:     MOVL    DEVICE_ADDR, WORKSTATION_DEVICES                ; 2297
              00000000G  00      0C   A4  D0 0027F          MOVL    VERSION_ADDR, WORKSTATION_VERSION               ; 2298
              00000000G  00      10   A4  D0 00287          MOVL    XFER_ADDR, WORKSTATION_XFER_ADDR                ; 2299
              00000000G  00      F8   A4  D0 0028F          MOVL    IMAGE_ADDR, WORKSTATION_IMAGE                   ; 2300
                         50           64  D0 00297          MOVL    SIZE, R0                                        ; 2301
         47   A5         50           01  83 0029A          SUBB3   #1, R0, MAX_WORKSTATION_TYPE                    ;
                         51    00A4   C5  D0 0029F          MOVL    WORKSTATION_DEVICES, R1                         ; 2302
```

```
                            61         50  B0 002A4        MOVW     R0, (R1)                                          ;
                        EC  A4    46   8F  9B 002A7        MOVZBW   #70, DC_CLASS                                     ; 2303
                            53    39   A5  9A 002AC  19$:  MOVZBL   MAX_CLASSES, R3                                   ; 2318
                            52    EC   A4  3C 002B0        MOVZWL   DC_CLASS, R2                                      ; 2320
                                       50  D4 002B4        CLRL     COUNT                                             ;
                                       19  11 002B6        BRB      21$                                               ;
                            51    FC   A5  D0 002B8  20$:  MOVL     DEV_CLASS_PTR, R1                                 ;
                                     6140  3F 002BC        PUSHAW   (R1)[COUNT]                                       ;
       52          9E       10         00  ED 002BF        CMPZV    #0, #16, @(SP)+, R2                               ;
                                       0B  12 002C4        BNEQ     21$                                               ;
                            51    F8   A5  D0 002C6        MOVL     DEV_ADDRS_PTR, R1                                 ; 2322
                          6140    F0   A4  D0 002CA        MOVL     DEVICE_ADDR, (R1)[COUNT]                          ;
                                       11  11 002CF        BRB      22$                                               ; 2323
                   E3       50         53  F3 002D1  21$:  AOBLEQ   R3, COUNT, 20$                                    ; 2318
                                       52  DD 002D5        PUSHL    R2                                                ; 2327
                                       01  DD 002D7        PUSHL    #1                                                ;
                                00000000G  8F  DD 002D9    PUSHL    #ERF_CLSTBLERR                                    ;
                            67         03  FB 002DF        CALLS    #3, LIB$SIGNAL                                    ;
                            50         01  D0 002E2  22$:  MOVL     #1, R0                                            ; 2328
                                       04     002E5        RET                                                        ; 2329

; Routine Size:  742 bytes,    Routine Base:  $CODE + 0B6A
```

```
1 Routine PARSE_DEVICE_DESC_RECORD =
2 BEGIN
2 !++
2 ! Functional description :
2 !       Each call to 'PARSE_TEXT_RECORD' returns the next item in
2 !       the record (comma seperated list). The state of CONTEXT
2 !       determines if more items are available in the record. The
2 !       item count (I) does not consider the INDEX as a item in the
2 !       record. The first item in the record is the item after the
2 !       '='.
2 !         Record format:
2 !               INDEX = ITEM_ONE,ITEM_TWO,..ITEM_FOUR
2 !         See 'DEVICES' module in SYS$LIBRARY:ERFLIB.TLB for
2 !         more information.
2 !
2 !       For the first and second items returned convert them to dec.
2 !
2 !       If this is ITEM_ONE then save it as a device class. For
2 !       each device class (ITEM_ONE) see if the device type (INDEX)
2 !       is greater the is max allowable value.
2 !
2 !       If this is ITEM_TWO the save it as the allowable version
2 !       number for a loadable routine.
2 !
2 !       If this is ITEM_THREE then it is a two character device
2 !       name. Use INDEX to obtain the address in which this string
2 !       should be copied to.
2 !
2 !       If this is ITEM_FOUR then it is the name of the loadable image
2 !       that will interpurt this deviced error packet. This string
2 !       is copied to a table indexed by INDEX.
2 !
2 ! Calling sequence
2 !
2 ! Input parameters
2 !
2 !       RECORD_DESC global descriptor pointing at the record read
2 !                   from the text library.
2 !
2 ! Output parameters
2 !
2 !       None
2 !
2 ! Routine value
2 !
2 !       Worst error is returned.
2 !
2 !----
2
2 OWN
2       Context,
2       Dc_class:       BYTE,
2       I,
2       Index,
2       Item_address:   REF $BBLOCK[dsc$k_d_bln],
2       Size,
2       Status,
                Table_addr,
                Temp,
                Value_addr,
                Version;
```

```
2 Context = 0;      ! Clear the context
2 I = 1;            ! Set the item count
2
2 Do
3  Begin
3  CALL_FUNCTION ( Parse_text_record ( context, index, value_addr, size ) );
3
3  If .I LEQ 2 then         ! If first or second item
4    Begin                  ! then convert to binary
4    Status = LIB$CVT_DTB ( .size, .Value_addr, size );
4    If NOT .status then Signal (erf_cvterr, 2,.size,value_addr) ;
4
4    If .I EQL 1 then dc_class = .size;   ! If first item save
3    End;
3
3  !
3  ! Determine which device class is being processed.
3  !
3  Selectoneu .dc_class of
3   Set
3
3     [DC$_ZERO_CLASS]:
4      Begin
4       !
4       ! Make sure it's a valid device type for this class of devices.
4       !
4      If .index GTR .max_misc_type then
4         Signal (erf_badevtyp, 2, .index, .module_name_desc);
4
4        !
4        ! Determine which portion of the record is being processed and
4        ! get the address of the location to store the data.
4        !
4       Case .I from 2 to 4 of
4        Set
4         [2]:
4           Packet_processor_version [ .index ] = .size;
4         [3]:
4           Table_addr = Packet_processor_devices[.index];
4         [4]:
4           Table_addr = Packet_processor_image[.index, desc_one];
4         [Outrange]:
4        TES;
3      End ;
3
3     [DC$_DISK]:
4      Begin
4       !
4       ! Make sure it's a valid device type for this class of devices.
4       !
4      If .index GTR .max_disk_type then
4         Signal (erf_badevtyp, 2, .index, .Module_name_desc);
4
4         !
4         ! Determine which portion of the record is being processed and
4         ! get the address of the location to store the data.
4         !
```

```
4            Case .I from 2 to 4 of
4              Set
4                [2]:
4                    Disk_version [ .index ] = .size;
4                [3]:
4                    Table_addr = Disk_devices[.index];
4                [4]:
4                    Table_addr = Disk_image[.index, desc_one];
4                [Outrange]:
4              TES;
3        End ;
3
3       [DC$_TAPE]:
4        Begin
4          !
4          ! Make sure it's a valid device type for this class of devices.
4          !
4          If .index GTR .max_tape_type then
4            Signal (erf_badevtyp, 2, .index, .Module_name_desc);
4
4            !
4            ! Determine which portion of the record is being processed and
4            ! get the address of the location to store the data.
4            !
4            Case .I from 2 to 4 of
4              Set
4                [2]:
4                    Tape_version [ .index ] = .size;
4                [3]:
4                    Table_addr = tape_devices[.index];
4                [4]:
4                    Table_addr = Tape_image[.index, desc_one];
4                [Outrange]:
4              TES;
3        End ;
3
3       [DC$_SCOM]:
4        Begin
4        !
4        ! Make sure it's a valid device type for this class of devices.
4        !
4        If .index GTR .max_scom_type
4        Then
4            Signal (erf_badevtyp, 2, .index, .Module_name_desc) ;
4
4        !
4        ! Determine which portion of the record is being processed and
4        ! get the address of the location to store the data.
4        !
4        Case .I from 2 to 4 of
4          Set
4            [2]:
4              Scom_version[.index] = .size ;
4
4            [3]:
4              Table_addr = scom_devices[.index];
4
```

```
   [4]:
     Table_addr = scom_image[.index, desc_one];

   [Outrange]:
   TES ;
 End ;

[DC$_LP]:
 Begin
 !
 ! Make sure it's a valid device type for this class of devices.
 !
 If .index GTR .max_lp_type
 Then
     Signal (erf_badevtyp, 2, .index, .Module_name_desc) ;

 !
 ! Determine which portion of the record is being processed and
 ! get the address of the location to store the data.
 !
 Case .I from 2 to 4 of
  Set
   [2]:
     Lp_version[.index] = .size ;

   [3]:
     Table_addr = lp_devices[.index];

   [4]:
     Table_addr = lp_image[.index, desc_one];

   [Outrange]:
   TES ;
 End ;

[DC$_REALTIME]:
 Begin
 !
 ! Make sure it's a valid device type for this class of devices.
 !
 If .index GTR .max_realtime_type
 Then
     Signal (erf_badevtyp, 2, .index, .Module_name_desc) ;

 !
 ! Determine which portion of the record is being processed and
 ! get the address of the location to store the data.
 !
 Case .I from 2 to 4 of
  Set
   [2]:
     Realtime_version[.index] = .size ;

   [3]:
     Table_addr = realtime_devices[.index];

   [4]:
```

```
                Table_addr = realtime_image[.index, desc_one];

              [Outrange]:
              TES ;
            End ;

          [DC$_BUS]:
            Begin
            !
            ! Make sure it's a valid device type for this class of devices.
            !
            If .index GTR .max_bus_type
            Then
                Signal (erf_badevtyp, 2, .index, .Module_name_desc) ;

            !
            ! Determine which portion of the record is being processed and
            ! get the address of the location to store the data.
            !
            Case .I from 2 to 4 of
              Set
              [2]:
                Bus_version[.index] = .size ;

              [3]:
                Table_addr = bus_devices[.index];

              [4]:
                Table_addr = bus_image[.index, desc_one];

              [Outrange]:
              TES ;
            End ;

          [DC$_WORKSTATION]:
            Begin
             !
             ! Make sure it's a valid device type for this class of devices.
             !
             If .index GTR .max_workstation_type then
                Signal (erf_badevtyp, 2, .index, .Module_name_desc);

               !
               ! Determine which portion of the record is being processed and
               ! get the address of the location to store the data.
               !
              Case .I from 2 to 4 of
                Set
                [2]:
                    Workstation_version [ .index ] = .size;
                [3]:
                    Table_addr = Workstation_devices[.index];
                [4]:
                    Table_addr = Workstation_image[.index, desc_one];
                [Outrange]:
                TES;
            End ;
```

```
; 2073      2620 3
; 2074      2621 3   TES;
; 2075      2622 3
; 2076      2623 3   If .I EQL 3 then CH$MOVE (.size, .value_addr, .table_addr );
; 2077      2624 3
; 2078      2625 3   If .I EQL 4 then
; 2079      2626 4     Begin
; 2080      2627 4     Item_address = .table_addr;
; 2081      2628 4     Item_address[dsc$w_length] = .size ;
; 2082      2629 4     Item_address[dsc$a_pointer] = get_vm(.size);
; 2083      2630 4     CH$MOVE (.size, .value_addr, .item_address[dsc$a_pointer]);
; 2084      2631 3     End;
; 2085      2632 3
; 2086      2633 3   I = .I + 1;
; 2087      2634 3
; 2088      2635 3   End
; 2089      2636 3
; 2090      2637 2 While .context EQL 1;
; 2091      2638 2
; 2092      2639 2 Return true ;
; 2093      2640 1 End ;


                                                  .PSECT  $OWN$,NOEXE,  PIC,2

                                   00058 CONTEXT:.BLKB    4
                                   0005C DC_CLASS:
                                                 .BLKB    1
                                   0005D         .BLKB    3
                                   00060 I:      .BLKB    4
                                   00064 INDEX:  .BLKB    4
                                   00068 ITEM_ADDRESS:
                                                 .BLKB    4
                                   0006C SIZE:   .BLKB    4
                                   00070 STATUS: .BLKB    4
                                   00074 TABLE_ADDR:
                                                 .BLKB    4
                                   00078 TEMP:   .BLKB    4
                                   0007C VALUE_ADDR:
                                                 .BLKB    4
                                   00080 VERSION:.BLKB    4


                                                  .PSECT  $CODE,NOWRT,  PIC,2

                              07FC 00000 PARSE_DEVICE_DESC_RECORD:
                                                 .WORD    Save R2,R3,R4,R5,R6,R7,R8,R9,R10                 ; 2330
            5A 00000000G   8F   D0 00002          MOVL     #ERF_BADEVTYP, R10                              ;
            59 00000000G   00   9E 00009          MOVAB    LIB$SIGNAL, R9                                  ;
            58 00000000'   00   9E 00010          MOVAB    MODULE_NAME_DESC, R8                            ;
            57 00000000'   00   9E 00017          MOVAB    I, R7                                           ;
                      F8   A7   D4 0001E          CLRL     CONTEXT                                         ; 2392
            67             01   D0 00021          MOVL     #1, I                                           ; 2393
                      0C   A7   9F 00024 1$:      PUSHAB   SIZE                                            ; 2397
                      1C   A7   9F 00027          PUSHAB   VALUE_ADDR                                      ;
```

```
                              04      A7  9F 0002A         PUSHAB  INDEX                                  :
                              F8      A7  9F 0002D         PUSHAB  CONTEXT                                :
              00000000V  00           04  FB 00030         CALLS   #4, PARSE_TEXT_RECORD                  :
                         01           50  E8 00037         BLBS    STATUS, 2$                             :
                                      04  0003A            RET                                            :
                         02           67  D1 0003B  2$:    CMPL    I, #2                                  : 2399
                                      33  14 0003E         BGTR    4$                                     :
                              0C      A7  9F 00040         PUSHAB  SIZE                                   : 2401
                              1C      A7  DD 00043         PUSHL   VALUE_ADDR                             :
                              0C      A7  DD 00046         PUSHL   SIZE                                   :
              00000000G  00           03  FB 00049         CALLS   #3, LIB$CVT_DTB                        :
                    10   A7           50  D0 00050         MOVL    R0, STATUS                             :
                         11   10      A7  E8 00054         BLBS    STATUS, 3$                             : 2402
                              1C      A7  9F 00058         PUSHAB  VALUE_ADDR                             :
                              0C      A7  DD 0005B         PUSHL   SIZE                                   :
                                      02  DD 0005E         PUSHL   #2                                     :
                        00000000G     8F  DD 00060         PUSHL   #ERF_CVTERR                            :
                         69           04  FB 00066         CALLS   #4, LIB$SIGNAL                         :
                         01           67  D1 00069  3$:    CMPL    I, #1                                  : 2404
                                      05  12 0006C         BNEQ    4$                                     :
                    FC   A7   0C      A7  90 0006E         MOVB    SIZE, DC_CLASS                         :
                         50   FC      A7  9A 00073  4$:    MOVZBL  DC_CLASS, R0                           : 2410
                                      3E  12 00077         BNEQ    10$                                    : 2413
                         51   04      A7  D0 00079         MOVL    INDEX, R1                              : 2418
51 00000000G  00         08           00  ED 0007D         CMPZV   #0, #8, MAX_MISC_TYPE, R1              :
                                      0B  18 00086         BGEQ    5$                                     :
                                      68  DD 00088         PUSHL   MODULE_NAME_DESC                       : 2419
                                      51  DD 0008A         PUSHL   R1                                     :
                                      02  DD 0008C         PUSHL   #2                                     :
                                      5A  DD 0008E         PUSHL   R10                                    :
                         69           04  FB 00090         CALLS   #4, LIB$SIGNAL                         :
             02          02           67  CF 00093  5$:    CASEL   I, #2, #2                              : 2425
           0017        0011     0008         00097  6$:    .WORD   7$-6$,-                                :
                                                                   8$-6$,-                                :
                                                                   9$-6$                                  :
                                      7E  11 0009D         BRB     19$                                    :
                         51 00000000G 00  D0 0009F  7$:    MOVL    PACKET_PROCESSOR_VERSION, R1           : 2428
                                      7E  11 000A6         BRB     21$                                    :
                         51   04      A8  D0 000A8  8$:    MOVL    PACKET_PROCESSOR_DEVICES, R1           : 2430
                                      7E  11 000AC         BRB     23$                                    :
                         51 00000000G 00  D0 000AE  9$:    MOVL    PACKET_PROCESSOR_IMAGE, R1             : 2432
                                      7E  11 000B5         BRB     25$                                    :
                         01           50  91 000B7  10$:   CMPB    R0, #1                                 : 2437
                                      3B  12 000BA         BNEQ    16$                                    :
                         51   04      A7  D0 000BC         MOVL    INDEX, R1                              : 2442
51          E4    A8     08           00  ED 000C0         CMPZV   #0, #8, MAX_DISK_TYPE, R1              :
                                      0B  18 000C6         BGEQ    11$                                    :
                                      68  DD 000C8         PUSHL   MODULE_NAME_DESC                       : 2443
                                      51  DD 000CA         PUSHL   R1                                     :
                                      02  DD 000CC         PUSHL   #2                                     :
                                      5A  DD 000CE         PUSHL   R10                                    :
                         69           04  FB 000D0         CALLS   #4, LIB$SIGNAL                         :
             02          02           67  CF 000D3  11$:   CASEL   I, #2, #2                              : 2449
           0017        0011     0008         000D7  12$:   .WORD   13$-12$,-                              :
                                                                   14$-12$,-                              :
                                                                   15$-12$                                :
                                      7E  11 000DD         BRB     29$                                    :
```

```
                          51 00000000G  00  D0 000DF 13$:  MOVL   DISK_VERSION, R1                        ; 2452
                                        7E  11 000E6       BRB    31$                                     ;
                          51        A8  A8  D0 000E8 14$:  MOVL   DISK_DEVICES, R1                        ; 2454
                                        7E  11 000EC       BRB    33$                                     ;
                          51 00000000G  00  D0 000EE 15$:  MOVL   DISK_IMAGE, R1                          ; 2456
                                        7E  11 000F5       BRB    35$                                     ;
                          02            50  91 000F7 16$:  CMPB   R0, #2                                  ; 2461
                                        3B  12 000FA       BNEQ   26$                                     ;
                          51        04  A7  D0 000FC       MOVL   INDEX, R1                               ; 2466
     51       EE    A8    08            00  ED 00100       CMPZV  #0, #8, MAX_TAPE_TYPE, R1               ;
                                        0B  18 00106       BGEQ   17$                                     ;
                                        68  DD 00108       PUSHL  MODULE_NAME_DESC                        ; 2467
                                        51  DD 0010A       PUSHL  R1                                      ;
                                        02  DD 0010C       PUSHL  #2                                      ;
                                        5A  DD 0010E       PUSHL  R10                                     ;
                          69            04  FB 00110       CALLS  #4, LIB$SIGNAL                          ;
              02          02            67  CF 00113 17$:  CASEL  I, #2, #2                               ; 2473
            0017        0011          0008     00117 18$:  .WORD  20$-18$,-                               ;
                                                                  22$-18$,-                               ;
                                                                  24$-18$                                 ;
                                        3E  11 0011D 19$:  BRB    29$                                     ;
                          51 00000000G  00  D0 0011F 20$:  MOVL   TAPE_VERSION, R1                        ; 2476
                                        3E  11 00126 21$:  BRB    31$                                     ;
                          51        24  A8  D0 00128 22$:  MOVL   TAPE_DEVICES, R1                        ; 2478
                                        3E  11 0012C 23$:  BRB    33$                                     ;
                          51 00000000G  00  D0 0012E 24$:  MOVL   TAPE_IMAGE, R1                          ; 2480
                                        3E  11 00135 25$:  BRB    35$                                     ;
                          20            50  91 00137 26$:  CMPB   R0, #32                                 ; 2485
                                        3B  12 0013A       BNEQ   36$                                     ;
                          51        04  A7  D0 0013C       MOVL   INDEX, R1                               ; 2490
     51       ED    A8    08            00  ED 00140       CMPZV  #0, #8, MAX_SCOM_TYPE, R1               ;
                                        0B  18 00146       BGEQ   27$                                     ;
                                        68  DD 00148       PUSHL  MODULE_NAME_DESC                        ; 2492
                                        51  DD 0014A       PUSHL  R1                                      ;
                                        02  DD 0014C       PUSHL  #2                                      ;
                                        5A  DD 0014E       PUSHL  R10                                     ;
                          69            04  FB 00150       CALLS  #4, LIB$SIGNAL                          ;
              02          02            67  CF 00153 27$:  CASEL  I, #2, #2                               ; 2498
            0017        0011          0008     00157 28$:  .WORD  30$-28$,-                               ;
                                                                  32$-28$,-                               ;
                                                                  34$-28$                                 ;
                                        42  11 0015D 29$:  BRB    39$                                     ;
                          51 00000000G  00  D0 0015F 30$:  MOVL   SCOM_VERSION, R1                        ; 2501
                                        42  11 00166 31$:  BRB    41$                                     ;
                          51        18  A8  D0 00168 32$:  MOVL   SCOM_DEVICES, R1                        ; 2504
                                        42  11 0016C 33$:  BRB    43$                                     ;
                          51 00000000G  00  D0 0016E 34$:  MOVL   SCOM_IMAGE, R1                          ; 2507
                                        42  11 00175 35$:  BRB    45$                                     ;
                    43    8F            50  91 00177 36$:  CMPB   R0, #67                                 ; 2513
                                        3E  12 0017B       BNEQ   46$                                     ;
                          51        04  A7  D0 0017D       MOVL   INDEX, R1                               ; 2518
     51 00000000G  00     08            00  ED 00181       CMPZV  #0, #8, MAX_LP_TYPE, R1                 ;
                                        0B  18 0018A       BGEQ   37$                                     ;
                                        68  DD 0018C       PUSHL  MODULE_NAME_DESC                        ; 2520
                                        51  DD 0018E       PUSHL  R1                                      ;
                                        02  DD 00190       PUSHL  #2                                      ;
                                        5A  DD 00192       PUSHL  R10                                     ;
                          69           04 FB 00194        CALLS   #4, LIB$SIGNAL                            ;
              02          02           67 CF 00197 37$:   CASEL   I, #2, #2                                 ; 2526
            0017        0011         0008    0019B 38$:   .WORD   40$-38$,-                                 ;
                                                                  42$-38$,-                                 ;
                                                                  44$-38$                                   ;
                                       3F 11 001A1 39$:   BRB     49$                                       ;
                          51 00000000G 00 D0 001A3 40$:   MOVL    LP_VERSION, R1                            ; 2529
                                       3F 11 001AA 41$:   BRB     51$                                       ;
                          51        DC A8 D0 001AC 42$:   MOVL    LP_DEVICES, R1                            ; 2532
                                       3F 11 001B0 43$:   BRB     53$                                       ;
                          51 00000000G 00 D0 001B2 44$:   MOVL    LP_IMAGE, R1                              ; 2535
                                       3F 11 001B9 45$:   BRB     55$                                       ;
                     60   8F           50 91 001BB 46$:   CMPB    R0, #96                                   ; 2541
                                       3B 12 001BF        BNEQ    56$                                       ;
                          51        04 A7 D0 001C1        MOVL    INDEX, R1                                 ; 2546
51        EC  A8          08           00 ED 001C5        CMPZV   #0, #8, MAX_REALTIME_TYPE, R1             ;
                                       0B 18 001CB        BGEQ    47$                                       ;
                                       68 DD 001CD        PUSHL   MODULE_NAME_DESC                          ; 2548
                                       51 DD 001CF        PUSHL   R1                                        ;
                                       02 DD 001D1        PUSHL   #2                                        ;
                                       5A DD 001D3        PUSHL   R10                                       ;
                          69           04 FB 001D5        CALLS   #4, LIB$SIGNAL                            ;
              02          02           67 CF 001D8 47$:   CASEL   I, #2, #2                                 ; 2554
            0017        0011         0008    001DC 48$:   .WORD   50$-48$,-                                 ;
                                                                  52$-48$,-                                 ;
                                                                  54$-48$                                   ;
                                       3F 11 001E2 49$:   BRB     59$                                       ;
                          51 00000000G 00 D0 001E4 50$:   MOVL    REALTIME_VERSION, R1                      ; 2557
                                       3F 11 001EB 51$:   BRB     61$                                       ;
                          51        0C A8 D0 001ED 52$:   MOVL    REALTIME_DEVICES, R1                      ; 2560
                                       3F 11 001F1 53$:   BRB     63$                                       ;
                          51 00000000G 00 D0 001F3 54$:   MOVL    REALTIME_IMAGE, R1                        ; 2563
                                       3F 11 001FA 55$:   BRB     65$                                       ;
                     80   8F           50 91 001FC 56$:   CMPB    R0, #128                                  ; 2569
                                       3B 12 00200        BNEQ    66$                                       ;
                          50        04 A7 D0 00202        MOVL    INDEX, R0                                 ; 2574
50        E0  A8          08           00 ED 00206        CMPZV   #0, #8, MAX_BUS_TYPE, R0                  ;
                                       0B 18 0020C        BGEQ    57$                                       ;
                                       68 DD 0020E        PUSHL   MODULE_NAME_DESC                          ; 2576
                                       50 DD 00210        PUSHL   R0                                        ;
                                       02 DD 00212        PUSHL   #2                                        ;
                                       5A DD 00214        PUSHL   R10                                       ;
                          69           04 FB 00216        CALLS   #4, LIB$SIGNAL                            ;
              02          02           67 CF 00219 57$:   CASEL   I, #2, #2                                 ; 2582
            0017        0011         0008    0021D 58$:   .WORD   60$-58$,-                                 ;
                                                                  62$-58$,-                                 ;
                                                                  64$-58$                                   ;
                                       72 11 00223 59$:   BRB     75$                                       ;
                          51 00000000G 00 D0 00225 60$:   MOVL    BUS_VERSION, R1                           ; 2585
                                       3F 11 0022C 61$:   BRB     70$                                       ;
                          51        98 A8 D0 0022E 62$:   MOVL    BUS_DEVICES, R1                           ; 2588
                                       48 11 00232 63$:   BRB     72$                                       ;
                          51 00000000G 00 D0 00234 64$:   MOVL    BUS_IMAGE, R1                             ; 2591
                                       51 11 0023B 65$:   BRB     74$                                       ;
                     46   8F           50 91 0023D 66$:   CMPB    R0, #70                                   ; 2597
                                       54 12 00241        BNEQ    75$                                       ;
                          50        04 A7 D0 00243        MOVL    INDEX, R0                                 ; 2602
```

```
     50     EF    A8          08            00  ED 00247         CMPZV   #0, #8, MAX_WORKSTATION_TYPE, R0              ;
                                            0B  18 0024D         BGEQ    67$                                           ;
                                            68  DD 0024F         PUSHL   MODULE_NAME_DESC                              ; 2603
                                            50  DD 00251         PUSHL   R0                                            ;
                                            02  DD 00253         PUSHL   #2                                            ;
                                            5A  DD 00255         PUSHL   R10                                           ;
                              69            04  FB 00257         CALLS   #4, LIB$SIGNAL                                ;
                        02    02            67  CF 0025A 67$:    CASEL   I, #2, #2                                     ; 2609
                  0029      001A          0008     0025E 68$:    .WORD   69$-68$,-                                     ;
                                                                         71$-68$,-                                     ;
                                                                         73$-68$                                       ;
                                            31  11 00264         BRB     75$                                           ;
                              51 00000000G  00  D0 00266 69$:    MOVL    WORKSTATION_VERSION, R1                       ; 2612
                              50        04  A7  D0 0026D 70$:    MOVL    INDEX, R0                                     ;
                            6140        0C  A7  B0 00271         MOVW    SIZE, (R1)[R0]                                ;
                                            1F  11 00276         BRB     75$                                           ;
                              51        4C  A8  D0 00278 71$:    MOVL    WORKSTATION_DEVICES, R1                       ; 2614
                              50        04  A7  D0 0027C 72$:    MOVL    INDEX, R0                                     ;
                       14     A7          6140  3E 00280         MOVAW   (R1)[R0], TABLE_ADDR                          ;
                                            10  11 00285         BRB     75$                                           ;
                              51 00000000G  00  D0 00287 73$:    MOVL    WORKSTATION_IMAGE, R1                         ; 2616
                              50        04  A7  D0 0028E 74$:    MOVL    INDEX, R0                                     ;
                       14     A7          6140  7E 00292         MOVAQ   (R1)[R0], TABLE_ADDR                          ;
                              56            67  D0 00297 75$:    MOVL    I, R6                                         ; 2623
                              03            56  D1 0029A         CMPL    R6, #3                                        ;
                                            0D  12 0029D         BNEQ    76$                                           ;
                              51        1C  A7  D0 0029F         MOVL    VALUE_ADDR, R1                                ;
                              50        14  A7  D0 002A3         MOVL    TABLE_ADDR, R0                                ;
                 60           61        0C  A7  28 002A7         MOVC3   SIZE, (R1), (R0)                              ;
                              04            56  D1 002AC 76$:    CMPL    R6, #4                                        ; 2625
                                            2B  12 002AF         BNEQ    77$                                           ;
                       08     A7        14  A7  D0 002B1         MOVL    TABLE_ADDR, ITEM_ADDRESS                      ; 2627
                              52        08  A7  D0 002B6         MOVL    ITEM_ADDRESS, R2                              ; 2628
                              50        0C  A7  D0 002BA         MOVL    SIZE, R0                                      ;
                              62            50  B0 002BE         MOVW    R0, (R2)                                      ;
                                            50  DD 002C1         PUSHL   R0                                            ; 2629
                 00000000G    00            01  FB 002C3         CALLS   #1, GET_VM                                    ;
                       04     A2            50  D0 002CA         MOVL    R0, 4(R2)                                     ;
                              51        1C  A7  D0 002CE         MOVL    VALUE_ADDR, R1                                ; 2630
                              50        08  A7  D0 002D2         MOVL    ITEM_ADDRESS, R0                              ;
          04    B0            61        0C  A7  28 002D6         MOVC3   SIZE, (R1), @4(R0)                            ;
                                            67  D6 002DC 77$:    INCL    I                                             ; 2633
                       01           F8  A7  D1 002DE             CMPL    CONTEXT, #1                                   ; 2637
                                            03  12 002E2         BNEQ    78$                                           ;
                                          FD3D  31 002E4         BRW     1$                                            ;
                              50            01  D0 002E7 78$:    MOVL    #1, R0                                        ; 2639
                                                04 002EA         RET                                                   ; 2640

; Routine Size:  747 bytes,    Routine Base:  $CODE + 0E50
```

```
1 Routine PARSE_MODULE_NAMES =
2 Begin
2 !++
2 ! Functional description
2 !
2 !       This routine builds a descriptor table, which contains the names
2 !       of library modules to be processed.
2 !
2 ! Calling sequence
2 !
2 ! Input parameters
2 !
2 !       None.
2 !
2 ! Output parameters
2 !
2 !
2 ! Routine value
2 !
2 !
2 !----
2
2 Local
2         Desc: REF $bblock [],
2         Context: initial (0),
2         Index,
2         Value_addr,
2         Size;
2
2 Do
3   Begin
3
3   Item_count = .item_count + 1;
3
3   If .item_count GTR .table_length then
3       (signal (erf_badevtyp, 2, .item_count, .Module_name_desc); Return true);
3
3   Call_function (Parse_text_record ( context, index, value_addr, size) );
3
3   Desc = desc_table_address[.item_count,desc_one];
3
3   Desc[dsc$w_length] = .size;
3   Desc[dsc$b_class] = dsc$k_class_d;
3   Desc[dsc$b_dtype] = dsc$k_dtype_t;
3 ! SIZE could be zero if the lib. module had ",," or ",<EOL>"
3 ! This is could be a problem.
3   Desc[dsc$a_pointer] = get_vm(.size);
3
3   CH$MOVE (.desc[dsc$w_length], .value_addr, .desc[dsc$a_pointer]);
3
3   End
2 While .context EQL 1;
2
2 Return true;
1 End;
```

```
                                           00FC 00000 PARSE_MODULE_NAMES:
                                                                  .WORD    Save R2,R3,R4,R5,R6,R7               ; 2641
                            57 00000000'  00  9E 00002            MOVAB    ITEM_COUNT, R7                       ;
                            5E            10  C2 00009            SUBL2    #16, SP                              ;
                                      0C  AE  D4 0000C            CLRL     CONTEXT                              ; 2642
                                          67  D6 0000F 1$:        INCL     ITEM_COUNT                           ; 2673
                            50            67  D0 00011            MOVL     ITEM_COUNT, R0                       ; 2675
50      60   A7             08            00  ED 00014            CMPZV    #0, #8, TABLE_LENGTH, R0             ;
                                          16  18 0001A            BGEQ     2$                                   ;
                                      34  A7  DD 0001C            PUSHL    MODULE_NAME_DESC                     ; 2676
                                          50  DD 0001F            PUSHL    R0                                   ;
                                          02  DD 00021            PUSHL    #2                                   ;
                               00000000G  8F  DD 00023            PUSHL    #ERF_BADEVTYP                        ;
                 00000000G  00            04  FB 00029            CALLS    #4, LIB$SIGNAL                       ;
                                          42  11 00030            BRB      3$                                   ;
                                          5E  DD 00032 2$:        PUSHL    SP                                   ; 2678
                                      08  AE  9F 00034            PUSHAB   VALUE_ADDR                           ;
                                      10  AE  9F 00037            PUSHAB   INDEX                                ;
                                      18  AE  9F 0003A            PUSHAB   CONTEXT                              ;
                 00000000V  00            04  FB 0003D            CALLS    #4, PARSE_TEXT_RECORD                ;
                            30            50  E9 00044            BLBC     STATUS, 4$                           ;
                            51        D0  A7  D0 00047            MOVL     DESC_TABLE_ADDRESS, R1               ; 2680
                            50            67  D0 0004B            MOVL     ITEM_COUNT, R0                       ;
                            56          6140  7E 0004E            MOVAQ    (R1)[R0], DESC                       ;
                            66            6E  B0 00052            MOVW     SIZE, (DESC)                         ; 2682
                        02  A6      020E  8F  B0 00055            MOVW     #526, 2(DESC)                        ; 2684
                                          6E  DD 0005B            PUSHL    SIZE                                 ; 2687
                 00000000G  00            01  FB 0005D            CALLS    #1, GET_VM                           ;
                        04  A6            50  D0 00064            MOVL     R0, 4(DESC)                          ;
        04   B6         04  BE            66  28 00068            MOVC3    (DESC), @VALUE_ADDR, @4(DESC)        ; 2689
                            01        0C  AE  D1 0006E            CMPL     CONTEXT, #1                          ; 2692
                                          9B  13 00072            BEQL     1$                                   ;
                            50            01  D0 00074 3$:        MOVL     #1, R0                               ; 2694
                                              04 00077 4$:        RET                                           ; 2695

; Routine Size:  120 bytes,    Routine Base:  $CODE + 113B


; 2150         2696  1
```

```
Routine PARSE_TEXT_RECORD ( context, index, value_addr, size ) =
BEGIN
!++
! Functional description
!
!       This routine parses a record that was previously read from
!       the text library. Each call to this routine returns the next
!       item in the record(comma seperated list). CONTEXT is set after
!       returning all items in the list. The value of INDEX, in binary,
!       is constant for all items in a record. VALUE_ADDR is the starting
!       address of the next item. Its size is returned in SIZE. All records
!       processed by this routine are expected to have 4 items after there
!       index. See 'DEVICES' module in SYS$LIBRARY:ERFLIB.TLB for more
!       information.
!
!
! Calling sequence
!
! Input parameters
!
!                    Context : Should always be zero in first call to this routine
!                              on return from this routine it is set to one to
!                              specify that there more values in the list.
!
!
! Output parameters
!
!                      Index : Binary value of the number to the left of the equal
!                              sign.
!
!                 Value_addr : Starting address of the string to be returned.
!
!                       Size : The length of the field pointed to by VALUE_ADDR
!
!                    Context : Binary 1 to indicate more values in the comma
!                              separated list.
!                              Binary 0 to indicate no more in the list.
!
!
! Routine value
!
!       Worst error is returned.
!
!----

LITERAL
        Max_deliminters = 3,
        TAB = 9;

OWN
        Context_length,
        Context_pointer,
        Delim_position:         INITIAL (0),
        Length_to_move,
        Offset,
        Status,
        Temp_ptr;
```

```
: 2209          2754  2
: 2210          2755  2
```

```
2 !
2 ! If context equals 1 then compress the record and get the index.
2 !
2
2 If ..context EQL 0 then
3  Begin
3
3    !
3    ! Setup pointer to start of record and record length.
3    !
3   Context_pointer = CH$PTR ( .Record_desc [dsc$a_pointer] );
3   Context_length = .Record_desc [dsc$w_length];
3
3    !
3    ! Search the record for a '!'.
3    !
3   Delim_position = CH$FIND_CH ( .context_length, .context_pointer, %c'!' );
3
3    !
3    ! If the '!' is in the first character position then this is a comment,
3    ! so return to get another record. If it is not in the first position
3    ! then reset the record length to the '!' position.
3    !
3   If .Delim_position NEQ 0 then
4    Begin
4    Offset = CH$DIFF ( .delim_position , .context_pointer);
4    If .Offset LEQ 3 then              ! If "!" is in the first three positions
4     Return false                      !  get the next record
4    else                               !  else adjust the context length
4     Context_length = .Offset - 1 ;    !  (Record length )
3    End;
3
3   Offset = 0;                         ! reset to zero for counting characters.
3
3    !
3    ! Search the record for a ' '. If a blank then compress record.
3    !
3   Delim_position = 1;
3   While .Delim_position NEQ 0 do
4    Begin
4    Delim_position = CH$FIND_CH (.context_length,.context_pointer,%C' ');
4    If .Delim_position EQL 0 then
4      Delim_position = CH$FIND_CH (.context_length,.context_pointer,TAB);
4    If .Delim_position NEQ 0 then
5      Begin
5      offset = .offset + 1;      ! Count number of characters removed
5      Temp_ptr = CH$DIFF(.delim_position+1, .context_pointer);
5      Length_to_move = .context_length - .temp_ptr ;
5      Temp_ptr = CH$COPY (.length_to_move, .delim_position + 1, %C'*',
5                          .length_to_move + 1, .delim_position);
4      End;
3    End;
3
3    Context_length = .context_length - .offset;
3
```

```
3     !
3     ! Search the record for '='. If an equal sign is found get index and value.
3     !
3
3     Delim_position = CH$FIND_CH (.context_length,.context_pointer,%C'=');
3     If .delim_position NEQ 0 then
4       Begin
4       Status = LIB$CVT_DTB ( (.delim_position - .context_pointer),  ! Calculate index field length
4                               .context_pointer,                   ! Start of index string
4                               .index );
4       If NOT .status then Signal (erf_cvterr, 2,
4                           (.delim_position - .context_pointer),context_pointer) ;
4       Context_length = .context_length - (.delim_position - .context_pointer);
3       End;
2    End;
2
2   Temp_ptr = 0;            ! Clear pointer
2
2     !
2     ! Get the value_addr, the size of the value field and
2     ! if no commas are found set context to 0.
2     !
2
2   .Value_addr = .delim_position + 1;
2
2   Temp_ptr = CH$FIND_CH (.context_length-1, .delim_position + 1, %C',');
2   If .temp_ptr EQL 0 then
3     Begin
3     If ..context EQL 1 then
3       .Size = .context_length - 1
3     else
3       .Size = .context_length - ( CH$DIFF (.delim_position, .context_pointer) );
3     .Context = 0;
3     End
2   else
3     Begin
3     .Size = CH$DIFF (.temp_ptr, .delim_position);
3     Context_length = .context_length - ..size;
3     .Size = ..size - 1;
3     Delim_position = .temp_ptr;
3     .Context = 1;
2     End;
2
2   Return true ;
1   End ;


                                                         .PSECT  $OWN$,NOEXE,  PIC,2

                                         00084 CONTEXT_LENGTH:
                                                         .BLKB   4
                                         00088 CONTEXT_POINTER:
                                                         .BLKB   4
                               00000000  0008C DELIM_POSITION:
                                                         .LONG   0                                          ;
                                         00090 LENGTH_TO_MOVE:
```

```
                                                              .BLKB   4
                                        00094 OFFSET:   .BLKB   4
                                        00098 STATUS:   .BLKB   4
                                        0009C TEMP_PTR:
                                                        .BLKB   4


                                                        .PSECT  $CODE,NOWRT,  PIC,2

                                   07FC 00000 PARSE_TEXT_RECORD:
                                                        .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10                ; 2697
                     5A 00000000'    00   9E 00002      MOVAB   DELIM_POSITION, R10                             ;
                     59        04    AC   D0 00009      MOVL    CONTEXT, R9                                     ; 2760
                                     69   D5 0000D      TSTL    (R9)                                            ;
                                     03   13 0000F      BEQL    1$                                              ;
                                   00E1   31 00011      BRW     12$                                             ;
                FC   AA 00000000'    00   D0 00014 1$:  MOVL    RECORD_DESC+4, CONTEXT_POINTER                  ; 2766
                F8   AA 00000000'    00   3C 0001C      MOVZWL  RECORD_DESC, CONTEXT_LENGTH                     ; 2767
                          57   FC    AA   D0 00024      MOVL    CONTEXT_POINTER, R7                             ; 2772
          67    F8   AA              21   3A 00028      LOCC    #33, CONTEXT_LENGTH, (R7)                       ;
                                     02   12 0002D      BNEQ    2$                                              ;
                                     51   D4 0002F      CLRL    R1                                              ;
                          6A         51   D0 00031 2$:  MOVL    R1, DELIM_POSITION                              ;
                          50         6A   D0 00034      MOVL    DELIM_POSITION, R0                              ; 2779
                                     16   13 00037      BEQL    4$                                              ;
     08   AA              50         57   C3 00039      SUBL3   R7, R0, OFFSET                                  ; 2781
                          50   08    AA   D0 0003E      MOVL    OFFSET, R0                                      ; 2782
                          03         50   D1 00042      CMPL    R0, #3                                          ;
                                     03   14 00045      BGTR    3$                                              ;
                                   0100   31 00047      BRW     18$                                             ;
                F8   AA        FF    A0   9E 0004A 3$:  MOVAB   -1(R0), CONTEXT_LENGTH                          ; 2785
                               08    AA   D4 0004F 4$:  CLRL    OFFSET                                          ; 2788
                          6A         01   D0 00052      MOVL    #1, DELIM_POSITION                              ; 2793
                          56         6A   D0 00055      MOVL    DELIM_POSITION, R6                              ; 2794
                          58   F8    AA   D0 00058      MOVL    CONTEXT_LENGTH, R8                              ; 2796
                                     56   D5 0005C 5$:  TSTL    R6                                              ; 2794
                                     47   13 0005E      BEQL    9$                                              ;
          67              58         20   3A 00060      LOCC    #32, R8, (R7)                                   ; 2796
                                     02   12 00064      BNEQ    6$                                              ;
                                     51   D4 00066      CLRL    R1                                              ;
                          6A         51   D0 00068 6$:  MOVL    R1, DELIM_POSITION                              ;
                                     0B   12 0006B      BNEQ    8$                                              ; 2797
          67              58         09   3A 0006D      LOCC    #9, R8, (R7)                                    ; 2798
                                     02   12 00071      BNEQ    7$                                              ;
                                     51   D4 00073      CLRL    R1                                              ;
                          6A         51   D0 00075 7$:  MOVL    R1, DELIM_POSITION                              ;
                          56         6A   D0 00078 8$:  MOVL    DELIM_POSITION, R6                              ; 2799
                                     DF   13 0007B      BEQL    5$                                              ;
                               08    AA   D6 0007D      INCL    OFFSET                                          ; 2801
          50              56         57   C3 00080      SUBL3   R7, R6, R0                                      ; 2802
                     10   AA   01    A0   9E 00084      MOVAB   1(R0), TEMP_PTR                                 ;
     04   AA              58   10    AA   C3 00089      SUBL3   TEMP_PTR, R8, LENGTH_TO_MOVE                    ; 2803
                          51   04    AA   D0 0008F      MOVL    LENGTH_TO_MOVE, R1                              ; 2804
                          50         56   D0 00093      MOVL    R6, R0                                          ;
                          52   01    A1   9E 00096      MOVAB   1(R1), R2                                       ; 2805
52        2A         01   A0         51   2C 0009A      MOVC5   R1, 1(R0), #42, R2, (R6)                        ;
```

```
                                                        66    000A0                                                               ;
                                    10    AA            53 D0 000A1        MOVL    R3, TEMP_PTR                                   ;
                                                        B5 11 000A5        BRB     5$                                             ; 2794
                                    F8    AA       08   AA C2 000A7 9$:    SUBL2   OFFSET, CONTEXT_LENGTH                         ; 2809
                           67       F8    AA            3D 3A 000AC        LOCC    #61, CONTEXT_LENGTH, (R7)                      ; 2815
                                                        02 12 000B1        BNEQ    10$                                            ;
                                                        51 D4 000B3        CLRL    R1                                             ;
                                          6A            51 D0 000B5 10$:   MOVL    R1, DELIM_POSITION                             ;
                                          50            6A D0 000B8        MOVL    DELIM_POSITION, R0                             ; 2816
                                                        38 13 000BB        BEQL    12$                                            ;
                                                   08   AC DD 000BD        PUSHL   INDEX                                          ; 2820
                                                        57 DD 000C0        PUSHL   R7                                             ; 2819
                           7E             50            57 C3 000C2        SUBL3   R7, R0, -(SP)                                  ; 2818
                             00000000G    00            03 FB 000C6        CALLS   #3, LIB$CVT_DTB                                ;
                                    0C    AA            50 D0 000CD        MOVL    R0, STATUS                                     ;
                                          17       0C   AA E8 000D1        BLBS    STATUS, 11$                                    ; 2821
                                                   FC   AA 9F 000D5        PUSHAB  CONTEXT_POINTER                                ;
                           7E             6A       FC   AA C3 000D8        SUBL3   CONTEXT_POINTER, DELIM_POSITION, -(SP)         ; 2822
                                                        02 DD 000DD        PUSHL   #2                                             ; 2821
                                            00000000G   8F DD 000DF        PUSHL   #ERF_CVTERR                                    ;
                             00000000G    00            04 FB 000E5        CALLS   #4, LIB$SIGNAL                                 ;
                           50       FC    AA            6A C3 000EC 11$:   SUBL3   DELIM_POSITION, CONTEXT_POINTER, R0            ; 2823
                                    F8    AA            50 C0 000F1        ADDL2   R0, CONTEXT_LENGTH                             ;
                                                   10   AA D4 000F5 12$:   CLRL    TEMP_PTR                                       ; 2827
                                          52            6A D0 000F8        MOVL    DELIM_POSITION, R2                             ; 2834
                           0C       BC             01   A2 9E 000FB        MOVAB   1(R2), @VALUE_ADDR                             ;
                                          55       F8   AA D0 00100        MOVL    CONTEXT_LENGTH, R5                             ; 2836
                                          54       FF   A5 9E 00104        MOVAB   -1(R5), R4                                     ;
                      01   A2             54            2C 3A 00108        LOCC    #44, R4, 1(R2)                                 ;
                                                        02 12 0010D        BNEQ    13$                                            ;
                                                        51 D4 0010F        CLRL    R1                                             ;
                                    10    AA            51 D0 00111 13$:   MOVL    R1, TEMP_PTR                                   ;
                                          50       10   AC D0 00115        MOVL    SIZE, R0                                       ; 2840
                                          53       10   AA D0 00119        MOVL    TEMP_PTR, R3                                   ; 2837
                                                        17 12 0011D        BNEQ    16$                                            ;
                                          01            69 D1 0011F        CMPL    (R9), #1                                       ; 2839
                                                        05 12 00122        BNEQ    14$                                            ;
                                          60            54 D0 00124        MOVL    R4, (R0)                                       ; 2840
                                                        09 11 00127        BRB     15$                                            ;
                           51       FC    AA            52 C3 00129 14$:   SUBL3   R2, CONTEXT_POINTER, R1                        ; 2842
                           60             51            55 C1 0012E        ADDL3   R5, R1, (R0)                                   ;
                                                        69 D4 00132 15$:   CLRL    (R9)                                           ; 2843
                                                        10 11 00134        BRB     17$                                            ; 2837
                           60             53            52 C3 00136 16$:   SUBL3   R2, R3, (R0)                                   ; 2847
                                    F8    AA            60 C2 0013A        SUBL2   (R0), CONTEXT_LENGTH                           ; 2848
                                                        60 D7 0013E        DECL    (R0)                                           ; 2849
                                          6A            53 D0 00140        MOVL    R3, DELIM_POSITION                             ; 2850
                                          69            01 D0 00143        MOVL    #1, (R9)                                       ; 2851
                                          50            01 D0 00146 17$:   MOVL    #1, R0                                         ; 2854
                                                           04 00149        RET                                                    ;
                                                        50 D4 0014A 18$:   CLRL    R0                                             ; 2855
                                                           04 0014C        RET                                                    ;

; Routine Size:  333 bytes,    Routine Base:  $CODE + 11B3
; 2314             2856  1 Routine INIT_COMMONS  =
; 2315             2857  2 Begin
; 2316             2858  2
; 2317             2859  2 !++
; 2318             2860  2 !
; 2319             2861  2 ! Functional Description:
; 2320             2862  2 !
; 2321             2863  2 !       This routine initializes some of the commons in
; 2322             2864  2 !       ERFSHR (qiocommon, opcodes, modes).
; 2323             2865  2 !
; 2324             2866  2 ! Calling Sequence:
; 2325             2867  2 !
; 2326             2868  2 !       Init_commons ()
; 2327             2869  2 !
; 2328             2870  2 ! Input parameters
; 2329             2871  2 !
; 2330             2872  2 !       None
; 2331             2873  2 !
; 2332             2874  2 ! Output parameters
; 2333             2875  2 !
; 2334             2876  2 !       None
; 2335             2877  2 !
; 2336             2878  2 !--
; 2337             2879  2
; 2338             2880  2 LOCAL
; 2339             2881  2     Array_addr,
; 2340             2882  2     Array_size,
; 2341             2883  2     Status,
; 2342             2884  2     Xfer_addr ;
; 2343             2885  2
; 2344             2886  2
; 2345             2887  2 !
; 2346             2888  2 ! Get the image name and attempt to load it.
; 2347             2889  2 ! Determine if a loading error occurred and signal it
; 2348             2890  2 ! if necessary.
; 2349             2891  2 !
; 2350             2892  2 Status = Map_image ( AD ('SYS$SYSTEM:ERFINICOM.EXE'), xfer_addr) ;
; 2351             2893  2 If NOT .status then return false ;
; 2352             2894  2
; 2353             2895  2 !
; 2354             2896  2 ! Execute the image. Then set the flag indicateing that the commons have been
; 2355             2897  2 ! initialized.
; 2356             2898  2 !
; 2357             2899  2 EXEC_IMAGE (xfer_addr) ;
; 2358             2900  2
; 2359             2901  2 Inited_commons = true ;
; 2360             2902  2
; 2361             2903  2 Return true ;
; 2362             2904  1 End ;           ! Routine


                                                                .PSECT  $PLIT,NOWRT,NOEXE,  PIC,2

49  46  52  45  3A  4D  45  54  53  59  53  24  53  59  53  001D4 P.ABV:  .ASCII  \SYS$SYSTEM:ERFINICOM.EXE\         ;
                        45  58  45  2E  4D  4F  43  49  4E  001E3                                                      ;
                                             00000018  001EC P.ABU:  .LONG   24                                       ;
```

```
                                          00000000' 001F0        .ADDRESS P.ABV                                              ;

                                                                 .PSECT  $CODE,NOWRT,  PIC,2

                                              0000 00000 INIT_COMMONS:
                                                                 .WORD   Save nothing                                        ; 2856
                     5E              04   C2 00002               SUBL2   #4, SP                                              ;
                                     5E   DD 00005               PUSHL   SP                                                  ; 2892
                           00000000' 00   9F 00007               PUSHAB  P.ABU                                               ;
           00000000G  00             02   FB 0000D               CALLS   #2, MAP_IMAGE                                       ;
                      14             50   E9 00014               BLBC    STATUS, 1$                                          ; 2893
                                     5E   DD 00017               PUSHL   SP                                                  ; 2899
           00000000G  00             01   FB 00019               CALLS   #1, EXEC_IMAGE                                      ;
           00000000'  00             01   D0 00020               MOVL    #1, INITED_COMMONS                                  ; 2901
                      50             01   D0 00027               MOVL    #1, R0                                              ; 2903
                                          04 0002A               RET                                                         ;
                      50                  D4 0002B 1$:           CLRL    R0                                                  ; 2904
                                          04 0002D               RET                                                         ;

; Routine Size:  46 bytes,    Routine Base:  $CODE + 1300

; 2363          2905  1
```

```
1 Global routine VALIDATE_PACKET =
2 BEGIN
2 !++
2 ! Functional description
2 !
2 !               This routine checks the error log packet entry type
2 !               to see if the type is valid for the CPU type it was
2 !               logged on.
2 !
2 ! Calling sequence
2 !
2 !       Validate_packet()
2 !
2 ! Input parameters
2 !
2 !
2 ! Output parameters
2 !
2 !       If valid_cpu, valid_class, valid_type or valid_entry are false
2 !       then return false.
2 !
2 ! Routine value
2 !
2 !       Worst error is returned.
2 !
2 !----
2
2 ! THERE SHOULD BE A MAX_ENTRY_TYPE.
2
2 GLOBAL
2       Processor_type:         LONG,
2       Device_class:           BYTE,
2       Device_type:            BYTE ;
2
2 LOCAL
2       Table_size:             WORD INITIAL (0),
2       Max_value:              BYTE,
2       Min_range:              REF VECTOR[,WORD],
2       Max_range:              REF VECTOR[,WORD],
2       Begin_bit_pos:          LONG INITIAL (24),
2       Version:                REF VECTOR[,WORD],
2       Field_size:             LONG INITIAL (8);
```

```
2 !
2 ! Set default state for valid flags
2 !
2 Syecom[sye$b_Valid_CPU] = true;
2 Syecom[sye$b_Valid_class] = true;
2
2 !
2 ! Obtain processor type. Determine if the processor type in the SID has
2 ! been set up. (Early VAX systems did not have the processor type set).
2 !
2 Processor_type = LIB$EXTZV ( Begin_bit_pos, Field_size, emb[emb$l_hd_sid] );
2
2 If .processor_type EQLU 255 then processor_type = 1;
2
2
2 !
2 ! Depending on processor type, determine which set of tables to use.
2 ! These tables specify invalid entry type ranges for specific cpu's.
2 !
2 Incr loop_count from 1 to .max_cpu_types do
2  If .processor_type_table[.loop_count] EQL .Processor_type then
3    Begin
3    Min_range = .Min_range_table_addr[.loop_count];
3    Max_range = .Max_range_table_addr[.loop_count];
3    Table_size = .Min_max_table_sizes[.loop_count];
3    Exitloop;
2    End;
2
2 If .table_size EQL 0 then syecom[sye$b_Valid_CPU] = False;
```

```
2 !
2 ! Ensure processing a device type entry.
2 !
2 If DEVICE_TYPE_ENTRY ()
2 Then
3        Begin
3         !
3         ! Determine the type of device entry and set up device class
3         ! and type from the appropriate fields in the EMB buffer.
3         !
3        Selectoneu .emb[emb$w_hd_entry] of
3            Set
3            [EMB$C_DE, EMB$C_DT, EMB$C_DA]:
4                Begin
4                Device_class = .emb[emb$b_dv_class] ;
4                Device_type = .emb[emb$b_dv_type] ;
3                End ;
3
3            [EMB$C_LM]:
4                Begin
4                Device_class = .emb[emb$b_lm_class] ;
4                Device_type = .emb[emb$b_lm_type] ;
3                End ;
3
3            [EMB$C_SP]:
4                Begin
4                Device_class = .emb[emb$b_sp_class] ;
4                Device_type = .emb[emb$b_sp_type] ;
3                End ;
3
3            [EMB$K_LOGMSCP]:
4                Begin
4                If CH$EQL (2,emb[driver_type],2,CH$PTR(uplit('DISK')))
4                Then
5                    Begin
5                    Device_class = DC$_DISK ;
5                    Device_type = 1 ;
4                    End ;
4
4                If CH$EQL (2,emb[driver_type],2,CH$PTR(uplit('TAPE')))
4                Then
5                    Begin
5                    Device_class = DC$_TAPE ;
5                    Device_type = 1 ;
4                    End ;
3                End ;
3
3            Tes ;
3
3         !
3         ! Determine the device class and set up the maximum number
3         ! of device types.
3         ! If class is out of range then VALID_class = False
3         !
3         Selectoneu .device_class of
3            Set
3
```

```
3               [DC$_DISK]:     ! Disk
4                       BEGIN
4                       Max_value = .max_disk_type;
4                       Version = .disk_version;
3                       END;
3
3               [DC$_TAPE]:     ! Tape
4                       BEGIN
4                       Max_value = .max_tape_type;
4                       Version = .tape_version;
3                       END;
3
3               [DC$_SCOM]:     ! Scom
4                       BEGIN
4                       Max_value = .max_scom_type;
4                       Version = .scom_version;
3                       END;
3
3               [DC$_LP]:       ! Printers
4                       BEGIN
4                       Max_value = .max_lp_type;
4                       Version = .lp_version;
3                       END;
3
3               [DC$_REALTIME]: ! Realtime
4                       BEGIN
4                       Max_value = .max_realtime_type;
4                       Version = .realtime_version;
3                       END;
3
3               [DC$_BUS]:      ! Buses
4                       BEGIN
4                       Max_value = .max_bus_type;
4                       Version = .bus_version;
3                       END;
3
3               [DC$_WORKSTATION]:      ! Workstations
4                       BEGIN
4                       Max_value = .max_workstation_type;
4                       Version = .workstation_version;
3                       END;
3
3               [OTHERWISE]:
4                 Begin
4                   Max_value = 0;
4                   Version = 0;
4                   Syecom[sye$b_Valid_class] = false;
3                 End;
3
3               TES;
3
3       !
3       ! If device type is less then 1 or greater then max
3       ! value or the version number is zero, then set flags false.
3       !
3               If ( .device_type LSSU  1 ) OR
3                  ( .device_type GTRU .max_value ) OR
```

```
4           ( .version EQLU 0 )
3         Then
3           Syecom[sye$b_Valid_type] = false
3         Else
3           If .version[.device_type] EQLU 0
3           then
3             Syecom[sye$b_Valid_type] = false
3           else
3             Syecom[sye$b_Valid_type] = true;
3       End
2   Else
2       Syecom[sye$b_valid_type] = true ;
2
2
2   !
2   ! Ensure a valid cpu type was found. Otherwise don't attempt
2   ! to do entry type verification.
2   !
2   If .syecom[sye$b_valid_cpu]
2   Then
3       Begin
3       Incr I from 1 to .table_size do
4           Begin
4
4           If ( .emb[emb$w_hd_entry] GEQU .min_range[.I] ) AND
5              ( .emb[emb$w_hd_entry] LEQU .max_range[.I] )
4           then
5               Begin
5               Syecom[sye$b_Valid_entry] = false ;
5               Exitloop;
5               End
4           else
4               Syecom[sye$b_Valid_entry] = true ;
3           End;
2       End ;
2
2   If NOT .syecom[sye$b_valid_cpu] OR
2      NOT .syecom[sye$b_valid_class] OR
2      NOT .syecom[sye$b_valid_type] OR
2      NOT .syecom[sye$b_valid_entry]
2   Then return false;
2
2   Return true;
1   End;


                                                          .PSECT  $PLIT,NOWRT,NOEXE,  PIC,2

                          4B  53  49  44  001F4 P.ABW:   .ASCII  \DISK\                              ;
                          45  50  41  54  001F8 P.ABX:   .ASCII  \TAPE\                              ;

                                                          .PSECT  $GLOBAL$,NOEXE,  PIC,2

                                    000C0 PROCESSOR_TYPE::
                                                          .BLKB   4
                                    000C4 DEVICE_CLASS::
```

```
                                                                   .BLKB    1
                                                 000C5 DEVICE_TYPE::
                                                                   .BLKB    1



                                                                   .PSECT   $CODE,NOWRT,  PIC,2

                                          01FC 00000               .ENTRY   VALIDATE_PACKET, Save R2,R3,R4,R5,R6,R7,R8          ; 2906
                    58 00000000G    00      9E 00002               MOVAB    EMB+4, R8                                           ;
                    57 00000000G    00      9E 00009               MOVAB    SYECOM+25, R7                                       ;
                    56 00000000'    00      9E 00010               MOVAB    DEVICE_CLASS, R6                                    ;
                                    52      B4 00017               CLRW     TABLE_SIZE                                          ; 2907
                                    18      DD 00019               PUSHL    #24                                                 ;
                                    08      DD 0001B               PUSHL    #8                                                  ;
                    67       0101   8F      B0 0001D               MOVW     #257, SYECOM+25                                     ; 2952
                               FC   A8      9F 00022               PUSHAB   EMB                                                 ; 2958
                               04   AE      9F 00025               PUSHAB   FIELD_SIZE                                          ;
                               0C   AE      9F 00028               PUSHAB   BEGIN_BIT_POS                                       ;
          00000000G    00           03      FB 0002B               CALLS    #3, LIB$EXTZV                                       ;
                 FC    A6           50      D0 00032               MOVL     R0, PROCESSOR_TYPE                                  ;
          000000FF     8F      FC   A6      D1 00036               CMPL     PROCESSOR_TYPE, #255                                ; 2960
                                    04      12 0003E               BNEQ     1$                                                  ;
                 FC    A6           01      D0 00040               MOVL     #1, PROCESSOR_TYPE                                  ;
                       53      86   A6      3C 00044 1$:           MOVZWL   MAX_CPU_TYPES, R3                                   ; 2967
                                    50      D4 00048               CLRL     LOOP_COUNT                                          ; 2968
                                    29      11 0004A               BRB      3$                                                  ;
                       51      AC   A6      D0 0004C 2$:           MOVL     PROCESSOR_TYPE_TABLE, R1                            ;
                                  6140      3F 00050               PUSHAW   (R1)[LOOP_COUNT]                                    ;
    FC    A6          9E          10   00   ED 00053               CMPZV    #0, #16, @(SP)+, PROCESSOR_TYPE                     ;
                                    1A      12 00059               BNEQ     3$                                                  ;
                       51      A0   A6      D0 0005B               MOVL     MIN_RANGE_TABLE_ADDR, R1                            ; 2970
                       55         6140      D0 0005F               MOVL     (R1)[LOOP_COUNT], MIN_RANGE                         ;
                       51      8C   A6      D0 00063               MOVL     MAX_RANGE_TABLE_ADDR, R1                            ; 2971
                       54         6140      D0 00067               MOVL     (R1)[LOOP_COUNT], MAX_RANGE                         ;
                       51      9C   A6      D0 0006B               MOVL     MIN_MAX_TABLE_SIZES, R1                             ; 2972
                       52         6140      B0 0006F               MOVW     (R1)[LOOP_COUNT], TABLE_SIZE                        ;
                                    04      11 00073               BRB      4$                                                  ; 2969
                      D3       50   53      F3 00075 3$:           AOBLEQ   R3, LOOP_COUNT, 2$                                  ; 2968
                               53   52      3C 00079 4$:           MOVZWL   TABLE_SIZE, R3                                      ; 2976
                                    03      12 0007C               BNEQ     5$                                                  ;
                               01   A7      94 0007E               CLRB     SYECOM+26                                           ;
          00000000G    00           00      FB 00081 5$:           CALLS    #0, DEVICE_TYPE_ENTRY                               ; 2980
                       03           50      E8 00088               BLBS     R0, 6$                                              ;
                                  00FE      31 0008B               BRW      22$                                                 ;
                       50           68      3C 0008E 6$:           MOVZWL   EMB+4, R0                                           ; 2987
                       01           50      B1 00091               CMPW     R0, #1                                              ; 2989
                                    0E      13 00094               BEQL     7$                                                  ;
                0060   8F           50      B1 00096               CMPW     R0, #96                                             ;
                                    07      13 0009B               BEQL     7$                                                  ;
                0062   8F           50      B1 0009D               CMPW     R0, #98                                             ;
                                    06      12 000A2               BNEQ     8$                                                  ;
                       66      18   A8      B0 000A4 7$:           MOVW     EMB+28, DEVICE_CLASS                                ; 2991
                                    3B      11 000A8               BRB      12$                                                 ; 2987
                0064   8F           50      B1 000AA 8$:           CMPW     R0, #100                                            ; 2995
                                    07      13 000AF               BEQL     9$                                                  ;
                0063   8F           50      B1 000B1               CMPW     R0, #99                                             ; 3001
```

```
                          06 12 000B6            BNEQ    10$                                         ;
         66          0C   A8 B0 000B8 9$:        MOVW    EMB+16, DEVICE_CLASS                        ; 3003
                          27 11 000BC            BRB     12$                                         ; 2987
   0065  8F               50 B1 000BE 10$:       CMPW    R0, #101                                    ; 3007
                          20 12 000C3            BNEQ    12$                                         ;
         50          0E   A8 3C 000C5            MOVZWL  EMB+18, R0                                  ; 3009
         50 00000000'     00 B1 000C9            CMPW    P.ABW, R0                                   ;
                          05 12 000D0            BNEQ    11$                                         ;
         66        0101   8F B0 000D2            MOVW    #257, DEVICE_CLASS                          ; 3012
         50 00000000'     00 B1 000D7 11$:       CMPW    P.ABX, R0                                   ; 3016
                          05 12 000DE            BNEQ    12$                                         ;
         66        0102   8F B0 000E0            MOVW    #258, DEVICE_CLASS                          ; 3019
         50               66 9A 000E5 12$:       MOVZBL  DEVICE_CLASS, R0                            ; 3031
         01               50 91 000E8            CMPB    R0, #1                                      ; 3034
                          0D 12 000EB            BNEQ    13$                                         ;
         52          88   A6 90 000ED            MOVB    MAX_DISK_TYPE, MAX_VALUE                    ; 3036
         51 00000000G     00 D0 000F1            MOVL    DISK_VERSION, VERSION                       ; 3037
                          79 11 000F8            BRB     20$                                         ; 3031
         02               50 91 000FA 13$:       CMPB    R0, #2                                      ; 3040
                          0D 12 000FD            BNEQ    14$                                         ;
         52          92   A6 90 000FF            MOVB    MAX_TAPE_TYPE, MAX_VALUE                    ; 3042
         51 00000000G     00 D0 00103            MOVL    TAPE_VERSION, VERSION                       ; 3043
                          67 11 0010A            BRB     20$                                         ; 3031
         20               50 91 0010C 14$:       CMPB    R0, #32                                     ; 3046
                          0D 12 0010F            BNEQ    15$                                         ;
         52          91   A6 90 00111            MOVB    MAX_SCOM_TYPE, MAX_VALUE                    ; 3048
         51 00000000G     00 D0 00115            MOVL    SCOM_VERSION, VERSION                       ; 3049
                          55 11 0011C            BRB     20$                                         ; 3031
     43  8F               50 91 0011E 15$:       CMPB    R0, #67                                     ; 3052
                          10 12 00122            BNEQ    16$                                         ;
         52 00000000G     00 90 00124            MOVB    MAX_LP_TYPE, MAX_VALUE                      ; 3054
         51 00000000G     00 D0 0012B            MOVL    LP_VERSION, VERSION                         ; 3055
                          3F 11 00132            BRB     20$                                         ; 3031
     60  8F               50 91 00134 16$:       CMPB    R0, #96                                     ; 3058
                          0D 12 00138            BNEQ    17$                                         ;
         52          90   A6 90 0013A            MOVB    MAX_REALTIME_TYPE, MAX_VALUE                ; 3060
         51 00000000G     00 D0 0013E            MOVL    REALTIME_VERSION, VERSION                   ; 3061
                          2C 11 00145            BRB     20$                                         ; 3031
     80  8F               50 91 00147 17$:       CMPB    R0, #128                                    ; 3064
                          0D 12 0014B            BNEQ    18$                                         ;
         52          84   A6 90 0014D            MOVB    MAX_BUS_TYPE, MAX_VALUE                     ; 3066
         51 00000000G     00 D0 00151            MOVL    BUS_VERSION, VERSION                        ; 3067
                          19 11 00158            BRB     20$                                         ; 3031
     46  8F               50 91 0015A 18$:       CMPB    R0, #70                                     ; 3070
                          0D 12 0015E            BNEQ    19$                                         ;
         52          93   A6 90 00160            MOVB    MAX_WORKSTATION_TYPE, MAX_VALUE             ; 3072
         51 00000000G     00 D0 00164            MOVL    WORKSTATION_VERSION, VERSION                ; 3073
                          06 11 0016B            BRB     20$                                         ; 3031
                          52 94 0016D 19$:       CLRB    MAX_VALUE                                   ; 3078
                          51 D4 0016F            CLRL    VERSION                                     ; 3079
                          67 94 00171            CLRB    SYECOM+25                                   ; 3080
         50          01   A6 9A 00173 20$:       MOVZBL  DEVICE_TYPE, R0                             ; 3089
                          0E 13 00177            BEQL    21$                                         ;
         50               52 91 00179            CMPB    MAX_VALUE, R0                               ; 3090
                          09 1F 0017C            BLSSU   21$                                         ;
                          51 D5 0017E            TSTL    VERSION                                     ; 3091
                          05 13 00180            BEQL    21$                                         ;
```

```
                                      6140  B5 00182          TSTW     (VERSION)[R0]                              ; 3095
                                        05  12 00185          BNEQ     22$                                        ;
                                 03     A7  94 00187 21$:     CLRB     SYECOM+28                                  ; 3097
                                        04  11 0018A          BRB      23$                                        ;
                         03     A7      01  90 0018C 22$:     MOVB     #1, SYECOM+28                              ; 3102
                                37      01  A7  E9 00190 23$: BLBC     SYECOM+26, 28$                             ; 3109
                                51      68  3C 00194          MOVZWL   EMB+4, R1                                  ; 3115
                                        50  D4 00197          CLRL     I                                          ;
                                        19  11 00199          BRB      26$                                        ;
                                      6540  3F 0019B 24$:     PUSHAW   (MIN_RANGE)[I]                             ;
          51            9E      10      00  ED 0019E          CMPZV    #0, #16, @(SP)+, R1                        ;
                                        0B  1A 001A3          BGTRU    25$                                        ;
                                51    6440  B1 001A5          CMPW     (MAX_RANGE)[I], R1                         ; 3116
                                        05  1F 001A9          BLSSU    25$                                        ;
                                 02     A7  94 001AB          CLRB     SYECOM+27                                  ; 3119
                                        08  11 001AE          BRB      27$                                        ; 3118
                         02     A7      01  90 001B0 25$:     MOVB     #1, SYECOM+27                              ; 3123
                        E3      50      53  F3 001B4 26$:     AOBLEQ   R3, I, 24$                                 ; 3112
                                0F      01  A7  E9 001B8 27$: BLBC     SYECOM+26, 28$                             ; 3127
                                0C      67  E9 001BC          BLBC     SYECOM+25, 28$                             ; 3128
                                08      03  A7  E9 001BF      BLBC     SYECOM+28, 28$                             ; 3129
                                04      02  A7  E9 001C3      BLBC     SYECOM+27, 28$                             ; 3130
                                50      01  D0 001C7          MOVL     #1, R0                                     ; 3133
                                        04 001CA              RET                                                 ;
                                        50  D4 001CB 28$:     CLRL     R0                                         ; 3134
                                        04 001CD              RET                                                 ;

; Routine Size:  462 bytes,    Routine Base:  $CODE + 132E
; 2597      3135 1 Routine HANDLER (sig, mech ) =
; 2598      3136 1
; 2599      3137 1 !---
; 2600      3138 1 !
; 2601      3139 1 !        This condition handler gets control on any signalled
; 2602      3140 1 !        condition in order to save the highest severity error
; 2603      3141 1 !        to be returned by exit from the image.
; 2604      3142 1 !
; 2605      3143 1 ! Inputs:
; 2606      3144 1 !
; 2607      3145 1 !        signal_args = Address of signal argument list
; 2608      3146 1 !        mechanism_args = Address of mechanism argument list
; 2609      3147 1 !
; 2610      3148 1 ! Outputs:
; 2611      3149 1 !
; 2612      3150 1 !        WORST_ERROR is updated with highest severity error.
; 2613      3151 1 !
; 2614      3152 1 !---
; 2615      3153 1
; 2616      3154 2 BEGIN
; 2617      3155 2
; 2618      3156 2 External worst_error: $BBLOCK [LONG] ;  ! Holds worst error encountered
; 2619      3157 2
; 2620      3158 2 MAP                                     ! Standard VMS condition handler parameters.
; 2621      3159 2     sig:        REF $BBLOCK,            ! Address of signal argument list
; 2622      3160 2     mech:       REF $BBLOCK;            ! Address of mechanism argument list
; 2623      3161 2
; 2624      3162 2 BIND
; 2625      3163 2     COND = SIG[CHF$L_SIG_NAME]: $BBLOCK ;! Condition
; 2626      3164 2
; 2627      3165 2
; 2628      3166 2
; 2629      3167 2 If .COND eql RMS$_EOF then return true;
; 2630      3168 2
; 2631      3169 2 If .cond[sts$v_fac_no] eql erf$_facility then
; 2632      3170 2    return ss$_resignal;
; 2633      3171 2
; 2634      3172 2 If
; 2635      3173 2    .cond[sts$v_severity] gtru .worst_error [sts$v_severity]
; 2636      3174 2 then
; 2637      3175 2     worst_error = .cond or sts$m_inhib_msg;
; 2638      3176 2
; 2639      3177 2 sig[chf$l_sig_args] = .sig[chf$l_sig_args] - 2; ! Dont count pc/psl
; 2640      3178 2 $putmsg ( msgvec = sig[chf$l_sig_args], actrtn = write_err_msg);
; 2641      3179 2 sig[chf$l_sig_args] = .sig[chf$l_sig_args] + 2;
; 2642      3180 2
; 2643      3181 2 ss$_resignal                                    ! Continue signalling
; 2644      3182 2
; 2645      3183 1 END;



                                   000C 00000 HANDLER:.WORD    Save R2,R3                                       : 3135
                  53 00000000G  00  9E 00002         MOVAB    WORST_ERROR, R3                                  :
                  52         04  AC  D0 00009         MOVL     SIG, R2                                          : 3163
```

```
                       0001827A   8F        04  A2  D1 0000D          CMPL    4(R2), #98938                            ; 3167
                                                04  12 00015          BNEQ    1$                                       ;
                                 50             01  D0 00017          MOVL    #1, R0                                   ;
                                                    04 0001A          RET                                              ;
         08         06   A2                 0C  00  ED 0001B 1$:      CMPZV   #0, #12, 6(R2), #8                       ; 3169
                                                2D  13 00021          BEQL    3$                                       ;
         50              63                 03  00  EF 00023          EXTZV   #0, #3, WORST_ERROR, R0                  ; 3173
         50         04   A2                 03  00  ED 00028          CMPZV   #0, #3, 4(R2), R0                        ;
                                                09  1B 0002E          BLEQU   2$                                       ;
                         63         04  A2 10000000  8F  C9 00030     BISL3   #268435456, 4(R2), WORST_ERROR           ; 3175
                                 62             02  C2 00039 2$:      SUBL2   #2, (R2)                                 ; 3177
                                                7E  7C 0003C          CLRQ    -(SP)                                    ; 3178
                                  00000000V     00  9F 0003E          PUSHAB  WRITE_ERR_MSG                            ;
                                                52  DD 00044          PUSHL   R2                                       ;
                       00000000G  00            04  FB 00046          CALLS   #4, SYS$PUTMSG                           ;
                                 62             02  C0 0004D          ADDL2   #2, (R2)                                 ; 3179
                                 50       0918  8F  3C 00050 3$:      MOVZWL  #2328, R0                                ; 3183
                                                    04 00055          RET                                              ;

; Routine Size:  86 bytes,    Routine Base:  $CODE + 14FC
; 2647          3184   1 Routine WRITE_ERR_MSG (Error_msg_desc) =
; 2648          3185   1 !---
; 2649          3186   1 !
; 2650          3187   1 !        This routine writes the error message to the output file.
; 2651          3188   1 !
; 2652          3189   1 ! Inputs:
; 2653          3190   1 !
; 2654          3191   1 !        error_msg_desc = Address of descriptor for message
; 2655          3192   1 !
; 2656          3193   1 ! Outputs:
; 2657          3194   1 !
; 2658          3195   1 !
; 2659          3196   1 !---
; 2660          3197   2 Begin
; 2661          3198   2
; 2662          3199   2 Local
; 2663          3200   2          Rmserror;
; 2664          3201   2
; 2665          3202   2 Map
; 2666          3203   2          Error_msg_desc : REF BLOCK[,BYTE];
; 2667          3204   2
; 2668          3205   2 If .Lstlun_rab_address EQL 0 then return false;
; 2669          3206   2
; 2670          3207   2 Lstlun_rab_address[rab$l_rbf] = .error_msg_desc[dsc$a_pointer];
; 2671          3208   2 Lstlun_rab_address[rab$w_rsz] = .error_msg_desc[dsc$w_length];
; 2672          3209   2
; 2673          3210   2 If NOT (rmserror = $put(rab = .Lstlun_rab_address)) then
; 2674          3211   2   ( Signal (.rmserror); Return .rmserror);
; 2675          3212   2
; 2676          3213   2 Return false;
; 2677          3214   1 End;


                                                                       .EXTRN  SYS$PUT

                                                0004 00000 WRITE_ERR_MSG:
                                                                       .WORD   Save R2                                     ; 3184
                          51 00000000G  00  D0 00002                   MOVL    LSTLUN_RAB_ADDRESS, R1                      ; 3205
                                        29  13 00009                   BEQL    1$                                          ;
                          50        04  AC  D0 0000B                   MOVL    ERROR_MSG_DESC, R0                          ; 3207
                    28    A1        04  A0  D0 0000F                   MOVL    4(R0), 40(R1)                               ;
                    22    A1            60  B0 00014                   MOVW    (R0), 34(R1)                                ; 3208
                                        51  DD 00018                   PUSHL   R1                                          ; 3210
                 00000000G  00          01  FB 0001A                   CALLS   #1, SYS$PUT                                 ;
                            52          50  D0 00021                   MOVL    R0, RMSERROR                                ;
                            0D          52  E8 00024                   BLBS    RMSERROR, 1$                                ;
                                        52  DD 00027                   PUSHL   RMSERROR                                    ; 3211
                 00000000G  00          01  FB 00029                   CALLS   #1, LIB$SIGNAL                              ;
                            50          52  D0 00030                   MOVL    RMSERROR, R0                                ;
                                            04 00033                   RET                                                 ;
                                        50  D4 00034 1$:               CLRL    R0                                          ; 3214
                                            04 00036                   RET                                                 ;

; Routine Size:  55 bytes,    Routine Base:  $CODE + 1552
; 2679          3215  1 Global routine WRITE_BINARY (BUFFER, RAB) =
; 2680          3216  1
; 2681          3217  1 !----
; 2682          3218  1 !
; 2683          3219  1 ! Functional description
; 2684          3220  1 !
; 2685          3221  1 !       This routine accepts a pointer to a buffer and writes
; 2686          3222  1 !       the buffer to an output stream in binary format.
; 2687          3223  1 !
; 2688          3224  1 ! Input parameters
; 2689          3225  1 !
; 2690          3226  1 !       BUFFER = address of an input record buffer
; 2691          3227  1 !
; 2692          3228  1 !       RAB =    address of output rab
; 2693          3229  1 !
; 2694          3230  1 !----
; 2695          3231  1
; 2696          3232  2 BEGIN
; 2697          3233  2
; 2698          3234  2 MAP
; 2699          3235  2    rab:         ref $bblock,             ! Pointer to rab
; 2700          3236  2    buffer:      ref $bblock;             ! Describe the input buffer
; 2701          3237  2
; 2702          3238  2 LOCAL
; 2703          3239  2         desc:   vector [2, long];        ! Temporary string descriptor
; 2704          3240  2
; 2705          3241  2
; 2706          3242  2 If .rab eql 0 then return true;         ! Exit immediately if no output
; 2707          3243  2
; 2708          3244  2 !
; 2709          3245  2 !INITIALIZE THE RAB
; 2710          3246  2 !       Store the buffer address and length in the RAB.
; 2711          3247  2 !
; 2712          3248  2
; 2713          3249  2 rab [rab$l_rbf] = .buffer;                        ! Store buffer address in RAB
; 2714          3250  2 rab [rab$w_rsz] = .input_rab[rab$w_rsz];          ! Store buffer size in RAB
; 2715          3251  2
; 2716          3252  2
; 2717          3253  2
; 2718          3254  2 !
; 2719          3255  2 !WRITE TO FILE ---
; 2720          3256  2 !       Output the buffer via RMS.
; 2721          3257  2 !
; 2722          3258  2
; 2723        P 3259  2 CALL_FUNCTION  ($put (                ! Call RMS with
; 2724        P 3260  2                rab = .rab,            !  -record stream identifier
; 2725          3261  2                err = log_filename));  !  -error action routine
; 2726          3262  2
; 2727          3263  2 return true;
; 2728          3264  1 END;




                                     0000 00000         .ENTRY  WRITE_BINARY, Save nothing                           ; 3215
                        5E        08  C2 00002          SUBL2   #8, SP                                               ;
```

```
                         50          08   AC  D0 00005          MOVL    RAB, R0                                                 ; 3242
                                          1F  13 00009          BEQL    1$                                                      ;
                   28    A0          04   AC  D0 0000B          MOVL    BUFFER, 40(R0)                                          ; 3249
                   22    A0 00000000G     00  B0 00010          MOVW    INPUT_RAB+34, 34(R0)                                    ; 3250
                            00000000G     00  9F 00018          PUSHAB  LOG_FILENAME                                            ; 3261
                                          50  DD 0001E          PUSHL   R0                                                      ;
              00000000G  00               02  FB 00020          CALLS   #2, SYS$PUT                                             ;
                         03               50  E9 00027          BLBC    STATUS, 2$                                              ;
                         50               01  D0 0002A 1$:      MOVL    #1, R0                                                  ; 3263
                                              04 0002D 2$:      RET                                                             ; 3264

; Routine Size:  46 bytes,    Routine Base:  $CODE + 1589


; 2729          3265  1
; 2730          3266  1 END
; 2731          3267  0 ELUDOM



                                                                .EXTRN  LIB$SIGNAL, LIB$STOP

;                               PSECT SUMMARY
;
;       Name                       Bytes                      Attributes
;
;  $OWN$                              160  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;  $PLIT                              508  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;  $GLOBAL$                           198  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;  $CODE                             5559  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)


;                          Library Statistics
;
;                                        -------- Symbols --------        Pages        Processing
;       File                             Total    Loaded   Percent        Mapped       Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1         18619      115          0        1000          00:02.1




;                               COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:ERF/OBJ=OBJ$:ERF MSRC$:ERF/UPDATE=(ENH$:ERF)

; 2732          3268  0
; Size:         5559 code + 866 data bytes
; Run Time:        01:27.9
; Elapsed Time:    03:04.5
; Lines/CPU Min:    2230
```

```
ER
Sy

ER
ME
MN
PR
SB


PS
--

$$


Ph
--
In
Co
Pa
Sy
Pa
Sy
Ps
Cr
As

Th
60
Th
55
0


Ma
--
_$

0

Th

MA
```

```
; Lexemes/CPU-Min: 15987
; Memory Used:  356 pages
; Compilation Complete
```

0148 AH-BT13A-SE DIGITAL EQUIPMENT CORPORATION
VAX/VMS V4.0 CONFIDENTIAL AND PROPRIETARY