F 9

```
LLL                 IIIIIIIII   NNN         NNN  KKK         KKK  EEEEEEEEEEEEEEE  RRRRRRRRRRRR
LLL                 IIIIIIIII   NNN         NNN  KKK         KKK  EEEEEEEEEEEEEEE  RRRRRRRRRRRR
LLL                 IIIIIIIII   NNN         NNN  KKK         KKK  EEEEEEEEEEEEEEE  RRRRRRRRRRRR
LLL                    III      NNN         NNN  KKK         KKK  EEE              RRR         RRR
LLL                    III      NNN         NNN  KKK         KKK  EEE              RRR         RRR
LLL                    III      NNN         NNN  KKK         KKK  EEE              RRR         RRR
LLL                    III      NNNNNN      NNN  KKK      KKK     EEE              RRR         RRR
LLL                    III      NNNNNN      NNN  KKK      KKK     EEE              RRR         RRR
LLL                    III      NNNNNN      NNN  KKK      KKK     EEE              RRR         RRR
LLL                    III      NNN   NNN   NNN  KKKKKKKKK        EEEEEEEEEEEE     RRRRRRRRRRRR
LLL                    III      NNN   NNN   NNN  KKKKKKKKK        EEEEEEEEEEEE     RRRRRRRRRRRR
LLL                    III      NNN   NNN   NNN  KKKKKKKKK        EEEEEEEEEEEE     RRRRRRRRRRRR
LLL                    III      NNN      NNNNNN  KKK      KKK     EEE              RRR   RRR
LLL                    III      NNN      NNNNNN  KKK      KKK     EEE              RRR   RRR
LLL                    III      NNN      NNNNNN  KKK      KKK     EEE              RRR   RRR
LLL                    III      NNN         NNN  KKK         KKK  EEE              RRR      RRR
LLL                    III      NNN         NNN  KKK         KKK  EEE              RRR      RRR
LLL                    III      NNN         NNN  KKK         KKK  EEE              RRR      RRR
LLLLLLLLLLLLLLL     IIIIIIIII   NNN         NNN  KKK         KKK  EEEEEEEEEEEEEEE  RRR         RRR
LLLLLLLLLLLLLLL     IIIIIIIII   NNN         NNN  KKK         KKK  EEEEEEEEEEEEEEE  RRR         RRR
LLLLLLLLLLLLLLL     IIIIIIIII   NNN         NNN  KKK         KKK  EEEEEEEEEEEEEEE  RRR         RRR
```

**FILE**ID**LNKGETCMD

K 4

```
LL          NN      NN  KK      KK    GGGGGGGG  EEEEEEEEEE  TTTTTTTTTT    CCCCCCCC  MM      MM  DDDDDDDD
LL          NN      NN  KK      KK    GGGGGGGG  EEEEEEEEEE  TTTTTTTTTT    CCCCCCCC  MM      MM  DDDDDDDD
LL          NN      NN  KK      KK  GG          EE              TT      CC          MMMM  MMMM  DD      DD
LL          NN      NN  KK      KK  GG          EE              TT      CC          MMMM  MMMM  DD      DD
LL          NNNN    NN  KK    KK    GG          EE              TT      CC          MM  MM  MM  DD      DD
LL          NNNN    NN  KK    KK    GG          EE              TT      CC          MM  MM  MM  DD      DD
LL          NN  NN  NN  KKKKKK      GG          EEEEEEEE        TT      CC          MM      MM  DD      DD
LL          NN  NN  NN  KKKKKK      GG          EEEEEEEE        TT      CC          MM      MM  DD      DD
LL          NN    NNNN  KK    KK    GG  GGGGGG  EE              TT      CC          MM      MM  DD      DD
LL          NN    NNNN  KK    KK    GG  GGGGGG  EE              TT      CC          MM      MM  DD      DD
LL          NN      NN  KK      KK  GG      GG  EE              TT      CC          MM      MM  DD      DD    ....
LL          NN      NN  KK      KK  GG      GG  EE              TT      CC          MM      MM  DD      DD    ....
LLLLLLLLLL  NN      NN  KK      KK    GGGGGG    EEEEEEEEEE      TT        CCCCCCCC  MM      MM  DDDDDDDD      ....
LLLLLLLLLL  NN      NN  KK      KK    GGGGGG    EEEEEEEEEE      TT        CCCCCCCC  MM      MM  DDDDDDDD      ....

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
```

LNK
V04

```
module  lnk_getcmdinp
                (ident='V04-000'
                ,addressing_mode
                        (external    = general
                        ,nonexternal = long_relative
                        )
                ) =
begin
!
!
!****************************************************************************
!*                                                                          *
!*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
!*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
!*  ALL RIGHTS RESERVED.                                                    *
!*                                                                          *
!*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
!*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
!*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
!*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
!*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
!*  TRANSFERRED.                                                            *
!*                                                                          *
!*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
!*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
!*  CORPORATION.                                                            *
!*                                                                          *
!*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
!*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
!*                                                                          *
!*                                                                          *
!****************************************************************************
!
!
!
!
!
!++
!
! MODULE:         LNK_GETCMDINP
!
! FACILITY:       LINKER
!
! ABSTRACT:       Command Language Interface For The LINK Command
!
! HISTORY:
!
! VERSION:        X01.00
!
! AUTHOR:         T.J. PORTER 03-MAY-77
!
! MODIFIED BY:
!
!       V03-026 ADE0005         Alan D. Eldridge        24-Jul-1984
!               Save proper STV value on LIB$FIND_FILE error.
!
!       V03-025 ADE0004         Alan D. Eldridge        30-Jun-1984
```

```
1 !             Fix bug in /NOEXE.
1 !
1 !     V03-024 ADE0003         Alan D. Eldridge        22-Jun-1984
1 !             Add LNK$GB_LOCNOV_MAP, LNK$GB_LOCNOV_EXE, and LNK$GB_LOCNOV_SYM
1 !             for proper "Grammer Rule" support for constructing output file
1 !             names.  These are really flags which, when set, indicate that
1 !             appropriate switch had was used locally but had no value
1 !             associated with it
1 !
1 !     V03-023 ADE0002         Alan D. Eldridge        14-Apr-1984
1 !             More fixes to positional qualifier support.
1 !
1 !     V03-022 ADE0001         Alan D. Eldridge        4-Apr-1984
1 !             Convert routine INPUTFILE to use LIB$FIND_FILE rather than
1 !             $SEARCH and $PARSE.
1 !
1 !             Add support for proper defualting when /EXE, /MAP, or /SYM
1 !             are used as local, rather than command, qualifiers.
1 !
1 !     V03-021 MCN0001         Maria del C. Nasr       20-Jul-1983
1 !             Determine if the options file is being read from
1 !             SYS$INPUT or not.  This is needed so that there is
1 !             no duplication of input line on output when SET
1 !             VERIFY is on.
1 !
1 !     V03-020 JWT0125         Jim Teague              21-Jun-1983
1 !             Allow UNIV=* when accompanied by /NOEXE.  It can thus
1 !             be used only for STB generation from shareable image
1 !             links.
1 !
1 !     V03-019 JWT0119         Jim Teague              11-May-1983
1 !             Tweak new CLI stuff again... adjust map qualifiers.
1 !
1 !     V03-018 JWT0117         Jim Teague              29-Apr-1983
1 !             /NOEXE is equivalent to /EXE=NL:
1 !
1 !     V03-017 JWT0113         Jim Teague              20-Apr-1983
1 !             Call $getjpi to get number of open files left.
1 !
1 !     V03-016 JWT0108         Jim Teague              13-Apr-1983
1 !             More adjustments to new CLI interface.
1 !
1 !     V03-015 JWT0107         Jim Teague              07-Apr-1983
1 !             Fix problems with /MAP qualifier defaults.
1 !
1 !     V03-014 JWT0099         Jim Teague              03-Mar-1983
1 !             New cli interface.
1 !
1 !     V03-013 JWT0059         Jim Teague              19-Oct-1982
1 !             Corrected problem with basing system images explicitly.
1 !
1 !     V03-012 JWT0054         Jim Teague              15-Sep-1982
1 !             INPUTFILE is now able to more intelligently specify
1 !             a file to close.
1 !
1 !     V03-011 JWT0049         Jim Teague              06-Aug-1982
1 !             Call LNK$CLOSEFILE to close files, rather than $CLOSE.
```

```
1 !
1 !      V03-010 JWT0044        Jim Teague              30-Jul-1982
1 !              Open file performance boost.
1 !
1 !      V03-009 BLS0166        Benn Schreiber          29-Mar-1982
1 !              Set SQO when opening input files
1 !
1 !--
```

```
1 !
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !  This module contains all the routines and data structures to process the
1 !  input command parameters and options.  There is a global routine:
1 !
1 !       LNK$GETCMD (arglist)
1 !
1 !  Where arglist is the address of the original argument list with which the
1 !  linker was called.  This argument list is used to perform calls back to
1 !  the CLI for result parsing.
1 !
1 !  This module also contains parameter request blocks which are passed with
1 !  each call to the CLI, as well as parameter qualifier description blocks and
1 !  a set of action routines which are called by the CLI on the appropriate
1 !  conditions.
1 !
1 !  A global routine  LNK$ALLOFDB  is also included in this module to allocate
1 !  and initialize file descriptors and resultant file name string areas.  It is
1 !  described in detail below.
1 !
1 !--
```

```
     1 forward routine
     1         lnk$srcpscdef,                               ! Look for psectname in psect def by opt list
     1         lnk$collectpsct : novalue,                   ! Perform PSECT collection/definition
     1         setuserlib,                                  ! Selectively set logical name tables to search
     1         clierror        : novalue,                   ! called for all cli errors
     1         call_tparse,                                 ! Call tparse to parse a value
     1         lnk$allocluster : novalue,                   ! Allocate a cluster descriptor
     1         lnk$allofdb     : novalue,                   ! Allocate file descriptors and resultant string areas
     1         outputfile      : novalue,                   ! Called for all output files
     1         inputfile       : novalue,                   ! Called for each input file
     1         specialimage    : novalue,                   ! Called for system and shareable images
     1         lnk$setlibrin   : novalue,                   ! Called for library files
     1         lnk$insert_clu,                              ! Insert cluster into cluster tree
     1         optionfile      : novalue,                   !
     1         get_image       : novalue,                   ! Process image qualifiers
     1         get_map         : novalue,                   ! Process map qualifiers
     1         get_sym         : novalue,                   ! Process /SYM
     1         get_input,                                   ! Process input file name
     1         setup_inrelnam ;                             ! Setup related name for output files
     1 !
     1 LIBRARY
     1         'STARLETL32' ;                               ! System structure def'ns
     1 require
     1         'PREFIX' ;                                   ! Utility macros etc.
     1 library
     1         'DATBAS' ;                                   ! Internal data base definitions
     1 library
     1         'CLIMACL32' ;                                ! Command language structures
     1 library
     1         'TPAMACL32' ;                                ! Tparse macros
     1 !
     1 ! Set up string descriptors for qualifers
     1 !
P    1     sd  ('BRIEF',         'CONTIGUOUS',    'CROSS_REFERENCE',     'DEBUG',      'EXECUTABLE'
P    1         ,'FULL',          'HEADER',        'MAP',                 'POIMAGE',    'PROTECT'
P    1         ,'SHAREABLE',     'SYMBOL_TABLE',  'SYSLIB',              'SYSSHR',     'SYSTEM'
P    1         ,'TRACE',         'USERLIBRARY',   'INCLUDE',             'LIBRARY',    'OPTIONS'
P    1         ,'SELECTIVE_SEARCH',               'P1'
     1         ) ;
     1
M    1 macro   copy_desc (src, dst) =
M    1                 begin
M    1                 dst [dsc$w_length]  = .src [dsc$w_length] ;
M    1                 dst [dsc$b_class]   = .src [dsc$b_class] ;
M    1                 dst [dsc$b_dtype]   = .src [dsc$b_dtype] ;
M    1                 dst [dsc$a_pointer] = .src [dsc$a_pointer] ;
M    1                 end
     1         % ;
     1
     1 external literal
     1         cli$_negated,                                ! Qualifier explicitly negated
     1         cli$_locneg,                                 ! Qualifier was locally negated
     1         cli$_defaulted,                              ! Qualifier was defaulted
     1         cli$_locpres,                                ! Qualifier was locally present
     1         cli$_present,                                ! Qualifier was present
     1         lin$_univstar,                               ! UNIVERSAL=* ignored unless /NOEXE
     1         lin$_closein,                                ! Close error
```

```
1               lin$_cmdsyntax,                         ! Command syntax error
1               lin$_firstmod,                          ! First module requires extraction since library
1               lin$_illqualval,                        ! Illegal qualifier value
1               lin$_linerr,                            ! Display segment in error
1               lin$_nomods,                            ! No input modules specified
1               lin$_openin,                            ! Error opening input file
1               lin$_tracign ;                          ! Traceback supression overridden
1
1 external routine
1               str$copy_r,
1               lib$cvt_otb,
1               lib$cvt_dtb,
1               lib$cvt_htb,
1               cli$present,
1               cli$get_value,
1               lib$find_file,                          ! Search list compatible related name processing
1               lib$find_file_end,                      ! Cleanup after LIB$FIND_FILE
1               lib$insert_tree,                        ! Insert into balanced binary tree
1               lib$lookup_tree,                        ! Lookup entry in balanced binary tree
1               lib$tparse,                             ! Table driven parser
1               lnk$fndpscnam,                          ! Lookup/enter psect names
1               lnk$alloblk     : novalue,              ! Allocate dynamic memory
1               lnk$dealblk     : novalue,              ! Deallocate dynamic memory
1               lnk$procoptns,                          ! Option file processor
1               lnk$filnamdsc,                          ! Return string descriptor for file name
1               lnk$insudfsym,                          ! Insert symbol descriptor in undefined symbol list
1               lnk$insert,                             ! Symbol table insert routine
1               lnk$closefile,                          ! Routine to close a file
1               lnk$openlib,                            ! Routine to open a library
1               lnk$search ;                            ! Symbol table search routine
1 !
1 external
1               lnk$gl_inclst   : ref block [,byte],    ! Include list descriptor
1               lnk$gw_stack    : word,                 ! Stack size
1               lnk$gl_curfil   : ref block [,byte],
1               lnk$gl_ctlmsk   : block [,byte],        ! Control flags
1               lnk$gl_lastclu  : ref block [,byte],    ! Pointer to last cluster in list
1               lnk$gl_clulst   : vector [2],           ! Cluster descriptor listhead
1               lnk$gl_defclu   : block [,byte],        ! Default cluster descriptor
1               lnk$gl_curclu   : ref block [,byte],    ! Current cluster descriptor
1               lnk$gl_clutree  : ref block [,byte],    ! Listhead for cluster tree
1               lnk$gl_fvmlst   : ref block [,byte],    ! Free virtual memory listhead
1               lnk$gl_imgfil   : ref block [,byte],    ! Pointer to image file descriptor slot
1               lnk$gl_mapfil   : ref block [,byte],    ! Pointer to the map file descriptor
1               lnk$gw_nclusters : word,                ! Number of clusters allocated
1               lnk$gw_nsymbols : word,                 ! Number of symbols
1               lnk$gl_symfil   : ref block [,byte],    ! Pointer to symbol table output file
1               lnk$gl_cclulst,                         ! List head (singly linked)
1                                                       ! For clusters in collect option
1               lnk$gl_pscdflst ;                       ! Singly linked listhead
1                                                       ! For psects defined with psect option
1 !
1 Bind          firstcluster = lnk$gl_clulst [0]   : ref block [,byte], ! Pointer to first
1               null_device  = $descriptor ('NL:') : block [,byte]  ;   ! Null device for /NOEXE
1 !
1 global
1               lnk$gl_filesleft,                       ! Number of files left to open
```

```
        lnk$gl_ulibmask,                        ! Mask of enabled name tables to search
        lnk$gl_inrelnam,                        ! Address of nam block of first input file
        lnk$gl_relnam_exe,                      ! Address of related nam block for .EXE
        lnk$gl_relnam_map,                      ! Address of related nam block for .MAP
        lnk$gl_relnam_sym,                      ! Address of related nam block for .STB
        lnk$gb_locnov_exe       : byte,         ! Set if /EXE used locally with no specified value
        lnk$gb_locnov_map       : byte,         ! Set if /MAP used locally with no specified value
        lnk$gb_locnov_sym       : byte,         ! Set if /SYM used locally with no specified value
        lnk_gb_getcmd_sp1       : byte,         ! Spare, used for alignment
        lnk$gl_sysinput_flag,                   ! Set when option file is sys$input
        lnk$gl_cliwork ;                        ! Pointer to work area for cli
!
global bind
        lnk$gt_jpilst   = $itmlst_uplit ((itmcod = jpi$_filcnt, bufadr = lnk$gl_filesleft)),

        lnk$gt_imgsta   = cstring ('SYS$IMGSTA'),               ! Name of debugger activator
        libfiltype      = uplit (byte ('.OLB')),                ! Library file type
        defiletype      = uplit (byte ('.OBJ')),                ! Default file type
        shrfiltype      = uplit (byte ('.EXE')) ;               ! Shr img file type

map
        lnk$gl_ulibmask : bitvector [] ;                        ! Pretend its a bitvector
!
! OWN STORAGE
!
literal
        imgtabsiz       = 7,
        system_region   = %x'80000000',
        rsa_allosize    = nam$c_maxrss*3/2,                     ! Rsa allocation quantity
        typelen         = 4 ;                                   ! Length of file type string
!
own
        univstar_valid  : initial (false),                      ! UNIV=* valid only with /NOEXE
        local_qualifier : initial (false),                      ! Set to TRUE when /MAP, /EXE, and
                                                                ! /SYM are parsed as local qualifiers
        listptr,                                                ! Dynamic include list ptr
        dflt_name       : vector [nam$c_maxrss, byte],          ! Default input related name text
        dflt_inrel      : block [nam$c_bln, byte],              ! Default input related name nam$
        last_fnm_desc   : ref block [,byte] initial (0),        ! Last input file descriptor
        rlf_context     : ref block [,byte] initial (0),        ! Related filename context
        an_open_fdb     : initial (0),
        debug_desc      : dynamic_descriptor,
        module_desc     : dynamic_descriptor,
        share_desc      : dynamic_descriptor,                   ! /SHAREABLE descriptor name
        token_desc      : dynamic_descriptor,                   ! Dynamic desc for general usage
        systembase,                                             ! Base specified in /SYSTEM=NUMBER
        basedflag,                                              ! Set true if /SYSTEM=NUMBER
        outfile,                                                ! Ptr to ptr for the current output file
        nextrsa,                                                ! Ptr to slot for next resultant string on o
        typeadr         : initial (ch$ptr (defiletype)),        ! Address of the file type string
        infilflgs,                                              ! Flags of each input file
        tpdata,                                                 ! Data from tparse
        rsaremsize      : byte,                                 ! Size remaining in rsa
        image_qual_table : vector  [ imgtabsiz, long ]
                                initial
                                        (sd_trace,  sd_contiguous, sd_p0image, sd_protect
```

LNK
V04

; R

;

```
                                          ,sd_syslib, sd_sysshr,      sd_header
                                          ),
          bit_table         : vector  [imgtabsiz, byte]
                              initial
                                 (byte    ($bitposition (lnk$v_trace),    $bitposition (lnk$v_contig)
                                          ,$bitposition (lnk$v_p0image), $bitposition (lnk$v_protect)
                                          ,$bitposition (lnk$v_syslib),  $bitposition (lnk$v_sysshr)
                                          ,$bitposition (lnk$v_sysheadr)
                                 )        ) ;

!
! Tparse tables to parse /userlib qualifier values
!
$init_state (userlib_states,userlib_keys)  ;

$state (,        ('SYSTEM', tpa$_exit,,1,lnk$gl_ulibmask),
                 ('GROUP',  tpa$_exit,,2,lnk$gl_ulibmask),
                 ('PROCESS',tpa$_exit,,4,lnk$gl_ulibmask),
                 ('ALL',    tpa$_exit,,7,lnk$gl_ulibmask),
                 ('NONE',   tpa$_exit,,%x'80000000',lnk$gl_ulibmask)
        ) ;
```

```
1 global routine lnk$getcmd (arglist) : novalue =
1
1 !+
1 !
1 !  This routine initializes the CLI result parser then re-calls it for
1 !  processing of each command parameter.  It first calls for the primary output
1 !  parameter (the image), then the secondary output parameter (the map),
1 !  followed by the third output parameter (the symbol table file).  Next it
1 !  loops, calling for input parameters (primary only) until there are no more.
1 !  the actions taken on presence or absence of any parameter and its qualifiers
1 !  are taken in the other routines (below) which are called by CLI on the
1 !  appropriate condition.
1 !
1 !
1 !  ARGLIST is the address of the argument list with which the linker was
1 !          originally called.
1 !
1 !-
2 begin
2 map
2         arglist : ref block [,byte] ;                                ! Really a vector of longwords
2
2 lnk$gl_ctlmsk [lnk$v_verify] = .arglist [cli$v_verify] ;             ! Copy verify context of process
2
2 rsaremsize  = rsa_allosize*3 ;                                       ! Set up the resultant string area
2 lnk$alloblk (.rsaremsize,nextrsa) ;                                  ! For the file opens (triple size at first)
2
2 !
2 !
2 !    Loop through qualifiers that do nothing but set a bit in  LNK$GL_CTLMSK...
2 !
2 !
2 incr i  to imgtabsiz - 1
2 do   lnk$gl_ctlmsk [0,.bit_table [.i],1,0] = cli$present (.image_qual_table [.i]) ;
2
2 if   cli$present (sd_userlibrary)                                    ! /USERLIBRARY
2 then setuserlib () ;
2
2 if  cli$present (sd_system)                                          ! /SYSTEM
3 then begin
3      lnk$gl_ctlmsk [lnk$v_sys]   = true ;                            ! Set/clear bits
3      lnk$gl_ctlmsk [lnk$v_trace] = false ;                           !  and call
3      lnk$gl_ctlmsk [lnk$v_exe]   = false ;                           !  specialimage
3      specialimage () ;
3      end
2 else if   cli$present (sd_shareable)                                 ! /SHAREABLE
3      then begin
3           lnk$gl_ctlmsk [lnk$v_shr]   = true ;                       ! Set/clear bits
3           lnk$gl_ctlmsk [lnk$v_trace] = false ;                      !  and call
3           lnk$gl_ctlmsk [lnk$v_exe]   = false ;                      !  specialimage
3           specialimage () ;
2           end ;
2
2 if   cli$present (sd_debug)                                          ! If /DEBUG present,
3 then begin                                                           !   then attempt to grab a value for
3      cli$get_value (sd_debug, debug_desc) ;                          !   the qualifier  (no value is ok)
3      lnk$gl_ctlmsk [lnk$v_dbg] = true ;
```

```
2        end ;
2
2  get_image () ;                                                   ! Go get image file name
2  get_map   () ;                                                   ! Go get map file name and qualifiers
2  get_sym   () ;                                                   ! Get symbol table file name
2
2  local_qualifier = true ;                                         ! /MAP/EXE/SYM are now local qualifiers
2  while get_input ()                                               ! Loop while more input files...
3  do  begin
3       get_image () ;
3       get_map   () ;
3       get_sym   () ;
2       end ;
2
2
2  lib$find_file_end (rlf_context) ;                                ! Cleanup after input processing
2
2  lnk$gl_ctlmsk [lnk$v_image] =    .lnk$gl_ctlmsk [lnk$v_sys]  or
2                                   .lnk$gl_ctlmsk [lnk$v_shr]  or
2                                   .lnk$gl_ctlmsk [lnk$v_exe] ;
2  !
2  !
2  !     If a MAP was requested, process MAP qualifiers.  This could be done
2  !     sooner since /MAP is positional and the other map qualifiers are
2  !     global.
2  !
2  !
2  if  .lnk$gl_ctlmsk [lnk$v_map]
3  then begin
3       if    cli$present (sd_full)                                 ! Process map qualifiers
4       then begin
4            lnk$gl_ctlmsk [lnk$v_long]   = true ;
4            lnk$gl_ctlmsk [lnk$v_supsys] = false ;
3            end ;
3
3       if    cli$present (sd_brief)
3       then lnk$gl_ctlmsk [lnk$v_brief] = true ;
3
3       if    cli$present (sd_cross_reference)
3       then lnk$gl_ctlmsk [lnk$v_cros] = true ;
2       end ;
2
2  !
2  !
2  !     If /SYSTEM=number, then set the base now
2  !
2  !
2  if .basedflag
2  then
3       begin
3       if  .lnk$gl_ctlmsk [lnk$v_sysheadr]
3       then lnk$gl_defclu [clu$l_base] =  (.systembase + 511) and not 511  ! Only round if /SYSTEM/HEADER
3       else lnk$gl_defclu [clu$l_base] =   .systembase ;
3
3       if        .lnk$gl_defclu [clu$l_base] gequ control_region          ! Set correct allocation limits
3       then if   .lnk$gl_defclu [clu$l_base] gequ system_region
3           then      lnk$gl_fvmlst [fvm$l_address] = system_region
```

```
3            else      lnk$gl_fvmlst [fvm$l_address] = control_region
3        else          lnk$gl_fvmlst [fvm$l_address] = 0 ;
2        end ;
2    !
2    !
2    !   Check that there was at least one input module
2    !
2    !
2    if .firstcluster eql 0                                      ! if no clusters allocated
2        and
2       .lnk$gl_defclu [clu$l_fstfdb] eql 0                      ! And no files in default cluster
2    then
2        signal_stop (lin$_nomods) ;                             ! There were no input modules - fatal error
2
2    !
2    !
2    !   Now link the debugger into the input file list if DEBUG and/or TRACEBACK
2    !   was requested.
2    !
2    !   However suppress it if this is a system image
2    !
2    !
2    if  .lnk$gl_ctlmsk [lnk$v_sys] or .lnk$gl_ctlmsk [lnk$v_shr]
2    then lnk$gl_ctlmsk [lnk$v_trace] = false ;
2
2    if .lnk$gl_ctlmsk [lnk$v_dbg] or .lnk$gl_ctlmsk [lnk$v_trace]   ! If /DEBUG specified or /TRACE was
2    then                                                        ! present or implied
3        begin
3        if   not .lnk$gl_ctlmsk [lnk$v_trace]                   ! If only dbg then
3        then signal (lin$_tracign) ;                            ! Report conflict as info.
3
3        infilflgs = fdb$m_debuger ;                             ! Set the file flag
3        if  .debug_desc [dsc$w_length] neq 0                    ! If /DEBUG=FILENAME
4        then begin
4            inputfile (debug_desc) ;                            ! Then go do it
4            lib$find_file_end (rlf_context) ;
4            end
4        else begin                                              ! Else default debugger
4            local  symnam : ref block [,byte],                  ! Pointer to name part of symbol block
4                   symval : ref block [,byte] ;                 ! Pointer to value part of symbol block
4
4            lnk$search (lnk$gt_imgsta,symval,symnam) ;          ! Look up the symbol
4            lnk$insert (lnk$gt_imgsta,symval,symnam) ;          ! And insert it
4            lnk$insudfsym (.symval) ;                           ! Insert symbol in undefined list
4            lnk$gw_nsymbols = .lnk$gw_nsymbols + 1 ;            ! Count the symbol
3            end ;
2        end ;
2
2    lnk$gl_ctlmsk [lnk$v_verify] = false ;                      ! Turn off verify status
2
2    lnk$collectpsct () ;                                        ! Perform psect collection and definition
2
2    if  .lnk$gl_defclu [clu$l_fstfdb] neq 0                     ! If default cluster has some files
3    then begin
3        lnk$gl_lastclu [clu$l_nxtclu]     =  lnk$gl_defclu ;    ! Link it on to end of
3        if  (lnk$gl_defclu [clu$l_prevclu] = .lnk$gl_lastclu) eql 0! Cluster list
3        then lnk$gl_defclu [clu$l_prevclu] =  lnk$gl_clulst ;
```

```
;   3         lnk$gl_lastclu                    =   lnk$gl_defclu ;        ! And make it the last cluster then
;   2         end ;
;   2
;   2 if   .lnk$gl_ctlmsk [lnk$v_alluniv]                                  ! if univ=* requested, allow it only
;   2 then if not .univstar_valid                                          !   if /NOEXE specified for .STB
;   3       then begin                                                     !   generation
;   3            signal (lin$_univstar) ;
;   3            lnk$gl_ctlmsk [lnk$v_alluniv] = false ;                   ! otherwise disable it
;   2            end ;
;   2 return ;
;   1 end ;


                                                  .TITLE  LNK_GETCMDINP
                                                  .IDENT  \V04-000\

                                                  .PSECT  _LIB$KEY1$,NOWRT,  SHR,  PIC,1

                               00000 ;TPA$KEYST0
                                     U.2:         .BLKB   0
              4D 45 54 53 59 53 00000 ;TPA$KEYST
                                     U.4:         .ASCII  \SYSTEM\                              ;
                            FF 00006              .BYTE   -1                                    ;
                               00007 ;TPA$KEYST0
                                     U.9:         .BLKB   0
                 50 55 4F 52 47 00007 ;TPA$KEYST
                                     U.11:        .ASCII  \GROUP\                               ;
                            FF 0000C              .BYTE   -1                                    ;
                               0000D ;TPA$KEYST0
                                     U.16:        .BLKB   0
           53 53 45 43 4F 52 50 0000D ;TPA$KEYST
                                     U.18:        .ASCII  \PROCESS\                             ;
                            FF 00014              .BYTE   -1                                    ;
                               00015 ;TPA$KEYST0
                                     U.23:        .BLKB   0
                      4C 4C 41 00015 ;TPA$KEYST
                                     U.25:        .ASCII  \ALL\                                 ;
                            FF 00018              .BYTE   -1                                    ;
                               00019 ;TPA$KEYST0
                                     U.30:        .BLKB   0
                   45 4E 4F 4E 00019 ;TPA$KEYST
                                     U.32:        .ASCII  \NONE\                                ;
                            FF 0001D              .BYTE   -1                                    ;
                            FF 0001E ;TPA$KEYFILL
                                     U.37:        .BYTE   -1                                    ;

                                                  .PSECT  _LIB$STATE$,NOWRT,  SHR,  PIC,1

                               00000 USERLIB_STATES::
                                                  .BLKB   0
                          7100 00000 ;TPA$TYPE
                                     U.5:         .WORD   28928                                 ;
                     00000000* 00002 ;TPA$ADDR
                                     U.6:         .LONG   <<LNK$GL_ULIBMASK-U.6>-4>             ;
                      00000001 00006 ;TPA$MASK
                                     U.7:         .LONG   1                                     ;
                          FFFF 0000A ;TPA$TARGET
```

```
                        U.8:      .WORD    -1                              ;
     7101  0000C ;TPA$TYPE
                        U.12:     .WORD    28929                           ;
 00000000* 0000E ;TPA$ADDR
                        U.13:     .LONG    <<LNK$GL_ULIBMASK-U.13>-4>      ;
 00000002  00012 ;TPA$MASK
                        U.14:     .LONG    2                               ;
     FFFF  00016 ;TPA$TARGET
                        U.15:     .WORD    -1                              ;
     7102  00018 ;TPA$TYPE
                        U.19:     .WORD    28930                           ;
 00000000* 0001A ;TPA$ADDR
                        U.20:     .LONG    <<LNK$GL_ULIBMASK-U.20>-4>      ;
 00000004  0001E ;TPA$MASK
                        U.21:     .LONG    4                               ;
     FFFF  00022 ;TPA$TARGET
                        U.22:     .WORD    -1                              ;
     7103  00024 ;TPA$TYPE
                        U.26:     .WORD    28931                           ;
 00000000* 00026 ;TPA$ADDR
                        U.27:     .LONG    <<LNK$GL_ULIBMASK-U.27>-4>      ;
 00000007  0002A ;TPA$MASK
                        U.28:     .LONG    7                               ;
     FFFF  0002E ;TPA$TARGET
                        U.29:     .WORD    -1                              ;
     7504  00030 ;TPA$TYPE
                        U.33:     .WORD    29956                           ;
 00000000* 00032 ;TPA$ADDR
                        U.34:     .LONG    <<LNK$GL_ULIBMASK-U.34>-4>      ;
 80000000  00036 ;TPA$MASK
                        U.35:     .LONG    -2147483648                     ;
     FFFF  0003A ;TPA$TARGET
                        U.36:     .WORD    -1                              ;

                                  .PSECT   _LIB$KEY0$,NOWRT, SHR, PIC,1

           00000 USERLIB_KEYS::
                                  .BLKB    0
           00000 ;TPA$KEY0
                        U.1:      .BLKB    0
     0000* 00000 ;TPA$KEY
                        U.3:      .WORD    <U.2-U.1>                       ;
     0000* 00002 ;TPA$KEY
                        U.10:     .WORD    <U.9-U.1>                       ;
     0000* 00004 ;TPA$KEY
                        U.17:     .WORD    <U.16-U.1>                      ;
     0000* 00006 ;TPA$KEY
                        U.24:     .WORD    <U.23-U.1>                      ;
     0000* 00008 ;TPA$KEY
                        U.31:     .WORD    <U.30-U.1>                      ;

                                  .PSECT   $PLIT$,NOWRT,NOEXE,2

 46 45 49 52 42  00000 P.AAB:     .ASCII   \BRIEF\                         ;
                 00005            .BLKB    3
       00000005  00008 P.AAA:     .LONG    5                               ;
       00000000' 0000C            .ADDRESS P.AAB                           ;
```

```
                  53 55 4F 55 47 49 54 4E 4F 43  00010 P.AAD:  .ASCII  \CONTIGUOUS\                 ;
                                                 0001A         .BLKB   2
                                      0000000A   0001C P.AAC:  .LONG   10                           ;
                                      00000000'  00020         .ADDRESS P.AAD                       ;
45 43 4E 45 52 45 46 45 52 5F 53 53 4F 52 43     00024 P.AAF:  .ASCII  \CROSS_REFERENCE\            ;
                                                 00033         .BLKB   1
                                      0000000F   00034 P.AAE:  .LONG   15                           ;
                                      00000000'  00038         .ADDRESS P.AAF                       ;
                              47 55 42 45 44     0003C P.AAH:  .ASCII  \DEBUG\                      ;
                                                 00041         .BLKB   3
                                      00000005   00044 P.AAG:  .LONG   5                            ;
                                      00000000'  00048         .ADDRESS P.AAH                       ;
                  45 4C 42 41 54 55 43 45 58 45  0004C P.AAJ:  .ASCII  \EXECUTABLE\                 ;
                                                 00056         .BLKB   2
                                      0000000A   00058 P.AAI:  .LONG   10                           ;
                                      00000000'  0005C         .ADDRESS P.AAJ                       ;
                                 4C 4C 55 46     00060 P.AAL:  .ASCII  \FULL\                       ;
                                      00000004   00064 P.AAK:  .LONG   4                            ;
                                      00000000'  00068         .ADDRESS P.AAL                       ;
                           52 45 44 41 45 48     0006C P.AAN:  .ASCII  \HEADER\                     ;
                                                 00072         .BLKB   2
                                      00000006   00074 P.AAM:  .LONG   6                            ;
                                      00000000'  00078         .ADDRESS P.AAN                       ;
                                    50 41 4D     0007C P.AAP:  .ASCII  \MAP\                        ;
                                                 0007F         .BLKB   1
                                      00000003   00080 P.AAO:  .LONG   3                            ;
                                      00000000'  00084         .ADDRESS P.AAP                       ;
                        45 47 41 4D 49 30 50     00088 P.AAR:  .ASCII  \P0IMAGE\                    ;
                                                 0008F         .BLKB   1
                                      00000007   00090 P.AAQ:  .LONG   7                            ;
                                      00000000'  00094         .ADDRESS P.AAR                       ;
                        54 43 45 54 4F 52 50     00098 P.AAT:  .ASCII  \PROTECT\                    ;
                                                 0009F         .BLKB   1
                                      00000007   000A0 P.AAS:  .LONG   7                            ;
                                      00000000'  000A4         .ADDRESS P.AAT                       ;
                     45 4C 42 41 45 52 41 48 53  000A8 P.AAV:  .ASCII  \SHAREABLE\                  ;
                                                 000B1         .BLKB   3
                                      00000009   000B4 P.AAU:  .LONG   9                            ;
                                      00000000'  000B8         .ADDRESS P.AAV                       ;
         45 4C 42 41 54 5F 4C 4F 42 4D 59 53     000BC P.AAX:  .ASCII  \SYMBOL_TABLE\               ;
                                      0000000C   000C8 P.AAW:  .LONG   12                           ;
                                      00000000'  000CC         .ADDRESS P.AAX                       ;
                           42 49 4C 53 59 53     000D0 P.AAZ:  .ASCII  \SYSLIB\                     ;
                                                 000D6         .BLKB   2
                                      00000006   000D8 P.AAY:  .LONG   6                            ;
                                      00000000'  000DC         .ADDRESS P.AAZ                       ;
                           52 48 53 53 59 53     000E0 P.ABB:  .ASCII  \SYSSHR\                     ;
                                                 000E6         .BLKB   2
                                      00000006   000E8 P.ABA:  .LONG   6                            ;
                                      00000000'  000EC         .ADDRESS P.ABB                       ;
                           4D 45 54 53 59 53     000F0 P.ABD:  .ASCII  \SYSTEM\                     ;
                                                 000F6         .BLKB   2
                                      00000006   000F8 P.ABC:  .LONG   6                            ;
                                      00000000'  000FC         .ADDRESS P.ABD                       ;
                              45 43 41 52 54     00100 P.ABF:  .ASCII  \TRACE\                      ;
                                                 00105         .BLKB   3
                                      00000005   00108 P.ABE:  .LONG   5                            ;
```

```
                                    00000000' 0010C          .ADDRESS P.ABF                   ;
          59 52 41 52 42 49 4C 52 45 53 55    00110 P.ABH:   .ASCII   \USERLIBRARY\           ;
                                              0011B          .BLKB    1
                                    0000000B  0011C P.ABG:   .LONG    11                      ;
                                    00000000' 00120          .ADDRESS P.ABH                   ;
                      45 44 55 4C 43 4E 49    00124 P.ABJ:   .ASCII   \INCLUDE\               ;
                                              0012B          .BLKB    1
                                    00000007  0012C P.ABI:   .LONG    7                       ;
                                    00000000' 00130          .ADDRESS P.ABJ                   ;
                      59 52 41 52 42 49 4C    00134 P.ABL:   .ASCII   \LIBRARY\               ;
                                              0013B          .BLKB    1
                                    00000007  0013C P.ABK:   .LONG    7                       ;
                                    00000000' 00140          .ADDRESS P.ABL                   ;
                      53 4E 4F 49 54 50 4F    00144 P.ABN:   .ASCII   \OPTIONS\               ;
                                              0014B          .BLKB    1
                                    00000007  0014C P.ABM:   .LONG    7                       ;
                                    00000000' 00150          .ADDRESS P.ABN                   ;
43 52 41 45 53 5F 45 56 49 54 43 45 4C 45 53  00154 P.ABP:   .ASCII   \SELECTIVE_SEARCH\      ;
                                          48  00163                                           ;
                                    00000010  00164 P.ABO:   .LONG    16                      ;
                                    00000000' 00168          .ADDRESS P.ABP                   ;
                                       31 50  0016C P.ABR:   .ASCII   \P1\                    ;
                                              0016E          .BLKB    2
                                    00000002  00170 P.ABQ:   .LONG    2                       ;
                                    00000000' 00174          .ADDRESS P.ABR                   ;
                                    3A 4C 4E  00178 P.ABT:   .ASCII   \NL:\                   ;
                                              0017B          .BLKB    1
                                    00000003  0017C P.ABS:   .LONG    3                       ;
                                    00000000' 00180          .ADDRESS P.ABT                   ;
                                   0314 0004  00184 P.ABU:   .WORD    4, 788                  ;
                                    00000000' 00188          .ADDRESS LNK$GL_FILESLEFT        ;
                                    00000000  0018C          .LONG    0                       ;
                                    00000000  00190          .LONG    0                       ;
                                          0A  00194 P.ABV:   .BYTE    10                      ;
                41 54 53 47 4D 49 24 53 59 53  00195          .ASCII   \SYS$IMGSTA\            ;
                                              0019F          .BLKB    1
                                 42 4C 4F 2E  001A0 P.ABW:   .ASCII   \.OLB\                  ;
                                 4A 42 4F 2E  001A4 P.ABX:   .ASCII   \.OBJ\                  ;
                                 45 58 45 2E  001A8 P.ABY:   .ASCII   \.EXE\                  ;

                                                             .PSECT   $OWN$,NOEXE,2

                                    00000000  00000 UNIVSTAR_VALID:
                                                             .LONG    0                       ;
                                    00000000  00004 LOCAL_QUALIFIER:
                                                             .LONG    0                       ;
                                              00008 LISTPTR:.BLKB     4
                                              0000C DFLT_NAME:
                                                             .BLKB    255
                                              0010B          .BLKB    1
                                              0010C DFLT_INREL:
                                                             .BLKB    96
                                    00000000  0016C LAST_FNM_DESC:
                                                             .LONG    0                       ;
                                    00000000  00170 RLF_CONTEXT:
                                                             .LONG    0                       ;
                                    00000000  00174 AN_OPEN_FDB:
```

```
                                                                     .LONG   0                                     ;
                                                   0000  00178 DEBUG_DESC:
                                                                     .WORD   0                                     ;
                                                     00  0017A       .BYTE   0                                     ;
                                                     02  0017B       .BYTE   2                                     ;
                                               00000000  0017C       .LONG   0                                     ;
                                                   0000  00180 MODULE_DESC:
                                                                     .WORD   0                                     ;
                                                     00  00182       .BYTE   0                                     ;
                                                     02  00183       .BYTE   2                                     ;
                                               00000000  00184       .LONG   0                                     ;
                                                   0000  00188 SHARE_DESC:
                                                                     .WORD   0                                     ;
                                                     00  0018A       .BYTE   0                                     ;
                                                     02  0018B       .BYTE   2                                     ;
                                               00000000  0018C       .LONG   0                                     ;
                                                   0000  00190 TOKEN_DESC:
                                                                     .WORD   0                                     ;
                                                     00  00192       .BYTE   0                                     ;
                                                     02  00193       .BYTE   2                                     ;
                                               00000000  00194       .LONG   0                                     ;
                                                         00198 SYSTEMBASE:
                                                                     .BLKB   4
                                                         0019C BASEDFLAG:
                                                                     .BLKB   4
                                                         001A0 OUTFILE:.BLKB   4
                                                         001A4 NEXTRSA:.BLKB   4
                                              00000000'  001A8 TYPEADR:.ADDRESS DEFILETYPE                         ;
                                                         001AC INFILFLGS:
                                                                     .BLKB   4
                                                         001B0 TPDATA: .BLKB   4
                                                         001B4 RSAREMSIZE:
                                                                     .BLKB   1
                                                         001B5       .BLKB   3
00000000' 00000000' 00000000' 00000000' 00000000' 00000000'  001B8 IMAGE_QUAL_TABLE:
                                                                     .ADDRESS SD_TRACE, SD_CONTIGUOUS, SD_POIMAGE, -  ;
                                              00000000'  001D0                SD_PROTECT, SD_SYSLIB, SD_SYSSHR, -     ;
                                                                              SD_HEADER                               ;
                         19  14  0A  17  18  13  12  001D4 BIT_TABLE:
                                                                     .BYTE   18, 19, 24, 23, 10, 20, 25            ;

                                                                     .PSECT  $GLOBAL$,NOEXE,2

                                                         00000 LNK$GL_FILESLEFT::
                                                                     .BLKB   4
                                                         00004 LNK$GL_ULIBMASK::
                                                                     .BLKB   4
                                                         00008 LNK$GL_INRELNAM::
                                                                     .BLKB   4
                                                         0000C LNK$GL_RELNAM_EXE::
                                                                     .BLKB   4
                                                         00010 LNK$GL_RELNAM_MAP::
                                                                     .BLKB   4
                                                         00014 LNK$GL_RELNAM_SYM::
                                                                     .BLKB   4
                                                         00018 LNK$GB_LOCNOV_EXE::
                                                                     .BLKB   1
```

```
00019 LNK$GB_LOCNOV_MAP::
        .BLKB   1
0001A LNK$GB_LOCNOV_SYM::
        .BLKB   1
0001B LNK_GB_GETCMD_SP1::
        .BLKB   1
0001C LNK$GL_SYSINPUT_FLAG::
        .BLKB   4
00020 LNK$GL_CLIWORK::
        .BLKB   4

      SD_BRIEF=             P.AAA
      SD_CONTIGUOUS=        P.AAC
      SD_CROSS_REFERENCE=   P.AAE
      SD_DEBUG=             P.AAG
      SD_EXECUTABLE=        P.AAI
      SD_FULL=              P.AAK
      SD_HEADER=            P.AAM
      SD_MAP=               P.AAO
      SD_POIMAGE=           P.AAQ
      SD_PROTECT=           P.AAS
      SD_SHAREABLE=         P.AAU
      SD_SYMBOL_TABLE=      P.AAW
      SD_SYSLIB=            P.AAY
      SD_SYSSHR=            P.ABA
      SD_SYSTEM=            P.ABC
      SD_TRACE=             P.ABE
      SD_USERLIBRARY=       P.ABG
      SD_INCLUDE=           P.ABI
      SD_LIBRARY=           P.ABK
      SD_OPTIONS=           P.ABM
      SD_SELECTIVE_SEARCH=P.ABO
      SD_P1=                P.ABQ
      NULL_DEVICE=          P.ABS
      LNK$GT_JPILST==       P.ABU
      LNK$GT_IMGSTA==       P.ABV
      LIBFILTYPE==          P.ABW
      DEFILETYPE==          P.ABX
      SHRFILTYPE==          P.ABY
              .EXTRN  CLI$_NEGATED, CLI$_LOCNEG
              .EXTRN  CLI$_DEFAULTED, CLI$_LOCPRES
              .EXTRN  CLI$_PRESENT, LIN$_UNIVSTAR
              .EXTRN  LIN$_CLOSEIN, LIN$_CMDSYNTAX
              .EXTRN  LIN$_FIRSTMOD, LIN$_ILLQUALVAL
              .EXTRN  LIN$_LINERR, LIN$_NOMODS
              .EXTRN  LIN$_OPENIN, LIN$_TRACIGN
              .EXTRN  STR$COPY_R, LIB$CVT_OTB
              .EXTRN  LIB$CVT_DTB, LIB$CVT_HTB
              .EXTRN  CLI$PRESENT, CLI$GET_VALUE
              .EXTRN  LIB$FIND_FILE, LIB$FIND_FILE_END
              .EXTRN  LIB$INSERT_TREE
              .EXTRN  LIB$LOOKUP_TREE
              .EXTRN  LIB$TPARSE, LNK$FNDPSCNAM
              .EXTRN  LNK$ALLOBLK, LNK$DEALBLK
              .EXTRN  LNK$PROCOPTNS, LNK$FILNAMDSC
              .EXTRN  LNK$INSUDFSYM, LNK$INSERT
              .EXTRN  LNK$CLOSEFILE, LNK$OPENLIB
```

```
                                                        .EXTRN   LNK$SEARCH, LNK$GL_INCLST
                                                        .EXTRN   LNK$GW_STACK, LNK$GL_CURFIL
                                                        .EXTRN   LNK$GL_CTLMSK, LNK$GL_LASTCLU
                                                        .EXTRN   LNK$GL_CLULST, LNK$GL_DEFCLU
                                                        .EXTRN   LNK$GL_CURCLU, LNK$GL_CLUTREE
                                                        .EXTRN   LNK$GL_FVMLST, LNK$GL_IMGFIL
                                                        .EXTRN   LNK$GL_MAPFIL, LNK$GW_NCLUSTERS
                                                        .EXTRN   LNK$GW_NSYMBOLS
                                                        .EXTRN   LNK$GL_SYMFIL, LNK$GL_CCLULST
                                                        .EXTRN   LNK$GL_PSCDFLST

                                                        .PSECT   $CODE$,NOWRT,2

                                      0FFC 00000        .ENTRY   LNK$GETCMD, Save R2,R3,R4,R5,R6,R7,R8,R9,-   ; 0454
                                                                 R10,R11                                       ;
                     5B 00000000V  EF   9E 00002        MOVAB    GET_SYM, R11                                  ;
                     5A 00000000V  EF   9E 00009        MOVAB    GET_MAP, R10                                  ;
                     59 00000000V  EF   9E 00010        MOVAB    GET_IMAGE, R9                                 ;
                     58 00000000G  00   9E 00017        MOVAB    CLI$PRESENT, R8                               ;
                     57 00000000G  00   9E 0001E        MOVAB    LNK$GL_DEFCLU+76, R7                          ;
                     56 00000000'  EF   9E 00025        MOVAB    SD_DEBUG, R6                                  ;
                     55 00000000'  EF   9E 0002C        MOVAB    DEBUG_DESC, R5                                ;
                     54 00000000G  00   9E 00033        MOVAB    LNK$GL_CTLMSK, R4                             ;
                     5E            08   C2 0003A        SUBL2    #8, SP                                        ;
                     50       04   AC   D0 0003D        MOVL     ARGLIST, R0                                   ; 0476
    51       18  A0  01            02   EF 00041        EXTZV    #2, #1, 24(R0), R1                            ;
01  A4           01  04            51   F0 00047        INSV     R1, #4, #1, LNK$GL_CTLMSK+1                   ;
              3C     A5       7A   8F   90 0004D        MOVB     #122, RSAREMSIZE                              ; 0478
                     2C            A5   9F 00052        PUSHAB   NEXTRSA                                       ; 0479
                     7E       3C   A5   9A 00055        MOVZBL   RSAREMSIZE, -(SP)                             ;
           00000000G 00            02   FB 00059        CALLS    #2, LNK$ALLOBLK                               ;
                                   52   D4 00060        CLRL     I                                             ; 0486
                          40 A542       DD 00062 1$:    PUSHL    IMAGE_QUAL_TABLE[I]                           ; 0487
                     68            01   FB 00066        CALLS    #1, CLI$PRESENT                               ;
                     53       5C A542   9A 00069        MOVZBL   BIT_TABLE[I], R3                              ;
    64           01  53            50   F0 0006E        INSV     R0, R3, #1, LNK$GL_CTLMSK                     ;
                 EB  52            06   F3 00073        AOBLEQ   #6, I, 1$                                     ;
                          00D8     C6   9F 00077        PUSHAB   SD_USERLIBRARY                                ; 0489
                     68            01   FB 0007B        CALLS    #1, CLI$PRESENT                               ;
                     07            50   E9 0007E        BLBC     R0, 2$                                        ;
           00000000V EF            00   FB 00081        CALLS    #0, SETUSERLIB                                ; 0490
                          00B4     C6   9F 00088 2$:    PUSHAB   SD_SYSTEM                                     ; 0492
                     68            01   FB 0008C        CALLS    #1, CLI$PRESENT                               ;
                     05            50   E9 0008F        BLBC     R0, 3$                                        ;
                     64            08   88 00092        BISB2    #8, LNK$GL_CTLMSK                             ; 0494
                                   0C   11 00095        BRB      4$                                            ; 0495
                          70       A6   9F 00097 3$:    PUSHAB   SD_SHAREABLE                                  ; 0499
                     68            01   FB 0009A        CALLS    #1, CLI$PRESENT                               ;
                     11            50   E9 0009D        BLBC     R0, 5$                                        ;
                     64            04   88 000A0        BISB2    #4, LNK$GL_CTLMSK                             ; 0501
                     64 00040002   8F   CA 000A3 4$:    BICL2    #262146, LNK$GL_CTLMSK                        ; 0502
           00000000V EF            00   FB 000AA        CALLS    #0, SPECIALIMAGE                              ; 0504
                                   56   DD 000B1 5$:    PUSHL    R6                                            ; 0507
                     68            01   FB 000B3        CALLS    #1, CLI$PRESENT                               ;
                     0F            50   E9 000B6        BLBC     R0, 6$                                        ;
                                   55   DD 000B9        PUSHL    R5                                            ; 0509
                                   56   DD 000BB        PUSHL    R6                                            ;
```

```
                   00000000G  00               02  FB  000BD        CALLS   #2, CLI$GET_VALUE                         ;
                              64         40    8F  88  000C4        BISB2   #64, LNK$GL_CTLMSK                        ; 0510
                              69               00  FB  000C8  6$:   CALLS   #0, GET_IMAGE                             ; 0513
                              6A               00  FB  000CB        CALLS   #0, GET_MAP                               ; 0514
                              6B               00  FB  000CE        CALLS   #0, GET_SYM                               ; 0515
                        FEBC  C5               01  D0  000D1        MOVL    #1, LOCAL_QUALIFIER                       ; 0517
                   00000000V  EF               00  FB  000D6  7$:   CALLS   #0, GET_INPUT                             ; 0518
                              0B               50  E9  000DD        BLBC    R0, 8$                                    ;
                              69               00  FB  000E0        CALLS   #0, GET_IMAGE                             ; 0520
                              6A               00  FB  000E3        CALLS   #0, GET_MAP                               ; 0521
                              6B               00  FB  000E6        CALLS   #0, GET_SYM                               ; 0522
                                               EB  11  000E9        BRB     7$                                        ; 0518
                                         F8    A5  9F  000EB  8$:   PUSHAB  RLF_CONTEXT                               ; 0526
                   00000000G  00               01  FB  000EE        CALLS   #1, LIB$FIND_FILE_END                     ;
       50         64          01               03  EF  000F5        EXTZV   #3, #1, LNK$GL_CTLMSK, R0                 ; 0529
       51         64          01               02  EF  000FA        EXTZV   #2, #1, LNK$GL_CTLMSK, R1                 ;
                              50               51  C8  000FF        BISL2   R1, R0                                    ;
       52         64          01               01  EF  00102        EXTZV   #1, #1, LNK$GL_CTLMSK, R2                 ; 0530
                              52               50  88  00107        BISB2   R0, R2                                    ;
       64         01          00               52  F0  0010A        INSV    R2, #0, #1, LNK$GL_CTLMSK                 ;
                  2C          64               04  E1  0010F        BBC     #4, LNK$GL_CTLMSK, 11$                    ; 0538
                                         20    A6  9F  00113        PUSHAB  SD_FULL                                   ; 0540
                              68               01  FB  00116        CALLS   #1, CLI$PRESENT                           ;
                              09               50  E9  00119        BLBC    R0, 9$                                    ;
                          01  A4               01  88  0011C        BISB2   #1, LNK$GL_CTLMSK+1                       ; 0542
                          01  A4         40    8F  8A  00120        BICB2   #64, LNK$GL_CTLMSK+1                      ; 0543
                                         C4    A6  9F  00125  9$:   PUSHAB  SD_BRIEF                                  ; 0546
                              68               01  FB  00128        CALLS   #1, CLI$PRESENT                           ;
                              04               50  E9  0012B        BLBC    R0, 10$                                   ;
                          01  A4               02  88  0012E        BISB2   #2, LNK$GL_CTLMSK+1                       ; 0547
                                         F0    A6  9F  00132  10$:  PUSHAB  SD_CROSS_REFERENCE                        ; 0549
                              68               01  FB  00135        CALLS   #1, CLI$PRESENT                           ;
                              04               50  E9  00138        BLBC    R0, 11$                                   ;
                              64         80    8F  88  0013B        BISB2   #128, LNK$GL_CTLMSK                       ; 0550
                              4F         24    A5  E9  0013F  11$:  BLBC    BASEDFLAG, 16$                            ; 0558
       13         03          A4               01  E1  00143        BBC     #1, LNK$GL_CTLMSK+3, 12$                  ; 0561
       50         20          A5  000001FF     8F  C1  00148        ADDL3   #511, SYSTEMBASE, R0                      ; 0562
       67                     50  000001FF     8F  CB  00151        BICL3   #511, R0, LNK$GL_DEFCLU+76                ;
                              04               11      00159        BRB     13$                                       ;
                              67         20    A5  D0  0015B  12$:  MOVL    SYSTEMBASE, LNK$GL_DEFCLU+76              ; 0563
                              50  00000000G    00  D0  0015F  13$:  MOVL    LNK$GL_FVMLST, R0                         ; 0567
                              51               67  D0  00166        MOVL    LNK$GL_DEFCLU+76, R1                      ; 0565
                   40000000   8F               51  D1  00169        CMPL    R1, #1073741824                           ;
                              1D               1F      00170        BLSSU   15$                                       ;
                   80000000   8F               51  D1  00172        CMPL    R1, #-2147483648                          ; 0566
                              0A               1F      00179        BLSSU   14$                                       ;
                          04  A0  80000000     8F  D0  0017B        MOVL    #-2147483648, 4(R0)                       ; 0567
                              0D               11      00183        BRB     16$                                       ;
                          04  A0  40000000     8F  D0  00185  14$:  MOVL    #1073741824, 4(R0)                        ; 0568
                              03               11      0018D        BRB     16$                                       ; 0566
                                         04    A0  D4  0018F  15$:  CLRL    4(R0)                                     ; 0569
                                  00000000G    00  D5  00192  16$:  TSTL    FIRSTCLUSTER                              ; 0576
                                         12    12      00198        BNEQ    17$                                       ;
                                         BC    A7  D5  0019A        TSTL    LNK$GL_DEFCLU+8                           ; 0578
                                         0D    12      0019D        BNEQ    17$                                       ;
                                  00000000G    8F  DD  0019F        PUSHL   #LIN$_NOMODS                              ; 0580
                   00000000G  00               01  FB  001A5        CALLS   #1, LIB$STOP                              ;
```

```
04            64             03 E0 001AC 17$:   BBS     #3, LNK$GL_CTLMSK, 18$               ; 0590
04            64             02 E1 001B0        BBC     #2, LNK$GL_CTLMSK, 19$               ;
          02  A4             04 8A 001B4 18$:   BICB2   #4, LNK$GL_CTLMSK+2                  ; 0591
05            64             06 E0 001B8 19$:   BBS     #6, LNK$GL_CTLMSK, 20$               ; 0593
5F        02  A4             02 E1 001BC        BBC     #2, LNK$GL_CTLMSK+2, 23$             ;
0D        02  A4             02 E0 001C1 20$:   BBS     #2, LNK$GL_CTLMSK+2, 21$             ; 0596
                  00000000G  8F DD 001C6        PUSHL   #LIN$_TRACIGN                        ; 0597
    00000000G 00             01 FB 001CC        CALLS   #1, LIB$SIGNAL                       ;
           34 A5             20 D0 001D3 21$:   MOVL    #32, INFILFLGS                       ; 0599
                             65 B5 001D7        TSTW    DEBUG_DESC                           ; 0600
                             15 13 001D9        BEQL    22$                                  ;
                             55 DD 001DB        PUSHL   R5                                   ; 0602
    00000000V EF             01 FB 001DD        CALLS   #1, INPUTFILE                        ;
                      F8     A5 9F 001E4        PUSHAB  RLF_CONTEXT                          ; 0603
    00000000G 00             01 FB 001E7        CALLS   #1, LIB$FIND_FILE_END                ;
                             30 11 001EE        BRB     23$                                  ; 0600
                             5E DD 001F0 22$:   PUSHL   SP                                   ; 0609
                      08     AE 9F 001F2        PUSHAB  SYMVAL                               ;
                    0150     C6 9F 001F5        PUSHAB  LNK$GT_IMGSTA                        ;
    00000000G 00             03 FB 001F9        CALLS   #3, LNK$SEARCH                       ;
                             5E DD 00200        PUSHL   SP                                   ; 0610
                      08     AE 9F 00202        PUSHAB  SYMVAL                               ;
                    0150     C6 9F 00205        PUSHAB  LNK$GT_IMGSTA                        ;
    00000000G 00             03 FB 00209        CALLS   #3, LNK$INSERT                       ;
                      04     AE DD 00210        PUSHL   SYMVAL                               ; 0611
    00000000G 00             01 FB 00213        CALLS   #1, LNK$INSUDFSYM                    ;
                  00000000G  00 B6 0021A        INCW    LNK$GW_NSYMBOLS                      ; 0612
          01  A4             10 8A 00220 23$:   BICB2   #16, LNK$GL_CTLMSK+1                 ; 0616
    00000000V EF             00 FB 00224        CALLS   #0, LNK$COLLECTPSCT                  ; 0618
                      BC     A7 D5 0022B        TSTL    LNK$GL_DEFCLU+8                      ; 0620
                             21 13 0022E        BEQL    25$                                  ;
              50  00000000G  00 D0 00230        MOVL    LNK$GL_LASTCLU, R0                   ; 0622
              60         B4  A7 9E 00237        MOVAB   LNK$GL_DEFCLU, (R0)                  ;
          B8  A7             50 D0 0023B        MOVL    R0, LNK$GL_DEFCLU+4                  ; 0623
                             08 12 0023F        BNEQ    24$                                  ;
          B8  A7  00000000G  00 9E 00241        MOVAB   LNK$GL_CLULST, LNK$GL_DEFCLU+4       ; 0624
    00000000G 00         B4  A7 9E 00249 24$:   MOVAB   LNK$GL_DEFCLU, LNK$GL_LASTCLU        ; 0625
16        03  A4             02 E1 00251 25$:   BBC     #2, LNK$GL_CTLMSK+3, 26$             ; 0628
          11            FE88 C5 E8 00256        BLBS    UNIVSTAR_VALID, 26$                  ; 0629
                  00000000G  8F DD 0025B        PUSHL   #LIN$_UNIVSTAR                       ; 0631
    00000000G 00             01 FB 00261        CALLS   #1, LIB$SIGNAL                       ;
          03  A4             04 8A 00268        BICB2   #4, LNK$GL_CTLMSK+3                  ; 0632
                                04 0026C 26$:   RET                                          ; 0635

; Routine Size:  621 bytes,    Routine Base:  $CODE$ + 0000

;  525          0636  1
```

```
;  527        0637  1 routine clierror (desc,errorcode) : novalue =
;  528        0638  1
;  529        0639  1 !+
;  530        0640  1 !
;  531        0641  1 !   This routine is called by result parser if it detects any error in this
;  532        0642  1 !   command. reqestdesc is the address of the current parameter descriptor and
;  533        0643  1 !   errorcode is the encoded reason for the error.
;  534        0644  1 !
;  535        0645  1 !-
;  536        0646  2 begin
;  537        0647  2 map
;  538        0648  2         desc : ref block [,byte] ;                          ! Pointer to a block
;  539        0649  2
;  540        0650  2 signal_stop (lin$_cmdsyntax, 0,lin$_linerr, 1,desc [dsc$w_length], .errorcode) ;
;  541        0651  2
;  542        0652  1 end ;



                                         0000 00000 CLIERROR:
                                                          .WORD   Save nothing                       ; 0637
                        7E        04  AC  7D 00002        MOVQ    DESC, -(SP)                        ; 0650
                                      01  DD 00006        PUSHL   #1                                 ;
                           00000000G  8F  DD 00008        PUSHL   #LIN$_LINERR                       ;
                                      7E  D4 0000E        CLRL    -(SP)                              ;
                           00000000G  8F  DD 00010        PUSHL   #LIN$_CMDSYNTAX                    ;
                 00000000G  00        06  FB 00016        CALLS   #6, LIB$STOP                       ;
                                          04 0001D        RET                                        ; 0652

; Routine Size:  30 bytes,    Routine Base:  $CODE$ + 026D


;  543        0653  1
```

```
1 routine call_tparse (string,states,keys) =
1
1 !-
1 !
1 !     This routine calls TPARSE given the string, states and keys
1 !
1 !     Inputs:
1 !
1 !             STRING          address of string descriptor for string to parse
1 !             STATES          address of tparse states table
1 !             KEYS            address of tparse keys table
1 !
1 !-
2 begin
2 map
2           string  : ref block [,byte] ;
2 local
2           status,
2           tpablock : block [tpa$k_length0,byte] ;
2
2 !
2 !     Initialize the tparse block
2 !
2 ch$fill (0, tpa$k_length0, tpablock) ;
2
2 tpablock [tpa$l_count]      =  tpa$k_count0 ;
2 tpablock [tpa$l_options]    =  tpa$m_abbrev ;
2 tpablock [tpa$l_stringcnt] = .string [dsc$w_length] ;
2 tpablock [tpa$l_stringptr] = .string [dsc$a_pointer] ;
2
2 !
2 ! Initialize own storage used
2 !
2 tpdata = 0 ;
2
2 return lib$tparse (tpablock, .states, .keys)
1 end ;
```

```
                                          003C 00000 CALL_TPARSE:
                                                           .WORD   Save R2,R3,R4,R5                      ; 0654
                             5E        24  C2 00002        SUBL2   #36, SP
       24          00        6E        00  2C 00005        MOVC5   #0, (SP), #0, #36, TPABLOCK           ; 0677
                                       6E     0000A
                             6E        08  D0 0000B        MOVL    #8, TPABLOCK                          ; 0679
                     04      AE        02  D0 0000E        MOVL    #2, TPABLOCK+4                        ; 0680
                             50    04  AC  D0 00012        MOVL    STRING, R0                            ; 0681
                     08      AE        60  3C 00016        MOVZWL  (R0), TPABLOCK+8
                     0C      AE    04  A0  D0 0001A        MOVL    4(R0), TPABLOCK+12                    ; 0682
                                00000000'  EF  D4 0001F    CLRL    TPDATA                                ; 0687
                             7E    08  AC  7D 00025        MOVQ    STATES, -(SP)                         ; 0689
                                   08  AE  9F 00029        PUSHAB  TPABLOCK
                00000000G    00        03  FB 0002C        CALLS   #3, LIB$TPARSE
                                           04 00033        RET                                           ; 0690
```

```
; Routine Size:  52 bytes,    Routine Base:  $CODE$ + 028B

;  582          0691  1
```

```
1 routine specialimage : novalue =
1
1 !+
1 !
1 !   This routine is called by the CLI if the system or shareable image qualifier
1 !   is seen on the primary output parameter.
1 !
1 !   If a value is present on the qualifier, the CLI is re-called through CLIADDR
1 !   to convert the value which then sets the base address of the default cluster
1 !   of the image being linked.  If no value is present the base address is set
1 !   to the lower end of system or process space.  In both cases the stack size is
1 !   set to zero.  A system image will not but a shareable image will have a
1 !   header.
1 !
1 !-
2 begin
2 local
2         symbolstring     : vector [sym$c_maxlng+1, byte],         ! Base address symbol storeage
2         nament           : ref block [,byte],                     ! Pointer to symbol name block
2         symtabent        : ref block [,byte] ;                    ! Pointer to table entry for it
2
2
2 lnk$gw_stack = 0 ;                                                ! Init stack
2
2 if .lnk$gl_ctlmsk [lnk$v_shr]                                     ! If a shareable image
2 then
3     begin
3     cli$get_value (sd_shareable, share_desc) ;                    !   then get name now
3     return ;                                                      ! All done with /SHAREABLE...
2     end ;
2
2 if  .lnk$gl_ctlmsk [lnk$v_sys]                                    ! If a system image
2 then lnk$gl_fvmlst [fvm$l_address] = system_space ;               ! Set first available virtual memory
2
2 if not cli$get_value (sd_system, token_desc)                      ! If there was no qualifier value
2 then return ;                                                     ! Then all done
2
2 lnk$gl_defclu [clu$v_based]  = true ;                             ! Set the bit for fixed in address space
2 lnk$gl_ctlmsk [lnk$v_picimg] = false ;                            ! And image is not pic
2 basedflag                    = true ;
2
2 if .(.token_desc [dsc$a_pointer])<0,8>  eql  %c'%'                ! If value for the base was prefixed with
2 then                                                              !   a %, then we definitely have a numeric
3     begin                                                         !   base -- find out the radix
3     local       status ;
3
3     selectone .(.token_desc [dsc$a_pointer]+1)<0,8> of            ! Base specified as %?
3         set
3         [%c'X'] : status = lib$cvt_htb (.token_desc [dsc$w_length] - 2        ! Value is in hex
3                                        ,.token_desc [dsc$a_pointer] + 2
3                                        , systembase
3                                        ) ;
3         [%c'D'] : status = lib$cvt_dtb (.token_desc [dsc$w_length] - 2        ! Value is in decimal
3                                        ,.token_desc [dsc$a_pointer] + 2
3                                        , systembase
3                                        ) ;
3         [%c'O'] : status = lib$cvt_otb (.token_desc [dsc$w_length] - 2        ! Value is in octal
```

```
;  641       0749 3                                  ,.token_desc [dsc$a_pointer] + 2
;  642       0750 3                                  , systembase
;  643       0751 3                                  ) ;
;  644       0752 3          [otherwise] : clierror (token_desc, lin$_illqualval) ;        ! Anything else is illegal
;  645       0753 3          tes ;
;  646       0754 3
;  647       0755 3      if   not .status                                           ! If error occurred in value conversion,
;  648       0756 3      then clierror (token_desc, .status) ;                      !  report it here
;  649       0757 3      end
;  650       0758 2  else
;  651       0759 2      !
;  652       0760 2      !   Didn't explicitly specify a radix -- assume decimal
;  653       0761 2      !
;  654       0762 2      if not lib$cvt_dtb (.token_desc [dsc$w_length], .token_desc [dsc$a_pointer], systembase)
;  655       0763 2      then
;  656       0764 3          begin                                                  ! However... if puked on conversion    ,
;  657       0765 4          if (symbolstring [0] = minu (.token_desc [dsc$w_length]  !    attempt, then we must assume
;  658       0766 4                                      ,sym$c_maxlng              !    a symbolic base value
;  659       0767 4                                      )
;  660       0768 3              ) eql 0
;  661       0769 3          then
;  662       0770 3              return ;
;  663       0771 3
;  664       0772 3          ch$move (.symbolstring [0]
;  665       0773 3                  ,.token_desc [dsc$a_pointer]                    ! Otherwise build a counted
;  666       0774 3                  , symbolstring [1]                              ! Ascii string that long
;  667       0775 3                  ) ;
;  668       0776 3          lnk$search (symbolstring [0], symtabent, nament) ;     ! Then search for place to insert it
;  669       0777 3          lnk$insert (symbolstring [0], symtabent, nament) ;     ! and do it
;  670       0778 3          lnk$insudfsym (.symtabent) ;                           ! Insert undefined symbol
;  671       0779 3          lnk$gw_nsymbols             = .lnk$gw_nsymbols + 1 ;   ! Count as one more symbol
;  672       0780 3          lnk$gl_defclu [clu$l_base]    = .symtabent ;           ! Save the symbol table adress
;  673       0781 3          lnk$gl_defclu [clu$v_symbas] = true ;                  ! Flag as symbolic
;  674       0782 2          end ;                                                  ! And all done till it is defined
;  675       0783 2  return true
;  676       0784 1  end ;


                                         0FFC 00000 SPECIALIMAGE:
                                                    .WORD    Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11          ; 0692
                     5B          A9   AF   9E 00002          MOVAB    CLIERROR, R11                                 ;
                     5A  00000000G    00   9E 00006          MOVAB    LIB$CVT_DTB, R10                              ;
                     59  00000000G    00   9E 0000D          MOVAB    CLI$GET_VALUE, R9                             ;
                     58  00000000G    00   9E 00014          MOVAB    LNK$GL_DEFCLU+88, R8                          ;
                     57  00000000G    00   9E 0001B          MOVAB    LNK$GL_CTLMSK, R7                             ;
                     56  00000000'    EF   9E 00022          MOVAB    TOKEN_DESC, R6                                ;
                     5E               28   C2 00029          SUBL2    #40, SP                                       ;
                         00000000G    00   B4 0002C          CLRW     LNK$GW_STACK                                  ; 0714
        0D           67               02   E1 00032          BBC      #2, LNK$GL_CTLMSK, 1$                         ; 0716
                                 F8   A6   9F 00036          PUSHAB   SHARE_DESC                                    ; 0719
                         00000000'    EF   9F 00039          PUSHAB   SD_SHAREABLE                                  ;
                     69               02   FB 0003F          CALLS    #2, CLI$GET_VALUE                             ;
                                           04 00042          RET                                                    ; 0718
        0F           67               03   E1 00043 1$:      BBC      #3, LNK$GL_CTLMSK, 2$                         ; 0723
```

```
          50 00000000G    00    D0 00047        MOVL    LNK$GL_FVMLST, R0                   ; 0724
     04   A0 80000000     8F    D0 0004E        MOVL    #-2147483648, 4(R0)                 ;
                          56    DD 00056  2$:   PUSHL   R6                                  ; 0726
             00000000'    EF    9F 00058        PUSHAB  SD_SYSTEM                           ;
                    69    02    FB 0005E        CALLS   #2, CLI$GET_VALUE                   ;
                    01    50    E8 00061        BLBS    R0, 3$                              ;
                                04 00064        RET                                         ;
                    68    01    88 00065  3$:   BISB2   #1, LNK$GL_DEFCLU+88                ; 0729
          02        A7    02    8A 00068        BICB2   #2, LNK$GL_CTLMSK+2                 ; 0730
          0C        A6    01    D0 0006C        MOVL    #1, BASEDFLAG                       ; 0731
                    52    04 A6 D0 00070        MOVL    TOKEN_DESC+4, R2                    ; 0733
                    25    62    91 00074        CMPB    (R2), #37                           ;
                    67    12       00077        BNEQ    8$                                  ;
                    50    01 A2 9A 00079        MOVZBL  1(R2), R0                           ; 0738
          58        8F    50    91 0007D        CMPB    R0, #88                             ; 0740
                          15    12 00081        BNEQ    4$                                  ;
                       08 A6    9F 00083        PUSHAB  SYSTEMBASE                          ;
                       02 A2    9F 00086        PUSHAB  2(R2)                               ; 0741
                    7E    66    3C 00089        MOVZWL  TOKEN_DESC, -(SP)                   ; 0740
                    6E    02    C2 0008C        SUBL2   #2, (SP)                            ;
          00000000G 00    03    FB 0008F        CALLS   #3, LIB$CVT_HTB                     ;
                          3D    11 00096        BRB     7$                                  ;
          44        8F    50    91 00098  4$:   CMPB    R0, #68                             ; 0744
                          11    12 0009C        BNEQ    5$                                  ;
                       08 A6    9F 0009E        PUSHAB  SYSTEMBASE                          ;
                       02 A2    9F 000A1        PUSHAB  2(R2)                               ; 0745
                    7E    66    3C 000A4        MOVZWL  TOKEN_DESC, -(SP)                   ; 0744
                    6E    02    C2 000A7        SUBL2   #2, (SP)                            ;
                    6A    03    FB 000AA        CALLS   #3, LIB$CVT_DTB                     ;
                          26    11 000AD        BRB     7$                                  ;
          4F        8F    50    91 000AF  5$:   CMPB    R0, #79                             ; 0748
                          15    12 000B3        BNEQ    6$                                  ;
                       08 A6    9F 000B5        PUSHAB  SYSTEMBASE                          ;
                       02 A2    9F 000B8        PUSHAB  2(R2)                               ; 0749
                    7E    66    3C 000BB        MOVZWL  TOKEN_DESC, -(SP)                   ; 0748
                    6E    02    C2 000BE        SUBL2   #2, (SP)                            ;
          00000000G 00    03    FB 000C1        CALLS   #3, LIB$CVT_OTB                     ;
                          0B    11 000C8        BRB     7$                                  ;
             00000000G    8F    DD 000CA  6$:   PUSHL   #LIN$_ILLQUALVAL                    ; 0752
                          56    DD 000D0        PUSHL   R6                                  ;
                    6B    02    FB 000D2        CALLS   #2, CLIERROR                        ;
                    69    50    E8 000D5  7$:   BLBS    STATUS, 10$                         ; 0755
                          50    DD 000D8        PUSHL   STATUS                              ; 0756
                          56    DD 000DA        PUSHL   R6                                  ;
                    6B    02    FB 000DC        CALLS   #2, CLIERROR                        ;
                                04 000DF        RET                                         ; 0733
                       08 A6    9F 000E0  8$:   PUSHAB  SYSTEMBASE                          ; 0762
                          52    DD 000E3        PUSHL   R2                                  ;
                    7E    66    3C 000E5        MOVZWL  TOKEN_DESC, -(SP)                   ;
                    6A    03    FB 000E8        CALLS   #3, LIB$CVT_DTB                     ;
                    53    50    E8 000EB        BLBS    R0, 10$                             ;
                    50    66    3C 000EE        MOVZWL  TOKEN_DESC, R0                      ; 0765
                    1F    50    B1 000F1        CMPW    R0, #31                             ;
                          03    1B 000F4        BLEQU   9$                                  ;
                    50    1F    D0 000F6        MOVL    #31, R0                             ;
          08        AE    50    90 000F9  9$:   MOVB    R0, SYMBOLSTRING                    ;
                          50    D5 000FD        TSTL    R0                                  ; 0768
```

```
                                 40  13 000FF          BEQL    10$                                        :
                   50       08   AE  9A 00101          MOVZBL  SYMBOLSTRING, R0                           : 0772
  09   AE     04   B6            50  28 00105          MOVC3   R0, @TOKEN_DESC+4, SYMBOLSTRING+1          : 0774
                                 5E  DD 0010B          PUSHL   SP                                         : 0776
                            08   AE  9F 0010D          PUSHAB  SYMTABENT                                  :
                            10   AE  9F 00110          PUSHAB  SYMBOLSTRING                               :
          00000000G  00          03  FB 00113          CALLS   #3, LNK$SEARCH                             :
                                 5E  DD 0011A          PUSHL   SP                                         : 0777
                            08   AE  9F 0011C          PUSHAB  SYMTABENT                                  :
                            10   AE  9F 0011F          PUSHAB  SYMBOLSTRING                               :
          00000000G  00          03  FB 00122          CALLS   #3, LNK$INSERT                             :
                            04   AE  DD 00129          PUSHL   SYMTABENT                                  : 0778
          00000000G  00          01  FB 0012C          CALLS   #1, LNK$INSUDFSYM                          :
                     00000000G   00  B6 00133          INCW    LNK$GW_NSYMBOLS                            : 0779
                 F4  A8     04   AE  D0 00139          MOVL    SYMTABENT, LNK$GL_DEFCLU+76                : 0780
                     68          02  88 0013E          BISB2   #2, LNK$GL_DEFCLU+88                       : 0781
                                     04 00141  10$:    RET                                                : 0784

; Routine Size:  322 bytes,    Routine Base:  $CODE$ + 02BF

;  677          0785  1
```

```
0786 1 global routine lnk$setlibrin  (option_flag)  : novalue =
0787 1
0788 1 !+
0789 1 !
0790 1 !  This routine is called by the CLI if the LIBRARY input file qualifier is
0791 1 !  seen on an input parameter.
0792 1 !
0793 1 !-
0794 2 begin
0795 2 builtin
0796 2         nullparameter ;
0797 2 local
0798 2         called_by_option ;
0799 2
0800 2 called_by_option = not nullparameter (1) ;                   ! It is important to know
0801 2                                                               !    whether or not we were called
0802 2 if    not  .called_by_option                                  !    by lnk$procoptns in order to
0803 2 then  if   cli$present (sd_include)                           !    know who should do the
0804 2       then infilflgs = .infilflgs or fdb$m_libextr ;          !    parsing -- the cli or me
0805 2
0806 2 if         .lnk$gl_clulst [0] eql 0                           ! If no preceding clusters
0807 2     and  .lnk$gl_curclu [clu$l_fstfdb] eql 0                  ! And no files in current cluster
0808 2     and (.infilflgs and fdb$m_libextr) eql 0                  ! And not explicit inclusion from the
0809 2 then
0810 2     signal_stop (lin$_firstmod) ;                             ! File give up with an error
0811 2
0812 2 typeadr = ch$ptr (libfiltype) ;                               ! Set address of default file type
0813 2 return ;
0814 1 end ;


                               0004 00000        .ENTRY  LNK$SETLIBRIN, Save R2                    ; 0786
          52 00000000'   EF      9E 00002        MOVAB   INFILFLGS, R2                             ;
                         6C      95 00009        TSTB    (AP)                                      ; 0800
                         05      12 0000B        BNEQ    1$                                        ;
          50             01      D0 0000D        MOVL    #1, R0                                    ;
                         09      11 00010        BRB     2$                                        ;
                         50      D4 00012 1$:    CLRL    R0                                        ;
                  04     AC      D5 00014        TSTL    4(AP)                                     ;
                         02      12 00017        BNEQ    2$                                        ;
                         50      D6 00019        INCL    R0                                        ;
          50             50      D2 0001B 2$:    MCOML   R0, CALLED_BY_OPTION                      ;
          14             50      E8 0001E        BLBS    CALLED_BY_OPTION, 3$                      ; 0802
             00000000'   EF      9F 00021        PUSHAB  SD_INCLUDE                                ; 0803
     00000000G  00       01      FB 00027        CALLS   #1, CLI$PRESENT                           ;
                04       50      E9 0002E        BLBC    R0, 3$                                    ;
                62       40  8F  88 00031        BISB2   #64, INFILFLGS                            ; 0804
             00000000G   00      D5 00035 3$:    TSTL    LNK$GL_CLULST                             ; 0806
                         1D      12 0003B        BNEQ    4$                                        ;
          50 00000000G   00      D0 0003D        MOVL    LNK$GL_CURCLU, R0                         ; 0807
                  08     A0      D5 00044        TSTL    8(R0)                                     ;
                         11      12 00047        BNEQ    4$                                        ;
     0D         62       06      E0 00049        BBS     #6, INFILFLGS, 4$                         ; 0808
             00000000G   8F      DD 0004D        PUSHL   #LIN$_FIRSTMOD                            ; 0810
```

```
                    00000000G  00            01  FB 00053         CALLS   #1, LIB$STOP                       ;
                          FC   A2 00000000'  EF  9E 0005A 4$:     MOVAB   LIBFILTYPE, TYPEADR                ; 0812
                                                 04 00062         RET                                        ; 0814

; Routine Size:  99 bytes,    Routine Base:  $CODE$ + 0401


;  708          0815  1
```

```
;  710          0816   1 routine optionfile  : novalue =
;  711          0817   1
;  712          0818   1 !+
;  713          0819   1 !
;  714          0820   1 !   Set the default file type for link time options
;  715          0821   1 !
;  716          0822   1 !-
;  717          0823   2 begin
;  718          0824   2 bind    optfiltype = uplit (byte ('.OPT')) ;
;  719          0825   2
;  720          0826   2 typeadr = ch$ptr (optfiltype) ;
;  721          0827   2 return ;
;  722          0828   1 end ;


                                                          .PSECT  $PLIT$,NOWRT,NOEXE,2

                               54  50  4F  2E  001AC P.ABZ:  .ASCII  \.OPT\                              ;

                                                    OPTFILTYPE=          P.ABZ

                                                          .PSECT  $CODE$,NOWRT,2

                                           0000 00000 OPTIONFILE:
                                                          .WORD   Save nothing                        ; 0816
                  00000000'   EF 00000000'   EF   9E 00002     MOVAB   OPTFILTYPE, TYPEADR                 ; 0826
                                                 04 0000D     RET                                         ; 0828

; Routine Size:  14 bytes,    Routine Base:  $CODE$ + 0464


;  723          0829   1
```

```
;  725          0830  1 glcbal routine lnk$setshrblin  : novalue =
;  726          0831  1
;  727          0832  1 !+
;  728          0833  1 !
;  729          0834  1 !   Set the default file type for shareable images.
;  730          0835  1 !
;  731          0836  1 !-
;  732          0837  2 begin
;  733          0838  2 typeadr = ch$ptr (shrfiltype) ;                         ! Set default file type pointer
;  734          0839  2
;  735          0840  2 return ;
;  736          0841  1 end ;



                                      0000 00000          .ENTRY  LNK$SETSHRBLIN, Save nothing                        ; 0830
              00000000'  EF 00000000'  EF  9E 00002          MOVAB   SHRFILTYPE, TYPEADR                                 ; 0838
                                        04 0000D          RET                                                         ; 0841

; Routine Size:  14 bytes,    Routine Base:  $CODE$ + 0472


;  737          0842  1
```

```
;  739      0843  1 routine setuserlib  =
;  740      0844  1
;  741      0845  1 !+
;  742      0846  1 !
;  743      0847  1 !   This routine is called by the CLI for the /USERLIBRARY qualifier.
;  744      0848  1 !   The default is /USERLIBRARY= (PROCESS, GROUP, SYSTEM).  If there
;  745      0849  1 !   is a list of logical name tables, only those tables specified will
;  746      0850  1 !   be searched.
;  747      0851  1 !
;  748      0852  1 !-
;  749      0853  2 begin
;  750      0854  2
;  751      0855  2 lnk$gl_ctlmsk [lnk$v_usrlib] = true ;
;  752      0856  2 while cli$get_value (sd_userlibrary, token_desc)              ! Loop for all specified
;  753      0857  3 do  begin
;  754      0858  3     if .token_desc [dsc$w_length] neq 0                       ! If a string was there
;  755      0859  3     then
;  756      0860  3         if not call_tparse (token_desc [dsc$w_length], userlib_states, userlib_keys)
;  757      0861  3         then
;  758      0862  3             clierror (.token_desc, lin$_illqualval) ;           ! No return from this
;  759      0863  2     end ;
;  760      0864  2
;  761      0865  2 if  .lnk$gl_ulibmask eql 0                                    ! If nothing found
;  762      0866  2 then lnk$gl_ulibmask = 7                                      ! Then enable all logical name tables
;  763      0867  2 else if  .lnk$gl_ulibmask lss 0                               !  will be lss 0 if /USERLIB=NONE
;  764      0868  2     then lnk$gl_ulibmask = 0 ;
;  765      0869  2
;  766      0870  2 return true ;                                                 !
;  767      0871  1 end ;                                                         ! of setuserlib



                                        000C 00000 SETUSERLIB:
                                                         .WORD   Save R2,R3                                ; 0843
                   53 00000000'    EF   9E 00002         MOVAB   LNK$GL_ULIBMASK, R3                       ;
                   52 00000000'    EF   9E 00009         MOVAB   TOKEN_DESC, R2                            ;
     00000000G     00        40    8F   88 00010         BISB2   #64, LNK$GL_CTLMSK+2                      ; 0855
                                   52   DD 00018 1$:     PUSHL   R2                                        ; 0856
                      00000000'    EF   9F 0001A         PUSHAB  SD_USERLIBRARY                            ;
     00000000G     00              02   FB 00020         CALLS   #2, CLI$GET_VALUE                         ;
                   29              50   E9 00027         BLBC    R0, 2$                                    ;
                                   62   B5 0002A         TSTW    TOKEN_DESC                                ; 0858
                                   EA   13 0002C         BEQL    1$                                        ;
                      00000000'    EF   9F 0002E         PUSHAB  USERLIB_KEYS                              ; 0860
                      00000000'    EF   9F 00034         PUSHAB  USERLIB_STATES                            ;
                                   52   DD 0003A         PUSHL   R2                                        ;
          FDCA     CF              03   FB 0003C         CALLS   #3, CALL_TPARSE                           ;
                   D4              50   E8 00041         BLBS    R0, 1$                                    ;
                      00000000G    8F   DD 00044         PUSHL   #LIN$_ILLQUALVAL                          ; 0862
                                   62   DD 0004A         PUSHL   TOKEN_DESC                                ;
          FD9C     CF              02   FB 0004C         CALLS   #2, CLIERROR                              ;
                                   C5   11 00051         BRB     1$                                        ; 0856
                   50              63   D0 00053 2$:     MOVL    LNK$GL_ULIBMASK, R0                       ; 0865
                                   05   12 00056         BNEQ    3$                                        ;
                   63              07   D0 00058         MOVL    #7, LNK$GL_ULIBMASK                       ; 0866
```

```
                              04  11 0005B       BRB     4$                                    ;
                              02  18 0005D 3$:   BGEQ    4$                                    ; 0867
                              63  D4 0005F       CLRL    LNK$GL_ULIBMASK                       ; 0868
                  50          01  D0 00061 4$:   MOVL    #1, R0                                ; 0870
                                  04 00064       RET                                           ; 0871

; Routine Size:  101 bytes,    Routine Base:  $CODE$ + 0480


;  768          0872  1
```

```
1 global routine inputfile  (desc, option_flag) : novalue =
1
1 !+
1 !
1 !   This routine is called by the CLI for each file in the command input
1 !   parameter list.
1 !
1 !   The routine allocates an internal file descriptor block and then, using the
1 !   input file fab allocated in the command processing driver rouine attempts
1 !   to open the specified file.
1 !
1 !   If this is successful, the auxiliary filename block part of our FDB is
1 !   initialized by RMS.  Other useful fields of the FDB  (such as the file flags)
1 !   are initialized and then the FDB is added to the FDB list.  The file is then
1 !   closed, so that an open by file id may be performed  on the file hereafter.
1 !
1 !   Failure to open the file at this point is fatal and  the link aborts.
1 !
1 !
1 !       REQSTDESC       is the address of the input parameter descriptor being
1 !                       used by the CLI.
1 !
1 !       INFILFLGS       contains the settings of the input file control flags
1 !                       as determined by the CLI and qualifiers used.
1 !
1 !-
2 begin
2     routine get_module  ( module_dsc ) =
2     !
2     !   This routine plucks the modules out of the include list passed here from
2     !   LNK$PROCOPTNS and returns the individual module name descriptors to
2     !   inputfile
2     !
3     begin
3     local       length,
3                 address,
3                 remaining_length ;
3
3     map         module_dsc    : ref block [,byte] ;
3
3     bind        include_list = .lnk$gl_inclst :  block [,byte] ;         ! Here is the list...
3
3     if .listptr lss .include_list [dsc$a_pointer]                      ! If failure to find a module
3     then                                                               !     zeroed the pointer, then
3         return false ;                                                 !     quit here
3
3     address = .listptr ;                                               ! Current module is already pointed to
3     if ch$fail (listptr = ch$find_ch                                   ! If we fail to find a delimiter
3                         (remaining_length =
4                                 (.include_list [dsc$w_length] -
5                                 (.listptr - .include_list [dsc$a_pointer])
3                                 ),
3                         .listptr
3                         ','
3                         )
3             )
3     then length = .remaining_length                                    !  then the length is what was left
3     else length = .listptr - .address  ;                               !  otherwise the length is the difference
```

```
;  827          0930  3                                                          !    in the pointers
;  828          0931  3         if  .length lss 0                                ! Make sure module name length was > 0
;  829          0932  3         then length = 0 ;
;  830          0933  3
;  831          0934  3         listptr = .listptr + 1 ;                         ! Move ptr 1 char past delimiter
;  832          0935  3                                                          ! (tparse cleans out blanks and tabs)
;  833          0936  3         str$copy_r ( .module_dsc, length, .address) ;    ! Do this because inputfile expects a
;  834          0937  3                                                          ! dynamic descriptor
;  835          0938  3         return true ;
;  836          0939  2         end ;



                                          003C 00000 GET_MODULE:
                                                     .WORD   Save R2,R3,R4,R5                                 ; 0900
                  55 00000000'  EF   9E 00002        MOVAB   LISTPTR, R5                                      ;
                  5E            04   C2 00009        SUBL2   #4, SP                                           ;
                  50 00000000G  00   D0 0000C        MOVL    LNK$GL_INCLST, R0                                ; 0913
                  52            65   D0 00013        MOVL    LISTPTR, R2                                      ; 0915
              04  A0            52   D1 00016        CMPL    R2, 4(R0)                                        ;
                                3D   19 0001A        BLSS    5$                                               ;
                  54            52   D0 0001C        MOVL    R2, ADDRESS                                      ; 0919
          51  04  A0            52   C3 0001F        SUBL3   R2, 4(R0), R1                                    ; 0923
                  53            60   3C 00024        MOVZWL  (R0), REMAINING_LENGTH                           ;
                  53            51   C0 00027        ADDL2   R1, REMAINING_LENGTH                             ;
          62      53            2C   3A 0002A        LOCC    #44, REMAINING_LENGTH, (R2)                      ; 0921
                                02   12 0002E        BNEQ    1$                                               ;
                                51   D4 00030        CLRL    R1                                               ;
                  65            51   D0 00032 1$:    MOVL    R1, LISTPTR                                      ;
                                05   12 00035        BNEQ    2$                                               ; 0927
                  6E            53   D0 00037        MOVL    REMAINING_LENGTH, LENGTH                         ; 0928
                                04   11 0003A        BRB     3$                                               ;
          6E      65            54   C3 0003C 2$:    SUBL3   ADDRESS, LISTPTR, LENGTH                         ; 0929
                                02   18 00040 3$:    BGEQ    4$                                               ; 0931
                                6E   D4 00042        CLRL    LENGTH                                           ; 0932
                                65   D6 00044 4$:    INCL    LISTPTR                                          ; 0934
                                54   DD 00046        PUSHL   ADDRESS                                          ; 0936
                          04    AE   9F 00048        PUSHAB  LENGTH                                           ;
                          04    AC   DD 0004B        PUSHL   MODULE_DSC                                       ;
             00000000G    00    03   FB 0004E        CALLS   #3, STR$COPY_R                                   ;
                          50    01   D0 00055        MOVL    #1, R0                                           ; 0938
                                04      00058        RET                                                      ;
                          50    D4      00059 5$:    CLRL    R0                                               ; 0939
                                04      0005B        RET                                                      ;

; Routine Size:  92 bytes,    Routine Base:  $CODE$ + 04E5


;  837          0940  2
```

```
2 bind
2       sysinput_desc   = $descriptor ('SYS$INPUT') : $bblock,
2       include_list    = .lnk$gl_inclst            :  block [,byte] ;
2 literal
2       maxoptreclen    = 132 ;                                 ! Maximum length of option file records
2 map
2       desc            : ref block [,byte] ;                   ! Really a pointer
2 builtin
2       nullparameter ;
2 local
2       dnm_desc        : block [8, byte],                      ! Default filename descriptor
2       enm_desc        : block [8, byte],                      ! LIB$FIND_FILE expanded file name
2       fnm_desc        : ref block [,byte],                    ! File name descriptor in fdb
2       fildesblk       : ref block [,byte],                    ! Pointer to fdb
2       auxfnb          : ref block [,byte],                    ! Auxiliary file name block pointer
2       input_fab       : block [fab$c_bln, byte],              ! Fab for the file opens.
2       optionrab       : block [rab$c_bln, byte],              ! Rab for option file input stream
2       stvcode,                                                ! Stv value for error output
2       errorcode,                                              ! Rms error code
2       called_by_option ;                                      ! Set if called by lnk$_procoptions
2
2 called_by_option = not nullparameter (2) ;
2
2 !
2 !
2 !     Allocate an FDB
2 !
2 !
2 lnk$allofdb  ( fildesblk ) ;                                  ! Go allocate new file descriptor
2 auxfnb       = fildesblk [fdb$t_auxfnb] ;                     ! Point to filename block part
2 fnm_desc     = fildesblk [fdb$q_filename] ;                   ! Point to file name descriptor
2 copy_desc    (desc, fnm_desc) ;                               ! Copy initial file name descriptor
2
2 fiidesblk [fdb$b_filflgs] = .infilflgs and not fdb$m_newudf ;  ! Copy the flags turning off bit 0
2
2 !
2 !
2 !     If this is a library witih explicit module extraction, build the FDB's module list.
2 !     Call  CLI$GET_VALUE or GET_MODULE, based on whether or not the file is in an options file.
2 !
2 !
2 if  (.infilflgs and fdb$m_libextr) neq 0                      ! If explicit module extraction
2 then
3     begin
3     local       prevmodule : initial (0) ;                    ! Pointer to previous module name block
3
3     if .called_by_option                                      ! If called by LNK$PROCOPTNS, we must
3     then listptr = .include_list [dsc$a_pointer] ;            ! set up listptr
3
4     while  (if not .called_by_option  then cli$get_value (sd_include,module_desc)
4                                       else get_module (module_desc)
4            )
4     do  begin
4         local   currentmodule   : ref block [,byte] ;         !  Ptr to current module name block
4
4         lnk$alloblk (.module_desc [dsc$w_length]+5,currentmodule) ; !  Allocate space to save name
4         if .prevmodule eql 0                                  ! If this is the first module,
```

```
4              then  fildesblk [fdb$l_omdlst] = currentmodule [0,0,0,0]     ! then set the pointer in the fdb
4              else .prevmodule              = currentmodule [0,0,0,0] ;   ! Otherwise, point prev to current
4
4              prevmodule                    = currentmodule [0,0,0,0] ;   ! Reset for next pass
4              currentmodule [0,0,32,0]      = 0 ;                         ! Zero forward pointer
4              currentmodule [4,0, 8,0]      = .module_desc [dsc$w_length] ; ! Move in module length and name
4
4              ch$move (.module_desc [dsc$w_length], .module_desc [dsc$a_pointer], currentmodule [5,0,8,0]) ;
3              end ;
2          end ;
2
2
2   !
2   !   Setup up buffer descriptors and call LIB$FIND_FILE to setup an expanded filename
2   !   consistent with the stickiness rules for related file names and search lists.
2   !
2   infilflgs = 0 ;                                                        ! Reset file flags
2
2   dnm_desc [dsc$w_length]  =  typelen ;                                  ! Setup default filename descriptor
2   dnm_desc [dsc$b_class]   =  dsc$k_class_z ;                            ! for call to LIB$FIND_FILE
2   dnm_desc [dsc$b_dtype]   =  dsc$k_dtype_z ;
2   dnm_desc [dsc$a_pointer] = .typeadr ;
2
2   enm_desc [dsc$w_length]  =  0 ;                                        ! Init result filename descriptor
2   enm_desc [dsc$b_class]   =  dsc$k_class_d ;                            ! for call to LIB$FIND_FILE
2   enm_desc [dsc$b_dtype]   =  dsc$k_dtype_t ;
2   enm_desc [dsc$a_pointer] =  0 ;
2
2   errorcode = lib$find_file (.desc, enm_desc                             ! Determine resultant name
2                             , rlf_context, dnm_desc
2                             , 0, stvcode, %ref(3)
2   !                         , 0, input_fab [fab$l_stv], %ref(3)
2                             ) ;
2   if .rlf_context neq 0
2   then
3       begin
3       local       rss     : initial (.auxfnb [nam$b_rss]),               ! Original NAM$B_RSS
3                   rsa     : initial (.auxfnb [nam$l_rsa]),               ! Original NAM$L_RSA
3                   ess     : initial (.auxfnb [nam$b_ess]),               ! Original NAM$B_ESS
3                   esa     : initial (.auxfnb [nam$l_esa]) ;              ! Original NAM$L_ESS
3
3       ch$move (nam$c_bln, .rlf_context [fab$l_nam], .auxfnb) ;                 ! Copy NAM block
3       ch$move (.enm_desc [dsc$w_length], .enm_desc [dsc$a_pointer], .esa) ; ! Copy the name
3
3       auxfnb [nam$l_wcc] =  0 ;                                          ! Init ptr's we won't use
3       auxfnb [nam$l_rlf] =  0 ;
3       auxfnb [nam$l_rsa] = .rsa ;                                        ! Re-setup resultant name area
3       auxfnb [nam$b_rss] = .rss ;
3       auxfnb [nam$b_rsl] =  0 ;
3       auxfnb [nam$l_esa] = .esa ;                                        ! Re-setup expanded name area
3       auxfnb [nam$b_ess] = .ess ;
3       auxfnb [nam$b_esl] = .enm_desc [dsc$w_length] ;
3
3       fildesblk [fdb$w_usrnamlen] = .auxfnb [nam$b_esl] ;                ! Copy the user's filename
3       fildesblk [fdb$l_usrnamadr] = .esa ;                               ! Into the fdb also
3       copy_desc (enm_desc, fnm_desc) ;                                   ! Get a better file name descriptor
```

```
  3                                                                       ! for error message reporting
P 3     $fab_init   (fab =  input_fab                                     ! Intialize the fab
P 3                 ,fac =  get                                           ! For input
P 3                 ,fop =  sqo
P 3                 ,dna = .typeadr
P 3                 ,dns =  typelen
P 3                 ,fna = .esa
P 3                 ,fns = .auxfnb [nam$b_esl]
P 3                 ,nam = .auxfnb
P 3                 ,shr = (upi,get,put)
  3                 ) ;
  3     input_fab [fab$l_stv] = .stvcode ;                                ! Restore LIB$FIND_FILE stv value
  2     end ;
  2 !
  2 !
  2 !   If the parse/setup was successful, open the file to obtain the resultant filename,
  2 !   the file i.d., etc.
  2 !
  2 !
  2 if .errorcode
  3 then begin
  3     if .fildesblk [fdb$v_librl]                                       ! Library file
  4     then begin                                                        ! Library open by name block
  4         auxfnb [nam$b_rsl] = .auxfnb [nam$b_esl] ;                    ! Setup resultant name for map
  4         auxfnb [nam$b_rss] = .auxfnb [nam$b_ess] ;                    !  contents
  4         auxfnb [nam$l_rsa] = .auxfnb [nam$l_esa] ;
  4         lnk$gl_curfil = .fildesblk ;                                  ! Set ptr to current fdb for the temporary
  4         errorcode     = lnk$openlib (.auxfnb, .an_open_fdb) ;         ! benefit of LNK$OPENLIB
  4         lnk$gl_curfil = 0 ;                                           ! Must now clear it -- causes problems
  4         end                                                           ! later if non-zero
  4     else begin                                                        ! Not a library file
  4         $getjpi (itmlst = lnk$gt_jpilst) ;                            ! Close a file if we're near the limit
  4         if    .lnk$gl_filesleft leq 3
  4         then  if    .an_open_fdb neq 0
  4               then   lnk$closefile (.an_open_fdb) ;
  4
  4         if .fildesblk [fdb$v_shr]                                     ! If shareable image, gotta do:
  5         then begin
  5             input_fab [fab$v_sqo]    = false ;                        ! Not sequential only
  5             input_fab [fab$v_bro]    = true ;                         ! Record and block operations
  5             input_fab [fab$v_upi]    = true ;                         ! Shared access
  5             input_fab [fab$v_shrget] = true ;                         ! Set for shared gets
  5             input_fab [fab$v_shrput] = true ;                         !  and puts  (linker doesn't write to
  4             end ;                                                     !  it, but must set so we can read it
  4                                                                       !  in case installed /write)
  4         errorcode              = $open (fab=input_fab) ;              ! Do regular $open
  4         fildesblk [fdb$w_ifi] = .input_fab [fab$w_ifi] ;              ! Remember IFI
  3         end ;
  3     ch$move (dsc$c_s_bln, lnk$filnamdsc (input_fab), .fnm_desc) ; ! Copy better file name descriptor
  2     end ;
  2 !
  2 !
  2 !   If ERRORCODE has its low bit set at this point then the
  2 !   file open was successful.
  2 !
  2 !
```

```
   2 if not .errorcode                                                  ! If either of the opens was a failure,
   2 then stvcode = .input_fab [fab$l_stv]                              ! Then save status and flag
   3 else begin                                                         ! Setup input related name for output
   3      if  .lnk$gl_inrelnam eql 0                                    ! files if not done yet
   3      then if setup_inrelnam (1,.fnm_desc, dflt_inrel, dflt_name)
   3           then lnk$gl_inrelnam = dflt_inrel ;
   3
   3      if  not .called_by_option
   3      then last_fnm_desc = .fnm_desc ;
   3
   3      if  not .fildesblk [fdb$v_option]
   4      then begin
   4           !
   4           !
   4           !    Not an OPTIONS file.  Simply stick the FDB at the end
   4           !    of the current cluster FDB list.
   4           !
   4
   4           local lastfdb : ref block [,byte] initial (.lnk$gl_curclu [clu$l_lstfdb]) ;
   4
   4           lastfdb [fdb$l_nxtfdb]        = .fildesblk ;          ! Put FDB at end of current cluster
   4           lnk$gl_curclu [clu$l_lstfdb]  = .fildesblk ;          ! Remember end of list.
   4           end
   4      else begin
   4           !
   4           !
   4           !    OPTIONS file.  Connect to it and process the records inside it.
   4           !    The entire contents of the OPTIONS file begins a new context for
   4           !    related  filename processing - begining with 'SYS$DISK:[]'.
   4           !
   4
   4           bind  namdvi = auxfnb [nam$t_dvi] : vector  [,byte] ; ! Addr of nam block dvi
   4
   4           lnk$gl_sysinput_flag =                                ! Setup "sysinput" flag
   5             begin
   5             if   .bblock  [input_fab [fab$l_dev], dev$v_trm]     ! Is device a terminal?
   6             or (.auxfnb [nam$v_ppf]                              ! A process permanent file?
   6                  and
   6                 ch$eql (.sysinput_desc [dsc$w_length]            ! SYS$INPUT ?
   6                        ,.sysinput_desc [dsc$a_pointer]
   6                        ,.namdvi [0], namdvi [1], 0
   6                        )
   5                )
   5             then  1
   5             else  0
   4             end ;
   4
P  4           $rab_init       (rab = optionrab
P  4                           ,fab = input_fab
P  4                           ,usz = maxoptreclen
P  4                           ,rac = seq
P  4                           ,rop = loc
   4                           ) ;
   5           if  not (errorcode = $connect (rab=optionrab))        ! Connect the options file for reading
   4           then stvcode        = .optionrab [rab$l_stv]
   5           else begin                                            ! And if connected ok
   5                local  save_context : initial (.rlf_context) ;   ! Save current context
   5
```

```
5                    lnk$alloblk (maxoptreclen, optionrab [rab$l_ubf]) ; ! Allocate buffer for records
5                                                                      ! crossing block boundaries
5                    typeadr        =  ch$ptr (defiletype) ;           ! Reset the default file type
5                    rlf_context    =  0 ;                             ! Start new related name context
5                    lnk$procoptns (.fnm_desc, infilflgs, optionrab) ;! Go process it
5                    rlf_context    = .save_context ;                  ! Restore related name context
5                    lnk$gl_curclu =  lnk$gl_defclu ;                  ! And reset default as current
5                    lnk$dealblk (maxoptreclen, .optionrab [rab$l_ubf]) ; ! And deallocate
4                    end ;
3                end ;
2            end ;
2
2   if not .errorcode                                                  ! Report any errors
2   then  signal_stop (lin$_openin, 1, .fnm_desc, .errorcode, .stvcode) ;
2
2   !
2   !
2   !    For performance reasons, only close the file if we have to -- if its an Options file
2   !    or a shareable image
2   !
2   !
2   if .fildesblk [fdb$v_option]   or  .fildesblk [fdb$v_imglib]
2   then lnk$closefile (.fildesblk)
2   else an_open_fdb  = .fildesblk ;
2
2   typeadr = ch$ptr (defiletype) ;                                    ! Reset the default file type
2   return ;
1   end ;


                                                          .PSECT  $PLIT$,NOWRT,NOEXE,2

    54  55  50  4E  49  24  53  59  53  001B0 P.ACB:     .ASCII  \SYS$INPUT\                          ;
                                        001B9            .BLKB   3
                              00000009  001BC P.ACA:     .LONG   9                                    ;
                              00000000' 001C0            .ADDRESS P.ACB                               ;

                                             SYSINPUT_DESC=        P.ACA
                                                         .EXTRN  SYS$GETJPI, SYS$OPEN
                                                         .EXTRN  SYS$CONNECT

                                                         .PSECT  $CODE$,NOWRT,2

                                 0FFC 00000              .ENTRY  INPUTFILE, Save R2,R3,R4,R5,R6,R7,R8,R9,-    ; 0873
                                                                 R10,R11                                      ;
                      5E      FF40  CE  9E 00002         MOVAB   -192(SP), SP                                 ;
                      52 00000000G  00  D0 00007         MOVL    LNK$GL_INCLST, R2                            ; 0943
                      02            6C  91 0000E         CMPB    (AP), #2                                     ; 0962
                                    05  1E 00011         BGEQU   1$                                           ;
                      58            01  D0 00013         MOVL    #1, R8                                       ;
                                    09  11 00016         BRB     2$                                           ;
                                    58  D4 00018 1$:     CLRL    R8                                           ;
                              08    AC  D5 0001A         TSTL    8(AP)                                        ;
                                    02  12 0001D         BNEQ    2$                                           ;
                                    58  D6 0001F         INCL    R8                                           ;
                      58            58  D2 00021 2$:     MCOML   R8, CALLED_BY_OPTION                         ;
```

```
                         10  AE  9F 00024      PUSHAB  FILDESBLK                               ; 0969
        00000000V  EF        01  FB 00027      CALLS   #1, LNK$ALLOFDB                         ;
                   5A    10  AE  D0 0002E      MOVL    FILDESBLK, R10                          ; 0970
                   56    26  AA  9E 00032      MOVAB   38(R10), AUXFNB                         ;
                   57    14  AA  9E 00036      MOVAB   20(R10), FNM_DESC                       ; 0971
                   59    04  AC  D0 0003A      MOVL    DESC, R9                                ; 0972
                   67        69  7D 0003E      MOVQ    (R9), (FNM_DESC)                        ;
                   6E    0A  AA  9E 00041      MOVAB   10(R10), (SP)                           ; 0974
   00  BE 00000000'  EF      01  8B 00045      BICB3   #1, INFILFLGS, @0(SP)                   ;
       79 00000000'  EF      06  E1 0004E      BBC     #6, INFILFLGS, 9$                       ; 0982
                             5B  D4 00056      CLRL    PREVMODULE                              ; 0984
                   08        58  E9 00058      BLBC    CALLED_BY_OPTION, 3$                    ; 0987
        00000000'  EF    04  A2  D0 0005B      MOVL    4(R2), LISTPTR                          ; 0988
               0C  AE        58  D2 00063 3$:  MCOML   CALLED_BY_OPTION, 12(SP)                ; 0990
                   15    0C  AE  E9 00067 4$:  BLBC    12(SP), 5$                              ;
                 00000000'   EF  9F 0006B      PUSHAB  MODULE_DESC                             ;
                 00000000'   EF  9F 00071      PUSHAB  SD_INCLUDE                              ;
        00000000G  00        02  FB 00077      CALLS   #2, CLI$GET_VALUE                       ;
                             0B  11 0007E      BRB     6$                                      ;
                 00000000'   EF  9F 00080 5$:  PUSHAB  MODULE_DESC                             ; 0991
             FF19  CF        01  FB 00086      CALLS   #1, GET_MODULE                          ;
                   41        50  E9 0008B 6$:  BLBC    R0, 9$                                  ;
                         14  AE  9F 0008E      PUSHAB  CURRENTMODULE                           ; 0996
                7E 00000000' EF  3C 00091      MOVZWL  MODULE_DESC, -(SP)                      ;
                   6E        05  C0 00098      ADDL2   #5, (SP)                                ;
        00000000G  00        02  FB 0009B      CALLS   #2, LNK$ALLOBLK                         ;
                   50    14  AE  D0 000A2      MOVL    CURRENTMODULE, R0                       ; 0998
                             5B  D5 000A6      TSTL    PREVMODULE                              ; 0997
                             06  12 000A8      BNEQ    7$                                      ;
               04  AA        50  D0 000AA      MOVL    R0, 4(R10)                              ; 0998
                             03  11 000AE      BRB     8$                                      ;
                   6B        50  D0 000B0 7$:  MOVL    R0, (PREVMODULE)                        ; 0999
                   5B        50  D0 000B3 8$:  MOVL    R0, PREVMODULE                          ; 1001
                             60  D4 000B6      CLRL    (R0)                                    ; 1002
               04  A0 00000000'  EF  90 000B8  MOVB    MODULE_DESC, 4(R0)                      ; 1003
   05  A0 00000000'  FF 00000000'  EF  28 000C0  MOVC3 MODULE_DESC, @MODULE_DESC+4, 5(R0)      ; 1005
                             98  11 000CD      BRB     4$                                      ; 0990
                 00000000'   EF  D4 000CF 9$:  CLRL    INFILFLGS                               ; 1015
               F8  AD        04  D0 000D5      MOVL    #4, DNM_DESC                            ; 1017
               FC  AD 00000000'  EF  D0 000D9  MOVL    TYPEADR, DNM_DESC+4                     ; 1020
               F0  AD 020E0000   8F  D0 000E1  MOVL    #34471936, ENM_DESC                     ; 1022
                         F4  AD  D4 000E9      CLRL    ENM_DESC+4                              ; 1025
               0C  AE        03  D0 000EC      MOVL    #3, 12(SP)                              ; 1029
                         0C  AE  9F 000F0      PUSHAB  12(SP)                                  ;
                         1C  AE  9F 000F3      PUSHAB  STVCODE                                 ; 1027
                             7E  D4 000F6      CLRL    -(SP)                                   ;
                         F8  AD  9F 000F8      PUSHAB  DNM_DESC                                ;
                 00000000'   EF  9F 000FB      PUSHAB  RLF_CONTEXT                             ;
                         F0  AD  9F 00101      PUSHAB  ENM_DESC                                ;
                             59  DD 00104      PUSHL   R9                                      ;
        00000000G  00        07  FB 00106      CALLS   #7, LIB$FIND_FILE                       ;
               0C  AE        50  D0 0010D      MOVL    R0, ERRORCODE                           ;
               50 00000000'  EF  D0 00111      MOVL    RLF_CONTEXT, R0                         ; 1032
                             03  12 00118      BNEQ    10$                                     ;
                         0085  31 0011A        BRW     11$                                     ;
               08  AE    02  A6  9A 0011D 10$: MOVZBL  2(AUXFNB), RSS                          ; 1035
                   5B    04  A6  D0 00122      MOVL    4(AUXFNB), RSA                          ; 1036
```

```
                    04   AE        0A   A6  9A 00126        MOVZBL  10(AUXFNB), ESS                       ; 1037
                         59        0C   A6  D0 0012B        MOVL    12(AUXFNB), ESA                       ; 1038
              66    28   B0      0060   8F  28 0012F        MOVC3   #96, @40(R0), (AUXFNB)                ; 1040
              69    F4   BD        F0   AD  28 00136        MOVC3   ENM_DESC, @ENM_DESC+4, (ESA)          ; 1041
                                   30   A6  D4 0013C        CLRL    48(AUXFNB)                            ; 1043
                                   10   A6  D4 0013F        CLRL    16(AUXFNB)                            ; 1044
                    04   A6             5B  D0 00142        MOVL    RSA, 4(AUXFNB)                        ; 1045
                    02   A6        08   AE  9B 00146        MOVZBW  RSS, 2(AUXFNB)                        ; 1046
                    0C   A6             59  D0 0014B        MOVL    ESA, 12(AUXFNB)                       ; 1048
                    0A   A6        04   AE  90 0014F        MOVB    ESS, 10(AUXFNB)                       ; 1049
                    0B   A6        F0   AD  90 00154        MOVB    ENM_DESC, 11(AUXFNB)                  ; 1050
                    0C   AA        0B   A6  9B 00159        MOVZBW  11(AUXFNB), 12(R10)                   ; 1052
                    10   AA             59  D0 0015E        MOVL    ESA, 16(R10)                          ; 1053
                         67        F0   AD  7D 00162        MOVQ    ENM_DESC, (FNM_DESC)                  ; 1054
  0050   8F         00   6E             00  2C 00166        MOVC5   #0, (SP), #0, #80, $RMS_PTR           ; 1065
                                   60   AE     0016D
                    60   AE      5003   8F  B0 0016F        MOVW    #20483, $RMS_PTR                      ;
                    64   AE        40   8F  9A 00175        MOVZBL  #64, $RMS_PTR+4                       ;
                    76   AE      4302   8F  B0 0017A        MOVW    #17154, $RMS_PTR+22                   ;
                    7F   AE             02  90 00180        MOVB    #2, $RMS_PTR+31                       ;
                    C8   AD             56  D0 00184        MOVL    AUXFNB, $RMS_PTR+40                   ;
                    CC   AD             59  D0 00188        MOVL    ESA, $RMS_PTR+44                      ;
                    D0   AD 00000000'   EF  D0 0018C        MOVL    TYPEADR, $RMS_PTR+48                  ;
                    D4   AD        0B   A6  90 00194        MOVB    11(AUXFNB), $RMS_PTR+52               ;
                    D5   AD             04  90 00199        MOVB    #4, $RMS_PTR+53                       ;
                    6C   AE        18   AE  D0 0019D        MOVL    STVCODE, INPUT_FAB+12                 ; 1066
                         03        0C   AE  E8 001A2 11$:   BLBS    ERRORCODE, 12$                        ; 1075
                                     0096   31 001A6        BRW     17$                                   ;
              2C    00   BE             01  E1 001A9 12$:   BBC     #1, @0(SP), 13$                       ; 1077
                    02   A6        0A   A6  B0 001AE        MOVW    10(AUXFNB), 2(AUXFNB)                 ; 1080
                    04   A6        0C   A6  D0 001B3        MOVL    12(AUXFNB), 4(AUXFNB)                 ; 1081
         00000000G  00                  5A  D0 001B8        MOVL    R10, LNK$GL_CURFIL                    ; 1082
                          00000000'     EF  DD 001BF        PUSHL   AN_OPEN_FDB                           ; 1083
                                        56  DD 001C5        PUSHL   AUXFNB                                ;
         00000000G  00                  02  FB 001C7        CALLS   #2, LNK$OPENLIB                       ;
                    0C   AE             50  D0 001CE        MOVL    R0, ERRORCODE                         ;
                          00000000G     00  D4 001D2        CLRL    LNK$GL_CURFIL                         ; 1084
                                        53  11 001D8        BRB     16$                                   ; 1077
                                        7E  7C 001DA 13$:   CLRQ    -(SP)                                 ; 1087
                                        7E  D4 001DC        CLRL    -(SP)                                 ;
                          00000000'     EF  9F 001DE        PUSHAB  LNK$GT_JPILST                         ;
                                        7E  7C 001E4        CLRQ    -(SP)                                 ;
                                        7E  D4 001E6        CLRL    -(SP)                                 ;
         00000000G  00                  07  FB 001E8        CALLS   #7, SYS$GETJPI                        ;
                         03 00000000'   EF  D1 001EF        CMPL    LNK$GL_FILESLEFT, #3                  ; 1088
                                        12  14 001F6        BGTR    14$                                   ;
                         50 00000000'   EF  D0 001F8        MOVL    AN_OPEN_FDB, R0                       ; 1089
                                        09  13 001FF        BEQL    14$                                   ;
                                        50  DD 00201        PUSHL   R0                                    ; 1090
         00000000G  00                  01  FB 00203        CALLS   #1, LNK$CLOSEFILE                     ;
              0B    00   BE             02  E1 0020A 14$:   BBC     #2, @0(SP), 15$                       ; 1092
                    64   AE        40   8F  8A 0020F        BICB2   #64, INPUT_FAB+4                      ; 1094
                    76   AE      4340   8F  A8 00214        BISW2   #17216, INPUT_FAB+23                  ; 1098
                                   60   AE  9F 0021A 15$:   PUSHAB  INPUT_FAB                             ; 1101
         00000000G  00                  01  FB 0021D        CALLS   #1, SYS$OPEN                          ;
                    0C   AE             50  D0 00224        MOVL    R0, ERRORCODE                         ;
                    24   AA        62   AE  B0 00228        MOVW    INPUT_FAB+2, 36(R10)                  ; 1102
```

```
                              60    AE  9F 0022D 16$:    PUSHAB  INPUT_FAB                                          ; 1104
           00000000G  00            01  FB 00230         CALLS   #1, LNK$FILNAMDSC                                  ;
        67            60            08  28 00237         MOVC3   #8, (R0), (FNM_DESC)                               ;
                      07    0C      AE  E8 0023B         BLBS    ERRORCODE, 18$                                     ; 1112
              18  AE        6C      AE  D0 0023F 17$:    MOVL    INPUT_FAB+12, STVCODE                              ; 1113
                            4E      11     00244         BRB     21$                                                ;
                     00000000'      EF  D5 00246 18$:    TSTL    LNK$GL_INRELNAM                                    ; 1115
                            25      12     0024C         BNEQ    19$                                                ;
                     00000000'      EF  9F 0024E         PUSHAB  DFLT_NAME                                          ; 1116
                     00000000'      EF  9F 00254         PUSHAB  DFLT_INREL                                         ;
                            57      DD     0025A         PUSHL   FNM_DESC                                           ;
                            01      DD     0025C         PUSHL   #1                                                 ;
           00000000V  EF            04  FB 0025E         CALLS   #4, SETUP_INRELNAM                                 ;
                      0B            50  E9 00265         BLBC    R0, 19$                                            ;
           00000000'  EF 00000000'  EF  9E 00268         MOVAB   DFLT_INREL, LNK$GL_INRELNAM                        ; 1117
                      07            58  E8 00273 19$:    BLBS    CALLED_BY_OPTION, 20$                              ; 1119
           00000000'  EF            57  D0 00276         MOVL    FNM_DESC, LAST_FNM_DESC                            ; 1120
        14            00    BE      04  E0 0027D 20$:    BBS     #4, @0(SP), 22$                                    ; 1122
                      50 00000000G  00  D0 00282         MOVL    LNK$GL_CURCLU, R0                                  ; 1130
                      51    0C      A0  D0 00289         MOVL    12(R0), LASTFDB                                    ;
                      61            5A  D0 0028D         MOVL    R10, (LASTFDB)                                     ; 1132
              0C      A0            5A  D0 00290         MOVL    R10, 12(R0)                                        ; 1133
                            6A      11     00294 21$:    BRB     26$                                                ; 1122
                      50    14      A6  9E 00296 22$:    MOVAB   20(AUXFNB), R0                                     ; 1143
              18  E0  AD            02  E0 0029A         BBS     #2, INPUT_FAB+64, 23$                              ; 1147
                      19    36      A6  E9 0029F         BLBC    54(AUXFNB), 24$                                    ; 1148
                      51            60  9A 002A3         MOVZBL  (R0), R1                                           ; 1152
        51            00 00000000'  FF 00000000'  EF  2D 002A6  CMPC5  SYSINPUT_DESC, @SYSINPUT_DESC+4, #0, R1, -  ;
                            01      A0     002B3                 1(R0)                                              ;
                            05      12     002B5         BNEQ    24$                                                ;
              50            01      D0     002B7 23$:    MOVL    #1, R0                                             ; 1147
                            02      11     002BA         BRB     25$                                                ;
                            50      D4     002BC 24$:    CLRL    R0                                                 ;
           00000000'  EF            50  D0 002BE 25$:    MOVL    R0, LNK$GL_SYSINPUT_FLAG                           ; 1146
  0044  8F            00    6E      00  2C 002C5         MOVC5   #0, (SP), #0, #68, $RMS_PTR                        ; 1163
                            1C      AE     002CC                                                                    ;
              1C  AE      4401      8F  B0 002CE         MOVW    #17409, $RMS_PTR                                   ;
              20  AE  00010000      8F  D0 002D4         MOVL    #65536, $RMS_PTR+4                                 ;
                            3A      AE  94 002DC         CLRB    $RMS_PTR+30                                        ;
              3C  AE        84      8F  9B 002DF         MOVZBW  #132, $RMS_PTR+32                                  ;
              58  AE        60      AE  9E 002E4         MOVAB   INPUT_FAB, $RMS_PTR+60                             ;
                            1C      AE  9F 002E9         PUSHAB  OPTIONRAB                                          ; 1164
           00000000G  00            01  FB 002EC         CALLS   #1, SYS$CONNECT                                    ;
                  0C  AE            50  D0 002F3         MOVL    R0, ERRORCODE                                      ;
                      07    0C      AE  E8 002F7         BLBS    ERRORCODE, 27$                                     ;
              18  AE        28      AE  D0 002FB         MOVL    OPTIONRAB+12, STVCODE                              ; 1165
                            58      11     00300 26$:    BRB     28$                                                ;
                  52 00000000'      EF  D0 00302 27$:    MOVL    RLF_CONTEXT, SAVE_CONTEXT                          ; 1167
                            40      AE  9F 00309         PUSHAB  OPTIONRAB+36                                       ; 1169
                      7E    84      8F  9A 0030C         MOVZBL  #132, -(SP)                                        ;
           00000000G  00            02  FB 00310         CALLS   #2, LNK$ALLOBLK                                    ;
           00000000'  EF 00000000'  EF  9E 00317         MOVAB   DEFILETYPE, TYPEADR                                ; 1171
                     00000000'      EF  D4 00322         CLRL    RLF_CONTEXT                                        ; 1172
                            1C      AE  9F 00328         PUSHAB  OPTIONRAB                                          ; 1173
                     00000000'      EF  9F 0032B         PUSHAB  INFILFLGS                                          ;
                            57      DD     00331         PUSHL   FNM_DESC                                           ;
           00000000G  00            03  FB 00333         CALLS   #3, LNK$PROCOPTNS                                  ;
```

```
          00000000'  EF              52  D0 0033A        MOVL    SAVE_CONTEXT, RLF_CONTEXT         ; 1174
          00000000G  00 00000000G    00  9E 00341        MOVAB   LNK$GL_DEFCLU, LNK$GL_CURCLU      ; 1175
                             40      AE  DD 0034C        PUSHL   OPTIONRAB+36                      ; 1176
                     7E      84      8F  9A 0034F        MOVZBL  #132, -(SP)                       ;
          00000000G  00              02  FB 00353        CALLS   #2, LNK$DEALBLK                   ;
                     17      0C      AE  E8 0035A 28$:   BLBS    ERRORCODE, 29$                    ; 1181
                             18      AE  DD 0035E        PUSHL   STVCODE                           ; 1182
                             10      AE  DD 00361        PUSHL   ERRORCODE                         ;
                                     57  DD 00364        PUSHL   FNM_DESC                          ;
                                     01  DD 00366        PUSHL   #1                                ;
                          00000000G  8F  DD 00368        PUSHL   #LIN$_OPENIN                      ;
          00000000G  00              05  FB 0036E        CALLS   #5, LIB$STOP                      ;
       04        00  BE              04  E0 00375 29$:   BBS     #4, @0(SP), 30$                   ; 1190
                     0B      0B      AA  E9 0037A        BLBC    11(R10), 31$                      ;
                                     5A  DD 0037E 30$:   PUSHL   R10                               ; 1191
          00000000G  00              01  FB 00380        CALLS   #1, LNK$CLOSEFILE                 ;
                                     07  11 00387        BRB     32$                               ;
          00000000'  EF              5A  D0 00389 31$:   MOVL    R10, AN_OPEN_FDB                  ; 1192
          00000000'  EF 00000000'    EF  9E 00390 32$:   MOVAB   DEFILETYPE, TYPEADR               ; 1194
                                     04  0039B           RET                                       ; 1196

; Routine Size:  924 bytes,    Routine Base:  $CODE$ + 0541


; 1095          1197  1
```

```
global routine lnk$allofdb (fdbpointer) : novalue =

!+
!
!  This routine allocates some more dynmaic memory for file descriptor blocks
!  and RMS resultant strings and inserts into an auxiliary file name block the
!  length and address for resultant and expanded strings returned on opens on
!  any call.
!
!  If there is not sufficient resultant string area for the maximum, more is
!  allocated.
!
!  The routine checks rsl of the previously allocated auxiliary file name block
!  and if non zero, reduces the remaining string area by that actually used.
!  If still zero the area is reduced by the maximum size before the decision
!  to allocate more is made.
!
!-
begin
own
        prevfnb         : ref block [,byte] ;                     ! Previous auxiliary filename block

local
        fdblock         : ref block [,byte],                      ! File descriptor block allocated
        rsareduction,                                             !
        temprsapointer ;                                          ! For allocating resultant
                                                                  ! String buffers
map
        fdbpointer      : ref vector ;                            ! Passed an address at wich to store

if .prevfnb neq 0                                                 ! If one has been allocated
then
    begin                                                         ! Previously, try to use the actual
    if  (rsareduction = .prevfnb [nam$b_rsl]) eql 0               ! Amount consumed to reduce
    then rsareduction = .prevfnb [nam$b_rss] ;                    ! The available area but if none

    rsaremsize = .rsaremsize - .rsareduction ;                    ! Consumed, reduce by the total
    nextrsa    = .nextrsa + .rsareduction ;                       ! And update the address of next available
    end ;

if .rsaremsize lssu nam$c_maxrss
then                                                              ! Not enough space in resltant string
    begin                                                         !  buffer for maximum one
    lnk$alloblk  (rsa_allosize, temprsapointer) ;                 !  Go allocate some more
    if .temprsapointer eql .nextrsa + .rsaremsize                 ! If it was allocated continguously
    then                                                          !  with previous one
        rsaremsize = .rsaremsize + rsa_allosize                   !  then merge
    else
        begin                                                     ! Else, use new block and forget
        rsaremsize = rsa_allosize ;                               ! previous area
        nextrsa    = .temprsapointer ;
        end ;
    end ;

lnk$alloblk (fdb$c_size+nam$c_bln, fdblock) ;                     ! Allocate file descriptor
fdbpointer [0] = .fdblock ;                                       ! Return its address
prevfnb        =  fdblock [fdb$t_auxfnb] ;                        ! Get address of the file name block part
```

```
; 1154          1255  2 ch$fill (0,fdb$c_size+nam$c_bln, .fdblock) ;                          ! Zero the fdb
; 1155          1256  2
; 1156        P 1257  2 $nam_init (nam = .prevfnb                                             ! Initialize the nam block
; 1157        P 1258  2            ,rsa = .nextrsa
; 1158        P 1259  2            ,rss = nam$c_maxrss
; 1159        P 1260  2            ,esa = .nextrsa
; 1160        P 1261  2            ,ess = nam$c_maxrss
; 1161          1262  2            ) ;
; 1162          1263  2 return ;                                                              ! And that's it
; 1163          1264  1 end ;


                                                            .PSECT  $OWN$,NOEXE,2

                                              001DB         .BLKB   1
                                              001DC PREVFNB:.BLKB   4


                                                            .PSECT  $CODE$,NOWRT,2

                                         01FC 00000         .ENTRY  LNK$ALLOFDB, Save R2,R3,R4,R5,R6,R7,R8     ; 1198
                       58 00000000G  00   9E 00002          MOVAB   LNK$ALLOBLK, R8                            ;
                       57 00000000'  EF   9E 00009          MOVAB   RSAREMSIZE, R7                             ;
                       5E            08   C2 00010          SUBL2   #8, SP                                     ;
                       50        28  A7   D0 00013          MOVL    PREVFNB, R0                                ; 1228
                                     11   13 00017          BEQL    2$                                         ;
                       51        03  A0   9A 00019          MOVZBL  3(R0), RSAREDUCTION                        ; 1231
                                     04   12 0001D          BNEQ    1$                                         ;
                       51        02  A0   9A 0001F          MOVZBL  2(R0), RSAREDUCTION                        ; 1232
                       67            51   82 00023 1$:      SUBB2   RSAREDUCTION, RSAREMSIZE                   ; 1234
                 F0    A7            51   C0 00026          ADDL2   RSAREDUCTION, NEXTRSA                      ; 1235
                 FF    8F            67   91 0002A 2$:      CMPB    RSAREMSIZE, #255                           ; 1238
                                     24   1E 0002E          BGEQU   4$                                         ;
                                     5E   DD 00030          PUSHL   SP                                         ; 1241
                       7E      017E  8F   3C 00032          MOVZWL  #382, -(SP)                                ;
                       68            02   FB 00C37          CALLS   #2, LNK$ALLOBLK                            ;
                       50            67   9A 0003A          MOVZBL  RSAREMSIZE, R0                             ; 1242
                       50        F0  A7   C0 0003D          ADDL2   NEXTRSA, R0                                ;
                       50            6E   D1 00041          CMPL    TEMPRSAPOINTER, R0                         ;
                                     06   12 00044          BNEQ    3$                                         ;
                       67        7E  8F   80 00046          ADDB2   #126, RSAREMSIZE                           ; 1244
                                     08   11 0004A          BRB     4$                                         ;
                       67        7E  8F   90 0004C 3$:      MOVB    #126, RSAREMSIZE                           ; 1247
                 F0    A7            6E   D0 00050          MOVL    TEMPRSAPOINTER, NEXTRSA                    ; 1248
                                 04  AE   9F 00054 4$:      PUSHAB  FDBLOCK                                    ; 1252
                       7E        86  8F   9A 00057          MOVZBL  #134, -(SP)                                ;
                       68            02   FB 0005B          CALLS   #2, LNK$ALLOBLK                            ;
                 04    BC        04  AE   D0 0005E          MOVL    FDBLOCK, @FDBPOINTER                       ; 1253
        28   A7  04    AE            26   C1 00063          ADDL3   #38, FDBLOCK, PREVFNB                      ; 1254
0086  8F         00    6E            00   2C 00069          MOVC5   #0, (SP), #0, #134, @FDBLOCK               ; 1255
                                 04  BE      00070                                                     ;
                       56        28  A7   D0 00072          MOVL    PREVFNB, R6                                ; 1262
0060  8F         00    6E            00   2C 00076          MOVC5   #0, (SP), #0, #96, (R6)                    ;
                                     66      0007D                                                     ;
                       66      6002  8F   B0 0007E          MOVW    #24578, (R6)                               ;
```

```
                02   A6          01  8E 00083          MNEGB   #1, 2(R6)                          ;
                04   A6     F0   A7  D0 00087          MOVL    NEXTRSA, 4(R6)                     ;
                0A   A6          01  8E 0008C          MNEGB   #1, 10(R6)                         ;
                0C   A6     F0   A7  D0 00090          MOVL    NEXTRSA, 12(R6)                    ;
                                 04     00095          RET                                        ; 1264

; Routine Size:  150 bytes,    Routine Base:  $CODE$ + 08DD


; 1164          1265    1
```

```
; 1166      1266  1 global routine lnk$clunamcmp  (keyname, node) =
; 1167      1267  1
; 1168      1268  1 !+
; 1169      1269  1 !
; 1170      1270  1 !   This routine compares an ascic keyname with the current entry
; 1171      1271  1 !
; 1172      1272  1 !-
; 1173      1273  2 begin
; 1174      1274  2 map
; 1175      1275  2         keyname : ref vector [,byte],
; 1176      1276  2         node    : ref block [,byte] ;
; 1177      1277  2 local
; 1178      1278  2         clu     : ref block [,byte] ;
; 1179      1279  2
; 1180      1280  2 clu = .node [node$l_ptr] ;                               ! Point to cluster descriptor
; 1181      1281  2 return ch$compare (.keyname [0], keyname [1]
; 1182      1282  2                 ,.clu [clu$b_namlng], clu [clu$t_name]
; 1183      1283  2                 )
; 1184      1284  1 end ;


                                              001C 00000            .ENTRY  LNK$CLUNAMCMP, Save R2,R3,R4          ; 1266
                            50          08 AC   D0 00002            MOVL    NODE, R0                               ; 1280
                            50          0A A0   D0 00006            MOVL    10(R0), CLU                            ;
                            52          04 AC   D0 0000A            MOVL    KEYNAME, R2                            ; 1281
                            53             62   9A 0000E            MOVZBL  (R2), R3                               ;
                            51          5C A0   9A 00011            MOVZBL  92(CLU), R1                            ; 1282
                            54             01   D0 00015            MOVL    #1, R4                                 ;
          51        00        01           A2   53 2D 00018         CMPC5   R3, 1(R2), #0, R1, 93(CLU)             ;
                                        5D A0      0001E                                                       ;
                                           03   1A 00020            BGTRU   1$                                     ;
                            54             01   D9 00022            SBWC    #1, R4                                 ;
                            50             54   D0 00025 1$:        MOVL    R4, R0                                 ;
                                                04 00028            RET                                            ; 1284

; Routine Size:  41 bytes,    Routine Base:  $CODE$ + 0973


; 1185      1285  1
```

```
; 1187          1286  1 global routine lnk$compare_pdd  (keyname,pdd) =
; 1188          1287  1
; 1189          1288  1 !+
; 1190          1289  1 !
; 1191          1290  1 !  Routine to compare ascii psect name with current entry
; 1192          1291  1 !
; 1193          1292  1 !-
; 1194          1293  2 begin
; 1195          1294  2 map
; 1196          1295  2         keyname : ref vector [,byte],
; 1197          1296  2         pdd     : ref block [,byte] ;
; 1198          1297  2
; 1199          1298  2 return ch$compare (.keyname [0], keyname [1]
; 1200          1299  2                   ,(.pdd [pdd$b_namlng] and not %x'80'), pdd [pdd$t_name]
; 1201          1300  2                   )
; 1202          1301  1 end ;


                                          001C 00000        .ENTRY  LNK$COMPARE_PDD, Save R2,R3,R4              ; 1286
                          52        04  AC  D0 00002        MOVL    KEYNAME, R2                                 ; 1298
                          53            62  9A 00006        MOVZBL  (R2), R3
                          50        08  AC  D0 00009        MOVL    PDD, R0                                     ; 1299
      51        0F    A0  07            00  EF 0000D        EXTZV   #0, #7, 15(R0), R1
                          54            01  D0 00013        MOVL    #1, R4
      51              00        01  A2  53  2D 00016        CMPC5   R3, 1(R2), #0, R1, 16(R0)
                                    10  A0     0001C
                                        03  1A 0001E        BGTRU   1$
                          54            01  D9 00020        SBWC    #1, R4
                          50            54  D0 00023 1$:    MOVL    R4, R0
                                            04 00026        RET                                         ; 1301

; Routine Size:  39 bytes,    Routine Base:  $CODE$ + 099C


; 1203          1302  1
```

```
; 1205      1303  1 global routine lnk$alloc_pdd  (keyname,retadr) =
; 1206      1304  1
; 1207      1305  1 !+
; 1208      1306  1 !
; 1209      1307  1 !  Allocate a new pdd block
; 1210      1308  1 !
; 1211      1309  1 !-
; 1212      1310  2 begin
; 1213      1311  2 map
; 1214      1312  2         keyname : ref vector [,byte],
; 1215      1313  2         retadr  : ref vector [,long] ;
; 1216      1314  2 local
; 1217      1315  2         pdd     : ref block [,byte] ;
; 1218      1316  2
; 1219      1317  2 lnk$alloblk (pdd$c_size+.keyname [0], pdd) ;                 ! Allocate the pdd
; 1220      1318  2
; 1221      1319  2 ch$fill (0, pdd$c_size, .pdd) ;                              ! Zero it
; 1222      1320  2
; 1223      1321  2 pdd [pdd$w_flags] = psc$m_optpsc ;                           ! Set defined by option flag
; 1224      1322  2 pdd [pdd$b_align] = -1 ;                                     ! Set no alignment specified
; 1225      1323  2
; 1226      1324  2 ch$move (.keyname [0]+1, keyname [0], pdd [pdd$b_namlng]) ;  ! Insert the name
; 1227      1325  2
; 1228      1326  2 retadr [0] = .pdd ;
; 1229      1327  2 return true
; 1230      1328  1 end ;


                                   007C 00000      .ENTRY  LNK$ALLOC_PDD, Save R2,R3,R4,R5,R6      ; 1303
                   5E           04   C2 00002      SUBL2   #4, SP                                  ;
                                5E   DD 00005      PUSHL   SP                                      ; 1317
                   7E     04    BC   9A 00007      MOVZBL  @KEYNAME, -(SP)                         ;
                   6E           10   C0 0000B      ADDL2   #16, (SP)                               ;
          00000000G 00          02   FB 0000E      CALLS   #2, LNK$ALLOBLK                         ;
                   56           6E   D0 00015      MOVL    PDD, R6                                 ; 1319
     10       00   6E           00   2C 00018      MOVC5   #0, (SP), #0, #16, (R6)                 ;
                                66      0001D                                                      ;
              0A   A6   0400    8F   B0 0001E      MOVW    #1024, 10(R6)                           ; 1321
              0E   A6           01   8E 00024      MNEGB   #1, 14(R6)                              ; 1322
                   50     04    BC   9A 00028      MOVZBL  @KEYNAME, R0                            ; 1324
                                50   D6 0002C      INCL    R0                                      ;
         0F   A6   04   BC      50   28 0002E      MOVC3   R0, @KEYNAME, 15(R6)                    ;
                   08   BC      56   D0 00034      MOVL    R6, @RETADR                             ; 1326
                        50      01   D0 00038      MOVL    #1, R0                                  ; 1327
                                     04 0003B      RET                                             ; 1328

; Routine Size:  60 bytes,    Routine Base:  $CODE$ + 09C3


; 1231      1329  1
```

```
1 global routine lnk$collectpsct : novalue =
1
1 !+
1 !
1 !  This routine is called at the end of command processing to process the data
1 !  base set up by the PSECT definition and collection options.  A pass is made
1 !  over the list of PSECTS hanging off each cluster mentioned in a collect
1 !  option.  For each PSECT in that list, define it.
1 !
1 !-
2 begin
2 local
2         curcluster      : ref block [,byte],                  ! Pointer to current cluster descriptor
2         lstcluster      : ref block [,byte],                  ! Pointer to last cluster descriptor
2         curpsct         : ref block [,byte],                  ! Pointer to current psect name block
2         cludesc         : ref block [,byte],                  ! Pointer to cluster descriptor
2         psctdesc        : ref block [,byte],                  ! Pointer to psect descriptor
2         psctdef         : ref block [,byte] ;                 ! Pointer to psect definition block
2
2 curcluster = .lnk$gl_cclulst ;                                ! Point to first cluster name block
2 !
2 !  loop through all named clusters
2 !
2 while .curcluster neq 0 do
3     begin
3     if not lib$lookup_tree (lnk$gl_clutree                    ! Lookup up cluster
3                         ,curcluster [ccd$b_namlng]
3                         ,lnk$clunamcmp
3                         ,cludesc
3                         )
3     then
4         begin                                                 ! And if not defined yet
4         lnk$alloccluster (cludesc) ;                          ! Then do it now
4         ch$move (.curcluster [ccd$b_namlng]+1                 ! Copy in the name
4                 , curcluster [ccd$b_namlng]
4                 , cludesc [clu$b_namlng]
4                 ) ;
4         lnk$insert_clu (.cludesc) ;                           ! Insert into cluster tree
4         if .curcluster [ccd$b_protect]                        ! If the cluster should be protected
4         then
4             cludesc [clu$v_protect] = true ;                  ! Then flag it now
4         lnk$gl_curclu = .cludesc ;                            ! Make it the current cluster
4         end
3     else
3         lnk$gl_curclu  = .cludesc [node$l_ptr] ;              ! Found--make it the current cluster
3
3     curpsct     = .curcluster [ccd$l_psclst] ;                ! Point to first psect block
3     lstcluster  = .curcluster ;                               ! Remember this one
3     curcluster  = .curcluster [ccd$l_nxtclu] ;                ! And point to next one
3     lnk$dealblk (ccd$c_size, .lstcluster) ;                   ! Deallocate it
3     !
3     !  Now get all the psects collected into that cluster
3     !
3     while .curpsct neq 0
4     do  begin
4         local   defbyopt ;
4
```

```
            if  not  (defbyopt = lnk$srcpscdef (curpsct [cpd$b_namlng], psctdef))
            then
                lib$insert_tree (lnk$gl_pscdflst                            ! Insert into defined by option list
                                ,curpsct [cpd$b_namlng]
                                ,%ref (0)
                                ,lnk$compare_pdd
                                ,lnk$alloc_pdd,psctdef
                                ) ;
            psctdef [pdd$w_flags] = .psctdef [pdd$w_flags] or gps$m_gbl ;   ! Mark psect as global
            lnk$fndpscnam   (curpsct [cpd$b_namlng]                         ! Enter psect in current cluster
                            ,.psctdef [pdd$w_flags]
                            ,0
                            ,psctdesc
                            ) ;
            if .defbyopt                                                    ! If defined by option
            then                                                            ! then
                begin
                psctdef [pdd$b_namlng] = .psctdef [pdd$b_namlng] or %x'80' ; !  flag psect already defined
                psctdesc [psc$w_flags] = .psctdef [pdd$w_flags] ;           !   copy the defined flags
                psctdesc [psc$b_align] = .psctdef [pdd$b_align] ;           !   and the alignment

                if  .psctdef [pdd$b_align] eql %x'FF'                       !  if defaulted
                then psctdesc [psc$b_align] = 0 ;                           !  then use byte alignmenc
                end ;

            curpsct = .curpsct [cpd$l_nxtpsc] ;                             ! Point to next name block
            end                                                             ! Curpsct neq 0
        end ;                                                               ! Curcluter neq 0

    return ;
    end ;



                                       0FFC 00000          .ENTRY  LNK$COLLECTPSCT, Save R2,R3,R4,R5,R6,R7,R8,-; 1330
                                                                   R9,R10,R11                               ;
                     5B 00000000G   00   9E 00002          MOVAB   LNK$GL_CURCLU, R11                       ;
                     5A     FF67    CF   9E 00009          MOVAB   LNK$CLONAMCMP, R10                       ;
                     5E             10   C2 0000E          SUBL2   #16, SP                                  ;
                     57 00000000G   00   D0 00011          MOVL    LNK$GL_CCLULST, CURCLUSTER               ; 1349
                                    57   D5 00018  1$:     TSTL    CURCLUSTER                               ; 1353
                                    01   12 0001A          BNEQ    2$                                       ;
                                         04 0001C          RET                                              ;
                            04      AE   9F 0001D  2$:     PUSHAB  CLUDESC                                  ; 1356
                                    5A   DD 00020          PUSHL   R10                                      ;
                            04      A7   9F 00022          PUSHAB  4(CURCLUSTER)                            ;
                      00000000G     00   9F 00025          PUSHAB  LNK$GL_CLUTREE                           ; 1355
               00000000G   00       04   FB 0002B          CALLS   #4, LIB$LOOKUP_TREE                      ; 1356
                           31       50   E8 00032          BLBS    R0, 4$                                   ;
                            04      AE   9F 00035          PUSHAB  CLUDESC                                  ; 1362
               00000000V   EF       01   FB 00038          CALLS   #1, LNK$ALLOCLUSTER                      ;
                           50  04   A7   9A 0003F          MOVZBL  4(CURCLUSTER), R0                        ; 1363
                                    50   D6 00043          INCL    R0                                       ;
                           56  04   AE   D0 00045          MOVL    CLUDESC, R6                              ; 1365
          5C   A6     04   A7       50   28 00049          MOVC3   R0, 4(CURCLUSTER), 92(R6)                ;
```

```
                           56 DD 0004F        PUSHL   R6                              ; 1367
  00000000V EF             01 FB 00051        CALLS   #1, LNK$INSERT_CLU              ;
            05       30    A7 E9 00058        BLBC    48(CURCLUSTER), 3$              ; 1368
         58 A6       80    8F 88 0005C        BISB2   #128, 88(R6)                    ; 1370
            6B             56 D0 00061 3$:    MOVL    R6, LNK$GL_CURCLU               ; 1371
                           08 11 00064        BRB     5$                              ; 1355
            50       04    AE D0 00066 4$:    MOVL    CLUDESC, R0                     ; 1374
            6B       0A    A0 D0 0006A        MOVL    10(R0), LNK$GL_CURCLU           ;
            58       2C    A7 D0 0006E 5$:    MOVL    44(CURCLUSTER), CURPSCT         ; 1376
            59             57 D0 00072        MOVL    CURCLUSTER, LSTCLUSTER          ; 1377
            57             67 D0 00075        MOVL    (CURCLUSTER), CURCLUSTER        ; 1378
                           59 DD 00078        PUSHL   LSTCLUSTER                      ; 1379
                           31 DD 0007A        PUSHL   #49                             ;
  00000000G 00             02 FB 0007C        CALLS   #2, LNK$DEALBLK                 ;
                           58 D5 00083 6$:    TSTL    CURPSCT                         ; 1383
                           91 13 00085        BEQL    1$                              ;
                     08    AE 9F 00087        PUSHAB  PSCTDEF                         ; 1387
                     04    A8 9F 0008A        PUSHAB  4(CURPSCT)                      ;
  00000000V EF             02 FB 0008D        CALLS   #2, LNK$SRCPSCDEF               ;
            53             50 D0 00094        MOVL    R0, DEFBYOPT                    ;
            1F             53 E8 00097        BLBS    DEFBYOPT, 7$                    ;
                     08    AE 9F 0009A        PUSHAB  PSCTDEF                         ; 1390
                     50    AA 9F 0009D        PUSHAB  LNK$ALLOC_PDD                   ;
                     29    AA 9F 000A0        PUSHAB  LNK$COMPARE_PDD                 ;
                     0C    AE D4 000A3        CLRL    12(SP)                          ; 1391
                     0C    AE 9F 000A6        PUSHAB  12(SP)                          ;
                     04    A8 9F 000A9        PUSHAB  4(CURPSCT)                      ; 1390
             00000000G     00 9F 000AC        PUSHAB  LNK$GL_PSCDFLST                 ; 1389
  00000000G 00             06 FB 000B2        CALLS   #6, LIB$INSERT_TREE             ; 1390
            52       08    AE D0 000B9 7$:    MOVL    PSCTDEF, R2                     ; 1395
         0A A2             10 88 000BD        BISB2   #16, 10(R2)                     ;
                     0C    AE 9F 000C1        PUSHAB  PSCTDESC                        ; 1396
                           7E D4 000C4        CLRL    -(SP)                           ;
            7E       0A    A2 3C 000C6        MOVZWL  10(R2), -(SP)                   ; 1397
                     04    A8 9F 000CA        PUSHAB  4(CURPSCT)                      ; 1396
  00000000G 00             04 FB 000CD        CALLS   #4, LNK$FNDPSCNAM               ;
            1D             53 E9 000D4        BLBC    DEFBYOPT, 8$                    ; 1401
         0F A2       80    8F 88 000D7        BISB2   #128, 15(R2)                    ; 1404
            50       0C    AE D0 000DC        MOVL    PSCTDESC, R0                    ; 1405
         0A A0       0A    A2 B0 000E0        MOVW    10(R2), 10(R0)                  ;
         2C A0       0E    A2 90 000E5        MOVB    14(R2), 44(R0)                  ; 1406
         FF 8F       0E    A2 91 000EA        CMPB    14(R2), #255                    ; 1408
                           03 12 000EF        BNEQ    8$                              ;
                     2C    A0 94 000F1        CLRB    44(R0)                          ; 1409
            58             68 D0 000F4 8$:    MOVL    (CURPSCT), CURPSCT              ; 1412
                           8A 11 000F7        BRB     6$                              ; 1383
                              04 000F9        RET                                     ; 1417

; Routine Size:  250 bytes,    Routine Base:  $CODE$ + 09FF


; 1321          1418  1
```

```
; 1323          1419  1 global routine lnk$srcpscdef (pscnam,psctdesc) =
; 1324          1420  1
; 1325          1421  1 !+
; 1326          1422  1 !
; 1327          1423  1 !   Look up PSECT name in the psects defined by option.
; 1328          1424  1 !
; 1329          1425  1 !-
; 1330          1426  2 begin
; 1331          1427  2 map
; 1332          1428  2         pscnam          : ref vector [,byte],                   ! Ascic name
; 1333          1429  2         psctdesc        : ref block [,byte] ;                   ! Pointer
; 1334          1430  2
; 1335          1431  2 return lib$lookup_tree (lnk$gl_pscdflst,.pscnam,lnk$compare_pdd, .psctdesc)
; 1336          1432  2
; 1337          1433  1 end ;


                                  0000 00000          .ENTRY  LNK$SRCPSCDEF, Save nothing                   ; 1419
                             08    AC  DD 00002          PUSHL   PSCTDESC                                      ; 1431
                           FE9A    CF  9F 00005          PUSHAB  LNK$COMPARE_PDD                               ;
                             04    AC  DD 00009          PUSHL   PSCNAM                                        ;
                      00000000G    00  9F 0000C          PUSHAB  LNK$GL_PSCDFLST                               ;
          00000000G  00            04  FB 00012          CALLS   #4, LIB$LOOKUP_TREE                           ;
                                       04 00019          RET                                                   ; 1433

; Routine Size:  26 bytes,    Routine Base:  $CODE$ + 0AF9


; 1338          1434  1
```

```
; 1340      1435 1 global routine lnk$alloccluster (clusterdesc,dummyarg) : novalue =
; 1341      1436 1
; 1342      1437 1 !+
; 1343      1438 1 !
; 1344      1439 1 !   Allocate a cluster descriptor and link on end of the list and initialize
; 1345      1440 1 !   some fields of it now.
; 1346      1441 1 !
; 1347      1442 1 !   If the second argument is present, the cluster descriptor is not linked
; 1348      1443 1 !   into the list.
; 1349      1444 1 !
; 1350      1445 1 !-
; 1351      1446 2 begin
; 1352      1447 2 map
; 1353      1448 2         clusterdesc     : ref vector [,long] ;
; 1354      1449 2 local
; 1355      1450 2         newcluster      : ref block [,byte] ;
; 1356      1451 2 builtin
; 1357      1452 2         nullparameter ;
; 1358      1453 2
; 1359      1454 2 lnk$alloblk (clu$c_size,clusterdesc [0]) ;                   ! Allocate the block
; 1360      1455 2 newcluster = .clusterdesc [0] ;                               ! Retrieve its address
; 1361      1456 2 ch$fill (0,clu$c_size,.newcluster) ;                         ! Zero the descriptor
; 1362      1457 2
; 1363      1458 2 newcluster [clu$l_lstfdb]  = newcluster [clu$l_fstfdb] ;     ! So empty list
; 1364      1459 2 newcluster [clu$l_lstisd]  = newcluster [clu$l_fstisd] ;     !
; 1365      1460 2 newcluster [clu$l_lstadrl] = newcluster [clu$l_fstadrl] ;
; 1366      1461 2 lnk$gw_nclusters           = .lnk$gw_nclusters + 1 ;         ! Count the cluster
; 1367      1462 2
; 1368      1463 2 if nullparameter (2)                                         ! If second argument not supplied
; 1369      1464 2 then
; 1370      1465 3     begin                                                    !  then link into end of cluster list
; 1371      1466 3     lnk$gl_lastclu [clu$l_nxtclu] = .newcluster ;            ! Set pointer
; 1372      1467 3     newcluster [clu$l_prevclu]    = .lnk$gl_lastclu ;
; 1373      1468 3     lnk$gl_lastclu = .newcluster ;                           ! For last in list
; 1374      1469 2     end ;
; 1375      1470 2 return ;
; 1376      1471 1 end ;


                                   00FC 00000       .ENTRY  LNK$ALLOCLUSTER, Save R2,R3,R4,R5,R6,R7      ; 1435
              57 00000000G     00    9E 00002       MOVAB   LNK$GL_LASTCLU, R7                           ;
                          04   AC    DD 00009       PUSHL   CLUSTERDESC                                  ; 1454
              7E          88   8F    9A 0000C       MOVZBL  #136, -(SP)                                  ;
        00000000G         00   02    FB 00010       CALLS   #2, LNK$ALLOBLK                              ;
              56          04   BC    D0 00017       MOVL    @CLUSTERDESC, NEWCLUSTER                     ; 1455
 0088   8F        00      6E   00    2C 0001B       MOVC5   #0, (SP), #0, #136, (NEWCLUSTER)             ; 1456
                               66       00022                                                        ;
              0C   A6     08   A6    9E 00023       MOVAB   8(R6), 12(NEWCLUSTER)                        ; 1458
              1C   A6     18   A6    9E 00028       MOVAB   24(R6), 28(NEWCLUSTER)                       ; 1459
              40   A6     3C   A6    9E 0002D       MOVAB   60(R6), 64(NEWCLUSTER)                       ; 1460
                  00000000G    00    B6 00032       INCW    LNK$GW_NCLUSTERS                             ; 1461
              02               6C    91 00038       CMPB    (AP), #2                                     ; 1463
                               05    1F 0003B       BLSSU   1$                                           ;
                          08   AC    D5 0003D       TSTL    8(AP)                                        ;
```

```
                              0D  12 00040       BNEQ    2$                                    ;
                  50          67  D0 00042 1$:   MOVL    LNK$GL_LASTCLU, R0                    ; 1466
                  60          56  D0 00045       MOVL    NEWCLUSTER, (R0)                      ;
          04      A6          50  D0 00048       MOVL    R0, 4(NEWCLUSTER)                     ; 1467
                  67          56  D0 0004C       MOVL    NEWCLUSTER, LNK$GL_LASTCLU            ; 1468
                                  04 0004F 2$:   RET                                           ; 1471

; Routine Size:  80 bytes,    Routine Base:  $CODE$ + 0B13


; 1377          1472  1
```

```
1 global routine lnk$insert_clu  (cludesc) =
1
1 !+
1 !
1 !     This routine is called to insert a cluster descriptor into the balanced
1 !     binary tree of cluster descriptors.
1 !
1 !     Inputs:
1 !
1 !             CLUDESC         address of cluster descriptor
1 !
1 !
1 !-
2 begin
2         routine clunodealloc  (keyname,retadr,cludesc) =
3         begin
3         !
3         ! routine to allocate node for new cluster
3         !
3         map
3                 keyname         : ref vector [,byte],
3                 retadr          : ref vector [,long],
3                 cludesc         : ref block [,byte] ;
3         local
3                 nodeptr         : ref block [,byte] ;
3
3         lnk$alloblk (node$c_long+.keyname [0],nodeptr) ;
3         nodeptr [node$l_ptr] = .cludesc ;
3         retadr [0]           = .nodeptr ;
3
3         return true
2         end ;



                                  0000 00000 CLUNODEALLOC:
                                                   .WORD   Save nothing                          ; 1487
              5E              04   C2 00002        SUBL2   #4, SP
                              5E   DD 00005        PUSHL   SP                                    ; 1499
              7E         04   BC   9A 00007        MOVZBL  @KEYNAME, -(SP)
              6E              0E   C0 0000B        ADDL2   #14, (SP)
    00000000G 00              02   FB 0000E        CALLS   #2, LNK$ALLOBLK
              50              6E   D0 00015        MOVL    NODEPTR, R0                           ; 1500
        0A    A0         0C   AC   D0 00018        MOVL    CLUDESC, 10(R0)
        08    BC              50   D0 0001D        MOVL    R0, @RETADR                           ; 1501
              50              01   D0 00021        MOVL    #1, R0                                ; 1503
                                   04 00024        RET                                           ; 1504

; Routine Size:  37 bytes,    Routine Base:  $CODE$ + 0B63

2 !
2 ! Main body of lnk$insert_clu
2 !
2 map
```

```
; 1415        1509  2             cludesc          : ref block [,byte] ;
; 1416        1510  2 local
; 1417        1511  2             blockaddr ;
; 1418        1512  2
; 1419        1513  2 return lib$insert_tree (lnk$gl_clutree, cludesc [clu$b_namlng], %ref (0)
; 1420        1514  2                   ,lnk$clunamcmp, clunodealloc, blockaddr, .cludesc
; 1421        1515  2                   )
; 1422        1516  1 end ;



                                        0000 00000         .ENTRY  LNK$INSERT_CLU, Save nothing              ; 1473
                    5E            08  C2 00002         SUBL2   #8, SP                                    ;
                            04    AC  DD 00005         PUSHL   CLUDESC                                   ; 1514
                            08    AE  9F 00008         PUSHAB  BLOCKADDR                                 ; 1513
                            CD    AF  9F 0000B         PUSHAB  CLUNODEALLOC                              ;
                          FDD9    CF  9F 0000E         PUSHAB  LNK$CLUNAMCMP                             ;
                            10    AE  D4 00012         CLRL    16(SP)                                    ;
                            10    AE  9F 00015         PUSHAB  16(SP)                                    ;
          7E        04    AC 0000005C  8F  C1 00018    ADDL3   #92, CLUDESC, -(SP)                       ;
                        00000000G  00  9F 00021        PUSHAB  LNK$GL_CLUTREE                            ;
             00000000G  00        07  FB 00027         CALLS   #7, LIB$INSERT_TREE                       ;
                                  04 0002E             RET                                               ; 1516

; Routine Size:  47 bytes,    Routine Base:  $CODE$ + 0B88
```

```
1 routine get_image : novalue =
1
1 !++
1 !
1 !     This routine processes the /EXE switch and determines the name and
1 !     the associated related name of the .EXE file.
1 !
1 !--
2 begin
2 local
2               exe_desc        : dynamic_descriptor,                          ! Descriptor of .EXE file
2               new_file        : initial (0),                                 ! Set if new .EXE request is made
2               status ;
2 own
2               exe_name        : vector [nam$c_maxrss, byte],                 ! Related file for .EXE
2               exe_inrel       : block [nam$c_bln, byte] ;                    ! Related NAM$ for .EXE
2
3 if  (status = cli$present (sd_executable))                                  ! /EXECUTABLE
3 then begin
3       if  .local_qualifier
4       then begin
4            if  .status neq cli$_locpres
4            then return
5            else begin
5                 setup_inrelnam (0,.last_fnm_desc, exe_inrel, exe_name) ; ! Setup related name
5                 lnk$gl_relnam_exe = exe_inrel ;                          ! using last input file
5                 end
4            end
3       else lnk$gl_relnam_exe = dflt_inrel ;                                 ! Use default related name
3
3       new_file            = true ;                                          ! Setup new .EXE file spec
3       univstar_valid      = false ;                                         ! Disallow UNIV=* for STB generation
4       lnk$gb_locnov_exe = (if cli$get_value (sd_executable, exe_desc)       ! Get image name (no name is okay)
4                               then  false                                   ! and setup 'local no-value' flag
4                               else .local_qualifier
3                            ) ;
3       if  .lnk$gl_ctlmsk [lnk$v_shr]                                        ! If a shareable image
3       then if  .exe_desc [dsc$w_length] eql 0                               ! and /EXE didn't specify a name
3            then copy_desc (share_desc, exe_desc) ;                          ! then use /SHAREABLE name
3       end
3 else begin
4       if ((not .local_qualifier) and (.status eql cli$_negated))            ! But, if explicitly negated,
4       or ((    .local_qualifier) and (.status eql cli$_locneg))             !
4       then begin                                                            !  disable image production
4            new_file       = true ;                                          ! Setup new .EXE file spec
4            univstar_valid = true ;                                          ! Will now allow UNIV=* for STB gene
4            copy_desc (null_device, exe_desc) ;
3            end ;
2       end ;
2
2 if .new_file
3 then begin
3       outfile = lnk$gl_imgfil ;                                             ! Point to the pointer
3       outputfile (exe_desc) ;                                               ! Set up image output file
2       end ;
1 end ;
```

```
                                                    .PSECT  $OWN$,NOEXE,2

                                    001E0 EXE_NAME:
                                                    .BLKB   255
                                    002DF           .BLKB   1
                                    002E0 EXE_INREL:
                                                    .BLKB   96


                                                    .PSECT  $CODE$,NOWRT,2

                               003C 00000 GET_IMAGE:
                                                    .WORD   Save R2,R3,R4,R5                  ; 1517
           55 00000000'   EF   9E 00002             MOVAB   SD_EXECUTABLE, R5                 ;
           54 00000000'   EF   9E 00009             MOVAB   LNK$GL_RELNAM_EXE, R4             ;
           53 00000000'   EF   9E 00010             MOVAB   LOCAL_QUALIFIER, R3               ;
           5E             04   C2 00017             SUBL2   #4, SP                            ;
              02000000    8F   DD 0001A             PUSHL   #33554432                         ; 1527
                    04    AE   D4 00020             CLRL    EXE_DESC+4                        ;
                          52   D4 00023             CLRL    NEW_FILE                          ;
                          55   DD 00025             PUSHL   R5                                ; 1534
        00000000G  00     01   FB 00027             CALLS   #1, CLI$PRESENT                   ;
                   60     50   E9 0002E             BLBC    STATUS, 6$                        ;
                   26     63   E9 00031             BLBC    LOCAL_QUALIFIER, 2$               ; 1536
        00000000G  8F     50   D1 00034             CMPL    STATUS, #CLI$_LOCPRES             ; 1538
                          01   13 0003B             BEQL    1$                                ;
                               04 0003D             RET                                       ;
                  01DC    C3   9F 0003E 1$:         PUSHAB  EXE_NAME                          ; 1541
                  02DC    C3   9F 00042             PUSHAB  EXE_INREL                         ;
                  0168    C3   DD 00046             PUSHL   LAST_FNM_DESC                     ;
                          7E   D4 0004A             CLRL    -(SP)                             ;
        00000000V  EF     04   FB 0004C             CALLS   #4, SETUP_INRELNAM                ;
                   64   02DC  C3   9E 00053         MOVAB   EXE_INREL, LNK$GL_RELNAM_EXE      ; 1542
                          05   11 00058             BRB     3$                                ; 1537
                   64   0108  C3   9E 0005A 2$:     MOVAB   DFLT_INREL, LNK$GL_RELNAM_EXE     ; 1545
                   52     01   D0 0005F 3$:         MOVL    #1, NEW_FILE                      ; 1547
                    FC    A3   D4 00062             CLRL    UNIVSTAR_VALID                    ; 1548
                  4020    8F   BB 00065             PUSHR   #^M<R5,SP>                        ; 1549
        00000000G  00     02   FB 00069             CALLS   #2, CLI$GET_VALUE                 ;
                   04     50   E9 00070             BLBC    R0, 4$                            ;
                          50   D4 00073             CLRL    R0                                ;
                          03   11 00075             BRB     5$                                ;
                   50     63   D0 00077 4$:         MOVL    LOCAL_QUALIFIER, R0               ; 1551
              0C   A4     50   90 0007A 5$:         MOVB    R0, LNK$GB_LOCNOV_EXE             ; 1549
     2F 00000000G  00     02   E1 0007E             BBC     #2, LNK$GL_CTLMSK, 9$             ; 1553
                          6E   B5 00086             TSTW    EXE_DESC                          ; 1554
                          2B   12 00088             BNEQ    9$                                ;
                   6E   0184  C3   7D 0008A         MOVQ    SHARE_DESC, EXE_DESC              ; 1555
                          24   11 0008F             BRB     9$                                ; 1534
                   0C     63   E8 00091 6$:         BLBS    LOCAL_QUALIFIER, 7$               ; 1558
        00000000G  8F     50   D1 00094             CMPL    STATUS, #CLI$_NEGATED             ;
                          0C   13 0009B             BEQL    8$                                ;
                   15     63   E9 0009D             BLBC    LOCAL_QUALIFIER, 9$               ; 1559
        00000000G  8F     50   D1 000A0 7$:         CMPL    STATUS, #CLI$_LOCNEG              ;
```

```
                                 0C  12 000A7        BNEQ    9$                              ;
              52                 01  D0 000A9 8$:    MOVL    #1, NEW_FILE                    ; 1561
        FC    A3                 01  D0 000AC        MOVL    #1, UNIVSTAR_VALID              ; 1562
              6E      0124       C5  7D 000B0        MOVQ    NULL_DEVICE, EXE_DESC           ; 1563
              12                 52  E9 000B5 9$:    BLBC    NEW_FILE, 10$                   ; 1567
      019C    C3 00000000G       00  9E 000B8        MOVAB   LNK$GL_IMGFIL, OUTFILE          ; 1569
                                 5E  DD 000C1        PUSHL   SP                              ; 1570
 00000000V    EF                 01  FB 000C3        CALLS   #1, OUTPUTFILE                  ;
                                     04 000CA 10$:   RET                                     ; 1572

; Routine Size:  203 bytes,    Routine Base:  $CODE$ + 0BB7
```

```
1 routine get_sym : novalue =
1
1 !++
1 !
1 ! This routine retrieves the symbol table file name and qualifiers
1 !
1 !--
2 begin
2 local
2       sym_desc        : dynamic_descriptor,                             ! Descriptor of .SYM file
2       status ;
2 own
2       sym_name        : vector [nam$c_maxrss, byte],                    ! Related file for .SYM
2       sym_inrel       : block [nam$c_bln, byte] ;                       ! Related NAM$ for .SYM
2
3 if  (status = cli$present (sd_symbol_table))                           ! /SYMBOL_TABLE
3 then begin
3      if  .local_qualifier
4      then begin
4           if  .status neq cli$_locpres
4           then return
5           else begin
5                setup_inrelnam (0,.last_fnm_desc, sym_inrel, sym_name) ; ! Setup related name
5                lnk$gl_relnam_sym = sym_inrel ;                         ! using last input file
5                end
4           end
3      else lnk$gl_relnam_sym  = dflt_inrel ;                            ! Use default related name
3
3      lnk$gl_ctlmsk [lnk$v_symtbl] = true ;                             ! Ask for symbol table
4      lnk$gb_locnov_sym = (if cli$get_value (sd_symbol_table, sym_desc) ! Get .STB name (no name is okay)
4                          then  false                                   ! and setup 'local value' flag
4                          else .local_qualifier
3                          ) ;
3      outfile   = lnk$gl_symfil ;                                       ! Point to the pointer
3      outputfile (sym_desc) ;                                           ! And set up output file
3      end
3 else begin
4      if ((not .local_qualifier) and (.status eql cli$_negated))        ! But, if explicitly negated,
4      or ((    .local_qualifier) and (.status eql cli$_locneg))         !
3      then lnk$gl_ctlmsk [lnk$v_symtbl] = false ;                       !  disable production
2      end ;
1 end ;
```

```
                        .PSECT  $OWN$,NOEXE,2

00340 SYM_NAME:
                        .BLKB   255
0043F                   .BLKB   1
00440 SYM_INREL:
                        .BLKB   96


                        .PSECT  $CODE$,NOWRT,2
```

```
                                  001C 00000 GET_SYM:.WORD    Save R2,R3,R4                              ; 1573
               54 00000000'   EF    9E 00002         MOVAB    SD_SYMBOL_TABLE, R4
               53 00000000'   EF    9E 00009         MOVAB    LNK$GL_RELNAM_SYM, R3
               52 00000000'   EF    9E 00010         MOVAB    LOCAL_QUALIFIER, R2
               5E             04    C2 00017         SUBL2    #4, SP
                  02000000    8F    DD 0001A         PUSHL    #33554432                                  ; 1582
                        04    AE    D4 00020         CLRL     SYM_DESC+4
                              54    DD 00023         PUSHL    R4                                         ; 1588
     00000000G 00             01    FB 00025         CALLS    #1, CLI$PRESENT
               60             50    E9 0002C         BLBC     STATUS, 5$
               25             62    E9 0002F         BLBC     LOCAL_QUALIFIER, 1$                        ; 1590
     00000000G 8F             50    D1 00032         CMPL     STATUS, #CLI$_LOCPRES                      ; 1592
                              73    12 00039         BNEQ     8$
                      033C    C2    9F 0003B         PUSHAB   SYM_NAME                                   ; 1595
                      043C    C2    9F 0003F         PUSHAB   SYM_INREL
                      0168    C2    DD 00043         PUSHL    LAST_FNM_DESC
                              7E    D4 00047         CLRL     -(SP)
     00000000V EF             04    FB 00049         CALLS    #4, SETUP_INRELNAM
               63     043C    C2    9E 00050         MOVAB    SYM_INREL, LNK$GL_RELNAM_SYM               ; 1596
                              05    11 00055         BRB      2$                                         ; 1591
               63     0108    C2    9E 00057 1$:     MOVAB    DFLT_INREL, LNK$GL_RELNAM_SYM              ; 1599
     00000000G 00             20    88 0005C 2$:     BISB2    #32, LNK$GL_CTLMSK+1                       ; 1601
                      4010    8F    BB 00063         PUSHR    #^M<R4,SP>                                 ; 1602
     00000000G 00             02    FB 00067         CALLS    #2, CLI$GET_VALUE
               04             50    E9 0006E         BLBC     R0, 3$
                              50    D4 00071         CLRL     R0
                              03    11 00073         BRB      4$
               50             62    D0 00075 3$:     MOVL     LOCAL_QUALIFIER, R0                        ; 1604
          06   A3             50    90 00078 4$:     MOVB     R0, LNK$GB_LOCNOV_SYM                      ; 1602
        019C   C2 00000000G   00    9E 0007C         MOVAB    LNK$GL_SYMFIL, OUTFILE                     ; 1606
                              5E    DD 00085         PUSHL    SP                                         ; 1607
     00000000V EF             01    FB 00087         CALLS    #1, OUTPUTFILE
                                    04 0008E         RET                                                 ; 1588
               0C             62    E8 0008F 5$:     BLBS     LOCAL_QUALIFIER, 6$                        ; 1610
     00000000G 8F             50    D1 00092         CMPL     STATUS, #CLI$_NEGATED
                              0C    13 00099         BEQL     7$
               10             62    E9 0009B         BLBC     LOCAL_QUALIFIER, 8$                        ; 1611
     00000000G 8F             50    D1 0009E 6$:     CMPL     STATUS, #CLI$_LOCNEG
                              07    12 000A5         BNEQ     8$
     00000000G 00             20    8A 000A7 7$:     BICB2    #32, LNK$GL_CTLMSK+1                       ; 1612
                                    04 000AE 8$:     RET                                                 ; 1614

; Routine Size:  175 bytes,    Routine Base:  $CODE$ + 0C82


; 1523         1615  1
```

```
routine get_map : novalue =

!++
!
!  This routine retrieves the map file name and qualifiers
!
!--
begin
local
        map_desc        : dynamic_descriptor,                               ! Descriptor of .MAP file
        status ;
own
        map_name        : vector [nam$c_maxrss, byte],                      ! Related file for .MAP
        map_inrel       : block [nam$c_bln, byte] ;                         ! Related NAM$ for .MAP

if  (status = cli$present (sd_map))                                         ! /MAP
then begin
     if  .local_qualifier
     then begin
          if  .status neq cli$_locpres
          then return
          else begin
               setup_inrelnam (0,.last_fnm_desc, map_inrel, map_name) ;    ! Setup related name
               lnk$gl_relnam_map = map_inrel ;                              ! using last input file
               end
          end
     else lnk$gl_relnam_map = dflt_inrel ;                                  ! Use default related name

     lnk$gl_ctlmsk [lnk$v_map] = true ;                                     ! Ask for map
     lnk$gb_locnov_map = (if cli$get_value (sd_map, map_desc)               ! Get map name (no name is okay)
                          then  false                                       ! and setup 'local value' flag
                          else .local_qualifier
                         ) ;
     outfile    = lnk$gl_mapfil ;                                           ! Point to the pointer
     outputfile (map_desc) ;                                                ! And set up output file
     end
else begin
     if ((not .local_qualifier) and (.status eql cli$_negated))             ! But, if explicitly negated,
     or ((    .local_qualifier) and (.status eql cli$_locneg))              !
     then lnk$gl_ctlmsk [lnk$v_map] = false ;                               !  disable production
     end ;
end ;
```

```
                                        .PSECT  $OWN$,NOEXE,2

                        004A0 MAP_NAME:
                                        .BLKB   255
                        0059F           .BLKB   1
                        005A0 MAP_INREL:
                                        .BLKB   96


                                        .PSECT  $CODE$,NOWRT,2
```

```
                                  003C 00000 GET_MAP:.WORD   Save R2,R3,R4,R5                          ; 1616
          55 00000000'   EF   9E 00002          MOVAB   SD_MAP, R5                                ;
          54 00000000G   00   9E 00009          MOVAB   LNK$GL_CTLMSK, R4                         ;
          53 00000000'   EF   9E 00010          MOVAB   LNK$GL_RELNAM_MAP, R3                     ;
          52 00000000'   EF   9E 00017          MOVAB   LOCAL_QUALIFIER, R2                       ;
          5E             04   C2 0001E          SUBL2   #4, SP                                    ;
             02000000    8F   DD 00021          PUSHL   #33554432                                 ; 1625
                   04    AE   D4 00027          CLRL    MAP_DESC+4                                ;
                         55   DD 0002A          PUSHL   R5                                        ; 1631
 00000000G  00           01   FB 0002C          CALLS   #1, CLI$PRESENT                           ;
            5C           50   E9 00033          BLBC    STATUS, 5$                                ;
            25           62   E9 00036          BLBC    LOCAL_QUALIFIER, 1$                       ; 1633
 00000000G  8F           50   D1 00039          CMPL    STATUS, #CLI$_LOCPRES                     ; 1635
                         6B   12 00040          BNEQ    8$                                        ;
                 049C    C2   9F 00042          PUSHAB  MAP_NAME                                  ; 1638
                 059C    C2   9F 00046          PUSHAB  MAP_INREL                                 ;
                 0168    C2   DD 0004A          PUSHL   LAST_FNM_DESC                             ;
                         7E   D4 0004E          CLRL    -(SP)                                     ;
 00000000V  EF           04   FB 00050          CALLS   #4, SETUP_INRELNAM                        ;
            63   059C    C2   9E 00057          MOVAB   MAP_INREL, LNK$GL_RELNAM_MAP              ; 1639
                         05   11 0005C          BRB     2$                                        ; 1634
            63   0108    C2   9E 0005E 1$:      MOVAB   DFLT_INREL, LNK$GL_RELNAM_MAP             ; 1642
            64           10   88 00063 2$:      BISB2   #16, LNK$GL_CTLMSK                        ; 1644
                 4020    8F   BB 00066          PUSHR   #^M<R5,SP>                                ; 1645
 00000000G  00           02   FB 0006A          CALLS   #2, CLI$GET_VALUE                         ;
            04           50   E9 00071          BLBC    R0, 3$                                    ;
                         50   D4 00074          CLRL    R0                                        ;
                         03   11 00076          BRB     4$                                        ;
            50           62   D0 00078 3$:      MOVL    LOCAL_QUALIFIER, R0                       ; 1647
       09   A3           50   90 0007B 4$:      MOVB    R0, LNK$GB_LOCNOV_MAP                     ; 1645
     019C   C2 00000000G 00   9E 0007F          MOVAB   LNK$GL_MAPFIL, OUTFILE                    ; 1649
                         5E   DD 00088          PUSHL   SP                                        ; 1650
 00000000V  EF           01   FB 0008A          CALLS   #1, OUTPUTFILE                            ;
                              04 00091          RET                                               ; 1631
            0C           62   E8 00092 5$:      BLBS    LOCAL_QUALIFIER, 6$                       ; 1653
 00000000G  8F           50   D1 00095          CMPL    STATUS, #CLI$_NEGATED                     ;
                         0C   13 0009C          BEQL    7$                                        ;
            0C           62   E9 0009E          BLBC    LOCAL_QUALIFIER, 8$                       ; 1654
 00000000G  8F           50   D1 000A1 6$:      CMPL    STATUS, #CLI$_LOCNEG                      ;
                         03   12 000A8          BNEQ    8$                                        ;
            64           10   8A 000AA 7$:      BICB2   #16, LNK$GL_CTLMSK                        ; 1655
                              04 000AD 8$:      RET                                               ; 1657

; Routine Size:  174 bytes,    Routine Base:  $CODE$ + 0D31


; 1567            1658  1
```

```
; 1569      1659 1 routine get_input  =
; 1570      1660 1
; 1571      1661 1 !++
; 1572      1662 1 !
; 1573      1663 1 !  This routine retrieves the input files
; 1574      1664 1 !
; 1575      1665 1 !--
; 1576      1666 2 begin
; 1577      1667 2 local    status ;
; 1578      1668 2
; 1579      1669 2 if status = cli$get_value (sd_p1, token_desc)                ! Get input file
; 1580      1670 2 then
; 1581      1671 3     begin
; 1582      1672 3     if   cli$present (sd_library)                              ! /LIBRARY
; 1583      1673 4     then begin
; 1584      1674 4          infilflgs = .infilflgs or  (fdb$m_libr or fdb$m_libsrch) ;
; 1585      1675 4          lnk$setlibrin () ;
; 1586      1676 3          end ;
; 1587      1677 3
; 1588      1678 3     if   cli$present (sd_selective_search)                     ! /SELECTIVE_SEARCH
; 1589      1679 3     then infilflgs = .infilflgs or fdb$m_selser ;
; 1590      1680 3
; 1591      1681 3     if  (.infilflgs and fdb$m_libextr) eql 0
; 1592      1682 3     then if   cli$present (sd_include)                         ! /INCLUDE LIST
; 1593      1683 4          then begin
; 1594      1684 4               infilflgs = .infilflgs  or  (fdb$m_libextr or fdb$m_libr) ;
; 1595      1685 4               lnk$setlibrin () ;
; 1596      1686 3               end ;
; 1597      1687 3
; 1598      1688 3     if   cli$present (sd_options)                              ! /OPTIONS FILE
; 1599      1689 4     then begin
; 1600      1690 4          infilflgs = .infilflgs or fdb$m_option ;
; 1601      1691 4          optionfile () ;
; 1602      1692 3          end ;
; 1603      1693 3
; 1604      1694 3     inputfile (token_desc) ;                                   ! Set up input file FDB
; 1605      1695 2     end ;
; 1606      1696 2
; 1607      1697 2 return .status ;
; 1608      1698 1 end ;
```

```
                              007C 00000 GET_INPUT:
                                         .WORD   Save R2,R3,R4,R5,R6              ; 1659
          56      F61C   CF   9E 00002   MOVAB   LNK$SETLIBRIN, R6                ;
          55 00000000G   00   9E 00007   MOVAB   CLI$PRESENT, R5                  ;
          54 00000000'   EF   9E 0000E   MOVAB   SD_P1, R4                        ;
          53 00000000'   EF   9E 00015   MOVAB   INFILFLGS, R3                    ;
                    E4   A3   9F 0001C   PUSHAB  TOKEN_DESC                       ; 1669
                         54   DD 0001F   PUSHL   R4                               ;
   00000000G  00         02   FB 00021   CALLS   #2, CLI$GET_VALUE                ;
          52             50   D0 00028   MOVL    R0, STATUS                       ;
          48             52   E9 0002B   BLBC    STATUS, 5$                       ;
                    CC   A4   9F 0002E   PUSHAB  SD_LIBRARY                       ; 1672
```

```
                 65          01  FB 00031        CALLS   #1, CLI$PRESENT                 ;
                 07          50  E9 00034        BLBC    R0, 1$                          ;
                 63     82   8F  88 00037        BISB2   #130, INFILFLGS                 ; 1674
                 66          00  FB 0003B        CALLS   #0, LNK$SETLIBRIN               ; 1675
                        F4   A4  9F 0003E 1$:    PUSHAB  SD_SELECTIVE_SEARCH             ; 1678
                 65          01  FB 00041        CALLS   #1, CLI$PRESENT                 ;
                 03          50  E9 00044        BLBC    R0, 2$                          ;
                 63          08  88 00047        BISB2   #8, INFILFLGS                   ; 1679
     10          63          06  E0 0004A 2$:    BBS     #6, INFILFLGS, 3$               ; 1681
                        BC   A4  9F 0004E        PUSHAB  SD_INCLUDE                      ; 1682
                 65          01  FB 00051        CALLS   #1, CLI$PRESENT                 ;
                 07          50  E9 00054        BLBC    R0, 3$                          ;
                 63     42   8F  88 00057        BISB2   #66, INFILFLGS                  ; 1684
                 66          00  FB 0005B        CALLS   #0, LNK$SETLIBRIN               ; 1685
                        DC   A4  9F 0005E 3$:    PUSHAB  SD_OPTIONS                      ; 1688
                 65          01  FB 00061        CALLS   #1, CLI$PRESENT                 ;
                 07          50  E9 00064        BLBC    R0, 4$                          ;
                 63          10  88 00067        BISB2   #16, INFILFLGS                  ; 1690
            63   A6          00  FB 0006A        CALLS   #0, OPTIONFILE                  ; 1691
                        E4   A3  9F 0006E 4$:    PUSHAB  TOKEN_DESC                      ; 1694
          0140   C6          01  FB 00071        CALLS   #1, INPUTFILE                   ;
                 50          52  D0 00076 5$:    MOVL    STATUS, R0                      ; 1697
                                 04 00079        RET                                     ; 1698

; Routine Size:  122 bytes,    Routine Base:  $CODE$ + 0DDF


; 1609          1699  1
```

```
; 1611      1700  1 routine outputfile
; 1612      1701  1                 (desc : ref block [,byte])       : novalue =
; 1613      1702  1 !+
; 1614      1703  1 !
; 1615      1704  1 !   This routine is called by cli if an output file is to be created.
; 1616      1705  1 !
; 1617      1706  1 !   The variable "OUTFILE" points to the cell which is to receive the
; 1618      1707  1 !   address of the file descriptor block for the particular output file.
; 1619      1708  1 !
; 1620      1709  1 !-
; 1621      1710  2 begin
; 1622      1711  2 bind    fdb = .outfile  : ref block [,byte] ;                      ! Point to the pointer
; 1623      1712  2
; 1624      1713  2 if .fdb eql 0                                                      ! If FDB not yet allocated,
; 1625      1714  2 then
; 1626      1715  3     begin
; 1627      1716  3     lnk$allofdb (fdb) ;                                            ! Allocate one
; 1628      1717  3     lnk$alloblk (nam$c_maxrss, fdb [fdb$l_usrnamadr]) ;            ! Allocate space for filename
; 1629      1718  3     end ;
; 1630      1719  2
; 1631      1720  2 if (fdb [fdb$w_usrnamlen] = .desc [dsc$w_length]) neq 0          ! Update file descriptor
; 1632      1721  2 then
; 1633      1722  2     ch$move (.desc [dsc$w_length], .desc [dsc$a_pointer], .fdb [fdb$l_usrnamadr]) ;  ! Copy name
; 1634      1723  2
; 1635      1724  2 return ;                                                           !   and we are all done
; 1636      1725  1 end ;


                                         003C 00000 OUTPUTFILE:
                                                           .WORD   Save R2,R3,R4,R5                            ; 1700
                    52 00000000'    EF      D0 00002       MOVL    OUTFILE, R2                                 ; 1711
                                    62      D5 00009       TSTL    (R2)                                        ; 1713
                                    16      12 0000B       BNEQ    1$                                          ;
                                    52      DD 0000D       PUSHL   R2                                          ; 1716
                       FA70    CF   01      FB 0000F       CALLS   #1, LNK$ALLOFDB                             ;
               7E               62  10      C1 00014       ADDL3   #16, (R2), -(SP)                            ; 1717
                                7E      FF  8F  9A 00018   MOVZBL  #255, -(SP)                                 ;
                  00000000G     00  02      FB 0001C       CALLS   #2, LNK$ALLOBLK                             ;
                                50  62      D0 00023 1$:   MOVL    (R2), R0                                    ; 1720
                                51      04  AC  D0 00026   MOVL    DESC, R1                                    ;
                          0C    A0  61      B0 0002A       MOVW    (R1), 12(R0)                                ;
                                    06      13 0002E       BEQL    2$                                          ;
          10    B0        04    B1  61      28 00030       MOVC3   (R1), @4(R1), @16(R0)                       ; 1722
                                            04 00036 2$:   RET                                                 ; 1725

; Routine Size:  55 bytes,    Routine Base:  $CODE$ + 0E59
```

```
; 1638           1726  1 routine setup_inrelnam
; 1639           1727  1                 (ofp_flag
; 1640           1728  1                 ,fnm_desc   : ref block [,byte]
; 1641           1729  1                 ,nam        : ref block [,byte]
; 1642           1730  1                 ,file_name : ref vector [,byte]
; 1643           1731  1                 ) : =
; 1644           1732  1
; 1645           1733  2 begin
; 1646           1734  2 local
; 1647           1735  2         status,
; 1648           1736  2         fab                : block [fab$c_bln, byte] ;     ! Fab $PARSE
; 1649           1737  2
; 1650         P 1738  2 $fab_init      (fab =  fab                                  ! Intialize the fab
; 1651         P 1739  2 !              ,dna = .typeadr
; 1652         P 1740  2 !              ,dns =  typelen
; 1653         P 1741  2                ,fns = .fnm_desc [dsc$w_length]
; 1654         P 1742  2                ,fna = .fnm_desc [dsc$a_pointer]
; 1655         P 1743  2                ,nam = .nam
; 1656           1744  2                ) ;
; 1657           1745  2 fab [fab$v_ofp] = .ofp_flag ;                              ! Decide on 'output file parse' setting
; 1658           1746  2
; 1659         P 1747  2 $nam_init      (nam = .nam                                  ! Initialize the nam block
; 1660         P 1748  2                ,rsa = .file_name
; 1661         P 1749  2                ,rss =  nam$c_maxrss
; 1662         P 1750  2                ,esa = .file_name
; 1663         P 1751  2                ,ess =  nam$c_maxrss
; 1664           1752  2                ) ;
; 1665           1753  2
; 1666           1754  3 if (status = $parse (fab=fab))
; 1667           1755  2 then
; 1668           1756  2     nam [nam$b_rsl] = .nam [nam$b_esl] ;
; 1669           1757  2
; 1670           1758  2 return .status ;
; 1671           1759  1 end ;


                                                                   .EXTRN  SYS$PARSE

                                              007C 00000 SETUP_INRELNAM:
                                                                   .WORD   Save R2,R3,R4,R5,R6                  ; 1726
                                   5E      B0   AE  9E 00002      MOVAB   -80(SP), SP                          ;
  0050   8F           00           6E           00  2C 00006      MOVC5   #0, (SP), #0, #80, $RMS_PTR          ; 1744
                                                6E     0000D                                                   ;
                                   6E    5003   8F  B0 0000E      MOVW    #20483, $RMS_PTR                     ;
                          16       AE           02  90 00013      MOVB    #2, $RMS_PTR+22                      ;
                          1F       AE           02  90 00017      MOVB    #2, $RMS_PTR+31                      ;
                                   56      0C   AC  D0 0001B      MOVL    NAM, R6                              ;
                          28       AE           56  D0 0001F      MOVL    R6, $RMS_PTR+40                      ;
                                   50      08   AC  D0 00023      MOVL    FNM_DESC, R0                         ;
                          2C       AE      04   A0  D0 00027      MOVL    4(R0), $RMS_PTR+44                   ;
                          34       AE           60  90 0002C      MOVB    (R0), $RMS_PTR+52                    ;
    07   AE           01           05      04   AC  F0 00030      INSV    OFP_FLAG, #5, #1, FAB+7              ; 1745
  0060   8F           00           6E           00  2C 00037      MOVC5   #0, (SP), #0, #96, (R6)              ; 1752
                                                66     0003E                                                   ;
                                   66    6002   8F  B0 0003F      MOVW    #24578, (R6)                         ;
                          02       A6           01  8E 00044      MNEGB   #1, 2(R6)                            ;
```

```
                04   A6        10   AC  D0 00048          MOVL    FILE_NAME, 4(R6)                    ;
                0A   A6             01  8E 0004D          MNEGB   #1, 10(R6)                          ;
                0C   A6        10   AC  D0 00051          MOVL    FILE_NAME, 12(R6)                   ;
                                    5E  DD 00056          PUSHL   SP                                  ; 1754
        00000000G    00             01  FB 00058          CALLS   #1, SYS$PARSE                       ;
                     05             50  E9 0005F          BLBC    STATUS, 1$                          ;
                03   A6        0B   A6  90 00062          MOVB    11(R6), 3(R6)                       ; 1756
                                        04 00067 1$:      RET                                         ; 1759

; Routine Size:  104 bytes,    Routine Base:  $CODE$ + 0E90


; 1672          1760  1
; 1673          1761  1 end
; 1674          1762  0 eludom



                                                          .EXTRN  LIB$SIGNAL, LIB$STOP

;                              PSECT SUMMARY
;
;       Name                            Bytes                      Attributes
;
;  $PLIT$                                 452  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $GLOBAL$                                36  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $OWN$                                 1536  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  _LIB$KEY0$                              10  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(1)
;  _LIB$STATE$                             60  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(1)
;  _LIB$KEY1$                              31  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(1)
;  $CODE$                                3832  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)




;                         Library Statistics
;
;                                          -------- Symbols --------      Pages       Processing
;       File                               Total    Loaded   Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]STARLET.L32;1        9776       123         1         581      00:01.1
;  _$255$DUA28:[LINKER.OBJ]DATBAS.L32;1      538        76        14          28      00:00.4
;  _$255$DUA28:[SYSLIB]CLIMAC.L32;1           14         0         0           9      00:00.1
;  _$255$DUA28:[SYSLIB]TPAMAC.L32;1           42        21        50          14      00:00.1




;                               COMMAND QUALIFIERS

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

; 1675          1763  0
```

```
; Size:          3832 code + 2125 data bytes
; Run Time:         01:12.1
; Elapsed Time:     02:27.4
; Lines/CPU Min:      1466
; Lexemes/CPU-Min: 25965
; Memory Used:  386 pages
; Compilation Complete
```

0216 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

LNKDYNMEM
LIS

LNKMAPRTN
LIS

LINKMSG
LIS

LNKELAPST
LIS

LNKIMGOUT
LIS

LNKDATBAS
LIS

LNKGETCMD
LIS

LNKINI
LIS

LNKERROUT
LIS