```
LLL                 IIIIIIIII     BBBBBBBBBBBB     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
LLL                 IIIIIIIII     BBBBBBBBBBBB     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
LLL                 IIIIIIIII     BBBBBBBBBBBB     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBBBBBBBBBBB     RRRRRRRRRRRR           TTT        LLL
LLL                    III        BBBBBBBBBBBB     RRRRRRRRRRRR           TTT        LLL
LLL                    III        BBBBBBBBBBBB     RRRRRRRRRRRR           TTT        LLL
LLL                    III        BBB         BBB  RRR   RRR              TTT        LLL
LLL                    III        BBB         BBB  RRR   RRR              TTT        LLL
LLL                    III        BBB         BBB  RRR   RRR              TTT        LLL
LLL                    III        BBB         BBB  RRR      RRR           TTT        LLL
LLL                    III        BBB         BBB  RRR      RRR           TTT        LLL
LLL                    III        BBB         BBB  RRR      RRR           TTT        LLL
LLLLLLLLLLLLLLL     IIIIIIIII     BBBBBBBBBBBB     RRR         RRR        TTT        LLLLLLLLLLLLLLL
LLLLLLLLLLLLLLL     IIIIIIIII     BBBBBBBBBBBB     RRR         RRR        TTT        LLLLLLLLLLLLLLL
LLLLLLLLLLLLLLL     IIIIIIIII     BBBBBBBBBBBB     RRR         RRR        TTT        LLLLLLLLLLLLLLL
```

FILEID**LIBFILSCA

```
LL              IIIIII    BBBBBBBB    FFFFFFFFFF    IIIIII    LL              SSSSSSSS    CCCCCCCC    AAAAAA
LL              IIIIII    BBBBBBBB    FFFFFFFFFF    IIIIII    LL              SSSSSSSS    CCCCCCCC    AAAAAA
LL                II      BB      BB  FF              II      LL            SS          CC          AA      AA
LL                II      BB      BB  FF              II      LL            SS          CC          AA      AA
LL                II      BB      BB  FF              II      LL            SS          CC          AA      AA
LL                II      BB      BB  FF              II      LL            SS          CC          AA      AA
LL                II      BBBBBBBB    FFFFFFF         II      LL              SSSSSS    CC          AA      AA
LL                II      BBBBBBBB    FFFFFFF         II      LL              SSSSSS    CC          AA      AA
LL                II      BB      BB  FF              II      LL                    SS  CC          AAAAAAAAAA
LL                II      BB      BB  FF              II      LL                    SS  CC          AAAAAAAAAA
LL                II      BB      BB  FF              II      LL                    SS  CC          AA      AA      ....
LL                II      BB      BB  FF              II      LL                    SS  CC          AA      AA      ....
LLLLLLLLLL      IIIIII    BBBBBBBB    FF            IIIIII    LLLLLLLLLL  SSSSSSSS      CCCCCCCC  AA      AA      ....
LLLLLLLLLL      IIIIII    BBBBBBBB    FF            IIIIII    LLLLLLLLLL  SSSSSSSS      CCCCCCCC  AA      AA      ....

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
```

```
;     1          0001  0 MODULE LIB$FILESCAN (                                  ! LIBFILESCA.B32
;     2          0002  0                 %TITLE 'Search a file wildcard sequence of files'
;     3          0003  0                 IDENT = 'V03-024'
;     4          0004  0                 ) =
;     5          0005  1 BEGIN
;     6          0006  1 !
;     7          0007  1 !**********************************************************************
;     8          0008  1 !*                                                                    *
;     9          0009  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                          *
;    10          0010  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.           *
;    11          0011  1 !*  ALL RIGHTS RESERVED.                                           *
;    12          0012  1 !*                                                                 *
;    13          0013  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
;    14          0014  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;    15          0015  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
;    16          0016  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;    17          0017  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
;    18          0018  1 !*  TRANSFERRED.                                                   *
;    19          0019  1 !*                                                                *
;    20          0020  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;    21          0021  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;    22          0022  1 !*  CORPORATION.                                                   *
;    23          0023  1 !*                                                                *
;    24          0024  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
;    25          0025  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.        *
;    26          0026  1 !*                                                                *
;    27          0027  1 !*                                                                *
;    28          0028  1 !**********************************************************************
;    29          0029  1 !
;    30          0030  1 !++
;    31          0031  1 ! FACILITY: General Utility Library
;    32          0032  1 !
;    33          0033  1 ! ABSTRACT:
;    34          0034  1 !       This module contains routines which can be used to find all
;    35          0035  1 !       files that match a wildcard file specification.
;    36          0036  1 !
;    37          0037  1 ! ENVIRONMENT:
;    38          0038  1 !       VAX/VMS, User mode, Non-AST re-entrant
;    39          0039  1 !
;    40          0040  1 ! AUTHOR:       Tim Halvorsen, CREATION DATE: 1-AUG-1979
;    41          0041  1 !
;    42          0042  1 ! MODIFIED BY:
;    43          0043  1 !
;    44          0044  1 !       V03-024 BLS0331         Benn Schreiber          9-JUL-1984
;    45          0045  1 !               Remove conditional compilation.
;    46          0046  1 !
;    47          0047  1 !       V03-023 BLS0321         Benn Schreiber         22-MAY-1984
;    48          0048  1 !               If wild version, do not put it on related list over
;    49          0049  1 !               and over.
;    50          0050  1 !
;    51          0051  1 !       V03-022 BLS0319         Benn Schreiber         16-MAY-1984
;    52          0052  1 !               For find_file, never use move_default to put at the
;    53          0053  1 !               end.  Save address of newly created default nam block
;    54          0054  1 !               for future reference.
;    55          0055  1 !
;    56          0056  1 !       V03-021 BLS0317         Benn Schreiber         14-MAY-1984
;    57          0057  1 !               If a new default file spec is seen, put it in the
```

```
;    58          0058  1 !                 list of related files at current location, not at
;    59          0059  1 !                 end.
;    60          0060  1 !
;    61          0061  1 !       V03-020 BLS0316         Benn Schreiber          13-MAY-1984
;    62          0062  1 !                 Remove over-anxious edit in find_file.
;    63          0063  1 !
;    64          0064  1 !       V03-019 BLS0313         Benn Schreiber           7-MAY-1984
;    65          0065  1 !                 Fix checking of default string in find_file to correctly
;    66          0066  1 !                 decide whether to set default string in FAB.
;    67          0067  1 !
;    68          0068  1 !       V03-018 BLS0308         Benn Schreiber          27-APR-1984
;    69          0069  1 !                 In lib$find_file, fix wildcard version, and passing
;    70          0070  1 !                 same filespec twice if nowild not set.  Also, in
;    71          0071  1 !                 lib$file_scan_end, allow calling without fab argument.
;    72          0072  1 !
;    73          0073  1 !       V03-017 BLS0307         Benn Schreiber          26-APR-1984
;    74          0074  1 !                 Fix use of NOWILD in lib$find_file.
;    75          0075  1 !
;    76          0076  1 !       V03-016 BLS0297         Benn Schreiber           9-APR-1984
;    77          0077  1 !                 Correctly allow changing of the default file specification
;    78          0078  1 !                 on new file specs in multi-file parses (lib$find_file).
;    79          0079  1 !
;    80          0080  1 !       V03-015 BLS0283         Benn Schreiber           6-MAR-1984
;    81          0081  1 !                 Don't try to allocate 0-length string in findfile.
;    82          0082  1 !
;    83          0083  1 !       V03-014 BLS0275         Benn Schreiber          25-FEB-1984
;    84          0084  1 !                 Correct parse of null string to clear ESS and RSS
;    85          0085  1 !                                                                       ;
;    86          0086  1 !       V03-013 BLS0264         Benn Schreiber          24-Jan-1984
;    87          0087  1 !                 Add support for multiple input filename stickyness.
;    88          0088  1 !                 Add new routines to deallocate saved context. Add conditional
;    89          0089  1 !                 to compile new interface for V3, for shipment in 3.6.
;    90          0090  1 !
;    91          0091  1 !       V03-012 BLS0254         Benn Schreiber          19-Dec-1983
;    92          0092  1 !                 Correct handling of null file specs in LIB$FIND_FILE.
;    93          0093  1 !
;    94          0094  1 !       V03-011 BLS0243         Benn Schreiber          20-Oct-1983
;    95          0095  1 !                 Fix handling of related nam block for searchlists.
;    96          0096  1 !
;    97          0097  1 !       V03-010 BLS0198         Benn Schreiber          13-Dec-1982
;    98          0098  1 !                 If non-wildcard call, do a parse of null string to clear
;    99          0099  1 !                 RMS internal context.
;   100          0100  1 !
;   101          0101  1 !       V03-009 BLS0174         Benn Schreiber           1-JUN-1982
;   102          0102  1 !                 Use lib$analyze_sdesc_r2 for arguments passed as
;   103          0103  1 !                 string descriptors
;   104          0104  1 !
;   105          0105  1 !       V03-008 BLS0133         Benn Schreiber          11-Jan-1982
;   106          0106  1 !                 Make lib$file_scan continue when it gets nopriv.  Make
;   107          0107  1 !                 lib$file_scan always copy expanded name string to resultant
;   108          0108  1 !                 name string on errors and network non-wild files
;   109          0109  1 !
;   110          0110  1 !       V03-007 TMK0001         Todd M. katz            31-Dec-1981
;   111          0111  1 !                 Check for a PPF file before doing a $SEARCH. Do not do
;   112          0112  1 !                 searches on PPF files.
;   113          0113  1 !
;   114          0114  1 !       V03-006 MLJ0044         Martin L. Jack, 8-Sep-1981  14:00
```

```
;  115          0115  1 !                 Correct problems when $PARSE fails.
;  116          0116  1 !
;  117          0117  1 !       V03-005 BLS0071         Benn Schreiber          22-Aug-1981
;  118          0118  1 !                 Correct looping if priv violation in lib$find_file
;  119          0119  1 !
;  120          0120  1 !       V03-004 BLS0065         Benn Schreiber           4-Aug-1981
;  121          0121  1 !                 Fix handling of devices mounted foreign, and move
;  122          0122  1 !                 saved status into a longword out of the fab for lib$find_file.
;  123          0123  1 !
;  124          0124  1 !       V03-003 BLS0041         Benn Schreiber          23-Feb-1981
;  125          0125  1 !                 Correct error in call to lib$free_vm
;  126          0126  1 !
;  127          0127  1 !       V03-002 BLS0027         Benn Schreiber          28-Nov-1980
;  128          0128  1 !                 Correct protection violation handling in LIB$FIND_FILE
;  129          0129  1 !
;  130          0130  1 !       V03-001 LMK0001         Len Kawell              19-Sep-1980
;  131          0131  1 !                 Recode in BLISS and add LIB$FILE_SEARCH.
;  132          0132  1 !--
```

```
;  134          0133  1 %SBTTL  'Declarations';
;  135          0134  1
;  136          0135  1 SWITCHES
;  137          0136  1         ADDRESSING_MODE (EXTERNAL = GENERAL,             !Declare addressing modes
;  138          0137  1                         NONEXTERNAL = WORD_RELATIVE);
;  139          0138  1 LIBRARY
;  140          0139  1         'RTLSTARLE';                                    !System symbols
;  141          0140  1
;  142          0141  1 REQUIRE
;  143          0142  1         'RTLIN:RTLPSECT';                               !Define PSECT declaration macros
;  144          0237  1
;  145          0238  1 DECLARE_PSECTS (LIB);                                   !Declare PSECTs for LIB$ facility
;  146          0239  1
;  147          0240  1 !
;  148          0241  1 ! LINKAGES:
;  149          0242  1 !
;  150          0243  1
;  151          0244  1 LINKAGE
;  152          0245  1     JSB_ANALYZE_SDESC = JSB (REGISTER=0; REGISTER=1, REGISTER=2) :
;  153          0246  1                             NOTUSED (3,4,5,6,7,8,9,10,11);
;  154          0247  1
;  155          0248  1 FORWARD ROUTINE
;  156          0249  1         COPY_ESL_TO_RSL : NOVALUE,                      !Copies ESL to RSL
;  157          0250  1         COPY_FILE_STRING,                               !Copy file string to VM
;  158          0251  1         DUMMY_ROUTINE,                                  !Dummy suc/err routine
;  159          0252  1         LIB$FILE_SCAN,                                  !Wild card scan using FAB
;  160          0253  1         COPY_RESULT_NAME : NOVALUE,                     !Copy result string
;  161          0254  1         LIB$FIND_FILE;                                  !Wild card scan using context
;  162          0255  1
;  163          0256  1 EXTERNAL ROUTINE
;  164          0257  1         LIB$ANALYZE_SDESC_R2: JSB_ANALYZE_SDESC,!Analyze string descriptor
;  165          0258  1         LIB$FREE_VM,                                    !Deallocate virtual memory
;  166          0259  1         LIB$GET_VM,                                     !Allocate virtual memory
;  167          0260  1         LIB$SCOPY_R_DX;                                 !Copy string
;  168          0261  1 !
;  169          0262  1 ! Local storage
;  170          0263  1 !
;  171          0264  1 PSECT OWN = _LIB$CODE;
;  172          0265  1 PSECT PLIT = _LIB$CODE;
;  173          0266  1
;  174          0267  1 OWN
;  175          0268  1         RMSNMF : LONG INITIAL (RMS$_NMF);
;  176          0269  1 BIND
;  177          0270  1         WILD_VER = UPLIT(';*');
;  178          0271  1 !
;  179          0272  1 ! Define the storage context used by LIB$FIND_FILE
;  180          0273  1 !
;  181          0274  1 LITERAL
;  182          0275  1         NAM_OFF = FAB$C_BLN,                            ! Offset to NAM block
;  183          0276  1         RNAM_OFF = NAM_OFF + NAM$C_BLN,                 ! Offset to related NAM block
;  184          0277  1         ESBUF_OFF = RNAM_OFF + NAM$C_BLN,               ! Offset to expanded name
;  185          0278  1         RSBUF_OFF = ESBUF_OFF + NAM$C_MAXRSS,           ! Offset to result name
;  186          0279  1         STATUS_OFF = RSBUF_OFF + NAM$C_MAXRSS,          ! Offset to next status
;  187          0280  1         INTFLAGS_OFF = STATUS_OFF + 4,                  ! Offset to internal flags
;  188          0281  1         DNAM_PTR = INTFLAGS_OFF + 4,                    ! Pointer to default string
;  189          0282  1                                                         !  NAM block
;  190          0283  1         CONTEXT_SIZE = DNAM_PTR + 4;                    ! Total size of structure
```

```
;  191          0284  1 !
;  192          0285  1 ! Define shared messages
;  193          0286  1 !
;  194        P 0287  1 $SHR_MSGDEF(LIB,21,LOCAL,
;  195          0288  1           (NOWILD,ERROR));          !Wildcard filespec and NOWILD set
```

```
;  197     0289  1 %SBTTL  'COPY_FILE_STRING Copy filename string for next input file parse';
;  198     0290  1 ROUTINE COPY_FILE_STRING(CONTEXT,FAB) =
;  199     0291  1 !---
;  200     0292  1 !       This routine copies the file specified by fab$b_fns/l_fna to
;  201     0293  1 !       a block of memory allocated with lib$get_vm.  This block also
;  202     0294  1 !       contains a nam block.  These are used on a subsequent call to
;  203     0295  1 !       filescan to provide the related file name(s), and is done this
;  204     0296  1 !       way because RMS needs access to the filename strings of all previous
;  205     0297  1 !       file specifications should any of them contain a searchlist.
;  206     0298  1 !
;  207     0299  1 ! Inputs:
;  208     0300  1 !
;  209     0301  1 !       Context = 0 or address of context longword passed by user
;  210     0302  1 !       fab = address of fab
;  211     0303  1 !
;  212     0304  1 ! Outputs:
;  213     0305  1 !
;  214     0306  1 !       The memory is allocated and the block is added into the list
;  215     0307  1 !       of related nam blocks.  If no context was passed by the user,
;  216     0308  1 !       nothing is done.
;  217     0309  1 !
;  218     0310  1 ! NOTE: If compiling for V3 system, the expanded string from the NAM
;  219     0311  1 !       block is used, rather than fns/fna.  Also, the related NAM block
;  220     0312  1 !       (found via NAM$L_RLF) must already point to a valid related
;  221     0313  1 !       NAM block.
;  222     0314  1 !---
;  223     0315  2 BEGIN
;  224     0316  2 MAP
;  225     0317  2         FAB : REF $BBLOCK;
;  226     0318  2
;  227     0319  2 LOCAL
;  228     0320  2         CTX : REF VECTOR[,LONG],
;  229     0321  2         STRSIZE,
;  230     0322  2         RNAM : REF $BBLOCK,
;  231     0323  2         NAM : REF $BBLOCK,
;  232     0324  2         NEWBLOCK : REF $BBLOCK,
;  233     0325  2         STATUS;
;  234     0326  2
;  235     0327  2 !
;  236     0328  2 ! If no context passed by user, then nothing to do.
;  237     0329  2 !
;  238     0330  2 IF (CTX = .CONTEXT) EQL 0
;  239     0331  2     THEN RETURN 1;
;  240     0332  2 !
;  241     0333  2 ! Allocate a block big enough for a NAM block and the filename string
;  242     0334  2 !
;  243     0335  2 STRSIZE = .FAB[FAB$B_FNS];
;  244     0336  2 STATUS = LIB$GET_VM(%REF(NAM$C_BLN+.STRSIZE),NEWBLOCK);
;  245     0337  2 IF NOT .STATUS
;  246     0338  2     THEN RETURN .STATUS;
;  247     0339  2 !
;  248     0340  2 ! Initialize the NAM block, and copy the filename string
;  249     0341  2 !
;  250     0342  2 CH$MOVE(NAM$C_BLN,.FAB[FAB$L_NAM],.NEWBLOCK);
;  251     0343  2 NEWBLOCK[NAM$B_RSL] = .STRSIZE;
;  252     0344  2 NEWBLOCK[NAM$B_RSS] = .STRSIZE;
;  253     0345  2 NEWBLOCK[NAM$L_RSA] = .NEWBLOCK+NAM$C_BLN;
```

```
;  254          0346  2 NEWBLOCK[NAM$B_ESS] = 0;
;  255          0347  2 NEWBLOCK[NAM$B_ESL] = 0;
;  256          0348  2 CH$FILL(0,NAM$C_DVI,NEWBLOCK[NAM$T_DVI]);
;  257          0349  2 CH$MOVE(.STRSIZE,.FAB[FAB$L_FNA],.NEWBLOCK+NAM$C_BLN);
;  258          0350  2 !
;  259          0351  2 ! Link this nam/filespec block into the list of blocks
;  260          0352  2 !
;  261          0353  2 NEWBLOCK[NAM$L_RLF] = .CTX[0];
;  262          0354  2 CTX[0] = .NEWBLOCK;
;  263          0355  2 RETURN 1
;  264          0356  1 END;


;                                                       .TITLE  LIB$FILESCAN Search a file wildcard sequence of
                                                                        files
                                                       .IDENT  \V03-024\

                                                       .PSECT  _LIB$CODE,NOWRT,  SHR,  PIC,2

                                          000182CA  00000 RMSNMF: .LONG   99018                                           ;
                                    00 00 2A 3B     00004 P.AAA:  .ASCII  \;*\<0><0>                                      ;

                                                      WILD_VER=         P.AAA
                                                       .EXTRN  LIB$ANALYZE_SDESC_R2
                                                       .EXTRN  LIB$FREE_VM, LIB$GET_VM
                                                       .EXTRN  LIB$SCOPY_R_DX

                                          03FC 00000 COPY_FILE_STRING:
                                                       .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9                           ; 0290
                                    5E          08 C2 00002         SUBL2   #8, SP                                      ;
                                    59    04    AC D0 00005         MOVL    CONTEXT, CTX                               ; 0330
                                                4C 13 00009         BEQL    1$                                         ;
                                    58    08    AC D0 0000B         MOVL    FAB, R8                                    ; 0335
                                    56    34    A8 9A 0000F         MOVZBL  52(R8), STRSIZE                            ;
                                          04    AE 9F 00013         PUSHAB  NEWBLOCK                                   ; 0336
                                       04 AE    60 A6 9E 00016      MOVAB   96(R6), 4(SP)                              ;
                                          04    AE 9F 0001B         PUSHAB  4(SP)                                      ;
                                00000000G 00       02 FB 0001E      CALLS   #2, LIB$GET_VM                             ;
                                    32          50 E9 00025         BLBC    STATUS, 2$                                 ; 0337
                                    57    04    AE D0 00028         MOVL    NEWBLOCK, R7                               ; 0342
                                67  28    B8  0060  8F 28 0002C     MOVC3   #96, @40(R8), (R7)                         ;
                                    03    A7        56 90 00033     MOVB    STRSIZE, 3(R7)                             ; 0343
                                    02    A7        56 90 00037     MOVB    STRSIZE, 2(R7)                             ; 0344
                                    04    A7  60    A7 9E 0003B     MOVAB   96(R7), 4(R7)                              ; 0345
                                              0A    A7 B4 00040     CLRW    10(R7)                                     ; 0346
                             10       00       6E    00 2C 00043     MOVC5   #0, (SP), #0, #16, 20(R7)                  ; 0348
                                              14    A7    00048     ;                                                  ;
                          60 A7    2C  B8        56 28 0004A     MOVC3   STRSIZE, @44(R8), 96(R7)                   ; 0349
                                    10    A7        69 D0 00050     MOVL    (CTX), 16(R7)                              ; 0353
                                          69        57 D0 00054     MOVL    R7, (CTX)                                  ; 0354
                                          50        01 D0 00057 1$: MOVL    #1, R0                                     ; 0355
                                                    04 0005A 2$:    RET                                                ; 0356

; Routine Size:  91 bytes,    Routine Base:  _LIB$CODE + 0008
```

```
;  266      0357  1 %SBTTL  'COPY_ESL_TO_RSL Copy Expanded Name String to Resultant';
;  267      0358  1 ROUTINE COPY_ESL_TO_RSL(FAB,NAM) : NOVALUE =
;  268      0359  1 !---
;  269      0360  1 !           This routine sets up the resultant name string data
;  270      0361  1 !           in the NAM block.  It is called in the case of an
;  271      0362  1 !           error from $PARSE/$SEARCH, or on network non-wild
;  272      0363  1 !           card operations.
;  273      0364  1 !
;  274      0365  1 !    Inputs:
;  275      0366  1 !
;  276      0367  1 !           FAB = FAB address
;  277      0368  1 !           NAM = NAM address
;  278      0369  1 !
;  279      0370  1 !    Outputs:
;  280      0371  1 !
;  281      0372  1 !           NAM$B_RSL setup with length of string copied into
;  282      0373  1 !           resultant name string buffer pointed to by NAM$L_RSA.
;  283      0374  1 !---
;  284      0375  2 BEGIN
;  285      0376  2
;  286      0377  2 MAP
;  287      0378  2         FAB:    REF BLOCK[,BYTE],               ! FAB structure
;  288      0379  2         NAM:    REF BLOCK[,BYTE];               ! NAM structure
;  289      0380  2
;  290      0381  2 IF .NAM[NAM$B_RSL] EQL 0                        ! If name not set up
;  291      0382  2     THEN IF (NAM[NAM$B_RSL] = .NAM[NAM$B_ESL]) NEQ 0    ! If expanded string present
;  292      0383  2         THEN CH$MOVE(MINU(.NAM[NAM$B_RSS],
;  293      0384  2                           .NAM[NAM$B_ESL]),! then use it
;  294      0385  2                 .NAM[NAM$L_ESA],.NAM[NAM$L_RSA])
;  295      0386  3         ELSE BEGIN                              ! No expanded string, use
;  296      0387  3             NAM[NAM$B_RSL] = .FAB[FAB$B_FNS];   !  the filename string from FAB
;  297      0388  3             CH$MOVE(MINU(.NAM[NAM$B_RSS],.FAB[FAB$B_FNS]),
;  298      0389  3                         .FAB[FAB$L_FNA],.NAM[NAM$L_RSA]);
;  299      0390  2         END;
;  300      0391  2 RETURN;
;  301      0392  1 END;


                                        007C 00000 COPY_ESL_TO_RSL:
                                                          .WORD   Save R2,R3,R4,R5,R6          ; 0358
                    56      08   AC  D0 00002             MOVL    NAM, R6                      ; 0381
                            03   A6  95 00006             TSTB    3(R6)                        ;
                                 39  12 00009             BNEQ    4$                           ;
               03   A6      0B   A6  90 0000B             MOVB    11(R6), 3(R6)                ; 0382
                                 15  13 00010             BEQL    2$                           ;
                    51      02   A6  9A 00012             MOVZBL  2(R6), R1                    ; 0384
                    51      0B   A6  91 00016             CMPB    11(R6), R1                   ;
                                 04  1E 0001A             BGEQU   1$                           ;
                    51      0B   A6  9A 0001C             MOVZBL  11(R6), R1                   ;
      04   B6       0C      B6       51  28 00020 1$:     MOVC3   R1, @12(R6), @4(R6)          ; 0385
                                     04 00026             RET                                  ; 0383
                    50      04   AC  D0 00027 2$:         MOVL    FAB, R0                      ; 0387
               03   A6      34   A0  90 0002B             MOVB    52(R0), 3(R6)                ;
                    51      02   A6  9A 00030             MOVZBL  2(R6), R1                    ; 0388
```

```
                        51       34   A0  91 00034          CMPB    52(R0), R1                              ;
                                      04  1E 00038          BGEQU   3$                                      ;
                        51       34   A0  9A 0003A          MOVZBL  52(R0), R1                              ;
          04   B6       2C   B0           51  28 0003E 3$:  MOVC3   R1, @44(R0), @4(R6)                     ; 0389
                                          04 00044 4$:      RET                                             ; 0392

; Routine Size:  69 bytes,    Routine Base:  _LIB$CODE + 0063
```

```
;  303        0393  1 %SBTTL  'DUMMY_ROUTINE Dummy success/error routine';
;  304        0394  1 ROUTINE DUMMY_ROUTINE = RETURN 1;


                                  0000 00000 DUMMY_ROUTINE:
                                                  .WORD    Save nothing                                   ; 0394
                     50          01  D0 00002        MOVL     #1, R0                                         ;
                                     04 00005        RET                                                     ;

; Routine Size:  6 bytes,    Routine Base:  _LIB$CODE + 00A8
```

```
;   306          0395  1 %SBTTL  'PARSE_NULL_STRING Parse null string to deallocate RMS context';
;   307          0396  1 ROUTINE PARSE_NULL_STRING(FAB) =
;   308          0397  1 !---
;   309          0398  1 !       Parse the null string to force RMS to deallocate any context
;   310          0399  1 !       saved by NAM$V_SVCTX
;   311          0400  1 !
;   312          0401  1 ! Inputs:
;   313          0402  1 !
;   314          0403  1 !       fab = address of the fab
;   315          0404  1 !
;   316          0405  1 ! Implicit outputs:
;   317          0406  1 !
;   318          0407  1 !       $PARSE done on the fab to deallocate saved context
;   319          0408  1 !
;   320          0409  1 !--
;   321          0410  2 BEGIN
;   322          0411  2 MAP
;   323          0412  2         FAB : REF $BBLOCK;
;   324          0413  2
;   325          0414  2 LOCAL
;   326          0415  2         NAM : REF $BBLOCK;
;   327          0416  2 !
;   328          0417  2 ! Set up to parse the null string
;   329          0418  2 !
;   330          0419  2 NAM = .FAB[FAB$L_NAM];
;   331          0420  2 IF .NAM NEQ 0
;   332          0421  3 THEN    BEGIN
;   333          0422  3         NAM[NAM$V_SVCTX] = 0;
;   334          0423  3         NAM[NAM$V_SYNCHK] = 1;          !In case of network SET DEFAULT
;   335          0424  3         NAM[NAM$B_ESL] = 0;
;   336          0425  3         NAM[NAM$B_RSL] = 0;
;   337          0426  3         NAM[NAM$B_ESS] = 0;
;   338          0427  3         NAM[NAM$B_RSS] = 0;
;   339          0428  3         NAM[NAM$L_RLF] = 0;
;   340          0429  2         END;
;   341          0430  2 FAB[FAB$B_FNS] = 0;
;   342          0431  2 FAB[FAB$B_DNS] = 0;
;   343          0432  2 $PARSE(FAB=.FAB);
;   344          0433  2 RETURN 1
;   345          0434  1 END;


                                                .EXTRN  SYS$PARSE

                                        0000 00000 PARSE_NULL_STRING:
                                                .WORD   Save nothing                            ; 0396
                                    51      04  AC  D0 00002          MOVL    FAB, R1            ; 0419
                                    50      28  A1  D0 00006          MOVL    40(R1), NAM        ;
                                                12  13 0000A          BEQL    1$                 ; 0420
                                33  A0          80  8F  8A 0000C     BICB2   #128, 51(NAM)      ; 0422
                                08  A0          08  88 00011         BISB2   #8, 8(NAM)         ; 0423
                                                02  A0  B4 00015     CLRW    2(NAM)             ; 0427
                                                0A  A0  B4 00018     CLRW    10(NAM)            ; 0426
                                                10  A0  D4 0001B     CLRL    16(NAM)            ; 0428
                                                34  A1  B4 0001E 1$: CLRW    52(R1)             ; 0430
                                                    51  DD 00021     PUSHL   R1                 ; 0432
```

```
                    00000000G  00          01  FB 00023         CALLS   #1, SYS$PARSE                            ;
                               50          01  D0 0002A         MOVL    #1, R0                                  ; 0433
                                               04 0002D         RET                                             ; 0434

; Routine Size:  46 bytes,    Routine Base:  _LIB$CODE + 00AE
```

```
;  347          0435  1 ROUTINE MOVE_DEFAULT_STRING(CONTEXT,FAB,DNMPTR) =
;  348          0436  1 !---
;  349          0437  1 !       Move the default string from the FAB to a NAM block at the end
;  350          0438  1 !       of the related NAM block list.
;  351          0439  1 !
;  352          0440  1 ! Inputs:
;  353          0441  1 !
;  354          0442  1 !       context = address of context longword
;  355          0443  1 !       fab = fab address
;  356          0444  1 !       dnmptr = (optional) address of longword to store nam block address
;  357          0445  1 !
;  358          0446  1 ! Outputs:
;  359          0447  1 !
;  360          0448  1 !       fab[fab$b_dns] zeroed.  Default name string copied into allocated
;  361          0449  1 !       nam block which is linked at the end of the related file blocks.
;  362          0450  1 !
;  363          0451  1 !---
;  364          0452  2 BEGIN
;  365          0453  2 MAP
;  366          0454  2         CONTEXT : REF VECTOR[,LONG],
;  367          0455  2         FAB : REF $BBLOCK,
;  368          0456  2         DNMPTR : REF VECTOR[,LONG];
;  369          0457  2
;  370          0458  2 BUILTIN
;  371          0459  2         NULLPARAMETER;
;  372          0460  2
;  373          0461  2 LOCAL
;  374          0462  2         STATUS,
;  375          0463  2         PNAM : REF $BBLOCK,
;  376          0464  2         RNAM : REF $BBLOCK,
;  377          0465  2         TNAM : REF $BBLOCK;
;  378          0466  2
;  379          0467  2 IF .FAB[FAB$B_DNS] EQL 0
;  380          0468  2 THEN
;  381          0469  2         RETURN 1;
;  382          0470  2 !
;  383          0471  2 ! Search the NAM blocks looking for a default file string
;  384          0472  2 ! block (noted by [NAM$B_ESS] = %X'0D') and see if that string
;  385          0473  2 ! is same as new string.  Return successfully if so. If not,
;  386          0474  2 ! then deallocate the one from the list, as we need a new block.
;  387          0475  2 !
;  388          0476  2 TNAM = CONTEXT[0] - $BYTEOFFSET(NAM$L_RLF);
;  389          0477  2 PNAM = .TNAM;
;  390          0478  2 WHILE .TNAM[NAM$L_RLF] NEQ 0
;  391          0479  3 DO      BEGIN
;  392          0480  3         PNAM = .TNAM;
;  393          0481  3         TNAM = .TNAM[NAM$L_RLF];
;  394          0482  3         IF .TNAM[NAM$B_ESS] EQL %X'0D'                                  ;
;  395          0483  4         THEN    BEGIN
;  396          0484  4                 IF CH$EQL(.FAB[FAB$B_DNS],.FAB[FAB$L_DNA],
;  397          0485  4                                 .TNAM[NAM$B_RSL],.TNAM[NAM$L_RSA],0)
;  398          0486  5                 THEN    BEGIN
;  399          0487  5                         FAB[FAB$B_DNS] = 0;
;  400          0488  5                         RETURN 1;
;  401          0489  4                         END;
;  402          0490  4                 LIB$FREE_VM(%REF(NAM$C_BLN + .TNAM[NAM$B_RSL]),%REF(.TNAM));
;  403          0491  4                 PNAM[NAM$L_RLF] = 0;
```

```
      0492 4                    EXITLOOP;
      0493 3                    END;
      0494 2            END;
      0495 2 !
      0496 2 ! Allocate a NAM+string block
      0497 2 !
      0498 2 STATUS = LIB$GET_VM(%REF(NAM$C_BLN+.FAB[FAB$B_DNS]),RNAM);
      0499 2 IF NOT .STATUS
      0500 2 THEN
      0501 2         RETURN .STATUS;
      0502 2 !
      0503 2 ! Link into the list, initialize the NAM block, copy the default name string.
      0504 2 !
      0505 2 PNAM[NAM$L_RLF] = .RNAM;
    P 0506 2 $NAM_INIT(NAM=.RNAM,
      0507 2         RSA=.RNAM+NAM$C_BLN);
      0508 2 RNAM[NAM$B_RSL] = .FAB[FAB$B_DNS];
      0509 2 RNAM[NAM$B_ESS] = %X'0D';              !Identify it as default string nam block
      0510 2 CH$MOVE(.FAB[FAB$B_DNS],.FAB[FAB$L_DNA],.RNAM+NAM$C_BLN);
      0511 2 FAB[FAB$B_DNS] = 0;
      0512 2 IF NOT NULLPARAMETER(3)
      0513 2 THEN
      0514 2         DNMPTR[0] = .RNAM;
      0515 2 RETURN 1
      0516 1 END;



                                    01FC 00000 MOVE_DEFAULT_STRING:
                                               .WORD   Save R2,R3,R4,R5,R6,R7,R8        ; 0435
                   5E          0C  C2 00002    SUBL2   #12, SP                          ;
                   57      08  AC  D0 00005    MOVL    FAB, R7                          ; 0467
                   58      35  A7  9E 00009    MOVAB   53(R7), R8                       ;
                               68  95 0000D    TSTB    (R8)                             ;
                               2D  13 0000F    BEQL    2$                               ;
         54      04   AC       10  C3 00011    SUBL3   #16, CONTEXT, TNAM               ; 0476
                   55          54  D0 00016    MOVL    TNAM, PNAM                       ; 0477
                           10  A4  D5 00019 1$: TSTL   16(TNAM)                         ; 0478
                               43  13 0001C    BEQL    4$                               ;
                   55          54  D0 0001E    MOVL    TNAM, PNAM                       ; 0480
                   54      10  A4  D0 00021    MOVL    16(TNAM), TNAM                   ; 0481
                   0D      0A  A4  91 00025    CMPB    10(TNAM), #13                    ; 0482
                               EE  12 00029    BNEQ    1$                               ;
                   51          68  9A 0002B    MOVZBL  (R8), R1                         ; 0484
                   50      03  A4  9A 0002E    MOVZBL  3(TNAM), R0                      ; 0485
   50       00      30   B7    51  2D 00032    CMPC5   R1, @48(R7), #0, R0, @4(TNAM)    ; 0484
                           04  B4     00038                                             ;
                               04  12 0003A    BNEQ    3$                               ;
                               68  94 0003C    CLRB    (R8)                             ; 0487
                               78  11 0003E 2$: BRB    5$                               ; 0488
              04   AE          54  D0 00040 3$: MOVL   TNAM, 4(SP)                      ; 0490
                           04  AE  9F 00044    PUSHAB  4(SP)                            ;
              04   AE      03  A4  9A 00047    MOVZBL  3(TNAM), 4(SP)                   ;
              04   AE 00000060 8F  C0 0004C    ADDL2   #96, 4(SP)                       ;
                           04  AE  9F 00054    PUSHAB  4(SP)                            ;
```

```
                    00000000G  00              02  FB 00057        CALLS    #2, LIB$FREE_VM                    ;
                                     10        A5  D4 0005E        CLRL     16(PNAM)                           ; 0491
                                     08        AE  9F 00061 4$:    PUSHAB   RNAM                               ; 0498
                          08   AE              68  9A 00064        MOVZBL   (R8), 8(SP)                        ;
                          08   AE 00000060     8F  C0 00068        ADDL2    #96, 8(SP)                         ;
                                     08        AE  9F 00070        PUSHAB   8(SP)                              ;
                    00000000G  00              02  FB 00073        CALLS    #2, LIB$GET_VM                     ;
                               3E              50  E9 0007A        BLBC     STATUS, 6$                         ; 0499
                               56    08        AE  D0 0007D        MOVL     RNAM, R6                           ; 0505
                          10   A5              56  D0 00081        MOVL     R6, 16(PNAM)                       ;
   0060    8F          00      6E              00  2C 00085        MOVC5    #0, (SP), #0, #96, (R6)            ; 0507
                                               66     0008C                                                    ;
                               66  6002        8F  B0 0008D        MOVW     #24578, (R6)                       ;
                          04   A6    60        A6  9E 00092        MOVAB    96(R6), 4(R6)                      ;
                          03   A6              68  90 00097        MOVB     (R8), 3(R6)                        ; 0508
                          0A   A6              0D  90 0009B        MOVB     #13, 10(R6)                        ; 0509
                               50              68  9A 0009F        MOVZBL   (R8), R0                           ; 0510
              60   A6     30   B7              50  28 000A2        MOVC3    R0, @48(R7), 96(R6)                ;
                                               68  94 000A8        CLRB     (R8)                               ; 0511
                               03              6C  91 000AA        CMPB     (AP), #3                           ; 0512
                                               09  1F 000AD        BLSSU    5$                                 ;
                                     0C        AC  D5 000AF        TSTL     12(AP)                             ;
                                               04  13 000B2        BEQL     5$                                 ;
                          0C   BC              56  D0 000B4        MOVL     R6, @DNMPTR                        ; 0514
                               50              01  D0 000B8 5$:    MOVL     #1, R0                             ; 0515
                                                   04 000BB 6$:    RET                                         ; 0516

; Routine Size:  188 bytes,    Routine Base:  _LIB$CODE + 00DC
```

```
;  430          0517  1 %SBTTL  'LIB$FILE_SCAN File scan given FAB and NAM block';
;  431          0518  1 GLOBAL ROUTINE LIB$FILE_SCAN(FAB,SUCCESS_RTN,ERROR_RTN,CONTEXT) =
;  432          0519  1 !---
;  433          0520  1 !
;  434          0521  1 !       This routine is called with a wildcard file specification
;  435          0522  1 !       and calls a specified set of action routines for each file
;  436          0523  1 !       and/or error found in the wildcard sequence.  Certain errors
;  437          0524  1 !       are checked for in order to allow the search sequence to be
;  438          0525  1 !       completed even though errors like nopriv are present.
;  439          0526  1 !       Stickyness is also handled if this routine is called once
;  440          0527  1 !       for each file specification parameter in a command line.
;  441          0528  1 !
;  442          0529  1 ! Inputs:
;  443          0530  1 !
;  444          0531  1 !       FAB = FAB address.  FAB$L_NAM must point to a valid, initialized
;  445          0532  1 !                     NAM block with both expanded and resultant string
;  446          0533  1 !                     buffers present.
;  447          0534  1 !       SUCCESS_RTN = file success action routine address
;  448          0535  1 !                     The success routine is called with one argument,
;  449          0536  1 !                     which is a pointer to the FAB.
;  450          0537  1 !       ERROR_RTN  = error action routine address
;  451          0538  1 !                     The error routine is called with one argument,
;  452          0539  1 !                     which is a pointer to the FAB.
;  453          0540  1 !       CONTEXT = [OPTIONAL] address of longword used for keeping context
;  454          0541  1 !                     for multiple input file related file processing.
;  455          0542  1 !                     The longword should be zeroed on the first call,
;  456          0543  1 !                     and LIB$FILE_SCAN_END should be called after each
;  457          0544  1 !                     set (command line) has been processed to deallocate
;  458          0545  1 !                     the allocated context.
;  459          0546  1 !
;  460          0547  1 ! Implicit inputs:
;  461          0548  1 !
;  462          0549  1 !       The FAB must be initialized as a FAB with a pointer to a valid
;  463          0550  1 !       NAM block.
;  464          0551  1 !
;  465          0552  1 ! Outputs:
;  466          0553  1 !
;  467          0554  1 !       The action routines are called appropriately.  This
;  468          0555  1 !       routine returns when there are no more files.
;  469          0556  1 !
;  470          0557  1 ! Implicit outputs:
;  471          0558  1 !
;  472          0559  1 !
;  473          0560  1 ! Routine values:
;  474          0561  1 !
;  475          0562  1 !       Any valid RMS status code
;  476          0563  1 !
;  477          0564  1 !---
;  478          0565  2 BEGIN
;  479          0566  2
;  480          0567  2 GLOBAL BIND
;  481          0568  2         FMG$FILE_SCAN = LIB$FILE_SCAN;  ! Define old name
;  482          0569  2 LOCAL
;  483          0570  2         STATUS,                         ! Routine status
;  484          0571  2         SUC_ROUTINE,                    ! Address of success routine
;  485          0572  2         ERR_ROUTINE,                    ! Address of error routine
;  486          0573  2         CTX;                            ! Address of context longword
```

```
;  487          0574  2          NAM : REF $BBLOCK,                  ! NAM block address
;  488          0575  2          TNAM : REF $BBLOCK,                 ! Temporary NAM block pointer
;  489          0576  2          RNAM : REF $BBLOCK;                 ! Related file NAM block address
;  490          0577  2  MAP
;  491          0578  2          FAB:    REF BLOCK[,BYTE];           ! FAB structure address
;  492          0579  2  BUILTIN
;  493          0580  2          AP,CALLG,NULLPARAMETER;
;  494          0581  2
;  495        M 0582  2  MACRO CALL_SUCCESS =
;  496          0583  2          (CALLG(.AP,.SUC_ROUTINE))%;
;  497          0584  2
;  498        M 0585  2  MACRO CALL_ERROR =
;  499          0586  2          (CALLG(.AP,.ERR_ROUTINE))%;
;  500          0587  2  !
;  501          0588  2  ! Set up error and success routines
;  502          0589  2  !
;  503          0590  2  SUC_ROUTINE = DUMMY_ROUTINE;
;  504          0591  2  ERR_ROUTINE = .SUC_ROUTINE;
;  505          0592  2  IF NOT NULLPARAMETER(2)
;  506          0593  2  THEN
;  507          0594  2          SUC_ROUTINE = .SUCCESS_RTN;
;  508          0595  2  IF NOT NULLPARAMETER(3)
;  509          0596  2  THEN
;  510          0597  2          ERR_ROUTINE = .ERROR_RTN;
;  511          0598  2  !
;  512          0599  2  ! Tell RMS to save context over calls to speed things up.  This also
;  513          0600  2  ! causes directories to be read by RMS instead of the ACP.
;  514          0601  2  !
;  515          0602  2  NAM = .FAB[FAB$L_NAM];
;  516          0603  2  NAM[NAM$V_SVCTX] = 1;
;  517          0604  2  CTX = 0;
;  518          0605  2  !
;  519          0606  2  ! Set up previous file specifications NAM list pointer
;  520          0607  2  !
;  521          0608  2  IF NOT NULLPARAMETER(4)
;  522          0609  3  THEN    BEGIN
;  523          0610  3          CTX = .CONTEXT;                     !Get address of context longword
;  524          0611  3          NAM[NAM$L_RLF] = ..CTX;             !Set related nam block pointer
;  525          0612  2          END;
;  526          0613  2  !
;  527          0614  2  ! Parse the file spec
;  528          0615  2  !
;  529          0616  2  FAB[FAB$V_NAM] = 0;                         !Clear in case previously set
;  530          0617  2  STATUS = $PARSE(FAB = .FAB);
;  531          0618  2  IF NOT .STATUS
;  532          0619  3  THEN    BEGIN
;  533          0620  3          COPY_ESL_TO_RSL(.FAB,.NAM);
;  534          0621  3          CALL_ERROR;
;  535          0622  3          COPY_FILE_STRING(.CTX,.FAB);
;  536          0623  3          RETURN .STATUS;
;  537          0624  2          END;
;  538          0625  2  FAB[FAB$V_NAM] = 1;                         ! Use NAM block
;  539          0626  2  !
;  540          0627  2  ! Copy the default file string to the end of the nam block list
;  541          0628  2  ! if we have a context block.
;  542          0629  2  !
;  543          0630  3  IF (.CTX NEQ 0)
```

```
;  544          0631  3 THEN IF (..CTX EQL 0)
;  545          0632  2 THEN
;  546          0633  2         MOVE_DEFAULT_STRING(.CTX,.FAB);
;  547          0634  2 !
;  548          0635  2 ! Handle the case of being called with a related NAM block, but not
;  549          0636  2 ! the context block.  In this case, we save the expanded filename
;  550          0637  2 ! string.  This will provide the functionality seen in V4FT1.
;  551          0638  2 !
;  552          0639  2 RNAM = .NAM[NAM$L_RLF];
;  553          0640  3 IF (.NAM[NAM$B_ES] NEQ 0)
;  554          0641  3 AND (.RNAM NEQ 0)
;  555          0642  3 AND (.CTX EQL 0)
;  556          0643  3 THEN    BEGIN
;  557          0644  3         LOCAL
;  558          0645  3                 STATUS_1;
;  559          0646  3
;  560          0647  3         IF .RNAM[NAM$B_RSL] NEQ 0       !Deallocate any previous
;  561          0648  3         THEN
;  562          0649  3                 LIB$FREE_VM(%REF(.RNAM[NAM$B_RSL]),RNAM[NAM$L_RSA]);
;  563          0650  3         RNAM[NAM$B_RSL] = .NAM[NAM$B_ESL];
;  564          0651  3         STATUS_1 = LIB$GET_VM(%REF(.RNAM[NAM$B_RSL]),RNAM[NAM$L_RSA]);
;  565          0652  3         IF NOT .STATUS_1
;  566          0653  3         THEN
;  567          0654  3                 RETURN .STATUS_1;
;  568          0655  3         CH$MOVE(.RNAM[NAM$B_RS],.NAM[NAM$L_ESA],.RNAM[NAM$L_RSA]);
;  569          0656  2         END;
;  570          0657  2 FAB[FAB$B_DNS] = 0;                     ! Clear default name string
;  571          0658  2 !
;  572          0659  2 ! If a wildcard version number was specified on this filespec
;  573          0660  2 ! (via either FNM or DNM), then leave dnm set to ';*' so that
;  574          0661  2 ! the version will be sticky.  This is because RMS does not copy
;  575          0662  2 ! the version field from related file name string.
;  576          0663  2 !
;  577          0664  2 IF .NAM[NAM$V_WILD_VER]
;  578          0665  3 THEN    BEGIN
;  579          0666  3         FAB[FAB$B_DNS] = %CHARCOUNT(';*');
;  580          0667  3         FAB[FAB$L_DNA] = WILD_VER;
;  581          0668  2         END;
;  582          0669  2 !
;  583          0670  2 ! If the device is non-directory structured, then simply return
;  584          0671  2 ! to the caller's success action routine with the spec and
;  585          0672  2 ! avoid the SEARCH sequence. Also avoid the SEARCH sequence if
;  586          0673  2 ! the file is a PPF file.
;  587          0674  2 !
;  588          0675  2 IF NOT .(FAB[FAB$L_DEV])<$BITPOSITION(DEV$V_DIR),1>
;  589          0676  2 AND NOT .NAM[NAM$V_NODE]
;  590          0677  2 OR .(FAB[FAB$L_DEV])<$BITPOSITION(DEV$V_FOR),1>
;  591          0678  2 OR .NAM[NAM$V_PPF]
;  592          0679  3 THEN    BEGIN
;  593          0680  3         COPY_ESL_TO_RSL(.FAB,.NAM);
;  594          0681  3         CALL_SUCCESS;
;  595          0682  3         COPY_FILE_STRING(.CTX,.FAB);
;  596          0683  3         RETURN .STATUS;
;  597          0684  2         END;
;  598          0685  2 !
;  599          0686  2 ! If the file specification is non-wild, then SEARCH once to get
;  600          0687  2 ! the FID/DID filled in and do not repeat the search.
```

```
;   601          0688  2 ! If no wildcard in a network spec, no need for search.
;   602          0689  2 !
;   603          0690  2 IF NOT .NAM[NAM$V_WILDCARD]
;   604          0691  3 THEN     BEGIN
;   605          0692  3          IF NOT .NAM[NAM$V_NODE]
;   606          0693  4          THEN     BEGIN
;   607          0694  4                   STATUS = $SEARCH(FAB = .FAB);
;   608          0695  4                   IF NOT .STATUS
;   609          0696  5                   THEN     BEGIN
;   610          0697  5                            COPY_ESL_TO_RSL(.FAB,.NAM);
;   611          0698  5                            CALL_ERROR;
;   612          0699  5                            COPY_FILE_STRING(.CTX,.FAB);
;   613          0700  5                            RETURN .STATUS;
;   614          0701  4                            END;
;   615          0702  4                   END
;   616          0703  3          ELSE COPY_ESL_TO_RSL(.FAB,.NAM);
;   617          0704  3          CALL_SUCCESS;
;   618          0705  3          COPY_FILE_STRING(.CTX,.FAB);
;   619          0706  3          RETURN .STATUS;
;   620          0707  2          END;
;   621          0708  2 !
;   622          0709  2 ! Search for the each file which matches the wildcard sequence.  If
;   623          0710  2 ! success call success action routine and continue.  If no more files,
;   624          0711  2 ! quit.  If other error, call the error action routine and if not
;   625          0712  2 ! a wildcard directory or failure wasn't no privilege, then quit.
;   626          0713  2 !
;   627          0714  3 DO       BEGIN
;   628          0715  3          STATUS = $SEARCH(FAB = .FAB);
;   629          0716  3          IF .STATUS
;   630          0717  4          THEN     CALL_SUCCESS
;   631          0718  4          ELSE     BEGIN
;   632          0719  4                   IF .STATUS EQLU .RMSNMF
;   633          0720  5                   THEN     BEGIN
;   634          0721  5                            COPY_FILE_STRING(.CTX,.FAB);
;   635          0722  5                            RETURN .STATUS
;   636          0723  5                            END
;   637          0724  4                   ELSE
;   638          0725  5                            BEGIN
;   639          0726  5                            COPY_ESL_TO_RSL(.FAB,.NAM);
;   640          0727  5                            CALL_ERROR;
;   641          0728  5                            !
;   642          0729  5                            ! Quit if not a wildcard directory or system status
;   643          0730  5                            ! not NOPRIV.
;   644          0731  5                            !
;   645          0732  5                            IF NOT .NAM[NAM$V_WILD_DIR]
;   646          0733  5                             OR .FAB[FAB$L_STV] NEQU SS$_NOPRIV
;   647          0734  6                             THEN     BEGIN
;   648          0735  6                                      COPY_FILE_STRING(.CTX,.FAB);
;   649          0736  6                                      RETURN .STATUS;
;   650          0737  5                                      END;
;   651          0738  5                            IF .FAB[FAB$L_STV] EQL SS$_NOPRIV
;   652          0739  5                             THEN STATUS = 1;
;   653          0740  4                            END;
;   654          0741  3                   END;
;   655          0742  3
;   656          0743  3          END
;   657          0744  2 UNTIL NOT .STATUS;
```

```
;  658          0745  2 COPY_FILE_STRING(.CTX,.FAB);
;  659          0746  2 RETURN .STATUS
;  660          0747  1 END;


                                                    .EXTRN  SYS$SEARCH

                                        0FFC 00000          .ENTRY  LIB$FILE_SCAN, Save R2,R3,R4,R5,R6,R7,R8,-      ; 0518
                                                                    R9,R10,R11
                                    5E          04  C2 00002        SUBL2   #4, SP
                                    5A      FF07    CF  9E 00005    MOVAB   DUMMY_ROUTINE, SUC_ROUTINE              ; 0590
                                    5B              5A  D0 0000A    MOVL    SUC_ROUTINE, ERR_ROUTINE                ; 0591
                                    02              6C  91 0000D    CMPB    (AP), #2                                ; 0592
                                                    09  1F 00010    BLSSU   1$
                                                08  AC  D5 00012    TSTL    8(AP)
                                                    04  13 00015    BEQL    1$
                                    5A          08  AC  D0 00017    MOVL    SUCCESS_RTN, SUC_ROUTINE                ; 0594
                                    03              6C  91 0001B 1$:    CMPB    (AP), #3                            ; 0595
                                                    09  1F 0001E    BLSSU   2$
                                                0C  AC  D5 00020    TSTL    12(AP)
                                                    04  13 00023    BEQL    2$
                                    5B          0C  AC  D0 00025    MOVL    ERROR_RTN, ERR_ROUTINE                  ; 0597
                                    52          04  AC  D0 00029 2$:    MOVL    FAB, R2                             ; 0602
                                    56          28  A2  D0 0002D    MOVL    40(R2), NAM
                                33  A6          80  8F  88 00031    BISB2   #128, 51(NAM)                           ; 0603
                                                    58  D4 00036    CLRL    CTX                                     ; 0604
                                    04              6C  91 00038    CMPB    (AP), #4                                ; 0608
                                                    0D  1F 0003B    BLSSU   3$
                                                10  AC  D5 0003D    TSTL    16(AP)
                                                    08  13 00040    BEQL    3$
                                        58          10  AC  D0 00042    MOVL    CONTEXT, CTX                        ; 0610
                                    10  A6              68  D0 00046    MOVL    (CTX), 16(NAM)                      ; 0611
                                    07  A2              01  8A 0004A 3$:    BICB2   #1, 7(R2)                       ; 0616
                                                        52  DD 0004E    PUSHL   R2                                  ; 0617
                            00000000G   00              01  FB 00050    CALLS   #1, SYS$PARSE
                                        59              50  D0 00057    MOVL    R0, STATUS
                                        0F              59  E8 0005A    BLBS    STATUS, 5$                          ; 0618
                                                0044    8F  BB 0005D    PUSHR   #^M<R2,R6>                          ; 0620
                                    FE65    CF              02  FB 00061 4$:    CALLS   #2, COPY_ESL_TO_RSL
                                            6B              6C  FA 00066    CALLG   (AP), (ERR_ROUTINE)
                                                        010A    31  00069    BRW     20$                             ; 0622
                                        57          04  AC  D0 0006C 5$:    MOVL    FAB, R7                         ; 0625
                                    07  A7              01  88 00070    BISB2   #1, 7(R7)
                                                        58  D5 00074    TSTL    CTX                                 ; 0630
                                                        0D  13 00076    BEQL    6$
                                                        68  D5 00078    TSTL    (CTX)                               ; 0631
                                                        09  12 0007A    BNEQ    6$
                                                        57  DD 0007C    PUSHL   R7                                  ; 0633
                                                        58  DD 0007E    PUSHL   CTX
                                    FEBF    CF              02  FB 00080    CALLS   #2, MOVE_DEFAULT_STRING
                                            52          10  A6  D0 00085 6$:    MOVL    16(NAM), RNAM               ; 0639
                                                    0B  A6  95  00089    TSTB    11(NAM)                             ; 0640
                                                        44  13 0008C    BEQL    9$
                                                        52  D5 0008E    TSTL    RNAM                                ; 0641
                                                        40  13 00090    BEQL    9$
                                                        58  D5 00092    TSTL    CTX                                 ; 0642
```

```
                                3C  12 00094          BNEQ    9$                              ;
                          03    A2  95 00096          TSTB    3(RNAM)                         ; 0647
                          12    13 00099              BEQL    7$                              ;
                          04    A2  9F 0009B          PUSHAB  4(RNAM)                         ; 0649
               04    AE   03    A2  9A 0009E          MOVZBL  3(RNAM), 4(SP)                  ;
                          04    AE  9F 000A3          PUSHAB  4(SP)                           ;
      00000000G  00             02  FB 000A6          CALLS   #2, LIB$FREE_VM                 ;
             03  A2     0B      A6  90 000AD 7$:      MOVB    11(NAM), 3(RNAM)                ; 0650
                          04    A2  9F 000B2          PUSHAB  4(RNAM)                         ; 0651
               04    AE   03    A2  9A 000B5          MOVZBL  3(RNAM), 4(SP)                  ;
                          04    AE  9F 000BA          PUSHAB  4(SP)                           ;
      00000000G  00             02  FB 000BD          CALLS   #2, LIB$GET_VM                  ;
                 01             50  E8 000C4          BLBS    STATUS_1, 8$                    ; 0652
                                    04 000C7          RET                                     ;
                 50       03    A2  9A 000C8 8$:      MOVZBL  3(RNAM), R0                     ; 0655
   04   B2    0C  B6            50  28 000CC          MOVC3   R0, @12(NAM), @4(RNAM)          ;
                          35    A7  94 000D2 9$:      CLRB    53(R7)                          ; 0657
        0A    34  A6            03  E1 000D5          BBC     #3, 52(NAM), 10$                ; 0664
              35  A7            02  90 000DA          MOVB    #2, 53(R7)                      ; 0666
              30  A7      FD8A  CF  9E 000DE          MOVAB   WILD_VER, 48(R7)                ; 0667
        05    40  A7            03  E0 000E4 10$:     BBS     #3, 64(R7), 11$                 ; 0675
        08    36  A6            01  E1 000E9          BBC     #1, 54(NAM), 12$                ; 0676
                 04       43    A7  E8 000EE 11$:     BLBS    67(R7), 12$                     ; 0677
                 0B       36    A6  E9 000F2          BLBC    54(NAM), 14$                    ; 0678
                                56  DD 000F6 12$:     PUSHL   NAM                             ; 0680
                                57  DD 000F8          PUSHL   R7                              ;
           FDCC  CF             02  FB 000FA 13$:     CALLS   #2, COPY_ESL_TO_RSL             ;
                                28  11 000FF          BRB     16$                             ;
                 29       35    A6  E8 00101 14$:     BLBS    53(NAM), 17$                    ; 0690
        18    36  A6            01  E0 00105          BBS     #1, 54(NAM), 15$                ; 0692
                          04    AC  DD 0010A          PUSHL   FAB                             ; 0694
      00000000G  00             01  FB 0010D          CALLS   #1, SYS$SEARCH                  ;
                 59             50  D0 00114          MOVL    R0, STATUS                      ;
                 0F             59  E8 00117          BLBS    STATUS, 16$                     ; 0695
                                56  DD 0011A          PUSHL   NAM                             ; 0697
                          04    AC  DD 0011C          PUSHL   FAB                             ;
                              FF3F  31 0011F          BRW     4$                              ;
                                56  DD 00122 15$:     PUSHL   NAM                             ; 0703
                          04    AC  DD 00124          PUSHL   FAB                             ;
                                D1  11 00127          BRB     13$                             ;
                 6A             6C  FA 00129 16$:     CALLG   (AP), (SUC_ROUTINE)             ;
                                48  11 0012C          BRB     20$                             ; 0705
                          04    AC  DD 0012E 17$:     PUSHL   FAB                             ; 0715
      00000000G  00             01  FB 00131          CALLS   #1, SYS$SEARCH                  ;
                 59             50  D0 00138          MOVL    R0, STATUS                      ;
                 05             59  E9 0013B          BLBC    STATUS, 18$                     ; 0716
                 6A             6C  FA 0013E          CALLG   (AP), (SUC_ROUTINE)             ; 0717
                                30  11 00141          BRB     19$                             ;
           FD20  CF             59  D1 00143 18$:     CMPL    STATUS, RMS$_NMF                ; 0719
                                2C  13 00148          BEQL    20$                             ;
                                56  DD 0014A          PUSHL   NAM                             ; 0726
                          04    AC  DD 0014C          PUSHL   FAB                             ;
           FD77  CF             02  FB 0014F          CALLS   #2, COPY_ESL_TO_RSL             ;
                 6B             6C  FA 00154          CALLG   (AP), (ERR_ROUTINE)             ;
        1A    36  A6            04  E1 00157          BBC     #4, 54(NAM), 20$                ; 0732
                 50       04    AC  D0 0015C          MOVL    FAB, R0                         ; 0733
                 24       0C    A0  D1 00160          CMPL    12(R0), #36                     ;
```

```
                    10  12 00164         BNEQ    20$                         ;
           50   04  AC  D0 00166         MOVL    FAB, R0                     ; 0738
           24   0C  A0  D1 0016A         CMPL    12(R0), #36                 ;
                    03  12 0016E         BNEQ    19$                         ;
           59       01  D0 00170         MOVL    #1, STATUS                  ; 0739
           B8       59  E8 00173 19$:    BLBS    STATUS, 17$                 ; 0744
                04  AC  DD 00176 20$:    PUSHL   FAB                         ; 0745
                    58  DD 00179         PUSHL   CTX                         ;
     FCF0  CF       02  FB 0017B         CALLS   #2, COPY_FILE_STRING        ;
           50       59  D0 00180         MOVL    STATUS, R0                  ; 0746
                        04 00183         RET                                 ; 0747

; Routine Size:  388 bytes,    Routine Base:  _LIB$CODE + 0198
```

```
;  662        0748  1 %SBTTL 'COPY_RESULT_NAME Copy best name possible to result string';
;  663        0749  1 ROUTINE COPY_RESULT_NAME (FAB,RESULT_NAME) : NOVALUE =
;  664        0750  1 !---
;  665        0751  1 !
;  666        0752  1 !       This routine extracts the best possible result name from the
;  667        0753  1 !       fab/nam block and returns it in the result descriptor.
;  668        0754  1 !
;  669        0755  1 ! Inputs:
;  670        0756  1 !
;  671        0757  1 !       fab     address of the fab, which must also contain a nam block
;  672        0758  1 !       result_name address of the descriptor for the result string
;  673        0759  1 !
;  674        0760  1 ! Outputs:
;  675        0761  1 !
;  676        0762  1 !       Output string is copied to result_name using lib$s_copy_r_dx
;  677        0763  1 !
;  678        0764  1 !---
;  679        0765  2 BEGIN
;  680        0766  2 MAP
;  681        0767  2     FAB : REF BLOCK[,BYTE];
;  682        0768  2
;  683        0769  2 BIND
;  684        0770  2     NAM = FAB[FAB$L_NAM] : REF BLOCK[,BYTE];
;  685        0771  2
;  686        0772  2 LOCAL
;  687        0773  2     FNSIZE,
;  688        0774  2     FNADDR;
;  689        0775  2
;  690        0776  2 IF (FNSIZE = .NAM[NAM$B_RSL]) NEQ 0
;  691        0777  2     THEN FNADDR = .NAM[NAM$L_RSA]
;  692        0778  2     ELSE IF (FNSIZE = .NAM[NAM$B_ESL]) NEQ 0
;  693        0779  2         THEN FNADDR = .NAM[NAM$L_ESA]
;  694        0780  3         ELSE BEGIN
;  695        0781  3             FNSIZE = .FAB[FAB$B_FNS];
;  696        0782  3             FNADDR = .FAB[FAB$L_FNA];
;  697        0783  2             END;
;  698        0784  2
;  699        0785  2 RETURN LIB$SCOPY_R_DX(FNSIZE,.FNADDR,.RESULT_NAME)
;  700        0786  1 END;
```

```
                                 0004 00000 COPY_RESULT_NAME:
                                                  .WORD   Save R2                   ; 0749
        51      04   AC   D0 00002                MOVL    FAB, R1                   ; 0770
        50      28   A1   D0 00006                MOVL    40(R1), R0                ; 0776
        7E      03   A0   9A 0000A                MOVZBL  3(R0), FNSIZE             ;
                06   13   0000E                   BEQL    1$                        ;
        52      04   A0   D0 00010                MOVL    4(R0), FNADDR             ; 0777
                14   11   00014                   BRB     3$                        ;
        6E      0B   A0   9A 00016 1$:            MOVZBL  11(R0), FNSIZE            ; 0778
                06   13   0001A                   BEQL    2$                        ;
        52      0C   A0   D0 0001C                MOVL    12(R0), FNADDR            ; 0779
                08   11   00020                   BRB     3$                        ;
        6E      34   A1   9A 00022 2$:            MOVZBL  52(R1), FNSIZE            ; 0781
```

```
                          52       2C   A1  D0 00026        MOVL    44(R1), FNADDR                        ; 0782
                                   08   AC  DD 0002A 3$:    PUSHL   RESULT_NAME                           ; 0785
                                        52  DD 0002D        PUSHL   FNADDR                                ;
                                   08   AE  9F 0002F        PUSHAB  FNSIZE                                ;
                 00000000G  00          03  FB 00032        CALLS   #3, LIB$SCOPY_R_DX                    ;
                                            04 00039        RET                                           ; 0786

; Routine Size:  58 bytes,    Routine Base:  _LIB$CODE + 031C
```

```
;  702          0787  1 %SBTTL  'FIND_FILE_CLEANUP Internal routine to do find_file cleanup';
;  703          0788  1 ROUTINE FIND_FILE_CLEANUP(CONTEXT) =
;  704          0789  1 !---
;  705          0790  1 !       Deallocate the context associated with using LIB$FIND_FILE
;  706          0791  1 !
;  707          0792  1 ! Inputs:
;  708          0793  1 !
;  709          0794  1 !       context = address of longword containing context pointer
;  710          0795  1 !
;  711          0796  1 ! Outputs:
;  712          0797  1 !
;  713          0798  1 !       A parse of the null string is done.
;  714          0799  1 !       Context, related nam blocks, etc, all deallocated.  Context
;  715          0800  1 !       longword is not zeroed.
;  716          0801  1 !---
;  717          0802  2 BEGIN
;  718          0803  2 MAP
;  719          0804  2         CONTEXT : REF VECTOR[,LONG];
;  720          0805  2
;  721          0806  2 BIND
;  722          0807  2         INTFLAGS = .CONTEXT[0] + INTFLAGS_OFF : BITVECTOR;
;  723          0808  2
;  724          0809  2 LOCAL
;  725          0810  2         FAB : REF $BBLOCK,
;  726          0811  2         NAM : REF $BBLOCK,
;  727          0812  2         RNAM : REF $BBLOCK,
;  728          0813  2         BLOCKSIZE;
;  729          0814  2
;  730          0815  2 FAB = .CONTEXT[0];
;  731          0816  2 !
;  732          0817  2 ! Deallocate the filename string and the context block
;  733          0818  2 !
;  734          0819  2 BLOCKSIZE = .FAB[FAB$B_FNS];
;  735          0820  2 IF .FAB[FAB$B_FNS] NEQ 0
;  736          0821  2     AND .FAB[FAB$L_FNA] NEQ 0
;  737          0822  2 THEN
;  738          0823  2         LIB$FREE_VM(BLOCKSIZE,FAB[FAB$L_FNA]);
;  739          0824  2 !
;  740          0825  2 ! If doing multiple input related file processing, deallocate the related
;  741          0826  2 ! nam blocks
;  742          0827  2 !
;  743          0828  2 IF .INTFLAGS[0]
;  744          0829  3 THEN     BEGIN
;  745          0830  3         NAM = .FAB[FAB$L_NAM];
;  746          0831  3         IF .NAM NEQ 0
;  747          0832  3         THEN
;  748          0833  3                 NAM = .NAM[NAM$L_RLF];
;  749          0834  3         WHILE .NAM NEQ 0
;  750          0835  4         DO       BEGIN
;  751          0836  4                  RNAM = .NAM[NAM$L_RLF];
;  752          0837  4                  LIB$FREE_VM(%REF(NAM$C_BLN+.NAM[NAM$B_RSL]),NAM);
;  753          0838  4                  NAM = .RNAM;
;  754          0839  3                  END;
;  755          0840  2         END;
;  756          0841  2 !
;  757          0842  2 ! Parse the null string
;  758          0843  2 !
```

```
;  759          0844  2 PARSE_NULL_STRING(.FAB);
;  760          0845  2 LIB$FREE_VM(%REF(CONTEXT_SIZE),FAB)·
;  761          0846  2 RETURN 1
;  762          0847  1 END;


                                        001C 00000 FIND_FILE_CLEANUP:
                                                   .WORD   Save R2,R3,R4                          ; 0788
                          54 00000000G   00  9E 00002      MOVAB   LIB$FREE_VM, R4
                          5E             10  C2 00009      SUBL2   #16, SP
                    53    04   BC 00000312   8F  C1 0000C      ADDL3   #786, @CONTEXT, R3          ; 0807
                          0C   AE       04   BC  D0 00015      MOVL    @CONTEXT, FAB               ; 0815
                          52   0C       AE   D0  D0 0001A      MOVL    FAB, R2                     ; 0819
                          04   AE       34   A2  9A 0001E      MOVZBL  52(R2), BLOCKSIZE
                                             0E  13 00023      BEQL    1$                          ; 0820
                                       2C    A2  D5 00025      TSTL    44(R2)                      ; 0821
                                             09  13 00028      BEQL    1$
                                       2C    A2  9F 0002A      PUSHAB  44(R2)                      ; 0823
                                       08    AE  9F 0002D      PUSHAB  BLOCKSIZE
                          64                   02  FB 00030      CALLS   #2, LIB$FREE_VM
                          36                   63  E9 00033 1$:  BLBC    (R3), 3$                  ; 0828
                          08   AE       28    A2  D0 00036      MOVL    40(R2), NAM               ; 0830
                          50           08    AE  D0 0003B      MOVL    NAM, R0                   ; 0831
                                             05  13 0003F      BEQL    2$
                          08   AE       10    A0  D0 00041      MOVL    16(R0), NAM               ; 0833
                          50           08    AE  D0 00046 2$:  MOVL    NAM, R0                   ; 0834
                                             20  13 0004A      BEQL    3$
                                53     10    A0  D0 0004C      MOVL    16(R0), RNAM              ; 0836
                                       08    AE  9F 00050      PUSHAB  NAM                       ; 0837
                             04   AE       03    A0  9A 00053      MOVZBL  3(R0), 4(SP)
                             04   AE 00000060   8F  C0 00058      ADDL2   #96, 4(SP)
                                       04    AE  9F 00060      PUSHAB  4(SP)
                                64                   02  FB 00063      CALLS   #2, LIB$FREE_VM
                             08   AE                 53  D0 00066      MOVL    RNAM, NAM         ; 0838
                                                     DA  11 0006A      BRB     2$                ; 0834
                                                     52  DD 0006C 3$:  PUSHL   R2                ; 0844
                           FCE5   CF                 01  FB 0006E      CALLS   #1, PARSE_NULL_STRING
                                       0C    AE  9F 00073      PUSHAB  FAB                       ; 0845
                             04   AE      031A   8F  3C 00076      MOVZWL  #794, 4(SP)
                                       04    AE  9F 0007C      PUSHAB  4(SP)
                                64                   02  FB 0007F      CALLS   #2, LIB$FREE_VM
                                50                   01  D0 00082      MOVL    #1, R0            ; 0846
                                                         04 00085      RET                       ; 0847

; Routine Size:  134 bytes,    Routine Base:  _LIB$CODE + 0356
```

```
;  764          0848  1 %SBTTL 'LIB$FIND_FILE Find a file given a file name';
;  765          0849  1 GLOBAL ROUTINE LIB$FIND_FILE(FILE_NAME,RESULT_NAME,CONTEXT,
;  766          0850  1                              DEFAULT_NAME,RELATED_NAME,STV_ADDR,USER_FLAGS) =
;  767          0851  1 !---
;  768          0852  1 !       This routine is called with a wildcard file specification, which
;  769          0853  1 !       it searches for, and returns the next resultant file spec.
;  770          0854  1 !
;  771          0855  1 ! Inputs:
;  772          0856  1 !
;  773          0857  1 !       FILE_NAME = File name descriptor address.
;  774          0858  1 !       RESULT_NAME = Result file name descriptor address.
;  775          0859  1 !       CONTEXT = Address of a longword containing previous call "context".
;  776          0860  1 !               = Zero if no previous call.
;  777          0861  1 !       DEFAULT_NAME = Default file name descriptor address (optional).
;  778          0862  1 !       RELATED_NAME = Related file name descriptor address (optional).
;  779          0863  1 !       STV_ADDR = [OPTIONAL] Address of longword to store STV on failing
;  780          0864  1 !               RMS operation
;  781          0865  1 !       USER_FLAGS = Address of longword of flags to control operation
;  782          0866  1 !               [OPTIONAL]
;  783          0867  1 !               BIT 0 (NOWILD) Return an error if a wildcard is input
;  784          0868  1 !               BIT 1 (MULTIPLE) Perform multiple input file stickyness.
;  785          0869  1 !                       In this mode, the RELATED_NAME argument is ignored.
;  786          0870  1 !                       Each time LIB$FIND_FILE is called with a different
;  787          0871  1 !                       file specification, the one from the previous call
;  788          0872  1 !                       is added to the list of related file specifications.
;  789          0873  1 !                       This allows parsing of commands such as
;  790          0874  1 !                               $ ENCRYPT FILE1.TYP,FILE*2.TYP,...
;  791          0875  1 !                       Use of this feature is required to get the desired
;  792          0876  1 !                       defaulting with searchlists.
;  793          0877  1 !
;  794          0878  1 !                       Note that LIB$FIND_FILE_END must be called between
;  795          0879  1 !                       each command line in interactive use or the defaults
;  796          0880  1 !                       from the previous command line will affect the
;  797          0881  1 !                       next command line.
;  798          0882  1 !
;  799          0883  1 ! Implicit inputs:
;  800          0884  1 !
;  801          0885  1 !       CONTEXT is either 0 or as set up from a previous call to
;  802          0886  1 !       LIB$FIND_FILE.
;  803          0887  1 !
;  804          0888  1 ! Outputs:
;  805          0889  1 !
;  806          0890  1 !       CONTEXT = Address of internal FAB/NAM buffer.
;  807          0891  1 !       RESULT_NAME = Result file name.
;  808          0892  1 !
;  809          0893  1 ! Implicit outputs:
;  810          0894  1 !
;  811          0895  1 !       CONTEXT will point to a FAB/NAM block
;  812          0896  1 !
;  813          0897  1 ! Routine values:
;  814          0898  1 !
;  815          0899  1 !       Any valid RMS error code
;  816          0900  1 !       Error codes returned by LIB$GET_VM
;  817          0901  1 !       Error codes returned by LIB$SCOPY_R_DX
;  818          0902  1 !       SHR$_NOWILD with LIB facility code = Wildcard specification parsed
;  819          0903  1 !               and the NOWILD flag bit was set.
;  820          0904  1 !
```

```
;  821          0905  1 !---
;  822          0906  2 BEGIN
;  823          0907  2
;  824          0908  2 BUILTIN
;  825          0909  2         NULLPARAMETER;
;  826          0910  2
;  827          0911  2 LOCAL
;  828          0912  2         STATUS,                         ! Status
;  829          0913  2         STATUS_0,
;  830          0914  2         STATUS_1,
;  831          0915  2         STATUS_2,
;  832          0916  2         BLOCKSIZE,                      ! Size of request to lib$get/free vm
;  833          0917  2         FLAGS : BITVECTOR[32],          ! User flags
;  834          0918  2         INTFLAGS : REF BITVECTOR,       ! Internal flags
;  835          0919  2         STVADDR : REF VECTOR[,LONG],    ! Address of user's stv address
;  836          0920  2         FNBUF : REF VECTOR[,BYTE],      ! FAB/NAM buffer address
;  837          0921  2         FNBUF_SIZ,                      ! FAB/NAM buffer length
;  838          0922  2         FILE_SIZE,                      ! Length of FILE_NAME string
;  839          0923  2         FILE_ADDR,                      ! Address of FILE_NAME string
;  840          0924  2         DEFAULT_SIZE,                   ! Length of DEFAULT_NAME string
;  841          0925  2         DEFAULT_ADDR,                   ! Address of DEFAULT_NAME string
;  842          0926  2         RELATED_SIZE,                   ! Length of RELATED_NAME string
;  843          0927  2         RELATED_ADDR,                   ! Address of RELATED_NAME string
;  844          0928  2         FAB : REF $BBLOCK,              ! FAB address
;  845          0929  2         NAM : REF $BBLOCK,              ! NAM address
;  846          0930  2         RNAM : REF $BBLOCK,             ! Related NAM address
;  847          0931  2         NEXT_STATUS : REF VECTOR[,LONG];! Status to return next call
;  848          0932  2 MAP
;  849          0933  2         CONTEXT:        REF VECTOR[,LONG],      ! Pointer to context block
;  850          0934  2         FILE_NAME:      REF BLOCK[,BYTE],       ! File name string descriptor
;  851          0935  2         RESULT_NAME:    REF BLOCK[,BYTE],       ! Result name buffer descriptor
;  852          0936  2         DEFAULT_NAME:   REF BLOCK[,BYTE],       ! Default name descriptor
;  853          0937  2         RELATED_NAME:   REF BLOCK[,BYTE];       ! Related file name string desc
;  854          0938  2
;  855          0939  2 STATUS = 1;                               ! Preset success
;  856          0940  2 FILE_SIZE = RELATED_SIZE = DEFAULT_SIZE = 0;     ! Preset since they are words
;  857          0941  2 STVADDR = 0;
;  858          0942  2 IF NOT NULLPARAMETER(6)
;  859          0943  2 THEN
;  860          0944  2         STVADDR = .STV_ADDR;
;  861          0945  2 FLAGS = 0;
;  862          0946  2 IF NOT NULLPARAMETER(7)
;  863          0947  2 THEN
;  864          0948  2         FLAGS = ..USER_FLAGS;
;  865          0949  2 !
;  866          0950  2 ! If the specified previous "context" is zero, then there was no
;  867          0951  2 ! previous call, so the FAB/NAM block buffer needs to be allocated.
;  868          0952  2 !
;  869          0953  2 IF .CONTEXT[0] EQL 0
;  870          0954  3 THEN    BEGIN
;  871          0955  3         STATUS_0 = LIB$GET_VM(%REF(CONTEXT_SIZE),CONTEXT[0]);
;  872          0956  3         IF NOT .STATUS_0
;  873          0957  3         THEN
;  874          0958  3                 RETURN .STATUS_0;
;  875          0959  3         FNBUF = .CONTEXT[0];
;  876          0960  3         CH$FILL(0,CONTEXT_SIZE,.FNBUF);
;  877          0961  3       !
```

```
;  878          0962  3          ! Initialize the FAB and NAM blocks
;  879          0963  3          !
;  880        P 0964  3          $FAB_INIT(       FAB = .FNBUF,
;  881        P 0965  3                           FOP = NAM,
;  882          0966  3                           NAM = FNBUF[NAM_OFF]);
;  883        P 0967  3          $NAM_INIT(       NAM = FNBUF[NAM_OFF],
;  884        P 0968  3                           RLF = (IF .FLAGS[1] THEN 0
;  885        P 0969  3                                  ELSE FNBUF[RNAM_OFF]),
;  886        P 0970  3                           RSS = NAM$C_MAXRSS,
;  887        P 0971  3                           RSA = FNBUF[RSBUF_OFF],
;  888        P 0972  3                           ESS = NAM$C_MAXRSS,
;  889          0973  3                           ESA = FNBUF[ESBUF_OFF]);
;  890          0974  3          $NAM_INIT(       NAM = FNBUF[RNAM_OFF]);
;  891          0975  3          (.FNBUF + STATUS_OFF) = 1;
;  892          0976  3          END
;  893          0977  2 ELSE
;  894          0978  2          FNBUF = .CONTEXT[0];
;  895          0979  2 !
;  896          0980  2 ! Get the block addresses and check the validity of the FAB/NAM buffer.
;  897          0981  2 !
;  898          0982  2 FAB = .FNBUF;
;  899          0983  2 NAM = FNBUF[NAM_OFF];
;  900          0984  2 RNAM = FNBUF[RNAM_OFF];
;  901          0985  2 NEXT_STATUS = FNBUF[STATUS_OFF];
;  902          0986  2 INTFLAGS = FNBUF[INTFLAGS_OFF];
;  903          0987  2 IF .FAB[FAB$B_BID] NEQ FAB$C_BID
;  904          0988  2 OR .FAB[FAB$B_BLN] NEQ FAB$C_BLN
;  905          0989  2 THEN
;  906          0990  2          RETURN RMS$_FAB;
;  907          0991  2
;  908          0992  2 !
;  909          0993  2 ! Remember in context if doing multiple related filename processing
;  910          0994  2 !
;  911          0995  2 INTFLAGS[0] = .FLAGS[1];
;  912          0996  2 !
;  913          0997  2 ! Get the length and address of the filename string
;  914          0998  2 !
;  915          0999  3 IF NOT (STATUS_1 = LIB$ANALYZE_SDESC_R2(.FILE_NAME;FILE_SIZE,FILE_ADDR))
;  916          1000  2 THEN
;  917          1001  2          RETURN .STATUS_1;
;  918          1002  2
;  919          1003  2 !
;  920          1004  2 ! If specified, get the length and address of the default filename string
;  921          1005  2 !
;  922          1006  2 DEFAULT_ADDR = DEFAULT_SIZE;
;  923          1007  2 IF NOT NULLPARAMETER(4)
;  924          1008  2 THEN
;  925          1009  2          !
;  926          1010  2          ! Analyze default name descriptor if present
;  927          1011  2          !
;  928          1012  3          IF NOT (STATUS = LIB$ANALYZE_SDESC_R2(.DEFAULT_NAME;
;  929          1013  3                                                DEFAULT_SIZE,DEFAULT_ADDR))
;  930          1014  3          THEN BEGIN
;  931          1015  3                  COPY_RESULT_NAME(.FAB,,RESULT_NAME);
;  932          1016  3                  NEXT_STATUS[0] = .RMSNMF;       ! Require new FILE_NAME
;  933          1017  3                  RETURN .STATUS;
;  934          1018  2                  END;
```

```
;  935          1019  2 !                                                                                    ;
;  936          1020  2 ! If specified, get the length and address of the related file spec
;  937          1021  2 !
;  938          1022  2 RELATED_ADDR = RELATED_SIZE;
;  939          1023  2 IF NOT .FLAGS[1]
;  940          1024  2 AND NOT NULLPARAMETER(5)
;  941          1025  2 THEN
;  942          1026  3         IF NOT (STATUS = LIB$ANALYZE_SDESC_R2(.RELATED_NAME;
;  943          1027  3                                               RELATED_SIZE,RELATED_ADDR))
;  944          1028  3         THEN BEGIN
;  945          1029  3                 COPY_RESULT_NAME(.FAB,.RESULT_NAME);
;  946          1030  3                 NEXT_STATUS[0] = .RMSNMF;       ! Require new FILE_NAME
;  947          1031  3                 RETURN .STATUS;
;  948          1032  2                 END;
;  949          1033  2
;  950          1034  2 !
;  951          1035  2 ! If the specified file-name does not match the previous file-name,
;  952          1036  2 ! or if NOWILD, then set up the new filenames and parse them.
;  953          1037  2 ! (Also check for first call and file-name of all blanks)
;  954          1038  2 !
;  955          1039  2 IF .FLAGS[0]
;  956          1040  2     OR .INTFLAGS[1]
;  957          1041  2     OR CH$NEQ(.FAB[FAB$B_FNS],.FAB[FAB$L_FNA],
;  958          1042  2               .FILE_SIZE,.FILE_ADDR,' ')                                           ;
;  959          1043  2     OR CH$FAIL(CH$FIND_NOT_CH(.FILE_SIZE,.FILE_ADDR,' '))
;  960          1044  3     OR (
;  961          1045  3         BIND                                                                         ;
;  962          1046  3                 DNAM = FNBUF[DNAM_PTR] : REF $BBLOCK;
;  963          1047  3
;  964          1048  4         IF (.DNAM EQL 0)
;  965          1049  4                 OR (.DEFAULT_SIZE EQL 0)
;  966          1050  3         THEN
;  967          1051  3                 0
;  968          1052  3         ELSE
;  969          1053  3                 NOT CH$EQL(.DEFAULT_SIZE,.DEFAULT_ADDR,                              ;
;  970          1054  3                         .DNAM[NAM$B_RSL],.DNAM[NAM$L_RSA],0)                         ;
;  971          1055  3         )                                                                            ;
;  972          1056  3 THEN     BEGIN                                                                        ;
;  973          1057  3         BIND                                                                         ;
;  974          1058  3                 DNAM = FNBUF[DNAM_PTR] : REF $BBLOCK;                                ;
;  975          1059  3         !                                                                            ;
;  976          1060  3         ! If specified, set the default name.                                        ;
;  977          1061  3         !                                                                            ;
;  978          1062  5         IF ((.DEFAULT_SIZE NEQ 0)                                                    ;
;  979          1063  4                 AND (.(FNBUF[DNAM_PTR])<0,32,0> EQL 0))                              ;
;  980          1064  4                 OR (IF .(FNBUF[DNAM_PTR])<0,32,0> NEQ 0                              ;
;  981          1065  4                         THEN NOT CH$EQL(.DEFAULT_SIZE,.DEFAULT_ADDR,                 ;
;  982          1066  4                                         .DNAM[NAM$B_RSL],.DNAM[NAM$L_RSA],0)         ;
;  983          1067  4                         ELSE 0)                                                      ;
;  984          1068  4         THEN    BEGIN                                                                ;
;  985          1069  4                 FAB[FAB$B_DNS] = .DEFAULT_SIZE;                                      ;
;  986          1070  4                 FAB[FAB$L_DNA] = .DEFAULT_ADDR;                                      ;
;  987          1071  4                 END                                                                  ;
;  988          1072  3         ELSE                                                                         ;
;  989          1073  3                 FAB[FAB$B_DNS] = 0;                                                  ;
;  990          1074  3         !                                                                            ;
;  991          1075  3         ! If there is a previous name string, then delete it.  Then                  ;
```

```
;  992          1076  3          ! allocate space for new filename string.
;  993          1077  3          !
;  994          1078  3          IF (BLOCKSIZE = .FAB[FAB$B_FNS]) NEQ 0
;  995          1079  4          THEN    BEGIN
;  996          1080  4                  IF .FLAGS[1]
;  997          1081  5                  THEN    BEGIN
;  998          1082  5                          COPY_FILE_STRING(NAM[NAM$L_RLF],.FAB);
;  999          1083  4                          END;
; 1000          1084  4                  LIB$FREE_VM(BLOCKSIZE,FAB[FAB$L_FNA]);
; 1001          1085  4                  FAB[FAB$B_FNS] = 0;
; 1002          1086  3                  END;
; 1003          1087  3          BLOCKSIZE = .FILE_SIZE;
; 1004          1088  3          FAB[FAB$B_FNS] = .BLOCKSIZE;
; 1005          1089  3          IF .BLOCKSIZE NEQ 0
; 1006          1090  3          THEN
; 1007          1091  4                  BEGIN
; 1008          1092  5                  IF NOT (STATUS_2 = LIB$GET_VM(BLOCKSIZE,FAB[FAB$L_FNA]))
; 1009          1093  4                  THEN
; 1010          1094  4                      RETURN .STATUS_2;
; 1011          1095  4                  CH$MOVE(.FAB[FAB$B_FNS],.FILE_ADDR,.FAB[FAB$L_FNA]);
; 1012          1096  3                  END;
; 1013          1097  3          !
; 1014          1098  3          ! If specified, set the related default name.
; 1015          1099  3          !
; 1016          1100  3          IF NOT .FLAGS[1]
; 1017          1101  4          THEN    BEGIN
; 1018          1102  4                  IF .RELATED_SIZE NEQ 0
; 1019          1103  5                  THEN    BEGIN
; 1020          1104  5                          RNAM[NAM$B_RSL] = .RELATED_SIZE;
; 1021          1105  5                          RNAM[NAM$L_RSA] = .RELATED_ADDR;
; 1022          1106  5                          END
; 1023          1107  4                  ELSE
; 1024          1108  4                          RNAM[NAM$B_RSL] = 0;
; 1025          1109  3                  END;
; 1026          1110  3
; 1027          1111  3          !
; 1028          1112  3          ! Parse the file-spec.
; 1029          1113  3          !
; 1030          1114  3          INTFLAGS[1] = 0;
; 1031          1115  3          INTFLAGS[2] = 0;
; 1032          1116  3          NAM[NAM$V_SVCTX] = 1;                          ! Save RMS context
; 1033          1117  3          STATUS = $PARSE(FAB = .FAB);
; 1034          1118  3          NEXT_STATUS[0] = .STATUS;                      ! Save status for next call
; 1035          1119  3          IF .STVADDR NEQ 0
; 1036          1120  3          THEN
; 1037          1121  3                  STVADDR[0] = .FAB[FAB$L_STV];
; 1038          1122  3          IF NOT .STATUS
; 1039          1123  3          THEN
; 1040          1124  4                  BEGIN
; 1041          1125  4                  COPY_RESULT_NAME(.FAB,.RESULT_NAME);
; 1042          1126  4                  NEXT_STATUS[0] = .RMSNMF;
; 1043          1127  4                  RETURN .STATUS;
; 1044          1128  3                  END;
; 1045          1129  2          END;
; 1046          1130  2 !
; 1047          1131  2 ! If error parsing, or from the last search (could have been RMS$_NMF
; 1048          1132  2 ! set because of no wildcarding) deallocate the context unless MULTIPLE.
```

```
;1049     1133  2 ! The case of a wildcard directory and SS$_NOPRIV is special cased to
;1050     1134  2 ! allow a search to continue even if a particular directory is not accessable.
;1051     1135  2 !
;1052     1136  2 IF .NEXT_STATUS[0] EQL .RMSNMF
;1053     1137  3 THEN    BEGIN
;1054     1138  3         IF NOT .FLAGS[1]
;1055     1139  4         THEN    BEGIN
;1056     1140  4                 FIND_FILE_CLEANUP(.CONTEXT);
;1057     1141  4                 CONTEXT[0] = 0;
;1058     1142  3                 END;
;1059     1143  3         INTFLAGS[1] = 1;
;1060     1144  3         RETURN .RMSNMF;
;1061     1145  2         END;
;1062     1146  2 !
;1063     1147  2 ! Copy the default file string to a nam block at the end of the
;1064     1148  2 ! list of nam blocks if we have not yet done so.  If we already
;1065     1149  2 ! have copied the default string, then just insert it into the
;1066     1150  2 ! list of nam blocks at the current location.
;1067     1151  2 !
;1068     1152  2 IF .FAB[FAB$B_DNS] NEQ 0
;1069     1153  2         AND NOT .INTFLAGS[2]
;1070     1154  3 THEN    BEGIN
;1071     1155  3         LOCAL
;1072     1156  3                 NFAB : $FAB_DECL;
;1073     1157  3
;1074     1158  3         BIND
;1075     1159  3                 DNAMPTR = FNBUF[DNAM_PTR] : VECTOR[,LONG];
;1076     1160  3
;1077     1161  3         !
;1078     1162  3         ! Setup a dummy fab for copy_file_string.  Point default
;1079     1163  3         ! name pointer in the context block to newly created default nam block
;1080     1164  3         !
;1081     1165  3         CH$MOVE(FAB$C_BLN,.FAB,NFAB);
;1082     1166  3         NFAB[FAB$B_FNS] = .FAB[FAB$B_DNS];
;1083     1167  3         NFAB[FAB$L_FNA] = .FAB[FAB$L_DNA];
;1084     1168  3         COPY_FILE_STRING(NAM[NAM$L_R[F],NFAB);
;1085     1169  3         DNAMPTR[0] = .NAM[NAM$L_RLF];
;1086     1170  2         END;
;1087     1171  2
;1088     1172  2 IF .NAM[NAM$V_WILD_VER]
;1089     1173  2         AND NOT .INTFLAGS[2]
;1090     1174  3 THEN    BEGIN
;1091     1175  3         INTFLAGS[2] = 1;
;1092     1176  3         FAB[FAB$B_DNS] = %CHARCOUNT(';*');
;1093     1177  3         FAB[FAB$L_DNA] = WILD_VER;
;1094     1178  2         END;
;1095     1179  2 !
;1096     1180  2 ! If the device is non-directory structured, or the file is a PPF file,
;1097     1181  2 ! then simply return to the caller and avoid the SEARCH sequence.
;1098     1182  2 !
;1099     1183  2 IF NOT .(FAB[FAB$L_DEV])<$BITPOSITION(DEV$V_DIR),1>
;1100     1184  2 AND NOT .NAM[NAM$V_NODE]
;1101     1185  2 OR .(FAB[FAB$L_DEV])<$BITPOSITION(DEV$V_FOR),1>
;1102     1186  2 OR .NAM[NAM$V_PPF]
;1103     1187  3 THEN    BEGIN
;1104     1188  3         NEXT_STATUS[0] = .RMSNMF;                       ! No more files on next call
;1105     1189  3         COPY_RESULT_NAME(.FAB,.RESULT_NAME);
```

```
;  1106          1190  3          RETURN .STATUS;
;  1107          1191  2          END;
;  1108          1192  2 !
;  1109          1193  2 ! If wildcard processing is not wanted, check for it and return an
;  1110          1194  2 ! error if so.
;  1111          1195  2 !
;  1112          1196  2 IF .FLAGS[0]
;  1113          1197  2 AND .NAM[NAM$V_WILDCARD]
;  1114          1198  3 THEN     BEGIN
;  1115          1199  3          NEXT_STATUS[0] = .RMSNMF;
;  1116          1200  3          COPY_RESULT_NAME(.FAB,.RESULT_NAME);
;  1117          1201  3          RETURN LIB$_NOWILD;
;  1118          1202  2          END;
;  1119          1203  2 !
;  1120          1204  2 ! Search for the next file, unless it is a non-wildcard remote file,
;  1121          1205  2 ! in which case, don't bother because it's so expensive.
;  1122          1206  2 !
;  1123          1207  3 IF NOT (.NAM[NAM$V_NODE] AND NOT .NAM[NAM$V_WILDCARD])
;  1124          1208  2 THEN
;  1125          1209  2          STATUS = $SEARCH(FAB = .FAB);
;  1126          1210  2 !
;  1127          1211  2 ! Return the STV in case of an error
;  1128          1212  2 !
;  1129          1213  2 IF NOT .STATUS
;  1130          1214  3 AND (.STVADDR NEQ 0)
;  1131          1215  2 THEN
;  1132          1216  2          STVADDR[0] = .FAB[FAB$L_STV];
;  1133          1217  2
;  1134          1218  2 !
;  1135          1219  2 ! If privilege violation and non-wildcard directory spec then
;  1136          1220  2 ! set to return no more files on next call.
;  1137          1221  2 !
;  1138          1222  2 IF NOT .STATUS
;  1139          1223  3 AND NOT (.NAM[NAM$V_WILD_DIR] AND (.FAB[FAB$L_STV] EQL SS$_NOPRIV))
;  1140          1224  3 THEN     BEGIN
;  1141          1225  3          NEXT_STATUS[0] = .RMSNMF;                    ! No more files on next call
;  1142          1226  2          END;
;  1143          1227  2 !
;  1144          1228  2 ! If the filespec is non-wildcarded, set status so next call will return
;  1145          1229  2 ! no more files.
;  1146          1230  2 !
;  1147          1231  2 IF NOT .NAM[NAM$V_WILDCARD]
;  1148          1232  2 THEN
;  1149          1233  3          BEGIN
;  1150          1234  3          NEXT_STATUS[0] = .RMSNMF;
;  1151          1235  2          END;
;  1152          1236  2 !
;  1153          1237  2 ! Return the result name.  If the result name isn't set, return the expanded
;  1154          1238  2 ! name.
;  1155          1239  2 !
;  1156          1240  2 COPY_RESULT_NAME(.FAB,.RESULT_NAME);
;  1157          1241  2 !
;  1158          1242  2 ! If no more files and not MULTIPLE, deallocate the FAB/NAM buffer
;  1159          1243  2 !
;  1160          1244  2 IF .STATUS EQL .RMSNMF
;  1161          1245  2 AND NOT .FLAGS[1]
;  1162          1246  3 THEN     BEGIN
```

```
; 1163      1247 3          FIND_FILE_CLEANUP(.CONTEXT);
; 1164      1248 3          CONTEXT[0] = 0;
; 1165      1249 2          END;
; 1166      1250 2
; 1167      1251 2 RETURN .STATUS
; 1168      1252 2
; 1169      1253 1 END;



                                              0FFC 00000         .ENTRY  LIB$FIND_FILE, Save R2,R3,R4,R5,R6,R7,R8,-  ; 0849
                                                                         R9,R10,R11                                  ;
                              5E      98   AE   9E 00002         MOVAB   -104(SP), SP                                ;
                                           01   DD 00006         PUSHL   #1                                          ; 0939
                                      10   AE   7C 00008         CLRQ    DEFAULT_SIZE                                ; 0940
                                           7E   7C 0000B         CLRQ    STVADDR                                     ; 0941
                              06           6C   91 0000D         CMPB    (AP), #6                                    ; 0942
                                           09   1F 00010         BLSSU   1$                                          ;
                                      18   AC   D5 00012         TSTL    24(AP)                                      ;
                                           04   13 00015         BEQL    1$                                          ;
                              6E      18   AC   D0 00017         MOVL    STV_ADDR, STVADDR                           ; 0944
                                      14   AE   D4 0001B 1$:     CLRL    FLAGS                                       ; 0945
                              07           6C   91 0001E         CMPB    (AP), #7                                    ; 0946
                                           0A   1F 00021         BLSSU   2$                                          ;
                                      1C   AC   D5 00023         TSTL    28(AP)                                      ;
                                           05   13 00026         BEQL    2$                                          ;
                         14   AE      1C   BC   D0 00028         MOVL    @USER_FLAGS, FLAGS                          ; 0948
                                      0C   BC   D5 0002D 2$:     TSTL    @CONTEXT                                    ; 0953
                                           03   13 00030         BEQL    3$                                          ;
                                         0094   31 00032         BRW     7$                                          ;
                                      0C   AC   DD 00035 3$:     PUSHL   CONTEXT                                     ; 0955
                         14   AE    031A   8F   3C 00038         MOVZWL  #794, 20(SP)                                ;
                                      14   AE   9F 0003E         PUSHAB  20(SP)                                      ;
                  00000000G   00           02   FB 00041         CALLS   #2, LIB$GET_VM                              ;
                              01           50   E8 00048         BLBS    STATUS_0, 4$                                ; 0956
                                                04 0004B         RET                                                 ;
                              56      0C   BC   D0 0004C 4$:     MOVL    @CONTEXT, FNBUF                             ; 0959
   031A   8F          00      6E           00   2C 00050         MOVC5   #0, (SP), #0, #794, (FNBUF)                 ; 0960
                                           66      00057                                                             ;
   0050   8F          00      6E           00   2C 00058         MOVC5   #0, (SP), #0, #80, (FNBUF)                  ; 0966
                                           66      0005F                                                             ;
                              66    5003   8F   B0 00060         MOVW    #20483, (FNBUF)                             ;
                         04   A6 01000000  8F   D0 00065         MOVL    #16777216, 4(FNBUF)                         ;
                         16   A6           02   90 0006D         MOVB    #2, 22(FNBUF)                               ;
                         1F   A6           02   90 00071         MOVB    #2, 31(FNBUF)                               ;
                              57      50   A6   9E 00075         MOVAB   80(R6), R7                                  ;
                         28   A6           57   D0 00079         MOVL    R7, 40(FNBUF)                               ;
   0060   8F          00      6E           00   2C 0007D         MOVC5   #0, (SP), #0, #96, (R7)                     ; 0973
                                           67      00084                                                             ;
                              67    6002   8F   B0 00085         MOVW    #24578, (R7)                                ;
                         02   A7           01   8E 0008A         MNEGB   #1, 2(R7)                                   ;
                         04   A7    020F   C6   9E 0008E         MOVAB   527(R6), 4(R7)                              ;
                         0A   A7           01   8E 00094         MNEGB   #1, 10(R7)                                  ;
                         0C   A7    0110   C6   9E 00098         MOVAB   272(R6), 12(R7)                             ;
                  04     14   AE           01   E1 0009E         BBC     #1, FLAGS, 5$                               ;
```

```
                                    58  D4 000A3        CLRL    R8                                      ;
                                    05  11 000A5        BRB     6$                                      ;
                    58      00B0    C6  9E 000A7 5$:    MOVAB   176(R6), R8                             ;
               10   A7              58  D0 000AC 6$:    MOVL    R8, 16(R7)                              ;
                    57      00B0    C6  9E 000B0        MOVAB   176(FNBUF), R7                          ; 0974
  0060  8F      00  6E              00  2C 000B5        MOVC5   #0, (SP), #0, #96, (R7)                 ;
                    67                    000BC                                                         ;
                    67      6002    8F  B0 000BD        MOVW    #24578, (R7)                            ;
            030E    C6              01  D0 000C2        MOVL    #1, 782(FNBUF)                          ; 0975
                                    04  11 000C7        BRB     8$                                      ; 0953
                    56        0C    BC  D0 000C9 7$:    MOVL    @CONTEXT, FNBUF                         ; 0978
                    5B              56  D0 000CD 8$:    MOVL    FNBUF, FAB                              ; 0982
                    57        50    A6  9E 000D0        MOVAB   80(R6), NAM                             ; 0983
                    58      00B0    C6  9E 000D4        MOVAB   176(R6), RNAM                           ; 0984
                    59      030E    C6  9E 000D9        MOVAB   782(R6), NEXT_STATUS                    ; 0985
                    5A      0312    C6  9E 000DE        MOVAB   786(R6), INTFLAGS                       ; 0986
                    03              6B  91 000E3        CMPB    (FAB), #3                               ; 0987
                                    07  12 000E6        BNEQ    9$                                      ;
               50   8F        01    AB  91 000E8        CMPB    1(FAB), #80                             ; 0988
                                    08  13 000ED        BEQL    10$                                     ;
                    50 0001850C     8F  D0 000EF 9$:    MOVL    #99596, R0                              ; 0990
                                    04     000F6        RET                                             ;
        50    14  AE    01          01  EF 000F7 10$:   EXTZV   #1, #1, FLAGS, R0                       ; 0995
        6A        01    00          50  F0 000FD        INSV    R0, #0, #1, (INTFLAGS)                  ;
                    50        04    AC  D0 00102        MOVL    FILE_NAME, R0                           ; 0999
                       00000000G    00  16 00106        JSB     LIB$ANALYZE_SDESC_R2                    ;
               04   AE              51  D0 0010C        MOVL    R1, 4(SP)                               ;
               0C   AE              52  D0 00110        MOVL    R2, 12(SP)                              ;
                    01              50  E8 00114        BLBS    STATUS_1, 11$                           ;
                                    04     00117        RET                                             ;
                    55        18    AE  9E 00118 11$:   MOVAB   DEFAULT_SIZE, DEFAULT_ADDR              ; 1006
                    04              6C  91 0011C        CMPB    (AP), #4                                ; 1007
                                    1E  1F 0011F        BLSSU   12$                                     ;
                              10    AC  D5 00121        TSTL    16(AP)                                  ;
                                    19  13 00124        BEQL    12$                                     ;
                    50        10    AC  D0 00126        MOVL    DEFAULT_NAME, R0                        ; 1012
                       00000000G    00  16 0012A        JSB     LIB$ANALYZE_SDESC_R2                    ;
               08   AE              50  D0 00130        MOVL    R0, STATUS                              ;
                    55              52  D0 00134        MOVL    R2, R5                                  ;
               18   AE              51  D0 00137        MOVL    R1, DEFAULT_SIZE                        ;
                    2A        08    AE  E9 0013B        BLBC    STATUS, 13$                             ;
               10   AE        1C    AE  9E 0013F 12$:   MOVAB   RELATED_SIZE, RELATED_ADDR              ; 1022
          27   14   AE              01  E0 00144        BBS     #1, FLAGS, 14$                          ; 1023
                    05              6C  91 00149        CMPB    (AP), #5                                ; 1024
                                    22  1F 0014C        BLSSU   14$                                     ;
                              14    AC  D5 0014E        TSTL    20(AP)                                  ;
                                    1D  13 00151        BEQL    14$                                     ;
                    50        14    AC  D0 00153        MOVL    RELATED_NAME, R0                        ; 1026
                       00000000G    00  16 00157        JSB     LIB$ANALYZE_SDESC_R2                    ;
               08   AE              50  D0 0015D        MOVL    R0, STATUS                              ;
               10   AE              52  D0 00161        MOVL    R2, 16(SP)                              ;
               1C   AE              51  D0 00165        MOVL    R1, RELATED_SIZE                        ;
                    03        08    AE  E8 00169 13$:   BLBS    STATUS, 14$                             ;
                                  0101  31 0016D        BRW     29$                                     ;
                    3E        14    AE  E8 00170 14$:   BLBS    FLAGS, 17$                              ; 1039
          3A        6A              01  E0 00174        BBS     #1, (INTFLAGS), 17$                     ; 1040
                    50        34    AB  9A 00178        MOVZBL  52(FAB), R0                             ; 1041
```

```
04  AE      20      2C  BB           50  2D 0017C          CMPC5   R0, @44(FAB), #32, FILE_SIZE, @FILE_ADDR        ;
                                 0C  BE     00183                                                                  ;
                                     2B  12 00185          BNEQ    17$                                             ;
            0C  BE      04  AE       20  3B 00187          SKPC    #32, FILE_SIZE, @FILE_ADDR                      ; 1043
                                     02  12 0018D          BNEQ    15$                                             ;
                                     51  D4 0018F          CLRL    R1                                              ;
                                     51  D5 00191 15$:     TSTL    R1                                              ;
                                     1D  13 00193          BEQL    17$                                             ;
                        50     0316  C6  D0 00195          MOVL    790(FNBUF), R0                                  ; 1048
                                     11  13 0019A          BEQL    16$                                             ;
                                 18  AE  D5 0019C          TSTL    DEFAULT_SIZE                                    ; 1049
                                     0C  13 0019F          BEQL    16$                                             ;
                        51       03  A0  9A 001A1          MOVZBL  3(R0), R1                                       ; 1054
    51          00      65       18  AE  2D 001A5          CMPC5   DEFAULT_SIZE, (DEFAULT_ADDR), #0, R1, -         ; 1053
                                 04  B0     001AB                  @4(R0)                                          ;
                                     03  12 001AD 16$:     BNEQ    17$                                             ;
                               00D1  31 001AF          BRW     30$                                             ;
                        50     0316  C6  9E 001B2 17$:     MOVAB   790(FNBUF), R0                                  ; 1058
                        54       18  AE  D0 001B7          MOVL    DEFAULT_SIZE, R4                                ; 1062
                                     04  13 001BB          BEQL    18$                                             ;
                                     60  D5 001BD          TSTL    (R0)                                            ; 1063
                                     14  13 001BF          BEQL    19$                                             ;
                                     60  D5 001C1 18$:     TSTL    (R0)                                            ; 1064
                                     1A  13 001C3          BEQL    20$                                             ;
                        50           60  D0 001C5          MOVL    (R0), R0                                        ; 1066
                        51       03  A0  9A 001C8          MOVZBL  3(R0), R1                                       ;
    51          00      65           54  2D 001CC          CMPC5   R4, (DEFAULT_ADDR), #0, R1, @4(R0)              ; 1065
                                 04  B0     001D1                                                                  ;
                                     0A  13 001D3          BEQL    20$                                             ;
                    35  AB           54  90 001D5 19$:     MOVB    R4, 53(FAB)                                     ; 1069
                    30  AB           55  D0 001D9          MOVL    DEFAULT_ADDR, 48(FAB)                           ; 1070
                                     03  11 001DD          BRB     21$                                             ; 1062
                                 35  AB  94 001DF 20$:     CLRB    53(FAB)                                         ; 1073
                    20  AE       34  AB  9A 001E2 21$:     MOVZBL  52(FAB), BLOCKSIZE                              ; 1078
                                     1F  13 001E7          BEQL    23$                                             ;
            0A      14  AE           01  E1 001E9          BBC     #1, FLAGS, 22$                                  ; 1080
                                     5B  DD 001EE          PUSHL   FAB                                             ; 1082
                                 10  A7  9F 001F0          PUSHAB  16(NAM)                                         ;
                  FA34  CF           02  FB 001F3          CALLS   #2, COPY_FILE_STRING                            ;
                                 2C  AB  9F 001F8 22$:     PUSHAB  44(FAB)                                         ; 1084
                                 24  AE  9F 001FB          PUSHAB  BLOCKSIZE                                       ;
             00000000G  00           02  FB 001FE          CALLS   #2, LIB$FREE_VM                                 ;
                                 34  AB  94 00205          CLRB    52(FAB)                                         ; 1085
                    20  AE       04  AE  D0 00208 23$:     MOVL    FILE_SIZE, BLOCKSIZE                            ; 1087
                    34  AB       20  AE  90 0020D          MOVB    BLOCKSIZE, 52(FAB)                              ; 1088
                                 20  AE  D5 00212          TSTL    BLOCKSIZE                                       ; 1089
                                     1B  13 00215          BEQL    25$                                             ;
                                 2C  AB  9F 00217          PUSHAB  44(FAB)                                         ; 1092
                                 24  AE  9F 0021A          PUSHAB  BLOCKSIZE                                       ;
             00000000G  00           02  FB 0021D          CALLS   #2, LIB$GET_VM                                  ;
                        01           50  E8 00224          BLBS    STATUS_2, 24$                                   ;
                                         04 00227          RET                                                     ;
                        50       34  AB  9A 00228 24$:     MOVZBL  52(FAB), R0                                     ; 1095
            2C  BB      0C  BE       50  28 0022C          MOVC3   R0, @FILE_ADDR, @44(FAB)                        ;
            14          14  AE       01  E0 00232 25$:     BBS     #1, FLAGS, 27$                                  ; 1100
                                 1C  AE  D5 00237          TSTL    RELATED_SIZE                                    ; 1102
                                     0C  13 0023A          BEQL    26$                                             ;
```

```
                  03    A8        1C   AE  90 0023C          MOVB    RELATED_SIZE, 3(RNAM)            ; 1104
                  04    A8        10   AE  D0 00241          MOVL    RELATED_ADDR, 4(RNAM)            ; 1105
                                       03  11 00246          BRB     27$                              ; 1102
                                  03   A8  94 00248 26$:     CLRB    3(RNAM)                          ; 1108
                        6A             06  8A 0024B 27$:     BICB2   #6, (INTFLAGS)                   ; 1115
                  33    A7        80   8F  88 0024E          BISB2   #128, 51(NAM)                    ; 1116
                                       5B  DD 00253          PUSHL   FAB                              ; 1117
           00000000G    00             01  FB 00255          CALLS   #1, SYS$PARSE                    ;
                  08    AE             50  D0 0025C          MOVL    R0, STATUS                       ;
                        69        08   AE  D0 00260          MOVL    STATUS, (NEXT_STATUS)            ; 1118
                                       6E  D5 00264          TSTL    STVADDR                          ; 1119
                                       05  13 00266          BEQL    28$                              ;
                  00    BE        0C   AB  D0 00268          MOVL    12(FAB), @STVADDR                ; 1121
                        12        08   AE  E8 0026D 28$:     BLBS    STATUS, 30$                      ; 1122
                                  08   AC  DD 00271 29$:     PUSHL   RESULT_NAME                      ; 1125
                                       5B  DD 00274          PUSHL   FAB                              ;
                FCC5    CF             02  FB 00276          CALLS   #2, COPY_RESULT_NAME             ;
                        69      F9A5   CF  D0 0027B          MOVL    RMSNMF, (NEXT_STATUS)            ; 1126
                              0105     31     00280          BRW     45$                              ; 1127
                F99C    CF             69  D1 00283 30$:     CMPL    (NEXT_STATUS), RMSNMF            ; 1136
                                       19  12 00288          BNEQ    32$                              ;
        0B        14    AE             01  E0 0028A          BBS     #1, FLAGS, 31$                   ; 1138
                                  0C   AC  DD 0028F          PUSHL   CONTEXT                          ; 1140
                FCE3    CF             01  FB 00292          CALLS   #1, FIND_FILE_CLEANUP            ;
                                  0C   BC  D4 00297          CLRL    @CONTEXT                         ; 1141
                        6A             02  88 0029A 31$:     BISB2   #2, (INTFLAGS)                   ; 1143
                        50      F983   CF  D0 0029D          MOVL    RMSNMF, R0                       ; 1144
                                           04 002A2          RET                                      ;
                                  35   AB  95 002A3 32$:     TSTB    53(FAB)                          ; 1152
                                       26  13 002A6          BEQL    33$                              ;
        22              6A             02  E0 002A8          BBS     #2, (INTFLAGS), 33$              ; 1153
   24   AE              6B      0050   8F  28 002AC          MOVC3   #80, (FAB), NFAB                 ; 1165
                  58    AE        35   AB  90 002B3          MOVB    53(FAB), NFAB+52                 ; 1166
                  50    AE        30   AB  D0 002B8          MOVL    48(FAB), NFAB+44                 ; 1167
                                  24   AE  9F 002BD          PUSHAB  NFAB                             ; 1168
                                  10   A7  9F 002C0          PUSHAB  16(NAM)                          ;
                F964    CF             02  FB 002C3          CALLS   #2, COPY_FILE_STRING             ;
                0316    C6        10   A7  D0 002C8          MOVL    16(NAM), 790(FNBUF)              ; 1169
                        52        34   A7  9E 002CE 33$:     MOVAB   52(NAM), R2                      ; 1172
        11              62             03  E1 002D2          BBC     #3, (R2), 34$                    ;
        0D              6A             02  E0 002D6          BBS     #2, (INTFLAGS), 34$              ; 1173
                        6A             04  88 002DA          BISB2   #4, (INTFLAGS)                   ; 1175
                  35    AB             02  90 002DD          MOVB    #2, 53(FAB)                      ; 1176
                  30    AB      F943   CF  9E 002E1          MOVAB   WILD_VER, 48(FAB)                ; 1177
        04        40    AB             03  E0 002E7 34$:     BBS     #3, 64(FAB), 35$                 ; 1183
        08              62             11  E1 002EC          BBC     #17, (R2), 36$                   ; 1184
                        04        43   AB  E8 002F0 35$:     BLBS    67(FAB), 36$                     ; 1185
                        11        02   A2  E9 002F4          BLBC    2(R2), 37$                       ; 1186
                        69      F928   CF  D0 002F8 36$:     MOVL    RMSNMF, (NEXT_STATUS)            ; 1188
                                  08   AC  DD 002FD          PUSHL   RESULT_NAME                      ; 1189
                                       5B  DD 00300          PUSHL   FAB                              ;
                FC39    CF             02  FB 00302          CALLS   #2, COPY_RESULT_NAME             ;
                              [illegible]  11 00307          BRB     45$                              ; 1190
                        1B        14   AE  E9 00309 37$:     BLBC    FLAGS, 38$                       ; 1196
                        17        01   A2  E9 0030D          BLBC    1(R2), 38$                       ; 1197
                        69      F90F   CF  D0 00311          MOVL    RMSNMF, (NEXT_STATUS)            ; 1199
                                  08   AC  DD 00316          PUSHL   RESULT_NAME                      ; 1200
```

```
                                5B  DD 00319          PUSHL   FAB                          ;
        FC20    CF              02  FB 0031B          CALLS   #2, COPY_RESULT_NAME         ;
                50 0015112A     8F  D0 00320          MOVL    #1380650, R0                 ; 1201
                                    04 00327          RET
  04            62              11  E1 00328 38$:     BBC     #17, (R2), 39$               ; 1207
                0D          01  A2  E9 0032C          BLBC    1(R2), 40$                   ;
                                5B  DD 00330 39$:     PUSHL   FAB                          ; 1209
     00000000G  00              01  FB 00332          CALLS   #1, SYS$SEARCH               ;
          08    AE              50  D0 00339          MOVL    R0, STATUS                   ;
                1C          08  AE  E8 0033D 40$:     BLBS    STATUS, 43$                  ; 1213
                                6E  D5 00341          TSTL    STVADDR                      ; 1214
                                05  13 00343          BEQL    41$                          ;
          00    BE          0C  AB  D0 00345          MOVL    12(FAB), @STVADDR            ; 1216
                0F          08  AE  E8 0034A 41$:     BLBS    STATUS, 43$                  ; 1222
  06            62              14  E1 0034E          BBC     #20, (R2), 42$               ; 1223
                24          0C  AB  D1 00352          CMPL    12(FAB), #36                 ;
                                05  13 00356          BEQL    43$                          ;
                69        F8C8  CF  D0 00358 42$:     MOVL    RMSNMF, (NEXT_STATUS)        ; 1225
                05          01  A2  E8 0035D 43$:     BLBS    1(R2), 44$                   ; 1231
                69        F8BF  CF  D0 00361          MOVL    RMSNMF, (NEXT_STATUS)        ; 1234
                            08  AC  DD 00366 44$:     PUSHL   RESULT_NAME                  ; 1240
                                5B  DD 00369          PUSHL   FAB                          ;
        FBD0    CF              02  FB 0036B          CALLS   #2, COPY_RESULT_NAME         ;
        F8AE    CF          08  AE  D1 00370          CMPL    STATUS, RMSNMF               ; 1244
                                10  12 00376          BNEQ    45$                          ;
  0B      14    AE              01  E0 00378          BBS     #1, FLAGS, 45$               ; 1245
                            0C  AC  DD 0037D          PUSHL   CONTEXT                      ; 1247
        FBF5    CF              01  FB 00380          CALLS   #1, FIND_FILE_CLEANUP        ;
                            0C  BC  D4 00385          CLRL    @CONTEXT                     ; 1248
                50          08  AE  D0 00388 45$:     MOVL    STATUS, R0                   ; 1251
                                    04 0038C          RET                                  ; 1253

; Routine Size:  909 bytes,    Routine Base:  _LIB$CODE + 03DC
```

```
;  1171          1254  1 %SBTTL  'LIB$FILE_SCAN_END Clean up after LIB$FILE_SCAN';
;  1172          1255  1 GLOBAL ROUTINE LIB$FILE_SCAN_END(FAB,CONTEXT) =
;  1173          1256  1 !---
;  1174          1257  1 !       This routine is called after using LIB$FILE_SCAN.  It performs
;  1175          1258  1 !       a parse of the null string to deallocate any saved RMS context.
;  1176          1259  1 !       If LIB$FILE_SCAN was directed to perform multiple input file
;  1177          1260  1 !       specification processing, the saved file specifications are
;  1178          1261  1 !       deallocated.
;  1179          1262  1 !
;  1180          1263  1 ! Calling sequence:
;  1181          1264  1 !
;  1182          1265  1 !       status.wl = lib$file_scan_end(fab,context.wl.r)
;  1183          1266  1 !
;  1184          1267  1 ! Inputs:
;  1185          1268  1 !
;  1186          1269  1 !       fab =  [OPTIONAL] Address of the FAB used with LIB$FILE_SCAN
;  1187          1270  1 !       context = [OPTIONAL] Address of the context used with LIB$FILE_SCAN
;  1188          1271  1 !
;  1189          1272  1 ! Outputs:
;  1190          1273  1 !
;  1191          1274  1 !       NONE
;  1192          1275  1 !
;  1193          1276  1 ! Implicit outputs:
;  1194          1277  1 !
;  1195          1278  1 !       Saved context deallocated if context argument is supplied.
;  1196          1279  1 !
;  1197          1280  1 ! Routine values:
;  1198          1281  1 !
;  1199          1282  1 !       RMS$_FAB        fab argument is not address of a valid FAB
;  1200          1283  1 !       success
;  1201          1284  1 !---
;  1202          1285  2 BEGIN
;  1203          1286  2
;  1204          1287  2 BUILTIN
;  1205          1288  2         NULLPARAMETER;
;  1206          1289  2
;  1207          1290  2 LOCAL
;  1208          1291  2         RNAM : REF $BBLOCK,
;  1209          1292  2         NAM : REF $BBLOCK;
;  1210          1293  2
;  1211          1294  2 MAP
;  1212          1295  2         FAB : REF $BBLOCK,
;  1213          1296  2         CONTEXT : REF VECTOR[,LONG];
;  1214          1297  2
;  1215          1298  2 !
;  1216          1299  2 ! Ensure it's a FAB
;  1217          1300  2 !
;  1218          1301  2 IF NOT NULLPARAMETER(1)
;  1219          1302  2 THEN
;  1220          1303  3         BEGIN
;  1221          1304  3         IF .FAB[FAB$B_BID] NEQ FAB$C_BID
;  1222          1305  3             OR .FAB[FAB$B_BLN] NEQ FAB$C_BLN
;  1223          1306  3         THEN
;  1224          1307  3                 RETURN RMS$_FAB;
;  1225          1308  3 !
;  1226          1309  3 ! Parse the null string
;  1227          1310  3 !
```

```
; 1228        1311  3           PARSE_NULL_STRING(.FAB);
; 1229        1312  2           END;
; 1230        1313  2   !
; 1231        1314  2   ! If supplied, deallocate any input file context
; 1232        1315  2   !
; 1233        1316  2   IF NOT NULLPARAMETER(2)
; 1234        1317  3   THEN    BEGIN
; 1235        1318  3           NAM = .CONTEXT[0];
; 1236        1319  3           WHILE .NAM NEQ 0
; 1237        1320  4           DO      BEGIN
; 1238        1321  4                   RNAM = .NAM[NAM$L_RLF];
; 1239        1322  4                   LIB$FREE_VM(%REF(NAM$C_BLN+.NAM[NAM$B_RSL]),NAM);
; 1240        1323  4                   NAM = .RNAM;
; 1241        1324  3                   END;
; 1242        1325  3           !
; 1243        1326  3           ! Zero the context
; 1244        1327  3           !
; 1245        1328  3           CONTEXT[0] = 0;
; 1246        1329  2           END;
; 1247        1330  2   RETURN SS$_NORMAL
; 1248        1331  1   END;



                                         0004 00000          .ENTRY  LIB$FILE_SCAN_END, Save R2          ; 1255
                  5E               08   C2 00002          SUBL2   #8, SP                               ;
                                   6C   95 00005          TSTB    (AP)                                 ; 1301
                                   24   13 00007          BEQL    3$                                   ;
                          04       AC   D5 00009          TSTL    4(AP)                                ;
                                   1F   13 0000C          BEQL    3$                                   ;
                  50      04       AC   D0 0000E          MOVL    FAB, R0                              ; 1304
                  03               60   91 00012          CMPB    (R0), #3                             ;
                                   07   12 00015          BNEQ    1$                                   ;
           50     8F      01       A0   91 00017          CMPB    1(R0), #80                           ; 1305
                                   08   13 0001C          BEQL    2$                                   ;
                  50 0001850C      8F   D0 0001E  1$:     MOVL    #99596, R0                           ; 1307
                                        04 00025          RET                                          ;
                                   50   DD 00026  2$:     PUSHL   R0                                   ; 1311
         F918     CF               01   FB 00028          CALLS   #1, PARSE_NULL_STRING                ;
                  02               6C   91 0002D  3$:     CMPB    (AP), #2                             ; 1316
                                   37   1F 00030          BLSSU   6$                                   ;
                          08       AC   D5 00032          TSTL    8(AP)                                ;
                                   32   13 00035          BEQL    6$                                   ;
           04     AE      08       BC   D0 00037          MOVL    @CONTEXT, NAM                        ; 1318
                  50      04       AE   D0 0003C  4$:     MOVL    NAM, R0                              ; 1319
                                   24   13 00040          BEQL    5$                                   ;
                  52      10       A0   D0 00042          MOVL    16(R0), RNAM                         ; 1321
                          04       AE   9F 00046          PUSHAB  NAM                                  ; 1322
           04     AE      03       A0   9A 00049          MOVZBL  3(R0), 4(SP)                         ;
           04     AE 00000060      8F   C0 0004E          ADDL2   #96, 4(SP)                           ;
                          04       AE   9F 00056          PUSHAB  4(SP)                                ;
    00000000G     00               02   FB 00059          CALLS   #2, LIB$FREE_VM                      ;
           04     AE               52   D0 00060          MOVL    RNAM, NAM                            ; 1323
                                   D6   11 00064          BRB     4$                                   ; 1319
                          08       BC   D4 00066  5$:     CLRL    @CONTEXT                             ; 1328
```

```
                                50          01  D0 00069 6$:    MOVL    #1, R0                          ; 1330
                                                04 0006C        RET                                     ; 1331

; Routine Size:  109 bytes,    Routine Base:  _LIB$CODE + 0769
```

```
;  1250          1332  1 %SBTTL  'LIB$FIND_FILE_END Clean up after LIB$FIND_FILE';
;  1251          1333  1 GLOBAL ROUTINE LIB$FIND_FILE_END(CONTEXT) =
;  1252          1334  1 !---
;  1253          1335  1 !       This routine is called after using LIB$FIND_FILE.  It performs
;  1254          1336  1 !       a parse of the null string to deallocate any saved RMS context,
;  1255          1337  1 !       and then the allocated context block is deallocated.
;  1256          1338  1 !
;  1257          1339  1 ! Calling sequence:
;  1258          1340  1 !
;  1259          1341  1 !       status.wl = lib$find_file_end(context.wl.r)
;  1260          1342  1 !
;  1261          1343  1 ! Inputs:
;  1262          1344  1 !
;  1263          1345  1 !       context =  Address of the context used with LIB$FIND_FILE
;  1264          1346  1 !
;  1265          1347  1 ! Outputs:
;  1266          1348  1 !
;  1267          1349  1 !       NONE
;  1268          1350  1 !
;  1269          1351  1 ! Implicit outputs:
;  1270          1352  1 !
;  1271          1353  1 !       Saved context deallocated.
;  1272          1354  1 !
;  1273          1355  1 ! Routine values:
;  1274          1356  1 !
;  1275          1357  1 !       RMS$_FAB        context points to an invalid context block
;  1276          1358  1 !       success
;  1277          1359  1 !---
;  1278          1360  2 BEGIN
;  1279          1361  2 MAP
;  1280          1362  2         CONTEXT : REF VECTOR[,LONG];
;  1281          1363  2
;  1282          1364  2 LOCAL
;  1283          1365  2         FAB : REF $BBLOCK;
;  1284          1366  2 !
;  1285          1367  2 ! If context is 0, nothing to do
;  1286          1368  2 !
;  1287          1369  2 IF .CONTEXT[0] EQL 0
;  1288          1370  2 THEN
;  1289          1371  2         RETURN SS$_NORMAL;
;  1290          1372  2 !
;  1291          1373  2 ! Ensure that context points to a FAB
;  1292          1374  2 !
;  1293          1375  2 FAB = .CONTEXT[0];
;  1294          1376  2 IF .FAB[FAB$B_BID] NEQ FAB$C_BID
;  1295          1377  2     OR .FAB[FAB$B_BLN] NEQ FAB$C_BLN
;  1296          1378  2 THEN
;  1297          1379  2         RETURN RMS$_FAB;
;  1298          1380  2 !
;  1299          1381  2 ! Do most of the work
;  1300          1382  2 !
;  1301          1383  2 FIND_FILE_CLEANUP(.CONTEXT);
;  1302          1384  2 !
;  1303          1385  2 ! Zero the context pointer
;  1304          1386  2 !
;  1305          1387  2 CONTEXT[0] = 0;
;  1306          1388  2 RETURN SS$_NORMAL
```

```
; 1307          1389  1 END;

                                    0004 00000        .ENTRY  LIB$FIND_FILE_END, Save R2                ; 1333
                    52          04  AC  D0 00002      MOVL    CONTEXT, R2                               ; 1369
                                    62  D5 00006      TSTL    (R2)                                      ;
                                    20  13 00008      BEQL    3$                                        ;
                    50              62  D0 0000A      MOVL    (R2), FAB                                 ; 1375
                    03              60  91 0000D      CMPB    (FAB), #3                                 ; 1376
                                    07  12 00010      BNEQ    1$                                        ;
              50    8F          01  A0  91 00012      CMPB    1(FAB), #80                               ; 1377
                                    08  13 00017      BEQL    2$                                        ;
                    50 0001850C     8F  D0 00019 1$:  MOVL    #99596, R0                                ; 1379
                                        04 00020      RET                                               ;
                                    52  DD 00021 2$:  PUSHL   R2                                        ; 1383
            FB58    CF              01  FB 00023      CALLS   #1, FIND_FILE_CLEANUP                     ;
                                    62  D4 00028      CLRL    (R2)                                      ; 1387
                    50              01  D0 0002A 3$:  MOVL    #1, R0                                    ; 1388
                                        04 0002D      RET                                               ; 1389

; Routine Size:  46 bytes,    Routine Base:  _LIB$CODE + 07D6

; 1308          1390  0 END ELUDOM

                                                FMG$FILE_SCAN==     LIB$FILE_SCAN

;                          PSECT SUMMARY
;
;      Name                        Bytes                          Attributes
;
;   _LIB$CODE                         2052  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)


;                       Library Statistics
;
;                                            -------- Symbols --------      Pages        Processing
;      File                                  Total    Loaded   Percent      Mapped       Time
;
;   _$255$DUA28:[SYSLIB]STARLET.L32;1         9776        85         0         581         00:00.7



;                          COMMAND QUALIFIERS
```

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

; Size:          2044 code + 8 data bytes
; Run Time:         00:30.0
; Elapsed Time:     01:50.3
; Lines/CPU Min:     2780
; Lexemes/CPU-Min: 31542
; Memory Used:  330 pages
; Compilation Complete
```

LIBEMODH
LIS

LIBEMODD
LIS

LIBESTEMU
LIS

LIBEF
LIS

LIBEMULAT
LIS

LIBFFS
LIS

LIBFINCVT
LIS

LIBE2AREV
LIS

LIBFAO
LIS

LIBEMODG
LIS

LIBEXTV
LIS

LIBEDIV
LIS

LIBESTABL
LIS

LIBFFC
LIS

LIBEMODF
LIS

LIBEMUL
LIS

LIBFILSCA
LIS

LIBEXTZV
LIS

LIBEBCASC
LIS

LIBFAOL
LIS