```
MMM         MMM   TTTTTTTTTTTTTTT     AAAAAAAAA         AAAAAAAAA        CCCCCCCCCCCC   PPPPPPPPPPP
MMM         MMM   TTTTTTTTTTTTTTT     AAAAAAAAA         AAAAAAAAA        CCCCCCCCCCCC   PPPPPPPPPPP
MMM         MMM   TTTTTTTTTTTTTTT     AAAAAAAAA         AAAAAAAAA        CCCCCCCCCCCC   PPPPPPPPPPP
MMMMMM   MMMMMM         TTT        AAA         AAA   AAA         AAA  CCC               PPP         PPP
MMMMMM   MMMMMM         TTT        AAA         AAA   AAA         AAA  CCC               PPP         PPP
MMMMMM   MMMMMM         TTT        AAA         AAA   AAA         AAA  CCC               PPP         PPP
MMM   MMM   MMM         TTT        AAA         AAA   AAA         AAA  CCC               PPP         PPP
MMM   MMM   MMM         TTT        AAA         AAA   AAA         AAA  CCC               PPP         PPP
MMM   MMM   MMM         TTT        AAA         AAA   AAA         AAA  CCC               PPP         PPP
MMM         MMM         TTT        AAA         AAA   AAA         AAA  CCC               PPPPPPPPPPPP
MMM         MMM         TTT        AAA         AAA   AAA         AAA  CCC               PPPPPPPPPPPP
MMM         MMM         TTT        AAA         AAA   AAA         AAA  CCC               PPPPPPPPPPPP
MMM         MMM         TTT        AAAAAAAAAAAAAAA   AAAAAAAAAAAAAAA  CCC               PPP
MMM         MMM         TTT        AAAAAAAAAAAAAAA   AAAAAAAAAAAAAAA  CCC               PPP
MMM         MMM         TTT        AAAAAAAAAAAAAAA   AAAAAAAAAAAAAAA  CCC               PPP
MMM         MMM         TTT        AAA         AAA   AAA         AAA  CCC               PPP
MMM         MMM         TTT        AAA         AAA   AAA         AAA  CCC               PPP
MMM         MMM         TTT        AAA         AAA   AAA         AAA  CCC               PPP
MMM         MMM         TTT        AAA         AAA   AAA         AAA     CCCCCCCCCCCC   PPP
MMM         MMM         TTT        AAA         AAA   AAA         AAA     CCCCCCCCCCCC   PPP
MMM         MMM         TTT        AAA         AAA   AAA         AAA     CCCCCCCCCCCC   PPP
```

FILEID**PARSE

```
PPPPPPPP      AAAAAA    RRRRRRRR      SSSSSSSS  EEEEEEEEEE
PPPPPPPP      AAAAAA    RRRRRRRR      SSSSSSSS  EEEEEEEEEE
PP      PP  AA      AA  RR      RR  SS          EE
PP      PP  AA      AA  RR      RR  SS          EE
PP      PP  AA      AA  RR      RR  SS          EE
PP      PP  AA      AA  RR      RR  SS          EE
PPPPPPPP    AA      AA  RRRRRRRR      SSSSSS    EEEEEEEE
PPPPPPPP    AA      AA  RRRRRRRR      SSSSSS    EEEEEEEE
PP          AAAAAAAAAA  RR  RR              SS  EE
PP          AAAAAAAAAA  RR  RR              SS  EE
PP          AA      AA  RR    RR            SS  EE              ....
PP          AA      AA  RR    RR            SS  EE              ....
PP          AA      AA  RR      RR  SSSSSSSS    EEEEEEEEEE      ....
PP          AA      AA  RR      RR  SSSSSSSS    EEEEEEEEEE      ....

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
```

```
0 MODULE PARSE ( LANGUAGE ( BLISS32 ),
0                 IDENT = 'V04-000'
0                 ) =
1 BEGIN
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1
1 !++
1 !
1 ! FACILITY:  MTAACP
1 !
1 ! ABSTRACT:
1 !       This module parses the file identification field in the HDR1 and
1 !       in the case of VMS long file names continued in the HDR4 or file
1 !       name string, then generates and returns resultant file spec string.
1 !
1 ! ENVIRONMENT:
1 !
1 !       STARLET operating system, including privileged system services
1 !       and internal exec routines.
1 !
1 !--
1
1 !
1 !
1 ! AUTHOR:  David Michael Walp,   CREATION DATE:  17-NOV-1981
1 !
1 ! MODIFIED BY:
1 !
1 !       V03-003 MMD0282         Meg Dumont,     23-Mar-1984  10:31
1 !               Fix long file name support such that for ANSI version
1 !               3 volumes it converts the exentsion length to
1 !               ASCII before writing it to the label.
1 !
1 !       V03-002 MMD0184         Meg Dumont,     6-Jul-1983  18:30
1 !               Add support for $ and - as valid VMS characters to tape
```

```
!
!       V03-001 MMD0158         Meg Dumont,     26-Apr-1983  9:21
!               Add support for underscore as a valid character to tape. Add
!               support for VMS long file names includes: 1) Changing the
!               default file name sizes to support the long file name, uses
!               the symbol FILE_SPEC_MAX. 2) Change the routines PARSE_HDR1_FID
!               to PARSE_FID and enhance it to understand that the FID maybe
!               split between the HDR1 and HDR4 labels. 3) Change file name
!               length checked in PARSE_PATTERN_SPEC. 4) In RESULTANT_STRING
!               change length of RESULTANT_MAX_LENGTH.
!
!       V03-002 DMW00072        David Michael Walp      28-Jan-1981
!               Fix wildcard characters in quoted string problem
!
!       V03-001 DMW00067        David Michael Walp      11-Jan-1981
!               Accept ANSI name in quotes thru file name parameters
!
!**

LIBRARY 'SYS$LIBRARY:LIB.L32';

REQUIRE 'SRC$:MTADEF.B32';

FORWARD ROUTINE
    CALC_VERSION            : NOVALUE,      ! calculate VMS version from ANSI
    PARSE_FID               : COMMON_CALL,  ! parse the file id fields in HDR1 and HDR4
    PARSE_NAME_TYPE,                        ! parse a filespec into name + type
    PARSE_PATTERN_SPEC      : COMMON_CALL,  ! parse a input wildcard file spec
    PARSE_QUOTED_NAME       : COMMON_CALL NOVALUE, ! parse a quoted 17 ANSI "a"
                                            !       character file name
    RESULTANT_STRING        : COMMON_CALL NOVALUE, ! prepare and return the
                                            !       resultant string name
    RETURN_RESULTANT_SPEC,                  ! write back file name information
    STRIP_VERSION           : COMMON_CALL;  ! strips the version from file spec
```

```
1 !++
1 !
1 ! UPLIT was used instead of CH$TRANSTABLE here, the code produced is the
1 !  same (ie the constant string generated).  UPLIT was used because
1 !  CH$TRANSTABLE generates a warning error because more then a single
1 !  character at a time is specified in the %ASCII. ( BLISS KLUDGE )
1 !
1 !--
1
1 GLOBAL BIND
1
1       ! This table will upcase a..z and return '@' for any non ANSI
1       !   'a' characters.  Bad characters are thus flaged.
1       !
1       ANSI_A_BAD    = UPLIT BYTE ( %ASCII '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@',
1                                    %ASCII ' !"@@%&''()*+,-./0123456789:;<=>?',
1                                    %ASCII '@ABCDEFGHIJKLMNOPQRSTUVWXYZ@@@@_',
1                                    %ASCII '@ABCDEFGHIJKLMNOPQRSTUVWXYZ@@@@@',
1                                    %ASCII '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@',
1                                    %ASCII '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@',
1                                    %ASCII '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@',
1                                    %ASCII '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@' ),
1
1       ! This table will upcase a..z and return space for any non ANSI
1       !   'a' characters.  Thus bad characters are converted to good characters.
1       !
1       ANSI_A_GOOD   = UPLIT BYTE ( %ASCII '                                ',
1                                    %ASCII ' !"  %&''()*+,-./0123456789:;<=>?',
1                                    %ASCII ' ABCDEFGHIJKLMNOPQRSTUVWXYZ    _',
1                                    %ASCII ' ABCDEFGHIJKLMNOPQRSTUVWXYZ     ',
1                                    %ASCII '                                ',
1                                    %ASCII '                                ',
1                                    %ASCII '                                ',
1                                    %ASCII '                                ' ),
1
1       ! point to a '@'
1       !
1       ESC_CHAR = ANSI_A_BAD;
1
1 BIND
1
1       ! This table will upcase a..z and return '@' for any non-VMS character
1       !   including wild cards
1       !
1       VMS_NO_WILD  = UPLIT BYTE ( %ASCII '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@',
1                                   %ASCII '@@@@$@@@@@@@@-@@0123456789@@@@@@',
1                                   %ASCII '@ABCDEFGHIJKLMNOPQRSTUVWXYZ@@@@_',
1                                   %ASCII '@ABCDEFGHIJKLMNOPQRSTUVWXYZ@@@@@',
1                                   %ASCII '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@',
1                                   %ASCII '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@',
1                                   %ASCII '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@',
1                                   %ASCII '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@' ),
1
1
1       ! This table will upcase a..z and return '@' for any non-VMS character
1       !   wild cards ( %, * ) are valid characters in this table
1       !
```

```
;  1   VMS_WILD     = UPLIT BYTE ( %ASCII '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@',
;  1                               %ASCII '@@@@$%@@@@*@@-@@0123456789@@@@@@',
;  1                               %ASCII '@ABCDEFGHIJKLMNOPQRSTUVWXYZ@@@@_',
;  1                               %ASCII '@ABCDEFGHIJKLMNOPQRSTUVWXYZ@@@@@',
;  1                               %ASCII '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@',
;  1                               %ASCII '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@',
;  1                               %ASCII '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@',
;  1                               %ASCII '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@' );
;  1
```

```
1 GLOBAL ROUTINE PARSE_NAME_TYPE ( WILD_CARDS,
1                                  SPEC_LEN,
1                                  SPEC_PTR,
1                                  OUT_DESC ) =
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !       This routine parses a VMS file specification string into a name and type
1 !       fields
1 !
1 ! CALLING SEQUENCE:
1 !       error = PARSE_NAME_TYPE ( ARG1, ARG2, ARG3, ARG4 )
1 !
1 ! INPUT PARAMETERS:
1 !       ARG1 - flags for "if wild cards are valid"
1 !       ARG2 - the length of the file specification
1 !       ARG3 - the address of the file specification
1 !
1 ! IMPLICIT INPUTS:
1 !       none
1 !
1 ! OUTPUT PARAMETERS:
1 !       ARG4 - pointer to the output buffer - size is updated
1 !                 ASSUMED THAT OUTPUT BUFFER >= FILE_SPEC_MAX BYTES
1 !
1 ! IMPLICIT OUTPUTS:
1 !       none
1 !
1 ! ROUTINE VALUE:
1 !       error code - SS$_NORMAL
1 !                  - SS$_BADFILENAME
1 !
1 ! SIDE EFFECTS:
1 !       none
1 !
1 !--
1
2     BEGIN
2
2     MAP
2       WILD_CARDS   : BITVECTOR [ 1 ],
2       SPEC_LEN     : LONG,
2       SPEC_PTR     : REF VECTOR [ , BYTE ],
2       OUT_DESC     : REF VECTOR [ 2, LONG ];
2
2     EXTERNAL
2       LOCAL_FIB    : BBLOCK;
2
2     LITERAL
2       MAX_NAME_FIELD_LEN = 39,              ! Support for VMS long file
2       MAX_TYPE_FIELD_LEN = 39;              ! names
2
2     LOCAL
2       IN_INDEX     : LONG,                  ! index into input file spec
2       OUT_INDEX    : LONG,                  ! index into output file spec
2       REM_LEN      : LONG,                  ! size minus type field
```

```
2           TABLE       : REF VECTOR [ 128, BYTE ], ! ptr to VMS file spec chars
2           TYPE_LEN    : LONG;                     ! size of type field
2       BIND
2           OUT_BUF = OUT_DESC [ 1 ] : REF VECTOR [ , BYTE ];
2
2       ! set up which table is to be use for translation
2       !
2       TABLE = ( IF .WILD_CARDS [ 0 ] THEN VMS_WILD ELSE VMS_NO_WILD );
2
2       !++
2       !
2       ! PARSE the name field
2       !
2       !--
2       ! find the dot in the spec ( IN_INDEX points to it )
2       ! point to the end of the spec if not found
2       ! calculate the size of the name field
2       !
2       IN_INDEX = CH$FIND_CH ( .SPEC_LEN, SPEC_PTR [ 0 ], '.' );
3       IN_INDEX = ( IF CH$FAIL ( .IN_INDEX )
3                   THEN .SPEC_LEN
2                   ELSE .IN_INDEX - SPEC_PTR [ 0 ] );
2
2       ! test if the name field is too long
2       !
2       IF ( .IN_INDEX GTR MAX_NAME_FIELD_LEN ) THEN RETURN SS$_BADFILENAME;
2
2       ! move string into the output buffer and
2       ! test for invalid characters in the name field
2       !
2       IF 0 NEQ MOVTUC ( IN_INDEX, SPEC_PTR [ 0 ], ESC_CHAR, .TABLE,
2                         IN_INDEX, OUT_BUF  [ 0 ] )
2       THEN RETURN SS$_BADFILENAME;
2
2       ! if all wild name flag is set stuff "*" into output buffer
2       ! adjust output index
2       !
2       IF .WILD_CARDS [ 0 ] AND .LOCAL_FIB [ FIB$V_ALLNAM ]
2       THEN
3           BEGIN
3           OUT_BUF [ 0 ] = '*';
3           OUT_INDEX     = 1;
3           END
2       ELSE OUT_INDEX = .IN_INDEX;
2
2       ! stuff the dot into the output buffer
2       !
2       OUT_BUF [ .OUT_INDEX ] = '.';
2
2       !++
2       !
2       ! PARSE the type field
2       !
2       !--
2       ! OUT_INDEX now points to the next avaliable output character
2       ! IN_INDEX  now points to the start of type field
2       !
```

```
;  2      OUT_INDEX = .OUT_INDEX + 1;
;  2      IN_INDEX  = .IN_INDEX  + 1;
;  2
;  2      ! find the end of the type field
;  2      !
;  2      REM_LEN  = .SPEC_LEN - .IN_INDEX;
;  2      TYPE_LEN = CH$FIND_CH ( .REM_LEN, SPEC_PTR [ .IN_INDEX ], ' ' );
;  3      TYPE_LEN = ( IF CH$FAIL ( .TYPE_LEN )
;  3                   THEN .REM_LEN
;  2                   ELSE .TYPE_LEN - SPEC_PTR [ .IN_INDEX ] );
;  2
;  2      ! test if the type field is too long
;  2      !
;  2      IF ( .TYPE_LEN GTR MAX_TYPE_FIELD_LEN ) THEN RETURN SS$_BADFILENAME;
;  2
;  2      ! test if all trailing characters are spaces
;  2      !
;  2      REM_LEN  = .REM_LEN - .TYPE_LEN;
;  2      IF NOT CH$FAIL (
;  2            CH$FIND_NOT_CH ( .REM_LEN, SPEC_PTR [ .IN_INDEX + .TYPE_LEN ], ' ' ))
;  2      THEN RETURN SS$_BADFILENAME;
;  2
;  2      ! move string into the output buffer
;  2      ! test for invalid characters in the type field
;  2      !
;  2      REM_LEN = .OUT_DESC [ 0 ] - .OUT_INDEX;
;  2      IF 0 NEQ MOVTUC ( TYPE_LEN, SPEC_PTR [ .IN_INDEX  ], ESC_CHAR, .TABLE,
;  2                        REM_LEN,  OUT_BUF  [ .OUT_INDEX ] )
;  2      THEN RETURN SS$_BADFILENAME;
;  2
;  2      ! if all wild type flag is set stuff "*" into output buffer
;  2      ! adjust output length
;  2      !
;  2      IF .WILD_CARDS [ 0 ] AND .LOCAL_FIB [ FIB$V_ALLTYP ]
;  2      THEN
;  3          BEGIN
;  3          OUT_BUF [ .OUT_INDEX ] = '*';
;  3          OUT_DESC [ 0 ] = .OUT_INDEX + 1;
;  3          END
;  2      ELSE OUT_DESC [ 0 ] = .OUT_INDEX + .TYPE_LEN;
;  2
;  2      RETURN SS$_NORMAL;
;  1      END;                                 ! end of routine PARSE_NAME_TYPE

                                                           .TITLE  PARSE
                                                           .IDENT  \V04-000\

                                                           .PSECT  $CODE$,NOWRT,2

40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  00000 P.AAA:  .ASCII  \@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@\
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  0000F
                                        40 40  0001E
2E 2D 2C 2B 2A 29 28 27 26 25 40 40 22 21 20  00020         .ASCII  \ !"@@%&'()*+,-./0123456789:;<=>?\
3D 3C 3B 3A 39 38 37 36 35 34 33 32 31 30 2F  0002F
                                        3F 3E  0003E
4E 4D 4C 4B 4A 49 48 47 46 45 44 43 42 41 40  00040         .ASCII  \@ABCDEFGHIJKLMNOPQRSTUVWXYZ@@@@_\
```

```
40 40 40 5A 59 58 57 56 55 54 53 52 51 50 4F  0004F
                                          5F 40  0005E
4E 4D 4C 4B 4A 49 48 47 46 45 44 43 42 41 40  00060          .ASCII  \@ABCDEFGHIJKLMNOPQRSTUVWXYZ@@@@@\
40 40 40 5A 59 58 57 56 55 54 53 52 51 50 4F  0006F
                                          40 40  0007E
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  00080          .ASCII  \@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@\
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  0008F
                                          40 40  0009E
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  000A0          .ASCII  \@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@\
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  000AF
                                          40 40  000BE
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  000C0          .ASCII  \@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@\
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  000CF
                                          40 40  000DE
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  000E0          .ASCII  \@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@\
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  000EF
                                          40 40  000FE
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20  00100 P.AAB:  .ASCII  \                                \
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20  0010F
                                          20 20  0011E
2E 2D 2C 2B 2A 29 28 27 26 25 20 20 22 21 20  00120          .ASCII  \ !"  %&'()*+,-./0123456789:;<=>?\
3D 3C 3B 3A 39 38 37 36 35 34 33 32 31 30 2F  0012F
                                          3F 3E  0013E
4E 4D 4C 4B 4A 49 48 47 46 45 44 43 42 41 20  00140          .ASCII  \ ABCDEFGHIJKLMNOPQRSTUVWXYZ    _\
20 20 20 5A 59 58 57 56 55 54 53 52 51 50 4F  0014F
                                          5F 20  0015E
4E 4D 4C 4B 4A 49 48 47 46 45 44 43 42 41 20  00160          .ASCII  \ ABCDEFGHIJKLMNOPQRSTUVWXYZ     \
20 20 20 5A 59 58 57 56 55 54 53 52 51 50 4F  0016F
                                          20 20  0017E
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20  00180          .ASCII  \                                \
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20  0018F
                                          20 20  0019E
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20  001A0          .ASCII  \                                \
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20  001AF
                                          20 20  001BE
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20  001C0          .ASCII  \                                \
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20  001CF
                                          20 20  001DE
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20  001E0          .ASCII  \                                \
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20  001EF
                                          20 20  001FE
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  00200 P.AAC:  .ASCII  \@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@\
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  0020F
                                          40 40  0021E
40 2D 40 40 40 40 40 40 40 40 24 40 40 40 40  00220          .ASCII  \@@@@$@@@@@@@@-@@0123456789@@@@@@\
40 40 40 40 39 38 37 36 35 34 33 32 31 30 40  0022F
                                          40 40  0023E
4E 4D 4C 4B 4A 49 48 47 46 45 44 43 42 41 40  00240          .ASCII  \@ABCDEFGHIJKLMNOPQRSTUVWXYZ@@@@_\
40 40 40 5A 59 58 57 56 55 54 53 52 51 50 4F  0024F
                                          5F 40  0025E
4E 4D 4C 4B 4A 49 48 47 46 45 44 43 42 41 40  00260          .ASCII  \@ABCDEFGHIJKLMNOPQRSTUVWXYZ@@@@@\
40 40 40 5A 59 58 57 56 55 54 53 52 51 50 4F  0026F
                                          40 40  0027E
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  00280          .ASCII  \@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@\
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  0028F
                                          40 40  0029E
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  002A0          .ASCII  \@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@\
```

```
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 002AF
                                       40 40 002BE
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 002C0        .ASCII  \@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@\
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 002CF
                                       40 40 002DE
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 002E0        .ASCII  \@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@\
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 002EF
                                       40 40 002FE
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 00300 P.AAD: .ASCII  \@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@\
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 0030F
                                       40 40 0031E
40 2D 40 40 2A 40 40 40 40 25 24 40 40 40 40 00320        .ASCII  \@@@@$%@@@@*@@-@@0123456789@@@@@@\
40 40 40 40 39 38 37 36 35 34 33 32 31 30 40 0032F
                                       40 40 0033E
4E 4D 4C 4B 4A 49 48 47 46 45 44 43 42 41 40 00340        .ASCII  \@ABCDEFGHIJKLMNOPQRSTUVWXYZ@@@@_\
40 40 40 5A 59 58 57 56 55 54 53 52 51 50 4F 0034F
                                       5F 40 0035E
4E 4D 4C 4B 4A 49 48 47 46 45 44 43 42 41 40 00360        .ASCII  \@ABCDEFGHIJKLMNOPQRSTUVWXYZ@@@@@\
40 40 40 5A 59 58 57 56 55 54 53 52 51 50 4F 0036F
                                       40 40 0037E
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 00380        .ASCII  \@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@\
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 0038F
                                       40 40 0039E
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 003A0        .ASCII  \@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@\
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 003AF
                                       40 40 003BE
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 003C0        .ASCII  \@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@\
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 003CF
                                       40 40 003DE
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 003E0        .ASCII  \@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@\
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 003EF
                                       40 40 003FE

                                                   ANSI_A_BAD==          P.AAA
                                                   ANSI_A_GOOD==         P.AAB
                                                   ESC_CHAR==            P.AAA
                                                   VMS_NO_WILD=          P.AAC
                                                   VMS_WILD=             P.AAD
                                                          .EXTRN  LOCAL_FIB

                                        0FFC 00000        .ENTRY  PARSE_NAME_TYPE, Save R2,R3,R4,R5,R6,R7,R8,-   ; 0541
                                                                  R9,R10,R11
              58      10  AC        04  C1 00002          ADDL3   #4, OUT_DESC, R8                               ; 0601
                          07    04  AC  E9 00007          BLBC    WILD_CARDS, 1$                                 ; 0605
                          5B  FEF1  CF  9E 0000B          MOVAB   VMS_WILD, TABLE
                                    05  11 00010          BRB     2$
                          5B  FDEA  CF  9E 00012 1$:      MOVAB   VMS_NO_WILD, TABLE
                          57    0C  AC  D0 00017 2$:      MOVL    SPEC_PTR, R7                                   ; 0616
              67      08  AC        2E  3A 0001B          LOCC    #46, SPEC_LEN, (R7)
                                    02  12 00020          BNEQ    3$
                                    51  D4 00022          CLRL    R1
                          5A        51  D0 00024 3$:      MOVL    R1, IN_INDEX
                                    06  12 00027          BNEQ    4$                                             ; 0617
                          5A    08  AC  D0 00029          MOVL    SPEC_LEN, IN_INDEX                             ; 0618
                                    03  11 0002D          BRB     5$
                          5A        57  C2 0002F 4$:      SUBL2   R7, IN_INDEX                                   ; 0619
                          27        5A  D1 00032 5$:      CMPL    IN_INDEX, #39                                  ; 0623
```

```
                                              66  14 00035       BGTR    13$                                       ;
      6B     FBC3   CF            67          5A  2F 00037       MOVTUC  IN_INDEX, (R7), ESC_CHAR, (TABLE), -      ; 0629
                           00     B8          5A     0003E               IN_INDEX, @0(R8)                          ;
                                              02  1D 00041       BVS     6$                                        ;
                                              51  D4 00043       CLRL    R1                                        ;
                                              51  D5 00045 6$:   TSTL    R1                                        ; 0628
                                              6E  12 00047       BNEQ    15$                                       ;
                                  0F     04   AC  E9 00049       BLBC    WILD_CARDS, 7$                            ; 0635
                    09     0000G  CF          05  E1 0004D       BBC     #5, LOCAL_FIB+20, 7$                      ;
                           00     B8          2A  90 00053       MOVB    #42, @0(R8)                               ; 0638
                                  56          01  D0 00057       MOVL    #1, OUT_INDEX                             ; 0639
                                              03  11 0005A       BRB     8$                                        ; 0635
                                  56          5A  D0 0005C 7$:   MOVL    IN_INDEX, OUT_INDEX                       ; 0641
                           00   B846          2E  90 0005F 8$:   MOVB    #46, @0(R8)[OUT_INDEX]                    ; 0645
                                              56  D6 00064       INCL    OUT_INDEX                                 ; 0655
                                              5A  D6 00066       INCL    IN_INDEX                                  ; 0656
                    52     08     AC          5A  C3 00068       SUBL3   IN_INDEX, SPEC_LEN, REM_LEN               ; 0660
                    53            57          5A  C1 0006D       ADDL3   IN_INDEX, R7, R3                          ; 0661
                    63            52          20  3A 00071       LOCC    #32, REM_LEN, (R3)                        ;
                                              02  12 00075       BNEQ    9$                                        ;
                                              51  D4 00077       CLRL    R1                                        ;
                                  59          51  D0 00079 9$:   MOVL    R1, TYPE_LEN                              ;
                                              05  12 0007C       BNEQ    10$                                       ; 0662
                                  59          52  D0 0007E       MOVL    REM_LEN, TYPE_LEN                         ; 0663
                                              03  11 00081       BRB     11$                                       ;
                                  59          53  C2 00083 10$:  SUBL2   R3, TYPE_LEN                              ; 0664
                                  27          59  D1 00086 11$:  CMPL    TYPE_LEN, #39                             ; 0668
                                              2C  14 00089       BGTR    15$                                       ;
                                  52          59  C2 0008B       SUBL2   TYPE_LEN, REM_LEN                         ; 0672
                    50            5A          59  C1 0008E       ADDL3   TYPE_LEN, IN_INDEX, R0                    ; 0674
                  6047            52          20  3B 00092       SKPC    #32, REM_LEN, (R0)[R7]                    ;
                                              02  12 00097       BNEQ    12$                                       ;
                                              51  D4 00099       CLRL    R1                                        ;
                                              51  D5 0009B 12$:  TSTL    R1                                        ;
                                              18  12 0009D 13$:  BNEQ    15$                                       ;
                    52     10     BC          56  C3 0009F       SUBL3   OUT_INDEX, @OUT_DESC, REM_LEN             ; 0680
      6B     FB56   CF            63          59  2F 000A4       MOVTUC  TYPE_LEN, (R3), ESC_CHAR, (TABLE), -      ; 0682
                           00   B846          52     000AB               REM_LEN, @0(R8)[OUT_INDEX]                ;
                                              02  1D 000AF       BVS     14$                                       ;
                                              51  D4 000B1       CLRL    R1                                        ;
                                              51  D5 000B3 14$:  TSTL    R1                                        ; 0681
                                              06  13 000B5       BEQL    16$                                       ;
                                  50   0818   8F  3C 000B7 15$:  MOVZWL  #2072, R0                                 ; 0683
                                                  04 000BC       RET                                               ;
                                  12     04   AC  E9 000BD 16$:  BLBC    WILD_CARDS, 17$                           ; 0688
                    0C     0000G  CF          04  E1 000C1       BBC     #4, LOCAL_FIB+20, 17$                     ;
                           00   B846          2A  90 000C7       MOVB    #42, @0(R8)[OUT_INDEX]                    ; 0691
                           10     BC     01   A6  9E 000CC       MOVAB   1(R6), @OUT_DESC                          ; 0692
                                              05  11 000D1       BRB     18$                                       ; 0688
             10     BC            56          59  C1 000D3 17$:  ADDL3   TYPE_LEN, OUT_INDEX, @OUT_DESC            ; 0694
                                  50          01  D0 000D8 18$:  MOVL    #1, R0                                    ; 0696
                                                  04 000DB       RET                                               ; 0697

; Routine Size:  220 bytes,    Routine Base:  $CODE$ + 0400
```

```
GLOBAL ROUTINE PARSE_QUOTED_NAME ( SPEC_LEN,
                                   SPEC_PTR,
                                   OUT_DESC ) : COMMON_CALL NOVALUE =

!++
!
! FUNCTIONAL DESCRIPTION:
!       Parses a quoted name string.  It is ASSUMED that the string has had
!       the outer quotes stripped away ( STRIP_VERSION does this ).  This
!       routine will check all the characters in the string are valid ANSI "a",
!       will compress double quotes to single quotes and check that the final
!       string is no larger then FILE_SPEC_MAX characters.  It will error
!       exit if something is found to be incorrect.
!
! CALLING SEQUENCE:
!       PARSE_QUOTED_NAME ( ARG1, ARG2, ARG3 )
!
! INPUT PARAMETERS:
!       ARG1 - The size of the input string
!       ARG2 - The address of the input string
!
! IMPLICIT INPUTS:
!       none
!
! OUTPUT PARAMETERS:
!       ARG3 - The address of a descriptor pointing to the output buffer.  The
!              size field is updated.
!              ASSUMED OUTPUT BUFFER SIZE >=  FILE_SPEC_MAX BYTES
!
! IMPLICIT OUTPUTS:
!       none
!
! ROUTINE VALUE:
!       none
!
! SIDE EFFECTS:
!       none
!
!--

    BEGIN

    EXTERNAL REGISTER                                   ! so we can call ERR_EXIT
        COMMON_REG;

    MAP
        SPEC_LEN     : LONG,
        SPEC_PTR     : REF VECTOR [ , BYTE ],
        OUT_DESC     : REF VECTOR [ 2, LONG ];

    LOCAL
        IN_INDEX     : LONG,                   ! index into the input string
        MOVE_SIZE    : LONG,                   ! number of characters to be moved
        QUOTE_CHAR   : REF VECTOR [ , BYTE ];! location of a quota character

    BIND
        OUT_INDEX    = OUT_DESC [ 0 ],
```

```
2            OUT_BUF      = OUT_DESC [ 1 ] : REF VECTOR [ , BYTE ];
2
2        ! initialize the indices
2        !
2        IN_INDEX  = 0;
2        OUT_INDEX = 0;
2
2        ! loop for each string ending with a quote
2        !
2        WHILE NOT CH$FAIL (
2                 QUOTE_CHAR = CH$FIND_CH ( .SPEC_LEN, SPEC_PTR [ .IN_INDEX ], '"' )
2                                                                        ) DO
3            BEGIN
3
3            ! calculate the number of characters to be moved
3            ! and test for output buffer overflow
3            !
3            MOVE_SIZE = ( .QUOTE_CHAR - SPEC_PTR [ .IN_INDEX ] ) + 1;
3            IF ( .MOVE_SIZE + .OUT_INDEX ) GTR FILE_SPEC_MAX
3            THEN ERR_EXIT ( SS$_BADFILENAME );
3
3            ! adjust the number of characters remaining
3            ! check that the next character is there and is a quote
3            !
3            SPEC_LEN  = .SPEC_LEN - ( .MOVE_SIZE + 1 );
4            IF ( .SPEC_LEN LSS 0 ) OR ( .QUOTE_CHAR [ 1 ] NEQ '"' )
3            THEN ERR_EXIT ( SS$_BADFILENAME );
3
3            ! move and test for invalid the characters   Including the quote
3            ! character in the move
3            !
3            IF 0 NEQ MOVTUC ( MOVE_SIZE, SPEC_PTR [ .IN_INDEX  ], ESC_CHAR,
3                    ANSI_A_BAD, MOVE_SIZE, OUT_BUF  [ .OUT_INDEX ] )
3            THEN ERR_EXIT ( SS$_BADFILENAME );
3
3            ! adjust the indices
3            !
3            IN_INDEX  = .IN_INDEX  + .MOVE_SIZE + 1;
3            OUT_INDEX = .OUT_INDEX + .MOVE_SIZE;
2            END;
2
2        ! test for output buffer overflow
2        !
2        IF ( .SPEC_LEN + .OUT_INDEX ) GTR FILE_SPEC_MAX
2        THEN ERR_EXIT ( SS$_BADFILENAME );
2
2        ! move and check for invalid character
2        !
2        IF 0 NEQ MOVTUC ( SPEC_LEN, SPEC_PTR [ .IN_INDEX  ], ESC_CHAR, ANSI_A_BAD,
2                          SPEC_LEN, OUT_BUF  [ .OUT_INDEX ] )
2        THEN ERR_EXIT ( SS$_BADFILENAME );
2        OUT_INDEX = .OUT_INDEX + .SPEC_LEN;
2
1        END;                                   ! end of routine PARSE_QUOTED_NAME
```

```
                                     07FC 00000          .ENTRY  PARSE_QUOTED_NAME, Save R2,R3,R4,R5,R6,R7,-  ; 0698
                                                                 R8,R9,R10
                        57       0C   AC   D0 00002          MOVL    OUT_DESC, R7                            ; 0754
                        5A       04   A7   9E 00006          MOVAB   4(R7), R10                              ; 0755
                                      56   D4 0000A          CLRL    IN_INDEX                                ; 0759
                                      67   D4 0000C          CLRL    (R7)                                    ; 0760
              52        56       08   AC   C1 0000E 1$:      ADDL3   SPEC_PTR, IN_INDEX, R2                  ; 0765
              62   04   AC            22   3A 00013          LOCC    #34, SPEC_LEN, (R2)
                                      02   12 00018          BNEQ    2$
                                      51   D4 0001A          CLRL    R1
                        59            51   D0 0001C 2$:      MOVL    R1, QUOTE_CHAR
                                      56   13 0001F          BEQL    8$                                      ; 0766
              51        59            52   C3 00021          SUBL3   R2, QUOTE_CHAR, R1                      ; 0772
                        58       01   A1   9E 00025          MOVAB   1(R1), MOVE_SIZE
              51        58            67   C1 00029          ADDL3   (R7), MOVE_SIZE, R1                     ; 0773
                 0000004F 8F          51   D1 0002D          CMPL    R1, #79
                                      04   15 00034          BLEQ    3$
                             0818     8F   BF 00036          CHMU    #2072                                   ; 0774
              50   04   AC            58   C3 0003A 3$:      SUBL3   MOVE_SIZE, SPEC_LEN, R0                 ; 0779
                   04   AC       FF   A0   9E 0003F          MOVAB   -1(R0), SPEC_LEN
                                      06   19 00044          BLSS    4$                                      ; 0780
                        22       01   A9   91 00046          CMPB    1(QUOTE_CHAR), #34
                                      04   13 0004A          BEQL    5$
                             0818     8F   BF 0004C 4$:      CHMU    #2072                                   ; 0781
              50        6A            67   C1 00050 5$:      ADDL3   (R7), (R10), R0                         ; 0787
FAC5   CF   FAC8   CF   08 BC46       58   2F 00054          MOVTUC  MOVE_SIZE, @SPEC_PTR[IN_INDEX], ESC_CHAR, -
                        60            58      0005F                  ANSI_A_BAD, MOVE_SIZE, (R0)
                                      02   1D 00061          BVS     6$
                                      51   D4 00063          CLRL    R1
                                      51   D5 00065 6$:      TSTL    R1                                      ; 0786
                                      04   13 00067          BEQL    7$
                             0818     8F   BF 00069          CHMU    #2072                                   ; 0788
                        56      01 A846    9E 0006D 7$:      MOVAB   1(MOVE_SIZE)[IN_INDEX], IN_INDEX        ; 0792
                        67            58   C0 00072          ADDL2   MOVE_SIZE, (R7)                         ; 0793
                                      97   11 00075          BRB     1$                                      ; 0764
              50   04   AC            67   C1 00077 8$:      ADDL3   (R7), SPEC_LEN, R0                      ; 0798
                 0000004F 8F          50   D1 0007C          CMPL    R0, #79
                                      04   15 00083          BLEQ    9$
                             0818     8F   BF 00085          CHMU    #2072                                   ; 0799
              50        6A            67   C1 00089 9$:      ADDL3   (R7), (R10), R0                         ; 0804
FA8B   CF   FA8E   CF   08 BC46  04   AC   2F 0008D          MOVTUC  SPEC_LEN, @SPEC_PTR[IN_INDEX], ESC_CHAR, -
                        60       04   AC      00099                  ANSI_A_BAD, SPEC_LEN, (R0)
                                      02   1D 0009C          BVS     10$
                                      51   D4 0009E          CLRL    R1
                                      51   D5 000A0 10$:     TSTL    R1                                      ; 0803
                                      04   13 000A2          BEQL    11$
                             0818     8F   BF 000A4          CHMU    #2072                                   ; 0805
                        67       04   AC   C0 000A8 11$:     ADDL2   SPEC_LEN, (R7)                          ; 0806
                                           04 000AC          RET                                             ; 0808

; Routine Size:  173 bytes,    Routine Base:  $CODE$ + 04DC
```

```
1 GLOBAL ROUTINE STRIP_VERSION ( FILE_SPEC_LEN,
1                                FILE_SPEC_PTR,
1                                WILD_VERSION,
1                                QUOTED_NAME
1                              ) : COMMON_CALL =
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !       This routine strips the version number from a ASCII VMS file
1 !        specification, and if the file spec contains a quoted name the outer
1 !        quotes are stripped by adjusting the pointer and size parameters and
1 !        a check is make for a null type field.
1 !
1 ! CALLING SEQUENCE:
1 !       version_number = STRIP_VERSION ( ARG1, ARG2, ARG3, ARG4 )
1 !
1 ! INPUT PARAMETERS:
1 !       ARG1 - the length of the file specification
1 !       ARG2 - the address of the address of the file specification buffer
1 !       ARG3 - flag set if wild card version is allowed
1 !
1 ! IMPLICIT INPUTS:
1 !       none
1 !
1 ! OUTPUT PARAMETERS:
1 !       ARG1 - the length of the file specification minus the version number
1 !               and it delimiter
1 !       ARG2 - the address of the file specification
1 !       ARG4 - A flag set if a quoted name was found
1 !
1 ! IMPLICIT OUTPUTS:
1 !       none
1 !
1 ! ROUTINE VALUE:
1 !       the version number
1 !
1 ! SIDE EFFECTS:
1 !       none
1 !
1 !--
1
2     BEGIN
2
2     EXTERNAL ROUTINE                              ! ASCII decimal to binary
2         LIB$CVT_DTB     : ADDRESSING_MODE ( ABSOLUTE );
2
2     EXTERNAL REGISTER                             ! so we can call ERR_EXIT
2         COMMON_REG;
2
2     MAP
2         FILE_SPEC_LEN   : REF VECTOR [ 1, LONG ],! the size of the file spec
2         FILE_SPEC_PTR   : REF VECTOR [ 1, LONG ],! ptr to the file spec ptr
2         WILD_VERSION    : BITVECTOR  [ 1 ],      ! wild version allowed flag
2         QUOTED_NAME     : REF BITVECTOR  [ 1 ]; ! quoted name, return flag
2
2     BIND
```

```
2            FILE_SPEC_BUF   = FILE_SPEC_PTR [ 0 ] : REF VECTOR [ , BYTE ];
2                                                ! point to the file spec buffer
2
2        LOCAL
2            QUOTED_NAME_LEN : LONG,             ! length of the quoted string
2            QUOTED_NAME_PTR : LONG,             ! address of the quoted string
2            VERSION         : LONG,             ! the version number
2            VERSION_PTR     : REF VECTOR [ , BYTE ],! ptr to the ASCII version
2            VERSION_LEN     : LONG;             ! the size of ASCII version
2
2
2        ! test for the quoted string in the name field
2        !
2        QUOTED_NAME [ 0 ] = FALSE;
3        IF ( .FILE_SPEC_LEN [ 0 ] GEQ 2 ) AND ( .FILE_SPEC_BUF [ 0 ] EQL '"' )
2        THEN
3            BEGIN
3            LABEL FIND_END_QUOTE;
3
3            FIND_END_QUOTE:
4                BEGIN
4
4                ! Loop from the end of the string forward until a '"' is found
4                ! Note: if the last character is " the name is invalid because
4                !       at least a version delimitor is needed
4                !
4                DECR I FROM ( .FILE_SPEC_LEN [ 0 ] - 2 ) TO 1 DO
4                    IF .FILE_SPEC_BUF [ .I ] EQL '"'
4                    THEN
5                        BEGIN
5
5                        ! found a quoted string save the info and set FILE_SPEC_xxx
5                        !  to point to the rest of the string
5                        !
5                        QUOTED_NAME [ 0 ]   = TRUE;
5                        QUOTED_NAME_PTR     = FILE_SPEC_BUF [ 1 ];
5                        QUOTED_NAME_LEN     = .I - 1;
5                        FILE_SPEC_PTR [ 0 ] = FILE_SPEC_BUF [ .I + 1 ];
5                        FILE_SPEC_LEN [ 0 ] = .FILE_SPEC_LEN [ 0 ] - ( .I + 1 );
5                        LEAVE FIND_END_QUOTE;
4                        END;
4
4                ! Never found the end quote so err_exit
4                !
4                ERR_EXIT ( SS$_BADFILENAME );
3                END;
3
3            ! test for a null type field which is:
3            !   ;nnnnn
3            !   .;nnnnn         where nnnnn is zero or more digits
3            !   ..nnnnn
3            !
4            IF  ( .FILE_SPEC_LEN [ 0 ] GEQU 1 )
4            AND ( .FILE_SPEC_BUF [ 0 ] EQL ';')
3            THEN VERSION_PTR = FILE_SPEC_BUF [ 0 ]
3
4            ELSE IF ( .FILE_SPEC_LEN [ 0 ] GEQU 2 )
```

```
                AND ( .FILE_SPEC_BUF [ 0 ] EQL '.' )
                AND (     ( .FILE_SPEC_BUF [ 1 ] EQL '.')
                       OR ( .FILE_SPEC_BUF [ 1 ] EQL ';'))
                THEN VERSION_PTR = FILE_SPEC_BUF [ 1 ]
                ELSE ERR_EXIT ( SS$_BADFILENAME );
            END

        ELSE
            BEGIN

            ! We have a normal VMS file specification
            ! check for ";version number"
            !
            VERSION_PTR = CH$FIND_CH ( .FILE_SPEC_LEN [ 0 ],
                                       .FILE_SPEC_PTR [ 0 ],
                                       ';' );
            IF CH$FAIL ( .VERSION_PTR )
            THEN
                BEGIN

                ! now look for "..version number"
                !
                VERSION_PTR = CH$FIND_CH ( .FILE_SPEC_LEN [ 0 ],
                                           .FILE_SPEC_PTR [ 0 ],
                                           '.' );
                IF CH$FAIL ( .VERSION_PTR ) THEN ERR_EXIT ( SS$_BADFILENAME );

                ! move pass the delimitor and look for the second dot
                !
                VERSION_PTR = .VERSION_PTR + 1;
                VERSION_LEN = .FILE_SPEC_LEN [ 0 ] -
                                       ( .VERSION_PTR - .FILE_SPEC_PTR [ 0 ] );
                VERSION_PTR = CH$FIND_CH ( .VERSION_LEN, .VERSION_PTR, '.' );
                IF CH$FAIL ( .VERSION_PTR ) THEN ERR_EXIT ( SS$_BADFILEVER );
                END;
            END;

        ! move pass the delimitor and to the ASCII decimal number
        !
        VERSION_PTR = .VERSION_PTR + 1;

        ! ASCII to binary translation, if wildcards are allowed turn "*" into 0
        ! Note: a version number size of zero is valid and will be returned as zero
        ! Note: the version number size will never be less than zero
        !
        VERSION_LEN = .FILE_SPEC_LEN [ 0 ] -
                               ( .VERSION_PTR - .FILE_SPEC_PTR [ 0 ] );
        IF NOT LIB$CVT_DTB ( .VERSION_LEN, .VERSION_PTR, VERSION )
        THEN IF   .WILD_VERSION [ 0 ]
             AND ( .VERSION_LEN EQL 1 )
             AND ( .VERSION_PTR [ 0 ] EQL '*' )
             THEN VERSION = 0
             ELSE ERR_EXIT ( SS$_BADFILEVER );

        ! adjust the pointer and size fields of the name and type field
        !
        IF .QUOTED_NAME [ 0 ]
```

```
;  601      0980  2      THEN
;  602      0981  3          BEGIN
;  603      0982  3          !
;  604      0983  3          ! a quoted name return saved size and pointer
;  605      0984  3          !
;  606      0985  3          FILE_SPEC_LEN [ 0 ] = .QUOTED_NAME_LEN;
;  607      0986  3          FILE_SPEC_PTR [ 0 ] = .QUOTED_NAME_PTR;
;  608      0987  3          END
;  609      0988  3
;  610      0989  2      ELSE
;  611      0990  2          ! return the size of the file name and type minus the version number
;  612      0991  2          ! and its delimitor
;  613      0992  2          !
;  614      0993  2          FILE_SPEC_LEN [ 0 ] = .FILE_SPEC_LEN [ 0 ] - ( .VERSION_LEN + 1 );
;  615      0994  2
;  616      0995  2      ! return the version number
;  617      0996  2      !
;  618      0997  2      RETURN .VERSION;
;  619      0998  1      END;                                    ! end of routine STRIP_VERSION


                                                         .EXTRN  LIB$CVT_DTB

                               007C 00000                .ENTRY  STRIP_VERSION, Save R2,R3,R4,R5,R6               ; 0809
              5E            04   C2 00002                SUBL2   #4, SP
              53      08    AC   D0 00005                MOVL    FILE_SPEC_PTR, R3                                 ; 0866
              54            53   D0 00009                MOVL    R3, R4
        10    BC            01   8A 0000C                BICB2   #1, @QUOTED_NAME                                  ; 0879
              02      04    BC   D1 00010                CMPL    @FILE_SPEC_LEN, #2                                ; 0880
                            72   19 00014                BLSS    7$
              22      00    B4   91 00016                CMPB    @0(R4), #34
                            6C   12 0001A                BNEQ    7$
   50   04    BC            01   C3 0001C                SUBL3   #1, @FILE_SPEC_LEN, I                             ; 0892
                            27   11 00021                BRB     2$
              22      00 B440    91 00023 1$:            CMPB    @0(R4)[I], #34                                    ; 0893
                            20   12 00028                BNEQ    2$
        10    BC            01   88 0002A                BISB2   #1, @QUOTED_NAME                                  ; 0900
   56         64            01   C1 0002E                ADDL3   #1, (R4), QUOTED_NAME_PTR                         ; 0901
              52      FF    A0   9E 00032                MOVAB   -1(R0), QUOTED_NAME_LEN                           ; 0902
   51         64            50   C1 00036                ADDL3   I, (R4), R1                                       ; 0903
              63      01    A1   9E 0003A                MOVAB   1(R1), (R3)
   51   04    BC            50   C3 0003E                SUBL3   I, @FILE_SPEC_LEN, R1                             ; 0904
        04    BC      FF    A1   9E 00043                MOVAB   -1(R1), @FILE_SPEC_LEN
                            07   11 00048                BRB     3$                                                ; 0905
              D6            50   F5 0004A 2$:            SOBGTR  I, 1$                                             ; 0893
                   0818     8F   BF 0004D                CHMU    #2072                                             ; 0910
                      04    BC   D5 00051 3$:            TSTL    @FILE_SPEC_LEN                                    ; 0918
                            0B   13 00054                BEQL    4$
              3B      00    B4   91 00056                CMPB    @0(R4), #59                                       ; 0919
                            05   12 0005A                BNEQ    4$
              55            64   D0 0005C                MOVL    (R4), VERSION_PTR                                 ; 0920
                            66   11 0005F                BRB     12$
              02      04    BC   D1 00061 4$:            CMPL    @FILE_SPEC_LEN, #2                                ; 0922
                            1B   1F 00065                BLSSU   6$
              2E      00    B4   91 00067                CMPB    @0(R4), #46                                       ; 0923
                            15   12 0006B                BNEQ    6$
```

```
                    50                    64  D0 0006D         MOVL    (R4), R0                                  ; 0924
                    2E          01        A0  91 00070         CMPB    1(R0), #46                                ;
                    06                    13  00074            BEQL    5$                                        ;
                    3B          01        A0  91 00076         CMPB    1(R0), #59                                ; 0925
                    06                    12  0007A            BNEQ    6$                                        ;
          55        64                    01  C1 0007C  5$:    ADDL3   #1, (R4), VERSION_PTR                     ; 0926
                                          45  11 00080         BRB     12$                                       ;
                              0818        8F  BF 00082  6$:    CHMU    #2072                                     ; 0927
                                          3F  11 00086         BRB     12$                                       ; 0880
    00    B3          04    BC            3B  3A 00088  7$:    LOCC    #59, @FILE_SPEC_LEN, @0(R3)               ; 0936
                                          02  12 0008E         BNEQ    8$                                        ;
                                          51  D4 00090         CLRL    R1                                        ;
          55                              51  D0 00092  8$:    MOVL    R1, VERSION_PTR                           ;
                                          30  12 00095         BNEQ    12$                                       ; 0939
    00    B3          04    BC            2E  3A 00097         LOCC    #46, @FILE_SPEC_LEN, @0(R3)               ; 0945
                                          02  12 0009D         BNEQ    9$                                        ;
                                          51  D4 0009F         CLRL    R1                                        ;
          55                              51  D0 000A1  9$:    MOVL    R1, VERSION_PTR                           ;
                                          04  12 000A4         BNEQ    10$                                       ; 0948
                              0818        8F  BF 000A6         CHMU    #2072                                     ;
                                          55  D6 000AA  10$:   INCL    VERSION_PTR                               ; 0952
          50          08    BC            55  C3 000AC         SUBL3   VERSION_PTR, @FILE_SPEC_PTR, R0           ; 0954
          54                50      04    BC  C1 000B1         ADDL3   @FILE_SPEC_LEN, R0, VERSION_LEN           ;
          65                54            2E  3A 000B6         LOCC    #46, VERSION_LEN, (VERSION_PTR)           ; 0955
                                          02  12 000BA         BNEQ    11$                                       ;
                                          51  D4 000BC         CLRL    R1                                        ;
          55                              51  D0 000BE  11$:   MOVL    R1, VERSION_PTR                           ;
                                          04  12 000C1         BNEQ    12$                                       ; 0956
                              0820        8F  BF 000C3         CHMU    #2080                                     ;
                                          55  D6 000C7  12$:   INCL    VERSION_PTR                               ; 0962
          50          08    BC            55  C3 000C9         SUBL3   VERSION_PTR, @FILE_SPEC_PTR, R0           ; 0969
          54                50      04    BC  C1 000CE         ADDL3   @FILE_SPEC_LEN, R0, VERSION_LEN           ;
                              4030        8F  BB 000D3         PUSHR   #^M<R4,R5,SP>                             ; 0970
               00000000G    9F            03  FB 000D7         CALLS   #3, @#LIB$CVT_DTB                         ;
                            16            50  E8 000DE         BLBS    R0, 14$                                   ;
                            0E      0C    AC  E9 000E1         BLBC    WILD_VERSION, 13$                         ; 0971
                            01            54  D1 000E5         CMPL    VERSION_LEN, #1                           ; 0972
                                          09  12 000E8         BNEQ    13$                                       ;
                            2A            65  91 000EA         CMPB    (VERSION_PTR), #42                        ; 0973
                                          04  12 000ED         BNEQ    13$                                       ;
                                          6E  D4 000EF         CLRL    VERSION                                   ; 0974
                                          04  11 000F1         BRB     14$                                       ;
                              0820        8F  BF 000F3  13$:   CHMU    #2080                                     ; 0975
                            0A      10    BC  E9 000F7  14$:   BLBC    @QUOTED_NAME, 15$                         ; 0979
                      04    BC            52  D0 000FB         MOVL    QUOTED_NAME_LEN, @FILE_SPEC_LEN           ; 0985
                      08    BC            56  D0 000FF         MOVL    QUOTED_NAME_PTR, @FILE_SPEC_PTR           ; 0986
                                          0A  11 00103         BRB     16$                                       ; 0979
          50          04    BC            54  C3 00105  15$:   SUBL3   VERSION_LEN, @FILE_SPEC_LEN, R0           ; 0993
                      04    BC      FF    A0  9E 0010A         MOVAB   -1(R0), @FILE_SPEC_LEN                    ;
                            50            6E  D0 0010F  16$:   MOVL    VERSION, R0                               ; 0997
                                              04 00112         RET                                               ; 0998

; Routine Size:  275 bytes,     Routine Base:  $CODE$ + 0589
```

```
1 GLOBAL ROUTINE CALC_VERSION ( VERSION ) : NOVALUE =
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !       This routine calculates version number from generation and generation
1 !        version.
1 !
1 ! CALLING SEQUENCE:
1 !       CALC_VERSION ( ARG1 )
1 !
1 ! INPUT PARAMETERS:
1 !       None
1 !
1 ! IMPLICIT INPUTS:
1 !       HDR1 - address of hdr1 label for current file
1 !
1 ! OUTPUT PARAMETERS:
1 !       ARG1 - address of two bytes to receive binary version number
1 !
1 ! IMPLICIT OUTPUTS:
1 !       None
1 !
1 ! ROUTINE VALUE:
1 !       None
1 !
1 ! SIDE EFFECTS:
1 !       ARG1 receives binary version number
1 !
1 !--
1
2    BEGIN
2
2    EXTERNAL
2       HDR1      : REF BBLOCK;                     ! addr of HDR1 ( EOF1 ) label
2
2    EXTERNAL ROUTINE
2       LIB$CVT_DTB : ADDRESSING_MODE ( ABSOLUTE ); ! convert decimal to binary
2
2    MAP
2       VERSION : REF BBLOCK [ 2 ];                 ! a pointer to a word
2
2    MACRO REF_WORD  = 0, 0, 16, 0 %;               ! ref to the 1st word
2    MACRO REF_15BIT = 0, 0, 15, 0 %;               ! ref to the 1st fifteen bits
2
2    LOCAL
2       GEN_NUM : LONG,                             ! binary generation number
2       GEN_VER : LONG;                             ! binary generation version
2
2    ! clear version #
2    !
2    VERSION [ REF_WORD ] = 0;
2
2    ! binary version # = (generation number - 1) * 100 + generation version + 1
2    ! convert generation number to binary
2    !
2    IF LIB$CVT_DTB ( HD1$S_GENNO, HDR1[ HD1$T_GENNO ], GEN_NUM )
```

```
;  678       1056  2      THEN
;  679       1057  2
;  680       1058  2          ! convert gen version number to binary
;  681       1059  2          !
;  682       1060  2          IF LIB$CVT_DTB ( HD1$S_GENVER, HDR1 [ HD1$T_GENVER ], GEN_VER )
;  683       1061  2          THEN
;  684       1062  2
;  685       1063  2              ! NOTE: The HIGH BIT in the VERSION NUMBER is always ZERO
;  686       1064  2              !
;  687       1065  2              VERSION [ REF_15BIT ] = ( .GEN_NUM - 1 ) * 100 + .GEN_VER + 1;
;  688       1066  2
;  689       1067  1      END;                                     ! end of routine CALC_VERSION
```

```
                                                            .EXTRN  HDR1

                                          0004 00000        .ENTRY  CALC_VERSION, Save R2                      ; 0999
                  52 00000000G    9F   9E 00002             MOVAB   @#LIB$CVT_DTB, R2                          ;
                  5E              08   C2 00009             SUBL2   #8, SP                                     ;
                           04     BC   B4 0000C             CLRW    @VERSION                                   ; 1050
                                  5E   DD 0000F             PUSHL   SP                                         ; 1055
          7E      0000G   CF      23   C1 00011             ADDL3   #35, HDR1, -(SP)                           ;
                                  04   DD 00017             PUSHL   #4                                         ;
                  62              03   FB 00019             CALLS   #3, LIB$CVT_DTB                            ;
                  27              50   E9 0001C             BLBC    R0, 1$                                     ;
                           04     AE   9F 0001F             PUSHAB  GEN_VER                                    ; 1060
          7E      0000G   CF      27   C1 00022             ADDL3   #39, HDR1, -(SP)                           ;
                                  02   DD 00028             PUSHL   #2                                         ;
                  62              03   FB 0002A             CALLS   #3, LIB$CVT_DTB                            ;
                  16              50   E9 0002D             BLBC    R0, 1$                                     ;
          50      6E 00000064     8F   C5 00030             MULL3   #100, GEN_NUM, R0                          ; 1065
                  50       04     AE   C0 00038             ADDL2   GEN_VER, R0                                ;
                  51       9D     A0   9E 0003C             MOVAB   -99(R0), R1                                ;
   04   BC        0F      00      51   F0 00040             INSV    R1, #0, #15, @VERSION                      ;
                                  04 00046 1$:              RET                                                ; 1067

; Routine Size:  71 bytes,    Routine Base:  $CODE$ + 069C
```

```
GLOBAL ROUTINE PARSE_FID ( FILE_SPEC_DESC,
                           FILE_VERSION     ) : COMMON_CALL =

!++
!
! FUNCTIONAL DESCRIPTION:
!       This routine parses the File Identifier Field in HDR1 and HDR4, then
!       calculates the version number from the generation fields.
!
! CALLING SEQUENCE:
!       vms_spec = PARSE_FID ( ARG1, ARG2 )
!
! INPUT PARAMETERS:
!       ARG1 - descriptor to file specification output buffer
!               ASSUMED OUTPUT BUFFER SIZE >= FILE_SPEC_MAX BYTES
!       ARG2 - pointer to a word to hold the version number
!
! IMPLICIT INPUTS:
!       HDR1 and HDR4 label
!
! OUTPUT PARAMETERS:
!       ARG1 - descriptor to file specification output buffer
!       ARG2 - pointer to a word to hold the version number
!
! IMPLICIT OUTPUTS:
!       none
!
! ROUTINE VALUE:
!       TRUE  - if the spec in HDR1 is a "VMS type specification"
!       FALSE - otherwise
!
! SIDE EFFECTS:
!       none
!
!--
    BEGIN

    EXTERNAL REGISTER
        COMMON_REG;

    EXTERNAL ROUTINE                                ! ASCII decimal to binary
        LIB$CVT_DTB     : ADDRESSING_MODE ( ABSOLUTE );

    EXTERNAL
        HDR1    : REF BBLOCK,                       ! addr of HDR1 ( EOF1 ) label
        HDR4    : REF BBLOCK;                       ! addr of HDR4 ( EOF4 ) label

    LOCAL
        MVL     : REF BBLOCK,
        FILE_ID : VECTOR [FILE_SPEC_MAX,BYTE];

    MAP
        FILE_SPEC_DESC  : REF VECTOR [ 2, LONG ],
        FILE_VERSION    : REF VECTOR [ 1, WORD ];

    ! calculate the version number
```

```
CALC_VERSION ( .FILE_VERSION );

! Space fill the storage for the FILE_ID

CH$FILL(' ',FILE_SPEC_MAX,FILE_ID);

! Copy the file name from the HDR1 and HDR4 label into one common storage
! area for the various checks and the translation. The HDR4 label contains
! part of the FILE ID if the FILEID EXT SIZE is gtr then 0.
! Please note that the size of the FILEID extension stored in the HDR4 field
! is stored in different locations depending on the ANSI version stored
! in the VOL1 label.

! First copy the HDR1 file id

CH$MOVE(HD1$S_FILEID, HDR1[HD1$T_FILEID], FILE_ID[0]);

! The get the HDR4 file id extension

MVL  = .CURRENT_VCB[VCB$L_MVL];
IF .MVL[MVL$B_STDVER] GTR 3
    THEN
    BEGIN
        IF .HDR4[HD4$B_FILEID_EXT_SIZE] GTRU 0
            THEN
                CH$MOVE(.HDR4[HD4$B_FILEID_EXT_SIZE], HDR4[HD4$T_FILEID_EXT],
                                  FILE_ID[HD1$S_FILEID]);
    END
    ELSE
    BEGIN
        LOCAL SIZE;
        IF NOT LIB$CVT_DTB(HD4$S_FILEID_EXT_V3,
                           HDR4[HD4$T_FILEID_EXT_V3], SIZE)
            THEN SIZE = 0
            ELSE
               CH$MOVE(.SIZE, HDR4[HD4$T_FILEID_EXT],
                        FILE_ID[HD1$S_FILEID]);
    END;

! Parse the file id field in the HDR1

IF NOT PARSE_NAME_TYPE ( FALSE, ! do not allow wild cards
                         FILE_SPEC_MAX,
                         FILE_ID[0],
                         .FILE_SPEC_DESC )
THEN

    ! if the parse failed then it is a ANSI name which does not conform to
    !  VMS file spec standards so... move the field from the header to the
    !  output buffer
    !
    BEGIN
    CH$TRANSLATE ( ANSI_A_GOOD, FILE_SPEC_MAX, FILE_ID[0], ' ',
                        .FILE_SPEC_DESC [ 0 ], .FILE_SPEC_DESC [ 1 ] );
    FILE_SPEC_DESC [ 0 ] = FILE_SPEC_MAX;
    RETURN FALSE;
    END
```

```
;  805        1182  3
;  806        1183  2        ELSE RETURN TRUE;
;  807        1184  2
;  808        1185  1        END;                                 ! end of routine PARSE_HDR1_FID


                                                                 .EXTRN  HDR4

                                          007C 00000             .ENTRY  PARSE_FID, Save R2,R3,R4,R5,R6          ; 1068
                         5E       AC   AE  9E 00002             MOVAB   -84(SP), SP
                                  08   AC  DD 00006             PUSHL   FILE_VERSION                             ; 1125
                   AC    AF            01  FB 00009             CALLS   #1, CALC_VERSION
  004F    8F       20    6E            00  2C 0000D             MOVC5   #0, (SP), #32, #79, FILE_ID              ; 1129
                         04            AE     00014
                         50    0000G   CF  D0 00016             MOVL    HDR1, R0                                 ; 1140
          04   AE  04    A0            11  28 0001B             MOVC3   #17, 4(R0), FILE_ID
                         51       34   AB  D0 00021             MOVL    52(CURRENT_VCB), MVL                     ; 1144
                         50    0000G   CF  D0 00025             MOVL    HDR4, R0                                 ; 1148
                         03       22   A1  91 0002A             CMPB    34(MVL), #3                              ; 1145
                                       11  1B 0002E             BLEQU   1$
                                  04   A0  95 00030             TSTB    4(R0)                                    ; 1148
                                       2C  13 00033             BEQL    3$
                         51       04   A0  9A 00035             MOVZBL  4(R0), R1                                ; 1150
          15   AE  05    A0            51  28 00039             MOVC3   R1, 5(R0), FILE_ID+17                    ; 1151
                                       20  11 0003F             BRB     3$                                       ; 1145
                                       5E  DD 00041 1$:         PUSHL   SP                                       ; 1157
                                  43   A0  9F 00043             PUSHAB  67(R0)
                                       02  DD 00046             PUSHL   #2
                 00000000G  9F         03  FB 00048             CALLS   #3, @#LIB$CVT_DTB
                            04         50  E8 0004F             BLBS    R0, 2$
                                       6E  D4 00052             CLRL    SIZE                                     ; 1158
                                       0B  11 00054             BRB     3$
                         50    0000G   CF  D0 00056 2$:         MOVL    HDR4, R0                                 ; 1160
          15   AE  05    A0            6E  28 0005B             MOVC3   SIZE, 5(R0), FILE_ID+17                  ; 1161
                         56       04   AC  D0 00061 3$:         MOVL    FILE_SPEC_DESC, R6                       ; 1169
                                       56  DD 00065             PUSHL   R6
                                  08   AE  9F 00067             PUSHAB  FILE_ID                                  ; 1168
                         7E       4F   8F  9A 0006A             MOVZBL  #79, -(SP)
                                       7E  D4 0006E             CLRL    -(SP)
                   FCA8  CF            04  FB 00070             CALLS   #4, PARSE_NAME_TYPE
                         14            50  E8 00075             BLBS    R0, 4$
  F99B    CF       20    04    004F    8F  2E 00078             MOVTC   #79, FILE_ID, #32, ANSI_A_GOOD, (R6), -  ; 1178
                   04    B6            66     00082                     @4(R6)
                         66       4F   8F  9A 00085             MOVZBL  #79, (R6)                                ; 1179
                                       50  D4 00089             CLRL    R0                                       ; 1183
                                       04     0008B             RET
                         50            01  D0 0008C 4$:         MOVL    #1, R0
                                       04     0008F             RET                                              ; 1185

; Routine Size:  144 bytes,    Routine Base:  $CODE$ + 06E3
```

```
GLOBAL ROUTINE PARSE_PATTERN_SPEC ( FILE_SPEC_DESC,
                                    FILE_VERSION
                                  ) : COMMON_CALL =

!++
!
! FUNCTIONAL DESCRIPTION:
!       This routine parses a string given to the ACP by the user which is
!        used as a pattern string matching string by the find operation
!
! CALLING SEQUENCE:
!       vms_spec = PARSE_PATTERN_SPEC ( ARG1, ARG2 )
!
! INPUT PARAMETERS:
!       ARG1 - descriptor to file specification output buffer
!               ASSUMED OUTPUT BUFFER SIZE >= FILE_SPEC_MAX BYTES
!       ARG2 - pointer to a word to hold the version number
!
! IMPLICIT INPUTS:
!       IO_PACKET
!
! OUTPUT PARAMETERS:
!       ARG1 - descriptor to file specification output buffer
!       ARG2 - pointer to a word to hold the version number
!
! IMPLICIT OUTPUTS:
!       none
!
! ROUTINE VALUE:
!       Bit 0 set - if the spec is wild for quoted name. ie "*.*" or "*."
!       Bit 1 set - if it was a VMS spec
!
! SIDE EFFECTS:
!       none
!
!--
    BEGIN

    EXTERNAL REGISTER                                   ! so we can call ERR_EXIT
        COMMON_REG;

    EXTERNAL
        IO_PACKET       : REF BBLOCK;

    MAP
        FILE_SPEC_DESC  : REF VECTOR [ 2, LONG ],
        FILE_VERSION    : REF VECTOR [ 1, WORD ];

    LITERAL
        WILD_QUOTED     = %B'01',
        IS_VMS_SPEC     = %B'10';

    LOCAL
        ABD             : REF BBLOCKVECTOR [ , ABD$C_LENGTH ],
                                ! ACP buffer descriptors
        ASCII_NAME_SPEC : BITVECTOR [ 1 ] PRESET ( [ 0 ] = FALSE ),
                                ! flag set if using ASCNAME attribute
```

```
2            ATTR_PTR          : LONG,           ! pointer into the ACP buffer descr
2            ATTR_LENGTH       : LONG,           ! length of the attribute
2            INPUT_ADDR        : LONG,           ! address of input file name string
2            INPUT_LENGTH      : LONG,           ! length of input file name string
2            QUOTED_NAME       : BITVECTOR [ 1 ];! flag set if the file spec is in "s
2
2        ABD = .BBLOCK [ .IO_PACKET [ IRP$L_SVAPTE ], AIB$L_DESCRIPT ];
2
2        ! scan to see if the find should be done by the ASCII name attribute
2        !
2        DECRU I FROM ( .IO_PACKET[IRP$W_BCNT] - 1 ) TO ABD$C_ATTRIB DO
3            BEGIN
3
3            ! find the attribute code and see if it matches ASCII name
3            !
3            ATTR_PTR = .ABD [ .I, ABD$W_TEXT ] + ABD [ .I, ABD$W_TEXT ];
3            IF .(.ATTR_PTR)<0, 8> EQL ATR$C_ASCNAME
3            THEN
4                BEGIN
4
4                ! set up the size of the attribute
4                !
4                ATTR_LENGTH = .ABD[.I, ABD$W_COUNT];
4
4                ! check length of file name
4                !
4                IF .ATTR_LENGTH LEQ FILE_SPEC_MAX
4                THEN
5                    BEGIN
5
5                    ! set up the pointer to the attribute text
5                    !
5                    ATTR_PTR = .ATTR_PTR + 1;
5
5                    ! translate into upper case and check for invalid characters
5                    !
5                    IF 0 EQL MOVTUC( ATTR_LENGTH, .ATTR_PTR, ESC_CHAR, ANSI_A_BAD,
5                                     FILE_SPEC_DESC [ 0 ], .FILE_SPEC_DESC [ 1 ] )
5                    THEN
6                        BEGIN
6
6                        ! we have a valid ANSI file spec to the find operation on
6                        !
6                        ASCII_NAME_SPEC [ 0 ] = TRUE;
6                        FILE_SPEC_DESC [ 0 ] = .ATTR_LENGTH;
6                        EXITLOOP;
5                        END;
5
4                    END;
4
3                END;
3
2            END;
2
2        ! pick up the name pattern string from the name attribute
2        !
2        INPUT_LENGTH = .ABD [ ABD$C_NAME, ABD$W_COUNT ];
```

```
INPUT_ADDR    = .ABD [ ABD$C_NAME, ABD$W_TEXT ]
                + ABD [ ABD$C_NAME, ABD$W_TEXT ] + 1;

! strip the version number
!
FILE_VERSION [ 0 ] = STRIP_VERSION ( INPUT_LENGTH,
                                     INPUT_ADDR,
                                     TRUE,   ! allow wild cards
                                     QUOTED_NAME
                                   );

! parse the filename if not from ASCNAME
!
IF NOT .ASCII_NAME_SPEC [ 0 ]
THEN
    BEGIN

    ! call the correct parse routine
    !
    IF NOT .QUOTED_NAME [ 0 ]
    THEN
        BEGIN

        BIND
            STAR_DOT_STAR = UPLIT ( '*.*' ),
            STAR_DOT      = STAR_DOT_STAR;

        ! parse the filename into name and type
        !
        IF NOT PARSE_NAME_TYPE ( TRUE, ! allow wild cards
                                 .INPUT_LENGTH,
                                 .INPUT_ADDR,
                                 .FILE_SPEC_DESC )
        THEN ERR_EXIT ( SS$_BADFILENAME );

        ! test if "*.*" or "*." ( match all non-VMS file specs on tape )
        !
        IF CH$EQL ( 3,                      STAR_DOT_STAR,
                    .FILE_SPEC_DESC [ 0 ], .FILE_SPEC_DESC [ 1 ], 0 )
        THEN RETURN IS_VMS_SPEC OR WILD_QUOTED
        ELSE
            BEGIN
            IF CH$EQL ( 2,                      STAR_DOT,
                        .FILE_SPEC_DESC [ 0 ], .FILE_SPEC_DESC [ 1 ], 0 )
            THEN RETURN IS_VMS_SPEC OR WILD_QUOTED
            ELSE RETURN IS_VMS_SPEC
            END;
        END

    ELSE
        BEGIN

        ! Parse the quoted name string
        !
        PARSE_QUOTED_NAME ( .INPUT_LENGTH,
                            .INPUT_ADDR,
                            .FILE_SPEC_DESC );
```

```
;  981     1357  4            RETURN FALSE;
;  982     1358  3            END;
;  983     1359  3        END
;  984     1360  3
;  985     1361  2    ELSE RETURN FALSE;
;  986     1362  2
;  987     1363  1    END;                                  ! end of routine PARSE_PATTERN_SPEC


                                              00773              .BLKB    1
                         00  2A  2E  2A       00774 P.AAE:       .ASCII   \*.*\<0>                                   ;

                                                    STAR_DOT_STAR=        P.AAE
                                                    STAR_DOT=             P.AAE
                                                                 .EXTRN   IO_PACKET

                                         07FC 00000              .ENTRY   PARSE_PATTERN_SPEC, Save R2,R3,R4,R5,R6,R7,-; 1186
                                                                          R8,R9,R10                                   ;
                          5E           10  C2 00002              SUBL2    #16, SP                                     ;
                                       6E  94 00005              CLRB     ASCII_NAME_SPEC                             ; 1241
                          50    0000G  CF  D0 00007              MOVL     IO_PACKET, R0                               ; 1249
                          58       2C  B0  D0 0000C              MOVL     @44(R0), ABD                                ;
                          57       32  A0  3C 00010              MOVZWL   50(R0), I                                   ; 1258
                                       41  11 00014              BRB      3$                                          ;
                          50     6847  7E 00016 1$:              MOVAQ    (ABD)[I], R0                                ;
                          5A       60  3C 0001A                  MOVZWL   (R0), ATTR_PTR                              ;
                          5A       50  C0 0001D                  ADDL2    R0, ATTR_PTR                                ;
                          10       6A  91 00020                  CMPB     (ATTR_PTR), #16                             ; 1259
                                   32  12 00023                  BNEQ     3$                                          ;
                              02 A847  7F 00025                  PUSHAQ   2(ABD)[I]                                   ; 1265
                          59       9E  3C 00029                  MOVZWL   @(SP)+, ATTR_LENGTH                         ;
               0000004F   8F       59  D1 0002C                  CMPL     ATTR_LENGTH, #79                            ; 1269
                                   22  14 00033                  BGTR     3$                                          ;
                                   5A  D6 00035                  INCL     ATTR_PTR                                    ; 1275
                          56   04  AC  D0 00037                  MOVL     FILE_SPEC_DESC, R6                          ; 1280
  F844  CF   F847  CF     6A       59  2F 0003B                  MOVTUC   ATTR_LENGTH, (ATTR_PTR), ESC_CHAR, -        ;
                     04   B6       66     00044                           ANSI_A_BAD, (R6), @4(R6)                    ;
                                   02  1D 00047                  BVS      2$                                          ;
                                   51  D4 00049                  CLRL     R1                                          ;
                                   51  D5 0004B 2$:              TSTL     R1                                          ; 1279
                                   08  12 0004D                  BNEQ     3$                                          ;
                          6E       01  88 0004F                  BISB2    #1, ASCII_NAME_SPEC                         ; 1286
                          66       59  D0 00052                  MOVL     ATTR_LENGTH, (R6)                           ; 1287
                                   07  11 00055                  BRB      4$                                          ; 1282
                                   57  D7 00057 3$:              DECL     I                                           ; 1253
                          05       57  D1 00059                  CMPL     I, #5                                       ;
                                   B8  1E 0005C                  BGEQU    1$                                          ;
                     0C   AE   12  A8  3C 0005E 4$:              MOVZWL   18(ABD), INPUT_LENGTH                       ; 1299
                          51   10  A8  9E 00063                  MOVAB    16(ABD), R1                                 ; 1300
                          50       61  3C 00067                  MOVZWL   (R1), R0                                    ;
                     08   AE  01 A140  9E 0006A                  MOVAB    1(R1)[R0], INPUT_ADDR                       ; 1301
                               04  AE  9F 00070                  PUSHAB   QUOTED_NAME                                 ; 1305
                                   01  DD 00073                  PUSHL    #1                                          ;
                               10  AE  9F 00075                  PUSHAB   INPUT_ADDR                                  ;
                               18  AE  9F 00078                  PUSHAB   INPUT_LENGTH                                ;
                   FD91   CF       04  FB 0007B                  CALLS    #4, STRIP_VERSION                           ;
```

```
                    08   BC          50  B0 00080       MOVW    R0, @FILE_VERSION                        ;
                         4B          6E  E8 00084       BLBS    ASCII_NAME_SPEC, 9$                      ; 1313
                         39      04  AE  E8 00087       BLBS    QUOTED_NAME, 8$                          ; 1319
                                 04  AC  DD 0008B       PUSHL   FILE_SPEC_DESC                           ; 1332
                                 0C  AE  DD 0008E       PUSHL   INPUT_ADDR                               ; 1331
                                 14  AE  DD 00091       PUSHL   INPUT_LENGTH                             ; 1330
                                     01  DD 00094       PUSHL   #1                                       ; 1329
                  FBED   CF          04  FB 00096       CALLS   #4, PARSE_NAME_TYPE                      ;
                         04          50  E8 0009B       BLBS    R0, 5$                                   ;
                               0818  8F  BF 0009E       CHMU    #2072                                    ; 1333
                         54      04  AC  D0 000A2 5$:   MOVL    FILE_SPEC_DESC, R4                       ; 1338
 64         00    FF51   CF          03  2D 000A6       CMPC5   #3, STAR_DOT_STAR, #0, (R4), @4(R4)      ; 1337
                                 04  B4     000AD                                                        ;
                                     0B  13 000AF       BEQL    6$                                       ;
 64         00    FF46   CF          02  2D 000B1       CMPC5   #2, STAR_DOT, #0, (R4), @4(R4)           ; 1342
                                 04  B4     000B8                                                        ;
                                     04  12 000BA       BNEQ    7$                                       ;
                         50          03  D0 000BC 6$:   MOVL    #3, R0                                   ; 1344
                                     04     000BF       RET                                              ;
                         50          02  D0 000C0 7$:   MOVL    #2, R0                                   ; 1345
                                     04     000C3       RET                                              ; 1350
                                 04  AC  DD 000C4 8$:   PUSHL   FILE_SPEC_DESC                           ; 1356
                                 0C  AE  DD 000C7       PUSHL   INPUT_ADDR                               ; 1355
                                 14  AE  DD 000CA       PUSHL   INPUT_LENGTH                             ; 1354
                  FC92   CF          03  FB 000CD       CALLS   #3, PARSE_QUOTED_NAME                    ;
                                     50  D4 000D2 9$:   CLRL    R0                                       ; 1363
                                     04     000D4       RET                                              ;

; Routine Size:  213 bytes,    Routine Base:  $CODE$ + 0778
```

```
GLOBAL ROUTINE RESULTANT_STRING ( VMS_NAME,
                                  FILE_SPEC_LEN,
                                  FILE_SPEC_PTR,
                                  VERSION       ) : COMMON_CALL NOVALUE =

!++
!
! FUNCTIONAL DESCRIPTION:
!       This routine prepares and returns the resultant string name
!
! CALLING SEQUENCE:
!       RESULTANT_STRING ( ARG1, ARG2, ARG3, ARG4 )
!
! INPUT PARAMETERS:
!       ARG1 - flag ( set if a VMS file spec )
!       ARG2 - length of file specifiation ( name dot type )
!       ARG3 - pointer to file specification ( name dot type )
!       ARG4 - the file version number
!
! IMPLICIT INPUTS:
!       none
!
! OUTPUT PARAMETERS:
!       none
!
! IMPLICIT OUTPUTS:
!       none
!
! ROUTINE VALUE:
!       none
!
! SIDE EFFECTS:
!       resulant string is returned to the user
!
!--
    BEGIN

    EXTERNAL REGISTER COMMON_REG;                       ! so we can call ERR_EXIT

    EXTERNAL WORK_AREA;                                 ! general use scratch area

    EXTERNAL ROUTINE
        SYS$FAO         : ADDRESSING_MODE ( ABSOLUTE ); ! format the string

    ! map the input agruments into data structures
    !
    MAP
        VMS_NAME        : BITVECTOR [ 1 ],
        FILE_SPEC_LEN   : LONG,
        FILE_SPEC_PTR   : REF VECTOR [ , BYTE ],
        VERSION         : LONG;

    LITERAL
        RESULTANT_MAX_LEN = 164;                        ! max size of resultant name

    LOCAL
        OUT_INDEX       : LONG,                         ! index into output buffer
```

```
	RESULTANT_DESC  : VECTOR [ 2, LONG ];    ! descriptor for resultant name

    BIND
	RESULTANT_NAME  = WORK_AREA : VECTOR [ RESULTANT_MAX_LEN, BYTE ],
	FAO_VERSION     = DESCRIPTOR ( ';!SW' ),
	STRIP_SIZE      = OUT_INDEX;               ! redefine register

    ! string any trailing spaces from the file specification
    !
    STRIP_SIZE = 0;
    DECR I FROM ( .FILE_SPEC_LEN - 1 ) TO 0 DO
	IF .FILE_SPEC_PTR [ .I ] NEQ ' '
	THEN
	    BEGIN
	    STRIP_SIZE = .I + 1;
	    EXITLOOP;
	    END;
    FILE_SPEC_LEN = .STRIP_SIZE;

    ! move the ( name and type ) or ANSI file id into the output buffer
    !
    IF .VMS_NAME [ 0 ]
    THEN
	BEGIN

	! just move the fields if a VMS spec
	!
	CH$MOVE ( .FILE_SPEC_LEN, .FILE_SPEC_PTR, RESULTANT_NAME );
	OUT_INDEX = .FILE_SPEC_LEN;
	END

    ELSE
	BEGIN

	! ANSI spec is placed in quotes and all quotes goto double quotes
	!
	LOCAL
	    IN_INDEX     : LONG, ! index into the input string
	    MOVE_SIZE    : LONG, ! the number of characters to be moved
	    QUOTE_CHAR   : LONG; ! location of a quota character

	! stuff initial quote into output buffer
	!
	RESULTANT_NAME [ 0 ] = '"';

	! initialize the indices
	!
	IN_INDEX  = 0;
	OUT_INDEX = 1;

	! loop for each string ending with a quote
	!
	WHILE NOT CH$FAIL (
		QUOTE_CHAR = CH$FIND_CH ( .FILE_SPEC_LEN,
					  FILE_SPEC_PTR [ .IN_INDEX ],
					  '"' )
```

```
                                                                        ) DO
                BEGIN

                ! calculate the number of characters to be moved
                ! and move the characters including the quote character
                !
                MOVE_SIZE = ( .QUOTE_CHAR - FILE_SPEC_PTR [ .IN_INDEX ] ) + 1;
                CH$MOVE ( .MOVE_SIZE,
                          FILE_SPEC_PTR  [ .IN_INDEX ],
                          RESULTANT_NAME [ .OUT_INDEX ] );

                ! adjust the indices, and the number of characters remaining

                IN_INDEX      = .IN_INDEX      + .MOVE_SIZE;
                OUT_INDEX     = .OUT_INDEX     + .MOVE_SIZE;
                FILE_SPEC_LEN = .FILE_SPEC_LEN - .MOVE_SIZE;

                ! stuff extra quote characters, bump index
                !
                RESULTANT_NAME [ .OUT_INDEX ]      = '"';
                OUT_INDEX = .OUT_INDEX + 1;
                END;

            ! move the remaining character into the output buffer
            !
            CH$MOVE ( .FILE_SPEC_LEN,
                      FILE_SPEC_PTR  [ .IN_INDEX ],
                      RESULTANT_NAME [ .OUT_INDEX ] );
            OUT_INDEX = .OUT_INDEX + .FILE_SPEC_LEN;

            ! stuff final quote characters and null type, bump index
            !
            RESULTANT_NAME [ .OUT_INDEX ]     = '"';
            RESULTANT_NAME [ .OUT_INDEX + 1 ] = '.';
            OUT_INDEX = .OUT_INDEX + 2;
            END;

        ! adjust the descriptor to point to the remaining of the output buffer
        ! and FAO the version number into the output buffer
        !
        RESULTANT_DESC [ 0 ] = RESULTANT_MAX_LEN - .OUT_INDEX;
        RESULTANT_DESC [ 1 ] = RESULTANT_NAME [ .OUT_INDEX ];
        SYS$FAO ( FAO_VERSION, RESULTANT_DESC, RESULTANT_DESC, .VERSION );

        ! point the descriptor to the entire file spec in the output buffer
        !
        RESULTANT_DESC [ 0 ] = .RESULTANT_DESC [ 0 ] + .OUT_INDEX;
        RESULTANT_DESC [ 1 ] = RESULTANT_NAME;

        ! return the output buffer to the user
        !
        IF NOT KERNEL_CALL ( RETURN_RESULTANT_SPEC, RESULTANT_DESC )
        THEN ERR_EXIT ( );

        END;                                    ! end of routine RESULTANT_STRING
```

```
                57  53  21  3B  0084D P.AAG:   .ASCII   \;!SW\                                          ;
                                00851          .BLKB    3
                       00000004 00854 P.AAF:   .LONG    4                                               ;
                      00000000' 00858          .ADDRESS P.AAG                                           ;

                                      FAO_VERSION=        P.AAF
                                               .EXTRN   WORK_AREA, SYS$FAO
                                               .EXTRN   SYS$CMKRNL

                           07FC 00000          .ENTRY   RESULTANT_STRING, Save R2,R3,R4,R5,R6,R7,-   ; 1364
                                                        R8,R9,R10
              5A     0000G  CF  9E 00002       MOVAB    RESULTANT_NAME, R10
              5E            08  C2 00007       SUBL2    #8, SP
                            56  D4 0000A       CLRL     STRIP_SIZE                                   ; 1431
              50       08   AC  D0 0000C       MOVL     FILE_SPEC_LEN, I                             ; 1433
                            0D  11 00010       BRB      2$                                           ;
              20    0C BC40     91 00012 1$:   CMPB     @FILE_SPEC_PTR[I], #32                       ;
                            06  13 00017       BEQL     2$                                           ;
              56       01   A0  9E 00019       MOVAB    1(R0), STRIP_SIZE                            ; 1436
                            03  11 0001D       BRB      3$                                           ; 1435
              F0            50  F4 0001F 2$:   SOBGEQ   I, 1$                                        ; 1433
         08   AC            56  D0 00022 3$:   MOVL     STRIP_SIZE, FILE_SPEC_LEN                    ; 1439
              0C       04   AC  E9 00026       BLBC     VMS_NAME, 4$                                 ; 1443
   6A    0C   BC       08   AC  28 0002A       MOVC3    FILE_SPEC_LEN, @FILE_SPEC_PTR, -             ; 1449
                                                        RESULTANT_NAME
              56       08   AC  D0 00030       MOVL     FILE_SPEC_LEN, OUT_INDEX                     ; 1450
                            50  11 00034       BRB      8$                                           ; 1443
              6A            22  90 00036 4$:   MOVB     #34, RESULTANT_NAME                          ; 1465
              56            01  7D 00039       MOVQ     #1, OUT_INDEX                                ; 1470
   52         57       0C   AC  C1 0003C 5$:   ADDL3    FILE_SPEC_PTR, IN_INDEX, R2                  ; 1476
   62    08   AC            22  3A 00041       LOCC     #34, FILE_SPEC_LEN, (R2)                     ;
                            02  12 00046       BNEQ     6$                                           ;
                            51  D4 00048       CLRL     R1                                           ;
              59            51  D0 0004A 6$:   MOVL     R1, QUOTE_CHAR                               ;
                            1F  13 0004D       BEQL     7$                                           ; 1478
   51         59            52  C3 0004F       SUBL3    R2, QUOTE_CHAR, R1                           ; 1484
              58       01   A1  9E 00053       MOVAB    1(R1), MOVE_SIZE                             ;
 6A46         62            58  28 00057       MOVC3    MOVE_SIZE, (R2), RESULTANT_NAME[OUT_INDEX]   ; 1487
              57            58  C0 0005C       ADDL2    MOVE_SIZE, IN_INDEX                          ; 1491
              56            58  C0 0005F       ADDL2    MOVE_SIZE, OUT_INDEX                         ; 1492
         08   AC            58  C2 00062       SUBL2    MOVE_SIZE, FILE_SPEC_LEN                     ; 1493
            6A46            22  90 00066       MOVB     #34, RESULTANT_NAME[OUT_INDEX]               ; 1497
                            56  D6 0006A       INCL     OUT_INDEX                                    ; 1498
                            CE  11 0006C       BRB      5$                                           ; 1474
 6A46    0C BC47       08   AC  28 0006E 7$:   MOVC3    FILE_SPEC_LEN, @FILE_SPEC_PTR[IN_INDEX], -   ; 1505
                                                        RESULTANT_NAME[OUT_INDEX]
              56       08   AC  C0 00076       ADDL2    FILE_SPEC_LEN, OUT_INDEX                     ; 1506
            6A46            22  90 0007A       MOVB     #34, RESULTANT_NAME[OUT_INDEX]               ; 1510
         01 AA46            2E  90 0007E       MOVB     #46, RESULTANT_NAME+1[OUT_INDEX]             ; 1511
              56            02  C0 00083       ADDL2    #2, OUT_INDEX                                ; 1512
   6E 000000A4  8F          56  C3 00086 8$:   SUBL3    OUT_INDEX, #164, RESULTANT_DESC              ; 1518
04      AE      56          5A  C1 0008E       ADDL3    R10, OUT_INDEX, RESULTANT_DESC+4             ; 1519
                       10   AC  DD 00093       PUSHL    VERSION                                      ; 1520
                       04   AE  9F 00096       PUSHAB   RESULTANT_DESC                               ;
                       08   AE  9F 00099       PUSHAB   RESULTANT_DESC                               ;
                     FF58   CF  9F 0009C       PUSHAB   FAO_VERSION                                  ;
```

```
        00000000G  9F          04 FB 000A0         CALLS   #4, @#SYS$FAO                                  ;
                   6E          56 C0 000A7         ADDL2   OUT_INDEX, RESULTANT_DESC                       ; 1524
               04  AE          6A 9E 000AA         MOVAB   RESULTANT_NAME, RESULTANT_DESC+4                ; 1525
                               5E DD 000AE         PUSHL   SP                                              ; 1529
                               01 DD 000B0         PUSHL   #1                                              ;
                               5E DD 000B2         PUSHL   SP                                              ;
                        0000V  CF 9F 000B4         PUSHAB  RETURN_RESULTANT_SPEC                           ;
        00000000G  9F          04 FB 000B8         CALLS   #4, @#SYS$CMKRNL                                ;
                   02          50 E8 000BF         BLBS    R0, 9$                                          ;
                               00 BF 000C2         CHMU    #0                                              ; 1530
                                  04 000C4 9$:     RET                                                     ; 1532

; Routine Size:  197 bytes,    Routine Base:  $CODE$ + 085C
```

```
ROUTINE RETURN_RESULTANT_SPEC ( FILE_NAME_DESC ) =

!++
!
! FUNCTIONAL DESCRIPTION:
!       This routine enables the write-back of the result file name string.
!
! CALLING SEQUENCE:
!       RETURN_RESULTANT_SPEC ( ARG1 )
!       must be called in kernel access mode
!
! INPUT PARAMETERS:
!       ARG1 - address of file name string descriptor
!
! IMPLICIT INPUTS:
!       IO_PACKET - address of current request packet
!
! OUTPUT PARAMETERS:
!       none
!
! IMPLICIT OUTPUTS:
!       none
!
! ROUTINE VALUE:
!       0 - failure
!       1 - success
!
! SIDE EFFECTS:
!       enables write back
!
! USER ERRORS:
!       SS$_RESULTOVF - file name string overflows buffer
!--

    BEGIN

    LOCAL
        SIZE,                                       ! size of write back data
        ABD     : REF BBLOCKVECTOR [ , ABD$C_LENGTH ];! addr of buffer descr

    MAP FILE_NAME_DESC  : REF VECTOR [ 2, LONG ];! descr of file name string

    EXTERNAL
        IO_PACKET       : REF BBLOCK;           ! address of current IRP

    ! get buffer descriptor address address of buffer descriptors
    !
    ABD = .BBLOCK [ .IO_PACKET [ IRP$L_SVAPTE ], AIB$L_DESCRIPT ];

    ! return length of result string
    !
    SIZE = .FILE_NAME_DESC [ 0 ];

    IF .ABD [ ABD$C_RESL, ABD$W_COUNT ] GEQU 2
    THEN
        ( .ABD [ ABD$C_RESL, ABD$W_TEXT ] +
           ABD [ ABD$C_RESL, ABD$W_TEXT ] + 1 ) < 0, 16 > = .SIZE < 0, 16 >;
```

```
2
2
2       IF .ABD [ ABD$C_RES, ABD$W_COUNT ] NEQ 0
2       THEN
3           BEGIN
3
3           ! if result string overflows buffer, return error
3           !
3           IF .SIZE GTRU .ABD [ ABD$C_RES, ABD$W_COUNT ]
3           THEN
4               BEGIN
4               ERROR ( SS$_RESULTOVF );
4               RETURN 0;
4               END
3
3           ELSE
3               CH$COPY ( .SIZE, .FILE_NAME_DESC [ 1 ], 0,
3                           .ABD [ ABD$C_RES, ABD$W_COUNT ],
4                         ( .ABD [ ABD$C_RES, ABD$W_TEXT ] +
3                            ABD [ ABD$C_RES, ABD$W_TEXT ] + 1 ));
2           END;
2
2       RETURN 1;
2
1       END;                                        ! end of routine RETURN_RESULTANT_SPEC


                                                         .EXTRN  USER_STATUS

                                    003C 00000 RETURN_RESULTANT_SPEC:
                                                         .WORD   Save R2,R3,R4,R5                     ; 1533
                  50     0000G  CF  D0 00002             MOVL    IO_PACKET, R0                        ; 1580
                  51        2C  B0  D0 00007             MOVL    @44(R0), ABD
                  53        04  AC  D0 0000B             MOVL    FILE_NAME_DESC, R3                   ; 1584
                  54            63  D0 0000F             MOVL    (R3), SIZE
                  02        1A  A1  B1 00012             CMPW    26(ABD), #2                          ; 1586
                                0E  1F 00016             BLSSU   1$
                  52        18  A1  9E 00018             MOVAB   24(ABD), R2                          ; 1588
                  50            62  3C 0001C             MOVZWL  (R2), R0
                         01 A240    9F 0001F             PUSHAB  1(R2)[R0]                            ; 1589
                  9E            54  B0 00023             MOVW    SIZE, @(SP)+
                  52        22  A1  3C 00026 1$:         MOVZWL  34(ABD), R2                          ; 1592
                                1D  13 0002A             BEQL    3$
                  52            54  D1 0002C             CMPL    SIZE, R2                             ; 1598
                                09  1B 0002F             BLEQU   2$
           0000G  CF      0214  8F  B0 00031             MOVW    #532, USER_STATUS                    ; 1601
                                13  11 00038             BRB     4$                                   ; 1602
                  51            20  C0 0003A 2$:         ADDL2   #32, R1                              ; 1608
                  50            61  3C 0003D             MOVZWL  (R1), R0
    52        00  04  B3        54  2C 00040             MOVC5   SIZE, @4(R3), #0, R2, 1(R1)[R0]
                         01 A140       00046
                  50            01  D0 00049 3$:         MOVL    #1, R0                               ; 1612
                                04     0004C             RET
                            50  D4     0004D 4$:         CLRL    R0                                   ; 1614
                                04     0004F             RET
```

```
; Routine Size:  80 bytes,    Routine Base:  $CODE$ + 0921

; 1241          1615  1
; 1242          1616  1
; 1243          1617  1 END
; 1244          1618  1
; 1245          1619  0 ELUDOM


;                              PSECT SUMMARY
;
;     Name                     Bytes                        Attributes
;
;   $CODE$                        2417  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)


;                        Library Statistics
;
;                                          -------- Symbols --------      Pages        Processing
;     File                                 Total    Loaded   Percent      Mapped       Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1           18619       33         0      1000          00:01.9


;                              COMMAND QUALIFIERS

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

; Size:          1373 code + 1044 data bytes
; Run Time:         00:31.3
; Elapsed Time:     01:07.2
; Lines/CPU Min:      3099
; Lexemes/CPU-Min: 18319
; Memory Used:  154 pages
; Compilation Complete
```

LOCKDB
LIS

LOGIO
LIS

MATCHNAME
LIS

IODONE
LIS

LOCKDN
LIS

NXTVOL
LIS

MAIL
LIS

MOUVOL
LIS

OPRCOM
LIS

INIMTA
LIS

MODIFY
LIS

PARSE
LIS

0256 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

WTATTR
LIS

POSFID
LIS

RESACC
LIS

RWVOL
LIS

MTHRTL

STRVIO
LIS

MTHRTL
MAP

POSBE
LIS

RDATTR
LIS

REWSPC
LIS

UVMTHRTL
MAP