
MMM MMM TTTTTTTTTTTTTTT AAAAAAAAA AAAAAAAAA CCCCCCCCCCCC PPPPPPPPPPPP
MMM MMM TTTTTTTTTTTTTTT AAAAAAAAA AAAAAAAAA CCCCCCCCCCCC PPPPPPPPPPPP
MMM MMM TTTTTTTTTTTTTTT AAAAAAAAA AAAAAAAAA CCCCCCCCCCCC PPPPPPPPPPPP
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

```
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
```

```
;     1          0001  0 MODULE PARSE ( LANGUAGE ( BLISS32 ),
;     2          0002  0                 IDENT = 'V04-000'
;     3          0003  0                 ) =
;     4          0004  1 BEGIN
;     5          0005  1 !
;     6          0006  1 !**********************************************************************
;     7          0007  1 !*                                                                    *
;     8          0008  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                          *
;     9          0009  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.           *
;    10          0010  1 !*  ALL RIGHTS RESERVED.                                            *
;    11          0011  1 !*                                                                    *
;    12          0012  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  *
;    13          0013  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;    14          0014  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER  *
;    15          0015  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
;    16          0016  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY  *
;    17          0017  1 !*  TRANSFERRED.                                                       *
;    18          0018  1 !*                                                                    *
;    19          0019  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE  *
;    20          0020  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT  *
;    21          0021  1 !*  CORPORATION.                                                       *
;    22          0022  1 !*                                                                    *
;    23          0023  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS  *
;    24          0024  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.            *
;    25          0025  1 !*                                                                    *
;    26          0026  1 !*                                                                    *
;    27          0027  1 !**********************************************************************
;    28          0028  1
;    29          0029  1 !++
;    30          0030  1 !
;    31          0031  1 ! FACILITY:  MTAACP
;    32          0032  1 !
;    33          0033  1 ! ABSTRACT:
;    34          0034  1 !       This module parses the file identification field in the HDR1 and
;    35          0035  1 !       in the case of VMS long file names continued in the HDR4 or file
;    36          0036  1 !       name string, then generates and returns resultant file spec string.
;    37          0037  1 !
;    38          0038  1 ! ENVIRONMENT:
;    39          0039  1 !
;    40          0040  1 !       STARLET operating system, including privileged system services
;    41          0041  1 !       and internal exec routines.
;    42          0042  1 !
;    43          0043  1 !--
;    44          0044  1
;    45          0045  1 !
;    46          0046  1 !
;    47          0047  1 ! AUTHOR:  David Michael Walp,   CREATION DATE:  17-NOV-1981
;    48          0048  1 !
;    49          0049  1 ! MODIFIED BY:
;    50          0050  1 !
;    51          0051  1 !       V03-003 MMD0282         Meg Dumont,     23-Mar-1984  10:31
;    52          0052  1 !               Fix long file name support such that for ANSI version
;    53          0053  1 !               3 volumes it converts the exentsion length to
;    54          0054  1 !               ASCII before writing it to the label.
;    55          0055  1 !
;    56          0056  1 !       V03-002 MMD0184         Meg Dumont,     6-Jul-1983  18:30
;    57          0057  1 !               Add support for $ and - as valid VMS characters to tape
```

```
;    58          0058  1 !
;    59          0059  1 !       V03-001 MMD0158         Meg Dumont,     26-Apr-1983  9:21
;    60          0060  1 !               Add support for underscore as a valid character to tape. Add
;    61          0061  1 !               support for VMS long file names includes: 1) Changing the
;    62          0062  1 !               default file name sizes to support the long file name, uses
;    63          0063  1 !               the symbol FILE_SPEC_MAX. 2) Change the routines PARSE_HDR1_FID
;    64          0064  1 !               to PARSE_FID and enhance it to understand that the FID maybe
;    65          0065  1 !               split between the HDR1 and HDR4 labels. 3) Change file name
;    66          0066  1 !               length checked in PARSE_PATTERN_SPEC. 4) In RESULTANT_STRING
;    67          0067  1 !               change length of RESULTANT_MAX_LENGTH.
;    68          0068  1 !
;    69          0069  1 !       V03-002 DMW00072        David Michael Walp      28-Jan-1981
;    70          0070  1 !               Fix wildcard characters in quoted string problem
;    71          0071  1 !
;    72          0072  1 !       V03-001 DMW00067        David Michael Walp      11-Jan-1981
;    73          0073  1 !               Accept ANSI name in quotes thru file name parameters
;    74          0074  1 !
;    75          0075  1 !**
;    76          0076  1
;    77          0077  1 LIBRARY 'SYS$LIBRARY:LIB.L32';
;    78          0078  1
;    79          0079  1 REQUIRE 'SRC$:MTADEF.B32';
;    80          0463  1
;    81          0464  1 FORWARD ROUTINE
;    82          0465  1     CALC_VERSION            : NOVALUE,      ! calculate VMS version from ANSI
;    83          0466  1     PARSE_FID               : COMMON_CALL,  ! parse the file id fields in HDR1 and HDR4
;    84          0467  1     PARSE_NAME_TYPE,                        ! parse a filespec into name + type
;    85          0468  1     PARSE_PATTERN_SPEC      : COMMON_CALL,  ! parse a input wildcard file spec
;    86          0469  1     PARSE_QUOTED_NAME       : COMMON_CALL NOVALUE, ! parse a quoted 17 ANSI "a"
;    87          0470  1                                             !       character file name
;    88          0471  1     RESULTANT_STRING        : COMMON_CALL NOVALUE, ! prepare and return the
;    89          0472  1                                             !       resultant string name
;    90          0473  1     RETURN_RESULTANT_SPEC,                  ! write back file name information
;    91          0474  1     STRIP_VERSION           : COMMON_CALL;  ! strips the version from file spec
```

```
;    93          0475  1 !++
;    94          0476  1 !
;    95          0477  1 ! UPLIT was used instead of CH$TRANSTABLE here, the code produced is the
;    96          0478  1 !  same (ie the constant string generated).  UPLIT was used because
;    97          0479  1 !  CH$TRANSTABLE generates a warning error because more then a single
;    98          0480  1 !  character at a time is specified in the %ASCII. ( BLISS KLUDGE )
;    99          0481  1 !
;   100          0482  1 !--
;   101          0483  1
;   102          0484  1 GLOBAL BIND
;   103          0485  1
;   104          0486  1     ! This table will upcase a..z and return '@' for any non ANSI
;   105          0487  1     !  'a' characters.  Bad characters are thus flaged.
;   106          0488  1     !
;   107          0489  1     ANSI_A_BAD   = UPLIT BYTE ( %ASCII '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@',
;   108          0490  1                                 %ASCII ' !"@@%&''()*+,-./0123456789:;<=>?',
;   109          0491  1                                 %ASCII '@ABCDEFGHIJKLMNOPQRSTUVWXYZ@@@@_',
;   110          0492  1                                 %ASCII '@ABCDEFGHIJKLMNOPQRSTUVWXYZ@@@@@',
;   111          0493  1                                 %ASCII '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@',
;   112          0494  1                                 %ASCII '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@',
;   113          0495  1                                 %ASCII '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@',
;   114          0496  1                                 %ASCII '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@' ),
;   115          0497  1
;   116          0498  1     ! This table will upcase a..z and return space for any non ANSI
;   117          0499  1     !  'a' characters.  Thus bad characters are converted to good characters.
;   118          0500  1     !
;   119          0501  1     ANSI_A_GOOD  = UPLIT BYTE ( %ASCII '                                ',
;   120          0502  1                                 %ASCII ' !"  %&''()*+,-./0123456789:;<=>?',
;   121          0503  1                                 %ASCII ' ABCDEFGHIJKLMNOPQRSTUVWXYZ    ',
;   122          0504  1                                 %ASCII ' ABCDEFGHIJKLMNOPQRSTUVWXYZ    ',
;   123          0505  1                                 %ASCII '                                ',
;   124          0506  1                                 %ASCII '                                ',
;   125          0507  1                                 %ASCII '                                ',
;   126          0508  1                                 %ASCII '                                ' ),
;   127          0509  1
;   128          0510  1     ! point to a '@'
;   129          0511  1     !
;   130          0512  1     ESC_CHAR = ANSI_A_BAD;
;   131          0513  1
;   132          0514  1 BIND
;   133          0515  1
;   134          0516  1     ! This table will upcase a..z and return '@' for any non-VMS character
;   135          0517  1     !  including wild cards
;   136          0518  1     !
;   137          0519  1     VMS_NO_WILD  = UPLIT BYTE ( %ASCII '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@',
;   138          0520  1                                 %ASCII '@@@@$@@@@@@@@-@@0123456789@@@@@@',
;   139          0521  1                                 %ASCII '@ABCDEFGHIJKLMNOPQRSTUVWXYZ@@@@_',
;   140          0522  1                                 %ASCII '@ABCDEFGHIJKLMNOPQRSTUVWXYZ@@@@@',
;   141          0523  1                                 %ASCII '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@',
;   142          0524  1                                 %ASCII '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@',
;   143          0525  1                                 %ASCII '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@',
;   144          0526  1                                 %ASCII '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@' ),
;   145          0527  1
;   146          0528  1
;   147          0529  1     ! This table will upcase a..z and return '@' for any non-VMS character
;   148          0530  1     !  wild cards ( %, * ) are valid characters in this table
;   149          0531  1     !
```

```
;  150          0532  1      VMS_WILD     = UPLIT BYTE ( %ASCII '                                ',
;  151          0533  1                                  %ASCII '    $%    *  -  0123456789      ',
;  152          0534  1                                  %ASCII ' ABCDEFGHIJKLMNOPQRSTUVWXYZ    _ ',
;  153          0535  1                                  %ASCII ' ABCDEFGHIJKLMNOPQRSTUVWXYZ    % ',
;  154          0536  1                                  %ASCII '                                ',
;  155          0537  1                                  %ASCII '                                ',
;  156          0538  1                                  %ASCII '                                ',
;  157          0539  1                                  %ASCII '                                ' );
;  158          0540  1
```

```
;  160     0541  1 GLOBAL ROUTINE PARSE_NAME_TYPE ( WILD_CARDS,
;  161     0542  1                                   SPEC_LEN,
;  162     0543  1                                   SPEC_PTR,
;  163     0544  1                                   OUT_DESC ) =
;  164     0545  1
;  165     0546  1 !++
;  166     0547  1 !
;  167     0548  1 ! FUNCTIONAL DESCRIPTION:
;  168     0549  1 !       This routine parses a VMS file specification string into a name and type
;  169     0550  1 !       fields
;  170     0551  1 !
;  171     0552  1 ! CALLING SEQUENCE:
;  172     0553  1 !       error = PARSE_NAME_TYPE ( ARG1, ARG2, ARG3, ARG4 )
;  173     0554  1 !
;  174     0555  1 ! INPUT PARAMETERS:
;  175     0556  1 !       ARG1 - flags for "if wild cards are valid"
;  176     0557  1 !       ARG2 - the length of the file specification
;  177     0558  1 !       ARG3 - the address of the file specification
;  178     0559  1 !
;  179     0560  1 ! IMPLICIT INPUTS:
;  180     0561  1 !       none
;  181     0562  1 !
;  182     0563  1 ! OUTPUT PARAMETERS:
;  183     0564  1 !       ARG4 - pointer to the output buffer - size is updated
;  184     0565  1 !                  ASSUMED THAT OUTPUT BUFFER >= FILE_SPEC_MAX BYTES
;  185     0566  1 !
;  186     0567  1 ! IMPLICIT OUTPUTS:
;  187     0568  1 !       none
;  188     0569  1 !
;  189     0570  1 ! ROUTINE VALUE:
;  190     0571  1 !       error code - SS$_NORMAL
;  191     0572  1 !                  - SS$_BADFILENAME
;  192     0573  1 !
;  193     0574  1 ! SIDE EFFECTS:
;  194     0575  1 !       none
;  195     0576  1 !
;  196     0577  1 !--
;  197     0578  1
;  198     0579  2     BEGIN
;  199     0580  2
;  200     0581  2     MAP
;  201     0582  2         WILD_CARDS  : BITVECTOR [ 1 ],
;  202     0583  2         SPEC_LEN    : LONG,
;  203     0584  2         SPEC_PTR    : REF VECTOR [ , BYTE ],
;  204     0585  2         OUT_DESC    : REF VECTOR [ 2, LONG ];
;  205     0586  2
;  206     0587  2     EXTERNAL
;  207     0588  2         LOCAL_FIB   : BBLOCK;
;  208     0589  2
;  209     0590  2     LITERAL
;  210     0591  2         MAX_NAME_FIELD_LEN = 39,                ! Support for VMS long file
;  211     0592  2         MAX_TYPE_FIELD_LEN = 39;                ! names
;  212     0593  2
;  213     0594  2     LOCAL
;  214     0595  2         IN_INDEX    : LONG,                     ! index into input file spec
;  215     0596  2         OUT_INDEX   : LONG,                     ! index into output file spec
;  216     0597  2         REM_LEN     : LONG,                     ! size minus type field
```

```
;  217     0598  2          TABLE       : REF VECTOR [ 128, BYTE ], ! ptr to VMS file spec chars
;  218     0599  2          TYPE_LEN    : LONG;                     ! size of type field
;  219     0600  2      BIND
;  220     0601  2          OUT_BUF = OUT_DESC [ 1 ] : REF VECTOR [ , BYTE ];
;  221     0602  2
;  222     0603  2      ! set up which table is to be use for translation
;  223     0604  2
;  224     0605  2      TABLE = ( IF .WILD_CARDS [ 0 ] THEN VMS_WILD ELSE VMS_NO_WILD );
;  225     0606  2
;  226     0607  2      !++
;  227     0608  2      ! PARSE the name field
;  228     0609  2      !
;  229     0610  2      !--
;  230     0611  2      ! find the dot in the spec ( IN_INDEX points to it )
;  231     0612  2      ! point to the end of the spec if not found
;  232     0613  2      ! calculate the size of the name field
;  233     0614  2      !
;  234     0615  2      IN_INDEX = CH$FIND_CH ( .SPEC_LEN, SPEC_PTR [ 0 ], '.' );
;  235     0616  2      IN_INDEX = ( IF CH$FAIL ( .IN_INDEX )
;  236     0617  3                   THEN .SPEC_LEN
;  237     0618  3                   ELSE .IN_INDEX - SPEC_PTR [ 0 ] );
;  238     0619  2
;  239     0620  2      ! test if the name field is too long
;  240     0621  2      !
;  241     0622  2      IF ( .IN_INDEX GTR MAX_NAME_FIELD_LEN ) THEN RETURN SS$_BADFILENAME;
;  242     0623  2
;  243     0624  2      ! move string into the output buffer and
;  244     0625  2      ! test for invalid characters in the name field
;  245     0626  2      !
;  246     0627  2      IF 0 NEQ MOVTUC ( IN_INDEX, SPEC_PTR [ 0 ], ESC_CHAR, .TABLE,
;  247     0628  2                        IN_INDEX, OUT_BUF  [ 0 ] )
;  248     0629  2      THEN RETURN SS$_BADFILENAME;
;  249     0630  2
;  250     0631  2      ! if all wild name flag is set stuff "*" into output buffer
;  251     0632  2      ! adjust output index
;  252     0633  2      !
;  253     0634  2      IF .WILD_CARDS [ 0 ] AND .LOCAL_FIB [ FIB$V_ALLNAM ]
;  254     0635  2      THEN
;  255     0636  2          BEGIN
;  256     0637  3          OUT_BUF [ 0 ] = '*';
;  257     0638  3          OUT_INDEX     = 1;
;  258     0639  3          END
;  259     0640  3      ELSE OUT_INDEX = .IN_INDEX;
;  260     0641  2
;  261     0642  2      ! stuff the dot into the output buffer
;  262     0643  2      !
;  263     0644  2      OUT_BUF [ .OUT_INDEX ] = '.';
;  264     0645  2
;  265     0646  2      !++
;  266     0647  2      !
;  267     0648  2      ! PARSE the type field
;  268     0649  2      !
;  269     0650  2      !--
;  270     0651  2      ! OUT_INDEX now points to the next avaliable output character
;  271     0652  2      ! IN_INDEX  now points to the start of type field
;  272     0653  2      !
;  273     0654  2
```

```
;  274          0655  2      OUT_INDEX = .OUT_INDEX + 1;
;  275          0656  2      IN_INDEX  = .IN_INDEX  + 1;
;  276          0657  2
;  277          0658  2      ! find the end of the type field
;  278          0659  2      !
;  279          0660  2      REM_LEN  = .SPEC_LEN - .IN_INDEX;
;  280          0661  2      TYPE_LEN = CH$FIND_CH ( .REM_LEN, SPEC_PTR [ .IN_INDEX ], ' ' );
;  281          0662  3      TYPE_LEN = ( IF CH$FAIL ( .TYPE_LEN )
;  282          0663  3                   THEN .REM_LEN
;  283          0664  2                   ELSE .TYPE_LEN - SPEC_PTR [ .IN_INDEX ] );
;  284          0665  2
;  285          0666  2      ! test if the type field is too long
;  286          0667  2      !
;  287          0668  2      IF ( .TYPE_LEN GTR MAX_TYPE_FIELD_LEN ) THEN RETURN SS$_BADFILENAME;
;  288          0669  2
;  289          0670  2      ! test if all trailing characters are spaces
;  290          0671  2      !
;  291          0672  2      REM_LEN  = .REM_LEN - .TYPE_LEN;
;  292          0673  2      IF NOT CH$FAIL (
;  293          0674  2           CH$FIND_NOT_CH ( .REM_LEN, SPEC_PTR [ .IN_INDEX + .TYPE_LEN ], ' ' ))
;  294          0675  2      THEN RETURN SS$_BADFILENAME;
;  295          0676  2
;  296          0677  2      ! move string into the output buffer
;  297          0678  2      ! test for invalid characters in the type field
;  298          0679  2      !
;  299          0680  2      REM_LEN = .OUT_DESC [ 0 ] - .OUT_INDEX;
;  300          0681  2      IF 0 NEQ MOVTUC ( TYPE_LEN, SPEC_PTR [ .IN_INDEX  ], ESC_CHAR, .TABLE,
;  301          0682  2                        REM_LEN,  OUT_BUF  [ .OUT_INDEX ] )
;  302          0683  2      THEN RETURN SS$_BADFILENAME;
;  303          0684  2
;  304          0685  2      ! if all wild type flag is set stuff "*" into output buffer
;  305          0686  2      ! adjust output length
;  306          0687  2      !
;  307          0688  2      IF .WILD_CARDS [ 0 ] AND .LOCAL_FIB [ FIB$V_ALLTYP ]
;  308          0689  2      THEN
;  309          0690  3          BEGIN
;  310          0691  3          OUT_BUF [ .OUT_INDEX ] = '*';
;  311          0692  3          OUT_DESC [ 0 ] = .OUT_INDEX + 1;
;  312          0693  3          END
;  313          0694  2      ELSE OUT_DESC [ 0 ] = .OUT_INDEX + .TYPE_LEN;
;  314          0695  2
;  315          0696  2      RETURN SS$_NORMAL;
;  316          0697  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
2F 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  \aaaaaaaaaaaaaaaaaaaaaaaaaaaaaa\
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  \aaaaaaaaaaaaaaaaaaaaaaaaaaaaaa\
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  \aaaaaaaaaaaaaaaaaaaaaaaaaaaaaa\
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  \aaaa$%aaaa*aa-aa0123456789aaaaaa\
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  \@ABCDEFGHIJKLMNOPQRSTUVWXYZaaaa_\
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  \@ABCDEFGHIJKLMNOPQRSTUVWXYZaaaaa\
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  \aaaaaaaaaaaaaaaaaaaaaaaaaaaaaa\
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  \aaaaaaaaaaaaaaaaaaaaaaaaaaaaaa\
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  \aaaaaaaaaaaaaaaaaaaaaaaaaaaaaa\
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  \aaaaaaaaaaaaaaaaaaaaaaaaaaaaaa\
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    #45, @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

```
;  318     0698  1 GLOBAL ROUTINE PARSE_QUOTED_NAME ( SPEC_LEN,
;  319     0699  1                                     SPEC_PTR,
;  320     0700  1                                     OUT_DESC ) : COMMON_CALL NOVALUE =
;  321     0701  1
;  322     0702  1 !++
;  323     0703  1 !
;  324     0704  1 ! FUNCTIONAL DESCRIPTION:
;  325     0705  1 !       Parses a quoted name string.  It is ASSUMED that the string has had
;  326     0706  1 !       the outer quotes stripped away ( STRIP_VERSION does this ).  This
;  327     0707  1 !       routine will check all the characters in the string are valid ANSI "a",
;  328     0708  1 !       will compress double quotes to single quotes and check that the final
;  329     0709  1 !       string is no larger then FILE_SPEC_MAX characters.  It will error
;  330     0710  1 !       exit if something is found to be incorrect.
;  331     0711  1 !
;  332     0712  1 ! CALLING SEQUENCE:
;  333     0713  1 !       PARSE_QUOTED_NAME ( ARG1, ARG2, ARG3 )
;  334     0714  1 !
;  335     0715  1 ! INPUT PARAMETERS:
;  336     0716  1 !       ARG1 - The size of the input string
;  337     0717  1 !       ARG2 - The address of the input string
;  338     0718  1 !
;  339     0719  1 ! IMPLICIT INPUTS:
;  340     0720  1 !       none
;  341     0721  1 !
;  342     0722  1 ! OUTPUT PARAMETERS:
;  343     0723  1 !       ARG3 - The address of a descriptor pointing to the output buffer.  The
;  344     0724  1 !              size field is updated.
;  345     0725  1 !              ASSUMED OUTPUT BUFFER SIZE >=  FILE_SPEC_MAX BYTES
;  346     0726  1 !
;  347     0727  1 ! IMPLICIT OUTPUTS:
;  348     0728  1 !       none
;  349     0729  1 !
;  350     0730  1 ! ROUTINE VALUE:
;  351     0731  1 !       none
;  352     0732  1 !
;  353     0733  1 ! SIDE EFFECTS:
;  354     0734  1 !       none
;  355     0735  1 !
;  356     0736  1 !--
;  357     0737  1
;  358     0738  2     BEGIN
;  359     0739  2
;  360     0740  2     EXTERNAL REGISTER                           ! so we can call ERR_EXIT
;  361     0741  2         COMMON_REG;
;  362     0742  2
;  363     0743  2     MAP
;  364     0744  2         SPEC_LEN    : LONG,
;  365     0745  2         SPEC_PTR    : REF VECTOR [ , BYTE ],
;  366     0746  2         OUT_DESC    : REF VECTOR [ 2, LONG ];
;  367     0747  2
;  368     0748  2     LOCAL
;  369     0749  2         IN_INDEX    : LONG,                     ! index into the input string
;  370     0750  2         MOVE_SIZE   : LONG,                     ! number of characters to be moved
;  371     0751  2         QUOTE_CHAR  : REF VECTOR [ , BYTE ];    ! location of a quota character
;  372     0752  2
;  373     0753  2     BIND
;  374     0754  2         OUT_INDEX   = OUT_DESC [ 0 ],
```

```
;  375     0755  2          OUT_BUF     = OUT_DESC [ 1 ] : REF VECTOR [ , BYTE ];
;  376     0756  2
;  377     0757  2      ! initialize the indices
;  378     0758  2      !
;  379     0759  2      IN_INDEX  = 0;
;  380     0760  2      OUT_INDEX = 0;
;  381     0761  2
;  382     0762  2      ! loop for each string ending with a quote
;  383     0763  2      !
;  384     0764  2      WHILE NOT CH$FAIL (
;  385     0765  2              QUOTE_CHAR = CH$FIND_CH ( .SPEC_LEN, SPEC_PTR [ .IN_INDEX ], '"' )
;  386     0766  2                                                                      ) DO
;  387     0767  3          BEGIN
;  388     0768  3
;  389     0769  3          ! calculate the number of characters to be moved
;  390     0770  3          ! and test for output buffer overflow
;  391     0771  3          !
;  392     0772  3          MOVE_SIZE = ( .QUOTE_CHAR - SPEC_PTR [ .IN_INDEX ] ) + 1;
;  393     0773  3          IF ( .MOVE_SIZE + .OUT_INDEX ) GTR FILE_SPEC_MAX
;  394     0774  3          THEN ERR_EXIT ( SS$_BADFILENAME );
;  395     0775  3
;  396     0776  3          ! adjust the number of characters remaining
;  397     0777  3          ! check that the next character is there and is a quote
;  398     0778  3          !
;  399     0779  3          SPEC_LEN  = .SPEC_LEN - ( .MOVE_SIZE + 1 );
;  400     0780  3          IF ( .SPEC_LEN LSS 0 ) OR ( .QUOTE_CHAR [ 1 ] NEQ '"' )
;  401     0781  3          THEN ERR_EXIT ( SS$_BADFILENAME );
;  402     0782  3
;  403     0783  3          ! move and test for invalid the characters   Including the quote
;  404     0784  3          ! character in the move
;  405     0785  3          !
;  406     0786  3          IF 0 NEQ MOVTUC ( MOVE_SIZE, SPEC_PTR [ .IN_INDEX  ], ESC_CHAR,
;  407     0787  3                ANSI_A_BAD, MOVE_SIZE, OUT_BUF  [ .OUT_INDEX ] )
;  408     0788  3          THEN ERR_EXIT ( SS$_BADFILENAME );
;  409     0789  3
;  410     0790  3          ! adjust the indices
;  411     0791  3          !
;  412     0792  3          IN_INDEX  = .IN_INDEX  + .MOVE_SIZE + 1;
;  413     0793  3          OUT_INDEX = .OUT_INDEX + .MOVE_SIZE;
;  414     0794  2          END;
;  415     0795  2
;  416     0796  2      ! test for output buffer overflow
;  417     0797  2      !
;  418     0798  2      IF ( .SPEC_LEN + .OUT_INDEX ) GTR FILE_SPEC_MAX
;  419     0799  2      THEN ERR_EXIT ( SS$_BADFILENAME );
;  420     0800  2
;  421     0801  2      ! move and check for invalid character
;  422     0802  2      !
;  423     0803  2      IF 0 NEQ MOVTUC ( SPEC_LEN, SPEC_PTR [ .IN_INDEX  ], ESC_CHAR, ANSI_A_BAD,
;  424     0804  2                        SPEC_LEN, OUT_BUF  [ .OUT_INDEX ] )
;  425     0805  2      THEN ERR_EXIT ( SS$_BADFILENAME );
;  426     0806  2      OUT_INDEX = .OUT_INDEX + .SPEC_LEN;
;  427     0807  2
;  428     0808  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
```

```
;  430     0809  1 GLOBAL ROUTINE STRIP_VERSION ( FILE_SPEC_LEN,
;  431     0810  1                                 FILE_SPEC_PTR,
;  432     0811  1                                 WILD_VERSION,
;  433     0812  1                                 QUOTED_NAME
;  434     0813  1                               ) : COMMON_CALL =
;  435     0814  1
;  436     0815  1 !++
;  437     0816  1 !
;  438     0817  1 ! FUNCTIONAL DESCRIPTION:
;  439     0818  1 !       This routine strips the version number from a ASCII VMS file
;  440     0819  1 !        specification, and if the file spec contains a quoted name the outer
;  441     0820  1 !        quotes are stripped by adjusting the pointer and size parameters and
;  442     0821  1 !        a check is make for a null type field.
;  443     0822  1 !
;  444     0823  1 ! CALLING SEQUENCE:
;  445     0824  1 !       version_number = STRIP_VERSION ( ARG1, ARG2, ARG3, ARG4 )
;  446     0825  1 !
;  447     0826  1 ! INPUT PARAMETERS:
;  448     0827  1 !       ARG1 - the length of the file specification
;  449     0828  1 !       ARG2 - the address of the address of the file specification buffer
;  450     0829  1 !       ARG3 - flag set if wild card version is allowed
;  451     0830  1 !
;  452     0831  1 ! IMPLICIT INPUTS:
;  453     0832  1 !       none
;  454     0833  1 !
;  455     0834  1 ! OUTPUT PARAMETERS:
;  456     0835  1 !       ARG1 - the length of the file specification minus the version number
;  457     0836  1 !                and it delimiter
;  458     0837  1 !       ARG2 - the address of the file specification
;  459     0838  1 !       ARG4 - A flag set if a quoted name was found
;  460     0839  1 !
;  461     0840  1 ! IMPLICIT OUTPUTS:
;  462     0841  1 !       none
;  463     0842  1 !
;  464     0843  1 ! ROUTINE VALUE:
;  465     0844  1 !       the version number
;  466     0845  1 !
;  467     0846  1 ! SIDE EFFECTS:
;  468     0847  1 !       none
;  469     0848  1 !
;  470     0849  1 !--
;  471     0850  1
;  472     0851  2     BEGIN
;  473     0852  2
;  474     0853  2     EXTERNAL ROUTINE                                ! ASCII decimal to binary
;  475     0854  2         LIB$CVT_DTB     : ADDRESSING_MODE ( ABSOLUTE );
;  476     0855  2
;  477     0856  2     EXTERNAL REGISTER                               ! so we can call ERR_EXIT
;  478     0857  2         COMMON_REG;
;  479     0858  2
;  480     0859  2     MAP
;  481     0860  2         FILE_SPEC_LEN   : REF VECTOR [ 1, LONG ],! the size of the file spec
;  482     0861  2         FILE_SPEC_PTR   : REF VECTOR [ 1, LONG ],! ptr to the file spec ptr
;  483     0862  2         WILD_VERSION    : BITVECTOR  [ 1 ],     ! wild version allowed flag
;  484     0863  2         QUOTED_NAME     : REF BITVECTOR  [ 1 ]; ! quoted name, return flag
;  485     0864  2
;  486     0865  2     BIND
```

```
;  487     0866  2          FILE_SPEC_BUF   = FILE_SPEC_PTR [ 0 ] : REF VECTOR [ , BYTE ];
;  488     0867  2                                                 ! point to the file spec buffer
;  489     0868  2
;  490     0869  2      LOCAL
;  491     0870  2          QUOTED_NAME_LEN : LONG,                ! length of the quoted string
;  492     0871  2          QUOTED_NAME_PTR : LONG,                ! address of the quoted string
;  493     0872  2          VERSION         : LONG,                ! the version number
;  494     0873  2          VERSION_PTR     : REF VECTOR [ , BYTE ],! ptr to the ASCII version
;  495     0874  2          VERSION_LEN     : LONG;                ! the size of ASCII version
;  496     0875  2
;  497     0876  2
;  498     0877  2      ! test for the quoted string in the name field
;  499     0878  2      !
;  500     0879  2      QUOTED_NAME [ 0 ] = FALSE;
;  501     0880  3      IF ( .FILE_SPEC_LEN [ 0 ] GEQ 2 ) AND ( .FILE_SPEC_BUF [ 0 ] EQL '"' )
;  502     0881  2      THEN
;  503     0882  3          BEGIN
;  504     0883  3          LABEL FIND_END_QUOTE;
;  505     0884  3
;  506     0885  3          FIND_END_QUOTE:
;  507     0886  4              BEGIN
;  508     0887  4
;  509     0888  4              ! Loop from the end of the string forward until a '"' is found
;  510     0889  4              ! Note: if the last character is " the name is invalid because
;  511     0890  4              !       at least a version delimitor is needed
;  512     0891  4              !
;  513     0892  4              DECR I FROM ( .FILE_SPEC_LEN [ 0 ] - 2 ) TO 1 DO
;  514     0893  4                  IF .FILE_SPEC_BUF [ .I ] EQL '"'
;  515     0894  4                  THEN
;  516     0895  5                      BEGIN
;  517     0896  5
;  518     0897  5                      ! found a quoted string save the info and set FILE_SPEC_xxx
;  519     0898  5                      !  to point to the rest of the string
;  520     0899  5                      !
;  521     0900  5                      QUOTED_NAME [ 0 ]   = TRUE;
;  522     0901  5                      QUOTED_NAME_PTR     = FILE_SPEC_BUF [ 1 ];
;  523     0902  5                      QUOTED_NAME_LEN     = .I - 1;
;  524     0903  5                      FILE_SPEC_PTR [ 0 ] = FILE_SPEC_BUF [ .I + 1 ];
;  525     0904  5                      FILE_SPEC_LEN [ 0 ] = .FILE_SPEC_LEN [ 0 ] - ( .I + 1 );
;  526     0905  5                      LEAVE FIND_END_QUOTE;
;  527     0906  4                      END;
;  528     0907  4
;  529     0908  4              ! Never found the end quote so err_exit
;  530     0909  4              !
;  531     0910  4              ERR_EXIT ( SS$_BADFILENAME );
;  532     0911  3              END;
;  533     0912  3
;  534     0913  3      ! test for a null type field which is:
;  535     0914  3      !   ;nnnnn
;  536     0915  3      !   .;nnnnn          where nnnnn is zero or more digits
;  537     0916  3      !   ..nnnnn
;  538     0917  3      !
;  539     0918  4      IF  ( .FILE_SPEC_LEN [ 0 ] GEQU 1 )
;  540     0919  4      AND ( .FILE_SPEC_BUF [ 0 ] EQL ';')
;  541     0920  3      THEN VERSION_PTR = FILE_SPEC_BUF [ 0 ]
;  542     0921  3
;  543     0922  4      ELSE IF ( .FILE_SPEC_LEN [ 0 ] GEQU 2 )
```

```
;  544     0923  4              AND ( .FILE_SPEC_BUF [ 0 ] EQL '.' )
;  545     0924  5              AND (     ( .FILE_SPEC_BUF [ 1 ] EQL '.')
;  546     0925  4                   OR ( .FILE_SPEC_BUF [ 1 ] EQL ':'))
;  547     0926  3              THEN VERSION_PTR = FILE_SPEC_BUF [ 1 ]
;  548     0927  3              ELSE ERR_EXIT ( SS$_BADFILENAME );
;  549     0928  3          END
;  550     0929  3
;  551     0930  2      ELSE
;  552     0931  3          BEGIN
;  553     0932  3
;  554     0933  3          ! We have a normal VMS file specification
;  555     0934  3          ! check for ";version number"
;  556     0935  3          !
;  557     0936  3          VERSION_PTR = CH$FIND_CH ( .FILE_SPEC_LEN [ 0 ],
;  558     0937  3                                     .FILE_SPEC_PTR [ 0 ],
;  559     0938  3                                     ';' );
;  560     0939  3          IF CH$FAIL ( .VERSION_PTR )
;  561     0940  3          THEN
;  562     0941  4              BEGIN
;  563     0942  4
;  564     0943  4              ! now look for "..version number"
;  565     0944  4              !
;  566     0945  4              VERSION_PTR = CH$FIND_CH ( .FILE_SPEC_LEN [ 0 ],
;  567     0946  4                                         .FILE_SPEC_PTR [ 0 ],
;  568     0947  4                                         '.' );
;  569     0948  4              IF CH$FAIL ( .VERSION_PTR ) THEN ERR_EXIT ( SS$_BADFILENAME );
;  570     0949  4
;  571     0950  4              ! move pass the delimitor and look for the second dot
;  572     0951  4              !
;  573     0952  4              VERSION_PTR = .VERSION_PTR + 1;
;  574     0953  4              VERSION_LEN = .FILE_SPEC_LEN [ 0 ] -
;  575     0954  4                                    ( .VERSION_PTR - .FILE_SPEC_PTR [ 0 ] );
;  576     0955  4              VERSION_PTR = CH$FIND_CH ( .VERSION_LEN, .VERSION_PTR, '.' );
;  577     0956  4              IF CH$FAIL ( .VERSION_PTR ) THEN ERR_EXIT ( SS$_BADFILEVER );
;  578     0957  3              END;
;  579     0958  2          END;
;  580     0959  2
;  581     0960  2      ! move pass the delimitor and to the ASCII decimal number
;  582     0961  2      !
;  583     0962  2      VERSION_PTR = .VERSION_PTR + 1;
;  584     0963  2
;  585     0964  2      ! ASCII to binary translation, if wildcards are allowed turn "*" into 0
;  586     0965  2      ! Note: a version number size of zero is valid and will be returned as zero
;  587     0966  2      ! Note: the version number size will never be less than zero
;  588     0967  2      !
;  589     0968  2      VERSION_LEN = .FILE_SPEC_LEN [ 0 ] -
;  590     0969  2                                    ( .VERSION_PTR - .FILE_SPEC_PTR [ 0 ] );
;  591     0970  2      IF NOT LIB$CVT_DTB ( .VERSION_LEN, .VERSION_PTR, VERSION )
;  592     0971  2      THEN IF  .WILD_VERSION [ 0 ]
;  593     0972  3           AND ( .VERSION_LEN EQL 1 )
;  594     0973  3           AND ( .VERSION_PTR [ 0 ] EQL '*' )
;  595     0974  2           THEN VERSION = 0
;  596     0975  2           ELSE ERR_EXIT ( SS$_BADFILEVER );
;  597     0976  2
;  598     0977  2      ! adjust the pointer and size fields of the name and type field
;  599     0978  2      !
;  600     0979  2      IF .QUOTED_NAME [ 0 ]
```

```
0980 2      THEN
0981 3          BEGIN
0982 3
0983 3          !  a quoted name return saved size and pointer
0984 3          !
0985 3          FILE_SPEC_LEN [ 0 ] = .QUOTED_NAME_LEN;
0986 3          FILE_SPEC_PTR [ 0 ] = .QUOTED_NAME_PTR;
0987 3          END
0988 3
0989 2      ELSE
0990 2          ! return the size of the file name and type minus the version number
0991 2          ! and its delimitor
0992 2          !
0993 2          FILE_SPEC_LEN [ 0 ] = .FILE_SPEC_LEN [ 0 ] - ( .VERSION_LEN + 1 );
0994 2
0995 2      ! return the version number
0996 2      !
0997 2      RETURN .VERSION;
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
```

```
;  621     0999  1 GLOBAL ROUTINE CALC_VERSION ( VERSION ) : NOVALUE =
;  622     1000  1
;  623     1001  1 !++
;  624     1002  1 !
;  625     1003  1 ! FUNCTIONAL DESCRIPTION:
;  626     1004  1 !       This routine calculates version number from generation and generation
;  627     1005  1 !        version.
;  628     1006  1 !
;  629     1007  1 ! CALLING SEQUENCE:
;  630     1008  1 !       CALC_VERSION ( ARG1 )
;  631     1009  1 !
;  632     1010  1 ! INPUT PARAMETERS:
;  633     1011  1 !       None
;  634     1012  1 !
;  635     1013  1 ! IMPLICIT INPUTS:
;  636     1014  1 !       HDR1 - address of hdr1 label for current file
;  637     1015  1 !
;  638     1016  1 ! OUTPUT PARAMETERS:
;  639     1017  1 !       ARG1 - address of two bytes to receive binary version number
;  640     1018  1 !
;  641     1019  1 ! IMPLICIT OUTPUTS:
;  642     1020  1 !       None
;  643     1021  1 !
;  644     1022  1 ! ROUTINE VALUE:
;  645     1023  1 !       None
;  646     1024  1 !
;  647     1025  1 ! SIDE EFFECTS:
;  648     1026  1 !       ARG1 receives binary version number
;  649     1027  1 !
;  650     1028  1 !--
;  651     1029  1
;  652     1030  2     BEGIN
;  653     1031  2
;  654     1032  2     EXTERNAL
;  655     1033  2         HDR1     : REF BBLOCK;                      ! addr of HDR1 ( EOF1 ) label
;  656     1034  2
;  657     1035  2     EXTERNAL ROUTINE
;  658     1036  2         LIB$CVT_DTB : ADDRESSING_MODE ( ABSOLUTE ); ! convert decimal to binary
;  659     1037  2
;  660     1038  2     MAP
;  661     1039  2         VERSION : REF BBLOCK [ 2 ];                 ! a pointer to a word
;  662     1040  2
;  663     1041  2     MACRO REF_WORD  = 0, 0, 16, 0 %;                ! ref to the 1st word
;  664     1042  2     MACRO REF_15BIT = 0, 0, 15, 0 %;                ! ref to the 1st fifteen bits
;  665     1043  2
;  666     1044  2     LOCAL
;  667     1045  2         GEN_NUM : LONG,                             ! binary generation number
;  668     1046  2         GEN_VER : LONG;                             ! binary generation version
;  669     1047  2
;  670     1048  2     ! clear version #
;  671     1049  2     !
;  672     1050  2     VERSION [ REF_WORD ] = 0;
;  673     1051  2
;  674     1052  2     ! binary version # = (generation number - 1) * 100 + generation version + 1
;  675     1053  2     ! convert generation number to binary
;  676     1054  2     !
;  677     1055  2     IF LIB$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               51   F0 00040  INSV    R1, #0, #15, @VERSION      ;
                                                                04   00046 1$:  RET                             ; 1067

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

```
;  691          1068  1 GLOBAL ROUTINE PARSE_FID ( FILE_SPEC_DESC,
;  692          1069  1                            FILE_VERSION    ) : COMMON_CALL =
;  693          1070  1
;  694          1071  1 !++
;  695          1072  1 !
;  696          1073  1 ! FUNCTIONAL DESCRIPTION:
;  697          1074  1 !       This routine parses the File Identifier Field in HDR1 and HDR4, then
;  698          1075  1 !       calculates the version number from the generation fields.
;  699          1076  1 !
;  700          1077  1 ! CALLING SEQUENCE:
;  701          1078  1 !       vms_spec = PARSE_FID ( ARG1, ARG2 )
;  702          1079  1 !
;  703          1080  1 ! INPUT PARAMETERS:
;  704          1081  1 !       ARG1 - descriptor to file specification output buffer
;  705          1082  1 !              ASSUMED OUTPUT BUFFER SIZE >= FILE_SPEC_MAX BYTES
;  706          1083  1 !       ARG2 - pointer to a word to hold the version number
;  707          1084  1 !
;  708          1085  1 ! IMPLICIT INPUTS:
;  709          1086  1 !       HDR1 and HDR4 label
;  710          1087  1 !
;  711          1088  1 ! OUTPUT PARAMETERS:
;  712          1089  1 !       ARG1 - descriptor to file specification output buffer
;  713          1090  1 !       ARG2 - pointer to a word to hold the version number
;  714          1091  1 !
;  715          1092  1 ! IMPLICIT OUTPUTS:
;  716          1093  1 !       none
;  717          1094  1 !
;  718          1095  1 ! ROUTINE VALUE:
;  719          1096  1 !       TRUE  - if the spec in HDR' is a "VMS type specification"
;  720          1097  1 !       FALSE - otherwise
;  721          1098  1 !
;  722          1099  1 ! SIDE EFFECTS:
;  723          1100  1 !       none
;  724          1101  1 !
;  725          1102  1 !--
;  726          1103  2     BEGIN
;  727          1104  2
;  728          1105  2     EXTERNAL REGISTER
;  729          1106  2         COMMON_REG;
;  730          1107  2
;  731          1108  2     EXTERNAL ROUTINE                                ! ASCII decimal to binary
;  732          1109  2         LIB$CVT_DTB     : ADDRESSING_MODE ( ABSOLUTE );
;  733          1110  2
;  734          1111  2     EXTERNAL
;  735          1112  2         HDR1    : REF BBLOCK,                       ! addr of HDR1 ( EOF1 ) label
;  736          1113  2         HDR4    : REF BBLOCK;                       ! addr of HDR4 ( EOF4 ) label
;  737          1114  2
;  738          1115  2     LOCAL
;  739          1116  2         MVL     : REF BBLOCK,
;  740          1117  2         FILE_ID : VECTOR [FILE_SPEC_MAX,BYTE];
;  741          1118  2
;  742          1119  2     MAP
;  743          1120  2         FILE_SPEC_DESC  : REF VECTOR [ 2, LONG ],
;  744          1121  2         FILE_VERSION    : REF VECTOR [ 1, WORD ];
;  745          1122  2
;  746          1123  2     ! calculate the version number
;  747          1124  2
```

```
;  748     1125  2     CALC_VERSION ( .FILE_VERSION );
;  749     1126  2
;  750     1127  2     ! Space fill the storage for the FILE_ID
;  751     1128  2
;  752     1129  2     CH$FILL(' ',FILE_SPEC_MAX,FILE_ID);
;  753     1130  2
;  754     1131  2     ! Copy the file name from the HDR1 and HDR4 label into one common storage
;  755     1132  2     ! area for the various checks and the translation. The HDR4 label contains
;  756     1133  2     ! part of the FILE ID if the FILEID EXT SIZE is gtr then 0.
;  757     1134  2     ! Please note that the size of the FILEID extension stored in the HDR4 field
;  758     1135  2     ! is stored in different locations depending on the ANSI version stored
;  759     1136  2     ! in the VOL1 label.
;  760     1137  2
;  761     1138  2     ! First copy the HDR1 file id
;  762     1139  2
;  763     1140  2     CH$MOVE(HD1$S_FILEID, HDR1[HD1$T_FILEID], FILE_ID[0]);
;  764     1141  2
;  765     1142  2     ! The get the HDR4 file id extension
;  766     1143  2
;  767     1144  2     MVL  = .CURRENT_VCB[VCB$L_MVL];
;  768     1145  2     IF .MVL[MVL$B_STDVER] GTR 3
;  769     1146  2         THEN
;  770     1147  3         BEGIN
;  771     1148  3             IF .HDR4[HD4$B_FILEID_EXT_SIZE] GTRU 0
;  772     1149  3                 THEN
;  773     1150  3                     CH$ OVE(.HDR4[HD4$B_FILEID_EXT_SIZE], HDR4[HD4$T_FILEID_EXT],
;  774     1151  3                                           FILE_ID[HD1$S_FILEID]);
;  775     1152  3         END
;  776     1153  2         ELSE
;  777     1154  3         BEGIN
;  778     1155  3             LOCAL SIZE;
;  779     1156  3             IF NOT LIB$CVT_DTB(HD4$S_FILEID_EXT_V3,
;  780     1157  3                                HDR4[HD4$T_FILEID_EXT_V3], SIZE)
;  781     1158  3                 THEN SIZE = 0
;  782     1159  3                 ELSE
;  783     1160  3                     CH$MOVE(.SIZE, HDR4[HD4$T_FILEID_EXT],
;  784     1161  3                             FILE_ID[HD1$S_FILEID]);
;  785     1162  2         END;
;  786     1163  2
;  787     1164  2     ! Parse the file id field in the HDR1
;  788     1165  2
;  789     1166  2     IF NOT PARSE_NAME_TYPE ( FALSE, ! do not allow wild cards
;  790     1167  2                              FILE_SPEC_MAX,
;  791     1168  2                              FILE_ID[0],
;  792     1169  2                              .FILE_SPEC_DESC )
;  793     1170  2     THEN
;  794     1171  2
;  795     1172  2         ! if the parse failed then it is a ANSI name which does not conform to
;  796     1173  2         !  VMS file spec standards so... move the field from the header to the
;  797     1174  2         !  output buffer
;  798     1175  2         !
;  799     1176  3         BEGIN
;  800     1177  3         CH$TRANSLATE ( ANSI_A_GOOD, FILE_SPEC_MAX, FILE_ID[0], ' ',
;  801     1178  3                                .FILE_SPEC_DESC [ 0 ], .FILE_SPEC_DESC [ 1 ] );
;  802     1179  3         FILE_SPEC_DESC [ 0 ] = FILE_SPEC_MAX;
;  803     1180  3         RETURN FALSE;
;  804     1181  3         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   AE      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
```

```
;  810          1186  1 GLOBAL ROUTINE PARSE_PATTERN_SPEC ( FILE_SPEC_DESC,
;  811          1187  1                                     FILE_VERSION
;  812          1188  1                                   ) : COMMON_CALL =
;  813          1189  1
;  814          1190  1 !++
;  815          1191  1 !
;  816          1192  1 ! FUNCTIONAL DESCRIPTION:
;  817          1193  1 !       This routine parses a string given to the ACP by the user which is
;  818          1194  1 !        used as a pattern string matching string by the find operation
;  819          1195  1 !
;  820          1196  1 ! CALLING SEQUENCE:
;  821          1197  1 !       vms_spec = PARSE_PATTERN_SPEC ( ARG1, ARG2 )
;  822          1198  1 !
;  823          1199  1 ! INPUT PARAMETERS:
;  824          1200  1 !       ARG1 - descriptor to file specification output buffer
;  825          1201  1 !              ASSUMED OUTPUT BUFFER SIZE >= FILE_SPEC_MAX BYTES
;  826          1202  1 !       ARG2 - pointer to a word to hold the version number
;  827          1203  1 !
;  828          1204  1 ! IMPLICIT INPUTS:
;  829          1205  1 !       IO_PACKET
;  830          1206  1 !
;  831          1207  1 ! OUTPUT PARAMETERS:
;  832          1208  1 !       ARG1 - descriptor to file specification output buffer
;  833          1209  1 !       ARG2 - pointer to a word to hold the version number
;  834          1210  1 !
;  835          1211  1 ! IMPLICIT OUTPUTS:
;  836          1212  1 !       none
;  837          1213  1 !
;  838          1214  1 ! ROUTINE VALUE:
;  839          1215  1 !       Bit 0 set - if the spec is wild for quoted name. ie "*.*" or "*."
;  840          1216  1 !       Bit 1 set - if it was a VMS spec
;  841          1217  1 !
;  842          1218  1 ! SIDE EFFECTS:
;  843          1219  1 !       none
;  844          1220  1 !
;  845          1221  1 !--
;  846          1222  2     BEGIN
;  847          1223  2
;  848          1224  2     EXTERNAL REGISTER                          ! so we can call ERR_EXIT
;  849          1225  2         COMMON_REG;
;  850          1226  2
;  851          1227  2     EXTERNAL
;  852          1228  2         IO_PACKET       : REF BBLOCK;
;  853          1229  2
;  854          1230  2     MAP
;  855          1231  2         FILE_SPEC_DESC  : REF VECTOR [ 2, LONG ],
;  856          1232  2         FILE_VERSION    : REF VECTOR [ 1, WORD ];
;  857          1233  2
;  858          1234  2     LITERAL
;  859          1235  2         WILD_QUOTED     = %B'01',
;  860          1236  2         IS_VMS_SPEC     = %B'10';
;  861          1237  2
;  862          1238  2     LOCAL
;  863          1239  2         ABD             : REF BBLOCKVECTOR [ , ABD$C_LENGTH ],
;  864          1240  2                                           ! ACP buffer descriptors
;  865          1241  2         ASCII_NAME_SPEC : BITVECTOR [ 1 ] PRESET ( [ 0 ] = FALSE ),
;  866          1242  2                                           ! flag set if using ASCNAME attribute
```

```
;  867     1243  2          ATTR_PTR        : LONG,          ! pointer into the ACP buffer descr
;  868     1244  2          ATTR_LENGTH     : LONG,          ! length of the attribute
;  869     1245  2          INPUT_ADDR      : LONG,          ! address of input file name string
;  870     1246  2          INPUT_LENGTH    : LONG,          ! length of input file name string
;  871     1247  2          QUOTED_NAME     : BITVECTOR [ 1 ];! flag set if the file spec is in ''s
;  872     1248  2
;  873     1249  2      ABD = .BBLOCK [ .IO_PACKET [ IRP$L_SVAPTE ], AIB$L_DESCRIPT ];
;  874     1250  2
;  875     1251  2      ! scan to see if the find should be done by the ASCII name attribute
;  876     1252  2      !
;  877     1253  2      DECRU I FROM ( .IO_PACKET[IRP$W_BCNT] - 1 ) TO ABD$C_ATTRIB DO
;  878     1254  3          BEGIN
;  879     1255  3
;  880     1256  3          ! find the attribute code and see if it matches ASCII name
;  881     1257  3          !
;  882     1258  3          ATTR_PTR = .ABD [ .I, ABD$W_TEXT ] + ABD [ .I, ABD$W_TEXT ];
;  883     1259  3          IF .(.ATTR_PTR)<0, 8> EQL ATR$C_ASCNAME
;  884     1260  3          THEN
;  885     1261  4              BEGIN
;  886     1262  4
;  887     1263  4              ! set up the size of the attribute
;  888     1264  4              !
;  889     1265  4              ATTR_LENGTH = .ABD[.I, ABD$W_COUNT];
;  890     1266  4
;  891     1267  4              ! check length of file name
;  892     1268  4              !
;  893     1269  4              IF .ATTR_LENGTH LEQ FILE_SPEC_MAX
;  894     1270  4              THEN
;  895     1271  5                  BEGIN
;  896     1272  5
;  897     1273  5                  ! set up the pointer to the attribute text
;  898     1274  5                  !
;  899     1275  5                  ATTR_PTR = .ATTR_PTR + 1;
;  900     1276  5
;  901     1277  5                  ! translate into upper case and check for invalid characters
;  902     1278  5                  !
;  903     1279  5                  IF 0 EQL MOVTUC( ATTR_LENGTH, .ATTR_PTR, ESC_CHAR, ANSI_A_BAD,
;  904     1280  5                                   FILE_SPEC_DESC [ 0 ], .FILE_SPEC_DESC [ 1 ] )
;  905     1281  5                  THEN
;  906     1282  6                      BEGIN
;  907     1283  6
;  908     1284  6                      ! we have a valid ANSI file spec to the find operation on
;  909     1285  6                      !
;  910     1286  6                      ASCII_NAME_SPEC [ 0 ] = TRUE;
;  911     1287  6                      FILE_SPEC_DESC [ 0 ] = .ATTR_LENGTH;
;  912     1288  6                      EXITLOOP;
;  913     1289  5                      END;
;  914     1290  5
;  915     1291  4                  END;
;  916     1292  4
;  917     1293  3              END;
;  918     1294  3
;  919     1295  2          END;
;  920     1296  2
;  921     1297  2      ! pick up the name pattern string from the name attribute
;  922     1298  2      !
;  923     1299  2      INPUT_LENGTH = .ABD [ ABD$C_NAME, ABD$W_COUNT ];
```

```
;  924     1300  2      INPUT_ADDR   = .ABD [ ABD$C_NAME, ABD$W_TEXT ]
;  925     1301  2                   + ABD [ ABD$C_NAME, ABD$W_TEXT ] + 1;
;  926     1302  2
;  927     1303  2      ! strip the version number
;  928     1304  2      !
;  929     1305  2      FILE_VERSION [ 0 ] = STRIP_VERSION ( INPUT_LENGTH,
;  930     1306  2                                           INPUT_ADDR,
;  931     1307  2                                           TRUE,  ! allow wild cards
;  932     1308  2                                           QUOTED_NAME
;  933     1309  2                                         );
;  934     1310  2
;  935     1311  2      ! parse the filename if not from ASCNAME
;  936     1312  2      !
;  937     1313  2      IF NOT .ASCII_NAME_SPEC [ 0 ]
;  938     1314  2      THEN
;  939     1315  3          BEGIN
;  940     1316  3
;  941     1317  3          ! call the correct parse routine
;  942     1318  3          !
;  943     1319  3          IF NOT .QUOTED_NAME [ 0 ]
;  944     1320  3          THEN
;  945     1321  4              BEGIN
;  946     1322  4
;  947     1323  4              BIND
;  948     1324  4                  STAR_DOT_STAR = UPLIT ( '*.*' ),
;  949     1325  4                  STAR_DOT      = STAR_DOT_STAR;
;  950     1326  4
;  951     1327  4              ! parse the filename into name and type
;  952     1328  4              !
;  953     1329  4              IF NOT PARSE_NAME_TYPE ( TRUE, ! allow wild cards
;  954     1330  4                                       .INPUT_LENGTH,
;  955     1331  4                                       .INPUT_ADDR,
;  956     1332  4                                       .FILE_SPEC_DESC )
;  957     1333  4              THEN ERR_EXIT ( SS$_BADFILENAME );
;  958     1334  4
;  959     1335  4              ! test if "*.*" or "*." ( match all non-VMS file specs on tape )
;  960     1336  4              !
;  961     1337  4              IF CH$EQL ( 3,                          STAR_DOT_STAR,
;  962     1338  4                          .FILE_SPEC_DESC [ 0 ], .FILE_SPEC_DESC [ 1 ], 0 )
;  963     1339  4              THEN RETURN IS_VMS_SPEC OR WILD_QUOTED
;  964     1340  4              ELSE
;  965     1341  5                  BEGIN
;  966     1342  5                  IF CH$EQL ( 2,                          STAR_DOT,
;  967     1343  5                              .FILE_SPEC_DESC [ 0 ], .FILE_SPEC_DESC [ 1 ], 0 )
;  968     1344  5                  THEN RETURN IS_VMS_SPEC OR WILD_QUOTED
;  969     1345  5                  ELSE RETURN IS_VMS_SPEC
;  970     1346  4                  END;
;  971     1347  4              END
;  972     1348  4
;  973     1349  3          ELSE
;  974     1350  4              BEGIN
;  975     1351  4
;  976     1352  4              ! Parse the quoted name string
;  977     1353  4              !
;  978     1354  4              PARSE_QUOTED_NAME ( .INPUT_LENGTH,
;  979     1355  4                                  .INPUT_ADDR,
;  980     1356  4                                  .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   @4(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
```

```
;  989          1364  1 GLOBAL ROUTINE RESULTANT_STRING ( VMS_NAME,
;  990          1365  1                                   FILE_SPEC_LEN,
;  991          1366  1                                   FILE_SPEC_PTR,
;  992          1367  1                                   VERSION       ) : COMMON_CALL NOVALUE =
;  993          1368  1
;  994          1369  1 !++
;  995          1370  1 !
;  996          1371  1 ! FUNCTIONAL DESCRIPTION:
;  997          1372  1 !       This routine prepares and returns the resultant string name
;  998          1373  1 !
;  999          1374  1 ! CALLING SEQUENCE:
; 1000          1375  1 !       RESULTANT_STRING ( ARG1, ARG2, ARG3, ARG4 )
; 1001          1376  1 !
; 1002          1377  1 ! INPUT PARAMETERS:
; 1003          1378  1 !       ARG1 - flag ( set if a VMS file spec )
; 1004          1379  1 !       ARG2 - length of file specifiation ( name dot type )
; 1005          1380  1 !       ARG3 - pointer to file specification ( name dot type )
; 1006          1381  1 !       ARG4 - the file version number
; 1007          1382  1 !
; 1008          1383  1 ! IMPLICIT INPUTS:
; 1009          1384  1 !       none
; 1010          1385  1 !
; 1011          1386  1 ! OUTPUT PARAMETERS:
; 1012          1387  1 !       none
; 1013          1388  1 !
; 1014          1389  1 ! IMPLICIT OUTPUTS:
; 1015          1390  1 !       none
; 1016          1391  1 !
; 1017          1392  1 ! ROUTINE VALUE:
; 1018          1393  1 !       none
; 1019          1394  1 !
; 1020          1395  1 ! SIDE EFFECTS:
; 1021          1396  1 !       resulant string is returned to the user
; 1022          1397  1 !
; 1023          1398  1 !--
; 1024          1399  2     BEGIN
; 1025          1400  2
; 1026          1401  2     EXTERNAL REGISTER COMMON_REG;                ! so we can call ERR_EXIT
; 1027          1402  2
; 1028          1403  2     EXTERNAL WORK_AREA;                          ! general use scratch area
; 1029          1404  2
; 1030          1405  2     EXTERNAL ROUTINE
; 1031          1406  2         SYS$FAO          : ADDRESSING_MODE ( ABSOLUTE ); ! format the string
; 1032          1407  2
; 1033          1408  2     ! map the input agruments into data structures
; 1034          1409  2     !
; 1035          1410  2     MAP
; 1036          1411  2         VMS_NAME         : BITVECTOR [ 1 ],
; 1037          1412  2         FILE_SPEC_LEN    : LONG,
; 1038          1413  2         FILE_SPEC_PTR    : REF VECTOR [ , BYTE ],
; 1039          1414  2         VERSION          : LONG;
; 1040          1415  2
; 1041          1416  2     LITERAL
; 1042          1417  2         RESULTANT_MAX_LEN = 164;                 ! max size of resultant name
; 1043          1418  2
; 1044          1419  2     LOCAL
; 1045          1420  2         OUT_INDEX        : LONG,                 ! index into output buffer
```

```
;1046     1421  2          RESULTANT_DESC  : VECTOR [ 2, LONG ];   ! descriptor for resultant name
;1047     1422  2
;1048     1423  2      BIND
;1049     1424  2          RESULTANT_NAME  = WORK_AREA : VECTOR [ RESULTANT_MAX_LEN, BYTE ],
;1050     1425  2          FAO_VERSION     = DESCRIPTOR ( ';!SW' ),
;1051     1426  2          STRIP_SIZE      = OUT_INDEX;            ! redefine register
;1052     1427  2
;1053     1428  2
;1054     1429  2      ! string any trailing spaces from the file specification
;1055     1430  2      !
;1056     1431  2      STRIP_SIZE = 0;
;1057     1432  2      DECR I FROM ( .FILE_SPEC_LEN - 1 ) TO 0 DO
;1058     1433  2          IF .FILE_SPEC_PTR [ .I ] NEQ ' '
;1059     1434  2          THEN
;1060     1435  3              BEGIN
;1061     1436  3              STRIP_SIZE = .I + 1;
;1062     1437  3              EXITLOOP;
;1063     1438  3              END;
;1064     1439  2      FILE_SPEC_LEN = .STRIP_SIZE;
;1065     1440  2
;1066     1441  2      ! move the ( name and type ) or ANSI file id into the output buffer
;1067     1442  2      !
;1068     1443  2      IF .VMS_NAME [ 0 ]
;1069     1444  2      THEN
;1070     1445  3          BEGIN
;1071     1446  3
;1072     1447  3          ! just move the fields if a VMS spec
;1073     1448  3          !
;1074     1449  3          CH$MOVE ( .FILE_SPEC_LEN, .FILE_SPEC_PTR, RESULTANT_NAME );
;1075     1450  3          OUT_INDEX = .FILE_SPEC_LEN;
;1076     1451  3          END
;1077     1452  3
;1078     1453  2      ELSE
;1079     1454  3          BEGIN
;1080     1455  3
;1081     1456  3          ! ANSI spec is placed in quotes and all quotes goto double quotes
;1082     1457  3          !
;1083     1458  3          LOCAL
;1084     1459  3              IN_INDEX    : LONG, ! index into the input string
;1085     1460  3              MOVE_SIZE   : LONG, ! the number of characters to be moved
;1086     1461  3              QUOTE_CHAR  : LONG; ! location of a quota character
;1087     1462  3
;1088     1463  3          ! stuff initial quote into output buffer
;1089     1464  3          !
;1090     1465  3          RESULTANT_NAME [ 0 ] = '"';
;1091     1466  3
;1092     1467  3          ! initialize the indices
;1093     1468  3          !
;1094     1469  3          IN_INDEX  = 0;
;1095     1470  3          OUT_INDEX = 1;
;1096     1471  3
;1097     1472  3          ! loop for each string ending with a quote
;1098     1473  3          !
;1099     1474  3          WHILE NOT CH$FAIL (
;1100     1475  3                  QUOTE_CHAR = CH$FIND_CH ( .FILE_SPEC_LEN,
;1101     1476  3                                            FILE_SPEC_PTR [ .IN_INDEX ],
;1102     1477  3                                            '"' )
```

```
;  1103          1478  3                                                    ) DO
;  1104          1479  4          BEGIN
;  1105          1480  4
;  1106          1481  4          ! calculate the number of characters to be moved
;  1107          1482  4          ! and move the characters including the quote character
;  1108          1483  4          !
;  1109          1484  4          MOVE_SIZE = ( .QUOTE_CHAR - FILE_SPEC_PTR [ .IN_INDEX ] ) + 1;
;  1110          1485  4          CH$MOVE ( .MOVE_SIZE,
;  1111          1486  4                    FILE_SPEC_PTR  [ .IN_INDEX ],
;  1112          1487  4                    RESULTANT_NAME [ .OUT_INDEX ] );
;  1113          1488  4
;  1114          1489  4          ! adjust the indices, and the number of characters remaining
;  1115          1490  4          !
;  1116          1491  4          IN_INDEX      = .IN_INDEX      + .MOVE_SIZE;
;  1117          1492  4          OUT_INDEX     = .OUT_INDEX     + .MOVE_SIZE;
;  1118          1493  4          FILE_SPEC_LEN = .FILE_SPEC_LEN - .MOVE_SIZE;
;  1119          1494  4
;  1120          1495  4          ! stuff extra quote characters, bump index
;  1121          1496  4          !
;  1122          1497  4          RESULTANT_NAME [ .OUT_INDEX ]     = '"';
;  1123          1498  4          OUT_INDEX = .OUT_INDEX + 1;
;  1124          1499  3          END;
;  1125          1500  3
;  1126          1501  3      ! move the remaining character into the output buffer
;  1127          1502  3      !
;  1128          1503  3      CH$MOVE ( .FILE_SPEC_LEN,
;  1129          1504  3                FILE_SPEC_PTR  [ .IN_INDEX ],
;  1130          1505  3                RESULTANT_NAME [ .OUT_INDEX ] );
;  1131          1506  3      OUT_INDEX = .OUT_INDEX + .FILE_SPEC_LEN;
;  1132          1507  3
;  1133          1508  3      ! stuff final quote characters and null type, bump index
;  1134          1509  3      !
;  1135          1510  3          RESULTANT_NAME [ .OUT_INDEX ]     = '"';
;  1136          1511  3          RESULTANT_NAME [ .OUT_INDEX + 1 ] = '.';
;  1137          1512  3          OUT_INDEX = .OUT_INDEX + 2;
;  1138          1513  2          END;
;  1139          1514  2
;  1140          1515  2      ! adjust the descriptor to point to the remaining of the output buffer
;  1141          1516  2      ! and FAO the version number into the output buffer
;  1142          1517  2      !
;  1143          1518  2      RESULTANT_DESC [ 0 ] = RESULTANT_MAX_LEN - .OUT_INDEX;
;  1144          1519  2      RESULTANT_DESC [ 1 ] = RESULTANT_NAME [ .OUT_INDEX ];
;  1145          1520  2      SYS$FAO ( FAO_VERSION, RESULTANT_DESC, RESULTANT_DESC, .VERSION );
;  1146          1521  2
;  1147          1522  2      ! point the descriptor to the entire file spec in the output buffer
;  1148          1523  2      !
;  1149          1524  2      RESULTANT_DESC [ 0 ] = .RESULTANT_DESC [ 0 ] + .OUT_INDEX;
;  1150          1525  2      RESULTANT_DESC [ 1 ] = RESULTANT_NAME;
;  1151          1526  2
;  1152          1527  2      ! return the output buffer to the user
;  1153          1528  2      !
;  1154          1529  3      IF NOT KERNEL_CALL ( RETURN_RESULTANT_SPEC, RESULTANT_DESC )
;  1155          1530  2      THEN ERR_EXIT ( );
;  1156          1531  2
;  1157          1532  1      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$:     MOVB    #34, RESULTANT_NAME                           ; 1465
                          56             01  7D 00039        MOVQ    #1, OUT_INDEX                                 ; 1470
                    52    57    0C       C1 0003C 5$:     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
```

```
;1159     1533  1 ROUTINE RETURN_RESULTANT_SPEC ( FILE_NAME_DESC ) =
;1160     1534  1
;1161     1535  1 !++
;1162     1536  1 !
;1163     1537  1 ! FUNCTIONAL DESCRIPTION:
;1164     1538  1 !       This routine enables the write-back of the result file name string.
;1165     1539  1 !
;1166     1540  1 ! CALLING SEQUENCE:
;1167     1541  1 !       RETURN_RESULTANT_SPEC ( ARG1 )
;1168     1542  1 !       must be called in kernel access mode
;1169     1543  1 !
;1170     1544  1 ! INPUT PARAMETERS:
;1171     1545  1 !       ARG1 - address of file name string descriptor
;1172     1546  1 !
;1173     1547  1 ! IMPLICIT INPUTS:
;1174     1548  1 !       IO_PACKET - address of current request packet
;1175     1549  1 !
;1176     1550  1 ! OUTPUT PARAMETERS:
;1177     1551  1 !       none
;1178     1552  1 !
;1179     1553  1 ! IMPLICIT OUTPUTS:
;1180     1554  1 !       none
;1181     1555  1 !
;1182     1556  1 ! ROUTINE VALUE:
;1183     1557  1 !       0 - failure
;1184     1558  1 !       1 - success
;1185     1559  1 !
;1186     1560  1 ! SIDE EFFECTS:
;1187     1561  1 !       enables write back
;1188     1562  1 !
;1189     1563  1 ! USER ERRORS:
;1190     1564  1 !       SS$_RESULTOVF - file name string overflows buffer
;1191     1565  1 !--
;1192     1566  1
;1193     1567  2     BEGIN
;1194     1568  2
;1195     1569  2     LOCAL
;1196     1570  2         SIZE,                                   ! size of write back data
;1197     1571  2         ABD     : REF BBLOCKVECTOR [ , ABD$C_LENGTH ];! addr of buffer descr
;1198     1572  2
;1199     1573  2     MAP FILE_NAME_DESC  : REF VECTOR [ 2, LONG ];! descr of file name string
;1200     1574  2
;1201     1575  2     EXTERNAL
;1202     1576  2         IO_PACKET       : REF BBLOCK;           ! address of current IRP
;1203     1577  2
;1204     1578  2     ! get buffer descriptor address address of buffer descriptors
;1205     1579  2     !
;1206     1580  2     ABD = .BBLOCK [ .IO_PACKET [ IRP$L_SVAPTE ], AIB$L_DESCRIPT ];
;1207     1581  2
;1208     1582  2     ! return length of result string
;1209     1583  2     !
;1210     1584  2     SIZE = .FILE_NAME_DESC [ 0 ];
;1211     1585  2
;1212     1586  2     IF .ABD [ ABD$C_RESL, ABD$W_COUNT ] GEQU 2
;1213     1587  2     THEN
;1214     1588  3         ( .ABD [ ABD$C_RESL, ABD$W_TEXT ] +
;1215     1589  2           ABD [ ABD$C_RESL, ABD$W_TEXT ] + 1 ) < 0, 16 > = .SIZE < 0, 16 >;
```

```
;  1216          1590  2
;  1217          1591  2
;  1218          1592  2      IF .ABD [ ABD$C_RES, ABD$W_COUNT ] NEQ 0
;  1219          1593  2      THEN
;  1220          1594  3          BEGIN
;  1221          1595  3
;  1222          1596  3          ! if result string overflows buffer, return error
;  1223          1597  3          !
;  1224          1598  3          IF .SIZE GTRU .ABD [ ABD$C_RES, ABD$W_COUNT ]
;  1225          1599  3          THEN
;  1226          1600  4              BEGIN
;  1227          1601  4              ERROR ( SS$_RESULTOVF );
;  1228          1602  4              RETURN 0;
;  1229          1603  4              END
;  1230          1604  4
;  1231          1605  3          ELSE
;  1232          1606  3              CH$COPY ( .SIZE, .FILE_NAME_DESC [ 1 ], 0,
;  1233          1607  3                        .ABD [ ABD$C_RES, ABD$W_COUNT ],
;  1234          1608  4                      ( .ABD [ ABD$C_RES, ABD$W_TEXT ] +
;  1235          1609  3                        ABD [ ABD$C_RES, ABD$W_TEXT ] + 1 ));
;  1236          1610  2          END;
;  1237          1611  2
;  1238          1612  2      RETURN 1;
;  1239          1613  2
;  1240          1614  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