```
DDDDDDDDDDDD        CCCCCCCCCCCC  LLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDDDDDDDDDDD         CCCCCCCCCCC  LLLLLLLLLLLLLLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLLLLLLLLLLLLLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLLLLLLLLLLLLLL
```

```
**FILE**ID**PARSENT                              I  3
```

```
PPPPPPPP      AAAAAA    RRRRRRRR      SSSSSSSS  EEEEEEEEEE  NN      NN  TTTTTTTTTT
PPPPPPPP      AAAAAA    RRRRRRRR      SSSSSSSS  EEEEEEEEEE  NN      NN  TTTTTTTTTT
PP      PP  AA      AA  RR      RR  SS            EE          NN      NN      TT
PP      PP  AA      AA  RR      RR  SS            EE          NN      NN      TT
PP      PP  AA      AA  RR      RR  SS            EE          NNNN    NN      TT
PP      PP  AA      AA  RR      RR  SS            EE          NNNN    NN      TT
PPPPPPPP    AA      AA  RRRRRRRR      SSSSSS    EEEEEEEE    NN  NN  NN      TT
PPPPPPPP    AA      AA  RRRRRRRR      SSSSSS    EEEEEEEE    NN  NN  NN      TT
PP          AAAAAAAAAA  RR  RR          SS  EE          NN    NNNN      TT
PP          AAAAAAAAAA  RR  RR          SS  EE          NN    NNNN      TT
PP          AA      AA  RR    RR        SS  EE          NN      NN      TT          ....
PP          AA      AA  RR    RR        SS  EE          NN      NN      TT          ....
PP          AA      AA  RR      RR  SSSSSSSS    EEEEEEEEEE  NN      NN      TT          ....
PP          AA      AA  RR      RR  SSSSSSSS    EEEEEEEEEE  NN      NN      TT          ....

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
```

Table of contents

```
0000          .TITLE  PARSENT - PARSE PARAMETERS AND QUALIFIERS
0000          .IDENT  'V04-000'
0000
0000    ;
0000    ;****************************************************************************
0000    ;*                                                                          *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000    ;*  ALL RIGHTS RESERVED.                                                    *
0000    ;*                                                                          *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000    ;*  TRANSFERRED.                                                            *
0000    ;*                                                                          *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    ;*  CORPORATION.                                                            *
0000    ;*                                                                          *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    ;*                                                                          *
0000    ;*                                                                          *
0000    ;****************************************************************************
0000    ;
0000    ; D. N. CUTLER 24-MAR-77
0000    ;
0000    ; PROCESS QUALIFIER SPECIFICATION
0000    ;
0000    ; MODIFICATIONS:
0000    ;
0000    ;       V03-018 HWS0027         Harold Schultz          09-Mar-1984
0000    ;               Check for error when expanding an expression.
0000    ;
0000    ;       V03-017 HWS0021         Harold Schultz          06-Mar-1984
0000    ;               Don't issue warning message about ignored qualifiers due
0000    ;               to a syntax change when the only previous qualifiers
0000    ;               were the one responsible for the syntax change.
0000    ;               Fix typo's.
0000    ;
0000    ;       V03-016 PCG0019         Peter George            18-Nov-1983
0000    ;               Allow hyphens in filenames.
0000    ;
0000    ;       V03-015 PCG0018         Peter George            17-Aug-1983
0000    ;               Disable indirection when looking ahead for keyword values.
0000    ;
0000    ;       V03-014 PCG0017         Peter George            17-Jul-1983
0000    ;               Signal ignored qualifier message.
0000    ;               Support null node names.
0000    ;
0000    ;       V03-013 PCG0016         Peter George            27-Jun-1983
0000    ;               Fix bug in $REST_OF_LINE.
0000    ;
0000    ;       V03-012 PCG0015         Peter George            15-Jun-1983
0000    ;               Add support for $FILE, $EXPRESSION, $QUOTED_STRING, and
```

```
0000    ;               $PARENTHESIZED_VALUE.  Rename old $PARENTHESIZED_VALUE
0000    ;               $ACL.  Improve error reporting.
0000    ;
0000    ;       V03-011 PCG0014         Peter George            27-May-1983
0000    ;               Add support for $DELTATIME.
0000    ;               Remove DCL$SORT_TOKENS.
0000    ;
0000    ;       V03-010 PCG0013         Peter George            06-May-1983
0000    ;               Add DCL$SORT_TOKENS.
0000    ;
0000    ;       V03-009 PCG0012         Peter George            30-Apr-1983
0000    ;               Move disallow logic into DISALLOW.
0000    ;               Add WRK_B_PARMSUM.
0000    ;
0000    ;       V03-008 PCG0011         Peter George            20-Apr-1983
0000    ;               Undo the part of the last fix dealing with negated
0000    ;               entities and syntax checks.
0000    ;
0000    ;       V03-007 PCG0010         Peter George            07-Apr-1983
0000    ;               Add more parameter limit checks.
0000    ;               Modify ambiguous verb check.
0000    ;               Do not change syntax on negated qualifiers or keywords.
0000    ;
0000    ;       V03-006 PCG0009         Peter George            15-Feb-1983
0000    ;               Convert to new structure level.
0000    ;               Pass entity block address to DCL$GENDESCR.
0000    ;               Do not allow values on negated qualifiers.
0000    ;               Add support for $NUMBER, $PARENTHESIZED_VALUE,
0000    ;               and $DATETIME.
0000    ;
0000    ;       V03-005 PCG0008         Peter George            15-Jan-1983
0000    ;               Make NOQUALIFIERS work.
0000    ;
0000    ;       V03-004 PCG0007         Peter George            22-Nov-1982
0000    ;               Use prompt descriptor instead of WRK_L_PROMPT.
0000    ;
0000    ;       V03-003 PCG0006         Peter George            29-Oct-1982
0000    ;               Insert CRLF into DCL prompts.
0000    ;
0000    ;       V03-002 PCG0005         Peter George            19-Oct-1982
0000    ;               Call user input routine with address of string descriptor
0000    ;               instead of address of ascic string.
0000    ;               Use constant for maximum prompt string length.
0000    ;               Do the right thing when a CTRL/Z is entered.
0000    ;               Do not zero out parameters before a syntax change.
0000    ;               Check full spelling of negated keywords.
0000    ;               Zero R6 after parsing $REST_OF_LINE value.
0000    ;
0000    ;       V03-001 PCG0004         Peter George            30-Sep-1982
0000    ;               Add keyword parsing.  Create more common parsing
0000    ;               routines.  Add prompting for CLI$DCL_PARSE.
0000    ;
0000    ;---
0000
```

```
                     0000  ;
                     0000  ; MACRO LIBRARY CALLS
                     0000  ;
                     0000             PRCDEF                        ; DEFINE PROCESS WORK AREA
                     0000             WRKDEF                        ; DEFINE COMMAND WORK AREA
                     0000             PTRDEF                        ; DEFINE RESULT PARSE DESCRIPTOR
                     0000             $CLIMSGDEF                    ; CLI MESSAGES
                     0000             $FSCNDEF                      ; DEFINE FILE SCAN SYMBOLS
                     0000             $$CLITABDEF                   ; DEFINE TABLE STRUCTURES
                     0000
                 00000000             .PSECT  DCL$ZCODE,BYTE,RD,NOWRT
                     0000
          0000001A   0000  CTRLZ   = 26                           ; CONTROL/Z CHARACTER
                     0000
                     0000  ;
                     0000  ; DEFINE LOCAL FLAGS
                     0000  ;
          00000000   0000  V_NEGAT = 0
          00000001   0000  M_NEGAT = 1
          00000001   0000  V_KEYWD = 1
          00000002   0000  M_KEYWD = 2
          00000002   0000  V_SYNTAX = 2
          00000004   0000  M_SYNTAX = 4
          00000003   0000  V_AMBIG = 3
          00000008   0000  M_AMBIG = 8
                     0000
```

```
                    0000         .SBTTL  PARSE COMMAND QUALIFIERS AND PARAMETERS
                    0000    ;---
                    0000    ; DCL$PARSE_COMMAND - PARSE COMMAND QUALIFIERS AND PARAMETERS
                    0000    ;
                    0000    ; THIS ROUTINE IS CALLED AFTER THE COMMAND VERB HAS BEEN PARSED, TO PARSE
                    0000    ; WHATEVER QUALIFIERS AND PARAMETERS MAY APPEAR ON THE COMMAND LINE.
                    0000    ;
                    0000    ; INPUTS:
                    0000    ;
                    0000    ;       R10 =   ADDRESS OF COMMAND WORK AREA
                    0000    ;       R11 =   ADDRESS OF PROCESS WORK AREA (FOR SUPERVISOR MODE PARSING)
                    0000    ;
                    0000    ; OUTPUTS:
                    0000    ;
                    0000    ;       R0 =    STATUS
                    0000    ;       NO REGISTERS ARE SAVED
                    0000    ;
                    0000    ;+++
                    0000
                    0000    DCL$PARSE_COMMAND::
   0040 8F     A8   0000            BISW    #WRK_M_VERB,-           ;SET VERB PROCESSING FLAG
     F0 AA          0004                    WRK_Q_FLAGS(R10)
                    0006
                    0006    ;
                    0006    ; PROCESS ALL LEADING COMMAND QUALIFIERS
                    0006    ;
                    0006    PARSE_QUAL:
     FFF7'     30   0006            BSBW    DCL$SETCHAR             ;PEEK AT NEXT CHARACTER IN INPUT BUFFER
50     2F      91   0009            CMPB    #^A'/',R0               ;QUALIFIER?
       0C      12   000C            BNEQ    PARSE_PARMS             ;IF NEQ NO
     FFEF'     30   000E            BSBW    DCL$MARK                ;MARK CURRENT PARSE POSITION
     0143      30   0011            BSBW    DCL$PROCQUAL            ;PROCESS QUALIFIER
    EF 50      E8   0014            BLBS    R0,PARSE_QUAL           ;IF LBS SUCCESSFUL COMPLETION
     011A      31   0017            BRW     EXIT                    ;UNSUCCESSFUL, EXIT WITH STATUS
                    001A
                    001A    ;
                    001A    ; PROCESS EACH COMMAND PARAMETER AND ITS ASSOCIATED QUALIFIERS
                    001A    ;
                    001A    PARSE_PARMS:
   0040 8F     AA   001A    10$:    BICW    #WRK_M_VERB,-           ;CLEAR VERB PROCESSING
     F0 AA          001E                    WRK_Q_FLAGS(R10)        ;
58   C6 AA     D0   0020            MOVL    WRK_L_PROPTR(R10),R8    ;GET ADDRESS OF P1 ENTITY BLOCK
       0F      13   0024            BEQL    30$                     ;BRANCH IF NONE
50   CE AA     9A   0026            MOVZBL  WRK_B_PARMCNT(R10),R0   ;GET PARAMETER # BEING PARSED
       09      13   002A            BEQL    30$                     ;BRANCH IF NO LOOP NEEDED
     08 A8     C1   002C    20$:    ADDL3   ENT_L_NEXT(R8),-        ;GET ADDRESS OF NEXT ENTITY BLOCK
58   DE AA          002F                    WRK_L_TAB_VEC(R10),R8   ;
     F7 50     F5   0032            SOBGTR  R0,20$                  ;LOOP UNTIL PARAMETER FOUND
                    0035
     FFC8'     30   0035    30$:    BSBW    DCL$MARK                ;MARK CURRENT PARSE POSITION
     FFC5'     30   0038            BSBW    DCL$SETCHAR             ;PEEK AT NEXT CHARACTER IN INPUT BUFFER
       74      13   003B            BEQL    END_OF_LINE             ;IF EQL END OF LINE
50     20      91   003D            CMPB    #^A' ',R0               ;BLANK?
       63      12   0040            BNEQ    130$                    ;IF NEQ NO
     CE AA     96   0042            INCB    WRK_B_PARMCNT(R10)      ;INCREMENT COUNT OF PARAMETER
     CE AA     91   0045            CMPB    WRK_B_PARMCNT(R10),-    ;MAXIMUM PARAMETERS EXCEEDED?
     D0 AA          0048                    WRK_B_MAXPARM(R10)      ;
```

```
         2A  14  004A        BGTR    100$                    ;BR IF SO
                 004C
      FFB1'  30  004C  40$:  BSBW    DCL$MARK                ;MARK CURRENT PARSE POSITION
    53   03  9A  004F        MOVZBL  #PTR_K_PARAMETR,R3      ;SET CLASSIFICATION OF VALUE STRING
      CF AA  96  0052        INCB    WRK_B_PARMSUM(R10)      ;INCREMENT COUNT OF PARAMETER VALUES
         55  D4  0055        CLRL    R5                      ;CLEAR PARENTHESIS FLAG
       0320  30  0057        BSBW    DCL$PARSE_VALUE         ;PROCESS PARAMETER VALUE
      23 50  E9  005A  50$:  BLBC    R0,190$                 ;IF LBC, THEN PARSE ERROR
                 005D
      FFA0'  30  005D        BSBW    DCL$SETCHAR             ;PEEK AT NEXT CHARACTER IN INPUT BUFFER
    50   2F  91  0060        CMPB    #^A'/',R0               ;QUALIFIER?
         05  12  0063        BNEQ    60$                     ;IF NEQ NO
       00EF  30  0065        BSBW    DCL$PROCQUAL            ;PROCESS QUALIFIER
         F0  11  0068        BRB     50$                     ;CHECK FOR ANOTHER QUALIFIER
                 006A
    50   2C  91  006A  60$:  CMPB    #^A',',R0               ;MULTIPLE PARAMETERS?
         14  13  006D        BEQL    110$                    ;IF EQL YES
    50   2B  91  006F        CMPB    #^A'+',R0               ;CONCATENATED PARAMETERS?
         20  13  0072        BEQL    120$                    ;IF EQL YES
         A4  11  0074        BRB     10$                     ;PROCESS NEXT PARAMETER
                 0076
                 0076  ;
                 0076  ; MAXIMUM PARAMETER COUNT EXCEEDED
                 0076  ;
      FF87'  30  0076  100$: BSBW    DCL$GETOKEN             ;GET NEXT TOKEN
                 0079        STATUS  MAXPARM                 ;SET MAXIMUM PARAMETER COUNT EXCEEDED
       00B1  31  0080  190$: BRW     EXIT                    ;EXIT WITH STATUS
                 0083
                 0083  ;
                 0083  ; PARAMETER LIST SPECIFIED
                 0083  ;
         05  E0  0083  110$: BBS     #ENT_V_LIST,-           ;IF SET, LISTS ALLOWED
   C4 04 A8      0085                ENT_Q_FLAGS(R8),40$     ;
      FF75'  30  0088        BSBW    DCL$MOVTOKN             ;GET "," AND TOKEN
                 008B        STATUS  NOLIST                  ;SET NO LISTS ALLOWED STATUS
         EC  11  0092        BRB     190$                    ;EXIT WITH STATUS
                 0094
                 0094  ;
                 0094  ; PARAMETER CONCATENATION SPECIFIED
                 0094  ;
         06  E0  0094  120$: BBS     #ENT_V_CONCAT,-         ;IF SET, CONCATENATION ALLOWED
   B3 04 A8      0096                ENT_Q_FLAGS(R8),40$     ;
      FF64'  30  0099        BSBW    DCL$MOVTOKN             ;GET "+" AND TOKEN
                 009C        STATUS  NOCCAT                  ;SET NO CONCATENATION ALLOWED STATUS
         DB  11  00A3        BRB     190$                    ;EXIT WITH STATUS
                 00A5
                 00A5  ;
                 00A5  ; INVALID PARAMETER DELIMITER
                 00A5  ;
      FF58'  30  00A5  130$: BSBW    DCL$MOVTOKN             ;GET DELIMITER AND TOKEN
                 00A8        STATUS  PARMDEL                 ;SET INVALID PARAMETER DELIMITER STATUS
         CF  11  00AF        BRB     190$                    ;EXIT WITH STATUS
                 00B1
                 00B1  ;
                 00B1  ; ALL COMMAND INPUT PROCESSED - CHECK FOR SUFFICIENT PARAMETERS
                 00B1  ;
                 00B1  END_OF_LINE:
```

```
                 00B1               STATUS  MAXPARM                 ;ASSUME MAXIMUM PARAMETER COUNT EXCEEDED
      CE AA   91 00B8               CMPB    WRK_B_PARMCNT(R10),-    ;MAXIMUM PARAMETERS EXCEEDED?
      D0 AA      00BB                       WRK_B_MAXPARM(R10)      ;
         2D   14 00BD               BGTR    210$                    ;BR IF SO
         02   E2 00BF               BBSS    #WRK_V_PROMPT,-         ;IF SET ALREADY PROMPTING
   2B F0 AA      00C1                       WRK_Q_FLAGS(R10),PROMPT ;
      CE AA   91 00C4               CMPB    WRK_B_PARMCNT(R10),-    ;SUFFICIENT PARAMETERS?
      D1 AA      00C7                       WRK_B_MINPARM(R10)      ;
         6C   18 00C9               BGEQ    NORMAL_EXIT             ;IF GEQ YES
                 00CB
                 00CB     ;
                 00CB     ; DO NOT PROMPT IF WE ARE PARSING A DCL COMMAND WITHIN A COMMAND PROCEDURE OR
                 00CB     ; A BATCH JOB.  IF WE ARE PARSING A USER COMMAND, THEN DO NOT PROMPT IF THE
                 00CB     ; USER HAS NOT SUPPLIED A PROMPT ROUTINE.
                 00CB     ;
    F9A6 CA   D5 00CB               TSTL    WRK_L_PROMPTRTN(R10)    ;DO WE HAVE A PROMPT ROUTINE?
         14   13 00CF               BEQL    200$                    ;SIGNAL ERROR IF NOT
         0D   E0 00D1               BBS     #WRK_V_USRMODE,-        ;IF SET THEN USER PROMPT ROUTINE
   19 F0 AA      00D3                       WRK_Q_FLAGS(R10),PROMPT ;
         0B   E0 00D6               BBS     #PRC_V_YLEVEL,-         ;IF SET, AT CONTROL Y/C LEVEL
   14 68 AB      00D8                       PRC_Q_FLAGS(R11),PROMPT ;
      5C AB   D5 00DB               TSTL    PRC_L_INDEPTH(R11)      ;INDIRECT LEVEL ZERO?
         05   12 00DE               BNEQ    200$                    ;IF NEQ NO
         06   E1 00E0               BBC     #PRC_V_MODE,-           ;IF CLR, INTERACTIVE MODE
   0A 68 AB      00E2                       PRC_Q_FLAGS(R11),PROMPT ;
                 00E5     200$:     STATUS  INSFPRM                 ;SET INSUFFICIENT PARAMETERS
       0045   31 00EC     210$:     BRW     EXIT                    ;EXIT WITH ERROR STATUS
                 00EF
                 00EF     ;
                 00EF     ; PROMPT FOR MISSING PARAMETERS
                 00EF     ;
      CE AA   91 00EF     PROMPT:   CMPB    WRK_B_PARMCNT(R10),-    ;ALL PARAMETERS PROCESSED?
      D0 AA      00F2                       WRK_B_MAXPARM(R10)      ;
         41   18 00F4               BGEQ    NORMAL_EXIT             ;IF GEQ YES
       06D1   30 00F6               BSBW    ISSUE_PROMPT            ;PROMPT FOR INPUT
      05 51   E8 00F9               BLBS    R1,220$                 ;CONTINUE IF NO ERROR
   50    51   D0 00FC               MOVL    R1,R0                   ;MOVE STATUS BACK TO R0
         33   11 00FF               BRB     EXIT                    ;EXIT WITH ERROR
         50   95 0101     220$:     TSTB    R0                      ;CHECK FIRST CHARACTER
         07   12 0103               BNEQ    222$                    ;IF EQL NULL LINE
         04   E0 0105               BBS     #ENT_V_VALREQ,-         ;IF SET, PARAMETER REQUIRED
   E5 04 A8      0107                       ENT_Q_FLAGS(R8),PROMPT  ;
         2B   11 010A               BRB     NORMAL_EXIT             ;NOT REQUIRED, THEN EXIT
   1A    50   91 010C     222$:     CMPB    R0,#CTRLZ               ;CONTROL/Z?
         1C   13 010F               BEQL    NO_COMMAND              ;CONTROL/Z TYPED - EXIT
   2F    50   91 0111               CMPB    R0,#^A'/'               ;LEADING QUALIFIER?
         0B   12 0114               BNEQ    230$                    ;BRANCH IF NOT
      CE AA   95 0116               TSTB    WRK_B_PARMCNT(R10)      ;FIRST PARAMETER?
         03   12 0119               BNEQ    225$                    ;NO, THEN PARSE PARAMETER QUALIFIERS
       FEE2   31 011B               BRW     DCL$PARSE_COMMAND       ;YES, THEN PARSE COMMAND QUALIFIERS
       FEE5   31 011E     225$:     BRW     PARSE_QUAL              ;PROCESS PARAMETER QUALIFIER(S)
         20   90 0121     230$:     MOVB    #^A' ',-                ;INSERT BLANK AT FRONT OF BUFFER
    F48E DA      0123                       @WRK_L_CHARPTR(R10)     ;
    F48E CA   D7 0126               DECL    WRK_L_CHARPTR(R10)      ;BACK UP CHARACTER POINTER
       FEED   31 012A               BRW     PARSE_PARMS             ;PROCESS PARAMETER
                 012D
                 012D     NO_COMMAND:
```

```
             012D   311          STATUS  NOCOMD               ;SET NO COMMAND FOUND
     07  10  0134   312 EXIT:    BSBB    DCL$GENEOL           ;GENERATE AN EOL TOKEN
         05  0136   313          RSB                          ;RETURN
             0137   314
             0137   315 NORMAL_EXIT:
     04  10  0137   316          BSBB    DCL$GENEOL           ;GENERATE AN EOL TOKEN
  FEC4'  30  0139   317          BSBW    DCL$EVAL_DISALLOW    ;EVALUATE ANY DISALLOW EXPRESSIONS
         05  013C   318          RSB                          ;RETURN
```

```
                          013D          .SBTTL  GENERATE AN EOL DESCRIPTOR
                          013D  ;+
                          013D  ; DCL$GENEOL - GENERATE AN EOL DESCRIPTOR
                          013D  ;
                          013D  ; THIS ROUTINE IS CALLED TO GENERATE AN END-OF-LINE TOKEN DESCRIPTOR.
                          013D  ;
                          013D  ; INPUTS:
                          013D  ;
                          013D  ;       R10 = BASE ADDRESS OF COMMAND WORK AREA.
                          013D  ;
                          013D  ; OUTPUTS:
                          013D  ;
                          013D  ;       R0 = RETURN STATUS
                          013D  ;
                          013D  ;-
                          013D
                          013D  DCL$GENEOL::
                50   DD   013D          PUSHL   R0                              ;SAVE RETURN STATUS
                54   D4   013F          CLRL    R4                              ;CLEAR ITEM NUMBER
          55    04   9A   0141          MOVZBL  #PTR_K_ENDLINE,R5               ;SET ITEM TYPE TO END OF LINE
                56   D4   0144          CLRL    R6                              ;CLEAR FLAGS
          57    01   9A   0146          MOVZBL  #1,R7                           ;SET LENGTH OF ITEM
   58   F486    CA   D0   0149          MOVL    WRK_L_EXPANDPTR(R10),R8         ;SET STARTING ADDRESS OF ITEM
                59   D4   014E          CLRL    R9                              ;CLEAR ENTITY BLOCK ADDRESS
             FEAD'   30   0150          BSBW    DCL$GENDESCR                    ;GENERATE END-OF-LINE TOKEN
              50   8ED0   0153          POPL    R0                              ;RESTORE RETURN STATUS
                     05   0156          RSB                                     ;RETURN FROM SUBROUTINE
                          0157
```

```
                 0157        .SBTTL  PROCESS QUALIFIER SPECIFICATION
                 0157  ;+
                 0157  ; DCL$PROCQUAL - PROCESS QUALIFIER SPECIFICATION
                 0157  ;
                 0157  ; THIS ROUTINE IS CALLED TO PARSE A QUALIFIER SPECIFICATION AND EMIT A
                 0157  ; DESCRIPTOR TO THE RESULT PARSE TABLE.
                 0157  ;
                 0157  ; INPUTS:
                 0157  ;
                 0157  ;       R10 = BASE ADDRESS OF COMMAND WORK AREA.
                 0157  ;
                 0157  ; OUTPUTS:
                 0157  ;
                 0157  ;       R0 LOW BIT CLEAR INDICATES FAILURE TO PARSE QUALIFIER SPECIFICATION.
                 0157  ;
                 0157  ;               R0 = DCL$_ABKEYW - AMBIGUOUS KEYWORD IN QUALIFIER.
                 0157  ;               R0 = DCL$_IMCHNG - MULTIPLE ATTEMPT TO CHANGE IMAGE NAME.
                 0157  ;               R0 = DCL$_IVKEYW - INVALID KEYWORD IN QUALIFIER.
                 0157  ;               R0 = DCL$_IVQLOC - INVALID QUALIFIER LOCATION.
                 0157  ;               R0 = DCL$_IVVALU - INVALID QUALIFIER VALUE SYNTAX.
                 0157  ;               R0 = DCL$_NOKEYW - NO KEYWORD IN QUALIFIER.
                 0157  ;               R0 = DCL$_NOQUAL - NO QUALIFIERS ALLOWED ON COMMAND.
                 0157  ;               R0 = DCL$_NOVALU - NO VALUE ALLOWED ON QUALIFIER.
                 0157  ;
                 0157  ;       R0 LOW BIT SET INDICATES SUCCESSFUL PARSE WITH THE QUALIFIER
                 0157  ;               DESCRIPTOR EMITTED AND THE QUALIFIER SPECIFICATION COPIED
                 0157  ;               TO THE COMMAND BUFFER.
                 0157  ;
                 0157  ;               R0 = DCL$_NORMAL - NORMAL COMPLETION.
                 0157  ;-
                 0157
                 0157  DCL$PROCQUAL::                                  ;PROCESS QUALIFIER SPECIFICATION
   01C0 8F   BB  0157          PUSHR   #^M<R6,R7,R8>                   ;SAVE REGISTERS
        56   D4  015B          CLRL    R6                              ;CLEAR KEYWORD NEGATION FLAG
     C4 AA   96  015D          INCB    WRK_B_VALLEV(R10)               ;INCREASE THE VALUE LEVEL
                 0160
                 0160  ;
                 0160  ; GET QUALIFIER KEYWORD
                 0160  ;
     FE9D'   30  0160          BSBW    DCL$MOVCHAR                     ;MOVE SLASH TERMINATOR
     FE9A'   30  0163          BSBW    DCL$MARK                        ;MARK CURRENT POSITION IN BUFFER
     FE97'   30  0166          BSBW    DCL$GETOKEN                     ;GET QUALIFIER NAME
        16   13  0169          BEQL    90$                             ;IF NONE, THEN ERROR
                 016B
                 016B  ;
                 016B  ; CHECK THAT IT MATCHES AN ALLOWED QUALIFIER KEYWORD
                 016B  ;
                 016B          STATUS  NOQUAL                          ;ASSUME NO QUALIFIERS ALLOWED ON COMMAND
 58  CA AA   D0  0172          MOVL    WRK_L_QUABLK(R10),R8            ;GET ADDRESS OF QUALIFIER DESCRIPTORS
        10   13  0176          BEQL    100$                            ;IF EQL NONE
      0089   30  0178          BSBW    DCL$FIND_KEYWORD                ;LOOKUP QUALIFIER
     0D 50   E8  017B          BLBS    R0,110$                         ;IF SUCCESSFUL, THEN CONTINUE
      007B   31  017E          BRW     390$                            ;OTHERWISE, EXIT WITH ERROR STATUS
                 0181  90$:    STATUS  NOKEYW                          ;SET NO KEYWORD STATUS
      0071   31  0188  100$:   BRW     390$                            ;EXIT WITH ERROR STATUS
                 018B
                 018B  ;
```

```
                    018B    ; VALID KEYWORD MATCH FOUND - CHECK THAT IT IS POSITIONED CORRECTLY
                    018B    ;
                    018B    110$:   STATUS  IVQLOC                  ;ASSUME INVALID QUALIFIER LOCATION
           06   E0  0192            BBS     #WRK_V_VERB,-           ;IF SET, PROCESSING VERB
     05 F0 AA       0194                    WRK_Q_FLAGS(R10),120$   ;
           09   E0  0197            BBS     #ENT_V_PARM,-           ;IF SET, ALLOWED ON PARAMETERS
     05 04 A8       0199                    ENT_Q_FLAGS(R8),125$    ;
           08   E1  019C    120$:   BBC     #ENT_V_VERB,-           ;IF CLR, NOT ALLOWED ON VERBS
     E7 04 A8       019E                    ENT_Q_FLAGS(R8),100$    ;
                    01A1
                    01A1    ;
                    01A1    ; CHECK NEGATION
                    01A1    ;
                    01A1            ASSUME  V_NEGAT EQ 0            ;
        0C 56   E9  01A1    125$:   BLBC    R6,150$                 ;BRANCH IF QUALIFIER WAS NOT NEGATED
                    01A4            STATUS  NOTNEG                  ;ASSUME QUALIFIER NOT NEGATABLE
           01   E1  01AB            BBC     #ENT_V_NEG,-            ;IF CLR, NOT NEGATABLE
     D8 04 A8       01AD                    ENT_Q_FLAGS(R8),100$    ;
                    01B0
                    01B0    ;
                    01B0    ; PROCESS CHANGE LIST - IF ANY
                    01B0    ;
        0C A8   D5  01B0    150$:   TSTL    ENT_L_SYNTAX(R8)        ;NEW COMMAND SYNTAX?
           0B   13  01B3            BEQL    160$                    ;BRANCH IF NOT
         06C0   30  01B5            BSBW    DCL$CHANGE_SYNTAX       ;PROCESS CHANGE LIST
                    01B8            ASSUME  PTR_V_SYNTAX EQ 22      ;
     56    04   C8  01B8            BISL    #M_SYNTAX,R6            ;SET SYNTAX CHANGE BIT
        02 50   E9  01BB            BLBC    R0,160$                 ;BRANCH IF NO QUALIFIER CHANGE LIST
           57   D4  01BE            CLRL    R7                      ;INVALIDATE THIS QUALIFIER
                    01C0
                    01C0    ;
                    01C0    ; GENERATE QUALIFIER DESCRIPTOR
                    01C0    ;
     55    00   9A  01C0    160$:   MOVZBL  #PTR_K_COMDQUAL,R5      ;ASSUME COMMAND QUALIFIER
           06   E0  01C3            BBS     #WRK_V_VERB,-           ;IF SET, PROCESSING VERB
     08 F0 AA       01C5                    WRK_Q_FLAGS(R10),170$   ;
           09   E1  01C8            BBC     #ENT_V_PARM,-           ;IF CLR, COMMAND QUALIFIER
     03 04 A8       01CA                    ENT_Q_FLAGS(R8),170$    ;
     55    01   9A  01CD            MOVZBL  #PTR_K_PARMQUAL,R5      ;SET TYPE TO PARAMETER QUALIFIER
     7E    57   7D  01D0    170$:   MOVQ    R7,-(SP)                ;SAVE REGISTERS
     54    57   D0  01D3            MOVL    R7,R4                   ;GET QUALIFIER NUMBER
     59    58   D0  01D6            MOVL    R8,R9                   ;GET ENTITY BLOCK ADDRESS
        FE24'   30  01D9            BSBW    DCL$MARKEDTOKEN         ;GET QUALIFIER DESCRIPTOR
     57    51   7D  01DC            MOVQ    R1,R7                   ;
        FE1E'   30  01DF            BSBW    DCL$GENDESCR            ;GENERATE RESULT PARSE DESCRIPTOR
     57    8E   7D  01E2            MOVQ    (SP)+,R7                ;RESTORE REGISTERS
                    01E5
                    01E5    ;
                    01E5    ; CHECK FOR QUALIFIER VALUE(S)
                    01E5    ;
                    01E5            STATUS  NOVALU                  ;ASSUME VALUE NOT ALLOWED
     53    02   9A  01EC            MOVZBL  #PTR_K_QUALVALU,R3      ;SET VALUE TYPE
         00C2   30  01EF            BSBW    DCL$CHECK_VALUES        ;CHECK FOR VALUES
        07 50   E9  01F2            BLBC    R0,390$                 ;EXIT IF ERROR
                    01F5
                    01F5    ;
                    01F5    ; CLEAN UP AND EXIT
```

```
               01F5   463 ;
               01F5   464        STATUS  NORMAL              ;SET SUCCESSFUL COMPLETION
   C4 AA   97  01FC   465 390$:  DECB    WRK_B_VALLEV(R10)   ;DECREASE THE VALUE LEVEL
 01C0 8F   BA  01FF   466        POPR    #^M<R6,R7,R8>       ;RESTORE REGISTERS
           05  0203   467        RSB                         ;RETURN FROM SUBROUTINE
```

```
                    0204          .SBTTL  FIND KEYWORD
                    0204     ;+
                    0204     ; DCL$FIND_KEYWORD - LOOK UP SPECIFIED KEYWORD IN SPECIFIED LIST
                    0204     ;
                    0204     ; THIS ROUTINE IS CALLED TO LOOK UP A SPECIFIED QUALIFIER OR KEYWORD
                    0204     ; IN THE APPROPRIATE LIST.  IF THE SEARCH IS SUCCESSFUL, THE QUALIFIER
                    0204     ; OR KEYWORD NUMBER IS RETURNED ALONG WITH THE ADDRESS OF THE ENTITY
                    0204     ; DESCRIPTOR BLOCK.  OTHERWISE, AN ERROR STATUS IS RETURNED.
                    0204     ;
                    0204     ; INPUTS:
                    0204     ;
                    0204     ;       R1/R2 = DESCRIPTOR OF KEYWORD TO LOOK UP
                    0204     ;       R6 = 0 IF QUALIFIER KEYWORD, M_KEYWD IF REGULAR KEYWORD
                    0204     ;       R8 = ADDRESS OF LIST OF DESCRIPTOR BLOCKS
                    0204     ;       R10 = ADDRESS OF COMMAND WORK AREA
                    0204     ;
                    0204     ; OUTPUTS:
                    0204     ;
                    0204     ;       R1/R2 = DESCRIPTOR OF KEYWORD (NEEDED BY DCL$PARSE_VALUE)
                    0204     ;       R6 = NEGATION FLAG - LBS, IF KEYWORD NEGATED; LBC, IF NOT
                    0204     ;       R7 = QUALIFIER OR KEYWORD NUMBER (IF SUCCESSFUL)
                    0204     ;       R8 = ADDRESS OF ENTITY DESCRIPTOR BLOCK
                    0204     ;
                    0204     ;-
                    0204
                    0204     DCL$FIND_KEYWORD:
                    0204          ASSUME  PTR_V_NEGATE EQ 20                ;
     023E 8F   BB   0204          PUSHR   #^M<R1,R2,R3,R4,R5,R9>            ;SAVE REGISTERS
                    0208
                    0208     ;
                    0208     ; IF KEYWORD LIST, THEN SKIP PAST TYPE BLOCK.
                    0208     ;
  03   02 A8   91   0208          CMPB    ENT_B_TYPE(R8),#BLOCK_K_TYPE      ;TYPE BLOCK?
          06   12   020C          BNEQ    5$                                ;NO, THEN SKIP
       DE AA   C1   020E          ADDL3   WRK_L_TAB_VEC(R10),-              ;GET ADDR OF FIRST ENT BLOCK
  58   08 A8        0211                  TYPE_C_KEYWORDS(R8),R8            ;
                    0214
                    0214     ;
                    0214     ; SAVE ADDRESS OF LIST OF DESCRIPTOR BLOCKS.
                    0214     ;
     59   58   D0   0214     5$:  MOVL    R8,R9                             ;SAVE LIST ADDRESS
                    0217
                    0217     ;
                    0217     ; INITIALIZE SEARCH PARAMETERS.
                    0217     ;
          57   D4   0217     10$: CLRL    R7                                ;CLEAR QUALIFIER NUMBER
          7E   7C   0219          CLRQ    -(SP)                             ;SET NULL AS QUALIFIER FOUND
                    021B
                    021B     ;
                    021B     ; IF QUALIFIER THEN ONLY CHECK FIRST FOUR CHARACTERS OF KEYWORD,
                    021B     ; ELSE CHECK THE WHOLE THING.
                    021B     ;
     5C   51   D0   021B          MOVL    R1,AP                             ;SAVE ORIGINAL TOKEN SIZE
  08 56   01   E0   021E          BBS     #V_KEYWD,R6,30$                   ;ARE WE LOOKING FOR A KEYWORD?
     51   04   D1   0222          CMPL    #4,R1                             ;KEYWORD LESS THAN 5 CHARACTERS?
          03   18   0225          BGEQ    30$                               ;IF GEQ YES
     51   04   D0   0227          MOVL    #4,R1                             ;SET TO ONLY COMPARE 4 CHARACTERS
```

```
                          022A
                          022A     ;
                          022A     ; GET NEXT QUALIFIER IN LIST
                          022A     ;
  50    16 A8      32     022A     30$:    CVTWL   ENT_W_NAME(R8),R0          ;GET OFFSET TO ASCIC QUALIFIER NAME
     55   6840     9E     022E             MOVAB   (R8)[R0],R5                ;FIND ADDRESS OF QUALIFIER KEY LENGTH
 50    01 A840     9E     0232             MOVAB   1(R8)[R0],R0               ;FIND ADDRESS OF QUALIFIER KEY TEXT
        53  51     7D     0237             MOVQ    R1,R3                      ;COPY QUALIFIER STRING DESCRIPTOR
                          023A
                          023A     ;
                          023A     ; COMPARE THE QUALIFIER WITH THE INPUT
                          023A     ;
        60  20     91     023A     60$:    CMPB    #^A/ /,(R0)                ;END OF QUALIFIER TEXT?
            12     13     023D             BEQL    65$                        ;BR IF YES
        84  80     91     023F             CMPB    (R0)+,(R4)+                ;IS THIS THE QUALIFIER WERE LOOKIN FOR?
            18     12     0242             BNEQ    70$                        ;BR IF DEFINITELY NOT!
         F3 53     F5     0244             SOBGTR  R3,60$                     ;BR IF MORE TO CHECK
                          0247
                          0247     ;
                          0247     ; WHAT KIND OF MATCH DO WE HAVE?
                          0247     ;
        65  5C     91     0247             CMPB    AP,(R5)                    ;IS TABLE QUALIFIER SAME LENGTH
            05     12     024A             BNEQ    65$                        ;NO, THEN SKIP
        5E  08     C0     024C             ADDL    #8,SP                      ;YES, THEN WE HAVE A UNIQUE MATCH
            50     11     024F             BRB     83$                        ;RESTORE STACK, BRANCH TO EXIT
                          0251
                          0251     ;
                          0251     ; CHECK FOR AMBIGUITY AND SAVE MATCH.
                          0251     ;
         04 AE     D5     0251     65$:    TSTL    4(SP)                      ;FIND MATCH BEFORE?
            03     13     0254             BEQL    67$                        ;BR IF NOT AMBIGUOUS
        56  08     C8     0256             BISL    #M_AMBIG,R6                ;SET AMBIGUOUS BIT
        6E  57     7D     0259     67$:    MOVQ    R7,(SP)                    ;SAVE MATCHED QUALIFIER VALUES
                          025C
                          025C     ;
                          025C     ; CHECK NEXT QUALIFIER IN LIST.
                          025C     ;
            57     D6     025C     70$:    INCL    R7                         ;INCREMENT QUALIFIER NUMBER
  58     08 A8     D0     025E             MOVL    ENT_L_NEXT(R8),R8          ;GET OFFSET TO NEXT QUALIFIER DESCRIPTOR
            06     13     0262             BEQL    75$                        ;IF EQL THEN DONE
  58     DE AA     C0     0264             ADDL    WRK_L_TAB_VEC(R10),R8      ;CALCULATE ADDRESS
            C0     11     0268             BRB     30$                        ;CHECK NEXT QUALIFIER
                          026A
                          026A     ;
                          026A     ; ALL QUALIFERS HAVE BEEN CHECK WITHOUT AMBIGUITY, NOW SEE IF ANY MATCHED.
                          026A     ;
                          026A     75$:    STATUS  ABKEYW                     ;ASSUME AMBIGUOUS
   35 56    03     E4     0271             BBSC    #V_AMBIG,R6,85$            ;BR IF TRUE
      57    8E     7D     0275             MOVQ    (SP)+,R7                   ;RESTORE MATCHED QUALIFER PARAMETERS
            27     12     0278             BNEQ    83$                        ;BR IF ONE WAS FOUND
                          027A
                          027A     ;
                          027A     ; QUALIFIER DESCRIPTOR TABLE EXHAUSTED - TRY NEGATION
                          027A     ;
        56  01     CC     027A             XORL    #M_NEGAT,R6                ;COMPLEMENT NEGATION FLAG
  51    5C  02     C3     027D             SUBL3   #2,AP,R1                   ;REDUCE CHARACTER COUNT
            0A     15     0281             BLEQ    80$                        ;IF LEQ NO MATCH POSSIBLE
```

```
           58   59   D0  0283        MOVL    R9,R8                  ;RESTORE ADDRESS OF DESCRIPTOR LIST
 82      4F4E   8F   B1  0286        CMPW    #^A/NO/,(R2)+          ;KEYWORD START WITH 'NO'?
                8A   13  028B        BEQL    10$                    ;IF EQL YES
                         028D
                         028D  ;
                         028D  ; SET STATUS, CLEAN UP, AND RETURN
                         028D  ;
                         028D  80$:  STATUS  IVQUAL                 ;ASSUME INVALID QUALIFIER
      15 56     01   E1  0294        BBC     #V_KEYWD,R6,90$        ;BRANCH IF QUALIFIER PROCESSING
                         0298        STATUS  IVKEYW                 ;SET INVALID KEYWORD STATUS
                0C   11  029F        BRB     90$                    ;EXIT WITH ERROR STATUS
                         02A1
                         02A1  83$:  STATUS  NORMAL                 ;SET SUCCESS STATUS
                03   11  02A8        BRB     90$                    ;EXIT WITH ERROR STATUS
                         02AA
           57   8E   7D  02AA  85$:  MOVQ    (SP)+,R7               ;RESTORE MATCHED QUALIFIER PARAMETERS
                57   D6  02AD  90$:  INCL    R7                     ;ADJUST TO ACTUAL KEYWORD NUMBER
         023E   8F   BA  02AF        POPR    #^M<R1,R2,R3,R4,R5,R9> ;RESTORE REGISTERS
                     05  02B3        RSB                            ;RETURN FROM SUBROUTINE
```

```
                    02B4        .SBTTL  CHECK FOR VALUES
                    02B4   ;+
                    02B4   ; DCL$CHECK_VALUES - CHECK FOR VALUES AND THEN PROCESS THEM
                    02B4   ;
                    02B4   ; THIS ROUTINE IS CALLED TO DETERMINE IF A VALUE OR LIST OF VALUES IS
                    02B4   ; ASSOCIATED WITH THE LAST QUALIFIER OR PARAMETER PARSED, TO DETERMINE
                    02B4   ; THE SYNTACTIC LEGALITY OF THE EXISTENCE OF VALUES, AND TO CALL
                    02B4   ; DCL$PARSE_VALUE TO PARSE EACH VALUE FOUND.
                    02B4   ;
                    02B4   ; INPUTS:
                    02B4   ;
                    02B4   ;       R3 = CLASSIFICATION OF VALUE (PTR_K_QUALVALU OR PTR_K_PARAMETR)
                    02B4   ;       R6 = NEGATION FLAG - LBS, IF KEYWORD NEGATED; LBC, IF NOT
                    02B4   ;       R8 = ADDRESS OF ENTITY DESCRIPTOR BLOCK
                    02B4   ;       R10 = ADDRESS OF COMMAND WORK AREA
                    02B4   ;
                    02B4   ; IMPLICIT INPUTS:
                    02B4   ;
                    02B4   ;       WRK_L_EXPANDPTR IS POSITIONED AT THE TERMINATOR OF THE PREVIOUS
                    02B4   ;       QUALIFIER OR PARAMETER.
                    02B4   ;
                    02B4   ; OUTPUTS:
                    02B4   ;
                    02B4   ;       R0 INDICATES THE SYNTACTIC CORRECTNESS OR INCORRECTNESS OF
                    02B4   ;       WHAT WAS FOUND.
                    02B4   ;
                    02B4   ;-
                    02B4
                    02B4   DCL$CHECK_VALUES:
          53   DD   02B4        PUSHL   R3                          ;SAVE VALUE TYPE
                    02B6
                    02B6   ;
                    02B6   ; CHECK IF VALUE HAS BEEN SPECIFIED
                    02B6   ;
          7E   D4   02B6        CLRL    -(SP)                       ;ASSUME NO BLANKS
                    02B8        SETBIT  PRC_V_IND,PRC_W_FLAGS(R11)  ;DISABLE @ INDIRECTION
       FD41'   30   02BC        BSBW    DCL$SETCHAR                 ;PEEK AT NEXT CHAR IN INPUT BUFFER
    20    50   91   02BF        CMPB    R0,#^A' '                   ;BLANK?
          05   12   02C2        BNEQ    5$                          ;IF NEQ NO
          6E   D6   02C4        INCL    (SP)                        ;MARK BLANK SEEN
       FD37'   30   02C6        BSBW    DCL$SETNBLK                 ;PEEK AT NEXT NON-BLANK
                    02C9   5$:  CLRBIT  PRC_V_IND,PRC_W_FLAGS(R11)  ;ENABLE @ INDIRECTION
    3D    50   91   02CD        CMPB    R0,#^A'='                   ;VALUE SPECIFIED?
          21   13   02D0        BEQL    10$                         ;IF EQL YES
    3A    50   91   02D2        CMPB    R0,#^A':'                   ;VALUE SPECIFIED?
          1C   13   02D5        BEQL    10$                         ;IF EQL YES
          8E   D5   02D7        TSTL    (SP)+                       ;WAS A BLANK SEEN?
          06   13   02D9        BEQL    7$                          ;BRANCH IF NO BLANK SEEN
    50    20   90   02DB        MOVB    #^A' ',R0                   ;RESTORE THE BLANK
       FD1F'   30   02DE        BSBW    DCL$BACKUPCHAR              ;
                    02E1
                    02E1   ;
                    02E1   ; IS VALUE REQUIRED?
                    02E1   ;
                    02E1        ASSUME  V_NEGAT EQ 0                ;
       0C 56   E8   02E1   7$:  BLBS    R6,9$                       ;NO VALUE IS OK IF ENTITY WAS NEGATED
                    02E4        STATUS  VALREQ                      ;ASSUME VALUE REQUIRED ERROR
```

```
         04  E0  02EB          BBS     #ENT_V_VALREQ,-         ;IF SET, REPORT VALUE IS REQUIRED
   7F 04 A8      02ED                  ENT_Q_FLAGS(R8),95$     ;
       0075  31  02F0  9$:     BRW     90$                     ;OTHERWISE, ITS OK
                 02F3
                 02F3  ;
                 02F3  ; IS VALUE ALLOWED?
                 02F3  ;
         8E  D5  02F3  10$:    TSTL    (SP)+                   ;RESTORE THE STACK
      FD08'  30  02F5          BSBW    DCL$MOVCHAR             ;MOVE EQUAL SIGN
      FD05'  30  02F8          BSBW    DCL$GENTERM             ;SET TOKEN TERMINATOR
                 02FB          STATUS  NOVALU                  ;ASSUME NO VALUE ALLOWED
         00  E1  0302          BBC     #ENT_V_VAL,-            ;IF CLR, VALUE NOT ALLOWED
   68 04 A8      0304                  ENT_Q_FLAGS(R8),95$     ;
         02  91  0307          CMPB    #ENT_K_QUALIFIER,-      ;ARE WE PARSING A QUALIFIER VALUE?
      02 A8      0309                  ENT_B_TYPE(R8)          ;
         03  12  030B          BNEQ    15$                     ;NO, THEN SKIP
                 030D          ASSUME  V_NEGAT EQ 0            ;
      5F 56  E8  030D          BLBS    R6,95$                  ;YES, THEN ERROR IF QUALIFIER WAS NEGATED
                 0310
                 0310  ;
                 0310  ; CHECK FOR VALUE LIST AND PROCESS VALUE(S)
                 0310  ;
   55    02  CE  0310  15$:    MNEGL   #2,R5                   ;SET VALUE STATE VARIABLE
      FCEA'  30  0313          BSBW    DCL$SETNBLK             ;PEEK AT FIRST NON-BLANK AFTER (=)
   28    50  91  0316          CMPB    R0,#^A'('               ;START OF VALUE LIST?
         05  13  0319          BEQL    20$                     ;BR IF YES
      FCE2'  30  031B          BSBW    DCL$BACKUPMOVE          ;RESTORE EQUAL SIGN DELIMITER
         02  11  031E          BRB     30$                     ;AND PARSE THE SINGLE VALUE
         55  D6  0320  20$:    INCL    R5                      ;MARK LIST IS PRESENT
   53    6E  D0  0322  30$:    MOVL    (SP),R3                 ;SET TOKEN TYPE
       0052  30  0325          BSBW    DCL$PARSE_VALUE         ;PARSE VALUE STRING
      44 50  E9  0328          BLBC    R0,95$                  ;BRANCH IF ERROR DETECTED
         55  D6  032B          INCL    R5                      ;COUNT UP THE VALUES SEEN
         39  19  032D          BLSS    90$                     ;BR IF SINGLE VALUE QUALIFIER
   03    50  D1  032F          CMPL    R0,#3                   ;SHOULD PAREN CHECK BE IGNORED?
         34  13  0332          BEQL    90$                     ;YES, THEN BRANCH
      FCC9'  30  0334          BSBW    DCL$SETCHAR             ;PEEK AT NEXT CHARACTER
   2C    50  91  0337          CMPB    R0,#^A','               ;TERMINATER SAY MORE VALUES COMING?
         E6  13  033A          BEQL    30$                     ;IF MORE COMING, PROCESS THEM
   29    50  91  033C          CMPB    R0,#^A')'               ;END OF VALUE LIST?
         1B  12  033F          BNEQ    80$                     ;IF NO-ITS AN ERROR
      FCBC'  30  0341          BSBW    DCL$MOVCHAR             ;COPY RIGHT PARENTHESIS DELIMITER
      FCB9'  30  0344          BSBW    DCL$TESTBLANK           ;THROW AWAY INSIGNIFICANT TRAILING BLANKSK
      FCB6'  30  0347          BSBW    DCL$GENTERM             ;SET ACTUAL TERMINATOR INSTEAD OF RPAREN
                 034A                                          ;SINCE END OF LIST CAN BE DEDUCED BY
                 034A                                          ;NON-QUALVALU TOKEN; AND SINCE BLANK
                 034A                                          ;TERMINATOR MARKS A PARAMETER NEXT.
                 034A
                 034A  ;
                 034A  ; DID WE GET MORE THAN ONE VALUE?  ARE VALUE LISTS ALLOWED?
                 034A  ;
         55  D5  034A          TSTL    R5                      ;DID WE GET MORE THAN ONE VALUE?
         1A  13  034C          BEQL    90$                     ;BR IF NO (PARENS WERE NOP)
                 034E          STATUS  ONEVAL                  ;ASSUME ONLY ONE VALUE ALLOWED
         05  E1  0355          BBC     #ENT_V_LIST,-           ;ERROR IF NO LISTS ALLOWED
   15 04 A8      0357                  ENT_Q_FLAGS(R8),95$     ;
         0C  11  035A          BRB     90$                     ;CONTINUE
```

```
                  035C   717
                  035C   718 ;
                  035C   719 ; NO CLOSING PAREN SYNTAX ERROR
                  035C   720 ;
    FCA1'    30   035C   721 80$:     BSBW    DCL$MOVTOKN          ;GET DELIMITER AND FOLLOWING TOKEN
                  035F   722          STATUS  NOPAREN              ;SET INVALID SYNTAX
       07    11   0366   723          BRB     95$                  ;EXIT WITH STATUS
                  0368   724
                  0368   725 ;
                  0368   726 ; SINGLE VALUE QUALIFIER
                  0368   727 ;
                  0368   728 90$:     STATUS  NORMAL               ;SET SUCCESSFUL COMPLETION
     53 8ED0      036F   729 95$:     POPL    R3                   ;RESTORE TOKEN TYPE
             05   0372   730          RSB                          ;RETURN FROM SUBROUTINE
```

```
                        0373          .SBTTL  PARSE VALUE STRING
                        0373    ;+
                        0373    ; DCL$PARSE_VALUE - PARSE A SINGLE VALUE IN A VALUE LIST
                        0373    ;
                        0373    ; THIS ROUTINE IS CALLED TO SCAN A SINGLE VALUE AND STORE
                        0373    ; THE TOKEN DESCRIPTOR WHICH DESCRIBES IT.
                        0373    ;
                        0373    ; INPUTS:
                        0373    ;
                        0373    ;       R3 = CLASSIFICATION OF VALUE (PTR_K_QUALVALU OR PTR_K_PARAMETR)
                        0373    ;       R5 = -1 IF FIRST VALUE IN A PARENTHESIZED LIST
                        0373    ;       R8 = ADDRESS OF ENTITY DESCRIPTOR BLOCK
                        0373    ;       R10 = ADDRESS OF COMMAND WORK AREA
                        0373    ;
                        0373    ;       CHARACTER POINTER POINTS TO DELIMITER JUST BEFORE THE VALUE STRING
                        0373    ;
                        0373    ; OUTPUTS:
                        0373    ;
                        0373    ;       R0 = STATUS
                        0373    ;
                        0373    ;       INSIGNIFICANT BLANKS ARE THROWN AWAY FOLLOWING THE VALUE.
                        0373    ;       CHARACTER POINTER POINTS TO DELIMITER JUST AFTER THE VALUE STRING.
                        0373    ;
                        0373    ;       ONE (OR TWO) TOKEN DESCRIPTORS ARE OUTPUT DEPENDING ON WHETHER
                        0373    ;       THE SYNTAX WAS 'VALUE' OR 'KEYWORD=VALUE'.
                        0373    ;-
                        0373
                        0373    ;
                        0373    ; VALID TERMINATOR LIST FOR VALUE STRINGS
                        0373    ;
  00 29 28 2F 2C 2B 20  0373    TRMB:   .ASCII  \ +,/()\<0>             ; SPACE,PLUS,COMMA,SLASH,PARENS AND EOL
                        037A    TRME:
                        037A
                        037A    DCL$PARSE_VALUE:
         01F8 8F    BB  037A            PUSHR   #^M<R3,R4,R5,R6,R7,R8>  ;SAVE REGISTERS
           C4 AA    96  037E            INCB    WRK_B_VALLEV(R10)       ;INCREASE THE VALUE LEVEL
      50   15 A8    9A  0381            MOVZBL  ENT_B_VALTYPE(R8),R0    ;GET VALUE TYPE
         14    50   91  0385            CMPB    R0,#ENT_K_FILE          ;FILESPEC?
               34   13  0388            BEQL    5$                      ;IF SO, PROCESS FILESPEC
         01    50   91  038A            CMPB    R0,#ENT_K_INFILE        ;INPUT FILESPEC?
               2F   13  038D            BEQL    5$                      ;IF SO, PROCESS FILESPEC
         02    50   91  038F            CMPB    R0,#ENT_K_OUTFILE       ;OUTPUT FILESPEC?
               2A   13  0392            BEQL    5$                      ;IF SO, PROCESS FILESPEC
         0C    50   91  0394            CMPB    R0,#ENT_K_NODE          ;NODE NAME?
               25   13  0397            BEQL    5$                      ;IF SO, PROCESS AS FILESPEC
         0D    50   91  0399            CMPB    R0,#ENT_K_DEVICE        ;DEVICE NAME?
               20   13  039C            BEQL    5$                      ;IF SO, PROCESS AS FILESPEC
         0E    50   91  039E            CMPB    R0,#ENT_K_DIR           ;DIRECTORY SPEC?
               1B   13  03A1            BEQL    5$                      ;IF SO, PROCESS AS FILESPEC
         0F    50   91  03A3            CMPB    R0,#ENT_K_UIC           ;UIC SPEC?
               16   13  03A6            BEQL    5$                      ;IF SO, PROCESS AS FILESPEC
         10    50   91  03A8            CMPB    R0,#ENT_K_RESTOFLINE    ;REST OF LINE AS VALUE STRING?
               17   13  03AB            BEQL    10$                     ;IF SO, PROCESS IT
         15    50   91  03AD            CMPB    R0,#ENT_K_EXPRESSION    ;$EXPRESSION?
               18   13  03B0            BEQL    12$                     ;IF SO, PROCESS IT
         19    50   91  03B2            CMPB    R0,#ENT_K_ACL           ;$ACL?
               19   13  03B5            BEQL    13$                     ;IF SO, PROCESS IT
```

```
           11   50   91  03B7          CMPB    R0,#ENT_K_PARENVALUE    ;$PAREN_VALUE?
                1A   13  03BA          BEQL    15$                     ;IF SO, PROCESS IT
                1E   11  03BC          BRB     17$                     ;IF NOT, BRANCH
                         03BE
              00F5   30  03BE  5$:     BSBW    DCL$PROCFILE            ;PARSE THE FILE SPEC
              00A4   31  03C1          BRW     90$                     ;SKIP TO END
              01CC   30  03C4  10$:    BSBW    DCL$REST_OF_LINE        ;PARSE $REST_OF_LINE VALUE
              009E   31  03C7          BRW     90$                     ;EMIT A DESCRIPTOR
              03BC   30  03CA  12$:    BSBW    DCL$EXPRESSION          ;PARSE $EXPRESSION VALUE
              007C   31  03CD          BRW     60$                     ;EMIT A DESCRIPTOR
              0343   30  03D0  13$:    BSBW    DCL$ACL                 ;PARSE $ACL VALUE
              0092   31  03D3          BRW     90$                     ;SKIP TO END
              02CA   30  03D6  15$:    BSBW    DCL$PAREN_VALUE         ;PARSE VALUE
              008C   31  03D9          BRW     90$                     ;SKIP TO END
                         03DC
                         03DC  ;
                         03DC  ; GET UNTYPED VALUE TOKEN (STOP AT "=" OR ":" IF KEYWORDS ARE ALLOWED)
                         03DC  ;
             FC21'   30  03DC  17$:    BSBW    DCL$MOVCHAR             ;COPY TERMINATOR INTO EXPANSION BUFFER
             FC1E'   30  03DF          BSBW    DCL$SETNBLK             ;SKIP LEADING BLANKS
             FC1B'   30  03E2          BSBW    DCL$MARK                ;MARK START OF VALUE STRING
             FC18'   30  03E5  20$:    BSBW    DCL$GTBTOKEN            ;COPY TOKEN INTO EXPANSION BUFFER
             10 A8   D5  03E8          TSTL    ENT_L_USER_TYPE(R8)     ;DOES VALUE TAKE KEYWORDS?
                0A   13  03EB          BEQL    30$                     ;IF NOT, IGNORE EMBEDDED '=:'
           3D   50   91  03ED          CMPB    R0,#^A'='               ;VALUE SEPARATOR?
                12   13  03F0          BEQL    50$                     ;BR IF YES
           3A   50   91  03F2          CMPB    R0,#^A':'               ;CHECK IF END OF KEY AND START OF VALUE
                0D   13  03F5          BEQL    50$                     ;BR IF YES
 FF76 CF   07   50   3A  03F7  30$:    LOCC    R0,S^#TRME-TRMB,TRMB    ;CHECK FOR VALUE STRING TERMINATOR
                05   12  03FD          BNEQ    50$                     ;BR IF TERMINATOR FOUND
             FBFE'   30  03FF          BSBW    DCL$MOVCHAR             ;COPY CHARACTER WHICH STOPPED GETOKEN
                E1   11  0402          BRB     20$                     ;KEEP GETTING TOKENS UNTIL END OF VALUE
             FBF9'   30  0404  50$:    BSBW    DCL$MARKEDTOKEN         ;GET DESCRIPTOR OF VALUE STRING
                24   13  0407          BEQL    80$                     ;ERROR IF NO VALUE
             15 A8   91  0409          CMPB    ENT_B_VALTYPE(R8),-     ;$QUOTED_STRING?
                13       040C                  #ENT_K_QUOTEDSTRING     ;
                03   13  040D          BEQL    51$                     ;IF SO, THEN SKIP COMPRESSION
             FBEE'   30  040F          BSBW    DCL$COMPRESS            ;COMPRESS QUOTED STRING
                         0412
                         0412  ;
                         0412  ; PEFORM APPROPRIATE SYNTAX CHECKING OR CONVERSION ON THE VALUE JUST FETCHED.
                         0412  ;
        50   10 A8   D0  0412  51$:    MOVL    ENT_L_USER_TYPE(R8),R0  ;ARE KEYWORDS ALLOWED?
                1E   12  0416          BNEQ    53$                     ;BRANCH IF SO
        50   15 A8   9A  0418          MOVZBL  ENT_B_VALTYPE(R8),R0    ;GET VALUE TYPE
           03   50   91  041C          CMPB    R0,#ENT_K_NUMBER        ;NUMBER?
                20   13  041F          BEQL    56$                     ;BRANCH IF SO
           05   50   91  0421          CMPB    R0,#ENT_K_DATETIME      ;DATETIME?
                23   13  0424          BEQL    59$                     ;BRANCH IF SO
           12   50   91  0426          CMPB    R0,#ENT_K_DELTATIME     ;DELTATIME?
                1E   13  0429          BEQL    59$                     ;BRANCH IF SO
                22   11  042B          BRB     65$
                         042D
                         042D  80$:    STATUS  VALREQ                  ;SET MISSING VALUE STATUS
                32   11  0434          BRB     90$                     ;EXIT WITH ERROR STATUS
                         0436
           53   6E   D0  0436  53$:    MOVL    (SP),R3                 ;GET CLASSIFICATION OF VALUE TOKEN
```

```
    0034  30  0439      BSBW    DCL$PROCESS_KEYWORD       ;PROCESS KEYWORD
   29 50  E9  043C      BLBC    R0,90$                    ;BRANCH IF ERROR
      1E  11  043F      BRB     70$                       ;CLEAN UP AND RETURN
    0190  30  0441 56$: BSBW    DCL$NUMBER                ;PROCESS $NUMBER
   21 50  E9  0444      BLBC    R0,90$                    ;BRANCH IF ERROR
      06  11  0447      BRB     65$                       ;GENERATE DESCRIPTOR
    01BD  30  0449 59$: BSBW    DCL$DATETIME              ;PROCESS $DATETIME
   19 50  E9  044C 60$: BLBC    R0,90$                    ;BRANCH IF ERROR
              044F
              044F ;
              044F ; GENERATE VALUE DESCRIPTOR
              044F ;
      54  D4  044F 65$: CLRL    R4                        ;CLEAR ENTITY NUMBER
55    6E  D0  0451      MOVL    (SP),R5                   ;GET CLASSIFICATION OF VALUE TOKEN (R3)
      56  D4  0454      CLRL    R6                        ;CLEAR THE FLAGS
59    58  D0  0456      MOVL    R8,R9                     ;SET ADDRESS OF ENTITY BLOCK
57    51  7D  0459      MOVQ    R1,R7                     ;SET LENGTH AND OFFSET OF VALUE
   FBA1'  30  045C      BSBW    DCL$GENDESCR              ;GENERATE RESULT PARSE DESCRIPTOR
              045F
              045F ;
              045F ; CLEAN UP AND RETURN
              045F ;
   FB9E'  30  045F 70$: BSBW    DCL$TESTBLANK             ;THROW AWAY INSIGNIFICANT BLANKS
   FB9B'  30  0462      BSBW    DCL$GENTERM               ;AND SET TERMINATOR IN LAST TOKEN
50    01  D0  0465      MOVL    #1,R0                     ;SET SUCCESSFUL
   C4 AA  97  0468 90$: DECB    WRK_B_VALLEV(R10)         ;DECREASE THE VALUE LEVEL
01F8 8F   BA  046B      POPR    #^M<R3,R4,R5,R6,R7,R8>    ;RESTORE REGISTERS
          05  046F      RSB
```

```
                        0470          .SBTTL  PROCESS KEYWORD
                        0470    ;+
                        0470    ; DCL$PROCESS_KEYWORD - PROCESS KEYWORD
                        0470    ;
                        0470    ; THIS ROUTINE IS CALLED TO PROCESS A KEYWORD AND TO EMIT A DESCRIPTOR
                        0470    ; TO THE RESULT PARSE TABLE.
                        0470    ;
                        0470    ; INPUTS:
                        0470    ;
                        0470    ;       R0 = ADDRESS OF KEYWORD LIST HEADER
                        0470    ;       R1/R2 = KEYWORD DESCRIPTOR
                        0470    ;       R3 = ENTITY TYPE
                        0470    ;       R8 = ADDRESS OF ENTITY BLOCK
                        0470    ;       R10 = BASE ADDRESS OF COMMAND WORK AREA.
                        0470    ;
                        0470    ; OUTPUTS:
                        0470    ;
                        0470    ;       R0 = STATUS
                        0470    ;-
                        0470
                        0470    DCL$PROCESS_KEYWORD:                                    ;PROCESS KEYWORD
                        0470
                        0470    ;
                        0470    ; VALIDATE THE KEYWORD
                        0470    ;
58   DE AA   50    C1   0470            ADDL3   R0,WRK_L_TAB_VEC(R10),R8        ;GET ADDRESS OF KEYWORD LIST
                        0475            ASSUME  PTR_V_KEYWORD EQ 21             ;
                        0475            ASSUME  V_KEYWD EQ 1                    ;
        56   02    D0   0475            MOVL    #M_KEYWD,R6                     ;SPECIFY THAT FULL LENGTH BE VALIDAT
           FD89    30   0478            BSBW    DCL$FIND_KEYWORD                ;VALIDATE THE KEYWORD
          37 50    E9   047B            BLBC    R0,90$                          ;EXIT IF INVALID
                        047E    ;
                        047E    ; CHECK NEGATION
                        047E    ;
                        047E            ASSUME  V_NEGAT EQ 0                    ;
          0C 56    E9   047E            BLBC    R6,10$                          ;IF KEYWORD NOT NEGATED, THEN SKIP
                        0481            STATUS  NOTNEG                          ;ASSUME KEYWORD NOT NEGATABLE
             01    E1   0488            BBC     #ENT_V_NEG,-                    ;EXIT IF NEGATED
       28 04 A8         048A                    ENT_W_FLAGS(R8),90$             ;
                        048D
                        048D    ;
                        048D    ; PROCESS CHANGE LIST - IF ANY
                        048D    ;
          0C A8    D5   048D    10$:    TSTL    ENT_L_SYNTAX(R8)                ;ALTERNATE COMMAND SYNTAX SPECIFIED?
             0B    13   0490            BEQL    20$                             ;BRANCH IF NOT
             51    DD   0492            PUSHL   R1                              ;SAVE R1
           03E1    30   0494            BSBW    DCL$CHANGE_SYNTAX               ;CHANGE COMMAND SYNTAX
        51 8ED0         0497            POPL    R1                              ;RESTORE R1
                        049A            ASSUME  PTR_V_SYNTAX EQ 22              ;
                        049A            ASSUME  V_SYNTAX EQ 2                   ;
        56   04    C8   049A            BISL    #M_SYNTAX,R6                    ;SET SYNTAX CHANGE BIT
                        049D
                        049D    ;
                        049D    ; GENERATE VALUE (KEYWORD) DESCRIPTOR
                        049D    ;
        54   57    D0   049D    20$:    MOVL    R7,R4                           ;GET KEYWORD NUMBER
        55   53    D0   04A0            MOVL    R3,R5                           ;GET CLASSIFICATION OF VALUE TOKEN
```

```
7E    57   7D  04A3   932          MOVQ    R7,-(SP)               ;SAVE KEYWORD NUM AND ADDR OF ENTITY
59    58   D0  04A6   933          MOVL    R8,R9                  ;SET ADDRESS OF ENTITY BLOCK
57    51   7D  04A9   934          MOVQ    R1,R7                  ;SET LENGTH AND OFFSET OF VALUE
    FB51'  30  04AC   935          BSBW    DCL$GENDESCR           ;GENERATE RESULT PARSE DESCRIPTOR
57    8E   7D  04AF   936          MOVQ    (SP)+,R7               ;RESTORE SAVED REGISTERS
               04B2   937
               04B2   938 ;
               04B2   939 ; DO WE HAVE A KEYWORD VALUE LEFT TO PROCESS?
               04B2   940 ;
     FDFF  30  04B2   941          BSBW    DCL$CHECK_VALUES       ;MAKE INDIRECTLY RECURSIVE CALL
           05  04B5   942 90$:     RSB                            ;RETURN WITH STATUS
```

```
                  04B6          .SBTTL  PROCESS FILE SPECIFICATION
                  04B6  ;+
                  04B6  ; DCL$PROCFILE - PROCESS FILE SPECIFICATION.
                  04B6  ;
                  04B6  ; THIS ROUTINE IS CALLED TO SCAN A FILE SPECIFICATION FOR SYNTACTIC CORRECT-
                  04B6  ; NESS AND TO EMIT A FILE DESCRIPTOR TO THE RESULT PARSE TABLE.
                  04B6  ;
                  04B6  ; INPUTS:
                  04B6  ;
                  04B6  ;       R3 = VALUE CONTEXT (PTR_K_QUALVALU OR PTR_K_PARAMETR) FOR DESCRIPTOR
                  04B6  ;       R8 = ADDRESS OF ENTITY BLOCK
                  04B6  ;       R10 = BASE ADDRESS OF COMMAND WORK AREA.
                  04B6  ;
                  04B6  ; OUTPUTS:
                  04B6  ;
                  04B6  ;       R0 = STATUS
                  04B6  ;       R1/R2 = DESCRIPTOR OF FILE SPECIFICATION STRING, IF SUCCESSFUL
                  04B6  ;-
                  04B6
                  04B6  DCL$PROCFILE::                                  ;PROCESS FILE SPECIFICATION
                  04B6  ;
                  04B6  ; SET STATE TO START OF FILE SPECIFICATION.
                  04B6  ;
 01F8 8F     BB   04B6          PUSHR   #^M<R3,R4,R5,R6,R7,R8>          ;SAVE REGISTERS
    FB43'    30   04BA          BSBW    DCL$MOVCHAR                     ;MOVE TERMINATOR CHARACTER
    FB40'    30   04BD          BSBW    DCL$MARK                        ;MARK CURRENT POSITION IN BUFFER
                  04C0  DCL$PROCFILE1:
    FB3D'    30   04C0          BSBW    DCL$GETOKEN                     ;GET NEXT TOKEN
       13    12   04C3          BNEQ    5$                              ;IF NEQ THEN FOUND
                  04C5
                  04C5  ;
                  04C5  ; NO TOKEN OBTAINED - CHECK FOR NULL NODE.
                  04C5  ;
                  04C5  ;       '::DEVICE...'
                  04C5  ;
 50    3A    91   04C5          CMPB    #^A/:/,R0                       ;FIRST COLON PRESENT?
       2A    12   04C8          BNEQ    20$                             ;IF NEQ NO
    FB33'    30   04CA          BSBW    DCL$MOVCHAR                     ;MOVE TERMINATOR TO COMMAND BUFFER
    FB30'    30   04CD          BSBW    DCL$SETCHAR                     ;PEEK AT NEXT CHARACTER IN INPUT BUFFER
 50    3A    91   04D0          CMPB    #^A/:/,R0                       ;SECOND COLON PRESENT?
       13    13   04D3          BEQL    7$                              ;IF YES THEN LOOK FOR MORE NODES,DEVICES
     00AF    31   04D5          BRW     950$                            ;IF NO THEN ERROR
                  04D8
                  04D8  ;
                  04D8  ; TOKEN OBTAINED - FORM MUST BE:
                  04D8  ;
                  04D8  ;       'NODE::' OR,
                  04D8  ;       'DEVICE:' OR,
                  04D8  ;       'FILENAME.' OR,
                  04D8  ;       'FILENAME;' OR,
                  04D8  ;       'FILENAME-' OR,
                  04D8  ;       'FILENAME '.
                  04D8  ;
 50    3A    91   04D8  5$:     CMPB    #^A/:/,R0                       ;NODE OR DEVICE NAME?
       58    12   04DB          BNEQ    60$                             ;IF NEQ NO
                  04DD
                  04DD  ;
```

```
                  04DD      ; NODE OR DEVICE NAME
                  04DD      ;
       FB20'  30  04DD              BSBW    DCL$MOVCHAR             ;MOVE TERMINATOR TO COMMAND BUFFER
       FB1D'  30  04E0              BSBW    DCL$SETCHAR             ;PEEK AT NEXT CHARACTER IN INPUT BUFFER
   50    3A   91  04E3              CMPB    #^A/:/,R0               ;NODE NAME?
         07   12  04E6              BNEQ    10$                     ;IF NEQ NO
       FB15'  30  04E8      7$:     BSBW    DCL$MOVBTOKN            ;MOVE TERMINATOR AND GET BLANK TOKEN
         07   13  04EB              BEQL    20$                     ;IF EQL NONE
         E9   11  04ED              BRB     5$                      ;LOOP TO ALLOW ANOTHER NODE NAME
                  04EF
       FB0E'  30  04EF      10$:    BSBW    DCL$GTBTOKEN            ;GET NEXT BLANK TOKEN
         41   12  04F2              BNEQ    60$                     ;IF NEQ TOKEN OBTAINED
                  04F4
                  04F4      ;
                  04F4      ; NO TOKEN OBTAINED - FORM MUST BE:
                  04F4      ;
                  04F4      ;               '[DIRECTORY]' OR,
                  04F4      ;               '<DIRECTORY>' OR,
                  04F4      ;               '.TYPE' OR,
                  04F4      ;               ';VERSION'.
                  04F4      ;
   50  5B 8F  91  04F4      20$:    CMPB    #^A/[/,R0               ;DIRECTORY?
         05   13  04F8              BEQL    30$                     ;IF EQL YES
   50    3C   91  04FA              CMPB    #^A/</,R0               ;DIRECTORY?
         36   12  04FD              BNEQ    60$                     ;IF NEQ NO
   5C  02 A0  9E  04FF      30$:    MOVAB   2(R0),AP                ;SAVE DIRECTORY TERMINATOR CHARACTER
       FAFA'  30  0503              BSBW    DCL$MOVBTOKN            ;MOVE TERMINATOR AND GET BLANK TOKEN
   50    2C   91  0506              CMPB    #^A/,/,R0               ;OLD STYLE DIRECTORY?
         11   12  0509              BNEQ    40$                     ;IF NEQ NO
                  050B
                  050B      ;
                  050B      ; OLD STYLE DIRECTORY - FORM IS:
                  050B      ;
                  050B      ;               '[GROUP,MEMBER]'.
                  050B      ;
                  050B
       FAF2'  30  050B              BSBW    DCL$MOVBTOKN            ;MOVE TERMINATOR AND GET BLANK TOKEN
   5C    50   91  050E              CMPB    R0,AP                   ;EXPECTED TERMINATOR?
         1D   13  0511              BEQL    50$                     ;BRANCH IF YES
                  0513      120$:   STATUS  DIRECT                  ;RETURN INVALID DIRECTORY SYNTAX
         66   11  051A              BRB     900$                    ;AND EXIT WITH ERROR
                  051C
                  051C      ;
                  051C      ; NEW STYLE DIRECTORY - FORM IS:
                  051C      ;
                  051C      ;               '<SUB.SUB...SUB>'.
                  051C      ;
                  051C
   5C    50   91  051C      40$:    CMPB    R0,AP                   ;EXPECTED TERMINATOR?
         0F   13  051F              BEQL    50$                     ;IF EQL YES
   50    2E   91  0521              CMPB    #^A/./,R0               ;SUBDIRECTORY?
         05   13  0524              BEQL    45$                     ;IF EQL, TERMINATOR OK
   50    2D   91  0526              CMPB    #^A/-/,R0               ;BACKUP SYMBOL?
         E8   12  0529              BNEQ    120$                    ;IF NEQ, INVALID SYNTAX
       FAD2'  30  052B      45$:    BSBW    DCL$MOVBTOKN            ;MOVE TERMINATOR AND GET BLANK TOKEN
         EC   11  052E              BRB     40$                     ;
       FACD'  30  0530      50$:    BSBW    DCL$MOVBTOKN            ;MOVE TERMINATOR AND GET BLANK TOKEN
```

```
        BF     11  0533           BRB     20$                         ;CHECK FOR MORE DIRECTORIES
                   0535
                   0535      ;
                   0535      ; FILENAME, TYPE, OR VERSION - FORM MUST BE:
                   0535      ;
                   0535      ;            'FILENAME.' OR,
                   0535      ;            'FILENAME;' OR,
                   0535      ;            'FILENAME-FILENAME' OR,
                   0535      ;            'FILENAME ' OR,
                   0535      ;            '.TYPE' OR,
                   0535      ;            ';VERSION'.
                   0535      ;
                   0535
  50    2D     91  0535 60$:      CMPB    #^A/-/,R0                   ;CONTINUED FILENAME?
        05     12  0538           BNEQ    65$                         ;IF NEQ NO
     FAC3'     30  053A           BSBW    DCL$MOVBTOKN                ;MOVE TERMINATOR AND GET BLANK TOKEN
        F6     11  053D           BRB     60$                         ;LOOK FOR MORE OF FILENAME
  50    2E     91  053F 65$:      CMPB    #^A/./,R0                   ;FILE TYPE?
        0D     12  0542           BNEQ    70$                         ;IF NEQ NO
     FAB9'     30  0544 67$:      BSBW    DCL$MOVBTOKN                ;MOVE TERMINATOR AND GET BLANK TOKEN
  50    2D     91  0547           CMPB    #^A/-/,R0                   ;CONTINUED FILE TYPE?
        F8     13  054A           BEQL    67$                         ;IF EQL YES
  50    2E     91  054C           CMPB    #^A/./,R0                   ;NEW VERSION FORMAT?
        05     13  054F           BEQL    80$                         ;IF EQL YES
                   0551
                   0551      ;
                   0551      ; FILE VERSION - FORM MUST BE:
                   0551      ;
                   0551      ;            ';VERSION'.
                   0551      ;
                   0551
  50    3B     91  0551 70$:      CMPB    #^A/;/,R0                   ;FILE VERSION?
        0D     12  0554           BNEQ    100$                        ;IF NEQ NO
     FAA7'     30  0556 80$:      BSBW    DCL$MOVBTOKN                ;MOVE TERMINATOR AND GET BLANK TOKEN
        08     12  0559           BNEQ    100$                        ;IF NON-NULL, GOT VERSION STRING
  50    2D     91  055B           CMPB    #^A/-/,R0                   ;NEGATIVE VERSION?
        03     12  055E           BNEQ    100$                        ;IF NEQ NO
     FA9D'     30  0560           BSBW    DCL$MOVBTOKN                ;MOVE TERMINATOR AND GET BLANK TOKEN
                   0563
                   0563      ;
                   0563      ; FILE SPECIFICATION SCANNED - CHECK RESULT
                   0563      ;
     FA9A'     30  0563 100$:     BSBW    DCL$MARKEDTOKEN                     ;GET DESCRIPTOR OF PARAMETER STRING
        1F     13  0566           BEQL    950$                                ;IF NONE, EXIT WITH ERROR
                   0568
  7E    51     7D  0568           MOVQ    R1,-(SP)                            ;COPY DESCRIPTOR
                   056B      ;    CLRQ    -(SP)                               ;CREATE ITEM LIST
                   056B      ;    MOVL    #FSCN$_FILESPEC@16,-(SP)            ;SET ITEM CODE
                   056B      ;    MOVL    SP,R0                               ;GET ADDRESS OF ITEM LIST
                   056B      ;    $FILESCAN_S  SRCSTR=12(R0),-                ;SCAN THE FILE SPEC
                   056B      ;            VALUELST=(R0)                       ;
                   056B      ;    ADDL    #3*4,SP                             ;POP THE ITEM LIST
  57    8E     7D  056B           MOVQ    (SP)+,R7                            ;POP THE FILESPEC DESCR
                   056E      ;    BLBC    R0,950$                             ;RETURN ERROR IF LBC
                   056E
        54     D4  056E           CLRL    R4                                  ;CLEAR ENTITY BLOCK ADDRESS
        56     D4  0570           CLRL    R6                                  ;CLEAR FLAGS
```

```
55   6E  D0  0572  1115          MOVL    (SP),R5                      ;GET VALUE CONTEXT (R3 ON ENTRY)
   FA88'  30  0575  1116          BSBW    DCL$GENDESCR                 ;GENERATE RESULT PARSE DESCRIPTOR
51   57  7D  0578  1117          MOVQ    R7,R1                        ;RETURN FILESPEC DESCRIPTOR TO CALLE
             057B  1118          STATUS  NORMAL                       ;SET SUCCESSFUL COMPLETION
01F8 8F  BA  0582  1119 900$:    POPR    #^M<R3,R4,R5,R6,R7,R8>       ;RESTORE REGISTERS
         05  0586  1120          RSB                                  ;RETURN FROM SUBROUTINE
             0587  1121
             0587  1122 950$:    STATUS  NULFIL                       ;SET INVALID FILE SPEC STATUS
01F8 8F  BA  058E  1123          POPR    #^M<R3,R4,R5,R6,R7,R8>       ;RESTORE REGISTERS
         05  0592  1124          RSB                                  ;RETURN FROM SUBROUTINE
```

```
                0593          .SBTTL  PROCESS REST OF LINE VALUE
                0593    ;+
                0593    ; DCL$REST_OF_LINE  - PROCESS REST OF LINE VALUE
                0593    ;
                0593    ; THIS ROUTINE IS PROCESS A $REST_OF_LINE VALUE AND PREPARE TO EMIT A
                0593    ; DESCRIPTOR TO THE RESULT PARSE TABLE.
                0593    ;
                0593    ; INPUTS:
                0593    ;
                0593    ;       R3 = CLASSIFICATION OF VALUE (PTR_K_QUALVALU OR PTR_K_PARAMETR)
                0593    ;       R8 = ADDRESS OF ENTITY BLOCK
                0593    ;       R10 = BASE ADDRESS OF COMMAND WORK AREA.
                0593    ;
                0593    ; OUTPUTS:
                0593    ;
                0593    ;       R1/R2 = DESCRIPTOR OF $REST_OF_LINE VALUE
                0593    ;-
                0593
                0593    DCL$REST_OF_LINE:                              ;PROCESS REST OF LINE
                0593    ;
                0593    ; GET REST OF LINE.
                0593    ;
      53   DD   0593          PUSHL   R3                         ;SAVE TOKEN TYPE
      0D   E0   0595          BBS     #WRK_V_USRMODE,-           ;IF SET THEN USER MODE PARSE
04 F0 AA        0597                  WRK_Q_FLAGS(R10),10$       ;
                059A          SETBIT  PRC_V_IND,-                ;DISABLE @ INDIRECTION, SO THAT @
                059A                  PRC_W_FLAGS(R11)           ;CAN BE PASSED IN THE VALUE
   FA5F'   30   059E    10$:  BSBW    DCL$MOVCHAR                ;COPY TERMINATOR INTO EXPANSION BUFF
   FA5C'   30   05A1          BSBW    DCL$SETNBLK                ;SKIP LEADING BLANKS
   FA59'   30   05A4          BSBW    DCL$MARK                   ;MARK START OF VALUE STRING
   FA56'   30   05A7    20$:  BSBW    DCL$MOVCHAR                ;COPY NEXT CHARACTER
      FB   12   05AA          BNEQ    20$                        ;LOOP UNTIL END OF LINE
      0D   E0   05AC          BBS     #WRK_V_USRMODE,-           ;IF SET THEN USER MODE PARSE
04 F0 AA        05AE                  WRK_Q_FLAGS(R10),30$       ;
                05B1          CLRBIT  PRC_V_IND,-                ;RE-ENABLE @ INDIRECTION AGAIN
                05B1                  PRC_W_FLAGS(R11)           ;
   FA48'   30   05B5    30$:  BSBW    DCL$MARKEDTOKEN            ;GET DESCRIPTOR OF MARKED TOKEN
      51   D7   05B8          DECL    R1                         ;SUBTRACT OUT EOL CHARACTER
                05BA
                05BA    ;
                05BA    ; GENERATE VALUE DESCRIPTOR
                05BA    ;
      54   D4   05BA          CLRL    R4                         ;CLEAR ENTITY NUMBER
55    8E   D0   05BC          MOVL    (SP)+,R5                   ;GET CLASSIFICATION OF VALUE TOKEN (
      56   D4   05BF          CLRL    R6                         ;CLEAR THE FLAGS
59    58   D0   05C1          MOVL    R8,R9                      ;SET ADDRESS OF ENTITY BLOCK
57    51   7D   05C4          MOVQ    R1,R7                      ;SET LENGTH AND OFFSET OF VALUE
   FA36'   30   05C7          BSBW    DCL$GENDESCR               ;GENERATE RESULT PARSE DESCRIPTOR
                05CA
                05CA    ;
                05CA    ; CLEAN UP AND RETURN
                05CA    ;
   FA33'   30   05CA          BSBW    DCL$TESTBLANK              ;THROW AWAY INSIGNIFICANT BLANKS
   FA30'   30   05CD          BSBW    DCL$GENTERM                ;AND SET TERMINATOR IN LAST TOKEN
50    03   D0   05D0          MOVL    #3,R0                      ;SET SUCCESSFUL
      05        05D3          RSB                                ;RETURN
```

```
                         05D4           .SBTTL  PROCESS $NUMBER VALUE
                         05D4   ;+
                         05D4   ; DCL$NUMBER - PROCESS $NUMBER VALUE
                         05D4   ;
                         05D4   ; THIS ROUTINE IS CALLED TO SCAN A $NUMBER VALUE FOR SYNTACTIC CORRECTNESS,
                         05D4   ; CONVERT THAT NUMBER TO A DECIMAL STRING, AND PREPARE TO EMIT A DESCRIPTOR
                         05D4   ; FOR THAT DECIMAL STRING TO THE RESULT PARSE TABLE.
                         05D4   ;
                         05D4   ; INPUTS:
                         05D4   ;
                         05D4   ;       R1/R2 = DESCRIPTOR OF VALUE STRING
                         05D4   ;       R8 = ADDRESS OF ENTITY BLOCK
                         05D4   ;       R10 = BASE ADDRESS OF COMMAND WORK AREA.
                         05D4   ;
                         05D4   ; OUTPUTS:
                         05D4   ;
                         05D4   ;       R0 = STATUS
                         05D4   ;       R1/R2 = DESCRIPTOR OF DECIMAL STRING, IF SUCCESSFUL
                         05D4   ;-
                         05D4
                         05D4   DCL$NUMBER:                                     ;PROCESS $NUMBER
                51   DD  05D4           PUSHL   R1                              ;SAVE STRING LENGTH
           52   51   7D  05D6           MOVQ    R1,R2                           ;COPY DESCRIPTOR OF STRING
           51   01   D0  05D9           MOVL    #1,R1                           ;SET DECIMAL RADIX
              FA21'  30  05DC           BSBW    DCL$CNVNOEDIT                   ;CHECK THE NUMBER
                50   D5  05DF           TSTL    R0                              ;WAS STRING COMPLETELY PARSED?
                1B   12  05E1           BNEQ    80$                             ;BRANCH IF ERROR
                50 8ED0  05E3           POPL    R0                              ;GET THE STRING LENGTH
   F486 CA      50   C2  05E6           SUBL    R0,WRK_L_EXPANDPTR(R10)         ;REMOVE IT FROM THE EXPANSION BUFFER
           50   51   D0  05EB           MOVL    R1,R0                           ;GET THE VALUE TO CONVERT
              FA0F'  30  05EE           BSBW    DCL$CBTA_DEC                    ;CONVERT THE BINARY VALUE TO DECIMAL
   F486 CA      51   C0  05F1           ADDL    R1,WRK_L_EXPANDPTR(R10)         ;UPDATE THE EXPANSION BUFFER PTR
                         05F6           STATUS  NORMAL                          ;SET NORMAL STATUS
                     05  05FD           RSB                                     ;RETURN
                         05FE
                51 8ED0  05FE   80$:    POPL    R1                              ;POP STRING LENGTH
                         0601           STATUS  NUMBER                          ;SET STATUS
                     05  0608           RSB                                     ;RETURN
```

```
                              0609          .SBTTL  PROCESS $DATETIME VALUE
                              0609     ;+
                              0609     ; DCL$DATETIME - PROCESS $DATETIME VALUE
                              0609     ;
                              0609     ; THIS ROUTINE IS CALLED TO SCAN A $DATETIME VALUE FOR SYNTACTIC CORRECTNESS,
                              0609     ; CONVERT THAT TIME TO AN ABSOLUT TIME, AND PREPARE TO EMIT A DESCRIPTOR FOR
                              0609     ; THAT ABSOLUTE TIME STRING TO THE RESULT PARSE TABLE.
                              0609     ;
                              0609     ; INPUTS:
                              0609     ;
                              0609     ;       R0 = ENT_K_DATETIME OR ENT_K_DELTATIME
                              0609     ;       R1/R2 = DESCRIPTOR OF VALUE STRING
                              0609     ;       R8 = ADDRESS OF ENTITY BLOCK
                              0609     ;       R10 = BASE ADDRESS OF COMMAND WORK AREA.
                              0609     ;
                              0609     ; OUTPUTS:
                              0609     ;
                              0609     ;       R0 = STATUS
                              0609     ;       R1/R2 = DESCRIPTOR OF ABSOLUTE TIME STRING, IF SUCCESSFUL
                              0609     ;-
                              0609
                              0609     DCL$DATETIME:                                           ;PROCESS $DATETIME OR $DELTATIME
                              0609
                              0609     ;
                              0609     ; ALLOCATE INPUT DESCRIPTOR, TIME QUADWORD, AND OUTPUT DESCRIPTOR.
                              0609     ;
                 50     DD    0609          PUSHL   R0                                      ;SAVE TYPE CODE
            7E   51     7D    060B          MOVQ    R1,-(SP)                                ;SAVE STRING DESCRIPTOR
            53   5E     D0    060E          MOVL    SP,R3                                   ;GET ADDRESS OF THE DESCRIPTOR
                 7E     7C    0611          CLRQ    -(SP)                                   ;ALLOCATE A QUADWORD TIME BUFFER
            54   5E     D0    0613          MOVL    SP,R4                                   ;SAVE ITS ADDRESS
     50    F892  CA     9E    0616          MOVAB   WRK_G_BUFFER+WRK_C_CMDBUFSIZ(R10),R0    ;FIND END OF BUFFER
            51   50     C0    061B          ADDL    R0,R1                                   ;ADD LENGTH OF STRING TO BE REMOVED
     51    F486  CA     C2    061E          SUBL    WRK_L_EXPANDPTR(R10),R1                 ;CALCULATE AMOUNT LEFT
            7E   51     7D    0623          MOVQ    R1,-(SP)                                ;PUSH OUTPUT DESCRIPTOR
                              0626
                              0626     ;
                              0626     ; CALL LIB$CVT_TIME.
                              0626     ;
     50      18  AE     D0    0626          MOVL    24(SP),R0                               ;GET TYPE CODE
                 5E     DD    062A          PUSHL   SP                                      ;PUSH ADDRESS OF DESCRIPTOR TWICE
             04  AE     9F    062C          PUSHAB  4(SP)                                   ;
            7E   53     7D    062F          MOVQ    R3,-(SP)                                ;PUSH INPUT DESCRIPTOR AND BUFFER
            05   50     D1    0632          CMPL    R0,#ENT_K_DATETIME                      ;IS IT $DATETIME
            09   12           0635          BNEQ    50$                                     ;NO, THEN $DELTATIME
  00000000'EF    04     FB    0637          CALLS   #4,LIB$CVT_TIME                         ;PARSE THE TIME SPECIFICATION
            07   11           063E          BRB     60$                                     ;BRANCH
  00000000'EF    04     FB    0640     50$: CALLS   #4,LIB$CVT_DTIME                        ;PARSE THE TIME SPECIFICATION
                              0647
                              0647     ;
                              0647     ; RESTORE STACK AND BRANCH IF ERROR.
                              0647     ;
            51   8E     7D    0647     60$: MOVQ    (SP)+,R1                                ;GET OUTPUT DESCRIPTOR
            5E   08     C0    064A          ADDL    #8,SP                                   ;POP TIME BUFFER
            53   8E     7D    064D          MOVQ    (SP)+,R3                                ;GET INPUT DESCRIPTOR
            55   8ED0         0650          POPL    R5                                      ;POP THE TYPE CODE
         0B  50         E9    0653          BLBC    R0,90$                                  ;BRANCH IF ERROR
```

```
                             0656
                             0656  ;
                             0656  ; RESET EXPANSION BUFFER POINTER.
                             0656  ;
        F486 CA   53   C2    0656          SUBL    R3,WRK_L_EXPANDPTR(R10)      ;REMOVE INPUT FROM THE EXPANSION BUF
        F486 CA   51   C0    065B          ADDL    R1,WRK_L_EXPANDPTR(R10)      ;INSERT OUTPUT IN THE EXPANSION BUFF
                       05    0660          RSB                                  ;RETURN
                             0661
    00038018 8F   50   D1    0661  90$:    CMPL    R0,#CLI$_BUFOVF              ;WAS IT BUFFER OVERFLOW
                  0C   13    0668          BEQL    95$                          ;YES, THEN RETURN THAT STATUS
             05   55   D1    066A          CMPL    R5,#ENT_K_DATETIME           ;IS IT $DATETIME
                  08   12    066D          BNEQ    97$                          ;NO, THEN $DELTATIME
                             066F          STATUS  IVATIME                      ;SET INVALID ABSOLUTE TIME
                       05    0676  95$:    RSB                                  ;RETURN
                             0677  97$:    STATUS  IVDTIME                      ;SET INVALID DELTA TIME
                       05    067E          RSB                                  ;RETURN
                             067F
                             067F  ;
                             067F  ; DEFINE A STUB DCL$SCOPY_DXDX ROUTINE FOR USE BY LIB$CVTTIME.
                             067F  ;
                             067F  ; INPUTS:  INPUT DESCRIPTOR, OUTPUT DESCRIPTOR
                             067F  ;
                     003C    067F          .ENTRY  DCL$SCOPY_DXDX,^M<R2,R3,R4,R5>
          51   04 AC   D0    0681          MOVL    4(AP),R1                     ;GET INPUT DESCRIPTOR
          52   08 AC   D0    0685          MOVL    8(AP),R2                     ;GET OUTPUT DESCRIPTOR
             62   61   B1    0689          CMPW    (R1),(R2)                    ;WILL STRING FIT
                  0D   1A    068C          BGTRU   90$                          ;NO, THEN RETURN ERROR
             62   61   3C    068E          MOVZWL  (R1),(R2)                    ;COPY STRING LENGTH
    04 B2    04 B1   62   28    0691          MOVC3   (R2),@4(R1),@4(R2)           ;COPY DESCRIPTOR
             50   01   D0    0697          MOVL    #1,R0                        ;SET NORMAL STATUS
                       04    069A          RET                                  ;RETURN
                             069B
                             069B  90$:    STATUS  BUFOVF                       ;SET BUFFER OVERFLOW
                       04    06A2          RET                                  ;RETURN
```

```
                    06A3          .SBTTL  PROCESS $PARENTHESIZED_VALUE VALUE
                    06A3  ;+
                    06A3  ; DCL$PAREN_VALUE - PROCESS $PARENTHESIZED_VALUE VALUE
                    06A3  ;
                    06A3  ; THIS ROUTINE IS CALLED TO PROCESS A $PARENTHESIZED_VALUE VALUE AND TO EMIT A
                    06A3  ; DESCRIPTOR TO THE RESULT PARSE TABLE.
                    06A3  ;
                    06A3  ; INPUTS:
                    06A3  ;
                    06A3  ;       R3 = CLASSIFICATION OF VALUE (PTR_K_QUALVALU OR PTR_K_PARAMETR)
                    06A3  ;       R5 = -1 IF FIRST VALUE IN A PARENTHESIZED LIST
                    06A3  ;       R8 = ADDRESS OF ENTITY BLOCK
                    06A3  ;       R10 = BASE ADDRESS OF COMMAND WORK AREA.
                    06A3  ;
                    06A3  ; OUTPUTS:
                    06A3  ;
                    06A3  ;       R0 = STATUS
                    06A3  ;       R1/R2 = DESCRIPTOR OF VALUE
                    06A3  ;-
                    06A3
                    06A3  DCL$PAREN_VALUE:                              ;PROCESS LIST
      01F8 8F   BB  06A3          PUSHR   #^M<R3,R4,R5,R6,R7,R8>    ;SAVE REGISTERS
        54 01   D0  06A7          MOVL    #1,R4                     ;SET PAREN COUNT
                    06AA
                    06AA  ;
                    06AA  ; CHECK FOR AND SAVE LEADING PAREN.  IF NONE, PROCESS AS A FILE SPEC.
                    06AA  ;
         F953'  30  06AA          BSBW    DCL$MOVCHAR               ;COPY TERMINATOR INTO EXPANSION BUFFER
         F950'  30  06AD          BSBW    DCL$MARK                  ;MARK START OF VALUE STRING
55 FFFFFFFF 8F  D1  06B0          CMPL    #-1,R5                    ;WAS PAREN ALREADY SEEN?
            0B  13  06B7          BEQL    10$                       ;YES, THEN BRANCH
         F944'  30  06B9          BSBW    DCL$SETNBLK               ;SKIP LEADING BLANKS
        28  50  91  06BC          CMPB    R0,#^A'('                 ;IS FIRST CHAR AN OPEN PAREN?
            09  13  06BF          BEQL    20$                       ;YES, THEN BRANCH
          FDFC  31  06C1          BRW     DCL$PROCFILE1             ;NO, PROCESS AS A FILE SPEC
                    06C4
         F939'  30  06C4  10$:    BSBW    DCL$BACKUPMOVE            ;RESTORE FIRST PAREN
         F936'  30  06C7          BSBW    DCL$MARK                  ;MARK START OF VALUE STRING
         F933'  30  06CA  20$:    BSBW    DCL$MOVCHAR               ;COPY PAREN INTO EXPANSION BUFFER
                    06CD
                    06CD  ;
                    06CD  ; GET REST OF $PARENTHESIZED_VALUE
                    06CD  ;
         F930'  30  06CD  25$:    BSBW    DCL$GTBTOKEN              ;COPY TOKEN INTO EXPANSION BUFFER
         F92D'  30  06D0          BSBW    DCL$MOVCHAR               ;COPY CHARACTER WHICH STOPPED GETOKEN
            35  13  06D3          BEQL    90$                       ;ERROR IF END OF LINE
        29  50  91  06D5          CMPB    R0,#^A')'                 ;CLOSE PAREN?
            06  12  06D8          BNEQ    30$                       ;NO, THEN SKIP
            54  D7  06DA          DECL    R4                        ;DECREMENT PAREN COUNT
            0B  13  06DC          BEQL    40$                       ;BRANCH IF DONE
            ED  11  06DE          BRB     25$                       ;KEEP GETTING TOKENS
        28  50  91  06E0  30$:    CMPB    R0,#^A'('                 ;OPEN PAREN?
            E8  12  06E3          BNEQ    25$                       ;NO, THEN SKIP
            54  D6  06E5          INCL    R4                        ;INCREMENT PAREN COUNT
            E4  11  06E7          BRB     25$                       ;KEEP GETTING TOKENS
                    06E9
                    06E9  ;
```

```
                     06E9   ; GENERATE VALUE DESCRIPTOR
                     06E9   ;
        F914'   30   06E9   40$:    BSBW    DCL$MARKEDTOKEN              ;GET DESCRIPTOR OF VALUE STRING
           54   D4   06EC           CLRL    R4                           ;CLEAR ENTITY NUMBER
    55     6E   D0   06EE           MOVL    (SP),R5                      ;GET CLASSIFICATION OF VALUE TOKEN (R3)
           56   D4   06F1           CLRL    R6                           ;CLEAR THE FLAGS
    59     58   D0   06F3           MOVL    R8,R9                        ;SET ADDRESS OF ENTITY BLOCK
    57     51   7D   06F6           MOVQ    R1,R7                        ;SET LENGTH AND OFFSET OF VALUE
        F904'   30   06F9           BSBW    DCL$GENDESCR                 ;GENERATE RESULT PARSE DESCRIPTOR
                     06FC
                     06FC   ;
                     06FC   ; CLEAN UP AND RETURN
                     06FC   ;
        F901'   30   06FC           BSBW    DCL$TESTBLANK                ;THROW AWAY INSIGNIFICANT BLANKS
        F8FE'   30   06FF           BSBW    DCL$GENTERM                  ;AND SET TERMINATOR IN LAST TOKEN
    01F8 8F     BA   0702           POPR    #^M<R3,R4,R5,R6,R7,R8>       ;RESTORE REGISTERS
    50     03   D0   0706           MOVL    #3,R0                        ;SET STATUS
                05   0709           RSB                                  ;
                     070A
    01F8 8F     BA   070A   90$:    POPR    #^M<R3,R4,R5,R6,R7,R8>       ;RESTORE REGISTERS
                     070E           STATUS  NOPAREN                      ;SET INVALID SYNTAX
                05   0715           RSB                                  ;
```

```
                        0716            .SBTTL  PROCESS $ACL VALUE
                        0716      ;+
                        0716      ; DCL$ACL - PROCESS $ACL VALUE
                        0716      ;
                        0716      ; THIS ROUTINE IS CALLED TO PROCESS A $ACL VALUE AND TO EMIT A
                        0716      ; DESCRIPTOR TO THE RESULT PARSE TABLE.
                        0716      ;
                        0716      ; INPUTS:
                        0716      ;
                        0716      ;       R3 = CLASSIFICATION OF VALUE (PTR_K_QUALVALU OR PTR_K_PARAMETR)
                        0716      ;       R5 = -1 IF FIRST VALUE IN A PARENTHESIZED LIST
                        0716      ;       R8 = ADDRESS OF ENTITY BLOCK
                        0716      ;       R10 = BASE ADDRESS OF COMMAND WORK AREA.
                        0716      ;
                        0716      ; OUTPUTS:
                        0716      ;
                        0716      ;       R0 = STATUS
                        0716      ;       R1/R2 = DESCRIPTOR OF VALUE
                        0716      ;-
                        0716
                        0716      DCL$ACL:                                      ;PROCESS LIST
          7E    01  D0  0716              MOVL    #1,-(SP)                      ;CLEAR SIGNIFICANT PAREN FLAG
                53  DD  0719              PUSHL   R3                            ;SAVE R3
                        071B
                        071B      ;
                        071B      ; DETERMINE IF PROCESSING FIRST LIST AFTER AN EQUAL SIGN AND HANDLE
                        071B      ; SPECIAL CASE OF POSSIBLY ONE OR TWO OPEN PARENTHESES.
                        071B      ;
             F8E2'  30  071B              BSBW    DCL$MOVCHAR                   ;COPY TERMINATOR INTO EXPANSION BUFFER
             F8DF'  30  071E              BSBW    DCL$SETNBLK                   ;SKIP LEADING BLANKS
             F8DC'  30  0721              BSBW    DCL$MARK                      ;MARK START OF VALUE STRING
          28    50  91  0724              CMPB    R0,#^A'('                     ;IS NEXT CHAR OPEN PAREN?
                10  13  0727              BEQL    10$                           ;YES, THEN SKIP
 55   FFFFFFFF  8F  D1  0729              CMPL    #-1,R5                        ;FIRST VALUE IN LIST?
                49  12  0730              BNEQ    85$                           ;NO, THEN ERROR
             F8CB'  30  0732              BSBW    DCL$BACKUPMOVE                ;YES, RESTORE FIRST PAREN
       04 AE    03  D0  0735              MOVL    #3,4(SP)                      ;MARK FIRST PAREN WAS SIGNIFICANT
          54    01  D0  0739      10$:    MOVL    #1,R4                         ;SET PAREN COUNT
             F8C1'  30  073C              BSBW    DCL$MARK                      ;MARK START OF VALUE STRING
             F8BE'  30  073F              BSBW    DCL$MOVCHAR                   ;COPY PAREN INTO EXPANSION BUFFER
                        0742
                        0742      ;
                        0742      ; GET REST OF $ACL
                        0742      ;
             F8BB'  30  0742      20$:    BSBW    DCL$GTBTOKEN                  ;COPY TOKEN INTO EXPANSION BUFFER
             F8B8'  30  0745              BSBW    DCL$MOVCHAR                   ;COPY CHARACTER WHICH STOPPED GETOKEN
                34  13  0748              BEQL    90$                           ;ERROR IF END OF LINE
          29    50  91  074A              CMPB    R0,#^A')'                     ;CLOSE PAREN?
                06  12  074D              BNEQ    30$                           ;NO, THEN SKIP
                54  D7  074F              DECL    R4                            ;DECREMENT PAREN COUNT
                0B  13  0751              BEQL    40$                           ;BRANCH IF DONE
                ED  11  0753              BRB     20$                           ;KEEP GETTING TOKENS
          28    50  91  0755      30$:    CMPB    R0,#^A'('                     ;OPEN PAREN?
                E8  12  0758              BNEQ    20$                           ;NO, THEN SKIP
                54  D6  075A              INCL    R4                            ;INCREMENT PAREN COUNT
                E4  11  075C              BRB     20$                           ;KEEP GETTING TOKENS
                        075E
```

```
                075E  1451 ;
                075E  1452 ; GENERATE VALUE DESCRIPTOR
                075E  1453 ;
   F89F'   30   075E  1454 40$:     BSBW    DCL$MARKEDTOKEN            ;GET DESCRIPTOR OF VALUE STRING
      54   D4   0761  1455          CLRL    R4                         ;CLEAR ENTITY NUMBER
55    8E   D0   0763  1456          MOVL    (SP)+,R5                   ;GET CLASSIFICATION OF VALUE TOKEN (R3)
      56   D4   0766  1457          CLRL    R6                         ;CLEAR THE FLAGS
59    58   D0   0768  1458          MOVL    R8,R9                      ;SET ADDRESS OF ENTITY BLOCK
57    51   7D   076B  1459          MOVQ    R1,R7                      ;SET LENGTH AND OFFSET OF VALUE
   F88F'   30   076E  1460          BSBW    DCL$GENDESCR               ;GENERATE RESULT PARSE DESCRIPTOR
                0771  1461
                0771  1462 ;
                0771  1463 ; CLEAN UP AND RETURN
                0771  1464 ;
   F88C'   30   0771  1465          BSBW    DCL$TESTBLANK              ;THROW AWAY INSIGNIFICANT BLANKS
   F889'   30   0774  1466          BSBW    DCL$GENTERM                ;AND SET TERMINATOR IN LAST TOKEN
     50 8ED0    0777  1467          POPL    R0                         ;SET STATUS
           05   077A  1468          RSB                                ;
                077B  1469
   F882'   30   077B  1470 85$:     BSBW    DCL$GTBTOKEN               ;COPY TOKEN INTO EXPANSION BUFFER
5E    08   C0   077E  1471 90$:     ADDL    #8,SP                      ;POP STATUS AND R3
                0781  1472          STATUS  NOPAREN                    ;SET INVALID SYNTAX
           05   0788  1473          RSB                                ;
```

```
                        0789            .SBTTL  PROCESS $EXPRESSION VALUE
                        0789    ;+
                        0789    ; DCL$EXPRESSION - PROCESS $EXPRESSION VALUE
                        0789    ;
                        0789    ; THIS ROUTINE IS CALLED TO PROCESS A $EXPRESSION VALUE AND EMIT A
                        0789    ; DESCRIPTOR TO THE RESULT PARSE TABLE.
                        0789    ;
                        0789    ; ***** NOT SUPPORTED FOR CLI$DCL_PARSE CALLS *****
                        0789    ;
                        0789    ; INPUTS:
                        0789    ;
                        0789    ;       R3 = CLASSIFICATION OF VALUE (PTR_K_QUALVALU OR PTR_K_PARAMETR)
                        0789    ;       R5 = -1 IF FIRST VALUE IN A PARENTHESIZED LIST
                        0789    ;       R8 = ADDRESS OF ENTITY BLOCK
                        0789    ;       R10 = BASE ADDRESS OF COMMAND WORK AREA.
                        0789    ;
                        0789    ; OUTPUTS:
                        0789    ;
                        0789    ;       R0 = STATUS
                        0789    ;       R1/R2 = DESCRIPTOR OF VALUE
                        0789    ;-
                        0789
                        0789    DCL$EXPRESSION:                                         ;PROCESS VALUE
                        0789    ;
                        0789    ; EVALUATE THE EXPRESSION.
                        0789    ;
          F874'   30    0789            BSBW    DCL$MOVCHAR                     ;COPY TERMINATOR INTO EXPANSION BUFF
          F871'   30    078C            BSBW    DCL$SETNBLK                     ;SKIP LEADING BLANKS
        F486 CA   DD    078F            PUSHL   WRK_L_EXPANDPTR(R10)            ;SAVE PRE-EVALUATION EXPANSION PTR
          F86A'   30    0793            BSBW    DCL$EXPRESS                     ;EVALUATE THE EXPRESSION
          2E 50   E9    0796            BLBC    R0,95$                          ;BRANCH IF ERROR
   F486 CA   8E   D0    0799            MOVL    (SP)+,WRK_L_EXPANDPTR(R10)      ;RESET EXPANSION PTR
                        079E
                        079E    ;
                        079E    ; CONVERT BINARY RESULTS TO ASCII.
                        079E    ;
             52   D5    079E            TSTL    R2                              ;BINARY RESULT?
             08   12    07A0            BNEQ    50$                             ;NO, THEN SKIP CONVERSION
          50 51   D0    07A2            MOVL    R1,R0                           ;GET THE VALUE TO CONVERT
          F858'   30    07A5            BSBW    DCL$CBTA_DEC                    ;CONVERT THE BINARY VALUE TO DECIMAL
             10   11    07A8            BRB     80$                             ;DONE
                        07AA
                        07AA    ;
                        07AA    ; COPY THE ASCII EXPRESSION RESULT OVER THE ORIGINAL EXPRESSION
                        07AA    ;
             51   DD    07AA    50$:    PUSHL   R1                              ;SAVE STRING LENGTH
F486 DA   62 51   28    07AC            MOVC    R1,(R2),@WRK_L_EXPANDPTR(R10)   ;COPY THE STRING
          51 8ED0       07B2            POPL    R1                              ;RESTORE STRING LENGTH
     52   F486 CA   D0  07B5            MOVL    WRK_L_EXPANDPTR(R10),R2         ;GET THE BUFFER ADDRESS
                        07BA
                        07BA    ;
                        07BA    ; CLEANUP AND RETURN.
                        07BA    ;
   F486 CA   51   C0    07BA    80$:    ADDL    R1,WRK_L_EXPANDPTR(R10)         ;UPDATE THE EXPANSION BUFFER PTR
                        07BF            STATUS  NORMAL                          ;SET SUCCESS
             05         07C6    90$:    RSB                                     ;
                        07C7    ;
```

```
              07C7   1532 ; EXPRESSION EVALUATION ERROR
              07C7   1533 ;
      8E   D5 07C7   1534 95$:    TSTL    (SP)+                         ;FIX STACK
           05 07C9   1535         RSB
```

```
                     07CA         .SBTTL  ISSUE PROMPT AND GET RESPONSE
                     07CA  ;---
                     07CA  ; ISSUE_PROMPT - ISSUE PROMPT AND GET RESPONSE
                     07CA  ;
                     07CA  ; THIS ROUTINE IS CALLED WHEN A REQUIRED PARAMETER IS MISSING
                     07CA  ; FROM A COMMAND.  THE USER IS PROMPTED FOR MORE INPUT.
                     07CA  ;
                     07CA  ; INPUTS:
                     07CA  ;
                     07CA  ;         R8 =    ADDRESS OF ENTITY BLOCK
                     07CA  ;         R10 =   ADDRESS OF COMMAND WORK AREA
                     07CA  ;         R11 =   ADDRESS OF PROCESS WORK AREA (FOR SUPERVISOR MODE PROMPTING)
                     07CA  ;
                     07CA  ;         WRK_B_PARMCNT = # PARAMETERS OBTAINED SO FAR
                     07CA  ;
                     07CA  ; OUTPUTS:
                     07CA  ;
                     07CA  ;         R0 = NEXT CHARACTER IN INPUT BUFFER
                     07CA  ;         R1 = RETURN STATUS
                     07CA  ;---
                     07CA
                     07CA  ISSUE_PROMPT:
                     07CA  ;
                     07CA  ; ALLOCATE SPACE FOR THE PARAMETER PROMPT ON THE STACK.
                     07CA  ;
7E   F99E CA    7D   07CA         MOVQ    WRK_W_PMPTLEN(R10),-(SP);SAVE PREVIOUS PROMPT STRING DESCRIPTOR
     5E   28    C2   07CF         SUBL    #ENT_K_MAX_PROMPT+8,SP  ;GET SIZE OF THE BIGGEST PROMPT STRING
                     07D2                                         ;PLUS EXTRA FOR COUNT,COLON,SPACE,ETC.
     53   5E    D0   07D2         MOVL    SP,R3                   ;GET ADDRESS OF STORAGE TO BUILD PROMPT STRI
                     07D5
                     07D5  ;
                     07D5  ; INITIALIZE THE PROMPT.
                     07D5  ;
          83    94   07D5         CLRB    (R3)+                   ;ZERO LENGTH OF PROMPT STRING
          0D    E0   07D7         BBS     #WRK_V_USRMODE,-        ;IF SET THEN USER PROMPT ROUTINE
   05 F0  AA         07D9                 WRK_W_FLAGS(R10),10$
83   00F1 CB    B0   07DC         MOVW    PRC_W_PMPTCTRL(R11),(R3)+ ;INSERT CR/LF PAIR
 83    5F 8F    90   07E1  10$:   MOVB    #^A7_7,(R3)+            ;SET CONTINUATION CHARACTER
                     07E5
                     07E5  ;
                     07E5  ; GET THE PROMPT STRING FROM THE TABLES
                     07E5  ;
  50   1A A8    32   07E5         CVTWL   ENT_W_PROMPT(R8),R0     ;GET PROMPT STRING OFFSET
     50   58    C0   07E9         ADDL    R8,R0                   ;GET PROMPT STRING ADDRESS
                     07EC
                     07EC  ;
                     07EC  ; INSERT THE PARAMETER PROMPT ON THE STACK
                     07EC  ;
     51   80    9A   07EC         MOVZBL  (R0)+,R1                ;LENGTH IN R1, ADDRESS IN R0
     83   80    90   07EF  40$:   MOVB    (R0)+,(R3)+             ;MOVE CHARACTER TO PROMPT STRING
       FA 51    F5   07F2         SOBGTR  R1,40$                  ;ANY MORE TO MOVE?
83   203A 8F    B0   07F5         MOVW    #^A/: /,(R3)+           ;INSERT SUFFIX CHARACTERS
51   53   5E    C3   07FA         SUBL3   SP,R3,R1                ;GET NUMBER OF CHARACTERS IN PROMPT
6E   51   01    83   07FE         SUBB3   #1,R1,(SP)              ;SET NEW LENGTH
                     0802
                     0802  ;
                     0802  ; DO EITHER SUPERVISOR MODE OR USER MODE PROMPT
```

```
                           0802  ;
              0D    E0     0802          BBS     #WRK_V_USRMODE,-          ;IF SET THEN USER PROMPT ROUTINE
        37 F0 AA           0804                  WRK_Q_FLAGS(R10),60$      ;
                           0807
                           0807  ;
                           0807  ; PROMPT FOR THE PARAMETER USING DCL'S PROMPT ROUTINE
                           0807  ;
    F99E CA   6E    9B     0807          MOVZBW  (SP),WRK_W_PMPTLEN(R10)   ;SET PROMPT STRING LENGTH
   F9A2 CA    01 AE 9E     080C          MOVAB   1(SP),WRK_L_PMPTADDR(R10) ;SET PROMPT STRING ADDRESS
                           0812          SETBIT  WRK_V_INQUIRE,-           ;MAKE IT LOOK LIKE INQUIRE
                           0812                  WRK_W_FLAGS(R10)          ; SO WE GET BACK CONTROL/Z'S (EOF)
                           0817          SETBIT  WRK_V_CONTIN,-            ;MAKE IT LOOK LIKE A CONTINUATION
                           0817                  WRK_W_FLAGS(R10)          ; SO THE RECALL BUFFER IS UPDATED
           F7E2'    30     081B          BSBW    DCL$FORNBLK               ;FORCE NONBLANK CHARACTER
              10    12     081E          BNEQ    50$                       ;BRANCH IF NON-NULL
      51   08 AB    D0     0820          MOVL    PRC_L_INPRAB(R11),R1      ;GET ADDRESS OF INPUT RAB
         0000'C1    B1     0824          CMPW    RAB$L_STS(R1),-           ;END OF FILE?
         0000'8F           0828                  #RMS$_EOF&^XFFFF          ;
              03    12     082B          BNEQ    50$                       ;BRANCH IF NOT
      50      1A    9A     082D          MOVZBL  #CTRLZ,R0                 ;YES, THEN SKIP THIS COMMAND
                           0830  50$:    CLRBIT  WRK_V_INQUIRE,-           ;CLEAR INQUIRE FLAG
                           0830                  WRK_W_FLAGS(R10)          ;
                           0835          CLRBIT  WRK_V_CONTIN,-            ;CLEAR CONTINUATION FLAG
                           0835                  WRK_W_FLAGS(R10)          ;
      51      01    D0     0839          MOVL    #1,R1                     ;SET NORMAL STATUS
              31    11     083C          BRB     90$                       ;GO CLEAN-UP
                           083E
                           083E  ;
                           083E  ; PROMPT FOR THE PARAMETER USING THE USER'S PROMPT ROUTINE
                           083E  ;
      51      5E    D0     083E  60$:    MOVL    SP,R1                     ;GET ADDRESS OF ASCIC PROMPT STRING
   7E      01 A1    9E     0841          MOVAB   1(R1),-(SP)               ;PUSH PROMPT STRING ADDRESS
      7E      61    9A     0845          MOVZBL  (R1),-(SP)                ;PUSH PROMPT STRING LENGTH
      51      5E    D0     0848          MOVL    SP,R1                     ;GET ADDR OF PROMPT DESCRIPTOR
   50    F9A6 CA    D0     084B          MOVL    WRK_L_PROMPTRTN(R10),R0   ;GET ADDRESS OF PROMPT ROUTINE
           F7AD'    30     0850          BSBW    DCL$USER_INPUT            ;CALL USER INPUT ROUTINE
      5E      08    C0     0853          ADDL    #8,SP                     ;POP THE PROMPT DESCRIPTOR
50    00000000'8F   D1     0856          CMPL    #RMS$_EOF,R0              ;END OF FILE?
              08    12     085D          BNEQ    70$                       ;NO, THEN CONTINUE
      51      01    D0     085F          MOVL    #1,R1                     ;SET SUCCESSFUL STATUS
      50      1A    D0     0862          MOVL    #CTRLZ,R0                 ;RETURN CTRL/Z AS FIRST CHARACTER
              08    11     0865          BRB     90$                       ;
              50    DD     0867  70$:    PUSHL   R0                        ;SAVE RETURN STATUS
           F794'    30     0869          BSBW    DCL$SETNBLK               ;GET FIRST CHARACTER
         51 8ED0           086C          POPL    R1                        ;RETURN STATUS IN R1
                           086F
                           086F  ;
                           086F  ; CLEAN UP AND RETURN
                           086F  ;
      5E      28    C0     086F  90$:    ADDL    #ENT_K_MAX_PROMPT+8,SP    ;DEALLOCATE PROMPT BUFFER STORAGE
    F99E CA   8E    7D     0872          MOVQ    (SP)+,WRK_Q_PMPTLEN(R10)  ;RESTORE PREVIOUS PROMPT ADDRESS
              05           0877          RSB                               ;AND RETURN WITH FIRST CHARACTER IN R0
                           0878
```

```
                        0878      .SBTTL  CHANGE COMMAND SYNTAX
                        0878 ;+
                        0878 ; DCL$CHANGE_SYNTAX - PROCESS A CHANGE LIST AND CHANGE THE COMMAND SYNTAX
                        0878 ;
                        0878 ; THIS ROUTINE IS CALLED TO REDEFINE THE SYTNAX OF THE COMMAND BEING PARSED
                        0878 ; ACCORDING TO THE DESCRIPTION IN THE CHANGE LIST.
                        0878 ;
                        0878 ; INPUTS:
                        0878 ;
                        0878 ;        R7 = QUALIFIER NUMBER
                        0878 ;        R8 = ADDRESS OF ENTITY DESCRIPTOR BLOCK
                        0878 ;        R10 = ADDRESS OF COMMAND WORK AREA
                        0878 ;
                        0878 ; OUTPUTS:
                        0878 ;
                        0878 ;        R0 = 0 IF NO NEW QUALIFIER LIST
                        0878 ;             1 IF NEW QUALIFIER LIST
                        0878 ;
                        0878 ;        APPROPRIATE WRK BLOCK FIELDS ARE MODIFIED TO REFLECT THE
                        0878 ;        NEW COMMAND SYNTAX.
                        0878 ;
                        0878 ;                WRK_L_IMAGE -   NEW IMAGE ADDRESS
                        0878 ;                WRK_B_MINPARM - MINIMUM PARAMETER COUNT
                        0878 ;                WRK_B_MAXPARM - MAXIMUM PARAMETER COUNT
                        0878 ;                WRK_L_PROPTR -  PARAMETER LIST
                        0878 ;                WRK_L_QUABLK -  QUALIFIER LIST
                        0878 ;                WRK_G_RESULT -  TOKEN ARRAY
                        0878 ;
                        0878 ;-
                        0878
                        0878 DCL$CHANGE_SYNTAX:
              56    DD  0878          PUSHL   R6                                  ;SAVE REGISTERS
              7E    D4  087A          CLRL    -(SP)                               ;INIT PARAM/QUAL LIST STATUS
         0040 8F    A8  087C          BISW    #WRK_M_VERB,-                       ;SET VERB PROCESSING FLAG ONCE AGAIN
           F0 AA        0880                  WRK_W_FLAGS(R10)                    ;
           0C A8    C1  0882          ADDL3   ENT_L_SYNTAX(R8),-                  ;GET ADDRESS OF CHANGE LIST
    56     DE AA        0885                  WRK_L_TAB_VEC(R10),R6
           C3 AA    95  0888          TSTB    WRK_B_CMDOPT(R10)                   ;FIRST SYNTAX CHANGE SO FAR?
              04    12  088B          BNEQ    10$                                 ;ONLY RECORD THE FIRST CHANGE MADE
    C3 AA     57    90  088D          MOVB    R7,WRK_B_CMDOPT(R10)                ;SAVE QUAL # INDUCING THE FIRST CHAN
                        0891                                                      ; FOR GETOPT CALLBACK (SET/SHOW)
                        0891
                        0891 ;
                        0891 ; CHANGE IMAGE TO BE EXECUTED
                        0891 ;        IF USER ROUTINE - GET ROUTINE ADDRESS.
                        0891 ;        IF CLI ROUTINE OR IMAGE - GET ADDRESS OF ASCIC ROUTINE NAME OR FILE SPEC.
                        0891 ;
    14 A6     04    91  0891 10$:     CMPB    #CMD_K_SAME,CMD_B_HANDLER(R6)       ;IMAGE CHANGE REQUESTED?
              26    13  0895          BEQL    40$                                 ;BRANCH IF NOT
              03    AA  0897          BICW    #WRK_M_CLIRTN!WRK_M_USERRTN,-       ;CLEAR PREVIOUS IMAGE STATE
           F2 AA        0899                  WRK_W_FLAGS2(R10)                   ;
    50     1A A6    32  089B          CVTWL   CMD_W_IMAGE(R6),R0                  ;GET BRO TO IMAGE/ROUTINE INFO
       50     56    C0  089F          ADDL    R6,R0                               ;COMPUTE REAL ADDRESS
    14 A6     01    91  08A2          CMPB    #CMD_K_CLI,CMD_B_HANDLER(R6)        ;ARE WE DEALING WITH A CLI ROUTINE?
              04    12  08A6          BNEQ    15$                                 ;NO, THEN SKIP
                        08A8          SETBIT  WRK_V_CLIRTN,WRK_W_FLAGS2(R10)      ;SET CLI ROUTINE FLAG
    14 A6     02    91  08AC 15$:     CMPB    #CMD_K_USER,CMD_B_HANDLER(R6)       ;ARE WE DEALING WITH A USER ROUTINE?
```

```
            07   12  08B0              BNEQ    20$                              ;NO, THEN SKIP
                     08B2              SETBIT  WRK_V_USERRTN,WRK_W_FLAGS2(R10)  ;SET USER ROUTINE FLAG
      50    60   D0  08B6              MOVL    (R0),R0                          ;GET ROUTINE ADDRESS
   E2 AA    50   D0  08B9      20$:    MOVL    R0,WRK_L_IMAGE(R10)              ;SAVE IMAGE/ROUTINE INFO
                     08BD
                     08BD      ;
                     08BD      ; CLEAR/SET NOSTATUS BIT.
                     08BD      ;
                     08BD      40$:    CLRBIT  WRK_V_NOSTAT,WRK_W_FLAGS(R10)    ;CLEAR STATUS FLAG
            01   E1  08C2              BBC     #CMD_V_NOSTAT,-                  ;BRANCH IF STATUS SHOULD BE SET
      05 04 A6       08C4                      CMD_Q_FLAGS(R6),50$              ;
                     08C7              SETBIT  WRK_V_NOSTAT,WRK_W_FLAGS(R10)    ;SET STATUS FLAG
                     08CC
                     08CC      ;
                     08CC      ; CHANGE PARAMETER DEFINITIONS
                     08CC      ;
            05   E1  08CC      50$:    BBC     #CMD_V_PARMS,-                   ;BRANCH IF NO PARAMETER CHANGE
      22 04 A6       08CE                      CMD_Q_FLAGS(R6),60$              ;
            00   EF  08D1              EXTZV   #CMD_V_MINPARM,-                 ;EXTRACT MINIMUM PARAMETERS
            04       08D3                      #CMD_S_MINPARM,-                 ;
   50    15 A6       08D4                      CMD_B_PARMCNT(R6),R0             ;
   D1 AA    50   90  08D7              MOVB    R0,WRK_B_MINPARM(R10)
            04   EF  08DB              EXTZV   #CMD_V_MAXPARM,-                 ;EXTRACT MAXIMUM PARAMETERS
            04       08DD                      #CMD_S_MAXPARM,-                 ;
   50    15 A6       08DE                      CMD_B_PARMCNT(R6),R0             ;
   D0 AA    50   90  08E1              MOVB    R0,WRK_B_MAXPARM(R10)
   50    08 A6   D0  08E5              MOVL    CMD_L_PARMS(R6),R0               ;GET OFFSET TO NEW PARAMETER LIST
            04   13  08E9              BEQL    55$                              ;IF NONE, THEN NO MORE PARAMS ALLOWED
   50    DE AA   C0  08EB              ADDL    WRK_L_TAB_VEC(R10),R0            ;GET ADDRESS OF PARAMETER LIST
   C6 AA    50   D0  08EF      55$:    MOVL    R0,WRK_L_PROPTR(R10)             ;RESET ADDRESS OF PARAMETER LIST
                     08F3
                     08F3      ;
                     08F3      ; CHANGE QUALIFIER DEFINITIONS
                     08F3      ;
            06   E1  08F3      60$:    BBC     #CMD_V_QUALS,-                   ;BRANCH IF NO QUAL CHANGE
      42 04 A6       08F5                      CMD_Q_FLAGS(R6),80$              ;
   50    0C A6   D0  08F8              MOVL    CMD_L_QUALS(R6),R0               ;GET OFFSET TO NEW QUALIFIER LIST
            04   13  08FC              BEQL    65$                              ;IF NONE, THEN NO MORE QUALS ALLOWED
   50    DE AA   C0  08FE              ADDL    WRK_L_TAB_VEC(R10),R0            ;GET ADDRESS OF QUALIFIER LIST
   CA AA    50   D0  0902      65$:    MOVL    R0,WRK_L_QUABLK(R10)             ;RESET ADDRESS OF QUALIFIER LIST
                     0906
                     0906      ;
                     0906      ; INVALIDATE ALL QUALIFIERS IN THE TOKEN ARRAY, INCLUDING THE QUALIFIER WE
                     0906      ; MAY BE CURRENTLY PROCESSING, SO THAT ALL QUALIFIERS IN THE PREVIOUS SYNTAX
                     0906      ; ARE NO LONGER ACCESSIBLE.  THIS IS DONE TO PREVENT PREVIOUS QUALIFIERS
                     0906      ; FROM BEING CONFUSED WITH THE CURRENT QUALIFIER NUMBERING SCHEME.  THE
                     0906      ; INVALIDATION IS DONE BY ZEROING THE QUALIFIER NUMBER IN THE TOKEN ARRAY,
                     0906      ; ESSENTIALLY TO RECORD THAT A QUALIFIER WAS DETECTED IN THAT POSITION, BUT
                     0906      ; THAT IT DOESN'T REPRESENT ANY QUALIFIER IN THE CURRENT QUALIFIFER LIST.
                     0906      ;
50     F9AA CA   9E  0906              MOVAB   WRK_G_RESULT-PTR_C_LENGTH(R10),R0 ;GET ADDRESS OF TOKEN ARRAY
      50    0C   C0  090B      70$:    ADDL    #PTR_C_LENGTH,R0                 ;SKIP TO NEXT ENTRY
   BA AA    50   D1  090E              CMPL    R0,WRK_L_RSLNXT(R10)             ;REACHED LAST ENTRY STORED?
            18   1E  0912              BGEQU   75$                              ;BRANCH IF SO
                     0914              ASSUME  PTR_K_COMDQUAL EQ 0              ;
                     0914              ASSUME  PTR_K_PARMQUAL EQ 1              ;
                     0914              ASSUME  PTR_K_QUALVALU EQ 2              ;
```

```
               1C    ED  0914          CMPZV   #PTR_V_TYPE,-                  ;QUALIFIER?
        60     04        0916                  #PTR_S_TYPE,(R0),-             ;
               02        0918                  #PTR_K_QUALVALU                ;
               F0    1A  0919          BGTRU   70$                            ;SKIP IF NOT
               16    E0  091B          BBS     #PTR_V_SYNTAX,-                ;SKIP IF QUALI. CHANGED SYNTAX
           03  60        091D                  (R0),72$
        6E     01    D0  091F          MOVL    #1,(SP)                        ;SET BIT TO INVALIDATE THIS QUALIFIER
           08  A0    D4  0922  72$:    CLRL    PTR_L_ENTITY(R0)               ;ZERO ENTITY BLOCK ADDRESS (INVALIDATE IT)
        1C     05    F0  0925          INSV    #PTR_K_IGNORE,#PTR_V_TYPE,-       ;SET IGNORE FLAG IN TYPE FIELD
        60     04        0928                  #PTR_S_TYPE,(R0)               ;
               DF    11  092A          BRB     70$                            ;LOOP UNTIL ALL QUALIFIERS INVALIDATED
     0A 6E     00    E3  092C  75$:    BBCS    #0,(SP),80$                    ;BRANCH IF NO QUALIFIERS SEEN
                         0930          STATUS  IGNQUAL                        ;SET THE IGNORED QUALIFIER STATUS
        D6 BA        16  0937          JSB     @WRK_L_SIGNALRTN(R10)          ;SIGNAL THE INFORMATIONAL MESSAGE
                         093A
                         093A  ;
                         093A  ; GET NEW DISALLOW EXPRESSION TREE.
                         093A  ;
               07    E1  093A  80$:    BBC     #CMD_V_DISALLOWS,-             ;BRANCH IF NO DISALLOW CHANGE
     0E 04     A6        093C                  CMD_Q_FLAGS(R6),90$            ;
     50    10  A6    D0  093F          MOVL    CMD_L_DISALLOW(R6),R0          ;GET OFFSET TO DISALLOW EXPRESSION TREE
               04    13  0943          BEQL    85$                            ;IF EQL NONE
     50    DE  AA    C0  0945          ADDL    WRK_L_TAB_VEC(R10),R0          ;CALCULATE ACTUAL ADDRESS
     E6 AA     50    D0  0949  85$:    MOVL    R0,WRK_L_DISALLOW(R10)         ;SAVE ADDRESS OF DISALLOW EXPRESSION TREE
                         094D
                         094D  ;
                         094D  ; RESTORE REGISTERS AND RETURN
                         094D  ;
           50 8ED0       094D  90$:    POPL    R0                             ;SET QUAL/PARAM LIST STATUS
           56 8ED0       0950          POPL    R6                             ;RESTORE REGISTERS
              05         0953          RSB                                    ;RETURN FROM SUBROUTINE
```

```
                         0954          .SBTTL  SEARCH VERB TABLE
                         0954  ;---
                         0954  ; DCL$SEARCH_VERB - SEARCH VERB TABLE AND OBTAIN VERB INFORMATION
                         0954  ;
                         0954  ; THIS ROUTINE SEARCHES THE VERB TABLE FOR A SPECIFIED STRING.
                         0954  ; IF FOUND, THE VERB ATTRIBUTES ARE STORED IN THE COMMAND WORK AREA.
                         0954  ;
                         0954  ; INPUTS:
                         0954  ;
                         0954  ;       R8 = ADDRESS OF COMMAND TABLES
                         0954  ;       R1 = LENGTH OF VERB TOKEN
                         0954  ;       R2 = ADDRESS OF VERB TOKEN
                         0954  ;
                         0954  ; OUTPUTS:
                         0954  ;
                         0954  ;       R0 = STATUS CODE
                         0954  ;       R8 = ADDRESS OF CMD BLOCK
                         0954  ;       THE COMMAND WORK AREA IS INITIALIZED IF VERB FOUND
                         0954  ;---
                         0954
                         0954  DCL$SEARCH_VERB::
          02C0 8F    BB  0954          PUSHR   #^M<R6,R7,R9>                   ;SAVE REGISTERS
                         0958
                         0958  ;
                         0958  ; TRIM VERB TO FOUR CHARACTERS.
                         0958  ;
          54   51    7D  0958          MOVQ    R1,R4                           ;SAVE COMMAND VERB DESCRIPTOR
          54   04    D1  095B          CMPL    #4,R4                           ;COMMAND VERB LESS THAN 5 CHARACTERS
               03    18  095E          BGEQ    20$                             ;IF GEQ, USE ORIGINAL LENGTH
          54   04    D0  0960          MOVL    #4,R4                           ;SET TO SCAN ONLY 4 CHARACTERS
                         0963
                         0963  ;
                         0963  ; FIND BASE AND SIZE OF VERB AND POINTER TABLES.
                         0963  ;
  57   58   0C A8    C1  0963  20$:    ADDL3   VEC_L_COMDPTR(R8),R8,R7         ;GET BASE OF POINTER TABLE
          57   08    C0  0968          ADDL    #VEC_K_HEADER_LENGTH,R7         ;SKIP PAST POINTER TABLE HEADER
                         096B
  53   58   08 A8    C1  096B          ADDL3   VEC_L_VERBTBL(R8),R8,R3         ;GET BASE OF VERB TABLE
          52   63    32  0970          CVTWL   VEC_W_SIZE(R3),R2               ;GET SIZE OF TABLE + HEADER
          52   08    C2  0973          SUBL    #VEC_K_HEADER_LENGTH,R2         ;REMOVE HEADER
     56   52   04    C7  0976          DIVL3   #4,R2,R6                        ;GET SIZE IN VERBS (4 BYTES PER VERB
          53   08    C0  097A          ADDL    #VEC_K_HEADER_LENGTH,R3         ;SKIP PAST VERB TABLE HEADER
                         097D
                         097D  ;
                         097D  ; SEARCH VERB TABLE FOR SPECIFIED VERB.
                         097D  ;
63   52   65   54    39  097D  22$:    MATCHC  R4,(R5),R2,(R3)                 ;SCAN FOR VERB MATCH
               0A    13  0982          BEQL    25$                             ;BR IF VERB MATCH
                         0984          STATUS  IVVERB                          ;INVALID VERB
             00F0    31  098B          BRW     90$                             ;
                         098E
                         098E  ;
                         098E  ; IF MATCH WAS NOT ON LONGWORD BOUNDARY, THEN KEEP LOOKING.
                         098E  ;
     50   53   54    C3  098E  25$:    SUBL3   R4,R3,R0                        ;GET ADDRESS OF MATCHED VERB
     51   50   03    CB  0992          BICL3   #3,R0,R1                        ;ROUND ADDR TO LONGWORD ADDR
          51   50    D1  0996          CMPL    R0,R1                           ;IS IT SAME AS ORIGINAL?
```

```
                  0C  13 0999            BEQL    27$                          ;YES, THEN DONE
         50   54  01  C3 099B            SUBL3   #1,R4,R0                     ;START SEARCH ONE CHAR AFTER
              53  50  C2 099F            SUBL    R0,R3                        ; FIRST CHAR OF MATCH
              52  50  C0 09A2            ADDL    R0,R2                        ;
                  D6  11 09A5            BRB     22$                          ;KEEP LOOKING
                         09A7
                         09A7      ;
                         09A7      ; VERB MATCH FOUND - GET ADDRESS OF CMD BLOCK.
                         09A7      ;
           DE AA  58  D0 09A7      27$:  MOVL    R8,WRK_L_TAB_VEC(R10)        ;SAVE ADDRESS OF TABLES VECTOR
         59   52  04  C7 09AB            DIVL3   #4,R2,R9                     ;CALCULATE INVERSE COMMAND INDEX
         59   56  59  A3 09AF            SUBW3   R9,R6,R9                     ;CALCULATE VERB INDEX
              59  01  A2 09B3            SUBW    #1,R9                        ;ZERO BASE THE RESULT
    59  6749   DE AA  C1 09B6            ADDL3   WRK_L_TAB_VEC(R10),(R7)[R9],R9 ;GET ADDRESS OF COMMAND BLOCK
                         09BC
                         09BC      ;
                         09BC      ; CHECK FOR AMBIGUOUS VERBS.
                         09BC      ;
                  00  E0 09BC            BBS     #CMD_V_ABBREV,-              ;IF SET, NON UNIQUE ACCEPTABLE
            1C 04 A9     09BE                    CMD_B_FLAGS(R9),30$          ;
              53  54  C2 09C1            SUBL    R4,R3                        ;BACKUP TO START OF VERB ENTRY FOUND
  63   04   00  65  54  2D 09C4          CMPC5   R4,(R5),#0,#4,(R3)           ;EXACT MATCH?
                  11  13 09CA            BEQL    30$                          ;IF EXACT MATCH, IGNORE AMBIGUITY
       63   04  65  54  39 09CC          MATCHC  R4,(R5),#4,(R3)              ;SCAN FOR DUPLICATE MATCH
                  0A  12 09D1            BNEQ    30$                          ;IF NEQ UNIQUE MATCH
                         09D3            STATUS  ABVERB                       ;AMBIGUOUS COMMAND VERB
                00A1  31 09DA            BRW     90$                          ;
                         09DD
                         09DD      ;
                         09DD      ; VERB MATCH FOUND - EXTRACT ATTRIBUTES FROM COMMAND BLOCK
                         09DD      ;
            16 A9  90    09DD      30$:  MOVB    CMD_B_VERBTYP(R9),-          ;SET VERB TYPE FOR OLD INTERFACE
            C2 AA        09E0                    WRK_B_VERBTYP(R10)           ;
                         09E2
         50  18 A9  32   09E2            CVTWL   CMD_W_NAME(R9),R0            ;SAVE VERB NAME
            50  59  C0   09E6            ADDL    R9,R0                        ;
      BE AA  01 A0  9E   09E9            MOVAB   1(R0),WRK_L_VERB(R10)        ;SKIP PAST FIRST ASCIC COUNT
                         09EE
                00  EF   09EE            EXTZV   #CMD_V_MINPARM,-             ;EXTRA   MINIMUM NUMBER OF PARAMETER
                04       09F0                    #CMD_S_MINPARM,-             ;
         50  15 A9       09F1                    CMD_B_PARMCNT(R9),R0         ;
      D1 AA  50  90      09F4            MOVB    R0,WRK_B_MINPARM(R10)        ;
                04  EF   09F8            EXTZV   #CMD_V_MAXPARM,-             ;EXTRACT MAXIMUM NUMBER OF PARAMETER
                04       09FA                    #CMD_S_MAXPARM,-             ;
         50  15 A9       09FB                    CMD_B_PARMCNT(R9),R0         ;
      D0 AA  50  90      09FE            MOVB    R0,WRK_B_MAXPARM(R10)        ;
                         0A02
         50  0C A9  D0   0A02            MOVL    CMD_L_QUALS(R9),R0           ;GET OFFSET TO FIRST QUALIFIER DESC
                03  13   0A06            BEQL    40$                          ;IF EQL NONE
            50  58  C0   0A08            ADDL    R8,R0                        ;CALCULATE ACTUAL ADDRESS
      CA AA  50  D0      0A0B      40$:  MOVL    R0,WRK_L_QUABLK(R10)         ;SET ADDRESS OF QUALIFIER DESCRIPTOR
                         0A0F
         50  08 A9  D0   0A0F            MOVL    CMD_L_PARMS(R9),R0           ;GET OFFSET TO FIRST POSITIONAL ENTI
                03  13   0A13            BEQL    50$                          ;IF EQL NONE
            50  58  C0   0A15            ADDL    R8,R0                        ;CALCULATE ACTUAL ADDRESS
      C6 AA  50  D0      0A18      50$:  MOVL    R0,WRK_L_PROPTR(R10)         ;SAVE ADDRESS OF PARAMETER DESCRIPTO
                         0A1C
```

```
50    10 A9   D0  0A1C            MOVL    CMD_L_DISALLOW(R9),R0             ;GET OFFSET TO DISALLOW EXPRESSION T
         03   13  0A20            BEQL    55$                               ;IF EQL NONE
   50    58   C0  0A22            ADDL    R8,R0                             ;CALCULATE ACTUAL ADDRESS
E6 AA    50   D0  0A25  55$:      MOVL    R0,WRK_L_DISALLOW(R10)            ;SAVE ADDRESS OF DISALLOW EXPRESSION
                  0A29
50    1C A9   32  0A29            CVTWL   CMD_W_OUTPUTS(R9),R0              ;GET OFFSET TO OUTPUT LIST
         03   13  0A2D            BEQL    60$                               ;IF EQL NONE
   50    59   C0  0A2F            ADDL    R9,R0                             ;CALCULATE ACTUAL ADDRESS
D2 AA    50   D0  0A32  60$:      MOVL    R0,WRK_L_PAROUT(R10)              ;SAVE ADDRESS OF OUTPUT LIST
                  0A36
      CE AA   94  0A36            CLRB    WRK_B_PARMCNT(R10)                ;CLEAR COUNTS OF PARAMETERS
      CF AA   94  0A39            CLRB    WRK_B_PARMSUM(R10)                ;
      C3 AA   94  0A3C            CLRB    WRK_B_CMDOPT(R10)                 ;CLEAR ENTITY # CAUSING SYNTAX CHANG
                  0A3F            CLRBIT  WRK_V_NOSTAT,WRK_W_FLAGS(R10)     ;CLEAR STATUS FLAG
         01   E1  0A44            BBC     #CMD_V_NOSTAT,-                   ;BRANCH IF STATUS SHOULD BE SET
   05 04 A9       0A46                    CMD_W_FLAGS(R9),62$               ;
                  0A49            SETBIT  WRK_V_NOSTAT,WRK_W_FLAGS(R10)     ;SET STATUS FLAG
                  0A4E
                  0A4E  ;
                  0A4E  ; IF USER ROUTINE - GET ROUTINE ADDRESS.
                  0A4E  ; IF CLI ROUTINE OR IMAGE - GET ADDRESS OF ASCIC ROUTINE NAME OR FILE SPEC.
                  0A4E  ;
         03   AA  0A4E  62$:      BICW    #WRK_M_CLIRTN!WRK_M_USERRTN,-     ;CLEAR PREVIOUS IMAGE STATE
      F2 AA       0A50                    WRK_W_FLAGS2(R10)                 ;
50    1A A9   32  0A52            CVTWL   CMD_W_IMAGE(R9),R0                ;GET BRO TO IMAGE/ROUTINE INFO
   50    59   C0  0A56            ADDL    R9,R0                             ;COMPUTE REAL ADDRESS
14 A9    01   91  0A59            CMPB    #CMD_K_CLI,CMD_B_HANDLER(R9)      ;ARE WE DEALING WITH A CLI ROUTINE?
         04   12  0A5D            BNEQ    65$                               ;NO, THEN SKIP
                  0A5F            SETBIT  WRK_V_CLIRTN,WRK_W_FLAGS2(R10)    ;SET CLI ROUTINE FLAG
14 A9    02   91  0A63  65$:      CMPB    #CMD_K_USER,CMD_B_HANDLER(R9)     ;ARE WE DEALING WITH A USER ROUTINE?
         07   12  0A67            BNEQ    70$                               ;NO, THEN SKIP
                  0A69            SETBIT  WRK_V_USERRTN,WRK_W_FLAGS2(R10)   ;SET USER ROUTINE FLAG
   50    60   D0  0A6D            MOVL    (R0),R0                           ;GET ROUTINE ADDRESS
E2 AA    50   D0  0A70  70$:      MOVL    R0,WRK_L_IMAGE(R10)               ;SAVE IMAGE/ROUTINE INFO
                  0A74
                  0A74  ;
                  0A74  ; RESTORE STATE AND RETURN SUCCESS.
                  0A74  ;
                  0A74            STATUS  NORMAL                            ;SET SUCCESSFUL STATUS
   58    59   D0  0A7B            MOVL    R9,R8                             ;RETURN CMD BLOCK ADDRESS
   02C0 8F    BA  0A7E  90$:      POPR    #^M<R6,R7,R9>                     ;RESTORE REGISTERS
         05       0A82            RSB
                  0A83
                  0A83            .END
```

```
BLOCK_K_TYPE                  = 00000003
CLI$_ABKEYW                   = 00038010
CLI$_ABVERB                   = 00038008
CLI$_BUFOVF                   = 00038018
CLI$_DIRECT                   = 00038030
CLI$_IGNQUAL                  = 0003DE03
CLI$_INSFPRM                  = 00038048
CLI$_IVATIME                  = 00038290
CLI$_IVDTIME                  = 00038298
CLI$_IVKEYW                   = 00038060
CLI$_IVQLOC                   = 00038078
CLI$_IVQUAL                   = 00038240
CLI$_IVVERB                   = 00038090
CLI$_MAXPARM                  = 00038098
CLI$_NOCCAT                   = 000380A8
CLI$_NOCOMD                   = 000380B0
CLI$_NOKEYW                   = 000380B8
CLI$_NOLIST                   = 000380C0
CLI$_NOPAREN                  = 00038288
CLI$_NOQUAL                   = 000380C8
CLI$_NORMAL                   = 00030001
CLI$_NOTNEG                   = 000380D8
CLI$_NOVALU                   = 000380D0
CLI$_NULFIL                   = 000380E0
CLI$_NUMBER                   = 000380E8
CLI$_ONEVAL                   = 00038158
CLI$_PARMDEL                  = 00038110
CLI$_VALREQ                   = 00038150
CMD_B_HANDLER                 = 000C0014
CMD_B_PARMCNT                 = 00000015
CMD_B_VERBTYP                 = 00000016
CMD_K_CLI                     = 00000001
CMD_K_SAME                    = 00000004
CMD_K_USER                    = 00000002
CMD_L_DISALLOW                = 00000010
CMD_L_PARMS                   = 00000008
CMD_L_QUALS                   = 0000000C
CMD_S_MAXPARM                 = 00000004
CMD_S_MINPARM                 = 00000004
CMD_V_ABBREV                  = 00000000
CMD_V_DISALLOWS               = 00000007
CMD_V_MAXPARM                 = 00000004
CMD_V_MINPARM                 = 00000000
CMD_V_NOSTAT                  = 00000001
CMD_V_PARMS                   = 00000005
CMD_V_QUALS                   = 00000006
CMD_W_FLAGS                   = 00000004
CMD_W_IMAGE                   = 0000001A
CMD_W_NAME                    = 00000018
CMD_W_OUTPUTS                 = 0000001C
CTRLZ                         = 0000001A
DCL$ACL                         00000716 R     02
DCL$BACKUPCHAR                  ********   X   02
DCL$BACKUPMOVE                  ********   X   02
DCL$CBTA_DEC                    ********   X   02
DCL$CHANGE_SYNTAX               00000878 R     02
DCL$CHECK_VALUES                000002B4 R     02
DCL$CNVNOEDIT                   ********   X   02
DCL$COMPRESS                    ********   X   02
DCL$DATETIME                    00000609 R     02
DCL$EVAL_DISALLOW               ********   X   02
DCL$EXPRESS                     ********   X   02
DCL$EXPRESSION                  00000789 R     02
DCL$FIND_KEYWORD                00000204 R     02
DCL$FORNBLK                     ********   X   02
DCL$GENDESCR                    ********   X   02
DCL$GENEOL                      0000013D RG    02
DCL$GENTERM                     ********   X   02
DCL$GETOKEN                     ********   X   02
DCL$GTBTOKEN                    ********   X   02
DCL$MARK                        ********   X   02
DCL$MARKEDTOKEN                 ********   X   02
DCL$MOVBTOKN                    ********   X   02
DCL$MOVCHAR                     ********   X   02
DCL$MOVTOKN                     ********   X   02
DCL$NUMBER                      000005D4 R     02
DCL$PAREN_VALUE                 000006A3 R     02
DCL$PARSE_COMMAND               00000000 RG    02
DCL$PARSE_VALUE                 0000037A R     02
DCL$PROCESS_KEYWORD             00000470 R     02
DCL$PROCFILE                    000004B6 RG    02
DCL$PROCFILE1                   000004C0 R     02
DCL$PROCQUAL                    00000157 RG    02
DCL$REST_OF_LINE                00000593 R     02
DCL$SCOPY_DXDX                  0000067F RG    02
DCL$SEARCH_VERB                 00000954 RG    02
DCL$SETCHAR                     ********   X   02
DCL$SETNBLK                     ********   X   02
DCL$TESTBLANK                   ********   X   02
DCL$USER_INPUT                  ********   X   02
END_OF_LINE                     000000B1 R     02
ENT_B_TYPE                    = 00000002
ENT_B_VALTYPE                 = 00000015
ENT_K_ACL                     = 00000019
ENT_K_DATETIME                = 00000005
ENT_K_DELTATIME               = 00000012
ENT_K_DEVICE                  = 0000000D
ENT_K_DIR                     = 0000000E
ENT_K_EXPRESSION              = 00000015
ENT_K_FILE                    = 00000014
ENT_K_INFILE                  = 00000001
ENT_K_MAX_PROMPT              = 00000020
ENT_K_NODE                    = 0000000C
ENT_K_NUMBER                  = 00000003
ENT_K_OUTFILE                 = 00000002
ENT_K_PARENVALUE              = 00000011
ENT_K_QUALIFIER               = 00000002
ENT_K_QUOTEDSTRING            = 00000013
ENT_K_RESTOFLINE              = 00000010
ENT_K_UIC                     = 0000000F
ENT_L_NEXT                    = 00000008
ENT_L_SYNTAX                  = 0000000C
ENT_L_USER_TYPE               = 00000010
ENT_V_CONCAT                  = 00000006
```

```
ENT_V_LIST          = 00000005
ENT_V_NEG           = 00000001
ENT_V_PARM          = 00000009
ENT_V_VAL           = 00000000
ENT_V_VALREQ        = 00000004
ENT_V_VERB          = 00000008
ENT_W_FLAGS         = 00000004
ENT_W_NAME          = 00000016
ENT_W_PROMPT        = 0000001A
EXIT                  00000134 R     02
ISSUE_PROMPT          000007CA R     02
LIB$CVT_DTIME         ********   X   02
LIB$CVT_TIME          ********   X   02
M_AMBIG             = 00000008
M_KEYWD             = 00000002
M_NEGAT             = 00000001
M_SYNTAX            = 00000004
NORMAL_EXIT           00000137 R     02
NO_COMMAND            0000012D R     02
PARSE_PARMS           0000001A R     02
PARSE_QUAL            00000006 R     02
PRC_B_CONTINUE        000000F3
PRC_B_DEFRADIX        000000AE
PRC_B_EXMDEPMOD       000000AD
PRC_B_EXMDEPWID       000000AC
PRC_B_EXONLYL         0000012D
PRC_B_FLAGS2          000000AF
PRC_B_IMGFLAG         00000078
PRC_B_OUTFLAGS        0000012C
PRC_B_PROMPTLEN       000000F0
PRC_C_LENGTH          00000534
PRC_G_COMMANDS        00000133
PRC_G_PROMPT          000000F4
PRC_K_LENGTH          00000534
PRC_L_CURRKEY         00000048
PRC_L_EXMDEPADR       000000A8
PRC_L_EXTARG          00000094
PRC_L_EXTBLK          0000008C
PRC_L_EXTCOD          0000009C
PRC_L_EXTHND          00000090
PRC_L_EXTPRM          00000098
PRC_L_IDFLNK          000000BC
PRC_L_IMGACTSTS       00000080
PRC_L_INDCLOCK        0000007C
PRC_L_INDEPTH         0000005C
PRC_L_INDFAB          0000001C
PRC_L_INDINPRAB       00000014
PRC_L_INDOUTRAB       00000018
PRC_L_INPRAB          00000008
PRC_L_LASTKEY         0000004C
PRC_L_LSTSTATUS       000000B0
PRC_L_ONCTLY          000000B8
PRC_L_ONERROR         0000006C
PRC_L_OUTOFBAND       000000B4
PRC_L_OUTRAB          0000000C
PRC_L_OUTRABCTX       00000118
PRC_L_PPFLIST         00000070
PRC_L_RECALLPTR       0000012F
PRC_L_RESTART         00000058
PRC_L_SAVAP           00000000
PRC_L_SAVFP           00000004
PRC_L_SEVERITY        00000050
PRC_L_SPWN            000000C0
PRC_L_STACKLM         000000A4
PRC_L_STACKPT         000000A0
PRC_L_STATUS          00000054
PRC_L_STS             00000084
PRC_L_STV             00000088
PRC_L_SYMBOL          00000060
PRC_L_TMBX            00000074
PRC_L_TRMLIST         00000010
PRC_Q_ALLOCREG        00000020
PRC_Q_COMMAND         000000E0
PRC_Q_FLUSHTIME       000000D0
PRC_Q_GLOBAL          00000028
PRC_Q_IMAGENAME       000000D8
PRC_Q_KEYPAD          00000040
PRC_Q_LABEL           00000030
PRC_Q_LOCAL           00000038
PRC_Q_SAVEPRIV        000000E8
PRC_T_OUTDVI          0000011C
PRC_V_IND           = 00000005
PRC_V_MODE          = 00000006
PRC_V_YLEVEL        = 0000000B
PRC_W_ASTIOSB         000000C6
PRC_W_ASTRETN         000000C8
PRC_W_ASTSTATUS       000000C4
PRC_W_ATTMBX          0000007A
PRC_W_FLAGS           00000068
PRC_W_INPCHAN         00000064
PRC_W_ONLEVEL         0000006A
PRC_W_OUTIFI          00000114
PRC_W_OUTISI          00000116
PRC_W_OUTMBXCHN       000000CA
PRC_W_OUTMBXREF       000000CE
PRC_W_OUTMBXSIZ       000000CC
PRC_W_PMPTCTRL        000000F1
PRC_W_WAITIOSB        00000066
PROMPT                000000EF R     02
PTR_B_LEVEL           00000004
PTR_B_NUMBER          00000005
PTR_B_PARMCNT         00000006
PTR_B_VALUE           00000000
PTR_C_LENGTH          0000000C
PTR_K_COMDQUAL      = 00000000
PTR_K_ENDLINE       = 00000004
PTR_K_IGNORE        = 00000005
PTR_K_LENGTH          0000000C
PTR_K_PARAMETR      = 00000003
PTR_K_PARMQUAL      = 00000001
PTR_K_QUALVALU      = 00000002
PTR_L_DESCR           00000000
PTR_L_ENTITY          00000008
PTR_S_TYPE          = 00000004
```

```
PTR_V_KEYWORD                    = 00000015               WRK_V_CONTIN                     = 00000003
PTR_V_NEGATE                     = 00000014               WRK_V_INQUIRE                    = 00000007
PTR_V_SYNTAX                     = 00000016               WRK_V_NOSTAT                     = 00000008
PTR_V_TYPE                       = 0000001C               WRK_V_PROMPT                     = 00000002
RAB$L_STS                          ********   X    02     WRK_V_USERRTN                    = 00000001
RMS$_EOF                           ********   X    02     WRK_V_USRMODE                    = 0000000D
TRMB                               00000373 R      02     WRK_V_VERB                       = 00000006
TRME                               0000037A R      02     WRK_W_FLAGS                        FFFFFFF0
TYPE_L_KEYWORDS                  = 00000008               WRK_W_FLAGS2                       FFFFFFF2
VEC_K_HEADER_LENGTH              = 00000008               WRK_W_IMGCHAN                      FFFFFFEE
VEC_L_COMDPTR                    = 0000000C               WRK_W_PMPTLEN                      FFFFF99E
VEC_L_VERBTBL                    = 00000008               _$$_                             = 000000EF
VEC_W_SIZE                       = 00000000
V_AMBIG                          = 00000003
V_KEYWD                          = 00000001
V_NEGAT                          = 00000000
V_SYNTAX                         = 00000002
WRK_B_CMDOPT                       FFFFFFC3
WRK_B_MAXPARM                      FFFFFFD0
WRK_B_MINPARM                      FFFFFFD1
WRK_B_PARMCNT                      FFFFFFCE
WRK_B_PARMSUM                      FFFFFFCF
WRK_B_RECALLCNT                    FFFFFFC5
WRK_B_VALLEV                       FFFFFFC4
WRK_B_VERBTYP                      FFFFFFC2
WRK_C_CMDBUFSIZ                  = 00000400
WRK_C_LENGTH                       FFFFF486
WRK_G_BUFFER                       FFFFF492
WRK_G_INPBUF                       FFFFF896
WRK_G_RESULT                       FFFFF9B6
WRK_K_LENGTH                       FFFFF486
WRK_L_CHARPTR                      FFFFF48E
WRK_L_DISALLOW                     FFFFFFE6
WRK_L_ERRORRTN                     FFFFF9AE
WRK_L_EXPANDPTR                    FFFFF486
WRK_L_IMAGE                        FFFFFFE2
WRK_L_MARKPTR                      FFFFF48A
WRK_L_PAROUT                       FFFFFFD2
WRK_L_PMPTADDR                     FFFFF9A2
WRK_L_PROMPTRTN                    FFFFF9A6
WRK_L_PROPTR                       FFFFFFC6
WRK_L_QUABLK                       FFFFFFCA
WRK_L_READRTN                      FFFFF9AA
WRK_L_RECALLPTR                    FFFFFFEA
WRK_L_RSLEND                       FFFFFFB6
WRK_L_RSLNXT                       FFFFFFBA
WRK_L_SAVAP                        FFFFFFF8
WRK_L_SAVFP                        FFFFFFFC
WRK_L_SAVSP                        FFFFFFF4
WRK_L_SIGNALRTN                    FFFFFFD6
WRK_L_SPECRTN                      FFFFF9B2
WRK_L_TAB_VEC                      FFFFFFDE
WRK_L_VERB                         FFFFFFBE
WRK_M_CLIRTN                     = 00000001
WRK_M_USERRTN                    = 00000002
WRK_M_VERB                       = 00000040
WRK_V_CLIRTN                     = 00000000
```

```
                                           +----------------+
                                           ! Psect synopsis !
                                           +----------------+

PSECT name                       Allocation          PSECT No.  Attributes
----------                       ----------          ---------  ----------
.  ABS  .                        00000000  (     0.)  00 (   0.)  NOPIC    USR    CON    ABS    LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
$ABS$                            FFFFFFFC  (     0.)  01 (   1.)  NOPIC    USR    CON    ABS    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
DCL$ZCODE                        00000A83  (  2691.)  02 (   2.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD   NOWRT NOVEC BYTE

                                        +------------------------+
                                        ! Performance indicators !
                                        +------------------------+

Phase                     Page faults    CPU Time        Elapsed Time
-----                     -----------    --------        ------------
Initialization                      9    00:00:00.07     00:00:00.52
Command processing                 80    00:00:00.74     00:00:06.54
Pass 1                            298    00:00:12.52     00:00:40.79
Symbol table sort                   0    00:00:01.15     00:00:03.50
Pass 2                            327    00:00:04.61     00:00:20.91
Symbol table output                34    00:00:00.24     00:00:00.33
Psect synopsis output               2    00:00:00.03     00:00:00.03
Cross-reference output              0    00:00:00.00     00:00:00.00
Assembler run totals              750    00:00:19.37     00:01:12.62

The working set limit was 1800 pages.
69752 bytes (137 pages) of virtual memory were used to buffer the intermediate code.
There were 50 pages of symbol table space allocated to hold 688 non-local and 145 local symbols.
1951 source lines were read in Pass 1, producing 25 object records in Pass 2.
34 pages of virtual memory were used to define 20 macros.

                                       +--------------------------+
                                       ! Macro library statistics !
                                       +--------------------------+

Macro library name                        Macros defined
------------------                        --------------
_$255$DUA28:[SYSLIB]SYSBLDMLB.MLB;1                   0
_$255$DUA28:[DCL.OBJ]DCL.MLB;1                        9
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                        0
_$255$DUA28:[SYSLIB]STARLET.MLB;2                     5
TOTALS (all libraries)                               14

790 GETS were required to define 14 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:PARSENT/OBJ=OBJ$:PARSENT MSRC$:PARSENT/UPDATE=(ENH$:PARSENT)+EXECML$/LIB+LIB$:DCL/LIB+SYS$LIBRARY:SYSBLDMLB/LIB
```

0072 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

RPCLINT
LIS

RECALLSUB
LIS

MESSAGE
LIS

READREC
LIS

PARSENT
LIS

ON
LIS