M 11

```
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        CCCCCCCCCCCC  LLLLLLLLLLLLLLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLLLLLLLLLLLLLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLLLLLLLLLLLLLL
```

```
**FILE**ID**STATEMENT                                        L  2
```

```
  SSSSSSSS  TTTTTTTTTT    AAAAAA    TTTTTTTTTT  EEEEEEEEEE  MM      MM  EEEEEEEEEE  NN      NN  TTTTTTTTTT
  SSSSSSSS  TTTTTTTTTT    AAAAAA    TTTTTTTTTT  EEEEEEEEEE  MM      MM  EEEEEEEEEE  NN      NN  TTTTTTTTTT
SS              TT      AA      AA      TT      EE          MMMM  MMMM  EE          NN      NN      TT
SS              TT      AA      AA      TT      EE          MMMM  MMMM  EE          NN      NN      TT
SS              TT      AA      AA      TT      EE          MM  MM  MM  EE          NNNN    NN      TT
SS              TT      AA      AA      TT      EE          MM  MM  MM  EE          NNNN    NN      TT
  SSSSSS        TT      AA      AA      TT      EEEEEEEE    MM      MM  EEEEEEEE    NN  NN  NN      TT
  SSSSSS        TT      AA      AA      TT      EEEEEEEE    MM      MM  EEEEEEEE    NN  NN  NN      TT
        SS      TT      AAAAAAAAAA      TT      EE          MM      MM  EE          NN    NNNN      TT
        SS      TT      AAAAAAAAAA      TT      EE          MM      MM  EE          NN    NNNN      TT
        SS      TT      AA      AA      TT      EE          MM      MM  EE          NN      NN      TT          ....
        SS      TT      AA      AA      TT      EE          MM      MM  EE          NN      NN      TT          ....
SSSSSSSS        TT      AA      AA      TT      EEEEEEEEEE  MM      MM  EEEEEEEEEE  NN      NN      TT          ....
SSSSSSSS        TT      AA      AA      TT      EEEEEEEEEE  MM      MM  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  STATEMENT - ARITHMETIC ASSIGNMENT AND STRING EQUATE PROCESSING
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    ; ARITHMETIC ASSIGNMENT AND STRING EQUATE STATEMENT PROCESSING
0000    ;
0000    ; D. N. CUTLER 16-MAY-77
0000    ;
0000    ; MODIFIED BY:
0000    ;
0000    ;       V03-004 PCG0007         Peter George    27-Jul-1983
0000    ;               Add multi-national character support.
0000    ;
0000    ;       V03-003 PCG0006         Peter George    27-May-1983
0000    ;               Change error processing.
0000    ;
0000    ;       V03-002 PCG0005         Peter George    17-Jan-1983
0000    ;               Fix SETBIT typo.
0000    ;
C000    ;       V03-001 PCG0004         Peter George    09-Jun-1982
0000    ;               Check length of string substitution.
0000    ;---
```

```
    0000    ;
    0000    ; MACRO LIBRARY CALLS
    0000    ;
    0000
    0000             PRCDEF                            ;DEFINE PROCESS WORK AREA
    0000             WRKDEF                            ;DEFINE COMMAND WORK AREA
    0000             SYMDEF                            ;DEFINE TYPES OF SYMBOLS
    0000             $CLIMSGDEF                        ;DEFINE ERROR/STATUS VALUES
    0000
00000000             .PSECT  DCL$ZCODE,BYTE,RD,NOWRT
```

```
                        0000            .SBTTL  SUBSTRING ASSIGNMENT STATEMENT
                        0000    ;+
                        0000    ; DCL$SUBASSIGN - SUBSTRING ASSIGNMENT
                        0000    ;
                        0000    ; THIS ROUTINE IS CALLED WHEN A SYMBOL DEFINITION IS TERMINATED WITH A LEFT SQUARE
                        0000    ; BRACKET.
                        0000    ;
                        0000    ; INPUTS:
                        0000    ;
                        0000    ;       R1 = LENGTH OF SYMBOL NAME.
                        0000    ;       R2 = ADDRESS OF SYMBOL NAME.
                        0000    ;       R10 = BASE ADDRESS OF COMMAND WORK AREA.
                        0000    ;       R11 = BASE ADDRESS OF PROCESS WORK AREA.
                        0000    ;
                        0000    ; OUTPUTS:
                        0000    ;
                        0000    ;       THE POSITION AND SIZE PARAMETERS ARE PARSED AND THE ASSIGNMENT IS DISPATCHED
                        0000    ;       TO THE PROPER ASSIGNMENT STATEMENT PROCESSOR.
                        0000    ;-
                        0000
                        0000    DCL$SUBASSIGN::                         ;SUBSTRING ASSIGNMENT
        7E     51   7D  0000            MOVQ    R1,-(SP)                ;SAVE SYMBOL NAME PARAMETERS
  59  F486 CA       D0  0003            MOVL    WRK_L_EXPANDPTR(R10),R9 ;SAVE PTR TO FIELD SPECIFICATION
          FFF5'     30  0008            BSBW    DCL$MOVCHAR             ;MOVE RIGHT BRACKET
          FFF2'     30  000B            BSBW    DCL$EXPRESS             ;EVALUATE POSITION EXPRESSION
          42 50     E9  000E            BLBC    R0,90$                  ;BRANCH IF ERROR
        58     51   D0  0011            MOVL    R1,R8                   ;SAVE POSITION PARAMETER
          FFE9'     30  0014            BSBW    DCL$MOVCHAR             ;MOVE NEXT CHARACTER
        2C     50   91  0017            CMPB    R0,#^A','               ;COMMA?
               30   12  001A            BNEQ    20$                     ;ERROR IF NOT
          FFE1'     30  001C            BSBW    DCL$MARK                ;MARK CURRENT PARSE POSITION
          FFDE'     30  001F            BSBW    DCL$EXPRESS             ;EVALUATE SIZE EXPRESSION
          2E 50     E9  0022            BLBC    R0,90$                  ;BRANCH IF ERROR
        57     51   D0  0025            MOVL    R1,R7                   ;SAVE SIZE PARAMETER
          FFD5'     30  0028            BSBW    DCL$MOVCHAR             ;MOVE TERMINATOR TO COMMAND BUFFER
     5D 8F     50   91  002B            CMPB    R0,#^A']'               ;RIGHT BRACKET?
               1B   12  002F            BNEQ    20$                     ;ERROR IF NOT
          FFCC'     30  0031            BSBW    DCL$MARK                ;MARK CURRENT PARSE POSITION
          FFC9'     30  0034            BSBW    DCL$SETNBLK             ;PEEK AT NEXT CHARACTER IN COMMAND BUFFER
        50     3D   91  0037            CMPB    #^A/=/,R0               ;ARITHMETIC ASSIGNMENT?
               29   13  003A            BEQL    60$                     ;IF EQL YES
        50     3A   91  003C            CMPB    #^A/:/,R0               ;STRING EQUATE (:=)?
               0B   12  003F            BNEQ    20$                     ;IF NEQ NO
          FFBC'     30  0041            BSBW    DCL$MOVCHAR             ;MOVE NEXT CHARACTER TO COMMAND BUFFER
          FFB9'     30  0044            BSBW    DCL$SETCHAR             ;PEEK AT NEXT CHARACTER IN COMMAND BUFFER
        50     3D   91  0047            CMPB    #^A/=/,R0               ;STRING EQUATE (:=)?
               0B   13  004A            BEQL    50$                     ;BRANCH IF OK
                        004C    20$:    STATUS  SYMDEL                  ;INVALID SUBSTRING DELIMITER
        5E     08   C0  0053    90$:    ADDL    #8,SP                   ;POP SAVED SYMBOL NAME DESCRIPTOR
                    05  0056            RSB
                        0057    ;
                        0057    ; STRING ASSIGNMENT -- [POS,CHARS] := STRING
                        0057    ;
        51     8E   7D  0057    50$:    MOVQ    (SP)+,R1                ;RESTORE SYMBOL NAME PARAMETERS
     F48A CA   52   D0  005A            MOVL    R2,WRK_L_MARKPTR(R10)   ;RESET ERROR DISPLAY POINTER
        56     01   D0  005F            MOVL    #1,R6                   ;SET SUBSTRING INDICATOR
             010F   31  0062            BRW     EQUATE                  ;
```

```
                                  0065     115 ;
                                  0065     116 ; NUMERIC ASSIGNMENT -- [POS,BITS] = EXPRESSION
                                  0065     117 ;
                   51     8E   7D 0065     118 60$:      MOVQ     (SP)+,R1                    ;RESTORE SYMBOL NAME PARAMETERS
              F48A CA     52   D0 0068     119           MOVL     R2,WRK_L_MARKPTR(R10)       ;RESET ERROR DISPLAY POINTER
                   56     01   D0 006D     120           MOVL     #1,R6                       ;SET SUBSTRING INDICATOR
                          02   11 0070     121           BRB      EVALUATE                    ;
```

```
                0072        .SBTTL  EVALUATE ASSIGNMENT STATEMENT
                0072  ;+
                0072  ; DCL$EVALUATE - EVALUATE ASSIGNMENT STATEMENT
                0072  ;
                0072  ; THIS ROUTINE IS CALLED TO PROCESS AN ASSIGNMENT STATEMENT.
                0072  ;
                0072  ; INPUTS:
                0072  ;
                0072  ;       R' = LENGTH OF SYMBOL NAME.
                0072  ;       R2 = ADDRESS OF SYMBOL NAME.
                0072  ;       R9 = ADDRESS OF FIELD SPECIFICATION.
                0072  ;       R10 = BASE ADDRESS OF COMMAND WORK AREA.
                0072  ;       R11 = BASE ADDRESS OF PROCESS WORK AREA.
                0072  ;
                0072  ; OUTPUTS:
                0072  ;
                0072  ;       RO LOW BIT CLEAR INDICATES EXPRESSION ANALYSIS OR ASSIGNMENT FAILURE.
                0072  ;
                0072  ;               RO = DCL$_COMPLX - EXPRESSION TOO COMPLEX.
                0072  ;               RO = DCL$_EXPSYN - EXPRESSION SYNTAX ERROR.
                0072  ;               RO = DCL$_IVCHAR - INVALID NUMERIC CHARACTER.
                0072  ;               RO = DCL$_IVOPER - INVALID EXPRESSION OPERATOR.
                0072  ;               RO = DCL$_IVSYMB - INVALID SYMBOL CHARACTER.
                0072  ;               RO = DCL$_SYMDEL - INVALID SYMBOL DELIMITER.
                0072  ;               RO = DCL$_SYMOVF - SYMBOL TABLE OVERFLOW.
                0072  ;               RO = DCL$_UNDSYM - UNDEFINED SYMBOL.
                0072  ;               RO = DCL$_INVRANGE  - FIELD SPECIFICATION OUT OF BOUNDS
                0072  ;
                0072  ;       PO LOW BIT SET INDICATES SUCCESSFUL COMPLETION.
                0072  ;
                0072  ;               RO = DCL$_NORMAL - NORMAL COMPLETION.
                0072  ;-
                0072
                0072  DCL$EVALUATE::                                  ;EVALUATE ASSIGNMENT STATEMENT
      56    D4  0072          CLRL    R6                              ;CLEAR BIT FIELD SUBSTRING INDICATOR
                0074  ;
                0074  ; ON ENTRY, IF R6 = 1,
                0074  ;
                0074  ;       R7 = SUBSTRING FIELD WIDTH
                0074  ;       R8 = SUBSTRING STARTING BIT NUMBER
                0074  ;
                0074  EVALUATE:                                       ;EVALUATE EXPRESSION
                0074          SETBIT  PRC_V_IND,PRC_W_FLAGS(R11) ;DISABLE INDIRECT FILE RECOGNITION
  57   20   D1  0078          CMPL    #32,R7                          ;FIELD WIDTH GREATER THAN LONGWORD?
       03   1E  007B          BGEQU   10$                             ;IF GEQU NO
  57   20   D0  007D          MOVL    #32,R7                          ;REDUCE FIELD WIDTH TO A LONGWORD
     01EC   30  0080  10$:    BSBW    CHECKSYM                        ;CHECK SYMBOL AND SAVE PARAMETERS
  03C0 8F   BB  0083          PUSHR   #^M<R6,R7,R8,R9>                ;SAVE BIT FIELD SUBSTRING PARAMETERS
    FF76'   30  0087          BSBW    DCL$EXPRESS                     ;EVALUATE ARITHMETIC EXPRESSION
  03C0 8F   BA  008A          POPR    #^M<R6,R7,R8,R9>                ;RESTORE BIT FIELD SUBSTRING PARAMETERS
       38   BA  008E          POPR    #^M<R3,R4,R5>                   ;RETRIEVE SYMBOL NAME PARAMETERS
    01 50   E8  0090          BLBS    R0,12$                          ;BRANCH IF OK
            05  0093          RSB                                     ;EXIT WITH ERROR
                0094  ;
                0094  ; IF NO SUBSTRING SPECIFIED, STORE VALUE IN SYMBOL
                0094  ;
    10 56   E8  0094  12$:    BLBS    R6,15$                          ;IF LBC NO BIT FIELD SUBSTRING SPECIFIED
```

```
            50     00  D0  0097          MOVL    #SYM_K_STRING,R0          ;ASSUME STRING VALUE
                   52  D5  009A          TSTL    R2                        ;STRING VALUE?
                   03  12  009C          BNEQ    13$                       ;BRANCH IF SO
            50     02  D0  009E          MOVL    #SYM_K_BINARY,R0          ;SET TO BINARY VALUE
                FF5C'  30  00A1  13$:    BSBW    DCL$ALLOCSYMABR           ;ALLOCATE AND INSERT SYMBOL IN TABLE
                 00A7  31  00A4          BRW     70$                       ;CLEANUP INPUT STREAM
                           00A7  ;
                           00A7  ; FIND PREVIOUS VALUE OF SYMBOL
                           00A7  ;
                   52  D5  00A7  15$:    TSTL    R2                        ;INSERTING STRING INTO BIT FIELD?
                   03  13  00A9          BEQL    16$                       ;OK IF NOT
                 00B4  31  00AB          BRW     IVCHAR                    ;ERROR IF SO
                   3A  BB  00AE  16$:    PUSHR   #^M<R1,R3,R4,R5>          ;SAVE EXPRESSION AND SYMBOL NAME PARAMETERS
            51     53  7D  00B0          MOVQ    R3,R1                     ;SET SYMBOL NAME PARAMETERS FOR SEARCH
       50     38   AB  9E  00B3          MOVAB   PRC_Q_LOCAL(R11),R0       ;GET ADDRESS OF LOCAL SYMBOL TABLES
            50     55  D1  00B7          CMPL    R5,R0                     ;WHICH TABLES SHOULD WE SEARCH?
                   05  12  00BA          BNEQ    17$                       ;IF NEQ, THEN GLOBAL
                FF41'  30  00BC          BSBW    DCL$SEARCH_LOCAL          ;SEARCH LOCAL SYMBOL TABLES
                   03  11  00BF          BRB     18$                       ;SKIP NEXT SEARCH
                FF3C'  30  00C1  17$:    BSBW    DCL$SEARCH_GLOBAL         ;SEARCH GLOBAL SYMBOL TABLE
                02 50  E8  00C4  18$:    BLBS    R0,20$                    ;BRANCH IF SUCCESSFUL SEARCH
                   51  D4  00C7          CLRL    R1                        ;UNSUCESSFUL, ASSUME NULL STRING
                           00C9  ;
                           00C9  ; IF PREVIOUS VALUE WAS AN INTEGER, THEN PERFORM BOUNDS CHECKING AND THEN
                           00C9  ; INSERT THE NEW VALUE INTO THE LONGWORD.
                           00C9  ;
                   52  D5  00C9  20$:    TSTL    R2                        ;BINARY VALUE?
                   29  12  00CB          BNEQ    25$                       ;BRANCH IF NOT
            58     20  D1  00CD          CMPL    #32,R8                    ;STARTING BIT OUTSIDE LONGWORD?
                   14  1B  00D0          BLEQU   23$                       ;IF LEQU YES
       50     58   57  C1  00D2          ADDL3   R7,R8,R0                  ;FIND ENDING BIT
              50   20  D1  00D6          CMPL    #32,R0                    ;IS IT OUTSIDE THE LONGWORD?
                   0B  1F  00D9          BLSSU   23$                       ;IF LSSU YES
  51    57    58   8E  F0  00DB          INSV    (SP)+,R8,R7,R1            ;INSERT BIT FIELD INTO LONGWORD
            50     02  D0  00E0          MOVL    #SYM_K_BINARY,R0          ;SET TYPE OF SYMBOL
                 0063  31  00E3          BRW     55$                       ;STORE THE SYMBOL AWAY
       F48A CA     59  D0  00E6  23$:    MOVL    R9,WRK_L_MARKPTR(R10)     ;RESET PTR TO ERROR SEGMENT
              5E   10  C0  00EB          ADDL    #4*4,SP                   ;RESTORE THE STACK
  50     00038228  8F  D0  00EE          MOVL    #CLI$_INVRANGE,R0         ;SET INVALID RANGE
                       05  00F5          RSB
                           00F6  ;
                           00F6  ; INITIALIZE THE EXPANSION BUFFER (IT WILL HOLD THE RESULT) AND
                           00F6  ; CALCULATE THE EXTENT OF THE STRING INSERTION
                           00F6  ;
  58     00001808  8F  D1  00F6  25$:    CMPL    #<WRK_C_CMDBUFSIZ-255>*8,R8      ;STARTING BIT OUTSIDE BUFFER?
                   E7  1B  00FD          BLEQU   23$                       ;IF LEQU YES
       56     58   57  C1  00FF          ADDL3   R7,R8,R6                  ;CALCULATE ENDING BIT NUMBER
              56   07  C0  0103          ADDL    #7,R6                     ;ROUND UP TO NEXT BYTE
              56   08  C6  0106          DIVL    #8,R6                     ;CALCULATE ENDING BYTE NUMBER
  56     00000301  8F  D1  0109          CMPL    #WRK_C_CMDBUFSIZ-255,R6 ;WILL STRING FIT IN BUFFER?
                   D4  1F  0110          BLSSU   23$                       ;NO, THEN SIGNAL ERROR
  F486 CA   F591   CA  9E  0112          MOVAB   WRK_G_BUFFER+255(R10),WRK_L_EXPANDPTR(R10) ;RELOAD ADDRESS OF EXPANS
                FEE4'  30  0119          BSBW    DCL$MARK                  ;MARK CURRENT PARSE POSITION
            55     51  D0  011C          MOVL    R1,R5                     ;SAVE LENGTH OF SYMBOL STRING
                           011F  ;
                           011F  ; MOVE PREVIOUS STRING VALUE INTO THE EXPANSION BUFFER
                           011F  ;
                55  D7  011F  27$:     DECL    R5                          ;ANY MORE CHARACTERS TO COPY?
                08  19  0121           BLSS    30$                         ;IF LSS 0
           50   82  9A  0123           MOVZBL  (R2)+,R0                    ;GET NEXT SYMBOL CHARACTER
             FED7'  30  0126           BSBW    DCL$PUTCHAR                 ;PUT CHARACTER IN COMMAND BUFFER
                F4  11  0129           BRB     27$                         ;
                        012B     ;
                        012B     ; EXTEND THE STRING AS NEEDED
                        012B     ;
      55   56   51  C3  012B  30$:     SUBL3   R1,R6,R5                    ;CALCULATE NUMBER OF EXTENSION CHARACTERS
                08  15  012F           BLEQ    50$                         ;IF LEQ NO EXTENSION REQUIRED
           51   56  D0  0131           MOVL    R6,R1                       ;SET LENGTH OF RESULTANT STRING
                50  D4  0134  40$:     CLRL    R0                          ;CLEAR EXTENSION BYTE
             FEC7'  30  0136           BSBW    DCL$PUTCHAR                 ;PUT CHARACTER IN COMMAND BUFFER
             F8 55  F5  0139           SOBGTR  R5,40$                      ;ANY MORE EXTENSION CHARACTERS?
                        013C     ;
                        013C     ; INSERT THE VALUE INTO THE STRING
                        013C     ;
      52   F591 CA  9E  013C  50$:     MOVAB   WRK_G_BUFFER+255(R10),R2 ;GET ADDRESS OF NEW SYMBOL VALUE
 62   57   58   8E  F0  0141           INSV    (SP)+,R8,R7,(R2)            ;INSERT BIT FIELD SUBSTRING
           50   00  D0  0146           MOVL    #SYM_K_STRING,R0            ;SET TYPE OF SYMBOL
                        0149     ;
                        0149     ; STORE THE SYMBOL AWAY
                        0149     ;
                38  BA  0149  55$:     POPR    #^M<R3,R4,R5>               ;RETRIEVE SYMBOL PARAMETERS
             FEB2'  30  014B           BSBW    DCL$ALLOCSYM                ;ALLOCATE AND INSERT SYMBOL IN TABLE
                        014E     ;
                        014E     ; ENSURE THAT THERE ARE NO MORE CHARACTERS IN THE INPUT STREAM
                        014E     ;
                        014E  70$:     SETBIT  WRK_V_COMMAND,WRK_W_FLAGS(R10) ;SET COMMAND EXECUTION IN PROGRESS
             0C 50  E9  0152           BLBC    R0,90$                      ;IF LBC SYMBOL ALLOCATION ERROR
             FEA8'  30  0155           BSBW    DCL$SETCHAR                 ;PEEK AT NEXT CHARACTER IN INPUT BUFFER
                10  12  0158           BNEQ    SYMDEL                      ;IF NEQ INVALID SYMBOL DELIMITER
                        015A           STATUS  NORMAL                      ;SET NORMAL COMPLETION
                    05  0161  90$:     RSB                                 ;
                        0162
                        0162  IVCHAR:  STATUS  IVCHAR                      ;INVALID NUMERIC CHARACTER
                    05  0169           RSB
                        016A
                        016A  SYMDEL:  STATUS  SYMDEL                      ;SET INVALID SYMBOL DELIMITER
                    05  0171           RSB                                 ;
```

```
                              0172            .SBTTL  PROCESS STRING EQUATE STATEMENT
                              0172     ;+
                              0172     ; DCL$EQUATE - PROCESS STRING EQUATE STATEMENT
                              0172     ;
                              0172     ; THIS ROUTINE IS CALLED TO PROCESS A STRING EQUATE STATEMENT.
                              0172     ;
                              0172     ; INPUTS:
                              0172     ;
                              0172     ;       R0 = TERMINATOR CHARACTER.
                              0172     ;       R1 = LENGTH OF SYMBOL NAME.
                              0172     ;       R2 = ADDRESS OF SYMBOL NAME.
                              0172     ;       R9 = ADDRESS OF FIELD SPECIFICATION.
                              0172     ;       R10 = BASE ADDRESS OF COMMAND WORK AREA.
                              0172     ;       R11 = BASE ADDRESS OF PROCESS WORK AREA.
                              0172     ;
                              0172     ; OUTPUTS:
                              0172     ;
                              0172     ;       R0 LOW BIT CLEAR INDICATES SYMBOL ALLOCATION FAILURE.
                              0172     ;
                              0172     ;               R0 = DCL$_INSYMB - INVALID SYMBOL CHARACTER.
                              0172     ;               R0 = DCL$_SYMDEL - INVALID SYMBOL DELIMITER.
                              0172     ;               R0 = DCL$_SYMOVF - SYMBOL TABLE OVERFLOW.
                              0172     ;               R0 = DCL$_INVRANGE  - FIELD SPECIFICATION OUT OF BOUNDS
                              0172     ;
                              0172     ;       R0 LOW BIT SET INDICATES SUCCESSFUL ALLOCATION.
                              0172     ;
                              0172     ;               R0 = DCL$_NORMAL - NORMAL COMPLETION.
                              0172     ;-
                              0172
                              0172     DCL$EQUATE::                                    ;PROCESS STRING EQUATE STATEMENT
                   56    D4   0172            CLRL    R6                              ;CLEAR SUBSTRING INDICATOR
                              0174     ;
                              0174     ; ON ENTRY, IF R6 = 1,
                              0174     ;
                              0174     ;       R7 = SUBSTRING FIELD WIDTH
                              0174     ;       R8 = SUBSTRING STARTING CHARACTER NUMBER
                              0174     ;
                              0174     EQUATE:                                         ;EQUATE SYMBOL
                              0174            SETBIT  PRC_V_IND,PRC_W_FLAGS(R11) ;DISABLE INDIRECT FILE RECOGNITION
                 00F4    30   0178            BSBW    CHECKSYM                        ;CHECK SYMBOL AND SAVE PARAMETERS
                 FE82'   30   017B            BSBW    DCL$SETNBLK                     ;SET TO FIRST NON-BLANK CHARACTER
                 FE7F'   30   017E            BSBW    DCL$MARK                        ;MARK CURRENT POSITION IN BUFFER
                 FE7C'   30   0181     20$:   BSBW    DCL$MOVCHAR                     ;COPY NEXT CHARACTER INTO THE BUFFER
                   FB    12   0184            BNEQ    20$                             ;UNTIL END OF LINE
                 FE77'   30   0186            BSBW    DCL$MARKEDTOKEN                 ;GET DESCRIPTOR OF MARKED STRING
                   51    D7   0189            DECL    R1                              ;SUBTRACT OUT EOL
                   03    13   018B            BEQL    40$                             ;IF EQL NULL STRING VALUE
                 FE70'   30   018D            BSBW    DCL$COMPRESS                    ;COMPRESS QUOTED STRING
           00000000'EF   9F   0190     40$:   PUSHAB  DCL$ALLOCSYMABR                 ;GUESS AT ALLOWING ABBREVIATED SYMBOL
                03 56    E8   0196            BLBS    R6,41$                          ;IF LBS, SUBSTRING SPECIFIED
                 0080    31   0199            BRW     120$                            ;BRANCH IF NOT SUBSTRING ASSIGNMENT
                              019C     ;
                              019C     ; CHECK BOUNDS OF INSERTION.
                              019C     ;
     58    00000301 8F   D1   019C     41$:   CMPL    #WRK_C_CMDBUFSIZ-255,R8 ;STARTING CHAR OUTSIDE BUFFER?
                   0D    1B   01A3            BLEQU   42$                             ;IF LEQU YES
          50    58 57    C1   01A5            ADDL3   R7,R8,R0                        ;CALCULATE POSITION BEYOND SUBSTRING
```

```
       50   00000301 8F   D1  01A9   335               CMPL    #WRK_C_CMDBUFSIZ-255,R0 ;ENDING CHAR OUTSIDE BUFFER?
                   10     1E  01B0   336               BGEQU   43$                     ;IF GEQU NO
          F48A CA  59     D0  01B2   337 42$:          MOVL    R9,WRK_L_MARKPTR(R10)   ;RESET PTR TO ERROR SEGMENT
               5E  10     C0  01B7   338               ADDL    #4+4,SP                 ;RESTORE THE STACK
       50   00038228 8F   D0  01BA   339               MOVL    #CLI$_INVRANGE,R0       ;SET INVALID RANGE
                          05  01C1   340               RSB
                              01C2   341 ;
                              01C2   342 ; FIND PREVIOUS VALUE OF SYMBOL
                              01C2   343 ;
              7E   51     7D  01C2   344 43$:          MOVQ    R1,-(SP)                ;SAVE EQUIVALENCE STRING PARAMETERS
          51    0C AE     7D  01C5   345               MOVQ    12(SP),R1               ;RETRIEVE SYMBOL NAME PARAMETERS
          50    38 AB     9E  01C9   346               MOVAB   PRC_Q_LOCAL(R11),R0     ;GET ADDRESS OF LOCAL SYMBOL TABLES
          50    14 AE     D1  01CD   347               CMPL    20(SP),R0               ;WHICH TABLES SHOULD WE SEARCH?
                   05     12  01D1   348               BNEQ    47$                     ;IF NEQ, THEN GLOBAL
                FE2A'     30  01D3   349               BSBW    DCL$SEARCH_LOCAL        ;SEARCH LOCAL SYMBOL TABLES
                   03     11  01D6   350               BRB     48$                     ;SKIP NEXT SEARCH
                FE25'     30  01D8   351 47$:          BSBW    DCL$SEARCH_GLOBAL       ;SEARCH GLOBAL SYMBOL TABLE
                02 50     E8  01DB   352 48$:          BLBS    R0,49$                  ;BRANCH IF SUCCESSFUL SEARCH
                   51     D4  01DE   353               CLRL    R1                      ;UNSUCCESSFUL, ASSUME NULL STRING
                              01E0   354 ;
                              01E0   355 ; IF PREVIOUS VALUE WAS AN INTEGER, CONVERT IT TO A STRING
                              01E0   356 ;
  F486 CA     F591 CA     9E  01E0   357 49$:          MOVAB   WRK_G_BUFFER+255(R10),WRK_L_EXPANDPTR(R10) ;RELOAD ADDRESS OF COMMAN
                FE16'     30  01E7   358               BSBW    DCL$MARK                ;MARK CURRENT PARSE POSITION
                   56     D4  01EA   359               CLRL    R6                      ;CLEAR STRING POSITION COUNT
                   52     D5  01EC   360               TSTL    R2                      ;BINARY VALUE?
                   06     12  01EE   361               BNEQ    50$                     ;BRANCH IF NOT
          50       51     D0  01F0   362               MOVL    R1,R0                   ;BINARY VALUE TO BE CONVERTED TO ASCII
                FE0A'     30  01F3   363               BSBW    DCL$CBTA_DEC            ;PUT RESULT IN COMMAND BUFFER
                              01F6   364 ;
                              01F6   365 ; MOVE OLD STRING INTO BUFFER, UNTIL SUBSTRING POSITION IS REACHED
                              01F6   366 ;
          58       56     D1  01F6   367 50$:          CMPL    R6,R8                   ;SUBSTRING POSITION REACHED?
                   13     13  01F9   368               BEQL    70$                     ;IF EQL YES
          50       20     9A  01FB   369               MOVZBL  #^A' ',R0               ;ASSUME NO INPUT CHARACTERS LEFT
                   51     D5  01FE   370               TSTL    R1                      ;ANY MORE CHARACTERS IN INPUT STRING
                   05     13  0200   371               BEQL    60$                     ;IF EQL NO
                   51     D7  0202   372               DECL    R1                      ;REDUCE NUMBER OF CHARACTERS REMAINING
          50       82     9A  0204   373               MOVZBL  (R2)+,R0                ;GET NEXT CHARACTER FROM INPUT STRING
                FDF6'     30  0207   374 60$:          BSBW    DCL$PUTCHAR             ;PUT CHARACTER IN COMMAND BUFFER
                   56     D6  020A   375               INCL    R6                      ;INCREMENT STRING POSITION
                   E8     11  020C   376               BRB     50$                     ;
                              020E   377 ;
                              020E   378 ; SUBSTRING POSITION HAS BEEN FOUND, INSERT THE SUBSTRING
                              020E   379 ;
          58       57     C0  020E   380 70$:          ADDL    R7,R8                   ;CALCULATE POSITION BEYOND SUBSTRING
          53       8E     7D  0211   381               MOVQ    (SP)+,R3                ;RETRIEVE SUBSTRING PARAMETERS
          52       57     C0  0214   382               ADDL    R7,R2                   ;CALCULATE ADDRESS OF NEXT CHARACTER
          51       57     C2  0217   383               SUBL    R7,R1                   ;REDUCE NUMBER OF CHARACTERS REMAINING
                   02     18  021A   384               BGEQ    80$                     ;IF GEQ MORE CHARACTERS IN INPUT STRING
                   51     D4  021C   385               CLRL    R1                      ;CLEAR NUMBER OF CHARACTERS REMAINING
          58       56     D1  021E   386 80$:          CMPL    R6,R8                   ;END OF SUBSTRING?
                   13     13  0221   387               BEQL    100$                    ;IF EQL YES
          50       20     9A  0223   388               MOVZBL  #^A/ /,R0               ;ASSUME NO MORE CHARACTERS IN SUBSTRING
                   53     D5  0226   389               TSTL    R3                      ;ANY MORE CHARACTERS IN SUBSTRING?
                   05     13  0228   390               BEQL    90$                     ;IF EQL NO
                   53     D7  022A   391               DECL    R3                      ;REDUCE NUMBER OF CHARACTERS IN SUBSTRING
          50    84    9A 022C            MOVZBL  (R4)+,R0                ;GET NEXT CHARACTER FROM SUBSTRING
             FDCE'    30 022F     90$:   BSBW    DCL$PUTCHAR             ;PUT CHARACTER IN COMMAND BUFFER
                56    D6 0232            INCL    R6                      ;INCREMENT STRING POSITION
                E8    11 0234            BRB     80$                     ;
                         0236     ;
                         0236     ; APPEND ANY OLD VALUE CHARACTERS THAT ARE LEFT TO THE END OF THE STRING
                         0236     ;
                51    D7 0236     100$:  DECL    R1                      ;ANY MORE CHARACTERS IN INPUT STRING?
                08    19 0238            BLSS    110$                    ;IF LSS NO
          50    82    9A 023A            MOVZBL  (R2)+,R0                ;GET NEXT CHARACTER FROM INPUT STRING
             FDC0'    30 023D            BSBW    DCL$PUTCHAR             ;PUT CHARACTER IN COMMAND BUFFER
                F4    11 0240            BRB     100$                    ;
             FDBB'    30 0242     110$:  BSBW    DCL$MARKEDTOKEN         ;GET DESCRIPTOR OF RESULTANT STRING
6E    00000000'EF     9E 0245            MOVAB   DCL$ALLOCSYM,(SP)       ;DON'T ALLOW ANY ABBREVIATION
                         024C     ;
                         024C     ; STORE FINAL VALUE INTO SYMBOL
                         024C     ;
                         024C     ; R1 = DESCRIPTOR OF SYMBOL VALUE
                         024C     ; (SP) = ROUTINE TO INSERT ENTRY INTO SYMBOL TABLE
                         024C     ; 4(SP) = DESCRIPTOR OF SYMBOL NAME
                         024C     ; 12(SP) = ADDRESS OF SYMBOL TABLE LISTHEAD
                         024C     ;
             56 8ED0     024C     120$:  POPL    R6                      ;RETRIEVE SUBROUTINE ADDRESS
                38    BA 024F            POPR    #^M<R3,R4,R5>           ;RETRIEVE SYMBOL PARAMETERS
      FF 8F     51    91 0251            CMPB    R1,#255                 ;STRING TO LONG
                04    1B 0255            BLEQU   130$                    ;IF LEQU NO
   51    FF 8F        9A 0257            MOVZBL  #255,R1                 ;TRUNCATE TO MAXIMUM LENGTH
        50      00    D0 025B     130$:  MOVL    #SYM_K_STRING,R0        ;SET TYPE OF SYMBOL
                66    16 025E            JSB     (R6)                    ;ALLOCATE AND INSERT ENTRY IN SYMBOL TABLE
      F0 AA     02    A8 0260            BISW    #WRK_M_COMMAND,WRK_W_FLAGS(R10) ;SET COMMAND EXECUTION IN PROGRESS
             07 50    E9 0264            BLBC    R0,140$                 ;IF LBC ERROR
                         0267            STATUS  NORMAL                  ;SET NORMAL COMPLETION STATUS
                      05 026E     140$:  RSB                             ;
```

```
                                026F   426           .SBTTL  LOCAL SUBROUTINES
                                026F   427 ;
                                026F   428 ; CHECKSYM - CHECK SYMBOL AND SAVE PARAMETERS
                                026F   429 ;
                                026F   430
                                026F   431 CHECKSYM:                                        ;CHECK SYMBOL
                      10   BA   026F   432           POPR    #^M<R4>                  ;REMOVE RETURN ADDRESS FROM STACK
                    0039   30   0271   433           BSBW    DCL$CHKALPHA             ;VERIFY IT IS VALID
                   32 50   E9   0274   434           BLBC    R0,50$                   ;BR IF INVALID FIRST CHARACTER
   000000FF 8F        51   D1   0277   435           CMPL    R1,#255                  ;SYMBOL NAME TO LARGE
                      22   1A   027E   436           BGTRU   40$                      ;IF GTRU YES
          53    38 AB   9E   0280   437           MOVAB   PRC_Q_LOCAL(R11),R3      ;ASSUME LOCAL SYMBOL TABLE
                   FD79'   30   0284   438           BSBW    DCL$MOVCHAR              ;MOVE CHARACTER TO COMMAND BUFFER
                   FD76'   30   0287   439           BSBW    DCL$SETCHAR              ;PEEK AT NEXT CHARACTER IN INPUT BUFFER
             50       3D   91   028A   440           CMPB    #^A/=/,R0                ;ANOTHER EQUAL SIGN?
                      07   12   028D   441           BNEQ    20$                      ;IF NEQ NO
                   FD6E'   30   028F   442           BSBW    DCL$MOVCHAR              ;MOVE CHARACTER TO COMMAND BUFFER
          53    28 AB   9E   0292   443           MOVAB   PRC_Q_GLOBAL(R11),R3     ;SET FOR GLOBAL SYMBOL TABLE
                      0E   BB   0296   444 20$:      PUSHR   #^M<R1,R2,R3>            ;SAVE SYMBOL PARAMETERS
                                0298   445           DISABLE                          ;DISABLE CONTROL Y/C AST'S
                      8E   7C   029E   446           CLRQ    (SP)+                    ;REMOVE RETURN INFORMATION FROM STACK
                      64   17   02A0   447           JMP     (R4)                     ;
                                02A2   448
                                02A2   449 40$:      STATUS  SYMLNG                   ;SET SYMBOL TOO LONG STATUS
                           05   02A9   450 50$:      RSB                              ;
```

```
                02AA            .SBTTL  CHECK A CHARACTER FOR VALID ALPHA
                02AA      ;+
                02AA      ; DCL$CHKALPHA - CHECK A CHARACTER FOR VALID ALPHA
                02AA      ;
                02AA      ; THIS ROUTINE CHECKS THAT THE CHARACTER POINTED TO BY R2 IS VALID
                02AA      ; FIRST CHARACTER FOR A SYMBOL.
                02AA      ;
                02AA      ; INPUTS:
                02AA      ;
                02AA      ;         R2 = ADDRESS OF CHARACTER
                02AA      ;
                02AA      ; OUTPUT:
                02AA      ;
                02AA      ;         R0 = IVSYMB IS NOT ACCEPTABLE
                02AA      ;         R0 = NORMAL IS ACCEPTABLE CHARACTER
                02AA      ;-
                02AA      SPECIAL:
         5F 24  02AA            .ASCII  '$_'
            DF  02AC            .BYTE   ^XDF                          ; BETA
                02AD      ENDSPEC:
                02AD
                02AD      DCL$CHKALPHA::                              ; CHAECK VALID ALPHA
      51    DD  02AD            PUSHL   R1                            ; SAVE R1
  03  62    3A  02AF            LOCC    (R2),#ENDSPEC-SPECIAL,-       ; CHECK FOR SPECIAL CHARS
    F6 AF       02B2                    SPECIAL                       ;
      18    12  02B4            BNEQ    10$                           ; BRANCH IF FOUND
                02B6
41 8F 62    91  02B6            CMPB    (R2),#^A/A/                   ; CHECK LOW LIMIT OF LOW RANGE
      1D    1F  02BA            BLSSU   30$                           ; BR IF FAILED
5A 8F 62    91  02BC            CMPB    (R2),#^A/Z/                   ; CHECK HIGH LIMIT OF LOW RANGE
      0C    1B  02C0            BLEQU   10$                           ; BR IF VALID CHARACTER
C0 8F 62    91  02C2            CMPB    (R2),#^XC0                    ; CHECK LOW LIMIT OF HIGH RANGE
      11    1F  02C6            BLSSU   30$                           ; BR IF FAILED
DE 8F 62    91  02C8            CMPB    (R2),#^XDE                    ; CHECK HIGH LIMIT OF HIGH RANGE
      0B    1A  02CC            BGTRU   30$                           ; BR IF FAILED
                02CE
                02CE      10$:  STATUS  NORMAL                        ; SET SUCESSFUL STATUS
      51  8ED0  02D5      20$:  POPL    R1                            ; RESTORE R1
            05  02D8            RSB                                   ; RETURN WITH STATUS
                02D9
                02D9      30$:  STATUS  IVSYMB                        ;SET INVALID SYMBOL CHARACTER
      F3    11  02E0            BRB     20$                           ;
                02E2
                02E2            .END
```

```
CHECKSYM                        0000026F R      02      PRC_L_LASTKEY                 0000004C
CLI$_INVRANGE                 = 00038228                PRC_L_LSTSTATUS               000000B0
CLI$_IVCHAR                   = 00038050                PRC_L_ONCTLY                  000000B8
CLI$_IVSYMB                   = 00038080                PRC_L_ONERROR                 0000006C
CLI$_NORMAL                   = 00030001                PRC_L_OUTOFBAND               000000B4
CLI$_SYMDEL                   = 00038130                PRC_L_OUTRAB                  0000000C
CLI$_SYMLNG                   = 00038270                PRC_L_OUTRABCTX               00000118
DCL$ALLOCSYM                    ********    X   02      PRC_L_PPFLIST                 00000070
DCL$ALLOCSYMABR                 ********    X   02      PRC_L_RECALLPTR               0000012F
DCL$CBTA_DEC                    ********    X   02      PRC_L_RESTART                 00000058
DCL$CHKALPHA                    000002AD RG     02      PRC_L_SAVAP                   00000000
DCL$COMPRESS                    ********    X   02      PRC_L_SAVFP                   00000004
DCL$DISABLE                     ********    X   02      PRC_L_SEVERITY                00000050
DCL$EQUATE                      00000172 RG     02      PRC_L_SPWN                    000000C0
DCL$EVALUATE                    00000072 RG     02      PRC_L_STACKLM                 000000A4
DCL$EXPRESS                     ********    X   02      PRC_L_STACKPT                 000000A0
DCL$MARK                        ********    X   02      PRC_L_STATUS                  00000054
DCL$MARKEDTOKEN                 ********    X   02      PRC_L_STS                     00000084
DCL$MOVCHAR                     ********    X   02      PRC_L_STV                     00000088
DCL$PUTCHAR                     ********    X   02      PRC_L_SYMBOL                  00000060
DCL$SEARCH_GLOBAL               ********    X   02      PRC_L_TMBX                    00000074
DCL$SEARCH_LOCAL                ********    X   02      PRC_L_TRMLIST                 00000010
DCL$SETCHAR                     ********    X   02      PRC_Q_ALLOCREG                00000020
DCL$SETNBLK                     ********    X   02      PRC_Q_COMMAND                 000000E0
DCL$SUBASSIGN                   00000000 RG     02      PRC_Q_FLUSHTIME               000000D0
ENDSPEC                         000002AD R      02      PRC_Q_GLOBAL                  00000028
EQUATE                          00000174 R      02      PRC_Q_IMAGENAME               000000D8
EVALUATE                        00000074 R      02      PRC_Q_KEYPAD                  00000040
IVCHAR                          00000162 R      02      PRC_Q_LABEL                   00000030
PRC_B_CONTINUE                  000000F3                PRC_Q_LOCAL                   00000038
PRC_B_DEFRADIX                  000000AE                PRC_Q_SAVEPRIV                000000E8
PRC_B_EXMDEPMOD                 000000AD                PRC_T_OUTDVI                  0000011C
PRC_B_EXMDEPWID                 000000AC                PRC_V_IND                   = 00000005
PRC_B_EXONLYL                   0000012D                PRC_W_ASTIOSB                 000000C6
PRC_B_FLAGS2                    000000AF                PRC_W_ASTRETN                 000000C8
PRC_B_IMGFLAG                   00000078                PRC_W_ASTSTATUS               000000C4
PRC_B_OUTFLAGS                  0000012C                PRC_W_ATTMBX                  0000007A
PRC_B_PROMPTLEN                 000000F0                PRC_W_FLAGS                   00000068
PRC_C_LENGTH                    00000534                PRC_W_INPCHAN                 00000064
PRC_G_COMMANDS                  00000133                PRC_W_ONLEVEL                 0000006A
PRC_G_PROMPT                    000000F4                PRC_W_OUTIFI                  00000114
PRC_K_LENGTH                    00000534                PRC_W_OUTISI                  00000116
PRC_L_CURRKEY                   00000048                PRC_W_OUTMBXCHN               000000CA
PRC_L_EXMDEPADR                 000000A8                PRC_W_OUTMBXREF               000000CE
PRC_L_EXTARG                    00000094                PRC_W_OUTMBXSIZ               000000CC
PRC_L_EXTBLK                    0000008C                PRC_W_PMPTCTRL                000000F1
PRC_L_EXTCOD                    0000009C                PRC_W_WAITIOSB                00000066
PRC_L_EXTHND                    00000090                SPECIAL                       000002AA R      02
PRC_L_EXTPRM                    00000098                SYMDEL                        0000016A R      02
PRC_L_IDFLNK                    000000BC                SYM_B_FLAGS                   0000000B
PRC_L_IMGACTSTS                 00000080                SYM_B_NONUNIQUE               0000000B
PRC_L_INDCLOCK                  0000007C                SYM_B_TYPE                    0000000A
PRC_L_INDEPTH                   0000005C                SYM_K_BINARY                = 00000002
PRC_L_INDFAB                    0000001C                SYM_K_STRING                = 00000000
PRC_L_INDINPRAB                 00000014                SYM_L_BL                      00000004
PRC_L_INDOUTRAB                 00000018                SYM_L_FL                      00000000
PRC_L_INPRAB                    00000008                SYM_T_SYMBOL                  0000000C
```

```
SYM_W_SIZE                          00000008
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_COMMAND                     = 00000002
WRK_V_COMMAND                     = 00000001
WRK_W_FLAGS                         FFFFFFF0
WRK_W_FLAGS2                        FFFFFFF2
WRK_W_IMGCHAN                       FFFFFFEE
WRK_W_PMPTLEN                       FFFFF99E
_$$_                              = 000000EF

                                                  +----------------+
                                                  ! 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                         000002E2  (  738.)  02 (  2.)  NOPIC   USR   CON   REL   LCL NOSHR   EXE   RD   NOWRT NOVEC BYTE
```

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

Phase                       Page faults    CPU Time       Elapsed Time
-----                       -----------    --------       ------------
Initialization                        9    00:00:00.05    00:00:02.04
Command processing                   84    00:00:00.65    00:00:07.14
Pass 1                              198    00:00:06.40    00:00:22.93
Symbol table sort                     0    00:00:00.63    00:00:02.55
Pass 2                               89    00:00:01.45    00:00:05.82
Symbol table output                  19    00:00:00.14    00:00:00.41
Psect synopsis output                 1    00:00:00.02    00:00:00.02
Cross-reference output                0    00:00:00.00    00:00:00.00
Assembler run totals                400    00:00:09.34    00:00:40.91

The working set limit was 1050 pages.
30728 bvtes (61 pages) of virtual memory were used to buffer the intermediate code.
There were 30 pages of symbol table space allocated to hold 439 non-local and 45 local symbols.
495 source lines were read in Pass 1, producing 15 object records in Pass 2.
31 pages of virtual memory were used to define 17 macros.

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

Macro library name                                    Macros defined
------------------                                    --------------
_$255$DUA28:[SYSLIB]SYSBLDMLB.MLB;1                                0
_$255$DUA28:[DCL.OBJ]DCL.MLB;1                                     8
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                                     0
_$255$DUA28:[SYSLIB]STARLET.MLB;2                                  3
TOTALS (all libraries)                                            11

549 GETS were required to define 11 macros.

There were no errors, warnings or information messages.

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

0074 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY