```
DDDDDDDDDDD         CCCCCCCCCCC  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
```

FILEID**CONVERT

```
  CCCCCCCC    000000    NN      NN  VV      VV  EEEEEEEEEE  RRRRRRRR    TTTTTTTTTT
  CCCCCCCC    000000    NN      NN  VV      VV  EEEEEEEEEE  RRRRRRRR    TTTTTTTTTT
CC           00      00  NN      NN  VV      VV  EE          RR      RR      TT
CC           00      00  NN      NN  VV      VV  EE          RR      RR      TT
CC           00      00  NNNN    NN  VV      VV  EE          RR      RR      TT
CC           00      00  NNNN    NN  VV      VV  EE          RR      RR      TT
CC           00      00  NN  NN  NN  VV      VV  EEEEEEEE    RRRRRRRR        TT
CC           00      00  NN  NN  NN  VV      VV  EEEEEEEE    RRRRRRRR        TT
CC           00      00  NN    NNNN  VV      VV  EE          RR  RR          TT
CC           00      00  NN    NNNN  VV      VV  EE          RR  RR          TT
CC           00      00  NN      NN    VV  VV    EE          RR    RR        TT          ....
CC           00      00  NN      NN    VV  VV    EE          RR    RR        TT          ....
  CCCCCCCC    000000    NN      NN      VV      EEEEEEEEEE  RR      RR      TT          ....
  CCCCCCCC    000000    NN      NN      VV      EEEEEEEEEE  RR      RR      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     1          .TITLE CONVERT - DCL VALUE CONVERSION AND EDITING SUBROUTINES
0000     2          .IDENT  'V04-000'
0000     3
0000     4  ;
0000     5  ;*****************************************************************************
0000     6  ;*                                                                           *
0000     7  ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     8  ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     9  ;*  ALL RIGHTS RESERVED.                                                     *
0000    10  ;*                                                                           *
0000    11  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
0000    12  ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
0000    13  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
0000    14  ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
0000    15  ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
0000    16  ;*  TRANSFERRED.                                                             *
0000    17  ;*                                                                           *
0000    18  ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
0000    19  ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
0000    20  ;*  CORPORATION.                                                             *
0000    21  ;*                                                                           *
0000    22  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
0000    23  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    24  ;*                                                                           *
0000    25  ;*                                                                           *
0000    26  ;*****************************************************************************
0000    27  ;
0000    28
0000    29  ;++
0000    30  ; FACILITY:     DCL
0000    31  ;
0000    32  ; ABSTRACT:     Value conversion and editing subroutines
0000    33  ;
0000    34  ; AUTHOR:       Peter George
0000    35  ;
0000    36  ; DATE:         01-MAR-1983
0000    37  ;
0000    38  ; MODIFIED BY:
0000    39  ;
0000    40  ;       V03-006 HWS0096         Harold Schutlz          27-Jul-1984
0000    41  ;               Fix HWS0082 to correctly handle null strings.
0000    42  ;
0000    43  ;       V03-005 HWS0082         Harold Schultz          19-Jul-1984
0000    44  ;               Fix DCL$CNVASCBIN to not return success if processing
0000    45  ;               a string of all uniary operators without any numbers.
0000    46  ;
0000    47  ;       V03-004 PCG0004         Peter George            11-Jan-1984
0000    48  ;               Add lowercase to edit.
0000    49  ;
0000    50  ;       V03-003 PCG0003         Peter George            15-Jul-1983
0000    51  ;               Use multi-national upcase algorithm.
0000    52  ;
0000    53  ;       V03-002 PCG0002         Peter George            15-Jun-1983
0000    54  ;               Fix bug in DCL$CBTA_HEXSTR.
0000    55  ;
0000    56  ;       V03-001 PCG0001         Peter George            09-May-1983
0000    57  ;               Add DCL$CBTA_HEXSTR.
```

```
                0000    58 ;--
```

```
                                                I 10

                        0000     60          .SBTTL  DECLARATIONS
                        0000     61 ;
                        0000     62 ; MACRO LIBRARY CALLS
                        0000     63 ;
                        0000     64          WRKDEF                          ; DEFINE COMMAND WORK AREA
                        0000     65          EDITDEF                         ; DEFINE EDIT FLAGS
                        0000     66          $CLIMSGDEF                      ; CLI MESSAGE DEFINITIONS
                        0000     67
                    00000000     68          .PSECT  DCL$ZCODE       BYTE,RD,NOWRT
                        0000     69
                        0000     70 ;
                        0000     71 ; OWN STORAGE:
                        0000     72 ;
                    58 44 4F     0000     73 CNVRDX: .ASCII  \ODX\                ; CONVERSION RADIX CONTROLS
                    08 0A 10     0003     74 RADIX:  .BYTE   16,10,8              ; CORRESPONDING RADIX VALUES
                                 0006     75 DIGIT_TABLE:                         ; HEX DIGITS
42 41 39 38 37 36 35 34 33 32 31 30  0006     76          .ASCII  '0123456789ABCDEF'
                    46 45 44 43  0012
```

```
                                                J 10
                     0016    78          .SBTTL  ASCII NUMERIC VALUE CONVERSION ROUTINE
                     0016    79 ;++
                     0016    80 ;
                     0016    81 ; DCL$CNVASCBIN - CONVERT ASCII STRING TO BINARY VALUE
                     0016    82 ;
                     0016    83 ; THIS ROUTINE IS CALLED TO CONVERT AN ASCII STRING TO A BINARY VALUE.
                     0016    84 ;
                     0016    85 ; INPUTS:
                     0016    86 ;
                     0016    87 ;       R1 = DEFAULT RADIX INDICATOR, 0=HEX,1=DECIMAL,2=OCTAL
                     0016    88 ;       R2/R3 = QUADWORD DESCRIPTOR OF VALUE
                     0016    89 ;
                     0016    90 ; OUTPUTS:
                     0016    91 ;
                     0016    92 ;       R0 = 0 IF SUCCESSFUL, NONZERO IF FAILURE
                     0016    93 ;       R1 = CONVERTED BINARY VALUE
                     0016    94 ;
                     0016    95 ;--
                     0016    96 DCL$CNVASCBIN::                              ; CONVERT IN SPECIFIED RADIX
          51    DD   0016    97          PUSHL   R1                          ; SAVE R1
     51   52    7D   0018    98          MOVQ    R2,R1                       ; SET UP THE ARGUMENT REGISTERS
        0119    30   001B    99          BSBW    DCL$TRIM                    ; TRIM AND UPCASE THE STRING
     52   51    7D   001E   100          MOVQ    R1,R2                       ; RETURN STRING TO R2/R3
          51  8ED0   0021   101          POPL    R1                          ; RESTORE R1
                     0024   102
                     0024   103 DCL$CNVNOEDIT::                             ; CONVERT NUMERIC DECIMAL RADIX
          7E    D4   0024   104          CLRL    -(SP)                       ; SAVE ROOM FOR ''NUMBER SEEN'' FLAG
        6243    9F   0026   105          PUSHAB  (R2)[R3]                    ; SAVE ADDRESS OF END OF STRING
          00    DD   0029   106          PUSHL   #0                          ; SET NO NEGATE FLAG
          50    52   D0   002B   107          MOVL    R2,R0                       ; TEST FOR ZERO LENGTH STRING
                56   13   002E   108          BEQL    70$                         ; IF IT IS THE NULL STRING-RETURN ZERO
       08 AE    01   D0   0030   109          MOVL    #1,8(SP)                    ; INIT. ''NUMBER SEEN'' FLAG
          63    25   91   0034   110 5$:      CMPB    #^A/%/,(R3)                 ; RADIX CHANGE OPERATOR?
                0E   12   0037   111          BNEQ    10$                         ; BR IF NO
                53   D6   0039   112          INCL    R3                          ; SKIP OVER OPERATOR
    C0 AF       03   83   003B   113          LOCC    (R3)+,#3,CNVRDX             ; FIND RADIX SPECIFIER
                44   13   0040   114          BEQL    70$                         ; BR IF NONE RECOGNIZED
          51    70   9E   0042   115          MOVAB   -(R0),R1                    ; SET RADIX INDICATOR
                ED   11   0045   116          BRB     5$                          ; LOOK FOR AN OTHER SET
          63    2B   91   0047   117 10$:     CMPB    #^A/+/,(R3)                 ; CHECK FOR UNIARY OPERATOR OF PLUS
                07   13   004A   118          BEQL    20$                         ; BR IF YES
          63    2D   91   004C   119          CMPB    #^A/-/,(R3)                 ; HOW ABOUT MINUS
                06   12   004F   120          BNEQ    30$                         ; BR IF NO
                6E   D6   0051   121          INCL    (SP)                        ; NEGATE THE NEGATIVE FLAG
                53   D6   0053   122 20$:     INCL    R3                          ; SKIP THE UNIARY OPERATOR
                DD   11   0055   123          BRB     5$                          ; LOOK FOR ANOTHER
       52   A8 AF41   9A   0057   124 30$:     MOVZBL  RADIX[R1],R2                ; SET ACTUAL RADIX OF CONVERSION
                50   7C   005C   125          CLRQ    R0                          ; START WITH RESULT AND WORK VALUE OF 0
       51   63   30   83   005E   126 40$:     SUBB3   #^A/0/,(R3),R1              ; GET NEXT NUMBER MINUS ASCII BIAS
                22   19   0062   127          BLSS    70$                         ; BR IF NOT A NUMERIC CHARACTER
          09    51   91   0064   128          CMPB    R1,#9                       ; WAS IT A NUMERIC DIGIT?
                08   15   0067   129          BLEQ    60$                         ; BR IF YES
          11    51   91   0069   130          CMPB    R1,#<^A/A/-^A/0/>           ; HEX RADIX CHARACTER?
                18   19   006C   131          BLSS    70$                         ; BR IF NO - TERMINATE THE CONVERT
          51    07   82   006E   132          SUBB    #<<^A/A/-^A/0/>-10>,R1      ; CONVERT TO BINARY VALUE
          52    51   91   0071   133 60$:     CMPB    R1,R2                       ; WITH IN THE RADIX?
                10   18   0074   134          BGEQ    70$                         ; BR IF OUT OF THE RANGE
```

```
                50    51    50    52    7A  0076    135          EMUL    R2,R0,R1,R0             ; FIND CURRENT TOTAL
                                08  AE    D4  0078    136          CLRL    8(SP)                   ; INDICATE AT LEAST ONE NUMBER PROCESSED
                                      53    D6  007E    137          INCL    R3                      ; POINT AT NEXT CHARACTER
                                  04  AE    53    D1  0080    138          CMPL    R3,4(SP)                ; TIME TO QUIT?
                                            D8    12  0084    139          BNEQ    40$                     ; BR IF NO
                                      51    50    D0  0086    140 70$:     MOVL    R0,R1                   ; SET RESULT
                                        03  8E    E9  0089    141          BLBC    (SP)+,80$               ; TEST NEGATE FLAG
                                      51    51    CE  008C    142          MNEGL   R1,R1                   ; NEGATE THE SOURCE
                                50    8E    53    C3  008F    143 80$:     SUBL3   R3,(SP)+,R0             ; SET NUMBER OF UN-PROCESSED BYTES
                                        03  8E    E9  0093    144          BLBC    (SP)+,90$               ; IF A NUMBER SEEN, FLAG OK AS IS
                                      50    01    D0  0096    145          MOVL    #1,R0                   ; IF NO NUMBER SEEN, UNCONDITIONALLY
                                                      0099    146                                          ;       SET TO FAILURE
                                                  05  0099    147 90$:     RSB                             ; BACK TO THE CALLER
                                                      009A    148
```

```
                                009A   150          .SBTTL  CONVERT BINARY TO 8-DIGIT HEX
                                009A   151 ;+
                                009A   152 ; DCL$CBTA_HEX - CONVERT BINARY TO HEX STRING
                                009A   153 ;
                                009A   154 ; CONVERT A BINARY NUMBER TO A HEX STRING OF EXACTLY 8 DIGITS.
                                009A   155 ;
                                009A   156 ; INPUTS:
                                009A   157 ;
                                009A   158 ;       R0 = NUMBER TO BE CONVERTED.
                                009A   159 ;
                                009A   160 ; OUTPUTS:
                                009A   161 ;
                                009A   162 ;       R1 = LENGTH OF CONVERTED VALUE.
                                009A   163 ;       R2 = ADDRESS OF CONVERTED VALUE.
                                009A   164 ;-
                                009A   165
                                009A   166 DCL$CBTA_HEX::
                51    04    D0  009A   167          MOVL    #4,R1                   ;CREATE DESCRIPTOR OF HEX STRING
                      50    DD  009D   168          PUSHL   R0                      ;
                52    5E    D0  009F   169          MOVL    SP,R2                   ;
                      03    10  00A2   170          BSBB    DCL$CBTA_HEXSTR         ;GET CONVERTED VALUE
                      8E    D5  00A4   171          TSTL    (SP)+                   ;RESTORE THE STACK
                            05  00A6   172          RSB                             ;RETURN
                                00A7   173
                                00A7   174 DCL$CBTA_HEXSTR::
                5E    51    C2  00A7   175          SUBL    R1,SP                   ;ALLOCATE SCRATCH BUFFER ON THE STACK
                      51    DD  00AA   176          PUSHL   R1                      ;SAVE INPUT LENGTH
          04 AE     62    51  28  00AC   177          MOVC    R1,(R2),4(SP)           ;COPY INPUT BUFFER
                      55 8ED0   00B1   178          POPL    R5                      ;RESTORE INPUT LENGTH
             52   F486  CA    D0  00B4   179          MOVL    WRK_L_EXPANDPTR(R10),R2 ;MARK POSITION IN OUTPUT BUFFER
             53   55    01    78  00B9   180          ASHL    #1,R5,R3                ;DOUBLE THE NUMBER OF INPUT BYTES
                                00BD   181
                      55    D7  00BD   182          DECL    R5                      ;START WITH LAST BYTE OF DATA
 50    6E45    04    04    EF  00BF   183 10$:      EXTZV   #4,#4,(SP)[R5],R0       ;GET SECOND NIBBLE OF BYTE
       50   FF3C  CF40        9A  00C5   184          MOVZBL  DIGIT_TABLE[R0],R0      ;CONVERT DIGIT TO ASCII
                  FF32'       30  00CB   185          BSBW    DCL$PUTCHAR             ;PUT THE CHAR IN THE OUTPUT BUFFER
 50    6E45    04    00    EF  00CE   186          EXTZV   #0,#4,(SP)[R5],R0       ;GET FIRST NIBBLE OF BYTE
       50   FF2D  CF40        9A  00D4   187          MOVZBL  DIGIT_TABLE[R0],R0      ;CONVERT DIGIT TO ASCII
                  FF23'       30  00DA   188          BSBW    DCL$PUTCHAR             ;PUT THE CHAR IN THE OUTPUT BUFFER
                 DF 55        F4  00DD   189          SOBGEQ  R5,10$                  ;BRANCH WHILE MORE DIGITS
                                00E0   190
   51    F486  CA    52        C3  00E0   191 20$:      SUBL3   R2,WRK_L_EXPANDPTR(R10),R1 ;GET DESCRIPTOR OF RESULTANT STRING
         F486  CA    52        D0  00E6   192          MOVL    R2,WRK_L_EXPANDPTR(R10) ;RESET EXPANSION BUFFER POINTER
      50    51    FF  8F        78  00EB   193          ASHL    #-1,R1,R0               ;GET THE NUMBER OF INPUT BYTES
                  5E    50        C0  00F0   194          ADDL    R0,SP                   ;POP SCRATCH BUFFER OFF THE STACK
                                  05  00F3   195          RSB                             ;
```

```
                                00F4    197          .SBTTL  CONVERT BINARY TO ZERO SUPRESSED ASCII
                                00F4    198 ;+
                                00F4    199 ; CBTA_DEC - CONVERT BINARY TO ASCII BASE TEN
                                00F4    200 ; CBTA_OCT - CONVERT BINARY TO ASCII BASE EIGHT
                                00F4    201 ;
                                00F4    202 ; THESE ROUTINES ARE CALLED TO CONVERT A BINARY NUMBER TO A LEFT JUSTIFIED, ZERO
                                00F4    203 ; SUPRESSED, ASCII STRING.  THE RESULTANT STRING IS PLACED IN THE EXPANSION
                                00F4    204 ; BUFFER AND THE EXPANSION POINTER IS UPDATED.
                                00F4    205 ;
                                00F4    206 ; INPUTS:
                                00F4    207 ;
                                00F4    208 ;       R0 = NUMBER TO BE CONVERTED.
                                00F4    209 ;
                                00F4    210 ; OUTPUTS:
                                00F4    211 ;
                                00F4    212 ;       R1 = LENGTH OF CONVERTED VALUE.
                                00F4    213 ;       R2 = ADDRESS OF CONVERTED VALUE.
                                00F4    214 ;-
                                00F4    215          .ENABL  LSB
                                00F4    216
                                00F4    217 DCL$CBTA_OCT::                              ;CONVERT BINARY TO ASCII BASE EIGHT
     53    08    9A             00F4    218          MOVZBL  #8,R3                      ;SET CONVERSION RADIX
           03    11             00F7    219          BRB     10$                        ;
                                00F9    220 DCL$CBTA_DEC::                              ;CONVERT BINARY TO ASCII BASE TEN
     53    0A    9A             00F9    221          MOVZBL  #10,R3                     ;SET CONVERSION RADIX
52   F486  CA    D0             00FC    222 10$:     MOVL    WRK_L_EXPANDPTR(R10),R2    ;MARK POSITION IN BUFFER
           50    D5             0101    223          TSTL    R0                         ;IS NUMBER NEGATIVE?
           0C    18             0103    224          BGEQ    15$                        ;NO, THEN SKIP
     54    50    CE             0105    225          MNEGL   R0,R4                      ;YES, THEN CHANGE SIGN AND SAVE VALUE
     50    2D    9A             0108    226          MOVZBL  #^A"-",R0                  ;GET NEGATIVE SIGN
              FEF2'  30         010B    227          BSBW    DCL$PUTCHAR                ;PUT IT IN THE EXPANSION BUFFER
     50    54    D0             010E    228          MOVL    R4,R0                      ;RECOVER VALUE
           0C    10             0111    229 15$:     BSBB    20$                        ;PUT ASCII CHARACTERS INTO BUFFER
51   F486  CA    52    C3       0113    230          SUBL3   R2,WRK_L_EXPANDPTR(R10),R1 ;GET DESCRIPTOR OF RESULTANT STRING
     F486  CA    52    D0       0119    231          MOVL    R2,WRK_L_EXPANDPTR(R10)    ;RESET EXPANSION BUFFER POINTER
                 05             011E    232          RSB
                                011F    233 ;
                                011F    234 ; RECURSIVE ROUTINE TO OUTPUT THE ASCII NUMBER, HIGH ORDER DIGITS FIRST
                                011F    235 ; WITHOUT ANY LEADING SPACES OR ZEROS.
                                011F    236 ;
           51    D4             011F    237 20$:     CLRL    R1                         ;CLEAR HIGH PART OF DIVIDEND
51   50    50    53    7B       0121    238          EDIV    R3,R0,R0,R1                ;ISOLATE NEXT DIGIT
     7E    51    30    C1       0126    239          ADDL3   #^A'0',R1,-(SP)            ;CONVERT DIGIT TO ASCII AND SAVE
           50    D5             012A    240          TSTL    R0                         ;ANY MORE DIGITS TO CONVERT?
           02    13             012C    241          BEQL    30$                        ;IF EQL NO
           EF    10             012E    242          BSBB    20$                        ;CONVERT NEXT DIGIT
           50  8ED0             0130    243 30$:     POPL    R0                         ;RETRIEVE NEXT CHARACTER
              FECA'  30         0133    244          BSBW    DCL$PUTCHAR                ;PUT CHARACTER IN EXPANSION BUFFER
                 05             0136    245          RSB
                                0137    246
                                0137    247          .DSABL  LSB
                                0137    248
```

```
                     0137   250          .SBTTL  EDIT THE STRING
                     0137   251 ;+
                     0137   252 ; DCL$EDIT  - EDIT THE STRING
                     0137   253 ;
                     0137   254 ; THIS ROUTINE IS CALLED TO EDIT THE INPUT STRING.  THE RESULTANT STRING
                     0137   255 ; IS RETURNED VIA THE INPUT DESCRIPTOR.
                     0137   256 ;
                     0137   257 ; INPUTS:
                     0137   258 ;
                     0137   259 ;       R0 = EDIT FLAGS
                     0137   260 ;       R1 = LENGTH OF INPUT STRING
                     0137   261 ;       R2 = ADDRESS OF INPUT STRING
                     0137   262 ;
                     0137   263 ; OUTPUTS:
                     0137   264 ;
                     0137   265 ;       R1 = LENGTH OF EDITED STRING
                     0137   266 ;       R2 = ADDRESS OF EDITED STRING
                     0137   267 ;-
                     0137   268
                     0137   269 DCL$TRIM::
 50   18   D0        0137   270          MOVL    #EDIT_M_UPCASE!EDIT_M_TRIM,R0   ; SET UPCASE AND TRIM FLAGS
      08   11        013A   271          BRB     DCL$EDIT
                     013C   272
                     013C   273 DCL$SQUEEZE::
 50   14   D0        013C   274          MOVL    #EDIT_M_UPCASE!EDIT_M_COLLAPSE,R0 ; SET UPCASE AND COLLAPSE FLAGS
      03   11        013F   275          BRB     DCL$EDIT
                     0141   276
                     0141   277 DCL$UPCASE::
 50   10   D0        0141   278          MOVL    #EDIT_M_UPCASE,R0               ; SET UPCASE FLAG
                     0144   279
                     0144   280 DCL$EDIT::
      51   D5        0144   281          TSTL    R1                              ; IS LENGTH ZERO
      1C   13        0146   282          BEQL    6$                              ; YES, THEN DONE
      50   D5        0148   283          TSTL    R0                              ; ARE NO FLAGS SET
      18   13        014A   284          BEQL    6$                              ; YES, THEN DONE
 7E   53   7D        014C   285          MOVQ    R3,-(SP)                        ; SAVE R3,R4
 53   52   D0        014F   286          MOVL    R2,R3                           ; COPY ADDRESS OF STRING
 54   52   D0        0152   287          MOVL    R2,R4                           ; COPY ADDRESS OF STRING
 7E   52   D0        0155   288          MOVL    R2,-(SP)                        ; COPY ADDRESS OF STRING
      7E   D4        0158   289          CLRL    -(SP)                           ; CLEAR FLAGS
                     015A   290
                     015A   291 ;
                     015A   292 ; IF CHARACTER IS A QUOTE THEN TOGGLE THE FLAG, AND COPY THE CHARACTER
                     015A   293 ;
 63   22   91        015A   294 5$:      CMPB    #^A/"/,(R3)                     ; IS CHARACTER A QUOTE
      08   12        015D   295          BNEQ    7$                              ; NO, THEN BRANCH
 6E   01   CC        015F   296          XORL    #1,(SP)                         ; YES, TOGGLE QUOTE FLAG
      79   11        0162   297          BRB     30$                             ; COPY THE CHARACTER
                     0164   298
   0098   31        0164   299 6$:      BRW     50$                             ; ALL DONE
                     0167   300
                     0167   301 ;
                     0167   302 ; IF INSIDE QUOTES, THEN COPY THE CHARACTER
                     0167   303 ;
  73 6E   E8        0167   304 7$:      BLBS    (SP),30$                        ; IF IN QUOTES, COPY THE CHAR
                     016A   305
                     016A   306 ;
```

```
                          016A    307 ; IF REMOVING COMMENTS, AND COMMENT CHARACTER WAS FOUND, THEN WE'RE ALL DONE.
                          016A    308 ;
05 50    00    E1         016A    309         BBC     #EDIT_V_UNCOMMENT,R0,10$        ; BRANCH IF NOT IGNORING COMMENTS
   63    21    91         016E    310         CMPB    #^A/!/,(R3)                     ; IS CHARACTER A COMMENT DELIMITER
         7A    13         0171    311         BEQL    42$                             ; YES, THEN DONE
                          0173    312
                          0173    313 ;
                          0173    314 ; IF WE ARE COMPRESSING BLANKS, THEN EITHER SET THE BLANK FLAG, OR
                          0173    315 ; REMOVE THE CHARACTER.  ALSO, SAVE THE ADDRESS OF THE FIRST BLANK
                          0173    316 ; IN CASE WE ARE TRIMMING BLANKS.
                          0173    317 ;
   63    20    91         0173    318 10$:    CMPB    #^X20,(R3)                      ; IS CHARACTER A BLANK
         05    13         0176    319         BEQL    15$                             ; YES, PROCESS IT
   63    09    91         0178    320         CMPB    #^X09,(R3)                      ; IS CHARACTER A TAB
         1E    12         017B    321         BNEQ    20$                             ; NO, THEN BRANCH
16 50    02    E0         017D    322 15$:    BBS     #EDIT_V_COLLAPSE,R0,18$         ; BRANCH IF COLLAPSING
04 50    03    E1         0181    323         BBC     #EDIT_V_TRIM,R0,16$             ; BRANCH IF NOT TRIMMING
0E 6E    02    E1         0185    324         BBC     #2,(SP),18$                     ; REMOVE LEADING BLANKS
07 50    01    E1         0189    325 16$:    BBC     #EDIT_V_COMPRESS,R0,17$         ; BRANCH IF NOT COMPRESSING
   63    20    90         018D    326         MOVB    #^X20,(R3)                      ; CONVERT SPACE OR TAB TO SPACE
03 6E    01    E2         0190    327         BBSS    #1,(SP),18$                     ; IF NOT FIRST BLANK, THEN SKIP IT
   84    63    90         0194    328 17$:    MOVB    (R3),(R4)+                      ; COPY THE CHARACTER
         53    D6         0197    329 18$:    INCL    R3                              ; POINT PAST THE BLANK
         4F    11         0199    330         BRB     40$                             ; LOOP
                          019B    331
                          019B    332 ;
                          019B    333 ; IF WE ARE UPCASING THE STRING, THEN NOW DO SO.
                          019B    334 ;
1F 50    04    E1         019B    335 20$:    BBC     #EDIT_V_UPCASE,R0,26$           ; BRANCH IF NOT UPCASING
61 8F    63    91         019F    336         CMPB    (R3),#^A/a/                     ; CHECK LOW LIMIT OF LOW RANGE
         38    1F         01A3    337         BLSSU   30$                             ; BR IF FAILED
7A 8F    63    91         01A5    338         CMPB    (R3),#^A/z/                     ; CHECK HIGH LIMIT OF LOW RANGE
         0C    1B         01A9    339         BLEQU   25$                             ; BR IF VALID CHARACTER
E0 8F    63    91         01AB    340         CMPB    (R3),#^XE0                      ; CHECK LOW LIMIT OF HIGH RANGE
         2C    1F         01AF    341         BLSSU   30$                             ; BR IF FAILED
FE 8F    63    91         01B1    342         CMPB    (R3),#^XFE                      ; CHECK HIGH LIMIT OF HIGH RANGE
         26    1A         01B5    343         BGTRU   30$                             ; BR IF FAILED
   63    20    8A         01B7    344 25$:    BICB    #^X20,(R3)                      ; UPCASE THE CHARACTER
         21    11         01BA    345         BRB     30$                             ;
                          01BC    346
         9C    11         01BC    347 29$:    BRB     5$
                          01BE    348
                          01BE    349 ;
                          01BE    350 ; IF WE ARE LOWERCASING THE STRING, THEN NOW DO SO.
                          01BE    351 ;
1B 50    05    E1         01BE    352 26$:    BBC     #EDIT_V_LOWERCASE,R0,30$        ; BRANCH IF NOT LOWERCASING
41 8F    63    91         01C2    353         CMPB    (R3),#^A/A/                     ; CHECK LOW LIMIT OF LOW RANGE
         15    1F         01C6    354         BLSSU   30$                             ; BR IF FAILED
5A 8F    63    91         01C8    355         CMPB    (R3),#^A/Z/                     ; CHECK HIGH LIMIT OF LOW RANGE
         0C    1B         01CC    356         BLEQU   27$                             ; BR IF VALID CHARACTER
C0 8F    63    91         01CE    357         CMPB    (R3),#^XC0                      ; CHECK LOW LIMIT OF HIGH RANGE
         09    1F         01D2    358         BLSSU   30$                             ; BR IF FAILED
DE 8F    63    91         01D4    359         CMPB    (R3),#^XDE                      ; CHECK HIGH LIMIT OF HIGH RANGE
         03    1A         01D8    360         BGTRU   30$                             ; BR IF FAILED
   63    20    88         01DA    361 27$:    BISB    #^X20,(R3)                      ; LOWERCASE THE CHARACTER
                          01DD    362
                          01DD    363 ;
```

```
                                01DD    364 ; COPY THE CHARACTER TO THE RESULT STRING.  SUBTRACT ONE FROM THE INPUT LENGTH
                                01DD    365 ; AND CONTINUE.
                                01DD    366 ;
            84    83    90    01DD    367 30$:    MOVB    (R3)+,(R4)+                     ; MOVE CHARACTER
            6E    02    CA    01E0    368         BICL    #2,(SP)                         ; CLEAR BLANK FLAG
            6E    04    C8    01E3    369         BISL    #4,(SP)                         ; SET FIRST NON-BLANK SEEN
         04 AE    54    D0    01E6    370         MOVL    R4,4(SP)                        ; SAVE ADDR OF LAST NON-BLANK
               CF 51    F5    01EA    371 40$:    SOBGTR  R1,29$                          ; LOOP TILL DONE
                                01ED    372
                                01ED    373 ;
                                01ED    374 ; GET THE RESULT STRING.  DO ANY TRIMMING THAT IS NECESSARY
                                01ED    375 ;
      51    54    52    C3    01ED    376 42$:    SUBL3   R2,R4,R1                        ; CALCULATE NEW LENGTH
            53    8E    7D    01F1    377         MOVQ    (SP)+,R3                        ; RESTORE STACK
         04 50    03    E1    01F4    378         BBC     #EDIT_V_TRIM,R0,45$             ; BRANCH IF NOT TRIMMING
      51    54    52    C3    01F8    379         SUBL3   R2,R4,R1                        ; CALCULATE SIZE AFTER TRIM
            53    8E    7D    01FC    380 45$:    MOVQ    (SP)+,R3                        ; RESTORE REGISTERS
                        05    01FF    381 50$:    RSB
                                0200    382
                                0200    383         .END
```

```
CNVRDX                          00000000 R      02
DCL$CBTA_DEC                    000000F9 RG     02
DCL$CBTA_HEX                    0000009A RG     02
DCL$CBTA_HEXSTR                 000000A7 RG     02
DCL$CBTA_OCT                    000000F4 RG     02
DCL$CNVASCBIN                   00000016 RG     02
DCL$CNVNOEDIT                   00000024 RG     02
DCL$EDIT                        00000144 RG     02
DCL$PUTCHAR                     ******** GX     02
DCL$SQUEEZE                     0000013C RG     02
DCL$TRIM                        00000137 RG     02
DCL$UPCASE                      00000141 RG     02
DIGIT_TABLE                     00000006 R      02
EDIT_B_FLAGS                    00000000
EDIT_M_COLLAPSE                 = 00000004
EDIT_M_TRIM                     = 00000008
EDIT_M_UPCASE                   = 00000010
EDIT_V_COLLAPSE                 = 00000002
EDIT_V_COMPRESS                 = 00000001
EDIT_V_LOWERCASE                = 00000005
EDIT_V_TRIM                     = 00000003
EDIT_V_UNCOMMENT                = 00000000
EDIT_V_UPCASE                   = 00000004
RADIX                           00000003 R      02
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_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_W_FLAGS                     FFFFFFF0
WRK_W_FLAGS2                    FFFFFFF2
WRK_W_IMGCHAN                   FFFFFFEE
WRK_W_PMPTLEN                   FFFFF99E
```

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	00000200 (512.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC BYTE

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	11	00:00:00.05	00:00:01.86
Command processing	87	00:00:00.62	00:00:08.80
Pass 1	161	00:00:04.00	00:00:13.69
Symbol table sort	0	00:00:00.36	00:00:00.78
Pass 2	68	00:00:01.05	00:00:03.77
Symbol table output	8	00:00:00.06	00:00:00.06
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	337	00:00:06.17	00:00:28.99

The working set limit was 900 pages.
20016 bytes (40 pages) of virtual memory were used to buffer the intermediate code.
There were 20 pages of symbol table space allocated to hold 267 non-local and 33 local symbols.
383 source lines were read in Pass 1, producing 14 object records in Pass 2.
16 pages of virtual memory were used to define 11 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYSLIB]SYSBLDMLB.MLB;1	0
_255DUA28:[DCL.OBJ]DCL.MLB;1	3
_255DUA28:[SYS.OBJ]LIB.MLB;1	0
_255DUA28:[SYSLIB]STARLET.MLB;2	3
TOTALS (all libraries)	6

346 GETS were required to define 6 macros.

There were no errors, warnings or information messages.

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

INTDEF
SDL

COMMAND
LIS

DESCRVAL
LIS

CONVERT
LIS

CLIMAC
MAR

CLIMSG
LIS

CONNECT
LIS

DCLPARSE
LIS

CHARMANIP
LIS

EXAMDEP
LIS

EXIT
LIS

INTIMAGES
MAR

DCXSTART
LIS

CLIGBL
LIS

DISALLOW
LIS

CLINT
LIS

CANCEL
LIS