
MMM MMM AAAAAAAAA CCCCCCCCCCC RRRRRRRRRRR 000000000
MMM MMM AAAAAAAAA CCCCCCCCCCC RRRRRRRRRRR 000000000
MMM MMM AAAAAAAAA CCCCCCCCCCC RRRRRRRRRRR 000000000
MMMMMM MMMMMM AAA AAA CCC RRR RRR OOO OOO
MMMMMM MMMMMM AAA AAA CCC RRR RRR OOO OOO
MMMMMM MMMMMM AAA AAA CCC RRR RRR OOO OOO
MMM MMM MMM AAA AAA CCC RRR RRR OOO OOO
MMM MMM MMM AAA AAA CCC RRR RRR OOO OOO
MMM MMM MMM AAA AAA CCC RRR RRR OOO OOO
MMM MMM AAA AAA CCC RRRRRRRRRRR OOO OOO
MMM MMM AAA AAA CCC RRRRRRRRRRR OOO OOO
MMM MMM AAA AAA CCC RRRRRRRRRRR OOO OOO
MMM MMM AAAAAAAAAAAAAAA CCC RRR RRR OOO OOO
MMM MMM AAAAAAAAAAAAAAA CCC RRR RRR OOO OOO
MMM MMM AAAAAAAAAAAAAAA CCC RRR RRR OOO OOO
MMM MMM AAA AAA CCC RRR RRR OOO OOO
MMM MMM AAA AAA CCC RRR RRR OOO OOO
MMM MMM AAA AAA CCC RRR RRR OOO OOO
MMM MMM AAA AAA CCCCCCCCCCC RRR RRR 000000000
MMM MMM AAA AAA CCCCCCCCCCC RRR RRR 000000000
MMM MMM AAA AAA CCCCCCCCCCC RRR RRR 000000000
MMM MMM AAA AAA CCCCCCCCCCC RRR RRR 000000000

FILEID**MACLIN

```
MM      MM    AAAAAA      CCCCCCCC  LL            IIIIII    NN      NN
MM      MM    AAAAAA      CCCCCCCC  LL            IIIIII    NN      NN
MMMM  MMMM   AA      AA  CC        LL              II      NN      NN
MMMM  MMMM   AA      AA  CC        LL              II      NN      NN
MM  MM  MM   AA      AA  CC        LL              II      NNNN    NN
MM  MM  MM   AA      AA  CC        LL              II      NNNN    NN
MM      MM   AA      AA  CC        LL              II      NN  NN  NN
MM      MM   AA      AA  CC        LL              II      NN  NN  NN
MM      MM   AAAAAAAAAA  CC        LL              II      NN    NNNN
MM      MM   AAAAAAAAAA  CC        LL              II      NN    NNNN
MM      MM   AA      AA  CC        LL              II      NN      NN      ....
MM      MM   AA      AA  CC        LL              II      NN      NN      ....
MM      MM   AA      AA    CCCCCCCC  LLLLLLLLLL    IIIIII    NN      NN      ....
MM      MM   AA      AA    CCCCCCCC  LLLLLLLLLL    IIIIII    NN      NN      ....

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  MAC$MACLIN      GET MACRO LINE
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:     VAX MACRO ASSEMBLER OBJECT LIBRARY
0000    31  ;
0000    32  ; ABSTRACT:
0000    33  ;
0000    34  ; The VAX-11 MACRO assembler translates MACRO-32 source code into object
0000    35  ; modules for input to the VAX-11 LINKER.
0000    36  ;
0000    37  ; ENVIRONMENT:  USER MODE
0000    38  ;
0000    39  ; AUTHOR: Benn Schreiber, CREATION DATE: 20-AUG-78
0000    40  ;
0000    41  ; MODIFIED BY:
0000    42  ;
0000    43  ;       V03-001 MCN0162         Maria del C. Nasr       02-Apr-1984
0000    44  ;               If the address of the source string for the %LOCATE function
0000    45  ;               is zero, then it is the null string, and we do not need to make
0000    46  ;               a character match.  It is assumed that the string was not found.
0000    47  ;
0000    48  ;       V03.00  MTR0007         Mike Rhodes     15-Mar-1982
0000    49  ;               Modify the lexical function routines for %EXTARCT and %LOCATE.
0000    50  ;               The routine MACLIN_LXEXT now treats the starting position and
0000    51  ;               length values as unsigned integers.  MACLIN_LXLOC now, will not
0000    52  ;               count the next link byte which caused some problems previously.
0000    53  ;               Both modifications fix SPR #11-43353.
0000    54  ;
0000    55  ;       V02.12  CNH0044         Chris Hume      20-Nov-1980
0000    56  ;               Fixed handling of macro expansion block continuation linkages.
0000    57  ;
```

```
0000     58 ;
0000     59 ;       V01.11  RN0026          R. Newland      13-Jan-1980
0000     60 ;               Don't put unecessary CHKL's into intermediate code when
0000     61 ;               not listing macro expansions.
0000     62 ;
0000     63 ;       V01.10  RN0023          R. Newland       3-Nov-1979
0000     64 ;               New message codes to get error messages from system
0000     65 ;               message file.
0000     66 ;
0000     67 ;       V01.09  RN0010          R. Newland      5-Sep-1979
0000     68 ;               Mulitpage MXB blocks
0000     69 ;
0000     70 ;       V01.08  RN0005          R. Newland      18-Aug-1978
0000     71 ;               Variable symbol storage and remove .ALIGN LONG statements
0000     72 ;
0000     73 ;       V01.07  RN0004          R. Newland      22-Jun-1979
0000     74 ;               Fix truncation error
0000     75 ;--
```

```
                     0000     77          .SBTTL  DECLARATIONS
                     0000     78 ;
                     0000     79 ; INCLUDE FILES:
                     0000     80 ;
                     0000     81
                     0000     82 ;
                     0000     83 ; MACROS:
                     0000     84 ;
                     0000     85
                     0000     86          $MAC_GENVALDEF                  ;DEFINE GENERAL VALUES
                     0000     87          $MAC_MNBDEF                     ;DEFINE MNB OFFSETS
                     0008     88          $MAC_MTXDEF                     ;DEFINE MACRO SPECIAL OPERATORS
                     0008     89          $MAC_CTLFLGDEF                  ;DEFINE CONTROL FLAGS
                     0008     90          $MAC_SYMBLKDEF                  ;DEFINE SYMBOL BLOCK OFFSETS
                     0000     91          $MAC_INPBLKDEF                  ;DEFINE INPUT BLOCK OFFSETS
                     003C     92          $MAC_INTCODDEF                  ;DEFINE INT. BUFFER CODES
                     003C     93          $MACMSGDEF                      ; Define message codes
                     003C     94
                     003C     95 ;
                     003C     96 ; LOCAL DATA
                     003C     97 ;
                     003C     98
                 00000000     99          .PSECT  MAC$RO_DATA,NOWRT,NOEXE,GBL,LONG
                     0000    100
                     0000    101 MACLIN_MTXT_TAB:                        ;TABLE OF SPECIAL MACRO OPERATOR BYTES
                       FE    0000    102          .BYTE   MTX$_TXTLNK,-      ;TEXT LINK TO NEW PAGE
                     0001    103                  MTX$_ARGMRK,-      ;DUMMY ARGUMENT MARKER
                       FF    0001    104                  MTX$_LXLEN,-       ;LEXICAL LENGTH FUNCTION
                       FD    0002    105                  MTX$_LXLOC,-       ;LEXICAL LOCATE FUNCTION
                    FC FB    0003    106                  MTX$_LXEXT         ;LEXICAL EXTRACT FUNCTION
                 00000005    0005    107 LEN$K_MTXTAB    =       .-MACLIN_MTXT_TAB ;LENGTH OF TABLE
                     0005    108
                     0005    109          .ALIGN  LONG
                     0008    110 MACLIN_DISP_TAB:                        ;DISPATCH TABLE FOR SPECIAL OPERATORS
                     0008    111          .ADDRESS MACLIN_TXTLNK,-           ;TEXT LINK
             00000149'  0008    112                   MACLIN_ARGMRK,-           ;ARGUMENT MARKER
             0000011B'  000C    113                   MACLIN_LXLEN,-            ;LENGTH FUNCTION
             00000154'  0010    114                   MACLIN_LXLOC,-            ;LOCATE
        00000168'000001CB'  0014    115                   MACLIN_LXEXT              ;EXTRACT
                     001C    116
                 00000000    117          .PSECT  MAC$RO_CODE_MAC,NOWRT,GBL,LONG
```

```
                     0000   119          .SBTTL  MAC$GET_IRC_LIN GET NEXT LINE OF .IRPC
                     0000   120
                     0000   121 ;++
                     0000   122 ; FUNCTIONAL DESCRIPTION:
                     0000   123 ;
                     0000   124 ;       THIS ROUTINE IS CALLED TO GET THE NEXT TEXT LINE WHILE
                     0000   125 ;       EXPANDING AN INDEFINITE REPEAT CHARACTER 'MACRO'.  IF
                     0000   126 ;       THE END OF THE TEXT IS NOT THE NEXT LINE, MAC$GET_IRC_LIN
                     0000   127 ;       WILL RETURN THE NEXT LINE.  IF IT IS THE END, AND THERE
                     0000   128 ;       ARE STILL MORE CHARACTERS IN THE INDEFINTE REPEAT STRING,
                     0000   129 ;       THE REPEAT WILL BE RESTARTED WITH THE NEXT CHARACTER.  IF
                     0000   130 ;       THERE ARE NO MORE CHARACTERS, THE INPUT CONTEXT IS POPPED,
                     0000   131 ;       TERMINATING THE INDEFINITE REPEAT.
                     0000   132 ;
                     0000   133 ;--
                     0000   134
                     0000   135 MAC$GET_IRC_LIN::
 50   0000'CF    D0  0000   136          MOVL    W^MAC$GL_INPUTP,R0      ;POINT TO INPUT CONTEXT BLOCK
   51    04 A0    D0  0005   137 10$:     MOVL    INP$L_NXTL(R0),R1       ;GET NEXT LINE POINTER
         5E    10  0009   138          BSBB    MAC_RPT_END_CHK         ;SEE IF END OF REPEAT TEXT
                     000B   139                                          ;  GO TO MAC$GET_MAC_LIN IF
                     000B   140                                          ;  NOT END. RETURN ONLY IF END
   51    3F A0    D0  000B   141          MOVL    INP$K_IRPSIZ+3(R0),R1   ; Get pointer into IRPC string
      7E    81    9A  000F   142          MOVZBL  (R1)+,-(SP)             ;GET NEXT CHARACTER
   3F A0    51    D0  0012   143          MOVL    R1,INP$K_IRPSIZ+3(R0)   ; Set new pointer
         51 8ED0  0016   144          POPL    R1                      ;GET CHARACTER BACK
   3E A0    51    90  0019   145          MOVB    R1,INP$K_IRPSIZ+2(R0)   ; Set new argument
         03    12  001D   146          BNEQ    40$                     ;IF NEQ THERE IS MORE TO DO
      FFDE'    31  001F   147          BRW     MAC$POP_INPUT           ;NO--THIS IS THE END--POP INPUT CONTEXT
04 A0    14 A0    D0  0022   148 40$:     MOVL    INP$L_RPTCNT(R0),INP$L_NXTL(R0) ;RESET NEXT LINE POINTER
         DC    11  0027   149          BRB     10$                     ;REPEAT THE INDEFINITE REPEAT
                     0029   150                                          ;WITH THE NEW CHARACTER
      005C    31  0029   151 50$:     BRW     MAC$GET_MAC_LIN         ;BRANCH AID TO GET TO FAR PLACES!
```

```
                                002C   153          .SBTTL  MAC$GET_IRP_LIN GET NEXT LINE OF .IRP
                                002C   154
                                002C   155 ;++
                                002C   156 ; FUNCTIONAL DESCRIPTION:
                                002C   157 ;
                                002C   158 ;       THIS ROUTINE IS CALLED TO GET THE NEXT TEXT LINE WHILE IN
                                002C   159 ;       AN INDEFINITE REPEAT EXPANSION.  IF THE END OF THE TEXT
                                002C   160 ;       IS NOT THE NEXT LINE, MAC$GET_MAC_LIN WILL RETURN THE
                                002C   161 ;       NEXT LINE.  IF IT IS THE END, AND THERE IS ANOTHER REAL
                                002C   162 ;       ARGUMENT, THE REPEAT IS RESTARTED WITH THE NEW REAL
                                002C   163 ;       ARGUMENT.   IF THERE ARE NO MORE REAL ARGUMENTS, THE
                                002C   164 ;       INPUT CONTEXT IS POPPED, TERMINATING THE INDEFINITE
                                002C   165 ;       REPEAT.
                                002C   166 ;
                                002C   167 ;--
                                002C   168
                                002C   169 MAC$GET_IRP_LIN::
    50    0000'CF    D0    002C   170          MOVL    W^MAC$GL_INPUTP,R0      ;GET POINTER TO INPUT BLOCK
      51    04 A0    D0    0031   171 10$:     MOVL    INP$L_NXTL(R0),R1       ;GET NEXT LINE POINTER
                32    10    0035   172          BSBB    MAC_RPT_END_CHK         ;SEE IF END OF REPEAT RANGE
                                0037   173                                  ;  GO TO MAC$GET_MAC_LIN IF NOT
                                0037   174                                  ;  RETURN ONLY IF END
      51    3C A0    DE    0037   175          MOVAL   INP$K_IRPSIZ(R0),R1     ; Point to real arg queue header
      51    00 B1    0F    003B   176          REMQUE  @(R1),R1                ;GET THE ADDR OF NEXT REAL ARG
                03    1C    003F   177          BVC     40$                     ;IF V-CLEAR THEN WE GOT AN ENTRY
                 FFBC'    31    0041   178          BRW     MAC$POP_INPUT           ;NO MORE ARGS--POP INPUT CONTEXT
  1D A0    08 A1    9E    0044   179 40$:     MOVAB   8(R1),INP$L_ARGS(R0)    ;POINT TO ARG STRING IN INPUT BLOCK
  04 A0    14 A0    D0    0049   180          MOVL    INP$L_RPTCNT(R0),INP$L_NXTL(R0) ;RESET NEXT LINE POINTER
                E1    11    004E   181          BRB     10$                     ;PERFORM NEXT REPEAT
```

```
                     0050   183          .SBTTL  MAC$GET_RPT_LIN GET NEXT LINE OF REPEAT
                     0050   184
                     0050   185 ;++
                     0050   186 ; FUNCTIONAL DESCRIPTION:
                     0050   187 ;
                     0050   188 ;       THIS ROUTINE IS CALLED TO GET THE NEXT TEXT LINE WHILE IN
                     0050   189 ;       A .REPT DIRECTIVE EXPANSION.  IF THE END OF THE TEXT IS
                     0050   190 ;       NOT THE NEXT LINE, MAC$GET_MAC_LIN WILL RETURN THE NEXT
                     0050   191 ;       LINE.  IF IT IS THE END, THE REPEAT COUNT IS DECREMENTED,
                     0050   192 ;       AND IF IT HAS GONE TO ZERO, THE INPUT CONTEXT IS POPPED,
                     0050   193 ;       THUS TERMINATING THE REPEAT.  IF IT HAS NOT GONE TO ZERO,
                     0050   194 ;       THE NEXT LINE POINTER IS RESET AND THE REPEAT BLOCK IS
                     0050   195 ;       EXPANDED AGAIN.
                     0050   196 ;
                     0050   197 ;--
                     0050   198
                     0050   199 MAC$GET_RPT_LIN::
 50    0000'CF    D0  0050   200          MOVL    W^MAC$GL_INPUTP,R0      ;GET POINTER TO INPUT BLOCK
   51    04 A0    D0  0055   201 10$:     MOVL    INP$L_NXTL(R0),R1       ;GET NEXT LINE POINTER
            0E    10  0059   202          BSBB    MAC_RPT_END_CHK         ;SEE IF END OF REPEAT RANGE
                      005B   203                                          ;  GO TO MAC$GET_MAC_LIN IF NOT
                      005B   204                                          ;  RETURN ONLY IF END
      03 14 A0    F4  005B   205 30$:     SOBGEQ  INP$L_RPTCNT(R0),40$    ;DEC. REPEAT COUNT AND GO IF NOT DONE
         FF9E'    31  005F   206          BRW     MAC$POP_INPUT           ;DONE--POP INPUT CONTEXT
04 A0    21 A0    9E  0062   207 40$:     MOVAB   INP$K_BLKSIZ(R0),INP$L_NXTL(R0) ; Reset text pointer
            EC    11  0067   208          BRB     10$                     ;PERFORM NEXT REPEAT
                      0069   209
                      0069   210 ;++
                      0069   211 ; FUNCTIONAL DESCRIPTION:
                      0069   212 ;
                      0069   213 ;       THIS ROUTINE PICKS UP THE LINE LENGTH AND DETERMINES IF
                      0069   214 ;       IT IS REALLY A LINE OR THE END OF TEXT MARKER.  IF IT IS
                      0069   215 ;       REALLY A LINE, THE RETURN ADDRESS IS POPPED AND WE DISPATCH
                      0069   216 ;       TO MAC$GET_MAC_LIN.  RETURN ONLY IF END OF TEXT.
                      0069   217 ;
                      0069   218 ; INPUTS:
                      0069   219 ;
                      0069   220 ;       R1      POINTER TO WORD CONTAINING BYTE COUNT
                      0069   221 ;       R0      POINTER TO INPUT CONTEXT BLOCK
                      0069   222 ;
                      0069   223 ; OUTPUTS:
                      0069   224 ;
                      0069   225 ;       RETURN IS POPPED AND BRANCH IS MADE TO MAC$GET_MAC_LIN UNLESS
                      0069   226 ;       LINE LENGTH IS -1 (END OF TEXT MARKER).  TEXT LINKS ARE HANDLED.
                      0069   227 ;
                      0069   228 ;--
                      0069   229
                      0069   230 MAC_RPT_END_CHK:
      52    81    32  0069   231          CVTWL   (R1)+,R2                ;GET LENGTH OF LINE
            15    18  006C   232          BGEQ    10$                     ;IF GEQ CERTAINLY NOT END OF TEXT
53    52    01    A1  006E   233          ADDW3   #1,R2,R3                ;IS IT END OF TEXT (-1)?
            13    13  0072   234          BEQL    20$                     ;IF EQL YES
FFFE  8F    52    B1  0074   235          CMPW    R2,#MTX$_TXTLNK!^XFF00  ;NO--IS IT A TEXT LINK?
            08    12  0079   236          BNEQ    10$                     ;IF NEQ NO--TEXT LINE WITH
                      007B   237                                          ;  SPECIAL OPERATORS
      51    81    D0  007B   238          MOVL    (R1)+,R1                ;YES--GET POINTER TO NEXT PAGE
      51    08    C0  007E   239          ADDL2   #MXB$K_BLKSIZ,R1        ; Skip header infomation
```

```
E6   11  0081   240          BRB     MAC_RPT_END_CHK         ;CONTINUE CHECKING
8E   D5  0083   241 10$:     TSTL    (SP)+                   ;CLEAR RETURN FROM STACK
01   11  0085   242          BRB     MAC$GET_MAC_LIN         ;AND GO RETURN NEXT TEXT LINE
     05  0087   243 20$:     RSB                             ;RETURN ONLY IF END OF TEXT
```

```
                                0088   245          .SBTTL  MAC$GET_MAC_LIN EXPAND NEXT LINE OF CURRENT MACRO
                                0088   246
                                0088   247 ;++
                                0088   248 ; FUNCTIONAL DESCRIPTION:
                                0088   249 ;
                                0088   250 ;       THIS ROUTINE IS CALLED BY MAC$GETCHR WHEN IT IS TIME FOR A
                                0088   251 ;       NEW LINE.  THE NEXT LINE OF THE MACRO IS EXPANDED INTO
                                0088   252 ;       MAC$AB_LINEBF.
                                0088   253 ;
                                0088   254 ;--
                                0088   255
                                0088   256 MAC$GET_MAC_LIN::
            11F8 8F     BB      0088   257          PUSHR   #^M<R3,R4,R5,R6,R7,R8,R12> ;SAVE REGISTERS
        5C  0000'CF     D0      008C   258          MOVL    W^MAC$GL_INPUTP,R12     ;POINT TO CURRENT INPUT BLOCK
        56  0000'CF     9E      0091   259          MOVAB   W^MAC$AB_LINEBF,R6      ;POINT TO THE LINE BUFFER
  0000'CF   56          D0      0096   260          MOVL    R6,W^MAC$GL_LINEPT      ;RESET LINE POINTER
        55  03E8 8F     3C      009B   261          MOVZWL  #INP$K_BUFSIZ,R5        ;LOAD MAX CHARACTER COUNT
          57    04 AC    D0      00A0   262          MOVL    INP$L_NXTL(R12),R7      ;GET POINTER TO NEXT MACRO LINE
            58  87      32      00A4   263 10$:     CVTWL   (R7)+,R8                ;GET LENGTH OF LINE
                50      14      00A7   264          BGTR    MACLIN_COPY_LIN         ;IF GTR NOT SPECIAL LINE
                18      13      00A9   265          BEQL    30$                     ;IF EQL BLANK (NULL LINE)
        50      58      01      A1     00AB   266          ADDW3   #1,R8,R0                ;NEGATIVE--ARE WE DONE?
                        0F      13     00AF   267          BEQL    20$                     ;IF EQL YES--DO END OF MACRO PROCESSING
        FFFE 8F         58      B1     00B1   268          CMPW    R8,#MTX$_TXTLNK!^XFF00  ;NO--SPECIAL TEXT LINK?
                        0E      12     00B6   269          BNEQ    MACLIN_NEXT_CHO         ;IF NEQ NO--IT IS LINE WITH SPECIAL OPERATOR
                57      87      D0     00B8   270          MOVL    (R7)+,R7                ;YES--GET LINK TO NEXT
                57      08      C0     00BB   271          ADDL2   #MXB$K_BLKSIZ,R7        ; Skip header information
                        E4      11     00BE   272          BRB     10$                     ;CONTINUE
                    01F4        31     00C0   273 20$:     BRW     MACLIN_END_TEXT         ;ELSE END OF MACRO TEXT
                    0184        31     00C3   274 30$:     BRW     MACLIN_LINE_END         ;BRANCH AID
                                00C6   275 MACLIN_NEXT_CHO:
    58      FFFF8000 8F         CA     00C6   276          BICL2   #^X<^C<7FFF>>,R8        ;TRIM LINE LENGTH TO 15 BITS
                                00CD   277 ;
                                00CD   278 ; HERE TO GET NEXT CHARACTER AND DISPATCH TO PROCESS IT
                                00CD   279 ;
                                00CD   280 MACLIN_NEXT_CHR:
                58      D5      00CD   281          TSTL    R8                      ;DONE WITH LINE?
                25      15      00CF   282          BLEQ    30$                     ;IF LEQ YES
                58      D7      00D1   283          DECL    R8                      ;COUNT NEXT BYTE
            5A  87      90      00D3   284          MOVB    (R7)+,R10               ;GET NEXT CHARACTER
                08      14      00D6   285          BGTR    10$                     ;BRANCH IF DEFINITLY NOT SPECIAL
            05  5A      3A      00D8   286          LOCC    R10,#LEN$K_MTXTAB,-     ;SEE IF IT IS A SPECIAL OPERATOR
            0000'CF             00DB   287                  W^MACLIN_MTXT_TAB       ;...
                09      12      00DE   288          BNEQ    20$                     ;IF NEQ YES--GO DISPATCH TO PROCESSOR
                55      D7      00E0   289 10$:     DECL    R5                      ;COUNT THE CHARACTER
                E9      19      00E2   290          BLSS    MACLIN_NEXT_CHR         ;DON'T STORE IF THERE IS NO ROOM
            86  5A      90      00E4   291          MOVB    R10,(R6)+               ;STORE THE CHARACTER
                E4      11      00E7   292          BRB     MACLIN_NEXT_CHR         ;GO FOR MORE
                                00E9   293 ;
                                00E9   294 ; DISPATCH TO SPECIAL OPERATOR PROCESSING ROUTINE
                                00E9   295 ;
    51  00000000'8F         C2     00E9   296 20$:     SUBL2   #MACLIN_MTXT_TAB,R1     ;GET INDEX INTO DISPATCH TABLE
          0008'CF41         DD     00F0   297          PUSHL   W^MACLIN_DISP_TAB[R1]   ;GET ROUTINE ADDRESS
                            05     00F5   298          RSB                             ;RETURN TO EXECUTE ROUTINE
                    0151    31     00F6   299 30$:     BRW     MACLIN_LINE_END         ;DO END OF LINE PROCESSING
                                00F9   300 ;
                                00F9   301 ; HERE IF LINE CONTAINS NO SPECIAL OPERATORS OR TEXT LINKS. JUST COPY
```

```
                                00F9   302 ; THE TEXT TO THE OUTPUT BUFFER
                                00F9   303 ;
                                00F9   304 MACLIN_COPY_LIN:
      7E   000003E8 8F    58    C3  00F9   305          SUBC3   R8,#INP$K_BUFSIZ,-(SP)  ; Compute bytes that will remain in buffer
                          08    18  0101   306          BGEQ    10$                     ; If GEQ bytes will fit in buffer
           66   67   03E8 8F    28  0103   307          MOVC3   #INP$K_BUFSIZ,(R7),(R6) ; Copy only bytes that will fit in buffer
                          04    11  0109   308          BRB     20$
                                0108   309 10$:
                66   67   58    28  0108   310          MOVC3   R8,(R7),(R6)            ; Copy line to buffer
                                010F   311 20$:
                          55 8ED0   010F   312          POPL    R5                      ; Set number of bytes remaining
                     56   58    C0  0112   313          ADDL2   R8,R6                   ; Point past end of line in buffer
                     57   58    C0  0115   314          ADDL2   R8,R7                   ;UPDATE INPUT POINTER
                        012F    31  0118   315          BRW     MACLIN_LINE_END         ;DC END OF LINE PROCSSING
```

```
                          011B   317          .SBTTL  PROCESS ARGUMENT MARKERS AND TEXT LINKS
                          011B   318
                          011B   319 ;
                          011B   320 ; MTX$_ARGMRK -- ARGUMENT MARKER.  GET THE ARG NUMBER AND STORE THE
                          011B   321 ; REPLACEMENT STRING FOR THAT ARGUMENT.
                          011B   322 ;
                          011B   323 MACLIN_ARGMRK:
      5A   87   9A        011B   324          MOVZBL  (R7)+,R10                ;GET ARGUMENT NUMBER
           58   D7        011E   325          DECL    R8                       ;COUNT THE ARG MARKER
5A    19 AC4A   D0        0120   326          MOVL    INP$L_ARGS-4(R12)[R10],R10 ;GET ARG DESCRIPTOR POINTER
           A6   13        0125   327          BEQL    MACLIN_NEXT_CHR          ;IF EQL NO REPLACEMENT STRING
      54   8A   3C        0127   328          MOVZWL  (R10)+,R4                ;GET LENGTH OF REPLACEMENT STRING
           A1   13        012A   329          BEQL    MACLIN_NEXT_CHR          ;IF EQL NO REPLACEMENT STRING
      55   54   D1        012C   330          CMPL    R4,R5                    ;STRING TOO LONG?
           07   15        012F   331          BLEQ    10$                      ;IF LEQ NO
      54   55   D0        0131   332          MOVL    R5,R4                    ;YES--JUST COPY WHAT WILL FIT
   55    FF   8F   98     0134   333          CVTBL   #-1,R5                   ;AND DON'T COPY ANY MORE AFTER
      55   54   C2        0138   334 10$:     SUBL2   R4,R5                    ;UPDATE THE COUNTER
           55   DD        013B   335          PUSHL   R5                       ;SAVE THE COUNTER
66    6A   54   28        013D   336          MOVC3   R4,(R10),(R6)            ;COPY ARG INTO LINE BUFFER
      56   53   D0        0141   337          MOVL    R3,R6                    ;UPDATE BUFFER POINTER
           55 8ED0         0144   338          POPL    R5                       ;RETRIEVE COUNT
           84   11         0147   339 20$:     BRB     MACLIN_NEXT_CHR          ;NEXT CHARACTER
                           0149   340 ;
                           0149   341 ; MTX$_TXTLNK -- TEXT LINK.  GET THE LINK TO THE NEXT PAGE.
                           0149   342 ;
                           0149   343 MACLIN_TXTLNK:
      57   87   D0        0149   344          MOVL    (R7)+,R7                 ;LINK TO NEXT TEXT
      57   08   C0        014C   345          ADDL2   #MXB$K_BLKSIZ,R7         ; Skip header information
           58   D6        014F   346          INCL    R8                       ;UNCOUNT LINK BYTE THAT WAS COUNTED
         FF79   31        0151   347          BRW     MACLIN_NEXT_CHR          ; and continue (don't count byte tho)
```

```
                                                I 9
                     0154   349          .SBTTL  PROCESS LEXICAL LENGTH OPERATOR
                     0154   350
                     0154   351 ;
                     0154   352 ; MTX$_LXLEN -- LEXICAL LENGTH FUNCTION.  THE NEXT BYTE IS AN ARGUMENT
                     0154   353 ; DESCRIPTOR.  IT CAN BE 0 (SPECIAL CANCEL), A LITERAL STRING, OR AN
                     0154   354 ; ARGUMENT MARKER.  PROCESS IT.
                     0154   355 ;
                     0154   356 MACLIN_LXLEN:
   0201    30        0154   357          BSBW    MACLIN_GET_STR          ;GET THE STRING ARGUMENT
 08 50     E9        0157   358          BLBC    R0,10$                  ;BRANCH IF THERE WAS AN ERROR
52   54    D0        015A   359          MOVL    R4,R2                   ;POSITION FOR DECOUT
   023A    30        015D   360          BSBW    MACLIN_DECOUT           ;OUTPUT THE STRING LENGTH
     03    11        0160   361          BRB     20$                     ;GO DO REST OF LINE
   0233    30        0162   362 10$:     BSBW    MACLIN_ZEROUT           ;OUTPUT ZERO ON ERROR
   FF65    31        0165   363 20$:     BRW     MACLIN_NEXT_CHR         ;DO REST OF LINE
```

```
                     0168   365          .SBTTL  PROCESS LEXICAL EXTRACT OPERATOR
                     0168   366
                     0168   367 ;
                     0168   368 ; MTX$_LXEXT -- LEXICAL EXTRACT FUNCTION.  THERE ARE THREE ARGUMENTS:
                     0168   369 ; 1) A STARTING POSITION, 2) THE LENGTH OF THE STRING TO EXTRACT, AND
                     0168   370 ; 3) THE STRING TO EXTRACT FROM.
                     0168   371 ;
                     0168   372 MACLIN_LXEXT:
      0188    30     0168   373          BSBW    MACLIN_GET_VAL          ;GET THE STARTING POSITION
    51 50     E9     016B   374          BLBC    R0,60$                  ;BRANCH IF ERROR
       51     D5     016E   375          TSTL    R1                      ;IS IT AN UNSIGNED INTEGER?
       03     18     0170   376          BGEQ    10$                     ;YES -- SAVE IT
 51    51     CE     0172   377          MNEGL   R1,R1                   ;NO -- GET ITS ABS VALUE
       51     DD     0175   378 10$:     PUSHL   R1                      ;SAVE STARTING POSITION
      017C    30     0177   379          BSBW    MACLIN_GET_VAL          ;GET THE LENGTH
    49 50     E9     017A   380          BLBC    R0,110$                 ;BRANCH IF ERROR
       51     D5     017D   381          TSTL    R1                      ;IS IT AN UNSIGNED INTEGER?
       03     18     017F   382          BGEQ    20$                     ;YES -- SAVE IT
 51    51     CE     0181   383          MNEGL   R1,R1                   ;NO -- GET ITS ABS VALUE
       51     DD     0184   384 20$:     PUSHL   R1                      ;SAVE THE LENGTH
      01CF    30     0186   385          BSBW    MACLIN_GET_STR          ;GET THE STRING
    35 50     E9     0189   386          BLBC    R0,100$                 ;BRANCH IF ERROR
       54     D5     018C   387          TSTL    R4                      ;WAS STRING NULL?
       31     13     018E   388          BEQL    100$                    ;IF EQL YES
                     0190   389 ;
                     0190   390 ; HERE WITH R10 POINTING TO STRING, R4 HAS THE LENGTH OF THE STRING,
                     0190   391 ; 0(SP) HAS THE LENGTH OF DESIRED EXTRACTION, AND 4(SP) HAS THE POSITION
                     0190   392 ; FROM WHICH TO START EXTRACTING
                     0190   393 ;
          53 8ED0    0190   394 30$:     POPL    R3                      ;GET LENGTH OF STRING TO EXTRACT
          52 8ED0    0193   395          POPL    R2                      ;GET STARTING POSITION
       54    52     D1     0196   396          CMPL    R2,R4                   ;IS STARTPOS PAST STRING END?
             2D     14     0199   397          BGTR    MACLIN_LEX_EXIT         ;IF GTR YES--RESULT IS NULL
 51    53    52     C1     019B   398          ADDL3   R2,R3,R1                ;NO--FIGURE END POSITION
       54    51     D1     019F   399          CMPL    R1,R4                   ;PAST END OF STRING?
             04     15     01A2   400          BLEQ    40$                     ;IF LEQ NO
 53    54    52     C3     01A4   401          SUBL3   R2,R4,R3                ;YES--FIGURE # CHARS WE CAN GET
       55    53     D1     01A8   402 40$:     CMPL    R3,R5                   ;IS THERE ROOM FOR WHOLE STRING?
             03     15     01AB   403          BLEQ    50$                     ;IF LEQ YES
       53    55     D0     01AD   404          MOVL    R5,R3                   ;NO--JUST STORE WHAT WE CAN
 7E    55    53     C3     01B0   405 50$:     SUBL3   R3,R5,-(SP)             ;UPDATE COUNT OF SPACE LEFT AND STACK
66   6A42   53     28     01B4   406          MOVC3   R3,(R10)[R2],(R6)       ;COPY STRING INTO LINE BUFFER
       56    53     D0     01B9   407          MOVL    R3,R6                   ;UPDATE POINTER
             55 8ED0    01BC   408          POPL    R5                      ;RETRIEVE UPDATED COUNT
             07     11     01BF   409 60$:     BRB     MACLIN_LEX_EXIT         ;GO FINISH UP
                     01C1   410 ;
                     01C1   411 ; CLEAN TWO WORDS FROM STACK AND EXIT
                     01C1   412 ;
       5E    08     C0     01C1   413 100$:    ADDL2   #2*4,SP                 ;CLEAR TWO LONGWORDS FROM STACK
             02     11     01C4   414          BRB     MACLIN_LEX_EXIT         ;EXIT
                     01C6   415 ;
                     01C6   416 ; CLEAN ONE WORD FROM STACK AND EXIT
                     01C6   417 ;
             8E     D5     01C6   418 110$:    TSTL    (SP)+                   ;CLEAR ONE LONGWORD FROM STACK
                     01C8   419 MACLIN_LEX_EXIT:                            ;BRANCH AID
          FF02     31     01C8   420          BRW     MACLIN_NEXT_CHR         ;GO DO REST OF LINE
```

```
                                01CB   422          .SBTTL  PROCESS LEXICAL LOCATE OPERATOR
                                01CB   423
                                01CB   424 ;
                                01CB   425 ; MTX$_LXLOC -- LEXICAL LOCATE FUNCTION.  THERE ARE TWO MANDATORY ARGUMENTS:
                                01CB   426 ; 1) THE SUBSTRING TO LOCATE, 2) THE STRING TO LOCATE IN.  THE THIRD
                                01CB   427 ; ARGUMENT, THE STARTING POSITION, IS OPTIONAL.  IT WILL EITHER BE THERE
                                01CB   428 ; OR THERE WILL BE AN MTX$_NOMORE BYTE IN ITS PLACE.
                                01CB   429 ;
                                01CB   430 MACLIN_LXLOC:
                         018A   30   01CB   431          BSBW    MACLIN_GET_STR          ;GET THE STRING
                       73 50    E9   01CE   432          BLBC    R0,50$                  ;BRANCH IF ERROR
                                01D1   433 ;
                                01D1   434 ; SAVE POINTERS FOR LATER USE
                                01D1   435 ;
                   51   0000'CF   9E   01D1   436          MOVAB   W^MAC$AB_TMPBUF,R1      ;POINT TO SAVE AREA
                        81   54   D0   01D6   437          MOVL    R4,(R1)+                ;STORE LENGTH OF STRING
                        81   5A   D0   01D9   438          MOVL    R10,(R1)+               ;AND ITS ADDRESS
                                01DC   439 ;
                                01DC   440 ; PICK UP PARAMS FOR SECOND ARGUMENT (STRING TO LOCATE IN)
                                01DC   441 ;
                         0179   30   01DC   442          BSBW    MACLIN_GET_STR          ;GET THE SECOND ARGUMENT
                       62 50    E9   01DF   443          BLBC    R0,50$                  ;BRANCH IF ERROR
                   51   0008'CF   9E   01E2   444          MOVAB   W^MAC$AB_TMPBUF+8,R1    ;POINT TO TEMP BUFFER
                        81   54   D0   01E7   445          MOVL    R4,(R1)+                ;SAVE ITS LENGTH
                        81   5A   D0   01EA   446          MOVL    R10,(R1)+               ;AND LOCATION
                                01ED   447 ;
                                01ED   448 ; GET OPTIONAL STARTPOS ARGUMENT IF PRESENT
                                01ED   449 ;
                 EC 8F   67   91   01ED   450 10$:    CMPB    (R7),#MTX$_NOMORE       ;IS THERE NO STARTPOS ARG
                         08   12   01F1   451          BNEQ    15$                     ;IF NEQ IT MAY BE THERE
                         57   D6   01F3   452          INCL    R7                      ;ITS NOT THERE--BUMP POINTER
                         58   D7   01F5   453          DECL    R8                      ;DEC. COUNTER
                         51   D4   01F7   454          CLRL    R1                      ;USE STARTPOS OF 0
                         13   11   01F9   455          BRB     20$                     ;AND GO FINISH UP
                   FE 8F   67   91   01FB   456 15$:    CMPB    (R7),#MTX$_TXTLNK       ;IS IT A TEXT LINK?
                         07   12   01FF   457          BNEQ    17$                     ;IF NEQ NO
                         57   D6   0201   458          INCL    R7                      ;YES--SKIP THE BYTE, BUT DONT COUNT IT!
                      00E1   30   0203   459          BSBW    MACLIN_LINK_CHO         ;LINK TO NEXT PAGE
                         E5   11   0206   460          BRB     10$                     ;AND TRY AGAIN
                      00EB   30   0208   461 17$:    BSBW    MACLIN_GET_VAL          ;NO--GET THE VALUE
                    36 50    E9   020B   462          BLBC    R0,50$                  ;BRANCH IF ERROR
                50   0000'CF   9F   020E   463 20$:    MOVAB   W^MAC$AB_TMPBUF,R0      ;POINT TO TEMP BUFFER
                         55   DD   0213   464          PUSHL   R5                      ;SAVE R5
                     54   80   7D   0215   465          MOVQ    (R0)+,R4                ;R4=SUBSTR LENGTH, R5=SUBSTR ADDR
                     52   80   7D   0218   466          MOVQ    (R0)+,R2                ;R2=STRING LENGTH, R3=STRING ADDR
                         54   DD   021B   467          PUSHL   R4                      ;SAVE LENGTH OF SUBSTRING
                         52   DD   021D   468          PUSHL   R2                      ;SAVE LENGTH OF STRING
                         53   D5   021F   469          TSTL    R3                      ;IS SOURCE STRING ADDR NULL?
                         14   13   0221   470          BEQL    30$                     ;YES, ASSUME NOT FOUND
                     52   51   C2   0223   471          SUBL2   R1,R2                   ;DECREASE STRING LENGTH BY STARTPOS
              6341   52   65   54   39   0226   472          MATCHC  R4,(R5),R2,(R3)[R1]     ;PERFORM THE LOCATE
                         09   12   022C   473          BNEQ    30$                     ;IF NEQ NOT FOUND
                     52   8E   52   C3   022E   474          SUBL3   R2,(SP)+,R2             ;FIGURE STARTING POSITION
                     52   8E   C2   0232   475          SUBL2   (SP)+,R2                ;R2 HAS STARTING POSITION
                         05   11   0235   476          BRB     40$                     ;JOIN COMMON CODE
                              52 8ED0   0237   477 30$:    POPL    R2                      ;NOT FOUND--RETURN STRING LENGTH
                              8E   D5   023A   478          TSTL    (SP)+                   ;CLEAR STACK
```

```
   55 8ED0  023C   479 40$:     POPL    R5                      ;RESTORE R5
0158   30  023F   480           BSBW    MACLIN_DECOUT           ;OUTPUT THE DECIMAL VALUE
  84   11  0242   481           BRB     MACLIN_LEX_EXIT
           0244   482 ;
           0244   483 ; ERROR--OUTPUT A ZERO
           0244   484 ;
0151   30  0244   485 50$:     BSBW    MACLIN_ZEROUT           ;OUTPUT A ZERO
FE83   31  0247   486           BRW     MACLIN_NEXT_CHR         ;GO DO REST OF LINE
```

```
                                024A   488          .SBTTL  END OF LINE/END OF TEXT PROCESSING
                                024A   489
                                024A   490 ;
                                024A   491 ; LINE IS DONE
                                024A   492 ;
                                024A   493 MACLIN_LINE_END:
                          55    D5   024A   494          TSTL    R5                      ;WAS LINE TOO LONG?
                          10    18   024C   495          BGEQ    10$                     ;IF GEQ NO
                                024E   496          $INTOUT_LW INT$_ERR,<#MAC$_LINTOOLONG,#0> ; Yes--error
                          03    11   025C   497          brb     15$
                      86      0D   025E   498 10$:     MOVB    #CR,(R6)+               ;END LINE WITH CR
                55    56  00000001'8F   C3   0261   499 15$:     SUBL3   #MAC$AB_LINEBF+1,R6,R5  ;FIGURE LINE LENGTH
                    0000'CF   55    D0   0269   500          MOVL    R5,W^MAC$GL_LINELN      ;SAVE FOR LATER
                      04 AC   57    D0   026E   501 20$:     MOVL    R7,INP$L_NXTL(R12)      ;SET NEXT LINE POINTER
                      0B 0005'CF    E8   0272   502          BLBS    W^LST$G_MACROXPAN+SYM$L_VAL,30$ ;BRANCH IF LISTING MACRO EXPANSIONS
                      06 0005'CF    E8   0277   503          BLBS    W^LST$G_MACROBIN+SYM$L_VAL,30$ ;BRANCH IF LISTING MACRO BINARY
                         0000'CF    D5   027C   504          TSTL    W^MAC$GL_LIST_LVL       ;CHECK THE LISTING LEVEL
                                23    15   0280   505          BLEQ    40$                     ;BRANCH IF NOT LISTING
                                0282   506 ;
                                0282   507 ; MACRO LINE WILL BE LISTED.  EMIT LINE TO INTERMEDIATE BUFFER
                                0282   508 ;
                                0282   509 30$:
                                0282   510          $INTOUT_X INT$_CHKL             ; Align listing
                      50    55    04    C1   0288   511          ADDL3   #4,R5,R0        ; Figure size or record
                               FD71'  30   028C   512          BSBW    MAC$INTOUT_N            ;SET UP TO STORE THE LINE
                   FF A9     FF  8F   90   028F   513          MOVB    #-1,-1(R9)              ;SET SPECIAL MACRO LINE FLAG
                      89     16   90   0294   514          MOVB    #INT$_MACL,(R9)+        ;STORE CODE
                      89     55   B0   0297   515          MOVW    R5,(R9)+                ;STORE LENGTH OF LINE
                69    0000'CF    55   28   029A   516          MOVC3   R5,W^MAC$AB_LINEBF,(R9) ;COPY LINE INTO INT. BUFFER
                         59    53   D0   02A0   517          MOVL    R3,R9                   ;UPDATE THE POINTER
                         2D    11   02A3   518          BRB     MACLIN_EXIT             ;FINISH UP
                                02A5   519 ;
                                02A5   520 ; MACRO LINE WILL NOT BE LISTED.
                                02A5   521 ;
                      50    04    9A   02A5   522 40$:     MOVZBL  #4,R0                   ;SET TO STORE 4 BYTES
                               FD55'  30   02A8   523          BSBW    MAC$INTOUT_N            ;SET UP FOR THEM
                   FF A9     FF  8F   90   02AB   524          MOVB    #-1,-1(R9)              ;SET SPECIAL MACRO LINE FLAG
                      89     16   90   02B0   525          MOVB    #INT$_MACL,(R9)+        ;STORE CODE
                         89    B4   02B3   526          CLRW    (R9)+                   ;ZERO LENGTH LINE
                         1B    11   02B5   527          BRB     MACLIN_EXIT             ;GO FINISH UP
                                02B7   528 ;
                                02B7   529 ; WE REACHED THE END OF THE MACRO TEXT
                                02B7   530 ;
                                02B7   531 MACLIN_END_TEXT:
                      0000'CF   D5   02B7   532          TSTL    W^MAC$GL_IF_LEVEL       ;CHECK THE IF-LEVEL
                         10    13   02BB   533          BEQL    10$                     ;SHOULD BE AT LEVEL 0
                                02BD   534          $INTOUT_LW INT$_WRN,<#MAC$_UNTERMCOND,W^MAC$GL_LINEPT> ; No--error
                            FD30'  30   02CD   535 10$:     BSBW    MAC$POP_INPUT           ;POP AN INPUT LEVEL
                         03    11   02D0   536          BRB     MACLIN_EX0              ;SKIP R10 SET
                                02D2   537 MACLIN_EXIT:
                      5A    20    9A   02D2   538          MOVZBL  #BLNK,R10               ;RESET R10
                                02D5   539 MACLIN_EX0:
0000'CF               0000'CF   9E   02D5   540          MOVAB   W^MAC$AB_LINEBF,W^MAC$GL_ERRPTx ;SET ERROR TOKEN POINTER
                      11F8 8F   BA   02DC   541          POPR    #^M<R3,R4,R5,R6,R7,R8,R12> ;RESTORE REGISTERS
                                05   02E0   542          RSB
```

```
                     02E1    544          .SBTTL  CHECK FOR LINK BYTE IN LEXICAL OPERATORS
                     02E1    545
                     02E1    546 ;++
                     02E1    547 ; FUNCTIONAL DESCRIPTION:
                     02E1    548 ;
                     02E1    549 ;       THE LEXICAL OPERATORS MAY HAVE OPERANDS THAT SPAN ACROSS
                     02E1    550 ;       PAGES.  IF THIS IS THE CASE, THERE WILL BE A TXTLNK BYTE.
                     02E1    551 ;       THIS ROUTINE CHECKS FOR THAT, AND IF IT IS THE CASE, SPANS
                     02E1    552 ;       TO THE NEW PAGE.
                     02E1    553 ;
                     02E1    554 ; INPUTS:
                     02E1    555 ;
                     02E1    556 ;       R10     THE BYTE IN QUESTION
                     02E1    557 ;       R8      LINE COUNT
                     02E1    558 ;       R7      LINE POINTER
                     02E1    559 ;
                     02E1    560 ; OUTPUTS:
                     02E1    561 ;
                     02E1    562 ;       R10,R8,R7 UPDATED IF R10 WAS A TXTLNK
                     02E1    563 ;
                     02E1    564 ;--
                     02E1    565
                     02E1    566          .ENABL  LSB
                     02E1    567 MACLIN_LINK_CHK:
FE 8F    5A    91    02E1    568          CMPB    R10,#MTX$_TXTLNK        ;IS IT A TEXTLINK?
         07    12    02E5    569          BNEQ    10$                     ;IF NEQ NO
                     02E7    570 MACLIN_LINK_CHO:
   57    87    D0    02E7    571          MOVL    (R7)+,R7                ;YES--LINK TO NEXT PAGE
   57    08    C0    02EA    572          ADDL2   #MXB$K_BLKSIZ,R7        ; Skip header information
         05    02ED    573          RSB                             ;RETURN
0000'CF  00    FB    02EE    574 10$:     CALLS   #0,W^MAC$ERR_INTERN     ;REPORT INTERNAL CONFUSION
      FD0A'    31    02F3    575          BRW     MAC$ABORT_PASS1         ;TERMINATE PASS 1
                     02F6    576          .DSABL  LSB
```

```
                     02F6   578          .SBTTL  GET VALUE FOR LEXICAL OPERATOR
                     02F6   579
                     02F6   580 ;++
                     02F6   581 ; FUNCTIONAL DESCRIPTION:
                     02F6   582 ;
                     02F6   583 ;       THIS ROUTINE PICKS UP THE NEXT ARGUMENT (IT MUST BE NUMERIC)
                     02F6   584 ;       AND RETURNS IT TO THE CALLER.
                     02F6   585 ;
                     02F6   586 ; OUTPUTS:
                     02F6   587 ;
                     02F6   588 ;       R0      0       ERROR
                     02F6   589 ;               1       OK
                     02F6   590 ;       R1              VALUE IF R0=1
                     02F6   591 ;
                     02F6   592 ;--
                     02F6   593
                     02F6   594          .ENABL  LSB
                     02F6   595 MACLIN_GET_VAL:
          58   D7   02F6   596          DECL    R8                      ;COUNT NEXT BYTE
          50   D4   02F8   597          CLRL    R0                      ;CLEAR IN CASE CANCEL BYTE
       5A 87   9A   02FA   598 5$:      MOVZBL  (R7)+,R10               ;GET ARG DESCRIPTOR BYTE
          58   13   02FD   599          BEQL    60$                     ;IF EQL SPECIAL CANCEL BYTE
                    02FF   600 MACLIN_GET_VLO:
    EE 8F 5A   91   02FF   601          CMPB    R10,#MTX$_LITVAL        ;IS IT A LITERAL VALUE?
          0B   13   0303   602          BEQL    10$                     ;IF EQL YES
    ED 8F 5A   91   0305   603          CMPB    R10,#MTX$_SYMADR        ;IS IT AN ABSOLUTE SYMBOL ADDRESS?
          0A   13   0309   604          BEQL    20$                     ;IF EQL YES
        FFD3   30   030B   605          BSBW    MACLIN_LINK_CHK         ;NO--SEE IF A TEXT LINK
          EA   11   030E   606          BRB     5$                      ;TRY AGAIN
                    0310   607 ;
                    0310   608 ; IT WAS A LITERAL VALUE--PICK IT UP
                    0310   609 ;
       51   87   D0   0310   610 10$:    MOVL    (R7)+,R1                ;GET THE VALUE
            3C   11   0313   611          BRB     50$                     ;EXIT WITH SUCCESS
                    0315   612 ;
                    0315   613 ; IT WAS A SYMBOL ADDRESS
                    0315   614 ;
       51   87   D0   0315   615 20$:    MOVL    (R7)+,R1                ;GET THE SYMBOL BLOCK ADDRESS
            51   DD   0318   616          PUSHL   R1                      ;SAVE SYMBOL ADDRESS
 18 09 A1   00   E0   031A   617          BBS     #SYM$V_DEF,SYM$W_FLAG(R1),30$ ;BRANCH IF SYMBOL DEFINED
52  00000000'E5   9E   031F   618          MOVAB   L^MAC$AB_LINEBF(R5),R2  ;SET UP LINE POINTER
                    0326   619          $INTOUT_LW INT$_ERR,<#MAC$_UNDEFSYM,R2> ; Send error to pass 2
       51   6E   D0   0334   620          MOVL    (SP),R1                 ;RESET SYMBOL BLOCK ADDRESS
 0E 09 A1   04   E0   0337   621 30$:    BBS     #SYM$V_ABS,SYM$W_FLAG(R1),40$ ;BRANCH IF ABSOLUTE
                    033C   622          $INTOUT_LW INT$_ERR,<#MAC$_SYMNOTABS,R2> ; No--error to pass 2
            51 8ED0   034A   623 40$:    POPL    R1                      ;GET SYMBOL ADDRESS AGAIN
    51   05 A1   D0   034D   624          MOVL    SYM$L_VAL(R1),R1        ;GET THE SYMBOL VALUE
       58   04   C2   0351   625 50$:    SUBL2   #4,R8                   ;COUNT 4 BYTES GONE
       50   01   D0   0354   626          MOVL    #1,R0                   ;RETURN SUCCESS
                 05   0357   627 60$:    RSB
                    0358   628          .DSABL  LSB
```

```
                     0358   630          .SBTTL  GET STRING FOR LEXICAL OPERATOR
                     0358   631
                     0358   632
                     0358   633 ;++
                     0358   634 ; FUNCTIONAL DESCRIPTION:
                     0358   635 ;
                     0358   636 ;       THIS ROUTINE RETURNS THE ADDRESS AND LENGTH OF THE NEXT ARGUMENT
                     0358   637 ;       FOR A LEXICAL OPERATOR (MUST BE STRING ARGUMENT).
                     0358   638 ;
                     0358   639 ; OUTPUTS:
                     0358   640 ;
                     0358   641 ;       R0      0       CANCEL WAS SEEN
                     0358   642 ;               1       OK
                     0358   643 ;       R4              LENGTH OF STRING
                     0358   644 ;       R10             ADDRESS
                     0358   645 ;
                     0358   646 ;--
                     0358   647
                     0358   648 MACLIN_GET_STR:
           58   D7   0358   649          DECL    R8                      ;COUNT NEXT BYTE
           50   D4   035A   650          CLRL    R0                      ;CLEAR IN CASE CANCEL BYTE
       5A  87   9A   035C   651 10$:     MOVZBL  (R7)+,R10               ;GET ARG DESCRIPTOR BYTE
           36   13   035F   652          BEQL    50$                     ;IF EQL SPECIAL CANCEL BYTE
   EF  8F   5A   91   0361   653          CMPB    R10,#MTX$_LITSTR        ;IS IT A LITERAL STRING?
           1E   13   0365   654          BEQL    30$                     ;IF EQL YES
   FF  8F   5A   91   0367   655          CMPB    R10,#MTX$_ARGMRK        ;NO--IS IT AN ARG MARKER?
           05   13   036B   656          BEQL    20$                     ;IF EQL YES
         FF71   30   036D   657          BSBW    MACLIN_LINK_CHK         ;NO--SEE IF A TEXT LINK
           EA   11   0370   658          BRB     10$                     ;AND TRY AGAIN
                     0372   659 ;
                     0372   660 ; ARGUMENT MARKER
                     0372   661 ;
           58   D7   0372   662 20$:     DECL    R8                      ;COUNT ARG NUMBER BYTE
       5A  87   9A   0374   663          MOVZBL  (R7)+,R10               ;GET ARG NUMBER
           54   D4   0377   664          CLRL    R4                      ;CLEAR LENGTH IN CASE NULL STRING
   5A  19 AC4A   D0   0379   665          MOVL    INP$L_ARGS-4(R12)[R10],R10 ;GET DESCRIPTOR POINTER
           14   13   037E   666          BEQL    40$                     ;IF EQL NULL STRING
       54  8A   3C   0380   667          MOVZWL  (R10)+,R4               ;GET THE LENGTH OF THE STRING
                     0383   668                                          ;AND LEAVE R10 POINTING AT STRING
           0F   11   0383   669          BRB     40$                     ;GO RETURN
                     0385   670 ;
                     0385   671 ; LITERAL STRING
                     0385   672 ;
       54  87   3C   0385   673 30$:     MOVZWL  (R7)+,R4                ;GET LENGTH OF LITERAL STRING
       5A  57   D0   0388   674          MOVL    R7,R10                  ;POINT R10 TO START OF STRING
       57  54   C0   038B   675          ADDL2   R4,R7                   ;SKIP THE STRING
       58  54   C2   038E   676          SUBL2   R4,R8                   ;COUNT THE STRING
       58  02   C2   0391   677          SUBL2   #2,R8                   ;COUNT THE COUNT WORD
       50  01   D0   0394   678 40$:     MOVL    #1,R0                   ;RETURN GOODLY
                 05   0397   679 50$:     RSB                             ;DONE
```

```
                     0398   681          .SBTTL  OUTPUT DECIMAL NUMBER
                     0398   682
                     0398   683 ;++
                     0398   684 ; FUNCTIONAL DESCRIPTION:
                     0398   685 ;
                     0398   686 ;       THIS ROUTINE OUTPUTS A DECIMAL NUMBER TO THE LINE BUFFER BEING
                     0398   687 ;       CREATED.
                     0398   688 ;
                     0398   689 ; INPUTS:
                     0398   690 ;
                     0398   691 ;       R2      NUMBER TO OUTPUT
                     0398   692 ;       R5      LINE COUNTER
                     0398   693 ;       R6      LINE POINTER
                     0398   694 ;
                     0398   695 ; OUTPUTS:
                     0398   696 ;
                     0398   697 ;       R5,R6   UPDATED
                     0398   698 ;       NUMBER OUTPUT
                     0398   699 ;
                     0398   700 ;--
                     0398   701
                     0398   702 MACLIN_ZEROUT:
            52   D4  0398   703          CLRL    R2                      ;OUTPUT A ZERO
                     039A   704 MACLIN_DECOUT:
            53   D4  039A   705          CLRL    R3                      ;CLEAR HIGH WORD
     50     52   52  0A   7B  039C   706 10$:    EDIV    #10,R2,R2,R0            ;DIVIDE BY 10
            50   30  80   03A1   707          ADDB2   #^A/0/,R0               ;CONVERT TO ASCII
            50   DD  03A4   708          PUSHL   R0                      ;AND STACK IT
            52   D5  03A6   709          TSTL    R2                      ;ANY MORE TO DO?
            02   13  03A8   710          BEQL    20$                     ;IF EQL NO--UNSTACK AND OUTPUT
            F0   10  03AA   711          BSBB    10$                     ;YES--RECURSE TILL 0
            50 8ED0  03AC   712 20$:    POPL    R0                      ;GET THE DIGIT BACK
            55   D7  03AF   713          DECL    R5                      ;SEE IF ROOM TO STORE IT
            03   19  03B1   714          BLSS    30$                     ;IF LSS NO
    86      50   90  03B3   715          MOVB    R0,(R6)+                ;STORE THE DIGIT
            05   0386   716 30$:    RSB                             ;RECURSE OR RETURN
                 03B7   717
                 03B7   718          .END
```

```
$COUNT            = 0000003B
ARG$K_SIZE        = 000003E8
AUD$K_SIZE        = 00000010
BLNK              = 00000020
CHR$M_COMMA_CR    = 00000020
CHR$M_ILL_CHR     = 00000040
CHR$M_NUM_BER     = 00000010
CHR$M_SPA_MSK     = 00000001
CHR$M_SYM_CH1     = 00000008
CHR$M_SYM_CHR     = 00000004
CHR$M_SYM_DLM     = 00000002
CHR$V_COMMA_CR    = 00000005
CHR$V_CVTLWC      = 00000061
CHR$V_ILL_CHR     = 00000006
CHR$V_NOCVT       = 0000007F
CHR$V_NUM_BER     = 00000004
CHR$V_SPA_MSK     = 00000000
CHR$V_SYM_CH1     = 00000003
CHR$V_SYM_CHR     = 00000002
CHR$V_SYM_DLM     = 00000001
CNT               = 00000002
CR                = 0000000D
ERR               = 00000001
FF                = 0000000C
FLG$M_ALLCHR      = 00000001
FLG$M_BOL         = 00000002
FLG$M_CHKLPND     = 00100000
FLG$M_COMPEXPR    = 00000004
FLG$M_CONT        = 00000008
FLG$M_CRF         = 40000000
FLG$M_CRSEEN      = 00000001
FLG$M_DATRPT      = 00000010
FLG$M_DBGOUT      = 00004000
FLG$M_DLIMSTR     = 00008000
FLG$M_ENDMCH      = 00000020
FLG$M_EVALEXPR    = 00000040
FLG$M_EXPOPT      = 00000080
FLG$M_EXTERR      = 00010000
FLG$M_EXTWRN      = 00020000
FLG$M_FIRSTLN     = 00000200
FLG$M_IFSTAT      = 00800000
FLG$M_IIF         = 00400000
FLG$M_INSERT      = 00000100
FLG$M_IRPC        = 20000000
FLG$M_LEXOP       = 00000002
FLG$M_LSTXST      = 00000200
FLG$M_MAC2COL     = 00000800
FLG$M_MACL        = 00000800
FLG$M_MACLTB      = 08000000
FLG$M_MACTXT      = 00010000
FLG$M_MEBLST      = 00001000
FLG$M_MOREARG     = 00002000
FLG$M_MOREINP     = 00000008
FLG$M_NEWPND      = 00000400
FLG$M_NOREF       = 01000000
FLG$M_NTYPEPC     = 00000020
FLG$M_NULCHR      = 00040000
FLG$M_OBJXST      = 00200000
FLG$M_OPNDCHK     = 00000100
FLG$M_OPRND       = 00002000
FLG$M_OPTVFLIDX   = 00001000
FLG$M_ORDLST      = 00020000
FLG$M_P2          = 00004000
FLG$M_RPTIRP      = 10000000
FLG$M_SEQFIL      = 02000000
FLG$M_SKAN        = 00008000
FLG$M_SPECOP      = 00000004
FLG$M_SPLALL      = 04000000
FLG$M_STOIMF      = 00040000
FLG$M_SYM2COL     = 00000400
FLG$M_TOCFLG      = 00080000
FLG$M_UPAFLG      = 00000010
FLG$M_UPDFIL      = 00000080
FLG$M_UPMARG      = 00000040
FLG$M_XCRF        = 80000000
FLG$V_ALLCHR      = 00000000
FLG$V_BOL         = 00000001
FLG$V_CHKLPND     = 00000014
FLG$V_COMPEXPR    = 00000002
FLG$V_CONT        = 00000003
FLG$V_CRF         = 0000001E
FLG$V_CRSEEN      = 00000020
FLG$V_DATRPT      = 00000004
FLG$V_DBGOUT      = 0000002E
FLG$V_DLIMSTR     = 0000002F
FLG$V_ENDMCH      = 00000005
FLG$V_EVALEXPR    = 00000006
FLG$V_EXPOPT      = 00000007
FLG$V_EXTERR      = 00000030
FLG$V_EXTWRN      = 00000031
FLG$V_FIRSTLN     = 00000029
FLG$V_IFSTAT      = 00000017
FLG$V_IIF         = 00000016
FLG$V_INSERT      = 00000008
FLG$V_IRPC        = 0000001D
FLG$V_LEXOP       = 00000021
FLG$V_LSTXST      = 00000009
FLG$V_MAC2COL     = 0000002B
FLG$V_MACL        = 0000000B
FLG$V_MACLTB      = 0000001B
FLG$V_MACTXT      = 00000010
FLG$V_MEBLST      = 0000000C
FLG$V_MOREARG     = 0000002D
FLG$V_MOREINP     = 00000023
FLG$V_NEWPND      = 0000000A
FLG$V_NOREF       = 00000018
FLG$V_NTYPEPC     = 00000025
FLG$V_NULCHR      = 00000032
FLG$V_OBJXST      = 00000015
FLG$V_OPNDCHK     = 00000028
FLG$V_OPRND       = 0000000D
FLG$V_OPTVFLIDX   = 0000002C
FLG$V_ORDLST      = 00000011
FLG$V_P2          = 0000000E
FLG$V_RPTIRP      = 0000001C
FLG$V_SEQFIL      = 00000019
FLG$V_SKAN        = 0000000F
FLG$V_SPECOP      = 00000022
FLG$V_SPLALL      = 0000001A
FLG$V_STOIMF      = 00000012
FLG$V_SYM2COL     = 0000002A
FLG$V_TOCFLG      = 00000013
FLG$V_UPAFLG      = 00000024
FLG$V_UPDFIL      = 00000027
FLG$V_UPMARG      = 00000026
FLG$V_XCRF        = 0000001F
HASHSZ            = 0000007F
HYPHEN            = 0000002D
INP$B_ARGCT         0000001C
INP$K_BLKSIZ        00000021
INP$K_BUFSIZ      = 000003E8
INP$K_IRPSIZ        0000003C
INP$L_ARGS          0000001D
INP$L_GETL          00000008
INP$L_IFLVL         0000000C
INP$L_IFVAL         00000010
INP$L_LINK          00000000
INP$L_NXTL          00000004
INP$L_PAGP          00000018
INP$L_RPTCNT        00000014
INT$K_BUFSIZ      = 000013F4
INT$K_BUFWRN      = 00001390
INT$_ADD          = 00000001
INT$_AND          = 00000002
INT$_ASH          = 00000003
INT$_ASN          = 0000000C
INT$_AUGPC        = 0000000D
INT$_BDST         = 0000000E
INT$_CHKL         = 0000000F
INT$_DIV          = 00000004
INT$_END          = 00000010
INT$_EPT          = 00000011
INT$_ERR          = 00000012
INT$_ETX          = 00000013
INT$_FNEWL        = 00000014
INT$_ILG          = 00000000
INT$_INFO         = 0000003A
INT$_LGLAB        = 00000015
INT$_MACL         = 00000016
INT$_MUL          = 00000005
INT$_NEG          = 00000006
INT$_NEWL         = 00000017
INT$_NEWP         = 00000018
INT$_NOT          = 00000007
INT$_OP           = 00000019
INT$_OR           = 00000008
INT$_PRIL         = 0000001A
INT$_PRT          = 0000001B
INT$_PSECT        = 0000001C
INT$_REDEF        = 0000001D
INT$_REF          = 0000001E
```

```
INT$_REST       = 0000001F
INT$_SAME       = 00000009
INT$_SAVE       = 00000020
INT$_SBTTL      = 00000021
INT$_SETFLAG    = 00000022
INT$_SETLONG    = 00000023
INT$_SPIC       = 00000024
INT$_SPID       = 00000025
INT$_STIB       = 00000026
INT$_STIL       = 00000028
INT$_STIW       = 00000027
INT$_STKEPT     = 00000029
INT$_STKG       = 0000002A
INT$_STKL       = 0000002B
INT$_STKPC      = 0000002C
INT$_STKS       = 0000002D
INT$_STOB       = 00000034
INT$_STOL       = 0000002E
INT$_STOW       = 00000035
INT$_STRB       = 0000002F
INT$_STRL       = 00000031
INT$_STRSB      = 00000032
INT$_STRSW      = 00000033
INT$_STRW       = 00000030
INT$_STSB       = 00000036
INT$_STSW       = 00000037
INT$_SUB        = 0000000A
INT$_SUME       = 00000039
INT$_WRN        = 00000038
INT$_XOR        = 0000000B
LEN$K_MTXTAB    = 00000005
LST$G_MACROBIN  ********  X  04
LST$G_MACROXPAN ********  X  04
LST$K_BUFSIZ    = 00000086
LST$K_L_P_PAGE  = 0000003C
LST$K_TITLE_SIZ= 00000028
MAB$B_ARGNO     00000005
MAB$B_NAME      00000004
MAB$K_BLKSIZ    0000000C
MAB$L_DVPTR     00000008
MAB$L_LINK      00000000
MAB$W_DVLEN     00000006
MAC$ABORT_PASS1 ********  X  04
MAC$AB_LINEBF   ********  X  04
MAC$AB_TMPBUF   ********  X  04
MAC$ERR_INTERN  ********  X  04
MAC$GET_IRC_LIN 00000000 RG  04
MAC$GET_IRP_LIN 0000002C RG  04
MAC$GET_MAC_LIN 00000088 RG  04
MAC$GET_RPT_LIN 00000050 RG  04
MAC$GL_ERRPTX   ********  X  04
MAC$GL_IF_LEVEL ********  X  04
MAC$GL_INPUTP   ********  X  04
MAC$GL_LINELN   ********  X  04
MAC$GL_LINEPT   ********  X  04
MAC$GL_LIST_LVL ********  X  04
MAC$INTERR_2_LW ********  X  04
MAC$INTOUT_N    ********  X  04
MAC$INTOUT_X    ********  X  04
MAC$POP_INPUT   ********  X  04
MAC$_LINTOOLONG= 007D9132
MAC$_SYMNOTABS  = 007D91EA
MAC$_UNDEFSYM   = 007D9212
MAC$_UNTERMCOND= 007D9232
MACLIN_ARGMRK   0000011B R   04
MACLIN_COPY_LIN 000000F9 R   04
MACLIN_DECOUT   0000039A R   04
MACLIN_DISP_TAB 00000008 R   03
MACLIN_END_TEXT 000002B7 R   04
MACLIN_EXO      000002D5 R   04
MACLIN_EXIT     000002D2 R   04
MACLIN_GET_STR  00000358 R   04
MACLIN_GET_VAL  000002F6 R   04
MACLIN_GET_VLO  000002FF R   04
MACLIN_LEX_EXIT 000001C8 R   04
MACLIN_LINE_END 0000024A R   04
MACLIN_LINK_CHO 000002E7 R   04
MACLIN_LINK_CHK 000002E1 R   04
MACLIN_LXEXT    00000168 R   04
MACLIN_LXLEN    00000154 R   04
MACLIN_LXLOC    000001CB R   04
MACLIN_MTXT_TAB 00000000 R   03
MACLIN_NEXT_CHO 000000C6 R   04
MACLIN_NEXT_CHR 000000CD R   04
MACLIN_TXTLNK   00000149 R   04
MACLIN_ZEROUT   00000398 R   04
MAC_RPT_END_CHK 00000069 R   04
MAC_SUBSYS      = 0000007D
MNB$B_ARGCT     00000017
MNB$B_NAME      00000004
MNB$K_BLKSIZ    0000001C
MNB$L_ARGP      00000018
MNB$L_CRSYMF    00000013
MNB$L_LINK      00000000
MNB$L_PAGC      0000000F
MNB$L_PAGP      0000000B
MNB$L_TXTP      00000005
MNB$W_FLAG      00000009
MTX$_ARGMRK     = 000000FF
MTX$_LITSTR     = 000000EF
MTX$_LITVAL     = 000000EE
MTX$_LXEXT      = 000000FC
MTX$_LXLEN      = 000000FD
MTX$_LXLOC      = 000000FB
MTX$_NOMORE     = 000000EC
MTX$_SYMADR     = 000000ED
MTX$_TXTLNK     = 000000FE
MXB$K_BLKSIZ    00000008
MXB$L_LINK      00000000
MXB$L_PAGES     00000004
OBJ$K_BUFSIZ    = 00000200
OPF$M_LASTOPR   = 00002000
OPF$M_OPTEXP    = 00001000
OPF$V_LASTOPR   = 0000000D
OPF$V_OPTEXP    = 0000000C
PSC$B_NAME      00000004
PSC$B_SEG       0000000C
PSC$B_UNUSED    0000000B
PSC$K_BLKSIZ    00000013
PSC$K_NO_OPTNS  = 0000000A
PSC$L_CURLOC    0000000F
PSC$L_LINK      00000000
PSC$L_MAXLGTH   00000005
PSC$M_ABS       = FFFFFFF7
PSC$M_ALIGNFLG  = 00004000
PSC$M_ALLOPTNS  = 000003FF
PSC$M_BYTE      = 00004000
PSC$M_CON       = FFFFFFFB
PSC$M_DEFAULT   = 000001C8
PSC$M_EXE       = 000000C0
PSC$M_GBL       = 00000010
PSC$M_LCL       = FFFFFFEF
PSC$M_LIB       = 00000002
PSC$M_LONG      = 00004800
PSC$M_NOEXE     = FFFFFFBF
PSC$M_NOPIC     = FFFFFFFE
PSC$M_NORD      = FFFFFF7F
PSC$M_NOSHR     = FFFFFFDF
PSC$M_NOVEC     = FFFFFDFF
PSC$M_NOWRT     = FFFFFEFF
PSC$M_OVR       = 00000004
PSC$M_PAGE      = 00006400
PSC$M_PIC       = 00000001
PSC$M_QUAD      = 00004C00
PSC$M_RD        = 00000080
PSC$M_REL       = 00000008
PSC$M_SHR       = 00000020
PSC$M_USR       = FFFFFFFD
PSC$M_VEC       = 00000200
PSC$M_WORD      = 00004400
PSC$M_WRT       = 00000180
PSC$S_ALIGNMENT= 00000004
PSC$V_ALIGNFLG  = 0000000E
PSC$V_ALIGNMENT= 0000000A
PSC$V_EXE       = 00000006
PSC$V_GBL       = 00000004
PSC$V_LIB       = 00000001
PSC$V_OVR       = 00000002
PSC$V_PIC       = 00000000
PSC$V_RD        = 00000007
PSC$V_REL       = 00000003
PSC$V_SHR       = 00000005
PSC$V_VEC       = 00000009
PSC$V_WRT       = 00000008
PSC$W_FLAG      00000009
PSC$W_OPTIONS   0000000D
RDX$V_BINARY    = 00000000
RDX$V_DECIMAL   = 00000002
RDX$V_DOUBLE    = 00000005
RDX$V_FLOAT     = 00000004
RDX$V_GFLOAT    = 00000006
```

```
RDX$V_HEX       = 00000003
RDX$V_HFLOAT    = 00000007
RDX$V_OCTAL     = 00000001
REG$_PC         = 0000000F
SEMI            = 0000003B
SIB$K_PG_MISS   = 0000000A
SYM$B_NAME        00000004
SYM$B_SEG         0000000C
SYM$B_TOKEN       0000000B
SYM$K_BLKSIZ      0000000D
SYM$K_MAXLEN    = 0000001F
SYM$K_TWOCOL    = 00000010
SYM$L_LINK        00000000
SYM$L_VAL         00000005
SYM$M_ABS       = 00000010
SYM$M_ASN       = 00000100
SYM$M_CRFO      = 00002000
SYM$M_DEBUG     = 00000020
SYM$M_DEF       = 00000001
SYM$M_DELMAC    = 00000200
SYM$M_EPT       = 00000200
SYM$M_EXTRN     = 00000008
SYM$M_GLOBL     = 00000004
SYM$M_LOCAL     = 00000040
SYM$M_ODBG      = 00000400
SYM$M_REF       = 00000080
SYM$M_RELPSECT  = 00000800
SYM$M_SUPR      = 00004000
SYM$M_WEAK      = 00000002
SYM$M_XCRF      = 00001000
SYM$V_ABS       = 00000004
SYM$V_ASN       = 00000008
SYM$V_CRFO      = 0000000D
SYM$V_DEBUG     = 00000005
SYM$V_DEF       = 00000000
SYM$V_DELMAC    = 00000009
SYM$V_EPT       = 00000009
SYM$V_EXTRN     = 00000003
SYM$V_GLOBL     = 00000002
SYM$V_LOCAL     = 00000006
SYM$V_ODBG      = 0000000A
SYM$V_REF       = 00000007
SYM$V_RELPSECT  = 0000000B
SYM$V_SUPR      = 0000000E
SYM$V_WEAK      = 00000001
SYM$V_XCRF      = 0000000C
SYM$W_FLAG        00000009
TAB             = 00000009
X1              = 00000400
X2              = 0000000F
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
. BLANK .	00000000 (0.)	01 (1.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
ABS	0000003C (60.)	02 (2.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
MAC$RO_DATA	0000001C (28.)	03 (3.)	NOPIC USR CON REL GBL NOSHR NOEXE RD NOWRT NOVEC LONG
MAC$RO_CODE_MAC	000003B7 (951.)	04 (4.)	NOPIC USR CON REL GBL NOSHR EXE RD NOWRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	37	00:00:00.04	00:00:00.98
Command processing	128	00:00:00.36	00:00:02.51
Pass 1	222	00:00:03.72	00:00:18.55
Symbol table sort	0	00:00:00.49	00:00:01.35
Pass 2	138	00:00:01.19	00:00:06.70
Symbol table output	33	00:00:00.18	00:00:00.42
Psect synopsis output	3	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	563	00:00:06.00	00:00:30.53

The working set limit was 1500 pages.
36150 bytes (71 pages) of virtual memory were used to buffer the intermediate code.
There were 30 pages of symbol table space allocated to hold 468 non-local and 59 local symbols.
718 source lines were read in Pass 1, producing 19 object records in Pass 2.
14 pages of virtual memory were used to define 13 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[MACRO.OBJ]MACRO.MLB;1	12
_255DUA28:[SYSLIB]STARLET.MLB;2	3
TOTALS (all libraries)	15

574 GETS were required to define 15 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:MACLIN/OBJ=OBJ$:MACLIN MSRC$:MACLIN/UPDATE=(ENH$:MACLIN)+LIB$:MACRO/LIB
```