```
MMM         MMM     AAAAAAAAA        CCCCCCCCCCCC  RRRRRRRRRRRR        OOOOOOOOO
MMM         MMM     AAAAAAAAA        CCCCCCCCCCCC  RRRRRRRRRRRR        OOOOOOOOO
MMM         MMM     AAAAAAAAA        CCCCCCCCCCCC  RRRRRRRRRRRR        OOOOOOOOO
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             RRRRRRRRRRRR     OOO         OOO
MMM         MMM  AAA         AAA  CCC             RRRRRRRRRRRR     OOO         OOO
MMM         MMM  AAA         AAA  CCC             RRRRRRRRRRRR     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     CCCCCCCCCCCC  RRR         RRR     OOOOOOOOO
MMM         MMM  AAA         AAA     CCCCCCCCCCCC  RRR         RRR     OOOOOOOOO
MMM         MMM  AAA         AAA     CCCCCCCCCCCC  RRR         RRR     OOOOOOOOO
```

FILEID**P2ACT1

```
PPPPPPPP      222222      AAAAAA      CCCCCCCC  TTTTTTTTTT      11
PPPPPPPP      222222      AAAAAA      CCCCCCCC  TTTTTTTTTT      11
PP      PP  22      22  AA      AA  CC              TT        1111
PP      PP  22      22  AA      AA  CC              TT        1111
PP      PP          22  AA      AA  CC              TT          11
PP      PP          22  AA      AA  CC              TT          11
PPPPPPPP          22    AA      AA  CC              TT          11
PPPPPPPP          22    AA      AA  CC              TT          11
PP              22      AAAAAAAAAA  CC              TT          11
PP              22      AAAAAAAAAA  CC              TT          11
PP            22        AA      AA  CC              TT          11          ....
PP            22        AA      AA  CC              TT          11          ....
PP          2222222222  AA      AA    CCCCCCCC      TT        111111        ....
PP          2222222222  AA      AA    CCCCCCCC      TT        111111        ....

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

(2)	96	DECLARATIONS
(3)	114	P2$STKG STACK GLOBAL, EXTERNAL SYMBOL
(3)	133	P2$STKS STACK SYMBOL
(4)	221	P2$STKL STACK LITERAL VALUE
(5)	268	P2$BDST BRANCH DESTINATION GENERATION
(6)	405	P2$LGLAB LABEL ROUTINE--CHECK FOR PHASE ERRORS
(6)	424	P2$OP OPCODE GENERATOR
(7)	457	P2$REF OPERAND REFERENCE GENERATION
(8)	569	ROUTINES CALLED BY P2$REF
(14)	908	OPND CHECK
(15)	947	P2$END THE END OF PASS 2

```
0000          .TITLE  MAC$P2ACT1 ACTION ROUTINES FOR PASS 2
0000          .IDENT  'V04-000'
0000
0000    ;
0000    ;*******************************************************************************
0000    ;*                                                                             *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                    *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                     *
0000    ;*  ALL RIGHTS RESERVED.                                                       *
0000    ;*                                                                             *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED      *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE     *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER      *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY      *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY      *
0000    ;*  TRANSFERRED.                                                               *
0000    ;*                                                                             *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE      *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
0000    ;*  CORPORATION.                                                               *
0000    ;*                                                                             *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS      *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                    *
0000    ;*                                                                             *
0000    ;*                                                                             *
0000    ;*******************************************************************************
0000    ;
0000
0000    ;++
0000    ; FACILITY:     VAX MACRO ASSEMBLER OBJECT LIBRARY
0000    ;
0000    ; ABSTRACT:
0000    ;
0000    ; The VAX-11 MACRO assembler translates MACRO-32 source code into object
0000    ; modules for input to the VAX-11 LINKER.
0000    ;
0000    ; ENVIRONMENT:  USER MODE
0000    ;
0000    ; AUTHOR: Benn Schreiber, CREATION DATE: 25-AUG-78
0000    ;
0000    ; MODIFIED BY:
0000    ;
0000    ;       V03-004 MTR0037         Mike Rhodes     16-May-1984
0000    ;               Filter object code data values from being written
0000    ;               into psects with the absolute attribute, since the
0000    ;               object code command records have been removed.
0000    ;
0000    ;       V03-003 MCN0153         Maria del C. Nasr       15-Feb-1984
0000    ;               Eliminate an extra, unnecesary $VPUSH when processing
0000    ;               transfer vectors.
0000    ;
0000    ;       V03-002 MTR0033         Mike Rhodes     22-Apr-1983
0000    ;               Allow the removal of the blank psect if it's not referenced.
0000    ;
0000    ;       V03-001 MTR0031         Mike Rhodes     19-Apr-1983
0000    ;               Remove obsolete reference to $MAC_TIRCMDDEF macro.
0000    ;
```

```
0000    ;       V03.00  MTR0008         Mike Rhodes     15-Mar-1982
0000    ;               Modify routine P2$END to test the setting of the "external"
0000    ;               error flags FLG$V_EXTERR and FLG$V_EXTWRN, exiting with the
0000    ;               more severe status if set.  Else, it will check and set the
0000    ;               completion status as in the past, according to the type of
0000    ;               assembly errors encountered.  Fixes SPR #11-41651(A).
0000    ;
0000    ;       V02.09  CNH0039         Chris Hume      7-Oct-1980
0000    ;               Closed off a few remaining object record length checking
0000    ;               errors.  (IODAT.MAR 02.08, OHDOUT.MAR 02.02)
0000    ;
0000    ;       V02.08  CNH0036         Chris Hume      14-Jul-1980
0000    ;               Closed off some more areas where the assembler was writing
0000    ;               object records too long for the linker.  (P2ACT2.MAR 02.11)
0000    ;
0000    ;       V02.07  CNH0034         Chris Hume      14-May-1980
0000    ;               Restrained the assembler from occasionally writing object
0000    ;               records too long for the linker.
0000    ;
0000    ;       V01.06  RN0023          R. Newland      3-Nov-1979
0000    ;               New message codes to get error messages from system
0000    ;               message file.
0000    ;
0000    ;       V01.05  RN0007          R. Newland      28-Aug-1979
0000    ;               Store signed byte for non-optimised, not PC relative
0000    ;               8-bit displacment and deferred displacement
0000    ;
0000    ;       V01.04  RN0005          R. Newland      13-Aug-1979
0000    ;               Remove .ALIGN LONG statements
0000    ;
0000    ;       V01.03  RN0003          R. Newland      22-May-1979
0000    ;               Store signed word for not PC-relative, non-optimised
0000    ;               16-bit displacement and deferred displacement
0000    ;
0000    ;       V01.02  RN0002          R. Newland      01-Feb-1979
0000    ;               Changes for Source Update Merge
0000    ;--
```

```
     0000     96          .SBTTL  DECLARATIONS
     0000     97 ;
     0000     98 ; INCLUDE FILES:
     0000     99 ;
     0000    100
     0000    101 ;
     0000    102 ; MACROS:
     0000    103 ;
     0000    104
     0000    105          $MAC_SYMBLKDEF                  ;DEFINE SYMBOL BLOCK OFFSETS
     0000    106          $MAC_CTLFLGDEF                  ;DEFINE CONTROL FLAGS
     0000    107          $MAC_ADRMODDEF                  ;DEFINE ADDRESSING MODES
     0000    108          $MAC_OBJCODDEF                  ;DEFINE OBJECT MODULE CODES
     0000    109          $MACMSGDEF                      ; Define message codes
     0000    110          $MAC_GENVALDEF                  ; Define VAX-11 MACRO commom symbols
     0000    111
 00000000    112          .PSECT  MAC$RO_CODE_P2,NOWRT,GBL,LONG
```

```
                        0000          .SBTTL  P2$STKG STACK GLOBAL, EXTERNAL SYMBOL
                        0000
                        0000  ;++
                        0000  ; FUNCTIONAL DESCRIPTION:
                        0000  ;
                        0000  ;       EMIT TEXT TO OBJECT FILE TO STACK A GLOBAL SYMBOL.
                        0000  ;
                        0000  ; INPUTS:
                        0000  ;
                        0000  ;       R8      POINTS INTO INTERMEDIATE BUFFER
                        0000  ;
                        0000  ;--
                        0000
                        0000  P2$STKG::
56   88   D0            0000          MOVL    (R8)+,R6                ;GET SYMBOL BLOCK ADDRESS
                        0003          $VPUSH  SYM$L_VAL(R6)           ;PUSH SYMBOL VALUE ONTO STACK
                        000C          $OBJ_CHKBYT #TIR$C_STA_GBL      ;EMIT COMMAND FOR LINKER TO STACK GLOBAL
     FFEB'  31          0012          BRW     MAC$SYMNAMOUT           ;EMIT SYMBOL NAME AND RETURN
                        0015
                        0015          .SBTTL  P2$STKS STACK SYMBOL
                        0015
                        0015  ;++
                        0015  ; FUNCTIONAL DESCRIPTION:
                        0015  ;
                        0015  ;       STACK THE SYBMOL POINTED TO BY THE WORD IN THE INTERMEDIATE
                        0015  ;       FILE.  IF THE SYMBOL IS ABSOLUTE IT IS TREATED LIKE A
                        0015  ;       'STACK LITERAL'.  IF IT IS EXTERNAL IT IS TREATED LIKE A
                        0015  ;       'STACK GLOBAL'.
                        0015  ;
                        0015  ; INPUTS:
                        0015  ;
                        0015  ;       R8      POINTER INTO INTERMEDIATE FILE
                        0015  ;
                        0015  ;--
                        0015
                        0015  P2$STKS::
56   68   D0            0015          MOVL    (R8),R6                 ;GET SYMBOL BLOCK ADDRESS
     7E   13            0018          BEQL    80$                     ;IF EQL SPECIAL CANCEL COMMAND
                        001A                                          ;  FOR BRANCH DESTINATION
07 09 A6  04   E1       001A          BBC     #SYM$V_ABS,SYM$W_FLAG(R6),10$ ;BRANCH IF NOT ABSOLUTE SYMBOL
                        001F  ;
                        001F  ; PRETEND ITS A STACK LITERAL
                        001F  ;
58   05 A6  9E          001F          MOVAB   SYM$L_VAL(R6),R8        ;POINT TO SYMBOL VALUE
     0080   31          0023          BRW     P2$STKL                 ;AND TREAT LIKE STACK LITERAL
                        0026  ;
                        0026  ; RELOCATABLE SYMBOL
                        0026  ;
15 09 A6  00   E0       0026  10$:    BBS     #SYM$V_DEF,SYM$W_FLAG(R6),40$ ;BRANCH IF SYMBOL DEFINED
05 09 A6  06   E0       002B          BBS     #SYM$V_LOCAL,SYM$W_FLAG(R6),20$ ;BRANCH IF LOCAL SYMBOL
08 09 A6  03   E0       0030          BBS     #SYM$V_EXTRN,SYM$W_FLAG(R6),30$ ;BRANCH IF DECLARED EXTERNAL
                        0035  ;
                        0035  ; SYMBOL LOCAL AND NOT DEFINED, OR NOT EXTERNAL
                        0035  ;
                        0035  20$:    $MAC_P2_ERR UNDEFSYM            ; Report undefined symbol
                        003D  ;
                        003D  ; TREAT AS GLOBAL SYMBOL
```

```
                        003D   171 ;
            FFC0   31   003D   172 30$:    BRW     P2$STKG                  ;AND TREAT AS STACK GLOBAL
                        0040   173 ;
                        0040   174 ; SYMBOL IS DEFINED
                        0040   175 ;
                        0040   176 40$:    $VPUSH  SYM$L_VAL(R6)            ;STACK SYMBOL VALUE
      54   0C A6   9A   0049   177         MOVZBL  SYM$B_SEG(R6),R4         ;GET SYMBOL SEGMENT
      53   05 A6   9E   004D   178         MOVAB   SYM$L_VAL(R6),R3         ;POINT TO SYMBOL VALUE
      52      04   9A   0051   179         MOVZBL  #4,R2                    ;ASSUME LONGWORD
      51      06   9A   0054   180         MOVZBL  #TIR$C_STA_PL,R1         ;STACK PSECT BASE PLUS LONGWORD
         02 A3     B5   0057   181         TSTW    2(R3)                    ;CAN WE OPTIMIZE?
            19     12   005A   182         BNEQ    60$                      ;IF NEQ NO
            63     B5   005C   183         TSTW    (R3)                     ;
            15     19   005E   184         BLSS    60$                      ;IF LSS CANNOT OPTIMIZE
      52      02   9A   0060   185         MOVZBL  #2,R2                    ;MAKE WORD
      51      05   9A   0063   186         MOVZBL  #TIR$C_STA_PW,R1         ;STACK PSECT BASE PLUS WORD
         01 A3     95   0066   187         TSTB    1(R3)                    ;OPTIMIZE TO BYTE?
            0A     12   0069   188         BNEQ    60$                      ;IF NEQ NO
            63     95   006B   189         TSTB    (R3)                     ;
            06     19   006D   190         BLSS    60$                      ;IF LSS CANNOT OPTIMIZE
      52      01   9A   006F   191         MOVZBL  #1,R2                    ;MAKE BYTE
      51      04   9A   0072   192         MOVZBL  #TIR$C_STA_PB,R1         ;STACK PSECT BASE PLUS BYTE
                        0075   193 ;
                        0075   194 ; R1 -- COMMAND
                        0075   195 ; R4 -- SEGMENT
                        0075   196 ; R3 -- POINTS TO SYMBOL VALUE
                        0075   197 ;
                        0075   198 60$:    $OBJ_CHKBYT R1                   ;STACK COMMAND
            54   D5     007B   199         TSTL    R4                       ;MAKE SURE THAT WE DON'T FIDDLE WITH
            0A   13     007D   200         BEQL    65$                      ;THE ABSOLUTE PSECT.
            07   E0     007F   201         BBS     #SYM$V_REF,-             ;IF THE BLANK PSECT HASN'T BEEN REF'D
02 00000009'EF          0081   202                 PSECT$BLANK+SYM$W_FLAG,65$ ;IT'LL BE REMOVED, DECR THE SEG#
            54   D7     0087   203         DECL    R4                       ;TO PRESERVE THE PSECT ALIGNMENT
                        0089   204 65$:    $OBJ_OUTBYT R4                   ;STACK SEGMENT #
                        008F   205 70$:    $OBJ_OUTBYT (R3)+                ;EMIT VALUE
         F7 52      F5   0095   206         SOBGTR  R2,70$                   ;LOOP FOR ALL BYTES
                    05   0098   207 80$:    RSB
                        0099   208
                        0099   209 ;++
                        0099   210 ; FUNCTIONAL DESCRIPTION:
                        0099   211 ;
                        0099   212 ;       EMIT TEXT TO OBJECT FILE TO REDEFINE TRANSFER ADDRESS.
                        0099   213 ;
                        0099   214 ;--
                        0099   215
                        0099   216 P2$REDEF::
      56   88     D0     0099   217         MOVL    (R8)+,R6                 ;GET SYMBOL BLOCK ADDRESS
                        009C   218         $OBJ_CHKBYT #TIR$C_OPR_REDEF     ;EMIT REDEFINE COMMAND
         FF5A'      31   00A3   219         BRW     MAC$SYMNAMOUT            ;EMIT SYMBOL NAME AND RETURN
```

```
                        00A6          .SBTTL  P2$STKL STACK LITERAL VALUE
                        00A6
                        00A6    ;++
                        00A6    ; FUNCTIONAL DESCRIPTION:
                        00A6    ;
                        00A6    ;       STACK THE LITERL VALUE SUPPLIED IN THE INTERMEDIATE BUFFER.
                        00A6    ;       IF POSSIBLE, IT WILL BE OPTIMIZED TO A BYTE OR WORD.
                        00A6    ;
                        00A6    ; INPUTS:
                        00A6    ;
                        00A6    ;       R8      POINTS INTO INTERMEDIATE BUFFER
                        00A6    ;
                        00A6    ;--
                        00A6
                        00A6    P2$STKL::
                        00A6            $VPUSH  (R8)                    ;STACK THE VALUE
                55   04   9A    00AE            MOVZBL  #4,R5           ;ASSUME LONGWORD VALUE
                54   03   9A    00B1            MOVZBL  #TIR$C_STA_LW,R4        ;...
                  02 A8   B5    00B4            TSTW    2(R8)           ;CAN WE OPTIMIZE TO WORD?
                     13   12    00B7            BNEQ    10$             ;MAYBE--SEE IF NEGATIVE
                55   02   9A    00B9            MOVZBL  #2,R5           ;YES--MAKE WORD VALUE
                54   08   9A    00BC            MOVZBL  #TIR$C_STA_UW,R4        ;STACK UNSIGNED WORD
                  01 A8   95    00BF            TSTB    1(R8)           ;OPTIMIZE TO BYTE?
                     2B   12    00C2            BNEQ    20$             ;IF NEQ NO
                55   01   9A    00C4            MOVZBL  #1,R5           ;YES--MAKE BYTE VALUE
                54   07   9A    00C7            MOVZBL  #TIR$C_STA_UB,R4        ;STACK UNSIGNED BYTE
                     23   11    00CA            BRB     20$
                        00CC    ;
                        00CC    ; HIGH ORDER WORD IS NON-ZERO
                        00CC    ;
        02 A8   FFFF 8F   B1    00CC    10$:    CMPW    #-1,2(R8)       ;IS HI-ORDER WORD -1?
                     1B   12    00D2            BNEQ    20$             ;IF NE NO--CANNOT OPTIMIZE
                     68   B5    00D4            TSTW    (R8)            ;LOW ORDER WORD NEGATIVE?
                     17   18    00D6            BGEQ    20$             ;IF GE NO--CANNOT OPTIMIZE
                55   02   9A    00D8            MOVZBL  #2,R5           ;YES--OPTIMIZE TO WORD
                54   02   9A    00DB            MOVZBL  #TIR$C_STA_SW,R4        ;STACK SIGNED WORD
           01 A8   FF 8F   91    00DE            CMPB    #-1,1(R8)       ;CAN WE OPTIMIZE TO BYTE?
                     0A   12    00E3            BNEQ    20$             ;IF NE NO
                     68   95    00E5            TSTB    (R8)            ;MAYBE
                     06   18    00E7            BGEQ    20$             ;IF GE CANNOT OPTIMIZE
                55   01   9A    00E9            MOVZBL  #1,R5           ;MAKE BYTE
                54   01   9A    00EC            MOVZBL  #TIR$C_STA_SB,R4        ;STACK SIGNED BYTE
                        00EF    20$:    $OBJ_CHKBYT R4                  ;EMIT COMMAND
                        00F5    30$:    $OBJ_OUTBYT (R8)+               ;EMIT DATA
                  F7 55   F5    00FB            SOBGTR  R5,30$          ;LOOP FOR ALL BYTES
                     05   00FE            RSB
```

```
                        00FF        .SBTTL  P2$BDST BRANCH DESTINATION GENERATION
                        00FF
                        00FF  ;++
                        00FF  ; FUNCTIONAL DESCRIPTION:
                        00FF  ;
                        00FF  ;         THIS ROUTINE GENERATES BRANCH DESTINATIONS.
                        00FF  ;
                        00FF  ;
                        00FF  ; INPUTS:
                        00FF  ;
                        00FF  ;         R8 >>   2 BYTES OF MAC$GL_OPSIZE
                        00FF  ;                 4 BYTES OF 0 OR SYMBOL ID ADDRESS (IF SPECIAL RESOLUTION)
                        00FF  ;                 1 BYTE OF 0 OR PSECT # (IF SPECIAL RESOLUTION)
                        00FF  ;
                        00FF  ;--
                        00FF
                        00FF  P2$BDST::
      0000'CF   68   3C 00FF          MOVZWL  (R8),W^MAC$GL_OPSIZE    ;COPY FLAGS
         00 68   0D  E5 0104          BBCC    #OPF$V_LASTOPR,(R8),5$  ;CLEAR LAST OPERAND FLAG IN BUFFER
0000'8F  0000'CF   B1 0108  5$:       CMPW    W^MAC$GL_LIST_PTR,#MAC$AB_LST_OP1 ;IN OPERAND FIELD?
              0B   1B 010F          BLEQU   10$                     ;IF LEQU POINTER OK
         0000'CF   9E 0111          MOVAB   W^MAC$AB_LST_OP1,-      ;POSITION TO OPERAND
         0000'CF      0115                       W^MAC$GL_LIST_PTR
         0000'CF   D4 0118          CLRL    W^MAC$GL_MOPNUM
      56    02 A8   D0 011C  10$:    MOVL    2(R8),R6                ;GET SYMBOL ADDRESS
            01 A8   95 0120          TSTB    1(R8)                   ;SPECIAL PASS 2 RESOLUTION?
               28   12 0123          BNEQ    50$                     ;IF NEQ NO
               56   D5 0125          TSTL    R6                      ;YES--SYMBOL SPECIFIED?
               24   13 0127          BEQL    50$                     ;IF EQL NO
   0C A6    06 A8   91 0129          CMPB    6(R8),SYM$B_SEG(R6)     ;YES--SAME PSECT?
               10   12 012E          BNEQ    40$                     ;IF NEQ NO
   0B 09 A6    00   E1 0130          BBC     #SYM$V_DEF,SYM$W_FLAG(R6),40$ ;YES--SYMBOL DEFINED?
                        0135  ;
                        0135  ; SYMBOL IS DEFINED, AND LIVES IN SAME PSECT.  RESOLVE SYMBOL
                        0135  ; AND PRETEND OPTIMIZATION.
                        0135  ;
   02 A8    05 A6   D0 0135          MOVL    SYM$L_VAL(R6),2(R8)     ;SET SYMBOL VALUE
      01 A8    01   90 013A          MOVB    #1,1(R8)                ;PRETEND OPTIMIZED
               0D   11 013E          BRB     50$                     ;SKIP SOME
                        0140  ;
                        0140  ; SYMBOL IN DIFFERENT PSECT OR NOT DEFINED.  TREAT LIKE SYMBOL
                        0140  ;
         0160 8F   BB 0140  40$:     PUSHR   #^M<R5,R6,R8>           ;SAVE REGISTERS
              88   B5 0144          TSTW    (R8)+                   ;POINT TO ID POINTER
            FECC   30 0146          BSBW    P2$STKS                 ;STACK THE SYMBOL
         0160 8F   BA 0149          POPR    #^M<R5,R6,R8>           ;RESTORE REGISTERS
           01 A8   95 014D  50$:    TSTB    1(R8)                   ;OPTIMIZED?
              09   13 0150          BEQL    60$                     ;IF EQL NO
                   0152             $VPUSH  2(R8)                   ;YES--STACK TARGET ADDRESS
                   015B  60$:       $VPUSH  W^MAC$GL_PC             ;STACK CURRENT PC
         01    68   91 0165          CMPB    (R8),#1                 ;ONE BYTE BRANCH DESTINATION?
               59   12 0168          BNEQ    120$                    ;IF NE NO
                   016A  ;
                   016A  ; ONE BYTE BRANCH DESTINATION
                   016A  ;
      0000'CF47    D6 016A          INCL    W^MAC$AL_VALSTACK[R7]   ;ADD 1 TO PC
           FE8E'   30 016F          BSBW    P1$ARITH_SUB            ;COMPUTE BRANCH OFFSET
```

```
55   0000'CF47   DE  0172   325          MOVAL   W^MAC$AL_VALSTACK[R7],R5;POINT TO RESULT
          01 A8  95  0178   326          TSTB    1(R8)                   ;OPTIMIZED?
             33  13  017B   327          BEQL    100$                    ;IF EQL NO
          02 A5  B5  017D   328          TSTW    2(R5)                   ;YES--CHECK FOR DEST. OUT OF RANGE
             09  12  0180   329          BNEQ    70$                     ;IF NEQ CHECK NEGATIVE DEST. OUT OF RANGE
65     FF80 8F  B3  0182   330          BITW    #^C<^X7F>,(R5)          ;POSITIVE DEST. OUT OF RANGE?
             17  12  0187   331          BNEQ    80$                     ;IF NEQ YES--REPORT ERROR
             1D  11  0189   332          BRB     90$                     ;NO--CONTINUE
                     018B   333 ;
                     018B   334 ; CHECK NEGATIVE DEST. OUT OF RANGE
                     018B   335 ;
53     007F 8F  65  A9  018B   336 70$:    BISW3   (R5),#^X7F,R3           ;HOPEFULLY MAKE -1
  FFFF 8F    02 A5  B1  0191   337          CMPW    2(R5),#-1               ;HI WORD ALL ONES?
             07  12  0197   338          BNEQ    80$                     ;IF NEQ NO--DEST. OUT OF RANGE
     FFFF 8F 53  B1  0199   339          CMPW    R3,#-1                  ;YES--LOW WORD AND ^XFF ALL ONES?
             08  13  019E   340          BEQL    90$                     ;IF EQL YES--OK
                     01A0   341 ;
                     01A0   342 ; DEST. OUT OF RANGE
                     01A0   343 ;
                     01A0   344 80$:    $MAC_P2_ERR BRDESTRANG          ; Report branch destination out of range
         50  65  90  01A8   345 90$:    MOVB    (R5),R0                 ;GET DESTINATION OFFSET
           FE52' 30  01AB   346          BSBW    MAC$STOIM               ;STORE INTO OBJECT FILE
             0B  11  01AE   347          BRB     110$
                     01B0   348 ;
                     01B0   349 ; UN-OPTIMIZABLE BRANCH DESTINATION
                     01B0   350 ;
     0000'DF 27  90  01B0   351 100$:    MOVB    #^A/'/,@W^MAC$GL_LIST_PTR ;PUT QUOTE IN LISTING
                     01B5   352          $OBJ_CHKBYT #TIR$C_STO_BD       ;STORE BYTE DISPLACED
                     01BB   353 ;
                     01BB   354 ; FINISH UP 1-BYTE BRANCH DESTINATION PROCESSING
                     01BB   355 ;
                     01BB   356 110$:    $MAC_LIST_BYTE #1               ;SEND ONE BYTE TO LISTING
             54  11  01C1   357          BRB     180$                    ;
                     01C3   358 ;
                     01C3   359 ; TWO-BYTE BRANCH DESTINATION GENERATION
                     01C3   360 ;
0000'CF47    02  C0  01C3   361 120$:    ADDL2   #2,W^MAC$AL_VALSTACK[R7];ADD TWO TO PC
           FE34' 30  01C9   362          BSBW    P1$ARITH_SUB            ;COMPUTE OFFSET
55   0000'CF47   DE  01CC   363          MOVAL   W^MAC$AL_VALSTACK[R7],R5;POINT TO OFFSET RESULT
          01 A8  95  01D2   364          TSTB    1(R8)                   ;OPTIMIZED OFFSET?
             2F  13  01D5   365          BEQL    160$                    ;IF EQL NO
     53   02 A5  B0  01D7   366          MOVW    2(R5),R3                ;YES--CHECK FOR DEST. OUT OF RANGE
             0D  13  01DB   367          BEQL    130$                    ;IF EQL CHECK FOR FORWARD OUT OF RANGE
  FFFF 8F    53  B1  01DD   368          CMPW    R3,#-1                  ;NEGATIVE--IS HI WORD ALL ONES?
             0A  12  01E2   369          BNEQ    140$                    ;IF NEQ NO--RANGE ERROR
             65  B5  01E4   370          TSTW    (R5)                    ;YES--IS LO WORD NEGATIVE?
             06  18  01E6   371          BGEQ    140$                    ;IF GEQ NO--RANGE ERROR
             0C  11  01E8   372          BRB     150$                    ;YES--OK
                     01EA   373 ;
                     01EA   374 ; CHECK FORWARD RANGE ERROR
                     01EA   375 ;
             65  B5  01EA   376 130$:    TSTW    (R5)                    ;FORWARD DEST. OUT OF RANGE
             08  18  01EC   377          BGEQ    150$                    ;IF GEQ NO
                     01EE   378 ;
                     01EE   379 ; REPORT BRANCH DESTINATION OUT OF RANGE
                     01EE   380 ;
                     01EE   381 140$:    $MAC_P2_ERR BRDESTRANG          ; Emit message
```

```
                        01F6   ;
                        01F6   ; FINISH TWO-BYTE BRANCH DESTINATION PROCESSING
                        01F6   ;
          50    85   90  01F6   150$:   MOVB    (R5)+,R0                ;GET LOW BYTE
             FE04'   30  01F9           BSBW    MAC$STOIM               ;STORE INTO OBJECT FILE
          50    65   90  01FC           MOVB    (R5),R0                 ;GET HIGH BYTE
             FDFE'   30  01FF           BSBW    MAC$STOIM               ;STORE INTO OBJECT FILE
             75   95  0202           TSTB    -(R5)                   ;KEEP R5 POINTING AT TOP OF VALUE STACK
             0B   11  0204           BRB     170$                    ;FINISH BRANCH DEST. PROCESSING
                        0206   ;
                        0206   ; NON-OPTIMIZED TWO BYTE BRANCH DESTINATION
                        0206   ;
     0000'DF    27   90  0206   160$:   MOVB    #^A/'/,@W^MAC$GL_LIST_PTR ;STORE QUOTE IN LISTING
                        020B           $OBJ_CHKBYT #TIR$C_STO_WD       ;STORE WORD DISPLACED
                        0211   ;
                        0211   ; FINISH TWO-BYTE BRANCH DESTINATION PROCESSING
                        0211   ;
                        0211   170$:   $MAC_LIST_BYTE #2               ;SEND TWO BYTES TO LISTING
                        0217   ;
                        0217   ; FINISH BRANCH DESTINATION PROCESSING
                        0217   ;
                   05   0217   180$:   RSB                             ; Return
```

```
                                0218          .SBTTL  P2$LGLAB LABEL ROUTINE--CHECK FOR PHASE ERRORS
                                0218
                                0218  ;++
                                0218  ; FUNCTIONAL DESCRIPTION:
                                0218  ;
                                0218  ;       THIS ROUTINE CHECKS THAT THE VALUE FOR THE SYMBOL IS
                                0218  ;       THE SAME AS THE CURRENT PC.  IF NOT, AN ERROR MESSAGE
                                0218  ;       IS ISSUED.
                                0218  ;
                                0218  ;--
                                0218
                                0218  P2$LGLAB::
            56   68   D0        0218          MOVL    (R8),R6                 ;POINT TO SYMBOL BLOCK
     05 A6   0000'CF   D1       021B          CMPL    W^MAC$GL_PC,SYM$L_VAL(R6) ;PC MATCH SYMBOL VALUE?
                 08   13        0221          BEQL    10$                     ;IF EQL YES
                                0223          $MAC_P2_ERR SYMOUTPHAS          ; No--report phase error
        00 6B    0C   E3        022B  10$:    BBCS    #FLG$V_MEBLST,(R11),20$ ;FORCE LABELS TO BE LISTED
                      05        022F  20$:    RSB
                                0230
                                0230          .SBTTL  P2$OP   OPCODE GENERATOR
                                0230
                                0230  ;++
                                0230  ; FUNCTIONAL DESCRIPTION:
                                0230  ;
                                0230  ;       THIS ROUTINE WRITES THE OPCODE TO THE OBJECT AND LISTING
                                0230  ;       FILES.
                                0230  ; INPUTS:
                                0230  ;
                                0230  ;       R8 >>   OPCODE VALUE (1 WORD)
                                0230  ;
                                0230  ;--
                                0230
                                0230  P2$OP::
0000'CF   0000'CF   9E          0230          MOVAB   W^MAC$AB_LST_OPR,W^MAC$GL_LIST_PTR ;MOVE TO OPCODE FIELD
                 57   D4        0237          CLRL    R7                      ;PREVENT STACK UNDERFLOW
                                0239          $VPUSH  (R8)                    ;PUSH OPCODE VALUE
   55   0000'CF47   DE          0241          MOVAL   W^MAC$AL_VALSTACK[R7],R5;KEEP R5 POINTING TO TOP OF STACK
           50   01   9A         0247          MOVZBL  #1,R0                   ;ASSUME 1 BYTE OPCODE
           01 A8    95          024A          TSTB    1(R8)                   ;TWO BYTE OPCODE?
              02    13          024D          BEQL    10$                     ;IF EQL NO
              50    D6          024F          INCL    R0                      ;YES--EMIT TWO BYTES
                                0251  10$:    $MAC_LIST_BYTE R0               ;SEND OPCODE VALUE TO LISTING
           01 A8    95          0254          TSTB    1(R8)                   ;TWO BYTE OPCODE?
              05    12          0257          BNEQ    20$                     ;IF NEQ YES
     0000'CF   02   C2          0259          SUBL2   #2,W^MAC$GL_LIST_PTR    ;TWO SPACES FOR ONE BYTE OPCODE
           50   88   90         025E  20$:    MOVB    (R8)+,R0                ;EMIT LOW BYTE OF OPCODE
           FD9C'     30         0261          BSBW    MAC$STOIM
           50   68   90         0264          MOVB    (R8),R0                 ;TWO BYTE OPCODE?
              03    13          0267          BEQL    30$                     ;IF EQL NO
           FD94'     31         0269          BRW     MAC$STOIM               ;YES--STORE INTO OBJECT FILE
                     05         026C  30$:    RSB                             ;AND RETURN
```

```
                        026D          .SBTTL  P2$REF  OPERAND REFERENCE GENERATION
                        026D
                        026D    ;++
                        026D    ; FUNCTIONAL DESCRIPTION:
                        026D    ;
                        026D    ;       THIS ROUTINE GENERATES OPERAND REFERENCES.  THE VALUE ASSOCIATED
                        026D    ;       WITH THE REFERENCE (IF ANY) IS LEFT ON THE VALUE STACK.
                        026D    ;
                        026D    ; INPUTS:
                        026D    ;
                        026D    ;       R8 >>   MODE    (1 BYTE)        OPERAND MODE ('E' MODE IF INDEXED)
                        026D    ;               IMODE   (1 BYTE)
                        026D    ;               REG     (1 BYTE)        OPERAND REGISTER ('E' REG. IF INDEXED)
                        026D    ;               IREG
                        026D    ;               FLAGS   (1 WORD)
                        026D    ;               VALUE   (1 LONGWORD)    ONLY IF OPTIMIZED
                        026D    ;
                        026D    ;--
                        026D
                        026D    P2$REF::
0000'CF   04 A8   3C    026D            MOVZWL  4(R8),W^MAC$GL_OPSIZE   ;COPY FLAGS
  00 04 A8   0D   E5    0273            BBCC    #OPF$V_LASTOPR,4(R8),5$ ;CLEAR LAST OPERAND FLAG
        0000'CF   B1    0278    5$:     CMPW    W^MAC$GL_LIST_PTR,-     ;POINTER OK?
        0000'8F         027C                            #MAC$AB_LST_OP1
              07   1B   027F            BLEQU   10$                     ;IF LEQU OK
        0000'CF   9E    0281            MOVAB   W^MAC$AB_LST_OP1,-      ;NO--SET POINTER
        0000'CF         0285                            W^MAC$GL_LIST_PTR
     00 6B   07   E5    0288    10$:    BBCC    #FLG$V_EXPOPT,(R11),20$ ;CLEAR OPTIMIZED EXPRESSION FLAG
                        028C    20$:
                        028C    P2$XREF:
        05 A8   95      028C            TSTB    5(R8)                   ;OPTIMIZED?
           0F   13      028F            BEQL    10$                     ;IF EQL NO
           57   DD      0291            PUSHL   R7                      ;YES--SAVE VALUE STACK POINTER
                        0293            $VPUSH  6(R8)                   ;STACK VALUE
     00 6B   07   E3    029C            BBCS    #FLG$V_EXPOPT,(R11),10$ ;SET OPTIMIZED EXPRESSION FLAG
        50   68   9A    02A0    10$:    MOVZBL  (R8),R0                 ;GET MODE OF REFERENCE
           05   19      02A3            BLSS    20$                     ;IF LSS THEN ILLEGAL MODE
        0F   50   91    02A5            CMPB    R0,#15                  ;
           0A   15      02A8            BLEQ    30$                     ;IF LEQ THEN LEGAL MODE
                        02AA    20$:    $MAC_P2_ERR ILLMODE             ; Report mode error
           50   D4      02B2            CLRL    R0                      ;USE MODE 0
        53   50   9A    02B4    30$:    MOVZBL  R0,R3                   ;SAVE MODE
  52    50   04   78    02B7            ASHL    #4,R0,R2                ;SHIFT MODE TO UPPER 4 BITS OF BYTE
    54    02 A8   9A    02BB            MOVZBL  2(R8),R4                ;GET REGISTER OF REFERENCE
           05   19      02BF            BLSS    40$                     ;IF LSS THEN ILLEGAL REGISTER
        0F   54   91    02C1            CMPB    R4,#15                  ;LEGAL REGISTER?
           08   15      02C4            BLEQ    50$                     ;IF LEQ YES
                        02C6    40$:    $MAC_P2_ERR ILLREGNUM           ; No-report register error
                        02CE    50$:
        52   54   88    02CE            BISB2   R4,R2                   ;PACK MODE AND REGISTER
                        02D1            $VPUSH  R2                      ;AND STORE ONTO VALUE STACK
55   0000'CF47   DE     02D9            MOVAL   W^MAC$AL_VALSTACK[R7],R5 ;POINT TO VALUE ON TOP OF STACK
                        02DF    ;
                        02DF    ; CALL REFERENCE PROCESSING ROUTINE WITH:
                        02DF    ;
                        02DF    ;       R8      POINTER INTO INTERMEDIATE CODE BUFFER
                        02DF    ;       R7      VALUE STACK POINTER
```

```
                        02DF   ;        R6
                        02DF   ;        R5      POINTER TO TOP VALUE ON VALUE STACK (MAC$AL_VALSTACK[R7])
                        02DF   ;        R4      REGISTER FOR REFERENCE
                        02DF   ;        R2      Number of characters to display operand
                        02DF   ;
52   0335'CF43   90     02DF            MOVB    W^REF_SIZE[R3],R2        ; Get characters required
53   02F5'CF43   D0     02E5            MOVL    W^REF_DISP[R3],R3        ;GET DISPATCH ROUTINE ADDRESS
             63   16     02EB            JSB     (R3)                     ;DISPATCH TO MODE ROUTINE
     03 6B   07   E5     02ED            BBCC    #FLG$V_EXPOPT,(R11),60$ ;BRANCH IF NOT OPTIMIZED REFERENCE
        57   8E   D0     02F1            MOVL    (SP)+,R7                 ;YES--RESTORE VALUE STACK PTR
             05          02F4   60$:     RSB                              ; and return
                         02F5                                             ;AND RETURN
                         02F5
                         02F5            .MACRO  REF_PRO ROUTINE
                         02F5            .DEBUG  ROUTINE
                         02F5            .LONG   ROUTINE
                         02F5            .ENDM   REF_PRO
                         02F5   REF_DISP:                                 ;OPERAND REFERENCE DISPATCH TABLE
                         02F5            REF_PRO REF_0                    ;LITERAL
                         02F9            REF_PRO REF_1                    ;IMMEDIATE (PC)+
                         02FD            REF_PRO REF_2                    ;DEFERRED IMMEDIATE @(PC)+
                         0301            REF_PRO REF_3                    ;POSITION INDEPENDENT (G^)
                         0305            REF_PRO REF_4                    ;INDEXED
                         0309            REF_PRO REF_5                    ;REGISTER
                         030D            REF_PRO REF_6                    ;REGISTER DEFERRED
                         0311            REF_PRO REF_7                    ;AUTO DECREMENT
                         0315            REF_PRO REF_8                    ;AUTO INCREMENT
                         0319            REF_PRO REF_9                    ;DEFERRED AUTO INCREMENT
                         031D            REF_PRO REF_A                    ;BYTE DISPLACEMENT
                         0321            REF_PRO REF_B                    ;DEFERRED BYTE DISPLACEMENT
                         0325            REF_PRO REF_C                    ;WORD DISPLACEMENT
                         0329            REF_PRO REF_D                    ;DEFERRED WORD DISPLACEMENT
                         032D            REF_PRO REF_E                    ;LONG WORD DISPLACEMENT
                         0331            REF_PRO REF_F                    ;DEFERRED LONG WORD DISPLACEMENT
                         0335   ;
                         0335   ; Table of characters required to display operand
                         0335   ;
                         0335   REF_SIZE:
02                       0335            .BYTE   2                        ; Literal
03                       0336            .BYTE   3                        ; Immediate (PC)+
03                       0337            .BYTE   3                        ; Deferred immediate (PC)+
0B                       0338            .BYTE   11                       ; Position independent (G^)
02                       0339            .BYTE   2                        ; Indexed
02                       033A            .BYTE   2                        ; Register
02                       033B            .BYTE   2                        ; Register deferred
02                       033C            .BYTE   2                        ; Auto decrement
02                       033D            .BYTE   2                        ; Auto increment
02                       033E            .BYTE   2                        ; Deferred auto increment
05                       033F            .BYTE   5                        ; Byte displacement
05                       0340            .BYTE   5                        ; Deferred byte displacement
07                       0341            .BYTE   7                        ; Word displacment
07                       0342            .BYTE   7                        ; Deferred word displacment
0B                       0343            .BYTE   11                       ; Long word displacment
0B                       0344            .BYTE   11                       ; Deferred word long displacment
```

```
                        0345          .SBTTL  ROUTINES CALLED BY P2$REF
                        0345
                        0345  ;++
                        0345  ; FUNCTIONAL DESCRIPTION:
                        0345  ;
                        0345  ;       THIS ROUTINE PROCESSES LITERAL REFERENCES
                        0345  ;
                        0345  ;--
                        0345
                        0345  REF_0:                                          ;LITERAL MODE
        02D0    30      0345          BSBW    OPND_CHECK                      ; Check listing space for operand
          57    D7      0348          DECL    R7                              ;IGNORE MODE/REG ON STACK
          75    D5      034A          TSTL    -(R5)                           ;BACKUP STACK POINTER
   1A 6B  07    E1      034C          BBC     #FLG$V_EXPOPT,(R11),30$         ;BRANCH IF NOT OPTIMIZED EXPRESSION
      02 A5     B5      0350          TSTW    2(R5)                           ;CHECK FOR LEGAL LITERAL
          05    12      0353          BNEQ    10$                             ;IF NEQ THEN TRUNCATION
      3F  65    B1      0355          CMPW    (R5),#63                        ;LARGEST LITERAL
          08    1B      0358          BLEQU   20$                             ;IF LEQ THEN OK
                        035A  10$:    $MAC_P2_ERR DATATRUNC                   ; Error message
      50  65    90      0362  20$:    MOVB    (R5),R0                         ;GET BYTE OF DATA
        FC98'   30      0365          BSBW    MAC$STOIM                       ;STORE INTO OBJECT FILE
          0B    11      0368          BRB     40$
                        036A  ;
                        036A  ; NOT OPTIMIZED
                        036A  ;
0000'DF   27    90      036A  30$:    MOVB    #^A/'/,@W^MAC$GL_LIST_PTR ;PUT QUOTE IN LISTING
                        036F          $OBJ_CHKBYT #TIR$C_STO_LI               ;STORE LITERAL
                        0375  40$:    $MAC_LIST_BYTE #1                       ;LIST THE BYTE
                  05    037B          RSB
                        037C
                        037C  ;++
                        037C  ; FUNCTIONAL DESCRIPTION:
                        037C  ;
                        037C  ;       REF_2 PROCESSES ABSOLUTE [@(PC)+] REFERENCES.  REF_1
                        037C  ;       PROCESSES IMMEDIATE REFERENCES
                        037C  ;
                        037C  ;--
                        037C
                        037C  REF_2:                                          ;ABSOLUTE @(PC)+
50    009F 8F   3C      037C          MOVZWL  #<ADM$_DFRAUTOINC@4>+15,R0;SET MODE/REGISTER
          05    11      0381          BRB     IMM
                        0383  REF_1:                                          ;IMMEDIATE MODE (PC)+
50    008F 8F   3C      0383          MOVZWL  #<ADM$_REGAUTOINC@4>+15,R0;SET MODE/REGISTER
      65  50    D0      0388  IMM:    MOVL    R0,(R5)                         ;SET MODE/REGISTER ON STACK
        FC72'   30      038B          BSBW    MAC$STOIM                       ;STORE MODE/REGISTER IN OBJECT CODE
   50   04 A8   9A      038E          MOVZBL  4(R8),R0                        ; Get type of reference (byte, word, long)
50    50  01    78      0392          ASHL    #1,R0,R0                        ; Convert to number of chars required
      52  50    C0      0396          ADDL2   R0,R2                           ; add to initial size
        027C    30      0399          BSBW    OPND_CHECK                      ; Check listing space for operand
                        039C          $MAC_LIST_BYTE #1                       ;LIST MODE/REGISTER
   01   04 A8   91      03A2          CMPB    4(R8),#1                        ;BYTE FORMAT?
          21    12      03A6          BNEQ    30$                             ;IF NEQ NO
                        03A8  ;
                        03A8  ; BYTE FORMAT IMMEDIATE REFERENCE
                        03A8  ;
   0B 6B  07    E1      03A8          BBC     #FLG$V_EXPOPT,(R11),10$ ;BRANCH IF NOT OPTIMIZED EXPRESSION
        FC51'   30      03AC          BSBW    MAC$CK_BYT_TRUN                 ;YES--CHECK TRUNCATION
```

```
50   65   90  03AF   626          MOVB    (R5),R0                    ;GET THE VALUE
     FC4B' 30  03B2   627          BSBW    MAC$STOIM                  ;STORE INTO OBJECT CODE
     0B   11  03B5   628          BRB     20$
                03B7   629 ;
                03B7   630 ; NON-OPTIMIZED BYTE
                03B7   631 ;
0000'DF  27   90  03B7   632 10$:     MOVB    #^A/'/,@W^MAC$GL_LIST_PTR ;PUT QUOTE INTO LISTING
                03BC   633          $OBJ_CHKBYT #TIR$C_STO_B           ;STORE BYTE COMMAND
                03C2   634 20$:     $MAC_LIST_BYTE #1                  ;LIST BYTE VALUE
                05  03C8   635          RSB
                03C9   636 ;
                03C9   637 ; NOT BYTE REFERENCE
                03C9   638 ;
02   04 A8   91  03C9   639 30$:     CMPB    4(R8),#2                   ;WORD FORMAT?
     1E   12  03CD   640          BNEQ    60$                        ;IF NEQ NO--LONGWORD
08 6B  07   E1  03CF   641          BBC     #FLG$V_EXPOPT,(R11),40$ ;YES--BRANCH IF NOT OPTIMIZED
     FC2A' 30  03D3   642          BSBW    MAC$CK_WRD_TRUN            ;OPTIMIZED--CHECK TRUNCATION
54   02   9A  03D6   643          MOVZBL  #2,R4                      ;SET COUNT
     19   11  03D9   644          BRB     65$                        ;GO STORE BYTES
                03DB   645 ;
                03DB   646 ; NON-OPTIMIZED WORD REFERENCE
                03DB   647 ;
0000'DF  27   90  03DB   648 40$:     MOVB    #^A/'/,@W^MAC$GL_LIST_PTR ;STORE QUOTE IN LISTING
                03E0   649          $OBJ_CHKBYT #TIR$C_STO_W           ;STORE WORD COMMAND
                03E6   650 50$:     $MAC_LIST_BYTE #2                  ;LIST WORD
                05  03EC   651          RSB
                03ED   652 ;
                03ED   653 ; LONGWORD FORMAT
                03ED   654 ;
16 6B  07   E1  03ED   655 60$:     BBC     #FLG$V_EXPOPT,(R11),80$ ;BRANCH IF NOT OPTIMIZED
54   04   9A  03F1   656          MOVZBL  #4,R4                      ;GET LOOP COUNT
                03F4   657 ;
                03F4   658 ; EMIT VALUE TO OBJECT CODE, 1 BYTE AT A TIME
                03F4   659 ;
     54   DD  03F4   660 65$:     PUSHL   R4                         ;SAVE BYTE COUNT
50   85   90  03F6   661 70$:     MOVB    (R5)+,R0                   ;GET NEXT BYTE
     FC04' 30  03F9   662          BSBW    MAC$STOIM                  ;STORE INTO OBJECT CODE
   F7 54  F5  03FC   663          SOBGTR  R4,70$                     ;LOOP FOR ALL
54   8E   D0  03FF   664          MOVL    (SP)+,R4                   ;RESTORE BYTE COUNT
55   54   C2  0402   665          SUBL2   R4,R5                      ;ADJUST THE STACK POINTER
     0E   11  0405   666          BRB     90$
                0407   667 ;
                0407   668 ; NON-OPTIMIZED LONGWORD
                0407   669 ;
0000'DF  27   90  0407   670 80$:     MOVB    #^A/'/,@W^MAC$GL_LIST_PTR ;STORE QUOTE IN LISTING
                040C   671          $OBJ_CHKBYT #TIR$C_STO_LW          ;STORE LONGWORD COMMAND
54   04   9A  0412   672          MOVZBL  #4,R4                      ;SET TO LIST 4 BYTES
                0415   673 90$:     $MAC_LIST_BYTE R4                  ;LIST LONGWORD
                05  041B   674          RSB
```

```
                        041C   676 ;++
                        041C   677 ; FUNCTIONAL DESCRIPTION:
                        041C   678 ;
                        041C   679 ;       STORE POSITION INDEPENDENT REFERENCES.
                        041C   680 ;
                        041C   681 ;--
                        041C   682
                        041C   683 REF_3:
          01F9   30     041C   684          BSBW    OPND_CHECK                  ; Check listing space for operand
            57   D7     041F   685          DECL    R7                          ;IGNORE MODE/REGISTER
            75   D5     0421   686          TSTL    -(R5)                       ;BACK R5 UP TOO
     50 6B  07   E1     0423   687          BBC     #FLG$V_EXPOPT,(R11),30$     ;BRANCH IF NOT OPTIMIZED REFERENCE
     53     06   9A     0427   688          MOVZBL  #TIR$C_STA_PL,R3            ;ASSUME LONGWORD
     52     04   9A     042A   689          MOVZBL  #4,R2                       ;...
        02  A5   B5     042D   690          TSTW    2(R5)                       ;CAN WE OPTIMIZE SOME MORE?
            19   12     0430   691          BNEQ    10$                         ;IF NEQ NO
            65   B5     0432   692          TSTW    (R5)                        ;MAYBE--CHECK LOW WORD
            15   19     0434   693          BLSS    10$                         ;IF LSS CANNOT OPTIMIZE
     53     05   9A     0436   694          MOVZBL  #TIR$C_STA_PW,R3            ;CHANGE TO STACK WORD
     52     02   9A     0439   695          MOVZBL  #2,R2                       ;...
        01  A5   95     043C   696          TSTB    1(R5)                       ;WE'RE DOING GOOD--OPTIMIZE TO BYTE?
            0A   12     043F   697          BNEQ    10$                         ;IF NEQ NO
            65   95     0441   698          TSTB    (R5)                        ;MAYBE
            06   19     0443   699          BLSS    10$                         ;IF LSS NO
     53     04   9A     0445   700          MOVZBL  #TIR$C_STA_PB,R3            ;YES--CHANGE TO BYTE
     52     01   9A     0448   701          MOVZBL  #1,R2                       ;...
                        044B   702 10$:     $OBJ_CHKBYT R3                      ;OUTPUT COMMAND
     0000'CF    DD      0451   703          PUSHL   W^MAC$GL_PSECT              ;USE TEMP TO WRITE ADJUSTED SEGMENT #
            0A   13     0455   704          BEQL    15$                         ;DON'T FIDDLE WITH THE ABS PSECT!
            07   E0     0457   705          BBS     #SYM$V_REF,-                ;IF THE BLANK PSECT HASN'T BEEN REF'D
02 00000009'EF          0459   706                  PSECT$BLANK+SYM$W_FLAG,15$ ;IT'LL BE REMOVED, DECR THE SEG#
            6E   D7     045F   707          DECL    (SP)                        ;TO PRESERVE THE PSECT ALIGNMENT
                        0461   708 15$:     $OBJ_OUTBYT (SP)                    ;OUTPUT CURRENT PSECT NUMBER
            8E   D5     0467   709          TSTL    (SP)+                       ;AND CLEAN UP THE STACK.
            52   DD     0469   710          PUSHL   R2                          ;SAVE BYTE COUNT
                        046B   711 20$:     $OBJ_OUTBYT (R5)+                   ;EMIT ONE BYTE OF VALUE
        F7  52   F5     0471   712          SOBGTR  R2,20$                      ;LOOP FOR ALL
     55     8E   C2     0474   713          SUBL2   (SP)+,R5                    ;ADJUST STACK POINTER
                        0477   714 30$:     $OBJ_CHKBYT #TIR$C_STO_PICR         ;GENERATE POS. IND. REF.
     50   46 8F   9A     047D   715          MOVZBL  #^A/F/,R0                   ;PRINT 'F'
          FB7C'   30     0481   716          BSBW    MAC$LST_CHAR
     50   47 8F   9A     0484   717          MOVZBL  #^A/G/,R0                   ;PRINT 'G'
          FB75'   30     0488   718          BSBW    MAC$LST_CHAR
     50     27   9A     048B   719          MOVZBL  #^A/'/,R0                   ;PRINT QUOTE
          FB6F'   30     048E   720          BSBW    MAC$LST_CHAR
                        0491   721          $INC_PC                             ;COUNT MODE/REGISTER
                        0495   722          $MAC_LIST_BYTE #4                   ;PRINT LONGWORD ON STACK
                   05   049B   723          RSB
```

```
                        049C   725 ;++
                        049C   726 ; FUNCTIONAL DESCRIPTION:
                        049C   727 ;
                        049C   728 ;       PROCESS INDEX MODE REFERENCES
                        049C   729 ;
                        049C   730 ;--
                        049C   731
                        049C   732 REF_4:                                  ;INDEXED REFERENCE
          50   01 A8   9A 049C   733         MOVZBL  1(R8),R0              ; Get mode of reference
     52   FE90 CF40    80 04A0   734         ADDB2   W^REF_SIZE[R0],R2     ; and add bytes required
               016F    30 04A6   735         BSBW    OPND_CHECK            ; Check listing space for operand
          50   65      90 04A9   736         MOVB    (R5),R0               ;GET MODE/REGISTER
               FB51'   30 04AC   737         BSBW    MAC$STOIM             ;STORE INTO OBJECT CODE
                       04AF   738         $MAC_LIST_BYTE #1             ;LIST MODE/REGISTER
          0000'CF      D6 04B5   739         INCL    W^MAC$GL_LIST_PTR     ; Back up over space
          00 6B   28   E2 04B9   740         BBSS    #FLG$V_OPNDCHK,(R11),10$ ; Flag that operand check is done
                       04BD   741 10$:
          68   01 A8   90 04BD   742         MOVB    1(R8),(R8)            ;MOVE MODES
               01 A8   94 04C1   743         CLRB    1(R8)                 ;
       02 A8   03 A8   90 04C4   744         MOVB    3(R8),2(R8)           ;
               03 A8   94 04C9   745         CLRB    3(R8)
7E   6B   FFFFFF7F 8F   CB 04CC   746         BICL3   #^C<FLG$M_EXPOPT>,(R11),-(SP) ;SAVE OLD EXPOPT FLAG
               FDB5    30 04D4   747         BSBW    P2$XREF               ;CALL XREF FOR REST OF OPERAND
          6B   8E      C8 04D7   748         BISL2   (SP)+,(R11)           ;SET EXPOPT FLAG IF SET BEFORE
                       05 04DA   749         RSB                           ;ALL DONE
                       04DB   750
                       04DB   751 REF_5:                                  ;REGISTER
                       04DB   752 REF_6:                                  ;REGISTER DEFERRED
                       04DB   753 REF_7:                                  ;AUTODECREMENT
                       04DB   754 REF_8:                                  ;AUTOINCREMENT
                       04DB   755 REF_9:                                  ;AUTOINCREMENT DEFERRED
               013A    30 04DB   756         BSBW    OPND_CHECK            ; Check listing space for operand
          50   65      90 04DE   757         MOVB    (R5),R0               ;GET MODE/REGISTER
               FB1C'   30 04E1   758         BSBW    MAC$STOIM             ;STORE IN OBJECT CODE
                       04E4   759         $MAC_LIST_BYTE #1             ;LIST MODE/REGISTER
                       05 04EA   760         RSB
```

```
                        04EB   762 ;++
                        04EB   763 ; FUNCTIONAL DESCRIPTION:
                        04EB   764 ;
                        04EB   765 ;       THIS ROUTINE PROCESSES 8-BIT DISPLACEMENT REFERENCES
                        04EB   766 ;
                        04EB   767 ;--
                        04EB   768
                        04EB   769 REF_A:                                   ;8-BIT DISPLACEMENT
                        04EB   770 REF_B:                                   ;8-BIT DEFERRED DISPLACEMENT
          012A    30    04EB   771         BSBW    OPND_CHECK               ; Check listing space for operand
       50   65    90    04EE   772         MOVB    (R5),R0                  ;GET MODE/REGISTER
          FB0C'   30    04F1   773         BSBW    MAC$STOIM                ;OUTPUT TO OBJECT CODE
                        04F4   774         $MAC_LIST_BYTE #1                ;LIST MODE/REGISTER
    0F    02 A8   91    04FA   775         CMPB    2(R8),#15                ;PC RELATIVE REFERENCE?
           2F     12    04FE   776         BNEQ    20$                      ;IF NEQ NO
    1E 6B  07     E1    0500   777         BBC     #FLG$V_EXPOPT,(R11),10$  ;YES--BRANCH IF NOT OPTIMIZED
                        0504   778         $VPUSH  W^MAC$GL_PC              ;OPTIMIZED--STACK PC
           85     D5    050E   779         TSTL    (R5)+                    ;BUMP R5
           65     D6    0510   780         INCL    (R5)                     ;ADD 1 FOR OPERAND
          FAEB'   30    0512   781         BSBW    P1$ARITH_SUB             ;
           75     D5    0515   782         TSTL    -(R5)                    ;BACK UP MY POINTER
          FAE6'   30    0517   783         BSBW    MAC$CK_SBY_TRUN          ;CHECK FOR TRUNCATION
       50   65    90    051A   784         MOVB    (R5),R0                  ;GET BYTE
          FAE0'   30    051D   785         BSBW    MAC$STOIM                ;AND STORE IN OBJECT CODE
           27     11    0520   786         BRB     40$
                        0522   787 ;
                        0522   788 ; NON-OPTIMIZED PC-RELATIVE
                        0522   789 ;
  0000'DF  27     90    0522   790 10$:    MOVB    #^A/'/,@W^MAC$GL_LIST_PTR ;PUT QUOTE INTO LISTING
                        0527   791         $OBJ_CHKBYT #TIR$C_STO_BD        ;STORE BYTE DISPLACED
           1A     11    052D   792         BRB     40$
                        052F   793 ;
                        052F   794 ; NOT PC-RELATIVE
                        052F   795 ;
    0B 6B  07     E1    052F   796 20$:    BBC     #FLG$V_EXPOPT,(R11),30$ ;BRANCH IF NOT OPTIMIZED
          FACA'   30    0533   797         BSBW    MAC$CK_SBY_TRUN          ;YES--CHECK FOR TRUNCATION
       50   65    90    0536   798         MOVB    (R5),R0                  ;GET THE BYTE
          FAC4'   30    0539   799         BSBW    MAC$STOIM                ;STORE INTO OBJECT CODE
           0B     11    053C   800         BRB     40$
                        053E   801 ;
                        053E   802 ; NOT PC-RELATIVE, NOT OPTIMIZED
                        053E   803 ;
  0000'DF  27     90    053E   804 30$:    MOVB    #^A/'/,@W^MAC$GL_LIST_PTR ;PUT QUOTE INTO LISTING
                        0543   805         $OBJ_CHKBYT #TIR$C_STO_SB        ; Store signed byte
                        0549   806 ;
                        0549   807 ; FINISH PROCESSING
                        0549   808 ;
                        0549   809 40$:    $MAC_LIST_BYTE #1                ;LIST THE BYTE
                  05    054F   810         RSB
```

```
                        0550   ;++
                        0550   ; FUNCTIONAL DESCRIPTION:
                        0550   ;
                        0550   ;       PROCESS 16-BIT DISPLACEMENT AND 16-BIT DEFERRED DISPLACEMENT
                        0550   ;       REFERENCES.
                        0550   ;
                        0550   ;--
                        0550
                        0550   REF_C:                                      ;16-BIT DISPLACEMENT
                        0550   REF_D:                                      ;16-BIT DEFERRED DISPLACEMENT
          00C5   30     0550           BSBW    OPND_CHECK                 ; Check listing space for operand
     50     65   90     0553           MOVB    (R5),R0                    ;GET MODE/REGISTER
          FAA7'  30     0556           BSBW    MAC$STOIM                  ;WRITE TO OBJECT CODE
                        0559           $MAC_LIST_BYTE #1                  ;LIST MODE/REGISTER
     0F     54   91     055F           CMPB    R7,#15                     ;PC-RELATIVE?
            2A   12     0562           BNEQ    20$                        ;IF NEQ NO
  19 6B     07   E1     0564           BBC     #FLG$V_EXPOPT,(R11),10$    ;YES--BRANCH IF NOT OPTIMIZED
                        0568           $VPUSH  W^MAC$GL_PC                ;OPTIMIZED--STACK PC
            85   D5     0572           TSTL    (R5)+                      ;BUMP THE POINTER
     65     02   C0     0574           ADDL2   #2,(R5)                    ;PLUS SIZE OF OPERAND
          FA86'  30     0577           BSBW    P1$ARITH_SUB
            75   D5     057A           TSTL    -(R5)                      ;BACK UP POINTER
          FA81'  30     057C           BSBW    MAC$CK_SWD_TRUN            ;CHECK FOR TRUNCATION
            14   11     057F           BRB     30$                        ;GO EMIT CODE AND EXIT
                        0581   ;
                        0581   ; PC-RELATIVE, NOT OPTIMIZED
                        0581   ;
0000'DF     27   90     0581   10$:    MOVB    #^A/'/,@W^MAC$GL_LIST_PTR ;PUT QUOTE IN LISTING
                        0586           $OBJ_CHKBYT #TIR$C_STO_WD          ;STORE WORD DISPLACED
            22   11     058C           BRB     50$
                        058E   ;
                        058E   ; NOT PC-RELATIVE
                        058E   ;
  13 6B     07   E1     058E   20$:    BBC     #FLG$V_EXPOPT,(R11),40$    ;BRANCH IF NOT OPTIMIZED
          FA6B'  30     0592           BSBW    MAC$CK_SWD_TRUN            ;YES--CHECK TRUNCATION
     50     85   90     0595   30$:    MOVB    (R5)+,R0                   ;GET A BYTE
          FA65'  30     0598           BSBW    MAC$STOIM                  ;SEND IT TO OBJECT CODE
     50     65   90     059B           MOVB    (R5),R0                    ;GET SECOND BYTE
          FA5F'  30     059E           BSBW    MAC$STOIM                  ;SEND IT TO OBJECT CODE
            75   95     05A1           TSTB    -(R5)                      ;KEEP R5 POINTING TO TOP OF STACK
            0B   11     05A3           BRB     50$
                        05A5   ;
                        05A5   ; NOT PC-RELATIVE, NOT OPTIMIZED
                        05A5   ;
0000'DF     27   90     05A5   40$:    MOVB    #^A/'/,@W^MAC$GL_LIST_PTR ;PUT QUOTE IN LISTING
                        05AA           $OBJ_CHKBYT #TIR$C_STO_SW          ;STORE SIGNED WORD
                        05B0   50$:    $MAC_LIST_BYTE #2                  ;LIST THE WORD
                  05    05B6           RSB
```

```
                        05B7   ;++
                        05B7   ; FUNCTIONAL DESCRIPTION:
                        05B7   ;
                        05B7   ;       PROCESS 32-BIT DISPLACEMENT AND 32-BIT DEFERRED DISPLACEMENT
                        05B7   ;       REFERENCES
                        05B7   ;
                        05B7   ;--
                        05B7
                        05B7   REF_E:                                          ;32-BIT DISPLACEMENT
                        05B7   REF_F:                                          ;32-BIT DEFERRED DISPLACEMENT
          005E    30    05B7           BSBW    OPND_CHECK                      ; Check listing space for operand
     50     65    90    05BA           MOVB    (R5),R0                         ;GET MODE/REGISTER
          FA40'   30    05BD           BSBW    MAC$STOIM                       ;WRITE TO OBJECT CODE
                        05C0           $MAC_LIST_BYTE #1                       ;LIST MODE/REGISTER
     0F     54    91    05C6           CMPB    R4,#15                          ;PC-RELATIVE?
            27    12    05C9           BNEQ    20$                             ;IF NEQ NO
  16 6B     07    E1    05CB           BBC     #FLG$V_EXPOPT,(R11),10$         ;YES--BRANCH IF NOT OPTIMIZED
                        05CF           $VPUSH  W^MAC$GL_PC                     ;OPTIMIZED--STACK PC
            85    D5    05D9           TSTL    (R5)+                           ;BUMP THE POINTER
     65     04    C0    05DB           ADDL2   #4,(R5)                         ;PLUS SIZE OF OPERAND
          FA1F'   30    05DE           BSBW    P1$ARITH_SUB
            75    D5    05E1           TSTL    -(R5)                           ;BACKUP THE POINTER
            11    11    05E3           BRB     30$                             ;GO EMIT CODE AND EXIT
                        05E5   ;
                        05E5   ; PC-RELATIVE, NOT OPTIMIZED
                        05E5   ;
  0000'DF   27    90    05E5   10$:    MOVB    #^A/'/,@W^MAC$GL_LIST_PTR       ;PUT QUOTE IN LISTING
                        05EA           $OBJ_CHKBYT #TIR$C_STO_LD               ;STORE LONGWORD DISPLACED
            1F    11    05F0           BRB     60$
                        05F2   ;
                        05F2   ; NOT PC-RELATIVE
                        05F2   ;
  10 6B     07    E1    05F2   20$:    BBC     #FLG$V_EXPOPT,(R11),50$         ;BRANCH IF NOT OPTIMIZED
     53     04    9A    05F6   30$:    MOVZBL  #4,R3                           ;SET A LOOP COUNT
     50     85    90    05F9   40$:    MOVB    (R5)+,R0                        ;GET NEXT BYTE
          FA01'   30    05FC           BSBW    MAC$STOIM                       ;STORE IN OBJECT CODE
        F7 53     F5    05FF           SOBGTR  R3,40$                          ;LOOP FOR 4 BYTES
            75    D5    0602           TSTL    -(R5)                           ;KEEP R5 POINTING TO TOP OF STACK
            0B    11    0604           BRB     60$                             ;AND EXIT
                        0606   ;
                        0606   ; NOT PC-RELATIVE, NOT OPTIMIZED
                        0606   ;
  0000'DF   27    90    0606   50$:    MOVB    #^A/'/,@W^MAC$GL_LIST_PTR       ;PUT QUOTE IN LISTING
                        060B           $OBJ_CHKBYT #TIR$C_STO_LW               ;STORE LONGWORD
                        0611   60$:    $MAC_LIST_BYTE #4                       ;LIST THE WORD
                  05    0617           RSB
```

```
                                0618    908             .SBTTL  OPND_CHECK
                                0618    909 ;
                                0618    910 ; ++
                                0618    911 ; Functional description:
                                0618    912 ;
                                0618    913 ;           This routine checks that there is space on the current
                                0618    914 ;           listing line to write the next operand
                                0618    915 ;
                                0618    916 ; Inputs:
                                0618    917 ;
                                0618    918 ;           R2 = Number of characters required to write operand
                                0618    919 ;
                                0618    920 ; Outputs:
                                0618    921 ;
                                0618    922 ;           None
                                0618    923 ;
                                0618    924 ;
                                0618    925 OPND_CHECK:
            33 6B    28    E4   0618    926             BBSC    #FLG$V_OPNDCHK,(R11),30$ ; Branch if check has been performed
0000'CF   00000000'8F    C3   061C    927             SUBL3   #MAC$AB_LST_END, -       ; Compute number of characters still
                     50         0625    928                     W^MAC$GL_LIST_PTR,R0     ; available on line
            03 6B    27    E1   0626    929             BBC     #FLG$V_UPDFIL,(R11),10$  ; If updating file
               50    10    C2   062A    930             SUBL2   #AUD$K_SIZE,R0           ; subtract size of audit trail
                                062D    931 10$:
00000000'8F   0000'CF    D1   062D    932             CMPL    W^MAC$GL_LIST_PTR, -     ; First operand?
                                0636    933                     #MAC$AB_LST_OP1
                     08    13   0636    934             BEQL    20$                      ; Yes if EQL
               52    02    C0   0638    935             ADDL2   #2,R2                    ; Two more characters required for
                                063B    936                                              ; separating spaces
          0000'CF    02    C2   063B    937             SUBL2   #2,W^MAC$GL_LIST_PTR     ; Move pointer back two spaces
                                0640    938 20$:
               52    50    D1   0640    939             CMPL    R0,R2                    ; Required characters available?
                     0A    18   0643    940             BGEQ    30$                      ; Yes if GEQ
                  F9B8'    30   0645    941             BSBW    MAC$WRTLST               ; No, Write line
0000'CF   0000'CF    9E   0648    942             MOVAB   W^MAC$AB_LST_OP1, -      ; and reset pointer
                                064F    943                     W^MAC$GL_LIST_PTR
                                064F    944 30$:
                     05         064F    945             RSB
```

```
                        0650          .SBTTL  P2$END  THE END OF PASS 2
                        0650
                        0650  ;++
                        0650  ; FUNCTIONAL DESCRIPTION:
                        0650  ;
                        0650  ;       THIS ROUTINE IS CALLED WHEN THE END COMMAND IS ENCOUNTERED IN
                        0650  ;       THE INTERMEDIATE FILE.  THE PC IS SET, THE FINAL RECORDS ARE
                        0650  ;       WRITTEN OUT TO THE OBJECT MODULE, AND THEN A JMP IS EXECUTED
                        0650  ;       TO THE END OF PASS 2.
                        0650  ;
                        0650  ;--
                        0650
                        0650  P2$END::
          F9AD'   30    0650          BSBW    MAC$SET_PC                  ;UPDATE PC
          F9AA'   30    0653          BSBW    MAC$WRTLST                  ;WRITE OUT LISTING
          F9A7'   30    0656          BSBW    MAC$FINISH_ASM              ;FINISH ASSEMBLY, WRITE OUT
                        0659                                              ; TRACEBACK, DEBUG, AND SYMBOLS
0000'CF     03    9A    0659          MOVZBL  #OBJ$C_EOM,W^MAC$GL_RECTYP ;SET END OF MODULE RECORD TYPE
          F99F'   30    065E          BSBW    MAC$WRTOBJ                  ;WRITE OUT ANY PREVIOUS RECORD
      50    00    D0    0661          MOVL    #OBJ$C_EOM_OK,R0            ;ASSUME MODULE OK
      56    01    D0    0664          MOVL    #1,R6                       ;ASSUME SUCCESS
   0E 6B    30    E0    0667          BBS     #FLG$V_EXTERR,(R11),5$      ;ANY EXTERNAL FACILITY ERRORS?
   1A 6B    31    E0    066B          BBS     #FLG$V_EXTWRN,(R11),8$      ;ANY EXTERNAL FACILTY WARNINGS?
      0000'CF    C9    066F          BISL3   W^MAC$GL_ERRCT,-            ;WERE THERE ANY ERRORS?
51    0000'CF          0673                  W^MAC$GL_WARNCT,R1            ;OR WARNINGS?
            16    13    0677          BEQL    10$                         ;IF EQL NO--GOOD MODULE
      50    02    D0    0679  5$:     MOVL    #OBJ$C_EOM_ERR,R0           ;ASSUME SEVERE ERRORS
      56    02    D0    067C          MOVL    #2,R6                       ;...
      0000'CF     D5    067F          TSTL    W^MAC$GL_ERRCT              ;WERE THERE ERRORS?
            0A    12    0683          BNEQ    10$                         ;IF NEQ YES--GO SET STATUS
   06 6B    30    E0    0685          BBS     #FLG$V_EXTERR,(R11),10$     ;MAKE SURE ABOUT EXTERNAL ERRORS TOO!
      50    01    D0    0689  8$:     MOVL    #OBJ$C_EOM_WRN,R0           ;NO--SET WARNING STATUS
      56    00    D0    068C          MOVL    #0,R6                       ;...
0000'CF     56    D0    068F  10$:    MOVL    R6,W^MAC$GL_STATUS          ;SAVE STATUS FOR EXITING
                        0694          $OBJ_OUTBYT_0 R0                    ;STORE STATUS
55    0000'CF     D0    0697          MOVL    W^MAC$GL_XFRADR,R5          ;GET TRANSFER ADDRESS
            20    13    069C          BEQL    30$                         ;IF EQL NO TRANSFER ADDRESS
                        069E  ;
                        069E  ; OUTPUT TRANSFER ADDRESS INFORMATION
                        069E  ;
   50    0C A5    D0    069E          MOVL    SYM$B_SEG(R5),R0            ;GET THE SYMBOL'S PSECT NUMBER.
            0A    13    06A2          BEQL    15$                         ;DON'T FIDDLE WITH THE ABSOLUTE PSECT!
            07    E0    06A4          BBS     #SYM$V_REF,-                ;IF THE BLANK PSECT IS NOT REFERENCED
02 00000009'EF          06A6                  PSECT$BLANK+SYM$W_FLAG,15$ ;IT'LL BE REMOVED SO DECR THE PSECT#
            50    D7    06AC          DECL    R0                          ;TO MAINTAIN CORRECT PSECT ALIGNMENT.
                        06AE  15$:    $OBJ_OUTBYT_0 R0                    ;OUTPUT SYMBOL'S PSECT
   55    05 A5    9E    06B1          MOVAB   SYM$L_VAL(R5),R5            ;POINT TO SYMBOL VALUE
      54    04    9A    06B5          MOVZBL  #4,R4                       ;SET COUNT
                        06B8  20$:    $OBJ_OUTBYT_0 (R5)+                 ;OUTPUT BYTE OF THE VALUE
         FA 54    F5    06BB          SOBGTR  R4,20$                      ;LOOP FOR 4 BYTES
      0000'CF     D4    06BE  30$:    CLRL    W^MAC$GL_XFRADR             ;RESET TRANSFER ADDRESS POINTER
          F93B'   30    06C2          BSBW    MAC$WRTOBJ                  ;WRITE OUT FINAL RECORD TO OBJECT FILE
          F938'   31    06C5          BRW     W^MAC$PASS_2_END            ;PASS 2 IS COMPLETED
                        06C8
                        06C8          .END
```

```
ADM$_ABSOLUTE  = 00000002
ADM$_BYTE_DISP = 0000000A
ADM$_DFBYTEDISP= 0000000B
ADM$_DFLONGDISP= 0000000F
ADM$_DFRAUTOINC= 00000009
ADM$_DFWORDDISP= 0000000D
ADM$_IMMEDIATE = 00000001
ADM$_INDEX     = 00000004
ADM$_LITERAL   = 00000000
ADM$_LONG_DISP = 0000000E
ADM$_MAXMOD    = 0000000F
ADM$_PIC       = 00000003
ADM$_REGAUTODEC= 00000007
ADM$_REGAUTOINC= 00000008
ADM$_REGISTER  = 00000005
ADM$_RRIND     = 00000006
ADM$_WORD_DISP = 0000000C
AUD$R_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
CR             = 0000000D
EOM$C_ABORT    = 00000003
EOM$C_ERROR    = 00000002
EOM$C_SUCCESS  = 00000000
EOM$C_WARNING  = 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
IMM            = 00000388 R     03
INP$K_BUFSIZ   = 000003E8
INT$K_BUFSIZ   = 000013F4
INT$K_BUFWRN   = 00001390
LST$K_BUFSIZ   = 00000086
LST$K_L_P_PAGE = 0000003C
LST$K_TITLE_SIZ= 00000028
MAC$AB_LST_END    ********   X   03
MAC$AB_LST_OP1    ********   X   03
MAC$AB_LST_OPR    ********   X   03
MAC$AL_VALSTACK   ********   X   03
MAC$CHRBYT        ********   X   03
MAC$CK_BYT_TRUN   ********   X   03
MAC$CK_SBY_TRUN   ********   X   03
MAC$CK_SWD_TRUN   ********   X   03
MAC$CK_WRD_TRUN   ********   X   03
MAC$FINISH_ASM    ********   X   03
MAC$GL_ERRCT      ********   X   03
MAC$GL_LIST_PTR   ********   X   03
MAC$GL_MOPNUM     ********   X   03
MAC$GL_OPSIZE     ********   X   03
MAC$GL_PC         ********   X   03
MAC$GL_PSECT      ********   X   03
MAC$GL_RECTYP     ********   X   03
MAC$GL_STATUS     ********   X   03
MAC$GL_WARNCT     ********   X   03
```

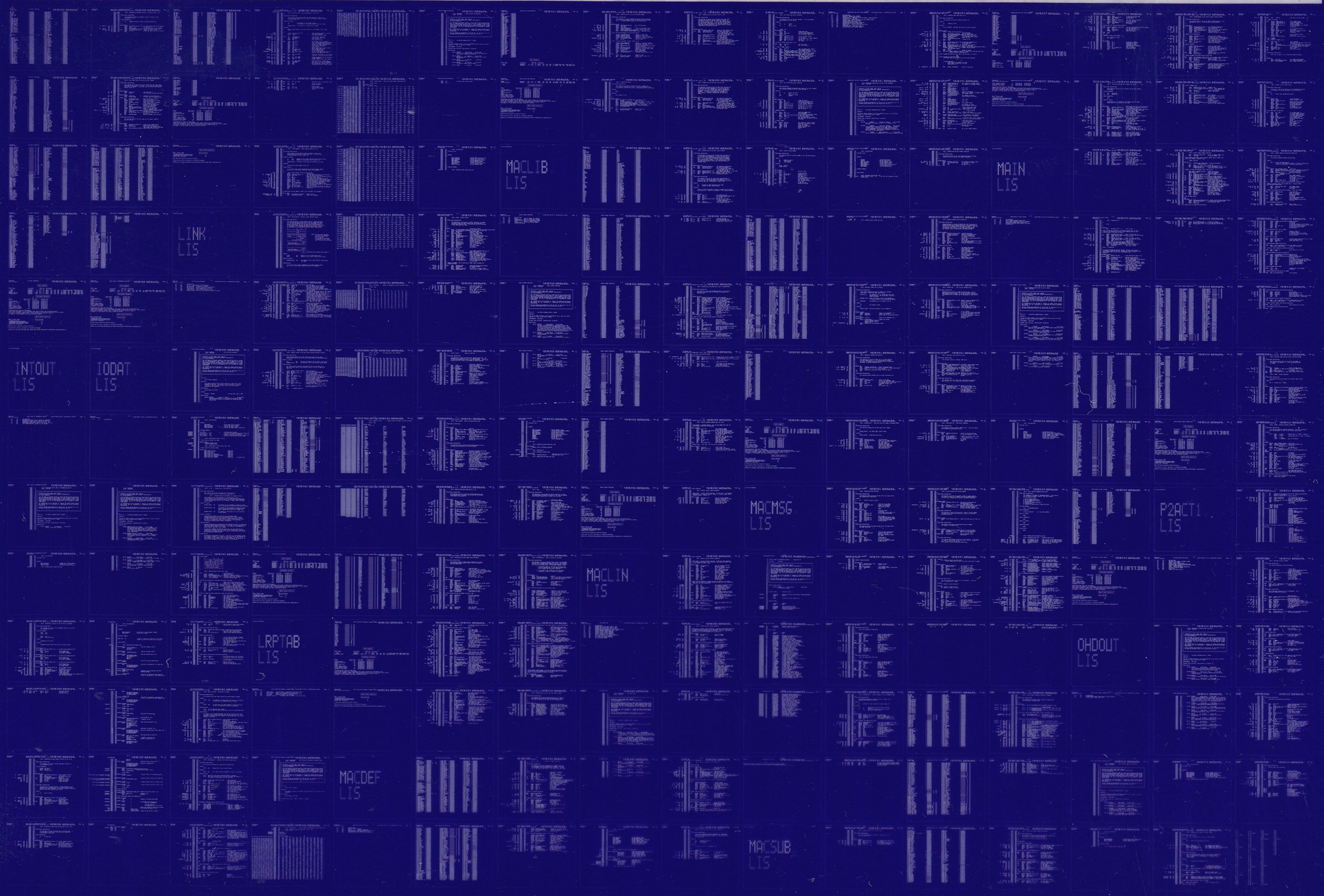
```
MAC$GL_XFRADR     ********   X   03      PSC$M_LIB        = 00000002
MAC$LIST_BYTES    ********   X   03      PSC$M_LONG       = 00004800
MAC$LST_CHAR      ********   X   03      PSC$M_NOEXE      = FFFFFFBF
MAC$OUTOBJ        ********   X   03      PSC$M_NOPIC      = FFFFFFFE
MAC$PASS_2_END    ********   X   03      PSC$M_NORD       = FFFFFF7F
MAC$PASS_2_ERR    ********   X   03      PSC$M_NOSHR      = FFFFFFDF
MAC$SET_PC        ********   X   03      PSC$M_NOVEC      = FFFFFDFF
MAC$STOIM         ********   X   03      PSC$M_NOWRT      = FFFFFEFF
MAC$SYMNAMOUT     ********   X   03      PSC$M_OVR        = 00000004
MAC$WRTLST        ********   X   03      PSC$M_PAGE       = 00006400
MAC$WRTOBJ        ********   X   03      PSC$M_PIC        = 00000001
MAC$_BRDESTRANG= 007D9052                 PSC$M_QUAD       = 00004C00
MAC$_DATATRUNC = 007D8800                 PSC$M_RD         = 00000080
MAC$_ILLMODE   = 007D9102                 PSC$M_REL        = 00000008
MAC$_ILLREGNUM = 007D9122                 PSC$M_SHR        = 00000020
MAC$_SYMOUTPHAS= 007D91F2                 PSC$M_USR        = FFFFFFFD
MAC$_UNDEFSYM  = 007D9212                 PSC$M_VEC        = 00000200
MAC_SUBSYS     = 0000007D                 PSC$M_WORD       = 00004400
OBJ$C_EOM      = 00000003                 PSC$M_WRT        = 00000180
OBJ$C_EOM_ABORT= 00000003                 PSC$S_ALIGNMENT= 00000004
OBJ$C_EOM_ERR  = 00000002                 PSC$V_ALIGNFLG = 0000000E
OBJ$C_EOM_OK   = 00000000                 PSC$V_ALIGNMENT= 0000000A
OBJ$C_EOM_WRN  = 00000001                 PSC$V_EXE        = 00000006
OBJ$K_BUFSIZ   = 00000200                 PSC$V_GBL        = 00000004
OPF$M_LASTOPR  = 00002000                 PSC$V_LIB        = 00000001
OPF$M_OPTEXP   = 00001000                 PSC$V_OVR        = 00000002
OPF$V_LASTOPR  = 0000000D                 PSC$V_PIC        = 00000000
OPF$V_OPTEXP   = 0000000C                 PSC$V_RD         = 00000007
OPND_CHECK        00000618 R       03     PSC$V_REL        = 00000003
P1$ARITH_SUB      ********     X   03     PSC$V_SHR        = 00000005
P2$BDST           000000FF RG       03     PSC$V_VEC        = 00000009
P2$END            00000650 RG       03     PSC$V_WRT        = 00000008
P2$LGLAB          00000218 RG       03     PSC$W_FLAG          00000009
P2$OP             00000230 RG       03     PSC$W_OPTIONS       0000000D
P2$REDEF          00000099 RG       03     PSECT$BLANK         ********   X   03
P2$REF            0000026D RG       03     RDX$V_BINARY     = 00000000
P2$STKG           00000000 RG       03     RDX$V_DECIMAL    = 00000002
P2$STKL           000000A6 RG       03     RDX$V_DOUBLE     = 00000005
P2$STKS           00000015 RG       03     RDX$V_FLOAT      = 00000004
P2$XREF           0000028C R        03     RDX$V_GFLOAT     = 00000006
PSC$B_NAME        00000004                 RDX$V_HEX        = 00000003
PSC$B_SEG         0000000C                 RDX$V_HFLOAT     = 00000007
PSC$B_UNUSED      0000000B                 RDX$V_OCTAL      = 00000001
PSC$K_BLKSIZ      00000013                 REF_0               00000345 R   D   03
PSC$K_NO_OPTNS = 0000000A                  REF_1               00000383 R   D   03
PSC$L_CURLOC      0000000F                 REF_2               0000037C R   D   03
PSC$L_LINK        00000000                 REF_3               0000041C R   D   03
PSC$L_MAXLGTH     00000005                 REF_4               0000049C R   D   03
PSC$M_ABS      = FFFFFFF7                  REF_5               000004DB R   D   03
PSC$M_ALIGNFLG = 00004000                  REF_6               000004DB R   D   03
PSC$M_ALLOPTNS = 000003FF                  REF_7               000004DB R   D   03
PSC$M_BYTE     = 00004000                  REF_8               000004DB R   D   03
PSC$M_CON      = FFFFFFFB                  REF_9               000004DB R   D   03
PSC$M_DEFAULT  = 000001C8                  REF_A               000004EB R   D   03
PSC$M_EXE      = 000000C0                  REF_B               000004EB R   D   03
PSC$M_GBL      = 00000010                  REF_C               00000550 R   D   03
PSC$M_LCL      = FFFFFFEF                  REF_D               00000550 R   D   03
REF_DISP          000002F5 R       03
REF_E             000005B7 R   D   03
REF_F             000005B7 R   D   03
REF_SIZE          00000335 R       03
REG$_PC        = 0000000F
SEMI           = 0000003B
STB$K_PG_MISS  = 0000000A
SYM$B_NAME        00000004
SYM$B_SEG         0000000C
SYM$B_TOKEN       0000000B
SYM$F_DEF      = 00000002
SYM$F_REL      = 00000008
SYM$F_UNI      = 00000004
SYM$F_VALIDATE = 00000010
SYM$F_WEAK     = 00000001
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
TIR$C_OPR_REDEF= 00000041
TIR$C_STA_GBL  = 00000000
TIR$C_STA_LW   = 00000003
```

```
TIR$C_STA_PB    = 00000004
TIR$C_STA_PL    = 00000006
TIR$C_STA_PW    = 00000005
TIR$C_STA_SB    = 00000001
TIR$C_STA_SW    = 00000002
TIR$C_STA_UB    = 00000007
TIR$C_STA_UW    = 00000008
TIR$C_STO_B     = 00000025
TIR$C_STO_BD    = 00000017
TIR$C_STO_L     = 00000016
TIR$C_STO_LD    = 00000019
TIR$C_STO_LI    = 0000001A
TIR$C_STO_LW    = 00000016
TIR$C_STO_PICR  = 0000001C
TIR$C_STO_SB    = 00000014
TIR$C_STO_SW    = 00000015
TIR$C_STO_W     = 00000026
TIR$C_STO_WD    = 00000018
X               = 00000010
X1              = 00000033
X2              = 00080000
```

```
+-----------------+
! 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
$ABS$                           00000013  (   19.)  02 (  2.)  NOPIC   USR   CON   ABS   LCL NOSHR   EXE   RD    WRT NOVEC BYTE
MAC$RO_CODE_P2                  000006C8  ( 1736.)  03 (  3.)  NOPIC   USR   CON   REL   GBL NOSHR   EXE   RD  NOWRT NOVEC LONG

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

Phase                     Page faults    CPU Time       Elapsed Time
-----                     -----------    --------       ------------
Initialization                     29    00:00:00.05    00:00:00.59
Command processing                106    00:00:00.40    00:00:02.11
Pass 1                            368    00:00:08.51    00:00:32.62
Symbol table sort                   1    00:00:00.90    00:00:03.60
Pass 2                            182    00:00:02.19    00:00:10.91
Symbol table output                31    00:00:00.16    00:00:00.61
Psect synopsis output               1    00:00:00.02    00:00:00.02
Cross-reference output              0    00:00:00.00    00:00:00.00
Assembler run totals              720    00:00:12.23    00:00:50.46

The working set limit was 1650 pages.
71071 bytes (139 pages) of virtual memory were used to buffer the intermediate code.
There were 50 pages of symbol table space allocated to hold 878 non-local and 85 local symbols.
1001 source lines were read in Pass 1, producing 24 object records in Pass 2.
44 pages of virtual memory were used to define 43 macros.
```

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

Macro library name                       Macros defined
------------------                       --------------
_$255$DUA28:[MACRO.OBJ]MACRO.MLB;1                  13
_$255$DUA28:[SYSLIB]STARLET.MLB;2                    4
TOTALS (all libraries)                              17

1060 GETS were required to define 17 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:P2ACT1/OBJ=OBJ$:P2ACT1 MSRC$:P2ACT1/UPDATE=(ENH$:P2ACT1)+LIB$:MACRO/LIB
```

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

MAIL

MAIL
MAP

P2DRVR
LIS

PARSER
LIS

RPTIRP
LIS

SCANER
LIS

TIMER
LIS

MAILCMDS
CLD

P2ACT2
LIS

SYMTAB
LIS