M 5

```
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**P2ACT2

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

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
```

MAC
V04

Table of contents

```
0000        .TITLE  MAC$P2ACT2 PASS 2 ACTION ROUTINES
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: 20-AUG-78
0000   ;
0000   ; MODIFIED BY:
0000   ;
0000   ;        V03-002 MTR0033         Mike Rhodes     22-Apr-1983
0000   ;                Allow 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 $MAC_TIRCMDDEF macro.
0000   ;
0000   ;        V02.13  PCG0002         Peter George    16-Apr-1981
0000   ;                Set and clear DBGOUT control flag.
0000   ;
0000   ;        V02.12  CNH0046         Chris Hume      2-Dec-1980
0000   ;                Taught P2$ERR (and relatives) to respect Parser Value Stack.
0000   ;
0000   ;        V02.11  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.  (P2ACT1.MAR 02.08)
```

```
0000    ;
0000    ;       V01.10  RN0029          R. Newland      20-Jan-1980
0000    ;               If error pointer is beyond column 132 do not print a
0000    ;               pointer.  This situation only occurs during macro
0000    ;               expansions which form a line longer than 132 characters.
0000    ;
0000    ;       V01.09  RN0023          R. Newland      3-Nov-1979
0000    ;               New message codes to get error messages from system
0000    ;               message file.
0000    ;
0000    ;       V01.07  RN0019          R. Newland      23-Oct-1979
0000    ;               Improve error pointer positioning
0000    ;
0000    ;       V01.16  RN0013          R. Newland      27-Sep-1979
0000    ;               Use new symbols for PSECT options processing
0000    ;       V01.05  RN0005          R. Newland      13-Aug-1979
0000    ;               Remove .ALIGN LONG statements
0000    ;
0000    ;       V01.04  RN0002          R. Newland      01-Feb-1979
0000    ;               Changes for Source Update Merge, form listing line
0000    ;               number differently for updated files, and SUM error
0000    ;               reporting
0000    ;       V0.08   RN0021          R. Newland      28-Oct-1979
0000    ;               Correct listing of .ENTRY register mask value.
0000    ;               SPR 11-26384
0000    ;
0000    ;       V01.03  0005            B. Schreiber    10-JAN-1979
0000    ;               Correct record too long error with .PRINT directive.
0000    ;
0000    ;--
```

```
                  0000          .SBTTL  DECLARATIONS
                  0000      ;
                  0000      ; INCLUDE FILES:
                  0000      ;
                  0000
                  0000      ;
                  0000      ; MACROS:
                  0000      ;
                  0000
                  0000          $MAC_OBJCODDEF                        ;DEFINE OBJECT CODE RECORD TYPES
                  0000          $MAC_SYMBLKDEF                        ;DEFINE SYMBOL BLOCK
                  0000          $MAC_CTLFLGDEF                        ;DEFINE CONTROL FLAGS
                  0000          $MAC_GENVALDEF                        ;DEFINE COMMON VALUES
                  0000          $MACMSGDEF                            ; Define message codes
                  0000          DEFSUMCBL                             ; Define SUM control block symbols
                  0000
                  0000      ;
                  0000      ; LOCAL DATA
                  0000      ;
              00000000          .PSECT  MAC$P2ACT2_DATA,NOEXE,LONG
                  0000
                  0000      MAC$GL_P2_ERRPT::
    00000000      0000          .LONG   0                             ;ERROR CODE GETS STORED HERE
    00000000      0004          .LONG   0                             ;POINTER TO WHERE ERROR HAPPENED
                  0008
              00000000          .PSECT  MAC$RO_CODE_P2,NOWRT,GBL,LONG
```

```
                        0000          .SBTTL  PASS 2 COMMANDS FOR PC
                        0000
                        0000  ;++
                        0000  ; FUNCTIONAL DESCRIPTION:
                        0000  ;
                        0000  ;       THIS ROUTINE STACKS THE CURRENT PC ON THE LINKER'S STACK.
                        0000  ;
                        0000  ;--
                        0000
                        0000  P2$STKPC::
                        0000          $OBJ_CHKBYT #TIR$C_STA_PL       ;COMMAND IS STACK PSECT BASE PLUS OFFSET
          0000'CF   DD  0006          PUSHL   W^MAC$GL_PSECT          ;PREPARE TO EMIT THE PSECT NUMBER
               0A   13  000A          BEQL    10$                     ;DON'T FIDDLE WITH THE ABS PSECT!
               07   E0  000C          BBS     #SYM$V_REF,-            ;IF THE BLANK PSECT HASN'T BEEN REF'D
   02 00000009'EF       000E                  PSECT$BLANK+SYM$W_FLAG,10$ ;IT'LL BE REMOVED SO DECR THE PSECT
               6E   D7  0014          DECL    (SP)                    ;NUMBER TO PRESERVE ALIGNMENT.
                        0016  10$:    $OBJ_OUTBYT_O (SP)              ;EMIT PSECT NUMBER
               8E   D5  0019          TSTL    (SP)+                   ;AND  CLEAN UP THE STACK.
     55   0000'CF   9E  001B          MOVAB   W^MAC$GL_PC,R5          ;POINT TO PC
                        0020          $VPUSH  (R5)                    ;PUSH ONTO MY 'LINKER STACK'
                        0028
                        0028  ;++
                        0028  ; FUNCTIONAL DESCRIPTION:
                        0028  ;
                        0028  ;       THIS ROUTINE EMITS 4 BYTES TO THE OBJECT FILE.
                        0028  ;
                        0028  ; INPUTS:
                        0028  ;
                        0028  ;       R5      POINTS TO 4 BYTES TO OUTPUT
                        0028  ;
                        0028  ;--
                        0028
                        0028  MAC$OUT_LW::
          54   04   9A  0028          MOVZBL  #4,R4                   ;SET UP COUNTER
                        002B  10$:    $OBJ_OUTBYT (R5)+               ;SEND A BYTE
            F7 54   F5  0031          SOBGTR  R4,10$                  ;LOOP FOR FOUR
                    05  0034          RSB
```

```
                         0035  ;++
                         0035  ; FUNCTIONAL DESCRIPTION:
                         0035  ;
                         0035  ;         THIS ROUTINE AUGMENTS THE PC.  THE PARAMETER PASSED IS THE
                         0035  ;         VALUE TO ADD TO THE CURRENT PC.
                         0035  ;
                         0035  ; INPUTS:
                         0035  ;
                         0035  ;         R8      POINTS TO LONGWORD VALUE TO ADD TO PC
                         0035  ;
                         0035  ;--
                         0035
                         0035  P2$AUGPC::
              FFC8'  30  0035          BSBW    MAC$SET_PC              ;RECORD PC
       0000'CF   68  C0  0038          ADDL2   (R8),W^MAC$GL_PC        ;ADJUST PC
              FFC0'  30  003D          BSBW    MAC$SET_PC              ;RECORD NEW PC
    50    0000'CF    D0  0040          MOVL    W^MAC$GL_PSECTPTR,R0    ;GET POINTER TO CURRENT PSECT
   0C 0D A0      03  E1  0045          BBC     #PSC$V_REL,PSC$W_OPTIONS(R0),10$ ;BRANCH IF NOT RELOCATABLE
                         004A          $OBJ_CHKBYT #TIR$C_CTL_AUGRB    ;YES--AUGMENT RELOCATION BASE
             55   58 D0  0051          MOVL    R8,R5                   ;POINT AT VALUE TO OUTPUT
                  D2 11  0054          BRB     MAC$OUT_LW              ;BRANCH TO OUTPUT LONGWORD PC AUGMENTATION
                     05  0056  10$:    RSB
```

```
                     0057        .SBTTL  STORE DATA FROM LINKER STACK
                     0057
                     0057 ;++
                     0057 ; FUNCTIONAL DESCRIPTION:
                     0057 ;
                     0057 ;      THESE ROUTINES, P2$STOB,P2$STOW,P2$SPID,P2$STOL,P2$STOV,
                     0057 ;      P2$STSB, AND P2$STSW
                     0057 ;      ALL STORE SIGNED DATA FROM THE LINKER STACK INTO THE OBJECT
                     0057 ;      FILE.
                     0057 ;
                     0057 ;--
                     0057
                     0057 P2$STSB::                                        ;STORE SIGNED BYTE
                     0057        $OBJ_CHKBYT #TIR$C_STO_SB                ;STORE SIGNED BYTE COMMAND
            06   11  005D        BRB     BYT_COM                          ;GO DO COMMON BYTE PROCESSING
                     005F
                     005F P2$STOB::                                        ;STORE BYTE
                     005F        $OBJ_CHKBYT #TIR$C_STO_B                 ;STORE BYTE COMMAND
                     0065 BYT_COM:
            01   DD  0065        PUSHL   #1                               ;STACK COUNT OF BYTES
            38   10  0067        BSBB    STO_COM                          ;GO TO COMMON CODE
       00000002'     0069        .ADDRESS MAC$AB_LST_END+2                ;NEW LINE IF PAST HERE
                     006D
                     006D P2$STSW::                                        ;STORE SIGNED WORD
                     006D        $OBJ_CHKBYT #TIR$C_STO_SW                ;STORE SIGNED WORD COMMAND
            06   11  0073        BRB     WORD_COM                         ;DO COMMON WORD PROCESSING
                     0075
                     0075 P2$STOW::                                        ;STORE WORD
                     0075        $OBJ_CHKBYT #TIR$C_STO_W                 ;STORE WORD COMMAND
                     007B WORD_COM:
            02   DD  007B        PUSHL   #2                               ;STACK COUNT OF BYTES
            22   10  007D        BSBB    STO_COM                          ;GO TO COMMON CODE
       00000004'     007F        .ADDRESS MAC$AB_LST_END+4                ;NEW LINE IF PAST HERE
                     0083
                     0083 P2$SPID::                                        ;STORE PIC ADDRESS
                     0083        $OBJ_CHKBYT #TIR$C_STO_PIDR              ;STORE PIC DATA REF
            0E   11  0089        BRB     P2_LONG
                     008B
                     008B P2$STOL::                                        ;STORE LONGWORD
                     008B        $OBJ_CHKBYT #TIR$C_STO_LW                ;STORE LONGWORD COMMAND
            06   11  0091        BRB     P2_LONG
                     0093
                     0093 P2$STOV::                                        ;STORE FIELD
                     0093        $OBJ_CHKBYT #TIR$C_STO_VPS               ;STORE FIELD COMMAND
                     0099 P2_LONG:
            04   DD  0099        PUSHL   #4                               ;STACK COUNT OF BYTES
            04   10  009B        BSBB    STO_COM                          ;GO TO COMMON CODE
       00000008'     009D        .ADDRESS MAC$AB_LST_END+8                ;NEW LINE IF PAST HERE
                     00A1
                     00A1 STO_COM:
     50   9E   D0    00A1        MOVL    @(SP)+,R0                        ; Get line limit (and clear stack)
  07 6B   27   E1    00A4        BBC     #FLG$V_UPDFIL,(R11),5$           ; Branch if file is not being updated
50  00000000'8F  C0  00A8        ADDL2   #<MAC$AB_LST_OP2-MAC$AB_LST_END>,R0 ; Reserve space for audit
                     00AF 5$:
50    0000'CF  D1    00AF        CMPL    W^MAC$GL_LIST_PTR,R0             ;TIME FOR NEW LINE?
          03   1E    00B4        BGEQU   10$                              ;IF GEQ NO
        037F   30    00B6        BSBW    P2$FNEWL                         ;YES--FORCE NEW LINE
```

```
                 50 8ED0  00B9   234 10$:    POPL    R0                          ;GET COUNT OF BYTES BACK
        0000'DF  27   90  00BC   235         MOVB    #^A/'/,@W^MAC$GL_LIST_PTR   ;SET QUOTE IN LISTING
                          00C1   236         $MAC_LIST_BYTE R0                   ;LIST THE BYTE(S)
                      05  00C4   237         RSB
```

```
                              00C5          .SBTTL  STORE REPEATED DATA
                              00C5
                              00C5  ;++
                              00C5  ; FUNCTIONAL DESCRIPTION:
                              00C5  ;
                              00C5  ;       THESE ROUTINES, P2$STRB, P2$STRW, P2$STRL, P2$STRSB, AND
                              00C5  ;       P2$STRSW STORE REPEATED DATA FROM THE LINKER STACK INTO THE
                              00C5  ;       OBJECT FILE.
                              00C5  ;
                              00C5  ;--
                              00C5
                              00C5  P2$STRSB::                              ;STORE REPEATED SIGNED BYTE
                              00C5          $OBJ_CHKBYT #TIR$C_STO_RSB      ;STORE REPEATED SIGNED BYTE COMMMAND
                   06    11   00CB          BRB     STRB_COM                ;JOIN COMMON CODE
                              00CD
                              00CD  P2$STRB::                               ;STORE REPEATED BYTE
                              00CD          $OBJ_CHKBYT #TIR$C_STO_RB       ;STORE REPEATED BYTE COMMAND
                              00D3  STRB_COM:
             0002'CF     9F   00D3          PUSHAB  W^MAC$AB_LST_END+2      ;STACK END OF LINE MARKER
                7E 01    90   00D7          MOVB    #1,-(SP)                ;AND COUNT OF BYTES TO LIST PER VALUE
                   24    11   00DA          BRB     STO_LOOP
                              00DC
                              00DC  P2$STRSW::                              ;STORE REPEATED SIGNED WORD
                              00DC          $OBJ_CHKBYT #TIR$C_STO_RSW      ;STORE REPEATED SIGNED WORD COMMAND
                   06    11   00E2          BRB     STRW_COM                ;JOIN COMMON CODE
                              00E4
                              00E4  P2$STRW::                               ;STORE REPEATED WORD
                              00E4          $OBJ_CHKBYT #TIR$C_STO_RW       ;STORE REPEATED WORD COMMAND
                              00EA  STRW_COM:
             0004'CF     9F   00EA          PUSHAB  W^MAC$AB_LST_END+4      ;STACK END OF LINE MARKER
                7E 02    90   00EE          MOVB    #2,-(SP)                ;STACK COUNT OF BYTES TO LIST PER VALUE
                   0D    11   00F1          BRB     STO_LOOP
                              00F3
                              00F3  P2$STRL::                               ;STORE REPEATED LONGWORD
                              00F3          $OBJ_CHKBYT #TIR$C_STO_RL       ;STORE REPEATED LONGWORD COMMAND
             0008'CF     9F   00F9          PUSHAB  W^MAC$AB_LST_END+8      ;STACK END OF LINE MARKER
                7E 04    90   00FD          MOVB    #4,-(SP)                ;STACK COUNT OF BYTES TO LIST PER VALUE
                              0100
                              0100  STO_LOOP:
                              0100
                              0100          $VPOP   R6                      ;GET REPEAT COUNT
                   56    D5   0108          TSTL    R6                      ;CHECK REPEAT COUNT
                   32    13   010A          BEQL    30$                     ;IF EQL ALL DONE
                              010C          $VPOP   -(SP)                   ;GET VALUE OFF VALUE STACK
       05 AE   0000'CF   D1   0114  10$:    CMPL    W^MAC$GL_LIST_PTR,5(SP) ;TIME FOR NEW LINE?
                   03    1E   011A          BGEQU   20$                     ;IF GEQU NO
                 0319    30   011C          BSBW    P2$FNEWL                ;YES--FORCE NEW LINE
        0000'DF    27    90   011F  20$:    MOVB    #^A/'/,@W^MAC$GL_LIST_PTR ;STORE QUOTE IN LISTING
                              0124          $VPUSH  (SP)                    ;STACK THE VALUE ON VALUE STACK
       55    0000'CF47   DE   012C          MOVAL   W^MAC$AL_VALSTACK[R7],R5;POINT R5 TO TOP OF VALUE STACK
                              0132          $MAC_LIST_BYTE 4(SP)            ;LIST THE VALUE
                D8 56    F5   0139          SOBGTR  R6,10$                  ;LOOP FOR ALL BYTES
                   8E    D5   013C          TSTL    (SP)+                   ;CLEAR SAVE VALUE FROM STACK
                5E 05    C0   013E  30$:    ADDL2   #5,SP                   ;CLEAR BYTE AND LONGWORD FROM STACK
                   05         0141          RSB
```

```
                          0142          .SBTTL  STORE IMMEDIATE DATA
                          0142
                          0142     ;++
                          0142     ; FUNCTIONAL DESCRIPTION:
                          0142     ;
                          0142     ;        THIS ROUTINE STORES A BYTE OF DATA INTO THE OBJECT CODE
                          0142     ;
                          0142     ;--
                          0142
                          0142     P2$STIB::
      50   0002'CF    9E  0142             MOVAB   W^MAC$AB_LST_END+2,R0   ;SET ADDRESS TO CHECK
                5D    10  0147             BSBB    CK_NWL                  ;SEE IF TIME FOR NEW LINE AND FORCE IF NEEDE
      50        68    9A  0149             MOVZBL  (R8),R0                 ;GET THE DATA BYTE
                          014C             $VPUSH  R0                      ;STACK IT
                85    D5  0154             TSTL    (R5)+                   ;KEEP R5 POINTING TO TOP OF STACK
             FEA7'    30  0156             BSBW    MAC$CK_BYT_TRUN         ;CHECK FOR TRUNCATION
      50        68    90  0159             MOVB    (R8),R0                 ;GET THE BYTE VALUE
             FEA1'    30  015C             BSBW    MAC$STOIM               ;STORE INTO OBJECT CODE
                          015F             $MAC_LIST_BYTE #1               ;LIST BYTE FROM STACK
                      05  0165             RSB                             ;EXIT
                          0166
                          0166     ;++
                          0166     ; FUNCTIONAL DESCRIPTION:
                          0166     ;
                          0166     ;        THIS ROUTINE STORES A WORD INTO THE OBJECT CODE
                          0166     ;
                          0166     ;--
                          0166
                          0166     P2$STIW::
      50   0004'CF    9E  0166             MOVAB   W^MAC$AB_LST_END+4,R0   ;GET ADDRESS TO CHECK
                39    10  016B             BSBB    CK_NWL                  ;SEE IF TIME FOR NEW LINE
                          016D             $VPUSH  (R8)                    ;STACK VALUE
                85    D5  0175             TSTL    (R5)+                   ;KEEP R5 POINTING TO TOP OF STACK
             FE86'    30  0177             BSBW    MAC$CK_WRD_TRUN         ;CHECK FOR TRUNCATION
      53        02    9A  017A             MOVZBL  #2,R3                   ;SET TO STORE TWO BYTES
                14    11  017D             BRB     STO_DAT                 ;STORE WORD AND RETURN
                          017F
                          017F     ;++
                          017F     ; FUNCTIONAL DESCRIPTION:
                          017F     ;
                          017F     ;        THIS ROUTINE STORES A LONGWORD INTO THE OBJECT CODE
                          017F     ;
                          017F     ;--
                          017F
                          017F     P2$STIL::
      50   0008'CF    9E  017F             MOVAB   W^MAC$AB_LST_END+8,R0   ;GET ADDRESS TO CHECK
                20    10  0184             BSBB    CK_NWL                  ;SEE IF TIME FOR A NEW LINE
                          0186             $VPUSH  (R8)                    ;STACK VALUE
                85    D5  018E             TSTL    (R5)+                   ;KEEP R5 POINTING TO TOP OF STACK
      53        04    9A  0190             MOVZBL  #4,R3                   ;SET TO STORE 4 BYTES
                          0193     STO_DAT:
      54        53    D0  0193             MOVL    R3,R4                   ;REMEMBER THE BYTE COUNT
      50        88    90  0196     10$:    MOVB    (R8)+,R0                ;GET A BYTE
             FE64'    30  0199             BSBW    MAC$STOIM               ;STORE INTO OBJECT CODE
             F7 54    F5  019C             SOBGTR  R4,10$                  ;LOOP FOR ALL BYTES
                          019F             $MAC_LIST_BYTE R3               ;LIST BYTES FROM STACK
                      05  01A5             RSB
```

```
                      01A6      
                      01A6      ;++
                      01A6      ; FUNCTIONAL DESCRIPTION:
                      01A6      ;
                      01A6      ;         THIS ROUTINE IS CALLED BY THE STORE IMMEDIATE ROUTINES TO
                      01A6      ;         SEE IF IT IS TIME FOR A NEW LISTING LINE.
                      01A6      ;
                      01A6      ; INPUTS:
                      01A6      ;
                      01A6      ;         R0      BOTTOM ADDRESS WE CAN GO TO
                      01A6      ;
                      01A6      ; OUTPUTS:
                      01A6      ;
                      01A6      ;         NEW LINE IF NOT GEQU R0.
                      01A6      ;
                      01A6      ;--
                      01A6      
                      01A6      CK_NWL:
  50  0000'CF    D1   01A6              CMPL    W^MAC$GL_LIST_PTR,R0    ;TIME FOR A NEW LINE?
           03    1E   01AB              BGEQU   10$                     ;IF GEQU NO
         0288    31   01AD              BRW     P2$FNEWL                ;YES--FORCE NEW LINE AND RETURN
                 05   01B0      10$:    RSB                             ;JUST RETURN
```

```
                   01B1        .SBTTL  PSECT ACTION ROUTINES
                   01B1
                   01B1  ;++
                   01B1  ; FUNCTIONAL DESCRIPTION:
                   01B1  ;
                   01B1  ;       P2$NEWP IS CALLED TO DEFINE A NEW PSECT DURING PASS 2.
                   01B1  ;       THE PREVIOUS OBJECT RECORD IS WRITTEIN, AND A GSD RECORD
                   01B1  ;       IS OUTPUT CONTAINING THE PSECT DEFINITION.  THE RECORD
                   01B1  ;       TYPE IS THEN RESET TO TIR.
                   01B1  ;
                   01B1  ;--
                   01B1
                   01B1  P2$NEWP::
       56   68  D0 01B1        MOVL    (R8),R6                      ;POINT TO PSECT DEFINITION BLOCK
   01  0C   A6  91 01B4        CMPB    PSC$B_SEG(R6), #1            ;IS THIS THE BLANK PSECT?
            05  12 01B8        BNEQ    5$                           ;BRANCH IF NOT.
46 09  A6   07  E1 01BA        BBC     #SYM$V_REF,PSC$W_FLAG(R6),20$ ;IF IT'S NOT REFERENCED REMOVE IT.
   00  6B   2E  E2 01BF  5$:   BBSS    #FLG$V_DBGOUT,(R11),.+1      ;SET DEBUGGER RECORD FLAG
 0000'CF    01  9A 01C3        MOVZBL  #OBJ$C_GSD,W^MAC$GL_RECTYP   ;SET GSD RECORD TYPE
      FE35'     30 01C8        BSBW    MAC$WRTOBJ                   ;WRITE PREVIOUS RECORD
                   01CB        $OBJ_OUTBYT_0 #OBJ$C_GSD_PSC         ;PSECT DEFINITION RECORD
            0A  EF 01CE        EXTZV   #PSC$V_ALIGNMENT,-           ;EXTRACT ALIGNMENT FIELD
            04     01D0                #PSC$S_ALIGNMENT,-
   50  0D   A6     01D1                PSC$W_OPTIONS(R6),R0
                   01D4        $OBJ_OUTBYT_0 R0                     ;OUTPUT ALIGNMENT
                   01D7        $OBJ_OUTBYT_0 PSC$W_OPTIONS(R6)      ;OUTPUT LOW BYTE OF FLAGS
       FC   8F  8B 01DB        BICB3   #^C<PSC$M_ALLOPTNS@-8>,-     ; Get remaining option bits
   50  0E   A6     01DE                PSC$W_OPTIONS+1(R6),R0
                   01E1        $OBJ_OUTBYT_0 R0                     ;EMIT HI BYTE OF FLAGS
03 0D  A6   03  E0 01E4        BBS     #PSC$V_REL,PSC$W_OPTIONS(R6),10$ ;BRANCH IF RELOCATABLE PSECT
       05   A6  D4 01E9        CLRL    PSC$L_MAXLGTH(R6)            ;ABSOLUTE--CLEAR MAX LENGTH
   55  05   A6  9E 01EC  10$:  MOVAB   PSC$L_MAXLGTH(R6),R5         ;POINT TO MAX LENGTH
       FE35     30 01F0        BSBW    MAC$OUT_LW                   ;OUTPUT THE MAX LENGTH TO OBJECT CODE
       FE0A'    30 01F3        BSBW    MAC$SYMNAMOUT                ;OUTPUT PSECT NAME
 0000'CF    02  9A 01F6        MOVZBL  #OBJ$C_TIR,W^MAC$GL_RECTYP   ;RESET RECORD TYPE TO TIR
       FE02'    30 01FB        BSBW    MAC$WRTOBJ                   ;WRITE OUT PSECT DEFINITION RECORD
       05   A6  D4 01FE        CLRL    PSC$L_MAXLGTH(R6)            ;START AT RELATIVE ZERO
   00  6B   2E  E5 0201        BBCC    #FLG$V_DBGOUT,(R11),.+1      ;CLEAR DEBUGGER RECORD FLAG
                05 0205  20$:  RSB
```

```
                        0206          .SBTTL  CHANGE PSECT / RESTORE PSECT FROM STACK
                        0206
                        0206  ;++
                        0206  ; FUNCTIONAL DESCRIPTION:
                        0206  ;
                        0206  ;       THIS ROUTINE IS CALLED TO CHANGE PSECTS OR TO RESTORE A PSECT
                        0206  ;       FROM THE PSECT STACK.  CODE IS EMITTED TO THE OBJECT FILE TO
                        0206  ;       INFORM THE LINKER OF THE NEW PSECT
                        0206  ;
                        0206  ;--
                        0206
                        0206  P2$PSECT::
            FDF7'  30   0206          BSBW    MAC$SET_PC                   ;RECORD PC
         56   68   D0   0209          MOVL    (R8),R6                      ;PICK UP POINTER TO PSECT BLOCK
     53   0000'CF  9E   020C          MOVAB   W^MAC$GL_PC,R3               ;POINT TO PC WORD
       63   05 A6  D0   0211          MOVL    PSC$L_MAXLGTH(R6),(R3)       ;SET NEW PC
              0F   11   0215          BRB     PSECT_REST                   ;JOIN COMMON CODE
                        0217
                        0217  P2$REST::
                        0217
            FDE6'  30   0217          BSBW    MAC$SET_PC                   ;RECORD PC
         56   68   D0   021A          MOVL    (R8),R6                      ;PICK UP POINTER TO PSECT BLOCK
     53   0000'CF  9E   021D          MOVAB   W^MAC$GL_PC,R3               ;POINT TO PC WORD
       63   0F A6  D0   0222          MOVL    PSC$L_CURLOC(R6),(R3)        ;SET NEW PC
                        0226  PSECT_REST:
 0000'CF    0C A6  9A   0226          MOVZBL  PSC$B_SEG(R6),W^MAC$GL_PSECT ;SET NEW PSECT NUMBER
   0000'CF    56   D0   022C          MOVL    R6,W^MAC$GL_PSECTPTR         ;SET POINTER TO NEW PSECT
   03 0D A6   03   E0   0231          BBS     #PSC$V_REL, -                ;IF REL PSECT THEN CONTINUE
                        0236                  PSC$W_OPTIONS(R6),3$
            0090   31   0236          BRW     40$                          ;ELSE ABS PSECT SO SKIP
                        0239  3$:     $VPUSH  (R3)                         ;PUSH NEW PC ONTO VALUE STACK
      00 6B   0C   E3   0241          BBCS    #FLG$V_MEBLST,(R11),5$       ;ENSURE DIRECTIVE GETS LISTED
         55   04   9A   0245  5$:     MOVZBL  #4,R5                        ;ASSUME LONGWORD OFFSET
         54   06   9A   0248          MOVZBL  #TIR$C_STA_PL,R4             ;...
         02 A3     B5   024B          TSTW    2(R3)                        ;HIGH WORD OF PC ZERO?
              19   12   024E          BNEQ    10$                          ;IF NEQ NO
              63   B5   0250          TSTW    (R3)                         ;YES--LOW WORD POSITIVE?
              15   19   0252          BLSS    10$                          ;IF LSS NO
         55   02   9A   0254          MOVZBL  #2,R5                        ;YES--OPTIMIZE TO WORD
         54   05   9A   0257          MOVZBL  #TIR$C_STA_PW,R4             ;...
         01 A3     95   025A          TSTB    1(R3)                        ;HIGH BYTE OF LOW WORD ZERO?
              0A   12   025D          BNEQ    10$                          ;IF NEQ NO
              63   95   025F          TSTB    (R3)                         ;YES--LOW BYTE POSITIVE?
              06   19   0261          BLSS    10$                          ;IF LSS NO
         55   01   9A   0263          MOVZBL  #1,R5                        ;YES--OPTIMIZE TO BYTE
         54   04   9A   0266          MOVZBL  #TIR$C_STA_PB,R4             ;...
                        0269  10$:    $OBJ_CHKBYT R4                       ;OUTPUT COMMAND
       7E   0C A6  9A   026F          MOVZBL  PSC$B_SEG(R6),-(SP)          ;PREPARE TO EMIT THE PSECT NUMBER.
              0A   13   0273          BEQL    15$                          ;DON'T FIDDLE WITH THE ABS PSECT!
              07   E0   0275          BBS     #SYM$V_REF,-                 ;IF THE BLANK PSECT HASN'T BEEN REF'D
 02 00000009'EF         0277                  PSECT$BLANK+SYM$W_FLAG,15$ ;IT'LL BE REMOVED, SO DECR THE PSECT
              6E   D7   027D          DECL    (SP)                         ;NUMBER TO PRESERVE ALIGNMENT.
                        027F  15$:    $OBJ_OUTBYT_0 (SP)                   ;OUTPUT PSECT NUMBER
              8E   D5   0282          TSTL    (SP)+                        ;AND CLEAN UP THE STACK.
                        0284  20$:    $OBJ_OUTBYT_0 (R3)+                  ;OUTPUT PC
         FA 55     F5   0287          SOBGTR  R5,20$                       ;LOOP FOR ALL BYTES
                        028A          $OBJ_CHKBYT #TIR$C_CTL_SETRB         ;SET RELOCATION BASE COMMAND
```

```
           54    04  9A  0291           MOVZBL  #4,R4                   ;ASSUME FULL PC VALUE
0000'8F  0000'CF     B1  0294           CMPW    W^MAC$GL_PSECTPTR,#PSECT$BLANK ;BLANK PSECT?
                 09  12  029B           BNEQ    30$                     ;IF NEQ NO
         0000'CF     D5  029D           TSTL    W^MAC$GL_PC             ;YES--AND IS PC ZERO?
                 03  12  02A1           BNEQ    30$                     ;IF NEQ NO
           54    02  9A  02A3           MOVZBL  #2,R4                   ;YES--LIST WORD ONLY
    55   0000'CF47   DE  02A6  30$:     MOVAL   W^MAC$AL_VALSTACK[R7],R5 ;POINT R5 TO TOP OF VALUE STACK
0000'CF  0000'CF     9E  02AC           MOVAB   W^MAC$AB_SEQ_NUM,W^MAC$GL_LIST_PTR ;POINT TO WHERE PC GOES
                         02B3           $MAC_LIST_BYTE R4               ;LIST PC
                         02B9           $DEC_PC R4                      ;WE REALLY DON'T WANT TO INC PC FOR THAT
      02   54        91  02BE           CMPB    R4,#2                   ;SHORT FORM PC?
                 06  12  02C1           BNEQ    40$                     ;IF NEQ NO
      50   20        9A  02C3           MOVZBL  #^A/ /,R0               ;YES--LIST A SPACE
            FD37'    30  02C6           BSBW    MAC$LST_CHAR            ;...
                     05  02C9  40$:     RSB
                         02CA
                         02CA  ;++
                         02CA  ; FUNCTIONAL DESCRIPTION:
                         02CA  ;
                         02CA  ;        THIS ROUTINE IS CALLED TO PROCESS A .SAVE DIRECTIVE IN PASS 2.
                         02CA  ;        THE CURRENT PC IS RECORDED IN THE PSECT BLOCK.
                         02CA  ;
                         02CA  ;--
                         02CA
                         02CA  P2$SAVE::
           50    68  D0  02CA           MOVL    (R8),R0                 ;POINT TO PSECT BLOCK
   0F A0  0000'CF    D0  02CD           MOVL    W^MAC$GL_PC,PSC$L_CURLOC(R0) ;SAVE PC IN PSECT BLOCK
     00 6B       0C  E2  02D3           BBSS    #FLG$V_MEBLST,(R11),10$ ;ENSURE DIRECTIVE GETS LISTED
                     05  02D7  10$:     RSB
                         02D8
```

```
                          02D8            .SBTTL  ENTRY POINT DEFINITIONS
                          02D8
                          02D8     ;++
                          02D8     ; FUNCTIONAL DESCRIPTION:
                          02D8     ;
                          02D8     ;       THIS ROUTINE PROCESSES AN ENTRY POINT DEFINITION IN PASS 2.
                          02D8     ;       A GSD RECORD IS OUTPUT TO THE OBJECT FILE CONTAINING THE
                          02D8     ;       EPT DEFINITION.
                          02D8     ;
                          02D8     ;--
                          02D8
                          02D8     P2$EPT::
      0000'CF    01   9A  02D8            MOVZBL  #OBJ$C_GSD,W^MAC$GL_RECTYP ;NEXT RECORD IS GSD
             FD20'   30   02DD            BSBW    MAC$WRTOBJ              ;WRITE OUT PREVIOUS RECORD
                          02E0            $OBJ_OUTBYT_O #OBJ$C_GSD_EPM    ;ENTRY POINT MASK
                          02E3            $OBJ_OUTBYT_O #0                ;DATA TYPE
           56    88   D0  02E6            MOVL    (R8)+,R6                ;GET ID POINTER
           50    02   9A  02E9            MOVZBL  #SYM$F_DEF,R0           ;SET DEFINED FLAG
     03 09 A6    04   E0  02EC            BBS     #SYM$V_ABS,SYM$W_FLAG(R6),10$ ;BRANCH IF ABSOLUTE
           50    08   88  02F1            BISB2   #SYM$F_REL,R0           ;NO--SET RELOCATABLE
     03 09 A6    01   E1  02F4     10$:   BBC     #SYM$V_WEAK,SYM$W_FLAG(R6),20$ ;BRANCH IF NOT WEAK
           50    01   88  02F9            BISB2   #SYM$F_WEAK,R0          ;YES--MARK IT
                          02FC     20$:   $OBJ_OUTBYT_O R0                ;EMIT FLAGS (LOW BYTE)
                          02FF            $OBJ_OUTBYT_O #0                ;EMIT FLAGS (HI BYTE)
      7E      0C A6   9A  0302            MOVZBL  SYM$B_SEG(R6), -(SP)    ;PREPARE TO EMIT THE PSECT NUMBER.
                 0A   13  0306            BEQL    30$                     ;DON'T FIDDLE WITH THE ABS PSECT!
                 07   E0  0308            BBS     #SYM$V_REF,-            ;IF THE BLANK PSECT HASN'T BEEN REF'D
      02 00000009'EF      030A                    PSECT$BLANK+SYM$W_FLAG,30$ ;IT'LL BE REMOVED, SO DECR THE PSECT
                 6E   D7  0310            DECL    (SP)                    ;NUMBER TO PRESERVE THE ALIGNMENT.
                          0312     30$:   $OBJ_OUTBYT_O (SP)              ;OUTPUT SEGMENT #
                 8E   D5  0315            TSTL    (SP)+                   ;AND CLEAN UP THE STACK.
      55      05 A6   9E  0317            MOVAB   SYM$L_VAL(R6),R5        ;POINT TO VALUE
               FD0A   30  031B            BSBW    MAC$OUT_LW              ;EMIT LONGWORD VALUE
                          031E            $OBJ_OUTBYT_O (R8)+             ;STORE ENTRY MASK
                          0321            $OBJ_OUTBYT_O (R8)              ;...
              FCD9'   30  0324            BSBW    MAC$SYMNAMOUT           ;EMIT SYMBOL NAME
      0000'CF    02   9A  0327            MOVZBL  #OBJ$C_TIR,W^MAC$GL_RECTYP ;NEXT RECORD IS TIR
           50    02   9A  032C            MOVZBL  #2,R0                   ; Set to list 2 bytes
              FCCE'   30  032F            BSBW    MAC$LIST_BYT_O          ; List two bytes from value stack
                          0332            $OBJ_OUTBYT #TIR$C_STO_W        ; Store word from linker stack
              FCC5'   31  0338            BRW     W^MAC$WRTOBJ            ;WRITE OUT GSD AND RETURN
                          033B
                          033B     ;++
                          033B     ; FUNCTIONAL DESCRIPTION:
                          033B     ;
                          033B     ;       THIS ROUTINE EMITS CODE TO THE OBJECT FILE FOR THE LINKER
                          033B     ;       TO STACK THE ENTRY-POINT MASK FOR A SYMBOL.
                          033B     ;
                          033B     ;--
                          033B
                          033B     P2$STKEPT::
                          033B            $OBJ_CHKBYT #TIR$C_STA_EPM      ;STACK ENTRY POINT MASK
           56    88   D0  0341            MOVL    (R8)+,R6                ;GET ID POINTER
              FCB9'   30  0344            BSBW    MAC$SYMNAMOUT           ;SEND SYMBOL NAME
                          0347            $VPUSH  #0                      ;STACK 0 (WE DON'T KNOW VALUE)
                      05  034F            RSB
```

```
                        0350         .SBTTL  PROCESS ASSIGNMENT STATEMENT
                        0350
                        0350 ;++
                        0350 ; FUNCTIONAL DESCRIPTION:
                        0350 ;
                        0350 ;       THIS ROUTINE PROCESS ASSIGNMENT STATEMENTS DURING PASS 2.
                        0350 ;       THE SYMBOL BLOCK IS UPDATED FROM THE INTERMEDIATE FILE.
                        0350 ;
                        0350 ;--
                        0350
                        0350 P2$ASN::
       50     88    D0  0350         MOVL    (R8)+,R0                ;GET POINTER TO SYMBOL BLOCK
    0C A0     88    90  0353         MOVB    (R8)+,SYM$B_SEG(R0)     ;UPDATE SEGMENT
    05 A0     88    D0  0357         MOVL    (R8)+,SYM$L_VAL(R0)     ;UPDATE VALUE
              68    95  035B         TSTB    (R8)                    ;IS SYMBOL RELOCATABLE?
              06    12  035D         BNEQ    10$                     ;IIF NEQ YES
    09 A0     10    A8  035F         BISW2   #SYM$M_ABS,SYM$W_FLAG(R0) ;NO--MAKE ABSOLUTE
              04    11  0363         BRB     20$
    09 A0     10    AA  0365 10$:    BICW2   #SYM$M_ABS,SYM$W_FLAG(R0) ;YES--MAKE RELOCATABLE
                    05  0369 20$:    RSB
                        036A
                        036A ;++
                        036A ; FUNCTIONAL DESCRIPTION:
                        036A ;
                        036A ;       THIS ROUTINE PROCESSES THE NEWLINE COMMAND.  THE LINE NUMBER
                        036A ;       PASSED IS MERELY STORED IN 'MAC$GL_P2_LINE'
                        036A ;
                        036A ;--
                        036A
                        036A P2$NEWL::
  0000'CF     68    3C  036A         MOVZWL  (R8),W^MAC$GL_P2_LINE   ;RECORD THE LINE NUMBER
                    05  036F         RSB
```

```
                          0370          .SBTTL  ALIGN SOURCE AND LISTING
                          0370
                          0370      ;++
                          0370      ; FUNCTIONAL DESCRIPTION:
                          0370      ;
                          0370      ;       THIS ROUTINE  ALIGNS THE LISTING FILE WITH THE SOURCE FILE.
                          0370      ;       SOURCE LINES ARE READ AND WRITTEN UNTIL MAC$GL_LINENUM IS
                          0370      ;       CAUGHT UP WITH MAC$GL_P2_LINE.
                          0370      ;
                          0370      ;--
                          0370
                          0370      P2$CHKL::
        03 6B    0B   E5  0370              BBCC    #FLG$V_MACL,(R11),10$   ;MACRO LINE PRESENT (AND CLEAR FLAG)?
              00A5    30  0374              BSBW    DUMP_MACRO              ;YES--COPY INTO LINEBF
           0000'CF    DD  0377      10$:    PUSHL   W^MAC$GL_LIST_IT        ;SAVE LISTING FLAG
 0000'CF   0000'CF    D1  037B      20$:    CMPL    W^MAC$GL_P2_LINE,W^MAC$GL_LINENUM ;ARE WE CAUGHT UP?
                1C    15  0382              BLEQ    40$                     ;IF LEQ YES
      0000'CF    6E   D0  0384              MOVL    (SP),W^MAC$GL_LIST_IT   ;NO--SET LISTING FLAG
             FC74'    30  0389              BSBW    MAC$WRTLST              ;WRITE CURRENT LINE
             FC71'    30  038C              BSBW    MAC$GETLIN              ;GET NEXT LINE
        00 6B    1B   E5  038F              BBCC    #FLG$V_MACLTB,(R11),25$ ;NO TEXT IN MACRO BUFFER
           0000'CF    DD  0393      25$:    PUSHL   W^MAC$GL_LIST_PTR       ;SAVE LINE POINTER
                0A    10  0397              BSB     MAC$FORM_LINENO         ; Form line no, audit trail
        0000'CF 8ED0      0399              POPL    W^MAC$GL_LIST_PTR       ;RESTORE POINTER
                DB    11  039E              BRB     20$                     ;LOOP TILL WE CATCH UP
                8E    D5  03A0      40$:    TSTL    (SP)+                   ;KEEP STACK CLEAN
                      05  03A2              RSB
```

```
                               03A3          .SBTTL  FORM LINE NUMBER AND AUDIT TRAIL
                               03A3
                               03A3    ;++
                               03A3    ; Functional description:
                               03A3    ;
                               03A3    ;       This routine forms the line number and audit trail of the
                               03A3    ;       current line
                               03A3    ;
                               03A3    ; --
                               03A3    ;
                               03A3    MAC$FORM_LINENO::
    0000'CF   FFFF'CF     9E   03A3          MOVAB   W^MAC$AB_LST_LIN-1, -    ; Point to where line number goes
                               03AA                  W^MAC$GL_LIST_PTR
         50   0000'CF     3C   03AA          MOVZWL  W^MAC$GW_LST_LINE,R0     ; Get listing line number
         56   0000'CF     DE   03AF          MOVAL   W^MAC$GT_SCB,R6          ; Set pointer to SUM control block
           05 6B      27  E1   03B4          BBC     #FLG$V_UPDFIL,(R11),10$  ; Branch if file is not being updated
        19 1C A6      02  E0   03B8          BBS     #SUM_V_SRCUPD, -         ; Branch if line is from update file
                               03BD                  SUM_B_FLAGS(R6),20$
                               03BD    10$:
                FC40'     30   03BD          BSBW    MAC$DEC_OUT_R2L          ; Output to listing
        2E 1C A6      04  E5   03C0          BBCC    #SUM_V_DELETE, -         ; Branch if no deleted line information
                               03C5                  SUM_B_FLAGS(R6),30$
         51   0000'CF     9E   03C5          MOVAB   W^MAC$AB_LST_END,R1      ; Set pointer to start of line
              81      2D  90   03CA          MOVB    #^A/-/,(R1)+             ; Insert '-'
         50    1A A6      3C   03CD          MOVZWL  SUM_W_INSERT_NO(R6),R0   ; Get number of lines deleted
                FC2C'     30   03D1          BSBW    MAC$DEC_OUT_L2X          ; Output to listing
                   1D     11   03D4          BRB     30$
                               03D6    20$:
         50    1A A6      3C   03D6          MOVZWL  SUM_W_INSERT_NO(R6),R0   ; Get insert number
                FC23'     30   03DA          BSBW    MAC$DEC_OUT_R2L          ; Output to listing
              0000'CF     D7   03DD          DECL    W^MAC$GL_LIST_PTR        ; Decrement listing pointer
           0000'DF    2E  90   03E1          MOVB    #^A/./,@W^MAC$GL_LIST_PTR ; and insert '.'
        08 1C A6      00  E1   03E6          BBC     #SUM_V_AUDIT,SUM_B_FLAGS(R6),30$ ; Branch if audit trail is off
0000'CF   0C B6   08 A6   28   03EB          MOVC3   SUM_Q_AUDDS(R6), -       ; Copy audit string to listing line
                               03F3                  @SUM_Q_AUDDS+4(R6),W^MAC$AB_LST_END
                               03F3    30$:
                          05   03F3          RSB
```

```
                               03F4          .SBTTL  ERROR TEXT HANDLING
                               03F4
                               03F4     ;++
                               03F4     ; FUNCTIONAL DESCRIPTION:
                               03F4     ;
                               03F4     ;       THIS ROUTINE COPIES THE ERROR/WARN/PRINT TEXT INTO ETXBUF
                               03F4     ;       TO MAKE IT AVAILABLE FOR PASS 2 HANDLING.
                               03F4     ;
                               03F4     ;--
                               03F4
                               03F4     P2$ETX::
              50      88   3C  03F4              MOVZWL  (R8)+,R0                    ;GET THE LENGTH OF THE LINE
         0000'CF      50   D0  03F7              MOVL    R0,W^MAC$GL_ETXLEN          ;SET ITS LENGTH
                      06   15  03FC              BLEQ    10$                         ;IF LEQ NO LINE TO SPEAK OF
 0000'CF      68      50   28  03FE              MOVC3   R0,(R8),W^MAC$AB_ETXBUF     ;COPY INTO ERROR TEXT BUFFER
                           05  0404     10$:     RSB
```

```
                         0405          .SBTTL  STORE MACRO SOURCE TEXT PASSED TO PASS 2
                         0405
                         0405     ;++
                         0405     ; FUNCTIONAL DESCRIPTION:
                         0405     ;
                         0405     ;        THIS ROUTINE COPIES THE MACRO SOURCE LINE FROM THE INT. BUFFER
                         0405     ;        INTO THE MACRO LINE BUFFER, MAC$AB_TMPBUF.
                         0405     ;
                         0405     ;--
                         0405
                         0405     P2$MACL::
       02 6B    0B   E3  0405          BBCS    #FLG$V_MACL,(R11),10$   ;IS THERE A MACRO LINE PRESENT?
                11   10  0409          BSBB    DUMP_MACRO              ;YES--COPY INTO LINE BUFFER
          50    88   3C  040B     10$: MOVZWL  (R8)+,R0                ;GET LENGTH OF MACRO LINE
     0000'CF    50   D0  040E          MOVL    R0,W^MAC$GL_MLIN_LEN    ;SAVE LENGTH OF LINE
                06   15  0413          BLEQ    20$                     ;IF NULL LINE WE ARE DONE
 0000'CF  68    50   28  0415          MOVC3   R0,(R8),W^MAC$AB_TMPBUF ;COPY LINE INTO MACRO LINE BUFFER
                     05  041B     20$: RSB
                         041C
                         041C     ;++
                         041C     ; FUNCTIONAL DESCRIPTION:
                         041C     ;
                         041C     ;        THIS ROUTINE WRITES THE CURRENT LISTING LINE, AND THEN
                         041C     ;        COPIES THE MACRO LINE FROM THE MACRO BUFFER (MAC$AB_TMPBUF)
                         041C     ;        INTO THE LINE BUFFER.
                         041C     ;
                         041C     ;--
                         041C
                         041C     DUMP_MACRO:
             FBE1'   30  041C          BSBW    MAC$WRTLST              ;WRITE CURRENT LINE
   50   0000'CF      D0  041F          MOVL    W^MAC$GL_MLIN_LEN,R0    ;GET LENGTH OF MACRO LINE
     0000'CF    50   D0  0424          MOVL    R0,W^MAC$GL_LINELN      ;SET THE LINE LENGTH
                08   15  0429          BLEQ    10$                     ;IF LEQ DONE
     0000'CF    50   28  042B          MOVC3   R0,W^MAC$AB_TMPBUF,-    ;COPY LINE INTO LINE BUFFER
        0000'CF          0430                  W^MAC$AB_LINEBF
       00 6B    1B   E3  0433     10$: BBCS    #FLG$V_MACLTB,(R11),20$ ;FLAG MACRO TEXT IN BUFFER
                     05  0437     20$: RSB
```

```
                     0438          .SBTTL  FORCE NEW LINE, PRINT LITERAL, SUBTITLE
                     0438
                     0438   ;++
                     0438   ; FUNCTIONAL DESCRIPTION:
                     0438   ;
                     0438   ;       THIS ROUTINE FORCES A NEW LISTING LINE, AND PRESERVES
                     0438   ;       THE PREVIOUS STATE OF THE LISTING FLAG MAC$GL_LIST_IT.
                     0438   ;
                     0438   ;--
                     0438
                     0438   P2$FNEWL::
     0000'CF   DD    0438          PUSHL   W^MAC$GL_LIST_IT             ;SAVE LISTING FLAG
       FBC1'   30    043C          BSBW    MAC$WRTLST                   ;FORCE NEW LINE
     0000'CF 8ED0    043F          POPL    W^MAC$GL_LIST_IT             ;RESTORE LISTING FLAG
               05    0444          RSB
                     0445
                     0445   ;++
                     0445   ; FUNCTIONAL DESCRIPTION:
                     0445   ;
                     0445   ;       THIS ROUTINE PRINTS A LITERAL.  THE PC IS NOT AUGMENTED.
                     0445   ;
                     0445   ;--
                     0445
                     0445   P2$PRIL::
                     0445          $VPUSH  (R8)                         ;STACK THE VALUE TO PRINT
          85   D5    044D          TSTL    (R5)+                        ;KEEP R5 POINTING TO TOP OF STACK
                     044F          $MAC_LIST_BYTE #4
                     0455          $DEC_PC #4                           ;DON'T INCREMENT PC THOUGH
               05    045A          RSB
                     045B
                     045B   ;++
                     045B   ; FUNCTIONAL DESCRIPTION:
                     045B   ;
                     045B   ;       THIS ROUTINE COPIES THE SUBTITLE FROM THE INTERMEDIATE
                     045B   ;       BUFFER INTO THE SUBTITLE BUFFER, SO IT IS AVAILABLE FOR
                     045B   ;       LISTING PAGES.
                     045B   ;
                     045B   ;--
                     045B
                     045B   P2$SBTTL::
     50   88   9A    045B          MOVZBL  (R8)+,R0                     ;GET LENGTH OF SUBTITLE
20   68   50   2C    045E          MOVC5   R0,(R8),#^A/ /,-             ;COPY INTO THE SUBTITLE BUFFER
          28         0462                  #LST$K_TITLE_SIZ,-
     0000'CF         0463                  W^MAC$AB_SBT_SBTL
               05    0466          RSB
```

```
                     0467          .SBTTL  SET LONGWORD VALUE, SET AND CLEAR FLAGS
                     0467
                     0467  ;++
                     0467  ; FUNCTIONAL DESCRIPTION:
                     0467  ;
                     0467  ;       THIS ROUTINE IS CALLED TO  SET THE VALUE CONTAINED IN
                     0467  ;       THE FIRST LONGWORD INTO THE LONGWORD POINTED TO BY
                     0467  ;       THE SECOND LONGWORD.  THIS IS USED FOR SETTING/CLEARING
                     0467  ;       LONGWORD FLAGS.
                     0467  ;
                     0467  ;--
                     0467
                     0467  P2$SETLONG::
   00 B8    88   D0  0467          MOVL    (R8)+,@(R8)              ;STORE THE LONGWORD
                 05  046B          RSB                              ;ALL DONE
                     046C
                     046C  ;++
                     046C  ; FUNCTIONAL DESCRIPTION:
                     046C  ;
                     046C  ;       THESE TWO ROUTINES, P2$SETFLAG AND P2$CLRFLAG, SET AND
                     046C  ;       CLEAR FLAGS IN THE MACRO FLAGS WORD MAC$GL_FLAGS.  THE
                     046C  ;       ONE BYTE OF DATA IS THE FLAG NUMBER TO SET7CLEAR.
                     046C  ;
                     046C  ;--
                     046C
                     046C  P2$SETFLAG::
      50    68   9A  046C          MOVZBL  (R8),R0                  ;GET FLAG NUMBER
   00 6B    50   E3  046F          BBCS    R0,(R11),.+1             ;SET THE FLAG
                 05  0473          RSB
                     0474
                     0474  P2$CLRFLAG::
      50    68   9A  0474          MOVZBL  (R8),R0                  ;GET FLAG NUMBER
   00 6B    50   E5  0477          BBCC    R0,(R11),.+1             ;CLEAR THE FLAG
                 05  047B          RSB
```

```
                            047C          .SBTTL  PASS 2 ERROR PROCESSING
                            047C
                            047C  ;++
                            047C  ; FUNCTIONAL DESCRIPTION:
                            047C  ;
                            047C  ;       THIS ROUTINE PROCESSES ERRORS DETECTED DURING PASS 2.
                            047C  ;       IT SETS UP A TWO LONGWORD BLOCK AND CALLS THE ERROR
                            047C  ;       PROCESSOR.
                            047C  ;
                            047C  ; INPUTS:
                            047C  ;
                            047C  ;       R0      MESSAGE INDEX
                            047C  ;
                            047C  ;--
                            047C
                            047C  MAC$PASS_2_ERR::
                 58    DD   047C          PUSHL   R8                      ;SAVE INT. BUFFER POINTER
      58  0004'CF      9E   047E          MOVAB   W^MAC$GL_P2_ERRPT+4,R8  ;POINT TO TEMP BUFFER
                 68    D4   0483          CLRL    (R8)                    ; No error pointer
            78   50    D0   0485          MOVL    R0,-(R8)                ;SET ERROR CODE
                 77    10   0488          BSBB    P2$ERR                  ;PROCESS THE ERROR
                 58  8ED0   048A          POPL    R8                      ;RESTORE POINTER
                       05   048D          RSB
```

```
                          048E          .SBTTL  .PRINT DIRECTIVE PROCESSING
                          048E
                          048E  ;++
                          048E  ; FUNCTIONAL DESCRIPTION:
                          048E  ;
                          048E  ;       THIS ROUTINE DOES THE PASS 2 PART OF .PRINT DIRECTIVE PROCESSING.
                          048E  ;       IF THERE WAS AN EXPRESSION, THE EXPRESSION IS SET INTO THE
                          048E  ;       BUFFER, FOLLOWED BY THE STRING FOLLOWING THE SEMI-COLON IN
                          048E  ;       THE .PRINT DIRECTIVE.  THE WHOLE LINE IS THEN OUTPUT TO THE
                          048E  ;       TERMINAL AND THE LISTING (IF THERE IS ONE).
                          048E  ;
                          048E  ;--
                          048E
                          048E  P2$PRT::
              FB6F'  30   048E          BSBW    MAC$WRTLST                      ;WRITE LINE TO LISTING IF WE CAN
            0000'CF  DD   0491          PUSHL   W^MAC$GL_LIST_LVL               ;SAVE LISTING LEVEL
 0000'CF    0064 8F  3C   0495          MOVZWL  #100,W^MAC$GL_LIST_LVL          ;ENSURE IT GETS LISTED
            00E0 8F  BB   049C          PUSHR   #^M<R5,R6,R7>                   ;SAVE REGISTERS
      51    0000'CF  9E   04A0          MOVAB   W^MAC$AB_LST_END,R1             ;POINT TO THE START OF THE BUFFER
                 26  10   04A5          BSBB    GET_VAL_AND_TXT                 ;SET VALUE AND TEXT INTO BUFFER
    53   00000000'8F  C2  04A7          SUBL2   #MAC$AB_LINEBF,R3               ;FIGURE LENGTH OF LINE
           0000'CF  53  D0  04AE        MOVL    R3,W^MAC$GL_LINELN              ;SET FOR OUTPUT ROUTINES
              FB4A'  30   04B3          BSBW    MAC$WRITE_TERM                  ;WRITE LINE TO TERMINAL
            00E0 8F  BA   04B6          POPR    #^M<R5,R6,R7>                   ;RESTORE REGISTERS
            0000'CF 8ED0  04BA          POPL    W^MAC$GL_LIST_LVL               ;AND THE LISTING LEVEL
            0000'CF  D4   04BF          CLRL    W^MAC$GL_VALUE                  ;ENSURE NO CARRY-OVER
 0000'CF  00000000'8F  CE 04C3          MNEGL   #MAC$K_LIST_SIZE,W^MAC$GL_LINELN ;PREVENT PRINTING
                          04CC                                                  ; OF LINE IN LISTING
                     05   04CC          RSB
                          04CD  ;++
                          04CD  ; FUNCTIONAL DESCRIPTION:
                          04CD  ;
                          04CD  ;       THIS ROUTINE GETS THE VALUE (IF ANY) SUPPLIED WITH
                          04CD  ;       .ERROR/.WARN/.PRINT DIRECTIVES AND SETS IT INTO THE
                          04CD  ;       BUFFER.  IT THEN STORES THE TEXT IN THE BUFFER.
                          04CD  ;
                          04CD  ; INPUTS:
                          04CD  ;
                          04CD  ;       R1      BUFFER POINTER
                          04CD  ;
                          04CD  ; OUTPUTS:
                          04CD  ;
                          04CD  ;       R3      UPDATED POINTER
                          04CD  ;
                          04CD  ;--
                          04CD
                          04CD  GET_VAL_AND_TXT:
      50    0000'CF  D0   04CD          MOVL    W^MAC$GL_VALUE,R0               ;GET THE EXPRESSION VALUE
                 09  13   04D2          BEQL    10$                             ;BRANCH IF NO EXPRESSION
              FB29'  30   04D4          BSBW    MAC$DEC_OUT_L2X                 ;OUTPUT EXPRESSION INTO BUFFER
            51   50  D0   04D7          MOVL    R0,R1                           ;SET UPDATED POINTER
            81   20  90   04DA          MOVB    #^A/ /,(R1)+                    ;SEPARATE BY A SPACE
            53   51  D0   04DD  10$:    MOVL    R1,R3                           ;POSITION POINTER INTO R3
      56    0000'CF  D0   04E0          MOVL    W^MAC$GL_ETXLEN,R6              ;GET LENGTH OF TEXT LINE
 0000'CF    56   3B  3A   04E5          LOCC    #^A/;/,R6,W^MAC$AB_ETXBUF ;LOCATE THE SEMI-COLON IN TEXT
                 07  13   04EB          BEQL    20$                             ;IF EQL NO SEMI-COLON--NO TEXT
                 50  D7   04ED          DECL    R0                              ;DON'T COUNT THE SEMI-COLON
```

```
      63   01 A1    50    28  04EF   876          MOVC3   R0,1(R1),(R3)              ;COPY TEXT INTO OUTPUT BUFFER
                          05  04F4   877 20$:     RSB                                ;DONE
```

```
                          04F5  ;++
                          04F5  ; FUNCTIONAL DESCRIPTION:
                          04F5  ;
                          04F5  ;         THESE ROUTINES PROCESS ERRORS.  THE TWO LONGWORDS OF DATA ARE
                          04F5  ;         1) THE MESSAGE INDEX AND 2)THE LINE POINTER (POINTING TO THE ERROR)
                          04F5  ;
                          04F5  ;--
                          04F5
                          04F5  P2$INFO::
           0000'CF    D6  04F5          INCL    W^MAC$GL_INFOCNT        ; Count an information message
                0A    11  04F9          BRB     ERR_COM                 ; Join common code
                          04FB
                          04FB  P2$WRN::
           0000'CF    D6  04FB          INCL    W^MAC$GL_WARNCT         ;COUNT A WARNING
                04    11  04FF          BRB     ERR_COM                 ;JOIN COMMON CODE
                          0501
                          0501  P2$ERR::
           0000'CF    D6  0501          INCL    W^MAC$GL_ERRCT          ;COUNT AN ERROR
                          0505
                          0505  ERR_COM:
           0000'CF    DD  0505          PUSHL   W^MAC$GL_LIST_LVL       ;SAVE CURRENT LISTING LEVEL
  0000'CF  0064 8F    3C  0509          MOVZWL  #100,W^MAC$GL_LIST_LVL  ;ENSURE ERRORS GET LISTED
           12FE 8F    BB  0510          PUSHR   #^M<R1,R2,R3,R4,R5,R6,R7,R9,R12> ;SAVE REGISTERS
                          0514  ;
                          0514  ; SAVE THE LINE/PAGE IN VIRTUAL MEMORY FOR PRINTING AT END
                          0514  ;
        50 0004'CF    9E  0514          MOVAB   W^MAC$GL_ERR_LIST+4,R0  ;POINT TO ERROR LIST HEAD
             59 60    D0  0519          MOVL    (R0),R9                 ;GET LAST BLOCK OF ERRORS ADDRESS
          56 08 A9    9E  051C          MOVAB   8(R9),R6                ;POINT TO BOOKEEPING WORDS IN CASE IT EXISTS
          FC B0 59    D1  0520          CMPL    R9,@-4(R0)              ;DOES THE LIST EXIST?
                19    12  0524          BNEQ    20$                     ;IF NEQ YES
                          0526  ;
                          0526  ; ALLOCATE TWO PAGES TO STORE LINES/PAGES ON
                          0526  ;
             FAD7'    30  0526  10$:    BSBW    MAC$ALL_2_PAGES         ;ALLOCATE TWO PAGES OF VM
             59 50    D0  0529          MOVL    R0,R9                   ;POINT TO THE BLOCK
          56 08 A9    9E  052C          MOVAB   8(R9),R6                ;POINT R6 PAST 2 LONGWORDS OF QUEUE LINKS
        0004'DF 69    0E  0530          INSQUE  (R9),@W^MAC$GL_ERR_LIST+4 ;INSERT INTO THE ERROR QUEUE
          66 10 A9    9E  0535          MOVAB   16(R9),(R6)             ;SET POINTER TO FIRST FREE
   0C A9  007E 8F     3C  0539          MOVZWL  #<<1024-16>/8>,12(R9)   ;SET COUNT OF LINES/PAGES IN THIS CHUNK
             0C A9    D5  053F  20$:    TSTL    12(R9)                  ;SEE IF ROOM IN THIS CHUNK
                E2    15  0542          BLEQ    10$                     ;IF LEQ NO--ALLOCATE ONE
             59 66    D0  0544          MOVL    (R6),R9                 ;YES--GET POINTER TO FREE SPOT
        50 0000'CF    3C  0547          MOVZWL  W^MAC$GW_LST_LINE,R0    ; Get listing line number
          50 50 10    78  054C          ASHL    #16,R0,R0               ; Shift to upper word (lower owrd 0)
          0B 6B 27    E1  0550          BBC     #FLG$V_UPDFIL,(R11),25$ ; Branch if file  not being updated
      001C'CF 02      E1  0554          BBC     #SUM_V_SRCUPD, -        ; Branch if line form source file
                05        0559                  W^MAC$GT_SCB+SUM_B_FLAGS,25$
        50 0000'CF    B0  055A          MOVW    W^MAC$GW_LST_INST,R0    ; Put insert number into low word
        51 0000'CF    D0  055F  25$:    MOVL    W^MAC$GL_SRCPAG,R1      ;AND CURRENT PAGE NUMBER
          FC A9 50    D1  0564          CMPL    R0,-4(R9)               ;LINE/PAGE ALREADY RECORDED?
                06    12  0568          BNEQ    30$                     ;IF NEQ NO
          F8 A9 51    D1  056A          CMPL    R1,-8(R9)               ;MAYBE--WHAT ABOUT PAGE?
                0B    13  056E          BEQL    40$                     ;IF EQL YES
             89 51    D0  0570  30$:    MOVL    R1,(R9)+                ;STORE PAGE NUMBER
             89 50    D0  0573          MOVL    R0,(R9)+                ;AND LINE NUMBER
             86 59    D0  0576          MOVL    R9,(R6)+                ;SET POINTER TO NEXT FREE
```

```
                   66  D7  0579          DECL    (R6)                    ;DECREASE COUNT OF REMAINING SPACE
                           057B 40$:
                           057B ERR_COM_0:
                   59  D4  057B          CLRL    R9                      ; Assume not generated error or warning
                02 A8  B5  057D          TSTW    2(R8)                   ; Is facility value zero?
                   08  12  0580          BNEQ    3$                      ; No if NEQ
   02 A8   007D 8F     A8  0582          BISW    #<MAC$_NORMAL@-16>,2(R8) ; Make it a MACRO error
                   08  11  0588          BRB     6$
                           058A 3$:
   0084 8F      02 A8  B1  058A          CMPW    2(R8),#<SUM$_NORMAL@-16> ; Is it a SUM error?
                   41  13  0590          BEQL    30$                     ; Yes if EQL
                           0592 6$:
      908A 8F      68  B1  0592          CMPW    (R8),#<MAC$_GENERR&^XFFFF> ; Is this a generated warning?
                   07  13  0597          BEQL    10$                     ; Yes if EQL
      8818 8F      68  B1  0599          CMPW    (R8),#<MAC$_GENWRN&^XFFFF> ; of a generated error?
                   04  12  059E          BNEQ    20$                     ; No if NEQ
                   59  D6  05A0 10$:     INCL    R9                      ; Flag generated error/warning
                   2F  11  05A2          BRB     30$                     ;AND SKIP PRINTING ON TERMINAL
                           05A4 ;
                           05A4 ; FIGURE COLUMN OF ERROR AND SAVE FOR LATER.  MUST BE DONE NOW SINCE
                           05A4 ; WRTLST RE-INITIALIZES THE BUFFER.
                           05A4 ;
      56      0000'CF  9E  05A4 20$:     MOVAB   W^MAC$AB_LINEBF,R6      ;POINT TO THE SOURCE BUFFER
                   54  D4  05A9          CLRL    R4                      ;INIT COLUMN COUNT
                04 A8  D5  05AB          TSTL    4(R8)                   ; Is there an error pointer?
                   1D  13  05AE          BEQL    25$                     ; No if EQL
             09    86  91  05B0 21$:     CMPB    (R6)+,#TAB              ;IS THIS A TAB?
                   03  12  05B3          BNEQ    22$                     ;IF NEQ NO
             54    07  C8  05B5          BISL2   #7,R4                   ;YES--SKIP TO NEXT TAB STOP
                   54  D6  05B8 22$:     INCL    R4                      ;NEXT CHARACTER POSITION
          04 A8    56  D1  05BA          CMPL    R6,4(R8)                ;UP TO THE ERROR YET?
                   F0  1F  05BE          BLSSU   21$                     ;IF LSSU NO
                   54  D7  05C0          DECL    R4                      ; Need 1 less space so that error
                           05C2                                          ; pointer is at this position
   00000084 8F     54  D1  05C2          CMPL    R4,#132                 ; Over end of line?
                   02  19  05C9          BLSS    25$                     ; No if LSS
                   54  D4  05CB          CLRL    R4                      ; Don't have error pointer
                           05CD 25$:
               FA30'   30  05CD          BSBW    MAC$WRITE_TERM          ;WRITE LINE TO TERMINAL
               FA2D'   30  05D0          BSBW    MAC$WRTLST              ;WRITE LINE TO LISTING FILE
                           05D3 30$:
                           05D3          $GETMSG_S (R8),W^MAC$GL_LINEPT, - ; Get message text
                           05D3                  L^MAC$GQ_LISTBFDS
   00000004'EF  0000'CF  C1  05EA        ADDL3   W^MAC$GL_LINEPT, -      ; Set pointer
                   53      05F3                  L^MAC$GQ_LISTBFDS+4,R3
             56    53  D0  05F4          MOVL    R3,R6                   ;UPDATE POINTER
             10    59  E9  05F7          BLBC    R9,35$                  ; Branch if not generated error/warning
             51    56  D0  05FA          MOVL    R6,R1                   ;YES--SET POINTER
                   54  DD  05FD          pushl   r4                      ; Protect the column count.
               FECB    30  05FF          BSBW    GET_VAL_AND_TXT         ;GET THE VALUE AND TEXT
                54   8ED0  0602          popl    r4
             56    53  D0  0605          MOVL    R3,R6                   ;SET UPDATED POINTER
                   1B  11  0608          BRB     60$                     ;JOIN COMMON CODE
                           060A ;
                           060A ; PAD LINE WITH SPACES UP TO THE POINT OF THE ERROR
                           060A ;
   55   56   00000000'8F  C3  060A 35$:  SUBL3   #MAC$AB_LINEBF,R6,R5    ;FIGURE CURRENT COLUMN
```

```
                  54    D5  0612   993        TSTL    R4                          ; Is there to be an error pointer?
                  0F    13  0614   994        BEQL    60$                         ; No if EQL
            54    55    D1  0616   995 40$:   CMPL    R5,R4                       ;HAVE WE PADDED ENOUGH?
                  07    18  0619   996        BGEQ    50$                         ;IF GEQ YES
            86    20    90  061B   997        MOVB    #^A/ /,(R6)+                ;NO--PAD WITH SPACES
                  55    D6  061E   998        INCL    R5                          ;MOVE TO NEXT COLUMN
                  F4    11  0620   999        BRB     40$                         ;CONTINUE TILL DONE
                            0622  1000 ;
                            0622  1001 ; NOW POINT TO ERROR WITH EXCLAMATION MARK
                            0622  1002 ;
            86    21    90  0622  1003 50$:   MOVB    #^A/!/,(R6)+                ;MARK THE ERROR
50    56  00000000'8F   C3  0625  1004 60$:   SUBL3   #MAC$AB_LINEBF,R6,R0        ;FIGURE LENGTH OF LINE
      0000'CF     50    3C  062D  1005        MOVZWL  R0,W^MAC$GL_LINELN          ;SAVE FOR OUTPUT ROUTINES
               F9CB'    30  0632  1006        BSBW    MAC$WRITE_TERM              ;WRITE TO THE TERMINAL
               F9C8'    30  0635  1007        BSBW    MAC$WRTLST                  ;WRITE TO LISTING
            12FE 8F     BA  0638  1008 80$:   POPR    #^M<R1,R2,R3,R4,R5,R6,R7,R9,R12> ;RESTORE REGISTERS
            0000'CF   8ED0  063C  1009        POPL    W^MAC$GL_LIST_LVL           ;RESTORE LISTING LEVEL
            0000'CF     D4  0641  1010        CLRL    W^MAC$GL_VALUE              ;ENSURE NO CARRY-OVER ON VALUE
                        05  0645  1011        RSB
```

```
                                  0646  ;++
                                  0646  ; Functional description:
                                  0646  ;
                                  0646  ;       This routine reports Source_update-merge errors
                                  0646  ;
                                  0646  ;--
                                  0646
                                  0646  P2$SUME::
                      F9B7'   30  0646          BSBW    MAC$GETLIN                      ; Get error line into buffer
              56    0000'CF   9E  0649          MOVAB   W^MAC$AB_LST_END,R6             ; Set pointer to start of line
              58    0000'CF   3C  064E          MOVZWL  W^MAC$GL_LINELN,R8              ; Get line length
        66  0000'CF      58   28  0653          MOVC3   R8,W^MAC$AB_LINEBF,(R6)         ; Copy to start of line
                  58     02   C0  0659          ADDL2   #2,R8                           ; Insert two spaces
                  56     58   C0  065C          ADDL2   R8,R6                           ; Point to end of line
              86    2020 8F   B0  065F          MOVW    #^A/  /,(R6)+
            58   00000000'EF  9E  0664          MOVAB   MAC$GT_SCB,R8                   ; Get address of SUM control block
       66   14 B8     10 A8   28  066B          MOVC3   SUM_Q_FILESP+0(R8), -           ; Copy file specificaition
                                  0671                  @SUM_Q_FILESP+4(R8),(R6)        ;  to line
              56      10 A8   A0  0671          ADDW2   SUM_Q_FILESP+0(R8),R6           ; Point to end of line
     50   56   00000000'8F    C3  0675          SUBL3   #MAC$AB_LINEBF,R6,R0            ; Figure length of line
           0000'CF       50   3C  067D          MOVZWL  R0,W^MAC$GL_LINELN              ; and save for output routines
                      F97B'   30  0682          BSBW    MAC$WRITE_TERM                  ; Write to terminal
                      F978'   30  0685          BSBW    MAC$WRTLST                      ; and to listing file
                              05  0688          RSB
                                  0689
                                  0689          .END
```

```
AUD$K_SIZE      = 00000010
BIT...          = 00000005
BLNK            = 00000020
BYT_COM           00000065 R    04
CHR$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$V_SYM_DLM   = 00000001
CK_NWC            000001A6 R    04
CR              = 0000000D
DUMP_MACRO        0000041C R    04
EOM$C_ABORT     = 00000003
EOM$C_ERROR     = 00000002
EOM$C_SUCCESS   = 00000000
EOM$C_WARNING   = 00000001
ERR_COM           00000505 R    04
ERR_COM_O         0000057B R    04
FF              = 0000000C
FLG$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$V_XCRF      = 0000001F
GET_VAL_AND_TXT   000004CD R    04
HASHSZ          = 0000007F
HYPHEN          = 0000002D
INP$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_ETXBUF     ********   X  04
MAC$AB_LINEBF     ********   X  04
MAC$AB_LST_END    ********   X  04
MAC$AB_LST_LIN    ********   X  04
MAC$AB_LST_OP2    ********   X  04
MAC$AB_SBT_SBTL   ********   X  04
MAC$AB_SEQ_NUM    ********   X  04
MAC$AB_TMPBUF     ********   X  04
MAC$ALC_2_PAGES   ********   X  04
MAC$AL_VALSTACK   ********   X  04
MAC$CHKBYT        ********   X  04
MAC$CK_BYT_TRUN   ********   X  04
MAC$CK_WRD_TRUN   ********   X  04
MAC$DEC_OUT_L2X   ********   X  04
MAC$DEC_OUT_R2L   ********   X  04
MAC$FORM_LINENO   000003A3 RG   04
MAC$GETLIN        ********   X  04
MAC$GL_ERRCT      ********   X  04
MAC$GL_ERR_LIST   ********   X  04
MAC$GL_ETXLEN     ********   X  04
MAC$GL_INFOCNT    ********   X  04
MAC$GL_LINELN     ********   X  04
MAC$GL_LINENUM    ********   X  04
MAC$GL_LINEPT     ********   X  04
MAC$GL_LIST_IT    ********   X  04
MAC$GL_LIST_LVL   ********   X  04
MAC$GL_LIST_PTR   ********   X  04
MAC$GL_MLIN_LEN   ********   X  04
MAC$GL_P2_ERRPT   00000000 RG   03
MAC$GL_P2_LINE    ********   X  04
```

```
MAC$GL_PC          *********  X   04
MAC$GL_PSECT       *********  X   04
MAC$GL_PSECTPTR    *********  X   04
MAC$GL_RECTYP      *********  X   04
MAC$GL_SRCPAG      *********  X   04
MAC$GL_VALUE       *********  X   04
MAC$GL_WARNCT      *********  X   04
MAC$GQ_LISTBFDS    *********  X   04
MAC$GT_SCB         *********  X   04
MAC$GW_LST_INST    *********  X   04
MAC$GW_LST_LINE    *********  X   04
MAC$K_LIST_SIZE    *********  X   04
MAC$LIST_BYTES     *********  X   04
MAC$LIST_BYT_O     *********  X   04
MAC$LST_CHAR       *********  X   04
MAC$OUTOBJ         *********  X   04
MAC$OUT_LW         00000028 RG    04
MAC$PASS_2_ERR     0000047C RG    04
MAC$SET_PC         *********  X   04
MAC$STOIM          *********  X   04
MAC$SYMNAMOUT      *********  X   04
MAC$WRITE_TERM     *********  X   04
MAC$WRTLST         *********  X   04
MAC$WRTOBJ         *********  X   04
MAC$_GENERR      = 007D908A
MAC$_GENWRN      = 007D8818
MAC$_NORMAL      = 007D8001
MAC_SUBSYS       = 0000007D
OBJ$C_EOM_ABORT= 00000003
OBJ$C_EOM_ERR    = 00000002
OBJ$C_EOM_OK     = 00000000
OBJ$C_EOM_WRN    = 00000001
OBJ$C_GSD        = 00000001
OBJ$C_GSD_EPM    = 00000002
OBJ$C_GSD_PSC    = 00000000
OBJ$C_TIR        = 00000002
OBJ$K_BUFSIZ     = 00000200
OPF$M_LASTOPR    = 00002000
OPF$M_OPTEXP     = 00001000
OPF$V_LASTOPR    = 0000000D
OPF$V_OPTEXP     = 0000000C
P2$ASN             00000350 RG    04
P2$AUGPC           00000035 RG    04
P2$CHKL            00000370 RG    04
P2$CLRFLAG         00000474 RG    04
P2$EPT             000002D8 RG    04
P2$ERR             00000501 RG    04
P2$ETX             000003F4 RG    04
P2$FNEWL           00000438 RG    04
P2$INFO            000004F5 RG    04
P2$MACL            00000405 RG    04
P2$NEWL            0000036A RG    04
P2$NEWP            000001B1 RG    04
P2$PRIL            00000445 RG    04
P2$PRT             0000048E RG    04
P2$PSECT           00000206 RG    04
P2$REST            00000217 RG    04
P2$SAVE            000002CA RG    04
P2$SBTTL           0000045B RG    04
P2$SETFLAG         0000046C RG    04
P2$SETLONG         00000467 RG    04
P2$SPID            00000083 RG    04
P2$STIB            00000142 RG    04
P2$STIL            0000017F RG    04
P2$STIW            00000166 RG    04
P2$STKEPT          0000033B RG    04
P2$STKPC           00000000 RG    04
P2$STOB            0000005F RG    04
P2$STOL            0000008B RG    04
P2$STOV            00000093 RG    04
P2$STOW            00000075 RG    04
P2$STRB            000000CD RG    04
P2$STRL            000000F3 RG    04
P2$STRSB           000000C5 RG    04
P2$STRSW           000000DC RG    04
P2$STRW            000000E4 RG    04
P2$STSB            00000057 RG    04
P2$STSW            0000006D RG    04
P2$SUME            00000646 RG    04
P2$WRN             000004FB RG    04
P2_LONG            00000099 R     04
PSC$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
PSECT$BLANK        *********  X   04
PSECT_REST         00000226 R     04
RDX$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
SIZ...           = 00000001
STB$K_PG_MISS    = 0000000A
STO_COM            000000A1 R     04
STO_DAT            00000193 R     04
STO_LOOP           00000100 R     04
STRB_COM           000000D3 R     04
STRW_COM           000000EA R     04
SUM$_NORMAL      = 00848001
SUM_B_FLAGS        0000001C
SUM_K_BLN          0000001D
SUM_L_ISDATA       00000004
SUM_L_STS          00000000
SUM_M_AUDIT      = 00000001
SUM_M_AUDITNEW   = 00000002
SUM_M_DELETE     = 00000010
SUM_M_SRCUPD     = 00000004
SUM_M_SUBCLSH    = 00000008
SUM_Q_AUDDS        00000008
SUM_Q_FILESP       00000010
SUM_V_AUDIT      = 00000000
SUM_V_AUDITNEW   = 00000001
SUM_V_DELETE     = 00000004
SUM_V_SRCUPD     = 00000002
SUM_V_SUBCLSH    = 00000003
SUM_W_INSERT_NO    0000001A
SUM_W_LINE_NO      00000018
SYM$B_NAME         00000004
```

```
SYM$B_SEG             0000000C                    TIR$C_STO_RB    = 00000027
SYM$B_TOKEN           0000000B                    TIR$C_STO_RL    = 0000001F
SYM$F_DEF           = 00000002                    TIR$C_STO_RSB   = 0000001D
SYM$F_REL           = 00000008                    TIR$C_STO_RSW   = 0000001E
SYM$F_UNI           = 00000004                    TIR$C_STO_RW    = 00000028
SYM$F_VALIDATE      = 00000010                    TIR$C_STO_SB    = 00000014
SYM$F_WEAK          = 00000001                    TIR$C_STO_SW    = 00000015
SYM$K_BLKSIZ          0000000D                    TIR$C_STO_VPS   = 00000020
SYM$K_MAXLEN        = 0000001F                    TIR$C_STO_W     = 00000026
SYM$K_TWOCOL        = 00000010                    WORD_COM          0000007B R      04
SYM$L_LINK            00000000                    X1              = 00000033
SYM$L_VAL             00000005                    X2              = 00080000
SYM$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
SYS$GETMSG            ********    GX    04
TAB                 = 00000009
TIR$C_CTL_AUGRB     = 00000051
TIR$C_CTL_SETRB     = 00000050
TIR$C_STA_EPM       = 0000000C
TIR$C_STA_PB        = 00000004
TIR$C_STA_PL        = 00000006
TIR$C_STA_PW        = 00000005
TIR$C_STO_B         = 00000025
TIR$C_STO_L         = 00000016
TIR$C_STO_LW        = 00000016
TIR$C_STO_PIDR      = 0000001B
```

```
                                +-----------------+
                                ! 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
$ABS$                           0000001D  (    29.)  02 (  2.)  NOPIC   USR   CON   ABS   LCL NOSHR   EXE   RD    WRT NOVEC BYTE
MAC$P2ACT2_DATA                 00000008  (     8.)  03 (  3.)  NOPIC   USR   CON   REL   LCL NOSHR NOEXE   RD    WRT NOVEC LONG
MAC$RO_CODE_P2                  00000689  (  1673.)  04 (  4.)  NOPIC   USR   CON   REL   GBL NOSHR   EXE   RD  NOWRT NOVEC LONG

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

Phase                     Page faults    CPU Time       Elapsed Time
-----                     -----------    --------       ------------
Initialization                     29    00:00:00.04    00:00:01.91
Command processing                106    00:00:00.43    00:00:01.55
Pass 1                            388    00:00:08.77    00:00:31.47
Symbol table sort                   0    00:00:00.97    00:00:02.18
Pass 2                            198    00:00:02.19    00:00:06.77
Symbol table output                33    00:00:00.18    00:00:00.18
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              759    00:00:12.60    00:00:44.08

The working set limit was 1650 pages.
71307 bytes (140 pages) of virtual memory were used to buffer the intermediate code.
There were 50 pages of symbol table space allocated to hold 933 non-local and 58 local symbols.
1038 source lines were read in Pass 1, producing 28 object records in Pass 2.
49 pages of virtual memory were used to define 48 macros.

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

Macro library name                         Macros defined
------------------                         --------------
_$255$DUA28:[SHRLIB]SUM.MLB;1                           3
_$255$DUA28:[MACRO.OBJ]MACRO.MLB;1                     12
_$255$DUA28:[SYSLIB]STARLET.MLB;2                       7
TOTALS (all libraries)                                 22

1098 GETS were required to define 22 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:P2ACT2/OBJ=OBJ$:P2ACT2 MSRC$:P2ACT2/UPDATE=(ENH$:P2ACT2)+LIB$:MACRO/LIB+SHRLIB$:SUM/LIB
```

0227 AH-BT13A-SE
VAX/VMS V4.0

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