M 5

```
MMM         MMM     AAAAAAAAA        CCCCCCCCCCCC  RRRRRRRRRRRR        000000000
MMM         MMM     AAAAAAAAA        CCCCCCCCCCCC  RRRRRRRRRRRR        000000000
MMM         MMM     AAAAAAAAA        CCCCCCCCCCCC  RRRRRRRRRRRR        000000000
MMMMMM   MMMMMM  AAA         AAA  CCC              RRR         RRR  000         000
MMMMMM   MMMMMM  AAA         AAA  CCC              RRR         RRR  000         000
MMMMMM   MMMMMM  AAA         AAA  CCC              RRR         RRR  000         000
MMM   MMM   MMM  AAA         AAA  CCC              RRR         RRR  000         000
MMM   MMM   MMM  AAA         AAA  CCC              RRR         RRR  000         000
MMM   MMM   MMM  AAA         AAA  CCC              RRR         RRR  000         000
MMM         MMM  AAA         AAA  CCC              RRRRRRRRRRRR     000         000
MMM         MMM  AAA         AAA  CCC              RRRRRRRRRRRR     000         000
MMM         MMM  AAA         AAA  CCC              RRRRRRRRRRRR     000         000
MMM         MMM  AAAAAAAAAAAAAAA  CCC              RRR   RRR        000         000
MMM         MMM  AAAAAAAAAAAAAAA  CCC              RRR   RRR        000         000
MMM         MMM  AAAAAAAAAAAAAAA  CCC              RRR   RRR        000         000
MMM         MMM  AAA         AAA  CCC              RRR      RRR     000         000
MMM         MMM  AAA         AAA  CCC              RRR      RRR     000         000
MMM         MMM  AAA         AAA  CCC              RRR      RRR     000         000
MMM         MMM  AAA         AAA     CCCCCCCCCCCC  RRR         RRR     000000000
MMM         MMM  AAA         AAA     CCCCCCCCCCCC  RRR         RRR     000000000
MMM         MMM  AAA         AAA     CCCCCCCCCCCC  RRR         RRR     000000000
```

```
**FILE**ID**MACSUB                                                  N 10
```

```
MM      MM    AAAAAA      CCCCCCCC    SSSSSSSS  UU      UU  BBBBBBBB
MM      MM    AAAAAA      CCCCCCCC    SSSSSSSS  UU      UU  BBBBBBBB
MMMM  MMMM  AA      AA  CC          SS          UU      UU  BB      BB
MMMM  MMMM  AA      AA  CC          SS          UU      UU  BB      BB
MM  MM  MM  AA      AA  CC          SS          UU      UU  BB      BB
MM  MM  MM  AA      AA  CC          SS          UU      UU  BB      BB
MM      MM  AA      AA  CC            SSSSSS    UU      UU  BBBBBBBB
MM      MM  AA      AA  CC            SSSSSS    UU      UU  BBBBBBBB
MM      MM  AAAAAAAAAA  CC                  SS  UU      UU  BB      BB
MM      MM  AAAAAAAAAA  CC                  SS  UU      UU  BB      BB
MM      MM  AA      AA  CC                  SS  UU      UU  BB      BB      ....
MM      MM  AA      AA  CC                  SS  UU      UU  BB      BB      ....
MM      MM  AA      AA    CCCCCCCC  SSSSSSSS    UUUUUUUUUU  BBBBBBBB        ....
MM      MM  AA      AA    CCCCCCCC  SSSSSSSS    UUUUUUUUUU  BBBBBBBB        ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

Table of contents

```
0000         .TITLE  MAC$MACSUB SUBROUTINES FOR VAX-11/780 ASSEMBLER
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: 21-AUG-78
0000   ;
0000   ; MODIFIED BY:
0000   ;
0000   ;       V03-002         RRB0031         Rowland R. Bradley      09-Jul-1984
0000   ;               Use Lib$Trim_Filespec to insert the filename on the
0000   ;               listing.
0000   ;
0000   ;       V03-001         RRB0030         Rowland R. Bradley      08-Jul-1984
0000   ;               Copy expanded filename string into the header.  Expanded
0000   ;               filename string is pointed to by FAB$L_FNA because the
0000   ;               routine LIB$FIND_FILE was used to determine the name in
0000   ;               module MAC$GETCMD.  The name was subsequently copied from
0000   ;               the result of the LIB$FIND_FILE.
0000   ;
0000   ;       V02-019         BLS0057         Benn Schreiber          13-Jun-1981
0000   ;               Correct reference to SUM$INIT_EDIT to be General addressing mode
0000   ;
0000   ;--
```

```
    0000                .SBTTL  DECLARATIONS
    0000        ;
    0000        ; INCLUDE FILES:
    0000        ;
    0000
    0000        ;
    0000        ; MACROS:
    0000        ;
    0000
    0000                $RABDEF                         ;DEFINE RAB OFFSETS
    0000                $FABDEF                         ;DEFINE FAB OFFSETS
    0000                $MAC_CTLFLGDEF                  ;DEFINE CONTROL FLAGS
    0000                $MAC_GENVALDEF                  ;DEFINE GENERAL VALUES
    0000                $MAC_MLFDEF                     ; Define MLF offsets
    0177                $MAC_SYMBLKDEF                  ;DEFINE SYMBOL BLOCK OFFSETS
    0000                $MACMSGDEF                      ; Define message codes
    0000                $MAC_MNBDEF                     ; Define MXB offsets
    0008
    0008        ;
    0008        ; EQUATED SYMBOLS:
    0008        ;
    0008
    0008        ;
    0008        ; OWN STORAGE:
    0008        ;
    0008
00000000                .PSECT  MAC$TEMP_STOR,WRT,GBL,LONG
    0000        $DEF    MAC$AB_TMP_SPEC .BLKB   255     ; TMP STOR FOR LIB$TRIM_FILESPEC
    00FF                .ALIGN  LONG
    0100     89
00000000     90          .PSECT  MAC$RO_CODE_P15,NOWRT,GBL,LONG
```

```
                                  0000          .SBTTL  MAC$FAOUT FORMAT ASCII STRINGS
                                  0000
                                  0000  ;++
                                  0000  ; FUNCTIONAL DESCRIPTION:
                                  0000  ;
                                  0000  ;       THESE THREE ROUTINES ARE USED TO FORMAT ASCII STRINGS
                                  0000  ;       USING $FAOL.  THERE ARE THREE ENTRY POINTS:
                                  0000  ;
                                  0000  ;       MAC$FAOUTS      THE ARGUMENT LIST IS PUSHED ONTO THE STACK
                                  0000  ;                       BEFORE CALLING.  THE CALLER MUST CLEAR THE
                                  0000  ;                       STACK ON RETURN
                                  0000  ;
                                  0000  ;       MAC$WRT_FAOUTS  SAME AS MAC$FAOUTS EXCEPT THAT MAC$WRTLST
                                  0000  ;                       IS CALLED TO WRITE THE LINE BEFORE RETURNING.
                                  0000  ;
                                  0000  ;       MAC$FAOUT       THE ARGUMENT LIST IS POINTED TO BY R1
                                  0000  ;
                                  0000  ; INPUTS:
                                  0000  ;
                                  0000  ;       R0              FAO CONTROL STRING
                                  0000  ;
                                  0000  ; OUTPUTS:
                                  0000  ;
                                  0000  ;       MAC$GL_LINELN   ADJUSTED AND READY TO CALL MAC$WRTLST
                                  0000  ;
                                  0000  ;--
                                  0000
                                  0000  MAC$WRT_FAOUTS::
            51     04 AE     9E   0000          MOVAB   4(SP),R1                ;POINT TO THE ARGUMENT LIST
                      07     10   0004          BSBB    MAC$FAOUT               ;FORMAT THE STRING
                   FFF7'     31   0006          BRW     MAC$WRTLST              ;WRITE LINE TO LISTING AND RETURN
                                  0009
                                  0009  MAC$FAOUTS::
            51     04 AE     9E   0009          MOVAB   4(SP),R1                ;POINT TO THE ARGUMENT LIST
                                  000D  MAC$FAOUT::
                                  000D          $FAOL_S CTRSTR=(R0),-           ;FORMAT THE STRING
                                  000D                  OUTBUF=L^MAC$G_LSTBUFDES,-
                                  000D                  OUTLEN=L^MAC$GL_LINELN,-
                                  000D                  PRMLST=(R1)
 0000'CF    00000000'8F      C2   0024          SUBL2   #MAC$K_LIST_SIZE,W^MAC$GL_LINELN
                             05   002D          RSB
```

```
                    002E          .SBTTL  MAC$SET_PC RECORD HIGH WATER PC
                    002E
                    002E    ;++
                    002E    ; FUNCTIONAL DESCRIPTION:
                    002E    ;
                    002E    ;       THIS ROUTINE IS CALLED TO STORE THE PC IN PSC$L_MAXLGTH
                    002E    ;       FOR THE CURRENT PSECT IF IT IS HIGHER THAN THE STORED
                    002E    ;       PC THERE.  IT SHOULD BE CALLED ANY TIME THE PC IS ADJUSTED
                    002E    ;       BACKWARDS, OR BEFORE CHANGING PSECTS.
                    002E    ;
                    002E    ;--
                    002E
                    002E    MAC$SET_PC::                               ;
      50  0000'CF  D0  002E         MOVL    W^MAC$GL_PSECTPTR,R0       ;POINT TO CURRENT PSECT
   05 A0  0000'CF  D1  0033         CMPL    W^MAC$GL_PC,PSC$L_MAXLGTH(R0) ;CURRENT PC HIGHER THAN STORED?
               06  1F  0039         BLSSU   10$                        ;IF LSSU NO
   05 A0  0000'CF  D0  003B         MOVL    W^MAC$GL_PC,PSC$L_MAXLGTH(R0) ;YES--SET NEW PC
                   05  0041    10$: RSB
                       0042
                       0042 ;++
                       0042 ; FUNCTIONAL DESCRIPTION:
                       0042 ;
                       0042 ;       THIS ROUTINE UPDATES MAC$GL_LSB TO THE NEXT AVAILABLE LSB.
                       0042 ;
                       0042 ;--
                       0042
                       0042 MAC$SET_NEW_LSB::
      50  0000'CF  9E  0042         MOVAB   W^MAC$GL_LSB,R0            ;POINT TO THE WORD
      04 A0     60  D1  0047         CMPL    (R0),4(R0)                 ;CAN WE JUST INCREMENT LSB?
               07  1F  004B         BLSSU   10$                        ;IF LSSU NO--MUST USE HIGHEST LSB PLUS 1
      04 A0     60  D0  004D         MOVL    (R0),4(R0)                 ;YES--STORE HIGHEST LSB SO FAR
                60  D6  0051         INCL    (R0)                       ;INCREMENT TO NEXT LSB
                    05  0053         RSB                                ;DONE
   60  04 A0    01  C1  0054   10$:  ADDL3   #1,4(R0),(R0)              ;USE HIGHEST LSB + 1
                    05  0059         RSB
```

```
                                005A             .SBTTL  MAC$HASH_SYM    FORM HASH VALUE FOR SYMBOL IN MAC$AB_TMPSYM
                                005A
                                005A     ;++
                                005A     ; FUNCTIONAL DESCRIPTION:
                                005A     ;
                                005A     ;       THIS ROUTINE ACCUMULATES THE HASH VALUE FOR A SYMBOL.
                                005A     ;
                                005A     ; INPUTS:
                                005A     ;
                                005A     ;       MAC$AB_TMPSYM   THE SYMBOL NAME (LENGTH, NAME)
                                005A     ;
                                005A     ; O'JTPUTS:
                                005A     ;
                                005A     ;       MAC$GL_HSHVAL   ACCUMULATED HASH VALUE
                                005A     ;
                                005A     ;--
                                005A
                                005A     MAC$HASH_SYM::
                         0F  BB 005A             PUSHR   #^M<R0,R1,R2,R3>        ;SAVE REGISTERS
              50    0000'CF  9E 005C             MOVAB   W^MAC$AB_TMPSYM,R0      ;POINT TO THE SYMBOL NAME BLOCK
                    51   80  9A 0061             MOVZBL  (R0)+,R1                ;GET THE # OF CHARS IN NAME
                    52   51  D0 0064             MOVL    R1,R2                   ;SET LOOP COUNT
                    53   80  9A 0067     10$:    MOVZBL  (R0)+,R3                ;GET NEXT CHARACTER
                    51   53  C0 006A             ADDL2   R3,R1                   ;IMPROVE HASH VALUE
                      F7 52  F5 006D             SOBGTR  R2,10$                  ;LOOP FOR ALL CHARACTERS
 0000'CF   51   FFFFFF80 8F  CB 0070             BICL3   #^C<HASHSZ>,R1,-        ;TRIM TO HASH TABLE SIZE
                                007A                     W^MAC$GL_HSHVAL         ;AND STORE HASH VALUE
                         0F  BA 007A             POPR    #^M<R0,R1,R2,R3>        ;RESTORE REGISTERS
                             05 007C             RSB
```

```
                           007D          .SBTTL  TRUNCATION CHECK ROUTINES
                           007D
                           007D  ;++
                           007D  ; FUNCTIONAL DESCRIPTION:
                           007D  ;
                           007D  ;         THIS ROUTINE CHECKS THE VALUE ON THE TOP OF THE VALUE STACK
                           007D  ;         FOR UNSIGNED BYTE TRUNCATION.
                           007D  ;
                           007D  ; INPUTS:
                           007D  ;
                           007D  ;         R5      POINTS TO VALUE TO CHECK
                           007D  ;--
                           007D
                           007D  MAC$CK_BYT_TRU1::
        55    04 AE    DE  007D          MOVAL   4(SP),R5                ;POINT R5 TO THE WORD
                 0A    10  0081          BSBB    MAC$CK_BYT_TRUN         ;CHECK FOR TRUNCATION
              06 50    E8  0083          BLBS    R0,10$                  ;BRANCH IF NO ERROR
              02 A5    B4  0086          CLRW    2(R5)                   ;YES--TRIM TO A BYTE
              01 A5    94  0089          CLRB    1(R5)                   ;...
                       05  008C  10$:    RSB                             ;DONE
                           008D  MAC$CK_BYT_TRUN::
              02 A5    B5  008D          TSTW    2(R5)                   ;POSITIVE VALUE?
                 0C    12  0090          BNEQ    10$                     ;IF NEQ NO
              01 A5    B5  0092          TSTW    1(R5)                   ;YES--UPPER 3 BYTES MUST BE 0
                 38    12  0095          BNEQ    MAC$TRUNC_ERR           ;IF NEQ TRUNCATION ERROR
              03 A5    95  0097          TSTB    3(R5)                   ;...
                 33    12  009A          BNEQ    MAC$TRUNC_ERR
                 0F    11  009C          BRB     20$                     ;ALL IS WELL
 FFFF 8F      02 A5    B1  009E  10$:    CMPW    2(R5),#-1               ;UPPER WORD MUST BE ALL ONES
                 29    12  00A4          BNEQ    MAC$TRUNC_ERR           ;IF NEQ THEN TRUNCATION ERROR
   FF 8F      01 A5    91  00A6          CMPB    1(R5),#-1               ;SECOND BYTE MUST BE ALL ONES
                 22    12  00AB          BNEQ    MAC$TRUNC_ERR           ;IF NEQ TRUNCATION ERROR
          50     01    D0  00AD  20$:    MOVL    #1,R0                   ;RETURN SUCCESS
                       05  00B0          RSB
                           00B1
                           00B1  ;++
                           00B1  ; FUNCTIONAL DESCRIPTION:
                           00B1  ;
                           00B1  ;         THIS ROUTINE CHECKS THE VALUE ON THE TOP OF THE VALUE STACK
                           00B1  ;         FOR WORD TRUNCATION.
                           00B1  ;
                           00B1  ;--
                           00B1
                           00B1  MAC$CK_WRD_TRU1::
        55    04 AE    DE  00B1          MOVAL   4(SP),R5                ;POINT TO WORD IN QUESTION
                 07    10  00B5          BSBB    MAC$CK_WRD_TRUN         ;GO CHECK FOR TRUNCATION
              03 50    E8  00B7          BLBS    R0,10$                  ;BRANCH IF ALL OK
              02 A5    B4  00BA          CLRW    2(R5)                   ;NO--CLEAR TRUNCATION ERROR
                       05  00BD  10$:    RSB                             ;ALL DONE
                           00BE  MAC$CK_WRD_TRUN::
              02 A5    B5  00BE          TSTW    2(R5)                   ;UPPER WORD 0?
                 08    13  00C1          BEQL    10$                     ;IF EQL YES--OK
 FFFF 8F      02 A5    B1  00C3          CMPW    2(R5),#-1               ;NO--IS IT ALL ONES?
                 04    12  00C9          BNEQ    MAC$TRUNC_ERR           ;IF NEQ NO--TRUNCATION ERROR
          50     01    D0  00CB  10$:    MOVL    #1,R0                   ;RETURN SUCCESS
                       05  00CE          RSB
                           00CF
```

```
                        00CF   257 ;
                        00CF   258 ; REPORT TRUNCATION ERROR
                        00CF   259 ;
                        00CF   260
                        00CF   261 MAC$TRUNC_ERR:
   09 6B    0E    E1    00CF   262          BBC     #FLG$V_P2,(R11),10$         ;BRANCH IF PASS 1
                        00D3   263          $MAC_P2_ERR DATATRUNC               ; Report error
                  05    00DB   264          RSB
                        00DC   265 10$:     $MAC_ERR DATATRUNC                  ; Get message code
          FF1C'   30    00E1   266          BSBW    MAC$ERRORPT                 ;ISSUE ERROR TO PASS 2 AND RETURN
            50    D4    00E4   267          CLRL    R0                          ;RETURN ERROR
                  05    00E6   268          RSB
```

```
                 00E7  ;++
                 00E7  ; FUNCTIONAL DESCRIPTION:
                 00E7  ;
                 00E7  ;       CHECK FOR SIGNED BYTE TRUNCATION
                 00E7  ;
                 00E7  ;--
                 00E7
                 00E7  MAC$CK_SBY_TRU1::
55    04 AE  DE  00E7          MOVAL   4(SP),R5                ;POINT TO WORD IN QUESTION
         14  10  00EB          BSBB    MAC$CK_SBY_TRUN         ;CHECK FOR TRUNCATION
      10 50  E8  00ED          BLBS    R0,20$                  ;BRANCH IF ALL IS WELL
         50  D4  00F0          CLRL    R0                      ;ASSUME POS BYTE
         65  95  00F2          TSTB    (R5)                    ;POS OR NEG. BYTE?
         02  18  00F4          BGEQ    10$                     ;IF GEQ POS
         50  D7  00F6          DECL    R0                      ;NEGATIVE--MAKE -1
02 A5    50  B0  00F8  10$:    MOVW    R0,2(R5)                ;STORE -1
01 A5    50  90  00FC          MOVB    R0,1(R5)                ;...
             05  0100  20$:    RSB                             ;DONE
                 0101  MAC$CK_SBY_TRUN::
         51  D4  0101          CLRL    R1                      ;ASSUME POSITIVE
         65  95  0103          TSTB    (R5)                    ;CHECK SIGN OF BYTE
         02  18  0105          BGEQ    10$                     ;BR IF GEQ
         51  D7  0107          DECL    R1                      ;MAKE 0 INTO -1
02 A5    51  B1  0109  10$:    CMPW    R1,2(R5)                ;DOES HIGH WORD HAVE RIGHT SIGN?
         C0  12  010D          BNEQ    MAC$TRUNC_ERR           ;IF NEQ NO--ERROR
01 A5    51  91  010F          CMPB    R1,1(R5)                ;YES--HOW ABOUT SECOND BYTE?
         BA  12  0113          BNEQ    MAC$TRUNC_ERR           ;IF NEQ NO
      50     01  D0  0115      MOVL    #1,R0                   ;RETURN SUCCESS
             05  0118          RSB
                 0119
                 0119  ;++
                 0119  ; FUNCTIONAL DESCRIPTION:
                 0119  ;
                 0119  ;       CHECK FOR SIGNED WORD TRUNCATION
                 0119  ;
                 0119  ;--
                 0119
                 0119  MAC$CK_SWD_TRU1::
55    04 AE  DE  0119          MOVAL   4(SP),R5                ;POINT TO WORD IN QUESTION
         10  10  011D          BSBB    MAC$CK_SWD_TRUN         ;CHECK FOR TRUNCATION
      0C 50  E8  011F          BLBS    R0,20$                  ;BRANCH IF ALL IS WELL
         50  D4  0122          CLRL    R0                      ;ASSUME POS. BYTE
         65  B5  0124          TSTW    (R5)                    ;POSITIVE?
         02  18  0126          BGEQ    10$                     ;IF GEQ YES
         50  D7  0128          DECL    R0                      ;NO--MAKE -1
02 A5    50  B0  012A  10$:    MOVW    R0,2(R5)                ;FIX THE ERROR
             05  012E  20$:    RSB
                 012F  MAC$CK_SWD_TRUN::
         51  D4  012F          CLRL    R1                      ;ASSUME POSITIVE
         65  B5  0131          TSTW    (R5)                    ;CHECK SIGN OF WORD
         02  18  0133          BGEQ    10$                     ;BR IF GEQ
         51  D7  0135          DECL    R1                      ;MAKE 0 INTO -1
02 A5    51  B1  0137  10$:    CMPW    R1,2(R5)                ;SIGN OF UPPER WORD CORRECT?
         92  12  013B          BNEQ    MAC$TRUNC_ERR           ;IF NEQ NO--ERROR
      50  01  D0  013D         MOVL    #1,R0                   ;RETURN SUCCESS
             05  0140          RSB
```

```
                              0141          .SBTTL  MAC$SRC_KEYS    SEARCH KEYWORD LIST WITH ABBREVIATIONS
                              0141
                              0141  ;++
                              0141  ; FUNCTIONAL DESCRIPTION:
                              0141  ;
                              0141  ;       THIS ROUTINE SEARCHES A LINKED LIST FOR A KEYWORD, WITH
                              0141  ;       ABBREVIATIONS TAKEN INTO ACCOUNT.
                              0141  ;
                              0141  ; INPUTS:
                              0141  ;
                              0141  ;       R5      POINTER TO FIRST ELEMENT OF LINKED SYMBOL LIST
                              0141  ;
                              0141  ; OUTPUTS:
                              0141  ;
                              0141  ;       R0      0       NOT FOUND OR MULTIPLE PARTIAL MATCHES
                              0141  ;               1       FOUND
                              0141  ;       R1      0       NOT FOUND OR MULTIPLE PARTIAL MATCHES
                              0141  ;               <>0     ADDRESS OF SYMBOL BLOCK.
                              0141  ;
                              0141  ;--
                              0141
                              0141  MAC$SRC_KEYS::
           03F0 8F       BB   0141          PUSHR   #^M<R4,R5,R6,R7,R8,R9> ;SAVE REGISTERS
             56 55       D0   0145          MOVL    R5,R6                   ;POINT TO LINKED LIST
       57  0000'CF       9E   0148          MOVAB   W^MAC$AB_TMPSYM,R7      ;POINT TO SYMBOL IN QUESTION
          58    87       9A   014D          MOVZBL  (R7)+,R8                ;GET LENGTH OF SYMBOL
                59       D4   0150          CLRL    R9                      ;CLEAR PARTIAL MATCH INDICATOR
                              0152  ;
                              0152  ; CHAIN THROUGH LIST LOOKING FOR SYMBOL
                              0152  ;
         50   04 A6      9A   0152  10$:    MOVZBL  SYM$B_NAME(R6),R0       ; Get offset to symbol count/name
        50   56   50     C3   0156          SUBL3   R0,R6,R0                ; and form its address
             51   80     9A   015A          MOVZBL  (R0)+,R1                ; Get count byte and advance pointer
   60   51   67   58     39   015D          MATCHC  R8,(R7),R1,(R0)         ; Find the substring in here?
                  1D     13   0162          BEQL    60$                     ;IF EQL YES
             56   66     D0   0164  20$:    MOVL    SYM$L_LINK(R6),R6       ; No--link to next
                  E9     12   0167          BNEQ    10$                     ;IF NEQ GO CHECK IT OUT
                  50     7C   0169  30$:    CLRQ    R0                      ;ASSUME NOT FOUND
                  59     D5   016B          TSTL    R9                      ;WAS THERE A PARTIAL MATCH?
                  0D     13   016D          BEQL    50$                     ;IF EQL NO--RETURN NOT FOUND
                  50     D6   016F          INCL    R0                      ;YES--RETURN 1 FOR FOUND
             51   59     D0   0171          MOVL    R9,R1                   ;RETURN ADDRESS IN R1
                  06     11   0174          BRB     50$                     ;GO RETURN
             50   01     D0   0176  40$:    MOVL    #1,R0                   ;RETURN SUCCESS
             51   56     D0   0179          MOVL    R6,R1                   ;...
           03F0   8F     BA   017C  50$:    POPR    #^M<R4,R5,R6,R7,R8,R9>  ;RESTORE REGISTERS
                         05   0180          RSB
                              0181  ;
                              0181  ; HERE IF FOUND
                              0181  ;
                              0181  60$:
         50   04 A6      9A   0181          MOVZBL  SYM$B_NAME(R6),R0       ; Get offset to symbol count/name
        50   56   50     C3   0185          SUBL3   R0,R6,R0                ; and form its address
          01 A0   67     91   0189          CMPB    (R7),1(R0)              ; Did it match on first character?
                  D5     12   018D          BNEQ    20$                     ;IF NEQ NO--NOT A MATCH
                  52     D5   018F          TSTL    R2                      ;WAS IT A PERFECT MATCH?
                  E3     13   0191          BEQL    40$                     ;IF EQL YES--GO RETURN
```

```
MAC
Sym

$$.
$$.
ARG
AUD
BLN
CHR
CHR
CHR
CHR
CHR
CHR
CHR
CHR
CHR
CHR
CHR
CHR
CHR
CHR
CHR
CHR
CR
DSC
DSC
DSC
DSC
DSC
DSC
FAB
FAB
FAB
FAB
FAB
FAB
FAB
FAB
FF
FLG
FLG
FLG
FLG
FLG
FLG
FLG
FLG
FLG
FLG
FLG
FLG
FLG
FLG
FLG
FLG
FLG
FLG
FLG
FLG
```

```
          59   D5  0193    384         TSTL    R9                              ;NO--WAS THERE ALREADY A PARTIAL MATCH?
               D2  12  0195    385         BNEQ    30$                             ;IF NEQ YES--CONFLICT MEANS NOT FOUND
     59   56   D0  0197    386         MOVL    R6,R9                           ;NO--SAVE ADDRESS OF THIS PARTIAL MATCH
          C8   11  019A    387         BRB     20$                             ;CONTINUE LOOKING
```

```
                       019C   389          .SBTTL  CONVERT LOWER CASE TO UPPER CASE
                       019C   390
                       019C   391  ;++
                       019C   392  ; FUNCTIONAL DESCRIPTION:
                       019C   393  ;
                       019C   394  ;        THIS ROUTINE CONVERTS THE (POSSIBLE) LOWER CASE LETTER
                       019C   395  ;        IN R10 TO UPPER CASE.
                       019C   396  ;
                       019C   397  ;--
                       019C   398
                       019C   399 MAC$CVT_LOWER::
 61 8F     5A     91   019C   400          CMPB    R10,#^A/A/+^X20         ;CAN CHARACTER BE LOWER CASE?
           09     1F   01A0   401          BLSSU   10$                     ;IF LSSU NO
 7A 8F     5A     91   01A2   402          CMPB    R10,#^A/Z/+^X20         ;STILL CHECKING FOR LOWER CASE
           03     1A   01A6   403          BGTRU   10$                     ;IF GTRU NOT LOWER CASE
    5A     20     8A   01A8   404          BICB2   #^X20,R10               ;YES--CONVERT TO UPPER CASE
                  05   01AB   405 10$:     RSB
```

```
MAC
Sym

MAC
MLF
MLF
MLF
MLF
MLF
MLF
MLF
MLF
MNB
MNB
MNB
MNB
MNB
MNB
MNB
MNB
MNB
MNB
MXB
MXB
MXB
NAM
NAM
NO
OBJ
OPF
OPF
OPF
OPF
PSC
PSC
PSC
PSC
PSC
PSC
PSC
PSC
PSC
PSC
PSC
PSC
PSC
PSC
PSC
PSC
PSC
PSC
PSC
PSC
PSC
PSC
PSC
PSC
PSC
PSC
PSC
```

```
                          01AC          .SBTTL  MAC$OPEN_INPUT OPEN INPUT FILE
                          01AC
                          01AC  ;++
                          01AC  ; FUNCTIONAL DESCRIPTION:
                          01AC  ;
                          01AC  ;       THIS ROUTINE IS CALLED TO OPEN THE INPUT FILE
                          01AC  ;
                          01AC  ; INPUTS:
                          01AC  ;
                          01AC  ;       R0      ADDRESS OF FDB
                          01AC  ;
                          01AC  ; OUTPUTS:
                          01AC  ;
                          01AC  ;       FILE IS OPENED.  CREATION TIME PLACED IN SUBTITLE BUFFER.
                          01AC  ;
                          01AC  ;--
                          01AC
                          01AC  MAC$OPEN_INPUT::
        00FE 8F      BB   01AC          PUSHR   #^M<R1,R2,R3,R4,R5,R6,R7>
                          01B0                                          ;SAVE REGISTERS
     0000'CF    50   D0   01B0          MOVL    R0,W^MAC$GL_CURINFDB    ;SET AS CURRENT FDB
  003C'CF    08 A0   9E   01B5          MOVAB   8(R0),W^MAC$INPUT_RAB+RAB$L_FAB ;POINT RAB TO FAB
      47 A0     02   90   01BB          MOVB    #2,FAB$B_FSZ+8(R0)      ;SET FIXED AREA SIZE OF 2
         27 A0       9F   01BF          PUSHAB  FAB$B_RFM+8(R0)         ;STACK RECORD FORMAT ADDRESS
                          01C2          $OPEN   FAB=8(R0),-             ;OPEN THE FILE
                          01C2                  ERR=W^MAC$ERR_OPN_INP
         03 50       E8   01D0          BLBS    R0,5$                   ; Branch if OK
          00E0       31   01D3          BRW     100$
                          01D6  5$:
   00 6B     19      E5   01D6          BBCC    #FLG$V_SEQFIL,(R11),10$ ;ASSUME NOT SEQUENCED FILE
      03     9E      91   01DA  10$:    CMPB    @(SP)+,#FAB$C_VFC       ;IS IT SEQUENCED?
             04      12   01DD          BNEQ    20$                     ;IF NEQ NO
   00 6B     19      E3   01DF          BBCS    #FLG$V_SEQFIL,(R11),20$ ;YES--SET FLAG FOR PASS 2
  50     0000'CF     9E   01E3  20$:    MOVAB   W^MAC$INPUT_RAB,R0      ;POINT TO INPUT RAB
2C A0    0000'CF     9E   01E8          MOVAB   W^MAC$GL_RECHDBUF,RAB$L_RHB(R0) ;SET RECORD HEADER BUFFER ADDRESS
                          01EE          $CONNECT RAB=(R0),-             ;CONNECT RECORD STREAM
                          01EE                  ERR=W^MAC$ERR_CONNECT
         03 50       E8   01FB          BLBS    R0, 21$
          00B5       31   01FE          BRW     100$                    ;BRANCH IF ERROR
                          0201  21$:    $ASCTIM_S TIMBUF=L^MAC$AL_FTIM_DSC,- ;CONVERT FILE CREATION DATE/TIME
                          0201                  TIMADR=L^MAC$INPUT_XAB+XAB$Q_CDT ;
                          0218  ;
                          0218  ; COPY FILENAME AND CREATION DATE TIME TO VM FOR PASS 2 (IF PASS 1)
                          0218  ;
                          0218  ;
                          0218  ; COPY FILENAME INTO SUBTITLE BUFFER
                          0218  ;       FIRST MAKE SURE THE FILENAME FITS IN THE SPACE PROVIDED
                          0218  ;
                          0218  .
                          0218          ;
                          0218          ;
                          0218          ; CREATE TWO DESCS
                          0218          ;       POINT ONE TO THE CURRENT FILE SPEC
                          0218          ;       POINT OTHER TO SUBTITLE FILE SPEC
                          0218          ;       THEN TRIM TO FIT
                          0218          ;
                          0218          ;
50     00000000'EF   D0   0218          MOVL    MAC$GL_CURINFDB,R0      ;POINT TO FDB AGAIN
```

```
             51   3C A0    9A  021F    464          MOVZBL  FAB$B_FNS+8(R0),R1        ;GET LENGTH OF FILENAME STRING
             52   34 A0    D0  0223    465          MOVL    FAB$L_FNA+8(R0),R2        ;GET ADDRESS OF FILENAME STRING
   00000000'EF    62   51  28  0227    466          MOVC3   R1,(R2), -
                               022F    467                  L^MAC$AB_TMP_SPEC
                      7E   7C  022F    468          CLRQ    -(SP)
       0000'CE    00'8F    90  0231    469          MOVB    #DSC$K_CLASS_S, -         ;DESC CLASS
                               0237    470                  DSC$B_CLASS(SP)
       0000'CE    00'8F    90  0237    471          MOVB    #DSC$K_DTYPE_T, -         ;DESC TYPE
                               023D    472                  DSC$B_DTYPE(SP)
      50   00000000'EF     D0  023D    473          MOVL    MAC$GL_CURINFDB,R0        ;POINT TO FDB AGAIN
             51   3C A0    9A  0244    474          MOVZBL  FAB$B_FNS+8(R0),R1        ;AGAIN...GET LENGTH OF FILENAME STRING
          0000'CE    51    90  0248    475          MOVB    R1, DSC$W_LENGTH(SP)      ;GET CURRENT LENGTH
   0000'CE   00000000'EF   DE  024D    476          MOVAL   L^MAC$AB_TMP_SPEC, -
                               0256    477                  DSC$A_POINTER(SP)         ;GET ADDRESS OF RESULT
               57     5E   D0  0256    478          MOVL    SP, R7                    ;SETUP POINTER TO STR DESC.
                               0259    479          ;
                      7E   7C  0259    480          CLRQ    -(SP)
       0000'CE    00'8F    90  025B    481          MOVB    #DSC$K_CLASS_S, -         ;DESC CLASS
                               0261    482                  DSC$B_CLASS(SP)
       0000'CE    00'8F    90  0261    483          MOVB    #DSC$K_DTYPE_T, -         ;DESC TYPE
                               0267    484                  DSC$B_DTYPE(SP)
          0000'CE    20    90  0267    485          MOVB    #32, DSC$W_LENGTH(SP)     ;GET CURRENT LENGTH
     0000'CE    0000'CF    DE  026C    486          MOVAL   W^MAC$AB_SBT_FILE, -
                               0273    487                  DSC$A_POINTER(SP)         ;GET ADDRESS OF RESULT
               56     5E   D0  0273    488          MOVL    SP, R6                    ;SETUP POINTER TO STR DESC.
                               0276    489          ;
                      56   DD  0276    490          PUSHL   R6                        ; OUTPUT STRING DESC ADDR
                      57   DD  0278    491          PUSHL   R7                        ; INPUT STRING DESC ADDR
        00000000'GF   02   FB  027A    492          CALLS   #2, G^LIB$TRIM_FILESPEC ; TRIM TO FIT
                   32 50   E9  0281    493          BLBC    R0, 100$                            ; EXIT ON ERROR
                               0284    494          ;
      50   00000000'EF     D0  0284    495          MOVL    MAC$GL_CURINFDB,R0        ; Point to FDB again
             00 6B    27   E5  028B    496          BBCC    #FLG$V_UPDFIL,(R11),30$ ; Assume file is not being updated
                               028F    497 30$:
              0000'CF      9F  028F    498          PUSHAB  W^MAC$INPUT_RAB           ; Push parameters: RAB address
              00B8 C0      DD  0293    499          PUSHL   FAB$C_BLN+NAM$C_BLN+8(R0) ; Update files list address
                               0297    500
                      04   13  0297    501          BEQL    40$                       ; If EQL zero their are no update files
             00 6B    27   E2  0299    502          BBSS    #FLG$V_UPDFIL,(R11),40$ ; Flag this file as being updated
                               029D    503 40$:
              0000'CF      9F  029D    504          PUSHAB  W^MAC$GT_SCB              ; SUM control block
        00000000'GF   03   FB  02A1    505          CALLS   #3,G^SUM$INIT_EDIT        ; Initialise update files
                   0B 50   E9  02A8    506          BLBC    R0,100$                   ; Error if LBC
                   5E 10   C0  02AB    507          ADDL    #16, SP                   ;RESTORE STACK POINTER
                 00FE 8F   BA  02AE    508          POPR    #^M<R1,R2,R3,R4,R5,R6,R7>
                               02B2    509                                            ;RESTORE REGISTERS
               50     01   D0  02B2    510          MOVL    #1,R0                     ;SET SUCCESS
                           05  02B5    511          RSB
                               02B6    512
                 FD47'     31  02B6    513 100$:    BRW     MAC$LAST_CHANCE           ;GO TO LAST CHANCE HANDLER
```

```
                       02B9          .SBTTL  MAC$RESCANCH RESCAN CURRENT CHARACTER
                       02B9
                       02B9      ;++
                       02B9      ; FUNCTIONAL DESCRIPTION:
                       02B9      ;
                       02B9      ;       THIS ROUTINE BACKS UP THE LINE POINTER AND RESETS THE NEXT
                       02B9      ;       CHARACTER SO AS TO RESCAN THE CURRENT CHARACTER.
                       02B9      ;
                       02B9      ;--
                       02B9
                       02B9      MAC$RESCANCH::
50   0000'CF     9E    02B9              MOVAB   W^MAC$GL_LINEPT,R0      ;GET POINTER TO MAC$GL_LINEPT
0000'8F     60   B1    02BE              CMPW    (R0),#MAC$AB_LINEBF     ;AT BEGINNING OF LINE?
            02   13    02C3              BEQL    10$                     ;IF EOL YES
            60   D7    02C5              DECL    (R0)                    ;NO--BACK IT UP
                 05    02C7      10$:    RSB
```

```
                        02C8    532             .SBTTL  MAC$OPTIMIZEXPR DELETE EXPRESSION
                        02C8    533
                        02C8    534 ;++
                        02C8    535 ; FUNCTIONAL DESCRIPTION:
                        02C8    536 ;
                        02C8    537 ;               MAC$OPTIMIZEXPR IS CALLED TO REMOVE THE CODE TO EVALUATE AN
                        02C8    538 ;               EXPRESSION FROM THE INTERMEDIATE BUFFER.  THE EXPRESSION IS
                        02C8    539 ;               POINTED TO BY 'MAC$GL_EXPPTR' AND 'MAC$GL_EXPEND'.  ANY MACRO
                        02C8    540 ;               TEXT WITHIN THIS RANGE IS COPIED DOWN.  THE REST OF THE
                        02C8    541 ;               INTERMEDIATE CODE IS DELETED. ALL POINTERS AND COUNTERS
                        02C8    542 ;               ARE UPDATED (EXCEPT MAC$GL_EXPPTR AND MAC$GL_EXPEND).
                        02C8    543 ;
                        02C8    544 ;--
                        02C8    545
                        02C8    546 MAC$OPTIMIZEXPR::
      01F8 8F     BB    02C8    547             PUSHR   #^M<R3,R4,R5,R6,R7,R8>  ;PRESERVE REGISTERS
58    0000'CF     D0    02CC    548             MOVL    W^MAC$GL_EXPPTR,R8      ;POINT TO EXPRESSION START
57    0000'CF     D0    02D1    549             MOVL    W^MAC$GL_EXPEND,R7      ;AND END OF EXPRESSION
      58    57    D1    02D6    550             CMPL    R7,R8                   ;IS THERE AN EXPRESSION?
            3C    13    02D9    551             BEQL    50$                     ;IF EQL NO
      56    58    D0    02DB    552             MOVL    R8,R6                   ;COPY START OF EXPRESSION POINTER
   FF 8F    66    91    02DE    553 10$:        CMPB    (R6),#^XFF              ;MACRO LINE?
            0D    13    02E2    554             BEQL    20$                     ;IF EQL YES
      50    66    9A    02E4    555             MOVZBL  (R6),R0                 ;NO--EXPRESSION COMMAND--GET LENGTH
      56    50    C0    02E7    556             ADDL2   R0,R6                   ;SKIP THE EXPRESSION
0000'CF     50    C0    02EA    557             ADDL2   R0,W^MAC$GL_INTCNT      ;INCREASE REMAINING BYTES IN BUFFER
            11    11    02EF    558             BRB     30$                     ;
                        02F1    559 ;
                        02F1    560 ; MACRO LINE--COPY DOWN
                        02F1    561 ;
   50    02 A6    B0    02F1    562 20$:        MOVW    2(R6),R0                ;GET MACRO LINE LENGTH
      50    04    C0    02F5    563             ADDL2   #4,R0                   ;COUNT OVERHEAD BYTES
68    66    50    28    02F8    564             MOVC3   R0,(R6),(R8)            ;MOVE MACRO LINE
      58    53    D0    02FC    565             MOVL    R3,R8                   ;UPDATE POINTER
      56    51    D0    02FF    566             MOVL    R1,R6                   ;POINT PAST MACRO TEXT
      57    56    D1    0302    567 30$:        CMPL    R6,R7                   ;END OF EXPRESSION?
            D7    1F    0305    568             BLSSU   10$                     ;IF LSS NO
      53    58    D0    0307    569             MOVL    R8,R3                   ;COPY END OF EXPR. POINTER
50    59    57    C3    030A    570             SUBL3   R7,R9,R0                ;COMPUTE LENGTH OF CODE TO MOVE
            04    15    030E    571             BLEQ    40$                     ;IF LEQ NOTHING
68    66    50    28    0310    572             MOVC3   R0,(R6),(R8)            ;MOVE CODE
      59    53    D0    0314    573 40$:        MOVL    R3,R9                   ;UPDATE FRAME POINTER
      01F8 8F     BA    0317    574 50$:        POPR    #^M<R3,R4,R5,R6,R7,R8>  ;RESTORE REGISTERS
                  05    031B    575             RSB
```

```
                              031C   577          .SBTTL  MAC$SKP_OPR      SKIP TO NEXT OPERAND OR EOL
                              031C   578
                              031C   579 ;++
                              031C   580 ; FUNCTIONAL DESCRIPTION:
                              031C   581 ;
                              031C   582 ;         MAC$SKP_OPR WILL SCAN TO A COMMA OR END-OF-LINE.  THIS IS
                              031C   583 ;         DONE WHEN AN ERROR IS DETECTED, SO AS TO PREVENT MULTIPLE
                              031C   584 ;         ERROR MESSAGES PER OPERAND.  IF AN END-OF-LINE IS ENCOUNTERED
                              031C   585 ;         THE OPERAND FLAG IS CLEARED AND THE BEGINNING OF LINE FLAG IS
                              031C   586 ;         SET.
                              031C   587 ;
                              031C   588 ;--
                              031C   539
                              031C   590 MAC$SKP_OPR::
        17 6B     0D    E1    031C   591          BBC     #FLG$V_OPRND,(R11),40$   ;BRANCH IF NOT IN OPERAND FIELD
           0D     5A    91    0320   592 10$:     CMPB    R10,#CR                  ;YES--IS CHARACTER CR?
                  0A    13    0323   593          BEQL    20$                      ;IF EQL YES
           2C     5A    91    0325   594          CMPB    R10,#^A/,/               ;NO--IS IT A COMMA?
                  0D    13    0328   595          BEQL    40$                      ;IF EQL YES
                FCD3'   30    032A   596          BSBW    MAC$GETCHR               ;NO--GET NEXT CHARACTER
                  F1    11    032D   597          BRB     10$                      ;LOOK FOR NEW LINE OR COMMA
        00 6B     0D    E5    032F   598 20$:     BBCC    #FLG$V_OPRND,(R11),30$   ;CLEAR OPERAND FLAG FOR NEW LINE
        00 6B     01    E2    0333   599 30$:     BBSS    #FLG$V_BOL.(R11),40$     ;FLAG AT BEGINNING OF LINE
                        05    0337   600 40$:     RSB
                              0338   601
                              0338   602 MAC$SKIPSP::
           20     5A    91    0338   603 10$:     CMPB    R10,#^A/ /               ;IS IT A SPACE?
                  10    1A    033B   604          BGTRU   30$                      ;IF GTR NO--NOT SPACE-LIKE EITHER
                  09    13    033D   605          BEQL    20$                      ;IF EQL YES--READ NEXT CHARACTER
     00000000'EF4A      D5    033F   606          TSTL    MAC$AL_CHRTAB[R10]       ;CHECK ENTRY IN TABLE
                  05    12    0346   607          BNEQ    30$                      ;IF NEQ NOT SPACE-LIKE CHARACTER
                FCB5'   30    0348   608 20$:     BSBW    MAC$GETCHR               ;GET NEXT CHARACTER
                  EB    11    034B   60,          BRB     10$                      ;CHECK IT OUT
                        05    034D   610 30$:     RSB                              ;RETURN
                              034E   611
                        034E   613         .SBTTL  MAC$CLOSE_FILES CLOSE ALL FILES
                        034E   614
                        034E   615 ;++
                        034E   616 ; FUNCTIONAL DESCRIPTION:
                        034E   617 ;
                        034E   618 ;       THIS ROUTINE CLOSES ALL POSSIBLY OPEN FILES
                        034E   619 ;
                        034E   620 ;--
                        034E   621
                        034E   622 MAC$CLOSE_FILES::
                        034E   623         $DISCONNECT RAB=W^MAC$INPUT_RAB ;DISCONNECT INPUT RECORD STREAM
       50   0000'CF  D0 0359   624         MOVL    W^MAC$GL_CURINFDB,R0    ;GET PTR TO CURRENT FDB
                        035E   625         $CLOSE  FAB=8(R0)               ;CLOSE INPUT FILE
        16 6B    15  E5 0368   626         BBCC    #FLG$V_OBJXST,(R11),10$ ;BRANCH IF NO OBJECT FILE
                        036C   627         $DISCONNECT RAB=W^MAC$OBJECT_RAB;DISCONNECT OBJECT FILE STREAM
                        0377   628         $CLOSE  FAB=W^MAC$OBJECT_FAB    ;CLOSE OBJECT FILE
                 03  10 0382   629 10$:    BSBB    MAC$CLOSE_LIST          ;CLOSE LISTING FILE
               0045  31 0384   630         BRW     MAC$CLOSE_LIB           ;CLOSE LIBRARIES
                        0387   631 MAC$CLOSE_LIST::
        16 6B    09  E5 0387   632         BBCC    #FLG$V_LSTXST,(R11),20$ ;BRANCH IF NO LISTING FILE
                        038B   633         $DISCONNECT RAB=W^MAC$LIST_RAB  ;DISCONNECT LISTING FILE STREAM
                        0396   634         $CLOSE  FAB=W^MAC$LIST_FAB      ;CLOSE LISTING FILE
                     05 03A1   635 20$:    RSB                             ;RETURN
                        03A2   636
                        03A2   637 ;++
                        03A2   638 ; FUNCTIONAL DESCRIPTION:
                        03A2   639 ;
                        03A2   640 ;       THIS ROUTINE CLOSES ALL OPEN FILES, AND DELETES THE OBJECT
                        03A2   641 ;       AND LISTING FILE.  THIS SHOULD BE CALLED ONLY ON ABORTS.
                        03A2   642 ;
                        03A2   643 ;--
                        03A2   644
                        03A2   645 MAC$CLS_DEL_OBJ::
        25 6B    15  E5 03A2   646         BBCC    #FLG$V_OBJXST,(R11),10$ ;BRANCH IF NO OBJECT FILE
       50   0000'CF  9E 03A6   647         MOVAB   W^MAC$OBJECT_RAB,R0     ;GET POINTER TO OBJECT RAB
        51    3C A0  D0 03AB   648         MOVL    RAB$L_FAB(R0),R1        ;GET POINTER TO FAB
     00 04 A1    0F  E3 03AF   649         BBCS    #FAB$V_DLT,FAB$L_FOP(R1),.+1 ;SET DELETE BIT IN FAB
                 51  DD 03B4   650         PUSHL   R1                      ;SAVE FAB ADDRESS
                        03B6   651         $DISCONNECT RAB=(R0)            ;DISCONNECT RECORD ACCESS
               50 8ED0  03BF   652         POPL    R0                      ;GET FAB ADDRESS BACK
                        03C2   653         $CLOSE  FAB=(R0)                ;CLOSE AND DELETE THE FILE
                     05 03CB   654 10$:    RSB
```

```
                                 03CC              .SBTTL  MAC$CLOSE_LIB   CLOSE MACRO LIBRARIES
                                 03CC
                                 03CC      ;++
                                 03CC      ; FUNCTIONAL DESCRIPTION:
                                 03CC      ;
                                 03CC      ;       THIS ROUTINE CLOSES ALL MACRO LIBRARY FILES.  THE FILES SHOULD
                                 03CC      ;       BE DISCONNECTED FROM RECORD ACCESS.
                                 03CC      ;
                                 03CC      ;--
                                 03CC
                                 03CC      MAC$CLOSE_LIB::
      52    0000'CF      D0      03CC              MOVL    W^MAC$GL_MLB_QUE,R2     ;POINT TO THE FIRST MLB FDB
                  16     13      03D1              BEQL    20$                     ;IF EQL NO LIBRARIES TO CLOSE
                                 03D3      10$:
               14 A2     9F      03D3              PUSHAB  MLF$L_CTINDEX(R2)       ; Address of control table index
   00000000'GF    01     FB      03D6              CALLS   #1,G^LBR$CLOSE          ; Close library file
             52   62     D0      03DD              MOVL    MLF$L_QLINK(R2),R2      ; Link to possible next library
   00000000'8F    52     D1      03E0              CMPL    R2,#MAC$GL_MLB_QUE      ;ALL DONE?
                  EA     12      03E7              BNEQ    10$                     ;IF NEQ NO
                         05      03E9      20$:    RSB                             ;DONE
```

```
                                03EA   677          .SBTTL  ALLOCATE/DEALLOCATE VIRTUAL MEMORY
                                03EA   678
                                03EA   679 ;++
                                03EA   680 ; FUNCTIONAL DESCRIPTION:
                                03EA   681 ;
                                03EA   682 ;        THIS ROUTINE IS CALLED TO ALLOCATE 1 PAGE OF VIRTUAL MEMORY.
                                03EA   683 ;        AN ATTEMPT IS MADE TO GET MEMORY FROM THE RETURNED PAGES LIST.
                                03EA   684 ;        IF THAT FAILS, LIB$GET_VM IS CALLED TO ALLOCATE A NEW PAGE.
                                03EA   685 ;
                                03EA   686 ;--
                                03EA   687
                                03EA   688 MAC$ALL_1_PAGE::
           50   0000'DF   0F    03EA   689          REMQUE  @W^MAC$GL_FREE_LST,R0    ;TRY TO GET A PAGE FROM THE
                                03EF   690                                           ;RETURNED PAGES LIST
                     13   1C    03EF   691          BVC     10$                      ;IF V-CLEAR WE GOT ONE
   00000000'GF   00000000'EF   FA    03F1   692          CALLG   L^MAC$G_1_PAGE,G^LIB$GET_VM ;NONE THERE--ALLOCATE A NEW PAGE
                  20 50   E9    03FC   693          BLBC    R0,NO_MEM                ;BRANCH IF ALLOCATION FAILURE
           50   0000'CF   D0    03FF   694          MOVL    W^MAC$GL_BASEADDR,R0     ;PICK UP THE BLOCK ADDRESS
                          05    0404   695 10$:     RSB                              ;RETURN WITH BLOCK ADDRESS IN R0
                                0405   696
                                0405   697 ;++
                                0405   698 ; FUNCTIONAL DESCRIPTION:
                                0405   699 ;
                                0405   700 ;        THIS ROUTINE IS CALLED TO DEALLOCATE 1 PAGE OF VIRTUAL MEMORY.
                                0405   701 ;        THE DEALLOCATED PAGES ARE PLACED ON A LINKED LIST POINTED TO
                                0405   702 ;        BY MAC$GL_FREE_LST.
                                0405   703 ;
                                0405   704 ;--
                                0405   705
                                0405   706 MAC$DEA_1_PAGE::
           0000'CF     60   0E    0405   707          INSQUE  (R0),W^MAC$GL_FREE_LST   ;INSERT THE PAGE INTO THE FREE LIST
                          05    040A   708          RSB
                                040B   709
                                040B   710 ;++
                                040B   711 ; FUNCTIONAL DESCRIPTION:
                                040B   712 ;
                                040B   713 ;        THIS ROUTINE ALLOCATES TWO CONTIGUOUS PAGES OF MEMORY.
                                040B   714 ;        THE ADDRESS IS RETURNED IN R0.
                                040B   715 ;
                                040B   716 ;--
                                040B   717
                                040B   718 MAC$ALL_2_PAGES::
   00000000'GF   00000000'EF   FA    040B   719          CALLG   L^MAC$G_2_PAGES,G^LIB$GET_VM ;TRY TO GET THE PAGES
                  06 50   E9    0416   720          BLBC    R0,NO_MEM                ;BRANCH IF ALLOCATION ERROR
           50   0000'CF   D0    0419   721          MOVL    W^MAC$GL_BASEADDR,R0     ;GOT IT--GET THE ADDRESS
                          05    041E   722          RSB
                  FBDE'   31    041F   723 NO_MEM:  BRW     MAC$ERR_NOMEM            ;REPORT NO MEMORY ERROR
                                0422   724                                           ;(NO RETURN)
                                0422   725
                                0422   726 ;++
                                0422   727 ; Functional description:
                                0422   728 ;
                                0422   729 ;        This routine deallocates a block of virtual memory.  If the block
                                0422   730 ;        is 1 page it is returned to the free list; if it is >1 page the
                                0422   731 ;        block is returned to the system.
                                0422   732 ;        This routine is used to deallocate MXB blocks.
                                0422   733 ;
```

```
                           0422  ; Inputs:
                           0422  ;         R0 = Address of block to deallocate
                           0422  ;                 Offset MXB$L_PAGES contains the size of the block in pages
                           0422  ;
                           0422  ;--
                           0422
                           0422  MAC$DEAL_BLOCK::
         01    04 A0   D1  0422          CMPL    MXB$L_PAGES(R0),#1       ; Is this block 1 page?
                  DD   13  0426          BEQL    MAC$DEA_1_PAGE           ; Yes if EQL
                  50   DD  0428          PUSHL   R0                       ; Put address on stack
             51   5E   D0  042A          MOVL    SP,R1                    ; and save stack address
      7E     04 A0     09  78  042D      ASHL    #9,MXB$L_PAGES(R0),-(SP) ; Put block size (in bytes) on stack
             50   5E   D0  0432          MOVL    SP,R0                    ; and save stack address
                  51   DD  0435          PUSHL   R1                       ; Form argument block
                  50   DD  0437          PUSHL   R0                       ; on stack
   00000000'GF    02   FB  0439          CALLS   #2,G^LIB$FREE_VM         ; Return virtual memory
             5E   08   C0  0440          ADDL    #<2*4>,SP                ; Clean up stack
                       05  0443          RSB
                           0444
                           0444
                           0444  ;++
                           0444  ; Functional description:
                           0444  ;
                           0444  ;         This routine allocates a block of virtual memory.  The block
                           0444  ;         size is rounded up to 1 page.
                           0444  ;
                           0444  ; Inputs:
                           0444  ;         R1 = Number of bytes required
                           0444  ;
                           0444  ; Outputs:
                           0444  ;         R0 = Address of memory block
                           0444  ;         R1 = Number of pages allocated
                           0444  ;
                           0444  ;--
                           0444
                           0444  MAC$ALL_BLOCK::
51    000001FF 8F      C0  0444          ADDL2   #511,R1                  ; Round number of bytes to multiple
51    000001FF 8F      CA  044B          BICL2   #511,R1                  ; of 512 bytes.
 7E     51     F7 8F   78  0452          ASHL    #-9,R1,-(SP)             ; Also convert to number of pages
             01   6E   D1  0457          CMPL    (SP),#1                  ; Is 1 page required?
                  04   12  045A          BNEQ    10$                      ; No if NEQ
                  8C   10  045C          BSBB    MAC$ALL_1_PAGE           ; Get 1 page
                  1D   11  045E          BRB     20$
                           0460  10$:
                  51   DD  0460          PUSHL   R1                       ; Stack bytes required
             50   5E   D0  0462          MOVL    SP,R0                    ; and save its address
             0000'CF   DF  0465          PUSHAL  W^MAC$GL_BASEADDR        ; Push address to return block address
                  50   DD  0469          PUSHL   R0                       ; and address of bytes required
   00000000'GF    02   FB  046B          CALLS   #2,G^LIB$GET_VM          ; Get memory
               AA 50   E9  0472          BLBC    R0,NO_MEM                ; Branch if error
     50      0000'CF   D0  0475          MOVL    W^MAC$GL_BASEADDR,R0     ; Get base address of allocated block
             5E   04   C0  047A          ADDL    #<1*4>,SP                ; Clean stack
                           047D  20$:
                51 8ED0    047D          POPL    R1                       ; Get pages allocated
                       05  0480          RSB
                           0481  ;
                           0481  ;
     0481   791          .END
```

```
$$.TMP1                  = 00000001                FLG$M_LEXOP                = 00000002
$$.TMP2                  = 00000060                FLG$M_LSTXST               = 00000200
ARG$K_SIZE               = 000003E8                FLG$M_MAC2COL              = 00000800
AUD$K_SIZE               = 00000010                FLG$M_MACL                 = 00000800
BLNK                     = 00000020                FLG$M_MACLTB               = 08000000
CHR$M_COMMA_CR           = 00000020                FLG$M_MACTXT               = 00010000
CHR$M_ILL_CHR            = 00000040                FLG$M_MEBLST               = 00001000
CHR$M_NUM_BER            = 00000010                FLG$M_MOREARG              = 00002000
CHR$M_SPA_MSK            = 00000001                FLG$M_MOREINP              = 00000008
CHR$M_SYM_CH1            = 00000008                FLG$M_NEWPND               = 00000400
CHR$M_SYM_CHR            = 00000004                FLG$M_NOREF                = 01000000
CHR$M_SYM_DLM            = 00000002                FLG$M_NTYPEPC              = 00000020
CHR$V_COMMA_CR           = 00000005                FLG$M_NULCHR               = 00040000
CHR$V_CVTLWC             = 00000061                FLG$M_OBJXST               = 00200000
CHR$V_ILL_CHR            = 00000006                FLG$M_OPNDCHK              = 00000100
CHR$V_NOCVT              = 0000007F                FLG$M_OPRND                = 00002000
CHR$V_NUM_BER            = 00000004                FLG$M_OPTVFLIDX            = 00001000
CHR$V_SPA_MSK            = 00000000                FLG$M_ORDLST               = 00020000
CHR$V_SYM_CH1            = 00000003                FLG$M_P2                   = 00004000
CHR$V_SYM_CHR            = 00000002                FLG$M_RPTIRP               = 10000000
CHR$V_SYM_DLM            = 00000001                FLG$M_SEQFIL               = 02000000
CR                       = 0000000D                FLG$M_SKAN                 = 00008000
DSC$A_POINTER              ********   X   04       FLG$M_SPECOP               = 00000004
DSC$B_CLASS                ********   X   04       FLG$M_SPLALL               = 04000000
DSC$B_DTYPE                ********   X   04       FLG$M_STOIMF               = 00040000
DSC$K_CLASS_S              ********   X   04       FLG$M_SYM2COL              = 00000400
DSC$K_DTYPE_T              ********   X   04       FLG$M_TOCFLG               = 00080000
DSC$W_LENGTH               ********   X   04       FLG$M_UPAFLG               = 00000010
FAB$B_FNS                = 00000034                FLG$M_UPDFIL               = 00000080
FAB$B_FSZ                = 0000003F                FLG$M_UPMARG               = 00000040
FAB$B_RFM                = 0000001F                FLG$M_XCRF                 = 80000000
FAB$C_BLN                = 00000050                FLG$V_ALLCHR               = 00000000
FAB$C_VFC                = 00000003                FLG$V_BOL                  = 00000001
FAB$L_FNA                = 0000002C                FLG$V_CHKLPND              = 00000014
FAB$L_FOP                = 00000004                FLG$V_COMPEXPR             = 00000002
FAB$V_DLT                = 0000000F                FLG$V_CONT                 = 00000003
FF                       = 0000000C                FLG$V_CRF                  = 0000001E
FLG$M_ALLCHR             = 00000001                FLG$V_CRSEEN               = 00000020
FLG$M_BOL                = 00000002                FLG$V_DATRPT               = 00000004
FLG$M_CHKLPND            = 00100000                FLG$V_DBGOUT               = 0000002E
FLG$M_COMPEXPR           = 00000004                FLG$V_DLIMSTR              = 0000002F
FLG$M_CONT               = 00000008                FLG$V_ENDMCH               = 00000005
FLG$M_CRF                = 40000000                FLG$V_EVALEXPR             = 00000006
FLG$M_CRSEEN             = 00000001                FLG$V_EXPOPT               = 00000007
FLG$M_DATRPT             = 00000010                FLG$V_EXTERR               = 00000030
FLG$M_DBGOUT             = 00004000                FLG$V_EXTWRN               = 00000031
FLG$M_DLIMSTR            = 00008000                FLG$V_FIRSTLN              = 00000029
FLG$M_ENDMCH             = 00000020                FLG$V_IFSTAT               = 00000017
FLG$M_EVALEXPR           = 00000040                FLG$V_IIF                  = 00000016
FLG$M_EXPOPT             = 00000080                FLG$V_INSERT               = 00000008
FLG$M_EXTERR             = 00010000                FLG$V_IRPC                 = 0000001D
FLG$M_EXTWRN             = 00020000                FLG$V_LEXOP                = 00000021
FLG$M_FIRSTLN            = 00000200                FLG$V_LSTXST               = 00000009
FLG$M_IFSTAT             = 00800000                FLG$V_MAC2COL              = 0000002B
FLG$M_IIF                = 00400000                FLG$V_MACL                 = 0000000B
FLG$M_INSERT             = 00000100                FLG$V_MACLTB               = 0000001B
FLG$M_IRPC               = 20000000                FLG$V_MACTXT               = 00000010
```

```
FLG$V_MEBLST                 = 0000000C                  MAC$CK_SWD_TRUN                    0000012F RG     04
FLG$V_MOREARG                = 0000002D                  MAC$CK_WRD_TRU1                    000000B1 RG     04
FLG$V_MOREINP                = 00000023                  MAC$CK_WRD_TRUN                    000000BE RG     04
FLG$V_NEWPND                 = 0000000A                  MAC$CLOSE_FILES                    0000034E RG     04
FLG$V_NOREF                  = 00000018                  MAC$CLOSE_LIB                      000003CC RG     04
FLG$V_NTYPEPC                = 00000025                  MAC$CLOSE_LIST                     00000387 RG     04
FLG$V_NULCHR                 = 00000032                  MAC$CLS_DEL_OBJ                    000003A2 RG     04
FLG$V_OBJXST                 = 00000015                  MAC$CVT_LOWER                      0000019C RG     04
FLG$V_OPNDCHK                = 00000028                  MAC$DEAL_BLOCK                     00000422 RG     04
FLG$V_OPRND                  = 0000000D                  MAC$DEA_T_PAGE                      0000405 RG     04
FLG$V_OPTVFLIDX              = 0000002C                  MAC$ERRORPT                        ********   X    04
FLG$V_ORDLST                 = 00000011                  MAC$ERR_CONNECT                    ********   X    04
FLG$V_P2                     = 0000000E                  MAC$ERR_NOMEM                      ********   X    04
FLG$V_RPTIRP                 = 0000001C                  MAC$ERR_OPN_INP                    ********   X    04
FLG$V_SEQFIL                 = 00000019                  MAC$FAOUT                          0000000D RG     04
FLG$V_SKAN                   = 0000000F                  MAC$FAOUTS                         00000009 RG     04
FLG$V_SPECOP                 = 00000022                  MAC$GETCHR                         ********   X    04
FLG$V_SPLALL                 = 0000001A                  MAC$GL_BASEADDR                    ********   X    04
FLG$V_STOIMF                 = 00000012                  MAC$GL_CURINFDB                    ********   X    04
FLG$V_SYM2COL                = 0000002A                  MAC$GL_EXPEND                      ********   X    04
FLG$V_TOCFLG                 = 00000013                  MAC$GL_EXPPTR                      ********   X    04
FLG$V_UPAFLG                 = 00000024                  MAC$GL_FREE_LST                    ********   X    04
FLG$V_UPDFIL                 = 00000027                  MAC$GL_HSHVAL                      ********   X    04
FLG$V_UPMARG                 = 00000026                  MAC$GL_INTCNT                      ********   X    04
FLG$V_XCRF                   = 0000001F                  MAC$GL_LINELN                      ********   X    04
HASHSZ                       = 0000007F                  MAC$GL_LINEPT                      ********   X    04
HYPHEN                       = 0000002D                  MAC$GL_LSB                         ********   X    04
INP$K_BUFSIZ                 = 000003E8                  MAC$GL_MLB_QUE                     ********   X    04
INT$K_BUFSIZ                 = 000013F4                  MAC$GL_PC                          ********   X    04
INT$K_BUFWRN                 = 00001390                  MAC$GL_PSECTPTR                    ********   X    04
LBR$CLOSE                      ********   X    04        MAC$GL_RECHDBUF                    ********   X    04
LIB$FREE_VM                    ********   X    04        MAC$GT_SCB                         ********   X    04
LIB$GET_VM                     ********   X    04        MAC$G_T_PAGE                       ********   X    04
LIB$TRIM_FILESPEC              ********   X    04        MAC$G_2_PAGES                      ********   X    04
LST$K_BUFSIZ                 = 00000086                  MAC$G_LSTBUFDES                    ********   X    04
LST$K_L_P_PAGE               = 0000003C                  MAC$HASH_SYM                       0000005A RG     04
LST$K_TITLE_SIZ              = 00000028                  MAC$INPUT_RAB                      ********   X    04
MAB$B_ARGNO                    00000005                  MAC$INPUT_XAB                      ********   X    04
MAB$B_NAME                     00000004                  MAC$K_LIST_SIZE                    ********   X    04
MAB$K_BLKSIZ                   0000000C                  MAC$LAST_CHANCE                    ********   X    04
MAB$L_DVPTR                    00000008                  MAC$LIST_FAB                       ********   X    04
MAB$L_LINK                     00000000                  MAC$LIST_RAB                       ********   X    04
MAB$W_DVLEN                    00000006                  MAC$OBJECT_FAB                     ********   X    04
MAC$AB_LINEBF                  ********   X    04        MAC$OBJECT_RAB                     ********   X    04
MAC$AB_SBT_FILE                ********   X    04        MAC$OPEN_INPUT                     000001AC RG     04
MAC$AB_TMPSYM                  ********   X    04        MAC$OPTIMIZEXPR                    000002C8 RG     04
MAC$AB_TMP_SPEC                00000000 R      03        MAC$PASS_2_ERR                     ********   X    04
MAC$ALL_1_PAGE                 000003EA RG     04        MAC$RESCANCH                       000002B9 RG     04
MAC$ALL_2_PAGES                0000040B RG     04        MAC$SET_NEW_LSB                    00000042 RG     04
MAC$ALL_BLOCK                  00000444 RG     04        MAC$SET_PC                         0000002E RG     04
MAC$AL_CHRTAB                  ********   X    04        MAC$SKIPSP                         00000338 RG     04
MAC$AL_FTIM_DSC                ********   X    04        MAC$SKP_OPR                        0000031C RG     04
MAC$CK_BYT_TRU1                0000007D RG     04        MAC$SRC_KEYS                       00000141 RG     04
MAC$CK_BYT_TRUN                0000008D RG     04        MAC$TRUNC_ERR                      000000CF R      04
MAC$CK_SBY_TRU1                000000E7 RG     04        MAC$WRTLST                         ********   X    04
MAC$CK_SBY_TRUN                00000101 RG     04        MAC$WRT_FAOUTS                     00000000 RG     04
MAC$CK_SWD_TRU1                00000119 RG     04        MAC$_DATATRUNC                   = 007D8800
```

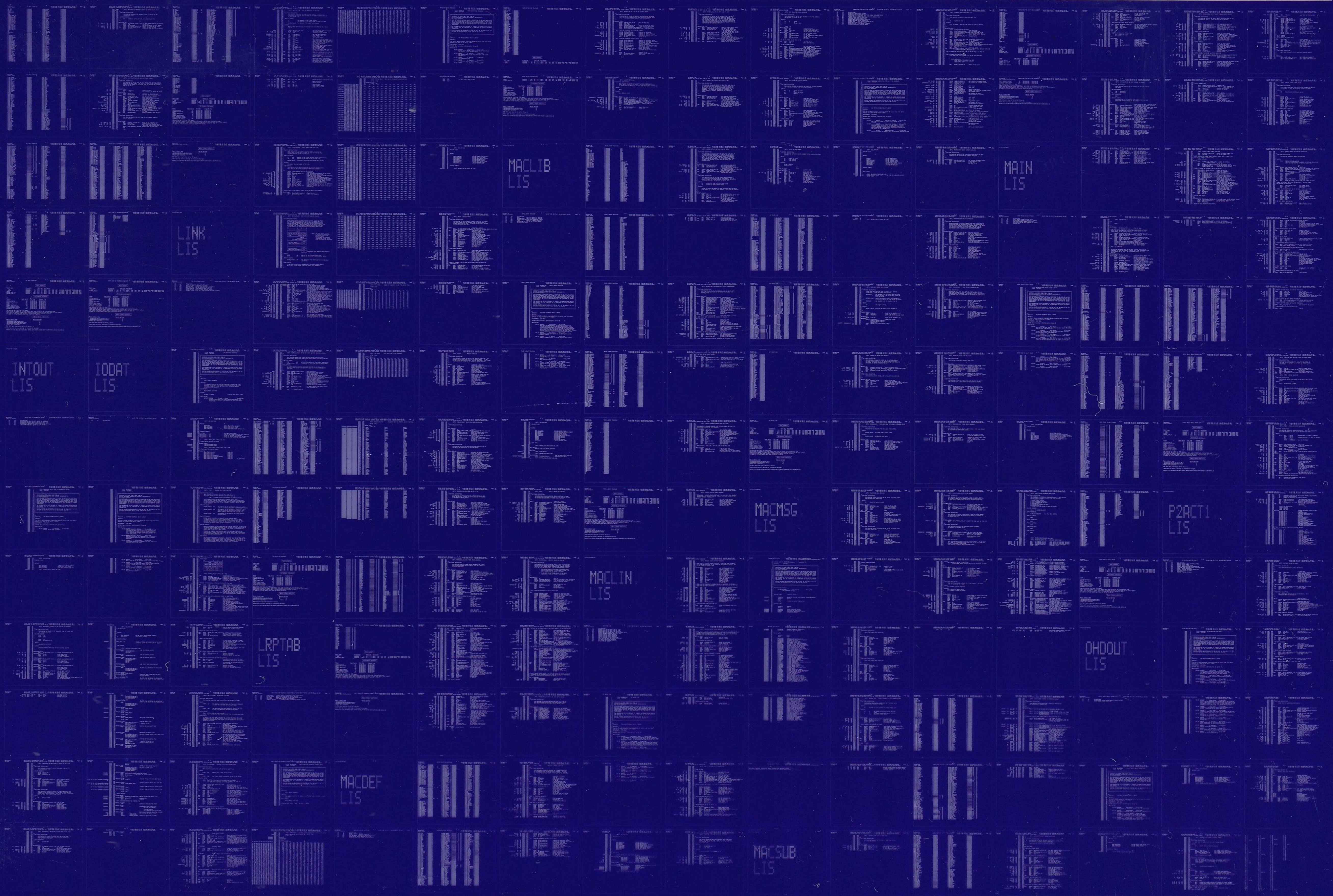
```
MAC_SUBSYS                    = 0000007D                PSC$M_PIC                     = 00000001
MLF$K_BLKSIZ                    00000177                PSC$M_QUAD                    = 00004C00
MLF$K_RSFNLN                  = 000000FF                PSC$M_RD                      = 00000080
MLF$L_CTINDEX                   00000014                PSC$M_REL                     = 00000008
MLF$L_MCDEF                     00000008                PSC$M_SHR                     = 00000020
MLF$L_QLINK                     00000000                PSC$M_USR                     = FFFFFFFD
MLF$Q_FNAMDS                    0000000C                PSC$M_VEC                     = 00000200
MLF$T_FNAM                      00000078                PSC$M_WORD                    = 00004400
MLF$X_NAMBLK                    00000018                PSC$M_WRT                     = 00000180
MNB$B_ARGCT                     00000017                PSC$S_ALIGNMENT               = 00000004
MNB$B_NAME                      00000004                PSC$V_ALIGNFLG                = 0000000E
MNB$K_BLKSIZ                    0000001C                PSC$V_ALIGNMENT               = 0000000A
MNB$L_ARGP                      00000018                PSC$V_EXE                     = 00000006
MNB$L_CRSYMF                    00000013                PSC$V_GBL                     = 00000004
MNB$L_LINK                      00000000                PSC$V_LIB                     = 00000001
MNB$L_PAGC                      0000000F                PSC$V_OVR                     = 00000002
MNB$L_PAGP                      0000000B                PSC$V_PIC                     = 0000000C
MNB$L_TXTP                      00000005                PSC$V_RD                      = 00000007
MNB$W_FLAG                      00000009                PSC$V_REL                     = 00000003
MXB$K_BLKSIZ                    00000008                PSC$V_SHR                     = 00000005
MXB$L_LINK                      00000000                PSC$V_VEC                     = 00000009
MXB$L_PAGES                     00000004                PSC$V_WRT                     = 00000008
NAM$C_BLN                     = 00000060                PSC$W_FLAG                      00000009
NAM$C_MAXRSS                  = 000000FF                PSC$W_OPTIONS                   0000000D
NO_MEM                          0000041F R      04      RAB$L_FAB                     = 0000003C
OBJ$K_BUFSIZ                  = 00000200                RAB$L_RHB                     = 0000002C
OPF$M_LASTOPR                 = 00002000                RDX$V_BINARY                  = 00000000
OPF$M_OPTEXP                  = 00001000                RDX$V_DECIMAL                 = 00000002
OPF$V_LASTOPR                 = 0000000D                RDX$V_DOUBLE                  = 00000005
OPF$V_OPTEXP                  = 0000000C                RDX$V_FLOAT                   = 00000004
PSC$B_NAME                      00000004                RDX$V_GFLOAT                  = 00000006
PSC$B_SEG                       0000000C                RDX$V_HEX                     = 00000003
PSC$B_UNUSED                    0000000B                RDX$V_HFLOAT                  = 00000007
PSC$K_BLKSIZ                    00000013                RDX$V_OCTAL                   = 00000001
PSC$K_NO_OPTNS                = 0000000A                REG$_PC                       = 0000000F
PSC$L_CURLOC                    0000000F                SEMI                          = 0000003B
PSC$L_LINK                      00000000                STB$K_PG_MISS                 = 0000000A
PSC$L_MAXLGTH                   00000005                SUM$INIT_EDIT                   ********  X    04
PSC$M_ABS                     = FFFFFFF7                SYM$B_NAME                      00000004
PSC$M_ALIGNFLG                = 00004000                SYM$B_SEG                       0000000C
PSC$M_ALLOPTNS                = 000003FF                SYM$B_TOKEN                     0000000B
PSC$M_BYTE                    = 00004000                SYM$K_BLKSIZ                    0000000D
PSC$M_CON                     = FFFFFFFB                SYM$K_MAXLEN                  = 0000001F
PSC$M_DEFAULT                 = 000001C8                SYM$K_TWOCOL                  = 00000010
PSC$M_EXE                     = 000000C0                SYM$L_LINK                      00000000
PSC$M_GBL                     = 00000010                SYM$L_VAL                       00000005
PSC$M_LCL                     = FFFFFFEF                SYM$M_ABS                     = 00000010
PSC$M_LIB                     = 00000002                SYM$M_ASN                     = 00000100
PSC$M_LONG                    = 00004800                SYM$M_CRFO                    = 00002000
PSC$M_NOEXE                   = FFFFFFBF                SYM$M_DEBUG                   = 00000020
PSC$M_NOPIC                   = FFFFFFFE                SYM$M_DEF                     = 00000001
PSC$M_NORD                    = FFFFFF7F                SYM$M_DELMAC                  = 00000200
PSC$M_NOSHR                   = FFFFFFDF                SYM$M_EPT                     = 00000200
PSC$M_NOVEC                   = FFFFFDFF                SYM$M_EXTRN                   = 00000008
PSC$M_NOWRT                   = FFFFFEFF                SYM$M_GLOBL                   = 00000004
PSC$M_OVR                     = 00000004                SYM$M_LOCAL                   = 00000040
PSC$M_PAGE                    = 00006400                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$ASCTIM                         ********  GX    04
SYS$CLOSE                          ********  GX    04
SYS$CONNECT                        ********  GX    04
SYS$DISCONNECT                     ********  GX    04
SYS$FAOL                           ********  GX    04
SYS$OPEN                           ********  GX    04
TAB                              = 00000009
X1                               = 00000400
X2                               = 0000000F
XAB$Q_CDT                          ********   X    04

                                              +-----------------+
                                              ! 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
$ABS$                            00000177  (   375.)  02 (  2.)  NOPIC    USR    CON    ABS    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
MAC$TEMP_STOR                    00000100  (   256.)  03 (  3.)  NOPIC    USR    CON    REL    GBL NOSHR   EXE   RD     WRT NOVEC LONG
MAC$RO_CODE_P15                  00000481  (  1153.)  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    00:00:01.32
Command processing                105    00:00:00.39    00:00:04.87
Pass 1                            294    00:00:06.35    00:00:38.21
Symbol table sort                   0    00:00:00.87    00:00:06.03
Pass 2                            149    00:00:01.70    00:00:08.82
Symbol table output                44    00:00:00.21    00:00:00.79
```

```
Psect synopsis output           2     00:00:00.02     00:00:00.02
Cross-reference output          0     00:00:00.00     00:00:00.00
Assembler run totals          625     00:00:09.58     00:01:00.06

The working set limit was 1500 pages.
55860 bytes (110 pages) of virtual memory were used to buffer the intermediate code.
There were 50 pages of symbol table space allocated to hold 898 non-local and 50 local symbols.
791 source lines were read in Pass 1, producing 24 object records in Pass 2.
23 pages of virtual memory were used to define 21 macros.

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

Macro library name                                Macros defined
------------------                                --------------
_$255$DUA28:[MACRO.OBJ]MACRO.MLB;1                       8
_$255$DUA28:[SYSLIB]STARLET.MLB;2                       14
TOTALS (all libraries)                                  22

990 GETS were required to define 22 macros.

There were no errors, warnings or information messages.

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

0226 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



MACLIB
LIS

MAIN
LIS

LINK
LIS

INTOUT
LIS

IODAT
LIS

MACMSG
LIS

P2ACT1
LIS

MACLIN
LIS

LRPTAB
LIS

OHDOUT
LIS

MACDEF
LIS

MACSUB
LIS