```
MMM          MMM     AAAAAAAAA         CCCCCCCCCCC   RRRRRRRRRRR        OOOOOOOOO
MMM          MMM     AAAAAAAAA         CCCCCCCCCCC   RRRRRRRRRRR        OOOOOOOOO
MMM          MMM     AAAAAAAAA         CCCCCCCCCCC   RRRRRRRRRRR        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            RRRRRRRRRRR       OOO         OOO
MMM         MMM    AAA         AAA   CCC            RRRRRRRRRRR       OOO         OOO
MMM         MMM    AAA         AAA   CCC            RRRRRRRRRRR       OOO         OOO
MMM         MMM    AAAAAAAAAAAAAAA   CCC            RRR   RRR         OOO         OOO
MMM         MMM    AAAAAAAAAAAAAAA   CCC            RRR   RRR         OOO         OOO
MMM         MMM    AAAAAAAAAAAAAAA   CCC            RRR   RRR         OOO         OOO
MMM         MMM    AAA         AAA   CCC            RRR      RRR      OOO         OOO
MMM         MMM    AAA         AAA   CCC            RRR      RRR      OOO         OOO
MMM         MMM    AAA         AAA   CCC            RRR      RRR      OOO         OOO
MMM         MMM    AAA         AAA      CCCCCCCCCCC   RRR         RRR      OOOOOOOOO
MMM         MMM    AAA         AAA      CCCCCCCCCCC   RRR         RRR      OOOOOOOOO
MMM         MMM    AAA         AAA      CCCCCCCCCCC   RRR         RRR      OOOOOOOOO
```

FILEID**GETARG

```
  GGGGGGGG  EEEEEEEEEE  TTTTTTTTTT    AAAAAA    RRRRRRRR      GGGGGGGG
  GGGGGGGG  EEEEEEEEEE  TTTTTTTTTT    AAAAAA    RRRRRRRR      GGGGGGGG
GG          EE              TT      AA    AA  RR      RR  GG
GG          EE              TT      AA    AA  RR      RR  GG
GG          EE              TT      AA    AA  RR      RR  GG
GG          EE              TT      AA    AA  RR      RR  GG
GG          EEEEEEEE        TT      AA    AA  RRRRRRRR    GG
GG          EEEEEEEE        TT      AA    AA  RRRRRRRR    GG
GG  GGGGGG  EE              TT      AAAAAAAAAA  RR  RR      GG  GGGGGG
GG  GGGGGG  EE              TT      AAAAAAAAAA  RR  RR      GG  GGGGGG
GG      GG  EE              TT      AA    AA  RR    RR    GG      GG      ....
GG      GG  EE              TT      AA    AA  RR    RR    GG      GG      ....
  GGGGGG    EEEEEEEEEE      TT      AA    AA  RR      RR    GGGGGG        ....
  GGGGGG    EEEEEEEEEE      TT      AA    AA  RR      RR    GGGGGG        ....

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

Table of contents

```
0000     1          .TITLE  MAC$GETARG      GET MACRO FORMAL ARGUMENTS
0000     2          .IDENT  'V04-000'
0000     3
0000     4  ;
0000     5  ;*****************************************************************************
0000     6  ;*                                                                           *
0000     7  ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     8  ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     9  ;*  ALL RIGHTS RESERVED.                                                     *
0000    10  ;*                                                                           *
0000    11  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
0000    12  ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
0000    13  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
0000    14  ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
0000    15  ;*  OTHER PERSON.  NO TITL  TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
0000    16  ;*  TRANSFERRED.                                                             *
0000    17  ;*                                                                           *
0000    18  ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
0000    19  ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
0000    20  ;*  CORPORATION.                                                             *
0000    21  ;*                                                                           *
0000    22  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
0000    23  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    24  ;*                                                                           *
0000    25  ;*                                                                           *
0000    26  ;*****************************************************************************
0000    27  ;
0000    28
0000    29  ;++
0000    30  ; FACILITY:     VAX MACRO ASSEMBLER OBJECT LIBRARY
0000    31  ;
0000    32  ; ABSTRACT:
0000    33  ;
0000    34  ; The VAX-11 MACRO assembler translates MACRO-32 source code into object
0000    35  ; modules for input to the VAX-11 LINKER.
0000    36  ;
0000    37  ; ENVIRONMENT:  USER MODE
0000    38  ;
0000    39  ; AUTHOR: Benn Schreiber, CREATION DATE: 30-AUG-78
0000    40  ;
0000    41  ; MODIFIED BY:
0000    42  ;
0000    43  ;       V03-002 MCN0163         Maria del C. Nasr       04-Apr-1984
0000    44  ;               The error TOOMNYARG should have an error severity, not
0000    45  ;               warning.
0000    46  ;
0000    47  ;       V03.01  MTR0019         Mike Rhodes      7-Jul-1982
0000    48  ;               Fix argument detection for macros with no formal arguments
0000    49  ;               which are invoked with >1 actual arguments.
0000    50  ;
0000    51  ;       V02.06  CNH0047         Chris Hume      22-Dec-1980
0000    52  ;               Count null argument after trailing comma for .NARG directive.
0000    53  ;               (ARGSCN 02.08, DEFINE.MAR 02.18)
0000    54  ;
0000    55  ;       V01.05  RN0023          R. Newland       3-Nov-1979
0000    56  ;               New message codes to get error messages from system
0000    57  ;               message file.
```

```
0000     58 ;
0000     59 ;       V01.04  RN0010          R. Newland      5-Sep-1979
0000     60 ;               Multi-page MXB blocks
0000     61 ;
0000     62 ;--
```

```
                        .SBTTL  DECLARATIONS
                ;
                ; INCLUDE FILES:
                ;

                ;
                ; MACROS:
                ;

                        $MAC_CTLFLGDEF                          ;DEFINE CONTROL FLAGS
                        $MAC_INTCODDEF                          ;DEFINE INTERMEDIATE CODES
                        $MAC_GENVALDEF                          ;DEFINE GENERAL VALUES
                        $MAC_SYMBLKDEF                          ;DEFINE SYMBOL BLOCK OFFSETS
                        $MAC_MNBDEF                             ;DEFINE MNB, MAB
                        $MAC_INPBLKDEF                          ;DEFINE INPUT BLOCK OFFSETS
                        $MACMSGDEF                      ; Define message codes

00000000                .PSECT  MAC$RO_CODE_MAC,NOWRT,GBL,LONG
```

```
                     0000     83          .SBTTL  MAC$GET_ARGS    SCAN REAL MACRO ARGUMENTS
                     0000     84
                     0000     85 ;++
                     0000     86 ; FUNCTIONAL DESCRIPTION:
                     0000     87 ;
                     0000     88 ;       THIS ROUTINE SCANS THE REAL MACRO ARGUMENTS ON A MACRO
                     0000     89 ;       INVOCATION.  AN INPUT BLOCK IS CREATED TO READ THE MACRO
                     0000     90 ;       TEXT WITH THE REAL ARGUMENTS ATTACHED TO THIS BLOCK.
                     0000     91 ;
                     0000     92 ;--
                     0000     93
                     0000     94 MAC$GET_ARGS::
     1180 8F   BB    0000     95          PUSHR   #^M<R7,R8,R12>          ;SAVE REGISTERS
        FFF9'  30    0004     96          BSBW    MAC$ALL_1_PAGE          ;ALLOCATE PAGE FOR INPUT BLOCK
       58   50   D0  0007     97          MOVL    R0,R8                   ;POINT R8 TO INPUT BLOCK
0000'CF     58   D0  000A     98          MOVL    R8,W^MAC$GL_BLKPTR      ;SAVE INPUT BLOCK ADDRESS
88     0000'CF   D0  000F     99          MOVL    W^MAC$GL_INPUTP,(R8)+   ;SET LINK
  88     05 A6   D0  0014    100          MOVL    MNB$L_TXTP(R6),(R8)+    ;SET POINTER TO FIRST LINE OF MACRO TEXT
88     0000'CF   9E  0018    101          MOVAB   W^MAC$GET_MAC_LIN,(R8)+ ;SET ROUTINE ADDRESS TO GET NEXT LINE
50     0000'CF   9E  001D    102          MOVAB   W^MAC$GL_IF_LEVEL,R0    ;POINT TO IF LEVEL/VALUE LONGWORDS
       88   60   D0  0022    103          MOVL    (R0),(R8)+              ;SAVE OLD IF LEVEL
            80   D4  0025    104          CLRL    (R0)+                   ;SET NEW IF LEVEL
       88   60   D0  0027    105          MOVL    (R0),(R8)+              ;SAVE OLD IF VALUES
            60   D4  002A    106          CLRL    (R0)                    ;CLEAR NEW IF VALUES
       88   56   D0  002C    107          MOVL    R6,(R8)+                ;STORE POINTER TO MACRO MNB IN
                    002F    108                                          ;THE RPTCNT WORD
            88   D4  002F    109          CLRL    (R8)+                   ;CLEAR PAGE POINTER
  50    17 A6   9A  0031    110          MOVZBL  MNB$B_ARGCT(R6),R0      ;GET ARGUMENT COUNT
0000'CF     50   D0  0035    111          MOVL    R0,W^MAC$GL_MC_ARGCT    ;SET MACRO ARGUMENT COUNT
       88   50   90  003A    112          MOVB    R0,(R8)+                ;STORE ARG COUNT IN INPUT BLOCK
            1B   12  003D    113          BNEQ    10$                     ;IF NEQ GO PROCESS ARGS
        FFBE'   30  003F    114          BSBW    MAC$SKIPSP              ;ELSE CHECK FOR EOL
       0D   5A   91  0042    115          CMPB    R10, #CR                ;ANY ACTUAL ARGS PRESENT?
            10   13  0045    116          BEQL    5$                      ;NO -- VALID INVOCATION.
                    0047    117          $INTOUT_LW      INT$_ERR,<#MAC$_TOOMNYARGS, W^MAC$GL_LINEPT> ; YES -- WARN I
           0155  31  0057    118 5$:      BRW     GET_ARG_EXIT            ;ELSE WE ARE ALL DONE
```

```
                                005A   120 ;
                                005A   121 ; ZERO ARGUMENT POINTERS, THEN SCAN ARGS AND FILL IN POINTERS
                                005A   122 ;
          0000'CF    58    D0   005A   123 10$:    MOVL    R8,W^MAC$GL_ARGPTR      ;SAVE POINTER TO ARG POINTERS
                     88    D4   005F   124 20$:    CLRL    (R8)+                   ;ZERO ARG POINTER
                  FB 50    F5   0061   125         SOBGTR  R0,20$                  ;ZERO THEM ALL
                     FF99'  30   0064   126         BSBW    MAC$SKIPSP              ;SKIP LEADING SPACES
       57   58    0000'CF  C3   0067   127         SUBL3   W^MAC$GL_BLKPTR,R8,R7   ;FIGURE SPACE LEFT ON PAGE
  57   00000200  8F    57   C3   006D   128         SUBL3   R7,#512,R7              ;NOW WE HAVE IT
             00 6B    2D   E5   0075   129         BBCC    #FLG$V_MOREARG,(R11),.+1; Clear trailing comma flag.
             00 6B    05   E5   0079   130         BBCC    #FLG$V_ENDMCH,(R11),30$ ;CLEAR DONE FLAG
                               007D   131 ;
                               007D   132 ; HERE FOR EACH NEW ARGUMENT
                               007D   133 ;
                0D   5A    91   007D   134 30$:    CMPB    R10,#CR                 ;END OF LINE?
                     1C    12   0080   135         BNEQ    40$                     ;IF NEQ NO
             18 6B   05    E2   0082   136         BBSS    #FLG$V_ENDMCH,(R11),40$ ;YES--HAVE WE BEEN HERE BEFORE?
                                0086   137                                         ;(AND SET FLAG)
             50    17 A6   9A   0086   138         MOVZBL  MNB$B_ARGCT(R6),R0      ;GET # OF ARGS IN  MACRO
          50    0000'CF    C2   008A   139         SUBL2   W^MAC$GL_MC_ARGCT,R0    ;COUNT OF ARGS
          51    0000'CF    D0   008F   140         MOVL    W^MAC$GL_BLKPTR,R1      ;POINT TO INPUT BLOCK
             02 6B    2D   E1   0094   141         BBC     #FLG$V_MOREARG,(R11),35$; Was one more argument expected?
                      50   D6   0098   142         INCL    R0                      ; Bump count for .NARG if so.
             1C A1    50   90   009A   143 35$:    MOVB    R0,INP$B_ARGCT(R1)      ;SET COUNT INTO BLOCK
                  FF5F'  30   009E   144 40$:    BSBW    MAC$MAC_ARG_SCN         ;SCAN MACRO ARGUMENT
                      50   D5   00A1   145         TSTL    R0                      ;NULL ARG?
                      5E   13   00A3   146         BEQL    90$                     ;IF EQL YES
                   57   02   C2   00A5   147         SUBL2   #2,R7                   ;COUNT THE COUNT WORD ALSO
                   57   50   C2   00A8   148         SUBL2   R0,R7                   ;NO--IS THERE ROOM TO STORE IT?
                      12   14   00AB   149         BGTR    50$                     ;IF GTR YES
                      50   DD   00AD   150         PUSHL   R0                      ;NO--SAVE SIZE OF ARG
          51   50    0A   C1   00AF   151         ADDL3   #<MXB$K_BLKSIZ+2>,R0,R1 ; Compute total size of block required
                 012E   30   00B3   152         BSBW    ALL_INP_PAGE            ;ALLOCATE A PAGE
                   57   02   C2   00B6   153         SUBL2   #2,R7                   ; Sub size of count word from new count
                      50 8ED0   00B9   154         POPL    R0                      ;GET SIZE OF MACRO ARG BACK
                   57   50   C2   00BC   155         SUBL2   R0,R7                   ;UDPATE FREE SPACE
          51   0000'CF    D0   00BF   156 50$:    MOVL    W^MAC$GL_KEYPTR,R1      ;GET POINTER TO POSSIBLE KEYWORD MAB
                      05   13   00C4   157         BEQL    60$                     ;IF EQL NOT KEYWORD ARGUMENT
                    0107   30   00C6   158         BSBW    SET_ARG_PKW             ;SET ARG POINTER FOR KEYWORD
                      03   11   00C9   159         BRB     70$                     ;GO COPY ARG TO VM
                    00EB   30   00CB   160 60$:    BSBW    SET_ARG_PTR             ;SET ARG PTR FOR THIS ARG
                88    50   B0   00CE   161 70$:    MOVW    R0,(R8)+                ;STORE LINE LENGTH
  68   0000'CF    50   28   00D1   162         MOVC3   R0,W^MAC$AB_TMPBUF,(R8) ;COPY ARG INTO VM
                58   53   D0   00D7   163         MOVL    R3,R8                   ;UPDATE POINTER
                0000'CF    D5   00DA   164 80$:    TSTL    W^MAC$GL_KEYPTR         ;DON'T COUNT KEYWORD ARGS
                      9D   12   00DE   165         BNEQ    30$                     ;IF NEQ KEYWORD ARG
             98 0000'CF    F5   00E0   166         SOBGTR  W^MAC$GL_MC_ARGCT,30$   ;LOOP FOR ALL ARGS
                  FF18'  30   00E5   167         BSBW    MAC$SKIPSP              ;SKIP ANY SPACES LEFT
             05 6B    2D   E0   00E8   168         BBS     #FLG$V_MOREARG,(R11),85$; Complain if trailing comma.
                0D   5A    91   00EC   169         CMPB    R10,#CR                 ;DONE--DID WE GET TO EOL?
                      5F   13   00EF   170         BEQL    FILL_DEF_ARGS           ;YES--GO FILL DEFAULT ARGS
                                00F1   171 85$:    $INTOUT_LW INTS_ERR,<#MAC$_TOOMNYARGS,W^MAC$GL_LINEPT> ; No--warn in pass 2
                      4D   11   0101   172         BRB     FILL_DEF_ARGS           ;GO FILL DEFAULT ARGS
                                0103   173                                         ; (STAT4 FORCES EOL TO READ MACRO
                                0103   174                                         ;  SO WE DON'T NEED TO FIND IT HERE)
```

```
                     0103   176 ;
                     0103   177 ; THE ARGUMENT WAS NULL. IF THIS IS A CREATED SYMBOL, CREATE THE
                     0103   178 ; SYMBOL NOW.
                     0103   179 ;
   50   17 A6   9A   0103   180 90$:    MOVZBL  MNB$B_ARGCT(R6),R0      ;GET TOTAL # ARGS
 50   0000'CF   C2   0107   181         SUBL2   W^MAC$GL_MC_ARGCT,R0    ;FIGURE ARGUMENT # -1
      1F   50   D1   010C   182         CMPL    R0,#31                  ;ONLY FIRST 32 SYMBOLS ARE CREATED
           C9   14   010F   183         BGTR    80$                     ;IF GTR THEN JUST SKIP IT
C4 13 A6   50   E1   0111   184         BBC     R0,MNB$L_CRSYMF(R6),80$ ;BRANCH IF NOT CREATED SYMBOL
 50   57   08   C3   0116   185         SUBL3   #8,R7,R0                ;IT IS--IS THERE ROOM FOR 5 DIGITS,
                     011A   186                                         ;DOLLAR SIGN, AND CHAR COUNT?
           06   14   011A   187         BGTR    100$                    ;IF GTR YES
      51   08   D0   011C   188         MOVL    #8,R1                   ; 8 bytes required
         00C2   30   011F   189         BSBW    ALL_INP_PAGE            ;NO--ALLOCATE A PAGE
 50   0000'CF   D0   0122   190 100$:   MOVL    W^MAC$GL_CRSYM,R0       ;GET NEXT CREATED SYMBOL NUMBER
      0000'CF   D6   0127   191         INCL    W^MAC$GL_CRSYM          ;INC TO NEXT CREATED SYMBOL
           58   DD   012B   192         PUSHL   R8                      ;REMEMBER WHERE COUNT WORD IS
           88   B4   012D   193         CLRW    (R8)+                   ;RESERVE SPACE FOR COUNT WORD
      51   58   D0   012F   194         MOVL    R8,R1                   ;R1 HAS OUTPUT POINTER
        FECB'   30   0132   195         BSBW    MAC$DEC_OUT_L2X         ;OUTPUT NUMBER IN DECIMAL
      80   24   90   0135   196         MOVB    #^A/$/,(R0)+            ;FINISH SYMBOL WITH '$'
51    50   58   C3   0138   197         SUBL3   R8,R0,R1                ;FIGURE LENGTH OF NUMBER
           58 8ED0   013C   198         POPL    R8                      ;RESTORE PTR TO COUNT WORD
      68   51   B0   013F   199         MOVW    R1,(R8)                 ;STORE CHARACTER COUNT
         0074   30   0142   200         BSBW    SET_ARG_PTR             ;STORE ARG POINTER IN INPUT BLOCK
      58   50   D0   0145   201         MOVL    R0,R8                   ;UPDATE POINTER
      57   51   C2   0148   202         SUBL2   R1,R7                   ;UPDATE COUNTER
      57   02   C2   014B   203         SUBL2   #2,R7                   ;COUNT THE COUNT WORD ALSO
           8A   11   014E   204         BRB     80$                     ;CONTINUE
```

```
                        0150   206 ;
                        0150   207 ; NOW TRY TO FILL IN NULL ARGS WITH KEYWORD DEFAULT VALUES
                        0150   208 ;
                        0150   209
                        0150   210 FILL_DEF_ARGS:
0000'CF    01    9A     0150   211          MOVZBL  #1,W^MAC$GL_MC_ARGCT     ;FIRST ARGUMENT
5C    0000'CF    D0     0155   212          MOVL    W^MAC$GL_ARGPTR,R12      ;GET POINTER TO ARG POINTERS
  7E    17 A6    9A     015A   213          MOVZBL  MNB$B_ARGCT(R6),-(SP)    ;STACK # OF ARGS
           4D    13     015E   214          BEQL    60$                      ;IF EQL NOTHING TO DO
        18 A6    D5     0160   215          TSTL    MNB$L_ARGP(R6)           ;ARE THERE ANY ARGS?
           48    13     0163   216          BEQL    60$                      ;IF EQL NOTHING TO DEFAULT WITH
           8C    D5     0165   217 10$:     TSTL    (R12)+                   ;THIS ARG NULL?
           3D    12     0167   218          BNEQ    50$                      ;IF NEQ NO
     55    18 A6    D0     0169   219          MOVL    MNB$L_ARGP(R6),R5        ;YES--POINT TO ARGS
50    0000'CF    D0     016D   220          MOVL    W^MAC$GL_MC_ARGCT,R0     ;GET CURRENT ARG NUMBER
  05 A5    50    91     0172   221 20$:     CMPB    R0,MAB$B_ARGNO(R5)       ;IS THIS THE RIGHT ARG?
           07    13     0176   222          BEQL    30$                      ;IF EQL YES
        55    65    D0     0178   223          MOVL    (R5),R5                  ;NO--LINK TO NEXT MAB
           F5    12     017B   224          BNEQ    20$                      ;IF NEQ KEEP LOOKING
           27    11     017D   225          BRB     50$                      ;???--NOT THERE???--JUST SKIP IT
                        017F   226 ;
                        017F   227 ; FOUND KEYWORD MAB. IF THERE IS A DEFAULT VALUE, COPY IT INTO THE
                        017F   228 ; INPUT BLOCK
                        017F   229 ;
  54    06 A5    3C     017F   230 30$:     MOVZWL  MAB$W_DVLEN(R5),R4       ;GET LENGTH OF DEFAULT STRING
           21    13     0183   231          BEQL    50$                      ;IF EQL THERE IS NONE
        57    02    C2     0185   232          SUBL2   #2,R7                    ;LEAVE ROOM FOR COUNT WORD
        57    54    C2     0188   233          SUBL2   R4,R7                    ;SEE IF ROOM IN BLOCK
           0A    14     018B   234          BGTR    40$                      ;IF GTR YES
  51    54    0A    C1     018D   235          ADDL3   #<MXB$K_BLKSIZ+2>,R4,R1 ; Compute total size of block required
              0050  30     0191   236          BSBW    ALL_INP_PAGE             ;NO--GET A PAGE
        57    02    C2     0194   237          SUBL2   #2,R7                    ; Sub size of count word from new count
     FC AC    58    D0     0197   238 40$:     MOVL    R8,-4(R12)               ;STORE POINTER TO STRING IN ARG POINTERS
           88    54    B0     019B   239          MOVW    R4,(R8)+                 ;STORE CHARACTER COUNT
68    08 B5    54    28     019E   240          MOVC3   R4,@MAB$L_DVPTR(R5),(R8) ;COPY THE DEFAULT STRING
           58    53    D0     01A3   241          MOVL    R3,R8                    ;UPDATE FREE POINTER
           0000'CF    D6     01A6   242 50$:     INCL    W^MAC$GL_MC_ARGCT        ;NEXT ARGUMENT
             B8 6E    F5     01AA   243          SOBGTR  (SP),10$                 ;CHECK ALL ARGS
           8E    D5     01AD   244 60$:     TSTL    (SP)+                    ;CLEAR EXHAUSTED COUNT FROM STACK
                        01AF   245
                        01AF   246 GET_ARG_EXIT:
50    0000'CF    D0     01AF   247          MOVL    W^MAC$GL_BLKPTR,R0       ;RETURN INPUT BLOCK ADDRESS IN R0
      1180 8F    BA     01B4   248          POPR    #^M<R7,R8,R12>           ;RESTORE REGISTERS
                  05     01B8   249          RSB
```

```
                     01B9   251 ;++
                     01B9   252 ; FUNCTIONAL DESCRIPTION:
                     01B9   253 ;
                     01B9   254 ;       THIS ROUTINE UPDATES THE ARGUMENT POINTER POINTER IN THE INPUT
                     01B9   255 ;       BLOCK.
                     01B9   256 ;
                     01B9   257 ;--
                     01B9   258
                     01B9   259 SET_ARG_PTR:
          03     BB  01B9   260         PUSHR   #^M<R0,R1>                  ;SAVE REGISTERS
50   0000'CF     D0  01BB   261         MOVL    W^MAC$GL_ARGPTR,R0          ;GET POINTER TO ARGUMENT POINTERS
  51     17 A6   9A  01C0   262         MOVZBL  MNB$B_ARGCT(R6),R1          ;GET TOTAL # ARGS
51   0000'CF     C2  01C4   263         SUBL2   W^MAC$GL_MC_ARGCT,R1        ;FIGURE INDEX FOR THIS ARG
   6041     58     D0  01C9   264         MOVL    R8,(R0)[R1]                 ;STORE POINTER TO ARG
          03     BA  01CD   265         POPR    #^M<R0,R1>                  ;RESTORE REGISTERS
          05     01CF   266         RSB
                     01D0   267
                     01D0   268 ;++
                     01D0   269 ; FUNCTIONAL DESCRIPTION:
                     01D0   270 ;
                     01D0   271 ;       THIS ROUTINE UPDATES THE ARGUMENT POINTER POINTER FOR THE
                     01D0   272 ;       GIVEN KEYWORD.
                     01D0   273 ;
                     01D0   274 ;--
                     01D0   275
                     01D0   276 SET_ARG_PKW:
          50     DD  01D0   277         PUSHL   R0                          ;SAVE R0
50   0000'CF     D0  01D2   278         MOVL    W^MAC$GL_ARGPTR,R0          ;POINT TO ARGUMENT POINTERS
  51     05 A1   9A  01D7   279         MOVZBL  MAB$B_ARGNO(R1),R1          ;GET KEYWORD ARGUMENT NUMBER
FC A041     58     D0  01DB   280         MOVL    R8,-4(R0)[R1]               ;STORE POINTER TO ARG
          50 8ED0   01E0   281         POPL    R0                          ;RESTORE R0
          05     01E3   282         RSB
                     01E4   283
                     01E4   284 ;++
                     01E4   285 ; FUNCTIONAL DESCRIPTION:
                     01E4   286 ;
                     01E4   287 ;       THIS ROUTINE ALLOCATES A PAGE OF VM AND LINKS IT INTO THE
                     01E4   288 ;       PAGE LIST FOR THE CURRENT INPUT BLOCK (POINTED TO BY MAC$GL_BLKPTR).
                     01E4   289 ;
                     01E4   290 ; Inputs:
                     01E4   291 ;
                     01E4   292 ;       R1 = Number of bytes required
                     01E4   293 ;
                     01E4   294 ; Outputs:
                     01E4   295 ;
                     01E4   296 ;       R7 = Number of free bytes in block
                     01E4   297 ;       R8 = Pointer to first free byte
                     01E4   298 ;
                     01E4   299 ;--
                     01E4   300
                     01E4   301 ALL_INP_PAGE:
      FE19'     30   01E4   302         BSBW    MAC$ALL_BLOCK               ; Allocate block
        51     DD   01E7   303         PUSHL   R1                          ; Save number of pages allocated
57   51     09   78  01E9   304         ASHL    #9,R1,R7                    ; Convert size to bytes
     57     08   C2  01ED   305         SUBL2   #MXB$K_BLKSIZ,R7            ; and compute number of free bytes
     58     50   D0  01F0   306         MOVL    R0,R8                       ;SET R8 POINTING TO NEW PAGE
51   0000'CF     D0  01F3   307         MOVL    W^MAC$GL_BLKPTR,R1          ;GET POINTER TO INPUT BLOCK
```

```
                     01F8   308          ASSUME  MXB$L_LINK  EQ 0
                     01F8   309          ASSUME  MXB$L_PAGES EQ MXB$L_LINK+4
88   18 A1   D0     01F8   310          MOVL    INP$L_PAGP(R1),(R8)+    ;LINK NEW PAGE INTO PAGE LIST
18 A1    50   D0     01FC   311          MOVL    R0,INP$L_PAGP(R1)       ;...
   88    8E   D0     0200   312          MOVL    (SP)+,(R8)+             ; Store block size in block
               05     0203   313          RSB
                     0204   314
                     0204   315          .END
```

```
$COUNT              = 0000003B
ALL_INP_PAGE          000001E4 R      03
ARG$K_SIZE          = 000003E8
AUD$K_SIZE          = 00000010
BLNK                = 00000020
CHR$M_COMMA_CR      = 00000020
CHR$M_ILL_CHR       = 00000040
CHR$M_NUM_BER       = 00000010
CHR$M_SPA_MSK       = 00000001
CHR$M_SYM_CH1       = 00000008
CHR$M_SYM_CHR       = 00000004
CHR$M_SYM_DLM       = 00000002
CHR$V_COMMA_CR      = 00000005
CHR$V_CVTLWC        = 00000061
CHR$V_ILL_CHR       = 00000006
CHR$V_NOCVT         = 0000007F
CHR$V_NUM_BER       = 00000004
CHR$V_SPA_MSK       = 00000000
CHR$V_SYM_CH1       = 00000003
CHR$V_SYM_CHR       = 00000002
CHR$V_SYM_DLM       = 00000001
CNT                 = 00000002
CR                  = 0000000D
ERR                 = 00000001
FF                  = 0000000C
FILL_DEF_ARGS         00000150 R      03
FLG$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$V_XCRF          = 0000001F
GET_ARG_EXIT          000001AF R      03
HASASZ              = 0000007F
HYPHEN              = 0000002D
INP$B_ARGCT           0000001C
INP$K_BLKSIZ          00000021
INP$K_BUFSIZ        = 000003E8
INP$K_IRPSIZ          0000003C
INP$L_ARGS            0000001D
INP$L_GETL            00000008
INP$L_IFLVL           0000000C
INP$L_IFVAL           00000010
INP$L_LINK            00000000
INP$L_NXTL            00000004
INP$L_PAGP            00000018
INP$L_RPTCNT          00000014
INT$K_BUFSIZ        = 000013F4
INT$K_BUFWRN        = 00001390
INT$_ADD            = 00000001
INT$_AND            = 00000002
INT$_ASH            = 00000003
INT$_ASN            = 0000000C
INT$_AUGPC          = 0000000D
INT$_BDST           = 0000000E
INT$_CHKL           = 0000000F
INT$_DIV            = 00000004
INT$_END            = 00000010
INT$_EPT            = 00000011
INT$_ERR            = 00000012
INT$_ETX            = 00000013
INT$_FNEWL          = 00000014
INT$_ILG            = 00000000
INT$_INFO           = 0000003A
INT$_LGLAB          = 00000015
INT$_MACL           = 00000016
INT$_MUL            = 00000005
INT$_NEG            = 00000006
INT$_NEWL           = 00000017
INT$_NEWP           = 00000018
INT$_NOT            = 00000007
INT$_OP             = 00000019
INT$_OR             = 00000008
INT$_PRIL           = 0000001A
INT$_PRT            = 0000001B
```

Symbol table

```
INT$_PSECT      = 0000001C
INT$_REDEF      = 0000001D
INT$_REF        = 0000001E
INT$_REST       = 0000001F
INT$_SAME       = 00000009
INT$_SAVE       = 00000020
INT$_SBTTL      = 00000021
INT$_SETFLAG    = 00000022
INT$_SETLONG    = 00000023
INT$_SPIC       = 00000024
INT$_SPID       = 00000025
INT$_STIB       = 00000026
INT$_STIL       = 00000028
INT$_STIW       = 00000027
INT$_STKEPT     = 00000029
INT$_STKG       = 0000002A
INT$_STKL       = 0000002B
INT$_STKPC      = 0000002C
INT$_STKS       = 0000002D
INT$_STOB       = 00000034
INT$_STOL       = 0000002E
INT$_STOW       = 00000035
INT$_STRB       = 0000002F
INT$_STRL       = 00000031
INT$_STRSB      = 00000032
INT$_STRSW      = 00000033
INT$_STRW       = 00000030
INT$_STSB       = 00000036
INT$_STSW       = 00000037
INT$_SUB        = 0000000A
INT$_SUME       = 00000039
INT$_WRN        = 00000038
INT$_XOR        = 0000000B
LST$R_BUFSIZ    = 00000086
LST$K_L_P_PAGE  = 0000003C
LST$K_TITLE_SIZ= 00000028
MAB$B_ARGNO     00000005
MAB$B_NAME      00000004
MAB$K_BLKSIZ    0000000C
MAB$L_DVPTR     00000008
MAB$L_LINK      00000000
MAB$W_DVLEN     00000006
MAC$AB_TMPBUF   ********    X   03
MAC$ALL_1_PAGE  ********    X   03
MAC$ALL_BLOCK   ********    X   03
MAC$DEC_OUT_L2X ********    X   03
MAC$GET_ARGS    00000000 RG     03
MAC$GET_MAC_LIN ********    X   03
MAC$GL_ARGPTR   ********    X   03
MAC$GL_BLKPTR   ********    X   03
MAC$GL_CRSYM    ********    X   03
MAC$GL_IF_LEVEL ********    X   03
MAC$GL_INPUTP   ********    X   03
MAC$GL_KEYPTR   ********    X   03
MAC$GL_LINEPT   ********    X   03
MAC$GL_MC_ARGCT ********    X   03
MAC$INTERR_2_LW ********    X   03
MAC$MAC_ARG_SCN ********    X   03
MAC$SKIPSP      ********    X   03
MAC$_TOOMNYARGS= 007D91FA
MAC_SUBSYS      = 0000007D
MNB$B_ARGCT     00000017
MNB$B_NAME      00000004
MNB$K_BLKSIZ    0000001C
MNB$L_ARGP      00000018
MNB$L_CRSYMF    00000013
MNB$L_LINK      00000000
MNB$L_PAGC      0000000F
MNB$L_PAGP      0000000B
MNB$L_TXTP      00000005
MNB$W_FLAG      00000009
MXB$K_BLKSIZ    00000008
MXB$L_LINK      00000000
MXB$L_PAGES     00000004
OBJ$K_BUFSIZ    = 00000200
OPF$M_LASTOPR   = 00002000
OPF$M_OPTEXP    = 00001000
OPF$V_LASTOPR   = 0000000D
OPF$V_OPTEXP    = 0000000C
PSC$B_NAME      00000004
PSC$B_SEG       0000000C
PSC$B_UNUSED    0000000B
PSC$K_BLKSIZ    00000013
PSC$K_NO_OPTNS  = 0000000A
PSC$L_CURLOC    0000000F
PSC$L_LINK      00000000
PSC$L_MAXLGTH   00000005
PSC$M_ABS       = FFFFFFF7
PSC$M_ALIGNFLG  = 00004000
PSC$M_ALLOPTNS  = 000003FF
PSC$M_BYTE      = 00004000
PSC$M_CON       = FFFFFFFB
PSC$M_DEFAULT   = 000001C8
PSC$M_EXE       = 000000C0
PSC$M_GBL       = 00000010
PSC$M_LCL       = FFFFFFEF
PSC$M_LIB       = 00000002
PSC$M_LONG      = 00004800
PSC$M_NOEXE     = FFFFFFBF
PSC$M_NOPIC     = FFFFFFFE
PSC$M_NORD      = FFFFFF7F
PSC$M_NOSHR     = FFFFFFDF
PSC$M_NOVEC     = FFFFFDFF
PSC$M_NOWRT     = FFFFFEFF
PSC$M_OVR       = 00000004
PSC$M_PAGE      = 00006400
PSC$M_PIC       = 00000001
PSC$M_QUAD      = 00004C00
PSC$M_RD        = 00000080
PSC$M_REL       = 00000008
PSC$M_SHR       = 00000020
PSC$M_USR       = FFFFFFFD
PSC$M_VEC       = 00000200
PSC$M_WORD      = 00004400
PSC$M_WRT       = 00000180
PSC$S_ALIGNMENT= 00000004
PSC$V_ALIGNFLG  = 0000000E
PSC$V_ALIGNMENT= 0000000A
PSC$V_EXE       = 00000006
PSC$V_GBL       = 00000004
PSC$V_LIB       = 00000001
PSC$V_OVR       = 00000002
PSC$V_PIC       = 00000000
PSC$V_RD        = 00000007
PSC$V_REL       = 00000003
PSC$V_SHR       = 00000005
PSC$V_VEC       = 00000009
PSC$V_WRT       = 00000008
PSC$W_FLAG      00000009
PSC$W_OPTIONS   0000000D
RDX$V_BINARY    = 00000000
RDX$V_DECIMAL   = 00000002
RDX$V_DOUBLE    = 00000005
RDX$V_FLOAT     = 00000004
RDX$V_GFLOAT    = 00000006
RDX$V_HEX       = 00000003
RDX$V_HFLOAT    = 00000007
RDX$V_OCTAL     = 00000001
REG$_PC         = 0000000F
SEMI            = 0000003B
SET_ARG_PKW     000001D0 R      03
SET_ARG_PTR     000001B9 R      03
STB$K_PG_MISS   = 0000000A
SYM$B_NAME      00000004
SYM$B_SEG       0000000C
SYM$B_TOKEN     0000000B
SYM$K_BLKSIZ    0000000D
SYM$K_MAXLEN    = 0000001F
SYM$K_TWOCOL    = 00000010
SYM$L_LINK      00000000
SYM$L_VAL       00000005
SYM$M_ABS       = 00000010
SYM$M_ASN       = 00000100
SYM$M_CRFO      = 00002000
SYM$M_DEBUG     = 00000020
SYM$M_DEF       = 00000001
SYM$M_DELMAC    = 00000200
SYM$M_EPT       = 00000200
SYM$M_EXTRN     = 00000008
SYM$M_GLOBL     = 00000004
SYM$M_LOCAL     = 00000040
SYM$M_ODBG      = 00000400
SYM$M_REF       = 00000080
SYM$M_RELPSECT  = 00000800
SYM$M_SUPR      = 00004000
SYM$M_WEAK      = 00000002
SYM$M_XCRF      = 00001000
SYM$V_ABS       = 00000004
SYM$V_ASN       = 00000008
SYM$V_CRFO      = 0000000D
SYM$V_DEBUG     = 00000005
```

```
SYM$V_DEF       = 00000000
SYM$V_DELMAC    = 00000009
SYM$V_EPT       = 00000009
SYM$V_EXTRN     = 00000003
SYM$V_GLOBL     = 00000002
SYM$V_LOCAL     = 00000006
SYM$V_ODBG      = 0000000A
SYM$V_REF       = 00000007
SYM$V_RELPSECT  = 0000000B
SYM$V_SUPR      = 0000000E
SYM$V_WEAK      = 00000001
SYM$V_XCRF      = 0000000C
SYM$W_FLAG        00000009
TAB             = 00000009
X1              = 00000400
X2              = 0000000F
```

+----------------+
! Psect synopsis !
+----------------+

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
. BLANK .	00000000 (0.)	01 (1.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
ABS	0000003C (60.)	02 (2.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
MAC$RO_CODE_MAC	00000204 (516.)	03 (3.)	NOPIC USR CON REL GBL NOSHR EXE RD NOWRT NOVEC LONG

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

Phase	Page faults	CPU Time	Elapsed Time
Initialization	30	00:00:00.06	00:00:00.87
Command processing	138	00:00:00.42	00:00:04.80
Pass 1	201	00:00:03.13	00:00:14.89
Symbol table sort	0	00:00:00.43	00:00:01.84
Pass 2	73	00:00:00.77	00:00:03.50
Symbol table output	29	00:00:00.18	00:00:01.02
Psect synopsis output	2	00:00:00.01	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	475	00:00:05.02	00:00:26.95

The working set limit was 1200 pages.
28334 bytes (56 pages) of virtual memory were used to buffer the intermediate code.
There were 30 pages of symbol table space allocated to hold 437 non-local and 19 local symbols.
315 source lines were read in Pass 1, producing 16 object records in Pass 2.
13 pages of virtual memory were used to define 12 macros.

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

Macro library name                                Macros defined
------------------                                --------------
_$255$DUA28:[MACRO.OBJ]MACRO.MLB;1                        10
_$255$DUA28:[SYSLIB]STARLET.MLB;2                          4
TOTALS (all libraries)                                    14

563 GETS were required to define 14 macros.

There were no errors, warnings or information messages.

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

DATA
LIS

DEFINE
LIS

FLOAT
LIS

ERRMSG
LIS

GETARG
LIS

DATA1
LIS

INPUT
LIS

ERROR
LIS

FINISH
LIS

GETCMD
LIS