```
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
```

```
  AAAAAA      CCCCCCCC  TTTTTTTTTT  RRRRRRRR    EEEEEEEEEE  FFFFFFFFFF
  AAAAAA      CCCCCCCC  TTTTTTTTTT  RRRRRRRR    EEEEEEEEEE  FFFFFFFFFF
AA      AA  CC            TT      RR      RR  EE          FF
AA      AA  CC            TT      RR      RR  EE          FF
AA      AA  CC            TT      RR      RR  EE          FF
AA      AA  CC            TT      RR      RR  EE          FF
AA      AA  CC            TT      RRRRRRRR    EEEEEEEE    FFFFFFFF
AA      AA  CC            TT      RRRRRRRR    EEEEEEEE    FFFFFFFF
AAAAAAAAAA  CC            TT      RR  RR      EE          FF
AAAAAAAAAA  CC            TT      RR  RR      EE          FF
AA      AA  CC            TT      RR    RR    EE          FF              ....
AA      AA  CC            TT      RR    RR    EE          FF              ....
AA      AA    CCCCCCCC    TT      RR      RR  EEEEEEEEEE  FF              ....
AA      AA    CCCCCCCC    TT      RR      RR  EEEEEEEEEE  FF              ....

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$ACTREF OPERAND REFERENCES
0000     2          .IDENT  'V04-000'
0000     3
0000     4  ;
0000     5  ;*****************************************************************************
0000     6  ;*                                                                           *
0000     7  ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     8  ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     9  ;*  ALL RIGHTS RESERVED.                                                     *
0000    10  ;*                                                                           *
0000    11  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
0000    12  ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
0000    13  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
0000    14  ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
0000    15  ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
0000    16  ;*  TRANSFERRED.                                                             *
0000    17  ;*                                                                           *
0000    18  ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
0000    19  ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
0000    20  ;*  CORPORATION.                                                             *
0000    21  ;*                                                                           *
0000    22  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
0000    23  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    24  ;*                                                                           *
0000    25  ;*                                                                           *
0000    26  ;*****************************************************************************
0000    27  ;
0000    28
0000    29  ;++
0000    30  ; FACILITY:     VAX MACRO ASSEMBLER OBJECT LIBRARY
0000    31  ;
0000    32  ; ABSTRACT:
0000    33  ;
0000    34  ; The VAX-11 MACRO assembler translates MACRO-32 source code into object
0000    35  ; modules for input to the VAX-11 LINKER.
0000    36  ;
0000    37  ; ENVIRONMENT:  USER MODE
0000    38  ;
0000    39  ; AUTHOR: Benn Schreiber, CREATION DATE: 20-AUG-78
0000    40  ;
0000    41  ; MODIFIED BY:
0000    42  ;
0000    43  ;       V02.18  MTR0001         Mike Rhodes     28-Oct-1981
0000    44  ;               Fix immediate mode H-Float short literal optimization/
0000    45  ;               generation code.  Also fixed broken BRBs to BRWs.
0000    46  ;
0000    47  ;       V02.17  PCG0011         Peter George    12-Oct-1981
0000    48  ;               The previous fix only handles explicit short literals.
0000    49  ;               Now extend that code to cover implicit short literals.
0000    50  ;
0000    51  ;       V02.16  PCG0010         Peter George    08-Sep-1981
0000    52  ;               Correct generation of short floating point literals.
0000    53  ;
0000    54  ;       V02.15  CNH0042         Chris Hume      1-Dec-1980
0000    55  ;               De-optimize boundary valued backward references if indexing
0000    56  ;               requested.  Allow the architecturally legal immediate mode in
0000    57  ;               address and vield contexts and also the practically useless
```

```
0000     58 ;                  indexed immediate mode.
0000     59 ;                  (ACTSTA.MAR 02.15, DEFINE.MAR 02.17, SYMTAB.MAR 02.18)
0000     60 ;
0000     61 ;          V01.14  RN0030          R. Newland      17-Mar-1980
0000     62 ;                  In explicit PC relative addressing test for displacement
0000     63 ;                  expression being absolute.  SPR 11-29354
0000     64 ;
0000     65 ;          V01.13  RN0029          R. Newland       9-Mar-1980
0000     66 ;                  In relative and relative deferred references always
0000     67 ;                  optimize if displacement is known, and use default
0000     68 ;                  displacement if not known. SPR 11-29062
0000     69 ;
0000     70 ;          V01.12  RN0023          R. Newland       3-Nov-1979
0000     71 ;                  New message codes to get error messages from system
0000     72 ;                  message file.
0000     73 ;
0000     74 ;          V01.11  RN0019          R. Newland      25-Oct-1979
0000     75 ;                  Improve error pointer positioning
0000     76 ;
0000     77 ;          V01.10  RN0017          R. Newland      20-Oct-1979
0000     78 ;                  Support for G_floating, H_floating and Octaword data types.
0000     79 ;                  Check all bits of operand for optimisation test.
0000     80 ;
0000     81 ;          V01.08  RN0014          R. Newland      14-Oct-1979
0000     82 ;                  Support for .REF16 directive
0000     83 ;
0000     84 ;          V01.07  RN0005          R. Newland      12-Aug-1979
0000     85 ;                  Remove .ALIGN LONG statements
0000     86 ;
0000     87 ;          V01.09  RN0017          R. Newland      20-Oct-1979
0000     88 ;                  Fix problem with optimising a quadword literal when
0000     89 ;                  bits 0-31 were all zero.
0000     90 ;
0000     91 ;          V01.08  RN0016          R. Newland      19-Oct-1979
0000     92 ;                  Don't output error messages when .NTYPE operand
0000     93 ;                  argument is the PC.  SPR 11-26392
0000     94 ;
0000     95 ;          V01.06  007             B. SCHREIBER    22-JAN-79
0000     96 ;                  Fix problem with negative displacement from PC.
0000     97 ;
0000     98 ;--
```

```
                        0000    100          .SBTTL  DECLARATIONS
                        0000    101 ;
                        0000    102 ; INCLUDE FILES:
                        0000    103 ;
                        0000    104
                        0000    105 ;
                        0000    106 ; MACROS:
                        0000    107 ;
                        0000    108
                        0000    109          $MAC_ADRMODDEF                  ;DEFINE ADDRESSING MODES
                        0000    110          $MAC_CTLFLGDEF                  ;DEFINE CONTROL FLAGS
                        0000    111          $MAC_GENVALDEF                  ;DEFINE GENERAL VALUES
                        0000    112          $MAC_INTCODDEF                  ;DEFINE INT. FILE CODES
                        0000    113          $MAC_OPRDEF                     ; Define operand descriptor bits
                        0000    114          $MAC_SYMBLKDEF                  ;DEFINE SYMBOL BLOCK OFFSETS
                        0000    115          $MACMSGDEF                      ; Define message codes
                        0000    116
                        0000    117 ;
                        0000    118 ; EQUATED SYMBOLS:
                        0000    119 ;
                        0000    120
                        0000    121 ;
                        0000    122 ; OWN STORAGE:
                        0000    123 ;
                        0000    124
                        0000    125
                    00000000    126          .PSECT  MAC$RO_DATA,NOWRT,NOEXE,GBL,LONG
                        0000    127
                        0000    128 PC_DISPL_DISP:                           ;JUMP TABLE USED WHEN
                        0000    129                                          ;.DEFAULT HAS BEEN SPECIFIED
                        0000    130                                          ;FOR PC-DISPLACMENT REFS
                    00000000    0000    131          .LONG   0               ;0--NOT USED
                    00000228'   0004    132          .LONG   DISPL3          ;1--BYTE DISPLACEMENT
                    00000238'   0008    133          .LONG   DISPL5          ;2--WORD DISPLACEMENT
                    00000230'   000C    134          .LONG   DISPL4          ;3--LONGWORD DISPLACEMENT
```

```
0010   136 ;++
0010   137 ; OPERANDS OF MACHINE INSTRUCTIONS ARE SPECIFIED BY REFERENCES.  BEFORE
0010   138 ; THESE ROUTINES ARE CALLED, THE FOLLOWING FLAGS MUST BE SET:
0010   139 ;
0010   140 ;       MAC$GL_OPSIZE   SIZE OF OPERAND IN BYTES (DEPENDS ON INSTRUCTION)
0010   141 ;
0010   142 ;       MAC$GL_REFSIZ   USER SUPPLIED REF WIDTH (IE MOVL ^S#1,R0)
0010   143 ;                       THIS VALUE MUST BE SET TO THE DEFAULT SIZE
0010   144 ;                       IF THE USER SPECIFIED NO WIDTH.
0010   145 ;
0010   146 ;
0010   147 ; THESE ROUTINES GENERATE INTERMEDIATE CODE SPECIFYING THE MODE(S)
0010   148 ; AND REGISTER(S) USED BY THE REFERENCE.
0010   149 ;
0010   150 ; THE RESULTS ARE RETURNED AS FOLLOWS:
0010   151 ;
0010   152 ;       MAC$GB_VAL1     MAC$GB_MODE     MODE OF OPERAND
0010   153 ;       MAC$GB_VAL2     MAC$GB_IMODE    ('E' MODE IF INDEX MODE)
0010   154 ;       MAC$GB_VAL3     MAC$GB_REG      (REGISTER USED BY OPERAND)
0010   155 ;       MAC$GB_VAL4     MAC$GB_IREG     ('E' REGISTER IF INDEX MODE)
0010   156 ;
0010   157 ; THE VALUE OF 'MAC$GL_PC' IS UPDATED BY THE SIZE OF THE REFERENCE
0010   158 ; AND THE ASSOCIATED VALUES, IF ANY.
0010   159 ;
0010   160 ;--
```

```
                     0010   162          .SBTTL  REFERENCE DIRECTIVE ROUTINES (.REFn)
                     0010   163
                 00000000   164          .PSECT  MAC$RO_CODE_P1,NOWRT,GBL,LONG
                     0000   165
                     0000   166 ;++
                     0000   167 ; FUNCTIONAL DESCRIPTION:
                     0000   168 ;
                     0000   169 ;       THIS ROUTINE IS INVOKED AFTER A REFERENCE DIRECTIVE HAS
                     0000   170 ;       BEEN PROCESSED.  THE ROUTINES RFHD1/2/4/8/16 SET UP THE
                     0000   171 ;       INFORMATION, THE REFERENCE IS THEN PARSED, AND DIRREF
                     0000   172 ;       IS THEN CALLED TO FINISH THE PRODUCTION.  THE REFERENCE
                     0000   173 ;       IS EMITTED TO THE INTERMEDIATE FILE.
                     0000   174 ;
                     0000   175 ;--
                     0000   176
                     0000   177 DIRREF::                                ;DIRECTIVE = REF_HEAD REF
                     0000   178          $INTOUT_LW INT$_REF,<W^MAC$GL_VALUE,W^MAC$GL_OPSIZE> ;OUTPUT REFERENCE
  08   0000'CF   91   000E   179          CMPB    W^MAC$GL_OPSIZE,#8         ;QUADWORD REFERENCE?
              11   12   0013   180          BNEQ    10$                        ;IF NEQ NO
  01   0000'CF   91   0015   181          CMPB    W^MAC$GB_VAL1,#ADM$_IMMEDIATE ;YES--IMMEDIATE?
              0A   12   001A   182          BNEQ    10$                        ;IF NEQ NO
                     001C   183          $INTOUT_LW INT$_STIL,<W^MAC$GL_VAL3> ;YES--STORE UPPER 32 BITS
                 05  0026   184 10$:      RSB
                     0027   185
                     0027   186 ;++
                     0027   187 ; FUNCTIONAL DESCRIPTION:
                     0027   188 ;
                     0027   189 ;       THESE ROUTINES (RFHD1/2/4/8) ARE INVOKED WHEN A REFERENCE
                     0027   190 ;       DIRECTIVE IS ENCOUNTERED.  RFHDX SET THE SIZE OF THE OPERAND
                     0027   191 ;       AND INITIALIZE STORAGE FOR PARSING THE EXPRESSION TO FOLLOW.
                     0027   192 ;
                     0027   193 ;--
                     0027   194
                     0027   195 RFHD1::                                  ;REF_HEAD = KREF1
                 0D  10   0027   196          BSBB    RFHD                       ;GO TO COMMON ROUTINE
                     01   0029   197          .BYTE   1                          ;OPERAND SIZE IS ONE BYTE
                     002A   198
                     002A   199 RFHD2::                                  ;REF_HEAD = KREF2
                 0A  10   002A   200          BSBB    RFHD                       ;GO TO COMMON ROUTINE
                     02   002C   201          .BYTE   2                          ;SIZE OF OPERAND IS 2 BYTES
                     002D   202
                     002D   203 RFHD4::                                  ;REF_HEAD = KREF4
                 07  10   002D   204          BSBB    RFHD                       ;GO TO COMMON ROUTINE
                     04   002F   205          .BYTE   4                          ;SIZE OF OPERAND IS 4 BYTES
                     0030   206
                     0030   207 RFHD8::                                  ;REF_HEAD = KREF8
                 04  10   0030   208          BSBB    RFHD                       ;GO TO COMMON ROUTINE
                     08   0032   209          .BYTE   8                          ;SIZE OF OPERAND IS 8 BYTES
                     0033   210
                     0033   211 RFHD16::                                 ;REF_HEAD = KREF16
                 01  10   0033   212          BSBB    RFHD                       ;GOTO COMMON ROUTINE
                     10   0035   213          .BYTE   16                         ;SIZE OF OPERAND IS 16 BYTES
                     0036   214
                     0036   215 RFHD:
      0000'CF    9E   9A   0036   216          MOVZBL  @(SP)+,W^MAC$GL_OPSIZE     ;SET OPERAND SIZE
0000'CF   0000'CF    9E   003B   217          MOVAB   W^MAC$GK_ZERO,W^MAC$GL_MOPPTR ;ALL MODES ARE LEGAL
          0000'CF    D4   0042   218          CLRL    W^MAC$GB_MODE              ;CLEAR MODE,IMODE,REG,IREG
```

```
00000044 8F   C8  0046   219          BISL2   #<FLG$M_EVALEXPR!FLG$M_COMPEXPR>,- ;EVALUATE EXPRESSION
         6B       004C   220                          (R11)                       ;AND ASSUME COMPILE TIME EXPR
                  05  004D   221          RSB                                    ;RETURN TO CALLER'S CALLER
```

```
                     004E   223          .SBTTL  OPERAND REFERENCE ROUTINES
                     004E   224
                     004E   225 ;++
                     004E   226 ; FUNCTIONAL DESCRIPTION:
                     004E   227 ;
                     004E   228 ;       REF1 AND REF3 ARE INVOKED WHEN A BASIC REFERENCE OR
                     004E   229 ;       AN INDEX REFERENCE IS PROCESSED TO COPY THE MODE AND
                     004E   230 ;       REGISTER VALUES INTO MAC$GB_VAL1-MAC$GB_VAL4.
                     004E   231 ;
                     004E   232 ;--
                     004E   233
                     004E   234 REF1::                                  ;REF = BASIC_REF
                     004E   235 REF3::                                  ;REF = INDEX_REF
                     004E   236 REF_EXIT:                               ;GENERAL REF EXIT
0000'CF   0000'CF    D0   004E   237          MOVL    W^MAC$GB_MODE,W^MAC$GB_VAL1 ;ENCODE MODES AND REGISTERS
                          0055   238                                     ;INTO GB_VAL1-GB_VAL4
                          0055   239 AUTOI::                            ;BASIC_REF = AUTO_INC
                          0055   240 DISP::                             ;BASIC_REF = DISPLACEMENT
                       05 0055   241          RSB
                          0056   242
                          0056   243 ;++
                          0056   244 ; FUNCTIONAL DESCRIPTION:
                          0056   245 ;
                          0056   246 ;       REF2 IS INVOKED WHEN A BASIC REGISTER REFERENCE IS ENCOUNTERED.
                          0056   247 ;       THE MODE (REGISTER MODE) IS ENCODED INTO MAC$GB_MODE, AND THE
                          0056   248 ;       REGISTER NUMBER IS SET IN MAC$GB_REG.  THE FINAL RESULTS ARE
                          0056   249 ;       THEN ENCODED INTO MAC$GB_VAL1-MAC$GB_VAL4.
                          0056   250 ;
                          0056   251 ; INPUTS:
                          0056   252 ;
                          0056   253 ;       MAC$GB_VAL1     REGISTER NUMBER
                          0056   254 ;
                          0056   255 ; OUTPUTS:
                          0056   256 ;
                          0056   257 ;       MAC$GB_MODE     SET TO 'ADM$_REGISTER'
                          0056   258 ;       MAC$GB_REG      REGISTER NUMBER
                          0056   259 ;       MAC$GB_VAL1-VAL4 ENCODED MODES AND REGISTERS
                          0056   260 ;
                          0056   261 ;--
                          0056   262
                          0056   263 REF2::                                  ;REF = RRREG
     0000'CF    05    90   0056   264          MOVB    #ADM$_REGISTER,W^MAC$GB_MODE ;MODE IS REGISTER MODE
0000'CF   0000'CF    90   005B   265          MOVB    W^MAC$GB_VAL1,W^MAC$GB_REG ;GET REGISTER VALUE
                     53   10 0062   266          BSBB    ILL_REG_CHK             ;CHECK FOR ILLEGAL PC USAGE
                     E8   11 0064   267          BRB     REF_EXIT                ;FINISH UP
```

```
                     0066   269          .SBTTL  DEFERRED REGISTER MODE OPERAND
                     0066   270
                     0066   271 ;++
                     0066   272 ; FUNCTIONAL DESCRIPTION:
                     0066   273 ;
                     0066   274 ;       THIS ROUTINE IS INVOKED WHEN A DEFERRED REGISTER MODE OPERAND
                     0066   275 ;       IS ENCOUNTERED.  THE MODE IS SET TO INDIRECT REGISTER, AND
                     0066   276 ;       THE REGISTER NUMBER IS STORED.
                     0066   277 ;
                     0066   278 ; INPUTS:
                     0066   279 ;
                     0066   280 ;       MAC$AL_VALSTACK-4[R7]   REGISTER NUMBER
                     0066   281 ;
                     0066   282 ; OUTPUTS:
                     0066   283 ;
                     0066   284 ;       MAC$GB_MODE             'ADM$_RRIND'--INDIRECT REGISTER MODE
                     0066   285 ;       MAC$GB_REG              REGISTER NUMBER
                     0066   286 ;
                     0066   287 ;--
                     0066   288
                     0066   289 RDEFER::                                    ;BASIC_REF = DOPN RRREG DCLS
          0000'CF    06   90  0066   290          MOVB    #ADM$_RRIND,W^MAC$GB_MODE ;SET MODE AS INDIRECT REG
0000'CF    FFFC'CF47    F6  006B   291          CVTLB   W^MAC$AL_VALSTACK-4[R7],- ;STORE REGISTER NUMBER
                        0073   292                  W^MAC$GB_REG
                      42   11  0073   293          BRB     ILL_REG_CHK             ;CHECK ILLEGAL PC AND RETURN
```

```
                     0075   295 ;++
                     0075   296 ; FUNCTIONAL DESCRIPTION:
                     0075   297 ;
                     0075   298 ;       THIS ROUTINE IS INVOKED WHEN THE PRODUCTION 'DAT AUTO_INC'
                     0075   299 ;       IS FOUND IN THE TEXT.  IF LITERAL MODE IS USED IT IS FORCED
                     0075   300 ;       TO IMMEDIATE LONGWORD.  THE MODE IS SET AS ABSOLUTE.
                     0075   301 ;
                     0075   302 ;--
                     0075   303
                     0075   304 AUTOII::                                ;BASIC_REF = DAT AUTO_INC
          03F8   30  0075   305         BSBW    SPESHL_MODE_CHK         ;FORCE LONGWORD TO IMMEDIATE
          03F5   30  0078   306         BSBW    SPESHL_MODE_CHK         ;FORCE LONGWORD CONTEXT
    0000'CF   96     007B   307         INCB    W^MAC$GB_MODE           ;MAKE INDIRECT MODE
          05         007F   308         RSB
                     0080   309
                     0080   310 ;++
                     0080   311 ; FUNCTIONAL DESCRIPTION:
                     0080   312 ;
                     0080   313 ;       THIS ROUTINE IS INVOKED WHEN AN AUTO-DECREMENT ADDRESSING MODE
                     0080   314 ;       IS ENCOUNTERED.  THE MODE AND REGISTER ARE SET UP, AND A CHECK
                     0080   315 ;       IS MADE FOR ILLEGAL USE OF 'PC'.
                     0080   316 ;
                     0080   317 ; INPUTS:
                     0080   318 ;
                     0080   319 ;       MAC$AL_VALSTACK-4[R7]   REGISTER NUMBER
                     0080   320 ;
                     0080   321 ; OUTPUTS:
                     0080   322 ;
                     0080   323 ;       MAC$GB_MODE             'ADM$_REGAUTODEC'
                     0080   324 ;       MAC$GB_REG              REGISTER NUMBER
                     0080   325 ;
                     0080   326 ;--
                     0080   327
                     0080   328 AUTODC::                                ;BASIC_REF = DMINUS DOPN RRREG DCLS
          0000'CF   07   90  0080   329         MOVB    #ADM$_REGAUTODEC,W^MAC$GB_MODE ;SET MODE
0000'CF   FFFC'CF47   F6   0085   330         CVTLB   W^MAC$AL_VALSTACK-4[R7],- ;PICK UP REGISTER NUMBER
                           008D   331                         @^MAC$GB_REG
                     28    11  008D   332         BRB     ILL_REG_CHK             ;CHECK ILLEGAL REGISTER AND RETURN
```

```
                        008F   334          .SBTTL  INDIRECT DISPLACEMENT REFERENCES
                        008F   335
                        008F   336 ;++
                        008F   337 ; FUNCTIONAL DESCRIPTION:
                        008F   338 ;
                        008F   339 ;       THIS ROUTINE PROCESSES INDIRECT DISPLACEMENT REFERENCES.
                        008F   340 ;       ABSOLUTE AND PIC MODES ARE CHANGED TO LONGWORD DISPLACEMENT,
                        008F   341 ;       LONGWORD DISPLACEMENTS ARE FORCE TO IMMEDIATE LONGWORD
                        008F   342 ;       CONTEXT. THE MODE IS THEN SET TO DEFERRED DISPLACEMENT MODE.
                        008F   343 ;
                        008F   344 ; INPUTS:
                        008F   345 ;
                        008F   346 ;       MAC$GB_MODE             MODE OF DISPLACEMENT
                        008F   347 ;
                        008F   348 ; OUTPUTS:
                        008F   349 ;
                        008F   350 ;       MAC$GB_MODE             DEFERRED DISPLACEMENT MODE
                        008F   351 ;       MAC$GL_PC               UPDATED IF NECESSARY
                        008F   352 ;
                        008F   353 ;--
                        008F   354
                        008F   355 DISPI::                                      ;BASIC REF = DAT DISPLACEMENT
 50  0000'CF    9E      008F   356          MOVAB   W^MAC$GB_MODE,R0           ;GET ADDRESS OF MODE
     02    60    91      0094   357          CMPB    (R0),#ADM$_ABSOLUTE        ;ABSOLUTE MODE?
           05    12      0097   358          BNEQ    20$                        ;IF NEQ NO
     60    0E    90      0099   359 10$:     MOVB    #ADM$_LONG_DISP,(R0)       ;YES--FORCE LONGWORD DISPLACEMENT
           11    11      009C   360          BRB     30$
     03    60    91      009E   361 20$:     CMPB    (R0),#ADM$_PIC             ;PIC CODE?
           F6    13      00A1   362          BEQL    10$                        ;IF EQL YES
     06    60    91      00A3   363          CMPB    (R0),#ADM$_RRIND           ;IF (R) THEN DE-OPTIMIZE
           07    12      00A6   364          BNEQ    30$                        ;NOT (R)
                 00A8   365          $INC_PC                                    ;ONE MORE BYTE
     60    0A    90      00AC   366          MOVB    #ADM$_BYTE_DISP,(R0)       ;USE BYTE DISPLACEMENT
        03BE    30      00AF   367 30$:     BSBW    SPESH$_MODE_CHK            ;CHECK MODES FOR DEFERRAL
0000'CF         96      00B2   368          INCB    W^MAC$GB_MODE              ;MAKE DEFERRED DISPLACEMENT
                05      00B6   369          RSB
```

```
                     00B7   371          .SBTTL  CHECK FOR ILLEGAL USE OF PC
                     00B7   372
                     00B7   373 ;++
                     00B7   374 ; FUNCTIONAL DESCRIPTION:
                     00B7   375 ;
                     00B7   376 ;       THIS ROUTINE CHECKS FOR ILLEGAL USE OF PC.  IF THE REGISTER
                     00B7   377 ;       IS PC, AN ERROR MESSAGE IS SENT TO PASS 2.  THE LOCATION COUNTER
                     00B7   378 ;       IS BUMPED BY 1.
                     00B7   379 ;
                     00B7   380 ; INPUTS:
                     00B7   381 ;
                     00B7   382 ;       MAC$GB_REG              REGISTER IN QUESTION
                     00B7   383 ;       MAC$GB_MODE             CURRENT MODE
                     00B7   384 ;
                     00B7   385 ; OUTPUTS:
                     00B7   386 ;
                     00B7   387 ;       MAC$GL_PC               INCREMENTED BY 1
                     00B7   388 ;
                     00B7   389 ;--
                     00B7   390
                     00B7   391 ILL_REG_CHK:
OF   0000'CF   91    00B7   392          CMPB    W^MAC$GB_REG,#REG$_PC   ;'PC' IS ILLEGAL REGISTER HERE
                18   12    00BC   393          BNEQ    10$             ;IF NEQ THEN LEGAL REGISTER
  14 6B   25         E4    00BE   394          BBSC    #FLG$V_NTYPEPC,(R11),10$ ; If .NTYPE directive no error
                     00C2   395          $MAC_ERR ILLREGHERE             ; Load message code
05   0000'CF   91    00C7   396          CMPB    W^MAC$GB_MODE,#ADM$_REGISTER ; Is addressing mode register?
                05   12    00CC   397          BNEQ    5$              ; No if NEQ
                FF2F' 30    00CE   398          BSBW    MAC$ERRORPX     ; Send error to pass-2
                03   11    00D1   399          BRB     10$
                     00D3   400 5$:
                FF2A' 30    00D3   401          BSBW    MAC$ERRORPT             ;SEND ERROR TO PASS2
                     00D6   402 10$:    $INC_PC                         ;ADD ONE TO LOCATION COUNTER
                05   00DA   403          RSB
```

```
                     00DB   405          .SBTTL  DISPLACEMENT REFERENCES
                     00DB   406
                     00DB   407 ;++
                     00DB   408 ; FUNCTIONAL DESCRIPTION:
                     00DB   409 ;
                     00DB   410 ;       THIS ROUTINE PROCESSES DISPLACEMENTS OFF OF A SPECIFIC
                     00DB   411 ;       REGISTER.  IF THE REGISTER IS 'PC', THE OPERAND IS
                     00DB   412 ;       TREATED AS 'EXPR'.  IF THE VALUE OF THE EXPRESSION IS 0
                     00DB   413 ;       AND WE CAN OPTIMIZE EXPRESSION, THE CODE FOR THE EXPRESSION
                     00DB   414 ;       IS DELETED FROM THE INTERMEDIATE BUFFER.  THE DISPLACEMENT
                     00DB   415 ;       IS SET TO BYTE, WORD, OR LONGWORD WITH WORD DISPLACEMENT
                     00DB   416 ;       THE DEFAULT.
                     00DB   417 ;
                     00DB   418 ; INPUTS:
                     00DB   419 ;
                     00DB   420 ;       MAC$AL_VALSTACK-12[R7]          EXPRESSION VALUE TO CONSIDER
                     00DB   421 ;       MAC$A' -VALSTACK-4[R7]          REGISTER BEING DISPLACED
                     00DB   422 ;
                     00DB   423 ; OUTPUTS:
                     00DB   424 ;
                     00DB   425 ;       MAC$GB_MODE                     SET WITH CORRECT MODE
                     00DB   426 ;       MAC$GL_PC                       UPDATED BY SIZE OF OPERAND
                     00DB   427 ;
                     00DB   428 ;--
                     00DB   429
                     00DB   430 DISPL1::                                ;DISPLACEMENT = EXPR DOPN RRREG DCLS
51   FFF4'CF47   D0  00DB   431          MOVL    W^MAC$AL_VALSTACK-12[R7],R1 ;GET EXPRESSION TO OPTIMIZE
       56   51   D0  00E1   432          MOVL    R1,R6                  ;COPY VALUE IN CASE REGISTER IS PC
52   FFFC'CF47   D0  00E4   433          MOVL    W^MAC$AL_VALSTACK-4[R7],R2      ;GET REGISTER BEING DISPLACED
  0000'CF   52   90  00EA   434          MOVB    R2,W^MAC$GB_REG        ;STORE FOR WORLD TO SEE
       0F   52   91  00EF   435          CMPB    R2,#REG$_PC            ;IS REGISTER 'PC'?
            09   12  00F2   436          BNEQ    5$                     ; No if NEQ
       0000'CF   D5  00F4   437          TSTL    W^MAC$GL_ABSFLAG       ; Is expression absolute?
            03   13  00F8   438          BEQL    5$                     ; Yes if EQL
          0087   31  00FA   439          BRW     DISPPC
                     00FD   440 5$:
    0000'CF   51   D0  00FD   441          MOVL    R1,W^MAC$GL_EXPOPVL1   ;LOAD EXPR VALUE
    0000'CF   0C   90  0102   442          MOVB    #ADM$_WORD_DISP,W^MAC$GB_MODE ;SET DEFAULT DISPLACEMENT
       56   03   9A  0107   443          MOVZBL  #3,R6                  ;DEFAULT IS 3 BYTES LONG
       0000'CF   D5  010A   444          TSTL    W^MAC$GL_ABSFLAG       ;ABSOLUTE EXPRESSION?
            0E   13  010E   445          BEQL    10$                    ;IF EQL YES
       6B   04   CA  0110   446          BICL2   #FLG$M_COMPEXPR,(R11)  ;NO--NOT COMPILE TIME EXPRESSION
51   FFFF0000   8F   D3  0113   447          BITL    #^XFFFF0000,R1         ;REQUIRE LONGWORD?
            3E   13  011A   448          BEQL    50$                    ;IF EQL NO--LINKER MAY COMPLAIN THO
            34   11  011C   449          BRB     40$                    ;YES--SET LONGWORD AND EXIT
            51   D5  011E   450 10$:      TSTL    R1                     ;VALUE 0?
            0F   12  0120   451          BNEQ    20$                    ;IF NEQ NO
     0B 6B   07   E1  0122   452          BBC     #FLG$V_EXPOPT,(R11),20$ ;BRANCH IF CANNOT OPTIMIZE EXPR
         FED7'   30  0126   453          BSBW    MAC$OPTIMIZEXPR        ;WE CAN--WIPE OUT EXPRESSION
    0000'CF   59   D0  0129   454          MOVL    R9,W^MAC$GL_EXPEND     ;SET NEW END OF EXPR POINTER
         FF35   31  012E   455          BRW     W^RDEFER               ;USE REGISTER DEFERRED MODE
            03   18  0131   456 20$:      bgeq    30$                    ; If negative
       51   51   D2  0133   457          mcoml   r1,r1                  ; then make it positive magnitude.
00007FFF   8F   51   D1  0136   458 30$:      cmpl    r1,#^x7fff             ; Byte or Word Displacement?
            13   1A  013D   459          bgtru   40$                    ; If GTRU no.
0000007F   8F   51   D1  013F   460          cmpl    r1,#^x7f               ; Yes:  Byte Displacement?
            12   1A  0146   461          bgtru   50$                    ; If GTRU no:  Word Disp. and PC OK.
```

```
                0000'CF   0A   90  0148   462           movb    #ADM$_BYTE_DISP,w^MAC$GB_MODE ; Yes:  Set Byte Displacement.
                      56  02   9A  014D   463           MOVZBL  #2,R6                         ;2 BYTE OPERAND
                          08   11  0150   464           BRB     50$
                0000'CF   0E   90  0152   465 40$:      MOVB    #ADM$_LONG_DISP,W^MAC$GB_MODE ;SET LONGWORD DISPLACEMENT
                      56  05   9A  0157   466           MOVZBL  #5,R6                         ;5 BYTES FOR OPERAND
                                  015A   467 50$:       $INC_PC R6                            ;UPDATE LOCATION COUNTER
                0F    0000'CF   91  015F   468           CMPB    W^MAC$GB_REG,#REG$_PC         ; Is reference from PC?
                          07    12  0164   469           BNEQ    60$                           ; No if NEQ
          0000'CF   0000'CF   C0  0166   470           ADDL2   W^MAC$GL_PC,W^MAC$GL_EXPOPVL1 ; Send expression value to pass-2
                                  016D   471                                                 ; with PC added in, it will be subtracted
                                  016D   472                                                 ; by pass-2.
                                  016D   473 60$:
                              05  016D   474           RSB
```

```
                     016E   476          .SBTTL  DISPLACEMENT OFF 'PC'
                     016E   477
                     016E   478 ;++
                     016E   479 ; FUNCTIONAL DESCRIPTION:
                     016E   480 ;
                     016E   481 ;       THIS ROUTINE IS INVOKED WHEN A DISPLACEMENT OF THE FORM
                     016E   482 ;       'EXPR' IS ENCOUNTERED.  THE CORRECT SIZE OF THE DISPLACEMENT
                     016E   483 ;       IS DETERMINED AND THE MODE IS SET AND THE PC IS UPDATED.
                     016E   484 ;
                     016E   485 ; INPUTS:
                     016E   486 ;
                     016E   487 ;       MAC$AL_VALSTACK[R7]     VALUE OF EXPRESSION
                     016E   488 ;
                     016E   489 ; OUTPUTS:
                     016E   490 ;
                     016E   491 ;       MAC$GB_MODE             SET TO CORRECT MODE
                     016E   492 ;       MAC$GL_EXPOPVL1         SET TO VALUE OF EXPRESSION
                     016E   493 ;       MAC$GL_PC               UPDATED BY SIZE OF OPERAND
                     016E   494 ;
                     016E   495 ;--
                     016E   496
                     016E   497 DISPL2::                                    ;DISPLACEMENT = EXPR
        06 0005'CF   E9   016E   498          BLBC    W^ENB$G_ABSADDR+SYM$L_VAL,10$ ;BRANCH IF NOT ENABLE AMA
              0195   30   0173   499          BSBW    AINC2                  ;YES--TREAT AS IMMEDIATE
              FEFC   31   0176   500          BRW     AUTOII                 ;THEN MAKE ABSOLUTE AND EXIT
56    0000'CF47   D0   0179   501 10$:       MOVL    W^MAC$AL_VALSTACK[R7],R6 ;GET VALUE
  0000'CF   0F   90   017F   502          MOVB    #REG$_PC,W^MAC$GB_REG  ;SET PC AS REGISTER
  0000'CF   56   D0   0184   503 DISPPC:  MOVL    R6,W^MAC$GL_EXPOPVL1   ;COPY VALUE
              51   D4   0189   504          CLRL    R1                     ; Initialise absolute flag
         0000'CF   D5   018B   505          TSTL    W^MAC$GL_ABSFLAG       ; Absolute expression?
              04   12   018F   506          BNEQ    1$                     ; No if NEQ
              51   D6   0191   507          INCL    R1                     ; Set flag for later
              29   11   0193   508          BRB     5$
                   0195   509 1$:
      25 6B   02   E0   0195   510          BBS     #FLG$V_COMPEXPR,(R11),5$ ; Branch if compile time expression
                   0199   511 2$:
   50   0000'CF   D0   0199   512          MOVL    W^MAC$GL_DFPC_DSP,R0   ;GET THE DISPLACEMENT FROM .DEFAULT
              03   12   019E   513          BNEQ    3$                     ; Branch if has been specified
         50   03   D0   01A0   514          MOVL    #3,R0                  ; Use 'longword' displacment as default
                   01A3   515 3$:
         57   04   C0   01A3   516          ADDL2   #4,R7                  ;FAKE UP STACK FRAME
FFFC'CF47   0F   D0   01A6   517          MOVL    #REG$_PC,W^MAC$AL_VALSTACK-4[R7] ;SET REGISTER
FFF4'CF47   56   D0   01AC   518          MOVL    R6,W^MAC$AL_VALSTACK-12[R7] ;AND VALUE
50    0000'CF40   D0   01B2   519          MOVL    W^PC_DISPL_DISP[R0],R0 ;GET THE ROUTINE ADDRESS
              60   16   01B8   520          JSB     (R0)                   ;GO TO  DISPL3/4/5
         57   04   C2   01BA   521          SUBL2   #4,R7                  ;ADJUST STACK
              05   01BD   522          RSB                            ;DONE
  0000'CF   0E   90   01BE   523 5$:      MOVB    #ADM$_LONG_DISP,W^MAC$GB_MODE ;SET DEFAULT MODE
         54   01   9A   01C3   524          MOVZBL  #1,R4                  ;INC/DEC VALUE FOR PC
                   01C6   525          $INC_PC #2                     ;OPERAND IS AT LEAST 2 BYTES
        51 51   E8   01CB   526          BLBS    R1,50$                 ; Branch if expression was not absolute
                   01CE   527 ;
                   01CE   528 ; EXPRESSION CAN BE OPTIMIZED
                   01CE   529 ;
   56   0000'CF   C2   01CE   530          SUBL2   W^MAC$GL_PC,R6         ;GET DISPLACEMENT
              06   18   01D3   531          bgeq    10$                    ; Branch if positive displacement
         54   54   CE   01D5   532          mnegl   r4,r4                  ; Need to add one if not Byte
```

```
                56   56   D2   01D8   533                mcoml   r6,r6                           ; Get positive magnitude -1
      0000007F  8F   56   D1   01DB   534 10$:           cmpl    r6,#^x7f                        ; Byte Displacement?
                     11   1A   01E2   535                bgtru   20$                             ; If greater in magnitude:  No.
                     08   12   01E4   536                bnequ   15$                             ; Flag at boundary in case of indexing.
                     54   D5   01E6   537                tstl    r4                              ; If forward ref. indexing is no prob.
                     04   18   01E8   538                bgeq    15$
            00   6B   2C   E2   01EA   539                bbss    #FLG$V_OPTVFLIDX,(r11),15$
          0000'  CF   0A   90   01EE   540 15$:           movb    #ADM$_BYTE_DISP,w^MAC$GB_MODE ; Yes:  Set Byte Displacement,
                     32   11   01F3   541                brb     60$                             ; PC is correct now.
                56   54   C2   01F5   542 20$:           subl2   r4,r6                           ; Include extra byte in displacement.
      00007FFF  8F   56   D1   01F8   543                cmpl    r6,#^x7fff                      ; Word Displacement?
                     14   1A   01FF   544                bgtru   30$                             ; If greater in magnitude:  No.
                     08   12   0201   545                bnequ   25$                             ; Flag at boundary in case of indexing.
                     54   D5   0203   546                tstl    r4                              ; If forward ref. indexing is no prob.
                     04   18   0205   547                bgeq    25$
            00   6B   2C   E2   0207   548                bbss    #FLG$V_OPTVFLIDX,(r11),25$
          0000'  CF   0C   90   020B   549 25$:           movb    #ADM$_WORD_DISP,w^MAC$GB_MODE ; Yes:  Set Word Displacement,
                54   01   9A   0210   550                movzbl  #1,r4                           ; Update PC by one.
                     03   11   0213   551                brb     40$
                54   03   9A   0215   552 30$:           movzbl  #3,r4                           ; Longword:  update PC by 3.
                          0218   553 40$:           $INC_PC R4                              ;UPDATE LOCATION COUNTER
                     08   11   021D   554                BRB     60$
                          021F   555 ;
                          021F   556 ; Not compile time expression (not Absolute)
                          021F   557 ;
                6B   04   CA   021F   558 50$:           BICL2   #FLG$M_COMPEXPR,(R11)           ;FLAG NOT COMPILE TIME EXPRESSION
                          0222   559                $INC_PC #3                              ;UPDATE PC FOR DEFAULT LONGWORD
                     05   0227   560 60$:           RSB
```

```
                     0228   562          .SBTTL  EXPLICIT DISPLACEMENTS AND PIC REFERENCES
                     0228   563
                     0228   564 ;++
                     0228   565 ; FUNCTIONAL DESCRIPTION:
                     0228   566 ;
                     0228   567 ;       THESE ROUTINES (DISPL3/4/5) ARE INVOKED WHEN AN EXPLICIT
                     0228   568 ;       DISPLACEMENT REFERENCE IS ENCOUNTERED.  THE CORRECT MODE
                     0228   569 ;       IS SET, THE REGISTER IS SET, AND THE PC IS UPDATED BY
                     0228   570 ;       THE SIZE OF THE OPERAND.
                     0228   571 ;
                     0228   572 ; INPUTS:
                     0228   573 ;
                     0228   574 ;       MAC$AL_VALSTACK-4[R7]   REGISTER VALUE
                     0228   575 ;       MAC$AL_VALSTACK-12[R7]  EXPRESSION VALUE
                     0228   576 ;
                     0228   577 ; OUTPUTS:
                     0228   578 ;
                     0228   579 ;       MAC$GB_MODE             SET TO CORRECT MODE
                     0228   580 ;       MAC$GB_REG              SET TO REGISTER
                     0228   581 ;       MAC$GL_PC               UPDATED BY SIZE OF OPERAND
                     0228   582 ;
                     0228   583 ;--
                     0228   584
                     0228   585          .ENABL  LSB
                     0228   586
                     0228   587 DISPL3::                                ;DISPLACEMENT = DBUP EXPR DOPN RRREG DCLS
 50   0A   9A        0228   588          MOVZBL  #ADM$_BYTE_DISP,R0      ;SET BYTE DISPLACEMENT
 51   02   9A        022B   589          MOVZBL  #2,R1                   ;LENGTH OF OPERAND
      0E   11        022E   590          BRB     10$
                     0230   591
                     0230   592 DISPL4::                                ;DISPLACEMENT = DLUP EXPR DOPN RRREG DCLS
 50   0E   9A        0230   593          MOVZBL  #ADM$_LONG_DISP,R0      ;SET LONGWORD DISPLACEMENT
 51   05   9A        0233   594          MOVZBL  #5,R1                   ;LENGTH OF OPERAND
      06   11        0236   595          BRB     10$
                     0238   596
                     0238   597 DISPL5::                                ;DISPLACEMENT = DWUP EXPR DOPN RRREG DCLS
 50   0C   9A        0238   598          MOVZBL  #ADM$_WORD_DISP,R0      ;SET WORD DISPLACEMENT
 51   03   9A        023B   599          MOVZBL  #3,R1                   ;LENGTH OF OPERAND
0000'CF  FFFC'CF47   F6     023E   600 10$:     CVTLB   W^MAC$AL_VALSTACK-4[R7],- ;GET THE REGISTER VALUE
                     0246   601                         @^MAC$GB_REG
0000'CF  FFF4'CF47   D0     0246   602          MOVL    W^MAC$AL_VALSTACK-12[R7],- ;COPY EXPRESSION VALUE
                     024E   603                         @^MAC$GL_EXPOPVL1
          0000'CF    D5     024E   604          TSTL    W^MAC$GL_ABSFLAG        ;ABSOLUTE EXPRESSION?
                03   13     0252   605          BEQL    20$                     ;IF EQL YES
            6B  04   CA     0254   606          BICL2   #FLG$M_COMPEXPR,(R11)   ;ELSE NOT COMPILE TIME EXPRESSION
                     0257   607 DISPLX_EXIT:
     0000'CF    50   90     0257   608 20$:     MOVB    R0,W^MAC$GB_MODE        ;SET MODE OF OPERAND
                     025C   609          $INC_PC R1                      ;UPDATE PC BY SIZE OF OPERAND
                05   05     0261   610          RSB
                     0262   611
                     0262   612          .DSABL  LSB
```

```
                     0262   614 ;++
                     0262   615 ; FUNCTIONAL DESCRIPTION:
                     0262   616 ;
                     0262   617 ;       THESE ROUTINES (DISPL6/7/8 AND GENLDS) ARE INVOKED WHEN
                     0262   618 ;       A SPECIFIC SIZE DISPLACEMENT REFERENCE OR A SPECIFIC PIC
                     0262   619 ;       REFERENCE IS ENCOUNTERED.  THE REGISTER IS SET TO 'PC',
                     0262   620 ;       AND THE PC IS UPDATED BY THE SIZE OF THE OPERAND.
                     0262   621 ;
                     0262   622 ; INPUTS:
                     0262   623 ;
                     0262   624 ;       MAC$AL_VALSTACK[R7]     EXPRESSION VALUE
                     0262   625 ;
                     0262   626 ; OUTPUTS:
                     0262   627 ;
                     0262   628 ;       MAC$GB_MODE             SET TO MODE
                     0262   629 ;       MAC$GB_REG              SET TO 'PC' (15.)
                     0262   630 ;       MAC$GL_PC               UPDATED BY SIZE OF OPERAND
                     0262   631 ;       MAC$GL_EXPOPVL1         SET TO VALUE OF EXPRESSION
                     0262   632 ;
                     0262   633 ;--
                     0262   634
                     0262   635         .ENABL  LSB
                     0262   636
                     0262   637 DISPL6::                                    ;DISPLACEMENT = DBUP EXPR
     50    0A    9A  0262   638         MOVZBL  #ADM$_BYTE_DISP,R0          ;SET BYTE DISPLACEMENT
     51    02    9A  0265   639         MOVZBL  #2,R1                       ;SET SIZE OF OPERAND
           16    11  0268   640         BRB     10$
                     026A   641
                     026A   642 DISPL7::                                    ;DISPLACEMENT = DLUP EXPR
     50    0E    9A  026A   643         MOVZBL  #ADM$_LONG_DISP,R0          ;SET LONGWORD DISPLACEMENT
     51    05    9A  026D   644         MOVZBL  #5,R1                       ;SIZE IS 5 BYTES
           0E    11  0270   645         BRB     10$
                     0272   646
                     0272   647 DISPL8::                                    ;DISPLACEMENT = DWUP EXPR
     50    0C    9A  0272   648         MOVZBL  #ADM$_WORD_DISP,R0          ;SET WORD DISPLACEMENT
     51    03    9A  0275   649         MOVZBL  #3,R1                       ;SIZE IS 3 BYTES
           06    11  0278   650         BRB     10$                         ;
                     027A   651
                     027A   652 GENLDS::                                    ;DISPLACEMENT = DGUP EXPR
     50    03    9A  027A   653         MOVZBL  #ADM$_PIC,R0                ;SET PIC MODE
     51    05    9A  027D   654         MOVZBL  #5,R1                       ;LENGTH IS 5 BYTES
   0000'CF    0F    90  0280   655 10$:  MOVB    #REG$_PC,W^MAC$GB_REG       ;SET REGISTER TO PC
0000'CF 0000'CF47    D0  0285   656         MOVL    W^MAC$AL_VALSTACK[R7],-     ;COPY EXPRESSION
                     028D   657                         W^MAC$GL_EXPOPVL1
   0000'CF    D5    12  028D   658         TSTL    W^MAC$GL_ABSFLAG            ;ABSOLUTE EXPRESSION?
           C4    12  0291   659         BNEQ    DISPLX_EXIT                 ;IF NEQ NO--GO FINISH UP
     6B    04    CA  0293   660         BICL2   #FLG$M_COMPEXPR,(R11)       ;YES--ABS REQUIRES LINKER ACTION
           BF    11  0296   661         BRB     DISPLX_EXIT                 ;GO EXIT
                     0298   662
                     0298   663         .DSABL  LSB
```

```
                        0298   665          .SBTTL  LITERAL MODE
                        0298   666
                        0298   667 ;++
                        0298   668 ; FUNCTIONAL DESCRIPTION:
                        0298   669 ;
                        0298   670 ;       THIS ROUTINE IS INVOKED WHEN A LITERAL REFERENCE IS SCANNED.
                        0298   671 ;       IF THE OPERAND IS A FLOATING POINT OPERAND, IT IS CONVERTED
                        0298   672 ;       TO A SHORT FLOATING LITERAL.
                        0298   673 ;
                        0298   674 ; INPUTS:
                        0298   675 ;
                        0298   676 ;       MAC$AL_VALSTACK[R7]     VALUE OF EXPRESSION
                        0298   677 ;
                        0298   678 ;--
                        0298   679
                        0298   680 LITERL::                                     ;REF = DSUP DPOUND EXPR
          0000'CF   D5  0298   681          TSTL    W^MAC$GL_ABSFLAG            ;ABSOLUTE?
                04  13  029C   682          BEQL    10$                         ;IF EQL YES
          00 6B   07  E5  029E   683          BBCC    #FLG$V_EXPOPT,(R11),10$ ;NO--DO NOT ALLOW OPTIMIZATION
        0000'CF   00  90  02A2   684 10$:     MOVB    #ADM$_LITERAL,W^MAC$GB_MODE ;SET LITERAL MODE
                      02A7   685          $INC_PC                             ;COUNT 1 BYTE
0000'CF 0000'CF47   D0  02AB   686          MOVL    W^MAC$AL_VALSTACK[R7],- ;COPY THE VALUE
                      02B3   687                      @^MAC$GL_EXPOPVL1
          0000'CF   91  02B3   688          CMPB    W^MAC$GB_RDXNDX,-           ;READING FLT. PT.?
                04      02B7   689                  #RDX$V_FLOAT
                39  1F  02B8   690          BLSSU   40$                         ;IF LESS THEN NO
        50  0000'CF   D0  02BA   691          MOVL    W^MAC$GL_EXPOPVL1,R0        ;MOVE FLOATING PT. NUMBER INTO R0
          0000'CF   91  02BF   692          CMPB    W^MAC$GB_RDXNDX,-           ;H FLOAT?
                07      02C3   693                  #RDX$V_HFLOAT
                16  12  02C4   694          BNEQ    20$                         ;NO, THEN SKIP TO OTHER FLT. PT. CODE
      51    50  03  1D  EF  02C6   695          EXTZV   #29,#3,R0,R1            ;GET FRACTION
      52    50  03  00  EF  02CB   696          EXTZV   #0,#3,R0,R2             ;GET EXPONENT
      51    03  03  52  F0  02D0   697          INSV    R2,#3,#3,R1             ;COMBINE THEM
            0000'CF   51  9A  02D5   698          MOVZBL  R1,W^MAC$GL_EXPOPVL1    ;STORE RESULT
                17  11  02DA   699          BRB     40$                         ;DONE
          0000'CF   91  02DC   700 20$:     CMPB    W^MAC$GB_RDXNDX,-           ;G FLOAT?
                06      02E0   701                  #RDX$V_GFLOAT
                09  12  02E1   702          BNEQ    30$                         ;NO, SKIP TO F AND D FLOAT COMMON CODE
0000'CF   50  06  01  EF  02E3   703          EXTZV   #1,#6,R0,-                  ;YES, STORE RESULT
                      02EA   704                  W^MAC$GL_EXPOPVL1
                07  11  02EA   705          BRB     40$
0000'CF   50  06  04  EF  02EC   706 30$:     EXTZV   #4,#6,R0,-                  ;F OR D FLOAT, STORE RESULT
                      02F3   707                  W^MAC$GL_EXPOPVL1
              FD58  31  02F3   708 40$:     BRW     REF_EXIT                    ;EXIT FROM REFERENCE
```

```
                     02F6   710          .SBTTL  AUTO-INCREMENT MODE
                     02F6   711
                     02F6   712 ;++
                     02F6   713 ; FUNCTIONAL DESCRIPTION:
                     02F6   714 ;
                     02F6   715 ;       THIS ROUTINE IS CALLED WHEN AN AUTO-INCREMENT REFERENCE IS
                     02F6   716 ;       SCANNED.  THE MODE AND REGISTER ARE SET, AND A CHECK IS MADE
                     02F6   717 ;       FOR AN ILLEGAL REGISTER REFERENCE.
                     02F6   718 ;
                     02F6   719 ; INPUTS:
                     02F6   720 ;
                     02F6   721 ;       MAC$AL_VALSTACK-8[R7]   RESGISTER NUMBER
                     02F6   722 ;
                     02F6   723 ; OUTPUTS:
                     02F6   724 ;
                     02F6   725 ;       MAC$GB_MODE             'ADM$_REGAUTOINC'
                     02F6   726 ;       MAC$GB_REG              SET TO REGISTER
                     02F6   727 ;
                     02F6   728 ;--
                     02F6   729
                     02F6   730 AINC1::                                  ;AUTO_INC = DOPN RRREG DCLS DPLUS
                     02F6   731
        0000'CF    08   90   02F6   732          MOVB    #ADM$_REGAUTOINC,W^MAC$GB_MODE ;SET MODE
0000'CF    FFF8'CF47   F6   02FB   733          CVTLB   W^MAC$AL_VALSTACK-8[R7],-  ;GET REGISTER NUMBER
                     0303   734                          W^MAC$GB_REG
                   FDB1   31   0303   735          BRW     ILL_REG_CHK             ;CHK REG ERROR AND RETURN
```

```
                     0306   737          .SBTTL  IMMEDIATE MODE
                     0306   738
                     0306   739 ;++
                     0306   740 ; FUNCTIONAL DESCRIPTION:
                     0306   741 ;
                     0306   742 ;       THIS ROUTINE IS INVOKED WHEN AN IMMEDATE REFERENCE IS SCANNED.
                     0306   743 ;       (ALSO FOR AUTO-INCRMENT OF PC [#EXPR]).
                     0306   744 ;
                     0306   745 ;--
                     0306   746
                     0306   747 AINC3::                                      ;BASIC_REF = DIUP DPOUND EXPR
0000'CF   01   9A    0306   748          MOVZBL  #1,W^MAC$GL_ABSFLAG         ;DO NOT ALLOW OPTIMIZATION
                     030B   749
                     030B   750          .SBTTL  AUTO-INCREMENT PC
                     030B   751
                     030B   752 AINC2::                                      ;AUTO_INC = DPOUND DEXPR
     0000'CF   D5    030B   753          TSTL    W^MAC$GL_ABSFLAG            ;ABSOLUTE EXPRESSION?
          04   13    030F   754          BEQL    10$                         ;IF EQL YES
  00 6B   07   E5    0311   755          BBCC    #FLG$V_EXPOPT,(R11),10$ ;NO--DO NOT ALLOW OPTIMIZATION
56   0000'CF   9E    0315   756 10$:     MOVAB   W^MAC$GL_EXPOPVL1,R6        ;POINT TO RESULT
66 0000'CF47   D0    031A   757          MOVL    W^MAC$AL_VALSTACK[R7],(R6) ;SET RESULT
     0000'CF   D5    0320   758          TSTL    W^MAC$GL_ABSFLAG            ;ABSOULTE EXPRESSION?
          2E   12    0324   759          BNEQ    17$                         ;IF NEQ NO
10   0000'CF   91    0326   760          CMPB    W^MAC$GL_OPSIZE,#16         ; Is operand octaword?
          0E   12    032B   761          BNEQ    12$                         ; No if NEQ
     0004'CF   D5    032D   762          TSTL    W^MAC$GQ_HIGH_64+4          ; Are bits 96-127 all zero?
          21   12    0331   763          BNEQ    17$                         ; No if NEQ
     0000'CF   D5    0333   764          TSTL    W^MAC$GQ_HIGH_64+0          ; Are bits 64-95 all zero?
          1B   12    0337   765          BNEQ    17$                         ; No if NEQ
          07   11    0339   766          BRB     14$
                     033B   767 12$:
08   0000'CF   91    033B   768          CMPB    W^MAC$GL_OPSIZE,#8          ; Is operand quadword?
          06   12    0340   769          BNEQ    16$                         ; No if NEQ?
                     0342   770 14$:
     0000'CF   D5    0342   771          TSTL    W^MAC$GL_HIGH_32            ; Are bits 32-63 all zero?
          0C   12    0346   772          BNEQ    17$                         ; No if NEQ
                     0348   773 16$:
     0000'CF   91    0348   774          CMPB    W^MAC$GB_RDXNDX,-           ;H FLOAT?
          07         034C   775                  #RDX$V_HFLOAT
          08   13    034D   776          BEQL    18$                         ;YES, THEN LEAVE UPPER WORD FOR LATER
       02 A6   B5    034F   777          TSTW    2(R6)                       ;NO, UPPER 16 BITS 0?
          03   13    0352   778          BEQL    18$                         ;IF EQL THEN YES
        0081   31    0354   779 17$:     BRW     40$
                     0357   780                                              ; Yes:  Get Operand bits, isolate Mode.
50   0000'CF   D0    0357   781 18$:     movl    w^MAC$GL_MOPPTR,r0
          68   13    035C   782          beql    26$                         ; If null pointer assume integer.
51 60 05   05   EE    035E   783          extv    #OPD$V_MODE,#OPD$S_MODE,(r0),r1
       51 00   B1    0363   784          cmpw    #OPD$M_ADDR,r1              ; Can't allow short lit. if Addr. ref.,
          70   13    0366   785          beql    40$
51   0080 8F   B1    0368   786          cmpw    #OPD$M_VIELD,r1             ; ditto for Vield references.
          69   13    036D   787          beql    40$
          60   B5    036F   788          tstw    (r0)                        ; Floating Operand?
          53   18    0371   789          bgeq    26$                         ; No if positive.
                     0373   790 ;
                     0373   791 ; ABSOLUTE FLOATING OPERAND--MAKE SHORT FLOATING IF WE CAN
                     0373   792 ;
50   66   C000 8F   A1    0373   793          ADDW3   #^XC000,(R6),R0         ;CLEAR HIGH ORDER BIT IN EXPONENT
```

```
                0000'CF   91   0379   794          CMPB    W^MAC$GB_RDXNDX,-       ;H FLOAT?
                      07       037D   795                  #RDX$V_HFLOAT
                      23   12  037E   796          BNEQ    20$                     ;NO, THEN SKIP TO OTHER FLT. PT. CODE
50   10   10   02  A6   F0  0380   797          INSV    2(R6), #16, #16, R0     ;YES, MOVE FRACTION TO LOW WORD OF R0
     50   1FFFFFF8 8F   D3  0386   798          BITL    #^X1FFFFFF8,R0          ;ARE NECESSARY BITS ZERO?
                      49   12  038D   799          BNEQ    40$                     ;IF NEQ NO
     51   50   03   1D  EF   038F   800          EXTZV   #29,#3,R0,R1            ;GET FRACTION
     52   50   03   00  EF   0394   801          EXTZV   #0,#3,R0,R2             ;GET EXPONENT
     51   03   03   52  F0   0399   802          INSV    R2,#3,#3,R1             ;COMBINE THEM
                 66   51  9A   039E   803          MOVZBL  R1,(R6)                 ;STORE RESULT
                      2A   11  03A1   804          BRB     30$                     ;DONE
                           03A3   805
                0000'CF   91   03A3   806 20$:    CMPB    W^MAC$GB_RDXNDX,-       ;G FLOAT?
                      06       03A7   807                  #RDX$V_GFLOAT
                      0E   12  03A8   808          BNEQ    23$                     ;NO, SKIP TO F AND D FLOAT COMMON CODE
           50   FF81 8F   B3  03AA   809          BITW    #^XFF81,R0              ;ARE NECESSARY BITS ZERO?
                      27   12  03AF   810          BNEQ    40$                     ;IF NEQ NO
     66    50   06   01  EF   03B1   811          EXTZV   #1,#6,R0,(R6)           ;YES, STORE RESULT
                      15   11  03B6   812          BRB     30$
                           03B8   813
           50   FC0F 8F   B3  03B8   814 23$:    BITW    #^XFC0F,R0              ;ARE NECESSARY BITS ZERO?
                      19   12  03BD   815          BNEQ    40$                     ;IF NEQ NO
     66    50   06   04  EF   03BF   816          EXTZV   #4,#6,R0,(R6)           ;F OR D FLOAT, STORE RESULT
                      07   11  03C4   817          BRB     30$
                           03C6   818 ;
                           03C6   819 ; ABSOLUTE NON-FLOATING--MAKE SHORT LITERAL IF WE CAN
                           03C6   820 ;
           66   FFC0 8F   B3  03C6   821 26$:    BITW    #^C<^X3F>,(R6)          ;INTEGER LESS THAN 64?
                      0B   12  03CB   822          BNEQ    40$                     ;IF NEQ NO
           0000'CF   00   90  03CD   823 30$:    MOVB    #ADM$_LITERAL,W^MAC$GB_MODE ;SET LITERAL MODE
                           03D2   824          $INC_PC                         ;COUNT THE BYTE
                      16   11  03D6   825          BRB     50$
                           03D8   826 ;
                           03D8   827 ; MUST USE IMMEDIATE MODE
                           03D8   828 ;
0000'CF    01   90  03D8   829 40$:    MOVB    #ADM$_IMMEDIATE,W^MAC$GB_MODE ;SET THE MODE
0000'CF    0F   90  03DD   830          MOVB    #REG$_PC,W^MAC$GB_REG   ;REGISTER IS PC
50   0000'CF   9A  03E2   831          MOVZBL  W^MAC$GL_OPSIZE,R0      ;GET THE OPERAND SIZE
           50   D6  03E7   832          INCL    R0                      ;PLUS ONE BYTE
                    03E9   833          $INC_PC R0                      ;UPDATE LOCATION COUNTER
                05  03EE   834 50$:    RSB
```

```
                     03EF    836          .SBTTL  DEFERRED INDIRECT REGISTER REFERENCE
                     03EF    837
                     03EF    838 ;++
                     03EF    839 ; FUNCTIONAL DESCRIPTION:
                     03EF    840 ;
                     03EF    841 ;       ROUTINE IS INVOKED WHEN A DEFERRED INDIRECT REGISTER REFERENCE
                     03EF    842 ;       IS SCANNED.  THE MODE AND REGISTER ARE SET, AND THE PC IS
                     03EF    843 ;       INCREMENTED BY TWO BYTES.
                     03EF    844 ;
                     03EF    845 ; INPUTS:
                     03EF    846 ;
                     03EF    847 ;       MAC$AL_VALSTACK-4[R7]   REGISTER
                     03EF    848 ;
                     03EF    849 ; OUTPUTS:
                     03EF    850 ;
                     03EF    851 ;       MAC$GB_MODE             'ADM$_DFBYTEDISP'
                     03EF    852 ;       MAC$GB_REG              SET TO REGISTER
                     03EF    853 ;       MAC$GL_PC               INCREMENTED BY 2
                     03EF    854 ;
                     03EF    855 ;--
                     03EF    856
                     03EF    857 AUTOI3::                                    ;BASIC_REF = DAT DOPN RRREG DCLS
          0000'CF    D4  03EF    858          CLRL    W^MAC$GL_EXPOPVL1       ;VALUE IS 0
     0000'CF    0B    90  03F3    859          MOVB    #ADM$_DFBYTEDISP,W^MAC$GB_MODE ;SET THE MODE
0000'CF    FFFC'CF47    F6  03F8    860          CVTLB   W^MAC$AL_VALSTACK-4[R7],- ;GET THE REGISTER
                         0400    861                          @^MAC$GB_REG
                         0400    862          $INC_PC #2                      ;EAT TWO BYTES
                     05  0405    863          RSB
```

```
                     0406   865          .SBTTL  INDEXED REFERENCES
                     0406   866
                     0406   867 ;++
                     0406   868 ; FUNCTIONAL DESCRIPTION:
                     0406   869 ;
                     0406   870 ;       THIS ROUTINE IS INVOKED WHEN AN INDEX REFERENCE IS FOUND. CHECKS
                     0406   871 ;       ARE MADE FOR ILLEGAL INDEX MODES, AND MESSAGES ISSUED TO PASS 2
                     0406   872 ;       IF ILLEGAL INDEX MODES ARE FOUND.  THE MODES AND REGISTERS OF
                     0406   873 ;       THE BASIC REFERENCE AND THE INDEX REFERENCE ARE SET UP AND THE
                     0406   874 ;       PC IS INCREMENTED TO COUNT THE INDEX REGISTER REFERENCE.
                     0406   875 ;
                     0406   876 ; INPUTS:
                     0406   877 ;
                     0406   878 ;       MAC$AL_VALSTACK-4[R7]           INDEX REGISTER NUMBER
                     0406   879 ;       MAC$GB_MODE                     MODE OF BASIC REFERENCE
                     0406   880 ;       MAC$GB_REG                      REGISTER OF BASIC REFERENCE
                     0406   881 ;
                     0406   882 ; OUTPUTS:
                     0406   883 ;
                     0406   884 ;       MAC$GB_MODE                     'ADM$_INDEX'
                     0406   885 ;       MAC$GB_REG                      INDEX REGISTER NUMBER
                     0406   886 ;       MAC$GB_IMODE                    MODE OF BASIC REFERENCE
                     0406   887 ;       MAC$GB_REG                      REGISTER OF BASIC REFERENCE
                     0406   888 ;       MAC$GL_PC                       INCREMENTED BY 1 FOR INDEX REF.
                     0406   889 ;
                     0406   890 ;--
                     0406   891
                     0406   892 INDEX::                                 ;INDEX_REF = BASIC_REF DSQOPN RRREG DSQCLS
     50  0000'CF    90  0406   893          MOVB    W^MAC$GB_MODE,R0    ;GET BASIC_REF MODE
         00    50   91  040B   894          CMPB    R0,#ADM$_LITERAL    ;SURELY LITERAL MODE IS ILLEGAL
               08   12  040E   895          BNEQ    20$                 ;NO ERROR MESSAGE TODAY
                    0410   896 10$:    $MAC_ERR MAYNOTINDX             ; Set message code
             FBE8'  30  0415   897          BSBW    MAC$ERRORPT         ;REPORT ERROR TO PASS 2
     56  FFFC'CF47   D0  0418   898 20$:    MOVL    W^MAC$AL_VALSTACK-4[R7],R6 ;GET INDEX REGISTER NUMBER
           0F    56   91  041E   899          CMPB    R6,#REG$_PC         ;PC IS AN ILLEGAL REGISTER
                 16   13  0421   900          BEQL    30$                 ;IF EQL ILLEGAL INDEX REG
       0000'CF   56   91  0423   901          CMPB    R6,W^MAC$GB_REG     ;SAME AS BASIC_REF REGISTER?
                 17   12  0428   902          BNEQ    40$                 ;IF NEQ NO
           07    50   91  042A   903          CMPB    R0,#ADM$_REGAUTODEC ;YES--IS MODE -(RX)[RX]
                 0A   13  042D   904          BEQL    30$                 ;IF EQL ILLEGAL
           08    50   91  042F   905          CMPB    R0,#ADM$_REGAUTOINC ;IS MODE (RX)+[RX]
                 05   13  0432   906          BEQL    30$                 ;IF EQL YES
           09    50   91  0434   907          CMPB    R0,#ADM$_DFRAUTOINC ;IS MODE @(RX)+[RX]
                 08   12  0437   908          BNEQ    40$                 ;IF NEQ NO
                    0439   909 30$:    $MAC_ERR ILLINDXREG             ; Yes--set message code
             FBBF'  30  043E   910          BSBW    MAC$ERRORPT         ;ISSUE MESSAGE TO PASS 2
     10  6B   2C   E1  0441   911 40$:    bbc     #FLG$V_OPTVFLIDX,(r11),60$ ; Occasionally, the presence of
           50    02   80  0445   912          addb2   #2,r0               ; indexing forces de-optimization.
           0E    50   91  0448   913          cmpb    r0,#ADM$_LONG_DISP  ; If we de-optimize to Longword Disp.
                 04   12  044B   914          bnequ   50$                 ; the increase is two bytes, otherwise
                    044D   915          $inc_pc                         ; the increase is only one byte.
                    0451   916 50$:    $inc_pc
       0000'CF    50   90  0455   917 60$:    MOVB    r0,W^MAC$GB_IMODE   ;MOVE REGISTERS AND MODES
       0000'CF    04   90  045A   918          MOVB    #ADM$_INDEX,W^MAC$GB_MODE ;SET INDEX MODE
0000'CF    0000'CF   90  045F   919          MOVB    W^MAC$GB_REG,W^MAC$GB_IREG ;...
       0000'CF    56   90  0466   920          MOVB    R6,W^MAC$GB_REG     ;STORE INDEX REGISTER VALUE
                    046B   921          $INC_PC                         ;ALLOW ROOM FOR INDEX REG
```

05 046F 922 RSB

```
                     0470   924          .SBTTL  SPESHL_MODE_CHK
                     0470   925
                     0470   926 ;++
                     0470   927 ; FUNCTIONAL DESCRIPTION:
                     0470   928 ;
                     0470   929 ;       THIS ROUTINE CHECKS FOR SPECIAL MODES THAT MUST BE HANDLED
                     0470   930 ;       SPECIALLY SINCE THEY ARE USED AS ADDRESSES IN INDEX MODE.
                     0470   931 ;       LITERAL MODE REFERENCES ARE CHANGED TO IMMEDIATE MODE
                     0470   932 ;       REFERENCES AND THE PC IS UPDATED BY THE SIZE OF THE OPERAND.
                     0470   933 ;       IMMEDIATE MODE REFERENCES ARE FORCED TO LONGWORD SIZE.
                     0470   934 ;
                     0470   935 ; INPUTS:
                     0470   936 ;
                     0470   937 ;       MAC$GB_MODE             MODE OF OPERAND
                     0470   938 ;
                     0470   939 ; OUTPUTS:
                     0470   940 ;
                     0470   941 ;       MAC$GB_MODE             NEW MODE OF OPERAND
                     0470   942 ;       MAC$GB_REG              SET TO PC IF ORIGINAL REF WAS LITERAL
                     0470   943 ;       MAC$GL_PC               UPDATED.
                     0470   944 ;
                     0470   945 ;--
                     0470   946
                     0470   947 SPESHL_MODE_CHK:
50   0000'CF    90   0470   948          MOVB    W^MAC$GB_MODE,R0        ;GET THE MODE SPECIFIER
           16    12   0475   949          BNEQ    10$                     ;BRANCH IF NOT LITERAL (0)
0000'CF    01    90   0477   950          MOVB    #ADM$_IMMEDIATE,W^MAC$GB_MODE ;YES--SET MODE TO IMEDIATE
0000'CF    0F    90   047C   951          MOVB    #REG$_PC,W^MAC$GB_REG   ;USE (PC)+
50   0000'CF    9A   0481   952          MOVZBL  W^MAC$GL_OPSIZE,R0      ;GET OPERAND SIZE
                     0486   953          $INC_PC R0                      ;UPDATE PC
           18    11   048B   954          BRB     20$                     ;EXIT ROUTINE
           50    97   048D   955 10$:     DECB    R0                      ;IS MODE IMMEDIATE (1)?
           14    12   048F   956          BNEQ    20$                     ;IF NEQ NO
50   0000'CF    9A   0491   957          MOVZBL  W^MAC$GL_OPSIZE,R0      ;YES--GET OPERAND SIZE
                     0496   958          $DEC_PC R0                      ;SUBTRACT DEFAULT SIZE
                     049B   959          $INC_PC #4                      ;FORCE TO LONGWORD
0000'CF    04    90   04A0   960          MOVB    #4,W^MAC$GL_OPSIZE      ;...
           05    05   04A5   961 20$:     RSB
                     04A6   962
                     04A6   963          .END
```

```
$COUNT              =  0000003B
AB                  =  00000001
AD                  =  0000C008
ADM$_ABSOLUTE       =  00000002
ADM$_BYTE_DISP      =  0000000A
ADM$_DFBYTEDISP     =  0000000B
ADM$_DFLONGDISP     =  0000000F
ADM$_DFRAUTOINC     =  00000009
ADM$_DFWORDDISP     =  0000000D
ADM$_IMMEDIATE      =  00000001
ADM$_INDEX          =  00000004
ADM$_LITERAL        =  00000000
ADM$_LONG_DISP      =  0000000E
ADM$_MAXMOD         =  0000000F
ADM$_PIC            =  00000003
ADM$_REGAUTODEC     =  00000007
ADM$_REGAUTOINC     =  00000008
ADM$_REGISTER       =  00000005
ADM$_RRIND          =  00000006
ADM$_WORD_DISP      =  0000000C
AF                  =  00008004
AG                  =  0000A008
AH                  =  00009010
AINC1               000002F6  RG  04
AINC2               0000030B  RG  04
AINC3               00000306  RG  04
AL                  =  00000004
AO                  =  00000010
AQ                  =  00000008
ARG$K_SIZE          =  000003E8
AUD$K_SIZE          =  00000010
AUTOD               00000080  RG  04
AUTOI               00000055  RG  04
AUTOI3              000003EF  RG  04
AUTOI1              00000075  RG  04
AW                  =  00000002
B                   =  00000001
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                 =  00000001
CR                  =  0000000D
D                   =  0000C008

DIRREF              00000000  RG  04
DISP                00000055  RG  04
DISPI               0000008F  RG  04
DISPL1              000000DB  RG  04
DISPL2              0000016E  RG  04
DISPL3              00000228  RG  04
DISPL4              00000230  RG  04
DISPL5              00000238  RG  04
DISPL6              00000262  RG  04
DISPL7              0000026A  RG  04
DISPL8              00000272  RG  04
DISPLX_EXIT         00000257  R   04
DISPPC              00000184  R   04
ENB$G_ABSADDR       ********    X 04
ERR                 =  00000000
F                   =  00008004
FF                  =  0000000C
FLG$M_ALLCHR        =  00000001
FLG$M_BOL           =  00000002
FLG$M_CHKLPND       =  00100000
FLG$M_COMPEXPR      =  00000004
FLG$M_CONT          =  00000008
FLG$M_CRF           =  40000000
FLG$M_CRSEEN        =  00000001
FLG$M_DATRPT        =  00000010
FLG$M_DBGOUT        =  00004000
FLG$M_DLIMSTR       =  00008000
FLG$M_ENDMCH        =  00000020
FLG$M_EVALEXPR      =  00000040
FLG$M_EXPOPT        =  00000080
FLG$M_EXTERR        =  00010000
FLG$M_EXTWRN        =  00020000
FLG$M_FIRSTLN       =  00000200
FLG$M_IFSTAT        =  00800000
FLG$M_IIF           =  00400000
FLG$M_INSERT        =  00000100
FLG$M_IRPC          =  20000000
FLG$M_LEXOP         =  00000002
FLG$M_LSTXST        =  00000200
FLG$M_MAC2COL       =  00000800
FLG$M_MACL          =  00000800
FLG$M_MACLTB        =  08000000
FLG$M_MACTXT        =  00010000
FLG$M_MEBLST        =  00001000
FLG$M_MOREARG       =  00002000
FLG$M_MOREINP       =  00000008
FLG$M_NEWPND        =  00000400
FLG$M_NOREF         =  00000000
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$V_XCRF      =  0000001F
G               =  0000A008
GENLDS             0000027A RG     04
H               =  00009010
HASHSZ          =  0000007F
HYPHEN          =  0000002D
ILL_REG_CHK        000000B7 R      04
INDEX              00000406 RG     04
INP$K_BUFSIZ    =  000003E8
INT$K_BUFSIZ    =  000013F4
INT$K_BUFWRN    =  00001390
INT$_ADD        =  00000001
INT$_AND        =  00000002
INT$_ASH        =  00000003
INT$_ASN        =  0000000C
INT$_AUGPC      =  0000000D
INT$_BDST       =  0000000E
INT$_CHKL       =  0000000F
INT$_DIV        =  00000004
INT$_END        =  00000010
INT$_EPT        =  00000011
INT$_ERR        =  00000012
INT$_ETX        =  00000013
INT$_FNEWL      =  00000014
INT$_ILG        =  00000000
INT$_INFO       =  0000003A
INT$_LGLAB      =  00000015
INT$_MACL       =  00000016
INT$_MUL        =  00000005
INT$_NEG        =  00000006
INT$_NEWL       =  00000017
INT$_NEWP       =  00000018
INT$_NOT        =  00000007
INT$_OP         =  00000019
INT$_OR         =  00000008
INT$_PRIL       =  0000001A
INT$_PRT        =  0000001B
INT$_PSECT      =  0000001C
INT$_REDEF      =  0000001D
INT$_REF        =  0000001E
INT$_REST       =  0000001F
INT$_SAME       =  00000009
INT$_SAVE       =  00000020
INT$_SBTTL      =  00000021
INT$_SETFLAG    =  00000022
INT$_SETLONG    =  00000023
INT$_SPIC       =  00000024
INT$_SPID       =  00000025
INT$_STIB       =  00000026
INT$_STIL       =  00000028
INT$_STIW       =  00000027
INT$_STKEPT     =  00000029
INT$_STKG       =  0000002A
INT$_STKL       =  0000002B
INT$_STKPC      =  0000002C
INT$_STKS       =  0000002D
INT$_STOB       =  00000034
INT$_STOL       =  0000002E
INT$_STOW       =  00000035
INT$_STRB       =  0000002F
INT$_STRL       =  00000031
INT$_STRSB      =  00000032
INT$_STRSW      =  00000033
INT$_STRW       =  00000030
INT$_STSB       =  00000036
INT$_STSW       =  00000037
INT$_SUB        =  0000000A
INT$_SUME       =  00000039
INT$_WRN        =  00000038
INT$_XOR        =  0000000B
L               =  00000004
LITERL             00000298 RG     04
LST$K_BUFSIZ    =  00000086
LST$K_L_P_PAGE  =  0000003C
LST$K_TITLE_SIZ=   00000028
MAC$AL_VALSTACK    ********   X    04
MAC$ERRORPT        ********   X    04
MAC$ERRORPX        ********   X    04
MAC$GB_IMODE       ********   X    04
MAC$GB_IREG        ********   X    04
MAC$GB_MODE        ********   X    04
MAC$GB_RDXNDX      ********   X    04
MAC$GB_REG         ********   X    04
MAC$GB_VAL1        ********   X    04
MAC$GK_ZERO        ********   X    04
MAC$GL_ABSFLAG     ********   X    04
MAC$GL_DFPC_DSP    ********   X    04
MAC$GL_EXPEND      ********   X    04
MAC$GL_EXPOPVL1    ********   X    04
MAC$GL_HIGH_32     ********   X    04
MAC$GL_MOPPTR      ********   X    04
MAC$GL_OPSIZE      ********   X    04
MAC$GL_PC          ********   X    04
MAC$GL_VAL3        ********   X    04
MAC$GL_VALUE       ********   X    04
MAC$GQ_HIGH_64     ********   X    04
MAC$INTOUT_T_LW    ********   X    04
MAC$INTOUT_2_LW    ********   X    04
MAC$OPTIMIZEXPR    ********   X    04
MAC$_ILLINDXREG=   007D90E2
MAC$_ILLREGHERE=   007D911A
MAC$_MAYNOTINDX=   007D9142
MAC_SUBSYS      =  0000007D
MB              =  00000041
MD              =  0000C048
MF              =  00008044
MG              =  0000A048
MH              =  00009050
ML              =  00000044
MO              =  00000050
MQ              =  00000048
MW              =  00000042
O               =  00000010
OBJ$K_BUFSIZ    =  00000200
OPD$M_ADDR      =  00000000
OPD$M_BB        =  000000A1
OPD$M_BW        =  000000C2
OPD$M_D_FLOAT   =  0000C000
OPD$M_F_FLOAT   =  00008000
OPD$M_G_FLOAT   =  0000A000
OPD$M_H_FLOAT   =  00009000
OPD$M_MODE      =  000003E0
OPD$M_MODIFY    =  00000040
OPD$M_NOT_32F   =  00007000
OPD$M_READ      =  00000020
OPD$M_VIELD     =  00000080
OPD$M_WRITE     =  00000060
OPD$S_MODE      =  00000005
OPD$S_SIZE      =  00000005
OPD$V_D_FLOAT   =  0000000E
OPD$V_F_FLOAT   =  0000000F
OPD$V_G_FLOAT   =  0000000D
OPD$V_H_FLOAT   =  0000000C
OPD$V_MODE      =  00000005
OPD$V_SIZE      =  00000000
OPF$M_LASTOPR   =  00002000
OPF$M_OPTEXP    =  00001000
OPF$V_LASTOPR   =  0000000D
OPF$V_OPTEXP    =  0000000C
PC_DISPL_DISP      00000000 R      03
PSC$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$M_PIC       = 00000001          SYM$B_NAME        00000004          WL              = 00000064
PSC$M_QUAD      = 00004C00          SYM$B_SEG         0000000C          WO              = 00000070
PSC$M_RD        = 00000080          SYM$B_TOKEN       0000000B          WQ              = 00000068
PSC$M_REL       = 00000008          SYM$K_BLKSIZ      0000000D          WW              = 00000062
PSC$M_SHR       = 00000020          SYM$K_MAXLEN    = 0000001F          X               = 00000010
PSC$M_USR       = FFFFFFFD          SYM$K_TWOCOL    = 00000010          X1              = 00000400
PSC$M_VEC       = 00000200          SYM$L_LINK        00000000          X2              = 0000000F
PSC$M_WORD      = 00004400          SYM$L_VAL         00000005
PSC$M_WRT       = 00000180          SYM$M_ABS       = 00000010
PSC$S_ALIGNMENT= 00000004           SYM$M_ASN       = 00000100
PSC$V_ALIGNFLG  = 0000000E          SYM$M_CRFO      = 00002000
PSC$V_ALIGNMENT= 0000000A           SYM$M_DEBUG     = 00000020
PSC$V_EXE       = 00000006          SYM$M_DEF       = 00000001
PSC$V_GBL       = 00000004          SYM$M_DELMAC    = 00000200
PSC$V_LIB       = 00000001          SYM$M_EPT       = 00000200
PSC$V_OVR       = 00000002          SYM$M_EXTRN     = 00000008
PSC$V_PIC       = 00000000          SYM$M_GLOBL     = 00000004
PSC$V_RD        = 00000007          SYM$M_LOCAL     = 00000040
PSC$V_REL       = 00000003          SYM$M_ODBG      = 00000400
PSC$V_SHR       = 00000005          SYM$M_REF       = 00000080
PSC$V_VEC       = 00000009          SYM$M_RELPSECT  = 00000800
PSC$V_WRT       = 00000008          SYM$M_SUPR      = 00004000
PSC$W_FLAG        00000009          SYM$M_WEAK      = 00000002
PSC$W_OPTIONS     0000000D          SYM$M_XCRF      = 00001000
Q               = 00000008          SYM$V_ABS       = 00000004
RB              = 00000021          SYM$V_ASN       = 00000008
RD              = 0000C028          SYM$V_CRFO      = 0000000D
RDEFER            00000066 RG 04    SYM$V_DEBUG     = 00000005
RDX$V_BINARY    = 00000000          SYM$V_DEF       = 00000000
RDX$V_DECIMAL   = 00000002          SYM$V_DELMAC    = 00000009
RDX$V_DOUBLE    = 00000005          SYM$V_EPT       = 00000009
RDX$V_FLOAT     = 00000004          SYM$V_EXTRN     = 00000003
RDX$V_GFLOAT    = 00000006          SYM$V_GLOBL     = 00000002
RDX$V_HEX       = 00000003          SYM$V_LOCAL     = 00000006
RDX$V_HFLOAT    = 00000007          SYM$V_ODBG      = 0000000A
RDX$V_OCTAL     = 00000001          SYM$V_REF       = 00000007
REF1              0000004E RG 04    SYM$V_RELPSECT  = 0000000B
REF2              00000056 RG 04    SYM$V_SUPR      = 0000000E
REF3              0000004E RG 04    SYM$V_WEAK      = 00000001
REF_EXIT          0000004E R  04    SYM$V_XCRF      = 0000000C
REG$_PC         = 0000000F          SYM$W_FLAG        00000009
RF              = 00008024          TAB             = 00000009
RFHD              00000036 R  04    VB              = 00000081
RFHD1             00000027 RG 04    VD              = 0000C088
RFHD16            00000033 RG 04    VF              = 00008084
RFHD2             0000002A RG 04    VG              = 0000A088
RFHD4             0000002D RG 04    VH              = 00009090
RFHD8             00000030 RG 04    VL              = 00000084
RG              = 0000A028          VO              = 00000090
RH              = 00009030          VQ              = 00000088
RL              = 00000024          VW              = 00000082
RO              = 00000030          W               = 00000002
RQ              = 00000028          WB              = 00000061
RW              = 00000022          WD              = 0000C068
SEMI            = 0000003B          WF              = 00008064
SPESHL_MODE_CHK   00000470 R  04    WG              = 0000A068
STB$K_PG_MISS   = 0000000A          WH              = 00009070
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
. BLANK .	00000000 (0.)	01 (1.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
ABS	00000013 (19.)	02 (2.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
MAC$RO_DATA	00000010 (16.)	03 (3.)	NOPIC USR CON REL GBL NOSHR NOEXE RD NOWRT NOVEC LONG
MAC$RO_CODE_P1	000004A6 (1190.)	04 (4.)	NOPIC USR CON REL GBL NOSHR EXE RD NOWRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.06	00:00:01.43
Command processing	103	00:00:00.38	00:00:01.98
Pass 1	238	00:00:04.18	00:00:18.71
Symbol table sort	0	00:00:00.54	00:00:01.13
Pass 2	183	00:00:01.51	00:00:03.59
Symbol table output	39	00:00:00.21	00:00:00.67
Psect synopsis output	1	00:00:00.02	00:00:00.24
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	595	00:00:06.90	00:00:27.75

The working set limit was 1650 pages.
41361 bytes (81 pages) of virtual memory were used to buffer the intermediate code.
There were 30 pages of symbol table space allocated to hold 540 non-local and 53 local symbols.
963 source lines were read in Pass 1, producing 24 object records in Pass 2.
19 pages of virtual memory were used to define 15 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[MACRO.OBJ]MACRO.MLB;1	13
_255DUA28:[SYSLIB]STARLET.MLB;2	3
TOTALS (all libraries)	16

616 GETS were required to define 16 macros.

There were no errors, warnings or information messages.

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

ACTPRI LIS

ARGSCN LIS

BDYSCN LIS

CRFSUB LIS

ACTOPC LIS

ACTSTA LIS

APSECT LIS

CRFDAT LIS

ACTREF LIS

COMPUT LIS