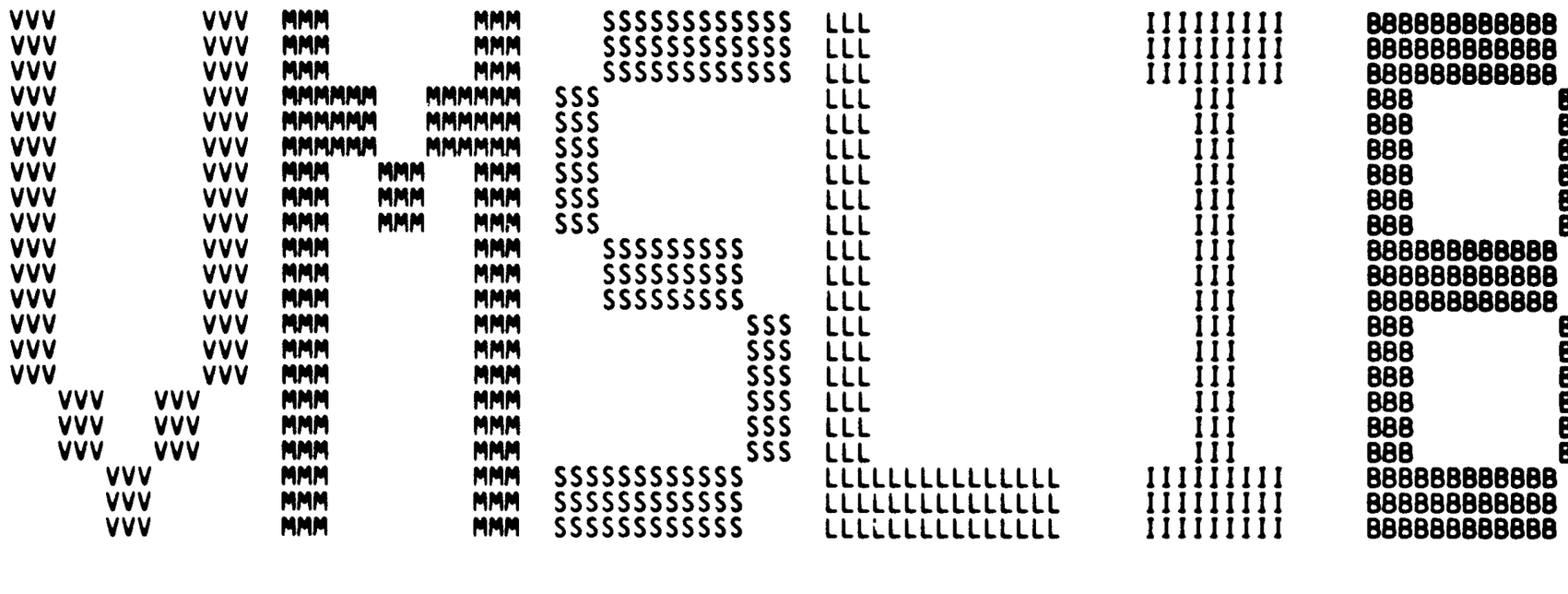
VVV VVV MMM MMM SSSSSSSSSSS LLL IIIIIIIII BBBBBBBBBBB
VVV VVV MMM MMM SSSSSSSSSSS LLL IIIIIIIII BBBBBBBBBBB
VVV VVV MMM MMM SSSSSSSSSSS LLL IIIIIIIII BBBBBBBBBBB
VVV VVV MMMMMM MMMMMM SSS LLL III BBB BBB
VVV VVV MMMMMM MMMMMM SSS LLL III BBB BBB
VVV VVV MMMMMM MMMMMM SSS LLL III BBB BBB
VVV VVV MMM MMM MMM SSS LLL III BBB BBB
VVV VVV MMM MMM MMM SSS LLL III BBB BBB
VVV VVV MMM MMM MMM SSS LLL III BBB BBB
VVV VVV MMM MMM SSSSSSSSS LLL III BBBBBBBBBBB
VVV VVV MMM MMM SSSSSSSSS LLL III BBBBBBBBBBB
VVV VVV MMM MMM SSSSSSSSS LLL III BBBBBBBBBBB
VVV VVV MMM MMM SSS LLL III BBB BBB
VVV VVV MMM MMM SSS LLL III BBB BBB
VVV VVV MMM MMM SSS LLL III BBB BBB
VVV VVV MMM MMM SSS LLL III BBB BBB
VVV VVV MMM MMM SSS LLL III BBB BBB
VVV VVV MMM MMM SSS LLL III BBB BBB
VVV VVV MMM MMM SSSSSSSSSSS LLLLLLLLLLLLLLL IIIIIIIII BBBBBBBBBBB
VVV VVV MMM MMM SSSSSSSSSSS LLLLLLLLLLLLLLL IIIIIIIII BBBBBBBBBBB
VVV MMM MMM SSSSSSSSSSS LLLLLLLLLLLLLLL IIIIIIIII BBBBBBBBBBB

```
LL              IIIIII    BBBBBBBB    MM      MM  EEEEEEEEEE  RRRRRRRR      GGGGGGGG  EEEEEEEEEE
LL              IIIIII    BBBBBBBB    MM      MM  EEEEEEEEEE  RRRRRRRR      GGGGGGGG  EEEEEEEEEE
LL                II      BB      BB  MMMM  MMMM  EE          RR      RR  GG            EE
LL                II      BB      BB  MMMM  MMMM  EE          RR      RR  GG            EE
LL                II      BB      BB  MM  MM  MM  EE          RR      RR  GG            EE
LL                II      BB      BB  MM  MM  MM  EE          RR      RR  GG            EE
LL                II      BBBBBBBB    MM      MM  EEEEEEEE    RRRRRRRR    GG            EEEEEEEE
LL                II      BBBBBBBB    MM      MM  EEEEEEEE    RRRRRRRR    GG            EEEEEEEE
LL                II      BB      BB  MM      MM  EE          RR   RR     GG   GGGGGG   EE
LL                II      BB      BB  MM      MM  EE          RR   RR     GG   GGGGGG   EE
LL                II      BB      BB  MM      MM  EE          RR    RR    GG       GG   EE              ....
LL                II      BB      BB  MM      MM  EE          RR    RR    GG       GG   EE              ....
LLLLLLLLLL      IIIIII    BBBBBBBB    MM      MM  EEEEEEEEEE  RR      RR    GGGGGG     EEEEEEEEEE      ....
LLLLLLLLLL      IIIIII    BBBBBBBB    MM      MM  EEEEEEEEEE  RR      RR    GGGGGG     EEEEEEEEEE      ....

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 LIB$MERGE - Merge Image Activate an Image
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:     VMS-specific library
0000    31 ;
0000    32 ; ABSTRACT:     The routines in this module merge an image into P0 or P1 space.
0000    33 ;
0000    34 ; ENVIRONMENT:
0000    35 ;
0000    36 ; AUTHOR: Kathleen D. Morse , CREATION DATE: 10-Jan-80
0000    37 ;
0000    38 ; MODIFIED BY:
0000    39 ;
0000    40 ;       V03-002 LJK0277         Lawrence J. Kenah       1-May-1984
0000    41 ;               Do not reorder privileged vectors until $IMGFIX has been called.
0000    42 ;               This change reflects the change in the way that privileged
0000    43 ;               vectors are added to the dispatch lists.
0000    44 ;
0000    45 ;       V03-001 LJK0261         Lawrence J. Kenah       8-Feb-1984
0000    46 ;               Allow message and change mode vectors that are part of an
0000    47 ;               image mapped into P1 space to survive image rundown.
0000    48 ;--
```

```
                     0000     50          .SBTTL  DECLARATIONS
                     0000     51 ;
                     0000     52 ; INCLUDE FILES:
                     0000     53 ;
                     0000     54
                     0000     55          $IACDEF                         ;IMAGE ACTIVATION FLAGS
                     0000     56          $PHDDEF                         ;PROCESS HEADER DEFINITIONS
                     0000     57          $OPDEF                          ;SYMBOLIC OPCODE DEFINITIONS
                     0000     58          $SSDEF                          ;SYSTEM SEPVICE STATUS CODES
                     0000     59
                     0000     60 ;
                     0000     61 ; MACROS:
                     0000     62 ;
                     0000     63
                     0000     64 ;
                     0000     65 ; EQUATED SYMBOLS:
                     0000     66 ;
                     0000     67
                     0000     68 ;
                     0000     69 ; OFFSETS (FROM AP) FOR INPUT ARGUMENT LIST FOR LIB$Px_MERGE
                     0000     70 ;
                     0000     71
00000004             0000     72 FILNAM=4                                ; Address of image file name descriptor
00000008             0000     73 DFLTNAM=8                               ; Address of default file name descriptor
0000000C             0000     74 HDRBUF=12                               ; Address of image header buffer
00000010             0000     75 RETADR=16                               ; Address of quadword for range of
                     0000     76                                         ;  virtual addresses actually mapped
                     0000     77
                     0000     78 ;
                     0000     79 ; OWN STORAGE:
                     0000     80 ;
```

```
0000     82          .SBTTL LIB$Px_MERGE - Merge an Image into P0 or P1 Space
0000     83 ;+
0000     84 ; FUNCTIONAL DESCRIPTION:
0000     85 ;
0000     86 ; This routine merges an image into P0 or P1 space.
0000     87 ;
0000     88 ; If called at LIB$P0_MERGE, a simple merged image activation is performed,
0000     89 ; placing the designated image at the high address end of P0 space.
0000     90 ;
0000     91 ; A more complicated sequence of events is required to merge an image into
0000     92 ; the low address end of P1 space. Because the image activator must know the
0000     93 ; exact amount of P1 space needed for the image being merged, LIB$P1_MERGE
0000     94 ; first merges the image into P0 space with the expand region option.  After
0000     95 ; this merge is completed, the routine computes the P1 range that will hold
0000     96 ; the image.  This is done by finding the first free P1 page and computing
0000     97 ; the number of pages just mapped into P0 space.  A P1 virtual address range
0000     98 ; is then computed.  The pages mapped into P0 space are deleted and second
0000     99 ; merge image activate is done.  This time the expand region ortion is turned
0000    100 ; off and the exact P1 range needed is specified.
0000    101 ;
0000    102 ; CALLING SEQUENCE:
0000    103 ;
0000    104 ;       CALLS   #4,LIB$Px_MERGE
0000    105 ;
0000    106 ; INPUT PARAMETERS:
0000    107 ;
0000    108 ;       FILNAM(AP)  = Address of image file name descriptor
0000    109 ;       DFLTNAM(AP) = Address of default file name descriptor
0000    110 ;       HDRBUF(AP)  = Address of 512-byte image header buffer
0000    111 ;       RETADR(AP)  = Address of quadword for range of virtual addresses mapped
0000    112 ;
0000    113 ; IMPLICIT INPUT:
0000    114 ;
0000    115 ;       none
0000    116 ;
0000    117 ; OUTPUT PARAMETERS:
0000    118 ;
0000    119 ;       none
0000    120 ;
0000    121 ; IMPLICIT OUTPUT:
0000    122 ;
0000    123 ;       The address range actually mapped is returned in the RETADR quadword
0000    124 ;       provided as an input parameter.
0000    125 ;
0000    126 ;       The first block of the image header is copied to the 512-byte buffer
0000    127 ;       provided as an input parameter.
0000    128 ;
0000    129 ; COMPLETION CODES:
0000    130 ;
0000    131 ;       R0 low bit set => Image successfully merged
0000    132 ;
0000    133 ;               SS$_NORMAL
0000    134 ;
0000    135 ;       R0 low bit clear => Error occurred while activating image
0000    136 ;
0000    137 ;               SS$_NOPRIV      Entry was as LIB$P1_MERGE and process
0000    138 ;                               does not have CMKRNL privilege.
```

```
0000   139 ;
0000   140 ;                    Various errors returned by $IMGACT and $DELTVA
0000   141 ;
0000   142 ; SIDE EFFECTS:
0000   143 ;
0000   144 ;       If entry is at LIB$P1_MERGE, the permanent portion of P1 space is
0000   145 ;       expanded to accommodate the merged image. The message and change
0000   146 ;       mode vectors are reordered and the reset points are adjusted to
0000   147 ;       allow any such vectors in a P1 image to survive image rundown.
0000   148 ;-
```

```
                                0000   150          .ENABL          LOCAL_BLOCK
                                0000   151
                          00000000     152          .PSECT  _LIB$CODE PIC,SHR,BYTE,RD,EXE,NOWRT
                                0000   153
                                0000   154 LIB$P0_MERGE::
                             00FC  0000   155          .WORD   ^M<R2,R3,R4,R5,R6,R7>   ;REGISTER SAVE MASK
                       57    30   D0  0002   156          MOVL    #<IAC$M_MERGE!IAC$M_EXPREG>,R7 ;IMAGE ACTIVATION FLAGS
                             09   11  0005   157          BRB     10$                     ;JOIN COMMON CODE PATH
                                0007   158
                                0007   159 LIB$P1_MERGE::
                             00FC  0007   160          .WORD   ^M<R2,R3,R4,R5,R6,R7>   ;REGISTER SAVE MASK
                    57  00000070  8F   D0  0009   161          MOVL    #<IAC$M_MERGE!IAC$M_EXPREG!IAC$M_P1MERGE>,R7
                 7E    03FF  8F   3C  0010   162 10$:     MOVZWL  #^X3FF,-(SP)            ;END VA FOR BLUEPRINT P0 VA RANGE
                 7E    0200  8F   3C  0015   163          MOVZWL  #^X200,-(SP)            ;START VA FOR BLUEPRINT P0 VA RANGE
                       54   5E   D0  001A   164          MOVL    SP,R4                   ;ADR OF INPUT VA RANGE
                            7E   7C  001D   165          CLRQ    -(SP)                   ;RETURN VA RANGE
                       55   5E   D0  001F   166          MOVL    SP,R5                   ;ADR OF RETURN VA RANGE
                            7E   7C  0022   167          CLRQ    -(SP)                   ;PLACE TO HOLD IDENT
                       56   5E   D0  0024   168          MOVL    SP,R6                   ;ADR OF IDENT QUADWORD
                                0027   169          $IMGACT_S       -                       ;P0 MERGE IMAGE ACTIVATE CALL
                                0027   170                  NAME=@FILNAM(AP),-              ;FILE NAME
                                0027   171                  DFLNAM=@DFLTNAM(AP),-           ;DEFAULT FILE NAME
                                0027   172                  HDRBUF=@HDRBUF(AP),-            ;IMAGE HEADER BUFFER ADDRESS
                                0027   173                  IMGCTL=R7,-                     ;IMAGE ACTIVATION FLAGS
                                0027   174                  INADR=(R4),-                    ;INPUT BLUEPRINT P0 RANGE
                                0027   175                  RETADR=(R5),-                   ;RETURN VA RANGE
                                0027   176                  IDENT=(R6)                      ;NO IDENT PARAMETER
                          79 50   E9  0041   177          BLBC    R0,IMGACT_ERR           ;BR IF ERROR IN $IMGACT
                    4C 57   06   E1  0044   178          BBC     #IAC$V_P1MERGE,R7,50$   ;ALL DONE IF P0 MERGE
                       52   65   7D  0048   179          MOVQ    (R5),R2                 ;R2=START VA, R3=END VA  (MAPPED)
                       53   52   C2  004B   180          SUBL2   R2,R3                   ;GET # OF BYTES MAPPED
                    50   00000000'9F   D0  004E   181          MOVL    @#CTL$GL_PHD,R0         ;GET ADR OF PROCESS HEADER
              04 A4    30 A0   000001FF  8F   C1  0055   182          ADDL3   #^X1FF,PHD$L_FREP1VA(R0),4(R4) ;GET END VA IN P1 SPACE
                    64   04 A4   53   C3  005F   183          SUBL3   R3,4(R4),(R4)           ;GET START VA IN P1 SPACE
                                0064   184          $DELTVA_S       -                       ;DELETE P0 VIRTUAL ADDRESS RANGE MAPPED
                                0064   185                  INADR=(R5)                      ;VA RANGE RETURNED BY $IMGACT
                          48 50   E9  0071   186          BLBC    R0,100$                 ;BRANCH IF ERROR IN $DELTVA
                                0074   187          ASSUME  IAC$V_EXPREG LE 7       ;BIT MUST BE IN LOW-ORDER BYTE
                       57   20   8A  0074   188          BICB2   #IAC$M_EXPREG,R7        ;TURN OFF EXPREG BIT BEFORE P1 ACTIVATE
                                0077   189          $IMGACT_S       -                       ;NOW ACTIVATE IMAGE INTO P1 SPACE
                                0077   190                  NAME=@FILNAM(AP),-              ;FILE NAME
                                0077   191                  DFLNAM=@DFLTNAM(AP),-           ;DEFAULT FILE NAME
                                0077   192                  HDRBUF=@HDRBUF(AP),-            ;IMAGE HEADER BUFFER ADDRESS
                                0077   193                  IMGCTL=R7,-                     ;IMAGE ACTIVATION FLAGS
                                0077   194                  INADR=(R4),-                    ;INPUT EXACT P1 RANGE
                                0077   195                  RETADR=(R5),-                   ;RETURN VA RANGE
                                0077   196                  IDENT=(R6)                      ;NO IDENT PARAMETER
                          29 50   E9  0091   197          BLBC    R0,IMGACT_ERR           ;BRANCH IF ERROR IN $IMGACT
                                0094   198 50$:     $IMGFIX_S                       ;PERFORM ADDRESS RELOCATION
                          1F 50   E9  009B   199          BLBC    R0,IMGACT_ERR           ;QUIT IF ERROR OCCURS
                    13 57   06   E1  009E   200          BBC     #IAC$V_P1MERGE,R7,75$   ;ALL DONE IF P0 MERGE
                            55   DD  00A2   201          PUSHL   R5                      ;PASS RETADR ARRAY TO KERNEL MODE
                            01   DD  00A4   202          PUSHL   #1                      ;ONE ARGUMENT FOR ROUTINE
                                00A6   203          $CMKRNL_S -                     ;SET A NEW CONTROL REGION BASE
                                00A6   204                  ROUTIN=REORDER_VECTOR,- ; AND MANIPULATE VECTORS
                                00A6   205                  ARGLST=(SP)
                    10 BC   65   7D  00B5   206 75$:     MOVQ    (R5),@RETADR(AP)        ;SET RETURN VA RANGE ACTUALLY MAPPED
```

```
                50   01    9A  00B9    207          MOVZBL  #SS$_NORMAL,R0          ;REPORT SUCCESSFUL MERGE IMAGE ACTIVATE
                                04  00BC    208 100$:    RET                         ; AND RETURN TO CALLER
                                    00BD    209
                                    00BD    210 IMGACT_ERR:
          51    65   01    C1  00BD    211          ADDL3   #1,(R5),R1          ;ANYTHING AT ALL MAPPED?
                     F9    13  00C1    212          BEQL    100$                ;BR IF NOTHING MAPPED (SKIP $DELTVA)
                     50    DD  00C3    213          PUSHL   R0                  ;REMEMBER RETURN STATUS CODE
                                00C5    214          $DELTVA_S       -           ;DELETE P0 VIRTUAL ADDRESS RANGE MAPPED
                                00C5    215                  INADR=(R5)         ;VA RANGE RETURNED BY $IMGACT
                50   8E    D0  00D2    216          MOVL    (SP)+,R0            ;RESTORE ORIGINAL ERROR CODE
                     E5    11  00D5    217          BRB     100$                ;JOIN COMMON CODE
                                00D7    218          .DSABL  LSB
```

```
                        00D7   220          .SUBTITLE       REORDER_VECTOR - Reorder Privileged Vector List
                        00D7   221 ;+
                        00D7   222 ; Functional Description:
                        00D7   223 ;
                        00D7   224 ;       This routine scans the various privileged vector lists and reorders
                        00D7   225 ;       each list so that the vectors that point into the image that was just
                        00D7   226 ;       activated into P1 space precede the other vectors. In addition, the
                        00D7   227 ;       cells that contain the reset points are altered to locate the first
                        00D7   228 ;       vector that is NOT in the image just mapped.
                        00D7   229 ;
                        00D7   230 ;       The net result of this change is that the vectors associated with an
                        00D7   231 ;       image permanently mapped into P1 space survive image rundown.
                        00D7   232 ;
                        00D7   233 ; Input Parameter:
                        00D7   234 ;
                        00D7   235 ;       4(AP) - Address of two-longword array that describes the address
                        00D7   236 ;               range into which the image was just mapped.
                        00D7   237 ;
                        00D7   238 ; Implicit Input:
                        00D7   239 ;
                        00D7   240 ;       CTL$A_DISPVEC - The address of two pages devoted to dispatch vectors
                        00D7   241 ;               for kernel, exec, rundown, and message sections.
                        00D7   242 ;
                        00D7   243 ;       IAC$AW_VECRESET - The address of a four-word array that contains the
                        00D7   244 ;               reset points for each of the four dispatch tables.
                        00D7   245 ;
                        00D7   246 ; Output Parameters:
                        00D7   247 ;
                        00D7   248 ;       none
                        00D7   249 ;
                        00D7   250 ; Implicit Ou put:
                        00D7   251 ;
                        00D7   252 ;       If any of the addresses in any of the vectors below the reset point
                        00D7   253 ;       are located in the image just mapped into P1 space, these vectors are
                        00D7   254 ;       switched around and the reset point modified so that the vectors
                        00D7   255 ;       survive image rundown.
                        00D7   256 ;
                        00D7   257 ; Calling Sequence:
                        00D7   258 ;
                        00D7   259 ;       $CMKRNL routin = REORDER_VECTOR
                        00D7   260 ;-
                        00D7   261
                0000009F 00D7   262 ABSOLUTE_MODE = ^X9F
                00009F16 00D7   263 JSB_ABSOLUTE = ABSOLUTE_MODE@8 ! OP$_JSB ; Opcode and mode for JSB @#address
                00000006 00D7   264 VECTOR_SIZE  = 6                        ; Six bytes in such an instruction
                00000100 00D7   265 VECTOR_PAGE  = 256                      ; One half page for each vector
                00000004 00D7   266 VECTOR_COUNT = 4                        ; There are four of them
                         00D7   267
                         00D7   268 REORDER_VECTOR:
                    003C 00D7   269         .WORD   ^M<R2,R3,R4,R5>
          52    04 AC    D0 00D9   270         MOVL    4(AP),R2                ; Save address range pointer in R2
    54    00000000'9F    DE 00DD   271         MOVAL   @#CTL$A_DISPVEC,R4      ; Get base address of vector pages
                  55    D4 00E4   272         CLRL    R5                      ; R5 is an index on the section type
                         00E6   273
                         00E6   274 ; The outer loop is executed for each different type of dispatch vector
                         00E6   275
50  00000000'9F45    3C 00E6   276 10$:    MOVZWL  @#IAC$AW_VECRESET[R5],- ; Get initial reset point
```

```
                                00EE    277                     R0
                          50    54    C0  00EE    278          ADDL2   R4,R0                  ; Convert offset to an address
                        51  02    A0    9E  00F1    279          MOVAB   2(R0),R1                ; Point R1 to first JSB destination
                                          00F5    280
                                          00F5    281 ; The inner loop is executed for each vector in the page
                                          00F5    282
                    9F16 8F       80    B1  00F5    283 20$:     CMPW    (R0)+,#JSB_ABSOLUTE     ; First two bytes of JSB @#address?
                                  21    12  00FA    284          BNEQ    50$                     ; Nope. Must be end of list
                              62  60    D1  00FC    285          CMPL    (R0),(R2)               ; Is address below lower limit?
                                  17    1F  00FF    286          BLSSU   40$                     ; Yup, skip to next vector
                              04  A2    60    D1  0101    287          CMPL    (R0),4(R2)              ; Is address above upper limit?
                                  11    1A  0105    288          BGTRU   40$                     ; Yup, skip to next vector
                                  51    50  D1  0107    289          CMPL    R0,R1                   ; Does RESET locate the same place?
                                        09  13  010A    290          BEQL    30$                     ; Yes. No exchange necessary
                                            010C    291
                                            010C    292 ; The next three instructions exchange the two vectors located by R0 and R1.
                                            010C    293
                                        53  61  D0  010C    294          MOVL    (R1),R3                 ; Save a vector
                                        61  60  D0  010F    295          MOVL    (R0),(R1)               ; Move the new vector up the list
                                        60  53  D0  0112    296          MOVL    R3,(R0)                 ; Restore the saved vector in a new home
                                            0115    297
                                        51  06  C0  0115    298 30$:     ADDL    #VECTOR_SIZE,R1         ; Advance the reset point
                                            0118    299
                                        50  04  C0  0118    300 40$:     ADDL    #4,R0                   ; Skip over address
                                            D8  11  011B    301          BRB     20$                     ; ... and get the next vector
                                            011D    302
                                            011D    303 ; At this point, we have successfully traversed a vector page. We need
                                            011D    304 ; to establish a new reset point.
                                            011D    305
                                        51  54  C2  011D    306 50$:     SUBL    R4,R1                   ; Get new reset offset
                00000000'EF45           51  02  A3  0120    307          SUBW3   #2,R1,IAC$AW_VECRESET[R5]       ; Store it minus bias
                              54  00000100  8F  C0  0129    308          ADDL    #VECTOR_PAGE,R4         ; Advance base of next "page"
                                    B2  55  04  F2  0130    309          AOBLSS  #VECTOR_COUNT,R5,10$    ; Have we done all vector sections?
                                            0134    310
                              00000000'GF   62  D0  0134    311          MOVL    (R2),G^CTL$GL_CTLBASVA  ; Set new control region base address
                                        50  01  3C  013B    312          MOVZWL  #SS$_NORMAL,R0          ; Indicate success
                                            04  013E    313          RET                             ; ... and return
                                            013F    314
                                            013F    315          .END
```

```
$$T1                            = 00000000
ABSOLUTE_MODE                   = 0000009F
CTL$A_DISPVEC                     ********    X   02
CTL$GL_CTLBASVA                   ********    X   02
CTL$GL_PHD                        ********    X   02
DFLTNAM                         = 00000008
FILNAM                          = 00000004
HDRBUF                          = 0000000C
IAC$AW_VECRESET                   ********    X   02
IAC$M_EXPREG                    = 00000020
IAC$M_MERGE                     = 00000010
IAC$M_P1MERGE                   = 00000040
IAC$V_EXPREG                    = 00000005
IAC$V_P1MERGE                   = 00000006
IMGACT_ERR                        000000BD R      02
JSB_ABSOLUTE                    = 00009F16
LIB$PC_MERGE                      00000000 RG     02
LIB$P1_MERGE                      00000007 RG     02
OP$_JSB                         = 00000016
PHD$L_FREP1VA                   = 00000030
REORDER_VECTOR                    000000D7 R      02
RETADR                          = 00000010
SS$_NORMAL                      = 00000001
SYS$CMKRNL                        ********   GX   02
SYS$DELTVA                        ********   GX   02
SYS$IMGACT                        ********   GX   02
SYS$IMGFIX                        ********   GX   02
VECTOR_COUNT                    = 00000004
VECTOR_PAGE                     = 00000100
VECTOR_SIZE                     = 00000006
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
_LIB$CODE	0000013F (319.)	02 (2.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC BYTE

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	33	00:00:00.09	00:00:00.56
Command processing	145	00:00:00.52	00:00:02.49
Pass 1	287	00:00:08.46	00:00:19.16
Symbol table sort	0	00:00:01.43	00:00:03.03
Pass 2	70	00:00:01.65	00:00:03.44
Symbol table output	5	00:00:00.05	00:00:00.05
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	544	00:00:12.24	00:00:28.76

```
The working set limit was 1200 pages.
48896 bytes (96 pages) of virtual memory were used to buffer the intermediate code.
There were 50 pages of symbol table space allocated to hold 953 non-local and 9 local symbols.
315 source lines were read in Pass 1, producing 10 object records in Pass 2.
18 pages of virtual memory were used to define 17 macros.
```

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

Macro library name	Macros defined
_255DUA28:[SYS.OBJ]LIB.MLB;1	1
_255DUA28:[SYSLIB]STARLET.MLB;2	13
TOTALS (all libraries)	14

```
1061 GETS were required to define 14 macros.

There were no errors, warnings or information messages.

MACRO/DISA=TRACE/LIS=LIS$:LIBMERGE/OBJ=OBJ$:LIBMERGE MSRC$:LIBMERGE/UPDATE=(ENH$:LIBMERGE)+EXECML$/LIB
```

READOBJ
LIS

SCREEN
LIS

LIBEXECLI
LIS

MOUNTMSG
LIS

PASMSG
LIS

LIBFILPRO
LIS

LIBNETCON
LIS

LIBEXTCON
LIS

LIBMERGE
LIS

LNKMSG
LIS

REPORTIOE
LIS

LIBUNLFIL
LIS

LIBFIDNAM
LIS

MATCHNAME
LIS

OPCMSG
LIS

PLIMSG
LIS