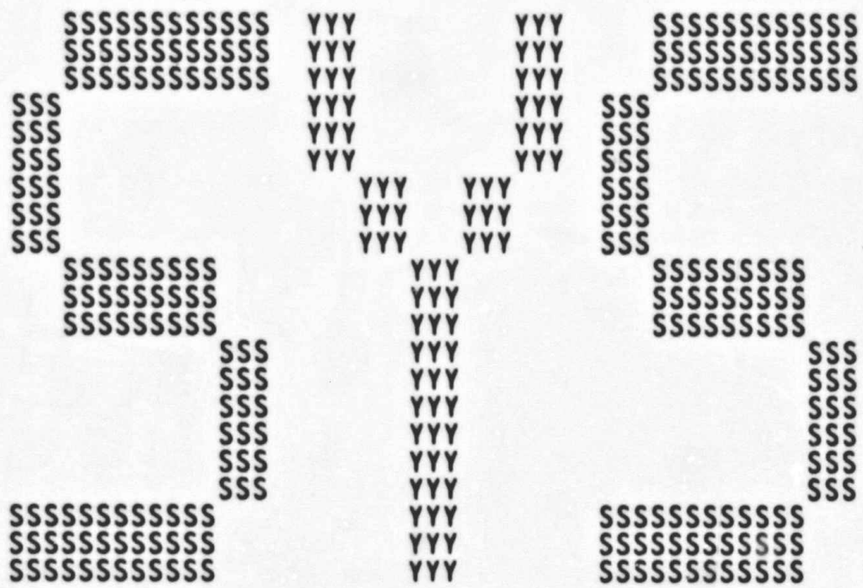
SSSSSSSSSSS YYY YYY SSSSSSSSSSS
SSSSSSSSSSS YYY YYY SSSSSSSSSSS
SSSSSSSSSSS YYY YYY SSSSSSSSSSS
SSS YYY YYY SSS
SSS YYY YYY SSS
SSS YYY YYY SSS
SSS YYY YYY SSS
SSS YYY YYY SSS
SSS YYY YYY SSS
SSSSSSSSS YYY SSSSSSSSS
SSSSSSSSS YYY SSSSSSSSS
SSSSSSSSS YYY SSSSSSSSS
SSS YYY SSS
SSS YYY SSS
SSS YYY SSS
SSS YYY SSS
SSS YYY SSS
SSS YYY SSS
SSSSSSSSSSS YYY SSSSSSSSSSS
SSSSSSSSSSS YYY SSSSSSSSSSS
SSSSSSSSSSS YYY SSSSSSSSSSS

FILEID**LOADMREG

```
LL              OOOOOO      AAAAAA    DDDDDDDD    MM      MM  RRRRRRRR    EEEEEEEEEE    GGGGGGGG
LL              OOOOOO      AAAAAA    DDDDDDDD    MM      MM  RRRRRRRR    EEEEEEEEEE    GGGGGGGG
LL            OO      OO  AA      AA  DD      DD  MMMM  MMMM  RR      RR  EE            GG
LL            OO      OO  AA      AA  DD      DD  MMMM  MMMM  RR      RR  EE            GG
LL            OO      OO  AA      AA  DD      DD  MM  MM  MM  RR      RR  EE            GG
LL            OO      OO  AA      AA  DD      DD  MM  MM  MM  RR      RR  EE            GG
LL            OO      OO  AA      AA  DD      DD  MM      MM  RRRRRRRR    EEEEEEEE      GG
LL            OO      OO  AA      AA  DD      DD  MM      MM  RRRRRRRR    EEEEEEEE      GG
LL            OO      OO  AAAAAAAAAA  DD      DD  MM      MM  RR  RR      EE            GG  GGGGGG
LL            OO      OO  AAAAAAAAAA  DD      DD  MM      MM  RR  RR      EE            GG  GGGGGG
LL            OO      OO  AA      AA  DD      DD  MM      MM  RR    RR    EE            GG      GG      ....
LL            OO      OO  AA      AA  DD      DD  MM      MM  RR    RR    EE            GG      GG      ....
LLLLLLLLLL      OOOOOO    AA      AA  DDDDDDDD    MM      MM  RR      RR  EEEEEEEEEE      GGGGGG        ....
LLLLLLLLLL      OOOOOO    AA      AA  DDDDDDDD    MM      MM  RR      RR  EEEEEEEEEE      GGGGGG        ....

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

Table of contents

```
0000     1          .TITLE  LOADMREG - LOAD MBA AND UBA MAP REGISTERS
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 ; D. N. CUTLER 1-NOV-77
0000    29 ;
0000    30 ; LOAD MBA MAP REGISTERS
0000    31 ;
0000    32 ; MODIFIED BY:
0000    33 ;
0000    34 ;       V03-003 RLRLBCNT        Robert L. Rappaport     6-Jul-1984
0000    35 ;               In IOC$LUBAUDAMAP, pickup longword BCNT from IRP rather
0000    36 ;               than word BCNT.
0000    37 ;
0000    38 ;       V03-002 RLRPDTADP       Robert L. Rappaport     9-Apr-1984
0000    39 ;               Modify IOC$LUBAUDAMAP so as to pickup the pointer to the
0000    40 ;               ADP from PDT$L_ADP(R4).
0000    41 ;
0000    42 ;       V03-001 KDM0002         Kathleen D. Morse       28-Jun-1982
0000    43 ;               Add $VADEF.
0000    44 ;
```

```
0000     46 ;
0000     47 ; MACRO LIBRARY CALLS
0000     48 ;
0000     49
0000     50          $ADPDEF                         ;DEFINE ADP OFFSETS
0000     51          $CDRPDEF                        ;DEFINE CDRP OFFSETS
0000     52          $CRBDEF                         ;DEFINE CRB OFFSETS
0000     53          $MBADEF                         ;DEFINE MBA REGISTER OFFSET DEFINITIONS
0000     54          $PDTDEF                         ;DEFINE PDT OFFSETS
0000     55          $PTEDEF                         ;DEFINE PAGE TABLE ENTRY FIELDS
0000     56          $UBADEF                         ;DEFINE UCB OFFSETS
0000     57          $UBMDDEF                        ;DEFINE UBMD OFFSETS
0000     58          $UCBDEF                         ;DEFINE UCB OFFSETS
0000     59          $VADEF                          ;DEFINE VIRTUAL ADDRESS FIELDS
0000     60          $VECDEF                         ;DEFINE CRB TRANSFER VECTOR OFFSETS
```

```
                                0000     62          .SBTTL  LOAD MASSBUS ADAPTER MAP REGISTERS
                                0000     63 ;+
                                0000     64 ; IOC$LOADMBAMAP - LOAD MASSBUS ADAPTER MAP REGISTERS
                                0000     65 ;
                                0000     66 ; THIS ROUTINE IS CALLED TO LOAD THE MASSBUS ADAPTER MAP REGISTERS, THE
                                0000     67 ; BYTE COUNT REGISTER, AND THE VIRTUAL ADDRESS REGISTER.
                                0000     68 ;
                                0000     69 ; INPUTS:
                                0000     70 ;
                                0000     71 ;       R4 = ADDRESS OF MBA CONFIGURATION STATUS REGISTER.
                                0000     72 ;       R5 = UCB ADDRESS OF UNIT TRANSFER IS TO OCCUR ON.
                                0000     73 ;
                                0000     74 ; OUTPUTS:
                                0000     75 ;
                                0000     76 ;       THE TRANSFER BYTE COUNT, STARTING PAGE OFFSET, AND ADDRESS OF THE
                                0000     77 ;       PAGE TABLE ENTIRES THAT DESCRIBE THE TRANSFER ARE RETRIEVED FROM
                                0000     78 ;       THE SPECIFED UCB AND USED TO LOAD THE MBA BYTE COUNT, VIRTUAL ADDRESS,
                                0000     79 ;       AND MAP REGISTERS. ONE ADDITIONAL MAP REGISTER IS LOADED AS INVALID
                                0000     80 ;       TO STOP THE TRANSFER IF A HARDWARE FAILURE SHOULD OCCUR.
                                0000     81 ;
                                0000     82 ;       R3 IS PRESERVED ACROSS CALL.
                                0000     83 ;-
                                0000     84
                           00000000     85          .PSECT  WIONONPAGED
                                0000     86 IOC$LOADMBAMAP::                             ;LOAD MASSBUS ADAPTER MAP REGISTERS
                     53         DD 0000  87          PUSHL   R3                         ;SAVE REGISTERS
           52       7E    A5    3C 0002  88          MOVZWL  UCB$W_BCNT(R5),R2          ;GET TRANSFER BYTE COUNT
       10  A4          52    CE 0006  89          MNEGL   R2,MBA$L_BCR(R4)           ;LOAD BYTE COUNT REGISTER
           51       7C    A5    3C 000A  90          MOVZWL  UCB$W_BOFF(R5),R1          ;GET BYTE OFFSET IN PAGE
       0C  A4          51    D0 000E  91          MOVL    R1,MBA$L_VAR(R4)           ;LOAD STARTING VIRTUAL ADDRESS
       0C  A4          51    D0 0012  92          MOVL    R1,MBA$L_VAR(R4)           ;******TEMP UNTIL MBA ECO ******
                 52    01FF  C241    9E 0016  93          MOVAB   ^X1FF(R2)[R1],R2           ;CALCULATE HIGHEST RELATIVE BYTE AND ROUND
                 52       52    F7    8F 78 001C  94          ASHL    #-9,R2,R2                  ;CALCULATE NUMBER OF MAP REGISTERS TO LOAD
                 51     0800    C4    DE 0021  95          MOVAL   MBA$L_MAP(R4),R1           ;GET ADDRESS OF MBA MAP REGISTERS
                 50       78    A5    D0 0026  96          MOVL    UCB$L_SVAPTE(R5),R0        ;GET ADDRESS OF PAGE TABLE
                 81       80    D0 002A  97 10$:    MOVL    (R0)+,(R1)+                ;LOAD MAP REGISTER
                          09    18 002D  98          BGEQ    30$                        ;IF GEQ PTE INVALID
                       F8 52    F5 002F  99 20$:    SOBGTR  R2,10$                     ;ANY MORE TO LOAD?
                          61    D4 0032 100          CLRL    (R1)                       ;LOAD INVALID MAP ENTRY
                       53 8E    D0 0034 101          MOVL    (SP)+,R3                   ;RESTORE REGISTER
                          05    00 0037 102          RSB                                ;
                       53 FC    A0    D0 0038 103 30$:    MOVL    -4(R0),R3                  ;GET THE PTE (NOT FROM MAP REGISTER!)
                          0095  30 003C 104          BSBW    IOC$PTETOPFN               ;GET PFN FROM INVALID PTE
FC A1  53  80000000    8F    C9 003F 105          BISL3   #^X80000000,R3,-4(R1)      ;AND LOAD THE MAP REGISTER
                          E5    11 0048 106          BRB     20$                        ;
```

```
                                                N 2
                        .SBTTL  LOAD UNIBUS ADAPTER MAP REGISTERS
                ;+
                ; IOC$LOADUBAMAP  - LOAD UNIBUS ADAPTER MAP REGISTERS
                ; IOC$LOADUBAMAPA - LOAD UNIBUS ADAPTER MAP REGISTERS ALTERNATE ENTRY FOR
                ;       BYTE ALIGNED UNIBUS DMA DEVICES WHICH NEVER WISH TO SET THE BYTE
                ;       OFFSET BIT IN MAP REGISTERS.  IN ALL OTHER RESPECTS THESE TWO
                ;       ENTRYPOINTS PRODUCE IDENTICAL RESULTS.
                ;
                ; THIS ROUTINE IS CALLED TO LOAD THE UNIBUS ADAPTER MAP REGISTERS.
                ;
                ; INPUTS:
                ;
                ;       R5 = UCB ADDRESS OF UNIT TRANSFER IS TO OCCUR ON.
                ;
                ;       IT IS ASSUMED THAT THE DATAPATH AND MAP REGISTERS HAVE BEEN PREVIOUSLY
                ;       ASSIGNED.
                ;
                ; OUTPUTS:
                ;
                ;       EACH MAP REGISTER IS LOADED WITH THE APPROPRIATE PAGE FRAME NUMBER
                ;       MERGED WITH THE DATAPATH DESIGNATOR AND BYTE OFFSET BIT. ONE ADDITIONAL
                ;       MAP REGISTER IS LOADED AS INVALID TO STOP THE TRANSFER IF A HARDWARE
                ;       FAILURE SHOULD OCCUR.
                ;
                ;       R3 IS PRESERVED ACROSS CALL.
                ;-
                        .ENABL  LSB
IOC$LOADUBAMAPA::                               ;LOAD UNIBUS ADAPTER MAP REGISTERS - ALTERNATE
                                                ; HERE WE DUPLICATE THE CODE IN THE OTHER ENTRY
                                                ;  EXCEPT THAT WE DO NOT CHECK WHETHER THE BYTE
                                                ;  OFFSET IS ODD.  INSTEAD WE BRANCH DIRECTLY
                                                ;   PAST THE SETTING OF THE BYTE OFFSET BIT.
      7E    53    7D                          MOVQ    R3,-(SP)                ;SAVE REGISTERS
51    7C    A5    3C                          MOVZWL  UCB$W_BOFF(R5),R1       ;GET BYTE OFFSET IN PAGE
52    7E    A5    3C                          MOVZWL  UCB$W_BCNT(R5),R2       ;GET TRANSFER BYTE COUNT
53    24    A5    D0                          MOVL    UCB$L_CRB(R5),R3        ;GET ADDRESS OF CRB
            00    EF                          EXTZV   #VEC$V_DATAPATH,-       ;GET DATAPATH
            05                                #VEC$S_DATAPATH,-               ; NUMBER
54    37    A3                                CRB$L_INTD+VEC$B_DATAPATH(R3),R4
            1B    11                          BRB     10$                     ; BRANCH AROUND TO JOIN COMMON CODE
                                                IOC$LOADUBAMAP::                ;LOAD UNIBUS ADAPTER MAP REGISTERS
      7E    53    7D                          MOVQ    R3,-(SP)                ;SAVE REGISTERS
51    7C    A5    3C                          MOVZWL  UCB$W_BOFF(R5),R1       ;GET BYTE OFFSET IN PAGE
52    7E    A5    3C                          MOVZWL  UCB$W_BCNT(R5),R2       ;GET TRANSFER BYTE COUNT
53    24    A5    D0                          MOVL    UCB$L_CRB(R5),R3        ;GET ADDRESS OF CRB
            00    EF                          EXTZV   #VEC$V_DATAPATH,-       ;GET DATAPATH
            05                                #VEC$S_DATAPATH,-               ; NUMBER
54    37    A3                                CRB$L_INTD+VEC$B_DATAPATH(R3),R4
      03    51    E9                          BLBC    R1,10$                  ;IF LBC WORD ALIGNED TRANSFER
      54    10    88                          BISB    #^X10,R4                ;SET BYTE OFFSET BIT
54    0400  8F    A8                  10$:    BISW    #^X400,R4               ;MERGE VALID WITH BYTE OFFSET AND DATAPATH
            05    E1                          BBC     #VEC$V_LWAE,-           ;BRANCH IF LONGWORD ACCESS NOT ENABLED
      03    37    A3                                CRB$L_INTD+VEC$B_DATAPATH(R3),15$
      54    20    88                          BISB    #^X20,R4                ;ELSE SET LWAE FOR MAP REG
52    01FF  C241  9E                  15$:    MOVAB   ^X1FF(R2)[R1],R2        ;CALCULATE HIGHEST RELATIVE BYTE AND ROUND
52    52    F7    8F    78                    ASHL    #-9,R2,R2               ;CALCULATE NUMBER OF MAP REGISTERS TO LOAD
```

```
                                                B 3
                36 A3    52   91  0094   165          CMPB    R2,CRB$L_INTD+VEC$B_NUMREG(R3) ;ENOUGH MAP REGISTERS ASSIGNED?
                   2C    1E   98  0098   166          BGEQU   40$                            ;IF GEQU NO
                51   38 B3    D0  009A   167          MOVL    @CRB$L_INTD+VEC$L_ADP(R3),R1 ;GET ADDRESS OF CONFIGURATION REGISTER
                      00 EF    9E  009E   168          EXTZV   #VEC$V_MAPREG,-                ;GET STARTING REGISTER
                      0F       A0  00A0   169                  #VEC$S_MAPREG,-
                50   34 A3       A1  00A1   170                  CRB$L_INTD+VEC$W_MAPREG(R3),R0
             51   0800 C140  DE   A4  00A4   171          MOVAL   UBA$L_MAP(R1)[R0],R1     ;GET ADDRESS OF FIRST MAP REGISTER TO LOAD
                50   78 A5    D0  AA  00AA   172          MOVL    UCB$L_SVAPTE(R5),R0      ;GET ADDRESS OF PAGE TABLE
                   53 80      D0  AE  00AE   173 20$:     MOVL    (R0)+,R3                 ;GET NEXT PAGE TABLE ENTRY
                      02      19  B1  00B1   174          BLSS    30$                      ;IF LSS VALID PAGE TABLE ENTRY
                      1F      10  B3  00B3   175          BSBB    IOC$PTETOPFN             ;GET PFN FROM INVALID PTE
          53   0B   15   54    F0  B5  00B5   176 30$:     INSV    R4,#21,#11,R3            ;INSERT VALID, BYTE OFFSET, AND DATAPATH
                   81   53    D0  BA  00BA   177          MOVL    R3,(R1)+                 ;LOAD UBA MAP REGISTER
                      EE 52    F5  BD  00BD   178          SOBGTR  R2,20$                   ;ANY MORE TO LOAD?
                      61       D4  C0  00C0   179          CLRL    (R1)                     ;LOAD INVALID MAP ENTRY
                   53   8E     7D  C2  00C2   180          MOVQ    (SP)+,R3                 ;RESTORE REGISTERS
                           05   00  C5  00C5   181          RSB                              ;
                                    C6  00C6   182 40$:     BUG_CHECK UBMAPEXCED,FATAL       ;UNIBUS MAP REGISTER ALLOCATION EXCEEDED
                                    CA  00CA   183          .DSABL  LSB
```

```
                        00CA   185          .SBTTL  GET PFN FROM INVALID PTE
                        00CA   186 ;+
                        00CA   187 ; IOC$PTETOPFN - GET PFN FROM INVALID PTE
                        00CA   188 ;
                        00CA   189 ; THIS ROUTINE IS CALLED TO RETURN THE PAGE FRAME NUMBER FROM A
                        00CA   190 ; PAGE TABLE ENTRY WHICH HAS ALREADY BEEN DETERMINED TO BE NOT VALID.
                        00CA   191 ;
                        00CA   192 ; INPUTS:
                        00CA   193 ;
                        00CA   194 ;       R3 = PAGE TABLE ENTRY
                        00CA   195 ;
                        00CA   196 ; OUTPUTS:
                        00CA   197 ;
                        00CA   198 ;       R3 = PAGE FRAME NUMBER AND MAY INCLUDE THE FOLLOWING FIELDS
                        00CA   199 ;            VALID BIT, MODIFY BIT, PROTECTION FIELD, OWNER FIELD
                        00CA   200 ;
                        00CA   201 ;       ALL OTHER REGISTERS PRESERVED
                        00CA   202 ;-
                        00CA   203
                        00CA   204          .ENABL  LSB
                        00CA   205 GLOBAL:
 53   00000000'FF43   D0  00CA   206          MOVL    @MMG$GL_GPTBASE[R3],R3  ;GLOBAL PAGE TABLE ENTRY
                  0F   19  00D2   207          BLSS    10$                     ;BRANCH IF VALID
                       00D4   208 IOC$PTETOPFN::
  53   FB800000 8F   CA  00D4   209          BICL    #^C<PTE$M_TYP1 ! PTE$M_TYP0 !- ;PTE TYPE BITS
                       00DB   210                  PTE$M_GPTX>,R3          ;AND GPTX/PFN
        05 53   1A   E0  00DB   211          BBS     #PTE$V_TYP1,R3,20$      ;BRANCH IF BAD PTE FOR I/O
        E7 53   16   E4  00DF   212          BBSC    #PTE$V_TYP0,R3,GLOBAL   ;BRANCH IF GLOBAL PAGE
                  05   00E3   213 10$:       RSB
                       00E4   214 20$:       BUG_CHECK INVPTEFMT,FATAL       ;INVALID PAGE TABLE ENTRY FORMAT
                       00E8   215          .DSABL  LSB
                       00E8   216
```

```
                                00E8   218          .SBTTL  Load UBA map registers
                                00E8   219 ;++
                                00E8   220 ; IOC$LOADUBAMAPN - Load UBA map registers
                                00E8   221 ;
                                00E8   222 ;  Functional description:
                                00E8   223 ;
                                00E8   224 ;       This routine is called to load the UNIBUS adapter map registers.
                                00E8   225 ;       It differs from IOC$LOADUBAMAPN in that it does not obtain its
                                00E8   226 ;       variant inputs from the UCB and CRB (which are normally synchronized
                                00E8   227 ;       at fork IPL).  Also, the byte offset and longword aligned
                                00E8   228 ;       capabilities are not supported.
                                00E8   229 ;
                                00E8   230 ; Inputs:
                                00E8   231 ;
                                00E8   232 ;       R1 = buffer address
                                00E8   233 ;       R2 = number of map registers allocated
                                00E8   234 ;            (last one should be extra one for wild transfer stopper)
                                00E8   235 ;       R3 = starting map register allocated
                                00E8   236 ;       R4 = datapath number
                                00E8   237 ;       R5 = UCB address
                                00E8   238 ;
                                00E8   239 ; Outputs:
                                00E8   240 ;
                                00E8   241 ;       R0-R4 destroyed.  R5 preserved.
                                00E8   242 ;--
                                00E8   243 IOC$LOADUBAMAPN::                            ;
                    54    DD    00E8   244          PUSHL   R4                         ; Save datapath number
          51   51   15    09   EF  00EA   245          EXTZV   S^#VA$V_VPN,S^#VA$S_VPN,R1,R1 ; Get buffer virtual page number
           54   00000000'GF    D0  00EF   246          MOVL    G^MMG$GL_SPTBASE,R4       ; Get system page table address
                 50    6441   DE  00F6   247          MOVAL   (R4)[R1],R0               ; Get first PTE address
                54    24 A5   D0  00FA   248          MOVL    UCB$L_CRB(R5),R4          ; Get CRB address
                54    38 B4   D0  00FE   249          MOVL    @CRB$L_INTD+VEC$L_ADP(R4),R4 ; Get first UBA register address
             51    0800 C443   DE  0102   250          MOVAL   UBA$L_MAP(R4)[R3],R1      ; Get address of first map register
              00000400 8F      C9  0108   251          BISL3   #1@<UBA$V_MAP_VALID-UBA$V_MAP_DPD>,- ; Get datapath number
                    54    8E        010E   252                  (SP)+,R4                  ; and set map register valid bit
                    52    D7  0110   253          DECL    R2                        ; Subtract last register from count
                53  80    D0  0112   254 20$:     MOVL    (R0)+,R3                  ; Get next page table entry
                    02    19  0115   255          BLSS    30$                       ; Br if valid page
                    BB    10  0117   256          BSBB    IOC$PTETOPFN              ; Get PFN from invalid (global) page
                15  54    F0  0119   257 30$:     INSV    R4,#UBA$V_MAP_DPD,-       ; Insert datapath and valid bit,
                53  0B          011C   258                  #32-UBA$V_MAP_DPD,R3      ; clearing other PTE flags
                81  53    D0  011E   259          MOVL    R3,(R1)+                  ; Load the map register
                   EE 52    F5  0121   260          SOBGTR  R2,20$                    ; Loop through all registers
                    61    D4  0124   261          CLRL    (R1)                      ; Invalidate last one to stop wild xfer
                          05  0126   262          RSB                               ;
```

```
                     0127   264          .SBTTL  LOAD UNIBUS ADAPTER MAP REGISTERS FOR UDA PORT
                     0127   265 ;+
                     0127   266 ; IOC$LUBAUDAMAP - LOAD UNIBUS ADAPTER MAP REGISTERS FOR UDA PORT
                     0127   267 ;
                     0127   268 ; INPUTS:
                     0127   269 ;
                     0127   270 ;       R4 => PDT.
                     0127   271 ;       R5 => CDRP OF I/O REQUEST.
                     0127   272 ;
                     0127   273 ;       IT IS ASSUMED THAT THE DATAPATH AND MAP REGISTERS HAVE BEEN PREVIOUSLY
                     0127   274 ;       ASSIGNED.
                     0127   275 ;
                     0127   276 ; OUTPUTS:
                     0127   277 ;
                     0127   278 ;       EACH MAP REGISTER IS LOADED WITH THE APPROPRIATE PAGE FRAME NUMBER
                     0127   279 ;       MERGED WITH THE DATAPATH DESIGNATOR AND BYTE OFFSET BIT. ONE ADDITIONAL
                     0127   280 ;       MAP REGISTER IS LOADED AS INVALID TO STOP THE TRANSFER IF A HARDWARE
                     0127   281 ;       FAILURE SHOULD OCCUR.
                     0127   282 ;
                     0127   283 ;       R3 IS PRESERVED ACROSS CALL.
                     0127   284 ;-
                     0127   285
                     0127   286 IOC$LUBAUDAMAP::                ;LOAD UNIBUS ADAPTER MAP REGISTERS
                     0127   287
        7E   53   7D 0127   288          MOVQ    R3,-(SP)                ;SAVE REGISTERS
                     012A   289
                     012A   290          ASSUME  ADP$L_CSR       EQ      0
      00E0 D4    DD 012A   291          PUSHL   @PDT$[_ADP(R4)          ; Push UBA CSR address on stack.
                     012E   292
    51   D0 A5   3C 012E   293          MOVZWL  CDRP$W_BOFF(R5),R1      ; R1=BYTE OFFSET IN PAGE
    52   D2 A5   D0 0132   294          MOVL    CDRP$L_BCNT(R5),R2      ; R2=TRANSFER BYTE COUNT
    50   3C A5   9E 0136   295          MOVAB   CDRP$L_UBARSRCE(R5),R0  ; R0 => MAPPING RESOURCE DESCRIPTOR
              00   EF 013A   296          EXTZV   #VEC$V_DATAPATH,-       ; GET DATAPATH
              05      013C   297                  #VEC$S_DATAPATH,-       ; NUMBER
    54   03 A0      013D   298                  UBMD$B_DATAPATH(R0),R4
                     0140   299
         03 51   E9 0140   300          BLBC    R1,10$                  ;IF LBC WORD ALIGNED TRANSFER
      54    10   88 0143   301          BISB    #^X10,R4                ;SET BYTE OFFSET BIT
   54   0400 8F   A8 0146   302 10$:    BISW    #^X400,R4               ;MERGE VALID WITH BYTE OFFSET AND DATAPATH
 52   01FF C241   9E 014B   303          MOVAB   ^X1FF(R2)[R1],R2        ;CALCULATE HIGHEST RELATIVE BYTE AND ROUND
52   52    F7 8F   78 0151   304          ASHL    #-9,R2,R2               ;CALCULATE NUMBER OF MAP REGISTERS TO LOAD
                     0156   305
              51 8ED0 0156   306          POPL    R1                      ; R1 => UBA CSR.
    02 A0    52   91 0159   307          CMPB    R2,UBMD$B_NUMREG(R0)    ;ENOUGH MAP REGISTERS ASSIGNED?
              28   1E 015D   308          BGEQU   40$                     ;IF GEQU NO
                     015F   309
              00   EF 015F   310          EXTZV   #VEC$V_MAPREG,-         ; GET STARTING REGISTER
              0F      0161   311                  #VEC$S_MAPREG,-
      50   60      0162   312                  UBMD$W_MAPREG(R0),R0
                     0164   313
 51   0800 C140   DE 0164   314          MOVAL   UBA$L_MAP(R1)[R0],R1    ;GET ADDRESS OF FIRST MAP REGISTER TO LOAD
    50    CC A5   D0 016A   315          MOVL    CDRP$[_SVAPTE(R5),R0    ;GET ADDRESS OF PAGE TABLE
      53   80   D0 016E   316 20$:    MOVL    (R0)+,R3                ;GET NEXT PAGE TABLE ENTRY
              03   19 0171   317          BLSS    30$                     ;IF LSS VALID PAGE TABLE ENTRY
           FF5E   30 0173   318          BSBW    IOC$PTETOPFN            ;GET PFN FROM INVALID PTE
53   0B   15    54   F0 0176   319 30$:    INSV    R4,#21,#11,R3           ;INSERT VALID, BYTE OFFSET, AND DATAPATH
              81   53   D0 017B   320          MOVL    R3,(R1)+                ;LOAD UBA MAP REGISTER
```

```
                ED 52   F5  017E   321          SOBGTR  R2,20$                  ;ANY MORE TO LOAD?
                   61   D4  0181   322          CLRL    (R1)                    ;LOAD INVALID MAP ENTRY
             53    8E   7D  0183   323          MOVQ    (SP)+,R3                ;RESTORE REGISTERS
                   05   0186   324              RSB                             ;
                        0187   325 40$:         BUG_CHECK UBMAPEXCED,FATAL      ;UNIBUS MAP REGISTER ALLOCATION EXCEEDED
                        018B   326              .END
```

```
ADP$L_CSR                       = 00000000
BUG$_INVPTEFMT                    ********    X   02
BUG$_UBMAPEXCED                   ********    X   02
CDRP$L_BCNT                     = FFFFFFD2
CDRP$L_SVAPTE                   = FFFFFFCC
CDRP$L_UBARSRCE                 = 0000003C
CDRP$W_BOFF                     = FFFFFFD0
CRB$L_INTD                      = 00000024
GLOBAL                            000000CA R      02
IOC$LOADMBAMAP                    00000000 RG     02
IOC$LOADUBAMAP                    00000061 RG     02
IOC$LOADUBAMAPA                   0000004A RG     02
IOC$LOADUBAMAPN                   000000E8 RG     02
IOC$LUBAUDAMAP                    00000127 RG     02
IOC$PTETOPFN                      000000D4 RG     02
MBA$L_BCR                       = 00000010
MBA$L_MAP                       = 00000800
MBA$L_VAR                       = 0000000C
MMG$GL_GPTBASE                    ********    X   02
MMG$GL_SPTBASE                    ********    X   02
PDT$L_ADP                       = 000000E0
PTE$M_GPTX                      = 003FFFFF
PTE$M_TYP0                      = 00400000
PTE$M_TYP1                      = 04000000
PTE$V_TYP0                      = 00000016
PTE$V_TYP1                      = 0000001A
UBA$L_MAP                       = 00000800
UBA$V_MAP_DPD                   = 00000015
UBA$V_MAP_VALID                 = 0000001F
UBMD$B_DATAPATH                 = 00000003
UBMD$B_NUMREG                   = 00000002
UBMD$W_MAPREG                   = 00000000
UCB$L_CRB                       = 00000024
UCB$L_SVAPTE                    = 00000078
UCB$W_BCNT                      = 0000007E
UCB$W_BOFF                      = 0000007C
VA$S_VPN                        = 00000015
VA$V_VPN                        = 00000009
VEC$B_DATAPATH                  = 00000013
VEC$B_NUMREG                    = 00000012
VEC$L_ADP                       = 00000014
VEC$S_DATAPATH                  = 00000005
VEC$S_MAPREG                    = 0000000F
VEC$V_DATAPATH                  = 00000000
VEC$V_LWAE                      = 00000005
VEC$V_MAPREG                    = 00000000
VEC$W_MAPREG                    = 00000010
```

+----------------+
! 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
WIONONPAGED                     0000018B  (  395.)  02 (  2.)  NOPIC   USR   CON   REL   LCL NOSHR   EXE   RD    WRT NOVEC BYTE
```

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	30	00:00:00.09	00:00:01.41
Command processing	107	00:00:00.52	00:00:03.18
Pass 1	287	00:00:08.90	00:00:23.96
Symbol table sort	0	00:00:01.41	00:00:03.55
Pass 2	75	00:00:01.70	00:00:03.86
Symbol table output	6	00:00:00.07	00:00:00.07
Psect synopsis output	2	00:00:00.03	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	509	00:00:12.73	00:00:36.07

The working set limit was 1200 pages.
49798 bytes (98 pages) of virtual memory were used to buffer the intermediate code.
There were 50 pages of symbol table space allocated to hold 913 non-local and 16 local symbols.
326 source lines were read in Pass 1, producing 14 object records in Pass 2.
20 pages of virtual memory were used to define 19 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYS.OBJ]LIB.MLB;1	12
_255DUA28:[SYSLIB]STARLET.MLB;2	4
TOTALS (all libraries)	16

1012 GETS were required to define 16 macros.

There were no errors, warnings or information messages.

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