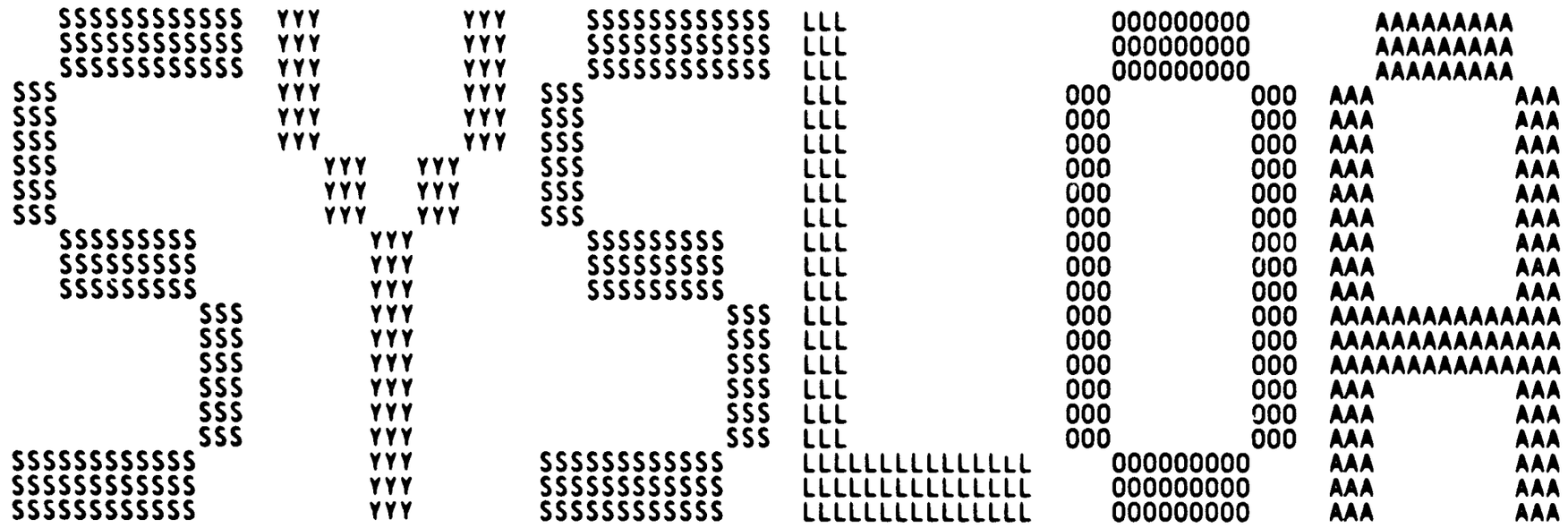
SSSSSSSSSSS YYY YYY SSSSSSSSSSS LLL 000000000 AAAAAAAAA
SSSSSSSSSSS YYY YYY SSSSSSSSSSS LLL 000000000 AAAAAAAAA
SSSSSSSSSSS YYY YYY SSSSSSSSSSS LLL 000000000 AAAAAAAAA
SSS YYY YYY SSS LLL 000 000 AAA AAA
SSS YYY YYY SSS LLL 000 000 AAA AAA
SSS YYY YYY SSS LLL 000 000 AAA AAA
SSS YYY YYY SSS LLL 000 000 AAA AAA
SSS YYY YYY SSS LLL 000 000 AAA AAA
SSS YYY YYY SSS LLL 000 000 AAA AAA
SSSSSSSSS YYY SSSSSSSSS LLL 000 000 AAA AAA
SSSSSSSSS YYY SSSSSSSSS LLL 000 000 AAA AAA
SSSSSSSSS YYY SSSSSSSSS LLL 000 000 AAA AAA
SSS YYY SSS LLL 000 000 AAAAAAAAAAAAAAA
SSS YYY SSS LLL 000 000 AAAAAAAAAAAAAAA
SSS YYY SSS LLL 000 000 AAAAAAAAAAAAAAA
SSS YYY SSS LLL 000 000 AAA AAA
SSS YYY SSS LLL 000 000 AAA AAA
SSS YYY SSS LLL 000 000 AAA AAA
SSSSSSSSSSS YYY SSSSSSSSSSS LLLLLLLLLLLLLLL 000000000 AAA AAA
SSSSSSSSSSS YYY SSSSSSSSSSS LLLLLLLLLLLLLLL 000000000 AAA AAA
SSSSSSSSSSS YYY SSSSSSSSSSS LLLLLLLLLLLLLLL 000000000 AAA AAA

FILEID**ADPSUB780

```
  AAAAAA    DDDDDDDD   PPPPPPPP      SSSSSSSS  UU      UU  BBBBBBBB    77777777      888888      000000
  AAAAAA    DDDDDDDD   PPPPPPPP      SSSSSSSS  UU      UU  BBBBBBBB    77777777      888888      000000
AA      AA  DD      DD  PP      PP  SS        UU      UU  BB      BB          77  88      88  00      00
AA      AA  DD      DD  PP      PP  SS        UU      UU  BB      BB          77  88      88  00      00
AA      AA  DD      DD  PP      PP  SS        UU      UU  BB      BB          77  88      88  00    0000
AA      AA  DD      DD  PP      PP  SS        UU      UU  BB      BB          77  88      88  00    0000
AA      AA  DD      DD  PPPPPPPP      SSSSSS  UU      UU  BBBBBBBB        77      888888    00  00  00
AA      AA  DD      DD  PPPPPPPP      SSSSSS  UU      UU  BBBBBBBB        77      888888    00  00  00
AAAAAAAAAA  DD      DD  PP                  SS  UU      UU  BB      BB    77      88      88  0000    00
AAAAAAAAAA  DD      DD  PP                  SS  UU      UU  BB      BB    77      88      88  0000    00
AA      AA  DD      DD  PP                  SS  UU      UU  BB      BB  77        88      88  00      00      ....
AA      AA  DD      DD  PP                  SS  UU      UU  BB      BB  77        88      88  00      00      ....
AA      AA  DDDDDDDD    PP          SSSSSSSS    UUUUUUUUUU  BBBBBBBB  77          888888      000000        ....
AA      AA  DDDDDDDD    PP          SSSSSSSS    UUUUUUUUUU  BBBBBBBB  77          888888      000000        ....

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          .NOSHOW CONDITIONALS
0000     3          .TITLE  ADPSUB780 - ADAPTER SUBROUTINES FOR VAX 11/780
0000     5
0000     9
0000    13
0000    17
0000    21
0000    22          .IDENT  'V04-000'
0000    23
0000    24 ;****************************************************************************
0000    25 ;*                                                                          *
0000    26 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                *
0000    27 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000    28 ;*  ALL RIGHTS RESERVED.                                                    *
0000    29 ;*                                                                          *
0000    30 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000    31 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
0000    32 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000    33 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
0000    34 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY  *
0000    35 ;*  TRANSFERRED.                                                            *
0000    36 ;*                                                                          *
0000    37 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000    38 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT  *
0000    39 ;*  CORPORATION.                                                            *
0000    40 ;*                                                                          *
0000    41 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000    42 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    43 ;*                                                                          *
0000    44 ;*                                                                          *
0000    45 ;****************************************************************************
0000    46 ;
0000    47 ; Facility: System bootstrapping and initialization
0000    48 ;
0000    49 ; Abstract: This module contains initialization routines that are loaded
0000    50 ;           during system initialization (rather than linked into the system).
0000    51 ;
0000    52 ; Environment: Mode = KERNEL, Executing on INTERRUPT stack, IPL=31
0000    53 ;
0000    54 ; Author:  Kerbey T. Altmann            Creation date: 30-Oct-1982
0000    55 ;
0000    56 ; Modification history:
0000    57 ;
0000    58 ;       V03-007 TCM0002         Trudy C. Matthews       04-Jun-1984
0000    59 ;               Include more 780-specific code for the 11/790 version of
0000    60 ;               this routine.
0000    61 ;
0000    62 ;       V03-006 KPL0001         Peter Lieberwirth       12-Apr-1984
0000    63 ;               Init ADP$L_SHB properly again; V03-004 ASSUMEd this field
0000    64 ;               was at a certain constant offset, and a change to the ADP
0000    65 ;               moved it.  Note - this is a 780 change only.
0000    66 ;
0000    67 ;       V03-005 KDM0081         Kathleen D. Morse       13-Sep-1983
0000    68 ;               Create version for Micro-VAX I.
0000    69 ;
0000    70 ;       V03-004 ROW0196         Ralph O. Weber          27-JUL-1983
0000    71 ;               Correct INI$MPMADP so the ADP$L_SHB is correctly initialized
```

```
                     ;                  to zero.
                     ;
                     ;       V03-003 MSH0001         Maryann Hinden          06-Dec-1982
                     ;               Add initialization for DW750.
                     ;
                     ;       V03-002 ROW0142         Ralph O. Weber          23-NOV-1982
                     ;               Correct JMP in multiport memory interrupt dispatching code
                     ;               prototype, MPMINTD, to a JSB.  MA$INT expects to receive
                     ;               control via a JSB.
                     ;
                     ;       V03-001 TCM0001         Trudy C. Matthews       8-Nov-1982
                     ;               Initialize field ADP$L_AVECTOR in IDB$MPMADP.
                     ;
                     ;--
```

```
00000001  0000     88          C780_LIKE = 1
          0000     90
          0000     94
          0000     98
          0000    102
          0000    106 ;
          0000    107 ; MACRO LIBRARY CALLS
          0000    108 ;
          0000    109          $ADPDEF                         ; Define ADP offsets.
          0000    110          $CRBDEF                         ; Define CRB offsets.
          0000    111          $DCDEF                          ; Define AT codes.
          0000    112          $DDBDEF                         ; Define DDB offsets.
          0000    113          $DDTDEF                         ; Define DDT offsets.
          0000    114          $DYNDEF                         ; Define data structure type codes.
          0000    115          $IDBDEF                         ; Define interrupt dispatcher offsets.
          0000    116          $MBADEF                         ; Define MASSBUS registers.
          0000    117          $MCHKDEF                        ; Define machine check masks.
          0000    118          $MPMDEF                         ; Define multi-port memory.
          0000    119          $NDTDEF                         ; Define nexus device types.
          0000    120          $PRDEF                          ; Define IPR numbers.
          0000    121          $PTEDEF                         ; Define Page Table Entry bits.
          0000    122          $RPBDEF                         ; Define Restart Parameter Block fields.
          0000    123          $SSDEF                          ; Define system service codes.
          0000    124          $UBADEF                         ; Define UBA register offsets.
          0000    125          $UBIDEF                         ; Define UNIBUS interconnect
          0000    126                                          ;  register offsets.
          0000    127          $UCBDEF                         ; Define unit control block.
          0000    128          $VADEF                          ; Define virtual address fields.
          0000    129          $VECDEF                         ; Define vec offsets.
          0000    130
          0000    132          $CEBDEF                         ; COMMON EVENT BLOCK
          0000    133          $FKBDEF                         ; FORK BLOCK
          0000    134          $IPLDEF                         ; INTERRUPT PRIORITY LEVELS
          0000    135          $PRIDEF                         ; PRIORITY INCREMENT DEFINITIONS
          0000    136          $PRQDEF                         ; INTER-PROCESSOR REQUEST
          0000    137          $RSNDEF                         ; RESOURCE NUMBER DEFINITIONS
          0000    138          $SHBDEF                         ; SHARED MEMORY CONTROL BLOCK
          0000    139          $SHDDEF                         ; SHARED MEMORY DATAPAGE
          0000    141
          0000    145
00000000  0000    146          .PSECT  SYSLOA,LONG
```

```
                        .SBTTL  CI$INT - CI INTERRUPT HANDLER
                ;+
                ; CI$INT - CI INTERRUPT HANDLER
                ;
                ;       THIS MODULE IS A DUMMY CI32 INTERRUPT HANDLER WHICH IS USED
                ;       UNTIL THE REAL CI DRIVER (PADRIVER) IS LOADED. IT ALSO CONTAINS
                ;       A DUMMY CI32 CONTROLLER INITIALIZATION  ENTRY POINT.
                ;
                ; INPUTS:
                ;
                ;       THE STACK ON ENTRY IS AS FOLLOWS:
                ;
                ;                0(SP)          ADDRESS OF IDB ADDRESS
                ;           4(SP) - 16(SP)      SAVED R2 - R5
                ;                  20(SP)       INTERRUPT PC
                ;                  24(SP)       INTERRUPT PSL
                ;
                ; OUTPUTS:
                ;
                ;       NONE
                ;
                ; SIDE EFFECTS:
                ;
                ;       INTERRUPTS ARE DISABLED ON THE CI32
                ;-


                ;
                ; $PAREGDEF -- Define offsets to CI registers and fields in the registers.
                ;

                        $DEFINI PAREG

                        $DEF    PA_CNF  .BLKL   1               ; Configuration register

                        _VIELD  PA_CNF,0,<-                     ; Define config register fields:
                        <ADPTYP,8,M>,-                          ;  Adapter type code
                        <PFD,,M>,-                              ;  Powerfail disable
                        <TDEAD,,M>,-                            ;  Transmit dead
                        <TFAIL,,M>,-                            ;  Transmit fail
                        <,5>,-                                  ;  5 unused bits
                        <CRD,,M>,-                              ;  CRD on port init'd read
                        <RDS,,M>,-                              ;  RDS on port init'd read
                        <CXTER,,M>,-                            ;  SBI error confirm
                        <RDTO,,M>,-                             ;  Port init'd read timeout on SBI
                        <CSTMO,,M>,-                            ;  Port init'd command xmit timeout
                        <,1>,-                                  ;  1 unused bit
                        <PUP,,M>,-                              ;  Adapter power up
                        <PDN,,M>,-                              ;  Adaptor power down
                        >                                       ;

                        $DEF    PA_PMC  .BLKL   1               ; Port maint control/status register

                        _VIELD  PA_PMC,0,<-                     ; Define register fields:
                        <MIN,,M>,-                              ;  Maint initialized
                        <MTD,,M>,-                              ;  Maint timer disable
                        <MIE,,M>,-                              ;  Maint interrupt enable
```

```
                                                                                                                                    I 16
                     0008   207          <MIF,,M>,-                                                                              ;  Maint intterupt flag
                     0008   208          >                                                                                       ;
                     0008   209
                     0008   210          $DEFEND PAREG
                     0000   211
                     0000   212 CI$INT::
          53    9E   D0  0000   213          MOVL    @(SP)+,R3                                                                   ; GET ADDRESS OF IDB
          54    63   D0  0003   214          MOVL    IDB$L_CSR(R3),R4                                                            ; GET ADDRESS OF FIRST CSR
64  00400000    8F   D0  0006   215          MOVL    #PA_CNF_M_PUP,PA_CNF(R4)                                                    ; CLEAR POWER UP
64  00800000    8F   D0  000D   216          MOVL    #PA_CNF_M_PDN,PA_CNF(R4)                                                    ; CLEAR POWER DOWN
    04 A4       01   D0  0014   217          MOVL    #PA_PMC_M_MIN,PA_PMC(R4)                                                    ; SET MAINTENCE INITIALIZE
          52    8E   7D  0018   218          MOVQ    (SP)+,R2                                                                    ; RESTORE REGISTERS
          54    8E   7D  001B   219          MOVQ    (SP)+,R4
                     02  001E   220          REI
                     001F   221
                     001F   224
                     001F   225 CI$INITIAL::                                                                                     ; CONTROLLER INITIALIZATION
                     001F   226 CI$SHUTDOWN::                                                                                    ; CONTROLLER SHUTDOWN
                     001F   227
                     001F   230
    04 A4       01   D0  001F   231          MOVL    #PA_PMC_M_MIN,PA_PMC(R4)                                                    ; SET MAINTENCE INITIALIZE
                     0023   234
                     05  0023   235          RSB
```

```
          0024        .SBTTL  DR$INT - DR INTERRUPT HANDLER
          0024  ;+
          0024  ; DR$INT - DR INTERRUPT HANDLER
          0024  ;
          0024  ;       THIS MODULE IS A DUMMY DR32 INTERRUPT HANDLER WHICH IS USED
          0024  ;       UNTIL THE REAL DR DRIVER (XFDRIVER) IS LOADED. IT ALSO CONTAINS
          0024  ;       A DUMMY DR32 CONTROLLER INITIALIZATION  ENTRY POINT.
          0024  ;
          0024  ; INPUTS:
          0024  ;
          0024  ;       THE STACK ON ENTRY IS AS FOLLOWS:
          0024  ;
          0024  ;               0(SP)           ADDRESS OF IDB ADDRESS
          0024  ;       4(SP) - 16(SP)          SAVED R2 - R5
          0024  ;              20(SP)           INTERRUPT PC
          0024  ;              24(SP)           INTERRUPT PSL
          0024  ;
          0024  ; OUTPUTS:
          0024  ;
          0024  ;       NONE
          0024  ;
          0024  ; SIDE EFFECTS:
          0024  ;
          0024  ;       INTERRUPTS ARE DISABLED ON THE DR32
          0024  ;-
          0024
          0024
          0024  ;
          0024  ; DR32 DCR REGISTER DEFINITIONS
          0024  ;-
          0024
          0024          $DEFINI DR
          0000  $DEF    DR_DCR          .BLKL   1       ; DR32 CONTROL REGISTER
          0004          _VIELD  DR_DCR,0,<-
          0004                  <ADPTYP,8>,-            ; ADAPTER TYPE
          0004                  <ID2ERR,,M>,-           ; ID2 ERROR
          0004                  <ID2TOS,2>,-            ; ID2 TIME-OUT STATUS
          0004                  <,1>,-                  ; RESERVED
          0004                  <ID1ERR,,M>,-           ; ID1 ERROR
          0004                  <ID1TOS,2>,-            ; ID1 TIME-OUT STATUS
          0004                  <RDS,,M>,-              ; READ DATA SUBSTITUTE
          0004                  <CRD,,M>,-              ; CORRECTED READ DATA
          0004                  <DCRHLT,,M>,-           ; DCR HALT
          0004                  <DCRABT,,M>,-           ; DCR ABORT INTERRUPT
          0004                  <PKTINT,,M>,-           ; PACKET INTERRUPT
          0004                  <INTENB,,M>,-           ; INTERRUPT ENABLE
          0004                  <,1>,-                  ; RESERVED
          0004                  <PWR_UP,,M>,-           ; ADAPTER POWER UP
          0004                  <PWR_DN,,M>,-           ; ADAPTER POWER DOWN
          0004                  <EXTABT,,M>,-           ; EXTERNAL ABORT
          0004                  <,1>,-                  ; RESERVED
          0004                  <IMPDEP,6>,-            ; IMPLEMENTATION DEPENDENT BITS
          0004          >
          0004
          0004  ; DCR CONTROL FIELD A CODES (USED WHEN WRITING TO DCR)
          0004
00000100  0004          DCR_K_CLRPWRUP=^X100
```

```
                     00000200  0004   296          DCR_K_CLRPWRDN=^X200                    ; CLEAR POWER DOWN
                     00000300  0004   297          DCR_K_CLREXTABT=^X300                   ; CLEAR EXTERNAL ABORT
                     00000400  0004   298          DCR_K_CLRABTINT=^X400                   ; CLEAR ABORT INTERRUPT
                     00000500  0004   299          DCR_K_CLRINTENB=^X500                   ; CLEAR INTERRUPT ENABLE
                     00000600  0004   300          DCR_K_SETINTENB=^X600                   ; SET INTERRUPT ENABLE
                     00000700  0004   301          DCR_K_CLRHLT=^X700                      ; CLEAR HALT
                               0004   302
                               0004   303  ; DCR CONTROL FIELD B CODES (USED WHEN WRITING TO DCR)
                               0004   304
                     00001000  0004   305          DCR_K_CLRCRD=^X1000                     ; CLEAR CRD
                     00002000  0004   306          DCR_K_SETEXTABT=^X2000                  ; SET EXTERNAL ABORT
                     00003000  0004   307          DCR_K_CLRPKTINT=^X3000                  ; CLEAR PACKET INTERRUPT
                     00004000  0004   308          DCR_K_RESET=^X4000                      ; RESET
                     00005000  0004   309          DCR_K_SETOSQTST=^X5000                  ; SET OSEQ TEST
                     00006000  0004   310          DCR_K_CLROSQTST=^X6000                  ; CLEAR OSEQ TEST
                               0004   311          $DEFEND DR
                               0024   312
                               0024   313  DR$INT::
          53    9E    D0       0024   314          MOVL    @(SP)+,R3                       ; GET ADDRESS OF IDB
          54    63    D0       0027   315          MOVL    IDB$L_CSR(R3),R4                ; GET ADDRESS OF FIRST CSR
64  00000100 8F    D0       002A   316          MOVL    #DCR_K_CLRPWRUP,DR_DCR(R4)      ; CLEAR POWER UP
64  00000200 8F    D0       0031   317          MOVL    #DCR_K_CLRPWRDN,DR_DCR(R4)      ; CLEAR POWER DOWN
          52    8E    7D       0038   318          MOVQ    (SP)+,R2                        ; RESTORE REGISTERS
          54    8E    7D       003B   319          MOVQ    (SP)+,R4
                      02       003E   320          REI
                               003F   321
                               003F   324
                               003F   325  DR$INITIAL::                                   ; CONTROLLER INITIALIZATION
                               003F   326  DR$SHUTDOWN::                                  ; CONTROLLER SHUTDOWN
                               003F   327
                               003F   330
64    4000 8F    3C       003F   331          MOVZWL  #DCR_K_RESET,(R4)                ; RESET DR (R4 POINTS TO CSR)
                               0044   334
                      05       0044   335          RSB
```

```
                     0045   337          .SBTTL  UBA$INITIAL - CPU-DEPENDENT UNIBUS ADAPTER INITIALIZATION
                     0045   338 ;+
                     0045   339 ; UBA$INITIAL - UNIBUS ADAPTER INITIALIZATION
                     0045   340 ;
                     0045   341 ; THIS ROUTINE IS CALLED VIA A JSB INSTRUCTION AT SYSTEM STARTUP AND AFTER
                     0045   342 ; A POWER RECOVERY RESTART TO ALLOW INITIALIZATION OF UNIBUS ADAPTERS.
                     0045   343 ;       (POWERFAIL AND INITADP)
                     0045   344 ;
                     0045   345 ; INPUTS:
                     0045   346 ;
                     0045   347 ;       R2 = ADDRESS OF ADAPTER CONTROL BLOCK  (11/780 AND 11/750)
                     0045   348 ;       R4 = ADDRESS OF UNIBUS ADAPTER CONFIGURATION STATUS REGISTER (11/780)
                     0045   349 ;
                     0045   350 ;       ALL INTERRUPTS ARE LOCKED OUT.
                     0045   351 ;
                     0045   352 ; OUTPUTS:
                     0045   353 ;
                     0045   354 ;       THE UNIBUS ADAPTER IS INITIALIZED AND INTERRUPTS ARE ENABLED.
                     0045   355 ;-
                     0045   356
                     0045   357 UBA$INITIAL::                           ;UNIBUS ADAPTER INITIALIZATION
                     0045   358                                         ;
                     0045   360
     64    00   D2    0045   361          MCOML   #0,UBA$L_CSR(R4)       ;CLEAR ALL ADAPTER CONFIGURATION ERRORS
  08 A4    00   D2    0048   362          MCOML   #0,UBA$L_SR(R4)        ;CLEAR ALL ADAPTER STATUS BITS
50   0256  C2   3C    004C   363          MOVZWL  ADP$W_UMR_DIS(R2),R0   ;PICK UP THE NUMBER OF UMR'S TO DISABLE
50   50    16   78    0051   364          ASHL    #UBA$V_CR_MRDSB-4,R0,R0 ;DIVIDE BY 16 THEN SHIFT INTO POSITION
               C9    0055   365          BISL3   #UBA$M_CR_SUEFIE!-     ;ENABLE INTERRUPTS
                     0056   366                   UBA$M_CR_BRIE!-
                     0056   367                   UBA$M_CR_CNFIE!-
                     0056   368                   UBA$M_CR_USEFIE!-
                     0056   369                   UBA$M_CR_IFSIE,-
04 A4    50   0000007C 8F   0056   370                   R0,UBA$L_CR(R4)
                     005E   371
                     005E   373
                     005E   383
                     005E   384 10$:                                    ;NO SPECIAL INIT FOR 11/730 OR uVAX I
                05   005E   385          RSB                            ;
                     005F   386 ;
                     005F   387 ; IGNORE UNEXPECTED UNIBUS INTERRUPTS
                     005F   388 ;
                     005F   389
                     005F   390          .ALIGN  LONG
                     0060   391
                     0060   392 UBA$INTO::                              ; PASSIVE RELEASES THROUGH VECTOR 0
                     0060   393
     00000000'9F     D6    0060   394          INCL    @#IO$GL_UBA_INTO       ; COUNT THEM
                00   11    0066   395          BRB     UBA_UNEXINT            ; JOIN COMMON CODE, VECTORS ARE ALLIGNED
                     0068   396
                     0068   397          .ALIGN  LONG
                     0068   398
                     0068   399 ;
                     0068   400 ; NOTE: UBA$UNEXINT is the label in the EXEC that is a JMP @#UBA_UNEXINT.
                     0068   401 ;       This seeming duplicity is necesary since there is code that must
                     0068   402 ;       refer to the EXEC address from within the SYSLOA image.
                     0068   403 ;
                     0068   404 UBA_UNEXINT::                           ; UNEXPECTED INTERRUPT CODE
```

```
                0068   405
                0068   407                                              ; FOR 780-LIKE PROCESSORS, RESTORE
                0068   408                                              ; SAVED REGISTERS
3F    BA        0068   409          POPR    #^M<R0,R1,R2,R3,R4,R5>      ;
                006A   412                                              ; FOR 11/750, NO REGISTERS SAVED
      02        006A   414          REI                                 ; IGNORE INTERRUPT
```

```
                            006B   418         .SBTTL  MASSBUS ADAPTER INTERRUPT DISPATCHER
                            006B   419 ;+
                            006B   420 ; MBA$INT - MASSBUS ADAPTER INTERRUPT DISPATCHER
                            006B   421 ;
                            006B   422 ; THIS ROUTINE IS ENTERED VIA A JSB INSTRUCTION WHEN AN INTERRUPT OCCURS
                            006B   423 ; ON A MASSBUS ADAPTER. THE STATE OF THE STACK ON ENTRY IS:
                            006B   424 ;
                            006B   425 ;       00(SP) = ADDRESS OF IDB ADDRESS.
                            006B   426 ;       04(SP) = SAVED R2.
                            006B   427 ;       08(SP) = SAVED R3.
                            006B   428 ;       12(SP) = SAVED R4.
                            006B   429 ;       16(SP) = SAVED R5.
                            006B   430 ;       20(SP) = INTERRUPT PC.
                            006B   431 ;       24(SP) = INTERRUPT PSL.
                            006B   432 ;
                            006B   433 ; INTERRUPT DISPATCHING OCCURS AS FOLLOWS:
                            006B   434 ;
                            006B   435 ;       IF THE INTERRUPTING ADAPTER IS CURRENTLY OWNED AND THE OWNER UNIT
                            006B   436 ;       IS EXPECTING AN INTERRUPT, THEN THAT UNIT IS DISPATCHED FIRST. ALL
                            006B   437 ;       OTHER UNITS ARE DISPATCHED BY READING THE ATTENTION SUMMARY REG-
                            006B   438 ;       ISTER AND SCANNING FOR UNITS THAT HAVE ATTENTION SET. AS EACH UNIT
                            006B   439 ;       IS FOUND, ITS ATTENTION SUMMARY BIT IS CLEARED AND THEN A TEST IS
                            006B   440 ;       MADE TO DETERMINE IF AN INTERRUPT IS EXPECTED ON THE UNIT. IF YES,
                            006B   441 ;       THEN THE DRIVER IS CALLED AT ITS INTERRUPT RETURN ADDRESS. ELSE
                            006B   442 ;       THE DRIVER IS CALLED AT ITS UNSOLICITED INTERRUPT ADDRESS. AS EACH
                            006B   443 ;       CALL TO THE DRIVER RETURNS, THE ATTENTION SUMMARY REGISTER IS RE-
                            006B   444 ;       READ AND AN ATTEMPT IS MADE TO FIND ANOTHER UNIT TO DISPATCH. WHEN
                            006B   445 ;       NO UNITS REQUESTING ATTENTION REMAIN, THE INTERRUPT IS DISMISSED.
                            006B   446 ;-
                            006B   447
                            006B   448         .ALIGN  LONG
                            006C   449
                            006C   450 MBA$INT::                                       ;MASSBUS ADAPTER INTERRUPT DISPATCHER
        53   00 BE      D0  006C   451         MOVL    @(SP),R3                        ;GET ADDRESS OF IDB
           54   63      D0  0070   452         MOVL    IDB$L_CSR(R3),R4                ;GET ADDRESS OF CONFIGURATION STATUS REGISTE
                            0073   453
                            0073   455
     00800000   8F      D3  0073   456         BITL    #MBA$M_CSR_PD,-
                64          0079   457                  MBA$L_CSR(R4)                  ;CHECK FOR MBA POWER DOWN
                61      12  007A   458         BNEQ    45$                             ;BRANCH IF POWERFAIL
                            007C   459
                            007C   467
        55   04 A3      D0  007C   468         MOVL    IDB$L_OWNER(R3),R5              ;GET OWNER UNIT UCB ADDRESS
                0A      13  0080   469         BEQL    10$                             ;IF EQL NO OWNER
      52   0090 C5      9A  0082   470         MOVZBL  UCB$B_SLAVE(R5),R2              ;GET OWNER SLAVE CONTROLLER NUMBER
    21 64 A5    01      E0  0087   471         BBS     #UCB$V_INT,UCB$W_STS(R5),20$ ;IF SET, INTERRUPT EXPECTED
        53   00 BE      D0  008C   472 10$:    MOVL    @(SP),R3                        ;RETRIEVE ADDRESS OF IDB
           54   63      D0  0090   473         MOVL    IDB$L_CSR(R3),R4                ;RETRIEVE MBA CONFIGURATION REGISTER ADDRESS
        08 A4   00      D2  0093   474         MCOML   #0,MBA$L_SR(R4)                 ;CLEAR ALL MBA STATUS BITS
      52   0410 C4      D0  0097   475         MOVL    MBA$L_AS(R4),R2                 ;READ ATTENTION SUMMARY REGISTER
  52  52   08   00      EA  009C   476         FFS     #0,#8,R2,R2                     ;FIND FIRST UNIT REQUESTING ATTENTION
                0A      12  00A1   477         BNEQ    20$                             ;IF NEQ UNIT FOUND
           5E   04      C0  00A3   478         ADDL    #4,SP                           ;REMOVE IDB ADDRESS FROM STACK
           52   8E      7D  00A6   479         MOVQ    (SP)+,R2                        ;RESTORE REGISTERS
           54   8E      7D  00A9   480         MOVQ    (SP)+,R4                        ;
                        02  00AC   481         REI                                     ;
                            00AD   482
```

ADP
V04

```
                                                     C 1
                55   18 A342   D0  00AD   483 20$:    MOVL    IDB$L_UCBLST(R3)[R2],R5 ;GET ADDRESS OF UCB OR INTERRUPT DISPATCHER
                       22 55   E8  00B2   484         BLBS    R5,40$                  ;IF LBS INTERRUPT DISPATCHER FOR MULTI-
                               00B5   485                                             ; DEVICE CONTROLLER
          0410 C4    01   52   78  00B5   486         ASHL    R2,#1,MBA$L_AS(R4)      ;CLEAR ATTENTION SUMMARY BIT
                          55   D5  00BB   487         TSTL    R5                      ;SEE IF UCB DEFINED
                          CD   13  00BD   488         BEQL    10$                     ;IF EQL NONE DEFINED
              09 64 A5    01   E5  00BF   489         BBCC    #UCB$V_INT,UCB$W_STS(R5),30$ ;IF CLR, INTERRUPT NOT EXPECTED
                53   10 A5   7D  00C4   490         MOVQ    UCB$L_FR3(R5),R3        ;RESTORE DRIVER CONTEXT
                        0C B5   16  00C8   491         JSB     @UCB$L_FPC(R5)          ;CALL DRIVER AT INTERRUPT RETURN ADDRESS
                           BF   11  00CB   492         BRB     10$                     ;
                               00CD   493
                53   0088 C5   D0  00CD   494 30$:    MOVL    UCB$L_DDT(R5),R3        ;GET ADDRESS OF DDT
                        04 B3   16  00D2   495         JSB     @DDT$L_UNSOLINT(R3)     ;CALL UNSOLICITED INTERRUPT ROUTINE
                           B5   11  00D5   496         BRB     10$                     ;
                               00D7   497
                           7E   DC  00D7   498 40$:    MOVPSL  -(SP)                   ;READ CURRENT PSL
                           75   16  00D9   499         JSB     -(R5)                   ;CALL SLAVE CONTROLLER INTERRUPT DISPATCHER
                           AF   11  00DB   500         BRB     10$                     ;
                               00DD   501
                               00DD   503
                               00DD   504 ;
                               00DD   505 ; IN CASE OF ADAPTER POWER DOWN BIT ASSERTED, RETRIEVE ADP ADDRESS AND JUMP
                               00DD   506 ; TO ADAPTER ERROR ROUTINE IN SYSLOA780.
                               00DD   507 ;
                               00DD   508
                54   14 A3   D0  00DD   509 45$:    MOVL    IDB$L_ADP(R3),R4        ;GET ADP ADDRESS
                        FF1C' 31  00E1   510         BRW     EXE$RA780_INT           ;JUMP TO ERROR ROUTINE
                               00E4   511
                               00E4   533
```

```
                     00E4    535          .SBTTL  MASSBUS ADAPTER INITIALIZATION
                     00E4    536 ;+
                     00E4    537 ; MBA$INITIAL - MASSBUS ADAPTER INITIALIZATION
                     00E4    538 ;
                     00E4    539 ; THIS ROUTINE IS CALLED VIA A JSB INSTRUCTION AT SYSTEM STARTUP AND AFTER
                     00E4    540 ; A POWER RECOVERY RESTART TO ALLOW INITIALIZATION OF MASSBUS ADAPTERS.
                     00E4    541 ;
                     00E4    542 ; INPUTS:
                     00E4    543 ;
                     00E4    544 ;       R4 = CSR ADDRESS OF MASSBUS ADAPTER.
                     00E4    545 ;       R5 = ADDRESS OF ADAPTER IDB.
                     00E4    546 ;
                     00E4    547 ;       ALL INTERRUPTS ARE LOCKED OUT.
                     00E4    548 ;
                     00E4    549 ; OUTPUTS:
                     00E4    550 ;
                     00E4    551 ;       THE MASSBUS ADAPTER IS INITIALIZED AND INTERRUPTS ARE ENABLED.
                     00E4    552 ;-
                     00E4    553
                     00E4    554 MBA$INITIAL::                           ;MASSBUS ADAPTER INITIALIZATION
                     00E4    555
   01    D0          00E4    558          MOVL    #MBA$M_CR_INIT,-
04 A4                00E6    559                   MBA$L_CR(R4)          ;INITIALIZE MASSBUS ADAPTER
   04    D0          00E8    560          MOVL    #MBA$M_CR_IE,-
04 A4                00EA    561                   MBA$L_CR(R4)          ;ENABLE INTERRUPTS
                     00EC    564
         05          00EC    565          RSB
```

```
                                                E 1
                                00ED   567          .SBTTL  INI$MPMADP - BUILD ADP AND INITIALIZE MULTI-PORT MEMORY
                                00ED   568 ;+
                                00ED   569 ; INI$MPMADP IS CALLED AFTER  MAPPING THE REGISTERS FOR A MULTI-PORT
                                00ED   570 ; MEMORY ADAPTER.  AN ADAPTER CONTROL BLOCK IS ALLOCATED AND FILLED.
                                00ED   571 ; THE HARDWARE ADAPTER IS THEN INITIALIZED BY CALLING MPM$INITIAL.
                                00ED   572 ;
                                00ED   573 ; NOTE: THIS ROUTINE HAS BEEN LOCATED HERE IN SYSLOAXXX.EXE INSTEAD OF
                                00ED   574 ;       INILOA.EXE BECAUSE IT CAN BE CALLED WHILE THE SYSTEM IS RUNNING
                                00ED   575 ;       LONG AFTER INILOA.EXE HAS BEEN DELETED!!!
                                00ED   576 ;
                                00ED   577 ; INPUT:
                                00ED   578 ;       R4 - nexus identification number of this nexus
                                00ED   579 ;
                                00ED   580 ; OUTPUTS:
                                00ED   581 ;       ALL REGISTERS PRESERVED
                                00ED   582 ;-
                                00ED   583
                      00000010  00ED   584 NUMMPMVEC = 16                          ; NUMBER OF INTER-PORT INTERRUPT VECTORS
                                00ED   585
                                00ED   586 INI$MPMADP::                           ; INITIALIZE MPM DATA STRUCTURES
                                00ED   587
                   07FF 8F    BB  00ED   592          PUSHR   #^M<R0,R1,R2,R3,R4,R5,R6,R7,R8,R9,R10> ; SAVE REGISTERS
                                00F1   593 ;
                                00F1   594 ; Allocate and initialize Adapter Control Block (ADP).
                                00F1   595 ;
              51    0084 8F    3C  00F1   596          MOVZWL  #ADP$C_MPMADPLEN+-        ; GET SIZE OF ADP PLUS VECTOR
                                00F6   597                  <NUMMPMVEC * 4>,R1        ; DISPATCH TABLE
               00000000'9F    16  00F6   598          JSB     @#EXE$ALONONPAGED         ; ALLOCATE ADP FOR ADAPTER
                    53    52  D0  00FC   599          MOVL    R2,R3                     ; COPY ADDRESS
           58   00000000'9F    D0  00FF   600          MOVL    @#MMG$GL_SBICONF,R8       ; GET SYSTEM ADDRESS OF CONFIG ARRAY
                                0106   601          ASSUME  ADP$L_CSR EQ 0
                  83    6844  D0  0106   602          MOVL    (R8)[R4],(R3)+            ; SET ADDRESS OF CONFIG REGISTER
                                010A   603          ASSUME  ADP$L_LINK EQ 4
                        83    D4  010A   604          CLRL    (R3)+                     ; CLEAR LINK FIELD
                                010C   605          ASSUME  ADP$W_SIZE EQ 8
                     83    51  B0  010C   606          MOVW    R1,(R3)+                  ; SET SIZE OF STRUCTURE
                     83    01  9B  010F   607          MOVZBW  #DYN$C_ADP,(R3)+          ; SET TYPE OF STRUCTURE
                                0112   608          ASSUME  ADP$W_TR EQ 12
                     83    54  B0  0112   609          MOVW    R4,(R3)+                  ; SET NEXUS NUMBER OF ADAPTER
                                0115   610          ASSUME  ADP$W_ADPTYPE EQ 14
                     83    03  B0  0115   611          MOVW    #AT$_MPM,(R3)+            ; SET THE ADAPTER TYPE
                                0118   612          ASSUME  ADP$L_VECTOR EQ 16
                  51    40 A2  DE  0118   613          MOVAL   ADP$L_INTD+8(R2),R1       ; GET ADDRESS OF DISPATCH TABLE
                     83    51  D0  011C   614          MOVL    R1,(R3)+                  ; SET ADDRESS OF DISPATCH TABLE
                                011F   615          ASSUME  ADP$L_PRQQFL EQ 20
                                011F   616          ASSUME  ADP$L_PRQQBL EQ 24
                     83    53  D0  011F   617          MOVL    R3,(R3)+                  ; INIT PRQ WAIT QUEUE FORWARD PTR.
                  83    14 A2  DE  0122   618          MOVAL   ADP$L_PRQQFL(R2),(R3)+    ; INIT PRQ WAIT QUEUE BACKWARD PTR.
                     30 A2  7C  0126   619          CLRQ    ADP$L_SHB(R2)             ; CLEAR SHB FIELD
                                0129   620 ;
                                0129   621 ; Initialize adapter interrupt vectors in System Control Block.
                                0129   622 ;
           50   00000000'9F    D0  0129   623          MOVL    @#EXE$GL_SCB,R0           ; GET ADDRESS OF SCB
        54   54    04    00    EF  0130   624          EXTZV   #0,#4,R4,R4               ; Get low 4 bits of nexus number.
              50    0100 C044  DE  0135   625          MOVAL   ^X100(R0)[R4],R0          ; COMPUTE ADDR OF 1ST INT VECTOR
                 1C A2    50    D0  013B   626          MOVL    R0,ADP$L_AVECTOR(R2)      ; SAVE ADDRESS OF ADAPTER'S SCB VECTORS
                                013F   627                                            ; *** VECTORS WITHOUT JUMPER ***
```

```
                                                        ;  1
                60    39 A2    9E  013F    628          MOVAB   ADP$L_INTD+1(R2),(R0)    ; CONNECT IPL 20 TO DISPATCHER
         40 A0    0001'CF    9E  0143    629          MOVAB   W^EXE$INT58+1,64(R0)     ; CONNECT IPL 21 TO ERROR LOGGER
                             0149    630                                               ; *** VECTORS WITH JUMPER ***
           0080 C0    39 A2    9E  0149    631          MOVAB   ADP$L_INTD+1(R2),128(R0) ; CONNECT IPL 22 TO DISPATCHER
        00C0 C0    0001'CF    9E  014F    632          MOVAB   W^EXE$INT58+1,192(R0)    ; CONNECT IPL 23 TO ERROR LOGGER
                             0156    633                                               ; ***
                             0156    634 ;
                             0156    635 ; Initialize inter-port dispatcher.
                             0156    636 ;
           38 A2    82'AF    D0  0156    637          MOVL    B^MPMINTD,ADP$L_INTD(R2) ; INIT INTER-PORT DISPATCHER
        3C A2    02CE'CF    9E  015B    638          MOVAB   W^MA$INT,ADP$L_INTD+4(R2);  AND ADDRESS
                 50    10    D0  0161    639          MOVL    #NUMMPMVEC,R0            ; GET NUMBER OF VECTORS IN TABLE
                             0164    640 10$:
                       81    D4  0164    641          CLRL    (R1)+                    ; SET VECTOR TO "NOT IN USE"
                    FB 50    F5  0166    642          SOBGTR  R0,10$                   ; DECREMENT COUNT AND LOOP
                 54    62    D0  0169    643          MOVL    ADP$L_CSR(R2),R4         ; GET ADAPTER CSR ADDRESS
                 50    64    D0  016C    644          MOVL    MPM$L_CSR(R4),R0         ; GET CSR VALUE
                       00    EF  016F    645          EXTZV   #MPM$V_CSR_PORT,-        ; GET PORT NUMBER
              50    50    02        0171    646                  #MPM$S_CSR_PORT,R0,R0    ;
           34 A2    50    90  0174    647          MOVB    R0,ADP$B_PORT(R2)        ; SAVE PORT NUMBER
                 FE85'    30  0178    648          BSBW    ADPLINK                  ; LINK ADP TO END OF CHAIN
                             0178    649 ;
                             0178    650 ; Initialize adapter hardware.
                             0178    651 ;
                       0D    10  0178    652          BSBB    MA$INITIAL               ; INITIALIZE THE ADAPTER
                 07FF 8F    BA  017D    653          POPR    #^M<R0,R1,R2,R3,R4,R5,R6,R7,R8,R9,R10> ; RESTORE REGS
                       05    0181    654          RSB
                             0182    655
                             0182    656 MPMINTD:                                      ; MULTI-PORT INTERRUPT DISPATCHER
                       3F    BB  0182    657          PUSHR   #^M<R0,R1,R2,R3,R4,R5>   ;
             00000000  9F    16  0184    658          JSB     @#0                      ;
```

```
                                018A   661          .SBTTL  MA$INITIAL - INITIALIZE MULTI-PORT MEMORY ADAPTER
                                018A   662 ;++
                                018A   663 ;
                                018A   664 ; MPM$INITIAL - INITIALIZE MULTI-PORT MEMORY ADAPTER
                                018A   665 ;
                                018A   666 ; THIS ROUTINE IS CALLED AT SYSTEM INTIALIZATION AND AFTER A POWER
                                018A   667 ; RECOVERY RESTART TO INITIALIZE THE PORT ADAPTER BY CLEARING ANY
                                018A   668 ; ERRORS AND ENABLING ALL INTERRUPTS.
                                018A   669 ;
                                018A   670 ; INPUTS:
                                018A   671 ;
                                018A   672 ;       R4 = ADDR OF ADAPTER CSR.
                                018A   673 ;
                                018A   674 ;       IPL = 31
                                018A   675 ;
                                018A   676 ; OUPUTS:
                                018A   677 ;
                                018A   678 ;       ANY ERRORS IN PORT ARE CLEARED AND ALL INTERRUPTS ARE ENABLED.
                                018A   679 ;--
                                018A   680
                                019A   681 MA$INITIAL::                                ; INTIALIZE PORT
                                018A   682
            04 A4    D4         018A   687          CLRL    MPM$L_CR(R4)                ; CLEAR INTERRUPT ENABLE AND DIAG BITS
            08 A4    D4         018D   688          CLRL    MPM$L_SR(R4)                ; CLEAR INVALIDATE DISABLE AND DIAG BITS
      50    0C A4    D0         0190   689          MOVL    MPM$L_INV(R4),R0            ; GET INVALIDATION REGISTER CONTENTS
50    800FFFFF 8F    CA         0194   690          BICL    #^C<MPM$M_INV_STADR>,R0     ; CLEAR ALL BITS BUT STARTING ADDRESS
                                019B   691          ASSUME  MPM$V_INV_ID EQ 0           ; CACHED ID'S START AT BIT 0
 0C A4    50    01    C9         019B   692          BISL3   #1@0,R0,MPM$L_INV(R4)       ; SET CPU (NEXUS 0) AS CACHED
            18 A4    D4         01A0   693          CLRL    MPM$L_CSR1(R4)              ; CLEAR DIAGNOSTIC BITS
            1C A4    D4         01A3   694          CLRL    MPM$L_MR(R4)                ; CLEAR DIAGNOSTIC BITS
      00400000 8F    D0         01A6   695          MOVL    #MPM$M_CSR_PU,-             ; CLEAR ANY POWER-UP STATUS
            64                 01AC   696                  MPM$L_CSR(R4)               ;
      FF000000 8F    D0         01AD   697          MOVL    #MPM$M_CR_ERRS,-            ; CLEAR ANY PORT ERRORS
            04 A4              01B3   698                  MPM$L_CR(R4)                ;
                     D0         01B5   699          MOVL    #MPM$M_SR_SS!-              ; CLEAR ANY STATUS ERRORS AND
                                01B6   700                  MPM$M_SR_IDL!-              ;  ENABLF ERROR INTERRUPTS
                                01B6   701                  MPM$M_SR_IT!-               ;
                                01B6   702                  MPM$M_SR_AGP!-              ;
                                01B6   703                  MPM$M_SR_MXF!-              ;
                                01B6   704                  MPM$M_SR_ACA!-              ;
                                01B6   705                  MPM$M_SR_EIE,-              ;
08 A4    D000E002 8F            01B6   706                  MPM$L_SR(R4)                ;
                     D0         01BD   707          MOVL    #MPM$M_ERR_ELR!-            ; CLEAR ANY ARRAY ERRORS
                                01BE   708                  MPM$M_ERR_IMP,-             ;
10 A4    90000000 8F            01BE   709                  MPM$L_ERR(R4)               ;
                  00'    E0     01C5   710          BBS     S^#EXE$V_CRDENABL,-         ; BRANCH IF CRD ERROR LOGGING ENABLED
      08 00000000'GF            01C7   711                  G^EXE$GL_DEFFLAGS,10$       ;
         40000000 8F    D0     01CD   712          MOVL    #MPM$M_ERR_ICRD,-           ; ELSE, INHIBIT CRD ERROR LOGGING
               10 A4            01D3   713                  MPM$L_ERR(R4)               ;
         00000400 8F    D0     01D5   714 10$:     MOVL    #MPM$M_CSR1_MIA,-           ; CLEAR ANY ERROR
               18 A4            01DB   715                  MPM$L_CSR1(R4)              ;
               1C A4    D4     01DD   716          CLRL    MPM$L_MR(R4)                ; CLEAR ANY DIAGNOSTIC SETTINGS
            50    64    D0     01E0   717          MOVL    MPM$L_CSR(R4),R0            ; GET CSR REGISTER
                  00    EF     01E3   718          EXTZV   #MPM$V_CSR_PORT,-           ; GET PORT NUMBER
         50    50    02       01E5   719                  #MPM$S_CSR_PORT,R0,R0       ;
               50    04    C4 01E8   720          MULL    #MPM$C_PORTS,R0             ; COMPUTE INTERRUPT ENABLE BIT #
               50    10    C0 01EB   721          ADDL    #MPM$V_IIE_CTL,R0           ; ...
```

```
                                                     H 1
                24 A4    0F    50    78  01EE   722          ASHL    R0,#^XF,MPM$L_IIE(R4)   ; ENABLE INTERPORT INTERRUPTS
                                   D0  01F3   723          MOVL    #MPM$M_CR_MIE!-         ; ENABLE ALL INTERRUPTS
                                       01F4   724                  MPM$M_CR_EIE,-          ;
                     04 A4    03           01F4   725                  MPM$L_CR(R4)            ;
                                           01F7   726                                          ;  FROM ALL PORTS
                                       05  01F7   727          RSB                             ; RETURN
```

ADP
Pse

PSE

$AB
SYS

Pha

Ini
Com
Pas
Sym
Pas
Sym
Pse
Cro
Ass

The
138
The
117
44

Mac

-$2
-$2
TOT

226

The

M▸C

```
                                                01F8   730          .SBTTL  INTER-PROCESSOR REQUEST HANDLER
                                                01F8   731 ;++
                                                01F8   732 ;
                                                01F8   733 ; FUNCTIONAL DESCRIPTION:
                                                01F8   734 ;
                                                01F8   735 ;       THIS ROUTINE IS CALLED BY A DRIVER OR AN EXEC FUNCTION TO
                                                01F8   736 ;       EITHER SEND A REQUEST TO OR JUST INTERRUPT ANOTHER PROCESSOR
                                                01F8   737 ;       THAT IS CONNECTED TO A PORT OF THE MULTIPORT MEMORY.
                                                01F8   738 ;
                                                01F8   739 ; INPUTS:
                                                01F8   740 ;
                                                01F8   741 ;       R4 = ADAPTER CONTROL BLOCK ADDRESS.
                                                01F8   742 ;       R5 = IF LSS 0 - ADDRESS OF A FORK BLOCK TO USE IF REQUEST
                                                01F8   743 ;                       BLOCK IS NOT AVAILABLE.
                                                01F8   744 ;            IF GEQ 0 - PORT NUMBER OF PROCESSOR TO JUST INTERRUPT.
                                                01F8   745 ;
                                                01F8   746 ; OUTPUTS:
                                                01F8   747 ;
                                                01F8   748 ;       WHEN THIS ROUTINE IS CALLED WITH A FORK BLOCK ADDRESS, IT WILL
                                                01F8   749 ;       ATTEMPT TO ALLOCATE A REQUEST BLOCK.  IF THE REQUEST FAILS,
                                                01F8   750 ;       THE CONTEXT OF THE CALLER WILL BE SAVED IN THE FORK BLOCK, THE
                                                01F8   751 ;       FORK BLOCK BLOCK WILL BE INSERTED IN THE REQUEST BLOCK WAIT
                                                01F8   752 ;       QUEUE AND A RETURN TO THE CALLER'S CALLER IS EXECUTED.
                                                01F8   753 ;
                                                01F8   754 ;       IF A REQUEST BLOCK IS ALLOCATED SUCCESSFULLY, CONTROL WILL
                                                01F8   755 ;       RETURN TO THE CALLER VIA A CO-ROUTINE CALL SO THE CALLER CAN
                                                01F8   756 ;       FILL-IN THE REQUEST BLOCK.
                                                01F8   757 ;
                                                01F8   758 ;       THE CALLER WILL THEN PERFORM ANOTHER CO-ROUTINE CALL TO RETURN
                                                01F8   759 ;       TO THIS ROUTINE SO THE BLOCK CAN BE INSERTED IN THE DESIRED
                                                01F8   760 ;       PROCESSOR'S INTER-PROCESSOR REQUEST QUEUE.  IF IT IS THE
                                                01F8   761 ;       FIRST REQUEST IN THE QUEUE AN INTER-PORT INTERRUPT WILL
                                                01F8   762 ;       ALSO BE REQUESTED TO WAKE-UP THE DISPATCHER ON THE PORT.
                                                01F8   763 ;
                                                01F8   764 ;
                                                01F8   765 ;       IF THIS ROUTINE IS CALLED WITH A PORT NUMBER INSTEAD OF A
                                                01F8   766 ;       FORK BLOCK ADDRESS, IT WILL JUST REQUEST AN INTERRUPT FOR
                                                01F8   767 ;       THE PROCESSOR ON THE SPECIFIED PORT.  IT IS THEN UP TO THE
                                                01F8   768 ;       INTERRUPTED PROCESSOR TO DETERMINE WHAT THE INTERRUPT WAS
                                                01F8   769 ;       FOR.
                                                01F8   770 ;
                                                01F8   771 ;       R0 = SUCCESS OR FAILURE OF OPERATION.  THIS SHOULD BE CHECKED
                                                01F8   772 ;            BY THE CALLER BOTH TIMES THIS ROUTINE RETURNS.
                                                01F8   773 ;
                                                01F8   774 ;       R3,R4,R5 ARE PRESERVED.
                                                01F8   775 ;
                                                01F8   776 ;--
                                                01F8   777
                                                01F8   778 MA$REQUEST::                             ; REQUEST HANDLER
                                                01F8   779
          51    55    D0                      01F8   784          MOVL    R5,R1                     ; FORK BLOCK ADDRESS SPECIFIED?
                6D    18                      01FB   785          BGEQ    REQ_INTERRUPT             ; IF GEQ, NO - JUST AN INTERRUPT
          51    30 A4 D0                      01FD   786          MOVL    ADP$L_SHB(R4),R1          ; GET SHB ADDRESS
          51    04 A1 D0                      0201   787          MOVL    SHB$L_DATAPAGE(R1),R1     ; GET DATAPAGE ADDRESS
          50    34 A4 9A                      0205   788          MOVZBL  ADP$B_PORT(R4),R0         ; GET OUR PORT NUMBER
 00 00A4  C1    50    E6                      0209   789          BBSSI   R0,SHD$W_PRQWAIT(R1),5$   ; ASSUME FAILURE (AVOID MISSING NOTIFICATION
                                              020F   790 5$:      QRETRY  SUCCESS=TO$-              ; GET A REQUEST BLOCK
```

```
                                020F   791          REMQHI  SHD$Q_PRQ(R1),R2        ;
      50    0394 8F    3C     0220   792          MOVZWL  #SS$_BADQUEUEHDR,R0     ; SET FAILURE STATUS CODE
                       05     0225   793          RSB
                              0226   794 10$:
                    5B 1D     0226   795          BVS     NOBLOCKS                ; IF V-SET, NO BLOCKS LEFT
        50    34 A4    9A     0228   796          MOVZBL  ADP$B_PORT(R4),R0        ; GET OUR PORT NUMBER
00 00A4 C1    50    E7        022C   797          BBCCI   R0,SHD$W_PRQWAIT(R1),RETURN_BLOCK ; CLEAR WAITING FLAG
                              0232   798 ;
                              0232   799 ; RETURN ADDRESS OF REQUEST BLOCK TO CALLER
                              0232   800 ;
                              0232   801 RETURN_BLOCK:                           ; RETURN ADDRESS OF BLOCK
            50    01    D0     0232   802          MOVL    #SS$_NORMAL,R0          ; SET SUCCESS
                  9E    16     0235   803          JSB     @(SP)+                  ; CO-ROUTINE CALL CALLER BACK
                              0237   804 ;
                              0237   805 ; INSERT BLOCK IN DESIRED PORT'S REQUEST QUEUE
                              0237   806 ;
                  54    DD     0237   807          PUSHL   R4                      ; SAVE REGISTER
          51    18 A2    3C     0239   808          MOVZWL  PRQ$W_TO_PORT(R2),R1    ; GET DESIRED PORT NUMBER
          50    30 A4    D0     023D   809          MOVL    ADP$L_SHB(R4),R0        ; GET SHB ADDRESS
          54    04 A0    D0     0241   810          MOVL    SHB$L_DATAPAGE(R0),R4   ; GET DATAPAGE ADDRESS
                              0245   811          QRETRY  SUCCESS=10$-            ; INSERT REQUEST IN PORT'S WORK QUEUE
                              0245   812          INSQTI  (R2),SHD$Q_PRQWRK(R4)[R1]
      50    0394 8F    3C     0257   813          MOVZWL  #SS$_BADQUEUEHDR,R0     ; SET FAILURE STATUS CODE
                       05     11     025C   814          BRB     20$             ;
                       07     13     025E   815 10$:     BEQL    30$             ; IF EQL, FIRST ENTRY IN QUEUE
            50    01    D0     0260   816          MOVL    #SS$_NORMAL,R0          ; SET SUCCESS
                  54 8ED0     0263   817 20$:     POPL    R4                      ; RESTORE REGISTER
                       05     0266   818          RSB                             ;
                              0267   819
                  54 8ED0     0267   820 30$:     POPL    R4                      ; RESTORE REGISTER
                              026A   821 ;
                              026A   822 ; REQUEST AN INTER-PORT INTERRUPT TO WAKE-UP PROCESSOR ON DESIRED
                              026A   823 ; PORT.
                              026A   824 ;
                              026A   825 REQ_INTERRUPT:                          ; REQUEST AN INTER-PORT INTERRUPT
        50    34 A4    9A     026A   826          MOVZBL  ADP$B_PORT(R4),R0        ; GET OUR PORT NUMBER
           50    04    C4     026E   827          MULL    #MPM$C_PORTS,R0         ; COMPUTE INTERRUPT REQUEST BIT #
           50    51    C0     0271   828          ADDL    R1,R0                   ; ...
           50    10    C0     0274   829          ADDL    #MPM$V_IIR_CTL,R0       ; ...
           51    64    D0     0277   830          MOVL    ADP$L_CSR(R4),R1        ; GET ADAPTER CSR ADDRESS
20 A1      01    50    78     027A   831          ASHL    R0,#1,MPM$L_IIR(R1)     ; SET PORT INTERRUPT REQUEST BIT
           50    01    D0     027F   832          MOVL    #SS$_NORMAL,R0          ; SET SUCCESS
                       05     0282   833          RSB                             ;
                              0283   834 ;
                              0283   835 ; NO BLOCKS ARE AVAILABLE.  SAVE THE CALLER'S CONTEXT IN THE FORK
                              0283   836 ; BLOCK, INSERT THE FORK BLOCK IN THE REQUEST BLOCK WAIT QUEUE, AND
                              0283   837 ; RETURN TO THE CALLER'S CALLER.
                              0283   838 ;
                              0283   839 NOBLOCKS:                             ; NO REQUEST BLOCKS AVAILABLE
        10 A5      53    7D     0283   840          MOVQ    R3,FKB$L_FR3(R5)        ; SAVE REGISTERS
              0C A5 8ED0     0287   841          POPL    FKB$L_FPC(R5)           ; SAVE RETURN ADDRESS
        18 B4      65    0E     028B   842          INSQUE  (R5),@ADP$L_PRQQBL(R4)  ; INSERT FORK BLOCK IN WAIT QUEUE
           50    01    D0     028F   843          MOVL    #SS$_NORMAL,R0          ; SET SUCCESS
                       05     0292   844          RSB                             ; RETURN TO CALLER'S CALLER
```

```
                     0293   847          .SBTTL  REPORT RESOURCE AVAILABILITY TO INTERESTED PORTS
                     0293   848 ;++
                     0293   849 ;
                     0293   850 ; FUNCTIONAL DESCRIPTION:
                     0293   851 ;
                     0293   852 ;       THIS ROUTINE IS CALLED TO REPORT TO ANY PROCESSORS THAT A RESOURCE
                     0293   853 ;       HAS BEEN MADE AVAILABLE.
                     0293   854 ;
                     0293   855 ; INPUTS:
                     0293   856 ;
                     0293   857 ;       R0 = RESOURCE NUMBER OF RESOURCE MADE AVAILABLE.
                     0293   858 ;       R1 = SHARED MEMORY CONTROL BLOCK (SHB) ADDRESS.
                     0293   859 ;
                     0293   860 ; OUTPUTS:
                     0293   861 ;
                     0293   862 ;       ANY PROCESSORS WAITING FOR THE SPECIFIED RESOURCE ARE INTERRUPTED
                     0293   863 ;       TO NOTIFY THEM THE RESOURCE IS AVAILABLE.
                     0293   864 ;
                     0293   865 ;       R0,R1,R2,R3 ARE NOT PRESERVED.
                     0293   866 ;--
                     0293   867
                     0293   868 MA$RAVAIL::
                     0293   869
        52   04 A1   D0   0293   874          MOVL    SHB$L_DATAPAGE(R1),R2   ; GET ADDRESS OF DATAPAGE
         00A8 C240   B5   0297   875          TSTW    SHD$W_RESWAIT(R2)[R0]   ; ANYONE WAITING FOR THE RESOURCE?
               2F    13   029C   876          BEQL    30$                     ; IF EQL, NO
            1C A1    DD   029E   877          PUSHL   SHB$L_ADP(R1)           ; SAVE ADDRESS OF ADAPTER CONTROL BLOCK
               53    D4   02A1   878          CLRL    R3                      ; INIT PORT NUMBER
                     02A3   879
    51   00A8 C240   3E   02A3   880 10$:     MOVAW   SHD$W_RESWAIT(R2)[R0],R1 ; GET ADDRESS OF RESOURCE WAIT MASK
        10 61    53   E1   02A9   881          BBC     R3,(R1),20$             ; IF CLR, NO ONE WAITING AT PORT
    51   00C8 C240   3E   02AD   882          MOVAW   SHD$W_RESAVAIL(R2)[R0],R1 ; GET ADDRESS OF AVAILABLE MASK
        00 61    53   E6   02B3   883          BBSSI   R3,(R1),15$             ; SET PORT'S RESOURCE AVAIL BIT
00 00E8 C2       53   E6   02B7   884 15$:     BBSSI   R3,SHD$W_RESSUM(R2),20$ ; SET PORT'S RESOURCE AVAIL SUMMARY BIT
        E2 53    04   F2   02BD   885 20$:     AOBLSS  #MPM$C_PORTS,R3,10$     ; INCREMENT PORT NUMBER AND LOOP
           53 8ED0       02C1   886          POPL    R3                      ; GET ADDRESS OF ADAPTER CONTROL BLOCK
    51   00A8 C240   3C   02C4   887          MOVZWL  SHD$W_RESWAIT(R2)[R0],R1 ; GET RESOURCE WAIT MASK
              0140   30   02CA   888          BSBW    INTERRUPT_PORTS         ; INTERRUPT WAITING PORTS
                     05   02CD   889 30$:     RSB                             ;
```

```
                        02CE   891          .SBTTL  INTER-PORT INTERRUPT DISPATCHER
                        02CE   892 ;++
                        02CE   893 ;
                        02CE   894 ; FUNCTIONAL DESCRIPTION:
                        02CE   895 ;
                        02CE   896 ;       THIS ROUTINE IS ENTERED V.A A JSB INSTRUCTION WHEN AN INTERRUPT
                        02CE   897 ;       OCCURS ON A MULTI-PORT MEMORY.  THE STATE OF THE STACK ON ENTRY IS:
                        02CE   898 ;
                        02CE   899 ;       00(SP) = ADDRESS OF BYTE FOLLOWING "JSB  @#MA$INT" IN ADP$L_INTD.
                        02CE   900 ;       04(SP) - 24(SP) = SAVED R0 - R5.
                        02CE   901 ;       28(SP) = INTERRUPT PC.
                        02CE   902 ;       32(SP) = INTERRUPT PSL.
                        02CE   903 ;
                        02CE   904 ; INTERRUPT DISPATCHING OCCURS AS FOLLOWS:
                        02CE   905 ;
                        02CE   906 ;       THE FIRST REQUEST BLOCK IN THIS PORT'S INTER-PROCESSOR REQUEST
                        02CE   907 ;       QUEUE IS DEQUEUED.  THE REQUEST BLOCK IS THEN USED AS A FORK
                        02CE   908 ;       BLOCK WHICH IS THEN ENTERED INTO THE LOWEST IPL DEVICE DRIVER
                        02CE   909 ;       FORK QUEUE.  IF THE FORK BLOCK WAS THE FIRST IN THE FORK QUEUE,
                        02CE   910 ;       A SOFTWARE INTERRUPT IS POSTED TO DISPATCH THE FORK PROCESS WHEN
                        02CE   911 ;       THE IPL IS LOW ENOUGH.  WHEN ALL THE REQUEST BLOCKS HAVE BEEN
                        02CE   912 ;       DEQUEUED FROM THE PORT'S INTER-PROCESSOR REQUEST QUEUE, AND
                        02CE   913 ;       REQUEUED TO THE FORK QUEUE, THE INTERRUPT IS DISMISSED.
                        02CE   914 ;--
                        02CE   915
                        02CE   916 MA$INT:                                      ; MA780 INTERRUPT DISPATCHER
                        02CE   917
      53    8E    D0    02CE   918          MOVL    (SP)+,R3                   ; GET ADDRESS OF BYTE IN ADP
    53    C0 A3   9E    02D1   919          MOVAB   -<ADP$L_INTD+8>(R3),R3     ; COMPUTE ADDRESS OF ADP
      50    63    D0    02D5   920          MOVL    ADP$L_CSR(R3),R0           ; GET CSR ADDRESS
      52    34 A3   9A    02D8   921          MOVZBL  ADP$B_PORT(R3),R2          ; GET PORT NUMBER
  51    52    04    C5    02DC   922          MULL3   #MPM$C_PORTS,R2,R1         ; COMPUTE INTERRUPT REQUEST BIT #
20 A0    0F    51    78    02E0   923          ASHL    R1,#^XF,MPM$L_IIR(R0)      ; CLEAR ANY INTERRUPT REQUESTS
      54    30 A3   D0    02E5   924          MOVL    ADP$L_SHB(R3),R4           ; GET ADDRESS OF SHB
            03    12    02E9   925          BNEQ    10$                        ; IF NEQ, MEMORY CONNECTED
          00D5    31    02EB   926          BRW     INT_EXIT                   ; ELSE, IGNORE INTERRUPT
      54    04 A4   D0    02EE   927 10$:     MOVL    SHB$L_DATAPAGE(R4),R4      ; GET ADDRESS OF DATAPAGE
                            02F2   928                                         ; R2 = PORT NUMBER
                            02F2   929                                         ; R3 = ADP ADDRESS
                            02F2   930                                         ; R4 = DATAPAGE ADDRESS
            18    BB    02F2   931          PUSHR   #^M<R3,R4>                 ; SAVE R3-R4
00 00A6 C4    52    E6    02F4   932          BBSSI   R2,SHD$W_POLL(R4),DEQUEUE_BLOCK ; INDICATE THIS PROCESSOR ACTIVE
                            02FA   933 ;
                            02FA   934 ; DEQUEUE THE NEXT REQUEST BLOCK IN OUR WORK QUEUE AND REQUEUE TO THE
                            02FA   935 ; APPROPRIATE FORK QUEUE.
                            02FA   936 ;
                            02FA   937 DEQUEUE_BLOCK:                           ; DEQUEUE NEXT REQUEST BLOCK
      53    6E    7D    02FA   938          MOVQ    (SP),R3                    ; RESTORE R3-R4
                            02FD   939          QRETRY  SUCCESS=10$-               ; DEQUEUE THE NEXT REQUEST BLOCK
                            02FD   940          REMQHI  SHD$Q_PRQWRK(R4)[R2],R5    ;
                            030F   941          BUG_CHECK BADQHDR                  ; REMQHI FAILED - BAD QUEUE HEADER
            0D    1D    0313   942 10$:     BVS     BLOCK_AVAIL                ; IF V-SET, NO BLOCKS LEFT
        E2 AF   9F    0315   943          PUSHAB  DEQUEUE_BLOCK              ; SET RETURN PC
    03C6'CF   9F    0318   944          PUSHAB  W^REQUEST_DISP             ; SET FORK PC
00000000'GF   17    031C   945          JMP     G^EXE$FORK                 ; INSERT BLOCK IN FORK QUEUE
                            0322   946 ;
                            0322   947 ; IF THERE ARE ANY FORK PROCESSES ON THIS PROCESSOR WAITING FOR INTER-PROCESSOR
```

```
                                                M 1
                     0322   948 ; REQUEST BLOCKS, AND IF ANY BLOCKS ARE NOW AVAILABLE, GIVE THEM TO THE
                     0322   949 ; PROCESSES AND RESTART THEM.
                     0322   950 ;
                     0322   951 BLOCK_AVAIL:                                  ; CHECK IF ANY BLOCKS AVAILABLE
50 00A4 C4   52  E1  0322   952         BBC     R2,SHD$W_PRQWAIT(R4),RESOURCE_AVAIL ; IF CLR, NO PROCESSES WAITING
        53   6E  7D  0328   953 10$:    MOVQ    (SP),R3                         ; RESTORE R3-R4
                     032B   954         QRETRY  SUCCESS=20$-                    ; ATTEMPT TO ALLOCATE A FREE BLOCK
                     032B   955         REMQHI  SHD$Q_PRQ(R4),R5                ;
                     033C   956         BUG_CHECK BADQHDR                       ; REMQHI FAILED - BAD QUEUE HEADER
           36   1D  0340   957 20$:    BVS     RESOURCE_AVAIL                  ; IF V-SET, NO BLOCK AVAILABLE
     51    14 B3   0F  0342   958         REMQUE  @ADP$L_PRQQFL(R3),R1            ; GET NEXT WAITING FORK BLOCK
           06   12  0346   959         BNEQ    30$                             ; IF NEQ, NOT LAST ENTRY
00 00A4 C4   52  E7  0348   960         BBCCI   R2,SHD$W_PRQWAIT(R4),30$ ; ELSE LAST, CLEAR WAITING FLAG
           25   1D  034E   961 30$:    BVS     50$                             ; IF SET, NO PROCESSES LEFT
        53   51  D0  0350   962         MOVL    R1,R3                           ; SET ADDRESS OF FORK BLOCK
  0B A5    0B A3  90  0353   963         MOVB    FKB$B_FIPL(R3),FKB$B_FIPL(R5) ; SET FORK IPL
           CD AF  9F  0358   964         PUSHAB  B^10$                           ; SET RETURN PC
           64'AF  9F  035B   965         PUSHAB  B^40$                           ; SET FORK PC
     00000000'GF  17  035E   966         JMP     G^EXE$FORK                      ; INSERT BLOCK IN FORK QUEUE
                     0364   967 40$:
        52   55  D0  0364   968         MOVL    R5,R2                           ; SET ADDRESS OF REQUEST BLOCK
        55   53  D0  0367   969         MOVL    R3,R5                           ; SET ADDRESS OF DRIVER FORK BLOCK
     53   10 A5  7D  036A   970         MOVQ    FKB$L_FR3(R5),R3                ; RESTORE REGISTERS
           0C A5  DD  036E   971         PUSHL   FKB$L_FPC(R5)                   ; SET RETURN ADDRESS OF HANDLER CALLER
            FEBE  31  0371   972         BRW     RETURN_BLOCK                    ; RETURN TO HANDLER
                  05  0374   973         RSB                                     ;
                     0375   974 50$:
            0076   30  0375   975         BSBW    DEALLOC_BLOCK                   ; DEALLOCATE UNEEDED BLOCK
                     0378   976 ;
                     0378   977 ; IF THERE IS A RESOURCE NOW AVAILABLE THAT PROCESS(S) ON THIS PROCESSOR
                     0378   978 ; ARE WAITING FOR, CREATE A FORK PROCESS TO REPORT THE AVAILABILITY TO
                     0378   979 ; THE SCHEDULER.
                     0378   980 ;
                     0378   981 RESOURCE_AVAIL:                              ;
42 00E8 C4   52  E1  0378   982         BBC     R2,SHD$W_RESSUM(R4),50$ ; IF CLR, NONE TO REPORT
3C 00A4 C4   52  E6  037E   983         BBSSI   R2,SHD$W_PRQWAIT(R4),50$ ; ASSUME NO BLOCKS AVAILABLE
                     0384   984                                                 ; AND IF THERE ALREADY AREN'T, EXIT
                     0384   985         QRETRY  SUCCESS=20$-                    ; ATTEMPT TO ALLOCATE A BLOCK
                     0384   986         REMQHI  SHD$Q_PRQ(R4),R5                ;
                     0395   987         BUG_CHECK BADQHDR                       ; REMQHI FAILED - BAD QUEUE HEADER
           25   1D  0399   988 20$:    BVS     50$                             ; IF V-SET, NO BLOCK AVAILABLE
00 00A4 C4   52  E7  039B   989         BBCCI   R2,SHD$W_PRQWAIT(R4),30$ ; CLEAR WAIT FLAG
00 00E8 C4   52  E7  03A1   990 30$:    BBCCI   R2,SHD$W_RESSUM(R4),40$ ; CLEAR RESOURCE REPORT SUMMARY
     0B A5   06  90  03A7   991 40$:    MOVB    #IPL$_QUEUEAST,FKB$B_FIPL(R5) ; SET FORK IPL
     1C A5   00  B0  03AB   992         MOVW    #PRQ$C_EXEC,PRQ$W_DISPATCH(R5) ; SET EXEC DISPATCHER ID
     20 A5   01  B0  03AF   993         MOVW    #PRQ$C_RESAVL,PRQ$W_REQTYPE(R5) ; SET RESOURCE AVAILABLE TYPE
          C0'AF  9F  03B3   994         PUSHAB  B^50$                           ; SET RETURN PC
       03C6'CF  9F  03B6   995         PUSHAB  W^REQUEST_DISP                  ; SET FORK PC
     00000000'GF  17  03BA   996         JMP     G^EXE$FORK                      ; CREATE FORK PROCESS
        5E   08  C0  03C0   997 50$:    ADDL    #8,SP                           ; REMOVE SAVED R3-R4
                     03C3   998 INT_EXIT:                                    ; EXIT INTERRUPT
              3F   BA  03C3   999         POPR    #^M<R0,R1,R2,R3,R4,R5>          ; RESTORE REGISTERS
                   02  03C5  1000         REI                                     ;
```

```
                                                N 1
                03C6  1002          .SBTTL  INTER-PROCESSOR REQUEST DISPATCHER
                03C6  1003 ;++
                03C6  1004 ;
                03C6  1005 ; FUNCTIONAL DESCRIPTION:
                03C6  1006 ;
                03C6  1007 ;       THIS ROUTINE IS CALLED BY THE FORK PROCESS DISPATCHER WHEN
                03C6  1008 ;       IT DISPATCHES A FORK BLOCK THAT IS AN INTER-PROCESSOR REQUEST
                03C6  1009 ;       BLOCK.
                03C6  1010 ;
                03C6  1011 ; INPUTS:
                03C6  1012 ;
                03C6  1013 ;       R0-R2 = SCRATCH.
                03C6  1014 ;       R3 = ADAPTER CONTROL BLOCK ADDRESS.
                03C6  1015 ;       R4 = SHARED MEMORY DATAPAGE ADDRESS.
                03C6  1016 ;       R5 = INTER-PROCESSOR REQUEST BLOCK ADDRESS.
                03C6  1017 ;
                03C6  1018 ; DISPATCHING OCCURS AS FOLLOWS:
                03C6  1019 ;
                03C6  1020 ;       THE REQUEST DISPATCHER ID CODE IS EXAMINED AND IF IT IS
                03C6  1021 ;       AN EXECUTIVE REQUEST (PRQ$C_EXEC) THEN THE EXEC REQUEST HANDLER
                03C6  1022 ;       IS CALLED.  IF IT IS NOT THE EXECUTIVE REQUEST ID, THE ID
                03C6  1023 ;       CODE IS USED AS AN INDEX INTO THE DRIVER DISPATCHER VECTOR
                03C6  1024 ;       TABLE TO CALL THE DRIVER INTER-PROCESSOR REQUEST DISPATCHER.
                03C6  1025 ;
                03C6  1026 ;       WHEN THE CALLED DISPATCHER RETURNS, THE REQUEST BLOCK IS
                03C6  1027 ;       DEALLOCATED TO THE SHARED MEMORY REQUEST QUEUE AND A RETURN
                03C6  1028 ;       TO THE FORK PROCESS DISPATCHER IS EXECUTED.
                03C6  1029 ;
                03C6  1030 ;--
                03C6  1031 REQUEST_DISP:                        ; PROCESSOR REQUEST DISPATCHER
     38    BB   03C6  1032          PUSHR   #^M<R3,R4,R5>        ; SAVE REGISTERS
                03C8  1033          ASSUME  PRQ$C_EXEC EQ 0
51   1C A5   3C  03C8  1034          MOVZWL  PRQ$W_DISPATCH(R5),R1 ; GET DISPATCHER ID
     12   12     03CC  1035          BNEQ    10$                  ; IF NEQ, NOT EXECUTIVE REQUEST
                03CE  1036 ;
                03CE  1037 ; CALL APPROPRIATE EXECUTIVE INTER-PROCESSOR REQUEST HANDLER
                03CE  1038 ;
     EC'AF   9F   03CE  1039          PUSHAB  B^15$                ; SET RETURN ADDRESS
                03D1  1040          CASE    PRQ$W_REQTYPE(R5),<-  ; CALL REQUEST HANDLER
                03D1  1041                  SETEF,-               ;  UPDATE EVENT FLAGS
                03D1  1042                  RESAVL-               ;  REPORT RESOURCE AVAILABLE
                03D1  1043                  >                     ;
                03DA  1044 5$:      BUG_CHECK UNKNPRQ             ; UNKNOWN REQUEST ID
     0C   11     03DE  1045          BRB     15$                  ;
                03E0  1046 ;
                03E0  1047 ; CALL CLASS DRIVER INTER-PROCESSOR REQUEST DISPATCHER
                03E0  1048 ;
                03E0  1049 10$:
50   10 A3   D0  03E0  1050          MOVL    ADP$L_VECTOR(R3),R0  ; GET ADDRESS OF VECTOR TABLE
 50   6041   D0  03E4  1051          MOVL    (R0)[R1],R0          ; GET ADDRESS OF DISPATCHER
     02   13     03E8  1052          BEQL    15$                  ; IF EQL, UNUSED VECTOR
                03EA  1053                                       ; (CAN OCCUR AFTER CRASH IF REQUESTS ARE LEF
     60   16     03EA  1054          JSB     (R0)                 ; CALL DRIVER DISPATCHER
                03EC  1055 15$:
     38   BA     03EC  1056          POPR    #^M<R3,R4,R5>        ; RESTORE REGISTERS
                03EE  1057 ;
                03EE  1058 ; DEALLOCATE THE REQUEST BLOCK
```

```
                                                                B 2
                     03EE  1059 ;
                     03EE  1060 DEALLOC_BLOCK:                               ; DEALLOCATE THE REQUEST BLOCK
                     03EE  1061         QRETRY  SUCCESS=10$-                 ; DEALLOCATE REQUEST BLOCK
                     03EE  1062         INSQTI  (R5),SHD$Q_PRQ(R4)           ;
                     03FF  1063         BUG_CHECK BADQHDR                    ; INSQTI FAILED - BAD QUEUE HEADER
                     0403  1064 10$:
     51   00A4 C4    3C    0403  1065         MOVZWL  SHD$W_PRQWAIT(R4),R1     ; AN PROCESSORS WAITING FOR A BLOCK?
          02   13    0408  1066         BEQL    20$                          ; IF EQL, NO
          01   10    040A  1067         BSBB    INTERRUPT_PORTS              ; NOTIFY WAITING PORTS
               05    040C  1068 20$:    RSB                                  ;
```

```
                                040D  1070          .SBTTL  INTERRUPT_PORTS - ROUTINE TO INTERRUPT SELECTED PORTS
                                040D  1071 ;++
                                040D  1072 ; FUNCTIONAL DESCRIPTION:
                                040D  1073 ;
                                040D  1074 ;       THIS ROUTINE SETS THE INTERRUPT REQUEST BITS FOR THE SELECTED PORTS.
                                040D  1075 ;
                                040D  1076 ; INPUTS:
                                040D  1077 ;
                                040D  1078 ;       R1 = MASK OF PORTS TO BE INTERRUPTED.
                                040D  1079 ;       R3 = ADAPTER CONTROL BLOCK ADDRESS
                                040D  1080 ;--
                                040D  1081 INTERRUPT_PORTS:                             ;
   50    34 A3    9A  040D  1082          MOVZBL  ADP$B_PORT(R3),R0                    ; GET OUR PORT NUMBER
      50    04    C4  0411  1083          MULL    #MPM$C_PORTS,R0                      ; COMPUTE INTERRUPT REQUEST BIT #
      50    10    C0  0414  1084          ADDL    #MPM$V_IIR_CTL,R0                    ; ...
50    51    50    78  0417  1085          ASHL    R0,R1,R0                             ; ...
      51    63    D0  041B  1086          MOVL    ADP$L_CSR(R3),R1                     ; GET ADAPTER CSR ADDRESS
   20 A1    50    D0  041E  1087          MOVL    R0,MPM$L_IIR(R1)                     ; SET PORTS' INTERRUPT REQUEST BIT(S)
                    05  0422  1088          RSB                                          ;
```

```
                                0423  1090          .SBTTL  UPDATE LOCAL COPY OF EVENT FLAG CLUSTER
                                0423  1091 ;++
                                0423  1092 ;
                                0423  1093 ; FUNCTIONAL DESCRIPTION:
                                0423  1094 ;
                                0423  1095 ;       THIS ROUTINE HANDLES THE INTER-PROCESSOR REQUEST TO COPY THE MASTER
                                0423  1096 ;       COMMON EVENT FLAGS INTO THE SLAVE COMMON EVENT BLOCK.  SOME PRQS
                                0423  1097 ;       MAY BE DELIVERED AFTER THE SLAVE COMMON EVENT BLOCK HAS BEEN
                                0423  1098 ;       DELETED.  THIS HAPPENS FREQUENTLY AFTER A PROCESSOR CRASHES
                                0423  1099 ;       AND REBOOTS.  THE LOGIC HANDLES THIS BY IGNORING THE PRQ.
                                0423  1100 ;
                                0423  1101 ; INPUTS:
                                0423  1102 ;
                                0423  1103 ;       R4 = SHARED MEMORY DATA PAGE ADDRESS
                                0423  1104 ;       R5 = INTER-PROCESSOR REQUEST BLOCK ADDRESS
                                0423  1105 ;
                                0423  1106 ; OUTPUTS:
                                0423  1107 ;
                                0423  1108 ;       R0 = SS$_NORMAL - SUCCESSFUL RETURN
                                0423  1109 ;--
                                0423  1110 SETEF:
                                0423  1111          DSBINT  #IPL$_SYNCH             ; RAISE TO SYNCH FOR REFCNT CHANGE
     50    24 A5    3C          0429  1112          MOVZWL  PRQ$L_PARAM(R5),R0      ; GET INDEX TO MASTER CEB
53   54    08 A4    C1          042D  1113          ADDL3   SHD$L_CEFPTR(R4),R4,R3  ; GET ADR OF 1ST MASTER CEB
     52    08 A3    3C          0432  1114          MOVZWL  CEB$W_SIZE(R3),R2       ; GET THE SIZE OF ONE MASTER CEB
        52    50    C4          0436  1115          MULL2   R0,R2                   ; GET BYTE OFFSET TO THIS MASTER
        53    52    C0          0439  1116          ADDL2   R2,R3                   ; R3=ADR OF MASTER CEB
     51    18 A5    3C          043C  1117          MOVZWL  PRQ$W_TO_PORT(R5),R1    ; RECEIVER PORT #
  51    38 A341    D0          0440  1118          MOVL    CEB$L_VASLAVE1(R3)[R1],R1 ; R1=ADR OF SLAVE CEB, OR 0
           2E    13          0445  1119          BEQL    40$                     ; BR IF SLAVE NO LONGER EXISTS
10 A1    10 A3    D0          0447  1120          MOVL    CEB$L_EFC(R3),CEB$L_EFC(R1) ; COPY FLAGS FROM MASTER
     0078 8F    BB          044C  1121          PUSHR   #^M<R3,R4,R5,R6>        ; SAVE REGISTERS
56    14 A1    9E          0450  1122          MOVAB   CEB$L_WQFL(R1),R6       ; GET HEAD OF WAIT QUEUE FOR CEFC
     52    01    9A          0454  1123          MOVZBL  #PRI$_IOCOM,R2          ; SET PRIORITY INCREMENT
     54    66    D0          0457  1124          MOVL    (R6),R4                 ; GET FIRST PCB IN WAIT QUEUE
     56    54    D1          045A  1125 20$:     CMPL    R4,R6                   ; IS THIS THE END OF THE QUEUE?
           12    13          045D  1126          BEQL    30$                     ; BR IF END OF QUEUE
     55    64    D0          045F  1127          MOVL    (R4),R5                 ; REMEMBER NEXT PCB IN QUEUE
50    FC A6    DE          0462  1128          MOVAL   <CEB$L_EFC-CEB$L_WQFL>(R6),R0 ; POINT TO EVENT FLAG MASK
00000000'GF    16          0466  1129          JSB     G^EXE$CHKWAIT2          ; CHECK IF THE PROCESS CAN RUN NOW
     54    55    D0          046C  1130          MOVL    R5,R4                   ; GET NEXT PCB IN WAIT QUEUE
           E9    11          046F  1131          BRB     20$                     ; CONTINUE LOOPING THROUGH ALL OF QUEUE
                             0471  1132 30$:                                    ; NO MORE PCB'S IN WAIT QUEUE
     0078 8F    BA          0471  1133          POPR    #^M<R3,R4,R5,R6>        ; RESTORE REGISTERS
                             0475  1134 40$:                                    ; NO SLAVE CEB EXISTED, SO NO WAITERS
     50    01    9A          0475  1135          MOVZBL  #SS$_NORMAL,R0          ; RETURN SUCCESS STATUS
                             0478  1136          ENBINT                          ; LOWER FROM SYNCH
                          05 047B  1137          RSB                             ; RETURN
```

```
                     047C  1139          .SBTTL  REPORT RESOURCE AVAILABILITY TO LOCAL SYSTEM
                     047C  1140 ;++
                     047C  1141 ;
                     047C  1142 ; FUNCTIONAL DESCRIPTION:
                     047C  1143 ;
                     047C  1144 ;       THIS ROUTINE HANDLES THE INTERPROCESSOR REQUEST TO REPORT
                     047C  1145 ;       THAT A RESOURCE IS AVAILABLE TO THE LOCAL SYSTEM.
                     047C  1146 ;
                     047C  1147 ; INPUTS:
                     047C  1148 ;
                     047C  1149 ;       R3 = ADAPTER CONTROL BLOCK ADDRESS.
                     047C  1150 ;       R4 = SHARED MEMORY DATAPAGE ADDRESS.
                     047C  1151 ;       R5 = INTER-PROCESSOR REQUEST BLOCK ADDRESS.
                     047C  1152 ;
                     047C  1153 ; OUTPUTS:
                     047C  1154 ;
                     047C  1155 ;       RESOURCE AVAILABILITY IS REPORTED, THEREBY UNBLOCKING ANY PROCESSES
                     047C  1156 ;       THAT ARE WAITING FOR THE RESOURCE.
                     047C  1157 ;--
                     047C  1158 RESAVL:                                      ;
                     047C  1159          DSBINT  #IPL$_SYNCH                 ; SYNCHRONIZE DATABASE ACCESS
     52   34 A3  9A  0482  1160          MOVZBL  ADP$B_PORT(R3),R2           ; GET OUR PORT NUMBER
          0C     BB  0486  1161          PUSHR   #^M<R2,R3>                  ; SAVE REGISTERS
        50   01   D0  0488  1162          MOVL    #1,R0                       ; INIT RESOURCE NUMBER
                     048B  1163
  51   00C8 C440  3E  048B  1164 10$:     MOVAW   SHD$W_RESAVAIL(R4)[R0],R1 ; GET ADDRESS OF AVAILABLE MASK
     10 61    6E  E7  0491  1165          BBCCI   (SP),(R1),20$               ; IF CLR, RESOURCE NOT AVAILABLE
  51   00A8 C440  3E  0495  1166          MOVAW   SHD$W_RESWAIT(R4)[R0],R1 ; GET ADDRESS OF WAIT MASK
     06 61    6E  E7  049B  1167          BBCCI   (SP),(R1),20$               ; IF CLR, NO PROCESSES WAITING
  00000000'GF   16  049F  1168          JSB     G^SCH$RAVAIL                ; REPORT RESOURCE AVAILABLE
     E2 50    0F  F2  04A5  1169 20$:     AOBLSS  #RSN$_MAX,R0,10$            ; INCREMENT RESOURCE NUMBER AND LOOP
          0C     BA  04A9  1170          POPR    #^M<R2,R3>                  ; RESTORE REGISTERS
                     04AB  1171          ENBINT                              ; RESTORE IPL
                 05  04AE  1172          RSB
                     04AF  1173
                     04AF  1175          .END
```

```
ADP$B_PORT                = 00000034
ADP$C_MPMADPLEN           = 00000044
ADP$L_AVECTOR             = 0000001C
ADP$L_CSR                 = 00000000
ADP$L_INTD                = 00000038
ADP$L_LINK                = 00000004
ADP$L_PRQQBL              = 00000018
ADP$L_PRQQFL              = 00000014
ADP$L_SHB                 = 00000030
ADP$L_VECTOR              = 00000010
ADP$W_ADPTYPE             = 0000000E
ADP$W_SIZE                = 00000008
ADP$W_TR                  = 0000000C
ADP$W_UMR_DIS             = 00000256
ADPLINK                     ********    X   02
ATS_MPM                   = 00000003
BLOCK_AVAIL                 00000322 R      02
BUG$_BADQHDR                ********    X   02
BUG$_UNKNPRQ                ********    X   02
C780_LIKE                 = 00000001
CEB$C_EFC                 = 00000010
CEB$L_VASLAVE1            = 00000038
CEB$L_WQFL                = 00000014
CEB$W_SIZE                = 00000008
CI$INITIAL                  0000001F RG     02
CI$INT                      00000000 RG     02
CI$SHUTDOWN                 0000001F RG     02
CPU_TYPE                  = 00000001
DCR_K_CLRPWRDN            = 00000200
DCR_K_CLRPWRUP            = 00000100
DCR_K_RESET               = 00004000
DDT$L_UNSOLINT            = 00000004
DEALLOC_BLOCK               000003EE R      02
DEQUEUE_BLOCK               000002FA R      02
DR$INITIAL                  0000003F RG     02
DR$INT                      00000024 RG     02
DR$SHUTDOWN                 0000003F RG     02
DR_DCR                      00000000
DYN$C_ADP                 = 00000001
EXE$ALONONPAGED             ********    X   02
EXE$CHKWAIT2                ********    X   02
EXE$FORK                    ********    X   02
EXE$GL_DEFFLAGS             ********    X   02
EXE$GL_LOCKRTRY             ********    X   02
EXE$GL_SCB                  ********    X   02
EXE$INT58                   ********    X   02
EXE$RH780_INT               ********    X   02
EXE$V_CRDENABL              ********    X   02
FKB$B_FIPL                = 0000000B
FKB$L_FPC                 = 0000000C
FKB$L_FR3                 = 00000010
IDB$L_ADP                 = 00000014
IDB$L_CSR                 = 00000000
IDB$L_OWNER               = 00000004
IDB$L_UCBLST              = 00000018
INI$MPMADP                  000000ED RG     02
INTERRUPT_PORTS             0000040D R      02
INT_EXIT                    000003C3 R      02
IO$GL_UBA_INTO              ********    X   02
IPL$_QUEUEAST             = 00000006
IPL$_SYNCH                = 00000008
MA$INITIAL                  0000018A RG     02
MA$INT                      000002CE R      02
MA$RAVAIL                   00000293 RG     02
MA$REQUEST                  000001F8 RG     02
MBA$INITIAL                 000000E4 RG     02
MBA$INT                     0000006C RG     02
MBA$L_AS                  = 00000410
MBA$L_CR                  = 00000004
MBA$L_CSR                 = 00000000
MBA$L_SR                  = 00000008
MBA$M_CR_IE               = 00000004
MBA$M_CR_INIT             = 00000001
MBA$M_CSR_PD              = 00800000
MMG$GL_SBICONF              ********    X   02
MPM$C_PORTS               = 00000004
MPM$L_CR                  = 00000004
MPM$L_CSR                 = 00000000
MPM$L_CSR1                = 00000C18
MPM$L_ERR                 = 00000010
MPM$L_IIE                 = 00000024
MPM$L_IIR                 = 00000020
MPM$L_INV                 = 0000000C
MPM$L_MR                  = 0000001C
MPM$L_SR                  = 00000008
MPM$M_CR_EIE              = 00000002
MPM$M_CR_ERRS             = FF000000
MPM$M_CR_MIE              = 00000001
MPM$M_CSR1_MIA            = 00000400
MPM$M_CSR_PU              = 00400000
MPM$M_ERR_ELR             = 10000000
MPM$M_ERR_ICRD            = 40000000
MPM$M_ERR_IMP             = 80000000
MPM$M_INV_STADR           = 7FF00000
MPM$M_SR_ACA              = 80000000
MPM$M_SR_AGP              = 10000000
MPM$M_SR_EIE              = 00000002
MPM$M_SR_IDL              = 00004000
MPM$M_SR_IT               = 00008000
MPM$M_SR_MXF              = 40000000
MPM$M_SR_SS               = 00002000
MPM$S_CSR_PORT            = 00000002
MPM$V_CSR_PORT            = 00000000
MPM$V_IIE_CTL             = 00000010
MPM$V_IIR_CTL             = 00000010
MPM$V_INV_ID              = 00000000
MPMINTD                     00000182 R      02
NOBLOCKS                    00000283 R      02
NUMMPMVEC                 = 00000010
PA_CNF                      00000000
PA_CNF_M_PDN              = 00800000
PA_CNF_M_PUP              = 00400000
PA_PMC                      00000004
PA_PMC_M_MIN              = 00000001
```

```
PR$_IPL                         =  00000012
PR$_SID_TYP730                  =  00000003
PR$_SID_TYP750                  =  00000002
PR$_SID_TYP780                  =  00000001
PR$_SID_TYP790                  =  00000004
PR$_SID_TYPUV1                  =  00000007
PRI$_IOCOM                      =  00000001
PRQ$C_EXEC                      =  00000000
PRQ$C_RESAVL                    =  00000001
PRQ$L_PARAM                     =  00000024
PRQ$W_DISPATCH                  =  0000001C
PRQ$W_REQTYPE                   =  00000020
PRQ$W_TO_PORT                   =  00000018
REQUEST_DISP                       000003C6 R      02
REQ_INTERRUPT                      0000C26A R      02
RESAVL                             0000047C R      02
RESOURCE_AVAIL                     00000378 R      02
RETURN_BLOCK                       00000232 R      02
RSN$_MAX                        =  0000000F
SCH$RAVAIL                         ******** X      02
SETEF                              00000423 R      02
SHB$L_ADP                       =  0000001C
SHB$L_DATAPAGE                  =  00000004
SHD$L_CEFPTR                    =  00000008
SHD$Q_PRQ                       =  000000F0
SHD$Q_PRQWRK                    =  00000100
SHD$W_POLL                      =  000000A6
SHD$W_PRQWAIT                   =  000000A4
SHD$W_RESAVAIL                  =  000000C8
SHD$W_RESSUM                    =  000000E8
SHD$W_RESWAIT                   =  000000A8
SIZ...                          =  00000006
SS$_BADQUEUEHDR                 =  00000394
SS$_NORMAL                      =  00000001
UBA$INITIAL                        00000045 RG     02
UBA$INTO                           00000060 RG     02
UBA$L_CR                        =  00000004
UBA$L_CSR                       =  00000000
UBA$L_SR                        =  00000008
UBA$M_CR_BRIE                   =  00000020
UBA$M_CR_CNFIE                  =  00000004
UBA$M_CR_IFSIE                  =  00000040
UBA$M_CR_SUEFIE                 =  00000008
UBA$M_CR_USEFIE                 =  00000010
UBA$V_CR_MRDSB                  =  0000001A
UBA_UNEXINT                        00000068 RG     02
UCB$B_SLAVE                     =  00000090
UCB$L_DDT                       =  00000088
UCB$L_FPC                       =  0000000C
UCB$L_FR3                       =  00000010
UCB$V_INT                       =  00000001
UCB$W_STS                       =  00000064
```

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	00000008 (8.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
SYSLOA	000004AF (1199.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	30	00:00:00.03	00:00:01.95
Command processing	106	00:00:00.44	00:00:04.39
Pass 1	554	00:00:14.85	00:00:56.01
Symbol table sort	0	00:00:02.25	00:00:07.96
Pass 2	196	00:00:03.19	00:00:12.12
Symbol table output	20	00:00:00.13	00:00:01.04
Psect synopsis output	2	00:00:00.01	00:00:00.01
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	910	00:00:20.90	00:01:23.49

The working set limit was 1800 pages.
138898 bytes (272 pages) of virtual memory were used to buffer the intermediate code.
There were 120 pages of symbol table space allocated to hold 2165 non-local and 52 local symbols.
1179 source lines were read in Pass 1, producing 17 object records in Pass 2.
44 pages of virtual memory were used to define 43 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYS.OBJ]LIB.MLB;1	30
_255DUA28:[SYSLIB]STARLET.MLB;2	8
TOTALS (all libraries)	38

2268 GETS were required to define 38 macros.

There were no errors, warnings or information messages.

```
M> CRO/LIS=LIS$:ADPSUB780/OBJ=OBJ$:ADPSUB780 MSRC$:CPUSW780/UPDATE=(ENH$:CPUSW780)+MSRC$:ADPSUB/UPDATE=(ENH$:ADPSUB)+EXECML$/LIB
```

ADPSUB780
LIS

ACKMSG
LIS

MCF790
SDL

MCDEF
MDL

ADPERR750
LIS

ADPSUB730
LIS

CSPDEF
SDL

CLUMBX
SDL

ADPERR780
LIS

ADPSUB750
LIS

CLUSTRMAC
MAR

CLUSTER
SDL

0392 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

CNXOPT
LIS

ADPSUBUV1
LIS

CNXMAN
LIS

CJFCLUSTR
LIS

CLUSTRLOA
LIS

CONMAN
LIS

ADPSUB790
LIS

CLUMESSAG
LIS