```
   SSSSSSSSSSSS  YYY         YYY     SSSSSSSSSSSS  LLL                 OOOOOOOOO        AAAAAAAAA
   SSSSSSSSSSSS  YYY         YYY     SSSSSSSSSSSS  LLL                 OOOOOOOOO        AAAAAAAAA
   SSSSSSSSSSSS  YYY         YYY     SSSSSSSSSSSS  LLL                 OOOOOOOOO        AAAAAAAAA
SSS              YYY         YYY  SSS              LLL              OOO         OOO  AAA         AAA
SSS              YYY         YYY  SSS              LLL              OOO         OOO  AAA         AAA
SSS              YYY         YYY  SSS              LLL              OOO         OOO  AAA         AAA
SSS                 YYY   YYY     SSS              LLL              OOO         OOO  AAA         AAA
SSS                 YYY   YYY     SSS              LLL              OOO         OOO  AAA         AAA
SSS                 YYY   YYY     SSS              LLL              OOO         OOO  AAA         AAA
   SSSSSSSSS           YYY           SSSSSSSSS     LLL              OOO         OOO  AAA         AAA
   SSSSSSSSS           YYY           SSSSSSSSS     LLL              OOO         OOO  AAA         AAA
   SSSSSSSSS           YYY           SSSSSSSSS     LLL              OOO         OOO  AAA         AAA
            SSS        YYY                    SSS  LLL              OOO         OOO  AAAAAAAAAAAAAAA
            SSS        YYY                    SSS  LLL              OOO         OOO  AAAAAAAAAAAAAAA
            SSS        YYY                    SSS  LLL              OOO         OOO  AAAAAAAAAAAAAAA
            SSS        YYY                    SSS  LLL              OOO         OOO  AAA         AAA
            SSS        YYY                    SSS  LLL              OOO         OOO  AAA         AAA
            SSS        YYY                    SSS  LLL              OOO         OOO  AAA         AAA
SSSSSSSSSSSS           YYY        SSSSSSSSSSSS     LLLLLLLLLLLLLLL     OOOOOOOOO     AAA         AAA
SSSSSSSSSSSS           YYY        SSSSSSSSSSSS     LLLLLLLLLLLLLLL     OOOOOOOOO     AAA         AAA
SSSSSSSSSSSS           YYY        SSSSSSSSSSSS     LLLLLLLLLLLLLLL     OOOOOOOOO     AAA         AAA
```

C 16

FILEID**ADPSUB780

```
  AAAAAA    DDDDDDDD    PPPPPPPP      SSSSSSSS  UU      UU  BBBBBBBB    7777777777    888888      000000
  AAAAAA    DDDDDDDD    PPPPPPPP      SSSSSSSS  UU      UU  BBBBBBBB    7777777777    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        .NOSHOW CONDITIONALS
0000        .TITLE  ADPSUB780 - ADAPTER SUBROUTINES FOR VAX 11/780
0000
0000
0000
0000
0000
0000        .IDENT  'V04-000'
0000
0000 ;****************************************************************************
0000 ;*                                                                          *
0000 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000 ;*  ALL RIGHTS RESERVED.                                                    *
0000 ;*                                                                          *
0000 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000 ;*  TRANSFERRED.                                                            *
0000 ;*                                                                          *
0000 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000 ;*  CORPORATION.                                                            *
0000 ;*                                                                          *
0000 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000 ;*                                                                          *
0000 ;*                                                                          *
0000 ;****************************************************************************
0000 ;
0000 ; Facility: System bootstrapping and initialization
0000 ;
0000 ; Abstract: This module contains initialization routines that are loaded
0000 ;             during system initialization (rather than linked into the system).
0000 ;
0000 ; Environment: Mode = KERNEL, Executing on INTERRUPT stack, IPL=31
0000 ;
0000 ; Author:  Kerbey T. Altmann               Creation date: 30-Oct-1982
0000 ;
0000 ; Modification history:
0000 ;
0000 ;       V03-007 TCM0002         Trudy C. Matthews       04-Jun-1984
0000 ;               Include more 780-specific code for the 11/790 version of
0000 ;               this routine.
0000 ;
0000 ;       V03-006 KPL0001         Peter Lieberwirth       12-Apr-1984
0000 ;               Init ADP$L_SHB properly again; V03-004 ASSUMEd this field
0000 ;               was at a certain constant offset, and a change to the ADP
0000 ;               moved it.  Note - this is a 780 change only.
0000 ;
0000 ;       V03-005 KDM0081         Kathleen D. Morse       13-Sep-1983
0000 ;               Create version for Micro-VAX I.
0000 ;
0000 ;       V03-004 ROW0196         Ralph O. Weber          27-JUL-1983
0000 ;               Correct INI$MPMADP so the ADP$L_SHB is correctly initialized
```

```
0000   ;               to zero.
0000   ;
0000   ;       V03-003 MSH0001         Maryann Hinden          06-Dec-1982
0000   ;               Add initialization for DW750.
0000   ;
0000   ;       V03-002 ROW0142         Ralph O. Weber          23-NOV-1982
0000   ;               Correct JMP in multiport memory interrupt dispatching code
0000   ;               prototype, MPMINTD, to a JSB.  MA$INT expects to receive
0000   ;               control via a JSB.
0000   ;
0000   ;       V03-001 TCM0001         Trudy C. Matthews       8-Nov-1982
0000   ;               Initialize field ADP$L_AVECTOR in INI$MPMADP.
0000   ;
0000   ;--
```

```
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   146          .PSECT  SYSLOA,LONG
```

```
0000            .SBTTL  CI$INT - CI INTERRUPT HANDLER
0000      ;+
0000      ; CI$INT - CI INTERRUPT HANDLER
0000      ;
0000      ;       THIS MODULE IS A DUMMY CI32 INTERRUPT HANDLER WHICH IS USED
0000      ;       UNTIL THE REAL CI DRIVER (PADRIVER) IS LOADED. IT ALSO CONTAINS
0000      ;       A DUMMY CI32 CONTROLLER INITIALIZATION  ENTRY POINT.
0000      ;
0000      ; INPUTS:
0000      ;
0000      ;       THE STACK ON ENTRY IS AS FOLLOWS:
0000      ;
0000      ;               0(SP)           ADDRESS OF IDB ADDRESS
0000      ;       4(SP) - 16(SP)          SAVED R2 - R5
0000      ;               20(SP)          INTERRUPT PC
0000      ;               24(SP)          INTERRUPT PSL
0000      ;
0000      ; OUTPUTS:
0000      ;
0000      ;       NONE
0000      ;
0000      ; SIDE EFFECTS:
0000      ;
0000      ;       INTERRUPTS ARE DISABLED ON THE CI32
0000      ;-
0000
0000
0000      ;
0000      ; $PAREGDEF -- Define offsets to CI registers and fields in the registers.
0000      ;
0000
0000            $DEFINI PAREG
0000
0000            $DEF    PA_CNF  .BLKL   1               ; Configuration register
0004
0004            _VIELD  PA_CNF,0,<-                     ; Define config register fields:
0004            <ADPTYP,8,M>,-                          ;  Adapter type code
0004            <PFD,,M>,-                              ;  Powerfail disable
0004            <TDEAD,,M>,-                            ;  Transmit dead
0004            <TFAIL,,M>,-                            ;  Transmit fail
0004            <,5>,-                                  ;  5 unused bits
0004            <CRD,,M>,-                              ;  CRD on port init'd read
0004            <RDS,,M>,-                              ;  RDS on port init'd read
0004            <CXTER,,M>,-                            ;  SBI error confirm
0004            <RDTO,,M>,-                             ;  Port init'd read timeout on SBI
0004            <CSTMO,,M>,-                            ;  Port init'd command xmit timeout
0004            <,1>,-                                  ;  1 unused bit
0004            <PUP,,M>,-                              ;  Adapter power up
0004            <PDN,,M>,-                              ;  Adaptor power down
0004            >                                       ;
0004
0004            $DEF    PA_PMC  .BLKL   1               ; Port maint control/status register
0008
0008            _VIELD  PA_PMC,0,<-                     ; Define register fields:
0008            <MIN,,M>,-                              ;  Maint initialized
0008            <MTD,,M>,-                              ;  Maint timer disable
0008            <MIE,,M>,-                              ;  Maint interrupt enable
```

```
                            0008          <MIF,,M>,-                        ;  Maint intterupt flag
                            0008          >                                 ;
                            0008
                            0008          $DEFEND PAREG
                            0000
                            0000  CI$INT::
           53    9E    D0   0000          MOVL    @(SP)+,R3                  ; GET ADDRESS OF IDB
           54    63    D0   0003          MOVL    IDB$L_CSR(R3),R4           ; GET ADDRESS OF FIRST CSR
   64  00400000  8F    D0   0006          MOVL    #PA_CNF_M_PUP,PA_CNF(R4)   ; CLEAR POWER UP
   64  00800000  8F    D0   000D          MOVL    #PA_CNF_M_PDN,PA_CNF(R4)   ; CLEAR POWER DOWN
        04 A4    01    D0   0014          MOVL    #PA_PMC_M_MIN,PA_PMC(R4)   ; SET MAINTENCE INITIALIZE
           52    8E    7D   0018          MOVQ    (SP)+,R2                   ; RESTORE REGISTERS
           54    8E    7D   001B          MOVQ    (SP)+,R4
                       02   001E          REI
                            001F
                            001F
                            001F  CI$INITIAL::                               ; CONTROLLER INITIALIZATION
                            001F  CI$SHUTDOWN::                              ; CONTROLLER SHUTDOWN
                            001F
                            001F
        04 A4    01    D0   001F          MOVL    #PA_PMC_M_MIN,PA_PMC(R4)   ; SET MAINTENCE INITIALIZE
                            0023
                       05   0023          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      DCR_K_CLRPWRDN=^X200               ; CLEAR POWER DOWN
                  00000300  0004      DCR_K_CLREXTABT=^X300              ; CLEAR EXTERNAL ABORT
                  00000400  0004      DCR_K_CLRABTINT=^X400              ; CLEAR ABORT INTERRUPT
                  00000500  0004      DCR_K_CLRINTENB=^X500              ; CLEAR INTERRUPT ENABLE
                  00000600  0004      DCR_K_SETINTENB=^X600              ; SET INTERRUPT ENABLE
                  00000700  0004      DCR_K_CLRHLT=^X700                 ; CLEAR HALT
                            0004
                            0004  ; DCR CONTROL FIELD B CODES (USED WHEN WRITING TO DCR)
                            0004
                  00001000  0004      DCR_K_CLRCRD=^X1000                ; CLEAR CRD
                  00002000  0004      DCR_K_SETEXTABT=^X2000             ; SET EXTERNAL ABORT
                  00003000  0004      DCR_K_CLRPKTINT=^X3000             ; CLEAR PACKET INTERRUPT
                  00004000  0004      DCR_K_RESET=^X4000                 ; RESET
                  00005000  0004      DCR_K_SETOSQTST=^X5000             ; SET OSEQ TEST
                  00006000  0004      DCR_K_CLROSQTST=^X6000             ; CLEAR OSEQ TEST
                            0004      $DEFEND DR
                            0024
                            0024  DR$INT::
            53    9E    D0  0024      MOVL    @(SP)+,R3                  ; GET ADDRESS OF IDB
            54    63    D0  0027      MOVL    IDB$L_CSR(R3),R4           ; GET ADDRESS OF FIRST CSR
   64  00000100 8F    D0  002A      MOVL    #DCR_K_CLRPWRUP,DR_DCR(R4) ; CLEAR POWER UP
   64  00000200 8F    D0  0031      MOVL    #DCR_K_CLRPWRDN,DR_DCR(R4) ; CLEAR POWER DOWN
            52    8E    7D  0038      MOVQ    (SP)+,R2                   ; RESTORE REGISTERS
            54    8E    7D  003B      MOVQ    (SP)+,R4
                        02  003E      REI
                            003F
                            003F
                            003F  DR$INITIAL::                           ; CONTROLLER INITIALIZATION
                            003F  DR$SHUTDOWN::                          ; CONTROLLER SHUTDOWN
                            003F
                            003F
      64   4000 8F    3C  003F      MOVZWL  #DCR_K_RESET,(R4)          ; RESET DR (R4 POINTS TO CSR)
                            0044
                        05  0044      RSB
```

```
                                 0045           .SBTTL  UBA$INITIAL - CPU-DEPENDENT UNIBUS ADAPTER INITIALIZATION
                                 0045  ;+
                                 0045  ; UBA$INITIAL - UNIBUS ADAPTER INITIALIZATION
                                 0045  ;
                                 0045  ; THIS ROUTINE IS CALLED VIA A JSB INSTRUCTION AT SYSTEM STARTUP AND AFTER
                                 0045  ; A POWER RECOVERY RESTART TO ALLOW INITIALIZATION OF UNIBUS ADAPTERS.
                                 0045  ;        (POWERFAIL AND INITADP)
                                 0045  ;
                                 0045  ; INPUTS:
                                 0045  ;
                                 0045  ;        R2 = ADDRESS OF ADAPTER CONTROL BLOCK  (11/780 AND 11/750)
                                 0045  ;        R4 = ADDRESS OF UNIBUS ADAPTER CONFIGURATION STATUS REGISTER (11/780)
                                 0045  ;
                                 0045  ;        ALL INTERRUPTS ARE LOCKED OUT.
                                 0045  ;
                                 0045  ; OUTPUTS:
                                 0045  ;
                                 0045  ;        THE UNIBUS ADAPTER IS INITIALIZED AND INTERRUPTS ARE ENABLED.
                                 0045  ;-
                                 0045
                                 0045  UBA$INITIAL::                              ;UNIBUS ADAPTER INITIALIZATION
                                 0045
                                 0045                                             ;
                64      00   D2  0045          MCOML   #0,UBA$L_CSR(R4)           ;CLEAR ALL ADAPTER CONFIGURATION ERRORS
             08 A4      00   D2  0048          MCOML   #0,UBA$L_SR(R4)            ;CLEAR ALL ADAPTER STATUS BITS
          50   0256     C2   3C  004C          MOVZWL  ADP$W_UMR_DIS(R2),R0       ;PICK UP THE NUMBER OF UMR'S TO DISABLE
          50     50     16   78  0051          ASHL    #UBA$V_CR_MRDSB-4,R0,R0    ;DIVIDE BY 16 THEN SHIFT INTO POSITION
                             C9  0055          BISL3   #UBA$M_CR_SUEFIE!-         ;ENABLE INTERRUPTS
                                 0056                   UBA$M_CR_BRIE!-
                                 0056                   UBA$M_CR_CNFIE!-
                                 0056                   UBA$M_CR_USEFIE!-
                                 0056                   UBA$M_CR_IFSIE,-
   04 A4    50   0000007C 8F     0056                  R0,UBA$L_CR(R4)
                                 005E
                                 005E
                                 005E
                                 005E  10$:                                       ;NO SPECIAL INIT FOR 11/730 OR uVAX I
                             05  005E          RSB                                ;
                                 005F  ;
                                 005F  ; IGNORE UNEXPECTED UNIBUS INTERRUPTS
                                 005F  ;
                                 005F
                                 005F          .ALIGN  LONG
                                 0060
                                 0060  UBA$INTO::                                 ; PASSIVE RELEASES THROUGH VECTOR 0
                                 0060
                00000000'9F  D6  0060          INCL    @#IO$GL_UBA_INTO           ; COUNT THEM
                         00  11  0066          BRB     UBA_UNEXINT                ; JOIN COMMON CODE, VECTORS ARE ALLIGNED
                                 0068
                                 0068          .ALIGN  LONG
                                 0068
                                 0068  ;
                                 0068  ; NOTE: UBA$UNEXINT is the label in the EXEC that is a JMP @#UBA_UNEXINT.
                                 0068  ;       This seeming duplicity is necesary since there is code that must
                                 0068  ;       refer to the EXEC address from within the SYSLOA image.
                                 0068  ;
                                 0068  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
```

```
        55    18 A342   D0  00AD   20$:    MOVL    IDB$L_UCBLST(R3)[R2],R5 ;GET ADDRESS OF UCB OR INTERRUPT DISPATCHER
                 22 55  E8  00B2           BLBS    R5,40$                  ;IF LBS INTERRUPT DISPATCHER FOR MULTI-
                            00B5                                           ; DEVICE CONTROLLER
   0410 C4    01    52  78  00B5           ASHL    R2,#1,MBA$L_AS(R4)      ;CLEAR ATTENTION SUMMARY BIT
                    55  D5  00BB           TSTL    R5                      ;SEE IF UCB DEFINED
                    CD  13  00BD           BEQL    10$                     ;IF EQL NONE DEFINED
      09 64 A5      01  E5  00BF           BBCC    #UCB$V_INT,UCB$W_STS(R5),30$ ;IF CLR, INTERRUPT NOT EXPECTED
        53    10 A5     7D  00C4           MOVQ    UCB$L_FR3(R5),R3        ;RESTORE DRIVER CONTEXT
              0C B5     16  00C8           JSB     @UCB$L_FPC(R5)          ;CALL DRIVER AT INTERRUPT RETURN ADDRESS
                    BF  11  00CB           BRB     10$                     ;
                            00CD
        53    0088 C5   D0  00CD   30$:    MOVL    UCB$L_DDT(R5),R3        ;GET ADDRESS OF DDT
                 04 B3  16  00D2           JSB     @DDT$L_UNSOLINT(R3)     ;CALL UNSOLICITED INTERRUPT ROUTINE
                    B5  11  00D5           BRB     10$                     ;
                            00D7
                    7E  DC  00D7   40$:    MOVPSL  -(SP)                   ;READ CURRENT PSL
                    75  16  00D9           JSB     -(R5)                   ;CALL SLAVE CONTROLLER INTERRUPT DISPATCHER
                    AF  11  00DB           BRB     10$                     ;
                            00DD
                            00DD
                            00DD   ;
                            00DD   ; IN CASE OF ADAPTER POWER DOWN BIT ASSERTED, RETRIEVE ADP ADDRESS AND JUMP
                            00DD   ; TO ADAPTER ERROR ROUTINE IN SYSLOA780.
                            00DD   ;
                            00DD
        54    14 A3     D0  00DD   45$:    MOVL    IDB$L_ADP(R3),R4        ;GET ADP ADDRESS
              FF1C'     31  00E1           BRW     EXE$RH780_INT           ;JUMP TO ERROR ROUTINE
                            00E4
                            00E4
```

```
                00E4         .SBTTL  MASSBUS ADAPTER INITIALIZATION
                00E4   ;+
                00E4   ; MBA$INITIAL - MASSBUS ADAPTER INITIALIZATION
                00E4   ;
                00E4   ; THIS ROUTINE IS CALLED VIA A JSB INSTRUCTION AT SYSTEM STARTUP AND AFTER
                00E4   ; A POWER RECOVERY RESTART TO ALLOW INITIALIZATION OF MASSBUS ADAPTERS.
                00E4   ;
                00E4   ; INPUTS:
                00E4   ;
                00E4   ;       R4 = CSR ADDRESS OF MASSBUS ADAPTER.
                00E4   ;       R5 = ADDRESS OF ADAPTER IDB.
                00E4   ;
                00E4   ;       ALL INTERRUPTS ARE LOCKED OUT.
                00E4   ;
                00E4   ; OUTPUTS:
                00E4   ;
                00E4   ;       THE MASSBUS ADAPTER IS INITIALIZED AND INTERRUPTS ARE ENABLED.
                00E4   ;-
                00E4
                00E4   MBA$INITIAL::                                    ;MASSBUS ADAPTER INITIALIZATION
                00E4
      01    D0  00E4         MOVL    #MBA$M_CR_INIT,-
   04 A4        00E6                 MBA$L_CR(R4)               ;INITIALIZE MASSBUS ADAPTER
      04    D0  00E8         MOVL    #MBA$M_CR_IE,-
   04 A4        00EA                 MBA$L_CR(R4)               ;ENABLE INTERRUPTS
                00EC
            05  00EC         RSB
```

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

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

```
                            018A          .SBTTL  MA$INITIAL - INITIALIZE MULTI-PORT MEMORY ADAPTER
                            018A  ;++
                            018A  ;
                            018A  ; MPM$INITIAL - INITIALIZE MULTI-PORT MEMORY ADAPTER
                            018A  ;
                            018A  ; THIS ROUTINE IS CALLED AT SYSTEM INTIALIZATION AND AFTER A POWER
                            018A  ; RECOVERY RESTART TO INITIALIZE THE PORT ADAPTER BY CLEARING ANY
                            018A  ; ERRORS AND ENABLING ALL INTERRUPTS.
                            018A  ;
                            018A  ; INPUTS:
                            018A  ;
                            018A  ;       R4 = ADDR OF ADAPTER CSR.
                            018A  ;
                            018A  ;       IPL = 31
                            018A  ;
                            018A  ; OUPUTS:
                            018A  ;
                            018A  ;       ANY ERRORS IN PORT ARE CLEARED AND ALL INTERRUPTS ARE ENABLED.
                            018A  ;--
                            018A
                            018A  MA$INITIAL::                                   ; INTIALIZE PORT
                            018A
              04 A4     D4  018A          CLRL    MPM$L_CR(R4)                   ; CLEAR INTERRUPT ENABLE AND DIAG BITS
              08 A4     D4  018D          CLRL    MPM$L_SR(R4)                   ; CLEAR INVALIDATE DISABLE AND DIAG BITS
         50   0C A4     D0  0190          MOVL    MPM$L_INV(R4),R0               ; GET INVALIDATION REGISTER CONTENTS
    50  800FFFFF 8F     CA  0194          BICL    #^C<MPM$M_INV_STADR>,R0        ; CLEAR ALL BITS BUT STARTING ADDRESS
                            019B          ASSUME  MPM$V_INV_ID EQ 0              ; CACHED ID'S START AT BIT 0
  0C A4    50    01     C9  019B          BISL3   #1@0,R0,MPM$L_INV(R4)          ; SET CPU (NEXUS 0) AS CACHED
              18 A4     D4  01A0          CLRL    MPM$L_CSR1(R4)                 ; CLEAR DIAGNOSTIC BITS
              1C A4     D4  01A3          CLRL    MPM$L_MR(R4)                   ; CLEAR DIAGNOSTIC BITS
         00400000 8F    D0  01A6          MOVL    #MPM$M_CSR_PU,-                ; CLEAR ANY POWER-UP STATUS
                   64       01AC                  MPM$L_CSR(R4)
         FF000000 8F    D0  01AD          MOVL    #MPM$M_CR_ERRS,-               ; CLEAR ANY PORT ERRORS
                04 A4       01B3                  MPM$L_CR(R4)
                        D0  01B5          MOVL    #MPM$M_SR_SS!-                 ; CLEAR ANY STATUS ERRORS AND
                            01B6                  MPM$M_SR_IDL!-                 ;  ENABLF ERROR INTERRUPTS
                            01B6                  MPM$M_SR_IT!-
                            01B6                  MPM$M_SR_AGP!-
                            01B6                  MPM$M_SR_MXF!-
                            01B6                  MPM$M_SR_ACA!-
                            01B6                  MPM$M_SR_EIE,-
  08 A4    D000E002 8F      01B6                  MPM$L_SR(R4)
                        D0  01BD          MOVL    #MPM$M_ERR_ELR!-               ; CLEAR ANY ARRAY ERRORS
                            01BE                  MPM$M_ERR_IMP,-
  10 A4    90000000 8F      01BE                  MPM$L_ERR(R4)
                   00'  E0  01C5          BBS     S^#EXE$V_CRDENABL,-            ; BRANCH IF CRD ERROR LOGGING ENABLED
       08 00000000'GF       01C7                  G^EXE$GL_DEFFLAGS,10$
         40000000 8F    D0  01CD          MOVL    #MPM$M_ERR_ICRD,-              ; ELSE, INHIBIT CRD ERROR LOGGING
              10 A4         01D3                  MPM$L_ERR(R4)
         00000400 8F    D0  01D5  10$:    MOVL    #MPM$M_CSR1_MIA,-              ; CLEAR ANY ERROR
              18 A4         01DB                  MPM$L_CSR1(R4)
              1C A4     D4  01DD          CLRL    MPM$L_MR(R4)                   ; CLEAR ANY DIAGNOSTIC SETTINGS
         50   64        D0  01E0          MOVL    MPM$L_CSR(R4),R0               ; GET CSR REGISTER
              00        EF  01E3          EXTZV   #MPM$V_CSR_PORT,-              ; GET PORT NUMBER
    50   50   02            01E5                  #MPM$S_CSR_PORT,R0,R0
         50   04        C4  01E8          MULL    #MPM$C_PORTS,R0                ; COMPUTE INTERRUPT ENABLE BIT #
         50   10        C0  01EB          ADDL    #MPM$V_IIE_CTL,R0              ; ...
```

```
           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
```

```
                      01F8       .SBTTL  INTER-PROCESSOR REQUEST HANDLER
                      01F8  ;++
                      01F8  ;
                      01F8  ; FUNCTIONAL DESCRIPTION:
                      01F8  ;
                      01F8  ;        THIS ROUTINE IS CALLED BY A DRIVER OR AN EXEC FUNCTION TO
                      01F8  ;        EITHER SEND A REQUEST TO OR JUST INTERRUPT ANOTHER PROCESSOR
                      01F8  ;        THAT IS CONNECTED TO A PORT OF THE MULTIPORT MEMORY.
                      01F8  ;
                      01F8  ; INPUTS:
                      01F8  ;
                      01F8  ;        R4 = ADAPTER CONTROL BLOCK ADDRESS.
                      01F8  ;        R5 = IF LSS 0 - ADDRESS OF A FORK BLOCK TO USE IF REQUEST
                      01F8  ;                        BLOCK IS NOT AVAILABLE.
                      01F8  ;             IF GEQ 0 - PORT NUMBER OF PROCESSOR TO JUST INTERRUPT.
                      01F8  ;
                      01F8  ; OUTPUTS:
                      01F8  ;
                      01F8  ;        WHEN THIS ROUTINE IS CALLED WITH A FORK BLOCK ADDRESS, IT WILL
                      01F8  ;        ATTEMPT TO ALLOCATE A REQUEST BLOCK.  IF THE REQUEST FAILS,
                      01F8  ;        THE CONTEXT OF THE CALLER WILL BE SAVED IN THE FORK BLOCK, THE
                      01F8  ;        FORK BLOCK BLOCK WILL BE INSERTED IN THE REQUEST BLOCK WAIT
                      01F8  ;        QUEUE AND A RETURN TO THE CALLER'S CALLER IS EXECUTED.
                      01F8  ;
                      01F8  ;        IF A REQUEST BLOCK IS ALLOCATED SUCCESSFULLY, CONTROL WILL
                      01F8  ;        RETURN TO THE CALLER VIA A CO-ROUTINE CALL SO THE CALLER CAN
                      01F8  ;        FILL-IN THE REQUEST BLOCK.
                      01F8  ;
                      01F8  ;        THE CALLER WILL THEN PERFORM ANOTHER CO-ROUTINE CALL TO RETURN
                      01F8  ;        TO THIS ROUTINE SO THE BLOCK CAN BE INSERTED IN THE DESIRED
                      01F8  ;        PROCESSOR'S INTER-PROCESSOR REQUEST QUEUE.  IF IT IS THE
                      01F8  ;        FIRST REQUEST IN THE QUEUE AN INTER-PORT INTERRUPT WILL
                      01F8  ;        ALSO BE REQUESTED TO WAKE-UP THE DISPATCHER ON THE PORT.
                      01F8  ;
                      01F8  ;
                      01F8  ;        IF THIS ROUTINE IS CALLED WITH A PORT NUMBER INSTEAD OF A
                      01F8  ;        FORK BLOCK ADDRESS, IT WILL JUST REQUEST AN INTERRUPT FOR
                      01F8  ;        THE PROCESSOR ON THE SPECIFIED PORT.  IT IS THEN UP TO THE
                      01F8  ;        INTERRUPTED PROCESSOR TO DETERMINE WHAT THE INTERRUPT WAS
                      01F8  ;        FOR.
                      01F8  ;
                      01F8  ;        R0 = SUCCESS OR FAILURE OF OPERATION.  THIS SHOULD BE CHECKED
                      01F8  ;             BY THE CALLER BOTH TIMES THIS ROUTINE RETURNS.
                      01F8  ;
                      01F8  ;        R3,R4,R5 ARE PRESERVED.
                      01F8  ;
                      01F8  ;--
                      01F8
                      01F8 MA$REQUEST::                                     ; REQUEST HANDLER
                      01F8
       51    55   D0  01F8        MOVL    R5,R1                         ; FORK BLOCK ADDRESS SPECIFIED?
             6D   18  01FB        BGEQ    REQ_INTERRUPT                 ; IF GEQ, NO - JUST AN INTERRUPT
    51    30 A4   D0  01FD        MOVL    ADP$L_SHB(R4),R1              ; GET SHB ADDRESS
    51    04 A1   D0  0201        MOVL    SHB$L_DATAPAGE(R1),R1         ; GET DATAPAGE ADDRESS
    50    34 A4   9A  0205        MOVZBL  ADP$B_PORT(R4),R0             ; GET OUR PORT NUMBER
00 00A4 C1   50   E6  0209        BBSSI   R0,SHD$W_PRQWAIT(R1),5$       ; ASSUME FAILURE (AVOID MISSING NOTIFICATION
                      020F 5$:    QRETRY  SUCCESS=10$-                  ; GET A REQUEST BLOCK
```

```
                     020F             REMQHI  SHD$Q_PRQ(R1),R2           ;
     50  0394 8F  3C 0220             MOVZWL  #SS$_BADQUEUEHDR,R0        ; SET FAILURE STATUS CODE
                  05 0225             RSB
                     0226     10$:
              5B  1D 0226             BVS     NOBLOCKS                   ; IF V-SET, NO BLOCKS LEFT
     50    34 A4  9A 0228             MOVZBL  ADP$B_PORT(R4),R0          ; GET OUR PORT NUMBER
00 00A4 C1    50  E7 022C             BBCCI   R0,SHD$W_PRQWAIT(R1),RETURN_BLOCK ; CLEAR WAITING FLAG
                     0232     ;
                     0232     ; RETURN ADDRESS OF REQUEST BLOCK TO CALLER
                     0232     ;
                     0232     RETURN_BLOCK:                              ; RETURN ADDRESS OF BLOCK
     50       01  D0 0232             MOVL    #SS$_NORMAL,R0             ; SET SUCCESS
              9E  16 0235             JSB     @(SP)+                     ; CO-ROUTINE CALL CALLER BACK
                     0237     ;
                     0237     ; INSERT BLOCK IN DESIRED PORT'S REQUEST QUEUE
                     0237     ;
              54  DD 0237             PUSHL   R4                         ; SAVE REGISTER
     51    18 A2  3C 0239             MOVZWL  PRQ$W_TO_PORT(R2),R1       ; GET DESIRED PORT NUMBER
     50    30 A4  D0 023D             MOVL    ADP$L_SHB(R4),R0           ; GET SHB ADDRESS
     54    04 A0  D0 0241             MOVL    SHB$L_DATAPAGE(R0),R4      ; GET DATAPAGE ADDRESS
                     0245             QRETRY  SUCCESS=10$-               ; INSERT REQUEST IN PORT'S WORK QUEUE
                     0245             INSQTI  (R2),SHD$Q_PRQWRK(R4)[R1]
     50  0394 8F  3C 0257             MOVZWL  #SS$_BADQUEUEHDR,R0        ; SET FAILURE STATUS CODE
              05  11 025C             BRB     20$                        ;
              07  13 025E     10$:    BEQL    30$                        ; IF EQL, FIRST ENTRY IN QUEUE
     50       01  D0 0260             MOVL    #SS$_NORMAL,R0             ; SET SUCCESS
           54 8ED0   0263     20$:    POPL    R4                         ; RESTORE REGISTER
                  05 0266             RSB                                ;
                     0267     ;
           54 8ED0   0267     30$:    POPL    R4                         ; RESTORE REGISTER
                     026A     ;
                     026A     ; REQUEST AN INTER-PORT INTERRUPT TO WAKE-UP PROCESSOR ON DESIRED
                     026A     ; PORT.
                     026A     ;
                     026A     REQ_INTERRUPT:                             ; REQUEST AN INTER-PORT INTERRUPT
     50    34 A4  9A 026A             MOVZBL  ADP$B_PORT(R4),R0          ; GET OUR PORT NUMBER
         50   04  C4 026E             MULL    #MPM$C_PORTS,R0            ; COMPUTE INTERRUPT REQUEST BIT #
         50   51  C0 0271             ADDL    R1,R0                      ; ...
         50   10  C0 0274             ADDL    #MPM$V_IIR_CTL,R0          ; ...
         51   64  D0 0277             MOVL    ADP$L_CSR(R4),R1           ; GET ADAPTER CSR ADDRESS
20 A1    01   50  78 027A             ASHL    R0,#1,MPM$L_IIR(R1)        ; SET PORT INTERRUPT REQUEST BIT
         50   01  D0 027F             MOVL    #SS$_NORMAL,R0             ; SET SUCCESS
                  05 0282             RSB                                ;
                     0283     ;
                     0283     ; NO BLOCKS ARE AVAILABLE.  SAVE THE CALLER'S CONTEXT IN THE FORK
                     0283     ; BLOCK, INSERT THE FORK BLOCK IN THE REQUEST BLOCK WAIT QUEUE, AND
                     0283     ; RETURN TO THE CALLER'S CALLER.
                     0283     ;
                     0283     NOBLOCKS:                                  ; NO REQUEST BLOCKS AVAILABLE
   10 A5      53  7D 0283             MOVQ    R3,FKB$L_FR3(R5)           ; SAVE REGISTERS
      0C A5 8ED0     0287             POPL    FKB$L_FPC(R5)              ; SAVE RETURN ADDRESS
   18 B4      65  0E 028B             INSQUE  (R5),@ADP$L_PRQQBL(R4)     ; INSERT FORK BLOCK IN WAIT QUEUE
      50      01  D0 028F             MOVL    #SS$_NORMAL,R0             ; SET SUCCESS
              05     0292             RSB                                ; RETURN TO CALLER'S CALLER
```

```
                             0293            .SBTTL  REPORT RESOURCE AVAILABILITY TO INTERESTED PORTS
                             0293       ;++
                             0293       ;
                             0293       ; FUNCTIONAL DESCRIPTION:
                             0293       ;
                             0293       ;       THIS ROUTINE IS CALLED TO REPORT TO ANY PROCESSORS THAT A RESOURCE
                             0293       ;       HAS BEEN MADE AVAILABLE.
                             0293       ;
                             0293       ; INPUTS:
                             0293       ;
                             0293       ;       R0 = RESOURCE NUMBER OF RESOURCE MADE AVAILABLE.
                             0293       ;       R1 = SHARED MEMORY CONTROL BLOCK (SHB) ADDRESS.
                             0293       ;
                             0293       ; OUTPUTS:
                             0293       ;
                             0293       ;       ANY PROCESSORS WAITING FOR THE SPECIFIED RESOURCE ARE INTERRUPTED
                             0293       ;       TO NOTIFY THEM THE RESOURCE IS AVAILABLE.
                             0293       ;
                             0293       ;       R0,R1,R2,R3 ARE NOT PRESERVED.
                             0293       ;--
                             0293
                             0293       MA$RAVAIL::
                             0293
         52    04 A1    D0   0293               MOVL    SHB$L_DATAPAGE(R1),R2     ; GET ADDRESS OF DATAPAGE
            00A8 C240    B5   0297               TSTW    SHD$W_RESWAIT(R2)[R0]     ; ANYONE WAITING FOR THE RESOURCE?
                   2F    13   029C               BEQL    30$                       ; IF EQL, NO
                1C A1    DD   029E               PUSHL   SHB$L_ADP(R1)             ; SAVE ADDRESS OF ADAPTER CONTROL BLOCK
                   53    D4   02A1               CLRL    R3                        ; INIT PORT NUMBER
                              02A3
   51       00A8 C240    3E   02A3       10$:    MOVAW   SHD$W_RESWAIT(R2)[R0],R1  ; GET ADDRESS OF RESOURCE WAIT MASK
          10 61    53    E1   02A9               BBC     R3,(R1),20$               ; IF CLR, NO ONE WAITING AT PORT
   51       00C8 C240    3E   02AD               MOVAW   SHD$W_RESAVAIL(R2)[R0],R ; GET ADDRESS OF AVAILABLE MASK
          00 61    53    E6   02B3               BBSSI   R3,(R1),15$               ; SET PORT'S RESOURCE AVAIL BIT
   00 00E8 C2      53    E6   02B7       15$:    BBSSI   R3,SHD$W_RESSUM(R2),20$   ; SET PORT'S RESOURCE AVAIL SUMMARY BIT
          E2 53    04    F2   02BD       20$:    AOBLSS  #MPM$C_PORTS,R3,10$       ; INCREMENT PORT NUMBER AND LOOP
                53     8ED0   02C1               POPL    R3                        ; GET ADDRESS OF ADAPTER CONTROL BLOCK
   51       00A8 C240    3C   02C4               MOVZWL  SHD$W_RESWAIT(R2)[R0],R1  ; GET RESOURCE WAIT MASK
                 0140    30   02CA               BSBW    INTERRUPT_PORTS           ; INTERRUPT WAITING PORTS
                         05   02CD       30$:    RSB                               ;
```

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

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

```
                          03C6            .SBTTL  INTER-PROCESSOR REQUEST DISPATCHER
                          03C6      ;++
                          03C6      ;
                          03C6      ; FUNCTIONAL DESCRIPTION:
                          03C6      ;
                          03C6      ;       THIS ROUTINE IS CALLED BY THE FORK PROCESS DISPATCHER WHEN
                          03C6      ;       IT DISPATCHES A FORK BLOCK THAT IS AN INTER-PROCESSOR REQUEST
                          03C6      ;       BLOCK.
                          03C6      ;
                          03C6      ; INPUTS:
                          03C6      ;
                          03C6      ;       R0-R2 = SCRATCH.
                          03C6      ;       R3 = ADAPTER CONTROL BLOCK ADDRESS.
                          03C6      ;       R4 = SHARED MEMORY DATAPAGE ADDRESS.
                          03C6      ;       R5 = INTER-PROCESSOR REQUEST BLOCK ADDRESS.
                          03C6      ;
                          03C6      ; DISPATCHING OCCURS AS FOLLOWS:
                          03C6      ;
                          03C6      ;       THE REQUEST DISPATCHER ID CODE IS EXAMINED AND IF IT IS
                          03C6      ;       AN EXECUTIVE REQUEST (PRQ$C_EXEC) THEN THE EXEC REQUEST HANDLER
                          03C6      ;       IS CALLED.  IF IT IS NOT THE EXECUTIVE REQUEST ID, THE ID
                          03C6      ;       CODE IS USED AS AN INDEX INTO THE DRIVER DISPATCHER VECTOR
                          03C6      ;       TABLE TO CALL THE DRIVER INTER-PROCESSOR REQUEST DISPATCHER.
                          03C6      ;
                          03C6      ;       WHEN THE CALLED DISPATCHER RETURNS, THE REQUEST BLOCK IS
                          03C6      ;       DEALLOCATED TO THE SHARED MEMORY REQUEST QUEUE AND A RETURN
                          03C6      ;       TO THE FORK PROCESS DISPATCHER IS EXECUTED.
                          03C6      ;
                          03C6      ;--
                          03C6      REQUEST_DISP:                               ; PROCESSOR REQUEST DISPATCHER
             38     BB    03C6            PUSHR   #^M<R3,R4,R5>                 ; SAVE REGISTERS
                          03C8            ASSUME  PRQ$C_EXEC EQ 0
  51    1C A5       3C    03C8            MOVZWL  PRQ$W_DISPATCH(R5),R1         ; GET DISPATCHER ID
             12     12    03CC            BNEQ    10$                           ; IF NEQ, NOT EXECUTIVE REQUEST
                          03CE      ;
                          03CE      ; CALL APPROPRIATE EXECUTIVE INTER-PROCESSOR REQUEST HANDLER
                          03CE      ;
          EC'AF     9F    03CE            PUSHAB  B^15$                         ; SET RETURN ADDRESS
                          03D1            CASE    PRQ$W_REQTYPE(R5),<-          ; CALL REQUEST HANDLER
                          03D1                    SETEF,-                       ;  UPDATE EVENT FLAGS
                          03D1                    RESAVL-                       ;  REPORT RESOURCE AVAILABLE
                          03D1                    >                             ;
                          03DA      5$:   BUG_CHECK UNKNPRQ                     ; UNKNOWN REQUEST ID
             0C     11    03DE            BRB     15$                           ;
                          03E0      ;
                          03E0      ; CALL CLASS DRIVER INTER-PROCESSOR REQUEST DISPATCHER
                          03E0      ;
                          03E0      10$:
  50    10 A3       D0    03E0            MOVL    ADP$L_VECTOR(R3),R0           ; GET ADDRESS OF VECTOR TABLE
   50    6041       D0    03E4            MOVL    (R0)[R1],R0                   ; GET ADDRESS OF DISPATCHER
             02     13    03E8            BEQL    15$                           ; IF EQL, UNUSED VECTOR
                          03EA                                                  ; (CAN OCCUR AFTER CRASH IF REQUESTS ARE LEF
             60     16    03EA            JSB     (R0)                          ; CALL DRIVER DISPATCHER
                          03EC      15$:
             38     BA    03EC            POPR    #^M<R3,R4,R5>                 ; RESTORE REGISTERS
                          03EE      ;
                          03EE      ; DEALLOCATE THE REQUEST BLOCK
```

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

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

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

```
ADP$B_PORT                      = 00000034                  INT_EXIT                        000003C3 R      02
ADP$C_MPMADPLEN                 = 00000044                  IO$GL_UBA_INTO                  ********    X   02
ADP$L_AVECTOR                   = 0000001C                  IPL$_QUEUEAST                 = 00000006
ADP$L_CSR                       = 00000000                  IPL$_SYNCH                    = 00000008
ADP$L_INTD                      = 00000038                  MA$INITIAL                      0000018A RG     02
ADP$L_LINK                      = 00000004                  MA$INT                          000002CE R      02
ADP$L_PRQQBL                    = 00000018                  MA$RAVAIL                       00000293 RG     02
ADP$L_PRQQFL                    = 00000014                  MA$REQUEST                      000001F8 RG     02
ADP$L_SHB                       = 00000030                  MBA$INITIAL                     000000E4 RG     02
ADP$L_VECTOR                    = 00000010                  MBA$INT                         0000006C RG     02
ADP$W_ADPTYPE                   = 0000000E                  MBA$L_AS                      = 00000410
ADP$W_SIZE                      = 00000008                  MBA$L_CR                      = 00000004
ADP$W_TR                        = 0000000C                  MBA$L_CSR                     = 00000000
ADP$W_UMR_DIS                   = 00000256                  MBA$L_SR                      = 00000008
ADPLINK                           ********    X   02        MBA$M_CR_IE                   = 00000004
AT$_MPM                         = 00000003                  MBA$M_CR_INIT                 = 00000001
BLOCK_AVAIL                       00000322 R      02        MBA$M_CSR_PD                  = 00800000
BUG$_BADQHDR                      ********    X   02        MMG$GL_SBICONF                  ********    X   02
BUG$_UNKNPRQ                      ********    X   02        MPM$C_PORTS                   = 00000004
C780_LIKE                       = 00000001                  MPM$L_CR                      = 00000004
CEB$L_EFC                       = 00000010                  MPM$L_CSR                     = 00000000
CEB$L_VASLAVE1                  = 00000038                  MPM$L_CSR1                    = 00000018
CEB$L_WQFL                      = 00000014                  MPM$L_ERR                     = 00000010
CEB$W_SIZE                      = 00000008                  MPM$L_IIE                     = 00000024
CI$INITIAL                        0000001F RG     02        MPM$L_IIR                     = 00000020
CI$INT                            00000000 RG     02        MPM$L_INV                     = 0000000C
CI$SHUTDOWN                       0000001F RG     02        MPM$L_MR                      = 0000001C
CPU_TYPE                        = 00000001                  MPM$L_SR                      = 00000008
DCR_K_CLRPWRDN                  = 00000200                  MPM$M_CR_EIE                  = 00000002
DCR_K_CLRPWRUP                  = 00000100                  MPM$M_CR_ERRS                 = FF000000
DCR_K_RESET                     = 00004000                  MPM$M_CR_MIE                  = 00000001
DDT$L_UNSOLINT                  = 00000004                  MPM$M_CSR1_MIA                = 00000400
DEALLOC_BLOCK                     000003EE R      02        MPM$M_CSR_PU                  = 00400000
DEQUEUE_BLOCK                     000002FA R      02        MPM$M_ERR_ELR                 = 10000000
DR$INITIAL                        0000003F RG     02        MPM$M_ERR_ICRD                = 40000000
DR$INT                            00000024 RG     02        MPM$M_ERR_IMP                 = 80000000
DR$SHUTDOWN                       0000003F RG     02        MPM$M_INV_STADR               = 7FF00000
DR_DCR                            00000000                  MPM$M_SR_ACA                  = 80000000
DYN$C_ADP                       = 00000001                  MPM$M_SR_AGP                  = 10000000
EXE$ALONONPAGED                   ********    X   02        MPM$M_SR_EIE                  = 00000002
EXE$CHKWAIT2                      ********    X   02        MPM$M_SR_IDL                  = 00004000
EXE$FORK                          ********    X   02        MPM$M_SR_IT                   = 00008000
EXE$GL_DEFFLAGS                   ********    X   02        MPM$M_SR_MXF                  = 40000000
EXE$GL_LOCKRTRY                   ********    X   02        MPM$M_SR_SS                   = 00002000
EXE$GL_SCB                        ********    X   02        MPM$S_CSR_PORT                = 00000002
EXE$INT58                         ********    X   02        MPM$V_CSR_PORT                = 00000000
EXE$RH780_INT                     ********    X   02        MPM$V_IIE_CTL                 = 00000010
EXE$V_CRDENABL                    ********    X   02        MPM$V_IIR_CTL                 = 00000010
FKB$B_FIPL                      = 0000000B                  MPM$V_INV_ID                  = 00000000
FKB$L_FPC                       = 0000000C                  MPMINTD                         00000182 R      02
FKB$L_FR3                       = 00000010                  NOBLOCKS                        00000283 R      02
IDB$L_ADP                       = 00000014                  NUMMPMVEC                     = 00000010
IDB$L_CSR                       = 00000000                  PA_CNF                          00000000
IDB$L_OWNER                     = 00000004                  PA_CNF_M_PDN                  = 00800000
IDB$L_UCBLST                    = 00000018                  PA_CNF_M_PUP                  = 00400000
INI$MPMADP                        000000ED RG     02        PA_PMC                          00000004
INTERRUPT_PORTS                   0000040D R      02        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$W_TO_PORT                 = 00000018
REQUEST_DISP                    000003C6 R       02
REQ_INTERRUPT                   0000026A R       02
RESAVL                          0000047C R       02
RESOURCE_AVAIL                  00000378 R       02
RETURN_BLOCK                    00000232 R       02
RSN$_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
------------------                         --------------
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                   30
_$255$DUA28:[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.

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

0391 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

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