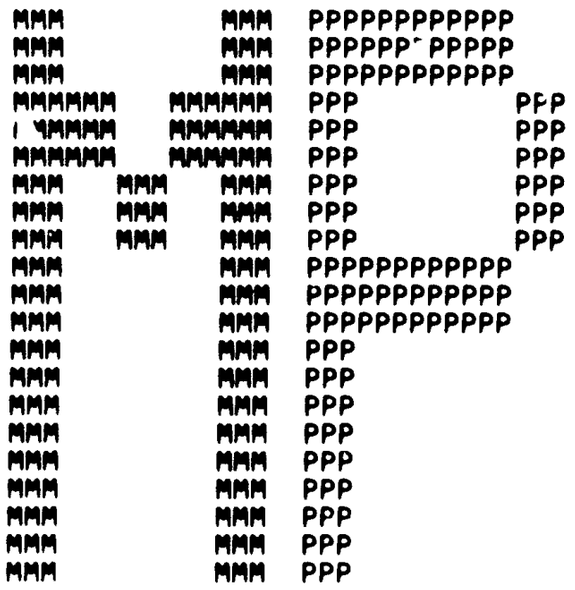
MMM MMM PPPPPPPPPPPP
MMM MMM PPPPPP PPPPP
MMM MMM PPPPPPPPPPPP
MMMMMM MMMMMM PPP PPP
MMMMMM MMMMMM PPP PPP
MMMMMM MMMMMM PPP PPP
MMM MMM MMM PPP PPP
MMM MMM MMM PPP PPP
MMM MMM MMM PPP PPP
MMM MMM PPPPPPPPPPPP
MMM MMM PPPPPPPPPPPP
MMM MMM PPPPPPPPPPPP
MMM MMM PPP
MMM MMM PPP
MMM MMM PPP
MMM MMM PPP
MMM MMM PPP
MMM MMM PPP
MMM MMM PPP
MMM MMM PPP
MMM MMM PPP

```
MM      MM  PPPPPPPP    MM      MM    CCCCCCCC  HH      HH  EEEEEEEEEE    CCCCCCCC  KK      KK
MM      MM  PPPPPPPP    MM      MM    CCCCCCCC  HH      HH  EEEEEEEEEE    CCCCCCCC  KK      KK
MMMM  MMMM  PP      PP  MMMM  MMMM  CC          HH      HH  EE          CC          KK      KK
MMMM  MMMM  PP      PP  MMMM  MMMM  CC          HH      HH  EE          CC          KK      KK
MM  MM  MM  PP      PP  MM  MM  MM  CC          HH      HH  EE          CC          KK    KK
MM  MM  MM  PP      PP  MM  MM  MM  CC          HH      HH  EE          CC          KK    KK
MM      MM  PPPPPPPP    MM      MM  CC          HHHHHHHHHH  EEEEEEEE    CC          KKKKK
MM      MM  PPPPPPPP    MM      MM  CC          HHHHHHHHHH  EEEEEEEE    CC          KKKKK
MM      MM  PP          MM      MM  CC          HH      HH  EE          CC          KK    KK
MM      MM  PP          MM      MM  CC          HH      HH  EE          CC          KK    KK
MM      MM  PP          MM      MM  CC          HH      HH  EE          CC          KK      KK      ....
MM      MM  PP          MM      MM  CC          HH      HH  EE          CC          KK      KK      ....
MM      MM  PP          MM      MM    CCCCCCCC  HH      HH  EEEEEEEEEE    CCCCCCCC  KK      KK      ....
MM      MM  PP          MM      MM    CCCCCCCC  HH      HH  EEEEEEEEEE    CCCCCCCC  KK      KK      ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
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 ;
                     0000     2 ; Version:      'V04-000'
                     0000     3 ;
                     0000     4
                     0000     5         .MCALL  MFPR
00000001             0000     1 MPSWITCH = 1
                     0000     1         .NLIST CND
                     0000     5         .TITLE  MPMCHECK - MACHINE CHECK EXCEPTION HANDLER FOR MP SECONDARY
                     0000     7
                     0000     8         .IDENT  'V04-000'
                     0000     9 ;
                     0000    10 ;*****************************************************************************
                     0000    11 ;*                                                                           *
                     0000    12 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
                     0000    13 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
                     0000    14 ;*  ALL RIGHTS RESERVED.                                                     *
                     0000    15 ;*                                                                           *
                     0000    16 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
                     0000    17 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
                     0000    18 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
                     0000    19 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
                     0000    20 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
                     0000    21 ;*  TRANSFERRED.                                                             *
                     0000    22 ;*                                                                           *
                     0000    23 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
                     0000    24 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
                     0000    25 ;*  CORPORATION.                                                             *
                     0000    26 ;*                                                                           *
                     0000    27 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
                     0000    28 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
                     0000    29 ;*                                                                           *
                     0000    30 ;*                                                                           *
                     0000    31 ;*****************************************************************************
                     0000    32
                     0000    33 ;++
                     0000    34 ; FACILITY:     EXECUTIVE, ERROR HANDLING
                     0000    35 ;
                     0000    36 ; ABSTRACT: IN A NUTSHELL, LOG IT AND TRY TO RECOVER.
                     0000    37 ;
                     0000    38 ; ENVIRONMENT:  RUNS ON INTER.U.T STACK AT IPL 31 UNTIL ERROR TYPE IS KNOWN
                     0000    39 ;               AND (IF POSSIBLE) CORRECTED, THEN RUNS AT SYNCH LEVEL
                     0000    40 ;               TO DO THE ERROR LOGGING.
                     0000    42 ;
                     0000    43 ;       EXECUTES ON SECONDARY PROCESSOR.
                     0000    44 ;
                     0000    45 ;       THE CONFIGURATION ARRAY (EXE$GL_CONFREG) IS VALID ONLY FOR THE
                     0000    46 ;       PRIMARY PROCESSOR, NOT FOR THE SECONDARY PROCESSOR.  THE ONLY
                     0000    47 ;       INFORMATION THAT IS VALID FOR BOTH IS THAT FOR THE MA780S.  THE
                     0000    48 ;       MA780 MEMORIES MUST BE ON THE SAME TR'S AND AT THE SAME ADDRESSES
                     0000    49 ;       ON BUTH PROCESSORS, IN ORDER FOR BOTH PROCESSORS TO SHARE ONE
                     0000    50 ;       SYSTEM PAGE TABLE.
                     0000    51 ;
                     0000    52 ;       UNTIL A CONFIGURATION ARRAY IS CREATED FOR THE SECONDARY PROCESSOR,
                     0000    53 ;       IT WILL BE LIMITED TO LOGGING ONLY MA780 MEMORY REGISTERS.  CODE FOR
                     0000    54 ;       SUPPORT OF OTHER MEMORY CONTROLLERS IS REMOVED (TO CONSERVE SPACE)
                     0000    55 ;       VIA ASSEMBLY SWITCHES.
                     0000    57
```

```
0000     58 ;--
0000     59         .SBTTL  HISTORY                 ; DETAILED
0000     60 ;
0000     61 ; AUTHOR: RICHARD LARY  , CREATION DATE: 6-NOV-77
0000     62 ;
0000     63 ; MODIFIED BY:
0000     64 ;
0000     65 ;       V03-013 WMC0002         Wayne Cardoza           25-Jul-1984
0000     66 ;               Add H memory to the tables.
0000     67 ;
0000     68 ;       V03-012 WMC0001         Wayne Cardoza           14-Jun-1984
0000     69 ;               Preserve cache state when handling machine check.
0000     70 ;               Properly clear group 1 cache parity errors.
0000     71 ;
0000     72 ;       V03-011 NPK3049         N. Kronenberg           10-Apr-1984
0000     73 ;               Tighten up check for BRRVR reference from unibus
0000     74 ;               interrupt service routine in CPTIMOUT.  Test for
0000     75 ;               PC as well as VA.
0000     76 ;
0000     77 ;       V03-010 RLRSBICONF      Robert L. Rappaport     22-Mar-1984
0000     78 ;               Test MMG$GL_SBICONF array elements for valid system
0000     79 ;               virtual address (high bit set) before using.  Also
0000     80 ;               correct error introduced by CONFREGL change.
0000     81 ;
0000     82 ;       V03-009 KPL0100         Peter Lieberwirth       10-Feb-1984
0000     83 ;               Change to use CONFREGL.
0000     84 ;
0000     85 ;       V03-008 KDM0053         Kathleen D. Morse       11-Jul-1983
0000     86 ;               Replace cpu-specific IPR references with the new
0000     87 ;               cpu-specific $PR780DEF symbols.
0000     88 ;
0000     89 ;       V03-007 TCM0011         Trudy C. Matthews       24-Jan-1983
0000     90 ;               Correct bug in MA780 logging routine that checked for
0000     91 ;               Multiple Interlock Accepted error bit in the wrong
0000     92 ;               MA780 register.
0000     93 ;
0000     94 ;       V03-006 KDM0040         Kathleen D. Morse       13-Jan-1983
0000     95 ;               Change PRMSW to MPSWITCH and integrate into multi-processing
0000     96 ;               code replacing [MP.SRC]MPMCHECK.MAR.  Fix bug that referenced
0000     97 ;               devices attached to primary (via CONFREG array) from the
0000     98 ;               secondary processor's machine-check code.
0000     99 ;
0000    100 ;       V03-005 RNG0001         Rod N. Gamache          15-Oct-1982
0000    101 ;               Fixed code that enabled the MS780-E memory CRD (corrected
0000    102 ;               read data) interrupts. Fixed code that re-enabled the MS780-C
0000    103 ;               CRD interrupts.
0000    104 ;
0000    105 ;
```

```
0000   107          .SBTTL  MEMORY_ROUTINES Macro
0000   108 ;++
0000   109 ; Macro MEMORY_ROUTINES
0000   110 ;       Build action routine vectors for different memory types.
0000   111 ;
0000   112 ; Inputs:
0000   113 ;       MEMTYPES        - A list of 'NDT$_" type codes for this controller.
0000   114 ;       LOGERR_RTN      - Action routine that determines if an error was
0000   115 ;                         reported for this controller; if so, it logs it.
0000   116 ;       LOGALL_RTN      - Action routine to unconditionally log this
0000   117 ;                         controller's registers.
0000   118 ;       ENAB_RTN        - Action routine to enable CRD interrupts for this
0000   119 ;                         memory controller.
0000   120 ;
0000   121 ; Outputs:
0000   122 ;       Additions to LOGERR_ROUTINES, LOGALL_ROUTINES, and ENAB_ROUTINES arrays.
0000   123 ;
0000   124 ; Note: Each invocation of this macro corresponds to one "general" memory type.
0000   125 ;       Each element in MEMTYPES list corresponds to one "specific" type.
0000   126 ;--
0000   127          .MACRO  MEMORY_ROUTINES         MEMTYPES,LOGERR_RTN,LOGALL_RTN,ENAB_RTN
0000   128          .SAVE
0000   129 ;
0000   130 ; Create arrays to map a se. of specific type codes to one general memory type.
0000   131 ; Note: Psects MCHK$DATA0 and MCHK$DATA1 must be contiguous.
0000   132 ;
0000   133          .IRP    MEMTYP,MEMTYPES         ; Repeat for each memory type...
0000   134
0000   135          .IF     NDF,MPSWITCH            ;***** ONLY PRIMARY PROCESSOR...
0000   136          .PSECT  MCHK$DATA0,LONG,WRT     ; Add specific-type entry to MEMTYP
0000   137          .IFF                            ;***** ONLY SECONDARY PROCESSOR...
0000   138          .PSECT  Y$MPDATA0,LONG,WRT      ; Add specific-type entry to MEMTYP
0000   139          .ENDC                           ;***** PRIMARY and SECONDARY PROCESSORS
0000   140          .BYTE   MEMTYP                  ; array.
0000   141
0000   142          .IF     NDF,MPSWITCH            ;***** ONLY PRIMARY PROCESSOR...
0000   143          .PSECT  MCHK$DATA1              ; Add general-type entry to MEMTYP
0000   144          .IFF                            ;***** ONLY SECONDARY PROCESSOR...
0000   145          .PSECT  Y$MPDATA1               ; Add general-type entry to MEMTYP
0000   146          .ENDC                           ;***** PRIMARY and SECONDARY PROCESSORS
0000   147          .BYTE   GENERAL_MEMTYP
0000   148
0000   149 MEMTYPCNT = MEMTYPCNT + 1
0000   150          .ENDR
0000   151 GENERAL_MEMTYP = GENERAL_MEMTYP + 1
0000   152 ;
0000   153 ; Now create action routine vectors.
0000   154 ;
0000   155          .IF     NDF,MPSWITCH            ;***** ONLY PRIMARY PROCESSOR...
0000   156          .PSECT  MCHK$DATA2,LONG,WRT     ; LOGERR_ROUTINES array:
0000   157          .IFF                            ;***** ONLY SECONDARY PROCESSOR...
0000   158          .PSECT  Y$MPDATA2,LONG,WRT      ; LOGERR_ROUTINES array:
0000   159          .ENDC                           ;***** PRIMARY and SECONDARY PROCESSORS
0000   160          .LONG   <LOGERR_RTN-.>          ; Add self-relative offset to routine.
0000   161
0000   162          .IF     NDF,MPSWITCH            ;***** ONLY PRIMARY PROCESSOR...
0000   163          .PSECT  MCHK$DATA3,LONG,WRT     ; LOGALL_ROUTINES array:
```

```
0000    164          .IFF                                ;***** ONLY SECONDARY PROCESSOR...
0000    165          .PSECT  Y$MPDATA3,LONG,WRT          ; LOGALL_ROUTINES array:
0000    166          .ENDC                               ;***** PRIMARY and SECONDARY PROCESSORS
0000    167          .LONG   <LOGALL_RTN-.>              ; Add self-relative offset to routine.
0000    168
0000    169          .IF     NDF,MPSWITCH                ;***** ONLY PRIMARY PROCESSOR...
0000    170          .PSECT  MCHK$DATA4,LONG,WRT         ; ENA3_ROUTINES array:
0000    171          .IFF                                ;***** ONLY SECONDARY PROCESSOR...
0000    172          .PSECT  Y$MPDATA4,LONG,WRT          ; ENAB_ROUTINES array:
0000    173          .ENDC                               ;***** PRIMARY and SECONDARY PROCESSORS
0000    174          .LONG   <ENAB_RTN-.>                ; Add self-relative offset to routine.
0000    175
0000    176          .RESTORE
0000    177          .ENDM   MEMORY_ROUTINES
```

```
                        .SBTTL  SYMBOL DEFINITIONS

0000000A  CH_THRESHOLD  =       10.             ;3 ERRORS IN 100 MS TO DISABLE CACHE
00010000  CH_MISSG0     =       ^X10000         ;"FORCE MISS GROUP 0" BIT
00008000  CH_MISSG1     =       ^X8000          ;"FORCE MISS GROUP 1" BIT
00004000  CH_REPLG0     =       ^X4000          ;"FORCE REPLACE GROUP 0" BIT
00002000  CH_REPLG1     =       ^X2000          ;"FORCE REPLACE GROUP 1" BIT
0000000D  CH$V_REPLG1   =       13
00000004  CH$S_CONTROL  =       4               ;SIZE OF CACHE CONTROL FIELD
0021C000  CH_REPAIR     =       ^X21C000        ;BITS TO SET IN SBIMT ON CACHE ERRORS
0021A000  CH_REPAIR_1   =       ^X21A000        ;BITS CLEAR GROUP 1 CACHE ERRORS
00000003  CH$V_GOERRS   =       3               ;START OF GROUP 0 ERRORS IN PARITY REG
00000007  CH$S_GOERRS   =       7               ;LENGTH OF GROUP 0 ERROR BITS
00000001  CHLOG_DISAB0  =       1               ;LOG BIT SAYING WE DISABLED GROUP 0
00000002  CHLOG_DISAB1  =       2               ;LOG BIT SAYING WE DISABLED GROUP 1

00000019  SBIFS$V_NEF   =       25              ;NESTED ERROR FLAG IN SBI FAULT/STATUS

                        ;THE FOLLOWING 5 DEFINITIONS ARE IN THE SBI ERROR REGISTER
00000040  SBIER$M_IBTO  =       ^X40            ;IB TIMEOUT LATCH
00000080  SBIER$M_IBRDS =       ^X80            ;IB RDS LATCH
00001000  SBIER$M_CPTO  =       ^X1000          ;CP TIMEOUT LATCH
00002000  SBIER$M_RDS   =       ^X2000          ;RDS LATCH
00004000  SBIER$M_CRD   =       ^X4000          ;CRD LATCH

                        ;       MACHINE CHECK HARDWARE LOG OFFSETS

00000000  MCL_COUNT     =       0               ;BYTE LENGTH OF AREA (28 HEX)
00000004  MCL_SUMMARY   =       4               ;SUMMARY WORD - BYTE 0=CODE, BYTE 1=
                                                ;TIMEOUT PENDING FLAG
00000008  MCL_CES       =       8               ;CPU ERROR STATUS
0000000C  MCL_UPC       =       12.             ;MICRO-PC AT FAULT TIME
00000010  MCL_VA        =       16.             ;VIRTUAL ADDR AT FAULT TIME
00000014  MCL_D         =       20.             ;CPU D REGISTER AT FAULT TIME
00000018  MCL_TBER0     =       24.             ;TRANSLATION BUFFER STATUS REG 0
0000001C  MCL_TBER1     =       28.             ;TBUF STATUS REG 1
00000020  MCL_TIMOADDR  =       32.             ;PHYSICAL ADDRESS CAUSING SBI TIMEOUT
00000024  MCL_PARITY    =       36.             ;CACHE STATUS REGISTER
00000028  MCL_SBIERR    =       40.             ;SBI ERROR REGISTER
0000002C  MCL_PC        =       44.             ;PC OF INSTRUCTION WHICH CAUSED CHECK
00000030  MCL_PSL       =       48.             ;PSL OF MACHINE AT FAULT TIME
```

```
                        .SBTTL MEMORY CONTROLLOR AND ERROR DEFINITIONS

                ;
                ; Common error bit definitions.
                ;
0000001C        MRC$V_ELSRF     =       28              ;ERROR LOG SERVICE REQUEST
10000000        MRC$M_ELSRF     =       ^X10000000      ;WRITE 1 TO CLEAR FLAG
0000001D        MRC$V_HERIMF    =       29              ;HIGH ERROR RATE IN MEMORY
20000000        MRC$M_HERIMF    =       ^X20000000      ;WRITE 1 TO CLEAR FLAG
0000001E        MRC$V_INHBCRD   =       30              ;1 DISABLES CRD INTERRUPT
40000000        MRC$M_INHBCRD   =       ^X40000000      ;0 CRD INTERRUPT ENABLE, 1 CRD DISABLE
                ;
                ; MA780-specific error bit definitions (in Array Error Register).
                ;
0000001F        MRC$V_INVMAPPTY =       31              ;INVALID MAP PARITY ERROR
80000000        MRC$M_INVMAPPTY =       ^X80000000      ;WRITE 1 TO CLEAR THE FLAG
                ;
                ; MS780E-specific error bit definitions.
                ;
00000014        MRC$V_SUMMARY   =       20              ;ERROR SUMMARY BIT
00100000        MRC$M_SUMMARY   =       ^X00100000      ;OR OF ALL ERROR BITS -- READ ONLY
00000013        MRC$V_CTL1PTY   =       19              ;PARITY ERROR ON READ DATA FROM
00080000        MRC$M_CTL1PTY   =       ^X00080000      ;CONTROLLER 1 TO SBI INTERFACE.
00000012        MRC$V_CTL0PTY   =       18              ;PARITY ERROR ON READ DATA FROM
00040000        MRC$M_CTL0PTY   =       ^X00040000      ;CONTROLLER 0 TO SBI INTFRFACE.
                                                        ;FOLLOWING BITS ARE IN REGISTERS C & D
00000007        MRC$V_MSEQPTY   =       7               ;MICROSEQUENCER PARITY ERROR
00000080        MRC$M_MSEQPTY   =       ^X00000080      ;
00000008        MRC$V_IFPTY     =       8               ;PARITY ERROR ON WRITE DATA FROM
00000100        MRC$M_IFPTY     =       ^X00000100      ;SBI INTERFACE TO CONTROLLER.
00000009        MRC$V_CRDERR    =       9               ;CORRECTED READ DATA ERROR
00000200        MRC$M_CRDERR    =       ^X00000200      ;

00000384        REENABTIME      = 60*15                 ;REENABLE INTERRUPT ERROR LOGGING
                                                        ;EVERY 15 MINUTES
0000003C        SOMETIME        = 60                    ;SCAN FOR NON-INTERRUPT ERRORS
                                                        ;EVERY 60 SECONDS
00000003        CRDINTMAX       = 3                     ;MAXIMUM NUMBER OF INTERRUPTS A CONT
                                                        ;IS ALLOWED WITHIN REENABTIME
00000006        CRDWATCHMAX     = 6                     ;MAXIMUM NUMBER OF ERRORS TO BE LOGGED
                                                        ;WITHIN REENABTIME

                ;       INCLUDED SYMBOL DEFINITIONS

                        $ADPDEF                         ;DEFINE ADAPTER CONTROL BLOCK SYMBOLS
                        $EMBDEF <MC,SB,SE>              ;DEFINE EMB OFFSETS
                        $IPLDEF                         ;PROCESSOR INTERRUPT LEVELS
                        $MCHKDEF                        ;DEFINE RECOVERY BLOCK MASK BITS
                        $NDTDEF                         ;DEFINE NEXUS DEVICE TYPES
                        $PCBDEF                         ;PROCESS CTL BLOCK
                        $PFNDEF                         ;PFN DATA BASE
                        $PRDEF                          ;DEFINE PROCESSOR REGISTER NUMBERS
                        $PR780DEF                       ;DEFINE 780-SPECIFIC PROCESSOR REGISTERS
                        $PSLDEF                         ;DEFINE PSL
                        $PTEDEF                         ;PTE SYMBOLS
                        $SSDEF                          ;DEFINE SYSTEM STATUS VALUES
                        $VADEF                          ;DEF IN PFN PITS
```

```
                     0000   282          .SBTTL  MEMORY ACTION ROUTINE ARRAYS
                     0000   283
                     0000   287          .PSECT  Y$MPDATA0,LONG,WRT
                     0000   289 MEMTYP:                                   ; Define base of array of memory type
                     0000   290                                           ; codes.
                     0000   294          .PSECT  Y$MPDATA2,LONG,WRT
                     0000   296 LOGERR_ROUTINES:                          ; Define base of array of routines to
                     0000   297                                           ; log memories with errors.
                     0000   301          .PSECT  Y$MPDATA3,LONG,WRT
                     0000   303 LOGALL_ROUTINES:                          ; Define base of array of routines to
                     0000   304                                           ; unconditionally log memories.
                     0000   308          .PSECT  Y$MPDATA4,LONG,WRT
                     0000   310 ENAB_ROUTINES:                           ; Define base of array of routines to
                     0000   311                                           ; enable CRD interrupts in memories.
                     0000   312 ;
                     0000   313 ; The following macro invocations add elements to the above arrays for each
                     0000   314 ; memory type.
                     0000   315 ;
                     0000   316
                     0000   317 MEMTYPCNT = 0
                     0000   318 GENERAL_MEMTYP = 0
                     0000   319
                     0000   320          MEMORY_ROUTINES -                ; MS780C memory controller.
                     0000   321                  MEMTYPES=<NDT$_MEM4NI,NDT$_MEM4I,NDT$_MEM16NI,NDT$_MEM16I>, -
                     0000   322                  LOGERR_RTN = LOG_MS780C, -
                     0000   323                  LOGALL_RTN = LOGC, -
                     0000   324                  ENAB_RTN = ENAB_MS780C
                     0000   325
                     0000   326          MEMORY_ROUTINES -                ; MA780 memory controller.
                     0000   327                  MEMTYPES=<NDT$_MPM0,NDT$_MPM1,NDT$_MPM2,NDT$_MPM3>, -
                     0000   328                  LOGERR_RTN = LOG_MA780, -
                     0000   329                  LOGALL_RTN = LOGMA, -
                     0000   330                  ENAB_RTN = ENAB_MA780
                     0000   331
                     0000   332          MEMORY_ROUTINES -                ; MS780E memory controller.
                     0000   333                  MEMTYPES=<NDT$_MEM64NIL,NDT$_MEM64EIL,NDT$_MEM64NIU, -
                     0000   334                            NDT$_MEM64EIU,NDT$_MEM64I, -
                     0000   335                            NDT$_MEM256NIL,NDT$_MEM256EIL,NDT$_MEM256NIU, -
                     0000   336                            NDT$_MEM256EIU,NDT$_MEM256I>, -
                     0000   337                  LOGERR_RTN = LOG_MS780E, -
                     0000   338                  LOGALL_RTN = LOGE, -
                     0000   339                  ENAB_RTN = ENAB_MS780E
```

```
                     0000   341          .SBTTL  LOCAL DATA STORAGE
                     0000   342
                     0000   343 ;
                     0000   344 ; Macro that will define a global name of the form MPS$ if
                     0000   345 ; MPSWITCH is defined, else EXE$.  It will also define a local name
                     0000   346 ; to be used within this module.
                     0000   347 ;
                     0000   348          .MACRO  GBLDEF  NAME
                     0000   349          .IF     DF,MPSWITCH                     ; For secondary processor only code...
                     0000   350 MPS$'NAME'::
                     0000   351          .IFF                                    ; For MCHECK780...
                     0000   352 EXE$'NAME'::
                     0000   353          .ENDC
                     0000   354 'NAME':                                         ; For local use...
                     0000   355          .ENDM   GBLDEF
                     0000   356
                00000000    360          .PSECT  $$$$MPDATA,QUAD,WRT
                     0000   362 ; The following symbol is defined for a transfer vectror in SYSLOAVEC
                     0000   363 ; This location is NEVER JUMPED TO.  It is defined so these counters
                     0000   364 ; Can be located using a global symbol in the system map.
                     0000   365
                     0000   366 GBLDEF  MCHK_ERRCNT                             ;GLOBAL SYMBOL FOR SYSLOAVEC POINTER
                     0000   367 GBLDEF  GL_CSBITA                               ;USED TO HOLD COMPLEMENT OF SBITA
00000000             0000   368                  .LONG   0
                     0004   369 GBLDEF  GL_CH1OLD                               ;TIME OF LAST CACHE ERROR
00000000             0004   370                  .LONG   0
                     0008   371 GBLDEF  GL_CH2OLD                               ;TIME OF NEXT-TO-LAST CACHE ERROR
00000000             0008   372                  .LONG   0
                     000C   373 GBLDEF  GL_CPTIMOUT                            ;TIME OF LAST CP TIMEOUT/SBI ERROR
00000000             000C   374                  .LONG   0
                     0010   375 GBLDEF  AB_MEMERR                              ;ERROR COUNTERS FOR 16 ADAPTERS
00000020             0010   376          .BLKB   16
                     0020   377 GBLDEF  GW_REENAB                              ;REENABLE TIMER
        0000         0020   378                  .WORD   0
                     0022   379 GBLDEF  GW_WATCH                               ;SCAN MEMORY CONTROLLER TIMER
        0000         0022   380                  .WORD   0
                     0024   381 GBLDEF  GL_CRDCNT                              ;COUNT OF CORRECTED MEMORY ERRORS
00000000             0024   382                  .LONG   0
                     0028   383 GBLDEF  GL_CHSTATE                             ;CURRENT STATE OF CACHE
00200000             0028   384                  .LONG   ^X200000
                     002C   385 GBLDEF  GL_BADTIMOUT                           ;TIME SINCE LAST BAD MCHK CODE
00000000             002C   386                  .LONG   0
                     0030   387
```

```
                                0030   389          .SBTTL  MACHINE CHECK ENTRY POINT
                           00000000   393          .PSECT  YY$MPCODE,QUAD,RD,WRT
                                0000   395
                                0000   396 ;        MACHINE CHECK ENTRY POINT - SCB VECTOR POINTS HERE.
                                0000   397 ;        IPL ^X1F = 31
                                0000   398
                                0000   399          .ALIGN  LONG                        ;A VECTOR MUST HAVE LONGWORD ALIGNMENT
                                0000   400 GBLDEF   MCHK                               ;EITHER EXE$MCHK:: OR MPS$MCHK::
                                0000   401
                          01    DD      0000   402          PUSHL   #MCHK$M_LOG                 ;MASK WORD FOR PRTCTEST
                       30 AE    DF      0002   403          PUSHAL  MCL_PC+4(SP)                ;PC,PSL POINTER FOR PRTCTEST
                    103F 8F     BB      0005   404          PUSHR   #^M<R0,R1,R2,R3,R4,R5,AP>
                5C  5E   24     C1      0009   405          ADDL3   #<9*4>,SP,AP                ;POINT AP TO LOG FRAME ON STACK
                                0000D  406                                                  ;ALL INTERRUPTS ARE LOCKED OUT!
                                000D   407
                                000D   408          MFPR    #PR780$_SBIMT,R0            ;GET CURRENT CACHE STATE
          50    50   F3 8F     78      0010   409          ASHL    #-CH$V_REPLG1,R0,R0         ;POSITION THE CACHE CONTROL BITS
                   0D   50     F0      0015   410          INSV    R0,#CH$V_REPLG1,-           ;SAVE THE CURRENT CACHE CONTROL BITS
             0028'CF    04             0018   411                  #CH$S_CONTROL,W^GL_CHSTATE
          33  0021C000 8F     DA      001C   412          MTPR    #CH_REPAIR,#PR780$_SBIMT    ;FORCE MISSES AND GROUP 0 REPLACE,
                                0023   413                                                  ; BUT ALLOW SBI TO INVALIDATE CACHE
       7E  04 AC   F0 8F     8B      0023   414          BICB3   #^XF0,MCL_SUMMARY(AP),-(SP) ;GET LOW 4 BITS OF TYPE CODE
                                0029   415          CASE    (SP)+,<-                    ;BREAKOUT TYPE CODE
                                0029   416                  CPTIMEOUT,-                 ;CPU TIMEOUT/SBI ERROR CONFIRMATION
                                0029   417                  CSPARITY,-                  ;CONTROL STORE PARITY ERROR
                                0029   418                  TBUFPARITY,-                ;TRANSLATION BUFFER PARITY ERROR
                                0029   419                  CACHEPARITY,-               ;CACHE PARITY ERROR
                                0029   420                  BADTYPE,-                   ;THIS CODE DOESN'T EXIST
                                0029   421                  READSUBST,-                 ;READ DATA SUBSTITUTE (MEM READ ERROR)
                                0029   422                  IBROMCHECK,-                ;"CAN'T GET HERE" ERROR FROM INST ROMS
                                0029   423                  BADTYPE,-
                                0029   424                  BADTYPE,-
                                0029   425                  BADTYPE,-
                                0029   426                  TBUFPARITY,-                ;IB-DETECTED TBUF ERROR
                                0029   427                  BADTYPE,-
                                0029   428                  READSUBST,-                 ;IB-DETECTED MEMORY ERROR
                                0029   429                  CPTIMEOUT,-                 ;IB-DETECTED TIMEOUT OR SBI ERROR CONF
                                0029   430                  BADTYPE,-
                                0029   431                  CACHEPARITY>, TYPE=B        ;IB-DETECTED CACHE PROBLEM
                                004D   432
                                004D   433 BADTYPE:
    33  0028'CF     DA          004D   434          MTPR    W^GL_CHSTATE,#PR780$_SBIMT  ;RE-ENABLE THE CACHE
      FC AC   02    C8          0052   435          BISL    #MCHK$M_MCK,-4(AP)          ;MASK FOR PRTCTEST
          002C'CF   DD          0056   436          PUSHL   W^GL_BADTIMOUT              ;TIME OF LAST BAD TYPE FAULT
                                005A   437          MFPR    #PR780$_TODR,W^GL_BADTIMOUT ;TIME OF CURRENT FAULT
002C'CF     8E      D1          009F   438          CMPL    (SP)+,W^GL_BADTIMOUT        ;COMING TO FAST?
              1B    12          00A4   439          BNEQ    DAMPUTATE                   ;YES, ABORT
                                00A6   440 100$:
          103F 8F   BA          00A6   441          POPR    #^M<R0,R1,R2,R3,R4,R5,AP>
                                00AA   446          SECBUG_CHECK    MPBADMCK,FATAL  ;BAD MACHINE CHECK CODE
```

```
                                00AF   449          .SBTTL  TRANSLATION BUFFER PARITY ERRORS
                                00AF   450
                                00AF   451 TBUFPARITY:
      33   0028'CF    DA      00AF   452          MTPR    W^GL_CHSTATE,#PR780$_SBIMT ;RE-ENABLE CACHE
           39    00    DA      00B4   453          MTPR    #0,#PR$_TBIA               ;CLEAR ENTIRE TBUF
        FC AC    02    C8      00B7   454          BISL    #MCHK$M_MCK,-4(AP)         ;SET MACHINE CHECK CODE FOR PRTCTEST
                                00BB   455
                                00BB   456
                                00BB   457 TRYRESUME:
   04 AC    01F0 8F    B3      00BB   458          BITW    #^X1F0,MCL_SUMMARY(AP)     ;IS ERROR ABORT OR TIMEOUT PENDING
                                00C1   459 DAMPUTATE:
                 2F    12      00C1   460          BNEQ    AMPUTATE                   ;BRANCH IF YES, NO HOPE OF RESUMING
         04 AC    08    93      00C3   461          BITB    #8,MCL_SUMMARY(AP)         ;SEE IF ERROR WAS IB ERROR
                  0A    12      00C7   462          BNEQ    10$                        ;IF SO, WE CAN "DEFINITELY" RESUME
         7E    2C BC    9A      00C9   463          MOVZBL  @MCL_PC(AP),-(SP)          ;GET OPCODE FOR RESTARTABILITY CHECK
   1F 052D'CF    8E    E1      00CD   464          BBC     (SP)+,W^RESUMABLE,AMPUTATE ;BRANCH IF INST NOT RESUMABLE,ABORT
                                00D3   465 10$:     ;THERE IS A LOW PROBABILITY CASE HERE THAT MAY ALLOW THIS CODE TO
                                00D3   466          ;CONTINUE WHEN WE CAN'T - IF A LOCATION IS READ FROM THE IO PAGE AND
                                00D3   467          ;HAS A SIDE AFFECT WHICH MODIFIES THAT LOCATION, THE INSTRUCTION IS
                                00D3   468          ;NOT RETRYABLE, A SOFTWARE SOLUTION IS TO IMPLEMENT A FLAG SET BY ANY
                                00D3   469          ;POTENTIAL REFERENCE TO THE IO PAGE THAT MAY CAUSE A SIDE AFFECT.
                                00D3   470          ;BBS     #IOSAFLAG,FLAG,AMPUTATE ;BRANCH IF INST MAY OF HAD SIDE AFFECT
                                00D3   471 RESUME:
            53    02    B0      00D3   472          MOVW    #EMB$K_MC,R3               ;SET TYPE OF LOG ENTRY
               016E   30      00D6   473          BSBW    LOGGER                     ;WE'RE GOING TO MAKE IT - LOG ERROR
            103F 8F    BA      00D9   474          POPR    #^M<R0,R1,R2,R3,R4,R5,AP>  ;RESTORE REGISTERS
            5E    08   C0      00DD   475          ADDL    #8,SP                      ;REMOVE PRTCTEST STUFF FROM STACK
            5E    8E   C0      00E0   476          ADDL    (SP)+,SP                   ;POP HARDWARE LOG FROM STACK
                       02      00E3   477          REI                                ;AND TRY AGAIN
```

```
                                00E4   479          .SBTTL  ERRORS DETECTED IN INSTRUCTION DECODE ROMS
                                00E4   480          .SBTTL  CONTROL STORE PARITY ERRORS
                                00E4   481
                                00E4   482 IBROMCHECK:
                                00E4   483 CSPARITY:
                 33   0028'CF   DA   00E4   484          MTPR    W^GL_CHSTATE,#PR780$_SBIMT ;CACHE PROBABLY OK - ENABLE IT
               06 AC     2C BC   90   00E9   485          MOVB    @MCL_PC(AP),MCL_SUMMARY+2(AP) ;SAVE OPCODE IN LOG
                  FC AC     02   C8   00EE   486          BISL    #MCHK$M_MCK,-4(AP)      ;SET MASK CODE FOR PRTCTEST
                                00F2   487 AMPUTATE:
                     53     02   B0   00F2   488          MOVW    #EMB$K_MC,R3            ;SET TYPE OF LOG ENTRY
                         014F   30   00F5   489          BSBW    LOGGER                  ;LOG THE ERROR
               09 30 AC     19   E0   00F8   490          BBS     #PSL$V_CURMOD+1,MCL_PSL(AP),REFLECTCHK ;BRANCH IF
                                00FD   491                                              ;FAILURE IN USER OR SUPERVISOR MODE
                                00FD   492
                   103F 8F     BA   00FD   493          POPR    #^M<R0,R1,R2,R3,R4,R5,AP> ;RESTORE REGS
                                0101   498          SECBUG_CHECK    MPMCHECK,FATAL  ;MACHINE CHECK IN KERNEL OR EXEC MODE
                                0106   500
                                0106   502 ;
                                0106   503 ; THIS CODE FOLDS UP A PROCESS AND HANDS IT BACK TO THE PRIMARY,
                                0106   504 ; FORCING IT TO EXECUTE IN THE EXCEPTION HANDLER.  THIS IS DONE BY
                                0106   505 ; REMOVING ALL TRACES OF THE INTERRUPT FROM THE SECONDARY'S INTERRUPT
                                0106   506 ; STACK, AND PLACING THE APPROPRIATE PC AND PSL PAIRS ON THE PROCESS'S
                                0106   507 ; KERNEL STACK.  THEN A NORMAL RESCHEDULE REQUEST IS MADE BY THE SECONDARY.
                                0106   508 ;
                                0106   510 REFLECTCHK:
                                0106   511          MFPR    #PR$_KSP,R0             ;GET THE KERNEL MODE STACK POINTER
                     70   2C AC     7D 0109   512          MOVQ    MCL_PC(AP),-(R0)        ;INTERRUPT PC,PSL TO KERNEL STACK
                                010D   513                                              ;IT IS NOT NECCESARY TO PROBE KERNEL
                                010D   514                                              ;STACK FOR VALIDITY, THE FAILURE WILL
                                010D   515                                              ;BE A KERNEL STACK NOT VALID BUGCHECK
                                010D   516                                              ;FROM WITHIN MACHINE CHECK
                         00     50   DA 010D   517          MTPR    R0,#PR$_KSP             ;REPLACE THE NEW KERNEL STACK POINTER
                         103F   8F   BA 0110   518          POPR    #^M<R0,R1,R2,R3,R4,R5,AP>       ;RESTORE REGISTERS
                         5E     08   C0 0114   519          ADDL    #8,SP                   ;CLEAR PRTCTST STUFF
                         5E     8E   C0 0117   520          ADDL    (SP)+,SP                ;POP HARDWARE LOG FROM STACK
                   6E   00000000'GF   9E 011A   521          MOVAB   G^EXE$MCHECK,(SP)       ;SET UP A PC AND PSL FOR EXCEPTION
            04 AE   04 AE     02   18   EF 0121   522          EXTZV   #PSL$V_CURMOD,#PSL$S_CURMOD,4(SP),4(SP)
                                0128   523                                              ;GET MODE WE WERE EXECUTING IN
                  04 AE   04 AE     16   9C 0128   524          ROTL    #PSL$V_PRVMOD,4(SP),4(SP) ;CREATE A PSL OF CURRENT TO BE
                                012E   525                                              ;KERNEL WITH CORRECT PREVIOUS MODE
                                012E   526                                              ;AS FROM A FAULT, 0'S IN REST OF PSL
                                012E   530          SETIPL  #IPL$_SYNCH             ;LOWER IPL, ENABLING INTER-PROC INT.
                                0131   531
                                0131   535
                           FECC'   31   0131   536          BRW     MPS$MPSCHED2            ;RETURN PROCESS TO PRIMARY
```

```
                                0134    539          .SBTTL  CACHE PARITY ERROR
                                0134    540
                                0134    541 CACHEPARITY:                                ;ENTER WITH CACHE DISABLED REPLACING
                                0134    542                                             ; GROUP 0
                FC AC    02     C8  0134    543          BISL    #MCHK$M_MCK,-4(AP)      ;SET MACHINE CHECK TYPE FOR PRTCTEST
                   10 BC        95  0138    544          TSTB    @MCL_VA(AP)             ;FORCE DATA INTO GROUP 0 OF BAD CACHE
         33   0021A000 8F        DA  013B    545          MTPR    #CH_REPAIR_1,#PR780$_SBIMT ;NOW FORCE GROUP 1 REPLACEMENT AND
                   10 BC        95  0142    546          TSTB    @MCL_VA(AP)             ;FORCE GROUP 1 OF BAD LINE TO GOOD DATA
                                0145    547          MFPR    #PR780$_TODR,-(SP)      ;GET TIME-OF-YEAR IN 10MS TICKS
      7E    6E    0008'CF     C3  0188    548          SUBL3   W^GL_CH2OLD,(SP),-(SP)  ;GET TIME SPAN OF LAST 2 ERRORS-NOW
                      0A    8E  D1  018E    549          CMPL    (SP)+,#CH_THRESHOLD     ;ARE THE ERRORS WIDELY SPACED
                            2E  1A  0191    550          BGTRU   20$                     ;BRANCH IF YES TO FORGIVE THE CACHE
0000007F 8F    24 AC    07    03  ED  0193    551          CMPZV   #CH$V_GOERRS,#CH$S_GOERRS,MCL_PARITY(AP),#^B11111111
                                019D    552                                             ;IS GROUP 0 ALL GOOD?
                            11  12  019D    553          BNEQ    10$                     ;BRANCH IF GROUP 0 WAS BAD
         0029'CF    C0     8F  88  019F    554          BISB    #CH_MISSG1!CH_REPLG0@-8,W^GL_CHSTATE+1 ;DISABLE GROUP 1
            0029'CF    20    8A  01A5    555          BICB    #CH_REPLG1@-8,W^GL_CHSTATE+1 ;CANNOT FORCE REPLACE IN BOTH
              07 AC    02    90  01AA    556          MOVB    #CH_LOG_DISAB1,MCL_SUMMARY+3(AP) ;LOG THAT WE DID IT
                            11  11  01AE    557          BRB     20$
         0029'CF    0120    8F  A8  01B0    558 10$:     BISW    #CH_MISSG0!CH_REPLG1@-8,W^GL_CHSTATE+1 ;DISABLE GROUP 0
            0029'CF    40    8F  8A  01B7    559          BICB    #CH_REPLG0@-8,W^GL_CHSTATE+1 ;DON'T FORCE REPLACE IN BOTH!
              07 AC    01    90  01BD    560          MOVB    #CH_LOG_DISAB0,MCL_SUMMARY+3(AP) ;LOG THAT WE DID IT
         0008'CF    0004'CF   D0  01C1    561 20$:     MOVL    W^GL_CH1OLD,W^GL_CH2OLD ;MAINTAIN THE TIMING HISTORY
              0004'CF    8E    D0  01C8    562          MOVL    (SP)+,W^GL_CH1OLD       ;UNTO THE THIRD GENERATION
              33   0028'CF     DA  01CD    563          MTPR    W^GL_CHSTATE,#PR780$_SBIMT ;RE-ENABLE THE CACHE - FINALLY!
                      FEE6      31  01D2    564 BRESUM:  BRW     TRYRESUME               ;SEE IF WE CAN CONTINUE FROM THE ERROR
```

```
                        01D5   566          .SBTTL  CP TIMEOUT / SBI ERROR CONFIRMATION
                        01D5   567
                        01D5   568 CPTIMEOUT:
33   0028'CF    DA      01D5   569          MTPR    W^GL_CHSTATE,#PR780$_SBIMT ;ENABLE THE CACHE
  FC AC    06    C8      01DA   570          BISL    #MCHK$M_MCK!MCHK$M_NEXM,-4(AP) ;SET TYPE FOR PRTCTEST
                        01DE   571
     000C'CF    DD      01DE   607          PUSHL   W^GL_CPTIMOUT           ;WE ONLY KEEP TRACK OF ONE TIMEOUT
                        01E2   608          MFPR    #PR780$_TODR,W^GL_CPTIMOUT ;UPDATE THAT HISTORY
000C'CF    8E    D1      0227   609          CMPL    (SP)+,W^GL_CPTIMOUT     ;ARE TIMEOUTS LESS THAN 10 MS APART?
        A4    12      022C   610          BNEQ    BRESUM                  ;BRANCH IF NOT TO TRY AND CONTINUE
      FEC1    31      022E   611          BRW     AMPUTATE                ;OTHERWISE SOMETHING IS VERY WRONG
                        0231   612
```

```
                        0231   631          .SBTTL  READ DATA SUBSTITUTE ERROR
                        0231   632
                        0231   633          .ENABL  LSB
                        0231   634 READSUBST:
33   0028'CF    DA      0231   635          MTPR    W^GL_CHSTATE,#PR780$_SBIMT ;REENABLE CACHE
  FC AC    02   C8      0236   636          BISL    #MCHK$M_MCK,-4(AP)         ;SET MACHINE CHECK TYPE FOR PRTCTEST
  51    2C AC   DE      023A   637          MOVAL   MCL_PC(AP),R1              ;SET POINTER TO PC,PSL
     53    08   D0      023E   638          MOVL    #EMB$K_HE,R3               ;SET HARD MEMORY ERROR TYPE
        018E   30      0241   639          BSBW    LOG_ERROR_MEM              ;LOG MEMORY ERROR
        FEAB   31      0244   641          BRW     AMPUTATE                   ;ABORT -- RECOVERY IS USELESS
```

```
                     0247   730          .SBTTL  INTERFACE FROM MACHINE CHECK HANDLER TO ERROR LOGGER
                     0247   731 ;++
                     0247   732 ; LOGGER - Routine to log Machine Check interrupts and aborts
                     0247   733 ;
                     0247   734 ; INPUTS:
                     0247   735 ;
                     0247   736 ;       R3 - Error log type
                     0247   737 ;       AP - Pointer to Machine Check error log frame
                     0247   738 ;       -4(AP) - MASK FOR PRTCTEST
                     0247   739 ;       -8(AP) - PC,PSL POINTER FOR PRTCTEST
                     0247   740 ;
                     0247   741 ; OUTPUTS:
                     0247   742 ;
                     0247   743 ;       Entry made in error log conditional on PRTCTEST
                     0247   744 ;       R0-R5 destroyed
                     0247   745 ;--
                     0247   746
                     0247   747 LOGGER:
      54   08   6C   C1  0247   748          ADDL3   MCL_COUNT(AP),#<2*4>,R4 ;GET SIZE OF ENTRY IN BYTES
        55   04 AC   9E  024B   749          MOVAB   MCL_SUMMARY(AP),R5      ;GET ADDRESS OF ENTRY
        51   F8 AC   7D  024F   750          MOVQ    -8(AP),R1               ;GET MASK AND PC POINTER FOR PRTCTEST
 00000000'GF   D6  0253   755          INCL    G^EXE$GL_MCHKERRS       ;KEEP COUNT OF MACHINE CHECKS
                     0259   756 10$:                                  ;FALL THROUGH TO "LOGIT"
                     0259   757
                     0259   758 ;++
                     0259   759 ; LOGIT - INTERFACE TO SYSTEM ERROR LOG
                     0259   760 ;
                     0259   761 ; INPUTS:
                     0259   762 ;
                     0259   763 ;       R1 = PC,PSL POINTER FOR PRTCTEST
                     0259   764 ;       R2 = MASK FOR PRTCTEST
                     0259   765 ;       R3 = ERROR LOG TYPE
                     0259   766 ;       R4 = SIZE OF LOG ENTRY IN BYTES
                     0259   767 ;       R5 = ADDRESS OF LOG ENTRY
                     0259   768 ;       (SP) = RETURN ADDRESS
                     0259   769 ;--
                     0259   770          .ENABL  LSB
                     0259   771
                     0259   772 LOGIT:
                     0259   773          MFPR    #PR780$_SBIFS,R0        ;GET SBI FAULT/STATUS REGISTER
    00 50   19   E5  025C   774          BBCC    #SBIFS$V_NEF,R0,10$     ;CLEAR NESTED ERROR FLAG
       30   50   DA  0260   775 10$:     MTPR    R0,#PR780$_SBIFS        ;WRITE IT BACK TO CLEAR SILO LOCK
                     0263   776                                          ;AND FAULT LATCH
                     0263   777          MFPR    #PR780$_SBIER,R0        ;GET SBI ERROR REGISTER
50   70C0 8F   A8  0266   778          BISW    #SBIER$M_IBTO!SBIER$M_IBRDS!SBIER$M_CPTO!SBIER$M_RDS!-
                     026B   779                  SBIER$M_CRD,R0          ;SET BITS FOR ERRORS WE'RE HANDLING
       34   50   DA  026B   780          MTPR    R0,#PR780$_SBIER        ;WRITE IT BACK TO CLEAR LATCHES
                     026E   781
                     026E   786
                     026E   787 MCHK$GL_LOG::
                     026E   788
51   54   10   C1  026E   789          ADDL3   #EMB$B_MC_SUMCOD,R4,R1  ;ADD SPACE FOR HEADER FOR BUFFER SIZE
                     0272   790
 00000000'GF   16  0272   794          JSB     G^MPS$ALLOCEMB          ;GET AN ERROR LOGGING BUFFER
                     0278   796
       14 50   E9  0278   797          BLBC    R0,20$                  ;BRANCH IF DIDN'T GET IT
          52   DD  027B   798          PUSHL   R2                      ;SAVE ADDRESS OF ERROR LOG BUFFER
```

```
                04 A2    53    B0  027D    799          MOVW    R3,EMB$W_MC_ENTRY(R2)    ;SET ENTRY TYPE TO FAULT TYPE
            10 A2 65     54    28  0281    800          MOVC3   R4,(R5),EMB$B_MC_SUMCOD(R2) ;IN ONE SWELL FOOP.....
                52       8E    D0  0286    801          MOVL    (SP)+,R2                 ;GET POINTER TO BUFFER START IN R2
                                   0289    802
        00000000'GF      16    0289    806          JSB     G^MPS$RELEASEMB          ;INDICATE BUFFER READY TO LOG
                                   028F    808
                         05    028F    809 20$:     RSB                              ;EXIT WITH HARDWARE LOG STILL ON STACK
                                   0290    810
                               0290    811          .DSABL  LSB
```

```
                     0290   813
                     0290   814          .SBTTL  SBI ERROR INTERRUPTS
                     0290   815 ;++
                     0290   816 ; Handle SBI Faults and Asynchronous Write Timeouts on the SBI.
                     0290   817 ;
                     0290   818 ; SBI Fault:
                     0290   819 ;       Log the error; try to resume normal execution.
                     0290   820 ;
                     0290   821 ; Asynchronous Write Timeouts:
                     0290   822 ;       Log the error.
                     0290   823 ;       Set up a "fake" machine check log on the stack.  This is so we
                     0290   824 ;       can share the exception exit path (REFLECTCHK) that machine checks
                     0290   825 ;       take if the current process is executing in USER or SUPER mode.
                     0290   826 ;       If the current process is in EXEC or KERNEL mode, bugcheck.
                     0290   827 ;--
                     0290   828          .ALIGN  LONG                           ;THIS IS VECTORED TO
                     0290   829 GBLDEF   INT5C                                 ;SBI FAULT VECTOR
                     0290   830 GBLDEF   LOGSBF
                     0290   831          SETIPL  #^X1F                          ;DISABLE ALL INTERRUPTS
00FF 8F    BB        0293   832          PUSHR   #^M<R0,R1,R2,R3,R4,R5,R6,R7>   ;SAVE SOME WORK REGS
53   04    9A        0297   833          MOVZBL  #EMB$K_BE,R3                   ;ERROR LOG TYPE
52   03    D0        029A   834          MOVL    #MCHK$M_MCK!MCHK$M_LOG,R2      ;MASK FOR PRTCTEST
     3E    10        029D   835          BSBB    LOGSBI                         ;USE SAME CODE AS ASYNC WRITE FAILURE
00FF 8F    BA        029F   836          POPR    #^M<R0,R1,R2,R3,R4,R5,R6,R7>   ;RESTORE R0-R7
           02        02A3   837          REI                                    ;TRY TO CONTINUE
                     02A4   838
                     02A4   839          .ALIGN  LONG                           ;THIS IS VECTORED TO
                     02A4   840 GBLDEF   INT60                                  ;ASYNCHRONOUS WRITE TIMEOUT
                     02A4   841 GBLDEF   LOGAWE
                     02A4   842          SETIPL  #^X1F                          ;DISABLE ALL INTERRUPTS
          00FF 8F    BB     02A7   843          PUSHR   #^M<R0,R1,R2,R3,R4,R5,R6,R7>   ;SAVE SOME WORK REGS
          53   07    9A     02AB   844          MOVZBL  #EMB$K_AW,R3                   ;ERROR LOG TYPE
          52   07    D0     02AE   845          MOVL    #MCHK$M_LOG!MCHK$M_MCK!MCHK$M_NEXM,R2 ;PRTCTEST MASK
               2A    10     02B1   846          BSBB    LOGSBI                         ;USE SAME CODE AS SBI FAULT ERROR
          00FF 8F    BA     02B3   847          POPR    #^M<R0,R1,R2,R3,R4,R5,R6,R7>   ;RESTORE R0-R7
          5E   28    C2     02B7   848          SUBL    #40,SP                         ;ALLOCATE FAKE MACHINE CHECK FRAME
               28    DD     02BA   849          PUSHL   #40                            ;SIZE OF FRAME
               07    DD     02BC   850          PUSHL   #MCHK$M_MCK!MCHK$M_LOG!MCHK$M_NEXM
                 30  AE     02BE   851          PUSHAL  MCL_PC+4(SP)                   ;MASK AND PC,PSL FOR PRTCTEST
          103F 8F    BB     02C1   852          PUSHR   #^M<R0,R1,R2,R3,R4,R5,AP>      ;SAVE REGISTERS FOR COMMON CODE
     5C   5E    24    C1     02C5   853          ADDL3   #<9*4>,SP,AP                   ;POINT AP TO FAKE MACHINE CHECK FRAME
0003'CF    A0 8F    93     02C9   854          BITB    #^B10100000,W^GL_CSBITA+3      ;WAS WRITE IN USER OR SUPERVISOR
                          02CF   855                                                 ;MODE AND NOT UPDATING A PAGE TABLE
               03    12     02CF   856          BNEQ    10$                            ;IF NOT, MUST BUGCHECK
                  FE32 31     02D1   857          BRW     REFLECTCHK                     ;BRANCH IF OK TO CONTINUE
                          02D4   858 10$:
          103F 8F    BA     02D4   859          POPR    #^M<R0,R1,R2,R3,R4,R5,AP>
                          02D8   865          SECBUG_CHECK    MPASYNCWRT,FATAL;WRITE ERROR IN KERNEL OR EXEC MODE
```

```
                                02DD   868 ;++
                                02DD   869 ; LOGSBI -- Subroutine to log SBI errors.
                                02DD   870 ;
                                02DD   871 ; Implicit Inputs:
                                02DD   872 ;       +-----------------------+
                                02DD   873 ;       |   return address      | :(SP)
                                02DD   874 ;       +-----------------------+
                                02DD   875 ;       |       saved           |
                                02DD   876 ;       |       R0 - R7         |
                                02DD   877 ;       +-----------------------+
                                02DD   878 ;       |   interrupt PC        |
                                02DD   879 ;       +-----------------------+
                                02DD   880 ;       |   interrupt PSL       |
                                02DD   881 ;       +-----------------------+
                                02DD   882 ;
                                02DD   883 ; Create an SBI error log buffer that contains:
                                02DD   888 ;       The contents of the configuration register of every MA780
                                02DD   889 ;       SBI adapter on the bus or 0 (16 longwords).
                                02DD   891 ;       A copy of the SBI silo (16 longwords).
                                02DD   892 ;       SBI processor registers SBITA, SBIER, SBIMT, SBISC, and SBIFS.
                                02DD   893 ;--
                                02DD   894 LOGSBI:                                      ;LOG SBI ERROR
    00000000'GF     D6          02DD   899         INCL    G^EXE$GL_MCHKERRS            ;KEEP COUNT OF MACHINE CHECKS
                                02E3   900 5$:
     7E   24 AE     7D          02E3   901         MOVQ    <9*4>(SP),-(SP)              ;MAKE A SECOND COPY OF PC,PSL
57   00000000'GF     D0          02E7   902         MOVL    G^EXE$GL_CONFREGL,R7         ;ARRAY OF NEXUS DEVICE TYPE CODES
55   00000000'GF     D0          02EE   903         MOVL    G^MMG$GL_SBICONF,R5          ;ARRAY OF ADAPTER VA'S
        50   0F     D0          02F5   904         MOVL    #15,R0                       ;INDEX OF LAST POSSIBLE ITEM ON SBI
             7E     D4          02F8   905 10$:     CLRL    -(SP)                        ;ASSUME NO ADAPTOR HERE
      51   6540     D0          02FA   906         MOVL    (R5)[R0],R1                  ;GET VA OF CONTROLLER/ADAPTER
             1C     18          02FE   907         BGEQ    20$                          ;GEQ IMPLIES NO VALID SYSTEM VA.
           6740     D5          0300   908         TSTL    (R7)[R0]                     ;TEST ADAPTER TYPE (ONLY WORKS FOR SBI)
             17     13          0303   909         BEQL    20$                          ;IF EQL, NO ADAPTOR HERE
                                0305   911         ASSUME  <NDT$_MPM0+1> EQ NDT$_MPM1
                                0305   912         ASSUME  <NDT$_MPM0+2> EQ NDT$_MPM2
                                0305   913         ASSUME  <NDT$_MPM0+3> EQ NDT$_MPM3
00000040 8F  6740     D1          0305   914         CMPL    (R7)[R0],#NDT$_MPM0          ;IS THIS AN MA780?  IF NOT, THEN
               0D     1F          030D   915         BLSSU   20$                          ; THE SECONDARY CANNOT TOUCH IT AS
00000043 8F  6740     D1          030F   916         CMPL    (R7)[R0],#NDT$_MPM3          ; I/O SPACE IS DIFFERENT THAN ON THE
               03     1A          0317   917         BGTRU   20$                          ; ON THE PRIMARY (ONLY MA780S ARE SAME).
          6E   61     D0          0319   919         MOVL    (R1),(SP)                    ;STORE ADAPTOR CSR0 ON STACK
            D9 50     F4          031C   920 20$:     SOBGEQ  R0,10$                       ;LOOP THRU ALL POSSIBLE 16
          50   0F     D0          031F   921         MOVL    #15,R0                       ;SET UP COUNT OF NUMBER OF TIMES TO
                                0322   922                                              ;READ SILO
                                0322   923 30$:     MFPR    #PR780$_SBIS,-(SP)           ;SAVE INFORMATION FOR ERROR LOGGER
             FA 50     F4          0325   924         SOBGEQ  R0,30$                       ;LOOP THRU ALL 16
                                0328   925         MFPR    #PR780$_SBITA,-(SP)          ;SAVE SBI TIMEOUT REGISTER
    0000'CF   6E     D2          032B   926         MCOML   (SP),W^GL_CSBITA             ;SAVE COMPLEMENT SBITA FOR LATER CHECK
                                0330   927         MFPR    #PR780$_SBIER,-(SP)          ;SAVE SBI ERROR REGISTER
                                0333   928         MFPR    #PR780$_SBIMT,-(SP)          ;SAVE SBI MAINTENANCE REGISTER
                                0336   929         MFPR    #PR780$_SBISC,-(SP)          ;SAVE SBI SILO COMPARATOR
                                0339   930         MFPR    #PR780$_SBIFS,-(SP)          ;SAVE SBI FAULT/STATUS REGISTER
    7E   009C 8F     3C          033C   931         MOVZWL  #<16*4>+<16*4>+<7*4>,-(SP) ;SAVE NUMBER OF BYTES OF ENTRY
                                0341   932
    51   0098 CE     DE          0341   933         MOVAL   <<16*4>+<16*4>+<6*4>>(SP),R1 ;ADDRESS OF PC,PSL FOR PRTCTEST
         54   6E     D0          0346   934         MOVL    (SP),R4                      ;# OF BYTES TO LOG
      55   04 AE     9E          0349   935         MOVAB   4(SP),R5                     ;ADDRESS OF LOG ENTRY
```

```
                FF09    30  034D   936          BSBW    LOGIT                   ;CALL ERROR LOGGER
             5E   8E    C0  0350   937          ADDL    (SP)+,SP                ;CLEAN STACK OF LOG AND FAKE PC,PSL
                   05    05  0353   938          RSB                             ;RETURN
                         0354   939
```

```
                                0354   941          .SBTTL  MEMORY TIMER SCAN
                                0354   942
                                0354   946 MPS$REENABLE::                          ; Define timer scan entry point.
                                0354   948
                 0022'CF    B7  0354   949          DECW    W^GW_WATCH              ; Count seconds down.
                       19   14  0358   950          BGTR    REENABLE_INTS           ; Br if time hasn't elapsed yet.
                                035A   951 ;
                                035A   952 ; Scan all memory controllers for unreported errors.  This will yield a
                                035A   953 ; representative sample of memory errors in the error log, even if CRD
                                035A   954 ; interrupts are disabled.
                                035A   955 ;
               0022'CF    3C   B0  035A   956          MOVW    #SOMETIME,W^GW_WATCH    ; Reset time interval.
                          0A   BB  035F   957          PUSHR   #^M<R1,R3>              ; Save working registers.
                          7E   DC  0361   958          MOVPSL  -(SP)                   ; Set up interrupt PSL.
                          00   10  0363   959          BSBB    10$                     ; Fake interrupt PC on stack.
                    51    6E   DE  0365   960 10$:     MOVAL   (SP),R1                 ; Point to exception PC,PSL.
                    53    06   D0  0368   961          MOVL    #EMB$K_SE,R3            ; Log soft memory errors.
                       0064   30  036B   962          BSBW    LOG_ERROR_MEM           ; Scan all memory controllers, and
                                036E   963                                          ; log any that report errors.
                    5E    08   C0  036E   964          ADDL    #8,SP                   ; Pop PC,PSL pair from stack.
                          0A   BA  0371   965          POPR    #^M<R1,R3>              ; Restore registers.
                                0373   966
                                0373   967 REENABLE_INTS:
                 0020'CF    B7  0373   968          DECW    W^GW_REENAB             ; Count seconds down.
                          2B   14  0377   969          BGTR    10$                     ; Branch if reenable time not elapsed.
          0020'CF    0384 8F   B0  0379   970          MOVW    #REENABTIME,W^GW_REENAB ; Reset the time interval.
                          3F   BB  0380   971          PUSHR   #^M<R0,R1,R2,R3,R4,R5>  ; Save working registers.
0010'CF   10   00    00 8F   00  2C  0382   972          MOVC5   #0,#0,#0,#16,W^AB_MEMERR; Reset 16 bytes of error count to 0.
             00000000'GF    00'  E1  038B   973          BBC     S^#EXE$V_CRDENABL, -    ; Branch if SYSGEN parameter specifies
                          0F       0392   974                  G^EXE$GL_FLAGS,5$       ; no CRD interrupts.
             55   00000000'GF    D0  0393   975          MOVL    G^MMG$GL_SBICONF,R5     ; Get addr of SBICONF for action routines.
                  53    0000'CF   DE  039A   976          MOVAL   W^ENAB_ROUTINES,R3      ; Array of action routine vectors.
                           008F   30  039F   977          BSBW    LOCATE_MEM              ; Locate mem and call action routines
                                  03A2   978                                          ; to re-enable CRD interrupts.
                          3F   BA  03A2   979 5$:      POPR    #^M<R0,R1,R2,R3,R4,R5>  ; Restore registers.
                          05   03A4   980 10$:     RSB
```

```
                        03A5   982          .SBTTL  Memory Error Interrupts
                        03A5   983 ;
                        03A5   984 ; SBI Alert interrupts are vectored here.
                        03A5   985 ;
                        03A5   986          .ALIGN  LONG
                        03A8   987 GBLDEF   INT58                           ; EXE$INT58:: or MPS$INT58::
                        03A8   988 GBLDEF   LOGSBA                          ; EXE$LOGSBA:: or MPS$LOGSBA::
                        03A8   989
          0A    BB      03A8   990          PUSHR   #^M<R1,R3>              ; Save some registers.
                        03AA   991          SETIPL  #^X1F                   ; Disable all interrupts.
    51  08 AE   DE      03AD   992          MOVAL   8(SP),R1                ; Set pointer to interrupt PC,PSL.
        53   05 D0      03B1   993          MOVL    #EMB$K_SA,R3            ; Set SBI Alert error log type.
         001B  30      03B4   994          BSBW    LOG_ERROR_MEM           ; Log memory controller registers.
          0A   BA      03B7   995          POPR    #^M<R1,R3>              ; Restore registers.
               02      03B9   996          REI
                        03BA   997
                        03BA   998 ;
                        03BA   999 ; CRD (Soft, or Corrected) memory error interrupts are vectored here.
                        03BA  1000 ;
                        03BA  1001          .ALIGN  LONG
                        03BC  1002 GBLDEF   INT54                           ; EXE$INT54:: or MPS$INT54::
                        03BC  1003 GBLDEF   LOGCRD                          ; EXE$LOGCRD:: or MPS$LOGCRD::
                        03BC  1004
          0A    BB      03BC  1005          PUSHR   #^M<R1,R3>              ; Save some registers.
                        03BE  1006          SETIPL  #^X1F                   ; Disable all interrupts.
  0024'CF     D6      03C1  1007          INCL    W^GL_CRDCNT             ; Keep count of these errors.
    51  08 AE   DE      03C5  1008          MOVAL   8(SP),R1                ; Set pointer to interrupt PC,PSL.
        53   06 D0      03C9  1009          MOVL    #EMB$K_SE,R3            ; Set soft memory error type.
         0003  30      03CC  1010          BSBW    LOG_ERROR_MEM           ; Log memory controller registers.
          0A   BA      03CF  1011          POPR    #^M<R1,R3>
               02      03D1  1012          REI
```

```
                        03D2  1014          .SBTTL  LOGMEM Master Routine
                        03D2  1015 ;++
                        03D2  1016 ;
                        03D2  1017 ; FUNCTIONAL DESCRIPTION:
                        03D2  1018 ;       This routine is called to build an errorlog containing the device
                        03D2  1019 ;       registers of the memory controllers on an 11/780 system.  If called
                        03D2  1020 ;       at the LOG_ERROR_MEM entry point, it will scan the memory controller
                        03D2  1021 ;       status registers, and only log those controllers which report errors.
                        03D2  1022 ;       If called at the LOG_ALL_MEM entry point, it will unconditionally log
                        03D2  1023 ;       all memory controllers on the system.
                        03D2  1024 ;
                        03D2  1025 ; INPUTS:
                        03D2  1026 ;       R1      - pointer to exception PC,PSL
                        03D2  1027 ;       R3      - Error log type code (e.g. EMB$K_type)
                        03D2  1028 ;
                        03D2  1029 ; OUTPUTS:
                        03D2  1030 ;       Format of error log:
                        03D2  1031 ;               # of memory controllers logged
                        03D2  1032 ;               memory type-specific log #1
                        03D2  1033 ;               memory type-specific log #2
                        03D2  1034 ;                       .
                        03D2  1035 ;                       .
                        03D2  1036 ;               PC of instruction at fault time
                        03D2  1037 ;               PSL at fault time
                        03D2  1038 ;
                        03D2  1039 ;       All registers are preserved.
                        03D2  1040 ;
                        03D2  1041 ;--
                        03D2  1042
                        03D2  1043 LOG_ERROR_MEM:                          ; Log controllers with errors.
      1FFF 8F   BB      03D2  1044          PUSHR   #^M<R0,R1,R2,R3,R4,R5,R6,R7,R8,R9,R10,R11,AP>
 53   0000'CF   DE      03D6  1045          MOVAL   W^LOGERR_ROUTINES,R3    ; Array of action routine vectors.
           09   11      03DB  1046          BRB     LOGMEM                  ; Join common code.
                        03DD  1047
                        03DD  1048 LOG_ALL_MEM:                            ; Unconditionally log all controllers.
      1FFF 8F   BB      03DD  1049          PUSHR   #^M<R0,R1,R2,R3,R4,R5,R6,R7,R8,R9,R10,R11,AP>
 53   0000'CF   DE      03E1  1050          MOVAL   W^LOGALL_ROUTINES,R3    ; Array of action routine vectors.
                        03E6  1051
                        03E6  1052 LOGMEM:                                 ; Log memory controller registers.
           55   7C      03E6  1053          CLRQ    R5                      ; Zero error log byte count and number
                        03E8  1054                                          ; of controllers logged.
57  00000000'GF   D0    03E8  1055          MOVL    G^MMG$GL_SBICONF,R7     ; For use by action routines.
         5C   01   D0    03EF  1056          MOVL    #SS$_NORMAL,AP          ; Assume no fatal memory errors.
                        03F2  1057 ;
                        03F2  1058 ; Locate all memory controllers on the SBI.  When a memory controller is
                        03F2  1059 ; found, call the appropriate action routine to create that controller's
                        03F2  1060 ; portion of the common error log buffer on the stack.
                        03F2  1061 ;
             003C   30  03F2  1062          BSBW    LOCATE_MEM
                        03F5  1063 ;
                        03F5  1064 ; The error log buffer has been built on the stack; SP points to the beginning.
                        03F5  1065 ; Add the number of memory controllers logged, then log the errors.
                        03F5  1066 ; Current register usage:
                        03F5  1067 ;       R5      - Number of bytes in the error log.
                        03F5  1068 ;       R6      - Number of memory controllers logged.
                        03F5  1069 ;       SP      - Points to the beginning of the error log buffer.
                        03F5  1070 ;       AP      - LBS if no fatal memory errors were discovered, else LBC.
```

```
                        03F5  1071 ;
   51    6E45    9E     03F5  1072          MOVAB   (SP)[R5],R1              ; Get address of saved R0 on stack.
  53    0C A1    D0     03F9  1073          MOVL    12(R1),R3               ; Restore input value of R3.
  51    04 A1    D0     03FD  1074          MOVL    4(R1),R1                ; Restore input value of R1.
           56    DD     0401  1075          PUSHL   R6                      ; Add # of controllers to log buffer.
7E   55    04    C1     0403  1076          ADDL3   #<1*4>,R5,-(SP)         ; Total # bytes in error log buffer.
           55    D5     0407  1077          TSTL    R5                      ; Were any memory registers logged?
           12    13     0409  1078          BEQL    10$                     ; No.  Skip call to error logger.
 00000000'GF     D6     040B  1079          INCL    G^EXE$GL_MEMERRS        ; Keep count of memory errors
     54    6E    D0     0411  1080          MOVL    (SP),R4                 ; Use # bytes as input to LOGIT.
  55    04 AE    DE     0414  1081          MOVAL   4(SP),R5                ; Address of error log buffer.
           52    D4     0418  1082          CLRL    R2                      ; Always log memory errors.
        FE3C    30     041A  1083          BSBW    LOGIT                   ; Log the error.
     5E    8E    C0     041D  1084 10$:     ADDL    (SP)+,SP                ; Remove error log buffer from stack.
       09 5C    E8     0420  1085          BLBS    AP,20$                  ; Br if fatal error not signalled.
     1FFF 8F    BA     0423  1086          POPR    #^M<R0,R1,R2,R3,R4,R5,R6,R7,R8,R9,R10,R11,AP>
                       0427  1090          SECBUG_CHECK    MPASYNCWRT,FATAL; Unrecoverable memory controller err
                       042C  1092 20$:
     1FFF 8F    BA     042C  1093          POPR    #^M<R0,R1,R2,R3,R4,R5,R6,R7,R8,R9,R10,R11,AP>
                05     0430  1094          RSB
```

```
                     0431  1096          .SBTTL  LOCATE_MEM Dispatching Routine
                     0431  1097 ;++
                     0431  1098 ;       Routine to locate memory controllers on 11/780 SBI.
                     0431  1099 ;
                     0431  1100 ; FUNCTIONAL DESCRIPTION:
                     0431  1101 ;       This routine scans an array of adapter type codes that tell which
                     0431  1102 ;       adapters are attached to the SBI.  When it finds a memory controller
                     0431  1103 ;       adapter, it dispatches to an action routine for that memory controller
                     0431  1104 ;       type.
                     0431  1105 ;
                     0431  1106 ; INPUTS:
                     0431  1107 ;       R3 - address of action routine table; 1 action routine/memory controller
                     0431  1108 ;       Current format of action routine tables (the tables are created by the
                     0431  1109 ;       MEMORY_ROUTINES macro):
                     0431  1110 ;               (R3):  self-relative offset to MS780C action routine
                     0431  1111 ;              4(R3):  self-relative offset to MA780 action routine
                     0431  1112 ;              8(R3):  self-relative offset to MS780E action routine
                     0431  1113 ;
                     0431  1114 ; On entry to memory action routine:
                     0431  1115 ;       R0,R1   - local registers, not preserved across calls to action routines
                     0431  1116 ;       R2      - TR# of this memory controller
                     0431  1117 ;       R3      - not available to be used by action routines
                     0431  1118 ;       R4      - address of CONFREGL array (If the 780 ever gets a BI, code
                     0431  1119 ;                 must change, because TSTL assumes no high-order bits set.)
                     0431  1120 ;       R5-AP   - available; contents are preserved across calls to multiple
                     0431  1121 ;                 action routines (i.e. can be used for global storage)
                     0431  1122 ;
                     0431  1123 ;       Note: an action routine may deposit a -1 in R2 to cause LOCATE_MEM
                     0431  1124 ;       to prematurely exit the memory scan loop (and not call any other
                     0431  1125 ;       memory action routines).
                     0431  1126 ;
                     0431  1127 ; OUTPUTS:
                     0431  1128 ;       R0-R4 destroyed.  (Other registers may be destroyed by action routines.)
                     0431  1129 ;--
                     0431  1130
                     0431  1131 LOCATE_MEM:
      54  00000000'GF   D0  0431  1132          MOVL    G^EXE$GL_CONFREGL,R4    ; Get address of CONFREGL.
52    00000000'GF   01   C3  0438  1133          SUBL3   #1,G^EXE$GL_NUMNEXUS,R2 ; Get index into nexus arrays.
                     0440  1134 ;
                     0440  1135 ; Loop through all nexuses.  If a memory controller is found at any of the
                     0440  1136 ; nexus slots, then call the action routine associated with that memory.
                     0440  1137 ;
            51   6442   D0  0440  1138 10$:     MOVL    (R4)[R2],R1             ; Get nexus device type from CONFREGL.
                   14   13  0444  1139          BEQL    20$                     ; Not a memory; go to next nexus.
    0000'CF   12   51   3A  0446  1140          LOCC    R1,#MEMTYPCNT,W^MEMTYP  ; Find type in memory type array.
                     044C  1141                                                  ; R1 <- addr of type code (if found).
                   0C   13  044C  1142          BEQL    20$                     ; Not a memory; go to next nexus.
          51   12 A1   9A  044E  1143          MOVZBL  MEMTYPCNT(R1),R1        ; Use offset to get general memory type.
           51   6341   DE  0452  1144          MOVAL   (R3)[R1],R1             ; Get self-relative address of action
             00 B141   16  0456  1145          JSB     @(R1)[R1]               ; routine, and call it.
                E3 52   F4  045A  1146 20$:     SOBGEQ  R2,10$                  ; Loop through all nexuses.
                        05  045D  1147          RSB                             ; Return.
```

```
                        045E  1149          .SBTTL  ENAB Action Routines
                        045E  1150 ;++
                        045E  1151 ;
                        045E  1152 ; FUNCTIONAL DESCRIPTION:
                        045E  1153 ;       These action routines re-enable CRD interrupts for each 11/780 memory
                        045E  1154 ;       controller.  Memory types currently supported:
                        045E  1155 ;
                        045E  1156 ;               MS780C (local memory - 4k and 16k chips)
                        045E  1157 ;               MS780E (local memory - 64k chips)
                        045E  1158 ;               MA780  (multiport memory)
                        045E  1159 ;
                        045E  1160 ; INPUTS:
                        045E  1161 ;       R2      - TR# of this memory
                        045E  1162 ;       R4      - address of EXE$GL_CONFREGL array
                        045E  1163 ;       R5      - address of MMG$GL_SBICONF array
                        045E  1164 ;
                        045E  1165 ; OUTPUTS:
                        045E  1166 ;       R0,R1 destroyed; all other registers preserved.
                        045E  1167 ;
                        045E  1168 ;--
                        045E  1169 ENAB_MS780C:
                     05 045E  1175          RSB                             ; That's it.
                        045F  1176
                        045F  1177 ENAB_MS780E:
                     05 045F  1185          RSB                             ; That's it.
                        0460  1186
                        0460  1187 ENAB_MA780:
                        0460  1188          MOVL    (R5)[R2],R1             ; Get address of controller registers.
          10 A1 30000000 8F  C8 0464  1193          BISL    #<MRC$M_ELSRF!MRC$M_HERIMF>,16(R1) ; Enable interrupts
                        046C  1194                                          ; and clear error flags.
                     05 046C  1198          RSB                             ; That's it.
```

```
046D  1200          .SBTTL  LOGMEM Action Routines
046D  1201 ;++
046D  1202 ;  FUNCTIONAL DESCRIPTION:
046D  1203 ;       One action routine per memory controller type follows.  These
046D  1204 ;       routines create an 11/780 memory error log entry.  Currently, the
046D  1205 ;       following memory controllers are supported:
046D  1206 ;
046D  1207 ;               MS780C (local memory - 4K and 16K chips)
046D  1208 ;               MS780E (local memory - 64K chips)
046D  1209 ;               MA780  (multiport memory)
046D  1210 ;
046D  1211 ;       Each action routine contributes to the common error log buffer being
046D  1212 ;       built on the stack.  Because different routines are being used to build
046D  1213 ;       a common error log buffer on the stack, the contents of the stack is
046D  1214 ;       significant at all times.
046D  1215 ;
046D  1216 ; INPUTS:
046D  1217 ;       R2 - nexus index for this memory (TR #)
046D  1218 ;       R3 - not available for use by action routines
046D  1219 ;       R4 - address of SBI configuration array (CONFREGL)
046D  1220 ;       R5 - current errorlog byte count
046D  1221 ;       R6 - current number of controllers logged
046D  1222 ;       R7 - address of array of SBI virtual addresses (SBICONF)
046D  1223 ;       R8-R11 - scratch
046D  1224 ;       AP - memory controller status: LBC = fatal controller error discovered
046D  1225 ;
046D  1226 ; IMPLICIT INPUTS:
046D  1227 ;                   +-----------------------------------------+
046D  1228 ;       (SP):       |       caller's return address           |
046D  1229 ;                   +-----------------------------------------+
046D  1230 ;                   |       return to caller's caller         |
046D  1231 ;                   +-----------------------------------------+
046D  1232 ;                   |         previous error log              |
046D  1233 ;                   |                 :                       |
046D  1234 ;                   |                 .                       |
046D  1235 ;
046D  1236 ; OUTPUTS:
046D  1237 ;       R2-R4 preserved
046D  1238 ;       R5 and R6 updated
046D  1239 ;
046D  1240 ; IMPLICIT OUTPUTS:
046D  1241 ;                   +-----------------------------------------+
046D  1242 ;       (SP):       |       return to caller's caller         |
046D  1243 ;                   +-----------------------------------------+
046D  1244 ;                   |  error log entry for this controller    |
046D  1245 ;                   |  (null if no error for this memory)     |
046D  1246 ;                   +-----------------------------------------+
046D  1247 ;                   |       previous error log                |
046D  1248 ;                   |                 :                       |
046D  1249 ;                   |                 .                       |
046D  1250 ;
046D  1251 ;--
```

```
                        .SBTTL  LOG_MS780C
                ;++
                ;       LOG_MS780C - Build error log for MS780C memory controller
                ;
                ;       The portion of the error log built for the MS780C memory controller
                ;       has the following format:
                ;               +--------------------------------+
                ;               |       adapter TR#              |
                ;               +--------------------------------+
                ;               |       memory register A        |
                ;               +--------------------------------+
                ;               |       memory register B        |
                ;               +--------------------------------+
                ;               |       memory register C        |
                ;               +--------------------------------+
                ;
                ;       Register A contains the type code in the low-order byte.  For MS780C
                ;       memories, this type code is in the range of 8 - 11 (hex).
                ;--
                
                LOG_MS780C:
                ;
                ; Determine whether to log this controller.
                ;
                LOGC:
05              RSB                             ; Else return.
```

```
                        .SBTTL  LOG_MS780E
                ;++
                ;       LOG_MS780E - Build error log for MS780E memory controller
                ;
                ;       The portion of the error log built for the MS780E memory controller
                ;       has the following format:
                ;               +--------------------------------+
                ;               |       adapter TR#              |
                ;               +--------------------------------+
                ;               |       memory register A        |
                ;               +--------------------------------+
                ;               |       memory register B        |
                ;               +--------------------------------+
                ;               |       memory register C        |
                ;               +--------------------------------+
                ;               |       memory register D        |
                ;               +--------------------------------+
                ;
                ;       Register A contains the type code in the low-order byte.  For MS780E
                ;       memories, this type code is in the range of 68 - 6C (hex).
                ;
                ;--
                
                LOG_MS780E:
                ;
                ; Determine whether to log any errors for this controller.
                ;
                LOGE:
05              RSB                             ; Else return.
```

```
                        046F  1458          .SBTTL  LOG_MA780
                        046F  1459 ;++
                        046F  1460 ;       LOG_MA780 - Build error log for MA780 memory controller
                        046F  1461 ;
                        046F  1462 ;       The portion of the error log built for the MA780 memory controller
                        046F  1463 ;       has the following format:
                        046F  1464 ;                +--------------------------------+
                        046F  1465 ;                |       adapter TR#              |
                        046F  1466 ;                +--------------------------------+
                        046F  1467 ;                | Port Configuration Register    |
                        046F  1468 ;                +--------------------------------+
                        046F  1469 ;                | Port Interface Control Reg     |
                        046F  1470 ;                +--------------------------------+
                        046F  1471 ;                | Port Controller Status Reg     |
                        046F  1472 ;                +--------------------------------+
                        046F  1473 ;                | Port Invalidation Control Reg  |
                        046F  1474 ;                +--------------------------------+
                        046F  1475 ;                | Array Error Register           |
                        046F  1476 ;                +--------------------------------+
                        046F  1477 ;                | Configuration Status Reg 0     |
                        046F  1478 ;                +--------------------------------+
                        046F  1479 ;                | Configuration Status Reg 1     |
                        046F  1480 ;                +--------------------------------+
                        046F  1481 ;                | Maintenance Control Register   |
                        046F  1482 ;                +--------------------------------+
                        046F  1483 ;
                        046F  1484 ;       The Port Configuration Register contains the type code in the
                        046F  1485 ;       low-order byte.  For MA780 memories, this type code is in the
                        046F  1486 ;       range of 40 - 43 (hex).
                        046F  1487 ;
                        046F  1488 ;--
                        046F  1489
                        046F  1490 LOG_MA780:
                        046F  1491 ;
                        046F  1492 ; Determine whether to log any errors for this controller.
                        046F  1493 ;
                        046F  1494
          58    6742    D0  046F  1495          MOVL    (R7)[R2],R8             ; Get VA of controller register.
                5A    D4  0473  1496          CLRL    R10                     ; Use R10 as memory error flag; assume
                        0475  1497                                          ; there is an error condition.
    68  00400000  8F    D3  0475  1502          BITL    #^X00400000,(R8)        ; Check for power-up interrupt.
                28    12  047C  1503          BNEQ    5$                      ; Branch if found.
04  A8  FF000000  8F    D3  047E  1504          BITL    #^XFF000000,4(R8)       ; Check Port Interface Control Reg.
                21    12  0486  1505          BNEQ    5$                      ; Branch if found error.
10  A8  10000000  8F    D3  0488  1506          BITL    #MRC$M_ELSRF,16(R8)     ; Check Array Error register.
                17    12  0490  1507          BNEQ    5$                      ; Branch if found error.
18  A8  00000400  8F    D3  0492  1508          BITL    #^X00000400,24(R8)      ; Check Multiple Interlock Accepted err.
                0D    12  049A  1509          BNEQ    5$                      ; Branch if found error.
08  A8  D000C000  8F    D3  049C  1510          BITL    #^XD000C000,8(R8)       ; Lastly, check Port Contr. Status Reg.
                03    12  04A4  1511          BNEQ    5$                      ; Branch if found error.
            5A    01    D0  04A6  1512          MOVL    #1,R10                  ; Signal no errors found.
                        04A9  1513 5$:
              01  5A    E9  04A9  1518          BLBC    R10,LOGMA               ; If any errors were found, log them.
                05    04  04AC  1519 20$:     RSB                             ; Else return.
                        04AD  1520 ;
                        04AD  1521 ; This is the entry point used when unconditionally logging all memories.
                        04AD  1522 ;
```

```
                                04AD  1523 ; Build error log on stack.  First set SP to where the top of the buffer
                                04AD  1524 ; will be, and use R9 as a temporary stack pointer while the log is being
                                04AD  1525 ; built.  This is so the machine check protection routines can freely use the
                                04AD  1526 ; stack above where the error log is being built.
                                04AD  1527 ;
                                04AD  1528 LOGMA:
                    0802 8F BA  04AD  1529         POPR    #^M<R1,R11>             ; Get return address in R1, caller's
                                04B1  1530                                         ; return in R11.
                        59 5E D0  04B1  1531         MOVL    SP,R9                  ; Use R9 as temporary stack pointer.
                        5E 24 C2  04B4  1532         SUBL    #<9*4>,SP              ; Point SP to where stack top will be.
                                04B7  1537         DSBINT  DST=R10                ; Raise IPL while logging registers.
                  79   1C A8 D0  04BD  1538         MOVL    28(R8),-(R9)           ; Maintenance Control Register
                  79   18 A8 D0  04C1  1539         MOVL    24(R8),-(R9)           ; Configuration Status Register 1
                  79   14 A8 D0  04C5  1540         MOVL    20(R8),-(R9)           ; Configuration Status Register 0
                  79   10 A8 D0  04C9  1541         MOVL    16(R8),-(R9)           ; Array Error Register
                  79   0C A8 D0  04CD  1542         MOVL    12(R8),-(R9)           ; Port Invalidation Control Register
                  79   08 A8 D0  04D1  1547         MOVL    8(R8),-(R9)            ; Read Port Controller Status Register.
                     79   00 D0  04D5  1552         MOVL    #0,-(R9)               ; Else put fake copy of register in log.
                                04D8  1553 15$:
                  79   04 A8 D0  04D8  1554         MOVL    4(R8),-(R9)            ; Port Interface Control Register
                     79   68 D0  04DC  1555         MOVL    (R8),-(R9)             ; Port Configuration Register
                                04DF  1556         ENBINT  SRC=R10                ; Restore IPL to previous level.
                     79   52 D0  04E2  1557         MOVL    R2,-(R9)               ; Save TR# in error log.
                                04E5  1558 ;
                                04E5  1559 ; Clear errors from registers.
                                04E5  1560 ;
                  68   04 A9 C8  04E5  1561         BISL    4(R9),(R8)             ; Clear errs in Port Config Reg (pwr-up)
               04 A8   08 A9 C8  04E9  1562         BISL    8(R9),4(R8)            ; Clear errors in Port Interface
                                04EE  1563                                         ; Control Register.
               08 A8   0C A9 C8  04EE  1568         BISL    12(R9),8(R8)           ; Clear errors in Port Controller
                                04F3  1569                                         ; Status register.
               14 A8   18 A9 C8  04F3  1573         BISL    24(R9),20(R8)          ; Clear errors in Port Configuration
                                04F8  1574                                         ; Status Register (Mult Interlock Accpt)
                     14 A9 DD  04F8  1575         PUSHL   20(R9)                 ; Get copy of Array Error Register
                                04FB  1576                                         ; on top of stack.
                                04FB  1577 ;
                                04FB  1578 ; Check for CRD error.  If the # of recent CRD errors > CRDINTMAX, then disable
                                04FB  1579 ; CRD interrupts for this controller.  If the # of recent CRD errors >
                                04FB  1580 ; CRDWATCHMAX, then don't log another CRD error for this controller.
                                04FB  1581 ;
                1E 6E    1C E1  04FB  1582         BBC     #MRC$V_ELSRF,(SP),40$  ; Branch if this wasn't a data error.
                 0010'CF42 96  04FF  1583         INCB    W^AB_MEMERR[R2]        ; Count data errors for this contr.
            03   0010'CF42 91  0504  1584         CMPB    W^AB_MEMERR[R2],#CRDINTMAX     ; Too many CRD interrupts?
                       04 15  050A  1585         BLEQ    30$                    ; No, skip CRD interrupt disable.
                00 6E    1E E2  050C  1586         BBSS    #MRC$V_INHBCRD,(SP),30$ ; Set bit to inhibit CRD interrupts.
                                0510  1587 30$:
                   5A   01 D0  0510  1588         MOVL    #1,R10                 ; Assume error will be logged.
            06   0010'CF42 91  0513  1589         CMPB    W^AB_MEMERR[R2],#CRDWATCHMAX   ; Too many CRD error logs?
                       02 15  0519  1590         BLEQ    40$                    ; No, go ahead and log this one.
                       5A D4  051B  1591         CLRL    R10                    ; Signal "don't log this error".
                                051D  1592 40$:
                10 A8   8E D0  051D  1593         MOVL    (SP)+,16(R8)           ; Clear errors from Array Error Reg.
                                0521  1594
                                0521  1598 ;
                                0521  1599 ; Note: If no machine check occurred, R9 and SP are now identical.  We can
                                0521  1600 ; resume using SP.
                                0521  1601 ;
```

```
                                                M 11
00 5A   E8  0521  1605                  BLBS    R10,LOG_MA              ; Branch to log the error.
                0524  1611  LOG_MA:
55   24 C0  0524  1612                  ADDL    #<9*4>,R5               ; Add # of bytes in this log to total.
     56 D6  0527  1613                  INCL    R6                      ; Increment count of memories logged.
                0529  1614  EXIT_MA:
     5B DD  0529  1615                  PUSHL   R11                     ; Restore caller's caller to stack.
     61 17  052B  1616                  JMP     (R1)                    ; Return to caller.
```

```
                        052D  1618          .SBTTL  TABLE OF RESUMABLE INSTRUCTIONS.
                        052D  1619 ;        EACH BIT IN THE TABLE IS A 1 IF THE INSTRUCTION IS RESUMABLE,
                        052D  1620 ;        AND A 0 IF IT IS NOT.
                        052D  1621
                        052D  1622 RESUMABLE:
3C3B                    052D  1623          .WORD   ^B0011110000111011      ;REI, LDPCTX, SVPCTX, INSQUE, REMQUE
                        052F  1624                                          ;CVTPS, CVTSP
FFFF                    052F  1625          .WORD   ^B1111111111111111
FF00                    0531  1626          .WORD   ^B1111111100000000      ;PACKED DECIMAL INSTRUCTIONS
FEFF                    0533  1627          .WORD   ^B1111111011111111      ;EDITPC
FFFF                    0535  1628          .WORD   ^B1111111111111111
002F                    0537  1629          .WORD   ^B0000000000101111      ;EMODF,CVTFD,INTERLOCKED INSTRUCTIONS
0F00                    0539  1630          .WORD   ^B0000111100000000      ;DOUBLE PRECISION FLOATING POINT
C14A                    053B  1631          .WORD   ^B1100000101001010      ;MORE DOUBLE PREC/QUAD, EMUL, EDIV
FFFF                    053D  1632          .WORD   ^B1111111111111111
FFFF                    053F  1633          .WORD   ^B1111111111111111
FFFF                    0541  1634          .WORD   ^B1111111111111111
F3FF                    0543  1635          .WORD   ^B1111001111111111      ;PUSHR, POPR
FFFF                    0545  1636          .WORD   ^B1111111111111111
F4FF                    0547  1637          .WORD   ^B1111010011111111      ;ADWC, SBWC, MFPR
FF3F                    0549  1638          .WORD   ^B1111111100111111      ;BBSSI, BBCCI
00FF                    054B  1639          .WORD   ^B0000000011111111      ;ASHP, CVTLP, CALLG, CALLS, XFC, EXPANSION
                        054D  1640
                        054D  1641          .end
```

```
AB_MEMERR                       00000010 R      07
AMPUTATE                        000000F2 R      08
BADTYPE                         0000004D R      08
BRESUM                          000001D2 R      08
BUG$_MPASYNCWRT                 ******** X      08
BUG$_MPBADMCK                   ******** X      08
BUG$_MPMCHECK                   ******** X      08
CACHEPARITY                     00000134 R      08
CH$S_CONTROL                =   00000004
CH$S_GOERRS                 =   00000007
CH$V_GOERRS                 =   00000003
CH$V_REPLG1                 =   0000000D
CHLOG_DISAB0                =   00000001
CHLOG_DISAB1                =   00000002
CH_MISSG0                   =   00010000
CH_MISSG1                   =   00008000
CH_REPAIR                   =   0021C000
CH_REPAIR_1                 =   0021A000
CH_REPLG0                   =   00004000
CH_REPLG1                   =   00002000
CH_THRESHOLD                =   0000000A
CPTIMEOUT                       000001D5 R      08
CRDINTMAX                   =   00000003
CRDWATCHMAX                 =   00000006
CSPARITY                        000000E4 R      08
DAMPUTATE                       000000C1 R      08
EMB$B_MC_SUMCOD             =   00000010
EMB$K_AW                    =   00000007
EMB$K_BE                    =   00000004
EMB$K_HE                    =   00000008
EMB$K_MC                    =   00000002
EMB$K_SA                    =   00000005
EMB$K_SE                    =   00000006
EMB$W_MC_ENTRY              =   00000004
ENAB_MA780                      00000460 R      08
ENAB_MS780C                     0000045E R      08
ENAB_MS780E                     0000045F R      08
ENAB_ROUTINES                   00000000 R      05
EXE$GL_CONFREGL                 ******** X      08
EXE$GL_FLAGS                    ******** X      08
EXE$GL_MCHKERRS                 ******** X      08
EXE$GL_MEMERRS                  ******** X      08
EXE$GL_NUMNEXUS                 ******** X      08
EXE$GL_TODR                     ******** X      08
EXE$GQ_SYSTIME                  ******** X      08
EXE$GQ_TODCBASE                 ******** X      08
EXE$MCHECK                      ******** X      08
EXE$V_CRDENABL                  ******** X      08
EXIT_MA                         00000529 R      08
GENERAL_MEMTYP              =   00000003
GL_BADTIMOUT                    0000002C R      07
GL_CH1OLD                       00000004 R      07
GL_CH2OLD                       00000008 R      07
GL_CHSTATE                      00000028 R      07
GL_CPTIMOUT                     0000000C R      07
GL_CRDCNT                       00000024 R      07
GL_CSBITA                       00000000 R      07
GW_REENAB                       00000020 R      07
GW_WATCH                        00000022 R      07
IBROMCHECK                      000000E4 R      08
INT54                           000003BC R      08
INT58                           000003A8 R      08
INT5C                           00000290 R      08
INT60                           000002A4 R      08
IPL$_SYNCH                  =   00000008
LOCATE_MEM                      00000431 R      08
LOGALL_ROUTINES                 00000000 R      04
LOGAWE                          000002A4 R      08
LOGC                            0000046D R      08
LOGCRD                          000003BC R      08
LOGE                            0000046E R      08
LOGERR_ROUTINES                 00000000 R      03
LOGGER                          00000247 R      08
LOGIT                           00000259 R      08
LOGMA                           000004AD R      08
LOGMEM                          000003E6 R      08
LOGSBA                          000003A8 R      08
LOGSBF                          00000290 R      08
LOGSBI                          000002DD R      08
LOG_ALL_MEM                     000003DD R      08
LOG_ERROR_MEM                   000003D2 R      08
LOG_MA                          00000524 R      08
LOG_MA780                       0000046F R      08
LOG_MS780C                      0000046D R      08
LOG_MS780E                      0000046E R      08
MCHK                            00000000 R      08
MCHK$GL_LOG                     0000026E RG     08
MCHK$M_LOG                  =   00000001
MCHK$M_MCK                  =   00000002
MCHK$M_NEXM                 =   00000004
MCHK_ERRCNT                     00000000 R      07
MCL_CES                     =   00000008
MCL_COUNT                   =   00000000
MCL_D                       =   00000014
MCL_PARITY                  =   00000024
MCL_PC                      =   0000002C
MCL_PSL                     =   00000030
MCL_SBIERR                  =   00000028
MCL_SUMMARY                 =   00000004
MCL_TBER0                   =   00000018
MCL_TBER1                   =   0000001C
MCL_TIMOADDR                =   00000020
MCL_UPC                     =   0000000C
MCL_VA                      =   00000010
MEMTYP                          00000000 R      02
MEMTYPCNT                   =   00000012
MMG$GL_SBICONF                  ******** X      08
MPS$AB_MEMERR                   00000010 RG     07
MPS$ALLOCEMB                    ******** X      08
MPS$GL_BADTIMOUT                0000002C RG     07
MPS$GL_CH1OLD                   00000004 RG     07
MPS$GL_CH2OLD                   00000008 RG     07
MPS$GL_CHSTATE                  00000028 RG     07
MPS$GL_CPTIMOUT                 0000000C RG     07
```

```
MPS$GL_CRDCNT                   00000024 RG    07
MPS$GL_CSBITA                   00000000 RG    07
MPS$GW_REENAB                   00000020 RG    07
MPS$GW_WATCH                    00000022 RG    07
MPS$INT54                       000003BC RG    08
MPS$INT58                       000003A8 RG    08
MPS$INT5C                       00000290 RG    08
MPS$INT60                       000002A4 RG    08
MPS$LOGAWE                      000002A4 RG    08
MPS$LOGCRD                      000003BC RG    08
MPS$LOGSBA                      000003A8 RG    08
MPS$LOGSBF                      00000290 RG    08
MPS$MCHK                        00000000 RG    08
MPS$MCHK_ERRCNT                 00000000 RG    07
MPS$MPSCHED2                    ********   X   08
MPS$REENABLE                    00000354 RG    08
MPS$RELEASEMB                   ********   X   08
MPS$SECBUGCHK                   ********   X   08
MPSWITCH                    =   00000001
MRC$M_CRDERR                =   00000200
MRC$M_CTLOPTY               =   00040000
MRC$M_CTL1PTY               =   00080000
MRC$M_ELSRF                 =   10000000
MRC$M_HERIMF                =   20000000
MRC$M_IFPTY                 =   00000100
MRC$M_INHBCRD               =   40000000
MRC$M_INVMAPPTY             =   80000000
MRC$M_MSEQPTY               =   00000080
MRC$M_SUMMARY               =   00100000
MRC$V_CRDERR                =   00000009
MRC$V_CTLOPTY               =   00000012
MRC$V_CTL1PTY               =   00000013
MRC$V_ELSRF                 =   0000001C
MRC$V_HERIMF                =   0000001D
MRC$V_IFPTY                 =   00000008
MRC$V_INHBCRD               =   0000001E
MRC$V_INVMAPPTY             =   0000001F
MRC$V_MSEQPTY               =   00000007
MRC$V_SUMMARY               =   00000014
NDT$_MEM16I                 =   00000011
NDT$_MEM16NI                =   00000010
NDT$_MEM256EIL              =   00000071
NDT$_MEM256EIU              =   00000073
NDT$_MEM256I                =   00000074
NDT$_MEM256NIL              =   00000070
NDT$_MEM256NIU              =   00000072
NDT$_MEM4I                  =   00000009
NDT$_MEM4NI                 =   00000008
NDT$_MEM64EIL               =   00000069
NDT$_MEM64EIU               =   0000006B
NDT$_MEM64I                 =   0000006C
NDT$_MEM64NIL               =   00000068
NDT$_MEM64NIU               =   0000006A
NDT$_MPM0                   =   00000040
NDT$_MPM1                   =   00000041
NDT$_MPM2                   =   00000042
NDT$_MPM3                   =   00000043
PR$_IPL                     =   00000012
PR$_KSP                     =   00000000
PR$_TBIA                    =   00000039
PR780$_SBIER                =   00000034
PR780$_SBIFS                =   00000030
PR780$_SBIMT                =   00000033
PR780$_SBIS                 =   00000031
PR780$_SBISC                =   00000032
PR780$_SBITA                =   00000035
PR780$_TODR                 =   0000001B
PSL$S_CURMOD                =   00000002
PSL$V_CURMOD                =   00000018
PSL$V_PRVMOD                =   00000016
READSOBST                       00000231 R     08
REENABLE_INTS                   00000373 R     08
REENABTIME                  =   00000384
REFLECTCHK                      00000106 R     08
RESUMABLE                       0000052D R     08
RESUME                          000000D3 R     08
SBIER$M_CPTO                =   00001000
SBIER$M_CRD                 =   00004000
SBIER$M_IBRDS               =   00000080
SBIER$M_IBTO                =   00000040
SBIER$M_RDS                 =   00002000
SBIFS$V_NEF                 =   00000019
SOMETIME                    =   0000003C
SS$_NORMAL                  =   00000001
TBUFPARITY                      000000AF R     08
TRYRESUME                       000000BB R     08
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
Y$MPDATA0	00000012 (18.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
Y$MPDATA2	0000000C (12.)	03 (3.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
Y$MPDATA3	0000000C (12.)	04 (4.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
Y$MPDATA4	0000000C (12.)	05 (5.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
Y$MPDATA1	00000012 (18.)	06 (6.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
$$$$MPDATA	00000030 (48.)	07 (7.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC QUAD
YY$MPCODE	0000054D (1357.)	08 (8.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC QUAD

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.09	00:00:01.04
Command processing	112	00:00:01.12	00:00:07.91
Pass 1	424	00:00:16.13	00:00:46.90
Symbol table sort	0	00:00:01.87	00:00:03.81
Pass 2	234	00:00:04.56	00:00:12.82
Symbol table output	23	00:00:00.20	00:00:00.72
Psect synopsis output	4	00:00:00.06	00:00:00.17
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	828	00:00:24.03	00:01:13.38

The working set limit was 1800 pages.
97324 bytes (191 pages) of virtual memory were used to buffer the intermediate code.
There were 60 pages of symbol table space allocated to hold 1187 non-local and 29 local symbols.
1647 source lines were read in Pass 1, producing 30 object records in Pass 2.
38 pages of virtual memory were used to define 33 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[MP.OBJ]MP.MLB;1	16
_255DUA28:[SYS.OBJ]LIB.MLB;1	18
_255DUA28:[SYSLIB]STARLET.MLB;2	8
TOTALS (all libraries)	42

1539 GETS were required to define 42 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:MPMCHECK/OBJ=OBJ$:MPMCHECK MSRC$:MPPREFIX/UPDATE=(ENH$:MPPREFIX)+MSRC$:MPSWT/UPDATE=(ENH$:MPSWT)+MASD$:[SYSLOA.SRC]MC
```

0248 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

MPERRLOG
LIS

MPSCBVEC
LIS

MPINT
LIS

MPPFM
LIS

MPDAT
LIS

MPPWRFAIL
LIS

MPMCHECK
LIS

MPINTEXC
LIS

MPLOG
LIS

MPERRMSG
LIS

MPSCHED
LIS

MPSHWPFM
LIS

MPLOAD
LIS

MPINIT
LIS