```
MMM        MMM   PPPPPPPPPPPP
MMM        MMM   PPPPPP PPPPP
MMM        MMM   PPPPPPPPPPPP
MMMMMM  MMMMMM   PPP         PPP
 MMMMM  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
```

```
MM      MM  PPPPPPPP   EEEEEEEEEE  RRRRRRRR   RRRRRRRR   LL            000000      GGGGGGGG
MM      MM  PPPPPPPP   EEEEEEEEEE  RRRRRRRR   RRRRRRRR   LL            000000      GGGGGGGG
MMMM  MMMM  PP      PP  EE          RR     RR  RR     RR  LL          00    00    GG
MMMM  MMMM  PP      PP  EE          RR     RR  RR     RR  LL          00    00    GG
MM  MM  MM  PP      PP  EE          RR     RR  RR     RR  LL          00    00    GG
MM  MM  MM  PP      PP  EE          RR     RR  RR     RR  LL          00    00    GG
MM      MM  PPPPPPPP   EEEEEEE      RRRRRRRR   RRRRRRRR   LL          00    00    GG
MM      MM  PPPPPPPP   EEEEEEE      RRRRRRRR   RRRRRRRR   LL          00    00    GG
MM      MM  PP          EE          RR  RR     RR  RR     LL          00    00    GG  GGGGGG
MM      MM  PP          EE          RR  RR     RR  RR     LL          00    00    GG  GGGGGG
MM      MM  PP          EE          RR    RR   RR    RR   LL          00    00    GG      GG   ....
MM      MM  PP          EEEEEEEEEE  RR     RR  RR     RR  LLLLLLLLLL   000000      GGGGGG      ....
MM      MM  PP          EEEEEEEEEE  RR     RR  RR     RR  LLLLLLLLLL   000000      GGGGGG      ....
```

```
LL            IIIIII    SSSSSSSS
LL            IIIIII    SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II            SS
LL              II            SS
LL              II            SS
LL              II            SS
LLLLLLLLL     IIIIII    SSSSSSSS
LLLLLLLLL     IIIIII    SSSSSSSS
```

```
0000    1 ;
0000    2 ; Version:       'V04-000'
0000    3 ;
0000    4
0000    5        .MCALL  MFPR
0000    1        .TITLE  MPERRLOG - MULTI-PROCESSING ERROR LOG SUPPORT ROUTINES
0000    2        .IDENT  'V04-000'
0000    3
0000    4 ;
0000    5 ;************************************************************************
0000    6 ;*                                                                      *
0000    7 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                             *
0000    8 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.              *
0000    9 ;*  ALL RIGHTS RESERVED.                                               *
0000   10 ;*                                                                      *
0000   11 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED *
0000   12 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE *
0000   13 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER *
0000   14 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY *
0000   15 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY *
0000   16 ;*  TRANSFERRED.                                                         *
0000   17 ;*                                                                      *
0000   18 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE *
0000   19 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT *
0000   20 ;*  CORPORATION.                                                         *
0000   21 ;*                                                                      *
0000   22 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS *
0000   23 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.              *
0000   24 ;*                                                                      *
0000   25 ;*                                                                      *
0000   26 ;************************************************************************
0000   27 ;
0000   28
0000   29 ;++
0000   30 ;
0000   31 ; Facility: Executive   Hardware fault handling
0000   32 ;
0000   33 ; Abstract:  Error logging routines used by secondary processor.
0000   34 ;
0000   35 ; Environment: MODE=Kernel
0000   36 ;
0000   37 ; Author:  Kathleen D. Morse, Creation date:  07-Jul-1981
0000   38 ;
0000   39 ; Modified by:
0000   40 ;
0000   41 ;        V03-003 KDM0066          Kathleen D. Morse       3-Aug-1983
0000   42 ;                Change PR$_TODR to PR780$_TODR.
0000   43 ;
0000   44 ;        V03-002 KDM0012          Kathleen D. Morse       20-Sep-1982
0000   45 ;                Add second error log buffer.
0000   46 ;
0000   47 ;    Version  ,
0000   48 ; 01      -
0000   49 ;--
```

MPERRLOG
VC4-000

E 2
– MULTI-PROCESSING ERROR LOG SUPPORT ROU 16-SEP-1984 02:00:51   VAX/VMS Macro V04-00   Page 2
                                          5-SEP-1984 02:06:17 [MP.SRC]MPERRLOG.MAR;1   (1)

```
0000      51 :
0000      52 : Macro Library Calls
0000      53 :
0000      54
0000      55     .      $EMBDEF <SU>                    ;Error log message offsets
0000      56            $MPSDEF                         ;Define secondary request bits
0000      57            $PRDEF                          ;Processor register definitions
0000      58            $PR780DEF                       ;11/780-specific IPR definitions
0000      59
0000      60 :
0000      61 : Equated Symbols
0000      62 :
0000      63
0000      64 :
0000      65 : Local Data
0000      66 :
0000      67
00000000  68            .PSECT   WIONONPAGED
```

```
                        0000      70              .SBTTL MPS$ALLOCEMB - Allocate secondary error message buffer
                        0000      71  ;+
                        0000      72  ; MPS$ALLOCEMB - Allocate secondary error message buffer
                        0000      73  ;
                        0000      74  ; This routine is called by the secondary processor to allocate
                        0000      75  ; the error log buffer.  There is currently only one buffer, which
                        0000      76  ; must be emptied by the primary before it can be re-used.
                        0000      77  ;
                        0000      78  ;
                        0000      79  ; INPUTS:
                        0000      80  ;
                        0000      81  ;       R1 - Size of message buffer required
                        0000      82  ;
                        0000      83  ; OUTPUTS:
                        0000      84  ;
                        0000      85  ;       R0 - Low bit clear if allocation failure
                        0000      86  ;          - Low bit set if successful allocation
                        0000      87  ;       R1 - Secondary error log sequence number
                        0000      88  ;       R2 - Address of error log buffer, if successful allocation
                        0000      89  ;
                        0000      90  ;-
                        0000      91
                        0000      92  MPS$ALLOCEMB::                                 ;Allocate secondary error message buffer
           0000'CF   D6 0000      93          INCL    W^MPS$GL_ERLSEQNUM             ;Record another error log attempt
   00000200 8F   51   D1 0004      94          CMPL    R1,#MPS$K_ERLBUFSIZ           ;Is secondary buffer large enough?
                48   1A 000B      95          BGTRU   40$                            ;Br if not, return error
   35 0000'CF   C0   E6 000D      96          BBSSI   #MPS$V_ERLBUF1,W^MPS$GL_ERLBUFIND,30$ ;Br if buffer busy
   52   0000'CF   9E 0013      97          MOVAB   W^MPS$AL_ERLBUF1,R2            ;Get address of error log buffer
      FC A2   51   B0 0018      98  10$:    MOVW    R1,EMB$W_SIZE(R2)             ;Set size of this error message
         51   0000'CF   D0 001C      99          MOVL    W^MPS$GL_ERLSEQNUM,R1         ;Secondary error log sequence # for msg
      0E A2   51   B0 0021     100          MOVW    R1,EMB$W_HD_ERRSEQ(R2)        ;Set secondary error log sequence #
06 A2   00000000'GF   7D 0025     101  20$:    MOVQ    G^EXE$GQ_SYSTIME,EMB$Q_HD_TIME(R2) ;Get system time
06 A2   00000000'GF   D1 002D     102          CMPL    G^EXE$GQ_SYSTIME,EMB$Q_HD_TIME(R2) ;Verify that the time
                EE   12 0035     103          BNEQ    20$                            ; acquired is valid and
0A A2   00000004'GF   D1 0037     104          CMPL    G^EXE$GQ_SYSTIME+4,EMB$Q_HD_TIME+4(R2) ; is not being modified
                E4   12 003F     105          BNEQ    20$                            ; by the primary.
                     0041     106          MFPR    #PR$_SID,EMB$L_HD_SID(R2)    ;Set system ID for this processor
         50   01   9A 0044     107          MOVZBL  #1,R0                         ;Indicate successful allocation
                05   0047     108          RSB                                    ;Return
07 0000'CF   01   E6 0048     109  30$:    BBSSI   #MPS$V_ERLBUF2,W^MPS$GL_ERLBUFIND,40$ ;Br if buffer busy
   52   0000'CF   9E 004E     110          MOVAB   W^MPS$AL_ERLBUF2,R2           ;Get address of error log buffer
            C3   11 0053     111          BRB     10$                            ;Join common code
         50   D4 0055     112  40$:    CLRL    R0                             ;Indicate failure to allocate buffer
              05 0057     113          RSB                                    ;Return
```

```
                    0058   115              .SBTTL MPS$RELEASEMB - Release secondary error message buffer
                    0058   116  ;+
                    0058   117  ; MPS$RELEASEMB - Release secondary error message buffer
                    0058   118  ;
                    0058   119  ; This routine is called by the secondary when the error log buffer
                    0058   120  ; contains a completed error message.  The primary processor is interrupted,
                    0058   121  ; with a request to enter the error log message for the secondary.
                    0058   122  ;
                    0058   123  ; INPUTS:
                    0058   124  ;
                    0058   125  ;        None
                    0058   126  ;
                    0058   127  ; OUTPUTS:
                    0058   128  ;
                    0058   129  ;        The primary processor is interrupted to log the secondary's message.
                    0058   130  ;
                    0058   131  ;-
                    0058   132
                    0058   133  MPS$RELEASEMB::                                  ;Release error message buffer
00 0000'CF   01  E6 0058   134              BBSSI   #MPS$V_SECERRLOG,W^MPS$GL_SECREQFLG,10$ ;Error log message ready
         FF9F'  30 005E   135  10$:         BSBW    W^MPS$INTPRIM                    ;Interrupt primary to log message
                05 0061   136              RSB                                      ;
```

```
                        0062  138           .SBTTL  MPS$COLDSTART/MPS$WARMSTART - Build secondary error log messages
                        0062  139 ;+
                        0062  140 ; MPS$COLDSTART - log coldstart (system boot)
                        0062  141 ;
                        0062  142 ; This routine is called by SYSINIT after correctly setting the system
                        0062  143 ; time to log the booting of the system.
                        0062  144 ;
                        0062  145 ; MPS$WARMSTART - Log warmstart (power recovery)
                        0062  146 ;
                        0062  147 ; This routine is called by powerfail after correcting the system time
                        0062  148 ; to log the power failure and recovery.
                        0062  149 ;
                        0062  150 ; INPUTS:
                        0062  151 ;
                        0062  152 ;       None
                        0062  153 ;
                        0062  154 ; OUTPUTS:
                        0062  155 ;
                        0062  156 ;       The error log buffer is allocated and filled with the appropriate
                        0062  157 ;       message if possible.
                        0062  158 ;-
                        0062  159           .ENABL  LSB
                        0062  160 MPS$COLDSTART::                              ;
          53   20   3C  0062  161           MOVZWL  #EMB$C_CS,R3             ;Set type of message to coldstart
               03   11  0065  162           BRB     10$                      ;
                        0067  163 MPS$WARMSTART·:                             ;
          53   24   3C  0067  164           MOVZWL  #EMB$C_WS,R3             ;Set type of message to warmstart
          51   14   3C  006A  165 10$:      MOVZWL  #EMB$C_SU_LENGTH,R1      ;Set size of message to allocate
               91   10  006D  166           BSBB    B^MPS$ALLOCEMB           ;Allocate secondary error log buffer
         4A 50      E9  006F  167           BLBC    R0,20$                   ;Br if not available
                        0072  168           MFPR    #PR780$_TODR,EMB$L_SU_DAYTIM(R2) ;Log time of day clock
    04 A2   53      B0  00B6  169           MOVW    R3,EMB$U_SU_ENTRY(R2)    ;Set message type
          9C      10     00BA  170           BSBB    B^MPS$RELEASEMB          ;Release buffer
               05        00BC  171 20$:      RSB                              ;
                        00BD  172           .DSABL  LSB
                        00BD  173           .END
```

```
EMB$C_CS                 = 00000020
EMB$C_SU_LENGTH          = 00000014
EMB$C_WS                 = 00000024
EMB$L_HD_SID             = 00000000
EMB$L_SU_DAYTIM          = 00000010
EMB$Q_HD_TIME            = 00000006
EMB$W_HD_ERRSEQ          = 0000000E
EMB$W_SIZE               = FFFFFFFC
EMB$W_SU_ENTRY           = 00000004
EXE$GL_TODR                ********   X   02
EXE$GQ_SYSTIME             ********   X   02
EXE$GQ_TODCBASE            ********   X   02
MP$$ALLOCEMB             00000000 RG     02
MP$$AL_ERLBUF1             ********   X   02
MP$$AL_ERLBUF2             ********   X   02
MP$$COLDSTART            00000062 RG     02
MP$$GL_ERLBUF_IND          ********   X   02
MP$$GL_ERLSEQNUM           ********   X   02
MP$$GL_SECREQFLG           ********   X   02
MP$$INTPRIM                ********   X   02
MP$$K_ERLBUFSIZ          = 00000200
MP$$RELEASEMB            00000058 RG     02
MP$$V_ERLBUF1            = 00000000
MP$$V_ERLBUF2            = 00000001
MP$$V_SECERRLOG          = 00000001
MP$$WARMSTART            00000067 RG     02
PR$_SID                  = 0000003E
PR780$_TODR              = 0000001B
```

```
                          +-----------------+
                          ! Psect synopsis !
                          +-----------------+
```

PSECT name	Allocation	PSECT No.	Attributes										
. ABS .	00000000 (0.)	00 (0.)	NOPIC	USR	CON	ABS	LCL	NOSHR	NOEXE	NORD	NOWRT	NOVEC	BYTE
ABS	00000000 (0.)	01 (1.)	NOPIC	USR	CON	ABS	LCL	NOSHR	EXE	RD	WRT	NOVEC	BYTE
WIONONPAGED	000000BD (189.)	02 (2.)	NOPIC	USR	CON	REL	LCL	NOSHR	EXE	RD	WRT	NOVEC	BYTE

```
                          +-------------------------+
                          ! Performance indicators !
                          +-------------------------+
```

Phase	Page faults	CPU Time	Elapsed Time
Initialization	32	00:00:00.09	00:00:00.59
Command processing	128	00:00:00.82	00:00:05.12
Pass 1	172	00:00:03.28	00:00:14.96
Symbol table sort	0	00:00:00.22	00:00:00.63
Pass 2	54	00:00:00.75	00:00:03.04
Symbol table output	5	00:00:00.04	00:00:00.04
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	395	00:00:05.22	00:00:24.40

The working set limit was 1200 pages.
14377 bytes (29 pages) of virtual memory were used to buffer the intermediate code.

There were 20 pages of symbol table space allocated to hold 220 non-local and 8 local symbols.
178 source lines were read in Pass 1, producing 13 object records in Pass 2.
15 pages of virtual memory were used to define 14 macros.

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

Macro library name   Macros defined
-----------------   --------------
_255DUA28:[MP.OBJ]MP.MLB;1   4
_255DUA28:[SYS.OBJ]LIB.MLB;1   4
_255DUA28:[SYSLIB]STARLET.MLB;2   5
TOTALS (all libraries)   13

367 GETS were required to define 13 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:MPERRLOG/OBJ=OBJ$:MPERRLOG MSRC$:MPPREFIX/UPDATE=(ENH$:MPPREFIX)+MSRC$:MPERRLOG/UPDATE=(ENH$:MPERRLOG)+EXECML$/LIB+LI

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