```
MMM          MMM  PPPPPPPPPPPP
MMM          MMM  PPPPPPPPPPPP
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    LL          OOOOOO      GGGGGGGG
MM      MM  PPPPPPPP    LL          OOOOOO      GGGGGGGG
MMMM  MMMM  PP      PP  LL        OO      OO  GG
MMMM  MMMM  PP      PP  LL        OO      OO  GG
MM  MM  MM  PP      PP  LL        OO      OO  GG
MM  MM  MM  PP      PP  LL        OO      OO  GG
MM      MM  PPPPPPPP    LL        OO      OO  GG
MM      MM  PPPPPPPP    LL        OO      OO  GG
MM      MM  PP          LL        OO      OO  GG  GGGGGG
MM      MM  PP          LL        OO      OO  GG  GGGGGG
MM      MM  PP          LL        OO      OO  GG      GG      ....
MM      MM  PP          LL        OO      OO  GG      GG      ....
MM      MM  PP          LLLLLLLLLL    OOOOOO      GGGGGG      ....
MM      MM  PP          LLLLLLLLLL    OOOOOO      GGGGGG      ....


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

Table of contents

(1) 74 MPS$LOGFOUND - Log scheduling decision result
(1) 134 MPS$LOGCHOOSE - Log scheduling decision making data

```
0000    ;
0000    ; Version:      'V04-000'
0000    ;
0000
0000            .MCALL  MFPR
0000            .TITLE  MPLOG - Multi-processor event logger
0000            .IDENT  'V04-000'
0000
0000    ;
0000    ;*******************************************************************************
0000    ;*                                                                             *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                    *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                     *
0000    ;*  ALL RIGHTS RESERVED.                                                       *
0000    ;*                                                                             *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED      *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE     *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER      *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY      *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY      *
0000    ;*  TRANSFERRED.                                                               *
0000    ;*                                                                             *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE      *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
0000    ;*  CORPORATION.                                                               *
0000    ;*                                                                             *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS      *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                    *
0000    ;*                                                                             *
0000    ;*                                                                             *
0000    ;*******************************************************************************
0000    ;
0000
0000    ;++
0000    ;
0000    ; Facility:  Executive , Hardware fault handling
0000    ;
0000    ; Abstract:  This module contains an event logger that may be called
0000    ;                to log scheduling information.
0000    ;
0000    ; Environment: MODE=Kernel
0000    ;
0000    ; Author:  Kathleen D. Morse, Creation date:  08-Apr-1981
0000    ;
0000    ; Modified by:
0000    ;
0000    ;   Version  ,
0000    ; 01    -
0000    ;--
0000
0000    ;
0000    ; Include files:
0000    ;
0000            $PCBDEF                         ; Process control block
0000            $PHDDEF                         ; Process header
0000
0000    ;
```

```
                0000    ; MACROS:
                0000    ;
                0000
                0000    ;
                0000    ; Equated Symbols:
                0000    ;
00000005        0000    ONE_ENTRY = 5                           ; # of longwords in one log entry
                0000
                0000    ;
                0000    ; Data Area
                0000    ;
        00000000        .PSECT  LOGGER,LONG                     ; Event logger
                0000    MPS$GL_LOGIND::                         ;
00000000        0000            .LONG   0                       ; Indicator whether or not to log info
                0004    MPS$GL_LOGPTR::                         ;
00000000        0004            .LONG   0                       ; Pointer to next free byte in log bufr
                0008    MPS$GL_LOG::                            ;
000007D8        0008            .BLKL   <ONE_ENTRY*100>         ; Log buffer
                07D8    MPS$GL_LOGEND::                         ; End of log buffer
                07D8            .LIST   MEB                     ; Show macro expansions
```

```
                        07D8          .SBTTL  MPS$LOGFOUND - Log scheduling decision result
                        07D8    ;++
                        07D8    ; Functional Description:
                        07D8    ;
                        07D8    ; This module is used to log scheduling information as processes as
                        07D8    ; scheduled for the secondary.  It is a debugging tool for the
                        07D8    ; multi-processing code.
                        07D8    ;
                        07D8    ; Calling Sequence:
                        07D8    ;
                        07D8    ;       BSBW    MPS$LOGFOUND
                        07D8    ;
                        07D8    ; Input Parameters:
                        07D8    ;
                        07D8    ;       None
                        07D8    ;
                        07D8    ; Output Parameters:
                        07D8    ;
                        07D8    ;       None
                        07D8    ;
                        07D8    ; Implicit Inputs:
                        07D8    ;
                        07D8    ;       MPS$GL_CURPCB - address of current process on secondary
                        07D8    ;       MPS$GL_LOGIND - log indicator, LBS = log info, LBC = don't log info
                        07D8    ;       MPS$GL_LOGPTR - pointer to next free byte in log buffer
                        07D8    ;       MPS$GL_LOG - beginning of log buffer
                        07D8    ;       MPS$GL_LOGEND - end of log buffer
                        07D8    ;
                        07D8    ; Implicit Outputs:
                        07D8    ;
                        07D8    ;       Logging data may be entered into the log buffer.
                        07D8    ;
                        07D8    ;--
                        07D8
                        07D8    MPS$LOGFOUND::                                  ;
3E F823 CF    00   E1   07D8          BBC     #0,W^MPS$GL_LOGIND,20$          ; Is logging requested?
              0F   BB   07DE          PUSHR   #^M<R0,R1,R2,R3>                ; Save registers
   53   FFF4 CF   9E   07E0          MOVAB   W^MPS$GL_LOGEND,R3              ; Adr of end of log buffer
   50   F81B CF   D0   07E5          MOVL    W^MPS$GL_LOGPTR,R0              ; Adr of next free byte in log buffer
     51     14 A0   9E   07EA          MOVAB   <ONE_ENTRY*4>(R0),R1            ; Adr of next byte past this log entry
        53      51   D1   07EE          CMPL    R1,R3                           ; Entry to large to fit in buffer?
              2A   1E   07F1          BGEQU   30$                             ; Br on yes, go turn off event logging
   51   0000'CF   D0   07F3          MOVL    W^MPS$GL_CURPCB,R1              ; Adr of current PCB on secondary
     52     6C A1   D0   07F8          MOVL    PCB$L_PHD(R1),R2                ; Adr of current PHD on secondary
80   00000000'GF   D0   07FC          MOVL    G^SCH$GL_COMQS,(R0)+            ; Log compute queue states
     80     60 A1   D0   0803          MOVL    PCB$L_PID(R1),(R0)+             ; Log PID of current process
   80   00C0 C2   D0   0807          MOVL    PHD$L_PC(R2),(R0)+              ; Log PC of current process
   80   00C4 C2   D0   080C          MOVL    PHD$L_PSL(R2),(R0)+             ; Log PSL of current process
     80     0B A1   9A   0811          MOVZBL  PCB$B_PRI(R1),(R0)+             ; Log priority of current process
   F7EA CF   50   D0   0815          MOVL    R0,W^MPS$GL_LOGPTR              ; Update pointer to next free byte
              0F   BA   081A    10$:  POPR    #^M<R0,R1,R2,R3>                ; Restore registers
              05   081C    20$:  RSB                                           ; Return to caller
                        081D
                        081D    ;
                        081D    ; Turn off event logging as buffer is full.
                        081D    ;
00 F7DE CF    00   E7   081D    30$:  BBCCI   #0,W^MPS$GL_LOGIND,40$          ; Turn off event logging, buffer full
```

```
F5   11  0823   131 40$:    BRB     10$                     ; Continue with common code
         0825   132         .LIST   MEB                     ; Show macro expansions
```

```
                                  0825    134          .SBTTL  MPS$LOGCHOOSE - Log scheduling decision making data
                                  0825    135 ;++
                                  0825    136 ; Functional Description:
                                  0825    137 ;
                                  0825    138 ; This module is used to log scheduling information as processes as
                                  0825    139 ; examined for scheduling on the secondary.  It is a debugging tool for the
                                  0825    140 ; multi-processing code.
                                  0825    141 ;
                                  0825    142 ; Calling Sequence:
                                  0825    143 ;
                                  0825    144 ;        BSBW    MPS$LOGCHOOSE
                                  0825    145 ;
                                  0825    146 ; Input Parameters:
                                  0825    147 ;
                                  0825    148 ;        R4 - PCB address of scheduling candidate
                                  0825    149 ;        R5 - PHD address of scheduling candidate
                                  0825    150 ;
                                  0825    151 ; Output Parameters:
                                  0825    152 ;
                                  0825    153 ;        None
                                  0825    154 ;
                                  0825    155 ; Implicit Inputs:
                                  0825    156 ;
                                  0825    157 ;        MPS$GL_LOGIND - log indicator, LBS = log info, LBC = don't log info
                                  0825    158 ;        MPS$GL_LOGPTR - pointer to next free byte in log buffer
                                  0825    159 ;        MPS$GL_LOG - beginning of log buffer
                                  0825    160 ;        MPS$GL_LOGEND - end of log buffer
                                  0825    161 ;
                                  0825    162 ; Implicit Outputs:
                                  0825    163 ;
                                  0825    164 ;        Logging data may be entered into the log buffer.
                                  0825    165 ;
                                  0825    166 ;--
                                  0825    167
                                  0825    168 MPS$LOGCHOOSE::                     ;
38 F7D6 CF    00    E1            0825    169          BBC     #0,W^MPS$GL_LOGIND,20$  ; Is logging requested?
              0F    BB            082B    170          PUSHR   #^M<R0,R1,R2,R3>        ; Save registers
   53    FFA7 CF    9E            082D    171          MOVAB   W^MPS$GL_LOGEND,R3      ; Adr of end of log buffer
   50    F7CE CF    D0            0832    172          MOVL    W^MPS$GL_LOGPTR,R0      ; Adr of next free byte in log buffer
   51      14 A0    9E            0837    173          MOVAB   <ONE_ENTRY*4>(R0),R1    ; Adr of next byte past this log entry
   53         51    D1            083B    174          CMPL    R1,R3                   ; Entry to large to fit in buffer?
              24    1E            083E    175          BGEQU   30$                     ; Br on yes, go turn off event logging
80   00000000'GF    D0            0840    176          MOVL    G^SCH$GL_COMQS,(R0)+    ; Log compute queue states
   80      60 A4    D0            0847    177          MOVL    PCB$L_PID(R4),(R0)+     ; Log PID of current process
   80    00C0 C5    D0            084B    178          MOVL    PHD$L_PC(R5),(R0)+      ; Log PC of current process
   80    00C4 C5    D0            0850    179          MOVL    PHD$L_PSL(R5),(R0)+     ; Log PSL of current process
   80      0B A4    9B            0855    180          MOVZBW  PCB$B_PRI(R4),(R0)+     ; Log priority of current process
   80         01    9B            0859    181          MOVZBW  #1,(R0)+                ; Log indicator for decision process
   F7A3 CF    50    D0            085C    182          MOVL    R0,W^MPS$GL_LOGPTR      ; Update pointer to next free byte
              0F    BA            0861    183 10$:     POPR    #^M<R0,R1,R2,R3>        ; Restore registers
              05    11            0863    184 20$:     RSB                             ; Return to caller
                                  0864    185
                                  0864    186 ;
                                  0864    187 ; Turn off event logging as buffer is full.
                                  0864    188 ;
00 F797 CF    00    E7            0864    189 30$:     BBCCI   #0,W^MPS$GL_LOGIND,40$  ; Turn off event logging, buffer full
              F5    11            086A    190 40$:     BRB     10$                     ; Continue with common code
```

```
086C   191          .END
```

```
MPS$GL_CURPCB          ********   X   02
MPS$GL_LOG             00000008 RG    02
MPS$GL_LOGEND          000007D8 RG    02
MPS$GL_LOGIND          00000000 RG    02
MPS$GL_LOGPTR          00000004 RG    02
MPS$LOGCHOOSE          00000825 RG    02
MPS$LOGFOUND           000007D8 RG    02
ONE_ENTRY            = 00000005
PCB$B_PRI            = 0000000B
PCB$L_PHD            = 0000006C
PCB$L_PID            = 00000060
PHD$L_PC             = 000000C0
PHD$L_PSL            = 000000C4
SCH$GL_COMQS           ********   X   02
```

+----------+
! 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
LOGGER	0000086C	(2156.)	02 (2.)	NOPIC	USR	CON	REL	LCL	NOSHR	EXE	RD	WRT	NOVEC	LONG

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

Phase	Page faults	CPU Time	Elapsed Time
Initialization	35	00:00:00.09	00:00:00.56
Command processing	152	00:00:00.95	00:00:04.00
Pass 1	176	00:00:03.12	00:00:09.10
Symbol table sort	0	00:00:00.35	00:00:00.41
Pass 2	52	00:00:00.86	00:00:03.20
Symbol table output	3	00:00:00.04	00:00:00.06
Psect synopsis output	2	00:00:00.03	00:00:00.08
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	422	00:00:05.44	00:00:17.41

The working set limit was 1200 pages.
14682 bytes (29 pages) of virtual memory were used to buffer the intermediate code.
There were 20 pages of symbol table space allocated to hold 260 non-local and 8 local symbols.
196 source lines were read in Pass 1, producing 13 object records in Pass 2.
10 pages of virtual memory were used to define 9 macros.

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

Macro library name                       Macros defined
------------------                       --------------
_$255$DUA28:[MP.OBJ]MP.MLB;1                      1
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                    2
_$255$DUA28:[SYSLIB]STARLET.MLB;2                 3
TOTALS (all libraries)                            6

336 GETS were required to define 6 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:MPLOG/OBJ=OBJ$:MPLOG MSRC$:MPPREFIX/UPDATE=(ENH$:MPPREFIX)+MSRC$:MPLOG/UPDATE=(ENH$:MPLOG)+EXECML$/LIB+LIB$:MP.MLB/LI
```

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

MPSWPFM
LIS

MPLOAD
LIS

MPINIT
LIS