

[illegible]

[illegible]

MPTIMER  
Table of contents

- MP TIMER INTERRUPT HANDLER

N 3

16-SEP-1984 02:07:43 VAX/VMS Macro V04-00

Page 0

(1) 77  
(1) 148

HARDWARE CLOCK INTERRUPTS  
SOFTWARE TIMER INTERRUPTS

MPW  
V04



```
0000 1 :
0000 2 : Version: 'V04-000'
0000 3 :
0000 4 :
0000 5 : .MCALL MFPR
0000 1 : .TITLE MPTIMER - MP TIMER INTERRUPT HANDLER
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: Timer interrupt handler for secondary processor
0000 34 :
0000 35 : Environment: MODE=Kernel
0000 36 :
0000 37 : Author: RICHARD I. HUSTVEDT, Creation date: 15-MAY-1979
0000 38 :
0000 39 : Modified by:
0000 40 :
0000 41 : V03-008 KDM0094 Kathleen D. Morse 22-Mar-1984
0000 42 : Do not reschedule a real-time process that goes through
0000 43 : quantum end on the secondary processor.
0000 44 :
0000 45 : V03-007 ROW0151 Ralph O. Weber 29-DEC-1982
0000 46 : Fix SOFTINT in MPSSHCLKINT to use IPL$TIMERFORK and
0000 47 : MPSS$WTIMINT to begin by raising IPL to IPL$TIMER. This
0000 48 : change is required to match similar changes on the primary.
0000 49 : The net effect of these changes is to raise IPL$SYNCH to 8.
0000 50 :
0000 51 : V03-006 KDM0025 Kathleen D. Morse 10-Oct-1982
0000 52 : Change secondary specific routines to be MPSS$xxx instead
```

```
0000 53 : of EXE$xxx (MP$SWTIMINT, MP$HWCLKINT).
0000 54 :
0000 55 : V03-005 KDM0010 Kathleen D. Morse 31-Aug-1982
0000 56 : Lower IPL to SYNCH before rescheduling for quantum end.
0000 57 :
0000 58 : V03-004 KDM0009 Kathleen D. Morse 31-Aug-1982
0000 59 : Do an interlocked operation before testing MP$GL_STATE.
0000 60 :
0000 61 :--
0000 62 :
0000 63 :
0000 64 : MACRO LIBRARY CALLS
0000 65 :
0000 66 :
0000 67 : $CADEF ; Define conditional assembly parameters
0000 68 : $IPLDEF ; Define interrupt priority levels
0000 69 : $LCKDEF ; Interlock bit definitions
0000 70 : $MPSDEF ; Secondary processor state definitions
0000 71 : $PCBDEF ; Define PCB offsets
0000 72 : $PHDDEF ; Define PHD offsets
0000 73 : $PRDEF ; Define processor registers
0000 74 : $PRIDEF ; Define priority increments
0000 75 : $PSLDEF ; Define processor status fields
```



```
0000 77 .SBTTL  HARDWARE CLOCK INTERRUPTS
0000 78 :+
0000 79 : EX$HWCLKINT - HARDWARE CLOCK INTERRUPT
0000 80 :
0000 81 : This routine is automatically vectored to when the clock count register
0000 82 : overflows. The accounting integral of the current process is incremented,
0000 83 : and the quantum of the current process is incremented. If the process quantum
0000 84 : transists to zero then a software interrupt is requested on the timer level.
0000 85 : The clock interrupt is then dismissed.
0000 86 :
0000 87 : INPUTS:
0000 88 :
0000 89 :     (SP) - PC at time of interrupt
0000 90 :     4(SP) - PSL at time of interrupt
0000 91 :
0000 92 : ENVIRONMENT:
0000 93 :
0000 94 :     Executes on secondary processor.
0000 95 :     The routine executes at IPL ^X18.
0000 96 :
0000 97 :-
0000 98
00000000 99 .PSECT  A$EXENONPAGED, LONG
18 7E 50 7D 0000 100 MP$HWCLKINT:: : Hardware clock interrupt routine
      800000C1 8F DA 0003 101      MOVQ  R0, -(SP) : Save register R0 and R1
      00000002 000A 102      MTPR  #^X800000C1, #PRS_ICCS : Clr err & intrpt and re-enable
50 0F AE 98 000A 103
      00000000 000A 104      .IF NE  CAS$ MEASURE
      0F AE 98 000A 105      CVTBL  15(SP), R0 : Get upper byte of saved PSL
50 0F 69 19 000E 106      BLSS   70$ : Br if CM bit set
      F8 8F 8A 0010 107 10$: BICB   #^XF8, R0 : Convert extended byte to index
      0000'CF40 D6 0014 108      INCL   W^MP$SAL_CPUTIME[R0] : Increment statistics vector
      0019 109      .ENDC
      0019 110
      0019 111      .IF      DF, MPPFMSWT
      0019 112      BSBW   MP$SPFM_RUNTIME : Gather performance measurement data
      0019 113      .ENDC
      0019 114
00 0000'CF 00 E6 0019 115      BBSSI  #LCK$V_INTERLOCK, W^MP$SGL_INTERLOCK, 15$ : Flush cache queue
01 0000'CF 00 D1 001F 116 15$: CMPL   W^MP$SGL_STATE, #MP$SK_IDLESTATE : Is a process actually running?
      0000'CF 3F D1 0024 117      BEQL   60$ : Br on no, go accumulate null time
04 0000'CF 11 D1 0026 118      CMPL   W^MP$SGL_STATE, #MP$SK_EXECSTATE : Is a process actually running?
      0000'CF 11 D1 002B 119      BNEQ   20$ : Br on no, don't accumulate time in PHD
50 0000'CF D0 002D 120      MOVL   W^MP$SGL_CURPCB, R0 : Get address of current process PCB
51 6C A0 D0 0032 121      MOVL   PCB$P_PHD(R0), R1 : Get address of process header
      38 A1 D6 0036 122      INCL   PHD$P_CPUTIM(R1) : Increment accounting integral
      3C A1 B6 0039 123      INCW   PHD$P_QUANT(R1) : Increment time quantum
      04 18 003C 124      BGEQ   30$ : If EQL quantum ran out
      50 8E 7D 003E 125 20$: MOVQ   (SP)+, R0 : Restore R0 and R1
      02 0041 126      REI
      0042 127
      10 0B A0 91 0042 128 30$: CMPB   PCB$P_PRI(R0), #16 : Is this a real-time process?
      05 19 0046 129      BLSS   40$ : Br if yes, it is real-time
      0048 130      SOFTINT #IPL$_TIMERFORK : Request software timer fork int
      004B 131      BRB    20$ : Join common code
00 24 A0 03 E5 004D 132 40$: BBCC   #PCB$V_INQUAN, PCB$P_STS(R0), 50$ : Clear initial quan flag
3C A1 00000000'GF B0 0052 133 50$: MOVW   G^SCH$GW_QUAN, PHD$P_QUANT(R1) : Set new quantum
```

```
0118 C0 00000000'GF D0 005A 134 MOVL G^EXESGL_ABSTIM,PCBSL_WAITIME(R0) ; Record event time
          D9 11 0063 135 BRB 20$ ; Join common code
          0065 136
          0000'CF D6 0065 137 60$: INCL W^MPSSGL_NULLCPU ; Accumulate null cpu time for secondary
50 00000000'GF 9E 0069 138 MOVAB G^SCH$GL_NULLPCB,R0 ; Get address of PCB for null process
    50 6C A0 D0 0070 139 MOVL PCBSL_PHD(R0),R0 ; Get address of process header
      38 A0 D6 0074 140 INCL PHD$SL_CPUTIM(R0) ; Increment accounting integral
      C5 11 0077 141 BRB 20$ ; Join common code
          0079 142
          00000002 0079 143
50 05 9A 0079 144 70$: .IF NE CAS MEASURE
      92 11 007C 145 MOVZBL #5,R0 ; Insert CM index
          007E 146 BRB 10$ ; Joint common code
          .ENDC
```



```
007E 148 .SBTTL SOFTWARE TIMER INTERRUPTS
007E 149 :+
007E 150 : EXESSWTIMINT - SOFTWARE TIMER INTERRUPTS
007E 151 :
007E 152 : This routine is automatically vectored to when a software interrupt
007E 153 : is requested on the timer level. Timer interrupts are requested when
007E 154 : the current process has exceeded its cpu time quantum.
007E 155 :
007E 156 : A kernel mode PSL and a PC pointing to MPSSQEND are placed on the kernel
007E 157 : stack so that the primary will continue executing quantum end for this
007E 158 : process.
007E 159 :
007E 160 : INPUTS:
007E 161 :
007E 162 : (SP) - PC at time of interrupt
007E 163 : 4(SP) - PSL at time of interrupt
007E 164 :
007E 165 : ENVIRONMENT:
007E 166 :
007E 167 : Executes on secondary processor.
007E 168 :
007E 169 : This routine is entered at IPL$TIMERFORK. It immediately raises
007E 170 : to IPL$TIMER (which equals IPL$_SYNCH) and completes execution at
007E 171 : IPL$_SYNCH.
007E 172 :
007E 173 :-
007E 174 :
007E 175 .ENABL LSB
007E 176 .ALIGN LONG
0080 177 ASSUME IPL$_SYNCH EQ IPL$TIMER
0080 178
0080 179 MPSS$WTIMINT::
0080 180 SETIPL #IPL$TIMER ; Software timer interrupt routine
0083 181 BITB #<<PS[$M_CURMOD> - ; Raise to normal IPL.
0087 182 @-<PSL$V_CURMOD>> - ; Check if this interrupt
0087 183 4+<PSL$V_CURMOD@-3>(SP) ; occurred during
0087 184 BEQL 20$ ; kernel mode execution
0089 185 PUSHL R0 ; Br if kernel mode, forget interrupt
008B 186 MOVL W*MPSS$GL_CURPCB,R0 ; Save register
0090 187 MOVL PCB$P_PHD(R0),R0 ; A second check for quantum must be
0094 188 SETIPL #IPL$POWER ; done here in case the process found
0097 189 TSTL W*MPSS$GL_PFAILTIM ; Prevent powerfail window here
009B 190 BNEQ 5$ ; Check if powerfail occurred
009D 191 TSTW PHD$W_QUANT(R0) ; Br if it did, always give process back
00A0 192 BLSS 30$ ; by the hardware clock routine was
00A2 193 5$: MOVL (SP),R0 ; rescheduled before this interrupt.
00A5 194 CLRL (SP) ; Restore register
00A7 195 ; Make PSL look like kernel mode
00A7 196 .IF DF,MPPFMSWT
00A7 197 BSBW MPSS$PFM_QEND ; Gather performance measurement data
00A7 198 .ENDC
00A7 199
00A7 200 SETIPL #IPL$ SYNCH ; Lower to SYNCH before scheduling
00AA 201 BSBW MPSS$MPSCHED2 ; Go return process to primary
00AD 202 ; and make primary start execution at
00AD 203 ; MPSSQEND, which must follow this line
00AD 204
```

07 AE 03 93  
50 0000'CF 50 DD  
50 6C A0 D0  
0000'CF D5  
05 12  
3C A0 B5  
24 19  
50 6E D0  
6E D4  
FF53' 30



```
00AD 205
00AD 206
00AD 207 :+
00AD 208 : MPSSQEND - QUANTUM END PERFORMED BY PRIMARY ON BEHALF OF SECONDARY
00AD 209 :
00AD 210 : This routine is the place that the primary starts executing when the
00AD 211 : secondary passes a process to it that has just finished its quantum.
00AD 212 :
00AD 213 : ENVIRONMENT:
00AD 214 :
00AD 215 : Executed by primary processor.
00AD 216 :
00AD 217 :-
00AD 218
00AD 219 MPSSQEND:
00AD 220 SETIPL #IPL$ SYNCH ; Synch scheduler with other reporting
54 00000000'GF 3F BB 00B0 221 PUSHR #^M<R0,R1,R2,R3,R4,R5> ; Save registers R0 through R5
55 6C A4 D0 00B2 222 MOVL G^SCH$GL CURPCB,R4 ; Get current process PCB address
00000000'GF 16 00B9 223 MOVL PCB$L PHD(R4),R5 ; Get address of process header
3F BA 00C3 224 JSB G^SCH$QEND ; Perform normal quantum end
02 00C5 225 10$: POPR #^M<R0,R1,R2,R3,R4,R5> ; Restore registers R0 thru R5
00C6 226 20$: REI ; Return from interrupt
50 8E D0 00C6 227 30$: MOVL (SP)+,R0 ; Restore register
02 00C9 228 REI
00CA 229
00CA 230
00CA 231
00CA 232 .END
```

MPTIMER  
Symbol table

- MP TIMER INTERRUPT HANDLER

H 4

16-SEP-1984 02:07:43 VAX/VMS Macro V04-00  
5-SEP-1984 02:07:38 [MP.SRC]MPTIMER.MAR;1

Page 7  
(1)

```

CAS_MEASURE      = 00000002
EXESGL_ABSTIM    ***** X 02
IPLS_POWER      = 0000001F
IPLS_SYNCH      = 00000008
IPLS_TIMER      = 00000008
IPLS_TIMERFORK  = 00000007
LCK$V_INTERLOCK = 00000000
MPSSAC_CPUTIME   ***** X 02
MPSSGL_CURPCB    ***** X 02
MPSSGL_INTERLOCK ***** X 02
MPSSGL_NULLCPU   ***** X 02
MPSSGL_PFAILTIM  ***** X 02
MPSSGL_STATE     ***** X 02
MPSSHWCLKINT     = 00000000 RG 02
MPSSK_EXECSTATE  = 00000004
MPSSK_IDLESTATE  = 00000001
MPSSMPSCHED2     ***** X 02
MPSSQEND         = 000000AD R 02
MPSSSWTIMINT     = 00000080 RG 02
PCBSB_PRI        = 0000000B
PCBSL_PHD        = 0000006C
PCBSL_STS        = 00000024
PCBSL_WAITIME    = 00000118
PCBSV_INQUAN     = 00000003
PHDSL_CPUTIM     = 00000038
PHDSW_QUANT      = 0000003C
PRS_ICCS         = 00000018
PRS_IPL          = 00000012
PRS_SIRR         = 00000014
PSLSM_CURMOD     = 03000000
PSLSV_CURMOD     = 00000018
SCH$GL_CURPCB    ***** X 02
SCH$GL_NULLPCB   ***** X 02
SCH$GW_QUAN      ***** X 02
SCH$QEND         ***** 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
\$AB\$\$	00000000 ( 0.)	01 ( 1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
\$EXENONPAGED	000000CA ( 202.)	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	00:00:00.38
Command processing	134	00:00:00.82	00:00:05.07
Pass 1	211	00:00:05.19	00:00:14.50
Symbol table sort	0	00:00:00.63	00:00:00.77
Pass 2	61	00:00:01.13	00:00:04.11



Symbol table output	7	00:00:00.06	00:00:00.14
Psect synopsis output	1	00:00:00.02	00:00:00.05
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	451	00:00:07.94	00:00:25.04

The working set limit was 1350 pages.

25383 bytes (50 pages) of virtual memory were used to buffer the intermediate code.

There were 30 pages of symbol table space allocated to hold 429 non-local and 12 local symbols.

237 source lines were read in Pass 1, producing 13 object records in Pass 2.

20 pages of virtual memory were used to define 19 macros.

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

Macro library name	Macros defined
-----	-----
\$255\$DUA28:[MP.OBJ]MP.MLB;1	3
\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	7
\$255\$DUA28:[SYSLIB]STARLET.MLB;2	6
TOTALS (all libraries)	16

555 GETS were required to define 16 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LISS:MPTIMER/OBJ=OBJ\$:MPTIMER MSRC\$:MPPREFIX/UPDATE=(ENH\$:MPPREFIX)+MSRC\$:MPTIMER/UPDATE=(ENH\$:MPTIMER)+EXECMLS/LIB+LIB\$:M



0249

AH-BT13A-SE  
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY