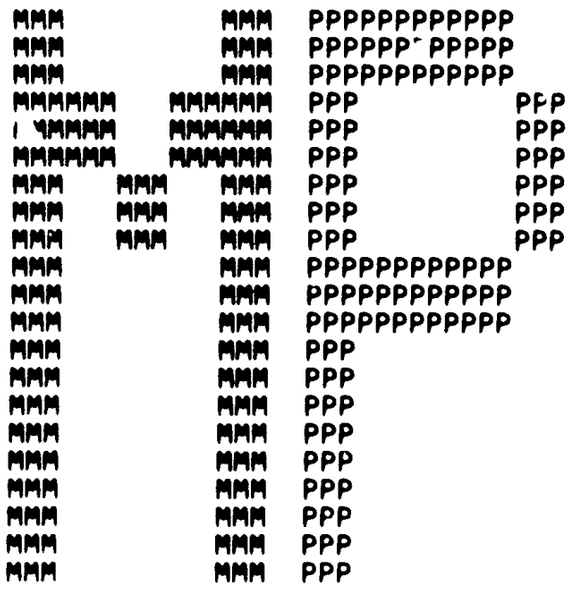
MMM MMM PPPPPPPPPPPP
MMM MMM PPPPPP PPPPP
MMM MMM PPPPPPPPPPPP
MMMMMM MMMMMM PPP PPP
MMMMMM MMMMMM PPP PPP
MMMMMM MMMMMM PPP PPP
MMM MMM MMM PPP PPP
MMM MMM MMM PPP PPP
MMM MMM MMM PPP PPP
MMM MMM PPPPPPPPPPPP
MMM MMM PPPPPPPPPPPP
MMM MMM PPPPPPPPPPPP
MMM MMM PPP
MMM MMM PPP
MMM MMM PPP
MMM MMM PPP
MMM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

FILEID**MPPFM


```
MM      MM  PPPPPPPP    PPPPPPPP    FFFFFFFFFF  MM      MM
MM      MM  PPPPPPPPP   PPPPPPPPP   FFFFFFFFFF  MM      MM
MMMM  MMMM  PP      PP  PP      PP  FF          MMMM  MMMM
MMMM  MMMM  PP      PP  PP      PP  FF          MMMM  MMMM
MM  MM  MM  PP      PP  PP      PP  FF          MM  MM  MM
MM  MM  MM  PP      PP  PP      PP  FF          MM  MM  MM
MM      MM  PPPPPPPPP   PPPPPPPPP   FFFFFFFF    MM      MM
MM      MM  PPPPPPPP    PPPPPPPP    FFFFFFFF    MM      MM
MM      MM  PP          PP          FF          MM      MM
MM      MM  PP          PP          FF          MM      MM
MM      MM  PP          PP          FF          MM      MM      ....
MM      MM  PP          PP          FF          MM      MM      ....
MM      MM  PP          PP          FF          MM      MM      ....
MM      MM  PP          PP          FF          MM      MM      ....

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
0000     1         .TITLE  MPPFM - Multi-processing Performance Measurement 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, Multi-processing performance measuring
0000    32 ;
0000    33 ; Abstract:  This module contains performance measurement routines
0000    34 ;               for gathering data on context switches.
0000    35 ;
0000    36 ; Environment: MODE=Kernel
0000    37 ;
0000    38 ; Author:  Kathleen D. Morse, Creation date:  05-Aug-1981
0000    39 ;
0000    40 ; Modified by:
0000    41 ;
0000    42 ;       V03-005 KDM0066         Kathleen D. Morse       3-Aug-1983
0000    43 ;               Change PR$_ICR to cpu-specific definition, PR780$_ICR.
0000    44 ;
0000    45 ;       V03-004 KDM0032         Kathleen D. Morse       18-Nov-1982
0000    46 ;               Turn off histogram collection for wait time spent on
0000    47 ;               secondary awaiting a reschedule.
0000    48 ;
0000    49 ;       V03-003 KDM0031         Kathleen D. Morse       18-Nov-1982
0000    50 ;               Add performance measurement for secondary-executed
0000    51 ;               kernel system services.
0000    52 ;
```

```
0000     53 ;       V03-002 KDM0022         Kathleen D. Morse       07-Oct-1982
0000     54 ;               Increase number of histogram cells for kernel mode
0000     55 ;               system services.
0000     56 ;
0000     57 ;       V03-001 KDM0015         Kathleen D. Morse       30-Sep-1982
0000     58 ;               Increase number of histogram cells for kernel mode
0000     59 ;               system services.
0000     60 ;
0000     61 ;--
```

```
                     0000     63
                     0000     64 ;
                     0000     65 ; Include files:
                     0000     66 ;
                     0000     67
                     0000     68 ;
                     0000     69 ; Macros:
                     0000     70 ;
                     0000     71 ;
                     0000     72 ; This macro generates a table of longwords that is used to
                     0000     73 ; collect histogram data.
                     0000     74 ;
                     0000     75         .MACRO  HISTO   NAME,CELLCNT,CELLWIDTH
                     0000     76
                     0000     77         PFM$A_HIST_'NAME'::
                     0000     78                 .LONG   CELLCNT
                     0000     79                 .LONG   CELLWIDTH
                     0000     80                 .LONG   0
                     0000     81                 .LONG   0
                     0000     82                 .REPT   CELLCNT
                     0000     83                 .LONG   0
                     0000     84                 .ENDR
                     0000     85                 .LONG   0
                     0000     86
                     0000     87         .ENDM
                     0000     88
                     0000     89 ;
                     0000     90 ; Equated Symbols:
                     0000     91 ;
                     0000     92
                     0000     93         $IPLDEF                         ; Define interrupt priority levels
                     0000     94         $MPSDEF                         ; Define secondary processor states
                     0000     95         $PCBDEF                         ; Define process control block
                     0000     96         $PHDDEF                         ; Define process header block
                     0000     97         $PRDEF                          ; Define processor register numbers
                     0000     98         $PR780DEf                       ; Define 11/780-specific IPR numbers
                     0000     99
                     0000    100 ;
                     0000    101 ; Histogram offsets
                     0000    102 ;
                     0000    103
00000000             0000    104 HST_L_CELLCOUNT = 0                     ; Count of cells in this histogram
00000004             0000    105 HST_L_CELLWIDTH = 4                     ; Width of each cell in histogram
00000008             0000    106 HST_Q_OVRFLOW = 8                       ; Accumulation of overflow values
00000010             0000    107 HST_L_FIRSTCELL = 16                    ; Offset to first cell in histogram
```

```
                     0000   109 ;
                     0000   110 ; Data Area:
                     0000   111 ;
                     0000   112
                 00000000   113         .PSECT  MPPFM,LONG
                     0000   114
                     0000   115 PFM$L_START::                           ; Size of MP perf. meas data
            00000C44'0000   116         .LONG   <PFM$L_END-PFM$L_START> ;
                     0004   117
                     0004   118 PFM$L_CNT_CTXSW::                       ; Count of number of context switches
            00000000 0004   119         .LONG   0                       ;  done on secondary
                     0008   120
                     0008   121 PFM$L_CNT_RESCH::                       ; Count of number of reschedule
            00000000 0008   122         .LONG   0                       ;  requests made by secondary
                     000C   123
                     000C   124 PFM$L_CNT_SCHDS::                       ; Count of number of successful
            00000000 000C   125         .LONG   0                       ;  reschedules of secondary
                     0010   126
                     0010   127 PFM$L_CNT_EXCHG::                       ; Number of times a process was
            00000000 0010   128         .LONG   0                       ;  exchanged between primary & secondary
                     0014   129
                     0014   130 PFM$L_CNT_AST$C::                       ; Number of times an EXEC mode AST
            00000000 0014   131         .LONG   0                       ;  was used to cause a reschedule
                     0018   132
                     0018   133 PFM$L_CNT_INVAL::                       ; Number of invalidates requested by
            00000000 0018   134         .LONG   0                       ;  primary processor
                     001C   135
                     001C   136 PFM$L_CNT_IWAIT::                       ; Number of times primary looped waiting
            00000000 001C   137         .LONG   0                       ;  for secondary to answer invalid req
                     0020   138
                     0020   139 PFM$L_CNT_NWAIT::                       ; Number of wait for event flag system
            00000000 0020   140         .LONG   0                       ;  services that did not wait
                     0024   141
                     0024   142 PFM$L_WHY_CTXSW::                       ; Reason for the next context switch
            00000000 0024   143         .LONG   0                       ; (This value is the offset into the
                     0028   144                                         ;  SCB, for which a request occurred.)
                     0028   145
                     0028   146 PFM$L_WHAT_SRV::                        ; Reason for the next context switch
            00000000 0028   147         .LONG   0                       ; (This is the number specified
                     002C   148                                         ;  in the CHMK instruction.)
                     002C   149
                     002C   150 PFM$L_RUN_TIME::                        ; Accumulator for length of time
            00000000 002C   151         .LONG   0                       ;  process has been running on secondary
                     0030   152
                     0030   153 PFM$L_RSCH_TIME::                       ; Accumulator for length of time
            00000000 0030   154         .LONG   0                       ;  secondary waits for reschedule
                     0034   155
                     0034   156         HISTO   TIME,100,50             ; Histogram of compute time on secondary
                     01D8   157                                         ; (PFM$L_RUN_TIME values)
                     01D8   158                                         ; Cell width = 50 microseconds
                     01D8   159                                         ; Cell count = 100
                     01D8   160
                     01D8   161         HISTO   SRV,84,1                ; Histogram of system services requested
                     033C   162                                         ; (PFM$L_WHAT_SRV values)
                     033C   163                                         ; (One cell for each system service)
                     033C   164
                     033C   165         HISTO   CTX,64,4                ; Histogram of reasons for context
```

```
0450   166                                          ;  switches (PFM$L_WHY_CTXSW values)
0450   167                                          ; (One cell for each SCB vector)
0450   168
0450   169          HISTO   PGFL,100,50             ; Histogram of # pagefaults on secondary
05F4   170                                          ;  per PFM$L_RUN_TIME value
05F4   171                                          ; Cell width = 50 microseconds
05F4   172                                          ; Cell count = 100
05F4   173
05F4   174          HISTO   CHMK,100,50             ; Histogram of system services on
0798   175                                          ;  secondary per PFM$L_RUN_TIME value
0798   176                                          ; Cell width = 50 microseconds
0798   177                                          ; Cell count = 100
0798   178
0798   179          HISTO   OTHR,100,50             ; Histogram of other reasons on
093C   180                                          ;  secondary per PFM$L_RUN_TIME value
093C   181                                          ; Cell width = 50 microseconds
093C   182                                          ; Cell count = 100
093C   183
093C   184          HISTO   SSRV,100,50             ; Histogram of last system service on
0AE0   185                                          ;  secondary per PFM$L_RUN_TIME value
0AE0   186                                          ; Cell width = 50 microseconds
0AE0   187                                          ; Cell count = 100
0AE0   188
0AE0   189          HISTO   KSRV,84,1               ; Histogram of system services executed
0C44   190                                          ;  on secondary processor in kernel mode
0C44   191                                          ; (One cell for each system service)
0C44   192
0C44   193 PFM$L_END::
```

```
                                0C44   195          .SBTTL  MPS$PFM_RUNTIME - Increment Run Time Accumulator
                                0C44   196 ;++
                                0C44   197 ; Functional Description:
                                0C44   198 ;
                                0C44   199 ; This routine is called from the hardware clock interrupt service
                                0C44   200 ; routine.  It increments the amount of run time that a process has
                                0C44   201 ; accumulated while running on the secondary processor.
                                0C44   202 ;
                                0C44   203 ; Calling Sequence:
                                0C44   204 ;
                                0C44   205 ;       BSBW    MPS$PFM_RUNTIME
                                0C44   206 ;
                                0C44   207 ; Input Parameters:
                                0C44   208 ;
                                0C44   209 ;       None
                                0C44   210 ;
                                0C44   211 ; Environment:
                                0C44   212 ;
                                0C44   213 ;       Executes on secondary processor.
                                0C44   214 ;
                                0C44   215 ;--
                                0C44   216
                                0C44   217
                                0C44   218 MPS$PFM_RUNTIME::
F3DF CF    00002710 8F    C0    0C44   219          ADDL2   #10000,W^PFM$L_RUN_TIME ; Update run time accumulator
F3DA CF    00002710 8F    C0    0C4D   220          ADDL2   #10000,W^PFM$L_RSCH_TIME ; Update run time accumulator
                          05    0C56   221          RSB                             ; Return
```

```
                     0C57   223          .SBTTL  MPS$PFM_CTXSW - Increment Context Switch Accumulator
                     0C57   224 ;++
                     0C57   225 ; Functional Description:
                     0C57   226 ;
                     0C57   227 ; This routine is called from the primary's reschedule interrupt service
                     0C57   228 ; routine.  It increments the number off times that the secondary has
                     0C57   229 ; been scheduled to run a process.
                     0C57   230 ;
                     0C57   231 ; Calling Sequence:
                     0C57   232 ;
                     0C57   233 ;       BSBW    MPS$PFM_CTXSW
                     0C57   234 ;
                     0C57   235 ; Input Parameters:
                     0C57   236 ;
                     0C57   237 ;       None
                     0C57   238 ;
                     0C57   239 ; Environment:
                     0C57   240 ;
                     0C57   241 ;       Executes on primary processor.
                     0C57   242 ;
                     0C57   243 ;--
                     0C57   244
                     0C57   245
                     0C57   246 MPS$PFM_CTXSW::
F3A9 CF    D6        0C57   247          INCL    W^PFM$L_CNT_CTXSW       ; Update context switch accumulator
           05        0C5B   248          RSB                             ; Return
```

```
                     0C5C    250          .SBTTL  MPS$PFM_RESCHD - Increment Reschedule Request Accumulator
                     0C5C    251 ;++
                     0C5C    252 ; Functional Description:
                     0C5C    253 ;
                     0C5C    254 ; This routine is called from the primary's reschedule interrupt service
                     0C5C    255 ; routine.  It increments the number off times that the secondary has
                     0C5C    256 ; requested a reschedule event.
                     0C5C    257 ;
                     0C5C    258 ; Calling Sequence:
                     0C5C    259 ;
                     0C5C    260 ;       BSBW    MPS$PFM_RESCHD
                     0C5C    261 ;
                     0C5C    262 ; Input Parameters:
                     0C5C    263 ;
                     0C5C    264 ;       None
                     0C5C    265 ;
                     0C5C    266 ; Environment:
                     0C5C    267 ;
                     0C5C    268 ;       Executes on primary processor.
                     0C5C    269 ;
                     0C5C    270 ;--
                     0C5C    271
                     0C5C    272
                     0C5C    273 MPS$PFM_RESCHD::
F3A8 CF    D6        0C5C    274          INCL    W^PFM$L_CNT_RESCH       ; Update reschedule request accumulator
     05              0C60    275          RSB                             ; Return
                     0C61    276          .SBTTL  MPS$PFM_NWAIT - Increment Null Wait for Event Flag Accumulator
                     0C61    277 ;++
                     0C61    278 ; Functional Description:
                     0C61    279 ;
                     0C61    280 ; This routine is called from the primary's check-event-flag routine that
                     0C61    281 ; it performs on behalf of the secondary (MPS$WAITCK).  It increments the
                     0C61    282 ; number of times the process is returned to the secondary without going
                     0C61    283 ; through a full reschedule onto the primary.
                     0C61    284 ;
                     0C61    285 ; Calling Sequence:
                     0C61    286 ;
                     0C61    287 ;       BSBW    MPS$PFM_NWAIT
                     0C61    288 ;
                     0C61    289 ; Input Parameters:
                     0C61    290 ;
                     0C61    291 ;       None
                     0C61    292 ;
                     0C61    293 ; Environment:
                     0C61    294 ;
                     0C61    295 ;       Executes on primary processor.
                     0C61    296 ;
                     0C61    297 ;--
                     0C61    298
                     0C61    299
                     0C61    300 MPS$PFM_NWAIT::
F3BB CF    D6        0C61    301          INCL    W^PFM$L_CNT_NWAIT       ; Update null wait accumulator
     05              0C65    302          RSB                             ; Return
```

```
                     0C66   304          .SBTTL  MPS$PFM_SCHDSUC - Increment Successful Reschedule Accumulator
                     0C66   305 ;++
                     0C66   306 ; Functional Description:
                     0C66   307 ;
                     0C66   308 ; This routine is called from the primary's reschedule interrupt service
                     0C66   309 ; routine.  It increments the number off times that the secondary has
                     0C66   310 ; been successfully rescheduled.
                     0C66   311 ;
                     0C66   312 ; Calling Sequence:
                     0C66   313 ;
                     0C66   314 ;       BSBW    MPS$PFM_SCHDSUC
                     0C66   315 ;
                     0C66   316 ; Input Parameters:
                     0C66   317 ;
                     0C66   318 ;       None
                     0C66   319 ;
                     0C66   320 ; Environment:
                     0C66   321 ;
                     0C66   322 ;       Executes on primary processor.
                     0C66   323 ;
                     0C66   324 ;--
                     0C66   325
                     0C66   326
                     0C66   327 MPS$PFM_SCHDSUC::
01   0000'CF   D1    0C66   328          CMPL    W^MPS$GL_STATE,#MPS$K_IDLESTATE ; Was reschedule successful?
          04   13    0C6B   329          BEQL    10$                             ; Br if not successful, sec still idle
     F39B CF   D6    0C6D   330          INCL    W^PFM$L_CNT_SCHDS               ; Inc successful reschedule accumulator
               05    0C71   331 10$:     RSB                                     ; Return
```

```
                                0C72   333          .SBTTL  MPS$PFM_EXCHG - Increment Accumulator of Process Exchanges
                                0C72   334 ;++
                                0C72   335 ; Functional Description:
                                0C72   336 ;
                                0C72   337 ; This routine is called from the routine that schedules a process for
                                0C72   338 ; the secondary.  It counts the number of times a process moves from
                                0C72   339 ; the primary to the secondary.
                                0C72   340 ;
                                0C72   341 ; Calling Sequence:
                                0C72   342 ;
                                0C72   343 ;       BSBW    MPS$PFM_EXCHG
                                0C72   344 ;
                                0C72   345 ; Input Parameters:
                                0C72   346 ;
                                0C72   347 ;       None
                                0C72   348 ;
                                0C72   349 ; Environment:
                                0C72   350 ;
                                0C72   351 ;       Executes on primary processor.
                                0C72   352 ;
                                0C72   353 ;--
                                0C72   354
                                0C72   355
                                0C72   356 MPS$PFM_EXCHG::
0000'CF   00000000'GF   D1      0C72   357          CMPL    G^SCH$GL_CURPCB,W^MPS$GL_CURPCB ; Was process exchanged?
                  04    12      0C7B   358          BNEQ    10$                             ; Br if not exchanged
              F38F CF   D6      0C7D   359          INCL    W^PFM$L_CNT_EXCHG               ; Inc exchanged process accumulator
                        05      0C81   360 10$:     RSB                                     ; Return
```

```
                     0C82   362          .SBTTL  MPS$PFM_ASTSC - Increment Accumulator of Exec AST Reschedules
                     0C82   363 ;++
                     0C82   364 ; Functional Description:
                     0C82   365 ;
                     0C82   366 ; This routine is called from the code that determines if a process
                     0C82   367 ; running on the primary should be forced onto the secondary as
                     0C82   368 ; soon as it exits from kernel mode.  It counts the number of times
                     0C82   369 ; an AST interrupt is scheduled to occur, causing a rescheduling
                     0C82   370 ; event as soon as the process exits from kernel mode.
                     0C82   371 ;
                     0C82   372 ; Calling Sequence:
                     0C82   373 ;
                     0C82   374 ;       BSBW    MPS$PFM_ASTSC
                     0C82   375 ;
                     0C82   376 ; Input Parameters:
                     0C82   377 ;
                     0C82   378 ;       None
                     0C82   379 ;
                     0C82   380 ; Environment:
                     0C82   381 ;
                     0C82   382 ;       Executes on primary processor.
                     0C82   383 ;
                     0C82   384 ;--
                     0C82   385
                     0C82   386
                     0C82   387 MPS$PFM_ASTSC::
F38E CF    D6        0C82   388          INCL    W^PFM$L_CNT_ASTSC       ; Inc exchanged process accumulator
           05        0C86   389          RSB                             ; Return
```

```
                                0C87   391          .SBTTL  MPS$PFM_UNEXP - Set Unexpected Interrupt Indicators
                                0C87   392 ;++
                                0C87   393 ; Functional Description:
                                0C87   394 ;
                                0C87   395 ; This routine is called from the unexpected interrupt service
                                0C87   396 ; routine.  It sets indicators for the reason of the interrupt,
                                0C87   397 ; which will be recorded later on in histograms.
                                0C87   398 ;
                                0C87   399 ; Calling Sequence:
                                0C87   400 ;
                                0C87   401 ;       BSBW    MPS$PFM_UNEXP
                                0C87   402 ;
                                0C87   403 ; Input Parameters:
                                0C87   404 ;
                                0C87   405 ;        (SP) - Return address
                                0C87   406 ;       4(SP) - Vector offset within SCB
                                0C87   407 ;       8(SP) - Optional parameters, if CHMK then numerical argument
                                0C87   408 ;       ?(SP) - PC at time of exception
                                0C87   409 ;       ?(SP) - PSL at time of exception
                                0C87   410 ;
                                0C87   411 ; Environment:
                                0C87   412 ;
                                0C87   413 ;       Executes on secondary processor.
                                0C87   414 ;
                                0C87   415 ;--
                                0C87   416
                                0C87   417
                                0C87   418 MPS$PFM_UNEXP::
        F397 CF    04 AE    D0  0C87   419          MOVL    4(SP),W^PFM$L_WHY_CTXSW ; Remember offset into PCB (reason
                                0C8D   420                                          ;  for this interrupt)
00000040 8F    F393 CF    D1  0C8D   421          CMPL    W^PFM$L_WHY_CTXSW,#^X40 ; Is this a CHMK request?
                     06    12  0C96   422          BNEQ    10$                     ; Br if not a CHMK request
        F38A CF    08 AE    D0  0C98   423          MOVL    8(SP),W^PFM$L_WHAT_SRV  ; Remember the number argument to CHMK
                     05    0C9E   424 10$:    RSB                                 ; Return
```

```
                     0C9F   426          .SBTTL  MPS$PFM_ASTDEL - Set AST Delivery Indicator
                     0C9F   427 ;++
                     0C9F   428 ; Functional Description:
                     0C9F   429 ;
                     0C9F   430 ; This routine is called from the AST delivery interrupt service
                     0C9F   431 ; routine.  It sets an indicator for the reason of the interrupt,
                     0C9F   432 ; which will be recorded later on in a histogram.
                     0C9F   433 ;
                     0C9F   434 ; Calling Sequence:
                     0C9F   435 ;
                     0C9F   436 ;       BSBW    MPS$PFM_ASTDEL
                     0C9F   437 ;
                     0C9F   438 ; Input Parameters:
                     0C9F   439 ;
                     0C9F   440 ;       None
                     0C9F   441 ;
                     0C9F   442 ; Environment:
                     0C9F   443 ;
                     0C9F   444 ;       Executes on secondary processor.
                     0C9F   445 ;
                     0C9F   446 ;--
                     0C9F   447
                     0C9F   448 MPS$PFM_ASTDEL::
F37C CF   00000088 8F DO 0C9F   449          MOVL    #^X88,W^PFM$L_WHY_CTXSW ; Indicator is offset in SCB to
                     0CA8   450                                           ;  AST delivery routine
                  05 0CA8   451          RSB                             ; Return
```

```
                     0CA9   453          .SBTTL  MPS$PFM_MCHK - Set Machine Check Indicator
                     0CA9   454 ;++
                     0CA9   455 ; Functional Description:
                     0CA9   456 ;
                     0CA9   457 ; This routine is called from the machine check interrupt service
                     0CA9   458 ; routine.  It sets an indicator for the reason of the interrupt,
                     0CA9   459 ; which will be recorded later on in a histogram.
                     0CA9   460 ;
                     0CA9   461 ; Calling Sequence:
                     0CA9   462 ;
                     0CA9   463 ;       BSBW    MPS$PFM_MCHK
                     0CA9   464 ;
                     0CA9   465 ; Input Parameters:
                     0CA9   466 ;
                     0CA9   467 ;       None
                     0CA9   468 ;
                     0CA9   469 ; Environment:
                     0CA9   470 ;
                     0CA9   471 ;       Executes on secondary processor.
                     0CA9   472 ;
                     0CA9   473 ;--
                     0CA9   474
                     0CA9   475 MPS$PFM_MCHK::
F376 CF    04    D0  0CA9   476          MOVL    #^X04,W^PFM$L_WHY_CTXSW ; Indicator is offset in SCB to
                     0CAE   477                                          ;  machine check routine
                  05 0CAE   478          RSB                             ; Return
```

```
                        0CAF   480          .SBTTL  MPS$PFM_QEND - SetQuantum End Indicator
                        0CAF   481 ;++
                        0CAF   482 ; Functional Description:
                        0CAF   483 ;
                        0CAF   484 ; This routine is called from the quantum end interrupt service
                        0CAF   485 ; routine.  It sets an indicator for the reason of the interrupt,
                        0CAF   486 ; which will be recorded later on in a histogram.
                        0CAF   487 ;
                        0CAF   488 ; Calling Sequence:
                        0CAF   489 ;
                        0CAF   490 ;       BSBW    MPS$PFM_QEND
                        0CAF   491 ;
                        0CAF   492 ; Input Parameters:
                        0CAF   493 ;
                        0CAF   494 ;       None
                        0CAF   495 ;
                        0CAF   496 ; Environment:
                        0CAF   497 ;
                        0CAF   498 ;       Executes on secondary processor.
                        0CAF   499 ;
                        0CAF   500 ;--
                        0CAF   501
                        0CAF   502 MPS$PFM_QEND::
F36C CF   0000009C 8F   D0  0CAF   503          MOVL    #^X9C,W^PFM$L_WHY_CTXSW ; Indicator is offset in SCB to
                            0CB8   504                                          ;  software timer interrupt routine
                            0CB9   505                                          ; (i.e., quantum end)
                            05  0CB8   506          RSB                         ; Return
```

```
                                0CB9    508          .SBTTL  MPS$PFM_SVPCTX - Save Process Context Measurement Routine
                                0CB9    509 ;++
                                0CB9    510 ; Functional Description:
                                0CB9    511 ;
                                0CB9    512 ; This routine is called from the scheduling routine for the secondary
                                0CB9    513 ; processor, at the time it folds up a process and hands it back
                                0CB9    514 ; to the primary.  It is used to compute the actual run time accumulated
                                0CB9    515 ; by the process while it was running on the secondary, and store this
                                0CB9    516 ; value in a histogram.
                                0CB9    517 ;
                                0CB9    518 ; Calling Sequence:
                                0CB9    519 ;
                                0CB9    520 ;       BSBW    MPS$PFM_SVPCTX
                                0CB9    521 ;
                                0CB9    522 ; Input Parameters:
                                0CB9    523 ;
                                0CB9    524 ;       None
                                0CB9    525 ;
                                0CB9    526 ; Environment:
                                0CB9    527 ;
                                0CB9    528 ;       Executes on secondary processor, at IPLs SYNCH and HWCLK.
                                0CB9    529 ;
                                0CB9    530 ;--
                                0CB9    531
                                0CB9    532 MPS$PFM_SVPCTX::
                          0F    BB    0CB9    533          PUSHR   #^M<R0,R1,R2,R3>        ; Save registers
                                0CBB    534          SETIPL  #IPL$_HWCLK             ; Lock out secondary clock interrupts
                                0CBE    535          MFPR    #PR780$_ICR,R0          ; Get usec offset from 10 milsec marker
                                0CC1    536          MFPR    #PR$_ICCS,R1            ; Get status register
        52    F364 CF    D0    0CC4    537          MOVL    W^PFM$L_RUN_TIME,R2     ; Get run time accumulator
                                0CC9    538          SETIPL  #IPL$_SYNCH             ; Enable secondary clock interrupts
          10 51    07    E1    0CCC    539          BBC     #7,R1,10$               ; If BC, no overflow
    50    FFFFEC78 8F    D1    0CD0    540          CMPL    #-5000,R0               ; Overflow after read ?
                   07    19    0CD7    541          BLSS    10$                     ; If LSS, yes.  Correction not needed
    52    00002710 8F    C0    0CD9    542          ADDL    #10000,R2               ; Correct run time accumulator
                                0CE0    543 10$:
    50    00002710 8F    C0    0CE0    544          ADDL2   #10000,R0               ; (10^4 - X)
          52    50       C0    0CE7    545          ADDL2   R0,R2                   ; Total run time (usec)
                                0CEA    546
                                0CEA    547 ;
                                0CEA    548 ; Increment delta time histogram and compute index into this histogram
                                0CEA    549 ;
        53    F346 CF    9E    0CEA    550          MOVAB   W^PFM$A_HIST_TIME,R3    ; Get address of histogram
              51   52    D0    0CEF    551          MOVL    R2,R1                   ; Remember amount in case of overflow
            52   04 A3    C6    0CF2    552          DIVL2   HST_L_CELLWIDTH(R3),R2  ; Compute the histogram index
              52   63    D1    0CF6    553          CMPL    HST_L_CELLCOUNT(R3),R2  ; Out of range ?
                   0B    1E    0CF9    554          BGEQU   20$                     ; If GEQ, no
              52   63    D0    0CFB    555          MOVL    HST_L_CELLCOUNT(R3),R2  ; Set index to overflow cell
            08 A3   51    C0    0CFE    556          ADDL    R1,HST_Q_OVRFLOW(R3)    ; Add to overflow accumulator
            0C A3   00    D8    0D02    557          ADWC    #0,HST_Q_OVRFLOW+4(R3)  ;  in quadword arithmetic
                                0D06    558 20$:
              10 A342    D6    0D06    559          INCL    HST_L_FIRSTCELL(R3)[R2] ; Update histogram
                                0D0A    560
                                0D0A    561 ;
                                0D0A    562 ; Increment histogram of reasons why context switch occurred --
                                0D0A    563 ; (one for each SCB entry)
                                0D0A    564 ;
```

```
                51   F316 CF    D0   0D0A   565          MOVL    W^PFM$L_WHY_CTXSW,R1     ; Get offset into SCB (reason for ctxsw)
                53   F629 CF    9E   0D0F   566          MOVAB   W^PFM$A_HIST_CTX,R3      ; Get address of histogram
                  51    04 A3    C6   0D14   567          DIVL2   HST_L_CELLWIDTH(R3),R1   ; Compute the histogram index
                     51    63    D1   0D18   568          CMPL    HST_L_CELLCOUNT(R3),R1   ; Out of range ?
                           03    1E   0D1B   569          BGEQU   30$                      ; If GEQ, no
                     51    63    D0   0D1D   570          MOVL    HST_L_CELLCOUNT(R3),R1   ; Set index to overflow cell
                                    0D20   571 30$:
                     10 A341    D6   0D20   572          INCL    HST_L_FIRSTCELL(R3)[R1] ; Update histogram
                                    0D24   573
                                    0D24   574 ;
                                    0D24   575 ; If reason was a CHMK #n, then increment the histogram of which system
                                    0D24   576 ; service was requested.
                                    0D24   577 ;
00000040 8F    F2FC CF    D1   0D24   578          CMPL    W^PFM$L_WHY_CTXSW,#^X40 ; Is this a CHMK request?
                           31   12   0D2D   579          BNEQ    50$                      ; Br if not a CHMK request
                51   F2F5 CF    D0   0D2F   580          MOVL    W^PFM$L_WHAT_SRV,R1      ; Get the argument to the CHMK instr
                53   F4A0 CF    9E   0D34   581          MOVAB   W^PFM$A_HIST_SRV,R3      ; Get address of histogram
                  51    04 A3    C6   0D39   582          DIVL2   HST_L_CELLWIDTH(R3),R1   ; Compute the histogram index
                     51    63    D1   0D3D   583          CMPL    HST_L_CELLCOUNT(R3),R1   ; Out of range ?
                           03    1E   0D40   584          BGEQU   40$                      ; If GEQ, no
                     51    63    D0   0D42   585          MOVL    HST_L_CELLCOUNT(R3),R1   ; Set index to overflow cell
                                    0D45   586 40$:
                     10 A341    D6   0D45   587          INCL    HST_L_FIRSTCELL(R3)[R1] ; Update histogram
                                    0D49   588
                                    0D49   589 ;
                                    0D49   590 ; Increment system service histogram that corresponds to the
                                    0D49   591 ; delta time histogram.
                                    0D49   592 ;
                53   F8A7 CF    9E   0D49   593          MOVAB   W^PFM$A_HIST_CHMK,R3     ; Get address of histogram
                     10 A342    D6   0D4E   594          INCL    HST_L_FIRSTCELL(R3)[R2] ; Update histogram
                                    0D52   595
                                    0D52   596 ; Set indicator histogram cell, showing last system service in this
                                    0D52   597 ; delta time interval.  This corresponds to the delta time histogram.
                                    0D52   598 ;
                  53   FBE6 CF    9E   0D52   599          MOVAB   W^PFM$A_HIST_SSRV,R3     ; Get address of histogram
             10 A342   F2CD CF    D0   0D57   600          MOVL    W^PFM$L_WHAT_SRV,HST_L_FIRSTCELL(R3)[R2] ; Record latest srv
                           1B   11   0D5E   601          BRB     100$
                                    0D60   602
                                    0D60   603 ;
                                    0D60   604 ; If reason was a pagefault, increment pagefault histogram that
                                    0D60   605 ; corresponds to the delta time histogram.
                                    0D60   606 ;
                                    0D60   607 50$:
                  24   F2C0 CF    D1   0D60   608          CMPL    W^PFM$L_WHY_CTXSW,#^X24 ; Is this a pagefault request?
                           0B   12   0D65   609          BNEQ    60$                      ; Br if not a pagefault request
                  53   F6E5 CF    9E   0D67   610          MOVAB   W^PFM$A_HIST_PGFL,R3     ; Get address of histogram
                     10 A342    D6   0D6C   611          INCL    HST_L_FIRSTCELL(R3)[R2] ; Update histogram
                           09   11   0D70   612          BRB     100$
                                    0D72   613
                                    0D72   614 ;
                                    0D72   615 ; This was neither pagefault or CHMK reason, increment histogram of
                                    0D72   616 ; other reasons that corresponds to delta time histogram.
                                    0D72   617 ;
                                    0D72   618 60$:
                  53   FA22 CF    9E   0D72   619          MOVAB   W^PFM$A_HIST_OTHR,R3     ; Get address of histogram
                     10 A342    D6   0D77   620          INCL    HST_L_FIRSTCELL(R3)[R2] ; Update histogram
                                    0D7B   621
```

```
                OF   BA   0D7B    622 100$:   POPR    #^M<R0,R1,R2,R3>        ; Restore registers
                     05   0D7D    623         RSB                             ; Return
```

```
                                0D7E   625          .SBTTL  MPS$PFM_LDPCTX - Load Process Context Measurement Routine
                                0D7E   626 ;++
                                0D7E   627 ; Furctional Description:
                                0D7E   628 ;
                                0D7E   629 ; This routine is called from the scheduling routine for the secondary
                                0D7E   630 ; processor, at the time it starts executing a new process.  It is
                                0D7E   631 ; used to initialize accumulators used in computing the amount of run time
                                0D7E   632 ; accumulated by the process while it was running on the secondary.
                                0D7E   633 ;
                                0D7E   634 ; Calling Sequence:
                                0D7E   635 ;
                                0D7E   636 ;       BSBW    MPS$PFM_LDPCTX
                                0D7E   637 ;
                                0D7E   638 ; Input Parameters:
                                0D7E   639 ;
                                0D7E   640 ;       None
                                0D7E   641 ;
                                0D7E   642 ; Environment:
                                0D7E   643 ;
                                0D7E   644 ;       Executes on secondary processor, at IPLs SYNCH and HWCLK.
                                0D7E   645 ;
                                0D7E   646 ;--
                                0D7E   647
                                0D7E   648 MPS$PFM_LDPCTX::
                      7E   50   7D 0D7E   649          MOVQ    R0,-(SP)                ; Save registers
                                0D81   650          SETIPL  #IPL$_HWCLK             ; Lock out secondary clock interrupts
                                0D84   651          MFPR    #PR780$_ICR,R0          ; (-X)
                                0D87   652          MFPR    #PR$_ICCS,R1            ; Clock status register
                   10 51    07  E1 0D8A   653          BBC     #7,R1,10$               ; If BC, no overflow
                50 FFFFEC78 8F  D1 0D8E   654          CMPL    #-5000,R0               ; Overflow after read ?
                            07  19 0D95   655          BLSS    10$                     ; If LSS, yes.  Correction not needed
                50 00002710 8F  C0 0D97   656          ADDL2   #10000,R0               ; Correct for pending clock interrupt
                50 00002710 8F  C0 0D9E   657 10$:     ADDL2   #10000,R0               ; (10^4 - X)
                    F282 CF     50 CE 0DA5   658          MNEGL   R0,PFM$L_RUN_TIME       ; CPU = CPU - (10^4 - X)
                                0DAA   659          SETIPL  #IPL$_SYNCH             ; Enable secondary clock interrupts
                      50    8E   7D 0DAD   660          MOVQ    (SP)+,R0                ; Restore registers
                                05 0DB0   661          RSB                             ; Return
```

```
                                0DB1   663          .SBTTL  MPS$PFM_INTP - Remember Time Interrupted Primary for Reschedule
                                0DB1   664 ;++
                                0DB1   665 ; Functional Description:
                                0DB1   666 ;
                                0DB1   667 ; This routine is called from the scheduling code for the secondary
                                0DB1   668 ; processor, at the time that it requests the primary to reschedule it.
                                0DB1   669 ; It is used to initialize accumulators used in computing the amount
                                0DB1   670 ; of run time accumulated by the process while it was running on
                                0DB1   671 ; the secondary.
                                0DB1   672 ;
                                0DB1   673 ; Calling Sequence:
                                0DB1   674 ;
                                0DB1   675 ;       BSBW    MPS$PFM_INTP
                                0DB1   676 ;
                                0DB1   677 ; Input Parameters:
                                0DB1   678 ;
                                0DB1   679 ;       None
                                0DB1   680 ;
                                0DB1   681 ; Environment:
                                0DB1   682 ;
                                0DB1   683 ;       Executes on secondary processor, at IPLs SYNCH and HWCLK.
                                0DB1   684 ;
                                0DB1   685 ;--
                                0DB1   686
                                0DB1   687 MPS$PFM_INTP::
                      7E    50  7D    0DB1   688          MOVQ    R0,-(SP)                ; Save registers
                                0DB4   689          SETIPL  #IPL$_HWCLK             ; Lock out secondary clock interrupts
                                0DB7   690          MFPR    #PR780$_ICR,R0          ; (-X)
                                0DBA   691          MFPR    #PR$_ICCS,R1            ; Clock status register
                   10 51    07  E1    0DBD   692          BBC     #7,R1,10$               ; If BC, no overflow
                50    FFFFEC78 8F  D1    0DC1   693          CMPL    #-5000,R0               ; Overflow after read ?
                         07    19    0DC8   694          BLSS    10$                     ; If LSS, yes.  Correction not needed
                50    00002710 8F  C0    0DCA   695          ADDL2   #10000,R0               ; Correct for pending clock interrupt
                50    00002710 8F  C0    0DD1   696 10$:     ADDL2   #10000,R0               ; (10^4 - X)
                   F253 CF    50  CE    0DD8   697          MNEGL   R0,PFM$L_RSCH_TIME      ; CPU = CPU - (10^4 - X)
                                0DDD   698          SETIPL  #IPL$_SYNCH             ; Enable secondary clock interrupts
                         50    8E  7D    0DE0   699          MOVQ    (SP)+,R0                ; Restore registers
                         05        0DE3   700          RSB                             ; Return
                                0DE4   701          .SBTTL  MPS$PFM_KSRV - Count Secondary Kernel System Services
                                0DE4   702 ;++
                                0DE4   703 ; Functional Description:
                                0DE4   704 ;
                                0DE4   705 ; This routine is called from the secondary's wait-for-event-flag
                                0DE4   706 ; system services.  It is incremented once for each service that does
                                0DE4   707 ; not return to the primary for handling.
                                0DE4   708 ;
                                0DE4   709 ; Calling Sequence:
                                0DE4   710 ;
                                0DE4   711 ;       BSBW    MPS$PFM_KSRV
                                0DE4   712 ;
                                0DE4   713 ; Input Parameters:
                                0DE4   714 ;
                                0DE4   715 ;       None
                                0DE4   716 ;
                                0DE4   717 ; Environment:
                                0DE4   718 ;
                                0DE4   719 ;       Executes on secondary processor.
```

```
                     0DE4    720 ;
                     0DE4    721 ;--
                     0DE4    722
                     0DE4    723
                     0DE4    724 MPS$PFM_KSRV::
      7E    51   7D  0DE4    725          MOVQ    R1,-(SP)                        ; Save registers
51    F23D  CF   D0  0DE7    726          MOVL    W^PFM$L_WHAT_SRV,R1             ; Get the argument to the CHMK instr
52    FCF0  CF   9E  0DEC    727          MOVAB   W^PFM$A_HIST_KSRV,R2            ; Get the address of the histogram
  51    04  A2   C6  0DF1    728          DIVL2   HST_L_CELLWIDTH(R2),R1          ; Compute the histogram index
      51    62   D1  0DF5    729          CMPL    HST_L_CELLCOUNT(R2),R1          ; Out of range?
            03   1E  0DF8    730          BGEQU   10$                             ; If GEQ, no
      51    62   D0  0DFA    731          MOVL    HST_L_CELLCOUNT(R2),R1          ; Set index to overflow cell
                   0DFD    732 10$:
      10 A241    D6  0DFD    733          INCL    HST_L_FIRSTCELL(R2)[R1] ; Update histogram
      51    8E   7D  0E01    734          MOVQ    (SP)+,R1                        ; Restore registers
                   05  0E04    735          RSB                                     ; Return
```

```
                                0E05    737          .SBTTL  MPS$PFM_CLRDATA - Clear All Performance Measurement Data
                                0E05    738 ;++
                                0E05    739 ; Functional Description:
                                0E05    740 ;
                                0E05    741 ; This routine is called from the initialization code for the secondary
                                0E05    742 ; processor.  It is used to initialize accumulators used so that both
                                0E05    743 ; the primary and secondary times can be displayed.
                                0E05    744 ;
                                0E05    745 ; Calling Sequence:
                                0E05    746 ;
                                0E05    747 ;       BSBW    MPS$PFM_CLRDATA
                                0E05    748 ;
                                0E05    749 ; Input Parameters:
                                0E05    750 ;
                                0E05    751 ;       None
                                0E05    752 ;
                                0E05    753 ; Environment:
                                0E05    754 ;
                                0E05    755 ;       Executes on secondary processor.
                                0E05    756 ;
                                0E05    757 ;--
                                0E05    758
                                0E05    759 MPS$PFM_CLRDATA::
                       007F 8F  BB    0E05    760          PUSHR   #^M<R0,R1,R2,R3,R4,R5,R6>
                56  00000000'GF  D0    0E09    761          MOVL    G^EXE$GL_MP,R6                  ;Get adr of loaded MP code
                         51  05  9A    0E10    762 5$:      MOVZBL  #5,R1
                   00000000'GF41  D4    0E13    763 10$:     CLRL    G^MPS$AL_CPUTIME[R1]
                         F6 51    F4    0E1A    764          SOBGEQ  R1,10$
                                  0E1D    765
                         51  05  9A    0E1D    766          MOVZBL  #5,R1
                   00000000'GF41  D4    0E20    767 20$:     CLRL    G^PMS$GL_KERNEL[R1]
                         F6 51    F4    0E27    768          SOBGEQ  R1,20$
                                  0E2A    769
                50  00000000'GF  9E    0E2A    770          MOVAB   G^SCH$GL_NULLPCB,R0
                      50  6C A0  D0    0E31    771          MOVL    PCB$L_PHD(R0),R0
                          38 A0  D4    0E35    772          CLRL    PHD$L_CPUTIM(R0)
                                  0E38    773
                    00000004'GF  D4    0E38    774          CLRL    G^PFM$L_CNT_CTXSW
                    00000008'GF  D4    0E3E    775          CLRL    G^PFM$L_CNT_RESCH
                    0000000C'GF  D4    0E44    776          CLRL    G^PFM$L_CNT_SCHDS
                    00000018'GF  D4    0E4A    777          CLRL    G^PFM$L_CNT_INVAL
                    0000001C'GF  D4    0E50    778          CLRL    G^PFM$L_CNT_IWAIT
                    00000010'GF  D4    0E56    779          CLRL    G^PFM$L_CNT_EXCHG
                    00000014'GF  D4    0E5C    780          CLRL    G^PFM$L_CNT_ASTSC
                         F1BA CF  D4    0E62    781          CLRL    PFM$L_CNT_NWAIT
                                  0E66    782
                      50 F1CA CF  9E    0E66    783          MOVAB   W^PFM$A_HIST_TIME,R0            ;Get address of histogram
                      51 04    60  C5    0E6B    784          MULL3   HST_L_CELLCOUNT(R0),#4,R1
                         51    0C  C0    0E6F    785          ADDL    #12,R1                          ;Add in overflow cell
                         50    08  C0    0E72    786          ADDL    #HST_Q_OVRFLOW,R0               ;Point past cell count and size
60    51    00    60    00    2C    0E75    787          MOVC5   #0,(R0),#0,R1,(R0)              ;Clear performance meas data
                                  0E7B    788
                      50 F359 CF  9E    0E7B    789          MOVAB   W^PFM$A_HIST_SRV,R0             ;Get address of histogram
                      51 04    60  C5    0E80    790          MULL3   HST_L_CELLCOUNT(R0),#4,R1
                         51    0C  C0    0E84    791          ADDL    #12,R1                          ;Add in overflow cell
                         50    08  C0    0E87    792          ADDL    #HST_Q_OVRFLOW,R0               ;Point past cell count and size
60    51    00    60    00    2C    0E8A    793          MOVC5   #0,(R0),#0,R1,(R0)              ;Clear performance meas data
```

```
                                0E90    794
                50   F4A8 CF    9E   0E90    795          MOVAB   W^PFM$A_HIST_CTX,R0              ;Get address of histogram
                51   04   60    C5   0E95    796          MULL3   HST_L_CELLCOUNT(R0),#4,R1
                     51   0C    C0   0E99    797          ADDL    #12,R1                          ;Add in overflow cell
                     50   08    C0   0E9C    798          ADDL    #HST_Q_OVRFLOW,R0               ;Point past cell count and size
     60   51   00   60   00    2C   0E9F    799          MOVC5   #0,(R0),#0,R1,(R0)              ;Clear performance meas data
                                0EA5    800
                50   F5A7 CF    9E   0EA5    801          MOVAB   W^PFM$A_HIST_PGFL,R0            ;Get address of histogram
                51   04   60    C5   0EAA    802          MULL3   HST_L_CELLCOUNT(R0),#4,R1
                     51   0C    C0   0EAE    803          ADDL    #12,R1                          ;Add in overflow cell
                     50   08    C0   0EB1    804          ADDL    #HST_Q_OVRFLOW,R0               ;Point past cell count and size
     60   51   00   60   00    2C   0EB4    805          MOVC5   #0,(R0),#0,R1,(R0)              ;Clear performance meas data
                                0EBA    806
                50   F736 CF    9E   0EBA    807          MOVAB   W^PFM$A_HIST_CHMK,R0            ;Get address of histogram
                51   04   60    C5   0EBF    808          MULL3   HST_L_CELLCOUNT(R0),#4,R1
                     51   0C    C0   0EC3    809          ADDL    #12,R1                          ;Add in overflow cell
                     50   08    C0   0EC6    810          ADDL    #HST_Q_OVRFLOW,R0               ;Point past cell count and size
     60   51   00   60   00    2C   0EC9    811          MOVC5   #0,(R0),#0,R1,(R0)              ;Clear performance meas data
                                0ECF    812
                50   F8C5 CF    9E   0ECF    813          MOVAB   W^PFM$A_HIST_OTHR,R0            ;Get address of histogram
                51   04   60    C5   0ED4    814          MULL3   HST_L_CELLCOUNT(R0),#4,R1
                     51   0C    C0   0ED8    815          ADDL    #12,R1                          ;Add in overflow cell
                     50   08    C0   0EDB    816          ADDL    #HST_Q_OVRFLOW,R0               ;Point past cell count and size
     60   51   00   60   00    2C   0EDE    817          MOVC5   #0,(R0),#0,R1,(R0)              ;Clear performance meas data
                                0EE4    818
                50   FA54 CF    9E   0EE4    819          MOVAB   W^PFM$A_HIST_SSRV,R0            ;Get address of histogram
                51   04   60    C5   0EE9    820          MULL3   HST_L_CELLCOUNT(R0),#4,R1
                     51   0C    C0   0EED    821          ADDL    #12,R1                          ;Add in overflow cells
                     50   08    C0   0EF0    822          ADDL    #HST_Q_OVRFLOW,R0               ;Point past cell count and size
     60   51   00   60   00    2C   0EF3    823          MOVC5   #0,(R0),#0,R1,(R0)              ;Clear performance meas data
                                0EF9    824
                50   FBE3 CF    9E   0EF9    825          MOVAB   W^PFM$A_HIST_KSRV,R0            ;Get address of histogram
                51   04   60    C5   0EFE    826          MULL3   HST_L_CELLCOUNT(R0),#4,R1
                     51   0C    C0   0F02    827          ADDL    #12,R1                          ;Add in overflow cell
                     50   08    C0   0F05    828          ADDL    #HST_Q_OVRFLOW,R0               ;point past cell count and size
     60   51   00   60   00    2C   0F08    829          MOVC5   #0,(R0),#0,R1,(R0)              ;Clear performance meas data
                                0F0E    830
                     007F 8F    BA   0F0E    831          POPR    #^M<R0,R1,R2,R3,R4,R5,R6>
                                05   0F12    832          RSB                                     ;Return
                                     0F13    833
                                     0F13    834          .END
```

```
EXE$GL_MP                       ********    X   02
HST_L_CELLCOUNT                 = 00000000
HST_L_CELLWIDTH                 = 0000C004
HST_L_FIRSTCELL                 = 00000010
HST_Q_OVRFLOW                   = 00000008
IPL$_AWCLK                      = 00000018
IPL$_SYNCH                      = 00000008
MPS$AL_CPUTIME                  ********    X   02
MPS$GL_CURPCB                   ********    X   02
MPS$GL_STATE                    ********    X   02
MPS$K_IDLESTATE                 = 00000001
MPS$PFM_ASTDEL                  00000C9F RG     02
MPS$PFM_ASTSC                   00000C82 RG     02
MPS$PFM_CLRDATA                 00000E05 RG     02
MPS$PFM_CTXSW                   00000C57 RG     02
MPS$PFM_EXCHG                   00000C72 RG     02
MPS$PFM_INTP                    00000DB1 RG     02
MPS$PFM_KSRV                    00000DE4 RG     02
MPS$PFM_LDPCTX                  00000D7E RG     02
MPS$PFM_MCHK                    00000CA9 RG     02
MPS$PFM_NWAIT                   00000C61 RG     02
MPS$PFM_QEND                    00000CAF RG     02
MPS$PFM_RESCHD                  00000C5C RG     02
MPS$PFM_RUNTIME                 00000C44 RG     02
MPS$PFM_SCHDSUC                 00000C66 RG     02
MPS$PFM_SVPCTX                  00000CB9 RG     02
MPS$PFM_UNEXP                   00000C87 RG     02
PCB$L_PRD                       = 0000006C
PFM$A_HIST_CHMK                 000005F4 RG     02
PFM$A_HIST_CTX                  0000033C RG     02
PFM$A_HIST_KSRV                 00000AE0 RG     02
PFM$A_HIST_OTHR                 00000798 RG     02
PFM$A_HIST_PGFL                 00000450 RG     02
PFM$A_HIST_SRV                  000001D8 RG     02
PFM$A_HIST_SSRV                 0000093C RG     02
PFM$A_HIST_TIME                 00000034 RG     02
PFM$L_CNT_ASTSC                 00000014 RG     02
PFM$L_CNT_CTXSW                 00000004 RG     02
PFM$L_CNT_EXCHG                 00000010 RG     02
PFM$L_CNT_INVAL                 00000018 RG     02
PFM$L_CNT_IWAIT                 0000001C RG     02
PFM$L_CNT_NWAIT                 00000020 RG     02
PFM$L_CNT_RESCH                 00000008 RG     02
PFM$L_CNT_SCHDS                 0000000C RG     02
PFM$L_END                       00000C44 RG     02
PFM$L_RSCH_TIME                 00000030 RG     02
PFM$L_RUN_TIME                  0000002C RG     02
PFM$L_START                     00000000 RG     02
PFM$L_WHAT_SRV                  00000028 RG     02
PFM$L_WHY_CTXSW                 00000024 RG     02
PHD$L_CPUTIM                    = 00000038
PMS$GL_KERNEL                   ********    X   02
PR$_ICCS                        = 00000018
PR$_IPL                         = 00000012
PR780$_ICR                      = 0000001A
SCH$GL_CURPCB                   ********    X   02
SCH$GL_NULLPCB                  ********    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
MPPFM	00000F13 (3859.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	30	00:00:00.12	00:00:00.95
Command processing	138	00:00:00.87	00:00:05.58
Pass 1	256	00:00:08.52	00:00:22.17
Symbol table sort	0	00:00:00.62	00:00:00.66
Pass 2	156	00:00:02.76	00:00:08.84
Symbol table output	7	00:00:00.08	00:00:00.40
Psect synopsis output	2	00:00:00.03	00:00:00.06
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	591	00:00:13.01	00:00:38.67

The working set limit was 1650 pages.
47910 bytes (94 pages) of virtual memory were used to buffer the intermediate code.
There were 30 pages of symbol table space allocated to hold 428 non-local and 16 local symbols.
839 source lines were read in Pass 1, producing 23 object records in Pass 2.
16 pages of virtual memory were used to define 15 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[MP.OBJ]MP.MLB;1	8
_255DUA28:[SYS.OBJ]LIB.MLB;1	4
_255DUA28:[SYSLIB]STARLET.MLB;2	5
TOTALS (all libraries)	17

627 GETS were required to define 17 macros.

There were no errors, warnings or information messages.

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

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