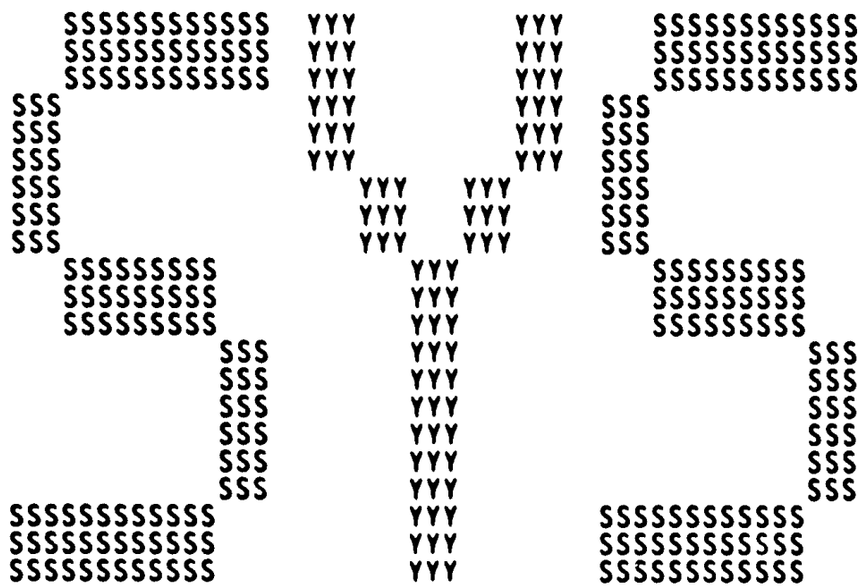

```
IIIIII      000000    PPPPPPPP    EEEEEEEEEE  RRRRRRRR    FFFFFFFFFF    000000    RRRRRRRR    MM      MM
IIIIII      000000    PPPPPPPP    EEEEEEEEEE  RRRRRRRR    FFFFFFFFFF    000000    RRRRRRRR    MM      MM
  II      00      00  PP      PP  EE          RR      RR  FF          00      00  RR      RR  MMMM  MMMM
  II      00      00  PP      PP  EE          RR      RR  FF          00      00  RR      RR  MMMM  MMMM
  II      00      00  PP      PP  EE          RR      RR  FF          00      00  RR      RR  MM  MM  MM
  II      00      00  PP      PP  EE          RR      RR  FF          00      00  RR      RR  MM  MM  MM
  II      00      00  PPPPPPPP    EEEEEEEE    RRRRRRRR    FFFFFFF     00      00  RRRRRRRR    MM      MM
  II      00      00  PPPPPPPP    EEEEEEEE    RRRRRRRR    FFFFFFF     00      00  RRRRRRRR    MM      MM
  II      00      00  PP          EE          RR  RR      FF          00      00  RR  RR      MM      MM
  II      00      00  PP          EE          RR  RR      FF          00      00  RR  RR      MM      MM
  II      00      00  PP          EE          RR    RR    FF          00      00  RR    RR    MM      MM      ....
  II      00      00  PP          EE          RR    RR    FF          00      00  RR    RR    MM      MM      ....
IIIIII      000000    PP          EEEEEEEEEE  RR      RR  FF            000000    RR      RR  MM      MM      ....
IIIIII      000000    PP          EEEEEEEEEE  RR      RR  FF            000000    RR      RR  MM      MM      ....

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
```

```
0000     1          .TITLE  IOPERFORM - I/O PERFORMANCE DATA COLLECTION
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  ; D. N. CUTLER 8-NOV-77
0000    29  ;
0000    30  ; I/O PERFORMANCE MEASUREMENT DATA COLLECTION ROUTINES
0000    31  ;
0000    32  ; MODIFIED BY:
0000    33  ;
0000    34  ;       V03-005 SSA00024        Stan Amway              9-Apr-1984
0000    35  ;               If available, store file access information from WCB
0000    36  ;               (SRQ transaction). When monitoring disks, this will allow
0000    37  ;               trace-driven reduction & simulation programs (e.g., disk
0000    38  ;               cache analysis) to make use of the information.
0000    39  ;
0000    40  ;       V03-004 SSA00008        Stan Amway              10-Feb-1984
0000    41  ;               Changed time stamp to quadword.
0000    42  ;
0000    43  ;       V03-003 SSA00002        Stan Amway              30-Sep-1983
0000    44  ;               Added routine PMS$ABORT_RQ to be called by FDT routines
0000    45  ;               (usually BACKOUT_QIO in SYSQIOFDT) that have aborted or
0000    46  ;               restarted an I/O request AND the IRP will not be posted
0000    47  ;               via IOCIOPOST.
0000    48  ;
0000    49  ;               Changed IPL 15 references to IPL$_PERFMON, a new symbol.
0000    50  ;
0000    51  ;       V03-002 SSA00001        Stan Amway              13-Sep-1983
0000    52  ;               Changed PMS$START_IO to capture longword transfer
0000    53  ;               byte count.
0000    54  ;
0000    55  ;       V03-001 LJK47917        Lawrence J. Kenah       10-Aug-1982
0000    56  ;               Insure that stack looks the same at all entry points so that
0000    57  ;               common exit code can be used.
```

```
0000     58 ;
0000     59 ; MACRO LIBRARY CALLS
0000     60 ;
0000     61
0000     62          $CADEF                          ;DEFINE CONDITIONAL ASSEMBLY PARAMETERS
0000     63          $IPLDEF                         ;DEFINE PROCESSOR IPL LEVELS
0000     64          $IRPDEF                         ;DEFINE IRP OFFSETS
0000     65          $PBHDEF                         ;DEFINE PBH OFFSETS
0000     66          $PDBDEF                         ;DEFINE PDB OFFSETS
0000     67          $PIBDEF                         ;DEFINE PIB OFFSETS
0000     68          $UCBDEF                         ;DEFINE UCB OFFSETS
0000     69          $WCBDEF                         ;Define WCB offsets
```

```
                     0000     71          .SBTTL  ABORTED I/O REQUEST
                     0000     72 ;+
                     0000     73 ; PMS$ABORT_RQ - ABORTED I/O REQUEST
                     0000     74 ;
                     0000     75 ; THIS ROUTINE IS CALLED WHEN AN I/O REQUEST HAS BEEN ABORTED
                     0000     76 ; TO ENTER A MESSAGE IN THE I/O PERFORMANCE DATA COLLECTION BUFFE .
                     0000     77 ;
                     0000     78 ; INPUTS:
                     0000     79 ;
                     0000     80 ;       R3 = ADDRESS OF I/O REQUEST PACKET.
                     0000     81 ;
                     0000     82 ;       IPL CAN BE ANY SOFTWARE IPL >= IPL$_ASTDEL
                     0000     83 ;
                     0000     84 ; OUTPUTS:
                     0000     85 ;
                     0000     86 ;       IF I/O PERFORMANCE DATA COLLECTION IS ENABLED AND THE CURRENT I/O PACKET
                     0000     87 ;       CONFORMS TO THE SELECTION CRITERIA, THEN AN ABORTED I/O REQUEST IS
                     0000     88 ;       PLACED IN THE CURRENT DATA BUFFER.
                     0000     89 ;
                     0000     90 ;       ALL REGISTERS ARE PRESERVED ACROSS CALL.
                     0000     91 ;-
                     0000     92
                     0000     93          .IF DF  CA$_MEASURE_IOT
                     0000     94
                00000000     95          .PSECT  $AEXENUNPAGED
                     0000     96 PMS$ABORT_RQ::
                     0000     97          TSTL    W^PMS$GL_IOPFMPDB       ;ABORTED I/O REQUEST
0000'CF    D5    0000     98          BNEQ    5$                      ;DATA COLLECTION ENABLED?
      01    12    0004     99          RSB                             ;BR IF YES
            05    0006    100 5$:     DSBINT  #IPL$_PERFMON           ;ELSE RETURN
                0007    101          PUSHR   #^M<R0,R1,R2,R3>        ;DISABLE SOFTWARE INTERRUPTS
      0F    BB    000D    102          MOVZBL  #PIB$K_ARQ_SIZE,R1      ;SAVE REGISTERS
51    10    9A    000F    103          BSBW    ALLOCATE                ;SET SIZE OF MESSAGE BUFFER
   00D3    30    0012    104          MOVB    #PIB$K_ARQ,PIB$B_TYPE(R0) ;ALLOCATE MESSAGE BUFFER
60    04    90    0015    105          BRW     COMMON                  ;INSERT MESSAGE TYPE
   00BF    31    0018    106                                          ;
```

```
                                                C 2
                     001B   107          .SBTTL  END OF I/O TRANSACTION
                     001B   108 ;+
                     001B   109 ; PMS$END_IO - END OF I/O TRANSACTION
                     001B   110 ;
                     001B   111 ; THIS ROUTINE IS CALLED AT THE END OF AN I/O TRANSACTION TO ENTER A MESSAGE
                     001B   112 ; IN THE I/O PERFORMANCE DATA COLLECTION BUFFER.
                     001B   113 ;
                     001B   114 ; INPUTS:
                     001B   115 ;
                     001B   116 ;       R3 = ADDRESS OF I/O REQUEST PACKET.
                     001B   117 ;
                     001B   118 ; OUTPUTS:
                     001B   119 ;
                     001B   120 ;       IF I/O PERFORMANCE DATA COLLECTION IS ENABLED AND THE CURRENT I/O PACKET
                     001B   121 ;       CONFORMS TO THE SELECTION CRITERIA, THEN AN END OF I/O TRANSACTION IS
                     001B   122 ;       PLACED IN THE CURRENT DATA BUFFER.
                     001B   123 ;
                     001B   124 ;       ALL REGISTERS ARE PRESERVED ACROSS CALL.
                     001B   125 ;-
                     001B   126
                     001B   127 PMS$END_IO::                                ;END OF I/O TRANSACTION
0000'CF    D5        001B   128          TSTL    W^PMS$GL_IOPFMPDB         ;DATA COLLECTION ENABLED?
      01   12        001F   129          BNEQ    5$                        ;BR IF YES
           05        0021   130          RSB                               ;ELSE RETURN
                     0022   131 5$:      SAVIPL                            ;STORE OLD IPL ON STACK
      0F   BB        0025   132          PUSHR   #^M<R0,R1,R2,R3>          ;SAVE REGISTERS
51    18   9A        0027   133          MOVZBL  #PIB$K_EIO_SIZE,R1        ;SET SIZE OF MESSAGE BUFFER
                     002A   134          SETIPL  #IPL$_PERFMON             ;DISABLE SOFTWARE INTERRUPTS
    00B8   30        002D   135          BSBW    ALLOCATE                  ;ALLOCATE MESSAGE BUFFER
60    02   90        0030   136          MOVB    #PIB$K_EIO,PIB$B_TYPE(R0) ;INSERT MESSAGE TYPE
10 A0 38 A3   7D     0033   137          MOVQ    IRP$L_MEDIA(R3),PIB$Q_EIO_IOSB(R0) ;INSERT FINAL I/O STATUS
      009F   31        0038   138          BRW     COMMON                    ;
```

```
                     003B   140          .SBTTL  END OF I/O REQUEST
                     003B   141 ;+
                     003B   142 ; PMS$END_RQ - END OF I/O REQUEST
                     003B   143 ;
                     003B   144 ; THIS ROUTINE IS CALLED AT THE END OF AN I/O REQUEST TO ENTER A MESSAGE IN THE
                     003B   145 ; I/O PERFORMANCE DATA COLLECTION BUFFER.
                     003B   146 ;
                     003B   147 ; INPUTS:
                     003B   148 ;
                     003B   149 ;       R5 = ADDRESS OF I/O REQUEST PACKET.
                     003B   150 ;
                     003B   151 ;       ROUTINE IS ASSUMED TO HAVE BEEN CALLED FROM I/O POST AT I/O POST
                     003B   152 ;       LEVEL.
                     003B   153 ;
                     003B   154 ; OUTPUTS:
                     003B   155 ;
                     003B   156 ;       IF I/O PERFORMANCE DATA COLLECTION IS ENABLED AND THE CURRENT I/O PACKET
                     003B   157 ;       CONFORMS TO THE SELECTION CRITERIA, THEN AN END OF I/O REQUEST TRANSACTION
                     003B   158 ;       IS PLACED IN THE CURRENT DATA BUFFER.
                     003B   159 ;
                     003B   160 ;       IF ANY BUFFERS ARE READY TO PROCESS, THEN THE I/O PERFORMANCE DATA
                     003B   161 ;       COLLECTION PROCESS IS AWAKENED.
                     003B   162 ;
                     003B   163 ;       ALL REGISTERS ARE PRESERVED ACROSS CALL.
                     003B   164 ;-
                     003B   165
                     003B   166 PMS$END_RQ::                                ;END OF I/O REQUEST
0000'CF    D5        003B   167          TSTL    W^PMS$GL_IOPFMPDB          ;DATA COLLECTION ENABLED?
      01   12        003F   168          BNEQ    5$                         ;BR IF YES
           05        0041   169          RSB                                ;ELSE RETURN
                     0042   170 5$:      SAVIPL                             ;STORE OLD IPL ON STACK
           0F   BB   0045   171          PUSHR   #^M<R0,R1,R2,R3>           ;SAVE REGISTERS
      53   55   D0   0047   172          MOVL    R5,R3                      ;COPY ADDRESS OF I/O PACKET
      51   10   9A   004A   173          MOVZBL  #PIB$K_ERQ_SIZE,R1         ;SET SIZE OF MESSAGE BUFFER
                     004D   174          SETIPL  #IPL$_PERFMON              ;DISABLE SOFTWARE INTERRUPTS
         0095   30   0050   175          BSBW    ALLOCATE                   ;ALLOCATE MESSAGE BUFFER
      60   03   90   0053   176          MOVB    #PIB$K_ERQ,PIB$B_TYPE(R0)  ;INSERT MESSAGE TYPE
        2A A2   B5   0056   177          TSTW    PDB$W_BUFCNT(R2)           ;ANY FULL BUFFERS?
           0D   13   0059   178          BEQL    10$                        ;IF EQL NO
      7E   54   7D   005B   179          MOVQ    R4,-(SP)                   ;SAVE REGISTERS
51    20 A2   D0   005E   180          MOVL    PDB$L_PID(R2),R1           ;GET DATA COLLECTION PROCESS INDENTIFICATION
         FF9B' 30   0062   181          BSBW    SCH$WAKE                   ;WAKE PROCESS
      54   8E   7D   0065   182          MOVQ    (SP)+,R4                   ;RESTORE REGISTERS
           70   11   0068   183 10$:     BRB     COMMON                     ;
```

```
                     006A   185          .SBTTL  START OF I/O TRANSACTION
                     006A   186 ;+
                     006A   187 ; PMS$START_IO - START OF I/O TRANSACTION
                     006A   188 ;
                     006A   189 ; THIS ROUTINE IS CALLED AT THE START OF AN I/O TRANSACTION TO ENTER A MESSAGE
                     006A   190 ; IN THE CURRENT I/O PERFORMANCE DATA COLLECTION BUFFER.
                     006A   191 ;
                     006A   192 ; INPUTS:
                     006A   193 ;
                     006A   194 ;       R3 = ADDRESS OF I/O REQUEST PACKET.
                     006A   195 ;
                     006A   196 ; OUTPUTS:
                     006A   197 ;
                     006A   198 ;       IF I/O PERFORMANCE DATA COLLECTION IS ENABLED AND THE CURRENT I/O PACKET
                     006A   199 ;       CONFORMS TO THE SELECTION CRITERIA, THEN AN START OF I/O TRANSACTION IS
                     006A   200 ;       PLACED IN THE CURRENT DATA BUFFER.
                     006A   201 ;
                     006A   202 ;       ALL REGISTERS ARE PRESERVED ACROSS CALL.
                     006A   203 ;-
                     006A   204
                     006A   205 PMS$START_IO::                          ;START OF I/O TRANSACTION
          0000'CF    D5  006A   206          TSTL    W^PMS$GL_IOPFMPDB   ;DATA COLLECTION ENABLED?
                01   12  006E   207          BNEQ    5$                  ;BR IF YES
                     05  0070   208          RSB                         ;ELSE RETURN
                         0071   209 5$:      SAVIPL                      ;STORE OLD IPL ON STACK
                0F   BB  0074   210          PUSHR   #^M<R0,R1,R2,R3>    ;SAVE REGISTERS
          51    18   9A  0076   211          MOVZBL  #PIB$K_SIO_SIZE,R1  ;SET SIZE OF MESSAGE BUFFER
                         0079   212          SETIPL  #IPL$_PERFMON       ;DISABLE SOFTWARE INTERRUPTS
                6A   10  007C   213          BSBB    ALLOCATE            ;ALLOCATE MESSAGE BUFFER
          60    01   90  007E   214          MOVB    #PIB$K_SIO,PIB$B_TYPE(R0) ;INSERT MESSAGE TYPE
    14 A0   32   A3   D0  0081   215          MOVL    IRP$L_BCNT(R3),PIB$L_SIO_BCNT(R0) ;INSERT TRANSFER BYTE COUNT
    10 A0   38   A3   D0  0086   216          MOVL    IRP$L_MEDIA(R3),PIB$L_SIO_MEDIA(R0) ;INSERT MEDIA ADDRESS
                4D   11  008B   217          BRB     COMMON              ;
```

```
                                                F 2
                     008D   219          .SBTTL  START OF I/O REQUEST
                     008D   220 ;+
                     008D   221 ; PMS$START_RQ - START OF I/O REQUEST
                     008D   222 ;
                     008D   223 ; THIS ROUTINE IS CALLED AT THE START OF AN I/O REQUEST TO ENTER A MESSAGE IN THE
                     008D   224 ; I/O PERFORMANCE DATA COLLECTION BUFFER.
                     008D   225 ;
                     008D   226 ; INPUTS:
                     008D   227 ;
                     008D   228 ;       R3 = ADDRESS OF I/O REQUEST PACKET.
                     008D   229 ;
                     008D   230 ; OUTPUTS:
                     008D   231 ;
                     008D   232 ;       IF I/O PERFORMANCE DATA COLLECTION IS ENABLED AND THE CURRENT I/O PACKET
                     008D   233 ;       CONFORMS TO THE SELECTION CRITERIA, THEN AN START OF I/O REQUEST TRANSACTION
                     008D   234 ;       IS PLACED IN THE CURRENT DATA BUFFER.
                     008D   235 ;
                     008D   236 ;       ALL REGISTERS ARE PRESERVED ACROSS CALL.
                     008D   237 ;-
                     008D   238
                     008D   239 PMS$START_RQ::                                  ;START OF I/O REQUEST
                     008D   240          DSBINT  #IPL$_PERFMON                  ;DISABLE SOFTWARE INTERRUPTS
     50 A3  0000'CF  D0  0093   241          MOVL    W^PMS$GL_IOPFMSEQ,IRP$L_SEQNUM(R3) ;INSERT PACKET SEQUENCE NUMBER
            0000'CF  D6  0099   242          INCL    W^PMS$GL_IOPFMSEQ              ;INCREMENT I/O TRANSACTION SEQUENCE NUMBER
            0000'CF  D5  009D   243          TSTL    W^PMS$GL_IOPFMPDB              ;DATA COLLECTION ENABLED?
                 04  12  00A1   244          BNEQ    5$                             ;BR IF YES
                     00A3   245          ENBINT                                 ;ENABLE INTERRUPTS
                 05  00A6   246          RSB                                    ;AND RETURN
                 0F  BB  00A7   247 5$:       PUSHR   #^M<R0,R1,R2,R3>               ;SAVE REGISTERS
           51    20  9A  00A9   248          MOVZBL  #PIB$K_SRQ_SIZE,R1             ;SET SIZE OF MESSAGE BUFFER
                 3A  10  00AC   249          BSBB    ALLOCATE                       ;ALLOCATE MESSAGE BUFFER
           60    00  90  00AE   250          MOVB    #PIB$K_SRQ,PIB$B_TYPE(R0)      ;INSERT MESSAGE TYPE
                     00B1   251          .ENABLE         LSB
        52   18  A3  D0  00B1   252          MOVL    IRP$L_WIND(R3),R2              ; R2 <= WCB address
                 29  18  00B5   253          BGEQ    20$                            ; BR if not SVA
02 A0    14  A2  B0  00B7   254          MOVW    WCB$W_ACON(R2),PIB$W_SRQ_ACON(R0) ; Insert file access
1C A0    0B  A2  90  00BC   255          MOVB    WCB$B_ACCESS(R2),PIB$B_SRQ_ACCESS(R0) ; information from WCB
01 A0    23  A3  90  00C1   256 10$:      MOVB    IRP$B_PRI(R3),PIB$B_SRQ_PRI(R0) ;INSERT REQUEST PRIORITY
10 A0    0C  A3  D0  00C6   257          MOVL    IRP$L_PID(R3),PIB$L_SRQ_PID(R0) ;INSERT REQUESTER PID
14 A0    1C  A3  D0  00CB   258          MOVL    IRP$L_UCB(R3),PIB$L_SRQ_UCB(R0) ;INSERT DEVICE UCB ADDRESS
18 A0    20  A3  B0  00D0   259          MOVW    IRP$W_FUNC(R3),PIB$W_SRQ_FUNC(R0) ;INSERT I/O FUNCTION CODE
1A A0    2A  A3  B0  00D5   260          MOVW    IRP$W_STS(R3),PIB$W_SRQ_STS(R0) ;INSERT PACKET STATUS WORD
                 0F  BA  00DA   261 COMMON:   POPR    #^M<R0,R1,R2,R3>               ;RESTORE REGISTERS
                     00DC   262          ENBINT                                 ;ENABLE INTERRUPTS
                 05  00DF   263          RSB                                    ;
                     00E0   264
              02 A0  B4  00E0   265 20$:      CLRW    PIB$W_SRQ_ACON(R0)             ; Zero ACON and ACCESS fields
              1C A0  94  00E3   266          CLRB    PIB$B_SRQ_ACCESS(R0)           ; to indicate no WCB available
                 D9  11  00E6   267          BRB     10$                            ; Rejoin mainline code
                     00E8   268          .DISABLE        LSB
```

```
                     00E8   270          .SBTTL  ALLOCATE MESSAGE BUFFER
                     00E8   271 ;
                     00E8   272 ; ALLOCATE - ALLOCATE MESSAGE BUFFER
                     00E8   273 ;
                     00E8   274 ; THIS ROUTINE IS CALLED TO:
                     00E8   275 ;
                     00E8   276 ;       1. TEST IF I/O PERFORMANCE MEASUREMENT IS ENABLED,
                     00E8   277 ;       2. TEST IF THE DEVICE CLASS MATCHES,
                     00E8   278 ;       3. TEST IF THE DEVICE TYPE MATCHES,
                     00E8   279 ;       4. TEST IF THE FUNCTION CODE IS TO BE RECORDED, AND
                     00E8   280 ;       5. TEST IF THE STATUS FIELD MATCHES.
                     00E8   281 ;
                     00E8   282 ; IF ALL OF THESE TESTS ARE SUCCESSFUL, THEN AN ATTEMPT IS MADE TO ALLOCATE
                     00E8   283 ; AN I/O PERFORMANCE MEASUREMENT MESSAGE BUFFER.
                     00E8   284 ;
                     00E8   285 ; INPUTS:
                     00E8   286 ;
                     00E8   287 ;       R1 = SIZE OF MESSAGE TO ALLOCATE.
                     00E8   288 ;       R3 = ADDRESS OF I/O REQUEST PACKET.
                     00E8   289 ;
                     00E8   290 ;       REGISTERS R0, R1, R2, AND R3 ARE ASSUMED TO HAVE BEEN SAVED AND ALL
                     00E8   291 ;       SOFTWARE INTERRUPTS ARE DISABLED.
                     00E8   292 ;
                     00E8   293 ; OUTPUTS:
                     00E8   294 ;
                     00E8   295 ;       IF I/O PERFORMANCE MEASUREMENT IS NOT ENABLED OR ANY OF THE SELECTION
                     00E8   296 ;       CRITERIA FAILS, THEN THE RETURN ADDRESS IS REMOVED FROM THE STACK, THE
                     00E8   297 ;       INTERRUPT LEVEL AND REGISTERS ARE RESTORED, AND A RETURN TO THE ORIGINAL
                     00E8   298 ;       CALLER IS EXECUTED. ELSE AN ATTEMPT IS MADE TO ALLOCATE AN I/O PERFORM-
                     00E8   299 ;       ANCE MEASUREMENT MESSAGE BUFFER.
                     00E8   300 ;
                     00E8   301 ;       IF THE ALLOCATION ATTEMPT FAILS, THEN THE RETURN ADDRESS IS REMOVED FROM
                     00E8   302 ;       THE STACK, THE INTERRUPT LEVEL AND REGISTERS ARE RESTORED, AND A RETURN
                     00E8   303 ;       TO THE ORIGINAL CALLER IS EXECUTED. ELSE THE MESSAGE BUFFER IS ALLOCATED,
                     00E8   304 ;       THE TIME AND SEQUENCE NUMBER ARE FILLED IN, AND A RETURN TO THE CALLER
                     00E8   305 ;       IS EXECUTED.
                     00E8   306 ;
                     00E8   307
                     00E8   308 ALLOCATE:                                    ;ALLOCATE MESSAGE BUFFER
      52   0000'CF   D0   00E8   309          MOVL    W^PMS$GL_IOPFMPDB,R2    ;GET ADDRESS OF I/O PERFORMANCE DATA BLOCK
                6C   13   00ED   310          BEQL    60$                     ;IF EQL I/O PERFORMANCE MEASUREMENT DISABLED
         50    1C A3   D0   00EF   311          MOVL    IRP$L_UCB(R3),R0        ;GET ADDRESS OF DEVICE UCB
               24 A2   95   00F3   312          TSTB    PDB$B_DEVCLASS(R2)      ;ALL CLASSES MATCH?
                07   19   00F6   313          BLSS    10$                     ;IF LSS YES
      24 A2    40 A0   91   00F8   314          CMPB    UCB$B_DEVCLASS(R0),PDB$B_DEVCLASS(R2) ;DEVICE CLASS MATCH?
                5C   12   00FD   315          BNEQ    60$                     ;IF NEQ NO
               25 A2   95   00FF   316 10$:     TSTB    PDB$B_DEVTYPE(R2)       ;ALL TYPES MATCH?
                07   19   0102   317          BLSS    20$                     ;IF LSS YES
      25 A2    41 A0   91   0104   318          CMPB    UCB$B_DEVTYPE(R0),PDB$B_DEVTYPE(R2) ;DEVICE TYPE MATCH?
                50   12   0109   319          BNEQ    60$                     ;IF NEQ NO
            06   00   EF   010B   320 20$:     EXTZV   #IRP$V_FCODE,#IRP$S_FCODE,- ;GET I/O FUNCTION CODE
         50    20 A3        010E   321                  IRP$W_FUNC(R3),R0       ;
      45 2C A2    50   E1   0111   322          BBC     R0,PDB$Q_FUNC(R2),60$   ;IF CLR, FUNCTION NOT ENABLED
50    2A A3    26 A2   AB   0116   323          BICW3   PDB$W_ANDM(R2),IRP$W_STS(R3),R0 ;CLEAR MISCELLANEOUS BITS
         50    28 A2   AC   011C   324          XORW    PDB$W_XORM(R2),R0       ;TEST FOR EQUALITY
                39   12   0120   325          BNEQ    60$                     ;IF NEQ NO MATCH
               14 A2   D5   0122   326 30$:     TSTL    PDB$L_CURBUF(R2)        ;ANY BUFFER CURRENTLY AVAILABLE?
```

```
                16   13  0125   327          BEQL    40$                             ;IF EQL NO
50   18 A2      51   C1  0127   328          ADDL3   R1,PDB$L_NXTBUF(R2),R0          ;CALCULATE ADDRESS OF FREE SPACE
     1C A2      50   D1  012C   329          CMPL    R0,PDB$L_ENDBUF(R2)             ;BEYOND END OF BUFFER?
                31   1B  0130   330          BLEQU   70$                             ;IF LEQU NO
  10 B2    14 B2   0E  0132   331          INSQUE  @PDB$L_CURBUF(R2),@PDB$L_FILLBL(R2) ;INSERT BUFFER IN FILLED LIST
           2A A2   B6  0137   332          INCW    PDB$W_BUFCNT(R2)                ;INCREMENT FULL BUFFER COUNT
           14 A2   D4  013A   333          CLRL    PDB$L_CURBUF(R2)                ;CLEAR ADDRESS OF CURRENT BUFFER
     50    00 B2   0F  013D   334 40$:     REMQUE  @PDB$L_FREEFL(R2),R0            ;GET NEXT BUFFER FROM FREE LIST
                14   1D  0141   335          BVS     50$                             ;IF VS NO BUFFER AVAILABLE
     14 A2      50   D0  0143   336          MOVL    R0,PDB$L_CURBUF(R2)             ;SET ADDRESS OF AVAILABLE BUFFER
  18 A2      0D A0   9E  0147   337          MOVAB   PBH$K_START(R0),PDB$L_NXTBUF(R2) ;SET ADDRESS OF FREE SPACE
     7E      08 A0   3C  014C   338          MOVZWL  PBH$W_SIZE(R0),-(SP)            ;GET LENGTH OF DATA BUFFER
   1C A2      9E40   9E  0150   339          MOVAB   @(SP)+[R0],PDB$L_ENDBUF(R2) ;SET ENDING ADDRESS OF BUFFER
                CB   11  0155   340          BRB     30$                             ;
     0B A2      01   88  0157   341 50$:     BISB    #1,PDB$B_OVERRUN(R2)            ;SET OVERRUN INDICATOR
                8E   D5  015B   342 60$:     TSTL    (SP)+                           ;REMOVED RETURN FROM STACK
                0F   BA  015D   343          POPR    #^M<R0,R1,R2,R3>                ;RESTORE REGISTERS
                     015F   344          ENBINT                                  ;ENABLE INTERRUPTS
                05   0162   345          RSB                                     ;
                     0163   346
                     0163   347 ;
                     0163   348 ; MESSAGE BUFFER SUCCESSFULLY ALLOCATED
                     0163   349 ;
                     0163   350
     50    18 A2   D0  0163   351 70$:     MOVL    PDB$L_NXTBUF(R2),R0             ;GET ADDRESS OF NEXT SPACE IN BUFFER
18 A2      51   50   C1  0167   352          ADDL3   R0,R1,PDB$L_NXTBUF(R2)          ;CALCULATE ADDRESS OF NEXT AVAILABLE SPACE
     51    14 A2   D0  016C   353          MOVL    PDB$L_CURBUF(R2),R1             ;GET ADDRESS OF CURRENT BUFFER
           0B A1   B6  0170   354          INCW    PBH$W_MSGCNT(R1)                ;INCREMENT MESSAGE COUNT
04 A0    0000'CF   7D  0173   355          MOVQ    W^EXE$GQ_SYSTIME,PIB$Q_SRQ_TIME(R0) ;INSERT CURRENT TIME
  0C A0    50 A3   D0  0179   356          MOVL    IRP$L_SEQNUM(R3),PIB$L_SRQ_SEQN(R0) ;INSERT SEQUENCE NUMBER
                05   017E   357          RSB                                     ;
                     017F   358
                     017F   359          .ENDC
                     017F   360
                     017F   361          .END
```

```
ALLOCATE                    000000E8 R      02
COMMON                      000000DA R      02
EXE$GQ_SYSTIME              ******** X      02
IPL$_PERFMON              = 0000000F
IRP$B_PRI                 = 00000023
IRP$L_BCNT                = 00000032
IRP$L_MEDIA               = 00000038
IRP$L_PID                 = 0000000C
IRP$L_SEQNUM              = 00000050
IRP$L_UCB                 = 0000001C
IRP$L_WIND                = 00000018
IRP$S_FCODE               = 00000006
IRP$V_FCODE               = 00000000
IRP$W_FUNC                = 00000020
IRP$W_STS                 = 0000002A
PBH$K_START               = 0000000D
PBH$W_MSGCNT              = 0000000B
PBH$W_SIZE                = 00000008
PDB$B_DEVCLASS            = 00000024
PDB$B_DEVTYPE             = 00000025
PDB$B_OVERRUN             = 0000000B
PDB$L_CURBUF              = 00000014
PDB$L_ENDBUF              = 0000001C
PDB$L_FILLBL              = 00000010
PDB$L_FREEFL              = 00000000
PDB$L_NXTBUF              = 00000018
PDB$L_PID                 = 00000020
PDB$Q_FUNC                = 0000002C
PDB$W_ANDM                = 00000026
PDB$W_BUFCNT              = 0000002A
PDB$W_XORM                = 00000028
PIB$B_SRQ_ACCESS          = 0000001C
PIB$B_SRQ_PRI             = 00000001
PIB$B_TYPE                = 00000000
PIB$K_ARQ                 = 00000004
PIB$K_ARQ_SIZE            = 00000010
PIB$K_EIO                 = 00000002
PIB$K_EIO_SIZE            = 00000018
PIB$K_ERQ                 = 00000003
PIB$K_ERQ_SIZE            = 00000010
PIB$K_SIO                 = 00000001
PIB$K_SIO_SIZE            = 00000018
PIB$K_SRQ                 = 00000000
PIB$K_SRQ_SIZE            = 00000020
PIB$L_SIO_BCNT            = 00000014
PIB$L_SIO_MEDIA           = 00000010
PIB$L_SRQ_PID             = 00000010
PIB$L_SRQ_SEQN            = 0000000C
PIB$L_SRQ_UCB             = 00000014
PIB$Q_EIO_IOSB            = 00000010
PIB$Q_SRQ_TIME            = 00000004
PIB$W_SRQ_ACON            = 00000002
PIB$W_SRQ_FUNC            = 00000018
PIB$W_SRQ_STS             = 0000001A
PMS$ABORT_RQ                00000000 RG     02
PMS$END_IO                  0000001B RG     02
PMS$END_RQ                  0000003B RG     02
PMS$GL_IOPFMPDB             ******** X      02
PMS$GL_IOPFMSEQ             ******** X      02
PMS$START_IO                0000006A RG     02
PMS$START_RQ                0000008D RG     02
PR$_IPL                     ******** X      02
SCH$WAKE                    ******** X      02
UCB$B_DEVCLASS            = 00000040
UCB$B_DEVTYPE             = 00000041
WCB$B_ACCESS              = 0000000B
WCB$W_ACON                = 00000014
```

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
$AEXENONPAGED	0000017F (383.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	35	00:00:00.10	00:00:01.16
Command processing	131	00:00:00.51	00:00:02.77
Pass 1	253	00:00:06.89	00:00:22.63
Symbol table sort	0	00:00:01.01	00:00:02.57
Pass 2	84	00:00:01.43	00:00:04.61
Symbol table output	9	00:00:00.08	00:00:00.30
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	516	00:00:10.06	00:00:34.10

The working set limit was 1350 pages.
38553 bytes (76 pages) of virtual memory were used to buffer the intermediate code.
There were 40 pages of symbol table space allocated to hold 672 non-local and 15 local symbols.
361 source lines were read in Pass 1, producing 14 object records in Pass 2.
19 pages of virtual memory were used to define 18 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYS.OBJ]LIB.MLB;1	12
_255DUA28:[SYSLIB]STARLET.MLB;2	3
TOTALS (all libraries)	15

776 GETS were required to define 15 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:IOPERFORM/OBJ=OBJ$:IOPERFORM MSRC$:IOPERFORM/UPDATE=(ENH$:IOPERFORM)+EXECML$/LIB
```

IOSUBRAMS
LIS

IPCONTROL
LIS

IOSUBPAGD
LIS

LNMSUB
LIS

IOPERFORM
LIS

IOSUBNPAG
LIS

LINKVEC
LIS