```
DDDDDDDDDDDD     RRRRRRRRRRRR        IIIIIIIII     VVV         VVV  EEEEEEEEEEEEEEE  RRRRRRRRRRRR
DDDDDDDDDDDD     RRRRRRRRRRRR        IIIIIIIII     VVV         VVV  EEEEEEEEEEEEEEE  RRRRRRRRRRRR
DDDDDDDDDDDD     RRRRRRRRRRRR        IIIIIIIII     VVV         VVV  EEEEEEEEEEEEEEE  RRRRRRRRRRRR
DDD          DDD RRR          RRR       III      VVV         VVV  EEE              RRR          RRR
DDD          DDD RRR          RRR       III      VVV         VVV  EEE              RRR          RRR
DDD          DDD RRR          RRR       III      VVV         VVV  EEE              RRR          RRR
DDD          DDD RRR          RRR       III      VVV         VVV  EEE              RRR          RRR
DDD          DDD RRR          RRR       III      VVV         VVV  EEE              RRR          RRR
DDD          DDD RRRRRRRRRRRR           III      VVV         VVV  EEEEEEEEEEEE     RRRRRRRRRRRR
DDD          DDD RRRRRRRRRRRR           III      VVV         VVV  EEEEEEEEEEEE     RRRRRRRRRRRR
DDD          DDD RRRRRRRRRRRR           III      VVV         VVV  EEEEEEEEEEEE     RRRRRRRRRRRR
DDD          DDD RRR   RRR              III      VVV         VVV  EEE              RRR   RRR
DDD          DDD RRR   RRR              III      VVV         VVV  EEE              RRR   RRR
DDD          DDD RRR   RRR              III      VVV         VVV  EEE              RRR   RRR
DDD          DDD RRR       RRR          III        VVV     VVV    EEE              RRR       RRR
DDD          DDD RRR       RRR          III        VVV     VVV    EEE              RRR       RRR
DDD          DDD RRR       RRR          III        VVV     VVV    EEE              RRR       RRR
DDDDDDDDDDDD     RRR          RRR    IIIIIIIII       VVV         EEEEEEEEEEEEEEE  RRR          RRR
DDDDDDDDDDDD     RRR          RRR    IIIIIIIII       VVV         EEEEEEEEEEEEEEE  RRR          RRR
DDDDDDDDDDDD     RRR          RRR    IIIIIIIII       VVV         EEEEEEEEEEEEEEE  RRR          RRR
```

```
PPPPPPPP      AAAAAA    MM      MM    000000    NN      NN    IIIIII    TTTTTTTTTT
PPPPPPPP      AAAAAA    MM      MM    000000    NN      NN    IIIIII    TTTTTTTTTT
PP      PP   AA    AA   MMMM  MMMM   00    00   NN      NN      II        TT
PP      PP   AA    AA   MMMM  MMMM   00    00   NN      NN      II        TT
PP      PP   AA    AA   MM  MM  MM   00    00   NNNN    NN      II        TT
PP      PP   AA    AA   MM  MM  MM   00    00   NNNN    NN      II        TT
PPPPPPPP     AA    AA   MM      MM   00    00   NN  NN  NN      II        TT
PPPPPPPP     AA    AA   MM      MM   00    00   NN  NN  NN      II        TT
PP           AAAAAAAAAA MM      MM   00    00   NN    NNNN      II        TT
PP           AAAAAAAAAA MM      MM   00    00   NN    NNNN      II        TT
PP           AA    AA   MM      MM   00    00   NN      NN      II        TT          ....
PP           AA    AA   MM      MM   00    00   NN      NN      II        TT          ....
PP           AA    AA   MM      MM    000000    NN      NN    IIIIII      TT          ....
PP           AA    AA   MM      MM    000000    NN      NN    IIIIII      TT          ....

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

Table of contents

```
0000     1          .TITLE  PAMONIT
0000     2          .IDENT  'V04-000'
0000     3
0000     4  ;****************************************************************************
0000     5  ;*                                                                          *
0000     6  ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000     7  ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     8  ;*  ALL RIGHTS RESERVED.                                                    *
0000     9  ;*                                                                          *
0000    10  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000    11  ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    12  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0000    13  ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000    14  ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0000    15  ;*  TRANSFERRED.                                                            *
0000    16  ;*                                                                          *
0000    17  ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0000    18  ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
0000    19  ;*  CORPORATION.                                                            *
0000    20  ;*                                                                          *
0000    21  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0000    22  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    23  ;*                                                                          *
0000    24  ;*                                                                          *
0000    25  ;****************************************************************************
0000    26  ;
0000    27  ;++
0000    28  ;
0000    29  ; FACILITY:
0000    30  ;
0000    31  ;       VAX/VMS EXECUTIVE, I/O DRIVERS
0000    32  ;
0000    33  ; ABSTRACT:
0000    34  ;
0000    35  ; AUTHOR:  N. KRONENBERG,  MAY 1981
0000    36  ;
0000    37  ; MODIFIED BY:
0000    38  ;
0000    39  ;       V03-004 NPK3029         N. Kronenberg           22-Jul-1983
0000    40  ;               Eliminate copy of interval count register (for time
0000    41  ;               stamp) in trace buffer entries in favor of logging
0000    42  ;               PDT address.
0000    43  ;
0000    44  ;       V03-003 NPK3024         N. Kronenberg           19-Apr-1983
0000    45  ;               Modified queue checker to check header soft interlock
0000    46  ;               on every element check and to put a code into R1
0000    47  ;               to show time of failure for other than maximum number
0000    48  ;               of entries found on queue.  Codes are: -1/-2/-3 for
0000    49  ;               back link to previous entry broken/structure type
0000    50  ;               wrong/soft link cleared respectively.
0000    51  ;               Increased maximum number of queue entries tolerated
0000    52  ;               to 64.
0000    53  ;               Removed LRP identification and address checks to allow
0000    54  ;               variable network header sizes.
0000    55  ;
0000    56  ;       V03-002 NPK2016         N. Kronenberg           18-Mar-1982
0000    57  ;               Fixed .TITLE
```

```
                        ;
                        ;--
                        
00000000                .PSECT  $$$115_DRIVER,LONG
                        
                        
                        $DYNDEF
                        $PDTDEF
                        $PAPDTDEF
                        $PAREGDEF
                        $PPDDEF
```

```
                     0000     76          .SBTTL  INTERLOCKED QUEUE MONITOR ROUTINES
                     0000     77          .SBTTL  QUEUE MONITOR CONTROL FLAGS
                     0000     78
                     0000     79 ;+
                     0000     80 ; MON$FLAGS is a bit mask of control flags.
                     0000     81 ;-
                     0000     82
                     0000     83 MON$FLAGS::
                     0000     84
00000001             0000     85          .LONG   1                       ; Default is queue checks disabled
                     0004     86
                     0004     87 ;
                     0004     88 ; Flag definitions:
                     0004     89 ;
                     0004     90
00000001             0004     91 MON$M_QCHK = 1
00000000             0004     92 MON$V_QCHK = 0
```

```
                        .SBTTL  -       CHKQ MACRO AND CONTROL
                        .SBTTL  -       FLAGS LONGWORD

                ;+
                ; Macro CHKQ generates inline code for checking a relative queue,
                ;
                ; Inputs:
                ;
                ;       R3                      -Addr of Q header or entry
                ;
                ; Outputs:
                ;
                ;       R0-R2,R5                        -Destroyed
                ;-

                        .MACRO  CHKQ,?LOOP,?ERR,?OK,?LOCK,?LOWERIPL,?TYPOK,?ERR1,?ERR2,?ERR3

                         MOVL   R3,R2                   ; Get copy of listhd addr
                         CLRL   R1                      ; Zero entry counter
                         DSBINT                         ; Disable interrupts

                LOCK:
                         BBSSI  #0,(R3),LOCK            ; Lock queue before reading

                LOOP:
                         MOVL   R2,R5                   ; Save addr of this entry

                         MOVL   (R2),R0                 ; Get offset to next entry
                         BICL   #1,R0                   ; Clear interlock bit
                         MOVAB  (R2)[R0],R2             ; Get addr of next entry
                         MOVL   4(R2),R0                ; Get back link from this entry
                         MOVAB  (R2)[R0],R0             ; Compute prev entry addr
                         CMPL   R0,R5                   ; Computed addr = saved?
                         BNEQ   ERR1                    ; Branch if not
                         CMPL   R2,R3                   ; Back at start?
                         BEQL   OK                      ; Branch if so
                         CMPB   PPD$B_TYPE(R2),#DYN$C_CIDG  ; CI dg?
                         BEQL   TYPOK                   ; Branch if so
                         CMPB   PPD$B_TYPE(R2),#DYN$C_CIMSG ; CI msg?
                         BNEQ   ERR2                    ; Branch if not
                TYPOK:   BBC    #0,(R3),ERR3           ; Branch if somebody grabbed soft
                                                       ;  interlock while we had it
                         AOBLSS #63,R1,LOOP             ; Else check max count and continue
                         BRB    ERR                     ; Branch if max count expired

                ERR1:    MNEGL  #1,R1                   ; Set error code to bad blink
                         BRB    ERR                     ; Join common error handling

                ERR2:    MNEGL  #2,R1                   ; Set error code to bad struc type
                         BRB    ERR                     ; Join common error handling

                ERR3:    MNEGL  #3,R1                   ; Set error code to broken soft lock

                ERR:
                         MOVL   #1,@PDT$L_PMC(R4)       ; Min port to prevent further queue
                                                        ;  operations by port
                         BUGCHECK  BADQHDR              ; Notify debugger
```

```
0004    151
0004    152 OK:
0004    153                 BBCCI   #0,(R3),LOWERIPL                ; Check succeeded
0004    154                                                         ; Unlock queue for port
0004    155 LOWERIPL:
0004    156                 ENBINT                                  ; Enable interrupts again
0004    157
0004    158
0004    159                 .ENDM   CHKQ
0004    160
0004    161
0004    162                 .SHOW
0004    163
0004    164
0004    165
```

```
                     0004   167          .SBTTL  -       MON$CHKQ,       CHECK ALL Q'S ON THE PORT
                     0004   168
                     0004   169 ;+
                     0004   170 ; Checks all port queues (designed to be fast rather than short).
                     0004   171 ;
                     0004   172 ; Inputs:
                     0004   173 ;
                     0004   174 ;       R4                      -Addr of PDT
                     0004   175 ;
                     0004   176 ; Outputs:
                     0004   177 ;
                     0004   178 ;       R0, condition codes     -Destroyed
                     0004   179 ;-
                     0004   180
                     0004   181          .ENABL  LSB
                     0004   182
                     0004   183 MON$CHKQ::
          00   E0   0004   184          BBS     #MON$V_QCHK,-                   ; Branch if checking queues
 03 F8 AF             0006   185                  MON$FLAGS,CHKQ_ALT              ;  is enabled
     021D   31   0009   186          BRW     20$                             ; Else skip whole chedk
                     000C   187
                     000C   188 CHKQ_ALT:                                       ; Entry for MON$CHKQ_POST
       55   DD   000C   189          PUSHL   R5                              ; Save registers...
       53   DD   000E   190          PUSHL   R3                              ;
       52   DD   0010   191          PUSHL   R2                              ;
       51   DD   0012   192          PUSHL   R1                              ;
 53  01E0 C4   0014   193          MOVAL   PDT$Q_COMQL(R4),R3              ; Get addr of 1st command Q
                     0019   194          CHKQ                            ; Verify
                     0019
       52   53   D0   0019                   MOVL    R3,R2                   ; Get copy of listhd addr
            51   D4   001C                   CLRL    R1                      ; Zero entry counter
                     001E                   DSBINT                          ; Disable interrupts
                     001E                           .IF B
       7E   00'  DB   001E                           MFPR    S^#PR$_IPL,-(SP)
                     0021                           .IFF
                     0021                           MFPR    S^#PR$_IPL,
                     0021                           .ENDC
                     0021                           .IF B
       00'  1F   DA   0021                           MTPR    #31,S^#PR$_IPL
                     0024                           .IFF
                     0024                           MTPR    ,S^#PR$_IPL
                     0024                           .ENDC
                     0024
                     0024
                     0024   30003$:
 FC 63   00   E6   0024                   BBSSI   #0,(R3),30003$          ; 30003$ queue before reading
                     0028
                     0028   30000$:
       55   52   D0   0028                   MOVL    R2,R5                   ; Save addr of this entry
                     002B
       50   62   D0   002B                   MOVL    (R2),R0                 ; Get offset to next entry
       50   01   CA   002E                   BICL    #1,R0                   ; Clear interlock bit
  52   6240   9E   0031                   MOVAB   (R2)[R0],R2             ; Get addr of next entry
50   04 A2   D0   0035                   MOVL    4(R2),R0                ; Get back link from this entry
 50   6240   9E   0039                   MOVAB   (R2)[R0],R0             ; Compute prev entry addr
   55   50   D1   003D                   CMPL    R0,R5                   ; Computed addr = saved?
        1B   12   0040                   BNEQ    30006$                  ; Branch if not
```

```
                53   52   D1  0042                CMPL   R2,R3                      ; Back at start?
                     2F   13  0045                BEQL   30002$                     ; Branch if so
             3B   0A A2   91  0047                CMPB   PPD$B_TYPE(R2),#DYN$C_CIDG  ; CI dg?
                     06   13  004B                BEQL   30005$                     ; Branch if so
             3C   0A A2   91  004D                CMPB   PPD$B_TYPE(R2),#DYN$C_CIMSG ; CI msg?
                     0F   12  0051                BNEQ   30007$                     ; Branch if not
             10 63   00   E1  0053    30005$:     BBC    #0,(R3),30008$             ; Branch if somebody grabbed soft
                                                                                    ;  interlock while we had it
             CD 51   3F   F2  0057                AOBLSS #63,R1,30000$              ; Else check max count and continue
                     0D   11  005B                BRB    30001$                     ; Branch if max count expired
                                  005D
                51   01   CE  005D    30006$:     MNEGL  #1,R1                      ; Set error code to bad blink
                     08   11  0060                BRB    30001$                     ; Join common error handling
                                  0062
                51   02   CE  0062    30007$:     MNEGL  #2,R1                      ; Set error code to bad struc type
                     03   11  0065                BRB    30001$                     ; Join common error handling
                                  0067
                51   03   CE  0067    30008$:     MNEGL  #3,R1                      ; Set error code to broken soft 30003$
                                  006A
                                  006A    30001$:
            00E8 D4   01   D0  006A                MOVL   #1,@PDT$L_PMC(R4)          ; Min port to prevent further queue
                                  006F                                              ;  operations by port
                                  006F                BUGCHECK  BADQHDR             ; Notify debugger
                                  006F                .IF    IDN ,NONFATAL
                                  006F                BSBW   ERR$BUGCHECKNF
                                  006F                BUG_CHECK BADQHDR
                                  006F                .IFF
                    FF8E'  30     006F                BSBW   ERR$BUGCHECK
                                  0072                BUG_CHECK BADQHDR,TYPE=FATAL
                         FEFF     0072                       .WORD   ^XFEFF
                         00[ '    0074                       .IIF IDN <FATAL>,<FATAL> , .WORD        BUG$_BADQHDR!4
                                  0076                       .IIF DIF <FATAL>,<FATAL> , .WORD        BUG$_BADQHDR
                                  0076
                                  0076                .ENDC
                                  0076
                                  0076    30002$:                                   ; Check succeeded
                00 63   00   E7  0076                BBCCI  #0,(R3),30004$          ; Unlock queue for port
                                  007A
                                  007A    30004$:
                                  007A                ENBINT                        ; Enable interrupts again
                                  007A                       .IF B
                    00'  8E   DA  007A                       MTPR   (SP)+,S^#PR$_IPL
                                  007D                       .IFF
                                  007D                       MTPR   ,S^#PR$_IPL
                                  007D                       .ENDC
                                  007D
                                  007D
                                  007D
                                  007D
                53   08   C0  007D    195         ADDL   #8,R3                      ; Step to 2nd command Q
                                  0080    196         CHKQ                          ; Verify
                                  0080
                52   53   D0  0080                MOVL   R3,R2                      ; Get copy of listhd addr
                     51   D4  0083                CLRL   R1                         ; Zero entry counter
                                  0085                DSBINT                        ; Disable interrupts
                                  0085                       .IF B
```

```
                7E   00'  DB  0085                    MFPR    S^#PR$_IPL,-(SP)
                                0088                    .IFF
                                0088                    MFPR    S^#PR$_IPL,
                                0088                    .ENDC
                                0088                    .IF B
                00'  1F   DA    0088                    MTPR    #31,S^#PR$_IPL
                                008B                    .IFF
                                008B                    MTPR    ,S^#PR$_IPL
                                008B                    .ENDC
                                008B
                                008B
                                008B            30012$:
            FC  63   00   E6    008B                    BBSSI   #0,(R3),30012$          ; 30012$ queue before reading
                                008F
                                008F            30009$:
                55   52   D0    008F                    MOVL    R2,R5                   ; Save addr of this entry
                                0092
                50   62   D0    0092                    MOVL    (R2),R0                 ; Get offset to next entry
                50   01   CA    0095                    BICL    #1,R0                   ; Clear interlock bit
              52   6240   9E    0098                    MOVAB   (R2)[R0],R2             ; Get addr of next entry
            50   04 A2   D0    009C                    MOVL    4(R2),R0                ; Get back link from this entry
             50   6240   9E    00A0                    MOVAB   (R2)[R0],R0             ; Compute prev entry addr
                55   50   D1    00A4                    CMPL    R0,R5                   ; Computed addr = saved?
                     1B   12    00A7                    BNEQ    30015$                  ; Branch if not
                53   52   D1    00A9                    CMPL    R2,R3                   ; Back at start?
                     2F   13    00AC                    BEQL    30011$                  ; Branch if so
            3B   0A A2   91    00AE                    CMPB    PPD$B_TYPE(R2),#DYN$C_CIDG  ; CI dg?
                     06   13    00B2                    BEQL    30014$                  ; Branch if so
            3C   0A A2   91    00B4                    CMPB    PPD$B_TYPE(R2),#DYN$C_CIMSG ; CI msg?
                     0F   12    00B8                    BNEQ    30016$                  ; Branch if not
            10 63   00   E1    00BA            30014$: BBC     #0,(R3),30017$          ; Branch if somebody grabbed soft
                                00BE                                                    ;  interlock while we had it
            CD 51   3F   F2    00BE                    AOBLSS  #63,R1,30009$           ; Else check max count and continue
                     0D   11    00C2                    BRB     30010$                  ; Branch if max count expired
                                00C4
                51   01   CE    00C4            30015$: MNEGL   #1,R1                   ; Set error code to bad blink
                     08   11    00C7                    BRB     30010$                  ; Join common error handling
                                00C9
                51   02   CE    00C9            30016$: MNEGL   #2,R1                   ; Set error code to bad struc type
                     03   11    00CC                    BRB     30010$                  ; Join common error handling
                                00CE
                51   03   CE    00CE            30017$: MNEGL   #3,R1                   ; Set error code to broken soft 30012$
                                00D1
                                00D1            30010$:
        00E8 D4   01   D0    00D1                    MOVL    #1,@PDT$L_PMC(R4)       ; Min port to prevent further queue
                                00D6                                                    ;  operations by port
                                00D6                    BUGCHECK  BADQHDR               ; Notify debugger
                                00D6                    .IF     IDN ,NONFATAL
                                00D6                    BSBW    ERR$BUGCHECKNF
                                00D6                    BUG_CHECK BADQHDR
                                00D6                    .IFF
                FF27'  30       00D6                    BSBW    ERR$BUGCHECK
                                00D9                    BUG_CHECK BADQHDR,TYPE=FATAL
                     FEFF       00D9                            .WORD   ^XFEFF
                     0004'      00DB                            .IIF IDN <FATAL>,<FATAL> , .WORD        BUG$_BADQHDR!4
                                00DD                            .IIF DIF <FATAL>,<FATAL> , .WORD        BUG$_BADQHDR
                                00DD
```

```
                     00DD                   .ENDC
                     00DD
                     00DD
                     00DD           30011$:
00 63   00   E7      00DD                   BBCCI   #0,(R3),30013$          ; Check succeeded
                     00E1                                                   ; Unlock queue for port
                     00E1           30013$:
                     00E1                   ENBINT                         ; Enable interrupts again
                     00E1                           .IF B
   00'  8E   DA      00E1                           MTPR    (SP)+,S^#PR$_IPL
                     00E4                           .IFF
                     00E4                           MTPR    ,S^#PR$_IPL
                     00E4                           .ENDC
                     00E4
                     00E4
                     00E4
                     00E4
   53   18   C0      00E4    197            ADDL    #24,R3                  ; Step to response Q
                     00E7    198            CHKQ                            ; Verify
                     00E7
   52   53   D0      00E7                   MOVL    R3,R2                   ; Get copy of listhd addr
        51   D4      00EA                   CLRL    R1                      ; Zero entry counter
                     00EC                   DSBINT                          ; Disable interrupts
                     00EC                           .IF B
   7E   00'  DB      00EC                           MFPR    S^#PR$_IPL,-(SP)
                     00EF                           .IFF
                     00EF                           MFPR    S^#PR$_IPL,
                     00EF                           .ENDC
                     00EF                           .IF B
   00'  1F   DA      00EF                           MTPR    #31,S^#PR$_IPL
                     00F2                           .IFF
                     00F2                           MTPR    ,S^#PR$_IPL
                     00F2                           .ENDC
                     00F2
                     00F2
                     00F2           30021$:
FC 63   00   E6      00F2                   BBSSI   #0,(R3),30021$          ; 30021$ queue before reading
                     00F6
                     00F6           30018$:
   55   52   D0      00F6                   MOVL    R2,R5                   ; Save addr of this entry
                     00F9
   50   62   D0      00F9                   MOVL    (R2),R0                 ; Get offset to next entry
   50   01   CA      00FC                   BICL    #1,R0                   ; Clear interlock bit
 52   6240   9E      00FF                   MOVAB   (R2)[R0],R2             ; Get addr of next entry
50   04 A2   D0      0103                   MOVL    4(R2),R0                ; Get back link from this entry
 50   6240   9E      0107                   MOVAB   (R2)[R0],R0             ; Compute prev entry addr
   55   50   D1      010B                   CMPL    R0,R5                   ; Computed addr = saved?
        1B   12      010E                   BNEQ    30024$                  ; Branch if not
   53   52   D1      0110                   CMPL    R2,R3                   ; Back at start?
        2F   13      0113                   BEQL    30020$                  ; Branch if so
3B   0A A2   91      0115                   CMPB    PPD$B_TYPE(R2),#DYN$C_CIDG  ; CI dg?
        06   13      0119                   BEQL    30023$                      ; Branch if so
3C   0A A2   91      011B                   CMPB    PPD$B_TYPE(R2),#DYN$C_CIMSG ; CI msg?
        0F   12      011F                   BNEQ    30025$                      ; Branch if not
10 63   00   E1      0121    30023$:        BBC     #0,(R3),30026$          ; Branch if somebody grabbed soft
                     0125                                                   ;  interlock while we had it
CD 51   3F   F2      0125                   AOBLSS  #63,R1,30018$           ; Else check max count and continue
```

```
                 OD   11  0129                  BRB     30019$                  ; Branch if max count expired
                                                 .SBTTL  - MON$CHKQ, CHECK ALL Q'S ON THE PORT
                      0129
                      012B
               51 01  CE  012B          30024$: MNEGL   #1,R1                   ; Set error code to bad blink
                  08  11  012E                  BRB     30019$                  ; Join common error handling
                      0130
               51 02  CE  0130          30025$: MNEGL   #2,R1                   ; Set error code to bad struc type
                  03  11  0133                  BRB     30019$                  ; Join common error handling
                      0135
               51 03  CE  0135          30026$: MNEGL   #3,R1                   ; Set error code to broken soft 30021$
                      0138
                      0138          30019$:
         00E8 D4 01   D0  0138                  MOVL    #1,@PDT$L_PMC(R4)       ; Min port to prevent further queue
                      013D                                                      ;  operations by port
                      013D                  BUGCHECK  BADQHDR                 ; Notify debugger
                      013D                  .IF     IDN ,NONFATAL
                      013D                  BSBW    ERR$BUGCHECKNF
                      013D                  BUG_CHECK BADQHDR
                      013D                  .IFF
                      013D                  BSBW    ERR$BUGCHECK
                      0140                  BUG_CHECK BADQHDR,TYPE=FATAL
                 FEFF 0140                          .WORD   ^XFEFF
                 0004' 0142                         .IIF IDN <FATAL>,<FATAL> , .WORD        BUG$_BADQHDR!4
                      0144                          .IIF DIF <FATAL>,<FATAL> , .WORD        BUG$_BADQHDR
                      0144
                      0144                  .ENDC
                      0144
                      0144
                      0144          30020$:                                   ; Check succeeded
            00 63 00  E7  0144                  BBCCI   #0,(R3),30022$          ; Unlock queue for port
                      0148
                      0148          30022$:
                      0148                  ENBINT                            ; Enable interrupts again
                      0148                  .IF B
               00' 8E DA  0148                  MTPR    (SP)+,S^#PR$_IPL
                      014B                  .IFF
                      014B                  MTPR    ,S^#PR$_IPL
                      014B                  .ENDC
                      014B
                      014B
                      014B
                      014B
         53 0208 C4   D0  014B  199       MOVL    PDT$L_DFQHDR(R4),R3     ; Get addr of dg free Q
                      0150  200           CHKQ                            ; Verify
                      0150
               52 53  D0  0150            MOVL    R3,R2                   ; Get copy of listhd addr
                  51  D4  0153            CLRL    R1                      ; Zero entry counter
                      0155                DSBINT                          ; Disable interrupts
                      0155                .IF B
               7E 00' DB  0155                  MFPR    S^#PR$_IPL,-(SP)
                      0158                .IFF
                      0158                MFPR    S^#PR$_IPL,
                      0158                .ENDC
                      0158                .IF B
               00' 1F DA  0158                  MTPR    #31,S^#PR$_IPL
                      015B                .IFF
                      015B                MTPR    ,S^#PR$_IPL
                      015B                .ENDC
```

```
                     015B
                     015B
                     015B      30030$:
FC 63   00   E6      015B              BBSSI   #0,(R3),30030$          ; 30030$ queue before reading
                     015F
                     015F      30027$:
   55   52   D0      015F              MOVL    R2,R5                   ; Save addr of this entry
                     0162
   50   62   D0      0162              MOVL    (R2),R0                 ; Get offset to next entry
   50   01   CA      0165              BICL    #1,R0                   ; Clear interlock bit
 52   6240   9E      0168              MOVAB   (R2)[R0],R2             ; Get addr of next entry
50   04 A2   D0      016C              MOVL    4(R2),R0                ; Get back link from this entry
 50   6240   9E      0170              MOVAB   (R2)[R0],R0             ; Compute prev entry addr
   55   50   D1      0174              CMPL    R0,R5                   ; Computed addr = saved?
        1B   12      0177              BNEQ    30033$                  ; Branch if not
   53   52   D1      0179              CMPL    R2,R3                   ; Back at start?
        2F   13      017C              BEQL    30029$                  ; Branch if so
3B   0A A2   91      017E              CMPB    PPD$B_TYPE(R2),#DYN$C_CIDG  ; CI dg?
        06   13      0182              BEQL    30032$                  ; Branch if so
3C   0A A2   91      0184              CMPB    PPD$B_TYPE(R2),#DYN$C_CIMSG ; CI msg?
        0F   12      0188              BNEQ    30034$                  ; Branch if not
10 63   00   E1      018A      30032$: BBC     #0,(R3),30035$          ; Branch if somebody grabbed soft
                     018E                                              ;  interlock while we had it
CD 51   3F   F2      018E              AOBLSS  #63,R1,30027$           ; Else check max count and continue
        0D   11      0192              BRB     30028$                  ; Branch if max count expired
                     0194
   51   01   CE      0194      30033$: MNEGL   #1,R1                   ; Set error code to bad blink
        08   11      0197              BRB     30028$                  ; Join common error handling
                     0199
   51   02   CE      0199      30034$: MNEGL   #2,R1                   ; Set error code to bad struc type
        03   11      019C              BRB     30028$                  ; Join common error handling
                     019E
   51   03   CE      019E      30035$: MNEGL   #3,R1                   ; Set error code to broken soft 30030$
                     01A1
                     01A1      30028$:
00E8 D4   01   D0    01A1              MOVL    #1,@PDT$L_PMC(R4)       ; Min port to prevent further queue
                     01A6                                              ;  operations by port
                     01A6              BUGCHECK  BADQHDR               ; Notify debugger
                     01A6              .IF     IDN ,NONFATAL
                     01A6              BSBW    ERR$BUGCHECKNF
                     01A6              BUG_CHECK BADQHDR
                     01A6              .IFF
         FE57'  30   01A6              BSBW    ERR$BUGCHECK
                     01A9              BUG_CHECK BADQHDR,TYPE=FATAL
              FEFF   01A9                      .WORD   ^XFEFF
              0004'  01AB                      .IIF IDN <FATAL>,<FATAL> , .WORD        BUG$_BADQHDR!4
                     01AD                      .IIF DIF <FATAL>,<FATAL> , .WORD        BUG$_BADQHDR
                     01AD
                     01AD              .ENDC
                     01AD
                     01AD
                     01AD      30029$:                                 ; Check succeeded
00 63   00   E7      01AD              BBCCI   #0,(R3),30031$          ; Unlock queue for port
                     01B1
                     01B1      30031$:
                     01B1              ENBINT                          ; Enable interrupts again
                     01B1                      .IF B
```

```
                00' 8E   DA  01B1                       MTPR    (SP)+,S^#PR$_IPL
                             01B4                       .IFF
                             01B4                       MTPR    ,S^#PR$_IPL
                             01B4                       .ENDC
                             01B4
                             01B4
                             01B4
                             01B4
            53  020C C4   D0  01B4  201          MOVL    PDT$L_MFQHDR(R4),R3     ; Get addr of msg free Q
                             01B9  202          CHKQ                            ; Verify
                             01B9
                52   53   D0  01B9               MOVL    R3,R2                   ; Get copy of listhd addr
                     51   D4  01BC               CLRL    R1                      ; Zero entry counter
                             01BE               DSBINT                          ; Disable interrupts
                             01BE                       .IF B
                7E   00'  DB  01BE                       MFPR    S^#PR$_IPL,-(SP)
                             01C1                       .IFF
                             01C1                       MFPR    S^#PR$_IPL,
                             01C1                       .ENDC
                             01C1                       .IF B
                00'  1F   DA  01C1                       MTPR    #31,S^#PR$_IPL
                             01C4                       .IFF
                             01C4                       MTPR    ,S^#PR$_IPL
                             01C4                       .ENDC
                             01C4
                             01C4
                             01C4           30039$:
              FC 63   00   E6  01C4                   BBSSI   #0,(R3),30039$          ; 30039$ queue before reading
                             01C8
                             01C8           30036$:
                55   52   D0  01C8                   MOVL    R2,R5                   ; Save addr of this entry
                             01CB
                50   62   D0  01CB                   MOVL    (R2),R0                 ; Get offset to next entry
                50   01   CA  01CE                   BICL    #1,R0                   ; Clear interlock bit
             52   6240   9E  01D1                   MOVAB   (R2)[R0],R2             ; Get addr of next entry
            50   04 A2   D0  01D5                   MOVL    4(R2),R0                ; Get back link from this entry
             50   6240   9E  01D9                   MOVAB   (R2)[R0],R0             ; Compute prev entry addr
                55   50   D1  01DD                   CMPL    R0,R5                   ; Computed addr = saved?
                     1B   12  01E0                   BNEQ    30042$                  ; Branch if not
                53   52   D1  01E2                   CMPL    R2,R3                   ; Back at start?
                     2F   13  01E5                   BEQL    30038$                  ; Branch if so
            3B   0A A2   91  01E7                   CMPB    PPD$B_TYPE(R2),#DYN$C_CIDG  ; CI dg?
                     06   13  01EB                   BEQL    30041$                  ; Branch if so
            3C   0A A2   91  01ED                   CMPB    PPD$B_TYPE(R2),#DYN$C_CIMSG ; CI msg?
                     0F   12  01F1                   BNEQ    30043$                  ; Branch if not
            10 63   00   E1  01F3           30041$: BBC     #0,(R3),30044$          ; Branch if somebody grabbed soft
                             01F7                                                   ;  interlock while we had it
            CD 51   3F   F2  01F7                   AOBLSS  #63,R1,30036$           ; Else check max count and continue
                     0D   11  01FB                   BRB     30037$                  ; Branch if max count expired
                             01FD
                51   01   CE  01FD           30042$: MNEGL   #1,R1                   ; Set error code to bad blink
                     08   11  0200                   BRB     30037$                  ; Join common error handling
                             0202
                51   02   CE  0202           30043$: MNEGL   #2,R1                   ; Set error code to bad struc type
                     03   11  0205                   BRB     30037$                  ; Join common error handling
                             0207
                51   03   CE  0207           30044$: MNEGL   #3,R1                   ; Set error code to broken soft 30039$
```

```
                          020A
                          020A      30037$:
                00E8 D4   020A                MOVL   #1,@PDT$L_PMC(R4)        ; Min port to prevent further queue
                          020F                                                  ;  operations by port
                          020F                BUGCHECK  BADQHDR                 ; Notify debugger
                          020F                .IF    IDN ,NONFATAL
                          020F                BSBW   ERR$BUGCHECKNF
                          020F                BUG_CHECK BADQHDR
                          020F                .IFF
                    FDEE' 30  020F                BSBW   ERR$BUGCHECK
                          0212                BUG_CHECK BADQHDR,TYPE=FATAL
                    FEFF  0212                      .WORD   ^XFEFF
                    0004' 0214                      .IIF IDN <FATAL>,<FATAL> , .WORD        BUG$_BADQHDR!4
                          0216                      .IIF DIF <FATAL>,<FATAL> , .WORD        BUG$_BADQHDR
                          0216
                          0216                .ENDC
                          0216
                          0216
                          0216      30038$:                                   ; Check succeeded
                  00 63   00  E7  0216                BBCCI  #0,(R3),30040$           ; Unlock queue for port
                          021A
                          021A      30040$:
                          021A                ENBINT                            ; Enable interrupts again
                          021A                      .IF B
                      00' 8E  DA  021A                MTPR    (SP)+,S^#PR$_IPL
                          021D                      .IFF
                          021D                      MTPR    ,S^#PR$_IPL
                          021D                      .ENDC
                          021D
                          021D
                          021D
                          021D
                          021D
                51 8ED0   021D    203               POPL    R1                    ; Restore registers
                52 8ED0   0220    204               POPL    R2                    ;
                53 8ED0   0223    205               POPL    R3                    ;
                55 8ED0   0226    206               POPL    R5                    ;
                          0229    207
                     05   0229    208 20$:          RSB                           ; Return
                          022A    209
                          022A    210               .DSABL  LSB
```

```
                     022A   212          .SBTTL  -       MON$CHKQ_POST,  CHECK ALL QUEUES AFTER
                     022A   213          .SBTTL  -                       A QUEUE OPERATION
                     022A   214
                     022A   215 ;+
                     022A   216 ; Checks all queues saving condition codes.  Otherwise, it is the
                     022A   217 ; same as the subroutine MON$CHKQ.
                     022A   218 ;-
                     022A   219
                     022A   220          .ENABL  LSB
                     022A   221
                     022A   222 MON$CHKQ_POST::
                     022A   223
          00   E1    022A   224          BBC     #MON$V_QCHK,-           ; Branch if queue checking
 0C FDD1   CF         022C   225                  MON$FLAGS,20$           ;  is disabled
        7E   DC      0230   226          MOVPSL  -(SP)                   ; Save PSL
0000023C'EF   DF     0232   227          PUSHAL  20$                     ; Save continuation addr
        FDD1   30    0238   228          BSBW    CHKQ_ALT                ; Verify all queues
               02    023B   229          REI                             ; Restore condition codes, continue PC
                     023C   230
               05    023C   231 20$:     RSB                             ; Return to caller
                     023D   232
                     023D   233          .DSABL  LSB
                     023D   234
                     023D   235
                     023D   236
                     023D   237
```

```
                     023D   239          .SBTTL  TRACE FACILITY
                     023D   240          .SBTTL  -       TRACE DEFINITIONS
                     023D   241
                     023D   242 ;
                     023D   243 ; Misc data:
                     023D   244 ;
                     023D   245
                     023D   246 TRC$ENABL::                                  ; Low bit set/clear for
                     023D   247                                              ;  enable/disable
00000000             023D   248          .LONG   0                           ; Default is disabled
                     0241   249
                     0241   250 TRC$BUFFER::                                 ; Addr of trace buffer
                     0241   251
00000000             0241   252          .LONG   0                           ;
                     0245   253
                     0245   254 ;
                     0245   255 ; The trace buffer is allocated from pool.  It consists of a header
                     0245   256 ; and a series of fixed length entries.  The occupied entries are
                     0245   257 ; maintained on a doubly linked list, youngest is at the head of the
                     0245   258 ; list and the oldest is on the tail.
                     0245   259 ;
                     0245   260
                     0245   261          $DEFINI TRC,GLOBAL
                     0245                .SAVE   LOCAL_BLOCK
                     0245                .NOCROSS
                     0245                .IIF    DIF <GLOBAL> <GLOBAL>,.ENABLE   SUPPRESSION
                     0245                .PSECT  $ABS$,ABS
                     0944                $GBLINI GLOBAL
                     0944                .IF     IDN <GLOBAL> <GLOBAL>
                     0944                .MACRO  $DEF    SYM,ALLOC,SIZ
                     0944                .IIF    NB,SYM, SYM::
                     0944                .IIF    NB,ALLOC,       ALLOC   SIZ
                     0944                .ENDM   $DEF
                     0944                .MACRO  $EQU    SYM,VAL
                     0944                SYM==VAL
                     0944                .ENDM   $EQU
                     0944                .MACRO  $VIELD1 MOD,SEP,SYM,SIZ,MSK
                     0944                SIZ...=1
                     0944                .IIF    NB,SIZ, SIZ...=SIZ
                     0944                .IF     NB,SYM
                     0944                MOD'SEP'V_'SYM==BIT...
                     0944                .IIF    NB,SIZ, MOD'SEP'S_'SYM==SIZ
                     0944                .IIF    NB,MSK, MOD'SEP'M_'SYM==<<<1@SIZ...>-1>@BIT...>
                     0944                .ENDC
                     0944                BIT...=BIT...+SIZ...
                     0944                .ENDM   $VIELD1
                     0944                .IFF
                     0944                .IIF    DIF <GLOBAL> <LOCAL>,.ERROR ;ARG MUST BE "GLOBAL","LOCAL",OR NULL
                     0944                .MACRO  $DEF    SYM,ALLOC,SIZ
                     0944                .IIF    NB,SYM, SYM:
                     0944                .IIF    NB,ALLOC,       ALLOC   SIZ
                     0944                .ENDM   $DEF
                     0944                .MACRO  $EQU    SYM,VAL
                     0944                SYM=VAL
                     0944                .ENDM   $EQU
                     0944                .MACRO  $VIELD1 MOD,SEP,SYM,SIZ,MSK
                     0944                SIZ...=1
```

```
                     0944                .IIF    NB,SIZ, SIZ...=SIZ
                     0944                .IF     NB,SYM
                     0944                MOD'SEP'V_'SYM=BIT...
                     0944                .IIF    NB,SIZ, MOD'SEP'S_'SYM=SIZ
                     0944                .IIF    NB,MSK, MOD'SEP'M_'SYM=<<<1@SIZ...>-1>@BIT...>
                     0944                .ENDC
                     0944                BIT...=BIT...+SIZ...
                     0944                .ENDM   $VIELD1
                     0944                .ENDC
                     0944
00000000             0944                .=0
                     0000
                     0000    262
                     0000    263         $DEF    TRC$L_NEXTENT   .BLKL    1       ; Addr of next entry to use
                     0000                .IIF    NB,TRC$L_NEXTENT,       TRC$L_NEXTENT::
00000004             0000                .IIF    NB,.BLKL,       .BLKL   1
                     0004
                     0004    264
                     0004    265         $DEF    TRC$Q_QHDR      .BLKQ   1       ; Queue header of entries
                     0004                .IIF    NB,TRC$Q_QHDR,  TRC$Q_QHDR::
0000000C             0004                .IIF    NB,.BLKQ,       .BLKQ   1
                     000C
                     000C    266
                     000C    267         $DEF    TRC$L_SPR       .BLKL   1       ; Spare longwd
                     000C                .IIF    NB,TRC$L_SPR,   TRC$L_SPR::
00000010             000C                .IIF    NB,.BLKL,       .BLKL   1
                     0010
                     0010    268
                     0010    269         $DEF    TRC$C_FIRSTENT                  ; Addr of first entry in table
                     0010                .IIF    NB,TRC$C_FIRSTENT,      TRC$C_FIRSTENT::
                     0010                .IIF    NB,,
                     0010
                     0010    270
                     0010    271         $EQU    TRC$C_ENTSIZ    <96>            ; 96 bytes per entry
00000060             0010                TRC$C_ENTSIZ==96
                     0010
                     0010    272
                     0010    273         $EQU    TRC$C_ENTCNT    <64>            ; Room for 64 entries
00000040             0010                TRC$C_ENTCNT==64
                     0010
                     0010    274
                     0010    275         $EQU    TRC$C_BUFSIZ    <TRC$C_ENTCNT*TRC$C_ENTSIZ+TRC$C_FIRSTENT>
00001810             0010                TRC$C_BUFSIZ==TRC$C_ENTCNT*TRC$C_ENTSIZ+TRC$C_FIRSTENT
                     0010
                     0010    276
                     0010    277                                                 ; Total buffer size
                     0010    278
                     0010    279         $DEFEND TRC
                     0010                .MACRO  $TRCDEF A
                     0010                .ENDM   $TRCDEF
                     0010                .IIF    DIF <> <GLOBAL>,.DISABLE        SUPPRESSION
                     0010                .CROSS
                00000245                .RESTORE
                     0245
                     0245    280
                     0245    281 ;
                     0245    282 ; Trace entries consist of a common header.  The structure type field
```

```
                     0245    283 ; contains a type code indicative of the type of data in the entry.
                     0245    284 ; If the entry type offsets are read into sda, sda should be able to
                     0245    285 ; format the trace buffer for us.
                     0245    286 ;
                     0245    287
                     0245    288         $DEFINI TRCE,GLOBAL
                     0245                .SAVE   LOCAL_BLOCK
                     0245                .NOCROSS
                     0245                .IIF    DIF <GLOBAL> <GLOBAL>,.ENABLE   SUPPRESSION
                     0245                .PSECT  $ABS$,ABS
                     0944                $GBLINI GLOBAL
                     0944                .IF     IDN <GLOBAL> <GLOBAL>
                     0944                .MACRO  $DEF    SYM,ALLOC,SIZ
                     0944                .IIF    NB,SYM, SYM::
                     0944                .IIF    NB,ALLOC,       ALLOC   SIZ
                     0944                .ENDM   $DEF
                     0944                .MACRO  $EQU    SYM,VAL
                     0944                SYM==VAL
                     0944                .ENDM   $EQU
                     0944                .MACRO  $VIELD1 MOD,SEP,SYM,SIZ,MSK
                     0944                SIZ...=1
                     0944                .IIF    NB,SIZ, SIZ...=SIZ
                     0944                .IF     NB,SYM
                     0944                MOD'SEP'V_'SYM==BIT...
                     0944                .IIF    NB,SIZ, MOD'SEP'S_'SYM==SIZ
                     0944                .IIF    NB,MSK, MOD'SEP'M_'SYM==<<<1@SIZ...>-1>@BIT...>
                     0944                .ENDC
                     0944                BIT...=BIT...+SIZ...
                     0944                .ENDM   $VIELD1
                     0944                .IFF
                     0944                .IIF    DIF <GLOBAL> <LOCAL>,.ERROR ;ARG MUST BE "GLOBAL","LOCAL",OR NULL
                     0944                .MACRO  $DEF    SYM,ALLOC,SIZ
                     0944                .IIF    NB,SYM, SYM:
                     0944                .IIF    NB,ALLOC,       ALLOC   SIZ
                     0944                .ENDM   $DEF
                     0944                .MACRO  $EQU    SYM,VAL
                     0944                SYM=VAL
                     0944                .ENDM   $EQU
                     0944                .MACRO  $VIELD1 MOD,SEP,SYM,SIZ,MSK
                     0944                SIZ...=1
                     0944                .IIF    NB,SIZ, SIZ...=SIZ
                     0944                .IF     NB,SYM
                     0944                MOD'SEP'V_'SYM=BIT...
                     0944                .IIF    NB,SIZ, MOD'SEP'S_'SYM=SIZ
                     0944                .IIF    NB,MSK, MOD'SEP'M_'SYM=<<<1@SIZ...>-1>@BIT...>
                     0944                .ENDC
                     0944                BIT...=BIT...+SIZ...
                     0944                .ENDM   $VIELD1
                     0944                .ENDC
                     0944
00000000             0944                .=0
                     0000
                     0000    289
                     0000    290         $DEF    TRCE$L_FL       .BLKL   1       ; Fwd link
                     0000                .IIF    NB,TRCE$L_FL,   TRCE$L_FL::
00000004             0000                .IIF    NB,.BLKL,       .BLKL   1
                     0004
```

```
                     0004    291
                     0004    292          $DEF    TRCE$L_BL       .BLKL   1       ; Back link
                     0004                 .IIF    NB,TRCE$L_BL,   TRCE$L_BL::
00000008             0004                 .IIF    NB,.BLKL,       .BLKL   1
                     0008
                     0008    293
                     0008    294          $DEF    TRCE$W_SIZE     .BLKW   1       ; Size of an entry
                     0008                 .IIF    NB,TRCE$W_SIZE, TRCE$W_SIZE::
0000000A             0008                 .IIF    NB,.BLKW,       .BLKW   1
                     000A
                     000A    295
                     000A    296          $DEF    TRCE$B_TYPE     .BLKB   1       ; Entry type code (struct type)
                     000A                 .IIF    NB,TRCE$B_TYPE, TRCE$B_TYPE::
0000000B             000A                 .IIF    NB,.BLKB,       .BLKB   1
                     000B
                     000B    297
                     000B    298          $DEF    TRCE$B_SPR      .BLKB   1       ; Spare byte
                     000B                 .IIF    NB,TRCE$B_SPR,  TRCE$B_SPR::
0000000C             000B                 .IIF    NB,.BLKB,       .BLKB   1
                     000C
                     000C    299
                     000C    300          $DEF    TRCE$L_TIME     .BLKL   1       ; Time entry was filled
                     000C                 .IIF    NB,TRCE$L_TIME, TRCE$L_TIME::
00000010             000C                 .IIF    NB,.BLKL,       .BLKL   1
                     0010
                     0010    301
                     0010    302          $DEF    TRCE$L_PDT      .BLKL   1       ; Caller's PDT addr (R4)
                     0010                 .IIF    NB,TRCE$L_PDT,  TRCE$L_PDT::
00000014             0010                 .IIF    NB,.BLKL,       .BLKL   1
                     0014
                     0014    303
                     0014    304          $DEF    TRCE$C_BASE                     ; Start of type specific data
                     0014                 .IIF    NB,TRCE$C_BASE, TRCE$C_BASE::
                     0014                 .IIF    NB,,
                     0014
                     0014    305
                     0014    306 ;
                     0014    307 ; Entry type specific formats:
                     0014    308 ;
                     0014    309 ; Message (or datagram) trace:
                     0014    310 ;
                     0014    311
00000014             0014    312 .=TRCE$C_BASE
                     0014    313
                     0014    314          $EQU    DYN$C_TRCMSG    <^X81>
00000081             0014                 DYN$C_TRCMSG==^X81
                     0014
                     0014    315
                     0014    316          $DEF    TRCE$L_PC       .BLKL   1       ; Caller's PC
                     0014                 .IIF    NB,TRCE$L_PC,   TRCE$L_PC::
00000018             0014                 .IIF    NB,.BLKL,       .BLKL   1
                     0018
                     0018    317
                     0018    318          $DEF    TRCE$L_PSL      .BLKL   1       ; Caller's PSL
                     0018                 .IIF    NB,TRCE$L_PSL,  TRCE$L_PSL::
0000001C             0018                 .IIF    NB,.BLKL,       .BLKL   1
                     001C
```

```
                     001C   319
                     001C   320          $DEF    TRCE$L_MSGADDR  .BLKL   1       ; Addr of message being traced
                     001C                .IIF    NB,TRCE$L_MSGADDR,      TRCE$L_MSGADDR::
          00000020   001C                .IIF    NB,.BLKL,       .BLKL   1
                     0020
                     0020   321
                     0020   322          $DEF    TRCE$C_MSGDATA                  ; Start of message data
                     0020                .IIF    NB,TRCE$C_MSGDATA,      TRCE$C_MSGDATA::
                     0020                .IIF    NB,,
                     0020
                     0020   323
                     0020   324 ;
                     0020   325 ; PC trace:
                     0020   326 ;
                     0020   327
          00000014   0020   328 .=TRCE$C_BASE
                     0014   329
                     0014   330          $EQU    DYN$C_TRCPC     <^X82>
          00000082   0014                DYN$C_TRCPC==^X82
                     0014
                     0014   331
          00000018   0014   332 .=.+4                                           ; Caller's PC
                     0018   333
          0000001C   0018   334 .=.+4                                           ; Caller's PSL
                     001C   335
                     001C   336          $DEF    TRCE$L_R0       .BLKL   1       ; Caller's R0-R5
                     001C                .IIF    NB,TRCE$L_R0,   TRCE$L_R0::
          00000020   001C                .IIF    NB,.BLKL,       .BLKL   1
                     0020
                     0020   337          $DEF    TRCE$L_R1       .BLKL   1
                     0020                .IIF    NB,TRCE$L_R1,   TRCE$L_R1::
          00000024   0020                .IIF    NB,.BLKL,       .BLKL   1
                     0024
                     0024   338          $DEF    TRCE$L_R2       .BLKL   1
                     0024                .IIF    NB,TRCE$L_R2,   TRCE$L_R2::
          00000028   0024                .IIF    NB,.BLKL,       .BLKL   1
                     0028
                     0028   339          $DEF    TRCE$L_R3       .BLKL   1
                     0028                .IIF    NB,TRCE$L_R3,   TRCE$L_R3::
          0000002C   0028                .IIF    NB,.BLKL,       .BLKL   1
                     002C
                     002C   340          $DEF    TRCE$L_R4       .BLKL   1
                     002C                .IIF    NB,TRCE$L_R4,   TRCE$L_R4::
          00000030   002C                .IIF    NB,.BLKL,       .BLKL   1
                     0030
                     0030   341          $DEF    TRCE$L_R5       .BLKL   1
                     0030                .IIF    NB,TRCE$L_R5,   TRCE$L_R5::
          00000034   0030                .IIF    NB,.BLKL,       .BLKL   1
                     0034
                     0034   342
                     0034   343          $DEFEND TRCE
                     0034                .MACRO  $TRCEDEF A
                     0034                .ENDM   $TRCEDEF
                     0034                .IIF    DIF <> <GLOBAL>,.DISABLE        SUPPRESSION
                     0034                .CROSS
                00000245                 .RESTORE
                     0245
```

```
                                0245   345          .SBTTL  -       TRACE INITIALIZATION
                                0245   346
                                0245   347 ;+
                                0245   348 ; TRC$INIT allocates the trace buffer from pool, formats the header,
                                0245   349 ; and saves its address.
                                0245   350 ;
                                0245   351 ; Inputs:
                                0245   352 ;
                                0245   353 ;       IPL                     -Fork IPL or greater
                                0245   354 ;
                                0245   355 ; Outputs:
                                0245   356 ;
                                0245   357 ;       TRC$BUFFER              -0 if insufficient memory, else
                                0245   358 ;                                addr of start of buffer
                                0245   359 ;       TRC$ENABL               -Low bit clear if insufficient memory; else
                                0245   360 ;                                unchanged
                                0245   361 ;       All registers           -Preserved
                                0245   362 ;-
                                0245   363
                                0245   364          .ENABL  LSB
                                0245   365
                                0245   366 TRC$INIT::
                                0245   367
                    F9 AF    D5 0245   368          TSTL    TRC$BUFFER              ; Is there already a buffer (in case
                                0248   369                                          ;  there are multiple ports)
                       46    12 0248   370          BNEQ    20$                     ; Branch if so
                       3F    BB 024A   371          PUSHR   #^M<R0,R1,R2,R3,R4,R5>  ; Save registers
                 51  1820 8F    3C 024C   372          MOVZWL  #TRC$C_BUFSIZ+16,R1     ; Get total buffer size
              54  00000000'GF    DE 0251   373          MOVAL   G^EXE$GL_NONPAGED,R4    ; Fiddle with allocate
                       64    DD 0258   374          PUSHL   (R4)                    ;  IPL to allow
              64  00000000'8F    DB 025A   375          MFPR    #PR$_IPL,(R4)           ;  greater than fork IPL
                  00000000'GF    16 0261   376          JSB     G^EXE$ALONONPAGED       ;  and allocate pool
                       64 8ED0  0267   377          POPL    (R4)                    ; Restore allocate IPL
                          06 50    E8 026A   378          BLBS    R0,5$                   ; Branch if got pool
                    CC AF    01    CA 026D   379          BICL    #1,TRC$ENABL            ; Else disable trace function
                       1B    11 0271   380          BRB     10$                     ;  and return
                                0273   381
                       82    7C 0273   382 5$:       CLRQ    (R2)+                   ; Clear out header
          82   51    00130000 8F    C1 0275   383          ADDL3   #<DYN$C_BUFIO@16>,R1,(R2)+ ; Set size and type
                       82    D4 027D   384          CLRL    (R2)+                   ; Clear out junk
                    BE AF    52    D0 027F   385          MOVL    R2,TRC$BUFFER           ; Save buffer address
                    82    10 A2    DE 0283   386          MOVAL   TRC$C_FIRSTENT(R2),(R2)+ ; Set addr of 1st entry
                       62    52    D0 0287   387          MOVL    R2,(R2)                 ; Set filled entry
                    04 A2    52    D0 028A   388          MOVL    R2,4(R2)                ;  queue to empty
                                028E   389
                       3F    BA 028E   390 10$:      POPR    #^M<R0,R1,R2,R3,R4,R5>  ; Restore registers
                                0290   391
                             05 0290   392 20$:      RSB                             ; Return
                                0291   393
                                0291   394          .DSABL  LSB
```

```
                                0291   396          .SBTTL  TRC$LOGMSG,     Log a Message or Datagram
                                0291   397
                                0291   398 ;+
                                0291   399 ; This routine allocates an entry in the trace buffer and fills it with
                                0291   400 ; the PC of the caller, addr of the message, and first few longwords of
                                0291   401 ; the message.
                                0291   402 ;
                                0291   403 ; Inputs:
                                0291   404 ;
                                0291   405 ;       R2                      -Addr of message being traced
                                0291   406 ;       R4                      -PDT addr
                                0291   407 ;
                                0291   408 ; Outputs:
                                0291   409 ;
                                0291   410 ;       All registers, PSL      -Preserved
                                0291   411 ;-
                                0291   412
                                0291   413          .ENABL  LSB
                                0291   414
                                0291   415 TRC$LOGMSG::
                                0291   416
32 AB AF    00    E1            0291   417          BBC     #0,TRC$ENABL,10$        ; Branch if trace disabled
              7E    DC          0296   418          MOVPSL  -(SP)                   ; Save PSL
   000002C8'EF    DF          0298   419          PUSHAL  10$                     ;  and PC of RSB from this routine
                              029E   420          DSBINT                          ; Raise IPL
                              029E                .IF B
        7E    00'   DB        029E                MFPR    S^#PR$_IPL,-(SP)
                              02A1                .IFF
                              02A1                MFPR    S^#PR$_IPL,
                              02A1                .ENDC
                              02A1                .IF B
        00'   1F    DA        02A1                MTPR    #31,S^#PR$_IPL
                              02A4                .IFF
                              02A4                MTPR    ,S^#PR$_IPL
                              02A4                .ENDC
                              02A4
              3F    BB        02A4   421          PUSHR   #^M<R0,R1,R2,R3,R4,R5>  ; Save registers
     50     81    8F    9A    02A6   422          MOVZBL  #DYN$C_TRCMSG,R0        ; Get entry type code
              004F    30      02AA   423          BSBW    TRC$ALLOC_ENT           ; Allocate and init next entry
14 A1      24    AE    D0    02AD   424          MOVL    <9*4>(SP),TRCE$L_PC(R1) ; Copy caller's PC
18 A1      20    AE    D0    02B2   425          MOVL    <8*4>(SP),TRCE$L_PSL(R1) ;  and PSL
   1C A1         52    D0    02B7   426          MOVL    R2,TRCE$L_MSGADDR(R1)   ; Copy message addr to trace entry
      0040 8F    28          02BB   427          MOVC3   #<TRC$C_ENTSIZ-TRCE$C_MSGDATA>,-
   20 A1         62          02BF   428                  (R2),TRCE$C_MSGDATA(R1) ; Copy as much message as possible
              3F    BA        02C2   429          POPR    #^M<R0,R1,R2,R3,R4,R5>  ; Restore registers
                              02C4   430          ENBINT                          ; Lower IPL
                              02C4                .IF B
        00'   8E    DA        02C4                MTPR    (SP)+,S^#PR$_IPL
                              02C7                .IFF
                              02C7                MTPR    ,S^#PR$_IPL
                              02C7                .ENDC
                              02C7
              02    02C7   431          REI                             ; Restore PC, PSL
                           02C8   432
              05    02C8   433 10$:     RSB                             ; Return
                           02C9   434
                           02C9   435          .DSABL  LSB
```

```
                                02C9    437          .SBTTL  TRC$LOGPC,      Log PC and Registers
                                02C9    438
                                02C9    439 ;+
                                02C9    440 ; This routine logs the caller's PC, PSL, and R0-R5.
                                02C9    441 ;
                                02C9    442 ; Inputs:
                                02C9    443 ;
                                02C9    444 ;       R4                          -PDT addr
                                02C9    445 ;
                                02C9    446 ; Outputs:
                                02C9    447 ;
                                02C9    448 ;       All registers, PSL          -Preserved
                                02C9    449 ;-
                                02C9    450
                                02C9    451          .ENABL  LSB
                                02C9    452
                                02C9    453 TRC$LOGPC::
                                02C9    454
2C FF6F CF    00    E1          02C9    455          BBC     #0,TRC$ENABL,10$        ; Branch if trace disabled
                  7E    DC      02CF    456          MOVPSL  -(SP)                   ; Save caller's PSL
    000002FB'EF     DF          02D1    457          PUSHAL  10$                     ; Save addr of RSB
                                02D7    458          DSBINT                          ; Raise IPL to 31
                                02D7                          .IF B
                  7E    00'   DB  02D7                        MFPR    S^#PR$_IPL,-(SP)
                                02DA                          .IFF
                                02DA                          MFPR    S^#PR$_IPL,
                                02DA                          .ENDC
                                02DA                          .IF B
                  00'   1F    DA  02DA                        MTPR    #31,S^#PR$_IPL
                                02DD                          .IFF
                                02DD                          MTPR    ,S^#PR$_IPL
                                02DD                          .ENDC
                                02DD
                        3F      BB  02DD    459          PUSHR   #^M<R0,R1,R2,R3,R4,R5>  ; Save registers
          50    82    8F      9A  02DF    460          MOVZBL  #DYN$C_TRCPC,R0         ; Get trace entry type code
                  0016          30  02E3    461          BSBW    TRC$ALLOC_ENT           ; Allocate and init next entry
14 A1               24  AE      D0  02E6    462          MOVL    <9*4>(SP),TRCE$L_PC(R1) ; Copy caller's PC
18 A1               20  AE      D0  02EB    463          MOVL    <8*4>(SP),TRCE$L_PSL(R1) ;  and caller's PSL
          6E        18      28  02F0    464          MOVC3   #<6*4>,(SP),-           ; Copy registers from stack to
                1C  A1          02F3    465                  TRCE$L_R0(R1)          ;  to trace entry
                        3F      BA  02F5    466          POPR    #^M<R0,R1,R2,R3,R4,R5>  ; Restore registers
                                02F7    467          ENBINT                          ; Lower IPL
                                02F7                          .IF B
                  00'   8E    DA  02F7                        MTPR    (SP)+,S^#PR$_IPL
                                02FA                          .IFF
                                02FA                          MTPR    ,S^#PR$_IPL
                                02FA                          .ENDC
                                02FA
                                02FA    468          REI                             ; Restore PC, PSL
                                02FB    469
                                02FB    470 10$:     RSB                             ; Return to caller
                                02FC    471
                                02FC    472          .DSABL  LSB
```

```
                        02FC    474          .SBTTL  TRC$ALLOC_ENT,  ALLOCATE TRACE ENTRY
                        02FC    475
                        02FC    476 ;+
                        02FC    477 ; This routine allocates the next trace entry.  If the new entry is currently
                        02FC    478 ; the oldest entry on the queue (if it's on the tail), it is removed from the
                        02FC    479 ; queue and inserted on the head of the queue, making it the youngest entry.
                        02FC    480 ; The standard information is set in the entry:
                        02FC    481 ;
                        02FC    482 ;       size = TRC$C_ENTSIZ
                        02FC    483 ;       type = specified by caller
                        02FC    484 ;       time = read from interval count register (usec accuracy)
                        02FC    485 ;       PDT  = R4
                        02FC    486 ;
                        02FC    487 ; Inputs:
                        02FC    488 ;
                        02FC    489 ;       R0                      -Structure/trace entry type code
                        02FC    490 ;
                        02FC    491 ; Outputs:
                        02FC    492 ;
                        02FC    493 ;       R1                      -Addr of trace entry
                        02FC    494 ;       R5                      -Destroyed
                        02FC    495 ;       Other registers         -Preserved
                        02FC    496 ;-
                        02FC    497
                        02FC    498          .ENABL  LSB
                        02FC    499
                        02FC    500 TRC$ALLOC_ENT::
                        02FC    501
 55   FF41 CF    D0     02FC    502          MOVL    TRC$BUFFER,R5           ; Get addr of trace buffer
      51   65    D0     0301    503          MOVL    (R5),R1                 ; Get addr of next entry
   08 A5   51    D1     0304    504          CMPL    R1,TRC$Q_QHDR+4(R5)     ; Is it on the tail?
           04    12     0308    505          BNEQ    10$                     ; Branch if not
   51   08 B5    0F     030A    506          REMQUE  @TRC$Q_QHDR+4(R5),R1    ; Remove the entry from the tail
                        030E    507
   04 A5   61    0E     030E    508 10$:     INSQUE  (R1),TRC$Q_QHDR(R5)     ; Put entry on head of queue
      0060 8F    3C     0312    509          MOVZWL  #TRC$C_ENTSIZ,-         ; Set structure size
        08 A1           0316    510                  TRCE$W_SIZE(R1)         ;
   0A A1   50    90     0318    511          MOVB    R0,TRCE$B_TYPE(R1)      ;  and type
   00000000'GF  D0     031C    512          MOVL    G^EXE$GQ_SYSTIME,-      ; Time stamp entry
        0C A1           0322    513                  TRCE$L_TIME(R1)         ;
   10 A1   54    D0     0324    514          MOVL    R4,TRCE$L_PDT(R1)       ; Save PDT addr
   65   60 A1    DE     0328    515          MOVAL   TRC$C_ENTSIZ(R1),(R5)   ; Step to addr of next entry
50 55   00001810 8F     C1     032C    516          ADDL3   #TRC$C_BUFSIZ,R5,R0     ; Compute end of buffer
      50   65    D1     0334    517          CMPL    (R5),R0                 ; Next entry at end of past it?
           04    19     0337    518          BLSS    20$                     ; Branch if not
   65   10 A5    DE     0339    519          MOVAL   TRC$C_FIRSTENT(R5),(R5) ; Else cycle to top of buffer
                        033D    520                                          ;  for next entry.
                        033D    521
                  05    033D    522 20$:     RSB                             ; Return
                        033E    523
                        033E    524          .DSABL  LSB
                        033E    525          .END
```

```
$$$CURSIZ                       =  000001C4
$$$NEWSIZ                       =  000001D0
BUG$_BADQHDR                       ********    X   01
CHKQ_ALT                           0000000C R      01
DYN$C_BUFIO                     =  00000013
DYN$C_CIDG                      =  0000003B
DYN$C_CIMSG                     =  0000003C
DYN$C_TRCMSG                    =  00000081   G
DYN$C_TRCPC                     =  00000082   G
ERR$BUGCHECK                       ********    X   01
EXE$ALONONPAGED                    ********    X   01
EXE$GL_NONPAGED                    ********    X   01
EXE$GQ_SYSTIME                     ********    X   01
MON$CHKQ                           00000004 RG     01
MON$CHKQ_POST                      0000022A RG     01
MON$FLAGS                          00000000 RG     01
MON$M_QCHK                      =  00000001
MON$V_QCHK                      =  00000000
PA_CNF                             00000000
PA_CQ0                             00000908
PA_CQ1                             0000090C
PA_CQ2                             00000910
PA_CQ3                             00000914
PA_DFQ                             00000928
PA_MADR                            00000014
PA_MDATR                           00000018
PA_MFQ                             0000092C
PA_MTC                             00000930
PA_MTEC                            00000934
PA_PDC                             00000920
PA_PEC                             0000091C
PA_PESR                            0000093C
PA_PFAR                            00000938
PA_PIC                             00000924
PA_PMC                             00000004
PA_PPR                             00000940
PA_PQBBR                           00000904
PA_PS                              00000900
PA_PSR                             00000918
PDT$B_DQIMAP                       00000154
PDT$B_HSHUT_DG                     000001B0
PDT$B_MAX_PORT                     0000017C
PDT$B_NXT_PORT                     0000017E
PDT$B_P0_LBSTS                     00000180
PDT$B_P1_LBSTS                     00000181
PDT$B_PLOGMAP                      00000134
PDT$B_PORTMAP                      00000114
PDT$B_PORT_NUM                     0000017D
PDT$B_REQIDPS                      0000017F
PDT$C_LENGTH                    =  000000E4
PDT$C_PAREGBASE                    000000E4
PDT$C_PAREGEND                     00000110
PDT$C_PQB                       =  000001E0
PDT$L_CNF                          000000E4
PDT$L_CQ0                          000000F0
PDT$L_CQ1                          000000F4
PDT$L_DFQ                          000000FC
PDT$L_DFQHDR                       00000208
PDT$L_DGHDRSZ                      00000190
PDT$L_DGNETHD                      00000194
PDT$L_DQELOGOUT                    000002E0
PDT$L_GPTBASE                      0000022C
PDT$L_GPTLEN                       00000230
PDT$L_LBDG                         00000184
PDT$L_MFQ                          00000100
PDT$L_MFQHDR                       0000020C
PDT$L_MQELOGOUT                    00000320
PDT$L_MTC                          00000104
PDT$L_PFAR                         00000108
PDT$L_PMC                          000000E8
PDT$L_POLLERDUE                    0000018C
PDT$L_POOLDUE                      00000188
PDT$L_PPR                          0000010C
PDT$L_PS                           000000EC
PDT$L_PSR                          000000F8
PDT$L_SPTBASE                      00000224
PDT$L_SPTLEN                       00000228
PDT$L_VBDT                         0000021C
PDT$L_VPQB                         00000218
PDT$Q_COMQ2                        000001F0
PDT$Q_COMQ3                        000001F8
PDT$Q_COMQBASE                     000001E0
PDT$Q_COMQH                        000001E8
PDT$Q_COMQL                        000001E0
PDT$Q_DFREEQ                       000001D0
PDT$Q_FORMPB                       00000174
PDT$Q_MFREEQ                       000001D8
PDT$Q_RSPQ                         00000200
PDT$Q_TEMP_RSPQ                    0000019C
PDT$W_BDTLEN                       00000220
PDT$W_DQELEN                       00000210
PDT$W_LPORT_STS                    00000110
PDT$W_MQELEN                       00000214
PDT$W_PBCOUNT                      00000112
PDT$W_STDGDYN                      00000198
PDT$W_STDGUSED                     0000019A
PPD$B_DEF_ST                       0000001C
PPD$B_FLAGS                        0000000F
PPD$B_HWVERS                       00000034
PPD$B_LBDATA                       00000012
PPD$B_LCB_0                        00000012
PPD$B_LCB_LPORT                    00000010
PPD$B_LCB_NPORT                    0000000F
PPD$B_LCB_OPC                      00000011
PPD$B_LCB_PORT                     0000000E
PPD$B_OPC                          0000000E
PPD$B_PORT                         0000000C
PPD$B_PROTOCOL                     0000001A
PPD$B_RSTATE                       00000025
PPD$B_RST_PORT                     00000024
PPD$B_STATUS                       0000000D
PPD$B_SWFLAG                       0000000B
PPD$B_SYSTEMID                     00000014
PPD$B_TYPE                         0000000A
```

```
PPD$C_LB_LENGTH                 00000046
PPD$C_LCB_DATA                  00000013
PPD$C_LENGTH                    00000012
PPD$C_MIN_DGSIZ                 00000050
PPD$K_LB_LENGTH                 00000046
PPD$K_LENGTH                    00000012
PPD$L_BLINK                     00000004
PPD$L_DG_DISC                   00000028
PPD$L_FLINK                     00000000
PPD$L_IN_VCD                    00000018
PPD$L_LBCRC                     00000042
PPD$L_PO_ACK                    00000010
PPD$L_PO_NAK                    00000014
PPD$L_PO_NRSP                   00000018
PPD$L_P1_ACK                    0000001C
PPD$L_P1_NAK                    00000020
PPD$L_P1_NRSP                   00000024
PPD$L_REC_BOFF                  00000028
PPD$L_REC_NAME                  00000024
PPD$L_RPORT_FCN                 00000020
PPD$L_RPORT_REV                 0000001C
PPD$L_RPORT_TYP                 00000018
PPD$L_SND_BOFF                  00000020
PPD$L_SND_NAME                  0000001C
PPD$L_ST_ADDR                   00000018
PPD$L_XCT_LEN                   00000018
PPD$Q_CURTIME                   00000048
PPD$Q_NODENAME                  00000040
PPD$Q_SWINCARN                  00000028
PPD$Q_XCT_ID                    00000010
PPD$T_HWTYPE                    00000030
PPD$T_SWTYPE                    00000020
PPD$T_SWVERS                    00000024
PPD$W_LCB_LEN7                  0000000C
PPD$W_LENGTH                    00000010
PPD$W_MASK                      00000010
PPD$W_MAXDG                     0000001C
PPD$W_MAXMSG                    0000001E
PPD$W_MTYPE                     00000012
PPD$W_M_VAL                     00000014
PPD$W_SIZE                      00000008
PR$_IPL                         ********    X   01
SIZ...                        = 00000001
TRC$ALLOC_ENT                   000002FC  RG    01
TRC$BUFFER                      00000241  RG    01
TRC$C_BUFSIZ                  = 00001810    G
TRC$C_ENTCNT                  = 00000040    G
TRC$C_ENTSIZ                  = 00000060    G
TRC$C_FIRSTENT                  00000010    G
TRC$ENABL                       0000023D  RG    01
TRC$INIT                        00000245  RG    01
TRC$LOGMSG                      00000291  RG    01
TRC$LOGPC                       000002C9  RG    01
TRC$L_NEXTENT                   00000000    G
TRC$L_SPR                       0000000C    G
TRC$Q_QHDR                      00000004    G
TRCE$B_SPR                      0000000B    G
TRCE$B_TYPE                     0000000A    G
TRCE$C_BASE                     00000014    G
TRCE$C_MSGDATA                  00000020    G
TRCE$L_BL                       00000004    G
TRCE$L_FL                       00000000    G
TRCE$L_MSGADDR                  0000001C    G
TRCE$L_PC                       00000014    G
TRCE$L_PDT                      00000010    G
TRCE$L_PSL                      00000018    G
TRCE$L_R0                       0000001C    G
TRCE$L_R1                       00000020    G
TRCE$L_R2                       00000024    G
TRCE$L_R3                       00000028    G
TRCE$L_R4                       0000002C    G
TRCE$L_R5                       00000030    G
TRCE$L_TIME                     0000000C    G
TRCE$W_SIZE                     00000008    G
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
$$$115_DRIVER	0000033E (830.)	01 (1.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
ABS	00000944 (2372.)	02 (2.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	36	00:00:00.04	00:00:01.17
Command processing	132	00:00:00.41	00:00:03.98
Pass 1	313	00:00:06.69	00:00:27.04
Symbol table sort	0	00:00:00.70	00:00:01.80
Pass 2	185	00:00:01.70	00:00:11.59
Symbol table output	22	00:00:00.11	00:00:00.29
Psect synopsis output	2	00:00:00.01	00:00:00.01
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	692	00:00:09.67	00:00:45.89

The working set limit was 1650 pages.
66128 bytes (130 pages) of virtual memory were used to buffer the intermediate code.
There were 40 pages of symbol table space allocated to hold 652 non-local and 54 local symbols.
525 source lines were read in Pass 1, producing 16 object records in Pass 2.
23 pages of virtual memory were used to define 19 macros.

Macro library statistics

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

844 GETS were required to define 13 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:PAMONIT/OBJ=OBJ$:PAMONIT MSRC$:PAMONIT/UPDATE=(ENH$:PAMONIT)+EXECML$/LIB+LIB$:PALIB.MLB/LIB

PAINIT
LIS

PAFPCALL
LIS

PAMONIT
LIS

PAINTR
LIS

PUDRIVER
LIS

RTTDRIVER
LIS

PATABLES
LIS

PASCSCTL
LIS