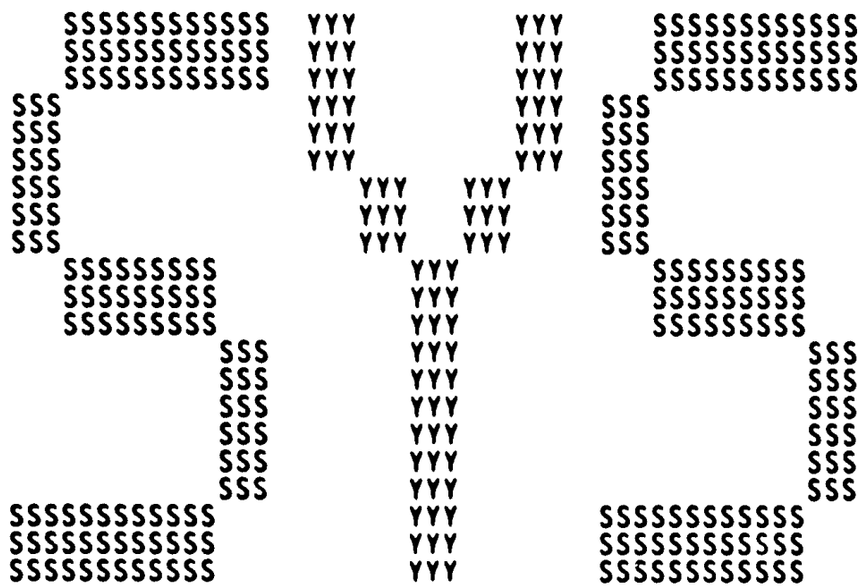

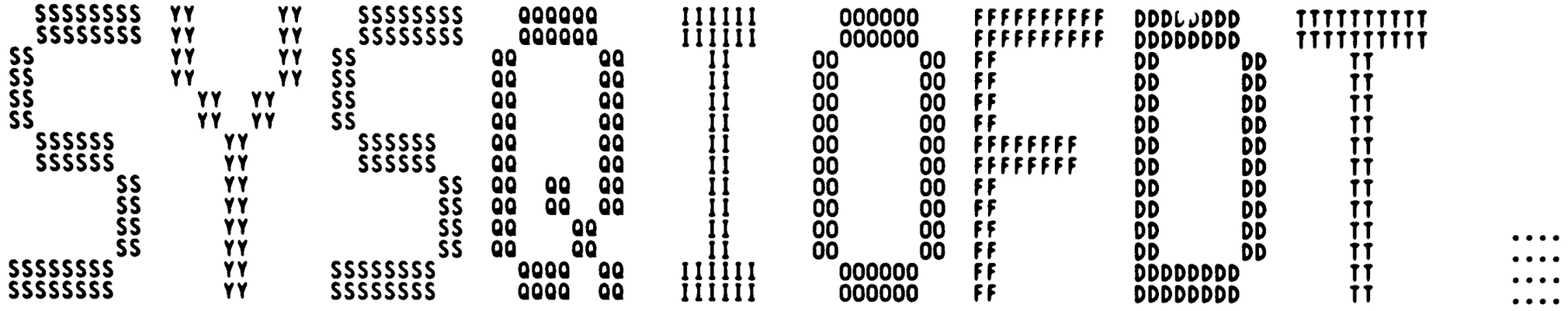
SSSSSSSS YY YY SSSSSSSS QQQQQQ IIIIII OOOOOO FFFFFFFFFF DDDDDDDD TTTTTTTTTT
SSSSSSSS YY YY SSSSSSSS QQQQQQ IIIIII OOOOOO FFFFFFFFFF DDDDDDDD TTTTTTTTTT
SS YY YY SS QQ QQ II OO OO FF DD DD TT
SS YY YY SS QQ QQ II OO OO FF DD DD TT
SS YY YY SS QQ QQ II OO OO FF DD DD TT
SS YY YY SS QQ QQ II OO OO FF DD DD TT
SSSSSS YY SSSSSS QQ QQ II OO OO FFFFFFFF DD DD TT
SSSSSS YY SSSSSS QQ QQ II OO OO FFFFFFFF DD DD TT
SS YY SS QQ QQ QQ II OO OO FF DD DD TT
SS YY SS QQ QQ QQ II OO OO FF DD DD TT
SS YY SS QQ QQ II OO OO FF DD DD TT
SS YY SS QQ QQ II OO OO FF DD DD TT
SSSSSSSS YY SSSSSSSS QQQQ QQ IIIIII OOOOOO FF DDDDDDDD TT
SSSSSSSS YY SSSSSSSS QQQQ QQ IIIIII OOOOOO FF DDDDDDDD 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

(1) 110 PLACE PROCESS IN I/O RESOURCE WAIT
(1) 166 ONE PARAMETER FUNCTION PROCESSING
(1) 199 ZERO PARAMETER FUNCTION PROCESSING
(1) 232 LOCAL DISK VALID FUNCTION PROCESSING
(1) 301 READ AND WRITE FUNCTION PROCESSING
(1) 354 READ AND WRITE FUNCTION BUFFER CHECK AND LOCK ROUTINES
(1) 397 READ AND WRITE BUFFER CHECK AND LOCK AND RETURN ROUTINES
(1) 487 BACKOUT A QIO
(2) 526 CHECK BUFFER ACCESSIBILITY FOR READ FUNCTION
(2) 562 CHECK BUFFER ACCESSIBILITY FOR WRITE FUNCTION
(2) 601 CHECK BUFFER ACCESSIBILITY FOR READ FUNCTION AND RETURN
(2) 675 CHECK BUFFER ACCESSIBILITY FOR WRITE FUNCTION AND RETURN
(2) 740 SET DEVICE MODE AND CHARACTERISTICS FUNCTIONS (AT FDT LEVEL)
(2) 785 SET DEVICE MODE AND CHARACTERISTICS FUNCTIONS
(2) 836 SENSE DEVICE MODE AND CHARACTERISTICS FUNCTIONS
(2) 874 CARRIAGE CONTROL INTERPRETATION

```
0000     1          .TITLE  SYSQIOFDT - SYSTEM SERVICE QUEUE I/O FDT SUBROUTINES
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 15-SEP-76
0000    29 ;
0000    30 ; MODIFIED BY:
0000    31 ;
0000    32 ;       V03-009 WMC0001         Wayne Cardoza           23-Apr-1984
0000    33 ;               Add a comment warning about general use of EXE$IORSNWAIT.
0000    34 ;
0000    35 ;       V03-008 ROW0259         Ralph O. Weber          20-NOV-1983
0000    36 ;               For IO$_PACKACK operations passing through EXE$LCLDSKVALID,
0000    37 ;               always allow the PACKACK request to go to the driver when
0000    38 ;               UCB$V_VALID in UCB$L_STS is clear, regardless of any other
0000    39 ;               conditions.  However, UCB$V_LCL_VALID and UCB$B_ONLCNT must
0000    40 ;               still be correctly adjusted.  This is believed to allow
0000    41 ;               PACKACKs to fail and be retried.
0000    42 ;
0000    43 ;       V03-007 SSA00002        Stan Amway              30-SEP-1983
0000    44 ;               Modified BACKOUT_QIO to call new routine PMS$ABORT_RQ
0000    45 ;               to insure complete traces of I/O activity.
0000    46 ;
0000    47 ;       V03-006 ROW0224         Ralph O. Weber          15-SEP-1983
0000    48 ;               Change EXE$LCLDSKVALID to alter UCB$B_ONLCNT either up or down
0000    49 ;               only if the local processor has not already performed such an
0000    50 ;               alteration.  Use UCB$V_LCL_VALID in UCB$L_STS to determine
0000    51 ;               state of device with respect to the local processor.
0000    52 ;
0000    53 ;       V03-005 PRD0030         Paul R. DeStefano       09-Sep-1983
0000    54 ;               Added EXE$LCLDSKVALID routine to track disk online count
0000    55 ;               and local valid status.
0000    56 ;
0000    57 ;       V03-004 ROW0192         Ralph O. Weber          20-AUG-1983
```

```
                                J 9
0000     58 ;                  Fix EXE$WRITE and EXE$READ to allow longword byte counts.
0000     59 ;                  This should allow virtual disk transfers to exceed 65K bytes.
0000     60 ;                  (This will be distributed in V3.5 as SYS ECO 65.)
0000     61 ;
0000     62 ;          V03-003 ROW49973        Ralph O. Weber          29-OCT-1982
0000     63 ;                  Change calling requirements for EXE$IORSNWAIT from an entry
0000     64 ;                  IPL of IPL$_SYNCH to an entry IPL of IPL$_ASTDEL.  Have the
0000     65 ;                  call to BACKOUT_QIO made at IPL$_ASTDEL.  Then, raise to
0000     66 ;                  IPL$_SYNCH to perform scheduler operations.  This eliminates
0000     67 ;                  undesirable page faults at an IPL above IPL$_ASTDEL when
0000     68 ;                  BACKOUT_QIO references a channel control block.
0000     69 ;
0000     70 ;          V03-002 ROW49577        Ralph O. Weber          27-SEP-1982
0000     71 ;          ECO 25  Change EXE$SETCHAR and EXE$SETMODE to return SS$_ILLIOFUNC if
0000     72 ;                  UCB$B_DEVCLASS equals DC$_DISK.  This is to prohibit SETMODE
0000     73 ;                  (set mode) and SETCHAR (set characteristics) functions on disk
0000     74 ;                  devices.  On disk devices, those functions overwrite the disk
0000     75 ;                  geometry information which results in abbarant system
0000     76 ;                  behavior.
0000     77 ;
0000     78 ; SYSTEM SERVICE QUEUE I/O FUNCTION DECISION TABLE SUBROUTINES
0000     79 ;
0000     80 ; MACRO LIBRARY CALLS
0000     81 ;
0000     82
0000     83          $ACBDEF                         ;DEFINE ACB OFFSETS
0000     84          $CCBDEF                         ;DEFINE CCB OFFSETS
0000     85          $DCDEF                          ;DEFINE DEVICE CLASSES
0000     86          $DEVDEF                         ;DEFINE DEVICE CHARACTERISTICS
0000     87          $IODEF                          ;DEFINE I/O FUNCTION CODES
0000     88          $IPLDEF                         ;DEFINE SYSTEM IPLS
0000     89          $IRPDEF                         ;DEFINE IRP OFFSETS
0000     90          $PCBDEF                         ;DEFINE PCB VALUES
0000     91          $PRDEF                          ;DEFINE PROCESSOR REGISTERS
0000     92          $SSDEF                          ;DEFINE SYSTEM STATUS VALUES
0000     93          $UCBDEF                         ;DEFINE UCB OFFSETS
0000     94          $VADEF                          ;DEFINE VIRTUAL ADDRESS FIELDS
0000     95          $SFDEF                          ;DEFINE CALL FRAME
0000     96
0000     97 ;
0000     98 ; LOCAL SYMBOLS
0000     99 ;
0000    100 ; ARGUMENT LIST OFFSET DEFINITIONS
0000    101 ;
0000    102
00000000 0000    103 P1=0                            ;FIRST FUNCTION DEPENDENT PARAMETER
00000004 0000    104 P2=4                            ;SECOND FUNCTION DEPENDENT PARAMETER
00000008 0000    105 P3=8                            ;THIRD FUNCTION DEPENDENT PARAMETER
0000000C 0000    106 P4=12                           ;FOURTH FUNCTION DEPENDENT PARAMETER
00000010 0000    107 P5=16                           ;FIFTH FUNCTION DEPENDENT PARAMETER
00000014 0000    108 P6=20                           ;SIXTH FUNCTION DEPENDENT PARAMETER
```

```
                                        .SBTTL  PLACE PROCESS IN I/O RESOURCE WAIT
                                ;+
                                ; EXE$IORSNWAIT - PLACE PROCESS IN I/O RESOURCE WAIT
                                ;
                                ; FUNCTIONAL DESCRIPTION:
                                ;
                                ; THIS ROUTINE IS USED BY FDT PROCEEDURES TO RE-START A QIO REQUEST
                                ; AFTER A RESOURCE WAIT. THE CURRENT I/O IS CLEANED UP AND THE PRE-QIO
                                ; STACK IS SET UP. THEN THE PROCESS IS PLACED IN THE WAIT STATE.
                                ; IF THE PROCESS DOES NOT HAVE RESOURCE WIAT ENABLED, THE I/O IS ABORTED
                                ; WITH A STATUS SPECIFIED BY THE CALLER.
                                ;
                                ; **CAUTION** THIS ROUTINE IS NOT CALLED AT SYNCH SO THE RESOURCE MAY ALREADY
                                ; HAVE BEEN DECLARED AVAILABLE.  THIS ROUTINE SHOULD ONLY BE USED FOR RESOURCES
                                ; WHICH ARE GUARANTEED TO BE PERIODICALLY (TIMESCHDL) DECLARED AVAILABLE.
                                ;
                                ;
                                ; IMPLICIT INPUTS:
                                ;
                                ;       CALLER MUST BE AT IPL=IPL$_ASTDEL
                                ;
                                ; INPUTS:
                                ;
                                ;       R0 = STATUS TO RETURN IF NO WAIT REQUESTED
                                ;       R1 = RESOURCE NUMBER TO WAIT FOR
                                ;       R3 = ADDRESS OF CURRENT PACKET
                                ;       R4 = ADDRESS OF THE CURRENT PCB
                                ;       R6 = ADDRESS OF CHANNEL CONTROL BLOCK
                                ;
                                ; OUTPUTS:
                                ;
                                ;       R0,R1,R2,R3 ARE USED.
                                ;
                                ;       CONTROL IS TRANSFERED TO EXE$ABORTIO IF NO RESOURCE WAIT
                                ;       HAS BEEN REQUESTED,
                                ;
                                ;       OR TO SCH$WAIT IF RESOURCE WAIT IS REQUESTED.
                                ;
                                ;-
                                
                                EXE$IORSNWAIT::                         ;PLACE PROCESS IN RESOURCE WAIT
21 24 A4    0A    E0          BBS     #PCB$V_SSRWAIT,PCB$L_STS(R4),50$ ;BR IF NO WAIT REQUEST
             51    DD          PUSHL   R1                              ;REMEMBER RESOURCE NUMBER
          00F5    30          BSBW    BACKOUT_QIO                     ;CLEANUP QIO
                              SETIPL  #IPL$_SYNCH                     ;SYNCHRONIZE WITH SCHEDULER DATABASE
   4C A4    6E    9A          MOVZBL  (SP),PCB$L_EFWM(R4)             ;SET UP WAIT MARKER
00 0000'CF    8E    E2        BBSS    (SP)+,W^SCH$GL_RESMASK,30$ ;INDICATE PROCESS IS WAITING
   5C    08 AD    D0          MOVL    SF$L_SAVE_AP(FP),AP             ;RESTORE PRE-QIO ARGUMENT LIST POINTER
      5E    5D    D0          MOVL    FP,SP                           ;CLEAN STACK BACK TO CALL FRAME
   52   0000'CF    7E          MOVAQ   W^SCH$GQ_MWAIT,R2               ;ADDRESS WAIT LIST
          FFDA'    31          BRW     SCH$WAIT                        ;PLACE PROCESS IN WAIT STATE
                              ;
                              ; NO RESOURCE WAIT REQUESTED - ABORT THE I/O
                              ;
          FFD7'    31        50$:    BRW     EXE$ABORTIO               ;
```

```
                     0029   166          .SBTTL  ONE PARAMETER FUNCTION PROCESSING
                     0029   167 ;+
                     0029   168 ; EXE$ONEPARM - ONE PARAMETER FUNCTION PROCESSING
                     0029   169 ;
                     0029   170 ; THIS ROUTINE IS CALLED FROM THE FUNCTION DECISION TABLE DISPATCHER TO
                     0029   171 ; PROCESS A ONE PARAMETER FUNCTION THAT REQUIRES NO SPECIAL CHECKING.
                     0029   172 ;
                     0029   173 ; INPUTS:
                     0029   174 ;
                     0029   175 ;       R0 = SCRATCH.
                     0029   176 ;       R1 = SCRATCH.
                     0029   177 ;       R2 = SCRATCH.
                     0029   178 ;       R3 = ADDRESS OF I/O REQUEST PACKET.
                     0029   179 ;       R4 = CURRENT PROCESS PCB ADDRESS.
                     0029   180 ;       R5 = ASSIGNED DEVICE UCB ADDRESS.
                     0029   181 ;       R6 = ADDRESS OF CCB.
                     0029   182 ;       R7 = I/O FUNCTION CODE BIT NUMBER.
                     0029   183 ;       R8 = FUNCTION DECISION TABLE DISPATCH ADDRESS.
                     0029   184 ;       R9 = SCRATCH.
                     0029   185 ;       R10 = SCRATCH.
                     0029   186 ;       R11 = SCRATCH.
                     0029   187 ;       AP = ADDRESS OF FIRST FUNCTION DEPENDENT PARAMETER.
                     0029   188 ;
                     0029   189 ; OUTPUTS:
                     0029   190 ;
                     0029   191 ; ***TBS***
                     0029   192 ;-
                     0029   193
                     0029   194          .ENABL  LSB
                     0029   195 EXE$ONEPARM::                                ;ONE PARAMETER FUNCTION PROCESSING
 38 A3   6C   D0     0029   196          MOVL    P1(AP),IRP$L_MEDIA(R3)      ;STORE PARAMETER IN MEDIA ADDRESS
         03   11     002D   197          BRB     10$                         ;
```

```
                     002F   199          .SBTTL  ZERO PARAMETER FUNCTION PROCESSING
                     002F   200 ;+
                     002F   201 ; EXE$ZEROPARAM - ZERO PARAMETER FUNCTION PROCESSING
                     002F   202 ;
                     002F   203 ; THIS ROUTINE IS CALLED FROM THE FUNCTION DECISION TABLE DISPATCHER TO
                     002F   204 ; PROCESS A ZERO PARAMETER FUNCTION THAT REQUIRES NO ADDITION CHECKING.
                     002F   205 ;
                     002F   206 ; INPUTS:
                     002F   207 ;
                     002F   208 ;       R0 = SCRATCH.
                     002F   209 ;       R1 = SCRATCH.
                     002F   210 ;       R2 = SCRATCH.
                     002F   211 ;       R3 = ADDRESS OF I/O REQUEST PACKET.
                     002F   212 ;       R4 = CURRENT PROCESS PCB ADDRESS.
                     002F   213 ;       R5 = ASSIGNED DEVICE UCB ADDRESS.
                     002F   214 ;       R6 = ADDRESS OF CCB.
                     002F   215 ;       R7 = I/O FUNCTION CODE BIT NUMBER.
                     002F   216 ;       R8 = FUNCTION DECISION TABLE DISPATCH ADDRESS.
                     002F   217 ;       R9 = SCRATCH.
                     002F   218 ;       R10 = SCRATCH.
                     002F   219 ;       R11 = SCRATCH.
                     002F   220 ;       AP = ADDRESS OF FIRST FUNCTION DEPENDENT PARAMETER.
                     002F   221 ;
                     002F   222 ; OUTPUTS:
                     002F   223 ;
                     002F   224 ; ***TBS***
                     002F   225 ;-
                     002F   226
                     002F   227 EXE$ZEROPARM::                          ;ZERO PARAMETER FUNCTION PROCESSING
38 A3   D4           002F   228          CLRL    IRP$L_MEDIA(R3)       ;CLEAR PARAMETER
 FFCB'  31           0032   229 10$:     BRW     EXE$QIODRVPKT         ;QUEUE I/O PACKET TO DRIVER
                     0035   230          .DSABL  LSB
```

```
                     0035   232          .SBTTL  LOCAL DISK VALID FUNCTION PROCESSING
                     0035   233 ;+
                     0035   234 ; EXE$LCLDSKVALID - LOCAL DISK VALID FUNCTION FROCESSING
                     0035   235 ;
                     0035   236 ;       This routine is called from the function decision table dispatcher to
                     0035   237 ;       process functions which affect the online count and local valid status
                     0035   238 ;       of a disk.
                     0035   239 ;
                     0035   240 ;       If the function is the first local pack acknowedge function
                     0035   241 ;       (UCB$V_LCL_VALID is clear), the online count, UCB$B_ONLCNT, is
                     0035   242 ;       incremented and UCB$V_LCL_VALID is set.  If the online count was
                     0035   243 ;       previously zero, the I/O packet is queued to the driver for further
                     0035   244 ;       PACKACK processing.  If the online count was not previously zero but
                     0035   245 ;       the UCB$V_VALID bit is clear, the I/O packet is also queued to the
                     0035   246 ;       driver for further processing.
                     0035   247 ;
                     0035   248 ;       If the function is the first local available or unload function
                     0035   249 ;       (UCB$V_LCL_VALID is set), the online count, UCB$B_ONLCNT, is
                     0035   250 ;       decremented and UCB$V_LCL_VALID is cleared.  If the decremented online
                     0035   251 ;       count is zero, the I/O packet is queued to the driver for further
                     0035   252 ;       AVAILABLE or UNLOAD processing.
                     0035   253 ;
                     0035   254 ; INPUTS:
                     0035   255 ;
                     0035   256 ;       R0 = SCRATCH.
                     0035   257 ;       R3 = ADDRESS OF I/O REQUEST PACKET.
                     0035   258 ;       R5 = ASSIGNED DEVICE UCB ADDRESS.
                     0035   259 ;       R7 = I/O FUNCTION CODE BIT NUMBER.
                     0035   260 ;
                     0035   261 ; OUTPUTS:
                     0035   262 ;
                     0035   263 ;       UCB$B_ONLCNT is altered to reflect the number of hosts which have set
                     0035   264 ;       the drive online (i.e. issued PACKACK functions to the drive).
                     0035   265 ;
                     0035   266 ;       UCB$V_LCL_VALID in UCB$L_STS is set for PACKACK functions and cleared
                     0035   267 ;       for AVAILABLE or UNLOAD functions.
                     0035   268 ;-
                     0035   269
                     0035   270 EXE$LCLDSKVALID::                          ; LOCAL DISK VALID FUNCTION PROCESSING.
                     0035   271
      08   57   91   0035   272          CMPB    R7, #IO$_PACKACK          ; Pack acknowledge function?
           1B   12   0038   273          BNEQ    50$                       ; Branch if not a PACKACK.
0E 64 A5   11   E2   003A   274          BBSS    #UCB$V_LCL_VALID, -       ; Is this the first local PACKACK?
                     003F   275                  UCB$L_STS(R5), 20$        ; Branch if not first local PACKACK.
                     003F   276          SETIPL  #IPL$_SCS                 ; Synchronize with the MSCP server.
    00AE C5   96   0042   277          INCB    UCB$B_ONLCNT(R5)          ; Increment online count.
 01 00AE C5   91   0046   278          CMPB    UCB$B_ONLCNT(R5), #1      ; Is this the first cluster PACKACK?
           05   13   004B   279          BEQL    30$                       ; Branch if first cluster PACKACK.
11 64 A5   0B   E0   004D   280 20$:     BBS     #UCB$V_VALID, -           ; Is the volume already valid?
                     0052   281                  UCB$L_STS(R5), 80$        ; Branch if volume is already valid.
                     0052   282
                     0052   283
         FFAB'   31   0052   284 30$:     BRW     EXE$QIODRVPKT             ; For first cluster PACKACK, last
                     0055   285                                            ; cluster UNLOAD or AVAILABLE, or
                     0055   286                                            ; truely invalid volume, ask driver
                     0055   287                                            ; to really perform the function.
                     0055   288
```

```
                                B 10
                     0055   289 50$:                                ; UNLOAD and AVAILABLE come here.
09 64 A5   11   E5   0055   290          BBCC    #UCB$V_LCL_VALID, -  ; First local UNLOAD or AVAILABLE?
                     005A   291                  UCB$L_STS(R5), 80$   ; Branch if not first.
                     005A   292          SETIPL  #IPL$_SCS            ; Synchronize with MSCP server.
     00AE C5   97    005D   293          DECB    UCB$B_ONLCNT(R5)     ; Decrement online count.
           EF   13   0061   294          BEQL    30$                  ; Branch if the online count is zero.
                     0063   295
                     0063   296 80$:                                ; For requests which are not being
                     0063   297                                      ; passed on to the driver.
     50    01   3C   0063   298          MOVZWL  #SS$_NORMAL, R0      ; Set normal completion status.
        FF97'   31   0066   299          BRW     EXE$FINISHIOC        ; Finish I/O operation.
```

```
                     0069   301          .SBTTL  READ AND WRITE FUNCTION PROCESSING
                     0069   302 ;+
                     0069   303 ; EXE$READ - READ FUNCTION PROCESSING
                     0069   304 ; EXE$WRITE - WRITE FUNCTION PROCESSING
                     0069   305 ; EXE$MODIFY - MODIFY FUNCTION PROCESSING
                     0069   306 ;
                     0069   307 ; THESE ROUTINES ARE CALLED FROM THE FUNCTION DECISION TABLE DISPATCHER TO
                     0069   308 ; PROCESS A READ OR WRITE PHYSICAL OR LOGICAL FUNCTION.
                     0069   309 ; EXE$MODIFY IS USED FOR FUNCTIONS THAT READ AND WRITE MEMORY.
                     0069   310 ;
                     0069   311 ; INPUTS:
                     0069   312 ;
                     0069   313 ;       R0 = SCRATCH.
                     0069   314 ;       R1 = SCRATCH.
                     0069   315 ;       R2 = SCRATCH.
                     0069   316 ;       R3 = ADDRESS OF I/O REQUEST PACKET.
                     0069   317 ;       R4 = CURRENT PROCESS PCB ADDRESS.
                     0069   318 ;       R5 = ASSIGNED DEVICE UCB ADDRESS.
                     0069   319 ;       R6 = ADDRESS OF CCB.
                     0069   320 ;       R7 = I/O FUNCTION CODE BIT NUMBER.
                     0069   321 ;       R8 = FUNCTION DECISION TABLE DISPATCH ADDRESS.
                     0069   322 ;       R9 = SCRATCH.
                     0069   323 ;       R10 = SCRATCH.
                     0069   324 ;       R11 = SCRATCH.
                     0069   325 ;       AP = ADDRESS OF FIRST FUNCTION DEPENDENT PARAMETER.
                     0069   326 ;
                     0069   327 ; OUTPUTS:
                     0069   328 ;
                     0069   329 ; ***TBS***
                     0069   330 ;-
                     0069   331
                     0069   332          .ENABL  LSB
                     0069   333 EXE$MODIFY::                                ;MODIFY FUNCTION PROCESSING
    52   A1'AF   DE  0069   334          MOVAL   B^EXE$MODIFYLOCK,R2        ;SET ADDRESS OF BUFFER CHECK ROUTINE
         04   11    006D   335          BRB     5$
                     006F   336 EXE$READ::                                  ;READ FUNCTION PROCESSING
    52   9B'AF   DE  006F   337          MOVAL   B^EXE$READLOCK,R2          ;SET ADDRESS OF BUFFER CHECK ROUTINE
04 2A A3    01   E3  0073   338 5$:      BBCS    #IRP$V_FUNC,IRP$W_STS(R3),10$ ;SET READ FUNCTION STATUS
                     0078   339 EXE$WRITE::                                 ;WRITE FUNCTION PROCESSING
    52   9E'AF   DE  0078   340          MOVAL   B^EXE$WRITELOCK,R2         ;SET ADDRESS OF BUFFER CHECK ROUTINE
3C A3    0C AC   D0  007C   341 10$:     MOVL    P4(AP),IRP$B_CARCON(R3)    ;INSERT CARRIAGE CONTROL BYTE
      06    00   ED  0081   342          CMPZV   #IRP$V_FCODE,#IRP$S_FCODE,- ;PHYSICAL I/O FUNCTION?
    1F   20 A3       0084   343                  IRP$W_FUNC(R3),#IO$_PHYSICAL ;
         04   15    0087   344          BLEQ    20$                        ;IF LEQ YES
         15   A2    0089   345          SUBW    #IO$_READLBLK-IO$_READPBLK,- ;CONVERT TO PHYSICAL FUNCTION
      20 A3       008B   346                  IRP$W_FUNC(R3)             ;
    51   04 AC   D0  008D   347 20$:     MOVL    P2(AP),R1                  ;GET NUMBER OF BYTES TO TRANSFER
         05   13    0091   348          BEQL    30$                        ;IF EQL NONE
      50   6C   D0  0093   349          MOVL    P1(AP),R0                  ;GET STARTING VIRTUAL ADDRESS OF TRANSFER
         62   16    0096   350          JSB     (R2)                       ;CHECK BUFFER AND LOCK IN MEMORY
        FF65'   31  0098   351 30$:     BRW     EXE$QIODRVPKT              ;QUEUE I/O PACKET TO DRIVER
                     009B   352          .DSABL  LSB
```

```
                     009B   354          .SBTTL  READ AND WRITE FUNCTION BUFFER CHECK AND LOCK ROUTINES
                     009B   355 ;+
                     009B   356 ; EXE$READLOCK - CHECK BUFFER FOR READ ACCESSIBILITY AND LOCK
                     009B   357 ; EXE$WRITELOCK - CHECK BUFFER FOR WRITE ACCESSIBILITY AND LOCK
                     009B   358 ; EXE$MODIFYLOCK - CHECK BUFFER FOR READ ACCESSIBILITY AND LOCK
                     009B   359 ;
                     009B   360 ; THESE ROUTINES ARE CALLED TO CHECK THE ACCESSIBILITY OF AN I/O BUFFER AND
                     009B   361 ; TO LOCK THE BUFFER IN MEMORY FOR A DIRECT MEMORY TRANSFER.
                     009B   362 ;
                     009B   363 ; INPUTS:
                     009B   364 ;
                     009B   365 ;       R0 = STARTING ADDRESS OF I/O BUFFER.
                     009B   366 ;       R1 = LENGTH OF TRANSFER IN BYTES.
                     009B   367 ;       R4 = CURRENT PROCESS PCB ADDRESS.
                     009B   368 ;       R6 = ADDRESS OF CCB.
                     009B   369 ;
                     009B   370 ; OUTPUTS:
                     009B   371 ;
                     009B   372 ;       THE I/O BUFFER IS CHECKED FOR THE PROPER ACCESSIBILITY. IF THE
                     009B   373 ;       CHECK SUCCEEDS, THEN THE BUFFER IS LOCKED IN MEMORY AND THE STARTING
                     009B   374 ;       ADDRESS OF THE PAGE TABLE ENTRIES THAT MAP THE TRANSFER IS STORED
                     009B   375 ;       IN THE I/O PACKET. ELSE THE I/O IS COMPLETED WITH A STATUS OF
                     009B   376 ;       ACCESS VIOLATION.
                     009B   377 ;-
                     009B   378
                     009B   379 EXE$READLOCK::                           ;CHECK BUFFER FOR READ FUNCTION AND LOCK
          11   10    009B   380          BSBB    EXE$READLOCKR          ;EXE$READLOCKR RETURNS NORMALLY ON
                     009D   381                                         ;SUCCESS, VIA COROUTINE CALL ON FAILURE
               05    009D   382          RSB                            ;RETURNS TO CALLER ON SUCCESS, TO
                     009E   383                                         ;EXE$READLOCKR ON FAILURE
                     009E   384
                     009E   385 EXE$WRITELOCK::                          ;CHECK BUFFER FOR WRITE FUNCTION AND LOCK
          15   10    009E   386          BSBB    EXE$WRITELOCKR         ;EXE$WRITELOCKR RETURNS NORMALLY ON
                     00A0   387                                         ;SUCCESS, VIA COROUTINE CALL ON FAILURE
               05    00A0   388          RSB                            ;RETURNS TO CALLER ON SUCCESS, TO
                     00A1   389                                         ;EXE$WRITELOCKR ON FAILURE
                     00A1   390
                     00A1   391 EXE$MODIFYLOCK::                         ;CHECK BUFFER FOR MODIFY FUNCTION AND LOCK
          01   10    00A1   392          BSBB    EXE$MODIFYLOCKR        ;EXE$MODIFYLOCKR RETURNS NORMALLY ON
                     00A3   393                                         ;SUCCESS, VIA COROUTINE CALL ON FAILURE
               05    00A3   394          RSB                            ;RETURNS TO CALLER ON SUCCESS, TO
                     00A4   395                                         ;EXE$MODIFYLOCKR ON FAILURE
```

```
                        00A4   397          .SBTTL  READ AND WRITE BUFFER CHECK AND LOCK AND RETURN ROUTINES
                        00A4   398 ;+
                        00A4   399 ; EXE$READLOCKR - CHECK BUFFER FOR READ ACCESSIBILITY AND LOCK AND RETURN
                        00A4   400 ;                 ON ERROR
                        00A4   401 ; EXE$WRITELOCKR - CHECK BUFFER FOR WRITE ACCESSIBILITY AND LOCK AND RETURN
                        00A4   402 ;                  ON ERROR
                        00A4   403 ; EXE$MODIFYLOCKR - CHECK BUFFER FOR READ ACCESSIBILITY AND LOCK AND RETURN
                        00A4   404 ;                   ON ERROR
                        00A4   405 ;
                        00A4   406 ; THESE ROUTINES ARE CALLED TO CHECK THE ACCESSIBILITY OF AN I/O BUFFER
                        00A4   407 ; AND TO LOCK THE BUFFER IN MEMORY FOR A DIRECT MEMORY TRANSFER.  IN
                        00A4   408 ; ADDITION, THESE ROUTINES PERFORM A COROUTINE CALL IF THERE IS AN ERROR
                        00A4   409 ; OR ANY PAGES HAVE TO BE FAULTED IN.  THE PURPOSE OF THE COROUTINE
                        00A4   410 ; CALL IS TO ALLOW THE CALLER TO PERFORM ANY NECESSARY CLEANUP BEFORE
                        00A4   411 ; THE QIO IS BACKED UP OR ABORTED.  THESE ROUTINES ARE TYPICALLY CALLED
                        00A4   412 ; BY DRIVERS THAT MUST LOCK MULTIPLE AREAS INTO MEMORY.  SINCE THESE
                        00A4   413 ; ROUTINES CANNOT UNLOCK AREAS PREVIOUSLY LOCKED, THE COROUTINE CALL ALLOWS
                        00A4   414 ; THE CALLER (THE DRIVER) TO UNLOCK PREVIOUSLY LOCKED AREAS (AND PERFORM
                        00A4   415 ; ANY OTHER CLEANUP) AND THEN RETURN HERE TO BACK UP OR ABORT THE I/O.
                        00A4   416 ;
                        00A4   417 ; EXE$MODIFYLOCKR IS USED WHEN THE BUFFER WILL BE READ AND WRITTEN BY THE
                        00A4   418 ; I/O DEVICE.  IT DISABLES AN OPTIMIZATION IN MMG$IOLOCK WHICH IS USED
                        00A4   419 ; WHEN THE BUFFER IS ONLY WRITTEN.
                        00A4   420 ;
                        00A4   421 ; INPUTS:
                        00A4   422 ;
                        00A4   423 ;       R0 = STARTING ADDRESS OF I/O BUFFER.
                        00A4   424 ;       R1 = LENGTH OF BUFFER IN BYTES.
                        00A4   425 ;       R4 = CURRENT PROCESS PCB ADDRESS.
                        00A4   426 ;       R6 = ADDRESS OF CCB.
                        00A4   427 ;
                        00A4   428 ; OUTPUTS:
                        00A4   429 ;
                        00A4   430 ;       THE I/O BUFFER IS CHECKED FOR THE PROPER ACCESSIBILITY.  IF THE
                        00A4   431 ;       CHECK SUCCEEDS, THEN THE BUFFER IS LOCKED IN MEMORY AND THE STARTING
                        00A4   432 ;       ADDRESS OF THE PAGE TABLE ENTRIES THAT MAP THE TRANSFER IS STORED
                        00A4   433 ;       IN THE I/O PACKET.
                        00A4   434 ;
                        00A4   435 ;       R0 = RETURN CODE
                        00A4   436 ;
                        00A4   437 ;       NOTE THAT IF THERE ARE NO ERRORS AND NO PAGES HAVE TO BE FAULTED
                        00A4   438 ;       IN, THEN THESE ROUTINES RETURN NORMALLY.  HOWEVER, IF THERE IS AN
                        00A4   439 ;       ERROR OR A PAGE HAS TO BE FAULTED IN, THEN THE CALLER IS CALLED
                        00A4   440 ;       BY A COROUTINE CALL.  THE CALLER'S RSB THEN RETURNS HERE WHERE
                        00A4   441 ;       THE QIO IS EITHER BACKED UP OR ABORTED.  NOTE THAT IN THIS CASE
                        00A4   442 ;       THE CALLER'S ERROR HANDLING CODE MUST PRESERVE ALL REGISTERS,
                        00A4   443 ;       INCLUDING R0 AND R1.
                        00A4   444 ;-
                        00A4   445          .ENABL  LSB
                        00A4   446 EXE$MODIFYLOCKR::                            ;CHECK BUFFER FOR MODIFY FUNCTION AND LOCK
     50   DD            00A4   447          PUSHL   R0                          ;SAVE STARTING ADDRESS OF BUFFER
   009D   30            00A6   448          BSBW    EXE$READCHKR                ;CHECK BUFFER FOR READ FUNCTION
52   04   C8            00A9   449          BISL    #4,R2                       ;DISABLE OPTIMIZATION IN MMG$IOLOCK
     0C   11            00AC   450          BRB     10$
                        00AE   451
                        00AE   452 EXE$READLOCKR::                              ;CHECK BUFFER FOR READ FUNCTION AND LOCK
     50   DD            00AE   453          PUSHL   R0                          ;SAVE STARTING ADDRESS OF BUFFER
```

```
                 0093   30  00B0   454          BSBW    EXE$READCHKR            ;CHECK BUFFER FOR READ FUNCTION
                   05   11  00B3   455          BRB     10$
                                    00B5   456
                                    00B5   457 EXE$WRITELOCKR::                 ;CHECK BUFFER FOR WRITE FUNCTION AND LOCK
                   50   DD  00B5   458          PUSHL   R0                      ;SAVE STARTING ADDRESS OF BUFFER
                 00EA   30  00B7   459          BSBW    EXE$WRITECHKR           ;CHECK BUFFER FOR WRITE FUNCTION
                1A 50   E9  00BA   460 10$:     BLBC    R0,15$                  ;BRANCH IF ERROR
                   50 8ED0  00BD   461          POPL    R0                      ;RESTORE STARTING ADDRESS OF BUFFER
        30 A3   50 FE00 8F   AB  00C0   462          BICW3   #^C<VA$M_BYTE>,R0,IRP$W_BOFF(R3) ;SET BYTE OFFSET IN PAGE
                   53   DD  00C7   463          PUSHL   R3                      ;SAVE ADDRESS OF I/O PACKET
                FF34'   30  00C9   464          BSBW    MMG$IOLOCK              ;LOCK PAGES FOR I/O
                53   8E   D0  00CC   465          MOVL    (SP)+,R3                ;RETRIEVE ADDRESS OF I/O PACKET
                  08 50   E9  00CF   466          BLBC    R0,20$                  ;IF LBC LOCK FAILURE
              2C A3   51   D0  00D2   467          MOVL    R1,IRP$L_SVAPTE(R3)     ;INSERT ADDRESS OF FIRST PTE IN PACKET
                           05  00D6   468          RSB                             ;
                5E   04   C0  00D7   469 15$:     ADDL    #4,SP                   ;THROW AWAY OLD R0
                     9E   16  00DA   470 20$:     JSB     @(SP)+                  ;COROUTINE CALL TO CLEANUP
                     50   D5  00DC   471          TSTL    R0                      ;ERRORS ENCOUNTERED?
                     1C   12  00DE   472          BNEQ    50$                     ;IF NEQ YES
                     51   DD  00E0   473          PUSHL   R1                      ;SAVE VIRTUAL ADDRESS OF PAGE TO FAULT
                     1B   10  00E2   474          BSBB    BACKOUT_QIO             ;CLEANUP QIO
                     02   BA  00E4   475          POPR    #^M<R1>                 ;RETRIEVE VIRTUAL ADDRESS OF PAGE TO FAULT
                5E   5D   D0  00E6   476          MOVL    FP,SP                   ;TRIM STACK BACK TO CHANGE MODE FRAME
              5C   08 AD   7D  00E9   477          MOVQ    8(FP),AP                ;RESTORE USER ARGUMENT AND FRAME POINTERS
                5E   00'   C0  00ED   478          ADDL    S^#EXE$C_CMSTKSZ,SP     ;REMOVE CHANGE MODE CALL FRAME FROM STACK
             50   8E   04   C3  00F0   479          SUBL3   #4,(SP)+,R0             ;CALCULATE RESTART ADDRESS
                F8'AF   9F  00F4   480          PUSHAB  B^40$                   ;SET NEW RETURN ADDRESS
                           02  00F7   481          REI                             ;
                   61   95  00F8   482 40$:     TSTB    (R1)                    ;FAULT USER BUFFER AGAIN
                   60   17  00FA   483          JMP     (R0)                    ;REPEAT SYSTEM SERVICE
                FF01'   31  00FC   484 50$:     BRW     EXE$ABORTIO             ;ABORT I/O REQUEST
                                    00FF   485          .DSABL  LSB
```

```
                     00FF    487          .SBTTL  BACKOUT A QIO
                     00FF    488 ;+
                     00FF    489 ; BACKOUT_QIO - BACKOUT A QIO
                     00FF    490 ;
                     00FF    491 ; THIS ROUTINE IS CALLED TO BACKOUT A QIO.  IT DECREMENTS THE CHANNEL I/O
                     00FF    492 ; COUNT, INCREMENTS THE DIRECT OR BUFFERED I/O COUNT, DEALLOCATES THE
                     00FF    493 ; DIAGNOSTIC BUFFER (IF PRESENT), OPTIONALLY INCREMENTS THE AST COUNT, AND
                     00FF    494 ; FINALLY DEALLOCATES THE IRP.
                     00FF    495 ;
                     00FF    496 ; INPUTS:
                     00FF    497 ;
                     00FF    498 ;       R3 = ADDRESS OF I/O REQUEST PACKET
                     00FF    499 ;       R4 = CURRENT PROCESS PCB ADDRESS
                     00FF    500 ;       R6 = ADDRESS OF CCB
                     00FF    501 ;
                     00FF    502 ; OUTPUTS:
                     00FF    503 ;
                     00FF    504 ;       R0 - R3 = CLOBBERED
                     00FF    505 ;
                     00FF    506 ;-
                     00FF    507
                     00FF    508 BACKOUT_QIO:                                ;BACKOUT A QIO
           FEFE'  30 00FF    509          BSBW    PMS$ABORT_RQ               ;RECORD ABORT IF I/O MONITORING ENABLED
        0A A6   B7 0102    510          DECW    CCB$W_IOC(R6)              ;DECREMENT CHANNEL I/O COUNT
05 2A A3   00   E1 0105    511          BBC     #IRP$V_BUFIO,IRP$W_STS(R3),10$ ;BR IF NOT BUFFERED I/O
        3A A4   B6 010A    512          INCW    PCB$W_BIOCNT(R4)           ;ADJUST COUNT OF BUFFERED I/O
           03   11 010D    513          BRB     20$                        ;CONTINUE
        3E A4   B6 010F    514 10$:     INCW    PCB$W_DIOCNT(R4)           ;ADJUST DIRECT I/O COUNT
0C 2A A3   07   E1 0112    515 20$:     BBC     #IRP$V_DIAGBUF,IRP$W_STS(R3),30$  ;BR. IF NO DIAGNOSTIC BUFFER
   50   4C A3   D0 0117    516          MOVL    IRP$L_DIAGBUF(R3),R0       ;GET ADDRESS OF DIAGNOSTIC BUFFER
           53   DD 011B    517          PUSHL   R3                         ;SAVE R3
           FEE0'  30 011D    518          BSBW    EXE$DEANONPAGED            ;DEALLOCATE DIAGNOSTIC BUFFER
           53 8ED0 0120    519          POPL    R3                         ;RESTORE R3
03 0B A3   06   E1 0123    520 30$:     BBC     #ACB$V_QUOTA,IRP$B_RMOD(R3),40$ ;BR IF AST NOT REQUESTED
        38 A4   B6 0128    521          INCW    PCB$W_ASTCNT(R4)           ;ADJUST AST COUNT
        50   53   D0 012B    522 40$:     MOVL    R3,R0                      ;DEALLOCATE PACKET
           FECF'  30 012E    523          BSBW    EXE$DEANONPAGED            ;
                  05 0131    524          RSB
```

```
                     0132   526          .SBTTL  CHECK BUFFER ACCESSIBILITY FOR READ FUNCTION
                     0132   527 ;+
                     0132   528 ; EXE$READCHK - CHECK BUFFER ACCESSIBILITY FOR READ FUNCTION
                     0132   529 ;
                     0132   530 ; THIS ROUTINE IS CALLED TO CHECK BUFFER ACCESSIBILITY FOR A READ I/O
                     0132   531 ; FUNCTION.
                     0132   532 ;
                     0132   533 ; INPUTS:
                     0132   534 ;
                     0132   535 ;       R0 = ADDRESS OF BUFFER.
                     0132   536 ;       R1 = SIZE OF TRANSFER IN BYTES.
                     0132   537 ;       R3 = ADDRESS OF I/O REQUEST PACKET.
                     0132   538 ;
                     0132   539 ; OUTPUTS:
                     0132   540 ;
                     0132   541 ;       IF BUFFER IS NOT WRITE ACCESSIBLE, THEN THE I/O REQUEST IS TERM-
                     0132   542 ;       INATED VIA EXE$IOFINISH WITH A STATUS OF SS$_ACCVIO.
                     0132   543 ;
                     0132   544 ;       IF BUFFER IS WRITE ACCESSIBLE, THEN THE FOLLOWING VALUES ARE RE-
                     0132   545 ;       TURNED:
                     0132   546 ;
                     0132   547 ;               R0 = ADDRESS OF BUFFER.
                     0132   548 ;               R1 = SIZE OF TRANSFER IN BYTES.
                     0132   549 ;               R2 = READ FUNCTION INDICATOR (1).
                     0132   550 ;               R3 = ADDRESS OF I/O REQUEST PACKET.
                     0132   551 ;
                     0132   552 ;               IRP$W_BCNT(R3) = SIZE OF TRANSFER IN BYTES.
                     0132   553 ;               IRP$W_FUNC(R3) = READ.
                     0132   554 ;-
                     0132   555
                     0132   556          .ENABL  LSB
                     0132   557 EXE$READCHK::                                ;CHECK BUFFER FOR READ FUNCTION
 50    DD            0132   558          PUSHL   R0                          ;SAVE ADDRESS OF BUFFER
 10    10            0134   559          BSBB    EXE$READCHKR                ;CHECK BUFFER
 04    11            0136   560          BRB     10$
```

```
                                                 I 10
                     0138   562          .SBTTL  CHECK BUFFER ACCESSIBILITY FOR WRITE FUNCTION
                     0138   563 ;+
                     0138   564 ; EXE$WRITECHK - CHECK BUFFER ACCESSIBILITY FOR WRITE FUNCTION
                     0138   565 ;
                     0138   566 ; THIS ROUTINE IS CALLED TO CHECK BUFFER ACCESSIBILITY FOR A WRITE I/O
                     0138   567 ; FUNCTION.
                     0138   568 ;
                     0138   569 ; INPUTS:
                     0138   570 ;
                     0138   571 ;       R0 = ADDRESS OF BUFFER.
                     0138   572 ;       R1 = SIZE OF TRANSFER IN BYTES.
                     0138   573 ;       R3 = ADDRESS OF I/O REQUEST PACKET.
                     0138   574 ;
                     0138   575 ; OUTPUTS:
                     0138   576 ;
                     0138   577 ;       IF BUFFER IS NOT READ ACCESSIBLE, THEN THE I/O REQUEST IS TERM-
                     0138   578 ;       INATED VIA EXE$IOFINISH WITH A STATUS OF SS$_ACCVIO.
                     0138   579 ;
                     0138   580 ;       IF BUFFER IS READ ACCESSIBLE, THEN THE FOLLOWING VALUES ARE RE-
                     0138   581 ;       TURNED:
                     0138   582 ;
                     0138   583 ;               R0 = ADDRESS OF BUFFER.
                     0138   584 ;               R1 = SIZE OF TRANSFER IN BYTES.
                     0138   585 ;               R2 = WRITE FUNCTION INDICATOR (0).
                     0138   586 ;               R3 = ADDRESS OF I/O REQUEST PACKET.
                     0138   587 ;
                     0138   588 ;               IRP$W_BCNT(R3) = SIZE OF TRANSFER IN BYTES.
                     0138   589 ;               IRP$W_FUNC(R3) = WRITE.
                     0138   590 ;-
                     0138   591
                     0138   592 EXE$WRITECHK::                          ;CHECK BUFFER FOR WRITE FUNCTION
          50    DD   0138   593          PUSHL   R0                     ;SAVE ADDRESS OF BUFFER
          68    10   013A   594          BSBB    EXE$WRITECHKR          ;CHECK BUFFER
       03 50    E8   013C   595 10$:     BLBS    R0,20$                 ;BRANCH IF SUCCESS
        FEBE'   31   013F   596          BRW     EXE$ABORTIO            ;ABORT I/O
          50 8ED0   0142   597 20$:     POPL    R0                     ;RESTORE ADDRESS OF BUFFER
                05   0145   598          RSB
                     0146   599          .DSABL  LSB
```

```
                     0146   601          .SBTTL  CHECK BUFFER ACCESSIBILITY FOR READ FUNCTION AND RETURN
                     0146   602 ;+
                     0146   603 ; EXE$READCHKR - CHECK BUFFER ACCESSIBILITY FOR READ FUNCTION AND RETURN
                     0146   604 ;
                     0146   605 ; THIS ROUTINE IS CALLED TO CHECK BUFFER ACCESSIBILITY FOR A READ I/O
                     0146   606 ; FUNCTION.  STATUS IS RETURNED IN R0.
                     0146   607 ;
                     0146   608 ; INPUTS:
                     0146   609 ;
                     0146   610 ;       R0 = ADDRESS OF BUFFER.
                     0146   611 ;       R1 = SIZE OF TRANSFER IN BYTES.
                     0146   612 ;       R3 = ADDRESS OF I/O REQUEST PACKET.
                     0146   613 ;
                     0146   614 ; OUTPUTS:
                     0146   615 ;
                     0146   616 ;       IF THE BUFFER IS NOT WRITE ACCESSIBLE, THEN THE FOLLOWING
                     0146   617 ;       VALUE IS RETURNED:
                     0146   618 ;
                     0146   619 ;       R0 = SS$_ACCVIO
                     0146   620 ;
                     0146   621 ;       IF BUFFER IS WRITE ACCESSIBLE, THEN THE FOLLOWING VALUES ARE RE-
                     0146   622 ;       TURNED:
                     0146   623 ;
                     0146   624 ;               R0 = SS$_NORMAL
                     0146   625 ;               R1 = SIZE OF TRANSFER IN BYTES.
                     0146   626 ;               R2 = READ FUNCTION INDICATOR (1).
                     0146   627 ;               R3 = ADDRESS OF I/O REQUEST PACKET.
                     0146   628 ;
                     0146   629 ;               IRP$L_BCNT(R3) = SIZE OF TRANSFER IN BYTES.
                     0146   630 ;               IRP$W_FUNC(R3) = READ.
                     0146   631 ;-
                     0146   632
                     0146   633          .ENABL  LSB
                     0146   634 EXE$READCHKR::                               ; CHECK BUFFER FOR READ FUNCTION
  32 A3    51    D0  0146   635          MOVL    R1,IRP$L_BCNT(R3)           ; SAVE R1
        0F    10    014A   636          BSBB    10$                         ; CHECK ACCESS
  51    32 A3    D0  014C   637          MOVL    IRP$L_BCNT(R3),R1           ; RESTORE R1
     07 50    E9    0150   638          BLBC    R0,5$                       ; IF LBC, NO ACCESS
  2A A3    02    A8  0153   639          BISW    #IRP$M_FUNC,IRP$W_STS(R3)   ; SET READ FUNCTION
     52    01    D0  0157   640          MOVL    #1,R2                       ; SET READ FUNCTION INDICATOR
                     05  015A   641 5$:     RSB
                         015B   642
     51    50    C0  015B   643 10$:     ADDL    R0,R1                       ; ENDING ADDRESS OF BUFFER
50   01FF 8F    AA  015E   644          BICW    #VA$M_BYTE,R0               ; TRUNCATE TO START OF PAGE
     51    50    C2  0163   645          SUBL    R0,R1                       ; CALCULATE LENGTH OF BUFFER TO PROBE
52   FE00 8F    32  0166   646          CVTWL   #-^X200,R2                  ; SET ADDRESS ADJUSTMENT CONSTANT
     51    51    F7  016B   647 15$:     CVTLW   R1,R1                       ; GREATER THAN 32k?
           13    1D  016E   648          BVS     30$                         ; IF VS, YES; CHECK BY CHUNKS
                         0170   649
                         0170   650 20$:     IFNOWRT R1,(R0),ACCVIO          ; CAN ENDS OF USER'S BUFFER BE WRITTEN?
     50    52    C2  0176   651          SUBL    R2,R0                       ; CALCULATE VA OF NEXT PAGE
  51    6142 3E    0179   652          MOVAW   (R1)[R2],R1                 ; CALCULATE NEW LENGTH
           F1    14  017D   653          BGTR    20$                         ; IF GTR THEN MORE TO TEST
     50    01    3C  017F   654          MOVZWL  #SS$_NORMAL,R0              ; INDICATE SUCCESS
                     05  0182   655          RSB                             ;  AND RETURN
                         0183   656
     7E    50    7D  0183   657 30$:     MOVQ    R0,-(SP)                    ; SAVE CURRENT VALUES ON STACK
```

```
                51   7E00 8F    3C   0186   658          MOVZWL  #^X7E00,R1              ; SIZE OF CHUNK USED STEPPING THRU BUF.
                                     018B   659                                         ;  (32K - 1 PAGE)
                     6E   51    C0   018B   660          ADDL    R1,(SP)               ; ADVANCE ADDRESS BY THIS AMOUNT
                  04 AE   51    C2   018E   661          SUBL    R1,4(SP)              ; DECREASE COUNT
                          DC    10   0192   662          BSBB    20$                   ; PROBE CHUNK
                       05 50    E9   0194   663          BLBC    R0,ACCVIO1            ; IF LBC, NO ACCESS
                     50   8E    7D   0197   664          MOVQ    (SP)+,R0              ; POP PRE-ADJUSTED VALUES OFF STACK
                          CF    11   019A   665          BRB     15$                   ; SEE IF LENGTH NOW LT 32K
                                     019C   666
                                     019C   667 ACCVIO1:
                     5E   08    C0   019C   668          ADDL    #8,SP
                                05   019F   669          RSB
                                     01A0   670 ACCVIO:
                     50   0C    3C   01A0   671          MOVZWL  #SS$_ACCVIO,R0
                                05   01A3   672          RSB
                                     01A4   673
```

```
                                01A4   675          .SBTTL  CHECK BUFFER ACCESSIBILITY FOR WRITE FUNCTION AND RETURN
                                01A4   676 ;+
                                01A4   677 ; EXE$WRITECHKR - CHECK BUFFER ACCESSIBILITY FOR WRITE FUNCTION AND RETURN
                                01A4   678 ;
                                01A4   679 ; THIS ROUTINE IS CALLED TO CHECK BUFFER ACCESSIBILITY FOR A WRITE I/O
                                01A4   680 ; FUNCTION.  STATUS IS RETURNED IN R0
                                01A4   681 ;
                                01A4   682 ; INPUTS:
                                01A4   683 ;
                                01A4   684 ;       R0 = ADDRESS OF BUFFER.
                                01A4   685 ;       R1 = SIZE OF TRANSFER IN BYTES.
                                01A4   686 ;       R3 = ADDRESS OF I/O REQUEST PACKET.
                                01A4   687 ;
                                01A4   688 ; OUTPUTS:
                                01A4   689 ;
                                01A4   690 ;       IF BUFFER IS NOT READ ACCESSIBLE, THEN THE FOLLOWING VALUE IS
                                01A4   691 ;       RETURNED:
                                01A4   692 ;
                                01A4   693 ;       R0 = SS$_ACCVIO
                                01A4   694 ;
                                01A4   695 ;       IF BUFFER IS READ ACCESSIBLE, THEN THE FOLLOWING VALUES ARE RE-
                                01A4   696 ;       TURNED:
                                01A4   697 ;
                                01A4   698 ;               R0 = SS$_NORMAL
                                01A4   699 ;               R1 = SIZE OF TRANSFER IN BYTES.
                                01A4   700 ;               R2 = WRITE FUNCTION INDICATOR (0).
                                01A4   701 ;               R3 = ADDRESS OF I/O REQUEST PACKET.
                                01A4   702 ;
                                01A4   703 ;               IRP$L_BCNT(R3) = SIZE OF TRANSFER IN BYTES.
                                01A4   704 ;               IRP$W_FUNC(R3) = WRITE.
                                01A4   705 ;-
                                01A4   706
                                01A4   707 EXE$WRITECHKR::                              ;CHECK BUFFER FOR WRITE FUNCTION
  32 A3    51    D0            01A4   708          MOVL    R1,IRP$L_BCNT(R3)           ; SAVE R1
           0A    10            01A8   709          BSBB    40$                         ; CHECK ACCESS
  51    32 A3    D0            01AA   710          MOVL    IRP$L_BCNT(R3),R1           ; RESTORE R1
        02 50    E9            01AE   711          BLBC    R0,35$                      ; IF LBC, NO ACCESS
           52    D4            01B1   712          CLRL    R2                          ; SET WRITE FUNCTION INDICATOR
                 05            01B3   713 35$:     RSB
                               01B4   714
     51    50    C0            01B4   715 40$:     ADDL    R0,R1                       ; ENDING ADDRESS OF BUFFER
50   01FF 8F    AA            01B7   716          BICW    #VA$M_BYTE,R0               ; TRUNCATE TO START OF PAGE
     51    50    C2            01BC   717          SUBL    R0,R1                       ; CALCULATE LENGTH OF BUFFER TO PROBE
52   FE00 8F    32            01BF   718          CVTWL   #-^X200,R2                  ; SET ADDRESS ADJUSTMENT CONSTANT
     51    51    F7            01C4   719 45$:     CVTLW   R1,R1                       ; GREATER THAN 32k?
           13    1D            01C7   720          BVS     60$                         ; IF VS, YES; CHECK BY CHUNKS
                               01C9   721
                               01C9   722 50$:     IFNORD  R1,(R0),ACCVIO              ; CAN ENDS OF USER'S BUFFER BE READ?
     50    52    C2            01CF   723          SUBL    R2,R0                       ; CALCULATE VA OF NEXT PAGE
  51    6142    3E            01D2   724          MOVAW   (R1)[R2],R1                 ; CALCULATE NEW LENGTH
           F1    14            01D6   725          BGTR    50$                         ; IF GTR THEN MORE TO TEST
     50    01    3C            01D8   726          MOVZWL  #SS$_NORMAL,R0              ; INDICATE SUCCESS
                 05            01DB   727          RSB                                 ;  AND RETURN
                               01DC   728
     7E    50    7D            01DC   729 60$:     MOVQ    R0,-(SP)                    ; SAVE CURRENT VALUES ON STACK
51   7E00 8F    3C            01DF   730          MOVZWL  #^X7E00,R1                  ; SIZE OF CHUNK USED STEPPING THRU BUF.
                               01E4   731                                             ;  (32K - 1 PAGE)
```

```
                                                M 10
                6E    51    C0  01E4   732          ADDL    R1,(SP)                 ; ADVANCE ADDRESS BY THIS AMOUNT
             04 AE    51    C2  01E7   733          SUBL    R1,4(SP)                ; DECREASE COUNT
                      DC    10  01EB   734          BSBB    50$                     ; PROBE CHUNK
                   AC 50    E9  01ED   735          BLBC    R0,ACCVIO1              ; IF LBC, NO ACCESS
                50    8E    7D  01F0   736          MOVQ    (SP)+,R0                ; POP PRE-ADJUSTED VALUES OFF STACK
                      CF    11  01F3   737          BRB     45$                     ; SEE IF LENGTH NOW LT 32K
                                01F5   738          .DSABL  LSB
```

```
                     01F5   740          .SBTTL  SET DEVICE MODE AND CHARACTERISTICS FUNCTIONS (AT FDT LEVEL)
                     01F5   741 ;+
                     01F5   742 ; EXE$SETCHAR - SET DEVICE MODE AND CHARACTERISTICS FUNCTIONS (AT FDT LEVEL)
                     01F5   743 ;
                     01F5   744 ; THIS ROUTINE PLACES THE NEW CHARACTERISTICS SPECIFIED BY THE QUADWORD POINTED
                     01F5   745 ; TO BY P1 INTO THE SECOND AND THIRD LONGWORDS OF THE DEVICE UCB.
                     01F5   746 ;
                     01F5   747 ; INPUTS:
                     01F5   748 ;
                     01F5   749 ;       R0 = SCRATCH.
                     01F5   750 ;       R1 = SCRATCH.
                     01F5   751 ;       R2 = SCRATCH.
                     01F5   752 ;       R3 = ADDRESS OF I/O REQUEST PACKET.
                     01F5   753 ;       R4 = CURRENT PROCESS PCB ADDRESS.
                     01F5   754 ;       R5 = ASSIGNED DEVICE UCB ADDRESS.
                     01F5   755 ;       R6 = ADDRESS OF CCB.
                     01F5   756 ;       R7 = I/O FUNCTION CODE BIT NUMBER.
                     01F5   757 ;       R8 = FUNCTION DECISION TABLE DISPATCH ADDRESS.
                     01F5   758 ;       R9 = SCRATCH.
                     01F5   759 ;       R10 = SCRATCH.
                     01F5   760 ;       R11 = SCRATCH.
                     01F5   761 ;       AP = ADDRESS OF FIRST FUNCTION DEPENDENT PARAMETER.
                     01F5   762 ;
                     01F5   763 ; OUTPUTS:
                     01F5   764 ;
                     01F5   765 ;       THE CHARACTERISTICS SPECIFIED BY THE QUADWORD POINTER TO BY P1 ARE STORED
                     01F5   766 ;       IN THE SECOND AND THIRD LONGWORDS OF THE DEVICE UCB.
                     01F5   767 ;
                     01F5   768 ; COMPLETION CODES:
                     01F5   769 ;
                     01F5   770 ;       SS$_NORMAL - SUCCESSFUL
                     01F5   771 ;       SS$_ACCVIO - BUFFER ACCESS VIOLATION
                     01F5   772 ;       SS$_ILLIOFUNC - FUNCTION IS ILLEGAL ON DISK DEVICES
                     01F5   773 ;-
                     01F5   774
                     01F5   775          .ENABL  LSB
                     01F5   776 EXE$SETCHAR::                                ;SET DEVICE MODE AND CHARACTERISTICS
          1E   10    01F5   777          BSBB    CHECK_SET                   ;IS THIS SET FUNCTION VAILD?
       57 23   D1    01F7   778          CMPL    #IO$_SETMODE,R7             ;SET MODE FUNCTION?
          04   13    01FA   779          BEQL    10$                         ;IF EQL YES
    40 A5 61   B0    01FC   780          MOVW    (R1),UCB$B_DEVCLASS(R5)     ;SET DEVICE TYPE AND CLASS
42 A5   02 A1   B0    0200   781 10$:     MOVW    2(R1),UCB$W_DEVBUFSIZ(R5)  ;SET DEFAULT BUFFER SIZE
44 A5   04 A1   D0    0205   782          MOVL    4(R1),UCB$L_DEVDEPEND(R5)  ;SET DEVICE CHARACTERISTICS
          2A   11    020A   783          BRB     20$                         ;
```

```
                     020C   785          .SBTTL  SET DEVICE MODE AND CHARACTERISTICS FUNCTIONS
                     020C   786 ;+
                     020C   787 ; EXE$SETMODE - SET DEVICE CHARACTERISTICS AND MODE
                     020C   788 ;
                     020C   789 ; FUNCTIONAL DESCRIPTION:
                     020C   790 ;
                     020C   791 ; THIS ROUTINE PLACES THE NEW CHARACTERISTICS SPECIFIED BY P1 INTO
                     020C   792 ; THE I/O PACKET FOR INSERTION INTO THE UCB WHEN THE UNIT IS IDLE.
                     020C   793 ; THE INPUT DATA IS IN THE FORM RETURNED BY $GTCHAN. THE SPECIFIED BUFFER
                     020C   794 ; IS ASSUMED TO BE 12 BYTES IN LENGTH. THE P2 LENGTH SPECIFIER IS IGNORED.
                     020C   795 ;
                     020C   796 ; THE NEW CHARACTERISTICS ARE PLACED IN IRP$L_MEDIA/MEDIA+4 AND THE
                     020C   797 ; PACKET IS QUEUED VIA EXE$QIODRVPKT.
                     020C   798 ;
                     020C   799 ; INPUTS:
                     020C   800 ;
                     020C   801 ;       R3 = I/O PACKET ADDRESS
                     020C   802 ;       R4 = CURRENT PCB
                     020C   803 ;       R5 = ACB ADDRESS
                     020C   804 ;       R6 = ASSIGNED CCB ADDRESS
                     020C   805 ;       AP = ADDRESS OF THE QIO ARGUMEMT P1
                     020C   806 ;
                     020C   807 ; OUTPUTS:
                     020C   808 ;
                     020C   809 ;       R0 = STATUS OF THE OPERATION
                     020C   810 ;       R3+ ARE PRESERVED.
                     020C   811 ;
                     020C   812 ; COMPLETION CODES:
                     020C   813 ;
                     020C   814 ;       SS$_NORMAL - SUCCESSFUL
                     020C   815 ;       SS$_ACCVIO - BUFFER ACCESS VIOLATION
                     020C   816 ;       SS$_ILLIOFUNC - FUNCTION IS ILLEGAL ON DISK DEVICES
                     020C   817 ;-
                     020C   818
                     020C   819 EXE$SETMODE::                                ;SET DEVICE MODE AND CHARACTERISTICS
          07   10    020C   820          BSBB    CHECK_SET                   ;IS THIS SET FUNCTION VAILD?
 38 A3    61   7D    020E   821          MOVQ    (R1),IRP$L_MEDIA(R3)        ;INSERT CHARACTERISTICS IN I/O PACKET
       FDEB'  31    0212   822          BRW     EXE$QIODRVPKT               ;QUEUE THE PACKET
                     0215   823
                     0215   824 CHECK_SET:
 40 A5    01   91    0215   825          CMPB    #DC$_DISK, UCB$B_DEVCLASS(R5) ; Is this a disk device?
          0A   13    0219   826          BEQL    91$                         ; Branch if disk; they can't be set.
    51    6C   D0    021B   827          MOVL    P1(AP), R1                  ; Get buffer address.
                     021E   828          IFNORD  #8, (R1), 93$               ; Branch if no read access to buffer.
               05    0224   829          RSB                                 ; Else, all is ok; return to caller.
                     0225   830
 50  00F4 8F    3C   0225   831 91$:     MOVZWL  #SS$_ILLIOFUNC, R0          ; Setup illegal I/O function status.
          03   11    022A   832          BRB     99$                         ; or
    50    0C   3C    022C   833 93$:     MOVZWL  #SS$_ACCVIO, R0             ; Setup access violation status.
       FDCE'  31    022F   834 99$:     BRW     EXE$ABORTIO                 ; Then blow the I/O request away.
```

```
                     0232    836          .SBTTL  SENSE DEVICE MODE AND CHARACTERISTICS FUNCTIONS
                     0232    837 ;+
                     0232    838 ; EXE$SENSEMODE - SENSE DEVICE MODE AND CHARACTERISTICS FUNCTIONS
                     0232    839 ;
                     0232    840 ; THIS ROUTINE OBTAINS THE CURRENT DEVICE MODE/CHARACTERISTICS FROM THE DEVICE
                     0232    841 ; DEPENDENT CHARACTERISTICS LONGWORD IN THE UCB AND IMMEDIATELY COMPLETES THE
                     0232    842 ; I/O OPERATION WITH THE SECOND LONGWORD OF THE FINAL I/O STATUS EQUAL TO THE
                     0232    843 ; DEVICE DEPENDENT CHARACTERSITICS.
                     0232    844 ;
                     0232    845 ; INPUTS:
                     0232    846 ;
                     0232    847 ;       R0 = SCRATCH.
                     0232    848 ;       R1 = SCRATCH.
                     0232    849 ;       R2 = SCRATCH.
                     0232    850 ;       R3 = ADDRESS OF I/O REQUEST PACKET.
                     0232    851 ;       R4 = CURRENT PROCESS PCB ADDRESS.
                     0232    852 ;       R5 = ASSIGNED DEVICE UCB ADDRESS.
                     0232    853 ;       R6 = ADDRESS OF CCB.
                     0232    854 ;       R7 = I/O FUNCTION CODE BIT NUMBER.
                     0232    855 ;       R8 = FUNCTION DECISION TABLE DISPATCH ADDRESS.
                     0232    856 ;       R9 = SCRATCH.
                     0232    857 ;       R10 = SCRATCH.
                     0232    858 ;       R11 = SCRATCH.
                     0232    859 ;       AP = ADDRESS OF FIRST FUNCTION DEPENDENT PARAMETER.
                     0232    860 ;
                     0232    861 ; OUTPUTS:
                     0232    862 ;
                     0232    863 ;       THE DEVICE DEPENDENT CHARACTERISTICS ARE OBTAINED FROM THE UCB AND
                     0232    864 ;       THE I/O IS COMPLETED WITH THE SECOND I/O STATUS LONGWORD EQUAL TO THE
                     0232    865 ;       DEVICE CHARACTERISTICS.
                     0232    866 ;-
                     0232    867
                     0232    868 EXE$SENSEMODE::                             ;SENSE DEVICE MODE/CHARACTERISTICS
51   44 A5   D0      0232    869          MOVL    UCB$L_DEVDEPEND(R5),R1     ;GET DEVICE DEPENDENT CHARACTERSITICS
   50   01   3C      0236    870 20$:     MOVZWL  #SS$_NORMAL,R0             ;SET NORMAL COMPLETION STATUS
      FDC4'  31      0239    871          BRW     EXE$FINISHIO               ;FINISH I/O OPERATION
                     023C    872          .DSABL  LSB
```

```
                     023C   874          .SBTTL  CARRIAGE CONTROL INTERPRETATION
                     023C   875 ;+
                     023C   876 ; EXE$CARRIAGE - INTERPRET CARRIAGE CONTROL SPECIFIER
                     023C   877 ;
                     023C   878 ; FUNCTIONAL DESCRIPTION:
                     023C   879 ;
                     023C   880 ; THIS ROUTINE IS USED BY THE LINE PRINTER DRIVER AND THE TERMINAL
                     023C   881 ; DRIVER TO INTERPRET THE CARRIAGE CONTROL SPECIFIER IN IRP$B_CARCON .
                     023C   882 ; NOTE THAT IRP$B_CARCON IS USED AS A LONGWORD!
                     023C   883 ;
                     023C   884 ; THE SPECIFIER IS AS FOLLOWS:
                     023C   885 ;
                     023C   886 ;       .BYTE 1 -- FORTRAN CARRIAGE CONTROL CHARACTER IF NOT 0
                     023C   887 ;       .BYTE 2 -- ***** IGNORED *****
                     023C   888 ;       .BYTE 3 -- PREFIX CARRIAGE CONTROL
                     023C   889 ;       .BYTE 4 -- SUFFIX CARRIAGE CONTROL
                     023C   890 ;
                     023C   891 ; THE PRE/SUF FIELDS ARE AS FOLLOWS
                     023C   892 ;
                     023C   893 ;       IF BIT 7=0 THEN BITS 6-0 ARE THE NUMBER OF NEWLINES TO INSERT.
                     023C   894 ;       IF BIT 7=1 AND BIT 6=0 THEN BITS 4-0 ARE THE ASCII CHARACTER TO
                     023C   895 ;                                  OUTPUT. ASCII SET C0 OR C1 IS SPECIFIED BY BIT 5.
                     023C   896 ;       IF BIT 7=1 AND BIT 6=1 THEN BITS 5-0 ARE THE PRINTER CHANNEL NUMBER
                     023C   897 ;
                     023C   898 ; ASCII SET C0 IS ASSUMED AND BIT 6 IS IGNORED IF BIT 7=0.
                     023C   899 ;
                     023C   900 ; INPUTS:
                     023C   901 ;
                     023C   902 ;       R3 = ADDRESS OF THE I/O PACKET
                     023C   903 ;       R5 = ADDRESS OF THE UCB
                     023C   904 ;
                     023C   905 ; OUTPUTS:
                     023C   906 ;
                     023C   907 ;       IRP$B_CARCON IS SET UP TO REFLECT THE PRE/SUF CHARACTERS TO SEND.
                     023C   908 ;
                     023C   909 ;       BYTE 0 = NUMBER OF CHARACTERS TO SEND
                     023C   910 ;       BYTE 1 = CHARACTER, IF 0 THEN NEWLINE
                     023C   911 ;
                     023C   912 ;       IRP$B_CARCON+2 HAS THE SUFFIX CONTROL.
                     023C   913 ;
                     023C   914 ;       R0,R1 ARE USED.
                     023C   915 ;
                     023C   916 ;-
                     023C   917 ;
                     023C   918 ; LOCAL DATA TABLE
                     023C   919 ;
                     023C   920 CCTABLE:                                     ; CARRIAGE CONTROL TO FORTRAN MATCH TABLE
0D 01 00 01          023C   921          .BYTE   1,0,1,13                   ; SPACE => 1 NL, 1 CR
          20         0240   922          .ASCII  / /
0D 01 00 02          0241   923          .BYTE   2,0,1,13                   ; "0" => 2 NL, 1 CR
          30         0245   924          .ASCII  /0/
0D 01 0C 01          0246   925          .BYTE   1,12,1,13                  ; "1" => 1 FF, 1 CR
          31         024A   926          .ASCII  /1/
0D 01 00 00          024B   927          .BYTE   0,0,1,13                   ; "+" => NOTHING, 1 CR
          2B         024F   928          .ASCII  /+/
00 00 00 01          0250   929          .BYTE   1,0,0,0                    ; "$" => 1 NL, NOTHING
          24         0254   930          .ASCII  /$/
```

```
                OD 01 00 01   0255   931          .BYTE   1,0,1,13                ; DEFAULT => 1 NL, 1 CR
                      00      0259   932          .BYTE   0                       ; TABLE END
                              025A   933 ;
                              025A   934 ;
                              025A   935 ;
                              025A   936 EXE$CARRIAGE::                           ; INTERPRET CARRIAGE CONTROL
51    3C A3      9A           025A   937          MOVZBL  IRP$B_CARCON(R3),R1     ; GET FORTRAN SPECIFIER
                 12           025E   938          BEQL    20$                     ; IF EQL THEN TRY PRE/SUF
50    D9 AF      9E           0260   939          MOVAB   B^CCTABLE,R0            ; ADDRESS MATCH TABLE
3C A3    80      D0           0264   940 10$:     MOVL    (R0)+,IRP$B_CARCON(R3)  ; ASSUME MATCH
            60   95           0268   941          TSTB    (R0)                    ; END OF TABLE?
            05   13           026A   942          BEQL    15$                     ; IF EQL THEN YES
   51       80   91           026C   943          CMPB    (R0)+,R1                ; MATCH?
            F3   12           026F   944          BNEQ    10$                     ; NO THEN SEARCH
                 05           0271   945 15$:     RSB                             ; ELSE RETURN
                              0272   946 ;
                              0272   947 ; PRE/SUF CARRIAGE CONTROL
                              0272   948 ;
51    3E A3      9A           0272   949 20$:     MOVZBL  IRP$B_CARCON+2(R3),R1   ; GET PREFIX SPECIFIER
                 02           0276   950          BEQL    30$                     ; IF EQL THEN NONE
                 19           0278   951          BSBB    100$                    ; INTERPRET THE SPECIFIER
3C A3            51   90      027A   952 30$:     MOVB    R1,IRP$B_CARCON(R3)     ; INSERT NUMBER
3D A3            50   90      027E   953          MOVB    R0,IRP$B_CARCON+1(R3)   ; INSERT CHARACTER
51    3F A3      9A           0282   954          MOVZBL  IRP$B_CARCON+3(R3),R1   ; GET SUFFIX SPECIFIER
                 02           0286   955          BEQL    40$                     ; IF EQL THEN NONE
                 09           0288   956          BSBB    100$                    ; CONVERT THE SPECIFIER
3E A3            51   90      028A   957 40$:     MOVB    R1,IRP$B_CARCON+2(R3)   ; INSERT NUMBER
3F A3            50   90      028E   958          MOVB    R0,IRP$B_CARCON+3(R3)   ; INSERT CHARACTER
                 05           0292   959          RSB                             ; RETURN
                              0293   960 ;
                              0293   961 ; SUBROUTINE TO INTERPRET PRE/SUF SPECIFIER
                              0293   962 ;
                      50      D4     0293   963 100$:    CLRL    R0              ; ASSUME NEWLINE
         08 51                07     0295   964          BBC     #7,R1,110$      ; IF BIT 7 CLEAR THEN DONE
50       51      E0 8F        8B     0299   965          BICB3   #^XE0,R1,R0     ; REMOVE OTHER BITS
               51             01     029E   966          MOVZBL  #1,R1           ; SET ONE CHARACTER
                              05     02A1   967 110$:    RSB                     ; RETURN
                              02A2   968
                              02A2   969          .END
```

```
ACB$V_QUOTA             =  00000006
ACCVIO                     000001A0 R      01
ACCVIO1                    0000019C R      01
BACKOUT_QIO                000000FF R      01
CCB$W_IOC               =  0000000A
CCTABLE                    0000023C R      01
CHECK_SET                  00000215 R      01
DC$_DISK                =  00000001
EXE$ABORTIO                ********   X    01
EXE$CARRIAGE               0000025A RG     01
EXE$C_CMSTKSZ              ********   X    01
EXE$DEANONPAGED            ********   X    01
EXE$FINISHIO               ********   X    01
EXE$FINISHIOC              ********   X    01
EXE$IORSNWAIT              00000000 RG     01
EXE$LCLDSKVALID            00000035 RG     01
EXE$MODIFY                 00000069 RG     01
EXE$MODIFYLOCK             000000A1 RG     01
EXE$MODIFYLOCKR            000000A4 RG     01
EXE$ONEPARM                00000029 RG     01
EXE$QIODRVPKT              ********   X    01
EXE$READ                   0000006F RG     01
EXE$READCHK                00000132 RG     01
EXE$READCHKR               00000146 RG     01
EXE$READLOCK               0000009B RG     01
EXE$READLOCKR              000000AE RG     01
EXE$SENSEMODE              00000232 RG     01
EXE$SETCHAR                000001F5 RG     01
EXE$SETMODE                0000020C RG     01
EXE$WRITE                  00000078 RG     01
EXE$WRITECHK               00000138 RG     01
EXE$WRITECHKR              000001A4 RG     01
EXE$WRITELOCK              0000009E RG     01
EXE$WRITELOCKR             000000B5 RG     01
EXE$ZEROPARM               0000002F RG     01
IO$_PACKACK             =  00000008
IO$_PHYSICAL            =  0000001F
IO$_READLBLK            =  00000021
IO$_READPBLK            =  0000000C
IO$_SETMODE             =  00000023
IPL$_SCS                =  00000008
IPL$_SYNCH              =  00000008
IRP$B_CARCON            =  0000003C
IRP$B_RMOD              =  0000000B
IRP$L_BCNT              =  00000032
IRP$L_DIAGBUF           =  0000004C
IRP$L_MEDIA             =  00000038
IRP$L_SVAPTE            =  0000002C
IRP$M_FUNC              =  00000002
IRP$S_FCODE             =  00000006
IRP$V_BUFIO             =  00000000
IRP$V_DIAGBUF           =  00000007
IRP$V_FCODE             =  00000000
IRP$V_FUNC              =  00000001
IRP$W_BOFF              =  00000030
IRP$W_FUNC              =  00000020
IRP$W_STS               =  0000002A
MMG$IOLOCK                 ********   X    01
P1                      =  00000000
P2                      =  00000004
P3                      =  00000008
P4                      =  0000000C
P5                      =  00000010
P6                      =  00000014
PCB$L_EFWM              =  0000004C
PCB$L_STS               =  00000024
PCB$V_SSRWAIT           =  0000000A
PCB$W_ASTCNT            =  00000038
PCB$W_BIOCNT            =  0000003A
PCB$W_DIOCNT            =  0000003E
PMS$ABORT_RQ               ********   X    01
PR$_IPL                 =  00000012
SCH$GL_RESMASK             ********   X    01
SCH$GQ_MWAIT               ********   X    01
SCH$WAIT                   ********   X    01
SF$L_SAVE_AP            =  00000008
SS$_ACCVIO              =  0000000C
SS$_ILLIOFUNC           =  000000F4
SS$_NORMAL              =  00000001
UCB$B_DEVCLASS          =  00000040
UCB$B_ONLCNT            =  000000AE
UCB$L_DEVDEPEND         =  00000044
UCB$L_STS               =  00000064
UCB$V_LCL_VALID         =  00000011
UCB$V_VALID             =  0000000B
UCB$W_DEVBUFSIZ         =  00000042
VA$M_BYTE               =  000001FF
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
. BLANK .	000002A2 (674.)	01 (1.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
ABS	00000000 (0.)	02 (2.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.08	00:00:00.37
Command processing	105	00:00:00.55	00:00:02.66
Pass 1	449	00:00:16.67	00:00:35.37
Symbol table sort	0	00:00:02.88	00:00:05.37
Pass 2	179	00:00:03.67	00:00:07.65
Symbol table output	12	00:00:00.11	00:00:00.11
Psect synopsis output	1	00:00:00.03	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	777	00:00:23.99	00:00:51.56

The working set limit was 1800 pages.
99830 bytes (195 pages) of virtual memory were used to buffer the intermediate code.
There were 100 pages of symbol table space allocated to hold 1877 non-local and 44 local symbols.
969 source lines were read in Pass 1, producing 15 object records in Pass 2.
23 pages of virtual memory were used to define 22 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYS.OBJ]LIB.MLB;1	10
_255DUA28:[SYSLIB]STARLET.MLB;2	9
TOTALS (all libraries)	19

1959 GETS were required to define 19 macros.

There were no errors, warnings or information messages.

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

0387 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

SYSPURGWS
LIS

SYSPUTMSG
LIS

SYSPCNTRL
LIS

SYSQIOFDT
LIS

SYSQIOREQ
LIS

SYSRUNDWN
LIS

SYSRDBRES
LIS

SYSRTSLST
LIS