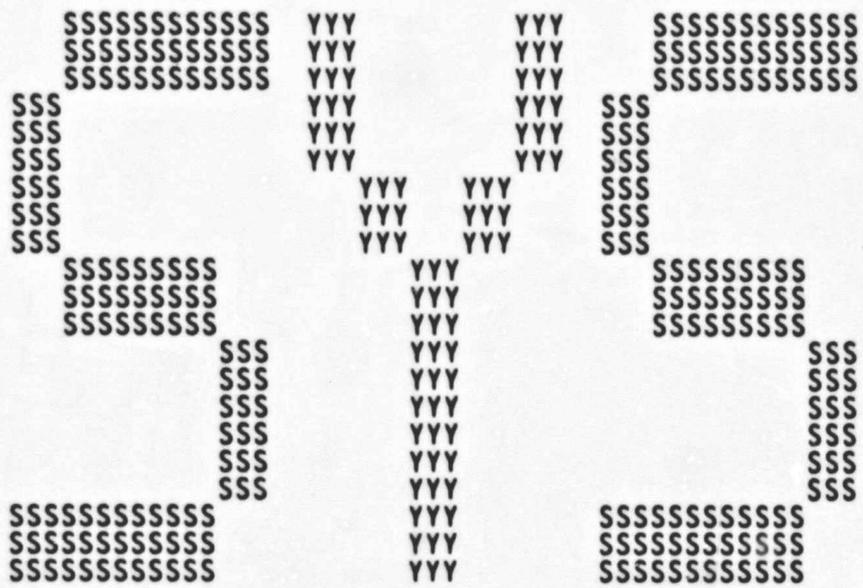
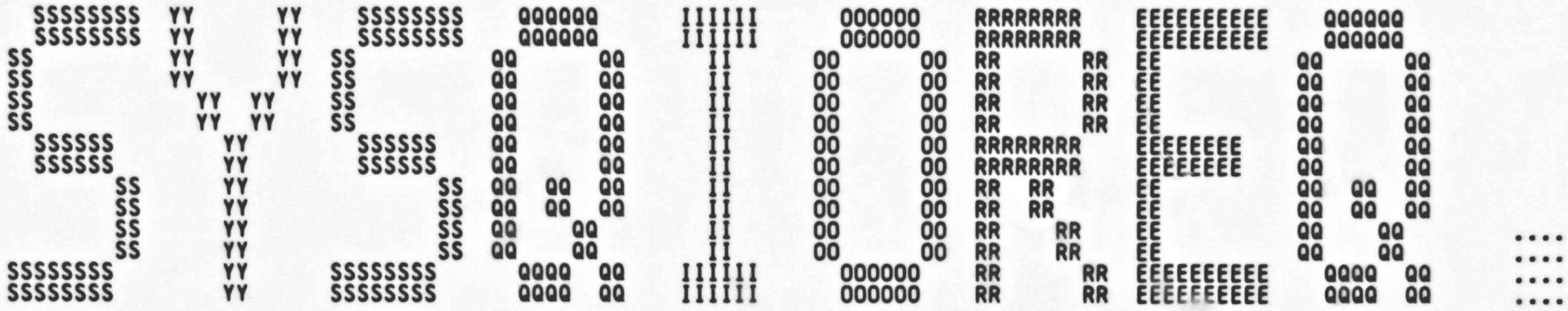

SSSSSSSSSSS YYY YYY SSSSSSSSSSS
SSSSSSSSSSS YYY YYY SSSSSSSSSSS
SSSSSSSSSSS YYY YYY SSSSSSSSSSS
SSS YYY YYY SSS
SSS YYY YYY SSS
SSS YYY YYY SSS
SSS YYY YYY SSS
SSS YYY YYY SSS
SSS YYY YYY SSS
SSSSSSSSS YYY SSSSSSSSS
SSSSSSSSS YYY SSSSSSSSS
SSSSSSSSS YYY SSSSSSSSS
SSS YYY SSS
SSS YYY SSS
SSS YYY SSS
SSS YYY SSS
SSS YYY SSS
SSS YYY SSS
SSSSSSSSSSS YYY SSSSSSSSSSS
SSSSSSSSSSS YYY SSSSSSSSSSS
SSSSSSSSSSS YYY SSSSSSSSSSS

FILEID**SYSQIOREQ



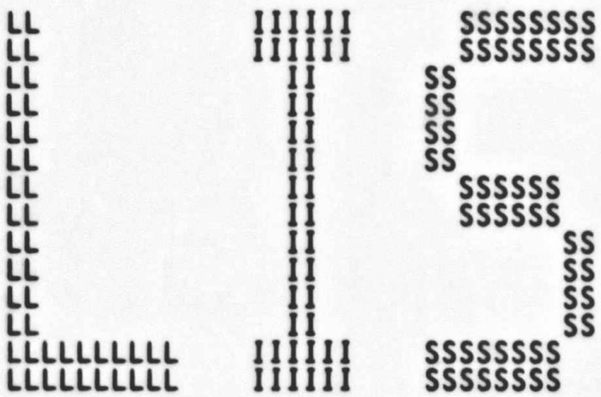


Table of contents

(1) 94 DECLARATIONS
(1) 183 QIO ERROR AND EXCEPTION HANDLING ROUTINES
(1) 236 QUEUE I/O REQUEST
(1) 656 BUILD I/O PACKET FOR PAGE READ/WRITE
(1) 790 COMPLETE I/O OPERATION
(1) 841 QUEUE I/O PACKET TO DRIVER
(1) 864 EXE$ALTQUEPKT - Call driver ALTSTART entry point
(1) 899 QUEUE I/O PACKET TO ACP
(1) 946 EXE$QXQPPKT - QUEUE I/O PACKET TO XQP
(1) 988 INSERT I/O PACKET IN UNIT QUEUE
(1) 1012 INSERT I/O PACKET IN QUEUE BY PRIORITY

```
0000     1          .TITLE  SYSQIOREQ - QUEUE I/O REQUEST SYSTEM SERVICE
0000     2          .IDENT  'V04-001'
0000     3
0000     4  ;
0000     5  ;*****************************************************************************
0000     6  ;*                                                                           *
0000     7  ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     8  ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     9  ;*  ALL RIGHTS RESERVED.                                                     *
0000    10  ;*                                                                           *
0000    11  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
0000    12  ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
0000    13  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
0000    14  ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
0000    15  ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
0000    16  ;*  TRANSFERRED.                                                             *
0000    17  ;*                                                                           *
0000    18  ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
0000    19  ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
0000    20  ;*  CORPORATION.                                                             *
0000    21  ;*                                                                           *
0000    22  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
0000    23  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    24  ;*                                                                           *
0000    25  ;*                                                                           *
0000    26  ;*****************************************************************************
0000    27  ;
0000    28  ;++
0000    29  ;
0000    30  ; AUTHOR:
0000    31  ;
0000    32  ;       D. N. CUTLER,   14-JUN-76
0000    33  ;
0000    34  ; FACILITY:
0000    35  ;
0000    36  ;       SYSTEM SERVICE QUEUE I/O REQUEST
0000    37  ;
0000    38  ; MODIFIED BY:
0000    39  ;
0000    40  ;       V04-001 ACG0467         Andrew C. Goldstein,    12-Sep-1984  17:33
0000    41  ;               Fix protection holes in QIO device protection check
0000    42  ;
0000    43  ;       V03-012 LMP0221         L. Mark Pilant,         30-Mar-1984  17:08
0000    44  ;               Remove references to UCB$L_OWNUIC and UCNB$W_VPROT.
0000    45  ;
0000    46  ;       V03-011 SRB0118         Steve Beckhardt         23-Mar-1984
0000    47  ;               Changed waiting for DIOCNT or BIOCNT to wait at IPL 0.
0000    48  ;
0000    49  ;       V03-010 CDS0005         Christian D. Saether    20-Mar-1984
0000    50  ;               Use symbolic definition to locate XQP queue header.
0000    51  ;               Give the XQP ast a priority boost.
0000    52  ;
0000    53  ;       V03-009 SSA0017         Stan Amway              9-Mar-1984
0000    54  ;               Maintain device queue length in UCB.
0000    55  ;               Efficiently supports MONITORs disk class and provides
0000    56  ;               accurate queue lengths for HSC and UDA disks.
0000    57  ;
```

```
0000     58 ;       V03-008 LMP0206         L. Mark Pilant,         7-Mar-1984  12:20
0000     59 ;               Only do an access check on the first QIO to the channel for
0000     60 ;               logical and physical requests.
0000     61 ;
0000     62 ;       V03-007 LMP0185         L. Mark Pilant,         27-Jan-1984  11:05
0000     63 ;               Add support for ACLs on devices.
0000     64 ;
0000     65 ;       V03-006 CDS0004         Christian D. Saether    20-May-1983
0000     66 ;               Get the PID from the PCB instead of the IRP so that
0000     67 ;               journalling works (it uses the irp pid field for
0000     68 ;               something else).
0000     69 ;
0000     70 ;       V03-005 RLRMXBCNTc      Robert L. Rappaport     28-Mar-1983
0000     71 ;               Verify IRP$L_DIAGBUF is non-zero before assuming that it
0000     72 ;               contains the original value of IRP$L_SVAPTE in VIRTUAL_LOGIO.
0000     73 ;
0000     74 ;       V03-004 CDS0003         Christian D. Saether    14-Mar-1983
0000     75 ;               Return from EXE$QXQPPKT with status, rather than
0000     76 ;               assuming success.
0000     77 ;
0000     78 ;       V03-003 CDS0002         Christian D. Saether    12-Mar-1983
0000     79 ;               Do not insque packet to xqp work queue in EXE$QXQPPKT,
0000     80 ;               but rather pass it as the ast parameter and queue it
0000     81 ;               in the xqp, if necessary.  This avoids a problem
0000     82 ;               where the packet is processed by the xqp before the
0000     83 ;               ast is dequeued.
0000     84 ;
0000     85 ;       V03-002 CDS0001         C Saether               13-Aug-1982
0000     86 ;               Changes to send file system packets to XQP.
0000     87 ;
0000     88 ;       V03-001 LJK0172         Lawrence J. Kenah       18-June-1982
0000     89 ;               Count I/O operations in EXE$BLDPKTxxxx to allow file
0000     90 ;               expiration to work correctly for mapped files.
0000     91 ;
0000     92 ;--
```

```
                     0000     94          .SBTTL  DECLARATIONS
                     0000     95
                     0000     96 ;
                     0000     97 ; MACRO LIBRARY CALLS
                     0000     98 ;
                     0000     99
                     0000    100          $ACBDEF                         ;DEFINE ACB OFFSETS
                     0000    101          $AQBDEF                         ;DEFINE AQB OFFSETS
                     0000    102          $CADEF                          ;DEFINE CONDITIONAL ASSEMBLY PARAMETERS
                     0000    103          $CCBDEF                         ;DEFINE CCB OFFSETS
                     0000    104          $CDRPDEF                        ;DEFINE CDRP OFFSETS
                     0000    105          $DDBDEF                         ;DEFINE DDB OFFSETS
                     0000    106          $DDTDEF                         ;DEFINE DDT OFFSETS
                     0000    107          $DEVDEF                         ;DEFINE DEV VALUES
                     0000    108          $DYNDEF                         ;DEFINE DATA STRUCTURE TYPE CODES
                     0000    109          $F11BDEF                        ;DEFINE F11BXQP OFFSETS
                     0000    110          $IODEF                          ;DEFINE IO FUNCTION CODES
                     0000    111          $IPLDEF                         ;DEFINE INTERRUPT PRIORITY LEVELS
                     0000    112          $IRPDEF                         ;DEFINE IRP OFFSETS
                     0000    113          $PCBDEF                         ;DEFINE PCB OFFSETS
                     0000    114          $PHDDEF                         ;DEFINE PHD OFFSETS
                     0000    115          $PRDEF                          ;DEFINE PROCESSOR REGISTERS
                     0000    116          $PRIDEF                         ;DEFINE PRIORITY CLASS INCREMENTS
                     0000    117          $PRVDEF                         ;DEFINE PRIVILEGE BITS
                     0000    118          $PSLDEF                         ;DEFINE PROCESSOR STATUS FIELDS
                     0000    119          $RSNDEF                         ;DEFINE RESOURCE WAIT NUMBERS
                     0000    120          $SECDEF                         ;DEFINE SEC OFFSETS
                     0000    121          $SSDEF                          ;DEFINE STATUS VALUES
                     0000    122          $UCBDEF                         ;DEFINE UCB OFFSETS
                     0000    123          $VCBDEF                         ;DEFINE VCB OFFSETS
                     0000    124          $WCBDEF                         ;DEFINE WINDOW CONTROL BLOCK OFFSETS
                     0000    125
                     0000    126 ;
                     0000    127 ; LOCAL SYMBOLS
                     0000    128 ;
                     0000    129 ; ARGUMENT LIST OFFSET DEFINITIONS
                     0000    130 ;
                     0000    131
00000004             0000    132 EFN=4                                   ;EVENT FLAG NUMBER
00000008             0000    133 CHAN=8                                  ;I/O CHANNEL NUMBER
0000000C             0000    134 FUNC=12                                 ;I/O FUNCTION CODE
00000010             0000    135 IOSB=16                                 ;ADDRESS OF I/O STATUS BLOCK
00000014             0000    136 ASTADR=20                               ;ADDRESS OF AST SERVICE ROUTINE
00000018             0000    137 ASTPRM=24                               ;AST SERVICE ROUTINE PARAMETER
0000001C             0000    138 P1=28                                   ;FIRST FUNCTION DEPENDENT PARAMETER
00000020             0000    139 P2=32                                   ;SECOND FUNCTION DEPENDENT PARAMETER
00000024             0000    140 P3=36                                   ;THIRD FUNCTION DEPENDENT PARAMETER
00000028             0000    141 P4=40                                   ;FOURTH FUNCTION DEPENDENT PARAMETER
0000002C             0000    142 P5=44                                   ;FIFTH FUNCTION DEPENDENT PARAMETER
00000030             0000    143 P6=48                                   ;SIXTH FUNCTION DEPENDENT PARAMETER
                     0000    144
                     0000    145 ;
                     0000    146 ; FUNCTION DECISION TABLE OFFSET DEFINITIONS
                     0000    147 ;
                     0000    148
00000000             0000    149 LEGAL=0                                 ;LEGAL FUNCTION MASK
00000008             0000    150 IOTYPE=8                                ;I/O FUNCTION TYPE MASK
```

```
00000010  0000   151 FDTACT=16                                ;ACTION ROUTINE MASKS
          0000   152
          0000   153 ;
          0000   154 ; TABLES FOR DETERMINING THE ACCESS DESIRED, BASED UPON THE I/O FUNCTION
          0000   155 ; CODE.  THIS IS NECESSARY FOR THE FIRST TIME THROUGH PROTECTION CHECK DONE
          0000   156 ; ON SHARABLE, NON-MOUNTABLE (NON-FILES ORIENTED) DEVICES.
          0000   157 ;
          0000   158
          0000   159          .MACRO  ACCMSK  CODES
          0000   160 MASKL    = 0
          0000   161 MASKH    = 0
          0000   162
          0000   163          .IRP    X,<CODES>
          0000   164          .IF     GT      <IO$_'X&IO$_VIRTUAL>-31
          0000   165 MASKH    = MASKH!<1@<<IO$_'X&IO$_VIRTUAL>-32>>
          0000   166          .IFF
          0000   167 MASKL    = MASKL!<1@<IO$_'X&IO$_VIRTUAL>>
          0000   168          .ENDC;  GT      <IO$_'X&IO$_VIRTUAL>-31
          0000   169          .ENDR ; X,<CODES>
          0000   170          .LONG   MASKL,MASKH
          0000   171          .ENDM   ACCMSK
          0000   172
          0000   173 READ_ACCESS:
          0000   174          ACCMSK  <READPBLK,READLBLK,READVBLK,-
          0000   175           READHEAD,READTRACKD,REREADN,REREADP,-
          0000   176           READPROMPT,TTYREADALL,TTYREADPALL>
          0008   177
          0008   178 WRITE_ACCESS:
          0008   179          ACCMSK  <WRITEPBLK,WRITELBLK,WRITEVBLK,-
          0008   180           WRITECHECK,WRITECHECKH,-
          0008   181           WRITEHEAD,WRITETRACKD,WRITERET>
```

```
                     0010   183          .SBTTL  QIO ERROR AND EXCEPTION HANDLING ROUTINES
                     0010   184 ;
                     0010   185 ; Device is not mountable.  See if it is necessary to do the protection check.
                     0010   186 ;
                     0010   187 NOT_FILE_DEV:
28 38 A5    10   E1    0010   188          BBC     #DEV$V_SHR,UCB$L_DEVCHAR(R5),20$        ;XFER IF NOT SHARABLE
0F E7 AF    57   E1    0015   189          BBC     R7,READ_ACCESS,10$      ;XFER IF NOT A READ FUNCTION
0A 08 A6    02   E0    001A   190          BBS     #CCB$V_RDCHKDON,CCB$B_STS(R6),10$       ;XFER IF HERE ALREADY
          FFDE'  30    001F   191          BSBW    EXE$CHKRDACCES          ;DO PROTECTION CHECK
        2A 50    E9    0022   192          BLBC    R0,ERRORB               ;XFER IF NOT SUCCESSFUL
   08 A6    04   88    0025   193          BISB    #CCB$M_RDCHKDON,CCB$B_STS(R6)           ;NOTE PROT CHECK MADE
0F DB AF    57   E1    0029   194 10$:     BBC     R7,WRITE_ACCESS,20$     ;XFER IF NOT A WRITE FUNCTION
0A 08 A6    03   E0    002E   195          BBS     #CCB$V_WRTCHKDON,CCB$B_STS(R6),20$      ;XFER IF HERE ALREADY
          FFCA'  30    0033   196          BSBW    EXE$CHKWRTACCES         ;DO PROTECTION CHECK
        16 50    E9    0036   197          BLBC    R0,ERRORB               ;XFER IF NOT SUCCESSFUL
   08 A6    08   88    0039   198          BISB    #CCB$M_WRTCHKDON,CCB$B_STS(R6)          ;NOTE PROT CHECK MADE
          0092   31    003D   199 20$:     BRW     CHKDON                  ;ELSE REJOIN MAINLINE CODE
                     0040   200 ;
                     0040   201 ; MISCELLANEOUS ERROR HANDLING AND EXCEPTION HANDLING ROUTINES.  THESE HAVE
                     0040   202 ; BEEN MOVED OUT OF LINE TO MAKE THE COMMON PATH NEARLY BRANCH FREE.
                     0040   203 ;
                     0040   204
                     0040   205 CLREF:                                          ;
          FFBD'  30    0040   206          BSBW    SCH$CLREF               ;CLEAR SPECIFIED EVENT FLAG
            40   11    0043   207          BRB     VCHAN                   ;CONTINUE WITH QIO
50   013C 8F    3C    0045   208 IVCHAN:  MOVZWL  #SS$_IVCHAN,R0           ;SET ERROR STATUS
            03   11    004A   209          BRB     ERRORB                  ;AND ERROR REQUEST
50   24   3C    004C   210 PRIVERR: MOVZWL #SS$_NOPRIV,R0                  ;SET ERROR STATUS
          00DF   31    004F   211 ERRORB: BRW     ERROR                   ;AND ERROR REQUEST
                     0052   212
                     0052   213 ;
                     0052   214 ; An access or deaccess operation is pending for this channel.  Wait for
                     0052   215 ; it to complete, then retry the QIO.
                     0052   216 ;
                     0052   217
                     0052   218 DACSPND:SETIPL  #IPL$_SYNCH                  ;SYNCHRONIZE ACCESS TO SYSTEM DATA BASE
      4C A4    01   3C    0055   219          MOVZWL  #RSN$_ASTWAIT,PCB$L_EFWM(R4) ;SET AST WAIT RESOURCE NUMBER
   52    0000'CF   7E    0059   220          MOVAQ   W^SCH$GQ_MWAIT,R2       ;SET ADDRESS OF WAIT QUEUE
00 0000'CF   4C A4   E2    005E   221          BBSS    PCB$L_EFWM(R4),W^SCH$GL_RESMASK,10$ ;SET WAITING FLAG
          FF98'  31    0065   222 10$:     BRW     SCH$WAIT                ;WAIT FOR AST
                     0068   223 ;
                     0068   224 ; Device is marked spooled.  Acquire intermediate UCB address if virtual funtion.
                     0068   225 ;
      57   2F    D1    0068   226 SPOOL:   CMPL    S^#IO$_LOGICAL,R7       ;VIRTUAL I/O FUNCTION?
            60   18    006B   227          BGEQ    NSPOOL                  ;IF GEQ NO
   55   60 A5    D0    006D   228          MOVL    UCB$L_AMB(R5),R5        ;GET INTERMEDIATE DEVICE UCB ADDRESS
            5A   11    0071   229          BRB     NSPOOL                  ;
                     0073   230 ;
                     0073   231 ; Intermediate branch to the protection checking routine.
                     0073   232 ;
                     0073   233 NOT_FILE_DEVB:
            9B   11    0073   234          BRB     NOT_FILE_DEV
```

```
0075   236          .SBTTL  QUEUE I/O REQUEST
0075   237 ;+
0075   238 ; EXE$QIOREQ - QUEUE I/O REQUEST
0075   239 ;
0075   240 ; THIS SERVICE PROVIDES THE CAPABILITY TO INITIATE AN I/O OPERATION
0075   241 ; BY QUEUEING A REQUEST TO A DEVICE'S ASSOCIATED DRIVER.  ONCE THE
0075   242 ; OPERATION HAS BEEN INITIATED, CONTROL WILL RETURN TO THE CALLER
0075   243 ; WHO CAN SYNCHRONIZE I/O COMPLETION IN ONE OF THREE WAYS:
0075   244 ;
0075   245 ;       1) SPECIFY THE ADDRESS OF AN AST ROUTINE THAT WILL BE
0075   246 ;          EXECUTED WHEN THE I/O COMPLETES.
0075   247 ;
0075   248 ;       2) WAIT FOR THE SPECIFIED EVENT FLAG TO BE SET.
0075   249 ;
0075   250 ;       3) POLL THE SPECIFIED I/O STATUS BLOCK FOR A COMPLETION
0075   251 ;          STATUS.
0075   252 ;
0075   253 ; THIS ROUTINE VERIFIES THE FUNCTION INDEPENDENT PARAMETERS, ALLOCATES
0075   254 ; AN I/O REQUEST PACKET, COPIES THE FUNCTION INDEPENDENT PARAMETERS AND
0075   255 ; PROCESS INFORMATION TO THE I/O PACKET, CHECKS ACCESS TO THE DEVICE,
0075   256 ; AND CALLS THE DRIVER'S FUNCTION DECISION TABLE ROUTINE(S) THAT CORRESPOND
0075   257 ; TO THE SPECIFIED FUNCTION.  IT IS THEN UP TO THE FDT ROUTINE TO EITHER
0075   258 ; COMPLETE THE REQUEST IMMEDIATELY (EXE$ABORTIO OR EXE$FINISHIO) OR TO
0075   259 ; QUEUE THE I/O REQUEST FOR FURTHER PROCESSING BY THE DRIVER'S STARTIO
0075   260 ; ROUTINE (EXE$QIODRVPKT).
0075   261 ;
0075   262 ; INPUTS:
0075   263 ;
0075   264 ;       EFN(AP) = EVENT FLAG NUMBER.
0075   265 ;       CHAN(AP) = I/O CHANNEL NUMBER.
0075   266 ;       FUNC(AP) = I/O FUNCTION CODE.
0075   267 ;       IOSB(AP) = ADDRESS OF I/O STATUS BLOCK.
0075   268 ;       ASTADR(AP) = ADDRESS OF AST SERVICE ROUTINE.
0075   269 ;       ASTPRM(AP) = AST SERVICE ROUTINE PARAMETER.
0075   270 ;       P1(AP) TO P6(AP) = FUNCTION DEPENDENT PARAMETERS.
0075   271 ;
0075   272 ;       R4 = CURRENT PROCESS PCB ADDRESS.
0075   273 ;
0075   274 ; OUTPUTS:
0075   275 ;
0075   276 ;       R0 LOW BIT CLEAR INDICATES FAILURE TO INITIATE THE I/O REQUEST.
0075   277 ;
0075   278 ;               R0 = SS$_ABORT - A NETWORK LOGICAL LINK WAS BROKEN.
0075   279 ;
0075   280 ;               R0 = SS$_ACCVIO - THE I/O STATUS BLOCK CANNOT BE WRITTEN BY
0075   281 ;                    THE CALLER.
0075   282 ;
0075   283 ;               R0 = SS$_DEVOFFLINE - THE SPECIFIED DEVICE IS OFFLINE.
0075   284 ;
0075   285 ;               R0 = SS$_EXQUOTA - THE PROCESS HAS EXCEEDED ITS BUFFERED I/O
0075   286 ;                    QUOTA, DIRECT I/O QUOTA, OR BUFFERED I/O BYTE COUNT
0075   287 ;                    QUOTA AND HAS DISABLED RESOURCE WAIT MODE.  OR, THE
0075   288 ;                    PROCESS HAS EXCEEDED ITS AST LIMIT QUOTA.
0075   289 ;
0075   290 ;               R0 = SS$_ILLEFC - AN ILLEGAL EVENT FLAG NUMBER WAS SPECIFIED.
0075   291 ;
0075   292 ;               R0 = SS$_INSFMEM - INSUFFICIENT DYNAMIC MEMORY IS AVAILABLE
```

```
                                0075   293 ;                       TO ALLOCATE AN I/O REQUEST PACKET AND THE PROCESS HAS
                                0075   294 ;                       DISABLED RESOURCE WAIT MODE.
                                0075   295 ;
                                0075   296 ;               R0 = SS$_IVCHAN - AN INVALID CHANNEL NUMBER WAS SPECIFIED.
                                0075   297 ;
                                0075   298 ;               R0 = SS$_NOPRIV - THE SPECIFIED CHANNEL DOES NOT EXIST OR WAS
                                0075   299 ;                    ASSIGNED FROM A MORE PRIVILEGED ACCESS MODE.  OR, THE
                                0075   300 ;                    PROCESS DOES NOT HAVE THE PRIVILEGE TO PERFORM THE
                                0075   301 ;                    SPECIFIED TYPE OF I/O FUNCTION ON THE DEVICE.
                                0075   302 ;
                                0075   303 ;               R0 = SS$_UNASEFC - UNASSOCIATED EVENT FLAG CLUSTER SPECIFIED.
                                0075   304 ;
                                0075   305 ;          R0 LOW BIT SET INDICATES SUCCESSFUL COMPLETION.
                                0075   306 ;
                                0075   307 ;               R0 = SS$_NORMAL - NORMAL COMPLETION.
                                0075   308 :-
                                0075   309
                        0FFC    0075   310          .ENTRY  EXE$QIO,^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                                0077   311 ;
                                0077   312 ; Clear specified event flag.  For local event flags, this is done in line.
                                0077   313 ;
                                0077   314
          53    04 AC    9A     0077   315 QIO:     MOVZBL  EFN(AP),R3              ;GET EVENT FLAG NUMBER
              3F    53    91     007B   316          CMPB    R3,#63                  ;CHECK FOR LOCAL
                    C0    1A     007E   317          BGTRU   CLREF                   ;IF NO, MUST DO FULL CLREF
       00 50 A4     53    E5     0080   318          BBCC    R3,PCB$L_EFCS(R4),VCHAN ;CLEAR SPECIFIED EVENT FLAG
                                0085   319
                                0085   320 ;
                                0085   321 ; Validate channel number, compute CCB address and acquire UCB address.
                                0085   322 ;
                                0085   323
            FFFF000F  8F        CB     0085   324 VCHAN:  BICL3   #<^XFFFF0000!<CCB$C_LENGTH-1>>,-;FETCH CHANNEL NUMBER AND
             50    08 AC                0088   325          CHAN(AP),R0             ;CLEAR EXTRANEOUS BITS
                       B5    13        008B   326          BEQL    IVCHAN          ;IF EQL INVALID CHANNEL
          00000000'9F    50    B1     008D   327          CMPW    R0,@#CTL$GW_CHINDX ;LEGAL CHANNEL NUMBER?
                       AC    1A        0094   328          BGTRU   IVCHAN          ;IF GTRU NO
                 59    50    CE     0096   329          MNEGL   R0,R9           ;CONVERT TO CHANNEL INDEX
      56   00000000'FF49    9E     009A   330          MOVAB   @CTL$GL_CCBBASE[R9],R6 ;GET ADDRESS OF CORRESPONDING CCB
                       53    DC     00A1   331          MOVPSL  R3              ;READ CURRENT PSL
       5B    53    02    16     EF     00A3   332          EXTZV   #PSL$V_PRVMOD,#PSL$S_PRVMOD,R3,R11 ;EXTRACT PREVIOUS MODE FIELD
             59    59    10     78     00A8   333          ASHL    #16,R9,R9       ;PREPARE CHANNEL INDEX FOR LATER MERGE
               09 A6    5B     91     00AC   334          CMPB    R11,CCB$B_AMOD(R6) ;CALLER HAVE PRIVILEGE TO ACCESS CHANNEL?
                       97    18     00B0   335          BGEQ    PRIVERR         ;IF GEQ NO
                    55    66    D0     00B2   336          MOVL    CCB$L_UCB(R6),R5 ;GET ASSIGNED DEVICE UCB ADDRESS
                  96 04 A6    E8     00B5   337          BLBS    CCB$L_WIND(R6),DACSPND ;IF LBS ACCESS/DEACCESS PENDING
                                00B9   338
                                00BC   339 ;
                                00BC   340 ; Isolate function code and begin decoding
                                00BC   341 ;
                                00BC   342
             5A    0C AC    3C     00BC   343          MOVZWL  FUNC(AP),R10            ;GET I/O FUNCTION CODE AND MODIFIERS
      57     5A    FFFFFFC0 8F     CB     00C0   344          BICL3   #^C<IO$M_FCODE>,R10,R7  ;CLEAR ALL BUT I/O FUNCTION CODE
               9B 38 A5    06     E0     00C8   345          BBS     S^#DEV$V_SPL,UCB$L_DEVCHAR(R5),SPOOL ;IF SET, DEVICE IS SPOOLED
                                00CD   346
                                00CD   347 ;
                                00CD   348 ; Acquire FDT address.
                                00CD   349 ;
```

```
                     00CD    350
A1 38 A5    0E   E1  00CD    351 NSPOOL: BBC     #DEV$V_FOD,UCB$L_DEVCHAR(R5),NOT_FILE_DEVB      ;XFER IF NOT MOUNTAB
 50    0088 C5   D0  00D2    352 CHKDON: MOVL    UCB$L_DDT(R5),R0            ;GET ADDRESS OF DDT
    58    08 A0   D0  00D7    353         MOVL    DDT$L_FDT(R0),R8            ;GET ADDRESS OF FDT
    37 68    57   E1  00DB    354         BBC     R7,LEGAL(R8),ILLIO          ;IF CLR, ILLEGAL I/O FUNCTION
39 64 A5    04   E1  00DF    355         BBC     #UCB$V_ONLINE,UCB$W_STS(R5),OFFLINE ;IF CLR, DEVICE OFFLINE
                     00E4    356
                     00E4    357 ;
                     00E4    358 ; Probe and clear IOSB if it is specified.
                     00E4    359 ;
                     00E4    360
    51    10 AC   D0  00E4    361 PRIOSB: MOVL    IOSB(AP),R1                 ;GET ADDRESS OF I/O STATUS BLOCK
              08   13  00E8    362         BEQL    NOIOSB                      ;IF EQL NONE SPECIFIED
                     00EA    363         IFNOWRT #8,(R1),ACCVIO              ;CAN I/O STATUS BLOCK BE WRITTEN?
              61   7C  00F0    364         CLRQ    (R1)                        ;CLEAR I/O STATUS BLOCK
                     00F2    365
                     00F2    366 ;
                     00F2    367 ; Charge appropriate I/O counts depending upon type.  Counts will have to
                     00F2    368 ; be backed out if no I/O packet is available.  Set IPL to block process
                     00F2    369 ; deletion once we are committed.
                     00F2    370 ;
                     00F2    371
                     00F2    372 NOIOSB: SETIPL  #IPL$_ASTDEL                ;PREVENT PROCESS DELETION
7B 08 A8    57   E1  00F5    373         BBC     R7,IOTYPE(R8),DIRECT        ;IF CLR, DIRECT I/O FUNCTION
                     00FA    374         ASSUME  IRP$M_BUFIO EQ 1            ;TO ALLOW INCREMENT BELOW
              59   B6  00FA    375         INCW    R9                          ;SET IRP$M_BUFIO
           3A A4   B7  00FC    376         DECW    PCB$W_BIOCNT(R4)            ;CHARGE FOR ANOTHER BUFFERED I/O
              79   18  00FF    377         BGEQ    OK                          ;OK IF NOT NEGATIVE
           3A A4   3F  0101    378         PUSHAW  PCB$W_BIOCNT(R4)            ;SET ADDRESS OF QUOTA CELL
        52    6E   D0  0104    379 NOCNT:  MOVL    (SP),R2                     ;FETCH QUOTA ADDRESS
              62   B6  0107    380         INCW    (R2)                        ;BACKOUT CHARGE
                   0109    381         SETIPL  #0                          ;LOWER IPL TO WAIT AT IPL 0
         FEF1'     30  010C    382         BSBW    EXE$SNGLEQUOTA              ;CHECK UNIT QUOTA OF I/O FUNCTION TYPE
      1F 50        E9  010F    383         BLBC    R0,ERROR                    ;IF LBC QUOTA EXCEEDED
              9E   B7  0112    384         DECW    @(SP)+                      ;CHARGE FOR I/O OF TYPE
              64   11  0114    385         BRB     OK                          ;
                   0116    386
50    00F4 8F   3C  0116    387 ILLIO:  MOVZWL  #SS$_ILLIOFUNC,R0           ;SET ILLEGAL I/O FUNCTION STATUS
              14   11  011B    388         BRB     ERROR                       ;
                   011D    389
    57    34   91  011D    390 OFFLINE:CMPB    #IO$_DEACCESS,R7            ;CHECK FOR DEACCESS I/O FUNCTION
              C2   13  0120    391         BEQL    PRIOSB                      ;ALLOW IT TO PROCEED
    57    38   91  0122    392         CMPB    #IO$_ACPCONTROL,R7          ;LIKEWISE FOR ACP CONTROL
              BD   13  0125    393         BEQL    PRIOSB                      ;SO THAT A FILE ON AN OFFLINE DEVICE
                   0127    394                                             ;MAY BE CLOSED
50    0084 8F   3C  0127    395         MOVZWL  #SS$_DEVOFFLINE,R0          ;SET DEVICE OFFLINE STATUS
              03   11  012C    396         BRB     ERROR                       ;
                   012E    397
    50    0C   3C  012E    398 ACCVIO: MOVZWL  S^#SS$_ACCVIO,R0            ;SET ACCESS VIOLATION STATUS
                   0131    399 ERROR:  SETIPL  #0                          ;ALLOW INTERRUPTS
              50   DD  0134    400         PUSHL   R0                          ;SAVE FINAL STATUS
 51    60 A4   D0  0136    401         MOVL    PCB$L_PID(R4),R1            ;GET PROCESS ID OF CURRENT PROCESS
              52   D4  013A    402         CLRL    R2                          ;SET PRIORITY CLASS INCREMENT
 53    04 AC   9A  013C    403         MOVZBL  EFN(AP),R3                  ;GET SPECIFIED EVENT FLAG NUMBER
         FEBD'     30  0140    404         BSBW    SCH$POSTEF                  ;POST SPECIFIED EVENT FLAG
              01   BA  0143    405         POPR    #^M<R0>                     ;RESTORE FINAL STATUS
                   04  0145    406         RET                                 ;
```

```
                     0146   407
                     0146   408 ;
                     0146   409 ; ALLOCATE REQUEST I/O PACKET - WHEN THE LOOKASIDE LIST IS EMPTY.
                     0146   410 ;
                     0146   411
          FEB7'  30  0146   412 ALLOC:  BSBW    EXE$ALLOCIRP            ;ALLOCATE I/O REQUEST PACKET
        35 50    E8  0149   413         BLBS    R0,SUCCES               ;IF LBS SUCCESSFUL ALLOCATION
        3A A4    3F  014C   414         PUSHAW  PCB$W_BIOCNT(R4)        ;ASSUME BUFFERED I/O
        03 59    E8  014F   415         BLBS    R9,NALLOC               ;IF SET, BUFFERED I/O
        3E A4    3F  0152   416         PUSHAW  PCB$W_DIOCNT(R4)        ;ELSE DIRECT I/O
           9E    B6  0155   417 NALLOC: INCW    @(SP)+                  ;RESTORE COUNT, SINCE NO I/O STARTED
           D8    11  0157   418         BRB     ERROR                   ;
                     0159   419
                     0159   420 ;
                     0159   421 ; Convert section index to window address.
                     0159   422 ;
                     0159   423
     50    04 A6    32  0159   424 SECTION:CVTWL   CCB$L_WIND(R6),R0      ;SIGN EXTEND SECTION INDEX
51   00000000'9F    D0  015D   425         MOVL    @#CTL$GL_PHD,R1         ;GET ADDRESS OF PROCESS HEADER
     51    20 A1    C0  0164   426         ADDL    PHD$L_PSTBASOFF(R1),R1  ;CALCULATE BASE ADDRESS OF SECTION TABLE
 FC A2    0C A140  D0  0168   427         MOVL    SEC$L_WINDOW(R1)[R0],-4(R2)     ;GET ADDRESS OF REAL WINDOW
             52    11  016E   428         BRB     NOSECT                  ;
                     0170   429
          3E A4    3F  0170   430 NODCNT: PUSHAW  PCB$W_DIOCNT(R4)        ;SET FOR DIRECT I/O FUNCTION
             8F    11  0173   431         BRB     NOCNT                   ;
                     0175   432
          3E A4    B7  0175   433 DIRECT: DECW    PCB$W_DIOCNT(R4)        ;CHARGE FOR ANOTHER DIRECT I/O
             F6    19  0178   434         BLSS    NODCNT                  ;BR IF NONE ALLOWED
                     017A   435 OK:
     52  0000'DF    0F  017A   436 GTPKT:  REMQUE  @W^IOC$GL_IRPFL,R2      ;GET I/O PACKET FROM LOOK ASIDE LIST
             C5    1D  017F   437         BVS     ALLOC                   ;IF VS EMPTY LIST
                     0181   438
                     0181   439 ;
                     0181   440 ; BUILD DEVICE INDEPENDENT PART OF I/O PACKET
                     0181   441 ;
                     0181   442 ; R2 - IRP Address
                     0181   443 ; R4 - PCB Address
                     0181   444 ; R5 - UCB Address
                     0181   445 ; R6 - CCB Address
                     0181   446 ; R7 - Function code (original)
                     0181   447 ; R8 - FDT address
                     0181   448 ; R9 - Channel index @ 16 + (IRP$M_BUFIO -- if buffered I/O)
                     0181   449 ; R10 - Function code (transformed)
                     0181   450 ; R11 - Access mode
                     0181   451 ;
          0A A6      B6  0181   452 SUCCES: INCW    CCB$W_IOC(R6)           ;INCREMENT OUTSTANDING I/O ON CHANNEL
                     0184   453         ASSUME  IRP$W_SIZE EQ 8         ;FOR FOLLOWING OPTIMIZATION
        53    82      7E  0184   454         MOVAQ   (R2)+,R3                ;COPY ADDRESS AND ADD IRP$W_SIZE TO R2
     50    14 AC      7D  0187   455         MOVQ    ASTADR(AP),R0           ;INSERT AST ADDRESS AND PARAMETER
58 A3   008C C4      D0  018B   456         MOVL    PCB$L_ARB(R4),IRP$L_ARB(R3) ;COPY ACCESS RIGHTS BLOCK ADDRESS
                     0191   457         ASSUME  IRP$B_RMOD EQ 11        ;FOR SHIFT BELOW
             50      D5  0191   458         TSTL    R0                      ;CHECK FOR AST
             07      13  0193   459         BEQL    5$                      ;NONE
        5B   40 8F    88  0195   460         BISB    #ACB$M_QUOTA,R11        ;NOTE QUOTA CHARGE
          38 A4      B7  0199   461         DECW    PCB$W_ASTCNT(R4)        ;CHARGE QUOTA
     5B   5B   18      78  019C   462 5$:     ASHL    #24,R11,R11             ;ALIGN ACCESS MODE
                     01A0   463         ASSUME  IRP$B_TYPE EQ 10        ;FOR BISL BELOW
```

```
                82 000A00C4 8F    5B    C9  01A0   464          BISL3   R11,#<<DYN$C_IRP@16>!IRP$C_LENGTH>,(R2)+ ;INSERT TYPE AND LENGTH
                  5B    2C A4    10    9C  01A8   465          ROTL    #16,PCB$B_PRIB-3(R4),R11;FETCH AND ALIGN PRIORITY
                                         01AD   466          ASSUME  IRP$L_PID EQ 12         ;
                        82   60 A4    D0  01AD   467          MOVL    PCB$L_PID(R4),(R2)+     ;INSERT PROCESS ID OF CURRENT PROCESS
                        5B   04 AC    90  01B1   468          MOVB    EFN(AP),R11             ;MERGE EVENT FLAG NUMBER
                                         01B5   469
                                         01B5   470          ASSUME  IRP$L_AST EQ 16         ;FOR MOVQ BELOW
                                         01B5   471          ASSUME  IRP$L_ASTPRM EQ 20      ;FOR MOVQ BELOW
                        82   50    7D    01B5   472          MOVQ    R0,(R2)+                ;INSERT AST ADDRESS AND PARAMETER
                     5B   5B    10    9C  01B8   473          ROTL    #16,R11,R11             ;ALIGN PRIORITY AND EVENT FLAG NUMBER
                                         01BC   474
                                         01BC   475          ASSUME  IRP$L_WIND EQ 24
                        82   04 A6    D0  01BC   476          MOVL    CCB$L_WIND(R6),(R2)+    ;GET WINDOW ADDRESS
                             97    14    01C0   477          BGTR    SECTION                 ;BR IF SECTION INDEX
                                         01C2   478
                                         01C2   479          ASSUME  IRP$L_UCB EQ 28
                        82   55    D0  01C2   480 NOSECT:     MOVL    R5,(R2)+                ;INSERT DEVICE UCB ADDRESS
                                         01C5   481          ASSUME  IRP$W_FUNC EQ 32
                        5B   5A    B0  01C5   482          MOVW    R10,R11                 ;MERGE I/O FUNCTION CODE
                                         01C8   483          ASSUME  IRP$B_EFN EQ 34
                                         01C8   484          ASSUME  IRP$B_PRI EQ 35
                                         01C8   485          ASSUME  IRP$L_IOSB EQ 36
                        82   5B    D0  01C8   486          MOVL    R11,(R2)+               ;INSERT PRI,EFN,FUNC
                        58   04    C0  01CB   487          ADDL    #FDTACT-12,R8           ;POINT TO ACTION ROUTINE MASKS
                     82   10 AC    D0  01CE   488          MOVL    IOSB(AP),(R2)+          ;INSERT I/O STATUS BLOCK ADDRESS
                        5C   1C    C0  01D2   489          ADDL    #P1,AP                  ;POINT TO FIRST FUNCTION DEPENDENT PARAMETER
                                         01D5   490          ASSUME  IRP$W_CHAN EQ 40
                                         01D5   491          ASSUME  IRP$W_STS EQ 42
                  82   59    10    9C  01D5   492          ROTL    #16,R9,(R2)+            ;INSERT CHANNEL INDEX AND STATUS
                  59   57    01    CB  01D9   493          BICL3   #1,R7,R9                ;PREPARE FOR VIRTUAL CHECK BELOW
                              82    7C  01DD   494          CLRQ    (R2)+                   ;CLEAR PTE ADDRESS, BYTE OFFSET, AND BYTE CO
                              62    D4  01DF   495          CLRL    (R2)                    ;
                           4C A3    D4  01E1   496          CLRL    IRP$L_DIAGBUF(R3)       ;INIT ORIGINAL PTE ADDR (LOG OR VIRT I/O
                                         01E4   497
                                         01E4   498          .IF DF  CA$_MEASURE_IOT
                     0000'CF    D5  01E4   499          TSTL    W^PMS$GL_IOPFMPDB       ;DATA COLLECTION ENABLED?
                           05    13  01E8   500          BEQL    3$                      ;BR IF NO
                        FE13'    30  01EA   501          BSBW    PMS$START_RQ            ;INSERT START OF I/O REQUEST MESSAGE
                           10    11  01ED   502          BRB     4$                      ;
                                         01EF   503          .ENDC
                                         01EF   504
                                         01EF   505 3$:      SETIPL  #15                     ;DISABLE SOFTWARE INTERRUPTS
50 A3               0000'CF    D0  01F2   506          MOVL    W^PMS$GL_IOPFMSEQ,IRP$L_SEQNUM(R3) ;INSERT PACKET SEQUENCE NUMBER
                    0000'CF    D6  01F8   507          INCL    W^PMS$GL_IOPFMSEQ       ;INCREMENT I/O TRANSACTION SEQUENCE NUMBER
                                         01FC   508          SETIPL  #IPL$_ASTDEL            ;ENABLE INTERRUPTS
                  5B   38 A5    D0  01FF   509 4$:      MOVL    UCB$L_DEVCHAR(R5),R11   ;GET DEVICE CHARACTISTICS FOR MANY
                                         0203   510                                  ;COMPARES BELOW
                                         0203   511 ;
                                         0203   512 ; CHECK IF REQUESTING PROCESS HAS PRIVILEGE TO ACCESS DEVICE
                                         0203   513 ;
                                         0203   514 ; NOTE: LOW BIT OF FUNCTION CODE WAS CLEARED ABOVE
                                         0203   515 ;
                                         0203   516          ASSUME  IO$_READVBLK-IO$_WRITEVBLK EQ 1
                        59   30    D1  0203   517          CMPL    S^#IO$_WRITEVBLK,R9     ;VIRTUAL READ OR WRITE?
                             35    12  0206   518          BNEQ    15$                     ;IF NEQ NO
                  0D 5B    0E    E1  0208   519          BBC     S^#DEV$V_FOD,R11,5$     ;IF CLR, NOT FILE DEVICE
                                         020C   520
```

```
                     020C   521 ;
                     020C   522 ; THE FOLLOWING TEST IS NECESSITATED BY THE SYSTEM INITIALIZATION SEQUENCE
                     020C   523 ;
                     020C   524
          18 A3   D5 020C   525          TSTL    IRP$L_WIND(R3)          ;WINDOW ADDRESS SPECIFIED?
             18   12 020F   526          BNEQ    90$                     ;IF NEQ YES
       5D 5B    13   E1 0211   527          BBC     S^#DEV$V_MNT,R11,60$    ;IF CLR, DEVICE NOT MOUNTED
       6B 5B    18   E1 0215   528          BBC     S^#DEV$V_FOR,R11,80$    ;IF CLR, MOUNTED STRUCTURED
                     0219   529
                     0219   530 ;
                     0219   531 ; CONVERT VIRTUAL READ/WRITE FUNCTION TO ITS LOGICAL COUNTERPART
                     0219   532 ;
                     0219   533
          57   10   C2 0219   534 5$:       SUBL    S^#IO$_READVBLK-IO$_READLBLK,R7 ;CONVERT TO LOGICAL FUNCTION
       20 A3   10   A2 021C   535          SUBW    S^#IO$_READVBLK-IO$_READLBLK,IRP$W_FUNC(R3) ;
    5B 00014040 8F   D3 0220   536          BITL    #<DEV$M_SPL!-           ;NOT SPOOLED,
                     0227   537                  DEV$M_FOD!-             ;NOT FILE DEVICE,
                     0227   538                  DEV$M_SHR>,R11          ;AND NOT SHARABLE
             14   12 0227   539          BNEQ    15$                     ;BR IF SATISFIED
                     0229   540 ;
                     0229   541 ; CHECK IF AST QUOTA IS EXCEEDED
                     0229   542 ;
                     0229   543
          38 A4   B5 0229   544 90$:      TSTW    PCB$W_ASTCNT(R4)        ;AST QUEUE ENTRY QUOTA EXCEEDED?
             4A   19 022C   545          BLSS    75$                     ;IF LEQ YES
                     022E   546
                     022E   547 ;
                     022E   548 ; SCAN FUNCTION DECISION TABLE CALLING EACH SELECTED ACTION ROUTINE WITH:
                     022E   549 ;
                     022E   550 ;       R0 = ADDRESS OF ACTION ROUTINE ENTRY POINT.
                     022E   551 ;       R1 = SCRATCH.
                     022E   552 ;       R2 = SCRATCH.
                     022E   553 ;       R3 = ADDRESS OF I/O REQUEST PACKET.
                     022E   554 ;       R4 = CURRENT PROCESS PCB ADDRESS.
                     022E   555 ;       R5 = ASSIGNED DEVICE UCB ADDRESS.
                     022E   556 ;       R6 = ADDRESS OF CCB.
                     022E   557 ;       R7 = I/O FUNCTION CODE BIT NUMBER.
                     022E   558 ;       R8 = FDT DISPATCH ADDRESS. (UPDATED TO POINT TO ACTION ROUTINE MASKS)
                     022E   559 ;       R9 = SCRATCH.
                     022E   560 ;       R10 = SCRATCH.
                     022E   561 ;       R11 = SCRATCH.
                     022E   562 ;       AP = ADDRESS OF FIRST FUNCTION DEPENDENT PARAMETER.
                     022E   563 ;
                     022E   564 ;       NB: in the Guide to Writing a Device Driver, we document the contents
                     022E   565 ;       of R0 as being the address of the FDT action routine entry point.
                     022E   566 ;       This is the only reason that the dispatch code below does not read:
                     022E   567 ;               JSB     @8(R8)
                     022E   568 ;       Should future gererations wish to modify FDT dispatching to use the
                     022E   569 ;       single dispatch instruction, they must bear the responsibility for
                     022E   570 ;       breaking user written drivers.
                     022E   571
       58   0C   C0 022E   572 110$:     ADDL    #12,R8                  ;POINT TO NEXT FUNCTION MASK
    F9 68   57   E1 0231   573          BBC     R7,(R8),110$            ;IF CLR, THEN ACTION NOT SELECTED
    50   08 A8   D0 0235   574          MOVL    8(R8),R0                ;GET ADDRESS OF ACTION ROUTINE
             60   16 0239   575          JSB     (R0)                    ;CALL ACTION ROUTINE
             F1   11 023B   576          BRB     110$                    ;
                     023D   577
```

```
                     023D   578 ;
                     023D   579 ; CONTINUE DECODING FUNCTIONS OTHER THAN VIRTUAL READ/WRITE
                     023D   580 ;
                     023D   581
        57   2F   D1  023D   582 15$:    CMPL    S^#IO$_LOGICAL,R7       ;VIRTUAL I/O FUNCTION?
             42   19  0240   583         BLSS    80$                     ;IF LSS YES
                     0242   584
                     0242   585 ;
                     0242   586 ; LOGICAL OR PHYSICAL I/O FUNCTION
                     0242   587 ;
                     0242   588
                     0242   589         IFPRIV  PHY_IO,80$              ;PROCESS HAVE PHYSICAL I/O PRIVILEGE?
        57   1F   D1  0248   590         CMPL    S^#IO$_PHYSICAL,R7      ;PHYSICAL I/O FUNCTION?
             78   19  024B   591         BLSS    20$                     ;IF LSS NO
                     024D   592         IFNPRIV LOG_IO,60$              ;PROCESS HAVE LOGICAL I/O PRIVILEGE?
        59   05   D0  0253   593         MOVL    #CCB$V_PHYCHKDON,R9     ;
  5A    0000'CF   9E  0256   594         MOVAB   W^EXE$CHKPHYACCES,R10   ;SET FOR PHYSICAL I/O FUNCTION CHECK
             76   11  025B   595         BRB     30$                     ;
                     025D   596
     23 5B   10   E1  025D   597 40$:    BBC     S^#DEV$V_SHR,R11,80$    ;IF CLR, DEVICE NOT SHAREABLE
                     0261   598 50$:                                     ; R4 - PCB ADDRESS
                     0261   599                                         ; R5 - UCB ADDRESS
 1E 08 A6    59   E0  0261   600         BBS     R9,CCB$B_STS(R6),80$    ;HAS PROT CHECK BEEN MADE?
             6A   16  0266   601         JSB     (R10)                   ;CHECK ACCESS TO VOLUME
          0A 50   E9  0268   602         BLBC    R0,70$                  ;EXIT ON FAILURE
 00 08 A6    59   E2  026B   603         BBSS    R9,CCB$B_STS(R6),55$    ;MARK PROT CHECK DONE
             12   11  0270   604 55$:    BRB     80$                     ;ACCESS ALLOWED
                     0272   605
        50   24   3C  0272   606 60$:    MOVZWL  #SS$_NOPRIV,R0          ;SET NO PRIVILEGE STATUS
           00FA   31  0275   607 70$:    BRW     EXE$ABORTIO             ;
          10 A3   D5  0278   608 75$:    TSTL    IRP$L_AST(R3)           ;DOES THIS REQUEST NEED AN AST?
             B1   13  027B   609         BEQL    110$                    ;NO, THEN CAN'T BE QUOTA EXCEEDED.
        50   1C   3C  027D   610         MOVZWL  #SS$_EXQUOTA,R0         ;AST QUOTA EXCEEDED
             F3   11  0280   611         BRB     70$                     ;
             D9   11  0282   612 77$:    BRB     40$                     ;INTERMEDIATE BRANCH
                     0284   613 ;
                     0284   614 ; PROCESS HAS ACCESS TO DEVICE
                     0284   615 ;
                     0284   616
        57   1F   D1  0284   617 80$:    CMPL    S^#IO$_PHYSICAL,R7      ;LOGICAL OR VIRTUAL I/O FUNCTION?
             A0   19  0287   618         BLSS    90$                     ;IF LSS YES
 2A A3    0100 8F   A8  0289   619         BISW    #IRP$M_PHYSIO,IRP$W_STS(R3) ;SET PHYSICAL I/O FLAG
     59    14 AC   D0  028F   620         MOVL    <P6-P15>(AP),R9         ;GET ADDRESS OF DIAGNOSTIC BUFFER
             94   13  0293   621         BEQL    90$                     ;IF EQL THEN NOT SPECIFIED
                     0295   622 ;
                     0295   623 ; Process diagnostic buffer parameter
                     0295   624 ;
                     0295   625         IFNPRIV DIAGNOSE,60$            ;PROCESS HAVE PRIVILEGE TO DIAGNOSE?
   51    0088 C5   D0  029B   626         MOVL    UCB$L_DDT(R5),R1        ;GET ADDRESS OF DDT
     51    14 A1   3C  02A0   627         MOVZWL  DDT$W_DIAGBUF(R1),R1    ;GET SIZE OF DIAGNOSTIC BUFFER
             1C   13  02A4   628         BEQL    130$                    ;IF EQL NO DIAGNOSTIC FUNCTIONS
             53   DD  02A6   629         PUSHL   R3                      ;SAVE I/O PACKET ADDRESS
           FD55'  30  02A8   630         BSBW    EXE$ALLOCBUF            ;ALLOCATE DIAGNOSTIC BUFFER
             53 8ED0  02AB   631         POPL    R3                      ;RETRIEVE I/O PACKET ADDRESS
          C4 50   E9  02AE   632         BLBC    R0,70$                  ;IF LBC ALLOCATION FAILURE
 4C A3       52   D0  02B1   633         MOVL    R2,IRP$L_DIAGBUF(R3)    ;SAVE ADDRESS OF DIAGNOSTIC BUFFER
   82    0C A2   9E  02B5   634         MOVAB   12(R2),(R2)+            ;SET POINTER TO DATA AREA
```

```
                62    59    D0   02B9   635          MOVL    R9,(R2)                         ;SAVE USER ADDRESS OF DIAGNOSTIC BUFFER
          2A A3 0080  8F    A8   02BC   636          BISW    #IRP$M_DIAGBUF,IRP$W_STS(R3) ;SET DIAGNOSTIC BUFFER PRESENT
                  FF64    31   02C2   637 130$:     BRW     90$
                                02C5   638 ;
                                02C5   639 ; LOGICAL I/O FUNCTION
                                02C5   640 ;
                                02C5   641
                                02C5   642 20$:      IFPRIV  LOG_IO,130$                     ;PROCESS HAVE LOGICAL I/O PRIVILEGE?
                59    04    D0   02CB   643          MOVL    #CCB$V_LOGCHKDON,R9             ;
             5A 0000'CF    9E   02CE   644          MOVAB   W^EXE$CHKLOGACCES,R10           ;SET FOR LOGICAL I/O FUNCTION CHECK
                                02D3   645
                                02D3   646 ;
                                02D3   647 ; PHYSICAL OR LOGICAL I/O FUNCTION - CHECK ACCESSIBILITY OF DEVICE
                                02D3   648 ;
                                02D3   649
             9B 5B    06    E0   02D3   650 30$:      BBS     S^#DEV$V_SPL,R11,60$            ;IF SET, SPOOLED DEVICE
             A7 5B    0E    E1   02D7   651          BBC     S^#DEV$V_FOD,R11,77$            ;IF CLR, NOT FILE DEVICE
             93 5B    13    E1   02DB   652          BBC     S^#DEV$V_MNT,R11,60$            ;IF CLR, DEVICE NOT MOUNTED
             8F 5B    18    E1   02DF   653          BBC     S^#DEV$V_FOR,R11,60$            ;IF CLR, MOUNTED STRUCTURED
                  FF7B    31   02E3   654          BRW     50$                             ;
```

```
02E6    656          .SBTTL  BUILD I/O PACKET FOR PAGE READ/WRITE
02E6    657 ;+
02E6    658 ; EXE$BUILDPKTR - BUILD I/O PACKET FOR PAGE READ
02E6    659 ; EXE$BUILDPKTW - BUILD I/O PACKET FOR PAGE WRITE
02E6    660 ; EXE$BLDPKTSWPR - BUILD I/O PACKET FOR SWAP READ
02E6    661 ; EXE$BLDPKTSWPW - BUILD I/O PACKET FOR SWAP WRITE
02E6    662 ; EXE$BLDPKTGSR - BUILD I/O PACKET FOR SHARED MEMORY GLOBAL SECTION READ
02E6    663 ; EXE$BLDPKTGSW - BUILD I/O PACKET FOR SHARED MEMORY GLOBAL SECTION WRITE
02E6    664 ;
02E6    665 ; THIS ROUTINE IS CALLED TO FILL OUT AND QUEUE AN I/O PACKET
02E6    666 ; FOR A SWAPPING OR PAGING READ OR WRITE.
02E6    667 ;
02E6    668 ; INPUTS:
02E6    669 ;
02E6    670 ;       R0 = VIRTUAL BLOCK NUMBER
02E6    671 ;       R1 = NUMBER OF BYTES TO TRANSFER (PAGE INCREMENTS)
02E6    672 ;       R2 = WINDOW ADDRESS FOR MAPPING VBN TO LBN
02E6    673 ;       R3 = SYSTEM VIRTUAL ADDRESS OF PAGE TABLE ENTRY
02E6    674 ;       R4 = CURRENT PROCESS CONTROL BLOCK ADDRESS
02E6    675 ;            PCB$W_DIOCNT(R4) IS ASSUMED GREATER THAN ZERO
02E6    676 ;            AND MUST BE CHECKED BY THE CALLER.
02E6    677 ;       R5 = I/O REQUEST PACKET ADDRESS
02E6    678 ;            WITH THE FOLLOWING FIELDS ALREADY FILLED IN
02E6    679 ;
02E6    680 ;       IRP$W_SIZE(R5) AND IRP$B_TYPE(R5)
02E6    681 ;           FOR ENTRY AT EXE$BUILDPKTW, EXE$BLDPKTGSR, AND EXE$BLDPKTGSW,
02E6    682 ;               THESE ARE FILLED IN BY THE CALL.  FOR ALL OTHER ENTRY POINTS
02E6    683 ;               THEY ARE FILLED IN BY THIS CODE.
02E6    684 ;       IRP$L_AST(R5) =
02E6    685 ;            FOR PAGE READ CASE - SYSTEM VIRTUAL ADDRESS OF SLAVE (PROCESS)
02E6    686 ;               PAGE TABLE ENTRY FOR THE CASE OF A GLOBAL PAGE READ.
02E6    687 ;               THIS MUST BE 0 FOR A SYSTEM OR PROCESS PAGE READ.
02E6    688 ;            FOR PAGE WRITE CASE - STANDARD QI/O AST ADDRESS
02E6    689 ;            FOR SWAPIO CASE - THIS PARAMETER IS CURRENTLY NOT USED
02E6    690 ;
02E6    691 ;       IRP$L_ASTPRM(R5) =
02E6    692 ;            FOR PAGE READ CASE - THE CONTENTS OF THE FAULTED PAGE TABLE ENTRY
02E6    693 ;               USED TO RECOVER THE ORIGINAL BACKING STORE ADDRESS WHEN A PAGE
02E6    694 ;               READ ERROR OCCURRED FOR A COPY ON REFERENCE PAGE.
02E6    695 ;            FOR PAGE WRITE CASE - STANDARD QI/O AST PARAMETER
02E6    696 ;            FOR SWAPIO CASE - ADDRESS OF KERNEL AST ROUTINE TO CALL
02E6    697 ;
02E6    698 ;       IRP$B_PRI(R5) = THE PRIORITY AT WHICH THE TRANSFER IS TO BE QUEUED
02E6    699 ;
02E6    700 ;       IRP$B_RMOD(R5) =
02E6    701 ;           FOR PAGE WRITE CASE - STANDARD QI/O MODE OF REQUESTER
02E6    702 ;           FOR ALL OTHER CASES - CONTAINS GARBAGE WHICH IS IGNORED
02E6    703 ;
02E6    704 ;       IRP$B_EFN(R5)  =
02E6    705 ;           FOR PAGE WRITE CASE - STANDARD QI/O EVENT FLAG NUMBER
02E6    706 ;           FOR ALL OTHER CASES - CONTAINS GARBAGE WHICH IS IGNORED
02E6    707 ;
02E6    708 ;       IRP$L_IOSB(R5) =
02E6    709 ;           FOR PAGE WRITE CASE - STANDARD QI/O I/O STATUS BLOCK ADDRESS
02E6    710 ;           FOR ALL OTHER CASES - CONTAINS GARBAGE WHICH IS IGNORED
02E6    711 ;
02E6    712 ; OUTPUTS:
```

```
                                                    K 12
                     02E6   713 ;
                     02E6   714 ;       R4,R5 ALTERED
                     02E6   715 ;-
                     02E6   716
                     02E6   717         .ENABL  LSB
                     02E6   718
                     02E6   719 ; Note that the differentiation between READ and WRITE operations is
                     02E6   720 ; encoded in the setting of the IRP$M_FUNC bit.
                     02E6   721 ;
                     02E6   722 ;       IRP$M_FUNC = 1 => IO$_READPBLK  ; Read operation
                     02E6   723 ;       IRP$M_FUNC = 0 => IO$_WRITEPBLK ; Write operation
                     02E6   724
                     02E6   725 EXE$BLDPKTGSR::                          ;BUILD PACKET FOR SHMGSD READ
00520000 8F    DD    02E6   726         PUSHL   #<IRP$M_SWAPIO ! IRP$M_VIRTUAL ! IRP$M_FUNC>@16
         30    11    02EC   727         BRB     20$                       ;TYPE/SIZE ALREADY SET IN PACKET
                     02EE   728
                     02EE   729 EXE$BLDPKTGSW::                          ;BUILD PACKET FOR SHMGSD WRITE
00500000 8F    DD    02EE   730         PUSHL   #<IRP$M_SWAPIO ! IRP$M_VIRTUAL>@16
         28    11    02F4   731         BRB     20$                       ;TYPE/SIZE ALREADY SET IN PACKET
                     02F6   732
                     02F6   733 EXE$BLDPKTSWPR::                         ;BUILD SWAP READ PACKET
00520000 8F    DD    02F6   734         PUSHL   #<IRP$M_SWAPIO ! IRP$M_VIRTUAL ! IRP$M_FUNC>@16
         16    11    02FC   735         BRB     10$                       ;
                     02FE   736
                     02FE   737 EXE$BLDPKTSWPW::                         ;BUILD SWAP WRITE PACKET
00500000 8F    DD    02FE   738         PUSHL   #<IRP$M_SWAPIO ! IRP$M_VIRTUAL>@16
         0E    11    0304   739         BRB     10$                       ;
                     0306   740
                     0306   741 EXE$BUILDPKTW::                          ;BUILD I/O PACKET FOR PAGE WRITE
00140000 8F    DD    0306   742         PUSHL   #<IRP$M_PAGIO ! IRP$M_VIRTUAL>@16
         10    11    030C   743         BRB     20$                       ;
                     030E   744
                     030E   745 EXE$BUILDPKTR::                          ;BUILD I/O PACKET FOR PAGE READ
00160000 8F    DD    030E   746         PUSHL   #<IRP$M_PAGIO ! IRP$M_VIRTUAL ! IRP$M_FUNC>@16
                     0314   747
000A00C4 8F    F0    0314   748 10$:    INSV    #<DYN$C_IRP@16 ! IRP$C_LENGTH>,- ;SET SIZE
   08 A5    18    00  031A   749                 #0,#24,IRP$W_SIZE(R5)     ;AND TYPE OF PACKET
      2C A5    53    D0  031E   750 20$:    MOVL    R3,IRP$L_SVAPTE(R5)       ;SYSTEM VIRTUAL ADR OF PAGE TABLE ENTRY
      4C A5    53    D0  0322   751         MOVL    R3,IRP$L_DIAGBUF(R5)      ;NEED COPY OF ORIGINAL FOR SEGMENTED XFERS
         53    55    D0  0326   752         MOVL    R5,R3                     ;PACKET ADDRESS TO R3
55   6E    01    11    EE  0329   753         EXTV    #<IRP$V_FUNC+16>,#1,(SP),R5
                     032E   754
                     032E   755 ; R5 = -1 for read
                     032E   756 ; R5 =  0 for write
                     032E   757
                     032E   758         ASSUME  <IO$_WRITEPBLK + 1> EQ IO$_READPBLK
                     032E   759         ASSUME  <WCB$L_WRITES - 4> EQ WCB$L_READS
                     032E   760
      28 A245    D6  032E   761         INCL    WCB$L_WRITES(R2)[R5]      ;USE CODE TO BUMP READ OR WRITE COUNT
   20 A3    0B    55  A3  0332   762         SUBW3   R5,#IO$_WRITEPBLK,IRP$W_FUNC(R3) ;SET REAL FUNCTION CODE
55   8E  0000FFFF 8F    CB  0337   763         BICL3   #^XFFFF,(SP)+,R5          ;GET STATUS BITS AND CLEAR CHANNEL
                     033F   764
                     033F   765         ASSUME  IRP$W_STS EQ IRP$W_CHAN+2
                     033F   766
      28 A3    55    D0  033F   767         MOVL    R5,IRP$W_CHAN(R3)         ;SET CHANNEL AND STATUS
      55    10 A2    D0  0343   768         MOVL    WCB$L_ORGUCB(R2),R5       ;GET UCB ADDRESS FROM WINDOW
      1C A3    55    D0  0347   769         MOVL    R5,IRP$L_UCB(R3)          ;SET UCB ADDRESS
```

```
                OC A3    60 A4   D0   034B   770          MOVL    PCB$L_PID(R4),IRP$L_PID(R3) ;PROCESS ID FROM PCB
                   48 A3    50   D0   0350   771          MOVL    R0,IRP$L_SEGVBN(R3)         ;STARTING VIRTUAL BLOCK NUMBER
                   18 A3    52   D0   0354   772          MOVL    R2,IRP$L_WIND(R3)           ;WINDOW ADDRESS
            58 A3    008C C4   D0   0358   773          MOVL    PCB$L_ARB(R4),IRP$L_ARB(R3) ;ACCESS RIGHTS BLOCK ADDRESS
                                    035E   774
                      30 A3   B4   035E   775          CLRW    IRP$W_BOFF(R3)              ;ZERO BYTE OFFSET
                   32 A3    51   D0   0361   776          MOVL    R1,IRP$L_BCNT(R3)           ;SET BYTE COUNT
                      40 A3   D4   0365   777          CLRL    IRP$L_ABCNT(R3)             ;ZERO ACCUMULATED BYTE COUNT
                   44 A3    51   D0   0368   778          MOVL    R1,IRP$L_OBCNT(R3)          ;SET ORIGINAL BYTE COUNT
                                    036C   779
                                    036C   780          .IF     DF,CA$_MEASURE_IOT
                                    036C   781
                         FC91'  30   036C   782          BSBW    PMS$START_RQ                ;INSERT START OF I/O REQUEST MESSAGE
                                    036F   783
                                    036F   784          .ENDC
                                    036F   785
                         FC8E'  31   036F   786          BRW     IOC$QNXTSEG1                ;QUEUE THE FIRST SEGMENT OF THE I/O REQUEST
                                    0372   787                                              ;AND RETURN
                                    0372   788          .DSABL  LSB
```

```
                     0372   790          .SBTTL  COMPLETE I/O OPERATION
                     0372   791 ;+
                     0372   792 ; EXE$ABORTIO - ABORT I/O OPERATION
                     0372   793 ;
                     0372   794 ; THIS ROUTINE IS JUMPED TO FROM A FUNCTION DECISION TABLE ACTION ROUTINE
                     0372   795 ; TO FINISH AN I/O OPERATION WITHOUT RETURNING THE FINAL I/O STATUS.
                     0372   796 ;
                     0372   797 ; EXE$FINISHIO - FINISH I/O OPERATION
                     0372   798 ;
                     0372   799 ; THIS ROUTINE IS JUMPED TO FROM A FUNCTION DECISION TABLE ACTION ROUTINE
                     0372   800 ; TO FINISH AN I/O OPERATION AND RETURN THE FINAL I/O STATUS.
                     0372   801 ;
                     0372   802 ; EXE$FINISHIOC - FINISH I/O OPERATION WITH SECOND I/O STATUS LONGWORD CLEARED
                     0372   803 ;
                     0372   804 ; THIS ROUTINE IS JUMPED TO FROM A FUNCTION DESCISION TABLE ACTION ROUTINE
                     0372   805 ; TO FINISH AN I/O OPERATION AND RETURN THE FINAL I/O STATUS WITH THE
                     0372   806 ; SECOND I/O STATUS LONGWORD CLEARED.
                     0372   807 ;
                     0372   808 ; INPUTS:
                     0372   809 ;
                     0372   810 ;       R0 = FIRST LONGWORD OF FINAL I/O STATUS.
                     0372   811 ;       R1 = SECOND LONGWORD OF FINAL I/O STATUS.
                     0372   812 ;       R3 = ADDRESS OF I/O REQUEST PACKET.
                     0372   813 ;       R4 = CURRENT PROCESS PCB ADDRESS.
                     0372   814 ;       R5 = UCB ADDRESS OF DEVICE UNIT.
                     0372   815 ;
                     0372   816 ; OUTPUTS:
                     0372   817 ;
                     0372   818 ;       THE FINAL I/O STATUS IS STORED IN THE I/O PACKET AND THE PACKET IS
                     0372   819 ;       INSERTED IN THE I/O POST PROCESSING QUEUE. A SOFTWARE INTERRUPT
                     0372   820 ;       IS GENERATED TO INITIATE I/O POST PROCESSING AND THE FIRST WORD
                     0372   821 ;       OF THE FINAL I/O STATUS IS RETURNED AS THE SERVICE STATUS.
                     0372   822 ;-
                     0372   823
                     0372   824          .ENABL  LSB
                     0372   825 EXE$ABORTIO::                              ;ABORT I/O OPERATION
          24 A3  D4  0372   826          CLRL    IRP$L_IOSB(R3)           ;CLEAR ADDRESS OF I/O STATUS BLOCK
11 0B A3    06  E5  0375   827          BBCC    #ACB$V_QUOTA,IRP$B_RMOD(R3),10$ ;IF CLR, NO AST SPECIFIED
          38 A4  B6  037A   828          INCW    PCB$W_ASTCNT(R4)         ;UPDATE AVAILABLE AST QUEUE ENTRIES
             0C  11  037D   829          BRB     10$                      ;
                     037F   830 EXE$FINISHIOC::                           ;FINISH I/O OPERATION CLEAR SECOND LONGWORD
             51  D4  037F   831          CLRL    R1                       ;CLEAR SECOND I/O STATUS LONGWORD
                     0381   832 EXE$FINISHIO::                            ;FINISH I/O OPERATION
          70 A5  D6  0381   833          INCL    UCB$L_OPCNT(R5)          ;INCREMENT OPERATIONS COMPLETED
    38 A3    50  7D  0384   834          MOVQ    R0,IRP$L_MEDIA(R3)       ;STORE FINAL I/O STATUS
       50    01  3C  0388   835          MOVZWL  S^#SS$_NORMAL,R0         ;SET NORMAL COMPLETION STATUS
0000'DF      63  0E  038B   836 10$:     INSQUE  (R3),@@^IOC$GL_PSBL      ;INSERT I/O PACKET IN POST PROCESS QUEUE
                     0390   837          SOFTINT #IPL$_IOPOST             ;INITIATE SOFTWARE INTERRUPT
             39  11  0393   838          BRB     QIORETURN                ;
                     0395   839          .DSABL  LSB
```

```
                     0395   841          .SBTTL  QUEUE I/O PACKET TO DRIVER
                     0395   842 ;+
                     0395   843 ; EXE$QIODRVPKT - QUEUE I/O PACKET TO DRIVER
                     0395   844 ;
                     0395   845 ; THIS ROUTINE IS JUMPED TO FROM A FUNCTION DECISION TABLE ACTION ROUTINE
                     0395   846 ; TO QUEUE AN I/O PACKET TO THE APPROPRIATE DRIVER.
                     0395   847 ;
                     0395   848 ; INPUTS:
                     0395   849 ;
                     0395   850 ;       R3 = ADDRESS OF I/O REQUEST PACKET.
                     0395   851 ;       R4 = CURRENT PROCESS PCB ADDRESS.
                     0395   852 ;       R5 = UCB ADDRESS OF DEVICE UNIT.
                     0395   853 ;
                     0395   854 ; OUTPUTS:
                     0395   855 ;
                     0395   856 ;       THE I/O PACKET IS QUEUED BY PRIORITY IN THE APPROPRIATE DEVICE
                     0395   857 ;       QUEUE AND A NORMAL COMPLETION STATUS IS RETURNED.
                     0395   858 ;-
                     0395   859
                     0395   860 EXE$QIODRVPKT::                          ;QUEUE I/O PACKET
66   10              0395   861          BSBB    EXE$INSIOQ              ;INSERT I/O PACKET IN DEVICE QUEUE
32   11              0397   862          BRB     EXE$QIORETURN           ;
```

```
                     0399   864          .SBTTL  EXE$ALTQUEPKT - Call driver ALTSTART entry point
                     0399   865
                     0399   866 ;
                     0399   867 ; EXE$ALTQUEPKT - activates a driver at its ALTSTART entry point
                     0399   868 ;
                     0399   869 ; Routine description:
                     0399   870 ;
                     0399   871 ;       Locates and calls a driver entry point supplied as an alternate
                     0399   872 ;       START I/O entry point. Does not test for unit busy before the
                     0399   873 ;       call. Exits by returning to caller.
                     0399   874 ;
                     0399   875 ;       The routine expects to gain control at or below driver fork
                     0399   876 ;       level. The routine raises to driver fork IPL before the call,
                     0399   877 ;       and restores the previous IPL before returning to its caller.
                     0399   878 ;
                     0399   879 ; Inputs:
                     0399   880 ;
                     0399   881 ;       R3      - address of packet or buffer
                     0399   882 ;       R5      - address of UCB
                     0399   883 ;
                     0399   884 ; Outputs:
                     0399   885 ;
                     0399   886 ;       Control returns to the requesting process.
                     0399   887 ;
                     0399   888 ;       The routine destroys R0-R1.
                     0399   889 ;
                     0399   890 ;--
                     0399   891
                     0399   892 EXE$ALTQUEPKT::                          ; Start I/O in driver.
                     0399   893          DSBINT  UCB$B_FIPL(R5)          ; Raise to fork IPL.
50   0088 C5   D0   03A0   894          MOVL    UCB$L_DDT(R5),R0        ; Get address of unit's DDT.
       1C B0   16   03A5   895          JSB     @DDT$L_ALTSTART(R0)     ; Call alternate start I/O routine.
                    03A8   896          ENBINT                          ; Reenable interrupts.
                    05     03AB   897          RSB                             ; Return to caller.
```

```
                                                C 13
                     03AC   899          .SBTTL  QUEUE I/O PACKET TO ACP
                     03AC   900 ;+
                     03AC   901 ; EXE$QIOACPPKT - QUEUE I/O PACKET TO ACP
                     03AC   902 ;
                     03AC   903 ; THIS ROUTINE IS JUMPED TO FROM A FUNCTION DECISION TABLE ACTION ROUTINE
                     03AC   904 ; TO QUEUE AN I/O PACKET TO THE APPROPRIATE ACP OR XQP.
                     03AC   905 ;
                     03AC   906 ; INPUTS:
                     03AC   907 ;
                     03AC   908 ;       R3 = ADDRESS OF I/O REQUEST PACKET.
                     03AC   909 ;       R4 = CURRENT PROCESS PCB ADDRESS.
                     03AC   910 ;       R5 = UCB ADDRESS OF DEVICE UNIT.
                     03AC   911 ;
                     03AC   912 ;       CURRENT IPL MUST BE AT SYNCH OR HIGHER LEVEL.
                     03AC   913 ;
                     03AC   914 ; OUTPUTS:
                     03AC   915 ;
                     03AC   916 ;       R4 ALTERED
                     03AC   917 ;       THE I/O PACKET IS QUEUED AT THE END OF THE APPROPRIATE ACP OR XQP QUEUE
                     03AC   918 ;       AND A NORMAL COMPLETION STATUS IS RETURNED.
                     03AC   919 ;-
                     03AC   920
                     03AC   921 EXE$QIOACPPKT::                              ;QUEUE I/O PACKET TO ACP
     52   34 A5   D0 03AC   922          MOVL    UCB$L_VCB(R5),R2            ;GET ADDRESS OF VCB
     52   10 A2   D0 03B0   923          MOVL    VCB$L_AQB(R2),R2            ;GET ADDRESS OF ACP AQB
          0C A2   D5 03B4   924          TSTL    AQB$L_ACPPID(R2)            ;GET ADDRESS OF AQB
             19   13 03B7   925          BEQL    XQP                         ;EQL IF IT'S FOR XQP
             60   10 03B9   926          BSBB    EXE$INSERTIRP               ;INSERT I/O PACKET IN ACP QUEUE
             0E   12 03BB   927          BNEQ    EXE$QIORETURN               ;IF NEQ NOT FIRST ENTRY IN QUEUE
     51      0C A2   D0 03BD   928          MOVL    AQB$L_ACPPID(R2),R1         ;GET ACP PROCESS ID
              FC3C'  30 03C1   929          BSBW    SCH$WAKE                    ;WAKE UP ACP PROCESS
          04  50   E8 03C4   930          BLBS    R0,EXE$QIORETURN            ;IF LBS ACP STILL PRESENT
                     03C7   931          BUG_CHECK NONEXSTACP                ;NONEXISTENT ACP PROCESS
                     03CB   932 EXE$QIORETURN::                              ;QUEUE I/O REQUEST COMPLETION STATUS RETURN
        50   01   3C 03CB   933          MOVZWL  #SS$_NORMAL,R0              ;SET NORMAL COMPLETION STATUS
                     03CE   934 QIORETURN:                                   ;RETURN SPECIFIED STATUS
                     03CE   935          SETIPL  #0                          ;ALLOW ALL INTERRUPTS
                     03D1   936          RET                                 ;
                     03D2   937 XQP:
     55   60 A3   9E 03D2   938          MOVAB   IRP$L_FQFL(R3), R5          ;USE CDRP PART OF IRP AS ACB
                     03D6   939          SETIPL  #IPL$_ASTDEL                ;ALLOW PAGEFAULTS
          F2 AF   9F 03D9   940          PUSHAB  QIORETURN                   ;RETURN ADDRESS FROM EXE$QXQPPKT
                     03DC   941 ;
                     03DC   942 ; FALL THROUGH TO XQP QUEUEING ROUTINE IMMEDIATELY FOLLOWING.
                     03DC   943 ; RSB FROM THIS ROUTINE RETURNS TO EXIT ABOVE.
                     03DC   944 ;
                     03DC   945
                     03DC   946          .SBTTL  EXE$QXQPPKT - QUEUE I/O PACKET TO XQP
                     03DC   947 ;+
                     03DC   948 ; EXE$QXQPPKT - INSERT I/O PACKET IN XQP QUEUE
                     03DC   949 ;
                     03DC   950 ; THIS ROUTINE IS CALLED TO INSERT AN I/O PACKET IN THE XQP QUEUE
                     03DC   951 ; AND START THE THREAD OF EXECUTION IF IT IS THE ONLY REQUEST.
                     03DC   952 ;
                     03DC   953 ; CALLING SEQUENCE:
                     03DC   954 ;       BSB/JSB EXE$QXQPPKT - THIS IS EITHER CALLED FROM QIO OR
                     03DC   955 ;       AS A SPECIAL KERNEL AST INVOKED BY IOPOST.
```

```
                                                     D 13
                     03DC   956 ;
                     03DC   957 ; INPUTS:
                     03DC   958 ;
                     03DC   959 ;       R4 = CURRENT PROCESS PCB ADDRESS.
                     03DC   960 ;       R5 = ADDRESS OF TEMP ACB PART OF IRP.
                     03DC   961 ;
                     03DC   962 ; OUTPUTS:
                     03DC   963 ;
                     03DC   964 ;       R0 = status from SCH$QAST.
                     03DC   965 ;
                     03DC   966 ;       IF SUCCESS:
                     03DC   967 ;       A KERNEL AST IS QUEUED TO THE DISPATCH ROUTINE OF THE XQP
                     03DC   968 ;       IF NO PACKETS WERE ALREADY ON THE REQUEST QUEUE OF THE XQP.
                     03DC   969 ;
                     03DC   970 ;       THIS ROUTINE MUST BE CALLED AT IPL ASTDEL SO THAT THE
                     03DC   971 ;       IRP CANNOT BE LOST (BECAUSE OF PROCESS DELETION) UNTIL IT
                     03DC   972 ;       IS PLACED ON THE XQP REQUEST QUEUE.
                     03DC   973 ;
                     03DC   974 ;-
                     03DC   975
                     03DC   976 EXE$QXQPPKT::
50   00000000'GF   D0   03DC   977          MOVL    G^CTL$GL_F11BXQP, R0    ;ADDR OF XQP QUEUE HEAD
   14 A5    A0 A5   9E   03E3   978          MOVAB   CDRP$L_IOQFL(R5), -     ;ADDRESS OF IRP
                     03E8   979                  ACB$L_ASTPRM(R5)        ;IS AST PARAMETER.
            20   90   03E8   980          MOVB    #PSL$C_KERNEL!ACB$M_NODELETE,- ;KERNEL MODE, DON'T DELETE IRP
         0B A5        03EA   981                  ACB$B_RMOD(R5)
0C A5    60 A4   D0   03EC   982          MOVL    PCB$L_PID(R4), ACB$L_PID(R5) ;COPY PID.
10 A5    08 A0   D0   03F1   983          MOVL    F11B$C_DISPATCH(R0), ACB$L_AST(R5) ;XQP DISPATCHER ADDRESS.
      52    02   D0   03F6   984          MOVL    #PRI$_RESAVL, R2        ;SAME AS AFTER WAITING FOR A LOCK.
         FC04'   30   03F9   985          BSBW    SCH$QAST                ;QUEUE THE AST.
                 05   03FC   986          RSB                             ;AND RETURN
```

```
                     03FD   988          .SBTTL  INSERT I/O PACKET IN UNIT QUEUE
                     03FD   989 ;+
                     03FD   990 ; EXE$INSIOQ - INSERT I/O PACKET IN UNIT QUEUE
                     03FD   991 ;
                     03FD   992 ; THIS ROUTINE IS CALLED TO INSERT AN I/O PACKET IN A UNIT QUEUE AND CALL
                     03FD   993 ; THE APPROPRIATE I/O DRIVER IF THE UNIT IS NOT BUSY.
                     03FD   994 ;
                     03FD   995 ; INPUTS:
                     03FD   996 ;
                     03FD   997 ;       R3 = ADDRESS OF I/O REQUEST PACKET.
                     03FD   998 ;       R5 = UCB ADDRESS OF DEVICE UNIT.
                     03FD   999 ;-
                     03FD  1000
                     03FD  1001 EXE$INSIOQ::                                ;INSERT IN I/O QUEUE
                     03FD  1002          DSBINT  UCB$B_FIPL(R5)             ;RAISE IPL TO FORK LEVEL
          6A A5   B6 0404  1003          INCW    UCB$W_QLEN(R5)             ; Bump device queue length
05 64 A5    08   E2 0407  1004          BBSS    #UCB$V_BSY,UCB$W_STS(R5),10$ ;IF SET, THEN DEVICE IS BUSY
        FBF1'    30 040C  1005          BSBW    IOC$INITIATE               ;INITIATE I/O FUNCTION
          06     11 040F  1006          BRB     20$                        ;
    52   4C A5  DE 0411  1007 10$:     MOVAL   UCB$L_IOQFL(R5),R2         ;GET ADDRESS OF I/O QUEUE LISTHEAD
          04     10 0415  1008          BSBB    EXE$INSERTIRP              ;INSERT I/O PACKET IN DEVICE QUEUE
                 0417  1009 20$:      ENBINT                             ;ENABLE INTERRUPTS
                05 041A  1010          RSB                                ;
```

```
                     041B  1012          .SBTTL  INSERT I/O PACKET IN QUEUE BY PRIORITY
                     041B  1013 ;+
                     041B  1014 ; EXE$INSERTIRP - INSERT I/O PACKET IN QUEUE BY PRIORITY
                     041B  1015 ;
                     041B  1016 ; THIS ROUTINE IS CALLED TO INSERT AN I/O PACKET IN A SPECIFIED QUEUE BY
                     041B  1017 ; PRIORITY.
                     041B  1018 ;
                     041B  1019 ; INPUTS:
                     041B  1020 ;
                     041B  1021 ;       R2 = ADDRESS OF QUEUE LISTHEAD.
                     041B  1022 ;       R3 = ADDRESS OF I/O PACKET.
                     041B  1023 ;
                     041B  1024 ;       CURRENT IPL MUST BE THE FORK LEVEL OF THE RESPECTIVE DRIVER PROCESS
                     041B  1025 ;       OR HIGHER.
                     041B  1026 ;
                     041B  1027 ; OUTPUTS:
                     041B  1028 ;
                     041B  1029 ;       THE I/O PACKET IS INSERTED IN THE SPECIFIED QUEUE BY PRIORITY AND
                     041B  1030 ;       THE 'Z' CONDITION CODE IS RETURNED TO THE CALLER.
                     041B  1031 ;
                     041B  1032 ;               'Z' = 1 = ENTRY WAS FIRST ENTRY IN THE QUEUE.
                     041B  1033 ;
                     041B  1034 ;               'Z' = 0 = ENTRIES WERE ALREADY IN THE QUEUE.
                     041B  1035 ;
                     041B  1036 ;       R2 AND R3 ARE PRESERVED ACROSS THE CALL.
                     041B  1037 ;-
                     041B  1038
                     041B  1039 EXE$INSERTIRP::                              ;INSERT I/O PACKET IN QUEUE BY PRIORITY
          51   52    D0  041B  1040          MOVL    R2,R1                   ;COPY LISTHEAD ADDRESS
     51   04  A1    D0  041E  1041 10$:     MOVL    IRP$L_IOQBL(R1),R1      ;GET ADDRESS OF NEXT ENTRY
          51   52    D1  0422  1042          CMPL    R2,R1                   ;END OF QUEUE?
               07    13  0425  1043          BEQL    20$                     ;IF EQL YES
23 A1    23   A3    91  0427  1044          CMPB    IRP$B_PRI(R3),IRP$B_PRI(R1) ;NEW ENTRY PRIORITY GREATER?
               F0    1F  042C  1045          BLSSU   10$                     ;IF LSS YES
          61   63    0E  042E  1046 20$:     INSQUE  IRP$L_IOQFL(R3),IRP$L_IOQFL(R1) ;INSERT PACKET IN I/O QUEUE
                     05  0431  1047          RSB                             ;
                         0432  1048
                         0432  1049          .END
```

```
ACB$B_RMOD                = 0000000B
ACB$L_AST                 = 00000010
ACB$L_ASTPRM              = 00000014
ACB$L_PID                 = 0000000C
ACB$M_NODELETE            = 00000020
ACB$M_QUOTA               = 00000040
ACB$V_QUOTA               = 00000006
ACCVIO                      0000012E R      01
ALLOC                       00000146 R      01
AQB$L_ACPPID              = 0000000C
ASTADR                    = 00000014
ASTPRM                    = 00000018
BUG$_NONEXSTACP             ********   X   01
CCB$B_AMOD                = 00000009
CCB$B_STS                 = 00000008
CCB$C_LENGTH              = 00000010
CCB$L_UCB                 = 00000000
CCB$L_WIND                = 00000004
CCB$M_RDCHKDON            = 00000004
CCB$M_WRTCHKDON           = 00000008
CCB$V_LOGCHKDON           = 00000004
CCB$V_PHYCHKDON           = 00000005
CCB$V_RDCHKDON            = 00000002
CCB$V_WRTCHKDON           = 00000003
CCB$W_IOC                 = 0000000A
CDRP$C_IOQFL              = FFFFFFA0
CHAN                      = 00000008
CHKDON                      000000D2 R      01
CLREF                       00000040 R      01
CTL$GL_CCBBASE              ********   X    01
CTL$GL_F11BXQP              ********   X    01
CTL$GL_PHD                  ********   X    01
CTL$GW_CHINDX               ********   X    01
DACSPND                     00000052 R      01
DDT$L_ALTSTART            = 0000001C
DDT$L_FDT                 = 00000008
DDT$W_DIAGBUF             = 00000014
DEV$M_FOD                 = 00004000
DEV$M_SHR                 = 00010000
DEV$M_SPL                 = 00000040
DEV$V_FOD                 = 0000000E
DEV$V_FOR                 = 00000018
DEV$V_MNT                 = 00000013
DEV$V_SHR                 = 00000010
DEV$V_SPL                 = 00000006
DIRECT                      00000175 R      01
DYN$C_IRP                 = 0000000A
EFN                       = 00000004
ERROR                       00000131 R      01
ERRORB                      0000004F R      01
EXE$ABORTIO                 00000372 RG     01
EXE$ALLOCBUF                ********   X    01
EXE$ALLOCIRP                ********   X    01
EXE$ALTQUEPKT               00000399 RG     01
EXE$BLDPKTGSR               000002E6 RG     01
EXE$BLDPKTGSW               000002EE RG     01
EXE$BLDPKTSWPR              000002F6 RG     01
EXE$BLDPKTSWPW              000002FE RG     01
EXE$BUILDPKTR               0000030E RG     01
EXE$BUILDPKTW               00000306 RG     01
EXE$CHKLOGACCES             ********    X   01
EXE$CHKPHYACCES             ********    X   01
EXE$CHKRDACCES              ********    X   01
EXE$CHKWRTACCES             ********    X   01
EXE$FINISHIO                00000381 RG     01
EXE$FINISHIOC               0000037F RG     01
EXE$INSERTIRP               0000041B RG     01
EXE$INSIOQ                  000003FD RG     01
EXE$QIO                     00000075 RG     01
EXE$QIOACPPKT               000003AC RG     01
EXE$QIODRVPKT               00000395 RG     01
EXE$QIORETURN               000003CB RG     01
EXE$QXQPPKT                 000003DC RG     01
EXE$SNGLEQUOTA              ********    X   01
F11B$L_DISPATCH           = 00000008
FDTACT                    = 00000010
FUNC                      = 0000000C
GTPKT                       0000017A R      01
ILLIO                       00000116 R      01
IO$M_FCODE                = 0000003F
IO$_ACPCONTROL            = 00000038
IO$_DEACCESS              = 00000034
IO$_LOGICAL               = 0000002F
IO$_PHYSICAL              = 0000001F
IO$_READHEAD              = 0000000E
IO$_READLBLK              = 00000021
IO$_READPBLK              = 0000000C
IO$_READPROMPT            = 00000037
IO$_READTRACKD            = 00000010
IO$_READVBLK              = 00000031
IO$_REREADN               = 00000016
IO$_REREADP               = 00000017
IO$_TTYREADALL            = 0000003A
IO$_TTYREADPALL           = 0000003B
IO$_VIRTUAL               = 0000003F
IO$_WRITECHECK            = 0000000A
IO$_WRITECHECKH           = 00000018
IO$_WRITEHEAD             = 0000000D
IO$_WRITELBLK             = 00000020
IO$_WRITEPBLK             = 0000000B
IO$_WRITERET              = 00000018
IO$_WRITETRACKD           = 0000000F
IO$_WRITEVBLK             = 00000030
IOC$GL_IRPFL                ********   X   01
IOC$GL_PSBL                 ********   X   01
IOC$INITIATE                ********   X   01
IOC$QNXTSEG1                ********   X   01
IOSB                      = 00000010
IOTYPE                    = 00000008
IPL$_ASTDEL               = 00000002
IPL$_IOPOST               = 00000004
IPL$_SYNCH                = 00000008
IRP$B_EFN                 = 00000022
IRP$B_PRI                 = 00000023
```

IRP$B_RMOD	=	0000000B			PCB$L_EFWM	=	0000004C			
IRP$B_TYPE	=	0000000A			PCB$L_PID	=	00000060			
IRP$C_LENGTH	=	000000C4			PCB$Q_PRIV	=	00000084			
IRP$L_ABCNT	=	00000040			PCB$W_ASTCNT	=	00000038			
IRP$L_ARB	=	00000058			PCB$W_BIOCNT	=	0000003A			
IRP$L_AST	=	00000010			PCB$W_DIOCNT	=	0000003E			
IRP$L_ASTPRM	=	00000014			PHD$L_PSTBASOFF	=	00000020			
IRP$L_BCNT	=	00000032			PMS$GL_IOPFMPDB		********	X	01	
IRP$L_DIAGBUF	=	0000004C			PMS$GL_IOPFMSEQ		********	X	01	
IRP$L_FQFL	=	00000060			PMS$START_RQ		********	X	01	
IRP$L_IOQBL	=	00000004			PR$_IPL	=	00000012			
IRP$L_IOQFL	=	00000000			PR$_SIRR	=	00000014			
IRP$L_IOSB	=	00000024			PRI$_RESAVL	=	00000002			
IRP$L_MEDIA	=	00000038			PRIOSB		000000E4	R	01	
IRP$L_OBCNT	=	00000044			PRIVERR		0000004C	R	01	
IRP$L_PID	=	0000000C			PRV$V_DIAGNOSE	=	00000006			
IRP$L_SEGVBN	=	00000048			PRV$V_LOG_IO	=	00000007			
IRP$L_SEQNUM	=	00000050			PRV$V_PHY_IO	=	00000016			
IRP$L_SVAPTE	=	0000002C			PSL$C_KERNEL	=	00000000			
IRP$L_UCB	=	0000001C			PSL$S_PRVMOD	=	00000002			
IRP$L_WIND	=	00000018			PSL$V_PRVMOD	=	00000016			
IRP$M_BUFIO	=	00000001			QIO		00000077	R	01	
IRP$M_DIAGBUF	=	00000080			QIORETURN		000003CE	R	01	
IRP$M_FUNC	=	00000002			READ_ACCESS		00000000	R	01	
IRP$M_PAGIO	=	00000004			RSN$_ASTWAIT	=	00000001			
IRP$M_PHYSIO	=	00000100			SCH$CLREF		********	X	01	
IRP$M_SWAPIO	=	00000040			SCH$GL_RESMASK		********	X	01	
IRP$M_VIRTUAL	=	00000010			SCH$GQ_MWAIT		********	X	01	
IRP$V_FUNC	=	00000001			SCH$POSTEF		********	X	01	
IRP$W_BOFF	=	00000030			SCH$QAST		********	X	01	
IRP$W_CHAN	=	00000028			SCH$WAIT		********	X	01	
IRP$W_FUNC	=	00000020			SCH$WAKE		********	X	01	
IRP$W_SIZE	=	00000008			SEC$L_WINDOW	=	0000000C			
IRP$W_STS	=	0000002A			SECTION		00000159	R	01	
IVCHAN		00000045	R	01	SPOOL		00000068	R	01	
LEGAL	=	00000000			SS$_ACCVIO	=	0000000C			
MASKH	=	00010001			SS$_DEVOFFLINE	=	00000084			
MASKL	=	0100AC00			SS$_EXQUOTA	=	0000001C			
NALLOC		00000155	R	01	SS$_ILLIOFUNC	=	000000F4			
NOCNT		00000104	R	01	SS$_IVCHAN	=	0000013C			
NODCNT		00000170	R	01	SS$_NOPRIV	=	00000024			
NOIOSB		000000F2	R	01	SS$_NORMAL	=	00000001			
NOSECT		000001C2	R	01	SUCCES		00000181	R	01	
NOT_FILE_DEV		00000010	R	01	UCB$B_FIPL	=	0000000B			
NOT_FILE_DEVB		00000073	R	01	UCB$L_AMB	=	00000060			
NSPOOL		000000CD	R	01	UCB$L_DDT	=	00000088			
OFFLINE		0000011D	R	01	UCB$L_DEVCHAR	=	00000038			
OK		0000017A	R	01	UCB$L_IOQFL	=	0000004C			
P1	=	0000001C			UCB$L_OPCNT	=	00000070			
P2	=	00000020			UCB$L_VCB	=	00000034			
P3	=	00000024			UCB$V_BSY	=	00000008			
P4	=	00000028			UCB$V_ONLINE	=	00000004			
P5	=	0000002C			UCB$W_QLEN	=	0000006A			
P6	=	00000030			UCB$W_STS	=	00000064			
PCB$B_PRIB	=	0000002F			VCB$L_AQB	=	00000010			
PCB$L_ARB	=	0000008C			VCHAN		00000085	R	01	
PCB$L_EFCS	=	00000050			WCB$L_ORGUCB	=	00000010			

```
WCB$L_READS                     = 00000024
WCB$L_WRITES                    = 00000028
WRITE_ACCESS                      00000008 R      01
XQP                               000003D2 R      01
```

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 .	00000432 (1074.)	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	00:00:00.07	00:00:00.32
Command processing	107	00:00:00.64	00:00:03.38
Pass 1	577	00:00:24.34	00:00:53.61
Symbol table sort	0	00:00:04.13	00:00:09.31
Pass 2	195	00:00:04.76	00:00:09.87
Symbol table output	28	00:00:00.20	00:00:00.42
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	940	00:00:34.16	00:01:16.94

```
The working set limit was 1800 pages.
140721 bytes (275 pages) of virtual memory were used to buffer the intermediate code.
There were 140 pages of symbol table space allocated to hold 2622 non-local and 28 local symbols.
1049 source lines were read in Pass 1, producing 18 object records in Pass 2.
42 pages of virtual memory were used to define 41 macros.
```

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYS.OBJ]LIB.MLB;1	26
_255DUA28:[SYSLIB]STARLET.MLB;2	11
TOTALS (all libraries)	37

```
2754 GETS were required to define 37 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:SYSQIOREQ/OBJ=OBJ$:SYSQIOREQ MSRC$:SYSQIOREQ/UPDATE=(ENH$:SYSQIOREQ)+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