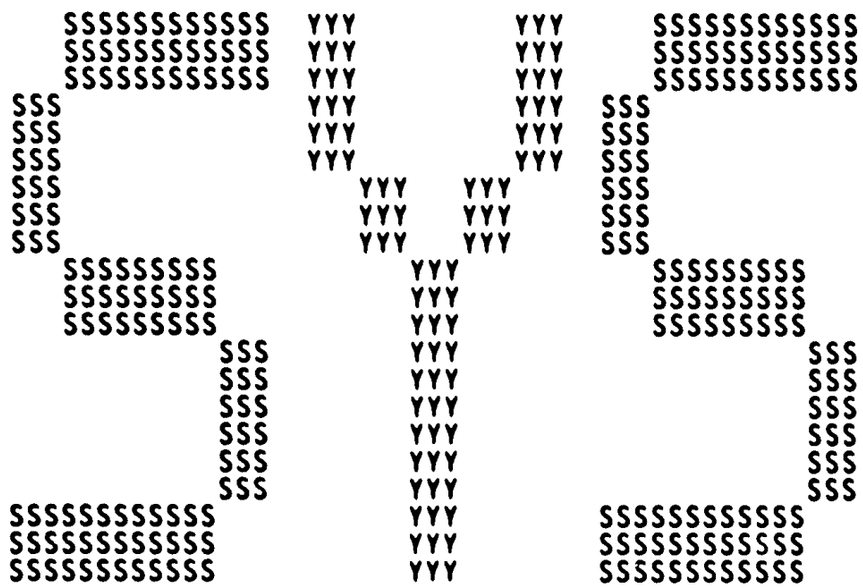

```
MM      MM  BBBBBBBB    DDDDDDDD    RRRRRRRR      IIIIII    VV      VV  EEEEEEEEEE  RRRRRRRR
MM      MM  BBBBBBBB    DDDDDDDD    RRRRRRRR      IIIIII    VV      VV  EEEEEEEEEE  RRRRRRRR
MMMM  MMMM  BB      BB  DD      DD  RR      RR      II      VV      VV  EE          RR      RR
MMMM  MMMM  BB      BB  DD      DD  RR      RR      II      VV      VV  EE          RR      RR
MM  MM  MM  BB      BB  DD      DD  RR      RR      II      VV      VV  EE          RR      RR
MM  MM  MM  BB      BB  DD      DD  RR      RR      II      VV      VV  EE          RR      RR
MM      MM  BBBBBBBB    DD      DD  RRRRRRRR        II      VV      VV  EEEEEEEE    RRRRRRRR
MM      MM  BBBBBBBB    DD      DD  RRRRRRRR        II      VV      VV  EEEEEEEE    RRRRRRRR
MM      MM  BB      BB  DD      DD  RR  RR          II      VV      VV  EE          RR  RR
MM      MM  BB      BB  DD      DD  RR  RR          II      VV      VV  EE          RR  RR
MM      MM  BB      BB  DD      DD  RR    RR        II        VV  VV    EE          RR    RR        ....
MM      MM  BB      BB  DD      DD  RR    RR        II        VV  VV    EE          RR    RR        ....
MM      MM  BBBBBBBB    DDDDDDDD    RR      RR    IIIIII        VV      EEEEEEEEEE  RR      RR      ....
MM      MM  BBBBBBBB    DDDDDDDD    RR      RR    IIIIII        VV      EEEEEEEEEE  RR      RR      ....

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

MB
V0

```
0000     1          .TITLE  MBDRIVER - VAX/VMS MAILBOX DEVICE DRIVER
0000     2          .IDENT  'V04-001'
0000     3 ;
0000     4 ;*****************************************************************************
0000     5 ;*                                                                           *
0000     6 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     7 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     8 ;*  ALL RIGHTS RESERVED.                                                     *
0000     9 ;*                                                                           *
0000    10 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000    11 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    12 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0000    13 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000    14 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0000    15 ;*  TRANSFERRED.                                                             *
0000    16 ;*                                                                           *
0000    17 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0000    18 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    19 ;*  CORPORATION.                                                             *
0000    20 ;*                                                                           *
0000    21 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0000    22 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    23 ;*                                                                           *
0000    24 ;*                                                                           *
0000    25 ;*****************************************************************************
0000    26 ;
0000    27 ;++
0000    28 ; FACILITY:
0000    29 ;
0000    30 ;       VAX/VMS MAILBOX DEVICE DRIVER
0000    31 ;
0000    32 ; ABSTRACT:
0000    33 ;
0000    34 ;       THIS MODULE CONTAINS THE MAILBOX DRIVER I/O ROUTINES.
0000    35 ;
0000    36 ; AUTHOR: R. HEINEN 16-SEPT-76
0000    37 ;
0000    38 ; MODIFIED BY:
0000    39 ;
0000    40 ;       V04-001 ACG0467         Andrew C. Goldstein,    12-Sep-1984  22:07
0000    41 ;               Fix protection holes in QIO device protection check
0000    42 ;
0000    43 ;       V03-019 LMP0289         L. Mark Pilant,         30-Jul-1984  8:39
0000    44 ;               Fix a bug introduced in LMP0265.  Read checks are still doing
0000    45 ;               the protection check on each I/O.
0000    46 ;
0000    47 ;       V03-018 LMP0265         L. Mark Pilant,         26-Jun-1984  13:52
0000    48 ;               Only do a protection check for the first I/O to the channel.
0000    49 ;
0000    50 ;       V03-017 CWH3017         CW Hobbs                8-May-1984
0000    51 ;               Use a JSB to reach IOC$CVT_DEVNAM, a BSBW just wouldn't
0000    52 ;               do it.
0000    53 ;
0000    54 ;       V03-016 CWH3016         CW Hobbs                4-May-1984
0000    55 ;               Rewrite EXE$SNDEVMSG to use IOC$CVT_DEVNAM so that
0000    56 ;               NODE$CONTROLLER form device names will be used for
0000    57 ;               cluster-wide devices.  Old routine used controller
```

```
0000     58 ;                 only - this was often ambiguous in a cluster.
0000     59 ;
0000     60 ;        V03-015 WMC0001         Wayne Cardoza           30-Apr-1984
0000     61 ;                Declare MBX resource available when message is read.
0000     62 ;
0000     63 ;        V03-014 TMK0001         Todd M. Katz            21-Apr-1984
0000     64 ;                When deleting the logical name associated with a mailbox,
0000     65 ;                delete the logical name block by calling LNM$DELETE_LNMB
0000     66 ;                instead of LNM$DELETE. Doing so will ensure that this deletion
0000     67 ;                takes place as if the system service $DELLNM had been called
0000     68 ;                to delete the logical name. In other words, not only will the
0000     69 ;                target logical name be deleted, but so will all outer access
0000     70 ;                mode aliases. Also, remove the $LOGDEF logical name definitions.
0000     71 ;
0000     72 ;        V03-013 MHB0138         Mark Bramhall           12-Apr-1984
0000     73 ;                Ensure allocated blocks are at least FKB$C_LENGTH.
0000     74 ;
0000     75 ;        V03-12  LMP0221         L. Mark Pilant,         30-Mar-1984  14:53
0000     76 ;                Change UCB$L_OWNUIC to ORB$L_OWNER and UCB$W_VPROT to
0000     77 ;                ORB$W_PROT.
0000     78 ;
0000     79 ;        V03-011 LMP0185         L. Mark Pilant,         31-Jan-1984  10:45
0000     80 ;                Track interface change to EXE$CHKxxxACCES routines.
0000     81 ;
0000     82 ;        V03-010 ROW0277         Ralph O. Weber          11-JAN-1984
0000     83 ;                Implement use of IO$M_NORSWAIT modifier to prevent resource
0000     84 ;                waits.
0000     85 ;
0000     86 ;        V03-009 DMW4032         DMWalp                  26-May-1983
0000     87 ;                Intergate new logical name structures.
0000     88 ;
0000     89 ;        V03-008 ROW0170         Ralph O. Weber          12-MAR-1983
0000     90 ;                Insert delete mailbox functionality from IOC$DELMBX in
0000     91 ;                CANCELIO.  This moves the mailbox specific knowledge of how to
0000     92 ;                delete a mailbox from $DASSGN into this driver.
0000     93 ;
0000     94 ;        V03-007 CWH1002         CW Hobbs                24-Feb-1983
0000     95 ;                Modify to return extended pid in second longword of
0000     96 ;                iosb.
0000     97 ;
0000     98 ;        V03-006 ROW49973        Ralph O. Weber          29-OCT-1982
0000     99 ;                Make all changes necessary to have control transfered to
0000    100 ;                EXE$IORSNWAIT at IPL$_ASTDEL rather than IPL$_SYNCH.  This is
0000    101 ;                necessary to conform with internal changes in EXE$IORSNWAIT.
0000    102 ;
0000    103 ;        V03-005 ROW0117         Ralph O. Weber          7-JUL-1982
0000    104 ;                Change FINISHREAD to return SS$_BUFFEROVF instead of
0000    105 ;                SS$_DATAOVERUN.  SS$_BUFFEROVF is an alternate success status.
0000    106 ;                Its use in place of SS$_DATAOVERUN will allow the buffer
0000    107 ;                overflow condition to be reported to interested programs
0000    108 ;                without hassling uninterested programs with an error status.
0000    109 ;                Also fix FINISHREAD return bytes written equal to transfer
0000    110 ;                byte count for the mailbox writer:  mailbox write operations
0000    111 ;                always transfer the requested number of bytes to the mailbox.
0000    112 ;
0000    113 ;        V03-004 KDM0002         Kathleen D. Morse       28-Jun-1982
0000    114 ;                Added $DCDEF and $PRVDEF.
```

```
0000   115 ;
0000   116 ;       V03-003 ROW0103         Ralph O. Weber          16-JUN-1982
0000   117 ;               Change FINISHREAD to return SS$_DATAOVERUN when number of
0000   118 ;               bytes in mail box message being read exceeds number of bytes
0000   119 ;               in user supplied buffer.
0000   120 ;
0000   121 ;       V03-002 ROW0102         Ralph O. Weber          14-JUN-1982
0000   122 ;               Make several changes to improve handling of zero length
0000   123 ;               messages in mailboxes.  Change READCHECKIO and WRITECHECKIO
0000   124 ;               to allow zero-byte messages, and provide a dummy buffer address
0000   125 ;               for such messages.  Change to above mentioned routines and
0000   126 ;               FDTEOF to always return buffer address and size information
0000   127 ;               in IRP$L_MEDIA and IRP$W_BCNT respectively.  Change write
0000   128 ;               function processing to always use IRP fields as source of
0000   129 ;               buffer address and size information.
0000   130 ;               This change is distributed as part of SYS.EXE ECO 14 in
0000   131 ;               Version 3.1.
0000   132 ;
0000   133 ;       V03-001 ROW0076         Ralph O. Weber          27-MAR-1982
0000   134 ;               Fix READCHECKIO and WRITECHECKIO to test length of transfer
0000   135 ;               and return immediate success if the length is zero.  The form
0000   136 ;               of this fix is duplicated from MBXDRIVER, the shared memory
0000   137 ;               mailbox driver.
0000   138 ;
0000   139 ;       V02-013 SRB0045         Steve Beckhardt         13-Jan-1981
0000   140 ;               Fixed synchronization bug involving going into MWAIT
0000   141 ;               due to insufficient pool or mailbox full.
0000   142 ;
0000   143 ;       V02-012 STJ0135         Steven T. Jeffreys,     30-Oct-1981
0000   144 ;               Use the symbol SYS$C_MBXUCBSIZ for the size of a mailbox UCB.
0000   145 ;
0000   146 ;       V02-011 STJ0025         Steven T. Jeffreys      05-Feb-1981
0000   147 ;               Modify FDTSET to default to IO$M_WRTATTN if no function
0000   148 ;               modifier is present.
0000   149 ;
0000   150 ;       V02-010 STJ0018         Steven T. Jeffreys      28-Jan-1981
0000   151 ;               Modified FDTSET to support SETPROT function.  Changed
0000   152 ;               EXE$WRTMAILBOX path to check for system write access
0000   153 ;               allowed and return SS$_INSFMEM if nonpaged pool
0000   154 ;               cannot be allocated.
0000   155 ;
0000   156 ;       V2-009  LMK0002         Len Kawell              09-Jun-1980
0000   157 ;               Allow zero length messages.
0000   158 ;
0000   159 ;       V0008   LMK0001         Len Kawell              8-Feb-1980
0000   160 ;               Remove maximum number of messages checks (UCB$W_MSGMAX)
0000   161 ;               since there is no way to set it on a per mailbox basis
0000   162 ;               and the buffer quota is sufficient protection.
0000   163 ;
0000   164 ;               Also, return SS$_INSFMEM and SS$_MBFULL when resource wait is
0000   165 ;               disabled.
0000   166 ;
0000   167 ;       V0007   ACG0047         Andrew C. Goldstein,    8-Aug-1979  17:10
0000   168 ;               Protection check interface changes
0000   169 ;
0000   170 ;--
0000   171 ; EXTERNAL SYMBOLS
```

```
                     0000   172 ;
                     0000   173         $ACBDEF                         ; DEFINE AST CONTROL BLOCK
                     0000   174         $ARMDEF                         ; DEFINE ACCESS BIT VALUES
                     0000   175         $CADEF                          ; DEFINE CONDITIONAL ASSEMBLY
                     0000   176         $CANDEF                         ; CANCEL REASON CODES
                     0000   177         $CCBDEF                         ; DEFINE CHANNEL CONTROL BLOCK OFFSETS
                     0000   178         $DCDEF                          ; DEFINE DEVICE TYPES
                     0000   179         $DDBDEF                         ; DEFINE DDB
                     0000   180         $DYNDEF                         ; DEFINE DYNAMIC BLOCK TYPES
                     0000   181         $FKBDEF                         ; DEFINE FORK BLOCK
                     0000   182         $IODEF                          ; DEFINE FUNCTION CODES
                     0000   183         $IRPDEF                         ; DEFINE I/O PACKET OFFSETS
                     0000   184         $IPLDEF                         ; DEFINE IPL NUMBERS
                     0000   185         $ORBDEF                         ; DEFINE OBJECT'S RIGHTS BLOCK
                     0000   186         $PCBDEF                         ; DEFINE PCB OFFSETS
                     0000   187         $PRDEF                          ; DEFINE PROCESSOR REGISTERS
                     0000   188         $PRIDEF                         ; DEFINE PRIORITY INCREMENTS
                     0000   189         $PRVDEF                         ; DEFINE PRIVILEGES
                     0000   190         $RSNDEF                         ; DEFINE RESOURCE NUMBERS
                     0000   191         $SSDEF                          ; DEFINE SYSTEM STATUS CODES
                     0000   192         $UCBDEF                         ; DEFINE UCB OFFSETS
                     0000   193
                     0000   194 ;
                     0000   195 ; LOCAL DEFINITIONS
                     0000   196 ;
00000000             0000   197 UCB$L_MB_MSGQ   = UCB$L_FQFL            ; MAILBOX MESSAGE QUEUE LISTHEAD
0000000C             0000   198 UCB$L_MB_W_AST  = UCB$L_ASTQFL          ; MAILBOX WRITE ATTN AST LIST
00000010             0000   199 UCB$L_MB_R_AST  = UCB$L_ASTQBL          ; MAILBOX READ ATTN AST LIST
                     0000   200
                     0000   201 ;+
                     0000   202 ; THE DEFINITIONS BELOW AREN'T ACTUALLY USED BY THE CODE, BUT DO
                     0000   203 ; ACCURATELY REFLECT THE STRUCTURE OF THE BLOCKS IN THE MAILBOX
                     0000   204 ; MESSAGE QUEUE.
                     0000   205 ;
                     0000   206 ; SOMEDAY, THESE DEFINITIONS SHOULD BE CHANGED TO MAKE THE IRP
                     0000   207 ; ADDRESS AND SENDER'S PID LONGWORD ALIGNED AND THE CODE SHOULD
                     0000   208 ; USE THESE DEFINITIONS!
                     0000   209 ;-
                     0000   210
                     0000   211         .PSECT  $ABS$, ABS
                     0000   212
00000000             0000   213         . = 0
                     0000   214
00000004             0000   215 MBMSG_L_FLINK:   .BLKL                  ; FORWARD LINK
00000008             0004   216 MBMSG_L_BLINK:   .BLKL                  ; BACKWARD LINK
0000000A             0008   217 MBMSG_W_SIZE:    .BLKW                  ; BLOCK SIZE
0000000B             000A   218 MBMSG_B_TYPE:    .BLKB                  ; BLOCK TYPE
0000000C             000B   219 MBMSG_B_FUNC:    .BLKB                  ; FUNCTION CODE
0000000E             000C   220 MBMSG_W_DATSIZ: .BLKW                   ; MESSAGE DATA SIZE
00000012             000E   221 MBMSG_L_IRP:     .BLKL                  ; IRP ADDRESS
00000016             0012   222 MBMSG_L_PID:     .BLKL                  ; SENDER'S PID
                     0016   223 MBMSG_C_LENGTH:                         ; LENGTH OF FIXED FORMAT HEADER
                     0016   224 MBMSG_T_DATA:                           ; START OF MESSAGE DATA
                     0016   225
                     0016   226 ;+
                     0016   227 ; THE FIXED FORMAT HEADER LENGTH MUST BE AT LEAST FKB$C_LENGTH.
                     0016   228 ; THIS IS BECAUSE COM$DRVDEALMEM WANTS TO BE ABLE TO TURN THE
```

```
                     0016   229 ; BLOCK INTO A FORK BLOCK FOR DELAYED DEALLOCATION, EVEN FOR
                     0016   230 ; ZERO LENGTH MESSAGE DATA BLOCKS.
                     0016   231 ;-
                     0016   232
00000016             0016   233 MBMSG_C_HEADER  = MBMSG_C_LENGTH          ; ASSUME FIXED HEADER IS LONG ENOUGH
FFFFFFFE             0016   234 .IF LT MBMSG_C_HEADER-FKB$C_LENGTH
00000018             0016   235 MBMSG_C_HEADER  = FKB$C_LENGTH            ; LENGTHEN TO AT LEAST A FORK BLOCK
                     0016   236 .ENDC
                     0016   237
00000000             0016   238 P1              = 0                       ; OFFSET TO BUFFER ADDRESS IN ARGUMENT BLOCK
00000004             0016   239 P2              = 4                       ; OFFSET TO REQUEST SIZE IN ARGUMENT BLOCK
00000008             0016   240 P3              = 8                       ; OFFSET FOR PARAMETER 3
0000000C             0016   241 P4              = 12                      ; OFFSET FOR PARAMETER 4
                     0016   242
                     0016   243 ;
                     0016   244 ; LOCAL DATA STORAGE
                     0016   245 ;
                 00000000   246         .PSECT  $$$105_PROLOGUE
                     0000   247 MB$DPT::                                  ; DRIVER START
                     0000   248         DPTAB   -                         ; DRIVER PROLOGUE TABLE
                     0000   249                 END=MB_END,-              ; END OF DRIVER
                     0000   250                 ADAPTER=UBA,-             ; FAKE ADAPTER
                     0000   251                 UCBSIZE=SYS$C_MBXUCBSIZ,- ; SIZE OF UCB
                     0000   252                 NAME=MBDRIVER             ; DRIVER NAME
                     0038   253         DPT_STORE INIT                    ;
                     0038   254         DPT_STORE REINIT                  ;
                     0038   255         DPT_STORE END                     ; START AND END OF CONTROLLER INIT
                     0000   256
                 00000000   257         .PSECT  $$$115_DRIVER,LONG
                     0000   258
                     0000   259         DDTAB   MB,-                      ; DRIVER DISPATCH TABLE
                     0000   260                 STARTIO,-                 ; STARTIO OPERATION
                     0000   261                 0,-                       ; NO UNSOLICITED INTERRUTPS
                     0000   262                 FUNCTABLE,-               ; FUNCTION DECISION TABLE
                     0000   263                 CANCELIO,-                ; CANCEL I/O
                     0000   264                 0,-                       ; REGISTER DUMP ROUTINE
                     0000   265                 0,-                       ; SIZE OF DIAGNOSTIC BUFFER
                     0000   266                 0                         ; SIZE OF ERROR LOG BUFFER
                     0038   267
                     0038   268 ;
                     0038   269 ; FUNCTION DECISION TABLE
                     0038   270 ;
                     0038   271
                     0038   272 FUNCTABLE:                                ; FUNCTION DECISION TABLE
                     0038   273         FUNCTAB ,<-                       ; LEGAL FUNCTIONS
                     0038   274                 SETMODE,-                 ; ASK FOR READ OR WRITE AST'S
                     0038   275                 WRITEOF,-                 ; WRITE EOF
                     0038   276                 READLBLK,WRITELBLK,-
                     0038   277                 READVBLK,WRITEVBLK,-
                     0038   278                 READPBLK,WRITEPBLK>
                     0040   279         FUNCTAB,<READLBLK,READVBLK,READPBLK,-
                     0040   280                 WRITELBLK,WRITEVBLK,WRITEPBLK>
                     0048   281         FUNCTAB FDTREAD,<READLBLK,READPBLK,READVBLK>; READ FUNCTION
                     0054   282         FUNCTAB FDTWRITE,<WRITELBLK,WRITEPBLK,WRITEVBLK>; WRITE FUNCTION
                     0060   283         FUNCTAB FDTSET,<SETMODE>          ; SET AST CONTROL
                     006C   284         FUNCTAB FDTEOF,<WRITEOF>          ; WRITE EOF
```

```
                     0078   286          .SBTTL  EXE$SNDEVMSG - SEND DEVICE DRIVER MAILBOX MESSAGE
                     0078   287 ;++
                     0078   288 ; EXE$SNDEVMSG - SEND DEVICE SPECIFIC MESSAGE ON BEHALF OF DRIVER
                     0078   289 ;
                     0078   290 ; FUNCTIONAL DESCRIPTION:
                     0078   291 ;
                     0078   292 ; THIS ROUTINE BUILDS AND SENDS A DEVICE SPECIFIC MESSAGE TO A MAILBOX.
                     0078   293 ; THE MESSAGE IS FORMATTED AS FOLLOWS:
                     0078   294 ;
                     0078   295 ;       WORD 0    = TYPE OF MESSAGE
                     0078   296 ;       WORD 1    = UNIT OF DEVICE
                     0078   297 ;       REMAINDER = COUNTED STRING OF DEVICE CONTROLLER NAME - FORMATTED AS
                     0078   298 ;                   NODE$CONTROLLER FOR CLUSTER-WIDE DEVICES
                     0078   299 ;
                     0078   300 ; INPUTS:
                     0078   301 ;
                     0078   302 ;       R3 = MAILBOX UCB ADDRESS
                     0078   303 ;       R4 = TYPE OF MESSAGE
                     0078   304 ;       R5 = DEVICE UCB ADDRESS
                     0078   305 ;
                     0078   306 ;
                     0078   307 ; OUTPUTS:
                     0078   308 ;
                     0078   309 ;       R0 = STATUS OF THE OPERATION
                     0078   310 ;       R1,R2,R3,R4 ARE DESTROYED
                     0078   311 ;
                     0078   312 ; STATUS RETURNS:
                     0078   313 ;
                     0078   314 ;       SEE EXE$WRTMAILBOX.
                     0078   315 ;--
                     0078   316
                     0078   317
                     0078   318 EXE$SNDEVMSG::                          ; SEND MESSAGE FOR DEVICE DRIVER
                     0078   319
                     0078   320 ;
                     0078   321 ; SET THE PROPER IPL FOR INTERLOCK, WRTMAILBOX WAS GOING TO DO THIS ANYWAY
                     0078   322 ; AND IOC$CVT_DEVNAM ASSUMES WE HAVE THE IO DATABASE LOCKED FOR READING
                     0078   323 ;
          7E    12   DB  0078   324          MFPR    #PR$_IPL,-(SP)          ; SAVE CURRENT IPL
          6E    0B   91  007B   325          CMPB    #IPL$_MAILBOX,(SP)      ; HIGH ENOUGH?
                03   1B  007E   326          BLEQU   10$                     ; IF LEQU THEN YES
                     0080   327          SETIPL  #IPL$_MAILBOX           ; SET THE PROPER IPL
                     0083   328 ;
                     0083   329 ; MAKE A SPACE FOR THE NAME BUFFER, SET MESSAGE CODE AND DEVICE UNIT NUMBER
                     0083   330 ;       (note that we are allocating a buffer which may have to be
                     0083   331 ;       extended when device names are allowed to be 64 bytes long,
                     0083   332 ;       in the short term it saves space on the interrupt stack)
                     0083   333 ;
          7E    55   7D  0083   334 10$:     MOVQ    R5,-(SP)                ; SAVE DEVICE UCB, GET R6 AS SCRATCH REG
          5E    1C   C2  0086   335          SUBL2   #28,SP                  ; RESERVE 28 BYTES SPACE TO BUILD MESSAGE
                54   DD  0089   336          PUSHL   R4                      ; MOVE MESSAGE CODE TO BUFFER, NOW 32 BYTES
          56    53   D0  008B   337          MOVL    R3,R6                   ; SAVE THE INPUT MAILBOX UCB ADDRESS
    02 AE    54   A5   B0  008E   338          MOVW    UCB$W_UNIT(R5),2(SP)    ; INSERT UNIT NUMBER OF DEVICE IN MESSAGE
                     0093   339 ;
                     0093   340 ; CALL IOC$CVT_DEVNAM TO PLACE THE NODE$CONTROLLER NAME IN THE BUFFER, CHECK
                     0093   341 ; STATUS AND STORE ASCIC COUNT BYTE IN THE MESSAGE (ON TOP OF THE LEADING "_")
                     0093   342 ;
```

```
                50    1C    D0  0093    343          MOVL    #28,R0                  ; 28 BYTES OF MESSAGE BUFFER AVAILABLE
              51  04    AE    9E  0096    344          MOVAB   4(SP),R1                ; ADDRESS OF AVAILABLE BUFFER FOR DEVICE NAM
                54    04    D0  009A    345          MOVL    #4,R4                   ; WANT NODE$CONTROLLER ONLY, NO UNIT#
                                  009D    346                                      ; DEVICE UCB IS ALREADY IN R5
          00000000'GF         16  009D    347          JSB     G^IOC$CVT_DEVNAM        ; LET THE COMMON ROUTINE FIGURE IT OUT
                50    01    B1  00A3    348          CMPW    S^#SS$_NORMAL,R0        ; DID IT WORK? (SS$_BUFFEROVF IS ERROR)
                      04    13  00A6    349          BEQL    30$                     ; YES, IT RETURNED SUCCESS
                                  00A8    350 20$:     BUG_CHECK INCONSTATE            ; FAILURE SHOULD NOT BE POSSIBLE
                      51    D7  00AC    351 30$:     DECL    R1                      ; REMOVE THE LEADING UNDERSCORE FROM LEN
                      F8    15  00AE    352          BLEQ    20$                     ; ZERO LENGTH IS ALSO SERIOUS ERROR
              04 AE   51    90  00B0    353          MOVB    R1,4(SP)                ; STORE THE ASCIC COUNT IN THE MESSAGE
                                  00B4    354 ;
                                  00B4    355 ; GET R3=MESSAGE LENGTH, R4=MESSAGE ADDRESS, R5=MAILBOX UCB ADDRESS AND THEN WRITE
                                  00B4    356 ;
            53    51    05    C1  00B4    357          ADDL3   #<2+2+1>,R1,R3          ; CODE + UNIT# + COUNT + NAME
                54    5E    D0  00B8    358          MOVL    SP,R4                   ; GET MESSAGE ADDRESS TO R4
                55    56    D0  00BB    359          MOVL    R6,R5                   ; MAILBOX UCB ADDRESS TO R5
                      0A    10  00BE    360          BSBB    EXE$WRTMAILBOX          ; DO THE MAILBOX WRITE
                                  00C0    361 ;
                                  00C0    362 ; CLEAN UP THE STACK, REGISTERS, RESTORE IPL AND EXIT
                                  00C0    363 ;
                5E    20    C0  00C0    364          ADDL    #32,SP                  ; REMOVE MESSAGE BUFFER FROM STACK
                55    8E    7D  00C3    365          MOVQ    (SP)+,R5                ; RESTORE SENDING UCB AND SCRATCH REGISTER
                                  00C6    366          ENBINT                          ; ENABLE INTERUPTS TO CALLER'S IPL
                            05  00C9    367          RSB                             ; RETURN
```

```
                          00CA         .SBTTL  SYSTEM INTERNAL WRITE TO MAILBOX SUBROUTINE
                          00CA   ;++
                          00CA   ; EXE$WRTMAILBOX - WRITE TO MAILBOX SUBROUTINE FOR EXECUTIVE USE
                          00CA   ;
                          00CA   ; FUNCTIONAL DESCRIPTION:
                          00CA   ;
                          00CA   ; THIS ROUTINE IS USED BY SYSTEM ROUTINES TO WRITE A MESSAGE TO A MAILBOX.
                          00CA   ;
                          00CA   ; INPUTS:
                          00CA   ;
                          00CA   ;       R3 = SIZE OF MESSAGE
                          00CA   ;       R4 = MESSAGE ADDRESS
                          00CA   ;       R5 = MAILBOX UCB ADDRESS
                          00CA   ;
                          00CA   ; OUTPUTS:
                          00CA   ;
                          00CA   ;       R0 = STATUS OF OPERATION
                          00CA   ;
                          00CA   ;       R1,R2 USED.
                          00CA   ;
                          00CA   ; COMPLETION CODES:
                          00CA   ;
                          00CA   ;       SS$_NORMAL
                          00CA   ;       SS$_MBTOOSML - MESSAGE TOO LARGE FOR MAILBOX
                          00CA   ;       SS$_MBFULL - MAILBOX FULL OF MESSAGES
                          00CA   ;       SS$_INSFMEM - MEMORY ALLOCATION PROBLEM
                          00CA   ;       SS$_NOPRIV - NO OWNER WRITE ACCESS
                          00CA   ;
                          00CA   ;--
                          00CA   EXE$WRTMAILBOX::                                ;
                          00CA   ;
                          00CA   ; SET THE PROPER IPL FOR INTERLOCK
                          00CA   ;
          7E    12    DB  00CA          MFPR    #PR$_IPL,-(SP)          ; SAVE CURRENT IPL
          6E    0B    91  00CD          CMPB    #IPL$_MAILBOX,(SP)      ; HIGH ENOUGH?
                03    1B  00D0          BLEQU   10$                     ; IF LEQU THEN YES
                          00D2          SETIPL  #IPL$_MAILBOX           ; SET THE PROPER IPL
                          00D5   ;
                          00D5   ; MAIL THE MESSAGE
                          00D5   ;
       50   08D8 8F   3C  00D5   10$:   MOVZWL  #SS$_MBFULL,R0          ; ASSUME MESSAGE WILL NOT FIT
          18 A5  53   B1  00DA          CMPW    R3,UCB$W_BUFQUO(R5)     ; MESSAGE FIT?
                 6B   1A  00DE          BGTRU   40$                     ; IF GTRU THEN NO
       50   019C 8F   3C  00E0          MOVZWL  #SS$_MBTOOSML,R0        ; ASSUME MESSAGE TOO BIG
          42 A5  53   B1  00E5          CMPW    R3,UCB$W_DEVBUFSIZ(R5)  ; BIGGER THAN ALLOWED?
                  D   1A  00E9          BGTRU   40$                     ; IF YES THEN ALSO ERROR
          50     24   3C  00EB          MOVZWL  #SS$_NOPRIV,R0          ; ASSUME NO WRITE PRIVILEGE
       51     1C A5   D0  00EE          MOVL    UCB$L_ORB(R5),R1        ; GET ORB ADDRESS
                          00F2
                          00F2   ; THE FOLLOWING ASSUMES THAT THE OWNER PROTECTION FIELD IS IN BITS 4-7 OF THE
                          00F2   ; STANDARD PROTECTION WORD.
                          00F2
 7E     18 A1    04   04  EF  00F2          EXTZV   #4,#4,ORB$W_PROT(R1),-(SP)      ; ASSUME SOGW PROTECTION WORD
        04 0B A1      00  E0  00F8          BBS     #ORB$V_PROT_16,ORB$B_FLAGS(R1),15$      ; XFER IF CORRECT
          6E     1C A1    D0  00FD          MOVL    ORB$L_OWN_PROT(R1),(SP) ; ELSE USE VECTOR
             8E     02    D3  0101   15$:   BITL    #ARM$M_WRITE,(SP)+      ; CHECK FOR WRITE ACCESS
                    42    12  0104          BNEQ    40$                     ; XFER IF NO WRITE ACCESS
```

```
                51   53    18   C1  0106   426          ADDL3   #MBMSG_C_HEADER,R3,R1   ; COMPUTE SIZE OF MESSAGE BLOCK
                           38   BB  010A   427          PUSHR   #^M<R3,R4,R5>           ; SAVE REGISTERS FROM MOVC
                 00000000'GF    16  010C   428          JSB     G^EXE$ALONONPAGED       ; GET THE MEMORY BLOCK
                       07 50    E8  0112   429          BLBS    R0,20$                  ; BRANCH IS SUCCESS
                50   0124 8F    3C  0115   430          MOVZWL  #SS$_INSFMEM,R0         ; SET CORRECT ERROR STATUS
                           2A    11  011A   431          BRB     30$                     ; RETURN ERROR STATUS
                                     011C   432 ;
                                     011C   433 ; FILL IN BLOCK
                                     011C   434 ;
                           82    7F  011C   435 20$:     PUSHAQ  (R2)+                   ; SAVE BLOCK ADDRESS AND PASS LINK WORDS
                      82   51    B0  011E   436          MOVW    R1,(R2)+                ; INSERT BLOCK SIZE
                      82   13    B0  0121   437          MOVW    #DYN$C_BUFIO,(R2)+      ; INSERT BLOCK TYPE AND ZERO FUNCTION
                 82   04 AE    B0  0124   438          MOVW    4(SP),(R2)+             ; INSERT SIZE OF MESSAGE
                           82    D4  0128   439          CLRL    (R2)+                   ; SET NO PACKET
             50  00000000'GF    D0  012A   440          MOVL    G^SCH$GL_CURPCB,R0      ; GET CURRENT PCB
                 82   60 A0    D0  0131   441          MOVL    PCB$L_PID(R0),(R2)+     ; INSERT IT ( WHATEVER IT IS! )
                                     0135   442 ;
                                     0135   443 ; COPY DATA
                                     0135   444 ;
          62   64   04 AE    28  0135   445          MOVC3   4(SP),(R4),(R2)         ; MOVE DATA
                                     013A   446 ;
                                     013A   447 ; INSERT IN MESSAGE QUEUE
                                     013A   448 ;
                      04    BA  013A   449          POPR    #^M<R2>                 ; RESTORE BLOCK ADDRESS
                55   08 AE    D0  013C   450          MOVL    8(SP),R5                ; RESTORE MAILBOX UCB ADDRESS
                       02AD    30  0140   451          BSBW    INSMBQUEUE              ; INSERT ON QUEUE
                   50   01    9A  0143   452          MOVZBL  #SS$_NORMAL,R0          ; SET SUCCESS
                           38    BA  0146   453 30$:     POPR    #^M<R3,R4,R5>           ; RESTORE REGISTERS
                                     0148   454 40$:     ENBINT                          ; ENABLE INTERUPTS TO CALLER'S IPL
                                     014B   455          RSB                             ; AND RETURN
```

```
                     014C   457          .SBTTL  CANCELIO - CANCEL I/O ON MAILBOX UNIT
                     014C   458 ;++
                     014C   459 ; CANCELIO - CANCEL I/O ON MAILBOX UNIT
                     014C   460 ;
                     014C   461 ; FUNCTIONAL DESCRIPTION:
                     014C   462 ;
                     014C   463 ; THIS ROUTINE IS ENTERED TO CANCEL ALL OUTSTANDING I/O FOR A PARTICULAR
                     014C   464 ; PROCESS AND CHANNEL ON A MAILBOX UNIT.
                     014C   465 ; IF THE UNIT IS BUSY, THE CURRENT READ PACKET IS CHECKED AND COMPLETED
                     014C   466 ; IF IT BELONGS TO THE CANCELLING PROCESS. ALL QUEUED REQUESTS HAVE BEEN REMOVED.
                     014C   467 ; IF NO READER EXISTS THEN THE QUEUE OF OUTSTANDING MESSAGES IS SEARCHED
                     014C   468 ; FOR MESSAGES AND WAITING WRITES. IF A PID MATCH EXISTS THEN THESE I/O
                     014C   469 ; ARE ALSO COMPLETED ALONG WITH REMOVING THE MESSAGES.
                     014C   470 ; THE FINAL ACTION IS TO SEARCH THE QUEUE OF AST REQUESTS TO REMOVE THE ONES
                     014C   471 ; ASSOCIATED WITH THE CANCELLING PROCESS.
                     014C   472 ;
                     014C   473 ; INPUTS:
                     014C   474 ;
                     014C   475 ;       R2 = NEGITIVE OF CHANNEL NUMBER
                     014C   476 ;       R3 = CURRENT PACKET ADDRESS
                     014C   477 ;       R4 = PCB OF CANCELLING PROCESS
                     014C   478 ;       R5 = UCB OF UNIT
                     014C   479 ;       R8 = CANCEL REASON CODE (CAN$C_CANCEL, CAN$C_DASSGN, or CAN$C_AMBXDGN)
                     014C   480 ;
                     014C   481 ; OUTPUTS:
                     014C   482 ;
                     014C   483 ;       R4,R5 ARE PRESERVED
                     014C   484 ;
                     014C   485 ; IPL = MAILBOX IPL
                     014C   486 ;--
                     014C   487 CANCELIO:                                    ; CANCEL I/O ON MAILBOX UNIT
        58   02   D1  014C   488          CMPL    #CAN$C_AMBXDGN, R8        ; Branch if this is an associated
             24   13  014F   489          BEQL    3$                        ; mailbox last ref. deassign.
        00F0 8F   BB  0151   490          PUSHR   #^M<R4,R5,R6,R7>          ; SAVE R4-R7
        56   52   D0  0155   491          MOVL    R2,R6                     ; COPY CHANNEL NUMBER
1A 64   A5   08   E1  0158   492          BBC     #UCB$V_BSY,UCB$W_STS(R5),10$; BUSY?
0C A3     60 A4   D1  015D   493          CMPL    PCB$L_PID(R4),IRP$L_PID(R3); PIDS MATCH?
             56   12  0162   494          BNEQ    40$                       ; IF NO THEN CANCEL DONE
   28 A3     56   B1  0164   495          CMPW    R6,IRP$W_CHAN(R3)         ; CHANNEL MATCH?
             50   12  0168   496          BNEQ    40$                       ; IF NEQ THEN NO
        50   2C   7D  016A   497          MOVQ    #SS$_ABORT,R0             ; SET STATUS FOR ABORT
  00000000'GF   16   16D   498          JSB     G^IOC$REQCOM              ; COMPLETE THE REQUEST
             45   11  0173   499          BRB     40$                       ; AND CANCEL IS DONE
             74   11  0175   500 3$:      BRB     100$                      ; Assoc. MBX deassign br. assist.
                      0177   501 ;
                      0177   502 ; NO READER WAITING - CHECK MESSAGE QUEUE
                      0177   503 ;
        52   65   9E  0177   504 10$:     MOVAB   UCB$L_MB_MSGQ(R5),R2      ; ADDRESS MESSAGE QUEUE
        50   52   D0  017A   505          MOVL    R2,R0                     ; COPY LIST HEAD ADDRESS
        52   62   D0  017D   506 20$:     MOVL    (R2),R2                   ; ADDRESS LIST ENTRY
        52   50   D1  0180   507          CMPL    R0,R2                     ; END OF LIST?
             35   13  0183   508          BEQL    40$                       ; IF YES THEN DONE
12 A2     60 A4   D1  0185   509          CMPL    PCB$L_PID(R4),18(R2)      ; MESSAGE BELONG TO CANCELLING PROCESS?
             F1   12  018A   510          BNEQ    20$                       ; IF NO THEN SEARCH MORE
   53   0E A2   D0  018C   511          MOVL    14(R2),R3                 ; ADDRESS PACKET IF ANY
             EB   13  0190   512          BEQL    20$                       ; IF EQL THEN NO ASSOC PACKET
   28 A3     56   B1  0192   513          CMPW    R6,IRP$W_CHAN(R3)         ; CHANNEL MATCH?
```

```
                E5   12  0196   514          BNEQ    20$                          ; IF NEQ THEN NO
                     0198   515                                                   ; BUT GET RID OF MESSAGE
          38 A3  2C  7D  0198   516          MOVQ    #SS$_ABORT,IRP$L_MEDIA(R3); SET STATUS
      00000000'GF  16  019C   517          JSB     G^COM$POST                   ; COMPLETE THE OPERATION
           50  62  0F  01A2   518 30$:      REMQUE  (R2),R0                      ; REMOVE MESSAGE FROM QUEUE
    18 A5  0C A2  A0  01A5   519          ADDW    12(R2),UCB$W_BUFQUO(R5)      ; ADJUST QUOTA
          16 A5  B7  01AA   520          DECW    UCB$W_MSGCNT(R5)             ; AND MESSAGE COUNT
    44 A5  16 A5  B0  01AD   521          MOVW    UCB$W_MSGCNT(R5),UCB$L_DEVDEPEND(R5); SAVE IT
      00000000'GF  16  01B2   522          JSB     G^COM$DRVDEALMEM             ; DEALLOCATE MESSAGE
                BD  11  01B8   523          BRB     10$                          ; SEARCH LIST FROM THE START
                     01BA   524 ;
                     01BA   525 ; SEARCH AST QUEUE
                     01BA   526 ;
           57  0C A5  9E  01BA   527 40$:      MOVAB   UCB$L_MB_W_AST(R5),R7        ; ADDRESS LIST OF AST'S
               54  6E  D0  01BE   528          MOVL    (SP),R4                      ; GET CANCEL PCB
      00000000'GF  16  01C1   529          JSB     G^COM$FLUSHATTNS             ; FLUSH ATTENTION AST'S
           57  10 A5  9E  01C7   530          MOVAB   UCB$L_MB_R_AST(R5),R7        ; ADDRESS WRITER AST'S
      00000000'GF  16  01CB   531          JSB     G^COM$FLUSHATTNS             ; FLUSH THAT LIST
               00F0 8F  BA  01D1   532          POPR    #^M<R4,R5,R6,R7>             ; RESTORE REGISTERS R4-R7
                     01D5   533          SETIPL  #IPL$_SYNCH                  ; LOWER IPL
               50  02  9A  01D8   534          MOVZBL  #RSN$_MAILBOX,R0             ; DECLARE RESOURCE AVAILABLE
      00000000'GF  16  01DB   535          JSB     G^SCH$RAVAIL
                     01E1   536 ;
                     01E1   537 ; CHECK FOR LAST CHANNEL DEASSIGN
                     01E1   538 ;
               58  01  D1  01E1   539          CMPL    #CAN$C_DASSGN, R8            ; Deassigning channel?
                   3E  12  01E4   540          BNEQ    900$                         ; Branch if not channel deassign.
                 5C A5  B5  01E6   541          TSTW    UCB$W_REFC(R5)               ; Is reference count zero?
                   39  12  01E9   542          BNEQ    900$                         ; Branch if ref. count not zero.
    34 68 A5       01  E1  01EB   543 100$:     BBC     #UCB$V_DELMBX, -             ; Branch if mailbox is not
                           01F0   544                  UCB$W_DEVSTS(R5), 900$       ; to be deleted.
                           01F0   545          SETIPL  #IPL$_ASTDEL                 ; Lower IPL
                    74 A5  D5  01F3   546          TSTL    UCB$L_LOGADR(R5)             ; Test address of logical name entry.
                       16  13  01F6   547          BEQL    120$                         ; Branch if none.
              00000000'GF  16  01F8   548          JSB     G^LNM$LOCKW                  ; Lock name table for write.
               51  74 A5  D0  01FE   549          MOVL    UCB$L_LOGADR(R5), R1         ; Get address of logical name entry.
              00000000'GF  16  0202   550          JSB     G^LNM$DELETE_LNMB            ; Delete logical name block.
              00000000'GF  16  0208   551          JSB     G^LNM$UNLOCK                 ; Unlock name table.
                           020E   552 ;
                           020E   553 ;       Clean up mailbox messages
                           020E   554 ;       there are no outstanding I/O operations.
                           020E   555 ;
                50  00 B5  0F  020E   556 120$:     REMQUE  @UCB$L_MB_MSGQ(R5), R0       ; Fetch message block.
                       08  1D  0212   557          BVS     150$                         ; Branch if none.
              00000000'GF  16  0214   558          JSB     G^EXE$DEANONPAGED            ; Deallocate message block.
                       F2  11  021A   559          BRB     120$                         ; Repeat until exhausted.
    64 A5       00010000 8F  C8  021C   560 150$:     BISL    #UCB$M_DELETEUCB, -          ; Mark UCB for deletion, DASSGN
                           0224   561                  UCB$L_STS(R5)                ; will do the rest including crediting
                           0224   562                                               ; quotas for temporary mailboxes.
                           0224   563
                           05  0224   564 900$:     RSB                          ; Return to caller.
```

```
                     0225   566          .SBTTL  CHECKIO - CHECK READ AND WRITE PARAMETERS
                     0225   567 ;++
                     0225   568 ; READCHECKIO - CHECK READ PARAMETERS
                     0225   569 ; WRITECHECKIO - CHECK WRITE PARAMETERS
                     0225   570 ;
                     0225   571 ; FUNCTIONAL DESCRIPTION:
                     0225   572 ;
                     0225   573 ; THIS ROUTINE IS USED BY THE READ AND WRITE FDT ROUTINES TO VALIDATE THE
                     0225   574 ; I/O REQUEST. THE CHECKS ARE MADE BASED ON THE SETTING OF THE IRP$V_FUNC
                     0225   575 ; OPERATION DIRECTION BIT. THE CHECKS ARE, 1) ACCESS TO UNIT BY UIC,
                     0225   576 ; 2) MESSAGE REQUEST SIZE WITHIN MAX MESSAGE SIZE, 3) BUFFER ACCESSIBLE.
                     0225   577 ;
                     0225   578 ; ACCESS VIOLATIONS CAUSE COMPLETIONS HERE.
                     0225   579 ;
                     0225   580 ; INPUTS:
                     0225   581 ;
                     0225   582 ;       R3 = PACKET ADDRESS
                     0225   583 ;       R4 = PCB ADDRESS
                     0225   584 ;       R5 = UCB ADDRESS
                     0225   585 ;       R6 = CCB ADDRESS
                     0225   586 ;       R7 = FUNCTION CODE
                     0225   587 ;       R9 = (SCRATCH)
                     0225   588 ;       AP = ADDRESS OF THE FIRST QIO PARAMETER
                     0225   589 ;
                     0225   590 ; OUTPUTS:
                     0225   591 ;
                     0225   592 ;       R3 = PACKET ADDRESS
                     0225   593 ;       R4 = PCB ADDRESS
                     0225   594 ;       R5 = UCB ADDRESS
                     0225   595 ;
                     0225   596 ;       IRP$L_MEDIA(R3) = BUFFER ADDRESS
                     0225   597 ;       IRP$L_BCNT(R3) = BUFFER SIZE (low order only)
                     0225   598 ;
                     0225   599 ;--
                     0225   600          .ENABL  LSB
                     0225   601 READCHECKIO:                                ; CHECK READ PARAMETERS
    00000000'GF   9F 0225   602          PUSHAB  G^EXE$READCHK              ; READ CHECKS NEEDED
59  00000000'GF   9E 022B   603          MOVAB   G^EXE$CHKRDACCES,R9        ;
       5A   02   D0 0232   604          MOVL    #CCB$V_RDCHKDON,R10        ;
            10   11 0235   605          BRB     10$                        ; CONTINUE IN COMMON
                     0237   606 WRITECHECKIO:                               ; SET UP FOR WRITE CHECK
59  00000000'GF   9E 0237   607          MOVAB   G^EXE$CHKWRTACCES,R9       ; SET UP WRITE CHECK
       5A   03   D0 023E   608          MOVL    #CCB$V_WRTCHKDON,R10       ;
    00000000'GF   9F 0241   609          PUSHAB  G^EXE$WRITECHK             ;
0A 08 A6    5A   E0 0247   610 10$:     BBS     R10,CCB$B_STS(R6),20$      ; SKIP CHECK IF ALREADY DONE
                     024C   611                                             ; R4 = PCB ADDRESS
                     024C   612                                             ; R5 = UCB ADDRESS
            69   16 024C   613          JSB     (R9)                       ; CHECK UIC ACCESS
          1E 50   E9 024E   614          BLBC    R0,ERROR                   ; BR IF ACCESS FAILURE
00 08 A6    5A   E2 0251   615          BBSS    R10,CCB$B_STS(R6),20$      ; MARK PROT CHECK DONE
   51    04 AC   3C 0256   616 20$:     MOVZWL  P2(AP),R1                  ; GET BUFFER SIZE
            19   13 025A   617          BEQL    ZEROLENGTH                 ; IF EQL THEN COMPLETE HERE
   42 A5    51   B1 025C   618          CMPW    R1,UCB$W_DEVBUFSIZ(R5)     ; MESSAGE SIZE IN RANGE?
            08   1A 0260   619          BGTRU   50$                        ; IF GTRU THEN NO
       50   6C   D0 0262   620          MOVL    P1(AP),R0                  ; GET BUFFER ADDRESS
   38 A3    50   D0 0265   621          MOVL    R0,IRP$L_MEDIA(R3)         ; SAVE BUFFER ADDRESS
                05 0269   622          RSB                                ; RETURN AND CHECK BUFFER
```

```
                50    019C 8F    3C   026A    623 50$:    MOVZWL  #SS$_MBTOOSML,R0          ; SET BOX TOO SMALL
                 00000000'GF    17   026F    624 ERROR:  JMP     G^EXE$ABORTIO             ; ABORT THE I/O
                                     0275    625
                                     0275    626 ;+
                                     0275    627 ; PROCESS ZERO LENGTH TRANSFERS
                                     0275    628 ;
                                     0275    629 ;  For a zero byte transfer, a dummy buffer (whose address is the current top
                                     0275    630 ;  of the current stack) of zero bytes length is constructed.  The normal
                                     0275    631 ;  access checks must be bypassed for this buffer because the previous caller
                                     0275    632 ;  may not have access to the current stack.
                                     0275    633 ;-
                                     0275    634
                                     0275    635 ZEROLENGTH:
                        8E    D5   0275    636         TSTL    (SP)+                       ; Pop checking rout. addr. from stack.
                      32 A3   D4   0277    637         CLRL    IRP$L_BCNT(R3)              ; Set zero byte count.
                38 A3   6E    9E   027A    638         MOVAB   (SP), IRP$L_MEDIA(R3)       ; Set top-of-stack buffer address.
                              05   027E    639         RSB                                 ; Return directly to routines caller.
                                   027F    640
                                   027F    641         .DSABL  LSB
```

```
                     027F   643          .SBTTL  FDTREAD - READ FUNCTION DECISION ROUTINE
                     027F   644 ;++
                     027F   645 ; FDTREAD - FUNCTION DECISION ROUTINE FOR READ OPERATIONS
                     027F   646 ;
                     027F   647 ; FUNCTIONAL DESCRIPTION:
                     027F   648 ;
                     027F   649 ; THE USER REQUEST IS VAILDATED FOR:
                     027F   650 ;
                     027F   651 ;       1.ACCESS BY UIC.
                     027F   652 ;       2.REQUEST WITHIN MAXIMUM MESSAGE SIZE.
                     027F   653 ;       3.BUFFER ACCESSIBLE FOR WRITE.
                     027F   654 ;
                     027F   655 ; THEN THE PACKET IS QUEUED ONTO THE UCB FOR PROCESSING WHEN THE UNIT IS IDLE.
                     027F   656 ;  FOR READ NOW FUNCTIONS, IF NO MESSAGES ARE PRESENT IN THE MAILBOX
                     027F   657 ; THE OPERATION IS COMPLETED.
                     027F   658 ;
                     027F   659 ; INPUTS:
                     027F   660 ;
                     027F   661 ;       R3 = I/O PACKET ADDRESS
                     027F   662 ;       R4 = CURRENT PCB ADDRESS
                     027F   663 ;       R5 = UCB ADDRESS
                     027F   664 ;       R6 = CCB ADDRESS
                     027F   665 ;       R7 = FUNCTION CODE
                     027F   666 ;       AP = PARAMETER BLOCK AT USER ARGUMENT ''P1''
                     027F   667 ;
                     027F   668 ; OUTPUTS:
                     027F   669 ;
                     027F   670 ;       THE PACKET IS QUEUED VIA ''EXE$QIODRVPKT'' OR
                     027F   671 ;       THE REQUEST IS COMPLETED WITH AN ERROR VIA ''EXE$ABORTIO''
                     027F   672 ;
                     027F   673 ; STATUS CODES:
                     027F   674 ;
                     027F   675 ;       SS$_NOPRIV - USER DOES NOT HAVE PRIVILEGE TO READ MAILBOX
                     027F   676 ;       SS$_ACCVIO - BUFFER ACCESS VIOLATION ( ''EXE$READCHK'' )
                     027F   677 ;       SS$_MBTOOSML - REQUEST EXCEEDS THE MAXIMUM MESSAGE SIZE
                     027F   678 ;       SS$_ENDOFFILE - FOR ''READNOW'' NO DATA RETURN
                     027F   679 ;       SS$_NORMAL - NORMAL STATUS ( ''STARTIO'' )
                     027F   680 ;--
                     027F   681 FDTREAD:                                     ;
                A4   10 027F   682          BSBB    READCHECKIO             ; VALIDATE THE REQUEST
2A A2   0400 8F   A8 0281   683          BISW    #IRP$M_MBXIO,IRP$W_STS(R3); SET MAILBOX READ
                     0287   684 ;
                     0287   685 ; UPDATE MEASUREMENT COUNTER IF ENABLED
                     0287   686 ;
            00000002 0287   687          .IF NE CA$_MEASURE
  00000000'EF   D6 0287   688          INCL    PMS$GL_MBREADS          ; COUNT MAILBOX READS
                     028D   689          .ENDC
                     028D   690 ;
                     028D   691 ; CHECK FOR READ NOW FUNCTIONS
                     028D   692 ;
 13 20 A3   06   E1 028D   693          BBC     #IO$V_NOW,IRP$W_FUNC(R3),50$; BR IF NOT ''NOW''
                     0292   694 ;
                     0292   695 ; READ NOW
                     0292   696 ;
                     0292   697 ; TEST TO SEE IF THERE ARE ANY MESSAGES WAITING
                     0292   698 ;
                     0292   699          SETIPL  #IPL$_MAILBOX           ; SET TO FORK IPL
```

```
                16 A5   B5  0295   700          TSTW    UCB$W_MSGCNT(R5)        ; ANY MESSAGES?
                   0B   12  0298   701          BNEQ    50$                     ; IF NEQ THEN YES
                            029A   702 ;
                            029A   703 ; COMPLETE ''READNOW'' FUNCTIONS BECAUSE NO MESSAGES ARE AVAILABLE
                            029A   704 ;
         50   0870 8F   3C  029A   705          MOVZWL  #SS$_ENDOFFILE,R0       ; SET NO TRANSFER AND STATUS
          00000000'GF   17  029F   706          JMP     G^EXE$FINISHIOC         ; COMPLETE THE I/O
                            02A5   707 ;
                            02A5   708 ; QUEUE PACKET TO DRIVER LIST
                            02A5   709 ;
          00000000'GF   17  02A5   710 50$:     JMP     G^EXE$QIODRVPKT         ; QUEUE PACKET ON UCB
```

```
                                        .SBTTL  FDTSET - HANDLE SET MODE FUNCTION
                                ;++
                                ; FDTSET - HANDLE SET MODE FUNCTION
                                ;
                                ; FUNCTIONAL DESCRIPTION:
                                ;
                                ; THIS ROUTINE IMPLEMENTS THE IO$_SETMODE FUNCTIONS.
                                ; THE DIFFERENT FUNCTIONS ARE SELECTED BY A FUNCTION CODE MODIFIER.
                                ; THE FUNCTIONS ARE:
                                ;
                                ;       IO$M_SETPROT    - SET VOLUME PROTECTION
                                ;       IO$M_READATTN   - SET READ ATTENTION AST
                                ;       IO$M_WRTATTN    - SET WRITE ATTENTION AST
                                ;
                                ; INPUTS:
                                ;
                                ;       R3 = I/O PACKET ADDRESS
                                ;       R4 = CURRENT PCB
                                ;       R5 = UCB ADDRESS FOR MAILBOX UNIT
                                ;
                                ; OUTPUTS:
                                ;
                                ;       NONE, THE I/O IS COMPLETED
                                ;
                                ; STATUS RETURNS:
                                ;
                                ;       SS$_NORMAL - SUCCESSUFL COMPLETION
                                ;       SS$_INSFMEM - INSUFICIENT MEMORY
                                ;       SS$_EXQUOTA - AST QUOTA EXCEEDED
                                ;       SS$_ILLIOFUNC - ILLEGAL SET MODE FUNCTION
                                ;       SS$_NOPRIV - NO PRIVILEGE FOR SETPROT FUNCTION
                                ;--
                                FDTSET:                                 ; SET RECEIVE AST FUNCTION
                                ;
                                ; GET THE FUNCTION MODIFIER BITS, AND MAKE SURE THAT ONLY ONE
                                ; FUNCTION IS SPECIFIED.  IF NO BITS ARE SET, THEN IT IS A
                                ; SET-WRITE-ATTENTION-AST FUNCTION.
                                ;
      50    00F4 8F    3C      MOVZWL  #SS$_ILLIOFUNC,R0       ; ASSUME ILLEGAL I/O FUNCTION
51  20 A3    0A   06    EF      EXTZV   #IRP$V_FMOD,#IRP$S_FMOD,IRP$W_FUNC(R3),R1
                  12    13      BEQL    5$                      ; IF EQL THEN ASSUME DEFAULT
      52    51   01    C3      SUBL3   #1,R1,R2                ; SUBTRACT ONE FROM THE FUNCTION MODIFIER
      52    51   52    CB      BICL3   R2,R1,R2                ; CLEAR THE RESULT INTO THE ORIGINAL
            52   51    D1      CMPL    R1,R2                   ; ARE THEY EQUAL?
                 AA    12      BNEQ    ERROR                   ; IF NOT EQL, THEN MORE THAN ONE BIT SET
                               ;
                               ; SEE IF THIS IS A SETPROT FUNCTION
                               ;
                  09   E0      BBS     #IO$V_SETPROT,-         ; BRANCH IF SETPROT FUNCTION
               43 20 A3                 IRP$W_FUNC(R3),50$      ;
                               ;
                               ; SEE IF USER CAN READ THIS MAILBOX
                               ;
 00000000'GF     16            5$:     JSB     G^EXE$CHKRDACCES        ; CHECK THE ACCESS OF THIS USER
                                                                       ; R4 = PCB ADDRESS
```

```
                        02D0   769                                     ; R5 = UCB ADDRESS
          9C 50   E9    02D0   770          BLBC    R0,ERROR             ; IF LOW CLEAR THEN ERROR
    57    0C A5   9E    02D3   771          MOVAB   UCB$L_MB_W_AST(R5),R7 ; ASSUME WRITER AST
02 20 A3    07    E1    02D7   772          BBC     #IO$V_READATTN,IRP$W_FUNC(R3),10$; BR IF NOT READER AST
              87    D5    02DC   773          TSTL    (R7)+                ; POINT TO READER AST LIST
          0090  8F    BB    02DE   774 10$:     PUSHR   #^M<R4,R7>           ; SAVE PCB AND LIST HEAD
  00000000'GF     16    02E2   775          JSB     G^COM$SETATTNAST     ; CONTINUE IN COMMON
              12    BA    02E8   776          POPR    #^M<R1,R4>           ; POP PCB AND SET LIST HEAD ADDRESS
                        02EA   777          SETIPL  #IPL$_MAILBOX        ; SET UP THE IPL
07 20 A3    07    E0    02ED   778          BBS     #IO$V_READATTN,IRP$W_FUNC(R3),15$; BR IF READER AST
                        02F2   779                                     ; DEFUALT IS WRITE ATTN
          16 A5   B5    02F2   780          TSTW    UCB$W_MSGCNT(R5)     ; ANY MESSAGES?
              0D  13    02F5   781          BEQL    25$                  ; IF EQL THEN NONE
              05  11    02F7   782          BRB     20$                  ; IF NEQ THEN DELIVER AST
06 64 A5    08    E1    02F9   783 15$:     BBC     #UCB$V_BSY,UCB$W_STS(R5),25$; BR IF NOT BUSY
  00000000'GF     16    02FE   784 20$:     JSB     G^COM$DELATTNAST     ; DELIVER THE ASTS
          54    51    D0    0304   785 25$:     MOVL    R1,R4                ; RESTORE PCB
  00000000'GF     17    0307   786 30$:     JMP     G^EXE$FINISHIOC      ; COMPLETE THE I/O
                        030D   787 ;
                        030D   788 ; HANDLE THE SETPROT FUNCTION
                        030D   789 ;
          50    24    3C    030D   790 50$:     MOVZWL  #SS$_NOPRIV,R0       ; ASSUME NO PRIVILEGE
    51    1C A5   D0    0310   791          MOVL    UCB$L_ORB(R5),R1     ; GET THE ORB ADDRESS
          00BC  C4    D1    0314   792          CMPL    PCB$L_UIC(R4),-      ; IS THIS THE VOLUME OWNER?
              61            0318   793                  ORB$L_OWNER(R1)
              14  12    0319   794          BNEQ    52$                  ; BRANCH IF NOT
    50    04 AC   3C    031B   795 51$:     MOVZWL  P2(AP),R0            ; GET THE PROTECTION MASK
                        031F   796          SETIPL  #IPL$_MAILBOX        ; DISABLE DEVICE INTERRUPTS
    0B A1   01  88    0322   797          BISB2   #ORB$M_PROT_16,ORB$B_FLAGS(R1) ; PROTECTION WORD NOT VECTOR
    18 A1   50  B0    0326   798          MOVW    R0,ORB$W_PROT(R1)    ; SET THE NEW PROTECTION MASK
          50    01  3C    032A   799          MOVZWL  #SS$_NORMAL,R0       ; SET SUCCESS STATUS
              D8  11    032D   800          BRB     30$                  ; COMPLETE THE I/O
              1D  E0    032F   801 52$:     BBS     #PRV$V_BYPASS,-      ; BRANCH IF USER HAS BYPASS
E7 6C B4            0331   802                  @PCB$L_PHD(R4),51$   ;
        FF38    31    0334   803          BRW     ERROR                ; ABORT THE I/O
                        0337   804          .SBTTL  FDTEOF - WRITE EOF MESSAGE TO MAILBOX
                        0337   805 ;++
                        0337   806 ; FDTEOF - WRITE EOF MESSAGE TO THE MAILBOX
                        0337   807 ;
                        0337   808 ; FUNCTIONAL DESCRIPTION:
                        0337   809 ;
                        0337   810 ; THIS IS THE FDT ROUTINE FOR IO$WRITEOF. THE ACTION IS TO BUILD A
                        0337   811 ; ZERO LENGTH MESSAGE AND TO INSERT IT IN THE MAILBOX.  THE FUNCTION
                        0337   812 ; CODE IS SAVED IN THE MESSAGE AND INDICATES THAT THE MESSAGE WAS AN
                        0337   813 ; END-OF-FILE.
                        0337   814 ;
                        0337   815 ; INPUTS:
                        0337   816 ;
                        0337   817 ;       R3 = I/O PACKET ADDRESS
                        0337   818 ;       R4 = CURRENT PCB ADDRESS
                        0337   819 ;       R5 = MAILBOX UCB ADDRESS
                        0337   820 ;       R7 = I/O FUNCTION CODE.
                        0337   821 ;
                        0337   822 ; OUTPUTS:
                        0337   823 ;
                        0337   824 ;       IRP$L_MEDIA(R3) = TOP-OF-STACK (dummy "buffer" address)
                        0337   825 ;       IRP$L_BCNT(R3) = ZERO (dummy "buffer" size)
```

```
                     0337   826 ;
                     0337   827 ;       THE I/O IS COMPLETED IN THE WRITE FDT LOGIC. ( SEE BELOW)
                     0337   828 ;--
                     0337   829 FDTEOF:                                                ;
          30 A3      D4  0337   830          CLRL    IRP$W_BOFF(R3)                    ; SET NO TRANSFER AND NO QUOTA
00000000'GF   16      033A   831          JSB     G^EXE$CHKWRTACCES                 ; CHECK THE ACCESS
                     0340   832                                                    ; R4 = PCB ADDRESS
                     0340   833                                                    ; R5 = UCB ADDRESS
          09 50      E9  0340   834          BLBC    R0,10$                            ; IF ERROR THEN BR
          32 A3      D4  0343   835          CLRL    IRP$L_BCNT(R3)                    ; SET NO DATA
    38 A3    6E      9E  0346   836          MOVAB   (SP), IRP$L_MEDIA(R3)             ; FAKE GOOD ADDRESS FOR THE FUTURE MOVC
            06       11  034A   837          BRB     WRITE                             ; WRITE THE MESSAGE
          FF20       31  034C   838 10$:     BRW     ERROR                             ; CONTINUE
```

```
                     034F   840          .SBTTL  FDTWRITE - WRITE OPERATION FDT ROUTINE
                     034F   841 ;++
                     034F   842 ; FDTWRITE -- FUNCTION DECISION ACTION ROUTINE FOR WRITE FUNCTIONS
                     034F   843 ;
                     034F   844 ; FUNCTIONAL DESCRIPTION:
                     034F   845 ;
                     034F   846 ; THE USER REQUEST IS VALIDATED FOR PRIVILEGE, SIZE, ACCESS AND AVAILABLE
                     034F   847 ; SPACE. IF VALID, A BUFFERED I/O BLOCK IS ALLOCATED (IMPLIED RESOURCE WAIT).
                     034F   848 ; THE BLOCK IS SET UP AND QUEUED TO THE UNIT MESSAGE LIST. IF THE UNIT
                     034F   849 ; IS BUSY, THE OUTSTANDING READ OPERATION IS COMPLETED DIRECTLY.
                     034F   850 ; IN THE CASE OF "WRITENOW" FUNCTIONS THE I/O IS COMPLETED BEFORE THE
                     034F   851 ; MESSAGE IS QUEUED. OTHERWISE THE READ COMPLETE ROUTINE COMPLETES
                     034F   852 ; THE MESSAGE ASSOCIATED WRITE.
                     034F   853 ;
                     034F   854 ; INPUTS:
                     034F   855 ;
                     034F   856 ;       R3 = I/O PACKET ADDRESS
                     034F   857 ;       R4 = CURRENT PCB ADDRESS
                     034F   858 ;       R5 = UCB ADDRESS
                     034F   859 ;       R6 = CCB ADDRESS
                     034F   860 ;       R7 = FUNCTION CODE
                     034F   861 ;       AP = ADDRESS OF USER ARGUMENT BLOCK AT "P1"
                     034F   862 ;
                     034F   863 ; OUTPUTS:
                     034F   864 ;
                     034F   865 ;       THE I/O IS COMPLETED IN ERROR, THE I/O IS RESTARTED BECAUSE OF
                     034F   866 ;       RESOURCE WAIT, OR THE I/O IS COMPETED NORMALLY.
                     034F   867 ;
                     034F   868 ; STATUS RETURNS:
                     034F   869 ;
                     034F   870 ;       SS$_MBTOOSML - MESSAGE IS TOO BIG
                     034F   871 ;       SS$_ACCVIO - BUFFER ACCESS VIOLATION ( "EXE$WRITECHK" )
                     034F   872 ;       SS$_MBFULL - MAILBOX IS FULL
                     034F   873 ;       SS$_NOPRIV - USER DOES NOT HAVE WRITE PRIVILEGE
                     034F   874 ;       SS$_NORMAL - SUCCESSFUL STATUS
                     034F   875 ;       SS$_INSFMEM - NO MEMORY FOR BUFFER ALLOCATION
                     034F   876 ;--
                     034F   877 FDTWRITE:                                ;
           FEE5   30 034F   878         BSBW    WRITECHECKIO             ; CHECK OPERATION PARAMETERS
                     0352   879
                     0352   880 ;+
                     0352   881 ;  At this point, the following inputs are assumed:
                     0352   882 ;
                     0352   883 ;       R3 = I/O PACKET ADDRESS
                     0352   884 ;       R4 = CURRENT PCB ADDRESS
                     0352   885 ;       R5 = UCB ADDRESS
                     0352   886 ;       R7 = FUNCTION CODE
                     0352   887 ;
                     0352   888 ;       IRP$L_MEDIA(R3) = BUFFER ADDRESS
                     0352   889 ;       IRP$L_BCNT(R3) = BUFFER SIZE (low order only)
                     0352   890 ;-
                     0352   891
                     0352   892 WRITE:
     51    32 A3   18   C1 0352   893         ADDL3   #MBMSG_C_HEADER,IRP$W_BCNT(R3),R1 ; COMPUTE SIZE W/ HEADER
                     0357   894 ;
                     0357   895 ; GET BUFFER
                     0357   896 ;
```

```
                53   DD  0357   897          PUSHL   R3                              ; SAVE IRP ADDRESS
  00000000'GF   16   0359   898          JSB     G^EXE$ALONONPAGED               ; ALLOCATE A BUFFER
                53 8ED0  035F   899          POPL    R3                              ; RESTORE IRP ADDRESS
              57 50   E9  0362   900          BLBC    R0,55$                          ; CONTINUE
                        0365   901 ;
                        0365   902 ; SET UP BLOCK
                        0365   903 ;
                3C   BB  0365   904          PUSHR   #^M<R2,R3,R4,R5>                ; SAVE BLOCK,PACKET,PCB AND UCB
            52   08   C0  0367   905          ADDL    #8,R2                           ; POINT PAST FIXED PART
            82   51   B0  036A   906          MOVW    R1,(R2)+                        ; INSERT SIZE
            82   13   90  036D   907          MOVB    #DYN$C_BUFIO,(R2)+              ; INSERT TYPE
            82   57   90  0370   908          MOVB    R7,(R2)+                        ; INSERT FUNCTION CODE
         82   32 A3   B0  0373   909          MOVW    IRP$W_BCNT(R3),(R2)+            ; INSERT MESSAGE SIZE
            82   53   D0  0377   910          MOVL    R3,(R2)+                        ; INSERT SAVED PACKET ADDRESS
   03 20 A3   06   E1  037A   911          BBC     #IO$V_NOW,IRP$W_FUNC(R3),15$; BR IF NOT "NOW"
            FC A2   D4  037F   912          CLRL    -4(R2)                          ; RESET MESSAGE PACKET POINTER
         82   60 A4   D0  0382   913 15$:    MOVL    PCB$L_PID(R4),(R2)+             ; INSERT PID OF SENDER
                        0386   914 ;
                        0386   915 ; COPY DATA FROM USER TO SYSTEM
                        0386   916 ;
62 38 B3   32 A3   28   0386   917          MOVC3   IRP$W_BCNT(R3), -               ; MOVE CHARACTERS TO SYSTEM SPACE
                                038C   918                  @IRP$L_MEDIA(R3), (R2)
                3C   BA  038C   919          POPR    #^M<R2,R3,R4,R5>                ; RESTORE REGISTERS
                        038E   920 ;
                        038E   921 ; CHECK TO SEE IF THERE IS ROOM IN MAILBOX FOR MESSAGE.  THAT IS, THAT
                        038E   922 ; THE MAILBOX'S BUFFER QUOTA IS NOT EXCEEDED.  THIS MUST BE DONE AT FORK
                        038E   923 ; IPL TO INTERLOCK WITH OTHER MAILBOX WRITERS.
                        038E   924 ;
                        038E   925 20$:    DSBINT  #IPL$_MAILBOX                   ; RAISE TO DRIVER FORK LEVEL
18 A5   32 A3   B1  0394   926          CMPW    IRP$W_BCNT(R3),UCB$W_ UFQUO(R5); MESSAGE FIT?
                2B   1A  0399   927          BGTRU   60$                             ; IF GTR THEN NO
                        039B   928 ;
                        039B   929 ; QUEUE THE MESSAGE
                        039B   930 ;
                28   BB  039B   931          PUSHR   #^M<R3,R5>                      ; SAVE UCB ADDRESS AND PACKET
                51   10  039D   932          BSBB    INSMBQUEUE                      ; INSERT THE MESSAGE
                28   BA  039F   933          POPR    #^M<R3,R5>                      ; RESTORE UCB ADDRESS AND PACKET
                        03A1   934          ENBINT                                  ; LOWER IPL
                        03A4   935 ;
                        03A4   936 ; SEE IF WRITE I/O GETS COMPLETED NOW
                        03A4   937 ;
06 20 A3   06   E0  03A4   938          BBS     #IO$V_NOW,IRP$W_FUNC(R3),50$; BR IF WRITE NOW
  00000000'GF   17  03A9   939          JMP     G^EXE$QIORETURN                 ; RETURN TO CALLER
                        03AF   940 ;
                        03AF   941 ; FINISH WRITE I/O OPERATION
                        03AF   942 ;
 50   30 A3   D0  03AF   943 50$:    MOVL    IRP$W_BCNT-2(R3),R0             ; GET TRANSFER COUNT
      50   01   B0  03B3   944          MOVW    #SS$_NORMAL,R0                  ; SET STATUS IN LOW
  00000000'GF   17  03B6   945          JMP     G^EXE$FINISHIOC                 ; COMPLETE THE I/O
                        03BC   946 ;
                        03BC   947 ; INSUFFICIENT MEMORY TO BUFFER MESSAGE - WAIT FOR RESOURCE
                        03BC   948 ;
50   0124 8F   3C  03BC   949 55$:    MOVZWL  #SS$_INSFMEM,R0                 ; SET INSUFFICIENT MEMORY STATUS
     51   03   9A  03C1   950          MOVZBL  #RSN$_NPDYNMEM,R1               ; SET RESOURCE TO AWAIT
          19   11  03C4   951          BRB     65$                             ;
                        03C6   952 ;
                        03C6   953 ; MAILBOX IS FULL - WAIT FOR A MESSAGE TO BE READ
```

```
                        03C6   954 ;
                        03C6   955 60$:     ENBINT                          ; RESTORE IPL TO IPL$_ASTDEL
          50    52      D0    03C9   956         MOVL    R2,R0               ; DEALLOCATE MESSAGE BLOCK
                52      DD    03CC   957         PUSHL   R2                  ; SAVE PACKET
  00000000'GF   16      03CE   958         JSB     G^EXE$DEANONPAGED           ;
                53 8ED0 03D4   959         POPL    R2                  ; RESTORE PACKET
  50    08D8 8F   3C    03D7   960         MOVZWL  #SS$_MBFULL,R0              ; SET MAILBOX FULL STATUS
          51    02      9A    03DC   961         MOVZBL  #RSN$_MAILBOX,R1            ; SET RESOURCE TO AWAIT
06 20 A3    0A      E0    03DF   962 65$:     BBS     #IO$V_NORSWAIT, -           ; IS NO RESOURCE WAIT MODIFIER SET?
                        03E4   963                     IRP$W_FUNC(R2), 69$         ; BRANCH IF MODIFER IS SET.
  00000000'GF   17      03E4   964         JMP     G^EXE$IORSNWAIT             ; ELSE, DO POSSIBLE RESOURCE WAIT.
  00000000'GF   17      03EA   965 69$:     JMP     G^EXE$ABORTIO               ; ABORT I/O TO AVOID RESOURCE WAITS.
```

```
                        03F0   967
                        03F0   968          .SBTTL INSERT MESSAGE IN MAILBOX QUEUE
                        03F0   969 ;++
                        03F0   970 ; INSMBQUEUE - INSERT MESSAGE ON MAILBOX QUEUE
                        03F0   971 ;
                        03F0   972 ; INPUTS:
                        03F0   973 ;
                        03F0   974 ;       R2 = ADDRESS OF MESSAGE BLOCK
                        03F0   975 ;       R5 = UCB OF MAILBOX
                        03F0   976 ;
                        03F0   977 ; OUTPUTS:
                        03F0   978 ;
                        03F0   979 ;       THE MESSAGE IS QUEUED AND IF THE UNIT IS BUSY THEN
                        03F0   980 ;       CONTROL IS TRANSFERED TO "FINISHREAD" TO COMPLETE THE
                        03F0   981 ;       WAITING READ REQUEST.
                        03F0   982 ;--
                        03F0   983 INSMBQUFUE:                          ;
          16 A5   B6    03F0   984          INCW    UCB$W_MSGCNT(R5)      ; ADJUST MESSAGE COUNT
44 A5     16 A5   B0    03F3   985          MOVW    UCB$W_MSGCNT(R5),UCB$L_DEVDEPEND(R5); SAVE IT
18 A5     0C A2   A2    03F8   986          SUBW    12(R2),UCB$W_BUFQUO(R5) ; ADJUST BYTE QUOTA BY MESSAGE SIZE
              00000002  03FD   987          .IF NE CA$_MEASURE              ; CHECK FOR MEASUREMENT ENABLED
  00000000'EF   D6      03FD   988          INCL    PMS$GL_MBWRITES         ; COUNT MAILBOX WRITES
                        0403   989          .ENDC
                        0403   990 ;
                        0403   991 ; TEST UNIT BUSY -- IF BUSY FINISH OUTSTANDING READ
                        0403   992 ;
1E 64 A5    08    E0    0403   993          BBS     #UCB$V_BSY,UCB$W_STS(R5),FINISHREAD; BRANCH IF BUSY
                        0408   994 ;
                        0408   995 ; INSERT MESSAGE IN QUEUE
                        0408   996 ;
   04 B5    62    0E    0408   997          INSQUE  (R2),@UCB$L_MB_MSGQ+4(R5); INSERT MESSAGE IN QUEUE
                        040C   998 ;
                        040C   999 ; DELIVER ALL AST'S WAITING FOR MESSAGES ON THIS MAILBOX
                        040C  1000 ;
 54    0C A5    9E    040C  1001          MOVAB   UCB$L_MB_W_AST(R5),R4   ; ADDRESS ATTENTION LIST HEAD
00000000'GF    17    0410  1002          JMP     G^COM$DECATTNAST        ; DELIVER THE AST'S
```

```
                     0416  1004          .SBTTL  STARTIO - STARTIO OPERATION
                     0416  1005 ;++
                     0416  1006 ; STARTIO - START READ OPERATION ON IDLE MAILBOX UNIT
                     0416  1007 ;
                     0416  1008 ; FUNCTIONAL DESCRIPTION:
                     0416  1009 ;
                     0416  1010 ; THIS ROUTINE IS ENTERED WHEN THE UNIT IS NOT BUSY AND THERE IS A
                     0416  1011 ; PACKET TO PROCESS. IF THERE IS ANY MESSAGE WAITING THE READ IS COMPLETED
                     0416  1012 ; OTHERWISE, AN RSB IS DONE LEAVING THE UNIT BUSY AND THE PACKET IN
                     0416  1013 ; LIMBO.
                     0416  1014 ;
                     0416  1015 ; INPUTS:
                     0416  1016 ;
                     0416  1017 ;       R3 = I/O PACKET ADDRESS
                     0416  1018 ;       R5 = UCB ADDRESS
                     0416  1019 ;
                     0416  1020 ; OUTPUTS:
                     0416  1021 ;
                     0416  1022 ;       R2 = MESSAGE ADDRESS ON TRANSFER TO ''FINISHREAD''.
                     0416  1023 ;
                     0416  1024 ;       OTHERWISE AN RSB IS DONE.
                     0416  1025 ;--
                     0416  1026 STARTIO:                                             ;
 52    00 B5    0F   0416  1027          REMQUE  @UCB$L_MB_MSGQ(R5),R2               ; GET MESSAGE IF ANY FROM QUEUE
         0A    1C   041A  1028          BVC     FINISHREAD                          ; IF V-CLEAR THEN COMPLETE THE READ
 54    10 A5    9E   041C  1029          MOVAB   UCB$L_MB_R_AST(R5),R4               ; ADDRESS LIST OF READER AST'S
00000000'GF    17   0420  1030          JMP     G^COM$DELATTNAST                    ; DELIVER AST'S
```

```
                     0426  1032          .SBTTL  FINISHREAD - FINISH READ I/O OPERATION
                     0426  1033 ;++
                     0426  1034 ; FINISHREAD - FINISH READ OPERATION
                     0426  1035 ;
                     0426  1036 ; FUNCTIONAL DECRIPTION:
                     0426  1037 ;
                     0426  1038 ; THIS ROUTINE IS ENTERED WHEN THE UNIT IS BUSY AND A MESSAGE
                     0426  1039 ; IS AVAILABLE.
                     0426  1040 ; THE WAITING READ IS COMPLETED ALONG WITH THE MATCHING WRITE
                     0426  1041 ; REQUEST IF THE WRITE WAS A WAIT TYPE.
                     0426  1042 ;
                     0426  1043 ; INPUTS:
                     0426  1044 ;
                     0426  1045 ;       R2 = MESSAGE ADDRESS
                     0426  1046 ;       R5 = UCB ADDRESS
                     0426  1047 ;
                     0426  1048 ; OUTPUTS:
                     0426  1049 ;
                     0426  1050 ;--
                     0426  1051 FINISHREAD:                                  ;
      53   58 A5    D0  0426  1052          MOVL    UCB$L_IRP(R5),R3          ; GET CURRENT I/O PACKET
      2C A3   52    D0  042A  1053          MOVL    R2,IRP$L_SVAPTE(R3)       ; INSERT BLOCK ADDRESS IN PACKET
      82   16 A2    9E  042E  1054          MOVAB   22(R2),(R2)+              ; INSERT ADDRESS OF DATA
      82   38 A3    D0  0432  1055          MOVL    IRP$L_MEDIA(R3),(R2)+     ; INSERT USER VIRTUAL ADDRESS
           82        D5  0436  1056          TSTL    (R2)+                     ; PASS TYPE WORD
      18 A5   62    A0  0438  1057          ADDW    (R2),UCB$W_BUFQUO(R5)     ; ADJUST QUOTA BY MESSAGE BLOCK SIZE
         7E   52    7D  043C  1058          MOVQ    R2,-(SP)
         50   02    9A  043F  1059          MOVZBL  #RSN$_MAILBOX,R0          ; DECLARE RESOURCE AVAILABLE
00000000'GF   16    0442  1060          JSB     G^SCH$RAVAIL
         52   8E    7D  0448  1061          MOVQ    (SP)+,R2
      50   0601 8F    B0  044B  1062          MOVW    #SS$_BUFFEROVF, R0        ; Assume buffer overflow.
        82   32 A3    B1  0450  1063          CMPW    IRP$W_BCNT(R3), (R2)+     ; Was there a buffer overflow?
             08        1F  0454  1064          BLSSU   10$                       ; Branch if buffer overflow.
      32 A3   FE    A2  B0  0456  1065          MOVW    -2(R2), IRP$W_BCNT(R3)    ; Else, xfer only bytes in message
      50      01    B0  045B  1066          MOVW    #SS$_NORMAL, R0           ; and set normal xfer completed.
50 10   10   32 A3    F0  045E  1067 10$:     INSV    IRP$W_BCNT(R3), #16, #16, R0 ; Plant bytes transfered count.
      28   FD A2    91  0464  1068          CMPB    -3(R2),#IO$_WRITEOF       ; WAS FUNCTION END-OF-FILE?
           05        12  0468  1069          BNEQ    15$                       ; BR IF NOT
      50   0870 8F    B0  046A  1070          MOVW    #SS$_ENDOFFILE,R0         ; SET EOF STATUS
        16   A5    B7  046F  1071 15$:     DECW    UCB$W_MSGCNT(R5)          ; ADJUST MESSAGE COUNT
      44 A5   16    A5  B0  0472  1072          MOVW    UCB$W_MSGCNT(R5),UCB$L_DEVDEPEND(R5); SAVE IT
           09        BB  0477  1073          PUSHR   #^M<R0,R3>                ; SAVE PACKET OF READER AND STATUS
         51   0C A3    D0  0479  1074          MOVL    IRP$L_PID(R3),R1          ; GET READER PID
           53   82    D0  047D  1075          MOVL    (R2)+,R3                  ; GET WRITER PACKET
                1C    13  0480  1076          BEQL    20$                       ; IF EQL THEN NONE
      38 A3   01    B0  0482  1077          MOVW    #SS$_NORMAL, -            ; Return success to the writer.
                        0486  1078                  IRP$L_MEDIA(R3)
      3A A3   32 A3    B0  0486  1079          MOVW    IRP$W_BCNT(R3), -         ; Get writer bytes transfer equal
                        048B  1080                  IRP$L_MEDIA+2(R3)         ; to request byte count.
         50   51    D0  048B  1081          MOVL    R1, R0                    ; Move internal reader pid for call
00000000'GF   16    048E  1082          JSB     G^EXE$IPID_TO_EPID        ; Convert to extended pid
      3C A3   50    D0  0494  1083          MOVL    R0, IRP$L_MEDIA+4(R3)     ; Return reader EPID to writer.
00000000'GF   16    0498  1084          JSB     G^COM$POST                ; Complete the i/o
         50   62    D0  049E  1085 20$:     MOVL    (R2),R0                   ; Get internal writer pid ready for call
00000000'GF   16    04A1  1086          JSB     G^EXE$IPID_TO_EPID        ; Convert to extended pid
         51   50    D0  04A7  1087          MOVL    R0,R1                     ; Put the writer EPID where REQCOM wants it
              09    BA  04AA  1088          POPR    #^M<R0,R3>                ; Restore status and packet address
```

```
04AC  1089          REQCOM                  ; Complete request!!!
04B2  1090 MB_END:
04B2  1091          .END
```

```
$$$                             = 00000020 R      02
ARM$M_WRITE                     = 00000002
AT$_UBA                         = 00000001
BUG$_INCONSTATE                   ********    X   03
CAS_MEASURE                     = 00000002
CAN$C_AMBXDGN                   = 00000002
CAN$C_DASSGN                    = 00000001
CANCELIO                          0000014C R      03
CCB$B_STS                       = 00000008
CCB$V_RDCHKDON                  = 00000002
CCB$V_WRTCHKDON                 = 00000003
COM$DELATTNAST                    ********    X   03
COM$DRVDEALMEM                    ********    X   03
COM$FLUSHATTNS                    ********    X   03
COM$POST                          ********    X   03
COM$SETATTNAST                    ********    X   03
DPT$C_LENGTH                    = 00000038
DPT$C_VERSION                   = 00000004
DPT$INITAB                        00000038 R      02
DPT$REINITAB                      00000038 R      02
DPT$TAB                           00000000 R      02
DYN$C_BUFIO                     = 00000013
DYN$C_DPT                       = 0000001E
ERROR                             0000026F R      03
EXE$ABORTIO                       ********    X   03
EXE$ALONONPAGED                   ********    X   03
EXE$CHKRDACCES                    ********    X   03
EXE$CHKWRTACCES                   ********    X   03
EXE$DEANONPAGED                   ********    X   03
EXE$FINISHIOC                     ********    X   03
EXE$IORSNWAIT                     ********    X   03
EXE$IPID_TO_EPID                  ********    X   03
EXE$QIODRVPRT                     ********    X   03
EXE$QIORETURN                     ********    X   03
EXE$READCHK                       ********    X   03
EXE$SNDEVMSG                      00000078 RG     03
EXE$WRITECHK                      ********    X   03
EXE$WRTMAILBOX                    000000CA RG     03
FDTEOF                            00000337 R      03
FDTREAD                           0000027F R      03
FDTSET                            000002AB R      03
FDTWRITE                          0000034F R      03
FINISHREAD                        00000426 R      03
FKB$C_LENGTH                    = 00000018
FUNCTABLE                         00000038 R      03
FUNCTAB_LEN                     = 00000040
INSMBQUEUE                        000003F0 R      03
IO$V_NORSWAIT                   = 0000000A
IO$V_NOW                        = 00000006
IO$V_READATTN                   = 00000007
IO$V_SETPROT                    = 00000009
IO$_READLBLK                    = 00000021
IO$_READPBLK                    = 0000000C
IO$_READVBLK                    = 00000031
IO$_SETMODE                     = 00000023
IO$_VIRTUAL                     = 0000003F
IO$_WRITELBLK                   = 00000020
IO$_WRITEOF                     = 00000028
IO$_WRITEPBLK                   = 0000000B
IO$_WRITEVBLK                   = 00000030
IOC$CVT_DEVNAM                    ********    X   03
IOC$MNTVER                        ********    X   03
IOC$REQCOM                        ********    X   03
IOC$RETURN                        ********    X   03
IPL$_ASTDEL                     = 00000002
IPL$_MAILBOX                    = 0000000B
IPL$_SYNCH                      = 00000008
IRP$C_BCNT                      = 00000032
IRP$L_MEDIA                     = 00000038
IRP$L_PID                       = 0000000C
IRP$L_SVAPTE                    = 0000002C
IRP$M_MBXIO                     = 00000400
IRP$S_FMOD                      = 0000000A
IRP$V_FMOD                      = 00000006
IRP$W_BCNT                      = 00000032
IRP$W_BOFF                      = 00000030
IRP$W_CHAN                      = 00000028
IRP$W_FUNC                      = 00000020
IRP$W_STS                       = 0000002A
LNM$DELETE_LNMB                   ********    X   03
LNM$LOCKW                         ********    X   03
LNM$UNLOCK                        ********    X   03
MASKH                           = 00000100
MASKL                           = 00000000
MB$DDT                            00000000 RG     03
MB$DPT                            00000000 RG     02
MBMSG_B_FUNC                      0000000B
MBMSG_B_TYPE                      0000000A
MBMSG_C_HEADER                  = 00000018
MBMSG_C_LENGTH                    00000016
MBMSG_L_BLINK                     00000004
MBMSG_L_FLINK                     00000000
MBMSG_L_IRP                       0000000E
MBMSG_L_PID                       00000012
MBMSG_T_DATA                      00000016
MBMSG_W_DATSIZ                    0000000C
MBMSG_W_SIZE                      00000008
MB_END                            000004B2 R      03
ORB$B_FLAGS                     = 0000000B
ORB$L_OWNER                     = 00000000
ORB$L_OWN_PROT                  = 0000001C
ORB$M_PROT_16                   = 00000001
ORB$V_PROT_16                   = 00000000
ORB$W_PROT                      = 00000018
P1                              = 00000000
P2                              = 00000004
P3                              = 00000008
P4                              = 0000000C
PCB$L_PHD                       = 0000006C
PCB$L_PID                       = 00000060
PCB$L_UIC                       = 000000BC
PMS$GL_MBREADS                    ********    X   03
PMS$GL_MBWRITES                   ********    X   03
PR$_IPL                         = 00000012
```

```
PRV$V_BYPASS                    = 0000001D
READCHECKIO                       00000225 R      03
RSN$_MAILBOX                    = 00000002
RSN$_NPDYNMEM                   = 00000003
SCH$GL_CURPCB                     ******** X    03
SCH$RAVAIL                        ******** X    03
SS$_ABORT                       = 0000002C
SS$_BUFFEROVF                   = 00000601
SS$_ENDOFFILE                   = 00000870
SS$_ILLIOFUNC                   = 000000F4
SS$_INSFMEM                     = 00000124
SS$_MBFULL                      = 000008D8
SS$_MBTOOSML                    = 0000019C
SS$_NOPRIV                      = 00000024
SS$_NORMAL                      = 00000001
STARTIO                           00000416 R      03
SYS$C_MBXUCBSIZ                   ******** X    02
UCB$L_ASTQBL                    = 00000010
UCB$L_ASTQFL                    = 0000000C
UCB$L_DEVDEPEND                 = 00000044
UCB$L_FQFL                      = 00000000
UCB$L_IRP                       = 00000058
UCB$L_LOGADR                    = 00000074
UCB$L_MB_MSGQ                   = 00000000
UCB$L_MB_R_AST                  = 00000010
UCB$L_MB_W_AST                  = 0000000C
UCB$L_ORB                       = 0000001C
UCB$L_STS                       = 00000064
UCB$M_DELETEUCB                 = 00010000
UCB$V_BSY                       = 00000008
UCB$V_DELMBX                    = 00000001
UCB$W_BUFQUO                    = 00000018
UCB$W_DEVBUFSIZ                 = 00000042
UCB$W_DEVSTS                    = 00000068
UCB$W_MSGCNT                    = 00000016
UCB$W_REFC                      = 0000005C
UCB$W_STS                       = 00000064
UCB$W_UNIT                      = 00000054
WRITE                             00000352 R      03
WRITECHECKIO                      00000237 R      03
ZEROLENGTH                        00000275 R      03
```

+----------------+
! Psect synopsis !
+----------------+

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	00000016 (22.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
$$$105_PROLOGUE	00000039 (57.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
$$$115_DRIVER	000004B2 (1202.)	03 (3.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	31	00:00:00.03	00:00:02.00
Command processing	121	00:00:00.55	00:00:03.33
Pass 1	518	00:00:21.23	00:01:07.75
Symbol table sort	0	00:00:03.49	00:00:11.67
Pass 2	201	00:00:04.44	00:00:17.31
Symbol table output	19	00:00:00.17	00:00:00.74
Psect synopsis output	2	00:00:00.03	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	894	00:00:29.95	00:01:42.84

The working set limit was 1800 pages.
122911 bytes (241 pages) of virtual memory were used to buffer the intermediate code.
There were 120 pages of symbol table space allocated to hold 2162 non-local and 41 local symbols.
1091 source lines were read in Pass 1, producing 20 object records in Pass 2.
40 pages of virtual memory were used to define 37 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYS.OBJ]LIB.MLB;1	25
_255DUA28:[SYSLIB]STARLET.MLB;2	10
TOTALS (all libraries)	35

2435 GETS were required to define 35 macros.

There were no errors, warnings or information messages.

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