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

```
TTTTTTTTTT  MM      MM  DDDDDDDD    RRRRRRRR      IIIIII    VV      VV  EEEEEEEEEE  RRRRRRRR
TTTTTTTTTT  MM      MM  DDDDDDDD    RRRRRRRR      IIIIII    VV      VV  EEEEEEEEEE  RRRRRRRR
    TT      MMMM  MMMM  DD      DD  RR      RR      II      VV      VV  EE          RR      RR
    TT      MMMM  MMMM  DD      DD  RR      RR      II      VV      VV  EE          RR      RR
    TT      MM  MM  MM  DD      DD  RR      RR      II      VV      VV  EE          RR      RR
    TT      MM  MM  MM  DD      DD  RR      RR      II      VV      VV  EE          RR      RR
    TT      MM      MM  DD      DD  RRRRRRRR        II      VV      VV  EEEEEEEE    RRRRRRRR
    TT      MM      MM  DD      DD  RRRRRRRR        II      VV      VV  EEEEEEEE    RRRRRRRR
    TT      MM      MM  DD      DD  RR  RR          II      VV      VV  EE          RR  RR
    TT      MM      MM  DD      DD  RR  RR          II      VV      VV  EE          RR  RR
    TT      MM      MM  DD      DD  RR    RR        II        VV  VV    EE          RR    RR        ....
    TT      MM      MM  DD      DD  RR    RR        II        VV  VV    EE          RR    RR        ....
    TT      MM      MM  DDDDDDDD    RR      RR    IIIIII        VV      EEEEEEEEEE  RR      RR      ....
    TT      MM      MM  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
```

```
0000     1          .TITLE  TMDRIVER - TM03-TE16/TU77 MAGTAPE DRIVER
0000     2          .IDENT  'V04-000'
0000     3
0000     4  ;
0000     5  ;*****************************************************************************
0000     6  ;*                                                                           *
0000     7  ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     8  ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     9  ;*  ALL RIGHTS RESERVED.                                                     *
0000    10  ;*                                                                           *
0000    11  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
0000    12  ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
0000    13  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
0000    14  ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
0000    15  ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
0000    16  ;*  TRANSFERRED.                                                             *
0000    17  ;*                                                                           *
0000    18  ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
0000    19  ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
0000    20  ;*  CORPORATION.                                                             *
0000    21  ;*                                                                           *
0000    22  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
0000    23  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    24  ;*                                                                           *
0000    25  ;*                                                                           *
0000    26  ;*****************************************************************************
0000    27  ;
0000    28  ; D. N. CUTLER 20-JUN-77
0000    29  ;
0000    30  ; MODIFIED BY:
0000    31  ;
0000    32  ;       V03-014 MMD0311         Meg Dumont,     19-Jul-1984  9:37
0000    33  ;               Add support for the UCB$L_MEDIA_ID field
0000    34  ;
0000    35  ;       V03-013 MMD0302         Meg Dumont,     20-Jun-1984  13:44
0000    36  ;               Fix to make READ forward into EOT return SS$_NORMAL.
0000    37  ;
0000    38  ;       V03-012 RAS0300         Ron Schaefer            27-Apr-1984
0000    39  ;               Add DEV$M_NNM characteristic to DECHAR2 so that these
0000    40  ;               devices will have the "node$" prefix.
0000    41  ;
0000    42  ;       V03-011 TMK0001         Todd M. Katz            08-Dec-1983
0000    43  ;               Fix a broken branch.
0000    44  ;
0000    45  ;       V03-010 ROW0258         Ralph O. Weber          17-NOV-1983
0000    46  ;                       The Paul Painter Memorial Enhancement
0000    47  ;               Named for one of the unfortunate customers who suffered much
0000    48  ;               to determine the great UCB$L_MT_RECORD secret while trying to
0000    49  ;               create a user-written magtape driver, this change eliminates
0000    50  ;               use of the device dependent field, UCB$L_MT_RECORD in favor of
0000    51  ;               the device independent field, UCB$L_RECORD.
0000    52  ;
0000    53  ;       V03-009 ROW0213         Ralph O. Weber          20-AUG-1983
0000    54  ;               Change basing for device-dependent UCB from UCB$L_DP_LINK+4 to
0000    55  ;               a field independent UCB$K_LCL_TAPE_LENGTH.  This allows the
0000    56  ;               device-indpendent UCB to be altered without having to edit
0000    57  ;               this module.
```

```
0000     58 ;
0000     59 ;       V03-008 RLRDPATH1       Robert L. Rappaport     31-May-1983
0000     60 ;               Allow UCB to include new DUAL PORT extension by
0000     61 ;               changing base of where we begin the private TMDRIVER
0000     62 ;               extension from UCB$L_DPC+4 to UCB$L_DP_LINK+4.
0000     63 ;
0000     64 ;       V03-007 RLR54598        Robert L. Rappaport     8-Mar-1982
0000     65 ;               Correct problem that crashes system when dealing with
0000     66 ;               unit number 7.
0000     67 ;
0000     68 ;       V03-006 RLR50799        Robert L. Rappaport     15-Dec-1982
0000     69 ;               Correct problem that caused clearing of Volume Valid
0000     70 ;               when attempting to mount or init tapes.
0000     71 ;
0000     72 ;       V03-005 RLR47029        Robert L. Rappaport     17-Aug-1982
0000     73 ;               Correct error that indicated records skipped when
0000     74 ;               skip reverse record occurred at BOT.  Also reset density
0000     75 ;               in DEVDEPEND after POSITIONING operations.
0000     76 ;
0000     77 ;       V03-004 RLREC002        Robert L. Rappaport     20-July-1982
0000     78 ;               Fix second error introduced by fix to FCE that caused 800
0000     79 ;               bpi tapes to ignore Tape Marks on reads.
0000     80 ;
0000     81 ;       V03-003 KDM0002         Kathleen D. Morse       28-Jun-1982
0000     82 ;               Added $DYNDEF, $DCDEF, $SSDEF,  and $VADEF.
0000     83 ;
0000     84 ;       V03-002 RLREC001a       Robert L. Rappaport     21-June-1982
0000     85 ;               Correct error introduced in the elimination of testing
0000     86 ;               for FCR (Frame Count Error) errors on reads.  This
0000     87 ;               inadvertently eliminated testing for Tape Marks.
0000     88 ;
0000     89 ;       V03-001 RLREC001        Robert L. Rappaport     17-May-1982
0000     90 ;               SPR44595.
0000     91 ;               Ignore FCE (Frame Count Error) errors on Read commands
0000     92 ;               and thereby allow Reads with IO$M_INHRETRY set to
0000     93 ;               succeed when the buffer is larger than the record.
0000     94 ;
0000     95 ;               QAR 1404.
0000     96 ;               Do not request Primary channel until possible REWIND
0000     97 ;               done in FDISPATCH.  Also RELCHAN before REQCOM.
0000     98 ;
0000     99 ;               Only try to clear SLA bit, once per interrupt, per drive
0000    100 ;               in TM$INT.
0000    101 ;               This tracks patch made in V3.1.
0000    102 ;
0000    103 ;
0000    104 ; TM03-TE16/TU77 MAGTAPE DRIVER
0000    105 ;
0000    106 ; MACRO LIBRARY CALLS
0000    107 ;
0000    108
0000    109         $ADPDEF                         ;DEFINE ADP OFFSETS
0000    110         $CRBDEF                         ;DEFINE CRB OFFSETS
0000    111         $DCDEF                          ;DEFINE DEVICE CLASSES
0000    112         $DDBDEF                         ;DEFINE DDB OFFSETS
0000    113         $DEVDEF                         ;DEFINE DEVICE CHARACTERISTICS BITS
0000    114         $DPTDEF                         ;DEFINE DPT OFFSETS
```

```
0000    115          $DYNDEF                         ;DEFINE DYNAMIC DATA STRUCTURE TYPES
0000    116          $EMBDEF                         ;DEFINE EMB OFFSETS
0000    117          $IDBDEF                         ;DEFINE IDB OFFSETS
0000    118          $IODEF                          ;DEFINE I/O FUNCTION CODES
0000    119          $IRPDEF                         ;DEFINE IRP OFFSETS
0000    120          $MBADEF                         ;DEFINE MBA REGISTER OFFSETS
0000    121          $MTDEF                          ;DEFINE MAGTAPE STATUS BITS
0000    122          $SSDEF                          ;DEFINE SYSTEM STATUS CODES
0000    123          $UCBDEF                         ;DEFINE UCB OFFSETS
0000    124          $VADEF                          ;DEFINE VIRTUAL ADDRESS FIELDS
0000    125          $VECDEF                         ;DEFINE INTERRUPT DISPATCH VECTOR OFFSETS
0000    126          $WCBDEF                         ;DEFINE WCB OFFSETS
0000    127
0000    128 ;
0000    129 ; LOCAL MACROS
0000    130 ;
0000    131 ; CHECK FOR FATAL OR RETRIABLE ERROR
0000    132 ;
0000    133
0000    134          .MACRO  CHECK_ERROR
0000    135                  BSBW    CHECK_ERROR
0000    136          .ENDM   CHECK_ERROR
0000    137
0000    138 ;
0000    139 ; EXECUTE FUNCTION AND BRANCH ON RETRIABLE ERROR CONDITION
0000    140 ;
0000    141
0000    142          .MACRO  EXFUNC BDST,FCODE
0000    143                  .IF NB  FCODE
0000    144                  MOVZBL  #CD'FCODE,R0
0000    145                  .ENDC
0000    146                  BSBW    FEX
0000    147                  .WORD   BDST-.-2
0000    148          .ENDM   EXFUNC
0000    149
0000    150 ;
0000    151 ; GENERATE FUNCTION TABLE ENTRY AND CASE TABLE INDEX SYMBOL
0000    152 ;
0000    153
0000    154          .MACRO  GENF FCODE
0000    155                  CD'FCODE=.-FTAB
0000    156                  .BYTE   FCODE!MT_CS1_M_GO
0000    157          .ENDM   GENF
0000    158
0000    159 ;
0000    160 ; GENERATE ERROR MASK TABLE ENTRY
0000    161 ;
0000    162
0000    163          .MACRO  MASK LIST
0000    164 $.$=0
0000    165          .IRP    X,<LIST>
0000    166 $.$=$.$!MT_ER_M_'X
0000    167          .ENDM
0000    168          .WORD   $.$
0000    169          .ENDM   MASK
0000    170
0000    171 ;
```

```
                     0000   172 ; TEST IF ANY RETRIES REMAINING
                     0000   173 ;
                     0000   174
                     0000   175         .MACRO  TESTR BDST
                     0000   176                 BSBW    TESTR
                     0000   177                 .WORD   BDST-.-2
                     0000   178         .ENDM   TESTR
                     0000   179
                     0000   180 ;
                     0000   181 ; LOCAL SYMBOLS
                     0000   182
6D285010             0000   183 MEDIA_ID_TE16 = ^X<6D285010>            ; Media id for the TE16
6D29502D             0000   184 MEDIA_ID_TU45 = ^X<6D29502D>            ; Media id for the TU45
6D29504D             0000   185 MEDIA_ID_TU77 = ^X<6D29504D>            ; Media id for the TU77
                     0000   186
                     0000   187 ; TE16/TU77 MASSBUS REGISTER OFFSETS
                     0000   188 ;
                     0000   189
                     0000   190         $DEFINI MT
                     0000   191
                     0000   192 $DEF     MT_CS1              .BLKL   1       ;DRIVE CONTROL REGISTER
                     0004   193         _VIELD  MT_CS1,0,<-                 ; DRIVE CONTROL REGISTER BIT DEFINITIONS
                     0004   194                 <GO,,M>,-                   ; GO BIT
                     0004   195                 <FCODE,5>-                  ; FUNCTION CODE
                     0004   196                 >                           ;
                     0004   197 $DEF     MT_DS               .BLKL   1       ;DRIVE STATUS REGISTER
                     0008   198         _VIELD  MT_DS,0,<-                  ; DRIVE STATUS REGISTER BIT DEFINITIONS
                     0008   199                 <SLA,,M>,-                  ; SLAVE ATTENTION
                     0008   200                 <BOT,,M>,-                  ; BEGINNING OF TAPE
                     0008   201                 <TM,,M>,-                   ; TAPE MARK
                     0008   202                 <IDB,,M>,-                  ; IDENTIFICATION BURST
                     0008   203                 <SDWN,,M>,-                 ; SLOWING DOWN
                     0008   204                 <PES,,M>,-                  ; PHASE ENCODED
                     0008   205                 <SSC,,M>,-                  ; SLAVE STATUS CHANGE
                     0008   206                 <DRY,,M>,-                  ; DRIVE READY
                     0008   207                 <DPR,,M>,-                  ; DRIVE PRESENT
                     0008   208                 <,1>,-                      ; RESERVED BIT
                     0008   209                 <EOT,,M>,-                  ; END OF TAPE
                     0008   210                 <WRL,,M>,-                  ; DRIVE WRITE LOCKED
                     0008   211                 <MOL,,M>,-                  ; MEDIUM ONLINE
                     0008   212                 <PIP,,M>,-                  ; POSITIONING IN PROGRESS
                     0008   213                 <ERR,,M>,-                  ; COMPOSIT ERROR
                     0008   214                 <ATA,,M>-                   ; ATTENTION ACTIVE
                     0008   215                 >                           ;
                     0008   216 $DEF     MT_ER               .BLKL   1       ;ERROR REGISTER
                     000C   217         _VIELD  MT_ER,0,<-                  ; ERROR REGISTER BIT DEFINITIONS
                     000C   218                 <ICF,,M>,-                  ; ILLEGAL FUNCTION
                     000C   219                 <ILR,,M>,-                  ; ILLEGAL REGISTER
                     000C   220                 <RMR,,M>,-                  ; REGISTER MODIFY REFUSED
                     000C   221                 <CPAR,,M>,-                 ; CONTROL BUS PARITY ERROR
                     000C   222                 <FMT,,M>,-                  ; FORMAT ERROR
                     000C   223                 <DPAR,,M>,-                 ; DATA BUS PARITY ERROR
                     000C   224                 <VPE,,M>,-                  ; VERTICLE PARITY ERROR (NRZI)
                     000C   225                 <LRC,,M>,-                  ; LONGITUDINAL PARITY ERROR (NRZI)
                     000C   226                 <NSG,,M>,-                  ; NONSTANDARD GAP
                     000C   227                 <FCE,,M>,-                  ; FRAME COUNT ERROR
                     000C   228                 <ITM,,M>,-                  ; ILLEGAL TAPE MARK
```

```
                     000C   229                     <NEF,,M>,-              ; NONEXECUTABLE FUNCTION
                     000C   230                     <DTE,,M>,-              ; DRIVE TIMING ERROR
                     000C   231                     <OPI,,M>,-              ; OPERATION INCOMPLETE
                     000C   232                     <UNS,,M>,-              ; DRIVE UNSAFE
                     000C   233                     <CRC,,M>,-              ; CRC ERROR (NRZI)
                     000C   234             >                               ;
                     000C   235 ; Explicitly define alternate names for some bits in MT_ER.
                     000C   236
00000040             000C   237 MT_ER_M_INC=MT_ER_M_VPE
00000006             000C   238 MT_ER_V_INC=MT_ER_V_VPE
00000080             000C   239 MT_ER_M_PEF=MT_ER_M_LRC
00000007             000C   240 MT_ER_V_PEF=MT_ER_V_LRC
00000400             000C   241 MT_ER_M_CS=MT_ER_M_ITM
0000000A             000C   242 MT_ER_V_CS=MT_ER_V_ITM
00008000             000C   243 MT_ER_M_COR=MT_ER_M_CRC
0000000F             000C   244 MT_ER_V_COR=MT_ER_V_CRC
                     000C   245
                     000C   246 $DEF     MT_MR                   .BLKL   1       ;MAINTENANCE REGISTER
                     0010   247 $DEF     MT_AS                   .BLKL   1       ;ATTENTION SUMMARY REGISTER
                     0014   248 $DEF     MT_FC                   .BLKL   1       ;FRAME COUNT REGISTER
                     0018   249 $DEF     MT_DT                   .BLKL   1       ;DRIVE TYPE REGISTER
                     001C   250         _VIELD  MT_DT,0,<-                      ; DRIVE TYPE REGISTER FIELD DEFINITIONS
                     001C   251                 <DTN,9>,-                       ; DRIVE TYPE NUMBER
                     001C   252                 <,1>,-                          ; RESERVED BIT
                     001C   253                 <SPR,,M>,-                      ; SLAVE PRESENT
                     001C   254                 <DRQ,,M>,-                      ; DRIVE REQUEST REQUIRED (ALWAYS 0)
                     001C   255                 <7CH,,M>,-                      ; 7-CHANNEL TAPE (ALWAYS 0)
                     001C   256                 <MOH,,M>,-                      ; MOVING HEAD (ALWAYS 0)
                     001C   257                 <TAP,,M>,-                      ; TAPE DRIVE (ALWAYS 1)
                     001C   258         >                                       ;
                     001C   259 $DEF     MT_CC                   .BLKL   1       ;CHECK CHARACTER REGISTER
                     0020   260 $DEF     MT_SN                   .BLKL   1       ;SERIAL NUMBER REGISTER
                     0024   261 $DEF     MT_TC                   .BLKL   1       ;MAGTAPE CONTROL REGISTER
                     0028   262         _VIELD  MT_TC,0,<-                      ; TAPE CONTROL REGISTER FIELD DEFFINITIONS
                     0028   263                 <SSEL,3>,-                      ; SLAVE SELECT
                     0028   264                 <EPAR,1,M>,-                    ; EVEN PARITY
                     0028   265                 <FSEL,4,M>,-                    ; FORMAT SELECT
                     0028   266                 <DEN,3,M>,-                     ; DENSITY
                     0028   267                 <,1>,-                          ; RESERVED BIT
                     0028   268                 <EABO,,M>,-                     ; ENABLE ABORT ON TRANSFER ERROR
                     0028   269                 <TCW,,M>,-                      ; TAPE CONTROL WRITE
                     0028   270                 <FCS,,M>,-                      ; FRAME COUNT STATUS
                     0028   271                 <ACCL,,M>,-                     ; ACCELERATOR
                     0028   272         >                                       ;
                     0028   273
                     0028   274         $DEFEND MT
                     0000   275
                     0000   276 ;
                     0000   277 ; DEFINE DEVICE DEPENDENT UNIT CONTROL BLOCK OFFSETS
                     0000   278 ;
                     0000   279
                     0000   280         $DEFINI UCB
                     0000   281
                     0000   282         $VIELD  UCB,0,<-                        ;DEVICE DEPENDENT STATUS BITS
                     0000   283                 <MT_REWIND,,M>,-                ; REWIND IN PROGRESS
                     0000   284                 <MT_PRVMOL,,M>,-                ; PREVIOUS MEDIUM ONLINE STATE
                     0000   285                 <MT_PWRFL,,M>,-                 ; Currently in Powerfail recovery
```

```
                     0000   286                 <MT_CNCLP,,M>,-          ; Cancel Pending(looked at in Powerfail)
                     0000   287                 >                        ;
                     0000   288
000000B4             0000   289 .=UCB$K_LCL_TAPE_LENGTH
                     00B4   290
                     00B4   291 $DEF     UCB$L_MT_SR      .BLKL   1       ;SAVED MBA STATUS REGISTER
                     00B8   292 $DEF     UCB$W_MT_DS      .BLKW   1       ;SAVED DRIVE STATUS REGISTER
                     00BA   293 $DEF     UCB$W_MT_ER      .BLKW   1       ;SAVED DRIVE ERROR REGISTER
                     00BC   294 $DEF     UCB$W_MT_FC      .BLKW   1       ;SAVED DRIVE FRAME COUNT REGISTER
                     00BE   295 $DEF     UCB$W_MT_SPACNT  .BLKW   1       ;CURRENT SPACING COUNT
                     00C0   296 $DEF     UCB$W_MT_CS1     .BLKW   1       ;SAVED DRIVE CONTROL REGISTER
                     00C2   297 $DEF     UCB$W_MT_TC_SAV  .BLKW   1       ;SAVED TAPE CONTROL REGISTER
                     00C4   298 $DEF     UCB$W_MT_FORCNT  .BLKW   1       ;FORWARD SPACE COUNT DURING RETRY
                     00C6   299 $DEF     UCB$W_MT_TC      .BLKW   1       ;TAPE CONTROL REGISTER CONTENTS
                     00C8   300 $DEF     UCB$L_MT_PREVTM  .BLKL   1       ; Position of previous TAPEMARK, used
                     00CC   301                                           ;  in forward SKIPFILE and SPACEFILE
                     00CC   302                                           ;  operations in detecting consecutive
                     00CC   303                                           ;  TAPEMARKs
                     00CC   304 $DEF     UCB$L_MT_ORGPOS  .BLKL   1       ; Here store value of UCB$L_RECORD
                     00D0   305                                           ;  at STARTIO time.
                     00D0   306 $DEF     UCB$W_MT_CC_SAV  .BLKW   1       ; Space to save controller register.
000000D4             00D2   307                      .BLKW   1       ; RESERVED for now.
000000D4             00D4   308 UCB$K_MT_LENGTH=.                         ; Length of MT UCB.
                     00D4   309
                     00D4   310         $DEFEND UCB
                     0000   311
                     0000   312 ;
                     0000   313 ; MAXIMUM SPACING ON READ AND WRITECHECK ERRORS
                     0000   314 ;
                     0000   315
00000005             0000   316 ERR_SPACING=5                             ;FIVE RECORDS
                     0000   317
                     0000   318 ;
                     0000   319 ; HARDWARE FUNCTION CODES
                     0000   320 ;
                     0000   321
00000000             0000   322 F_NOP=0*2                                 ;NO OPERATION
00000000             0000   323 F_PACKACK=0*2                             ;PACK ACKNOWLEDGE
00000000             0000   324 F_SENSECHAR=0*2                           ;SENSE TAPE CHARACTERISTICS
00000000             0000   325 F_SETCHAR=0*2                             ;SET TAPE CHARACTERISTICS
00000002             0000   326 F_UNLOAD=1*2                              ;UNLOAD DRIVE
00000006             0000   327 F_REWIND=3*2                              ;REWIND
00000008             0000   328 F_DRVCLR=4*2                              ;DRIVE CLEAR
00000010             0000   329 F_READPRESET=8*2                          ;READ IN PRESET
00000014             0000   330 F_ERASE=10*2                              ;ERASE TAPE
00000016             0000   331 F_WRITEMARK=11*2                          ;WRITE TAPE MARK
00000018             0000   332 F_SPCFILFOR=12*2                          ;SPACE FILE FORWARD
0000001A             0000   333 F_SPCFILREV=13*2                          ;SPACE FILE REVERSE
00000018             0000   334 F_SPCRECFOR=12*2                          ;SPACE RECORD FORWARD
0000001A             0000   335 F_SPCRECREV=13*2                          ;SPACE RECORD REVERSE
00000028             0000   336 F_INTSPCFOR=20*2                          ;INTERNAL SPACE RECORD FORWARD
0000002E             0000   337 F_INTSPCREV=23*2                          ;INTERNAL SPACE RECORD REVERSE
00000028             0000   338 F_WRITECHECK=20*2                         ;WRITE CHECK DATA FORWARD
0000002E             0000   339 F_WRITECHECKR=23*2                        ;WRITE CHECK DATA REVERSE
00000030             0000   340 F_WRITE=24*2                              ;WRITE DATA FORWARD
00000030             0000   341 F_WRITEDATA=24*2                          ;WRITE DATA FORWARD
00000038             0000   342 F_READDATA=28*2                           ;READ DATA FORWARD
```

```
0000003E  0000    343 F_READDATAR=31*2                        ;READ DATA REVERSE
          0000    344
          0000    345 ;
          0000    346 ; MINIMUM RECORD SIZE
          0000    347 ;
          0000    348
0000000E  0000    349 MIN_RECORD=14                           ;FOURTEEN BYTES
          0000    350
          0000    351 ;
          0000    352 ; HARWARE DENSITY DEFINITIONS
          0000    353 ;
          0000    354
00000003  0000    355 NRZI=3                                  ;800 BPI
00000004  0000    356 PE=4                                    ;PHASE ENCODED
          0000    357
          0000    358 ;
          0000    359 ; ERROR COUNT THRESHOLD BEFORE ALTERNATE RECOVERY ATTEMPTED
          0000    360 ;
          0000    361
00000008  0000    362 THRESHOLD=8                             ;EIGHT RETRIES BEFORE ALTERNATE METHOD
          0000    363
          0000    364 ;
          0000    365 ; LOCAL DATA
          0000    366 ;
          0000    367 ; DRIVER PROLOGUE TABLE
          0000    368 ;
          0000    369
          0000    370         DPTAB   -                       ;DEFINE DRIVER PROLOGUE TABLE
          0000    371                 END=TM_END,-            ;END OF DRIVER
          0000    372                 FLAGS=DPT$M_SUBCNTRL,-  ;INDICATE SUBCONTROLLER
          0000    373                 ADAPTER=MBA,-           ;ADAPTER TYPE
          0000    374                 UCBSIZE=UCB$K_MT_LENGTH,- ;UCB SIZE
          0000    375                 NAME=TMDRIVER           ;DRIVER NAME
          0038    376         DPT_STORE INIT                  ;CONTROL BLOCK INIT VALUES
          0038    377         DPT_STORE DDB,DDB$L_ACPD,L,<^A\MTA\> ;DEFAULT ACP NAME
          003F    378         DPT_STORE UCB,UCB$B_FIPL,B,8    ;FORK IPL
          0043    379         DPT_STORE UCB,UCB$L_DEVCHAR,L,- ;DEVICE CHARACTERISTICS
          0043    380                 <DEV$M_FOD-             ; FILES ORIENTED
          0043    381                 !DEV$M_DIR-             ; DIRECTORY STRUCTURED
          0043    382                 !DEV$M_AVL-             ; AVAILABLE
          0043    383                 !DEV$M_ELG-             ; ERROR LOGGING ENABLED
          0043    384                 !DEV$M_IDV-             ; INPUT DEVICE
          0043    385                 !DEV$M_ODV-             ; OUTPUT DEVICE
          0043    386                 !DEV$M_SDI-             ; SINGLE DIRECTORY DEVICE
          0043    387                 !DEV$M_SQD>             ; SEQUENTIAL DEVICE
          004A    388         DPT_STORE UCB,UCB$L_DEVCHAR2,L,-; DEVICE CHARACTERISTICS
          004A    389                 <DEV$M_NNM>             ; PREFIX NAME WITH "node$"
          0051    390         DPT_STORE UCB,UCB$B_DEVCLASS,B,DC$_TAPE ;DEVICE CLASS
          0055    391         DPT_STORE UCB,UCB$W_DEVBUFSIZ,W,2048 ;DEFAULT BUFFER SIZE
          005A    392         DPT_STORE UCB,UCB$L_DEVDEPEND,W,<^X3C0> ;DEFAULT TAPE PARAMETERS
          005F    393         DPT_STORE UCB,UCB$B_DIPL,B,21   ;DEVICE IPL
          0063    394         DPT_STORE UCB,UCB$B_ERTCNT,B,16 ;ERROR RETRY COUNT
          0067    395         DPT_STORE UCB,UCB$B_ERTMAX,B,16 ;MAX ERROR RETRY COUNT
          006B    396         DPT_STORE UCB,UCB$W_MT_TC,W,<^X3C0> ;DEFAULT TAPE PARAMETERS
          0070    397         DPT_STORE REINIT                ;CONTROL BLOCK RE-INIT VALUES
          0070    398         DPT_STORE CRB,CRB$L_INTD+4,D,TM$INT ;INTERRUPT SERVICE ROUTINE ADDRESS
          0075    399         DPT_STORE CRB,CRB$L_INTD+VEC$L_UNITINIT,D,TM_TXXX_INIT ;UNIT INIT
```

```
                     007A   400          DPT_STORE DDB,DDB$L_DDT,D,TM$DDT ;DDT ADDRESS
                     007F   401          DPT_STORE END                    ;
           00000001  0000   402          .MDELETE DPT_STORE
                     0000   403
                     0000   404 ;
                     0000   405 ; DRIVER DISPATCH TABLE
                     0000   406 ;
                     0000   407
                     0000   408          DDTAB   TM,-                     ;DRIVER DISPATCH TABLE
                     0000   409                  TM_STARTIO,-             ;START I/O OPERATION
                     0000   410                  TM_UNSOLNT,-             ;UNSOLICITED INTERRUPT
                     0000   411                  TM_FUNCTABLE,-           ;FUNCTION DECISION TABLE
                     0000   412                  TM_CANCELIO,-            ;CANCEL I/O ENTRY POINT
                     0000   413                  TM_REGDUMP,-             ;REGISTER DUMP ROUTINE
                     0000   414                  <<MT_TC+4+MBA$L_BCR+4+8>+<<3+5+1>*4>>,- ;DIAGNOSTIC BUFFER SIZE
                     0000   415                  <<MT_TC+4+MBA$L_BCR+4+8>+<1*4>+<EMB$L_DV_REGSAV>> ;ERROR BUFFER SIZE
                     0038   416
                     0038   417 ;
                     0038   418 ; DENSITY CODE TRANSLATION TABLE
                     0038   419 ;
                     0038   420 ;       DENSITY CODES ARE TRANSLATED BY TAKING THE FIVE BIT ENCODED DENSITY
                     0038   421 ;       VALUE, MULTIPLYING BY FOUR TO FORM THE STARTING BIT NUMBER, AND THEN
                     0038   422 ;       EXTRACTING THE APPROPRIATE DENSITY CODE FROM THE TRANSLATION TABLE.
                     0038   423 ;
                     0038   424
                     0038   425 DENSITY:                                  ;
           33343333  0038   426          .LONG   ^X33343333               ;DENSITY CODES 0-7
           33333333  003C   427          .LONG   ^X33333333               ;DENSITY CODES 8-15
           33333333  0040   428          .LONG   ^X33333333               ;DENSITY CODES 16-23
           33333333  0044   429          .LONG   ^X33333333               ;DENSITY CODES 24-31
                     0048   430
                     0048   431 ;
                     0048   432 ; TXXX DRIVE TYPE DESCRIPTOR TABLE
                     0048   433 ;
                     0048   434
                     0048   435 TM_DTDESC:                                ;
               0029  0048   436          .WORD   ^X29                     ;TE16 45 IPS
                 01  004A   437          .BYTE   DT$_TE16                 ;
           00000003  004B   438 TM_DTDESCLEN=.-TM_DTDESC                  ;LENGTH OF DRIVE TYPE DESCRIPTOR
               002A  004B   439          .WORD   ^X2A                     ;TU45 45 IPS
                 02  004D   440          .BYTE   DT$_TU45                 ;
               002C  004E   441          .WORD   ^X2C                     ;TU77 125 IPS
                 03  0050   442          .BYTE   DT$_TU77                 ;
               0000  0051   443          .WORD   0                        ;END OF TABLE
           00000056  0053   444          .BLKB   TM_DTDESCLEN             ;SPARE DRIVE TYPE SLOT
                     0056   445
                     0056   446 ;
                     0056   447 ; HARDWARE I/O FUNCTION CODE TABLE
                     0056   448 ;
                     0056   449
                     0056   450 FTAB:                                    ;
                     0056   451          GENF    F_NOP                    ;NO OPERATION
                     0057   452          GENF    F_UNLOAD                 ;UNLOAD VOLUME
                     0058   453          GENF    F_SPCFILFOR              ;SPACE FILE FORWARD
                     0059   454          GENF    F_REWIND                 ;REWIND
                     005A   455          GENF    F_DRVCLR                 ;DRIVE CLEAR
                     005B   456          GENF    F_SPCFILREV              ;SPACE FILE REVERSE
```

```
                     005C   457          GENF    F_ERASE                 ;ERASE TAPE
                     005D   458          GENF    F_SPCRECREV             ;SPACE RECORD REVERSE
                     005E   459          GENF    F_PACKACK               ;PACK ACKNOWLEDGE
                     005F   460          GENF    F_SPCRECFOR             ;SPACE RECORD FORWARD
                     0060   461          GENF    F_WRITECHECK            ;WRITE CHECK FORWARD
                     0061   462          GENF    F_WRITEDATA             ;WRITE DATA FORWARD
                     0062   463          GENF    F_READDATA              ;READ DATA FORWARD
                     0063   464          GENF    F_WRITECHECKR           ;WRITE CHECK REVERSE
                     0064   465          GENF    F_WRITE                 ;WRITE DATA FORWARD
                     006C   466          GENF    F_READDATAR             ;READ DATA REVERSE
                     0066   467          GENF    F_READPRESET            ;READ IN PRESET
                     0067   468          GENF    F_SETCHAR               ;SET TAPE CHARACTERISTICS
                     0068   469          GENF    F_SENSECHAR             ;SENSE TAPE CHARACTERISTICS
                     0069   470          GENF    F_WRITEMARK             ;WRITE TAPE MARK
                     006A   471          GENF    F_INTSPCFOR             ;INTERNAL SPACE RECORD FORWARD
                     006B   472          GENF    F_INTSPCREV             ;INTERNAL SPACE RECORD REVERSE
                     006C   473
                     006C   474 ;
                     006C   475 ; FORMAT CODE TRANSLATION TABLE
                     006C   476 ;
                     006C   477 ;       FORMAT CODES ARE TRANSLATED BY TAKING THE FOUR BIT ENCODED FORMAT VALUE,
                     006C   478 ;       MULTIPLYING BY FOUR TO FORM THE STARTING BIT NUMBER, AND THEN EXTRACTING
                     006C   479 ;       THE APPROPRIATE FORMAT CODE FROM THE TRANSLATION TABLE.
                     006C   480 ;
                     006C   481
                     006C   482 FORMAT:                                    ;
CCCCCCCC             006C   483          .LONG   ^XCCCCCCCC              ;FORMAT CODES 0-7
CEDCCCCC             0070   484          .LONG   ^XCEDCCCCC              ;FORMAT CODES 8-15
                     0074   485
                     0074   486 ;
                     0074   487 ; FUNCTION TIME OUT TABLE
                     0074   488 ;
                     0074   489
                     0074   490 TIME_OUT:                                 ;
00000000             0074   491          .LONG   0                       ;NO OPERATION
00000006             0078   492          .LONG   6                       ;UNLOAD VOLUME
000002D0             007C   493          .LONG   60*12                   ; Space File Forward
000001A4             0080   494          .LONG   60*7                    ;REWIND
00000000             0084   495          .LONG   0                       ;DRIVE CLEAR
000C02D0             0088   496          .LONG   60*12                   ; Space File Reverse
00000006             008C   497          .LONG   6                       ;ERASE TAPE
000002D0             0090   498          .LONG   60*12                   ; Space Record Reverse
00000000             0094   499          .LONG   0                       ;PACK ACKNOWLEDGE
000002D0             0098   500          .LONG   60*12                   ; Space Record Forward
0000000C             009C   501          .LONG   12                      ;WRITE CHECK DATA FORWARD
0000000C             00A0   502          .LONG   12                      ;WRITE DATA FORWARD
0000000C             00A4   503          .LONG   12                      ;READ DATA FORWARD
0000000C             00A8   504          .LONG   12                      ;WRITE CHECK DATA REVERSE
0000000C             00AC   505          .LONG   12                      ;WRITE DATA FORWARD
0000000C             00B0   506          .LONG   12                      ;READ DATA REVERSE
00C001A4             00B4   507          .LONG   60*7                    ;READ IN PRESET
00000000             00B8   508          .LONG   0                       ;SET TAPE CHARACTERISTICS
00000000             00BC   509          .LONG   0                       ;SENSE TAPE CHARACTERISITCS
00000006             00C0   510          .LONG   6                       ;WRITE TAPE MARK
0000000C             00C4   511          .LONG   12                      ;INTERNAL SPACE RECORD FORWARD
0000000C             00C8   512          .LONG   12                      ;INTERNAL SPACE RECORD REVERSE
                     00CC   513
```

```
                        ;
                        ; DON'T CARE ERROR MASK TABLE
                        ;
                        ; THIS TABLE CONTAINS A MASK OF THE ERROR BITS THAT ARE TO BE IGNORED FOR EACH
                        ; FUNCTION WHEN EXAMINING THE DRIVE ERROR REGISTER.
                        ;

                        XTAB:                                   ;
00CC    522                     MASK    <FMT,DPAR,INC,PEF,NSG,FCE,CS,DTE,OPI,COR> ;NO-OP
00CE    523                     MASK    <FMT,DPAR,INC,PEF,NSG,FCE,CS,DTE,OPI,COR> ;UNLOAD
00D0    524                     MASK    <FMT,DPAR,INC,PEF,NSG,CS,DTE,COR> ;SPACE FILE FORWARD
00D2    525                     MASK    <FMT,DPAR,INC,PEF,NSG,FCE,CS,DTE,OPI,COR> ;REWIND
00D4    526                     MASK    <FMT,DPAR,INC,PEF,NSG,FCE,CS,NEF,DTE,OPI,COR> ;DRIVE CLEAR
00D6    527                     MASK    <FMT,DPAR,INC,PEF,NSG,CS,DTE,COR> ;SPACE FILE REVERSE
00D8    528                     MASK    <FMT,DPAR,INC,PEF,NSG,FCE,DTE,OPI,COR> ;ERASE
00DA    529                     MASK    <FMT,DPAR,INC,PEF,NSG,CS,DTE,COR> ;SPACE RECORD REVERSE
00DC    530                     MASK    <FMT,DPAR,INC,PEF,NSG,FCE,CS,DTE,OPI,COR> ;PACK ACKNOWLEDGE
00DE    531                     MASK    <FMT,DPAR,INC,PEF,NSG,CS,DTE,COR> ;SPACE RECORD FORWARD
00E0    532                     MASK    <DPAR>                  ;WRITE CHECK FORWARD
00E2    533                     MASK    <>                      ;WRITE DATA FORWARD
00E4    534                     MASK    <DPAR,FCE>              ;READ DATA FORWARD
00E6    535                     MASK    <DPAR>                  ;WRITE CHECK REVERSE
00E8    536                     MASK    <>                      ;WRITE DATA FORWARD
00EA    537                     MASK    <DPAR,FCE>              ;READ DATA REVERSE
00EC    538                     MASK    <FMT,DPAR,INC,PEF,NSG,FCE,CS,DTE,OPI,COR> ;READIN PRESET
00EE    539                     MASK    <FMT,DPAR,INC,PEF,NSG,FCE,CS,DTE,OPI,COR> ;SET CHARACTERISTICS
00F0    540                     MASK    <FMT,DPAR,INC,PEF,NSG,FCE,CS,DTE,OPI,COR> ;SENSE CHARACTERISTICS
00F2    541                     MASK    <FMT,DPAR,FCE,DTE,COR>  ;WRITE TAPE MARK
00F4    542                     MASK    <DPAR,INC,PEF,FCE,CS,DTE,COR> ;INTERNAL SPACE RECORD FORWARD
00F6    543                     MASK    <DPAR,INC,PEF,FCE,CS,DTE,COR> ;INTERNAL SPACE RECORD REVERSE
```

```
                     .SBTTL  TE16/TU77 FUNCTION DECISION TABLE
                ;+
                ; TE16/TU77 FUNCTION DECISION TABLE
                ;-

                TM_FUNCTABLE:                                   ;FUNCTION DECISION TABLE
                        FUNCTAB ,-                              ;LEGAL FUNCTIONS
                                <NOP,-                          ;NO OPERATION
                                 UNLOAD,-                       ;UNLOAD VOLUME
                                 SPACERECORD,-                  ;SPACE RECORDS
                                 RECAL,-                        ;RECALIBRATE (REWIND)
                                 DRVCLR,-                       ;DRIVE CLEAR
                                 READPRESET,-                   ;READ IN PRESET
                                 PACKACK,-                      ;PACK ACKNOWLEDGE
                                 ERASETAPE,-                    ;ERASE TAPE
                                 SENSECHAR,-                    ;SENSE TAPE CHARACTERISTICS
                                 SETCHAR,-                      ;SET CHARACTERISTICS
                                 SPACEFILE,-                    ;SPACE FILE
                                 WRITECHECK,-                   ;WRITE CHECK FORWARD
                                 WRITEPBLK,-                    ;WRITE PHYSICAL BLOCK
                                 READPBLK,-                     ;READ PHYSICAL BLOCK
                                 WRITEMARK,-                    ;WRITE TAPE MARK
                                 AVAILABLE,-                    ;AVAILABLE (REWIND/NOWAIT CLEAR VALID)
                                 READLBLK,-                     ;READ LOGICAL BLOCK
                                 WRITELBLK,-                    ;WRITE LOGICAL BLOCK
                                 SENSEMODE,-                    ;SENSE TAPE MODE
                                 SETMODE,-                      ;SET MODE
                                 REWIND,-                       ;REWIND
                                 REWINDOFF,-                    ;REWIND AND SET OFFLINE
                                 SKIPRECORD,-                   ;SKIP RECORDS
                                 SKIPFILE,-                     ;SKIP FILES
                                 WRITEOF,-                      ;WRITE END OF FILE
                                 READVBLK,-                     ;READ VIRTUAL BLOCK
                                 WRITEVBLK,-                    ;WRITE VIRTUAL BLOCK
                                 ACCESS,-                       ;ACCESS FILE AND/OR FIND DIRECTORY ENTRY
                                 ACPCONTROL,-                   ;ACP CONTROL FUNCTION
                                 CREATE,-                       ;CREATE FILE AND/OR CREATE DIRECTORY ENTRY
                                 DEACCESS,-                     ;DEACCESS FILE
                                 DELETE,-                       ;DELETE FILE AND/OR DIRECTORY ENTRY
                                 MODIFY,-                       ;MODIFY FILE ATTRIBUTES
                                 MOUNT>                         ;MOUNT VOLUME
                        FUNCTAB ,-                              ;BUFFERED I/O FUNCTIONS
                                <NOP,-                          ;NO OPERATION
                                 UNLOAD,-                       ;UNLOAD VOLUME
                                 SPACERECORD,-                  ;SPACE RECORDS
                                 RECAL,-                        ;RECALIBRATE (REWIND)
                                 DRVCLR,-                       ;DRIVE CLEAR
                                 READPRESET,-                   ;READ IN PRESET
                                 PACKACK,-                      ;PACK ACKNOWLEDGE
                                 ERASETAPE,-                    ;ERASE TAPE
                                 SENSECHAR,-                    ;SENSE CHARACTERISTICS
                                 SETCHAR,-                      ;SET CHARACTERISTICS
                                 SPACEFILE,-                    ;SPACE FILES
                                 WRITEMARK,-                    ;WRITE TAPE MARK
                                 SENSEMODE,-                    ;SENSE MODE
                                 SETMODE,-                      ;SET MODE
                                 REWIND,-                       ;REWIND
```

```
0100   602                    REWINDOFF,-             ;REWIND AND UNLOAD
0100   603                    SKIPRECORD,-            ;SKIP RECORDS
0100   604                    SKIPFILE,-              ;SKIP FILES
0100   605                    WRITEOF,-               ;WRITE END OF FILE
0100   606                    ACCESS,-                ;ACCESS FILE AND/OR FIND DIRECTORY ENTRY
0100   607                    ACPCONTROL,-            ;ACP CONTROL FUNCTION
0100   608                    CREATE,-                ;CREATE FILE AND/OR CREATE DIRECTORY ENTRY
0100   609                    DEACCESS,-              ;DEACCESS FILE
0100   610                    DELETE,-                ;DELETE FILE AND/OR DIRECTORY ENTRY
0100   611                    MODIFY,-                ;MODIFY FILE ATTRIBUTES
0100   612                    MOUNT>                  ;MOUNT VOLUME
0108   613            FUNCTAB +ACP$READBLK,-          ;READ FUNCTIONS
0108   614                   <READLBLK,-              ;READ LOGICAL BLOCK FORWARD
0108   615                    READPBLK,-              ;READ PHYSICAL BLOCK FORWARD
0108   616                    READVBLK>               ;READ VIRTUAL BLOCK
0114   617            FUNCTAB +ACP$WRITEBLK,-         ;WRITE FUNCTIONS
0114   618                   <WRITECHECK,-            ;WRITE CHECK FORWARD
0114   619                    WRITELBLK,-             ;WRITE LOGICAL BLOCK
0114   620                    WRITEPBLK,-             ;WRITE PHYSICAL BLOCK
0114   621                    WRITEVBLK>              ;WRITE VIRTUAL BLOCK
0120   622            FUNCTAB +ACP$ACCESS,<ACCESS,CREATE> ;ACCESS AND CREATE FILE OR DIRECTORY
012C   623            FUNCTAB +ACP$DEACCESS,<DEACCESS> ;DEACCESS FILE
0138   624            FUNCTAB +ACP$MODIFY,-           ;
0138   625                   <ACPCONTROL,-            ;ACP CONTROL FUNCTION
0138   626                    DELETE,-                ;DELETE FILE OR DIRECTORY ENTRY
0138   627                    MODIFY>                 ;MODIFY FILE ATTRIBUTES
0144   628            FUNCTAB +ACP$MOUNT,<MOUNT>      ;MOUNT VOLUME
0150   629            FUNCTAB +MT$CHECK_ACCESS,-      ;MAGTAPE CHECK ACCESS FUNCTIONS
0150   630                   <ERASETAPE,-             ;ERASE TAPE
0150   631                    WRITEMARK,-             ;WRITE TAPE MARK
0150   632                    WRITEOF>                ;WRITE END OF FILE
015C   633            FUNCTAB +EXE$ZEROPARM,-         ;ZERO PARAMETER FUNCTIONS
015C   634                   <NOP,-                   ;NO OPERATION
015C   635                    UNLOAD,-                ;UNLOAD VOLUME
015C   636                    RECAL,-                 ;RECALIBRATE (REWIND)
015C   637                    REWIND,-                ;REWIND
015C   638                    REWINDOFF,-             ;REWIND AND SET OFFLINE
015C   639                    DRVCLR,-                ;DRIVE CLEAR
015C   640                    READPRESET,-            ;READ IN PRESET
015C   641                    PACKACK,-               ;PACK ACKNOWLEDGE
015C   642                    ERASETAPE,-             ;ERASE TAPE
015C   643                    SENSECHAR,-             ;SENSE TAPE CHARACTERISTICS
015C   644                    SENSEMODE,-             ;SENSE TAPE MODE
015C   645                    WRITEMARK,-             ;WRITE TAPE MARK
015C   646                    AVAILABLE,-             ;AVAILABLE (REWIND/NOWAIT CLEAR VALID)
015C   647                    WRITEOF>                ;WRITE END OF FILE
0168   648            FUNCTAB +EXE$ONEPARM,-          ;ONE PARAMETER FUNCTIONS
0168   649                   <SPACERECORD,-           ;SPACE RECORDS
0168   650                    SPACEFILE,-             ;SPACE FILES
0168   651                    SKIPRECORD,-            ;SKIP RECORDS
0168   652                    SKIPFILE>               ;SKIP FILES
0174   653            FUNCTAB +EXE$SETMODE,-          ;SET TAPE CHARACTERISTICS
0174   654                   <SETCHAR,-               ;
0174   655                    SETMODE>                ;
```

```
                     0180   657          .SBTTL  CANCEL I/O ON CHANNEL
                     0180   658 ;+
                     0180   659 ; TM_CANCELIO - CANCEL I/O ON CHANNEL
                     0180   660 ;
                     0180   661 ; THIS ROUTINE IS CALLED WHEN THE LAST CHANNEL ASSIGNED TO A DEVICE IS DEASSIGNED,
                     0180   662 ; THE DEVICE IS DEALLOCATED, AND WHEN THE CANCEL I/O ON CHANNEL SYSTEM SERVICE IS
                     0180   663 ; EXECUTED.
                     0180   664 ;
                     0180   665 ; INPUTS:
                     0180   666 ;
                     0180   667 ;       R2 = NEGATIVE CHANNEL NUMBER.
                     0180   668 ;       R3 = ADDRESS OF CURRENT I/O REQUEST PACKET.
                     0180   669 ;       R4 = CURRENT PROCESS PCB ADDRESS.
                     0180   670 ;       R5 = DEVICE UCB ADDRESS.
                     0180   671 ;
                     0180   672 ; OUTPUTS:
                     0180   673 ;
                     0180   674 ;       IF THE DEVICE IS CURRENTLY BUSY, DOING A REWIND, AND IN A WAITFOR INTERRUPT
                     0180   675 ;       STATE, THEN THE REWIND FUNCTION IS CANCELLED.
                     0180   676 ;-
                     0180   677
                     0180   678 TM_CANCELIO:                                ;CANCEL I/O ON CHANNEL
  00000000'GF   16   0180   679          JSB     G^IOC$CANCELIO              ;TEST IF FUNCTION SHOULD BE CANCELLED
6F 64 A5    03   E1 0186   680          BBC     #UCB$V_CANCEL,UCB$W_STS(R5),30$ ;IF CLR, NO CANCEL PENDING
                     018B   681          DSBINT                              ;DISABLE INTERRUPTS
        02   E1   0191   682          BBC     #UCB$V_MT_PWRFL,-           ; See if currently repositioning tape
      31 68 A5          0193   683                  UCB$W_DEVSTS(R5),5$     ;  due to POWERFAIL.  If NOT, branch around
        08   A8   0196   684          BISW    #UCB$M_MT_CNCLP,-           ; If SO, set CANCEL PENDING flag on.
         68 A5          0198   685                  UCB$W_DEVSTS(R5)        ;  This bit is needed only if the driver
                     019A   686                                              ;   thread in POWERFAIL is currently
                     019A   687                                              ;   not waiting for an interrupt.  That
                     019A   688                                              ;   would imply that the driver thread
                     019A   689                                              ;   is on a resource wait queue and
                     019A   690                                              ;   would have to abort the operation
                     019A   691                                              ;   itself.
                     019A   692
        03   B3   019A   693          BITW    #UCB$M_INT!UCB$M_TIM,-
         64 A5          019C   694                  UCB$W_STS(R5)           ; Interrupt or timeout expected?
        57   13   019E   695          BEQL    20$                         ; If NOT, branch around and let CANCEL
                     01A0   696                                              ;  PENDING flag advise repositioning
                     01A0   697                                              ;  thread of the need to ABORT operation.
                     01A0   698
        0C   AA   01A0   699          BICW    #UCB$M_MT_PWRFL!UCB$M_MT_CNCLP,-
         68 A5          01A2   700                  UCB$W_DEVSTS(R5)        ; Here we are awaiting interrupt; so
                     01A4   701                                              ;  we can safely kill the driver thread.
                     01A4   702
     50   24 A5   D0   01A4   703          MOVL    UCB$L_CRB(R5),R0            ; R0 => CRB.
                     01A8   704          ASSUME  IDB$L_CSR  EQ  0
     50   2C B0   D0   01A8   705          MOVL    @CRB$L_INTD+VEC$L_IDB(R0),R0    ; R0 => TM03 CSR.
                     01AC   706
        24 A0   DD   01AC   707          PUSHL   MT_TC(R0)                       ; Save currently selected drive.
24 A0   00C6 C5   3C   01AF   708          MOVZWL  UCB$W_MT_TC(R5),MT_TC(R0)       ; Select this drive.
     51   04 A0   D0   01B5   709          MOVL    MT_DS(R0),R1                    ; Get this drive's status.
        24 A0 8ED0   01B9   710          POPL    MT_TC(R0)                       ; Restore selected drive.
                     01BD   711
   1F 51   0D   E0   01BD   712          BBS     #MT_DS_V_PIP,R1,13$         ; Branch if tape is moving and thereby
                     01C1   713                                              ;  assume that it is rewinding.
```

```
                01   AA  01C1   714          BICW    #UCB$M_MT_REWIND,-      ; Else since it is not moving, clear
              68 A5         01C3   715                  UCB$W_DEVSTS(R5)        ;  rewind in progress flag and
                1D   11     01C5   716          BRB     16$                     ; Branch around.
                            01C7   717 5$:
        0093 C5     03   91 01C7   718          CMPB    #CDF_REWIND,UCB$B_CEX(R5) ;REWIND IN PROGRESS?
                    0C   13 01CC   719          BEQL    10$                     ;IF EQL YES
        0093 C5     10   91 01CE   720          CMPB    #CDF_READPRESET,UCB$B_CEX(R5) ;READIN PRESET?
                    05   13 01D3   721          BEQL    10$                     ; If EQL yes.
                    00   E1 01D5   722          BBC     #UCB$V_MT_REWIND,-      ; Is a REWIND in progress?
                1D 68 A5         01D7   723                  UCB$W_DEVSTS(R5),20$    ;  If NOT, branch around.
          64 A5     03   B3 01DA   724 10$:     BITW    #UCB$M_INT!UCB$M_TIM,UCB$W_STS(R5) ;INTERRUPT OR TIMEOUT EXPECTED?
                    17   13 01DE   725          BEQL    20$                     ;IF EQL NO
                            01E0   726 13$:
          68 A5     01   A8 01E0   727          BISW    #UCB$M_MT_REWIND,UCB$W_DEVSTS(R5) ;SET REWIND IN PROGRESS
                            01E4   728 16$:
                    03   AA 01E4   729          BICW    #UCB$M_INT!UCB$M_TIM,-
              64 A5         01E6   730                  UCB$W_STS(R5)           ; Clear interrupt and timeout expected
                            01E8   731                                          ;  and thereby kill driver thread.
                            01E8   732          SETIPL  UCB$B_FIPL(R5)          ;LOWER TO FORK LEVEL
                50  2C   3C 01EC   733          MOVZWL  #SS$_ABORT,R0           ;SET ABORT STATUS
                    54   DD 01EF   734          PUSHL   R4                      ;SAVE CURRENT PROCESS PCB ADDRESS
                  0940   30 01F1   735          BSBW    STSXIT                  ;FINISH I/O OPERATION
                    54 8ED0 01F4   736          POPL    R4                      ;RESTORE CURRENT PROCESS PCB ADDRESS
                            01F7   737 20$:     ENBINT                          ;LOWER TO FORK LEVEL
                        05  01FA   738 30$:     RSB                             ;
```

```
                     01FB   740          .SBTTL  START I/O OPERATION
                     01FB   741 ;+
                     01FB   742 ; TM_STARTIO - START I/O OPERATION ON DEVICE UNIT
                     01FB   743 ;
                     01FB   744 ; THIS ENTRY POINT IS ENTERED TO START AN I/O OPERATION ON A DEVICE UNIT.
                     01FB   745 ;
                     01FB   746 ; INPUTS:
                     01FB   747 ;
                     01FB   748 ;       R3 = ADDRESS OF I/O PACKET.
                     01FB   749 ;       R5 = UCB ADDRESS OF DEVICE UNIT.
                     01FB   750 ;
                     01FB   751 ; OUTPUTS:
                     01FB   752 ;
                     01FB   753 ;       FUNCTION DEPENDENT PARAMETERS ARE STORED IN THE DEVICE UCB, THE ERROR
                     01FB   754 ;       RETRY COUNT IS RESET, AND THE FUNCTION IS EXECUTED. AT FUNCTION COMPLETION
                     01FB   755 ;       THE OPERATION IS TERMINATED THROUGH REQUEST COMPLETE.
                     01FB   756 ;-
                     01FB   757
                     01FB   758 TM_STARTIO:                                  ;START I/O OPERATION
          00B0 C5    D0   01FB   759          MOVL    UCB$L_RECORD(R5),-      ; Remember tape position at STARTIO.
          00CC C5         01FF   760                  UCB$L_MT_ORGPOS(R5)
0080 C5   0081 C5    90   0202   761          MOVB    UCB$B_ERTMAX(R5),UCB$B_ERTCNT(R5) ;INITIALIZE ERROR RETRY COUNT
  009A C5   20 A3    B0   0209   762          MOVW    IRP$W_FUNC(R3),UCB$W_FUNC(R5) ;SAVE FUNCTION CODE AND MODIFIERS
    00BE C5    01    AE   020F   763          MNEGW   #1,UCB$W_MT_SPACNT(R5)  ;SET DEFAULT SPACING COUNT
       50    38 A3    D0   0214   764          MOVL    IRP$L_MEDIA(R3),R0      ;GET PARAMETER LONGWORD
                     0218   765
                     0218   766 ;
                     0218   767 ; MOVE FUNCTION DEPENDENT PARAMETERS TO UCB
                     0218   768 ;
                     0218   769
          06    00   EF   0218   770          EXTZV   #IRP$V_FCODE,#IRP$S_FCODE,- ;EXTRACT I/O FUNCTION CODE
    51    20 A3         021B   771                  IRP$W_FUNC(R3),R1       ;
          51    02    D1   021E   772          CMPL    #IO$_SPACEFILE,R1       ;SPACE FILE FUNCTION?
                2B    13   0221   773          BEQL    10$                     ;IF EQL YES
          51    09    D1   0223   774          CMPL    #IO$_SPACERECORD,R1     ;SPACE RECORD FUNCTION?
                36    13   0226   775          BEQL    20$                     ;IF EQL YES
          51    1A    D1   0228   776          CMPL    #IO$_SETCHAR,R1         ;SET CHARACTERISTICS FUNCTION?
                55    13   022B   777          BEQL    50$                     ;IF EQL YES
          51    11    D1   022D   778          CMPL    #IO$_AVAILABLE,R1       ;AVAILABLE function?
                69    13   0230   779          BEQL    75$                     ;IF EQL YES
          51    0D    D1   0232   780          CMPL    #IO$_READPBLK+1,R1      ;DISJOINT FUNCTION CODE?
                7B    1A   0235   781          BGTRU   100$                    ;IF GTRU NO
                           0237   782          CASE    R1,<-                   ;DISPATCH LOGICAL FUNCTIONS
                           0237   783                  70$,-                   ;REWIND AND SET OFFLINE
                           0237   784                  60$,-                   ;SET MODE
                           0237   785                  80$,-                   ;REWIND
                           0237   786                  10$,-                   ;SKIP FILE
                           0237   787                  20$,-                   ;SKIP RECORD
                           0237   788                  90$,-                   ;SENSE TAPE MODE
                           0237   789                  90$,-                   ;WRITE EOF
                           0237   790                  >,LIMIT=#IO$_REWINDOFF ;
          51    09    A2   0249   791          SUBW    #IO$_READPRESET-IO$_READPBLK-4,R1 ;CONVERT TO DENSE FUNCTION CODE
                72    11   024C   792          BRB     110$                    ;
                           024E   793
                           024E   794 ;
                           024E   795 ; SPACE FILE FUNCTION - SET SPACE COUNT AND PROPER FUNCTION
                           024E   796 ;
```

```
                     024E   797
      51   02   9A  024E   798 10$:    MOVZBL  #CDF_SPCFILFOR,R1       ;SET FOR SPACE FILE FORWARD
   00BE C5   B6   0251   799           INCW    UCB$W_MT_SPACNT(R5)     ;SET DEFAULT SPACING COUNT TO LARGEST VALUE
      50   B5   0255   800           TSTW    R0                      ;SPACE FILE FORWARD?
      1A   14   0257   801           BGTR    40$                     ;IF GTR YES
      51   05   9A   0259   802           MOVZBL  #CDF_SPCFILREV,R1       ;SET FOR SPACE FILE REVERSE
      12   11   025C   803           BRB     30$                     ;
                     025E   804
                     025E   805 ;
                     025E   806 ; SPACE RECORD FUNCTION - SET SPACE COUNT AND PROPER FUNCTION
                     025E   807 ;
                     025E   808
      51   09   9A   025E   809 20$:    MOVZBL  #CDF_SPCRECFOR,R1       ;SET FOR SPACE RECORD FORWARD
   00BE C5   50   AE   0261   810           MNEGW   R0,UCB$W_MT_SPACNT(R5)  ;SET SPACING COUNT
      0B   19   0266   811           BLSS    40$                     ;IF LSS SPACE FORWARD FUNCTION
      51   07   9A   0268   812           MOVZBL  #CDF_SPCRECREV,R1       ;SET FOR SPACE RECORD REVERSE
   00BE C5   50   B0   026B   813           MOVW    R0,UCB$W_MT_SPACNT(R5)  ;SET SPACING COUNT
      50   50   AE   0270   814 30$:    MNEGW   R0,R0                   ;CONVERT TO POSITIVE COUNT
      7C A5   50   B0   0273   815 40$:    MOVW    R0,UCB$W_BOFF(R5)       ;SET SPACE COUNT
      7E A5   50   B0   0277   816           MOVW    R0,UCB$W_BCNT(R5)       ;SET SPACE COUNT
      43   12   027B   817           BNEQ    110$                    ;IF NEQ SPACING REQUIRED
      51   00   9A   027D   818           MOVZBL  #CDF_NOP,R1             ;SET FOR NO OPERATION
      3E   11   0280   819           BRB     110$                    ;
                     0282   820
                     0282   821 ;
                     0282   822 ; SET CHARACTERISTICS FUNCTION - STORE NEW TAPE CHARACTERISTICS
                     0282   823 ;
                     0282   824
   40 A5    38 A3   B0   0282   825 50$:    MOVW    IRP$L_MEDIA(R3),UCB$B_DEVCLASS(R5) ;SET NEW DEVICE CLASS AND TYPE
                     0287   826
                     0287   827 ;
                     0287   828 ; SET MODE FUNCTION - STORE NEW TAPE MODE
                     0287   829 ;
                     0287   830
   42 A5    3A A3   B0   0287   831 60$:    MOVW    IRP$L_MEDIA+2(R3),UCB$W_DEVBUFSIZ(R5) ;SET NEW DEFAULT BUFFER SIZE
   7C A5    3C A3   B0   028C   832           MOVW    IRP$L_MEDIA+4(R3),UCB$W_BOFF(R5) ;SAVE NEW TAPE CONTROL PARAMETERS
      51   11   9A   0291   833           MOVZBL  #CDF_SETCHAR,R1         ;SET FUNCTION DISPATCH INDEX
      2A   11   0294   834           BRB     110$                    ;
                     0296   835
                     0296   836 ;
                     0296   837 ; LOGICAL REWIND AND SET TAPE OFFLINE - CONVERT TO UNLOAD FUNCTION
                     0296   838 ;
                     0296   839
      51   01   9A   0296   840 70$:    MOVZBL  #CDF_UNLOAD,R1          ;SET FOR UNLOAD FUNCTION
      25   11   0299   841           BRB     110$                    ;
                     029B   842
                     029B   843 ;
                     029B   844 ; AVAILABLE FUNCTION - Equivalent of REWIND(NOWAIT) and clear of UCB$M_VALID.
                     029B   845 ;
                     029B   846
                     029B   847 75$:
   00A4 8F   B0   029B   848           MOVW    #IO$_REWIND!IO$M_NOWAIT,-; Simulate a REWIND NOWAIT.
   009A C5        029F   849                   UCB$B_FUNC(R5)
   0800 8F   AA   02A2   850           BICW    #UCB$M_VALID,-          ; And clear valid bit.
     64 A5        02A6   851                   UCB$W_STS(R5)           ;  and fall thru to rewind logic.
                     02A8   852
                     02A8   853 ;
```

```
                        02A8   854 ; LOGICAL REWIND FUNCTION - CONVERT TO PHYSICAL FUNCTION
                        02A8   855 ;
                        02A8   856
            51  03  9A  02A8   857 80$:    MOVZBL  #CDF_REWIND,R1          ;SET FOR REWIND FUNCTION
                13  11  02AB   858         BRB     110$                    ;
                        02AD   859
                        02AD   860 ;
                        02AD   861 ; LOGICAL WRITE EOF OR SENSE MODE FUNCTION - CONVERT TO PHYSICAL FUNCTION
                        02AD   862 ;
                        02AD   863
            51  15  A2  02AD   864 90$:    SUBW    #IO$_SENSEMODE-IO$_READPBLK-6,R1 ;CONVERT TO PHYSICAL FUNCTION
                0E  11  02B0   865         BRB     110$                    ;
                        02B2   866
                        02B2   867 ;
                        02B2   868 ; DENSE FUNCTION CODE - CHECK FOR READ, WRITE, OR WRITECHECK FUNCTION
                        02B2   869 ;
                        02B2   870
            51  0A  D1  02B2   871 100$:   CMPL    #IO$_WRITECHECK,R1      ;DATA TRANSFER FUNCTION?
                09  1A  02B5   872         BGTRU   110$                    ;IF GTRU NO
      03 009A C5  06  E1  02B7   873         BBC     #IO$V_REVERSE,UCB$W_FUNC(R5),110$ ;IF CLR, NOT REVERSE FUNCTION
            51  03  A0  02BD   874         ADDW    #CDF_WRITECHECKR-CDF_WRITECHECK,R1 ;CONVERT TO REVERSE FUNCTION
                        02C0   875
                        02C0   876 ;
                        02C0   877 ; FINISH PREPROCESSING
                        02C0   878 ;
                        02C0   879
        0092 C5  51  90 02C0   880 110$:   MOVB    R1,UCB$B_FEX(R5)        ;SAVE FUNCTION DISPATCH INDEX
                        02C5   881
                        02C5   882 ;
                        02C5   883 ; CENTRAL FUNCTION DISPATCH
                        02C5   884 ;
                        02C5   885
                        02C5   886 FDISPATCH:                           ;FUNCTION DISPATCH
           53   58 A5  D0  02C5   887         MOVL    UCB$L_IRP(R5),R3        ;RETRIEVE ADDRESS OF I/O PACKET
     0D 2A A3   08  E0  02C9   888         BBS     #IRP$V_PHYSIO,IRP$W_STS(R3),10$ ;IF SET, PHYSICAL I/O FUNCTION
     08 64 A5   0B  E0  02CE   889         BBS     #UCB$V_VALID,UCB$W_STS(R5),10$ ;IF SET, VOLUME SOFTWARE VALID
      50   0254 8F  3C  02D3   890         MOVZWL  #SS$_VOLINV,R0          ;SET VOLUME INVALID STATUS
             0B8C  31  02D8   891         BRW     RESETXFR                ;
                        02DB   892
                        02DB   893 ;
                        02DB   894 ; UNIT IS SOFTWARE VALID OR FUNCTION IS PHYSICAL I/O
                        02DB   895 ;
                        02DB   896
                        02DB   897 10$:
           53   24 A5  D0  02DB   898         MOVL    UCB$L_CRB(R5),R3        ;GET ADDRESS OF CRB
                        02DF   899         ASSUME  IDB$L_CSR       EQ      0
           53   2C B3  D0  02DF   900         MOVL    @CRB$L_INTD+VEC$L_IDB(R3),R3 ;GET ADDRESS OF TM03 CSR
                        02E3   901 20$:    DSBINT                          ;DISABLE INTERRUPTS
     2E 64 A5   05  E0  02E9   902         BBS     #UCB$V_POWER,UCB$W_STS(R5),30$ ;IF SET, POWER FAILED
     29 68 A5   00  E1  02EE   903         BBC     #UCB$V_MT_REWIND,UCB$W_DEVSTS(R5),30$ ;IF CLR, NO REWIND IN PROGRESS
                        02F3   904         WFIKPCH TIMEOUT,#60*7           ;WAITFOR REWIND TO COMPLETE
  24 A3   00C6 C5  3C  0301   905         MOVZWL  UCB$W_MT_TC(R5),MT_TC(R3) ;SELECT DRIVE
           52   04 A3  D0  0307   906         MOVL    MT_DS(R3),R2            ;READ DRIVE STATUS
           D4 52   0D  E0  030B   907         BBS     #MT_DS_V_PIP,R2,20$     ;IF SET, POSITIONING IN PROGRESS
           D0 52   01  E1  030F   908         BBC     #MT_DS_V_BOT,R2,20$     ;IF CLR, NOT AT BEGINNING OF TAPE
                        0313   909         IOFORK                          ;CREATE FORK PROCESS
                        0319   910         SAVIPL                          ;SAVE CURRENT IPL
```

```
                     031C   911 30$:    ENBINT                          ;ENABLE INTERRUPTS
                     031F   912          REQPCHAN                        ;REQUEST PRIMARY I/O CHANNEL
          53   54    D0  0325   913          MOVL    R4,R3                   ;SAVE ADDRESS OF TM03/DRIVE REGISTERS
    54    24   A5    D0  0328   914          MOVL    UCB$L_CRB(R5),R4        ;GET ADDRESS OF CRB
    54    20   A4    D0  032C   915          MOVL    CRB$L_LINK(R4),R4       ;GET ADDRESS OF SECONDARY CRB
                     0330   916          ASSUME  IDB$L_CSR       EQ      0
    54    2C   B4    D0  0330   917          MOVL    @CRB$L_INTD+VEC$L_IDB(R4),R4 ;GET ADDRESS OF MBA CSR
50        0092   C5    9A  0334   918          MOVZBL  UCB$B_FEX(R5),R0        ;GET DISPATCH FUNCTION CODE
                     0339   919          CASE    R0,<-                   ;DISPATCH TO FUNCTION HANDLING ROUTINE
                     0339   920                  NOP,-                   ;NO OPERATION
                     0339   921                  UNLOAD,-                ;UNLOAD VOLUME
                     0339   922                  SPCFILFOR,-             ;SPACE FILE FORWARD
                     0339   923                  REWIND,-                ;REWIND
                     0339   924                  DRVCLR,-                ;DRIVE CLEAR
                     0339   925                  SPCFILREV,-             ;SPACE FILE REVERSE
                     0339   926                  ERASE,-                 ;ERASE TAPE
                     0339   927                  SPCRECREV,-             ;SPACE RECORD REVERSE
                     0339   928                  PACKACK,-               ;PACK ACKNOWLEDGE
                     0339   929                  SPCRECFOR,-             ;SPACE RECORD FORWARD
                     0339   930                  WRITECHECK,-            ;WRITE CHECK FORWARD
                     0339   931                  WRITEDATA,-             ;WRITE DATA FORWARD
                     0339   932                  READDATA,-              ;READ DATA FORWARD
                     0339   933                  WRITECHECKR,-           ;WRITE CHECK REVERSE
                     0339   934                  WRITEDATA,-             ;WRITE DATA FORWARD
                     0339   935                  READDATAR,-             ;READ DATA REVERSE
                     0339   936                  READPRESET,-            ;READ IN PRESET
                     0339   937                  SETCHAR,-               ;SET TAPE CHARACTERISTICS
                     0339   938                  SENSECHAR,-             ;SENSE TAPE CHARACTERISTICS
                     0339   939                  >                       ;
```

```
                     0363   941          .SBTTL  WRITE TAPE MARK FUNCTION
                     0363   942 ;
                     0363   943 ; WRITE TAPE MARK FUNCTION
                     0363   944 ;
                     0363   945
                     0363   946 WRITEMARK:                                    ;WRITE TAPE MARK
                     0363   947 5$:       EXFUNC  10$,F_WRITEMARK             ;EXECUTE FUNCTION
          00B0 C5    D6     036B   948           INCL    UCB$L_RECORD(R5)     ;UPDATE TAPE POSITION
          50    01   3C     036F   949           MOVZWL  S^#SS$_NORMAL,R0     ;SET NORMAL COMPLETION STATUS
             06FE   31     0372   950           BRW     FUNCXT               ;
                     0375   951
                     0375   952 ;
                     0375   953 ; FUNCTION ENDED IN AN ERROR
                     0375   954 ;
                     0375   955 ; THE ERROR COULD BE A NONFATAL CONTROLLER OR DRIVE ERROR. FATAL ERRORS TERMINATE
                     0375   956 ; THE FUNCTION IN THE FUNCTION EXECUTOR.
                     0375   957 ;
                     0375   958 ; FATAL CONTROLLER ERRORS ARE:
                     0375   959 ;
                     0375   960 ;       ERCONF  = ERROR CONFIRMATION.
                     0375   961 ;       ISTO    = INTERFACE SEQUENCE TIMEOUT.
                     0375   962 ;       PGE     = PROGRAMMING ERROR.
                     0375   963 ;       NED     = NONEXISTENT DRIVE.
                     0375   964 ;       RDTO    = READ DATA TIMEOUT.
                     0375   965 ;
                     0375   966 ; FATAL DRIVE ERRORS ARE:
                     0375   967 ;
                     0375   968 ;       ILF     = ILLEGAL FUNCTION.
                     0375   969 ;       ILR     = ILLEGAL REGISTER.
                     0375   970 ;       NEF     = NONEXECUTABLE FUNCTION.
                     0375   971 ;       RMR     = REGISTER MODIFY REFUSE.
                     0375   972 ;       UNS     = UNSAFE.
                     0375   973 ;
                     0375   974 ; IGNORED DRIVE ERRORS ARE:
                     0375   975 ;
                     0375   976 ;       FMT     = FORMAT.
                     0375   977 ;       DPAR    = DATA BUS PARITY.
                     0375   978 ;       FCE     = FRAME COUNT.
                     0375   979 ;       DTE     = DRIVE TIMING.
                     0375   980 ;       COR/CRC = CORRECTABLE OR CHECK CHARACTER ERROR.
                     0375   981 ;
                     0375   982 ; NOTE THAT IT IS ASSUMED THAT MASSBUS EXCEPTION (MBEXC) WILL OCCUR ONLY IN
                     0375   983 ; COMBINATION WITH ANOTHER DRIVE OR CONTROLLER ERROR.
                     0375   984 ;
                     0375   985
                     0375   986 10$:      TESTR   20$                          ;TEST REMAINING RETRIES
                     037A   987           REQSCHAN                             ;REQUEST SECONDARY CHANNEL
                     0380   988           EXFUNC  DOUBLE,F_INTSPCREV           ;SPACE RECORD REVERSE
                     0388   989           RELSCHAN                             ;RELEASE SECONDARY CHANNEL
       08 009A C5    0C     E0     038E   990 20$:      BBS     #IO$V_INHEXTGAP,UCB$W_FUNC(R5),30$ ;IF SET, NO EXTENDED GAPS
                     0394   991           EXFUNC  DOUBLE,F_ERASE               ;WRITE EXTENDED INTER-RECORD GAP
                     C5     11     039C   992 30$:      BRB     5$                   ;
```

```
                     039E   994          .SBTTL  ERASE TAPE FUNCTION
                     039E   995 ;
                     039E   996 ; ERASE TAPE FUNCTION
                     039E   997 ;
                     039E   998
                     039E   999 ERASE:                                       ;ERASE TAPE
                     039E  1000          EXFUNC  10$                         ;EXECUTE FUNCTION
50   01    3C       03A3  1001          MOVZWL  S^#SS$_NORMAL,R0            ;SET NORMAL COMPLETION
00BC C5    B4       03A6  1002          CLRW    UCB$W_MT_FC(R5)             ;CLEAR FRAME COUNT
   06C6    31       03AA  1003          BRW     FUNCXT                      ;
                     03AD  1004
                     03AD  1005 ;
                     03AD  1006 ; FUNCTION ENDED IN AN ERROR
                     03AD  1007 ;
                     03AD  1008 ; THE ERROR COULD BE A NONFATAL CONTROLLER OR DRIVE ERROR. FATAL ERRORS TERMINATE
                     03AD  1009 ; THE FUNCTION IN THE FUNCTION EXECUTOR.
                     03AD  1010 ;
                     03AD  1011 ; FATAL CONTROLLER ERRORS ARE:
                     03AD  1012 ;
                     03AD  1013 ;       ERCONF  = ERROR CONFIRMATION.
                     03AD  1014 ;       ISTO    = INTERFACE SEQUENCE TIMEOUT.
                     03AD  1015 ;       PGE     = PROGRAMMING ERROR.
                     03AD  1016 ;       NED     = NONEXISTENT DRIVE.
                     03AD  1017 ;       RDTO    = READ DATA TIMEOUT.
                     03AD  1018 ;
                     03AD  1019 ; FATAL DRIVE ERRORS ARE:
                     03AD  1020 ;
                     03AD  1021 ;       ILF     = ILLEGAL FUNCTION.
                     03AD  1022 ;       ILR     = ILLEGAL REGISTER.
                     03AD  1023 ;       NEF     = NONEXECUTABLE FUNCTION.
                     03AD  1024 ;       RMR     = REGISTER MODIFY REFUSE.
                     03AD  1025 ;       UNS     = UNSAFE.
                     03AD  1026 ;
                     03AD  1027 ; IGNORED DRIVE ERRORS ARE:
                     03AD  1028 ;
                     03AD  1029 ;       FMT     = FORMAT.
                     03AD  1030 ;       DPAR    = DATA BUS PARITY.
                     03AD  1031 ;       INC/VPE = INCORRECTABLE OR VERTICLE PARITY ERROR.
                     03AD  1032 ;       PEF/LRC = FORMAT (PE) OR LONGITUDINAL PARITY ERROR.
                     03AD  1033 ;       NSG     = NONSTANDARD GAP.
                     03AD  1034 ;       FCE     = FRAME COUNT.
                     03AD  1035 ;       DTE     = DRIVE TIMING.
                     03AD  1036 ;       OPI     = OPERATION INCOMPLETE.
                     03AD  1037 ;       COR/CRC = CORRECTABLE OR CHECK CHARACTER ERROR.
                     03AD  1038 ;
                     03AD  1039
00B0 C5    D5       03AD  1040 10$:     TSTL    UCB$L_RECORD(R5)            ;ANY RECORDS ON TAPE?
     0A    12       03B1  1041          BNEQ    20$                         ;IF NEQ YES
                     03B3  1042          EXFUNC  DOUBLE,F_REWIND             ;REWIND TAPE
     1C    11       03BB  1043          BRB     30$                         ;
                     03BD  1044 20$:     REQSCHAN                            ;REQUEST SECONDARY CHANNEL
                     03C3  1045          EXFUNC  DOUBLE,F_INTSPCREV          ;BACK SPACE RECORD
                     03CB  1046          EXFUNC  DOUBLE,F_INTSPCFOR          ;SPACE RECORD FORWARD
                     03D3  1047          RELSCHAN                            ;RELEASE SECONDARY CHANNEL
   0615    31       03D9  1048 30$:     BRW     RETRY                       ;
```

```
                        1050          .SBTTL  HOUSEKEEPING FUNCTIONS
                        1051 ;
                        1052 ; HOUSEKEEPING FUNCTIONS INCLUDE:
                        1053 ;
                        1054 ;        PACK ACKNOWLEDGE,
                        1055 ;        NO OPERATION,
                        1056 ;        DRIVE CLEAR,
                        1057 ;        SENSE TAPE CHARACTERISTICS, AND
                        1058 ;        SET TAPE CHARACTERISTICS.
                        1059 ;
                        1060 ; IF THE FUNCTION ENDS IN A NONFATAL DRIVE ERROR IT IS RETRIED. FATAL ERRORS
                        1061 ; TERMINATE THE FUNCTION IN THE FUNCTION EXECUTOR.
                        1062 ;
                        1063 ; FATAL DRIVE ERRORS ARE:
                        1064 ;
                        1065 ;        ILF     = ILLEGAL FUNCTION.
                        1066 ;        ILR     = ILLEGAL REGISTER.
                        1067 ;        NEF     = NONEXECUTABLE FUNCTION (EXCEPT FOR DRIVE CLEAR).
                        1068 ;        RMR     = REGISTER MODIFY REFUSE.
                        1069 ;        UNS     = UNSAFE.
                        1070 ;
                        1071 ; IGNORED DRIVE ERRORS ARE:
                        1072 ;
                        1073 ;        FMT     = FORMAT.
                        1074 ;        DPAR    = DATA BUS PARITY.
                        1075 ;        INC/VPE = INCORRECTABLE OR VERTICLE PARITY ERROR.
                        1076 ;        PEF/LRC = FORMAT (PE) OR LONGITUDINAL PARITY ERROR.
                        1077 ;        NSG     = NONSTANDARD GAP.
                        1078 ;        FCE     = FRAME COUNT.
                        1079 ;        CS/ITM  = CORRECTABLE SKEW OR INVALID TAPE MARK.
                        1080 ;        DTE     = DRIVE TIMING.
                        1081 ;        OPI     = OPERATION INCOMPLETE.
                        1082 ;        COR/CRC = CORRECTABLE OR CHECK CHARACTER ERROR.
                        1083 ;
                        1084 ; ADDITIONAL IGNORED DRIVE ERRORS FOR DRIVE CLEAR ARE:
                        1085 ;
                        1086 ;        NEF     = NONEXECUTABLE FUNCTION.
                        1087 ;
                        1088 ; PACK ACKNOWLEDGE
                        1089 ;
                        1090
                        1091          .ENABL  LSB
                        1092 PACKACK:                                    ;PACK ACKNOWLEDGE
                        1093          EXFUNC  RETRY                     ;EXECUTE FUNCTION
64 A5   0800 8F   A8    03E1  1094          BISW    #UCB$M_VALID,UCB$W_STS(R5) ;SET VOLUME SOFTWARE VALID
                74   11   03E7  1095          BRB     30$               ;
                      03E9  1096
                      03E9  1097 ;
                      03E9  1098 ; NO OPERATION, SENSE CHARACTERISTICS, AND DRIVE CLEAR.
                      03E9  1099 ;
                      03E9  1100
                      03E9  1101 NOP:                                   ;NO OPERATION
                      03E9  1102 SENSECHAR:                             ;SENSE CHARACTERISTICS
                      03E9  1103 DRVCLR:                                ;DRIVE CLEAR
                      03E9  1104          EXFUNC  RETRY                 ;EXECUTE HOUSEKEEPING FUNCTION
                6D    11    03FE  1105          BRB     30$               ;
                      03r0  1106
```

```
                        03F0  1107 ;
                        03F0  1108 ; SET TAPE CHARACTERISTICS
                        03F0  1109 ;
                        03F0  1110
                        03F0  1111 SETCHAR:                                     ;SET TAPE CHARACTERISTICS
                        03F0  1112          EXFUNC  RETRY                       ;EXECUTE FUNCTION
              05    08  EF  03F5  1113          EXTZV   #MT$V_DENSITY,#MT$S_DENSITY,- ;EXTRACT DENSITY CODE
         50     7C A5      03F8  1114                  UCB$W_BOFF(R5),R0          ;
              50    04  C4  03FB  1115          MULL    #4,R0                   ;CALCULATE BIT NUMBER
   50   FC34 CF    04  50  EF  03FE  1116          EXTZV   R0,#4,DENSITY,R0        ;EXTRACT DENSITY CODE
              04    04  EF  0405  1117          EXTZV   #MT$V_FORMAT,#MT$S_FORMAT,- ;EXTRACT FORMAT CODE
         51     7C A5      0408  1118                  UCB$W_BOFF(R5),R1          ;
              51    04  C4  040B  1119          MULL    #4,R1                   ;CALCULATE BIT NUMBER
   51   FC58 CF    04  51  EF  040E  1120          EXTZV   R1,#4,FORMAT,R1         ;EXTRACT FORMAT CODE
         53    00F0 8F  3C  0415  1121          MOVZWL  #MT$M_FORMAT,R3         ;SET INITIAL MASK WORD
           0C 52    01  E1  041A  1122          BBC     #MT_DS_V_BOT,R2,10$     ;IF CLR, TAPE NOT AT BEGINNING
                        041E  1123
                        041E  1124 ;
                        041E  1125 ; TAPE DENSITY CAN ONLY BE SET WHEN THE SELECTED DRIVE IS AT BEGINNING OF TAPE
                        041E  1126 ;
                        041E  1127
              08    50  F0  041E  1128          INSV    R0,#MT_TC_V_DEN,-       ;SET NEW DENSITY
         00C6 C5    03      0421  1129                  #MT_TC_S_DEN,UCB$W_MT_TC(R5) ;
         53   1F00 8F  A8  0425  1130          BISW    #MT$M_DENSITY,R3        ;SET DENSITY MASK BITS
         00C6 C5    08  AA  042A  1131 10$:     BICW    #MT_TC_M_EPAR,UCB$W_MT_TC(R5) ;CLEAR EVEN PARITY
           44 A5    08  AA  042F  1132          BICW    #MT$M_PARITY,UCB$L_DEVDEPEND(R5) ;CLEAR EVEN PARITY
      0D 00C6 C5    0A  E0  0433  1133          BBS     #MT_TC_V_DEN+2,UCB$W_MT_TC(R5),20$ ;IF SET, PHASE ENCODED TAPE
                        0439  1134
                        0439  1135 ;
                        0439  1136 ; TAPE PARITY CAN ONLY BE SET IF NRZI FORMATTED TAPE IS BEING READ OR WRITTEN
                        0439  1137 ;
                        0439  1138
   08 7C A5    03  E1  0439  1139          BBC     #MT$V_PARITY,UCB$W_BOFF(R5),20$ ;IF CLR, ODD PARITY
    00C6 C5    08  A8  043E  1140          BISW    #MT_TC_M_EPAR,UCB$W_MT_TC(R5) ;SET EVEN PARITY
         53    08  A8  0443  1141          BISW    #MT$M_PARITY,R3         ;SET PARITY MASK BIT
         04    51  F0  0446  1142 20$:     INSV    R1,#MT_TC_V_FSEL,-      ;SET NEW FORMAT
    00C6 C5    04      0449  1143                  #MT_TC_S_FSEL,UCB$W_MT_TC(R5) ;
      44 A5    53  AA  044D  1144          BICW    R3,UCB$L_DEVDEPEND(R5)  ;CLEAR OLD FIELD VALUES
         53    53  D2  0451  1145          MCOML   R3,R3                   ;COMPLEMENT MASK
       7C A5    53  AA  0454  1146          BICW    R3,UCB$W_BOFF(R5)       ;CLEAR FIELDS NOT BE BE INSERTED
   44 A5    7C A5  A8  0458  1147          BISW    UCB$W_BOFF(R5),UCB$L_DEVDEPEND(R5) ;INSERT NEW FIELD VALUES
         00BC C5    B4  045D  1148 30$:     CLRW    UCB$W_MT_FC(R5)         ;CLEAR SAVED FRAME COUNT REGISTER
         50    01   3C  0461  1149          MOVZWL  S^#SS$_NORMAL,R0        ;SET NORMAL COMPLETION
             060C    31  0464  1150          BRW     FUNCXT                  ;
                        0467  1151          .DSABL  LSB
```

```
                     0467  1153          .SBTTL  REWIND AND UNLOAD FUNCTIONS
                     0467  1154 ;
                     0467  1155 ; REWIND AND UNLOAD FUNCTIONS INCLUDE:
                     0467  1156 ;
                     0467  1157 ;       READIN PRESET,
                     0467  1158 ;       REWIND, AND
                     0467  1159 ;       UNLOAD.
                     0467  1160 ;
                     0467  1161 ; IF THE FUNCTION ENDS WITH A NONFATAL DRIVE ERRORS IT IS RETRIED. FATAL ERRORS
                     0467  1162 ; TERMINATE THE FUNCTION IN THE FUNCTION EXECUTOR.
                     0467  1163 ;
                     0467  1164 ; FATAL DRIVE ERRORS ARE:
                     0467  1165 ;
                     0467  1166 ;       ILF     = ILLEGAL FUNCTION.
                     0467  1167 ;       ILR     = ILLEGAL REGISTER.
                     0467  1168 ;       NEF     = NONEXECUTABLE FUNCTION.
                     0467  1169 ;       RMR     = REGISTER MODIFY REFUSE.
                     0467  1170 ;       UNS     = UNSAFE.
                     0467  1171 ;
                     0467  1172 ; IGNORED DRIVE ERRORS ARE:
                     0467  1173 ;
                     0467  1174 ;       FMT     = FORMAT.
                     0467  1175 ;       DPAR    = DATA BUS PARITY.
                     0467  1176 ;       INC/VPE = INCORRECTABLE OR VERTICLE PARITY ERROR.
                     0467  1177 ;       PEF/LRC = FORMAT (PE) OR LONGITUDINAL PARITY ERROR.
                     0467  1178 ;       NSG     = NONSTANDARD GAP.
                     0467  1179 ;       FCE     = FRAME COUNT.
                     0467  1180 ;       CS/ITM  = CORRECTABLE SKEW OR INVALID TAPE MARK.
                     0467  1181 ;       DTE     = DRIVE TIMING.
                     0467  1182 ;       OPI     = OPERATION INCOMPLETE.
                     0467  1183 ;       COR/CRC = CORRECTABLE OR CHECK CHARACTER ERROR.
                     0467  1184 ;
                     0467  1185
                     0467  1186 READPRESET:                                     ;READ IN PRESET
       54 A5   B5    0467  1187          TSTW    UCB$W_UNIT(R5)                ;UNIT ZERO?
          03   13    046A  1188          BEQL    REWIND                        ;IF EQL YES
    50    03   9A    046C  1189          MOVZBL  #CDF_REWIND,R0                ;CONVERT FUNCTION TO REWIND
                     046F  1190
                     046F  1191 ;
                     046F  1192 ; UNLOAD AND REWIND FUNCTIONS
                     046F  1193 ;
                     046F  1194
                     046F  1195 REWIND:                                        ;REWIND TO BEGINNING OF TAPE
                     046F  1196 UNLOAD:                                        ;UNLOAD VOLUME
    00B0 C5    D4    046F  1197          CLRL    UCB$L_RECORD(R5)              ; Since we will REWIND, the current
                     0473  1198                                                ;  position is of no interest.
                     0473  1199                                                ; This insures that UCB$L_RECORD will
                     0473  1200                                                ;  be correct if we do a REWIND nowait.
    00CC C5    D4    0473  1201          CLRL    UCB$L_MT_ORGPOS(R5)           ; With REWIND the original position
                     0477  1202                                                ;  is of no interest.  If we should
                     0477  1203                                                ;  get a POWERFAIL this will reset us
                     0477  1204                                                ;  to BOT before retrying.
                     0477  1205          EXFUNC  LOSTPOS                       ;EXECUTE FUNCTION
    00BC C5    B4    047C  1206          CLRW    UCB$W_MT_FC(R5)               ;CLEAR SAVED FRAME COUNT REGISTER
    50    01   3C    0480  1207          MOVZWL  S^#SS$_NORMAL,R0              ;SET NORMAL COMPLETION STATUS
 06 52    0C   E0    0483  1208          BBS     #MT_DS_V_MOL,R2,10$           ;IF SET, MEDIUM ONLINE
64 A5   0800 8F    AA    0487  1209          BICW    #UCB$M_VALID,UCB$W_STS(R5) ;SET VOLUME SOFTWARE INVALID
```

```
                05E3   31  048D  1210 10$:    BRW     FUNCXT                  ;
```

```
0490  1212          .SBTTL  SPACING FUNCTIONS
0490  1213 ;
0490  1214 ; SPACING FUNCTIONS INCLUDE:
0490  1215 ;
0490  1216 ;       SPACE FILE FORWARD,
0490  1217 ;       SPACE FILE REVERSE,
0490  1218 ;       SPACE RECORD FORWARD, AND
0490  1219 ;       SPACE RECORD REVERSE.
0490  1220 ;
0490  1221 ; ALL ARE IMPLEMENTED VIA THE SPACE RECORD FUNCTIONS.
0490  1222 ;
0490  1223 ; A SPACING FUNCTION CAN END WITH A NONFATAL DRIVE ERROR. FATAL ERRORS TERMINATE
0490  1224 ; THE FUNCTION IN THE FUNCTION EXECUTOR.
0490  1225 ;
0490  1226 ; FATAL DRIVE ERRORS ARE:
0490  1227 ;
0490  1228 ;       ILF     = ILLEGAL FUNCTION.
0490  1229 ;       ILR     = ILLEGAL REGISTER.
0490  1230 ;       NEF     = NONEXECUTABLE FUNCTION (FORWARD FUNCTIONS).
0490  1231 ;       RMR     = REGISTER MODIFY REFUSE.
0490  1232 ;       UNS     = UNSAFE.
0490  1233 ;
0490  1234 ; NONFATAL DRIVE ERRORS ARE:
0490  1235 ;
0490  1236 ;       CPAR    = CONTROL BUS PARITY.
0490  1237 ;       FCE     = FRAME COUNT.
0490  1238 ;       NEF     = NONEXECUTABLE FUNCTION (REVERSE FUNCTIONS INTO BEGINNING OF TAPE).
0490  1239 ;       OPI     = OPERATION INCOMPLETE (REVERSE FUNCTIONS INTO BEGINNING OF TAPE).
0490  1240 ;
0490  1241 ; IGNORED DRIVE ERRORS ARE:
0490  1242 ;
0490  1243 ;       FMT     = FORMAT.
0490  1244 ;       DPAR    = DATA BUS PARITY.
0490  1245 ;       INC/VPE = INCORRECTABLE OR VERTICLE PARITY ERROR.
0490  1246 ;       PEF/LRC = FORMAT (PE) OR LONGITUDINAL PARITY ERROR.
0490  1247 ;       NSG     = NONSTANDARD GAP.
0490  1248 ;       CS/ITM  = CORRECTABLE SKEW OR INVALID TAPE MARK.
0490  1249 ;       DTE     = DRIVE TIMING.
0490  1250 ;       COR/CRC = CORRECTABLE OR CHECK CHARACTER ERROR.
0490  1251 ;
0490  1252 ; SPACE FILE FORWARD
0490  1253 ;
0490  1254 ;       SPACING FILES IS ACCOMPLISHED BY SPACING A VERY LARGE NUMBER OF RECORDS.
0490  1255 ;       IF THE RECORD SPACING OPERATION COMPLETES WITHOUT ERROR, THEN THE RECORD
0490  1256 ;       COUNT IS INCREASED BY 65,536 AND THE SPACING OPERATION IS CONTINUED.
0490  1257 ;
0490  1258 ;       SEVERAL SPECIAL CONDITIONS CAN ARISE DURING A SPACE FILE FORWARD:
0490  1259 ;
0490  1260 ;               1. A CONTROL BUS PARITY ERROR OCCURS.
0490  1261 ;
0490  1262 ;                       THE OPERATION IS MERELY RETRIED SINCE TAPE MOTION COULD
0490  1263 ;                       NOT HAVE OCCURED.
0490  1264 ;
0490  1265 ;               2. AN END OF TAPE IS ENCOUNTERED WITHOUT AN END OF FILE.
0490  1266 ;
0490  1267 ;                       THE OPERATION IS CONTINUED SINCE END OF TAPE WHILE FILE
0490  1268 ;                       SKIPPING DOES NOT TERMINATE THE OPERATION.
```

```
                                0490  1269 ;
                                0490  1270 ;               3. AN END OF FILE IS ENCOUNTERED.
                                0490  1271 ;
                                0490  1272 ;                       THE FILE SKIP COUNT IS DECREMENTED AND IF NONZERO, THE
                                0490  1273 ;                       OPERATION IS CONTINUED.
                                0490  1274 ;
                                0490  1275 ;       UNLESS A HARD ERROR IS ENCOUNTERED, NORMAL COMPLETION IS ALWAYS RETURNED
                                0490  1276 ;       FOR A SKIP FILE FORWARD FUNCTION.
                                0490  1277 ;
                                0490  1278 ;       AS RECORDS ARE SKIPPED FORWARD ON THE TAPE, THE CURRENT TAPE POSITION IS
                                0490  1279 ;       MAINTAINED BY ADDING THE NUMBER OF RECORDS AND TAPE MARKS SKIPPED OVER.
                                0490  1280 ;
                                0490  1281
                                0490  1282 SPCFILFOR:                                ;SPACE FILE FORWARD
                                0490  1283 ;
                                0490  1284 ; Before proceeding with the SKIP function, we backspace one tape position
                                0490  1285 ;       (be it record or TAPEMARK) and then forward space one position to
                                0490  1286 ;       determine if we are currently positioned immediately after a
                                0490  1287 ;       TAPEMARK.  If so we record the current position in UCB$L_MT_PREVTM.
                                0490  1288 ;       If we are not currently at a TAPEMARK, we move negative 2 to
                                0490  1289 ;       UCB$L_MT_PREVTM.
                                0490  1290 ;
                                0490  1291
          50   02   CE          0490  1292          MNEGL   #2,R0                           ; Initialize R0 = -2.
    52  04 A3   1E   78          0493  1293          ASHL    #31-MT_DS_V_BOT,MT_DS(R3),R2    ; See if at BOT.
                  48   19        0498  1294          BLSS    4$                              ; LSS implies at BOT.
               00B0 C5   D5       049A  1295          TSTL    UCB$L_RECORD(R5)        ; If NOT at physical BOT see if at logical
                                 049E  1296                                          ;  BOT.
                     42   13      049E  1297          BEQL    4$                      ; EQL implies logical BOT.
                                 04A0  1298
               00BE C5   B0       04A0  1299          MOVW    UCB$W_MT_SPACNT(R5),-   ; Copy spacing parameter to safe
               00C8 C5            04A4  1300                  UCB$L_MT_PREVTM(R5)     ;  keeping place.
          00BE C5   01   AE       04A7  1301          MNEGW   #1,UCB$W_MT_SPACNT(R5)  ; Indicate that we will backspace 1 position
                                 04AC  1302          EXFUNC  1$,F_SPCRECREV          ; Backspace 1 tape position.
                                 04B4  1303 1$:      CHECK_ERROR
               00B0 C5   D7       04B7  1304          DECL    UCB$L_RECORD(R5)        ; Update tape position marker.
                                 04BB  1305
          00BE C5   01   AE       04BB  1306          MNEGW   #1,UCB$W_MT_SPACNT(R5)  ; Now we will forward space 1 position.
                                 04C0  1307          EXFUNC  2$,F_SPCRECFOR          ; Forward space 1 tape position.
                                 04C8  1308 2$:      CHECK_ERROR
               00B0 C5   D6       04CB  1309          INCL    UCB$L_RECORD(R5)        ; Update tape position marker.
               50   02   CE       04CF  1310          MNEGL   #2,R0                   ; R0 = -2 which would imply that we are
                                 04D2  1311                                          ;  not immediately past a TAPEMARK.
            05 52   02   E1       04D2  1312          BBC     #MT_DS_V_TM,R2,3$       ; And if we are NOT at TAPEMARK, branch.
          50   00B0 C5   D0       04D6  1313          MOVL    UCB$L_RECORD(R5),R0     ; Else R0 = current tape position.
                                 04DB  1314 3$:
               00C8 C5   B0       04DB  1315          MOVW    UCB$L_MT_PREVTM(R5),-   ; Restore previous value to
               00BE C5            04DF  1316                  UCB$W_MT_SPACNT(R5)     ;  UCB$W_MT_SPACNT.
          00C8 C5   50   D0       04E2  1317 4$:      MOVL    R0,UCB$L_MT_PREVTM(R5)  ; Initialize previous TAPEMARK indicator.
                                 04E7  1318 5$:      EXFUNC  10$,F_SPCFILFOR         ;EXECUTE FUNCTION
               00B2 C5   B6       04EF  1319          INCW    UCB$L_RECORD+2(R5)      ;UPDATE TAPE POSITION
                     F2   11      04F3  1320          BRB     5$                      ;
                                 04F5  1321 10$:     CHECK_ERROR                     ;CHECK FOR FATAL OF RETRIABLE ERROR
          50   00BC C5   3C       04F8  1322          MOVZWL  UCB$W_MT_FC(R5),R0      ;GET NUMBER OF RECORDS SKIPPED OVER
          00B0 C5   50   C0       04FD  1323          ADDL    R0,UCB$L_RECORD(R5)     ;UPDATE TAPE POSITION
            E1 52   02   E1       0502  1324          BBC     #MT_DS_V_TM,R2,5$       ;IF CLR, TAPE MARK NOT ENCOUNTERED
                  7C A5   B7      0506  1325          DECW    UCB$W_BOFF(R5)          ;DECREMENT NUMBER OF FILES TO SKIP
```

```
                00C8 C5    C3  0509  1326          SUBL3   UCB$L_MT_PREVTM(R5),-
                                                           UCB$L_RECORD(R5),R0     ; R0 = distance between last 2 TAPEMARKs
          50    00B0 C5        050D  1327
                00B0 C5    D0  0511  1328          MOVL    UCB$L_RECORD(R5),-
                00C8 C5        0515  1329                  UCB$L_MT_PREVTM(R5)     ; Remember position of this TAPEMARK
                   13      E1  0518  1330          BBC     #DEV$V_MNT,-
                05 38 A5       051A  1331                  UCB$L_DEVCHAR(R5),20$   ; If NOT mounted, go test for EOV.
                   18      E1  051D  1332          BBC     #DEV$V_FOR,-
                04 38 A5       051F  1333                  UCB$L_DEVCHAR(R5),30$   ; If mounted NOT foreign (i.e. the ACP
                               0522  1334                                          ;  is involved) branch around test EOV.
                               0522  1335
                               0522  1336 20$:     ; If here volume is either NOT mounted or mounted FOREIGN.
                               0522  1337
                      50       D7  0522  1338          DECL    R0                  ; See if last 2 TAPEMARKs adjacent
                      08       13  0524  1339          BEQL    40$                 ; EQL implies adjacent TAPEMARKs
                               0526  1340
                               0526  1341 30$:     ; If here either ANSI mounted tape or EOV test failed.
                               0526  1342
                   7C A5       B5  0526  1343          TSTW    UCB$W_BOFF(R5)      ; See if we have more TAPEMARKs to skip
                      BC       12  0529  1344          BNEQ    5$                  ;IF NEQ MORE TO GO
                    00F9       31  052B  1345          BRW     NORXIT              ;
                    00FB       31  052E  1346 40$:     BRW     SETEOV              ; Branch around to return ENDOFVOLUME status
                               0531  1347
                               0531  1348 ;
                               0531  1349 ; SPACE FILE REVERSE
                               0531  1350 ;
                               0531  1351 ;       SPACING FILES IS ACCOMPLISHED BY SPACING A VERY LARGE NUMBER OF RECORDS.
                               0531  1352 ;       IF THE RECORD SPACING OPERATION COMPLETES WITHOUT ERROR, THE THE RECORD
                               0531  1353 ;       COUNT IS REDUCED BY 65,536 AND THE SPACING OPERATION IS CONTINUED.
                               0531  1354 ;
                               0531  1355 ;       SEVERAL SPECIAL CONDITIONS CAN ARISE DURING A SPACE FILE REVERSE:
                               0531  1356 ;
                               0531  1357 ;               1. A CONTROL BUS PARITY ERROR OCCURS.
                               0531  1358 ;
                               0531  1359 ;                       THE OPERATION IS MERELY RETRIED SINCE TAPE MOTION COULD
                               0531  1360 ;                       NOT HAVE OCCURED.
                               0531  1361 ;
                               0531  1362 ;               2. AN END OF TAPE IS ENCOUNTERED WITHOUT AN END OF FILE.
                               0531  1363 ;
                               0531  1364 ;                       THE OPERATION IS CONTINUED SINCE END OF TAPE WHILE FILE
                               0531  1365 ;                       SKIPPING DOES NOT TERMINATE THE OPERATION.
                               0531  1366 ;
                               0531  1367 ;               3. A BEGINNING OF TAPE IS ENCOUNTERED.
                               0531  1368 ;
                               0531  1369 ;                       THE OPERATION IS IMMEDIATELY TERMINATED.
                               0531  1370 ;
                               0531  1371 ;               4. AN END OF FILE IS ENCOUNTERED.
                               0531  1372 ;
                               0531  1373 ;                       THE FILE SKIP COUNT IS DECREMENTED AND IF NONZERO, THE
                               0531  1374 ;                       OPERATION IS CONTINUED.
                               0531  1375 ;
                               0531  1376 ;       UNLESS A HARD ERROR IS ENCOUNTERED, NORMAL COMPLETION IS ALWAYS RETURNED
                               0531  1377 ;       FOR A SKIP FILE REVERSE OPERATION.
                               0531  1378 ;
                               0531  1379 ;       AS RECORDS ARE SKIPPED BACKWARDS ON THE TAPE, THE CURRENT TAPE POSITION
                               0531  1380 ;       IS MAINTAINED BY SUBTRACTING THE NUMBER RECORDS AND TAPE MARKS SKIPPED
                               0531  1381 ;       OVER.
                               0531  1382 ;
```

```
                     0531  1383
                     0531  1384 SPCFILREV:                                  ;SPACE FILE REVERSE
                     0531  1385          EXFUNC  10$,F_SPCFILREV            ;EXECUTE FUNCTION
        00B2 C5   B7 0539  1386          DECW    UCB$L_RECORD+2(R5)         ;UPDATE RECORD POSITION
             F2   11 053D  1387          BRB     SPCFILREV                  ;
                     053F  1388 10$:      CHECK_ERROR                       ;CHECK FOR FATAL OR RETRIABLE ERROR
     50 00BC C5   3C 0542  1389          MOVZWL  UCB$W_MT_FC(R5),R0         ;GET NUMBER OF RECORDS SKIPPED OVER
00B0 C5   50   C2 0547  1390          SUBL    R0,UCB$L_RECORD(R5)        ;UPDATE TAPE POSITION
     52   06   B3 054C  1391          BITW    #MT_DS_M_BOT!MT_DS_M_TM,R2 ;NOT AT BOT AND NO TAPE MARK ENCOUNTERED?
          E0   13 054F  1392          BEQL    SPCFILREV                  ;IF EQL YES
  05 52   01   E0 0551  1393          BBS     #MT_DS_V_BOT,R2,20$        ;IF SET, BEGINNING OF TAPE
        7C A5  B7 0555  1394          DECW    UCB$W_BOFF(R5)             ;ANY MORE FILES TO SKIP?
           D7  12 0558  1395          BNEQ    SPCFILREV                  ;IF NEQ YES
         00CA  31 055A  1396 20$:     BRW     NORXIT                     ;
                     055D  1397
                     055D  1398 ;
                     055D  1399 ; SPACE RECORD FORWARD
                     055D  1400 ;
                     055D  1401 ;       SEVERAL SPECIAL CONDITIONS CAN ARISE DURING A SPACE FORWARD:
                     055D  1402 ;
                     055D  1403 ;               1. A CONTROL BUS PARITY ERROR OCCURS.
                     055D  1404 ;
                     055D  1405 ;                       THE OPERATION IS MERELY RETRIED SINCE TAPE MOTION COULD
                     055D  1406 ;                       NOT HAVE OCCURED.
                     055D  1407 ;
                     055D  1408 ;               2. AN END OF TAPE IS ENCOUNTERED WITHOUT AN END OF FILE.
                     055D  1409 ;
                     055D  1410 ;                       THE OPERATION IS TERMINATED WITH A FINAL STATUS OF END OF
                     055D  1411 ;                       TAPE.
                     055D  1412 ;
                     055D  1413 ;               3. AN END OF FILE IS ENCOUNTERED.
                     055D  1414 ;
                     055D  1415 ;                       THE OPERATION IS TERMINATED WITH A FINAL STATUS OF END OF
                     055D  1416 ;                       FILE.
                     055D  1417 ;
                     055D  1418 ;       UNLESS A HARD ERROR IS ENCOUNTERED, THE CURRENT TAPE POSITION IS MAINTAINED
                     055D  1419 ;       BY ADDING THE TOTAL NUMBER OF RECORDS SKIPPED OVER.
                     055D  1420 ;
                     055D  1421
                     055D  1422 SPCRECFOR:                                  ;SPACE RECORD FORWARD
                     055D  1423 ;
                     055D  1424 ; Before proceeding with the SKIP function, we backspace one tape position
                     055D  1425 ;       (be it record or TAPEMARK) and then forward space one position to
                     055D  1426 ;       determine if we are currently positioned immediately after a
                     055D  1427 ;       TAPEMARK.  If so we record the current position in UCB$L_MT_PREVTM.
                     055D  1428 ;       If we are not currently at a TAPEMARK, we move negative 2 to
                     055D  1429 ;       UCB$L_MT_PREVTM.
                     055D  1430 ;
                     055D  1431
           50   02   CE 055D  1432          MNEGL   #2,R0                              ; Initialize R0 = -2.
52   04 A3   1E   78 0560  1433          ASHL    #31-MT_DS_V_BOT,MT_DS(R3),R2       ; See if at BOT.
           48   19 0565  1434          BLSS    4$                                 ; LSS implies at BOT.
        00B0 C5   D5 0567  1435          TSTL    UCB$L_RECORD(R5)           ; If NOT at physical BOT see if at logical
                     056B  1436                                             ;  BOT.
           42   13 056B  1437          BEQL    4$                         ; EQL implies logical BOT.
                     056D  1438
        00BE C5   B0 056D  1439          MOVW    UCB$W_MT_SPACNT(R5),-      ; Copy spacing parameter to safe
```

```
                00C8 C5          0571  1440                   UCB$L_MT_PREVTM(R5)     ;  keeping place.
          00BE C5    01    AE    0574  1441           MNEGW   #1,UCB$W_MT_SPACNT(R5) ; Indicate that we will backspace 1 position
                               0579  1442           EXFUNC  1$,F_SPCRECREV          ; Backspace 1 tape position.
                               0581  1443 1$:       CHECK_ERROR
                00B0 C5    D7    0584  1444           DECL    UCB$L_RECORD(R5)        ; Update tape position marker.
                               0588  1445
          00BE C5    01    AE    0588  1446           MNEGW   #1,UCB$W_MT_SPACNT(R5) ; Now we will forward space 1 position.
                               058D  1447           EXFUNC  2$,F_SPCRECFOR          ; Forward space 1 tape position.
                               0595  1448 2$:       CHECK_ERROR
                00B0 C5    D6    0598  1449           INCL    UCB$L_RECORD(R5)        ; Update tape position marker.
                50   02    CE    059C  1450           MNEGL   #2,R0                   ; R0 = -2 which would imply that we are
                               059F  1451                                           ;  not immediately past a TAPEMARK.
             05 52   02    E1    059F  1452           BBC     #MT_DS_V_TM,R2,3$       ; And if we are NOT at TAPEMARK, branch.
          50 00B0 C5    D0    05A3  1453           MOVL    UCB$L_RECORD(R5),R0     ; Else R0 = current tape position.
                               05A8  1454 3$:
                00C8 C5    B0    05A8  1455           MOVW    UCB$L_MT_PREVTM(R5),-   ; Restore previous value to
                00BE C5          05AC  1456                   UCB$W_MT_SPACNT(R5)     ;  UCB$W_MT_SPACNT.
          00C8 C5    50    D0    05AF  1457 4$:       MOVL    R0,UCB$L_MT_PREVTM(R5) ; Initialize previous TAPEMARK indicator.
                               05B4  1458           EXFUNC  10$,F_SPCRECFOR         ;SPACE RECORD FORWARD
                      03   11    05BC  1459           BRB     20$                     ;
                               05BE  1460 10$:      CHECK_ERROR                     ;CHECK FOR FATAL OR RETRIABLE ERROR
                      50   D4    05C1  1461 20$:      CLRL    R0                      ;CLEAR UPPER HALF OF REGISTER
     50   7C A5    00BC C5    A1    05C3  1462           ADDW3   UCB$W_MT_FC(R5),UCB$W_BOFF(R5),R0 ;CALCULATE RECORDS SKIPPED
          00B0 C5    50    C0    05CA  1463           ADDL    R0,UCB$L_RECORD(R5)     ;UPDATE TAPE POSITION
          7C A5    00BC C5    AE    05CF  1464           MNEGW   UCB$W_MT_FC(R5),UCB$W_BOFF(R5) ;SET REMAINING RECORD SKIP COUNT
             4E 52   02    E1    05D5  1465           BBC     #MT_DS_V_TM,R2,NORXIT   ; If NOT at TAPEMARK, goto normal exit.
                      13   E1    05D9  1466           BBC     #DEV$V_MNT,-            ; If NOT mounted and at TAPEMARK,
                05 38 A5          05DB  1467                   UCB$L_DEVCHAR(R5),30$   ;  go test if at ENDOFVOLUME.
                      18   E1    05DE  1468           BBC     #DEV$V_FOR,-            ; If mounted ANSI (i.e NOT foreign),
                6E 38 A5          05E0  1469                   UCB$L_DEVCHAR(R5),SETEOF;  go to set EOF status.
                               05E3  1470
                               05E3  1471 30$: ; If here we are at TAPEMARK and the volume is either NOT mounted or
                               05E3  1472 ;                 mounted foreign.
                               05E3  1473
          00C8 C5    C3    05E3  1474           SUBL3   UCB$L_MT_PREVTM(R5),-   ; Calculate the distance between last
     50   00B0 C5          05E7  1475                   UCB$L_RECORD(R5),R0     ;  two TAPEMARKs.
                50   D7    05EB  1476           DECL    R0                      ; See if last 2 TAPEMARKs adjacent.
                3D   13    05ED  1477           BEQL    SETEOV                  ; EQL implies ENDOFVOLUME.
                60   11    05EF  1478           BRB     SETEOF                  ; If NOT EOV, then EOF.
                               05F1  1479
                               05F1  1480 ;
                               05F1  1481 ; SPACE RECORD REVERSE
                               05F1  1482 ;
                               05F1  1483 ;       SEVERAL SPECIAL CONDITIONS CAN ARISE DURING A SPACE RECORD REVERSE:
                               05F1  1484 ;
                               05F1  1485 ;               1. A CONTROL BUS PARITY ERROR.
                               05F1  1486 ;
                               05F1  1487 ;                       THE OPERATION IS MERELY RETRIED SINCE TAPE MOTION COULD
                               05F1  1488 ;                       NOT HAVE OCCURED.
                               05F1  1489 ;
                               05F1  1490 ;               2. A BEGINNING OF TAPE IS ENCOUNTERED.
                               05F1  1491 ;
                               05F1  1492 ;                       THE OPERATION IS TERMINATED WITH A FINAL STATUS OF END OF
                               05F1  1493 ;                       FILE.
                               05F1  1494 ;
                               05F1  1495 ;               3. AN END OF TAPE IS ENCOUNTERED WITHOUT AN END OF FILE.
                               05F1  1496 ;
```

```
                     05F1  1497 ;                          THE OPERATION IS CONTINUED SINCE AN END OF TAPE WHILE
                     05F1  1498 ;                          SKIPPING RECORDS BACKWARDS DOES NOT TERMINATE THE OPER-
                     05F1  1499 ;                          ATION.
                     05F1  1500 ;
                     05F1  1501 ;                   4. AN END OF FILE IS ENCOUNTERED.
                     05F1  1502 ;
                     05F1  1503 ;                          THE OPERATION IS TERMINATED WITH A FINAL STATUS OF END OF
                     05F1  1504 ;                          FILE.
                     05F1  1505 ;
                     05F1  1506 ;       UNLESS A HARD ERROR IS ENCOUNTERED, THE CURRENT TAPE POSITION IS MAINTAINED
                     05F1  1507 ;       BY SUBTRACTING THE TOTAL NUMBER OF RECORDS SKIPPED OVER.
                     05F1  1508 ;
                     05F1  1509
                     05F1  1510 SPCRECREV:                                   ;SPACE RECORD REVERSE
                     05F1  1511          EXFUNC  10$,F_SPCRECREV             ;EXECUTE FUNCTION
           03   11  05F9  1512          BRB     20$                         ;
                     05FB  1513 10$:     CHECK_ERROR                         ;CHECK FOR FATAL OR RETRIABLE ERROR
           50   D4  05FE  1514 20$:     CLRL    R0                          ;CLEAR UPPER HALF OF REGISTER
50  7C A5   00BC C5  A1  0600  1515          ADDW3   UCB$W_MT_FC(R5),UCB$W_BOFF(R5),R0 ;CALCULATE RECORD SKIP COUNT
      00B0 C5   50  C2  0607  1516          SUBL    R0,UCB$L_RECORD(R5)         ;UPDATE TAPE POSITION
   7C A5    00BC C5  AE  060C  1517          MNEGW   UCB$W_MT_FC(R5),UCB$W_BOFF(R5) ;SET REMAINING RECORD SKIP COUNT
00BE C5     00BC C5  B0  0612  1518          MOVW    UCB$W_MT_FC(R5),UCB$W_MT_SPACNT(R5) ;RESET SPACING COUNT
        34 52    02  E0  0619  1519          BBS     #MT_DS_V_TM,R2,SETEOF       ;IF SET, END OF FILE ENCOUNTERED
           08    13  061D  1520          BEQL    NORXIT                      ;IF EQL ALL RECORDS SPACED OVER
        2E 52    01  E0  061F  1521          BBS     #MT_DS_V_BOT,R2,SETEOF      ;IF SET, BEGINNING OF TAPE ENCOUNTERED
        CA 52    0A  E0  0623  1522          BBS     #MT_DS_V_EOT,R2,SPCRECREV ;IF SET, AT END OF TAPE
                     0627  1523
                     0627  1524 ;
                     0627  1525 ; NORMAL SPACING FUNCTION EXIT
                     0627  1526 ;
                     0627  1527
                     0627  1528          .ENABL  LSB
   50   01   3C  0627  1529 NORXIT: MOVZWL  S^#SS$_NORMAL,R0                ;SET NORMAL COMPLETION STATUS
        2A   11  062A  1530          BRB     10$                         ;
                     062C  1531
                     062C  1532 ;
                     062C  1533 ; Backspace over 2nd TAPEMARK and set ENDOFVOLUME return status
                     062C  1534 ;
                     062C  1535
                     062C  1536 SETEOV:
00BE C5   01   AE  062C  1537          MNEGW   #1,UCB$W_MT_SPACNT(R5)  ; Don't space back more than 1 tape position
                     0631  1538          EXFUNC  2$,F_SPCFILREV           ; Backspace one tape position to
                     0639  1539                                          ;  position us between the two TAPEMARKs.
                     0639  1540 2$:      CHECK_ERROR                      ; Check for normal spacing errors.
  0E 52    02  E1  063C  1541          BBC     #MT_DS_V_TM,R2,4$        ; If NOT at TAPEMARK, branch back to error.
   00B0 C5   D7  0640  1542          DECL    UCB$L_RECORD(R5)         ; Update position in UCB.
     7C A5   B6  0644  1543          INCW    UCB$W_BOFF(R5)           ; Add one to total of files or
                     0647  1544                                          ;  records left to skip.
50  09A0 8F   3C  0647  1545          MOVZWL  #SS$_ENDOFVOLUME,R0      ; Set ENDOFVOLUME status
        08   11  064C  1546          BRB     10$                      ; Branch around to return
      03AF   31  064E  1547 4$:      BRW     DOUBLE                   ; If not at a TAPEMARK, then error.
                     0651  1548
                     0651  1549 ;
                     0651  1550 ; SET EOF OF FILE STATUS
                     0651  1551 ;
                     0651  1552
50  0870 8F   3C  0651  1553 SETEOF: MOVZWL  #SS$_ENDOFFILE,R0        ;SET END OF FILE STATUS
```

```
                7E A5   7C A5   A3  0656  1554 10$:    SUBW3   UCB$W_BOFF(R5),UCB$W_BCNT(R5),- ;CALCULATE TOTAL NUMBER OF FILES
                    00BC C5         065B  1555                  UCB$W_MT_FC(R5)                 ;OR RECORDS SKIPPED OVER
                       0412  31     065E  1556          BRW     FUNCXT                          ;
                             0661  1557          .DSABL  LSB
```

```
                        0661  1559          .SBTTL  READ DATA FORWARD AND WRITECHECK DATA FORWARD FUNCTIONS
                        0661  1560 ;
                        0661  1561 ; READ DATA FORWARD AND WRITECHECK DATA FORWARD FUNCTIONS
                        0661  1562 ;
                        0661  1563
                        0661  1564 WRITECHECK:                                  ;WRITE CHECK FORWARD
009A C5   4000 8F   AA  0661  1565          BICW    #IO$M_DATACHECK,UCB$W_FUNC(R5) ;CLEAR DATA CHECK REQUEST
                        0668  1566 READDATA:                                   ;READ DATA FORWARD
                        0668  1567          REQSCHAN                           ;REQUEST SECONDARY I/O CHANNEL
     50   0092 C5   9A  066E  1568 5$:       MOVZBL  UCB$B_FEX(R5),R0           ;RETRIEVE FUNCTION INDEX
                        0673  1569          EXFUNC  20$                        ;EXECUTE FUNCTION
            05   E0    0678  1570          BBS     #MT_DS_V_PES,-             ; If at 1600 BPI, branch around test
       0B 00B8 C5         067A  1571                  UCB$W_MT_DS(R5),10$        ;  for minimum length record.
                        067E  1572 ; Before testing for short record on 800 bpi, test for Tape Mark.
       45 52    02   E0  067E  1573          BBS     #MT_DS_V_TM,R2,40$         ; If SET, Tape Mark encountered
     00BC C5    0E   B1  0682  1574          CMPW    #MIN_RECORD,UCB$W_MT_FC(R5) ;MINIMUM RECORD READ?
                E5   1A  0687  1575          BGTRU   5$                         ;IF GTRU NO
                0E   E1  0689  1576 10$:     BBC     #IO$V_DATACHECK,-
       38 009A C5         068B  1577                  UCB$W_FUNC(R5),40$         ; If CLR, no data check.
                        068F  1578          EXFUNC  DOUBLE,F_INTSPCREV         ;SPACE RECORD REVERSE
                        0697  1579          EXFUNC  20$,F_WRITECHECK           ;WRITE CHECK DATA
                26   11  069F  1580          BRB     40$                        ;
                        06A1  1581
                        06A1  1582 ;
                        06A1  1583 ; FUNCTION ENDED IN AN ERROR
                        06A1  1584 ;
                        06A1  1585 ; THE ERROR COULD BE A NONFATAL CONTROLLER OR DRIVE ERROR. FATAL ERRORS TERMINATE
                        06A1  1586 ; THE FUNCTION IN THE FUNCTION EXECUTOR.
                        06A1  1587 ;
                        06A1  1588 ; FATAL CONTROLLER ERRORS ARE:
                        06A1  1589 ;
                        06A1  1590 ;       ERCONF  = ERROR CONFIRMATION.
                        06A1  1591 ;       ISTO    = INTERFACE SEQUENCE TIMEOUT.
                        06A1  1592 ;       PGE     = PROGRAMMING ERROR.
                        06A1  1593 ;       NED     = NONEXISTENT DRIVE.
                        06A1  1594 ;       RDTO    = READ DATA TIMEOUT.
                        06A1  1595 ;
                        06A1  1596 ; FATAL DRIVE ERRORS ARE:
                        06A1  1597 ;
                        06A1  1598 ;       ILF     = ILLEGAL FUNCTION.
                        06A1  1599 ;       ILR     = ILLEGAL REGISTER.
                        06A1  1600 ;       NEF     = NONEXECUTABLE FUNCTION.
                        06A1  1601 ;       RMR     = REGISTER MODIFY REFUSE.
                        06A1  1602 ;       UNS     = UNSAFE.
                        06A1  1603 ;
                        06A1  1604 ; IGNORED DRIVE ERRORS ARE:
                        06A1  1605 ;
                        06A1  1606 ;       DPAR    = DATA BUS PARITY.
                        06A1  1607 ;
                        06A1  1608 ; NOTE THAT IT IS ASSUMED THAT MASSBUS EXCEPTION (MBEXC) WILL OCCUR ONLY IN
                        06A1  1609 ; COMBINATION WITH ANOTHER DRIVE OR CONTROLLER ERROR.
                        06A1  1610 ;
                        06A1  1611
     51   00024974 8F   D3  06A1  1612 20$:     BITL    #MBA$M_SR_DLT!-            ;DATA LATE OR,
                        06A8  1613                  MBA$M_SR_INVMAP!-          ;INVALID MAP REGISTER OR,
                        06A8  1614                  MBA$M_SR_MAPPE!-           ;MAP PARITY ERROR OR,
                        06A8  1615                  MBA$M_SR_MCPE!-            ;MASSBUS CONTROL PARITY ERROR OR,
```

```
                     06A8  1616                MBA$M_SR_SPE!-          ;MBA SILO PARITY ERROR OR,
                     06A8  1617                MBA$M_SR_MDPE!-         ;MASSBUS DATA PARITY ERROR OR,
                     06A8  1618                MBA$M_SR_MXF!-          ;MISSED TRANSFER OR,
                     06A8  1619                MBA$M_SR_RDS,R1         ;READ DATA SUBSTITUTE?
            62   12  06A8  1620          BNEQ  80$                     ;IF NEQ YES
      07 52    02   E0  06AA  1621          BBS   #MT_DS_V_TM,R2,30$    ;IF SET, TAPE MARK DETECTED
   51   0600 8F   B3  06AE  1622          BITW  #MBA$M_SR_WCKLWR!-      ;WRITE CHECK LOWER BYTE OR,
                     06B3  1623                MBA$M_SR_WCKUPR,R1      ;WRITE CHECK UPPER BYTE?
            57   12  06B3  1624          BNEQ  80$                     ;IF NEQ YES
   50   31D8 8F   B3  06B5  1625 30$:     BITW  #MT_ER_M_CPAR!-         ;CONTROL BUS PARITY ERROR OR,
                     06BA  1626                MT_ER_M_DTE!-           ;DRIVE TIMING ERROR OR,
                     06BA  1627                MT_ER_M_FMT!-           ;FORMAT ERROR OR,
                     06BA  1628                MT_ER_M_INC!-           ;INCORRECTABLE ERROR (PE) OR,
                     06BA  1629                MT_ER_M_LRC!-           ;LONGITUDINAL PARITY ERROR (NRZI) OR,
                     06BA  1630                MT_ER_M_NSG!-           ;NONSTANDARD GAP OR,
                     06BA  1631                MT_ER_M_OPI!-           ;OPERATION INCOMPLETE OR,
                     06BA  1632                MT_ER_M_PEF!-           ;FORMAT ERROR (PE, OR,
                     06BA  1633                MT_ER_M_VPE,R0          ;VERTICLE PARITY ERROR (NRZI)?
            43   12  06BA  1634          BNEQ  70$                     ;IF NEQ YES
      07 52    05   E0  06BC  1635          BBS   #MT_DS_V_PES,R2,40$   ;IF SET, PHASE ENCODED TAPE
   50   8400 8F   B3  06C0  1636          BITW  #MT_ER_M_CRC!MT_ER_M_ITM,R0 ;CRC OR INVALID TAPE MARK?
            45   12  06C5  1637          BNEQ  80$                     ;IF NEQ YES
                     06C7  1638
                     06C7  1639 ;
                     06C7  1640 ; FRAME COUNT ERROR
                     06C7  1641 ;
                     06C7  1642 ; IF THE RECORD CONTAINED MORE BYTES THAN THE SPECIFIED BUFFER, THEN A DATA OVERRUN
                     06C7  1643 ; ERROR IS RETURNED. ELSE A CHECK IS MADE FOR END OF FILE AND IMPLICIT WRITECHECK.
                     06C7  1644 ;
                     06C7  1645
   50   0838 8F   3C  06C7  1646 40$:     MOVZWL  #SS$_DATAOVERUN,R0    ;SET DATA OVER RUN STATUS
7E A5   00BC C5   B1  06CC  1647          CMPW    UCB$W_MT_FC(R5),UCB$W_BCNT(R5) ;DATA OVER RUN?
            24   1A  06D2  1648          BGTRU   60$                   ;IF GTR YES
   50   0870 8F   3C  06D4  1649          MOVZWL  #SS$_ENDOFFILE,R0     ;ASSUME END OF FILE
     1B 52    02   E0  06D9  1650          BBS     #MT_DS_V_TM,R2,60$    ;IF SET, TAPE MARK ENCOUNTERED
     07 52    05   E0  06DD  1651          BBS     #MT_DS_V_PES,R2,45$   ; If at 1600, branch around test of
                     06E1  1652                                        ;  minimum length record.
   00BC C5    0E   B1  06E1  1653          CMPW    #MIN_RECORD,UCB$W_MT_FC(R5) ;MINIMUM RECORD READ?
            86   1A  06E6  1654          BGTRU   5$                    ;IF GTRU NO
                     06E8  1655 45$:
            0E   E1  06E8  1656          BBC     #IO$V_DATACHECK,-
      07 009A C5       06EA  1657                  UCB$W_FUNC(R5),50$    ; If CLR, no data check.
   0093 C5    0C   91  06EE  1658          CMPB    #CDF_READDATA,UCB$B_CEX(R5) ;LAST FUNCTION READ DATA?
            94   13  06F3  1659          BEQL    10$                   ;IF EQL YES
                     06F5  1660 50$:
      50    01   3C  06F5  1661          MOVZWL  S^#SS$_NORMAL,R0      ;SET NORMAL COMPLETION STATUS
      0080 C5    D6  06F8  1662 60$:     INCL    UCB$L_RECORD(R5)      ;INCREMENT RECORD POSITION
          0374   31  06FC  1663          BRW     FUNCXT                ;
                     06FF  1664
                     06FF  1665 ;
                     06FF  1666 ; RECOVERABLE CONTROLLER OR DRIVE ERROR
                     06FF  1667 ;
                     06FF  1668 ; A CHECK IS MADE TO DETERMINE IF OPERATION INCOMPLETE IS THE ONLY ERROR AND THAT
                     06FF  1669 ; THE TRANSFERED BYTE COUNT WAS ZERO. IF THIS CHECK IS SATISFIED, THEN BLANK TAPE
                     06FF  1670 ; IS BEING READ AND THE FATAL ERROR EXIT IS TAKEN.
                     06FF  1671 ;
                     06FF  1672 ; A CHECK IS MADE TO DETERMINE IF ANY RETRIES REMAIN. IF THE NUMBER OF RETRIES IS
```

```
                     06FF  1673 ; EXHAUSTED, THEN AN ERROR IS RETURNED. ELSE RECOVERY IS ATTEMPTED. THE METHOD USED
                     06FF  1674 ; DEPENDS ON THE PREVIOUS NUMBER OF RETRIES.
                     06FF  1675 ;
                     06FF  1676 ; IF THE PREVIOUSLY ATTEMPTED RETRIES HAVE EXCEEDED A THRESHOLD, THEN THE TAPE IS
                     06FF  1677 ; BACKSPACED THE ERROR BACKSPACE COUNT, FORWARD SPACED THAT NUMBER MINUS ONE, AND
                     06FF  1678 ; THE OPERATION IS REPEATED.
                     06FF  1679 ;
                     06FF  1680 ; IF THE PREVIOULY ATTEMPTED RETRIES HAVE NOT EXCEEDED THE THRESHOLD, THEN THE TAPE
                     06FF  1681 ; IS BACKSPACED ONE RECORD AND THE OPERATION IS RETRIED.
                     06FF  1682 ;
                     06FF  1683
          50   DFFF 8F   B3 06FF  1684 70$:    BITW    #^C<MT_ER_M_OPI>,R0      ;OPERATION INCOMPLETE ONLY ERROR?
                     06   12 0704  1685         BNEQ    80$                      ;IF NEQ NO
               00BC  C5   B5 0706  1686         TSTW    UCB$W_MT_FC(R5)          ;TRANSFERED BYTE COUNT ZERO?
                     6E   13 070A  1687         BEQL    140$                     ;IF EQL YES
                          070C  1688 80$:     TESTR   130$                     ;TEST REMAINING RETRIES
          0080 C5    08   91 0711  1689         CMPB    #THRESHOLD,UCB$B_ERTCNT(R5) ;TIME TO USE ALTERNATE RECOVERY?
                     57   1B 0716  1690         BLEQU   120$                     ;IF LEQU NO
          00B0 C5    05   D1 0718  1691         CMPL    #ERR_SPACING,UCB$L_RECORD(R5) ;ENOUGH RECORDS WRITTEN ON TAPE?
                     28   1B 071D  1692         BLEQU   90$                      ;IF LEQU YES
                          071F  1693         RELSCHAN                         ;RELEASE SECONDARY CHANNEL
                          0725  1694         EXFUNC  DOUBLE,F_REWIND          ;REWIND TAPE
                          072D  1695         REQPCHAN                         ;REQUEST PRIMARY CHANNEL
               53   54   D0 0733  1696         MOVL    R4,R3                    ;SET ADDRESS OF DRIVE CONTROL REGISTER
                          0736  1697         REQSCHAN                         ;REQUEST SECONDARY CHANNEL
     00C4 C5   00B0 C5   B0 073C  1698         MOVW    UCB$L_RECORD(R5),UCB$W_MT_FORCNT(R5) ;SET PROPER RECORD COUNT
                     1A   12 0743  1699         BNEQ    110$                     ; NEQ implies that we were not originally
                          0745  1700                                          ;  at BOT, so we space forward to where
                          0745  1701                                          ;  were.
                     30   11 0745  1702         BRB     130$                     ; Else if we were at BOT we branch around.
     00C4 C5    05   B0 0747  1703 90$:     MOVW    #ERR_SPACING,UCB$W_MT_FORCNT(R5) ;SET RECORD COUNT
                          074C  1704 100$:    EXFUNC  DOUBLE,F_INTSPCREV       ;SPACE RECORD REVERSE
          00C4 C5   B7 0754  1705         DECW    UCB$W_MT_FORCNT(R5)      ;ANY MORE RECORDS TO SPACE?
                     F2   14 0758  1706         BGTR    100$                     ;IF GTR YES
     00C4 C5    04   B0 075A  1707         MOVW    #ERR_SPACING-1,UCB$W_MT_FORCNT(R5) ;SET FORWARD SPACING COUNT
                          075F  1708 110$:    EXFUNC  DOUBLE,F_INTSPCFOR       ;SPACE RECORD FORWARD
          00C4 C5   B7 0767  1709         DECW    UCB$W_MT_FORCNT(R5)      ;ANY MORE RECORDS TO SPACE OVER?
                     F2   14 076B  1710         BGTR    110$                     ;IF GTR YES
                     08   11 076D  1711         BRB     130$                     ;
                          076F  1712 120$:    EXFUNC  DOUBLE,F_INTSPCREV       ;SPACE RECORD REVERSE
                FEF4   31 0777  1713 130$:    BRW     5$                       ;
                0287   31 077A  1714 140$:    BRW     FATALERR                 ;
```

```
                                077D  1716          .SBTTL  READ DATA REVERSE AND WRITECHECK DATA REVERSE FUNCTIONS
                                077D  1717 ;
                                077D  1718 ; READ DATA REVERSE AND WRITECHECK DATA REVERSE FUNCTIONS
                                077D  1719 ;
                                077D  1720
                                077D  1721 WRITECHECKR:                                 ;WRITE CHECK REVERSE
009A C5   4000 8F    AA         077D  1722          BICW    #IO$M_DATACHECK,UCB$W_FUNC(R5) ;CLEAR DATA CHECK REQUEST
                                0784  1723 READDATAR:                                  ;READ DATA REVERSE
                                0784  1724          REQSCHAN                           ;REQUEST SECONDARY I/O CHANNEL
     50   0092 C5    9A         078A  1725 5$:      MOVZBL  UCB$B_FEX(R5),R0           ;RETRIEVE FUNCTION INDEX
                                078F  1726          EXFUNC  20$                        ;EXECUTE FUNCTION
       0F 52    05    E0         0794  1727          BBS     #MT_DS_V_PES,R2,10$        ; If at 1600 BPI, branch around test
                                0798  1728                                             ;  for minimum length record.
                                0798  1729 ; Before testing for short record on 800 bpi, test for Tape Mark or BOT.
       49 52    02    E0         0798  1730          BBS     #MT_DS_V_TM,R2,40$         ; If SET, Tape Mark encountered
       45 52    01    E0         079C  1731          BBS     #MT_DS_V_BOT,R2,40$        ; If SET, Beginning Of Tape
    00BC C5     0E    B1         07A0  1732          CMPW    #MIN_RECORD,UCB$W_MT_FC(R5) ;MINIMUM RECORD READ?
                E3    1A         07A5  1733          BGTRU   5$                         ;IF GTRU NO
                0E    E1         07A7  1734 10$:     BBC     #IO$V_DATACHECK,-
       38 009A C5               07A9  1735                  UCB$W_FUNC(R5),40$         ; If CLR, no data check.
                                07AD  1736          EXFUNC  DOUBLE,F_INTSPCFOR         ;SPACE RECORD FORWARD
                                07B5  1737          EXFUNC  20$,F_WRITECHECKR          ;WRITE CHECK DATA REVERSE
                26    11         07BD  1738          BRB     40$                        ;
                                07BF  1739
                                07BF  1740 ;
                                07BF  1741 ; FUNCTION ENDED IN AN ERROR
                                07BF  1742 ;
                                07BF  1743 ; THE ERROR COULD BE A NONFATAL CONTROLLER OR DRIVE ERROR. FATAL ERRORS TERMINATE
                                07BF  1744 ; THE FUNCTION IN THE FUNCTION EXECUTOR.
                                07BF  1745 ;
                                07BF  1746 ; FATAL CONTROLLER ERRORS ARE:
                                07BF  1747 ;
                                07BF  1748 ;       ERCONF  = ERROR CONFIRMATION.
                                07BF  1749 ;       ISTO    = INTERFACE SEQUENCE TIMEOUT.
                                07BF  1750 ;       PGE     = PROGRAMMING ERROR.
                                07BF  1751 ;       NED     = NONEXISTENT DRIVE.
                                07BF  1752 ;       RDTO    = READ DATA TIMEOUT.
                                07BF  1753 ;
                                07BF  1754 ; FATAL DRIVE ERRORS ARE:
                                07BF  1755 ;
                                07BF  1756 ;       ILF     = ILLEGAL FUNCTION.
                                07BF  1757 ;       ILR     = ILLEGAL REGISTER.
                                07BF  1758 ;       RMR     = REGISTER MODIFY REFUSE.
                                07BF  1759 ;       UNS     = UNSAFE.
                                07BF  1760 ;
                                07BF  1761 ; IGNORED DRIVE ERRORS ARE:
                                07BF  1762 ;
                                07BF  1763 ;       DPAR    = DATA BUS PARITY.
                                07BF  1764 ;
                                07BF  1765 ; NOTE THAT IT IS ASSUMED THAT MASSBUS EXCEPTION (MBEXC) WILL OCCUR ONLY IN
                                07BF  1766 ; COMBINATION WITH ANOTHER DRIVE OR CONTROLLER ERROR.
                                07BF  1767 ;
                                07BF  1768
  51   00024974 8F    D3         07BF  1769 20$:     BITL    #MBA$M_SR_DLT!-            ;DATA LATE OR,
                                07C6  1770                  MBA$M_SR_INVMAP!-          ;INVALID MAP REGISTER OR,
                                07C6  1771                  MBA$M_SR_MAPPE!-           ;MAP PARITY ERROR OR,
                                07C6  1772                  MBA$M_SR_MCPE!-            ;MASSBUS CONTROL PARITY ERROR OR,
```

```
                        07C6  1773                MBA$M_SR_SPE!-          ;MBA SILO PARITY ERROR OR,
                        07C6  1774                MBA$M_SR_MDPE!-         ;MASSBUS DATA PARITY ERROR OR,
                        07C6  1775                MBA$M_SR_MXF!-          ;MISSED TRANSFER OR,
                        07C6  1776                MBA$M_SR_RDS,R1         ;READ DATA SUBSTITUTE OR,
                  5C 12 07C6  1777          BNEQ    70$                   ;IF NEQ YES
              07 52  02 E0 07C8  1778          BBS     #MT_DS_V_TM,R2,30$    ;IF SET, TAPE MARK DETECTED
           51 0600 8F B3 07CC  1779          BITW    #MBA$M_SR_WCKLWR!-    ;WRITE CHECK LOWER BYTE OR,
                        07D1  1780                MBA$M_SR_WCKUPR,R1      ;WRITE CHECK UPPER BYTE?
                  51 12 07D1  1781          BNEQ    70$                   ;IF NEQ YES
           50   31D8 8F B3 07D3  1782 30$:     BITW    #MT_ER_M_CPAR!-       ;CONTROL BUS PARITY ERROR OR,
                        07D8  1783                MT_ER_M_DTE!-           ;DRIVE TIMING ERROR OR,
                        07D8  1784                MT_ER_M_FMT!-           ;FORMAT ERROR OR,
                        07D8  1785                MT_ER_M_INC!-           ;INCORRECTABLE ERROR (PE) OR,
                        07D8  1786                MT_ER_M_LRC!-           ;LONGITUDINAL PARITY ERROR (NRZI) OR,
                        07D8  1787                MT_ER_M_NSG!-           ;NONSTANDARD GAP OR,
                        07D8  1788                MT_ER_M_OPI!-           ;OPERATION INCOMPLETE OR,
                        07D8  1789                MT_ER_M_PEF!-           ;FORMAT ERROR (PE) OR,
                        07D8  1790                MT_ER_M_VPE,R0          ;VERTICLE PARITY ERROR (NRZI)?
                  4A 12 07D8  1791          BNEQ    70$                   ;IF NEQ YES
              07 52  05 E0 07DA  1792          BBS     #MT_DS_V_PES,R2,40$   ;IF SET, PHASE ENCODED TAPE
           50   8400 8F B3 07DE  1793          BITW    #MT_ER_M_CRC!MT_ER_M_ITM,R0 ;CRC OR INVALID TAPE MARK?
                  3F 12 07E3  1794          BNEQ    70$                   ;IF NEQ YES
                        07E5  1795
                        07E5  1796 ;
                        07E5  1797 ; FRAME COUNT OR NONEXECUTABLE FUNCTION
                        07E5  1798 ;
                        07E5  1799 ; IF THE RECORD CONTAINED MORE BYTES THAN THE SPECIFIED BUFFER, THEN A DATA OVERRUN
                        07E5  1800 ; IS RETURNED. ELSE A CHECK IS MADE FOR END OF FILE, BEGINNING OF TAPE, AND
                        07E5  1801 ; IMPLICIT WRITECHECK.
                        07E5  1802 ;
                        07E5  1803
           50   0838 8F 3C 07E5  1804 40$:     MOVZWL  #SS$_DATAOVERUN,R0    ;SET DATA OVER RUN STATUS
        7E A5   00BC C5 B1 07EA  1805          CMPW    UCB$W_MT_FC(R5),UCB$W_BCNT(R5) ;DATA OVER RUN?
                  2B 1A 07F0  1806          BGTRU   60$                   ;IF GTR YES
           50   0870 8F 3C 07F2  1807          MOVZWL  #SS$_ENDOFFILE,R0     ;SET END OF FILE STATUS
             22 52  02 E0 07F7  1808          BBS     #MT_DS_V_TM,R2,60$    ;IF SET, TAPE MARK ENCOUNTERED
             1E 52  01 E0 07FB  1809          BBS     #MT_DS_V_BOT,R2,60$   ;IF SET, BEGINNING OF TAPE
             0A 52  05 E0 07FF  1810          BBS     #MT_DS_V_PES,R2,45$   ; If at 1600, branch around test of
                        0803  1811                                        ;  minimum length record.
           00BC C5   0E B1 0803  1812          CMPW    #MIN_RECORD,UCB$W_MT_FC(R5) ;MINIMUM RECORD F AD?
                  03 1B 0808  1813          BLEQU   45$
                FF7D 31 080A  1814          BRW     5$                    ;IF GTRU NO
                        080D  1815 45$:
                  0E E1 080D  1816          BBC     #IO$V_DATACHECK,-
              07 009A C5    080F  1817                  UCB$W_FUNC(R5),50$    ; If CLR, no data check.
           0093 C5   0F 91 0813  1818          CMPB    #CDF_READDATAR,UCB$B_CEX(R5) ;LAST FUNCTION READ DATA?
                  8D 13 0818  1819          BEQL    10$                   ;IF EQL YES
                        081A  1820 50$:
                50   01 3C 081A  1821          MOVZWL  S^#SS$_NORMAL,R0      ;SET NORMAL COMPLETION STATUS
                00B0 C5 D7 081D  1822 60$:     DECL    UCB$L_RECORD(R5)      ;DECREMENT TAPE POSITION
                   024F 31 0821  1823          BRW     FUNCXT                ;
                        0824  1824
                        0824  1825 ;
                        0824  1826 ; FUNCTION ENDED IN A RETRIABLE ERROR
                        0824  1827 ;
                        0824  1828
                        0824  1829 70$:     TESTR   130$                  ;TEST FOR RETRIES
```

```
                0080 C5    08   91  0829  1830          CMPB    #THRESHOLD,UCB$B_ERTCNT(R5) ;TIME TO USE ALTERNATE RECOVERY?
                           26   1B  082E  1831          BLEQU   90$                         ;IF LEQU NO
                00C4 C5    05   B0  0830  1832          MOVW    #ERR_SPACING,UCB$W_MT_FORCNT(R5) ;SET FORWARD SPACING COUNT
                                    0835  1833 80$:      EXFUNC  DOUBLE,F_INTSPCFOR          ;SPACE RECORD FORWARD
                     00C4 C5    B7   083D  1834          DECW    UCB$W_MT_FORCNT(R5)         ;ANY MORE RECORDS TO SPACE?
                           F2   14   0841  1835          BGTR    80$                         ;IF GTR YES
                00C4 C5    05   B0  0843  1836          MOVW    #ERR_SPACING,UCB$W_MT_FORCNT(R5) ;SET REVERSE SPACE COUNT
                                    0848  1837 85$:      EXFUNC  DOUBLE,F_INTSPCREV          ;SPACE RECORD REVERSE
                     00C4 C5    B7   0850  1838          DECW    UCB$W_MT_FORCNT(R5)         ;ANY MORE RECORDS TO SPACE?
                           F2   14   0854  1839          BGTR    85$                         ;IF GTR YES
                0093 C5    0D   91  0856  1840 90$:      CMPB    #CDF_WRITECHECKR,UCB$B_CDX(R5) ;WRITE CHECK REVERSE?
                           57   13  085B  1841          BEQL    120$                        ;IF EQL YES
                     00BC  C5   B1  085D  1842          CMPW    UCB$W_MT_FC(R5),-           ; See if record size > BCNT.  If so
                       7E  A5       0861  1843                  UCB$W_BCNT(R5)              ;  then we cannot read forward and
                                    0863  1844                                              ;  accomodate the record equally.
                           4F   1A  0863  1845          BGTRU   120$                        ; GTR implies yes record > BCNT.
                       7E  A5   A0  0865  1846          ADDW    UCB$W_BCNT(R5),-            ; Here we will try to read forward so
                       7C  A5       0868  1847                  UCB$W_BOFF(R5)              ;  effectively relocate the buffer to a
                     00BC  C5   A2  086A  1848          SUBW    UCB$W_MT_FC(R5),-           ;  higher offset if the record is
                       7C  A5       086E  1849                  UCB$W_BOFF(R5)              ;  smaller than the BCNT (buffersize).
                     00BC  C5   B0  0870  1850          MOVW    UCB$W_MT_FC(R5),-           ; Here we set the byte count to the
                       7E  A5       0874  1851                  UCB$W_BCNT(R5)              ;  real record size.
                                    0876  1852          EXFUNC  100$,F_READDATA             ;READ DATA FORWARD
          51  00064974 8F   D3  087E  1853 100$:      BITL    #MBA$M_SR_DLT!-             ;DATA LATE OR,
                                    0885  1854                  MBA$M_SR_INVMAP!-           ;INVALID MAP REGISTER OR,
                                    0885  1855                  MBA$M_SR_MAPPE!-            ;MAP PARITY ERROR OR,
                                    0885  1856                  MBA$M_SR_MCPE!-             ;MASSBUS CONTROL PARITY ERROR OR,
                                    0885  1857                  MBA$M_SR_SPE!-              ;MBA SILO PARITY ERROR OR,
                                    0885  1858                  MBA$M_SR_MDPE!-             ;MASSBUS DATA PARITY ERROR OR,
                                    0885  1859                  MBA$M_SR_MXF!-              ;MISSED TRANSFER OR,
                                    0885  1860                  MBA$M_SR_NED!-              ;NONEXISTENT DISK OR,
                                    0885  1861                  MBA$M_SR_RDS,R1             ;READ DATA SUBSTITUTE.
                           22   12  0885  1862          BNEQ    110$                        ;IF NEQ YES
          50  3718 8F   B3  0887  1863          BITW    #MT_ER_M_CPAR!-             ;CONTROL BUS PARITY ERROR, OR
                                    088C  1864                  MT_ER_M_DTE!-               ;DRIVE TIMING ERROR ,OR
                                    088C  1865                  MT_ER_M_FCE!-               ;FRAME COUNT ERROR OR,
                                    088C  1866                  MT_ER_M_FMT!-               ;FORMAT ERROR OR,
                                    088C  1867                  MT_ER_M_ITM!-               ;INVALID TAPE MARK OR,
                                    088C  1868                  MT_ER_M_NSG!-               ;NONSTANDARD GAP OR,
                                    088C  1869                  MT_ER_M_OPI,R0              ;OPERATION INCOMPLETE.
                           1B   12  088C  1870          BNEQ    110$                        ;IF NEQ YES
                                    088E  1871          EXFUNC  DOUBLE,F_INTSPCREV          ;SPACE RECORD REVERSE
                                    0896  1872          EXFUNC  110$,F_READDATA             ;READ DATA FORWARD
                                    089E  1873          EXFUNC  DOUBLE,F_INTSPCREV          ;SPACE RECORD REVERSE
                         FEFE  31   08A6  1874          BRW     10$                         ;
                      C080 C5   96  08A9  1875 110$:     INCB    UCB$B_ERTCNT(R5)            ;BIAS ERROR RETRY COUNT
                                    08AD  1876          TESTR   120$                        ;TEST FOR TAPE MOVEMENT
                           08   11  08B2  1877          BRB     130$                        ;
                                    08B4  1878 120$:     EXFUNC  DOUBLE,F_INTSPCFOR          ;SPACE RECORD FORWARD
                                    08BC  1879 130$:
                50  58 A5   D0  08BC  1880          MOVL    UCB$L_IRP(R5),R0            ; Restore original transfer parameters
                                    08C0  1881                                              ;  in case they have been altered above.
                                    08C0  1882          ASSUME  IRP$W_BOFF+2 EQ IRP$W_BCNT
                                    08C0  1883          ASSUME  UCB$W_BOFF+2 EQ UCB$W_BCNT
                                    08C0  1884
                     30 A0   D0  08C0  1885          MOVL    IRP$W_BOFF(R0),-            ; Restore transfer parameters.
                     7C A5       08C3  1886                  UCB$W_BOFF(R5)
```

```
FEC2   31   08C5   1887          BRW     5$                      ; Go back and try again.
```

```
                        08C8  1889          .SBTTL  WRITE DATA FORWARD FUNCTION
                        08C8  1890 ;
                        08C8  1891 ; WRITE DATA FORWARD FUNCTION
                        08C8  1892 ;
                        08C8  1893
                        08C8  1894 WRITEDATA:                                   ;WRITE DATA FORWARD
                        08C8  1895          REQSCHAN                            ;REQUEST SECONDARY CHANNEL
                        08CE  1896 5$:      EXFUNC  20$,F_WRITEDATA             ;EXECUTE FUNCTION
10 009A C5    0E    E1  08D6  1897          BBC     #IO$V_DATACHECK,UCB$W_FUNC(R5),10$ ;IF CLR, NO DATA CHECK
                        08DC  1898          EXFUNC  DOUBLE,F_INTSPCREV          ;SPACE RECORD REVERSE
                        08E4  1899          EXFUNC  50$,F_WRITECHECK            ;WRITE CHECK DATA
00BC C5    7E A5    B0  08EC  1900 10$:     MOVW    UCB$W_BCNT(R5),UCB$W_MT_FC(R5) ;SET TRANSFER BYTE COUNT
         00B0 C5    D6  08F2  1901          INCL    UCB$L_RECORD(R5)            ;INCREMENT TAPE POSITION
         50    01    3C  08F6  1902          MOVZWL  S^#SS$_NORMAL,R0            ;SET NORMAL COMPLETION STATUS
              0177   31  08F9  1903          BRW     FUNCXT                      ;
                        08FC  1904
                        08FC  1905 ;
                        08FC  1906 ; WRITE FUNCTION ENDED IN AN ERROR
                        08FC  1907 ;
                        08FC  1908 ; THE ERROR COULD BE A NONFATAL CONTROLLER OR DRIVE ERROR. FATAL ERRORS TERMINATE
                        08FC  1909 ; THE FUNCTION IN THE FUNCTION EXECUTOR.
                        08FC  1910 ;
                        08FC  1911 ; FATAL CONTROLLER ERRORS ARE:
                        08FC  1912 ;
                        08FC  1913 ;       ERCONF  = ERROR CONFIRMATION.
                        08FC  1914 ;       ISTO    = INTERFACE SEQUENCE TIMEOUT.
                        08FC  1915 ;       PGE     = PROGRAMMING ERROR.
                        08FC  1916 ;       NED     = NONEXISTENT DRIVE.
                        08FC  1917 ;       RDTO    = READ DATA TIMEOUT.
                        08FC  1918 ;
                        08FC  1919 ; FATAL DRIVE ERRORS ARE:
                        08FC  1920 ;
                        08FC  1921 ;       ILF     = ILLEGAL FUNCTION.
                        08FC  1922 ;       ILR     = ILLEGAL REGISTER.
                        08FC  1923 ;       NEF     = NONEXECUTABLE FUNCTION.
                        08FC  1924 ;       RMR     = REGISTER MODIFY REFUSE.
                        08FC  1925 ;       UNS     = UNSAFE.
                        08FC  1926 ;
                        08FC  1927 ; IGNORED DRIVE ERRORS ARE:
                        08FC  1928 ;
                        08FC  1929 ;       NONE.
                        08FC  1930 ;
                        08FC  1931
009A C5    4000 8F    A8  08FC  1932 20$:     BISW    #IO$M_DATACHECK,UCB$W_FUNC(R5) ;FORCE DATA CHECK
  00BC C5    7E A5    A0  0903  1933          ADDW    UCB$W_BCNT(R5),UCB$W_MT_FC(R5) ;CALCULATE ACTUAL BYTES TRANSFERED
                        0909  1934          TESTR   30$                         ;TEST REMAINING RETRIES
                        090E  1935          EXFUNC  DOUBLE,F_INTSPCREV          ;SPACE RECORD REVERSE
14 009A C5    0C    E0  0916  1936 30$:     BBS     #IO$V_INHEXTGAP,UCB$W_FUNC(R5),40$ ;IF SET, NO EXTENDED GAP
                        091C  1937          RELSCHAN                            ;RELEASE SECONDARY CHANNEL
                        0922  1938          EXFUNC  DOUBLE,F_ERASE              ;ERASE TAPE
                        092A  1939          REQSCHAN                            ;REQUEST SECONDARY CHANNEL
                9C    11  0930  1940 40$:     BRB     5$                          ;
                        0932  1941
                        0932  1942 ;
                        0932  1943 ; WRITECHECK FUNCTION ENDED IN ERROR
                        0932  1944 ;
                        0932  1945 ; THE WRITECHECK ERROR IS NOT COUNTED AGAINST THE RETRY COUNT, BUT RATHER THE
```

```
                     0932  1946 ; TAPE IS REPOSITIONED AND THE WRITE IS RETRIED.
                     0932  1947 ;
                     0932  1948
00BC C5   7E A5   A0 0932  1949 50$:    ADDW    UCB$W_BCNT(R5),UCB$W_MT_FC(R5) ;CALCULATE TRANSFERED BYTE COUNT
          0080 C5   96 0938  1950       INCB    UCB$B_ERTCNT(R5)        ;BIAS ERROR RETRY COUNT
                     093C  1951       TESTR   5$                      ;TEST REMAINING RETRIES
                     0941  1952       EXFUNC  DOUBLE,F_INTSPCREV      ;SPACE RECORD REVERSE
                  83   11 0949  1953       BRB     5$                      ;
```

```
                     094B  1955          .SBTTL  CHECK FOR FATAL OR RETRIABLE SPACING ERROR
                     094B  1956 ;
                     094B  1957 ; CHECK_ERROR - CHECK FOR FATAL OR RETRIABLE SPACING ERROR
                     094B  1958 ;
                     094B  1959 ; THIS ROUTINE IS CALLED FROM THE SPACE FILE FORWARD, SPACE RECORD FORWARD, SPACE
                     094B  1960 ; FILE REVERSE, AND SPACE RECORD REVERSE FUNCTION ROUTINES TO TEST WHETHER A
                     094B  1961 ; SPACING ERROR IS FATAL AND WHETHER ANY RETRIES REMAIN.
                     094B  1962 ;
                     094B  1963 ; INPUTS:
                     094B  1964 ;
                     094B  1965 ;       R0 = DRIVE ERROR REGISTER.
                     094B  1966 ;       R1 = MBA STATUS REGISTER.
                     094B  1967 ;       R2 = DRIVE STATUS REGISTER.
                     094B  1968 ;
                     094B  1969 ;       UCB$B_CEX(R5) = FUNCTION INDEX OF LAST FUNCTION EXECUTED.
                     094B  1970 ;       UCB$B_ERTCNT(R5) = NUMBER OF ERROR RETRIES REMAINING.
                     094B  1971 ;       UCB$W_BCNT(R5) = ORIGINAL SPACING COUNT.
                     094B  1972 ;       UCB$W_BOFF(R5) = REMAINING SPACING COUNT.
                     094B  1973 ;
                     094B  1974 ; OUTPUTS:
                     094B  1975 ;
                     094B  1976 ;       IF ONLY A FRAME COUNT ERROR OCCURED, OR THE FUNCTION WAS A REVERSE DIRECTION
                     094B  1977 ;       FUNCTION AND A CONTROL BUS PARITY ERROR DID NOT OCCUR, THEN AN IMMEDIATE
                     094B  1978 ;       RETURN TO THE CALLER IS EXECUTED. ELSE THE REMAINING RETRY COUNT IS DE-
                     094B  1979 ;       CREMENTED, AND IF THE RESULT IS ZERO, THEN THE FUNCTION IS TERMINATED VIA
                     094B  1980 ;       THE DOUBLE ERROR EXIT. ELSE THE FUNCTION IS REDISPATCHED.
                     094B  1981 ;
                     094B  1982
                     094B  1983 CHECK_ERROR:                                 ;
          51 8ED0    0949  1984          POPL    R1                         ;REMOVE RETURN ADDRESS FROM STACK
50   FDFF 8F   B3    094E  1985          BITW    #^C<MT_ER_M_FCE>,R0        ;RETRIABLE OR FATAL ERROR?
          21   13    0953  1986          BEQL    20$                        ;IF EQL NO
                     0955  1987
                     0955  1988 ;
                     0955  1989 ; RETRIABLE OR FATAL DRIVE ERROR
                     0955  1990 ;
                     0955  1991 ;       CPAR    = CONTROL BUS PARITY ERROR.
                     0955  1992 ;       NEF     = NONEXECUTABLE FUNCTION.
                     0955  1993 ;       OPI     = OPERATION INCOMPLETE.
                     0955  1994 ;
                     0955  1995
  24 50    03   E0    0955  1996          BBS     #MT_ER_V_CPAR,R0,40$       ;IF SET, CONTROL BUS PARITY ERROR
                     0959  1997
                     0959  1998 ;
                     0959  1999 ; ERROR WAS EITHER A NONEXECUTABLE FUNCTION OR OPERATION INCOMPLETE
                     0959  2000 ;
                     0959  2001
0093 C5   05   91    0959  2002          CMPB    #CDF_SPCFILREV,UCB$B_CEX(R5) ;SPACE FILE REVERSE?
          07   13    095E  2003          BEQL    10$                        ;IF EQL YES
0093 C5   07   91    0960  2004          CMPB    #CDF_SPCRECREV,UCB$B_CEX(R5) ;SPACE RECORD REVERSE?
          11   12    0965  2005          BNEQ    30$                        ;IF NEQ NO
  0D 52   01   E1    0967  2006 10$:     BBC     #MT_DS_V_BOT,R2,30$        ;IF CLR, NOT AT BEGINNING OF TAPE
                     096B  2007 ; Here we have a KLUDGE because the TM03 leaves a zero in the Frame Count
                     096B  2008 ;       register (thereby indicating that the skip succeeded) when a skip
                     096B  2009 ;       record reverse operation begins at the BOT.  So if NEF is set,
                     096B  2010 ;       then we make correct the value that we have for the Frame Count to
                     096B  2011 ;       be the original value.  We know that we are in this state if NEF
```

```
                     096B  2012 ;       bit is set in MT_ER regi..er.
                     096B  2013
          07 50  0B  E1  096B  2014       BBC     #MT_ER_V_NEF,R0,20$     ; If NOT NEF, then branch around.
             00BE C5 B0  096F  2015       MOVW    UCB$W_MT_SPACNT(R5),-   ; Reset spacing count if screwed up
             00BC C5     0973  2016               UCB$W_MT_FC(R5)         ;  by hardware.
                  61 17  0976  2017 20$:   JMP     (R1)                    ;
                     0978  2018
                     0978  2019 ;
                     0978  2020 ; FORCE FATAL ERROR ON NONEXECUTABLE FUNCTION OR OPERATION INCOMPLETE FOR A FORWARD
                     0978  2021 ; DIRECTION FUNCTION, OR A REVERSE DIRECTION FUNCTION THAT DID NOT END UP AT BEGINNI
                     0978  2022 ; OF TAPE.
                     0978  2023 ;
                     0978  2024
      0080 C5    01  90  0978  2025 30$:    MOVB    #1,UCB$B_ERTCNT(R5)     ;SET RETRY COUNT TO ONE
                     097D  2026
                     097D  2027 ;
                     097D  2028 ; ERROR WAS A CONTROL BUS PARITY ERROR OR A NONEXECUTABLE FUNCTION OR OPERATION
                     097D  2029 ; INCOMPLETE IN COMBINATION WITH A REVERSE DIRECTION FUNCTION
                     097D  2030 ;
                     097D  2031
    7E A5   7C A5  A3  097D  2032 40$:    SUBW3   UCB$W_BOFF(R5),UCB$W_BCNT(R5),- ;CALCULATE TOTAL SPACE COUNT
          00BC C5      0982  2033               UCB$W_MT_FC(R5)         ;
    00000000'GF  16  0985  2034               JSB     G^ERL$DEVICERR          ;LOG DEVICE ERROR
          0080 C5  97  098B  2035               DECB    UCB$B_ERTCNT(R5)        ;ANY RETRIES REMAINING?
               6F  13  098F  2036               BEQL    DOUBLE                  ;IF EQL NO
             F931  31  0991  2037               BRW     FDISPATCH               ;
```

```
                     0994  2039          .SBTTL  TEST FOR REMAINING RETRIES
                     0994  2040 ;
                     0994  2041 ; TESTR - TEST FOR REMAINING RETRIES
                     0994  2042 ;
                     0994  2043 ; THIS ROUTINE IS CALLED FROM THE READ DATA, READ DATA REVERSE, WRITECHECK,
                     0994  2044 ; WRITECHECK REVERSE, WRITE DATA, AND WRITE TAPE MARK FUNCTION ROUTINES TO
                     0994  2045 ; TEST FOR REMAINING RETRIES.
                     0994  2046 ;
                     0994  2047 ; INPUTS:
                     0994  2048 ;
                     0994  2049 ;       R0 = DRIVE ERROR REGISTER.
                     0994  2050 ;       R1 = MBA STATUS REGISTER.
                     0994  2051 ;       R2 = DRIVE STATUS REGISTER.
                     0994  2052 ;
                     0994  2053 ;       UCB$B_CEX(R5) = FUNCTION INDEX OF LAST FUNCTION EXECUTED.
                     0994  2054 ;       UCB$B_ERTCNT(R5) = NUMBER OF ERROR RETRIES REMAINING.
                     0994  2055 ;       UCB$L_RECORD(R5) = CURRENT TAPE POSITION BEFORE FUNCTION EXECUTION.
                     0994  2056 ;       UCB$W_MT_DS(R5) = SAVED DRIVE STATUS REGISTER.
                     0994  2057 ;       UCB$W_FUNC(R5) = ORIGINAL FUNCTION WORD.
                     0994  2058 ;       UCB$W_MT_FC(R5) = NUMBER OF BYTES THAT WERE READ OR WRITTEN TO/FROM TAPE.
                     0994  2059 ;
                     0994  2060 ;       @(SP) = SIGNED BRANCH DISPLACEMENT TO CONDITIONAL EXIT POINT.
                     0994  2061 ;
                     0994  2062 ; OUTPUTS:
                     0994  2063 ;
                     0994  2064 ;       THE REMAINING RETRY COUNT IS DECREMENTED AND IF THE RESULT IS ZERO, THEN
                     0994  2065 ;       THE FUNCTION IS TERMINATED WITH AN ERROR VIA THE FATAL ERROR EXIT AFTER
                     0994  2066 ;       HAVING ADJUSTED THE CURRENT TAPE POSITION. ELSE THE CONDITIONAL BRANCH
                     0994  2067 ;       EXIT IS TAKEN IF NO TAPE MOVEMENT OCCURED.
                     0994  2068 ;
                     0994  2069
                     0994  2070 TESTR:                                                ;
 00000000'GF   16    0994  2071          JSB     G^ERL$DEVICERR                      ;LOG DEVICE ERROR
     0080 C5   97    099A  2072          DECB    UCB$B_ERTCNT(R5)                    ;ANY RETRIES REMAINING?
           1E  13    099E  2073          BEQL    20$                                 ;IF EQL NO
           05  E0    09A0  2074          BBS     #MT_DS_V_PES,-                      ; If at 1600 BPI, branch around test
        14 00B8 C5       09A2  2075                  UCB$W_MT_DS(R5),10$             ;  for minimum length record.
    00BC C5    0E   B1    09A6  2076          CMPW    #MIN_RECORD,UCB$W_MT_FC(R5) ;ANY TAPE MOVEMENT?
           0D  1B    09AB  2077          BLEQU   10$                                 ;IF LEQU YES
 07 00B8 C5    02   E0    09AD  2078          BBS     #MT_DS_V_TM,UCB$W_MT_DS(R5),10$ ;IF SET, TAPE MARK DETECTED
      7E    00 BE    32    09B3  2079          CVTWL   @(SP),-(SP)                 ;GET DISPLACEMENT VALUE
        6E    8E   C0    09B7  2080          ADDL    (SP)+,(SP)                  ;CALCULATE BRANCH ADDRESS
        6E    02   C0    09BA  2081 10$:     ADDL    #2,(SP)                     ;
              05   09BD  2082          RSB                                         ;
                   09BE  2083
                   09BE  2084 ;
                   09BE  2085 ; ERROR RETRIES EXHAUSTED
                   09BE  2086 ;
                   09BE  2087 ; CHECK FOR TAPE MOVEMENT, ADJUST TAPE POSITION AS APPROPRIATE, AND TAKE FATAL
                   09BE  2088 ; ERROR EXIT.
                   09BE  2089 ;
                   09BE  2090
           8E   D5    09BE  2091 20$:     TSTL    (SP)+                       ;REMOVE RETURN FROM STACK
 0D 00B8 C5    02   E0    09C0  2092          BBS     #MT_DS_V_TM,UCB$W_MT_DS(R5),30$ ;IF SET, TAPE MARK DETECTED
           05   E0    09C6  2093          BBS     #MT_DS_V_PES,-              ; If at 1600 BPI, branch around test
        07 00B8 C5       09C8  2094                  UCB$W_MT_DS(R5),30$         ;  for minimum length record.
    00BC C5    0E   B1    09CC  2095          CMPW    #MIN_RECORD,UCB$W_MT_FC(R5) ;ANY TAPE MOVEMENT?
```

```
                     17   1A  09D1  2096          BGTRU   40$                     ;IF GTRU NO
              00B0 C5    D7  09D3  2097 30$:      DECL    UCB$L_RECORD(R5)        ;ASSUME REVERSE TAPE OPERATION
        0093 C5    0F   91  09D7  2098          CMPB    #CDF_READDATAR,UCB$B_CEX(R5) ;READ DATA REVERSE?
                     0C   13  09DC  2099          BEQL    40$                     ;IF EQL YES
        0093 C5    0D   91  09DE  2100          CMPB    #CDF_WRITECHECKR,UCB$B_CEX(R5) ;WRITE CHECK REVERSE?
                     05   13  09E3  2101          BEQL    40$                     ;IF EQL YES
        00B0 C5    02   C0  09E5  2102          ADDL    #2,UCB$L_RECORD(R5)     ;ADJUST FOR FORWARD TAPE OPERATION
                0017   31  09EA  2103 40$:      BRW     FATALERR                ;
```

```
                     09ED  2105          .SBTTL  TAPE POSITION LOST
                     09ED  2106 ;
                     09ED  2107 ; TAPE POSITION LOST
                     09ED  2108 ;
                     09ED  2109
                     09ED  2110 LOSTPOS:                                             ;
 46 A5   10   A8     09ED  2111          BISW    #<MT$M_LOSTa-16>,UCB$L_DEVDEPEND+2(R5) ;SET TAPE POSITION LOST
                     09F1  2112
                     09F1  2113 ;
                     09F1  2114 ; TEST FOR RETRY
                     09F1  2115 ;
                     09F1  2116
                     09F1  2117 RETRY:                                               ;TEST FOR RETRY
00000000'GF   16     09F1  2118          JSB     G^ERL$DEVICERR                      ;LOG DEVICE ERROR
    0080 C5   97     09F7  2119          DECB    UCB$B_ERTCNT(R5)                    ;ANY RETRIES REMAINING?
         07   13     09FB  2120          BEQL    FATALERR                            ;IF EQL NO
       F8C5   31     09FD  2121          BRW     FDISPATCH                           ;
                     0A00  2122
                     0A00  2123 ;
                     0A00  2124 ; DOUBLE ERROR WHILE TRYING TO REPOSITION TAPE - TAPE POSITION LOST
                     0A00  2125 ;
                     0A00  2126
                     0A00  2127 DOUBLE:                                              ;
 46 A5   10   A8     0A00  2128          BISW    #<MT$M_LOSTa-16>,UCB$L_DEVDEPEND+2(R5) ;SET LOST POSITION STATUS
                     0A04  2129
                     0A04  2130 ;
                     0A04  2131 ; FATAL CONTROLLER/DRIVE ERROR, ERROR RETRY COUNT EXHAUSTED, ERROR RETRY
                     0A04  2132 ; INHIBITED, OR FINAL OFFSET TRIED
                     0A04  2133 ;
                     0A04  2134
                     0A04  2135 FATALERR:                                            ;FATAL ERROR - SET STATUS
     53   50   D0     0A04  2136          MOVL    R0,R3                               ;COPY ERROR STATUS REGISTER
50   01A4   8F   3C   0A07  2137          MOVZWL  #SS$_MEDOFL,R0                      ;SET MEDIUM OFFLINE STATUS
  63 52     0C   E1   0A0C  2138          BBC     #MT_DS_V_MOL,R2,FUNCXT              ;IF CLR, MEDIUM OFFLINE
  09 53     0B   E1   0A10  2139          BBC     #MT_ER_V_NEF,R3,10$                 ;IF CLR, EXECUTABLE FUNCTION
50   025C   8F   3C   0A14  2140          MOVZWL  #SS$_WRITLCK,R0                     ;SET WRITE LOCK ERROR STATUS
  56 52     0B   E0   0A19  2141          BBS     #MT_DS_V_WRL,R2,FUNCXT              ;IF SET, DRIVE HARDWARE WRITE LOCKED
50   023C   8F   3C   0A1D  2142 10$:     MOVZWL  #SS$_UNSAFE,R0                      ;SET DRIVE UNSAFE STATUS
  4D 53     0E   E0   0A22  2143          BBS     #MT_ER_V_UNS,R3,FUNCXT              ;IF SET, DRIVE UNSAFE
50   02D4   8F   3C   0A26  2144          MOVZWL  #SS$_OPINCOMPL,R0                   ;SET OPERATION INCOMPLETE STATUS
  44 53     0D   E0   0A2B  2145          BBS     #MT_ER_V_OPI,R3,FUNCXT              ;IF SET, OPERATION INCOMPLETE
50   00BC   8F   3C   0A2F  2146          MOVZWL  #SS$_FORMAT,R0                      ;SET FORMAT ERROR STATUS
  3B 53     04   E0   0A34  2147          BBS     #MT_ER_V_FMT,R3,FUNCXT              ;IF SET, FORMAT ERROR
50   008C   8F   3C   0A38  2148          MOVZWL  #SS$_DRVERR,R0                      ;SET DRIVE ERROR STATUS
53   1807   8F   B3   0A3D  2149          BITW    #MT_ER_M_DTE!-                      ;DRIVE TIMING ERROR OR,
                     0A42  2150                  MT_ER_M_ILF!-                       ;ILLEGAL FUNCTION OR,
                     0A42  2151                  MT_ER_M_ILR!-                       ;ILLEGAL REGISTER OR,
                     0A42  2152                  MT_ER_M_NEF!-                       ;NON-EXECUTABLE FUNCTION OR,
                     0A42  2153                  MT_ER_M_RMR,R3                      ;REGISTER MODIFY REFUSE
         2F   12     0A42  2154          BNEQ    FUNCXT                              ;IF NEQ YES
50   01F4   8F   3C   0A44  2155          MOVZWL  #SS$_PARITY,R0                      ;SET PARITY ERROR STATUS
53   85E8   8F   B3   0A49  2156          BITW    #MT_ER_M_CRC!-                      ;CRC ERROR OR,
                     0A4E  2157                  MT_ER_M_CPAR!-                      ;CONTROL BUS PARITY ERROR OR,
                     0A4E  2158                  MT_ER_M_COR!-                       ;CORRECTABLE DATA ERROR (PE) OR,
                     0A4E  2159                  MT_ER_M_CS!-                        ;CORRECTABLE SKEW (PE) OR,
                     0A4E  2160                  MT_ER_M_DPAR!-                      ;DATA PARITY ERROR OR,
                     0A4E  2161                  MT_ER_M_INC!-                       ;INCORRECTABLE ERROR (PE) OR,
```

```
                            0A4E  2162               MT_ER_M_ITM!-              ;INVALID TAPE MARK OR,
                            0A4E  2163               MT_ER_M_LRC!-              ;LONGITUDINAL PARITY ERROR (NRZI) OR,
                            0A4E  2164               MT_ER_M_NSG!-              ;NONSTANDARD GAP OR,
                            0A4E  2165               MT_ER_M_PEF!-              ;FORMAT ERROR (PE) OR,
                            0A4E  2166               MT_ER_M_VPE,R3             ;VERTICLE PARITY ERROR (NRZI)?
                   23    12 0A4E  2167        BNEQ   FUNCXT                     ;IF NEQ YES
51      00024064 8F      D3 0A50  2168        BITL   #MBA$M_SR_MAPPE!-          ;MAP PARITY ERROR OR,
                            0A57  2169               MBA$M_SR_MCPE!-            ;MASSBUS CONTROL PARITY ERROR OR,
                            0A57  2170               MBA$M_SR_SPE!-             ;MBA SILO PARITY ERROR OR,
                            0A57  2171               MBA$M_SR_MDPE!-            ;MASSBUS DATA PARITY ERROR OR,
                            0A57  2172               MBA$M_SR_RDS,R1            ;READ DATA SUBSTITUTE?
                   1A    12 0A57  2173        BNEQ   FUNCXT                     ;IF NEQ YES
     50     005C 8F      3C 0A59  2174        MOVZWL #SS$_DATACHECK,R0          ;SET DATA CHECK ERROR STATUS
     51     0600 8F      B3 0A5E  2175        BITW   #MBA$M_SR_WCKLWR!-         ;WRITE CHECK ERROR LOWER BYTE OR,
                            0A63  2176               MBA$M_SR_WCKUPR,R1         ;WRITE CHECK ERROR UPPER BYTE?
                   0E    12 0A63  2177        BNEQ   FUNCXT                     ;IF NEQ YES
     50     01C4 8F      3C 0A65  2178        MOVZWL #SS$_NONEXDRV,R0           ;SET NONEXISTENT DRIVE STATUS
       05 51     12      E0 0A6A  2179        BBS    #MBA$V_SR_NED,R1,FUNCXT    ;IF SET, NONEXISTENT DRIVE
     50     0054 8F      3C 0A6E  2180        MOVZWL #SS$_CTRLERR,R0            ;SET CONTROLLER ERROR STATUS
```

```
                     0A73  2182          .SBTTL  FUNCTION COMPLETION COMMON EXIT
                     0A73  2183 ;
                     0A73  2184 ; FUNCTION COMPLETION COMMON EXIT
                     0A73  2185 ;
                     0A73  2186 ; THIS ROUTINE IS JUMPED TO AT THE END OF ALL MAGTAPE OPERATIONS.
                     0A73  2187 ;
                     0A73  2188 ; INPUTS:
                     0A73  2189 ;
                     0A73  2190 ;       R0 = FINAL I/O COMPLETION STATUS.
                     0A73  2191 ;       R2 = DRIVE STATUS REGISTER.
                     0A73  2192 ;
                     0A73  2193 ;       UCB$B_FEX(R5) = FUNCTION EXECUTION INDEX.
                     0A73  2194 ;
                     0A73  2195 ; OUTPUTS:
                     0A73  2196 ;
                     0A73  2197 ;       THE FINAL DRIVE STATUS IS TESTED AND THE FOLLOWING STATUS BITS ARE SET
                     0A73  2198 ;       IN THE SECOND WORD OF THE DEVICE DEPENDENT CHARACTERISTICS LONGWORD.
                     0A73  2199 ;
                     0A73  2200 ;               MT$M_BOT = SET IF TAPE IS AT BEGINNING OF TAPE AT END OF FUNCTION.
                     0A73  2201 ;               MT$M_EOF = SET IF A VALID TAPE MARK WAS DETECTED DURING THE TAPE
                     0A73  2202 ;                       OPERATION.
                     0A73  2203 ;               MT$M_EOT = SET IF AN END OF TAPE CONDITION WAS PRESENT AT THE END
                     0A73  2204 ;                       OF THE TAPE OPERATION AND THE FUNCTION WAS A READ DATA
                     0A73  2205 ;                       FORWARD, WRITECHECK DATA FORWARD, WRITE DATA FORWARD, OR
                     0A73  2206 ;                       SPACE RECORD FORWARD FUNCTION.
                     0A73  2207 ;               MT$M_HWL = SET IF THE SLAVE DRIVE HAS A TAPE MOUNTED THAT DOES NOT
                     0A73  2208 ;                       CONTAIN A WRITE RING.
                     0A73  2209 ;
                     0A73  2210
                     0A73  2211 FUNCXT:                                             ;FUNCTION EXIT
51   0092 C5    9A   0A73  2212          MOVZBL  UCB$B_FEX(R5),R1                 ;GET FUNCTION DISPATCH INDEX
  46 A5    0F    AA   0A78  2213          BICW    #<MT$M_BOT!-                     ;CLEAR BEGINNING OF TAPE AND,
                     0A7C  2214                  MT$M_EOF!-                       ;END OF FILE AND,
                     0A7C  2215                  MT$M_EOT!-                       ;END OF TAPE AND,
                     0A7C  2216                  MT$M_HWL>@-16,UCB$L_DEVDEPEND+2(R5) ;HARDWARE WRITE LOCK
  0C 52    01    E1   0A7C  2217          BBC     #MT_DS_V_BOT,R2,10$              ;IF CLR, NOT AT BEGINNING OF TAPE
  46 A5    01    A8   0A80  2218          BISW    #<MT$M_BOT@-16>,UCB$L_DEVDEPEND+2(R5) ;SET BEGINNING OF TAPE
  46 A5    10    AA   0A84  2219          BICW    #<MT$M_LOST@-16>,UCB$L_DEVDEPEND+2(R5) ;CLEAR TAPE POSITION LOST
     00B0  C5    D4   0A88  2220          CLRL    UCB$L_RECORD(R5)                 ;CLEAR TAPE POSITION
  23 52    02    E1   0A8C  2221 10$:     BBC     #MT_DS_V_TM,R2,30$               ;IF CLR, NO TAPE MARK DETECTED
  46 A5    02    A8   0A90  2222          BISW    #<MT$M_EOF@-16>,UCB$L_DEVDEPEND+2(R5) ;SET END OF FILE
     51    0A    91   0A94  2223          CMPB    #CDF_WRITECHECK,R1               ;DATA TRANSFER FUNCTION?
           1A    1A   0A97  2224          BGTRU   30$                              ;IF GTRU NO
     51    13    91   0A99  2225          CMPB    #CDF_WRITEMARK,R1                ;WRITE TAPE MARK FUNCTION?
           05    13   0A9C  2226          BEQL    20$                              ;IF EQL YES
     51    10    91   0A9E  2227          CMPB    #CDF_READPRESET,R1               ;DATA TRANSFER FUNCTION?
           10    1B   0AA1  2228          BLEQU   30$                              ;IF LEQU NO
     00B4  C5    B4   0AA3  2229 20$:     CLRW    UCB$W_MT_FC(R5)                  ;CLEAR FRAME COUNT
50   005C  8F    B1   0AA7  2230          CMPW    #SS$_DATACHECK,R0                ;WRITE CHECK ERROR?
           05    12   0AAC  2231          BNEQ    30$                              ;IF NEQ NO
50   0870  8F    3C   0AAE  2232          MOVZWL  #SS$_ENDOFFILE,R0                ;SET END OF FILE STATUS
  25 52    0A    E1   0AB3  2233 30$:     BBC     #MT_DS_V_EOT,R2,40$              ;IF CLR, NOT AT END OF TAPE
     51    05    91   0AB7  2234          CMPB    #CDF_SPCFILREV,R1                ;SPACE FILE REVERSE?
           20    13   0ABA  2235          BEQL    40$                              ;IF EQL YES
     51    07    91   0ABC  2236          CMPB    #CDF_SPCRECREV,R1                ;SPACE RECORD REVERSE?
           1B    13   0ABF  2237          BEQL    40$                              ;IF EQL YES
     51    0D    91   0AC1  2238          CMPB    #CDF_WRITECHECKR,R1              ;WRITE CHECK REVERSE?
```

```
                     16   13   0AC4  2239          BEQL    40$                             ;IF EQL YES
                  51 0C   91   0AC6  2240          CMPB    #CDF_READDATA,R1                ;READ DATA?
                     11   13   0AC9  2241          BEQL    40$                             ;IF EQL YES
                  51 0F   91   0ACB  2242          CMPB    #CDF_READDATAR,R1               ;READ DATA REVERSE?
                     0C   13   0ACE  2243          BEQL    40$                             ;IF EQL YES
               46 A5 04   A8   0AD0  2244          BISW    #<MT$M_EOT@-16>,UCB$L_DEVDEPEND+2(R5) ;SET END OF TAPE
                  05 50   E9   0AD4  2245          BLBC    R0,40$                          ;IF LBC ALREADY RETURNING ERROR
            50    0878 8F   3C   0AD7  2246          MOVZWL  #SS$_ENDOFTAPE,R0               ;SET END OF TAPE STATUS
              04 52    0B   E1   0ADC  2247 40$:     BBC     #MT_DS_V_WRL,R2,50$             ;IF CLR, NOT HARDWARE WRITE LOCKED
               46 A5   08   A8   0AE0  2248          BISW    #<MT$M_HWL@-16>,UCB$L_DEVDEPEND+2(R5) ;SET HARDWARE WRITE LOCKED
                  50   DD   0AE4  2249 50$:     PUSHL   R0                              ;SAVE FINAL STATUS
             00000000'GF   16   0AE6  2250          JSB     G^IOC$DIAGBUFILL                ;FILL DIAGNOSTIC BUFFER IF PRESENT
          02 AE    00BC C5   B0   0AEC  2251          MOVW    UCB$W_MT_FC(R5),2(SP)           ;SET BYTES TRANSFERED OR RECORDS/FILES SKIPP
                36 6E   E8   0AF2  2252          BLBS    (SP),70$                        ;IF LBS SUCCESSFUL COMPLETION
            54    58 A5   D0   0AF5  2253          MOVL    UCB$L_IRP(R5),R4                ;GET ADDRESS OF CURRENT I/O PACKET
          2D 2A A4    04   E1   0AF9  2254          BBC     #IRP$V_VIRTUAL,IRP$W_STS(R4),70$ ;IF CLR, NOT VIRTUAL FUNCTION
            54    18 A4   D0   0AFE  2255          MOVL    IRP$L_WIND(R4),R4               ;GET ADDRESS OF WINDOW BLOCK
                  16 A4   B4   0B02  2256          CLRW    WCB$W_NMAP(R4)                  ;CLEAR NUMBER OF MAPPING POINTERS
            54    34 A5   D0   0B05  2257          MOVL    UCB$L_VCB(R5),R4                ;GET ADDRESS OF VCB LISTHEAD
            52    4C A5   9E   0B09  2258          MOVAB   UCB$L_IOQFL(R5),R2              ;GET ADDRESS OF I/O QUEUE
                  53   52   D0   0B0D  2259          MOVL    R2,R3                           ;SET ADDRESS OF PREVIOUS ENTRY
                  53   63   D0   0B10  2260 60$:     MOVL    (R3),R3                         ;GET ADDRESS OF NEXT ENTRY
                  52   53   D1   0B13  2261          CMPL    R3,R2                           ;END OF LIST?
                     13   13   0B16  2262          BEQL    70$                             ;IF EQL YES
          F3 2A A3    04   E1   0B18  2263          BBC     #IRP$V_VIRTUAL,IRP$W_STS(R3),60$ ;IF CLR, NOT VIRTUAL FUNCTION
            53    04 A3   D0   0B1D  2264          MOVL    4(R3),R3                        ;RETRIEVE ADDRESS OF PREVIOUS ENTRY
            51    00 B3   0F   0B21  2265          REMQUE  @(R3),R1                        ;REMOVE ENTRY FROM DRIVER QUEUE
            04 B4         61   0E   0B25  2266          INSQUE  (R1),@4(R4)             ;INSERT ENTRY IN BLOCKED I/O LIST
                     E5   11   0B29  2267          BRB     60$                             ;
                              0B2B  2268 70$:
                              0B2B  2269          RELCHAN                                 ; Release any channels held.
                  50 8ED0     0B31  2270          POPL    R0                              ;RETRIEVE FINAL STATUS
          51    44 A5    D0   0B34  2271 STSXIT: MOVL    UCB$L_DEVDEPEND(R5),R1          ;SET MAGTAPE STATUS AND CHARACTERISTICS
                              0B38  2272          REQCOM                                  ;COMPLETE REQUEST
```

```
0B3E  2274          .SBTTL  TM03-TE16/TU77 HARDWARE FUNCTION EXECUTION
0B3E  2275 ;
0B3E  2276 ; FEX - TM03-TE16/TU77 HARDWARE FUNCTION EXECUTION
0B3E  2277 ;
0B3E  2278 ; THIS ROUTINE IS CALLED VIA A BSB WITH A WORD IMMEDIATELY FOLLOWING THAT
0B3E  2279 ; SPECIFIES THE ADDRESS OF AN ERROR ROUTINE. ALL DATA IS ASSUMED TO HAVE BEEN
0B3E  2280 ; SET UP IN THE UCB BEFORE THE CALL. THE APPROPRIATE PARAMETERS ARE LOADED
0B3E  2281 ; INTO DEVICE REGISTERS AND THE FUNCTION IS INITIATED. IF THE FUNCTION IS AN
0B3E  2282 ; IMMEDIATE FUNCTION CONTROL RETURNS IMMEDIATELY. ELSE THE RETURN ADDRESS
0B3E  2283 ; IS STORED IN THE UCB AND A WAITFOR INTERRUPT IS EXECUTED. WHEN THE INTER-
0B3E  2284 ; RUPT OCCURS, CONTROL IS RETURNED TO THE CALLER.
0B3E  2285 ;
0B3E  2286 ; INPUTS:
0B3E  2287 ;
0B3E  2288 ;       R0 = FUNCTION TABLE DISPATCH INDEX.
0B3E  2289 ;       R3 = ADDRESS OF DRIVE CONTROL STATUS REGISTER 1.
0B3E  2290 ;       R4 = ADDRESS OF MBA CONFIGURATION STATUS REGISTER.
0B3E  2291 ;       R5 = DEVICE UNIT UCB ADDRESS.
0B3E  2292 ;
0B3E  2293 ;       00(SP) = RETURN ADDRESS OF CALLER.
0B3E  2294 ;       04(SP) = RETURN ADDRESS OF CALLER'S CALLER.
0B3E  2295 ;
0B3E  2296 ;       IMMEDIATELY FOLLOWING INLINE AT THE CALL SITE IS A WORD WHICH CONTAINS
0B3E  2297 ;       A BRANCH DESTINATION TO AN ERROR RETRY ROUTINE.
0B3E  2298 ;
0B3E  2299 ; OUTPUTS:
0B3E  2300 ;
0B3E  2301 ;       THERE ARE FOUR EXITS FROM THIS ROUTINE:
0B3E  2302 ;
0B3E  2303 ;       1. SPECIAL CONDITION - THIS EXIT IS TAKEN IF A POWER FAILURE OCCURS
0B3E  2304 ;              OR THE OPERATION TIMES OUT. IT IS A JUMP TO THE APPROPRIATE
0B3E  2305 ;              ERROR ROUTINE.
0B3E  2306 ;
0B3E  2307 ;       2. FATAL ERROR - THIS EXIT IS TAKEN IF A FATAL CONTROLLER OR DRIVE
0B3E  2308 ;              ERROR OCCURS OR IF ANY EPROR OCCURS AND ERROR RETRY IS
0B3E  2309 ;              INHIBITED. IT IS A JUMP TO THE FATAL ERROR EXIT ROUTINE.
0B3E  2310 ;
0B3E  2311 ;       3. RETRIABLE ERROR - THIS EXIT IS TAKEN IF A RETRIABLE CONTROLLER
0B3E  2312 ;              OR DRIVE ERROR OCCURS AND ERROR RETRY IS NOT INHIBITED.
0B3E  2313 ;              IT CONSISTS OF TAKING THE ERROR BRANCH EXIT.
0B3E  2314 ;
0B3E  2315 ;       4. SUCCESSFUL OPERATION - THIS EXIT IS TAKEN IF NO ERROR OCCURS
0B3E  2316 ;              DURING THE OPERATION. IT CONSISTS OF A RETURN INLINE.
0B3E  2317 ;
0B3E  2318 ;       IN ALL CASES IF AN ERROR OCCURS, AN ATTEMPT IS MADE TO LOG THE ERROR.
0B3E  2319 ;
0B3E  2320 ;       IN ALL CASES FINAL DRIVE AND CONTROLLER REGISTERS ARE RETURNED VIA
0B3E  2321 ;       THE GENERAL REGISTERS R0, R1, AND R2, AND THE UCB.
0B3E  2322 ;
0B3E  2323 ;       R0 = DRIVE ERROR REGISTER.
0B3E  2324 ;       R1 = MBA STATUS REGISTER.
0B3E  2325 ;       R2 = DRIVE STATUS REGISTER.
0B3E  2326 ;
0B3E  2327 ;       UCB$W_MT_FC(R5) = FRAME COUNT REGISTER.
0B3E  2328 ;
0B3E  2329
0B3E  2330 FEX:                                    ;FUNCTION EXECUTOR
```

```
                     009C C5 8ED0  0B3E  2331          POPL    UCB$L_DPC(R5)           ;SAVE DRIVER PC VALUE
                0093 C5    50  90  0B43  2332          MOVB    R0,UCB$B_CEX(R5)        ;SAVE CASE INDEX
                                   0B48  2333 RESTART:                                ;RESTART FUNCTION
              24 A3  00C6 C5    3C  0B48  2334          MOVZWL  UCB$W_MT_TC(R5),MT_TC(R3) ;SELECT DRIVE AND SET CHARACTERISTICS
                     63    09    9A  0B4E  2335          MOVZBL  #F_DRVCLR!1,MT_CS1(R3)  ;CLEAR DRIVE
                                   0B51  2336          CASE    R0,<-                   ;DISPATCH TO PROPER FUNCTION ROUTINE
                                   0B51  2337                  IMMED,-                 ;NO OPERATION
                                   0B51  2338                  POSIT,-                 ;UNLOAD VOLUME
                                   0B51  2339                  POSIT,-                 ;SPACE FILE FORWARD
                                   0B51  2340                  RECAL,-                 ;REWIND
                                   0B51  2341                  IMMED,-                 ;DRIVE CLEAR
                                   0B51  2342                  POSIT,-                 ;SPACE FILE REVERSE
                                   0B51  2343                  POSIT,-                 ;ERASE TAPE
                                   0B51  2344                  POSIT,-                 ;SPACE RECORD REVERSE
                                   0B51  2345                  IMMED,-                 ;NO OPERATION
                                   0B51  2346                  POSIT,-                 ;SPACE RECORD FORWARD
                                   0B51  2347                  XFER,-                  ;WRITE CHECK FORWARD
                                   0B51  2348                  XFER,-                  ;WRITE DATA FORWARD
                                   0B51  2349                  XFER,-                  ;READ DATA FORWARD
                                   0B51  2350                  XFER,-                  ;WRITECHECK REVERSE
                                   0B51  2351                  XFER,-                  ;WRITE DATA FORWARD
                                   0B51  2352                  XFER,-                  ;READ DATA REVERSE
                                   0B51  2353                  RECAL,-                 ;READ IN PRESET
                                   0B51  2354                  IMMED,-                 ;SET TAPE CHARACTERISTICS
                                   0B51  2355                  IMMED,-                 ;SENSE CHARACTERISTICS
                                   0B51  2356                  POSIT,-                 ;WRITE TAPE MARK
                                   0B51  2357                  >                       ;
                                   0B7D  2358
                                   0B7D  2359 ;
                                   0B7D  2360 ; INTERNAL SPACING FUNCTIONS
                                   0B7D  2361 ;
                                   0B7D  2362 ;       SPACE RECORD FORWARD, AND
                                   0B7D  2363 ;       SPACE RECORD REVERSE.
                                   0B7D  2364 ;
                                   0B7D  2365 ; THESE FUNCTIONS ARE EXECUTED TO EFFECT SPACING OPERATIONS DURING THE ERROR
                                   0B7D  2366 ; RECOVERY OF WRITE TAPE MARK, ERASE TAPE, AND ALL DATA TRANSFER FUNCTIONS.
                                   0B7D  2367 ; WRITE CHECK AND WRITE CHECK REVERSE FUNCTIONS ARE USED TO ACTUALLY POSITION
                                   0B7D  2368 ; THE TAPE. IF A NOISE RECORD IS ENCOUNTERED, IT IS IGNORED.
                                   0B7D  2369 ;
                                   0B7D  2370
                                   0B7D  2371 INTSPC:                                 ;INTERNAL SPACE FUNCTIONS
                         065A  30  0B7D  2372          BSBW    TM_SETINTSPC            ;SETUP FOR INTERNAL SPACE FUNCTION
                   50    0093 C5  9A  0B80  2373          MOVZBL  UCB$B_CEX(R5),R0        ;RETRIEVE FUNCTION INDEX
                         50   15  91  0B85  2374          CMPB    #CDF_INTSPCREV,R0       ;SPACE RECORD REVERSE?
                              06  12  0B88  2375          BNEQ    10$                     ;IF NEQ NO
                0C A4    01FF 8F  3C  0B8A  2376          MOVZWL  #511,MBA$L_VAR(R4)      ;ADJUST VIRTUAL ADDRESS REGISTER
                                   0B90  2377 10$:    DSBINT                          ;DISABLE INTERRUPTS
                  57 64 A5   05  E0  0B96  2378          BBS     #UCB$V_POWER,UCB$W_STS(R5),20$ ;IF SET, POWER FAILED
                 63   F486 CF 40  9A  0B9B  2379          MOVZBL  FTAB[R0],MT_CS1(R3)     ;EXECUTE FUNCTION
                                   0BA1  2380          WFIKPCH RETREG,TIME_OUT[R0]     ;WAIT FOR INTERRUPT AND KEEP CHANNEL
                00B4 C5    08 A4  D0  0BAE  2381          MOVL    MBA$L_SR(R4),UCB$L_MT_SR(R5) ;SAVE FINAL MBA STATUS
                         05FC  30  0BB4  2382          BSBW    TM_SAVDRVSTS            ;SAVE DRIVE STATUS
                                   0BB7  2383          IOFORK                          ;CREATE FORK PROCESS
                  33 64 A5   05  E0  0BBD  2384          BBS     #UCB$V_POWER,UCB$W_STS(R5),30$ ;IF SET, POWER FAILED
          00B4 C5   00013F80 8F  CA  0BC2  2385          BICL    #MBA$M_SR_ATTN!-        ;CLEAR ATTENTION AND,
                                   0BCB  2386                  MBA$M_SR_DLT!-          ;DATA LATE AND,
                                   0BCB  2387                  MBA$M_SR_DTABT!-        ;DATA TRANSFER ABORT AND,
```

```
                     0BCB  2388                 MBA$M_SR_DTCOMP!-       ;DATA TRANSFER COMPLETE AND,
                     0BCB  2389                 MBA$M_SR_MBEXC!-        ;MASSBUSS EXCEPTION AND,
                     0BCB  2390                 MBA$M_SR_MXF!-          ;MISSED TRANSFER AND,
                     0BCB  2391                 MBA$M_SR_WCKLWR!-       ;WRITE CHECK LOWER AND,
                     0BCB  2392                 MBA$M_SR_WCKUPR,UCB$L_MT_SR(R5) ;WRITE CHECK UPPER
            28   12   0BCB  2393         BNEQ    30$                     ;IF NEQ FATAL SPACING ERROR
          50   0093 C5   9A   0BCD  2394         MOVZBL  UCB$B_CEX(R5),R0        ;GET FUNCTION INDEX
    00BA C5    F4F5 CF40   AA   0BD2  2395         BICW    XTAB[R0],UCB$W_MT_ER(R5) ;CLEAR EXTRANEOUS DRIVE ERROR BITS
            19   12   0BDA  2396         BNEQ    30$                     ;IF NEQ FATAL SPACING ERROR
       13 00B8 C5   02   E0   0BDC  2397         BBS     #MT_DS_V_TM,UCB$W_MT_DS(R5),30$ ;IF SET, TAPE MARK ENCOUNTERED
            05   E0   0BE2  2398         BBS     #MT_DS_V_PES,-          ; If at 1600 BPI, branch around test
         0D 00B8 C5        0BE4  2399                 UCB$W_MT_DS(R5),30$     ;  for minimum length record.
       00BC C5   0E   B1   0BE8  2400         CMPW    #MIN_RECORD,UCB$W_MT_FC(R5) ;MINIMUM RECORD READ?
            06   1B   0BED  2401         BLEQU   30$                     ;IF LEQU YES
          FF56   31   0BEF  2402         BRW     RESTART                 ;RESTART FUNCTION
          0127   31   0BF2  2403 20$:    BRW     ENBXIT                  ;
          013E   31   0BF5  2404 30$:    BRW     RETREG                  ;
                     0BF8  2405
                     0BF8  2406 ;
                     0BF8  2407 ; POSITIONING FUNCTION EXECUTION
                     0BF8  2408 ;
                     0BF8  2409 ;       FUNCTIONS INCLUDE:
                     0BF8  2410 ;
                     0BF8  2411 ;               SPACE FILE FORWARD,
                     0BF8  2412 ;               SPACE FILE REVERSE,
                     0BF8  2413 ;               ERASE TAPE,
                     0BF8  2414 ;               SPACE RECORD REVERSE,
                     0BF8  2415 ;               SPACE RECORD FORWARD,
                     0BF8  2416 ;               UNLOAD, AND
                     0BF8  2417 ;               WRITE TAPE MARK.
                     0BF8  2418 ;
                     0BF8  2419 ; THE FUNCTION IS INITIATED AND A WAITFOR INTERRUPT IS EXECUTED. WHEN THE INTERRUPT
                     0BF8  2420 ; OCCURS FINAL DEVICE REGISTERS ARE RETURNED TO THE CALLER.
                     0BF8  2421 ;
                     0BF8  2422
                     0BF8  2423 POSIT:                                  ;POSITIONING FUNCTION EXECUTION
                     0BF8  2424         DSBINT                          ;DISABLE ALL INTERRUPTS
      2D 64 A5    05   E0   0BFE  2425         BBS     #UCB$V_POWER,UCB$W_STS(R5),ENBEXT ;IF SET, POWER HAS FAILED
    14 A3    00BE C5   32   0C03  2426         CVTWL   UCB$W_MT_SPACNT(R5),MT_FC(R3) ;LOAD FRAME COUNT REGISTER
     63    F448 CF40   9A   0C09  2427         MOVZBL  FTAB[R0],MT_CS1(R3)     ;EXECUTE FUNCTION
                     0C0F  2428         WFIKPCH RETREG,TIME_OUT[R0]     ;WAITFOR INTERRUPT AND KEEP CHANNEL
                00C9   31   0C1C  2429         BRW     SETDEN                  ;
                     0C1F  2430
                     0C1F  2431 ;
                     0C1F  2432 ; IMMEDIATE FUNCTION EXECUTION
                     0C1F  2433 ;
                     0C1F  2434 ;       FUNCTIONS INCLUDE:
                     0C1F  2435 ;
                     0C1F  2436 ;               NO OPERATION, AND
                     0C1F  2437 ;               DRIVE CLEAR.
                     0C1F  2438 ;
                     0C1F  2439 ; THESE FUNCTIONS ARE EXECUTED IMMEDIATELY AND THE FINAL DEVICE REGISTERS
                     0C1F  2440 ; ARE RETURNED TO THE CALLER.
                     0C1F  2441 ;
                     0C1F  2442
                     0C1F  2443 IMMED:                                  ;IMMEDIATE FUNCTION EXECUTION
                     0C1F  2444         DSBINT                          ;DISABLE INTERRUPTS
```

```
                06 64 A5    05    E0  0C25  2445          BBS     #UCB$V_POWER,UCB$W_STS(R5),ENBEXT ;IF SET, POWER HAS FAILED
             63   F427 CF40  9A    0C2A  2446          MOVZBL  FTAB[R0],MT_CS1(R3)     ;EXECUTE FUNCTION
                             0C30  2447 ENBEXT:                                      ;
                      00E9   31    0C30  2448          BRW     ENBXIT                  ;
                             0C33  2449
                             0C33  2450 ;
                             0C33  2451 ; RECALIBRATE FUNCTION EXECUTION
                             0C33  2452 ;
                             0C33  2453 ;       FUNCTIONS INCLUDE:
                             0C33  2454 ;
                             0C33  2455 ;               READ IN PRESET, AND
                             0C33  2456 ;               REWIND.
                             0C33  2457 ;
                             0C33  2458 ; THE FUNCTION IS INITIATED AND A WAITFOR INTERRUPT IS EXECUTED. WHEN THE INTERRUPT
                             0C33  2459 ; OCCURS A TEST IS MADE TO SEE IF POSITIONING IS STILL IN PROGRESS. IF POSITIONING
                             0C33  2460 ; IS STILL IN PROGRESS, THEN ANOTHER WAITFOR INTERRUPT IS EXECUTED. ELSE FINAL DRIVE
                             0C33  2461 ; REGISTERS ARE RETURNED TO THE CALLED.
                             0C33  2462 ;
                             0C33  2463
                             0C33  2464 RECAL:                                       ;RECALIBRATE FUNCTION EXECUTION
                             0C33  2465          DSBINT                              ;DISABLE INTERRUPTS
                F2 64 A5    05    E0  0C39  2466          BBS     #UCB$V_POWER,UCB$W_STS(R5),ENBEXT ;IF SET, POWER HAS FAILED
                 63   F413 CF40  9A    0C3E  2467          MOVZBL  FTAB[R0],MT_CS1(R3)     ;EXECUTE FUNCTION
                             0C44  2468          WFIKPCH RETREG,#60*7                ;WAITFOR INTERRUPT
             52   04 A3     12    78  0C52  2469          ASHL    #31-MT_DS_V_PIP,MT_DS(R3),R2 ;POSITIONING IN PROGRESS?
                       2C    18    0C57  2470          BGEQ    10$                 ;IF GEQ NO
                             0C59  2471          IOFORK                              ;CREATE FORK PROCESS
                             0C5F  2472          DSBINT                              ;DISABLE INTERRUPTS
             52   04 A3     12    78  0C65  2473          ASHL    #31-MT_DS_V_PIP,MT_DS(R3),R2 ;POSITIONING IN PROGRESS?
                       C4    18    0C6A  2474          BGEQ    ENBEXT              ;IF GEQ NO
               BF 64 A5    05    E0  0C6C  2475          BBS     #UCB$V_POWER,UCB$W_STS(R5),ENBEXT ;IF SET  POWER HAS FAILED
            11 009A C5     07    E0  0C71  2476          BBS     #IO$V_NOWAIT,UCB$W_FUNC(R5),20$ ;IF SET, NO WAIT FOR REWIND
                             0C77  2477          WFIRLCH RETREG,#60*7                ;WAITFOR FINAL INTERRUPT
                      0089   31    0C85  2478 10$:     BRW     DRVREG              ;
                   68 A5     01    A8  0C88  2479 20$:     BISW    #UCB$M_MT_REWIND,UCB$W_DEVSTS(R5) ;SET REWIND IN PROGRESS
            0088 C5     04 A3     F7    0C8C  2480          CVTLW   MT_DS(R3),UCB$W_MT_DS(R5) ;SAVE DRIVE STATUS
                   008A C5     B4    0C92  2481          CLRW    UCB$W_MT_ER(R5)     ;CLEAR SAVED ERROR STATUS
                             0C96  2482          ENBINT                              ;ENABLE INTERRUPTS
                      009A   31    0C99  2483          BRW     RETREG              ;
                             0C9C  2484
                             0C9C  2485 ;
                             0C9C  2486 ; TRANSFER FUNCTION EXECUTION
                             0C9C  2487 ;
                             0C9C  2488 ;       FUNCTIONS INCLUDE:
                             0C9C  2489 ;
                             0C9C  2490 ;               WRITE CHECK,
                             0C9C  2491 ;               WRITE DATA,
                             0C9C  2492 ;               READ DATA,
                             0C9C  2493 ;               READ DATA REVERSE, AND
                             0C9C  2494 ;               WRITECHECK DATA REVERSE.
                             0C9C  2495 ;
                             0C9C  2496 ; THE MAP REGISTERS, BYTE COUNT REGISTER, AND VIRTUAL ADDRESS REGISTERS ARE LOADED.
                             0C9C  2497 ; THE FUNCTION IS INITIATED AND A WAITFOR INTERRUPT IS EXECUTED. WHEN THE INTERRUPT
                             0C9C  2498 ; OCCURS FINAL DEVICE REGISTERS ARE RETURNED TO THE CALLER.
                             0C9C  2499 ;
                             0C9C  2500
                             0C9C  2501 XFER:                                        ;TRANSFER FUNCTION EXECUTION
```

```
                08 A4    00    D2  0C9C  2502          MCOML   #0,MBA$L_SR(R4)          ;CLEAR MASSBUS ADAPTER ERRORS
                                  0CA0  2503          LOADMBA                          ;LOAD MAP, BYTE COUNT, AND VIRTUAL ADDRESS
             50    0093 C5    9A  0CA6  2504          MOVZBL  UCB$B_CEX(R5),R0         ;RETRIEVE FUNCTION TABLE INDEX
                   50    0F   91  0CAB  2505          CMPB    #CDF_READDATAR,R0        ;READ DATA REVERSE?
                         05   13  0CAE  2506          BEQL    10$                      ;IF EQL YES
                   50    0D   91  0CB0  2507          CMPB    #CDF_WRITECHECKR,R0      ;WRITE CHECK DATA REVERSE?
                         0A   12  0CB3  2508          BNEQ    20$                      ;IF NEQ NO
             51     7E A5    3C  0CB5  2509 10$:      MOVZWL  UCB$W_BCNT(R5),R1        ;GET TRANSFER BYTE COUNT
                         51   D7  0CB9  2510          DECL    R1                       ;REDUCE BYTE COUNT BY ONE
                0C A4    51   C0  0CBB  2511          ADDL    R1,MBA$L_VAR(R4)         ;CALCULATE ENDING ADDRESS OF BUFFER
                                  0CBF  2512 20$:      DSBINT                           ;DISABLE INTERRUPTS
          52 64 A5       05   E0  0CC5  2513          BBS     #UCB$V_POWER,UCB$W_STS(R5),ENBXIT ;IF SET, POWER FAILED
          14 A3    10 A4       D0  0CCA  2514          MOVL    MBA$L_BCR(R4),MT_FC(R3)  ;LOAD FRAME COUNT REGISTER
       63   F382 CF40           9A  0CCF  2515          MOVZBL  FTAB[R0],MT_CS1(R3)      ;INITIATE FUNCTION
                                  0CD5  2516          WFIKPCH RETREG,TIME_OUT[R0]      ;WAITFOR INTERRUPT AND KEEP CHANNEL
    00B4 C5    08 A4       D0  0CE2  2517          MOVL    MBA$L_SR(R4),UCB$L_MT_SR(R5) ;SAVE FINAL CONTROLLER STATUS
                                  0CE8  2518 SETDEN:
                08    03    F0  0CE8  2519          INSV    #MT$K_NRZI_800,#MT$V_DENSITY,- ;SET CORRECT DENSITY CODE
                44 A5 05          0CEB  2520                  #MT$S_DENSITY,UCB$L_DEVDEPEND(R5) ;
                08    03    F0  0CEE  2521          INSV    #3,#MT_TC_V_DEN,-        ;
          00C6 C5     03          0CF1  2522                  #MT_TC_S_DEN,UCB$W_MT_TC(R5) ;
                04 A3 20    D3  0CF5  2523          BITL    #MT_DS_M_PES,MT_DS(R3)   ;NRZI ENCODED TAPE?
                      16    13  0CF9  2524          BEQL    DRVREG                   ;IF EQL YES
                08    04    F0  0CFB  2525          INSV    #MT$K_PE_1600,#MT$V_DENSITY,- ;SWITCH TO PHASE ENCODED TAPE
                44 A5 05          0CFE  2526                  #MT$S_DENSITY,UCB$L_DEVDEPEND(R5) ;
                08    04    F0  0D01  2527          INSV    #4,#MT_TC_V_DEN,-        ;
          00C6 C5     03          0D04  2528                  #MT_TC_S_DEN,UCB$W_MT_TC(R5) ;
                44 A5 08    AA  0D08  2529          BICW    #MT$M_PARITY,UCB$L_DEVDEPEND(R5) ;CLEAR EVEN PARITY
          00C6 C5     08    AA  0D0C  2530          BICW    #MT_TC_M_EPAR,UCB$W_MT_TC(R5) ;
                                  0D11  2531 DRVREG:                                   ;SAVE DRIVE REGISTERS
                   049F   30    0D11  2532          BSBW    TM_SAVDRVSTS             ;SAVE DRIVE STATUS
                                  0D14  2533          IOFORK                           ;CREATE FORK PROCESS
                      1A  11    0D1A  2534          BRB     RETREG                   ;
                                  0D1C  2535
                                  0D1C  2536 ;
                                  0D1C  2537 ; ENABLE INTERRUPTS
                                  0D1C  2538 ;
                                  0D1C  2539
                                  0D1C  2540 ENBXIT:                                   ;
    00BA C5    08 A3    F7    0D1C  2541          CVTLW   MT_ER(R3),UCB$W_MT_ER(R5) ;SAVE ERROR STATUS REGISTER
    00B8 C5    04 A3    F7    0D22  2542          CVTLW   MT_DS(R3),UCB$W_MT_DS(R5) ;SAVE DRIVE STATUS REGISTER
    00C2 C5    24 A3    F7    0D28  2543          CVTLW   MT_TC(R3),UCB$W_MT_TC_SAV(R5) ;SAVE TAPE CONTROL REGISTER
       00C0 C5    63    F7    0D2E  2544          CVTLW   MT_CS1(R3),UCB$W_MT_CS1(R5) ;SAVE DRIVE CONTROL REGISTER
                                  0D33  2545          ENBINT                           ;ENABLE INTERRUPTS
                                  0D36  2546
                                  0D36  2547 ;
                                  0D36  2548 ; RETURN REGISTERS
                                  0D36  2549 ;
                                  0D36  2550
                                  0D36  2551          .ENABL  LSB
                                  0D36  2552 RETREG:                                   ;RETURN FINAL DEVICE REGISTERS
                      05    E1    0D36  2553          BBC     #MT_DS_V_PES,-           ; If NOT at 1600 BPI, branch around
             14 00B8 C5           0D38  2554                  UCB$W_MT_DS(R5),0$       ;  dead track test.
                09    00    ED    0D3C  2555          CMPZV   #0,#9,-                  ; Test for all 9 tracks dead.
                   00D0 C5        0D3F  2556                  UCB$W_MT_CC_SAV(R5),-
          000001FF 8F             0D42  2557                  #^X<1FF>
                      07    12    0D47  2558          BNEQ    0$                       ; NEQ means all 9 tracks not dead.
```

```
                0040 8F    A8  0D49  2559          BISW    #MT_ER_M_INC,-          ; Correct hardware glitch that doesn't
                00BA C5        0D4D  2560                  UCB$W_MT_ER(R5)         ;  set INC bit when all tracks dead.
                               0D50  2561 0$:
          51    0093 C5    9A  0D50  2562          MOVZBL  UCB$B_CEX(R5),R1        ;GET CURRENT FUNCTION INDEX
    50  00BA C5  F372 CF41  AB  0D55  2563          BICW3   XTAB[R1],UCB$W_MT_ER(R5),R0 ;GET FINAL DRIVE ERROR STATUS
          51    00B4 C5    D0  0D5E  2564          MOVL    UCB$L_MT_SR(R5),R1      ;RETRIEVE FINAL CONTROLLER STATUS
          52    00B8 C5    3C  0D63  2565          MOVZWL  UCB$W_MT_DS(R5),R2      ;RETRIEVE CONTENTS OF DRIVE STATUS REGISTER
       64 A5    0060 8F    B3  0D68  2566          BITW    #UCB$M_POWER!-          ;POWER FAIL OR DEVICE TIMEOUT?
                               0D6E  2567                  UCB$M_TIMOUT,UCB$W_STS(R5) ;
                      03   13  0D6E  2568          BEQL    1$                      ; EQL implies NO special condition.
                    008E   31  0D70  2569          BRW     70$                     ; NEQ implies YES - SPECIAL CONDITION
                               0D73  2570 1$:
              0093 C5   0A   91  0D73  2571          CMPB    #CDF_WRITECHECK,UCB$B_CEX(R5) ;DRIVE RELATED FUNCTION?
                        29   1A  0D78  2572          BGTRU   10$                     ;IF GTRU YES
              0093 C5   14   91  0D7A  2573          CMPB    #CDF_INTSPCFOR,UCB$B_CEX(R5) ;INTERNAL SPACE FUNCTION?
                        07   15  0D7F  2574          BLEQ    5$                      ;IF LEQ YES
              0093 C5   10   91  0D81  2575          CMPB    #CDF_READPRESET,UCB$B_CEX(R5) ;DRIVE RELATED FUNCTION?
                        1B   1B  0D86  2576          BLEQU   10$                     ;IF LEQU YES
                               0D88  2577
                               0D88  2578 ;
                               0D88  2579 ; CONTROLLER RELATED FUNCTION
                               0D88  2580 ;
                               0D88  2581
          51    000E5FFF 8F    D3  0D88  2582 5$:      BITL    #MBA$M_ERROR,R1         ;ANY CONTROLLER ERRORS?
                        53   13  0D8F  2583          BEQL    50$                     ;IF EQL NO
          51    000C000B 8F    D3  0D91  2584          BITL    #MBA$M_SR_ERCONF!-      ;ERROR CONFIRMATION OR,
                               0D98  2585                  MBA$M_SR_ISTO!-         ;INTERFACE SEQUENCE TIMEOUT OR,
                               0D98  2586                  MBA$M_SR_PGE!-          ;PROGRAMMING ERROR OR,
                               0D98  2587                  MBA$M_SR_NED!-          ;NONEXISTENT DRIVE, OR
                               0D98  2588                  MBA$M_SR_RDTO,R1        ;READ TIMEOUT?
                        53   12  0D98  2589          BNEQ    60$                     ;IF NEQ YES - FATAL CONTROLLER ERROR
    51  00024F74 8F    D3  0D9A  2590          BITL    #MBA$M_SR_DLT!-         ;DATA LATE OR,
                               0DA1  2591                  MBA$M_SR_INVMAP!-       ;INVALID MAP REGISTER OR,
                               0DA1  2592                  MBA$M_SR_MAPPE!-        ;MAP REGISTER PARITY ERROR OR,
                               0DA1  2593                  MBA$M_SR_MCPE!-         ;MASSBUS CONTROL PARITY ERROR OR,
                               0DA1  2594                  MBA$M_SR_SPE!-          ;MBA SILO PARITY ERROR OR,
                               0DA1  2595                  MBA$M_SR_MDPE!-         ;MASSBUS DATA PARITY ERROR OR,
                               0DA1  2596                  MBA$M_SR_MXF!-          ;MISSED TRANSFER OR,
                               0DA1  2597                  MBA$M_SR_RDS!-          ;READ DATA SUBSTITUTE OR,
                               0DA1  2598                  MBA$M_SR_WCKLWR!-       ;WRITE CHECK LOWER BYTE OR,
                               0DA1  2599                  MBA$M_SR_WCKUPR,R1      ;WRITE CHECK UPPER BYTE?
                      06   12  0DA1  2600          BNEQ    20$                     ;IF NEQ YES - RETRIABLE CONTROLLER ERROR
                               0DA3  2601
                               0DA3  2602 ;
                               0DA3  2603 ; DRIVE RELATED FUNCTION
                               0DA3  2604 ;
                               0DA3  2605
                      51   D4  0DA3  2606 10$:     CLRL    R1                      ;CLEAR CONTENTS OF STATUS REGISTER
                      50   B5  0DA5  2607          TSTW    R0                      ;ANY DRIVE ERRORS?
                      3B   13  0DA7  2608          BEQL    50$                     ;IF EQL NO
        50    4007 8F    B3  0DA9  2609 20$:     BITW    #MT_ER_M_ILF!-          ;ILLEGAL FUNCTION OR,
                               0DAE  2610                  MT_ER_M_ILR!-           ;ILLEGAL REGISTER OR,
                               0DAE  2611                  MT_ER_M_RMR!-           ;REGISTER MODIFY REFUSE OR,
                               0DAE  2612                  MT_ER_M_UNS,R0          ;DRIVE UNSAFE
                      3D   12  0DAE  2613          BNEQ    60$                     ;IF NEQ YES - FATAL DRIVE ERROR
          20 50       0B   E1  0DB0  2614          BBC     #MT_ER_V_NEF,R0,40$     ;IF CLR, NOT NONEXECUTABLE FUNCTION
        0093 C5       0F   91  0DB4  2615          CMPB    #CDF_READDATAR,UCB$B_CEX(R5) ;READ DATA REVERSE?
```

```
                     15   13   0DB9  2616          BEQL    30$                             ;IF EQL YES
          0093 C5    05   91   0DBB  2617          CMPB    #CDF_SPCFILREV,UCB$B_CEX(R5)  ;SPACE FILE REVERSE?
                     0E   13   0DC0  2618          BEQL    30$                             ;IF EQL YES
          0093 C5    07   91   0DC2  2619          CMPB    #CDF_SPCRECREV,UCB$B_CEX(R5)  ;SPACE RECORD REVERSE?
                     07   13   0DC7  2620          BEQL    30$                             ;IF EQL YES
          0093 C5    0D   91   0DC9  2621          CMPB    #CDF_WRITECHECKR,UCB$B_CEX(R5) ;WRITECHECK REVERSE?
                     1D   12   0DCE  2622          BNEQ    60$                             ;IF NEQ NO
               00BC C5   B4   0DD0  2623 30$:      CLRW    UCB$W_MT_FC(R5)                 ;CLEAR SAVED FRAME COUNT REGISTER
                              0DD4  2624
                              0DD4  2625 ;
                              0DD4  2626 ; RETRIABLE ERROR EXIT
                              0DD4  2627 ;
                              0DD4  2628
          7E   009C D5   32   0DD4  2629 40$:      CVTWL   @UCB$L_DPC(R5),-(SP)    ;GET BRANCH DISPLACEMENT
               009C C5   8E   C0   0DD9  2630          ADDL    (SP)+,UCB$L_DPC(R5)     ;CALCULATE RETURN ADDRESS - 2
       09 009A C5   0F   E0   0DDE  2631          BBS     #IO$V_INHRETRY,UCB$W_FUNC(R5),60$ ;IF SET, RETRY INHIBITED
               009C C5   02   C0   0DE4  2632 50$:      ADDL    #2,UCB$L_DPC(R5)        ;ADJUST TO CORRECT RETURN ADDRESS
                    009C D5   17   0DE9  2633          JMP     @UCB$L_DPC(R5)          ;RETURN TO DRIVER
                              0DED  2634
                              0DED  2635 ;
                              0DED  2636 ; FATAL CONTROLLER OR DRIVE ERROR
                              0DED  2637 ;       Here we log a device error (unless the only error is that the media
                              0DED  2638 ;               on the device is OFF-LINE) and then we branch to FATALERR.
                              0DED  2639 ;
                              0DED  2640
                              0DED  2641 60$:
            07 52    0C   E0   0DFD  2642          BBS     #MT_DS_V_MOL,R2,63$     ; Branch to log error if media ON-LINE.
          50   4000 8F   B1   0DF1  2643          CMPW    #MT_ER_M_UNS,R0         ; Drive is OFF-LINE. See if ONLY error
                              0DF6  2644                                          ;  is UNSAFE.  If so, this implies
                              0DF6  2645                                          ;  ONLY error is that media is OFF-LINE.
                     06   13   0DF6  2646          BEQL    66$                     ; Branch around logging of error if
                              0DF8  2647                                          ;  ONLY off-line.
                              0DF8  2648 63$:
     00000000'GF      16   0DF8  2649          JSB     G^ERL$DEVICERR          ; Log Device Error.
                              0DFE  2650 66$:
             FL03     31   0DFE  2651          BRW     FATALERR                ;
                              0E01  2652
                              0E01  2653 ;
                              0E01  2654 ; SPECIAL CONDITION ( POWER FAILURE OR DEVICE TIME OUT)
                              0E01  2655 ;
                              0E01  2656
          68 A5    01   AA   0E01  2657 70$:      BICW    #UCB$M_MT_REWIND,UCB$W_DEVSTS(R5) ;CLEAR REWIND IN PROGRESS
          46 A5    10   A8   0E05  2658          BISW    #<MT$M_LOST@-16>,UCB$L_DEVDEPEND+2(R5) ;SET POSITION LOST
       65 64 A5    05   E4   0E09  2659          BBSC    #UCB$V_POWER,UCB$W_STS(R5),100$ ;IF SET, POWER FAILURE
                              0E0E  2660
                              0E0E  2661 ;
                              0E0E  2662 ; DEVICE TIME OUT
                              0E0E  2663 ;
                              0E0E  2664
               03A2   30   0E0E  2665          BSBW    TM_SAVDRVSTS            ; Save drive registers in UCB for ERROR LOGG
     00000000'GF      16   0E11  2666          JSB     G^ERL$DEVICTMO          ;LOG DEVICE TIME OUT
          53   24 A5   D0   0E17  2667          MOVL    UCB$L_CRB(R5),R3        ;GET ADDRESS OF PRIMARY CRB
          53   20 A3   D0   0E1B  2668          MOVL    CRB$L_LINK(R3),R3       ;GET ADDRESS OF SECONDARY CRB
          53   2C A3   D0   0E1F  2669          MOVL    CRB$L_INTD+VEC$L_IDB(R3),R3 ;GET ADDRESS OF SECONDARY IDB
          04 A3    55   D1   0E23  2670          CMPL    R5,IDB$L_OWNER(R3)      ;DEVICE OWN CONTROLLER?
                   22   12   0E27  2671          BNEQ    80$                     ;IF NEQ NO
                              0E29  2672          DSBINT                          ;DISABLE INTERRUPTS
```

```
                    06   D0  0E2F  2673          MOVL    #MBA$M_CR_ABORT!MBA$M_CR_IE,- ;ABORT THE DATA TRANSFER
                 04 A4       0E31  2674                  MBA$L_CR(R4)                  ;
                             0E33  2675          WFIKPCH 75$,#T5                       ;WAIT FOR ABORT AND KEEP CHANNEL
                             0E3D  2676          IOFORK                                ;CREATE FORK PROCESS
                             0E43  2677 75$:                                           ;
              04 A4    01   D0  0E43  2678          MOVL    #MBA$M_CR_INIT,MBA$L_CR(R4) ;INITIALIZE ENTIRE MBA
              04 A4    04   D0  0E47  2679          MOVL    #MBA$M_CR_IE,MBA$L_CR(R4) ;ENABLE MBA INTERRUPTS
                             0E4B  2680 80$:        SETIPL  UCB$B_FIPL(R5)                ;LOWER IPL TO DEVICE FORK LEVEL
                 00B0 C5     D5  0E4F  2681          TSTL    UCB$L_RECORD(R5)              ; Timeouts on READS at BOT are due to
                             0E53  2682                                                    ;  lack of an ID burst that should NOT
                             0E53  2683                                                    ;  cause a CLEAR of UCB$M_VALID.
                       07    12  0E53  2684          BNEQ    90$                           ; If NOT at BOT than branch around.
                       0C    91  0E55  2685          CMPB    #CDF_READDATA,-               ; See if last function was a READ.
                 0092 C5           0E57  2686                  UCB$B_FEX(R5)
                       06    13  0E5A  2687          BEQL    TIMEOUT                       ; If YES, branch around clearing of
                             0E5C  2688                                                    ;  volume valid bit.
           64 A5 0800 8F     AA  0E5C  2689 90$:     BICW    #UCB$M_VALID,UCB$W_STS(R5) ;SET VOLUME SOFTWARE INVALID
                             0E62  2690 TIMEOUT:                                       ;
              50 022C 8F     3C  0E62  2691          MOVZWL  #SS$_TIMEOUT,R0           ;SET FINAL COMPLETION STATUS
                             0E67  2692
                             0E67  2693 ;
                             0E67  2694 ; RESET TRANSFER BYTE COUNT TO ZERO
                             0E67  2695 ;
                             0E67  2696
                             0E67  2697 RESETXFR:                                      ;
                 00BC C5     B4  0E67  2698          CLRW    UCB$W_MT_FC(R5)           ;CLEAR SAVED FRAME COUNT
                       52    D4  0E6B  2699          CLRL    R2                        ;CLEAR DRIVE STATUS
                    FC03     31  0E6D  2700          BRW     FUNCXT                    ;
                             0E70  2701
                             0E70  2702 ;
                             0E70  2703 ; POWER FAILURE
                             0E70  2704 ;
                             0E70  2705
                      01B3   31  0E70  2706 99$.     BRW     200$                      ; Branch around to ABORT powerfail.
           E4 64 A5    0B    E1  0E73  2707 100$:    BBC     #UCB$V_VALID,UCB$W_STS(R5),90$ ;IF CLR, VOLUME SOFTWARE INVALID
                       04    A8  0E78  2708          BISW    #UCB$M_MT_PWRFL,-         ; Set flag that we are in REPOSITIONING
                    68 A5        0E7A  2709                  UCB$W_DEVSTS(R5)          ;  logic.  This is looked at in CANCEL.
                             0E7C  2710 102$:                                          ;
                             0E7C  2711          RELCHAN                           ;RELEASE ALL CHANNELS
                             0E82  2712
                             0E82  2713 ;
                             0E82  2714 ; Try to initiate REWIND of the drive by issuing a hardware rewind command.
                             0E82  2715 ;
                             0E82  2716
                             0E82  2717          REQPCHAN                          ; R4 => TM03 CSR.
                       03    E0  0E88  2718          BBS     #UCB$V_MT_CNCLP,-         ; See if IO$_CANCEL occurred while we
                 E3 68 A5        0E8A  2719                  UCB$W_DEVSTS(R5),99$      ;  were in resource wait. If SO, branch.
                             0E8D  2720
           24 A4  00C6 C5     3C  0E8D  2721          MOVZWL  UCB$W_MT_TC(R5),MT_TC(R4) ;SELECT DRIVE
                    64       09  0E93  2722          MOVZBL  #F_DRVCLR!1,MT_CS1(R4)  ;CLEAR DRIVE
                             0E96  2723          DSBINT
                       05    E4  0E9C  2724          BBSC    #UCB$V_POWER,-            ; If another POWERFAIL, clear bit and
                 1F 64 A5        0E9E  2725                  UCB$W_STS(R5),104$        ;  branch around.
                       07    9A  0EA1  2726          MOVZBL  #F_REWIND!MT_CS1_M_GO,- ; Initiate REWIND command, if the
                       64        0EA3  2727                  MT_CS1(R4)                ;  hardware accepts it.
                             0EA4  2728          WFIKPCH 103$,#2                   ; Wait for interrupt or for short
                             0EAE  2729                                            ;  timeout.  If we timeout before the
```

```
                                0EAE  2730                                    ;  rewind is completed the following
                                0EAE  2731                                    ;  wait loop will take care of this.
                                0EAE  2732
                                0EAE  2733          IOFORK
                                0EB4  2734 103$:    SETIPL  UCB$B_FIPL(R5)    ; Lower to FORK level in case we had
                                0EB8  2735                                    ;  the timeout.
                                0EB8  2736          RELCHAN                   ; Release the channel.
                          05    11  0EBE  2737          BRB     10$           ; Branch around to await arrival at BOT.
                                0EC0  2738
                                0EC0  2739 104$:    ENBINT                    ; If we had another POWERFAIL,
                          AE    11  0EC3  2740          BRB     100$          ;  branch back to try again.
                                0EC5  2741
                                0EC5  2742 ;
                                0EC5  2743 ; WAIT FOR UNIT TO BE REWOUND AND REACH BEGINNING OF TAPE
                                0EC5  2744 ;
                                0EC5  2745
                        0093 C5   94  0EC5  2746 105$:    CLRB    UCB$B_CEX(R5)     ;CLEAR MESSAGE TIME COUNTER
                                0EC9  2747 110$:    DSBINT                    ;DISABLE INTERRUPTS
                                0ECF  2748          WFIKPCH 115$,#2           ;WAITFOR INTERRUPT OR TIME OUT
                                0ED9  2749          IOFORK                    ;CREATE FORK PROCESS
                                0EDF  2750 115$:    SETIPL  UCB$B_FIPL(R5)    ;LOWER IPL TO FORK LEVEL
                                0EE3  2751          REQPCHAN                  ;REQUEST PRIMARY CHANNEL
                          03    E0  0EE9  2752          BBS     #UCB$V_MT_CNCLP,- ; See if IO$_CANCEL occurred while we
                      82 68 A5      0EEB  2753                  UCB$W_DEVSTS(R5),99$ ;  were in resource wait. If SO, branch.
                                0EEE  2754
                24 A4   00C6 C5   3C  0EEE  2755          MOVZWL  UCB$W_MT_TC(R5),MT_TC(R4) ;SELECT DRIVE
                          04 A4   DD  0EF4  2756          PUSHL   MT_DS(R4)         ;SAVE DRIVE STATUS
                                0EF7  2757          RELCHAN                   ;RELEASE ALL CHANNELS
                             52 8ED0  0EFD  2758          POPL    R2                ;RETRIEVE SAVED DRIVE STATUS
                       27 64 A5   05  E4  0F00  2759          BBSC    #UCB$V_POWER,UCB$W_STS(R5),122$ ;IF SET, ANOTHER POWER FAILURE
                          BC 52   0D  E0  0F05  2760          BBS     #MT_DS_V_PIP,R2,105$      ;IF SET, POSITIONING IN PROGRESS
                          04 52   0C  E1  0F09  2761          BBC     #MT_DS_V_MOL,R2,120$      ;IF CLR, MEDIUM OFFLINE
                          1E 52   01  E0  0F0D  2762          BBS     #MT_DS_V_BOT,R2,125$      ;IF SET, AT BEGINNING OF TAPE
            FFB0 0093 C5    01    0F  9D  0F11  2763 120$:    ACBB    #15,#1,UCB$B_CEX(R5),110$ ;TIME TO OUTPUT MESSAGE?
                             54   00'8F  9A  0F19  2764          MOVZBL  #MSG$_DEVOFFLIN,R4        ;SET MESSAGE NUMBER
                          53   00000000'GF  9E  0F1D  2765          MOVAB   G^SYS$GL_OPRMBX,R3        ;GET ADDRESS OF OPERATOR MAILBOX
                               00000000'GF  16  0F24  2766          JSB     G^EXE$SNDEVMSG            ;SEND MESSAGE TO OPERATOR
                                        99  11  0F2A  2767          BRB     105$                      ;
                                0F2C  2768
                                    FF44    31  0F2C  2769 122$:    BRW     100$              ; Branch back after another POWERFAIL.
                                0F2F  2770
                                0F2F  2771 ;
                                0F2F  2772 ; MEDIUM ONLINE AND UNIT POSITIONED TO BEGINNING OF TAPE
                                0F2F  2773 ;
                                0F2F  2774
                                0F2F  2775 125$:
                        00CC C5   D0  0F2F  2776          MOVL    UCB$L_MT_ORGPOS(R5),-     ; Force reposition to ORIGINAL value
                        00B0 C5       0F33  2777                  UCB$L_RECORD(R5)          ;  of UCB$L_RECORD.
                                0F36  2778
                78 A5   00B0 C5   D0  0F36  2779          MOVL    UCB$L_RECORD(R5),UCB$L_SVAPTE(R5) ;SAVE TAPE POSITION
                        00B0 C5   D4  0F3C  2780          CLRL    UCB$L_RECORD(R5)          ;CLEAR CURRENT POSITION
                                0F40  2781          REQPCHAN                  ;REQUEST PRIMARY CHANNEL
                          03    E0  0F46  2782          BBS     #UCB$V_MT_CNCLP,- ; See if IO$_CANCEL occurred while we
                      0E 68 A5      0F48  2783                  UCB$W_DEVSTS(R5),133$ ;  were in resource wait. If SO, branch.
                                0F4B  2784
                          53   54   D0  0F4B  2785          MOVL    R4,R3             ;SET ADDRESS OF DRIVE CONTROL REGISTER
                                0F4E  2786 130$:    REQSCHAN                  ;REQUEST SECONDARY CHANNEL
```

```
                                J 1
                03   E1  0F54  2787          BBC     #UCB$V_MT_CNCLP,-       ; See if IO$_CANCEL occurred while we
           03 68 A5         0F56  2788                  UCB$W_DEVSTS(R5),136$   ;  were in resource wait. If NOT, branch.
              00CA  31      0F59  2789 133$:    BRW     200$                    ; If SO, branch around to abort operation.
                            0F5C  2790 136$:
        24 A3  00C6 C5  3C  0F5C  2791          MOVZWL  UCB$W_MT_TC(R5),MT_TC(R3) ;SET TAPE CONTROL PARAMETERS
               63   09  9A  0F62  2792          MOVZBL  #F_DRVCLR!1,MT_CS1(R3)  ;CLEAR DRIVE
        78 A5  00B0 C5  D1  0F65  2793          CMPL    UCB$L_RECORD(R5),UCB$L_SVAPTE(R5) ;TAPE CORRECTLY POSITIONED?
                 03     12  0F6B  2794          BNEQ    138$                    ; NEQ implies NO, not yet done.
                0085    31  0F6D  2795          BRW     180$                    ; Else branch to end.
                            0F70  2796 138$:
                0267    30  0F70  2797          BSBW    TM_SETINTSPC            ;SETUP MBA FOR INTERNAL SPACE FUNCTION
                            0F73  2798          DSBINT                          ;DISABLE INTERRUPTS
          6B 64 A5   05     E0  0F79  2799          BBS     #UCB$V_POWER,UCB$W_STS(R5),160$ ;IF SET, ANOTHER POWER FAILURE
               63   29      9A  0F7E  2800          MOVZBL  #F_WRITECHECK!1,MT_CS1(R3) ;EXECUTE FUNCTION
                            0F81  2801          WFIKPCH 140$,#12                ;WAIT FOR FUNCTION TO COMPLETE
        00B4 C5    08 A4    D0  0F8B  2802          MOVL    MBA$L_SR(R4),UCB$L_MT_SR(R5) ;SAVE FINAL MBA STATUS
                 021F   30      0F91  2803          BSBW    TM_SAVDRVSTS            ;SAVE DRIVE STATUS
                            0F94  2804          IOFORK                          ;CREATE FORK PROCESS
                            0F9A  2805 140$:    SETIPL  UCB$B_FIPL(R5)          ;LOWER IPL TO FORK LEVEL
          49 64 A5   05     E0  0F9E  2806          BBS     #UCB$V_POWER,UCB$W_STS(R5),170$ ;IF SET, ANOTHER POWER FAILURE
          3E 64 A5   06     E0  0FA3  2807          BBS     #UCB$V_TIMOUT,UCB$W_STS(R5),155$ ;IF SET, DEVICE TIME OUT
                            0FA8  2808          RELSCHAN                        ;RELEASE SECONDARY CHANNEL
     50   00BA C5   F142 CF  AB  0FAE  2809          BICW3   XTAB+<CDF_INTSPCFOR*2>,UCB$W_MT_ER(R5),R0 ;GET DRIVE ERROR STATUS
          51   00B4 C5    D0  0FB6  2810          MOVL    UCB$L_MT_SR(R5),R1      ;GET FINAL MBA STATUS
          52   00B8 C5    3C  0FBB  2811          MOVZWL  UCB$W_MT_DS(R5),R2      ;GET FINAL DRIVE STATUS
           51    00013F80 8F  CA  0FC0  2812          BICL    #MBA$M_SR_DLT!-         ;CLEAR DATA LATE AND,
                            0FC7  2813                  MBA$M_SR_ATTN!-         ;ATTENTION AND,
                            0FC7  2814                  MBA$M_SR_DTABT!-        ;DATA TRANSFER ABORT AND,
                            0FC7  2815                  MBA$M_SR_DTCOMP!-       ;DATA TRANSFER COMPLETE AND,
                            0FC7  2816                  MBA$M_SR_MBEXC!-        ;MASSBUS EXCEPTION AND,
                            0FC7  2817                  MBA$M_SR_MXF!-          ;MISSED TRANSFER AND,
                            0FC7  2818                  MBA$M_SR_WCKLWR!-       ;WRITE CHECK LOWER AND,
                            0FC7  2819                  MBA$M_SR_WCKUPR,R1      ;WRITE CHECK UPPER
                   4C   12  0FC7  2820          BNEQ    190$                    ;IF NEQ FATAL SPACING ERROR
                   50   B5  0FC9  2821          TSTW    R0                      ;ANY FATAL DRIVE ERRORS?
                   48   12  0FCB  2822          BNEQ    190$                    ;IF NEQ YES
              0E 52     02  E0  0FCD  2823          BBS     #MT_DS_V_TM,R2,150$     ;IF SET, TAPE MARK ENCOUNTERED
              0A 52     05  E0  0FD1  2824          BBS     #MT_DS_V_PES,R2,150$    ; If at 1600 BPI, branch around test
                            0FD5  2825                                          ;  for minimum length record.
           00BC C5     0E  B1  0FD5  2826          CMPW    #MIN_RECORD,UCB$W_MT_FC(R5) ;MINIMUM RECORD READ?
                   03   1B  0FDA  2827          BLEQU   150$                    ; LEQ implies yes, minimum record read.
                FF6F    31  0FDC  2828          BRW     130$                    ; Else branch back to read next.
              00B0 C5   D6  0FDF  2829 150$:   INCL    UCB$L_RECORD(R5)        ;UPDATE CURRENT TAPE POSITION
                 FF68   31  0FE3  2830          BRW     130$                    ;
                 FE73   31  0FE6  2831 155$:   BRW     90$                     ;
                            0FE9  2832
                            0FE9  2833 ;
                            0FE9  2834 ; POWER FAILURE DURING REPOSITIONING OF TAPE
                            0FE9  2835 ;
                            0FE9  2836
                            0FE9  2837 160$:   ENBINT                          ;ENABLE INTERRUPTS
     00B0 C5    78 A5    D0  0FEC  2838 170$:   MOVL    UCB$L_SVAPTE(R5),UCB$L_RECORD(R5) ;RESTORE TAPE POSITION
                 FE87   31  0FF2  2839          BRW     102$                    ;
                            0FF5  2840
                            0FF5  2841 ;
                            0FF5  2842 ; TAPE SUCCESSFULLY REPOSITIONED - RESTART FUNCTION
                            0FF5  2843 ;
```

```
                     0FF5  2844
                     0FF5  2845 180$:   RELCHAN                                 ;RELEASE ALL CHANNELS
          04   AA    0FFB  2846         BICW    #UCB$M_MT_PWRFL,-               ; Clear flag signalling that we are
        68 A5         0FFD  2847                 UCB$W_DEVSTS(R5)                ;  repositioning.
     46 A5    10   AA  0FFF  2848         BICW    #<MT$M_LOSTA-16>,UCB$L_DEVDEPEND+2(R5) ;CLEAR LOST POSITION
64 A5   0800 8F   A8  1003  2849         BISW    #UCB$M_VALID,UCB$W_STS(R5) ;SET VOLUME VALID
     53   58 A5   D0  1009  2850         MOVL    UCB$L_IRP(R5),R3                ;RETRIEVE ADDRESS OF I/O PACKET
  78 A5   2C A3   7D  100D  2851         MOVQ    IRP$L_SVAPTE(R3),UCB$L_SVAPTE(R5) ;RESTORE TRANSFER PARAMETERS
        F1E6      31  1012  2852         BRW     TM_STARTIO                      ;
                      1015  2853
                      1015  2854 ;
                      1015  2855 ; FATAL SPACING ERROR DURING REPOSITIONING OF TAPE
                      1015  2856 ;
                      1015  2857
64 A5   0800 8F   AA  1015  2858 190$:   BICW    #UCB$M_VALID,UCB$W_STS(R5) ;SET VOLUME SOFTWARE INVALID
          04   AA    101B  2859         BICW    #UCB$M_MT_PWRFL,-               ; Clear flag signalling that we are
        68 A5         101D  2860                 UCB$W_DEVSTS(R5)                ;  repositioning.
        00BC C5   B4  101F  2861         CLRW    UCB$W_MT_FC(R5)                 ;CLEAR SAVED FRAME COUNT
           F9DE   31  1023  2862         BRW     FATALERR                        ;
                      1026  2863 200$:
          0C   AA    1026  2864         BICW    #UCB$M_MT_PWRFL!UCB$M_MT_CNCLP,-
        68 A5         1028  2865                 UCB$W_DEVSTS(R5)                ; Clear flag signalling that we are
                      102A  2866                                                 ;  repositioning AND that CANCEL was
                      102A  2867                                                 ;  signalled.
        0800 8F   AA  102A  2868         BICW    #UCB$M_VALID,-
          64 A5       102E  2869                 UCB$W_STS(R5)                   ; Set volume software invalid.
        00BC C5   B4  1030  2870         CLRW    UCB$W_MT_FC(R5)                 ; Clear saved frame count
        50   2C   D0  1034  2871         MOVL    #SS$_ABORT,R0                   ; Set abort status.
             52   D4  1037  2872         CLRL    R2                              ; Passed to FUNCXT.
           FA37   31  1039  2873         BRW     FUNCXT                          ;
                      103C  2874         .DSABL  LSB
```

```
                     103C  2876          .SBTTL  TE16/TU77 CLASSIFY DRIVE TYPE AND SET PARAMETERS
                     103C  2877 ;
                     103C  2878 ; TM_DTYPE - TE16/TU77 CLASSIFY DRIVE TYPE AND SET PARAMETERS
                     103C  2879 ;
                     103C  2880 ; THIS ROUTINE IS CALLED WHEN AN UNSOLICITED INTERRUPT OCCURS ON A DRIVE, DURING
                     103C  2881 ; SYSTEM INITIALIZATION, AND AT POWER RECOVERY TO DETERMINE THE DRIVE TYPE,
                     103C  2882 ; media id type, and  SET UNIT PARAMETERS.
                     103C  2883 ;
                     103C  2884 ; INPUTS:
                     103C  2885 ;
                     103C  2886 ;       R3 = ADDRESS OF TM03 DRIVE CONTROL REGISTER.
                     103C  2887 ;       R5 = DEVICE UNIT UCB ADDRESS.
                     103C  2888 ;
                     103C  2889 ; OUTPUTS:
                     103C  2890 ;
                     103C  2891 ;       THE DRIVE STATUS REGISTER IS INTERROGATED AND UNIT PARAMETERS ARE SET.
                     103C  2892 ;
                     103C  2893
                     103C  2894 TM_DTYPE:                                ;CLASSIFY DRIVE TYPE AND SET PARAMETERS
          18 A3   DD 103C  2895          PUSHL   MT_DT(R3)               ;READ DRIVE TYPE REGISTER
  6E  FE00 8F   AA 103F  2896          BICW    #^C<^X1FF>,(SP)         ;CLEAR EXTRANEOUS BITS
  52  F000 CF   9E 1044  2897          MOVAB   TM_DTDESC,R2            ;GET ADDRESS OF DESCRIPTOR TABLE
      82   6E   B1 1049  2898 10$:     CMPW    (SP),(R2)+              ;DRIVE TYPE MATCH?
           0E   13 104C  2899          BEQL    20$                     ;IF EQL YES
      52   01   C0 104E  2900          ADDL    #TM_DTDESCLEN-2,R2      ;ADVANCE TO NEXT ENTRY
           62   B5 1051  2901          TSTW    (R2)                    ;END OF TABLE?
           F4   12 1053  2902          BNEQ    10$                     ;IF NEQ NO
  64 A5    10   AA 1055  2903          BICW    #UCB$M_ONLINE,UCB$W_STS(R5) ;SET UNIT OFFLINE
      52   01   C2 1059  2904          SUBL    #TM_DTDESCLEN-2,R2      ;BACK UP TO LAST DRIVE DESCRIPTOR
  41 A5    62   90 105C  2905 20$:     MOVB    (R2),UCB$B_DEVTYPE(R5)  ;SET DRIVE TYPE
                     1060  2906
                     1060  2907 ; Determine the media id type usingn the device type found above.
                     1060  2908          ASSUME  DT$_TE16 EQ 1
                     1060  2909          ASSUME  DT$_TU45 EQ 2
                     1060  2910          ASSUME  DT$_TU77 EQ 3
       52   41 A5   9A 1060  2911          MOVZBL  UCB$B_DEVTYPE(R5),R2    ; Get type
          008C C5   D4 1064  2912          CLRL    UCB$L_MEDIA_ID(R5)      ; Assume no media id
                     1068  2913          CASE    R2,<-
                     1068  2914                  60$,-
                     1068  2915                  30$,-                   ; TE16
                     1068  2916                  40$,-                   ; TU45
                     1068  2917                  50$,-                   ; TU77
                     1068  2918                  >
           1F   11 1074  2919          BRB     60$
008C C5   6D285010 8F   D0 1076  2920 30$:     MOVL    #MEDIA_ID_TE16,UCB$L_MEDIA_ID(R5)
           14   11 107F  2921          BRB     60$
008C C5   6D29502D 8F   D0 1081  2922 40$:     MOVL    #MEDIA_ID_TU45,UCB$L_MEDIA_ID(R5)
           09   11 108A  2923          BRB     60$
008C C5   6D29504D 8F   D0 108C  2924 50$:     MOVL    #MEDIA_ID_TU77,UCB$L_MEDIA_ID(R5)
           8E   D5 1095  2925 60$:     TSTL    (SP)+                   ;REMOVE DRIVE TYPE FROM STACK
           05   1097  2926          RSB                             ;
```

```
                                1098  2928          .SBTTL  TM03-TE16/TU77 REGISTER DUMP ROUTINE
                                1098  2929 ;
                                1098  2930 ; TM_REGDUMP - TM03-TE16/TU77 REGISTER DUMP ROUTINE
                                1098  2931 ;
                                1098  2932 ; THIS ROUTINE IS CALLED TO SAVE THE CONTROLLER AND DRIVE REGISTERS IN A
                                1098  2933 ; SPECIFIED BUFFER. IT IS CALLED FROM THE DEVICE ERROR LOGGING ROUTINE AND
                                1098  2934 ; FROM THE DIAGNOSTIC BUFFER FILL ROUTINE.
                                1098  2935 ;
                                1098  2936 ; INPUTS:
                                1098  2937 ;
                                1098  2938 ;       R0 = ADDRESS OF REGISTER SAVE BUFFER.
                                1098  2939 ;       R4 = ADDRESS OF ADAPTER CONFIGURATION REGISTER.
                                1098  2940 ;       R5 = DEVICE UNIT UCB ADDRESS.
                                1098  2941 ;
                                1098  2942 ; OUTPUTS:
                                1098  2943 ;
                                1098  2944 ;       THE CONTROLLER AND DRIVE REGISTERS ARE SAVED IN THE SPECIFIED BUFFER.
                                1098  2945 ;
                                1098  2946
                                1098  2947 TM_REGDUMP:                                       ;TM03-TE16/TU77 REGISTER DUMP ROUTINE
                  80    11  D0  1098  2948          MOVL    #<MT_TC+4+MBA$L_BCR+4+8>/4,(R0)+ ;INSERT NUMBER OF DEVICE REGISTERS
                  80    64  D0  109B  2949          MOVL    MBA$L_CSR(R4),(R0)+      ;SAVE CONFIGURATION REGISTER
              80    04  A4  D0  109E  2950          MOVL    MBA$L_CR(R4),(R0)+       ;SAVE CONTROL REGISTER
            80    00B4  C5  D0  10A2  2951          MOVL    UCB$L_MT_SR(R5),(R0)+    ;SAVE STATUS REGISTER
              80    0C  A4  D0  10A7  2952          MOVL    MBA$L_VAR(R4),(R0)+      ;SAVE VIRTUAL ADDRESS REGISTER
              80    10  A4  D0  10AB  2953          MOVL    MBA$L_BCR(R4),(R0)+      ;SAVE BYTE COUNT REGISTER
      51    F8  A0    08  09  EF  10AF  2954          EXTZV   #9,#8,-8(R0),R1          ;GET FINAL MAP REGISTER NUMBER
        80    0800  C441  D0  10B5  2955          MOVL    MBA$L_MAP(R4)[R1],(R0)+  ;SAVE FINAL MAP REGISTER CONTENTS
                    80  D4  10BB  2956          CLRL    (R0)+                    ;ASSUME NO PREVIOUS MAP REGISTER
                    51  D7  10BD  2957          DECL    R1                       ;CALCULATE PREVIOUS MAP REGISTER NUMBER
                    07  19  10BF  2958          BLSS    10$                      ;IF LSS NONE
    FC  A0    0800  C441  D0  10C1  2959          MOVL    MBA$L_MAP(R4)[R1],-4(R0) ;SAVE PREVIOUS MAP REGISTER CONTENTS
              51    06  9A  10C8  2960 10$:       MOVZBL  #<MT_TC>/4-3,R1          ;SET NUMBER OF DRIVE REGISTERS TO SAVE
          52    24  A5  D0  10CB  2961          MOVL    UCB$L_CRB(R5),R2         ;GET ADDRESS OF PRIMARY CRB
    52    2C  B2    0C  C1  10CF  2962          ADDL3   #3*4,@CRB$L_INTD+VEC$L_IDB(R2),R2 ;GET ADDRESS OF TM03-TE16/TU77 REG
      80    00C0  C5  3C  10D4  2963          MOVZWL  UCB$W_MT_CST(R5),(R0)+   ;SAVE DRIVE CONTROL REGISTER
      80    00B8  C5  3C  10D9  2964          MOVZWL  UCB$W_MT_DS(R5),(R0)+    ;SAVE DRIVE STATUS REGISTER
      80    00BA  C5  3C  10DE  2965          MOVZWL  UCB$W_MT_ER(R5),(R0)+    ;SAVE DRIVE ERROR REGISTER
              80    82  D0  10E3  2966 20$:       MOVL    (R2)+,(R0)+              ;SAVE DRIVE REGISTER
                FA  51  F5  10E6  2967          SOBGTR  R1,20$                   ;ANY MORE TO SAVE?
              00D0  C5  3C  10E9  2968          MOVZWL  UCB$W_MT_CC_SAV(R5),-    ; Move previously saved register
                F8  A0        10ED  2969                  MT_CC-MT_TC(R0)          ;  contents back to where it should go.
      80    00C2  C5  3C  10EF  2970          MOVZWL  UCB$W_MT_TC_SAV(R5),(R0)+ ;SAVE TAPE CONTROL REGISTER
                    05  10F4  2971          RSB                              ;
```

```
                                10F5  2973          .SBTTL  TM03-TE16/TU77 TAPE DRIVE INITIALIZATION
                                10F5  2974 ;
                                10F5  2975 ; TM_TXXX_INIT - TM03-TE16/TU77 TAPE DRIVE INITIALIZATION
                                10F5  2976 ;
                                10F5  2977 ; THIS ROUTINE IS CALLED AT SYSTEM INITIALIZATION AND AT POWER RECOVERY TO SET
                                10F5  2978 ; DRIVE PARAMETERS.
                                10F5  2979 ;
                                10F5  2980 ; INPUTS:
                                10F5  2981 ;
                                10F5  2982 ;       R3 = ADDRESS OF TM03 DRIVE CONTROL REGISTER.
                                10F5  2983 ;       R4 = ADDRESS OF MBA CONFIGURATION STATUS REGISTER.
                                10F5  2984 ;       R5 = DEVICE UNIT UCB ADDRESS.
                                10F5  2985 ;
                                10F5  2986 ; OUTPUTS:
                                10F5  2987 ;
                                10F5  2988 ;       UNIT PARAMETERS ARE ESTABLISHED.
                                10F5  2989 ;
                                10F5  2990
                                10F5  2991 TM_TXXX_INIT:                            ;TE16/TU77 TAPE DRIVE INITIALIZATION
            52   53   54   C3  10F5  2992          SUBL3   R4,R3,R2                ;CALCULATE OFFSET TO DRIVE CONTROL REGISTER
            52   FC00 C2   9E  10F9  2993          MOVAB   -MBA$L_ERB(R2),R2       ;SUBTRACT OUT EXTERNAL REGISTER BASE
     0090 C5    52   0080 8F   A7  10FE  2994          DIVW3   #128.,R2,UCB$B_SLAVE(R5) ;SET ADAPTER DRIVE NUMBER
     0091 C5    0090 C5    20   85  1106  2995          MULB3   #128./4,UCB$B_SLAVE(R5),UCB$B_SLAVE+1(R5) ;SET DRIVE OFFSET CONSTANT
00C6 C5    03    00   54   A5   F0  110E  2996          INSV    UCB$W_UNIT(R5),#0,#3,UCB$W_MT_TC(R5) ;INSERT UNIT NUMBER
                      63   09   9A  1116  2997          MOVZBL  #<MT_DRVCLR!1>,MT_CS1(R3) ;CLEAR DRIVE
                   7E   64   A5   3C  1119  2998          MOVZWL  UCB$W_STS(R5),-(SP)     ;SAVE CURRENT UNIT STATUS
                        08   A4   DD  111D  2999          PUSHL   MBA$L_SR(R4)            ;READ MBA STATUS REGISTER
                68   A5   02   AA  1120  3000          BICW    #UCB$M_MT_PRVMOL,UCB$W_DEVSTS(R5) ;CLEAR PREVIOUS MOL STATE
             64   A5   0810 8F   AA  1124  3001          BICW    #UCB$M_ONLINE!UCB$M_VALID,UCB$W_STS(R5) ;SET UNIT OFFLINE/INVALID
                3B   6E   12   E0  112A  3002          BBS     #MBA$V_SR_NED,(SP),30$  ;IF SET, NONEXISTENT TM03
             24   A3   00C6 C5   3C  112E  3003          MOVZWL  UCB$W_MT_TC(R5),MT_TC(R3) ;SELECT SLAVE DRIVE
 52    18   A3   FFFF01FF 8F   CB  1134  3004          BICL3   #^C<^XFE00>,MT_DT(R3),R2 ;ISOLATE HIGH PART OF DRIVE TYPE
            52   C400 8F   B1  113D  3005          CMPW    #^XC400,R2              ;TAPE DRIVE AND SLAVE PRESENT?
                      22   12  1142  3006          BNEQ    20$                     ;IF NEQ NO
                68   A5   10   A8  1144  3007          BISW    #UCB$M_ONLINE,UCB$W_STS(R5) ;SET UNIT ONLINE
                     FEF1 30   1148  3008          BSBW    TM_DTYPE                ;CLASSIFY DRIVE TYPE
             16 64   A5   04   E1  114B  3009          BBC     #UCB$V_ONLINE,UCB$W_STS(R5),20$ ;IF CLR, UNKNOWN DRIVE TYPE
          52   04   A3   13   78  1150  3010          ASHL    #31-MT_DS_V_MOL,MT_DS(R3),R2 ;MEDIUM CURRENTLY ONLINE?
                      04   18  1155  3011          BGEQ    15$                     ;IF GEQ NO
                68   A5   02   A8  1157  3012          BISW    #UCB$M_MT_PRVMOL,UCB$W_DEVSTS(R5) ;SET PREVIOUS MOL STATE
             06 04   AE   0B   E1  115B  3013 15$:     BBC     #UCB$V_VALID,4(SP),20$  ;IF CLR, VOLUME SOFTWARE INVALID
          64   A5   0800 8F   A8  1160  3014          BISW    #UCB$M_VALID,UCB$W_STS(R5) ;SET VOLUME SOFTWARE VALID
                      63   09   9A  1166  3015 20$:     MOVZBL  #<MT_DRVCLR!1>,MT_CS1(R3) ;CLEAR DRIVE
          08   A4   8E   8E   C9  1169  3016 30$:     BISL3   (SP)+,(SP)+,MBA$L_SR(R4) ;CLEAR MBA STATUS
                           05   116E  3017          RSB                             ;
```

```
                                 116F  3019          .SBTTL  TM03-TE16/TU77 UNSOLICITED INTERRUPT PROCESSING
                                 116F  3020 ;
                                 116F  3021 ; TM_UNSOLNT - TM03-TE16/TU77 UNSOLICITED INTERRUPT PROCESSING
                                 116F  3022 ;
                                 116F  3023 ; THIS ROUTINE IS CALLED WHEN AN UNSOLICITED INTERRUPT IS RECEIVED FOR A SLAVE
                                 116F  3024 ; DRIVE.
                                 116F  3025 ;
                                 116F  3026 ; INPUTS:
                                 116F  3027 ;
                                 116F  3028 ;       R4 = ADDRESS OF TM03 DRIVE CONTROL REGISTER.
                                 116F  3029 ;       R5 = DEVICE UNIT UCB ADDRESS.
                                 116F  3030 ;
                                 116F  3031 ; OUTPUTS:
                                 116F  3032 ;
                                 116F  3033 ;       THE UNIT STATUS AND DRIVE TYPE IS ESTABLISHED.
                                 116F  3034 ;
                                 116F  3035
                                 116F  3036 TM_UNSOLNT:                                  ;UNSOLICITED INTERRUPT PROCESSING
          64 A5    10    AA     116F  3037          BICW    #UCB$M_ONLINE,UCB$W_STS(R5) ;SET UNIT OFFLINE
52  18 A4   FFFF01FF 8F    CB   1173  3038          BICL3   #^C<^XFE00>,MT_DT(R4),R2 ;ISOLATE HIGH PART OF DRIVE TYPE
        52    C400 8F    B1     117C  3039          CMPW    #^XC400,R2              ;TAPE DRIVE AND SLAVE PRESENT?
                26    12     1181  3040          BNEQ    10$                     ;IF NEQ NO
          64 A5    10    A8     1183  3041          BISW    #UCB$M_ONLINE,UCB$W_STS(R5) ;SET UNIT ONLINE
             53    54    D0     1187  3042          MOVL    R4,R3                   ;SET ADDRESS OF TM03 DRIVE CONTROL REGISTER
                FEAF    30     118A  3043          BSBW    TM_DTYPE                ;CLASSIFY DRIVE TYPE
       17 64 A5    04    E1     118D  3044          BBC     #UCB$V_ONLINE,UCB$W_STS(R5),10$ ;IF CLR, UNKNOWN DRIVE TYPE
       0D 38 A5    13    E1     1192  3045          BBC     #DEV$V_MNT,UCB$L_DEVCHAR(R5),5$ ;BRANCH IF NOT MOUNTED
       08 38 A5    18    E1     1197  3046          BBC     #DEV$V_FOR,UCB$L_DEVCHAR(R5),5$ ;BRANCH IF NOT FOREIGN
                                 119C  3047                                          ;TAPE IS MOUNTED FOREIGN, SO...
     64 A5    0800 8F    AA     119C  3048          BISW    #UCB$M_VALID,UCB$W_STS(R5) ;SET VOLUME VALID
                08    11     11A2  3049          BRB     20$
  06 64 A5    05    E0     11A4  3050 5$:       BBS     #UCB$V_POWER,UCB$W_STS(R5),20$ ;IF SET, POWER FAILURE
64 A5    0800 8F    AA     11A9  3051 10$:      BICW    #UCB$M_VALID,UCB$W_STS(R5) ;CLEAR VOLUME SOFTWARE VALID
        64    09    9A     11AF  3052 20$:      MOVZBL  #F_DRVCLR!1,MT_CS1(R4)  ;CLEAR DRIVE
                05    11     11B2  3053          RSB                             ;
```

```
                     11B3  3055          .SBTTL  TM03-TE16/TU77 DRIVE STATUS SAVE ROUTINE
                     11B3  3056 ;
                     11B3  3057 ; TM_SAVDRVSTS - TM03-TE16/TU77 DRIVE STATUS SAVE ROUTINE
                     11B3  3058 ;
                     11B3  3059 ; THIS ROUTINE IS CALLED FROM DEVICE INTERRUPT LEVEL TO SAVE FINAL DRIVE REGISTERS
                     11B3  3060 ; AND THEN CLEAR THE DRIVE.
                     11B3  3061 ;
                     11B3  3062 ; INPUTS:
                     11B3  3063 ;
                     11B3  3064 ;       R3 = ADDRESS OF DRIVE CONTROL STATUS REGISTER 1.
                     11B3  3065 ;       R4 = ADDRESS OF MBA CONFIGURATION STATUS REGISTER.
                     11B3  3066 ;       R5 = DEVICE UNIT UCB ADDRESS.
                     11B3  3067 ;
                     11B3  3068 ; OUTPUTS:
                     11B3  3069 ;
                     11B3  3070 ;       THE FINAL DRIVE REGISTERS ARE SAVED AND THE DRIVE IS CLEARED.
                     11B3  3071 ;
                     11B3  3072
                     11B3  3073 TM_SAVDRVSTS:                                ;SAVE DRIVE STATUS
00BC C5   14 A3  F7  11B3  3074         CVTLW   MT_FC(R3),UCB$W_MT_FC(R5) ;SAVE FRAME COUNT REGISTER
00BA C5   08 A3  F7  11B9  3075         CVTLW   MT_ER(R3),UCB$W_MT_ER(R5) ;SAVE ERROR STATUS REGISTER
00B8 C5   04 A3  F7  11BF  3076         CVTLW   MT_DS(R3),UCB$W_MT_DS(R5) ;SAVE DRIVE STATUS REGISTER
00C2 C5   24 A3  F7  11C5  3077         CVTLW   MT_TC(R3),UCB$W_MT_TC_SAV(R5) ;SAVE TAPE CONTROL REGISTER
   00C0 C5   63  F7  11CB  3078         CVTLW   MT_CS1(R3),UCB$W_MT_CS1(R5) ;SAVE DRIVE CONTROL REGISTER
00D0 C5   1C A3  F7  11D0  3079         CVTLW   MT_CC(R3),UCB$W_MT_CC_SAV(R5)   ; Save Check Character Register.
         63   09  9A  11D6  3080         MOVZBL  #F_DRVCLR!1,MT_CS1(R3)  ;CLEAR DRIVE
                  05  11D9  3081         RSB                                 ;
```

```
                                11DA  3083          .SBTTL  TM03-TE16/TU77 SETUP MBA FOR INTERNAL SPACING FUNCTION
                                11DA  3084 ;
                                11DA  3085 ; TM_SETINTSPC - TM03-TE16/TU77 SETUP MBA FOR INTERNAL SPACING FUNCTION
                                11DA  3086 ;
                                11DA  3087 ; THIS ROUTINE IS CALLED TO SETUP THE MBA REGISTERS FOR AN INTERNAL SPACING
                                11DA  3088 ; FUNCTION THAT USES THE INTERRUPT STACK AS A WRITECHECK BUFFER.
                                11DA  3089 ;
                                11DA  3090 ; INPUTS:
                                11DA  3091 ;
                                11DA  3092 ;       R3 = ADDRESS OF DRIVE CONTROL STATUS REGISTER 1.
                                11DA  3093 ;       R4 = ADDRESS OF MBA CONFIGURATION STATUS REGISTER.
                                11DA  3094 ;       R5 = DEVICE UNIT UCB ADDRESS.
                                11DA  3095 ;
                                11DA  3096 ; OUTPUTS:
                                11DA  3097 ;
                                11DA  3098 ;       THE MBA MAP REGISTERS, BYTE COUNT REGISTER, AND VIRTUAL ADDRESS REGISTER
                                11DA  3099 ;       IS LOADED WITH VALUES THAT MAP ONE PAGE OF THE INTERRUPT STACK.
                                11DA  3100 ;
                                11DA  3101
                                11DA  3102 TM_SETINTSPC:                                ;SETUP MBA FOR INTERNAL SPACING FUNCTION
          7E    78 A5    7D    11DA  3103          MOVQ    UCB$L_SVAPTE(R5),-(SP)    ;SAVE CURRENT TRANSFER PARAMETERS
                7C A5    B4    11DE  3104          CLRW    UCB$W_BOFF(R5)            ;CLEAR BYTE OFFSET IN PAGE
     7E A5    0200 8F    B0    11E1  3105          MOVW    #512,UCB$W_BCNT(R5)       ;SET TRANSFER BYTE COUNT
      50   00000000'GF    D0    11E7  3106          MOVL    G^MMG$GL_SPTBASE,R0       ;GET BASE ADDRESS OF SYSTEM PAGE TABLE
          51    EE0E CF    9E    11EE  3107          MOVAB   TM$DDT,R1                 ;GET STARTING ADDRESS OF DRIVER
   51     51    15    09    EF    11F3  3108          EXTZV   S^#VA$V_VPN,S^#VA$S_VPN,R1,R1 ;EXTRACT SYSTEM VIRTUAL PAGE NUMBER
          78 A5    6041    DE    11F8  3109          MOVAL   (R0)[R1],UCB$L_SVAPTE(R5) ;SET STARTING PTE ADDRESS
            08 A4    00    D2    11FD  3110          MCOML   #0,MBA$L_SR(R4)           ;CLEAR MBA STATUS REGISTER
                             1201  3111          LOADMBA                           ;LOAD MBA MAPPING REGISTERS
          78 A5    8E    7D    1207  3112          MOVQ    (SP)+,UCB$L_SVAPTE(R5)    ;RESTORE CURRENT TRANSFER PARAMETERS
                         05  120B  3113          RSB                               ;
```

```
                     120C  3115          .SBTTL  TM03-TE16/TU77 SLAVE CONTROLLER INTERRUPT DISPATCHER
                     120C  3116 ;+
                     120C  3117 ; TM$INT - TM03-TE16/TU77 SLAVE CONTROLLER INTERRUPT DISPATCHER
                     120C  3118 ;
                     120C  3119 ; THIS ROUTINE IS ENTERED VIA A JSB INSTRUCTION WHEN AN INTERRUPT OCCURS ON A
                     120C  3120 ; TM03-TE16/TU77 SLAVE CONTROLLER. THE STATE OF THE STACK ON ENTRY IS:
                     120C  3121 ;
                     120C  3122 ;         00(SP) = ADDRESS OF IDB ADDRESS.
                     120C  3123 ;         04(SP) = SAVED R2.
                     120C  3124 ;         08(SP) = SAVED R3.
                     120C  3125 ;         12(SP) = SAVED R4.
                     120C  3126 ;         16(SP) = SAVED R5.
                     120C  3127 ;         20(SP) = INTERRUPT PC.
                     120C  3128 ;         24(SP) = INTERRUPT PSL.
                     120C  3129 ;
                     120C  3130 ; INTERRUPT DISPATCHING OCCURS AS FOLLOWS:
                     120C  3131 ;
                     120C  3132 ;         IF THE INTERRUPTING CONTROLLER IS CURRENTLY OWNED AND THE OWNER UNIT IS
                     120C  3133 ;         EXPECTING AN INTERRUPT, THEN THAT UNIT IS DISPATCHED FIRST. ALL OTHER
                     120C  3134 ;         UNITS ARE DISPATCHED BY SELECTING THE CORRESPONDING SLAVE DRIVE, READING
                     120C  3135 ;         ITS STATUS, AND DISPATCHING IF AN ATTENTION CONDITION EXISTS. AS EACH UNIT
                     120C  3136 ;         IS FOUND, A TEST IS MADE TO DETERMINE IF AN INTERRUPT IS EXPECTED ON THE
                     120C  3137 ;         UNIT. IF YES, THEN THE DRIVER IS CALLED AT ITS INTERRUPT RETURN ADDRESS.
                     120C  3138 ;         ELSE THE DRIVER IS CALLED AT ITS UNSOLICITED INTERRUPT ADDRESS. AS EACH
                     120C  3139 ;         CALL TO THE DRIVER RETURNS, THE NEXT SLAVE UNIT IS SELECTED AND AN ATTEMPT
                     120C  3140 ;         IS MADE TO DISPATCH THAT UNIT. WHEN ALL UNITS HAVE BEEN SELECTED AND NO
                     120C  3141 ;         ATTENTION CONDITIONS REMAIN, THE INTERRUPT IS DISMISSED.
                     120C  3142 ;-
                     120C  3143
                     120C  3144 TM$INT::                                     ;TM03-TE16/TU77 SLAVE CONTROLLER INTERRUPT D
          53   00 BE    D0  120C  3145          MOVL    @(SP),R3               ;GET ADDRESS OF IDB
             54   63    D0  1210  3146          MOVL    IDB$L_CSR(R3),R4       ;GET ADDRESS OF TM03-TE16/TU77 REGISTERS
                          1213  3147 ; Here we clear the attention summary bit corresponding to the MASSBUS port
                          1213  3148 ;       number of this TM03.  To do this we obtain a pointer to the MBA-CSR
                          1213  3149 ;       indirectly thru the IDB=>ADP.  With the TM03-CSR address in hand,
                          1213  3150 ;       we subtract to determine the distance of the TM03-CSR to the base
                          1213  3151 ;       of the MBA external registers.  This difference divided by 128
                          1213  3152 ;       (or right shifted by 7) gives the port number of interest.
                          1213  3153 ;       Writing a 1 into the corresponding bit position of the attention
                          1213  3154 ;       summary register clears the bit. NOTE- this addition was made
                          1213  3155 ;       necessary by the addition of TM78 support in the MASSBUS interrupt
                          1213  3156 ;       dispatcher (MBAINTDSP).  In effect all Device Drivers for MASSBUS
                          1213  3157 ;       multi-device controllers now have the responsibility of clearing
                          1213  3158 ;       their attention bit.
                          1213  3159
                          1213  3160          ASSUME  ADP$L_CSR  EQ  0
          55   14 B3    D0  1213  3161          MOVL    @IDB$L_ADP(R3),R5      ; R5 => MBA-CSR
       52   0400 C5    9E  1217  3162          MOVAB   MBA$L_ERB(R5),R2       ; R2 => BASE OF MBA EXTERNAL REGISTERS
       52     54    52  C3  121C  3163          SUBL3   R2,R4,R2               ; R2 = DISTANCE OF TM03-CSR FROM BASE
    52   52    F9 8F    78  1220  3164          ASHL    #-7,R2,R2              ; R2 = MBA PORT NUMBER.
0410 C5     01    52    78  1225  3165          ASHL    R2,#1,MBA$L_AS(R5)     ; CLEAR ATTENTION SUMMARY BIT.
              24 A4    DD  122B  3166          PUSHL   MT_TC(R4)              ;SAVE CONTENTS OF TAPE CONTROL REGISTER
          55   04 A3    D0  122E  3167          MOVL    IDB$L_OWNER(R3),R5     ;GET OWNER UCB ADDRESS
                 0C    13  1232  3168          BEQL    10$                    ;IF EQL NONE
       07 64 A5    01    E5  1234  3169          BBCC    #UCB$V_INT,UCB$W_STS(R5),10$ ;IF CLR, INTERRUPT NOT EXPECTED
          53   10 A5    7D  1239  3170          MOVQ    UCB$L_FR3(R5),R3       ;RESTORE DRIVER CONTEXT
                 0C B5    16  123D  3171          JSB     @UCB$L_FPC(R5)         ;CALL DRIVER
```

```
                7E   D4   1240  3172 10$:    CLRL    -(SP)                   ;SET STARTING UNIT NUMBER
                          1242  3173 20$:
             52 6E   D0   1242  3174         MOVL    (SP),R2                 ; Unit number to register
          53   08 BE   D0   1245  3175         MOVL    @8(SP),R3               ;RETRIEVE ADDRESS OF IDB
             54   63   D0   1249  3176         MOVL    IDB$L_CSR(R3),R4        ;RETRIEVE ADDRESS OF TM03-TE16/TU77 REGISTER
          24 A4   52   D0   124C  3177         MOVL    R2,MT_TC(R4)            ;SELECT NEXT SLAVE UNIT
       55   18 A342   D0   1250  3178         MOVL    IDB$L_UCBLST(R3)[R2],R5 ;GET ADDRESS OF UCB
                52   13   1255  3179         BEQL    60$                     ;IF EQL NONE
          53   04 A4   D0   1257  3180         MOVL    MT_DS(R4),R3            ;READ SLAVE DRIVE STATUS
                          125B  3181
                          125B  3182 ;+
                          125B  3183 ; POLL STATUS FOR ATTENTION CONDITIONS
                          125B  3184 ;
                          125B  3185 ; THE SEQUENCE IN WHICH ATTENTION CONDITIONS ARE POLLED IS CRITICAL.
                          125B  3186 ; DUE TO AN ERROR IN THE TE-16 DRIVE DESIGN, THE ATTEMPT TO RESET AN
                          125B  3187 ; ATTENTION CONDITION WITH DRIVE-CLEAR, WHILE REWINDING, RESULTS IN
                          125B  3188 ; AN RMR ERROR (REGISTER MODIFICATION REFUSED). SINCE ALL ERRORS ARE
                          125B  3189 ; CLEARED WITH DRIVE-CLEAR, RMR IS REASSERTED SWAMPING THE SYSTEM
                          125B  3190 ; WITH INTERRUPTS UNTIL THE REWIND COMPLETES.
                          125B  3191 ;
                          125B  3192 ; TO AVOID THIS CONDITION, NO ATTENTION CONDITIONS ARE SERVICED IF
                          125B  3193 ; PIP IS SET FOR THE DRIVE. THIS HAS THE EFFECT OF BLOCKING INTERR-
                          125B  3194 ; UPTS FROM OTHER DRIVES BUT IS PREFERRABLE TO LOCKING UP THE ENTIRE
                          125B  3195 ; SYSTEM. THE CORRECT REMEDY IS TO FIX THE PROBLEM IN THE DRIVE ELEC-
                          125B  3196 ; TRONICS BY BLOCKING ALL ATTENTION CONDITIONS UNTIL REWIND IS DONE
                          125B  3197 ; (AS FOR THE TU-77). UNDER THESE CONDITIONS, PIP SHOULD NEVER OCCUR
                          125B  3198 ; IN CONJUNCTION WITH OTHER ATTENTION CONDITIONS.
                          125B  3199 ;
                          125B  3200 ;-
                          125B  3201
          1E 53   00   E0  125B  3202         BBS     #MT_DS_V_SLA,R3,30$     ;IF SET, SLAVE TRANSITION TO ONLINE
          27 53   0C   E1  125F  3203         BBC     #MT_DS_V_MOL,R3,40$     ;IF CLR, MEDIUM OFFLINE
          45 53   0D   E0  1263  3204         BBS     #MT_DS_V_PIP,R3,70$     ;IF SET POSITIONING IN PROGRESS
          41 53   01   E1  1267  3205         BBC     #MT_DS_V_BOT,R3,70$     ;IF CLR, NOT AT BEGINNING OF TAPE
          68 A5   01   AA  126B  3206         BICW    #UCB$M_MT_REWIND,UCB$W_DEVSTS(R5) ;CLEAR REWIND IN PROGRESS
       35 64 A5   01   E5  126F  3207         BBCC    #UCB$V_INT,UCB$W_STS(R5),60$ ;IF CLR, INTERRUPT NOT EXPECTED
          53   10 A5   7D  1274  3208         MOVQ    UCB$L_FR3(R5),R3        ;RESTORE DRIVER CONTEXT
               0C B5   16  1278  3209         JSB     @UCB$L_FPC(R5)          ;CALL DRIVER
                   2F   11  127B  3210         BRB     70$                     ;
          68 A5   02   A8  127D  3211 30$:    BISW    #UCB$M_MT_PRVMOL,UCB$W_DEVSTS(R5) ;SET PREVIOUS MOL STATE
          68 A5   01   AA  1281  3212         BICW    #UCB$M_MT_REWIND,UCB$W_DEVSTS(R5) ;CLEAR REWIND IN PROGRESS
                FEE7   30  1285  3213         BSBW    TM_UNSOLNT              ;CALL UNSOLICITED INTERRUPT ENTRY
                   22   11  1288  3214         BRB     70$                     ;
       1D 68 A5   01   E5  128A  3215 40$:    BBCC    #UCB$V_MT_PRVMOL,UCB$W_DEVSTS(R5),70$ ;IF CLR, PREVIOUSLY OFFLINE
          68 A5   01   AA  128F  3216         BICW    #UCB$M_MT_REWIND,UCB$W_DEVSTS(R5) ;CLEAR REWIND IN PROGRESS
          7E   64 A5   3C  1293  3217         MOVZWL  UCB$W_STS(R5),-(SP)     ;SAVE CURRENT DRIVE STATUS
               FED5   30  1297  3218         BSBW    TM_UNSOLNT              ;CALL UNSOLICITED INTERRUPT ENTRY
          06 6E   0B   E1  129A  3219         BBC     #UCB$V_VALID,(SP),45$   ;DON'T RESET VOLUME VALID IF FLAG WAS CLEAR
    64 A5    0800 8F   A8  129E  3220         BISW    #UCB$M_VALID,UCB$W_STS(R5) ;RESET SOFTWARE VOLUME VALID
          5E   04   C0  12A4  3221 45$:    ADDL    #4,SP                   ;CLEAN THE STACK
                   03   11  12A7  3222         BRB     70$                     ;
          64   09   9A  12A9  3223 60$:    MOVZBL  #F_DRVCLR!1,MT_CS1(R4)  ;CLEAR DRIVE
          92 6E   07   F3  12AC  3224 70$:    AOBLEQ  #7,(SP),20$             ;ANY MORE SLAVE UNITS TO SCAN?
                          12B0  3225                                          ; Refresh R3 and R4 after falling thru.
          53   08 BE   D0  12B0  3226         MOVL    @8(SP),R3               ; R3 => IDB
             54   63   D0  12B4  3227         MOVL    IDB$L_CSR(R3),R4        ; R4 => TM03-TE16/TU77 REGISTERS
                   8E   D5  12B7  3228         TSTL    (SP)+                   ; Clear loop count from stack
```

```
                52   04 A4    19    78   12B9  3229          ASHL    #31-MT_DS_V_SSC,MT_DS(R4),R2 ;ANY SLAVE CHANGE STATUS
                              80    19   12BE  3230          BLSS    10$                          ;IF LSS YES
                        24 A4 8ED0  12C0   3231          POPL    MT_TC(R4)                    ;RESTORE CONTENTS OF TAPE CONTROL REGISTER
                     5E   04   C0  12C4   3232          ADDL    #4,SP                        ;CLEAN STACK
                     52   8E   7D  12C7   3233          MOVQ    (SP)+,R2                     ;RESTORE REGISTERS
                     54   8E   7D  12CA   3234          MOVQ    (SP)+,R4                     ;
                                02  12CD   3235          REI                                  ;
                                    12CE   3236 TM_END:                                      ;ADDRESS OF LAST LOCATION IN DRIVER
                                    12CE   3237
                                    12CE   3238          .END
```

```
$$$                     =  00000020 R     02
$$OP                    =  00000002
$.$                     =  000096E0
ACP$ACCESS                 ********   X   03
ACP$DEACCESS               ********   X   03
ACP$MODIFY                 ********   X   03
ACP$MOUNT                  ********   X   03
ACP$READBLK                ********   X   03
ACP$WRITEBLK               ********   X   03
ADP$L_CSR               =  00000C00
AT$_MBA                 =  00000000
CDF_DRVCLR              =  00000004
CDF_ERASE               =  00000006
CDF_INTSPCFOR           =  00000014
CDF_INTSPCREV           =  00000015
CDF_NOP                 =  00000000
CDF_PACKACK             =  00000008
CDF_READDATA            =  0000000C
CDF_READDATAR           =  0000000F
CDF_READPRESET          =  00000010
CDF_REWIND              =  00000003
CDF_SENSECHAR           =  00000012
CDF_SETCHAR             =  00000011
CDF_SPCFILFOR           =  00000002
CDF_SPCFILREV           =  00000005
CDF_SPCRECFOR           =  00000009
CDF_SPCRECREV           =  00000007
CDF_UNLOAD              =  00000001
CDF_WRITE               =  0000000E
CDF_WRITECHECK          =  0000000A
CDF_WRITECHECKR         =  0000000D
CDF_WRITEDATA           =  0000000B
CDF_WRITEMARK           =  00000013
CHECK_ERROR                00000948 R     03
CRB$L_INTD              =  00000024
CRB$L_LINK              =  00000020
DC$_TAPE                =  00000002
DDB$L_ACPD              =  00000010
DDB$L_DDT               =  0000000C
DENSITY                    00000038 R     03
DEV$M_AVL               =  00040000
DEV$M_DIR               =  00000008
DEV$M_ELG               =  00400000
DEV$M_FOD               =  00004000
DEV$M_IDV               =  04000000
DEV$M_NNM               =  00000200
DEV$M_ODV               =  08000000
DEV$M_SDI               =  00000010
DEV$M_SQD               =  00000020
DEV$V_FOR               =  00000018
DEV$V_MNT               =  00000013
DOUBLE                     00000A00 R     03
DPT$C_LENGTH            =  00000038
DPT$C_VERSION           =  00000004
DPT$INITAB                 00000038 R     02
DPT$M_SUBCNTRL          =  00000001
DPT$REINITAB               00000070 R     02
DPT$TAB                    00000000 R     02
DRVCLR                     000003E9 R     03
DRVREG                     00000D11 R     03
DT$_TE16                =  00000001
DT$_TU45                =  00000002
DT$_TU77                =  00000003
DYN$C_CRB               =  00000005
DYN$C_DDB               =  00000006
DYN$C_DPT               =  0000001E
DYN$C_UCB               =  00000010
EMB$L_DV_REGSAV         =  0000004E
ENBEXT                     00000C30 R     03
ENBXIT                     00000D1C R     03
ERASE                      0000039E R     03
ERL$DEVICERR               ********     X 03
ERL$DEVICTMO               ********     X 03
ERR_SPACING             =  00000005
EXE$IOFORK                 ********     X 03
EXE$ONEPARM                ********     X 03
EXE$SETMODE                ********     X 03
EXE$SNDEVMSG               ********     X 03
EXE$ZEROPARM               ********     X 03
FATALERR                   00000A04 R     03
FDISPATCH                  000002C5 R     03
FEX                        00000B3E R     03
FORMAT                     0000006C R     03
FTAB                       00000056 R     03
FUNCTAB_LEN             =  00000088
FUNCXT                     00000A73 R     03
F_DRVCLR                =  00000008
F_ERASE                 =  00000014
F_INTSPCFOR             =  00000028
F_INTSPCREV             =  0000002E
F_NOP                   =  00000000
F_PACKACK               =  00000000
F_READDATA              =  00000038
F_READDATAR             =  0000003E
F_READPRESET            =  00000010
F_REWIND                =  00000006
F_SENSECHAR             =  00000000
F_SETCHAR               =  00000000
F_SPCFILFOR             =  00000018
F_SPCFILREV             =  0000001A
F_SPCRECFOR             =  00000018
F_SPCRECREV             =  0000001A
F_UNLOAD                =  00000002
F_WRITE                 =  00000030
F_WRITECHECK            =  00000028
F_WRITECHECKR           =  0000002E
F_WRITEDATA             =  00000030
F_WRITEMARK             =  00000016
IDB$L_ADP               =  00000014
IDB$L_CSR               =  00000000
IDB$L_OWNER             =  00000004
IDB$L_UCBLST            =  00000018
IMMED                      00000C1F R     03
INTSPC                     00000B7D R     03
```

```
IO$M_DATACHECK          =  00004000
IO$M_NOWAIT             =  00000080
IO$V_DATACHECK          =  0000000E
IO$V_INHEXTGAP          =  0000000C
IO$V_INHRETRY           =  0000000F
IO$V_NOWAIT             =  00000007
IO$V_REVERSE            =  00000006
IO$_ACCESS              =  00000032
IO$_ACPCONTROL          =  00000038
IO$_AVAILABLE           =  00000011
IO$_CREATE              =  00000033
IO$_DEACCESS            =  00000034
IO$_DELETE              =  00000035
IO$_DRVCLR              =  00000004
IO$_ERASETAPE           =  00000006
IO$_MODIFY              =  00000036
IO$_MOUNT               =  00000039
IO$_NOP                 =  00000000
IO$_PACKACK             =  00000008
IO$_READLBLK            =  00000021
IO$_READPBLK            =  0000000C
IO$_READPRESET          =  00000019
IO$_READVBLK            =  00000031
IO$_RECAL               =  00000003
IO$_REWIND              =  00000024
IO$_REWINDOFF           =  00000022
IO$_SENSECHAR           =  0000001B
IO$_SENSEMODE           =  00000027
IO$_SETCHAR             =  0000001A
IO$_SETMODE             =  00000023
IO$_SKIPFILE            =  00000025
IO$_SKIPRECORD          =  00000026
IO$_SPACEFILE           =  00000002
IO$_SPACERECORD         =  00000009
IO$_UNLOAD              =  00000001
IO$_VIRTUAL             =  0000003F
IO$_WRITECHECK          =  0000000A
IO$_WRITELBLK           =  00000020
IO$_WRITEMARK           =  0000001C
IO$_WRITEOF             =  00000028
IO$_WRITEPBLK           =  0000000B
IO$_WRITEVBLK           =  00000030
IOC$CANCELIO            ********  X  03
IOC$DIAGBUFILL          ********  X  03
IOC$LOADMBAMAP          ********  X  03
IOC$MNTVER              ********  X  03
IOC$RELCHAN             ********  X  03
IOC$RELSCHAN            ********  X  03
IOC$REQCOM              ********  X  03
IOC$REQPCHANL           ********  X  03
IOC$REQSCHANL           ********  X  03
IOC$RETURN              ********  X  03
IOC$WFIKPCH             ********  X  03
IOC$WFIRLCH             ********  X  03
IRP$L_MEDIA             =  00000038
IRP$L_SVAPTE            =  0000002C
IRP$L_WIND              =  00000018
IRP$S_FCODE             =  00000006
IRP$V_FCODE             =  00000000
IRP$V_PHYSIO            =  00000008
IRP$V_VIRTUAL           =  00000004
IRP$W_BCNT              =  00000032
IRP$W_BOFF              =  00000030
IRP$W_FUNC              =  00000020
IRP$W_STS               =  0000002A
LOSTPOS                    000009ED  R      03
MASKH                   =  00000008
MASKL                   =  04000000
MBA$L_AS                =  00000410
MBA$L_BCR               =  00000010
MBA$L_CR                =  00000004
MBA$L_CSR               =  00000000
MBA$L_ERB               =  00000400
MBA$L_MAP               =  00000800
MBA$L_SR                =  00000008
MBA$L_VAR               =  0000000C
MBA$M_CR_ABORT          =  00000002
MBA$M_CR_IE             =  00000004
MBA$M_CR_INIT           =  00000001
MBA$M_ERROR             =  000E5FFF
MBA$M_SR_ATTN           =  00010000
MBA$M_SR_DLT            =  00000800
MBA$M_SR_DTABT          =  00001000
MBA$M_SR_DTCOMP         =  00002000
MBA$M_SR_ERCONF         =  00000008
MBA$M_SR_INVMAP         =  00000010
MBA$M_SR_ISTO           =  00000002
MBA$M_SR_MAPPE          =  00000020
MBA$M_SR_MBEXC          =  00000080
MBA$M_SR_MCPE           =  00020000
MBA$M_SR_MDPE           =  00000040
MBA$M_SR_MXF            =  00000100
MBA$M_SR_NED            =  00040000
MBA$M_SR_PGE            =  00080000
MBA$M_SR_RDS            =  00000004
MBA$M_SR_RDTO           =  00000001
MBA$M_SR_SPE            =  00004000
MBA$M_SR_WCKLWR         =  00000200
MBA$M_SR_WCKUPR         =  00000400
MBA$V_SR_NED            =  00000012
MEDIA_ID_TE16           =  6D285010
MEDIA_ID_TU45           =  6D29502D
MEDIA_ID_TU77           =  6D29504D
MIN_RECORD              =  0000000E
MMG$GL_SPTBASE             ********  X  03
MSG$_DEVOFFLIN             ********  X  03
MT$CHECK_ACCESS            ********  X  03
MT$K_NRZI_800           =  00000003
MT$K_PE_1600            =  00000004
MT$M_BOT                =  00010000
MT$M_DENSITY            =  00001F00
MT$M_EOF                =  00020000
MT$M_EOT                =  00040000
MT$M_FORMAT             =  000000F0
```

```
MT$M_HWL                =  00080000
MT$M_LOST               =  00100000
MT$M_PARITY             =  00000008
MT$S_DENSITY            =  00000005
MT$S_FORMAT             =  00000004
MT$V_DENSITY            =  00000008
MT$V_FORMAT             =  00000004
MT$V_PARITY             =  00000003
MT_AS                      00000010
MT_CC                      0000001C
MT_CS1                     00000000
MT_CS1_M_GO             =  00000001
MT_DS                      00000004
MT_DS_M_BOT             =  00000002
MT_DS_M_PES             =  00000020
MT_DS_M_TM              =  00000004
MT_DS_V_BOT             =  00000001
MT_DS_V_EOT             =  0000000A
MT_DS_V_MOL             =  0000000C
MT_DS_V_PES             =  00000005
MT_DS_V_PIP             =  0000000D
MT_DS_V_SLA             =  00000000
MT_DS_V_SSC             =  00000006
MT_DS_V_TM              =  00000002
MT_DS_V_WRL             =  0000000B
MT_DT                      00000018
MT_ER                      00000008
MT_ER_M_COR             =  00008000
MT_ER_M_CPAR            =  00000008
MT_ER_M_CRC             =  00008000
MT_ER_M_CS              =  00000400
MT_ER_M_DPAR            =  00000020
MT_ER_M_DTE             =  00001000
MT_ER_M_FCE             =  00000200
MT_ER_M_FMT             =  00000010
MT_ER_M_ILF             =  00000001
MT_ER_M_ILR             =  00000002
MT_ER_M_INC             =  00000040
MT_ER_M_ITM             =  00000400
MT_ER_M_LRC             =  00000080
MT_ER_M_NEF             =  00000800
MT_ER_M_NSG             =  00000100
MT_ER_M_OPI             =  00002000
MT_ER_M_PEF             =  00000080
MT_ER_M_RMR             =  00000004
MT_ER_M_UNS             =  00004000
MT_ER_M_VPE             =  00000040
MT_ER_V_CPAR            =  00000003
MT_ER_V_CRC             =  0000000F
MT_ER_V_FMT             =  00000004
MT_ER_V_ITM             =  0000000A
MT_ER_V_LRC             =  00000007
MT_ER_V_NEF             =  0000000B
MT_ER_V_OPI             =  0000000D
MT_ER_V_UNS             =  0000000E
MT_ER_V_VPE             -  00000006
MT_FC                      00000014
MT_MR                      0000000C
MT_SN                      00000020
MT_TC                      00000024
MT_TC_M_EPAR            =  00000008
MT_TC_S_DEN             =  00000003
MT_TC_S_FSEL            =  00000004
MT_TC_V_DEN             =  00000008
MT_TC_V_FSEL            =  00000004
NOP                        000003E9  R      03
NORXIT                     00000627  R      03
NRZI                    =  00000003
PACKACK                    000003DC  R      03
PE                      =  00000004
POSIT                      00000BF8  R      03
PR$_IPL                    ********    X    03
READDATA                   00000668  R      03
READDATAR                  00000784  R      03
READPRESET                 00000467  R      03
RECAL                      00000C33  R      03
RESETXFR                   00000E67  R      03
RESTART                    00000B48  R      03
RETREG                     00000D36  R      03
RETRY                      000009F1  R      03
REWIND                     0000046F  R      03
SENSECHAR                  000003E9  R      03
SETCHAR                    000003F0  R      03
SETDEN                     00000CE8  R      03
SETEOF                     00000651  R      03
SETEOV                     0000062C  R      03
SIZ...                  =  00000001
SPCFILFOR                  00000490  R      03
SPCFILREV                  00000531  R      03
SPCRECFOR                  0000055D  R      03
SPCRECREV                  000005F1  R      03
SS$_ABORT               =  0000002C
SS$_CTRLERR             =  00000054
SS$_DATACHECK           =  0000005C
SS$_DATAOVERUN          =  00000838
SS$_DRVERR              =  0000008C
SS$_ENDOFFILE           =  00000870
SS$_ENDOFTAPE           =  00000878
SS$_ENDOFVOLUME         =  000009A0
SS$_FORMAT              =  000000BC
SS$_MEDOFL              =  000001A4
SS$_NONEXDRV            =  000001C4
SS$_NORMAL              =  00000001
SS$_OPINCOMPL           =  000002D4
SS$_PARITY              =  000001F4
SS$_TIMEOUT             =  0000022C
SS$_UNSAFE              =  0000023C
SS$_VOLINV              =  00000254
SS$_WRITLCK             =  0000025C
STSXIT                     00000B34  R      03
SYS$GL_OPRMBX              ********    X    03
TESTR                      00000994  R      03
THRESHOLD               =  00000008
TIMEOUT                    00000E62  R      03
```

```
TIME_OUT                        00000074 R      03
TM$DDT                          00000000 RG     03
TM$INT                          0000120C RG     03
TM_CANCELIO                     00000180 R      03
TM_DTDESC                       00000048 R      03
TM_DTDESCLEN                  = 00000003
TM_DTYPE                        0000103C R      03
TM_END                          000012CE R      03
TM_FUNCTABLE                    000000F8 R      03
TM_REGDUMP                      00001098 R      03
TM_SAVDRVSTS                    000011B3 R      03
TM_SETINTSPC                    000011DA R      03
TM_STARTIO                      000001FB R      03
TM_TXXX_INIT                    000010F5 R      03
TM_UNSOCNT                      0000116F R      03
UCB$B_CEX                     = 00000093
UCB$B_DEVCLASS                = 00000040
UCB$B_DEVTYPE                 = 00000041
UCB$B_DIPL                    = 0000005E
UCB$B_ERTCNT                  = 00000080
UCB$B_ERTMAX                  = 00000081
UCB$B_FEX                     = 00000092
UCB$B_FIPL                    = 0000000B
UCB$B_SLAVE                   = 00000090
UCB$K_LCL_TAPE_LENGTH         = 000000B4
UCB$K_MT_LENGTH               = 000000D4
UCB$L_CRB                     = 00000024
UCB$L_DEVCHAR                 = 00000038
UCB$L_DEVCHAR2                = 0000003C
UCB$L_DEVDEPEND               = 00000044
UCB$L_DPC                     = 0000009C
UCB$L_FPC                     = 0000000C
UCB$L_FR3                     = 00000010
UCB$L_IOQFL                   = 0000004C
UCB$L_IRP                     = 00000058
UCB$L_MEDIA_ID                = 0000008C
UCB$L_MT_ORGPOS                 000000CC
UCB$L_MT_PREVTM                 000000C8
UCB$L_MT_SR                     000000B4
UCB$L_RECORD                  = 000000B0
UCB$L_SVAPTE                  = 00000078
UCB$L_VCB                     = 00000034
UCB$M_INT                     = 00000002
UCB$M_MT_CNCLP                = 00000008
UCB$M_MT_PRVMOL               = 00000002
UCB$M_MT_PWRFL                = 00000004
UCB$M_MT_REWIND               = 00000001
UCB$M_ONLINE                  = 00000010
UCB$M_POWER                   = 00000020
UCB$M_TIM                     = 00000001
UCB$M_TIMOUT                  = 00000040
UCB$M_VALID                   = 00000800
UCB$V_CANCEL                  = 00000003
UCB$V_INT                     = 00000001
UCB$V_MT_CNCLP                = 00000003
UCB$V_MT_PRVMOL               = 00000001
UCB$V_MT_PWRFL                = 00000002
UCB$V_MT_REWIND               = 00000000
UCB$V_ONLINE                  = 00000004
UCB$V_POWER                   = 00000005
UCB$V_TIMOUT                  = 00000006
UCB$V_VALID                   = 0000000B
UCB$W_BCNT                    = 0000007E
UCB$W_BOFF                    = 0000007C
UCB$W_DEVBUFSIZ               = 00000042
UCB$W_DEVSTS                  = 00000068
UCB$W_FUNC                    = 0000009A
UCB$W_MT_CC_SAV                 000000D0
UCB$W_MT_CST                    000000C0
UCB$W_MT_DS                     000000B8
UCB$W_MT_ER                     000000BA
UCB$W_MT_FC                     000000BC
UCB$W_MT_FORCNT                 000000C4
UCB$W_MT_SPACNT                 000000BE
UCB$W_MT_TC                     000000C6
UCB$W_MT_TC_SAV                 000000C2
UCB$W_STS                     = 00000064
UCB$W_UNIT                    = 00000054
UNLOAD                          0000046F R      03
VA$S_VPN                      = 00000015
VA$V_VPN                      = 00000009
VEC$L_IDB                     = 00000008
VEC$L_UNITINIT                = 00000018
WCB$W_NMAP                    = 00000016
WRITECHECK                      00000661 R      03
WRITECHECKR                     0000077D R      03
WRITEDATA                       000008C8 R      03
WRITEMARK                       00000363 R      03
XFER                            00000C9C R      03
XTAB                            000000CC 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	000000D4 (212.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
$$$105_PROLOGUE	00000080 (128.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
$$$115_DRIVER	000012CE (4814.)	03 (3.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	34	00:00:00.03	00:00:01.53
Command processing	121	00:00:00.36	00:00:02.76
Pass 1	709	00:00:23.73	00:01:23.56
Symbol table sort	0	00:00:02.77	00:00:08.51
Pass 2	410	00:00:06.39	00:00:22.57
Symbol table output	1	00:00:00.22	00:00:00.73
Psect synopsis output	0	00:00:00.02	00:00:00.12
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	1277	00:00:33.52	00:01:59.91

The working set limit was 2550 pages.
200683 bytes (392 pages) of virtual memory were used to buffer the intermediate code.
There were 140 pages of symbol table space allocated to hold 2484 non-local and 183 local symbols.
3238 source lines were read in Pass 1, producing 31 object records in Pass 2.
55 pages of virtual memory were used to define 53 macros.

Macro library statistics

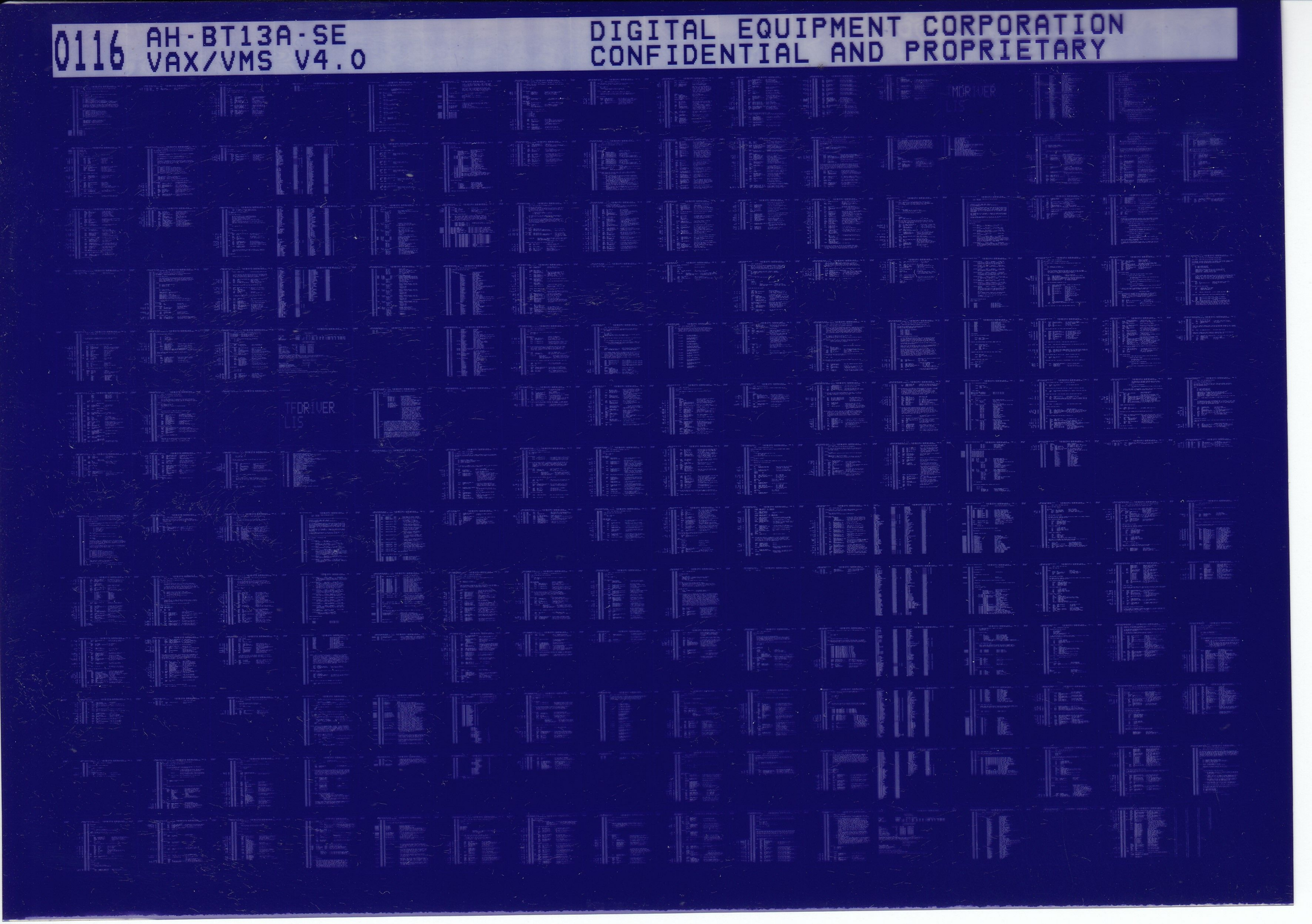
Macro library name	Macros defined
_255DUA28:[SYS.OBJ]LIB.MLB;1	36
_255DUA28:[SYSLIB]STARLET.MLB;2	11
TOTALS (all libraries)	47

2615 GETS were required to define 47 macros.

There were no errors, warnings or information messages.

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

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

TUDRIVER
LIS

XADRIVER
LIS

TSDRIVER
LIS