C 8

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

FILEID**TMDRIVER

B 13

```
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
```

Table of contents

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

```
0000    ;
0000    ;       V03-008 RLRDPATH1       Robert L. Rappaport     31-May-1983
0000    ;               Allow UCB to include new DUAL PORT extension by
0000    ;               changing base of where we begin the private TMDRIVER
0000    ;               extension from UCB$L_DPC+4 to UCB$L_DP_LINK+4.
0000    ;
0000    ;       V03-007 RLR54598        Robert L. Rappaport     8-Mar-1982
0000    ;               Correct problem that crashes system when dealing with
0000    ;               unit number 7.
0000    ;
0000    ;       V03-006 RLR50799        Robert L. Rappaport     15-Dec-1982
0000    ;               Correct problem that caused clearing of Volume Valid
0000    ;               when attempting to mount or init tapes.
0000    ;
0000    ;       V03-005 RLR47029        Robert L. Rappaport     17-Aug-1982
0000    ;               Correct error that indicated records skipped when
0000    ;               skip reverse record occurred at BOT.  Also reset density
0000    ;               in DEVDEPEND after POSITIONING operations.
0000    ;
0000    ;       V03-004 RLREC002        Robert L. Rappaport     20-July-1982
0000    ;               Fix second error introduced by fix to FCE that caused 800
0000    ;               bpi tapes to ignore Tape Marks on reads.
0000    ;
0000    ;       V03-003 KDM0002         Kathleen D. Morse       28-Jun-1982
0000    ;               Added $DYNDEF, $DCDEF, $SSDEF,  and $VADEF.
0000    ;
0000    ;       V03-002 RLREC001a       Robert L. Rappaport     21-June-1982
0000    ;               Correct error introduced in the elimination of testing
0000    ;               for FCR (Frame Count Error) errors on reads.  This
0000    ;               inadvertently eliminated testing for Tape Marks.
0000    ;
0000    ;       V03-001 RLREC001        Robert L. Rappaport     17-May-1982
0000    ;               SPR44595.
0000    ;               Ignore FCE (Frame Count Error) errors on Read commands
0000    ;               and thereby allow Reads with IO$M_INHRETRY set to
0000    ;               succeed when the buffer is larger than the record.
0000    ;
0000    ;               QAR 1404.
0000    ;               Do not request Primary channel until possible REWIND
0000    ;               done in FDISPATCH.  Also RELCHAN before REQCOM.
0000    ;
0000    ;               Only try to clear SLA bit, once per interrupt, per drive
0000    ;               in TM$INT.
0000    ;               This tracks patch made in V3.1.
0000    ;
0000    ;
0000    ; TM03-TE16/TU77 MAGTAPE DRIVER
0000    ;
0000    ; MACRO LIBRARY CALLS
0000    ;
0000
0000            $ADPDEF                         ;DEFINE ADP OFFSETS
0000            $CRBDEF                         ;DEFINE CRB OFFSETS
0000            $DCDEF                          ;DEFINE DEVICE CLASSES
0000            $DDBDEF                         ;DEFINE DDB OFFSETS
0000            $DEVDEF                         ;DEFINE DEVICE CHARACTERISTICS BITS
0000            $DPTDEF                         ;DEFINE DPT OFFSETS
```

```
0000            $DYNDEF                               ;DEFINE DYNAMIC DATA STRUCTURE TYPES
0000            $EMBDEF                               ;DEFINE EMB OFFSETS
0000            $IDBDEF                               ;DEFINE IDB OFFSETS
0000            $IODEF                                ;DEFINE I/O FUNCTION CODES
0000            $IRPDEF                               ;DEFINE IRP OFFSETS
0000            $MBADEF                               ;DEFINE MBA REGISTER OFFSETS
0000            $MTDEF                                ;DEFINE MAGTAPE STATUS BITS
0000            $SSDEF                                ;DEFINE SYSTEM STATUS CODES
0000            $UCBDEF                               ;DEFINE UCB OFFSETS
0000            $VADEF                                ;DEFINE VIRTUAL ADDRESS FIELDS
0000            $VECDEF                               ;DEFINE INTERRUPT DISPATCH VECTOR OFFSETS
0000            $WCBDEF                               ;DEFINE WCB OFFSETS
0000
0000  ;
0000  ; LOCAL MACROS
0000  ;
0000  ; CHECK FOR FATAL OR RETRIABLE ERROR
0000  ;
0000
0000            .MACRO  CHECK_ERROR
0000                    BSBW    CHECK_ERROR
0000            .ENDM   CHECK_ERROR
0000
0000  ;
0000  ; EXECUTE FUNCTION AND BRANCH ON RETRIABLE ERROR CONDITION
0000  ;
0000
0000            .MACRO  EXFUNC BDST,FCODE
0000                    .IF NB  FCODE
0000                    MOVZBL  #CD'FCODE,R0
0000                    .ENDC
0000                    BSBW    FEX
0000                    .WORD   BDST-.-2
0000            .ENDM   EXFUNC
0000
0000  ;
0000  ; GENERATE FUNCTION TABLE ENTRY AND CASE TABLE INDEX SYMBOL
0000  ;
0000
0000            .MACRO  GENF FCODE
0000                    CD'FCODE=.-FTAB
0000                    .BYTE   FCODE!MT_CS1_M_GO
0000            .ENDM   GENF
0000
0000  ;
0000  ; GENERATE ERROR MASK TABLE ENTRY
0000  ;
0000
0000            .MACRO  MASK LIST
0000  $.$=0
0000            .IRP    X,<LIST>
0000  $.$=$.$!MT_ER_M_'X
0000            .ENDM
0000            .WORD   $.$
0000            .ENDM   MASK
0000
0000  ;
```

```
          0000  ; TEST IF ANY RETRIES REMAINING
          0000  ;
          0000
          0000          .MACRO  TESTR BDST
          0000                  BSBW    TESTR
          0000                  .WORD   BDST-.-2
          0000          .ENDM   TESTR
          0000
          0000  ;
          0000  ; LOCAL SYMBOLS
          0000
6D285010  0000  MEDIA_ID_TE16 = ^X<6D285010>             ; Media id for the TE16
6D29502D  0000  MEDIA_ID_TU45 = ^X<6D29502D>             ; Media id for the TU45
6D29504D  0000  MEDIA_ID_TU77 = ^X<6D29504D>             ; Media id for the TU77
          0000
          0000  ; TE16/TU77 MASSBUS REGISTER OFFSETS
          0000  ;
          0000
          0000          $DEFINI MT
          0000
          0000  $DEF    MT_CS1          .BLKL   1        ;DRIVE CONTROL REGISTER
          0004          _VIELD  MT_CS1,0,<-              ; DRIVE CONTROL REGISTER BIT DEFINITIONS
          0004                  <GO,,M>,-                ; GO BIT
          0004                  <FCODE,5>-               ; FUNCTION CODE
          0004          >                                ;
          0004  $DEF    MT_DS           .BLKL   1        ;DRIVE STATUS REGISTER
          0008          _VIELD  MT_DS,0,<-               ; DRIVE STATUS REGISTER BIT DEFINITIONS
          0008                  <SLA,,M>,-               ; SLAVE ATTENTION
          0008                  <BOT,,M>,-               ; BEGINNING OF TAPE
          0008                  <TM,,M>,-                ; TAPE MARK
          0008                  <IDB,,M>,-               ; IDENTIFICATION BURST
          0008                  <SDWN,,M>,-              ; SLOWING DOWN
          0008                  <PES,,M>,-               ; PHASE ENCODED
          0008                  <SSC,,M>,-               ; SLAVE STATUS CHANGE
          0008                  <DRY,,M>,-               ; DRIVE READY
          0008                  <DPR,,M>,-               ; DRIVE PRESENT
          0008                  <,1>,-                   ; RESERVED BIT
          0008                  <EOT,,M>,-               ; END OF TAPE
          0008                  <WRL,,M>,-               ; DRIVE WRITE LOCKED
          0008                  <MOL,,M>,-               ; MEDIUM ONLINE
          0008                  <PIP,,M>,-               ; POSITIONING IN PROGRESS
          0008                  <ERR,,M>,-               ; COMPOSIT ERROR
          0008                  <ATA,,M>-                ; ATTENTION ACTIVE
          0008          >                                ;
          0008  $DEF    MT_ER           .BLKL   1        ;ERROR REGISTER
          000C          _VIELD  MT_ER,0,<-               ; ERROR REGISTER BIT DEFINITIONS
          000C                  <ILF,,M>,-               ; ILLEGAL FUNCTION
          000C                  <ILR,,M>,-               ; ILLEGAL REGISTER
          000C                  <RMR,,M>,-               ; REGISTER MODIFY REFUSED
          000C                  <CPAR,,M>,-              ; CONTROL BUS PARITY ERROR
          000C                  <FMT,,M>,-               ; FORMAT ERROR
          000C                  <DPAR,,M>,-              ; DATA BUS PARITY ERROR
          000C                  <VPE,,M>,-               ; VERTICLE PARITY ERROR (NRZI)
          000C                  <LRC,,M>,-               ; LONGITUDINAL PARITY ERROR (NRZI)
          000C                  <NSG,,M>,-               ; NONSTANDARD GAP
          000C                  <FCE,,M>,-               ; FRAME COUNT ERROR
          000C                  <ITM,,M>,-               ; ILLEGAL TAPE MARK
```

```
                 000C          <NEF,,M>,-                  ; NONEXECUTABLE FUNCTION
                 000C          <DTE,,M>,-                  ; DRIVE TIMING ERROR
                 000C          <OPI,,M>,-                  ; OPERATION INCOMPLETE
                 000C          <UNS,,M>,-                  ; DRIVE UNSAFE
                 000C          <CRC,,M>,-                  ; CRC ERROR (NRZI)
                 000C      >                               ;
                 000C ; Explicitly define alternate names for some bits in MT_ER.
                 000C
        00000040 000C MT_ER_M_INC=MT_ER_M_VPE
        00000006 000C MT_ER_V_INC=MT_ER_V_VPE
        00000080 000C MT_ER_M_PEF=MT_ER_M_LRC
        00000007 000C MT_ER_V_PEF=MT_ER_V_LRC
        00000400 000C MT_ER_M_CS=MT_ER_M_ITM
        0000000A 000C MT_ER_V_CS=MT_ER_V_ITM
        00008000 000C MT_ER_M_COR=MT_ER_M_CRC
        0000000F 000C MT_ER_V_COR=MT_ER_V_CRC
                 000C
                 000C $DEF    MT_MR           .BLKL   1       ;MAINTENANCE REGISTER
                 0010 $DEF    MT_AS           .BLKL   1       ;ATTENTION SUMMARY REGISTER
                 0014 $DEF    MT_FC           .BLKL   1       ;FRAME COUNT REGISTER
                 0018 $DEF    MT_DT           .BLKL   1       ;DRIVE TYPE REGISTER
                 001C         _VIELD  MT_DT,0,<-              ; DRIVE TYPE REGISTER FIELD DEFINITIONS
                 001C                 <DTN,9>,-               ; DRIVE TYPE NUMBER
                 001C                 <,1>,-                  ; RESERVED BIT
                 001C                 <SPR,,M>,-              ; SLAVE PRESENT
                 001C                 <DRQ,,M>,-              ; DRIVE REQUEST REQUIRED (ALWAYS 0)
                 001C                 <7CH,,M>,-              ; 7-CHANNEL TAPE (ALWAYS 0)
                 001C                 <MOH,,M>,-              ; MOVING HEAD (ALWAYS 0)
                 001C                 <TAP,,M>,-              ; TAPE DRIVE (ALWAYS 1)
                 001C         >                               ;
                 001C $DEF    MT_CC           .BLKL   1       ;CHECK CHARACTER REGISTER
                 0020 $DEF    MT_SN           .BLKL   1       ;SERIAL NUMBER REGISTER
                 0024 $DEF    MT_TC           .BLKL   1       ;MAGTAPE CONTROL REGISTER
                 0028         _VIELD  MT_TC,0,<-              ; TAPE CONTROL REGISTER FIELD DEFFINITIONS
                 0028                 <SSEL,3>,-              ; SLAVE SELECT
                 0028                 <EPAR,1,M>,-            ; EVEN PARITY
                 0028                 <FSEL,4,M>,-            ; FORMAT SELECT
                 0028                 <DEN,3,M>,-             ; DENSITY
                 0028                 <,1>,-                  ; RESERVED BIT
                 0028                 <EABO,,M>,-             ; ENABLE ABORT ON TRANSFER ERROR
                 0028                 <TCW,,M>,-              ; TAPE CONTROL WRITE
                 0028                 <FCS,,M>,-              ; FRAME COUNT STATUS
                 0028                 <ACCL,,M>,-             ; ACCELERATOR
                 0028         >                               ;
                 0028
                 0028         $DEFEND MT
                 0000
                 0000 ;
                 0000 ; DEFINE DEVICE DEPENDENT UNIT CONTROL BLOCK OFFSETS
                 0000 ;
                 0000
                 0000         $DEFINI UCB
                 0000
                 0000         $VIELD  UCB,0,<-                ;DEVICE DEPENDENT STATUS BITS
                 0000                 <MT_REWIND,,M>,-        ; REWIND IN PROGRESS
                 0000                 <MT_PRVMOL,,M>,-        ; PREVIOUS MEDIUM ONLINE STATE
                 0000                 <MT_PWRFL,,M>,-         ; Currently in Powerfail recovery
```

```
           0000           <MT_CNCLP,,M>,-                 ; Cancel Pending(looked at in Powerfail)
           0000           >                               ;
           0000
000000B4   0000      .=UCB$K_LCL_TAPE_LENGTH
           00B4
           00B4      $DEF    UCB$L_MT_SR     .BLKL   1    ;SAVED MBA STATUS REGISTER
           00B8      $DEF    UCB$W_MT_DS     .BLKW   1    ;SAVED DRIVE STATUS REGISTER
           00BA      $DEF    UCB$W_MT_ER     .BLKW   1    ;SAVED DRIVE ERROR REGISTER
           00BC      $DEF    UCB$W_MT_FC     .BLKW   1    ;SAVED DRIVE FRAME COUNT REGISTER
           00BE      $DEF    UCB$W_MT_SPACNT .BLKW   1    ;CURRENT SPACING COUNT
           00C0      $DEF    UCB$W_MT_CS1    .BLKW   1    ;SAVED DRIVE CONTROL REGISTER
           00C2      $DEF    UCB$W_MT_TC_SAV .BLKW   1    ;SAVED TAPE CONTROL REGISTER
           00C4      $DEF    UCB$W_MT_FORCNT .BLKW   1    ;FORWARD SPACE COUNT DURING RETRY
           00C6      $DEF    UCB$W_MT_TC     .BLKW   1    ;TAPE CONTROL REGISTER CONTENTS
           00C8      $DEF    UCB$L_MT_PREVTM .BLKL   1    ; Position of previous TAPEMARK, used
           00CC                                           ;   in forward SKIPFILE and SPACEFILE
           00CC                                           ;   operations in detecting consecutive
           00CC                                           ;   TAPEMARKs
           00CC      $DEF    UCB$L_MT_ORGPOS .BLKL   1    ; Here store value of UCB$L_RECORD
           00D0                                           ;   at STARTIO time.
           00D0      $DEF    UCB$W_MT_CC_SAV .BLKW   1    ; Space to save controller register.
000000D4   00D2                              .BLKW   1    ; RESERVED for now.
000000D4   00D4      UCB$K_MT_LENGTH=.                    ; Length of MT UCB.
           00D4
           00D4              $DEFEND UCB
           0000
           0000      ;
           0000      ; MAXIMUM SPACING ON READ AND WRITECHECK ERRORS
           0000      ;
           0000
00000005   0000      ERR_SPACING=5                        ;FIVE RECORDS
           0000
           0000      ;
           0000      ; HARDWARE FUNCTION CODES
           0000      ;
           0000
00000000   0000      F_NOP=0*2                            ;NO OPERATION
00000000   0000      F_PACKACK=0*2                        ;PACK ACKNOWLEDGE
00000000   0000      F_SENSECHAR=0*2                      ;SENSE TAPE CHARACTERISTICS
00000000   0000      F_SETCHAR=0*2                        ;SET TAPE CHARACTERISTICS
00000002   0000      F_UNLOAD=1*2                         ;UNLOAD DRIVE
00000006   0000      F_REWIND=3*2                         ;REWIND
00000008   0000      F_DRVCLR=4*2                         ;DRIVE CLEAR
00000010   0000      F_READPRESET=8*2                     ;READ IN PRESET
00000014   0000      F_ERASE=10*2                         ;ERASE TAPE
00000016   0000      F_WRITEMARK=11*2                     ;WRITE TAPE MARK
00000018   0000      F_SPCFILFOR=12*2                     ;SPACE FILE FORWARD
0000001A   0000      F_SPCFILREV=13*2                     ;SPACE FILE REVERSE
00000018   0000      F_SPCRECFOR=12*2                     ;SPACE RECORD FORWARD
0000001A   0000      F_SPCRECREV=13*2                     ;SPACE RECORD REVERSE
00000028   0000      F_INTSPCFOR=20*2                     ;INTERNAL SPACE RECORD FORWARD
0000002E   0000      F_INTSPCREV=23*2                     ;INTERNAL SPACE RECORD REVERSE
00000028   0000      F_WRITECHECK=20*2                    ;WRITE CHECK DATA FORWARD
0000002E   0000      F_WRITECHECKR=23*2                   ;WRITE CHECK DATA REVERSE
00000030   0000      F_WRITE=24*2                         ;WRITE DATA FORWARD
00000030   0000      F_WRITEDATA=24*2                     ;WRITE DATA FORWARD
00000038   0000      F_READDATA=28*2                      ;READ DATA FORWARD
```

```
0000003E  0000  F_READDATAR=31*2                         ;READ DATA REVERSE
          0000
          0000  ;
          0000  ; MINIMUM RECORD SIZE
          0000  ;
          0000
0000000E  0000  MIN_RECORD=14                            ;FOURTEEN BYTES
          0000
          0000  ;
          0000  ; HARWARE DENSITY DEFINITIONS
          0000  ;
          0000
00000003  0000  NRZI=3                                   ;800 BPI
00000004  0000  PE=4                                     ;PHASE ENCODED
          0000
          0000  ;
          0000  ; ERROR COUNT THRESHOLD BEFORE ALTERNATE RECOVERY ATTEMPTED
          0000  ;
          0000
00000008  0000  THRESHOLD=8                              ;EIGHT RETRIES BEFORE ALTERNATE METHOD
          0000
          0000  ;
          0000  ; LOCAL DATA
          0000  ;
          0000  ; DRIVER PROLOGUE TABLE
          0000  ;
          0000
          0000          DPTAB   -                        ;DEFINE DRIVER PROLOGUE TABLE
          0000                  END=TM_END,-             ;END OF DRIVER
          0000                  FLAGS=DPT$M_SUBCNTRL,-   ;INDICATE SUBCONTROLLER
          0000                  ADAPTER=MBA,-            ;ADAPTER TYPE
          0000                  UCBSIZE=UCB$K_MT_LENGTH,- ;UCB SIZE
          0000                  NAME=TMDRIVER            ;DRIVER NAME
          0038          DPT_STORE INIT                   ;CONTROL BLOCK INIT VALUES
          0038          DPT_STORE DDB,DDB$L_ACPD,L,<^A\MTA\> ;DEFAULT ACP NAME
          003F          DPT_STORE UCB,UCB$B_FIPL,B,8     ;FORK IPL
          0043          DPT_STORE UCB,UCB$L_DEVCHAR,L,-  ;DEVICE CHARACTERISTICS
          0043                  <DEV$M_FOD-              ; FILES ORIENTED
          0043                  !DEV$M_DIR-              ; DIRECTORY STRUCTURED
          0043                  !DEV$M_AVL-              ; AVAILABLE
          0043                  !DEV$M_ELG-              ; ERROR LOGGING ENABLED
          0043                  !DEV$M_IDV-              ; INPUT DEVICE
          0043                  !DEV$M_ODV-              ; OUTPUT DEVICE
          0043                  !DEV$M_SDI-              ; SINGLE DIRECTORY DEVICE
          0043                  !DEV$M_SQD>              ; SEQUENTIAL DEVICE
          004A          DPT_STORE UCB,UCB$L_DEVCHAR2,L,-; DEVICE CHARACTERISTICS
          004A                  <DEV$M_NNM>              ; PREFIX NAME WITH "node$"
          0051          DPT_STORE UCB,UCB$B_DEVCLASS,B,DC$_TAPE ;DEVICE CLASS
          0055          DPT_STORE UCB,UCB$W_DEVBUFSIZ,W,2048 ;DEFAULT BUFFER SIZE
          005A          DPT_STORE UCB,UCB$L_DEVDEPEND,W,<^X3C0> ;DEFAULT TAPE PARAMETERS
          005F          DPT_STORE UCB,UCB$B_DIPL,B,21    ;DEVICE IPL
          0063          DPT_STORE UCB,UCB$B_ERTCNT,B,16  ;ERROR RETRY COUNT
          0067          DPT_STORE UCB,UCB$B_ERTMAX,B,16  ;MAX ERROR RETRY COUNT
          006B          DPT_STORE UCB,UCB$W_MT_TC,W,<^X3C0> ;DEFAULT TAPE PARAMETERS
          0070          DPT_STORE REINIT                 ;CONTROL BLOCK RE-INIT VALUES
          0070          DPT_STORE CRB,CRB$L_INTD+4,D,TM$INT ;INTERRUPT SERVICE ROUTINE ADDRESS
          0075          DPT_STORE CRB,CRB$L_INTD+VEC$L_UNITINIT,D,TM_TXXX_INIT ;UNIT INIT
```

```
         007A           DPT_STORE DDB,DDB$L_DDT,D,TM$DDT ;DDT ADDRESS
         007F           DPT_STORE END                   ;
00000001 0000           .MDELETE DPT_STORE
         0000
         0000     ;
         0000     ; DRIVER DISPATCH TABLE
         0000     ;
         0000
         0000           DDTAB   TM,-                    ;DRIVER DISPATCH TABLE
         0000                   TM_STARTIO,-            ;START I/O OPERATION
         0000                   TM_UNSOLNT,-            ;UNSOLICITED INTERRUPT
         0000                   TM_FUNCTABLE,-          ;FUNCTION DECISION TABLE
         0000                   TM_CANCELIO,-           ;CANCEL I/O ENTRY POINT
         0000                   TM_REGDUMP,-            ;REGISTER DUMP ROUTINE
         0000                   <<MT_TC+4+MBA$L_BCR+4+8>+<<3+5+1>*4>>,- ;DIAGNOSTIC BUFFER SIZE
         0000                   <<MT_TC+4+MBA$L_BCR+4+8>+<1*4>+<EMB$L_DV_REGSAV>> ;ERROR BUFFER SIZE
         0038
         0038     ;
         0038     ; DENSITY CODE TRANSLATION TABLE
         0038     ;
         0038     ;     DENSITY CODES ARE TRANSLATED BY TAKING THE FIVE BIT ENCODED DENSITY
         0038     ;     VALUE, MULTIPLYING BY FOUR TO FORM THE STARTING BIT NUMBER, AND THEN
         0038     ;     EXTRACTING THE APPROPRIATE DENSITY CODE FROM THE TRANSLATION TABLE.
         0038     ;
         0038
         0038     DENSITY:                              ;
33343333 0038           .LONG   ^X33343333              ;DENSITY CODES 0-7
33333333 003C           .LONG   ^X33333333              ;DENSITY CODES 8-15
33333333 0040           .LONG   ^X33333333              ;DENSITY CODES 16-23
33333333 0044           .LONG   ^X33333333              ;DENSITY CODES 24-31
         0048
         0048     ;
         0048     ; TXXX DRIVE TYPE DESCRIPTOR TABLE
         0048     ;
         0048
         0048     TM_DTDESC:                            ;
    0029 0048           .WORD   ^X29                    ;TE16 45 IPS
      01 004A           .BYTE   DT$_TE16                ;
00000003 004B     TM_DTDESCLEN=.-TM_DTDESC              ;LENGTH OF DRIVE TYPE DESCRIPTOR
    002A 004B           .WORD   ^X2A                    ;TU45 45 IPS
      02 004D           .BYTE   DT$_TU45                ;
    002C 004E           .WORD   ^X2C                    ;TU77 125 IPS
      03 0050           .BYTE   DT$_TU77                ;
    0000 0051           .WORD   0                       ;END OF TABLE
00000056 0053           .BLKB   TM_DTDESCLEN            ;SPARE DRIVE TYPE SLOT
         0056
         0056     ;
         0056     ; HARDWARE I/O FUNCTION CODE TABLE
         0056     ;
         0056
         0056     FTAB:                                 ;
         0056           GENF    F_NOP                   ;NO OPERATION
         0057           GENF    F_UNLOAD                ;UNLOAD VOLUME
         0058           GENF    F_SPCFILFOR             ;SPACE FILE FORWARD
         0059           GENF    F_REWIND                ;REWIND
         005A           GENF    F_DRVCLR                ;DRIVE CLEAR
         005B           GENF    F_SPCFILREV             ;SPACE FILE REVERSE
```

```
          005C      GENF    F_ERASE                 ;ERASE TAPE
          005D      GENF    F_SPCRECREV             ;SPACE RECORD REVERSE
          005E      GENF    F_PACKACK               ;PACK ACKNOWLEDGE
          005F      GENF    F_SPCRECFOR             ;SPACE RECORD FORWARD
          0060      GENF    F_WRITECHECK            ;WRITE CHECK FORWARD
          0061      GENF    F_WRITEDATA             ;WRITE DATA FORWARD
          0062      GENF    F_READDATA              ;READ DATA FORWARD
          0063      GENF    F_WRITECHECKR           ;WRITE CHECK REVERSE
          0064      GENF    F_WRITE                 ;WRITE DATA FORWARD
          0065      GENF    F_READDATAR             ;READ DATA REVERSE
          0066      GENF    F_READPRESET            ;READ IN PRESET
          0067      GENF    F_SETCHAR               ;SET TAPE CHARACTERISTICS
          0068      GENF    F_SENSECHAR             ;SENSE TAPE CHARACTERISTICS
          0069      GENF    F_WRITEMARK             ;WRITE TAPE MARK
          006A      GENF    F_INTSPCFOR             ;INTERNAL SPACE RECORD FORWARD
          006B      GENF    F_INTSPCREV             ;INTERNAL SPACE RECORD REVERSE
          006C
          006C  ;
          006C  ; FORMAT CODE TRANSLATION TABLE
          006C  ;
          006C  ;       FORMAT CODES ARE TRANSLATED BY TAKING THE FOUR BIT ENCODED FORMAT VALUE,
          006C  ;       MULTIPLYING BY FOUR TO FORM THE STARTING BIT NUMBER, AND THEN EXTRACTING
          006C  ;       THE APPROPRIATE FORMAT CODE FROM THE TRANSLATION TABLE.
          006C  ;
          006C
          006C  FORMAT:                                 ;
CCCCCCCC  006C          .LONG   ^XCCCCCCCC              ;FORMAT CODES 0-7
CEDCCCCC  0070          .LONG   ^XCEDCCCCC              ;FORMAT CODES 8-15
          0074
          0074  ;
          0074  ; FUNCTION TIME OUT TABLE
          0074  ;
          0074
          0074  TIME_OUT:                               ;
00000000  0074          .LONG   0                       ;NO OPERATION
00000006  0078          .LONG   6                       ;UNLOAD VOLUME
000002D0  007C          .LONG   60*12                   ; Space File Forward
000001A4  0080          .LONG   60*7                    ;REWIND
00000000  0084          .LONG   0                       ;DRIVE CLEAR
000002D0  0088          .LONG   60*12                   ; Space File Reverse
00000006  008C          .LONG   6                       ;ERASE TAPE
000002D0  0090          .LONG   60*12                   ; Space Record Reverse
00000000  0094          .LONG   0                       ;PACK ACKNOWLEDGE
000002D0  0098          .LONG   60*12                   ; Space Record Forward
0000000C  009C          .LONG   12                      ;WRITE CHECK DATA FORWARD
0000000C  00A0          .LONG   12                      ;WRITE DATA FORWARD
0000000C  00A4          .LONG   12                      ;READ DATA FORWARD
0000000C  00A8          .LONG   12                      ;WRITE CHECK DATA REVERSE
0000000C  00AC          .LONG   12                      ;WRITE DATA FORWARD
0000000C  00B0          .LONG   12                      ;READ DATA REVERSE
000001A4  00B4          .LONG   60*7                    ;READ IN PRESET
00000000  00B8          .LONG   0                       ;SET TAPE CHARACTERISTICS
00000000  00BC          .LONG   0                       ;SENSE TAPE CHARACTERISITCS
00000006  00C0          .LONG   6                       ;WRITE TAPE MARK
0000000C  00C4          .LONG   12                      ;INTERNAL SPACE RECORD FORWARD
0000000C  00C8          .LONG   12                      ;INTERNAL SPACE RECORD REVERSE
          00CC
```

```
00CC    ;
00CC    ; DON'T CARE ERROR MASK TABLE
00CC    ;
00CC    ; THIS TABLE CONTAINS A MASK OF THE ERROR BITS THAT ARE TO BE IGNORED FOR EACH
00CC    ; FUNCTION WHEN EXAMINING THE DRIVE ERROR REGISTER.
00CC    ;
00CC
00CC    XTAB:
00CC            MASK    <FMT,DPAR,INC,PEF,NSG,FCE,CS,DTE,OPI,COR> ;NO-OP
00CE            MASK    <FMT,DPAR,INC,PEF,NSG,FCE,CS,DTE,OPI,COR> ;UNLOAD
00D0            MASK    <FMT,DPAR,INC,PEF,NSG,CS,DTE,COR> ;SPACE FILE FORWARD
00D2            MASK    <FMT,DPAR,INC,PEF,NSG,FCE,CS,DTE,OPI,COR> ;REWIND
00D4            MASK    <FMT,DPAR,INC,PEF,NSG,FCE,CS,NEF,DTE,OPI,COR> ;DRIVE CLEAR
00D6            MASK    <FMT,DPAR,INC,PEF,NSG,CS,DTE,COR> ;SPACE FILE REVERSE
00D8            MASK    <FMT,DPAR,INC,PEF,NSG,FCE,DTE,OPI,COR> ;ERASE
00DA            MASK    <FMT,DPAR,INC,PEF,NSG,CS,DTE,COR> ;SPACE RECORD REVERSE
00DC            MASK    <FMT,DPAR,INC,PEF,NSG,FCE,CS,DTE,OPI,COR> ;PACK ACKNOWLEDGE
00DE            MASK    <FMT,DPAR,INC,PEF,NSG,CS,DTE,COR> ;SPACE RECORD FORWARD
00E0            MASK    <DPAR>                  ;WRITE CHECK FORWARD
00E2            MASK    <>                      ;WRITE DATA FORWARD
00E4            MASK    <DPAR,FCE>              ;READ DATA FORWARD
00E6            MASK    <DPAR>                  ;WRITE CHECK REVERSE
00E8            MASK    <>                      ;WRITE DATA FORWARD
00EA            MASK    <DPAR,FCE>              ;READ DATA REVERSE
00EC            MASK    <FMT,DPAR,INC,PEF,NSG,FCE,CS,DTE,OPI,COR> ;READIN PRESET
00EE            MASK    <FMT,DPAR,INC,PEF,NSG,FCE,CS,DTE,OPI,COR> ;SET CHARACTERISTICS
00F0            MASK    <FMT,DPAR,INC,PEF,NSG,FCE,CS,DTE,OPI,COR> ;SENSE CHARACTERISTICS
00F2            MASK    <FMT,DPAR,FCE,DTE,COR>  ;WRITE TAPE MARK
00F4            MASK    <DPAR,INC,PEF,FCE,CS,DTE,COR> ;INTERNAL SPACE RECORD FORWARD
00F6            MASK    <DPAR,INC,PEF,FCE,CS,DTE,COR> ;INTERNAL SPACE RECORD REVERSE
```

```
00F8            .SBTTL  TE16/TU77 FUNCTION DECISION TABLE
00F8    ;+
00F8    ; TE16/TU77 FUNCTION DECISION TABLE
00F8    ;-
00F8
00F8    TM_FUNCTABLE:                                   ;FUNCTION DECISION TABLE
00F8            FUNCTAB ,-                              ;LEGAL FUNCTIONS
00F8                    <NOP,-                          ;NO OPERATION
00F8                    UNLOAD,-                        ;UNLOAD VOLUME
00F8                    SPACERECORD,-                   ;SPACE RECORDS
00F8                    RECAL,-                         ;RECALIBRATE (REWIND)
00F8                    DRVCLR,-                        ;DRIVE CLEAR
00F8                    READPRESET,-                    ;READ IN PRESET
00F8                    PACKACK,-                       ;PACK ACKNOWLEDGE
00F8                    ERASETAPE,-                     ;ERASE TAPE
00F8                    SENSECHAR,-                     ;SENSE TAPE CHARACTERISTICS
00F8                    SETCHAR,-                       ;SET CHARACTERISTICS
00F8                    SPACEFILE,-                     ;SPACE FILE
00F8                    WRITECHECK,-                    ;WRITE CHECK FORWARD
00F8                    WRITEPBLK,-                     ;WRITE PHYSICAL BLOCK
00F8                    READPBLK,-                      ;READ PHYSICAL BLOCK
00F8                    WRITEMARK,-                     ;WRITE TAPE MARK
00F8                    AVAILABLE,-                     ;AVAILABLE (REWIND/NOWAIT CLEAR VALID)
00F8                    READLBLK,-                      ;READ LOGICAL BLOCK
00F8                    WRITELBLK,-                     ;WRITE LOGICAL BLOCK
00F8                    SENSEMODE,-                     ;SENSE TAPE MODE
00F8                    SETMODE,-                       ;SET MODE
00F8                    REWIND,-                        ;REWIND
00F8                    REWINDOFF,-                     ;REWIND AND SET OFFLINE
00F8                    SKIPRECORD,-                    ;SKIP RECORDS
00F8                    SKIPFILE,-                      ;SKIP FILES
00F8                    WRITEOF,-                       ;WRITE END OF FILE
00F8                    READVBLK,-                      ;READ VIRTUAL BLOCK
00F8                    WRITEVBLK,-                     ;WRITE VIRTUAL BLOCK
00F8                    ACCESS,-                        ;ACCESS FILE AND/OR FIND DIRECTORY ENTRY
00F8                    ACPCONTROL,-                    ;ACP CONTROL FUNCTION
00F8                    CREATE,-                        ;CREATE FILE AND/OR CREATE DIRECTORY ENTRY
00F8                    DEACCESS,-                      ;DEACCESS FILE
00F8                    DELETE,-                        ;DELETE FILE AND/OR DIRECTORY ENTRY
00F8                    MODIFY,-                        ;MODIFY FILE ATTRIBUTES
00F8                    MOUNT>                          ;MOUNT VOLUME
0100            FUNCTAB ,-                              ;BUFFERED I/O FUNCTIONS
0100                    <NOP,-                          ;NO OPERATION
0100                    UNLOAD,-                        ;UNLOAD VOLUME
0100                    SPACERECORD,-                   ;SPACE RECORDS
0100                    RECAL,-                         ;RECALIBRATE (REWIND)
0100                    DRVCLR,-                        ;DRIVE CLEAR
0100                    READPRESET,-                    ;READ IN PRESET
0100                    PACKACK,-                       ;PACK ACKNOWLEDGE
0100                    ERASETAPE,-                     ;ERASE TAPE
0100                    SENSECHAR,-                     ;SENSE CHARACTERISTICS
0100                    SETCHAR,-                       ;SET CHARACTERISTICS
0100                    SPACEFILE,-                     ;SPACE FILES
0100                    WRITEMARK,-                     ;WRITE TAPE MARK
0100                    SENSEMODE,-                     ;SENSE MODE
0100                    SETMODE,-                       ;SET MODE
0100                    REWIND,-                        ;REWIND
```

```
0100                    REWINDOFF,-             ;REWIND AND UNLOAD
0100                    SKIPRECORD,-            ;SKIP RECORDS
0100                    SKIPFILE,-              ;SKIP FILES
0100                    WRITEOF,-               ;WRITE END OF FILE
0100                    ACCESS,-                ;ACCESS FILE AND/OR FIND DIRECTORY ENTRY
0100                    ACPCONTROL,-            ;ACP CONTROL FUNCTION
0100                    CREATE,-                ;CREATE FILE AND/OR CREATE DIRECTORY ENTRY
0100                    DEACCESS,-              ;DEACCESS FILE
0100                    DELETE,-                ;DELETE FILE AND/OR DIRECTORY ENTRY
0100                    MODIFY,-                ;MODIFY FILE ATTRIBUTES
0100                    MOUNT>                  ;MOUNT VOLUME
0108            FUNCTAB +ACP$READBLK,-          ;READ FUNCTIONS
0108                    <READLBLK,-             ;READ LOGICAL BLOCK FORWARD
0108                    READPBLK,-              ;READ PHYSICAL BLOCK FORWARD
0108                    READVBLK>               ;READ VIRTUAL BLOCK
0114            FUNCTAB +ACP$WRITEBLK,-         ;WRITE FUNCTIONS
0114                    <WRITECHECK,-           ;WRITE CHECK FORWARD
0114                    WRITELBLK,-             ;WRITE LOGICAL BLOCK
0114                    WRITEPBLK,-             ;WRITE PHYSICAL BLOCK
0114                    WRITEVBLK>              ;WRITE VIRTUAL BLOCK
0120            FUNCTAB +ACP$ACCESS,<ACCESS,CREATE> ;ACCESS AND CREATE FILE OR DIRECTORY
012C            FUNCTAB +ACP$DEACCESS,<DEACCESS> ;DEACCESS FILE
0138            FUNCTAB +ACP$MODIFY,-           ;
0138                    <ACPCONTROL,-           ;ACP CONTROL FUNCTION
0138                    DELETE,-                ;DELETE FILE OR DIRECTORY ENTRY
0138                    MODIFY>                 ;MODIFY FILE ATTRIBUTES
0144            FUNCTAB +ACP$MOUNT,<MOUNT>      ;MOUNT VOLUME
0150            FUNCTAB +MT$CHECK_ACCESS,-      ;MAGTAPE CHECK ACCESS FUNCTIONS
0150                    <ERASETAPE,-            ;ERASE TAPE
0150                    WRITEMARK,-             ;WRITE TAPE MARK
0150                    WRITEOF>                ;WRITE END OF FILE
015C            FUNCTAB +EXE$ZEROPARM,-         ;ZERO PARAMETER FUNCTIONS
015C                    <NOP,-                  ;NO OPERATION
015C                    UNLOAD,-                ;UNLOAD VOLUME
015C                    RECAL,-                 ;RECALIBRATE (REWIND)
015C                    REWIND,-                ;REWIND
015C                    REWINDOFF,-             ;REWIND AND SET OFFLINE
015C                    DRVCLR,-                ;DRIVE CLEAR
015C                    READPRESET,-            ;READ IN PRESET
015C                    PACKACK,-               ;PACK ACKNOWLEDGE
015C                    ERASETAPE,-             ;ERASE TAPE
015C                    SENSECHAR,-             ;SENSE TAPE CHARACTERISTICS
015C                    SENSEMODE,-             ;SENSE TAPE MODE
015C                    WRITEMARK,-             ;WRITE TAPE MARK
015C                    AVAILABLE,-             ;AVAILABLE (REWIND/NOWAIT CLEAR VALID)
015C                    WRITEOF>                ;WRITE END OF FILE
0168            FUNCTAB +EXE$ONEPARM,-          ;ONE PARAMETER FUNCTIONS
0168                    <SPACERECORD,-          ;SPACE RECORDS
0168                    SPACEFILE,-             ;SPACE FILES
0168                    SKIPRECORD,-            ;SKIP RECORDS
0168                    SKIPFILE>               ;SKIP FILES
0174            FUNCTAB +EXE$SETMODE,-          ;SET TAPE CHARACTERISTICS
0174                    <SETCHAR,-              ;
0174                    SETMODE>                ;
```

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

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

```
                              01FB          .SBTTL  START I/O OPERATION
                              01FB  ;+
                              01FB  ; TM_STARTIO - START I/O OPERATION ON DEVICE UNIT
                              01FB  ;
                              01FB  ; THIS ENTRY POINT IS ENTERED TO START AN I/O OPERATION ON A DEVICE UNIT.
                              01FB  ;
                              01FB  ; INPUTS:
                              01FB  ;
                              01FB  ;       R3 = ADDRESS OF I/O PACKET.
                              01FB  ;       R5 = UCB ADDRESS OF DEVICE UNIT.
                              01FB  ;
                              01FB  ; OUTPUTS:
                              01FB  ;
                              01FB  ;       FUNCTION DEPENDENT PARAMETERS ARE STORED IN THE DEVICE UCB, THE ERROR
                              01FB  ;       RETRY COUNT IS RESET, AND THE FUNCTION IS EXECUTED. AT FUNCTION COMPLETION
                              01FB  ;       THE OPERATION IS TERMINATED THROUGH REQUEST COMPLETE.
                              01FB  ;-
                              01FB
                              01FB  TM_STARTIO:                                     ;START I/O OPERATION
            00B0 C5     D0    01FB          MOVL    UCB$L_RECORD(R5),-              ; Remember tape position at STARTIO.
            00CC C5           01FF                  UCB$L_MT_ORGPOS(R5)
  0080 C5   0081 C5     90    0202          MOVB    UCB$B_ERTMAX(R5),UCB$B_ERTCNT(R5) ;INITIALIZE ERROR RETRY COUNT
   009A C5     20 A3    B0    0209          MOVW    IRP$W_FUNC(R3),UCB$W_FUNC(R5) ;SAVE FUNCTION CODE AND MODIFIERS
    00BE C5     01      AE    020F          MNEGW   #1,UCB$W_MT_SPACNT(R5)          ;SET DEFAULT SPACING COUNT
         50     38 A3   D0    0214          MOVL    IRP$L_MEDIA(R3),R0              ;GET PARAMETER LONGWORD
                              0218
                              0218  ;
                              0218  ; MOVE FUNCTION DEPENDENT PARAMETERS TO UCB
                              0218  ;
                              0218
         06     00      EF    0218          EXTZV   #IRP$V_FCODE,#IRP$S_FCODE,- ;EXTRACT I/O FUNCTION CODE
      51     20 A3            021B                  IRP$W_FUNC(R3),R1       ;
         51     02      D1    021E          CMPL    #IO$_SPACEFILE,R1               ;SPACE FILE FUNCTION?
                2B      13    0221          BEQL    10$                             ;IF EQL YES
         51     09      D1    0223          CMPL    #IO$_SPACERECORD,R1             ;SPACE RECORD FUNCTION?
                36      13    0226          BEQL    20$                             ;IF EQL YES
         51     1A      D1    0228          CMPL    #IO$_SETCHAR,R1                 ;SET CHARACTERISTICS FUNCTION?
                55      13    022B          BEQL    50$                             ;IF EQL YES
         51     11      D1    022D          CMPL    #IO$_AVAILABLE,R1               ;AVAILABLE function?
                69      13    0230          BEQL    75$                             ;IF EQL YES
         51     0D      D1    0232          CMPL    #IO$_READPBLK+1,R1              ;DISJOINT FUNCTION CODE?
                7B      1A    0235          BGTRU   100$                            ;IF GTRU NO
                              0237          CASE    R1,<-                           ;DISPATCH LOGICAL FUNCTIONS
                              0237                  70$,-                           ;REWIND AND SET OFFLINE
                              0237                  60$,-                           ;SET MODE
                              0237                  80$,-                           ;REWIND
                              0237                  10$,-                           ;SKIP FILE
                              0237                  20$,-                           ;SKIP RECORD
                              0237                  90$,-                           ;SENSE TAPE MODE
                              0237                  90$,-                           ;WRITE EOF
                              0237                  >,LIMIT=#IO$_REWINDOFF ;
         51     09      A2    0249          SUBW    #IO$_READPRESET-IO$_READPBLK-4,R1 ;CONVERT TO DENSE FUNCTION CODE
                72      11    024C          BRB     110$                            ;
                              024E
                              024E  ;
                              024E  ; SPACE FILE FUNCTION - SET SPACE COUNT AND PROPER FUNCTION
                              024E  ;
```

```
                        024E
          51   02   9A  024E   10$:    MOVZBL  #CDF_SPCFILFOR,R1          ;SET FOR SPACE FILE FORWARD
        00BE   C5   B6  0251           INCW    UCB$W_MT_SPACNT(R5)        ;SET DEFAULT SPACING COUNT TO LARGEST VALUE
               50   B5  0255           TSTW    R0                         ;SPACE FILE FORWARD?
               1A   14  0257           BGTR    40$                        ;IF GTR YES
          51   05   9A  0259           MOVZBL  #CDF_SPCFILREV,R1          ;SET FOR SPACE FILE REVERSE
               12   11  025C           BRB     30$                        ;
                        025E
                        025E   ;
                        025E   ; SPACE RECORD FUNCTION - SET SPACE COUNT AND PROPER FUNCTION
                        025E   ;
                        025E
          51   09   9A  025E   20$:    MOVZBL  #CDF_SPCRECFOR,R1          ;SET FOR SPACE RECORD FORWARD
     00BE C5   50   AE  0261           MNEGW   R0,UCB$W_MT_SPACNT(R5)     ;SET SPACING COUNT
               0B   19  0266           BLSS    40$                        ;IF LSS SPACE FORWARD FUNCTION
          51   07   9A  0268           MOVZBL  #CDF_SPCRECREV,R1          ;SET FOR SPACE RECORD REVERSE
     00BE C5   50   B0  026B           MOVW    R0,UCB$W_MT_SPACNT(R5)     ;SET SPACING COUNT
          50   50   AE  0270   30$:    MNEGW   R0,R0                      ;CONVERT TO POSITIVE COUNT
       7C A5   50   B0  0273   40$:    MOVW    R0,UCB$W_BOFF(R5)          ;SET SPACE COUNT
       7E A5   50   B0  0277           MOVW    R0,UCB$W_BCNT(R5)          ;SET SPACE COUNT
               43   12  027B           BNEQ    110$                       ;IF NEQ SPACING REQUIRED
          51   00   9A  027D           MOVZBL  #CDF_NOP,R1                ;SET FOR NO OPERATION
               3E   11  0280           BRB     110$                       ;
                        0282
                        0282   ;
                        0282   ; SET CHARACTERISTICS FUNCTION - STORE NEW TAPE CHARACTERISTICS
                        0282   ;
                        0282
    40 A5    38 A3  B0  0282   50$:    MOVW    IRP$L_MEDIA(R3),UCB$B_DEVCLASS(R5) ;SET NEW DEVICE CLASS AND TYPE
                        0287
                        0287   ;
                        0287   ; SET MODE FUNCTION - STORE NEW TAPE MODE
                        0287   ;
                        0287
    42 A5    3A A3  B0  0287   60$:    MOVW    IRP$L_MEDIA+2(R3),UCB$W_DEVBUFSIZ(R5) ;SET NEW DEFAULT BUFFER SIZE
    7C A5    3C A3  B0  028C           MOVW    IRP$L_MEDIA+4(R3),UCB$W_BOFF(R5)  ;SAVE NEW TAPE CONTROL PARAMETERS
          51   11   9A  0291           MOVZBL  #CDF_SETCHAR,R1            ;SET FUNCTION DISPATCH INDEX
               2A   11  0294           BRB     110$                       ;
                        0296
                        0296   ;
                        0296   ; LOGICAL REWIND AND SET TAPE OFFLINE - CONVERT TO UNLOAD FUNCTION
                        0296   ;
                        0296
          51   01   9A  0296   70$:    MOVZBL  #CDF_UNLOAD,R1             ;SET FOR UNLOAD FUNCTION
               25   11  0299           BRB     110$                       ;
                        029B
                        029B   ;
                        029B   ; AVAILABLE FUNCTION - Equivalent of REWIND(NOWAIT) and clear of UCB$M_VALID.
                        029B   ;
                        029B
                        029B   75$:
     00A4 8F        B0  029B           MOVW    #IO$_REWIND!IO$M_NOWAIT,-; Simulate a REWIND NOWAIT.
     009A C5            029F                   UCB$W_FUNC(R5)
     0800 8F        AA  02A2           BICW    #UCB$M_VALID,-             ; And clear valid bit.
       64 A5            02A6                   UCB$W_STS(R5)              ;  and fall thru to rewind logic.
                        02A8
                        02A8   ;
```

```
                        02A8    ; LOGICAL REWIND FUNCTION - CONVERT TO PHYSICAL FUNCTION
                        02A8    ;
                        02A8
        51    03   9A   02A8    80$:    MOVZBL  #CDF_REWIND,R1              ;SET FOR REWIND FUNCTION
              13   11   02AB            BRB     110$                        ;
                        02AD
                        02AD    ;
                        02AD    ; LOGICAL WRITE EOF OR SENSE MODE FUNCTION - CONVERT TO PHYSICAL FUNCTION
                        02AD    ;
                        02AD
        51    15   A2   02AD    90$:    SUBW    #IO$_SENSEMODE-IO$_READPBLK-6,R1 ;CONVERT TO PHYSICAL FUNCTION
              0E   11   02B0            BRB     110$                        ;
                        02B2
                        02B2    ;
                        02B2    ; DENSE FUNCTION CODE - CHECK FOR READ, WRITE, OR WRITECHECK FUNCTION
                        02B2    ;
                        02B2
        51    0A   D1   02B2    100$:   CMPL    #IO$_WRITECHECK,R1          ;DATA TRANSFER FUNCTION?
              09   1A   02B5            BGTRU   110$                        ;IF GTRU NO
  03 009A C5  06   E1   02B7            BBC     #IO$V_REVERSE,UCB$W_FUNC(R5),110$ ;IF CLR, NOT REVERSE FUNCTION
        51    03   A0   02BD            ADDW    #CDF_WRITECHECKR-CDF_WRITECHECK,R1 ;CONVERT TO REVERSE FUNCTION
                        02C0
                        02C0    ;
                        02C0    ; FINISH PREPROCESSING
                        02C0    ;
                        02C0
   0092 C5    51   90   02C0    110$:   MOVB    R1,UCB$B_FEX(R5)            ;SAVE FUNCTION DISPATCH INDEX
                        02C5
                        02C5    ;
                        02C5    ; CENTRAL FUNCTION DISPATCH
                        02C5    ;
                        02C5
                        02C5    FDISPATCH:                                  ;FUNCTION DISPATCH
      53    58 A5  D0   02C5            MOVL    UCB$L_IRP(R5),R3            ;RETRIEVE ADDRESS OF I/O PACKET
   0D 2A A3    08  E0   02C9            BBS     #IRP$V_PHYSIO,IRP$W_STS(R3),10$ ;IF SET, PHYSICAL I/O FUNCTION
   08 64 A5    0B  E0   02CE            BBS     #UCB$V_VALID,UCB$W_STS(R5),10$ ;IF SET, VOLUME SOFTWARE VALID
      50    0254 8F 3C  02D3            MOVZWL  #SS$_VOLINV,R0              ;SET VOLUME INVALID STATUS
            0B8C   31   02D8            BRW     RESETXFR                    ;
                        02DB
                        02DB    ;
                        02DB    ; UNIT IS SOFTWARE VALID OR FUNCTION IS PHYSICAL I/O
                        02DB    ;
                        02DB
                        02DB    10$:
      53    24 A5  D0   02DB            MOVL    UCB$L_CRB(R5),R3            ;GET ADDRESS OF CRB
                        02DF            ASSUME  IDB$L_CSR       EQ      0
      53    2C B3  D0   02DF            MOVL    @CRB$L_INTD+VEC$L_IDB(R3),R3 ;GET ADDRESS OF TM03 CSR
                        02E3    20$:    DSBINT                              ;DISABLE INTERRUPTS
   2E 64 A5    05  E0   02E9            BBS     #UCB$V_POWER,UCB$W_STS(R5),30$ ;IF SET, POWER FAILED
   29 68 A5    00  E1   02EE            BBC     #UCB$V_MT_REWIND,UCB$W_DEVSTS(R5),30$ ;IF CLR, NO REWIND IN PROGRESS
                        02F3            WFIKPCH TIMEOUT,#60*7               ;WAITFOR REWIND TO COMPLETE
   24 A3    00C6 C5 3C  0301            MOVZWL  UCB$W_MT_TC(R5),MT_TC(R3)   ;SELECT DRIVE
      52    04 A3  D0   0307            MOVL    MT_DS(R3),R2                ;READ DRIVE STATUS
   D4 52       0D  E0   030B            BBS     #MT_DS_V_PIP,R2,20$         ;IF SET, POSITIONING IN PROGRESS
   D0 52       01  E1   030F            BBC     #MT_DS_V_BOT,R2,20$         ;IF CLR, NOT AT BEGINNING OF TAPE
                        0313            IOFORK                              ;CREATE FORK PROCESS
                        0319            SAVIPL                              ;SAVE CURRENT IPL
```

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

```
                    0363        .SBTTL  WRITE TAPE MARK FUNCTION
                    0363   ;
                    0363   ; WRITE TAPE MARK FUNCTION
                    0363   ;
                    0363
                    0363   WRITEMARK:                                  ;WRITE TAPE MARK
                    0363   5$:     EXFUNC  10$,F_WRITEMARK             ;EXECUTE FUNCTION
   00B0 C5    D6    036B           INCL    UCB$L_RECORD(R5)            ;UPDATE TAPE POSITION
   50   01    3C    036F           MOVZWL  S^#SS$_NORMAL,R0            ;SET NORMAL COMPLETION STATUS
      06FE    31    0372           BRW     FUNCXT                      ;
                    0375
                    0375   ;
                    0375   ; FUNCTION ENDED IN AN ERROR
                    0375   ;
                    0375   ; THE ERROR COULD BE A NONFATAL CONTROLLER OR DRIVE ERROR. FATAL ERRORS TERMINATE
                    0375   ; THE FUNCTION IN THE FUNCTION EXECUTOR.
                    0375   ;
                    0375   ; FATAL CONTROLLER ERRORS ARE:
                    0375   ;
                    0375   ;       ERCONF  = ERROR CONFIRMATION.
                    0375   ;       ISTO    = INTERFACE SEQUENCE TIMEOUT.
                    0375   ;       PGE     = PROGRAMMING ERROR.
                    0375   ;       NED     = NONEXISTENT DRIVE.
                    0375   ;       RDTO    = READ DATA TIMEOUT.
                    0375   ;
                    0375   ; FATAL DRIVE ERRORS ARE:
                    0375   ;
                    0375   ;       ILF     = ILLEGAL FUNCTION.
                    0375   ;       ILR     = ILLEGAL REGISTER.
                    0375   ;       NEF     = NONEXECUTABLE FUNCTION.
                    0375   ;       RMR     = REGISTER MODIFY REFUSE.
                    0375   ;       UNS     = UNSAFE.
                    0375   ;
                    0375   ; IGNORED DRIVE ERRORS ARE:
                    0375   ;
                    0375   ;       FMT     = FORMAT.
                    0375   ;       DPAR    = DATA BUS PARITY.
                    0375   ;       FCE     = FRAME COUNT.
                    0375   ;       DTE     = DRIVE TIMING.
                    0375   ;       COR/CRC = CORRECTABLE OR CHECK CHARACTER ERROR.
                    0375   ;
                    0375   ; NOTE THAT IT IS ASSUMED THAT MASSBUS EXCEPTION (MBEXC) WILL OCCUR ONLY IN
                    0375   ; COMBINATION WITH ANOTHER DRIVE OR CONTROLLER ERROR.
                    0375   ;
                    0375
                    0375   10$:    TESTR   20$                         ;TEST REMAINING RETRIES
                    037A           REQSCHAN                            ;REQUEST SECONDARY CHANNEL
                    0380           EXFUNC  DOUBLE,F_INTSPCREV          ;SPACE RECORD REVERSE
                    0388           RELSCHAN                            ;RELEASE SECONDARY CHANNEL
08 009A C5   0C   E0    038E   20$:    BBS     #IO$V_INHEXTGAP,UCB$W_FUNC(R5),30$ ;IF SET, NO EXTENDED GAPS
                    0394           EXFUNC  DOUBLE,F_ERASE              ;WRITE EXTENDED INTER-RECORD GAP
          C5   11   039C   30$:    BRB     5$                          ;
```

```
                 039E        .SBTTL  ERASE TAPE FUNCTION
                 039E   ;
                 039E   ; ERASE TAPE FUNCTION
                 039E   ;
                 039E
                 039E   ERASE:                                          ;ERASE TAPE
                 039E           EXFUNC  10$                             ;EXECUTE FUNCTION
50    01    3C   03A3           MOVZWL  S^#SS$_NORMAL,R0                ;SET NORMAL COMPLETION
00BC  C5    B4   03A6           CLRW    UCB$W_MT_FC(R5)                 ;CLEAR FRAME COUNT
    06C6    31   03AA           BRW     FUNCXT                          ;
                 03AD
                 03AD   ;
                 03AD   ; FUNCTION ENDED IN AN ERROR
                 03AD   ;
                 03AD   ; THE ERROR COULD BE A NONFATAL CONTROLLER OR DRIVE ERROR. FATAL ERRORS TERMINATE
                 03AD   ; THE FUNCTION IN THE FUNCTION EXECUTOR.
                 03AD   ;
                 03AD   ; FATAL CONTROLLER ERRORS ARE:
                 03AD   ;
                 03AD   ;       ERCONF  = ERROR CONFIRMATION.
                 03AD   ;       ISTO    = INTERFACE SEQUENCE TIMEOUT.
                 03AD   ;       PGE     = PROGRAMMING ERROR.
                 03AD   ;       NED     = NONEXISTENT DRIVE.
                 03AD   ;       RDTO    = READ DATA TIMEOUT.
                 03AD   ;
                 03AD   ; FATAL DRIVE ERRORS ARE:
                 03AD   ;
                 03AD   ;       ILF     = ILLEGAL FUNCTION.
                 03AD   ;       ILR     = ILLEGAL REGISTER.
                 03AD   ;       NEF     = NONEXECUTABLE FUNCTION.
                 03AD   ;       RMR     = REGISTER MODIFY REFUSE.
                 03AD   ;       UNS     = UNSAFE.
                 03AD   ;
                 03AD   ; IGNORED DRIVE ERRORS ARE:
                 03AD   ;
                 03AD   ;       FMT     = FORMAT.
                 03AD   ;       DPAR    = DATA BUS PARITY.
                 03AD   ;       INC/VPE = INCORRECTABLE OR VERTICLE PARITY ERROR.
                 03AD   ;       PEF/LRC = FORMAT (PE) OR LONGITUDINAL PARITY ERROR.
                 03AD   ;       NSG     = NONSTANDARD GAP.
                 03AD   ;       FCE     = FRAME COUNT.
                 03AD   ;       DTE     = DRIVE TIMING.
                 03AD   ;       OPI     = OPERATION INCOMPLETE.
                 03AD   ;       COR/CRC = CORRECTABLE OR CHECK CHARACTER ERROR.
                 03AD   ;
                 03AD
00B0  C5    D5   03AD   10$:    TSTL    UCB$L_RECORD(R5)                ;ANY RECORDS ON TAPE?
      0A    12   03B1           BNEQ    20$                             ;IF NEQ YES
                 03B3           EXFUNC  DOUBLE,F_REWIND                 ;REWIND TAPE
      1C    11   03BB           BRB     30$                             ;
                 03BD   20$:    REQSCHAN                                ;REQUEST SECONDARY CHANNEL
                 03C3           EXFUNC  DOUBLE,F_INTSPCREV              ;BACK SPACE RECORD
                 03CB           EXFUNC  DOUBLE,F_INTSPCFOR              ;SPACE RECORD FORWARD
                 03D3           RELSCHAN                                ;RELEASE SECONDARY CHANNEL
    0615    31   03D9   30$:    BRW     RETRY                           ;
```

```
                        03DC          .SBTTL  HOUSEKEEPING FUNCTIONS
                        03DC  ;
                        03DC  ; HOUSEKEEPING FUNCTIONS INCLUDE:
                        03DC  ;
                        03DC  ;       PACK ACKNOWLEDGE,
                        03DC  ;       NO OPERATION,
                        03DC  ;       DRIVE CLEAR,
                        03DC  ;       SENSE TAPE CHARACTERISTICS, AND
                        03DC  ;       SET TAPE CHARACTERISTICS.
                        03DC  ;
                        03DC  ; IF THE FUNCTION ENDS IN A NONFATAL DRIVE ERROR IT IS RETRIED. FATAL ERRORS
                        03DC  ; TERMINATE THE FUNCTION IN THE FUNCTION EXECUTOR.
                        03DC  ;
                        03DC  ; FATAL DRIVE ERRORS ARE:
                        03DC  ;
                        03DC  ;       ILF     = ILLEGAL FUNCTION.
                        03DC  ;       ILR     = ILLEGAL REGISTER.
                        03DC  ;       NEF     = NONEXECUTABLE FUNCTION (EXCEPT FOR DRIVE CLEAR).
                        03DC  ;       RMR     = REGISTER MODIFY REFUSE.
                        03DC  ;       UNS     = UNSAFE.
                        03DC  ;
                        03DC  ; IGNORED DRIVE ERRORS ARE:
                        03DC  ;
                        03DC  ;       FMT     = FORMAT.
                        03DC  ;       DPAR    = DATA BUS PARITY.
                        03DC  ;       INC/VPE = INCORRECTABLE OR VERTICLE PARITY ERROR.
                        03DC  ;       PEF/LRC = FORMAT (PE) OR LONGITUDINAL PARITY ERROR.
                        03DC  ;       NSG     = NONSTANDARD GAP.
                        03DC  ;       FCE     = FRAME COUNT.
                        03DC  ;       CS/ITM  = CORRECTABLE SKEW OR INVALID TAPE MARK.
                        03DC  ;       DTE     = DRIVE TIMING.
                        03DC  ;       OPI     = OPERATION INCOMPLETE.
                        03DC  ;       COR/CRC = CORRECTABLE OR CHECK CHARACTER ERROR.
                        03DC  ;
                        03DC  ; ADDITIONAL IGNORED DRIVE ERRORS FOR DRIVE CLEAR ARE:
                        03DC  ;
                        03DC  ;       NEF     = NONEXECUTABLE FUNCTION.
                        03DC  ;
                        03DC  ; PACK ACKNOWLEDGE
                        03DC  ;
                        03DC
                        03DC          .ENABL  LSB
                        03DC  PACKACK:                                 ;PACK ACKNOWLEDGE
                        03DC          EXFUNC  RETRY                    ;EXECUTE FUNCTION
64 A5   0800 8F    A8   03E1          BISW    #UCB$M_VALID,UCB$W_STS(R5) ;SET VOLUME SOFTWARE VALID
             74    11   03E7          BRB     30$                      ;
                        03E9
                        03E9  ;
                        03E9  ; NO OPERATION, SENSE CHARACTERISTICS, AND DRIVE CLEAR.
                        03E9  ;
                        03E9
                        03E9  NOP:                                     ;NO OPERATION
                        03E9  SENSECHAR:                               ;SENSE CHARACTERISTICS
                        03E9  DRVCLR:                                  ;DRIVE CLEAR
                        03E9          EXFUNC  RETRY                    ;EXECUTE HOUSEKEEPING FUNCTION
             6D    11   03EE          BRB     30$                      ;
                        03F0
```

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

```
                        0467          .SBTTL  REWIND AND UNLOAD FUNCTIONS
                        0467    ;
                        0467    ; REWIND AND UNLOAD FUNCTIONS INCLUDE:
                        0467    ;
                        0467    ;         READIN PRESET,
                        0467    ;         REWIND, AND
                        0467    ;         UNLOAD.
                        0467    ;
                        0467    ; IF THE FUNCTION ENDS WITH A NONFATAL DRIVE ERRORS IT IS RETRIED. FATAL ERRORS
                        0467    ; TERMINATE THE FUNCTION IN THE FUNCTION EXECUTOR.
                        0467    ;
                        0467    ; FATAL DRIVE ERRORS ARE:
                        0467    ;
                        0467    ;         ILF     = ILLEGAL FUNCTION.
                        0467    ;         ILR     = ILLEGAL REGISTER.
                        0467    ;         NEF     = NONEXECUTABLE FUNCTION.
                        0467    ;         RMR     = REGISTER MODIFY REFUSE.
                        0467    ;         UNS     = UNSAFE.
                        0467    ;
                        0467    ; IGNORED DRIVE ERRORS ARE:
                        0467    ;
                        0467    ;         FMT     = FORMAT.
                        0467    ;         DPAR    = DATA BUS PARITY.
                        0467    ;         INC/VPE = INCORRECTABLE OR VERTICLE PARITY ERROR.
                        0467    ;         PEF/LRC = FORMAT (PE) OR LONGITUDINAL PARITY ERROR.
                        0467    ;         NSG     = NONSTANDARD GAP.
                        0467    ;         FCE     = FRAME COUNT.
                        0467    ;         CS/ITM  = CORRECTABLE SKEW OR INVALID TAPE MARK.
                        0467    ;         DTE     = DRIVE TIMING.
                        0467    ;         OPI     = OPERATION INCOMPLETE.
                        0467    ;         COR/CRC = CORRECTABLE OR CHECK CHARACTER ERROR.
                        0467    ;
                        0467
                        0467    READPRESET:                                     ;READ IN PRESET
          54 A5    B5   0467          TSTW    UCB$W_UNIT(R5)                  ;UNIT ZERO?
             03    13   046A          BEQL    REWIND                          ;IF EQL YES
       50    03    9A   046C          MOVZBL  #CDF_REWIND,R0                  ;CONVERT FUNCTION TO REWIND
                        046F
                        046F    ;
                        046F    ; UNLOAD AND REWIND FUNCTIONS
                        046F    ;
                        046F
                        046F    REWIND:                                         ;REWIND TO BEGINNING OF TAPE
                        046F    UNLOAD:                                         ;UNLOAD VOLUME
       00B0 C5     D4   046F          CLRL    UCB$L_RECORD(R5)                ; Since we will REWIND, the current
                        0473                                                    ;   position is of no interest.
                        0473                                                    ; This insures that UCB$L_RECORD will
                        0473                                                    ;   be correct if we do a REWIND nowait.
       00CC C5     D4   0473          CLRL    UCB$L_MT_ORGPOS(R5)             ; With REWIND the original position
                        0477                                                    ;   is of no interest.  If we should
                        0477                                                    ;   get a POWERFAIL this will reset us
                        0477                                                    ;   to BOT before retrying.
                        0477          EXFUNC  LOSTPOS                         ;EXECUTE FUNCTION
       00BC C5     B4   047C          CLRW    UCB$W_MT_FC(R5)                 ;CLEAR SAVED FRAME COUNT REGISTER
       50    01    3C   0480          MOVZWL  S^#SS$_NORMAL,R0                ;SET NORMAL COMPLETION STATUS
    06 52    0C    E0   0483          BBS     #MT_DS_V_MOL,R2,10$             ;IF SET, MEDIUM ONLINE
  64 A5  0800 8F   AA   0487          BICW    #UCB$M_VALID,UCB$W_STS(R5) ;SET VOLUME SOFTWARE INVALID
```

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

```
0490            .SBTTL  SPACING FUNCTIONS
0490    ;
0490    ; SPACING FUNCTIONS INCLUDE:
0490    ;
0490    ;       SPACE FILE FORWARD,
0490    ;       SPACE FILE REVERSE,
0490    ;       SPACE RECORD FORWARD, AND
0490    ;       SPACE RECORD REVERSE.
0490    ;
0490    ; ALL ARE IMPLEMENTED VIA THE SPACE RECORD FUNCTIONS.
0490    ;
0490    ; A SPACING FUNCTION CAN END WITH A NONFATAL DRIVE ERROR. FATAL ERRORS TERMINATE
0490    ; THE FUNCTION IN THE FUNCTION EXECUTOR.
0490    ;
0490    ; FATAL DRIVE ERRORS ARE:
0490    ;
0490    ;       ILF     = ILLEGAL FUNCTION.
0490    ;       ILR     = ILLEGAL REGISTER.
0490    ;       NEF     = NONEXECUTABLE FUNCTION (FORWARD FUNCTIONS).
0490    ;       RMR     = REGISTER MODIFY REFUSE.
0490    ;       UNS     = UNSAFE.
0490    ;
0490    ; NONFATAL DRIVE ERRORS ARE:
0490    ;
0490    ;       CPAR    = CONTROL BUS PARITY.
0490    ;       FCE     = FRAME COUNT.
0490    ;       NEF     = NONEXECUTABLE FUNCTION (REVERSE FUNCTIONS INTO BEGINNING OF TAPE).
0490    ;       OPI     = OPERATION INCOMPLETE (REVERSE FUNCTIONS INTO BEGINNING OF TAPE).
0490    ;
0490    ; IGNORED DRIVE ERRORS ARE:
0490    ;
0490    ;       FMT     = FORMAT.
0490    ;       DPAR    = DATA BUS PARITY.
0490    ;       INC/VPE = INCORRECTABLE OR VERTICLE PARITY ERROR.
0490    ;       PEF/LRC = FORMAT (PE) OR LONGITUDINAL PARITY ERROR.
0490    ;       NSG     = NONSTANDARD GAP.
0490    ;       CS/ITM  = CORRECTABLE SKEW OR INVALID TAPE MARK.
0490    ;       DTE     = DRIVE TIMING.
0490    ;       COR/CRC = CORRECTABLE OR CHECK CHARACTER ERROR.
0490    ;
0490    ; SPACE FILE FORWARD
0490    ;
0490    ;       SPACING FILES IS ACCOMPLISHED BY SPACING A VERY LARGE NUMBER OF RECORDS.
0490    ;       IF THE RECORD SPACING OPERATION COMPLETES WITHOUT ERROR, THEN THE RECORD
0490    ;       COUNT IS INCREASED BY 65,536 AND THE SPACING OPERATION IS CONTINUED.
0490    ;
0490    ;       SEVERAL SPECIAL CONDITIONS CAN ARISE DURING A SPACE FILE FORWARD:
0490    ;
0490    ;               1. A CONTROL BUS PARITY ERROR OCCURS.
0490    ;
0490    ;                       THE OPERATION IS MERELY RETRIED SINCE TAPE MOTION COULD
0490    ;                       NOT HAVE OCCURED.
0490    ;
0490    ;               2. AN END OF TAPE IS ENCOUNTERED WITHOUT AN END OF FILE.
0490    ;
0490    ;                       THE OPERATION IS CONTINUED SINCE END OF TAPE WHILE FILE
0490    ;                       SKIPPING DOES NOT TERMINATE THE OPERATION.
```

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

```
       00C8 C5   C3  0509        SUBL3   UCB$L_MT_PREVTM(R5),-
    50 00B0 C5       050D                UCB$L_RECORD(R5),R0    ; R0 = distance between last 2 TAPEMARKs
       00B0 C5   D0  0511        MOVL    UCB$L_RECORD(R5),-
       00C8 C5       0515                UCB$L_MT_PREVTM(R5)    ; Remember position of this TAPEMARK
            13   E1  0518        BBC     #DEV$V_MNT,-
       05 38 A5      051A                UCB$L_DEVCHAR(R5),20$  ; If NOT mounted, go test for EOV.
            18   E1  051D        BBC     #DEV$V_FOR,-
       04 38 A5      051F                UCB$L_DEVCHAR(R5),30$  ; If mounted NOT foreign (i.e. the ACP
                     0522                                       ;  is involved) branch around test EOV.
                     0522
                     0522  20$:  ; If here volume is either NOT mounted or mounted FOREIGN.
                     0522
            50   D7  0522        DECL    R0                     ; See if last 2 TAPEMARKs adjacent
            08   13  0524        BEQL    40$                    ; EQL implies adjacent TAPEMARKs
                     0526
                     0526  30$:  ; If here either ANSI mounted tape or EOV test failed.
                     0526
         7C A5   B5  0526        TSTW    UCB$W_BOFF(R5)         ; See if we have more TAPEMARKs to skip
            BC   12  0529        BNEQ    5$                     ;IF NEQ MORE TO GO
          00F9   31  052B        BRW     NORXIT                 ;
          00FB   31  052E  40$:  BRW     SETEOV                 ; Branch around to return ENDOFVOLUME status
                     0531
                     0531  ;
                     0531  ; SPACE FILE REVERSE
                     0531  ;
                     0531  ;       SPACING FILES IS ACCOMPLISHED BY SPACING A VERY LARGE NUMBER OF RECORDS.
                     0531  ;       IF THE RECORD SPACING OPERATION COMPLETES WITHOUT ERROR, THE THE RECORD
                     0531  ;       COUNT IS REDUCED BY 65,536 AND THE SPACING OPERATION IS CONTINUED.
                     0531  ;
                     0531  ;       SEVERAL SPECIAL CONDITIONS CAN ARISE DURING A SPACE FILE REVERSE:
                     0531  ;
                     0531  ;               1. A CONTROL BUS PARITY ERROR OCCURS.
                     0531  ;
                     0531  ;                       THE OPERATION IS MERELY RETRIED SINCE TAPE MOTION COULD
                     0531  ;                       NOT HAVE OCCURED.
                     0531  ;
                     0531  ;               2. AN END OF TAPE IS ENCOUNTERED WITHOUT AN END OF FILE.
                     0531  ;
                     0531  ;                       THE OPERATION IS CONTINUED SINCE END OF TAPE WHILE FILE
                     0531  ;                       SKIPPING DOES NOT TERMINATE THE OPERATION.
                     0531  ;
                     0531  ;               3. A BEGINNING OF TAPE IS ENCOUNTERED.
                     0531  ;
                     0531  ;                       THE OPERATION IS IMMEDIATELY TERMINATED.
                     0531  ;
                     0531  ;               4. AN END OF FILE IS ENCOUNTERED.
                     0531  ;
                     0531  ;                       THE FILE SKIP COUNT IS DECREMENTED AND IF NONZERO, THE
                     0531  ;                       OPERATION IS CONTINUED.
                     0531  ;
                     0531  ;       UNLESS A HARD ERROR IS ENCOUNTERED, NORMAL COMPLETION IS ALWAYS RETURNED
                     0531  ;       FOR A SKIP FILE REVERSE OPERATION.
                     0531  ;
                     0531  ;       AS RECORDS ARE SKIPPED BACKWARDS ON THE TAPE, THE CURRENT TAPE POSITION
                     0531  ;       IS MAINTAINED BY SUBTRACTING THE NUMBER RECORDS AND TAPE MARKS SKIPPED
                     0531  ;       OVER.
                     0531  ;
```

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

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

```
                                   05F1  ;                       THE OPERATION IS CONTINUED SINCE AN END OF TAPE WHILE
                                   05F1  ;                       SKIPPING RECORDS BACKWARDS DOES NOT TERMINATE THE OPER-
                                   05F1  ;                       ATION.
                                   05F1  ;
                                   05F1  ;               4. AN END OF FILE IS ENCOUNTERED.
                                   05F1  ;
                                   05F1  ;                       THE OPERATION IS TERMINATED WITH A FINAL STATUS OF END OF
                                   05F1  ;                       FILE.
                                   05F1  ;
                                   05F1  ;       UNLESS A HARD ERROR IS ENCOUNTERED, THE CURRENT TAPE POSITION IS MAINTAINED
                                   05F1  ;       BY SUBTRACTING THE TOTAL NUMBER OF RECORDS SKIPPED OVER.
                                   05F1  ;
                                   05F1
                                   05F1  SPCRECREV:                              ;SPACE RECORD REVERSE
                                   05F1          EXFUNC  10$,F_SPCRECREV         ;EXECUTE FUNCTION
                         03   11   05F9          BRB     20$                     ;
                                   05FB  10$:    CHECK_ERROR                     ;CHECK FOR FATAL OR RETRIABLE ERROR
                         50   D4   05FE  20$:    CLRL    R0                      ;CLEAR UPPER HALF OF REGISTER
      50   7C A5   00BC C5   A1   0600          ADDW3   UCB$W_MT_FC(R5),UCB$W_BOFF(R5),R0 ;CALCULATE RECORD SKIP COUNT
              00B0 C5   50   C2   0607          SUBL    R0,UCB$L_RECORD(R5)     ;UPDATE TAPE POSITION
           7C A5   00BC C5   AE   060C          MNEGW   UCB$W_MT_FC(R5),UCB$W_BOFF(R5) ;SET REMAINING RECORD SKIP COUNT
         00BE C5   00BC C5   B0   0612          MOVW    UCB$W_MT_FC(R5),UCB$W_MT_SPACNT(R5) ;RESET SPACING COUNT
              34 52      02   E0   0619          BBS     #MT_DS_V_TM,R2,SETEOF   ;IF SET, END OF FILE ENCOUNTERED
                         08   13   061D          BEQL    NORXIT                  ;IF EQL ALL RECORDS SPACED OVER
              2E 52      01   E0   061F          BBS     #MT_DS_V_BOT,R2,SETEOF  ;IF SET, BEGINNING OF TAPE ENCOUNTERED
              CA 52      0A   E0   0623          BBS     #MT_DS_V_EOT,R2,SPCRECREV ;IF SET, AT END OF TAPE
                                   0627
                                   0627  ;
                                   0627  ; NORMAL SPACING FUNCTION EXIT
                                   0627  ;
                                   0627
                                   0627          .ENABL  LSB
              50         01   3C   0627  NORXIT: MOVZWL  S^#SS$_NORMAL,R0        ;SET NORMAL COMPLETION STATUS
                         2A   11   062A          BRB     10$                     ;
                                   062C
                                   062C  ;
                                   062C  ; Backspace over 2nd TAPEMARK and set ENDOFVOLUME return status
                                   062C  ;
                                   062C
                                   062C  SETEOV:
         00BE C5         01   AE   062C          MNEGW   #1,UCB$W_MT_SPACNT(R5)  ; Don't space back more than 1 tape position
                                   0631          EXFUNC  2$,F_SPCFILREV          ; Backspace one tape position to
                                   0639                                          ;  position us between the two TAPEMARKs.
                                   0639  2$:     CHECK_ERROR                     ; Check for normal spacing errors.
              0E 52      02   E1   063C          BBC     #MT_DS_V_TM,R2,4$       ; If NOT at TAPEMARK, branch back to error.
              00B0 C5         D7   0640          DECL    UCB$L_RECORD(R5)        ; Update position in UCB.
                 7C A5        B6   0644          INCW    UCB$W_BOFF(R5)          ; Add one to total of files or
                                   0647                                          ;  records left to skip.
      50        09A0 8F       3C   0647          MOVZWL  #SS$_ENDOFVOLUME,R0     ; Set ENDOFVOLUME status
                         08   11   064C          BRB     10$                     ; Branch around to return
                    03AF      31   064E  4$:     BRW     DOUBLE                  ; If not at a TAPEMARK, then error.
                                   0651
                                   0651  ;
                                   0651  ; SET EOF OF FILE STATUS
                                   0651  ;
                                   0651
      50        0870 8F       3C   0651  SETEOF: MOVZWL  #SS$_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          .SBTTL  READ DATA FORWARD AND WRITECHECK DATA FORWARD FUNCTIONS
                              0661  ;
                              0661  ; READ DATA FORWARD AND WRITECHECK DATA FORWARD FUNCTIONS
                              0661  ;
                              0661
                              0661  WRITECHECK:                                     ;WRITE CHECK FORWARD
 009A C5   4000 8F       AA   0661          BICW    #IO$M_DATACHECK,UCB$W_FUNC(R5) ;CLEAR DATA CHECK REQUEST
                              0668  READDATA:                                       ;READ DATA FORWARD
                              0668          REQSCHAN                                ;REQUEST SECONDARY I/O CHANNEL
     50    0092 C5       9A   0665  5$:     MOVZBL  UCB$B_FEX(R5),R0                ;RETRIEVE FUNCTION INDEX
                              0673          EXFUNC  20$                             ;EXECUTE FUNCTION
                 05      E0   0678          BBS     #MT_DS_V_PES,-                  ; If at 1600 BPI, branch around test
         0B 00B8 C5           067A                  UCB$W_MT_DS(R5),10$             ;   for minimum length record.
                              067E  ; Before testing for short record on 800 bpi, test for Tape Mark.
          45 52  02      E0   067E          BBS     #MT_DS_V_TM,R2,40$              ; If SET, Tape Mark encountered
        00BC C5  0E      B1   0682          CMPW    #MIN_RECORD,UCB$W_MT_FC(R5) ;MINIMUM RECORD READ?
                 E5      1A   0687          BGTRU   5$                              ;IF GTRU NO
                 0E      E1   0689  10$:    BBC     #IO$V_DATACHECK,-
         38 009A C5           068B                  UCB$W_FUNC(R5),40$              ; If CLR, no data check.
                              068F          EXFUNC  DOUBLE,F_INTSPCREV              ;SPACE RECORD REVERSE
                              0697          EXFUNC  20$,F_WRITECHECK                ;WRITE CHECK DATA
                 26      11   069F          BRB     40$                             ;
                              06A1
                              06A1  ;
                              06A1  ; FUNCTION ENDED IN AN ERROR
                              06A1  ;
                              06A1  ; THE ERROR COULD BE A NONFATAL CONTROLLER OR DRIVE ERROR. FATAL ERRORS TERMINATE
                              06A1  ; THE FUNCTION IN THE FUNCTION EXECUTOR.
                              06A1  ;
                              06A1  ; FATAL CONTROLLER ERRORS ARE:
                              06A1  ;
                              06A1  ;       ERCONF  = ERROR CONFIRMATION.
                              06A1  ;       ISTO    = INTERFACE SEQUENCE TIMEOUT.
                              06A1  ;       PGE     = PROGRAMMING ERROR.
                              06A1  ;       NED     = NONEXISTENT DRIVE.
                              06A1  ;       RDTO    = READ DATA TIMEOUT.
                              06A1  ;
                              06A1  ; FATAL DRIVE ERRORS ARE:
                              06A1  ;
                              06A1  ;       ILF     = ILLEGAL FUNCTION.
                              06A1  ;       ILR     = ILLEGAL REGISTER.
                              06A1  ;       NEF     = NONEXECUTABLE FUNCTION.
                              06A1  ;       RMR     = REGISTER MODIFY REFUSE.
                              06A1  ;       UNS     = UNSAFE.
                              06A1  ;
                              06A1  ; IGNORED DRIVE ERRORS ARE:
                              06A1  ;
                              06A1  ;       DPAR    = DATA BUS PARITY.
                              06A1  ;
                              06A1  ; NOTE THAT IT IS ASSUMED THAT MASSBUS EXCEPTION (MBEXC) WILL OCCUR ONLY IN
                              06A1  ; COMBINATION WITH ANOTHER DRIVE OR CONTROLLER ERROR.
                              06A1  ;
                              06A1
 51   00024974 8F        D3   06A1  20$:    BITL    #MBA$M_SR_DLT!-                 ;DATA LATE OR,
                              06A8                  MBA$M_SR_INVMAP!-               ;INVALID MAP REGISTER OR,
                              06A8                  MBA$M_SR_MAPPE!-                ;MAP PARITY ERROR OR,
                              06A8                  MBA$M_SR_MCPE!-                 ;MASSBUS CONTROL PARITY ERROR OR,
```

```
                    06A8             MBA$M_SR_SPE!-          ;MBA SILO PARITY ERROR OR,
                    06A8             MBA$M_SR_MDPE!-         ;MASSBUS DATA PARITY ERROR OR,
                    06A8             MBA$M_SR_MXF!-          ;MISSED TRANSFER OR,
                    06A8             MBA$M_SR_RDS,R1         ;READ DATA SUBSTITUTE?
            62  12  06A8        BNEQ 80$                     ;IF NEQ YES
      07 52 02  E0  06AA        BBS  #MT_DS_V_TM,R2,30$      ;IF SET, TAPE MARK DETECTED
   51  0600 8F  B3  06AE        BITW #MBA$M_SR_WCKLWR!-      ;WRITE CHECK LOWER BYTE OR,
                    06B3             MBA$M_SR_WCKUPR,R1      ;WRITE CHECK UPPER BYTE?
            57  12  06B3        BNEQ 80$                     ;IF NEQ YES
   50  31D8 8F  B3  06B5 30$:   BITW #MT_ER_M_CPAR!-         ;CONTROL BUS PARITY ERROR OR,
                    06BA             MT_ER_M_DTE!-           ;DRIVE TIMING ERROR OR,
                    06BA             MT_ER_M_FMT!-           ;FORMAT ERROR OR,
                    06BA             MT_ER_M_INC!-           ;INCORRECTABLE ERROR (PE) OR,
                    06BA             MT_ER_M_LRC!-           ;LONGITUDINAL PARITY ERROR (NRZI) OR,
                    06BA             MT_ER_M_NSG!-           ;NONSTANDARD GAP OR,
                    06BA             MT_ER_M_OPI!-           ;OPERATION INCOMPLETE OR,
                    06BA             MT_ER_M_PEF!-           ;FORMAT ERROR (PE, OR,
                    06BA             MT_ER_M_VPE,R0          ;VERTICLE PARITY ERROR (NRZI)?
            43  12  06BA        BNEQ 70$                     ;IF NEQ YES
      07 52 05  E0  06BC        BBS  #MT_DS_V_PES,R2,40$     ;IF SET, PHASE ENCODED TAPE
   50  8400 8F  B3  06C0        BITW #MT_ER_M_CRC!MT_ER_M_ITM,R0 ;CRC OR INVALID TAPE MARK?
            45  12  06C5        BNEQ 80$                     ;IF NEQ YES
                    06C7
                    06C7 ;
                    06C7 ; FRAME COUNT ERROR
                    06C7 ;
                    06C7 ; IF THE RECORD CONTAINED MORE BYTES THAN THE SPECIFIED BUFFER, THEN A DATA OVERRUN
                    06C7 ; ERROR IS RETURNED. ELSE A CHECK IS MADE FOR END OF FILE AND IMPLICIT WRITECHECK.
                    06C7 ;
                    06C7
   50  0838 8F  3C  06C7 40$:   MOVZWL #SS$_DATAOVERUN,R0    ;SET DATA OVER RUN STATUS
7E A5  00BC C5  B1  06CC        CMPW UCB$W_MT_FC(R5),UCB$W_BCNT(R5) ;DATA OVER RUN?
            24  1A  06D2        BGTRU 60$                    ;IF GTR YES
   50  0870 8F  3C  06D4        MOVZWL #SS$_ENDOFFILE,R0     ;ASSUME END OF FILE
      1B 52 02  E0  06D9        BBS  #MT_DS_V_TM,R2,60$      ;IF SET, TAPE MARK ENCOUNTERED
      07 52 05  E0  06DD        BBS  #MT_DS_V_PES,R2,45$     ; If at 1600, branch around test of
                    06E1                                     ;  minimum length record.
   00BC C5  0E  B1  06E1        CMPW #MIN_RECORD,UCB$W_MT_FC(R5) ;MINIMUM RECORD READ?
            86  1A  06E6        BGTRU 5$                     ;IF GTRU NO
                    06E8 45$:
            0E  E1  06E8        BBC  #IO$V_DATACHECK,-
      07 009A C5    06EA             UCB$W_FUNC(R5),50$      ; If CLR, no data check.
   0093 C5  0C  91  06EE        CMPB #CDF_READDATA,UCB$B_CEX(R5) ;LAST FUNCTION READ DATA?
            94  13  06F3        BEQL 10$                     ;IF EQL YES
                    06F5 50$:
        50  01  3C  06F5        MOVZWL S^#SS$_NORMAL,R0      ;SET NORMAL COMPLETION STATUS
     00B0 C5    D6  06F8 60$:   INCL UCB$L_RECORD(R5)        ;INCREMENT RECORD POSITION
        0374    31  06FC        BRW  FUNCXT                  ;
                    06FF
                    06FF ;
                    06FF ; RECOVERABLE CONTROLLER OR DRIVE ERROR
                    06FF ;
                    06FF ; A CHECK IS MADE TO DETERMINE IF OPERATION INCOMPLETE IS THE ONLY ERROR AND THAT
                    06FF ; THE TRANSFERED BYTE COUNT WAS ZERO. IF THIS CHECK IS SATISFIED, THEN BLANK TAPE
                    06FF ; IS BEING READ AND THE FATAL ERROR EXIT IS TAKEN.
                    06FF ;
                    06FF ; A CHECK IS MADE TO DETERMINE IF ANY RETRIES REMAIN. IF THE NUMBER OF RETRIES IS
```

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

```
                            077D           .SBTTL  READ DATA REVERSE AND WRITECHECK DATA REVERSE FUNCTIONS
                            077D   ;
                            077D   ; READ DATA REVERSE AND WRITECHECK DATA REVERSE FUNCTIONS
                            077D   ;
                            077D
                            077D   WRITECHECKR:                                    ;WRITE CHECK REVERSE
 009A C5    4000 8F    AA   077D           BICW    #IO$M_DATACHECK,UCB$W_FUNC(R5) ;CLEAR DATA CHECK REQUEST
                            0784   READDATAR:                                      ;READ DATA REVERSE
                            0784           REQSCHAN                                ;REQUEST SECONDARY I/O CHANNEL
      50    0092 C5    9A   078A   5$:     MOVZBL  UCB$B_FEX(R5),R0                ;RETRIEVE FUNCTION INDEX
                            078F           EXFUNC  20$                             ;EXECUTE FUNCTION
     0F 52    05       E0   0794           BBS     #MT_DS_V_PES,R2,10$             ; If at 1600 BPI, branch around test
                            0798                                                   ;  for minimum length record.
                            0798   ; Before testing for short record on 800 bpi, test for Tape Mark or BOT.
     49 52    02       E0   0798           BBS     #MT_DS_V_TM,R2,40$              ; If SET, Tape Mark encountered
     45 52    01       E0   079C           BBS     #MT_DS_V_BOT,R2,40$             ; If SET, Beginning Of Tape
   00BC C5    0E       B1   07A0           CMPW    #MIN_RECORD,UCB$W_MT_FC(R5) ;MINIMUM RECORD READ?
              E3       1A   07A5           BGTRU   5$                              ;IF GTRU NO
              0E       E1   07A7   10$:    BBC     #IO$V_DATACHECK,-
      38 009A C5            07A9                   UCB$W_FUNC(R5),40$              ; If CLR, no data check.
                            07AD           EXFUNC  DOUBLE,F_INTSPCFOR              ;SPACE RECORD FORWARD
                            07B5           EXFUNC  20$,F_WRITECHECKR               ;WRITE CHECK DATA REVERSE
              26       11   07BD           BRB     40$                             ;
                            07BF
                            07BF   ;
                            07BF   ; FUNCTION ENDED IN AN ERROR
                            07BF   ;
                            07BF   ; THE ERROR COULD BE A NONFATAL CONTROLLER OR DRIVE ERROR. FATAL ERRORS TERMINATE
                            07BF   ; THE FUNCTION IN THE FUNCTION EXECUTOR.
                            07BF   ;
                            07BF   ; FATAL CONTROLLER ERRORS ARE:
                            07BF   ;
                            07BF   ;       ERCONF  = ERROR CONFIRMATION.
                            07BF   ;       ISTO    = INTERFACE SEQUENCE TIMEOUT.
                            07BF   ;       PGE     = PROGRAMMING ERROR.
                            07BF   ;       NED     = NONEXISTENT DRIVE.
                            07BF   ;       RDTO    = READ DATA TIMEOUT.
                            07BF   ;
                            07BF   ; FATAL DRIVE ERRORS ARE:
                            07BF   ;
                            07BF   ;       ILF     = ILLEGAL FUNCTION.
                            07BF   ;       ILR     = ILLEGAL REGISTER.
                            07BF   ;       RMR     = REGISTER MODIFY REFUSE.
                            07BF   ;       UNS     = UNSAFE.
                            07BF   ;
                            07BF   ; IGNORED DRIVE ERRORS ARE:
                            07BF   ;
                            07BF   ;       DPAR    = DATA BUS PARITY.
                            07BF   ;
                            07BF   ; NOTE THAT IT IS ASSUMED THAT MASSBUS EXCEPTION (MBEXC) WILL OCCUR ONLY IN
                            07BF   ; COMBINATION WITH ANOTHER DRIVE OR CONTROLLER ERROR.
                            07BF   ;
                            07BF
    51    00024974 8F  D3   07BF   20$:    BITL    #MBA$M_SR_DLT!-                 ;DATA LATE OR,
                            07C6                   MBA$M_SR_INVMAP!-               ;INVALID MAP REGISTER OR,
                            07C6                   MBA$M_SR_MAPPE!-                ;MAP PARITY ERROR OR,
                            07C6                   MBA$M_SR_MCPE!-                 ;MASSBUS CONTROL PARITY ERROR OR,
```

```
                    07C6             MBA$M_SR_SPE!-          ;MBA SILO PARITY ERROR OR,
                    07C6             MBA$M_SR_MDPE!-         ;MASSBUS DATA PARITY ERROR OR,
                    07C6             MBA$M_SR_MXF!-          ;MISSED TRANSFER OR,
                    07C6             MBA$M_SR_RDS,R1         ;READ DATA SUBSTITUTE OR,
            5C  12  07C6        BNEQ 70$                     ;IF NEQ YES
     07 52  02  E0  07C8        BBS  #MT_DS_V_TM,R2,30$      ;IF SET, TAPE MARK DETECTED
  51   0600 8F  B3  07CC        BITW #MBA$M_SR_WCKLWR!-      ;WRITE CHECK LOWER BYTE OR,
                    07D1             MBA$M_SR_WCKUPR,R1      ;WRITE CHECK UPPER BYTE?
            51  12  07D1        BNEQ 70$                     ;IF NEQ YES
  50   31D8 8F  B3  07D3 30$:   BITW #MT_ER_M_CPAR!-         ;CONTROL BUS PARITY ERROR OR,
                    07D8             MT_ER_M_DTE!-           ;DRIVE TIMING ERROR OR,
                    07D8             MT_ER_M_FMT!-           ;FORMAT ERROR OR,
                    07D8             MT_ER_M_INC!-           ;INCORRECTABLE ERROR (PE) OR,
                    07D8             MT_ER_M_LRC!-           ;LONGITUDINAL PARITY ERROR (NRZI) OR,
                    07D8             MT_ER_M_NSG!-           ;NONSTANDARD GAP OR,
                    07D8             MT_ER_M_OPI!-           ;OPERATION INCOMPLETE OR,
                    07D8             MT_ER_M_PEF!-           ;FORMAT ERROR (PE) OR,
                    07D8             MT_ER_M_VPE,R0          ;VERTICLE PARITY ERROR (NRZI)?
            4A  12  07D8        BNEQ 70$                     ;IF NEQ YES
     07 52  05  E0  07DA        BBS  #MT_DS_V_PES,R2,40$     ;IF SET, PHASE ENCODED TAPE
  50   8400 8F  B3  07DE        BITW #MT_ER_M_CRC!MT_ER_M_ITM,R0 ;CRC OR INVALID TAPE MARK?
            3F  12  07E3        BNEQ 70$                     ;IF NEQ YES
                    07E5
                    07E5 ;
                    07E5 ; FRAME COUNT OR NONEXECUTABLE FUNCTION
                    07E5 ;
                    07E5 ; IF THE RECORD CONTAINED MORE BYTES THAN THE SPECIFIED BUFFER, THEN A DATA OVERRUN
                    07E5 ; IS RETURNED. ELSE A CHECK IS MADE FOR END OF FILE, BEGINNING OF TAPE, AND
                    07E5 ; IMPLICIT WRITECHECK.
                    07E5 ;
                    07E5
  50   0838 8F  3C  07E5 40$:   MOVZWL #SS$_DATAOVERUN,R0    ;SET DATA OVER RUN STATUS
7E A5  00BC C5  B1  07EA        CMPW UCB$W_MT_FC(R5),UCB$W_BCNT(R5) ;DATA OVER RUN?
            2B  1A  07F0        BGTRU 60$                    ;IF GTR YES
  50   0870 8F  3C  07F2        MOVZWL #SS$_ENDOFFILE,R0     ;SET END OF FILE STATUS
     22 52  02  E0  07F7        BBS  #MT_DS_V_TM,R2,60$      ;IF SET, TAPE MARK ENCOUNTERED
     1E 52  01  E0  07FB        BBS  #MT_DS_V_BOT,R2,60$     ;IF SET, BEGINNING OF TAPE
     0A 52  05  E0  07FF        BBS  #MT_DS_V_PES,R2,45$     ; If at 1600, branch around test of
                    0803                                     ;  minimum length record.
  00BC C5   0E  B1  0803        CMPW #MIN_RECORD,UCB$W_MT_FC(R5) ;MINIMUM RECORD READ?
            03  1B  0808        BLEQU 45$
          FF7D  31  080A        BRW  5$                      ;IF GTRU NO
                    080D 45$:
            0E  E1  080D        BBC  #IO$V_DATACHECK,-
  07 009A C5        080F             UCB$W_FUNC(R5),50$      ; If CLR, no data check.
  0093 C5   0F  91  0813        CMPB #CDF_READDATAR,UCB$B_CEX(R5) ;LAST FUNCTION READ DATA?
            8D  13  0818        BEQL 10$                     ;IF EQL YES
                    081A 50$:
       50   01  3C  081A        MOVZWL S^#SS$_NORMAL,R0      ;SET NORMAL COMPLETION STATUS
     00B0 C5    D7  081D 60$:   DECL UCB$L_RECORD(R5)        ;DECREMENT TAPE POSITION
          024F  31  0821        BRW  FUNCXT                  ;
                    0824
                    0824 ;
                    0824 ; FUNCTION ENDED IN A RETRIABLE ERROR
                    0824 ;
                    0824
                    0824 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_CEX(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         .SBTTL  WRITE DATA FORWARD FUNCTION
                              08C8  ;
                              08C8  ; WRITE DATA FORWARD FUNCTION
                              08C8  ;
                              08C8
                              08C8  WRITEDATA:                                      ;WRITE DATA FORWARD
                              08C8         REQSCHAN                                 ;REQUEST SECONDARY CHANNEL
                              08CE  5$:    EXFUNC  20$,F_WRITEDATA                  ;EXECUTE FUNCTION
  10 009A C5    0E      E1    08D6         BBC     #IO$V_DATACHECK,UCB$W_FUNC(R5),10$ ;IF CLR, NO DATA CHECK
                              08DC         EXFUNC  DOUBLE,F_INTSPCREV               ;SPACE RECORD REVERSE
                              08E4         EXFUNC  50$,F_WRITECHECK                 ;WRITE CHECK DATA
  00BC C5    7E A5      B0    08EC  10$:   MOVW    UCB$W_BCNT(R5),UCB$W_MT_FC(R5) ;SET TRANSFER BYTE COUNT
          00B0 C5       D6    08F2         INCL    UCB$L_RECORD(R5)                 ;INCREMENT TAPE POSITION
          50   01       3C    08F6         MOVZWL  S^#SS$_NORMAL,R0                 ;SET NORMAL COMPLETION STATUS
             0177       31    08F9         BRW     FUNCXT                           ;
                              08FC
                              08FC  ;
                              08FC  ; WRITE FUNCTION ENDED IN AN ERROR
                              08FC  ;
                              08FC  ; THE ERROR COULD BE A NONFATAL CONTROLLER OR DRIVE ERROR. FATAL ERRORS TERMINATE
                              08FC  ; THE FUNCTION IN THE FUNCTION EXECUTOR.
                              08FC  ;
                              08FC  ; FATAL CONTROLLER ERRORS ARE:
                              08FC  ;
                              08FC  ;      ERCONF  = ERROR CONFIRMATION.
                              08FC  ;      ISTO    = INTERFACE SEQUENCE TIMEOUT.
                              08FC  ;      PGE     = PROGRAMMING ERROR.
                              08FC  ;      NED     = NONEXISTENT DRIVE.
                              08FC  ;      RDTO    = READ DATA TIMEOUT.
                              08FC  ;
                              08FC  ; FATAL DRIVE ERRORS ARE:
                              08FC  ;
                              08FC  ;      ILF     = ILLEGAL FUNCTION.
                              08FC  ;      ILR     = ILLEGAL REGISTER.
                              08FC  ;      NEF     = NONEXECUTABLE FUNCTION.
                              08FC  ;      RMR     = REGISTER MODIFY REFUSE.
                              08FC  ;      UNS     = UNSAFE.
                              08FC  ;
                              08FC  ; IGNORED DRIVE ERRORS ARE:
                              08FC  ;
                              08FC  ;      NONE.
                              08FC  ;
                              08FC
 009A C5   4000 8F      A8    08FC  20$:   BISW    #IO$M_DATACHECK,UCB$W_FUNC(R5) ;FORCE DATA CHECK
  00BC C5    7E A5      A0    0903         ADDW    UCB$W_BCNT(R5),UCB$W_MT_FC(R5) ;CALCULATE ACTUAL BYTES TRANSFERED
                              0909         TESTR   30$                              ;TEST REMAINING RETRIES
                              090E         EXFUNC  DOUBLE,F_INTSPCREV               ;SPACE RECORD REVERSE
  14 009A C5    0C      E0    0916  30$:   BBS     #IO$V_INHEXTGAP,UCB$W_FUNC(R5),40$ ;IF SET, NO EXTENDED GAP
                              091C         RELSCHAN                                 ;RELEASE SECONDARY CHANNEL
                              0922         EXFUNC  DOUBLE,F_ERASE                   ;ERASE TAPE
                              092A         REQSCHAN                                 ;REQUEST SECONDARY CHANNEL
               9C       11    0930  40$:   BRB     5$                               ;
                              0932
                              0932  ;
                              0932  ; WRITECHECK FUNCTION ENDED IN ERROR
                              0932  ;
                              0932  ; 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            .SBTTL  CHECK FOR FATAL OR RETRIABLE SPACING ERROR
                        094B      ;
                        094B      ; CHECK_ERROR - CHECK FOR FATAL OR RETRIABLE SPACING ERROR
                        094B      ;
                        094B      ; THIS ROUTINE IS CALLED FROM THE SPACE FILE FORWARD, SPACE RECORD FORWARD, SPACE
                        094B      ; FILE REVERSE, AND SPACE RECORD REVERSE FUNCTION ROUTINES TO TEST WHETHER A
                        094B      ; SPACING ERROR IS FATAL AND WHETHER ANY RETRIES REMAIN.
                        094B      ;
                        094B      ; INPUTS:
                        094B      ;
                        094B      ;       R0 = DRIVE ERROR REGISTER.
                        094B      ;       R1 = MBA STATUS REGISTER.
                        094B      ;       R2 = DRIVE STATUS REGISTER.
                        094B      ;
                        094B      ;       UCB$B_CEX(R5) = FUNCTION INDEX OF LAST FUNCTION EXECUTED.
                        094B      ;       UCB$B_ERTCNT(R5) = NUMBER OF ERROR RETRIES REMAINING.
                        094B      ;       UCB$W_BCNT(R5) = ORIGINAL SPACING COUNT.
                        094B      ;       UCB$W_BOFF(R5) = REMAINING SPACING COUNT.
                        094B      ;
                        094B      ; OUTPUTS:
                        094B      ;
                        094B      ;       IF ONLY A FRAME COUNT ERROR OCCURED, OR THE FUNCTION WAS A REVERSE DIRECTION
                        094B      ;       FUNCTION AND A CONTROL BUS PARITY ERROR DID NOT OCCUR, THEN AN IMMEDIATE
                        094B      ;       RETURN TO THE CALLER IS EXECUTED. ELSE THE REMAINING RETRY COUNT IS DE-
                        094B      ;       CREMENTED, AND IF THE RESULT IS ZERO, THEN THE FUNCTION IS TERMINATED VIA
                        094B      ;       THE DOUBLE ERROR EXIT. ELSE THE FUNCTION IS REDISPATCHED.
                        094B      ;
                        094B      
                        094B      CHECK_ERROR:                                  ;
              51 8ED0   094B            POPL    R1                            ;REMOVE RETURN ADDRESS FROM STACK
50    FDFF 8F     B3    094E            BITW    #^C<MT_ER_M_FCE>,R0           ;RETRIABLE OR FATAL ERROR?
           21     13    0953            BEQL    20$                           ;IF EQL NO
                        0955      
                        0955      ;
                        0955      ; RETRIABLE OR FATAL DRIVE ERROR
                        0955      ;
                        0955      ;       CPAR    = CONTROL BUS PARITY ERROR.
                        0955      ;       NEF     = NONEXECUTABLE FUNCTION.
                        0955      ;       OPI     = OPERATION INCOMPLETE.
                        0955      ;
                        0955      
   24 50   03     E0    0955            BBS     #MT_ER_V_CPAR,R0,40$          ;IF SET, CONTROL BUS PARITY ERROR
                        0959      
                        0959      ;
                        0959      ; ERROR WAS EITHER A NONEXECUTABLE FUNCTION OR OPERATION INCOMPLETE
                        0959      ;
                        0959      
 0093 C5   05     91    0959            CMPB    #CDF_SPCFILREV,UCB$B_CEX(R5)  ;SPACE FILE REVERSE?
           07     13    095E            BEQL    10$                           ;IF EQL YES
 0093 C5   07     91    0960            CMPB    #CDF_SPCRECREV,UCB$B_CEX(R5)  ;SPACE RECORD REVERSE?
           11     12    0965            BNEQ    30$                           ;IF NEQ NO
   0D 52   01     E1    0967      10$:  BBC     #MT_DS_V_BOT,R2,30$           ;IF CLR, NOT AT BEGINNING OF TAPE
                        096B      ; Here we have a KLUDGE because the TM03 leaves a zero in the Frame Count
                        096B      ;       register (thereby indicating that the skip succeeded) when a skip
                        096B      ;       record reverse operation begins at the BOT.  So if NEF is set,
                        096B      ;       then we make correct the value that we have for the Frame Count to
                        096B      ;       be the original value.  We know that we are in this state if NEF
```

```
                          096B  2012 ;         bit is set in MT_ER register.
                          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          .SBTTL  TEST FOR REMAINING RETRIES
                          0994     ;
                          0994     ; TESTR - TEST FOR REMAINING RETRIES
                          0994     ;
                          0994     ; THIS ROUTINE IS CALLED FROM THE READ DATA, READ DATA REVERSE, WRITECHECK,
                          0994     ; WRITECHECK REVERSE, WRITE DATA, AND WRITE TAPE MARK FUNCTION ROUTINES TO
                          0994     ; TEST FOR REMAINING RETRIES.
                          0994     ;
                          0994     ; INPUTS:
                          0994     ;
                          0994     ;       R0 = DRIVE ERROR REGISTER.
                          0994     ;       R1 = MBA STATUS REGISTER.
                          0994     ;       R2 = DRIVE STATUS REGISTER.
                          0994     ;
                          0994     ;       UCB$B_CEX(R5) = FUNCTION INDEX OF LAST FUNCTION EXECUTED.
                          0994     ;       UCB$B_ERTCNT(R5) = NUMBER OF ERROR RETRIES REMAINING.
                          0994     ;       UCB$L_RECORD(R5) = CURRENT TAPE POSITION BEFORE FUNCTION EXECUTION.
                          0994     ;       UCB$W_MT_DS(R5) = SAVED DRIVE STATUS REGISTER.
                          0994     ;       UCB$W_FUNC(R5) = ORIGINAL FUNCTION WORD.
                          0994     ;       UCB$W_MT_FC(R5) = NUMBER OF BYTES THAT WERE READ OR WRITTEN TO/FROM TAPE.
                          0994     ;
                          0994     ;       @(SP) = SIGNED BRANCH DISPLACEMENT TO CONDITIONAL EXIT POINT.
                          0994     ;
                          0994     ; OUTPUTS:
                          0994     ;
                          0994     ;       THE REMAINING RETRY COUNT IS DECREMENTED AND IF THE RESULT IS ZERO, THEN
                          0994     ;       THE FUNCTION IS TERMINATED WITH AN ERROR VIA THE FATAL ERROR EXIT AFTER
                          0994     ;       HAVING ADJUSTED THE CURRENT TAPE POSITION. ELSE THE CONDITIONAL BRANCH
                          0994     ;       EXIT IS TAKEN IF NO TAPE MOVEMENT OCCURED.
                          0994     ;
                          0994
                          0994     TESTR:                                          ;
     00000000'GF    16    0994          JSB     G^ERL$DEVICERR                  ;LOG DEVICE ERROR
         0080 C5    97    099A          DECB    UCB$B_ERTCNT(R5)                ;ANY RETRIES REMAINING?
              1E    13    099E          BEQL    20$                             ;IF EQL NO
              05    E0    09A0          BBS     #MT_DS_V_PES,-                  ; If at 1600 BPI, branch around test
      14 00B8 C5          09A2                  UCB$W_MT_DS(R5),10$             ;  for minimum length record.
     00BC C5  0E    B1    09A6          CMPW    #MIN_RECORD,UCB$W_MT_FC(R5) ;ANY TAPE MOVEMENT?
              0D    1B    09AB          BLEQU   10$                             ;IF LEQU YES
   07 00B8 C5  02   E0    09AD          BBS     #MT_DS_V_TM,UCB$W_MT_DS(R5),10$ ;IF SET, TAPE MARK DETECTED
      7E   00 BE    32    09B3          CVTWL   @(SP),-(SP)                     ;GET DISPLACEMENT VALUE
         6E   8E    C0    09B7          ADDL    (SP)+,(SP)                      ;CALCULATE BRANCH ADDRESS
         6E   02    C0    09BA     10$: ADDL    #2,(SP)                         ;
              05          09BD          RSB                                     ;
                          09BE
                          09BE     ;
                          09BE     ; ERROR RETRIES EXHAUSTED
                          09BE     ;
                          09BE     ; CHECK FOR TAPE MOVEMENT, ADJUST TAPE POSITION AS APPROPRIATE, AND TAKE FATAL
                          09BE     ; ERROR EXIT.
                          09BE     ;
                          09BE
              8E    D5    09BE     20$: TSTL    (SP)+                           ;REMOVE RETURN FROM STACK
   0D 00B8 C5  02   E0    09C0          BBS     #MT_DS_V_TM,UCB$W_MT_DS(R5),30$ ;IF SET, TAPE MARK DETECTED
              05    E0    09C6          BBS     #MT_DS_V_PES,-                  ; If at 1600 BPI, branch around test
      07 00B8 C5          09C8                  UCB$W_MT_DS(R5),30$             ;  for minimum length record.
     00BC C5  0E    B1    09CC          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                .SBTTL  TAPE POSITION LOST
                          09ED        ;
                          09ED        ; TAPE POSITION LOST
                          09ED        ;
                          09ED
                          09ED        LOSTPOS:
      46 A5    10    A8   09ED                BISW    #<MT$M_LOST@-16>,UCB$L_DEVDEPEND+2(R5) ;SET TAPE POSITION LOST
                          09F1
                          09F1        ;
                          09F1        ; TEST FOR RETRY
                          09F1        ;
                          09F1
                          09F1        RETRY:                                             ;TEST FOR RETRY
   00000000'GF         16 09F1                JSB     G^ERL$DEVICERR                     ;LOG DEVICE ERROR
        0080 C5        97 09F7                DECB    UCB$B_ERTCNT(R5)                   ;ANY RETRIES REMAINING?
             07        13 09FB                BEQL    FATALERR                           ;IF EQL NO
           F8C5        31 09FD                BRW     FDISPATCH                          ;
                          0A00
                          0A00        ;
                          0A00        ; DOUBLE ERROR WHILE TRYING TO REPOSITION TAPE - TAPE POSITION LOST
                          0A00        ;
                          0A00
                          0A00        DOUBLE:
      46 A5    10    A8   0A00                BISW    #<MT$M_LOST@-16>,UCB$L_DEVDEPEND+2(R5) ;SET LOST POSITION STATUS
                          0A04
                          0A04        ;
                          0A04        ; FATAL CONTROLLER/DRIVE ERROR, ERROR RETRY COUNT EXHAUSTED, ERROR RETRY
                          0A04        ; INHIBITED, OR FINAL OFFSET TRIED
                          0A04        ;
                          0A04
                          0A04        FATALERR:                                          ;FATAL ERROR - SET STATUS
        53     50    D0   0A04                MOVL    R0,R3                              ;COPY ERROR STATUS REGISTER
 50     01A4 8F      3C   0A07                MOVZWL  #SS$_MEDOFL,R0                     ;SET MEDIUM OFFLINE STATUS
   63 52       0C    E1   0A0C                BBC     #MT_DS_V_MOL,R2,FUNCXT             ;IF CLR, MEDIUM OFFLINE
   09 53       0B    E1   0A10                BBC     #MT_ER_V_NEF,R3,10$                ;IF CLR, EXECUTABLE FUNCTION
 50     025C 8F      3C   0A14                MOVZWL  #SS$_WRITLCK,R0                    ;SET WRITE LOCK ERROR STATUS
   56 52       0B    E0   0A19                BBS     #MT_DS_V_WRL,R2,FUNCXT             ;IF SET, DRIVE HARDWARE WRITE LOCKED
 50     023C 8F      3C   0A1D        10$:    MOVZWL  #SS$_UNSAFE,R0                     ;SET DRIVE UNSAFE STATUS
   4D 53       0E    E0   0A22                BBS     #MT_ER_V_UNS,R3,FUNCXT             ;IF SET, DRIVE UNSAFE
 50     02D4 8F      3C   0A26                MOVZWL  #SS$_OPINCOMPL,R0                  ;SET OPERATION INCOMPLETE STATUS
   44 53       0D    E0   0A2B                BBS     #MT_ER_V_OPI,R3,FUNCXT             ;IF SET, OPERATION INCOMPLETE
 50     00BC 8F      3C   0A2F                MOVZWL  #SS$_FORMAT,R0                     ;SET FORMAT ERROR STATUS
   3B 53       04    E0   0A34                BBS     #MT_ER_V_FMT,R3,FUNCXT             ;IF SET, FORMAT ERROR
 50     008C 8F      3C   0A38                MOVZWL  #SS$_DRVERR,R0                     ;SET DRIVE ERROR STATUS
 53     1807 8F      B3   0A3D                BITW    #MT_ER_M_DTE!-                     ;DRIVE TIMING ERROR OR,
                          0A42                        MT_ER_M_ILF!-                      ;ILLEGAL FUNCTION OR,
                          0A42                        MT_ER_M_ILR!-                      ;ILLEGAL REGISTER OR,
                          0A42                        MT_ER_M_NEF!-                      ;NON-EXECUTABLE FUNCTION OR,
                          0A42                        MT_ER_M_RMR,R3                     ;REGISTER MODIFY REFUSE
             2F        12 0A42                BNEQ    FUNCXT                             ;IF NEQ YES
 50     01F4 8F      3C   0A44                MOVZWL  #SS$_PARITY,R0                     ;SET PARITY ERROR STATUS
 53     85E8 8F      B3   0A49                BITW    #MT_ER_M_CRC!-                     ;CRC ERROR OR,
                          0A4E                        MT_ER_M_CPAR!-                     ;CONTROL BUS PARITY ERROR OR,
                          0A4E                        MT_ER_M_COR!-                      ;CORRECTABLE DATA ERROR (PE) OR,
                          0A4E                        MT_ER_M_CS!-                       ;CORRECTABLE SKEW (PE) OR,
                          0A4E                        MT_ER_M_DPAR!-                     ;DATA PARITY ERROR OR,
                          0A4E                        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            .SBTTL  FUNCTION COMPLETION COMMON EXIT
                        0A73       ;
                        0A73       ; FUNCTION COMPLETION COMMON EXIT
                        0A73       ;
                        0A73       ; THIS ROUTINE IS JUMPED TO AT THE END OF ALL MAGTAPE OPERATIONS.
                        0A73       ;
                        0A73       ; INPUTS:
                        0A73       ;
                        0A73       ;       R0 = FINAL I/O COMPLETION STATUS.
                        0A73       ;       R2 = DRIVE STATUS REGISTER.
                        0A73       ;
                        0A73       ;       UCB$B_FEX(R5) = FUNCTION EXECUTION INDEX.
                        0A73       ;
                        0A73       ; OUTPUTS:
                        0A73       ;
                        0A73       ;       THE FINAL DRIVE STATUS IS TESTED AND THE FOLLOWING STATUS BITS ARE SET
                        0A73       ;       IN THE SECOND WORD OF THE DEVICE DEPENDENT CHARACTERISTICS LONGWORD.
                        0A73       ;
                        0A73       ;               MT$M_BOT = SET IF TAPE IS AT BEGINNING OF TAPE AT END OF FUNCTION.
                        0A73       ;               MT$M_EOF = SET IF A VALID TAPE MARK WAS DETECTED DURING THE TAPE
                        0A73       ;                       OPERATION.
                        0A73       ;               MT$M_EOT = SET IF AN END OF TAPE CONDITION WAS PRESENT AT THE END
                        0A73       ;                       OF THE TAPE OPERATION AND THE FUNCTION WAS A READ DATA
                        0A73       ;                       FORWARD, WRITECHECK DATA FORWARD, WRITE DATA FORWARD, OR
                        0A73       ;                       SPACE RECORD FORWARD FUNCTION.
                        0A73       ;               MT$M_HWL = SET IF THE SLAVE DRIVE HAS A TAPE MOUNTED THAT DOES NOT
                        0A73       ;                       CONTAIN A WRITE RING.
                        0A73       ;
                        0A73
                        0A73       FUNCXT:                                                 ;FUNCTION EXIT
51      0092 C5     9A  0A73               MOVZBL  UCB$B_FEX(R5),R1                ;GET FUNCTION DISPATCH INDEX
   46 A5    0F      AA  0A78               BICW    #<MT$M_BOT!-                    ;CLEAR BEGINNING OF TAPE AND,
                        0A7C                       MT$M_EOF!-                      ;END OF FILE AND,
                        0A7C                       MT$M_EOT!-                      ;END OF TAPE AND,
                        0A7C                       MT$M_HWL>@-16,UCB$L_DEVDEPEND+2(R5) ;HARDWARE WRITE LOCK
   0C 52    01      E1  0A7C               BBC     #MT_DS_V_BOT,R2,10$             ;IF CLR, NOT AT BEGINNING OF TAPE
   46 A5    01      A8  0A80               BISW    #<MT$M_BOT@-16>,UCB$L_DEVDEPEND+2(R5) ;SET BEGINNING OF TAPE
   46 A5    10      AA  0A84               BICW    #<MT$M_LOST@-16>,UCB$L_DEVDEPEND+2(R5) ;CLEAR TAPE POSITION LOST
      00B0 C5       D4  0A88               CLRL    UCB$L_RECORD(R5)                ;CLEAR TAPE POSITION
   23 52    02      E1  0A8C       10$:    BBC     #MT_DS_V_TM,R2,30$              ;IF CLR, NO TAPE MARK DETECTED
   46 A5    02      A8  0A90               BISW    #<MT$M_EOF@-16>,UCB$L_DEVDEPEND+2(R5) ;SET END OF FILE
      51    0A      91  0A94               CMPB    #CDF_WRITECHECK,R1              ;DATA TRANSFER FUNCTION?
            1A      1A  0A97               BGTRU   30$                             ;IF GTRU NO
      51    13      91  0A99               CMPB    #CDF_WRITEMARK,R1               ;WRITE TAPE MARK FUNCTION?
            05      13  0A9C               BEQL    20$                             ;IF EQL YES
      51    10      91  0A9E               CMPB    #CDF_READPRESET,R1              ;DATA TRANSFER FUNCTION?
            10      1B  0AA1               BLEQU   30$                             ;IF LEQU NO
      00BC C5       B4  0AA3       20$:    CLRW    UCB$W_MT_FC(R5)                 ;CLEAR FRAME COUNT
50      005C 8F     B1  0AA7               CMPW    #SS$_DATACHECK,R0               ;WRITE CHECK ERROR?
            05      12  0AAC               BNEQ    30$                             ;IF NEQ NO
50      0870 8F     3C  0AAE               MOVZWL  #SS$_ENDOFFILE,R0               ;SET END OF FILE STATUS
   25 52    0A      E1  0AB3       30$:    BBC     #MT_DS_V_EOT,R2,40$             ;IF CLR, NOT AT END OF TAPE
      51    05      91  0AB7               CMPB    #CDF_SPCFILREV,R1               ;SPACE FILE REVERSE?
            20      13  0ABA               BEQL    40$                             ;IF EQL YES
      51    07      91  0ABC               CMPB    #CDF_SPCRECREV,R1               ;SPACE RECORD REVERSE?
            1B      13  0ABF               BEQL    40$                             ;IF EQL YES
      51    0D      91  0AC1               CMPB    #CDF_WRITECHECKR,R1             ;WRITE CHECK REVERSE?
```

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

```
0B3E          .SBTTL  TM03-TE16/TU77 HARDWARE FUNCTION EXECUTION
0B3E    ;
0B3E    ; FEX - TM03-TE16/TU77 HARDWARE FUNCTION EXECUTION
0B3E    ;
0B3E    ; THIS ROUTINE IS CALLED VIA A BSB WITH A WORD IMMEDIATELY FOLLOWING THAT
0B3E    ; SPECIFIES THE ADDRESS OF AN ERROR ROUTINE. ALL DATA IS ASSUMED TO HAVE BEEN
0B3E    ; SET UP IN THE UCB BEFORE THE CALL. THE APPROPRIATE PARAMETERS ARE LOADED
0B3E    ; INTO DEVICE REGISTERS AND THE FUNCTION IS INITIATED. IF THE FUNCTION IS AN
0B3E    ; IMMEDIATE FUNCTION CONTROL RETURNS IMMEDIATELY. ELSE THE RETURN ADDRESS
0B3E    ; IS STORED IN THE UCB AND A WAITFOR INTERRUPT IS EXECUTED. WHEN THE INTER-
0B3E    ; RUPT OCCURS, CONTROL IS RETURNED TO THE CALLER.
0B3E    ;
0B3E    ; INPUTS:
0B3E    ;
0B3E    ;       R0 = FUNCTION TABLE DISPATCH INDEX.
0B3E    ;       R3 = ADDRESS OF DRIVE CONTROL STATUS REGISTER 1.
0B3E    ;       R4 = ADDRESS OF MBA CONFIGURATION STATUS REGISTER.
0B3E    ;       R5 = DEVICE UNIT UCB ADDRESS.
0B3E    ;
0B3E    ;       00(SP) = RETURN ADDRESS OF CALLER.
0B3E    ;       04(SP) = RETURN ADDRESS OF CALLER'S CALLER.
0B3E    ;
0B3E    ;       IMMEDIATELY FOLLOWING INLINE AT THE CALL SITE IS A WORD WHICH CONTAINS
0B3E    ;       A BRANCH DESTINATION TO AN ERROR RETRY ROUTINE.
0B3E    ;
0B3E    ; OUTPUTS:
0B3E    ;
0B3E    ;       THERE ARE FOUR EXITS FROM THIS ROUTINE:
0B3E    ;
0B3E    ;       1. SPECIAL CONDITION - THIS EXIT IS TAKEN IF A POWER FAILURE OCCURS
0B3E    ;               OR THE OPERATION TIMES OUT. IT IS A JUMP TO THE APPROPRIATE
0B3E    ;               ERROR ROUTINE.
0B3E    ;
0B3E    ;       2. FATAL ERROR - THIS EXIT IS TAKEN IF A FATAL CONTROLLER OR DRIVE
0B3E    ;               ERROR OCCURS OR IF ANY ERROR OCCURS AND ERROR RETRY IS
0B3E    ;               INHIBITED. IT IS A JUMP TO THE FATAL ERROR EXIT ROUTINE.
0B3E    ;
0B3E    ;       3. RETRIABLE ERROR - THIS EXIT IS TAKEN IF A RETRIABLE CONTROLLER
0B3E    ;               OR DRIVE ERROR OCCURS AND ERROR RETRY IS NOT INHIBITED.
0B3E    ;               IT CONSISTS OF TAKING THE ERROR BRANCH EXIT.
0B3E    ;
0B3E    ;       4. SUCCESSFUL OPERATION - THIS EXIT IS TAKEN IF NO ERROR OCCURS
0B3E    ;               DURING THE OPERATION. IT CONSISTS OF A RETURN INLINE.
0B3E    ;
0B3E    ;       IN ALL CASES IF AN ERROR OCCURS, AN ATTEMPT IS MADE TO LOG THE ERROR.
0B3E    ;
0B3E    ;       IN ALL CASES FINAL DRIVE AND CONTROLLER REGISTERS ARE RETURNED VIA
0B3E    ;       THE GENERAL REGISTERS R0, R1, AND R2, AND THE UCB.
0B3E    ;
0B3E    ;       R0 = DRIVE ERROR REGISTER.
0B3E    ;       R1 = MBA STATUS REGISTER.
0B3E    ;       R2 = DRIVE STATUS REGISTER.
0B3E    ;
0B3E    ;       UCB$W_MT_FC(R5) = FRAME COUNT REGISTER.
0B3E    ;
0B3E    ;
0B3E    FEX:                                            ;FUNCTION EXECUTOR
```

```
         009C C5 8ED0  0B3E         POPL    UCB$L_DPC(R5)               ;SAVE DRIVER PC VALUE
      0093 C5    50 90  0B43         MOVB    R0,UCB$B_CEX(R5)            ;SAVE CASE INDEX
                        0B48 RESTART:                                    ;RESTART FUNCTION
  24 A3   00C6 C5    3C  0B48         MOVZWL  UCB$W_MT_TC(R5),MT_TC(R3)  ;SELECT DRIVE AND SET CHARACTERISTICS
        63    09    9A  0B4E         MOVZBL  #F_DRVCLR!1,MT_CS1(R3)     ;CLEAR DRIVE
                        0B51         CASE    R0,<-                       ;DISPATCH TO PROPER FUNCTION ROUTINE
                        0B51                 IMMED,-                     ;NO OPERATION
                        0B51                 POSIT,-                     ;UNLOAD VOLUME
                        0B51                 POSIT,-                     ;SPACE FILE FORWARD
                        0B51                 RECAL,-                     ;REWIND
                        0B51                 IMMED,-                     ;DRIVE CLEAR
                        0B51                 POSIT,-                     ;SPACE FILE REVERSE
                        0B51                 POSIT,-                     ;ERASE TAPE
                        0B51                 POSIT,-                     ;SPACE RECORD REVERSE
                        0B51                 IMMED,-                     ;NO OPERATION
                        0B51                 POSIT,-                     ;SPACE RECORD FORWARD
                        0B51                 XFER,-                      ;WRITE CHECK FORWARD
                        0B51                 XFER,-                      ;WRITE DATA FORWARD
                        0B51                 XFER,-                      ;READ DATA FORWARD
                        0B51                 XFER,-                      ;WRITECHECK REVERSE
                        0B51                 XFER,-                      ;WRITE DATA FORWARD
                        0B51                 XFER,-                      ;READ DATA REVERSE
                        0B51                 RECAL,-                     ;READ IN PRESET
                        0B51                 IMMED,-                     ;SET TAPE CHARACTERISTICS
                        0B51                 IMMED,-                     ;SENSE CHARACTERISTICS
                        0B51                 POSIT,-                     ;WRITE TAPE MARK
                        0B51                 >                           ;
                        0B7D
                        0B7D ;
                        0B7D ; INTERNAL SPACING FUNCTIONS
                        0B7D ;
                        0B7D ;       SPACE RECORD FORWARD, AND
                        0B7D ;       SPACE RECORD REVERSE.
                        0B7D ;
                        0B7D ; THESE FUNCTIONS ARE EXECUTED TO EFFECT SPACING OPERATIONS DURING THE ERROR
                        0B7D ; RECOVERY OF WRITE TAPE MARK, ERASE TAPE, AND ALL DATA TRANSFER FUNCTIONS.
                        0B7D ; WRITE CHECK AND WRITE CHECK REVERSE FUNCTIONS ARE USED TO ACTUALLY POSITION
                        0B7D ; THE TAPE. IF A NOISE RECORD IS ENCOUNTERED, IT IS IGNORED.
                        0B7D ;
                        0B7D
                        0B7D INTSPC:                                     ;INTERNAL SPACE FUNCTIONS
            065A    30  0B7D         BSBW    TM_SETINTSPC                ;SETUP FOR INTERNAL SPACE FUNCTION
     50   0093 C5    9A  0B80         MOVZBL  UCB$B_CEX(R5),R0            ;RETRIEVE FUNCTION INDEX
          50    15    91  0B85         CMPB    #CDF_INTSPCREV,R0           ;SPACE RECORD REVERSE?
                06    12  0B88         BNEQ    10$                         ;IF NEQ NO
  0C A4    01FF 8F    3C  0B8A         MOVZWL  #511,MBA$L_VAR(R4)          ;ADJUST VIRTUAL ADDRESS REGISTER
                        0B90 10$:    DSBINT                              ;DISABLE INTERRUPTS
   57 64 A5    05    E0  0B96         BBS     #UCB$V_POWER,UCB$W_STS(R5),20$ ;IF SET, POWER FAILED
  63   F4B6 CF 40    9A  0B9B         MOVZBL  FTAB[R0],MT_CS1(R3)        ;EXECUTE FUNCTION
                        0BA1         WFIKPCH RETREG,TIME_OUT[R0]         ;WAIT FOR INTERRUPT AND KEEP CHANNEL
  00B4 C5    08 A4    D0  0BAE         MOVL    MBA$L_SR(R4),UCB$L_MT_SR(R5) ;SAVE FINAL MBA STATUS
            05FC    30  0BB4         BSBW    TM_SAVDRVSTS                ;SAVE DRIVE STATUS
                        0BB7         IOFORK                              ;CREATE FORK PROCESS
   33 64 A5    05    E0  0BBD         BBS     #UCB$V_POWER,UCB$W_STS(R5),30$ ;IF SET, POWER FAILED
00B4 C5   00013F80 8F    CA  0BC2         BICL    #MBA$M_SR_ATTN!-            ;CLEAR ATTENTION AND,
                        0BCB                 MBA$M_SR_DLT!-              ;DATA LATE AND,
                        0BCB                 MBA$M_SR_DTABT!-            ;DATA TRANSFER ABORT AND,
```

```
                              0BCB                 MBA$M_SR_DTCOMP!-          ;DATA TRANSFER COMPLETE AND,
                              0BCB                 MBA$M_SR_MBEXC!-           ;MASSBUSS EXCEPTION AND,
                              0BCB                 MBA$M_SR_MXF!-             ;MISSED TRANSFER AND,
                              0BCB                 MBA$M_SR_WCKLWR!-          ;WRITE CHECK LOWER AND,
                              0BCB                 MBA$M_SR_WCKUPR,UCB$L_MT_SR(R5) ;WRITE CHECK UPPER
                  28     12   0BCB         BNEQ    30$                        ;IF NEQ FATAL SPACING ERROR
           50   0093 C5  9A   0BCD         MOVZBL  UCB$B_CEX(R5),R0           ;GET FUNCTION INDEX
  00BA C5    F4F5 CF40   AA   0BD2         BICW    XTAB[R0],UCB$W_MT_ER(R5)   ;CLEAR EXTRANEOUS DRIVE ERROR BITS
                  19     12   0BDA         BNEQ    30$                        ;IF NEQ FATAL SPACING ERROR
       13 00B8 C5   02   E0   0BDC         BBS     #MT_DS_V_TM,UCB$W_MT_DS(R5),30$ ;IF SET, TAPE MARK ENCOUNTERED
                    05   E0   0BE2         BBS     #MT_DS_V_PES,-             ; If at 1600 BPI, branch around test
          0D 00B8 C5          0BE4                 UCB$W_MT_DS(R5),30$        ;   for minimum length record.
        00BC C5     0E   B1   0BE8         CMPW    #MIN_RECORD,UCB$W_MT_FC(R5) ;MINIMUM RECORD READ?
                  06     1B   0BED         BLEQU   30$                        ;IF LEQU YES
                FF56     31   0BEF         BRW     RESTART                    ;RESTART FUNCTION
                0127     31   0BF2  20$:   BRW     ENBXIT                     ;
                013E     31   0BF5  30$:   BRW     RETREG                     ;
                              0BF8
                              0BF8  ;
                              0BF8  ; POSITIONING FUNCTION EXECUTION
                              0BF8  ;
                              0BF8  ;      FUNCTIONS INCLUDE:
                              0BF8  ;
                              0BF8  ;              SPACE FILE FORWARD,
                              0BF8  ;              SPACE FILE REVERSE,
                              0BF8  ;              ERASE TAPE,
                              0BF8  ;              SPACE RECORD REVERSE,
                              0BF8  ;              SPACE RECORD FORWARD,
                              0BF8  ;              UNLOAD, AND
                              0BF8  ;              WRITE TAPE MARK.
                              0BF8  ;
                              0BF8  ; THE FUNCTION IS INITIATED AND A WAITFOR INTERRUPT IS EXECUTED. WHEN THE INTERRUPT
                              0BF8  ; OCCURS FINAL DEVICE REGISTERS ARE RETURNED TO THE CALLER.
                              0BF8  ;
                              0BF8
                              0BF8  POSIT:                                    ;POSITIONING FUNCTION EXECUTION
                              0BF8         DSBINT                             ;DISABLE ALL INTERRUPTS
         2D 64 A5   05   E0   0BFE         BBS     #UCB$V_POWER,UCB$W_STS(R5),ENBEXT ;IF SET, POWER HAS FAILED
        14 A3   00BE C5  32   0C03         CVTWL   UCB$W_MT_SPACNT(R5),MT_FC(R3) ;LOAD FRAME COUNT REGISTER
           63    F448 CF40   9A   0C09     MOVZBL  FTAB[R0],MT_CS1(R3)        ;EXECUTE FUNCTION
                              0C0F         WFIKPCH RETREG,TIME_OUT[R0]        ;WAITFOR INTERRUPT AND KEEP CHANNEL
                00C9     31   0C1C         BRW     SETDEN                     ;
                              0C1F
                              0C1F  ;
                              0C1F  ; IMMEDIATE FUNCTION EXECUTION
                              0C1F  ;
                              0C1F  ;      FUNCTIONS INCLUDE:
                              0C1F  ;
                              0C1F  ;              NO OPERATION, AND
                              0C1F  ;              DRIVE CLEAR.
                              0C1F  ;
                              0C1F  ; THESE FUNCTIONS ARE EXECUTED IMMEDIATELY AND THE FINAL DEVICE REGISTERS
                              0C1F  ; ARE RETURNED TO THE CALLER.
                              0C1F  ;
                              0C1F
                              0C1F  IMMED:                                    ;IMMEDIATE FUNCTION EXECUTION
                              0C1F         DSBINT                             ;DISABLE INTERRUPTS
```

```
   06 64 A5    05    E0  0C25              BBS     #UCB$V_POWER,UCB$W_STS(R5),ENBEXT ;IF SET, POWER HAS FAILED
   63   F427 CF40    9A  0C2A              MOVZBL  FTAB[R0],MT_CS1(R3)       ;EXECUTE FUNCTION
                         0C30      ENBEXT:                                   ;
             00E9    31  0C30              BRW     ENBXIT                    ;
                         0C33
                         0C33      ;
                         0C33      ; RECALIBRATE FUNCTION EXECUTION
                         0C33      ;
                         0C33      ;       FUNCTIONS INCLUDE:
                         0C33      ;
                         0C33      ;               READ IN PRESET, AND
                         0C33      ;               REWIND.
                         0C33      ;
                         0C33      ; THE FUNCTION IS INITIATED AND A WAITFOR INTERRUPT IS EXECUTED. WHEN THE INTERRUPT
                         0C33      ; OCCURS A TEST IS MADE TO SEE IF POSITIONING IS STILL IN PROGRESS. IF POSITIONING
                         0C33      ; IS STILL IN PROGRESS, THEN ANOTHER WAITFOR INTERRUPT IS EXECUTED. ELSE FINAL DRIVE
                         0C33      ; REGISTERS ARE RETURNED TO THE CALLED.
                         0C33      ;
                         0C33
                         0C33      RECAL:                                    ;RECALIBRATE FUNCTION EXECUTION
                         0C33              DSBINT                            ;DISABLE INTERRUPTS
   F2 64 A5    05    E0  0C39              BBS     #UCB$V_POWER,UCB$W_STS(R5),ENBEXT ;IF SET, POWER HAS FAILED
   63   F413 CF40    9A  0C3E              MOVZBL  FTAB[R0],MT_CS1(R3)       ;EXECUTE FUNCTION
                         0C44              WFIKPCH RETREG,#60*7              ;WAITFOR INTERRUPT
   52    04 A3    12    78  0C52           ASHL    #31-MT_DS_V_PIP,MT_DS(R3),R2 ;POSITIONING IN PROGRESS?
                2C    18  0C57             BGEQ    10$                       ;IF GEQ NO
                         0C59              IOFORK                            ;CREATE FORK PROCESS
                         0C5F              DSBINT                            ;DISABLE INTERRUPTS
   52    04 A3    12    78  0C65           ASHL    #31-MT_DS_V_PIP,MT_DS(R3),R2 ;POSITIONING IN PROGRESS?
                C4    18  0C6A             BGEQ    ENBEXT                    ;IF GEQ NO
   BF 64 A5    05    E0  0C6C              BBS     #UCB$V_POWER,UCB$W_STS(R5),ENBEXT ;IF SET, POWER HAS FAILED
11 009A C5     07    E0  0C71              BBS     #IO$V_NOWAIT,UCB$W_FUNC(R5),20$ ;IF SET, NO WAIT FOR REWIND
                         0C77              WFIRLCH RETREG,#60*7              ;WAITFOR FINAL INTERRUPT
             0089    31  0C85      10$:    BRW     DRVREG                    ;
      68 A5    01    A8  0C88      20$:    BISW    #UCB$M_MT_REWIND,UCB$W_DEVSTS(R5) ;SET REWIND IN PROGRESS
00B8 C5    04 A3    F7  0C8C               CVTLW   MT_DS(R3),UCB$W_MT_DS(R5) ;SAVE DRIVE STATUS
        00BA C5    B4  0C92                CLRW    UCB$W_MT_ER(R5)           ;CLEAR SAVED ERROR STATUS
                         0C96              ENBINT                            ;ENABLE INTERRUPTS
             009A    31  0C99              BRW     RETREG                    ;
                         0C9C
                         0C9C      ;
                         0C9C      ; TRANSFER FUNCTION EXECUTION
                         0C9C      ;
                         0C9C      ;       FUNCTIONS INCLUDE:
                         0C9C      ;
                         0C9C      ;               WRITE CHECK,
                         0C9C      ;               WRITE DATA,
                         0C9C      ;               READ DATA,
                         0C9C      ;               READ DATA REVERSE, AND
                         0C9C      ;               WRITECHECK DATA REVERSE.
                         0C9C      ;
                         0C9C      ; THE MAP REGISTERS, BYTE COUNT REGISTER, AND VIRTUAL ADDRESS REGISTERS ARE LOADED.
                         0C9C      ; THE FUNCTION IS INITIATED AND A WAITFOR INTERRUPT IS EXECUTED. WHEN THE INTERRUPT
                         0C9C      ; OCCURS FINAL DEVICE REGISTERS ARE RETURNED TO THE CALLER.
                         0C9C      ;
                         0C9C
                         0C9C      XFER:                                     ;TRANSFER FUNCTION EXECUTION
```

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

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

```
                   15   13  0DB9              BEQL    30$                        ;IF EQL YES
          0093 C5  05   91  0DBB              CMPB    #CDF_SPCFILREV,UCB$B_CEX(R5) ;SPACE FILE REVERSE?
                   0E   13  0DC0              BEQL    30$                        ;IF EQL YES
          0093 C5  07   91  0DC2              CMPB    #CDF_SPCRECREV,UCB$B_CEX(R5) ;SPACE RECORD REVERSE?
                   07   13  0DC7              BEQL    30$                        ;IF EQL YES
          0093 C5  0D   91  0DC9              CMPB    #CDF_WRITECHECKR,UCB$B_CEX(R5) ;WRITECHECK REVERSE?
                   1D   12  0DCE              BNEQ    60$                        ;IF NEQ NO
             00BC C5    B4  0DD0     30$:     CLRW    UCB$W_MT_FC(R5)            ;CLEAR SAVED FRAME COUNT REGISTER
                            0DD4
                            0DD4     ;
                            0DD4     ; RETRIABLE ERROR EXIT
                            0DD4     ;
                            0DD4
       7E    009C D5    32  0DD4     40$:     CVTWL   @UCB$L_DPC(R5),-(SP)       ;GET BRANCH DISPLACEMENT
          009C C5  8E   C0  0DD9              ADDL    (SP)+,UCB$L_DPC(R5)        ;CALCULATE RETURN ADDRESS - 2
       09 009A C5  0F   E0  0DDE              BBS     #IO$V_INHRETRY,UCB$W_FUNC(R5),60$ ;IF SET, RETRY INHIBITED
          009C C5  02   C0  0DE4     50$:     ADDL    #2,UCB$L_DPC(R5)           ;ADJUST TO CORRECT RETURN ADDRESS
             009C D5    17  0DE9              JMP     @UCB$L_DPC(R5)             ;RETURN TO DRIVER
                            0DED
                            0DED     ;
                            0DED     ; FATAL CONTROLLER OR DRIVE ERROR
                            0DED     ;        Here we log a device error (unless the only error is that the media
                            0DED     ;                on the device is OFF-LINE) and then we branch to FATALERR.
                            0DED     ;
                            0DED
                            0DED     60$:
          07 52    0C   E0  0DED              BBS     #MT_DS_V_MOL,R2,63$         ; Branch to log error if media ON-LINE.
       50    4000 8F    B1  0DF1              CMPW    #MT_ER_M_UNS,R0             ; Drive is OFF-LINE. See if ONLY error
                            0DF6                                                  ;   is UNSAFE.  If so, this implies
                            0DF6                                                  ;   ONLY error is that media is OFF-LINE.
                   06   13  0DF6              BEQL    66$                         ; Branch around logging of error if
                            0DF8                                                  ;   ONLY off-line.
                            0DF8     63$:
        00000000'GF     16  0DF8              JSB     G^ERL$DEVICERR              ; Log Device Error.
                            0DFE     66$:
                 FC03   31  0DFE              BRW     FATALERR                    ;
                            0E01
                            0E01     ;
                            0E01     ; SPECIAL CONDITION ( POWER FAILURE OR DEVICE TIME OUT)
                            0E01     ;
                            0E01
          68 A5    01   AA  0E01     70$:     BICW    #UCB$M_MT_REWIND,UCB$W_DEVSTS(R5) ;CLEAR REWIND IN PROGRESS
          46 A5    10   A8  0E05              BISW    #<MT$M_LOST@-16>,UCB$L_DEVDEPEND+2(R5) ;SET POSITION LOST
       65 64 A5    05   E4  0E09              BBSC    #UCB$V_POWER,UCB$W_STS(R5),100$ ;IF SET, POWER FAILURE
                            0E0E
                            0E0E     ;
                            0E0E     ; DEVICE TIME OUT
                            0E0E     ;
                            0E0E
                 03A2   30  0E0E              BSBW    TM_SAVDRVSTS                ; Save drive registers in UCB for ERROR LOGG
        00000000'GF     16  0E11              JSB     G^ERL$DEVICTMO              ;LOG DEVICE TIME OUT
          53    24 A5   D0  0E17              MOVL    UCB$L_CRB(R5),R3            ;GET ADDRESS OF PRIMARY CRB
          53    20 A3   D0  0E1B              MOVL    CRB$L_LINK(R3),R3           ;GET ADDRESS OF SECONDARY CRB
          53    2C A3   D0  0E1F              MOVL    CRB$L_INTD+VEC$L_IDB(R3),R3 ;GET ADDRESS OF SECONDARY IDB
          04 A3    55   D1  0E23              CMPL    R5,IDB$L_OWNER(R3)          ;DEVICE OWN CONTROLLER?
                   22   12  0E27              BNEQ    80$                         ;IF NEQ NO
                            0E29              DSBINT                              ;DISABLE INTERRUPTS
```

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

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

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

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

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

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

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

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

```
                          11B3        .SBTTL  TM03-TE16/TU77 DRIVE STATUS SAVE ROUTINE
                          11B3  ;
                          11B3  ; TM_SAVDRVSTS - TM03-TE16/TU77 DRIVE STATUS SAVE ROUTINE
                          11B3  ;
                          11B3  ; THIS ROUTINE IS CALLED FROM DEVICE INTERRUPT LEVEL TO SAVE FINAL DRIVE REGISTERS
                          11B3  ; AND THEN CLEAR THE DRIVE.
                          11B3  ;
                          11B3  ; INPUTS:
                          11B3  ;
                          11B3  ;       R3 = ADDRESS OF DRIVE CONTROL STATUS REGISTER 1.
                          11B3  ;       R4 = ADDRESS OF MBA CONFIGURATION STATUS REGISTER.
                          11B3  ;       R5 = DEVICE UNIT UCB ADDRESS.
                          11B3  ;
                          11B3  ; OUTPUTS:
                          11B3  ;
                          11B3  ;       THE FINAL DRIVE REGISTERS ARE SAVED AND THE DRIVE IS CLEARED.
                          11B3  ;
                          11B3
                          11B3  TM_SAVDRVSTS:                             ;SAVE DRIVE STATUS
  00BC C5   14 A3   F7    11B3        CVTLW   MT_FC(R3),UCB$W_MT_FC(R5)  ;SAVE FRAME COUNT REGISTER
  00BA C5   08 A3   F7    11B9        CVTLW   MT_ER(R3),UCB$W_MT_ER(R5)  ;SAVE ERROR STATUS REGISTER
  00B8 C5   04 A3   F7    11BF        CVTLW   MT_DS(R3),UCB$W_MT_DS(R5)  ;SAVE DRIVE STATUS REGISTER
  00C2 C5   24 A3   F7    11C5        CVTLW   MT_TC(R3),UCB$W_MT_TC_SAV(R5) ;SAVE TAPE CONTROL REGISTER
    00C0 C5    63   F7    11CB        CVTLW   MT_CS1(R3),UCB$W_MT_CS1(R5) ;SAVE DRIVE CONTROL REGISTER
  00D0 C5   1C A3   F7    11D0        CVTLW   MT_CC(R3),UCB$W_MT_CC_SAV(R5)   ; Save Check Character Register.
        63    09    9A    11D6        MOVZBL  #F_DRVCLR!1,MT_CS1(R3)    ;CLEAR DRIVE
                    05    11D9        RSB                               ;
```

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

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

```
              7E    D4  1240  10$:    CLRL    -(SP)                           ;SET STARTING UNIT NUMBER
                        1242  20$:
          52  6E    D0  1242          MOVL    (SP),R2                         ; Unit number to register
       53  08 BE    D0  1245          MOVL    @8(SP),R3                       ;RETRIEVE ADDRESS OF IDB
          54  63    D0  1249          MOVL    IDB$L_CSR(R3),R4                ;RETRIEVE ADDRESS OF TM03-TE16/TU77 REGISTER
       24 A4  52    D0  124C          MOVL    R2,MT_TC(R4)                    ;SELECT NEXT SLAVE UNIT
    55  18 A342     D0  1250          MOVL    IDB$L_UCBLST(R3)[R2],R5         ;GET ADDRESS OF UCB
              52    13  1255          BEQL    60$                             ;IF EQL NONE
       53  04 A4    D0  1257          MOVL    MT_DS(R4),R3                    ;READ SLAVE DRIVE STATUS
                        125B
                        125B  ;+
                        125B  ; POLL STATUS FOR ATTENTION CONDITIONS
                        125B  ;
                        125B  ; THE SEQUENCE IN WHICH ATTENTION CONDITIONS ARE POLLED IS CRITICAL.
                        125B  ; DUE TO AN ERROR IN THE TE-16 DRIVE DESIGN, THE ATTEMPT TO RESET AN
                        125B  ; ATTENTION CONDITION WITH DRIVE-CLEAR, WHILE REWINDING, RESULTS IN
                        125B  ; AN RMR ERROR (REGISTER MODIFICATION REFUSED). SINCE ALL ERRORS ARE
                        125B  ; CLEARED WITH DRIVE-CLEAR, RMR IS REASSERTED SWAMPING THE SYSTEM
                        125B  ; WITH INTERRUPTS UNTIL THE REWIND COMPLETES.
                        125B  ;
                        125B  ; TO AVOID THIS CONDITION, NO ATTENTION CONDITIONS ARE SERVICED IF
                        125B  ; PIP IS SET FOR THE DRIVE. THIS HAS THE EFFECT OF BLOCKING INTERR-
                        125B  ; UPTS FROM OTHER DRIVES BUT IS PREFERRABLE TO LOCKING UP THE ENTIRE
                        125B  ; SYSTEM. THE CORRECT REMEDY IS TO FIX THE PROBLEM IN THE DRIVE ELEC-
                        125B  ; TRONICS BY BLOCKING ALL ATTENTION CONDITIONS UNTIL REWIND IS DONE
                        125B  ; (AS FOR THE TU-77). UNDER THESE CONDITIONS, PIP SHOULD NEVER OCCUR
                        125B  ; IN CONJUNCTION WITH OTHER ATTENTION CONDITIONS.
                        125B  ;
                        125B  ;-
                        125B
    1E 53      00   E0  125B          BBS     #MT_DS_V_SLA,R3,30$             ;IF SET, SLAVE TRANSITION TO ONLINE
    27 53      0C   E1  125F          BBC     #MT_DS_V_MOL,R3,40$             ;IF CLR, MEDIUM OFFLINE
    45 53      0D   E0  1263          BBS     #MT_DS_V_PIP,R3,70$             ;IF SET POSITIONING IN PROGRESS
    41 53      01   E1  1267          BBC     #MT_DS_V_BOT,R3,70$             ;IF CLR, NOT AT BEGINNING OF TAPE
    68 A5      01   AA  126B          BICW    #UCB$M_MT_REWIND,UCB$W_DEVSTS(R5) ;CLEAR REWIND IN PROGRESS
 35 64 A5      01   E5  126F          BBCC    #UCB$V_INT,UCB$W_STS(R5),60$ ;IF CLR, INTERRUPT NOT EXPECTED
    53      10 A5   7D  1274          MOVQ    UCB$L_FR3(R5),R3                ;RESTORE DRIVER CONTEXT
            0C B5   16  1278          JSB     @UCB$L_FPC(R5)                  ;CALL DRIVER
               2F   11  127B          BRB     70$                             ;
    68 A5      02   A8  127D  30$:    BISW    #UCB$M_MT_PRVMOL,UCB$W_DEVSTS(R5) ;SET PREVIOUS MOL STATE
    68 A5      01   AA  1281          BICW    #UCB$M_MT_REWIND,UCB$W_DEVSTS(R5) ;CLEAR REWIND IN PROGRESS
             FEE7   30  1285          BSBW    TM_UNSOLNT                      ;CALL UNSOLICITED INTERRUPT ENTRY
               22   11  1288          BRB     70$                             ;
 1D 68 A5      01   E5  128A  40$:    BBCC    #UCB$V_MT_PRVMOL,UCB$W_DEVSTS(R5),70$ ;IF CLR, PREVIOUSLY OFFLINE
    68 A5      01   AA  128F          BICW    #UCB$M_MT_REWIND,UCB$W_DEVSTS(R5) ;CLEAR REWIND IN PROGRESS
    7E   64 A5     3C   3C  1293      MOVZWL  UCB$W_STS(R5),-(SP)             ;SAVE CURRENT DRIVE STATUS
             FED5   30  1297          BSBW    TM_UNSOLNT                      ;CALL UNSOLICITED INTERRUPT ENTRY
    06 6E      0B   E1  129A          BBC     #UCB$V_VALID,(SP),45$           ;DON'T RESET VOLUME VALID IF FLAG WAS CLEAR
 64 A5       0800 8F A8  129E         BISW    #UCB$M_VALID,UCB$W_STS(R5) ;RESET SOFTWARE VOLUME VALID
       5E      04   C0  12A4  45$:    ADDL    #4,SP                           ;CLEAN THE STACK
               03   11  12A7          BRB     70$                             ;
       64      09   9A  12A9  60$:    MOVZBL  #F_DRVCLR!1,MT_CS1(R4)          ;CLEAR DRIVE
    92 6E      07   F3  12AC  70$:    AOBLEQ  #7,(SP),20$                     ;ANY MORE SLAVE UNITS TO SCAN?
                        12B0                                                  ; Refresh R3 and R4 after falling thru.
       53  08 BE    D0  12B0          MOVL    @8(SP),R3                       ; R3 => IDB
          54  63    D0  12B4          MOVL    IDB$L_CSR(R3),R4                ; R4 => TM03-TE16/TU77 REGISTERS
              8E    D5  12B7          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$L_CSR               = 00000000
AT$_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               0000094B 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$M_DATACHECK                  = 00004000                IRP$S_FCODE                     = 00000006
IO$M_NOWAIT                     = 00000080                IRP$V_FCODE                     = 00000000
IO$V_DATACHECK                  = 0000000E                IRP$V_PHYSIO                    = 00000008
IO$V_INHEXTGAP                  = 0000000C                IRP$V_VIRTUAL                   = 00000004
IO$V_INHRETRY                   = 0000000F                IRP$W_BCNT                      = 00000032
IO$V_NOWAIT                     = 00000007                IRP$W_BOFF                      = 00000030
IO$V_REVERSE                    = 00000006                IRP$W_FUNC                      = 00000020
IO$_ACCESS                      = 00000032                IRP$W_STS                       = 0000002A
IO$_ACPCONTROL                  = 00000038                LOSTPOS                           000009ED R     03
IO$_AVAILABLE                   = 00000011                MASKH                           = 00000008
IO$_CREATE                      = 00000033                MASKL                           = 04000000
IO$_DEACCESS                    = 00000034                MBA$L_AS                        = 00000410
IO$_DELETE                      = 00000035                MBA$L_BCR                       = 00000010
IO$_DRVCLR                      = 00000004                MBA$L_CR                        = 00000004
IO$_ERASETAPE                   = 00000006                MBA$L_CSR                       = 00000000
IO$_MODIFY                      = 00000036                MBA$L_ERB                       = 00000400
IO$_MOUNT                       = 00000039                MBA$L_MAP                       = 00000800
IO$_NOP                         = 00000000                MBA$L_SR                        = 00000008
IO$_PACKACK                     = 00000008                MBA$L_VAR                       = 0000000C
IO$_READLBLK                    = 00000021                MBA$M_CR_ABORT                  = 00000002
IO$_READPBLK                    = 0000000C                MBA$M_CR_IE                     = 00000004
IO$_READPRESET                  = 00000019                MBA$M_CR_INIT                   = 00000001
IO$_READVBLK                    = 00000031                MBA$M_ERROR                     = 000E5FFF
IO$_RECAL                       = 00000003                MBA$M_SR_ATTN                   = 00010000
IO$_REWIND                      = 00000024                MBA$M_SR_DLT                    = 00000800
IO$_REWINDOFF                   = 00000022                MBA$M_SR_DTABT                  = 00001000
IO$_SENSECHAR                   = 0000001B                MBA$M_SR_DTCOMP                 = 00002000
IO$_SENSEMODE                   = 00000027                MBA$M_SR_ERCONF                 = 00000008
IO$_SETCHAR                     = 0000001A                MBA$M_SR_INVMAP                 = 00000010
IO$_SETMODE                     = 00000023                MBA$M_SR_ISTO                   = 00000002
IO$_SKIPFILE                    = 00000025                MBA$M_SR_MAPPE                  = 00000020
IO$_SKIPRECORD                  = 00000026                MBA$M_SR_MBEXC                  = 00000080
IO$_SPACEFILE                   = 00000002                MBA$M_SR_MCPE                   = 00020000
IO$_SPACERECORD                 = 00000009                MBA$M_SR_MDPE                   = 00000040
IO$_UNLOAD                      = 00000001                MBA$M_SR_MXF                    = 00000100
IO$_VIRTUAL                     = 0000003F                MBA$M_SR_NED                    = 00040000
IO$_WRITECHECK                  = 0000000A                MBA$M_SR_PGE                    = 00080000
IO$_WRITELBLK                   = 00000020                MBA$M_SR_RDS                    = 00000004
IO$_WRITEMARK                   = 0000001C                MBA$M_SR_RDTO                   = 00000001
IO$_WRITEOF                     = 00000028                MBA$M_SR_SPE                    = 00004000
IO$_WRITEPBLK                   = 0000000B                MBA$M_SR_WCKLWR                 = 00000200
IO$_WRITEVBLK                   = 00000030                MBA$M_SR_WCKUPR                 = 00000400
IOC$CANCELIO                      ********    X     03    MBA$V_SR_NED                    = 00000012
IOC$DIAGBUFILL                    ********    X     03    MEDIA_ID_TE16                   = 6D285010
IOC$LOADMBAMAP                    ********    X     03    MEDIA_ID_TU45                   = 6D29502D
IOC$MNTVER                        ********    X     03    MEDIA_ID_TU77                   = 6D29504D
IOC$RELCHAN                       ********    X     03    MIN_RECORD                      = 0000000E
IOC$RELSCHAN                      ********    X     03    MMG$GL_SPTBASE                    ********    X     03
IOC$REQCOM                        ********    X     03    MSG$_DEVOFFLIN                    ********    X     03
IOC$REQPCHANL                     ********    X     03    MT$CHECK_ACCESS                   ********    X     03
IOC$REQSCHANL                     ********    X     03    MT$K_NRZI_800                   = 00000003
IOC$RETURN                        ********    X     03    MT$K_PE_1600                    = 00000004
IOC$WFIKPCH                       ********    X     03    MT$M_BOT                        = 00010000
IOC$WFIRLCH                       ********    X     03    MT$M_DENSITY                    = 00001F00
IRP$L_MEDIA                     = 00000038                MT$M_EOF                        = 00020000
IRP$L_SVAPTE                    = 0000002C                MT$M_EOT                        = 00040000
IRP$L_WIND                      = 00000018                MT$M_FORMAT                     = 000000F0
```

```
MT$M_HWL            = 00080000          MT_MR                 0000000C
MT$M_LOST           = 00100000          MT_SN                 00000020
MT$M_PARITY         = 00000008          MT_TC                 00000024
MT$S_DENSITY        = 00000005          MT_TC_M_EPAR        = 00000008
MT$S_FORMAT         = 00000004          MT_TC_S_DEN         = 00000003
MT$V_DENSITY        = 00000008          MT_TC_S_FSEL        = 00000004
MT$V_FORMAT         = 00000004          MT_TC_V_DEN         = 00000008
MT$V_PARITY         = 00000003          MT_TC_V_FSEL        = 00000004
MT_AS                 00000010          NOP                   000003E9 R     03
MT_CC                 0000001C          NORXIT                00000627 R     03
MT_CS1                00000000          NRZI                = 00000003
MT_CS1_M_GO         = 00000001          PACKACK               000003DC R     03
MT_DS                 00000004          PE                  = 00000004
MT_DS_M_BOT         = 00000002          POSIT                 00000BF8 R     03
MT_DS_M_PES         = 00000020          PR$_IPL               ********   X   03
MT_DS_M_TM          = 00000004          READDATA              00000668 R     03
MT_DS_V_BOT         = 00000001          READDATAR             00000784 R     03
MT_DS_V_EOT         = 0000000A          READPRESET            00000467 R     03
MT_DS_V_MOL         = 0000000C          RECAL                 00000C33 R     03
MT_DS_V_PES         = 00000005          RESETXFR              00000E67 R     03
MT_DS_V_PIP         = 0000000D          RESTART               00000B48 R     03
MT_DS_V_SLA         = 00000000          RETREG                00000D36 R     03
MT_DS_V_SSC         = 00000006          RETRY                 000009F1 R     03
MT_DS_V_TM          = 00000002          REWIND                0000046F R     03
MT_DS_V_WRL         = 0000000B          SENSECHAR             000003E9 R     03
MT_DT                 00000018          SETCHAR               000003F0 R     03
MT_ER                 00000008          SETDEN                00000CE8 R     03
MT_ER_M_COR         = 00008000          SETEOF                00000651 R     03
MT_ER_M_CPAR        = 00000008          SETEOV                0000062C R     03
MT_ER_M_CRC         = 00008000          SIZ...              = 00000001
MT_ER_M_CS          = 00000400          SPCFILFOR             00000490 R     03
MT_ER_M_DPAR        = 00000020          SPCFILREV             00000531 R     03
MT_ER_M_DTE         = 00001000          SPCRECFOR             0000055D R     03
MT_ER_M_FCE         = 00000200          SPCRECREV             000005F1 R     03
MT_ER_M_FMT         = 00000010          SS$_ABORT           = 0000002C
MT_ER_M_ILF         = 00000001          SS$_CTRLERR         = 00000054
MT_ER_M_ILR         = 00000002          SS$_DATACHECK       = 0000005C
MT_ER_M_INC         = 00000040          SS$_DATAOVERUN      = 00000838
MT_ER_M_ITM         = 00000400          SS$_DRVERR          = 0000008C
MT_ER_M_LRC         = 00000080          SS$_ENDOFFILE       = 00000870
MT_ER_M_NEF         = 00000800          SS$_ENDOFTAPE       = 00000878
MT_ER_M_NSG         = 00000100          SS$_ENDOFVOLUME     = 000009A0
MT_ER_M_OPI         = 00002000          SS$_FORMAT          = 000000BC
MT_ER_M_PEF         = 00000080          SS$_MEDOFL          = 000001A4
MT_ER_M_RMR         = 00000004          SS$_NONEXDRV        = 000001C4
MT_ER_M_UNS         = 00004000          SS$_NORMAL          = 00000001
MT_ER_M_VPE         = 00000040          SS$_OPINCOMPL       = 000002D4
MT_ER_V_CPAR        = 00000003          SS$_PARITY          = 000001F4
MT_ER_V_CRC         = 0000000F          SS$_TIMEOUT         = 0000022C
MT_ER_V_FMT         = 00000004          SS$_UNSAFE          = 0000023C
MT_ER_V_ITM         = 0000000A          SS$_VOLINV          = 00000254
MT_ER_V_LRC         = 00000007          SS$_WRITLCK         = 0000025C
MT_ER_V_NEF         = 0000000B          STSXIT                00000B34 R     03
MT_ER_V_OPI         = 0000000D          SYS$GL_OPRMBX         ********   X   03
MT_ER_V_UNS         = 0000000E          TESTR                 00000994 R     03
MT_ER_V_VPE         = 00000006          THRESHOLD           = 00000008
MT_FC                 00000014          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_UNSOLNT		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
------------------                         --------------
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                       36
_$255$DUA28:[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
```

MGRIVER
LIS

TFDRIVER
LIS

0117 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

TUDRIVER
LIS

XADRIVER
LIS

TSDRIVER
LIS