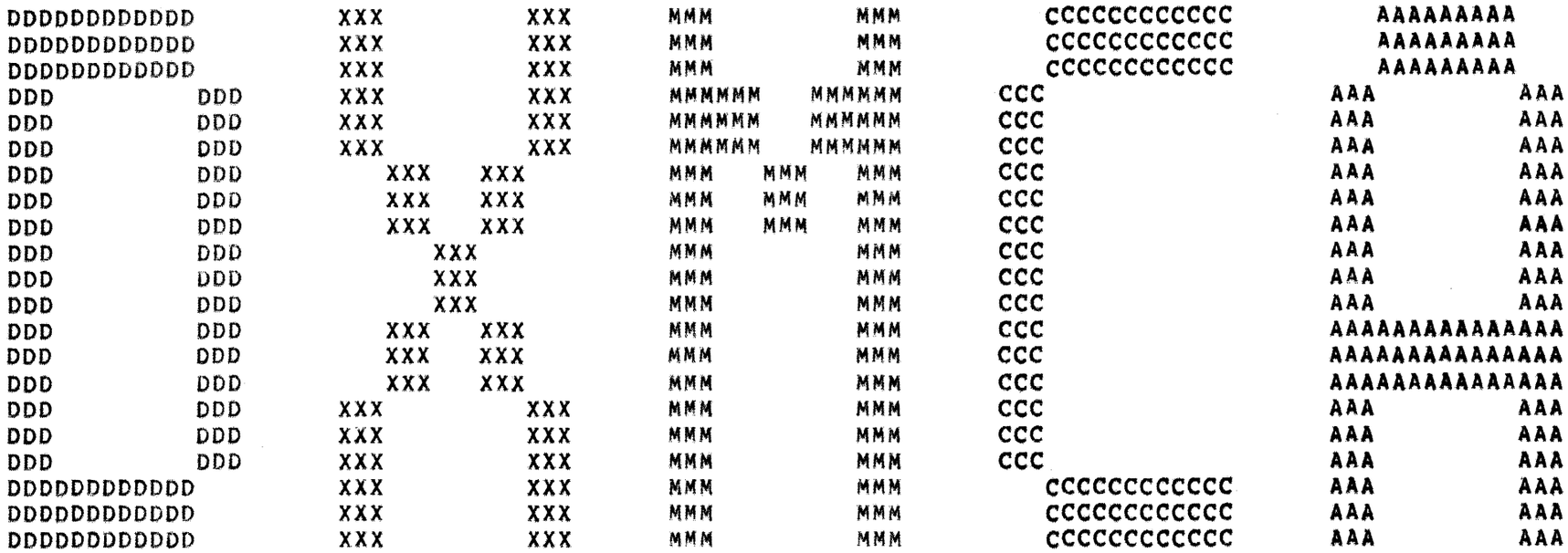
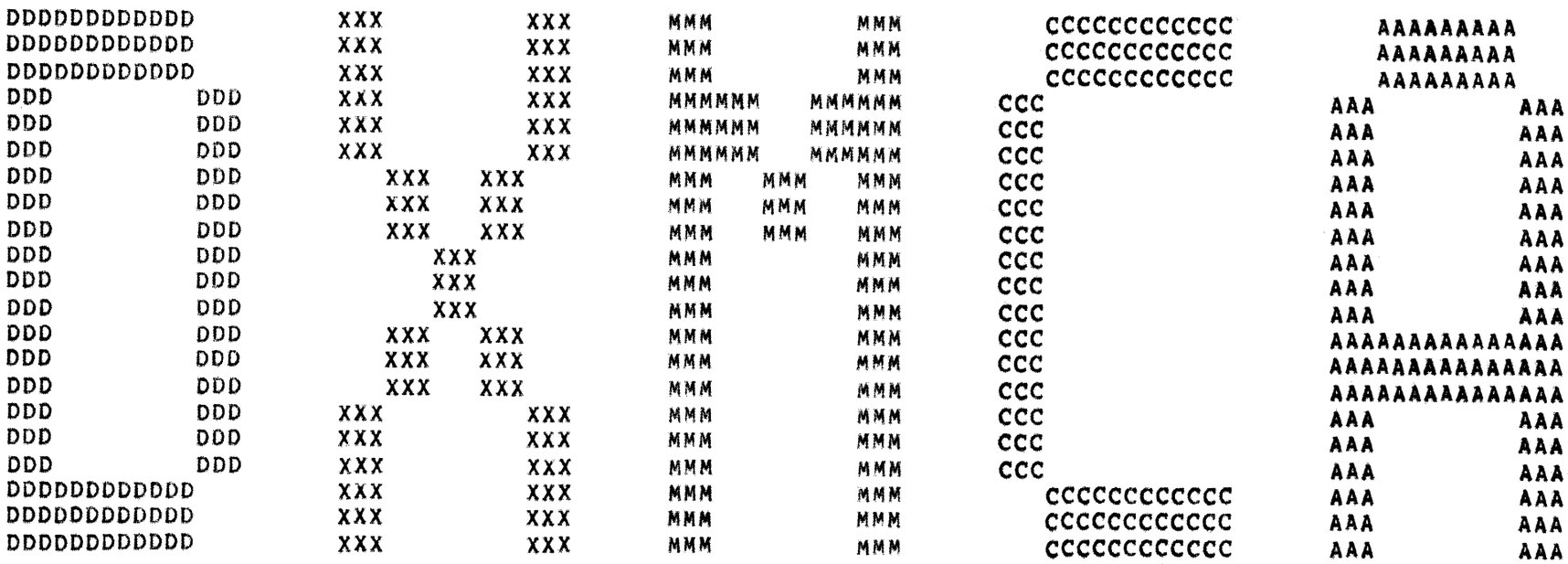
```
DDDDDDDDDDDD          XXX          XXX       MMM          MMM          CCCCCCCCCCCC          AAAAAAAAA
DDDDDDDDDDDD          XXX          XXX       MMM          MMM          CCCCCCCCCCCC          AAAAAAAAA
DDDDDDDDDDDD          XXX          XXX       MMM          MMM          CCCCCCCCCCCC          AAAAAAAAA
DDD          DDD      XXX          XXX       MMMMMM    MMMMMM    CCC                   AAA          AAA
DDD          DDD      XXX          XXX       MMMMMM    MMMMMM    CCC                   AAA          AAA
DDD          DDD      XXX          XXX       MMMMMM    MMMMMM    CCC                   AAA          AAA
DDD          DDD         XXX    XXX          MMM    MMM    MMM    CCC                   AAA          AAA
DDD          DDD         XXX    XXX          MMM    MMM    MMM    CCC                   AAA          AAA
DDD          DDD         XXX    XXX          MMM    MMM    MMM    CCC                   AAA          AAA
DDD          DDD            XXX               MMM          MMM    CCC                   AAA          AAA
DDD          DDD            XXX               MMM          MMM    CCC                   AAA          AAA
DDD          DDD            XXX               MMM          MMM    CCC                   AAA          AAA
DDD          DDD         XXX    XXX          MMM          MMM    CCC                   AAAAAAAAAAAAAAAA
DDD          DDD         XXX    XXX          MMM          MMM    CCC                   AAAAAAAAAAAAAAAA
DDD          DDD         XXX    XXX          MMM          MMM    CCC                   AAAAAAAAAAAAAAAA
DDD          DDD      XXX          XXX       MMM          MMM    CCC                   AAA          AAA
DDD          DDD      XXX          XXX       MMM          MMM    CCC                   AAA          AAA
DDD          DDD      XXX          XXX       MMM          MMM    CCC                   AAA          AAA
DDDDDDDDDDDD          XXX          XXX       MMM          MMM       CCCCCCCCCCCC       AAA          AAA
DDDDDDDDDDDD          XXX          XXX       MMM          MMM       CCCCCCCCCCCC       AAA          AAA
DDDDDDDDDDDD          XXX          XXX       MMM          MMM       CCCCCCCCCCCC       AAA          AAA

LLL                      SSSSSSSSSSS       TTTTTTTTTTTTTTT                                  111
LLL                      SSSSSSSSSSS       TTTTTTTTTTTTTTT                                  111
LLL                      SSSSSSSSSSS       TTTTTTTTTTTTTTT                                  111
LLL                   SSS                        TTT                                    111111
LLL                   SSS                        TTT                                    111111
LLL                   SSS                        TTT                                    111111
LLL                      SSSSSSSSS               TTT                                      111
LLL                      SSSSSSSSS               TTT                                      111
LLL                      SSSSSSSSS               TTT                                      111
LLL                               SSS            TTT                                      111
LLL                               SSS            TTT                                      111
LLL                               SSS            TTT                                      111
LLL                               SSS            TTT          ......                      111
LLL                               SSS            TTT          ......                      111
LLL                               SSS            TTT          ......                      111
LLLLLLLLLLLLLLL       SSSSSSSSSSS               TTT          ......                111111111
LLLLLLLLLLLLLLL       SSSSSSSSSSS               TTT          ......                111111111
LLLLLLLLLLLLLLL       SSSSSSSSSSS               TTT          ......                111111111
```

START Job DX20CA Req #100 for DEUFEL.TL Date 3-Mar-82 14:29:43 Monitor: 2102 TOPS-20 Development System, TOPS-20 Moni *START*
File KNET:<DEUFEL.TL>DXMCA.LST.1, created: 15-Feb-82 18:36:35, printed: 3-Mar-82 14:34:16
Job parameters: Request created: 3-Mar-82 14:17:18 Page limit:678 Forms:NORMAL Account:LOW
File parameters: Copy: 1 of 2 Spacing:SINGLE File format:ASCII Print mode:ASCII



START Job DX20CA Req #100 for DEUFEL.TL Date 3-Mar-82 14:29:43 Monitor: 2102 TOPS-20 Development System, TOPS-20 Moni *START*
File KNET:<DEUFEL.TL>DXMCA.LST.1, created: 15-Feb-82 18:36:35, printed: 3-Mar-82 14:41:55
Job parameters: Request created: 3-Mar-82 14:17:18 Page limit:678 Forms:NORMAL Account:LOW
File parameters: Copy: 2 of 2 Spacing:SINGLE File format:ASCII Print mode:ASCII



START Job DX20CA Req #100 for DEUFEL,TL Date 3-Mar-82 14:29:43 Monitor: 2102 TOPS-20 Development System, TOPS-20 Moni *START*
File KNET:<DEUFEL.TL>DXMCA.LST.1, created: 15-Feb-82 18:36:35, printed: 3-Mar-82 14:42:01
Job parameters: Request created: 3-Mar-82 14:17:18 Page limit:678 Forms:NORMAL Account:LOW
File parameters: Copy: 2 of 2 Spacing:SINGLE File format:ASCII Print mode:ASCII

```
                000004          EDIT=4
                000007          VERSION=7

                                XLIST
                                LIST



                                TITLE   DXMCA - DX20-V100 MAGNETIC TAPE SUBSYSTEM MICRO-CODE  VERSION 4.7


                                ^

```

```
                        SUBTTL  CONFIGURATION PARAMETERS

                        ;EDITS 1-4 WERE DONE FOR REV 0.2
                        ;
                        ;[1]    EDIT 1  3/22/79  FIXES WRONG MODE SETTING
                        ;               FOR 556 BPI 7 TRACK ODD PARITY  W.FARBER
                        ;[2]    EDIT 2  30-JUL-79  FIXES WRONG FLAG SETTING FOR 'STARTC' STARTING
                        ;               ADDRESS.
                        ;[3]    EDIT 3  4-OCT-79  ADDS 3-4 SEC TIMEOUT IF READ TRANSFER DOES NOT
                        ;               BEGIN.
                        ;[4]    EDIT 4  INHIBITS RUNNING OF TESTS 21,22 UNLESS MICROCODE IS STARTED
                        ;               AT STARTB,C,D.
                        ;
                        ;EDIT 5 WAS DONE FOR REV 0.3
                        ;
                        ;[5]    EDIT 5  RESTORED 'FLAGS' FROM MASSBUS REGISTER INTO AC4, WHICH WAS
                        ;               DESTROYED BY CODE ADDED FOR EDIT [3].
                        ;
                        ;EDIT 6 WAS DONE FOR REV 0.4
                        ;
                        ;[6]    EDIT 6  CHANGED CODE TO INDICATE CHAINING ON CONTROL UNIT INITIATED
                        ;               SEQUENCE WHEN DEVICE END IS NOT PRESENTED.
                        ;
                        ;EDIT 8 WAS DONE FOR REV 0.6
                        ;
                        ;[8]    EDIT 8  CHANGED SEQUENCE OF LOADING ASYNCH STATUS TO LOADING DRIVE
                        ;               NUMBER FIRST THEN STATUS.
                        ;
                        ;EDIT 9 WAS DONE FOR REV 0.7
                        ;[9]    EDIT 9  ADDED CODE TO CHECK FOR GO BIT BEING SET BEFORE SETTING ATA
                        ;               TO TELL HOST THAT ASYNCH STATUS IS AVAILABLE.  FIXES PI 5
                        ;               INTERRUPT PROBLEM


                        ;THIS MICRO-CODE IS DESIGNED FOR USE ONLY IN THE DX20 PROGRAMMED DATA
                        ;ADAPTER WHEN CONNECTED TO A TX01 OR TX02 MAGNETIC TAPE CONTROL UNIT


                        ;DRIVE TYPE OF DX20 IN CONFIGURATION

050060                          TYPE=050060                     ;ONLY THIS DRIVE TYPE WILL BE PERMITTED

                        ;THE FOLLOWING IS A KEY WORD USED TO IDENTIFY THE PRESENCE OF THIS
                        ;MICRO-CODE IN THE DX20 CONTROL STORE RAM

000100                          KEYWRD=100                      ;VERSION 100 MICRO-CODE




                        SEARCH  DX20CA                          ;READ THE CROSS ASSEMBLER
```

```
                        SUBTTL  MASSBUS REGISTER BIT DEFINITIONS


                        ;STATUS & CONTROL REGISTER 1

000000          MPSCR0= 0                       ;REGISTER NAME
000001                  RUN=    1               ;RUN LINE FROM RH20
000002                  GO=     2               ;GO BIT
000174                  FN=   174               ;FUNCTION CODE
000004                   F0=    4               ;FUNCTION BIT 0
000010                   F1=   10               ;FUNCTION BIT 1
000020                   F2=   20               ;FUNCTION BIT 2
000040                   F3=   40               ;FUNCTION BIT 3
000100                   F4=  100               ;FUNCTION BIT 4
000200                  WCLK=  200              ;WRITE CLOCK LINE FROM RH20

                        ;STATUS AND CONTROL REGISTER 2

000001          MPSCR1= 1                       ;REGISTER NAME
000001                  DONE=   1               ;DATA TRANSFER DONE (READ)
000001                  EBL=    1               ;SET EBL (WRITE)
000002                  EXC=    2               ;EXCEPTION LINE FROM RH20 (READ)
000002                  CLRGO=  2               ;SET TO CLEAR GO (WRITE)
000004                  CMPERR= 4               ;COMPOSITE ERROR FLAG (READ)
000004                  START=  4               ;START A DATA TRANSFER (WRITE)
000010                  DTD=   10               ;DATA TO DEVICE
000020                  OCC=   20               ;OCCUPIED
000040                  ILF=   40               ;ILLEGAL FUNCTION
000100                  MPERR= 100              ;MICRO-PROCESSOR DETECTED ERROR FLAG
000200                  ATA=   200              ;ATTENTION

                ;ERROR CODE REGISTER

000002          MPECR= 2                        ;REGISTER NAME


                ;DRIVE TYPE REGISTER

000003          MPDTR= 3                        ;REGISTER NAME


                ;HARDWARE VERSION REGISTER

000004          MPHVR= 4                        ;REGISTER NAME

```

```
                        ;DATA BUFFER REGISTER 0

000005          MPDB0= 5                                ;REGISTER NAME


                        ;DATA BUFFER REGISTER 1

000006          MPDB1= 6                                ;REGISTER NAME


                        ;DATA BUFFER REGISTER 2

000007          MPDB2= 7                                ;REGISTER NAME


000003                 DB=      3                       ;DATA BUFFER BITS 16 AND 17
000004                 DBPAR=   4                       ;PARITY BIT
000010                 DBPARE=  10                      ;PARITY ERROR (READ)
000020                 DBEVEN=  20                      ;DATA BUFFER EVEN PARITY CONTROL

                        ;GENERAL PURPOSE REGISTERS

000010          STINDX= 10                              ;STATUS INDEX
000177                 INDEX=  177                      ;INDEX CODE
000200                 STATRQ= 200                      ;STATUS REQUEST FLAG
000011          STBYTE= 11                              ;ENDING STATUS BYTE

000012          DR=     12                              ;DRIVE NUMBER FROM HOST
000013          MODE=   13                              ;DRIVE MODE (DENSITY & FORMAT)

000014          DUMPSZ= 14                              ;SIZE OF EXTENDED STATUS TABLE
000015          SPARE=  15                              ;NOT USED

000016          TIEBYT= 16                              ;TRACK-IN-ERROR BYTE FROM HOST
000017          FLAGS=  17                              ;FLAG REGISTER
000001                 SNDTIE=   1                      ;SEND TIE BYTE BEFORE READ COMMAND
000002                 FSENSE=   2                      ;FORCE SENSE BYTE READ AT TERMINATION
000004                 SUPRES=   4                      ;SUPPRESS SENSE ON ERROR

000020          ASYCDR= 20                              ;DRIVE PRESENTING ASYNC STATUS
000021          ASYCST= 21                              ;ASYNC STATUS FROM DRIVE

000023          CNTH=   23                              ;BYTE / BLOCK COUNTER
000022          CNTL=   22

000025          XSTAT0= 25                              ;EXTENDED STATUS BYTE 0
000024          XSTAT1= 24                              ;EXTENDED STATUS BYTE 1
000027          XSTAT2= 27                              ;EXTENDED STATUS BYTE 2
000026          XSTAT3= 26                              ;EXTENDED STATUS BYTE 3
```

```
                        ;MP STATUS REGISTER

000036          MPSTAT=36                       ;REGISTER NAME
000001                  INT0=    1              ;INTERRUPT LINE 0
000002                  INT1=    2              ;INTERRUPT LINE 1
000004                  INT2=    4              ;INTERRUPT LINE 2
000010                  INT3=   10              ;INTERRUPT LINE 3
000020                  C=      20              ;CARRY BIT
000040                  Z=      40              ;ZERO BIT

                        ;I/O BANK SELECT REGISTER

000037          IOSEL=37                        ;REGISTER NAME
000007                  INADR=   7              ;INPUT BANK ADDRESS
000070                  OUTADR=  70             ;OUTPUT BANK ADDRESS
000100                  SPRES=  100             ;STACK POINTER RESET
000200                  INIT=   200             ;INITIALIZE

                        ;INTERFACE SELECTION CODES

000001                  MBIN=   01              ;MASSBUS INTERFACE READ ONLY
000010                  MBOUT=  10              ;                  WRITE ONLY
000011                  MB=     11              ;                  READ AND WRITE
000002                  DPIN=   02              ;DATA PATH READ ONLY
000020                  DPOUT=  20              ;          WRITE ONLY
000022                  DP=     22              ;          READ AND WRITE
000003                  CHNIN=  03              ;CHANNEL INTERFACE READ ONLY
000030                  CHNOUT= 30              ;                  WRITE ONLY
000033                  CHN=    33              ;                  READ AND WRITE
```

```
                                 CHANNEL BUS INTERFACE REGISTER DEFINITIONS

                                 SUBTTL  CHANNEL BUS INTERFACE REGISTER DEFINITIONS

                                 ;CONTROL AND STATUS REGISTER 0

            000000               CSR0=    0                          ;REGISTER NAME
            000001                      CLRSLV=   1                 ;WRITE TO CLEAR SLAVE SELECTED
            000002                      CLRERS=   2                 ;WRITE TO CLEAR ERRORS
            000001                      SLVSEL=   1                 ;SLAVE SELECTED
            000002                      BUS0PE=   2                 ;BUS 0 PARITY ERROR
            000004                      BUS1PE=   4                 ;BUS 1 PARITY ERROR
            000010                      MKPE=    10                 ;MARK BUS PARITY ERROR
            000020                      UPBE=    20                 ;MICRO-BUS PARITY ERROR
            000040                      DPPE=    40                 ;DATA PATH PARITY ERROR
            000100                      TIMOUT=  100                ;TIME OUT ERROR
            000200                      EXFER=   200                ;END TRANSFER

                                 ;CONTROL AND STATUS REGISTER 1

            000001               CSR1=    1                          ;REGISTER NAME
            000001                      CHANL=    1                 ;CHANNEL MODE
            000002                      ONLINE=   2                 ;CHANNEL BUS ENABLE
            000004                      LOOPEN=   4                 ;DIAGNOSTIC LOOP ENABLE
            000010                      MOD360=  10                 ;360 TYPE BUS MODE
            000020                      EXTBUS=  20                 ;EXTENDED BUS ENABLE
            000040                      EVPAR=   40                 ;EVEN PARITY CONTROL
            000100                      DIHISP= 100                 ;DIAGNOSTIC HIGH SPEED
            000200                      SPEN=   200                 ;SCRATCH PAD ENABLE

                                 ;TAG OUT REGISTER 0

            000002               TOR0=    2                          ;REGISTER NAME
            000001                      CMDOUT=   1                 ;COMMAND OUT
            000002                      SELOUT=   2                 ;SELECT OUT
            000004                      TMREN=    4                 ;TIMER ENABLE AND TRIGGER
            000010                      HLDOUT=  10                 ;HOLD OUT
            000020                      ADROUT=  20                 ;ADDRESS OUT
            000040                      MTROUT=  40                 ;METER OUT
            000100                      CLKOUT= 100                 ;CLOCK OUT
            000200                      SRVOUT= 200                 ;SERVICE OUT

                                 ;TAG OUT REGISTER 1

            000003               TOR1=    3                          ;REGISTER NAME
            000001                      TOSOUT=   1
            000002                      TODOUT=   2
            000004                      DIMUX=    4
            000010                      DISACK=  10
            000020                      DATOUT=  20                 ;DATA OUT
            000040                      SUPOUT=  40                 ;SUPPRESS OUT
            000100                      DGOUT= 100
            000200                      OPLOUT= 200                 ;OPERATIONAL OUT
```

```
                                ;CHANNEL BUS INTERFACE REGISTER DEFINITIONS

                                ;TAG IN REGISTER 0

                        000004  TAGIN0=  4                              ;REGISTER NAME
                        000001          STAIN=     1                    ;STATUS IN
                        000002          SELIN=     2                    ;SELECT IN
                        000004          MTRIN=     4                    ;METER IN
                        000010          TOHOUT=   10
                        000020          ADRIN=    20                    ;ADDRESS IN
                        000040          MK1IN=    40                    ;MARK 1 IN
                        000100          MK0IN=   100                    ;MARK 0 IN
                        000200          OPLIN=   200                    ;OPERATIONAL IN

                                ;TAG IN REGISTER 1

                        000005  TAGIN1=  5                              ;REGISTER NAME
                        000001          SPA0=      1
                        000002          SPA1=      2
                        000004          SPA2=      4
                        000010          SPA3=     10
                        000020          DATIN=    20                    ;DATA IN
                        000040          REQIN=    40                    ;REQUEST IN
                        000100          DISIN=   100                    ;DISCONNECT IN
                        000200          SRVIN=   200                    ;SERVICE IN
```

```
                        ;DATA REGISTER 0

000006          DRLO=   6                       ;REGISTER NAME

                        ;BUS IN REGISTER 0

000007          CBILO=  7                       ;REGISTER NAME

                        ;SCRATCH PAD DATA REGISTER 0

000010          SPDALO= 10                      ;REGISTER NAME

                        ;BUS OUT REGISTER 0

000011          BORLO=  11                      ;REGISTER NAME

                        ;DATA REGISTER 1

000012          DRHI=   12                      ;REGISTER NAME

                        ;BUS IN REGISTER 1

000013          CBIHI=  13                      ;REGISTER NAME

                        ;SCRATCH PAD DATA REGISTER 1

000014          SPDAHI= 14                      ;REGISTER NAME

                        ;BUS OUT REGISTER 1

000015          BORHI=  15                      ;REGISTER NAME

                        ;CONTROL UNIT STATUS REGISTER

000016          CUSTAT= 16                      ;REGISTER NAME
```

```
                        SUBTTL  DATA PATH REGISTER DEFINITIONS

                        ;STATUS REGISTERS

000000          REG0=   0                       ;REGISTER NAME
000001                  UBPEFG=   1             ;MICROBUS PARITY ERROR FLAG
000002                  DPPEFG=   2             ;DATA PATH PARITY ERROR FLAG
000004                  BCOVF=    4             ;BYTE COUNT OVERFLOW FLAG
000010                  MCOVF=   10             ;MASSBUS COUNTER OVERFLOW FLAG

000001          REG1=   1                       ;REGISTER NAME
000001                  DXHISP=   1             ;DX HIGH SPEED
000002                  BCLKEN=   2             ;BASE CLOCK ENABLE
000004                  DSLVRQ=   4             ;DIAGNOSTIC SLAVE REQUEST
000010                  DMSTRQ=  10             ;DIAGNOSTIC MASTER REQUEST
000020                  SLVACK=  20             ;SLAVE ACK
000040                  MSTACK=  40             ;MASTER ACK
000100                  SLVRQ=  100             ;SLAVE REQUEST
000200                  MSTRQ=  200             ;MASTER REQUEST

000002          REG2=   2                       ;REGISTER NAME
000001                  RMADR8=   1             ;ROM ADR BIT 8
000002                  SEBCOV=   2             ;SLAVE END ON BC OVERFLOW ENABLE
000004                  MEMCOV=   4             ;MASTER END ON MC OVERFLOW ENABLE
000010                  MEONFE=  10             ;MASTER END ON FORMATTER END ENABLE

000003          REG3=   3                       ;REGISTER NAME
000002                  NSEXFR=   2             ;NOT SLAVE END TRANSFER
000004                  NFEXFR=   4             ;NOT FORMATTER END TRANSFER
000010                  NMEXFR=  10             ;NOT MASTER END TRANSFER

                        ;DATA PATH COUNTERS

000004          MCLO=   4                       ;MASSBUS COUNTER (LOW BYTE)
000005          MCHO=   5                       ;   "        "    (HIGH BYTE)
000006          BCLO=   6                       ;BYTE COUNTER (LOW BYTE)
000007          BCHO=   7                       ; "      "    (HIGH BYTE)

                        ;FORMATTER ROM START ADDRESS REGISTER

000010          DFRMAD= 10                      ;REGISTER NAME

                        ;HIGH SPEED INITIALIZE SIGNAL

000013          HSDPIN= 13                      ;WRITE OF THIS REGISTER TO INIT (DATA IGNORED)
```

```
                        SUBTTL  MISCELLANEOUS DEFINITIONS

                        ;A MACRO TO ESTABLISH A TABLE IN CRAM IN CONSECUTIVE LOCATIONS
                        ;IN A SINGLE PAGE.  DEFINES A NAME FOR THE TABLE AND SPECIFIES THE SIZE.

                        DEFINE  TABLE (NAME,SIZE),<
                                IFN <<.-8>-<<.+SIZE-1>-8>>,<.LOC <.!377>+1>
                                NAME:                           ;DEFINE TABLE
                        >

                        ;A MACRO TO DEPOSIT ZEROS INTO MEMORY TO FILL THE CURRENT PAGE
                        ;THE ZERO FILL WORDS ARE NOT LISTED

                        DEFINE  .MNEXT,<
                                XLIST
                                BLOCK <400-<.&377>>
                                LIST
                        >
```

```
                                SUBTTL  ERROR CODE DEFINITIONS

                                ;THE FOLLOWING CODES WILL BE PLACED IN THE ERROR CODE REGISTER
                                ;AND THE MP ERROR FLAG AND ATTENTION WILL BE SET WHEN
                                ;THE CORRESPONDING ERRORS ARE DETECTED.

                                ;THE CODE IS BROKEN INTO TWO EIGHT BIT FIELDS:
                                ;  THE LOW ORDER FOUR BITS DETERMINE A CLASS
                                ;  THE HIGH ORDER FOUR BITS DEFINE THE PARTICULAR ERROR IN THE CLASS

                                ;CLASS 1 - UNUSUAL DEVICE STATUS

                                ;STATUS BYTE IS PRESENTED IN REGISTER 20
                                ;REGISTERS ARE LOGGED AND SENSE BYTES ARE READ(UNLESS BUSY IS SET
                                ;  IN STATUS BYTE)
                                ;IF READ FUNCTION, NUMBER OF BYTES READ IN ERROR RECORD ARE REPORTED

                    000001      UDS.FN= 1!0_4                       ;STATUS IS FROM FINAL STATUS SEQUENCE
                    000021      UDS.IN= 1!1_4                       ;STATUS IS FROM INITIAL STATUS SEQUENCE

                                ;CLASS 2 - SHORT RECORD

                                ;STATUS BYTE IS PRESENTED IN REGISTER 20
                                ;REGISTERS ARE LOGGED AND SENSE BYTES ARE READ
                                ;ACTUAL LENGTH OF RECORD IS PRESENTED IN REGISTER 5

                    000002      SHREC.= 2

                                ;CLASS 3 - LONG RECORD

                                ;STATUS BYTE IS PRESENTED IN REGISTER 20
                                ;REGISTERS ARE LOGGED AND SENSE BYTES ARE READ
                                ;ACTUAL LENGTH OF RECORD, MODULO 200000(8), IS PRESENTED IN REGISTER 5

                    000003      LGREC.= 3

                                ;CLASS 4 - DEVICE SELECTION ERROR

                                ;REGISTERS ARE LOGGED
                                ;STATUS AND SENSE BYTES ARE RETURNED ZERO

                    000004      DSE.=   4

                                ;CLASS 5 - RECOVERABLE ERROR (TAPE MOTION)

                                ;REGISTERS ARE LOGGED, SENSE BYTES ARE READ
                                ;ACTUAL LENGTH OF RECORD IS PRESENTED IN REGISTER 5, IF READ

                    000005      REM.DP= 5!0_4                       ;DATA PATH PARITY ERROR
                    000025      REM.SP= 5!1_4                       ;ENDING STATUS BYTE PARITY ERROR
```

```
                        ;CLASS 6 - RECOVERABLE ERROR (NO TAPE MOTION)

                        ;REGISTERS ARE LOGGED
                        ;SELECTIVE RESET IS PERFORMED ON DRIVE (EXCEPT RE.NRN)
                        ;A ZERO STATUS BYTE IS PRESENTED

000006          RE.AM=  6!0_4                   ;ADDRESS MISMATCH
000026          RE.AP=  6!1_4                   ;ADDRESS PARITY ERROR
000046          RE.SP=  6!2_4                   ;INITIAL STATUS BYTE PARITY ERROR
000066          RE.NRN= 6!3_4                   ;RUN WAS NOT RECEIVED FROM RH20

                        ;CLASS 7 - NON-RECOVERABLE ERROR

                        ;REGISTERS ARE LOGGED
                        ;SELECTIVE RESET IS PERFORMED ON DRIVE
                        ;A ZERO STATUS BYTE IS PRESENTED

000007          NRE.DP= 7!0_4                   ;DATA PATH DETECTED MICRO-BUS PARITY ERROR
000027          NRE.CP= 7!1_4                   ;CHANNEL BUS DETECTED MICRO-BUS PARITY ERROR
000047          NRE.TO= 7!2_4                   ;CHANNEL BUS HANDSHAKE TIME-OUT
000067          NRE.DC= 7!3_4                   ;DEVICE ASSERTED DISCONNECT IN
000107          NRE.RT= 7!4_4                   ;TIME-OUT OF READ COMMAND

                        ;CLASS 8 - FATAL

                        ;DX20 IS HALTED BY A STACK POINTER UNDERFLOW ERROR
                        ;  AFTER SETTING ERROR CODE AND ERROR FLAG

000010          F.WDT=  8!0_4                   ;WRONG DRIVE TYPE FOR VERSION OF MICRO-CODE
000030          F.APCU= 8!1_4                   ;ADDRESS PARITY ERROR DURING CU INITIATED SEQUENCE
000050          F.SPCU= 8!2_4                   ;STATUS BYTE PARITY ERROR DURING CU INITIATED SEQ.
000070          F.FAIL= 8!3_4                   ;MASSBUS CONTROLLER POWER FAILED
000110          F.INT0= 8!4_4                   ;PROCESSOR INTERRUPTED FROM INTERFACE 0?????

                        ;CLASS 9 - DIAGNOSTIC ERROR

                        ;MICROPROCESSOR WILL LOOP ON ERROR IF STARTED AT 'BEGIN2' OR 'BEGIN3', ELSE
                        ; WILL HALT BY A STACK POINTER UNDERFLOW ERROR
                        ;  IN THE FORMER CASE, THE ERROR CODE AND ERROR FLAG WILL BE SET AND EXTENDED
                        ;  STATUS REGISTER 0 WILL CONTAIN THE ERROR NUMBER.  IN THE LATTER CASE, THE
                        ;  SAME WILL BE DONE WITH ATTENTION ALSO BEING ASSERTED.


000011          D.FAIL= 9
```

```
                                SUBTTL  START-UP

                                        .INIT   ^                       ;INITIALIZE THE ASSEMBLER

                                        SALL
000000  0 016004 07 0004

000001  0 100012 4 0 0012       BEGIN:  JMP     STARTA                  ;INITIALIZE START, NORMAL ENTRY
000002  0 100014 4 0 0014       BEGIN1: JMP     STARTB                  ;NO IDLE LOOP DIAG START
000003  0 100016 4 0 0016       BEGIN2: JMP     STARTC                  ;RUN IDLE LOOP DIAGS, LOOP ON ERROR START
000004  0 100020 4 0 0020       BEGIN3: JMP     STARTD                  ;RUN DIAGS ONLY START, LOOP ON ERROR
000005  0 100012 4 0 0012       BEGIN4: JMP     STARTA
000006  0 100024 4 0 0024       BEGIN5: JMP     STARTF                  ;RESTART ADDRESS

                                ;THE FOLLOWING ARE FIXED LOCATIONS USED TO IDENTIFY
                                ;THIS CODE IN THE DX20 CONTROL STORE RAM

000007  0 050060 050060                DATA    TYPE                    ;THE DRIVE TYPE OF THE DX20 CONTROLLER
000010  0 000100 000100                DATA    KEYWRD                  ;A KEY WORD IDENTIFYING THE MICRO-CODE
000011  0 000000 000000                DATA    0                       ;THIS WORD WILL ALWAYS LOAD WITH ZERO.
                                                                        ;THE MONITOR BOOTSTRAP WILL WRITE THIS
                                                                        ;LOCATION TO NON-ZERO AT LOAD TIME.


                                ;THE WORKING MEMORY LOCATION 'DFLAGS' IS USED TO HOLD DIAGNOSTIC FLAG BITS.
                                ;THE FACT THAT ALL BITS ARE ZEROED WHEN THE MICROCODE IS STARTED AT 'STARTA'
                                ;IS USED TO DETERMINE WHETHER OR NOT TESTS 21 AND 22 GET RUN AT STARTUP.
                                ;THE BITS ARE:          BIT 0 = SET IF DIAGS ARE NOT TO BE RUN IN IDLE LOOP
                                ;                       BIT 1 = SET IF LOOPING ON DIAG ERROR
                                ;                       BIT 4 = SET IF RUNNING DIAGS ONLY
                                ;                       BIT 7 = SET IF LOOP ON ERROR DESIRED

000012  0 002000 0 1 0 000      STARTA: LDBR    0                       ;SET TO RUN DIAGS IN IDLE LOOP, HALT ON ERROR
000013  0 100021 4 0 0021               JMP     STARTX
000014  0 002001 0 1 0 001      STARTB: LDBR    1                       ;SET TO NOT RUN DIAGS IN IDLE LOOP
000015  0 100021 4 0 0021               JMP     STARTX

                                ;EDIT [2] CORRECTS WRONG SETTING OF FLAG WORD.
                                ;[2] STARTC:    LDBR    201                     ;SET TO RUN DIAGS AND LOOP ON ERROR

000016  0 002200 0 1 0 200      STARTC: LDBR    200                     ;[2] SET TO RUN DIAGS AND LOOP ON ERROR
000017  0 100021 4 0 0021               JMP     STARTX
000020  0 002220 0 1 0 220      STARTD: LDBR    220                     ;SET TO RUN DIAGS ONLY AND LOOP ON ERROR
000021  0 000400 0 0 1 000      STARTX: LDMARX  0                       ;CLEAR MAR EXT BITS
000022  0 001001 0 0 2 001              LDMAR   DFLAGS                  ;SET MAR TO DLAGS ADDR
000023  0 070011 3 4 0 00 11            MOVB    MEM
000024  0 002161 0 1 0 161      STARTF: JUMP    CLENUP                  ;RUN DIAGS AND CLEAN UP ALL STATUS,
000025  0 160231 7 0 0 11 11
```

```
                                SUBTTL  IDLE LOOP

000026  0 000400 0 0 1 000      IDLE:   LDMARX  0               ;CLEAR MAR EXT BITS
000027  0 001003 0 0 2 003              LDMAR   IDLCNT          ;SET MAR TO IDLE WAIT CNR ADDR
000030  0 010100 0 4 0 100              LDMEM   100             ;SET WAIT TIME BEFORE RUNNING DIAGS
000031  0 002100 0 1 0 100      IDLED:  LDBR    SPRES           ;RESET THE STACK POINTER
000032  0 066371 3 3 0 17 11           MOVB    IOSEL
000033  0 002011 0 1 0 011      IDLED1: LDBR    MB              ;SELECT MASSBUS INTERFACE
000034  0 066371 3 3 0 17 11           MOVB    IOSEL

000035  0 116056 4 7 0056       IDLELP: JMPSUB  EXTREQ          ;CHECK IF AN EXTENDED STATUS REQUEST
000036  0 116112 4 7 0112               JMPSUB  CKASYC          ;CHECK FOR ASYNC STATUS TO PRESENT
000037  0 022000 1 1 0 00 00            DATI    MPSCR0,BR       ;READ FUNCTION CODE REGISTER
000040  0 014000 0 6 0 000              SHR                     ;SHIFT GO BIT TO BR0
000041  0 104147 4 2 0147               JMPB0   DOCMD           ;DO COMMAND IF GO IS SET
000042  0 103767 4 1 1767               JMPI    INTERUPT        ;CHECK ON INTERRUPTS
000043  0 000400 0 0 1 000              LDMARX  0               ;CLEAR MAR EXT BITS
000044  0 001001 0 0 2 001              LDMAR   DFLAGS          ;SET ADDR OF DIAG FLAGS LOC
000045  0 042011 2 1 0 00 11            MOVMEM  BR              ;PUT FLAGS INTO BR
000046  0 104035 4 2 0035               JMPB0   IDLELP          ;DON'T RUN DIAGS IF BIT 0 SET
000047  0 001003 0 0 2 003              LDMAR   IDLCNT          ;SET ADDR OF WAIT CNT
000050  0 052011 2 5 0 00 11            MOVMEM  AC0             ;GET WAIT CNT
000051  0 070007 3 4 0 00 07            DEC     AC0,MEM         ;DECREMENT WAIT CNT
000052  0 114054 4 6 0054               JMPZ    .+2             ;JUMP TO RUN DIAGS IF TIMED OUT
000053  0 100035 4 0 0035               JMP     IDLELP          ;STAY IN IDLE LOOP
000054  0 002244 0 1 0 244              JUMP    RNDIAG          ;GO RUN DIAGS
000055  0 161631 7 0 3 11 11
```

```
                                SUBTTL  EXTENDED STATUS REQUEST SERVICE

                                ;CHECK IF EXTENDED STATUS WAS REQUESTED

000056  0 022010 1 1 0 00 10   EXTREQ: DATI    STINDX,BR              ;READ INDEX REGISTER
000057  0 110061 4 4 0061              JMPB7   .+2                    ;SKIP IF REQUEST IS SET
000060  0 016000 0 7 0 000             RETURN                         ;NO

                                ;CHECK RANGE OF INDEX

000061  0 001003 0 0 2 003             LDMAR   IDLCNT                 ;RESET WAIT CNT FOR RUNNING DIAGS
000062  0 010100 0 4 0 100             LDMEM   100
000063  0 002223 0 1 0 223             LDBR    MAXIDX+200             ;GET MAXIMUM VALID CODE
000064  0 072011 3 5 0 00 11          MOVB    AC0                    ;MOVE MAX TO AC0
000065  0 022010 1 1 0 00 10          DATI    STINDX,BR              ;GET CODE AGAIN
000066  0 062016 3 1 0 00 16          TSB     AC0,BR                 ;SUBTRACT CODE FROM MAX
000067  0 110107 4 4 0107              JMPB7   BADIDX                 ;IF NEGATIVE RESULT, BAD INDEX VALUE RECEIVED
000070  0 032010 1 5 0 00 10          DATI    STINDX,AC0             ;READ INDEX CODE
000071  0 072005 3 5 0 00 05          SHLR    AC0                    ;MULTIPLY BY 4
000072  0 072005 3 5 0 00 05          SHLR    AC0
000073  0 000400 0 0 1 000             LDMARX  0                      ;SELECT MEMORY PAGE 0
000074  0 002052 0 1 0 052             LDBR    SBYT00                 ;GET START ADDRESS OF TABLE
000075  0 061000 3 0 2 00 00          ADB     AC0,MAR                ;ADD TO OBTAIN ADDRESS REQUESTED
000076  0 047531 2 3 3 05 11          MOVMEM  XSTAT0,I               ;LOAD EXTENDED STATUS REGISTERS
000077  0 047511 2 3 3 04 11          MOVMEM  XSTAT1,I
000100  0 047571 2 3 3 07 11          MOVMEM  XSTAT2,I
000101  0 046151 2 3 0 06 11          MOVMEM  XSTAT3
000102  0 032010 1 5 0 00 10          DATI    STINDX,AC0             ;READ INDEX AGAIN
000103  0 002177 0 1 0 177             LDBR    INDEX                  ;GET MASK OF INDEX BITS
000104  0 062013 3 1 0 00 13          LANDB   AC0,BR                 ;GET ONLY INDEX CODE
000105  0 064211 3 2 0 10 11          MOVB    STINDX                 ;CLEAR THE REQUEST BIT
000106  0 016000 0 7 0 000             RETURN

                                ;A BAD INDEX CODE HAS BEEN RECEIVED
                                ;MAKE THE CODE A ZERO THEN RE-ENTER

000107  0 002200 0 1 0 200     BADIDX: LDBR    200                    ;GET REQUEST AND CODE 0
000110  0 064211 3 2 0 10 11          MOVB    STINDX                 ;PUT IN INDEX, ZERO IS ALWAYS VALID
000111  0 100056 4 0 0056             JMP     EXTREQ                 ;NOW READ INDEX 0
```

```
                                        SUBTTL  CHECK FOR ASYNC STATUS TO BE PRESENTED

000112  0 132001 5 5 0 00 01    CKASYC: DATI    ASYCST,AC0              ;READ CURRENT ASYNC STATUS
000113  0 060007 3 0 0 00 07            DEC     AC0                     ;SUBTRACT 1 TO GEN -1 IF WAS ZERO
000114  0 114116 4 6 0116              JMPZ    CKREQI                  ;IF ZERO, CHECK IF REQ IN IS SET
000115  0 016000 0 7 0 000             RETURN                          ;CAN'T PRESENT STATUS UNLESS ZERO
000116  0 022001 1 1 0 00 01    CKREQI: DATI    MPSCR1,BR               ;READ STATUS REGISTER
000117  0 110146 4 4 0146              JMPB7   NRTN                    ;CAN'T PRESENT STATUS UNLESS ATTENTION IS OFF
000120  0 014000 0 6 0 000             SHR                             ;SHIFT COMPOSITE ERROR BIT
000121  0 014000 0 6 0 000             SHR                             ;TO BIT 0
000122  0 104146 4 2 0146              JMPB0   NRTN                    ;CAN'T PRESENT STATUS UNLESS ERROR IS CLEAR

                                        ;CHECK IF REQ IN IS SET ON CHANNEL BUS

000123  0 002033 0 1 0 033             LDBR    CHN                     ;SELECT CHANNEL BUS INTERFACE
000124  0 066371 3 3 0 17 11          MOVB    IOSEL
000125  0 022005 1 1 0 00 05          DATI    TAGIN1,BR               ;READ TAG IN LINES
000126  0 014000 0 6 0 000             SHR                             ;SHIFT REQ IN TO BR4
000127  0 106131 4 3 0131              JMPB4   .+2                     ;SKIP IF SET
000130  0 100134 4 0 0134              JMP     NOREQI                  ;  NO
000131  0 000400 0 0 1 000             LDMARX  0                       ;SELECT MEMORY PAGE 0
000132  0 117420 4 7 1420              JMPSUB  DEVREQ                  ;SERVICE THE DEVICE REQUEST
000133  0 114137 4 6 0137              JMPZ    REQATA                  ;JUMP IF STATUS WAS STORED
000134  0 002011 0 1 0 011    NOREQI: LDBR    MB                      ;SELECT MASSBUS INTERFACE
000135  0 066371 3 3 0 17 11          MOVB    IOSEL
000136  0 016000 0 7 0 000             RETURN

                                        ;ASYNC STATUS WAS STORED, SET ATTENTION TO INTERRUPT THE HOST

000137  0 002011 0 1 0 011    REQATA: LDBR    MB                      ;SELECT MASSBUS INTERFACE
000140  0 066371 3 3 0 17 11          MOVB    IOSEL
000141  0 022000 1 1 0 00 00          DATI    MPSCR0,BR               ;[9] READ FUNCTION CODE REGISTER
000142  0 014000 0 6 0 000             SHR                             ;[9] SHIFT GO BIT TO BR0
000143  0 104146 4 2 0146              JMPB0   NRTN                    ;[9] DON'T SET ATA IF GO IS SET
000144  0 002200 0 1 0 200             LDBR    ATA                     ;SET ATTENTION
000145  0 064031 3 2 0 01 11          MOVB    MPSCR1
000146  0 016000 0 7 0 000     NRTN:   RETURN
```

```
                                         SUBTTL  PERFORM FUNCTION REQUESTED BY HOST

                                         ;A COMMAND HAS BEEN REQUESTED BY HOST
                                         ;DETERMINE TYPE OF COMMAND REQUESTED

000147  0 002000 0 1 0 000       DOCMD:  LDBR    0                       ;GET A ZERO
000150  0 064031 3 2 0 01 11            MOVB    MPSCR1                  ;CLEAR ATA, IF SET
000151  0 064051 3 2 0 02 11            MOVB    MPECR                   ;CLEAR ERROR CODE FIELD
000152  0 000401 0 0 1 001              LDMARX  FUNC_-8                 ;SELECT MEMORY PAGE CONTAINING SETUP TABLES
000153  0 032020 1 5 0 01 00            DATI    MPSCR0,AC1              ;READ FUNCTION CODE
000154  0 002174 0 1 0 174              LDBR    FN                      ;GET MASK OF FUNCTION CODE BITS
000155  0 072033 3 5 0 01 13            LANDBR  AC1                     ;CLEAR WCLK, GO AND RUN
000156  0 002000 0 1 0 000              LDBR    FUNC&377                ;GET ADDRESS OF FUNCTION TABLE
000157  0 061020 3 0 2 01 00            ADB     AC1,MAR                 ;ADD INDEX OF FUNCTION CODE
000160  0 053471 2 5 3 03 11            MOVMEM  AC3,I                   ;GET THE CODE
000161  0 053571 2 5 3 07 11            MOVMEM  AC7,I                   ;GET CONTROL BITS
000162  0 103767 4 1 1767              JMPI    INTERUPT                ;CHECK ON INTERRUPTS
000163  0 140211 6 0 0 10 11            JMP     @MEM,FNCDSP             ;DISPATCH TO PROPER ROUTINE


                                         ;DISPATCH TABLE FOR MAJOR FUNCTION HANDLER ROUTINES

000164                                   TABLE   FNCDSP,9                ;DEFINE A TABLE
000164  0 100226 4 0 0226       IMMCMD: JMP     DOIMM                   ;IMMEDIATE COMMAND
000165  0 100006 4 0 0006       CLRCMD: JMP     BEGIN5                  ;DRIVE CLEAR
000166  0 100351 4 0 0351       WRTCMD: JMP     DOWRT                   ;WRITE COMMAND
000167  0 100552 4 0 0552       RDCMD:  JMP     DOREAD                  ;READ COMMAND
000170  0 101173 4 0 1173       SNSCMD: JMP     DOSNS                   ;SENSE COMMAND
000171  0 002000 0 1 0 000      DMPCMD: JUMP    DODUMP                  ;DUMP COMMAND
000172  0 160231 7 0 0 11 11
000173  0 100175 4 0 0175       BADCMD: JMP     ILFUNC                  ;ILLEGAL NON-DATA TRANSFER COMMAND
000174  0 100204 4 0 0204       BADXFR: JMP     ILXFER                  ;ILLEGAL DATA TRANSFER COMMAND
```

```
                                        SUBTTL  ILLEGAL FUNCTION HANDLER

000175  0 117712 4 7 1712       ILFUNC: JMPSUB  CLRSNS                  ;CLEAR SENSE BYTE AREA
000176  0 117734 4 7 1734               JMPSUB  LOGREG                  ;LOG THE REGISTERS
000177  0 002000 0 1 0 000              LDBR    0                       ;CLEAR THE FLAGS
000200  0 064371 3 2 0 17 11            MOVB    FLAGS
000201  0 002242 0 1 0 242              LDBR    ATA+ILF+CLRGO           ;CLEAR GO AND
000202  0 064031 3 2 0 01 11            MOVB    MPSCR1                  ;  SET ILF AND ATA
000203  0 100026 4 0 0026               JMP     IDLE                    ;GO IDLE

000204  0 117712 4 7 1712       ILXFER: JMPSUB  CLRSNS                  ;CLEAR THE SENSE BYTE AREA
000205  0 117734 4 7 1734               JMPSUB  LOGREG                  ;LOG THE REGISTERS^G
000206  0 002000 0 1 0 000              LDBR    0                       ;CLEAR THE FLAGS
000207  0 064371 3 2 0 17 11            MOVB    FLAGS
000210  0 002020 0 1 0 020              LDBR    OCC                     ;SET OCCUPIED
000211  0 064031 3 2 0 01 11            MOVB    MPSCR1
000212  0 002060 0 1 0 060       ILDFNC: LDBR    ILF+OCC                 ;SET ILF TO CAUSE EXCEPTION
000213  0 064031 3 2 0 01 11            MOVB    MPSCR1
000214  0 117734 4 7 1734               JMPSUB  LOGREG                  ;LOG ALL THE REGISTERS
000215  0 002040 0 1 0 040              LDBR    ILF                     ;CLEAR OCCUPIED
000216  0 064031 3 2 0 01 11            MOVB    MPSCR1
000217  0 002041 0 1 0 041              LDBR    ILF+EBL                 ;SEND AN EBL
000220  0 064031 3 2 0 01 11            MOVB    MPSCR1
000221  0 022000 1 1 0 00 00            DATI    MPSCR0,BR               ;WAIT FOR RUN TO DROP
000222  0 104221 4 2 0221               JMPB0   .-1
000223  0 002242 0 1 0 242              LDBR    ILF+ATA+CLRGO           ;SET ATA AND CLEAR GO
000224  0 064031 3 2 0 01 11            MOVB    MPSCR1
000225  0 100026 4 0 0026               JMP     IDLE                    ;GO IDLE

684
```

```
                                SUBTTL  IMMEDIATE COMMAND EXECUTION

                                ;CONTROL BITS IN AC7 FOR IMMEDIATE COMMANDS FROM SEARCH TABLES

                                ;       BIT 7   SET TO BYPASS ISSUING MODE SET COMMANDS
                                ;       BIT 6   SET TO FORCE END OF COMMAND ON INITIAL STATUS
                                ;       BIT 4   SET TO BYPASS BLOCK COUNT CHECKS
                                ;       BIT 3   SET FOR ALL COMMANDS EXCEPT DATA SECURITY ERASE
                                ;       BIT 2   SET FOR REW AND UNL COMMANDS TO PREVENT FORCED
                                ;               SENSE BYTE READ AT NORMAL COMMAND END
                                ;       BIT 0   ALWAYS SET

                                ;AN IMMEDIATE COMMAND HAS BEEN RECEIVED
                                ;CHECK IF DRIVE MODE IS LEGAL

000226  0 117712 4 7 1712      DOIMM:  JMPSUB  CLRSNS                  ;CLEAR SENSE BYTE AREA
000227  0 032132 1 5 0 05 12          DATI    DR,AC5                  ;READ DRIVE NUMBER
000230  0 062170 3 1 0 07 10          MOV     AC7,BR                  ;GET CONTROL BITS INTO BR
000231  0 110252 4 4 0252            JMPB7   NOMOD                   ;JUMP AROUND IF TO DO NO MODE SET
000232  0 000401 0 0 1 001           LDMARX  DRVMOD_-8               ;SELECT MEMORY BANK FOR MODE COMMAND
000233  0 032033 1 5 0 01 13          DATI    MODE,AC1                ;READ DRIVE MODE REGISTER
000234  0 002017 0 1 0 017           LDBR    17                      ;GET MASK OF DRIVE MODE BITS
000235  0 072033 3 5 0 01 13          LANDBR  AC1                     ;CLEAR OTHER BITS
000236  0 002200 0 1 0 200           LDBR    DRVMOD&377              ;GET ADDRESS OF MODE TABLE
000237  0 061020 3 0 2 01 00          ADB     AC1,MAR                 ;GET ADDRESS INDEXED BY CODE
000240  0 052151 2 5 0 06 11          MOVMEM  AC6                     ;COPY COMMAND BYTE TO AC6
000241  0 114175 4 6 0175            JMPZ    ILFUNC                  ;IF -1, CODE WAS ILLEGAL
000242  0 000400 0 0 1 000           LDMARX  0                       ;SELECT MEMORY BANK ZERO
000243  0 062150 3 1 0 06 10          MOV     AC6,BR                  ;COPY COMMAND BYTE TO BR
000244  0 104246 4 2 0246            JMPB0   .+2                     ;IF CODE NOT ZERO, SEND TO DRIVE
000245  0 100252 4 0 0252            JMP     NOMOD                   ;CODE IS ZERO, GO AROUND

                                ;ISSUE A MODE SET COMMAND TO DRIVE

000246  0 117270 4 7 1270            JMPSUB  SELECT                  ;SELECT DRIVE AND SEND COMMAND
000247  0 103622 4 1 1622            JMPI    ISTPE                   ;CHECK IF A STATUS BYTE PARITY ERROR
000250  0 115614 4 6 1614            JMPZ    ISTERR                  ;JUMP IF INITIAL STATUS ERROR
000251  0 117253 4 7 1253            JMPSUB  CHAIN                   ;ACCEPT STATUS AND INDICATE CHAINING
```

```
                                ;SEND AN ERASE GAP COMMAND IF DATA SECURITY ERASE

000252  0 062165 3 1 0 07 05    NOMOD:  SHL     AC7,BR          ;CHECK IF DATA SECURITY ERASE
000253  0 106266 4 3 0266               JMPB4   SENCMD          ;NO, GO SEND COMMAND
000254  0 002027 0 1 0 027              LDBR    27              ;YES, SEND AN ERASE GAP COMMAND
000255  0 072151 3 5 0 06 11           MOVB    AC6             ;PUT COMMAND IN AC6
000256  0 117270 4 7 1270              JMPSUB  SELECT          ;SELECT DEVICE AND SEND THE COMMAND
000257  0 103622 4 1 1622              JMPI    ISTPE           ;CHECK IF A STATUS BYTE PARITY ERROR
000260  0 115614 4 6 1614              JMPZ    ISTERR          ;JUMP IF AN INITIAL STATUS ERROR
000261  0 117224 4 7 1224              JMPSUB  ACCEPT          ;ACCEPT THE STATUS
000262  0 117211 4 7 1211              JMPSUB  WAITST          ;WAIT FOR ENDING STATUS
000263  0 103643 4 1 1643              JMPI    ESTPE           ;CHECK IF A STATUS BYTE PARITY ERROR
000264  0 115625 4 6 1625              JMPZ    ESTERR          ;JUMP IF AN ENDING STATUS ERROR
000265  0 117253 4 7 1253              JMPSUB  CHAIN           ;ACCEPT THE STATUS AND INDICATE CHAINING

                                ;SEND COMMAND AS REQUESTED

000266  0 002011 0 1 0 011     SENCMD: LDBR    MB              ;SELECT MASSBUS INTERFACE
000267  0 066371 3 3 0 17 11           MOVB    IOSEL
000270  0 032117 1 5 0 04 17           DATI    FLAGS,AC4       ;GET FLAGS INTO AC4
000271  0 062070 3 1 0 03 10           MOV     AC3,BR          ;GET COMMAND CODE FROM AC3
000272  0 072151 3 5 0 06 11           MOVB    AC6             ;PLACE IN AC6
000273  0 117270 4 7 1270      RPTCMD: JMPSUB  SELECT          ;SELECT DRIVE AND SEND COMMAND
000274  0 103622 4 1 1622              JMPI    ISTPE           ;CHECK IF A STATUS BYTE PARITY ERROR
000275  0 115614 4 6 1614              JMPZ    ISTERR          ;JUMP IF INITIAL STATUS ERROR
000276  0 062165 3 1 0 07 05           SHL     AC7,BR          ;CHECK IF COMMAND IS DONE
000277  0 110323 4 4 0323              JMPB7   QUICK           ;  ON INITIAL STATUS (BIT 6 OF AC7)
000300  0 117224 4 7 1224              JMPSUB  ACCEPT          ;ACCEPT THE STATUS
000301  0 117211 4 7 1211              JMPSUB  WAITST          ;WAIT FOR ENDING STATUS
000302  0 103643 4 1 1643              JMPI    ESTPE           ;CHECK IF A STATUS BYTE PARITY ERROR
000303  0 115625 4 6 1625              JMPZ    ESTERR          ;JUMP IF AN ENDING STATUS ERROR

                        ;IF A REPEATABLE COMMAND, INCREMENT THE BLOCK COUNTER
                        ;THEN EXIT ONLY IF BLOCK COUNT WENT TO ZERO

000304  0 062170 3 1 0 07 10           MOV     AC7,BR          ;GET CONTROL BITS
000305  0 106326 4 3 0326              JMPB4   CMDEND          ;END OF COMMAND IF NOT REPEATABLE
000306  0 002011 0 1 0 011             LDBR    MB              ;SELECT MASSBUS INTERFACE
000307  0 066371 3 3 0 17 11           MOVB    IOSEL
000310  0 132002 5 5 0 00 02           DATI    CNTL,AC0        ;GET LOW ORDER OF BLOCK COUNT
000311  0 062003 3 1 0 00 03           INC     AC0,BR          ;INCREMENT THE COUNT
000312  0 066051 3 3 0 02 11           MOVB    CNTL            ;RESTORE IT
000313  0 132003 5 5 0 00 03           DATI    CNTH,AC0        ;GET HIGH ORDER BITS
000314  0 062004 3 1 0 00 04           ADC     AC0,BR          ;ADD CARRY, IF ANY
000315  0 066071 3 3 0 03 11           MOVB    CNTH            ;RESTORE COUNT
000316  0 002033 0 1 0 033             LDBR    CHN             ;SELECT CHANNEL INTERFACE
000317  0 066371 3 3 0 17 11           MOVB    IOSEL
000320  0 112326 4 5 0326              JMPC    CMDEND          ;END OF COMMAND IF COUNT WENT TO ZERO
000321  0 117253 4 7 1253              JMPSUB  CHAIN           ;ACCEPT THE STATUS AND INDICATE CHAINING
000322  0 100273 4 0 0273              JMP     RPTCMD          ;REPEAT THE COMMAND
```

```
                                        ;CHECK IF COMMAND WAS A REW OR UNL
                                        ;IF SO, PREVENT READING SENSE BYTES

000323  0 002004 0 1 0 004      QUICK:  LDBR    SUPRES                  ;GET SUPPRESS SENSE BYTE BIT
000324  0 062173 3 1 0 07 13           LANDB   AC7,BR                  ;CHECK IF SET IN CONTROL BITS
000325  0 072114 3 5 0 04 14           LORBR   AC4                     ;IF SO, SET SUPPRESS BIT IN FLAGS

                                        ;END OF COMMAND EXECUTION
                                        ;REPORT NORMAL ENDING STATUS AND INTERRUPT HOST

000326  0 002011 0 1 0 011      CMDEND: LDBR    MB                      ;SELECT MASSBUS INTERFACE
000327  0 066371 3 3 0 17 11           MOVB    IOSEL
000330  0 062030 3 1 0 01 10           MOV     AC1,BR                  ;GET ENDING STATUS
000331  0 064231 3 2 0 11 11           MOVB    STBYTE                  ;PUT IN STATUS BYTE REGISTER
000332  0 062110 3 1 0 04 10           MOV     AC4,BR                  ;GET FLAGS
000333  0 014000 0 6 0 000             SHR                             ;SHIFT FSENSE TO BR0
000334  0 104346 4 2 0346              JMPB0   FORCST                  ;SKIP IF SET
000335  0 117234 4 7 1234              JMPSUB  DSELCT                  ;DESELECT FROM DEVICE
000336  0 002011 0 1 0 011      NORMDN: LDBR    MB                      ;SELECT MASSBUS
000337  0 066371 3 3 0 17 11           MOVB    IOSEL
000340  0 002000 0 1 0 000             LDBR    0                       ;GET A ZERO
000341  0 064371 3 2 0 17 11           MOVB    FLAGS                   ;CLEAR THE FLAGS
000342  0 103767 4 1 1767              JMPI    INTERUPT                ;CHECK ON INTERRUPTS
000343  0 002202 0 1 0 202             LDBR    ATA+CLRGO               ;GET ATTENTION BIT
000344  0 064031 3 2 0 01 11           MOVB    MPSCR1                  ;INTERRUPT THE HOST
000345  0 100026 4 0 0026              JMP     IDLE                    ;GO IDLE

000346  0 117734 4 7 1734      FORCST: JMPSUB  LOGREG                  ;LOG THE REGISTERS
000347  0 117602 4 7 1602              JMPSUB  CSENSE                  ;READ THE SENSE BYTES WITH CHAINED COMMAND
000350  0 100336 4 0 0336              JMP     NORMDN                  ;NOW TERMINATE COMMAND
```

```
                                SUBTTL  WRITE COMMAND EXECUTION

                                ;CONTROL BITS IN AC7 FOR WRITE COMMANDS FROM SEARCH TABLES

                                ;       BIT 6   SET FOR ALL COMMANDS EXCEPT DIAGNOSTIC MODE
                                ;       BIT 0   ALWAYS SET


                                ;A WRITE COMMAND HAS BEEN RECEIVED
                                ;CHECK IF DRIVE MODE IS LEGAL

000351  0 002030 0 1 0 030     DOWRT:  LDBR    OCC+DTD         ;SET OCCUPIED AND DATA TRANSFER
000352  0 064031 3 2 0 01 11             MOVB    MPSCR1          ; DIRECTION TO DEVICE
000353  0 000400 0 0 1 000              LDMARX  0               ;SELECT MEMORY PAGE 0
000354  0 001006 0 0 2 006              LDMAR   CMND            ;SELECT MEMORY LOCATION TO SAVE
000355  0 070070 3 4 0 03 10            MOV     AC3,MEM         ; THE REQUESTED COMMAND
000356  0 032033 1 5 0 01 13            DATI    MODE,AC1        ;READ DRIVE MODE REGISTER
000357  0 002017 0 1 0 017              LDBR    17              ;GET MASK OF DRIVE MODE BITS
000360  0 062033 3 1 0 01 13            LANDB   AC1,BR          ;CLEAR OTHER BITS
000361  0 072011 3 5 0 00 11            MOVB    AC0             ;SAVE IN AC0
000362  0 000401 0 0 1 001              LDMARX  DRVMOD_-8       ;SELECT MEMORY PAGE CONTAINING TABLES
000363  0 002200 0 1 0 200              LDBR    DRVMOD&377      ;GET ADDRESS OF DRIVE MODE TABLE
000364  0 061000 3 0 2 00 00            ADB     AC0,MAR         ;INDEX BY CODE RECEIVED
000365  0 052151 2 5 0 06 11            MOVMEM  AC6             ;COPY CODE COMMAND BYTE INTO AC6
000366  0 114212 4 6 0212               JMPZ    ILDFNC          ;IF -1, CODE WAS ILLEGAL

                                ;CHECK IF DATA MODE IS LEGAL

000367  0 062030 3 1 0 01 10            MOV     AC1,BR          ;GET DATA FORMAT CODE
000370  0 110212 4 4 0212               JMPB7   ILDFNC          ;IF 10-17, ILLEGAL
000371  0 014000 0 6 0 000              SHR                     ;RIGHT JUSITFY THE FORMAT CODE
000372  0 014000 0 6 0 000              SHR
000373  0 014000 0 6 0 000              SHR
000374  0 014000 0 6 0 000              SHR
000375  0 072031 3 5 0 01 11            MOVB    AC1             ;SAVE IT
000376  0 002220 0 1 0 220              LDBR    DATMOD&377      ;GET DATA MODE TABLE ADDRESS
000377  0 061020 3 0 2 01 00            ADB     AC1,MAR         ;INDEX BY CODE RECEIVED
000400  0 052011 2 5 0 00 11            MOVMEM  AC0             ;READ THE CONTENTS OF TABLE ENTRY
000401  0 114212 4 6 0212               JMPZ    ILDFNC          ;IF -1, ILLEGAL CODE
```

```
                                        ;COMPUTE HALF WORDS TO TRANSFER OVER MASSBUS

                                        ;[10]   JMPSUB  DIVIDE          ;DIVIDE BYTE COUNT BY BYTES/WORD
000402  0 002327 0 1 0 327              GOSUB   DIVIDE          ;[10] DIVIDE BYTE COUNT BY BYTES/WORD
000403  0 177231 7 7 2 11 11
000404  0 060047 3 0 0 02 07            DEC     AC2             ;GET REMAINDER -1
000405  0 114410 4 6 0410               JMPZ    .+3             ;JUMP IF REMAINDER WAS ZERO
000406  0 072103 3 5 0 04 03            INCR    AC4             ;REMAINDER WAS NON ZERO
000407  0 072064 3 5 0 03 04            ADCR    AC3             ;INCREASE WORD COUNT BY ONE
000410  0 062005 3 1 0 00 05            SHL     AC0,BR          ;GET BYTES/WORD TIMES 2
000411  0 106413 4 3 0413               JMPB4   .+2             ;SKIP IF HI-DEN MODE
000412  0 100425 4 0 0425               JMP     HIDMOD          ;NO
000413  0 072105 3 5 0 04 05            SHLR    AC4             ;DOUBLE THE WORD COUNT
000414  0 072066 3 5 0 03 06            ROTLR   AC3             ; AS BYTES/WORD WAS FOR 2 WORDS
000415  0 060047 3 0 0 02 07            DEC     AC2             ;GET REMAINDER -1
000416  0 114425 4 6 0425               JMPZ    HIDMOD          ;JUMP IF REMAINDER WAS ZERO
000417  0 002005 0 1 0 005               LDBR    5               ;SUBTRACT FIVE FROM REMAINING BYTES
000420  0 060056 3 0 0 02 16            TSB     AC2             ; TO SEE IF REMAINING BYTES
000421  0 112425 4 5 0425               JMPC    HIDMOD          ; WILL REQUIRE SECOND WORD
000422  0 072067 3 5 0 03 07            DECR    AC3             ;NO, DECREASE WORD COUNT
000423  0 072107 3 5 0 04 07            DECR    AC4             ; BY ONE
000424  0 072064 3 5 0 03 04            ADCR    AC3
000425  0 002000 0 1 0 000      HIDMOD: LDBR    0               ;PUT A ZERO IN AC2
000426  0 072051 3 5 0 02 11            MOVB    AC2
000427  0 062110 3 1 0 04 10            MOV     AC4,BR          ;NEGATE THE WORD COUNT
000430  0 062056 3 1 0 02 16            TSB     AC2,BR          ;IN AC3-AC4
000431  0 072111 3 5 0 04 11            MOVB    AC4
000432  0 062070 3 1 0 03 10            MOV     AC3,BR
000433  0 062042 3 1 0 02 02            OSBC    AC2,BR
000434  0 072071 3 5 0 03 11            MOVB    AC3
```

```
                                ;GET ADDRESS OF FORMATTER START ADDRESS FOR DATA MODE

000435  0 002227 0 1 0 227            LDBR    377&<FMTWRT-1>          ;GET ADDRESS OF TABLE
000436  0 061020 3 0 2 01 00          ADB     AC1,MAR                 ;INDEX BY DATA MODE
000437  0 042011 2 1 0 00 11          MOVMEM  BR                      ;GET START ADDRESS
000440  0 000400 0 0 1 000             LDMARX  0                       ;SELECT MEMORY PAGE 0
000441  0 001010 0 0 2 010             LDMAR   MCOUNT                  ;SELECT MASSBUS COUNTER STORAGE AREA
000442  0 071505 3 4 3 04 05          SHL     AC4,MEM,I               ;MULTIPLY WORD COUNT BY TWO
000443  0 071466 3 4 3 03 06          ROTL    AC3,MEM,I               ;TO OBTAIN HALF WORD COUNT
000444  0 070011 3 4 0 00 11          MOVB    MEM                     ;SAVE FORMATTER START ADDRESS
000445  0 117712 4 7 1712             JMPSUB  CLRSNS                  ;CLEAR THE SENSE BYTE AREA

                                ;WAIT FOR RH20 TO ASSERT RUN

000446  0 022000 1 1 0 00 00          DATI    MPSCR0,BR               ;READ STATUS REGISTER
000447  0 104452 4 2 0452             JMPB0   .+3                     ;SKIP IF RUN IS ASSERTED
000450  0 002035 0 1 0 035             GOSUB   WAITRN                  ;NO, WAIT FOR IT
000451  0 176231 7 7 0 11 11

                                ;ISSUE MODE SET TO DRIVE IF REQUESTED

000452  0 032132 1 5 0 05 12          DATI    DR,AC5                  ;GET DRIVE NUMBER
000453  0 062150 3 1 0 06 10          MOV     AC6,BR                  ;GET MODE SET COMMAND BYTE
000454  0 104456 4 2 0456             JMPB0   .+2                     ;SKIP IF NOT ZERO
000455  0 100464 4 0 0464             JMP     WNOMOD                  ;NO MODE SET REQUESTED
000456  0 117270 4 7 1270             JMPSUB  SELECT                  ;SELECT DRIVE AND SEND COMMAND
000457  0 103622 4 1 1622             JMPI    ISTPE                   ;CHECK IF A STATUS BYTE PARITY ERROR
000460  0 115614 4 6 1614             JMPZ    ISTERR                  ;JUMP IF INITIAL SELECTION STATUS
000461  0 117253 4 7 1253             JMPSUB  CHAIN                   ;ACCEPT STATUS AND INDICATE CHAINING

                                ;IF COMMAND IS DIAGNOSTIC WRITE:
                                ;SEND DIAGNOSTIC MODE SET COMMAND
                                ;THEN CHANGE COMMAND TO A WRITE

000462  0 002011 0 1 0 011             LDBR    MB                      ;SELECT MASSBUS INTERFACE
000463  0 066371 3 3 0 17 11          MOVB    IOSEL
000464  0 001006 0 0 2 006     WNOMOD: LDMAR   CMND                    ;SELECT MEMORY ADDRESS WITH SAVED COMMAND BYTE
000465  0 052151 2 5 0 06 11          MOVMEM  AC6                     ;MOVE COMMAND INTO AC6
000466  0 032117 1 5 0 04 17          DATI    FLAGS,AC4               ;GET FLAGS INTO AC4
000467  0 062165 3 1 0 07 05          SHL     AC7,BR                  ;CHECK IF DIAGNOSTIC MODE
000470  0 110477 4 4 0477             JMPB7   SNDWRT                  ;IF NOT, IT IS TIME TO WRITE
000471  0 117270 4 7 1270             JMPSUB  SELECT                  ;SELECT THE DRIVE AND SEND COMMAND
000472  0 103622 4 1 1622             JMPI    ISTPE                   ;CHECK IF A STATUS BYTE PARITY ERROR
000473  0 115614 4 6 1614             JMPZ    ISTERR                  ;JUMP IF AN INITIAL STATUS ERROR
000474  0 117253 4 7 1253             JMPSUB  CHAIN                   ;ACCEPT STATUS AND INDICATE CHAINING
000475  0 002001 0 1 0 001             LDBR    1                       ;GET A WRITE COMMAND
000476  0 072151 3 5 0 06 11          MOVB    AC6                     ;CHANGE COMMAND BYTE
```

```
                                        ;SEND THE WRITE COMMAND

000477  0 117270 4 7 1270      SNDWRT: JMPSUB  SELECT          ;SELECT DRIVE AND SEND COMMAND
000500  0 103622 4 1 1622              JMPI    ISTPE           ;CHECK IF A STATUS BYTE PARITY ERROR
000501  0 115614 4 6 1614              JMPZ    ISTERR          ;JUMP IF INITIAL SELECTION ERROR
000502  0 117224 4 7 1224              JMPSUB  ACCEPT          ;ACCEPT THE STATUS
                                                              ; START TAPE MOTION
000503  0 001007 0 0 2 007             LDMAR   WRTSET          ;SELECT MEMORY LIST FOR SETUP
000504  0 047771 2 3 3 17 11           MOVMEM  IOSEL,I         ;SELECT DATA PATH FOR OUTPUT, MB FOR INPUT
000505  0 124142 5 2 0 06 02           DATI    CNTL,BCLO       ;LOAD BYTE COUNTER
000506  0 124163 5 2 0 07 03           DATI    CNTH,BCHO
000507  0 045511 2 2 3 04 11           MOVMEM  MCLO,I          ;LOAD MASSBUS COUNTER
000510  0 045531 2 2 3 05 11           MOVMEM  MCHO,I
000511  0 045611 2 2 3 10 11           MOVMEM  DFRMAD,I        ;LOAD FORMATTER START ADDRESS
000512  0 045451 2 2 3 02 11           MOVMEM  REG2,I          ;SET END ENABLES, CLEAR ROM ADR 8
000513  0 044011 2 2 0 00 11           MOVMEM  REG0            ;CLEAR FLAGS
000514  0 044271 2 2 0 13 11           MOVMEM  HSDPIN          ;PULSE HIGH SPEED INIT
000515  0 045431 2 2 3 01 11           MOVMEM  REG1,I          ;SET DX HIGH SPEED
000516  0 047771 2 3 3 17 11           MOVMEM  IOSEL,I         ;SELECT MASSBUS INTERFACE
000517  0 045431 2 2 3 01 11           MOVMEM  MPSCR1,I        ;START THE TRANSFER

                                        ;WAIT FOR STATUS IN FROM DEVICE

000520  0 047771 2 3 3 17 11           MOVMEM  IOSEL,I         ;SELECT THE CHANNEL INTERFACE
000521  0 103767 4 1 1767              JMPI    INTERUPT        ;CHECK ON INTERRUPTS
000522  0 022004 1 1 0 00 04           DATI    TAGIN0,BR       ;READ THE TAG IN LINES
000523  0 104525 4 2 0525              JMPB0   .+2             ;SKIP IF STATUS IN IS SET
000524  0 100521 4 0 0521              JMP     .-3             ;NO, KEEP WAITING
000525  0 047771 2 3 3 17 11           MOVMEM  IOSEL,I         ;SELECT DATA PATH
000526  0 045431 2 2 3 01 11           MOVMEM  REG1,I          ;CLEAR HIGH SPEED
000527  0 103076 4 1 1076              JMPI    WRTCHK          ;CHECK INTERRUPTS FOR PARITY ERRORS
000530  0 047771 2 3 3 17 11           MOVMEM  IOSEL,I         ;SELECT CHANNEL INTERFACE
000531  0 117215 4 7 1215              JMPSUB  CHKSTS          ;GET AND CHECK ENDING STATUS
000532  0 103643 4 1 1643              JMPI    ESTPE           ;CHECK IF A STATUS BYTE PARITY ERROR
000533  0 115625 4 6 1625              JMPZ    ESTERR          ;JUMP IF AN ENDING STATUS ERROR
```

```
961                                     ;SEND EBL AND CHECK IF DONE
962
963  000534  0 047771 2 3 3 17 11       MOVMEM  IOSEL,I         ;SELECT MASSBUS INTERFACE
964  000535  0 032007 1 5 0 00 07       DATI    MPDB2,AC0       ;READ DATA BUFFER PARITY ERROR BIT
965  000536  0 062005 3 1 0 00 05       SHL     AC0,BR          ;MOVE TO BR BIT 4
966  000537  0 107117 4 3 1117          JMPB4   WRTPE           ;JUMP IF DATA BUFFER PARITY ERROR
967  000540  0 045431 2 2 3 01 11       MOVMEM  MPSCR1,I        ;SEND EBL TO RH20
968  000541  0 000000 0 0 0 000         NOP                     ;STALL
969  000542  0 000000 0 0 0 000         NOP                     ; ALLOW RUN TO DROP
970  000543  0 100544 4 0 0544          JMP     .+1
971  000544  0 022000 1 1 0 00 00       DATI    MPSCR0,BR       ;READ RUN BIT
972  000545  0 104547 4 2 0547          JMPB0   .+2             ;SKIP IF RUN IS STILL SET
973  000546  0 101020 4 0 1020          JMP     XFREND          ;NO, END OF COMMAND
974  000547  0 046371 2 3 0 17 11       MOVMEM  IOSEL           ;SELECT CHANNEL INTERFACE
975  000550  0 117253 4 7 1253          JMPSUB  CHAIN           ;ACCEPT THE STATUS AND INDICATE CHAINING
976  000551  0 100477 4 0 0477          JMP     SNDWRT          ;PERFORM ANOTHER WRITE OPERATION
977
```

```
                                SUBTTL  READ COMMAND EXECUTION

                                ;CONTROL BITS IN AC7 FOR READ COMMAND FROM SEARCH TABLES
                                ;       BIT 6   SET FOR READ BACKWARD
                                ;       BIT 0   ALWAYS ZERO


                                ;A READ COMMAND HAS BEEN RECEIVED
                                ;CHECK IF DRIVE MODE IS LEGAL

000552  0 002020 0 1 0 020      DOREAD: LDBR    OCC                     ;SET OCCUPIED AND DATA TRANSFER
000553  0 064031 3 2 0 01 11            MOVB    MPSCR1                  ; DIRECTION FROM DEVICE
000554  0 000400 0 0 1 000              LDMARX  0                       ;SELECT MEMORY PAGE 0
000555  0 001006 0 0 2 006              LDMAR   CMND                    ;SELECT MEMORY LOCATION TO SAVE COMMAND
000556  0 070070 3 4 0 03 10            MOV     AC3,MEM                 ;SAVE COMMAND BYTE
000557  0 032033 1 5 0 01 13            DATI    MODE,AC1                ;READ DRIVE MODE REGISTER
000560  0 002017 0 1 0 017              LDBR    17                      ;GET MASK OF DRIVE MODE BITS
000561  0 062033 3 1 0 01 13            LANDB   AC1,BR                  ;CLEAR OTHER BITS
000562  0 072011 3 5 0 00 11            MOVB    AC0                     ;SAVE IN AC0
000563  0 000401 0 0 1 001              LDMARX  DRVMOD_-8               ;SELECT MEMORY PAGE CONTAINING TABLES
000564  0 002200 0 1 0 200              LDBR    DRVMOD&377              ;SELECT DRIVE MODE TABLE
000565  0 061000 3 0 2 00 00            ADB     AC0,MAR                 ;INDEX BY CODE RECEIVED
000566  0 052151 2 5 0 06 11            MOVMEM  AC6                     ;COPY COMMAND BYTE TO AC6
000567  0 114212 4 6 0212               JMPZ    ILDFNC                  ;IF -1, ILLEGAL CODE

                                ;CHECK IF DATA MODE IS LEGAL

000570  0 062030 3 1 0 01 10            MOV     AC1,BR                  ;GET DATA FORMAT CODE
000571  0 110212 4 4 0212               JMPB7   ILDFNC                  ;IF 10-17, CODE IS ILLEGAL
000572  0 014000 0 6 0 000              SHR                             ;RIGHT JUSITFY THE CODE
000573  0 014000 0 6 0 000              SHR
000574  0 014000 0 6 0 000              SHR
000575  0 014000 0 6 0 000              SHR
000576  0 072031 3 5 0 01 11            MOVB    AC1                     ;SAVE IT
000577  0 002220 0 1 0 220              LDBR    DATMOD&377              ;GET ADDRESS OF DATA MODE TABLE
000600  0 061020 3 0 2 01 00            ADB     AC1,MAR                 ;INDEX BY CODE RECEIVED
000601  0 040011 2 0 0 00 11            MOVMEM                          ;READ THE CONTENTS
000602  0 114212 4 6 0212               JMPZ    ILDFNC                  ;IF -1, CODE IS ILLEGAL
```

```
1017                                    ;GET ADDRESS OF FORMATTER START ADDRESSES FOR DATA MODE
1018
1019  000603  0 062165 3 1 0 07 05              SHL     AC7,BR                  ;GET CONTROL BITS
1020  000604  0 110610 4 4 0610                 JMPB7   RDBADR                  ;JUMP IF READ BACKWARD
1021  000605  0 002234 0 1 0 234                LDBR    377&<FMTRD-1>           ;GET ADDRESS OF TABLE
1022  000606  0 061020 3 0 2 01 00              ADB     AC1,MAR                 ;INDEX BY DATA MODE
1023  000607  0 100615 4 0 0615                 JMP     RDADR
1024
1025                                    ;[10] RDBADR:   JMPSUB  DIVIDE          ;DIVIDE BYTE COUNT BY BYTES/WORD
1026  000610  0 002327 0 1 0 327     RDBADR: GOSUB   DIVIDE                  ;[10] DIVIDE BYTE COUNT BY BYTES/WORD
1027  000611  0 177231 7 7 2 11 11
1028                                                                            ;RETURNS WITH REMAINDER IN AC2
1029  000612  0 002241 0 1 0 241                LDBR    377&<FMTRDB-1>          ;SELECT ADDRESS OF TABLE
1030  000613  0 061020 3 0 2 01 00              ADB     AC1,MAR                 ;INDEX BY DATA MODE TO FIND TABLE
1031                                                                            ; OF FORMATTER ADDRESSES
1032  000614  0 041040 2 0 2 02 00              ADM     AC2,MAR                 ;INDEX BY REMAINING BYTES IN FIRST WORD
1033  000615  0 042011 2 1 0 00 11     RDADR:  MOVMEM  BR                      ;GET START ADDRESS FOR FORMATTER
1034  000616  0 000400 0 0 1 000                LDMARX  0                       ;SELECT MEMORY PAGE 0
1035  000617  0 001030 0 0 2 030                LDMAR   FMTADR                  ;SELECT STORAGE LOCATION
1036  000620  0 070011 3 4 0 00 11              MOVB    MEM                     ;SAVE IT
1037  000621  0 117712 4 7 1712                 JMPSUB  CLRSNS                  ;CLEAR THE SENSE BYTE AREA
1038
1039                                    ;WAIT FOR RH20 TO ASSERT RUN
1040
1041  000622  0 022000 1 1 0 00 00              DATI    MPSCR0,BR               ;READ STATUS REGISTER
1042  000623  0 104626 4 2 0626                 JMPB0   .+3                     ;SKIP IF RUN IS SET
1043  000624  0 002035 0 1 0 035                GOSUB   WAITRN                  ;NO, WAIT FOR IT
1044  000625  0 176231 7 7 0 11 11
1045
1046                                    ;ISSUE MODE SET TO DRIVE IF REQUESTED
1047
1048  000626  0 032132 1 5 0 05 12              DATI    DR,AC5                  ;GET DRIVE NUMBER
1049  000627  0 062150 3 1 0 06 10              MOV     AC6,BR                  ;GET MODE SET COMMAND
1050  000630  0 104632 4 2 0632                 JMPB0   .+2                     ;SKIP IF NOT ZERO
1051  000631  0 100640 4 0 0640                 JMP     RNOMOD                  ;NO MODE SET REQUESTED
1052  000632  0 117270 4 7 1270                 JMPSUB  SELECT                  ;SELECT DRIVE AND SEND COMMAND
1053  000633  0 103622 4 1 1622                 JMPI    ISTPE                   ;CHECK IF A STATUS BYTE PARITY ERROR
1054  000634  0 115152 4 6 1152                 JMPZ    RDISE                   ;JUMP IF INITIAL SELECTION STATUS
1055  000635  0 117253 4 7 1253                 JMPSUB  CHAIN                   ;ACCEPT STATUS AND INDICATE CHAINING
1056
```

```
1057                                    ;SEND TRACK-IN-ERROR IF REQUESTED
1058
1059  000636  0 002011 0 1 0 011              LDBR    MB                      ;SELECT MASSBUS INTERFACE
1060  000637  0 066371 3 3 0 17 11            MOVB    IOSEL
1061  000640  0 032117 1 5 0 04 17  RNOMOD:  DATI    FLAGS,AC4               ;GET FLAGS
1062  000641  0 062110 3 1 0 04 10            MOV     AC4,BR                  ;COPY TO BR
1063  000642  0 104644 4 2 0644               JMPB0   .+2                     ;SKIP IF TRACK-IN-ERROR REQUESTED
1064  000643  0 100675 4 0 0675               JMP     NOTIE                   ;NO, GO AROUND
1065  000644  0 002033 0 1 0 033              LDBR    33                      ;GET TIE COMMAND CODE
1066  000645  0 072151 3 5 0 06 11            MOVB    AC6                     ; INTO AC6
1067  000646  0 032056 1 5 0 02 16            DATI    TIEBYT,AC2              ;GET TRACK-IN-ERROR BYTE
1068  000647  0 117270 4 7 1270               JMPSUB  SELECT                  ;SELECT DRIVE AND SEND COMMAND
1069  000650  0 103622 4 1 1622               JMPI    ISTPE                   ;CHECK IF A STATUS BYTE PARITY ERROR
1070  000651  0 115152 4 6 1152               JMPZ    RDISE                   ;JUMP IF INITIAL STATUS ERROR
1071  000652  0 002014 0 1 0 014              LDBR    14                      ;[11] SETUP MASK OF CE AND DE IN BR
1072  000653  0 062037 3 1 0 01 17            OSB     AC1,BR                  ;[11] COMPARE WITH STATUS RECEIVED
1073  000654  0 114675 4 6 0675               JMPZ    NOTIE                   ;[11] JUMP IF BOTH CE AND DE ASSERTED
1074                                                                          ;[11] AND DON'T TRY TO SEND TIE BYTE
1075  000655  0 117224 4 7 1224               JMPSUB  ACCEPT                  ;ACCEPT THE STATUS
1076  000656  0 062050 3 1 0 02 10            MOV     AC2,BR                  ;GET TRACK-IN-ERROR BYTE
1077  000657  0 064231 3 2 0 11 11            MOVB    BORLO                   ;SEND TO DEVICE
1078  000660  0 022005 1 1 0 00 05            DATI    TAGIN1,BR               ;READ TAG IN LINES
1079  000661  0 110663 4 4 0663               JMPB7   .+2                     ;SKIP IF SERVICE IN IS UP
1080  000662  0 100660 4 0 0660               JMP     .-2                     ;NO, WAIT FOR IT
1081  000663  0 002352 0 1 0 352              LDBR    SRVOUT+HLDOUT+SELOUT+CLKOUT+MTROUT      ;SEND SERVICE OUT TO DRIVE
1082  000664  0 064051 3 2 0 02 11            MOVB    TOR0
1083  000665  0 022005 1 1 0 00 05            DATI    TAGIN1,BR               ;READ TAG IN LINES
1084  000666  0 110665 4 4 0665               JMPB7   .-1                     ;WAIT FOR SERVICE IN TO DROP
1085  000667  0 002152 0 1 0 152              LDBR    HLDOUT+SELOUT+CLKOUT+MTROUT     ;CLEAR SERVICE OUT
1086  000670  0 064051 3 2 0 02 11            MOVB    TOR0
1087  000671  0 117211 4 7 1211               JMPSUB  WAITST                  ;WAIT FOR ENDING STATUS
1088  000672  0 103656 4 1 1656               JMPI    ESTPE0                  ;CHECK IF A STATUS BYTE PARITY ERROR
1089  000673  0 115155 4 6 1155               JMPZ    RDESE0                  ;JUMP IF AN ENDING STATUS ERROR
1090  000674  0 117253 4 7 1253               JMPSUB  CHAIN                   ;ACCEPT THE STATUS AND INDICATE CHAINING
1091
```

```
1092                                    ;SEND THE READ COMMAND
1093
1094  000675  0 001006 0 0 2 006    NOTIE:  LDMAR   CMND            ;SELECT ADDRESS WHERE COMMAND CODE IS SAVED
1095  000676  0 052151 2 5 0 06 11          MOVMEM  AC6             ;MOVE CODE TO AC6
1096  000677  0 117270 4 7 1270     SNDRD:  JMPSUB  SELECT          ;SELECT DRIVE AND SEND COMMAND
1097  000700  0 103622 4 1 1622             JMPI    ISTPE           ;CHECK IF A STATUS BYTE PARITY ERROR
1098  000701  0 115152 4 6 1152             JMPZ    RDISE           ;JUMP IF INITIAL SELECTION ERROR
1099  000702  0 117224 4 7 1224             JMPSUB  ACCEPT          ;ACCEPT THE STATUS
1100                                                                 ;START TAPE MOTION
1101
1102                                    ;SET UP THE DATA PATH INTERFACE
1103
1104  000703  0 001026 0 0 2 026             LDMAR   RDSET           ;SELECT READ SETUP LIST
1105  000704  0 047771 2 3 3 17 11          MOVMEM  IOSEL,I         ;SELECT DATA PATH FOR OUTPUT, MB FOR INPUT
1106  000705  0 124142 5 2 0 06 02          DATI    CNTL,BCLO       ;LOAD BYTE COUNTER
1107  000706  0 124163 5 2 0 07 03          DATI    CNTH,BCHO       ; FROM MASSBUS INTERFACE
1108  000707  0 044111 2 2 0 04 11          MOVMEM  MCLO            ;CLEAR MASSBUS COUNTER
1109  000710  0 045531 2 2 3 05 11          MOVMEM  MCHO,I
1110  000711  0 045611 2 2 3 10 11          MOVMEM  DFRMAD,I        ;LOAD FORMATTER START ADDRESS
1111  000712  0 045451 2 2 3 02 11          MOVMEM  REG2,I          ;SET MASTER END ON FORMATTER END
1112  000713  0 044011 2 2 0 00 11          MOVMEM  REG0            ;CLEAR FLAGS
1113  000714  0 044271 2 2 0 13 11          MOVMEM  HSDPIN          ;PULSE HIGH SPEED INIT
1114  000715  0 045431 2 2 3 01 11          MOVMEM  REG1,I          ;SET DX HIGH SPEED
1115  000716  0 047771 2 3 3 17 11          MOVMEM  IOSEL,I         ;SELECT MASSBUS INTERFACE
1116  000717  0 045431 2 2 3 01 11          MOVMEM  MPSCR1,I        ;START THE TRANSFER
1117
```

```
1118                                    ;WAIT FOR TRANSFER TO COMPLETE
1119
1120  000720  0 047771 2 3 3 17 11              MOVMEM  IOSEL,I                 ;SELECT DATA PATH INTERFACE
1121
1122                                    ;THREE NESTED LOOPS ARE REQUIRED TO PROVIDE A TIMEOUT OF 3 SECS.  THIS
1123                                    ;TIMEOUT IS NEEDED FOR THE CASE OF READING VIRGIN TAPES.
1124                                    ;THE READ TRANSFER IS CONSIDERED TO HAVE BEGUN WHEN THE BYTE COUNT CHANGES.
1125                                    ;THE CODE BETWEEN STARRED LINES WAS ADDED FOR EDIT [3]
1126                                    ;*****************************************************************
1127  000721  0 032026 1 5 0 01 06              DATI    BCLO,AC1                ;GET LOW ORDER BITS OF BYTE COUNT
1128  000722  0 002100 0 1 0 100                 LDBR    100                     ;SET OUTERMOST LOOP CNT
1129  000723  0 072111 3 5 0 04 11              MOVB    AC4
1130  000724  0 002177 0 1 0 177     LOOPO:    LDBR    177                     ;SET MIDDLE LOOP CNT
1131  000725  0 072071 3 5 0 03 11              MOVB    AC3
1132  000726  0 002177 0 1 0 177     LOOPM:    LDBR    177                     ;SET INNER LOOP CNT
1133  000727  0 072051 3 5 0 02 11              MOVB    AC2
1134  000730  0 103767 4 1 1767      LOOPI:    JMPI    INTERUPT                ;CHECK ON INTERRUPTS
1135  000731  0 032003 1 5 0 00 03              DATI    REG3,AC0                ;READ FORMATTER END FLAG
1136  000732  0 062005 3 1 0 00 05              SHL     AC0,BR                  ;MOVE TO BR4
1137  000733  0 106735 4 3 0735                 JMPB4   .+2                     ;STAY IN LOOP IF NO DEVICE END
1138  000734  0 100763 4 0 0763                 JMP     RDDONE                  ;ELSE, XFER IS DONE
1139  000735  0 022006 1 1 0 00 06              DATI    BCLO,BR                 ;GET PRESENT BYTE COUNT
1140  000736  0 060037 3 0 0 01 17              OSB     AC1                     ;COMPARE WITH INITIAL COUNT
1141  000737  0 114741 4 6 0741                 JMPZ    .+2                     ;JUMP IF COUNT HASN'T CHANGED
1142  000740  0 100755 4 0 0755                 JMP     NOVIRG                  ;JUMP OUT OF WAIT LOOP
1143  000741  0 072047 3 5 0 02 07              DECR    AC2                     ;DECR INNER LOOP CNT
1144  000742  0 114744 4 6 0744                 JMPZ    .+2                     ;JUMP IF COUNT IS ZERO
1145  000743  0 100730 4 0 0730                 JMP     LOOPI                   ;ELSE, STAY IN INNER LOOP
1146  000744  0 072067 3 5 0 03 07              DECR    AC3                     ;DECR MIDDLE LOOP CNT
1147  000745  0 114747 4 6 0747                 JMPZ    .+2                     ;JUMP IF COUNT IS ZERO
1148  000746  0 100726 4 0 0726                 JMP     LOOPM                   ;ELSE, STAY IN MIDDLE LOOP
1149  000747  0 072107 3 5 0 04 07              DECR    AC4                     ;DECR OUTER LOOP CNT
1150  000750  0 114752 4 6 0752                 JMPZ    .+2                     ;JUMP IF COUNT IS ZERO
1151  000751  0 100724 4 0 0724                 JMP     LOOPO                   ;ELSE, STAY IN OUTER LOOP
1152  000752  0 002107 0 1 0 107               LDBR    NRE.RT                  ;SETUP READ TIMEOUT ERROR CODE
1153  000753  0 072011 3 5 0 00 11              MOVB    AC0
1154  000754  0 101351 4 0 1351                JMP     RESET1                  ;GO RESET DRIVE AND REPORT ERROR
1155                                    ;*****************************************************************
1156
1157  000755  0 002304 0 1 0 304      NOVIRG:   GOSUB   GETFLG                  ;[5] GO RESTORE FLAGS INTO AC4
1158  000756  0 177631 7 7 3 11 11
1159  000757  0 103767 4 1 1767      WTLP:     JMPI    INTERUPT                ;CHECK ON INTERRUPTS
1160  000760  0 032003 1 5 0 00 03              DATI    REG3,AC0                ;READ FORMATTER END FLAG
1161  000761  0 062005 3 1 0 00 05              SHL     AC0,BR                  ;MOVE TO BR4
1162  000762  0 106757 4 3 0757                 JMPB4   WTLP                    ;WAIT FOR IT TO GO TO ZERO
1163  000763  0 103767 4 1 1767      RDDONE:   JMPI    INTERUPT                ;CHECK ON INTERRUPTS
1164  000764  0 022001 1 1 0 00 01              DATI    REG1,BR                 ;READ MASTER REQUEST
1165  000765  0 110767 4 4 0767                 JMPB7   .+2                     ;SKIP IF LAST WORD HAS BEEN SENT
1166  000766  0 100763 4 0 0763                 JMP     .-3                     ;NOT YET, KEEP WAITING
1167  000767  0 045431 2 2 3 01 11              MOVMEM  REG1,I                  ;CLEAR HIGH SPEED IN DATA PATH
1168
1169                                    ;READ THE REMAINING BYTE COUNT
1170
1171  000770  0 032066 1 5 0 03 06              DATI    BCLO,AC3                ;READ THE CONTENTS OF
1172  000771  0 032047 1 5 0 02 07              DATI    BCHO,AC2                ; THE BYTE COUNTER
```

```
1173  000772  0 103137 4 1 1137          JMPI    RDCHK                ;CHECK INTERRUPTS FOR PARITY ERRORS
1174
1175                                  ;GET STATUS FROM DEVICE
1176
1177  000773  0 047771 2 3 3 17 11      MOVMEM  IOSEL,I              ;SELECT CHANNEL BUS INTERFACE
1178  000774  0 117211 4 7 1211         JMPSUB  WAITST               ;GET THE DEVICE STATUS
1179  000775  0 103662 4 1 1662         JMPI    ESTPEC               ;CHECK IF A STATUS BYTE PARITY ERROR
1180  000776  0 115160 4 6 1160         JMPZ    RDESEC               ;JUMP IF AN ENDING STATUS ERROR
1181
1182                                  ;CHECK FOR DATA PARITY ERRORS
1183
1184  000777  0 047771 2 3 3 17 11      MOVMEM  IOSEL,I              ;SELECT MASSBUS INTERFACE
1185  001000  0 032007 1 5 0 00 07      DATI    MPDB2,AC0            ;READ DATA BUFFER PARITY ERROR BIT
1186  001001  0 062005 3 1 0 00 05      SHL     AC0,BR               ;MOVE TO BR BIT 4
1187  001002  0 107147 4 3 1147         JMPB4   RDPE                 ;JUMP IF DATA BUFFER PARITY ERROR
1188
1189                                  ;CHECK IF SIZE OF RECORD IS SAME AS EXPECTED
1190
1191  001003  0 060067 3 0 0 03 07      DEC     AC3                  ;CHECK IF ZERO
1192  001004  0 115006 4 6 1006         JMPZ    .+2                  ;SKIP IF SO
1193  001005  0 101046 4 0 1046         JMP     SIZERR               ;SIZE OF RECORD NOT AS EXPECTED
1194  001006  0 060047 3 0 0 02 07      DEC     AC2                  ;CHECK REST OF COUNT
1195  001007  0 115011 4 6 1011         JMPZ    .+2                  ; FOR ZERO
1196  001010  0 101046 4 0 1046         JMP     SIZERR               ;SIZE OF RECORD NOT AS EXPECTED
1197
```

```
1198                                    ;SEND AN EBL AND CHECK IF DONE
1199
1200  001011  0 045431 2 2 3 01 11             MOVMEM  MPSCR1,I            ;SEND AN EBL TO RH20
1201  001012  0 000000 0 0 0 000               NOP                         ;STALL
1202  001013  0 000000 0 0 0 000               NOP                         ; GIVE RUN TIME TO FALL
1203  001014  0 101015 4 0 1015                JMP     .+1
1204  001015  0 022000 1 1 0 00 00             DATI    MPSCR0,BR           ;READ DONE BIT
1205  001016  0 105043 4 2 1043                JMPB0   RAGAIN              ;JUMP IF RUN IS STILL ASSERTED
1206
1207                                    ;END OF READ COMMAND EXECUTION
1208
1209  001017  0 117065 4 7 1065                JMPSUB  RPTSIZ              ;REPORT THE SIZE OF RECORD READ
1210  001020  0 002011 0 1 0 011       XFREND: LDBR    MB                  ;SELECT MASSBUS INTERFACE
1211  001021  0 066371 3 3 0 17 11             MOVB    IOSEL
1212  001022  0 062030 3 1 0 01 10             MOV     AC1,BR              ;GET ENDING STATUS
1213  001023  0 064231 3 2 0 11 11             MOVB    STBYTE              ;PUT IN STATUS BYTE REGISTER
1214  001024  0 062110 3 1 0 04 10             MOV     AC4,BR              ;GET FLAGS
1215  001025  0 014000 0 6 0 000               SHR                         ;SHIFT FSENSE TO BR0
1216  001026  0 105040 4 2 1040                JMPB0   FRCSNS              ;JUMP IF SET
1217  001027  0 117234 4 7 1234                JMPSUB  DSELCT              ;DESELECT FROM DRVICE
1218  001030  0 002011 0 1 0 011       NRMXFR: LDBR    MB                  ;SELECT MASSBUS INTERFACE
1219  001031  0 066371 3 3 0 17 11             MOVB    IOSEL
1220  001032  0 002000 0 1 0 000               LDBR    0                   ;CLEAR THE FLAGS
1221  001033  0 064371 3 2 0 17 11             MOVB    FLAGS
1222  001034  0 103767 4 1 1767                JMPI    INTERUPT            ;CHECK ON INTERRUPTS
1223  001035  0 002002 0 1 0 002               LDBR    CLRGO               ;CLEAR GO BIT
1224  001036  0 064031 3 2 0 01 11             MOVB    MPSCR1
1225  001037  0 100026 4 0 0026                JMP     IDLE
1226
1227  001040  0 117734 4 7 1734       FRCSNS: JMPSUB  LOGREG              ;LOG THE REGISTERS
1228  001041  0 117602 4 7 1602               JMPSUB  CSENSE              ;READ THE SENSE BYTES WITH A CHAINED COMMAND
1229  001042  0 101030 4 0 1030               JMP     NRMXFR
1230
1231                                    ;REPEAT THE READ AGAIN
1232
1233  001043  0 046371 2 3 0 17 11     RAGAIN: MOVMEM  IOSEL               ;SELECT CHANNEL INTERFACE
1234  001044  0 117253 4 7 1253               JMPSUB  CHAIN               ;ACCEPT THE STATUS AND INDICATE CHAINING
1235  001045  0 100677 4 0 0677               JMP     SNDRD               ;PERFORM ANOTHER READ OPERATION
1236
```

```
1237                                    ;A RECORD WAS READ THAT WAS A DIFFERENT SIZE THAN EXPECTED
1238
1239  001046  0 117734 4 7 1734         SIZERR: JMPSUB  LOGREG                  ;LOG THE REGISTERS
1240  001047  0 117063 4 7 1063                 JMPSUB  RPTSZ                   ;REPORT SIZE OF RECORD READ
1241  001050  0 001146 0 0 2 146                LDMAR   DPR0                    ;SELECT SAVED CONTENTS OF REG0
1242  001051  0 042011 2 1 0 00 11              MOVMEM  BR                      ;COPY BC OVERFLOW FLAG TO BR
1243  001052  0 014000 0 6 0 000                SHR                             ;SHIFT TO BIT 0
1244  001053  0 014000 0 6 0 000                SHR
1245  001054  0 105057 4 2 1057                 JMPB0   .+3                     ;JUMP IF SET
1246  001055  0 002002 0 1 0 002                LDBR    SHREC.                  ;GET SHORT RECORD ERROR CODE
1247  001056  0 101060 4 0 1060                 JMP     .+2
1248  001057  0 002003 0 1 0 003                LDBR    LGREC.                  ;GET LONG RECORD ERROR CODE
1249  001060  0 064051 3 2 0 02 11              MOVB    MPECR                   ;PUT INTO ERROR CODE REGISTER
1250  001061  0 117602 4 7 1602                 JMPSUB  CSENSE                  ;READ THE SENSE BYTES
1251  001062  0 101666 4 0 1666                 JMP     RPTERR                  ;REPORT THE ERROR
1252
```

```
                                        SUBTTL  DATA TRANSFER PARITY ERROR REPORTERS

                                        ;REPORT SIZE OF DATA RECORD READ

001063  0 002011 0 1 0 011     RPTSZ:  LDBR    MB                      ;SELECT MASSBUS INTERFACE
001064  0 066371 3 3 0 17 11          MOVB    IOSEL
001065  0 062030 3 1 0 01 10  RPTSIZ: MOV     AC1,BR                  ;GET ENDING STATUS
001066  0 064231 3 2 0 11 11          MOVB    STBYTE                  ;PUT IN STATUS BYTE REGISTER
001067  0 122002 5 1 0 00 02          DATI    CNTL,BR                 ;READ BYTE COUNT INTO BR
001070  0 062076 3 1 0 03 16          TSB     AC3,BR                  ;SUBTRACT FROM BYTE COUNTER IN DATA PATH
001071  0 066051 3 3 0 02 11          MOVB    CNTL                    ;PUT BACK INTO MASSBUS INTERFACE
001072  0 122003 5 1 0 00 03          DATI    CNTH,BR                 ;GET HIGH BYTE OF COUNT
001073  0 062042 3 1 0 02 02          OSBC    AC2,BR                  ;COMPLETE THE SUBTRACTION
001074  0 066071 3 3 0 03 11          MOVB    CNTH                    ;STORE THE REST
001075  0 016000 0 7 0 000            RETURN

                                        ;WRITE DATA PARITY ERROR REPORTERS

001076  0 117767 4 7 1767      WRTCHK: JMPSUB  INTERUPT                ;CHECK IF A FATAL INTERRUPT
001077  0 117734 4 7 1734              JMPSUB  LOGREG                  ;LOG THE REGISTERS
001100  0 117724 4 7 1724              JMPSUB  CLRPE                   ;CLEAR THE PARITY ERROR
001101  0 117215 4 7 1215              JMPSUB  CHKSTS                  ;CHECK ENDING STATUS
001102  0 103643 4 1 1643              JMPI    ESTPE                   ;CHECK IF A STATUS BYTE PARITY ERROR
001103  0 115625 4 6 1625              JMPZ    ESTERR                  ;JUMP IF A STATUS ERROR
001104  0 002011 0 1 0 011             LDBR    MB                      ;SELECT MASSBUS INTERFACE
001105  0 066371 3 3 0 17 11          MOVB    IOSEL
001106  0 062030 3 1 0 01 10  XFRCHK: MOV     AC1,BR                  ;GET ENDING STATUS BYTE
001107  0 064231 3 2 0 11 11          MOVB    STBYTE                  ;PRESENT IT IN ITS REGISTER
001110  0 117724 4 7 1724              JMPSUB  CLRPE                   ;CLEAR THE PARITY ERRORS
001111  0 117602 4 7 1602              JMPSUB  CSENSE                  ;READ THE SENSE BYTES
001112  0 002011 0 1 0 011             LDBR    MB                      ;SELECT THE MASSBUS INTERFACE
001113  0 066371 3 3 0 17 11          MOVB    IOSEL
001114  0 002005 0 1 0 005             LDBR    REM.DP                  ;GET ERROR CODE
001115  0 064051 3 2 0 02 11          MOVB    MPECR                   ;LOAD ERROR CODE REGISTER
001116  0 101666 4 0 1666              JMP     RPTERR                  ;REPORT THE ERROR

001117  0 117734 4 7 1734      WRTPE:  JMPSUB  LOGREG                  ;LOG THE REGISTERS
001120  0 062030 3 1 0 01 10  XFRPE:  MOV     AC1,BR                  ;GET ENDING STATUS BYTE
001121  0 064231 3 2 0 11 11          MOVB    STBYTE                  ;PRESENT IT IN ITS REGISTER
001122  0 117602 4 7 1602              JMPSUB  CSENSE                  ;READ THE SENSE BYTES
001123  0 002011 0 1 0 011             LDBR    MB                      ;SELECT MASSBUS INTERFACE
001124  0 066371 3 3 0 17 11          MOVB    IOSEL
001125  0 002000 0 1 0 000             LDBR    0                       ;CLEAR OCCUPIED
001126  0 064031 3 2 0 01 11          MOVB    MPSCR1
001127  0 064371 3 2 0 17 11          MOVB    FLAGS                   ;CLEAR THE FLAG REGISTER
001130  0 002001 0 1 0 001             LDBR    EBL                     ;SEND AN EBL
001131  0 064031 3 2 0 01 11          MOVB    MPSCR1
001132  0 022000 1 1 0 00 00          DATI    MPSCR0,BR               ;READ RUN BIT
001133  0 105132 4 2 1132              JMPB0   .-1                     ;WAIT FOR IT TO CLEAR
001134  0 002202 0 1 0 202             LDBR    ATA+CLRGO               ;SET ATA AND CLEAR GO
001135  0 064031 3 2 0 01 11          MOVB    MPSCR1
001136  0 100026 4 0 0026              JMP     IDLE                    ;GO IDLE
```

```
1306                                    ;READ PARITY ERROR REPORTERS
1307
1308  001137  0 117767 4 7 1767   RDCHK:  JMPSUB  INTERUPT          ;CHECK IF A FATAL INTERRUPT
1309  001140  0 117734 4 7 1734           JMPSUB  LOGREG            ;LOG THE REGISTERS
1310  001141  0 117724 4 7 1724           JMPSUB  CLRPE             ;CLEAR THE PARITY ERROR
1311  001142  0 117211 4 7 1211           JMPSUB  WAITST            ;WAIT FOR ENDING STATUS FROM DRIVE
1312  001143  0 103662 4 1 1662           JMPI    ESTPEC            ;CHECK IF A STATUS BYTE PARITY ERROR
1313  001144  0 115160 4 6 1160           JMPZ    RDESEC            ;JUMP IF A STATUS ERROR
1314  001145  0 117063 4 7 1063           JMPSUB  RPTSZ             ;REPORT SIZE OF RECORD READ
1315  001146  0 101106 4 0 1106           JMP     XFRCHK
1316
1317  001147  0 117734 4 7 1734   RDPE:   JMPSUB  LOGREG            ;LOG THE REGISTERS
1318  001150  0 117065 4 7 1065           JMPSUB  RPTSIZ            ;REPORT SIZE OF RECORD READ
1319  001151  0 101120 4 0 1120           JMP     XFRPE
1320
1321  001152  0 117734 4 7 1734   RDISE:  JMPSUB  LOGREG            ;LOG THE REGISTERS
1322  001153  0 117165 4 7 1165           JMPSUB  CLRBC             ;CLEAR THE BYTE COUNTER
1323  001154  0 101615 4 0 1615           JMP     ISTERD
1324
1325  001155  0 117734 4 7 1734   RDESE0: JMPSUB  LOGREG            ;LOG THE REGISTERS
1326  001156  0 117165 4 7 1165           JMPSUB  CLRBC             ;CLEAR THE BYTE COUNTER
1327  001157  0 101626 4 0 1626           JMP     ESTERD
1328
1329  001160  0 117734 4 7 1734   RDESEC: JMPSUB  LOGREG            ;LOG THE REGISTERS
1330  001161  0 002011 0 1 0 011          LDBR    MB                ;SELECT MASSBUS INTERFACE
1331  001162  0 066371 3 3 0 17 11        MOVB    IOSEL
1332  001163  0 117065 4 7 1065           JMPSUB  RPTSIZ            ;REPORT SIZE OF RECORD READ
1333  001164  0 101626 4 0 1626           JMP     ESTERD
1334
1335  001165  0 002011 0 1 0 011   CLRBC:  LDBR    MB                ;SELECT MASSBUS INTERFACE
1336  001166  0 066371 3 3 0 17 11        MOVB    IOSEL
1337  001167  0 002000 0 1 0 000          LDBR    0                 ;GET A ZERO
1338  001170  0 066051 3 3 0 02 11        MOVB    CNTL              ;CLEAR THE BYTE COUNT REGISTER
1339  001171  0 066071 3 3 0 03 11        MOVB    CNTH              ; SO HOST KNOWS NO BYTES WERE READ
1340  001172  0 016000 0 7 0 000          RETURN
1341
```

```
1342                                    SUBTTL  SENSE OPERATIONS
1343
1344                                    ;A SENSE COMMAND HAS BEEN ISSUED
1345                                    ;READ SENSE INFORMATION FROM DRIVE
1346                                    ;AND STORE IN EXTENDED STATUS TABLE
1347
1348  001173  0 117712 4 7 1712     DOSNS:  JMPSUB  CLRSNS          ;CLEAR SENSE BYTE AREA
1349  001174  0 032132 1 5 0 05 12          DATI    DR,AC5          ;READ DRIVE NUMBER
1350  001175  0 062070 3 1 0 03 10          MOV     AC3,BR          ;LOAD SENSE COMMAND BYTE
1351  001176  0 072151 3 5 0 06 11          MOVB    AC6             ;INTO AC6
1352  001177  0 117270 4 7 1270             JMPSUB  SELECT          ;SELECT DEVICE AND ISSUE COMMAND
1353  001200  0 103622 4 1 1622             JMPI    ISTPE           ;CHECK IF A STATUS BYTE PARITY ERROR
1354  001201  0 115614 4 6 1614             JMPZ    ISTERR          ;JUMP IF INITIAL STATUS ERROR
1355  001202  0 117542 4 7 1542             JMPSUB  GETSNS          ;COPY SENSE DATA TO TABLE
1356  001203  0 002011 0 1 0 011            LDBR    MB              ;SELECT MASSBUS INTERFACE
1357  001204  0 066371 3 3 0 17 11          MOVB    IOSEL
1358  001205  0 062030 3 1 0 01 10          MOV     AC1,BR          ;GET ENDING STATUS BYTE
1359  001206  0 064231 3 2 0 11 11          MOVB    STBYTE          ;PUT IN STATUS BYTE REGISTER
1360  001207  0 117734 4 7 1734             JMPSUB  LOGREG          ;LOG ALL THE REGISTERS
1361  001210  0 100336 4 0 0336             JMP     NORMDN          ;REPORT NORMAL ENDING STATUS
1362
```

```
                                        SUBTTL  DEVICE STATUS ROUTINES


                                        ;WAIT FOR STATUS FROM CONTROL UNIT
                                        ;RETURN WITH STATUS IN AC1
                                        ;  AND Z SET IF ANY ERROR BIT IS IN STATUS

001211  0 103767 4 1 1767       WAITST: JMPI    INTERUPT                ;CHECK ON INTERRUPTS
001212  0 022004 1 1 0 00 04          DATI    TAGIN0,BR               ;READ THE TAG IN LINES
001213  0 105215 4 2 1215              JMPB0   .+2                     ;SKIP IF STATUS IN IS SET
001214  0 101211 4 0 1211              JMP     .-3                     ;NO, KEEP WAITING
001215  0 032027 1 5 0 01 07    CHKSTS: DATI    CBILO,AC1               ;READ THE STATUS BYTE INTO AC1
001216  0 002014 0 1 0 014             LDBR    14                      ;GET CHANNEL AND DEVICE END BITS
001217  0 062033 3 1 0 01 13          LANDB   AC1,BR                  ;GET COPY OF THESE BITS
001220  0 060037 3 0 0 01 17          OSB     AC1                     ;COMPARE REST OF BITS FOR ZEROS
001221  0 115223 4 6 1223              JMPZ    .+2                     ;SKIP IF Z IS SET
001222  0 016377 0 7 0 377             RETURN  -1                      ;NO, ERROR RETURN
001223  0 016000 0 7 0 000             RETURN                          ;NORMAL RETURN

```

```
                                        ;ACCEPT STATUS FROM DEVICE
                                        
001224  0 002356 0 1 0 356      ACCEPT: LDBR    SRVOUT+HLDOUT+SELOUT+TMREN+CLKOUT+MTROUT        ;SEND SERVICE OUT
001225  0 064051 3 2 0 02 11          MOVB    TORO
001226  0 103767 4 1 1767              JMPI    INTERUPT                ;CHECK ON INTERRUPTS
001227  0 022004 1 1 0 00 04          DATI    TAGIN0,BR               ;READ TAG IN LINES
001230  0 105226 4 2 1226              JMPB0   .-2                     ;WAIT FOR STATUS IN TO DROP
001231  0 002152 0 1 0 152             LDBR    HLDOUT+SELOUT+CLKOUT+MTROUT     ;DROP SERVICE OUT
001232  0 064051 3 2 0 02 11          MOVB    TORO
001233  0 016000 0 7 0 000             RETURN
                                        
                                        ;ACCEPT THE STATUS FROM DEVICE AND DESELECT
                                        
001234  0 002033 0 1 0 033      DSELCT: LDBR    CHN                     ;SELECT CHANNEL INTERFACE
001235  0 066371 3 3 0 17 11          MOVB    IOSEL
001236  0 002344 0 1 0 344             LDBR    SRVOUT+TMREN+CLKOUT+MTROUT      ;DROP SELECT OUT AND SET SERVICE OUT
001237  0 064051 3 2 0 02 11          MOVB    TORO
001240  0 103767 4 1 1767              JMPI    INTERUPT                ;CHECK ON INTERRUPTS
001241  0 022004 1 1 0 00 04          DATI    TAGIN0,BR               ;READ TAG IN LINES
001242  0 105240 4 2 1240              JMPB0   .-2                     ;WAIT FOR STATUS IN TO DROP
001243  0 002144 0 1 0 144             LDBR    TMREN+CLKOUT+MTROUT     ;CLEAR SERVICE OUT
001244  0 064051 3 2 0 02 11          MOVB    TORO
001245  0 103767 4 1 1767              JMPI    INTERUPT                ;CHECK ON INTERRUPTS
001246  0 022004 1 1 0 00 04          DATI    TAGIN0,BR               ;READ TAG IN LINES
001247  0 111245 4 4 1245              JMPB7   .-2                     ;WAIT FOR OPL IN TO DROP
001250  0 002040 0 1 0 040             LDBR    MTROUT                  ;TURN OFF THE TIMER
001251  0 064051 3 2 0 02 11          MOVB    TORO
001252  0 016000 0 7 0 000             RETURN

1410
```

```
1411                                    ;ACCEPT STATUS FROM DEVICE AND INDICATE CHAINING
1412
1413  001253  0 002240 0 1 0 240   CHAIN:  LDBR    OPLOUT+SUPOUT              ;SEND SUPPRESS OUT
1414  001254  0 064071 3 2 0 03 11         MOVB    TOR1
1415  001255  0 002344 0 1 0 344          LDBR    SRVOUT+TMREN+CLKOUT+MTROUT      ;DROP SELECT OUT, SET SERVICE OUT
1416  001256  0 064051 3 2 0 02 11         MOVB    TOR0
1417  001257  0 103767 4 1 1767           JMPI    INTERUPT                   ;CHECK ON INTERRUPTS
1418  001260  0 022004 1 1 0 00 04         DATI    TAGIN0,BR                  ;READ TAG IN LINES
1419  001261  0 105257 4 2 1257           JMPB0   .-2                        ;WAIT FOR SERVICE IN TO DROP
1420  001262  0 002144 0 1 0 144          LDBR    TMREN+CLKOUT+MTROUT        ;DROP SERVICE OUT
1421  001263  0 064051 3 2 0 02 11         MOVB    TOR0
1422  001264  0 103767 4 1 1767           JMPI    INTERUPT                   ;CHECK ON INTERRUPTS
1423  001265  0 022004 1 1 0 00 04         DATI    TAGIN0,BR                  ;READ TAG IN LINES
1424  001266  0 111264 4 4 1264           JMPB7   .-2                        ;WAIT FOR OPERATIONAL IN TO DROP
1425  001267  0 016000 0 7 0 000          RETURN
1426
1427
```

```
                                SUBTTL  SELECT DEVICE

                                ;SELECT DRIVE NUMBER IN AC5 AND
                                ;ISSUE COMMAND BYTE FROM AC6
                                ;RETURN WITH STATUS BYTE IN AC1
                                ;  UNLESS DRIVE IS NON-EXISTANT, IN WHICH CASE ABORT TO SELERR
                                ;ALSO USES AC0

001270  0 001166 0 0 2 166      SELECT: LDMAR   LSTCMD          ;SAVE COMMAND
001271  0 071550 3 4 3 06 10             MOV     AC6,MEM,I       ;  AND DRIVE NUMBER
001272  0 070130 3 4 0 05 10             MOV     AC5,MEM         ;  IN MEMORY
001273  0 001043 0 0 2 043      SELCT2: LDMAR   SELSET          ;SELECT LIST IN MEMORY
001274  0 047771 2 3 3 17 11             MOVMEM  IOSEL,I         ;SELECT CHANNEL INTERFACE
001275  0 062130 3 1 0 05 10             MOV     AC5,BR          ;GET DRIVE NUMBER FROM AC5
001276  0 064231 3 2 0 11 11             MOVB    BORLO           ;PUT ADDRESS ON BUS OUT LINES
001277  0 045451 2 2 3 02 11             MOVMEM  TOR0,I          ;SET ADDRESS OUT
001300  0 045451 2 2 3 02 11             MOVMEM  TOR0,I          ;SET SELECT OUT AND HOLD OUT
001301  0 103767 4 1 1767       SELRES: JMPI    INTERUPT        ;CHECK ON INTERRUPTS
001302  0 022004 1 1 0 00 04             DATI    TAGIN0,BR       ;READ TAG IN LINES
001303  0 111310 4 4 1310               JMPB7   SELCTD          ;JUMP IF SELECTED
001304  0 105402 4 2 1402               JMPB0   DRVBSY          ;DRIVE IS BUSY
001305  0 014000 0 6 0 000              SHR                     ;SHIFT SEL IN TO BR
001306  0 105333 4 2 1333               JMPB0   SELERR          ;GET OUT IF NO DRIVE RESPONDED
001307  0 101301 4 0 1301               JMP     SELRES          ;NO, WAIT FOR SELECTION RESPONSE

001310  0 045451 2 2 3 02 11     SELCTD: MOVMEM  TOR0,I          ;DROP ADDRESS OUT
001311  0 103767 4 1 1767               JMPI    INTERUPT        ;CHECK ON INTERRUPTS
001312  0 022004 1 1 0 00 04             DATI    TAGIN0,BR       ;READ TAG IN LINES
001313  0 107315 4 3 1315               JMPB4   .+2             ;CHECK FOR ADR IN
001314  0 101311 4 0 1311               JMP     .-3             ;NO, WAIT FOR IT
001315  0 045471 2 2 3 03 11           MOVMEM  TOR1,I          ;DROP SUPPRESS OUT
001316  0 022007 1 1 0 00 07           DATI    CBILO,BR        ;READ BUS IN LINES
001317  0 103617 4 1 1617               JMPI    ADRPE           ;GET OUT IF ADR PARITY ERROR
001320  0 060137 3 0 0 05 17           OSB     AC5             ;COMPARE WITH ADDRESS SENT
001321  0 115323 4 6 1323               JMPZ    .+2             ;SKIP IF A MATCH
001322  0 101347 4 0 1347               JMP     NOMTCH          ;NO, ERROR
001323  0 062150 3 1 0 06 10           MOV     AC6,BR          ;GET COMMAND
001324  0 064231 3 2 0 11 11           MOVB    BORLO           ;SEND TO DEVICE
001325  0 045451 2 2 3 02 11           MOVMEM  TOR0,I          ;SEND COMMAND OUT TO DRIVE
001326  0 103767 4 1 1767               JMPI    INTERUPT        ;CHECK ON INTERRUPTS
001327  0 022004 1 1 0 00 04           DATI    TAGIN0,BR       ;READ TAG IN LINES
001330  0 107326 4 3 1326               JMPB4   .-2             ;WAIT FOR ADR IN TO DROP
001331  0 045451 2 2 3 02 11           MOVMEM  TOR0,I          ;DROP COMMAND OUT
001332  0 101211 4 0 1211               JMP     WAITST          ;WAIT FOR STATUS

```

```
1473                                    ;DRIVE DOES NOT EXIST, ABORT COMMAND
1474
1475  001333  0 117734 4 7 1734         SELERR: JMPSUB  LOGREG                  ;LOG THE REGISTERS
1476  001334  0 002200 0 1 0 200                 LDBR    OPLOUT                  ;DROP SUPPRESS OUT
1477  001335  0 064071 3 2 0 03 11              MOVB    TOR1
1478  001336  0 002000 0 1 0 000                 LDBR    0                       ;DROP ADDRESS OUT AND SELECT OUT
1479  001337  0 064051 3 2 0 02 11              MOVB    TOR0
1480  001340  0 002011 0 1 0 011                 LDBR    MB                      ;SELECT MASSBUS INTERFACE
1481  001341  0 066371 3 3 0 17 11              MOVB    IOSEL
1482  001342  0 002004 0 1 0 004                 LDBR    DSE.                    ;GET ERROR CODE
1483  001343  0 064051 3 2 0 02 11              MOVB    MPECR                   ;LOAD INTO REGISTER
1484  001344  0 002000 0 1 0 000                 LDBR    0                       ;CLEAR STATUS BYTE
1485  001345  0 064231 3 2 0 11 11              MOVB    STBYTE                  ; REGISTER
1486  001346  0 101666 4 0 1666                  JMP     RPTERR                  ;REPORT THE ERROR
1487
1488                                    ;ADDRESS RECEIVED WAS NOT ADDRESS SENT
1489
1490  001347  0 002006 0 1 0 006         NOMTCH: LDBR    RE.AM                   ;GET ERROR NUMBER
1491  001350  0 072011 3 5 0 00 11       RESET:  MOVB    AC0                     ;SAVE ERROR NUMBER IN AC0
1492  001351  0 117734 4 7 1734         RESET1: JMPSUB  LOGREG                  ;LOG THE REGISTERS
1493
1494                                    ;EDIT [3] NEEDED AS PART OF READ TIMEOUT FIX
1495  001352  0 002033 0 1 0 033                 LDBR    CHN                     ;[3] SETUP IOSEL REG TO ACCESS CHANNEL
1496  001353  0 066371 3 3 0 17 11              MOVB    IOSEL                   ;[3] BUS INTERFACE
1497
1498  001354  0 002240 0 1 0 240                 LDBR    SUPOUT+OPLOUT           ;SET SUPPRESS OUT
1499  001355  0 064071 3 2 0 03 11              MOVB    TOR1
1500  001356  0 002040 0 1 0 040                 LDBR    SUPOUT                  ;CLEAR OPERATIONAL OUT TO PERFORM
1501  001357  0 064071 3 2 0 03 11              MOVB    TOR1                    ;  A SELECTIVE RESET
1502  001360  0 002040 0 1 0 040                 LDBR    MTROUT                  ;DROP ALL OTHER TAG LINES
1503  001361  0 064051 3 2 0 02 11              MOVB    TOR0
1504  001362  0 022004 1 1 0 00 04              DATI    TAGIN0,BR               ;READ TAG IN LINES
1505  001363  0 111362 4 4 1362                  JMPB7   .-1                     ;WAIT FOR OPL IN TO DROP
1506  001364  0 002040 0 1 0 040                 LDBR    MTROUT                  ;TURN OFF THE TIMER
1507  001365  0 064051 3 2 0 02 11              MOVB    TOR0
1508  001366  0 002240 0 1 0 240                 LDBR    OPLOUT+SUPOUT           ;SET OPERATIONAL OUT AGAIN
1509  001367  0 064071 3 2 0 03 11              MOVB    TOR1
1510  001370  0 002200 0 1 0 200                 LDBR    OPLOUT                  ;DROP SUPPRESS OUT
1511  001371  0 064071 3 2 0 03 11              MOVB    TOR1
1512  001372  0 117724 4 7 1724                  JMPSUB  CLRPE                   ;CLEAR PARITY ERRORS
1513  001373  0 002011 0 1 0 011                 LDBR    MB                      ;SELECT MASSBUS INTERFACE
1514  001374  0 066371 3 3 0 17 11              MOVB    IOSEL
1515  001375  0 062010 3 1 0 00 10              MOV     AC0,BR                  ;GET ERROR NUMBER
1516  001376  0 064051 3 2 0 02 11              MOVB    MPECR                   ;LOAD ERROR CODE REGISTER
1517  001377  0 002000 0 1 0 000                 LDBR    0                       ;CLEAR STATUS REGISTER,
1518  001400  0 064231 3 2 0 11 11              MOVB    STBYTE                  ; NONE RECEIVED
1519  001401  0 101666 4 0 1666                  JMP     RPTERR                  ;REPORT THE ERROR AND GO IDLE
1520
```

```
1521                                    SUBTTL  SHORT BUSY HANDLER
1522
1523  001402  0 002040 0 1 0 040     DRVBSY: LDBR    MTROUT          ;CLEAR TAG OUT LINES
1524  001403  0 064051 3 2 0 02 11           MOVB    TOR0
1525  001404  0 002200 0 1 0 200             LDBR    OPLOUT          ;DROP SELECT OUT AND ADDRESS OUT
1526  001405  0 064071 3 2 0 03 11           MOVB    TOR1
1527  001406  0 103767 4 1 1767              JMPI    INTERUPT        ;CHECK ON INTERRUPTS
1528  001407  0 022005 1 1 0 00 05           DATI    TAGIN1,BR       ;READ TAG IN LINES
1529  001410  0 014000 0 6 0 000             SHR                     ;SHIFT REQ IN TO BR4
1530  001411  0 107413 4 3 1413              JMPB4   .+2             ;SKIP IF SET
1531  001412  0 101406 4 0 1406              JMP     .-4             ;NO, WAIT FOR IT
1532  001413  0 117420 4 7 1420              JMPSUB  DEVREQ          ;SERVICE THE DEVICE REQUEST
1533  001414  0 001166 0 0 2 166             LDMAR   LSTCMD          ;GET LAST COMMAND ATTEMPTED
1534  001415  0 053551 2 5 3 06 11           MOVMEM  AC6,I           ;COPY COMMAND INTO AC6
1535  001416  0 052131 2 5 0 05 11           MOVMEM  AC5             ;COPY DRIVE ADDRESS INTO AC5
1536  001417  0 101273 4 0 1273              JMP     SELCT2          ;NOW SELECT AGAIN
1537
```

```
                                         SUBTTL  GET INTERRUPT STATUS FROM DEVICE

                                         ;ENTER WHEN REQUEST IN IS ASSERTED BY A DRIVE
                                         ;SELECT THE DRIVE AND LOOK AT STATUS BYTE
                                         ;IF DEVICE END IS SET WITHOUT UNIT CHECK AND ASYNC STATUS REGISTER IS ZERO:
                                         ;  ACCEPT THE STATUS AND PLACE IN ASYNC STATUS REGISTER
                                         ;IF DEVICE END IS SET WITHOUT UNIT CHECK AND ASYNC STATUS REGISTER IS NOT ZERO:
                                         ;  STACK THE STATUS
                                         ;IF UNIT CHECK IS SET:
                                         ;  ACCEPT THE STATUS AND READ THE DRIVE'S SENSE BYTES

001420  0 002156 0 1 0 156     DEVREQ: LDBR    HLDOUT+SELOUT+TMREN+CLKOUT+MTROUT       ;SET SELECT OUT
001421  0 064051 3 2 0 02 11          MOVB    TORO
001422  0 103767 4 1 1767      WATREQ: JMPI    INTERUPT                ;CHECK ON INTERRUPTS
001423  0 022004 1 1 0 00 04           DATI    TAGIN0,BR               ;READ TAG IN LINES
001424  0 111443 4 4 1443              JMPB7   REQSLD                  ;WAIT FOR OPERATIONAL IN
001425  0 014000 0 6 0 000             SHR                             ;SHIFT SELECT IN TO BR0
001426  0 105502 4 2 1502              JMPB0   NOREQ                   ;IF SET, NO DRIVE RESPONDED
001427  0 022005 1 1 0 00 05           DATI    TAGIN1,BR               ;[10] READ OTHER TAG IN LINES
001430  0 014000 0 6 0 000             SHR                             ;[10] SHIFT 'REQ IN' TO BIT 4
001431  0 107422 4 3 1422              JMPB4   WATREQ                  ;[10] IF STILL SET, KEEP WAITING
001432  0 000000 0 0 0 000             NOP                             ;[10] ELSE, DELAY A WHILE AND
001433  0 000000 0 0 0 000             NOP                             ;[10] CHECK
001434  0 000000 0 0 0 000             NOP                             ;[10] FOR
001435  0 000000 0 0 0 000             NOP                             ;[10] 'OPL IN'
001436  0 022004 1 1 0 00 04           DATI    TAGIN0,BR               ;[10] READ TAG IN LINES
001437  0 111443 4 4 1443              JMPB7   REQSLD                  ;[10] JUMP IF 'OPL IN' IS SET
001440  0 002040 0 1 0 040             LDBR    MTROUT                  ;[10] ELSE, DROP TAG OUT LINES
001441  0 064051 3 2 0 02 11          MOVB    TORO                    ;[10]
001442  0 016000 0 7 0 000             RETURN                          ;[10]

                                 ;[10]  JMP     WATREQ                  ;KEEP WAITING FOR RESPONSE

001443  0 103767 4 1 1767      REQSLD: JMPI    INTERUPT                ;CHECK ON INTERRUPTS
001444  0 022004 1 1 0 00 04           DATI    TAGIN0,BR               ;READ TAG IN LINES
001445  0 107447 4 3 1447              JMPB4   .+2                     ;WAIT FOR ADDRESS IN TO SET
001446  0 101443 4 0 1443              JMP     .-3
001447  0 001171 0 0 2 171             LDMAR   INTDRV                  ;SELECT MEMORY LOCATION FOR ADDRESS
001450  0 030007 1 4 0 00 07           DATI    CBILO,MEM               ;READ DEVICE ADDRESS
001451  0 002157 0 1 0 157             LDBR    HLDOUT+SELOUT+CMDOUT+TMREN+CLKOUT+MTROUT        ;SET COMMAND OUT
001452  0 064051 3 2 0 02 11          MOVB    TORO
001453  0 103767 4 1 1767              JMPI    INTERUPT                ;CHECK ON INTERRTUPTS
001454  0 022004 1 1 0 00 04           DATI    TAGIN0,BR               ;READ TAG IN LINES
001455  0 107453 4 3 1453              JMPB4   .-2                     ;WAIT FOR ADDRESS IN TO DROP
001456  0 002156 0 1 0 156             LDBR    HLDOUT+SELOUT+TMREN+CLKOUT+MTROUT       ;DROP COMMAND OUT
001457  0 064051 3 2 0 02 11          MOVB    TORO
001460  0 103767 4 1 1767              JMPI    INTERUPT                ;CHECK ON INTERRUPTS
001461  0 022004 1 1 0 00 04           DATI    TAGIN0,BR               ;READ TAG IN LINES
001462  0 105464 4 2 1464              JMPB0   .+2                     ;WAIT FOR STATUS IN TO SET
001463  0 101460 4 0 1460              JMP     .-3
001464  0 001170 0 0 2 170             LDMAR   INTSTS                  ;SELECT MEMORY LOCATION FOR STATUS
001465  0 030007 1 4 0 00 07           DATI    CBILO,MEM               ;READ STATUS BYTE
001466  0 042011 2 1 0 00 11          MOVMEM  BR                      ;GET STATUS TO BR
001467  0 014000 0 6 0 000             SHR                             ;SHIFT UNIT CHECK TO BR0
```

```
 1593  001470  0 105526 4 2 1526          JMPB0   RDSNS                ;READ SENSE BYTES IF SET
 1594  001471  0 014000 0 6 0 000          SHR                          ;SHIFT DEVICE END TO BR0
 1595  001472  0 105505 4 2 1505          JMPB0   RPTSTS               ;REPORT STATUS IF SET
 1596  001473  0 042011 2 1 0 00 11       MOVMEM  BR                   ;[xx]
 1597  001474  0 014000 0 6 0 000          SHR                          ;[xx]
 1598  001475  0 107500 4 3 1500          JMPB4   FOO                  ;[XX]
 1599  001476  0 002106 0 1 0 106          JUMP    HALT                 ;[XX]
 1600  001477  0 160231 7 0 0 11 11
 1601  001500                       FOO:                               ;[XX]
 1602                                ;[6]    JMPSUB  DSELCT               ;ACCEPT THE STATUS
 1603  001500  0 117253 4 7 1253          JMPSUB  CHAIN                ;[6]ACCEPT STATUS AND INDICATE CHAINING
 1604  001501  0 016000 0 7 0 000          RETURN
 1605
```

```
1606  001502  0 002040 0 1 0 040      NOREQ:  LDBR    MTROUT                  ;DROP SELECT OUT
1607  001503  0 064051 3 2 0 02 11            MOVB    TOR0
1608  001504  0 016000 0 7 0 000             RETURN
1609
1610  001505  0 002011 0 1 0 011      RPTSTS: LDBR    MB                      ;SELECT MASSBUS INTERFACE
1611  001506  0 066371 3 3 0 17 11            MOVB    IOSEL
1612  001507  0 132001 5 5 0 00 01            DATI    ASYCST,AC0              ;READ ASYNC STATUS REGISTER
1613  001510  0 060007 3 0 0 00 07            DEC     AC0                     ;DECREMENT TO CHECK IF CURRENTLY ZERO
1614  001511  0 115517 4 6 1517               JMPZ    RPTNOW                  ;IF ZERO, REPORT NOW
1615  001512  0 002033 0 1 0 033              LDBR    CHN                     ;SELECT CHANNEL BUS INTERFACE
1616  001513  0 066371 3 3 0 17 11            MOVB    IOSEL
1617                                  ;[8]    JMPSUB  STACK                   ;STACK THE STATUS
1618  001514  0 002345 0 1 0 345              GOSUB   STACK                   ;[8] STACK THE STATUS
1619  001515  0 177631 7 7 3 11 11
1620  001516  0 016000 0 7 0 000             RETURN                          ;REPORT IT LATER
1621
1622                                  ;[8]RPTNOW:     MOVMEM  ASYCST,I                ;REPORT THE STATUS
1623  001517  0 043411 2 1 3 00 11    RPTNOW: MOVMEM  BR,I                    ;[8] GET THE STATUS INTO THE BR
1624  001520  0 046011 2 3 0 00 11            MOVMEM  ASYCDR                  ;REPORT THE DRIVE ADDRESS
1625  001521  0 066031 3 3 0 01 11            MOVB    ASYCST                  ;[8] REPORT THE STATUS
1626  001522  0 002033 0 1 0 033              LDBR    CHN                     ;SELECT CHANNEL BUS INTERFACE
1627  001523  0 066371 3 3 0 17 11            MOVB    IOSEL
1628  001524  0 117234 4 7 1234              JMPSUB  DSELCT                  ;ACCEPT THE STATUS AND DESELECT
1629  001525  0 016377 0 7 0 377             RETURN  -1                      ;SET Z TO SAY STATUS WAS PRESENTED
1630
1631  001526  0 117253 4 7 1253      RDSNS:  JMPSUB  CHAIN                   ;ACCEPT THE STATUS
1632  001527  0 003401 0 1 3 001              LDBR    1,I                     ;SET BIT 0 IN BR
1633  001530  0 052131 2 5 0 05 11            MOVMEM  AC5                     ;PUT DRIVE ADDRESS IN AC5
1634  001531  0 117533 4 7 1533              JMPSUB  SENSE                   ;READ THE SENSE BYTES
1635  001532  0 016000 0 7 0 000             RETURN
1636
```

```
                                 GET INTERRUPT STATUS FROM DEVICE

                                 ;READ SENSE BYTES FROM DRIVE NUMBER IN AC5
                                 ;TRANSFER BYTES TO MEMORY EXTENDED STATUS TABLE IF
                                 ;BR BIT 0 IS NOT SET ON ENTRY

                                 ;USES AC2

001533  0 072051 3 5 0 02 11   SENSE:  MOVB    AC2                     ;SAVE CONTROL BIT IN BIT 0
001534  0 002004 0 1 0 004             LDBR    4                       ;GET SENSE COMMAND
001535  0 072151 3 5 0 06 11           MOVB    AC6                     ;  INTO AC6
001536  0 117273 4 7 1273              JMPSUB  SELCT2                  ;SELECT THE DEVICE AND ISSUE SENSE COMMAND
001537  0 115600 4 6 1600              JMPZ    XITSNS                  ;JUMP IF INITIAL STATUS ERROR
001540  0 062050 3 1 0 02 10           MOV     AC2,BR                  ;GET CONTROL BIT
001541  0 105563 4 2 1563              JMPB0   ENDSNS                  ;DON'T READ BYTES IF BIT 0 IS SET
001542  0 117224 4 7 1224      GETSNS: JMPSUB  ACCEPT                  ;ACCEPT THE STATUS
001543  0 001052 0 0 2 052             LDMAR   SBYT00                  ;SELECT MEMORY TABLE ADDRESS
001544  0 103767 4 1 1767      NXTSNS: JMPI    INTERUPT                ;CHECK ON INTERRUPTS
001545  0 022004 1 1 0 00 04           DATI    TAGIN0,BR               ;READ TAG IN LINES
001546  0 105577 4 2 1577              JMPB0   LSTSNS                  ;END OF SENSE COMMAND IF STATUS IN IS SET
001547  0 022005 1 1 0 00 05           DATI    TAGIN1,BR               ;READ TAG LINES
001550  0 111552 4 4 1552              JMPB7   .+2                     ;SKIP IF SERVICE IN IS SET
001551  0 101544 4 0 1544              JMP     NXTSNS                  ;NO, KEEP WAITING
001552  0 031407 1 4 3 00 07           DATI    CBILO,MEM,I             ;READ SENSE BYTE, PLACE IN MEMORY
001553  0 002356 0 1 0 356             LDBR    SRVOUT+HLDOUT+SELOUT+TMREN+CLKOUT+MTROUT        ;SET SERVICE OUT
001554  0 064051 3 2 0 02 11           MOVB    TOR0
001555  0 103767 4 1 1767              JMPI    INTERUPT                ;CHECK ON INTERRUPTS
001556  0 022005 1 1 0 00 05           DATI    TAGIN1,BR               ;WAIT FOR SERVICE IN TO DROP
001557  0 111555 4 4 1555              JMPB7   .-2
001560  0 002156 0 1 0 156             LDBR    HLDOUT+SELOUT+TMREN+CLKOUT+MTROUT       ;DROP SERVICE OUT
001561  0 064051 3 2 0 02 11           MOVB    TOR0
001562  0 101544 4 0 1544              JMP     NXTSNS                  ;WAIT FOR NEXT BYTE

001563  0 117224 4 7 1224      ENDSNS: JMPSUB  ACCEPT                  ;ACCEPT THE STATUS
001564  0 103767 4 1 1767              JMPI    INTERUPT                ;CHECK ON INTERRUPTS
001565  0 022005 1 1 0 00 05           DATI    TAGIN1,BR               ;WAIT FOR SERVICE IN TO SET
001566  0 111570 4 4 1570              JMPB7   .+2
001567  0 101564 4 0 1564              JMP     .-3
001570  0 002157 0 1 0 157             LDBR    HLDOUT+SELOUT+CMDOUT+TMREN+CLKOUT+MTROUT        ;SET COMMAND OUT
001571  0 064051 3 2 0 02 11           MOVB    TOR0
001572  0 103767 4 1 1767              JMPI    INTERUPT                ;CHECK ON INTERRUPTS
001573  0 022005 1 1 0 00 05           DATI    TAGIN1,BR               ;WAIT FOR SERVICE IN TO DROP
001574  0 111572 4 4 1572              JMPB7   .-2
001575  0 002156 0 1 0 156             LDBR    HLDOUT+SELOUT+TMREN+CLKOUT+MTROUT       ;DROP COMMAND OUT
001576  0 064051 3 2 0 02 11           MOVB    TOR0
001577  0 117211 4 7 1211      LSTSNS: JMPSUB  WAITST                  ;WAIT FOR STATUS BYTE
001600  0 117234 4 7 1234      XITSNS: JMPSUB  DSELCT                  ;DESELECT FROM DEVICE
001601  0 016000 0 7 0 000             RETURN

001602  0 062110 3 1 0 04 10   CSENSE: MOV     AC4,BR                  ;READ FLAGS
001603  0 014000 0 6 0 000             SHR                             ;SHIFT INHIBIT SENSE BIT
001604  0 014000 0 6 0 000             SHR                             ; TO BR0
001605  0 105600 4 2 1600              JMPB0   XITSNS                  ;IF SET, DESELECT FROM DEVICE NOW
001606  0 002033 0 1 0 033             LDBR    CHN                     ;SELECT CHANNEL BUS INTERFACE
001607  0 066371 3 3 0 17 11           MOVB    IOSEL
001610  0 117253 4 7 1253              JMPSUB  CHAIN                   ;ACCEPT DEVICE STATUS AND INDICATE CHAINING
001611  0 002000 0 1 0 000             LDBR    0                       ;CLEAR BR0
```

```
1692  001612  0 117533 4 7 1533          JMPSUB  SENSE              ;READ THE SENSE BYTES
1693  001613  0 016000 0 7 0 000          RETURN
1694
```

```
                                SUBTTL  STATUS ERRORS

                                ;INITIAL STATUS ERROR

001614  0 117734 4 7 1734      ISTERR: JMPSUB  LOGREG          ;LOG THE REGISTERS
001615  0 002021 0 1 0 021      ISTERD: LDBR    UDS.IN          ;GET ERROR NUMBER
001616  0 101627 4 0 1627              JMP     STERR

001617  0 117767 4 7 1767      ADRPE:  JMPSUB  INTERUPT        ;CHECK IF A FATAL INTERRUPT
001620  0 002026 0 1 0 026              LDBR    RE.AP           ;GET ERROR CODE
001621  0 101350 4 0 1350              JMP     RESET           ;RESET THE DEVICE FIRST

001622  0 117767 4 7 1767      ISTPE:  JMPSUB  INTERUPT        ;CHECK IF A FATAL INTERRUPT
001623  0 002046 0 1 0 046              LDBR    RE.SP           ;GET ERROR CODE
001624  0 101350 4 0 1350              JMP     RESET           ;RESET THE DEVICE FIRST
```

```
1711                                        ;ENDING STATUS ERROR
1712
1713  001625  0 117734 4 7 1734     ESTERR: JMPSUB  LOGREG          ;LOG THE REGISTERS
1714  001626  0 002001 0 1 0 001     ESTERD: LDBR    UDS.FN          ;GET ERROR NUMBER
1715  001627  0 072011 3 5 0 00 11   STERR:  MOVB    AC0             ;SAVE IN AC0
1716  001630  0 002011 0 1 0 011             LDBR    MB              ;SELECT MASSBUS INTERFACE
1717  001631  0 066371 3 3 0 17 11           MOVB    IOSEL
1718  001632  0 062010 3 1 0 00 10           MOV     AC0,BR          ;GET ERROR NUMBER
1719  001633  0 064051 3 2 0 02 11           MOVB    MPECR           ;LOAD ERROR CODE REGISTER
1720  001634  0 062030 3 1 0 01 10           MOV     AC1,BR          ;GET STATUS BYTE
1721  001635  0 064231 3 2 0 11 11           MOVB    STBYTE          ;LOAD INTO STATUS REGISTER
1722  001636  0 107641 4 3 1641             JMPB4   BSYERR          ;JUMP AROUND IF BUSY STATUS
1723  001637  0 117602 4 7 1602     ERRSNS: JMPSUB  CSENSE          ;READ THE SENSE BYTES
1724  001640  0 101666 4 0 1666             JMP     RPTERR          ;REPORT THE ERROR
1725
1726  001641  0 117234 4 7 1234     BSYERR: JMPSUB  DSELCT          ;DESELECT FROM DEVICE
1727  001642  0 101666 4 0 1666             JMP     RPTERR          ;REPORT THE ERROR
1728
1729  001643  0 117767 4 7 1767     ESTPE:  JMPSUB  INTERUPT        ;CHECK IF A FATAL INTERRUPT
1730  001644  0 117734 4 7 1734             JMPSUB  LOGREG          ;NO, LOG THE REGISTERS
1731  001645  0 002011 0 1 0 011             LDBR    MB              ;SELECT MASSBUS INTERFACE
1732  001646  0 066371 3 3 0 17 11           MOVB    IOSEL
1733  001647  0 062030 3 1 0 01 10   ESTPEX: MOV     AC1,BR          ;GET STATUS BYTE
1734  001650  0 064231 3 2 0 11 11           MOVB    STBYTE          ;PRESENT IT
1735  001651  0 002025 0 1 0 025             LDBR    REM.SP          ;GET ERROR CODE
1736  001652  0 064051 3 2 0 02 11           MOVB    MPECR           ;LOAD IT
1737  001653  0 117724 4 7 1724             JMPSUB  CLRPE           ;CLEAR THE PARITY ERROR
1738  001654  0 117602 4 7 1602             JMPSUB  CSENSE          ;READ THE SENSE BYTES
1739  001655  0 101666 4 0 1666             JMP     RPTERR          ;REPORT THE ERROR
1740
1741  001656  0 117767 4 7 1767     ESTPE0: JMPSUB  INTERUPT        ;CHECK IF A FATAL INTERRUPT
1742  001657  0 117734 4 7 1734             JMPSUB  LOGREG          ;NO, LOG THE REGISTERS
1743  001660  0 117165 4 7 1165             JMPSUB  CLRBC           ;CLEAR THE BYTE COUNT REGISTER
1744  001661  0 101647 4 0 1647             JMP     ESTPEX
1745
1746  001662  0 117767 4 7 1767     ESTPEC: JMPSUB  INTERUPT        ;CHECK IF A FATAL INTERRUPT
1747  001663  0 117734 4 7 1734             JMPSUB  LOGREG          ;NO, LOG THE REGISTERS
1748  001664  0 117065 4 7 1065             JMPSUB  RPTSIZ          ;REPORT THE SIZE OF RECORD READ
1749  001665  0 101647 4 0 1647             JMP     ESTPEX
1750
```

```
                                        SUBTTL  ERROR REPORTER

                                        ;THIS ROUTINE WILL PROPERLY TERMINATE FROM THE RH20
                                        ;AND REPORT AN ERROR TO THE HOST BY SETTING MPERR AND ATA

                                        ;THE ERROR CODE MUST BE LOADED BEFORE CALLING THIS ROUTINE
                                        ;THIS ROUTINE ALWAYS EXITS DIRECTLY TO THE IDLE LOOP

001666  0 002011 0 1 0 011     RPTERR: LDBR    MB                      ;SELECT THE MASSBUS INTERFACE
001667  0 066371 3 3 0 17 11            MOVB    IOSEL
001670  0 002000 0 1 0 000             LDBR    0                       ;CLEAR THE FLAG REGISTER
001671  0 064371 3 2 0 17 11            MOVB    FLAGS
001672  0 022001 1 1 0 00 01            DATI    MPSCR1,BR               ;READ THE STATUS REGISTER
001673  0 107677 4 3 1677                JMPB4   XFRERR                  ;JUMP IF OCCUPIED IS ASSERTED
001674  0 002302 0 1 0 302             LDBR    ATA+MPERR+CLRGO         ;SET ERROR AND INTERRUPT THE HOST
001675  0 064031 3 2 0 01 11            MOVB    MPSCR1
001676  0 100026 4 0 0026              JMP     IDLE

001677  0 002120 0 1 0 120     XFRERR: LDBR    MPERR+OCC               ;SET MPERR TO CAUSE EXCEPTION
001700  0 064031 3 2 0 01 11            MOVB    MPSCR1
001701  0 002100 0 1 0 100             LDBR    MPERR                   ;CLEAR OCCUPIED
001702  0 064031 3 2 0 01 11            MOVB    MPSCR1
001703  0 002101 0 1 0 101             LDBR    MPERR+EBL               ;SEND AN EBL
001704  0 064031 3 2 0 01 11            MOVB    MPSCR1
001705  0 022000 1 1 0 00 00            DATI    MPSCR0,BR               ;WAIT FOR RUN TO DROP
001706  0 105705 4 2 1705              JMPB0   .-1
001707  0 002302 0 1 0 302             LDBR    MPERR+ATA+CLRGO         ;SET ATA AND CLEAR GO
001710  0 064031 3 2 0 01 11            MOVB    MPSCR1
001711  0 100026 4 0 0026              JMP     IDLE                    ;GO IDLE
```

```
                                         SUBTTL  CLEAR SENSE BYTE STORAGE AREA

                                         ;THIS ROUTINE WILL CLEAR THE SENSE BYTE STORAGE AREA IN MEMORY
                                         ;TO INDICATE THAT NO SENSE BYTES WERE READ

001712  0 000400 0 0 1 000       CLRSNS: LDMARX  0                       ;SELECT MEMORY BANK 0
001713  0 001052 0 0 2 052               LDMAR   SBYT00                  ;SELECT MEMORY ADDRESS
001714  0 002027 0 1 0 027               LDBR    ^D24-1                  ;GET COUNT OF SENSE BYTE
001715  0 072011 3 5 0 00 11             MOVB    AC0                     ; LOCATIONS
001716  0 011400 0 4 3 000               LDMEM   0,I                     ;CLEAR A LOCATION, INCREMENT TO NEXT
001717  0 072007 3 5 0 00 07             DECR    AC0                     ;DECREMENT COUNT
001720  0 113716 4 5 1716               JMPC    .-2                     ;LOOP IF STILL POSITIVE
001721  0 002200 0 1 0 200               LDBR    STATRQ                  ;SET REQUEST BIT
001722  0 064211 3 2 0 10 11             MOVB    STINDX                  ; TO GET INDEX ZERO AT END OF COMMAND
001723  0 016000 0 7 0 000               RETURN
```

```
                                        SUBTTL  CLEAR PARITY ERRORS IN DX20

001724  0 002022 0 1 0 022      CLRPE:  LDBR    DP                      ;SELECT THE DATA PATH
001725  0 066371 3 3 0 17 11           MOVB    IOSEL
001726  0 064031 3 2 0 01 11           MOVB    REG1                    ;CLEAR DX HIGH SPEED
001727  0 064011 3 2 0 00 11           MOVB    REG0                    ;CLEAR DATA PATH ERROR FLAGS
001730  0 002033 0 1 0 033             LDBR    CHN                     ;SELECT CHANNEL INTERFACE
001731  0 066371 3 3 0 17 11           MOVB    IOSEL
001732  0 064011 3 2 0 00 11           MOVB    CSR0                    ; ERROR FLAGS
001733  0 016000 0 7 0 000             RETURN
```

```
1808                                    SUBTTL  INTERNAL REGISTER LOGGING ROUTINE
1809
1810                                    ;THIS ROUTINE WILL LOG ALL INTERNAL REGISTERS INTO
1811                                    ;THE EXTENDED STATUS AREA IN MEMORY
1812                                    ;FOR EDIT [3], THE JMPSUBS TO THE 'LOG' ROUTINES WERE CHANGED TO GOSUBS.
1813
1814  001734  0 000400 0 0 1 000      LOGREG: LDMARX  0                       ;SELECT MEMORY PAGE 0
1815  001735  0 001104 0 0 2 104              LDMAR   STAT                    ;SELECT MEMORY LOCATION
1816  001736  0 131416 5 4 3 00 16           DATI    MPSTAT,MEM,I            ;LOG STATUS REGISTER
1817  001737  0 131417 5 4 3 00 17           DATI    IOSEL,MEM,I             ;LOG I/O SELECT REGISTER
1818  001740  0 002011 0 1 0 011            LDBR    MB                      ;SELECT MASSBUS INTERFACE
1819  001741  0 066371 3 3 0 17 11           MOVB    IOSEL
1820  001742  0 002312 0 1 0 312            GOSUB   LOG0                    ;[3]LOG REGISTERS 0-7
1821  001743  0 177631 7 7 3 11 11
1822  001744  0 002323 0 1 0 323            GOSUB   LOG10                   ;[3]LOG REGISTERS 10-17
1823  001745  0 177631 7 7 3 11 11
1824  001746  0 002334 0 1 0 334            GOSUB   LOG20                   ;[3]LOG REGISTERS 20-27
1825  001747  0 177631 7 7 3 11 11
1826  001750  0 002033 0 1 0 033            LDBR    CHN                     ;SELECT CHANNEL BUS INTERFACE
1827  001751  0 066371 3 3 0 17 11           MOVB    IOSEL
1828  001752  0 002312 0 1 0 312            GOSUB   LOG0                    ;[3]LOG REGISTERS 0-7
1829  001753  0 177631 7 7 3 11 11
1830  001754  0 002003 0 1 0 003            LDBR    3                       ;WRITE CSR0 TO CLEAR ANY
1831  001755  0 064011 3 2 0 00 11           MOVB    CSR0                    ; PARITY ERRORS MAY HAVE SET BY DATI'S
1832  001756  0 002022 0 1 0 022            LDBR    DP                      ;SELECT DATA PATH INTERFACE
1833  001757  0 066371 3 3 0 17 11           MOVB    IOSEL
1834  001760  0 002312 0 1 0 312            GOSUB   LOG0                    ;[3]LOG REGISTERS 0-7
1835  001761  0 177631 7 7 3 11 11
1836  001762  0 002323 0 1 0 323            GOSUB   LOG10                   ;[3]LOG REGISTERS 10-17
1837  001763  0 177631 7 7 3 11 11
1838  001764  0 001105 0 0 2 105            LDMAR   IOREG                   ;GET SAVE I/O SELECT REGISTER
1839  001765  0 046371 2 3 0 17 11           MOVMEM  IOSEL                   ;RESTORE IT
1840  001766  0 016000 0 7 0 000            RETURN
1841
```

```
                                SUBTTL  CRAM BANK LINKAGE

001767  0 002061 0 1 0 061      INTERUPT:JUMP   INTRPT              ;JUMP TO HIGH BANK TO ROUTINE
001770  0 160231 7 0 0 11 11

002000                          .LOC    2000                        ;SKIP TO NEXT BANK
```

```
1849                                    SUBTTL  DUMP STATUS ROUTINE
1850
1851
1852                                    ;A DUMP STATUS COMMAND HAS BEEN RECEIVED
1853                                    ;DUMP THE CURRENT CONTENTS OF THE EXTENDED STATUS TABLE
1854                                    ;OVER THE MASSBUS DATA INTERFACE
1855
1856  002000  1 002020 0 1 0 020    DODUMP: LDBR    OCC                     ;SET OCCUPIED
1857  002001  1 064031 3 2 0 01 11           MOVB    MPSCR1                  ;SEND TO RH20
1858
1859                                    ;WAIT FOR RH20 TO ASSERT RUN
1860
1861  002002  1 022000 1 1 0 00 00           DATI    MPSCR0,BR               ;READ RUN BIT
1862  002003  1 104005 4 2 0005              JMPB0   .+2                     ;SKIP IF SET
1863  002004  1 116035 4 7 0035              JMPSUB  WAITRN                  ;NO, WAIT FOR IT
1864
1865                                    ;SET UP AND TRANSFER DATA TO RH20
1866
1867  002005  1 002047 0 1 0 047    DMPSET: LDBR    MAXIDX*2+1              ;GET COUNT OF HALF WORDS TO WRITE
1868  002006  1 072011 3 5 0 00 11           MOVB    AC0                     ;SAVE IN AC0
1869  002007  1 002000 0 1 0 000             LDBR    0                       ;CLEAR THE TWO UNUSED
1870  002010  1 064171 3 2 0 07 11           MOVB    MPDB2                   ; BITS IN THE DATA REGISTER
1871  002011  1 000400 0 0 1 000             LDMARX  0                       ;SELECT MEMORY BANK 0
1872  002012  1 001052 0 0 2 052             LDMAR   SBYT00                  ;SELECT EXTENDED STATUS TABLE
1873  002013  1 002024 0 1 0 024             LDBR    OCC+START               ;GET START BIT INTO BR
1874  002014  1 045551 2 2 3 06 11  DATDMP: MOVMEM  MPDB1,I                 ;LOAD TWO BYTES OF STATUS
1875  002015  1 045531 2 2 3 05 11           MOVMEM  MPDB0,I                 ; INTO DATA BUFFER
1876  002016  1 064031 3 2 0 01 11           MOVB    MPSCR1                  ;SEND AN SCLK
1877  002017  1 000000 0 0 0 000             NOP                             ;ASSURE 1200 NS FOR SCLK
1878  002020  1 072007 3 5 0 00 07           DECR    AC0                     ;DECREMENT HALF WORD COUNT
1879  002021  1 112014 4 5 0014              JMPC    DATDMP                  ;REPEAT UNTIL ALL BYTES ARE SENT
1880
1881                                    ;ALL DATA HAS BEEN TRANSFERRED
1882                                    ;SEND AN EBL TO TERMINATE
1883
1884  002022  1 002021 0 1 0 021             LDBR    OCC+EBL                 ;SEND AN EBL TO RH20
1885  002023  1 064031 3 2 0 01 11           MOVB    MPSCR1
1886  002024  1 002000 0 1 0 000             LDBR    0                       ;CLEAR THE FLAGS
1887  002025  1 064371 3 2 0 17 11           MOVB    FLAGS
1888  002026  1 000000 0 0 0 000             NOP                             ;WAIT FOR EBL TO DROP
1889  002027  1 022000 1 1 0 00 00           DATI    MPSCR0,BR               ;READ RUN BIT
1890  002030  1 104005 4 2 0005              JMPB0   DMPSET                  ;SET UP, REPEAT IF STILL SET
1891  002031  1 002002 0 1 0 002             LDBR    CLRGO                   ;CLEAR GO AND OCCUPIED
1892  002032  1 064031 3 2 0 01 11           MOVB    MPSCR1
1893  002033  1 002026 0 1 0 026             JUMP    IDLE
1894  002034  1 160211 7 0 0 10 11
1895
```

```
                                         SUBTTL  WAIT FOR RH20 TO ASSERT RUN

002035  1 002353 0 1 0 353      WAITRN: LDBR    (<-<^D10M/^D1950>_-8>&377)      ;GET COUNT TO WAIT
002036  1 072011 3 5 0 00 11           MOVB    AC0                             ; 10 MILLI-SECONDS
002037  1 002370 0 1 0 370            LDBR    (-<^D10M/^D1950>&377)           ;  WAIT LOOP BELOW
002040  1 072031 3 5 0 01 11           MOVB    AC1                             ;  TAKES 1950NS
002041  1 022000 1 1 0 00 00     RNCNT:  DATI    MPSCR0,BR                       ;READ STATUS REGISTER
002042  1 104114 4 2 0114               JMPB0   NRTN1                           ;RETURN NOW IF SET
002043  1 072023 3 5 0 01 03           INCR    AC1                             ;INCREMENT THE WAIT COUNT
002044  1 072004 3 5 0 00 04           ADCR    AC0                             ; ALL OF IT
002045  1 112047 4 5 0047               JMPC    NORUN                           ;JUMP IF TIMED OUT
002046  1 100041 4 0 0041               JMP     RNCNT                           ;NO, KEEP COUNTING

002047  1 002334 0 1 0 334      NORUN:  GOSUB   LOGREG                          ;LOG THE REGISTERS
002050  1 177611 7 7 3 10 11
002051  1 002066 0 1 0 066            LDBR    RE.NRN                          ;GET ERROR CODE
002052  1 064051 3 2 0 02 11           MOVB    MPECR                           ;LOAD IT
002053  1 002000 0 1 0 000            LDBR    0                               ;CLEAR STATUS BYTE REGISTER
002054  1 064231 3 2 0 11 11           MOVB    STBYTE
002055  1 002160 0 1 0 160            LDBR    MPERR+ILF+OCC                   ;SET ILF TO SET EXC WITH CLASS B ERROR
002056  1 064031 3 2 0 01 11           MOVB    MPSCR1
002057  1 002266 0 1 0 266            JUMP    RPTERR                          ;REPORT THE ERROR
002060  1 161611 7 0 3 10 11
```

```
                                SUBTTL  INTERRUPT HANDLER

                                ;AN INTERRUPT HAS BEEN DETECTED
                                ;FIND THE INTERRUPTING CONDITION AND RESPOND AS FOLLOWS:
                                ; IF A DATA PARITY ERROR - RETURN
                                ; IF NOT - ABORT OPERATION AND REPORT ERROR

002061  1 132017 5 5 0 00 17   INTRPT: DATI    IOSEL,AC0          ;SAVE I/O SELECT REGISTER
002062  1 122016 5 1 0 00 16           DATI    MPSTAT,BR          ;READ INTERRUPT STATUS REGISTER
002063  1 104075 4 2 0075              JMPB0   INTON0             ;JUMP IF INTERRUPT ON INTERFACE 0
002064  1 014000 0 6 0 000             SHR                        ;SHIFT INT1 TO BR0
002065  1 104102 4 2 0102              JMPB0   INTON1             ;JUMP IF INTERRUPT ON INTERFACE 1
002066  1 014000 0 6 0 000             SHR
002067  1 104115 4 2 0115              JMPB0   INTON2             ;JUMP IF INTERRUPT ON INTERFACE 2
002070  1 014000 0 6 0 000   IGN2:     SHR
002071  1 104131 4 2 0131              JMPB0   INTON3             ;JUMP IF INTERRUPT ON INTERFACE 3
002072  1 062010 3 1 0 00 10 IGN3:     MOV     AC0,BR             ;RESTORE I/O SELECT
002073  1 066371 3 3 0 17 11           MOVB    IOSEL              ; REGISTER
002074  1 016000 0 7 0 000             RETURN                     ;RETURN

002075  1 002011 0 1 0 011   INTON0: LDBR    MB                 ;SELECT MASSBUS INTERFACE
002076  1 066371 3 3 0 17 11           MOVB    IOSEL
002077  1 002110 0 1 0 110             LDBR    F.INT0             ;GET ERROR CODE, THERE IS NOT HARDWARE
002100  1 064051 3 2 0 02 11           MOVB    MPECR              ; THAT CAN SET INT0
002101  1 100106 4 0 0106              JMP     HALT               ;HALT THE MICRO-PROCESSOR

002102  1 002011 0 1 0 011   INTON1: LDBR    MB                 ;SELECT MASSBUS INTERFACE
002103  1 066371 3 3 0 17 11           MOVB    IOSEL
002104  1 002070 0 1 0 070             LDBR    F.FAIL             ;GET ERROR CODE
002105  1 064051 3 2 0 02 11           MOVB    MPECR              ;LOAD INTO REGISTER
002106  1 002100 0 1 0 100   HALT:    LDBR    SPRES              ;RESET THE STACK POINTER
002107  1 066371 3 3 0 17 11           MOVB    IOSEL
002110  1 002011 0 1 0 011             LDBR    MB                 ;RE-SELECT THE MASSBUS INTERFACE
002111  1 066371 3 3 0 17 11           MOVB    IOSEL
002112  1 002102 0 1 0 102             LDBR    MPERR+CLRGO        ;SET ERROR FLAG
002113  1 064031 3 2 0 01 11           MOVB    MPSCR1
002114  1 016000 0 7 0 000   NRTN1:    RETURN                     ;HALT THE MICRO-PROCESSOR WITH A STACK UNDERFLOW

```

```
                                        ;INTERRUPT IS PRESENT ON INTERFACE 2

002115  1 002022 0 1 0 022     INTON2: LDBR    DP                      ;SELECT THE DATA PATH
002116  1 066371 3 3 0 17 11           MOVB    IOSEL
002117  1 022000 1 1 0 00 00           DATI    REG0,BR                 ;READ ERROR REGISTER
002120  1 104125 4 2 0125              JMPB0   INT2PE                  ;JUMP IF MICRO-BUS PARITY ERROR
002121  1 122016 5 1 0 00 16           DATI    MPSTAT,BR               ;READ INTERRUPT STATUS AGAIN
002122  1 014000 0 6 0 000             SHR                             ;INTERRUPT MUST BE DATA PARITY ERROR
002123  1 014000 0 6 0 000             SHR                             ;CONTINUE LOOKING FOR FATAL ERRORS
002124  1 100070 4 0 0070              JMP     IGN2

                                        ;INTERRUPT WAS A DATA PATH MICRO-BUS PARITY ERROR

002125  1 002007 0 1 0 007     INT2PE: LDBR    NRE.DP                  ;GET ERROR CODE
002126  1 072011 3 5 0 00 11           MOVB    AC0                     ;SAVE IN AC0
002127  1 002351 0 1 0 351             JUMP    RESET1                  ;RESET THE DEVICE FIRST
002130  1 161211 7 0 2 10 11
```

```
1976                                    ;INTERRUPT IS PRESENT ON INTERFACE 3
1977
1978  002131  1 002033 0 1 0 033  INTON3: LDBR    CHN                 ;SELECT CHANNEL INTERFACE
1979  002132  1 066371 3 3 0 17 11        MOVB    IOSEL
1980  002133  1 022000 1 1 0 00 00        DATI    CSR0,BR             ;READ STATUS REGISTER
1981  002134  1 106145 4 3 0145           JMPB4   INT3PE              ;JUMP IF MICRO-BUS PARITY ERROR
1982  002135  1 014000 0 6 0 000         SHR                         ;SHIFT TIME-OUT FLAG TO BR0
1983  002136  1 014000 0 6 0 000         SHR
1984  002137  1 106151 4 3 0151           JMPB4   TIMEO               ;JUMP IF TIME-OUT INTERRUPT
1985  002140  1 022005 1 1 0 00 05        DATI    TAGIN1,BR           ;READ TAG IN LINES
1986  002141  1 014000 0 6 0 000         SHR                         ;SHIFT DISCONNECT IN LINE
1987  002142  1 014000 0 6 0 000         SHR                         ; TO BR4
1988  002143  1 106155 4 3 0155           JMPB4   DISCON              ;JUMP IF SET
1989  002144  1 100072 4 0 0072           JMP     IGN3                ;RETURN IF DATA PARITY ERROR
1990
1991                                    ;INTERRUPT IS MICRO-BUS PARITY ERROR
1992
1993  002145  1 002027 0 1 0 027  INT3PE: LDBR    NRE.CP              ;GET ERROR CODE
1994  002146  1 072011 3 5 0 00 11        MOVB    AC0                 ;SAVE IT IN AC0
1995  002147  1 002351 0 1 0 351           JUMP    RESET1              ;RESET THE DEVICE FIRST
1996  002150  1 161211 7 0 2 10 11
1997
1998                                    ;INTERRUPT MAY BE A HANDSHAKE TIME-OUT
1999
2000  002151  1 002047 0 1 0 047  TIMEO:  LDBR    NRE.TO              ;GET ERROR CODE
2001  002152  1 072011 3 5 0 00 11        MOVB    AC0                 ;SAVE IN AC0
2002  002153  1 002351 0 1 0 351           JUMP    RESET1              ;RESET THE DEVICE FIRST
2003  002154  1 161211 7 0 2 10 11
2004
2005                                    ;DEVICE ASSERTED DISCONNECT IN
2006
2007  002155  1 002067 0 1 0 067  DISCON: LDBR    NRE.DC              ;GET ERROR CODE
2008  002156  1 072011 3 5 0 00 11        MOVB    AC0                 ;SAVE IT IN AC0
2009  002157  1 002351 0 1 0 351           JUMP    RESET1              ;RESET THE DEVICE
2010  002160  1 161211 7 0 2 10 11
2011
```

```
                                        SUBTTL  COMPLETE CLEAN-UP ROUTINE

                                        ;CHECK THAT THIS CODE IS LOADED IN THE CORRECT HARDWARE

002161  1 002300 0 1 0 300      CLENUP: LDBR    INIT+SPRES              ;INITIALIZE THE DX20
002162  1 066371 3 3 0 17 11          MOVB    IOSEL                   ;  AND THE STACK POINTER
002163  1 002011 0 1 0 011             LDBR    MB                      ;SELECT THE MASSBUS INTERFACE
002164  1 066371 3 3 0 17 11          MOVB    IOSEL                   ;  IN I/O SELECT REGISTER
002165  1 001004 0 0 2 004             LDMAR   DRVTYP&377              ;SELECT DRIVE TYPE CODE
002166  1 000400 0 0 1 000             LDMARX  DRVTYP_-8               ;REQUIRED BY THIS MICRO-CODE
002167  1 032003 1 5 0 00 03           DATI    MPDTR,AC0               ;READ DRIVE TYPE REGISTER
002170  1 040017 2 0 0 00 17           OSM     AC0                     ;COMPARE WITH REQUIRED TYPE
002171  1 114173 4 6 0173              JMPZ    .+2                     ;SKIP IF A MATCH
002172  1 101677 4 0 1677              JMP     WRONGD                  ;NO, WRONG DRIVE TYPE
002173  1 033404 1 5 3 00 04           DATI    MPHVR,AC0,I             ;READ HARDWARE VERSION REGISTER
002174  1 040017 2 0 0 00 17           OSM     AC0                     ;COMPARE WITH REQUIRED VERSION
002175  1 114177 4 6 0177              JMPZ    .+2                     ;SKIP IF A MATCH
002176  1 101677 4 0 1677              JMP     WRONGD                  ;NO, WRONG HARDWARE VERSION
```

```
                                ;RESET THE MASSBUS INTERFACE

002177  1 002000 0 1 0 000          LDBR    0                   ;GET ZERO DATA
002200  1 064231 3 2 0 11 11        MOVB    STBYTE              ;CLEAR STATUS BYTE
002201  1 064211 3 2 0 10 11        MOVB    STINDX              ;CLEAR STATUS INDEX
002202  1 064251 3 2 0 12 11        MOVB    DR                  ;CLEAR DRIVE NUMBER
002203  1 064271 3 2 0 13 11        MOVB    MODE                ;CLEAR DRIVE MODE COMMAND
002204  1 064331 3 2 0 15 11        MOVB    SPARE               ;CLEAR SPARE WORD
002205  1 064371 3 2 0 17 11        MOVB    FLAGS               ;CLEAR FLAG REGISTER
002206  1 064351 3 2 0 16 11        MOVB    TIEBYT              ;CLEAR TRACK-IN-ERROR BYTE
002207  1 066031 3 3 0 01 11        MOVB    ASYCST              ;CLEAR ASYNC STATUS
002210  1 066011 3 3 0 00 11        MOVB    ASYCDR              ;CLEAR ASYNC DRIVE NUMBER
002211  1 066071 3 3 0 03 11        MOVB    CNTH                ;CLEAR BYTE COUNTER
002212  1 066051 3 3 0 02 11        MOVB    CNTL                ;  LOW ORDER BITS, TOO
002213  1 066131 3 3 0 05 11        MOVB    XSTAT0              ;CLEAR EXTENDED STATUS REGISTERS
002214  1 066111 3 3 0 04 11        MOVB    XSTAT1
002215  1 066171 3 3 0 07 11        MOVB    XSTAT2
002216  1 066151 3 3 0 06 11        MOVB    XSTAT3
002217  1 002120 0 1 0 120          LDBR    EXSIZE              ;SET UP SIZE OF EXTENDED STATUS TABLE
002220  1 064311 3 2 0 14 11        MOVB    DUMPSZ              ;LOAD REGISTER
002221  1 002002 0 1 0 002          LDBR    CLRGO               ;CLEAR GO, ATTENTION AND ERROR
002222  1 064031 3 2 0 01 11        MOVB    MPSCR1

                                ;RUN THE DIAGNOSTICS

002223  1 001002 0 0 2 002          LDMAR   TSTADR              ;SET MAR TO DISPATCH ADDR ADDRESS
002224  1 010000 0 4 0 000  RERUN:  LDMEM   ONCDGT&377          ;SET DISPATCH TABLE ADDR
002225  1 002000 0 1 0 000  DLOOP:  LDBR    0                   ;CLEAR ERROR NUMBER REGISTER
002226  1 072171 3 5 0 07 11        MOVB    AC7
002227  1 042011 2 1 0 00 11        MOVMEM  BR                  ;PUT DISPATCH ADDR IN BR
002230  1 001000 0 0 2 000          LDMAR   0                   ;RESET MAR
002231  1 000400 0 0 1 000          LDMARX  0                   ;CLEAR MAR EXT BITS
002232  1 177631 7 7 3 11 11        JMPSUB  @BR,ONCDGT          ;GO RUN DIAG TEST
002233  1 001001 0 0 2 001          LDMAR   DFLAGS              ;SET ADDR OF FLAGS
002234  1 043411 2 1 3 00 11        MOVMEM  BR,I                ;GET FLAGS INTO BR
002235  1 014000 0 6 0 000          SHR                         ;RIGHT ADJUST LOOPING ON ERROR BIT
002236  1 104225 4 2 0225           JMPB0   DLOOP               ;REPEAT TEST IF LOOPING
002237  1 052011 2 5 0 00 11        MOVMEM  AC0                 ;GET DISPATCH ADDR
002240  1 070003 3 4 0 00 03        INC     AC0,MEM             ;INC IT AND RESTORE IT
002241  1 100225 4 0 0225           JMP     DLOOP
```

```
2072                                    ;TURN THE DX20 ON-LINE
2073
2074  002242  1 002033 0 1 0 033    CLNUPB: LDBR    CHN                     ;SELECT CHANNEL INTERFACE
2075  002243  1 066371 3 3 0 17 11           MOVB    IOSEL
2076  002244  1 002003 0 1 0 003            LDBR    ONLINE+CHANL            ;GET ON-LINE BIT
2077  002245  1 064031 3 2 0 01 11           MOVB    CSR1                    ;ENABLE CHANNEL BUS DRIVERS
2078  002246  1 002200 0 1 0 200            LDBR    OPLOUT                  ;GET OPERATIONAL OUT
2079  002247  1 064071 3 2 0 03 11           MOVB    TOR1                    ;BRING UP THE SUB-SYSTEM
2080  002250  1 002040 0 1 0 040            LDBR    MTROUT                  ;SET METER OUT
2081  002251  1 064051 3 2 0 02 11           MOVB    TOR0
2082
2083                                    ;TURN ON THE DATA PATH FORMATTER CLOCK
2084
2085  002252  1 002022 0 1 0 022            LDBR    DP                      ;SELECT THE DATA PATH
2086  002253  1 066371 3 3 0 17 11           MOVB    IOSEL
2087  002254  1 002002 0 1 0 002            LDBR    BCLKEN                  ;ENABLE THE CLOCK
2088  002255  1 064031 3 2 0 01 11           MOVB    REG1
2089
2090                                    ;CLEAR EXTENDED STATUS TABLE
2091
2092  002256  1 002312 0 1 0 312            GOSUB   CLRSNS                  ;CLEAR THE SENSE BYTE AREA
2093  002257  1 177611 7 7 3 10 11
2094  002260  1 001104 0 0 2 104            LDMAR   STAT                    ;SELECT MEMORY ADDRESS
2095  002261  1 002065 0 1 0 065            LDBR    INTDRV-STAT             ;GET COUNT OF LOCATIONS
2096  002262  1 072011 3 5 0 00 11           MOVB    AC0                     ; TO CLEAR
2097  002263  1 011400 0 4 3 000            LDMEM   0,I                     ;CLEAR EACH LOCATION
2098  002264  1 072007 3 5 0 00 07           DECR    AC0
2099  002265  1 112263 4 5 0263             JMPC    .-2
2100
2101  002266  1 102061 4 1 0061             JMPI    INTRPT                  ;CHECK ON INTERRUPTS
2102
2103  002267  1 002026 0 1 0 026            JUMP    IDLE
2104  002270  1 160211 7 0 0 10 11
2105
```

```
                                        ;*TEST THAT JMPZ CAN JUMP CONDITIONALLY ON THE Z BIT

002271  1 000000 0 0 0 000      DTST0:  NOP                             ;A NOP INSTRUCTION CLEARS Z
002272  1 115623 4 6 1623              JMPZ    DERR0                   ;SHOULD NOT JUMP
002273  1 002377 0 1 0 377              LDBR    -1                      ;SET Z
002274  1 114276 4 6 0276              JMPZ    .+2                     ;SKIP OVER HALT
002275  1 101622 4 0 1622              JMP     DERR1                   ;REPORT ERROR
002276  1 016000 0 7 0 000              RETURN

                                        ;*SELECT MEMORY ADDRESS 0.
                                        ;*CHECK ON OSB INSTRUCTION

002277  1 001000 0 0 2 000      DTST1:  LDMAR   0                       ;SELECT MEMORY LOCATION 0
002300  1 000400 0 0 1 000              LDMARX  0
002301  1 002000 0 1 0 000              LDBR    0                       ;PUT A ZERO IN BR
002302  1 072011 3 5 0 00 11            MOVB    AC0                     ;PUT A ZERO IN AC0
002303  1 070017 3 4 0 00 17            OSB     AC0,MEM                 ;COMPARE, PUT RESULT IN MEMORY
002304  1 114306 4 6 0306              JMPZ    .+2                     ;RESULT SHOULD HAVE BEEN -1
002305  1 101621 4 0 1621              JMP     DERR2                   ;REPORT ERROR
002306  1 002377 0 1 0 377              LDBR    -1                      ;PUT A -1 INTO BR
002307  1 070017 3 4 0 00 17            OSB     AC0,MEM                 ;COMPARE, PUT RESULT IN MEMORY
002310  1 115620 4 6 1620              JMPZ    DERR3                   ;RESULT SHOULD NOT HAVE BEEN -1
002311  1 072011 3 5 0 00 11            MOVB    AC0                     ;MOVE ONES FROM BR TO AC
002312  1 070017 3 4 0 00 17            OSB     AC0,MEM                 ;COMPARE, PUT RESULT IN MEMORY
002313  1 114315 4 6 0315              JMPZ    .+2                     ;RESULT SHOULD BE -1
002314  1 101617 4 0 1617              JMP     DERR4                   ;REPORT ERROR
002315  1 016000 0 7 0 000              RETURN

                                        ;*CHECK JMP ON BR BIT CONDITIONS

002316  1 002001 0 1 0 001      DTST2:  LDBR    1                       ;SET BR BIT 0
002317  1 104321 4 2 0321              JMPB0   .+2                     ;JMP ON BR BIT 0
002320  1 101616 4 0 1616              JMP     DERR5                   ;REPORT ERROR
002321  1 002376 0 1 0 376              LDBR    376                     ;SET ALL BUT BIT 0
002322  1 105615 4 2 1615              JMPB0   DERR6                   ;JMP ON BIT ZERO
002323  1 002020 0 1 0 020              LDBR    20                      ;SET BR BIT 4
002324  1 106326 4 3 0326              JMPB4   .+2                     ;JMP ON BIT 4
002325  1 101614 4 0 1614              JMP     DERR7                   ;REPORT ERROR
002326  1 002357 0 1 0 357              LDBR    357                     ;SET ALL BUT BIT 4
002327  1 107613 4 3 1613              JMPB4   DERR10                  ;JMP ON BIT 4
002330  1 002200 0 1 0 200              LDBR    200                     ;SET BR BIT 7
002331  1 110333 4 4 0333              JMPB7   .+2                     ;JMP ON BIT 7
002332  1 101612 4 0 1612              JMP     DERR11                  ;REPORT ERROR
002333  1 002177 0 1 0 177              LDBR    177                     ;SET ALL BUT BIT 7
002334  1 111611 4 4 1611              JMPB7   DERR12                  ;JMPON BIT 7
002335  1 016000 0 7 0 000              RETURN
```

```
                                        ;*CHECK OSM INSTRUCTION

002336  1 002377 0 1 0 377      DTST3:  LDBR    -1              ;SET ALL BITS IN BR
002337  1 072011 3 5 0 00 11            MOVB    AC0             ;PUT BITS IN AC0
002340  1 010377 0 4 0 377              LDMEM   -1              ;SET ALL BITS IN MEMORY LOC 0
002341  1 042017 2 1 0 00 17            OSM     AC0,BR          ;COMPARE MEM WITH AC0, PUT RESULT IN BR
002342  1 114344 4 6 0344              JMPZ    .+2             ;SHOULD JUMP
002343  1 101610 4 0 1610              JMP     DERR13          ;REPORT ERROR
002344  1 010000 0 4 0 000             LDMEM   0               ;LOAD ZEROS INTO MEMORY
002345  1 042017 2 1 0 00 17            OSM     AC0,BR          ;COMPARE MEM WITH AC0, PUT RESULT IN BR
002346  1 115607 4 6 1607              JMPZ    DERR14          ;SHOULD NOT JMP
002347  1 016000 0 7 0 000             RETURN
                                        ;*CHECK MOVMEM INSTRUCTION

002350  1 010377 0 4 0 377      DTST4:  LDMEM   -1              ;LOAD MEM WITH -1
002351  1 052011 2 5 0 00 11            MOVMEM  AC0             ;MOVE FROM MEM TO AC0
002352  1 042017 2 1 0 00 17            OSM     AC0,BR          ;COMPARE MEM AND AC0, PUT RESULT IN BR
002353  1 114355 4 6 0355              JMPZ    .+2             ;SKIP IF COMPARED
002354  1 101606 4 0 1606              JMP     DERR15          ;REPORT ERROR
002355  1 010022 0 4 0 022             LDMEM   22              ;LOAD MEM WITH 22
002356  1 052011 2 5 0 00 11            MOVMEM  AC0             ;MOVE FROM MEM TO AC0
002357  1 042017 2 1 0 00 17            OSM     AC0,BR          ;COMPARE MEM AND AC0, PUT RESULT IN BR
002360  1 114362 4 6 0362              JMPZ    .+2             ;SKIP IF COMPARED
002361  1 101605 4 0 1605              JMP     DERR16          ;REPORT ERROR
002362  1 016000 0 7 0 000             RETURN

                                        ;*CHECK INCREMENT FUNCTION
                                        ;INCREMENT 377 TO 0

002363  1 010377 0 4 0 377      DTST5:  LDMEM   377             ;LOAD A 377 INTO MEM
002364  1 052011 2 5 0 00 11            MOVMEM  AC0             ;MOVE TO AC0
002365  1 072003 3 5 0 00 03            INCR    AC0             ;INCREMENT AC0
002366  1 002000 0 1 0 000             LDBR    0               ;SET UP EXPECTED RESULT
002367  1 060017 3 0 0 00 17            OSB     AC0             ;COMPARE BR AND AC0
002370  1 114372 4 6 0372              JMPZ    .+2             ;SKIP IF COMPARED
002371  1 101604 4 0 1604              JMP     DERR17          ;REPORT ERROR
002372  1 016000 0 7 0 000             RETURN
```

```
2191                                    ;*CHECK DECREMENT FUNCTION
2192
2193                                    ;DECREMENT 0 TO -1
2194
2195  002373  1 002000 0 1 0 000    DTST6:  LDBR    0               ;LOAD A 0 INTO BR
2196  002374  1 072011 3 5 0 00 11          MOVB    AC0             ;MOVE TO AC0
2197  002375  1 072007 3 5 0 00 07          DECR    AC0             ;DECREMENT AC0
2198  002376  1 060017 3 0 0 00 17          OSB     AC0             ;COMPARE BR AND AC0
2199  002377  1 115603 4 6 1603             JMPZ    DERR20          ;DECR SHOULD HAVE CHANGED AC0
2200  002400  1 010377 0 4 0 377            LDMEM   -1              ;SET UP EXPECTED RESULT
2201  002401  1 040017 2 0 0 00 17          OSM     AC0             ;COMPARE MEM AND AC0
2202  002402  1 114404 4 6 0404             JMPZ    .+2             ;SKIP IF COMPARED
2203  002403  1 101602 4 0 1602             JMP     DERR21          ;REPORT ERROR
2204  002404  1 016000 0 7 0 000            RETURN
2205
2206                                    ;*CHECK CARRY FLAG
2207
2208  002405  1 002377 0 1 0 377    DTST7:  LDBR    -1              ;LOAD BR WITH -1
2209  002406  1 072011 3 5 0 00 11          MOVB    AC0             ;MOVE TO AC0
2210  002407  1 072003 3 5 0 00 03          INCR    AC0             ;INCREMENT AC0
2211  002410  1 112412 4 5 0412             JMPC    .+2             ;JMP IF CARRY SET
2212  002411  1 101601 4 0 1601             JMP     DERR22          ;REPORT ERROR
2213  002412  1 002000 0 1 0 000            LDBR    0               ;LOAD BR WITH ZEROS, SHOULD NOT CHANGE CARRY BIT
2214  002413  1 112415 4 5 0415             JMPC    .+2             ;JMP IF CARRY STILL SET
2215  002414  1 101600 4 0 1600             JMP     DERR23          ;REPORT ERROR
2216  002415  1 072003 3 5 0 00 03          INCR    AC0             ;INCREMENT AC0 TO A 1, SHOULD CLEAR CARRY BIT
2217  002416  1 113577 4 5 1577             JMPC    DERR24          ;JMP IF CARRY SET
2218  002417  1 002000 0 1 0 000            LDBR    0               ;LOAD BR WITH 0, SHOULD NOT CHANGE CARRY BIT
2219  002420  1 113576 4 5 1576             JMPC    DERR25          ;JMP IF CARRY SET
2220  002421  1 072007 3 5 0 00 07          DECR    AC0             ;DECREMENT AC0 TO A 0
2221  002422  1 112424 4 5 0424             JMPC    .+2             ;JMP IF CARRY IS SET
2222  002423  1 101575 4 0 1575             JMP     DERR26          ;REPORT ERROR
2223  002424  1 072007 3 5 0 00 07          DECR    AC0             ;DECREMENT AC0 TO -1
2224  002425  1 113574 4 5 1574             JMPC    DERR27          ;JMP IF CARRY SET
2225  002426  1 016000 0 7 0 000            RETURN
2226
```

```
                                ;*CHECK AC 0 THROUGH 6

                                ;PUT A 0 IN AC0

002427  1 002000 0 1 0 000      DTST8:  LDBR    0               ;CLEAR BR
002430  1 072011 3 5 0 00 11            MOVB    AC0             ;MOVE TO AC0

                                ;PUT A 1 IN AC1

002431  1 062003 3 1 0 00 03            INC     AC0,BR          ;INCREMENT TO 1, PUT IN BR
002432  1 072031 3 5 0 01 11            MOVB    AC1             ;PUT THE 1 IN AC1
002433  1 060037 3 0 0 01 17            OSB     AC1             ;COMPARE BR AND AC1
002434  1 114436 4 6 0436              JMPZ    .+2             ;SKIP IF A MATCH
002435  1 101573 4 0 1573              JMP     DERR30          ;REPORT ERROR

                                ;PUT A 2 IN AC2

002436  1 062023 3 1 0 01 03            INC     AC1,BR          ;INCREMENT TO 2, PUT IN BR
002437  1 072051 3 5 0 02 11            MOVB    AC2             ;PUT THE 2 IN AC2
002440  1 060057 3 0 0 02 17            OSB     AC2             ;COMPARE BR AND AC2
002441  1 114443 4 6 0443              JMPZ    .+2             ;SKIP IF A MATCH
002442  1 101572 4 0 1572              JMP     DERR31          ;REPORT ERROR

                                ;PUT A 3 IN AC3

002443  1 062043 3 1 0 02 03            INC     AC2,BR          ;INCREMENT TO 3, PUT IN BR
002444  1 072071 3 5 0 03 11            MOVB    AC3             ;PUT THE 3 IN AC3
002445  1 060077 3 0 0 03 17            OSB     AC3             ;COMPARE AC3 AND BR
002446  1 114450 4 6 0450              JMPZ    .+2             ;SKIP IF A MATCH
002447  1 101571 4 0 1571              JMP     DERR32          ;REPORT ERROR

                                ;PUT A 4 IN AC4

002450  1 062063 3 1 0 03 03            INC     AC3,BR          ;INCREMENT TO 4, PUT IN BR
002451  1 072111 3 5 0 04 11            MOVB    AC4             ;PUT THE 4 IN AC4
002452  1 060117 3 0 0 04 17            OSB     AC4             ;COMPARE AC4 AND BR
002453  1 114455 4 6 0455              JMPZ    .+2             ;SKIP IF A MACTH
002454  1 101570 4 0 1570              JMP     DERR33          ;REPORT ERROR

                                ;PUT A 5 IN AC5

002455  1 062103 3 1 0 04 03            INC     AC4,BR          ;INCREMENT TO 5, PUT IN BR
002456  1 072131 3 5 0 05 11            MOVB    AC5             ;PUT THE 5 IN AC5
002457  1 060137 3 0 0 05 17            OSB     AC5             ;COMPARE AC5 AND BR
002460  1 114462 4 6 0462              JMPZ    .+2             ;SKIP IF A MATCH
002461  1 101567 4 0 1567              JMP     DERR34          ;REPORT ERROR

                                ;PUT A 6 IN AC6

002462  1 062123 3 1 0 05 03            INC     AC5,BR          ;INCREMENT TO 6, PUT IN BR
002463  1 072151 3 5 0 06 11            MOVB    AC6             ;PUT THE 6 IN AC6
002464  1 060157 3 0 0 06 17            OSB     AC6             ;COMPARE AC6 AND BR
002465  1 114467 4 6 0467              JMPZ    .+2             ;SKIP IF A MATCH
002466  1 101566 4 0 1566              JMP     DERR35          ;REPORT ERROR
```

```
2282                                    ;CHECK AC6 FOR 6
2283
2284  002467  1 010006 0 4 0 006               LDMEM   6               ;PUT A 6 IN MEMORY
2285  002470  1 040157 2 0 0 06 17             OSM     AC6             ;COMPARE  WITH AC6
2286  002471  1 114473 4 6 0473                 JMPZ    .+2             ;SKIP IF A MATCH
2287  002472  1 101565 4 0 1565                 JMP     DERR36          ;REPORT ERROR
2288
2289                                    ;CHECK AC5 FOR 5
2290
2291  002473  1 070147 3 4 0 06 07             DEC     AC6,MEM         ;DECREMENT THE 6 TO 5, PUT IN MEM
2292  002474  1 040137 2 0 0 05 17             OSM     AC5             ;COMPARE WITH AC5
2293  002475  1 114477 4 6 0477                 JMPZ    .+2             ;SKIP IF A MATCH
2294  002476  1 101564 4 0 1564                 JMP     DERR37          ;REPORT ERROR
2295
2296                                    ;CHECK AC4 FOR 4
2297
2298  002477  1 070127 3 4 0 05 07             DEC     AC5,MEM         ;DECREMENT TO 4, PUT IN MEM
2299  002500  1 040117 2 0 0 04 17             OSM     AC4             ;COMPARE WITH AC4
2300  002501  1 114503 4 6 0503                 JMPZ    .+2             ;SKIP IF A MATCH
2301  002502  1 101563 4 0 1563                 JMP     DERR40          ;REPORT ERROR
2302
2303                                    ;CHECK AC3 FOR 3
2304
2305  002503  1 070107 3 4 0 04 07             DEC     AC4,MEM         ;DECREMENT TO 3, PUT IN MEM
2306  002504  1 040077 2 0 0 03 17             OSM     AC3             ;COMPARE WITH AC3
2307  002505  1 114507 4 6 0507                 JMPZ    .+2             ;SKIP IF A  MATCH
2308  002506  1 101562 4 0 1562                 JMP     DERR41          ;REPORT ERROR
2309
2310                                    ;CHECK AC2 FOR 2
2311
2312  002507  1 070067 3 4 0 03 07             DEC     AC3,MEM         ;DECREMENT TO A 2, PUT IN MEM
2313  002510  1 040057 2 0 0 02 17             OSM     AC2             ;COMPARE WITH AC2
2314  002511  1 114513 4 6 0513                 JMPZ    .+2             ;SKIP IF A MATCH
2315  002512  1 101561 4 0 1561                 JMP     DERR42          ;REPORT ERROR
2316
2317                                    ;CHECK AC1 FOR 1
2318
2319  002513  1 070047 3 4 0 02 07             DEC     AC2,MEM         ;DECREMENT TO A 1, PUT IN MEM
2320  002514  1 040037 2 0 0 01 17             OSM     AC1             ;COMPARE WITH AC1
2321  002515  1 114517 4 6 0517                 JMPZ    .+2             ;SKIP IF A MATCH
2322  002516  1 101560 4 0 1560                 JMP     DERR43          ;REPORT ERROR
2323
2324                                    ;CHECK AC0 FOR 0
2325
2326  002517  1 070027 3 4 0 01 07             DEC     AC1,MEM         ;DECREMENT TO A 0, PUT IN MEM
2327  002520  1 040017 2 0 0 00 17             OSM     AC0             ;COMPARE WITH AC0
2328  002521  1 114523 4 6 0523                 JMPZ    .+2             ;SKIP IF A MATCH
2329  002522  1 101557 4 0 1557                 JMP     DERR44          ;REPORT ERROR
2330  002523  1 016000 0 7 0 000               RETURN
2331
```

```
                                        ;*CHECK ADC INSTRUCTION

002524  1 002376 0 1 0 376      DTST9:  LDBR    -2              ;PUT A -2 INTO BR
002525  1 072071 3 5 0 03 11            MOVB    AC3             ;MOVE TO AC3
002526  1 072063 3 5 0 03 03            INCR    AC3             ;INCREMENT TO -1 TO CLEAR CARRY BIT
002527  1 062064 3 1 0 03 04            ADC     AC3,BR          ;ADD CARRY TO AC3, PUT RESULT IN BR
002530  1 114532 4 6 0532              JMPZ    .+2             ;SKIP IF RESULT WAS -1, CARRY WAS 0
002531  1 101556 4 0 1556              JMP     DERR45          ;REPORT ERROR
002532  1 072063 3 5 0 03 03            INCR    AC3             ;INCREMENT AC3 TO 0, SETS THE CARRY BIT
002533  1 072064 3 5 0 03 04            ADCR    AC3             ;ADD CARRY TO AC3
002534  1 002001 0 1 0 001              LDBR    1               ;SET UP BR WITH EXPECTED VALUE OF 1
002535  1 060077 3 0 0 03 17            OSB     AC3             ;COMPARE BR AND AC3
002536  1 114540 4 6 0540              JMPZ    .+2             ;SKIP IF A MATCH
002537  1 101555 4 0 1555              JMP     DERR46          ;REPORT ERROR
002540  1 016000 0 7 0 000             RETURN

                                        ;*CHECK SHL INSTRUCTION

002541  1 002252 0 1 0 252      DTST10: LDBR    252             ;LOAD 252 INTO AC5
002542  1 072131 3 5 0 05 11            MOVB    AC5
002543  1 072125 3 5 0 05 05            SHLR    AC5             ;SHIFT IT LEFT
002544  1 010124 0 4 0 124              LDMEM   124             ;SET EXPECTED DATA
002545  1 040137 2 0 0 05 17            OSM     AC5             ;CHECK FOR EXPECTED
002546  1 114550 4 6 0550              JMPZ    .+2             ;JUMP IF SHIFTED OKAY
002547  1 101554 4 0 1554              JMP     DERR47          ;REPORT ERROR
002550  1 010125 0 4 0 125              LDMEM   125             ;LOAD 125 INTO AC5
002551  1 052131 2 5 0 05 11            MOVMEM  AC5
002552  1 072125 3 5 0 05 05            SHLR    AC5             ;SHIFT IT LEFT
002553  1 060137 3 0 0 05 17            OSB     AC5             ;CHECK FOR CORRECT DATA
002554  1 114556 4 6 0556              JMPZ    .+2             ;JUMP IF SHIFTED OKAY
002555  1 101553 4 0 1553              JMP     DERR50          ;REPORT ERROR
002556  1 016000 0 7 0 000             RETURN
```

```
2365                                    ;*CHECK ADB INSTRUCTION
2366
2367  002557  1 010333 0 4 0 333    DTST11: LDMEM   333             ;SET DATA INTO AC5 FOR ADD
2368  002560  1 052131 2 5 0 05 11         MOVMEM  AC5
2369  002561  1 002111 0 1 0 111           LDBR    111             ;SET OTHER DATA TO ADD
2370  002562  1 072120 3 5 0 05 00         ADBR    AC5             ;ADD
2371  002563  1 112565 4 5 0565            JMPC    .+2             ;CARRY SHOULD HAVE SET
2372  002564  1 101552 4 0 1552            JMP     DERR51          ;REPORT ERROR
2373  002565  1 010044 0 4 0 044           LDMEM   44              ;SET EXPECTED SUM
2374  002566  1 040137 2 0 0 05 17         OSM     AC5             ;CHECK IT
2375  002567  1 114571 4 6 0571            JMPZ    .+2             ;JUMP IF OKAY
2376  002570  1 101551 4 0 1551            JMP     DERR52          ;REPORT ERROR
2377  002571  1 016000 0 7 0 000           RETURN
2378
2379                                    ;*CHECK ADBC INSTRUCTION
2380
2381  002572  1 002377 0 1 0 377    DTST12: LDBR    -1              ;LOAD BR WITH A -1
2382  002573  1 072051 3 5 0 02 11         MOVB    AC2             ;MOVE TO AC2
2383  002574  1 072043 3 5 0 02 03         INCR    AC2             ;INCREMENT AC2 TO 0 TO SET CARRY
2384  002575  1 002333 0 1 0 333           LDBR    333             ;LOAD BR WITH A 333
2385  002576  1 072041 3 5 0 02 01         ADBCR   AC2             ;ADD BR AND CARRY TO AC2
2386  002577  1 002334 0 1 0 334           LDBR    334             ;SET UP BR WITH EXPECTED VALUE OF 334
2387  002600  1 060057 3 0 0 02 17         OSB     AC2             ;COMPARE AC2 AND BR
2388  002601  1 114603 4 6 0603            JMPZ    .+2             ;SKIP IF A MATCH
2389  002602  1 101550 4 0 1550            JMP     DERR53          ;REPORT ERROR
2390  002603  1 016000 0 7 0 000           RETURN
2391
2392                                    ;*TEST SHR INSTRUCTION
2393
2394                                    ;SHIFT A 252 RIGHT, CHECK FOR RESULT OF 125
2395
2396  002604  1 002252 0 1 0 252    DTST13: LDBR    252             ;LOAD BR WITH A 252
2397  002605  1 014000 0 6 0 000           SHR                     ;SHIFT BR RIGHT ONE PLACE
2398  002606  1 010125 0 4 0 125           LDMEM   125             ;LOAD MEM WITH EXPECTED RESULT
2399  002607  1 052151 2 5 0 06 11         MOVMEM  AC6             ;MOVE TO AC6
2400  002610  1 060157 3 0 0 06 17         OSB     AC6             ;COMPARE RESULT IN BR WITH AC6
2401  002611  1 114613 4 6 0613            JMPZ    .+2             ;SKIP IF A MATCH
2402  002612  1 101547 4 0 1547            JMP     DERR54          ;REPORT ERROR
2403
2404                                    ;SHIFT A 125 RIGHT, CHECK FOR RESULT OF 52
2405
2406  002613  1 014000 0 6 0 000           SHR                     ;SHIFT BR RIGHT ONE PLACE
2407  002614  1 010052 0 4 0 052           LDMEM   52              ;LOAD MEM WITH EXPECTED RESULT
2408  002615  1 052151 2 5 0 06 11         MOVMEM  AC6             ;MOVE TO AC6
2409  002616  1 060157 3 0 0 06 17         OSB     AC6             ;COMPARE RESULT IN BR WITH AC6
2410  002617  1 114621 4 6 0621            JMPZ    .+2             ;SKIP IF A MATCH
2411  002620  1 101546 4 0 1546            JMP     DERR55          ;REPORT ERROR
2412  002621  1 016000 0 7 0 000           RETURN
2413
```

```
                                ;*CHECK LOGICAL INSTRUCTIONS
                                ;[10] [11] [12] THIS TEST WAS DELETED TO MAKE ROOM FOR THESE EDITS

                                ;AND 252 WITH 146, CHECK FOR RESULT OF 42

002622  1 016000 0 7 0 000      DTST14: RETURN                          ;[10] [11] [12]
                                COMMENT %
                                DTST14: LDBR    252                     ;LOAD BR WITH A 252
                                        MOVB    AC2                     ;MOVE TO AC2
                                        LDBR    146                     ;LOAD BR WITH A 146
                                        LANDB   AC2,MEM                 ;AND AC2 WITH BR, PUT RESULT IN MEM
                                        MOVMEM  AC3                     ;MOVE RESULT TO AC3
                                        LDMEM   42                      ;SET UP EXPECTED RESULT IN MEM
                                        OSM     AC3                     ;COMPARE RESULT
                                        JMPZ    .+2                     ;SKIP IF A MATCH
                                        JMP     DERR56                  ;REPORT ERROR

                                ;OR 252 WITH 146, CHECK FOR RESULT OF 356

                                        LORB    AC2,MEM                 ;OR AC2 WITH BR, PUT RESULT IN MEM
                                        MOVMEM  AC3                     ;MOVE RESULT TO AC3
                                        LDMEM   356                     ;SET UP EXPECTED RESULT IN MEM
                                        OSM     AC3                     ;COMPARE RESULT
                                        JMPZ    .+2                     ;SKIP IF A MATCH
                                        JMP     DERR57                  ;REPORT ERROR

                                ;OR 252 WITH NOT 146, CHECK FOR RESULT OF 273

                                        LORCB   AC2,MEM                 ;OR AC2 WITH COMP OF BR, PUT IN MEM
                                        MOVMEM  AC3                     ;MOVE RESULT TO AC3
                                        LDMEM   273                     ;SET UP EXPECTED RESULT IN MEM
                                        OSM     AC3                     ;COMPARE RESULT
                                        JMPZ    .+2                     ;SKIP IF A MATCH
                                        JMP     DERR60                  ;REPORT ERROR

                                ;XOR 252 WITH 146, CHECK FOR RESULT OF 314

                                        LXORB   AC2,MEM                 ;XOR AC2 WITH BR, PUT RESULT IN MEM
                                        MOVMEM  AC3                     ;MOVE RESULT TO AC3
                                        LDMEM   314                     ;SET UP EXPECTED RESULT IN MEM
                                        OSM     AC3                     ;COMPARE RESULT
                                        JMPZ    .+2                     ;SKIP IF A MATCH
                                        JMP     DERR61                  ;REPORT ERROR
                                        RETURN
                                %
```

```
2460                                    ;*CHECK JUMP TO SUBROUTINE AND RETURN INSTRUCTIONS
2461
2462  002623  1 002000 0 1 0 000    DTST15: LDBR    0                       ;LOAD BR WITH A ZERO
2463  002624  1 072011 3 5 0 00 11          MOVB    AC0                     ;MOVE TO AC0
2464  002625  1 116633 4 7 0633            JMPSUB  COMPAC                  ;GO TO SUBROUTINE WHICH SHOULD VERITY THAT AC0
2465                                                                        ;CONTAINS A ZERO, THEN CHANGES IT
2466                                                                        ;TO -1 BEFORE RETURNING
2467  002626  1 002377 0 1 0 377    SUBT1R: LDBR    -1                      ;SUBROUTINE SHOULD RETURN HERE
2468  002627  1 060017 3 0 0 00 17          OSB     AC0                     ;COMPARE AC0 WITH A -1 IN BR
2469  002630  1 114632 4 6 0632            JMPZ    .+2                     ;SKIP IF AC0 IS NOW A -1
2470  002631  1 101541 4 0 1541            JMP     DERR62                  ;REPORT ERROR
2471  002632  1 016000 0 7 0 000           RETURN
2472
2473  002633  1 002000 0 1 0 000    COMPAC: LDBR    0                       ;LOAD A ZERO INTO BR
2474  002634  1 060017 3 0 0 00 17          OSB     AC0                     ;COMPARE WITH AC0
2475  002635  1 114637 4 6 0637            JMPZ    .+2                     ;SKIP IF AC0 WAS ZERO
2476  002636  1 101540 4 0 1540            JMP     DERR63                  ;REPORT ERROR
2477  002637  1 002377 0 1 0 377           LDBR    -1                      ;GET A -1
2478  002640  1 072011 3 5 0 00 11          MOVB    AC0                     ;PUT IT INTO AC0
2479  002641  1 016000 0 7 0 000           RETURN                          ;RETURN TO SUBT1R
2480  002642  1 101537 4 0 1537            JMP     DERR64                  ;REPORT ERROR
2481
```

```
                                        ;*CHECK THAT STATUS REGISTER CAN BE READ

                                        ;CLEAR THE Z BIT, CHECK FOR Z BIT TO BE CLEAR IN STATUS REGISTER

002643  1 002000 0 1 0 000      DTST16: LDBR    0                       ;CLEAR THE BR
002644  1 122016 5 1 0 00 16           DATI    MPSTAT,BR               ;READ THE STATUS REGISTER
002645  1 014000 0 6 0 000             SHR                             ;SHIFT THE BR RIGHT
002646  1 107536 4 3 1536              JMPB4   DERR65                  ;JUMP IF Z BIT IS SET

                                        ;SET THE Z BIT, CHECK FOR Z BIT TO BE SET IN STATUS REGISTER

002647  1 002377 0 1 0 377             LDBR    -1                      ;LOAD BR WITH A -1 TO SET Z BIT
002650  1 122016 5 1 0 00 16           DATI    MPSTAT,BR               ;READ THE STATUS REGISTER
002651  1 014000 0 6 0 000             SHR                             ;SHIFT THE BR RIGHT
002652  1 106654 4 3 0654              JMPB4   .+2                     ;SKIP IF Z BIT IS SET
002653  1 101535 4 0 1535              JMP     DERR66                  ;REPORT ERROR

                                        ;CLEAR THE CARRY BIT, CHECK FOR CARRY TO BE CLEAR IN STATUS REGISTER

002654  1 002000 0 1 0 000             LDBR    0                       ;LOAD BR WITH 0
002655  1 072011 3 5 0 00 11           MOVB    AC0                     ;MOVE TO AC0
002656  1 060003 3 0 0 00 03           INC     AC0                     ;INCREMENT AC0 TO CLEAR THE C BIT
002657  1 122016 5 1 0 00 16           DATI    MPSTAT,BR               ;READ THE STATUS REGISTER
002660  1 107534 4 3 1534              JMPB4   DERR67                  ;JUMP IF C BIT IS SET

                                        ;SET THE CARRY BIT, CHECK FOR CARRY TO BE SET IN STATUS REGISTER

002661  1 002377 0 1 0 377             LDBR    -1                      ;LOAD BR WITH -1
002662  1 072011 3 5 0 00 11           MOVB    AC0                     ;MOVE TO AC0
002663  1 060003 3 0 0 00 03           INC     AC0                     ;INCREMENT AC0 TO SET C BIT
002664  1 122016 5 1 0 00 16           DATI    MPSTAT,BR               ;READ THE STATUS REGISTER
002665  1 106667 4 3 0667              JMPB4   .+2                     ;SKIP IF C BIT IS SET
002666  1 101533 4 0 1533              JMP     DERR70                  ;REPORT ERROR
002667  1 016000 0 7 0 000             RETURN

```

```
2517                                    ;*CHECK I/O SELECT REGISTER
2518
2519                                    ;WRITE 0 INTO I/O SELECT REGISTER, READ AND CHECK FOR 0
2520
2521  002670  1 002000 0 1 0 000    DTST17: LDBR    0                       ;LOAD BR WITH ZEROS
2522  002671  1 066371 3 3 0 17 11          MOVB    IOSEL                   ;WRITE INTO I/O SELECT REGISTER
2523  002672  1 132057 5 5 0 02 17          DATI    IOSEL,AC2               ;READ BACK INTO AC2
2524  002673  1 060057 3 0 0 02 17          OSB     AC2                     ;COMPARE WITH DATA WRITTEN
2525  002674  1 114676 4 6 0676            JMPZ    .+2                     ;SKIP IF A MATCH
2526  002675  1 101532 4 0 1532            JMP     DERR71                  ;REPORT ERROR
2527
2528                                    ;WRITE 77 INTO STATUS REGISTER, READ AND CHECK FOR 77
2529
2530  002676  1 002077 0 1 0 077          LDBR    77                      ;LOAD BR WITH 77
2531  002677  1 066371 3 3 0 17 11          MOVB    IOSEL                   ;WRITE INTO I/O SELECT REGISTER
2532  002700  1 132057 5 5 0 02 17          DATI    IOSEL,AC2               ;READ BACK INTO AC2
2533  002701  1 060057 3 0 0 02 17          OSB     AC2                     ;COMPARE WITH DATA WRITTEN
2534  002702  1 114704 4 6 0704            JMPZ    .+2                     ;SKIP IF A MATCH
2535  002703  1 101531 4 0 1531            JMP     DERR72                  ;REPORT ERROR
2536  002704  1 016000 0 7 0 000          RETURN
2537
2538                                    ;*CHECK DATI/O INSTRUCTION
2539
2540  002705  1 002022 0 1 0 022    DTST18: LDBR    DP                      ;SELECT DATA PATH
2541  002706  1 066371 3 3 0 17 11          MOVB    IOSEL                   ;FOR INPUT AND OUTPUT
2542  002707  1 002252 0 1 0 252          LDBR    252                     ;GET A DATA PATTERN
2543  002710  1 064131 3 2 0 05 11          MOVB    MCHO                    ;WRITE INTO MC CNTR REG
2544  002711  1 024105 1 2 0 04 05          DATI    MCHO,MCLO               ;COPY DATA INTO MCLO ON DATA PATH
2545  002712  1 032144 1 5 0 06 04          DATI    MCLO,AC6                ;READ MC DATA
2546  002713  1 060157 3 0 0 06 17          OSB     AC6                     ;COMPARE AC6 WITH DATA WRITTEN
2547  002714  1 114716 4 6 0716            JMPZ    .+2                     ;SKIP IF A MATCH
2548  002715  1 101530 4 0 1530            JMP     DERR73                  ;REPORT ERROR
2549  002716  1 016000 0 7 0 000          RETURN
2550
```

```
                                        ;*TEST THAT MICROPROCESSOR CAN TALK TO THE DATA PATH INTERFACE
                                        ;*WRITE DATA INTO MASSBUS COUNTER AND READ IT BACK
                                        
002717  1 002022 0 1 0 022      DTST19: LDBR    22                      ;SELECT DATA PATH
002720  1 066371 3 3 0 17 11          MOVB    IOSEL
002721  1 002125 0 1 0 125           LDBR    125                     ;SET DATA PATTERN
002722  1 064111 3 2 0 04 11          MOVB    MCLO                    ;WRITE IT TO DATA BUFFER
002723  1 032004 1 5 0 00 04          DATI    MCLO,AC0                ;READ IT BACK
002724  1 060017 3 0 0 00 17          OSB     AC0                     ;CHECK IT
002725  1 114727 4 6 0727             JMPZ    .+2                     ;JUMP IF OKAY
002726  1 101527 4 0 1527             JMP     DERR74                  ;ELSE, REPORT ERROR
002727  1 016000 0 7 0 000            RETURN
                                        
                                        ;*TEST THAT MICROPROCESSOR CAN TALK TO THE CHANNEL BUS INTERFACE
                                        ;*WRITE DATA INTO SPAD COUNTER AND READ IT BACK
                                        
002730  1 002033 0 1 0 033      DTST20: LDBR    33                      ;SELECT CHANNEL BUS INTERFACE
002731  1 066371 3 3 0 17 11          MOVB    IOSEL
002732  1 030001 1 4 0 00 01          DATI    CSR1,MEM                ;SAVE CSR1 CONTENTS
002733  1 052011 2 5 0 00 11          MOVMEM  AC0                     ;PUT THEM IN AC0
002734  1 002001 0 1 0 001            LDBR    CHANL                   ;SET CHANNEL MODE
002735  1 062014 3 1 0 00 14          LORB    AC0,BR
002736  1 064031 3 2 0 01 11          MOVB    CSR1                    ;SET CHANNEL MODE IN CSR1
002737  1 002017 0 1 0 017            LDBR    17                      ;SET DATA PATTERN
002740  1 064131 3 2 0 05 11          MOVB    TAGIN1                  ;WRITE IT TO DATA BUFFER
002741  1 032005 1 5 0 00 05          DATI    TAGIN1,AC0              ;READ IT BACK
002742  1 044031 2 2 0 01 11          MOVMEM  CSR1                    ;RESTORE CSR1 ORIGINAL CONTENTS
002743  1 072013 3 5 0 00 13          LANDBR  AC0                     ;ISOLATE SP BITS
002744  1 060017 3 0 0 00 17          OSB     AC0                     ;CHECK IT
002745  1 114747 4 6 0747             JMPZ    .+2                     ;JUMP IF OKAY
002746  1 101526 4 0 1526             JMP     DERR75                  ;ELSE, REPORT ERROR
002747  1 016000 0 7 0 000            RETURN
```

```
                                ;*TEST THAT THE CONTROL UNIT RESPONDS CORRECTLY TO THE "TEST I/O COMMAND" SEQUENCE.
                                ;CLEAR ALL TAG OUT LINES.
                                ;SET "OPL OUT".
                                ;PUT DEVICE ADDRESS 0 ON THE BUS OUT LINES.
                                ;SET "ADR OUT", "HLD OUT", THEN "SEL OUT".
                                ;CHECK THAT "OPL IN" SETS.

002750  1 001001 0 0 2 001      DTST21: LDMAR   DFLAGS                  ;[4] GET ADDRESS OF FLAGS
002751  1 052011 2 5 0 00 11           MOVMEM  AC0                     ;[4] GET FLAGS, ALL ZERO MEANS MICROCODE
                                                                        ;[4]  STARTED AT 'STARTA'
002752  1 001000 0 0 2 000             LDMAR   0                       ;[4] RESTORE MAR TO 0
002753  1 060007 3 0 0 00 07           DEC     AC0                     ;[4] DEC TO TEST FOR ZERO
002754  1 115100 4 6 1100              JMPZ    NOT21                   ;[4] SKIP TEST IF FLAGS ARE ZERO
002755  1 002003 0 1 0 003             LDBR    ONLINE+CHANL            ;SET ONLINE AND CHANNEL MODE
002756  1 064031 3 2 0 01 11           MOVB    CSR1
002757  1 117664 4 7 1664              JMPSUB  WAIT                    ;WAIT FOR BYPASS RELAY TO SETTLE
002760  1 002200 0 1 0 200             LDBR    OPLOUT                  ;SET "OPL OUT"
002761  1 064071 3 2 0 03 11           MOVB    TOR1
002762  1 002000 0 1 0 000             LDBR    0                       ;PUT DEV ADDR 0 ON BUS OUT
002763  1 064231 3 2 0 11 11           MOVB    BORLO
002764  1 002020 0 1 0 020             LDBR    ADROUT                  ;SET "ADR OUT"
002765  1 064051 3 2 0 02 11           MOVB    TOR0
002766  1 002030 0 1 0 030             LDBR    HLDOUT+ADROUT           ;SET "HLD OUT"
002767  1 064051 3 2 0 02 11           MOVB    TOR0
002770  1 002032 0 1 0 032             LDBR    SELOUT+HLDOUT+ADROUT    ;SET "SEL OUT"
002771  1 064051 3 2 0 02 11           MOVB    TOR0
002772  1 002200 0 1 0 200             LDBR    200                     ;SET WAIT CNT
002773  1 072031 3 5 0 01 11           MOVB    AC1
002774  1 022004 1 1 0 00 04    TIOLP1: DATI    TAGIN0,BR               ;READ TAG IN BITS
002775  1 111001 4 4 1001              JMPB7   .+4                     ;JUMP IF "OPL IN" SET
002776  1 072027 3 5 0 01 07           DECR    AC1                     ;DEC WAIT CNT
002777  1 115525 4 6 1525              JMPZ    DERR76                  ;JUMP IF TIMED OUT
003000  1 100774 4 0 0774              JMP     TIOLP1                  ;ELSE, KEEP WAITING

                                ;CLEAR "ADR OUT".
                                ;CHECK THAT "ADR IN" SETS.

003001  1 002012 0 1 0 012             LDBR    SELOUT+HLDOUT           ;CLEAR "ADR OUT"
003002  1 064051 3 2 0 02 11           MOVB    TOR0
003003  1 002200 0 1 0 200             LDBR    200                     ;SET WAIT CNT
003004  1 072031 3 5 0 01 11           MOVB    AC1
003005  1 022004 1 1 0 00 04    TIOLP2: DATI    TAGIN0,BR               ;READ TAG IN BITS
003006  1 107012 4 3 1012              JMPB4   .+4                     ;JUMP IF SET
003007  1 072027 3 5 0 01 07           DECR    AC1
003010  1 115524 4 6 1524              JMPZ    DERR77
003011  1 101005 4 0 1005              JMP     TIOLP2

                                ;CHECK THAT BUS IN LINES CONTAIN ADDRESS REQUESTED(0).

003012  1 032007 1 5 0 00 07           DATI    CBILO,AC0               ;READ BUS IN LINES
003013  1 010000 0 4 0 000             LDMEM   0                       ;SET EXPECTED DATA
003014  1 040017 2 0 0 00 17           OSM     AC0                     ;COMPARE
003015  1 115017 4 6 1017              JMPZ    .+2                     ;JUMP IF BUS IN LINES=0
003016  1 101523 4 0 1523              JMP     DER100                  ;REPORT ERROR
```

```
2639
2640                                    ;CHECK THAT "BUS0 PE FLAG" IS NOT SET.
2641  003017  1 022000 1 1 0 00 00             DATI    CSR0,BR                 ;READ REG0
2642  003020  1 014000 0 6 0 000               SHR                             ;RIGHT ADJ "BUS0 SE FLAG"
2643  003021  1 105522 4 2 1522                JMPB0   DER101                  ;JUMP IF PARITY ERROR
2644
```

```
2645                                    ;PUT TEST I-O COMMAND (0) ON BUS OUT LINES.
2646                                    ;SET "CMD OUT".
2647                                    ;CHECK THAT "ADR IN" CLEARS.
2648
2649  003022  1 002000 0 1 0 000               LDBR    0                       ;PUT TEST I/O CMD ON BUS LINES
2650  003023  1 064231 3 2 0 11 11            MOVB    BORLO
2651  003024  1 002013 0 1 0 013              LDBR    CMDOUT+HLDOUT+SELOUT    ;SET "CMD OUT"
2652  003025  1 064051 3 2 0 02 11            MOVB    TOR0
2653  003026  1 002200 0 1 0 200              LDBR    200                     ;SET WAIT CNT
2654  003027  1 072031 3 5 0 01 11            MOVB    AC1
2655  003030  1 022004 1 1 0 00 04  TIOLP3:   DATI    TAGIN0,BR               ;READ TAG IN LINES
2656  003031  1 107033 4 3 1033               JMPB4   .+2                     ;JUMP IF "ADR IN" STILL SET
2657  003032  1 101036 4 0 1036               JMP     .+4                     ;ELSE, IT DROPPED
2658  003033  1 072027 3 5 0 01 07            DECR    AC1                     ;DEC WAIT CNT
2659  003034  1 115521 4 6 1521               JMPZ    DER102                  ;JUMP IF TIMED OUT
2660  003035  1 101030 4 0 1030               JMP     TIOLP3                  ;ELSE, KEEP WAITING
2661
2662                                    ;CLEAR "CMD OUT".
2663                                    ;CHECK THAT "STA IN" SETS.
2664
2665  003036  1 002012 0 1 0 012               LDBR    HLDOUT+SELOUT           ;CLEAR "CMD OUT"
2666  003037  1 064051 3 2 0 02 11            MOVB    TOR0
2667  003040  1 002200 0 1 0 200              LDBR    200                     ;SET WAIT CNT.
2668  003041  1 072031 3 5 0 01 11            MOVB    AC1
2669  003042  1 022004 1 1 0 00 04  TIOLP4:   DATI    TAGIN0,BR               ;READ TAG IN LINES
2670  003043  1 105047 4 2 1047               JMPB0   .+4                     ;JUMP IF "STA IN" SET
2671  003044  1 072027 3 5 0 01 07            DECR    AC1                     ;DEC WAIT CNT
2672  003045  1 115520 4 6 1520               JMPZ    DER103                  ;JUMP IF TIMED OUT
2673  003046  1 101042 4 0 1042               JMP     TIOLP4                  ;ELSE, KEEP WAITING
2674
```

```
                                        ;SET "SRV OUT".
                                        ;CHECK THAT "STA IN" CLEARS.

003047  1 002212 0 1 0 212              LDBR    SRVOUT+HLDOUT+SELOUT    ;SET "SRV OUT"
003050  1 064051 3 2 0 02 11            MOVB    TOR0
003051  1 002200 0 1 0 200              LDBR    200                     ;SET WAIT CNT
003052  1 072031 3 5 0 01 11            MOVB    AC1
003053  1 022004 1 1 0 00 04  TIOLP5:    DATI    TAGIN0,BR               ;READ TAG IN LINES
003054  1 105056 4 2 1056               JMPB0   .+2                     ;JUMP IF STILL SET
003055  1 101061 4 0 1061               JMP     .+4                     ;ELSE, CLEARED OKAY
003056  1 072027 3 5 0 01 07            DECR    AC1                     ;DEC WAIT CNT
003057  1 115517 4 6 1517               JMPZ    DER104                  ;JUMP IF TIMED OUT
003060  1 101053 4 0 1053               JMP     TIOLP5                  ;ELSE, KEEP

                                        ;CLEAR "SRV OUT", "HLD OUT", THEN "SEL OUT".
                                        ; CHECK THAT "OPL IN" CLEARS.

003061  1 002012 0 1 0 012              LDBR    HLDOUT+SELOUT           ;CLEAR "SRV OUT"
003062  1 064051 3 2 0 02 11            MOVB    TOR0
003063  1 002002 0 1 0 002              LDBR    SELOUT                  ;CLEAR "HLD OUT"
003064  1 064051 3 2 0 02 11            MOVB    TOR0
003065  1 002000 0 1 0 000              LDBR    0                       ;CLEAR "SEL OUT"
003066  1 064051 3 2 0 02 11            MOVB    TOR0
003067  1 002200 0 1 0 200              LDBR    200                     ;SET WAIT CNT
003070  1 072031 3 5 0 01 11            MOVB    AC1
003071  1 010000 0 4 0 000              LDMEM   0                       ;SET EXPECTED DATA
003072  1 032004 1 5 0 00 04  TIOLP6:    DATI    TAGIN0,AC0              ;READ TAG IN LINES
003073  1 040017 2 0 0 00 17            OSM     AC0                     ;CHECK FOR ALL ZEROS
003074  1 115100 4 6 1100               JMPZ    .+4                     ;JUMP IF ZEROS
003075  1 072027 3 5 0 01 07            DECR    AC1                     ;DEC WAIT CNT
003076  1 115516 4 6 1516               JMPZ    DER105                  ;JUMP IF TIMED OUT
003077  1 101072 4 0 1072               JMP     TIOLP6                  ;ELSE, KEEP WAITING
003100  1 016000 0 7 0 000    NOT21:    RETURN
```

```
2709                                    ;*TEST THAT THE CONTROL UNIT RESPONDS CORRECTLY TO THE "SENSE COMMAND" SEQUENCE.
2710                                    ;*READ THE 24 SENSE BYTES.
2711
2712                                    ;CLEAR ALL TAG OUT LINES.
2713                                    ;SET "OPL OUT".
2714                                    ;PUT DRIVE ADDRESS ON THE BUS OUT LINES.
2715                                    ;SET "ADR OUT", "HLD OUT", THEN "SEL OUT".
2716                                    ; CHECK THAT "OPL IN" SETS.
2717
2718  003101  1 001001 0 0 2 001     DTST22: LDMAR   DFLAGS                  ;[4] GET ADDRESS OF FLAGS
2719  003102  1 052011 2 5 0 00 11           MOVMEM  AC0                     ;[4] GET FLAGS, ALL ZERO MEANS MICROCODE
2720                                                                          ;[4]  STARTED AT 'STARTA'
2721  003103  1 001000 0 0 2 000             LDMAR   0                       ;[4] RESTORE MAR TO 0
2722  003104  1 060007 3 0 0 00 07           DEC     AC0                     ;[4] DEC TO TEST FOR ZERO
2723  003105  1 115326 4 6 1326              JMPZ    NOT22                   ;[4] SKIP TEST IF FLAGS ARE ZERO
2724  003106  1 002027 0 1 0 027             LDBR    ^D23                    ;SET LOOP CNT MINUS 1 FOR SENSE BYTES
2725  003107  1 072051 3 5 0 02 11           MOVB    AC2
2726  003110  1 002200 0 1 0 200             LDBR    OPLOUT                  ;SET "OPL OUT"
2727  003111  1 064071 3 2 0 03 11           MOVB    TOR1
2728  003112  1 002000 0 1 0 000             LDBR    0                       ;SET DRIVE ADDRESS = 0
2729  003113  1 064231 3 2 0 11 11           MOVB    BORLO
2730  003114  1 001000 0 0 2 000             LDMAR   0                       ;RESET MAR
2731  003115  1 002020 0 1 0 020             LDBR    ADROUT                  ;SET "ADR OUT"
2732  003116  1 064051 3 2 0 02 11           MOVB    TOR0
2733  003117  1 002030 0 1 0 030             LDBR    HLDOUT+ADROUT           ;SET "HLD OUT"
2734  003120  1 064051 3 2 0 02 11           MOVB    TOR0
2735  003121  1 002032 0 1 0 032             LDBR    SELOUT+HLDOUT+ADROUT    ;SET "SEL OUT"
2736  003122  1 064051 3 2 0 02 11           MOVB    TOR0
2737  003123  1 002200 0 1 0 200             LDBR    200                     ;SET WAIT CNT
2738  003124  1 072031 3 5 0 01 11           MOVB    AC1
2739  003125  1 022004 1 1 0 00 04  SNSLP1: DATI    TAGIN0,BR               ;READ TAG IN BITS
2740  003126  1 111132 4 4 1132              JMPB7   SNSSET                  ;JUMP IF "OPL IN" SET
2741  003127  1 072027 3 5 0 01 07           DECR    AC1                     ;DEC WAIT CNT
2742  003130  1 115515 4 6 1515              JMPZ    DER106                  ;JUMP IF TIMED OUT
2743  003131  1 101125 4 0 1125              JMP     SNSLP1                  ;ELSE, KEEP WAITING
2744
2745                                    ;CLEAR "ADR OUT".
2746                                    ;CHECK THAT "ADR IN" SETS.
2747
2748  003132  1 002012 0 1 0 012     SNSSET: LDBR    SELOUT+HLDOUT           ;CLEAR "ADR OUT"
2749  003133  1 064051 3 2 0 02 11           MOVB    TOR0
2750  003134  1 002200 0 1 0 200             LDBR    200                     ;SET WAIT CNT
2751  003135  1 072031 3 5 0 01 11           MOVB    AC1
2752  003136  1 022004 1 1 0 00 04  SNSLP2: DATI    TAGIN0,BR               ;READ TAG IN BITS
2753  003137  1 107143 4 3 1143              JMPB4   .+4                     ;JUMP IF SET
2754  003140  1 072027 3 5 0 01 07           DECR    AC1                     ;DEC LOOP CNT
2755  003141  1 115514 4 6 1514              JMPZ    DER107                  ;JUMP IF TIMED OUT
2756  003142  1 101136 4 0 1136              JMP     SNSLP2
2757
2758                                    ;CHECK THAT BUS IN LINES CONTAIN ADDRESS REQUESTED(0).
2759
2760  003143  1 032007 1 5 0 00 07           DATI    CBILO,AC0               ;READ BUS IN LINES
2761  003144  1 010000 0 4 0 000             LDMEM   0                       ;SET EXPECTED ADDR TO 0
2762  003145  1 040017 2 0 0 00 17           OSM     AC0                     ;COMPARE
2763  003146  1 115150 4 6 1150              JMPZ    .+2                     ;JUMP IF BUS IN LINES=0
```

```
 2764  003147  1 101513 4 0 1513          JMP     DER110          ;REPORT ERROR
 2765
```

```
2766                                    ;CHECK THAT "BUS0 PE FLAG" IS NOT SET.
2767  003150  1 010000 0 4 0 000             LDMEM   0                       ;SET EXPECTED DATA
2768  003151  1 022000 1 1 0 00 00           DATI    CSR0,BR                 ;READ REG0
2769  003152  1 014000 0 6 0 000             SHR                             ;RIGHT ADJ "BUS0 SE FLAG"
2770  003153  1 105512 4 2 1512              JMPB0   DER111                  ;JUMP IF PARITY ERROR
2771
2772                                    ;PUT SENSE COMMAND (0) ON BUS OUT LINES.
2773                                    ;SET "CMD OUT".
2774                                    ;CHECK THAT "ADR IN" CLEARS.
2775
2776  003154  1 002004 0 1 0 004             LDBR    4                       ;PUT SENSE CMD ON BUS LINES
2777  003155  1 064231 3 2 0 11 11           MOVB    BORLO
2778  003156  1 002013 0 1 0 013             LDBR    CMDOUT+HLDOUT+SELOUT    ;SET "CMD OUT"
2779  003157  1 064051 3 2 0 02 11           MOVB    TOR0
2780  003160  1 002200 0 1 0 200             LDBR    200                     ;SET WAIT CNT
2781  003161  1 072031 3 5 0 01 11           MOVB    AC1
2782  003162  1 022004 1 1 0 00 04  SNSLP3: DATI    TAGIN0,BR               ;READ TAG IN LINES
2783  003163  1 107165 4 3 1165              JMPB4   .+2                     ;JUMP IF "ADR IN" STILL SET
2784  003164  1 101170 4 0 1170              JMP     .+4                     ;ELSE, IT DROPPED
2785  003165  1 072027 3 5 0 01 07           DECR    AC1                     ;DEC WAIT CNT
2786  003166  1 115511 4 6 1511              JMPZ    DER112                  ;JUMP IF TIMED OUT
2787  003167  1 101162 4 0 1162              JMP     SNSLP3                  ;ELSE, KEEP WAITING
2788
2789                                    ;CLEAR "CMD OUT".
2790                                    ;CHECK THAT "STA IN" SETS.
2791
2792  003170  1 002012 0 1 0 012             LDBR    HLDOUT+SELOUT           ;CLEAR "CMD OUT"
2793  003171  1 064051 3 2 0 02 11           MOVB    TOR0
2794  003172  1 002200 0 1 0 200             LDBR    200                     ;SET WAIT CNT.
2795  003173  1 072031 3 5 0 01 11           MOVB    AC1
2796  003174  1 022004 1 1 0 00 04  SNSLP4: DATI    TAGIN0,BR               ;READ TAG IN LINES
2797  003175  1 105201 4 2 1201              JMPB0   .+4                     ;JUMP IF "STA IN" SET
2798  003176  1 072027 3 5 0 01 07           DECR    AC1                     ;DEC WAIT CNT
2799  003177  1 115510 4 6 1510              JMPZ    DER113                  ;JUMP IF TIMED OUT
2800  003200  1 101174 4 0 1174              JMP     SNSLP4                  ;ELSE, KEEP WAITING
2801
2802                                    ;CHECK THAT INITIAL STATUS IS ZERO
2803
2804  003201  1 010000 0 4 0 000             LDMEM   0                       ;SET EXPECTED DATA
2805  003202  1 032007 1 5 0 00 07           DATI    CBILO,AC0               ;READ INITIAL STATUS
2806  003203  1 040017 2 0 0 00 17           OSM     AC0                     ;CHECK IT
2807  003204  1 115206 4 6 1206              JMPZ    .+2                     ;JUMP IF IT'S ZERO
2808  003205  1 101507 4 0 1507              JMP     DER114                  ;REPORT ERROR
2809  003206  1 022000 1 1 0 00 00           DATI    CSR0,BR                 ;READ FLAG STATUS
2810  003207  1 014000 0 6 0 000             SHR                             ;RIGHT ADJUST "BUS0 PE FLAG"
2811  003210  1 105506 4 2 1506              JMPB0   DER115                  ;JUMP IF IT SET
2812
2813                                    ;SET "SRV OUT".
2814                                    ;CHECK THAT "STA IN" CLEARS.
2815
2816  003211  1 002212 0 1 0 212             LDBR    SRVOUT+HLDOUT+SELOUT    ;SET "SRV OUT"
2817  003212  1 064051 3 2 0 02 11           MOVB    TOR0
2818  003213  1 002200 0 1 0 200             LDBR    200                     ;SET WAIT CNT
2819  003214  1 072031 3 5 0 01 11           MOVB    AC1
2820  003215  1 022004 1 1 0 00 04  SNSLP5: DATI    TAGIN0,BR               ;READ TAG IN LINES
```

```
2821  003216  1 105220 4 2 1220            JMPB0   .+2                     ;JUMP IF "STA IN" STILL SET
2822  003217  1 101223 4 0 1223            JMP     .+4                     ;ELSE, CLEARED OKAY
2823  003220  1 072027 3 5 0 01 07         DECR    AC1                     ;DEC WAIT CNT
2824  003221  1 115505 4 6 1505            JMPZ    DER116                  ;JUMP IF TIMED OUT
2825  003222  1 101215 4 0 1215            JMP     SNSLP5                  ;ELSE, KEEP WAITING
2826
2827                                  ;CLEAR "SRV OUT".
2828                                  ;CHECK THAT "SRV IN" SETS.
2829
2830  003223  1 002012 0 1 0 012           LDBR    HLDOUT+SELOUT           ;CLEAR "SRV OUT"
2831  003224  1 064051 3 2 0 02 11         MOVB    TOR0
2832  003225  1 002200 0 1 0 200           LDBR    200                     ;SET WAIT CNT
2833  003226  1 072031 3 5 0 01 11         MOVB    AC1
2834  003227  1 022005 1 1 0 00 05  SNSLP6: DATI    TAGIN1,BR               ;READ TAG IN LINES
2835  003230  1 111234 4 4 1234            JMPB7   .+4                     ;JUMP IF "SRV IN" SET
2836  003231  1 072027 3 5 0 01 07         DECR    AC1                     ;DEC WAIT CNT
2837  003232  1 115504 4 6 1504            JMPZ    DER117                  ;JUMP IF TIMED OUT
2838  003233  1 101227 4 0 1227            JMP     SNSLP6                  ;ELSE, KEEP WAITING
2839
2840                                  ;READ AND SAVE THE SENSE BYTE.
2841                                  ;CHECK THAT "BUS0 PE FLAG" DID NOT SET.
2842
2843  003234  1 032007 1 5 0 00 07  NXTSB:  DATI    CBILO,AC0               ;READ SENSE BYTE
2844  003235  1 022000 1 1 0 00 00         DATI    CSR0,BR                 ;READ FLAG STATUS
2845  003236  1 014000 0 6 0 000           SHR                             ;RIGHT ADJUST "BUS0 PE FLAG"
2846  003237  1 105503 4 2 1503            JMPB0   DER120                  ;JUMP IF IT SET
2847
2848                                  ;SET "SRV OUT".
2849                                  ;CHECK THAT "SRV IN" CLEARS.
2850
2851  003240  1 002212 0 1 0 212  SBYTER: LDBR    SRVOUT+HLDOUT+SELOUT    ;SET "SRV OUT"
2852  003241  1 064051 3 2 0 02 11         MOVB    TOR0
2853  003242  1 002200 0 1 0 200           LDBR    200                     ;SET WAIT CNT
2854  003243  1 072031 3 5 0 01 11         MOVB    AC1
2855  003244  1 022005 1 1 0 00 05  SNSLP7: DATI    TAGIN1,BR               ;READ TAG IN LINES
2856  003245  1 111247 4 4 1247            JMPB7   .+2                     ;JUMP IF "SRV IN" STILL SET
2857  003246  1 101252 4 0 1252            JMP     .+4                     ;ELSE, CLEARED OKAY
2858  003247  1 072027 3 5 0 01 07         DECR    AC1                     ;DEC WAIT CNT
2859  003250  1 115502 4 6 1502            JMPZ    DER121                  ;JUMP IF TIMED OUT
2860  003251  1 101244 4 0 1244            JMP     SNSLP7                  ;ELSE, KEEP
2861
2862                                  ;CLEAR "SRV OUT".
2863                                  ;IF RECEIVED ALL 24 SENSE BYTES, TERMINATE SEQUENCE.
2864                                  ;ELSE, CHECK THAT "SRV IN" SETS AND CONTINUE RECEIVING BYTES.
2865
2866  003252  1 002012 0 1 0 012           LDBR    HLDOUT+SELOUT           ;CLEAR "SRV OUT"
2867  003253  1 064051 3 2 0 02 11         MOVB    TOR0
2868  003254  1 072047 3 5 0 02 07         DECR    AC2                     ;DEC BYTE CNT
2869  003255  1 115266 4 6 1266            JMPZ    BYTDON                  ;JUMP IF READ ALL 24 BYTES
2870  003256  1 002200 0 1 0 200           LDBR    200                     ;SET WAIT CNT
2871  003257  1 072031 3 5 0 01 11         MOVB    AC1
2872  003260  1 022005 1 1 0 00 05  SNSLP8: DATI    TAGIN1,BR               ;READ TAG IN LINES
2873  003261  1 111265 4 4 1265            JMPB7   .+4                     ;JUMP IF "SRV IN" SET
2874  003262  1 072027 3 5 0 01 07         DECR    AC1                     ;DEC WAIT CNT
2875  003263  1 115501 4 6 1501            JMPZ    DER122                  ;JUMP IF TIMED OUT
```

```
2876  003264  1 101260 4 0 1260              JMP     SNSLP8                  ;ELSE, KEEP WAITING
2877  003265  1 101234 4 0 1234              JMP     NXTSB                   ;GO GET NEXT SENSE BYTE
2878
2879                                  ;TERMINATE THE SENSE COMMAND SEQUENCE.
2880                                  ;CHECK THAT "STA IN" SET IN RESPONSE TO LAST "SRV OUT" CLEARING.
2881
2882  003266  1 002200 0 1 0 200     BYTDON: LDBR    200                     ;SET WAIT CNT.
2883  003267  1 072031 3 5 0 01 11           MOVB    AC1
2884  003270  1 022004 1 1 0 00 04  SNSLP9: DATI    TAGIN0,BR               ;READ TAG IN LINES
2885  003271  1 105275 4 2 1275              JMPB0   .+4                     ;JUMP IF "STA IN" SET
2886  003272  1 072027 3 5 0 01 07          DECR    AC1                     ;DEC WAIT CNT
2887  003273  1 115500 4 6 1500              JMPZ    DER123                  ;JUMP IF TIMED OUT
2888  003274  1 101270 4 0 1270              JMP     SNSLP9                  ;ELSE, KEEP WAITING
2889
2890                                  ;SET "SRV OUT".
2891                                  ;CHECK THAT "STA IN" CLEARS.
2892
2893  003275  1 002212 0 1 0 212              LDBR    SRVOUT+HLDOUT+SELOUT    ;SET "SRV OUT"
2894  003276  1 064051 3 2 0 02 11           MOVB    TOR0
2895  003277  1 002200 0 1 0 200              LDBR    200                     ;SET WAIT CNT
2896  003300  1 072031 3 5 0 01 11           MOVB    AC1
2897  003301  1 022004 1 1 0 00 04  SNSLPX: DATI    TAGIN0,BR               ;READ TAG IN LINES
2898  003302  1 105304 4 2 1304              JMPB0   .+2                     ;JUMP IF "STA IN" STILL SET
2899  003303  1 101307 4 0 1307              JMP     .+4                     ;ELSE, CLEARED OKAY
2900  003304  1 072027 3 5 0 01 07          DECR    AC1                     ;DEC WAIT CNT
2901  003305  1 115477 4 6 1477              JMPZ    DER124                  ;JUMP IF TIMED OUT
2902  003306  1 101301 4 0 1301              JMP     SNSLPX                  ;ELSE, KEEP
2903
2904                                  ;CLEAR "SRV OUT", "HLD OUT", THEN "SEL OUT".
2905                                  ;CHECK THAT "OPL IN" CLEARS.
2906
2907  003307  1 002012 0 1 0 012              LDBR    HLDOUT+SELOUT           ;CLEAR "SRV OUT"
2908  003310  1 064051 3 2 0 02 11           MOVB    TOR0
2909  003311  1 002002 0 1 0 002              LDBR    SELOUT                  ;CLEAR "HLD OUT"
2910  003312  1 064051 3 2 0 02 11           MOVB    TOR0
2911  003313  1 002000 0 1 0 000              LDBR    0                       ;CLEAR "SEL OUT"
2912  003314  1 064051 3 2 0 02 11           MOVB    TOR0
2913  003315  1 002200 0 1 0 200              LDBR    200                     ;SET WAIT CNT
2914  003316  1 072031 3 5 0 01 11           MOVB    AC1
2915  003317  1 002000 0 1 0 000              LDBR    0                       ;SET COMPARE WORD
2916  003320  1 032004 1 5 0 00 04  SNSLPY: DATI    TAGIN0,AC0              ;READ TAG IN LINES
2917  003321  1 060017 3 0 0 00 17          OSB     AC0                     ;CHECK FOR ZEROS
2918  003322  1 115326 4 6 1326              JMPZ    .+4                     ;JUMP IF ZEROS
2919  003323  1 072027 3 5 0 01 07          DECR    AC1                     ;DEC WAIT CNT
2920  003324  1 115476 4 6 1476              JMPZ    DER125                  ;JUMP IF TIMED OUT
2921  003325  1 101320 4 0 1320              JMP     SNSLPY                  ;ELSE, KEEP WAITING
2922  003326  1 016000 0 7 0 000     NOT22:  RETURN
2923
```

```
                                SUBTTL  DIVIDE ROUTINE

                                ;THIS ROUTINE WILL DIVIDE THE NEGATIVE BYTE COUNT IN MASSBUS REGISTER 25
                                ;BY AN 8-BIT POSITIVE DIVISOR AND RETURN
                                ;BOTH QUOTIENT AND REMAINDER
                                ;[10] [11] [12] THIS ROUTINE WAS MOVED FROM BANK 1 TO BANK 2 OF MEMORY TO MAKE
                                ; ROOM FOR THESE EDITS

                                ;ENTER WITH BYTE COUNT IN MASSBUS REGISTER 25
                                ;           DIVISOR IN MEMORY
                                ;ON EXIT QUOTIENT WILL BE IN AC3-AC4 (POSITIVE)
                                ;         REMAINDER WILL BE IN AC2
                                ;         MEMORY WILL NOT BE CHANGED
                                ;AC5 IS USED AS SHIFT COUNTER AND WILL CONTAIN -1 ON EXIT

003327  1 002000 0 1 0 000      DIVIDE: LDBR    0                       ;GET A ZERO
003330  1 072051 3 5 0 02 11            MOVB    AC2                     ; IN AC2
003331  1 122002 5 1 0 00 02            DATI    CNTL,BR                 ;GET LOW BYTE OF COUNT
003332  1 062056 3 1 0 02 16            TSB     AC2,BR                  ;NEGATE IT, MAKING COUNT POSITIVE
003333  1 072111 3 5 0 04 11            MOVB    AC4                     ;PLACE IN AC4
003334  1 122003 5 1 0 00 03            DATI    CNTH,BR                 ;GET HIGH BYTE
003335  1 062042 3 1 0 02 02            OSBC    AC2,BR                  ;CONTINUE THE NEGATION
003336  1 072071 3 5 0 03 11            MOVB    AC3                     ;PLACE IN AC3
003337  1 072044 3 5 0 02 04            ADCR    AC2                     ;ADD OVERFLOW TO AC2
003340  1 002017 0 1 0 017              LDBR    ^D15                    ;SET UP SHIFT COUNT
003341  1 072131 3 5 0 05 11            MOVB    AC5                     ; IN AC5
003342  1 072105 3 5 0 04 05  DIVLOP: SHLR    AC4                     ;SHIFT NUMBER LEFT ONE PLACE
003343  1 072066 3 5 0 03 06            ROTLR   AC3
003344  1 072046 3 5 0 02 06            ROTLR   AC2
003345  1 040056 2 0 0 02 16            TSM     AC2                     ;CHECK IF OVERFLOW SIZE OF DIVISOR
003346  1 113350 4 5 1350               JMPC    .+2                     ;YES
003347  1 101352 4 0 1352               JMP     .+3                     ;NO, DON'T SUBTRACT
003350  1 052056 2 5 0 02 16            TSMR    AC2                     ;SUBTRACT OUT DIVISOR
003351  1 072103 3 5 0 04 03            INCR    AC4                     ;INCREMENT QUOTIENT
003352  1 072127 3 5 0 05 07            DECR    AC5                     ;DECREMENT SHIFT COUNT
003353  1 113342 4 5 1342               JMPC    DIVLOP                  ;SHIFT ALL BITS
003354  1 016000 0 7 0 000              RETURN
```

```
 2962  003400                                  .LOC    3400
 2963
 2964                                  ;*DISPATCH TABLE FOR DIAGS RUN AT STARTUP.
 2965
 2966  003400  1 100271 4 0 0271      ONCDGT: JMP     DTST0                   ;RUN DIAG TEST 0
 2967  003401  1 100277 4 0 0277              JMP     DTST1                   ;RUN DIAG TEST 1
 2968  003402  1 100316 4 0 0316              JMP     DTST2                   ;RUN DIAG TEST 2
 2969  003403  1 100336 4 0 0336              JMP     DTST3                   ;RUN DIAG TEST 3
 2970  003404  1 100350 4 0 0350              JMP     DTST4                   ;RUN DIAG TEST 4
 2971  003405  1 100363 4 0 0363              JMP     DTST5                   ;RUN DIAG TEST 5
 2972  003406  1 100373 4 0 0373              JMP     DTST6                   ;RUN DIAG TEST 6
 2973  003407  1 100405 4 0 0405              JMP     DTST7                   ;RUN DIAG TEST 7
 2974  003410  1 100427 4 0 0427              JMP     DTST8                   ;RUN DIAG TEST 8
 2975  003411  1 100524 4 0 0524              JMP     DTST9                   ;RUN DIAG TEST 9
 2976  003412  1 100541 4 0 0541              JMP     DTST10                  ;RUN DIAG TEST 10
 2977  003413  1 100557 4 0 0557              JMP     DTST11                  ;RUN DIAG TEST 11
 2978  003414  1 100572 4 0 0572              JMP     DTST12                  ;RUN DIAG TEST 12
 2979  003415  1 100604 4 0 0604              JMP     DTST13                  ;RUN DIAG TEST 13
 2980  003416  1 100622 4 0 0622              JMP     DTST14                  ;RUN DIAG TEST 14
 2981  003417  1 100623 4 0 0623              JMP     DTST15                  ;RUN DIAG TEST 15
 2982  003420  1 100643 4 0 0643              JMP     DTST16                  ;RUN DIAG TEST 16
 2983  003421  1 100670 4 0 0670              JMP     DTST17                  ;RUN DIAG TEST 17
 2984  003422  1 100705 4 0 0705              JMP     DTST18                  ;RUN DIAG TEST 18
 2985  003423  1 100717 4 0 0717              JMP     DTST19                  ;RUN DIAG TEST 19
 2986  003424  1 100730 4 0 0730              JMP     DTST20                  ;RUN DIAG TEST 20
 2987  003425  1 100750 4 0 0750              JMP     DTST21                  ;RUN DIAG TEST 21
 2988  003426  1 101101 4 0 1101              JMP     DTST22                  ;RUN DIAG TEST 22
 2989  003427  1 101430 4 0 1430              JMP     ONCEND                  ;STOP RUNNING DIAGS
 2990
 2991  003430  1 002100 0 1 0 100     ONCEND: LDBR    SPRES                   ;RESET STACK POINTER
 2992  003431  1 066371 3 3 0 17 11           MOVB    IOSEL
 2993  003432  1 001001 0 0 2 001             LDMAR   DFLAGS                  ;SET ADDR OF FLAGS
 2994  003433  1 043411 2 1 3 00 11           MOVMEM  BR,I                    ;GET FLAGS
 2995  003434  1 107437 4 3 1437              JMPB4   DGONLY                  ;JUMP IF RUNNING DIAGS ONLY
 2996  003435  1 010042 0 4 0 042             LDMEM   IDLDGT&377              ;SET DISPATCH ADDRESS TO TOP OF TABLE
 2997  003436  1 100242 4 0 0242              JMP     CLNUPB                  ;RESUME CLEANUP
 2998
 2999  003437  1 002011 0 1 0 011     DGONLY: LDBR    MB                      ;SELECT MASSBUS INTERFACE
 3000  003440  1 066371 3 3 0 17 11           MOVB    IOSEL
 3001  003441  1 100224 4 0 0224              JMP     RERUN                   ;RERUN TESTS
 3002
```

```
3003                                    ;*DISPATCH TABLE FOR DIAGS WHICH ARE RUN DURING IDLE LOOP.
3004
3005  003442  1 100271 4 0 0271  IDLDGT: JMP     DTST0                   ;RUN DIAG TEST 0
3006  003443  1 100277 4 0 0277          JMP     DTST1                   ;RUN DIAG TEST 1
3007  003444  1 100316 4 0 0316          JMP     DTST2                   ;RUN DIAG TEST 2
3008  003445  1 100336 4 0 0336          JMP     DTST3                   ;RUN DIAG TEST 3
3009  003446  1 100350 4 0 0350          JMP     DTST4                   ;RUN DIAG TEST 4
3010  003447  1 100363 4 0 0363          JMP     DTST5                   ;RUN DIAG TEST 5
3011  003450  1 100373 4 0 0373          JMP     DTST6                   ;RUN DIAG TEST 6
3012  003451  1 100405 4 0 0405          JMP     DTST7                   ;RUN DIAG TEST 7
3013  003452  1 100427 4 0 0427          JMP     DTST8                   ;RUN DIAG TEST 8
3014  003453  1 100524 4 0 0524          JMP     DTST9                   ;RUN DIAG TEST 9
3015  003454  1 100541 4 0 0541          JMP     DTST10                  ;RUN DIAG TEST 10
3016  003455  1 100557 4 0 0557          JMP     DTST11                  ;RUN DIAG TEST 11
3017  003456  1 100572 4 0 0572          JMP     DTST12                  ;RUN DIAG TEST 12
3018  003457  1 100604 4 0 0604          JMP     DTST13                  ;RUN DIAG TEST 13
3019  003460  1 100622 4 0 0622          JMP     DTST14                  ;RUN DIAG TEST 14
3020  003461  1 100623 4 0 0623          JMP     DTST15                  ;RUN DIAG TEST 15
3021  003462  1 100643 4 0 0643          JMP     DTST16                  ;RUN DIAG TEST 16
3022  003463  1 100670 4 0 0670          JMP     DTST17                  ;RUN DIAG TEST 17
3023  003464  1 100705 4 0 0705          JMP     DTST18                  ;RUN DIAG TEST 18
3024  003465  1 100717 4 0 0717          JMP     DTST19                  ;RUN DIAG TEST 19
3025  003466  1 100730 4 0 0730          JMP     DTST20                  ;RUN DIAG TEST 20
3026  003467  1 002071 0 1 0 071         JUMP    IDLEND
3027  003470  1 161631 7 0 3 11 11
3028
3029  003471  1 001002 0 0 2 002  IDLEND: LDMAR   TSTADR                 ;SET MAR TO ADDR OF DISPATCH ADDR
3030  003472  1 011442 0 4 3 042          LDMEM   IDLDGT&377,I            ;SET DISPATCH ADDRESS TO TOP OF TABLE
3031  003473  1 010000 0 4 0 000          LDMEM   0                       ;SET WAIT CNT TO ZERO
3032  003474  1 002031 0 1 0 031          JUMP    IDLED                   ;GO DO RESET AND ENTER IDLE LOOP
3033  003475  1 160211 7 0 0 10 11
3034
```

```
3035                                    ;*ERROR REPORTING ROUTINE FOR MICRODIAGNOSTICS.
3036                                    ;AC7 CONTAINS A 0 INITIALLY, THEN DEPENDING ON ENTRY POINT, IT GETS
3037                                    ;INCREMENTED TO BECOME THE ERROR NUMBER.
3038
3039  003476  1 072163 3 5 0 07 03    DER125: INCR    AC7                     ;INCREMENT ERROR NUMBER
3040  003477  1 072163 3 5 0 07 03    DER124: INCR    AC7                     ;INCREMENT ERROR NUMBER
3041  003500  1 072163 3 5 0 07 03    DER123: INCR    AC7                     ;INCREMENT ERROR NUMBER
3042  003501  1 072163 3 5 0 07 03    DER122: INCR    AC7                     ;INCREMENT ERROR NUMBER
3043  003502  1 072163 3 5 0 07 03    DER121: INCR    AC7                     ;INCREMENT ERROR NUMBER
3044  003503  1 072163 3 5 0 07 03    DER120: INCR    AC7                     ;INCREMENT ERROR NUMBER
3045  003504  1 072163 3 5 0 07 03    DER117: INCR    AC7                     ;INCREMENT ERROR NUMBER
3046  003505  1 072163 3 5 0 07 03    DER116: INCR    AC7                     ;INCREMENT ERROR NUMBER
3047  003506  1 072163 3 5 0 07 03    DER115: INCR    AC7                     ;INCREMENT ERROR NUMBER
3048  003507  1 072163 3 5 0 07 03    DER114: INCR    AC7                     ;INCREMENT ERROR NUMBER
3049  003510  1 072163 3 5 0 07 03    DER113: INCR    AC7                     ;INCREMENT ERROR NUMBER
3050  003511  1 072163 3 5 0 07 03    DER112: INCR    AC7                     ;INCREMENT ERROR NUMBER
3051  003512  1 072163 3 5 0 07 03    DER111: INCR    AC7                     ;INCREMENT ERROR NUMBER
3052  003513  1 072163 3 5 0 07 03    DER110: INCR    AC7                     ;INCREMENT ERROR NUMBER
3053  003514  1 072163 3 5 0 07 03    DER107: INCR    AC7                     ;INCREMENT ERROR NUMBER
3054  003515  1 072163 3 5 0 07 03    DER106: INCR    AC7                     ;INCREMENT ERROR NUMBER
3055  003516  1 072163 3 5 0 07 03    DER105: INCR    AC7                     ;INCREMENT ERROR NUMBER
3056  003517  1 072163 3 5 0 07 03    DER104: INCR    AC7                     ;INCREMENT ERROR NUMBER
3057  003520  1 072163 3 5 0 07 03    DER103: INCR    AC7                     ;INCREMENT ERROR NUMBER
3058  003521  1 072163 3 5 0 07 03    DER102: INCR    AC7                     ;INCREMENT ERROR NUMBER
3059  003522  1 072163 3 5 0 07 03    DER101: INCR    AC7                     ;INCREMENT ERROR NUMBER
3060  003523  1 072163 3 5 0 07 03    DER100: INCR    AC7                     ;INCREMENT ERROR NUMBER
3061  003524  1 072163 3 5 0 07 03    DERR77: INCR    AC7                     ;INCREMENT ERROR NUMBER
3062  003525  1 072163 3 5 0 07 03    DERR76: INCR    AC7                     ;INCREMENT ERROR NUMBER
3063  003526  1 072163 3 5 0 07 03    DERR75: INCR    AC7                     ;INCREMENT ERROR NUMBER
3064  003527  1 072163 3 5 0 07 03    DERR74: INCR    AC7                     ;INCREMENT ERROR NUMBER
3065  003530  1 072163 3 5 0 07 03    DERR73: INCR    AC7                     ;INCREMENT ERROR NUMBER
3066  003531  1 072163 3 5 0 07 03    DERR72: INCR    AC7                     ;INCREMENT ERROR NUMBER
3067  003532  1 072163 3 5 0 07 03    DERR71: INCR    AC7                     ;INCREMENT ERROR NUMBER
3068  003533  1 072163 3 5 0 07 03    DERR70: INCR    AC7                     ;INCREMENT ERROR NUMBER
3069  003534  1 072163 3 5 0 07 03    DERR67: INCR    AC7                     ;INCREMENT ERROR NUMBER
3070  003535  1 072163 3 5 0 07 03    DERR66: INCR    AC7                     ;INCREMENT ERROR NUMBER
3071  003536  1 072163 3 5 0 07 03    DERR65: INCR    AC7                     ;INCREMENT ERROR NUMBER
3072  003537  1 072163 3 5 0 07 03    DERR64: INCR    AC7                     ;INCREMENT ERROR NUMBER
3073  003540  1 072163 3 5 0 07 03    DERR63: INCR    AC7                     ;INCREMENT ERROR NUMBER
3074  003541  1 072163 3 5 0 07 03    DERR62: INCR    AC7                     ;INCREMENT ERROR NUMBER
3075  003542  1 072163 3 5 0 07 03    DERR61: INCR    AC7                     ;INCREMENT ERROR NUMBER
3076  003543  1 072163 3 5 0 07 03    DERR60: INCR    AC7                     ;INCREMENT ERROR NUMBER
3077  003544  1 072163 3 5 0 07 03    DERR57: INCR    AC7                     ;INCREMENT ERROR NUMBER
3078  003545  1 072163 3 5 0 07 03    DERR56: INCR    AC7                     ;INCREMENT ERROR NUMBER
3079  003546  1 072163 3 5 0 07 03    DERR55: INCR    AC7                     ;INCREMENT ERROR NUMBER
3080  003547  1 072163 3 5 0 07 03    DERR54: INCR    AC7                     ;INCREMENT ERROR NUMBER
3081  003550  1 072163 3 5 0 07 03    DERR53: INCR    AC7                     ;INCREMENT ERROR NUMBER
3082  003551  1 072163 3 5 0 07 03    DERR52: INCR    AC7                     ;INCREMENT ERROR NUMBER
3083  003552  1 072163 3 5 0 07 03    DERR51: INCR    AC7                     ;INCREMENT ERROR NUMBER
3084  003553  1 072163 3 5 0 07 03    DERR50: INCR    AC7                     ;INCREMENT ERROR NUMBER
3085  003554  1 072163 3 5 0 07 03    DERR47: INCR    AC7                     ;INCREMENT ERROR NUMBER
3086  003555  1 072163 3 5 0 07 03    DERR46: INCR    AC7                     ;INCREMENT ERROR NUMBER
3087  003556  1 072163 3 5 0 07 03    DERR45: INCR    AC7                     ;INCREMENT ERROR NUMBER
3088  003557  1 072163 3 5 0 07 03    DERR44: INCR    AC7                     ;INCREMENT ERROR NUMBER
3089
```

```
3090  003560  1 072163 3 5 0 07 03    DERR43: INCR    AC7             ;INCREMENT ERROR NUMBER
3091  003561  1 072163 3 5 0 07 03    DERR42: INCR    AC7             ;INCREMENT ERROR NUMBER
3092  003562  1 072163 3 5 0 07 03    DERR41: INCR    AC7             ;INCREMENT ERROR NUMBER
3093  003563  1 072163 3 5 0 07 03    DERR40: INCR    AC7             ;INCREMENT ERROR NUMBER
3094  003564  1 072163 3 5 0 07 03    DERR37: INCR    AC7             ;INCREMENT ERROR NUMBER
3095  003565  1 072163 3 5 0 07 03    DERR36: INCR    AC7             ;INCREMENT ERROR NUMBER
3096  003566  1 072163 3 5 0 07 03    DERR35: INCR    AC7             ;INCREMENT ERROR NUMBER
3097  003567  1 072163 3 5 0 07 03    DERR34: INCR    AC7             ;INCREMENT ERROR NUMBER
3098  003570  1 072163 3 5 0 07 03    DERR33: INCR    AC7             ;INCREMENT ERROR NUMBER
3099  003571  1 072163 3 5 0 07 03    DERR32: INCR    AC7             ;INCREMENT ERROR NUMBER
3100  003572  1 072163 3 5 0 07 03    DERR31: INCR    AC7             ;INCREMENT ERROR NUMBER
3101  003573  1 072163 3 5 0 07 03    DERR30: INCR    AC7             ;INCREMENT ERROR NUMBER
3102  003574  1 072163 3 5 0 07 03    DERR27: INCR    AC7             ;INCREMENT ERROR NUMBER
3103  003575  1 072163 3 5 0 07 03    DERR26: INCR    AC7             ;INCREMENT ERROR NUMBER
3104  003576  1 072163 3 5 0 07 03    DERR25: INCR    AC7             ;INCREMENT ERROR NUMBER
3105  003577  1 072163 3 5 0 07 03    DERR24: INCR    AC7             ;INCREMENT ERROR NUMBER
3106  003600  1 072163 3 5 0 07 03    DERR23: INCR    AC7             ;INCREMENT ERROR NUMBER
3107  003601  1 072163 3 5 0 07 03    DERR22: INCR    AC7             ;INCREMENT ERROR NUMBER
3108  003602  1 072163 3 5 0 07 03    DERR21: INCR    AC7             ;INCREMENT ERROR NUMBER
3109  003603  1 072163 3 5 0 07 03    DERR20: INCR    AC7             ;INCREMENT ERROR NUMBER
3110  003604  1 072163 3 5 0 07 03    DERR17: INCR    AC7             ;INCREMENT ERROR NUMBER
3111  003605  1 072163 3 5 0 07 03    DERR16: INCR    AC7             ;INCREMENT ERROR NUMBER
3112  003606  1 072163 3 5 0 07 03    DERR15: INCR    AC7             ;INCREMENT ERROR NUMBER
3113  003607  1 072163 3 5 0 07 03    DERR14: INCR    AC7             ;INCREMENT ERROR NUMBER
3114  003610  1 072163 3 5 0 07 03    DERR13: INCR    AC7             ;INCREMENT ERROR NUMBER
3115  003611  1 072163 3 5 0 07 03    DERR12: INCR    AC7             ;INCREMENT ERROR NUMBER
3116  003612  1 072163 3 5 0 07 03    DERR11: INCR    AC7             ;INCREMENT ERROR NUMBER
3117  003613  1 072163 3 5 0 07 03    DERR10: INCR    AC7             ;INCREMENT ERROR NUMBER
3118  003614  1 072163 3 5 0 07 03    DERR7:  INCR    AC7             ;INCREMENT ERROR NUMBER
3119  003615  1 072163 3 5 0 07 03    DERR6:  INCR    AC7             ;INCREMENT ERROR NUMBER
3120  003616  1 072163 3 5 0 07 03    DERR5:  INCR    AC7             ;INCREMENT ERROR NUMBER
3121  003617  1 072163 3 5 0 07 03    DERR4:  INCR    AC7             ;INCREMENT ERROR NUMBER
3122  003620  1 072163 3 5 0 07 03    DERR3:  INCR    AC7             ;INCREMENT ERROR NUMBER
3123  003621  1 072163 3 5 0 07 03    DERR2:  INCR    AC7             ;INCREMENT ERROR NUMBER
3124  003622  1 072163 3 5 0 07 03    DERR1:  INCR    AC7             ;INCREMENT ERROR NUMBER
3125  003623  1 002011 0 1 0 011      DERR0:  LDBR    MB              ;SELECT MASSBUS INTERFACE
3126  003624  1 066371 3 3 0 17 11            MOVB    IOSEL
3127  003625  1 062170 3 1 0 07 10            MOV     AC7,BR          ;PUT ERROR NUMBER INTO DXGP6
3128  003626  1 066111 3 3 0 04 11            MOVB    XSTAT1
3129  003627  1 002000 0 1 0 000              LDBR    0
3130  003630  1 066131 3 3 0 05 11            MOVB    XSTAT0
3131  003631  1 002011 0 1 0 011              LDBR    D.FAIL          ;SET ERROR CODE
3132  003632  1 064051 3 2 0 02 11            MOVB    MPECR
3133  003633  1 001001 0 0 2 001              LDMAR   DFLAGS          ;SET ADDR OF FLAGS
3134  003634  1 042011 2 1 0 00 11            MOVMEM  BR              ;PUT FLAGS INTO BR
3135  003635  1 111637 4 4 1637               JMPB7   .+2             ;JUMP IF SHOULD LOOP ON ERROR
3136  003636  1 100106 4 0 0106               JMP     HALT            ;ELSE, STOP MP
3137  003637  1 072171 3 5 0 07 11            MOVB    AC7             ;SAVE DFLAGS VALUE
3138  003640  1 002002 0 1 0 002              LDBR    2               ;SET MASK OF LOOPING FLAG
3139  003641  1 062174 3 1 0 07 14            LORB    AC7,BR          ;SET LOOPING BIT AND
3140  003642  1 071411 3 4 3 00 11            MOVB    MEM,I           ;STORE IT IN FLAGS WORDS
3141  003643  1 016000 0 7 0 000              RETURN                  ;RERUN FAILING TEST
3142
```

```
3143                                    SUBTTL  DIAGNOSTIC CONTROL
3144
3145                                    ;THIS CONTROL ROUTINE WAS MOVED AS PART OF EDIT [4]
3146
3147  003644  1 002000 0 1 0 000    RNDIAG: LDBR    0                       ;CLEAR ERROR NUMBER REGISTER
3148  003645  1 072171 3 5 0 07 11          MOVB    AC7
3149  003646  1 001002 0 0 2 002          LDMAR   TSTADR                  ;SET MAR TO DISPATCH ADDR
3150  003647  1 042011 2 1 0 00 11          MOVMEM  BR                      ;PUT DISPATCH ADDR IN BR
3151  003650  1 001000 0 0 2 000          LDMAR   0                       ;RESET MAR
3152  003651  1 000400 0 0 1 000          LDMARX  0                       ;CLEAR MAR EXT BITS
3153  003652  1 177631 7 7 3 11 11          JMPSUB  @BR,IDLDGT              ;GO RUN DIAG TEST
3154  003653  1 001001 0 0 2 001          LDMAR   DFLAGS                  ;SET ADDR OF FLAGS
3155  003654  1 043411 2 1 3 00 11          MOVMEM  BR,I                    ;GET FLAGS INTO BR
3156  003655  1 014000 0 6 0 000          SHR                             ;RIGHT ADJUST LOOPING ON ERROR BIT
3157  003656  1 105644 4 2 1644           JMPB0   RNDIAG                  ;REPEAT TEST IF LOOPING
3158  003657  1 052011 2 5 0 00 11          MOVMEM  AC0                     ;GET DISPATCH ADDR
3159  003660  1 071403 3 4 3 00 03          INC     AC0,MEM,I               ;INC AND STORE IT
3160  003661  1 010000 0 4 0 000          LDMEM   0                       ;SET WAIT CNT TO ZERO SO NEXT TEST WILL RUN
3161  003662  1 002033 0 1 0 033          JUMP    IDLED1                  ;GO BACK TO IDLE LOOP
3162  003663  1 160211 7 0 0 10 11
3163
```

```
3164                                    ;THIS WAIT LOOP IS USED TO WAIT AT LEAST 5 MS AFTER SETTING ONLINE AND
3165                                    ;BEFORE OPERATING ANY TAG LINES. THIS IS TO ALLOW BYPASS RELAY SETTLING.
3166
3167  003664  1 002036 0 1 0 036    WAIT:   LDBR    ^D30                    ;SET OUTER LOOP WAIT COUNT
3168  003665  1 072011 3 5 0 00 11          MOVB    AC0
3169  003666  1 002317 0 1 0 317    WAITO:  LDBR    ^D207                   ;SET INNER LOOP WAIT COUNT
3170  003667  1 072031 3 5 0 01 11          MOVB    AC1
3171  003670  1 072027 3 5 0 01 07    WAITI:  DECR    AC1                     ;DEC INNER LOOP COUNT
3172  003671  1 115673 4 6 1673              JMPZ    .+2                     ;JUMP IF INNER LOOP TIMED OUT
3173  003672  1 101670 4 0 1670              JMP     WAITI                   ;ELSE, CONTINUE
3174  003673  1 072007 3 5 0 00 07          DECR    AC0                     ;DEC OUTER LOOP COUNT
3175  003674  1 115676 4 6 1676              JMPZ    .+2                     ;JUMP IF OUTER LOOP TIMED OUT
3176  003675  1 101666 4 0 1666              JMP     WAITO                   ;ELSE CONTINUE
3177  003676  1 016000 0 7 0 000             RETURN
3178
```

```
                                        SUBTTL  FATAL ERROR HANDLERS

                                        ;WRONG DRIVE TYPE OR HARDWARE VERSION FOR THIS MICRO-CODE

003677  1 002010 0 1 0 010      WRONGD: LDBR    F.WDT                   ;GET ERROR NUMBER
003700  1 064051 3 2 0 02 11           MOVB    MPECR                   ;LOAD INTO ERROR CODE REGISTER
003701  1 002102 0 1 0 102              LDBR    MPERR+CLRGO             ;GET ERROR BIT
003702  1 064031 3 2 0 01 11           MOVB    MPSCR1                  ;SET IT
003703  1 016000 0 7 0 000              RETURN                          ;HALT THE DX20 WITH A PUSH LIST UNDERFLOW ERROR

                                        ;THIS ROUTINE RESTORES THE FLAGS TO AC4 AFTER THEY ARE DESTROYED BY THE READ
                                        ;TIMEOUT LOOP.  THIS WAS PUT IN AS PART OF EDIT [5].

003704  1 002011 0 1 0 011      GETFLG: LDBR    MB                      ;SELECT MASSBUS INTERFACE
003705  1 066371 3 3 0 17 11           MOVB    IOSEL
003706  1 032117 1 5 0 04 17           DATI    FLAGS,AC4               ;RESTORE FLAGS
003707  1 002022 0 1 0 022              LDBR    DP                      ;SELECT DATA PATH AGAIN
003710  1 066371 3 3 0 17 11           MOVB    IOSEL
003711  1 016000 0 7 0 000              RETURN
```

```
3199                                    ;EACH OF THE FOLLOWING ROUTINES WILL COPY THE CONTENTS OF 10
3200                                    ;REGISTERS FROM THE SELECTED INTERFACE INTO MEMORY
3201                                    ;THESE LOG ROUTINES WERE MOVED FROM THE LOWER BANK OF MEMORY TO HERE AS
3202                                    ;PART OF EDIT [3].
3203
3204  003712  1 031400 1 4 3 00 00      LOG0:   DATI    0,MEM,I
3205  003713  1 031401 1 4 3 00 01              DATI    1,MEM,I
3206  003714  1 031402 1 4 3 00 02              DATI    2,MEM,I
3207  003715  1 031403 1 4 3 00 03              DATI    3,MEM,I
3208  003716  1 031404 1 4 3 00 04              DATI    4,MEM,I
3209  003717  1 031405 1 4 3 00 05              DATI    5,MEM,I
3210  003720  1 031406 1 4 3 00 06              DATI    6,MEM,I
3211  003721  1 031407 1 4 3 00 07              DATI    7,MEM,I
3212  003722  1 016000 0 7 0 000               RETURN
3213
3214  003723  1 031410 1 4 3 00 10      LOG10:  DATI    10,MEM,I
3215  003724  1 031411 1 4 3 00 11              DATI    11,MEM,I
3216  003725  1 031412 1 4 3 00 12              DATI    12,MEM,I
3217  003726  1 031413 1 4 3 00 13              DATI    13,MEM,I
3218  003727  1 031414 1 4 3 00 14              DATI    14,MEM,I
3219  003730  1 031415 1 4 3 00 15              DATI    15,MEM,I
3220  003731  1 031416 1 4 3 00 16              DATI    16,MEM,I
3221  003732  1 031417 1 4 3 00 17              DATI    17,MEM,I
3222  003733  1 016000 0 7 0 000               RETURN
3223
3224  003734  1 131400 5 4 3 00 00      LOG20:  DATI    20,MEM,I
3225  003735  1 131401 5 4 3 00 01              DATI    21,MEM,I
3226  003736  1 131402 5 4 3 00 02              DATI    22,MEM,I
3227  003737  1 131403 5 4 3 00 03              DATI    23,MEM,I
3228  003740  1 131404 5 4 3 00 04              DATI    24,MEM,I
3229  003741  1 131405 5 4 3 00 05              DATI    25,MEM,I
3230  003742  1 131406 5 4 3 00 06              DATI    26,MEM,I
3231  003743  1 131407 5 4 3 00 07              DATI    27,MEM,I
3232  003744  1 016000 0 7 0 000               RETURN
3233
3234                                    ;STACK STATUS FROM DEVICE AND DISCONNECT
3235                                    ;[8] THIS ROUTINE WAS MOVED FROM BANK 1 OF THE MEMORY TO HERE TO MAKE ROOM
3236                                    ;       FOR FIX [8]
3237
3238  003745  1 002145 0 1 0 145       STACK:  LDBR    CMDOUT+TMREN+CLKOUT+MTROUT      ;SEND COMMAND OUT
3239  003746  1 064051 3 2 0 02 11              MOVB    TORO                    ;  AND CLEAR SELECT OUT
3240  003747  1 102061 4 1 0061                 JMPI    INTRPT                  ;CHECK ON INTERRUPTS
3241  003750  1 022004 1 1 0 00 04              DATI    TAGIN0,BR               ;READ TAG IN LINES
3242  003751  1 105747 4 2 1747                 JMPB0   .-2                     ;WAIT FOR STATUS IN TO DROP
3243  003752  1 002144 0 1 0 144               LDBR    TMREN+CLKOUT+MTROUT     ;DROP COMMAND OUT
3244  003753  1 064051 3 2 0 02 11              MOVB    TORO
3245  003754  1 102061 4 1 0061                 JMPI    INTRPT                  ;CHECK ON INTERRUPTS
3246  003755  1 022004 1 1 0 00 04              DATI    TAGIN0,BR               ;READ TAG IN LINES
3247  003756  1 111754 4 4 1754                 JMPB7   .-2                     ;WAIT FOR OPERATIONAL IN TO DROP
3248  003757  1 016000 0 7 0 000               RETURN
3249
```

```
                                SUBTTL  MICRO-CODE MEMORY

003760  777777  777777          .MEM
000000  000000  000000                  0
000001  000000  000000          DFLAGS: 0                       ;DIAGNOSTICS FLAGS
000002  000000  000000          TSTADR: 0                       ;DIAGNOSTIC DISPATCH ADDRESS
000003  000000  000000          IDLCNT: 0                       ;IDLE LOOP WAIT CNT

                                ;CHECKS ON PROPER HARDWARE CONFIGURATION

000004  000000  000060          DRVTYP: TYPE&377                ;DRIVE TYPE CODE (LOW 8 BITS)
000005  000000  000120                  TYPE_-8                 ;  (HIGH BITS)




                                ;TEMPORARY STORAGE AREA

000006  000000  000000          CMND:   0                       ;STORAGE LOCATION FOR DEVICE COMMAND
```

```
                                        ;DATA LISTS FOR SETUP SEQUENCES

                                        ;WRITE COMMAND SETUP LIST

3274  000007  000000  000021  WRTSET: MBIN+DPOUT                ;I/O MASSBUS IN AND DATA PATH OUT
3275  000010  000000  000000  MCOUNT: 0                         ;CALCULATED MASSBUS COUNTER
3276  000011  000000  000000          0                       ;  FOR WRITE OPERATION
3277  000012  000000  000000          0                       ;FORMATTER START ADDRESS
3278  000013  000000  000006          SEBCOV+MEMCOV           ;FORMATTER END ENABLE FLAGS
3279  000014  000000  000003          DXHISP+BCLKEN           ;SETS HIGH SPEED
3280  000015  000000  000011          MB                      ;MASSBUS INTERFACE SELECTION
3281  000016  000000  000034          OCC+DTD+START           ;STARTS THE TRANSFER
3282  000017  000000  000033          CHN                     ;CHANNEL INTERFACE SELECTION
3283  000020  000000  000022          DP                      ;DATA PATH SELECTION
3284  000021  000000  000002          BCLKEN                  ;CLEARS HIGH SPEED
3285  000022  000000  000033          CHN                     ;CHANNEL INTERFACE SELECTION
3286  000023  000000  000011          MB                      ;MASSBUS INTERFACE SELECTION
3287  000024  000000  000031          OCC+DTD+EBL             ;SENDS EBL
3288  000025  000000  000033          CHN                     ;CHANNEL INTERFACE SELECTION
```

```
                                        ;DATA LIST FOR READ COMMAND

3291  000026  000000  000021  RDSET:  MBIN+DPOUT              ;I/O FOR MASSBUS IN AND DATA PATH OUT
3292  000027  000000  000000          0                       ;TO CLEAR MASSBUS COUNTER
3293  000030  000000  000000  FMTADR: 0                       ;FORMATTER START ADDRESS
3294  000031  000000  000010          MEONFE                  ;END ENABLE FLAGS
3295  000032  000000  000003          DXHISP+BCLKEN           ;SETS HIGH SPEED
3296  000033  000000  000011          MB                      ;MASSBUS INTERFACE SELECTION
3297  000034  000000  000024          OCC+START               ;STARTS THE TRANSFER
3298  000035  000000  000022          DP                      ;DATA PATH INTERFACE SELECTION
3299  000036  000000  000002          BCLKEN                  ;CLEARS HIGH SPEED
3300  000037  000000  000033          CHN                     ;CHANNEL INTERFACE SELECTION
3301  000040  000000  000011          MB                      ;MASSBUS INTERFACE SELECTION
3302  000041  000000  000021          OCC+EBL                 ;SENDS AN EBL
3303  000042  000000  000033          CHN                     ;CHANNEL INTERFACE SELECTION
```

```
3304                                    ;DATA LIST FOR DEVICE SELECTION ROUTINE
3305
3306  000043  000000  000033           SELSET: CHN                                     ;CHANNEL INTERFACE SELECTION
3307  000044  000000  000164                   ADROUT+TMREN+CLKOUT+MTROUT              ;SETS ADDRESS OUT
3308  000045  000000  000176                   ADROUT+HLDOUT+SELOUT+TMREN+CLKOUT+MTROUT        ;SETS SELECT OUT AND HOLD OUT
3309  000046  000000  000156                   HLDOUT+SELOUT+TMREN+CLKOUT+MTROUT       ;DROPS ADDRESS OUT
3310  000047  000000  000200                   OPLOUT                                  ;CLEARS SUPPRESS OUT
3311  000050  000000  000157                   HLDOUT+SELOUT+CMDOUT+TMREN+CLKOUT+MTROUT        ;SETS COMMAND OUT
3312  000051  000000  000156                   HLDOUT+SELOUT+TMREN+CLKOUT+MTROUT       ;DROPS COMMAND OUT
```

```
                                SUBTTL  EXTENDED STATUS TABLE

                                ;0
000052  000000  000000                  SBYT00: 0
000053  000000  000000                  SBYT01: 0
000054  000000  000000                  SBYT02: 0
000055  000000  000000                  SBYT03: 0
                                ;1
000056  000000  000000                  SBYT04: 0
000057  000000  000000                  SBYT05: 0
000060  000000  000000                  SBYT06: 0
000061  000000  000000                  SBYT07: 0
                                ;2
000062  000000  000000                  SBYT08: 0
000063  000000  000000                  SBYT09: 0
000064  000000  000000                  SBYT10: 0
000065  000000  000000                  SBYT11: 0
                                ;3
000066  000000  000000                  SBYT12: 0
000067  000000  000000                  SBYT13: 0
000070  000000  000000                  SBYT14: 0
000071  000000  000000                  SBYT15: 0
                                ;4
000072  000000  000000                  SBYT16: 0
000073  000000  000000                  SBYT17: 0
000074  000000  000000                  SBYT18: 0
000075  000000  000000                  SBYT19: 0
                                ;5
000076  000000  000000                  SBYT20: 0
000077  000000  000000                  SBYT21: 0
000100  000000  000000                  SBYT22: 0
000101  000000  000000                  SBYT23: 0
                        ;6
000102  000000  000034          MCVER:  <<VERSION_2>&374>!<<EDIT_-8>&3> ;VERSION NUMBER
000103  000000  000004                  <EDIT&377>              ;EDIT NUMBER
000104  000000  000000          STAT:   0                       ;MICRO-PROCESSOR STATUS REGISTER
000105  000000  000000          IOREG:  0                       ;I/O SELECT REGISTER
```

```
3350                                    ;7
3351  000106  000000  000000                    MBR0:   0                       ;MASSBUS REGISTERS
3352  000107  000000  000000                    MBR1:   0
3353  000110  000000  000000                    MBR2:   0
3354  000111  000000  000000                    MBR3:   0
3355                                    ;10
3356  000112  000000  000000                    MBR4:   0
3357  000113  000000  000000                    MBR5:   0
3358  000114  000000  000000                    MBR6:   0
3359  000115  000000  000000                    MBR7:   0
3360                                    ;11
3361  000116  000000  000000                    MBR10:  0
3362  000117  000000  000000                    MBR11:  0
3363  000120  000000  000000                    MBR12:  0
3364  000121  000000  000000                    MBR13:  0
3365                                    ;12
3366  000122  000000  000000                    MBR14:  0
3367  000123  000000  000000                    MBR15:  0
3368  000124  000000  000000                    MBR16:  0
3369  000125  000000  000000                    MBR17:  0
3370                                    ;13
3371  000126  000000  000000                    MBR20:  0
3372  000127  000000  000000                    MBR21:  0
3373  000130  000000  000000                    MBR22:  0
3374  000131  000000  000000                    MBR23:  0
3375                                    ;14
3376  000132  000000  000000                    MBR24:  0
3377  000133  000000  000000                    MBR25:  0
3378  000134  000000  000000                    MBR26:  0
3379  000135  000000  000000                    MBR27:  0
```

```
3380                                    ;15
3381  000136  000000  000000                  CHNR0:  0                 ;CHANNEL BUS INTERFACE REGISTERS
3382  000137  000000  000000                  CHNR1:  0
3383  000140  000000  000000                  CHNR2:  0
3384  000141  000000  000000                  CHNR3:  0
3385                                    ;16
3386  000142  000000  000000                  CHNR4:  0
3387  000143  000000  000000                  CHNR5:  0
3388  000144  000000  000000                  CHNR6:  0
3389  000145  000000  000000                  CHNR7:  0
3390                                    ;17
3391  000146  000000  000000                  DPR0:   0                 ;DATA PATH REGISTERS
3392  000147  000000  000000                  DPR1:   0
3393  000150  000000  000000                  DPR2:   0
3394  000151  000000  000000                  DPR3:   0
3395                                    ;20
3396  000152  000000  000000                  DPR4:   0
3397  000153  000000  000000                  DPR5:   0
3398  000154  000000  000000                  DPR6:   0
3399  000155  000000  000000                  DPR7:   0
3400                                    ;21
3401  000156  000000  000000                  DPR10:  0
3402  000157  000000  000000                  DPR11:  0
3403  000160  000000  000000                  DPR12:  0
3404  000161  000000  000000                  DPR13:  0
3405                                    ;22
3406  000162  000000  000000                  DPR14:  0
3407  000163  000000  000000                  DPR15:  0
3408  000164  000000  000000                  DPR16:  0
3409  000165  000000  000000                  DPR17:  0
3410                                    ;23
3411  000166  000000  000000                  LSTCMD: 0                 ;LAST COMMAND ISSUED TO DRIVE
3412  000167  000000  000000                  LSTDRV: 0                 ;LAST DRIVE SELECTED
3413  000170  000000  000000                  INTSTS: 0                 ;LAST INTERRUPT STATUS RECEIVED
3414  000171  000000  000000                  INTDRV: 0                 ;LAST INTERRUPTING DRIVE
3415
3416
3417                    000120          EXSIZE=.-SBYT00                 ;NUMBER OF BYTES IN EXTENDED STATUS TABLE
3418                    000023          MAXIDX=EXSIZE/4-1               ;MAXIMUM VALID INDEX CODE
3419
3420                                    .MNEXT                          ;MOVE TO NEXT MEMORY PAGE
3421
```

```
3422                            ;FUNCTION CODE TABLE
3423                            ;EACH ENTRY CONSISTS OF FOUR WORDS
3424                            ; FIRST IS COMMAND BYTE TO SEND TO DRIVE
3425                            ; SECOND IS CONTROL BITS TO BE PLACED IN AC7
3426                            ; THIRD IS DISPATCH ADDRESS FOR ROUTINE
3427                            ; FOURTH IS NOT USED
3428
3429                            ;INDEX INTO TABLE BY FUNCTION CODE (EXCLUDING GO) TIMES 4
3430
3431  000400  000000  000003   FUNC:     3                                   ;1      NO OPERATION
3432  000401  000000  000331                       331
3433  000402  000000  000164                       IMMCMD
3434  000403  000000  000000                       0
3435  000404  000000  000017                    17                           ;3      REWIND AND UNLOAD
3436  000405  000000  000335                       335
3437  000406  000000  000164                       IMMCMD
3438  000407  000000  000000                       0
3439  000410  777777  777777                    -1                           ;5
3440  000411  000000  000001                       1
3441  000412  000000  000173                       BADCMD
3442  000413  000000  000000                       0
3443  000414  000000  000007                     7                          ;7      REWIND
3444  000415  000000  000335                       335
3445  000416  000000  000164                       IMMCMD
3446  000417  000000  000000                       0
3447  000420  000000  000000                     0                          ;11     DRIVE CLEAR
3448  000421  000000  000001                       1
3449  000422  000000  000165                       CLRCMD
3450  000423  000000  000000                       0
3451  000424  000000  000324                  324                             ;13     SENSE RELEASE
3452  000425  000000  000001                       1
3453  000426  000000  000170                       SNSCMD
3454  000427  000000  000000                       0
3455  000430  777777  777777                   -1                           ;15
3456  000431  000000  000001                       1
3457  000432  000000  000173                       BADCMD
3458  000433  000000  000000                       0
3459  000434  777777  777777                   -1                           ;17
3460  000435  000000  000001                       1
3461  000436  000000  000173                       BADCMD
3462  000437  000000  000000                       0
3463  000440  777777  777777                   -1                           ;21
3464  000441  000000  000001                       1
3465  000442  000000  000173                       BADCMD
3466  000443  000000  000000                       0
3467  000444  777777  777777                   -1                           ;23
3468  000445  000000  000001                       1
3469  000446  000000  000173                       BADCMD
3470  000447  000000  000000                       0
```

```
3471  000450  000000  000027          27                          ;25     ERASE GAP
3472  000451  000000  000011           11
3473  000452  000000  000164          IMMCMD
3474  000453  000000  000000          0
3475  000454  000000  000037          37                          ;27     WRITE TAPE MARK
3476  000455  000000  000011           11
3477  000456  000000  000164          IMMCMD
3478  000457  000000  000000          0
3479  000460  000000  000067          67                          ;31     FORWARD SPACE RECORD
3480  000461  000000  000011           11
3481  000462  000000  000164          IMMCMD
3482  000463  000000  000000          0
3483  000464  000000  000047          47                          ;33     BACK SPACE RECORD
3484  000465  000000  000011           11
3485  000466  000000  000164          IMMCMD
3486  000467  000000  000000          0
3487  000470  000000  000077          77                          ;35     FORWARD SPACE FILE
3488  000471  000000  000011           11
3489  000472  000000  000164          IMMCMD
3490  000473  000000  000000          0
3491  000474  000000  000057          57                          ;37     BACKWARD SPACE FILE
3492  000475  000000  000011           11
3493  000476  000000  000164          IMMCMD
3494  000477  000000  000000          0
3495  000500  000000  000000           0                          ;41     TEST I/O
3496  000501  000000  000331          331
3497  000502  000000  000164          IMMCMD
3498  000503  000000  000000          0
3499  000504  000000  000227          227                         ;43     DATA SECURITY ERASE
3500  000505  000000  000021           21
3501  000506  000000  000164          IMMCMD
3502  000507  000000  000000          0
3503  000510  000000  000004           4                          ;45     SENSE
3504  000511  000000  000001          1
3505  000512  000000  000170          SNSCMD
3506  000513  000000  000000          0
3507  000514  000000  000364          364                         ;47     SENSE RESERVE
3508  000515  000000  000001          1
3509  000516  000000  000170          SNSCMD
3510  000517  000000  000000          0
3511  000520  777777  777777          -1                          ;51
3512  000521  000000  000001          1
3513  000522  000000  000174          BADXFR
3514  000523  000000  000000          0
3515  000524  777777  777777          -1                          ;53
3516  000525  000000  000001          1
3517  000526  000000  000174          BADXFR
3518  000527  000000  000000          0
```

```
3519  000530  777777  777777          -1                              ;55
3520  000531  000000  000001                  1
3521  000532  000000  000174                  BADXFR
3522  000533  000000  000000                  0
3523  000534  777777  777777          -1                              ;57
3524  000535  000000  000001                  1
3525  000536  000000  000174                  BADXFR
3526  000537  000000  000000                  0
3527  000540  000000  000001           1                              ;61     WRITE DATA
3528  000541  000000  000101                  101
3529  000542  000000  000166                  WRTCMD
3530  000543  000000  000000                  0
3531  000544  000000  000013          13                              ;63     WRITE DIAGNOSTIC
3532  000545  000000  000001                  1
3533  000546  000000  000166                  WRTCMD
3534  000547  000000  000000                  0
3535  000550  000000  000213         213                              ;65     LOOP WRITE TO READ
3536  000551  000000  000101                  101
3537  000552  000000  000166                  WRTCMD
3538  000553  000000  000000                  0
3539  000554  777777  777777          -1                              ;67
3540  000555  000000  000001                  1
3541  000556  000000  000174                  BADXFR
3542  000557  000000  000000                  0
3543  000560  000000  000002           2                              ;71     READ DATA
3544  000561  000000  000000                  0
3545  000562  000000  000167                  RDCMD
3546  000563  000000  000000                  0
3547  000564  777777  777777          -1                              ;73
3548  000565  000000  000001                  1
3549  000566  000000  000174                  BADXFR
3550  000567  000000  000000                  0
3551  000570  000000  000000           0                              ;75     DUMP STATUS
3552  000571  000000  000001                  1
3553  000572  000000  000171                  DMPCMD
3554  000573  000000  000000                  0
3555  000574  000000  000014          14                              ;77     READ REVERSE
3556  000575  000000  000100                  100
3557  000576  000000  000167                  RDCMD
3558  000577  000000  000000                  0
```

```
3559                                    ;DRIVE MODE COMMANDS
3560                                    ;THIS TABLE CONTAINS THE COMMAND BYTE TO SEND FOR EACH
3561                                    ;DRIVE DENSITY SELECTED
3562                                    ;INDEX BY CODE RECEIVED
3563                                    ;EDIT [1] CORRECTS CODE FOR 7 TRK, 556 BPI
3564  000600  000000  000000           DRVMOD:   0                    ;NO MODE SET TO BE ISSUED
3565  000601  000000  000063                    63                    ;7 TRACK, 200 BPI, ODD PARITY
3566                                    ;[1]     153                   ;7 TRACK, 556 BPI, ODD PARITY
3567  000602  000000  000163                    163                   ;[1]7 TRACK,556 BPI, ODD PARITY
3568  000603  000000  000263                    263                   ;7 TRACK, 800 BPI, ODD PARITY
3569  000604  777777  777777                     -1                   ;ILLEGAL
3570  000605  000000  000043                     43                   ;7 TRACK, 200 BPI, EVEN PARITY
3571  000606  000000  000143                    143                   ;7 TRACK, 556 BPI, EVEN PARITY
3572  000607  000000  000243                    243                   ;7 TRACK, 800 BPI, EVEN PARITY
3573  000610  777777  777777                     -1                   ;ILLEGAL
3574  000611  777777  777777                     -1                   ;ILLEGAL
3575  000612  777777  777777                     -1                   ;ILLEGAL
3576  000613  000000  000313                    313                   ;9 TRACK, 800 BPI
3577  000614  000000  000303                    303                   ;9 TRACK, 1600 BPI
3578  000615  000000  000323                    323                   ;9 TRACK, 6250 BPI
3579  000616  777777  777777                     -1                   ;ILLEGAL
3580  000617  777777  777777                     -1                   ;ILLEGAL
3581
3582                                    ;DATA PACKING MODES
3583                                    ;THIS TABLE CONTAINS THE NUMBER OF BYTES PER WORD
3584                                    ;FOR EACH DATA MODE
3585
3586                                    ;INDEX BY DATA MODE CODE
3587
3588  000620  777777  777777           DATMOD: -1                     ;ILLEGAL
3589  000621  000000  000005                     5                    ;CORE DUMP
3590  000622  000000  000004                     4                    ;INDUSTRY COMPATIBLE
3591  000623  000000  000006                     6                    ;SIXBIT
3592  000624  000000  000005                     5                    ;ASCII
3593  000625  000000  000011                     9                    ;HIGH DENSITY (BYTES/2 WORDS)
3594  000626  777777  777777                    -1                    ;ILLEGAL
3595  000627  777777  777777                    -1                    ;ILLEGAL
```

```
                                ;FORMATTER START ADDRESSES

000630  000000  000030  FMTWRT:  30          ;CORE DUMP WRITE START ADDRESS
000631  000000  000000            0          ;BYTE MODE
000632  000000  000140          140          ;SIXBIT
000633  000000  000230          230          ;ASCII
000634  000000  000060           60          ;HI-DENSITY

000635  000000  000040  FMTRD:   40          ;CORE DUMP READ FORWARD START ADDRESS
000636  000000  000010           10          ;BYTE MODE
000637  000000  000150          150          ;SIXBIT
000640  000000  000240          240          ;ASCII
000641  000000  000100          100          ;HI-DENSITY

000642  000000  000647  FMTRDB: RDB1         ;TABLE ADDRESS OF READ BACKWARD START ADDRESSES
                                             ;CORE DUMP
000643  000000  000654          RDB2         ;BYTE MODE
000644  000000  000660          RDB3         ;SIXBIT
000645  000000  000666          RDB4         ;ASCII
000646  000000  000673          RDB5         ;HI-DENSITY

000647  000000  000050  RDB1:    50          ;CORE DUMP READ BACKWARD, FULL WORDS
000650  000000  000057           57          ;1 EXTRA BYTE
000651  000000  000053           53          ;2 EXTRA BYTES
000652  000000  000052           52          ;3 EXTRA BYTES
000653  000000  000051           51          ;4 EXTRA BYTES

000654  000000  000020  RDB2:    20          ;BYTE MODE READ BACKWARD, FULL WARDS
000655  000000  000027           27          ;1 EXTRA BYTE
000656  000000  000022           22          ;2 EXTRA BYTES
000657  000000  000021           21          ;3 EXTRA BYTES

000660  000000  000160  RDB3:   160          ;SIXBIT MODE READ BACKWARD, FULL WORDS
000661  000000  000177          177          ;1 EXTRA BYTE
000662  000000  000176          176          ;2 EXTRA BYTES
000663  000000  000163          163          ;3 EXTRA BYTES
000664  000000  000162          162          ;4 EXTRA BYTES
000665  000000  000161          161          ;5 EXTRA BYTES

000666  000000  000250  RDB4:   250          ;ASCII MODE READ BACKWARD, FULL WORDS
000667  000000  000257          257          ;1 EXTRA BYTE
000670  000000  000253          253          ;2 EXTRA BYTES
000671  000000  000252          252          ;3 EXTRA BYTES
000672  000000  000251          251          ;4 EXTRA BYTES

000673  000000  000120  RDB5:   120          ;HI-DENSITY READ BACKWARD, FULL WORDS
000674  000000  000137          137          ;1 EXTRA BYTE
000675  000000  000131          131          ;2 EXTRA BYTES
000676  000000  000130          130          ;3 EXTRA BYTES
000677  000000  000127          127          ;4 EXTRA BYTES
000700  000000  000136          136          ;5 EXTRA BYTES
000701  000000  000123          123          ;6 EXTRA BYTES
000702  000000  000122          122          ;7 EXTRA BYTES
000703  000000  000121          121          ;8 EXTRA BYTES
```

```
  3651                                  .END    DXASZ
  3652
  3653                                          END

NO ERRORS DETECTED

PROGRAM BREAK IS 000000
ABSOLUTE BREAK IS 004665
CPU TIME USED 02:44.723

36P CORE USED
```

ADB	567	636	709	828	842	880	999	1013	1022	1030				
ADBCR	2385													
ADBR	2370													
ADC	767	2337												
ADCR	854	867	1905	2341	2947									
ADM	1032													
DATA	480	481	482											
DATI	529	549	559	562	572	587	591	601	616	632	679	701	705	743
	763	766	822	891	898	915	934	935	950	964	971	993	1041	1048
	1061	1067	1078	1083	1106	1107	1127	1135	1139	1160	1164	1171	1172	1185
	1204	1261	1264	1300	1349	1371	1374	1387	1400	1405	1418	1423	1446	1455
	1459	1468	1504	1528	1553	1557	1564	1573	1577	1581	1586	1590	1612	1653
	1655	1658	1662	1670	1676	1763	1775	1816	1817	1861	1889	1902	1927	1928
	1962	1964	1980	1985	2022	2026	2487	2494	2504	2512	2523	2532	2544	2545
	2558	2569	2576	2613	2626	2634	2641	2655	2669	2682	2701	2739	2752	2760
	2768	2782	2796	2805	2809	2820	2834	2843	2844	2855	2872	2884	2897	2916
	2941	2944	3194	3204	3205	3206	3207	3208	3209	3210	3211	3214	3215	3216
	3217	3218	3219	3220	3221	3224	3225	3226	3227	3228	3229	3230	3231	3241
	3246													
DEC	539	588	851	860	1191	1194	1613	2291	2298	2305	2312	2319	2326	2596
	2722													
DECR	865	866	1143	1146	1149	1791	1878	2098	2197	2220	2223	2615	2628	2658
	2671	2685	2704	2741	2754	2785	2798	2823	2836	2858	2868	2874	2886	2900
	2919	2958	3171	3174										
GOSUB	849	893	1026	1043	1157	1618	1820	1822	1824	1828	1834	1836	1909	2092
INC	764	2069	2236	2244	2252	2260	2268	2276	2503	2511	3159			
INCR	853	1904	2184	2210	2216	2336	2340	2383	2957	3039	3040	3041	3042	3043
	3044	3045	3046	3047	3048	3049	3050	3051	3052	3053	3054	3055	3056	3057
	3058	3059	3060	3061	3062	3063	3064	3065	3066	3067	3068	3069	3070	3071
	3072	3073	3074	3075	3076	3077	3078	3079	3080	3081	3082	3083	3084	3085
	3086	3087	3088	3090	3091	3092	3093	3094	3095	3096	3097	3098	3099	3100
	3101	3102	3103	3104	3105	3106	3107	3108	3109	3110	3111	3112	3113	3114
	3115	3116	3117	3118	3119	3120	3121	3122	3123	3124				
JMP	470	471	472	473	474	475	501	503	509	515	541	543	583	604
	640	646	647	648	649	650	652	653	654	664	683	715	773	800
	804	857	901	952	970	973	976	1023	1051	1064	1080	1138	1142	1145
	1148	1151	1154	1166	1193	1196	1203	1225	1229	1235	1247	1251	1287	1304
	1315	1319	1323	1327	1333	1361	1373	1451	1457	1463	1471	1486	1519	1531
	1536	1575	1588	1600	1657	1666	1672	1701	1705	1709	1724	1727	1739	1744
	1749	1767	1779	1845	1894	1907	1918	1944	1967	1974	1989	1996	2003	2010
	2025	2029	2070	2104	2112	2124	2131	2138	2143	2148	2160	2171	2176	2188
	2203	2212	2215	2222	2240	2248	2256	2264	2272	2280	2287	2294	2301	2308
	2315	2322	2329	2339	2345	2356	2362	2372	2376	2389	2402	2411	2470	2476
	2480	2497	2514	2526	2535	2548	2561	2581	2617	2630	2638	2657	2660	2673
	2684	2687	2706	2743	2756	2764	2784	2787	2800	2808	2822	2825	2838	2857
	2860	2876	2877	2888	2899	2902	2921	2955	2966	2967	2968	2969	2970	2971
	2972	2973	2974	2975	2976	2977	2978	2979	2980	2981	2982	2983	2984	2985
	2986	2987	2988	2989	2997	3001	3005	3006	3007	3008	3009	3010	3011	3012
	3013	3014	3015	3016	3017	3018	3019	3020	3021	3022	3023	3024	3025	3027
	3033	3136	3162	3173	3176									
JMPB0	531	536	595	618	680	714	791	892	900	951	972	1042	1050	1063
	1205	1216	1245	1301	1372	1388	1401	1419	1448	1450	1556	1587	1593	1595
	1649	1654	1687	1776	1862	1890	1903	1929	1931	1933	1935	1963	2067	2137

	2140	2643	2670	2683	2770	2797	2811	2821	2846	2885	2898	3157	3242	
JMPB4	603	727	760	856	966	1137	1162	1187	1456	1469	1530	1559	1574	1582
	1598	1722	1764	1981	1984	1988	2142	2145	2489	2496	2505	2513	2627	2656
	2753	2783	2995											
JMPB7	550	561	592	703	750	835	917	1006	1020	1079	1084	1165	1406	1424
	1447	1505	1554	1565	1656	1663	1671	1677	2147	2150	2614	2740	2835	2856
	2873	3135	3247											
JMPC	771	864	1792	1879	1906	2099	2211	2214	2217	2219	2221	2224	2371	2954
	2959													
JMPI	532	639	720	731	735	747	753	797	903	919	928	949	955	958
	1053	1069	1088	1097	1134	1159	1163	1173	1179	1222	1275	1312	1353	1370
	1386	1399	1404	1417	1422	1445	1454	1460	1467	1527	1552	1572	1580	1585
	1652	1661	1669	1675	2101	3240	3245							
JMPSUB	527	528	606	658	659	666	667	674	700	719	722	730	733	734
	737	746	751	752	772	792	802	803	850	887	894	902	905	918
	921	927	930	957	975	1027	1037	1044	1052	1055	1068	1075	1087	1090
	1096	1099	1158	1178	1209	1217	1227	1228	1234	1239	1240	1250	1271	1272
	1273	1274	1281	1282	1289	1292	1308	1309	1310	1311	1314	1317	1318	1321
	1322	1325	1326	1329	1332	1348	1352	1355	1360	1475	1492	1512	1532	1603
	1619	1628	1631	1634	1646	1650	1668	1680	1681	1690	1692	1699	1703	1707
	1713	1723	1726	1729	1730	1737	1738	1741	1742	1743	1746	1747	1748	1821
	1823	1825	1829	1835	1837	1863	1910	2063	2093	2464	2600	3153		
JMPZ	540	589	607	711	721	732	736	748	754	830	844	852	861	904
	920	929	959	1001	1015	1054	1070	1073	1089	1098	1141	1144	1147	1150
	1180	1192	1195	1276	1313	1354	1378	1462	1614	1647	2024	2028	2109	2111
	2123	2127	2130	2159	2163	2170	2175	2187	2199	2202	2239	2247	2255	2263
	2271	2279	2286	2293	2300	2307	2314	2321	2328	2338	2344	2355	2361	2375
	2388	2401	2410	2469	2475	2525	2534	2547	2560	2580	2597	2616	2629	2637
	2659	2672	2686	2703	2705	2723	2742	2755	2763	2786	2799	2807	2824	2837
	2859	2869	2875	2887	2901	2918	2920	3172	3175					
JUMP	514	542	651	1599	1844	1893	1917	1973	1995	2002	2009	2103	3026	3032
	3161													
LANDB	574	779	824	995	1376									
LANDBR	634	707	2578											
LDBR	500	502	508	510	514	522	524	542	557	566	573	581	599	608
	614	619	628	633	635	651	660	662	668	670	672	675	677	681
	706	708	728	741	761	769	778	785	793	795	798	817	823	827
	841	849	862	868	879	893	911	922	988	994	998	1012	1021	1026
	1029	1043	1059	1065	1071	1081	1085	1128	1130	1132	1152	1157	1210	1218
	1220	1223	1246	1248	1257	1277	1283	1285	1293	1295	1298	1302	1330	1335
	1337	1356	1375	1384	1389	1395	1397	1402	1407	1413	1415	1420	1476	1478
	1480	1482	1484	1490	1495	1498	1500	1502	1506	1508	1510	1513	1517	1523
	1525	1550	1566	1578	1583	1599	1606	1610	1615	1618	1626	1632	1644	1659
	1664	1673	1678	1688	1691	1700	1704	1708	1714	1716	1731	1735	1759	1761
	1765	1769	1771	1773	1777	1788	1793	1799	1803	1818	1820	1822	1824	1826
	1828	1830	1832	1834	1836	1844	1856	1867	1869	1873	1884	1886	1891	1893
	1898	1900	1909	1911	1913	1915	1917	1940	1942	1946	1948	1950	1952	1954
	1960	1971	1973	1978	1993	1995	2000	2002	2007	2009	2016	2018	2033	2049
	2051	2058	2074	2076	2078	2080	2085	2087	2092	2095	2103	2110	2120	2125
	2136	2139	2141	2144	2146	2149	2155	2185	2195	2208	2213	2218	2231	2334
	2342	2350	2369	2381	2384	2386	2396	2462	2467	2473	2477	2486	2493	2501
	2509	2521	2530	2540	2542	2554	2556	2567	2571	2574	2598	2601	2603	2605
	2607	2609	2611	2622	2624	2649	2651	2653	2665	2667	2678	2680	2692	2694

```
**END** Job DX20CA Req #100 for DEUFEL.TL    Date  3-Mar-82 14:49:28 Monitor: 2102 TOPS-20 Development System, TOPS-20 Moni **END**
```



```
**END** Job DX20CA Req #100 for DEUFEL.TL    Date  3-Mar-82 14:49:28 Monitor: 2102 TOPS-20 Development System, TOPS-20 Moni **END**
```



```
                    * * * L P T S P L   R u n   L o g * * *

14:29:43 LPDAT  LPTSPL version 104(2744)        2102 TOPS-20 Development System, TOPS-20 Moni
14:29:43 LPDAT  Job DX20CA sequence #183 on Printer 1 at  3-Mar-82 14:29:43
14:29:46 LPMSG  Starting File KNET:<DEUFEL.TL>DX20CA.LST.1
14:34:02 LPMSG  Finished File KNET:<DEUFEL.TL>DX20CA.LST.1
14:34:03 LPMSG  Starting File KNET:<DEUFEL.TL>DXMCA.LST.1
14:49:28 LPMSG  Finished File KNET:<DEUFEL.TL>DXMCA.LST.1
14:49:28 LPEND  Summary:  304 Pages of Output
14:49:28 LPEND            113 Disk Pages Read
14:49:28 LPEND          8.596 Seconds CPU Time Used
```



```
**END** Job DX20CA Req #100 for DEUFEL.TL    Date  3-Mar-82 14:49:28 Monitor: 2102 TOPS-20 Development System, TOPS-20 Moni **END**

**END** Job DX20CA Req #100 for DEUFEL.TL    Date  3-Mar-82 14:49:28 Monitor: 2102 TOPS-20 Development System, TOPS-20 Moni **END**

**END** Job DX20CA Req #100 for DEUFEL.TL    Date  3-Mar-82 14:49:28 Monitor: 2102 TOPS-20 Development System, TOPS-20 Moni **END**

**END** Job DX20CA Req #100 for DEUFEL.TL    Date  3-Mar-82 14:49:28 Monitor: 2102 TOPS-20 Development System, TOPS-20 Moni **END**

**END** Job DX20CA Req #100 for DEUFEL.TL    Date  3-Mar-82 14:49:28 Monitor: 2102 TOPS-20 Development System, TOPS-20 Moni **END**

**END** Job DX20CA Req #100 for DEUFEL.TL    Date  3-Mar-82 14:49:28 Monitor: 2102 TOPS-20 Development System, TOPS-20 Moni **END**

**END** Job DX20CA Req #100 for DEUFEL.TL    Date  3-Mar-82 14:49:28 Monitor: 2102 TOPS-20 Development System, TOPS-20 Moni **END**

**END** Job DX20CA Req #100 for DEUFEL.TL    Date  3-Mar-82 14:49:28 Monitor: 2102 TOPS-20 Development System, TOPS-20 Moni **END**
```



```
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000011111111111111111111111111111111
```